

IOS Ders Planı

Kasım ADALAN

Elektronik ve Haberleşme Mühendisi

Senior Android – IOS Developer and Trainer

Swift

Swift Giriş

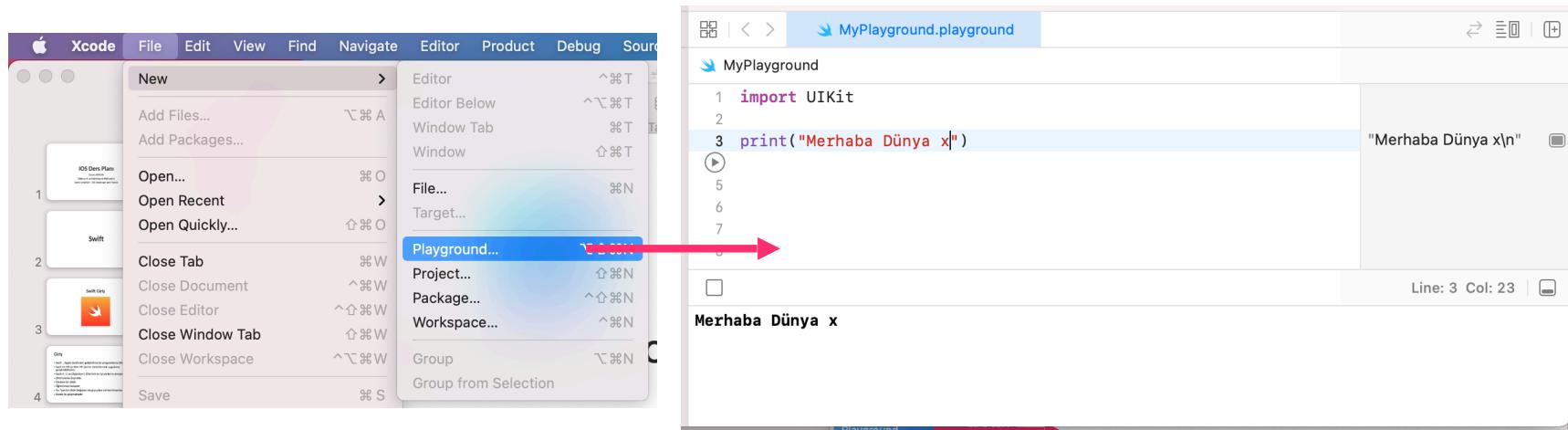


Giriş

- Swift , Apple tarafından geliştirilmiş bir programlama dilidir.
- Swift ile IOS ve Mac OS işletim sistemlerinde uygulama geliştirebilirsiniz.
- Swift 4 , C ve Objective-C dillerinin en iyi yönlerini almıştır.
- 2010 yılında duyuldu.
- Modern bir dildir.
- Öğrenilmesi kolaydır.
- No Type bir dildir.Değişken oluştururken tür belirtmenize gerek yoktur.
- Xcode ile çalışmaktadır.

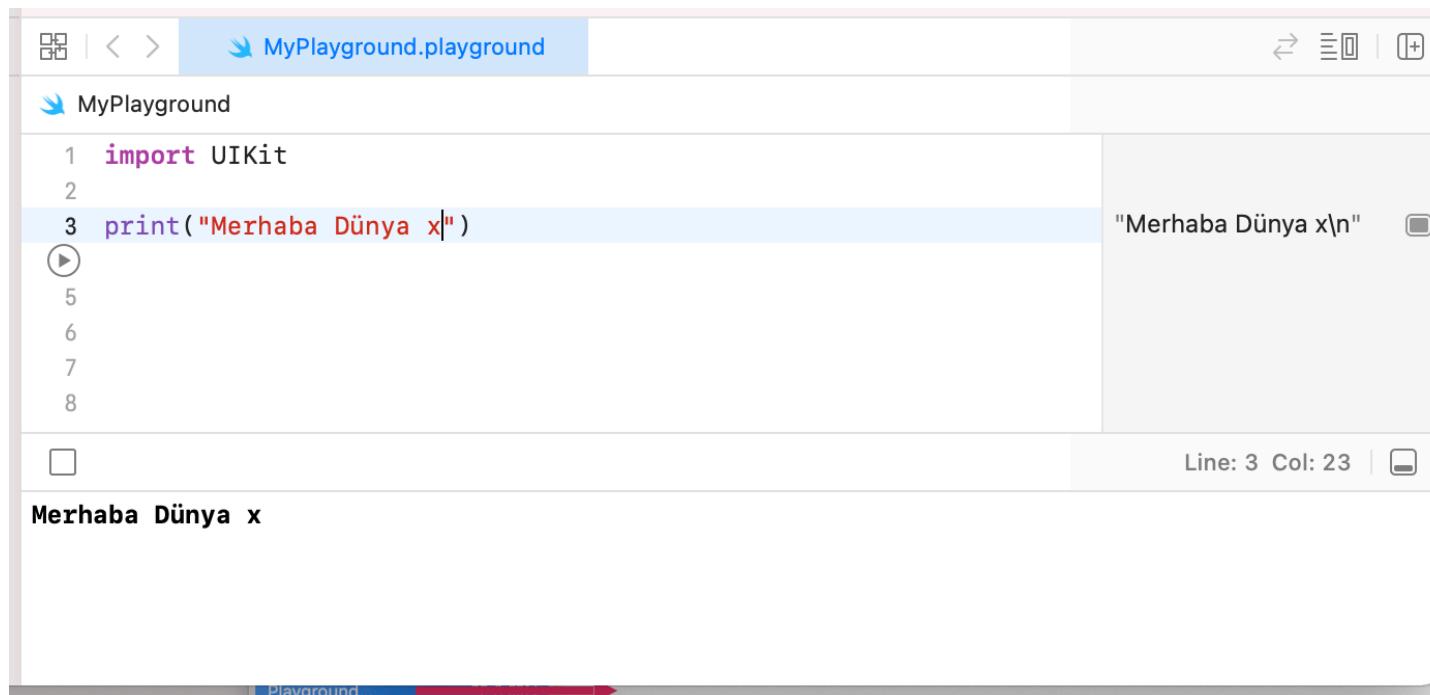
Playground - İlk Kodlama

- Xcode üzerinde Playground açarak sadece swift diline odaklanmak mümkündür.
- Bundan dolayı swift öğrenme aşamasında Playground kullanıyoruz.



Merhaba Dünya

- Kodlamaya başlama klasiğimizi yapmadan olmaz 😊



The screenshot shows a Xcode playground window titled "MyPlayground.playground". The code editor contains the following Swift code:

```
1 import UIKit
2
3 print("Merhaba Dünya x")
```

The third line, "print("Merhaba Dünya x")", is highlighted in blue. To the right of the editor, the output pane displays the result of the print statement: "Merhaba Dünya x\n". Below the editor, the playground output pane shows the text "Merhaba Dünya x". The status bar at the bottom indicates "Line: 3 Col: 23".

Değişkenler

Değişken Oluşturma

```
var ogrenciAdi = "Ahmet"
var ogrenciYas = 18
var ogrenciBoy = 1.78
var ogrenciBasharf = "A"
var ogrenciDevamEdiyorMu = true

print(ogrenciAdi)
print(ogrenciYas)
print(ogrenciBoy)
print(ogrenciBasharf)
print(ogrenciDevamEdiyorMu)

var urun_id:Int = 3416
var urun_adi:String = "Macbook Pro"
var urun_adet:Int = 100
var urun_fiyat:Int = 34999
var urun_tedarikci:String = "Apple"

print("Ürün id : \$(urun_id)")
print("Ürün adı : \$(urun_adi)")
print("Ürün adet : \$(urun_adet)")
print("Ürün fiyat : \$(urun_fiyat) ₺")//option + t
print("Ürün tedarikçi : \$(urun_tedarikci)")

print(urun_id,urun_adi)
```

Constant - Sabitler

```
var sayi = 10
```

```
print(sayi)
```

```
sayi = 20
```

```
print(sayi)
```

```
let numara = 100
```

```
print(numara)
```

Tür Dönüşümü

```
//Sayısalдан sayısala
var i = 42
var d = 56.78

var sonuc1 = Double(i)
var sonuc2 = Int(d)          //Metinden sayısala
                            var yazi = "34"
print(sonuc1)
print(sonuc2)
                            if let sonuc5 = Int(yazi) {
                                print(sonuc5)
}
//Sayısalдан Metine
var sonuc3 = String(i)
var sonuc4 = String(d)

print(sonuc3)
print(sonuc4)
```

Standart Programlama Yapıları

Karşılaştırma Operatörleri

```
//Karşılaştırma operatörleri
var a = 40
var b = 50

var x = 90
var y = 80

print("a == b : \\"(a == b)"")
print("a != b : \\"(a != b)"")
print("a > b : \\"(a > b)"")
print("a >= b : \\"(a >= b)"")
print("a < b : \\"(a < b)"")
print("a <= b : \\"(a <= b)"")

print("a > b || x > y : \\"(a > b || x > y)"")//||=OR(VEYA), false - false : false olur , diğer durumlarda
    hep true
print("a > b && x > y : \\"(a > b && x > y)"")//&&=AND(VE), true - true : true olur , diğer durumlarda hep
    false
```

İf çalışma yapısı

```
var yas = 17
var isim = "Mehmet"

if yas >= 18 {
    print("Reşitsiniz")
}else{
    print("Reşit değilsiniz")
}

if isim == "Ahmet" {
    print("Merhaba Ahmet")
}else if isim == "Mehmet" {
    print("Merhaba Mehmet")
}else{
    print("Tanınmayan kişi")
}

var ka = "admin"
var s = 12345

if ka == "admin" && s == 12345 {
    print("Hoşgeldiniz")
}else{
    print("Hatalı giriş")
}

var sonuc = 9

if sonuc == 9 || sonuc == 10 {
    print("Sonuç 9 veya 10 dur")
}else{
    print("Sonuç 9 veya 10 değildir")
}

var z = 10
var t = 20

z == t ? print("Eşit") : print("Eşit değil")
```

Switch Yapısı

```
var gun = 9

switch gun {
    case 1:
        print("Pazartesi")
    case 2 :
        print("Salı")
    case 3 :
        print("Çarşamba")
    case 4 :
        print("Perşembe")
    case 5 :
        print("Cumar")
    case 6 :
        print("Cumartesi")
    case 7 :
        print("Pazar")
    default:
        print("Böyle bir gün yok")
}
```

Döngüler

```
//1,2,3
for i in 1...3 {
    print("Döngü 1 : \$(i)")
}

//10 ile 20 , 5 er artış
for i in stride(from: 10, through: 20, by: 5) {
    print("Döngü 2 : \$(i)")
}

//20 ile 10 , 5 er azalış
for i in stride(from: 20, through: 10, by: -5) {
    print("Döngü 3 : \$(i)")
}

//1,2,3
var sayac = 1

while sayac < 4 {
    print("Döngü 4 : \$(sayac)")
    sayac+=1//sayac = sayac + 1
}

for i in 1...5 {
    if i == 3 {
        break
    }
    print("Döngü 5 : \$(i)")

    for i in 1...5 {
        if i == 3 {
            continue
        }
        print("Döngü 6 : \$(i)")
    }
}
```

Optionals

Optional ?

- Optional değişken değer içerebilir veya nil yani değer yoktur.
- ? ile temsil edilir.
- ! işaretini ile optional değişkenin nil gelmeyeceğini garanti edersin.
- Sonucun nasıl geleceğini bilmediğimiz durumlarda optional binding yapılır (if let)
- Kullanım amacı uygulamanın çökmesini engellemektir. Büyük oranda uygulamalar nil olan değişkenler ile çöker.

```
var str:String? = nil    nil durum

if str != nil {
    print(str)
}else{
    print("str nil değer içeriyor")
}
```

str nil değer içeriyor

```
var str:String? değer aktarılmamış
                    durum

if str != nil {
    print(str)
}else{
    print("str nil değer içeriyor")
}
```

Optional Unwrapping

(! işaretini ile optional olmaktan çıkışma)

```
var str:String?
```

```
str = "Merhaba"
```

```
if str != nil {  
    print(str)  
} else {  
    print("str nil değer içeriyor")  
}
```

```
var str:String?
```

```
str = "Merhaba"
```

```
if str != nil {  
    print(str!)  
} else {  
    print("str nil değer içeriyor")  
}
```

Optional("Merhaba")

Merhaba

Optional Binding - if let

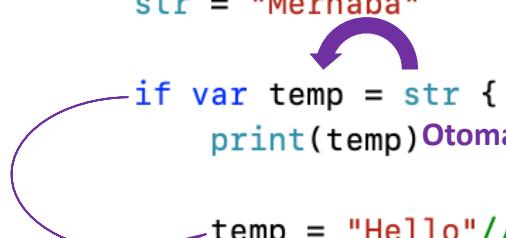
- Bu yapı kullanılabilecek değişkenin veri içerip içermediğinin kontrolü yapılır.
- Eğer nil değilse veya içerisine değer atanmış ise çalışır.
- ! işaretini koymadan otomatik olarak unwrap olur.
- Sonucun nasıl geleceğini bilmediğimiz durumlarda optional binding yapılır.

```
var str:String?  
  
str = "Merhaba"  
  
if let temp = str {  
    print(temp) Otomatik unwrapping  
} else {  
    print("str nil değer içeriyor")  
}
```

Merhaba

Optional Binding - if var

- let kullanıldığından değişkenin içeriği değiştirilemez böyle durumlarda var kullanılır.

```
var str:String?  
  
str = "Merhaba"  
  
if var temp = str {  
    print(temp) Otomatik unwrapping  
  
    temp = "Hello" //değişkenin değeri değiştirildi  
                //bundan dolayı var olmalıdır.  
    print(temp) Otomatik unwrapping  
}  
else{  
    print("str nil değer içeriyor")  
}
```

Merhaba
Hello

Nesne Tabanlı Programlama

Araba Analojisi

```
class Araba{
    var renk:String?
    var hiz:Int?
    var calisiyorMu:Bool?

    init(){
        print("Araba sınıfından boş constructor ile nesne oluşturuldu")
    }

    init(renk:String,hiz:Int,calisiyorMu:Bool){//Shadowing
        self.renk = renk
        self.hiz = hiz
        self.calisiyorMu = calisiyorMu
        print("Araba sınıfından dolu constructor ile nesne oluşturuldu")
    }

    func calistir(){
        calisiyorMu = true
        hiz = 5
    }

    func durdur(){
        calisiyorMu = false
        hiz = 0
    }
}

func hizlan(kacKm:Int){
    hiz!+=kacKm
}

func yavasla(kacKm:Int){
    hiz!-=kacKm
}

func bilgiAl(){
    print("*****")
    print("Renk : \(\(renk!)")")
    print("Hiz : \(\(hiz!)")")
    print("Çalışıyor mu : \(\(calisiyorMu!)")")
}
```

Araba Analojisi

```
var bmw = Araba(renk: "Kirmizi", hiz: 100, calisiyorMu: true)

//Değer Atama
//bmw.renk = "Kirmizi"
//bmw.hiz = 100
//bmw.calisiyorMu = true

//Değer okuma
bmw.bilgiAl()
bmw.durdur()
bmw.bilgiAl()

var limuzin = Araba(renk: "Beyaz", hiz: 0, calisiyorMu: false)

//limuzin.renk = "Beyaz"
//limuzin.hiz = 0
//limuzin.calisiyorMu = false

limuzin.bilgiAl()
limuzin.calistir()
limuzin.bilgiAl()

limuzin.hizlan(kacKm: 100)
limuzin.bilgiAl()
limuzin.yavasla(kacKm: 53)
limuzin.bilgiAl()
```

Initialization - Constructor

- Bir sınıfından (class veya structure) nesne oluşturmak için gerekli olan yapıdır.
- `init` kelimesi ile tanımlanır.
- Bir sınıfından (class veya structure) nesne oluşturma işleminde parametre alabilir.

```
class Kisiler {  
    var ad:String?  
    var yas:Int?  
  
    init() {  
    }  
    init(ad:String,yas:Int) {  
        self.ad = ad  
        self.yas = yas  
    }  
}
```

`var kisi1 = Kisiler()`
`kisi1.ad = "Ahmet"`
`kisi1.yas = 18`

`var kisi2 = Kisiler(ad:"Mehmet",yas:20)`

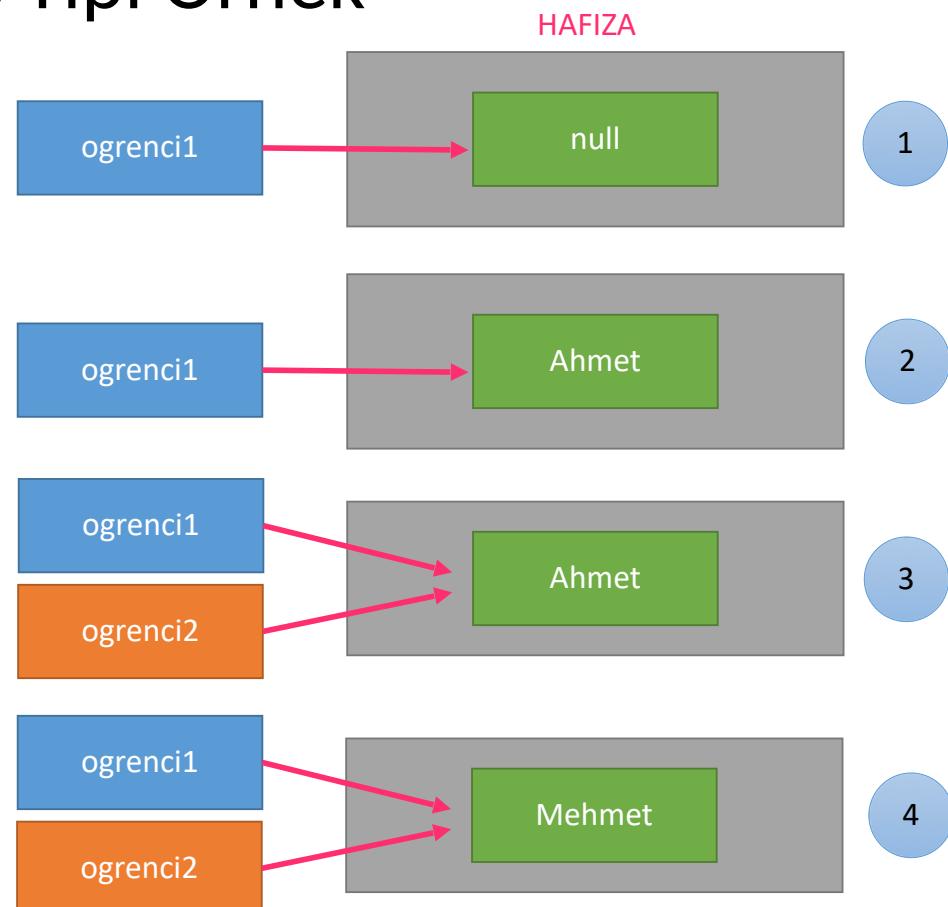
Class vs Structure Farkı

Referans Tipi Örnek

- Hafıza yönetiminde iki nesneninde aynı yeri işaret etmesidir.

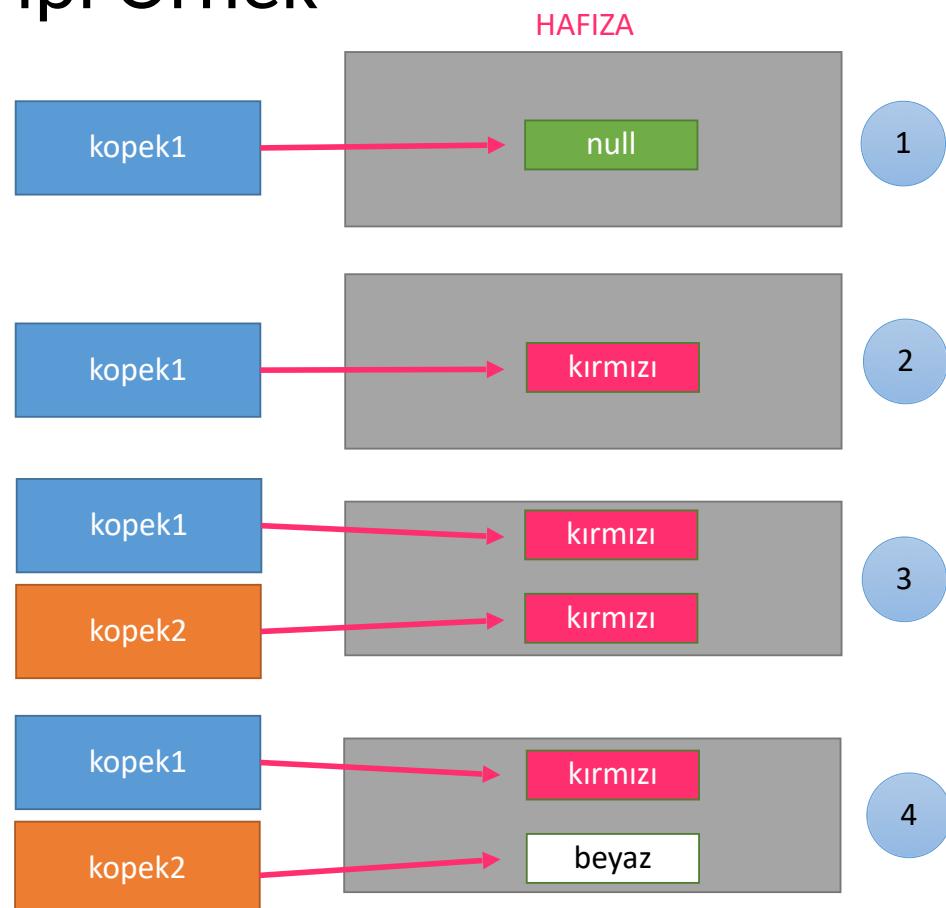
```
class Ogrenci {  
    var ad:String?  
}
```

```
1 var ogrenci1 = Ogrenci()  
2 ogrenci1.ad = "Ahmet"  
  
3 var ogrenci2 = ogrenci1  
4 ogrenci2.ad = "Mehmet"  
  
print(ogrenci1.ad!) //Mehmet
```



Değer Tipi Örnek

```
struct Kopek {  
    var renk:String?  
}  
  
1 var kopek1 = Kopek()  
2 kopek1.renk = "kırmızı"  
  
3 var kopek2 = kopek1  
4 kopek2.renk = "beyaz"  
  
print(kopek1.renk!) //kırmızı
```



Fonksiyonlar

```
class Foksiyonlar {
    //Geri dönüş değeri olmayan ( void )
    func selamla(){
        let sonuc = "Merhaba Ahmet"
        print(sonuc)
    }

    //Geri dönüş değeri olan ( return )
    func selamla1() -> String {
        let sonuc = "Merhaba Ahmet"
        return sonuc
    }

    //Parametre kullanımı
    func selamla2(isim:String){
        let sonuc = "Merhaba \(isim)"
        print(sonuc)
    }

    func toplama(sayi1:Int,sayi2:Int) -> Int {
        let toplam = sayi1 + sayi2
        return toplam
    }
}

//Overloading
func carpma(sayı1:Int,sayı2:Int){
    print("Çarpma : \(sayı1 * sayı2)")
}

func carpma(sayı1:Double,sayı2:Double){
    print("Çarpma : \(sayı1 * sayı2)")
}

func carpma(sayı1:Int,sayı2:Int,isim:String){
    print("Çarpma : \(sayı1 * sayı2) - İşlemi yapan : \(isim)")
}

let f = Foksiyonlar()

f.selamla()

let gelenSonuc = f.selamla1()
print("Gelen Sonuç : \(gelenSonuc)")

f.selamla2(isim: "Zeynep")

let gelenToplam = f.toplama(sayı1: 10, sayı2: 20)
print("Gelen Toplam : \(gelenToplam)")

Kasım ADALI f.carpma(sayı1: 5, sayı2: 8, isim: "Beyza")
```

Import

Import

- `import` kodlama yaparken ihtiyaç duyacağımız sınıfların kodlama yaptığımız sınıfta kullanılmasına izin verir.
- Örn : Harita işlemleri için MapKit sınıflarını import etmeliyiz.

```
import UIKit  
import Foundation  
import MapKit
```

Static Değişkenler ve Metodlar

Static Değişkenler ve Metodlar

- Bir değişkenin veya metodun, bulunduğu sınıfın (class veya structure) nesne oluşturmaya gerek kalmadan erişilmek istenirse kullanılır.

```
class ASinifi {  
    static var x = 10  
  
    static func metod(){  
        print("Metod çalıştı")  
    }  
}  
  
/*let a = ASinifi()  
  
print(ASinifi().x)//Sanal nesne  
  
ASinifi().metod()*/  
  
print(ASinifi.x)  
  
ASinifi.metod()
```

Enumeration

Enumeration

- `enum` ifadesi gösterilir.
- Parametrelerde kullanılır.
- Verilerin eşleşmesi sonucunda bir işlem yapılır.
- Kodlama yapan yazılımcıyı detaydan kurtarmaktadır.

```
enum Renkler {  
    case Beyaz  
    case Siyah  
}  
  
var renk = Renkler.Beyaz  
  
switch renk {  
    case .Beyaz:  
        print("#FFFFFF")  
    case .Siyah:  
        print("#000000")  
}
```

Enumeration - Örnek

```
//Enumeration
enum KonserveBoyut {
    case kucuk
    case orta
    case buyuk
}

func ucretHesapla(boyut:KonserveBoyut,adet:Int){
    switch boyut {
        case .kucuk:
            print("Fiyat : \(adet * 10) ₺")
        case .orta:
            print("Fiyat : \(adet * 20) ₺")
        case .buyuk:
            print("Fiyat : \(adet * 30) ₺")
    }
}

ucretHesapla(boyut: .orta, adet: 23)
```

Composition

Composition

- Başka sınıflardan (class veya structure) olmuş nesneler bir sınıfın değişkeni olabilir.

```
class Adres {  
    var il:String?  
    var ilce:String?  
  
    init(il:String,ilce:String) {  
        self.il = il  
        self.ilce = ilce  
    }  
}
```

```
class Kisiler {  
    var ad:String?  
    var yas:Int?  
    var adres:Adres?  
  
    init(ad:String,yas:Int,adres:Adres) {  
        self.ad = ad  
        self.yas = yas  
        self.adres = adres  
    }  
}
```

```
var adres = Adres(il: "Bursa", ilce: "Osmangazi")  
  
var kisi = Kisiler(ad: "Ahmet", yas: 20, adres: adres)  
  
print("Kisi ad : \$(kisi.ad!)")  
print("Kisi ya  : \$(kisi.yas!)")  
print("Adre il : \$(kisi.adres!.il!)")  
print("Adre ilce : \$(kisi.adres!.ilce!)")
```

```
Kisi ad : Ahmet  
Kisi ya  : 20  
Adre il : Bursa  
Adre ilce : Osmangazi
```

Composition

Kategoriler Tablosu

kategori_id	kategori_ad
1	Dram
2	Komedi
3	Bilim Kurgu

Yonetmenler Tablosu

yonetmen_id	yonetmen_ad
1	Nuri Bilge Ceylan
2	Quentin Tarantino
3	2013

Filmler Tablosu

film_id	film_ad	film_yil	kategori_id	yonetmen_id
1	Django	2013	1	2
2	Inception	2006	3	3

```

class Filmler {
    var film_id:Int?
    var film_ad:String?
    var film_yil:Int?
    var kategori:Kategoriler?
    var yonetmen:Yonetmenler?

    init(film_id:Int,film_ad:String,film_yil:Int,
        kategori:Kategoriler,yonetmen:Yonetmenler){
        self.film_id = film_id
        self.film_ad = film_ad
        self.film_yil = film_yil
        self.kategori = kategori
        self.yonetmen = yonetmen
    }
}

let k1 = Kategoriler(kategori_id: 1, kategori_ad: "Dram")
let k2 = Kategoriler(kategori_id: 2, kategori_ad: "Bilim Kurgu")

let y1 = Yonetmenler(yonetmen_id: 1, yonetmen_ad: "Quentin Tarantino")
let y2 = Yonetmenler(yonetmen_id: 2, yonetmen_ad: "Christopher Nolan")

let f1 = Filmler(film_id: 1, film_ad: "Interstellar", film_yil: 2019, kategori: k2, yonetmen: y2)

print("Film id : \((f1.film_id!))")
print("Film ad : \((f1.film_ad!))")
print("Film yil : \((f1.film_yili!))")
print("Film kategori : \((f1.kategori!.kategori_ad!))")
print("Film yonetmen : \((f1.yonetmen!.yonetmen_ad!))")

```

```

class Kategoriler {
    var kategori_id:Int?
    var kategori_ad:String?

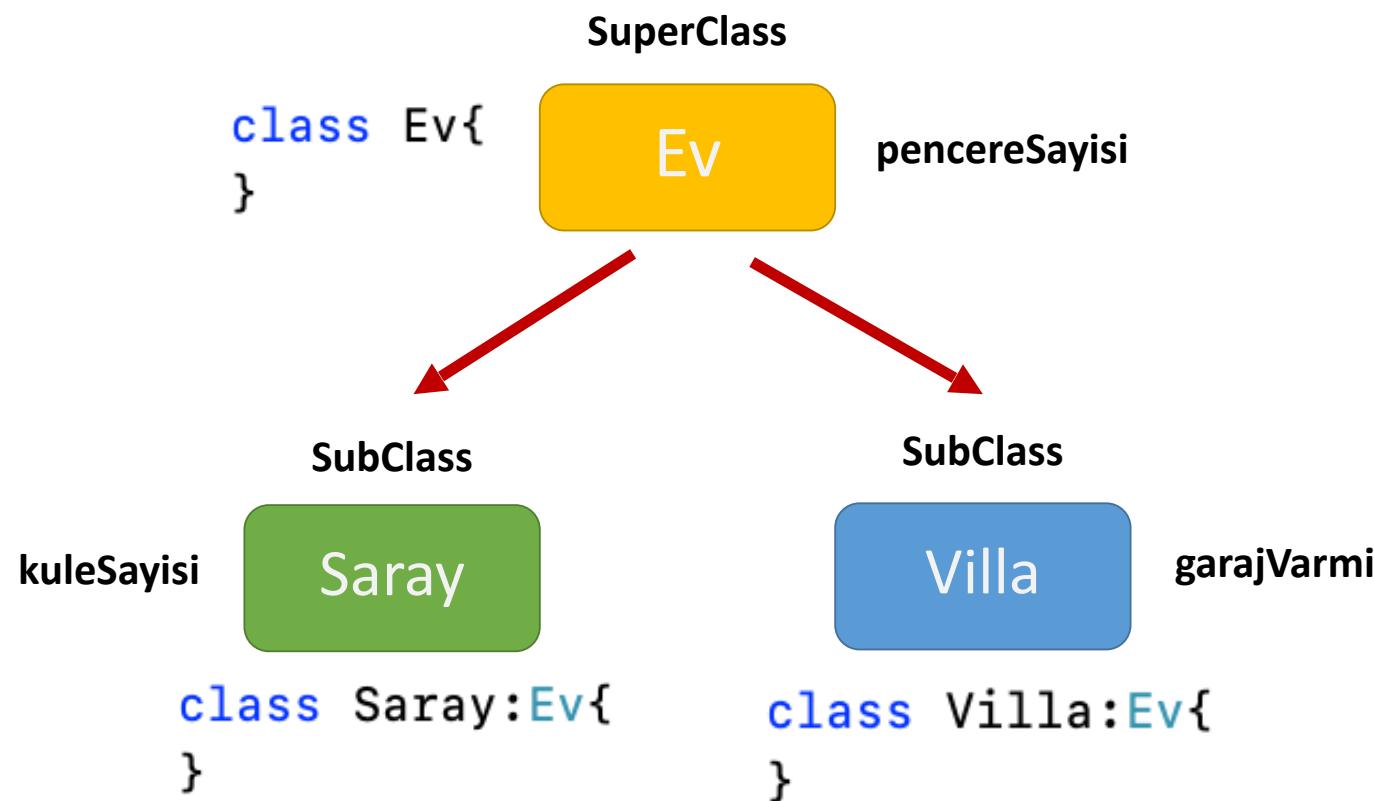
    init(kategori_id:Int,kategori_ad:String){
        self.kategori_id = kategori_id
        self.kategori_ad = kategori_ad
    }
}

class Yonetmenler {
    var yonetmen_id:Int?
    var yonetmen_ad:String?

    init(yonetmen_id:Int,yonetmen_ad:String){
        self.yonetmen_id = yonetmen_id
        self.yonetmen_ad = yonetmen_ad
    }
}

```

Kalıtım Hiyerarşisi Örnek



Kalıtım Hiyerarşisi Örnek

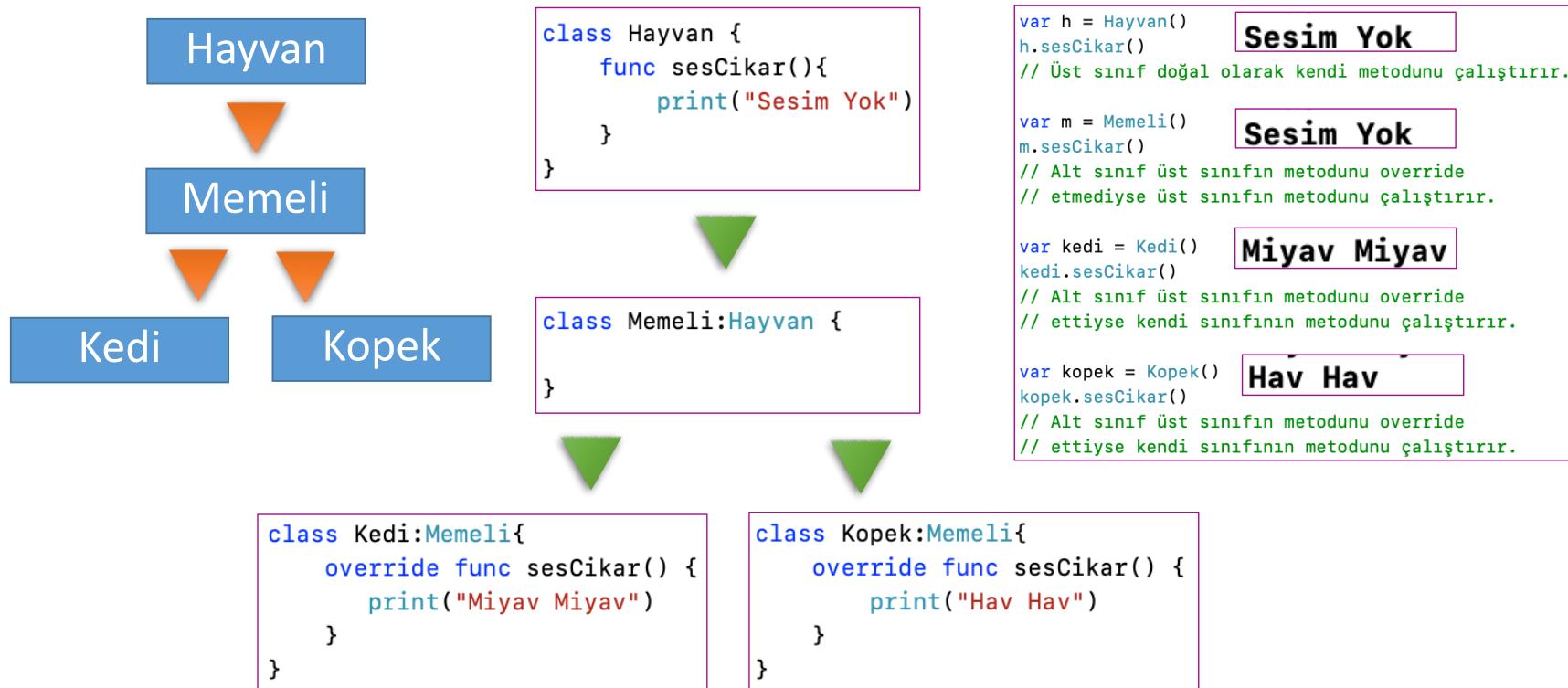
```
class Ev {  
    var pencereSayisi:Int?  
  
    init(pencereSayisi:Int) {  
        self.pencereSayisi = pencereSayisi  
    }  
}  
  
class Saray : Ev {  
    var kuleSayisi:Int?  
  
    init(kuleSayisi:Int,pencereSayisi:Int) {  
        self.kuleSayisi = kuleSayisi  
        super.init(pencereSayisi: pencereSayisi)  
    }  
}  
  
var topkapiSarayi = Saray(kuleSayisi: 5, pencereSayisi: 300)  
var bogazVilla = Villa(garajVarmi: true, pencereSayisi: 30)  
  
print(topkapiSarayi.kuleSayisi!)  
print(topkapiSarayi.pencereSayisi!)  
  
print(bogazVilla.garajVarmi!)  
print(bogazVilla.pencereSayisi!)  
  
class Villa : Ev {  
    var garajVarmi:Bool?  
  
    init(garajVarmi:Bool,pencereSayisi:Int) {  
        self.garajVarmi = garajVarmi  
        super.init(pencereSayisi: pencereSayisi)  
    }  
}
```



Override

Metodları Ezme : Overriding

- Kalıtım ilişkisinde üst sınıfın metodlarının alt sınıf tarafından tekrar kullanılmasıdır.



Nesnelerin Tip Dönüşümü

```
//Upcasting

var ev = Saray(kuleSayisi: 3, pencereSayisi: 10) as Ev

//Downcasting

var saray = Ev(pencereSayisi: 6) as? Saray

//Tip kontrolü

if ev is Ev {
    print("Nesne ev sınıfından türemiştir.")
}else{
    print("Nesne ev sınıfından türetilmemiştir.")
}
```

Protocol

Protocol Örnek

```
protocol MyProtocol {
    var degisken : Int {get set}

    func metod1()
    func metod2() -> String
}

class ClassA : MyProtocol {
    var degisken: Int = 10

    func metod1() {
        print("Metod 1 çalıştı")
    }

    func metod2() -> String {
        return "Metod 2 çalıştı"
    }
}

var a = ClassA()

print(a.degisken)
a.metod1()
print(a.metod2())
```

Extension

Extension – Metod Örnek

```
extension Int {  
    func carp(sayı:Int) -> Int {  
        return self * sayı  
    }  
}
```

```
let x = 3.carp(sayı: 10)  
  
print(x)
```

Closure

Closure Örnek : Geri Dönüş Olmayan

```
let ifade = {  
    print("Closure Konusuna Hoşgeldin")  
}  
  
ifade()
```

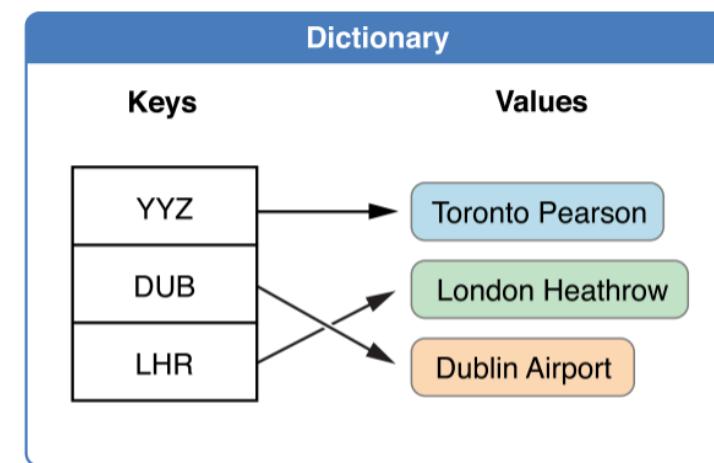
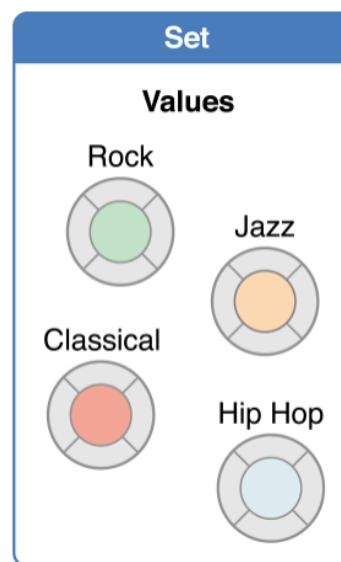
Çıktı

Closure Konusuna Hoşgeldin

Collections

Collection Types

Array	
Indexes	Values
0	Six Eggs
1	Milk
2	Flour
3	Baking Powder
4	Bananas



```

var numaralar = [10,20,30]
var meyveler = [String]()

//Veri ekleme
meyveler.append("Elma")//0.
meyveler.append("Muz")//1.
meyveler.append("Kiraz")//2.
print(meyveler)

//Güncelleme
meyveler[0] = "Yeni Elma"
print(meyveler)

//Insert
meyveler.insert("Portakal", at: 1)
print(meyveler)

//Veri okuma
print(meyveler[2])
print(meyveler.isEmpty)//Boş mu ?
print(meyveler.count)//boyut
print(meyveler.first!)//0. veya ilk eleman
print(meyveler.last!)//Son indeks elemanı
print(meyveler.contains("Muzx"))
print(meyveler.firstIndex(of: "Muz")!)//İçeriğe göre indeks değeri verir

for meyve in meyveler {
    print("Sonuç 1 : \(meyve)")
}

for (indeks,meyve) in meyveler.enumerated() {
    print("\(indeks). -> \(meyve)")
}

meyveler.remove(at: 1)
print(meyveler)

meyveler.removeAll()
print(meyveler)

```

Nesne Tabanlı

```
class Ogrenciler {
    var no:Int?
    var ad:String?
    var sinif:String?

    init(no:Int,ad:String,sinif:String){
        self.no = no
        self.ad = ad
        self.sinif = sinif
    }
}

var o1 = Ogrenciler(no: 200, ad: "Zeynep", sinif: "9C")
var o2 = Ogrenciler(no: 300, ad: "Ahmet", sinif: "11Z")
var o3 = Ogrenciler(no: 100, ad: "Beyza", sinif: "12A")

var ogrencilerListesi = [Ogrenciler]()
ogrencilerListesi.append(o1)
ogrencilerListesi.append(o2)
ogrencilerListesi.append(o3)

for o in ogrencilerListesi {
    print("No : \(o.no!) - Ad : \(o.ad!) - Sınıf : \(o.sinif!)" )
}
```

Array Filtreleme

```
var sayilar = [1,2,3,4,5,6,7,8,9,10]

var sonuc1 = sayilar.filter({$0>7})
print(sonuc1)//[8, 9, 10]

var sonuc2 = sayilar.filter({$0<7})
print(sonuc2)//[1, 2, 3, 4, 5, 6]

var sonuc3 = sayilar.filter({$0>3 && $0<7})
print(sonuc3)//[4, 5, 6]
```

```
var f1 = ogrencilerListesi.filter({$0.no! > 100})
print("Filtreleme 1")
for o in f1 {
    print("No : \(o.no!) - Ad : \(o.ad!) - Sınıf : \(o.sinif!)")
}

var f2 = ogrencilerListesi.filter({$0.ad!.contains("y")})
print("Filtreleme 2")
for o in f2 {
    print("No : \(o.no!) - Ad : \(o.ad!) - Sınıf : \(o.sinif!)")
}
```

Array sort()

```
print("Sayısal Büyüktен Küçüğe")
let siralamaArray1 = kisilerArray.sorted(by: {$0.kisiNo > $1.kisiNo} )
```

```
print("Sayısal Küçükten Büyüge")
let siralamaArray2 = kisilerArray.sorted(by: {$0.kisiNo < $1.kisiNo} )
```

```
print("Harf Küçükten Büyüge")
let siralamaArray3 = kisilerArray.sorted(by: {$0.kisiAd < $1.kisiAd} )
```

```
var s1 = ogrencilerListesi.sorted(by: {$0.no! > $1.no!})
print("Sayısal olarak büyükten küçüğe")
for o in s1 {
    print("No : \(o.no!) - Ad : \(o.ad!) - Sınıf : \(o.sinif!)")
}

var s2 = ogrencilerListesi.sorted(by: {$0.no! < $1.no!})
print("Sayısal olarak küçükten büyüğe")
for o in s2 {
    print("No : \(o.no!) - Ad : \(o.ad!) - Sınıf : \(o.sinif!)")
}

var s3 = ogrencilerListesi.sorted(by: {$0.ad! > $1.ad!})
print("Harfsel olarak büyükten küçüğe")
for o in s3 {
    print("No : \(o.no!) - Ad : \(o.ad!) - Sınıf : \(o.sinif!)")
}

var s4 = ogrencilerListesi.sorted(by: {$0.ad! < $1.ad!})
print("Harfsel olarak küçükten büyüğe")
for o in s4 {
    print("No : \(o.no!) - Ad : \(o.ad!) - Sınıf : \(o.sinif!)")
```

Set Çalışması

```
//Set Kullanımı
var meyveler1 = Set<String>()
meyveler1.insert("Elma")
meyveler1.insert("Portakal")
meyveler1.insert("Muz")

print(meyveler1)

meyveler1.insert("Amasaya Elması")
print(meyveler1)

for meyve in meyveler1 {
    print("Sonuç 1 : \(meyve)")
}

for (indeks,meyve) in meyveler1.enumerated() {
    print("\(indeks). -> \(meyve)")
}

print(meyveler1.count)

var indeks = meyveler1.firstIndex(of: "Elma")
meyveler1.remove(at: indeks!)
print(meyveler1)

meyveler1.remove("Muz")
print(meyveler1)

meyveler1.removeAll()
print(meyveler1)
```

Dictionary Çalışması

```
var iller = [Int:String]()
//Veri Ekleme
iller[16] = "BURSA"
iller[34] = "İSTANBUL"
print(iller)

//Veri Okuma
print(iller[16]!)

//Veri Güncelleme
iller[16] = "YENİ BURSA"
print(iller)

for (anahtar,deger) in iller {
    print("\(anahtar) -> \(deger)")
}

iller.removeValue(forKey: 16)
print(iller)
```

ileri Swift

guard

if

```
func kisiTanima(ad:String){  
    if ad == "Ahmet" {  
        print("Merhaba Ahmet")  
    }else{  
        print("Tanınmayan Kişi")  
    }  
  
    kisiTanima(ad: "Ahmet")
```

Guard

```
func kisiTanima(ad:String){  
    guard ad == "Ahmet" else {  
        print("Tanınmayan Kişi")  
        return  
    }  
    print("Merhaba Ahmet")  
  
    kisiTanima(ad: "Ahmet")
```

Return :

- Hata oluşduğunda veya şart sağlanmadığında metodu bitirir.

Optional ifade

if

```
func buyukHarfYap(str:String?){
    if let temp = str {
        print("\(temp.uppercased())")
        Otomatik unwrapping
    }else{
        print("str nil dir.İşlem yapılamaz")
        return
    }
}
buyukHarfYap(str:nil)
```

Guard

```
func buyukHarfYap(str:String?){
    guard let temp = str else {
        print("str nil dir.İşlem yapılamaz")
        return
    }
    Otomatik unwrapping
    print("\(temp.uppercased())")
}
```

Not : temp değeri guard'dan geçtikten sonra kullanılabilir ve doğal olarak buyukHarfYap(str:nil) unwrapping olur.

str nil dir.İşlem yapılamaz

Exception Hata Ayıklama

Örnek

```
enum Hatalar:Error {
    case sifiraBolunmeHatasi
}

func bolme(sayi1:Int,sayi2:Int) throws -> Int {
    if sayi2 == 0 {
        throw Hatalar.sifiraBolunmeHatasi
    }
    return sayi1 / sayi2
}

do {
    let sonuc = try bolme(sayi1: 10, sayi2: 2)
    print(sonuc)
}catch Hatalar.sifiraBolunmeHatasi {
    print("Sayı sıfıra bölünemez")
}

let sonuc1 = try? bolme(sayi1: 10, sayi2: 2)//sonuc1 hata olursa nil olur.

if let temp = sonuc1 {
    print(temp)
}else{
    print("Sayı sıfıra bölünemez")
}
```

try?

- Hata yok sayılır veya görmezden gelinir fakat hata oluşursa değişkeni **nil** yapar.
- **Do** ve **catch** bloğuna ihtiyaç yoktur.
- try ? dan önce yer alan ifade **throw** özelliği olan metod olmalıdır.

```
let sonuc = try? bolme(sayi1: 10, sayi2: 0)

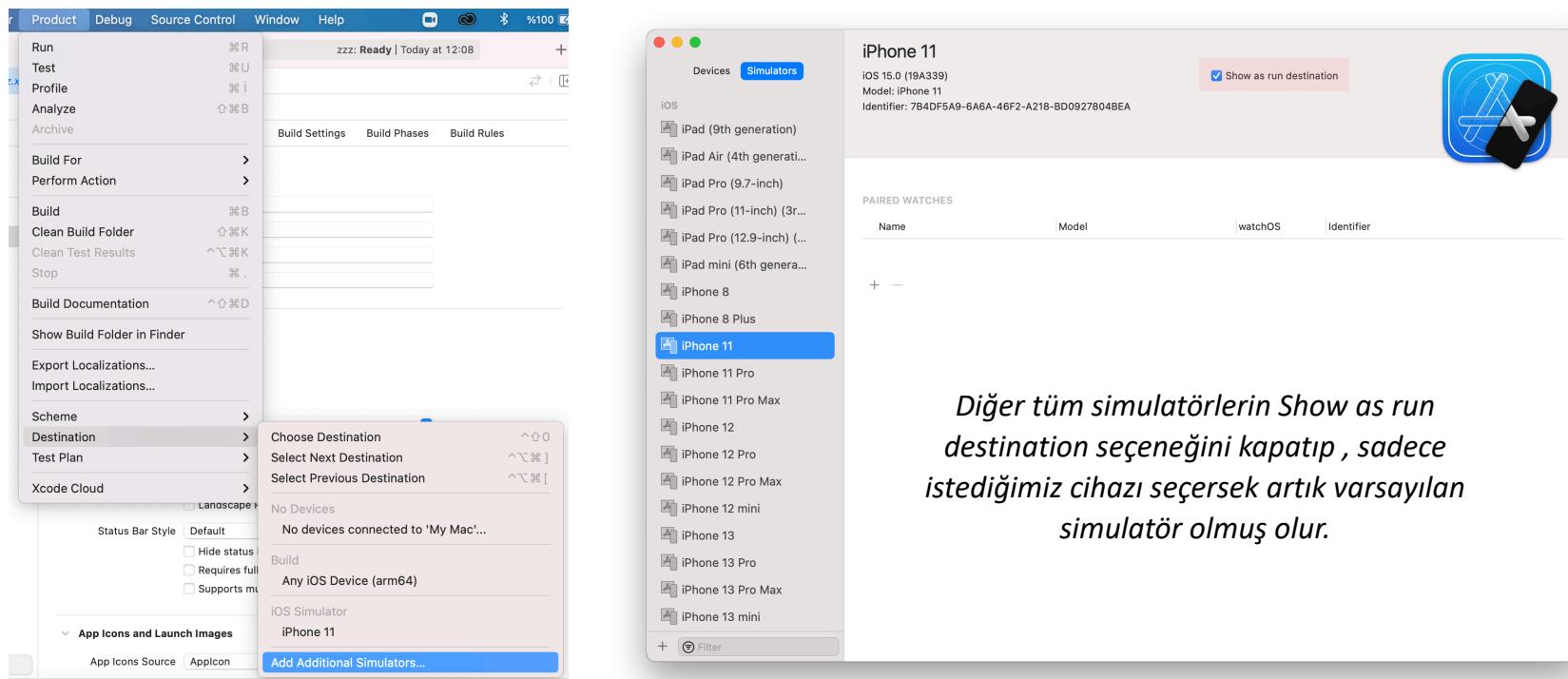
if let temp = sonuc {
    print(temp)
}else{
    print("Sayı sıfıra bölünemez")
}
```

IOS Giriş

Simulatör Oluşturma - ilk Uygulamayı Çalıştırma - Proje Dizini

Proje Açılışındaki Simulatörü Seçme

Xcode projesi oluşturduğumuzda xcode varsayılan bir simulatör seçimi yapar.
Bu seçimi istediğimiz cihaz ile değiştirebiliriz.



Diğer tüm simulatörlerin Show as run destination seçeneğini kapatıp , sadece istediğimiz cihazı seçersek artık varsayılan simulatör olmuş olur.

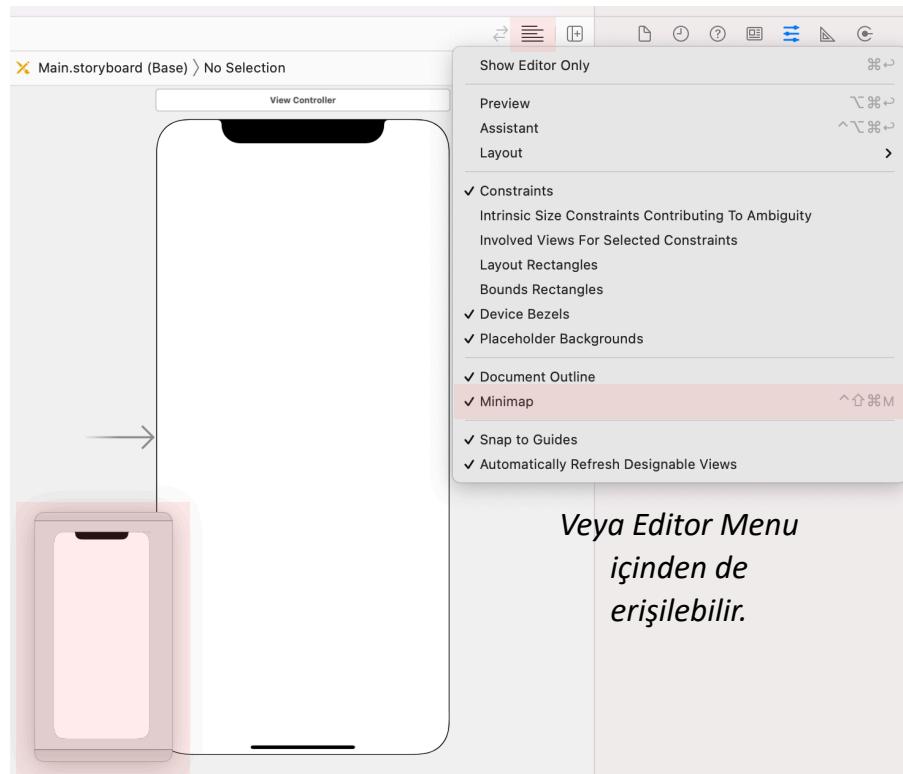
Bazı Kısıyollar

- Playground run etme : Shift + Cmd + Enter
- Hazır init metodu oluşturma : Right-click (Sınıf ismi üzerine) > Refactor > 'Generate Memberwise Initializer'
- Kodları düzenleme : cmd + a ile tüm kodu seç , cntrl + i yapılır.
- Kodların ismini değiştirme : Refactor
- Emülatör üzerinde Dark Mode : shift + cmd + n
- Xcode üzerinde dosya uzantısı görünür yapma : Preferences > General > File Extensions > Show All

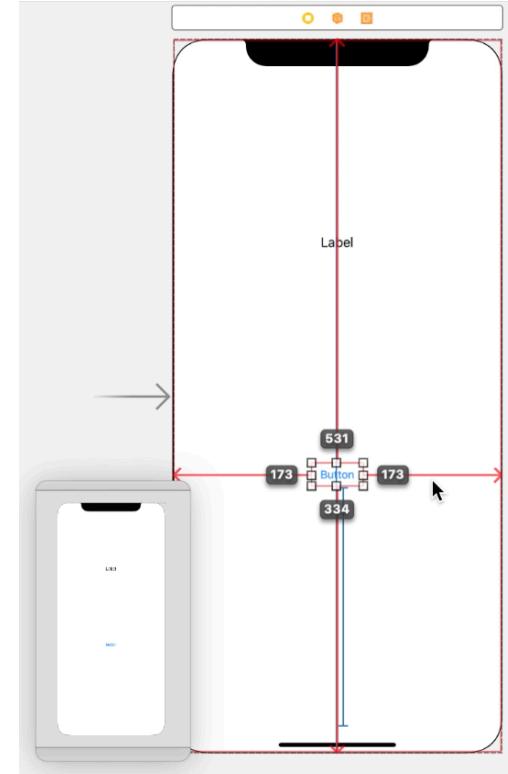
IOS Tasarım

Tasarım için Bazı Kısıyollar

Minimap kapatma

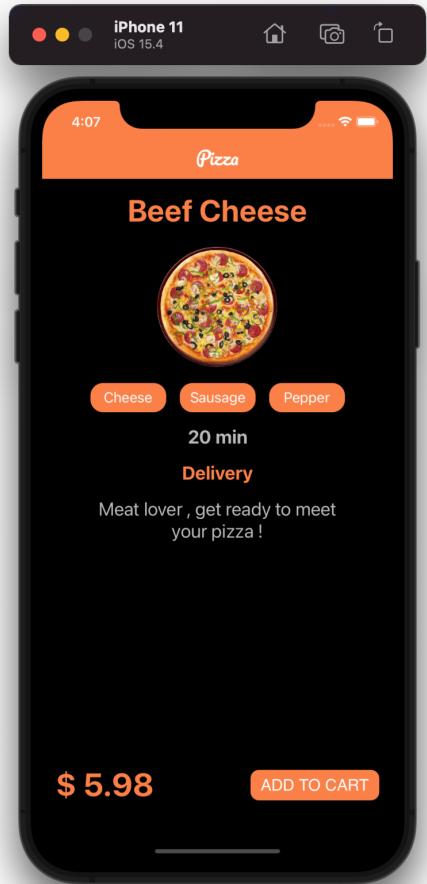
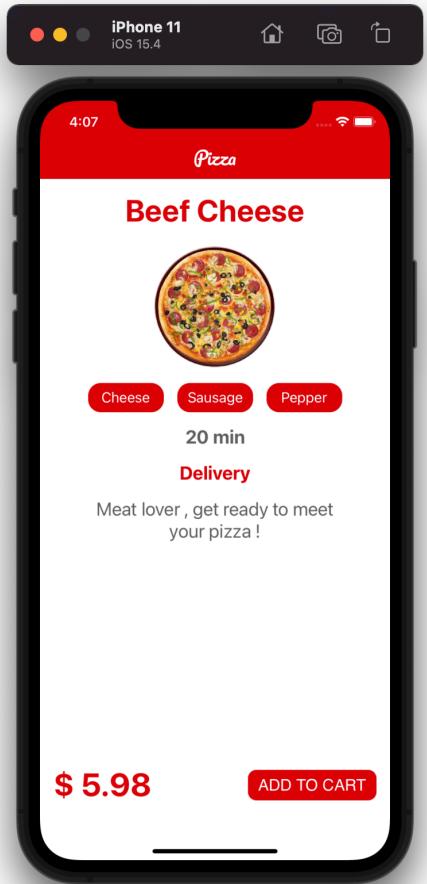


Mesafeleri Görme

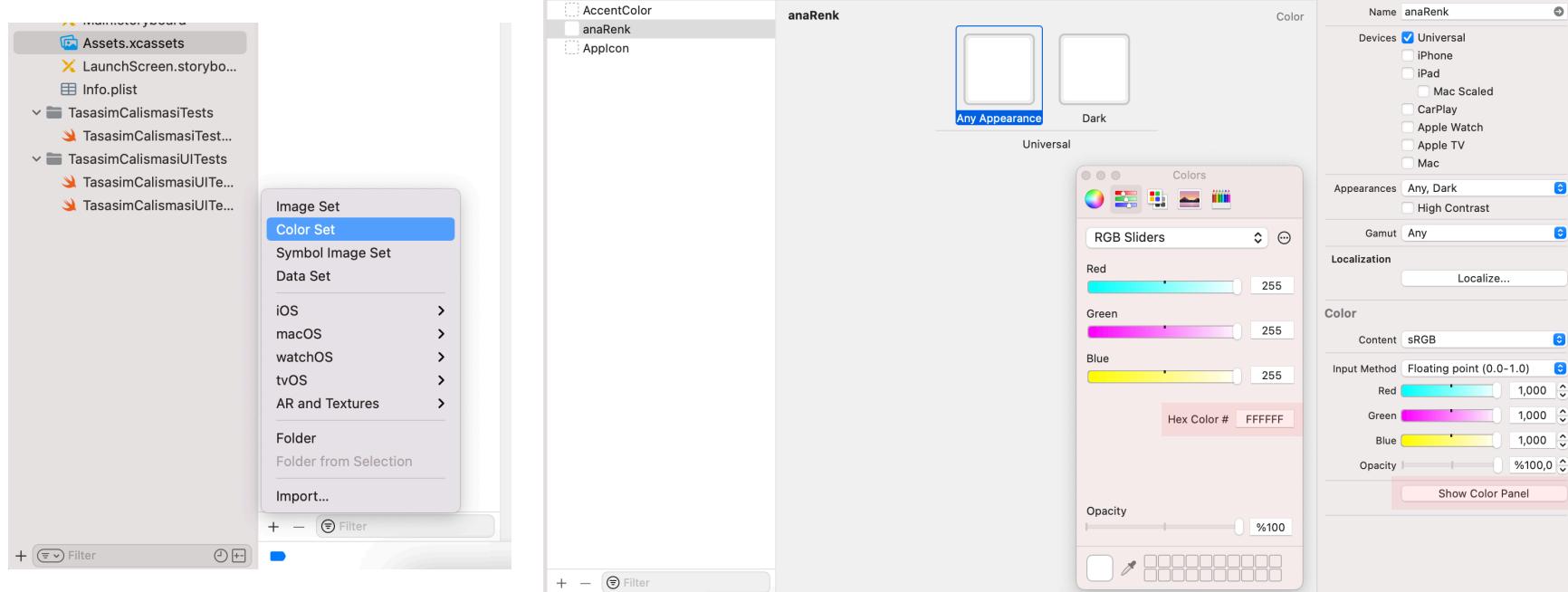


Tasarım alanında görsel nesneyi seçip

Option tuşunu basılı tutularak mesafeyi ölçmek istediğimiz görsel nesne üzerine gelmemiz yeterlidir.



Renk Oluşturma



Kullanılacak Renkler



Oluşturduğumuz dark mode renkleri otomatik çalışmaktadır.

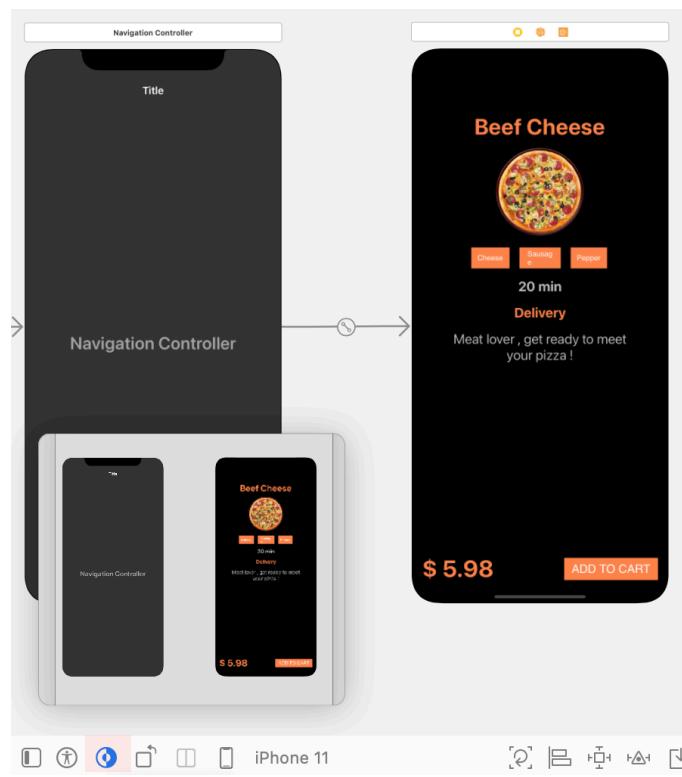


Her iki durumda aynı renk olabilir.

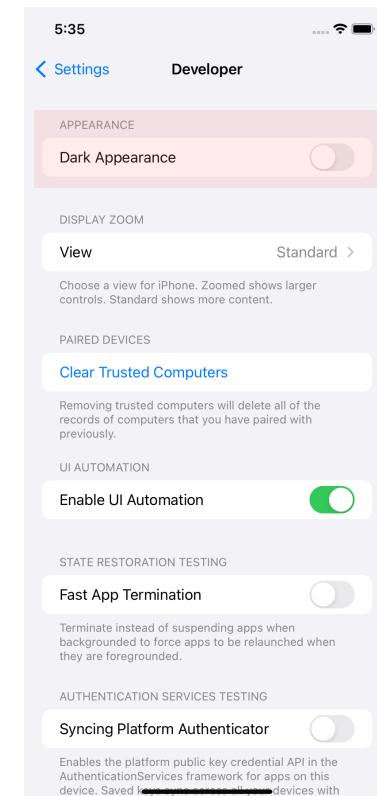


Dark Mode Test

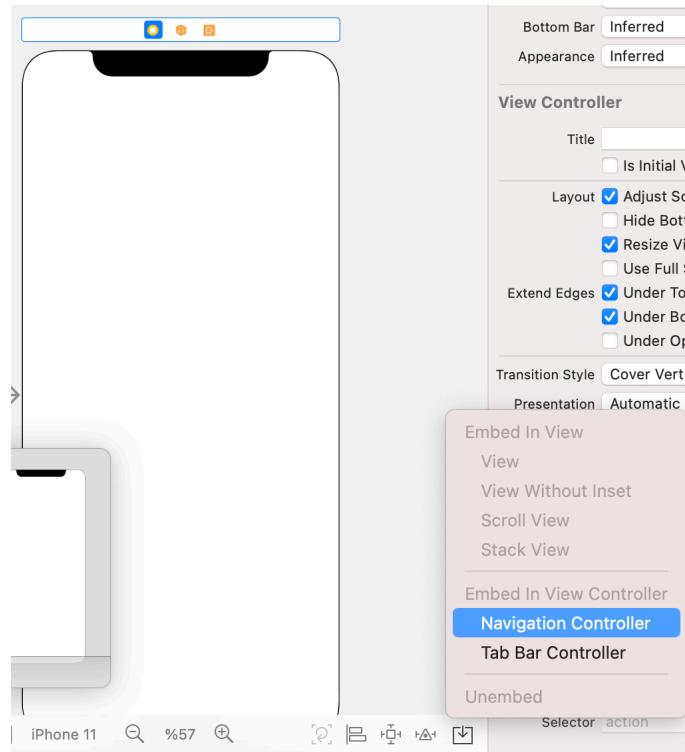
Storyboard üzerinde



Simulatör üzerinde dark mode aktif edilmelidir.



Navigation Controller Ekleme



Navigation Controller Başlık ve Renklendirme

```
class ViewController: UIViewController {

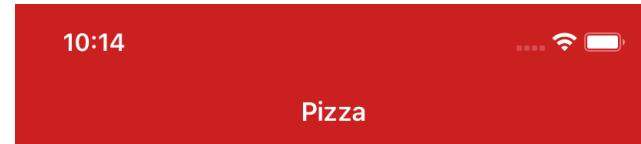
    override func viewDidLoad() {
        super.viewDidLoad()

        self.navigationItem.title = "Pizza"

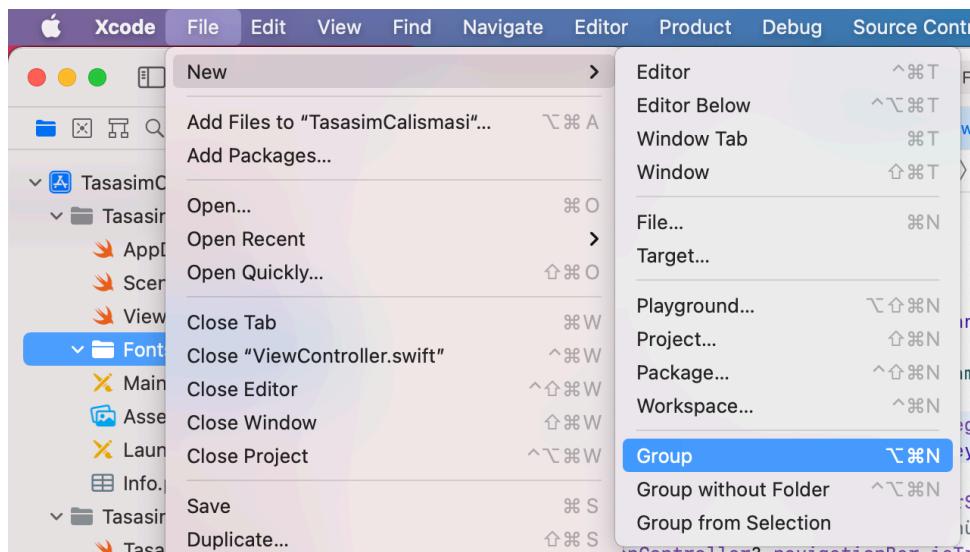
        let appearance = UINavigationBarAppearance()

        appearance.backgroundColor = UIColor(named: "anaRenk") Assets renklerine kodlama ile erişim
        appearance.titleTextAttributes = [.foregroundColor: UIColor(named: "yaziRenk1")!]
        navigationController?.navigationBar.barStyle = .black

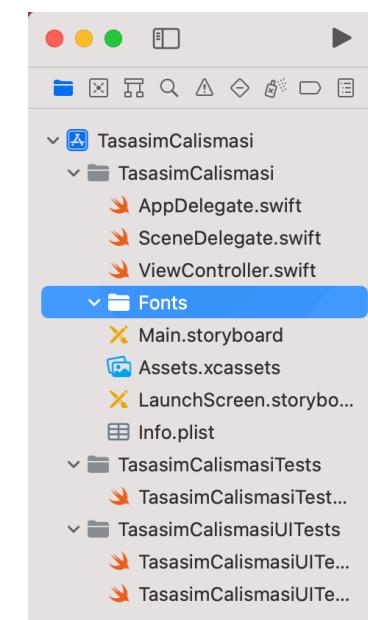
        navigationController?.navigationBar.standardAppearance = appearance
        navigationController?.navigationBar.compactAppearance = appearance
        navigationController?.navigationBar.scrollEdgeAppearance = appearance
    }
}
```



Navigation Controller Font Ekleme

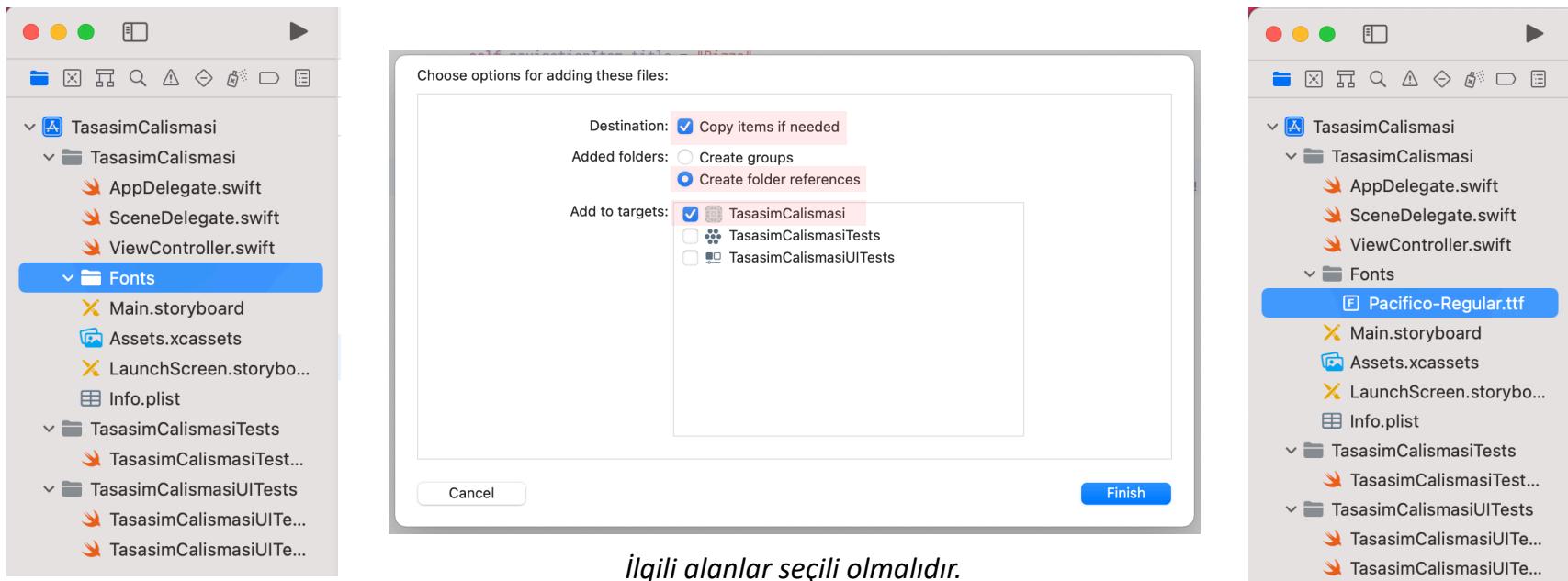


Dosyaları gruplamak için Group oluştur.



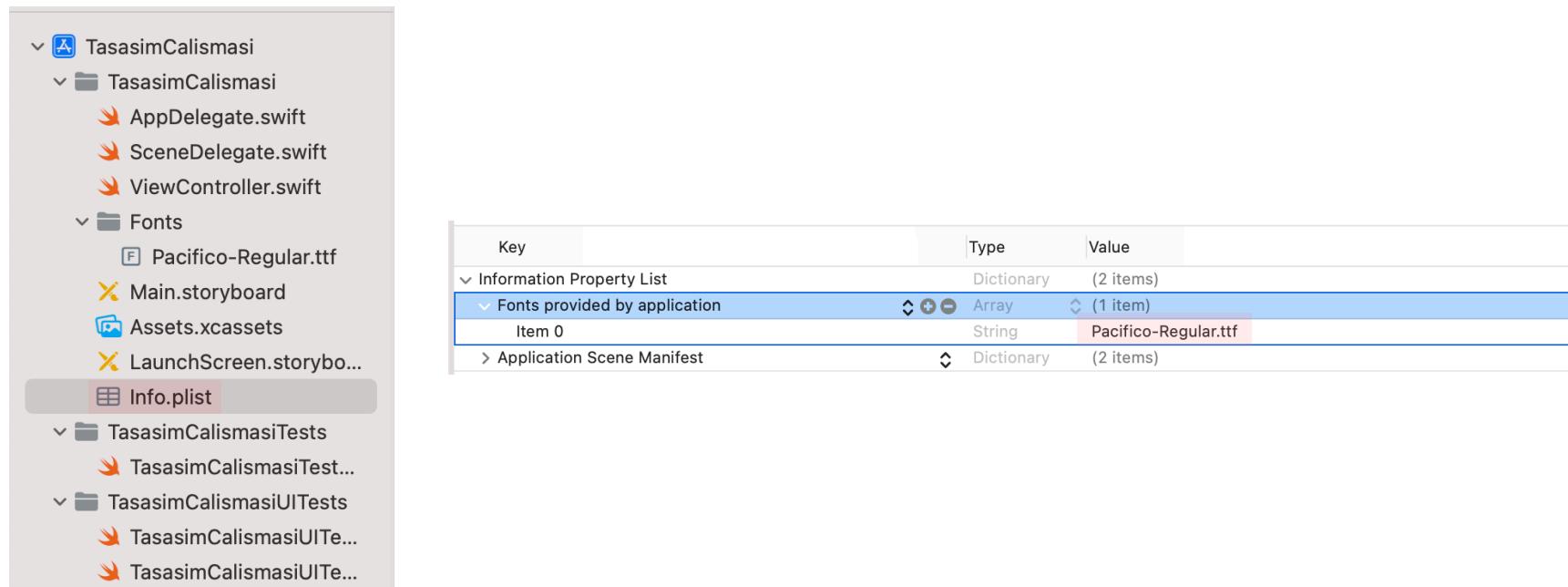
*Group ismi Fonts olmak zorunda
değil istediğimiz gibi
tanımlayabiliriz.*

Navigation Controller Font Ekleme



Font dosyasını Fonts klasörüne sürükle.

Navigation Controller Font Ekleme



Navigation Controller Font Ekleme

```
class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

        self.navigationItem.title = "Pizza"

        let appearance = UINavigationBarAppearance()

        appearance.backgroundColor = UIColor(named: "anaRenk")
        appearance.titleTextAttributes = [.foregroundColor: UIColor(named: "yaziRenk1")!,
                                         .font: UIFont(name: "Pacifico-Regular", size: 22)!]
        navigationController?.navigationBar.barStyle = .black

        navigationController?.navigationBar.standardAppearance = appearance
        navigationController?.navigationBar.compactAppearance = appearance
        navigationController?.navigationBar.scrollEdgeAppearance = appearance
    }
}
```



Color anaRenk () ()
Font System Bold 36.0 T () ()

Yatay hizalama



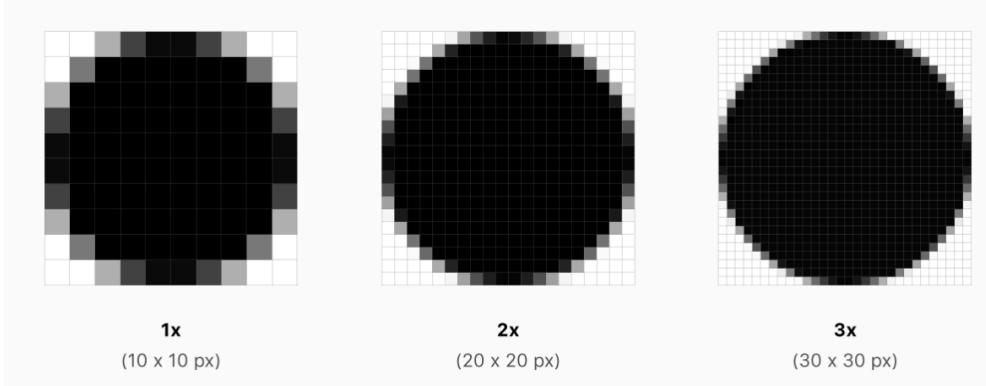
\$ 5.98

ADD TO CART

Çoklu Ekran Desteği

Resimlerin boyutlarını ayarla

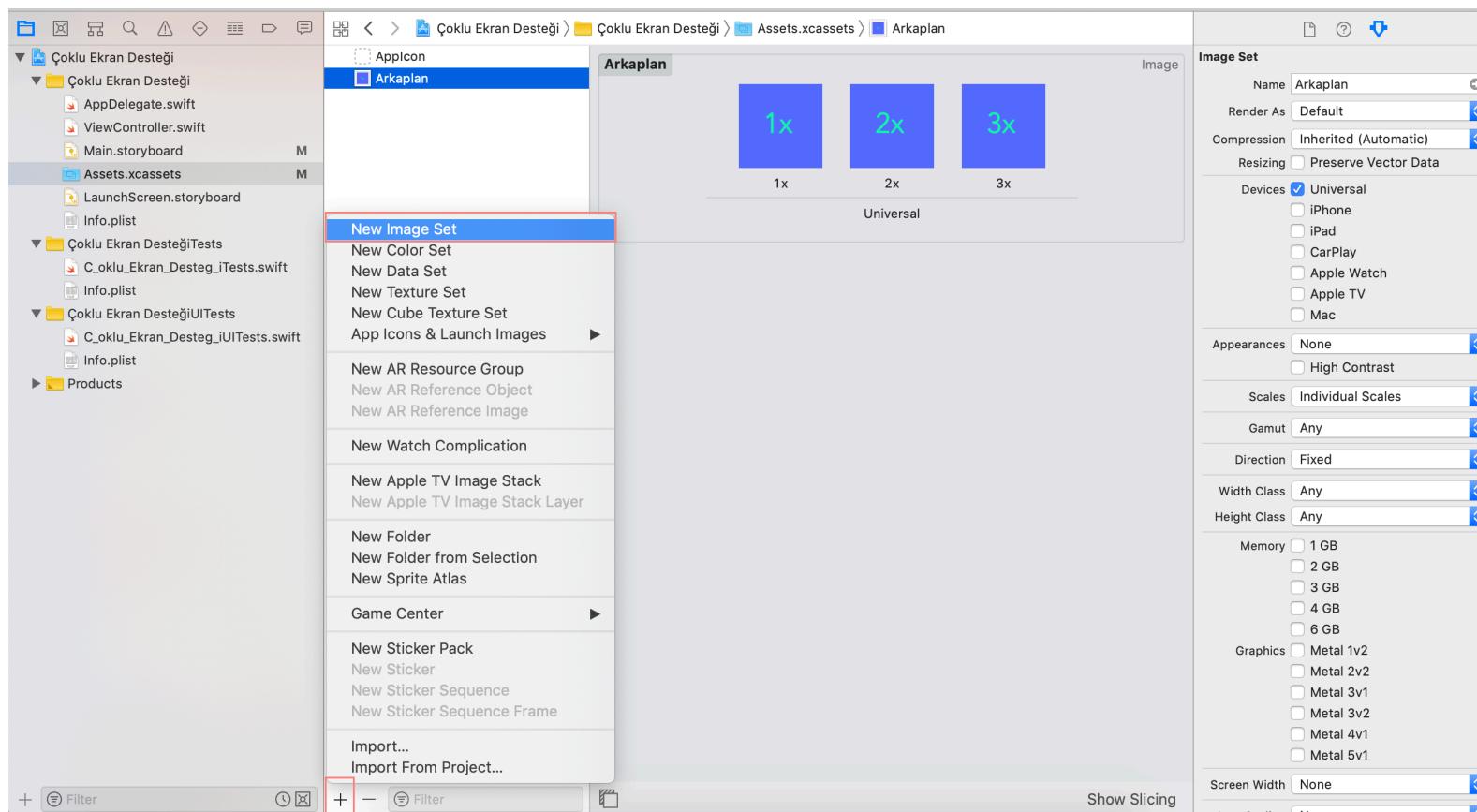
- Uygulamlarda resim kullanırken dikkatli olmalıyız.Bütün ekranlarda aynı kalitede olduğundan emin olmalıyız.
- Resim boyutlarını apple'ın istediği formatlarda belirlemeliyiz.
- **Aynı resmi 1x - 2x - 3x oranları ile boyutlandırılır.**
- Bu resimlere isim verirken [resim@1x.png](#) , [resim@2x.png](#) , [resim@3x.png](#) şeklinde isimlendirilir.



Apple Cihazlarında Boyut Çarpanları

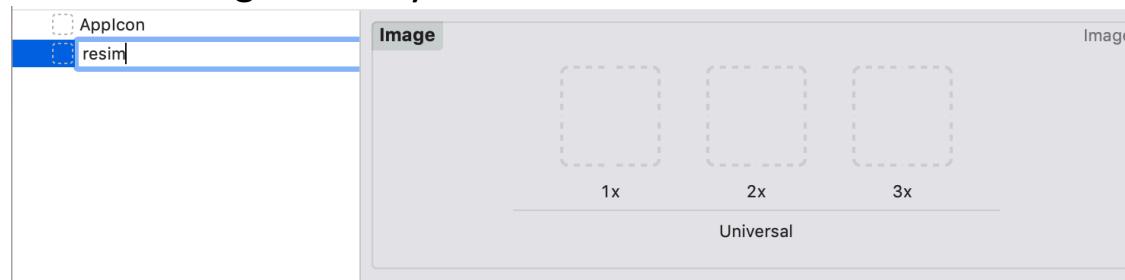
Device	Scale Factor
12.9" iPad Pro	@2x
11" iPad Pro	@2x
10.5" iPad Pro	@2x
9.7" iPad	@2x
7.9" iPad mini 4	@2x
iPhone Xs Max	@3x
iPhone Xs	@3x
iPhone XR	@2x
iPhone X	@3x
iPhone 8 Plus	@3x
iPhone 8	@2x
iPhone 7 Plus	@3x
iPhone 7	@2x
iPhone 6s Plus	@3x
iPhone 6s	@2x
iPhone SE	@2x

Xcode içerisinde image set oluşturma

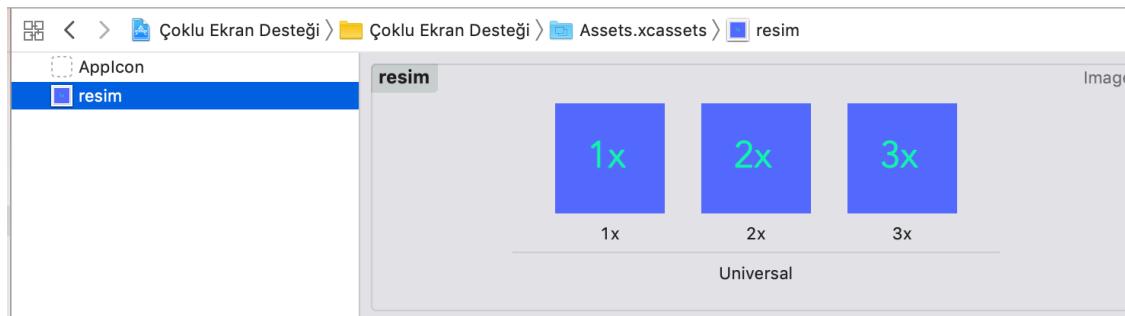


Xcode içerisinde image set oluşturma

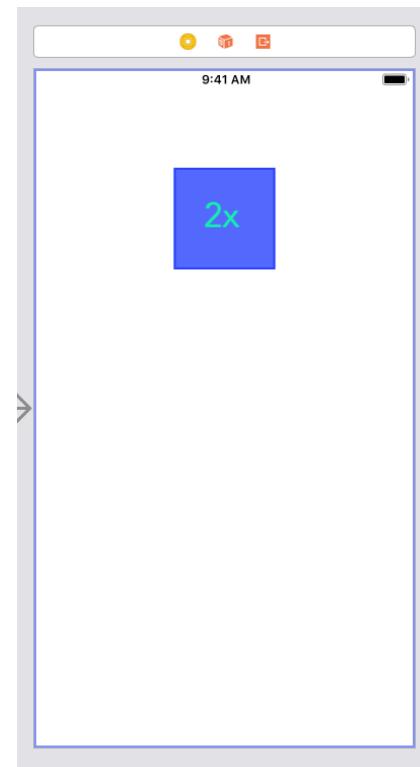
- Image set ismi resimlerin adları ile aynı olması iyi olur çünkü resimleri kodlama yaparken veya tasarım alanında image set adıyla kullanılır.



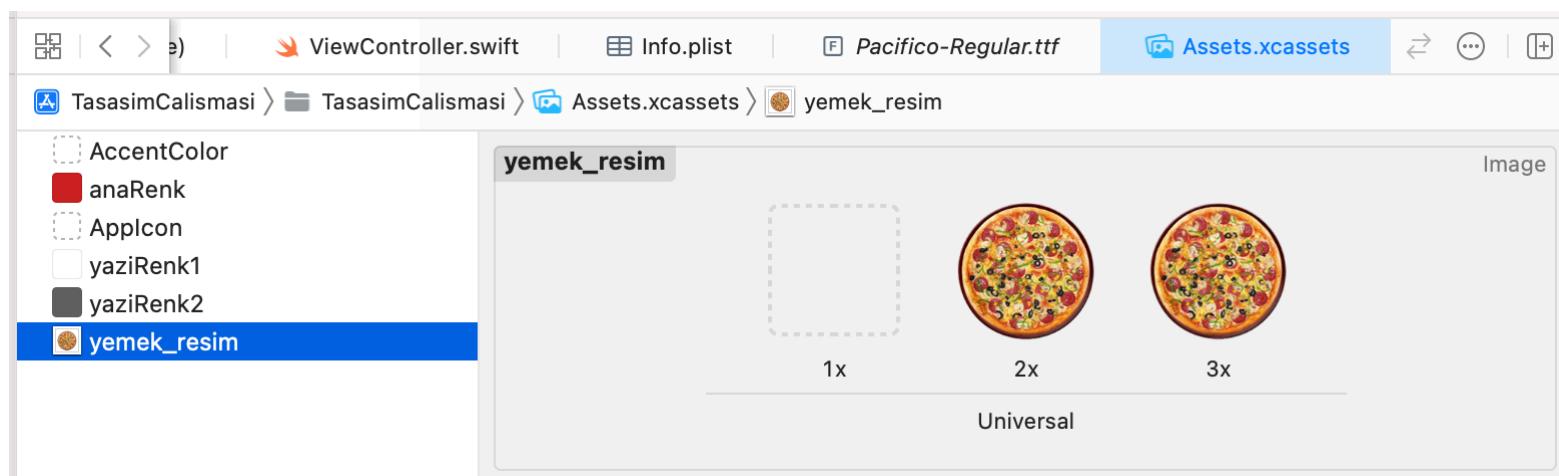
- Image set içerisinde görünen 1x 2x 3x oranlarına göre oluşturduğumuz resimlerinizi yerleştirin.



UYGULAMA



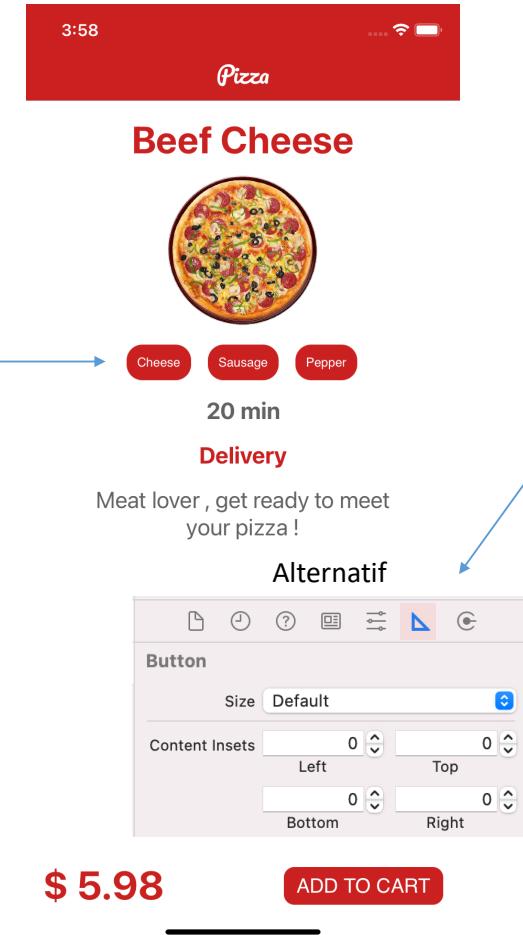
Pizza Resmi Çoklu Ekran Desteği



Button

Button : Yazı boyutu 12
Yazı rengi değişimi.

Boyut değişimde Style plain olmalı
bazen plain olunca boyut seçimi
yapılmıyor , attribute seçimi yapıp
ordan bir hızlama seçimi yapıp
tekrar plain seçebilirsin
düzeliyor.Ayrıca çoklu dil desteği için
plain olmalıdır.



Button yazı için iç boşluk
vermeliyiz.

Background Configuration

Background Default
Corner Style Small
Content Insets Custom
Leading Top
Bottom Trailing

Sonra kenar kıvrımı
verebiliriz.

Stack View

Stack View Kullanımı

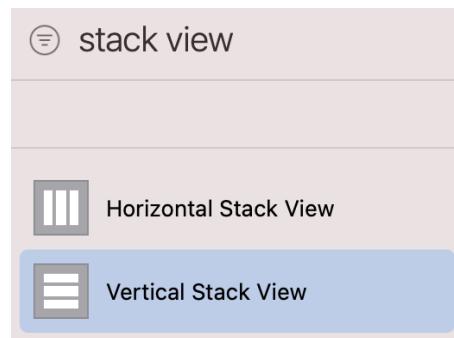
Görsel nesneleri yatay ve dikeyde istediğimiz gibi gruplayabiliriz.

Android tarafından Linear layouta benzemektedir.

Görsel nesneleri sıralı şekilde tutmak için idealdir.

Stack View içindeki görsel nesneleri Stack View içine sabitlememize gerek yoktur. Otomatik olarak sıralanmaktadır.

İstenirse görsel nesnelerin aralarına boşluk verilebilir ve boşluk verme tarzlarını değiştirebiliriz.



Stack View içinde görsel nesne olursa ekrana göre sabitlenmelidir.

View Controller

- View**
- Safe Area**
- Stack View**
 - XX**
 - XXXXXX**
 - XXXX**
- Constraints**

stack view

Horizontal Stack View

Vertical Stack View

Stack View

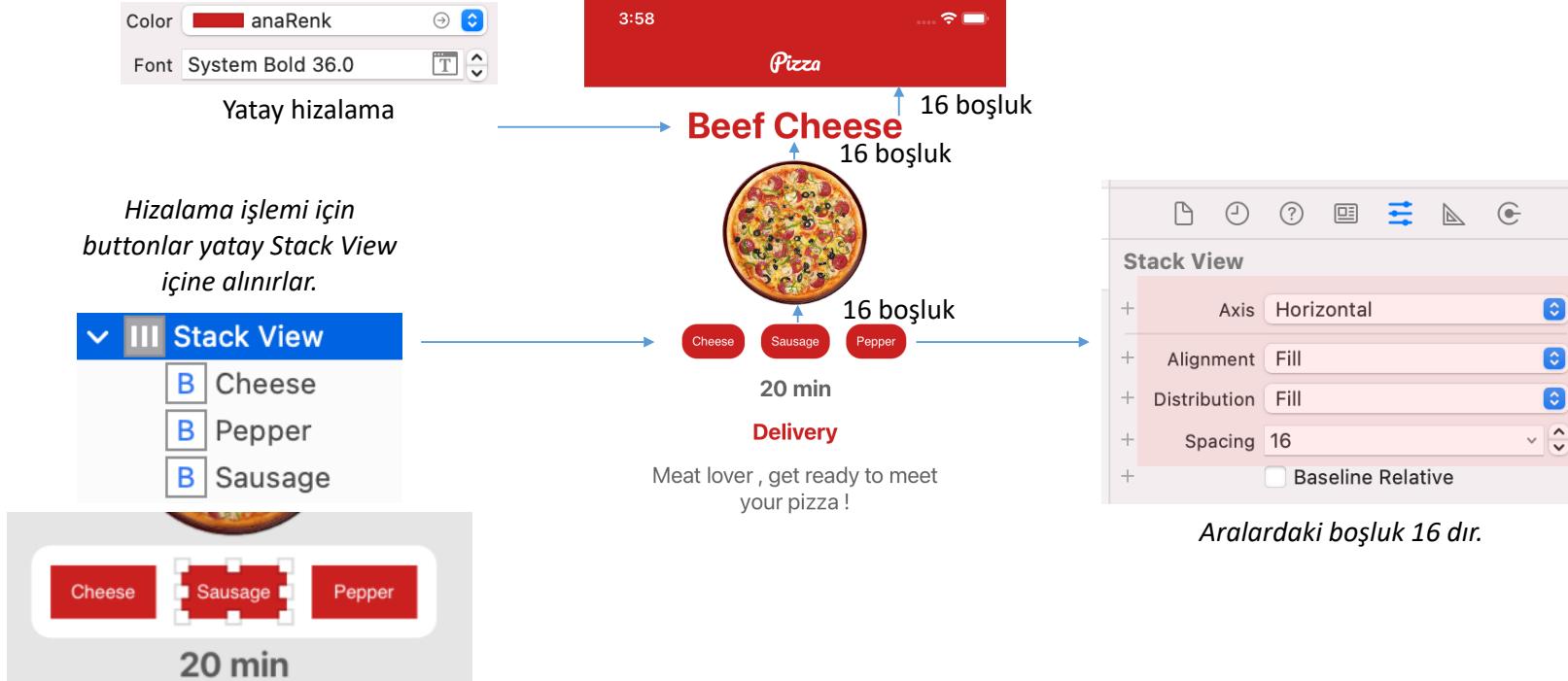
- Axis**: Vertical
- Alignment**: Center
- Distribution**: Equal Spacing
- Spacing**: 16
- Baseline Relative

Görsel nesne sıralanma yönü
Görsel nesnelerin Stack içindeki hizalanması
Görsel nesnelerin içeriklerinin dağılımı
Görsel nesnelerin aralarındaki boşluk miktarı

View

- Show**: Alignment Rectangle
 - X: 137,5
 - Y: 365,5
 - Width: 139
 - Height: 165
- Arrange**: Position View
- Layout**: Inferred (Constraints)
- Layout Margins**: Language Directional
 - Leading: 20
 - Top: 20
 - Bottom: 20
 - Trailing: 20

Stack View içeriğine iç boşluk verme



Stack View'i nerde durmasını istiyorsak oraya sürüklüyoruz.

Yatayda ortala ve yukarı 16 boşluk ile sabitleriz.

\$ 5.98

ADD TO CART

İngilizce Açıklama

Meat lover , get ready to meet your pizza !

Türkçe Açıklama

Et sever , pizzanla tanışmaya hazırlan !

Yatay hizalama

Color yaziRenk2

Font System Bold 22.0

Color anaRenk

Font System Bold 22.0

+ Color yaziRenk2

+ Font System 22.0

Dynamic Type Automatically Adjusts Font

Alignment

Lines

Bold değil - Yatayda hızla -
Kenarlardan 32 boşluk ver -
Satır sayısını 2 yap

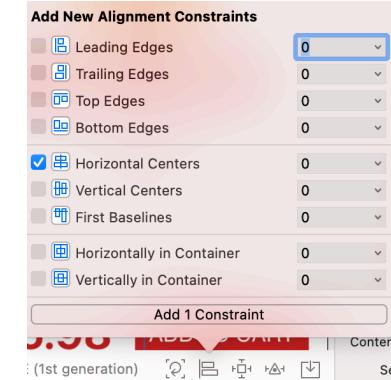


Beef Cheese



Cheese Sausage Pepper
20 min 16 boşluk
Delivery 16 boşluk
16 boşluk
Meat lover , get ready to meet
your pizza !

20 min ve Delivery seçip
yatayda hızlamalısın.



Ardından Delivery yukarıya
16 boşluk sabitlenmelidir.

Color anaRenk

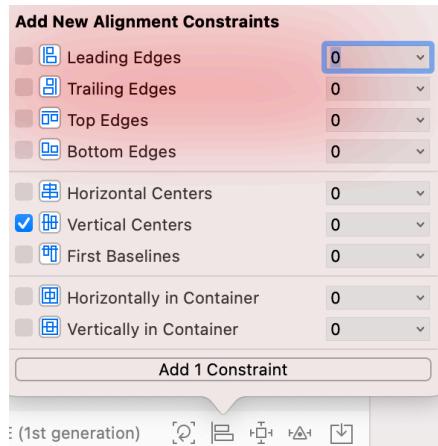
Font System Bold 38.0

Sol 16 , aşağı 32 boşluk

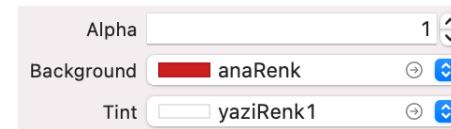
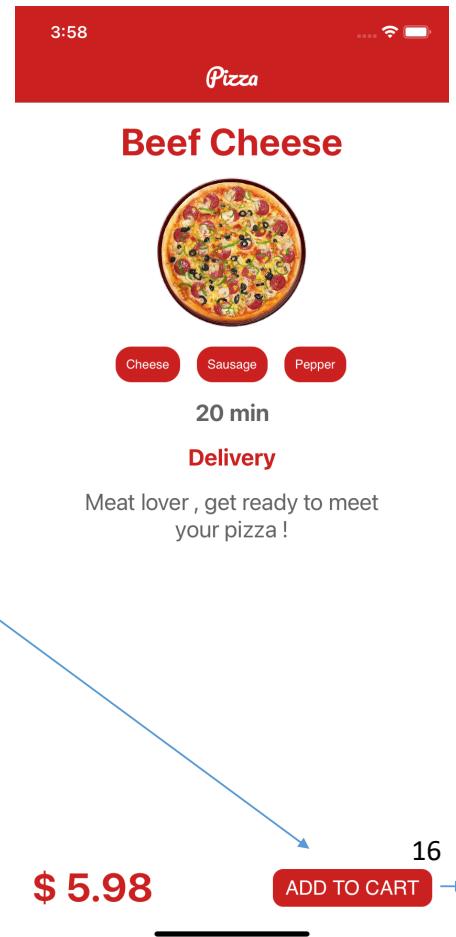
16
\$ 5.98
32

ADD TO CART

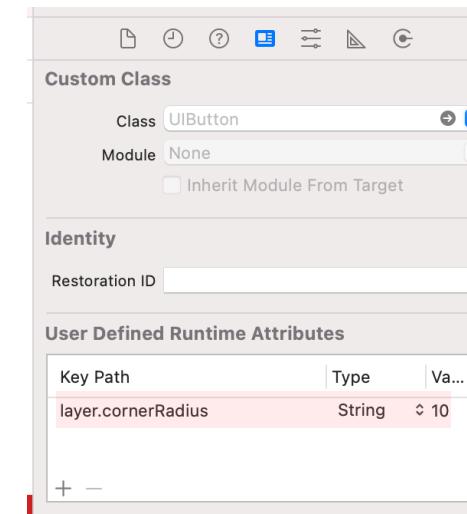
Fiyat ve Buttonu seçip
dikeyde hızlamalısın.



Ardından fiyatı sağa 16
boşluk sabitlenmelidir.



Button'a iç boşluk vermeye gerek yok direkt kıvrım verilebilir.



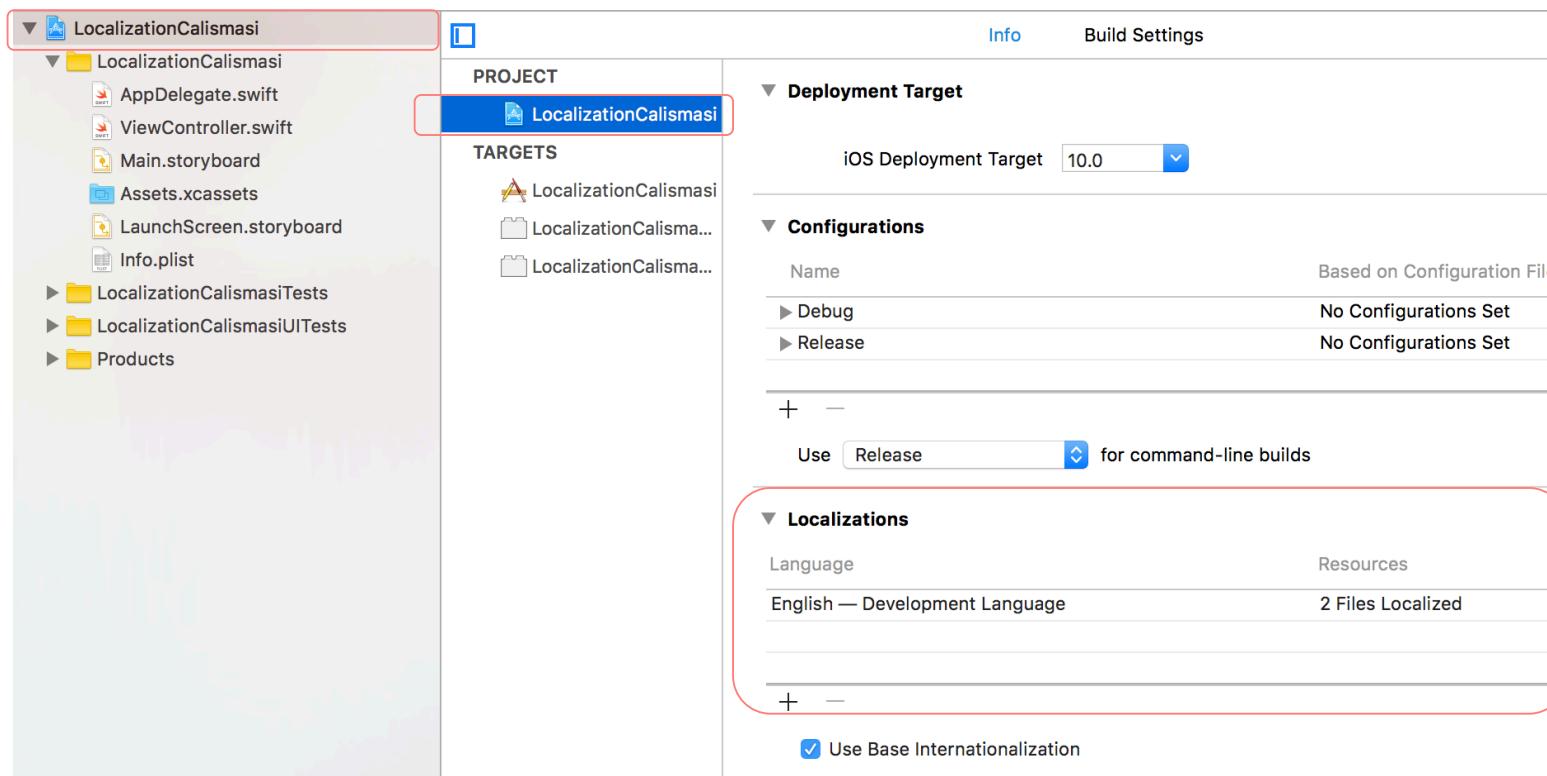
Çoklu Dil Desteği

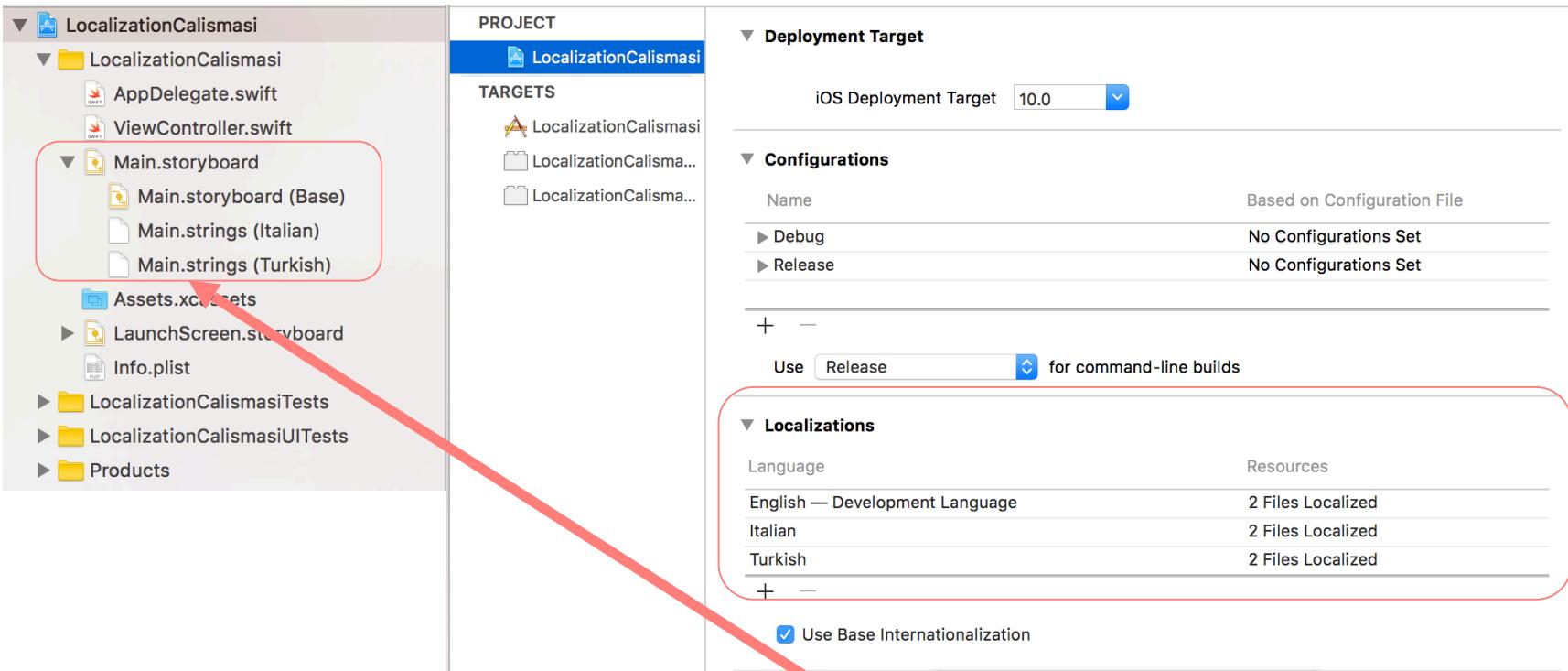
Çoklu Dil Desteği - (Localization)

- Uygulama ara yüzünün bir çok dile destek verebilmesini sağlayan bir yapıdır.
- Uygulama hangi cihazda açılırsa cihazın sistem diline göre otomatik olarak ara yüz değişecektir.



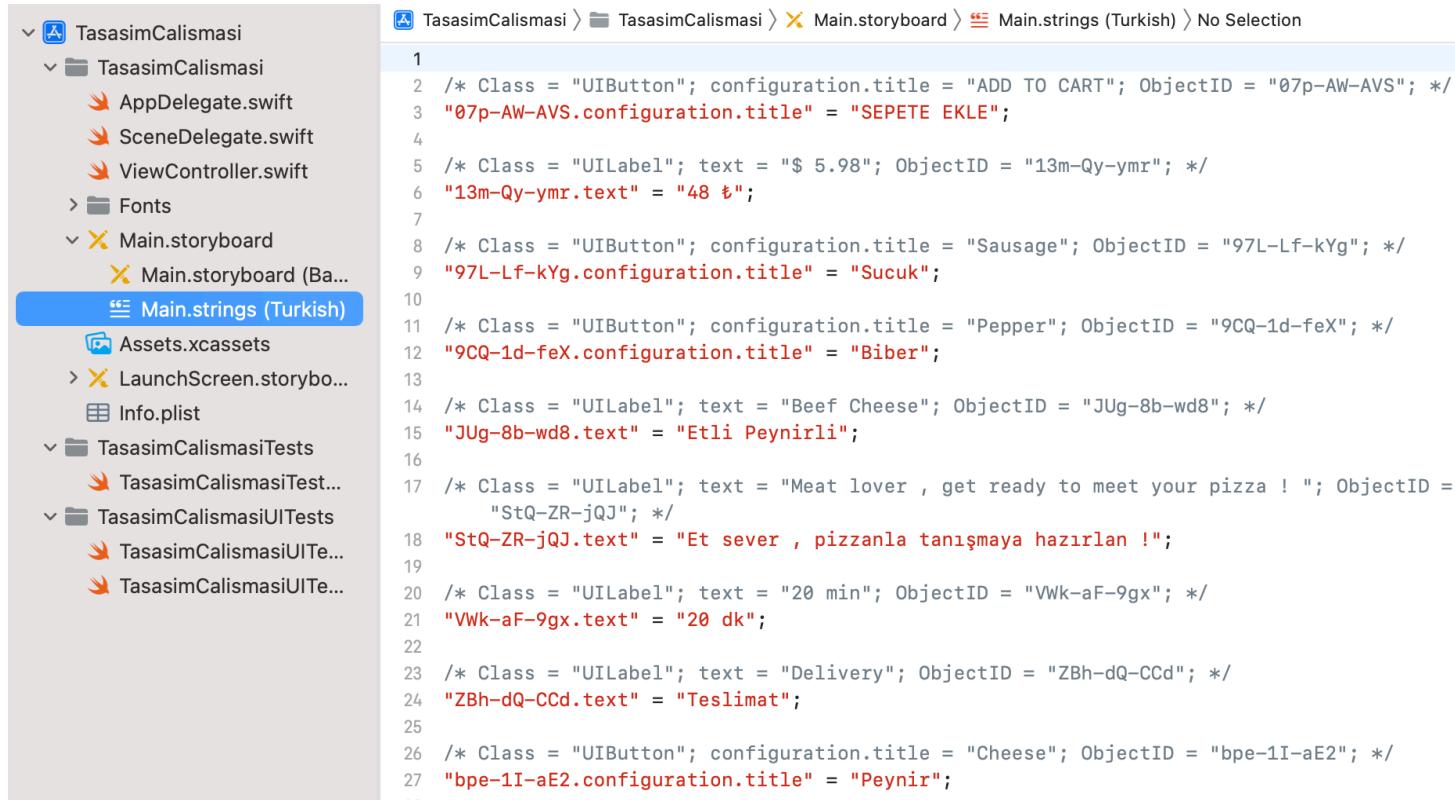
Dil Dosyaları Ekleme





Eklenen diller için string sınıfları oluşturulur.

Dil Sınıfı Yapısı



The screenshot shows the Xcode interface with the project structure on the left and the content of the Main.strings (Turkish) file on the right.

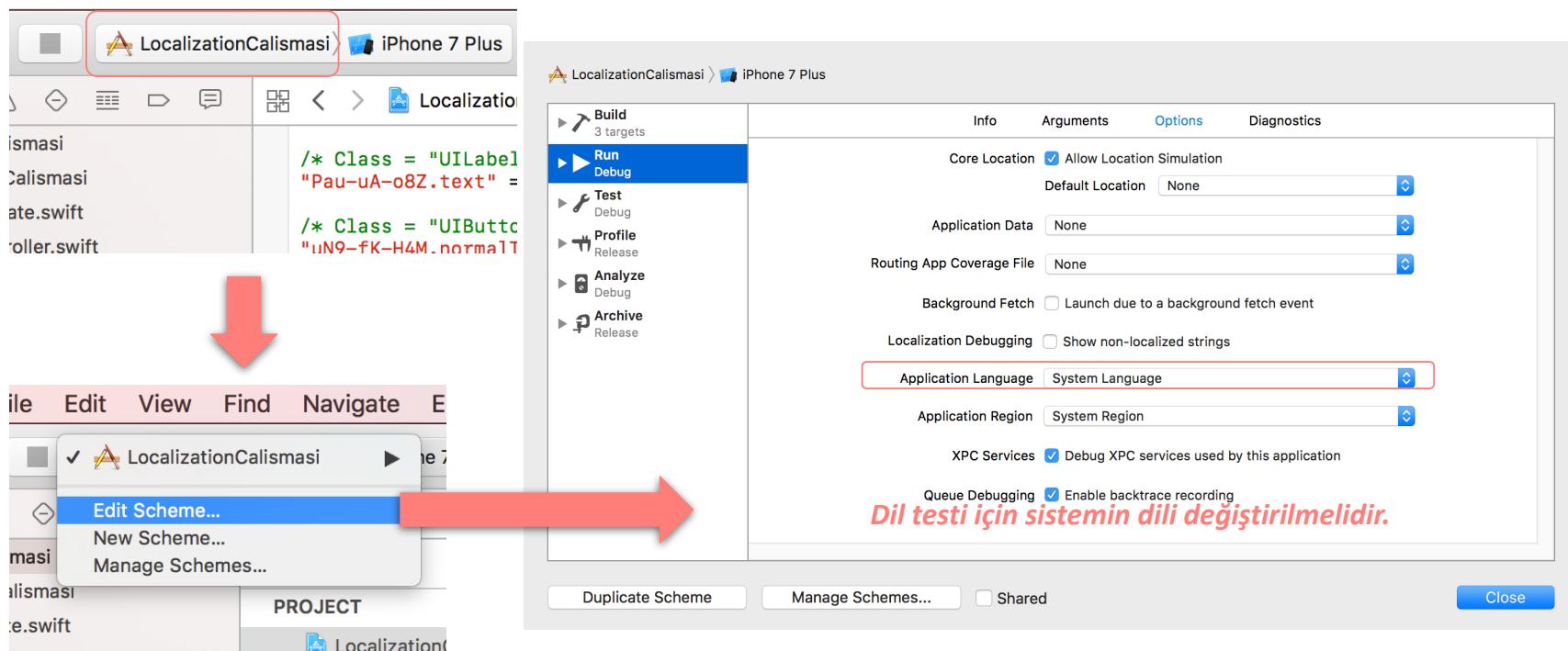
Project Structure:

- TasarimCalismasi
- ↳ TasarimCalismasi
 - AppDelegate.swift
 - SceneDelegate.swift
 - ViewController.swift
- ↳ Fonts
- ↳ Main.storyboard
 - Main.storyboard (Background)
 - Main.strings (Turkish) **(Selected)**
- ↳ Assets.xcassets
- ↳ LaunchScreen.storyboard
- Info.plist
- ↳ TasarimCalismasiTests
 - TasarimCalismasiTest...
- ↳ TasarimCalismasiUITests
 - TasarimCalismasiUITe...
 - TasarimCalismasiUITe...

Main.strings (Turkish) Content:

```
1 /* Class = "UIButton"; configuration.title = "ADD TO CART"; ObjectID = "07p-AW-AVS"; */
2 "07p-AW-AVS.configuration.title" = "SEPETE EKLE";
3
4 /* Class = "UILabel"; text = "$ 5.98"; ObjectID = "13m-Qy-ymr"; */
5 "13m-Qy-ymr.text" = "48 ₺";
6
7 /* Class = "UIButton"; configuration.title = "Sausage"; ObjectID = "97L-Lf-kYg"; */
8 "97L-Lf-kYg.configuration.title" = "Sucuk";
9
10 /* Class = "UIButton"; configuration.title = "Pepper"; ObjectID = "9CQ-1d-feX"; */
11 "9CQ-1d-feX.configuration.title" = "Biber";
12
13 /* Class = "UILabel"; text = "Beef Cheese"; ObjectID = "JUg-8b-wd8"; */
14 "JUg-8b-wd8.text" = "Etli Peynirli";
15
16 /* Class = "UILabel"; text = "Meat lover , get ready to meet your pizza ! "; ObjectID =
17 "StQ-ZR-jQJ"; */
18 "StQ-ZR-jQJ.text" = "Et sever , pizzanla tanışmaya hazırlan !";
19
20 /* Class = "UILabel"; text = "20 min"; ObjectID = "VWk-aF-9gx"; */
21 "VWk-aF-9gx.text" = "20 dk";
22
23 /* Class = "UILabel"; text = "Delivery"; ObjectID = "ZBh-dQ-CCd"; */
24 "ZBh-dQ-CCd.text" = "Teslimat";
25
26 /* Class = "UIButton"; configuration.title = "Cheese"; ObjectID = "bpe-1I-aE2"; */
27 "bpe-1I-aE2.configuration.title" = "Peynir";
28
```

Test edilmesi



Sonuç Görmek için uygulamanın emülatörde çalıştırılması gereklidir



Beef Cheese



Cheese Sausage Pepper

20 min

Delivery

Meat lover , get ready to meet
your pizza !

\$ 5.98

ADD TO CART



Eti Peynirli



Peynir Sucuk Biber

20 dk

Teslimat

Et sever , pizzanla
tanışmaya hazırlan !

48 ₺

SEPETE EKLE

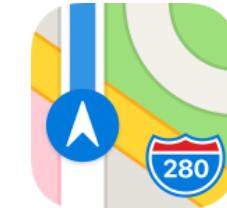
Icon Oluşturma

App Icon Özellikleri

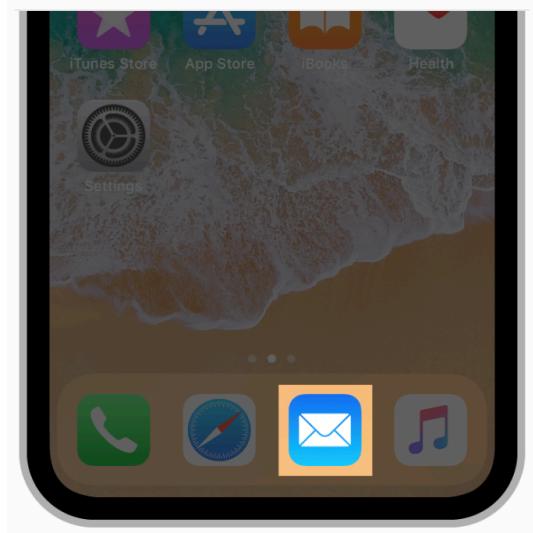
Format : PNG

Color Space : sRGB

Şekil : Köşeleri oval kare görünüm



Nerelerde Kullanılır





Boyut Standartları



Cihazlar için

Device or context	Icon size
iPhone	Sonuç 180px × 180px (60pt × 60pt @3x) 120px × 120px (60pt × 60pt @2x)
iPad Pro	167px × 167px (83.5pt × 83.5pt @2x)
iPad, iPad mini	152px × 152px (76pt × 76pt @2x)
App Store	1024px × 1024px (1024pt × 1024pt @1x)

Burada dikkat etmemiz gereken 2x 3x çarpanlarıdır.

Örn : 60 pt olan iPhone için 2x için 120px x 120px dir. 60 pt değeri oranlar ile çarpılır.

Boyut Standartları (Bildirim,Ayarlar vb.)



Device	Spotlight icon size
iPhone	120px × 120px (40pt × 40pt @3x) 80px × 80px (40pt × 40pt @2x)
iPad Pro, iPad, iPad mini	80px × 80px (40pt × 40pt @2x)



Device	Settings icon size
iPhone	87px × 87px (29pt × 29pt @3x) 58px × 58px (29pt × 29pt @2x)
iPad Pro, iPad, iPad mini	58px × 58px (29pt × 29pt @2x)

Device	Notification icon size
iPhone	60px × 60px (20pt × 20pt @3x) 40px × 40px (20pt × 20pt @2x)
iPad Pro, iPad, iPad mini	40px × 40px (20pt × 20pt @2x)

Yardımcı Siteler

- İcon Oluşturma için site
- <https://romannurik.github.io/AndroidAssetStudio/icons-launcher.html>
- İconu istenilen boyutlara dönüştürmek için site
- <https://appicon.co/>
- Oluşturulan icon çeşitli piksel boyutlarında oluşur ve isimleri boyutlarını belirtir.

	16.png
	20.png
	29.png
	32.png
	40.png
	48.png
	50.png
	55.png
	57.png
	58.png
	60.png
	64.png
	72.png
	76.png
	80.png
	87.png
	88.png
	100.png
	114.png
	120.png

Icon'un Assets.xcassets içine yerleştirme

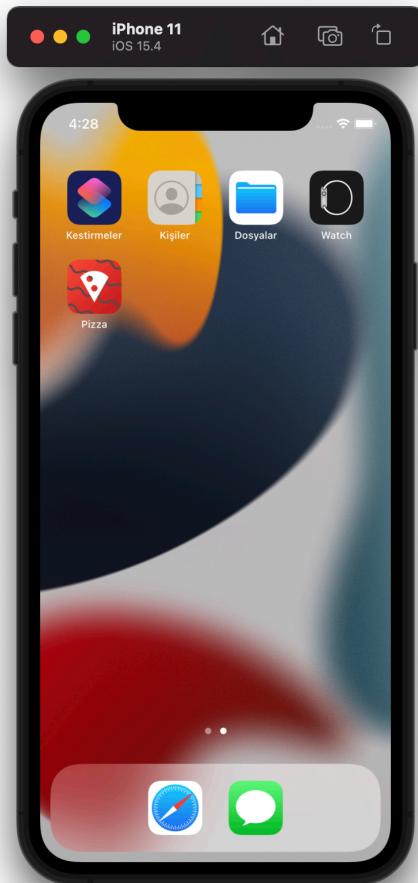


- 16.png
- 20.png
- 29.png
- 32.png
- 40.png
- 48.png
- 50.png
- 55.png
- 57.png
- 58.png
- 60.png
- 64.png
- 72.png
- 76.png
- 80.png
- 87.png
- 88.png
- 100.png
- 114.png
- 120.png

İlgili boyutları
gerekli alanlara
sürüklenmelidir.



Uygulama Adını Düzenleme



Uygulama Adı Düzenleme

The screenshot shows the Xcode interface with the project 'TasarimCalismasi' selected. The left sidebar lists files and folders: AppDelegate.swift, SceneDelegate.swift, ViewController.swift, Fonts (with Pacifico-Regular.ttf), Main.storyboard (marked with a red X), Assets.xcassets, LaunchScreen.storyboard, Info.plist, TasarimCalismasiTests, and TasarimCalismasiUITests. The right pane shows the 'General' tab of the project settings. Under the 'Identity' section, the 'Display Name' is set to 'Pizza', which is highlighted with a pink rectangle. Other fields include 'Bundle Identifier' (com.info.TasarimCalismasi), 'Version' (1.0), and 'Build' (1). Under the 'Deployment Info' section, 'iOS 15.4' is selected, and 'iPhone' and 'iPad' are checked, while 'Mac Catalyst' is unchecked.

Dark Mode

Kullanılacak Renkler



Oluşturduğumuz dark mode renkleri otomatik çalışmaktadır.

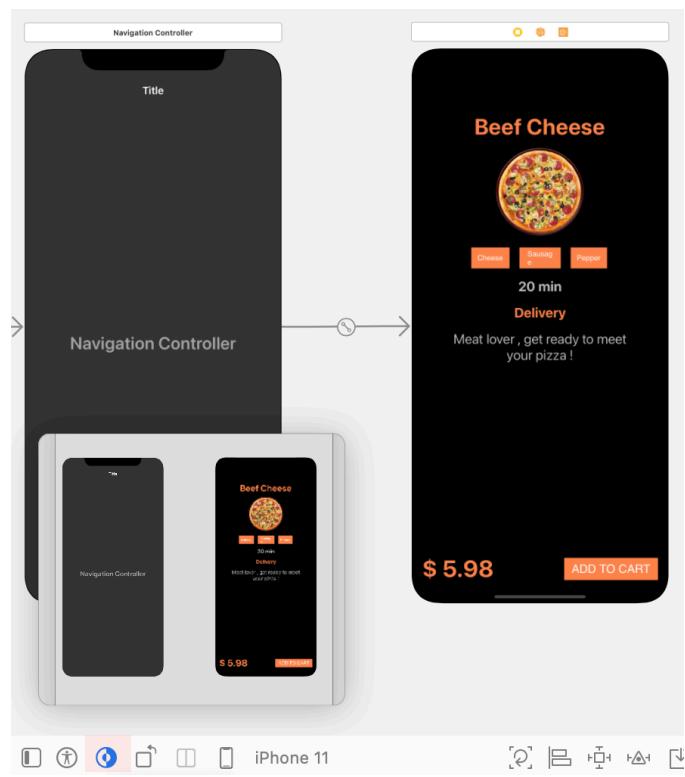


Her iki durumda aynı renk olabilir.

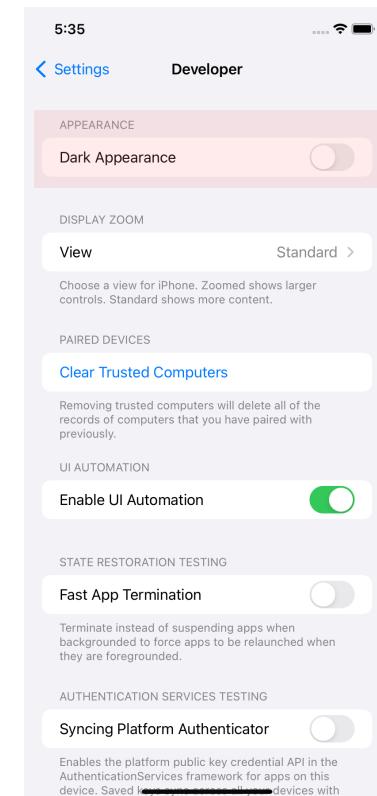


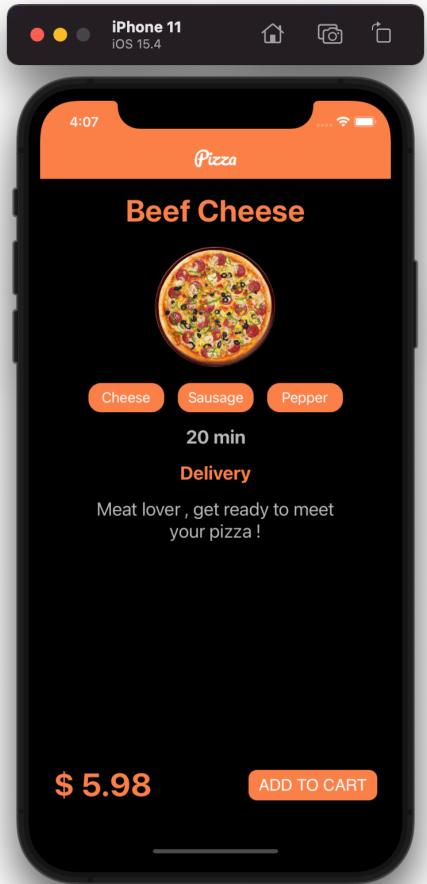
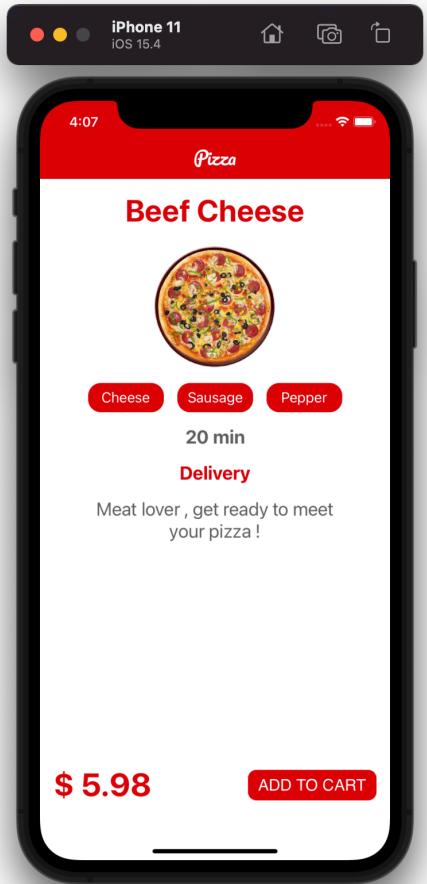
Dark Mode Test

Storyboard üzerinde



Simulatör üzerinde dark mode aktif edilmelidir.

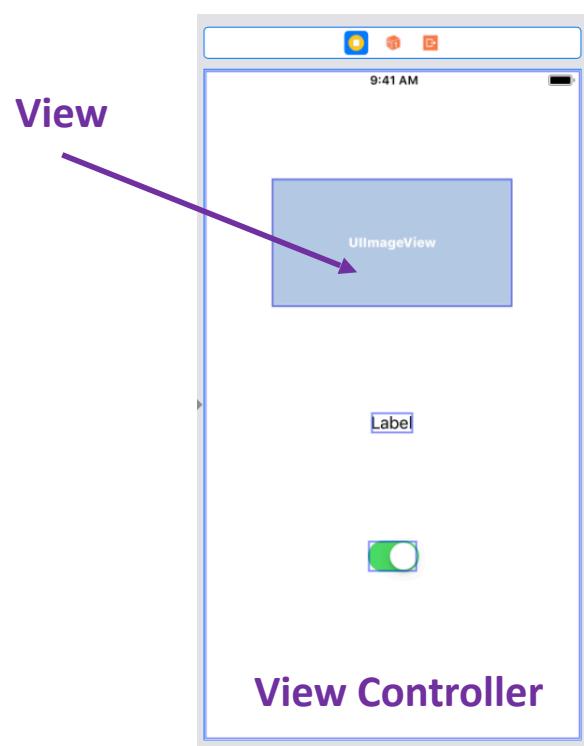




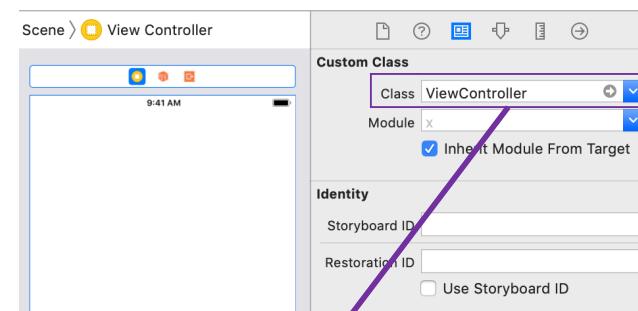
IOS Çalışma Yapısı

IOS Sayfa Yapısı

TASARIM



SWIFT SINIFI

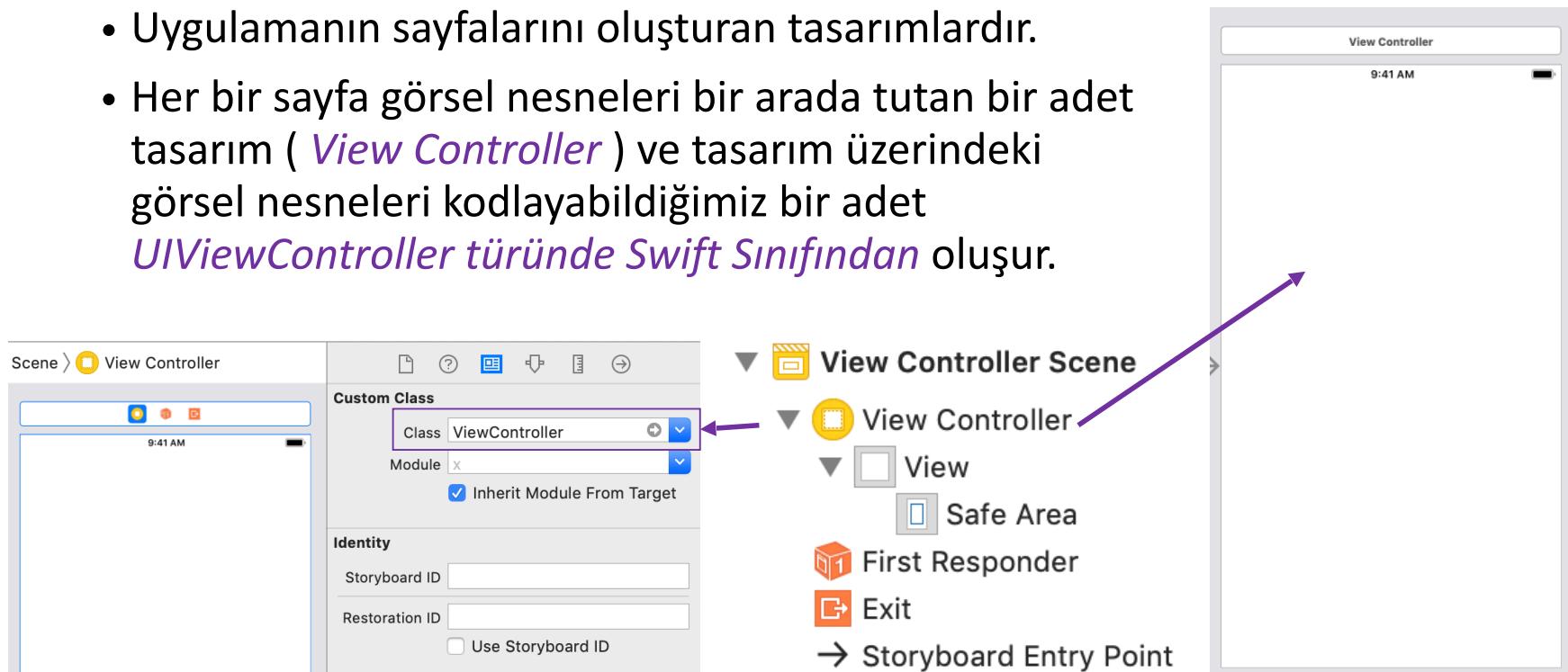


```
import UIKit  
class ViewController: UIViewController {  
    override func viewDidLoad() {  
        super.viewDidLoad()  
        // Do any additional setup after  
        // loading the view, typically from a  
        // nib.  
    }  
}
```

Not : Sınıf **UIViewController** türünde olmalıdır.

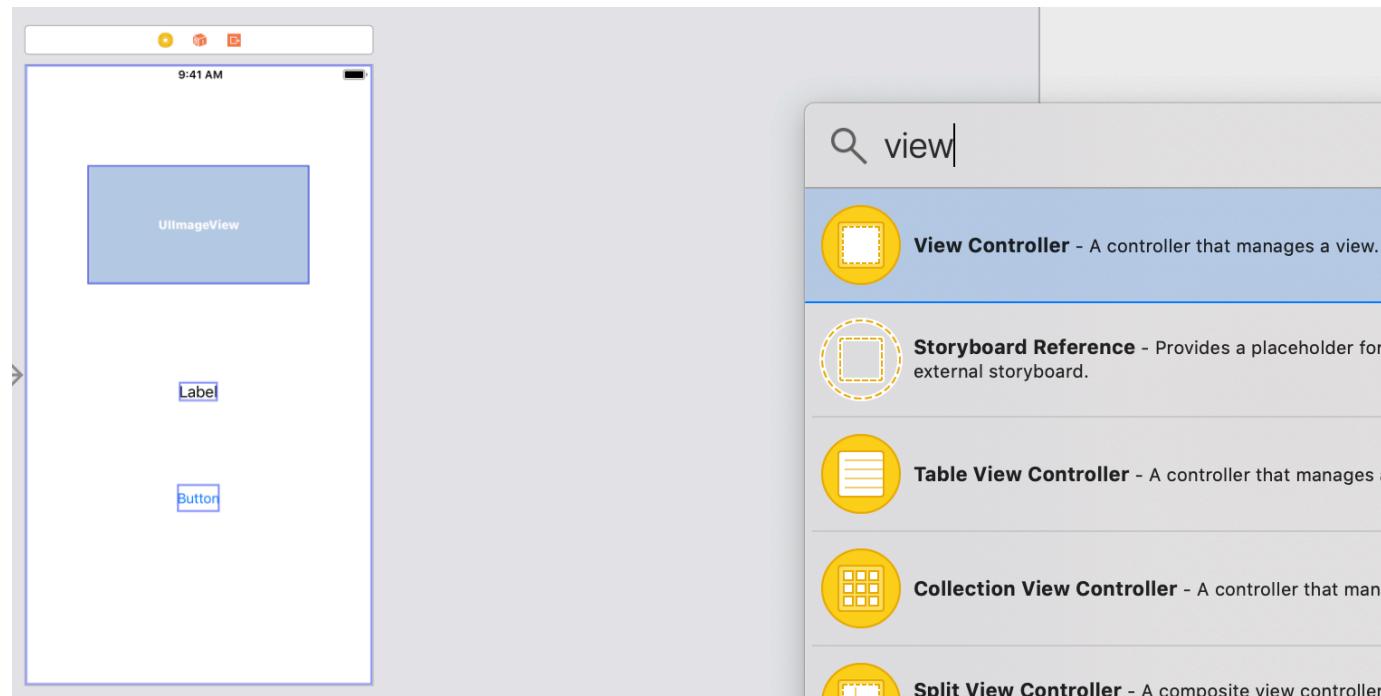
View Controller

- Uygulamanın sayfalarını oluşturan tasarımlardır.
- Her bir sayfa görsel nesneleri bir arada tutan bir adet tasarım (*View Controller*) ve tasarım üzerindeki görsel nesneleri kodlayabildiğimiz bir adet *UIViewController* türünde *Swift Sınıfından* oluşur.

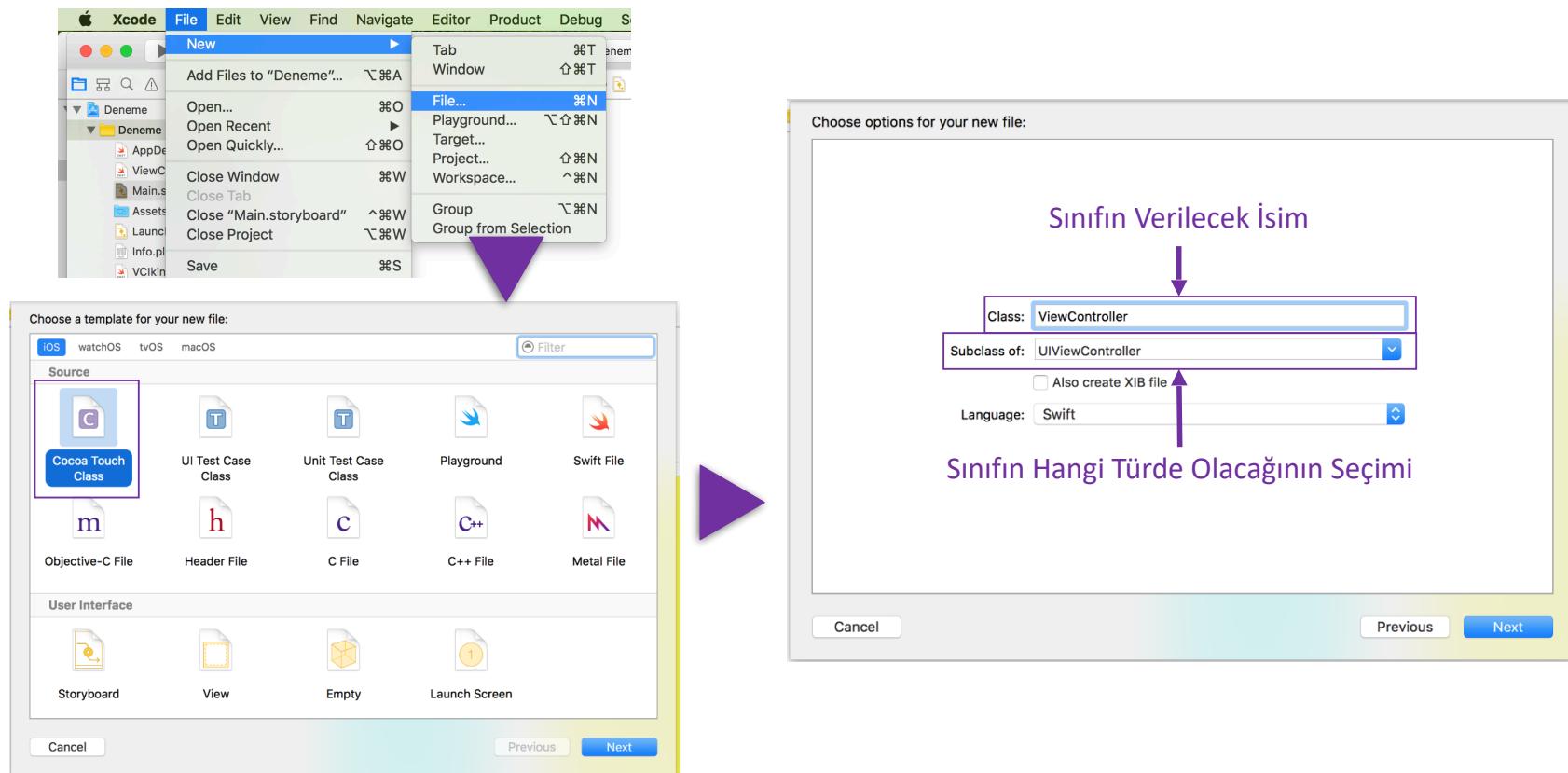


Yeni Bir Sayfa Oluşturma

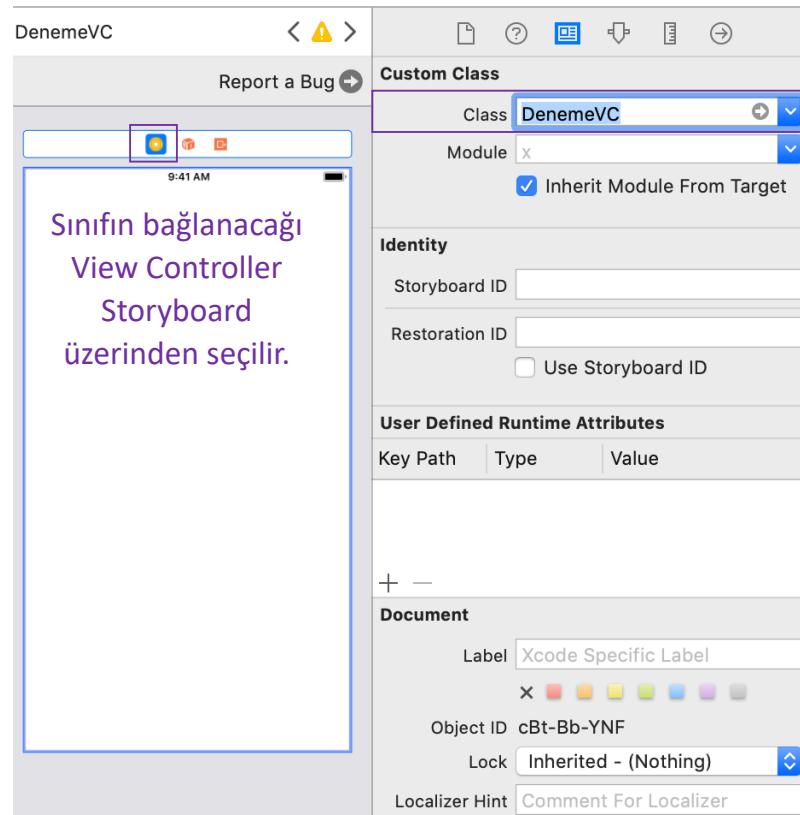
Storyboard üzerinde Yeni bir View Controller Oluşturulur.



View Controller için UIViewController türünde Swift Sınıf Oluşturulur.

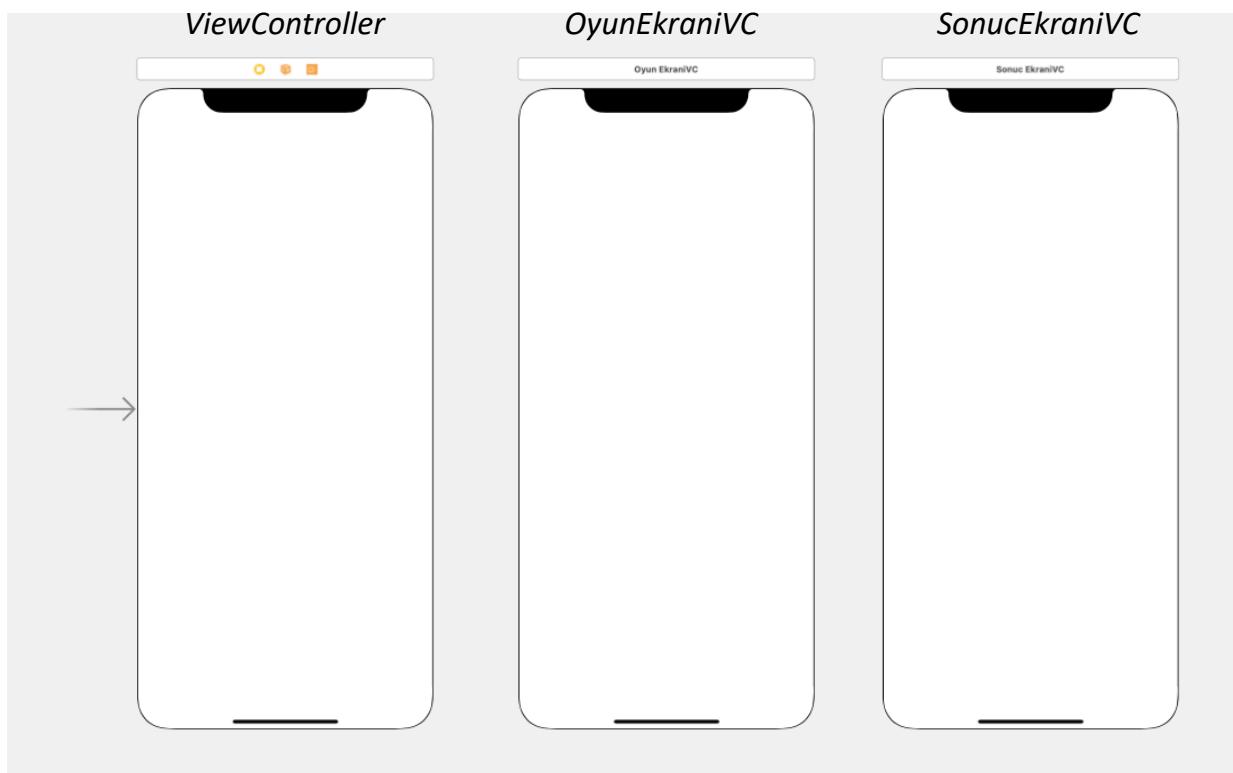


Oluşturulan Sınıfın View Controller'a Bağlanması



Bağlanacak sınıf bulunur ve bu alanada yerleştirilir.

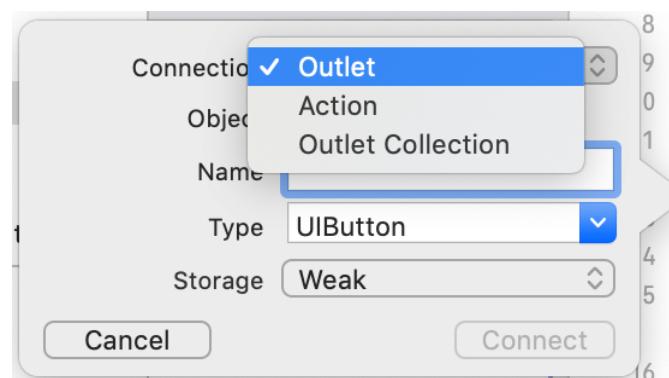
Sayfa Oluşturma



Görsel Nesnelerin Swift Sınıfına Bağlanması

Görsel Nesnenin Bağlantı Türleri

- Outlet : Görsel Nesnenin Swift Sınıfına **DEĞİŞKEN** olarak bağlanması.
- Action : Görsel Nesnenin Swift Sınıfına **AKSIYON (METOD)** olarak bağlanması.
- Outlet Collection : Görsel Nesnenin Swift Sınıfına **DEĞİŞKEN KÜMESİ** olarak bağlanması.



Görsel Nesne Bağlama

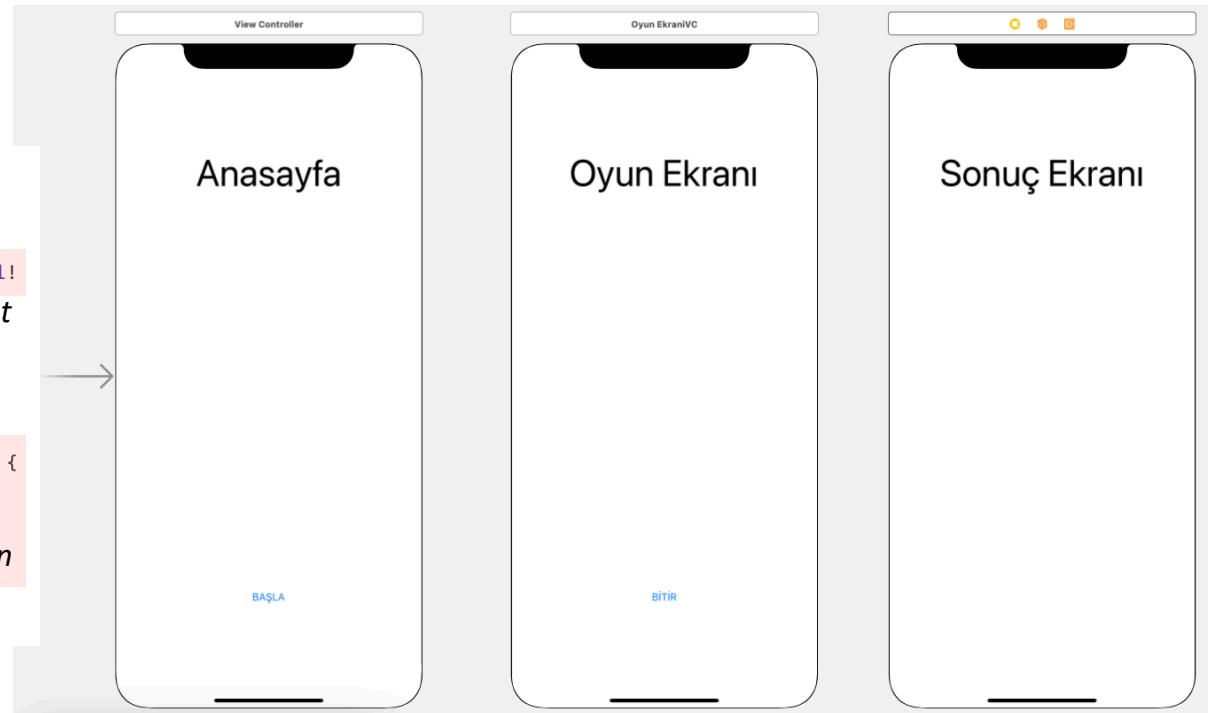
```
import UIKit

class ViewController: UIViewController {

    @IBOutlet weak var labelAnasayfa: UILabel!
    Outlet

    override func viewDidLoad() {
        super.viewDidLoad()
    }

    @IBAction func baslaTikla(_ sender: Any) {
        Action
        labelAnasayfa.text = "Merhaba"
    }
}
```



Label size : 50

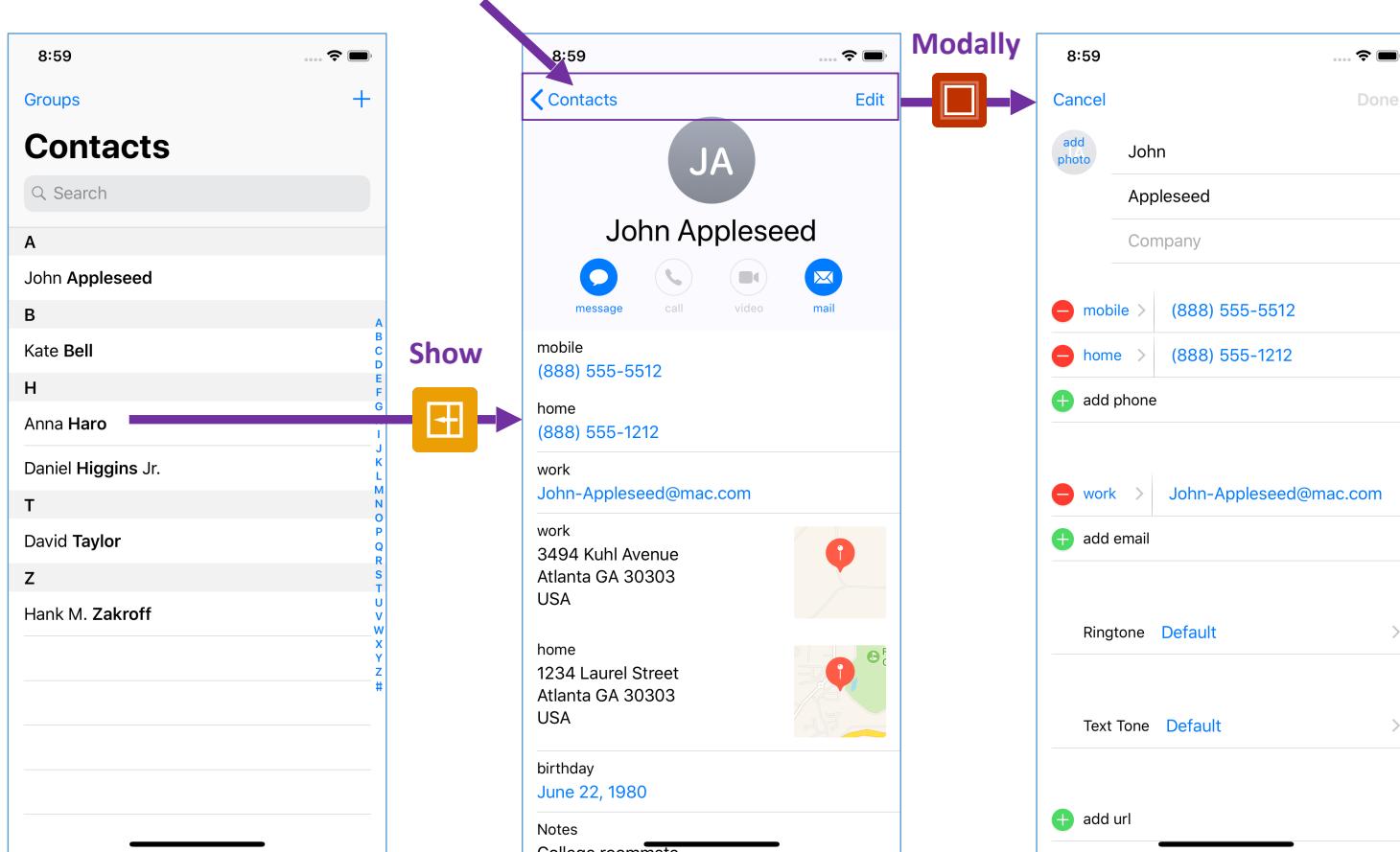
Yukarı ve aşağı mesafeler : 100

Present Modally

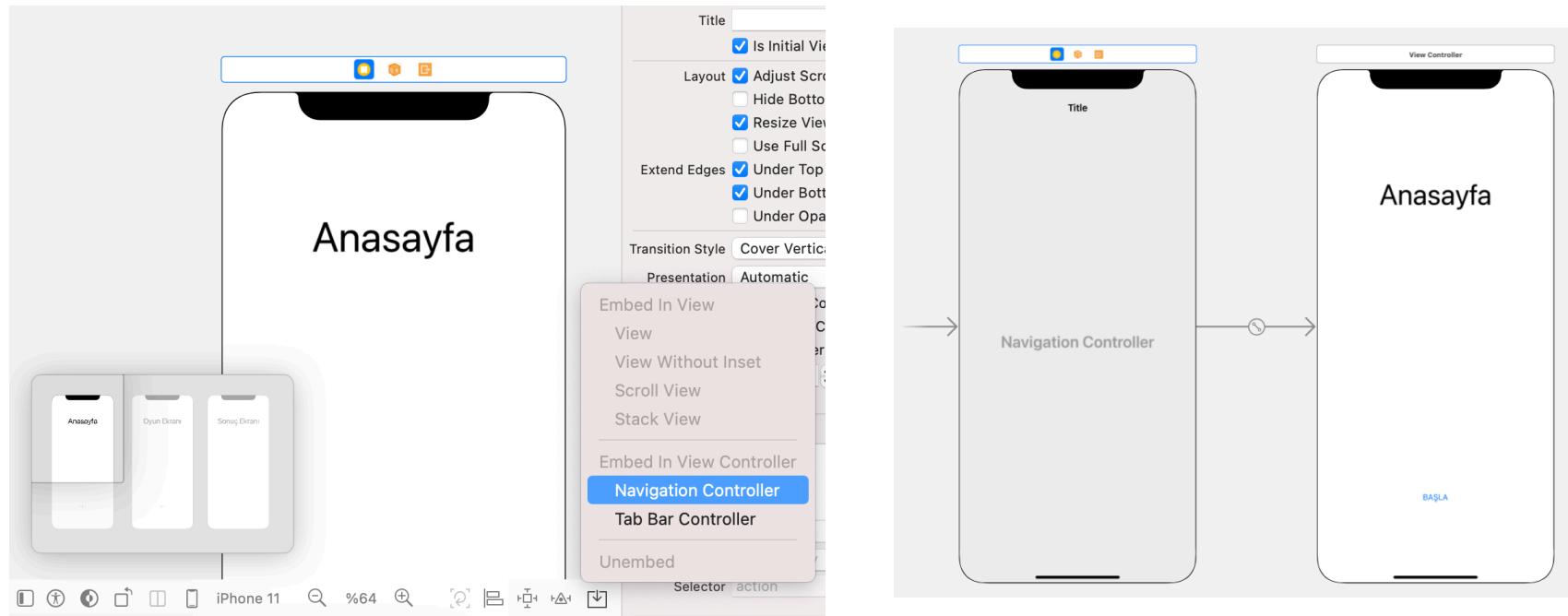
vs

Show (Push)

Show segue sayesinde otomatik geri buttonu

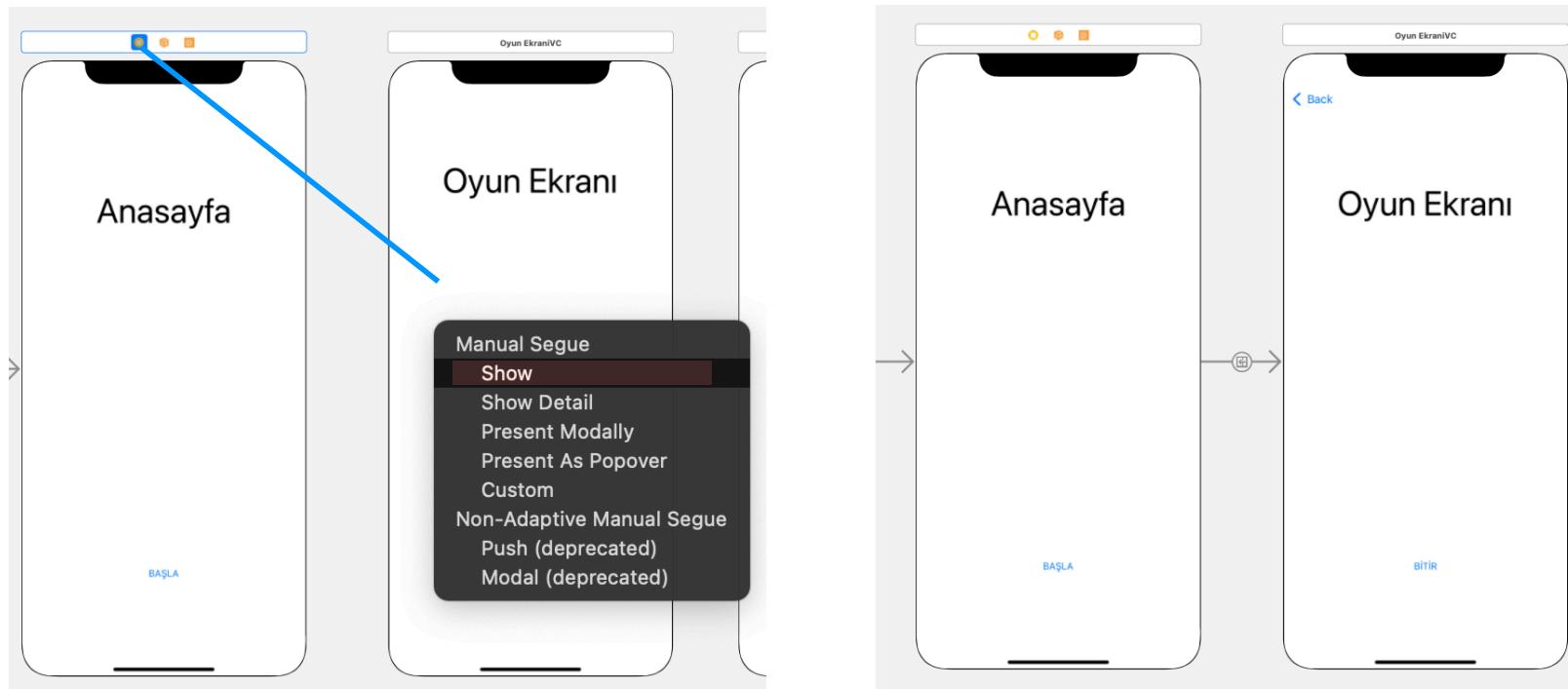


Navigation Controller Ekleme



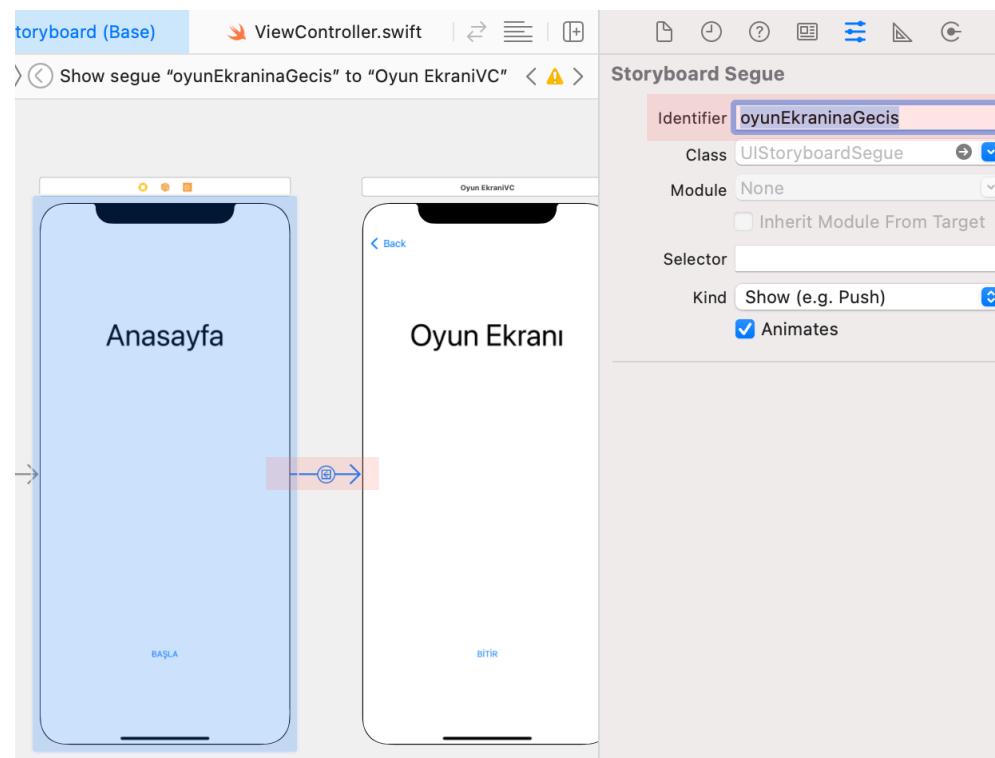
Show Geçişi Oluşturma

- Geçişi sayfa geneli kullanması için sayfadan sayfaya tanımlanmalıdır.



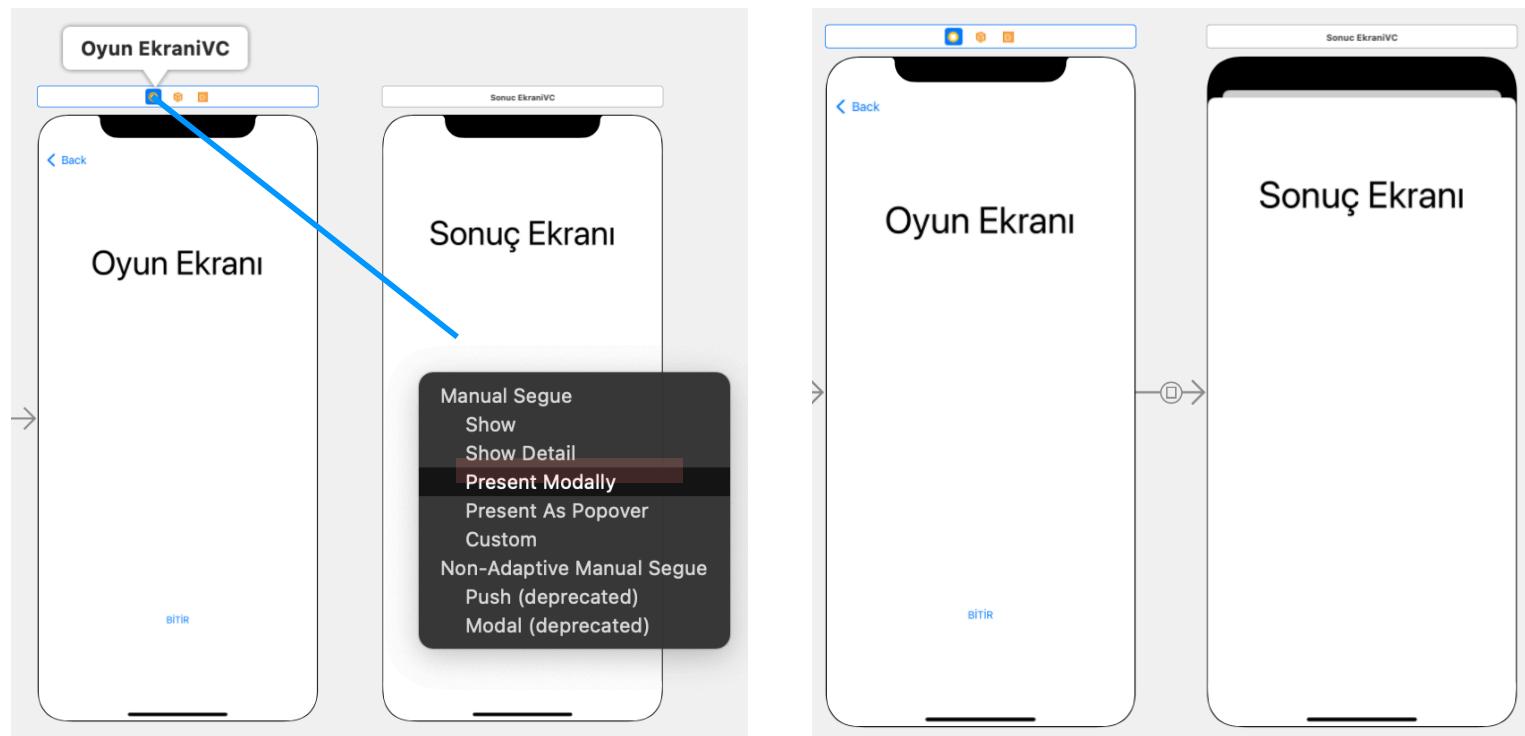
Show Geçişe ID verme

- Geçişe id verme



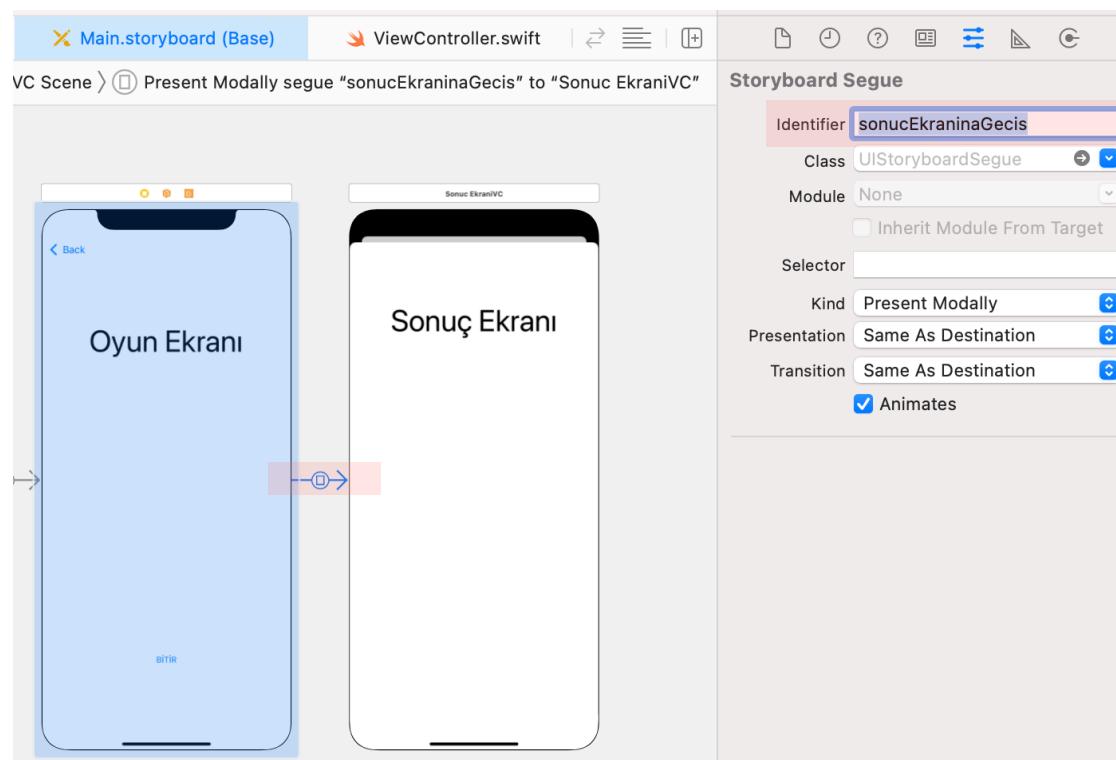
Present Modally Geçiş Oluşturma

- Geçışı sayfa geneli kullanması için sayfadan sayfaya tanımlanmalıdır.



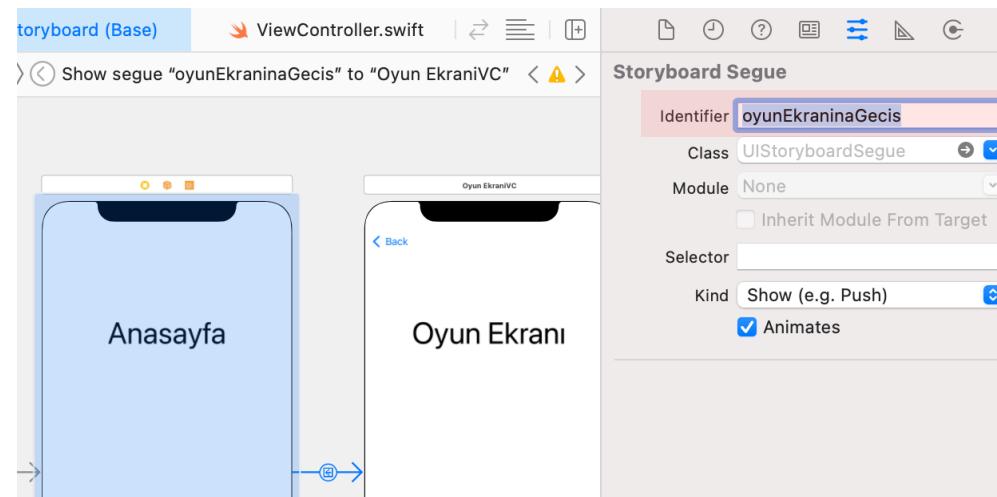
Present Modally Geçişe ID verme

- Geçişe id verme



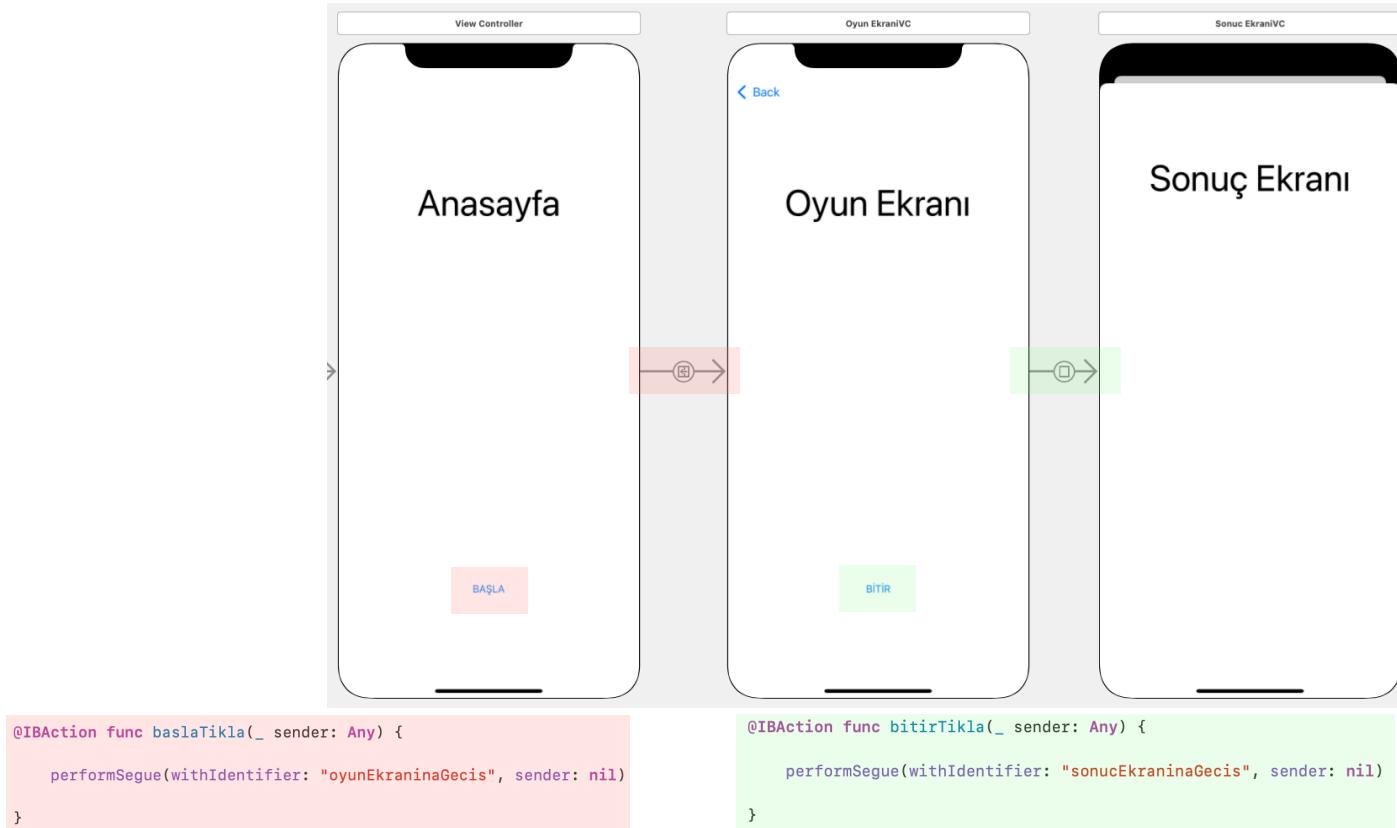
Geçişleri Tetikleme

- Storyboard üzerindeki geçişleri tetiklemek için `performSegue` metodu kullanılır.



```
performSegue(withIdentifier: "oyunEkranaGecis", sender: nil)  
Geçiş ID'si
```

Sayfa Geçişi

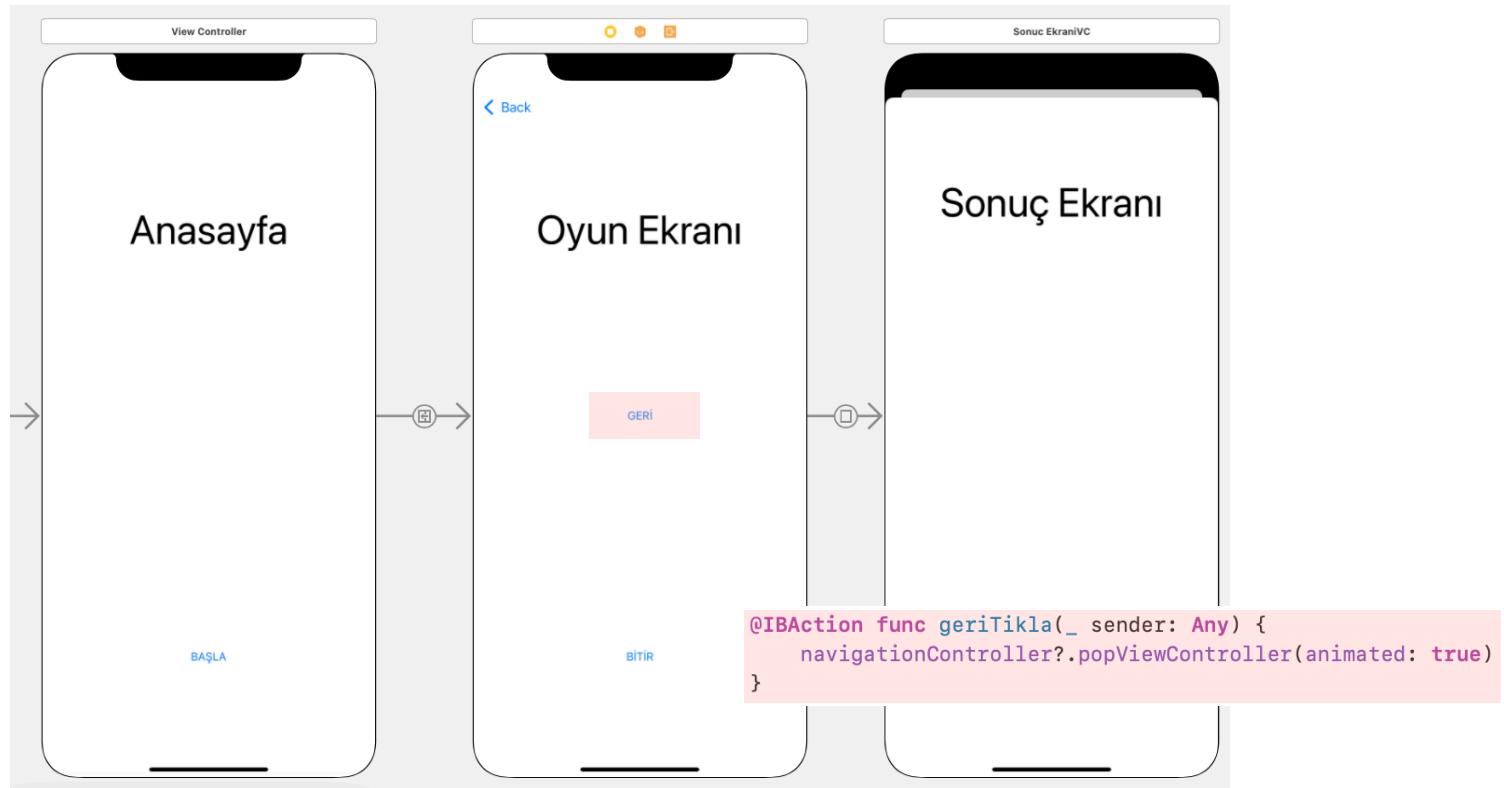


Kodlama ile GERİ gelme

- Show segue geçişleri görsel geriye gibi görünüşde hep bir sonraki view controller'ı açma üzerindedir.
- Hangi sayfadan gelinmiş ise o sayfaya döner.
- Bu işlemi kodlama ile yapabiliriz.

```
@IBAction func geri(_ sender: Any) {  
    navigationController?.popViewController(animated: true)  
    //Bir önceki sayfaya navigation controller özelliği ile geri döner  
}
```

Kodlama İle Geri Gelme

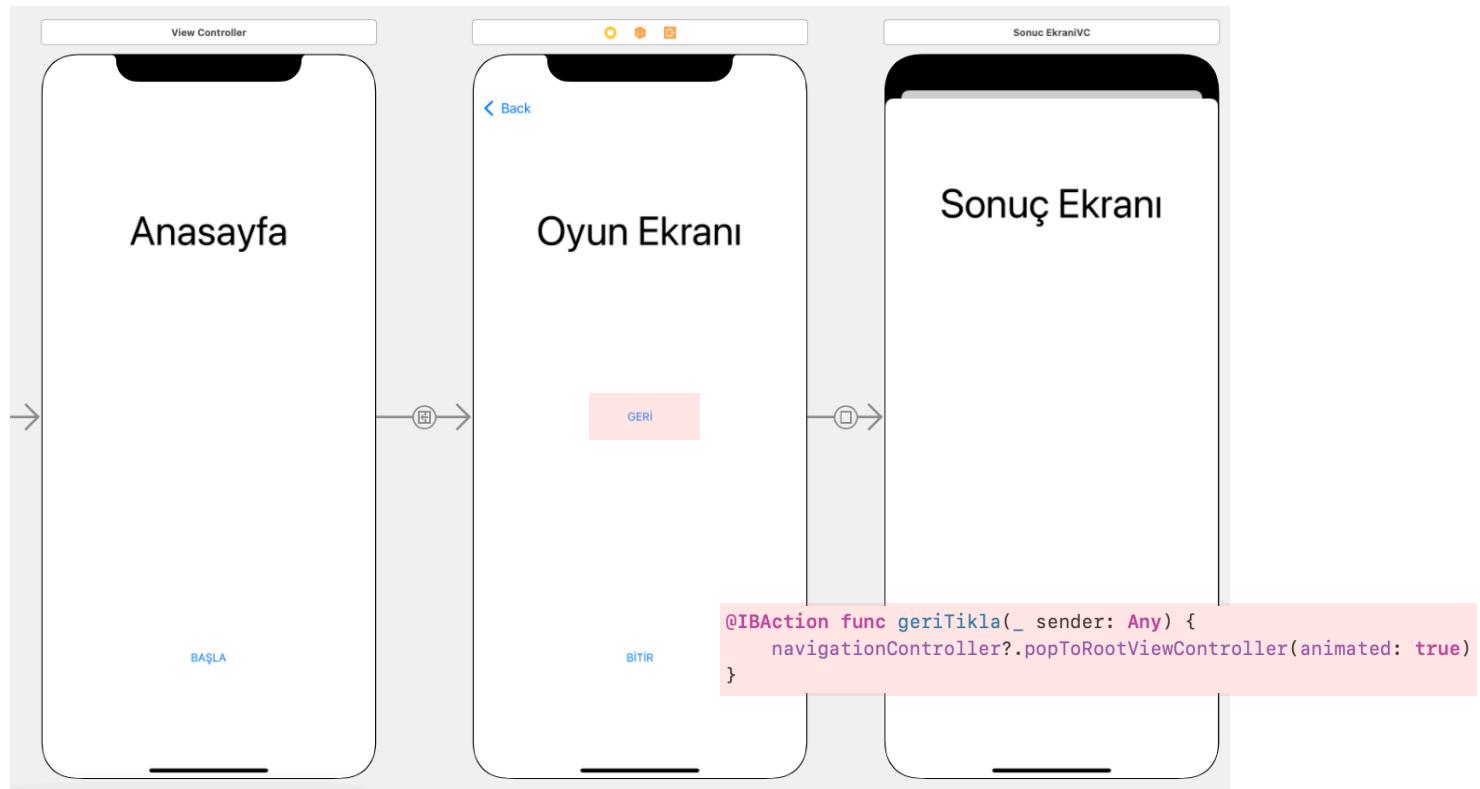


Kodlama ile EN BAŞA DÖN

- Hangi sayfa en baştaki ise ona dönülür.
- Bu işlemi kodlama ile yapabiliriz.

```
@IBAction func goto1(_ sender: Any) {  
    navigationController?.popToRootViewController(animated: true)  
    //En baştaki sayfaya döner  
}
```

Kodlama İle En Başa Dönme



Navigation Controller Back Tuşunu Gizleme

- Bazı durumlarda geçiş yapılan sayfadan bir önceki sayfaya dönmesini istemeyebiliriz.
- Bu durumda geri tuşunu gizleriz.

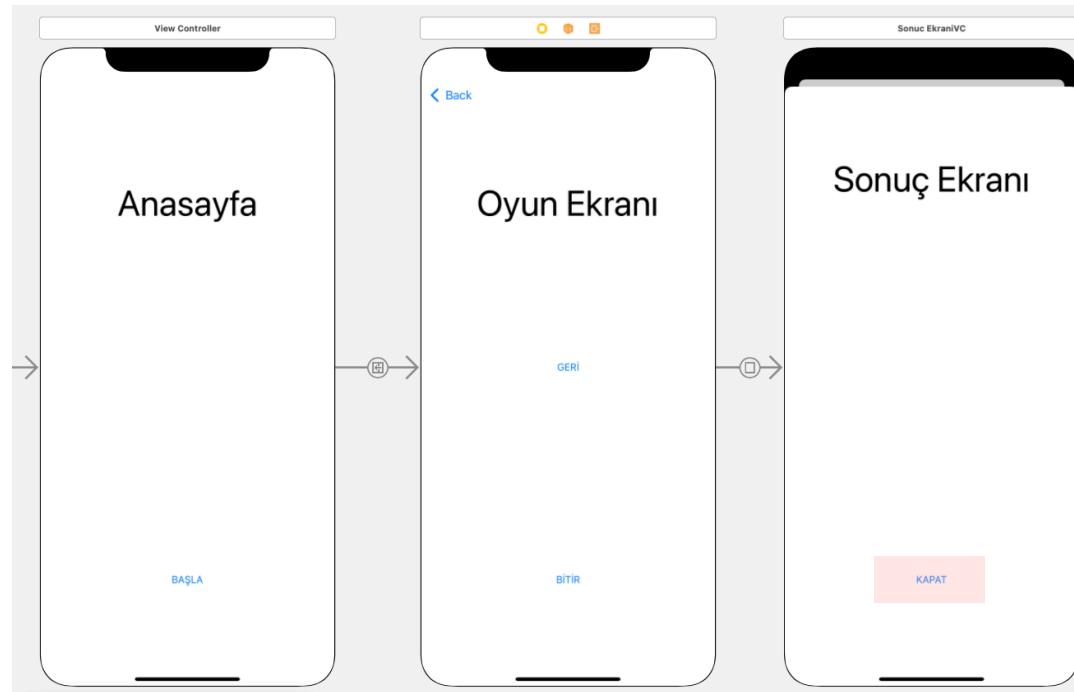
```
override func viewDidLoad() {  
    super.viewDidLoad()  
  
    self.navigationItem.hidesBackButton = true  
    //Bulunduğu sayfanın Geri dönüş tuşunu saklar  
}
```

Dismiss metodu

- Present modally geçişleri görsel geriye gibi görünserde hep bir sonraki view controller'ı açma üzerindedir.
- Bulunduğumuz sayfada geldiğimiz sayfaya dönmek istiyorsak yani GERİ dönüş dismiss metodu ile yapılır.
- Hangi sayfadan gelinmiş ise o sayfaya döner.
- Bu işlemi kodlama ile yapabiliriz.

```
@IBAction func tiklanDismiss(_ sender: Any) {  
    self.dismiss(animated: true, completion: nil)  
}
```

Dismiss Metodu Kullanımı



```
@IBAction func kapatTikla(_ sender: Any) {  
    self.dismiss(animated: true, completion: nil)  
}
```

Geçisi Dinleme

- Bir sayfada birden fazla geçiş olabilir ve bu şekilde takip edilebilir.

```
@IBAction func baslaTikla(_ sender: Any) {  
  
    performSegue(withIdentifier: "oyunEkraninaGecis", sender: nil)  
  
}  
  
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {  
    print("prepare metodу çalıştırıldı")  
  
    if segue.identifier == "oyunEkraninaGecis" { Geçiş idsini yakalama  
        print("oyunEkraninaGecis çalıştırıldı")  
    }  
}
```

Geçişe Veri Aktarma

```
@IBAction func baslaTikla(_ sender: Any) {          Bütün türde veri aktarılabilir.  
    performSegue(withIdentifier: "oyunEkraninaGecis", sender: "merhaba")  
}  
  
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {  
    print("prepare metodu çalıştı")  
  
    if segue.identifier == "oyunEkraninaGecis" {  
        print("oyunEkraninaGecis çalıştı")  
  
        if let veri = sender as? String {  
            print("Veri : \(veri)")      Hangi türse ona dönüştürmeliyiz  
        }  
    }  
}
```

Veri Transferi

- Bu yöntem hem Show hem Present Modally için geçerlidir.

Gönderen

```
@IBAction func baslaTikla(_ sender: Any) {  
  
    performSegue(withIdentifier: "oyunEkraninaGecis", sender: "merhaba")  
}  
  
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {  
    print("prepare metodу çalışты")  
  
    if segue.identifier == "oyunEkraninaGecis" {  
        print("oyunEkraninaGecis çalışты")  
  
        if let veri = sender as? String {  
            print("Veri : \(veri)")  
  
            let gidilecekVC = segue.destination as! OyunEkranıVC  
            gidilecekVC.mesaj = veri  
        }  
    }  
}
```

Alan

```
import UIKit  
  
class OyunEkranıVC: UIViewController {  
  
    var mesaj:String?  
  
    override func viewDidLoad() {  
        super.viewDidLoad()  
  
        self.navigationItem.hidesBackButton = true  
  
        if let m = mesaj {  
            print("Gelen mesaj : \(m)")  
        }  
    }  
}
```

Veri Transferi - Nesne Tabanlı

- Bu yöntem hem Show hem Present Modally için geçerlidir.

```
class Kisiler{
    var ad:String?
    var yas:Int?
    var boy:Double?
    var bekar:Bool?

    init(){

    }

    init(ad:String,yas:Int,boy:Double,bekar:Bool){
        self.ad = ad
        self.yas = yas
        self.boy = boy
        self.bekar = bekar
    }
}
```

Veri Transferi - Nesne Tabanlı

- Bu yöntem hem Show hem Present Modally için geçerlidir.

Gönderen

```
@IBAction func buttonBaslaTikla(_ sender: Any) {
    let kisi = Kisiler(ad: "Ahmet", yas: 23, boy: 1.78, bekar: true)
    performSegue(withIdentifier: "oyunEkranıGecis", sender: kisi)
}

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    print("prepare metodu çalıştı")

    if segue.identifier == "oyunEkranıGecis" {
        print("oyunEkranıGecis çalıştı")

        if let veri = sender as? Kisiler {
            let gidilecekVC = segue.destination as! OyunEkranıVC
            gidilecekVC.kisi = veri
        }
    }
}
```

Alan

```
class OyunEkranıVC: UIViewController {

    var kisi:Kisiler?

    override func viewDidLoad() {
        super.viewDidLoad()

        if let k = kisi {
            print("Kişi ad : \(k.ad!)")
            print("Kişi yaş : \(k.yas!)")
            print("Kişi boy : \(k.boy!)")
            print("Kişi bekar : \(k.bekar!)")
        }
    }
}
```

IOS Yaşam Döngüsü

UIViewController Yaşam Döngüsü



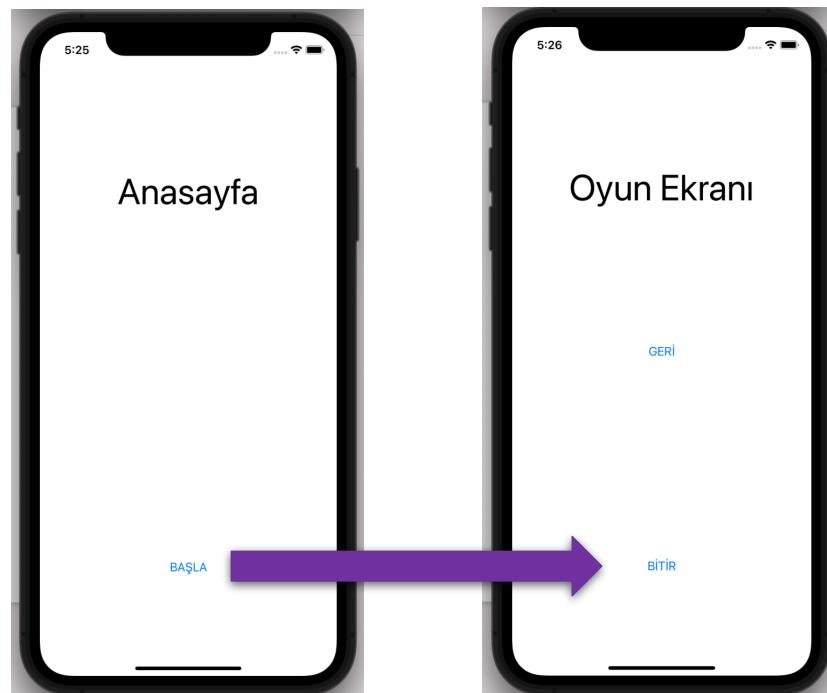
Çalışma Senaryoları

Uygulama veya Sayfa İlk Açıldığı Anda



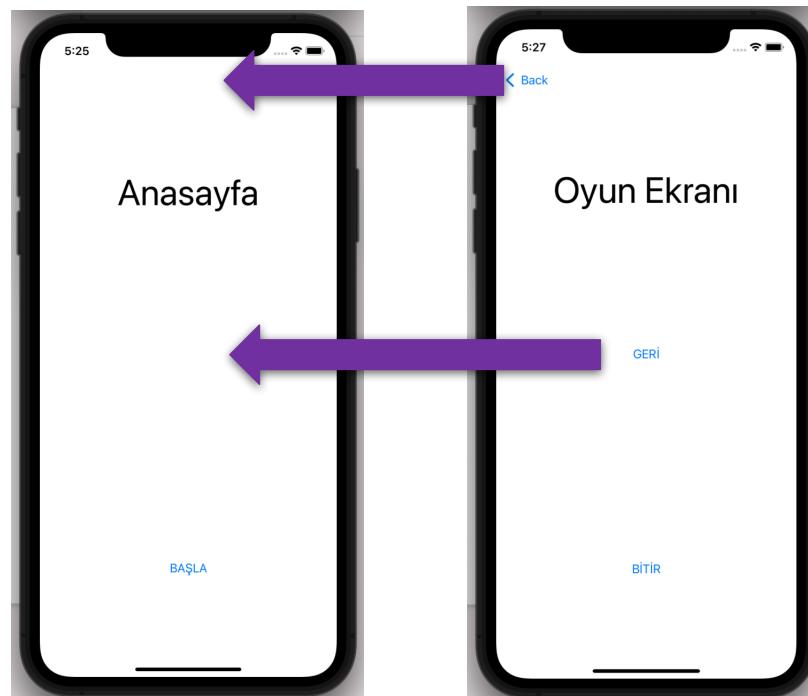
viewDidLoad çalıştı
viewWillAppear çalıştı
viewDidAppear çalıştı

Başka bir sayfaya geçiş olduğunda



**viewWillDisappear çalıştı
viewDidDisappear çalıştı**

Başka bir sayfadan geri geliş



viewWillAppear çalıştı
viewDidAppear çalıştı

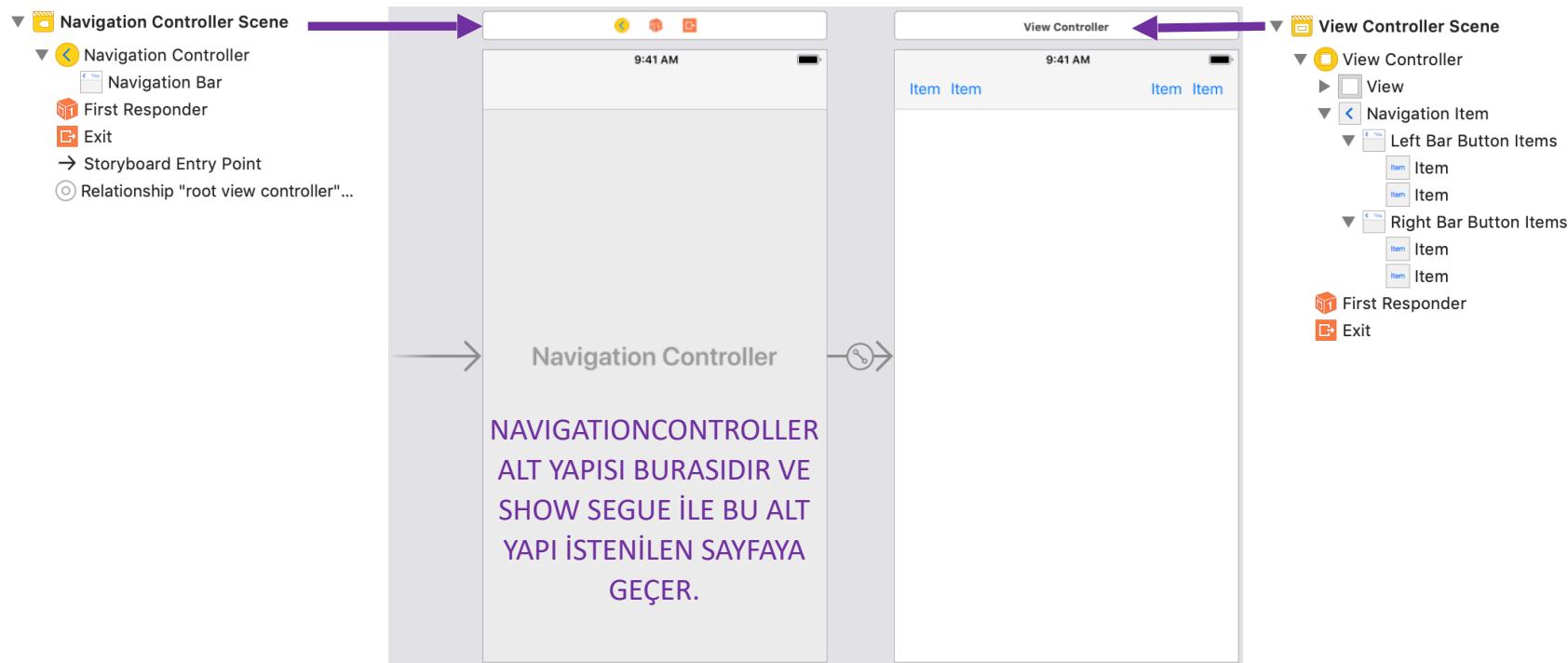
NAVIGATION CONTROLLER



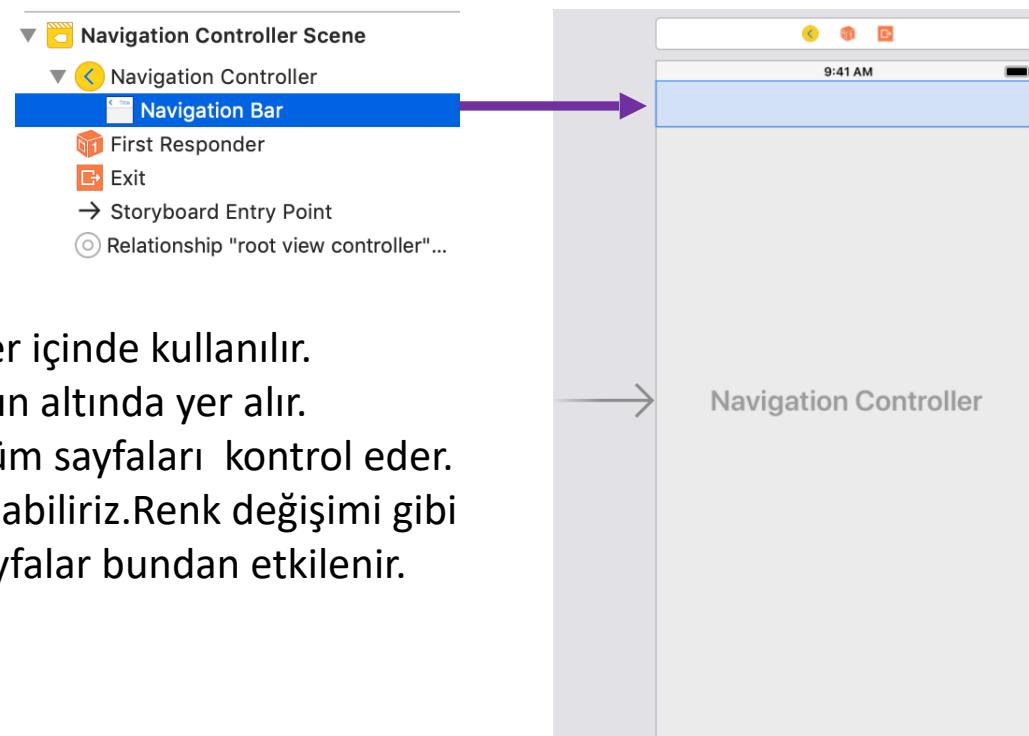
Navigation Controller

- Navigation Controller sayfalar arasında geçişlerde bize yardımcı olan bir yapıdır.
- Sayfa geçişlerinde otomatik olarak bir önceki sayfaya geri dönme işlemi yapılabilir.
- Aynı zamanda sayfanın en üst kısmında bulunur ve tasarımsal olarak her sayfaya başlık verebiliriz.
- Navigation controller üzerinde navigation bar,navigation item,left bar button items,right bar button items,bar button item,title,prompt,back button gibi yapılar bulunur.

Navigation Controller Yapısı

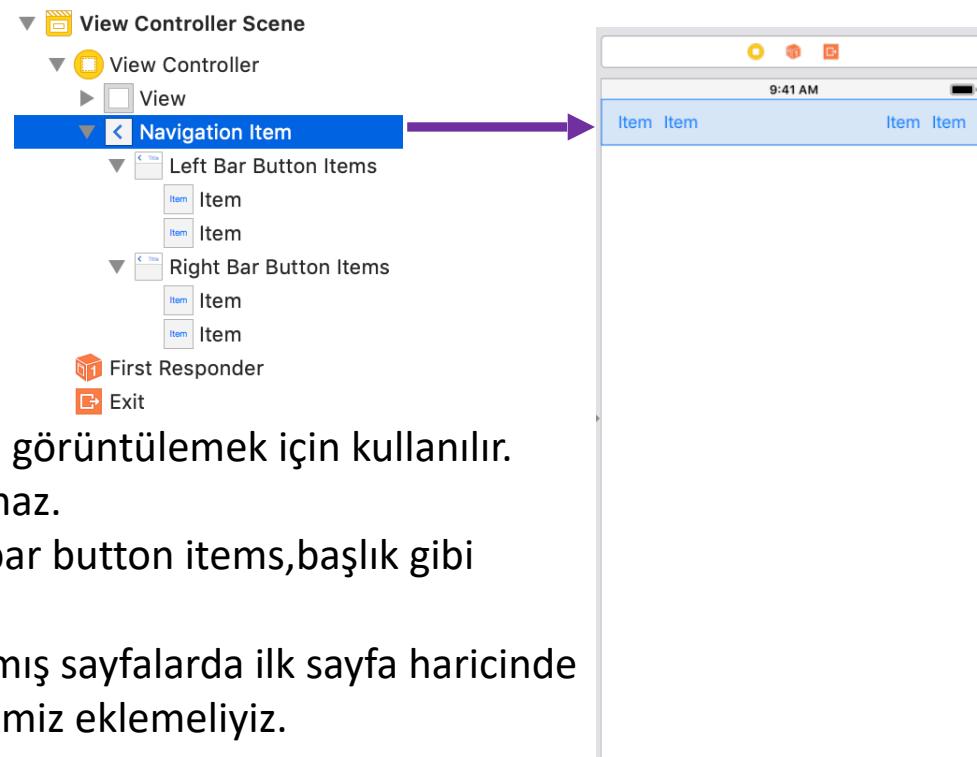


Navigation Bar



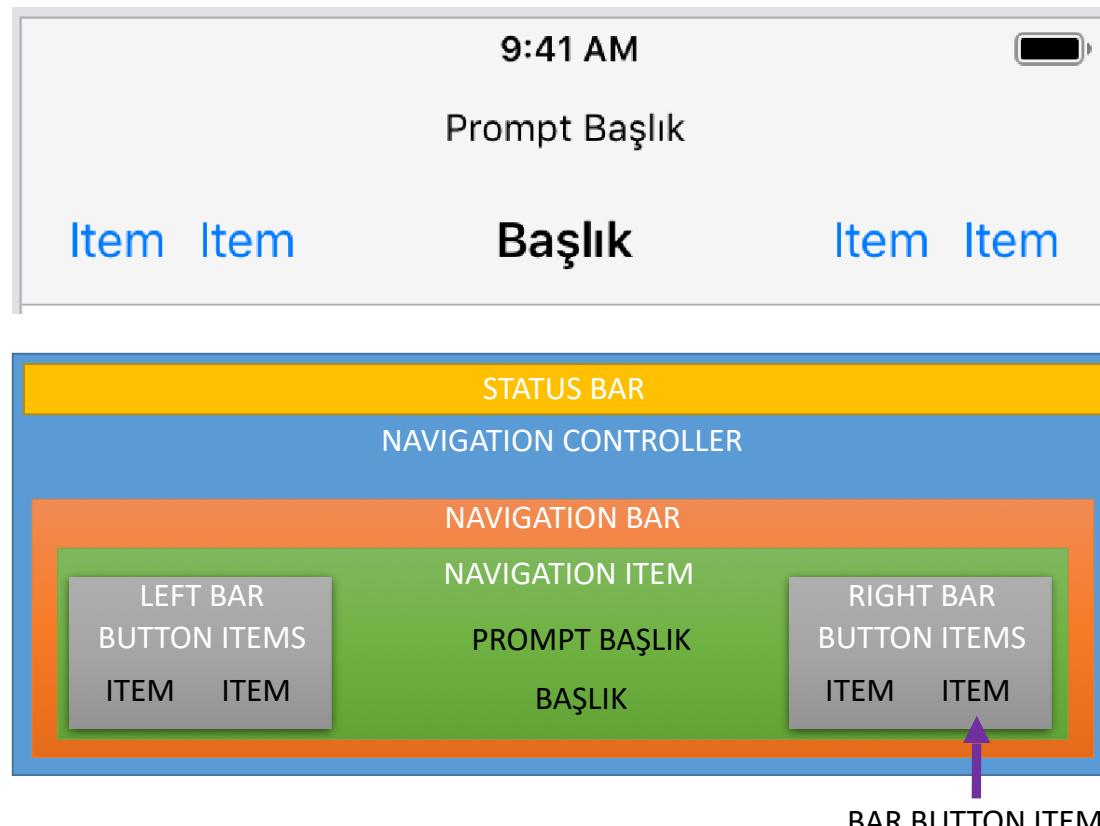
- Genellikle Navigation Controller içinde kullanılır.
- Ekranın en üstünde status bar'ın altında yer alır.
- Navigation Controller'a bağlı tüm sayfaları kontrol eder.
- Genel değişiklikleri burdan yapabiliriz. Renk değişimi gibi bu gerçekleştiği zaman tüm sayfalar bundan etkilenir.

Navigation Item

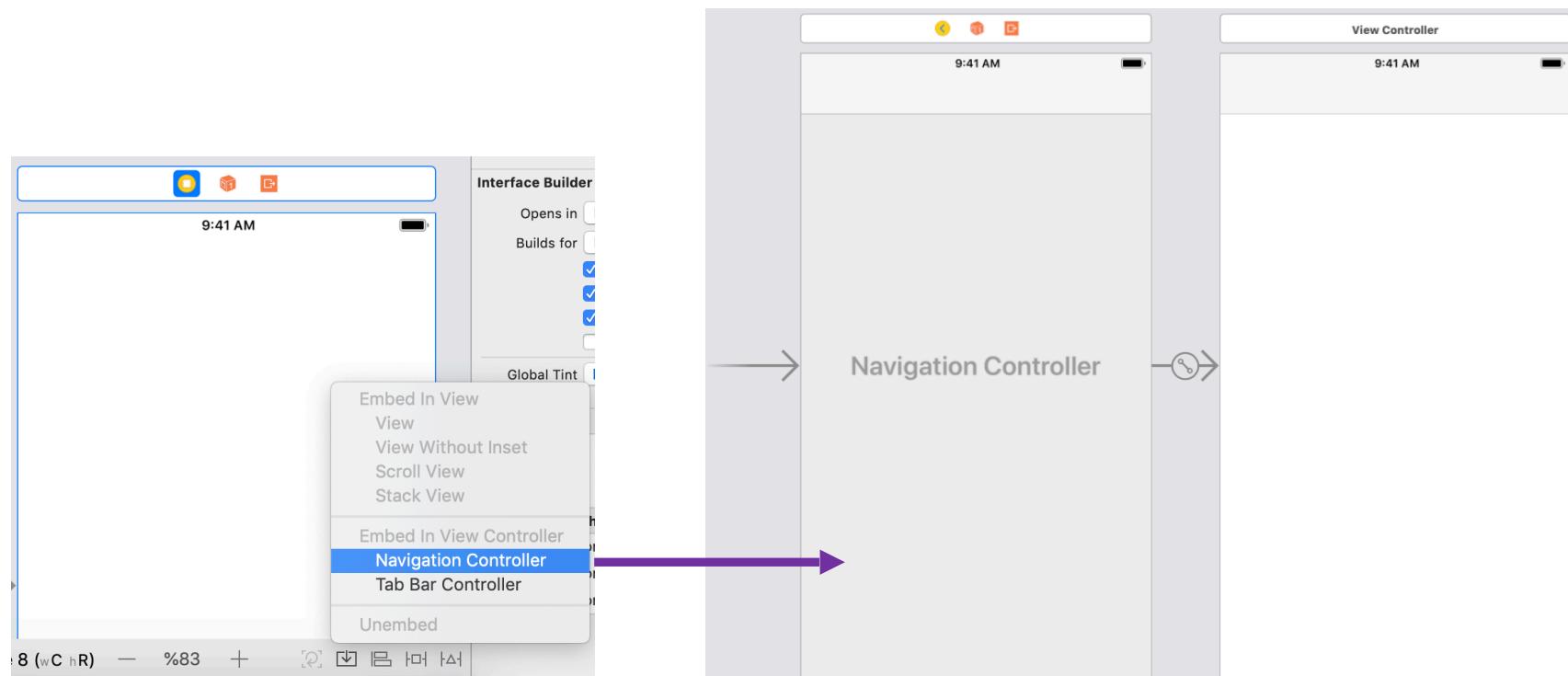


- Navigation Bar içerisinde item'ları görüntülemek için kullanılır.
- Aksi takdirde item'lar görünür olmaz.
- İçerisinde işlem yapabileceğimiz bar button items, başlık gibi yapılar vardır.
- Navigation Controller özelliğini almış sayfalarda ilk sayfa haricinde varsayılan olarak gelmez biz kendimiz eklemeliyiz.

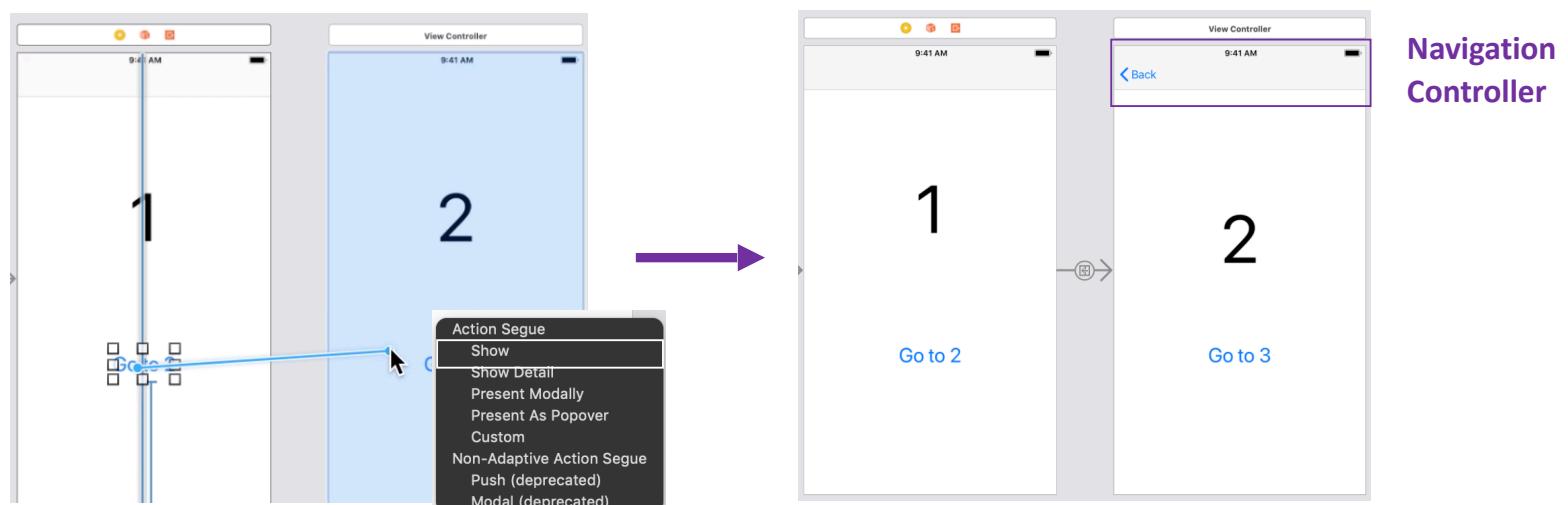
Genel İzlenim



Sayfaya Navigation Controller Ekleme

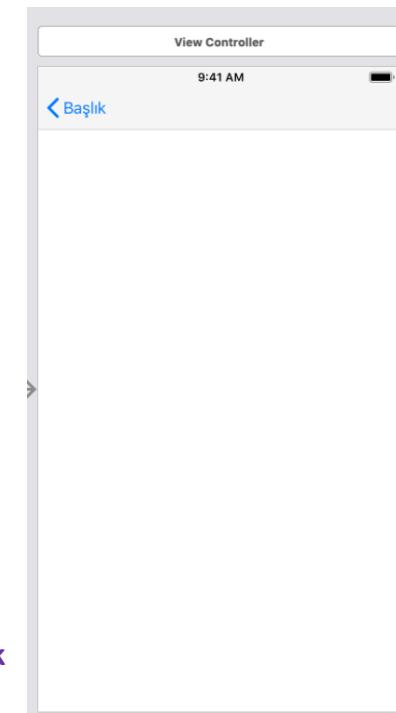
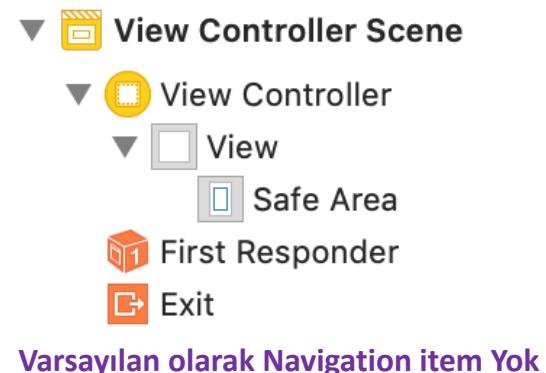


Yeni Bir Sayfaya Navigation Özelliğini Aktarma

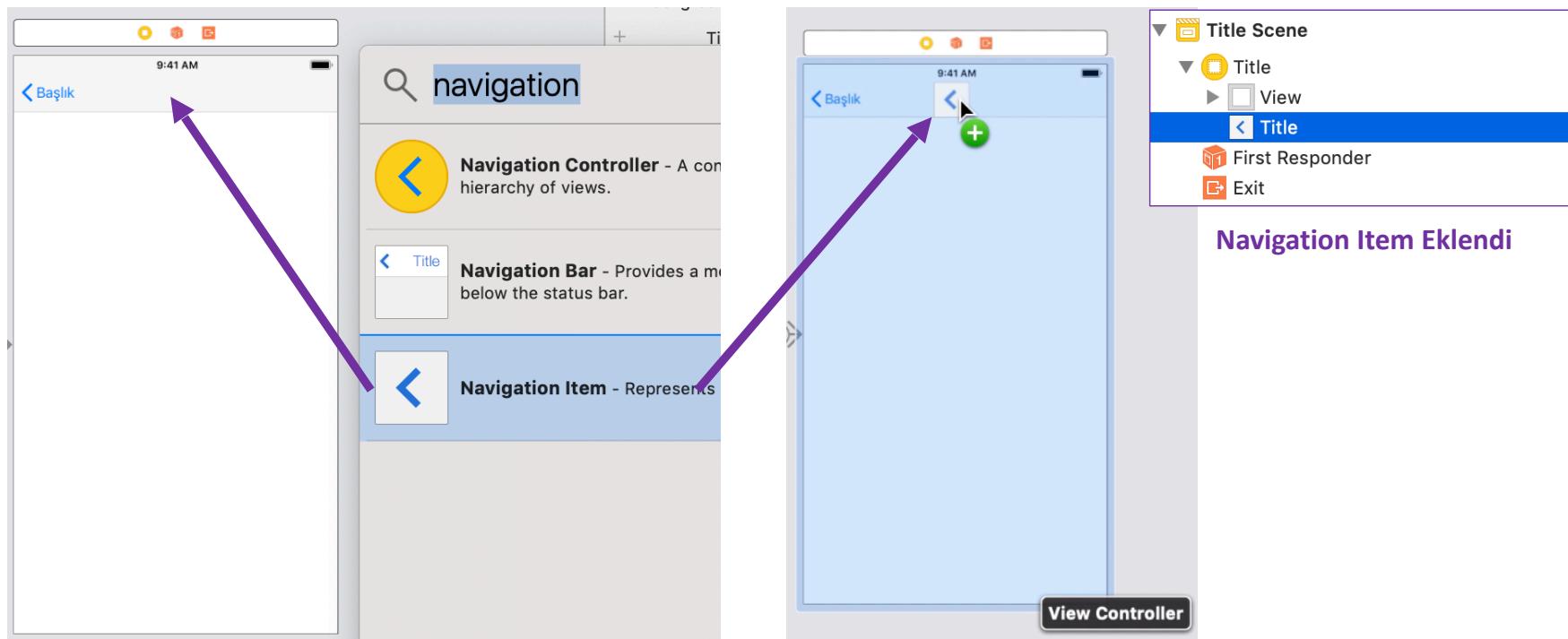


Yeni Bir Sayfaya Navigation Item Ekleme

- Navigation Özelliği aktarılan sınıfta varsayılan olarak navigation item olmaz.
- Eğer navigation bar üzerinde item ve başlık türevleri kullanmak isteniyorsa navigation item olmalıdır.

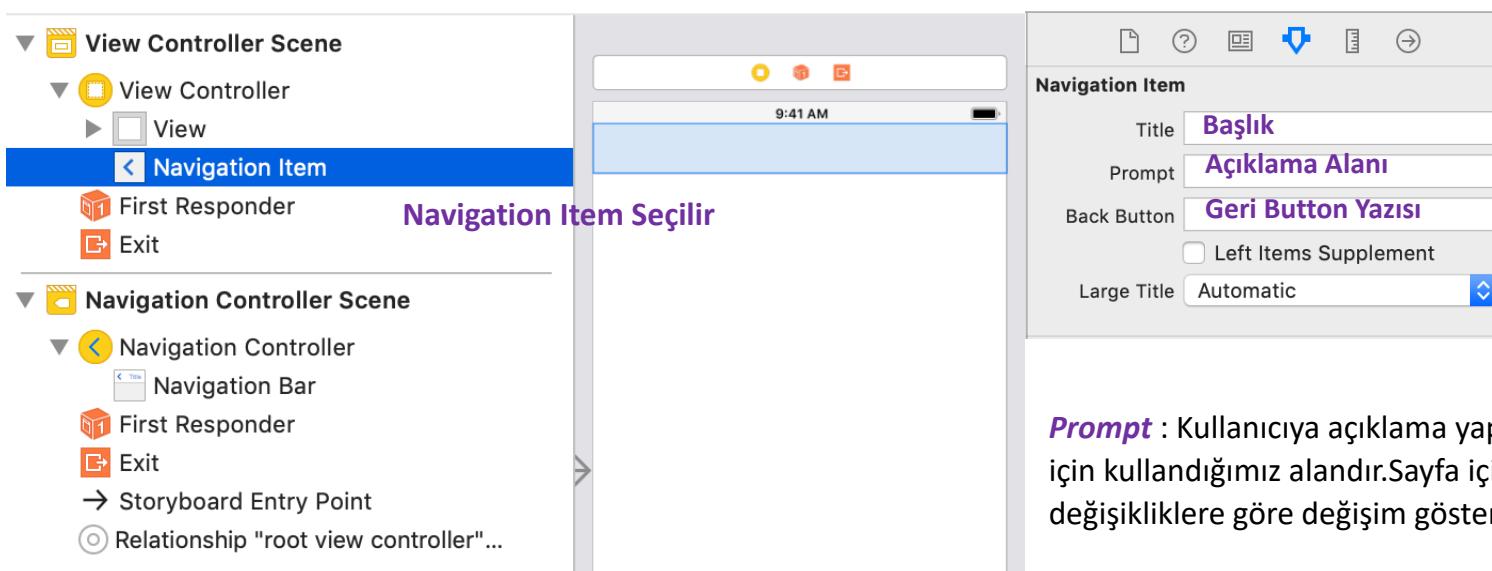


Yeni Bir Sayfaya Navigation Item Ekleme

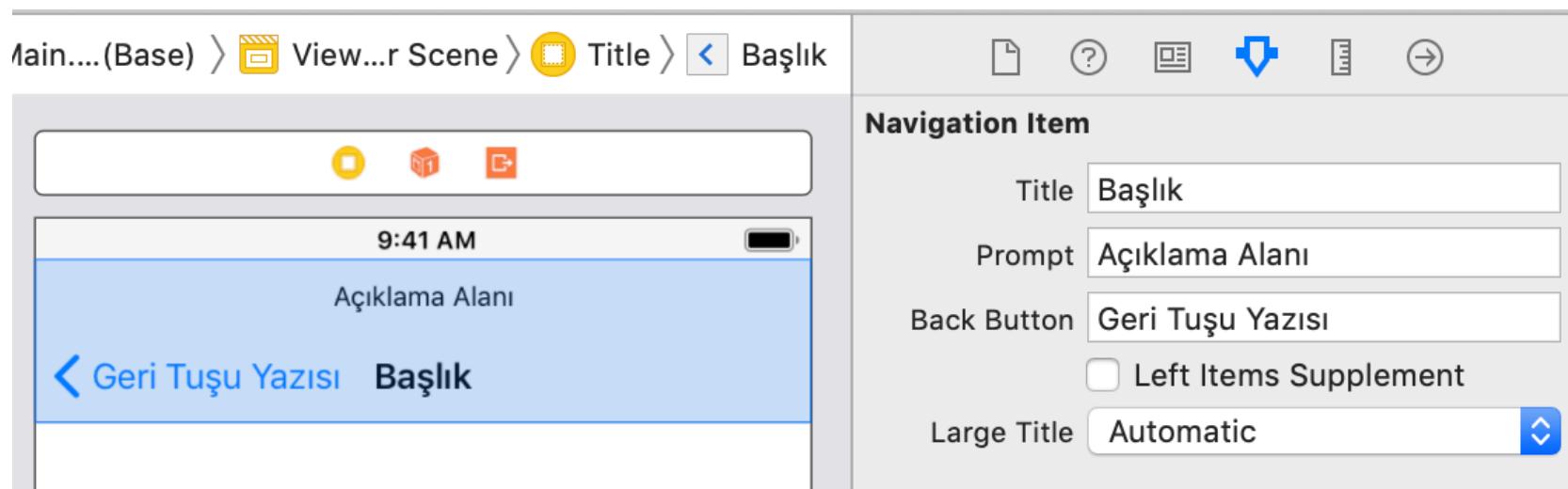


Navigation Item : Başlık Ekleme

- Başlık eklemek için öncelikli olarak ilgili sayfanın navigation item'ı seçilmelidir.



Navigation Item : Başlık ekleme



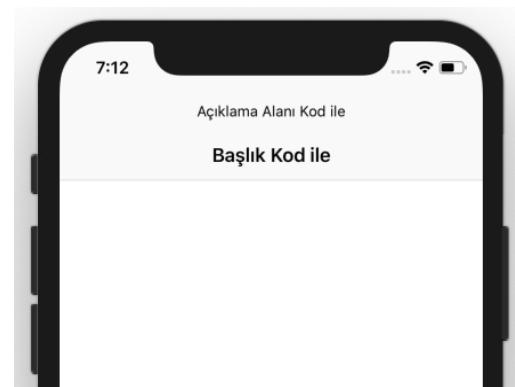
Navigation Item : Başlık Ekleme (Kod ile)

```
import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

        self.navigationItem.title = "Başlık Kod ile"
        self.navigationItem.prompt = "Açıklama Alanı Kod ile"
    }
}
```



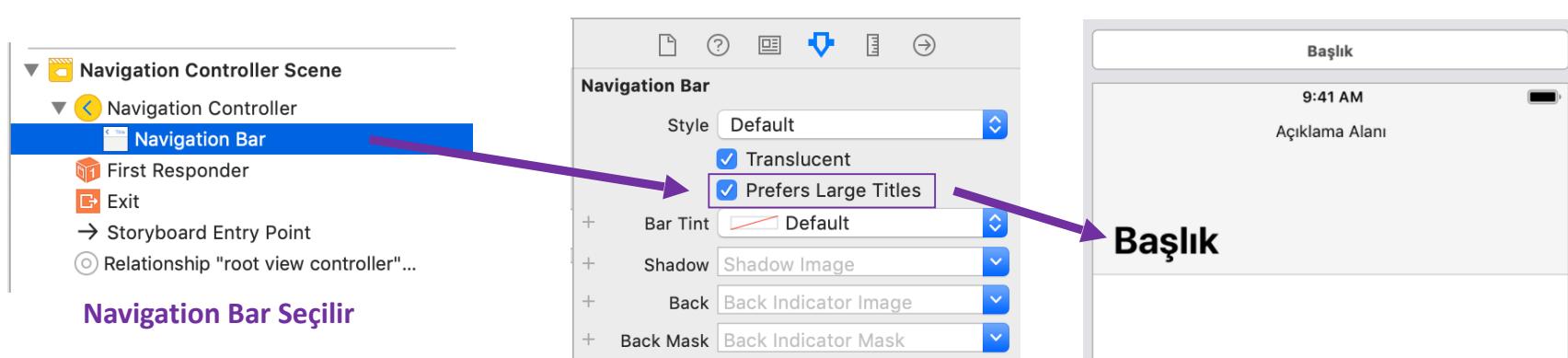
Navigation Item : Büyük Başlık Ekleme

- Yapının anlaşılması için IOS işletim sistemindeki settings uygulamasını inceleyebiliriz.
- Sayfa üzerindeki scroll özelliği yukarı doğru çalıştığında büyük başlık küçülür.
- İstenirse scroll özelliği olmasada başlık sürekli büyük durabilir.
- Görsel olarak güzel bir etki katar.



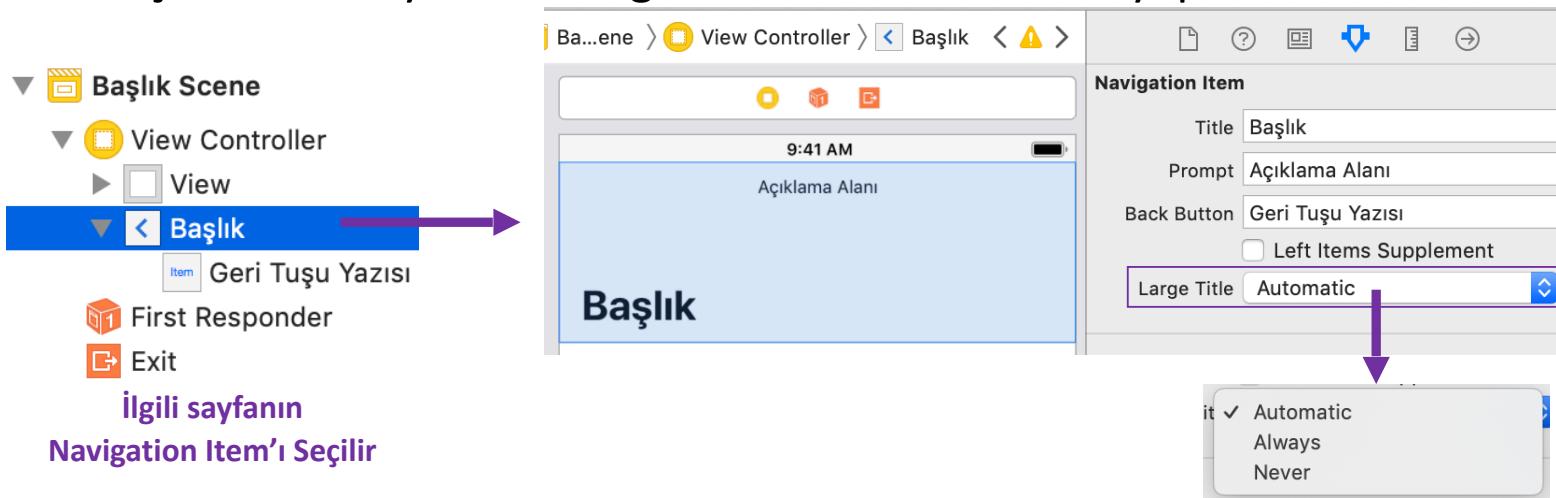
Navigation Item : Büyük Başlık Ekleme

- Öncelikle Navigation Bar üzerinden bu özelliği aktif etmeliyiz.
- Navigation Bar seçilir ve özellik aktif edilir.
- Aktif edilince bütün sayfalarda bu özellik çalışır hale gelir.



Navigation Item : Büyük Başlık Ekleme

- İsterseniz bazı sayfalarda bu özelliği kapatabiliriz veya çalışma şeklini değiştirebiliriz.
- Bu işlemi her sayfanın navigation item'ı üzerinden yapabiliriz.



Automatic : Duruma bağlı olarak büyür ve küçülür. Scroll hareketine duyarlı olur.

Always : Her zaman büyük olur.

Never : Her zaman küçük olur.

Navigation Item : Büyük Başlık Ekleme (Kod ile)

Kodlama ile büyük başlık önceliği değiştirilebilir.

```
import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

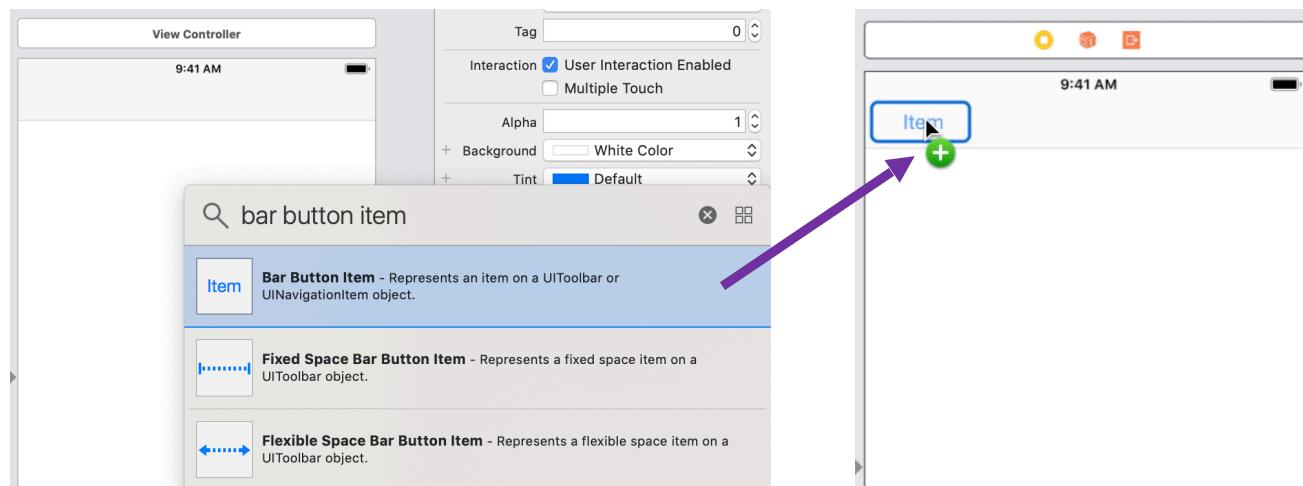
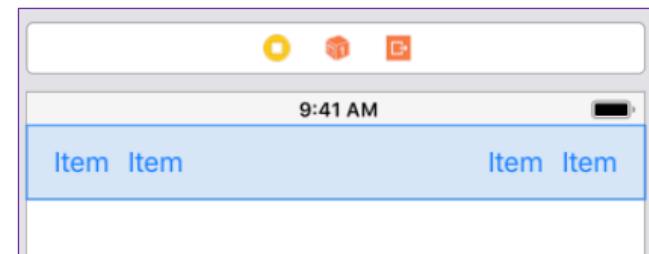
        //self.navigationItem.title = "Başlık Kod ile"

        //self.navigationItem.prompt = "Açıklama Alanı Kod ile"

        self.navigationItem.largeTitleDisplayMode = .automatic
        self.navigationItem.largeTitleDisplayMode = .always
        self.navigationItem.largeTitleDisplayMode = .never
    }
}
```

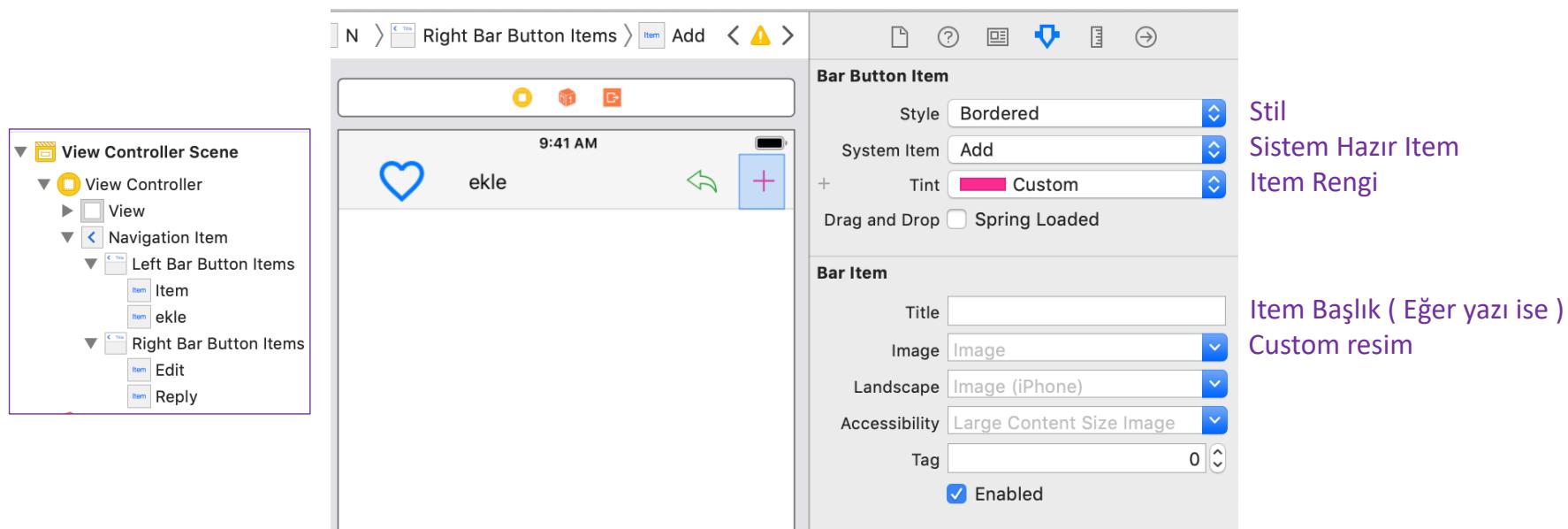
Navigation Item : Bar Button Item Ekleme

- Bar Button Item seçilir ve Navigation Item içine eklenir.
- En fazla 4 adet bar button item eklenebilir.



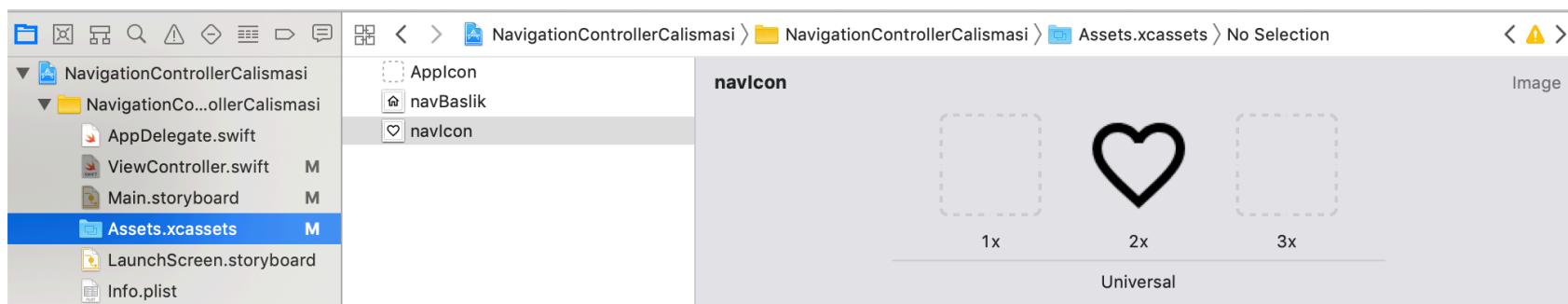
Navigation Item : Bar Button Item Ekleme

- İstenirse Bar Button Item üzerinde değişiklik yapılabilir.

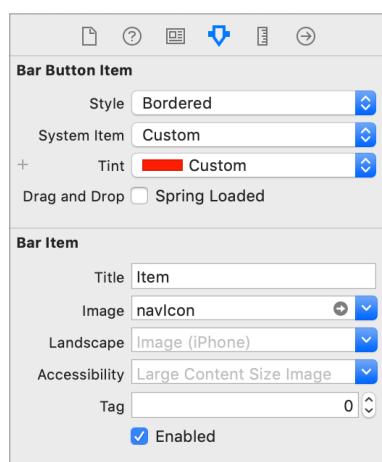


Navigation Item : Bar Button Item Ekleme

- Custom resim eklenecekse assets dosyasında image set oluşturulmalıdır.

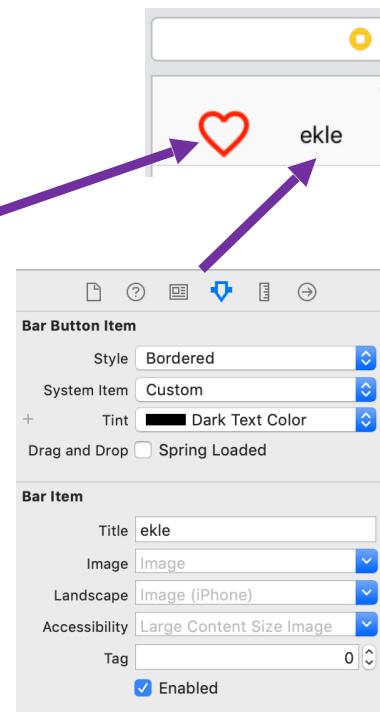


Navigation Item : Bar Button Item Ekleme

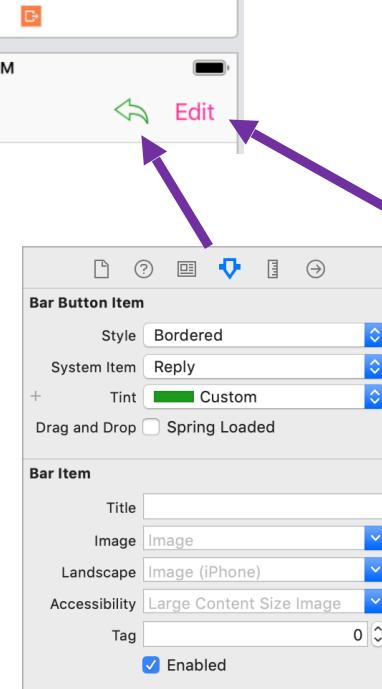


Custom Resim Ekleme

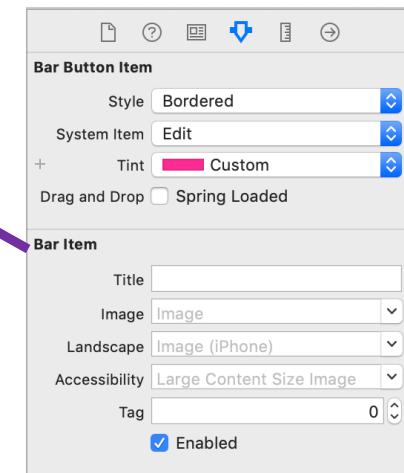
Resim ekleme için assets dosyasında image set oluşturulur.



Custom Yazı Ekleme



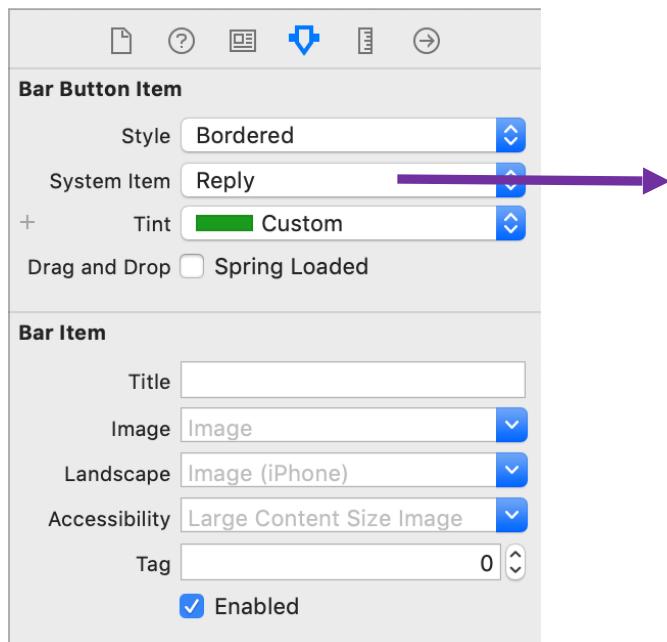
Sistem Icon Ekleme



Sistem Yazı Ekleme

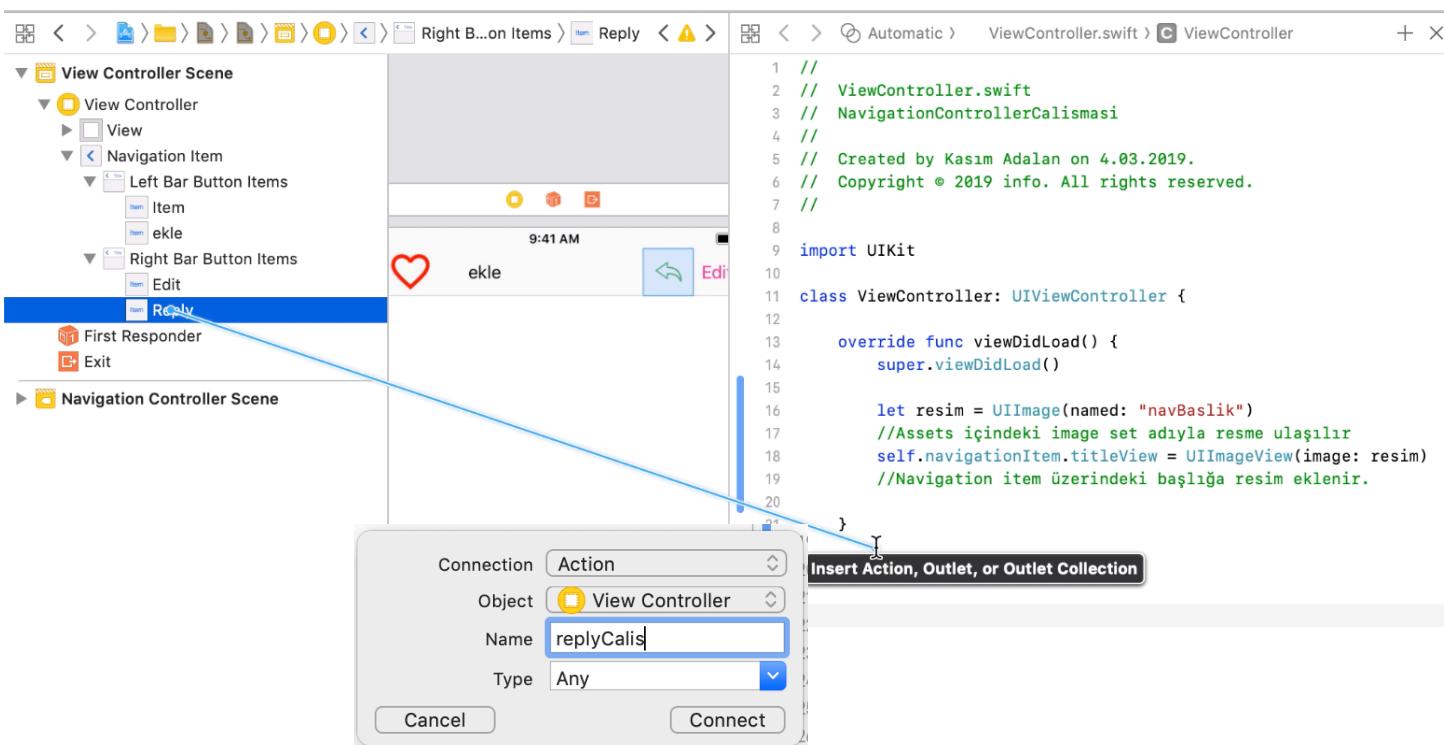
Bar Button Item : System Iconları

Sistem içinde hazır birçok icon vardır bunlar kullanılabilir.

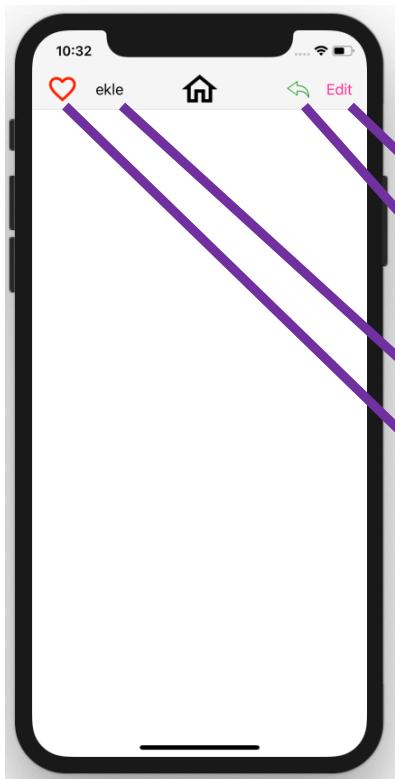


Icon	Name	Meaning	API
↑	Action (Share)	Shows a modal view containing share extensions, action extensions, and tasks, such as Copy, Favorite, or Find, that are useful in the current context.	action
+	Add	Creates a new item.	add
Bookmark	Bookmarks	Shows app-specific bookmarks.	bookmarks
Camera	Camera	Takes a photo or video, or shows the Photo Library.	camera
Cancel	Cancel	Closes the current view or ends edit mode without saving changes.	cancel
Compose	Compose	Opens a new view in edit mode.	compose
Done	Done	Saves the state and closes the current view, or exits edit mode.	done
Edit	Edit	Enters edit mode in the current context.	edit

Bar Button Item : Action olarak View Controller'a Bağlama



Bar Button Item : Action olarak View Controller'a Bağlama



```
import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

        let resim = UIImage(named: "navBaslik")
        //Assets içindeki image set adıyla resme ulaşılır
        self.navigationItem.titleView = UIImageView(image: resim)
        //Navigation item üzerindeki başlığa resim eklenir.

    }

    @IBAction func editCalis(_ sender: Any) {
        print("Edit Çalış")
    }

    @IBAction func replyCalis(_ sender: Any) {
        print("Reply Çalış")
    }

    @IBAction func ekleCalis(_ sender: Any) {
        print("Ekle Çalış")
    }

    @IBAction func begenCalis(_ sender: Any) {
        print("Beğen Çalış")
    }
}
```

Navigation Bar Özelleştirme

```
class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

        self.navigationItem.title = "Başlık"
        self.navigationItem.prompt = "Açıklama"

        //Özelleştirme : Bütün sayfaları etkiler
        navigationController?.navigationBar.prefersLargeTitles = true //Büyük başlık özelliği

        let appearance = UINavigationBarAppearance()

        appearance.backgroundColor = UIColor.red//Arkaplan rengi

        appearance.titleTextAttributes = [.foregroundColor: UIColor.white]//Başlık rengi

        appearance.largeTitleTextAttributes = [.foregroundColor: UIColor.white]//Büyük başlık rengi

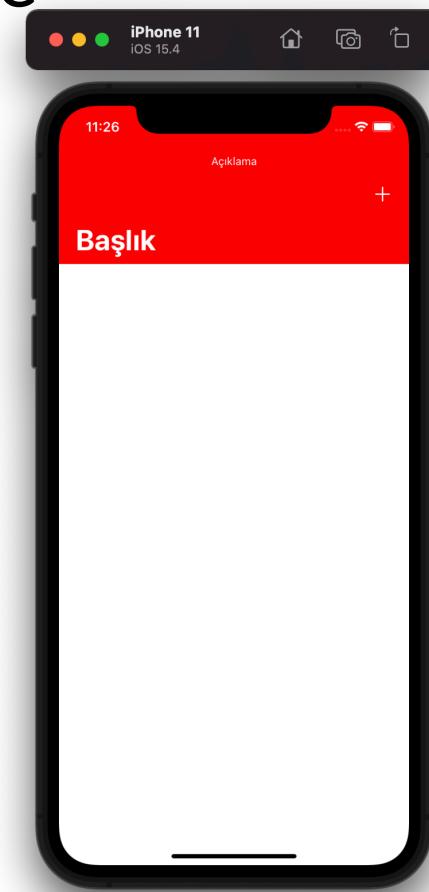
        navigationController?.navigationBar.tintColor = .white //Bar button item rengi

        navigationController?.navigationBar.barStyle = .black //Status bar ve prompt rengi değişimi

        navigationController?.navigationBar.isTranslucent = true //RGB renk paletine göre arkaplan görünür.

        navigationController?.navigationBar.standardAppearance = appearance
        navigationController?.navigationBar.compactAppearance = appearance
        navigationController?.navigationBar.scrollEdgeAppearance = appearance
    }

    @IBAction func ekleTikla(_ sender: Any) {
        print("Ekle tıklandı")
    }
}
```



TabBar Controller

Tab Bar Controller

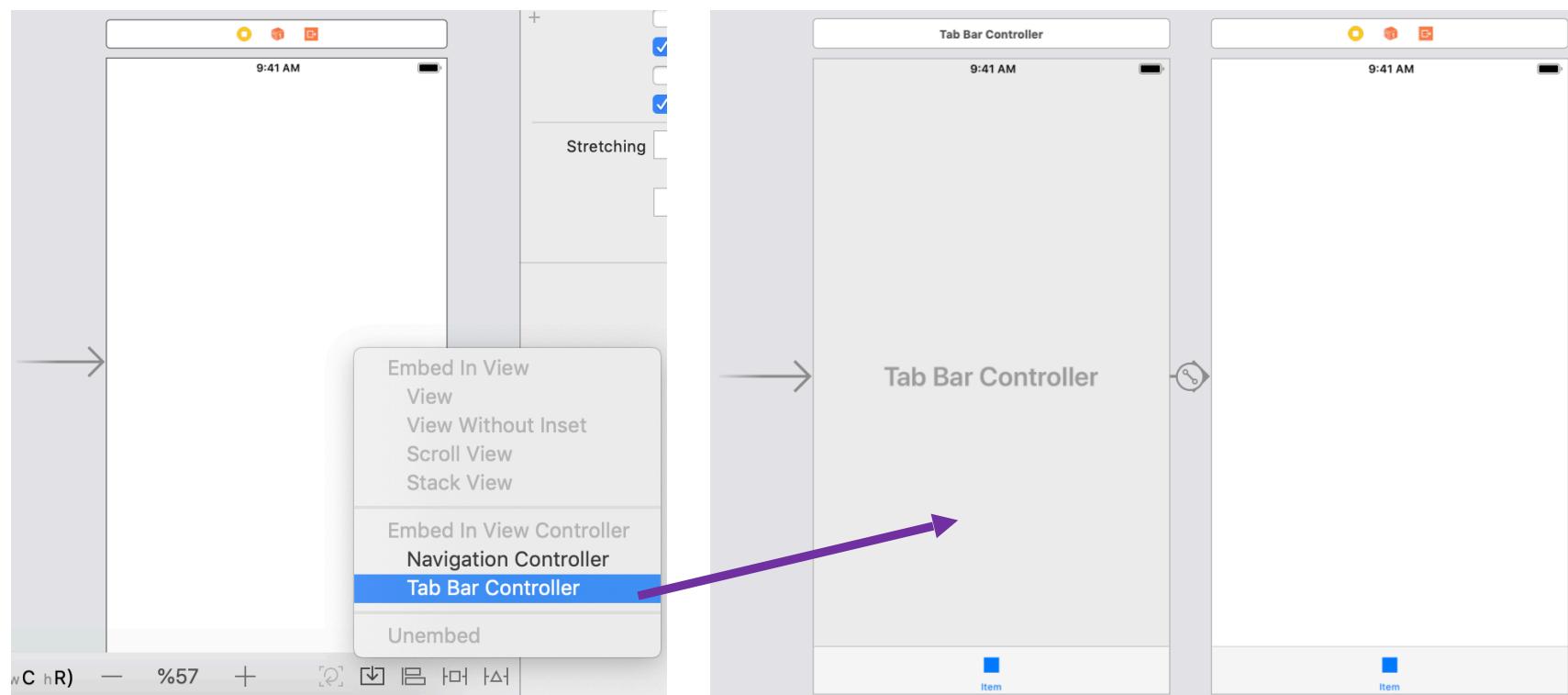
Alt segmeler ile sayfalar arasında geçiş sağlayan bir alt yapıdır.

Her bir segme bir View Controller'dan oluşur.

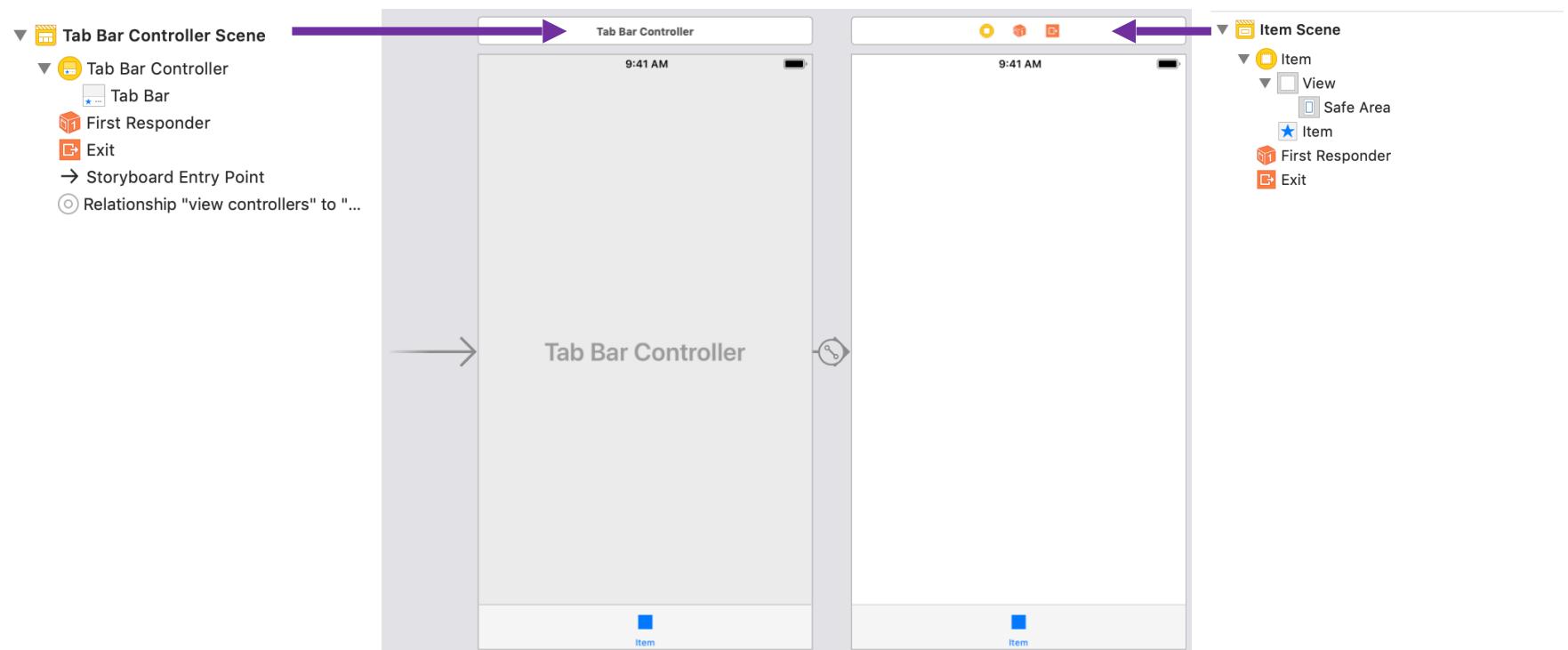
View Controller gibi üzerinde widgetlar bulundurmaz.



Tab Bar Controller Ekleme



Tab Bar Controller Yapısı

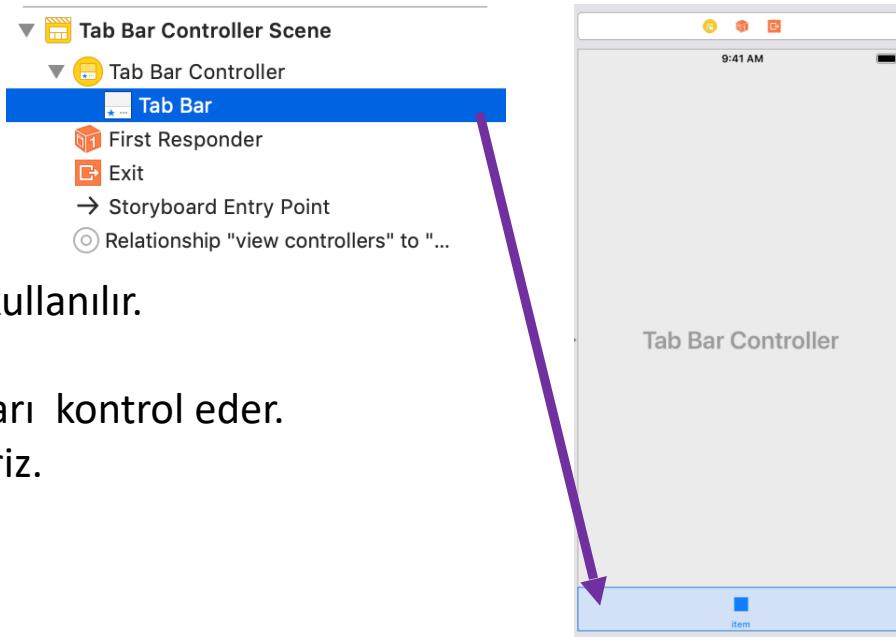


Tab Bar Controller Yapısı

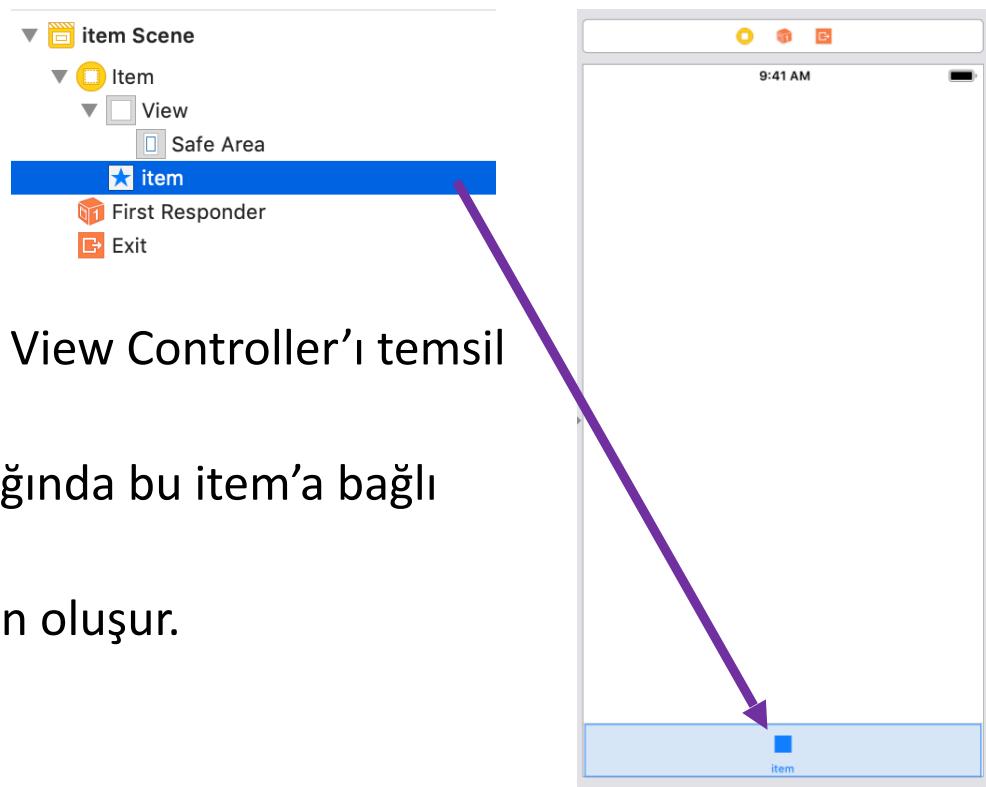


Tab Bar

- Genellikle Tab Bar Controller içinde kullanılır.
- Ekranın en altında yer alır.
- Tab Bar Controller'a bağlı tüm sayfaları kontrol eder.
- Genel değişiklikleri buradan yapabiliriz.
- Renk değişimi vb.



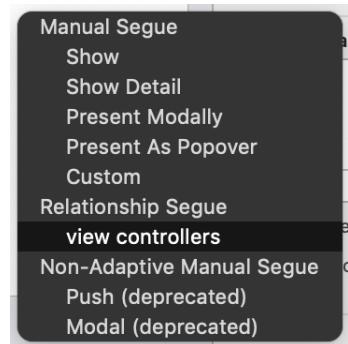
Tab Bar Item



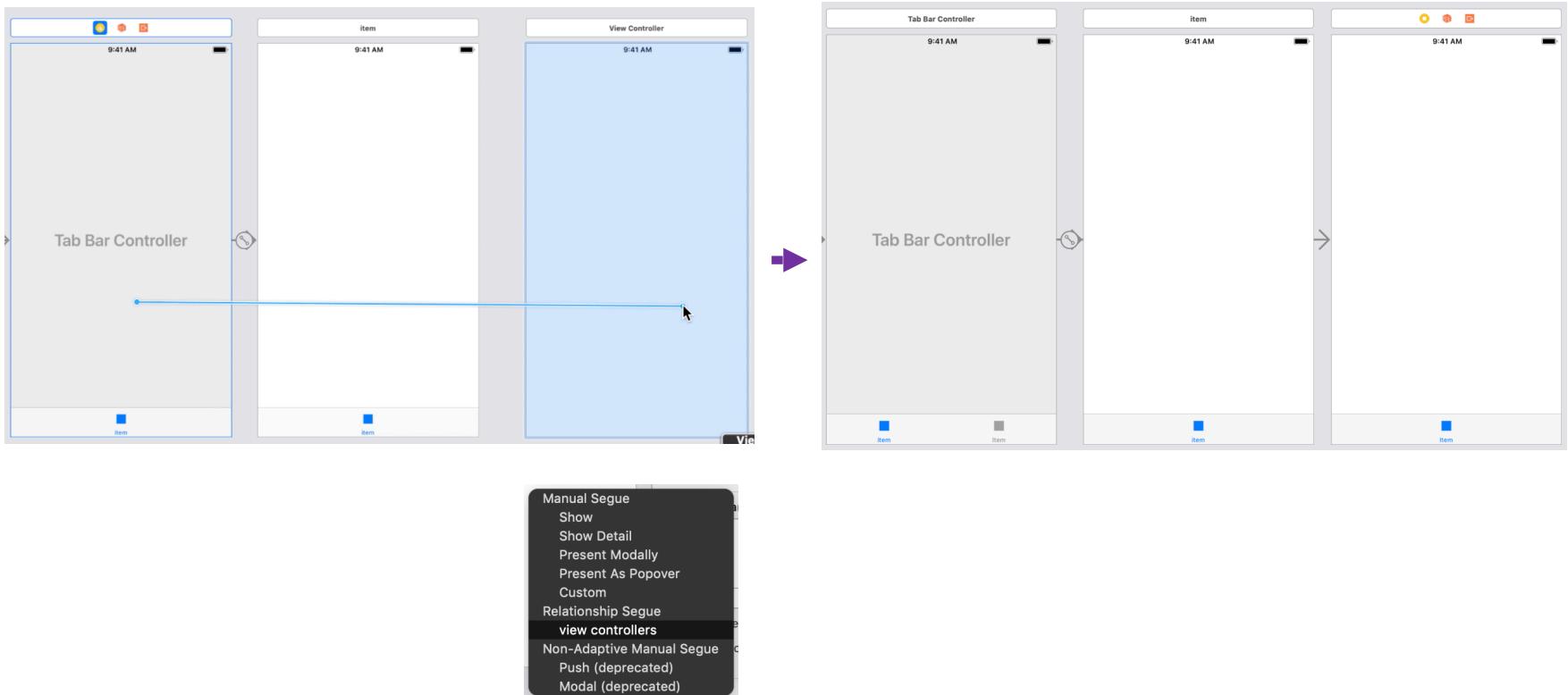
- Her bir Tab Bar Item , bir adet View Controller’ı temsil etmektedir.
- Tab Bar Item üzerine tıklanıldığında bu item'a bağlı View Controller görüntülenir.
- Tab bar item resim ve başlıktan oluşur.

Yeni Tab Ekleme

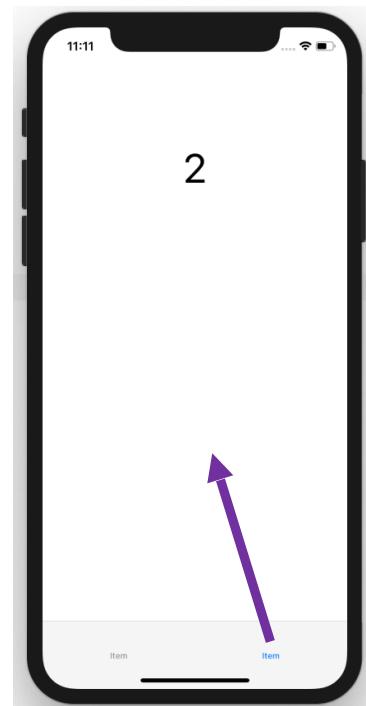
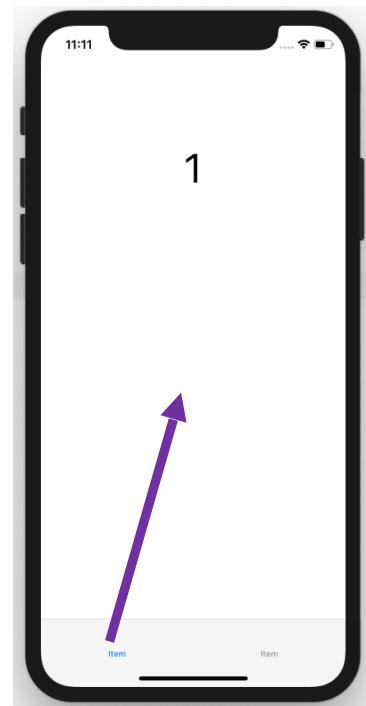
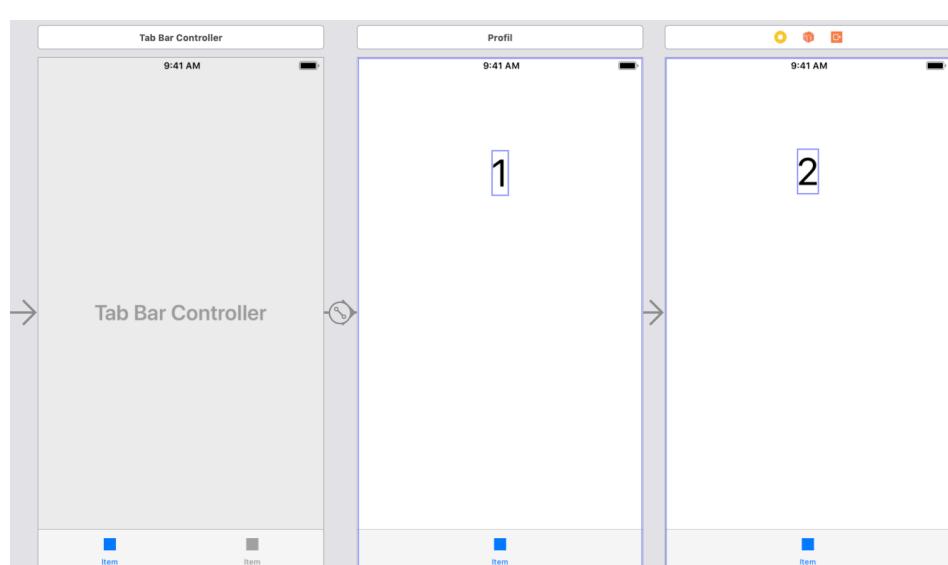
- Her tab için bir adet View Controller gerekmektedir.
- Öncelikle View Controller eklenir.
- Tab Bar Controller üzerinden, View Controller üzerine kntrl + sürükle yaparak segue oluşturmalıyız.
- Segue türü Relationship Segue : View Controller'dır.



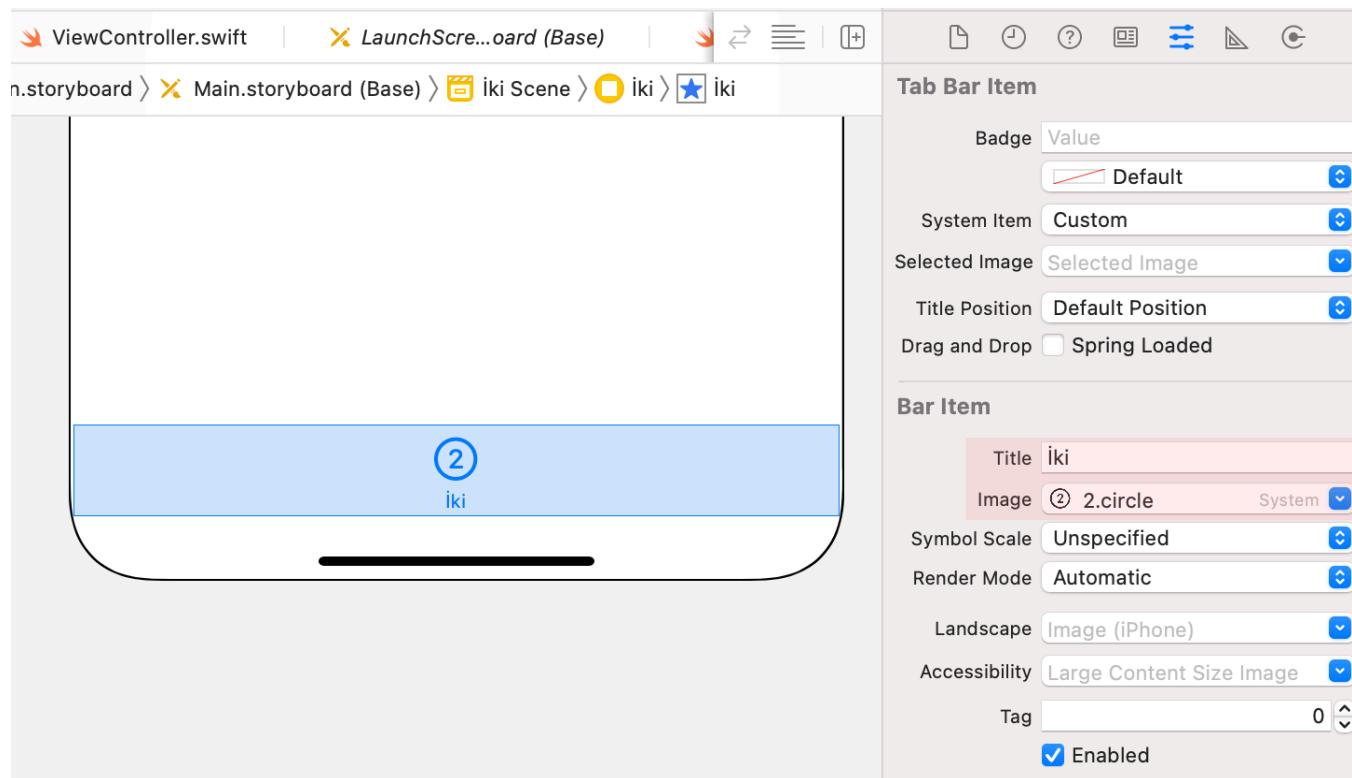
Yeni Tab Ekleme



Tab Bar Çalışmasını İncele

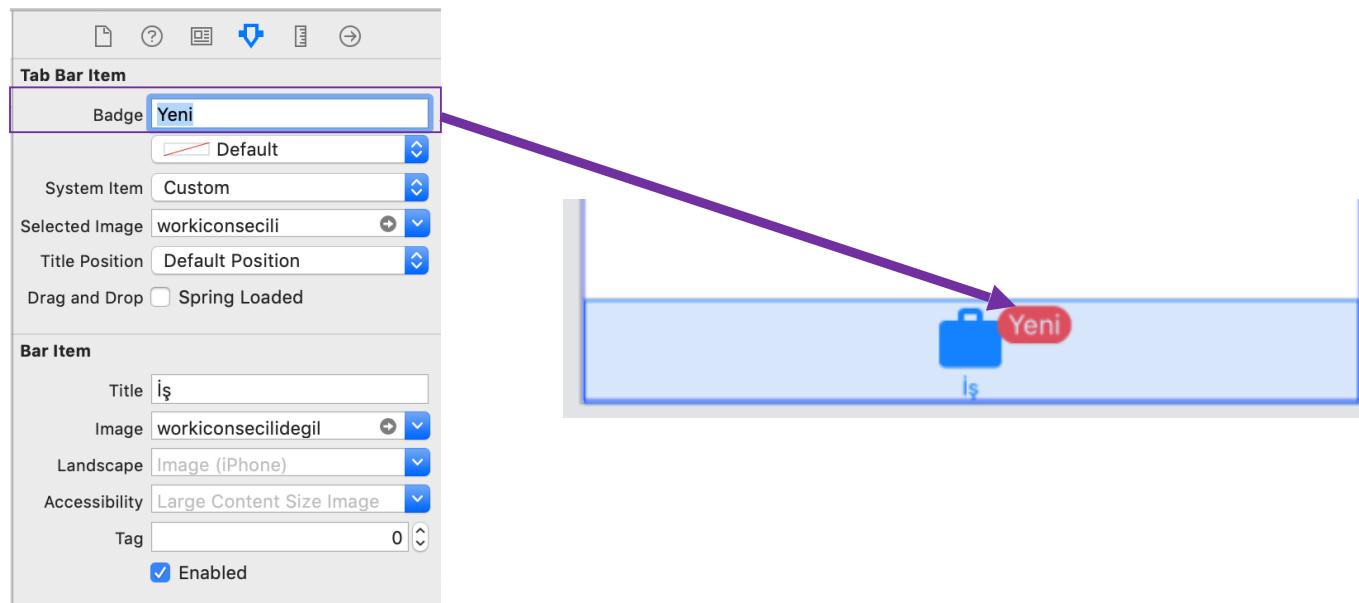


Tab Bar Item : Sistem Resmi ve Başlık

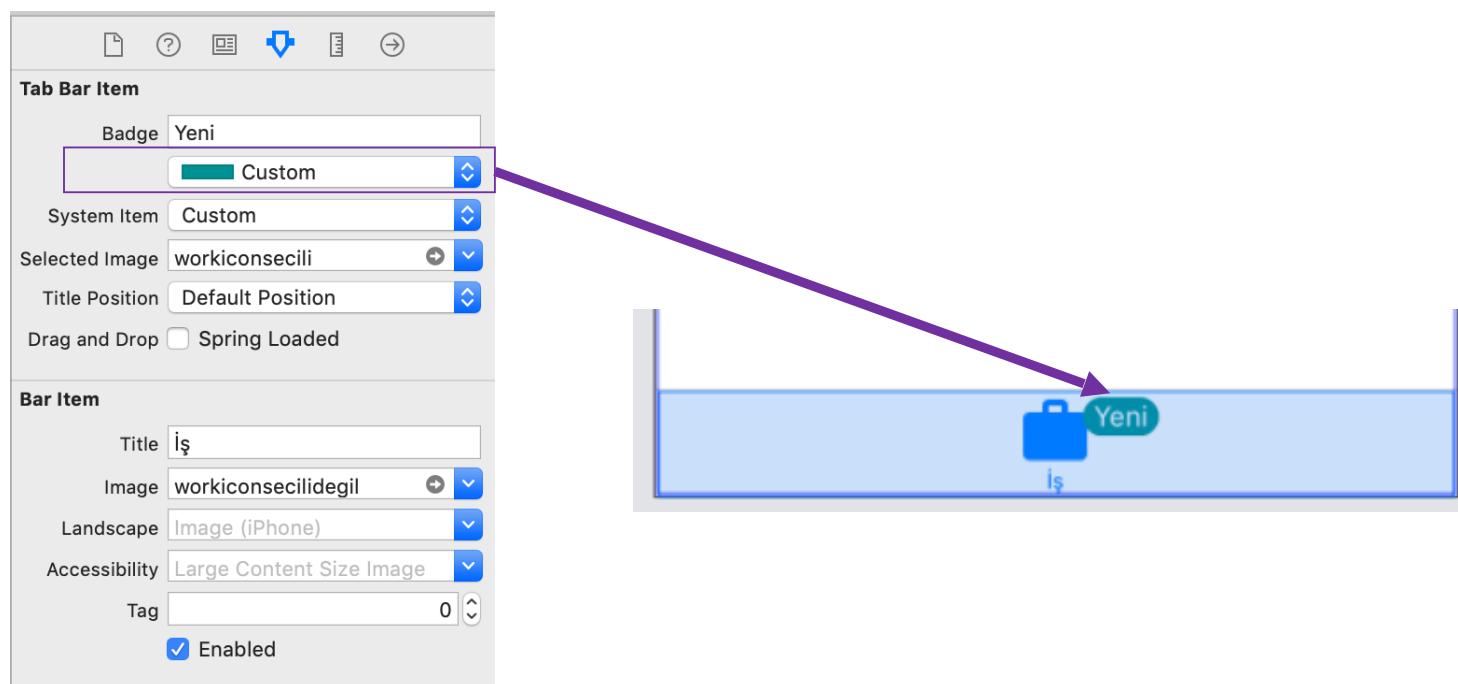


Tab Bar Item : Badge Ekleme

- Badge , tab bar itemları üzerinde belirteç olarak görev yapar.
 - Örn: Gelen mesaj sayısı , segme uyarısı vb.
- Storyboard üzerinden badge eklenebilir fakat ağırlıklı olarak kod ile kullanılır.

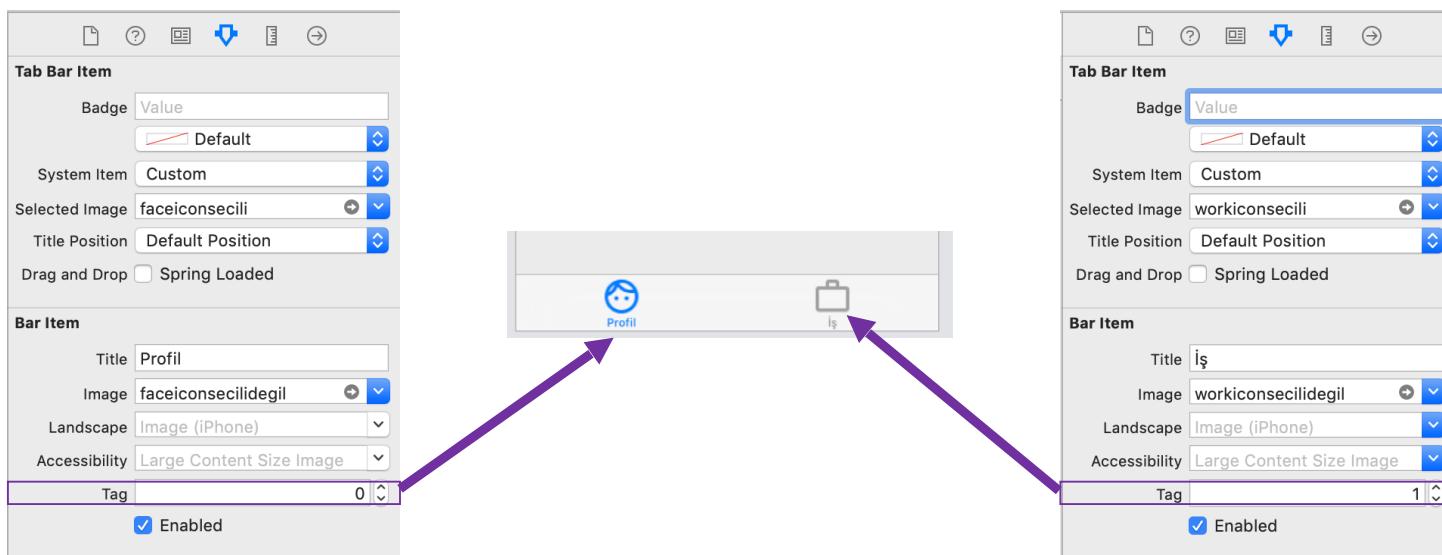


Tab Bar Item : Badge Renk Değişimi



Tab Bar Item : Badge Ekleme (Kod ile)

- Hangi tab bar item'a badge eklemek istiyorsak onun **tag** değerini bilmeliyiz.



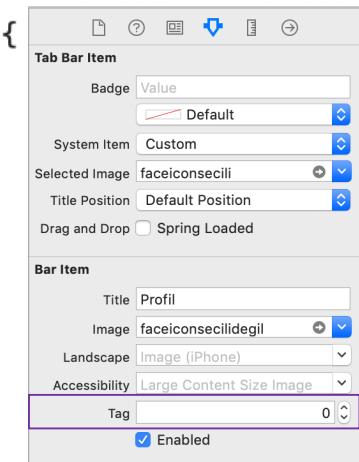
Tab Bar Item : Badge Ekleme (Kod ile)

```
import UIKit

class ViewController: UIViewController {

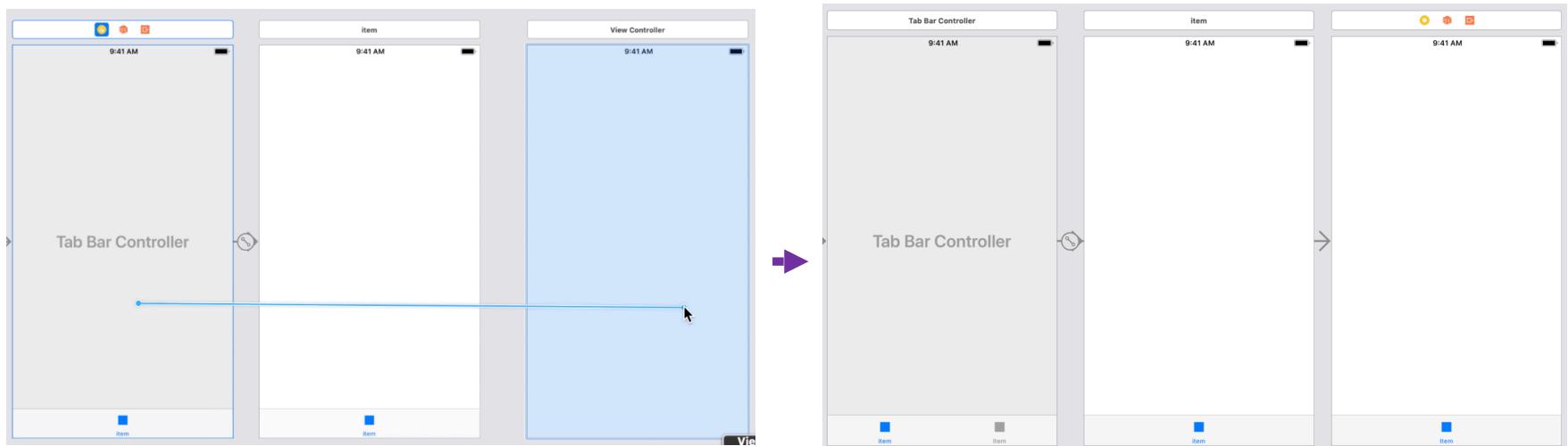
    override func viewDidLoad() {
        super.viewDidLoad()

        if let tabsItem = tabBarController?.tabBar.items {
            let item = tabsItem[0]//Tab indeks numarası
            item.badgeValue = "3"
            item.badgeColor = UIColor.red
        }
    }
}
```

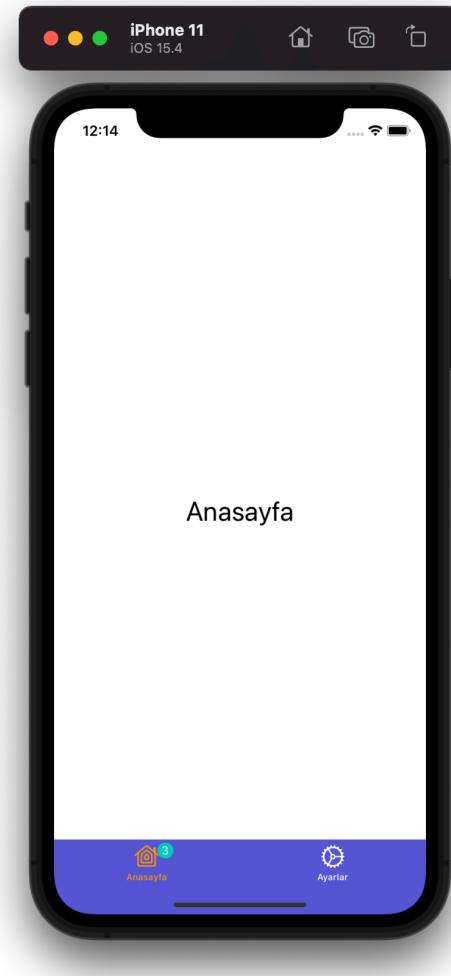
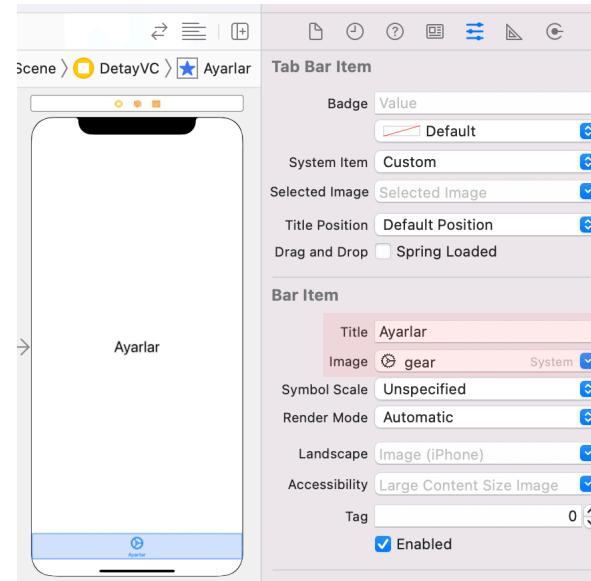
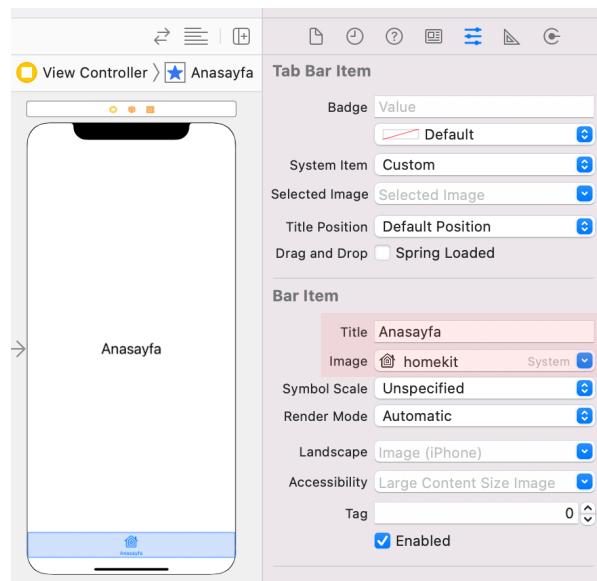


TabBar Controller Özelleştirme

Yeni Tab Ekleme



TabBar Controller Özelleştirme



TabBar Controller Özelleştirme

```
class ViewController: UIViewController {
    override func viewDidLoad() {
        super.viewDidLoad()

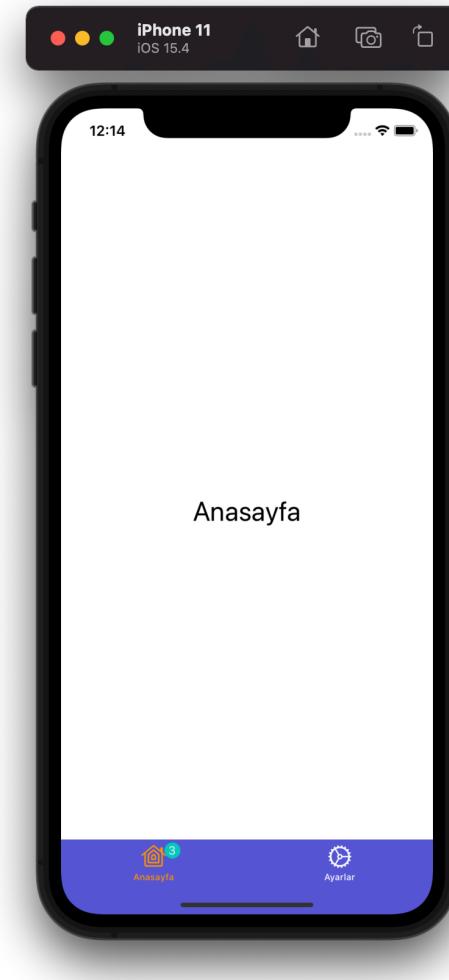
        if let tabItems = tabBarController?.tabBar.items {
            let item = tabItems[0]
            item.badgeValue = "3"
        }

        let appearance = UITabBarAppearance()
        appearance.backgroundColor = UIColor.systemIndigo

        renkDegistir(itemAppearance: appearance.stackedLayoutAppearance)
        renkDegistir(itemAppearance: appearance.inlineLayoutAppearance)
        renkDegistir(itemAppearance: appearance.compactInlineLayoutAppearance)

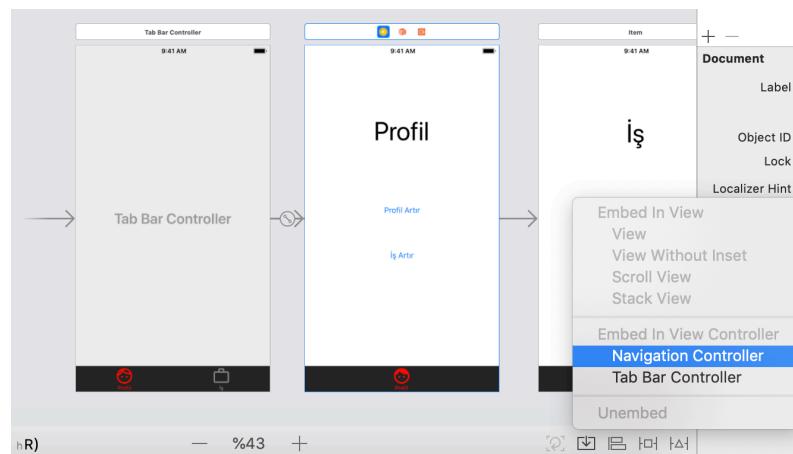
        tabBarController?.tabBar.standardAppearance = appearance
        tabBarController?.tabBar.scrollEdgeAppearance = appearance
    }

    func renkDegistir(itemAppearance:UITabBarItemAppearance){
        //Seçili durum
        itemAppearance.selected.iconColor = UIColor.systemOrange
        itemAppearance.selected.titleTextAttributes = [.foregroundColor: UIColor.systemOrange]
        itemAppearance.selected.badgeBackgroundColor = UIColor.systemMint
        //Seçili olmayan durum
        itemAppearance.normal.iconColor = UIColor.white
        itemAppearance.normal.titleTextAttributes = [.foregroundColor: UIColor.white]
        itemAppearance.normal.badgeBackgroundColor = UIColor.lightGray
    }
}
```

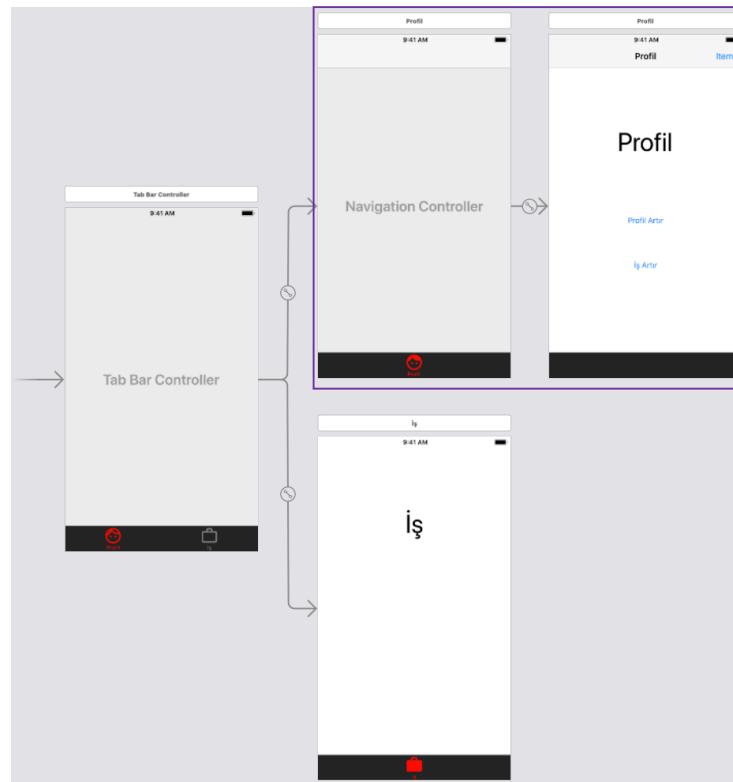


Tab Bar Üzerine Navigation Controller Ekleme

- Burada dikkat edilmesi gereken Navigation Controller her bir tab' tek eklenmelidir.
- Her tab bar item'a bağlı view controller kendisinden sorumludur.
- Tab bar'a bağlı navigation controller özelliğine sahip bir tab bar item başka bir view controllara show segue ile geçiş yapılırsa aynı tab bar item içinde açılır.Tab bar dışına çıkmaz.

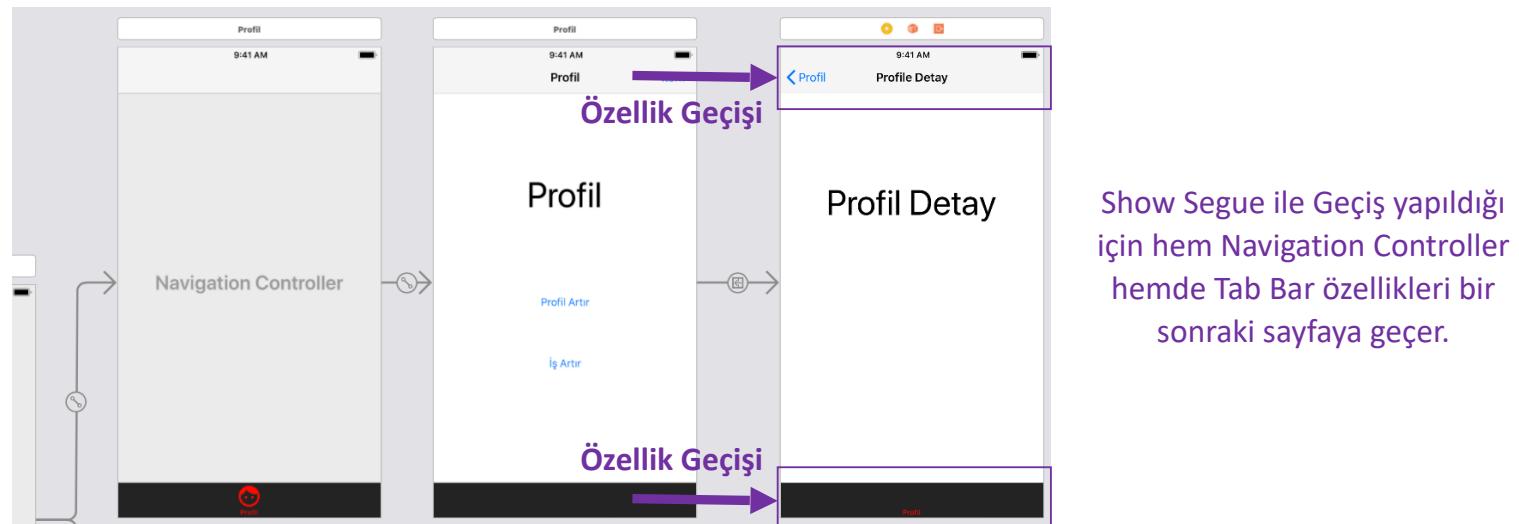


Tab Bar Üzerine Navigation Controller Ekleme

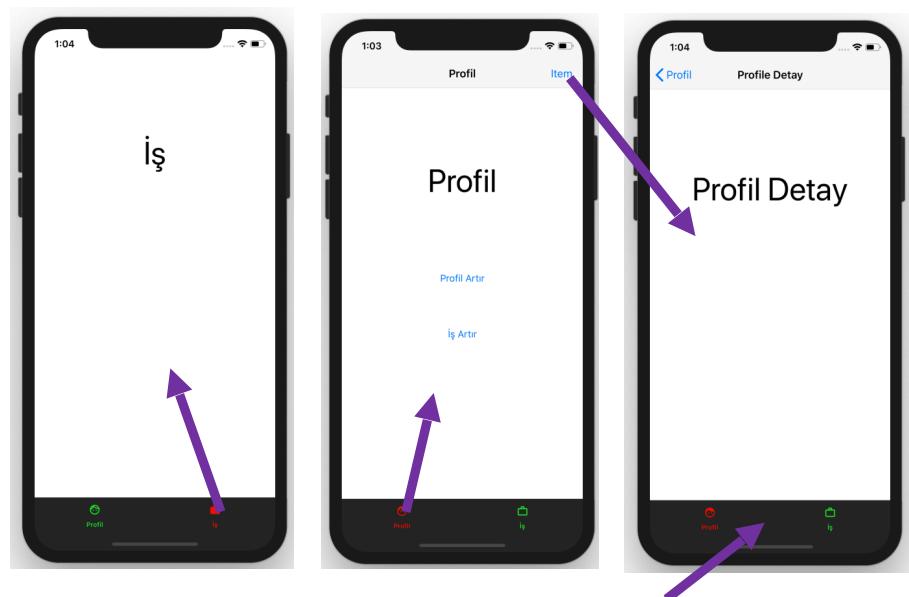
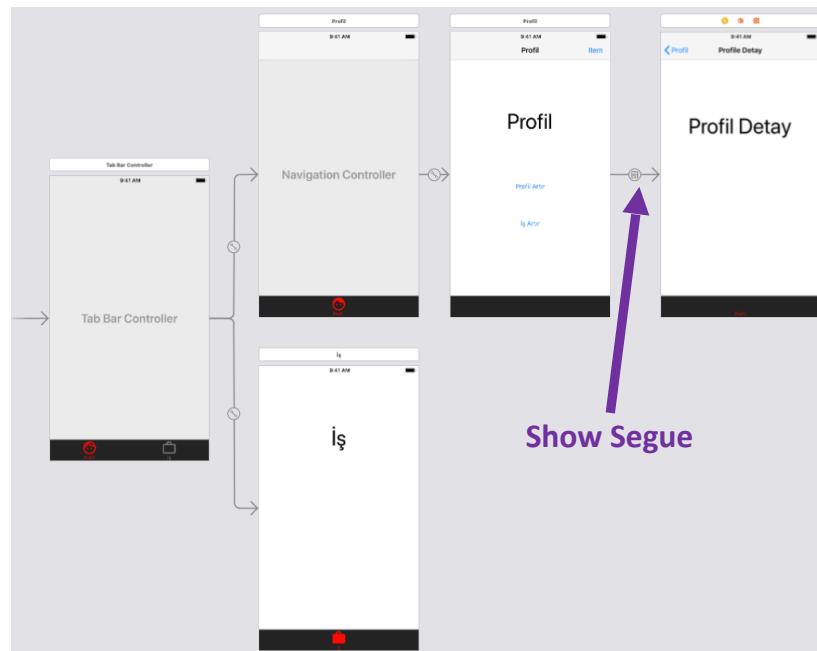


Yeni Bir View Controller Ekleme

- Navigation Controller özelliği olan tab bar item'daki View Controller'a show segue ile başka bir sayfaya geçiş yapılırsa.
- Tab bar item'dan çıkışız aynı tab bar item içinde sayfa açılır.



Yeni Bir View Controller Ekleme

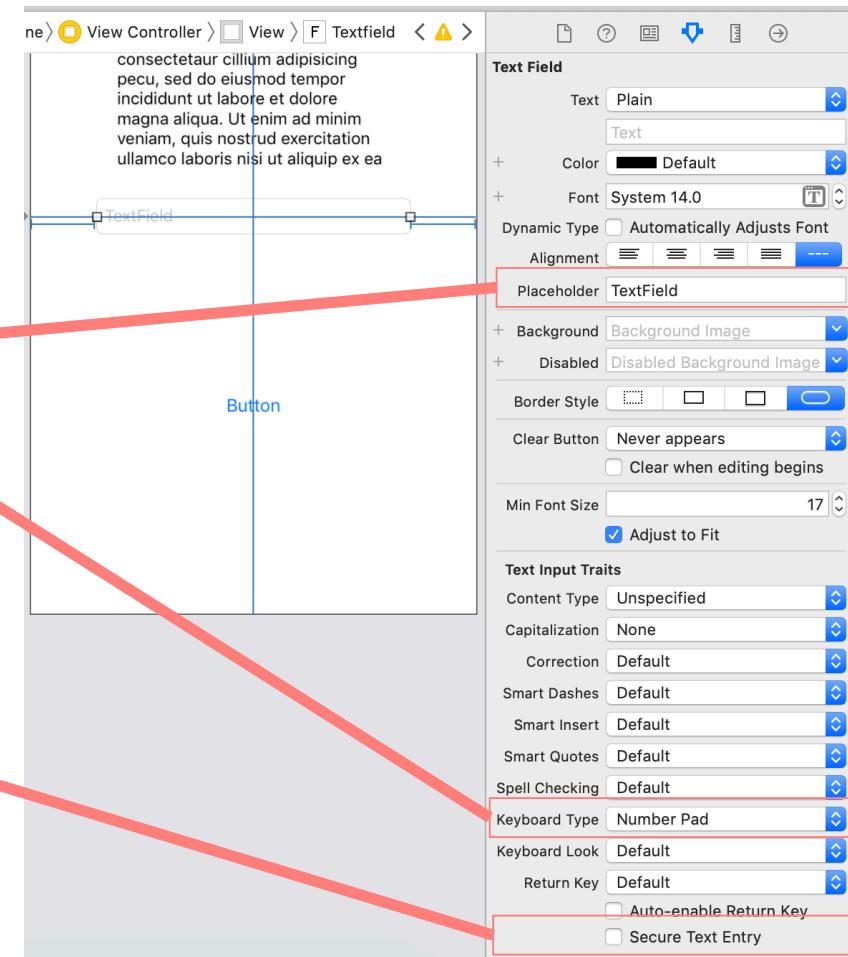


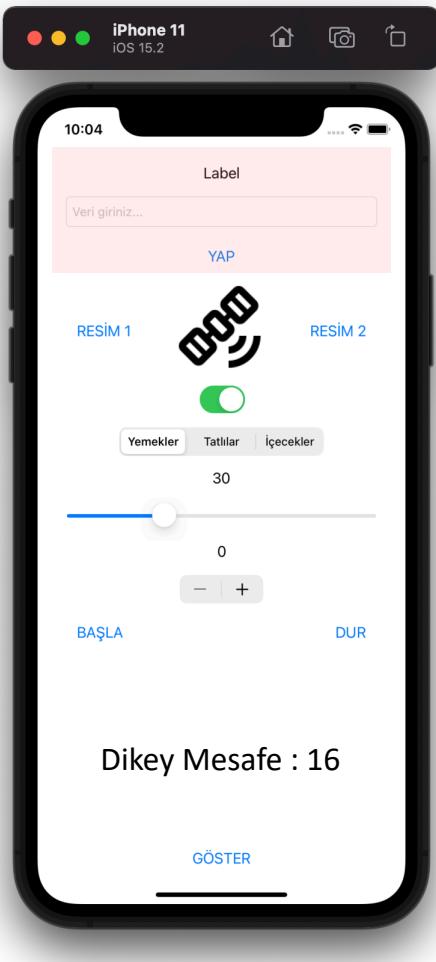
Başa Sayfaya Geçilmesine
rağmen Tab Bar Kaybolmadı.
Show Segue Bütün Özellikleri
bu sayfaya aktardı

IOS Widgets

TextField

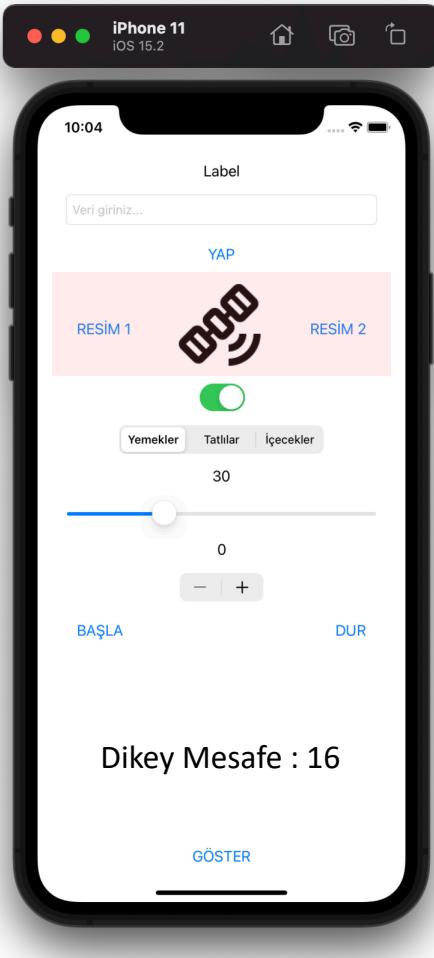
- Textfield içinde yönlendirici yazı koymak için **placeholder** kullanılır.
- **Keyboard Type** ile klavye türü seçilebilir. Kullanıcıya kolaylık sağlar.
- Sadece numara girme için **Number Pad** seçilir.
- Şifre için yazıldığın görülmemesini istemiyorsak **secure text entry**



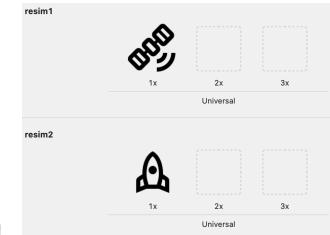


```
class ViewController: UIViewController {  
  
    @IBOutlet weak var labelSonuc: UILabel!  
    @IBOutlet weak var textfieldGirdi: UITextField!  
    @IBOutlet weak var imageView: UIImageView!  
    @IBOutlet weak var mSwitch: UISwitch!  
    @IBOutlet weak var segmentedControl: UISegmentedControl!  
    @IBOutlet weak var labelSlider: UILabel!  
    @IBOutlet weak var slider: UISlider!  
    @IBOutlet weak var labelStepper: UILabel!  
    @IBOutlet weak var stepper: UIStepper!  
    @IBOutlet weak var indicator: UIActivityIndicatorView!
```

```
@IBAction func buttonYap(_ sender: Any) {  
    if let alınanVeri = textfieldGirdi.text {  
        labelSonuc.text = alınanVeri  
    }  
}
```

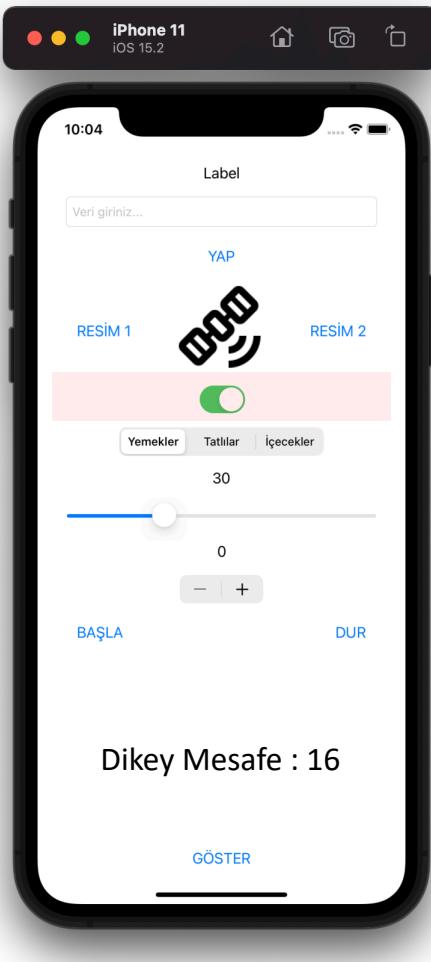


```
class ViewController: UIViewController {  
  
    @IBOutlet weak var labelSonuc: UILabel!  
    @IBOutlet weak var textfieldGirdi: UITextField!  
    @IBOutlet weak var imageView: UIImageView!  
    @IBOutlet weak var mSwitch: UISwitch!  
    @IBOutlet weak var segmentedControl: UISegmentedControl!  
    @IBOutlet weak var labelSlider: UILabel!  
    @IBOutlet weak var slider: UISlider!  
    @IBOutlet weak var labelStepper: UILabel!  
    @IBOutlet weak var stepper: UIStepper!  
    @IBOutlet weak var indicator: UIActivityIndicatorView!
```



48dp

```
@IBAction func buttonResim1(_ sender: Any) {  
    imageView.image = UIImage(named: "resim1")  
}  
@IBAction func buttonResim2(_ sender: Any) {  
    imageView.image = UIImage(named: "resim2")  
}
```



```
class ViewController: UIViewController {

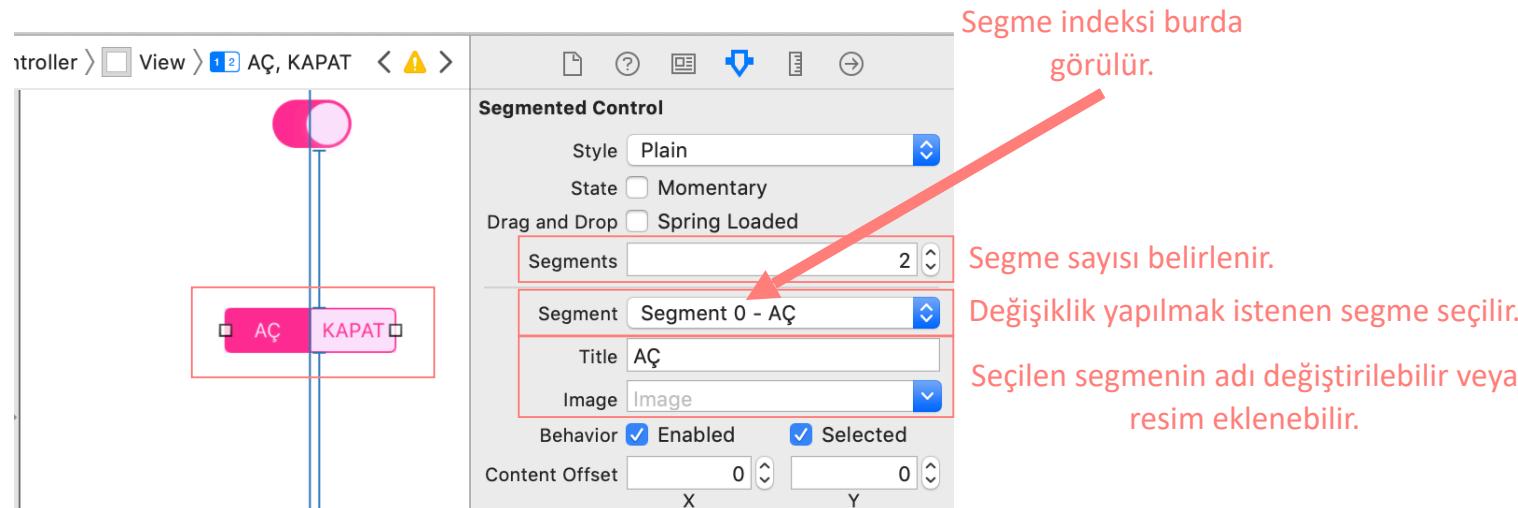
    @IBOutlet weak var labelSonuc: UILabel!
    @IBOutlet weak var textfieldGirdi: UITextField!
    @IBOutlet weak var imageView: UIImageView!
    @IBOutlet weak var mSwitch: UISwitch!
    @IBOutlet weak var segmentedControl: UISegmentedControl!
    @IBOutlet weak var labelSlider: UILabel!
    @IBOutlet weak var slider: UISlider!
    @IBOutlet weak var labelStepper: UILabel!
    @IBOutlet weak var stepper: UIStepper!
    @IBOutlet weak var indicator: UIActivityIndicatorView!

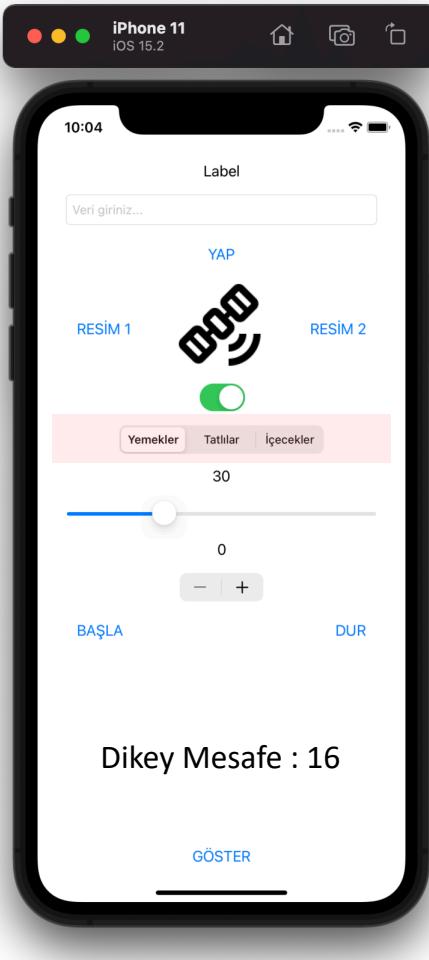
    @IBAction func switchKonumDegistir(_ sender: UISwitch) {
        if sender.isOn {
            print("Switch : ON")
        }else{
            print("Switch : OFF")
        }
    }

    @IBAction func buttonGoster(_ sender: Any) {
        print("Switch durum : \(mSwitch.isOn)")
        print("Segmented en son seçim : \(segmentedControl.titleForSegment(at: segmentedControl.selectedSegmentIndex)!)")
        print("Slider değer : \(Int(slider.value))")
        print("Stepper değer : \(Int(stepper.value))")
    }
}
```

Segmented Control

- Birden fazla seçim sunan buttonlar olarak düşünebiliriz.
- Seçilen segmenin seçildiği anlaşılması için renk değişimi olur.
- Segme sayıları ve içeriklerini değiştirebiliriz.
- Segmelerin index bilgileri önemlidir. Bu index bilgisine göre kodlama yapılır.





```
class ViewController: UIViewController {

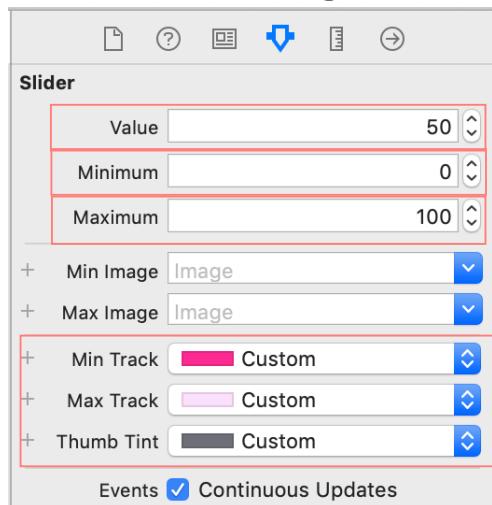
    @IBOutlet weak var labelSonuc: UILabel!
    @IBOutlet weak var textfieldGirdi: UITextField!
    @IBOutlet weak var imageView: UIImageView!
    @IBOutlet weak var mSwitch: UISwitch!
    @IBOutlet weak var segmentedControl: UISegmentedControl!
    @IBOutlet weak var labelSlider: UILabel!
    @IBOutlet weak var slider: UISlider!
    @IBOutlet weak var labelStepper: UILabel!
    @IBOutlet weak var stepper: UIStepper!
    @IBOutlet weak var indicator: UIActivityIndicatorView!

    @IBAction func segmentedDegisimKontrol(_ sender: UISegmentedControl) {
        let secilenIndeks = sender.selectedSegmentIndex
        print("Seçim : \(sender.titleForSegment(at: secilenIndeks)!)")
    }

    @IBAction func buttonGoster(_ sender: Any) {
        print("Switch durum : \(mSwitch.isOn)")
        print("Segmented en son seçim : \(segmentedControl.titleForSegment(at: segmentedControl.selectedSegmentIndex)!)")
        print("Slider değer : \(Int(slider.value))")
        print("Stepper değer : \(Int(stepper.value))")
    }
}
```

Slider

- Slider, dokunmaya duyarlı bir slayt türündür.
- Belirli aralık arasında çalışır.
- **value** özelliği ile anlık sonuç alınır.
- **Float** olarak değer döndürür.

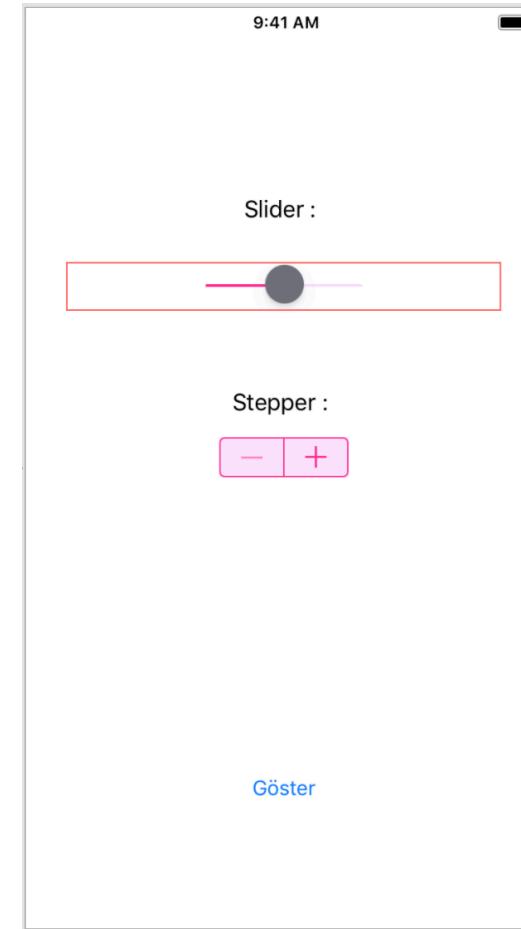


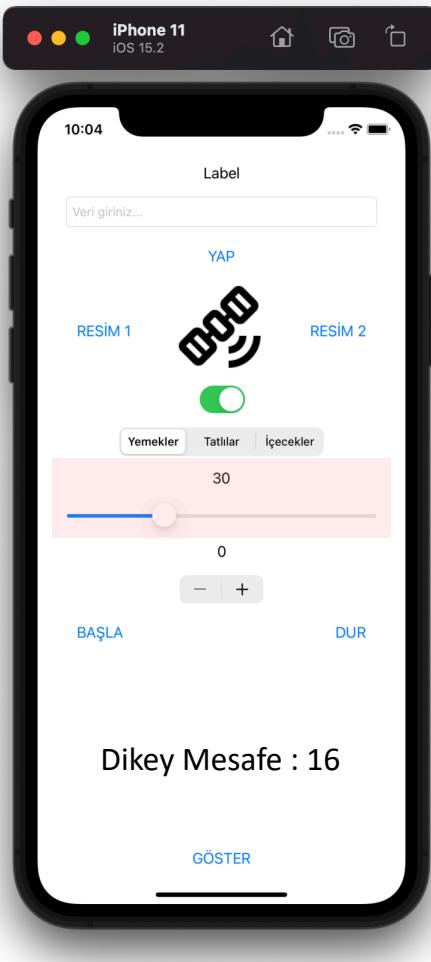
Anlık slider değeri

Minimum değeri

Maksimum değeri

Renk değişimi





```
class ViewController: UIViewController {

    @IBOutlet weak var labelSonuc: UILabel!
    @IBOutlet weak var textfieldGirdi: UITextField!
    @IBOutlet weak var imageView: UIImageView!
    @IBOutlet weak var mSwitch: UISwitch!
    @IBOutlet weak var segmentedControl: UISegmentedControl!
    @IBOutlet weak var labelSlider: UILabel!
    @IBOutlet weak var slider: UISlider!
    @IBOutlet weak var labelStepper: UILabel!
    @IBOutlet weak var stepper: UIStepper!
    @IBOutlet weak var indicator: UIActivityIndicatorView!

    override func viewDidLoad() {
        super.viewDidLoad()
        labelSlider.text = String(Int(slider.value))
        indicator.isHidden = true
    }

    @IBAction func sliderDegisimKontrol(_ sender: UISlider) {
        labelSlider.text = String(Int(sender.value))
    }

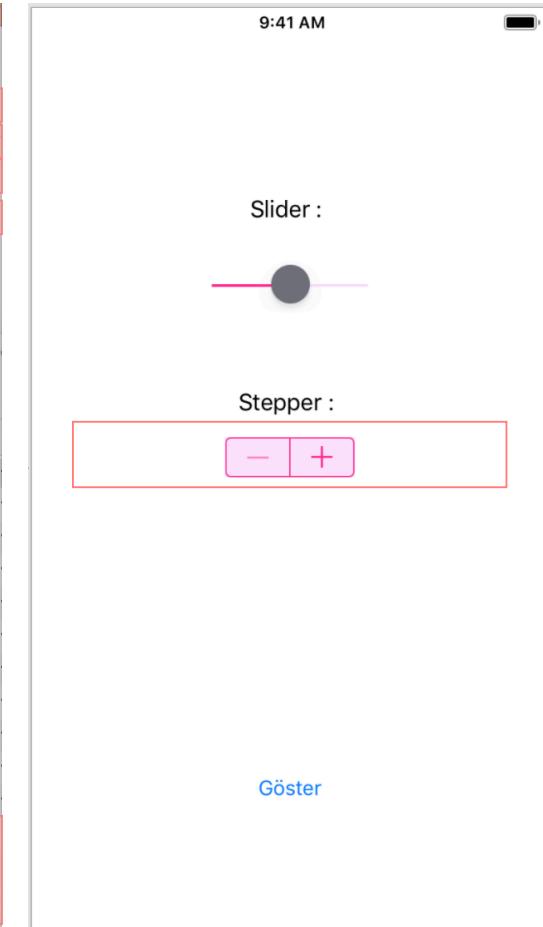
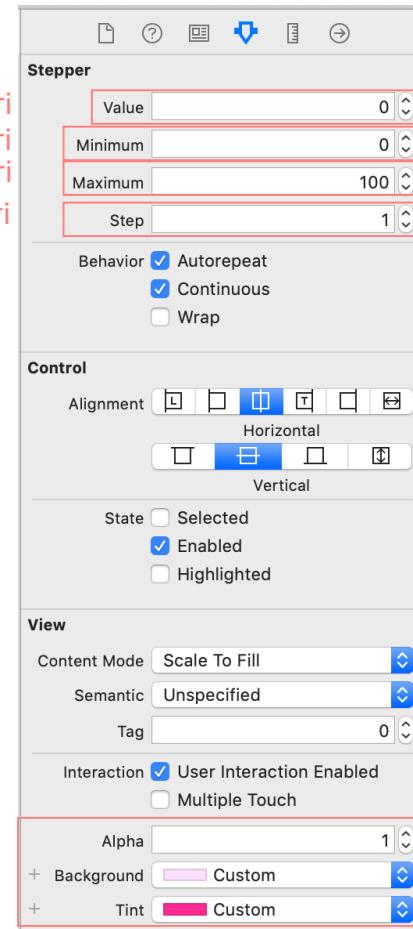
    @IBAction func buttonGoster(_ sender: Any) {
        print("Switch durum : \(mSwitch.isOn)")
        print("Segmented en son seçim : \(segmentedControl.titleForSegment(at: segmentedControl.selectedSegmentIndex)!)")
        print("Slider değer : \(Int(slider.value))")
        print("Stepper değer : \(Int(stepper.value))")
    }
}
```

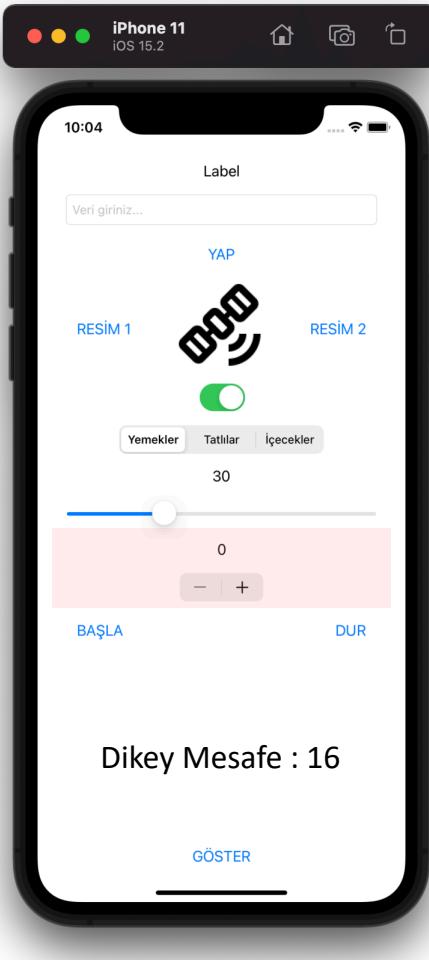
Stepper

Anlık stepper değeri
Minimum değeri
Maksimum değeri
Adım artış değeri

- Tıklanılmaya duyarlı artış ve azalış yapan buttonlardır.
- Belirli aralık arasında çalışır.
- **value** özelliği ile anlık sonuç alınır.
- **Double** olarak değer döndürür.

Renk Değişimi



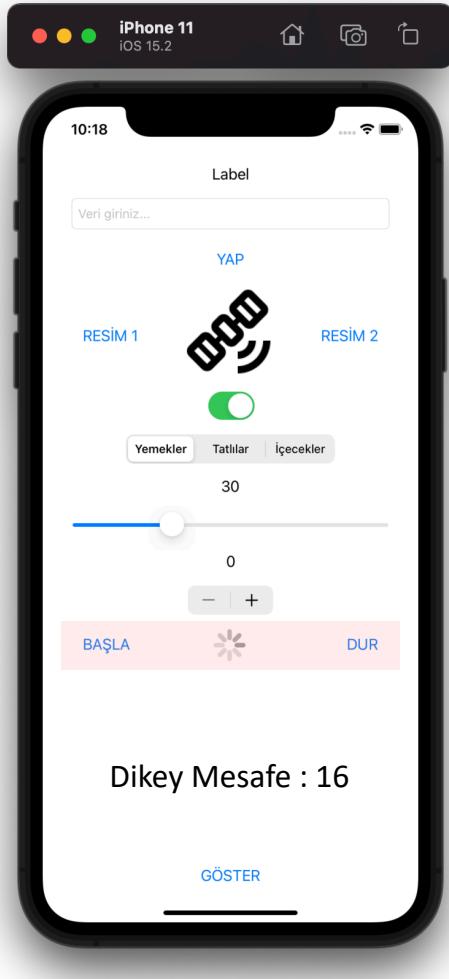


```
class ViewController: UIViewController {

    @IBOutlet weak var labelSonuc: UILabel!
    @IBOutlet weak var textfieldGirdi: UITextField!
    @IBOutlet weak var imageView: UIImageView!
    @IBOutlet weak var mSwitch: UISwitch!
    @IBOutlet weak var segmentedControl: UISegmentedControl!
    @IBOutlet weak var labelSlider: UILabel!
    @IBOutlet weak var slider: UISlider!
    @IBOutlet weak var labelStepper: UILabel!
    @IBOutlet weak var stepper: UIStepper!
    @IBOutlet weak var indicator: UIActivityIndicatorView!

    @IBAction func stepperDegisimKontrol(_ sender: UIStepper) {
        labelStepper.text = String(Int(sender.value))
    }

    @IBAction func buttonGoster(_ sender: Any) {
        print("Switch durum : \(mSwitch.isOn)")
        print("Segmented en son seçim : \(segmentedControl.titleForSegment(at: segmentedControl.selectedSegmentIndex)!)")
        print("Slider değer : \(Int(slider.value))")
        print("Stepper değer : \(Int(stepper.value))")
    }
}
```



```
class ViewController: UIViewController {

    @IBOutlet weak var labelSonuc: UILabel!
    @IBOutlet weak var textfieldGirdi: UITextField!
    @IBOutlet weak var imageView: UIImageView!
    @IBOutlet weak var mSwitch: UISwitch!
    @IBOutlet weak var segmentedControl: UISegmentedControl!
    @IBOutlet weak var labelSlider: UILabel!
    @IBOutlet weak var slider: UISlider!
    @IBOutlet weak var labelStepper: UILabel!
    @IBOutlet weak var stepper: UIStepper!
    @IBOutlet weak var indicator: UIActivityIndicatorView!

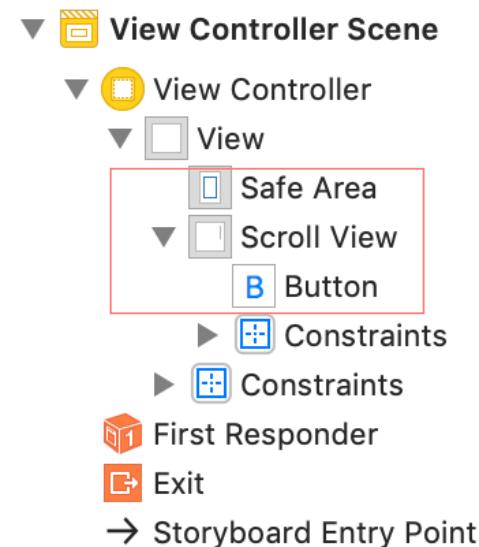
    override func viewDidLoad() {
        super.viewDidLoad()
        labelSlider.text = String(Int(slider.value))
        indicator.isHidden = true
    }

    @IBAction func buttonBasla(_ sender: Any) {
        indicator.isHidden = false
        indicator.startAnimating()
    }

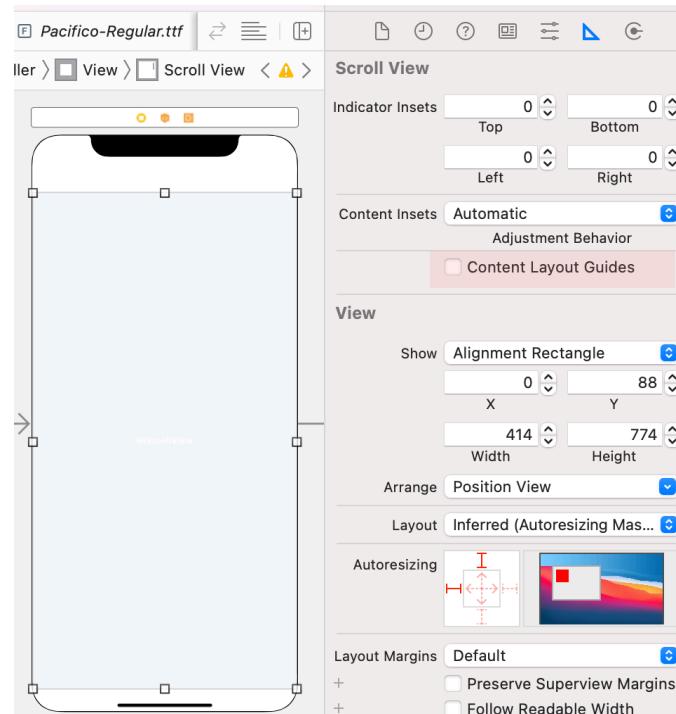
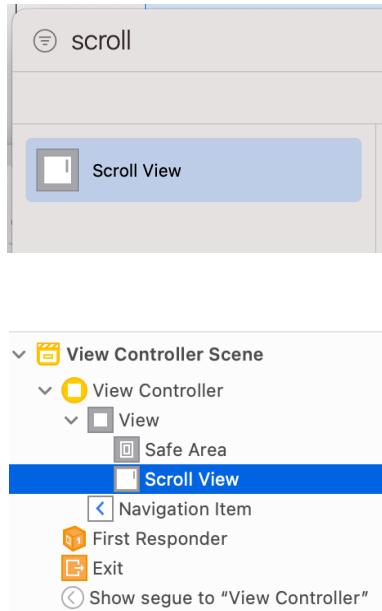
    @IBAction func buttonDur(_ sender: Any) {
        indicator.isHidden = true
        indicator.stopAnimating()
    }
}
```

ScrollView

- İçerisindeki görsel nesneler ekrandan taşıyorsa scroll özelliği gelmektedir.
- Küçük ekranlar için kullanılabilir.
- Tasarım alanı gibi en dışta durmalıdır.
- Diğer görsel nesneler içerisinde yer almmalıdır.

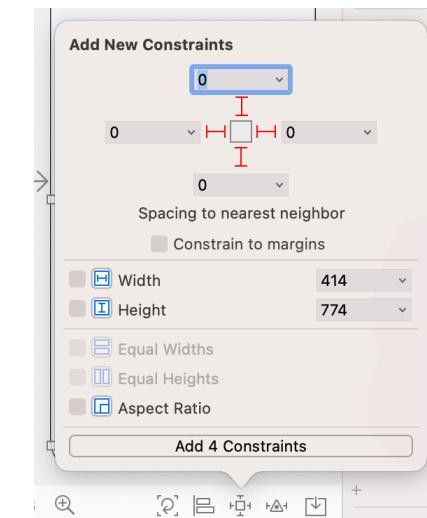


ScrollView Ekleme



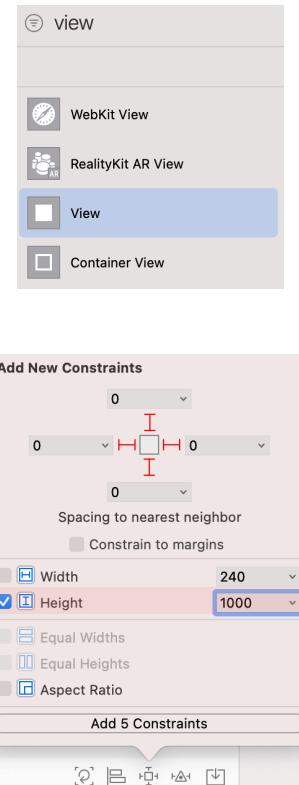
ScrollView safe area sınırlarına eklenmelidir.
Örnek Navigation controller altına gibi.

Bu özellik kaldırılmalıdır.

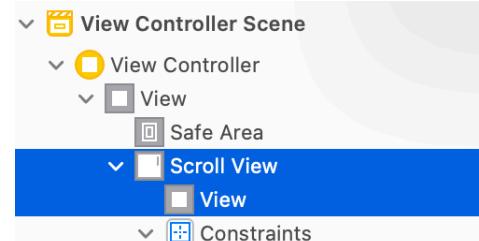


ScrollView İçinde Tasarım Yapma

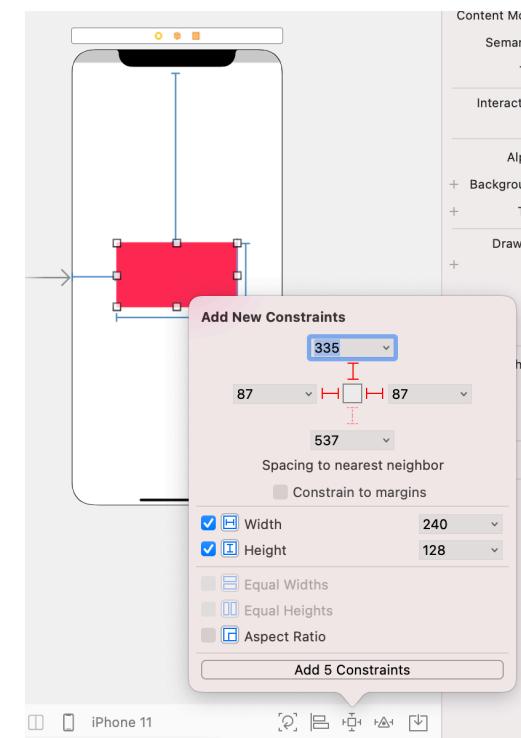
- Tasarımı doğru yapabilmek için scrollView içine view eklenmelidir.



ScrollView ve View Seçilir ardından genişlik eşitlenir



Test için view eklenebilir.



IOS Kullanıcı Etkileşimi

Basit Alert

```
@IBAction func buttonAlert(_ sender: Any) {
    let alertController = UIAlertController(title: "Başlık", message: "Mesaj",
                                            preferredStyle: .alert)

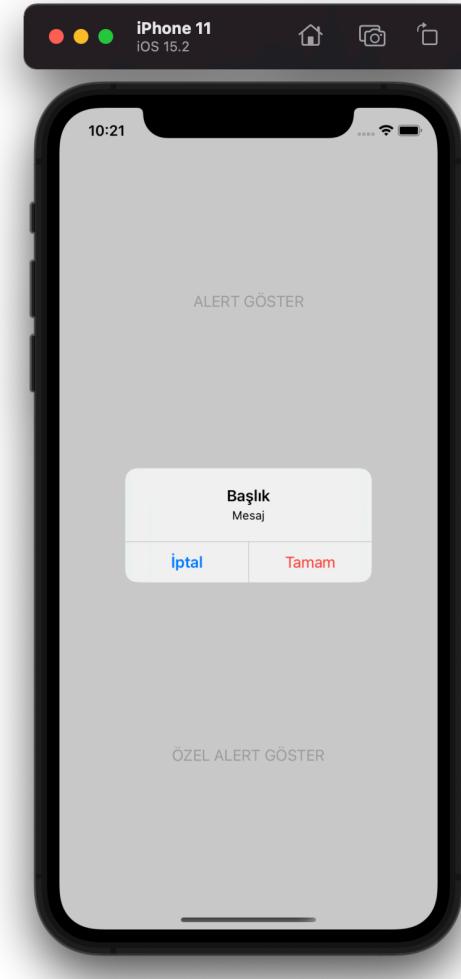
    let iptalAction = UIAlertAction(title: "İptal", style: .cancel){ action in
        print("İptal Seçildi")
    }

    alertController.addAction(iptalAction)

    let tamamAction = UIAlertAction(title: "Tamam", style: .destructive){ action in
        print("Tamam Seçildi")
    }

    alertController.addAction(tamamAction)

    self.present(alertController, animated: true)
}
```



Action Sheet

```
@IBAction func buttonActionSheet(_ sender: Any) {  
    let alertController = UIAlertController(title: "Başlık", message: "Mesaj",  
        preferredStyle: .actionSheet)  
  
    let iptalAction = UIAlertAction(title: "İptal", style: .cancel){ action in  
        print("İptal Seçildi")  
    }  
  
    alertController.addAction(iptalAction)  
  
    let tamamAction = UIAlertAction(title: "Tamam", style: .destructive){ action in  
        print("Tamam Seçildi")  
    }  
  
    alertController.addAction(tamamAction)  
  
    self.present(alertController, animated: true)  
}
```



Özel Alert

```
@IBAction func buttonOzelAlert(_ sender: Any) {
    let alertController = UIAlertController(title: "Başlık", message: "Mesaj",
                                            preferredStyle: .alert)

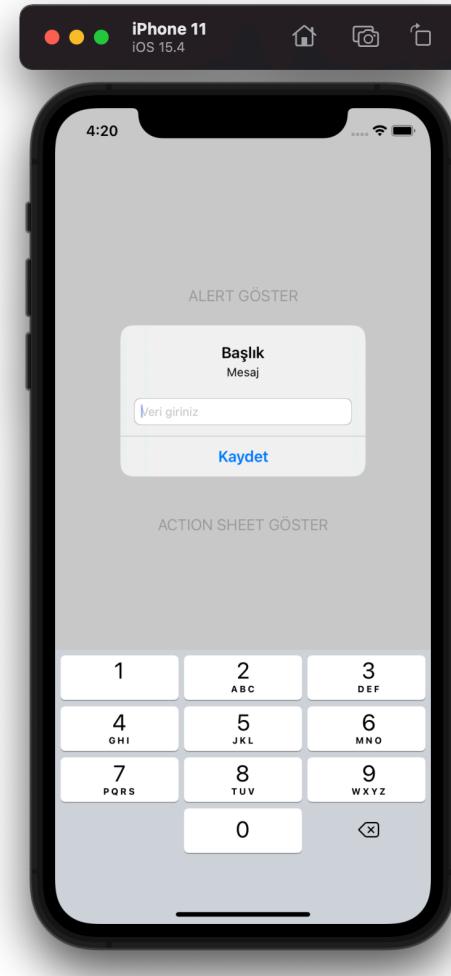
    alertController.addTextField { textfield in
        textfield.placeholder = "Veri giriniz"
        textfield.keyboardType = .numberPad
        textfield.isSecureTextEntry = true
    }

    let kaydetAction = UIAlertAction(title: "Kaydet", style: .cancel){ action in
        let textField = alertController.textFields![0] as UITextField

        if let alinanVeri = textField.text {
            print("Veri : \(alinanVeri)")
        }
    }

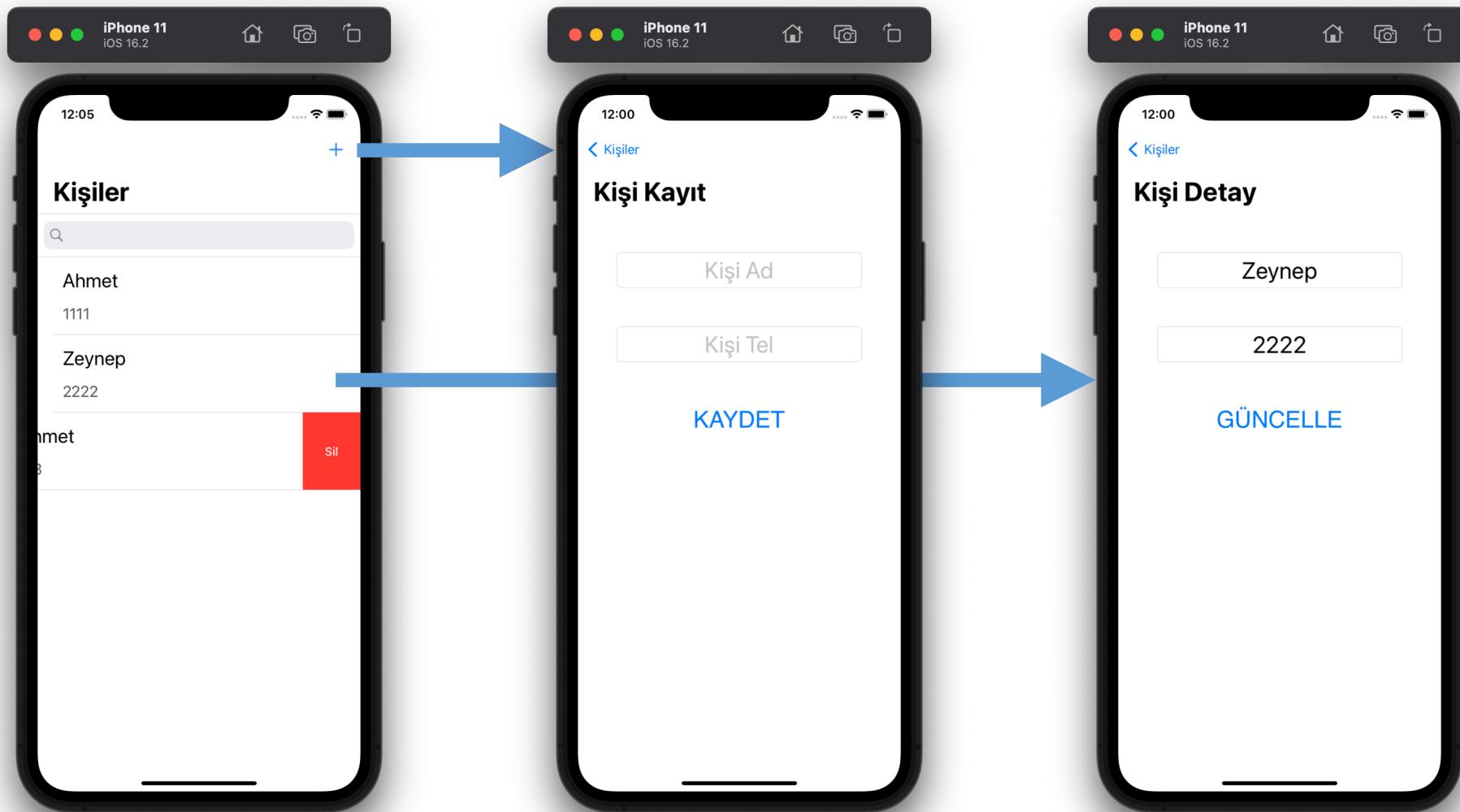
    alertController.addAction(kaydetAction)

    self.present(alertController, animated: true)
}
```



Listeleme

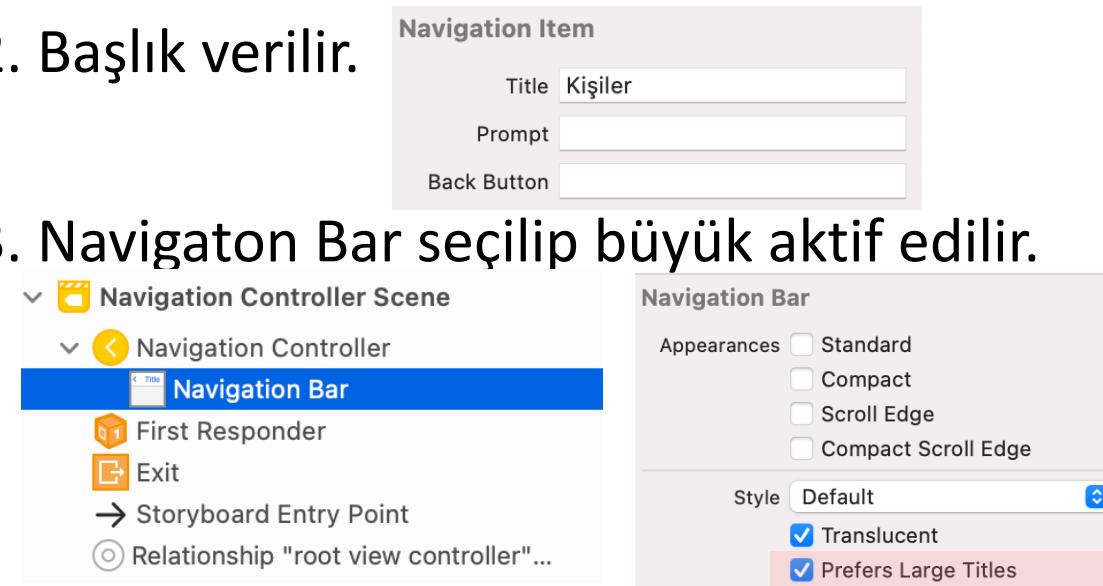
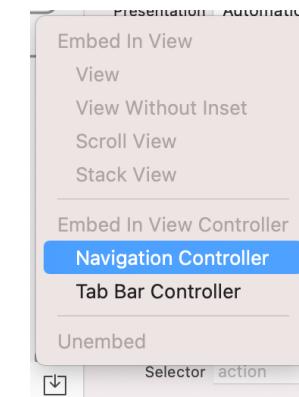
Kişiler Uygulaması

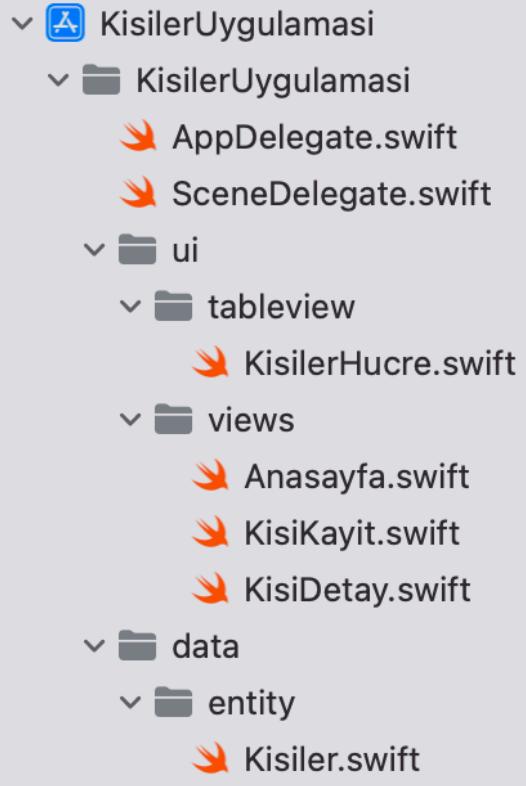




AnasayfaVC Tasarım

- 1. Navigation Controller eklenir.
- 2. Başlık verilir.
- 3. Navigaton Bar seçili büyük aktif edilir.
- 4. + işaretini için Bar Button Item eklenir.





```
class Kisiler {
    var kisi_id:Int?
    var kisi_ad:String?
    var kisi_tel:String?

    init(){
    }

    init(kisi_id: Int, kisi_ad: String, kisi_tel: String) {
        self.kisi_id = kisi_id
        self.kisi_ad = kisi_ad
        self.kisi_tel = kisi_tel
    }
}
```

```

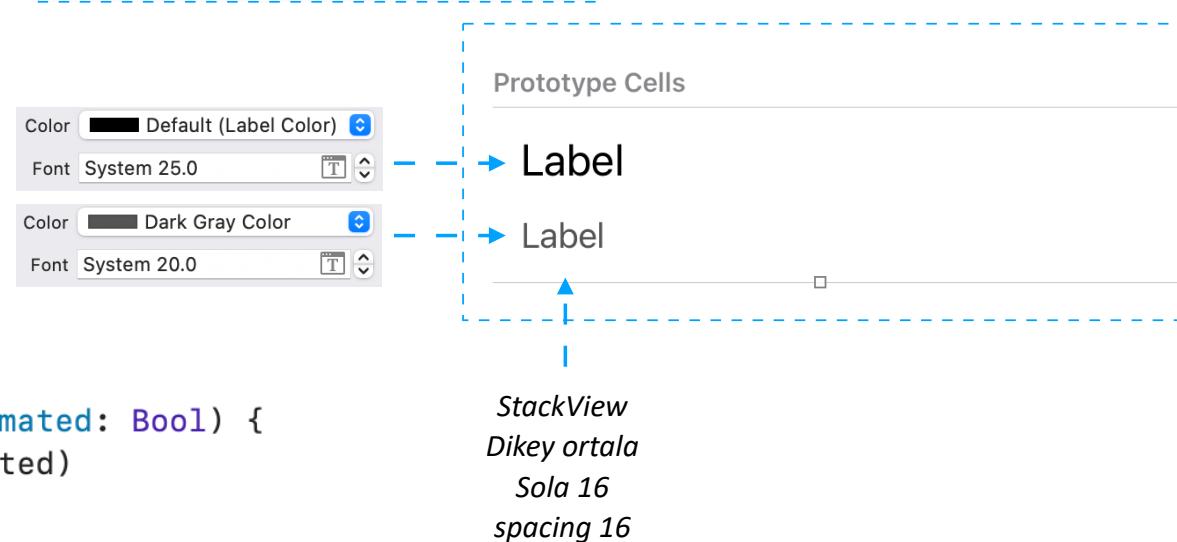
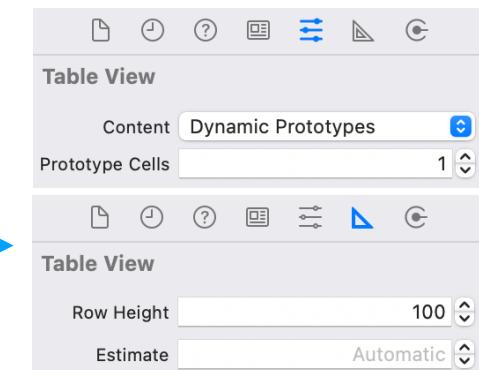
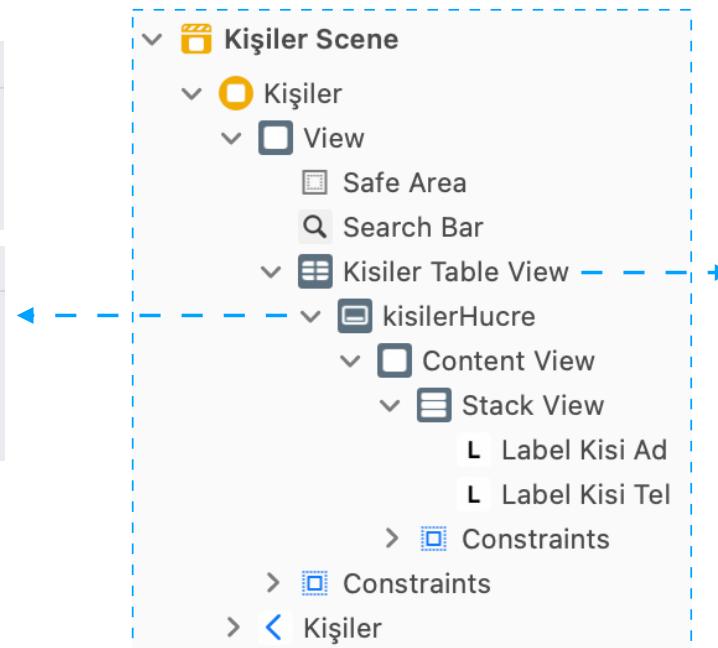
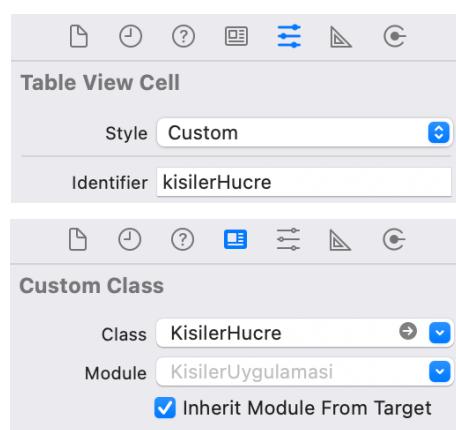
class KisilerHucre: UITableViewCell {

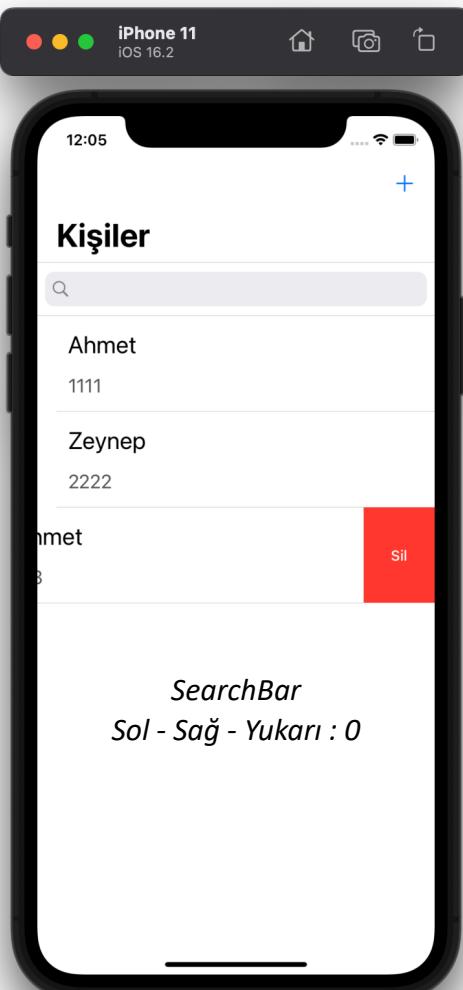
    @IBOutlet weak var labelKisiTel: UILabel!
    @IBOutlet weak var labelKisiAd: UILabel!

    override func awakeFromNib() {
        super.awakeFromNib()
    }

    override func setSelected(_ selected: Bool, animated: Bool) {
        super.setSelected(selected, animated: animated)
    }
}

```





```
class Anasayfa : UIViewController {
    @IBOutlet weak var searchBar: UISearchBar!
    @IBOutlet weak var kisilerTableView: UITableView!
    var kisilerListesi = [Kisiler]()

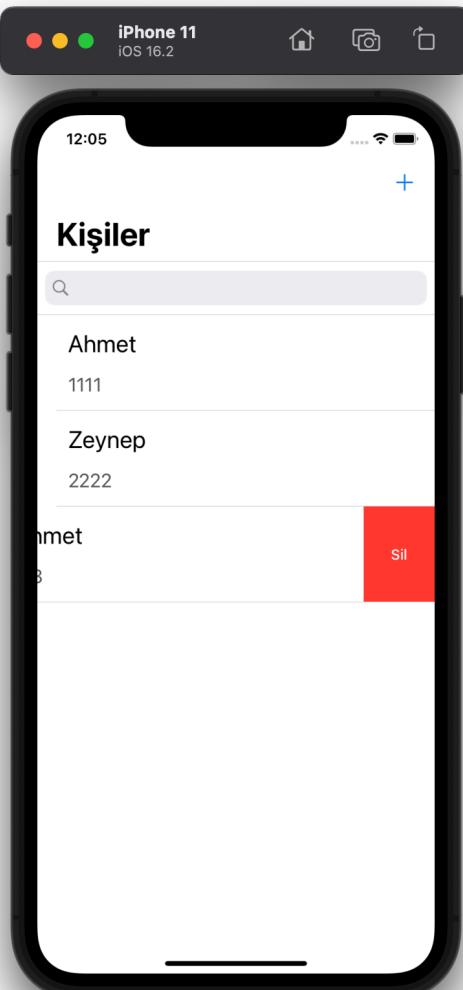
    override func viewDidLoad() {
        super.viewDidLoad()
        searchBar.delegate = self
        kisilerTableView.delegate = self
        kisilerTableView.dataSource = self

        let k1 = Kisiler(kisi_id: 1, kisi_ad: "Ahmet", kisi_tel: "1111")
        let k2 = Kisiler(kisi_id: 2, kisi_ad: "Zeynep", kisi_tel: "2222")
        let k3 = Kisiler(kisi_id: 3, kisi_ad: "Beyza", kisi_tel: "3333")
        kisilerListesi.append(k1)
        kisilerListesi.append(k2)
        kisilerListesi.append(k3)
    }

    override func viewWillAppear(_ animated: Bool) {
        print("Anasayfaya döndü")
    }

    override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
        if segue.identifier == "toDetay" {
            if let kisi = sender as? Kisiler {
                let gidilecekVC = segue.destination as! KisiDetay
                gidilecekVC.kisi = kisi
            }
        }
    }
}

extension Anasayfa : UISearchBarDelegate {
    func searchBar(_ searchBar: UISearchBar, textDidChange searchText: String) {
        print("Kişi Ara : \(searchText)")
    }
}
```



```
extension Anasayfa : UITableViewDelegate,UITableViewDataSource {
    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return kisilerListesi.count
    }

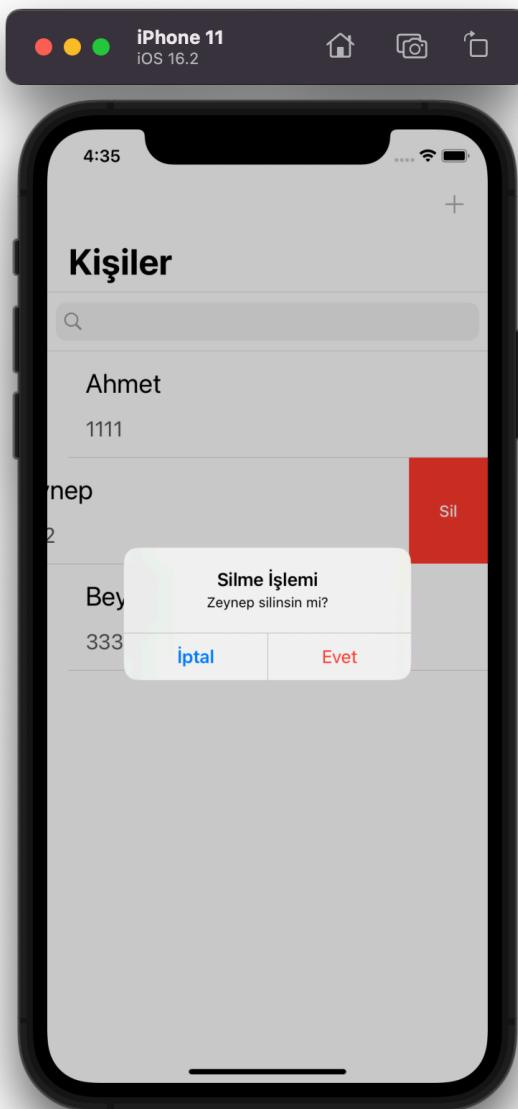
    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        let kisi = kisilerListesi[indexPath.row]

        let hucre = tableView.dequeueReusableCell(withIdentifier: "kisilerHucre") as!
            KisilerHucre

        hucre.labelKisiAd.text = kisi.kisi_ad
        hucre.labelKisiTel.text = kisi.kisi_tel

        return hucre
    }

    func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
        let kisi = kisilerListesi[indexPath.row]
        performSegue(withIdentifier: "toDetay", sender: kisi)
        tableView.deselectRow(at: indexPath, animated: true)//Seçim kaldırma
    }
}
```



```
func tableView(_ tableView: UITableView, trailingSwipeActionsConfigurationForRowAt indexPath: IndexPath) -> UISwipeActionsConfiguration? {  
  
    let silAction = UIContextualAction(style: .destructive, title: "Sil"){  
        contextualAction,view,bool in  
        let kisi = self.kisilerListesi[indexPath.row]  
  
        let alert = UIAlertController(title: "Silme İşlemi", message: "\((kisi.kisi_ad!)  
silinsin mi?", preferredStyle: .alert)  
  
        let iptalAction = UIAlertAction(title: "İptal", style: .cancel)  
        alert.addAction(iptalAction)  
  
        let evetAction = UIAlertAction(title: "Evet", style: .destructive){ action in  
            print("Kişi Sil : \((kisi.kisi_id!))")  
        }  
        alert.addAction(evetAction)  
  
        self.present(alert, animated: true)  
    }  
  
    return UISwipeActionsConfiguration(actions: [silAction])  
}
```



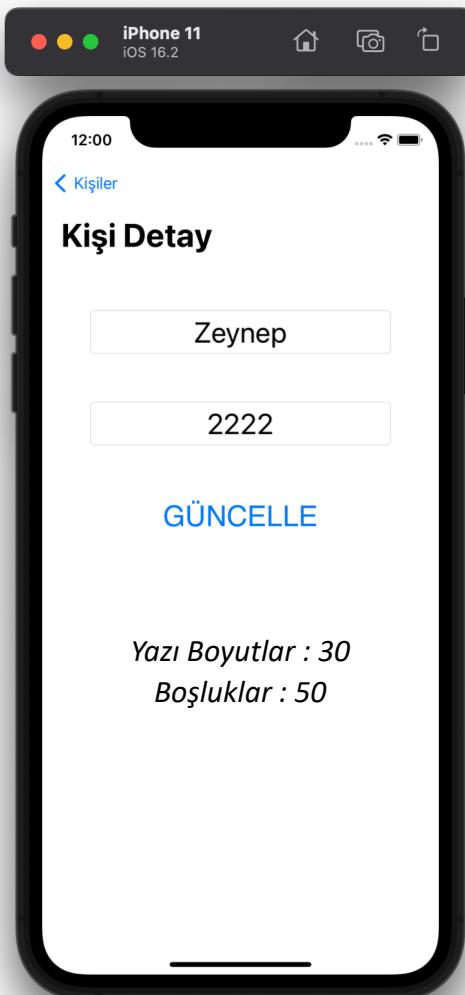
```
class KisiKayit: UIViewController {

    @IBOutlet weak var tfKisiAd: UITextField!
    @IBOutlet weak var tfKisiTel: UITextField!

    override func viewDidLoad() {
        super.viewDidLoad()
    }

    @IBAction func buttonKaydet(_ sender: Any) {
        if let ka = tfKisiAd.text,let kt = tfKisiTel.text {
            kaydet(kisi_ad: ka, kisi_tel: kt)
        }
    }

    func kaydet(kisi_ad:String,kisi_tel:String){
        print("Kişi Kaydet : \(kisi_ad) - \(kisi_tel)")
    }
}
```



```
class KisiDetay: UIViewController {

    @IBOutlet weak var tfKisiTel: UITextField!
    @IBOutlet weak var tfKisiAd: UITextField!

    var kisi:Kisiler?

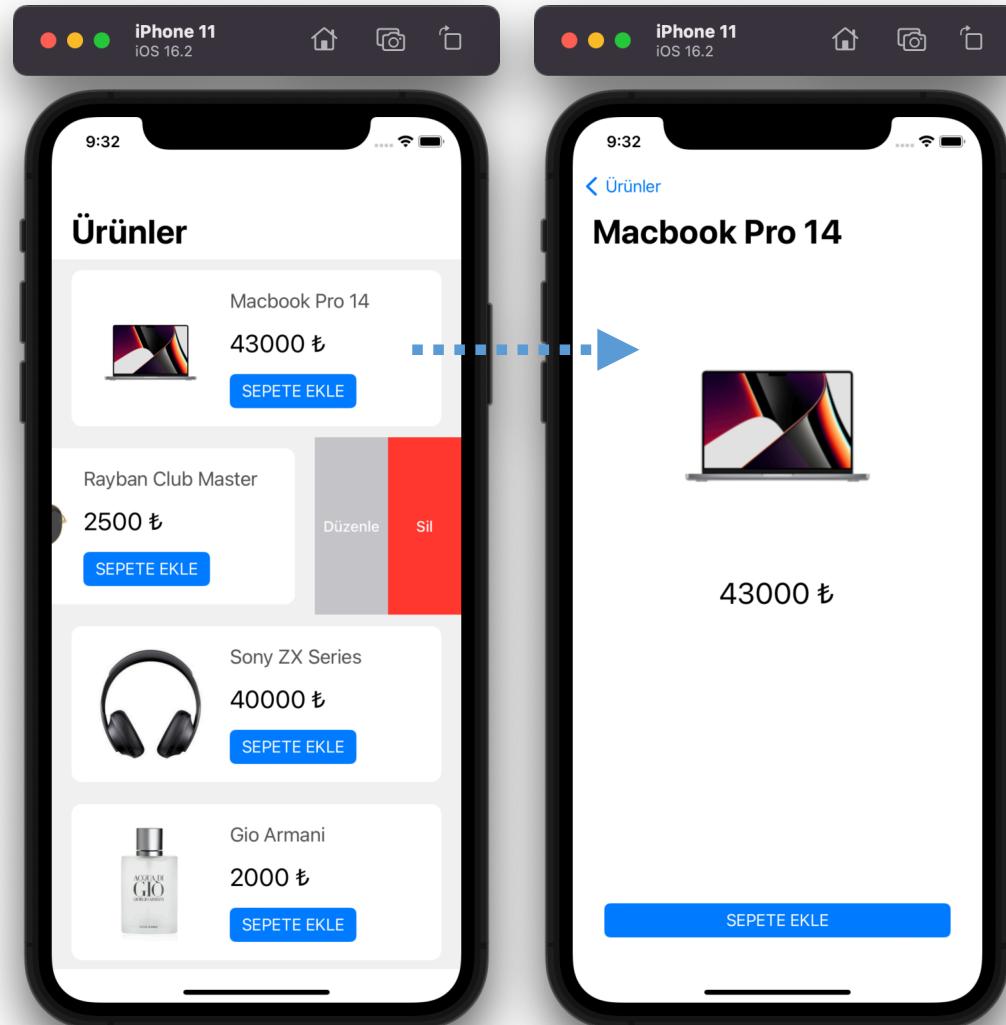
    override func viewDidLoad() {
        super.viewDidLoad()
        if let k = kisi {
            tfKisiAd.text = k.kisi_ad
            tfKisiTel.text = k.kisi_tel
        }
    }

    @IBAction func buttonGuncelle(_ sender: Any) {
        if let ka = tfKisiAd.text,let kt = tfKisiTel.text,let k = kisi {
            guncelle(kisi_id: k.kisi_id!, kisi_ad: ka, kisi_tel: kt)
        }
    }

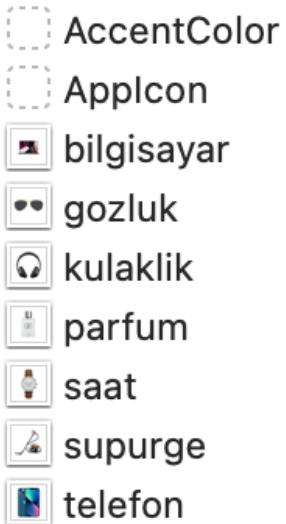
    func guncelle(kisi_id:Int,kisi_ad:String,kisi_tel:String){
        print("Kişi Güncelle : \(kisi_id) - \(kisi_ad) - \(kisi_tel)")
    }
}
```

Detaylı TableView

Ürünler Uygulaması



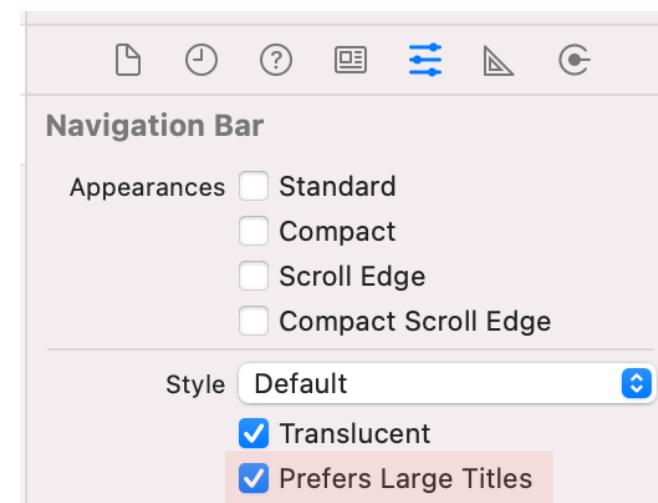
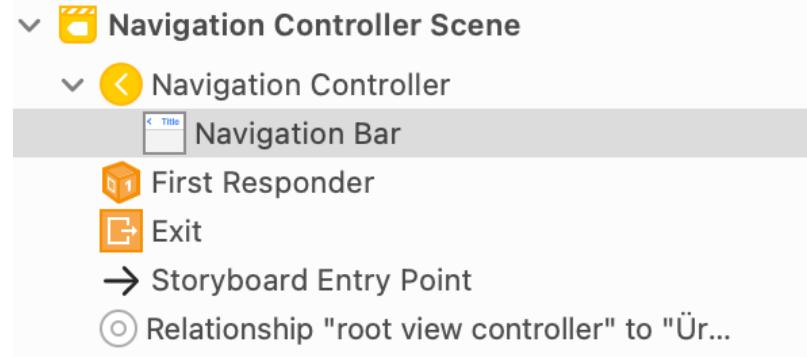
Assets dosyasına resim ekle



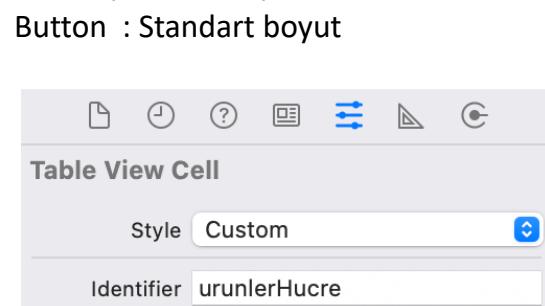
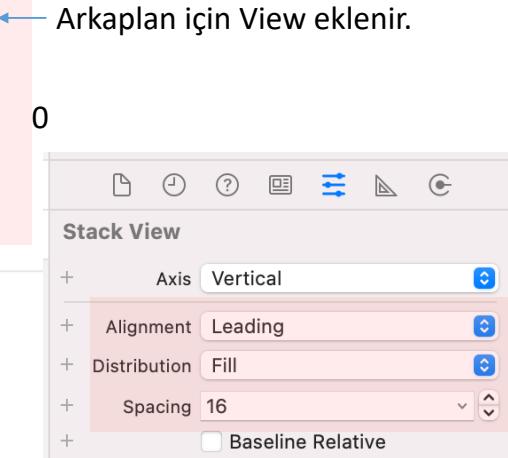
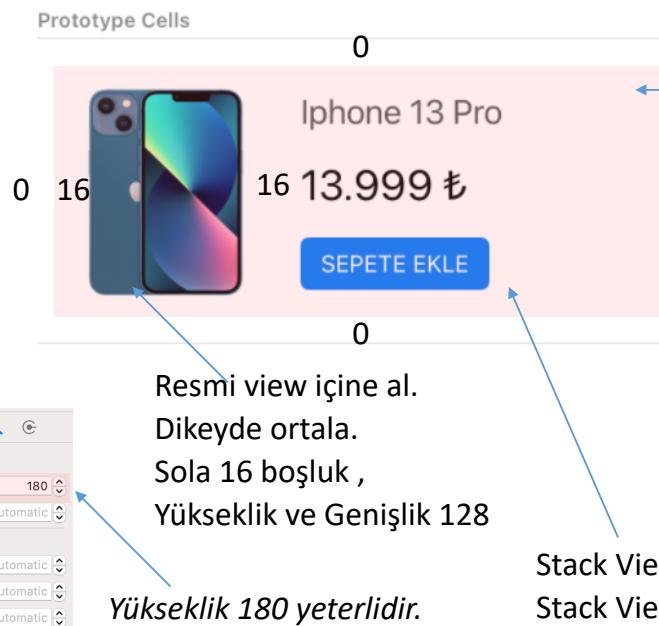
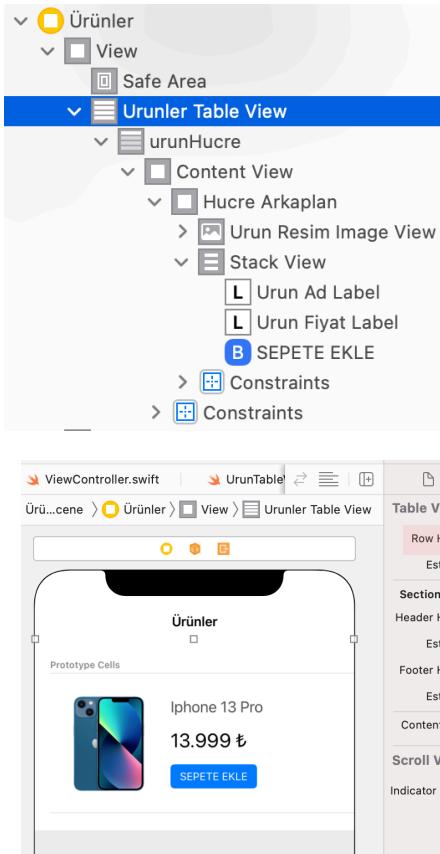
Model sınıfı oluştur.

```
class Urunler {  
    var id:Int?  
    var ad:String?  
    var resim:String?  
    var fiyat:Int?  
  
    init(){  
    }  
  
    init(id: Int, ad: String, resim: String, fiyat: Int) {  
        self.id = id  
        self.ad = ad  
        self.resim = resim  
        self.fiyat = fiyat  
    }  
}
```

Navigation Controller Ekleme



Tasarımın Oluşturulması



```
protocol HocreProtocol {
    func sepeteEkleTiklandi(indexPath:IndexPath)
}

class UrunlerHocre: UITableViewCell {
    @IBOutlet weak var imageViewUrun: UIImageView!
    @IBOutlet weak var labelUrunFiyat: UILabel!
    @IBOutlet weak var labelUrunAd: UILabel!
    @IBOutlet weak var hucreArkaplan: UIView!

    var hucreProtocol:HocreProtocol?
    var indexPath:IndexPath?

    override func awakeFromNib() {
        super.awakeFromNib()
        // Initialization code
    }

    override func setSelected(_ selected: Bool, animated: Bool) {
        super.setSelected(selected, animated: animated)

        // Configure the view for the selected state
    }

    @IBAction func buttonSepeteEkle(_ sender: Any) {
        hucreProtocol?.sepeteEkleTiklandi(indexPath: indexPath!)
    }
}
```

```

class Anasayfa: UIViewController {
    @IBOutlet weak var urunlerTableView: UITableView!
    var urunlerListesi = [Urunler]()

    override func viewDidLoad() {
        super.viewDidLoad()

        urunlerTableView.delegate = self
        urunlerTableView.dataSource = self

        let u1 = Urunler(id: 1 , ad: "Macbook Pro 14", resim : "bilgisayar",fiyat: 43000)
        let u2 = Urunler(id: 2 , ad: "Rayban Club Master", resim : "gozluk",fiyat: 2500)
        let u3 = Urunler(id: 3 , ad: "Sony ZX Series", resim : "kulaklik",fiyat: 40000)
        let u4 = Urunler(id: 4 , ad: "Gio Armani", resim : "parfum",fiyat: 2000)
        let u5 = Urunler(id: 5 , ad: "Casio X Series", resim : "saat",fiyat: 8000)
        let u6 = Urunler(id: 6 , ad: "Dyson V8", resim : "supurge",fiyat: 18000)
        let u7 = Urunler(id: 7 , ad: "IPhone 13", resim : "telefon",fiyat: 32000)

        urunlerListesi.append(u1)
        urunlerListesi.append(u2)
        urunlerListesi.append(u3)
        urunlerListesi.append(u4)
        urunlerListesi.append(u5)
        urunlerListesi.append(u6)
        urunlerListesi.append(u7)

        urunlerTableView.separatorColor = UIColor(white: 0.95, alpha: 1)
    }
}

```

```

let u1 = Urunler(id: 1 , ad: "Macbook Pro 14", resim : "bilgisayar",fiyat: 43000)
let u2 = Urunler(id: 2 , ad: "Rayban Club Master", resim : "gozluk",fiyat: 2500)
let u3 = Urunler(id: 3 , ad: "Sony ZX Series", resim : "kulaklik",fiyat: 40000)
let u4 = Urunler(id: 4 , ad: "Gio Armani", resim : "parfum",fiyat: 2000)
let u5 = Urunler(id: 5 , ad: "Casio X Series", resim : "saat",fiyat: 8000)
let u6 = Urunler(id: 6 , ad: "Dyson V8", resim : "supurge",fiyat: 18000)
let u7 = Urunler(id: 7 , ad: "IPhone 13", resim : "telefon",fiyat: 32000)

urunlerListesi.append(u1)
urunlerListesi.append(u2)
urunlerListesi.append(u3)
urunlerListesi.append(u4)
urunlerListesi.append(u5)
urunlerListesi.append(u6)
urunlerListesi.append(u7)

extension Anasayfa : UITableViewDelegate,UITableViewDataSource,HucreProtocol {
    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return urunlerListesi.count
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) ->
        UITableViewCell {
        let urun = urunlerListesi[indexPath.row]

        let hucre = tableView.dequeueReusableCell(withIdentifier: "urunlerHucre") as!
            UrunlerHucre

        hucre.imageViewUrun.image = UIImage(named: urun.resim!)
        hucre.labelUrunAd.text = urun.ad
        hucre.labelUrunFiyat.text = "\((urun.fiyat!) ₺"

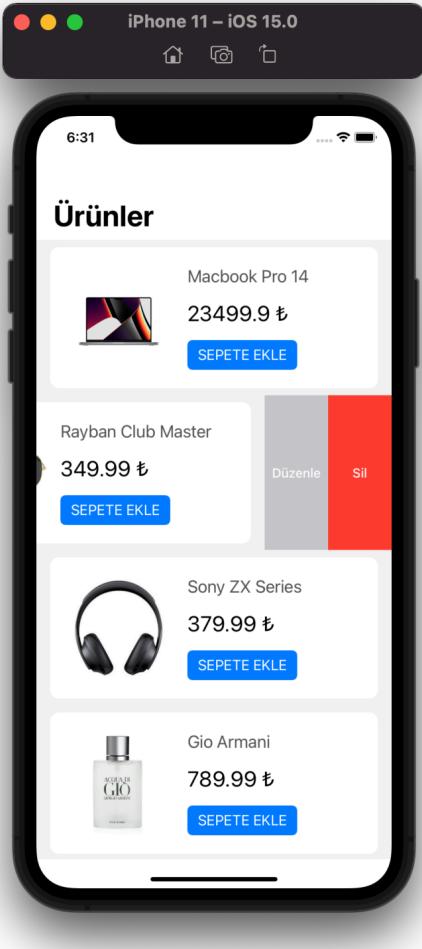
        hucre.backgroundColor = UIColor(white: 0.95, alpha: 1)
        hucre.hucreArkaplan.layer.cornerRadius = 10.0

        hucre.selectionStyle = .none

        hucre.hucreProtocol = self
        hucreIndexPath = indexPath

        return hucre
    }
}

```



Action Ekleme

```
func tableView(_ tableView: UITableView, trailingSwipeActionsConfigurationForRowAt indexPath: IndexPath) -> UISwipeActionsConfiguration? {
    let urun = urunlerListe[indexPath.row]

    let silAction = UIContextualAction(style: .destructive, title: "Sil") {(contextualAction,view,bool) in
        print("\(urun.urun_ad!) silindi")
    }

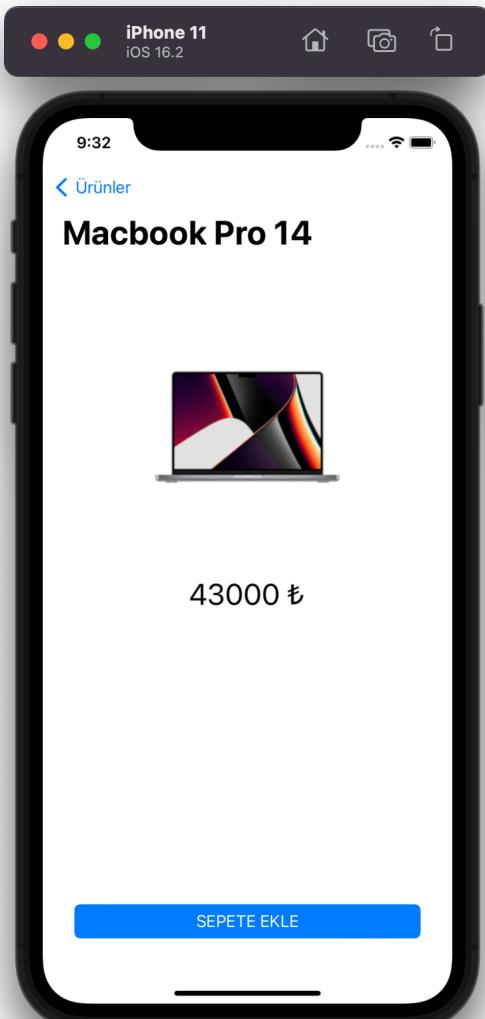
    let duzenleAction = UIContextualAction(style: .normal, title: "Düzenle") {(contextualAction,view,bool) in
        print("\(urun.urun_ad!) düzenlenendi")
    }

    return UISwipeActionsConfiguration(actions: [silAction,duzenleAction])
}
```

```
func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
    let urun = urunlerListesi[indexPath.row]
    performSegue(withIdentifier: "toDetay", sender: urun)
}

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    if segue.identifier == "toDetay" {
        if let urun = sender as? Urunler {
            let gidilecekVC = segue.destination as! DetaySayfa
            gidilecekVC.urun = urun
        }
    }
}

func sepeteEkleTiklandi(indexPath: IndexPath) {
    let urun = urunlerListesi[indexPath.row]
    print("\(urun.ad!) sepete eklendi")
}
```



Bütün mesafeler 32 dir.
Resim aynı boyuttadır.
Yazı boyut : 30
Button yazı boyut : 25

Show segue oluştur.
Segue idsi toDetay

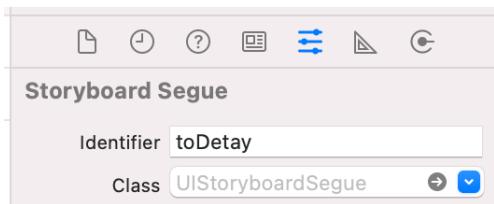
```
class DetaySayfa: UIViewController {
    @IBOutlet weak var labelUrunFiyat: UILabel!
    @IBOutlet weak var imageViewUrun: UIImageView!
    var urun:Urunler?

    override func viewDidLoad() {
        super.viewDidLoad()

        if let u = urun {
            self.navigationItem.title = u.ad
            imageViewUrun.image = UIImage(named: u.resim!)
            labelUrunFiyat.text = "\(u.fiyat!) ₺"
        }
    }

    @IBAction func buttonSepeteEkle(_ sender: Any) {
        if let u = urun {
            print("Detay Sayfa : \(u.ad!) sepete eklendi.")
        }
    }
}
```

Veri Transferi



```
func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
    let urun = urunlerListe[indexPath.row]

    performSegue(withIdentifier: "toDetay", sender: urun)
    //Seçilme animasyonunu kaldırır.
    tableView.deselectRow(at: indexPath, animated: true)
}

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    if segue.identifier == "toDetay" {
        let urun = sender as? Urunler
        let gidilecekVC = segue.destination as! DetayVC
        gidilecekVC.urun = urun
    }
}
```

Seçim animasyonunu tamamen kaldırma

```
cell.hucreProtocol = self
cellIndexPath = indexPath

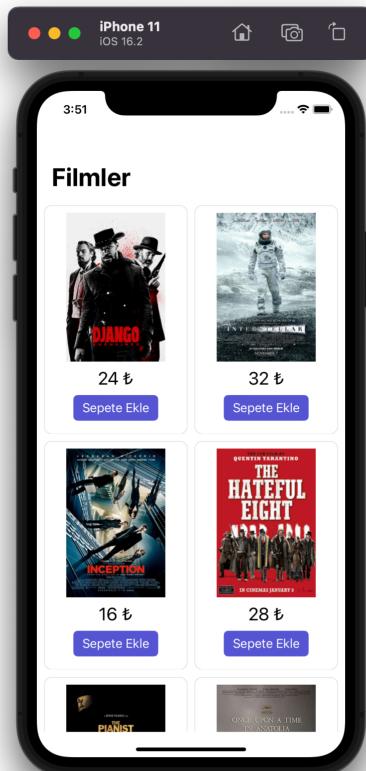
//Seçilme animasyonunu tamamen kaldırır.
cell.selectionStyle = .none

return cell
}
```

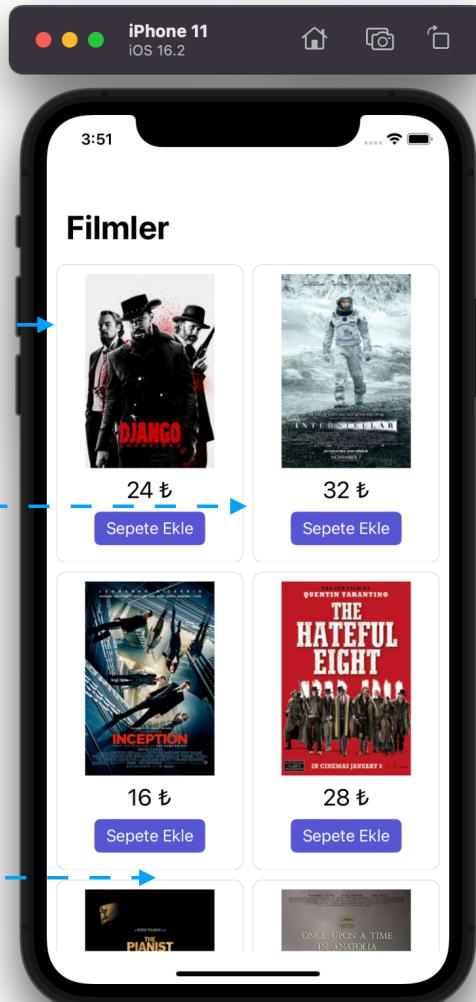
Collection View

Collection View

- Grid tasarımı ile listeleme yapabildiğimiz bir yapıdır.
- TableView ile benzer özelliklere sahiptir fakat bazı farklılıklar vardır.



Collection View



sectionInset

Collection View

Çevresindeki Boşluk

minimumInteritemSpacing

İtemler arası yatay boşluk

minimumLineSpacing

İtemler arası dikey boşluk

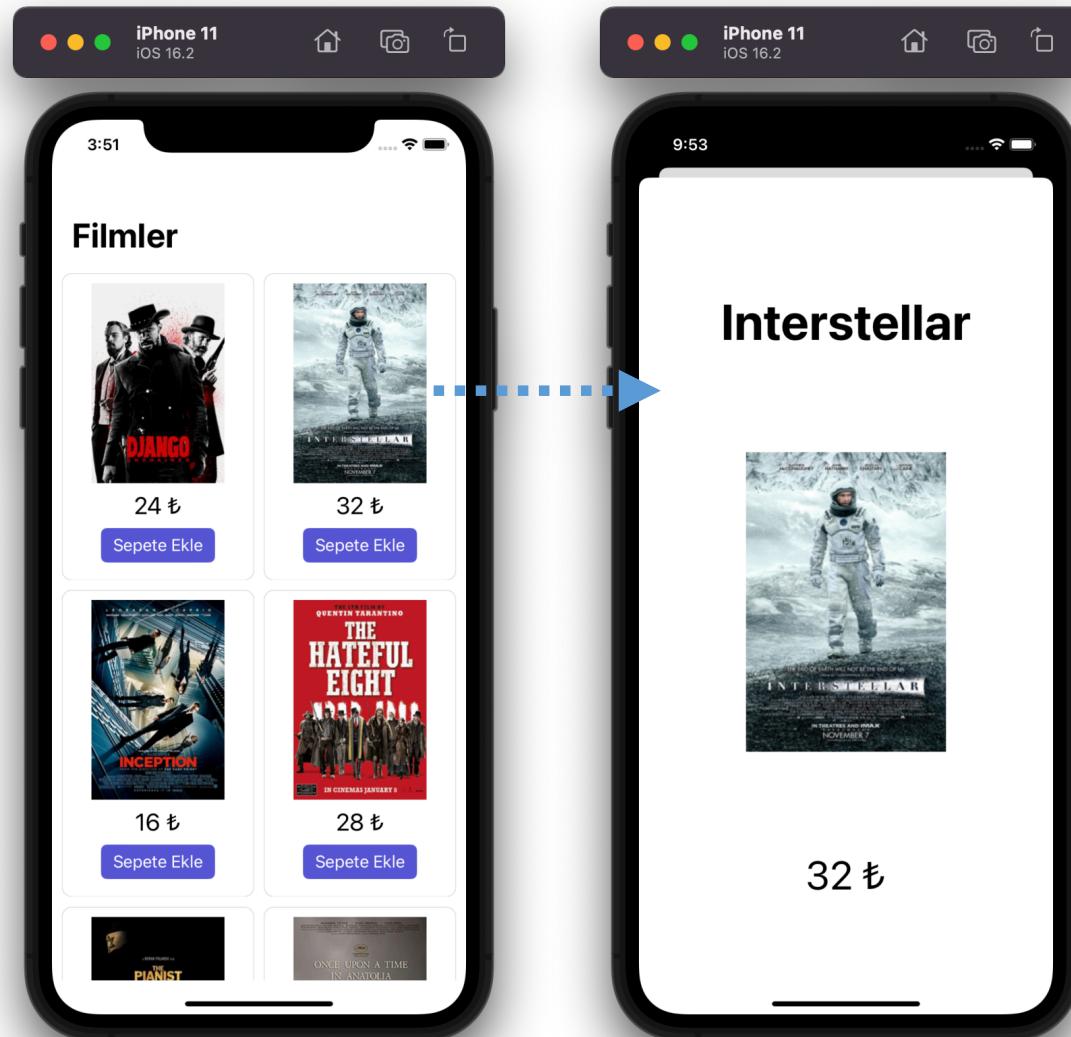
```
let tasarim = UICollectionViewFlowLayout()
```

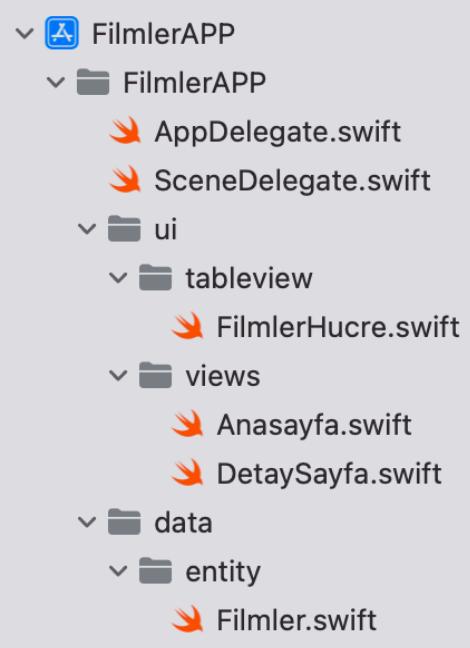
```
tasarim.sectionInset = UIEdgeInsets(top: 10, left: 10, bottom: 10, right: 10)  
tasarim.minimumInteritemSpacing = 10  
tasarim.minimumLineSpacing = 10
```

```
//10x10x10 = 30
```

```
let ekranGenislik = UIScreen.main.bounds.width  
let itemGenislik = (ekranGenislik-30)/2 //2 : Yatay item sayısını temsil etmektedir.  
tasarim.itemSize = CGSize(width: itemGenislik, height: itemGenislik*1.6)
```

Film Uygulaması

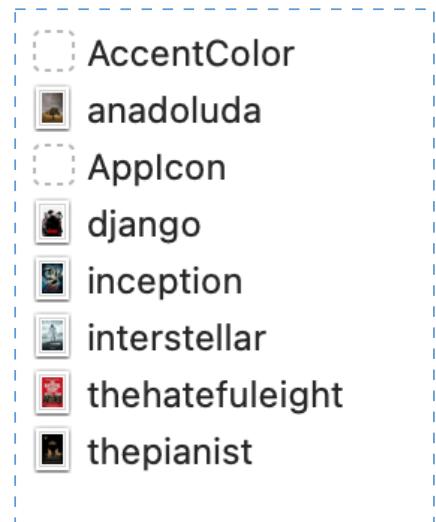


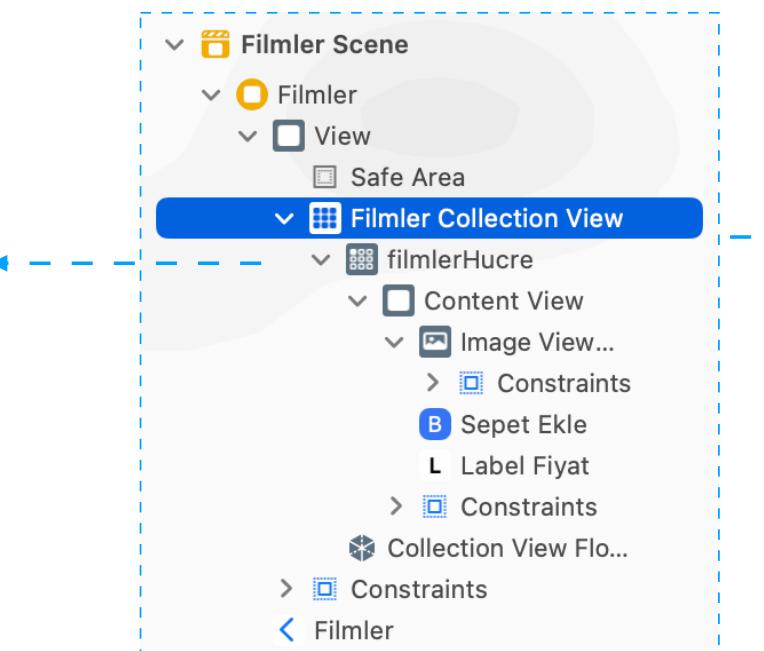
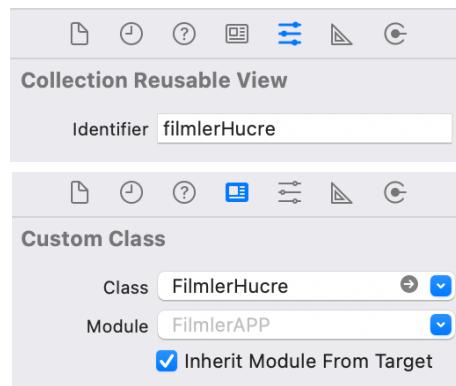


```
class Filmler {
    var id:Int?
    var ad:String?
    var resim:String?
    var fiyat:Int?

    init(){
    }

    init(id: Int, ad: String, resim: String, fiyat: Int) {
        self.id = id
        self.ad = ad
        self.resim = resim
        self.fiyat = fiyat
    }
}
```





```

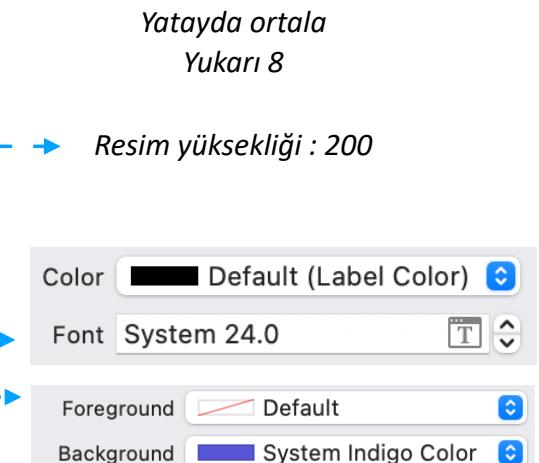
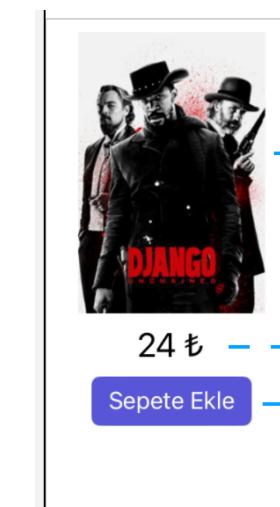
import UIKit
protocol HucreProtocol {
    func sepeteEkleTikla(indexPath: IndexPath)
}

class FilmlerHucre: UICollectionViewCell {
    @IBOutlet weak var imageViewFilm: UIImageView!
    @IBOutlet weak var labelFiyat: UILabel!

    var hucreProtocol: HucreProtocol?
    var indexPath: IndexPath?

    @IBAction func buttonSepeteEkle(_ sender: Any) {
        hucreProtocol?.sepeteEkleTikla(indexPath: indexPath!)
    }
}

```



```

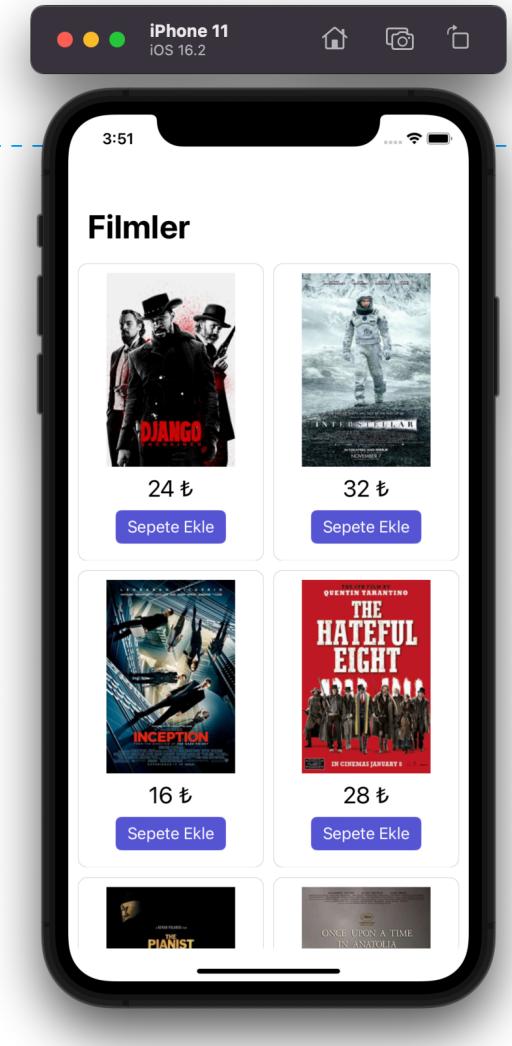
class Anasayfa: UIViewController {
    @IBOutlet weak var filmlerCollectionView: UICollectionView!
    var filmlerListesi = [Filmler]()
    override func viewDidLoad() {
        super.viewDidLoad()
        filmlerCollectionView.delegate = self
        filmlerCollectionView.dataSource = self
        let f1 = Filmler(id: 1, ad: "Django", resim: "django", fiyat: 24)
        let f2 = Filmler(id: 2, ad: "Interstellar", resim: "interstellar", fiyat: 32)
        let f3 = Filmler(id: 3, ad: "Inception", resim: "inception", fiyat: 16)
        let f4 = Filmler(id: 4, ad: "The Hateful Eight", resim: "thehatefuleight", fiyat: 28)
        let f5 = Filmler(id: 5, ad: "The Pianist", resim: "thepianist", fiyat: 18)
        let f6 = Filmler(id: 6, ad: "Anadoluda", resim: "anadoluda", fiyat: 10)
        filmlerListesi.append(f1)
        filmlerListesi.append(f2)
        filmlerListesi.append(f3)
        filmlerListesi.append(f4)
        filmlerListesi.append(f5)
        filmlerListesi.append(f6)
        let tasarim = UICollectionViewFlowLayout()
        tasarim.sectionInset = UIEdgeInsets(top: 10, left: 10, bottom: 10, right: 10)
        tasarim.minimumInteritemSpacing = 10
        tasarim.minimumLineSpacing = 10
        //10x10x10 = 30
        let ekranGenislik = UIScreen.main.bounds.width
        let itemGenislik = (ekranGenislik-30)/2
        tasarim.itemSize = CGSize(width: itemGenislik, height: itemGenislik*1.6)
        filmlerCollectionView.collectionViewLayout = tasarim
    }
}

```

```

let f1 = Filmler(id: 1, ad: "Django", resim: "django", fiyat: 24)
let f2 = Filmler(id: 2, ad: "Interstellar", resim: "interstellar", fiyat: 32)
let f3 = Filmler(id: 3, ad: "Inception", resim: "inception", fiyat: 16)
let f4 = Filmler(id: 4, ad: "The Hateful Eight", resim: "thehatefuleight", fiyat: 28)
let f5 = Filmler(id: 5, ad: "The Pianist", resim: "thepianist", fiyat: 18)
let f6 = Filmler(id: 6, ad: "Anadoluda", resim: "anadoluda", fiyat: 10)
filmlerListesi.append(f1)
filmlerListesi.append(f2)
filmlerListesi.append(f3)
filmlerListesi.append(f4)
filmlerListesi.append(f5)
filmlerListesi.append(f6)

```



Diger örnek : 15x10x10x15 , 3 item

```
extension Anasayfa : UICollectionViewDelegate, UICollectionViewDataSource, HucreProtocol {
    func collectionView(_ collectionView: UICollectionView, numberOfItemsInSection section: Int) -> Int {
        filmlerListesi.count
    }

    func collectionView(_ collectionView: UICollectionView, cellForItemAt indexPath: IndexPath) -> UICollectionViewCell {
        let film = filmlerListesi[indexPath.row]

        let hucre = collectionView.dequeueReusableCell(withIdentifier: "filmlerHucre",
            for: indexPath) as! FilmlerHucre

        hucre.imageViewFilm.image = UIImage(named: film.resim!)
        hucre.labelFiyat.text = "\u20ac(\(film.fiyat!) \u20ac"

        hucre.layer.borderColor = UIColor.lightGray.cgColor
        hucre.layer.borderWidth = 0.3
        hucre.layer.cornerRadius = 10.0

        hucre.hucreProtocol = self
        hucreIndexPath = indexPath

        return hucre
    }

    func sepeteEkleTikla(indexPath: IndexPath) {
        let film = filmlerListesi[indexPath.row]
        print("\u20ac(\(film.ad!) sepete ekle")
    }

    func sepeteEkleTikla(indexPath: IndexPath) {
        let film = filmlerListesi[indexPath.row]
        print("\u20ac(\(film.ad!) sepete ekle")
    }

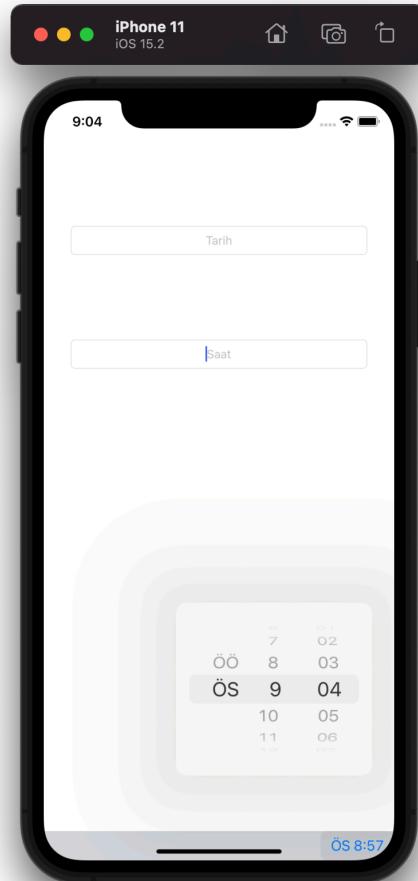
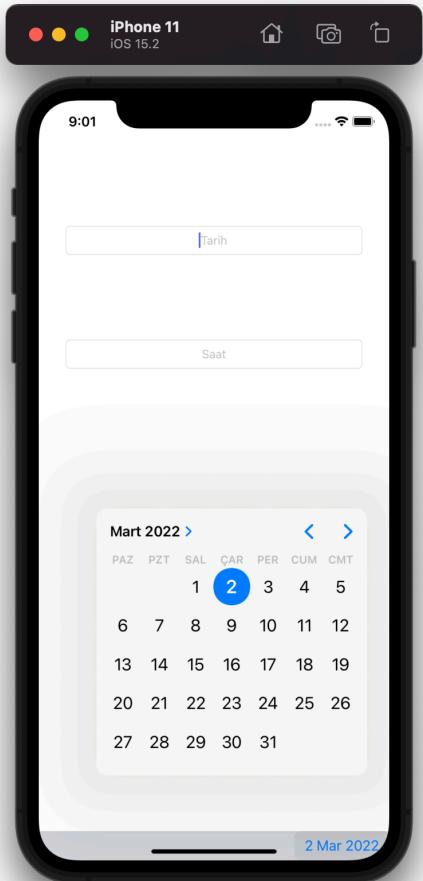
    func collectionView(_ collectionView: UICollectionView, didSelectItemAt indexPath: IndexPath) {
        let film = filmlerListesi[indexPath.row]
        performSegue(withIdentifier: "toDetay", sender: film)
    }

    override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
        if segue.identifier == "toDetay" {
            if let film = sender as? Filmler {
                let gidilecekVC = segue.destination as! DetaySayfa
                gidilecekVC.film = film
            }
        }
    }
}
```



```
class DetaySayfa: UIViewController {  
  
    @IBOutlet weak var labelFiyat: UILabel!  
    @IBOutlet weak var imageViewFilm: UIImageView!  
    @IBOutlet weak var labelFilm: UILabel!  
  
    var film:Filmler?  
  
    override func viewDidLoad() {  
        super.viewDidLoad()  
        if let f = film {  
            labelFilm.text = f.ad  
            imageViewFilm.image = UIImage(named: f.resim!)  
            labelFiyat.text = "\(f.fiyat!) ₺"  
        }  
    }  
}
```

Date & Time Picker



```
class ViewController: UIViewController {

    @IBOutlet weak var tfTarih: UITextField!
    @IBOutlet weak var tfSaat: UITextField!

    var datePicker: UIDatePicker?
    var timePicker: UIDatePicker?

    override func viewDidLoad() {
        super.viewDidLoad()
        datePicker = UIDatePicker()
        datePicker?.datePickerMode = .date
        tfTarih.inputView = datePicker

        timePicker = UIDatePicker()
        timePicker?.datePickerMode = .time
        tfSaat.inputView = timePicker

        let dokunmaAlgılama = UITapGestureRecognizer(target: self, action:
            #selector(dokunmaAlgılamaMetod))
        view.addGestureRecognizer(dokunmaAlgılama)
    }

    @objc func dokunmaAlgılamaMetod(){
        print("Ekrana dokunuldu")
        view.endEditing(true)
    }
}
```

Temel görüntülemeyi sağlar

Temel görüntülemeyi kapatır. Sayfa üzerinde açılan her şey için geçerlidir.

```
class ViewController: UIViewController {

    @IBOutlet weak var tfTarih: UITextField!
    @IBOutlet weak var tfSaat: UITextField!

    var datePicker:UIDatePicker?
    var timePicker:UIDatePicker?

    override func viewDidLoad() {
        super.viewDidLoad()
        datePicker = UIDatePicker()
        datePicker?.datePickerMode = .date
        tfTarih.inputView = datePicker

        timePicker = UIDatePicker()
        timePicker?.datePickerMode = .time
        tfSaat.inputView = timePicker

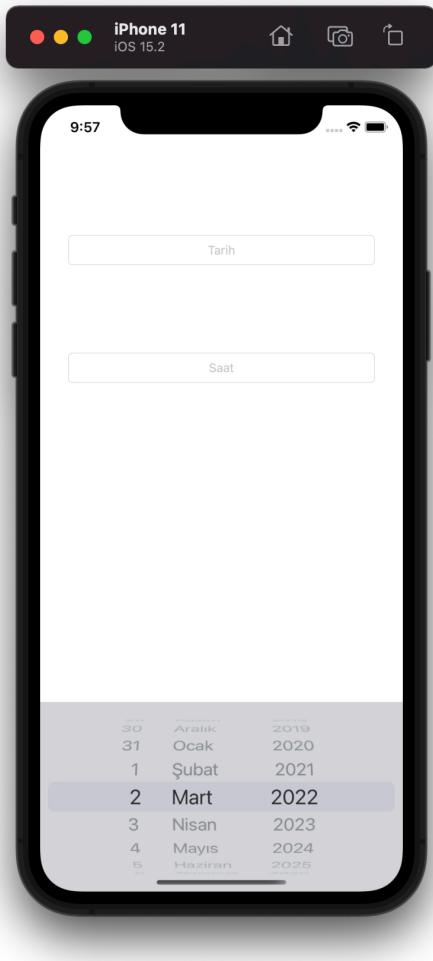
        let dokunmaAlgilama = UITapGestureRecognizer(target: self, action:
            #selector(dokunmaAlgilamaMetod))
        view.addGestureRecognizer(dokunmaAlgilama)

        datePicker?.addTarget(self, action: #selector(self.tarihGoster(uiDatePicker:)), for:
            .valueChanged)

        timePicker?.addTarget(self, action: #selector(self.saatGoster(uiDatePicker:)), for:
            .valueChanged)
    }

    @objc func tarihGoster(uiDatePicker:UIDatePicker){
        let tarihFormati = DateFormatter()
        tarihFormati.dateFormat = "MM/dd/yyyy"
        let alinanTarih = tarihFormati.string(from: uiDatePicker.date)
        tfTarih.text = alinanTarih
    }

    @objc func saatGoster(uiDatePicker:UIDatePicker){
        let saatFormati = DateFormatter()
        saatFormati.dateFormat = "hh:mm"
        let alinanSaat = saatFormati.string(from: uiDatePicker.date)
        tfSaat.text = alinanSaat
    }
}
```



Klasik Görünüm Çevirme

```
override func viewDidLoad() {  
    super.viewDidLoad()  
  
    datePicker = UIDatePicker()  
    datePicker?.datePickerMode = .date
```

```
if #available(iOS 13.4, *) {  
    datePicker?.preferredDatePickerStyle = .wheels  
}
```

```
textfieldTarih.inputView = datePicker  
  
timePicker = UIDatePicker()  
timePicker?.datePickerMode = .time
```

```
if #available(iOS 13.4, *) {  
    timePicker?.preferredDatePickerStyle = .wheels  
}
```

```
textfieldSaat.inputView = timePicker
```

*DatePicker için
Eski Görünüm
Kodlaması*

*TimePicker için
Eski Görünüm
Kodlaması*

Uygulama Mimarisi

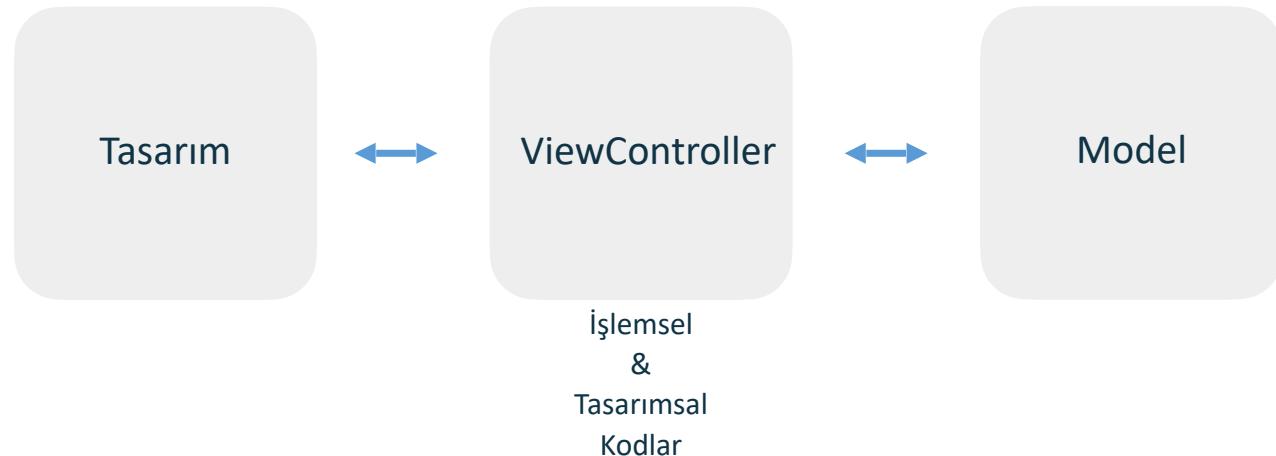
MVVM

MVVM

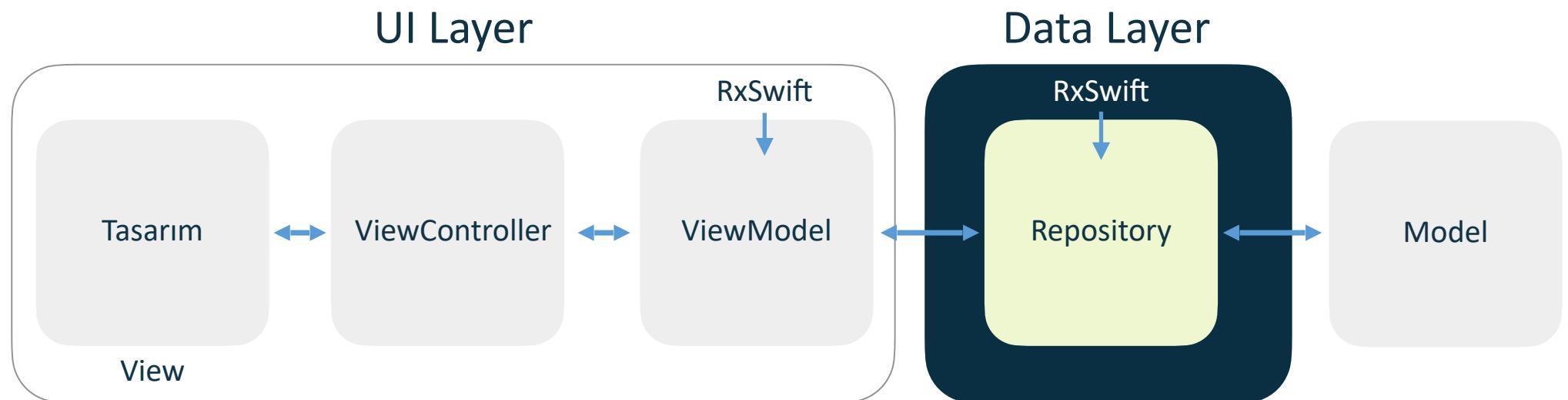
Bu mimari sayesinde daha profesyonel kodlamalar yapabiliriz.
Özellikle büyük projelerde oluşan kodlama karmaşasını azaltır.

Mimarının amacı proje içerisindeki işlemsel kodları ve tasarımsal kodları ayırip daha modüller ve düzenli hale getirmektir.

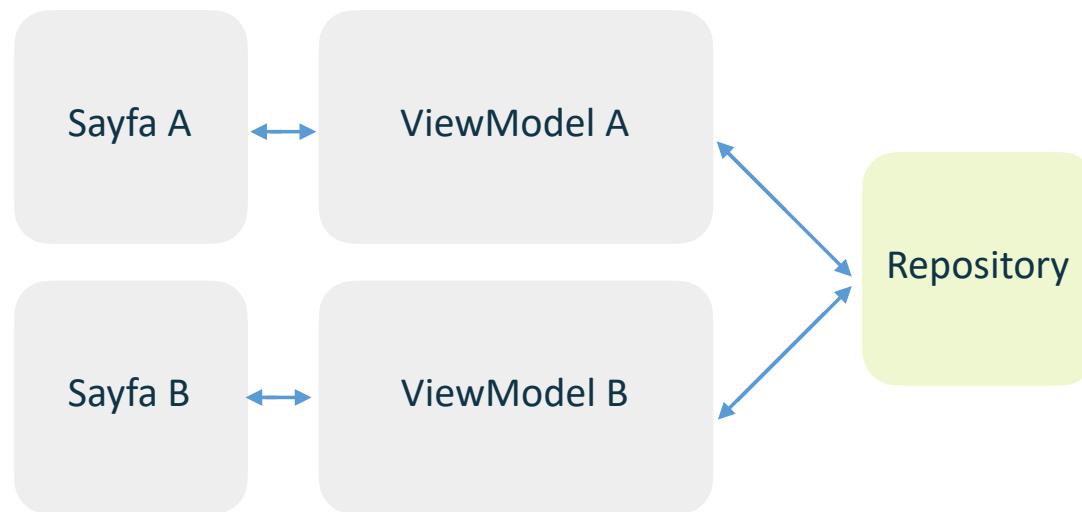
Klasik Yöntem



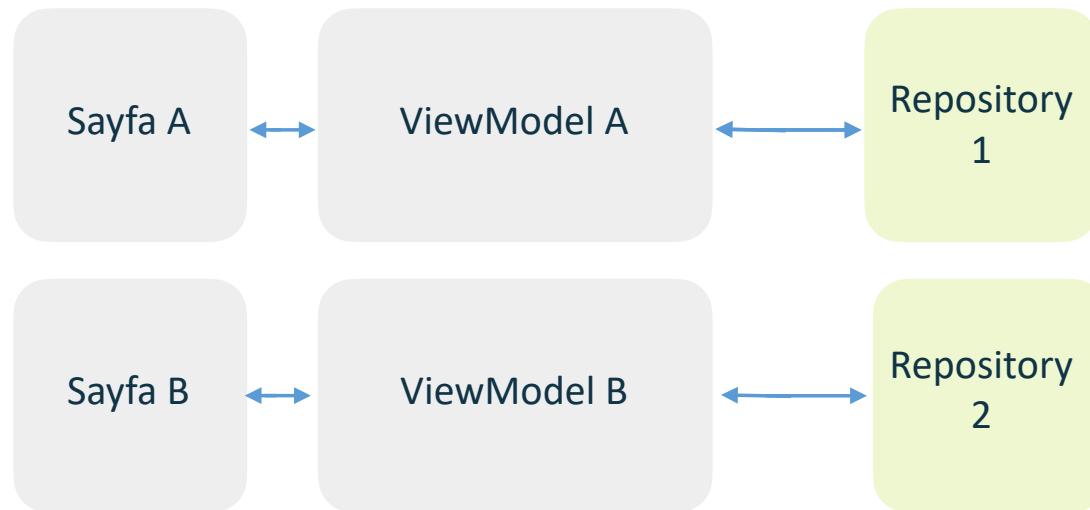
MVVM



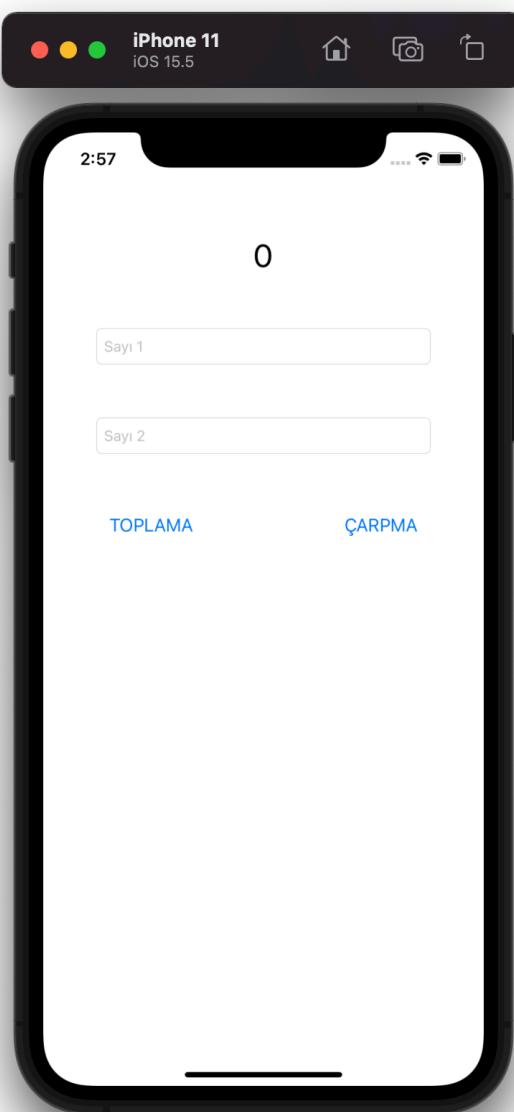
Farklı Senaryolar



Farklı Senaryolar



Klasik Yöntem



```
import UIKit

class Anasayfa: UIViewController {

    @IBOutlet weak var labelSonuc: UILabel!
    @IBOutlet weak var textFieldSayi1: UITextField!
    @IBOutlet weak var textFieldSayi2: UITextField!

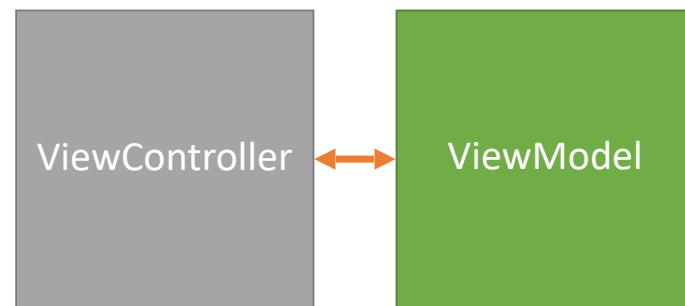
    override func viewDidLoad() {
        super.viewDidLoad()
        labelSonuc.text = "0"
    }

    @IBAction func buttonToplama(_ sender: Any) {
        if let alinanSayi1 = textFieldSayi1.text, let alinanSayi2 = textFieldSayi2.text {
            if let sayi1 = Int(alinanSayi1) , let sayi2 = Int(alinanSayi2) {
                let toplam = sayi1 + sayi2
                labelSonuc.text = String(toplam)
            }
        }
    }

    @IBAction func buttonCarpma(_ sender: Any) {
        if let alinanSayi1 = textFieldSayi1.text, let alinanSayi2 = textFieldSayi2.text {
            if let sayi1 = Int(alinanSayi1) , let sayi2 = Int(alinanSayi2) {
                let carpma = sayi1 * sayi2
                labelSonuc.text = String(carpma)
            }
        }
    }
}
```

ViewModel

- Viewmodel'in asıl amacı arayüzü besleyecek verileri organize etmektir.
- Arayüzü verilerden ayırarak daha kontrollü yapı oluşturabiliriz.
- ViewModel içinde sayfa üzerinde yapılacak işlemleri bulundurabiliriz.
- View Modelde veritabanından veri alma , arayüzde matematiksel işlem vb. gibi şeyler yapılır.
- Verilerdeki değişimleri gözlemlemek için *RxSwift* yapısını kullanabiliriz.



ViewModel

- Arayüzde kullanacağımız veriyi bu sınıfta tanımlarız ve yönetiriz.

```
import Foundation

class AnasayfaViewModel {
    var sonuc = "0" ← Yönetilecek Değer

    func toplamaYap(alinanSayi1:String,alinanSayi2:String){
        if let sayi1 = Int(alinanSayi1) , let sayi2 = Int(alinanSayi2) {
            let toplam = sayi1 + sayi2
            sonuc = String(toplam) ← Yönetilecek Değere Veri Aktarma
        }
    }

    func carpmaYap(alinanSayi1:String,alinanSayi2:String){
        if let sayi1 = Int(alinanSayi1) , let sayi2 = Int(alinanSayi2) {
            let carpma = sayi1 * sayi2
            sonuc = String(carpma) ← Yönetilecek Değere Veri Aktarma
        }
    }
}
```

ViewModel Kullanımı

```
import UIKit

class Anasayfa: UIViewController {

    @IBOutlet weak var labelSonuc: UILabel!
    @IBOutlet weak var textFieldSayi1: UITextField!
    @IBOutlet weak var textFieldSayi2: UITextField!

    var viewModel = AnasayfaViewModel() ← Tanımlama

    override func viewDidLoad() {
        super.viewDidLoad()
        labelSonuc.text = viewModel.sonuc ← Veri Okuma
    }

    @IBAction func buttonToplama(_ sender: Any) {
        if let alinanSayi1 = textFieldSayi1.text, let alinanSayi2 = textFieldSayi2.text {
            viewModel.toplamaYap(alinanSayi1: alinanSayi1, alinanSayi2: alinanSayi2) ← İşlem Yapma
            labelSonuc.text = viewModel.sonuc
        }
    }

    @IBAction func buttonCarpma(_ sender: Any) {
        if let alinanSayi1 = textFieldSayi1.text, let alinanSayi2 = textFieldSayi2.text {
            viewModel.carpmaYap(alinanSayi1: alinanSayi1, alinanSayi2: alinanSayi2) ← İşlem Yapma
            labelSonuc.text = viewModel.sonuc
        }
    }
}
```

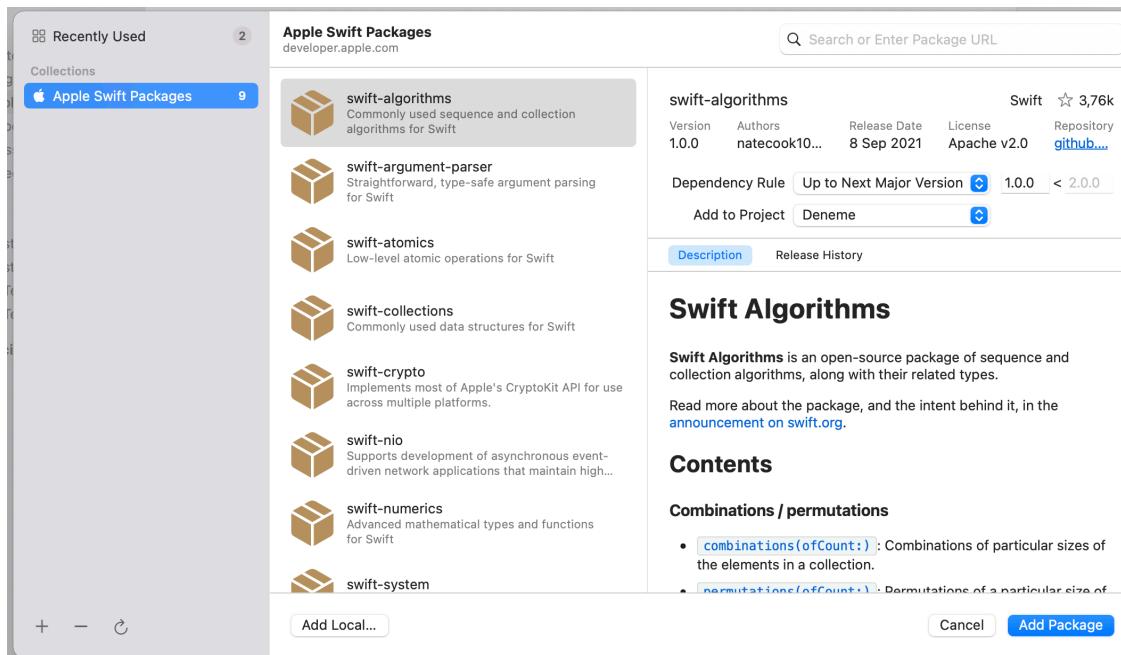
RxSwift

- Reactive kodlama yaklaşımıdır.
- Bir çok dil için bu yapı vardır ve Swift içinde RxSwift'dır.
- Asenkron işlemler yapabiliriz. View Model ile kullanılır.
- RxSwift View Model'in kullanımı kolaylaştırır.
- Temelde yaptığı işlem ViewModel içinde yönetilen verinin tetiklenmesini sağlamak ve değişimi dinlemektir.
- Bu tetikleme ve değişimi dinleme işlemi kodlama yapımızı sadeleştirir.



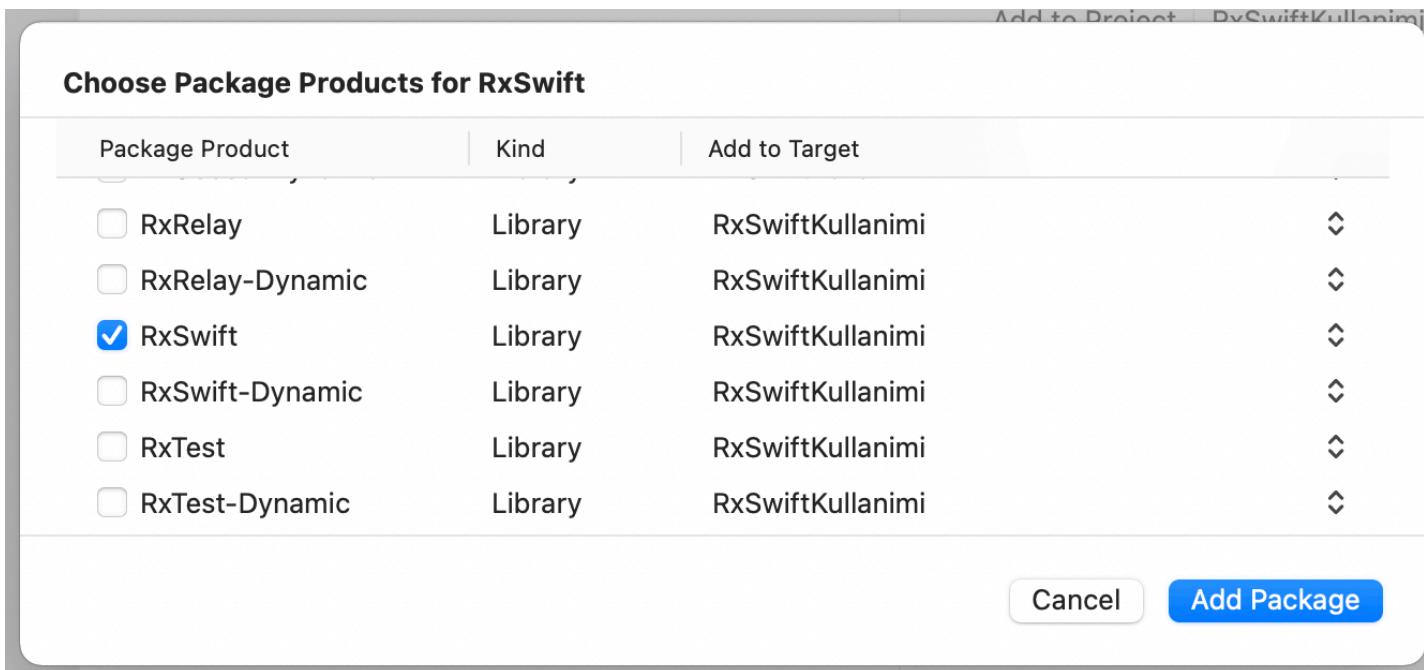
Swift Package Manager Kurulum

- Bu tool sayesinde harici kütüphaneleri kolaylıkla xcode projemize ekleyebiliriz.
- Gerekli olan şey ilgili kütüphanenin Swift Package Manager urlsidir.



Kurulum

<https://github.com/ReactiveX/RxSwift.git>



RxSwift - ViewModel İçinde Kullanımı

- ViewModel ile Sayfa arasındaki veri iletişimini kolaylaştırır.

```
import Foundation
import RxSwift

class AnasayfaViewModel {
    var sonuc = BehaviorSubject<String>(value: "0") ← Yönetilecek Değer

    func toplamaYap(alinanSayi1:String,alinanSayi2:String){
        if let sayi1 = Int(alinanSayi1) , let sayi2 = Int(alinanSayi2) {
            let toplam = sayi1 + sayi2
            sonuc.onNext(String(toplam)) ← Tetikleme
        }
    }

    func carpmaYap(alinanSayi1:String,alinanSayi2:String){
        if let sayi1 = Int(alinanSayi1) , let sayi2 = Int(alinanSayi2) {
            let carpma = sayi1 * sayi2
            sonuc.onNext(String(carpma)) ← Tetikleme
        }
    }
}
```

Varsayılan değer gerekli değilse PublishSubject kullanabilirsin.Kodlar çok benzer.
Tetikleme yapısı mevcut android gibi modelleyip proje oluşturulabilir.

RxSwift - ViewController İçinde Kullanımı

```
import UIKit
import RxSwift

class Anasayfa: UIViewController {
    @IBOutlet weak var labelSonuc: UILabel!
    @IBOutlet weak var textFieldSayi1: UITextField!
    @IBOutlet weak var textFieldSayi2: UITextField!

    var viewModel = AnasayfaViewModel()

    override func viewDidLoad() {
        super.viewDidLoad()

        _ = viewModel.sonuc.subscribe(onNext: { s in
            self.labelSonuc.text = String(s)
        })
    }

    @IBAction func buttonToplama(_ sender: Any) {
        if let alinanSayi1 = textFieldSayi1.text, let alinanSayi2 = textFieldSayi2.text {
            viewModel.toplamaYap(alinanSayi1: alinanSayi1, alinanSayi2: alinanSayi2)
        }
    }

    @IBAction func buttonCarpma(_ sender: Any) {
        if let alinanSayi1 = textFieldSayi1.text, let alinanSayi2 = textFieldSayi2.text {
            viewModel.carpmaYap(alinanSayi1: alinanSayi1, alinanSayi2: alinanSayi2)
        }
    }
}
```

← *Dinleme*

RxSwift - PublishSubject

- Oluşturulurken boş içerikle oluşturulur ve bu yapı sayesinde dinamik dinleme işlemi yapabiliriz. Veri kümesine veri eklendiğinde en son eklenen veri dinleme yapısına aktarılır.

```
import UIKit
import RxSwift

class ViewController: UIViewController {

    let ps = PublishSubject<String>()

    override func viewDidLoad() {
        super.viewDidLoad()

        _ = ps.subscribe(onNext:{string in
            print("PublishSubject Dinleme : \(string)") ————— Dindleme
        })
    }

    @IBAction func tikla(_ sender: Any) {
        ps.onNext("Heyy publishSubject") ————— Tetikleme
    }
}
```

PublishSubject Dinleme : Heyy publishSubject

RxSwift - BehaviorSubject

- PublishSubject aynı özelliklere sahiptir. Oluşturulurken başlangıç değeri verebiliriz.

```
import UIKit
import RxSwift

class ViewController: UIViewController {

    let bs = BehaviorSubject<String>(value: "İlk Eleman")

    override func viewDidLoad() {
        super.viewDidLoad()

        _ = bs.subscribe(onNext:{string in
            print("BehaviorSubject Dinleme : \(string)") ————— Dinleme
        })
    }

    @IBAction func tikla(_ sender: Any) {
        bs.onNext("Heyy behaviorSubject") ————— Tetikleme
    }
}
```

BehaviorSubject Dinleme : İlk Eleman
BehaviorSubject Dinleme : Heyy behaviorSubject

Repository Sınıfı

- Ortak kullanım için oluşturduğumuz metodların yer aldığı sınıfır.
- Veritabanı işlemlerinde bazı metodları bir çok sayfa kullanabilir.
- Ortak erişebilecek bir sınıf oluşturup kodlama tekrarlarını azaltırız.
- Aslında dao (Database access object) sınıfıdır.
- Alt yapısı [View Model](#) örnek alınarak oluşturulur.

Repository Oluşturma

```
import Foundation
import RxSwift

class MatematikRepository {
    var matematikselSonuc = BehaviorSubject<String>(value: "0")

    func topla(alinanSayi1:String,alinanSayi2:String){
        if let sayi1 = Int(alinanSayi1) , let sayi2 = Int(alinanSayi2) {
            let toplam = sayi1 + sayi2
            matematikselSonuc.onNext(String(toplam))
        }
    }

    func carp(alinanSayi1:String,alinanSayi2:String){
        if let sayi1 = Int(alinanSayi1) , let sayi2 = Int(alinanSayi2) {
            let carpma = sayi1 * sayi2
            matematikselSonuc.onNext(String(carpma))
        }
    }
}
```

ViewModel İçinde Kullanımı

```
import Foundation
import RxSwift

class AnasayfaViewModel {
    var sonuc = BehaviorSubject<String>(value: "0")
    var mrepo = MatematikRepository()

    init(){
        sonuc = mrepo.matematikselSonuc ← Repo içinde yer alan RxSwift Nesnesiyle ViewModel içinde yer alan RxSwift nesnesini bağlıyoruz.
    }

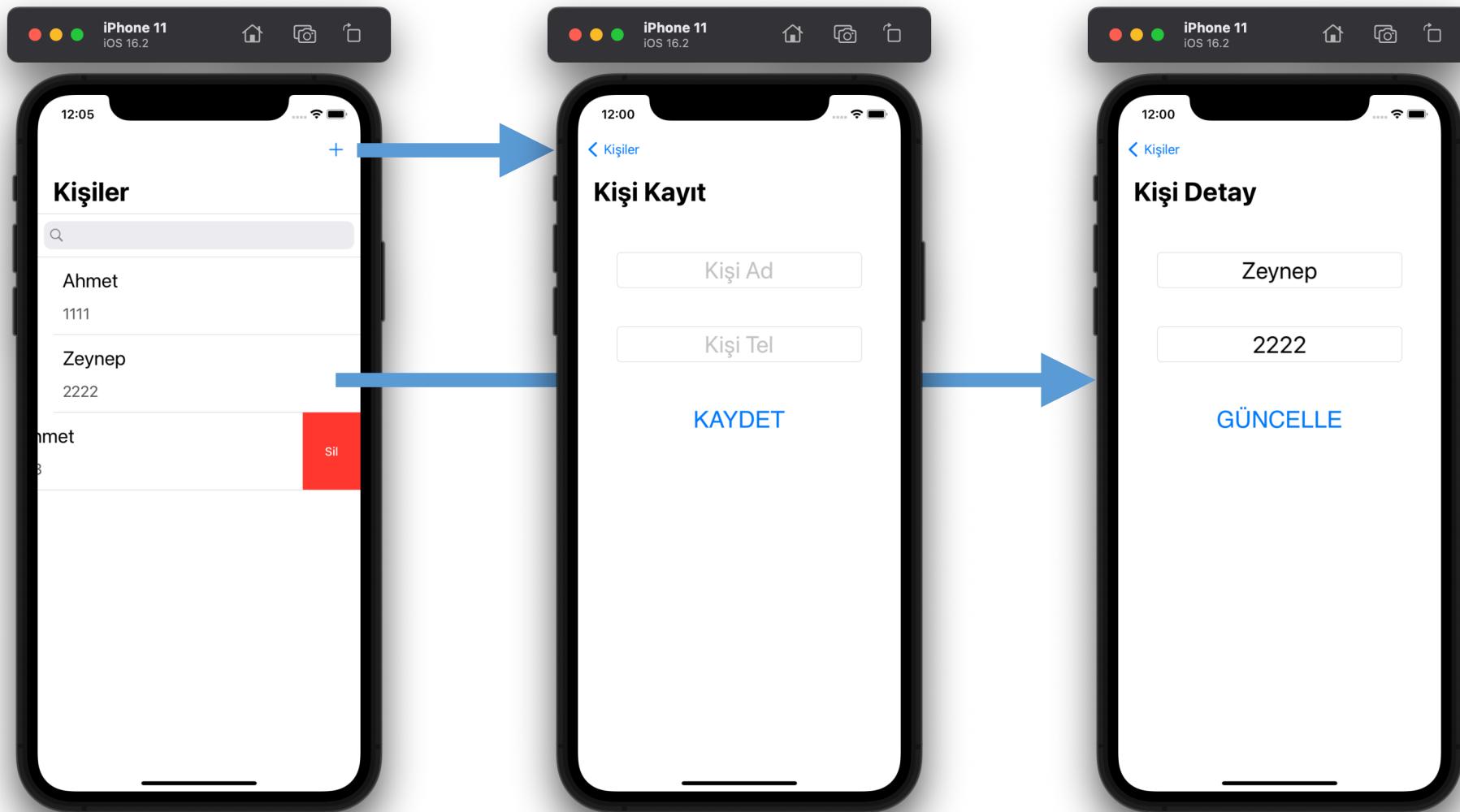
    func toplamaYap(alinanSayi1:String,alinanSayi2:String){
        mrepo.topla(alinanSayi1: alinanSayi1, alinanSayi2: alinanSayi2)
    }

    func carpmaYap(alinanSayi1:String,alinanSayi2:String){
        mrepo.carp(alinanSayi1: alinanSayi1, alinanSayi2: alinanSayi2)
    }
}
```

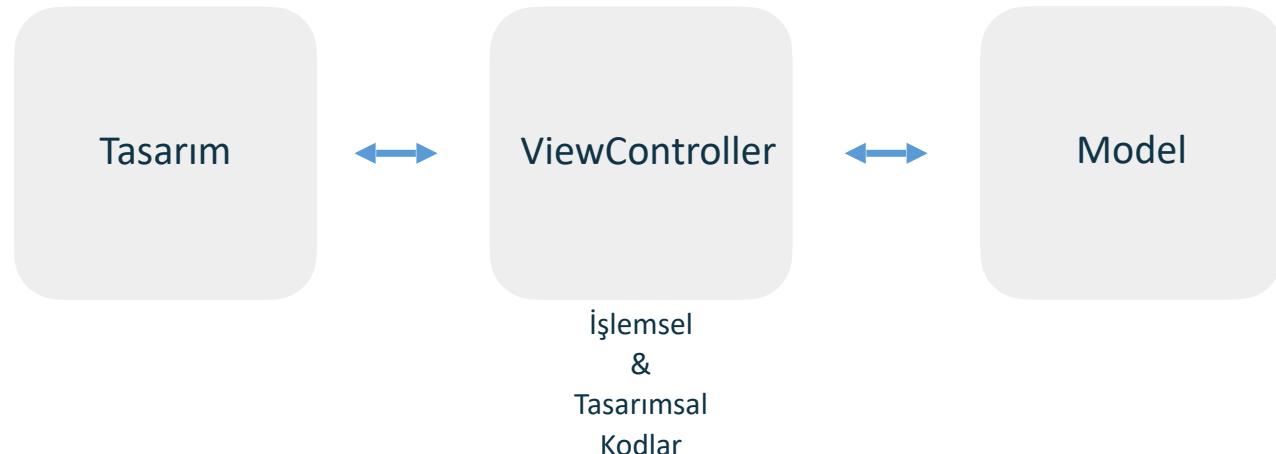
Kişiler Uygulaması

MVVM

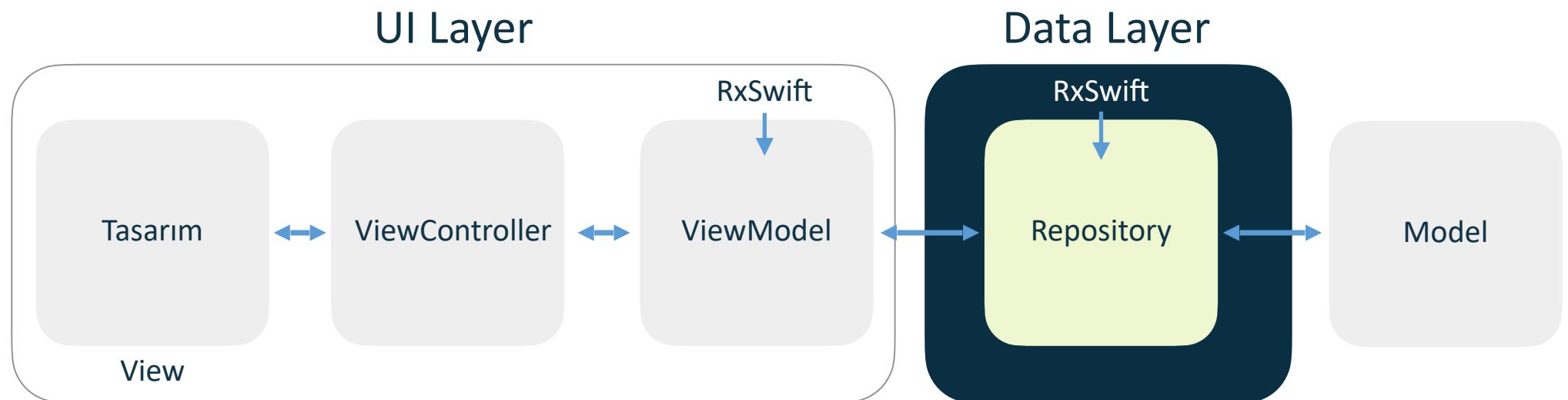
Kişiler Uygulaması



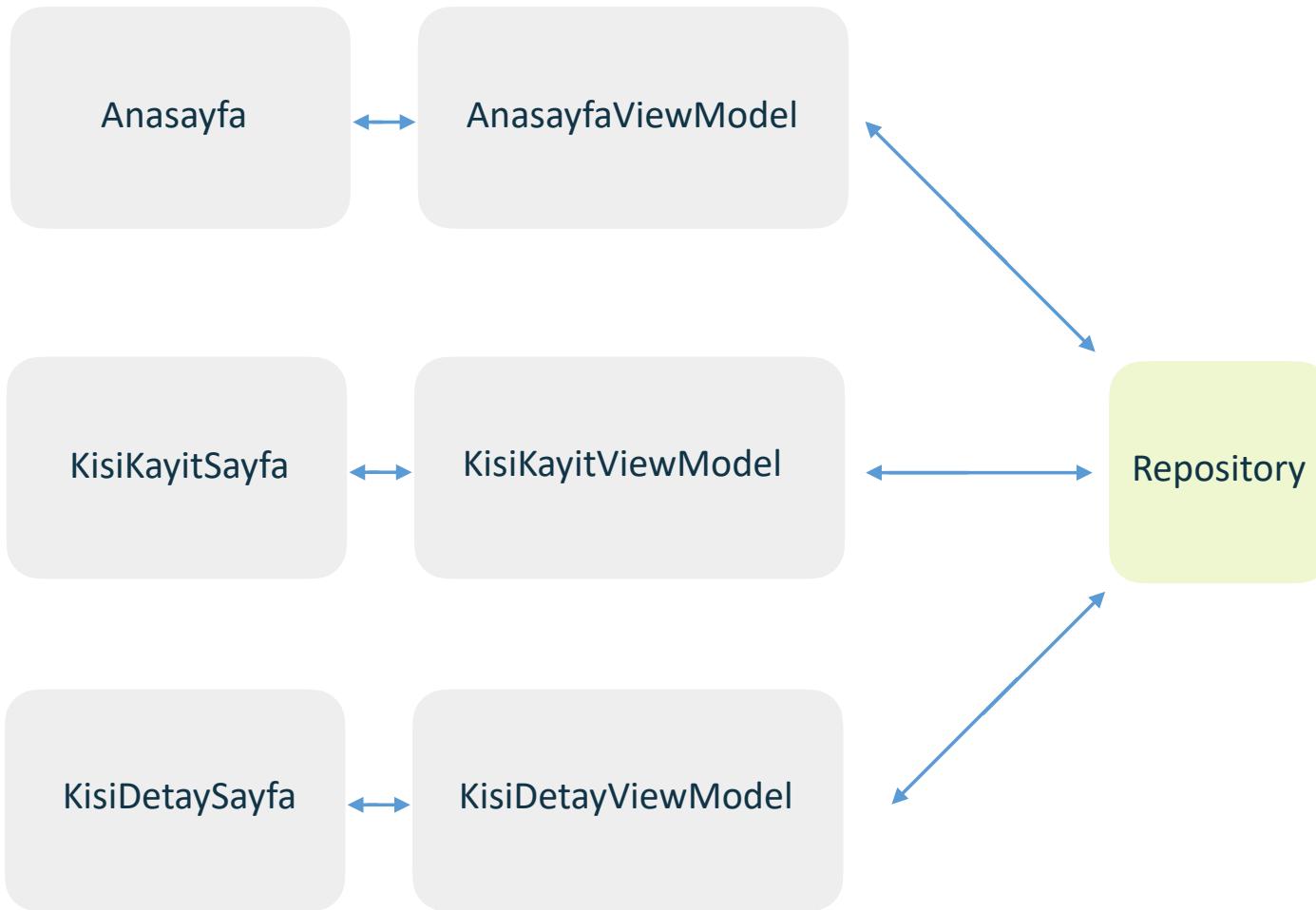
Klasik Yöntem



MVVM



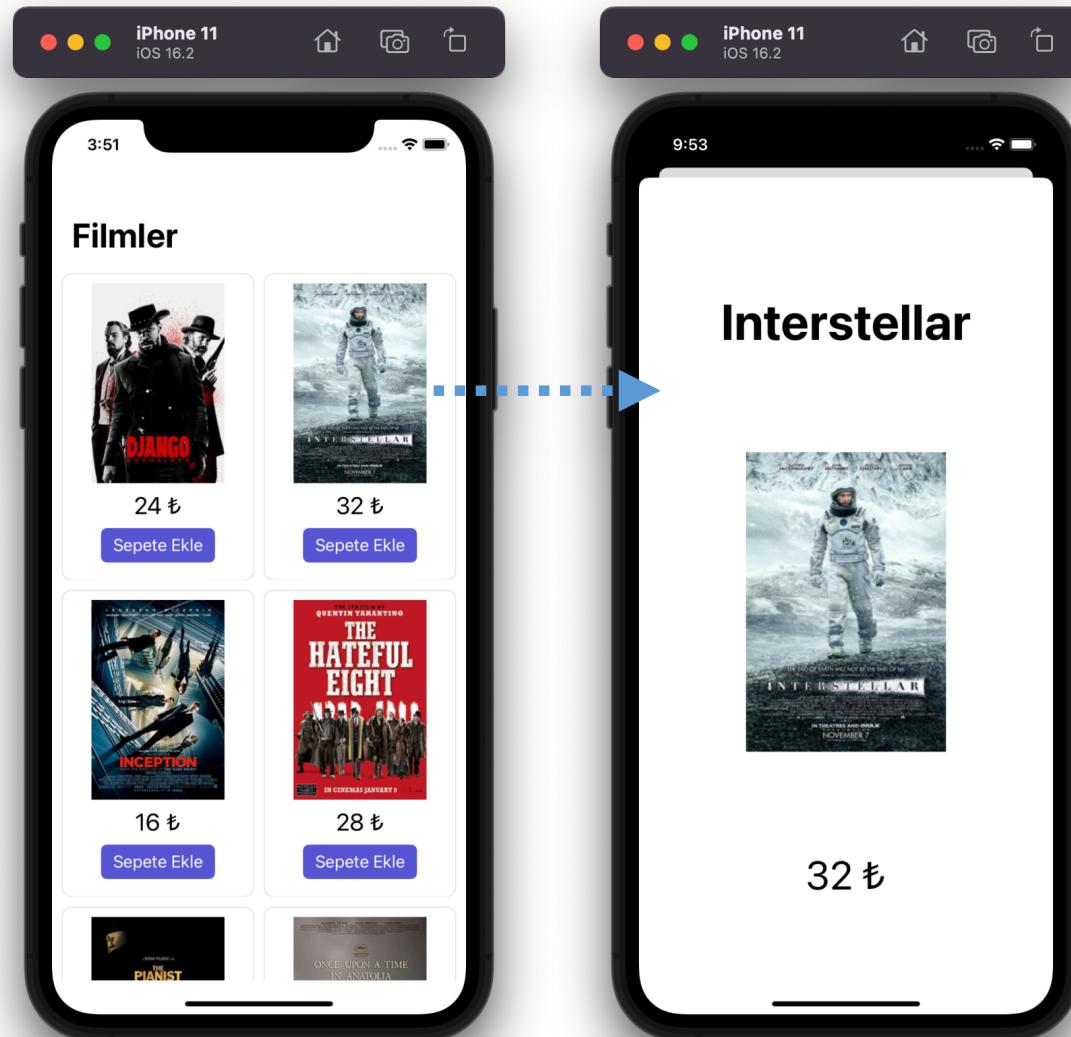
Kişiler Uygulaması



Filmler Uygulaması

MVVM

Film Uygulaması

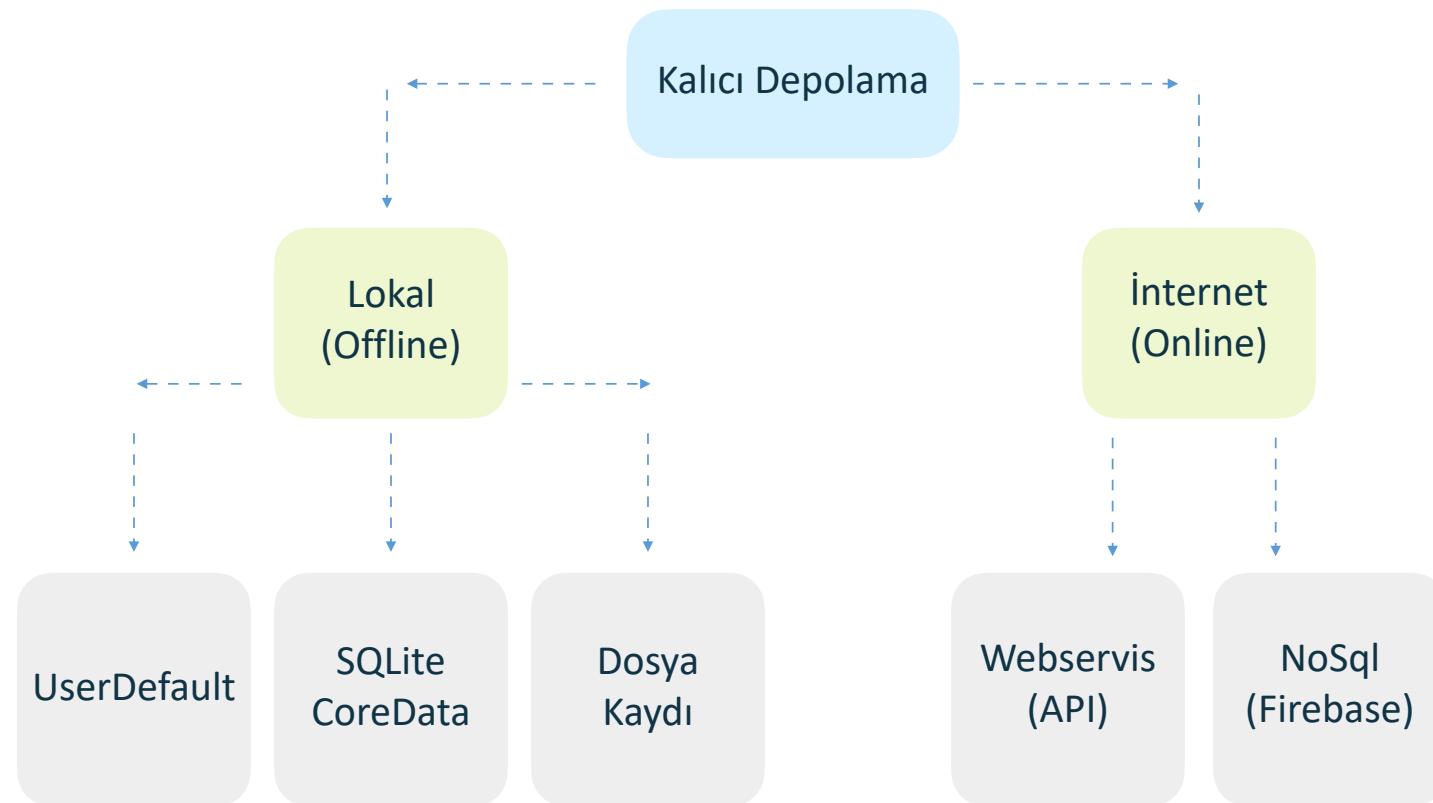


Filmler Uygulaması



Depolama İşlemleri

Depolama İşlemleri



User Default

User Default

- Basit verilerin tutulduğu depolama birimidir.
- Dosya tabanlı bir depolamadır.
- Kayıt yapıldığı anda uygulama silinmediği sürece uygulamada kalır.
- Uygulama açılış süresini olumsuz etkileyebileceğinden dolayı çok fazla bilgi bu depolanmamalıdır.
- Veriler key - value şeklinde depolanır.
- Uygulama ayarları veya oturum bilgilerini kullanmak için idealdir.

User Default

- Kullanılabilecek veri türleri sınırlıdır.
 - Integer
 - Double
 - Float
 - Bool
 - Array
 - Dictionary (Sadece [String:String] türünde)

Veri kaydı

```
let ud = UserDefaults.standard

//Kayıt
ud.set("Ahmet", forKey: "ad")
ud.set(23, forKey: "yas")
ud.set(1.78, forKey: "boy")
ud.set(true, forKey: "bekar")

let liste = ["ali", "ece"];
ud.set(liste, forKey: "liste")

let sehirler = ["16":"BURSA", "34":"İSTANBUL"]
ud.set(sehirler, forKey: "sehirler")
```

Veri Okuma

```
let gelenAd = ud.string(forKey: "ad") ?? "isim yok"
let gelenYas = ud.integer(forKey: "yas")      int,double ve bool kendi
let gelenBoy = ud.double(forKey: "boy")        varsayılan değerlerini kullanır.
let gelenBekar = ud.bool(forKey: "bekar")      varsayılan değerlerini kullanır.

print("Gelen Ad : \(gelenAd)")
print("Gelen Yaş : \(gelenYas)")
print("Gelen Boy : \(gelenBoy)")
print("Gelen Bekar : \(gelenBekar)")

let gelenListe = ud.array(forKey: "liste") ?? [String]()

for a in gelenListe {
    print("Gelen Arkadaş : \(a)")
}

let gelenSehirler = ud.dictionary(forKey: "sehirler") ?? [String:String]()

for (anahtar,deger) in gelenSehirler {
    print("Gelen Şehir : \(anahtar) - \(deger)")
}
```

Veri Silme

```
//Silme  
ud.removeObject(forKey: "ad")
```



Uygulama : Sayaç

```
class ViewController: UIViewController {

    @IBOutlet weak var labelSayac: UILabel!

    override func viewDidLoad() {
        super.viewDidLoad()

        let ud = UserDefaults.standard

        var sayac = ud.integer(forKey: "sayac")

        sayac = sayac + 1

        ud.set(sayac, forKey: "sayac")

        labelSayac.text = "Açılış Sayısı : \(sayac)"
    }
}
```

Veritabanı Giriş

Tablo Yapısı

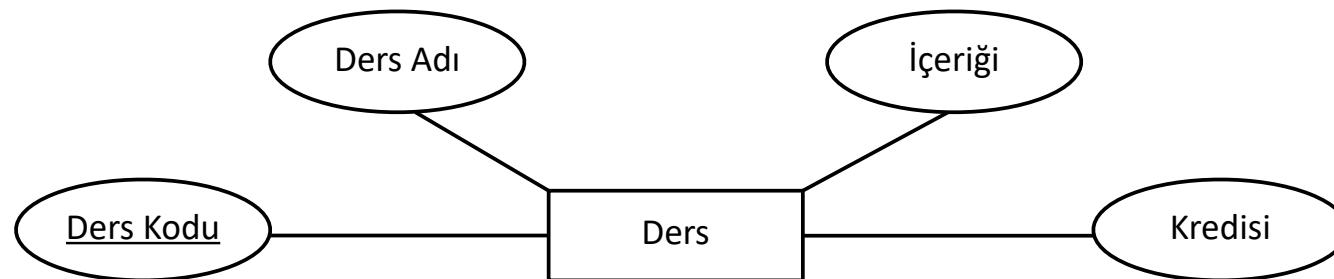
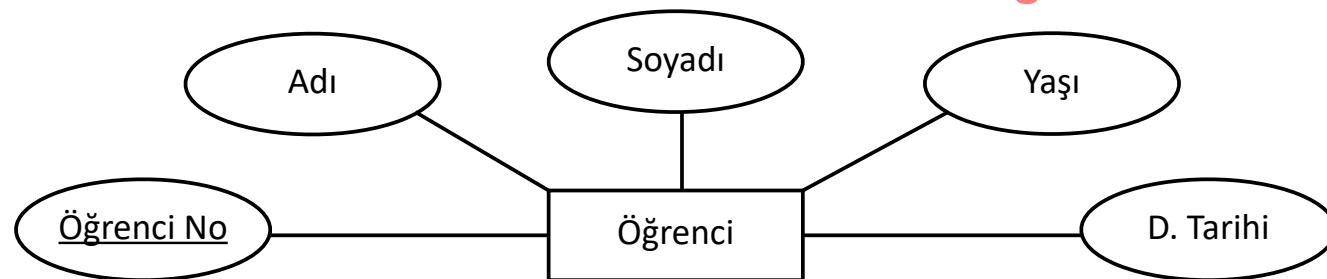
- Tüm Veritabanları içerisinde bilgiler tablolarda Alan

Veri

not_id	ogrenci_adi	ders_adi	not1	not2	
1	Mehmet ERSOY	Tarih	50	60	 Kayıt
2	Mehmet ERSOY	Fizik	70	80	
3	Zeynep GÜR	Tarih	70	90	
4	Cemal GELİR	Tarih	30	50	

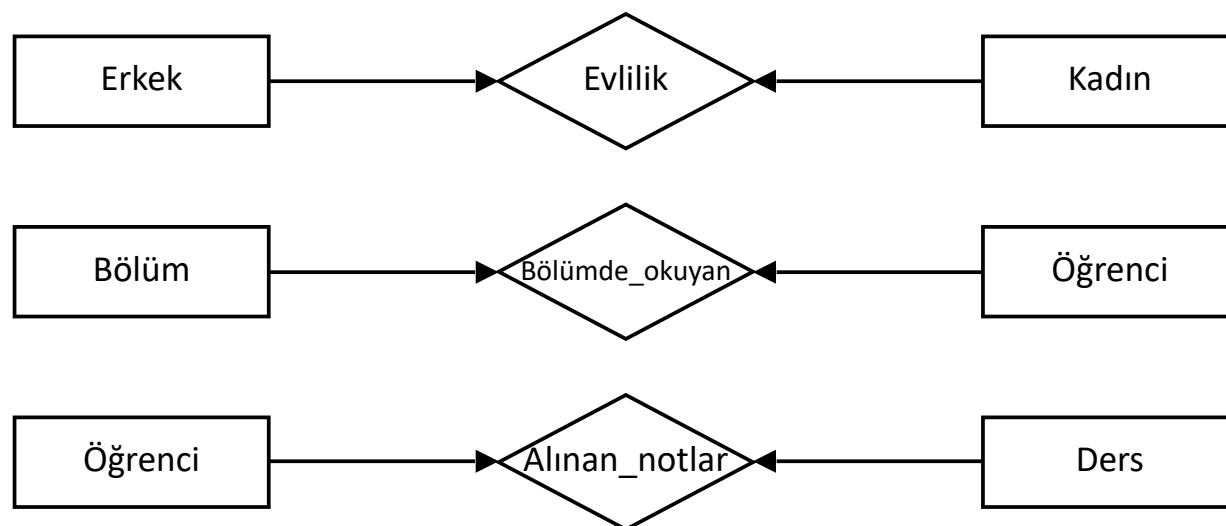
Tablo Tasarlama

- **Öncelikle her tablo için uygun alanların belirlenmelidir aynı sınıflardaki özelliklerin belirlenmesi gibi.**

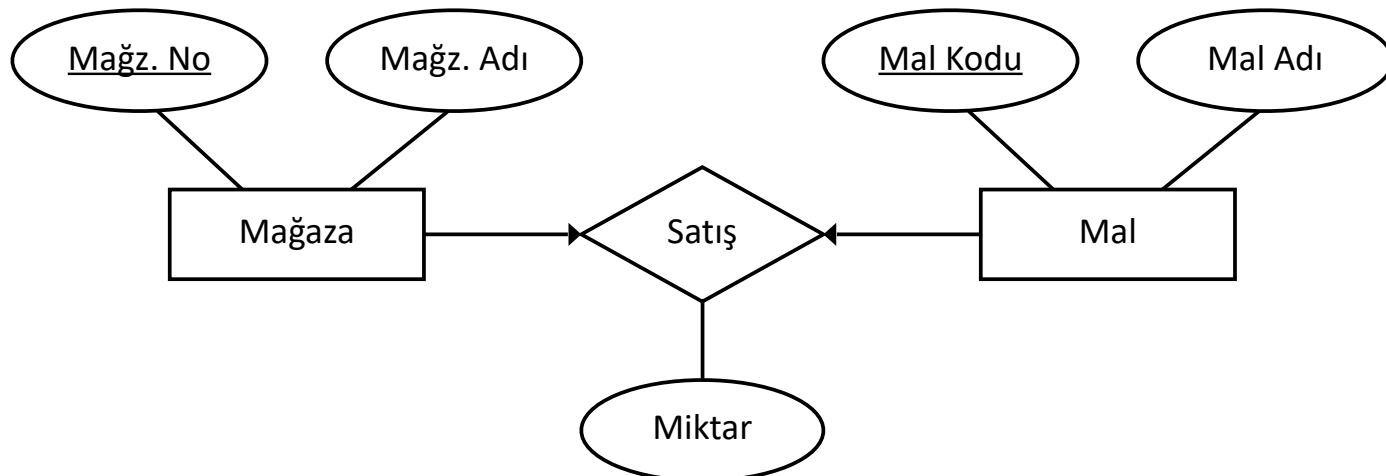


İlişkisel Tablo Modeli

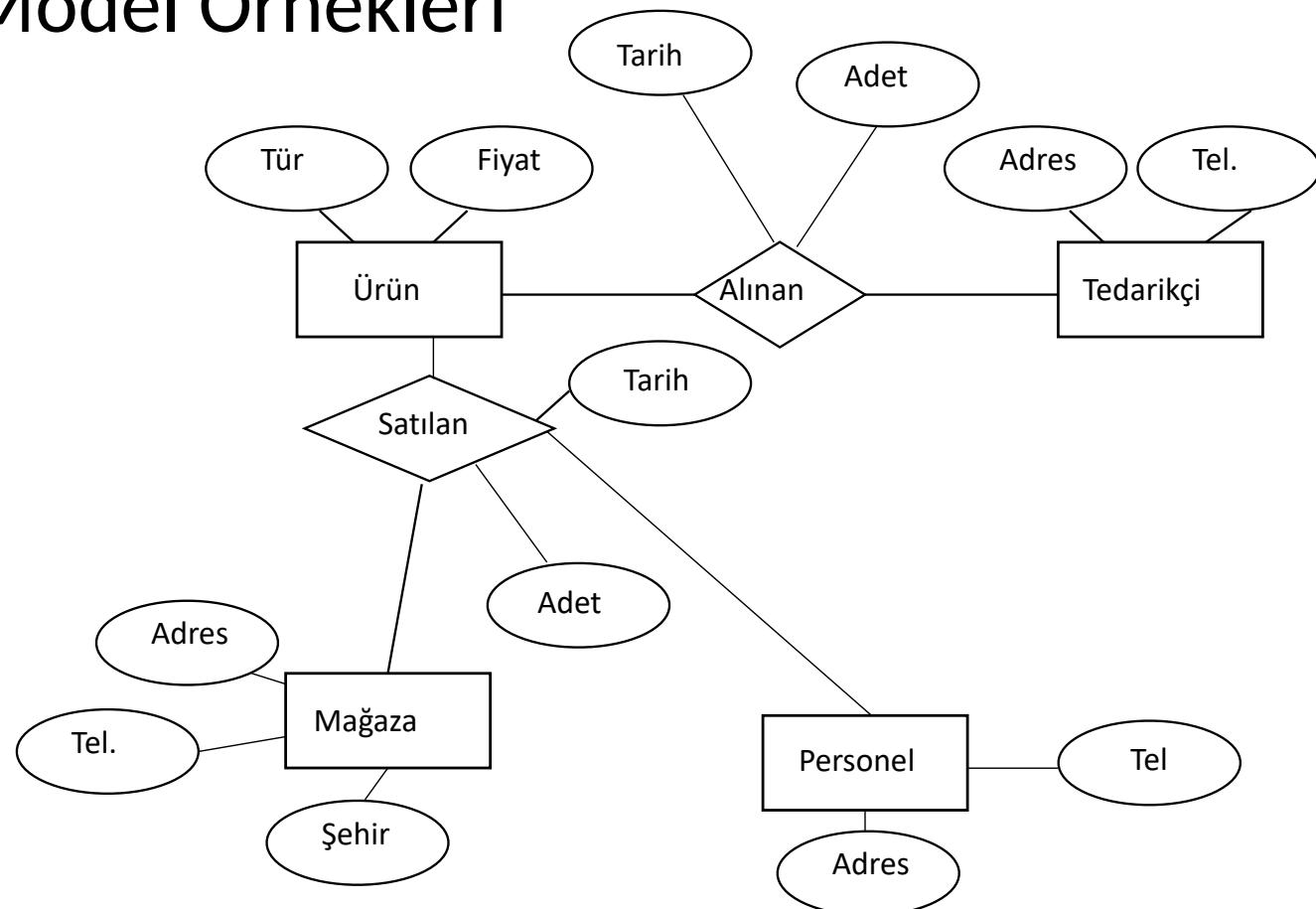
- Tablolardaki gereksiz veri tekrarlarını önlemek için kullanılan bir yapıdır.



İlişkisel Model Örnekleri



İlişkisel Model Örnekleri



Birincil Anahtar – PRIMARY KEY

- İlişkisel veri tabanında bir tablonun benzer değerler içermeyen (tekil : unique) bir sütunu ya da birkaç sütunu birlikte **birincil anahtar** (primary key - PK) olarak tanımlanabilir.
- Birincil anahtar bir aday anahtardır ve söz konusu varlığın kayıtlarını en iyi biçimde karakterize eden anahtardır.
- Birincil anahtar tanımlandığında şu şekilde bir sınırlama konulmuş olacaktır:
 - Birincil anahtar NULL (boş) değerleri veya birbirinin aynı değerleri içeremez.

Dış Anahtar – FOREIGN KEY

- Bir dış anahtar (foreign key - FK) bir sütun veya çok sayıdaki sütunların birleşiminden oluşur.
- Dış anahtar aynı tablo ya da başka bir tablodaki bir birincil anahtar ile eşleştirilir.

PK			FK		PK	
No	Adı	BölümNo			BölümNo	BölümAdı
25	Burak	10			10	Personel
13	Begüm	10			20	Muhasebe
28	Dilay	20			30	Satış

Filmler Uygulaması

- Bu senaryo için temel bir veri tabanı modeli düşünelim.

film_id	film_adi	film_yil	film_resim	kategori_ad	yonetmen_ad
3	Inception	2010	inception	Bilim kurgu	C. Nolan
5	Django	2008	django	Dram	Q. Tarantino
9	The Hateful Eight	2010	thehatefuleight	Dram	Q. Tarantino

Filmler Uygulaması

- Bu senaryo için temel bir veri tabanı modeli düşünelim.

film_id	film_ad	film_yil	film_resim	kategori_id	yonetmen_id
3	Inception	2010	inception	11	88
5	Django	2008	django	22	90
9	The Hateful Eight	2010	thehatefuleight	22	90

kategori_id	kategori_adi
11	Bilim kurgu
22	Dram
44	Komedи

yonetmen_id	yonetmen_adi
88	C. Nolan
90	Q. Tarantino
130	Yılmaz Erdoğan

Okul Uygulaması

- Bu senaryo için temel bir veri tabanı modeli düşünelim.

not_id	ogrenci_adi	ders_adi	not1	not2
1	Mehmet ERSOY	Tarih	50	60
2	Mehmet ERSOY	Fizik	70	80
3	Zeynep GÜR	Tarih	70	90
4	Cemal GELİR	Tarih	30	50

Okul Uygulaması

- Bu senaryo için olması gereken veri tabanı modeli.

not_id	ogrenci_id	ders_id	not1	not2
1	1	1	50	60
2	1	2	70	80
3	2	1	70	90
4	4	1	30	50

ogrenci_id	ogrenci_adi
1	Mehmet ERSOY
2	Zeynep GÜR
4	Cemal GELİR

ders_id	ders_adi
1	Tarih
2	Fizik
3	Kimya

Sipariş Girme Uygulaması

- Kategorilere ayrılmış ürünlerden adet girerek sipariş oluşturma.

siparis_id	siparis_adet	urun_adi	kategori_adi
1	2	Pizza	Yiyecekler
2	1	Baklava	Tatlılar
3	1	Sütlaç	Tatlılar
4	3	Ayran	İçecekler

Sipariş Girme Uygulaması

- Bu senaryo için olması gereken veri tabanı modeli.

siparis_id	siparis_adet	urun_adi
1	2	1
2	1	2
3	1	3
4	3	5

kategori_id	kategori_adi
1	Yiyecekler
2	Tatlılar
3	İçecekler

urun_id	urun_adi	kategori_id
1	Pizza	1
2	Baklava	2
3	Sütlaç	2
5	Ayran	3

Hastane Uygulaması

- Bu senaryo için temel bir veri tabanı modeli düşünelim.

randevu_id	bolum_adi	doktor_adi	hasta_adi	randevu_tarihi
1	Dermotoloji	Sedat AK	Mehmet ERSOY	12/02/2018 10:30
2	Kardiyoloji	Ceyda MERMER	Kemal ALTAY	10/04/2018 12:30
3	Dermotoloji	Sedat AK	Zeynep GÜR	07/01/2018 16:00
4	Dermotoloji	Sedat AK	Cemal GELİR	26/02/2018 9:30

Hastane Uygulaması

- Bu senaryo için olması gereken veri tabanı modeli.

randevu_id	doktor_id	hasta_adi	randevu_tarihi
1	1	Mehmet ERSOY	12/02/2018 10:30
2	2	Kemal ALTAY	10/04/2018 12:30
3	1	Zeynep GÜR	07/01/2018 16:00
4	1	Cemal GELİR	26/02/2018 9:30

bolum_id	bolum_adi
1	Dermotoloji
2	Kardiyoloji
3	Fizik Tedavi
4	Göz Hastalıkları

doktor_id	doktor_adi	bolum_id
1	Sedat AK	1
2	Ceyda MERMER	2
3	Ahmet Ziya	1
4	Ece Kasırga	3

VERİTABANI UYGULAMASI

urunler

urun_id	urun_adi	urun_fiyati
1	Bilgisayar	10000
2	Telefon	7000
3	Saat	3000

musteriler

musteri_id	musteri_adi	musteri_adresi
10	Ahmet	İSTANBUL
12	Zeynep	BURSA
19	Ece	ANKARA

siparisler

siparis_id	urun_id	musteri_id	siparis_adeti
3	1	10	1
5	1	19	2
9	3	12	1

urunler

▼ Gelişmiş

Alanlar Kısıtlar

Ekle Sil En yu...taşı Yukarı taşı Aşağı

İsim	Tip	NN	Biri	Otc	Ber
urun_id	INTEGER	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
urun_adi	TEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
urun_fiyati	INTEGER	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

musteriler

▼ Gelişmiş

Alanlar Kısıtlar

Ekle Sil En yu...taşı Yukarı taşı Aşağı

İsim	Tip	NN	Biri	Otc	Ber
musteri_id	INTEGER	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
musteri_adi	TEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
musteri_adresi	TEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

siparisler

▼ Gelişmiş

Alanlar Kısıtlar

Ekle Sil En yukarı taşı Yukarı taşı Aşağı taşı En aşağı taşı

İsim	Tip	NN	Biri	Otc	Ber	Və	Kç	Karşıla	Yabancı Anahtar
siparis_id	INTEGER	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				
urun_id	INTEGER	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				urunler("urun_id")
musteri_id	INTEGER	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				musteriler musteri_id
siparis_adeti	INTEGER	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				

SQL İFADELERİ

TABLO OLUŞTURMA

```
CREATE TABLE "urunler" (
    "urun_id" INTEGER,
    "urun_adi" TEXT,
    "urun_fiyati" INTEGER,
    PRIMARY KEY("urun_id" AUTOINCREMENT)
);
```

```
CREATE TABLE "siparisler" (
    "siparis_id" INTEGER,
    "urun_id" INTEGER,
    "musteri_id" INTEGER,
    "siparis_adeti" INTEGER,
    PRIMARY KEY("siparis_id" AUTOINCREMENT),
    FOREIGN KEY("musteri_id") REFERENCES "musteriler"("musteri_id"),
    FOREIGN KEY("urun_id") REFERENCES "urunler"("urun_id")
);
```

Veri Kaydı - Insert

INSERT INTO urunler (urun_adi,urun_fiyati) VALUES ('Bilgisayar',10000)

urun_id	urun_adi	urun_fiyati
Filtre	Filtre	Filtre
1	1 Bilgisayar	10000
2	2 Telefon	8000
3	3 Saat	3000
4	4 Mont	750
5	5 Kalem	200
6	6 TV	8000
7	7 Gözlük	1000

Update - Güncelleme

```
UPDATE urunler SET urun_fiyati = 1200 WHERE urun_id = 7 ;
```

urun_id	urun_adi	urun_fiyati
Filtre	Filtre	Filtre
1	1 Bilgisayar	10000
2	2 Telefon	8000
3	3 Saat	3000
4	4 Mont	750
5	5 Kalem	200
6	6 TV	8000
7	7 Gözlük	1200

Delete – Veri Silme

```
DELETE FROM urunler WHERE urun_id = 5 ;
```

	urun_id	urun_adi	urun_fiyati
	Filtre	Filtre	Filtre
1	1	Bilgisayar	10000
2	2	Telefon	8000
3	3	Saat	3000
4	4	Mont	750
5	6	TV	8000
6	7	Gözlük	1200

SELECT – Seçim Yapma

- Tüm verileri getir.

```
SELECT * FROM urunler
```

- İstenilen alanları getir.

```
SELECT urun_adi,urun_fiyati FROM urunler
```

- Kayıt kontrol.

```
SELECT count(*) as toplam FROM urunler WHERE urun_fiyati = 8000
```

- Foreign key li tablolardan verileri alma.

```
SELECT * FROM urunler,musteriler,siparisler WHERE  
urunler.urun_id = siparisler.urun_id and musteriler.musteri_id = siparisler.musteri_id
```

	urun_id	urun_adi	urun_fiyati
	Filtre	Filtre	Filtre
1	1	Bilgisayar	10000
2	2	Telefon	8000
3	3	Saat	3000
4	4	Mont	750
5	6	TV	8000
6	7	Gözlük	1200

WHERE – Şart Oluşturma

```
SELECT * FROM urunler WHERE urun_adi = 'Telefon'
```

```
SELECT * FROM urunler WHERE urun_fiyati > 5000
```

```
SELECT * FROM urunler WHERE urun_fiyati > 1000 and urun_fiyati < 5000
```

Not : Şart yazarken yazı mı ? Sayısal veri mi?
Buna dikkat edilmelidir.
Sayısal veriler normal yazılrken sayısal ifadeler ” veya ‘ ile yazılır.

urun_id	urun_adi	urun_fiyati
Filtre	Filtre	Filtre
1	1 Bilgisayar	10000
2	2 Telefon	8000
3	3 Saat	3000
4	4 Mont	750
5	6 TV	8000
6	7 Gözlük	1200

ORDER BY - SIRALAMA

- Ürünler tablosundaki ürünleri harf sırasına göre artan şekilde getir.

```
SELECT * FROM urunler ORDER BY urun_adi ASC
```

ASC : Artan
DESC : Azalan

- Kisiler tablosundan yaşı 18 olanları getir.

```
SELECT * FROM kisiler ORDER BY urun_fiyati DESC
```

	urun_id	urun_adi	urun_fiyati
	Filtre	Filtre	Filtre
1	1	Bilgisayar	10000
2	2	Telefon	8000
3	3	Saat	3000
4	4	Mont	750
5	6	TV	8000
6	7	Gözlük	1200

LIKE – BENZERLİK ARAMA

SELECT * FROM urunler WHERE urun_adi like '%a%'

urun_id	urun_adi	urun_fiyati
Filtre	Filtre	Filtre
1	1 Bilgisayar	10000
2	2 Telefon	8000
3	3 Saat	3000
4	4 Mont	750
5	6 TV	8000
6	7 Gözlük	1200

LIMIT - SINIRLI SAYIDA VERİ GETİR

```
SELECT * FROM urunler LIMIT 2
```

```
SELECT * FROM urunler WHERE urun_fiyati < 5000 LIMIT 2
```

```
SELECT * FROM urunler ORDER BY RANDOM() LIMIT 2
```

	urun_id	urun_adi	urun_fiyati
	Filtre	Filtre	Filtre
1	1	Bilgisayar	10000
2	2	Telefon	8000
3	3	Saat	3000
4	4	Mont	750
5	6	TV	8000
6	7	Gözlük	1200

SQLite

SQLite Tablo Oluşturma

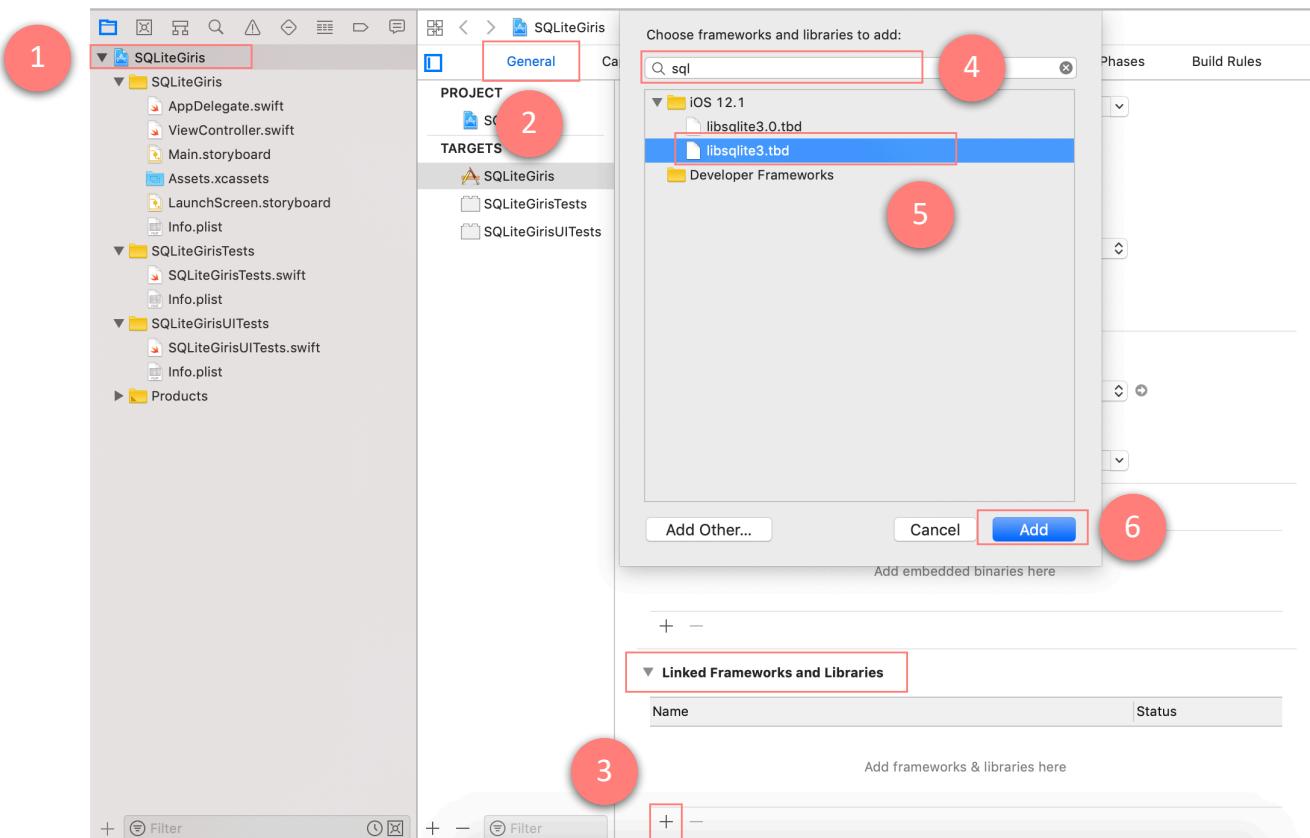
```
CREATE TABLE yonetmenler(  
    yonetmen_id INTEGER PRIMARY KEY AUTOINCREMENT,  
    yonetmen_ad TEXT  
);  
  
CREATE TABLE kategoriler(  
    kategori_id INTEGER PRIMARY KEY AUTOINCREMENT,  
    kategori_ad TEXT  
);  
  
CREATE TABLE filmler (  
    film_id INTEGER PRIMARY KEY AUTOINCREMENT,  
    film_ad TEXT,  
    film_yil TEXT,  
    film_resim TEXT,  
    kategori_id INTEGER,  
    yonetmen_id INTEGER,  
    FOREIGN KEY (kategori_id) REFERENCES kategoriler(kategori_id),  
    FOREIGN KEY (yonetmen_id) REFERENCES yonetmenler(yonetmen_id)  
);
```

SQLite İlişki İçeren Tablodan Veri Alma

```
SELECT * FROM filmler, yonetmenler, kategoriler  
WHERE filmler.yonetmen_id = yonetmenler.yonetmen_id  
AND filmler.kategori_id = kategoriler.kategori_id
```

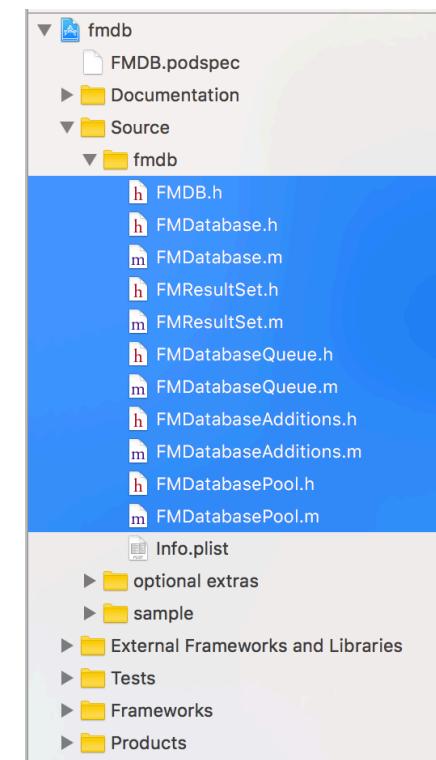
SQLite Kurulumu

Sqlite Kütüphanesinin Projeye Eklenmesi

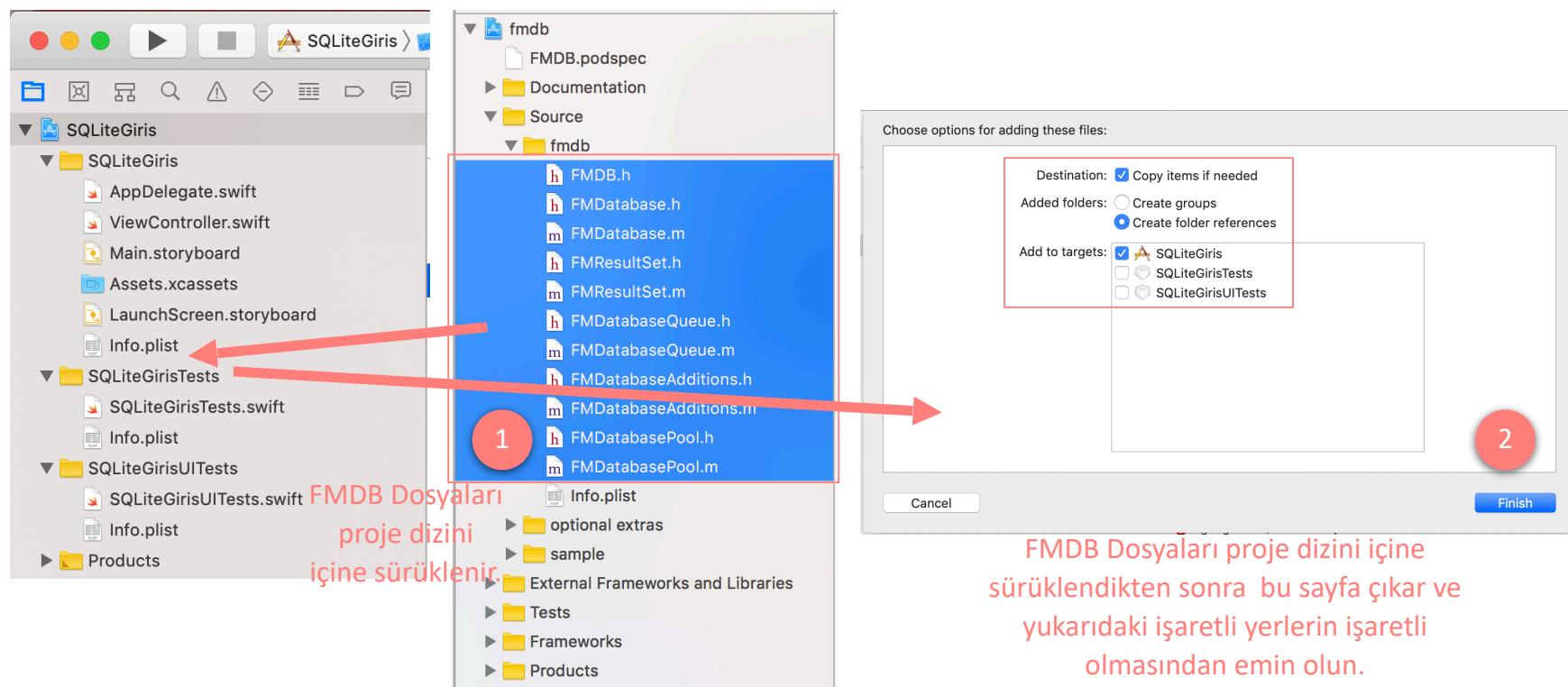


FMDB Dosyaları

- FMDB dosyaları <https://github.com/ccgus/fmdb.git> üzerinden indirilebilir.
- Bu dosyalar **Objective – c** dili ile yazılmıştır.
- Dosyalar sol tarafta mavi seçili olan dosyalardır.
- Bu dosyalar resimde olduğu gibi seçilir ve projeye eklenir.
- Eklenirken **if needed copy items** seçeneği seçili olmalıdır.

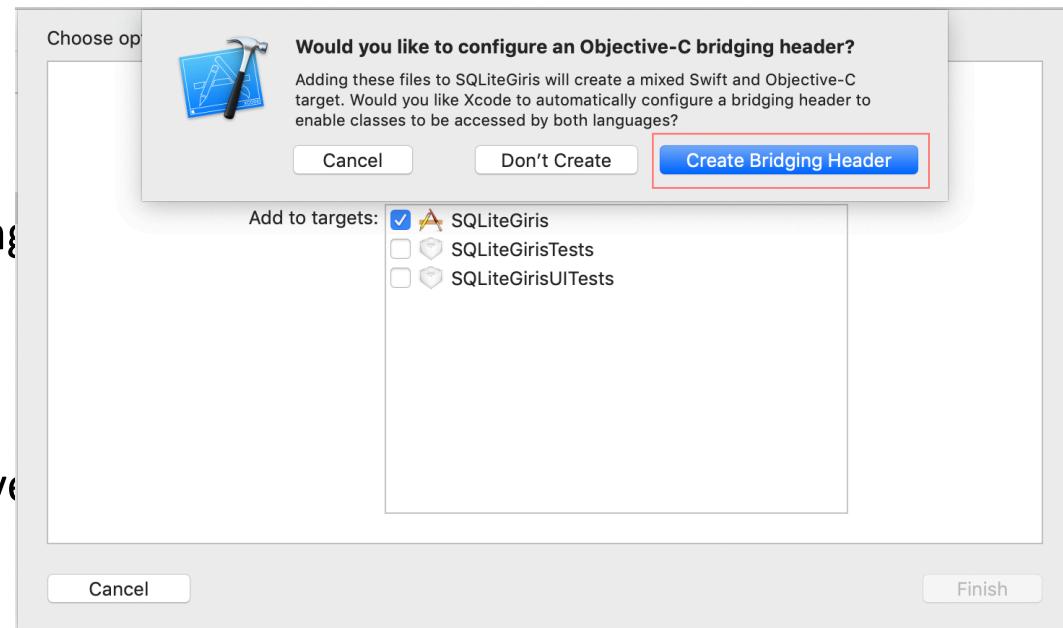


FMDB Dosyalarının Xcode Projesine Eklenmesi



FMDB Dosyaları ile Xcode Arasında Köprü Oluşturma

- FMDB Dosyaları proje eklendikten sonra Xcode üzerinde uyarı çıkar.
- Çıkan uyarıda Create Bridging Header seçersek FMDB Dosyaları ile Xcode arasında köprü oluşturur.
- Bu köprünün amacı Objective-C ile Swift arasındaki uyumu sağlar.



Köprü Dosyasının Kodlanması

- Köprü dosyasının içine `#import "FMDB.h"` ekliyor ve Xcode projesini kaydediyoruz.

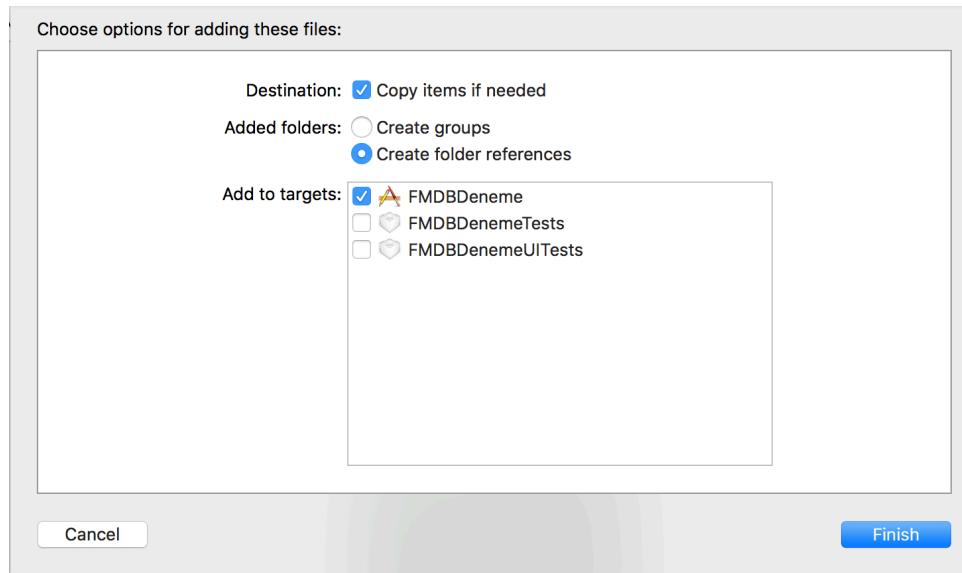
The screenshot shows the Xcode interface with the project structure on the left and the code editor on the right. The project navigator shows a folder named 'SQLiteGiris' containing a subfolder 'SQLiteGiris' which includes files like AppDelegate.swift, ViewController.swift, Main.storyboard, Assets.xcassets, LaunchScreen.storyboard, Info.plist, and several Objective-C header and source files for FMDatabase. The code editor displays the 'SQLiteGiris-Bridging-Header.h' file with the following content:

```
1 //  
2 // Use this file to import your target's public headers that you would  
3 //  
4 //  
5 #import "FMDB.h"  
6
```

The line `#import "FMDB.h"` is highlighted with a red rectangle. The file name 'SQLiteGiris-Bridging-Header.h' is also highlighted with a red rectangle in the project navigator.

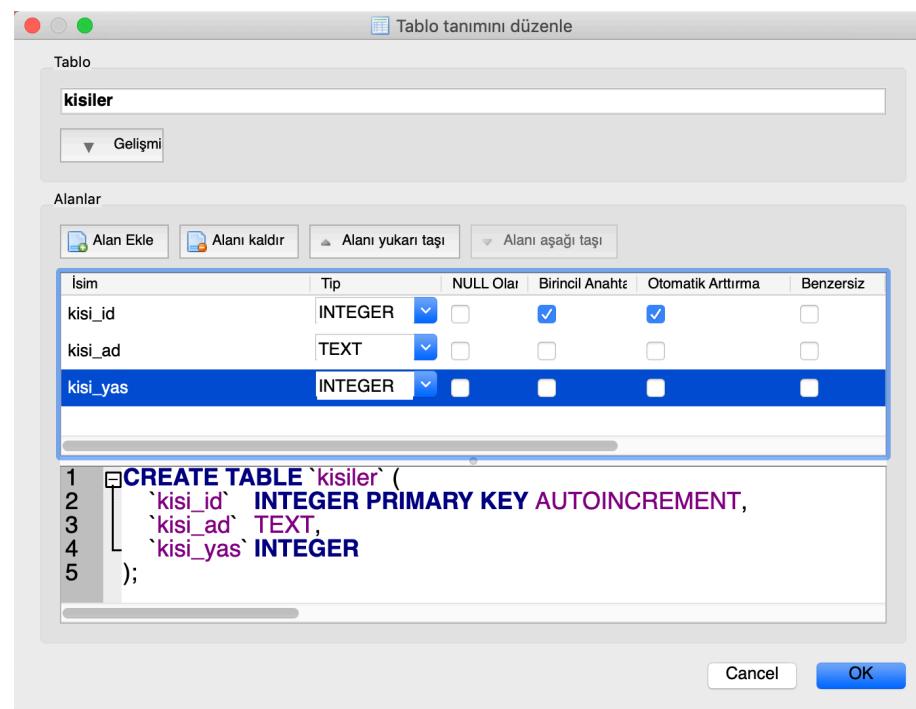
SQLite Veri tabanının Projemize Eklenmesi

- Bu işlemi yapmadan önce veri tabanımızı oluşturmalıyız.
- Oluşturduğumuz Sqlite veri tabanını Xcode projemizin içine sürüklüyoruz.
- Kopyalama işlemi unutulmamalıdır bunun için resimdeki yerler seçili olmalıdır.



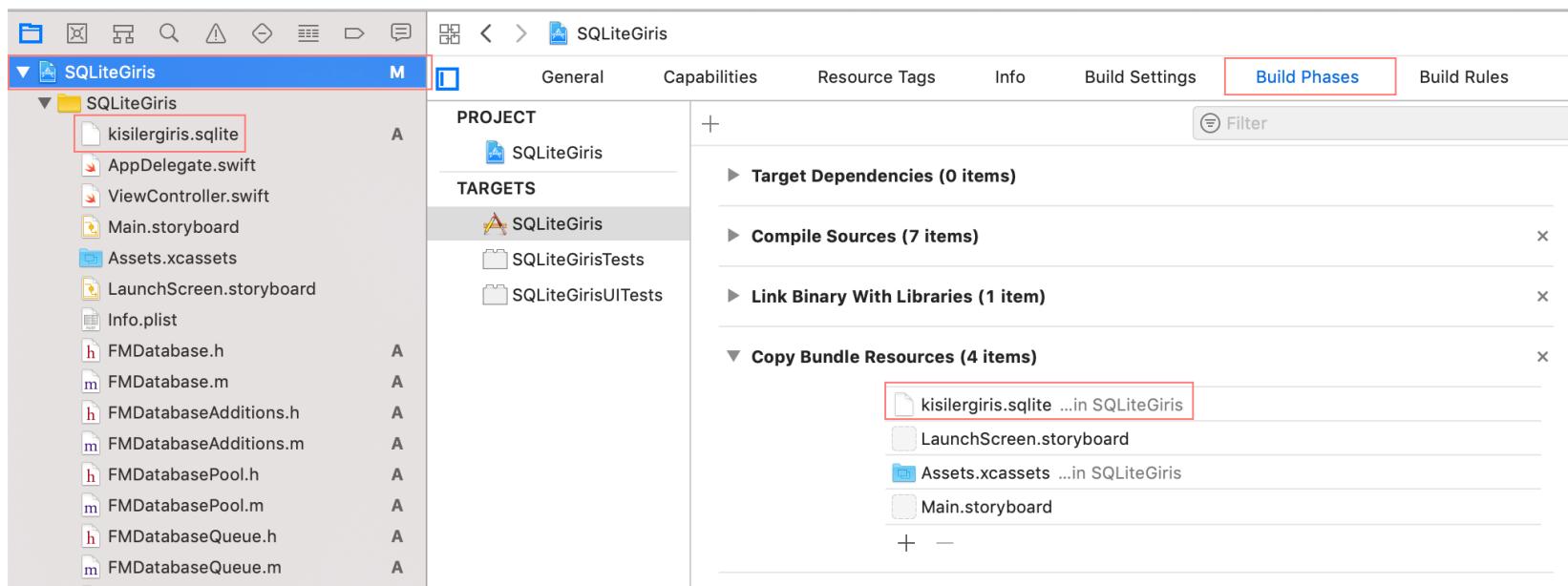
Veri tabanı Oluşturma

- DB Browser for SQLite programı üzerinde veri tabanı ve tablolar oluşturulmalıdır.



Veri tabanının Xcode projesi **Bundle** içinde olduğunun kontrol edilmesi

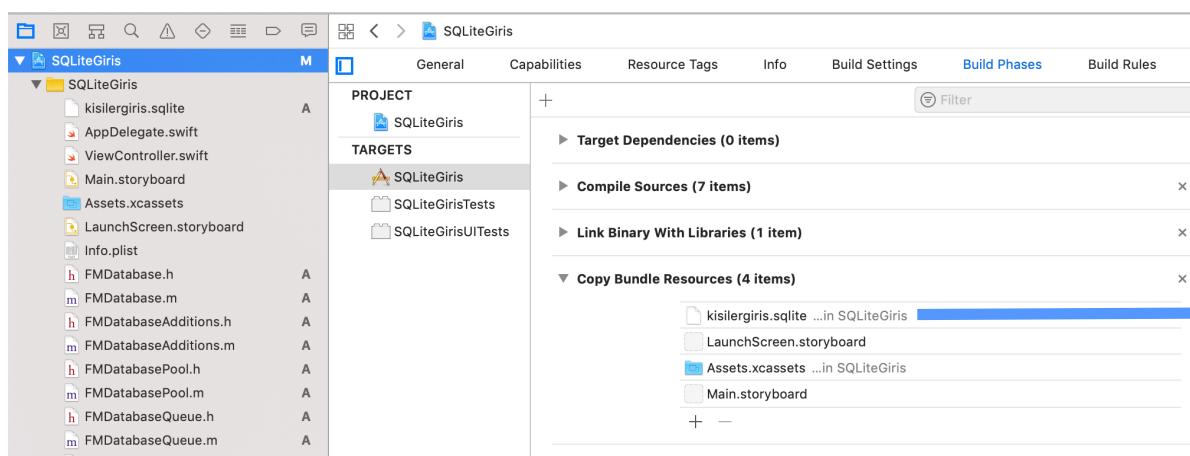
- Bundle Xcode projemiz içinde yer alan medya, veri tabanı vb dosyaların bulunduğu alandır.



Yazılımsal Kurulum

Veri Tabanının Kopyalanması

- Veri tabanı bundle içinde yer alır.
- Uygulama ilk çalıştırıldığında bundle'dan cihazın içine kopyalanmalıdır.
- Bir kere kopyalandıktan sonra kopyalandığı yerden veri tabanı üzerinde işlemler yapılabilir.
- Kopyalama işlemi uygulamanın ilk sayfası açıldığında çalıştırılmalıdır.



Veri Tabanı Kopyalama Kodlaması

```
import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

        veritabaniKopyala()
    }

    func veritabaniKopyala(){
        //Veritabanının bundle üzerindeki yeri
        let bundleYolu = Bundle.main.path(forResource: "kisilergiris", ofType: ".sqlite")

        let hedefYol = NSSearchPathForDirectoriesInDomains(.documentDirectory, .userDomainMask, true).first!

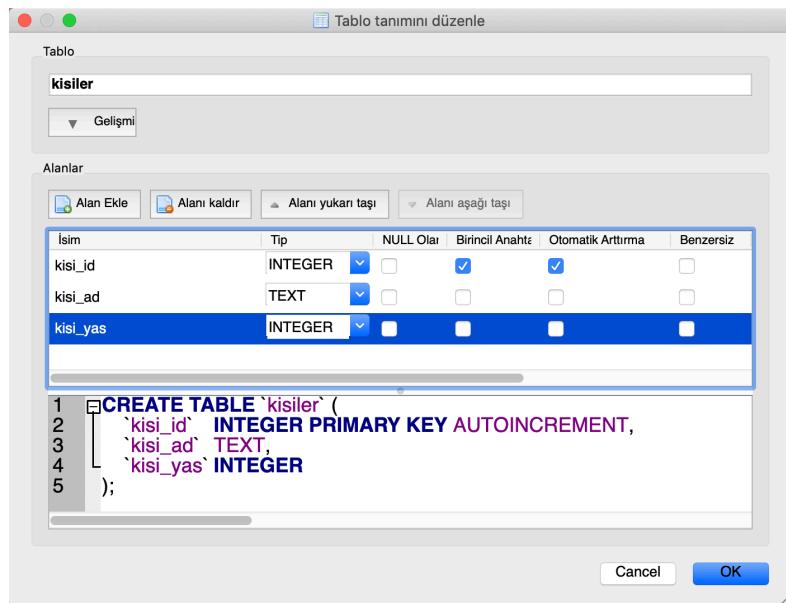
        let fileManager = FileManager.default //Kopyalama işlemi için FileManager gereklidir.

        //Kopyalama yapılacak yer
        let kopyalanacakYer = URL(fileURLWithPath: hedefYol).appendingPathComponent("kisilergiris.sqlite")

        if fileManager.fileExists(atPath: kopyalanacakYer.path) {
            //Kopyalama yaparken veritabanının daha önce kopyalanıp kopyalanmadığı sorulur.
            print("Veritabani zaten var.Kopyalamaya Gerek Yok.")
        }else{
            //Kopyalanmadıysa kopyalama işlemi yapılır.
            do {
                try fileManager.copyItem(atPath: bundleYolu!, toPath: kopyalanacakYer.path)
                //Kopyalama işlemi
            }catch{
                print(error)
            }
        }
    }
}
```

Veri Tabanı Modeli için Swift Sınıfı

- Her bir veri tabanı tablosunu temsil eden bir swift sınıfı oluşturulmalıdır.



```
import Foundation

class Kisiler {
    var kisi_id:Int?
    var kisi_ad:String?
    var kisi_yas:Int?

    init() {
    }

    init(kisi_id:Int,kisi_ad:String,kisi_yas:Int) {
        self.kisi_id = kisi_id
        self.kisi_ad = kisi_ad
        self.kisi_yas = kisi_yas
    }
}
```

Veri Tabanı İşlemleri için Swift Sınıfları

- Ayrıca her bir veri tabanı tablosu üzerinde yapılacak işlemleri temsil eden sınıf oluşturulur.
- Bu sınıfı dao (database access object) sınıfı denir.

```
import Foundation

class Kisilerdao {
    let db:FMDatabase?

    init() {
        //Kopyalanmış veritabanını kopyalandığı yerden alarak kullanılmaya hazır hale getirilir.
        let hedefYol = NSSearchPathForDirectoriesInDomains(.documentDirectory, .userDomainMask, true).first!
        let veritabaniURL = URL(fileURLWithPath: hedefYol).appendingPathComponent("kisilergiris.sqlite")

        db = FMDatabase(path: veritabaniURL.path)//Veritabanına bağlanmak için nesne
    }

    func tumKisileriAl() -> [Kisiler]{
        var liste = [Kisiler]()
        db?.open()//Veritabanı bağlantısı açılır.

        do {
            let rs = try db!.executeQuery("SELECT * FROM kisiler", values: nil)

            while rs.next() {
                let kisi = Kisiler(kisi_id: Int(rs.string(forColumn: "kisi_id"))!
                    , kisi_ad: rs.string(forColumn: "kisi_ad")!
                    , kisi_yas: Int(rs.string(forColumn: "kisi_yas"))!)

                liste.append(kisi)
            }
        }catch{
            print(error.localizedDescription)
        }

        db?.close()//Veritabanı bağlantısı kapanır.
    }

    return liste
}
```

Kopyalanmış veri tabanına ismi ile her seferinde erişmeliyiz.

Veri Tabanına Erişime Yakından Bakış

```
import Foundation

class Kisilerdao {

    let db:FMDatabase?

    init() {
        //Kopyalandmış veritabanını kopyalandığı yerden alarak kullanılmaya hazır hale getirilir.
        let hedefYol = NSearchPathForDirectoriesInDomains(.documentDirectory, .userDomainMask, true).first!
        let veritabaniURL = URL(fileURLWithPath: hedefYol).appendingPathComponent("kisilergiris.sqlite")

        db = FMDatabase(path: veritabaniURL.path)//Veritabanına bağlanmak için nesne
    }
}
```

SQLite Veri tabanı İşlemleri

Select : Veri Tabanından verinin alınması

```
func tumKisileriAl() -> [Kisiler]{

    var liste = [Kisiler]()

    db?.open()//Veritabanı bağlantısı açılır.

    do {
        let rs = try db!.executeQuery("SELECT * FROM kisiler", values: nil)

        while rs.next() {
            let kisi = Kisiler(kisi_id: Int(rs.string(forColumn: "kisi_id"))!
                , kisi_ad: rs.string(forColumn: "kisi_ad")!
                , kisi_yas: Int( rs.string(forColumn: "kisi_yas"))!)

            liste.append(kisi)
        }
    }catch{
        print(error.localizedDescription)
    }

    db?.close()//Veritabanı bağlantısı kapanır.

    return liste
}
```

Insert : Veri kaydı

```
func kisiEkle(kisi_ad:String,kisi_yas:Int){  
    db?.open()  
  
    do {  
  
        try db!.executeUpdate("INSERT INTO kisiler (kisi_ad,kisi_yas) VALUES (?,?)", values: [kisi_ad,kisi_yas])  
    } catch {  
        print(error.localizedDescription)  
    }  
  
    db?.close()  
}
```

? işaretleri sırayla values dizisi içindeki verileri temsil eder.

values dizisi çeşitli türde verileri bir arada tutabilir.

Update : Güncelleme İşlemi

```
func kisiGuncelle(kisi_id:Int,kisi_ad:String,kisi_yas:Int){  
    db?.open()  
  
    do {  
  
        try db!.executeUpdate("UPDATE kisiler SET kisi_ad = ?,kisi_yas=? WHERE kisi_id = ?", values: [kisi_ad,kisi_yas,kisi_id])  
  
    } catch {  
        print(error.localizedDescription)  
    }  
  
    db?.close()  
}
```

Delete : Silme İşlemi

```
func kisiSil(kisi_id:Int){  
    db?.open()  
  
    do {  
  
        try db!.executeUpdate("DELETE FROM kisiler WHERE kisi_id = ?", values: [kisi_id])  
  
    } catch {  
        print(error.localizedDescription)  
    }  
  
    db?.close()  
}
```

Kayıt Kontrol

```
func kisiKontrol(kisi_ad:String) -> Int{
    var sonuc = 0
    db?.open()
    do {
        let rs = try db!.executeQuery("SELECT count(*) as sonuc FROM kisiler WHERE kisi_ad = ?", values: [kisi_ad])
        while rs.next() {
            sonuc = Int(rs.string(forColumn: "sonuc"))!
        }
    } catch {
        print(error.localizedDescription)
    }
    db?.close()
    return sonuc
}
```

Arama Yapma

```
func aramaYap(kisi_ad:String) -> [Kisiler]{  
  
    var liste = [Kisiler]()  
  
    db?.open()//Veritabanı bağlantısı açılır.  
  
    do {  
        let rs = try db!.executeQuery("SELECT * FROM kisiler WHERE kisi_ad like '%\($kisi_ad)%'", values: nil)  
  
        while rs.next() {  
            let kisi = Kisiler(kisi_id: Int(rs.string(forColumn: "kisi_id"))!  
                , kisi_ad: rs.string(forColumn: "kisi_ad")!  
                , kisi_yas: Int( rs.string(forColumn: "kisi_yas"))!)  
  
            liste.append(kisi)  
        }  
    }catch{  
        print(error.localizedDescription)  
    }  
  
    db?.close()//Veritabanı bağlantısı kapanır.  
  
    return liste  
}
```

Tek Bir Veri Getir

```
func kisiGetir(kisi_id:Int) -> Kisiler {  
  
    var kisi = Kisiler()  
  
    db?.open()//Veritabanı bağlantısı açılır.  
  
    do {  
        let rs = try db!.executeQuery("SELECT * FROM kisiler WHERE kisi_id = ?", values: [kisi_id])  
  
        while rs.next() {  
            kisi = Kisiler(kisi_id: Int(rs.string(forColumn: "kisi_id"))!  
                           , kisi_ad: rs.string(forColumn: "kisi_ad")!  
                           , kisi_yas: Int( rs.string(forColumn: "kisi_yas"))!)  
        }  
    }catch{  
        print(error.localizedDescription)  
    }  
  
    db?.close()//Veritabanı bağlantısı kapanır.  
  
    return kisi  
}
```

LIMIT : Sınırlı Veri Alma

```
func kisileriAl2() -> [Kisiler]{

    var liste = [Kisiler]()

    db?.open()//Veritabanı bağlantısı açılır.

    do {
        let rs = try db!.executeQuery("SELECT * FROM Kisiler LIMIT 2", values: nil)

        while rs.next() {
            let kisi = Kisiler(kisi_id: Int(rs.string(forColumn: "kisi_id"))!
                , kisi_ad: rs.string(forColumn: "kisi_ad")!
                , kisi_yas: Int( rs.string(forColumn: "kisi_yas"))!)

            liste.append(kisi)
        }
    }catch{
        print(error.localizedDescription)
    }

    db?.close()//Veritabanı bağlantısı kapanır.

    return liste
}
```

Rasgele 2 Veri Getirme

```
func rasgele2Kisi() -> [Kisiler]{

    var liste = [Kisiler]()

    db?.open()//Veritabanı bağlantısı açılır.

    do {
        let rs = try db!.executeQuery("SELECT * FROM kisiler ORDER BY RANDOM() LIMIT 2", values: nil)

        while rs.next() {
            let kisi = Kisiler(kisi_id: Int(rs.string(forColumn: "kisi_id"))!
                , kisi_ad: rs.string(forColumn: "kisi_ad")!
                , kisi_yas: Int( rs.string(forColumn: "kisi_yas"))!)

            liste.append(kisi)
        }
    }catch{
        print(error.localizedDescription)
    }

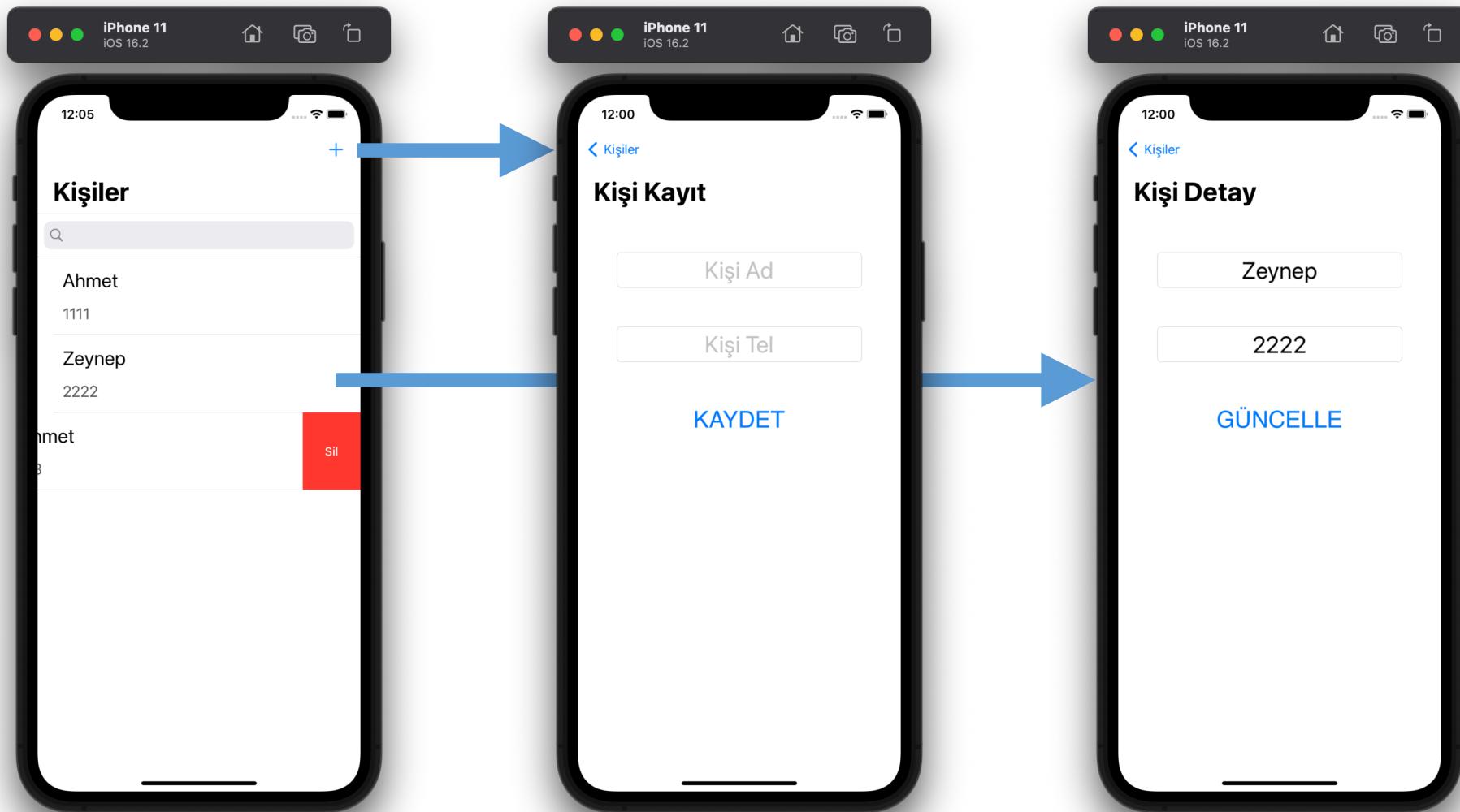
    db?.close()//Veritabanı bağlantısı kapanır.

    return liste
}
```

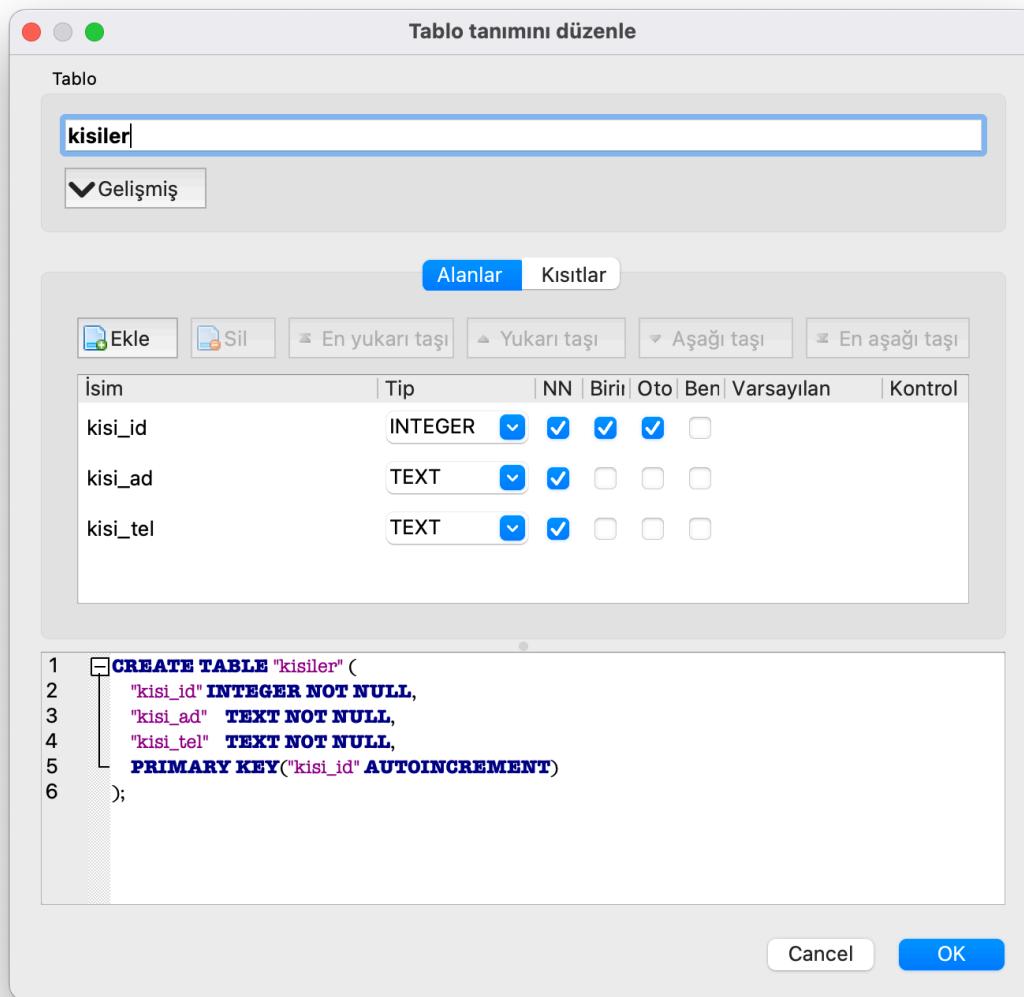
Kişiler Uygulaması

SQLite

Kişiler Uygulaması



Kişiler Uygulaması - SQLite



Tablo: **kisiler**

	kisi_id	kisi_ad	kisi_tel
Filtre	Filtre	Filtre	
1	1	Ali	9 9 9 9
2	2	Ece	8 8 8 8

The screenshot shows the SQLite database interface with the 'kisiler' table selected. The table has three columns: 'kisi_id', 'kisi_ad', and 'kisi_tel'. There are two rows of data: one for 'Ali' (kisi_id 1) and one for 'Ece' (kisi_id 2). The 'kisi_tel' column for both rows contains the value '9 9 9 9'.

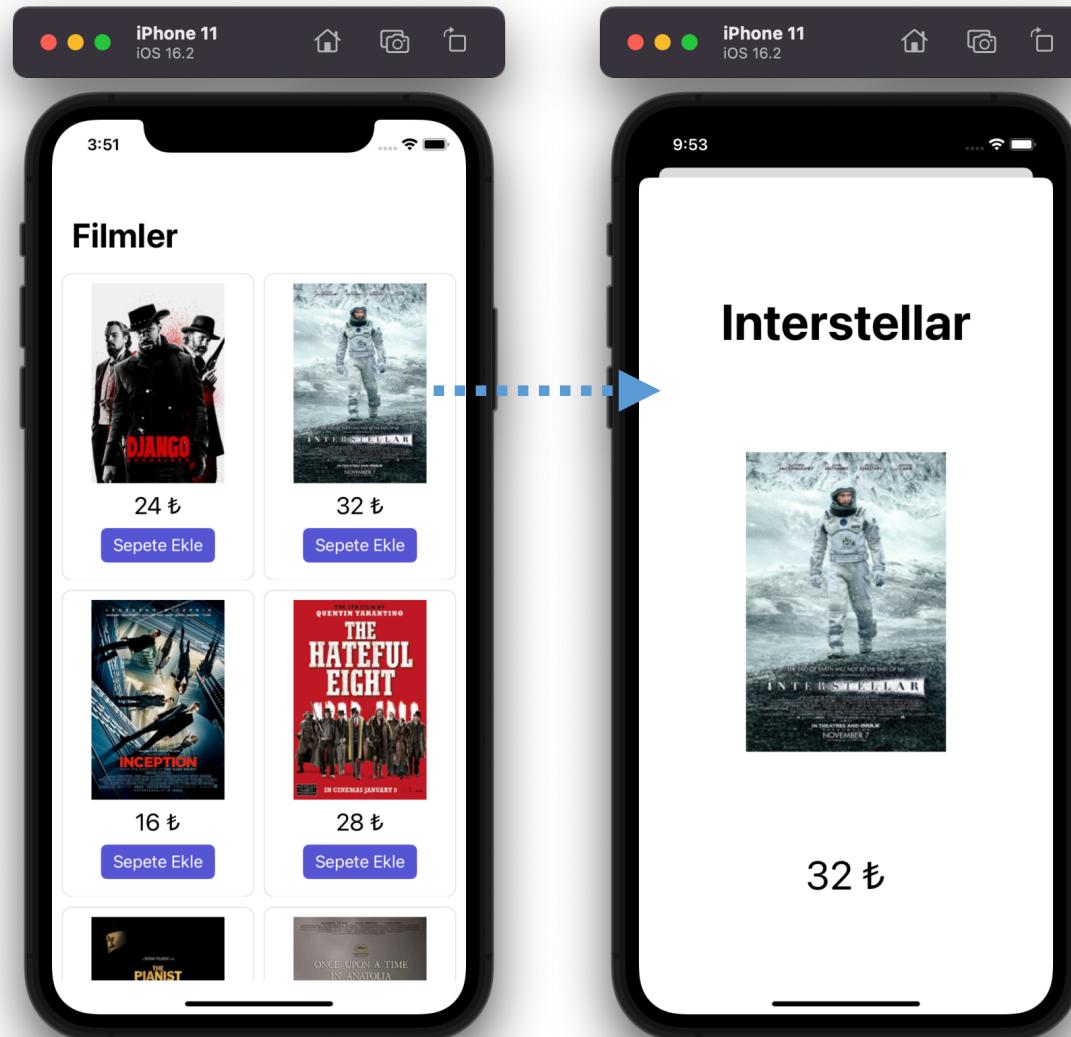
Veritabanı Kopyalama Kodlaması

```
func veritabaniKopyala(){
    let bundleYolu = Bundle.main.path(forResource: "rehber", ofType: ".sqlite")
    let hedefYol = NSSearchPathForDirectoriesInDomains(.documentDirectory, .userDomainMask, true).first!
    let kopyalanacakYer = URL(fileURLWithPath: hedefYol).appendingPathComponent("rehber.sqlite")
    let fileManager = FileManager.default
    if fileManager.fileExists(atPath: kopyalanacakYer.path){
        print("Veritabanı zaten var")
    }else{
        do{
            try fileManager.copyItem(atPath: bundleYolu!, toPath: kopyalanacakYer.path)
        }catch{}
    }
}
```

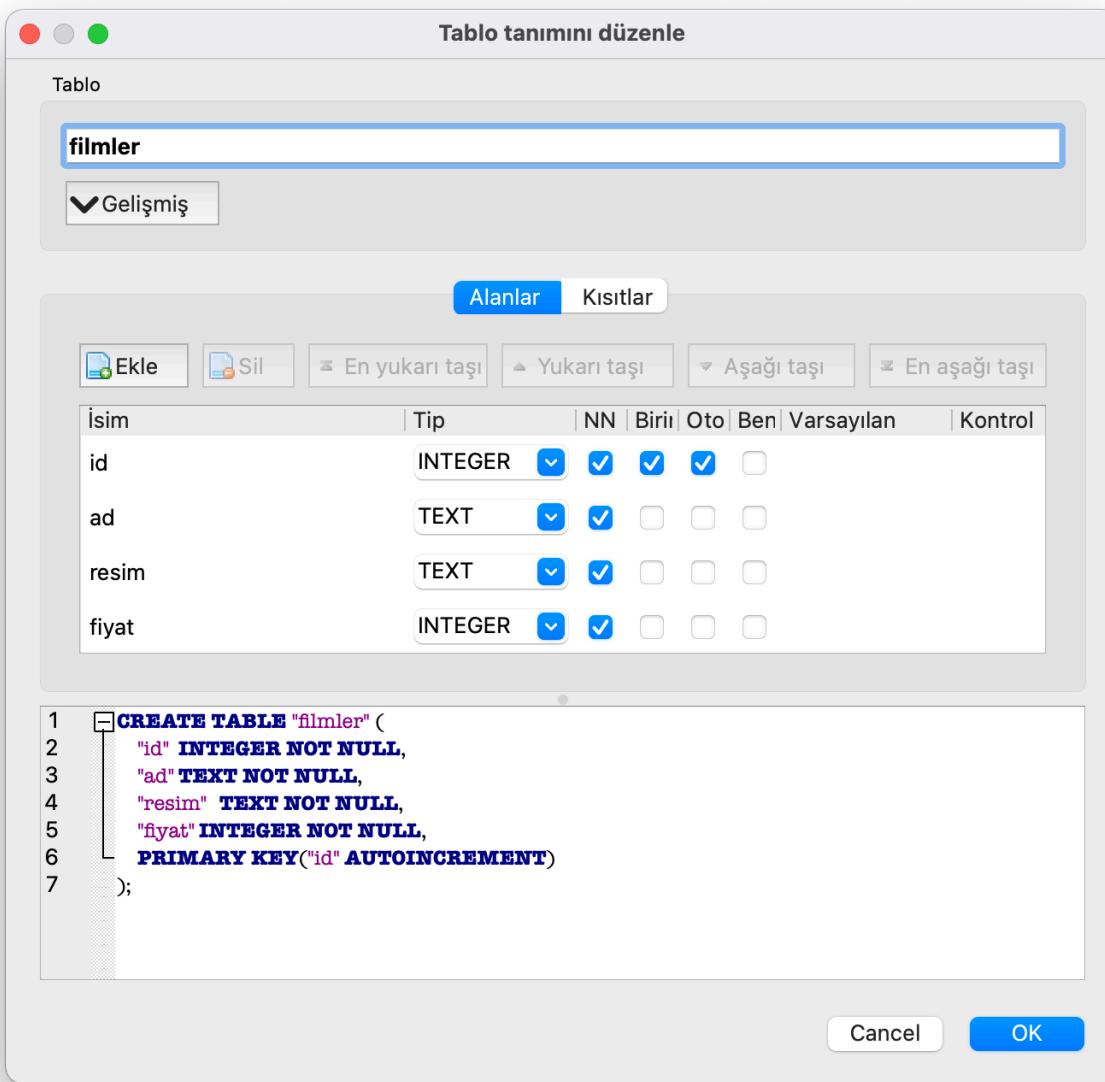
Filmler Uygulaması

SQLite

Film Uygulaması



Filmler Uygulaması - SQLite



Tablo: **filmers**

En haut En bas En haut En bas

	id	ad	resim	fiyat
	Filtr	Filtre	Filtre	Filtre
1	1	Django	django	2 4
2	2	Interstellar	interstellar	3 2
3	3	Inception	inception	1 6
4	4	The Hateful Eight	thehatefuleight	2 8
5	5	The Pianist	thepianist	1 8
6	6	Anadoluda	anadoluda	1 0

Veritabanı Kopyalama Kodlaması

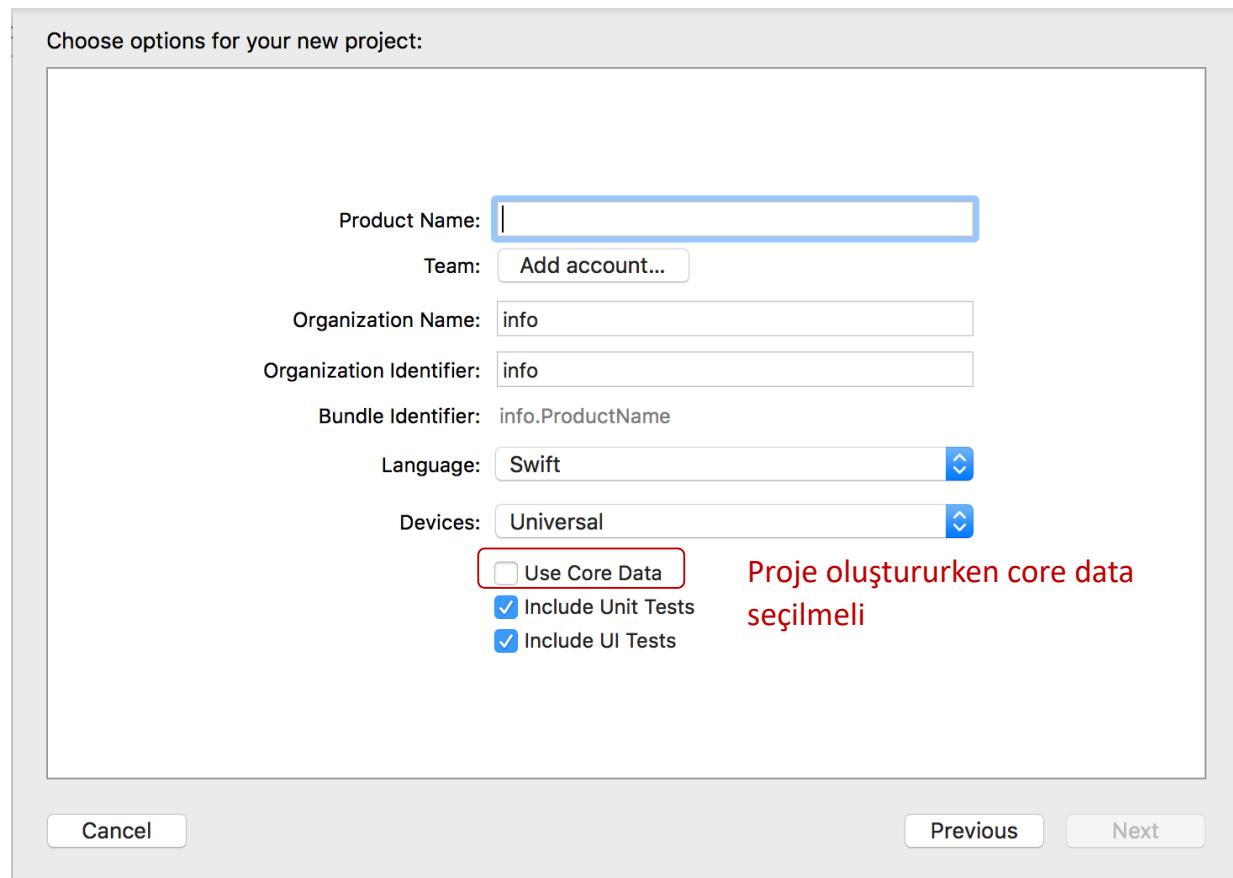
```
func veritabaniKopyala(){
    let bundleYolu = Bundle.main.path(forResource: "filmler_app", ofType: ".sqlite")
    let hedefYol = NSSearchPathForDirectoriesInDomains(.documentDirectory, .userDomainMask, true).first!
    let kopyalanacakYer = URL(fileURLWithPath: hedefYol).appendingPathComponent("filmler_app.sqlite")
    let fileManager = FileManager.default
    if fileManager.fileExists(atPath: kopyalanacakYer.path){
        print("Veritabanı zaten var")
    }else{
        do{
            try fileManager.copyItem(atPath: bundleYolu!, toPath: kopyalanacakYer.path)
        }catch{}
    }
}
```

CoreData

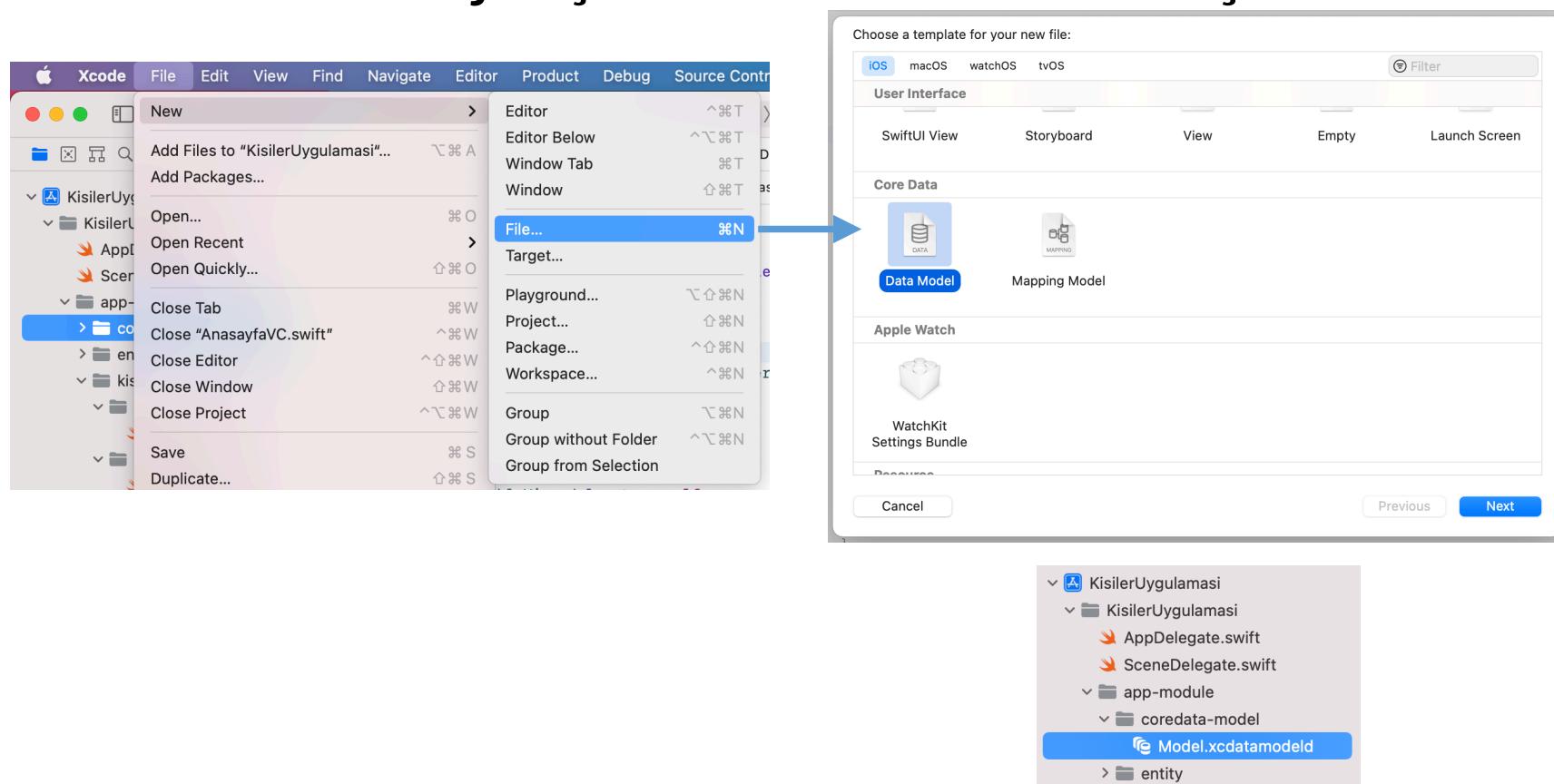
Core Data

- Core data nesnelerin düzenli şekilde saklanmasıdır.
- Core data geleneksel bir veri tabanı değildir.
- Sql sorguları içermez.
- Xcode içinde yer alan native bir framework'tur.
- Her bir veri tablosu için sınıflar oluşturmaktadır.
- Veri tabloları arasındaki ilişki nesne tabanlı dillerin sahip olduğu composition'dan yararlanmaktadır.

Yeni Proje Oluştururken Kurulum



Var Olan Proje için xcdatamodeld Oluşturma



ENTITIES**E** KisilerModel**FETCH REQUESTS****CONFIGURATIONS****C** Default**Attributes****Attribute****Type**

S kisi_ad

String



S kisi_tel

String



+ -

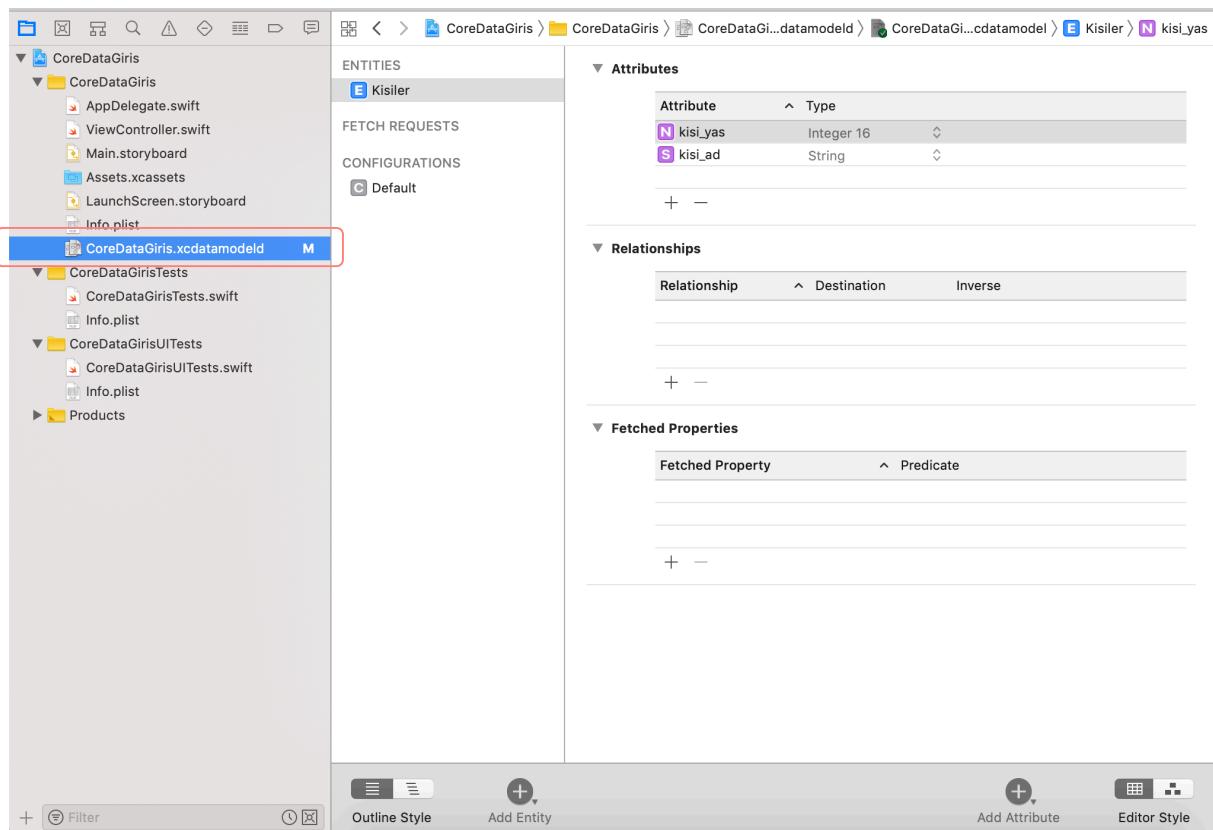
Relationships**Relationship****Destination****Inverse**

+ -

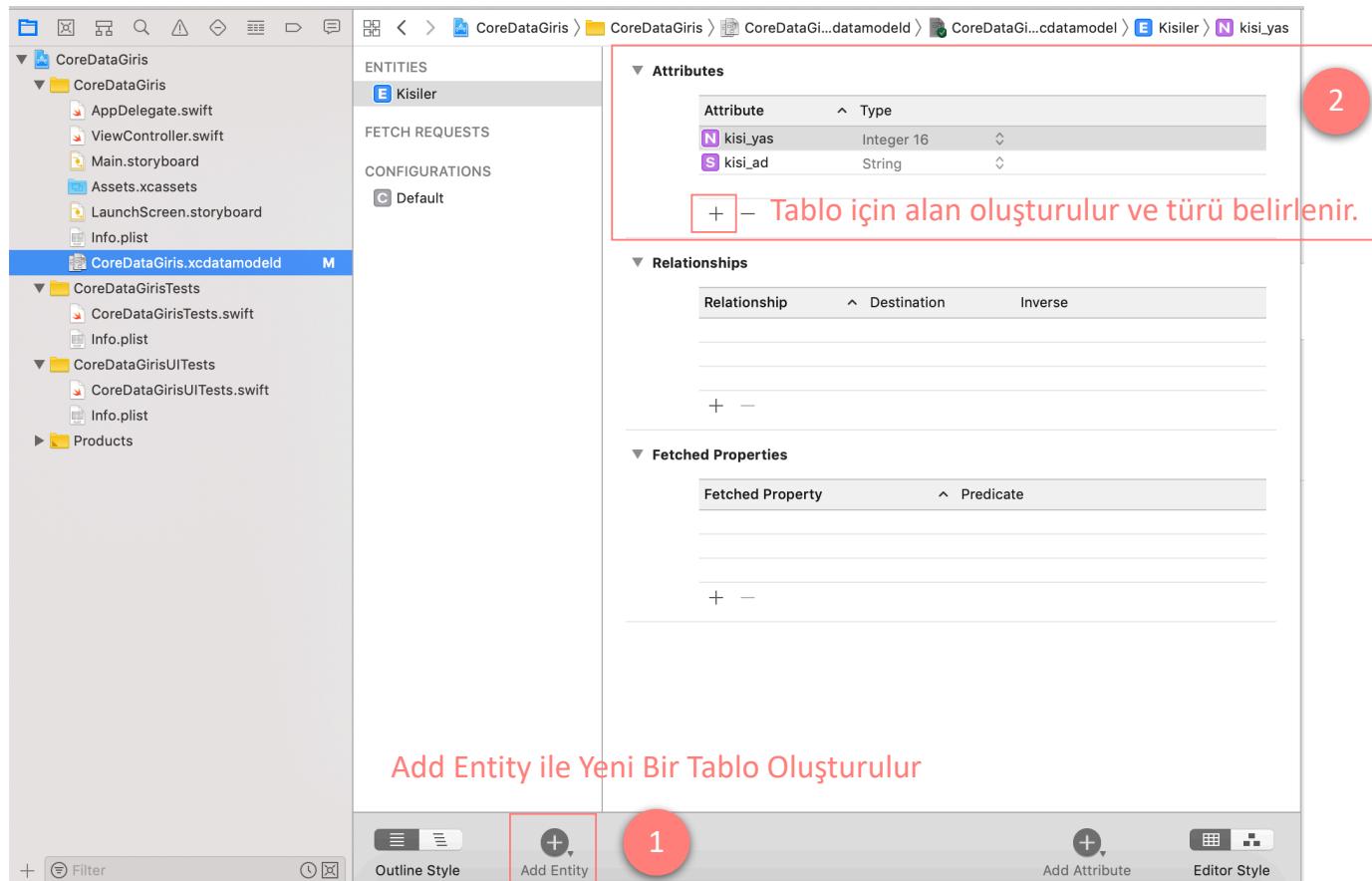
Fetched Properties**Fetched Property****Predicate**

+ -

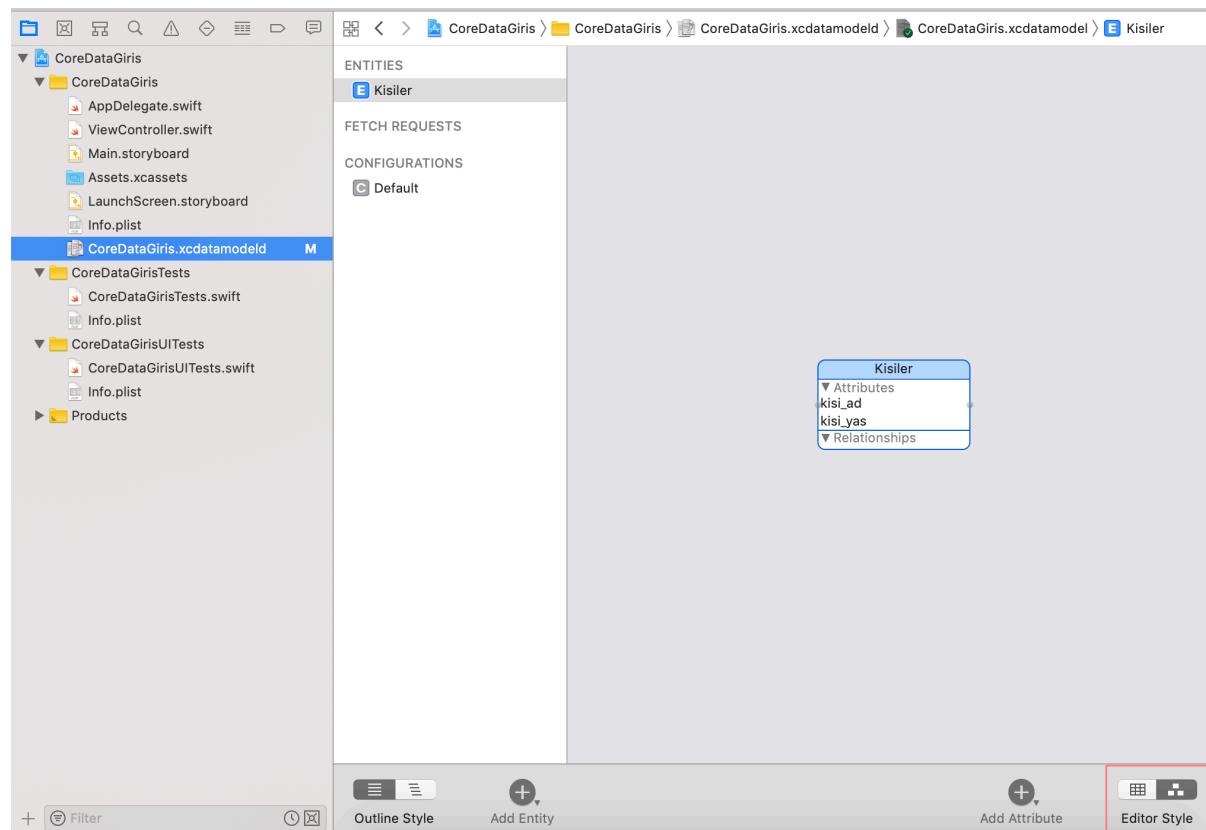
Core Data Editörün Açılması



Tablo Oluşturma

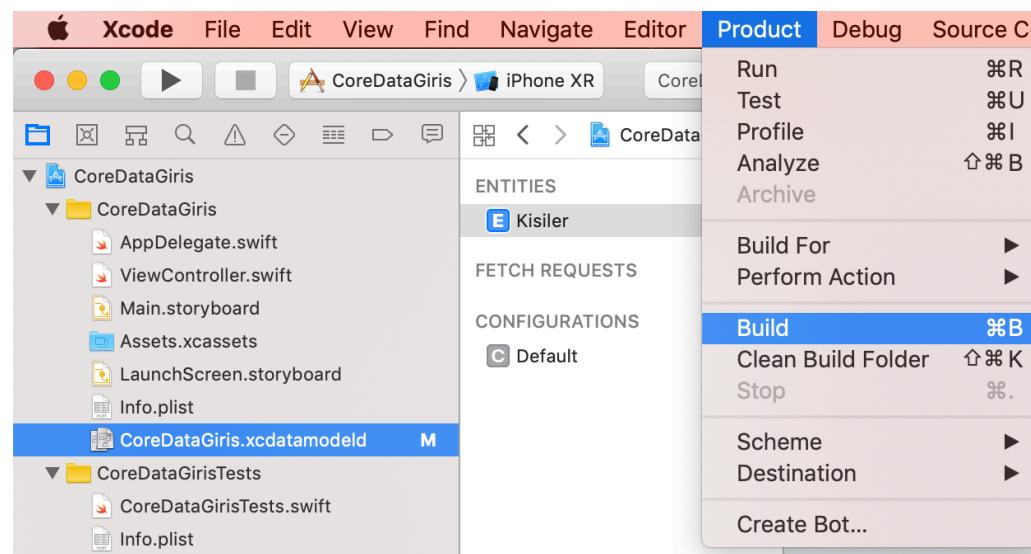


Tabloların Görüntülenme Şeklinin Değiştirilmesi



Veritabanının Xcode içinde aktif edilmesi

- Yapılan değişikler sonucunda projemizi build etmemiz gereklidir aksi halde gerekli kodlara erişemeyiz.



AppDelegate Kodlaması

```
import UIKit
import CoreData

@main
class AppDelegate: UIResponder, UIApplicationDelegate {

    lazy var persistentContainer: NSPersistentContainer = {
        let container = NSPersistentContainer(name: "Model") //Coredata dosyasının adı Model.xcdatamodeld
        container.loadPersistentStores(completionHandler: { (storeDescription, error) in
            if let error = error {
                fatalError("Unresolved error, \(error as NSError).userInfo")
            }
        })
        return container
   }()

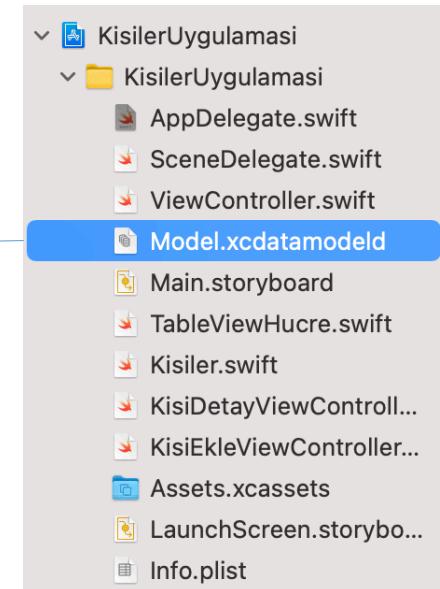
    func saveContext() {
        let context = persistentContainer.viewContext
        if context.hasChanges {
            do {
                try context.save()
            } catch {
                let nserror = error as NSError
                fatalError("Unresolved error \(nserror), \(nserror.userInfo)")
            }
        }
    }

    func application(_ application: UIApplication, didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
    }
}
```

Not : Bu kodlamalar projeyi ilk oluşturma kısmında coredata seçildiğinde otomatik oluşturmaktadır. Oluşmuyorsa kendimiz yazmalıyız.

Kurulum için gerekli metod

Veritabanı üzerinde işlem yapmak için gerekli metod



Kodal Kurulum

```
import UIKit  
import CoreData
```

Kütüphane import edilmelidir.
Her işlem için kullanılmamaktadır ama gerekli olabileceği için eklenir.

```
let appDelegate = UIApplication.shared.delegate as! AppDelegate  
//AppDelegate nesnesi gereklidir.
```



```
class ViewController: UIViewController {
```

AppDelegate dosyasında tanımlı metod

```
    let context = appDelegate.persistentContainer.viewContext  
    //context ile işlem yapılabilir.
```

Veri Kaydı

```
let kisi = Kisiler(context: context)
//Kisiler sınıfı oluşturduğumu tabloyu temsil eder.
//Kişiler sınıfın alanlarına veriler eklenir.Yani tablo alanlarına
kisi.kisi_ad = "Ahmet"
kisi.kisi_yas = 16

// Veri kaydı yapılır.
appDelegate.saveContext()
```

AppDelegate dosyasında tanımlı metod

Veri Okuma

```
let appDelegate = UIApplication.shared.delegate as! AppDelegate  
//AppDelegate nesnesi gereklidir.  
  
class ViewController: UIViewController {  
  
    let context = appDelegate.persistentContainer.viewContext  
    //context ile işlem yapılabilir.  
  
    var kisilerListe = [Kisiler]()  
    //Tablo alanlarını temsil eden sınıfından oluşan bir liste.  
    //Kayıtlar bu liste içinde tutulacak
```

```
do {  
    kisilerListe = try context.fetch(Kisiler.fetchRequest())  
    //İlgili tablodan bütün veriler alınır ve listeye aktarılır.  
} catch {  
    print("Fetching Failed")  
}  
  
//Verileri gösterme  
for k in kisilerListe {  
    print("Ad : \(k.kisi_ad!) - Yaş : \(k.kisi_yas)")  
}
```

Veri Silme

```
let kisi = kisilerListe[1]
//Belirtilen indeksteki kişi nesnesi alınır.
self.context.delete(kisi)
//Silme işlemi gerçekleşir.
appDelegate.saveContext()
```

Veri Güncelleme

```
let kisi = kisilerListe[1]
//Belirtilen indeksteki kisi nesnesi alınır.
kisi.kisi_ad = "Güncel Ad"
kisi.kisi_yas = 99
//Nesneye yeni veriler eklenir.
appDelegate.saveContext()
```

Veri Sıralama (Sort)

```
import CoreData

//Bu işlemi yapmak için coredata import edilmelidir.
let fetchRequest: NSFetchedRequest<Kisiler> = Kisiler.fetchRequest()

//Hangi alan üzerinde sıralama yapacağımızı belirtiyoruz.
//Kisiler tablosunun kisi_yas alanında sıralama yapılacak.
//ascending: true -> küçükten büyüğe
//ascending: false -> büyükten küçüğe
let sort = NSSortDescriptor(key: #keyPath(Kisiler.kisi_yas), ascending: true)

fetchRequest.sortDescriptors = [sort]//Sıralama fetchRequest nesnesine eklenir.

do {
    kisilerListe = try context.fetch(fetchRequest)
    //Yeni fetchRequest nesnesi ile veri alırken sıralama değişir.
} catch {
    print("Cannot fetch Expenses")
}

//Verileri gösterme
for k in kisilerListe {
    print("Ad : \(k.kisi_ad!) - Yaş : \(k.kisi_yas)")
}
```

Veri filtreleme (Predicate) : Sorgu

```
//@ = String i = integer Temsil eder.  
let fetchRequest: NSFetchedRequest<Kisiler> = Kisiler.fetchRequest()  
  
//Sorgu oluşturulur.kisi yaşı 34'e eşit olan veriler almak için.  
fetchRequest.predicate = NSPredicate(format: "kisi_yas == %i", 34)  
  
do {  
    kisilerListe = try context.fetch(fetchRequest)  
    //Yeni fetchRequest nesnesi ile sorgu sonucu veriler alınır.  
} catch {  
    print("Cannot fetch Expenses")  
}  
  
//Verileri gösterme  
for k in kisilerListe {  
    print("Ad : \(k.kisi_ad!) - Yaş : \(k.kisi_yas)")  
}
```

Çeşitli Sorgu Yapıları

```
//@ = String  i = integer
let fetchRequest: NSFetchedResultsController<Kisiler> = Kisiler.fetchRequest()
//sorgu oluşturulur.kisi yaşı 26 eşitse
fetchRequest.predicate = NSPredicate(format: "kisi_yas == %i", 26)
//kişinin adı ahmet'e eşitse
fetchRequest.predicate = NSPredicate(format: "kisi_ad == %@", "ahmet")
//kişinin yaşı 18 den büyükse
fetchRequest.predicate = NSPredicate(format: "kisi_yas > %i", 18)
//kişinin adı ahmet'e eşitse
fetchRequest.predicate = NSPredicate(format: "%K == %i", "kisi_ad", "ahmet")
//kişinin adı a harfini içeriyorsa
fetchRequest.predicate = NSPredicate(format: "kisi_ad CONTAINS %@", "a")
//iki şart kullanılabilir.
fetchRequest.predicate = NSPredicate(format: "kisi_yas == %i and kisi_ad == %@", 26, "ahmet")
//ilişkisi olduğu tablodan sorgu olabilir.
//composition yaparak erişim sağlanır.
fetchRequest.predicate = NSPredicate(format: "kisiler_is.is_ad == %@", "spor")
```

```
import UIKit
import CoreData

let appDelegate = UIApplication.shared.delegate as! AppDelegate

@main
class AppDelegate: UIResponder, UIApplicationDelegate {

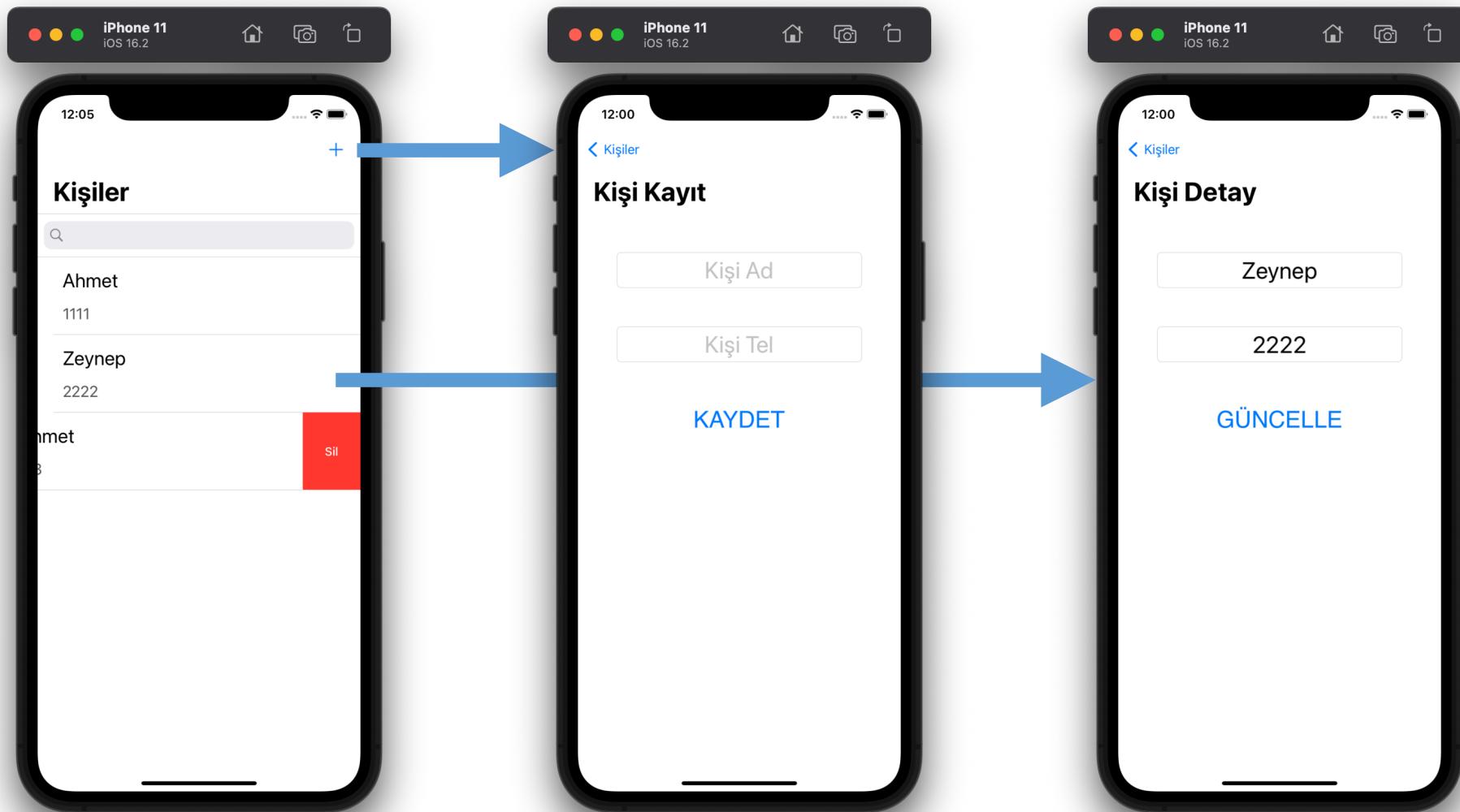
    lazy var persistentContainer: NSPersistentContainer = {
        let container = NSPersistentContainer(name: "Model")
        container.loadPersistentStores(completionHandler: { (storeDescription,error) in
            if let e = error {
                print("Hata : \(e as NSError).userInfo")
            }
        })
        return container
    }()

    func saveContext(){
        let context = persistentContainer.viewContext
        if context.hasChanges {
            do{
                try context.save()
            }catch{
                print("Hata : \((error as NSError).userInfo)")
            }
        }
    }
}
```

Kişiler Uygulaması

CoreData

Kişiler Uygulaması



CoreData Kurulum Kodlamasi

```
import CoreData

let appDelegate = UIApplication.shared.delegate as! AppDelegate

@main
class AppDelegate: UIResponder, UIApplicationDelegate {

    lazy var persistentContainer: NSPersistentContainer = {
        let container = NSPersistentContainer(name: "Model")
        container.loadPersistentStores(completionHandler: { (storeDescription,error) in
            if let e = error {
                print("Hata : \((e as NSError).userInfo)")
            }
        })
        return container
   }()

    func saveContext(){
        let context = persistentContainer.viewContext
        if context.hasChanges {
            do{
                try context.save()
            }catch{
                print("Hata : \((error as NSError).userInfo)")
            }
        }
    }
}
```

İnternet Tabanlı İşlemler

RESTful Mimarisi Aşamaları

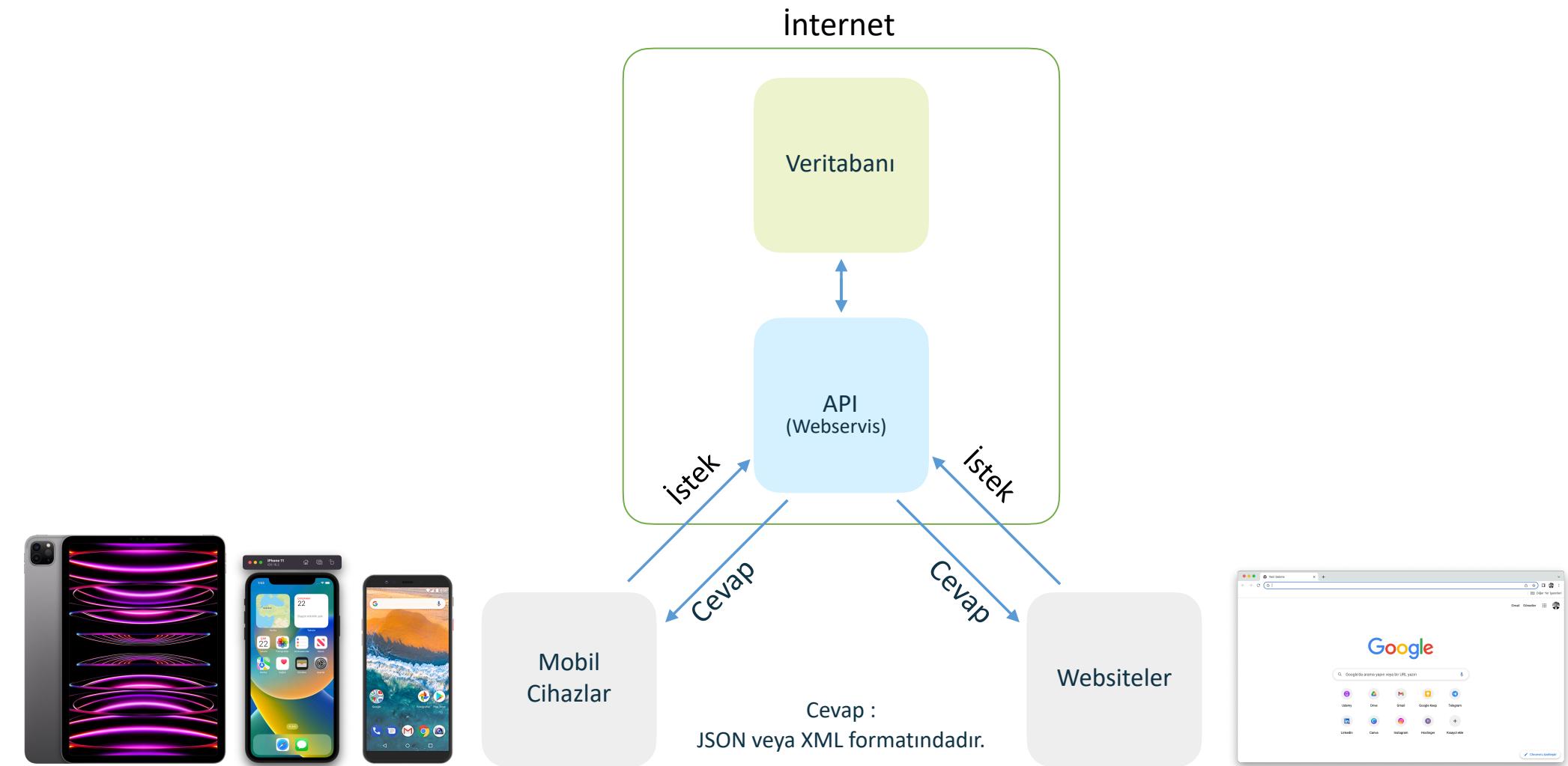
RESTful Mimarisi Aşamaları

- RESTful Mimarisi Giriş
- Domain Adresi Alma
- Hosting Hizmeti Alma
- Veritabanı Oluşturma
- Veritabanı Üzerinde Tablo Oluşturma
- Veritabanı - Insert - Update - Delete İşlemleri
- Veritabanı - Select Sorguları
- Webservis Giriş
- Veritabanı Bağlantı Bilgileri Dosyası Oluşturma
- Webservis Oluşturma
- Webservisin Hosting İşlemi
- Webservisin Çalıştırılması

RESTful Web Services

- REST : Representational State Transfer
- Web servise , Bilgileri basit bir formatta transfer etmek için kullanılır.
- XML ve JSON en çok kullanılan formattır.
- Web tabanlı olduğu için internet izni istenmektedir.

RESTful Mimarisi



Domain Adresi Alma

- İnternet üzerinde veritabanı ve dosyalama işlemleri için hosting hizmeti almamız gereklidir.
- Hosting hizmeti içinde bir tane domain adresi olması gereklidir.
- Bazı hosting hizmetlerinde ücretsiz domain verilmektedir.
- Domain adresi internet hizmetinize erişeceğiniz website ismi gibi düşünebilirsiniz.
- Örn : kasimadalan.pe.hu

Home | hPanel

hpanel.hostinger.com

HOSTINGER Home Websites Hosting Emails Domains VPS SSL Billing

Hi, KASIM 🙌

Search

Hosting

Single Web Hosting
Expires on 2024-06-23 [kasimadalan.pe.hu](#) Manage

Email

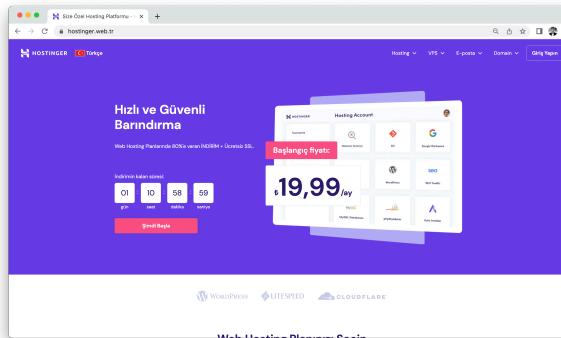
kasimadalan.pe.hu Manage

hpanel-main.hostinger.com bekleniyor...

390

Hosting Hizmeti Alma

- İnternet üzerinde veritabanı ve dosyalama işlemleri için hosting hizmeti almamız gereklidir.
- Hosting hizmeti temelde 24 saat sürekli çalışan bilgisayarları kullanmanızı sağlar.
- Bu hizmet genellikle ücretlidir.
- Hizmeti alırken veritabanı kullanımı sağlandığından emin olunmalıdır.
- Ek olarak dosyalama sistemi var mı , phpMyAdmin arayüzünü destekliyor mu bunlar araştırılmalıdır.



Veritabanı Oluşturma

The screenshot shows the Hostinger hPanel dashboard interface. At the top, there's a header bar with the Hostinger logo, navigation links for Home, Websites, Hosting, Emails, Domains, VPS, SSL, and Billing, and a user profile icon. Below the header is a search bar and a dropdown menu for the website name, currently set to "kasimadalan.pe.hu". The main dashboard area has a title "Dashboard" and a subtitle "- Hosting - kasimadalan.pe.hu". It displays several service status cards: "Single Web Hosting" (Active), "Domain" (Manage), "Hostinger Free Email" (Active), "Daily backups" (Disabled), "File manager" (Manage), "Databases" (Manage), "Auto installer" (Manage), "Loading time" (Run page speed test), and a "Safe" status card indicating "No malware found" with a "See malware scanner" link. A large green checkmark icon with the text "Your website is running smoothly" and "No issues were found" is prominently displayed at the bottom right. On the left side, a sidebar lists various management options: Dashboard (selected), Hosting, Performance, Security, Emails, Domains, Website, and Files.

Veritabanı Oluşturma

Management



- Hosting - kasimadalan.pe.hu - Databases - Management

+ Create a New MySQL Database And Database User

MySQL database name

u139539474_

db

MySQL username

u139539474_

admin

Password *

123456Ab

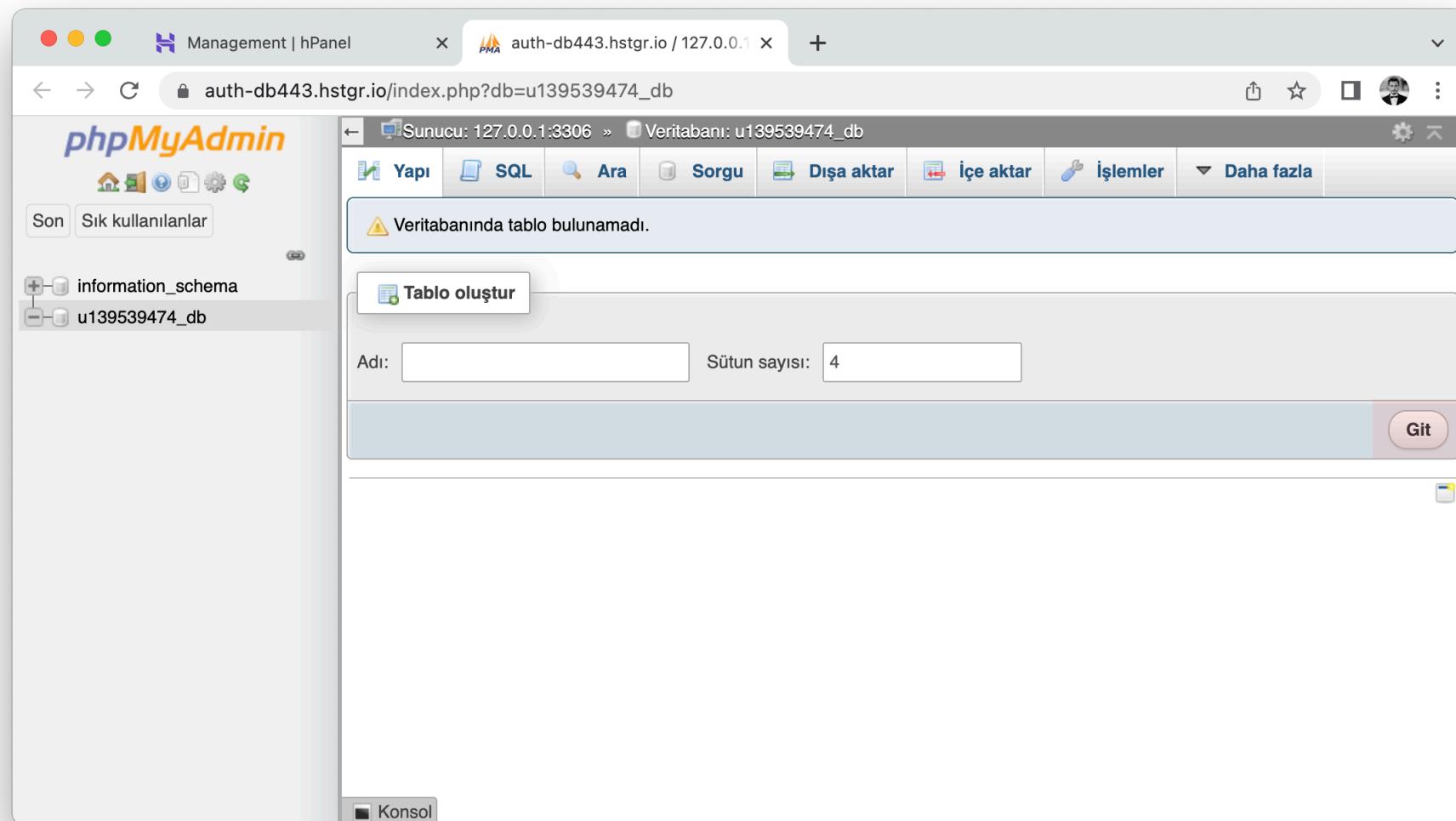


✓ Create

Veritabanı Oluşturma

List of Current MySQL Databases And Users					
MySQL Database	MySQL User	Created at	Website		
u139539474_db 1 MB	u139539474_admin	2023-02-22	kasimadalan.pe.hu	Enter phpMyAdmin	⋮
u139539474_movie 4 MB	u139539474_root	2020-05-28	+ Assign	Enter phpMyAdmin	⋮

Veritabanı Oluşturma



Veritabanı Üzerinde Tablo Oluşturma

Tablo oluştur

Adı: kisiler Sütun sayısı: 3

Git

Gözat **Yapı** **SQL** **Ara** **Ekle** **Dışa aktar** **İçe aktar** **İşlemler** **Tetikleyiciler**

Tablo adı: kisiler Ekle 1 sütun(lar) **Git**

Adı	Türü	Uzunluk/Değerler	Varsayılan	Karşılaştırma	Öznitelikler	Boş Index	A.I
kisi_id	INT		Yok			PRIMARY	<input checked="" type="checkbox"/> PRIMARY
kisi_ad	VARCHAR	100	Yok			---	<input type="checkbox"/>
kisi_tel	VARCHAR	100	Yok			---	<input type="checkbox"/>

Yapı

Tablo açıklamaları: Karşılaştırma: Depolama Motoru: InnoDB

Bölümleyen: (İfade veya sütun listesi)

Bölümler:

SQL Önizle **Kaydet**

Veritabanı Üzerinde Tablo Oluşturma

The screenshot shows the MySQL Workbench interface with the 'SQL' tab selected. A table named 'kisi' is being created with the following structure:

#	Adı	Türü	Karşılaştırma	Öznitelikler	Boş	Varsayılan	Açıklamalar	Ekstra	Eylem
<input type="checkbox"/>	1 kisi_id	int(11)			Hayır	Yok		AUTO_INCREMENT	Değiştir Kaldır Daha fazla
<input type="checkbox"/>	2 kisi_ad	varchar(100)	utf8mb4_unicode_ci		Hayır	Yok			Değiştir Kaldır Daha fazla
<input type="checkbox"/>	3 kisi_tel	varchar(100)	utf8mb4_unicode_ci		Hayır	Yok			Değiştir Kaldır Daha fazla

Veritabanı Üzerinde Tablo Oluşturma

The screenshot shows the MySQL Workbench interface for creating a new table. The top navigation bar includes buttons for Gözat (View), Yapı (Structure), SQL, Ara (Search), Ekle (Add), Dışa aktar (Export), İçe aktar (Import), İşlemler (Operations), and Tetikleyiciler (Triggers). The main area is titled 'Sütun' (Column) and 'Türü' (Type). A table structure is being defined:

	Sütun	Türü	İşlev	Boş	Değer
kisi_id	int(11)				
kisi_ad	varchar(100)				Ahmet
kisi_tel	varchar(100)				1111

At the bottom, there are buttons for 'Yeni satır olarak ekle' (Add new row), 've ondan sonra' (After it), 'Önceki sayfaya geri dön' (Go back to previous page), 'SQL Önizle' (Preview SQL), 'Sıfırla' (Clear), and 'Git' (Go).

Veritabanı Üzerinde Tablo Oluşturma

Gözat **Yapı** SQL Ara Ekle Dışa aktar İçe aktar İşlemler

✓ Gösterilen satır 0 - 1 (toplam 2, Sorgu 0.0013 saniye sürdü.)

```
SELECT * FROM `kisiler`
```

Profil çıkart [Satır içi düzenle] [Düzenle] [SQL'i açıkla] [PHP kodu oluştur] [Yenile]

```
UPDATE `kisiler` SET `kisi_tel` = '2222' WHERE `kisiler`.`kisi_id` = 2;
```

[Satır içi düzenle] [Düzenle] [PHP kodu oluştur]

Tümünü göster | Satır sayısı: 25 Satırları süz: Bu tabloda ara Anahtara göre sıra

+ Seçenekler

	Düzenle	Kopyala	Sil	kisi_id	kisi_ad	kisi_tel
<input type="checkbox"/>				1	Ahmet	1111
<input type="checkbox"/>				2	Zeynep	2222

↑ Tümünü işaretle **Seçilileri:** Düzenle Kopyala Sil Dışa aktar

Veritabanı Üzerinde İşlemler

Gözat Yapı SQL Ara Ekle Dışa aktar İçe aktar İşlemler Tetikleyiciler

u139539474_db.kisiler tablosu üzerinde SQL sorgusunu/sorgularını çalıştır:

```
1 SELECT * FROM `kisiler`
```

kisi_id	kisi_ad	kisi_tel
---------	---------	----------

SELECT *

Parametreleri bağla

Sınırlayıcı Bu sorguyu burada tekrar göster Sorgu kutusunu tut Tamamlandığında geri döndür Dış anahtar denetlemelerini etkinleştir

Veri Kaydı - Insert

```
INSERT INTO `kisiler`(`kisi_ad`, `kisi_tel`) VALUES ('Beyza', '9999')
```

kisi_id	kisi_ad	kisi_tel
1	Ahmet	1111
2	Zeynep	2222
4	Beyza	9999

Veri Güncelleme - Update

```
UPDATE `kisiler` SET `kisi_ad`='Yeni Beyza', `kisi_tel`='0000' WHERE kisi_id = 4
```

kisi_id	kisi_ad	kisi_tel
1	Ahmet	1111
2	Zeynep	2222
4	Yeni Beyza	0000

Veri Silme - Delete

```
DELETE FROM `kisiler` WHERE kisi_id = 4
```

kisi_id	kisi_ad	kisi_tel
1	Ahmet	1111
2	Zeynep	2222

Select Sorgusu

- Tüm verileri getir

```
SELECT * FROM `kisiler`
```

- İstenilen alanları getir

```
SELECT kisi_ad,kisi_tel FROM `kisiler`
```

WHERE Şartı

- Eşitlik

```
SELECT * FROM `kisiler` WHERE kisi_ad = 'Ahmet'
```

- Kayıt kontrol

```
SELECT COUNT(*) as sonuc FROM `kisiler` WHERE kisi_ad = 'Ahmet'
```

ORDER BY - Sıralama

- Kişiler tablosundaki kisi_adlarına göre artan şekilde getirir.

```
SELECT * FROM `kisiler` ORDER BY kisi_ad ASC
```

- Kişiler tablosundaki kisi_adlarına göre azalan şekilde getirir.

```
SELECT * FROM `kisiler` ORDER BY kisi_ad DESC
```

- Varsayılsan sıralama ASC dir.

LIKE - Benzerlik Arama

- Aratılan ifade ilgili alan içeriyor mu şeklinde arama yapılır.

```
SELECT * FROM `kisiler` WHERE kisi_ad LIKE '%e%'
```

LIMIT - Sınırlı Sayıda Veri Getirme

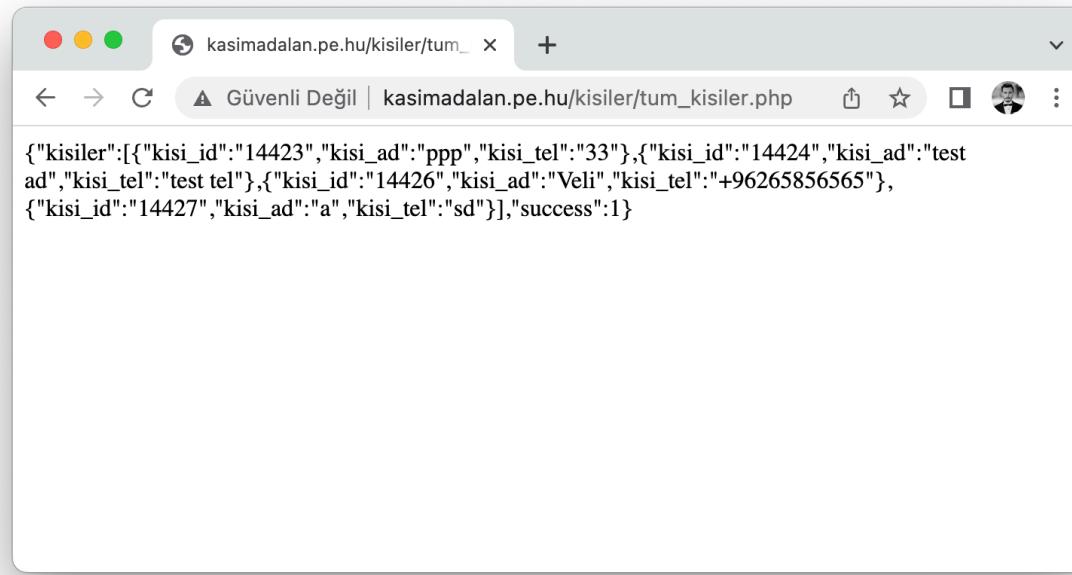
```
SELECT * FROM `kisiler` LIMIT 1
```

```
SELECT * FROM `kisiler` WHERE kisi_ad = 'Ahmet' LIMIT 1
```

```
SELECT * FROM `kisiler` ORDER BY RAND() LIMIT 1
```

API - Webservis Oluşturma

- Webservisler internet üzerindeki veritabanından verileri almamızı sağlayan metodlardır.
- Url olarak çalışırlar.
- İstedığınız dilde webservis oluşturabilirsiniz.



Veritabanı Bağlantı Bilgileri Dosyası Oluşturma

- Bağlanmak istediğimiz veritabanı ile ilgi bilgileri bir dosya içinde oluşturmalıyız.
- Bu bilgiler ile veritabanı bağlantısı oluşturabiliriz.
-

List of Current MySQL Databases And Users

MySQL Database	MySQL User	Created at	Website	
u139539474_db 1 MB	u139539474_admin	2023-02-22	kasimadalan.pe.hu	Enter phpMyAdmin ⋮
u139539474_movie 4 MB	u139539474_root	2020-05-28	+ Assign	Enter phpMyAdmin ⋮

db_config.php — Edited

```
<?php
define('DB_USER', "u139539474_admin");
define('DB_PASSWORD', "123456Ab");
define('DB_DATABASE', "u139539474_db");
define('DB_SERVER', "localhost");
?>
```

Line: 9 Col: 1

```
<?php
$response = array();

if (isset($_POST['kisi_id'])) {
    $kisi_id = $_POST['kisi_id'];

    //DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
    require_once __DIR__ . '/db_config.php';

    // Bağlantı oluşturuluyor.
    $baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

    // Bağlantı kontrolü yapılır.
    if (!$baglanti) {
        die("Hatalı bağlantı : " . mysqli_connect_error());
    }
    $sqlsorgu = "DELETE FROM kisiler WHERE kisiler.kisi_id = $kisi_id";

    if (mysqli_query($baglanti, $sqlsorgu)) {

        $response["success"] = 1;
        $response["message"] = "successfully ";
        echo json_encode($response);
    } else {

        $response["success"] = 0;
        $response["message"] = "No product found";
        echo json_encode($response);
    }
    //bağlantı koparılır.
    mysqli_close($baglanti);
} else {
    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";
    echo json_encode($response);
}
?>
```

{ -

"success":0,

"message":"Required field(s) is missing"

}

```
<?php
$response = array();

if (isset($_POST['kisi_ad']) && isset($_POST['kisi_tel'])) {
    $kisi_ad = $_POST['kisi_ad'];
    $kisi_tel = $_POST['kisi_tel'];

    //DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
    require_once __DIR__ . '/db_config.php';

    // Bağlantı oluşturuluyor.
    $baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

    // Bağlantı kontrolü yapılır.
    if (!$baglanti) {
        die("Hatalı bağlantı : " . mysqli_connect_error());
    }

    $sqlsorgu = "INSERT INTO kisiler (kisi_ad,kisi_tel) VALUES ('$kisi_ad','$kisi_tel')";

    if (mysqli_query($baglanti, $sqlsorgu)) {
        $response["success"] = 1;
        $response["message"] = "successfully ";
        echo json_encode($response);
    } else {
        $response["success"] = 0;
        $response["message"] = "No product found";
        echo json_encode($response);
    }
    //bağlantı koparılır.
    mysqli_close($baglanti);
} else {
    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";
    echo json_encode($response);
}
?>
```

{ "success":0,
"message":"Required field(s) is missing" }

```
<?php

$response = array();

if (isset($_POST['kisi_id']) && isset($_POST['kisi_ad']) && isset($_POST['kisi_tel'])) {
    $kisi_id = $_POST['kisi_id'];
    $kisi_ad = $_POST['kisi_ad'];
    $kisi_tel = $_POST['kisi_tel'];

    //DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
    require_once __DIR__ . '/db_config.php';

    // Bağlantı oluşturuluyor.
    $baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

    // Bağlantı kontrolü yapılır.
    if (!$baglanti) {
        die("Hatalı bağlantı : " . mysqli_connect_error());
    }

    $sqlsorgu = "UPDATE kisiler SET kisiler.kisi_ad = '$kisi_ad',kisiler.kisi_tel = '$kisi_tel' WHERE
                kisiler.kisi_id = $kisi_id  ";

    if (mysqli_query($baglanti, $sqlsorgu)) {
        $response["success"] = 1;
        $response["message"] = "successfully ";
        echo json_encode($response);
    } else {
        $response["success"] = 0;
        $response["message"] = "No product found";
        echo json_encode($response);
    }
    //bağlantı koparılır.
    mysqli_close($baglanti);
} else {
    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";
    echo json_encode($response);
}
?>
```

{ -

 "success":0,

 "message":"Required field(s) is missing"

}

```

<?php
// array for JSON response
$response = array();

//DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
require_once __DIR__ . '/db_config.php';

// Bağlantı oluşturuluyor.
$baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

// Bağlantı kontrolü yapılır.
if (!$baglanti) {
    die("Hatalı bağlantı : " . mysqli_connect_error());
}

$sqlsorgu = "SELECT * FROM kisiler";
$result = mysqli_query($baglanti, $sqlsorgu);

// result kontrolü yap
if (mysqli_num_rows($result) > 0) {

    $response["kisiler"] = array();

    while ($row = mysqli_fetch_assoc($result)) {
        // temp user array
        $kisiler = array();

        $kisiler["kisi_id"] = $row["kisi_id"];
        $kisiler["kisi_ad"] = $row["kisi_ad"];
        $kisiler["kisi_tel"] = $row["kisi_tel"];

        // push single product into final response array
        array_push($response["kisiler"], $kisiler);
    }
    // success
    $response["success"] = 1;
}

// echoing JSON response
echo json_encode($response);

```



```

{
    "kisiler": [
        {
            "kisi_id": "376",
            "kisi_ad": "Ahmet",
            "kisi_tel": "5348564412"
        },
        {
            "kisi_id": "379",
            "kisi_ad": "Zeynep",
            "kisi_tel": "05375483214"
        },
        {
            "kisi_id": "380",
            "kisi_ad": "Mehmet",
            "kisi_tel": "05375483215"
        },
        {
            "kisi_id": "381",
            "kisi_ad": "Ayşe",
            "kisi_tel": "05375483216"
        },
        {
            "kisi_id": "382",
            "kisi_ad": "Mustafa",
            "kisi_tel": "05375483217"
        }
    ],
    "success": 1
}
//bağlantı koparılır.
mysqli_close($baglanti);
?>

```

```
<?php
```

```
$response = array();

if (isset($_POST['kisi_ad'])) {
    $kisi_ad = $_POST['kisi_ad'];

    //DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
    require_once __DIR__ . '/db_config.php';

    // Bağlantı oluşturuluyor.
    $baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

    // Bağlantı kontrolü yapılır.
    if (!$baglanti) {
        die("Hatalı bağlantı : " . mysqli_connect_error());
    }

    $sqlsorgu = "SELECT * FROM kisiler WHERE kisiler.kisi_ad like '%$kisi_ad%'";
    $result = mysqli_query($baglanti, $sqlsorgu);

    if (mysqli_num_rows($result) > 0) {

        $response["kisiler"] = array();

        while ($row = mysqli_fetch_assoc($result)) {

            $kisiler = array();

            $kisiler["kisi_id"] = $row["kisi_id"];
            $kisiler["kisi_ad"] = $row["kisi_ad"];
            $kisiler["kisi_tel"] = $row["kisi_tel"];

            array_push($response["kisiler"], $kisiler);
        }

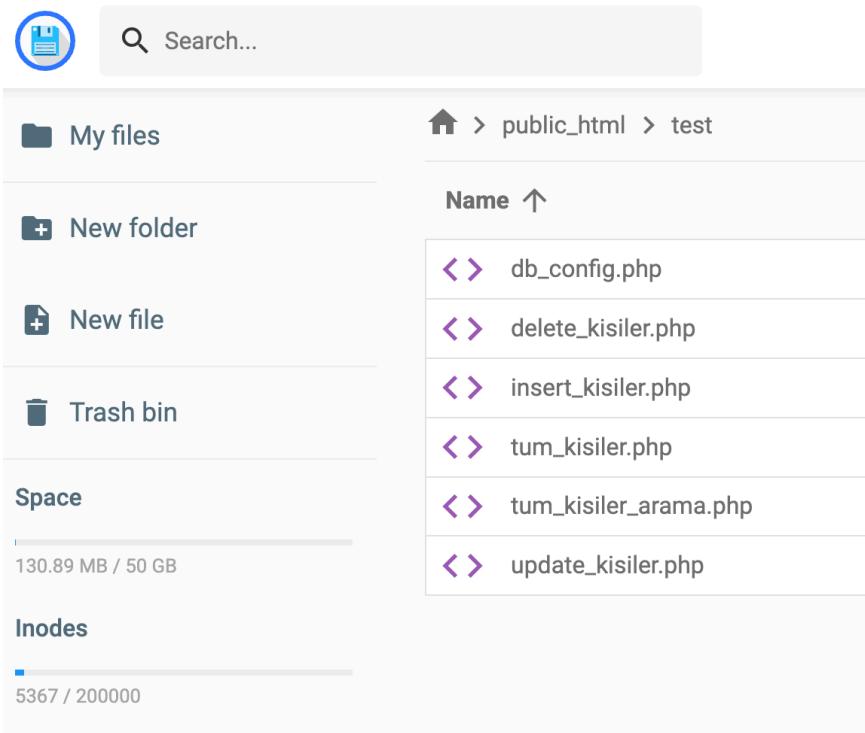
        $response["success"] = 1;
        echo json_encode($response);
    }
    //bağlantı koparılır.
    mysqli_close($baglanti);
}
```

```
{ [
    "kisiler": [
        {
            "kisi_id": "376",
            "kisi_ad": "Ahmet",
            "kisi_tel": "5348564412"
        },
        {
            "kisi_id": "379",
            "kisi_ad": "Zeynep",
            "kisi_tel": "05375483214"
        },
        {
            "+"
        },
        {
            "+"
        },
        {
            "+"
        },
        {
            "+"
        },
        {
            "+"
        }
    ],
    "success": 1
}
```

```
} else {
    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";
    echo json_encode($response);
}
?>
```

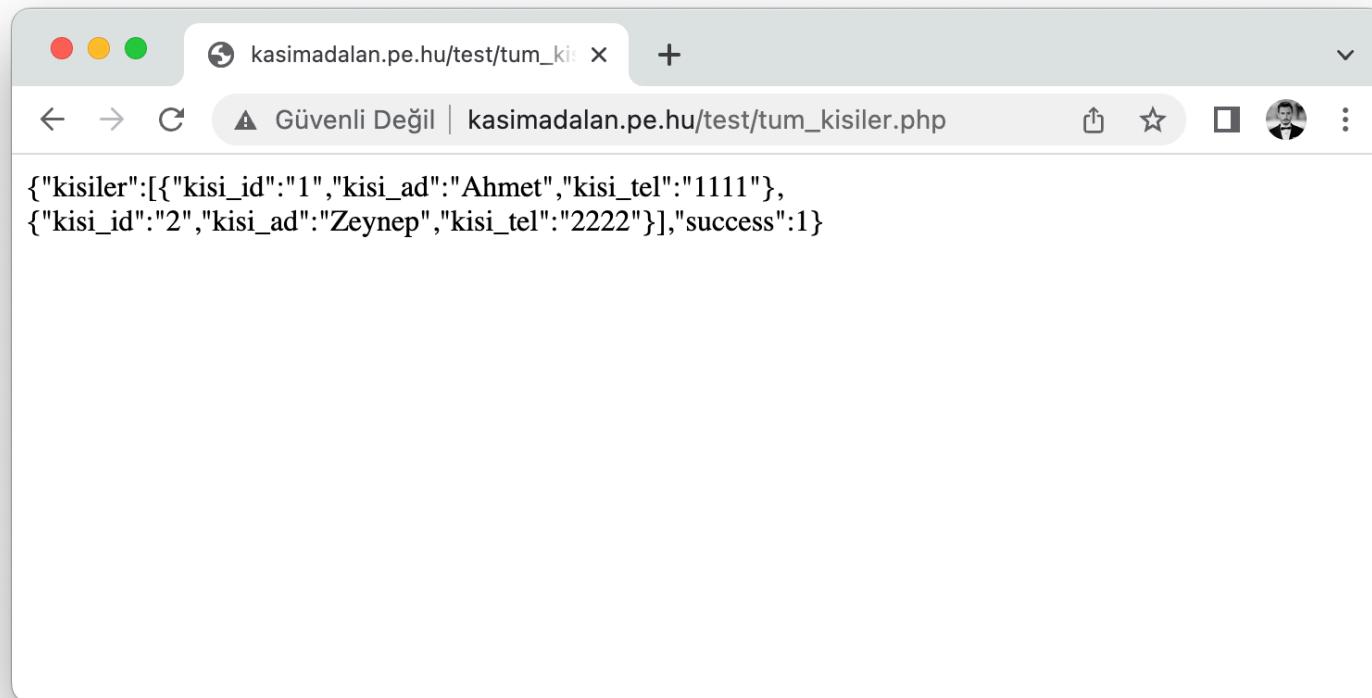
Webservis Hosting İşlemi

- Webservisleri çalıştırılmak için hosting hizmetinin dosya sistemine yüklenmelidir.



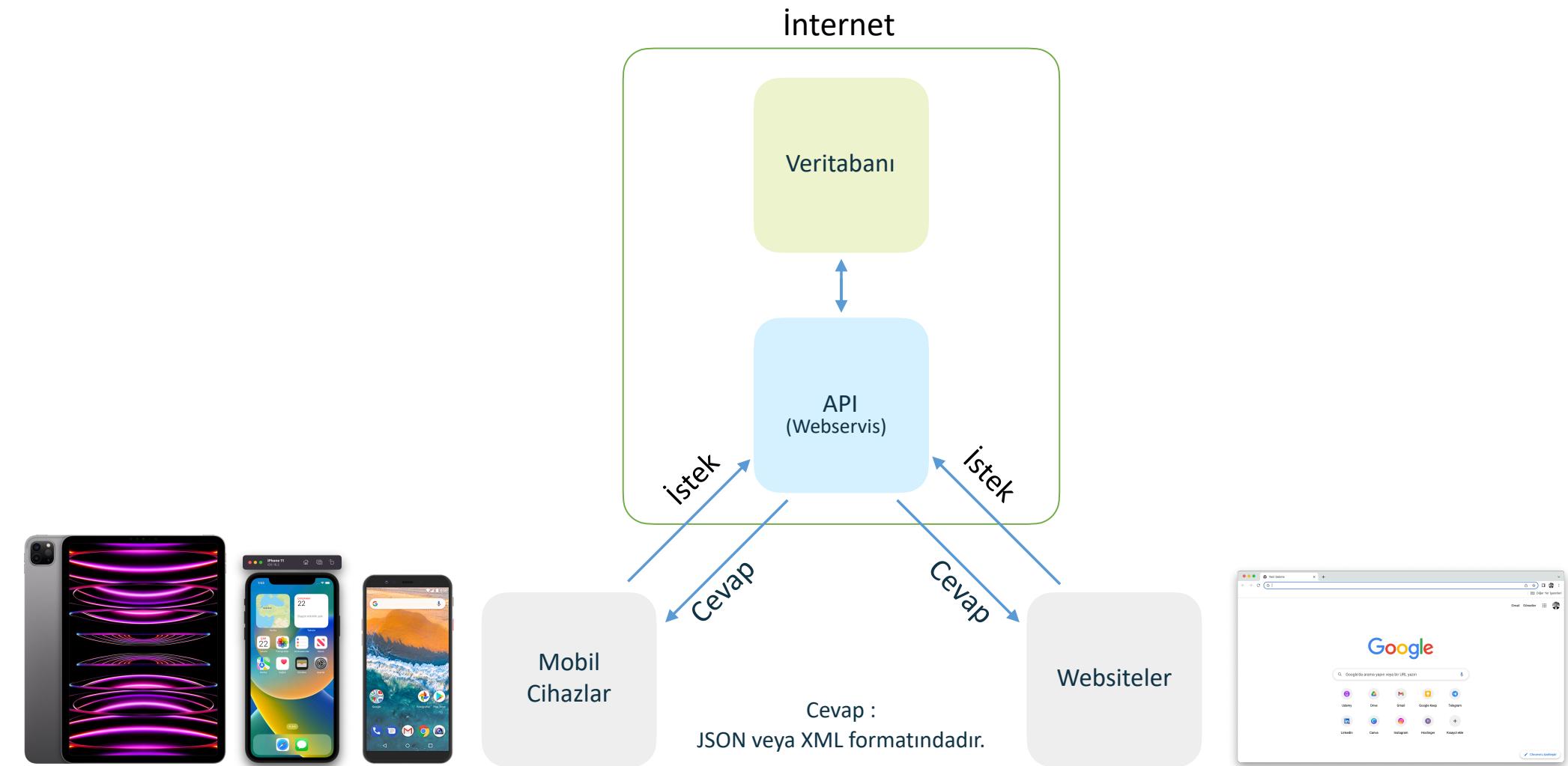
Webservisin Çalıştırılması

- Webservisin urlsini http isteği çalıştırabilen platformada kullanabiliriz.
- http://kasimadalan.pe.hu/test/tum_kisiler.php



JSON Parse İşlemi

RESTful Mimarisi



JSON VERİ MODELİ

```
{ [ parantez Json array ( dizidir yani listedir. )  
    "kisiler": [ [ ]  
{ parantez Json object ( nesnesidir ) { [ ]  
    "kisi_id": "1",  
    "kisi_ad": "Ahmet",  
    "kisi_tel": "12312312"  
},  
{ [ ]  
    "kisi_id": "2",  
    "kisi_ad": "Mehmet",  
    "kisi_tel": "912318212"  
}  
]  
}
```

Json nesnesi içeriği key ve value şeklinde oluşturulur.

JSON Cevabı Alma

JSON Cevabı İçin Codable Parse İşlemi

Codable JSON Parse

- Klasik parse işleminden daha pratik olan kodlama ağırlıklı bir yapıdır.
- Json örneğini nesne tabanlı sınıflara dönüştürüp pratik bir kullanım sağlar.
- Bize direk json cevabını liste olarak verir.
- Burada en önemli nokta gelen bilginin türüne uygun değişkenler oluşturulmalıdır.

```
{ [
  "kisiler": [
    {
      "kisi_id": "376",
      "kisi_ad": "Ahmet",
      "kisi_tel": "5348564412"
    },
    {
      "kisi_id": "379",
      "kisi_ad": "Zeynep",
      "kisi_tel": "05375483214"
    },
    { [+] },
    { [+] },
    { [+] },
    { [+] },
    { [+] }
  ],
  "success": 1
}]
```

```
class Kisiler:Codable {
  var kisi_id:String?
  var kisi_ad:String?
  var kisi_tel:String?

  init() {

  }

  init(kisi_id:String,kisi_ad:String,kisi_tel:String) {
    self.kisi_id = kisi_id
    self.kisi_ad = kisi_ad
    self.kisi_tel = kisi_tel
  }
}
```

```
class KisilerCevap:Codable {
  var kisiler:[Kisiler]?
  var success:Int?
}
```

```
let url = URL(string: "http://kasimdalanan.pe.hu/kisiler/tum_kisiler.php")!
URLSession.shared.dataTask(with: url){ data,response,error in
    let cevap = try JSONDecoder().decode(KisilerCevap.self, from: data!)
    if let kisiler = cevap.kisiler {
        for kisi in kisiler {
            print("*****")
            print("Kişi id : \(kisi.kisi_id!)")
            print("Kişi ad : \(kisi.kisi_ad!)")
            print("Kişi tel : \(kisi.kisi_tel!)")
        }
    }
    if let success = cevap.success {
        print("*****")
        print("Başarı : \(success)")
    }
}catch{
    print(error.localizedDescription)
}.resume()
```

```

    {
      "filmler": [
        {
          "film_id": "1",
          "film_ad": "Interstellar",
          "film_yil": "2015",
          "film_resim": "interstellar.png",
          "kategori": {
            "kategori_id": "4",
            "kategori_ad": "Bilim Kurgu"
          },
          "yonetmen": {
            "yonetmen_id": "1",
            "yonetmen_ad": "Christopher Nolan"
          }
        },
        {⊕},
        {⊕},
        {⊕},
        {⊕},
        {⊕}
      ],
      "success": 1
    }

    class FilmlerCevap:Codable {
      var filmler:[Filmler]?
      var success:Int?
    }

    class Kategoriler:Codable {
      var kategori_id:String?
      var kategori_ad:String?
    }

    class Yonetmenler:Codable {
      var yonetmen_id:String?
      var yonetmen_ad:String?
    }

    class Filmler:Codable {
      var film_id:String?
      var film_ad:String?
      var film_yil:String?
      var film_resim:String?
      var kategori:Kategoriler?
      var yonetmen:Yonetmenler?

      init() {
      }

      init(film_id:String,film_ad:String,film_yil:String,film_resim:String,kategori:Kategoriler,yonetmen:Yonetmenler) {
        self.film_id = film_id
        self.film_ad = film_ad
        self.film_yil = film_yil
        self.film_resim = film_resim
        self.kategori = kategori
        self.yonetmen = yonetmen
      }
    }
  }

```

```
let url = URL(string: "http://kasimadalan.pe.hu/filmller/tum_filmller.php")!

URLSession.shared.dataTask(with: url){ data,response,error in

    {
        "filmller": [
            {
                "film_id": "1",
                "film_ad": "Interstellar",
                "film_yil": "2015",
                "film_resim": "interstellar.png",
                "kategori": {
                    "kategori_id": "4",
                    "kategori_ad": "Bilim Kurgu"
                },
                "yonetmen": {
                    "yonetmen_id": "1",
                    "yonetmen_ad": "Christopher Nolan"
                }
            },
            { },
            { },
            { },
            { },
            { }
        ],
        "success": 1
    }

    URLSession.shared.dataTask(with: url){ data,response,error in
        do{
            let cevap = try JSONDecoder().decode(FilmllerCevap.self, from: data!)

            if let filmller = cevap.filmller {
                for film in filmller {
                    print("*****")
                    print("Film id : \(film.film_id!)")
                    print("Film ad : \(film.film_ad!)")
                    print("Film yil : \(film.film_yil!)")
                    print("Film resim : \(film.film_resim!)")
                    print("Kategori id : \(film.kategori!.kategori_id!)")
                    print("Kategori ad : \(film.kategori!.kategori_ad!)")
                    print("Yonetmen id : \(film.yonetmen!.yonetmen_id!)")
                    print("Yonetmen ad : \(film.yonetmen!.yonetmen_ad!)")
                }
            }

            if let success = cevap.success {
                print("*****")
                print("Başarı : \(success)")
            }
        }catch{print(error.localizedDescription)}
    }.resume()
}
```

ALAMofire

Elegant Networking in Swift

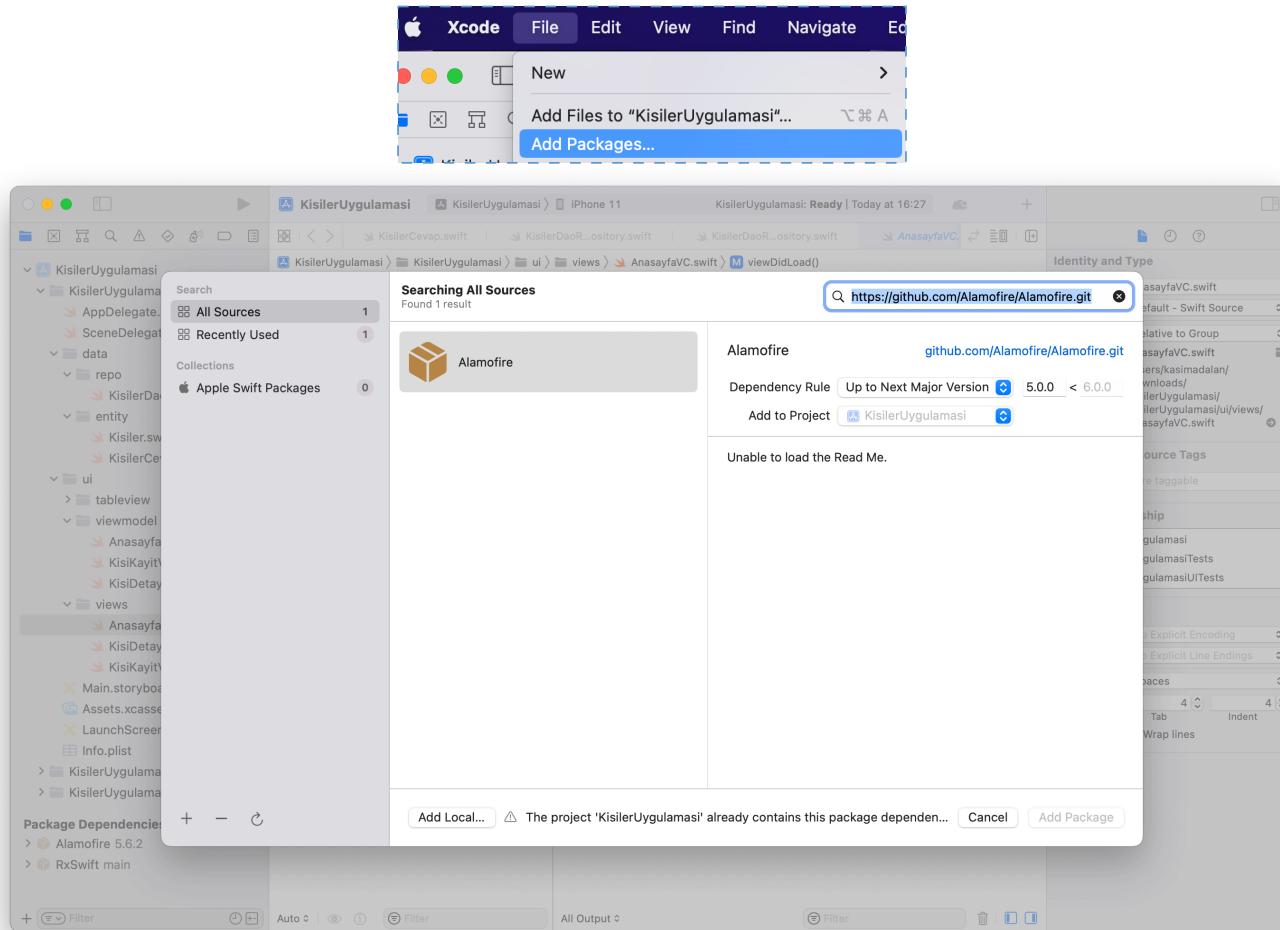
internet izni

The screenshot shows the Xcode Property List Editor with the following details:

File Path: KisilerUygulamasi > KisilerUygulamasi > Info.plist > No Selection

Key	Type	Value
Information Property List	Dictionary	(2 items)
App Transport Security Settings	Dictionary	(1 item)
Allow Arbitrary Loads	Boolean	YES
Application Scene Manifest	Dictionary	(2 items)

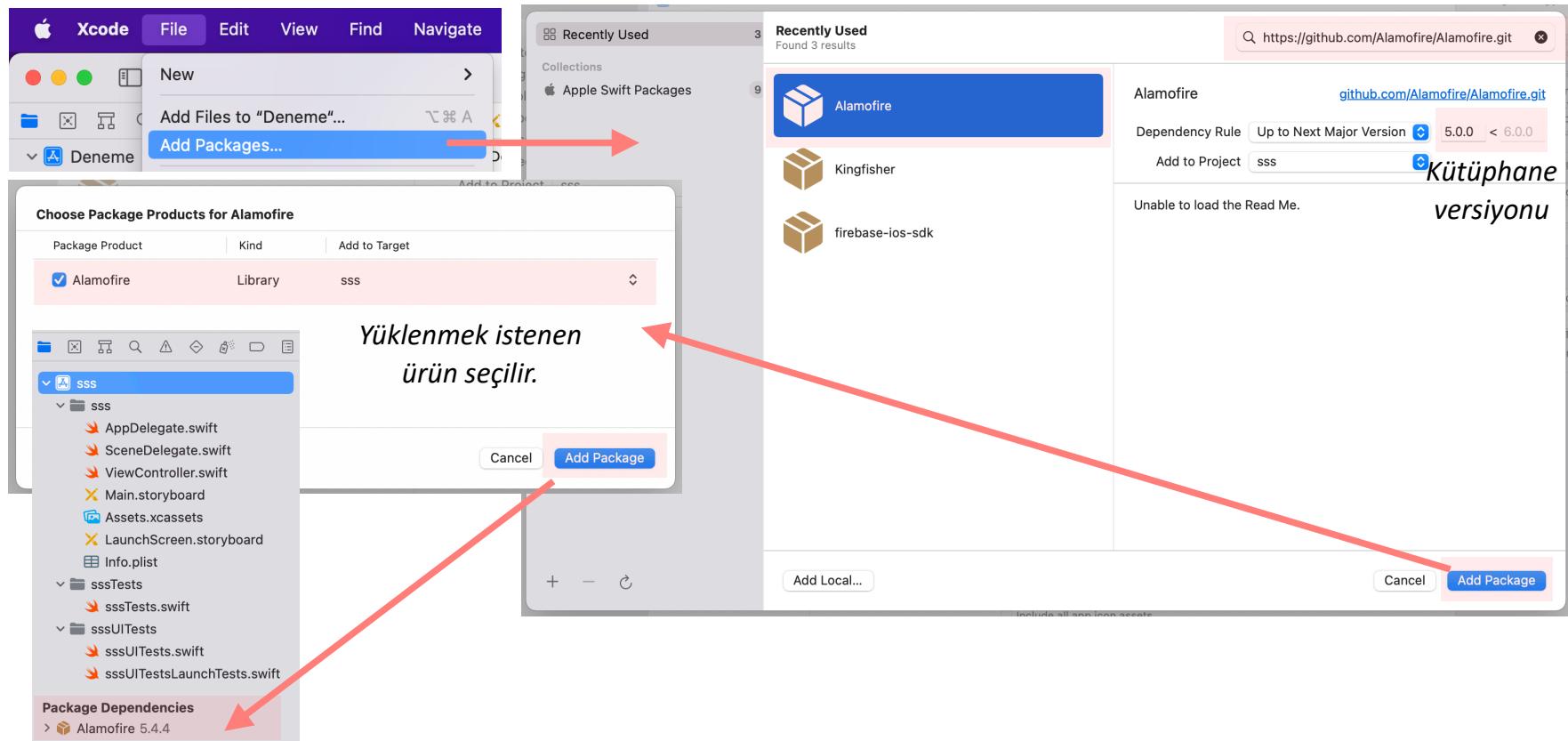
Alamofire Kurulum



<https://github.com/Alamofire/Alamofire.git>

Kurulum

Kütüphane urlsi aratılır



GET

İstek Yapma

```
AF.request("http://kasimadalan.pe.hu/kisiler/tum_kisiler.php", method: .get).response { response in
    if let data = response.data {
        do{
            let cevap = try JSONDecoder().decode(KisilerCevap.self, from: data)

            if let gelenListe = cevap.kisiler {
                for k in gelenListe {
                    print("*****")
                    print("Kişi id : \(k.kisi_id!)")
                    print("Kişi ad : \(k.kisi_ad!)")
                    print("Kişi tel : \(k.kisi_tel!)")
                }
            }
        }catch{
            print(error.localizedDescription)
        }
    }
}
```

POST

Post Request – Veri Gönderme

```
let params:Parameters = ["kisi_ad":"Ahmet","kisi_tel":"0512313421"]

AF.request("http://kasimadalan.pe.hu/kisiler/insert_kisiler.php",
    method: .post,parameters: params).response { response in

    if let data = response.data {
        do{

            let cevap = try JSONSerialization.jsonObject(with: data)
            print(cevap)

        }catch{
            print(error.localizedDescription)
        }
    }
}
```

Alamofire Örnekleri

Tüm Verilerin Alınması

tum_kisiler.php

```
<?php
// array for JSON response
$response = array();

//DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
require_once __DIR__ . '/db_config.php';

// Bağlantı oluşturuluyor.
$baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

// Bağlantı kontrolü yapılır.
if (!$baglanti) {
    die("Hatalı bağlantı : " . mysqli_connect_error());
}

$sqlsorgu = "SELECT * FROM kisiler";
$result = mysqli_query($baglanti, $sqlsorgu);

// result kontrolü yap
if (mysqli_num_rows($result) > 0) {

    $response["kisiler"] = array();

    while ($row = mysqli_fetch_assoc($result)) {
        // temp user array
        $kisiler = array();

        $kisiler["kisi_id"] = $row["kisi_id"];
        $kisiler["kisi_ad"] = $row["kisi_ad"];
        $kisiler["kisi_tel"] = $row["kisi_tel"];

        // push single product into final response array
        array_push($response["kisiler"], $kisiler);
    }
    // success
    $response["success"] = 1;

    // echoing JSON response
    echo json_encode($response);
} else {
    // no products found
    $response["success"] = 0;
    $response["message"] = "No data found";

    // echo no users JSON
    echo json_encode($response);
}
//bağlantı koparılır.
mysqli_close($baglanti);
?>
```

Çalıştırmak istediğimiz PHP kod



Swift Tarafı Kodlama – Codable Parse

```
AF.request("http://kasimadalan.pe.hu/kisiler/tum_kisiler.php", method: .get).response { response in
    if let data = response.data {
        do{
            let cevap = try JSONDecoder().decode(KisilerCevap.self, from: data)

            if let gelenListe = cevap.kisiler {
                for k in gelenListe {
                    print("*****")
                    print("Kişi id : \(k.kisi_id!)")
                    print("Kişi ad : \(k.kisi_ad!)")
                    print("Kişi tel : \(k.kisi_tel!)")
                }
            }
        }catch{
            print(error.localizedDescription)
        }
    }
}
```

```
class KisilerCevap:Codable {
    var kisiler:[Kisiler]?
    var success:Int?
}

class Kisiler:Codable {
    var kisi_id:String?
    var kisi_ad:String?
    var kisi_tel:String?

    init() {

    }

    init(kisi_id:String,kisi_ad:String,kisi_tel:String) {
        self.kisi_id = kisi_id
        self.kisi_ad = kisi_ad
        self.kisi_tel = kisi_tel
    }
}
```

Arama işlemi

```
<?php

$response = array();

if (isset($_POST['kisi_ad'])) {
    $kisi_ad = $_POST['kisi_ad'];

    //DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
    require_once __DIR__ . '/db_config.php';

    // Bağlantı oluşturuluyor.
    $baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

    // Bağlantı kontrolü yapılır.
    if (!$baglanti) {
        die("Hatalı bağlantı : " . mysqli_connect_error());
    }

    $sqlsorgu = "SELECT * FROM kisiler WHERE kisiler.kisi_ad like '%$kisi_ad%'";
    $result = mysqli_query($baglanti, $sqlsorgu);

    if (mysqli_num_rows($result) > 0) {

        $response["kisiler"] = array();

        while ($row = mysqli_fetch_assoc($result)) {

            $kisiler = array();

            $kisiler["kisi_id"] = $row["kisi_id"];
            $kisiler["kisi_ad"] = $row["kisi_ad"];
            $kisiler["kisi_tel"] = $row["kisi_tel"];

            array_push($response["kisiler"], $kisiler);
        }

        $response["success"] = 1;
        echo json_encode($response);
    }
    //bağlanti koparılır.
    mysqli_close($baglanti);
}
```

tum_kisiler_arama.php

Çalıştırmak
istediğimiz PHP
kod

```
} else {
    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";
    echo json_encode($response);
}

?>
```

Swift Tarafı Kodlama – Codable Parse

```
let params:Parameters = ["kisi_ad":"a"]

AF.request("http://kasimadalan.pe.hu/kisiler/tum_kisiler_arama.php",method: .post,parameters: params).response {
    response in
    if let data = response.data {
        do{
            let cevap = try JSONDecoder().decode(KisilerCevap.self, from: data)

            if let gelenListe = cevap.kisiler {
                for k in gelenListe {
                    print("*****")
                    print("Kişi id : \(k.kisi_id!)")
                    print("Kişi ad : \(k.kisi_ad!)")
                    print("Kişi tel : \(k.kisi_tel!)")
                }
            }
        }catch{
            print(error.localizedDescription)
        }
    }
}
```

Silme İşlemi

```
<?php
$response = array();

if (isset($_POST['kisi_id'])) {
    $kisi_id = $_POST['kisi_id'];

    //DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
    require_once __DIR__ . '/db_config.php';

    // Bağlanti oluşturuluyor.
    $baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

    // Bağlanti kontrolü yapılır.
    if (!$baglanti) {
        die("Hatalı bağlantı : " . mysqli_connect_error());
    }
    $sqlsorgu = "DELETE FROM kisiler WHERE kisiler.kisi_id = $kisi_id";

    if (mysqli_query($baglanti, $sqlsorgu)) {

        $response["success"] = 1;
        $response["message"] = "successfully ";
        echo json_encode($response);
    } else {

        $response["success"] = 0;
        $response["message"] = "No product found";
        echo json_encode($response);
    }
    //bağlantı koparılır.
    mysqli_close($baglanti);
} else {
    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";
    echo json_encode($response);
}
?>
```

delete_kisiler.php

Çalıştırmak istediğimiz PHP kod

Swift Tarafı Kodlama

```
let params:Parameters = ["kisi_id":10]

AF.request("http://kasimadalan.pe.hu/kisiler/delete_kisiler.php",
           method: .post,parameters: params).response { response in

    if let data = response.data {
        do{
            let cevap = try JSONSerialization.jsonObject(with: data)
            print(cevap)

        }catch{
            print(error.localizedDescription)
        }
    }
}
```

Kayıt İşlemi

```
<?php
$response = array();

if (isset($_POST['kisi_ad']) && isset($_POST['kisi_tel'])) {
    $kisi_ad = $_POST['kisi_ad'];
    $kisi_tel = $_POST['kisi_tel'];

    //DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
    require_once __DIR__ . '/db_config.php';

    // Bağlantı oluşturuluyor.
    $baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

    // Bağlantı kontrolü yapılır.
    if (!$baglanti) {
        die("Hatalı bağlantı : " . mysqli_connect_error());
    }

    $sqlsorgu = "INSERT INTO kisiler (kisi_ad,kisi_tel) VALUES ('$kisi_ad','$kisi_tel')";

    if (mysqli_query($baglanti, $sqlsorgu)) {
        $response["success"] = 1;
        $response["message"] = "successfully ";
        echo json_encode($response);
    } else {
        $response["success"] = 0;
        $response["message"] = "No product found";
        echo json_encode($response);
    }
    //bağlantı koparılır.
    mysqli_close($baglanti);
} else {
    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";
    echo json_encode($response);
}
?>
```

insert_kisiler.php

Çalıştırmak istediğimiz PHP kod

Swift Tarafı Kodlama

```
let params:Parameters = ["kisi_ad":"Ahmet","kisi_tel":"0512313421"]

AF.request("http://kasimadalan.pe.hu/kisiler/insert_kisiler.php",
    method: .post,parameters: params).response { response in

    if let data = response.data {
        do{
            let cevap = try JSONSerialization.jsonObject(with: data)
            print(cevap)

        }catch{
            print(error.localizedDescription)
        }
    }
}
```

Güncelleme İşlemi

```
<?php

$response = array();

if (isset($_POST['kisi_id']) && isset($_POST['kisi_ad']) && isset($_POST['kisi_tel'])) {
    $kisi_id = $_POST['kisi_id'];
    $kisi_ad = $_POST['kisi_ad'];
    $kisi_tel = $_POST['kisi_tel'];

    //DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
    require_once __DIR__ . '/db_config.php';

    // Bağlantı oluşturuluyor.
    $baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

    // Bağlantı kontrolü yapılır.
    if (!$baglanti) {
        die("Hatalı bağlantı : " . mysqli_connect_error());
    }

    $sqlsorgu = "UPDATE kisiler SET kisiler.kisi_ad = '$kisi_ad',kisiler.kisi_tel = '$kisi_tel' WHERE
                kisiler.kisi_id = $kisi_id ";

    if (mysqli_query($baglanti, $sqlsorgu)) {
        $response["success"] = 1;
        $response["message"] = "successfully ";
        echo json_encode($response);
    } else {
        $response["success"] = 0;
        $response["message"] = "No product found";
        echo json_encode($response);
    }
    //bağlantı koparılır.
    mysqli_close($baglanti);
} else {
    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";
    echo json_encode($response);
}
?>
```

update_kisiler.php

Çalıştırmak istediğimiz PHP kod

Swift Tarafı Kodlama

```
let params:Parameters = ["kisi_id":10,"kisi_ad":"Yeni Ahmet","kisi_tel":"9999999"]  
  
AF.request("http://kasimadalan.pe.hu/kisiler/update_kisiler.php",  
          method: .post, parameters: params).response { response in  
  
    if let data = response.data {  
        do{  
            let cevap = try JSONSerialization.jsonObject(with: data)  
            print(cevap)  
  
        }catch{  
            print(error.localizedDescription)  
        }  
    }  
}
```

Kingfisher ile Resim İşlemi

Kingfisher ile Resim İşlemi

- İnternet üzerindeki resimleri uygulama içinde görüntülemek için harici bir kütüphanedir.
- İnternet tabanlı resimleri yüklenmede performans olarak gayet iyidir.
- Kullanımı kolaydır ve bir çok özellik bizlere sunmaktadır.



<https://github.com/onevcat/Kingfisher>

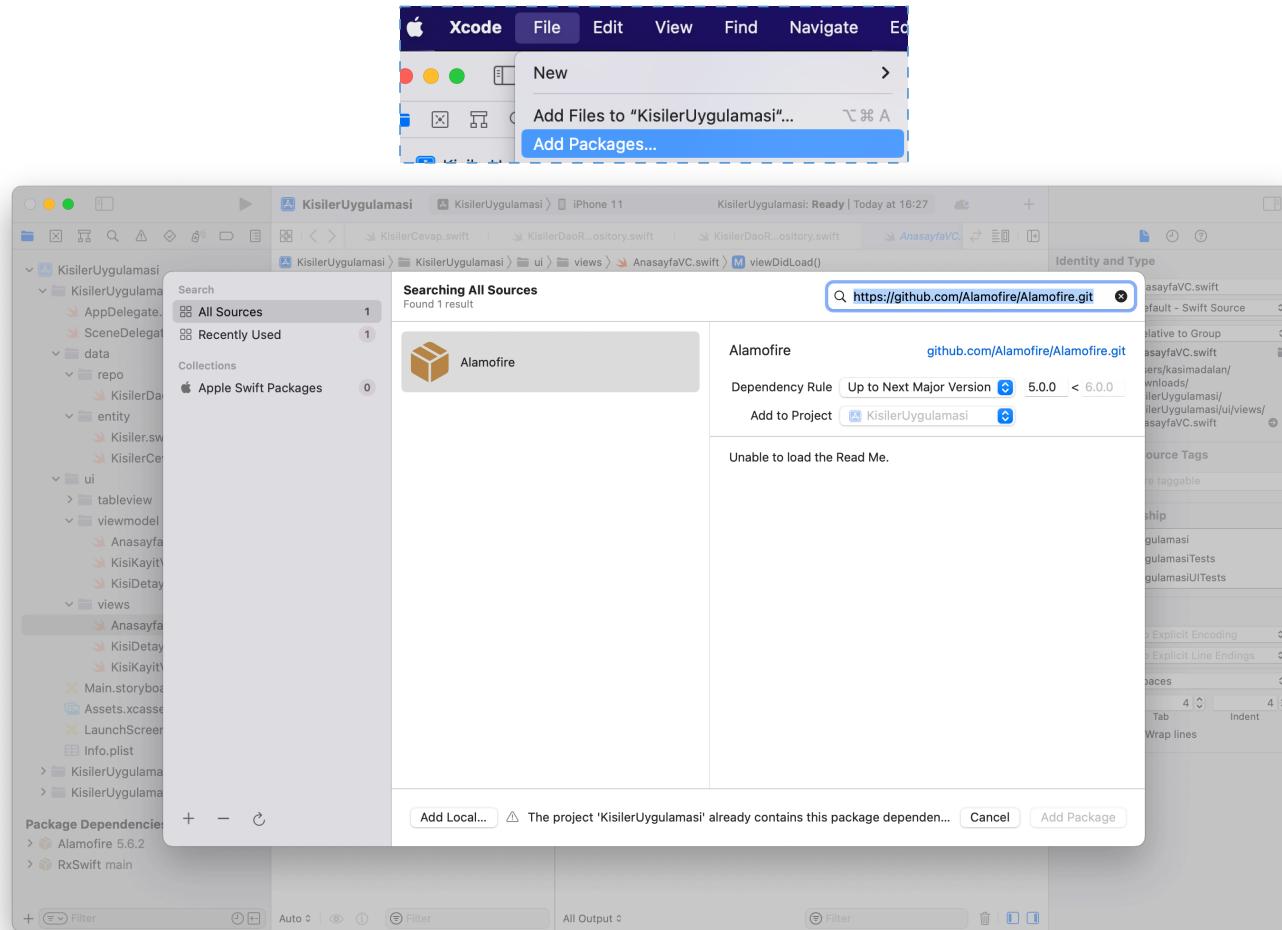
internet izni

The screenshot shows the Xcode Property List Editor with the following details:

File Path: KisilerUygulamasi > KisilerUygulamasi > Info.plist > No Selection

Key	Type	Value
Information Property List	Dictionary	(2 items)
App Transport Security Settings	Dictionary	(1 item)
Allow Arbitrary Loads	Boolean	YES
Application Scene Manifest	Dictionary	(2 items)

Kingfisher Kurulum



<https://github.com/onevcat/Kingfisher.git>

Kingfisher ile Resim Alma

```
import UIKit
import Kingfisher

class ViewController: UIViewController {
    @IBOutlet weak var imageView: UIImageView!

    override func viewDidLoad() {
        super.viewDidLoad()

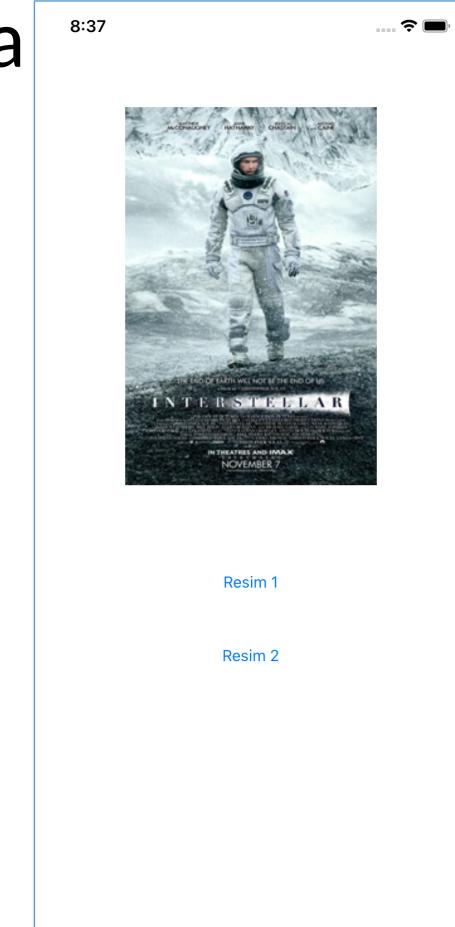
        resimGosterKingfisher(resimAdi:"django.png")

    }
    @IBAction func button1Tiklandi(_ sender: Any) {
        resimGosterKingfisher(resimAdi:"thehatefuleight.png")
    }

    @IBAction func button2Tiklandi(_ sender: Any) {
        resimGosterKingfisher(resimAdi:"interstellar.png")
    }

    func resimGosterKingfisher(resimAdi:String){
        if let url = URL(string: "http://kasimadalan.pe.hu/filmller/resimler/\(resimAdi)") {

            DispatchQueue.main.async {
                self.imageView.kf.setImage(with: url)
            }
        }
    }
}
```



Resim 1

Resim 2

Placeholder Resim Kullanma

- Bu yöntem ile resim yükleme işlemi olana kadar varsayılan bir resim gösterilebilir.
- Asset dosyasındaki bir resim ile bunu sağlayabiliriz.



Placeholder Resim Kullanma

- Bu yöntem ile resim yükleme işlemi olana kadar varsayılan bir resim gösterilebilir.
- Asset dosyasındaki bir resim ile bunu sağlayabiliriz.

```
if let url = URL(string: "http://kasimadalan.pe.hu/filmler/resimler/\\(resimAdi)") {  
    DispatchQueue.main.async {  
        let placeholder = UIImage(named: "placeholder") Resim adıyla asset içindeki resme erişim.  
        self.imageView.kf.setImage(with: url, placeholder: placeholder)  
    }  
}
```

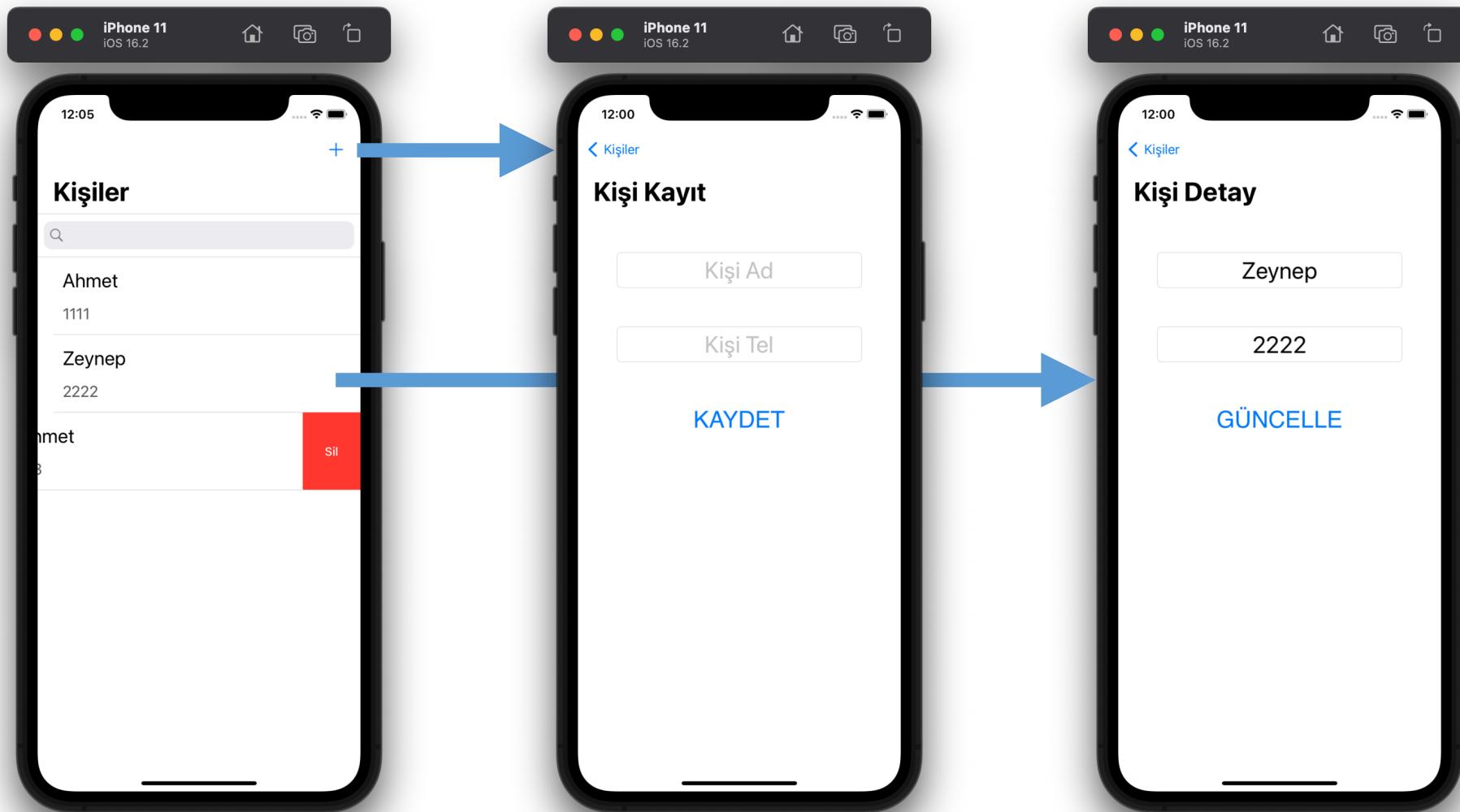
Resim Yüklenme İşlemiini Takip Etme

```
if let url = URL(string: "http://kasimadalan.pe.hu/filmller/resimler/\\(resimAdi)") {  
    DispatchQueue.main.async {  
        let placeholder = UIImage(named: "placeholder")  
  
        self.imageView.kf.setImage(with: url,placeholder: placeholder,completionHandler: { result in  
            switch result {  
                case .success(_)://Eğer resim değeri ile işlem yapmak istiyorsan .success(let value) kullan.  
                    print("Resim Yüklendi")  
                case .failure(let error):  
                    print("Hata : \(error)")  
            }  
        })  
    }  
}
```

Kişiler Uygulaması

Alamofire

Kişiler Uygulaması



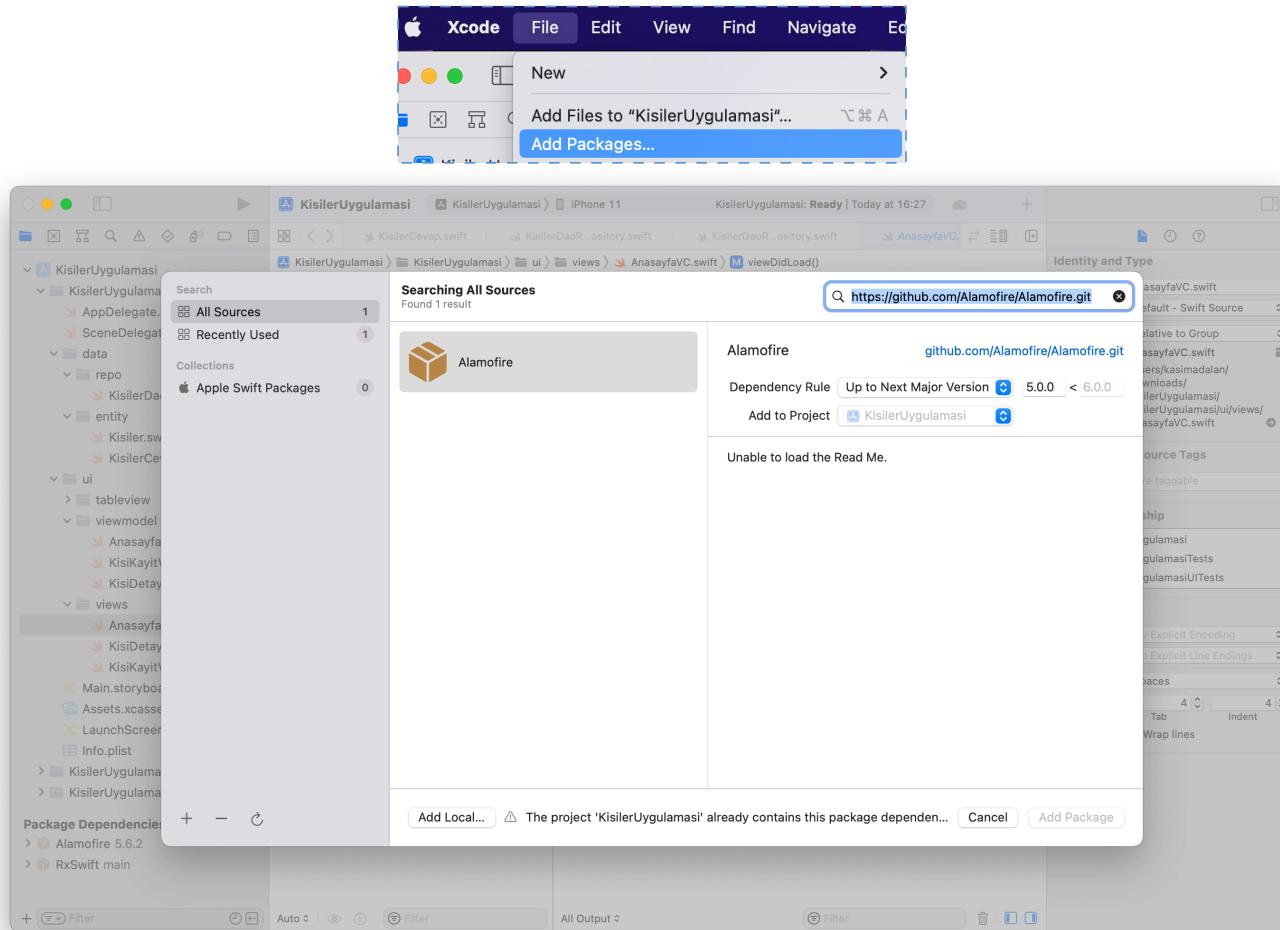
internet izni

The screenshot shows the Xcode Property List Editor with the following details:

File Path: KisilerUygulamasi > KisilerUygulamasi > Info.plist > No Selection

Key	Type	Value
Information Property List	Dictionary	(2 items)
App Transport Security Settings	Dictionary	(1 item)
Allow Arbitrary Loads	Boolean	YES
Application Scene Manifest	Dictionary	(2 items)

Alamofire Kurulum



<https://github.com/Alamofire/Alamofire.git>

```
<?php  
$response = array();  
  
if (isset($_POST['kisi_ad']) && isset($_POST['kisi_tel'])) {  
    $kisi_ad = $_POST['kisi_ad'];  
    $kisi_tel = $_POST['kisi_tel'];  
  
    //DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.  
    require_once __DIR__ . '/db_config.php';  
  
    // Bağlantı oluşturuluyor.  
    $baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);  
  
    // Bağlantı kontrolü yapılır.  
    if (!$baglanti) {  
        die("Hatalı bağlantı : " . mysqli_connect_error());  
    }  
  
    $sqlsorgu = "INSERT INTO kisiler (kisi_ad,kisi_tel) VALUES ('$kisi_ad','$kisi_tel')";  
  
    if (mysqli_query($baglanti, $sqlsorgu)) {  
        $response["success"] = 1;  
        $response["message"] = "successfully ";  
        echo json_encode($response);  
    } else {  
        $response["success"] = 0;  
        $response["message"] = "No product found";  
        echo json_encode($response);  
    }  
    //bağlantı koparılır.  
    mysqli_close($baglanti);  
} else {  
    $response["success"] = 0;  
    $response["message"] = "Required field(s) is missing";  
    echo json_encode($response);  
}  
?>
```

http://kasimadalan.pe.hu/kisiler/insert_kisiler.php

```
{ -  
  "success":0,  
  "message":"Required field(s) is missing"  
}
```

```
<?php  
  
$response = array();  
  
if (isset($_POST['kisi_id']) && isset($_POST['kisi_ad']) && isset($_POST['kisi_tel'])) {  
    $kisi_id = $_POST['kisi_id'];  
    $kisi_ad = $_POST['kisi_ad'];  
    $kisi_tel = $_POST['kisi_tel'];  
  
    //DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.  
    require_once __DIR__ . '/db_config.php';  
  
    // Bağlantı oluşturuluyor.  
    $baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);  
  
    // Bağlantı kontrolü yapılır.  
    if (!$baglanti) {  
        die("Hatalı bağlantı : " . mysqli_connect_error());  
    }  
  
    $sqlsorgu = "UPDATE kisiler SET kisiler.kisi_ad = '$kisi_ad',kisiler.kisi_tel = '$kisi_tel' WHERE  
                kisiler.kisi_id = $kisi_id ";  
  
    if (mysqli_query($baglanti, $sqlsorgu)) {  
        $response["success"] = 1;  
        $response["message"] = "successfully ";  
        echo json_encode($response);  
    } else {  
        $response["success"] = 0;  
        $response["message"] = "No product found";  
        echo json_encode($response);  
    }  
    //bağlantı koparılır.  
    mysqli_close($baglanti);  
} else {  
    $response["success"] = 0;  
    $response["message"] = "Required field(s) is missing";  
    echo json_encode($response);  
}  
?  
  
http://kasimadalan.pe.hu/kisiler/update\_kisiler.php
```

{ -
 "success":0,
 "message":"Required field(s) is missing"
}

```

<?php
$response = array();

if (isset($_POST['kisi_id'])) {
    $kisi_id = $_POST['kisi_id'];

    //DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
    require_once __DIR__ . '/db_config.php';

    // Bağlantı oluşturuluyor.
    $baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

    // Bağlantı kontrolü yapılır.
    if (!$baglanti) {
        die("Hatalı bağlantı : " . mysqli_connect_error());
    }
    $sqlsorgu = "DELETE FROM kisiler WHERE kisiler.kisi_id = $kisi_id";

    if (mysqli_query($baglanti, $sqlsorgu)) {

        $response["success"] = 1;
        $response["message"] = "successfully ";
        echo json_encode($response);
    } else {

        $response["success"] = 0;
        $response["message"] = "No product found";
        echo json_encode($response);
    }
    //bağlantı koparılır.
    mysqli_close($baglanti);
} else {
    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";
    echo json_encode($response);
}
?>

```

http://kasimadalan.pe.hu/kisiler/delete_kisiler.php

```
{
  "success":0,
  "message":"Required field(s) is missing"
}
```

```

<?php
// array for JSON response
$response = array();

//DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
require_once __DIR__ . '/db_config.php';

// Bağlantı oluşturuluyor.
$baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

// Bağlantı kontrolü yapılır.
if (!$baglanti) {
    die("Hatalı bağlantı : " . mysqli_connect_error());
}

$sqlsorgu = "SELECT * FROM kisiler";
$result = mysqli_query($baglanti, $sqlsorgu);

// result kontrolü yap
if (mysqli_num_rows($result) > 0) {

    $response["kisiler"] = array();

    while ($row = mysqli_fetch_assoc($result)) {
        // temp user array
        $kisiler = array();

        $kisiler["kisi_id"] = $row["kisi_id"];
        $kisiler["kisi_ad"] = $row["kisi_ad"];
        $kisiler["kisi_tel"] = $row["kisi_tel"];

        // push single product into final response array
        array_push($response["kisiler"], $kisiler);
    }
    // success
    $response["success"] = 1;

    // echoing JSON response
    echo json_encode($response);
}

```

http://kasimadalan.pe.hu/kisiler/tum_kisiler.php

```

{
    "kisiler": [
        {
            "kisi_id": "376",
            "kisi_ad": "Ahmet",
            "kisi_tel": "5348564412"
        },
        {
            "kisi_id": "379",
            "kisi_ad": "Zeynep",
            "kisi_tel": "05375483214"
        },
        {
            "kisi_id": "380",
            "kisi_ad": "Mehmet",
            "kisi_tel": "05375483215"
        },
        {
            "kisi_id": "381",
            "kisi_ad": "Ayşe",
            "kisi_tel": "05375483216"
        },
        {
            "kisi_id": "382",
            "kisi_ad": "Mustafa",
            "kisi_tel": "05375483217"
        }
    ],
    "success": 1
}

// no products found
$response["success"] = 0;
$response["message"] = "No data found";

// echo no users JSON
echo json_encode($response);
}

//bağlantı koparılır.
mysqli_close($baglanti);
?>

```

```

<?php

$response = array();

if (isset($_POST['kisi_ad'])) {
    $kisi_ad = $_POST['kisi_ad'];

    //DB_SERVER,DB_USER,DB_PASSWORD,DB_DATABASE değişkenleri alınır.
    require_once __DIR__ . '/db_config.php';

    // Bağlantı oluşturuluyor.
    $baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);

    // Bağlantı kontrolü yapılır.
    if (!$baglanti) {
        die("Hatalı bağlantı : " . mysqli_connect_error());
    }

    $sqlsorgu = "SELECT * FROM kisiler WHERE kisiler.kisi_ad like '%$kisi_ad%'";
    $result = mysqli_query($baglanti, $sqlsorgu);

    if (mysqli_num_rows($result) > 0) {

        $response["kisiler"] = array();

        while ($row = mysqli_fetch_assoc($result)) {

            $kisiler = array();

            $kisiler["kisi_id"] = $row["kisi_id"];
            $kisiler["kisi_ad"] = $row["kisi_ad"];
            $kisiler["kisi_tel"] = $row["kisi_tel"];

            array_push($response["kisiler"], $kisiler);
        }

        $response["success"] = 1;
        echo json_encode($response);
    }
    //bağlantı koparılır.
    mysqli_close($baglanti);
}

```

http://kasimadalan.pe.hu/kisiler/tum_kisiler_arama.php

```

{
    "kisiler": [
        {
            "kisi_id": "376",
            "kisi_ad": "Ahmet",
            "kisi_tel": "5348564412"
        },
        {
            "kisi_id": "379",
            "kisi_ad": "Zeynep",
            "kisi_tel": "05375483214"
        },
        {
            "kisi_id": "380",
            "kisi_ad": "Mehmet",
            "kisi_tel": "05375483215"
        },
        {
            "kisi_id": "381",
            "kisi_ad": "Ayşe",
            "kisi_tel": "05375483216"
        },
        {
            "kisi_id": "382",
            "kisi_ad": "Mustafa",
            "kisi_tel": "05375483217"
        }
    ],
    "success": 1
}

```



```

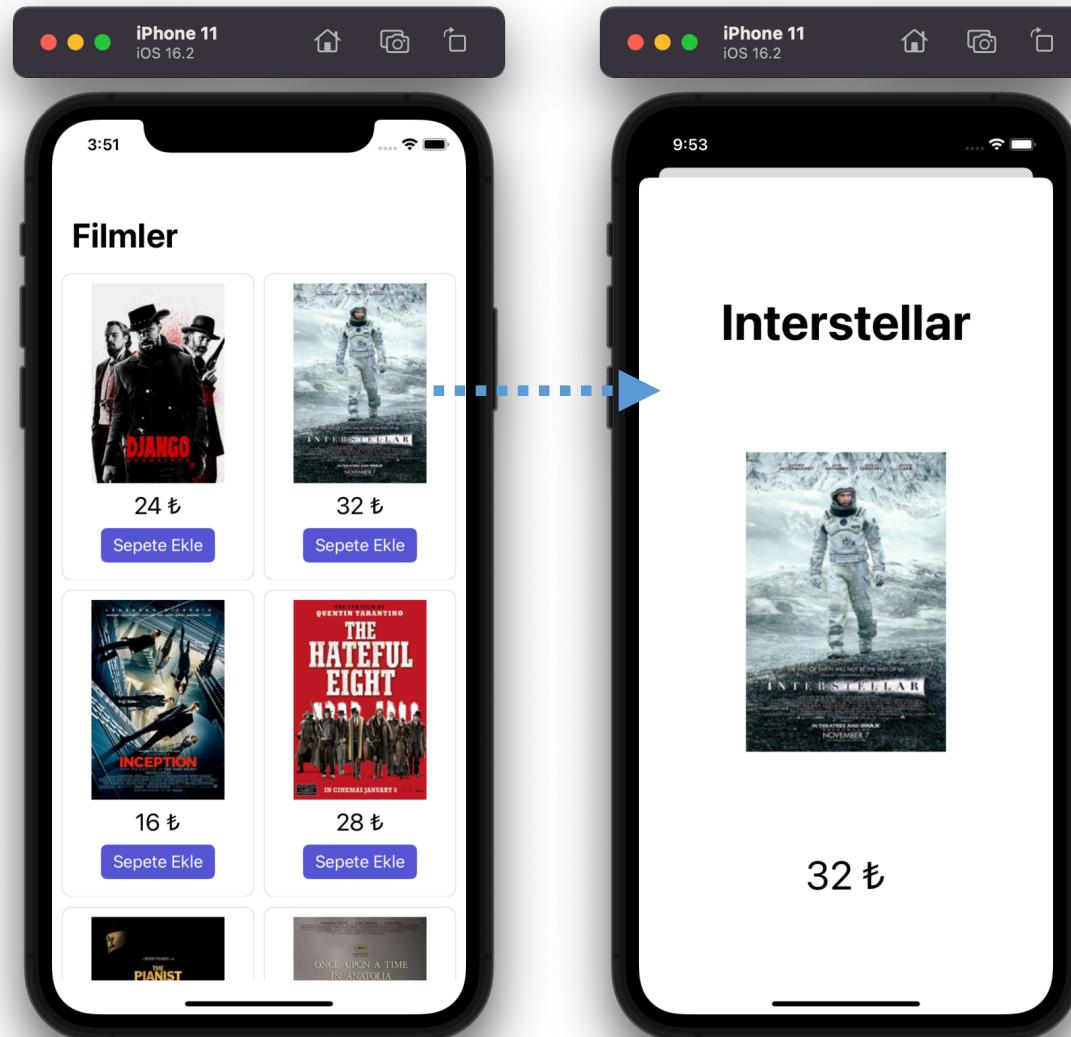
} else {
    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";
    echo json_encode($response);
}
?>

```

Filmler Uygulaması

Alamofire

Film Uygulaması



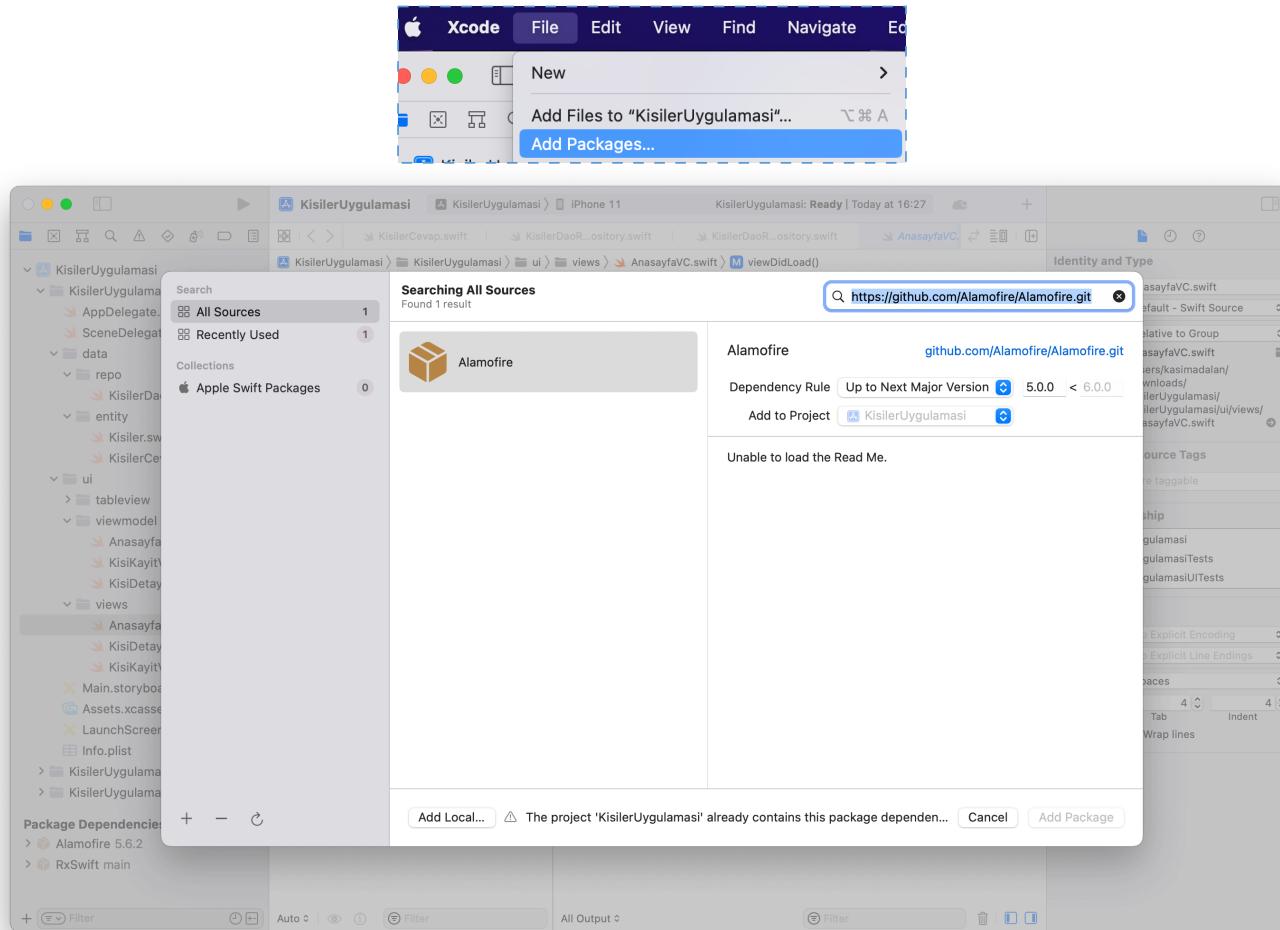
internet izni

The screenshot shows the Xcode Property List Editor with the following details:

File Path: KisilerUygulamasi > KisilerUygulamasi > Info.plist > No Selection

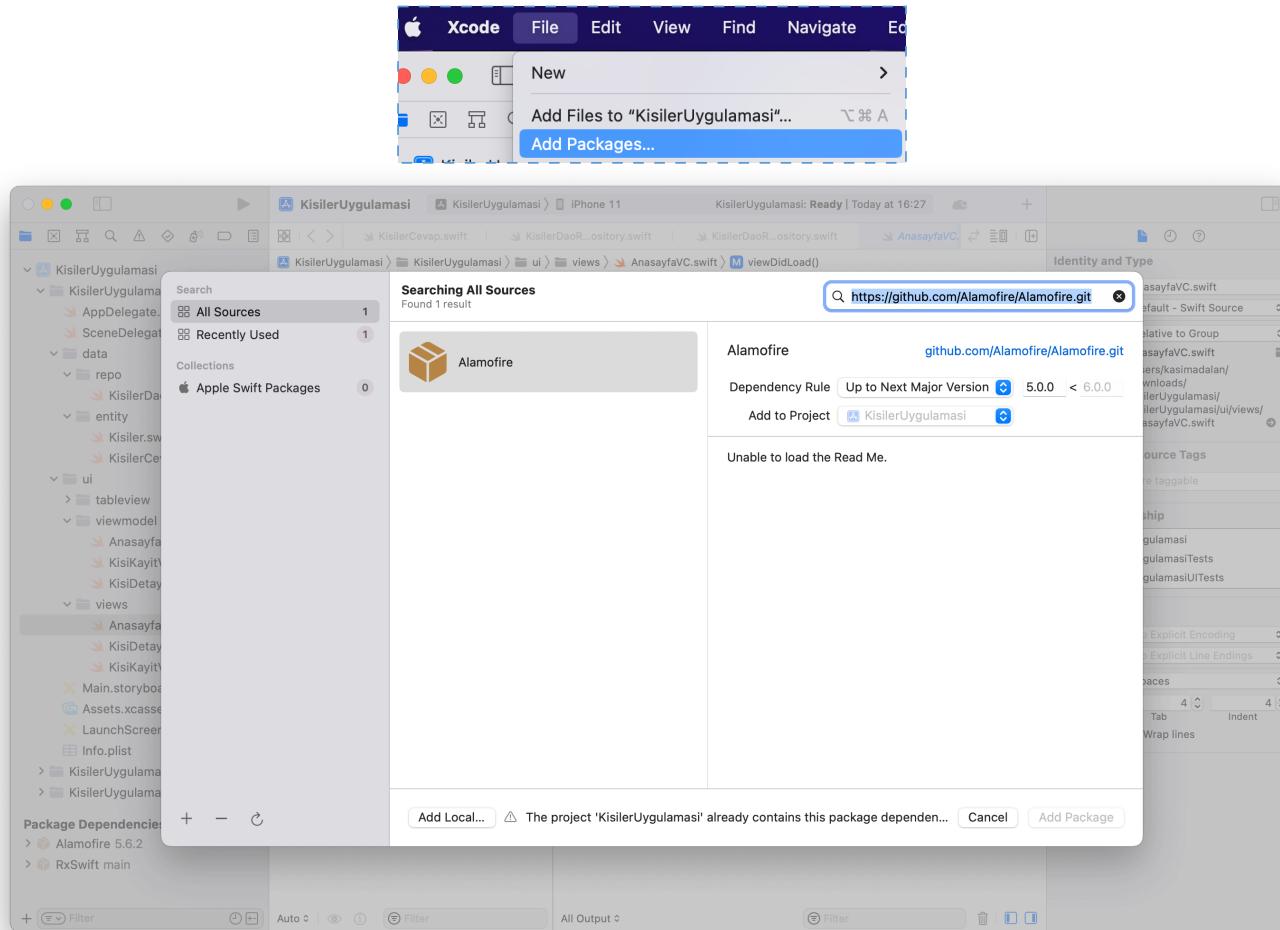
Key	Type	Value
Information Property List	Dictionary	(2 items)
App Transport Security Settings	Dictionary	(1 item)
Allow Arbitrary Loads	Boolean	YES
Application Scene Manifest	Dictionary	(2 items)

Alamofire Kurulum



<https://github.com/Alamofire/Alamofire.git>

Kingfisher Kurulum



<https://github.com/onevcat/Kingfisher.git>

```

<?php
$response = array();
require_once __DIR__ . '/db_config.php';
$baglanti = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);
if (!$baglanti) {
    die("Hatalı bağlantı : " . mysqli_connect_error());
}

$sqlsorgu = "SELECT * FROM filmler_yeni";
$result = mysqli_query($baglanti, $sqlsorgu);

if (mysqli_num_rows($result) > 0) {
    $response["filmler"] = array();

    while ($row = mysqli_fetch_assoc($result)) {
        $filmler = array();
        $filmler["id"] = intval($row["id"]);
        $filmler["ad"] = $row["ad"];
        $filmler["resim"] = $row["resim"];
        $filmler["fiyat"] = intval($row["fiyat"]);
        array_push($response["filmler"], $filmler);
    }

    $response["success"] = 1;
    echo json_encode($response);
} else {
    $response["success"] = 0;
    $response["message"] = "No data found";
    echo json_encode($response);
}
mysqli_close($baglanti);
?>

```

http://kasimadalan.pe.hu/filmler_yeni/tum_filmller.php

```

"filmler": [
    {
        "id": 1,
        "ad": "Interstellar",
        "resim": "interstellar.png",
        "fiyat": 30
    },
    {
        "id": 2,
        "ad": "Inception",
        "resim": "inception.png",
        "fiyat": 20
    },
    {
        "id": 3,
        "ad": "The Dark Knight",
        "resim": "dark_knight.png",
        "fiyat": 25
    },
    {
        "id": 4,
        "ad": "The Godfather",
        "resim": "godfather.png",
        "fiyat": 15
    }
],
"success": 1
}

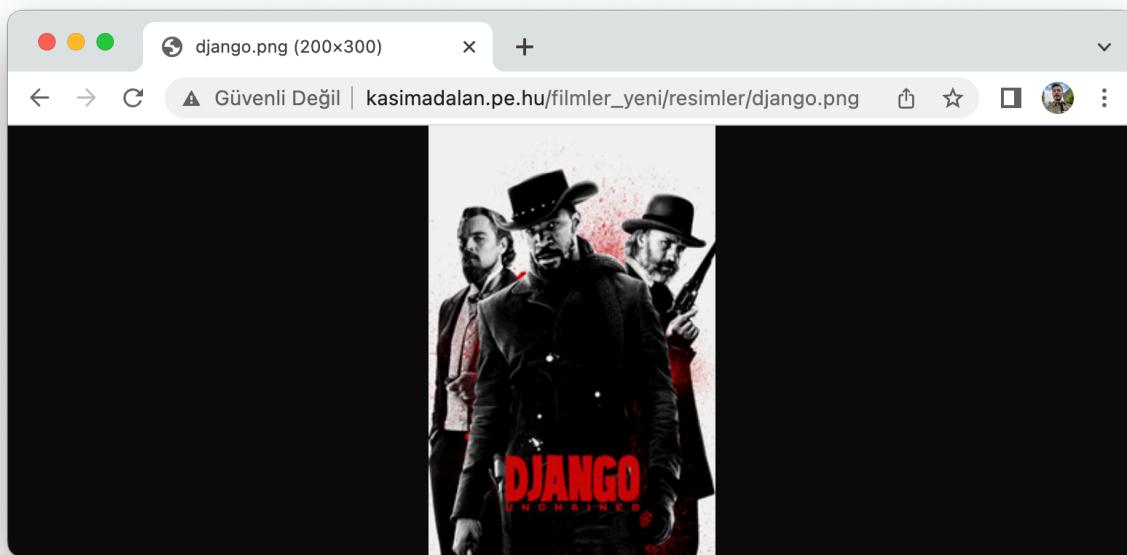
```

Resimleri Görüntüleme

`http://kasimadalan.pe.hu/filmler_yeni/resimler/django.png`

Base Url altında yer alan resimler

Name ↑
 anadoluda.png
 django.png
 inception.png
 interstellar.png
 thehatefuleight.png
 thepianist.png



Firebase Firestore

Firestore



Gerçek zamanlı çalışabilen bir veritabanıdır.

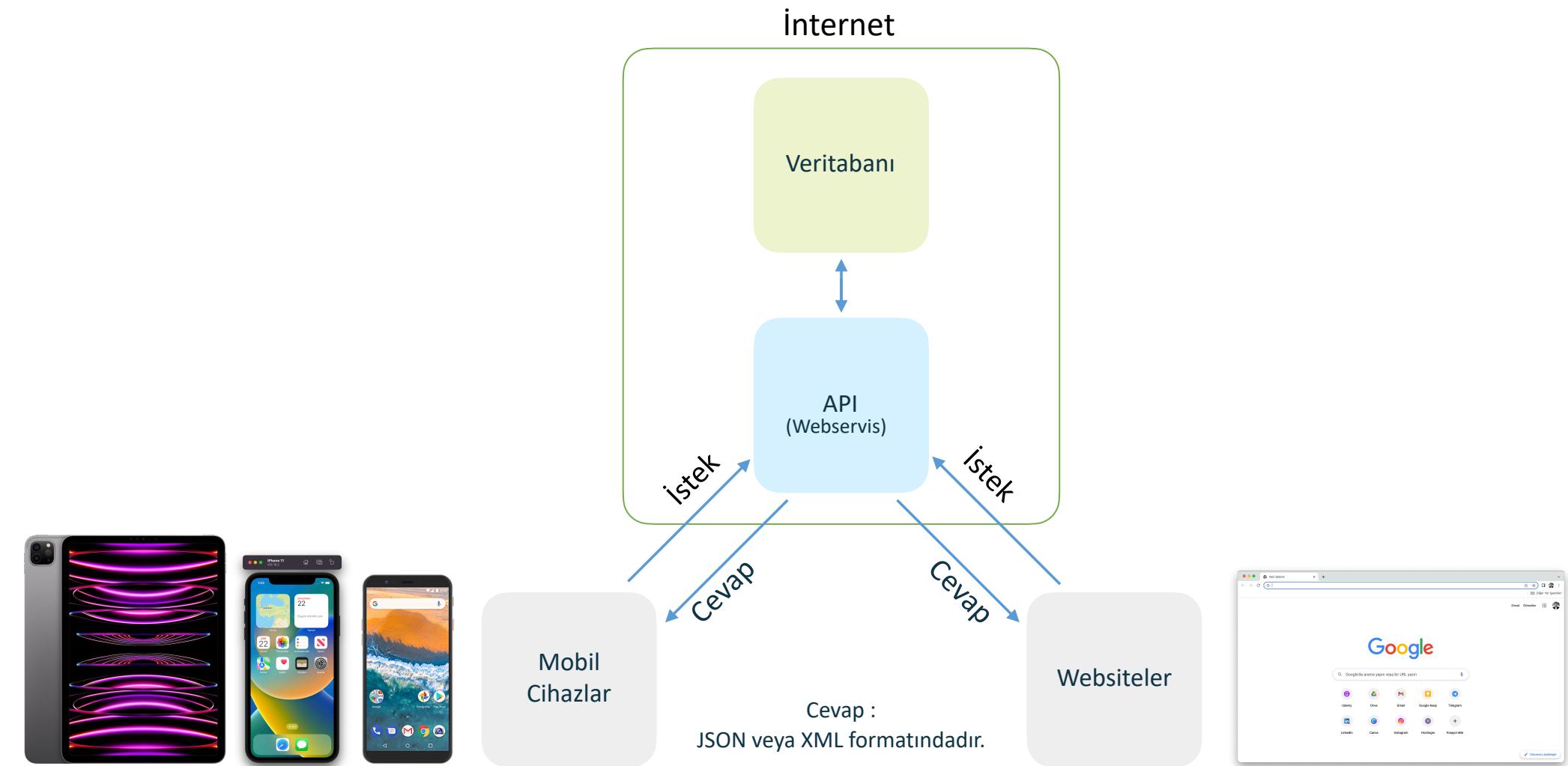
Veritabanında oluşan değişiklik anlık olarak uygulamaya aktarılabilir.

No SQL veritabanıdır , klasik sql sorguları çalışmamaktadır.

Webservis (API) kullanmanıza gerek yoktur.

Veriler json modeli ile kullanılmaktadır.

RESTful Mimarisi



Kurulum Aşamaları

1 Firebase Projesi Oluşturma

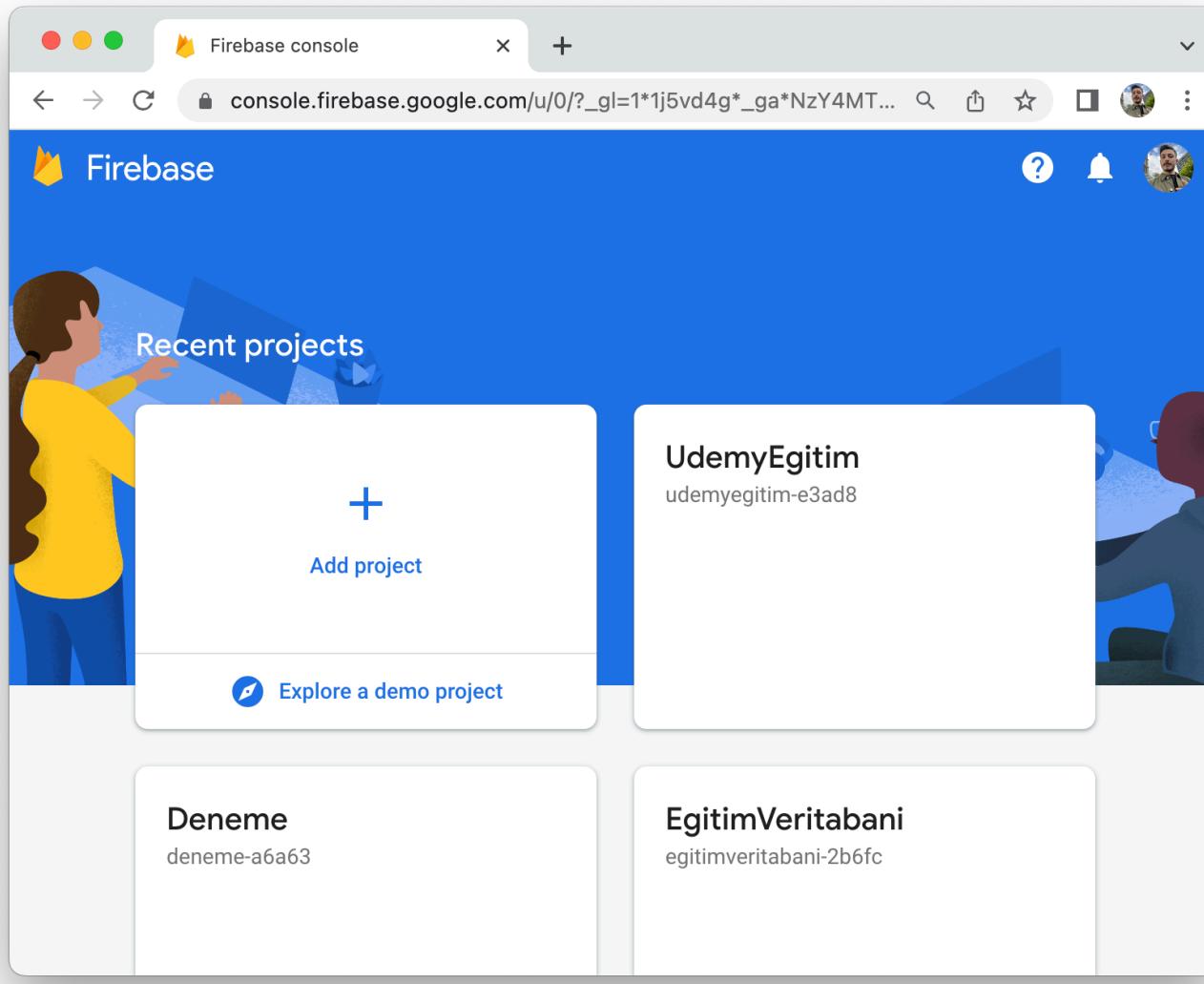
2 Firestore Veritabanı Oluşturma

3 Xcode Projesi Oluşturma

4 Firebase ve Xcode Bağlantısı Oluşturma

1

Firebase Projesi Oluşturma



2

Firestore Veritabanı Oluşturma

Cloud Firestore

Realtime updates, powerful queries, and automatic scaling

Create database



Is Cloud Firestore right for you? [Compare Databases](#)

Firebase Veritabanı Oluşturma

Create database

1 Secure rules for Cloud Firestore —— 2 Set Cloud Firestore location

After you define your data structure, **you will need to write rules to secure your data.**

[Learn more ↗](#)

Start in **production mode**
Your data is private by default. Client read/write access will only be granted as specified by your security rules.

Start in **test mode**
Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.

```
rules_version = '2';

service cloud.firestore {
    match /databases/{database}/documents {
        match /{document=**} {
            allow read, write: if false;
        }
    }
}
```

i All third party reads and writes will be denied

Enabling Cloud Firestore will prevent you from using Cloud Datastore with this project

Cancel

Next

Firebase Veritabanı Oluşturma

Cloud Firestore

Data Rules Indexes Usage  Extensions NEW



Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing

[Configure App Check](#)



[Panel view](#)

[Query builder](#)



[More in Google Cloud](#) ▾

 udemyegitim-e3ad8

 [Start collection](#)

Veritabanı Kuralı Düzenleme

Cloud Firestore

Data Rules **Rules** Indexes Usage Extensions NEW

Edit rules Monitor rules Develop & Test

Today • 12:52 PM	rules_version = '2'; service cloud.firestore { match /databases/{database}/documents { match /{document=**} { allow read, write: if true; } } }
Today • 12:51 PM	

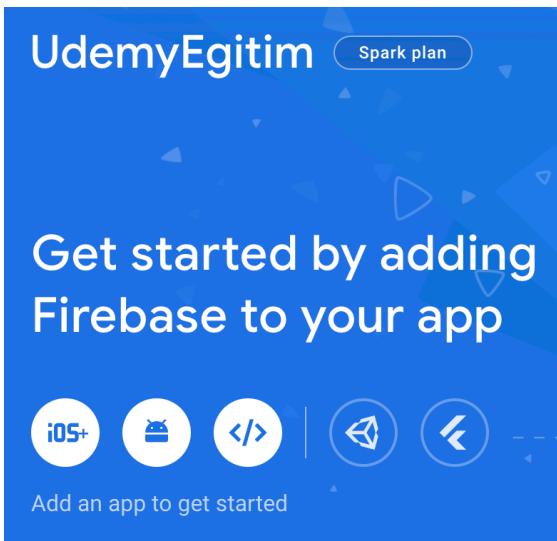
3

Xcode Projesi Oluşturma



4

Firebase ve Xcode Bağlantısı Oluşturma



× Add Firebase to your Apple app

1 Register app

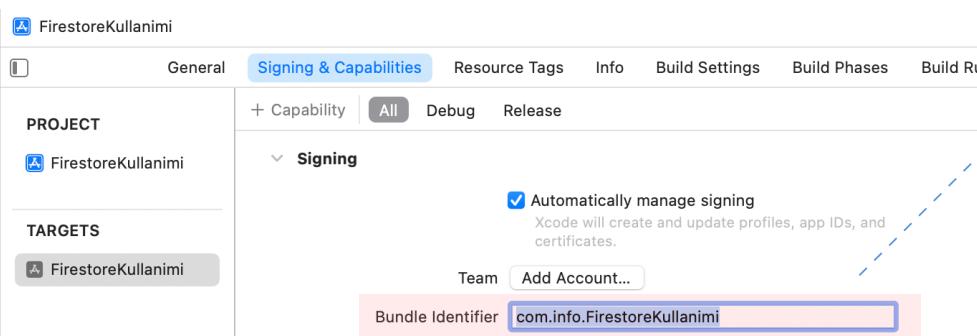
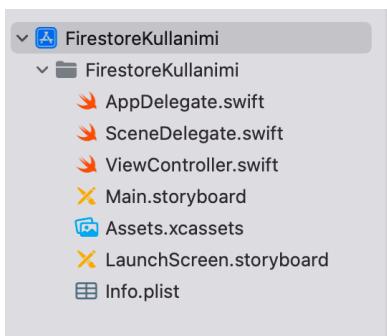
Apple bundle ID ②
com.info.FirestoreKullanimi

App nickname (optional) ②
My Apple App

App Store ID (optional) ②
123456789

Register app

A dashed arrow points from the 'Apple bundle ID' field in the Firebase registration form to the 'Bundle Identifier' field in the Xcode project settings.



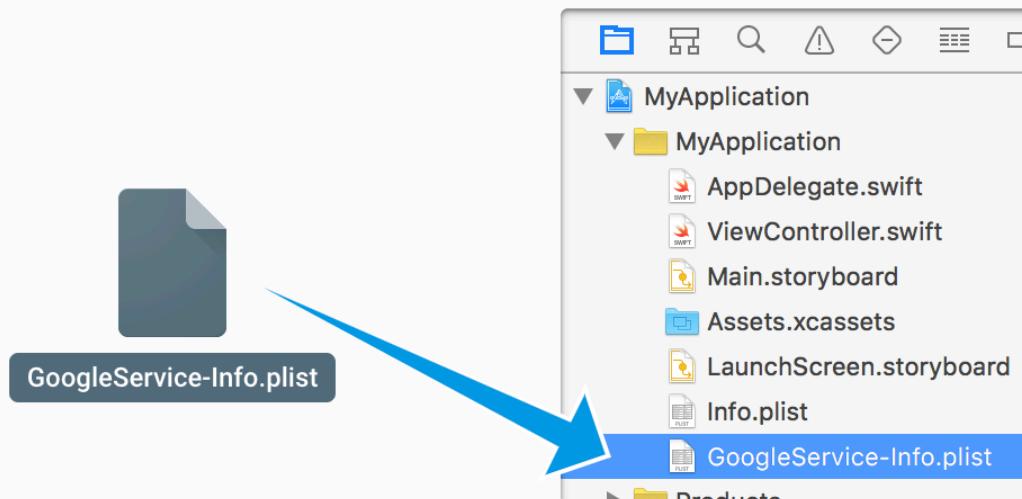
Config Dosyasını Projeye Ekleme

2 Download config file

Instructions for Xcode below | [Unity](#) [C++](#)

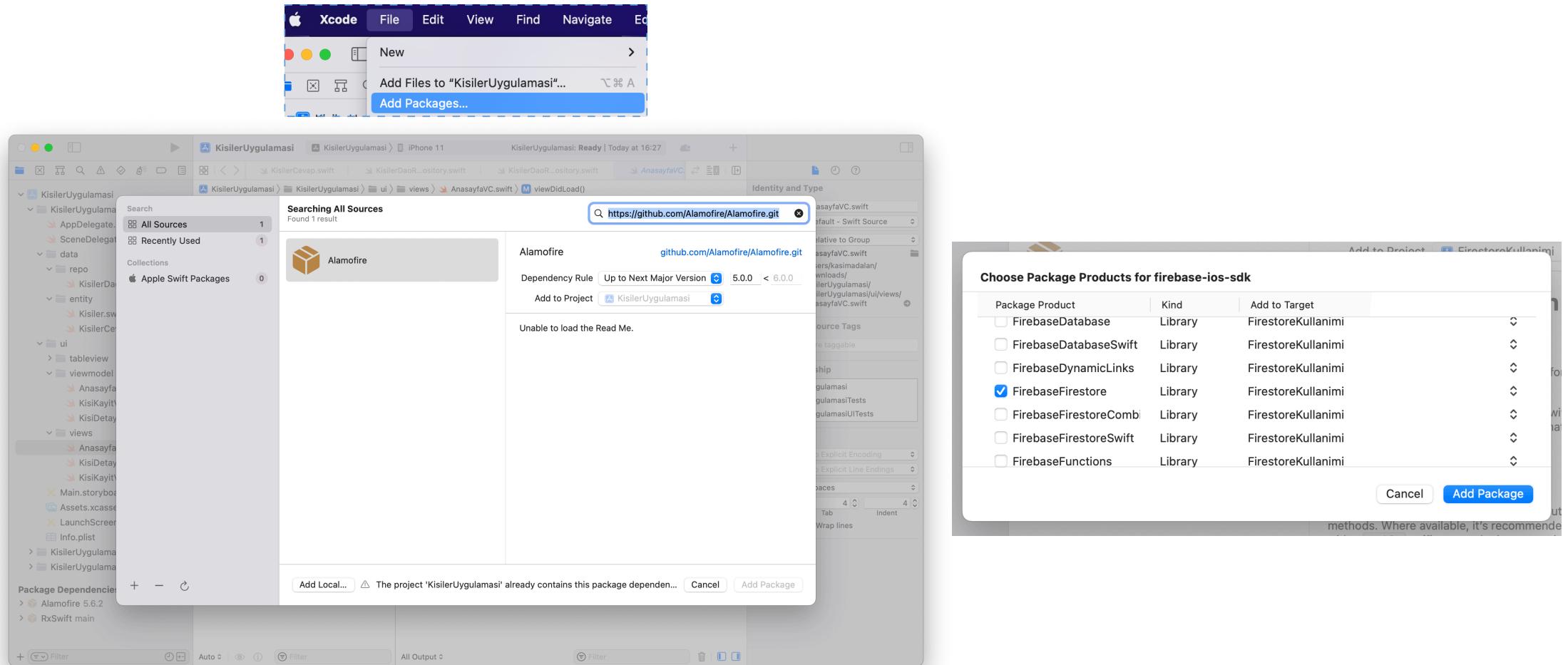
 [Download GoogleService-Info.plist](#)

Move the GoogleService-Info.plist file you just downloaded into the root of your Xcode project and add it to all targets.



[Next](#)

Firebase Kurulum



<https://github.com/firebase/firebase-ios-sdk>

Standart Kütüphane Ekleme

SwiftUI

Swift

Objective-C

```
import UIKit
import FirebaseCore

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {

    var window: UIWindow?

    func application(_ application: UIApplication,
                    didFinishLaunchingWithOptions launchOptions:
                        [UIApplicationLaunchOptionsKey: Any]?) -> Bool {
        FirebaseApp.configure()
        return true
    }
}
```



Previous

Next

Proje Kurulum Kontrol

4 Next steps

You're all set!

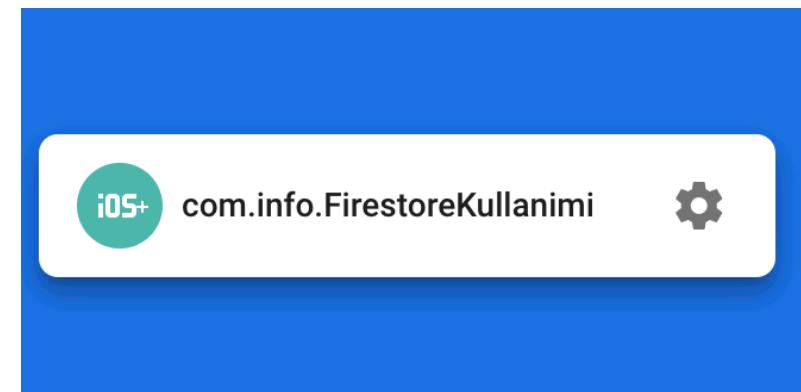
Make sure to check out the [documentation](#) to learn how to get started with each Firebase product that you want to use in your app.

You can also explore [sample Firebase apps](#).

Or, continue to the console to explore Firebase.

Previous

Continue to console



İşlem Yapılacak Tablonun (Collection) Seçilmesi

```
import UIKit
import FirebaseFirestore

class ViewController: UIViewController {
    var collectionUrunler:CollectionReference?
    override func viewDidLoad() {
        super.viewDidLoad()

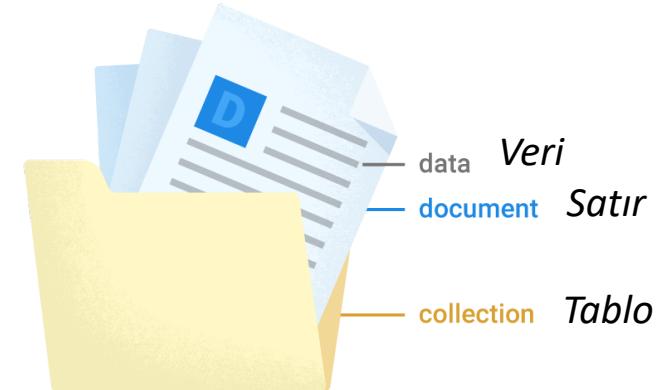
        let db = Firestore.firestore()
        collectionUrunler = db.collection("Urunler")
    }
}
```

The screenshot shows the Firebase Realtime Database interface. At the top, there is a navigation bar with icons for home, collections, and documents, followed by the path 'Urunler > rGE4uSMvvja7f...'. Below the path, there are three main sections: a sidebar with a collection named 'Urunler' highlighted in pink, a middle section with a document named 'rGE4uSMvvja7fWLqui4R' also highlighted in pink, and a right panel showing the fields of that document: 'ad: "TV"' and 'fiyat: 10000'.

udemyegitim-e3ad8	Urunler	rGE4uSMvvja7fWLqui4R
+ Start collection	+ Add document	+ Start collection
Urunler	rGE4uSMvvja7fWLqui4R	ad: "TV" fiyat: 10000

Veri Yapısı

```
class Urunler {  
    var ad:String?  
    var fiyat:Int?  
  
    init(ad: String, fiyat: Int) {  
        self.ad = ad  
        self.fiyat = fiyat  
    }  
}
```



Panel view Query builder

Home > Urunler > 5yHotBXmeqLc.. More in Google Cloud ▾

udemyegitim-e3ad8	Urunler	5yHotBXmeqLcHEPBLxpe
+ Start collection	+ Add document Satır	+ Start collection
Urunler	5yHotBXmeqLcHEPBLxpe	+ Add field
Tablo	DeSkdnMeg66zJcb4WKMz	ad: "Saat"
	QGW5BiCj35M97Bwgq8oc	fiyat: 5000 Veri
	nYtOylgCYQfHuvoZ1Yge	
	rGE4uSMvvja7fWLqui4R	

```

class Urunler {
    var ad:String?
    var fiyat:Int?

    init(ad: String, fiyat: Int) {
        self.ad = ad
        self.fiyat = fiyat
    }
}

```

Veri Yapısı

Panel view Query builder [View docs](#)

Run Clear

Query scope Path [?](#)

Collection /Urunler **Tablo**

[+ Add to query](#)

Veri
data

Satır
document

Collection
Table

Query results

Document ID	ad	fiyat
5yHotBXmeqLcHEPBLxpe	"Saat"	5000
DeSkdnMeg66zJcb4WKMz	"Kalem"	1000
QGW5BiCj35M97Bwgq8oc	"Laptop"	20000

Satır

Klasik Veritabanı Görünümü

Panel view [Query builder](#) [View docs ↗](#)

[Run](#) [Clear](#)

Query scope Path ⓘ
Collection /Urunler

[+ Add to query](#)

Query results ☰

Document ID	ad	fiyat
5yHotBXmeqLcHEPBLxpe	"Saat"	5000
DeSkdnMeg66zJcb4WKMz	"Kalem"	1000
QGW5BiCj35M97Bwgq8oc	"Laptop"	20000
nYtOylgCYQfHuvoZ1Yge	"Telefon"	30000
rGE4uSMvvja7fWLqui4R	"TV"	10000

Items per page: 50 ▾ 1 - 5 of 5 < >

Insert - Veri Kaydı

```
func kayit(){
    let yeniUrun:[String:Any] = ["ad":"TV","fiyat":10000]
    collectionUrunler!.document().setData(yeniUrun)
}
```

Query results		
Document ID	ad	fiyat
5yHotBXmeqLcHEPBLxpe	"Saat"	5000
DeSkdnMeg66zJcb4WKMz	"Kalem"	1000
QGW5BiCj35M97Bwgq8oc	"Laptop"	20000
nYtOylgCYQfHuvoZ1Yge	"Telefon"	30000
rGE4uSMvvja7fWLqui4R	"TV"	10000

Update - Veri Güncelleme

```
func guncelle(){
    let guncellenenUrun:[String:Any] = ["ad":"Yeni TV", "fiyat":9999]
    collectionUrunler!.document("3Xr8J4M2GrgfDPssbS0N").updateData(guncellenenUrun)
}
```

Query results

Document ID	ad	fiyat
3Xr8J4M2GrgfDPssbS0N	"Yeni TV"	9999
EkmeM44uSmVqcaan5xkk	"Saat"	5000
RP2CjmwQZGKxLQbVnevg	"Kalem"	1000
dyB1l7VnE7rd4DX5VOVV	"Telefon"	30000
zYaGD29xVQjmupS0AEfb	"Laptop"	20000

Items per page: 50 ▾ 1 - 5 of 5 < >

Delete - Veri Silme

```
func sil(){
    collectionUrunler!.document("3Xr8J4M2GrgfDPssbS0N").delete()
}
```

Query results

Document ID	ad	fiyat
5yHotBXmeqLcHEPBLxpe	"Saat"	5000
DeSkdnMeg66zJcb4WKMz	"Kalem"	1000
QGW5BiCj35M97Bwgq8oc	"Laptop"	20000
nYtOylgCYQfHuvoZ1Yge	"Telefon"	30000

Query - Veri Okuma

Klasik okuma işlemi

```
func veriOkuma(){
    collectionUrunler!.getDocuments() { snapshot, error in
        if let documents = snapshot?.documents {
            for document in documents {
                let data = document.data()
                let ad = data["ad"] as? String ?? ""
                let fiyat = data["fiyat"] as? Int ?? 0

                let urun = Urunler(ad: ad, fiyat: fiyat)
                print("-----")
                print("Ürün key : \(document.documentID)")
                print("Ürün ad : \(urun.ad!)")
                print("Ürün fiyat : \(urun.fiyat!)")


            }
        }
    }
}
```

Query results		
Document ID	ad	fiyat
5yHotBXmeqLcHEPBLxpe	"Saat"	5000
DeSkdnMeg66zJcb4WKMz	"Kalem"	1000
QGW5BiCj35M97Bwgq8oc	"Laptop"	20000
nYtOylgCYQfHuvoZ1Yge	"Telefon"	30000

Query - Veri Okuma

Realtime (Gerçek zamanlı) okuma işlemi

```
func veriOkumaRealtime(){
    collectionUrunler!.addSnapshotListener { snapshot , error in
        if let documents = snapshot?.documents {
            for document in documents {
                let data = document.data()
                let ad = data["ad"] as? String ?? ""
                let fiyat = data["fiyat"] as? Int ?? 0

                let urun = Urunler(ad: ad, fiyat: fiyat)
                print("-----")
                print("Ürün key : \(document.documentID)")
                print("Ürün ad : \(urun.ad!)")
                print("Ürün fiyat : \(urun.fiyat!)")
            }
        }
    }
}
```

Query results		
Document ID	ad	fiyat
5yHotBXmeqLcHEPBLxpe	"Saat"	5000
DeSkdnMeg66zJcb4WKMz	"Kalem"	1000
QGW5BiCj35M97Bwgq8oc	"Laptop"	20000
nYtOylgCYQfHuvoZ1Yge	"Telefon"	30000

Sorgu

QueryBuilder

Firestore konsolu üzerinden veriler üzerinde sorgu oluşturabilir.

Run Clear

Query scope Path ?
Collection /Urunler

Where fiyat > number 10000

+ Add to query

Count
Return the number of documents that match all conditions

And
Use specific operators to match conditions

Order by
Sort results in ascending or descending order by property

Limit
Return up to a specific number of results

Query results

Document ID	ad	fiyat
QGW5BiCj35M97Bwgq8oc	"Laptop"	20000
nYtOylgCYQfHuvoZ1Yge	"Telefon"	30000

Items per page: 50 1 - 2 of 2

Where

İstediğiniz alanda koşul oluşturarak veri okuyabilirsiniz.

Koşul	Firestore Karşılığı	Çalıştığı Tür
==	isEqualTo	Sayısal - Metinsel
!=	isNotEqualTo	Sayısal - Metinsel
>	isGreaterThan	Sayısal
>=	isGreaterThanOrEqualTo	Sayısal
<	isLessThan	Sayısal
<=	isLessThanOrEqualTo	Sayısal

```
func whereSorgu(){
    let sorgu = collectionUrunler!.whereField("ad", isEqualTo: "Saat")

    sorgu.getDocuments() { snapshot, error in
        if let documents = snapshot?.documents {
            for document in documents {
```

Where

```
func whereSorgu() {  
    let sorgu = collectionUrunler!.whereField("ad", isEqualTo: "Saat")  
  
    sorgu.getDocuments() { snapshot, error in  
        if let documents = snapshot?.documents {  
            for document in documents {  
                let data = document.data()  
                let ad = data["ad"] as? String ?? ""  
                let fiyat = data["fiyat"] as? Int ?? 0  
  
                let urun = Urunler(ad: ad, fiyat: fiyat)  
                print("-----")  
                print("Ürün key : \(document.documentID)")  
                print("Ürün ad : \(urun.ad!)")  
                print("Ürün fiyat : \(urun.fiyat!)")  
            }  
        }  
    }  
}
```

Where

```
func whereSorgu(){
    let sorgu = collectionUrunler!.whereField("fiyat", isGreaterThan: 10000)

    sorgu.getDocuments() { snapshot, error in
        if let documents = snapshot?.documents {
            for document in documents {
                let data = document.data()
                let ad = data["ad"] as? String ?? ""
                let fiyat = data["fiyat"] as? Int ?? 0

                let urun = Urunler(ad: ad, fiyat: fiyat)
                print("-----")
                print("Ürün key : \(document.documentID)")
                print("Ürün ad : \(urun.ad!)")
                print("Ürün fiyat : \(urun.fiyat!)")
            }
        }
    }
}
```

Where - Birleşik Koşul

```
func whereSorgu(){
    let sorgu = collectionUrunler!
        .whereField("fiyat", isGreaterThanOrEqualTo: 10000)
        .whereField("fiyat", isLessThanOrEqualTo: 25000)      fiyat > 10000 AND fiyat < 25000

    sorgu.getDocuments() { snapshot, error in
        if let documents = snapshot?.documents {
            for document in documents {
                let data = document.data()
                let ad = data["ad"] as? String ?? ""
                let fiyat = data["fiyat"] as? Int ?? 0

                let urun = Urunler(ad: ad, fiyat: fiyat)
                print("-----")
                print("Ürün key : \(document.documentID)")
                print("Ürün ad : \(urun.ad!)")
                print("Ürün fiyat : \(urun.fiyat!)")
            }
        }
    }
}
```

OrderBy

Verileri sıralama koşuludur.

```
func orderBySorgu(){
    let sorgu = collectionUrunler!.order(by: "fiyat")ASCENDING :  
küçükten büyüğe  
sıralama
    sorgu.getDocuments() { snapshot, error in
        if let documents = snapshot?.documents {
            for document in documents {
                let data = document.data()
                let ad = data["ad"] as? String ?? ""
                let fiyat = data["fiyat"] as? Int ?? 0

                let urun = Urunler(ad: ad, fiyat: fiyat)
                print("-----")
                print("Ürün key : \(document.documentID)")
                print("Ürün ad : \(urun.ad!)")
                print("Ürün fiyat : \(urun.fiyat!)")
            }
        }
    }
}
```

OrderBy

Verileri sıralama koşuludur.

```
func orderBySorgu(){
    let sorgu = collectionUrunler!.order(by: "fiyat",descending: true) DESCENDING :  
büyükten küçüğe  
sıralama
    sorgu.getDocuments() { snapshot, error in
        if let documents = snapshot?.documents {
            for document in documents {
                let data = document.data()
                let ad = data["ad"] as? String ?? ""
                let fiyat = data["fiyat"] as? Int ?? 0

                let urun = Urunler(ad: ad, fiyat: fiyat)
                print("-----")
                print("Ürün key : \(document.documentID)")
                print("Ürün ad : \(urun.ad!)")
                print("Ürün fiyat : \(urun.fiyat!)")
            }
        }
    }
}
```

Limit

Sınırlı sayıda veri okuma koşulu

```
func limitSorgu(){
    let sorgu = collectionUrunler!.order(by: "fiyat").limit(to: 2)
    sorgu.getDocuments() { snapshot, error in
        if let documents = snapshot?.documents {
            for document in documents {
                let data = document.data()
                let ad = data["ad"] as? String ?? ""
                let fiyat = data["fiyat"] as? Int ?? 0
                let urun = Urunler(ad: ad, fiyat: fiyat)
                print("-----")
                print("Ürün key : \(document.documentID)")
                print("Ürün ad : \(urun.ad!)")
                print("Ürün fiyat : \(urun.fiyat!)")
            }
        }
    }
}
```

*Sıralamaya göre
ilk 2
veriyi getirir.*

*Not :
limit özelliği orderBy
ile kullanılması
 önerilir.*

Limit

Sınırlı sayıda veri okuma koşulu

```
func limitSorgu(){
    let sorgu = collectionUrunler!.order(by: "fiyat").limit(toLast: 2) Sıralamaya göre  
son 2  
veriyi getirir.
    sorgu.getDocuments() { snapshot, error in
        if let documents = snapshot?.documents {
            for document in documents {
                let data = document.data()
                let ad = data["ad"] as? String ?? ""
                let fiyat = data["fiyat"] as? Int ?? 0

                let urun = Urunler(ad: ad, fiyat: fiyat)
                print("-----")
                print("Ürün key : \((document.documentID))")
                print("Ürün ad : \((urun.ad!))")
                print("Ürün fiyat : \((urun.fiyat!))")
            }
        }
    }
}
```

*Not :
limit özelliği orderBy ile kullanılması önerilir.*

Count

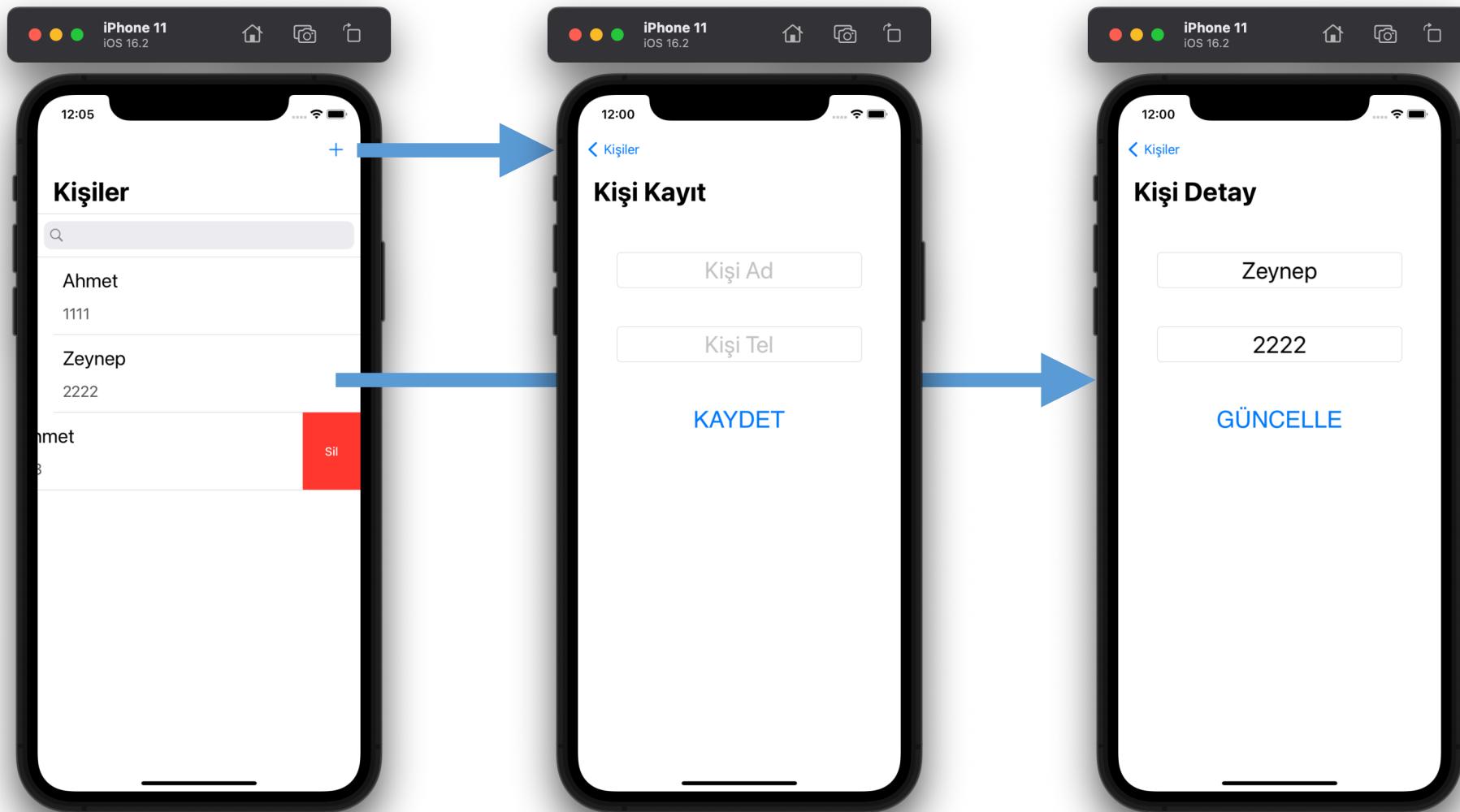
Sorgu sonucunda kaç adet kayıt olduğu bilgisini alabiliriz.

```
func countSorgu(){
    collectionUrunler!.getDocuments() { snapshot, error in
        if let documents = snapshot?.documents {
            print("Sonuç : \(documents.count)")
        }
    }
}
```

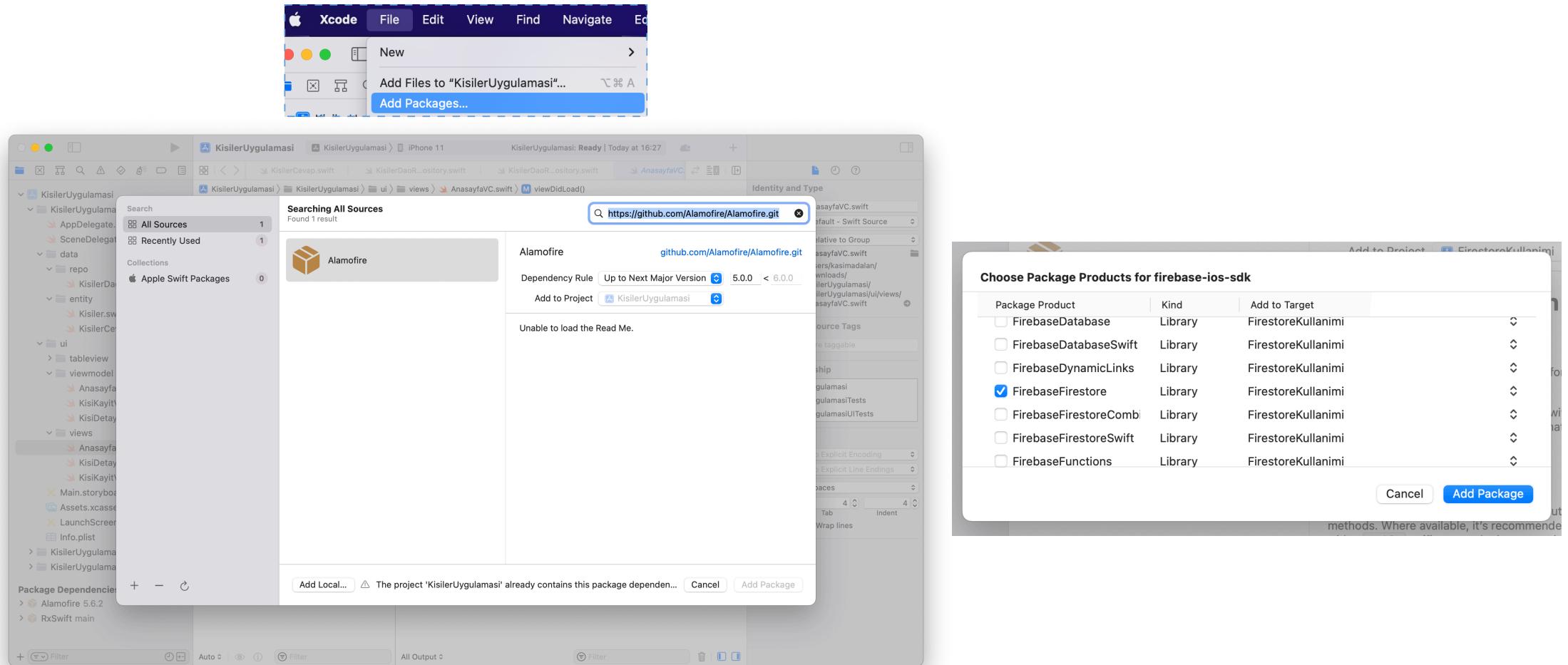
Kişiler Uygulaması

Firestore

Kişiler Uygulaması



Firebase Kurulum



<https://github.com/firebase/firebase-ios-sdk>

Query results



Document ID	kisi_ad	kisi_id	kisi_tel
DoDf7rgv25n51NuC3UQn	"Ali"	""	"4444"
ZQx0AkM4LqUpieZlpZi8	"Ece"	""	"6666"

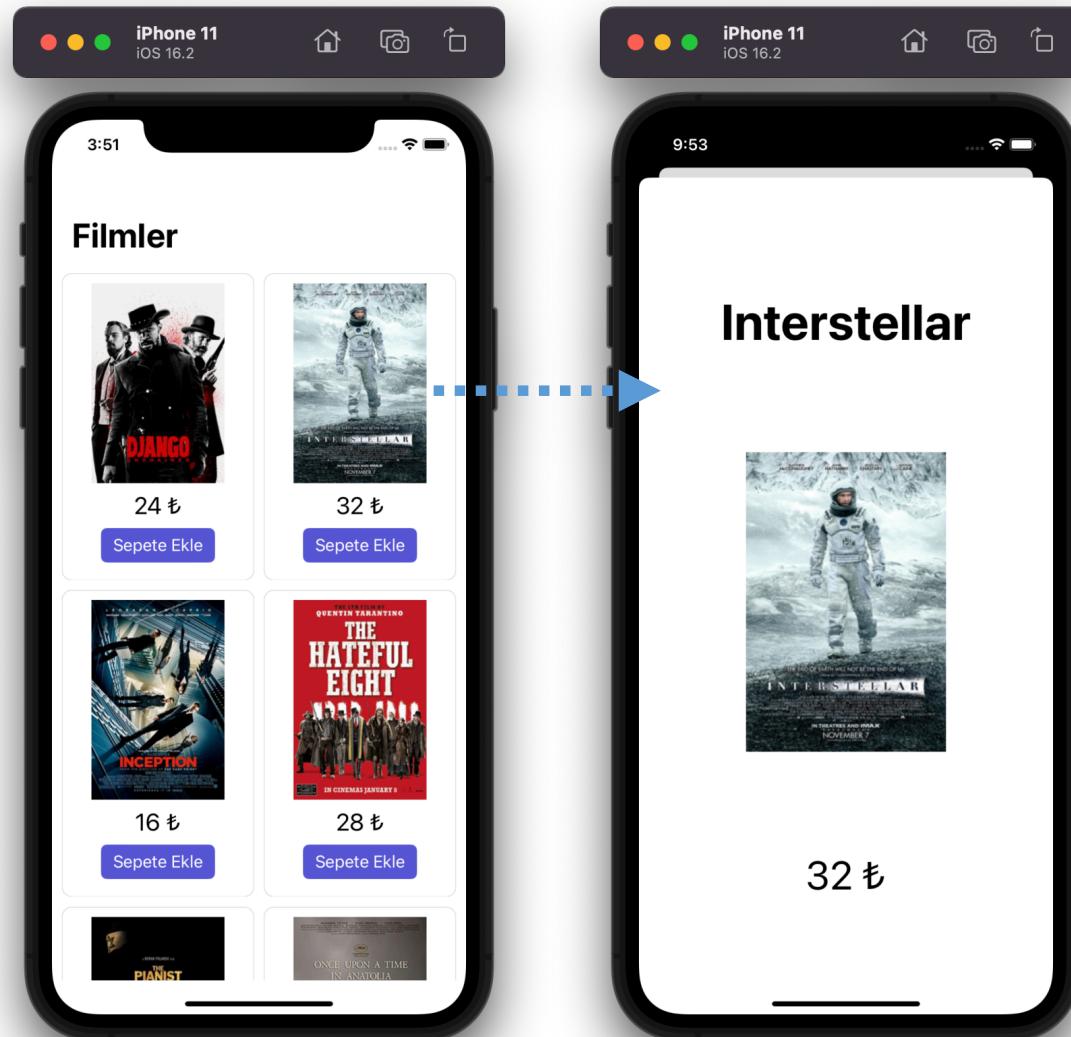
Items per page: 50 ▾ 1 - 2 of 2 ⏪ ⏩

```
class Kisiler {  
    var kisi_id:String?  
    var kisi_ad:String?  
    var kisi_tel:String?
```

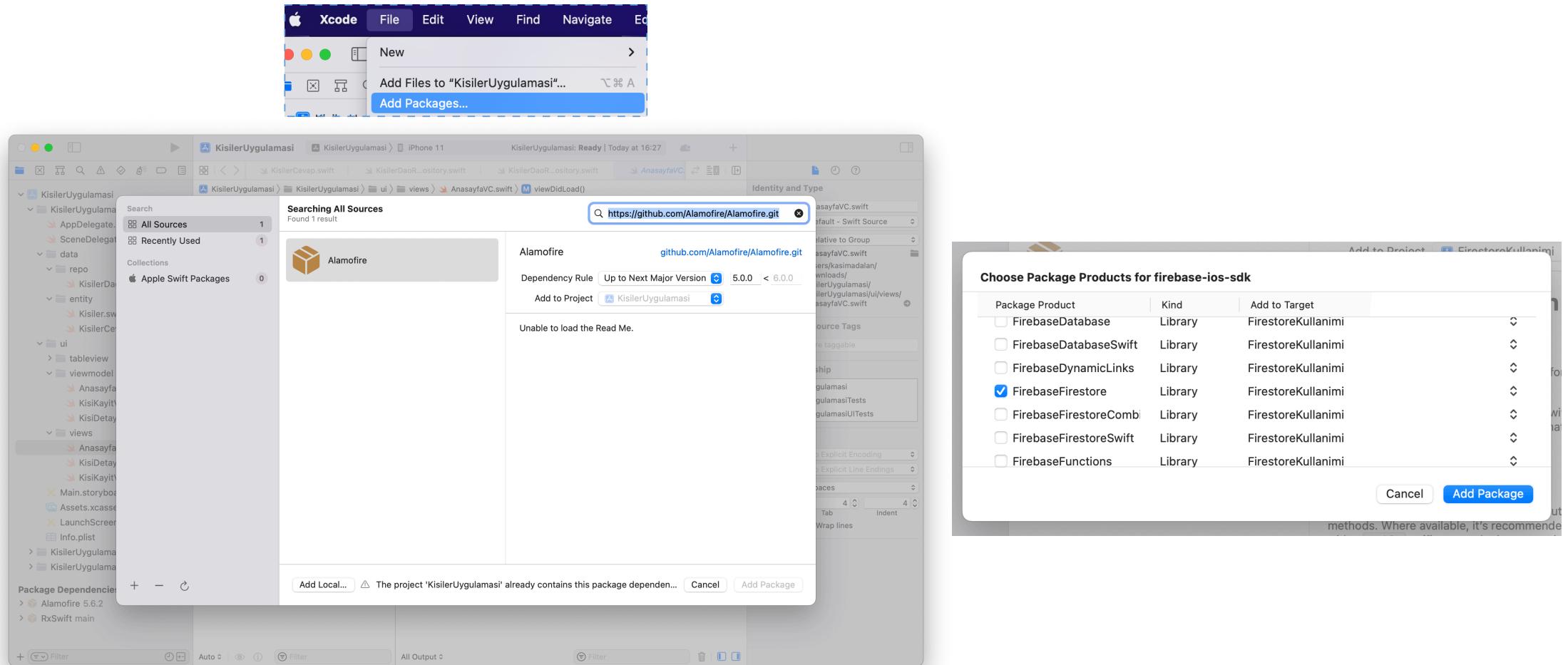
Filmler Uygulaması

Firestore

Film Uygulaması



Firebase Kurulum



<https://github.com/firebase/firebase-ios-sdk>

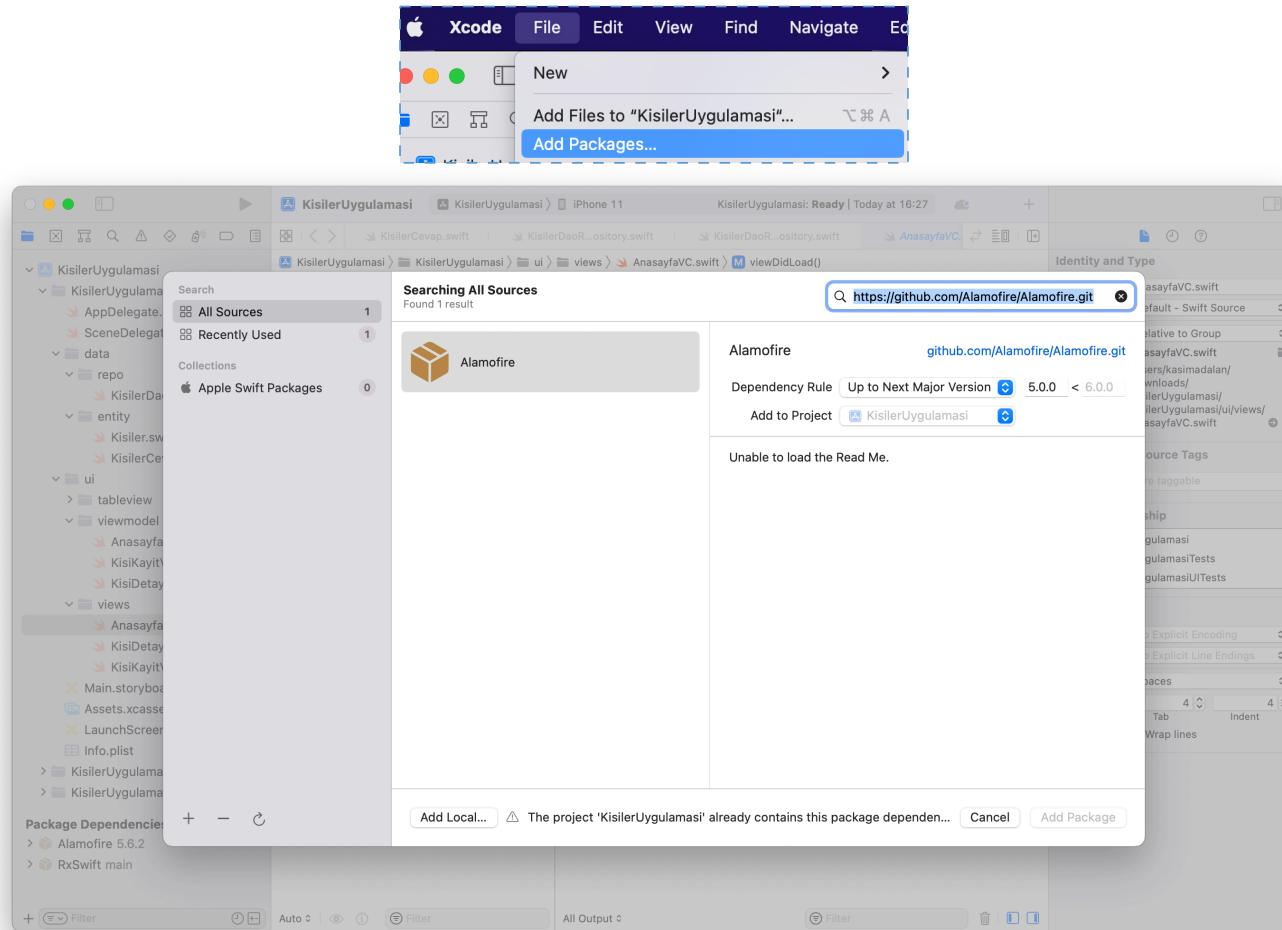
internet izni

The screenshot shows the Xcode Property List Editor with the following details:

File Path: KisilerUygulamasi > KisilerUygulamasi > Info.plist > No Selection

Key	Type	Value
Information Property List	Dictionary	(2 items)
App Transport Security Settings	Dictionary	(1 item)
Allow Arbitrary Loads	Boolean	YES
Application Scene Manifest	Dictionary	(2 items)

Kingfisher Kurulum



<https://github.com/onevcat/Kingfisher.git>

Query results



Document ID	ad	fiyat	id	resim
A4URtX9rNE0m0acbJDWe	"Inception"	16	""	"inception.png"
EszaHVdHibjRmbd0mHoB	"The Hateful Eight"	28	""	"thehatefuleight.png"
TZK76YSboenW9IQmhGMJ	"The Pianist"	18	""	"thepianist.png"
YCX2m9CHgHSnEwurGnLg	"Anadoluda"	10	""	"anadoluda.png"
dkzqxG0aPVEnRlfNbrJJ	"Interstellar"	32	""	"interstellar.png"
mhRv7UMqLHgHEELTRNUX	"Django"	24	""	"django.png"

```

class Filmler {
    var id:String?
    var ad:String?
    var resim:String?
    var fiyat:Int?

```

Data Oluşturma

- Firestore import veya export ücretli olduğu için pratik şekilde verilerimizi kayıt yaparak oluşturuyoruz.

```
let db = Firestore.firestore()
let collectionFilmler = db.collection("Filmler")

let f1:[String:Any] = ["id": "", "ad": "Django", "resim": "django.png", "fiyat": 24]
let f2:[String:Any] = ["id": "", "ad": "Interstellar", "resim": "interstellar.png", "fiyat": 32]
let f3:[String:Any] = ["id": "", "ad": "Inception", "resim": "inception.png", "fiyat": 16]
let f4:[String:Any] = ["id": "", "ad": "The Hateful Eight", "resim": "thehatefuleight.png", "fiyat": 28]
let f5:[String:Any] = ["id": "", "ad": "The Pianist", "resim": "thepianist.png", "fiyat": 18]
let f6:[String:Any] = ["id": "", "ad": "Anadoluda", "resim": "anadoluda.png", "fiyat": 10]

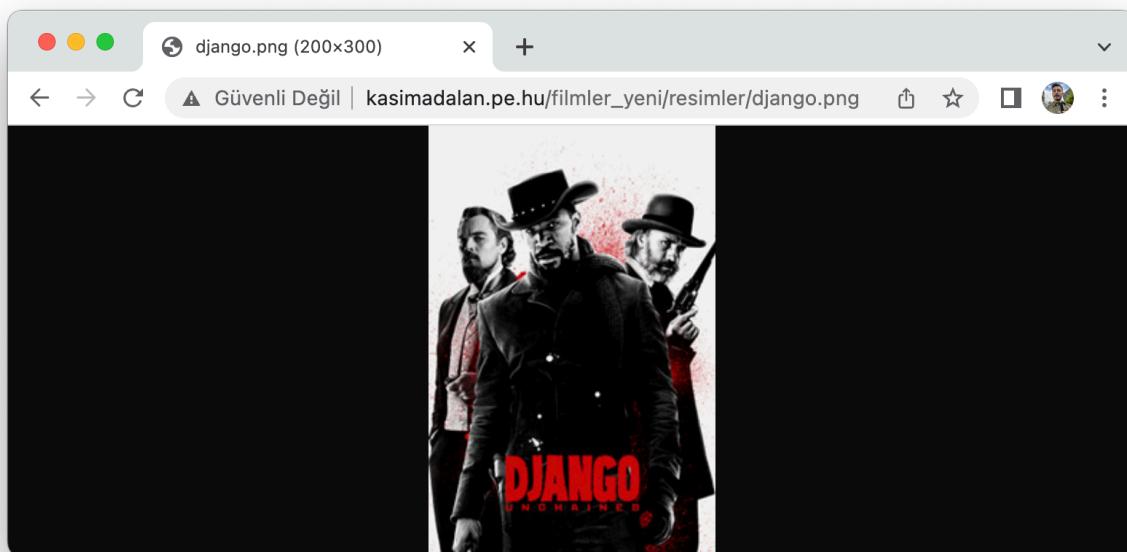
collectionFilmler.document().setData(f1)
collectionFilmler.document().setData(f2)
collectionFilmler.document().setData(f3)
collectionFilmler.document().setData(f4)
collectionFilmler.document().setData(f5)
collectionFilmler.document().setData(f6)
```

Resimleri Görüntüleme

`http://kasimadalan.pe.hu/filmler_yeni/resimler/django.png`

Base Url altında yer alan resimler

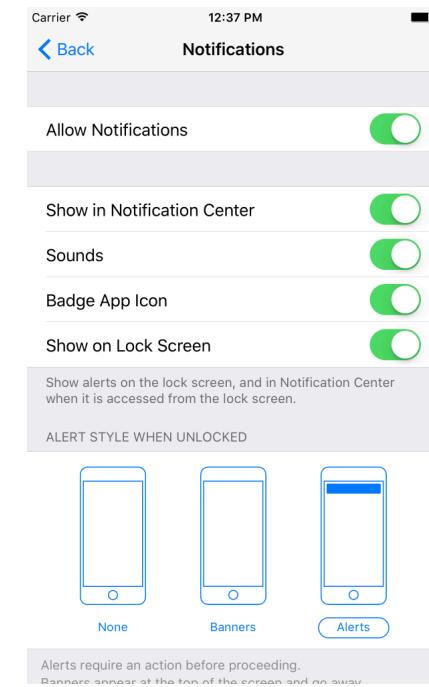
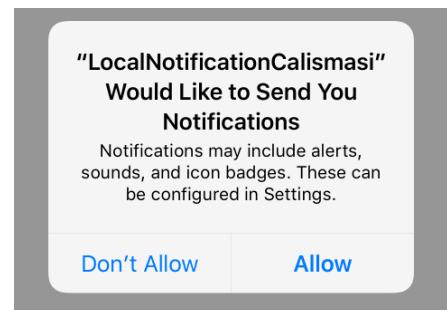
Name ↑
 anadoluda.png
 django.png
 inception.png
 interstellar.png
 thehatefuleight.png
 thepianist.png



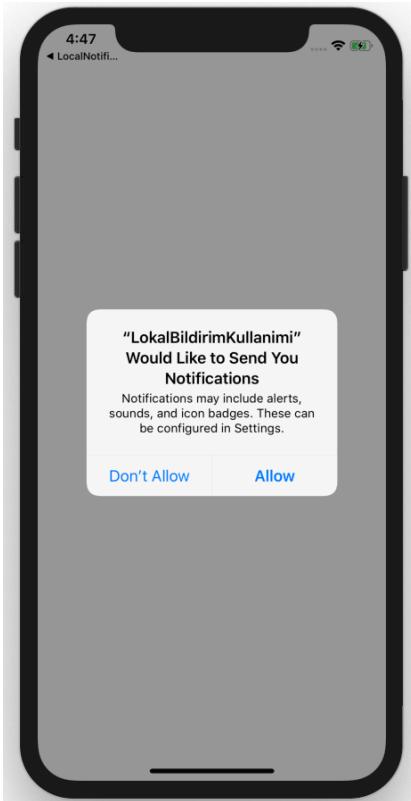
Notification - Bildirim

Uygulama İzni Alma

- Uygulama izni kullanıcıya uygulama içinde sorulabilir.
- Uygulama izni alınırsa ayarlardaki uygulama ile ilgili panelden izin verilmiş olunur.
- Bir defa izin alınırsa tekrar izin almaya gerek yoktur.
- *AppDelegate didFinishLaunchingWithOptions()* metodu içerisinde de izin kaldırılabilir.



Uygulama İzni Alma



```
import UIKit

import UserNotifications
//Gerekli framework

class ViewController: UIViewController {

    var izinKontrol:Bool = false
    //Kullanıcı bilgisi özelliğine iznini takip etmek için kullanılır.

    override func viewDidLoad() {
        super.viewDidLoad()

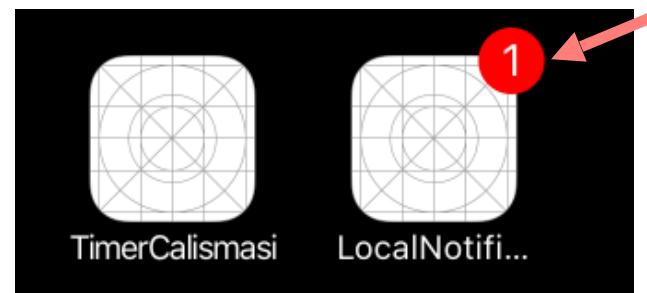
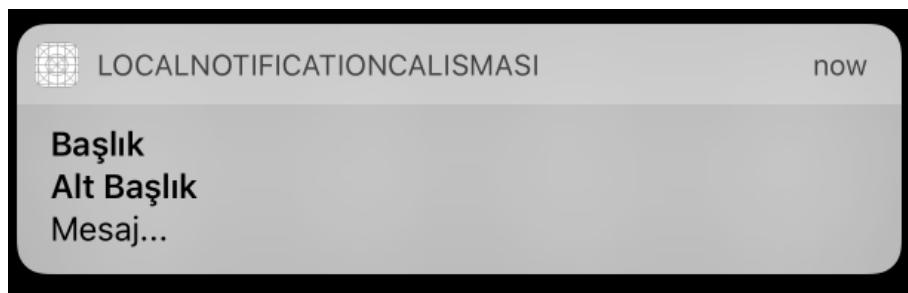
        //izin kontrolü alert,sound,badge özelliklerine onay verilir.
        UNUserNotificationCenter
            .current()
            .requestAuthorization(options: [.alert,.sound,.badge],completionHandler: {
                (granted,error) in

                self.izinKontrol = granted
                //izin verildiyse burası çalışır.
                //granted : true veya false döner.

                if granted {
                    print("İzin alma işlemi başarılı")
                }else{
                    print("İzin alma işlemi başarısız")
                }
            })
    }
}
```

Bildirim İçeriği

```
//Bildirim içeriğinin hazırlanması
let icerik = UNMutableNotificationContent()
icerik.title = "Başlık"
icerik.subtitle = "Alt Başlık"
icerik.body = "Mesaj"
icerik.badge = 1 //uygulama icon'unun üzerinde beliren rozet.
icerik.sound = UNNotificationSound.default //bilidirim sesi.
```



Badge

Bildirimin Zamansal Tetiklenmesi

```
//Bildirimin zamansal olarak tetiklenmesi
let tetikleme = UNTimeIntervalNotificationTrigger(
    timeInterval: 10, //Kaç saniye sonra
    repeats: false) // tekrarlanma ifadesi
```

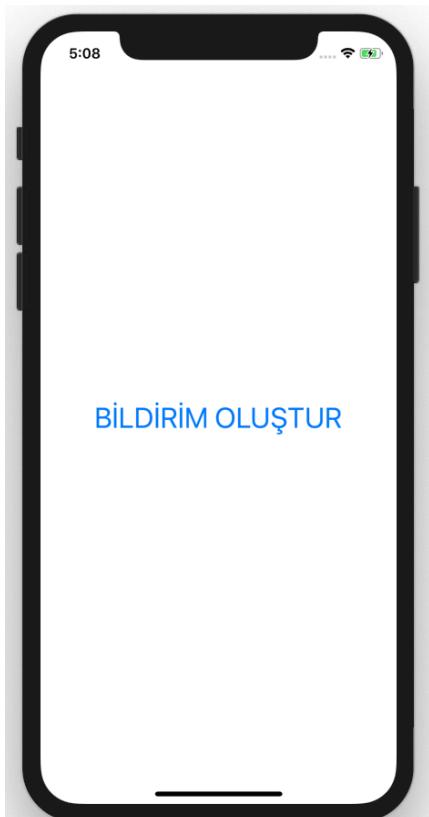
Bildirimin İsteği Oluşturma

```
//Bildirim içeriğinin ve zamansal tetiklenmenin isteğe dönüşümü
let bildirimIstegi = UNNotificationRequest(
    identifier: "Bildirim Calismasi", //Tanımlayıcı
    content: icerik, // içerik
    trigger: tetikleme // zamansal tetikleme
)
```

Bildirimin Çalıştırılması

```
//Bildirimi bildirim merkezine eklenmesi.
UNUserNotificationCenter.current().add(
    bildirimIstegi, completionHandler: nil)
```

UYGULAMA



```
import UIKit

import UserNotifications
//Gerekli framework

class ViewController: UIViewController {

    var izinKontrol:Bool = false
    //Kullanıcıın bildirim özelliğine iznini takip etmek için kullanılır.

    override func viewDidLoad() {
        super.viewDidLoad()

        //izin kontrolü alert,sound,badge özelliklerine onay verilir.
        UNUserNotificationCenter
            .current()
            .requestAuthorization(options: [.alert,.sound,.badge],completionHandler: {
                (granted,error) in

                self.izinKontrol = granted
                //izin verildiyse burası çalışır.
                //granted : true veya false döner.

                if granted {
                    print("İzin alma işlemi başarılı")
                }else{
                    print("İzin alma işlemi başarısız")
                }
            })
    }
}
```

```
@IBAction func bildirimOlustur(_ sender: Any) {  
  
    //izin verildiyse bildirim işlemi yapılsın  
    if izinKontrol {  
  
        //Bildirim içeriğinin hazırlanması  
        let icerik = UNMutableNotificationContent()  
        icerik.title = "Başlık"  
        icerik.subtitle = "Alt Başlık"  
        icerik.body = "Mesaj"  
        icerik.badge = 1 //uygulama icon'unun üzerinde beliren rozet.  
        icerik.sound = UNNotificationSound.default //bilidirim sesi.  
  
        //Bildirimin zamansal olarak tetiklenmesi  
        let tetikleme = UNTimeIntervalNotificationTrigger(  
            timeInterval: 10, //Kaç saniye sonra  
            repeats: false) // tekrarlanma ifadesi  
  
        //Bildirim içeriğinin ve zamansal tetiklenmenin istege dönüşümü  
        let bildirimIstegi = UNNotificationRequest(  
            identifier: "Bildirim Calismasi", //Tanımlayıcı  
            content: icerik, // içeriğin  
            trigger: tetikleme // zamansal tetikleme  
        )  
  
        //Bildirimi bildirim merkezine eklenmesi.  
        UNUserNotificationCenter.current().add(  
            bildirimIstegi, completionHandler: nil)  
    }  
}
```

İçerik

Tetikleme

**İstek
Oluşturma**

**Bildirim
Oluşumu**

Bildirimin Uygulama Önplanda iken Çalışır Olması

- Bildirim kodlamasının ilk hali ile sadece arkaplanda çalışır.
- Ön planda çalışır hale getirmek için aşağıdaki kodlama yapılmalıdır.

```
extension ViewController:UNUserNotificationCenterDelegate{  
    func userNotificationCenter(_ center: UNUserNotificationCenter, willPresent notification:  
        UNNotification, withCompletionHandler completionHandler: @escaping  
        (UNNotificationPresentationOptions) -> Void) {  
  
        completionHandler([.banner,.sound,.badge])  
    }  
}
```

```
import UIKit  
  
import UserNotifications  
//Gerekli framework  
  
class ViewController: UIViewController {  
  
    var izinKontrol:Bool = false  
    //Kullanıcın bildirim özelliğine iznini takip etmek için kullanılır.  
  
    override func viewDidLoad() {  
        super.viewDidLoad()  
  
        UNUserNotificationCenter.current().delegate = self  
        //Ön planda çalışması için extension bağlanır.
```

Bildirime Tıklama ve Badge Sıfırlama

```
extension ViewController:UNUserNotificationCenterDelegate {
    func userNotificationCenter(_ center: UNUserNotificationCenter, willPresent notification: UNNotification, withCompletionHandler completionHandler: @escaping (UNNotificationPresentationOptions) -> Void) {
        completionHandler([.banner,.sound,.badge])
    }

    func userNotificationCenter(_ center: UNUserNotificationCenter, didReceive response: UNNotificationResponse,
    withCompletionHandler completionHandler: @escaping () -> Void) {
        let app = UIApplication.shared

        if(app.applicationState == .active){
            print("Önplandayken Bildirim Tıklandı")
            app.applicationIconBadgeNumber = 0//Badge sıfırlama
        }

        if(app.applicationState == .inactive){
            print("Arkaplandayken Bildirim Tıklandı")
            app.applicationIconBadgeNumber = 0//Badge sıfırlama
        }

        completionHandler()
    }
}
```

```
import UIKit

import UserNotifications
//Gerekli framework

class ViewController: UIViewController {

    var izinKontrol:Bool = false
    //Kullanıcının bildirim özelliğine iznini takip etmek için kullanılır.

    override func viewDidLoad() {
        super.viewDidLoad()

        UNUserNotificationCenter.current().delegate = self
        //Ön planda çalışması için extension bağlanır.
```

Tekrarlı Bildirim

Zamansal Tekrarlar (Saniye Türüyle)

- Tekrarlanmalı ifadeler minimum 60 sn yani 1 dk olmalıdır.
- Çalıştırıldıktan 60 sn sonra ilk bildirimi oluşturur ve 60 sn aralıklarla tekrarlamaya başlar.

```
//Bildirimin zamansal olarak tetiklenmesi
let tetikleme = UNTimeIntervalNotificationTrigger(
    timeInterval: 60, //Kaç saniye sonra
    repeats: true)   // tekrarlanma ifadesi
```

Zamansal Tekrarlar (Saniye Türüyle)

- Çalıştırıldıkten 3600 sn sonra yani 1 saat sonra ilk bildirimi oluşturur ve 1 saat aralıklarla tekrarlamaya başlar.

```
//Bildirimin zamansal olarak tetiklenmesi
let tetikleme = UNTimeIntervalNotificationTrigger(
    timeInterval: 3600, //Kaç saniye sonra
    repeats: true)    // tekrarlanma ifadesi
```

Zamansal Tekrarlar (Saniye Türüyle)

- Çalıştırıldıkten 86400 sn sonra yani 1 gün sonra ilk bildirimi oluşturur ve 1 gün aralıklarla tekrarlamaya başlar.

```
//Bildirimin zamansal olarak tetiklenmesi
let tetikleme = UNTimeIntervalNotificationTrigger(
    timeInterval: 86400, //Kaç saniye sonra
    repeats: true)     // tekrarlanma ifadesi
```

Zamansal Tekrarlar (Tarih Türüyle)

- Belirtilen tarih gerçekleştiğinde ilk bildirim oluşur ve tarihe göre tekrarlama yapar.
- Aşağıdaki örnek yılda bir kere

```
var date = DateComponents()  
date.day = 29  
date.month = 4  
date.year = 2019  
date.hour = 15  
date.minute = 32  
  
let tetikleme = UNCalendarNotificationTrigger(dateMatching: date, repeats: true)
```

Zamansal Tekrarlar (Tarih Türüyle)

- Belirtilen tarih gerçekleştiğinde ilk bildirim oluşur ve tarihe göre tekrarlama yapar.
- Aşağıdaki her 8:30 olduğunda çalışır.

```
var date = DateComponents()  
//date.day = 29  
//date.month = 4  
//date.year = 2019  
date.hour = 8  
date.minute = 30  
  
let tetikleme = UNCalendarNotificationTrigger(dateMatching: date, repeats: true)
```

Lokasyon İşlemleri

Core Location Framework

Core Location FrameWork

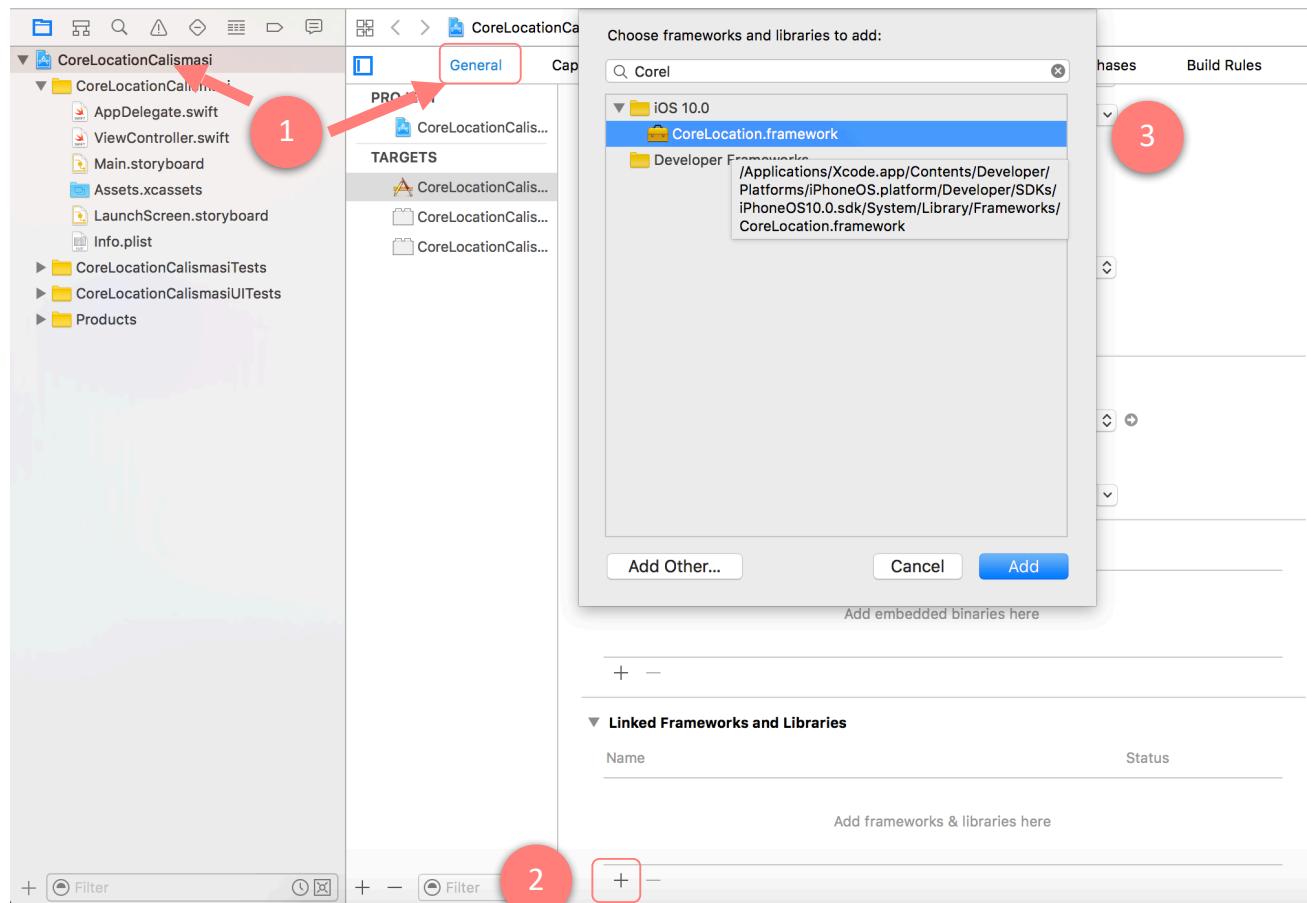
- Core Location framework lokasyon bilgilerinin işlendiği sınıfıtır.
- Enlem, Boylam,Hız vb bilgileri bu sınıfıtan alabiliriz.
- Harici bir framework olduğu için Xcode projesine eklenmelidir.

Enlem : 37.33500926

Boylam : -122.03272188

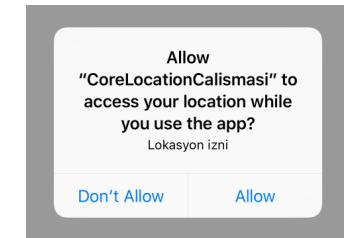
Hız : 7.74

Core Location Framework Xcode Ekleme



Lokasyon Kullanma İzni Alma

- Info.plist içerisinde izin alınmaktadır.
- Yeni bir kayıt açıp Key : **Privacy – Location When In Use Usage Description**
- Value : açıklama yazılır ve kaydedilir.



Key	Type	Value
Localization native development region	String	en
Privacy - Location When In Use Usage Description	String	Lokasyon izni
Executable file	String	\$(EXECUTABLE_NAME)
Bundle identifier	String	\$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	\$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle versions string, short	String	1.0
Bundle version	String	1
Application requires iPhone environment	Boolean	YES
Launch screen interface file base name	String	LaunchScreen
Main storyboard file base name	String	Main
Required device capabilities	Array	(1 item)
Supported interface orientations	Array	(3 items)
Supported interface orientations (iPad)	Array	(4 items)

Core Location Kurulum Kodları

```
import UIKit
import CoreLocation
//CoreLocation Framework kullanılır.

class ViewController: UIViewController {
    var locationManager: CLLocationManager = CLLocationManager()
    //CLLocationManager nesnesi ile lokasyon ayarları yapılır.
```

Lokasyon Ayarları

```
var locationManager: CLLocationManager = CLLocationManager()  
//CLLocationManager nesnesi ile lokasyon ayarları yapılır.
```

```
locationManager.desiredAccuracy = kCLLocationAccuracyBest  
//GPS bilgisinin kesinliği hakkında parametre.  
locationManager.delegate = self  
//Manager bu sınıfı bağlanıyor.  
locationManager.requestWhenInUseAuthorization()  
//lokasyon izni alınması sağlanıyor.  
locationManager.startUpdatingLocation()  
//lokasyon işlemi başlıyor.
```

Bu metod çalıştığı anda izin uyarısı çıkar ve lokasyon işlemi başlar.

Lokasyon Bilgilerinin Alınması

```
extension ViewController:CLLocationManagerDelegate {
    func locationManager(_ manager: CLLocationManager, didUpdateLocations locations: [CLLocation]) {
        let sonKonum: CLLocation = locations[locations.count - 1]

        enlemLabel.text = "Enlem : \(sonKonum.coordinate.latitude)"
        boylamLabel.text = "Boylam : \(sonKonum.coordinate.longitude)"
        hizLabel.text = "Hız : \(sonKonum.speed)"
    }
}
```

```
import UIKit
import CoreLocation
//CoreLocation Framework kullanılır.

class ViewController: UIViewController {
    var locationManager: CLLocationManager = CLLocationManager()
    //CLLocationManager nesnesi ile lokasyon ayarları yapılır.

    @IBOutlet weak var hizLabel: UILabel!
    @IBOutlet weak var boylamLabel: UILabel!
    @IBOutlet weak var enlemLabel: UILabel!

    override func viewDidLoad() {
        super.viewDidLoad()

        locationManager.desiredAccuracy = kCLLocationAccuracyBest
        //GPS bilgisinin kesinliği hakkında parametre.
        locationManager.delegate = self
        //Manager bu sınıfı bağlanıyor.
        locationManager.requestWhenInUseAuthorization()
        //lokasyon izni alınması sağlanıyor.
        locationManager.startUpdatingLocation()
        //lokasyon işlemi başlıyor.

    }
}

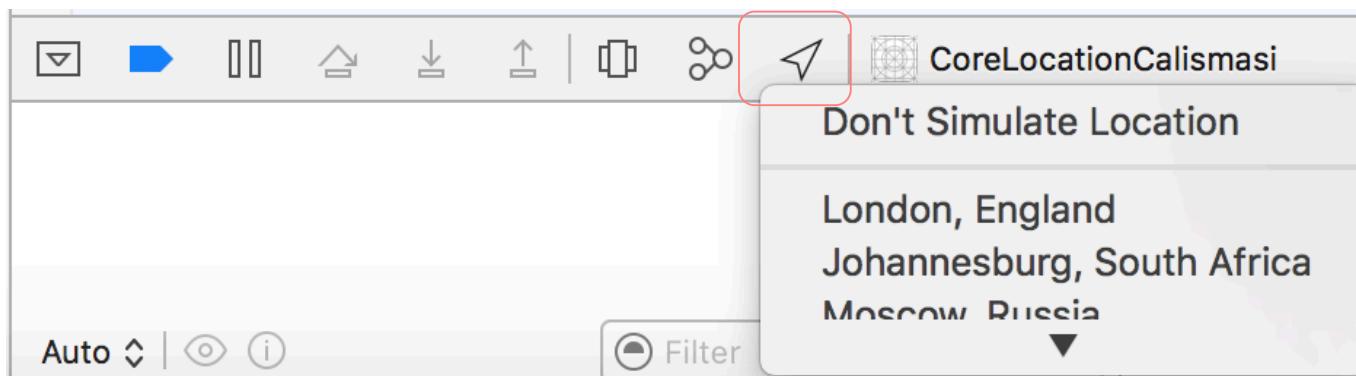
extension ViewController:CLLocationManagerDelegate {
    func locationManager(_ manager: CLLocationManager,didUpdateLocations locations: [CLLocation]){
        let sonKonum: CLLocation = locations[locations.count - 1]

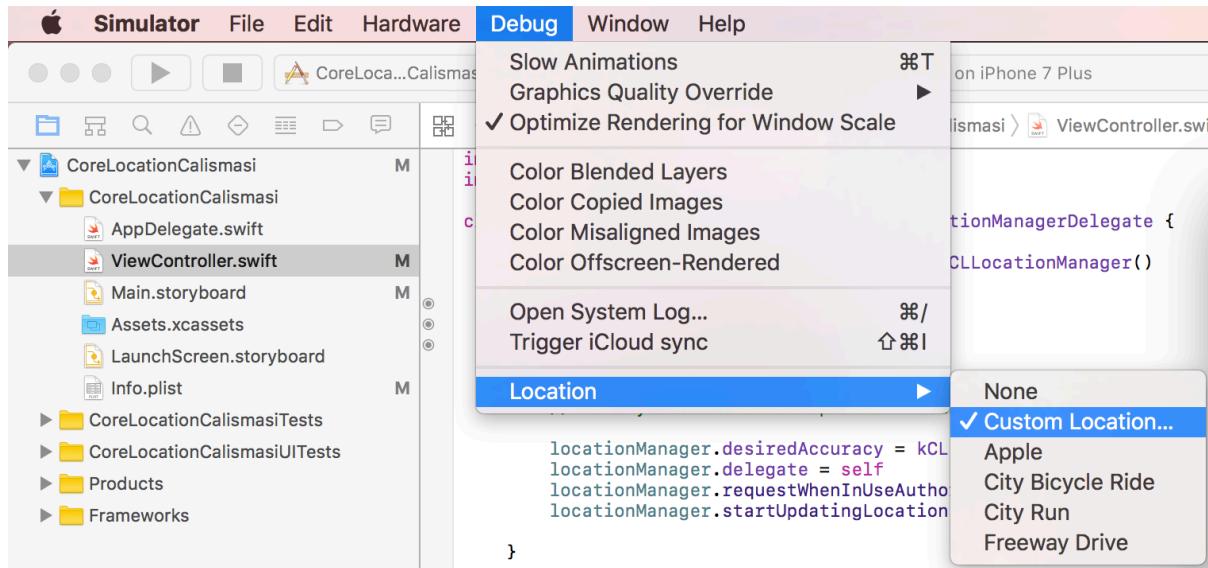
        enlemLabel.text  = "Enlem : \(sonKonum.coordinate.latitude)"
        boylamLabel.text = "Boylam : \(sonKonum.coordinate.longitude)"
        hizLabel.text    = "Hız     : \(sonKonum.speed)"
    }
}
```

Simulator Üzerinde Lokasyon İşlemleri.

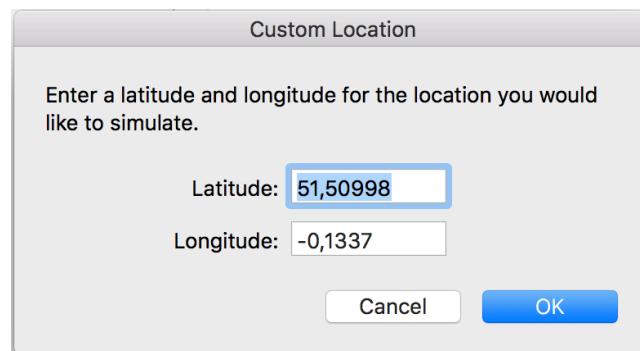
- Simülasyon ve Xcode lokasyon belirtmek için hazır yapılar sunmaktadır.
- Uygulama çalıştırıldıktan sonra lokasyon verilerini simulatöre gönderebiliriz.

1. Yöntem Hazır Konum Atma





2. Custom Konum Atma



Otomatik Konum Değişimi

City Biycle Ride : Bisiklet sürüşü

City Run : Koşu

Freeway Drive : Araba sürüşü

Map Kit Framework

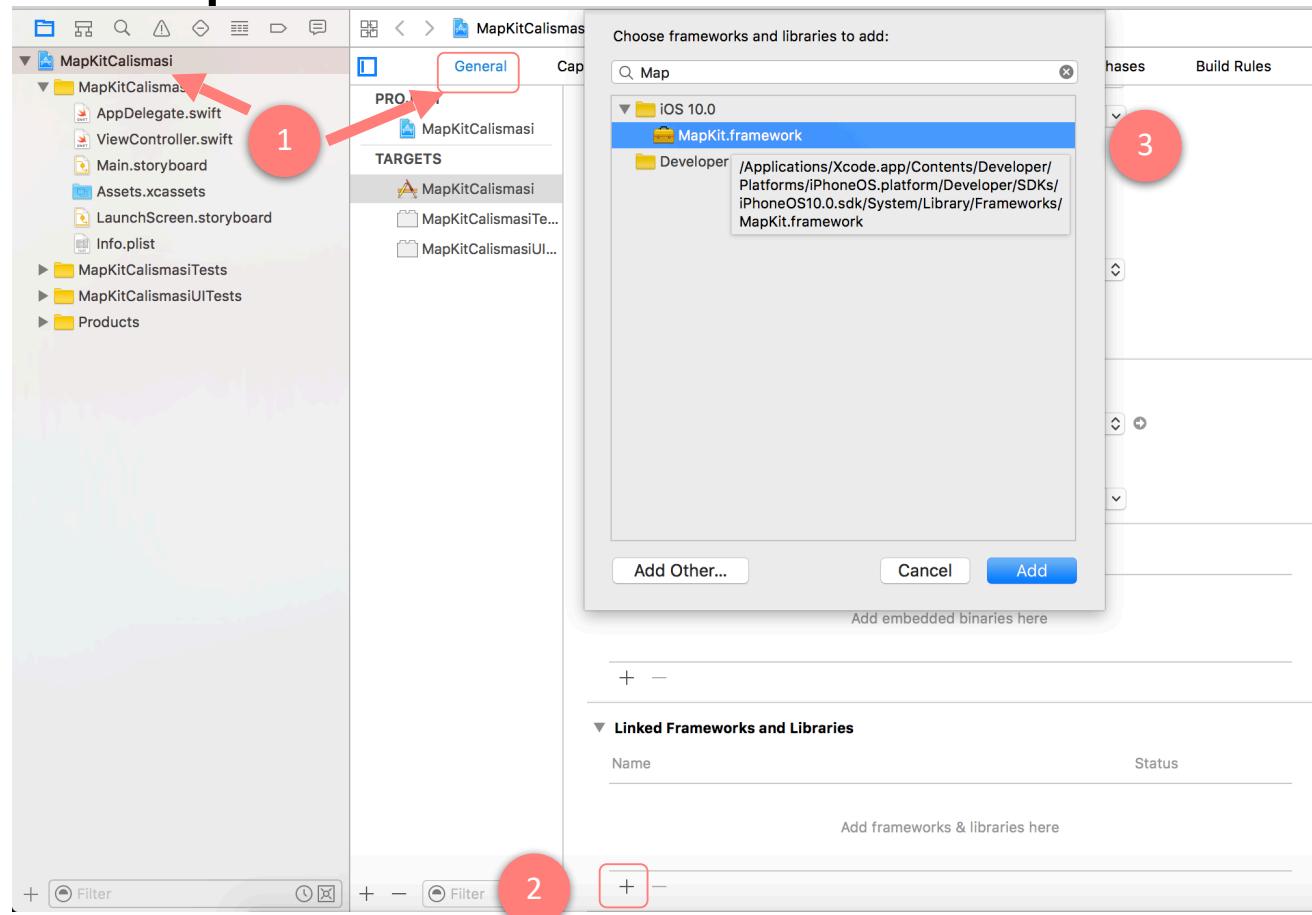
Map Kit FrameWork

- Map Kit harita üzerinde işlemler yapabildiğimiz bir yapıdır.
- Bu framework apple tarafından sunulmaktadır.
- Harici harita framework'leri vardır. Örnek : Google Maps API vb.
- Bu framework kullanılabilmesi için MapKit Sınıfı import edilmelidir.

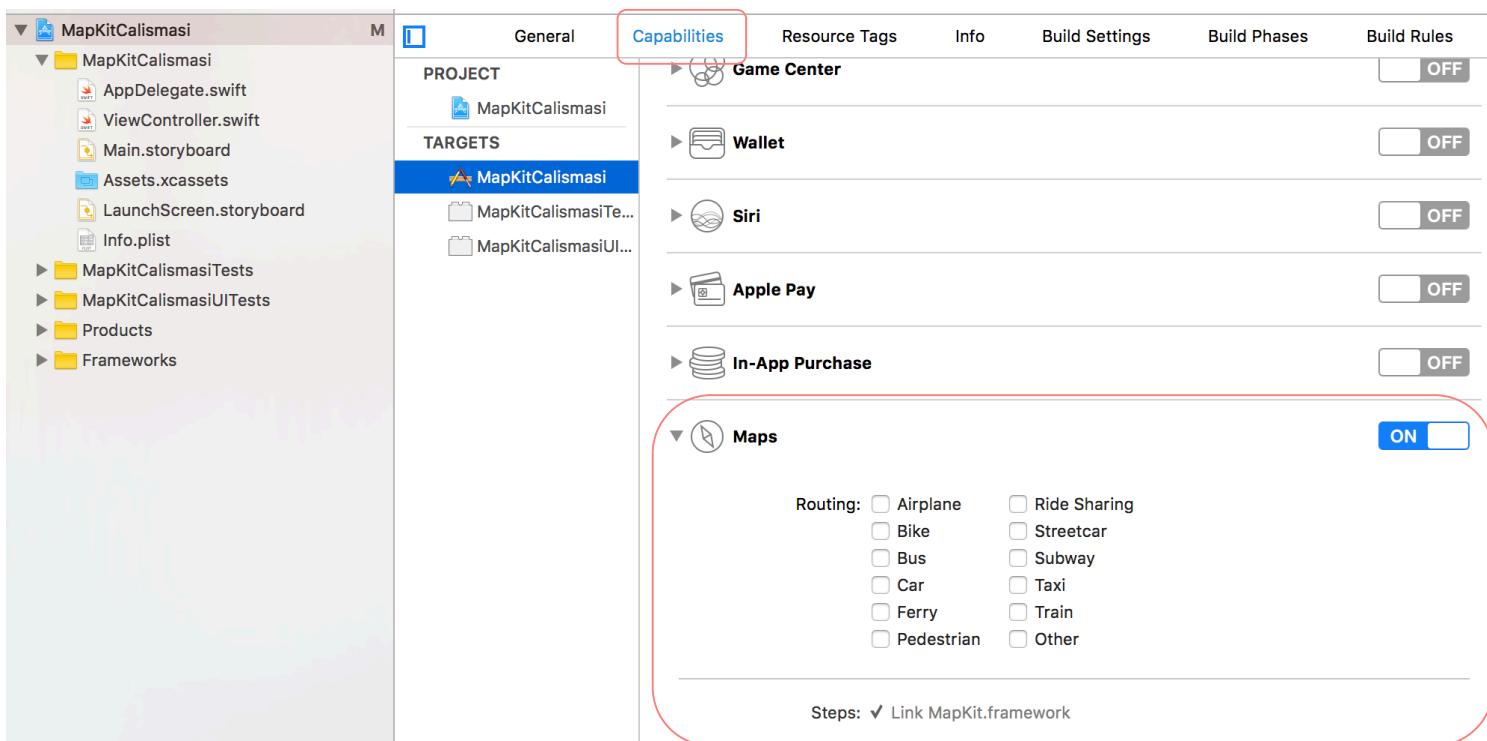
`import MapKit`



Map Kit Framework Xcode Ekleme



Map Kit FrameWork Ekleme 2. Yöntem

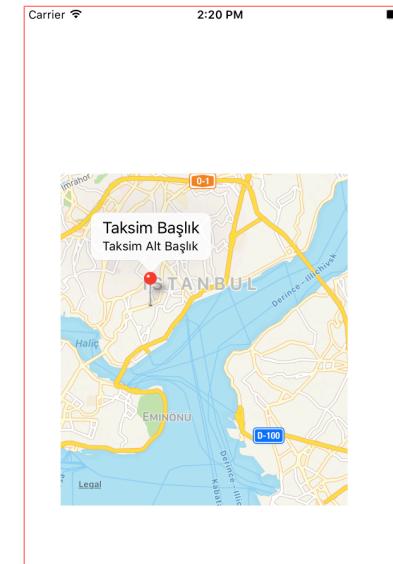


Harita Konuma Odaklanması

```
let konum = CLLocationCoordinate2D(latitude: 41.0370063, longitude: 28.982903)
//konum enlem ve boylam bilgisi
let span = MKCoordinateSpan(latitudeDelta: 0.5, longitudeDelta: 0.5) Bu değer küçülsürse yakınlaşır.
//haritanın yakınlık ayarı.
let bolge = MKCoordinateRegion(center: konum, span: span)
//konum ve spandan oluşan bolge degeri
mapView.setRegion(bolge, animated: true)
//haritaya bu konumun eklenmesi

let pin = MKPointAnnotation()
//konuma pin ekleme
pin.coordinate = konum
//konum eklenmesi
pin.title = "Taksim Başlık"
//başlık
pin.subtitle = "Taksim Alt Başlık"
//alt başlık

mapView.addAnnotation(pin)
```



```
import UIKit
import MapKit

class ViewController: UIViewController {

    @IBOutlet weak var mapView: MKMapView!

    override func viewDidLoad() {
        super.viewDidLoad()

        let konum = CLLocationCoordinate2D(latitude: 41.0370063, longitude: 28.982903)
        //konum enlem ve boylam bilgisi
        let span = MKCoordinateSpan(latitudeDelta: 0.5, longitudeDelta: 0.5)
        //haritanın yakınlık ayarı.
        let bolge = MKCoordinateRegion(center: konum, span: span)
        //konum ve spandan oluşan bolge degeri
        mapView.setRegion(bolge, animated: true)
        //haritaya bu konumun eklenmesi

        let pin = MKPointAnnotation()
        //konuma pin ekleme
        pin.coordinate = konum
        //konum eklenmesi
        pin.title = "Taksim Başlık"
        //başlık
        pin.subtitle = "Taksim Alt Başlık"
        //alt başlık

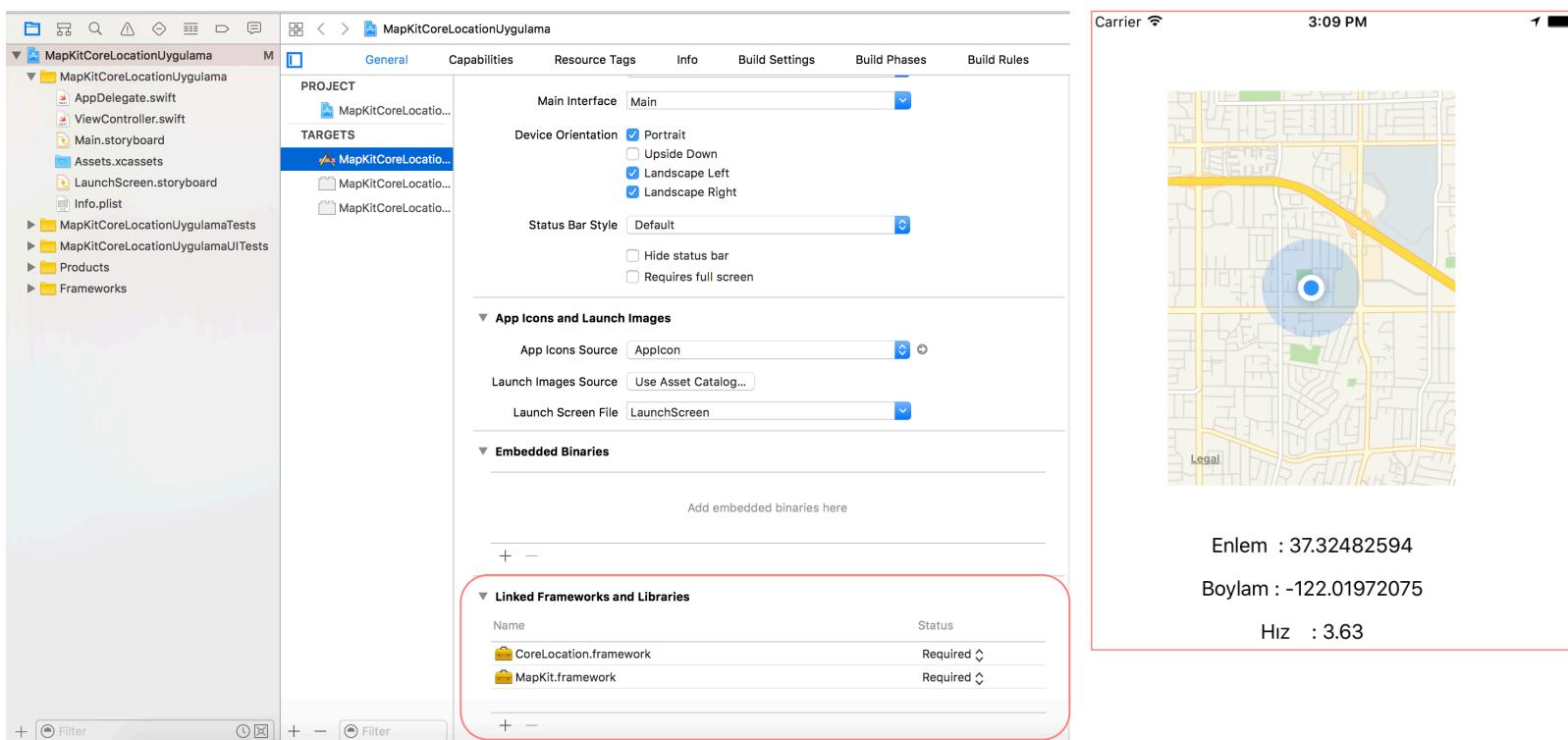
        mapView.addAnnotation(pin)
    }
}
```

Core Location ve Map Kit

Birlikte Kullanımı

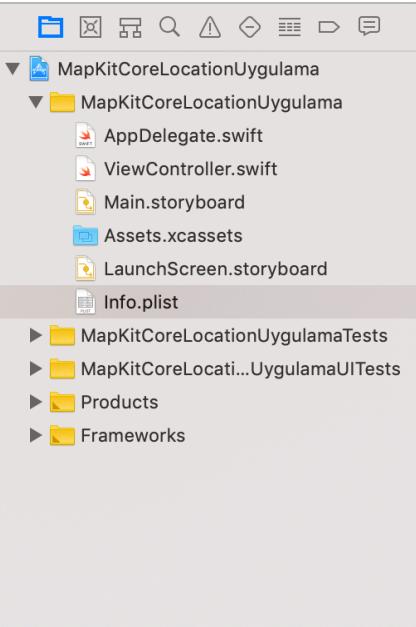
Core Location ve Map Kit Birlikte Kullanılması.

- FrameWorkler Xcode Projesine eklenir.



Lokasyon Kullanma İzni Alma

- Info.plist içerisinde izin alınmaktadır.



The screenshot shows the Xcode interface with the Project Navigator on the left and the Utilities panel on the right. In the Project Navigator, the 'MapKitCoreLocationUygulama' project is selected, showing its structure: 'MapKitCoreLocationUygulama' folder containing 'AppDelegate.swift', 'ViewController.swift', 'Main.storyboard', 'Assets.xcassets', 'LaunchScreen.storyboard', and 'Info.plist'. The 'Info.plist' file is currently selected. In the Utilities panel, the 'File Inspector' tab is active, displaying the contents of the 'Info.plist' file as a table.

Key	Type	Value
▼ Information Property List	Dictionary	(17 items)
Localization native development region	String	en
Privacy - Location Always Usage Description	String	Konunuz sürekli kullanılacaktır.
Executable file	String	\$(EXECUTABLE_NAME)
Bundle identifier	String	\$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	\$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle versions string, short	String	1.0
Bundle version	String	1
Application requires iPhone environment	Boolean	YES
Privacy - Location When In Use Usage Description	String	Konunuz Kullanılacak !!!
► Required background modes	Array	(1 item)
Launch screen interface file base name	String	LaunchScreen
Main storyboard file base name	String	Main
► Required device capabilities	Array	(1 item)
► Supported interface orientations	Array	(3 items)
► Supported interface orientations (iPad)	Array	(4 items)

```

import UIKit
import MapKit
import CoreLocation

class ViewController: UIViewController {
    @IBOutlet weak var mapView: MKMapView!
    @IBOutlet weak var enlemLabel: UILabel!
    @IBOutlet weak var boylamLabel: UILabel!
    @IBOutlet weak var hizLabel: UILabel!

    var locationManager:CLLocationManager = CLLocationManager()

    override func viewDidLoad() {
        super.viewDidLoad()
        locationManager.desiredAccuracy = kCLLocationAccuracyBest
        locationManager.delegate = self
        locationManager.requestWhenInUseAuthorization()
        locationManager.startUpdatingLocation()
    }
}

extension ViewController:CLLocationManagerDelegate{
    func locationManager(_ manager: CLLocationManager, didUpdateLocations locations: [CLLocation]) {
        let sonKonum:CLLocation = locations[locations.count-1]

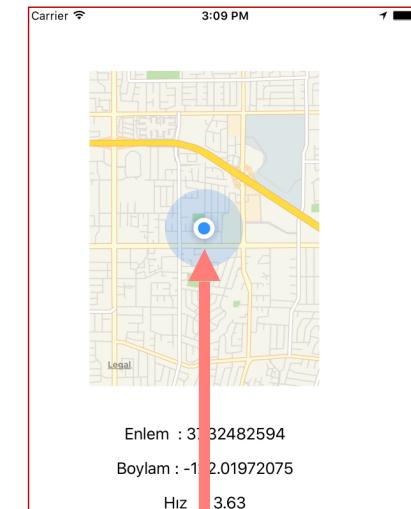
        enlemLabel.text = "Enlem : \(sonKonum.coordinate.latitude)"
        boylamLabel.text = "Boylam : \(sonKonum.coordinate.longitude)"
        hizLabel.text = "Hız : \(sonKonum.speed)"

        let konum = CLLocationCoordinate2D(latitude: sonKonum.coordinate.latitude, longitude: sonKonum.coordinate.longitude)
        let span = MKCoordinateSpan(latitudeDelta: 0.03, longitudeDelta: 0.03)
        let bolge = MKCoordinateRegion(center: konum, span: span)
        mapView.setRegion(bolge, animated: true)

        mapView.showsUserLocation = true
    }
}

```

Harita üzerinde anlık konumu gösterir.



Teşekkürler...



kasım-adalan



kasimadalan@gmail.com



kasimadalan