

PG5600
iOS programmering
Forelesning 4

Forrige gang

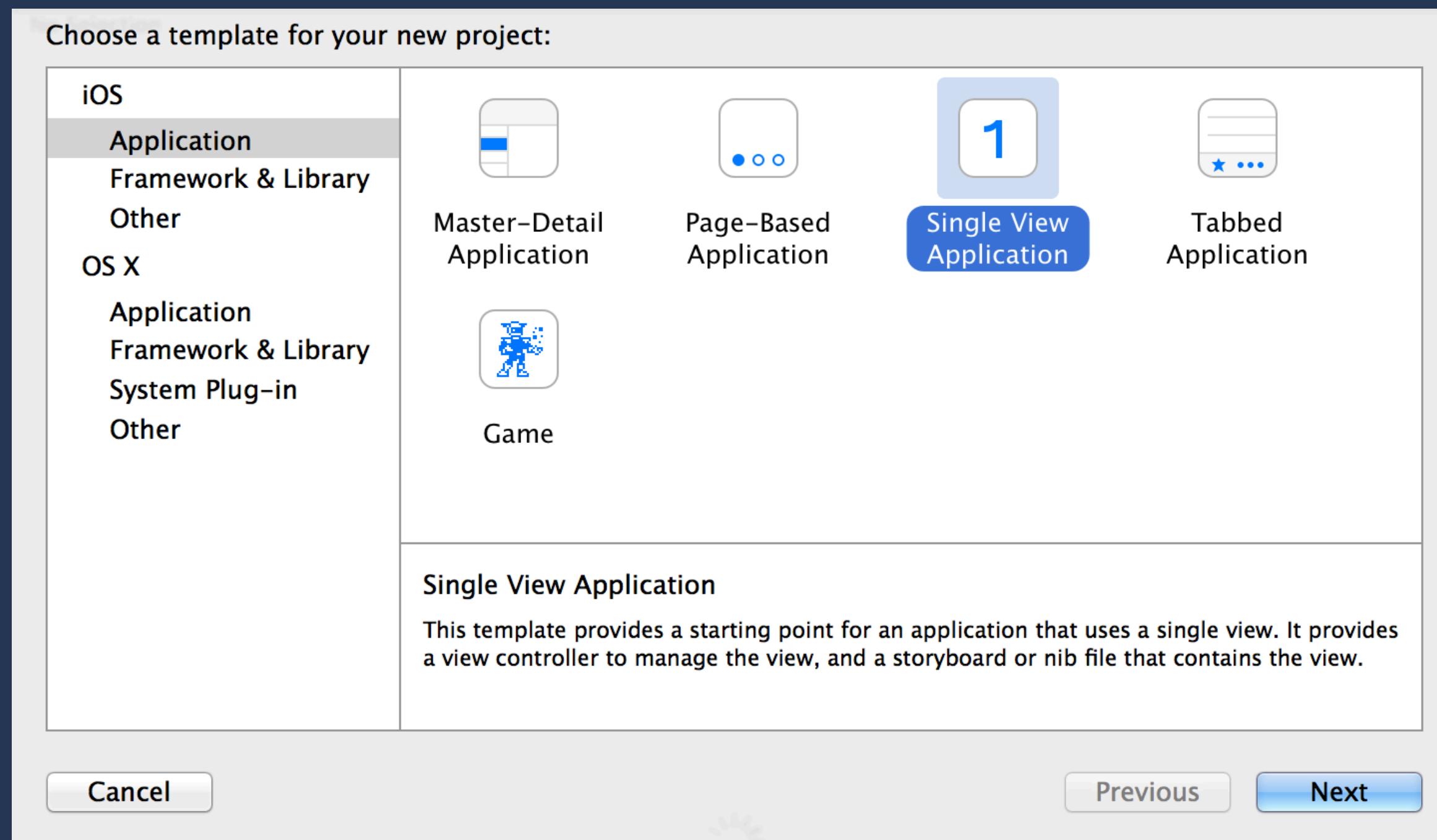
- Subscripts, Kontrutører og Arv
- deinit og ARC
- Optionals og Optional chaining
- Guard
- Type casting og Nested types
- Protocols
- Extentions

Agenda

- Sette opp et nytt iOS prosjekt i XCode
- Komponentene i en iOS app
- Launch flow
- Application lifecycle
- MVC
- UIView og UIViewController

Sette opp et nytt iOS prosjekt i XCode

Single View = tilnærmet tomt prosjekt



Choose options for your new project:

Product Name: Awesomeapp

Organization Name: Westerdals

Organization Identifier: no.westerdals

Bundle Identifier: no.westerdals.Awesomeapp

Language: Swift

Devices: Universal

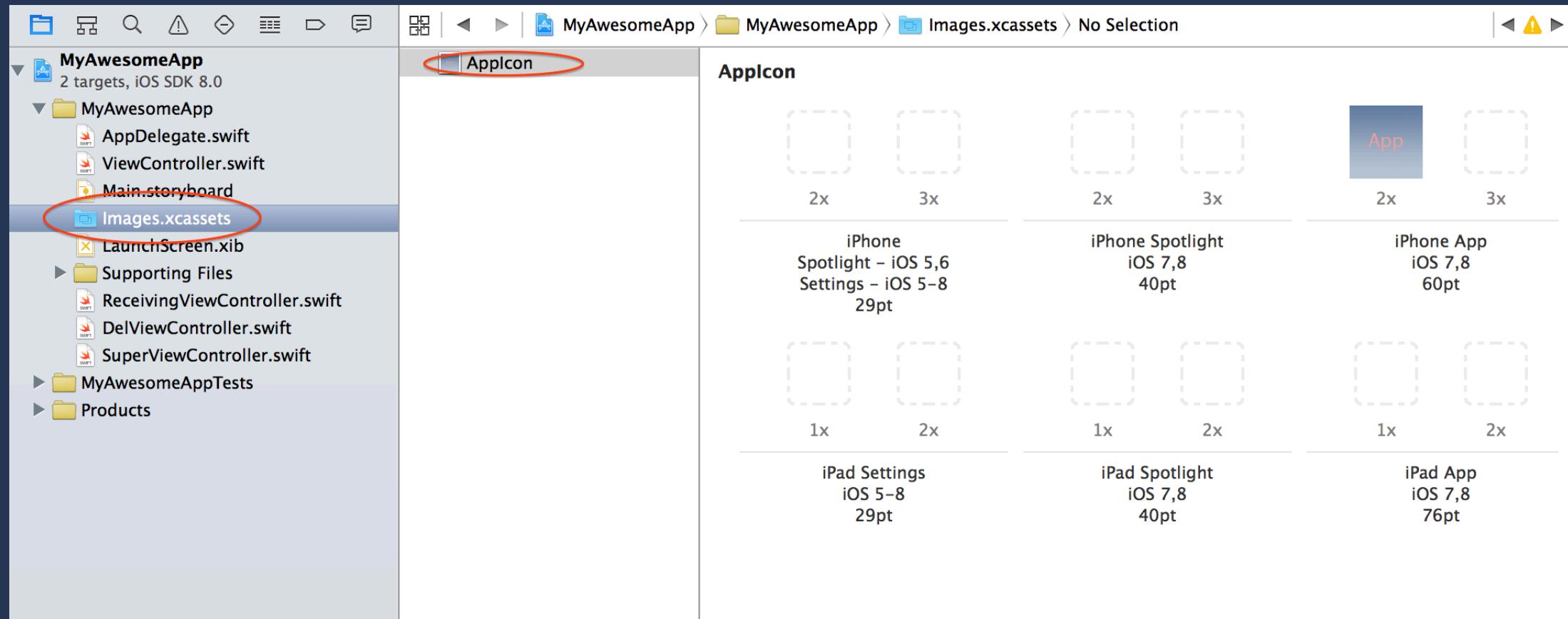
Use Core Data

Cancel

Previous

Next

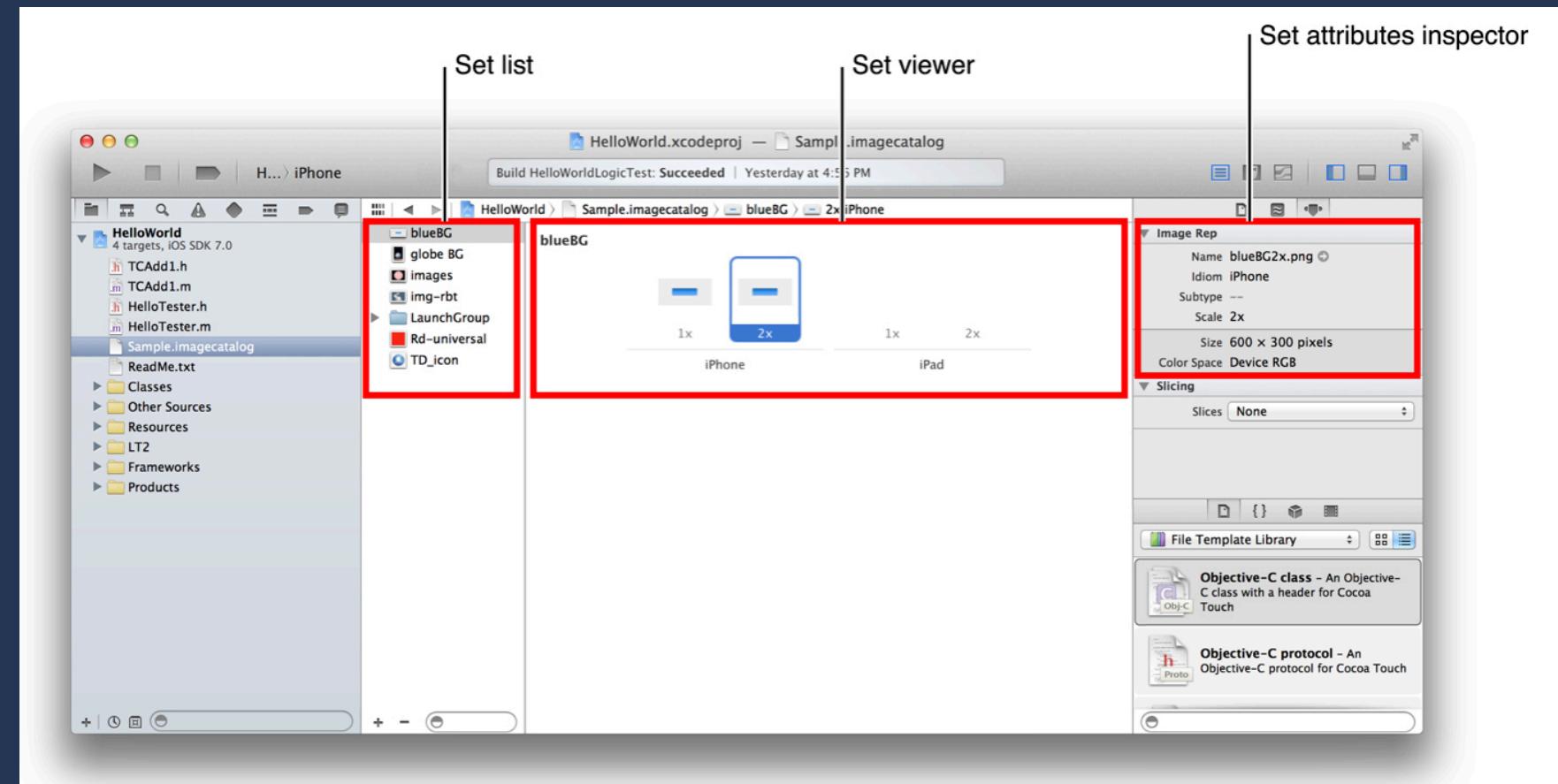
App ikoner



Søk etter "Icon and Image Sizes" i dokumentasjon for oversikt over størrelser og krav

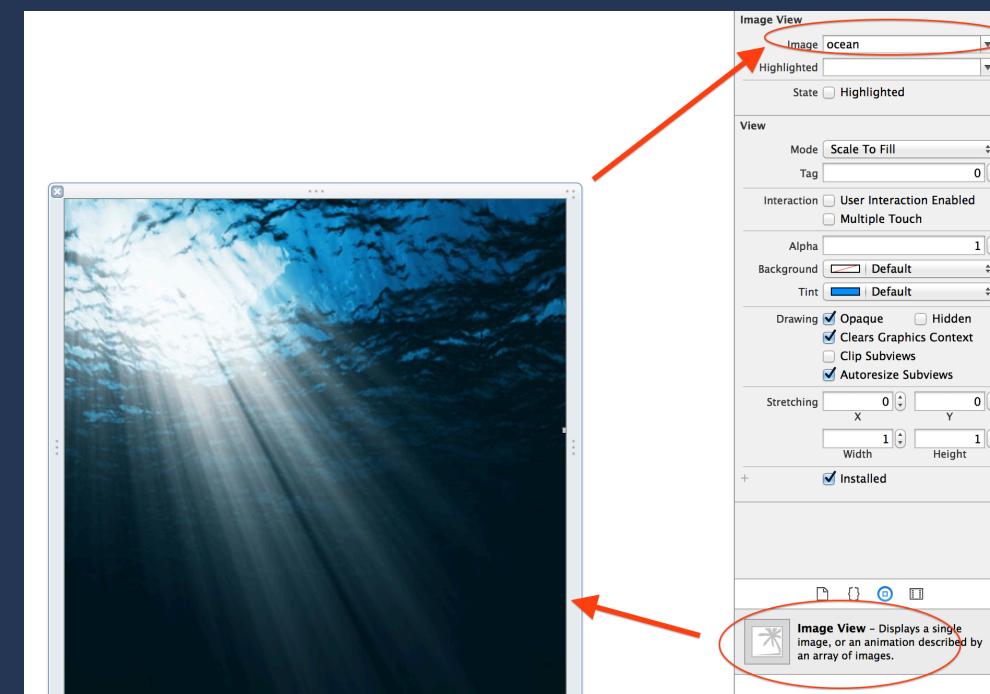
Asset catalogs

Består av image sets/icons/launch icons, og de forskjellige variantene som brukes på forskjellige enheter



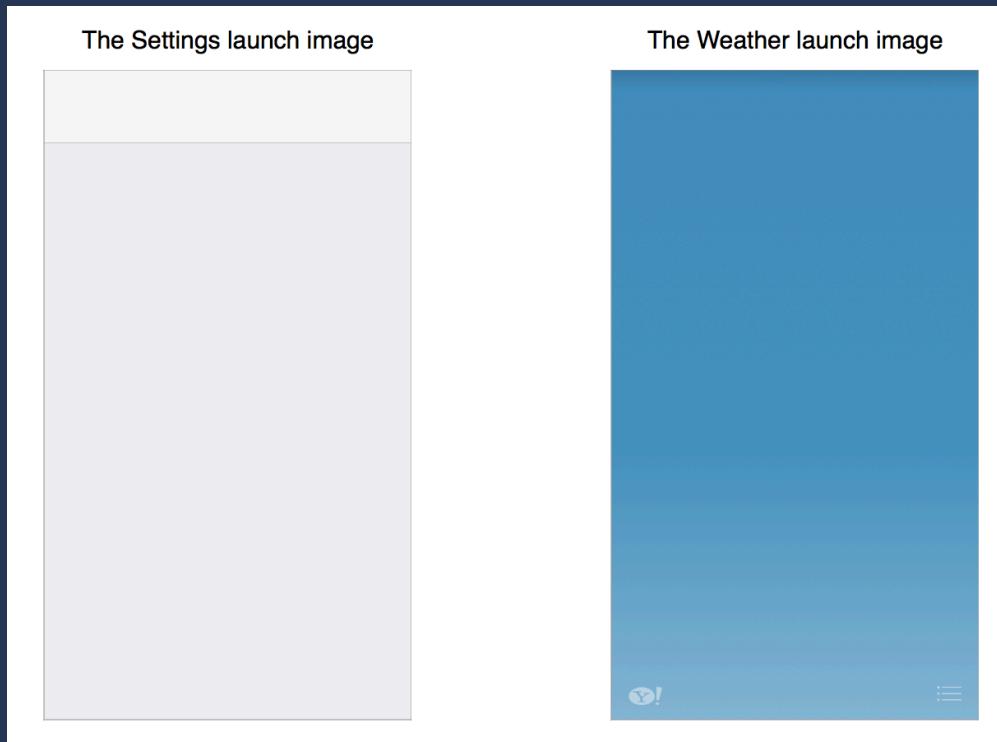
Asset catalogs

- Bilder kan lastes fra kode med: `UIImage(named: "ocean")`
- Bilder kan settes fra Interface builder, eksempelvis med å lage et `UIImageView` som du igjen setter bildet på:



Launch image / launch file (iOS 8)

Best practise: Lag et bilde/xib som er lik første skjermbilde i applikasjonen, men uten innhold. Dette for å gi en illusjon av at appen starter raskere:



Launch image / launch file (iOS 8)

Bad practise:

- Splash screen
- Brandingelementer (eks. stor logo)

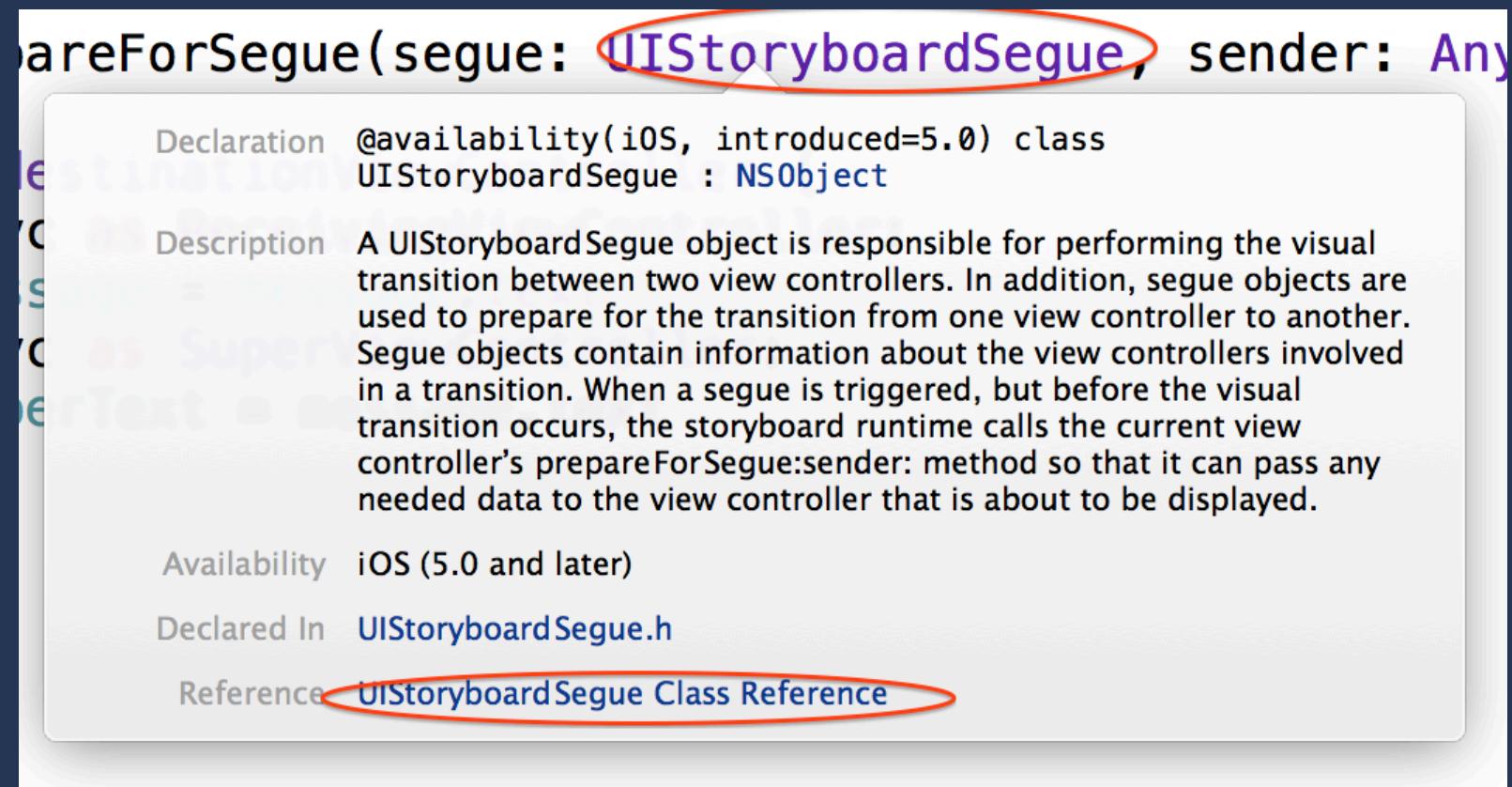
Sentral dokumentasjon

Sentral dokumentasjon

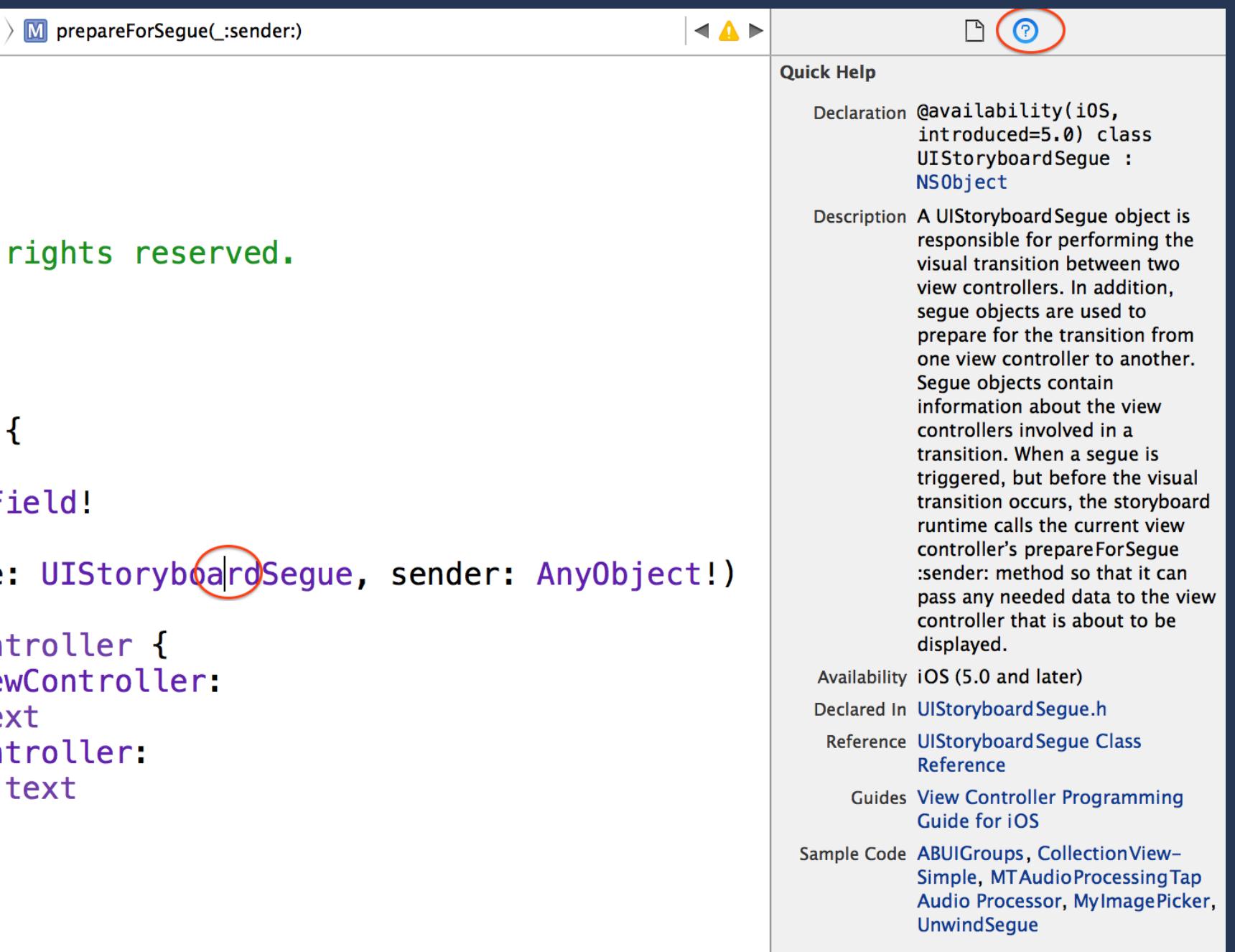
- Apple docs (XCode -> Help -> Documentation and API reference):
 - App Programming Guide for iOS
 - The App Life Cycle
 - View Programming Guide for iOS
 - View Controller Programming Guide for iOS

Sentral dokumentasjon

1. CMD-klikk på symbol for å åpne quick help
2. Klikk på linken i "Reference" for å gå til class reference



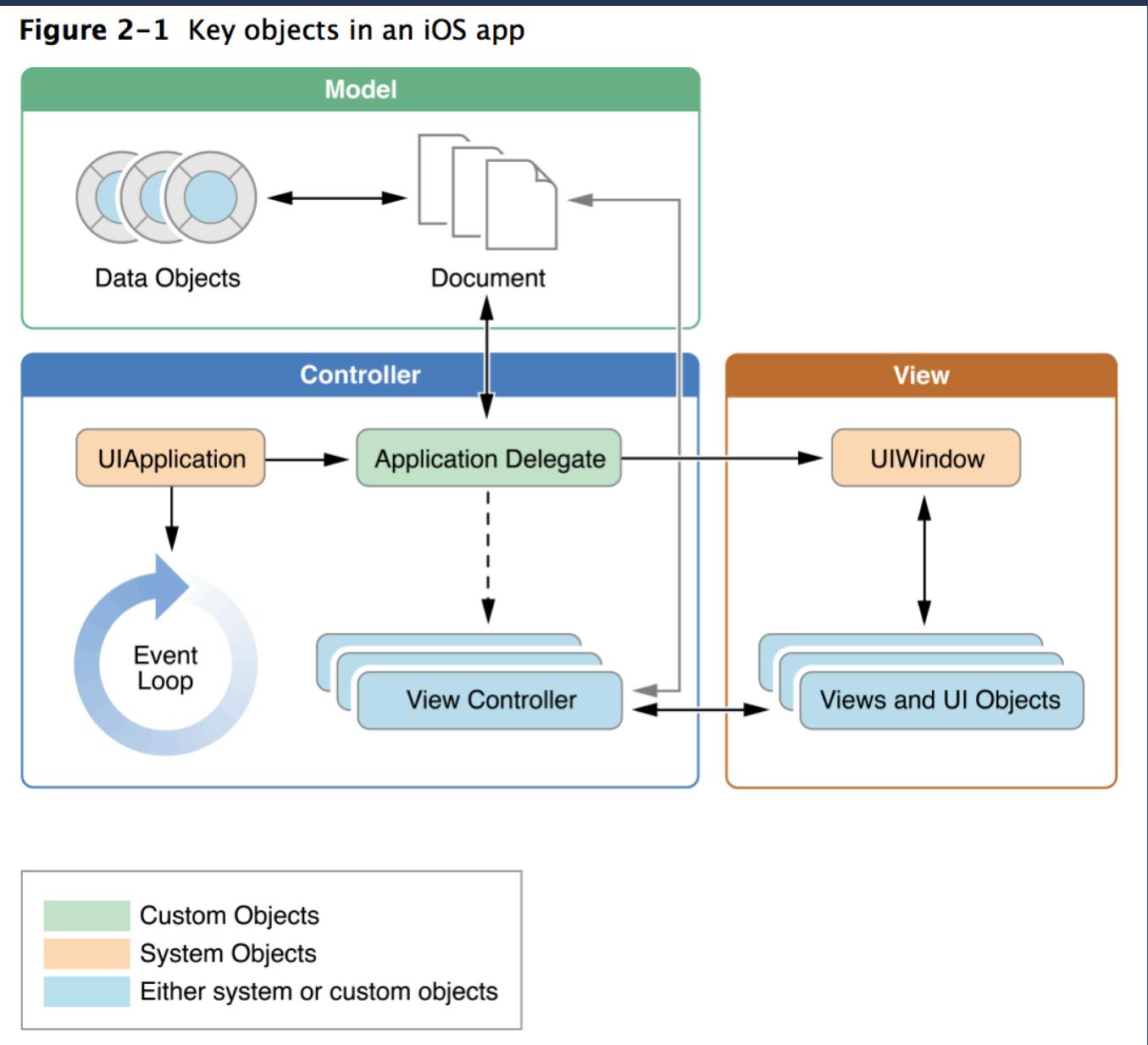
Sentral dokumentasjon



1. Sett cursor i symbol
2. Velg quick help i inspectoren

Komponentene i en iOS app

Oversikt



UIApplication

- Hver app har nøyaktig en (singleton), som blir opprettet av UIApplicationMain
- Aksesseres med UIApplication.sharedApplication()
- Håndterer eventloop og høynivå funksjonalitet
- Informerer om state transitions og eventer via app delegate, som er stedet du har mulighet til å håndtere disse

App delegate

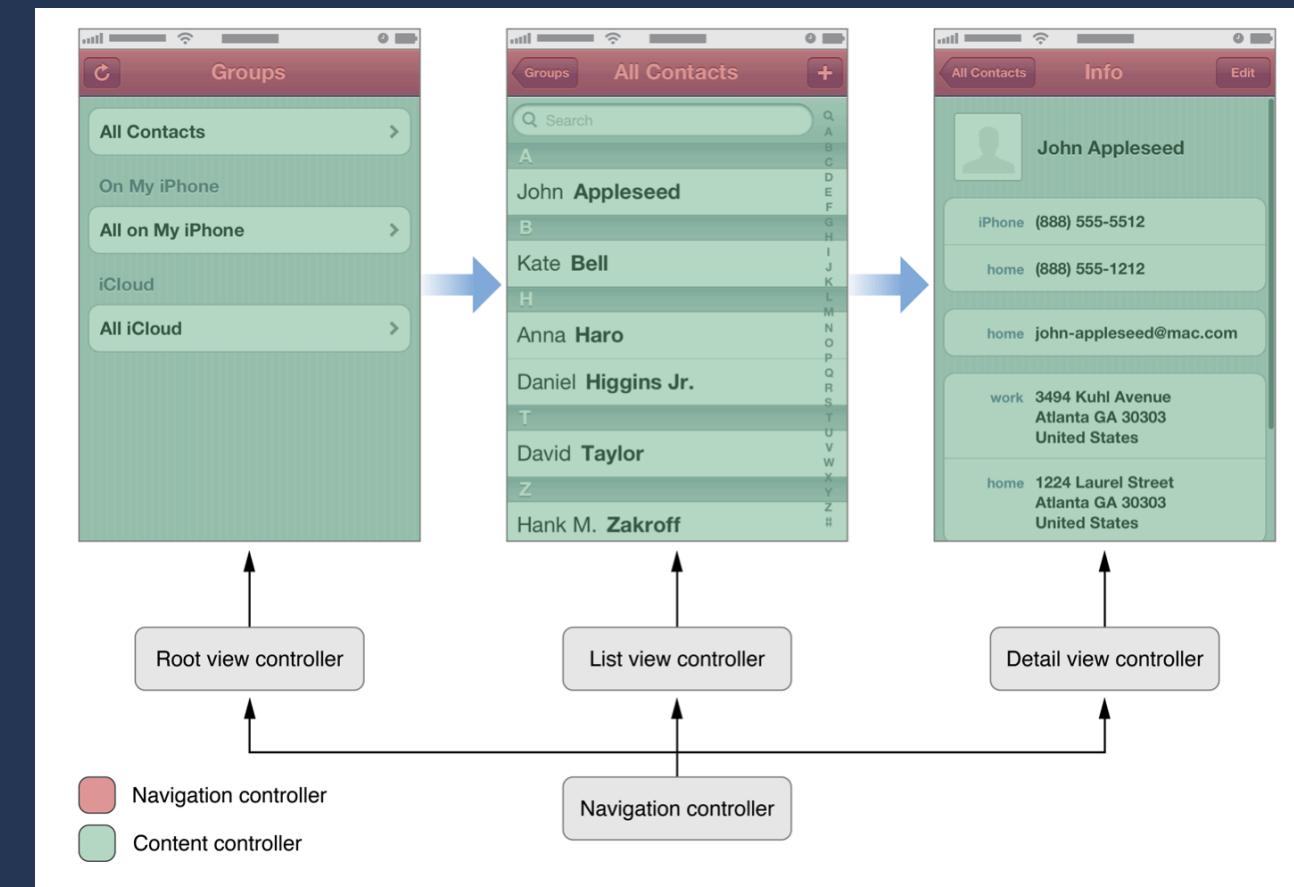
- Går hånd i hånd med UIApplication, og er stedet hvor du håndterer app events
- App initialisering (eks, views, initielle data)
- Håndterer state transitions
- Håndterer høynivå app events (eks. push-notifications)

UIWindow

- Som regel bare ett, med mindre man har innhold på flere skjermer
- Manipuleres ikke direkte. I stedet bruker man View Controllers, som igjen oppdaterer viewene som vises i UIWindow

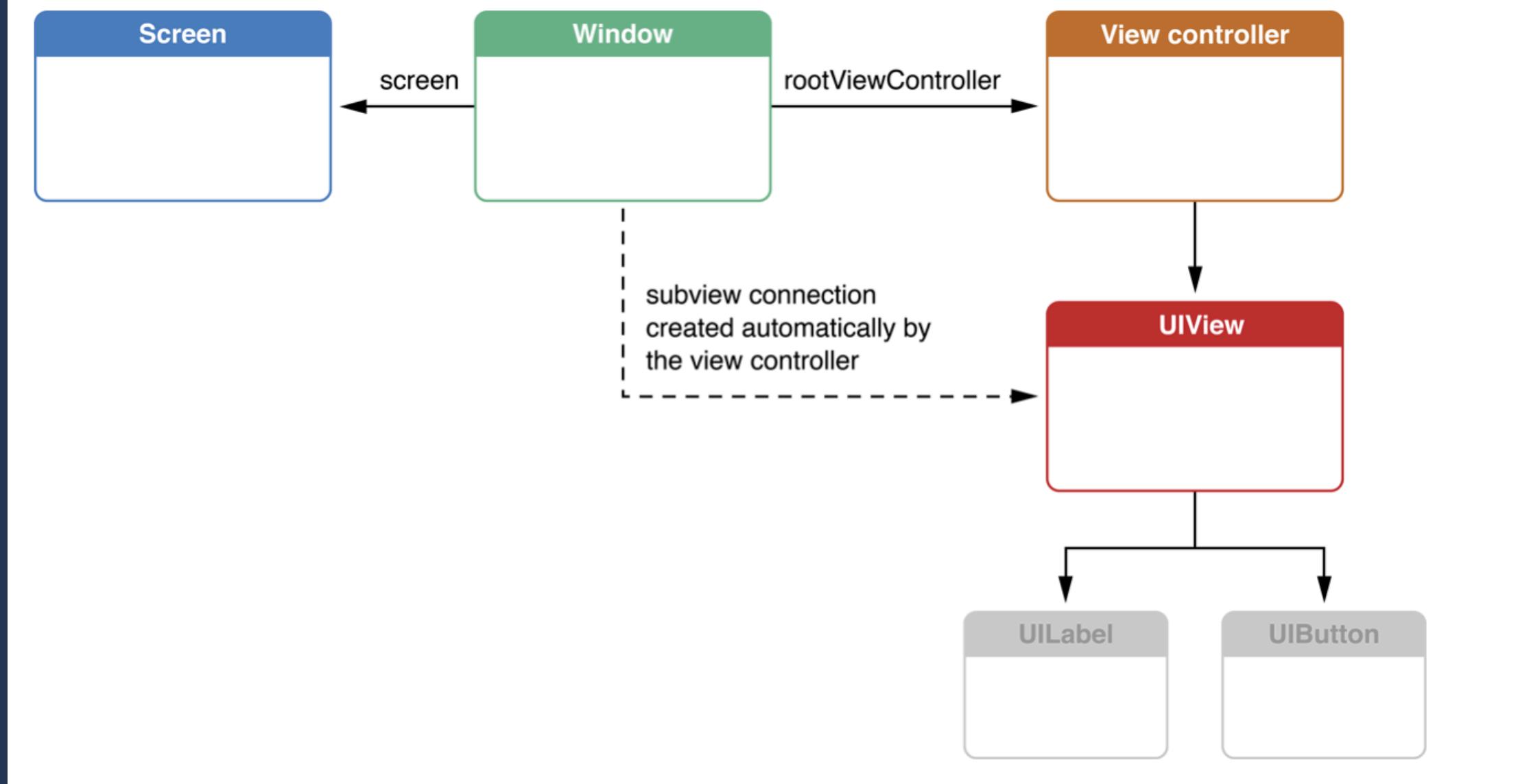
View Controllers

Håndterer et view og dens subview, koordinerer eventer og kommunikasjon mellom modeller og views



View Controllers

Figure 1–3 A view controller attached to a window automatically adds its view as a subview of the window



Views

- Tegner innhold på skjermen
- Reagerer på events
- UIKit inneholder standardviews (labels, text fields, tables, etc.)
- Kan subklasse UIView for å lage custom views

Launch flow

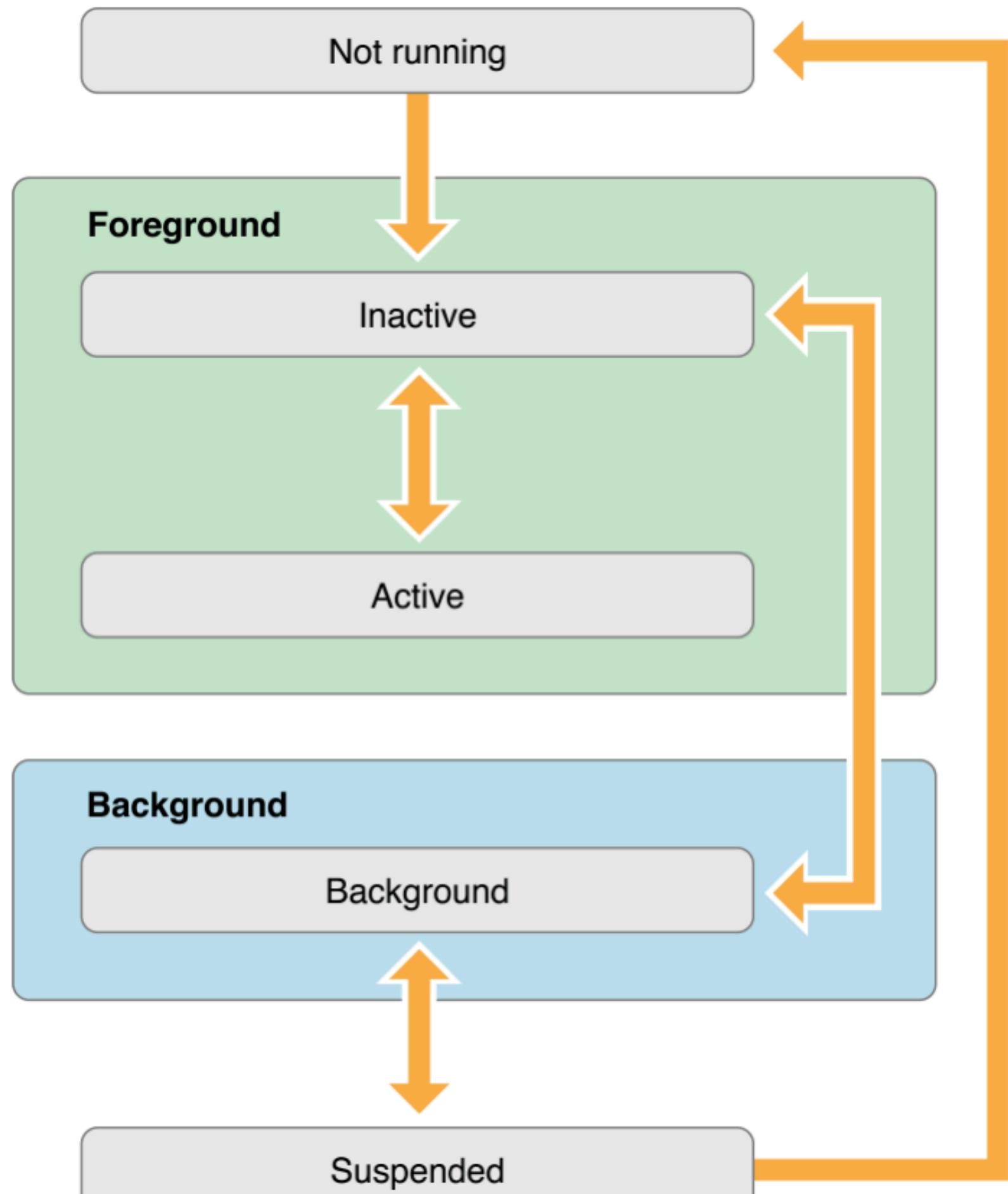
Launch

```
// @UIApplicationMain kaller UIApplicationMain() og bruker denne
// klassen som appdelegate
@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?

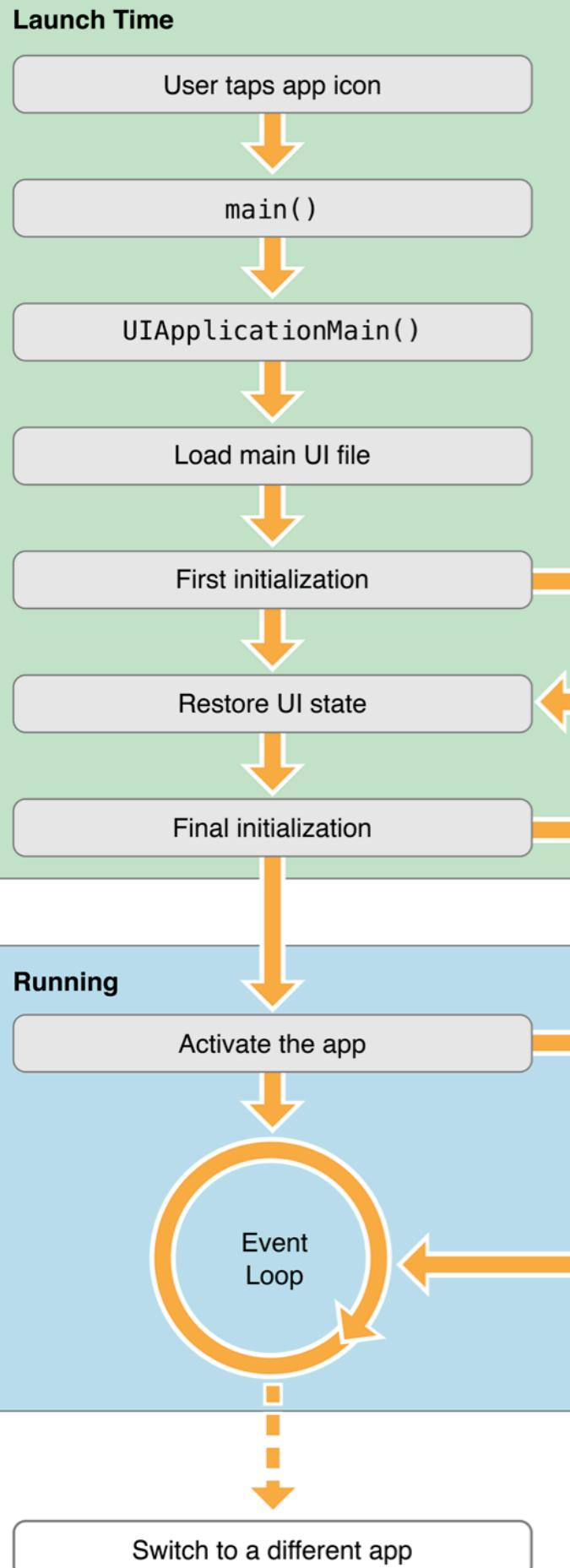
    func application(application: UIApplication,
                     didFinishLaunchingWithOptions launchOptions: [NSObject: AnyObject]?) -> Bool {
        // Gjør automatisk:
        // - Opprettning av UIWindow
        // - Laster main storyboard
        // - Setter initiell VC fra storyboard på UIWindow
        // - Viser vinduet
        return true
    }
}
```

Mer info [1](#) og [2](#)

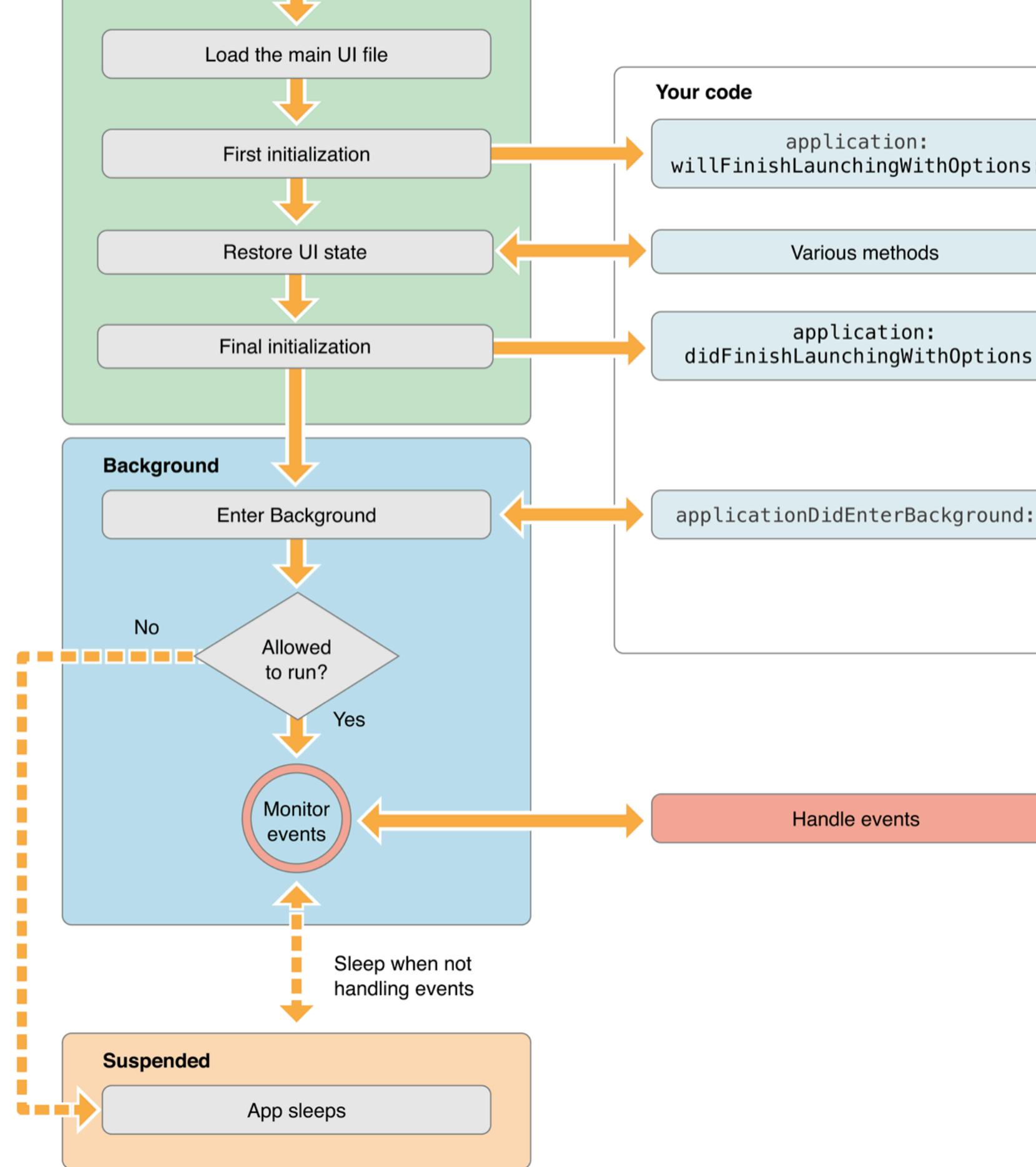
Application lifecycle



- **Inactive**: i forgrunnen, men mottar ikke eventer (eks. ved inkommande anrop)
- **Active**: normalstate. I forgrunn og mottar eventer
- **Background**: Mulighet for å kjøre enkelte operasjoner i bakgrunnen (eks. lyd) / Ved launch inn i background (eks. background fetch, push)
- **Suspended**: I minnet, men kjører ingen kode



- Det er forventet at appen starter på maks 5 sekunder, i motsatt fall kan appen bli terminated.



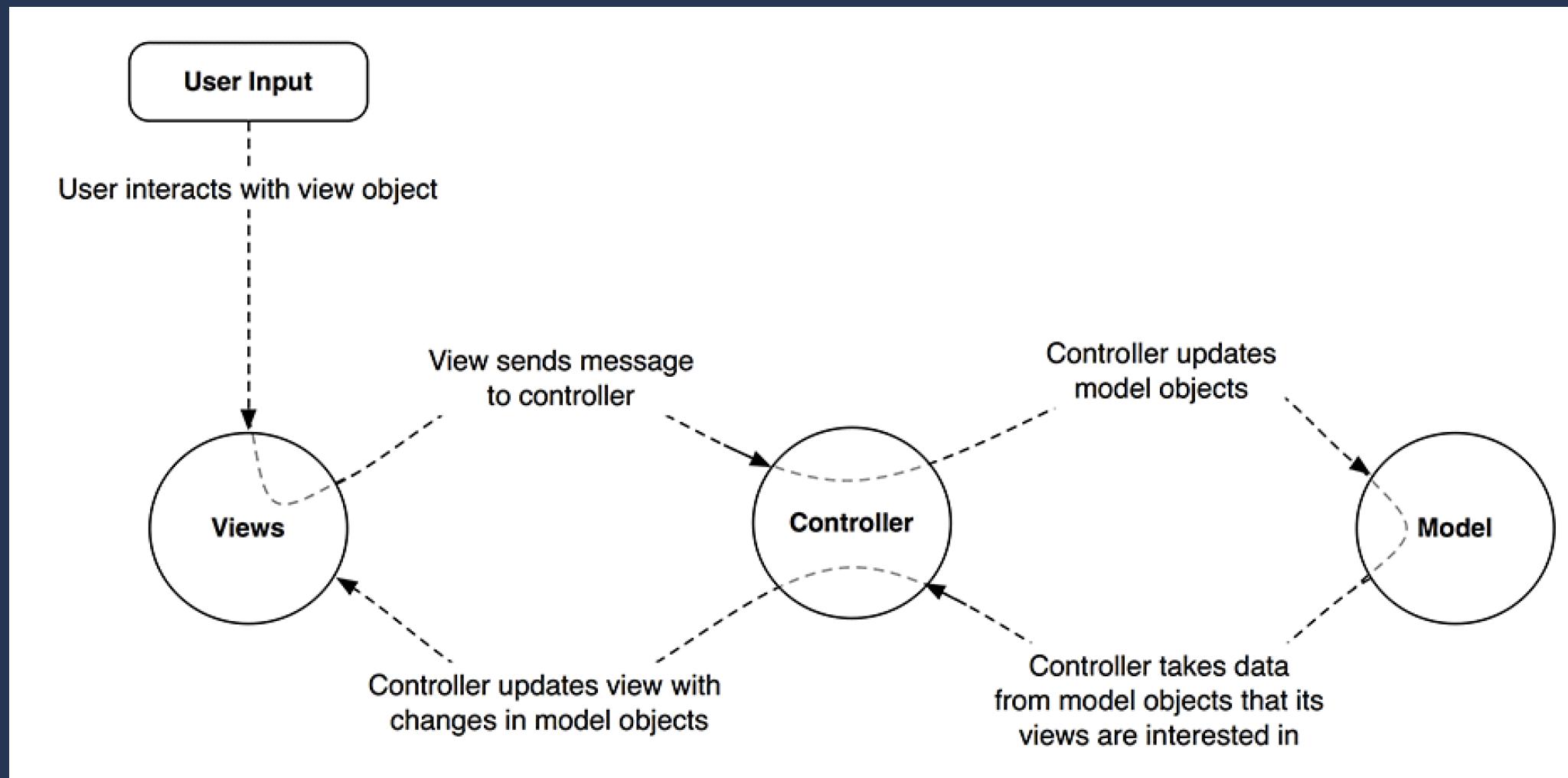
- Maks 5 sekunder til å kjøre kode før suspended
- Kan be om mer tid for enkelte oppgaver med `beginBackgroundTaskWithExpirationHandler:`

MVC

MVC

- Views: knapper, tekstfelt og andre komponenter synlige for brukeren
- Models: holder på data. Eks. arrays, dictionaries. Har ingen kjennskap til views
- Controllere:
 - kan motta hendelser fra views
 - kan oppdatere modeller
 - kan oppdatere views med modelldata

MVC

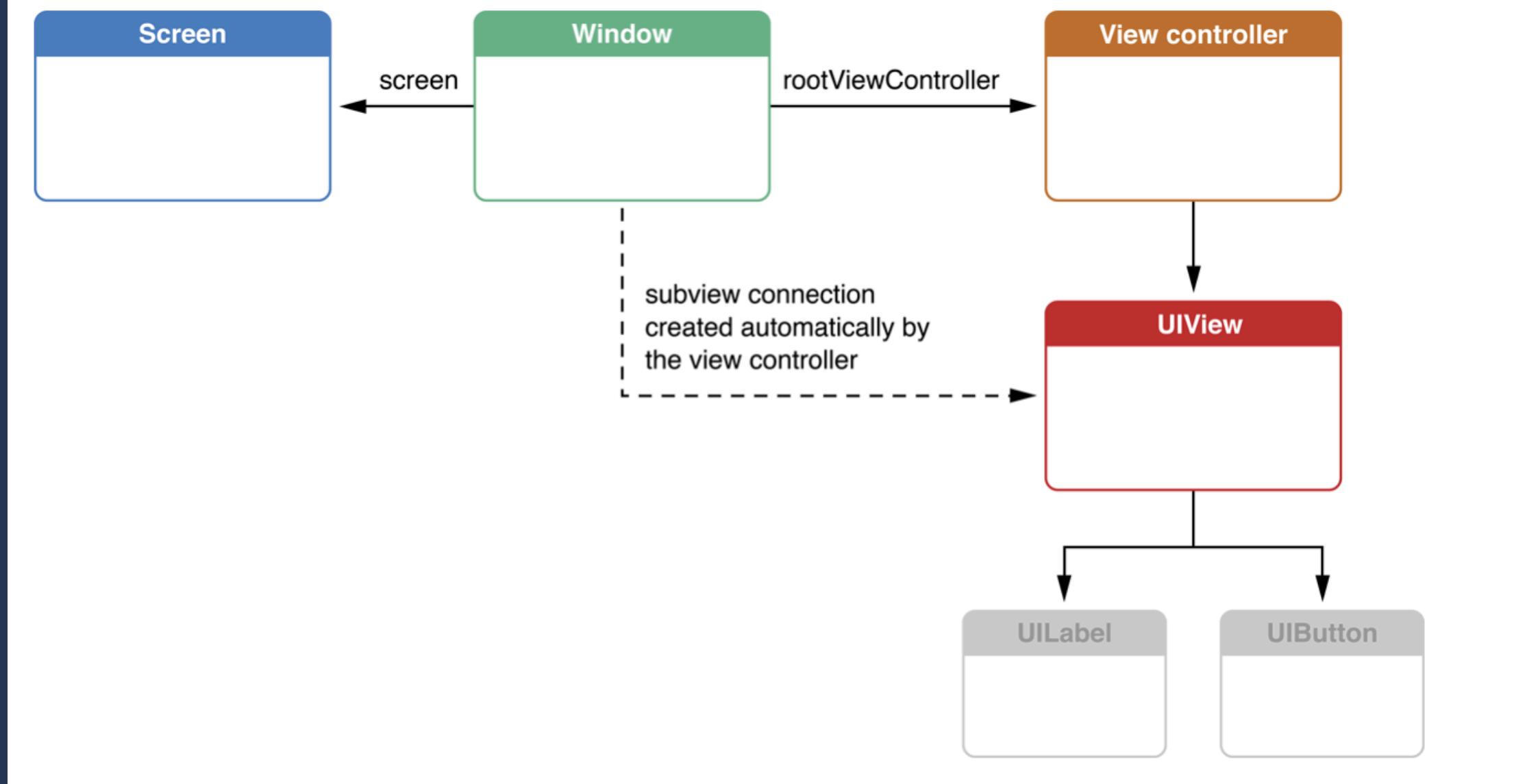


Bilde: The Big Nerd Ranch Guide, kap. 1

View Controllers (VC)

View Controllers

Figure 1–3 A view controller attached to a window automatically adds its view as a subview of the window



View Controllers

- **Du setter aldri views direkte på UIWindow selv.** I stedet setter man en view controller på UIWindow, som igjen vil legge sine views til vinduet
- View controllere og ressursstyring:
 - Laster bare inn viewet sitt når det trengs
 - Kan deallokere viewet, f.eks. ved lite minne

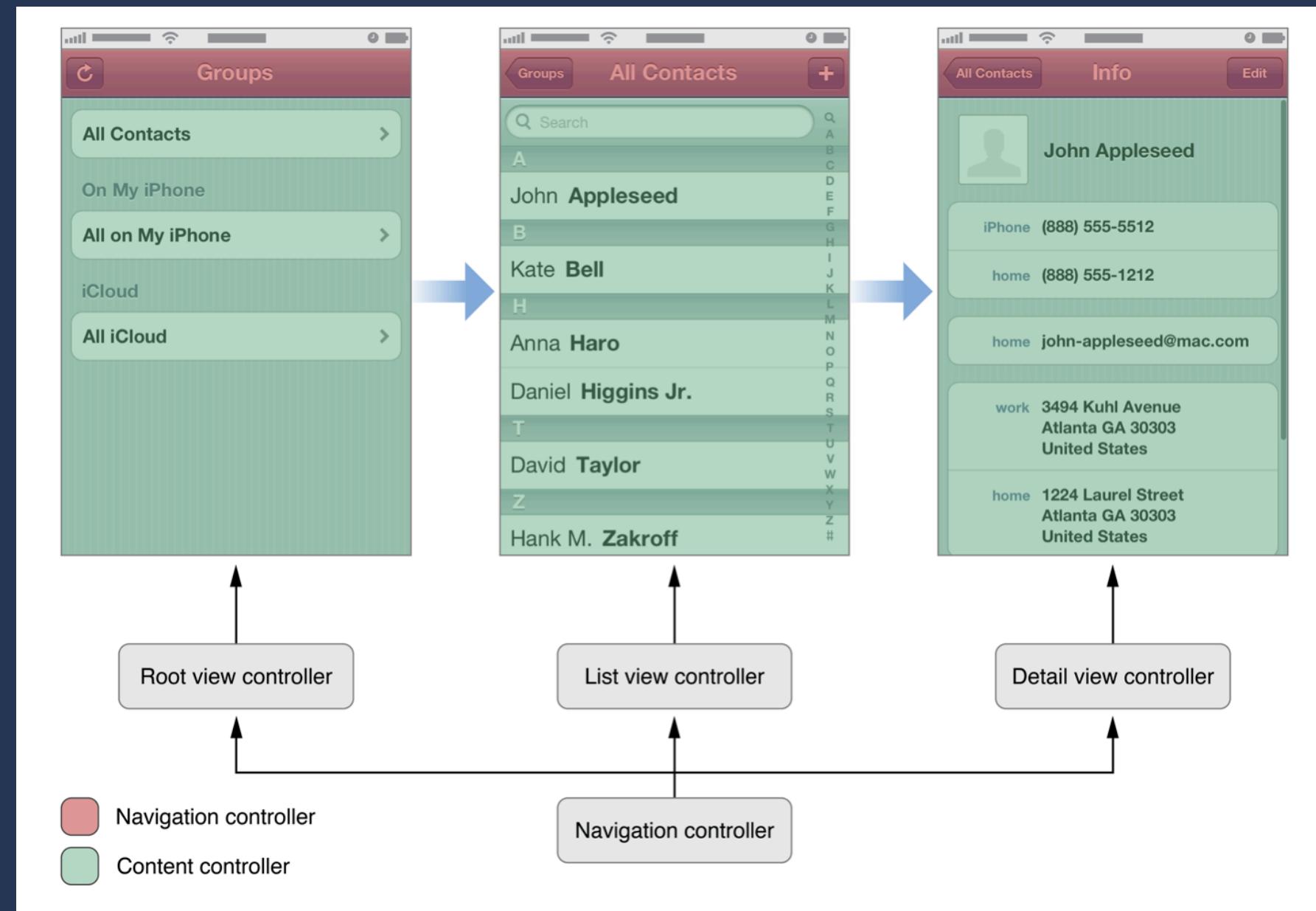
View Controllers

- View Controllere vil typisk bare kjenne til en del av applikasjonen, og vil derfor kommunisere med andre view controllere
- Views sender meldinger til sin VC. Eksempelvis når du trykker på en UIButton kan det sende en melding til sin VC, som definerer hva som skal skje

View Controllers

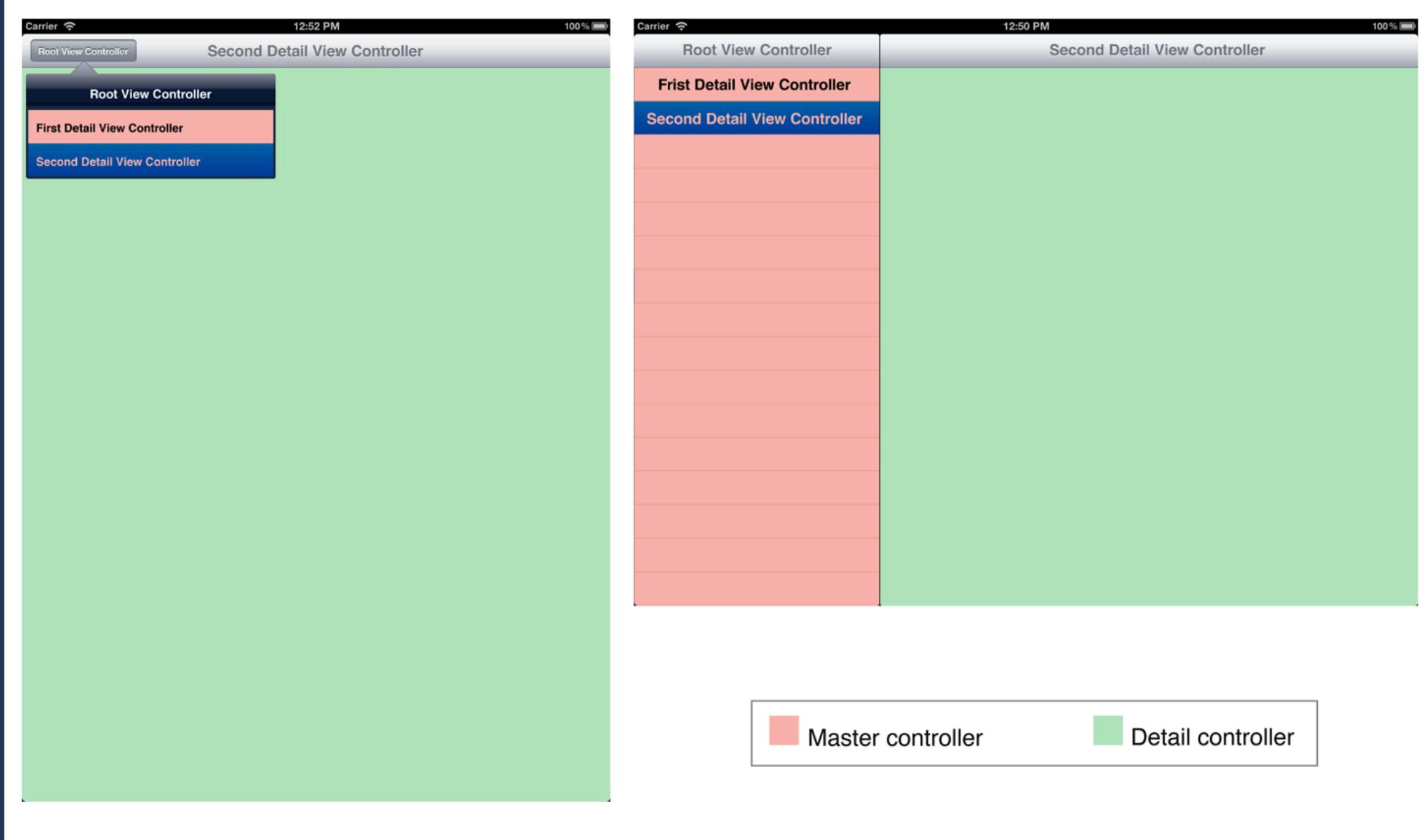
- Content view controllers: "vanlige" view controllere du definerer for å vise/hente data fra bruker
- Container view controllers: inneholder andre view controllere, eks. UINavigationController, UISplitViewController

Navigation Controller



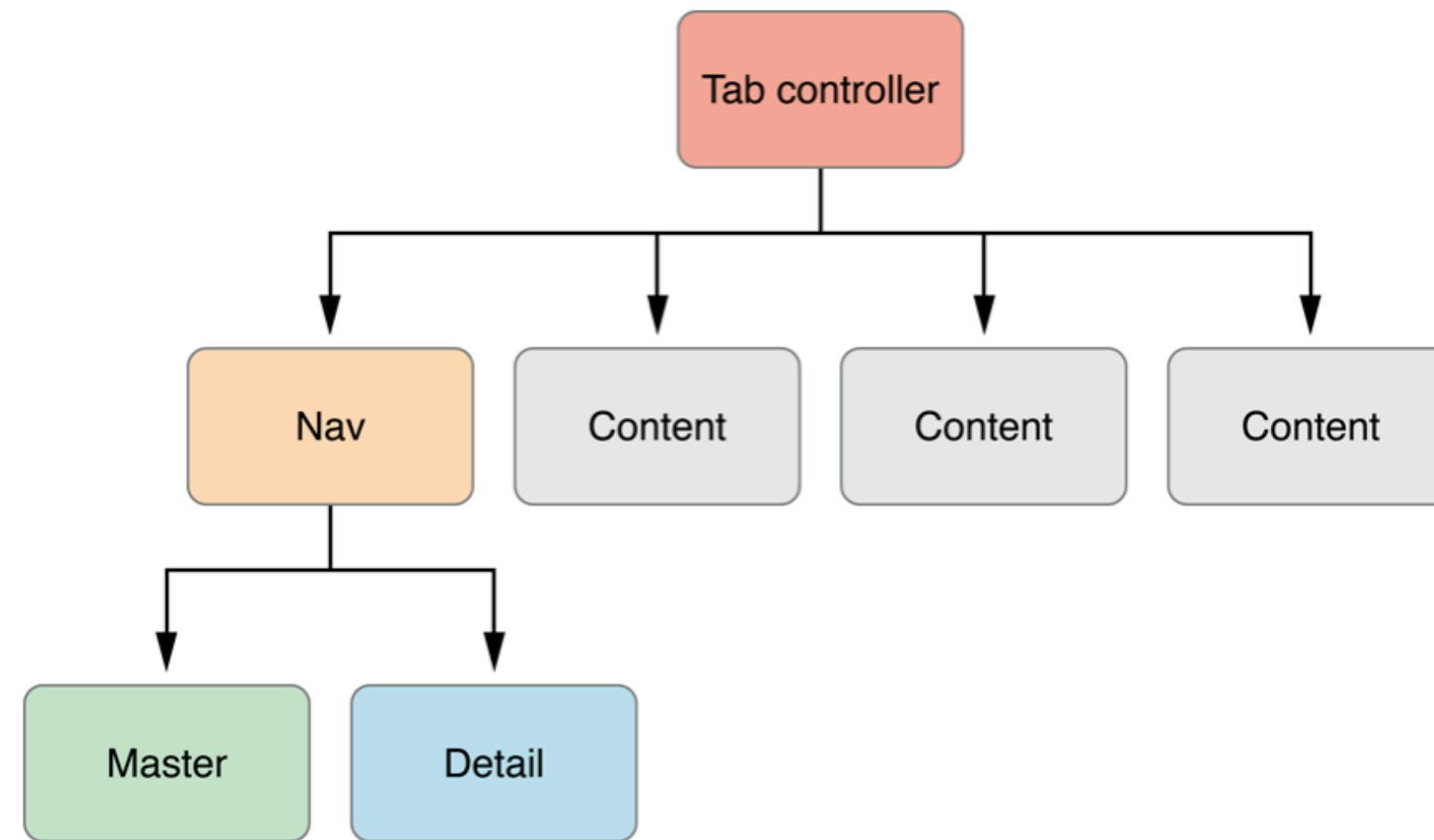
Split View Controller

Figure 1-9 A master–detail interface in portrait and landscape modes

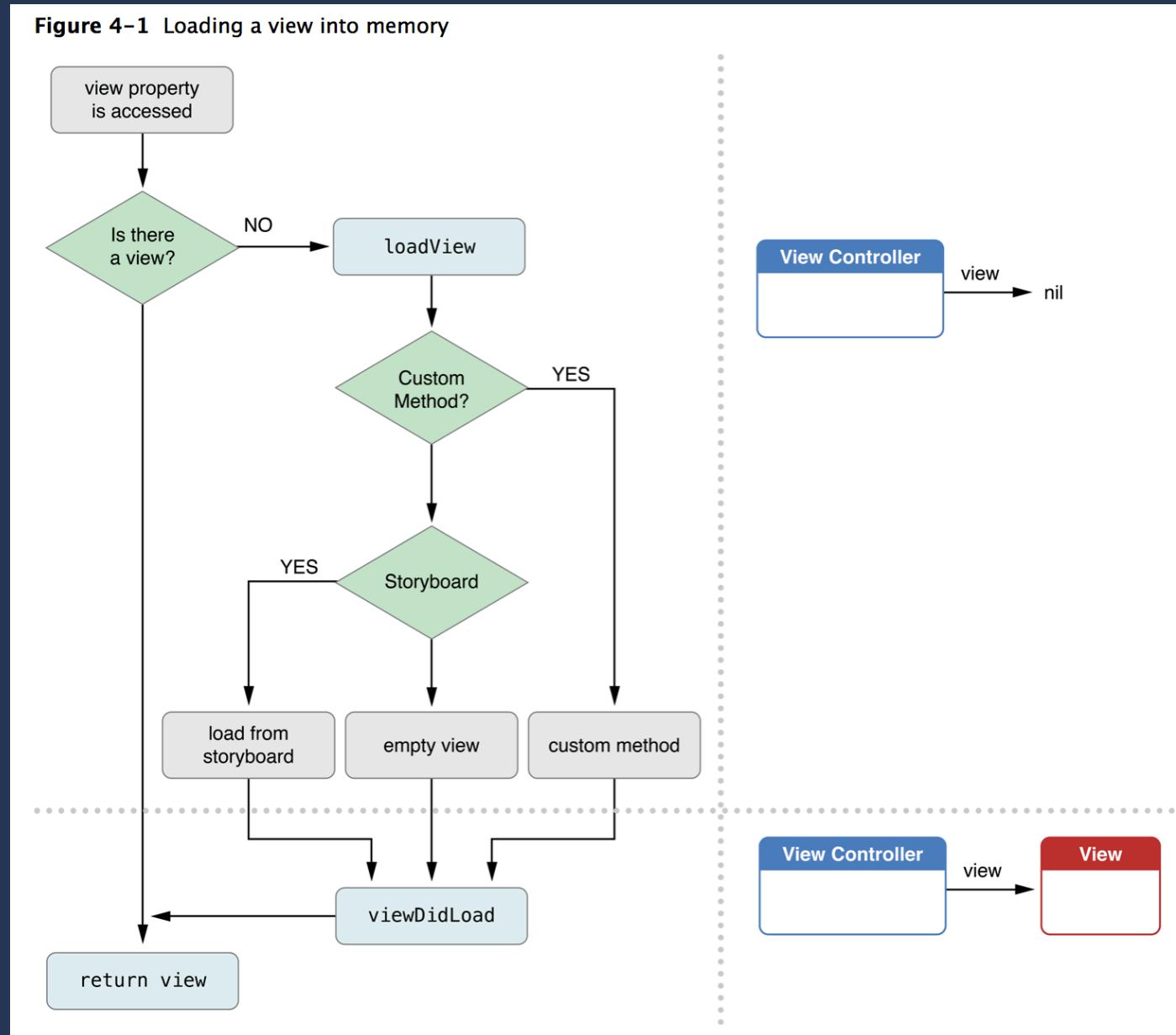


Nøstede view controllere

Figure 1-11 Parent-child relationships



Innlasting av views i minnet

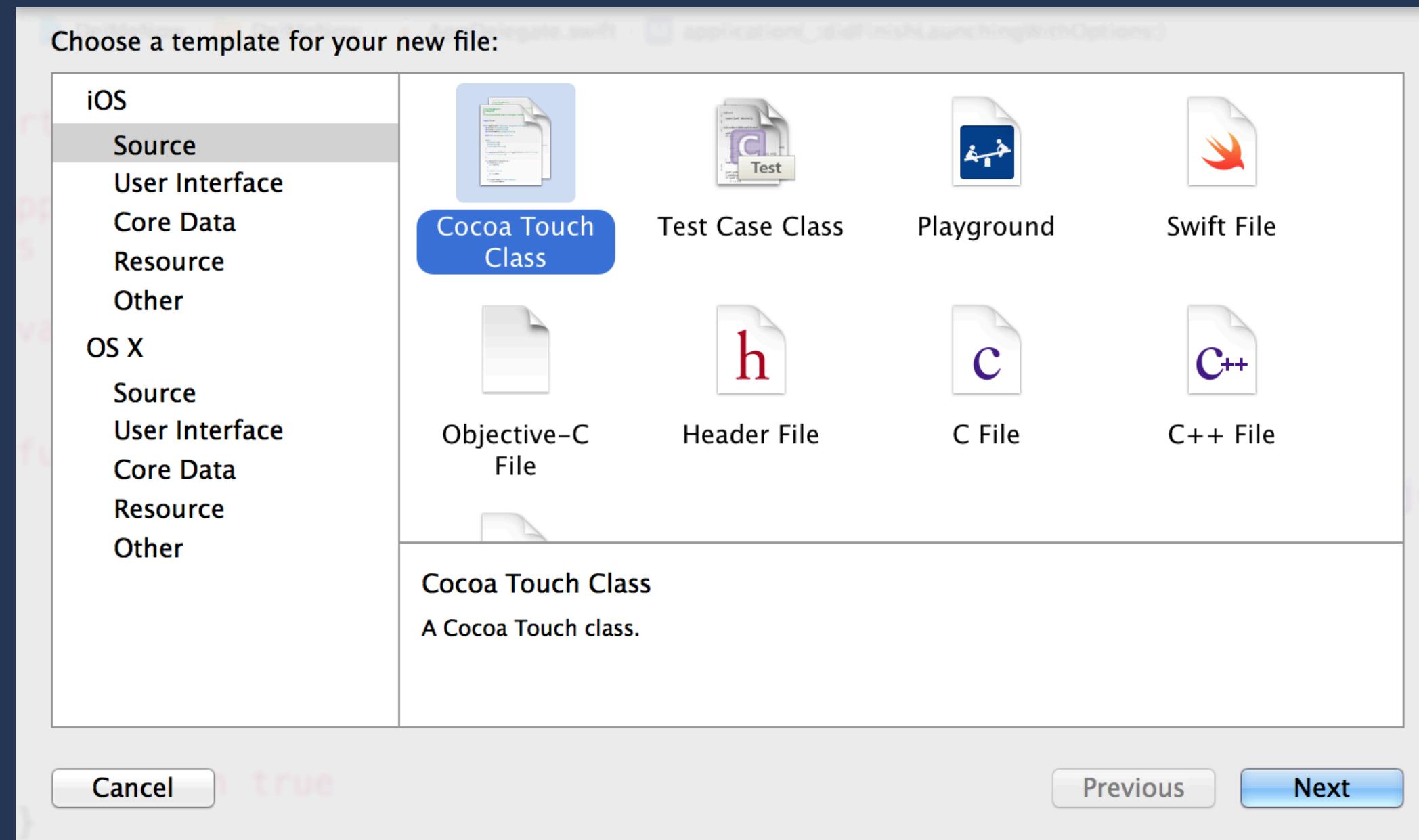


Hello world

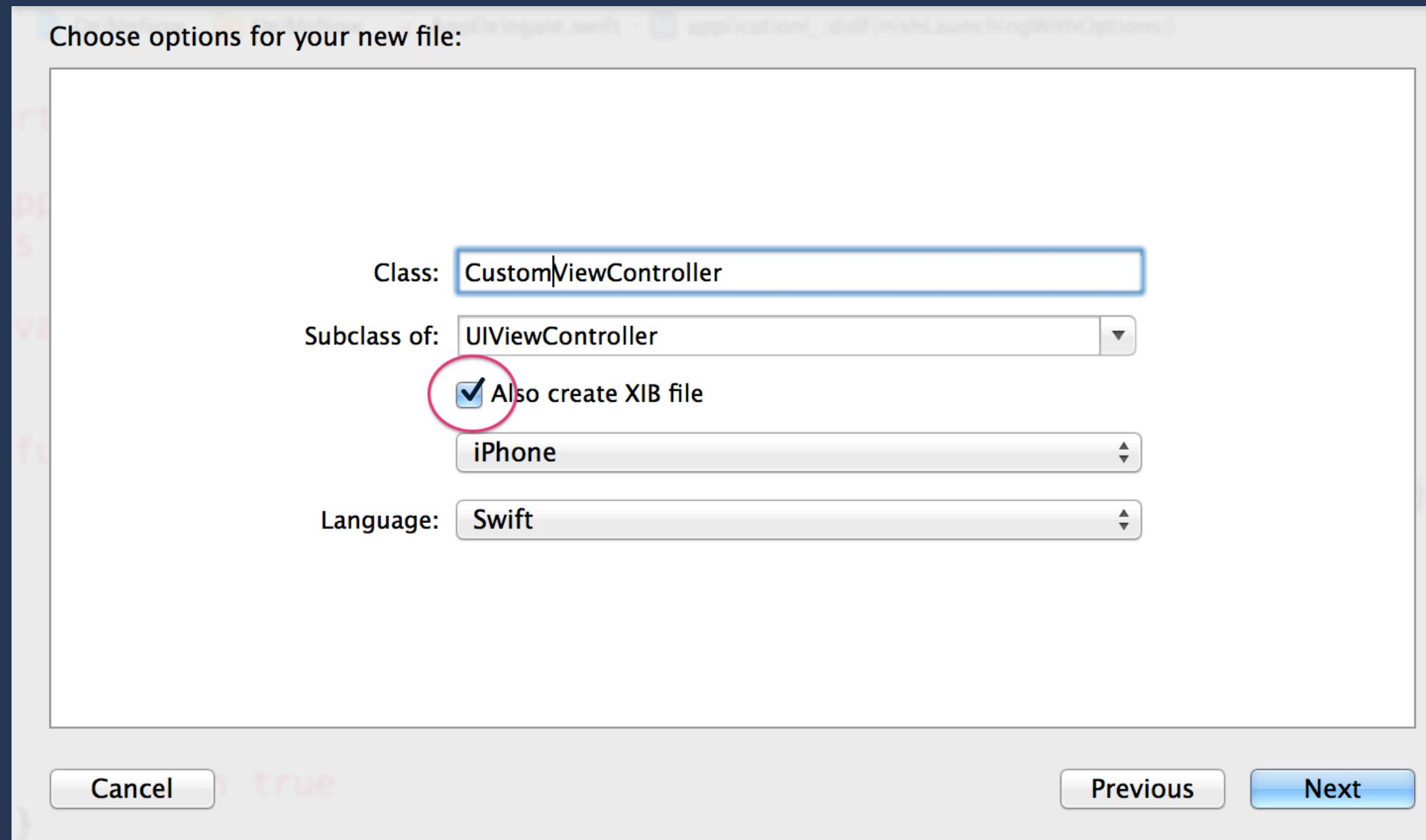
Måter å sette opp views og view controllere på

- XIB
- Storyboards

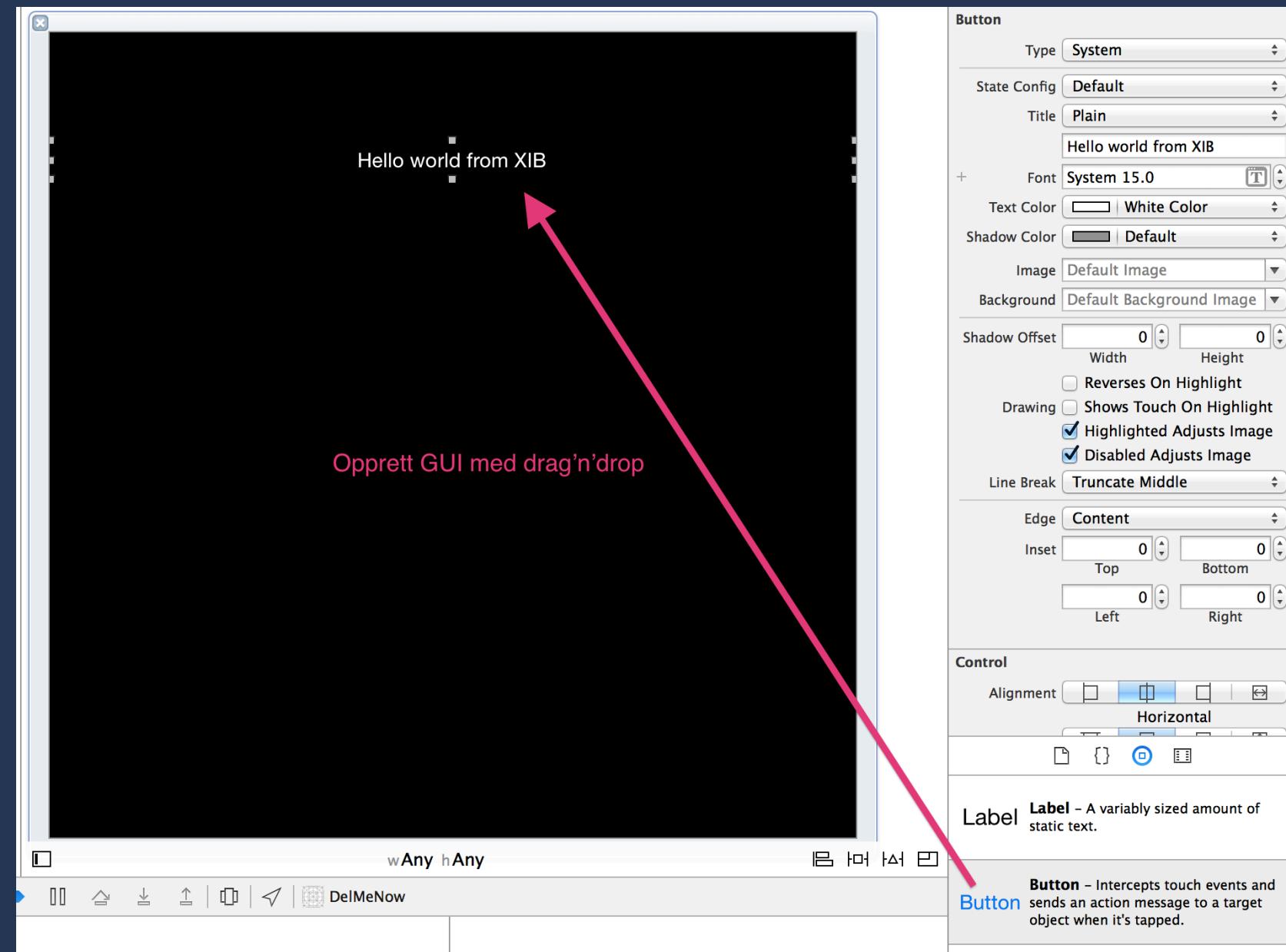
XIB eksempel



XIB eksempel

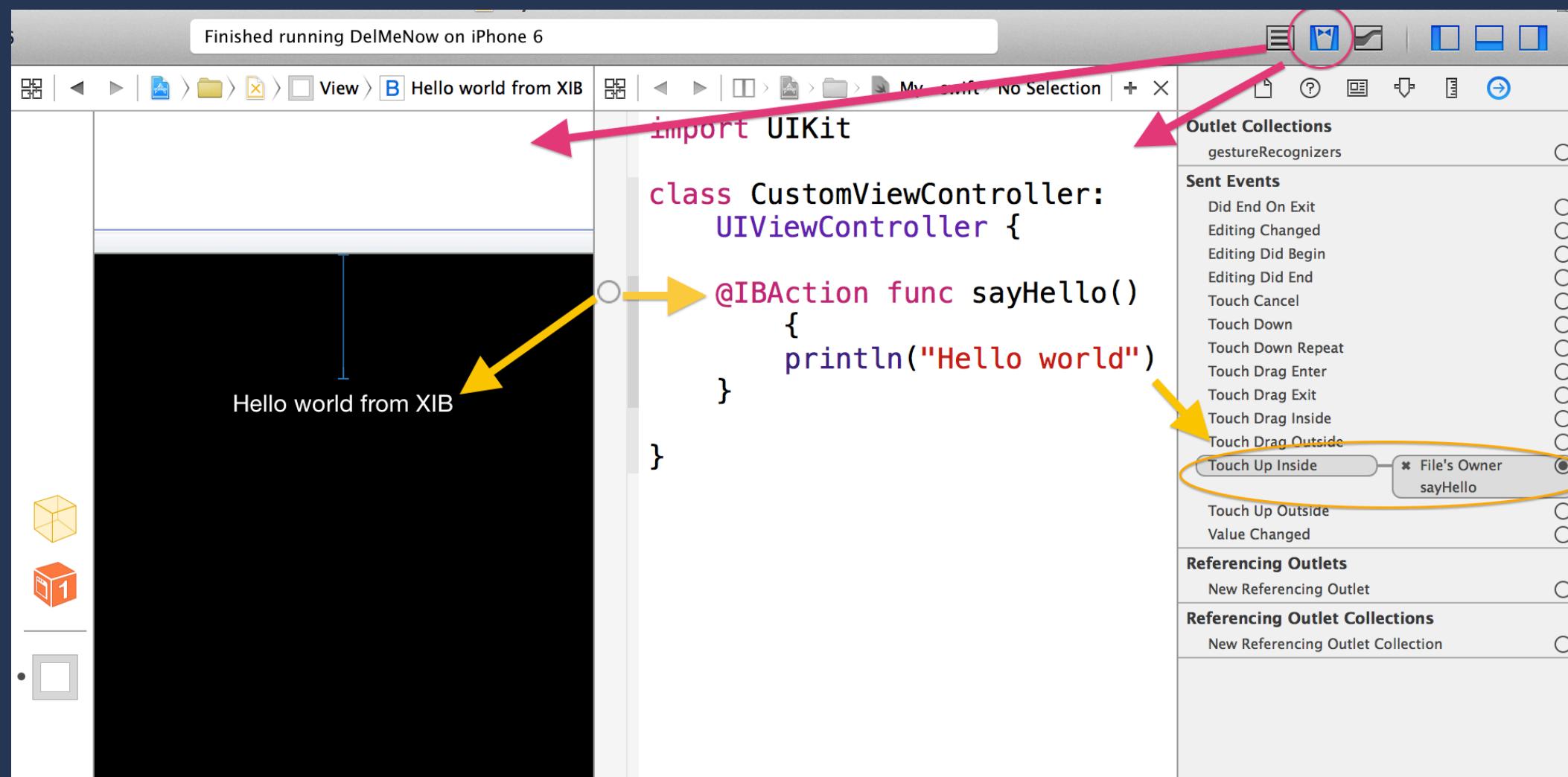


XIB eksempel



XIB eksempel

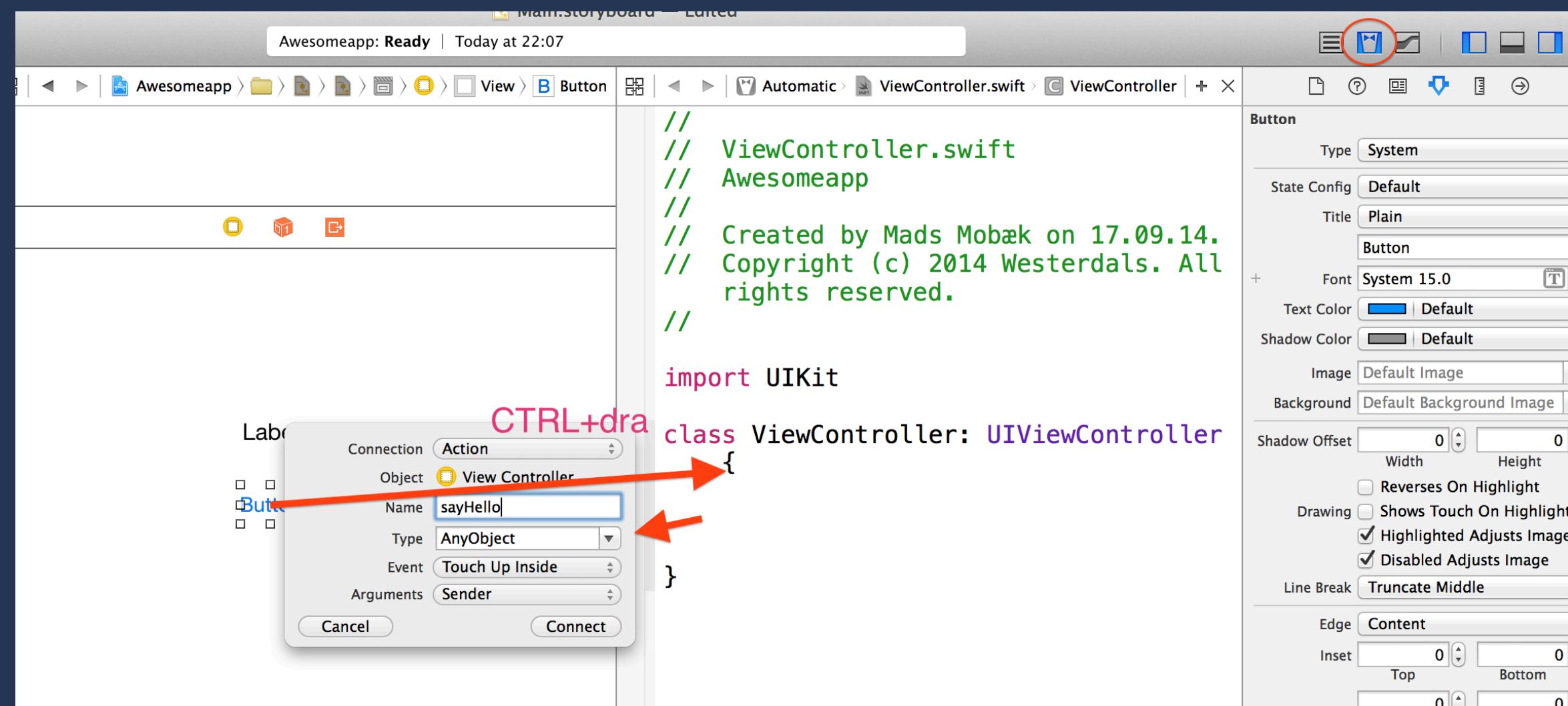
Koble view og view controllere sammen via assistant editor



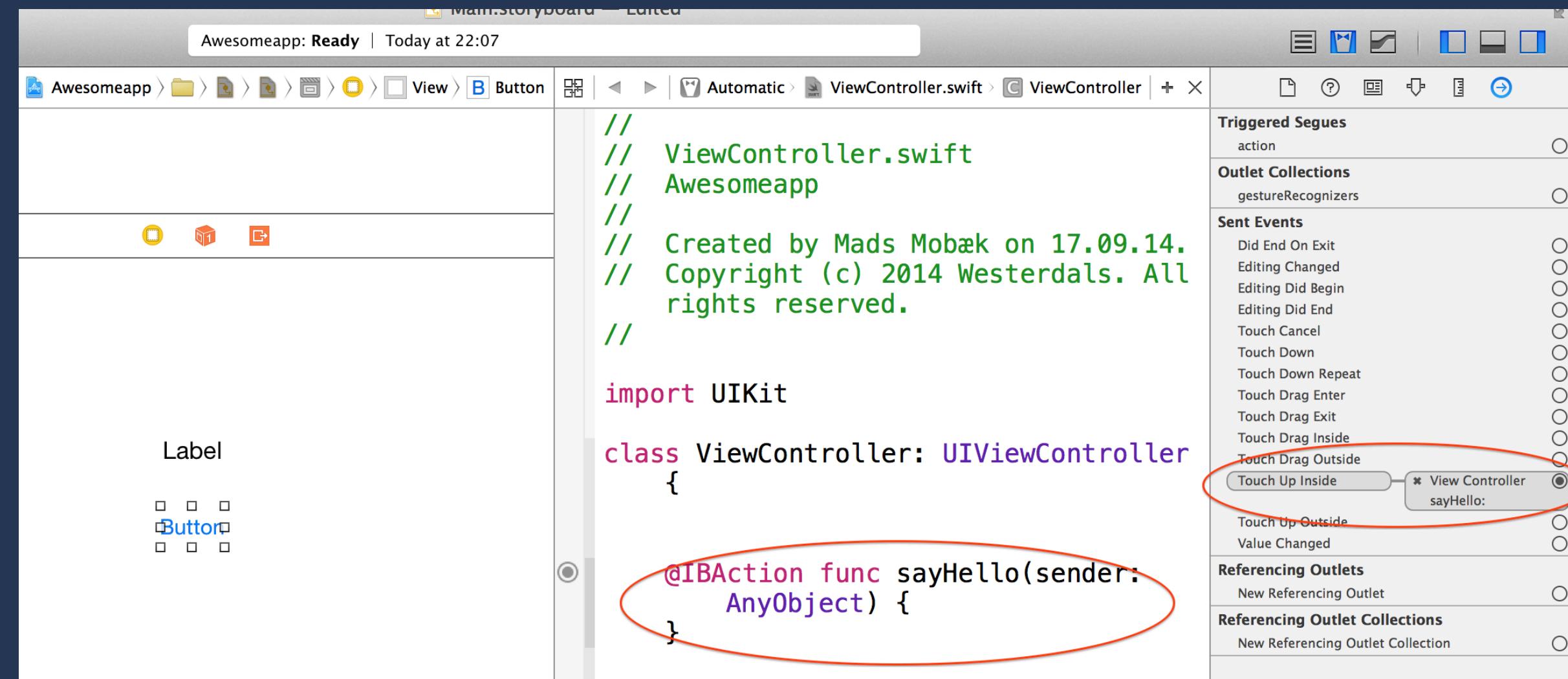
Koble view og view controllere sammen

- Outlets - referanser til UI elementer fra kode (manipulere/hente ut verdier)
- Targets - referanser fra UI til kode (kalle metoder)

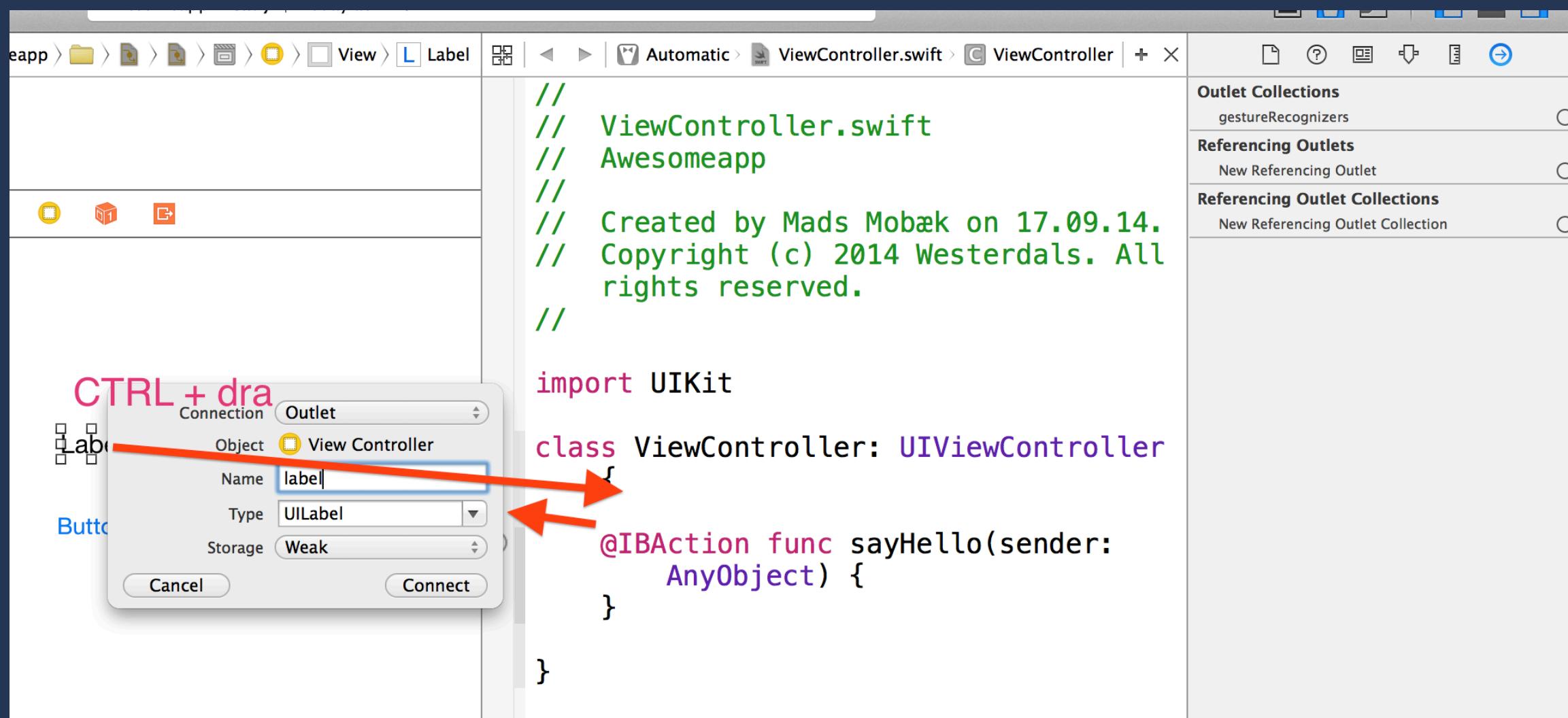
Koble view og view controllerere sammen



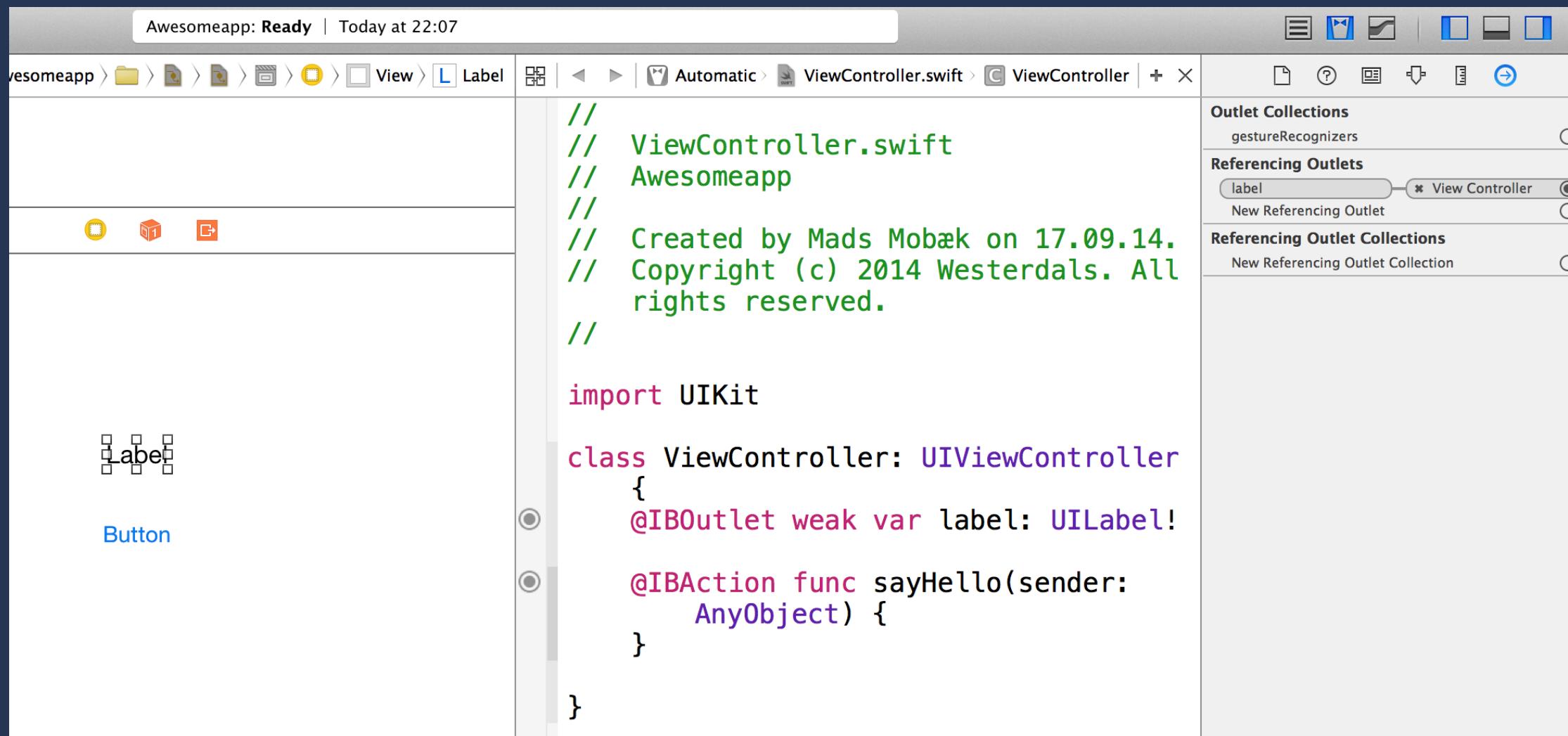
Koble view og view controllere sammen



Koble view og view controllere sammen

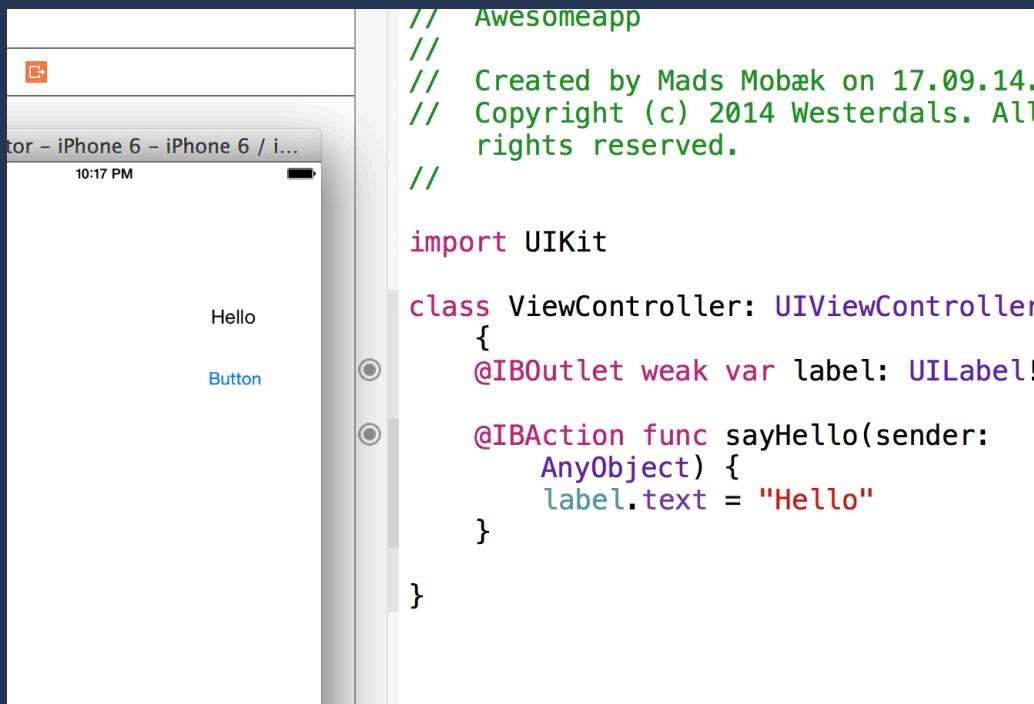


Koble view og view controllerere sammen



Koble view og view controllere sammen

Ferdig eksempel: action sayHello kalles når knappen trykkes på, og setter verdien på label (outlet)



The image shows a screenshot of the Xcode IDE. On the left, there is a storyboard preview showing a single view with a label containing the text "Hello" and a button below it. On the right, the code editor displays the `ViewController.swift` file. The code is as follows:

```
// Awesomeapp
// Created by Mads Mobæk on 17.09.14.
// Copyright (c) 2014 Westerdals. All rights reserved.

import UIKit

class ViewController: UIViewController {
    @IBOutlet weak var label: UILabel!

    @IBAction func sayHello(sender: AnyObject) {
        label.text = "Hello"
    }
}
```

XIB eksempel

```
let viewController = UIViewController(nibName: "CustomViewController", bundle: NSBundle mainBundle())
let nib = NSBundle mainBundle().loadNibNamed("myView", owner: self, options: nil)
let view = nib.firstObject as? UIView
```

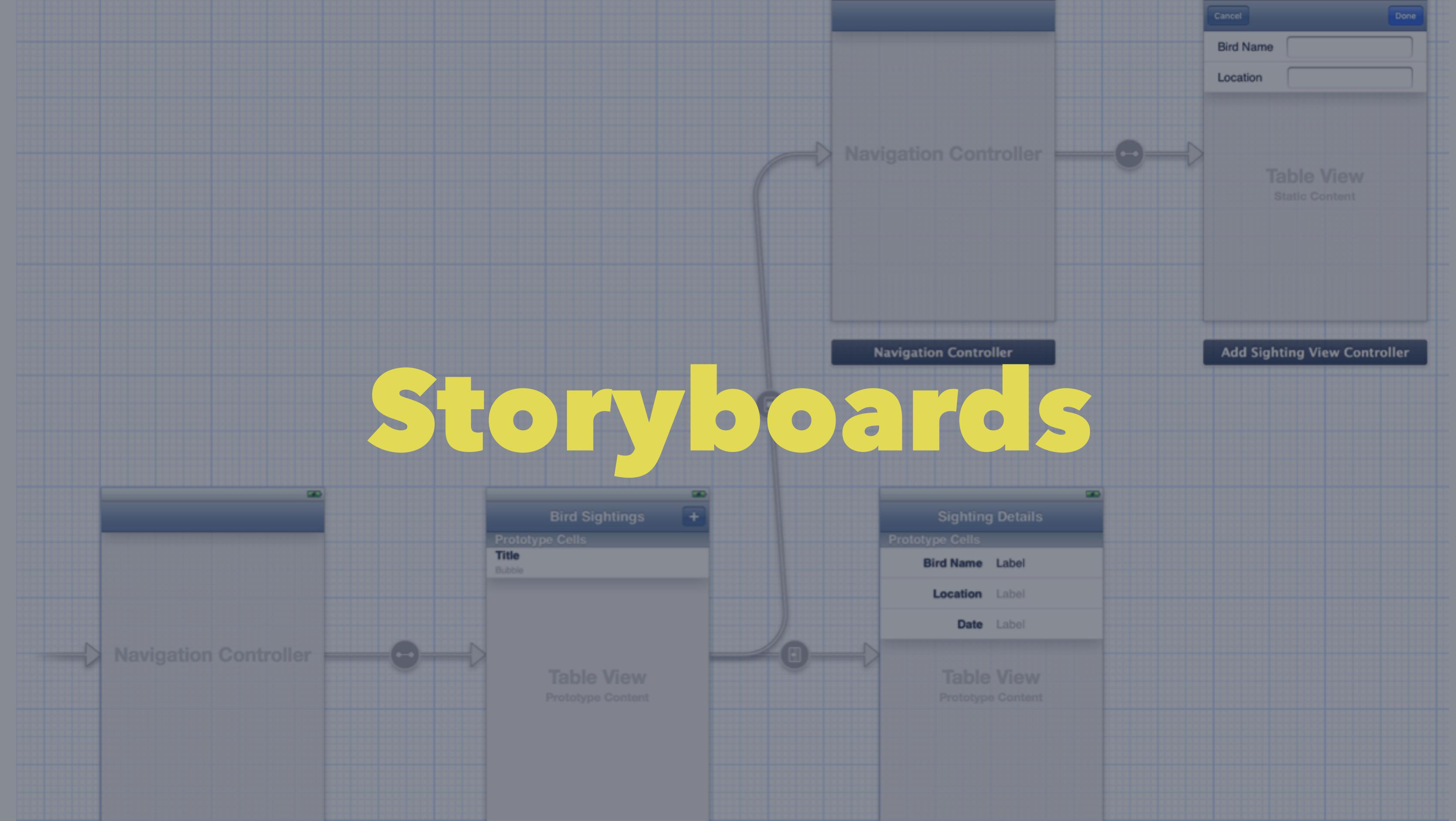
nibName? Var det ikke en XIB?

XIB (uttales sib) og storyboards kompileres til NIB:

The contents of .xib and .storyboard files are stored by Xcode in XML format. At build time, Xcode compiles your .xib and .storyboard files into binary files known as nibs. At runtime, nibs are loaded and instantiated to create new views.

- Apple Docs

Storyboards



Storyboards

- Standardmåten å lagviewsre på
- Kan gi deg et overblikk over flyten i applikasjonen
- Muliggjør drag'n'drop for å koble view controllere sammen
(kalt segues - på norsk: naturlig overgang)
- View controllere instansieres automatisk med verdiene fra interface builder når de trengs

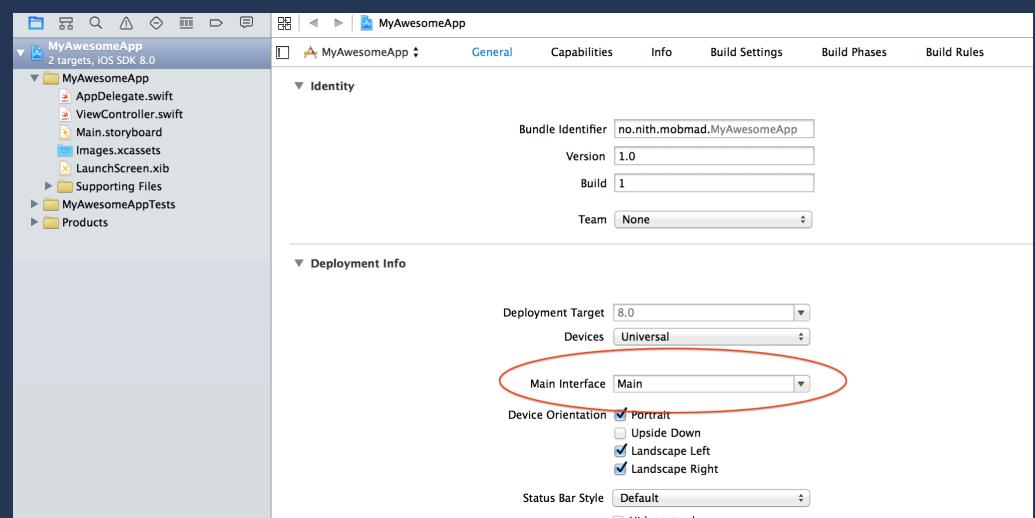
Storyboard eksempel

1. Åpne Main.storyboard og opprett GUI på samme måte som med XIB
2. Oppdater application:didFinishLaunchingWithOptions: til

```
func application(application: UIApplication,  
didFinishLaunchingWithOptions launchOptions: [NSObject: AnyObject]?) -> Bool {  
    return true  
}
```

Storyboard eksempel

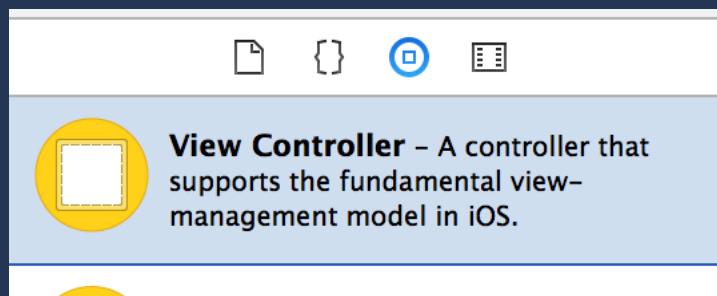
3 - Sett Main interface i prosjektet:



Lage flere view controllere

Lage flere view controller i storyboard

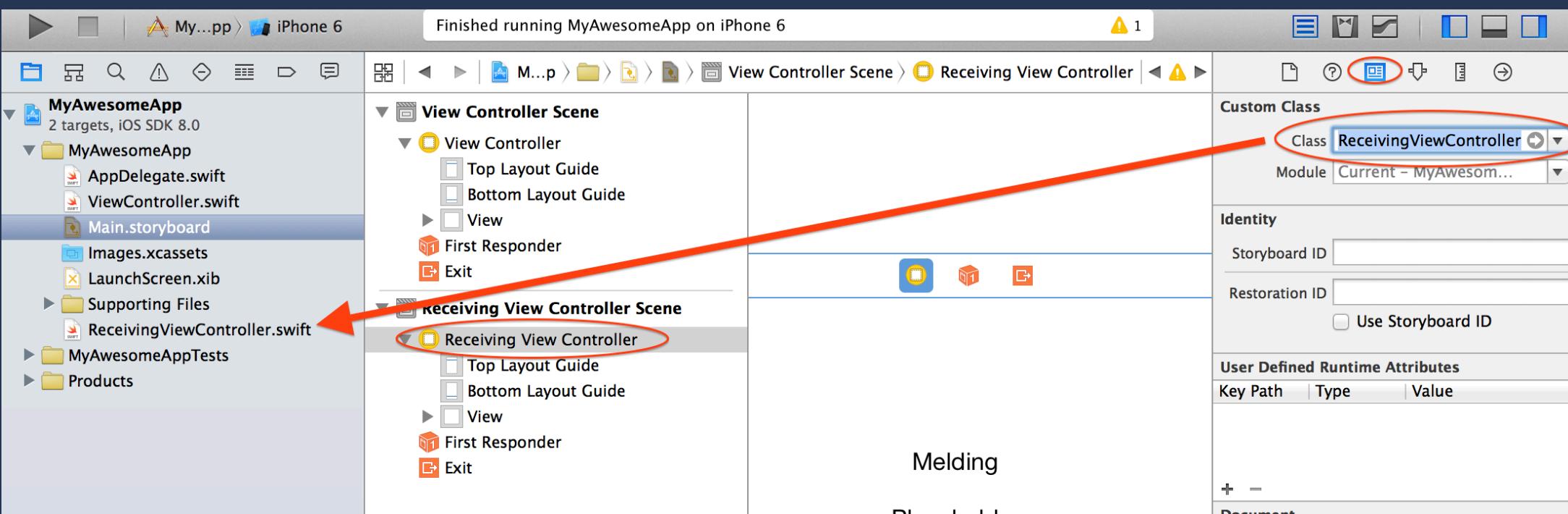
1 - Dra og slipp "View Controller" inn i storyboard fra Object Library:



2 - File -> New -> File -> Cocoa Touch, subclass of UIViewController

Lage flere view controller i storyboard

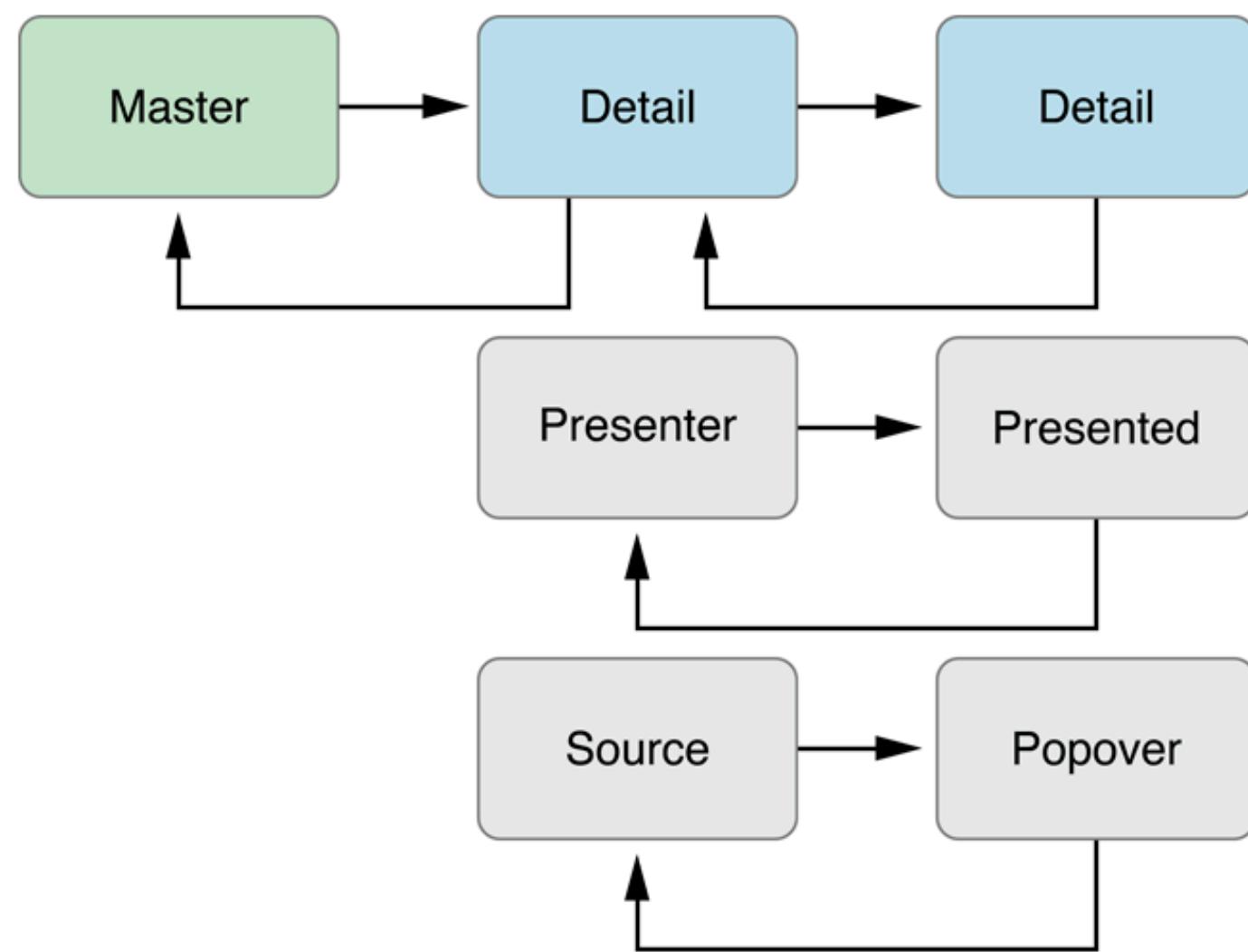
3 - Koble view controller objektet i storyboard sammen med sin faktiske klasse (fra steg 2)



Kommunikasjon mellom view controllere

Kommunisere mellom VC

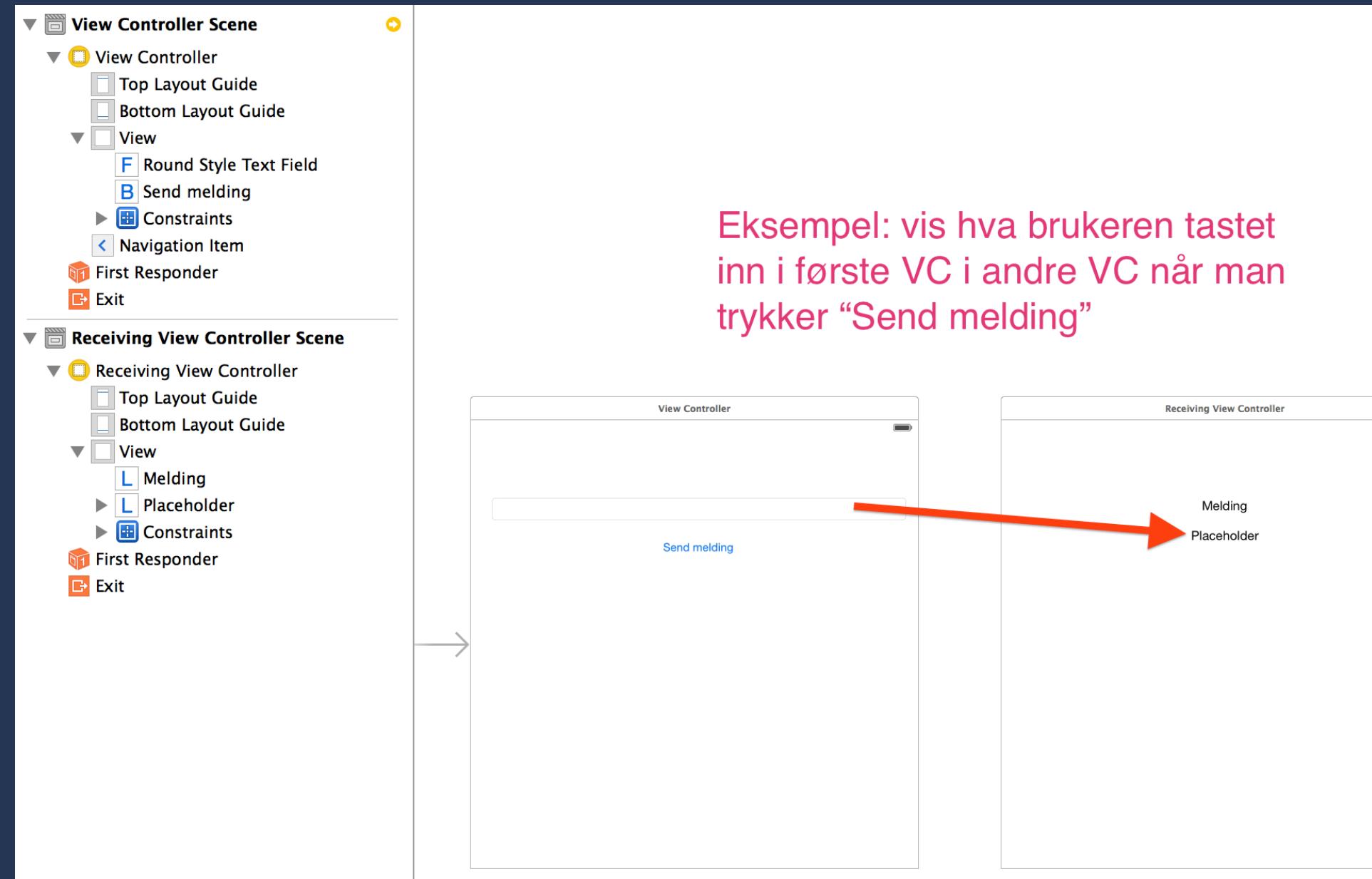
Figure 1-15 Communication between source and destination view controllers



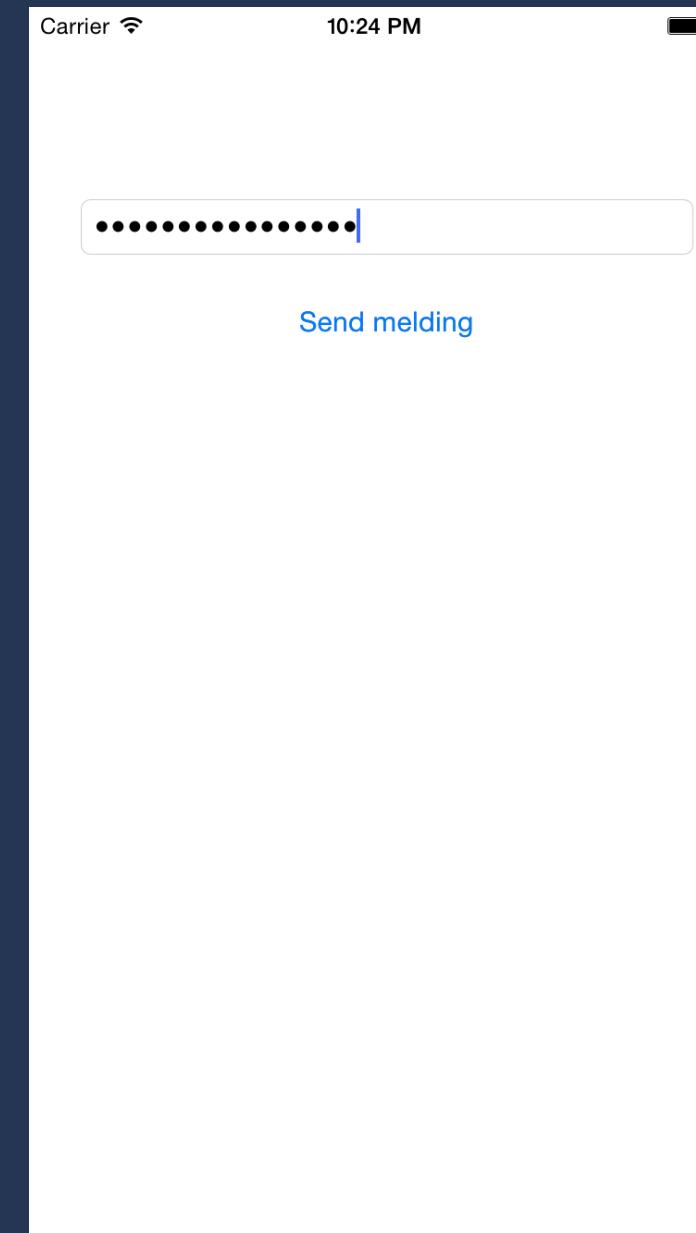
Kommunisere mellom VC

- Destination controller: eksponerer properties for data/presentasjon
- Source: setter properties på destination controller før den blir vist
- Når destination controller har behov for å kommunisere tilbake, gjøres dette via en delegate som sourcen oppgir. Destination controller har dermed ikke direkte kjennskap til sine foreldre

Kommunisere mellom VC



Kommunisere mellom VC



Kommunisere mellom VC



Kommunisere mellom VC (alt 1)

```
import UIKit

class ViewController: UIViewController {

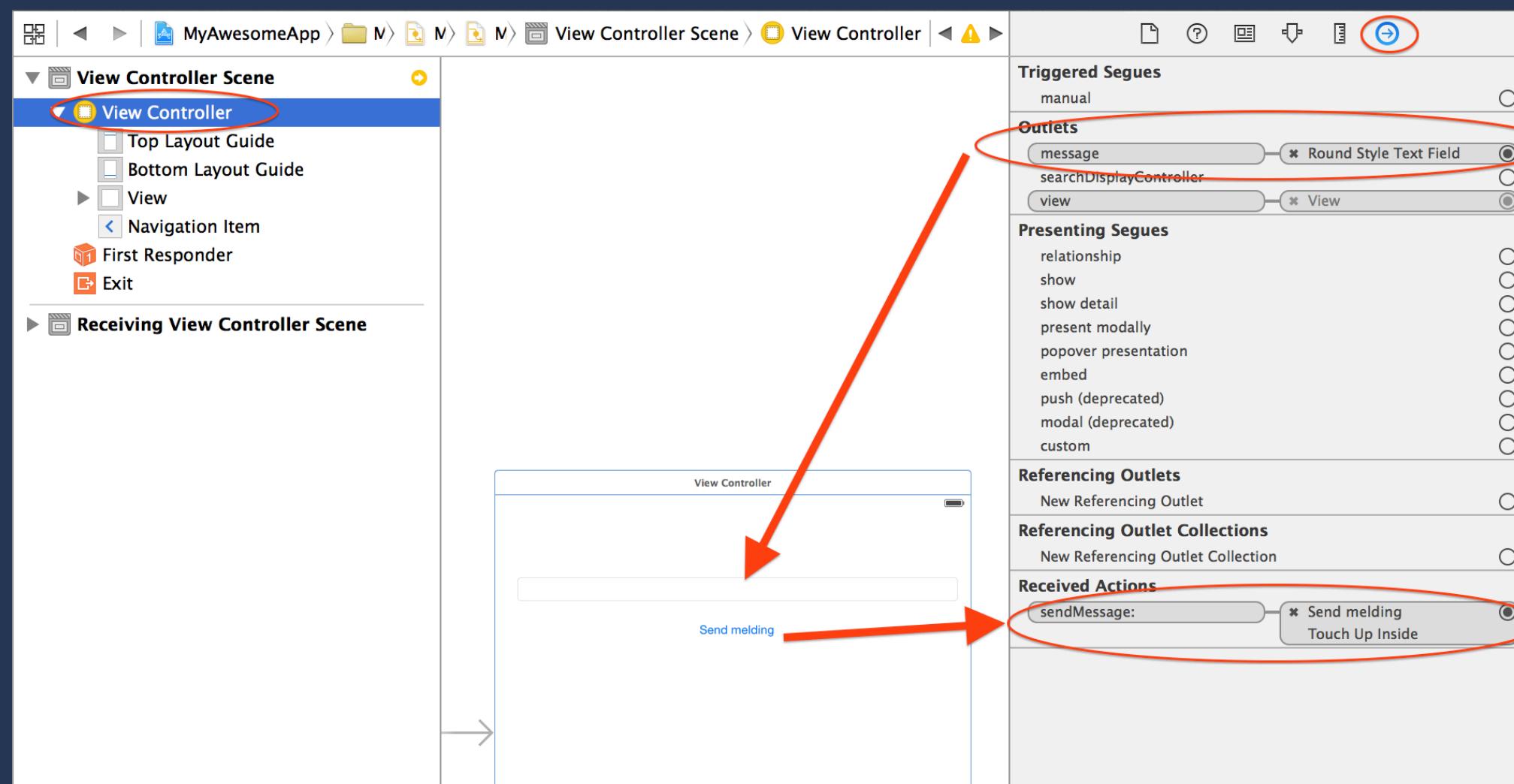
    // Referanse til inputfeltet
    @IBOutlet weak var messageTextField: UITextField!

    // Kalles når man trykker på "Send melding"
    @IBAction func didTapSendMessageButton(sender: AnyObject) {

        // Instansierer ny vc fra storyboard MANUELTT for å vise hvordan
        // Normalt lar vi storyboard gjøre dette for oss. Se alt 2. senere.
        if let receivingViewController = storyboard?.instantiateViewControllerWithIdentifier("receivingVC")
            as? ReceivingViewController {
            // Setter property på destination vc
            receivingViewController.message = messageTextField.text!
            // Før den vises
            presentViewController(receivingViewController, animated: false, completion: nil)
        }
    }
}
```

Kommunisere mellom VC (alt 1)

Connections inspector skal se slik ut:



Kommunisere mellom VC (alt 1)

```
// Den andre view controlleren
class ReceivingViewController: UIViewController {

    // Denne må settes av parent view controller
    var message: String?

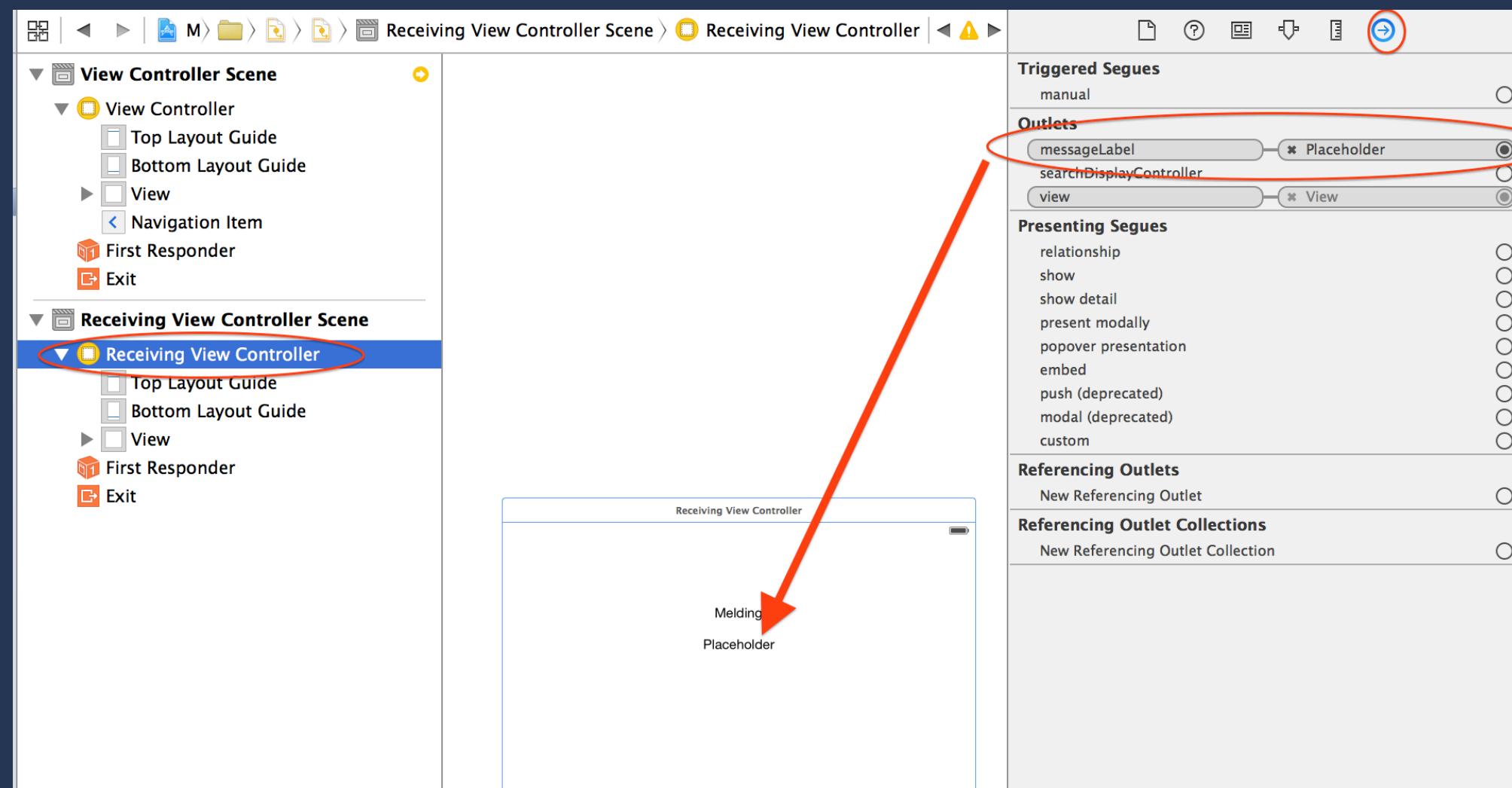
    @IBOutlet weak var messageLabel: UILabel!

    // Konfigurerer viewet, etter at det er lastet inn i minnet
    // Funker både for views som lastes fra nib (storyboard/xib)
    // og for de laget med ren kode (loadView:)
    override func viewDidLoad() {
        super.viewDidLoad()

        messageLabel.text = message
    }
}
```

Kommunisere mellom VC (alt 1)

Connections inspector skal se slik ut:

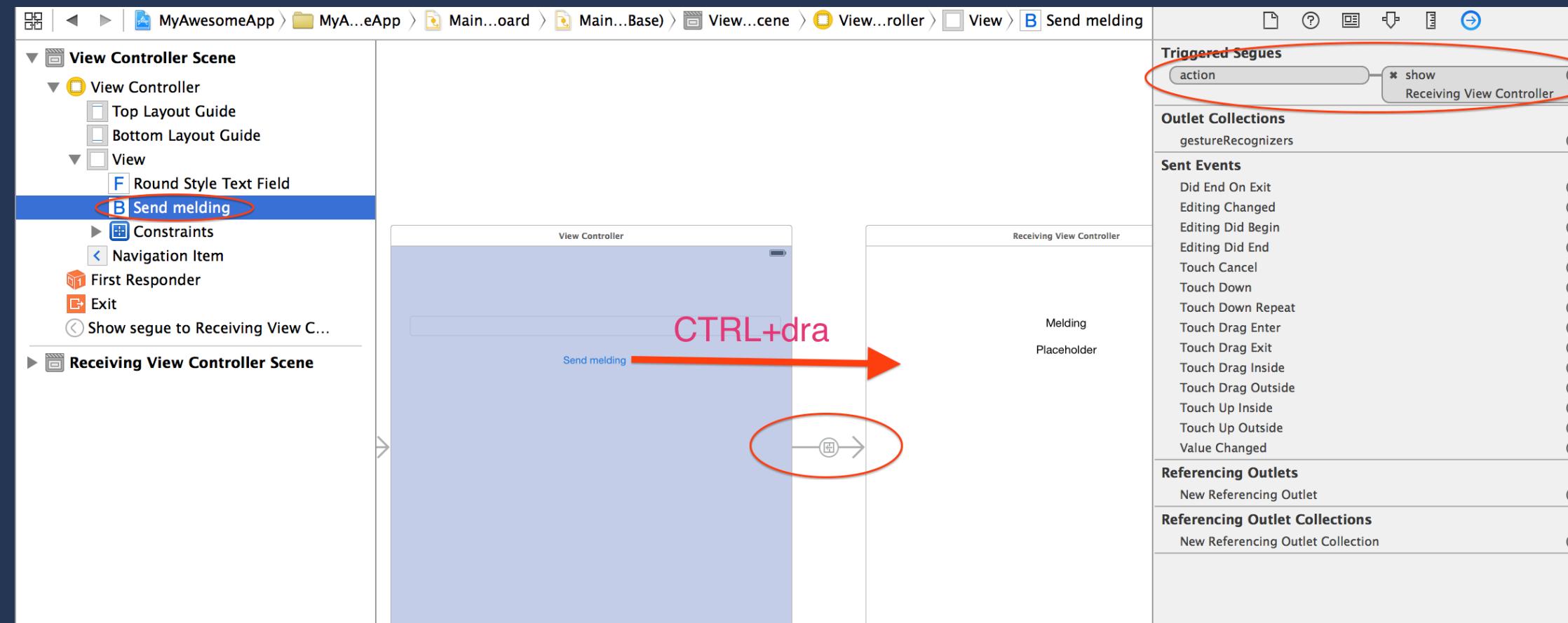


Yay!

Men det finnes en enkelere måte?

Storyboard og segues

Kan lage connections mellom view controllere med drag'n'drop i storyboard. Disse overgangene kalles segues.



Kommunisere mellom VC (alt 2)

```
// Kun denne view controlleren oppdateres. ReceivingViewController  
// er uendret  
class ViewController: UIViewController {  
  
    @IBOutlet weak var message: UITextField!  
  
    // Kalles før overgangen skjer  
    override func prepareForSegue(segue: UIStoryboardSegue, sender: AnyObject!) {  
  
        if let vc = segue.destinationViewController as? ReceivingViewController {  
            vc.message = message.text  
        }  
    }  
}
```

Videre lesning:

- Standford iOS Development forelesning 2
- <https://itunes.apple.com/us/course/developing-ios-8-apps-swift/id961180099>
- The basics i iOS 8 programming in Swift

Oppgaver

**Set It's Learning
Forelesningen er basert på fjorårets foiler, laget av
Hans Magnus Inderberg og Mads Møbæk**