

# Beginning iOS 10 Application Development

## Introduction

Yanping Zhao  
Nov. 2016

# About Me

- Yanping Zhao
- A freelance iOS developer
- 8 years iOS app development experience
- Email: yanping\_zhao@sina.cn



# iOS App Development Basics



“Making a Difference. One App at a Time”

# What is iOS?



The first and one of the most popular  
touch-screen mobile operation systems

# iOS Devices — iPhones



**iPhone 7 Plus**  
**iPhone 6x Plus**



**iPhone 7**  
**iPhone 6x**



**iPhone SE**  
**iPhone 5x**



**iPhone 4x**



**iPhone 3x**  
**iPhone**

**5.5in  
Retina**

**1080x1920  
px**

**4.7in  
Retina**

**750x1334  
px**

**4in  
Retina**

**640x1136  
px**

**3.5in  
Retina**

**640x960  
px**

**3.5in**

**320x480  
px**

# iOS Devices — iPads



iPad Pro



iPad Pro  
iPad Air 1/2  
iPad 3/4



iPad 1/2



iPad Mini  
2/3/4



iPad Mini

12.9in  
Retina

2048x2732  
px

9.7in  
Retina

1536x2048  
px

9.7in

1536x2048  
px

7.9in  
Retina

768x1024  
px

7.9in

768x1024  
px

# iOS Devices – Others



**AppleTV 4th  
Generation**  
**tvOS 10**



**Apple Watch**  
**watchOS 3**

# iOS Development Environment



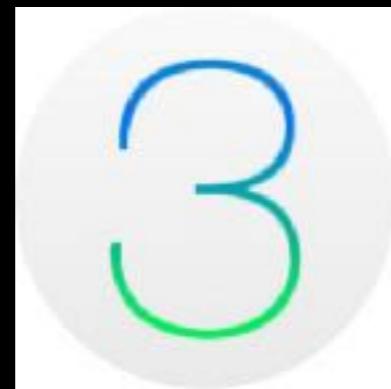
Mac OS X v12 (Sierra)



IDE - XCode 8



iOS 10



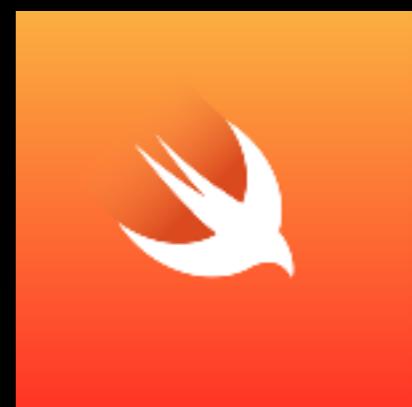
watchOS  
3



tvOS  
10



Programming  
Language



As of June 2016

2 Million Apps

130 Billion Downloads

As of June 2016

# The iPhone was Born

# The iPhone was Born

On January 10th, 2007, Steve Jobs announced the first iPhone and the first iPhone OS



# Revolutionary Features

# Revolutionary Features

- A direct manipulative UI

# Revolutionary Features

- A direct manipulative UI
  - touches, gestures, software keyboard

# Revolutionary Features

- A direct manipulative UI
  - touches, gestures, software keyboard
- New paradigms of user interactions

# Revolutionary Features

- A direct manipulative UI
  - touches, gestures, software keyboard
- New paradigms of user interactions
  - navigations, tab bars, date/time pickers, etc.

# Revolutionary Features

- A direct manipulative UI
  - touches, gestures, software keyboard
- New paradigms of user interactions
  - navigations, tab bars, date/time pickers, etc.
- A mobile web browser -- Safari

# Revolutionary Features

- A direct manipulative UI
  - touches, gestures, software keyboard
- New paradigms of user interactions
  - navigations, tab bars, date/time pickers, etc.
- A mobile web browser -- Safari
- A new distribution channel for mobile applications - App Store

# Revolutionary Features

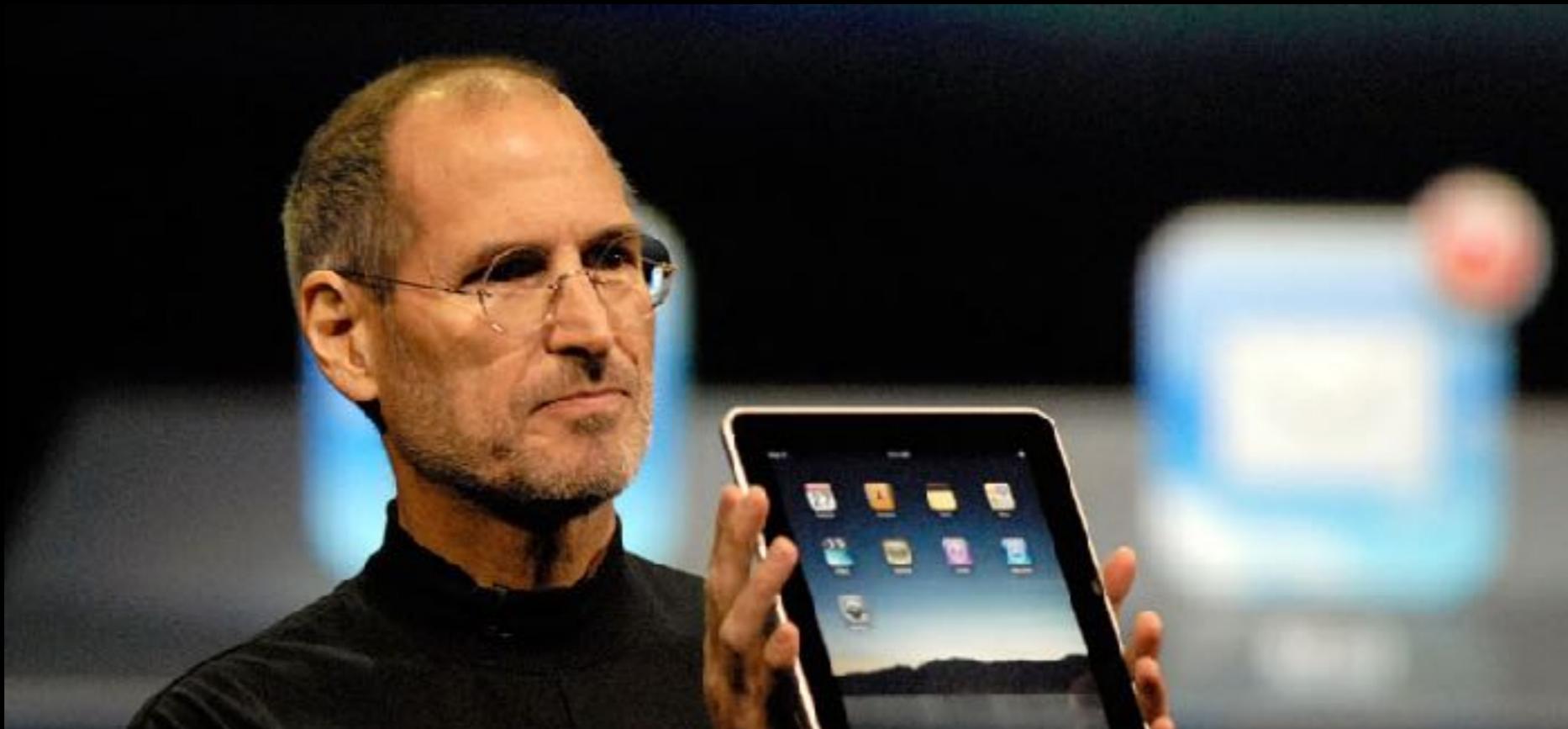
- A direct manipulative UI
  - touches, gestures, software keyboard
- New paradigms of user interactions
  - navigations, tab bars, date/time pickers, etc.
- A mobile web browser -- Safari
- A new distribution channel for mobile applications - App Store



# The iPad Arrived!

# The iPad Arrived!

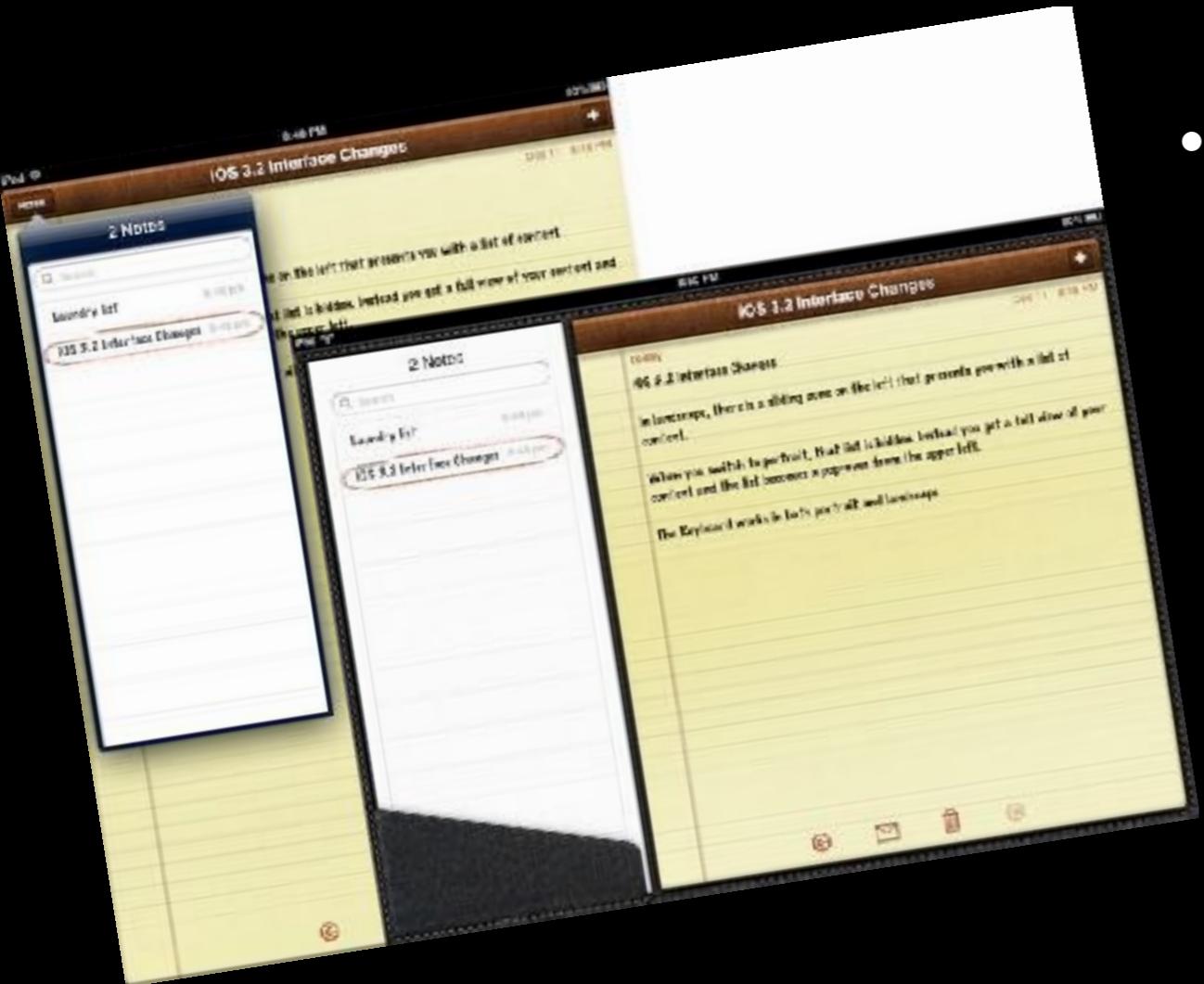
On January 27th, 2010, Steve Jobs announced the iPad.



iPhone OS 3.2

# Revolutionary Features

# Revolutionary Features



- New UI paradigms for wider screen

# Revolutionary Features



- New UI paradigms for wider screen
- A new spectrum of app designs

# Revolutionary Features



- New UI paradigms for wider screen
- A new spectrum of app designs
- iBook - redefined the reading experience and e-book industry

# I was Officially Called “iOS”

On June 20th, 2010, Steve Jobs announced the retina display iPhone4. The “iOS” was branded.



# I was Officially Called “iOS”

On June 20th, 2010, Steve Jobs announced the retina display iPhone4. The “iOS” was branded.



- Multi-tasking

# I was Officially Called “iOS”

On June 20th, 2010, Steve Jobs announced the retina display iPhone4. The “iOS” was branded.



- Multi-tasking
- High-resolution images

# I was Getting Smarter

In WWDC 2011, Steve Jobs announced iOS 5,  
featuring Siri and iCloud



iOS 5.0

# Bye-bye, Google Maps!

In WWDC 2012, Apple announced iOS 6.0 with a fleet of new features



# Bye-bye, Google Maps!

In WWDC 2012, Apple announced iOS 6.0 with a fleet of new features



# Bye-bye, Google Maps!

In WWDC 2012, Apple announced iOS 6.0 with a fleet of new features



# Bye-bye, Google Maps!

In WWDC 2012, Apple announced iOS 6.0 with a fleet of new features



# Bye-bye, Google Maps!

In WWDC 2012, Apple announced iOS 6.0 with a fleet of new features



# Bye-bye, Google Maps!

In WWDC 2012, Apple announced iOS 6.0 with a fleet of new features



# I was Growing Taller

On September 12 2012, Apple announced iPhone 5 with a 4-inch screen



# I was Growing Taller

On September 12 2012, Apple announced iPhone 5 with a 4-inch screen



- a new size for images

# iOS7 - A Massive Overhaul

In WWDC 2013, Apple announced the flat UI iOS 7.0



# iOS8 - A New Language Swift

In WWDC 2014, Apple announced iOS 8.0



Photos



Messages



Design



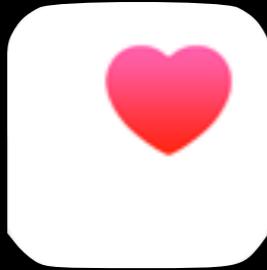
QuickType



Family Sharing



iCloud



Health Kit



Handoff



Swift



Metal

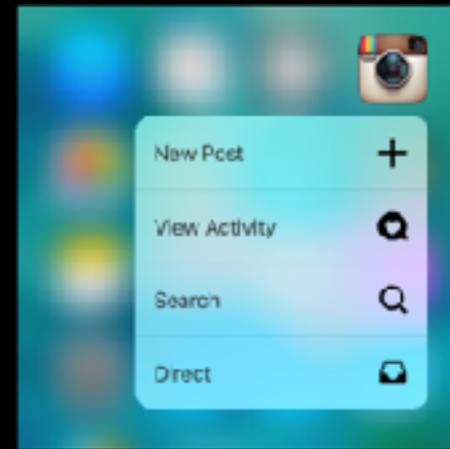
# I Have Bigger Screens!

On September 9 2014, Apple announced 4.7-inch iPhone 6 and 5.5-inch iPhone 6 Plus, and Apple Watch



# iOS9 – Enhancements in all Aspects

In WWDC 2015, Apple announced iOS 9.0.



# I Have New Family Members!

On September 9 2015, Apple announced 4.7-inch iPhone 6s and 5.5-inch iPhone 6s Plus, iPad Pro, Apple Watch 2, and tvOS



# A First Water-Resistant iPhone

On September 7 2016, Apple announced 4.7-inch iPhone 7 and 5.5-inch iPhone 7 Plus, Apple Watch Series 2

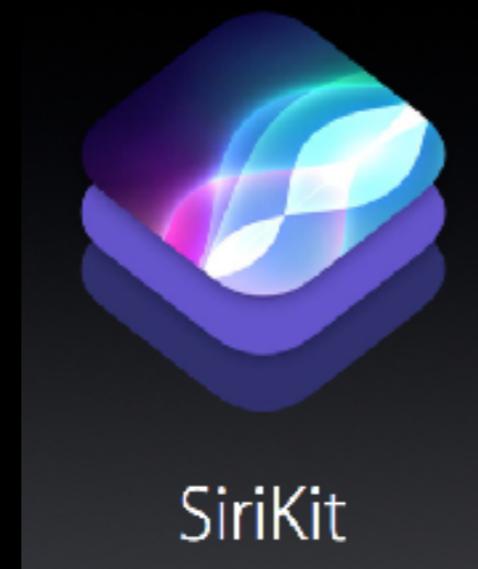


# iOS 10 – Opening Swift, Siri, iMessage

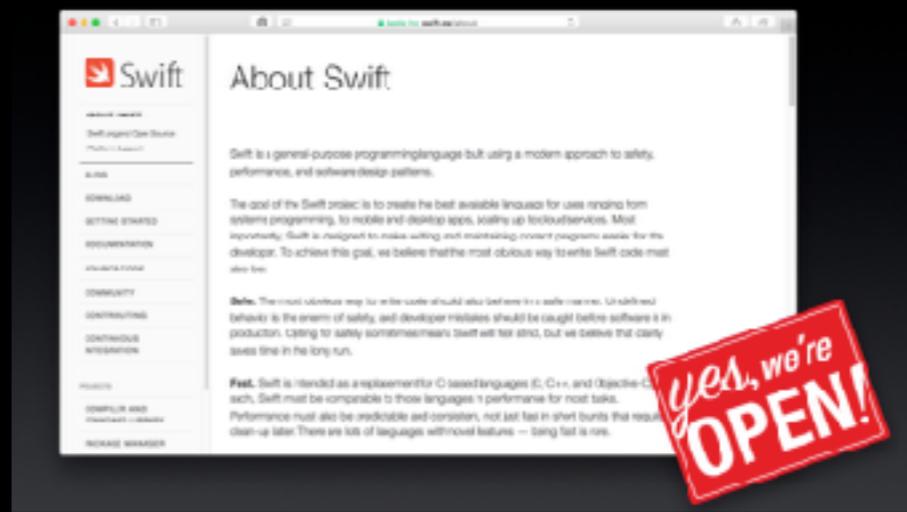
In WWDC 2016, Apple announced iOS 10.0.



iMessage Apps



SiriKit



# Class Schedule

# Class Schedule

	Schedule
1	A Single-View App
2	
3	A Table-based App with Navigation
4	Persistence & Networking
5	A Location-based App with MapKit
6	A SpriteKit Game
7	A WatchOS App
8	Q/A. Final exam.
Dec. 3rd	Project Demo and Presentation

# Evaluation

	<b>Weight</b>	<b>Due Day</b>
Assignments 1	15%	Midnight, Nov. 27
Assignment 2	20%	Midnight, Dec. 2
Course Project	50%	
Participation	15%	

# Class Participation

- Students must attend every classes
- The attendance will be enforced
  - Early leave/dismissal is not allowed
  - Grades will be subtracted.
- If you cannot attend a class due to reasons like sickness, please send me an email before the class
- If you cannot attend the final exam for any reasons, please get advanced approval from the graduate school

# Course Project

- The course project can be done in a group of no more than 3 students
- Identify a group leader
- Each group needs to make one app, which can be an iOS app, an Apple Watch app, or a game app
- The app must demonstrate the use of iOS 10, watchOS 3, or SpriteKit, Swift 3 elements
- Everyone needs to present his own work and clarify his own contributions
- The project can be guided or self-picked
- The project must be developed in Swift 3

# Reference

# Reference

- iOS Developer Library (<http://developer.apple.com/library>)

# Reference

- iOS Developer Library (<http://developer.apple.com/library>)
- [swift.org](http://swift.org)

# Reference

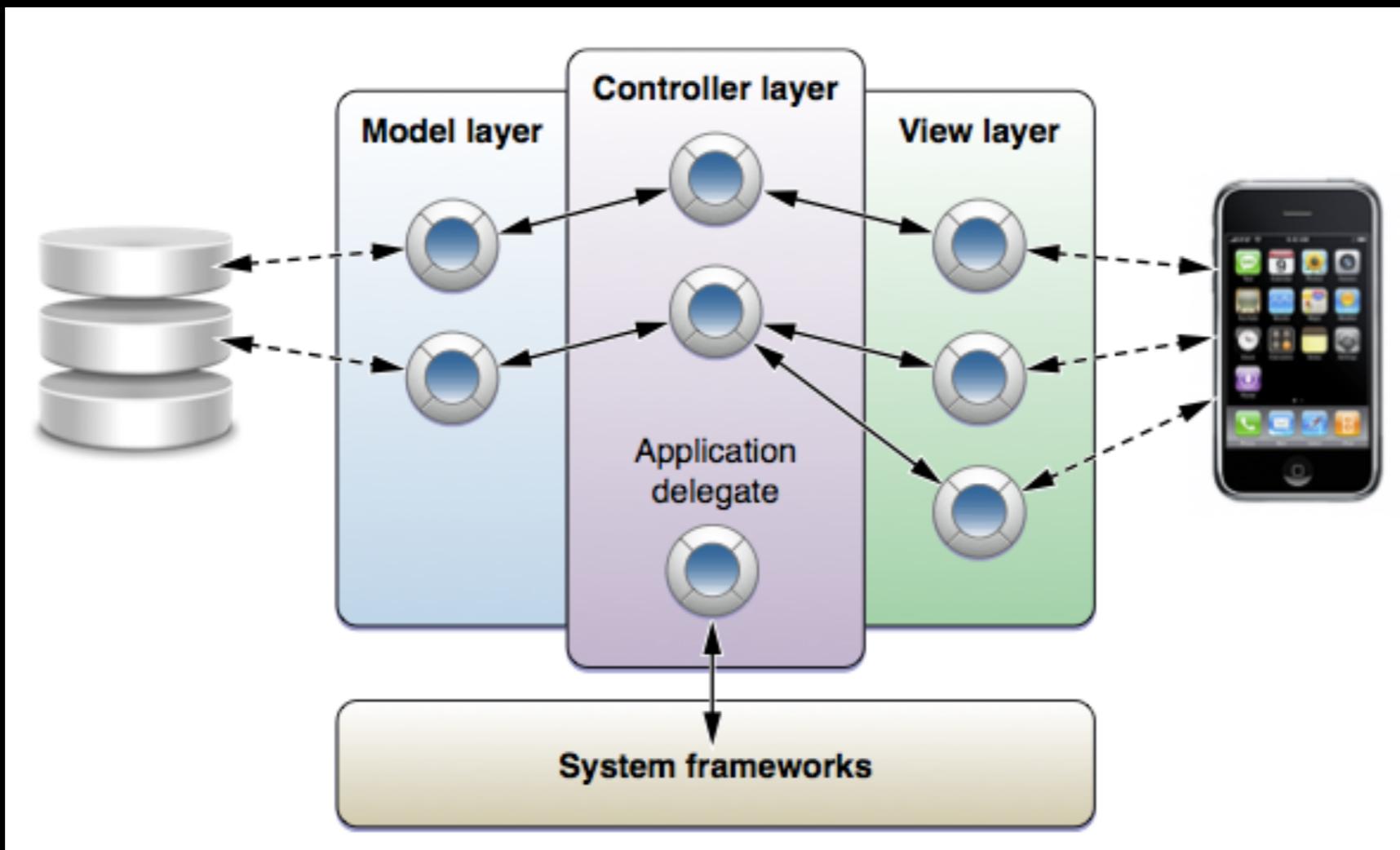
- iOS Developer Library (<http://developer.apple.com/library>)
- [swift.org](http://swift.org)
  - The Swift Programming Language (Swift 3.0.1)

# Class FTP Site

- ▶ <ftp://202.205.101.168/>
  - user name: zyp\_s, no password
  - submit assignments, download class materials

# iOS Design Patterns

# Model-View-Controller Design Pattern



# Views

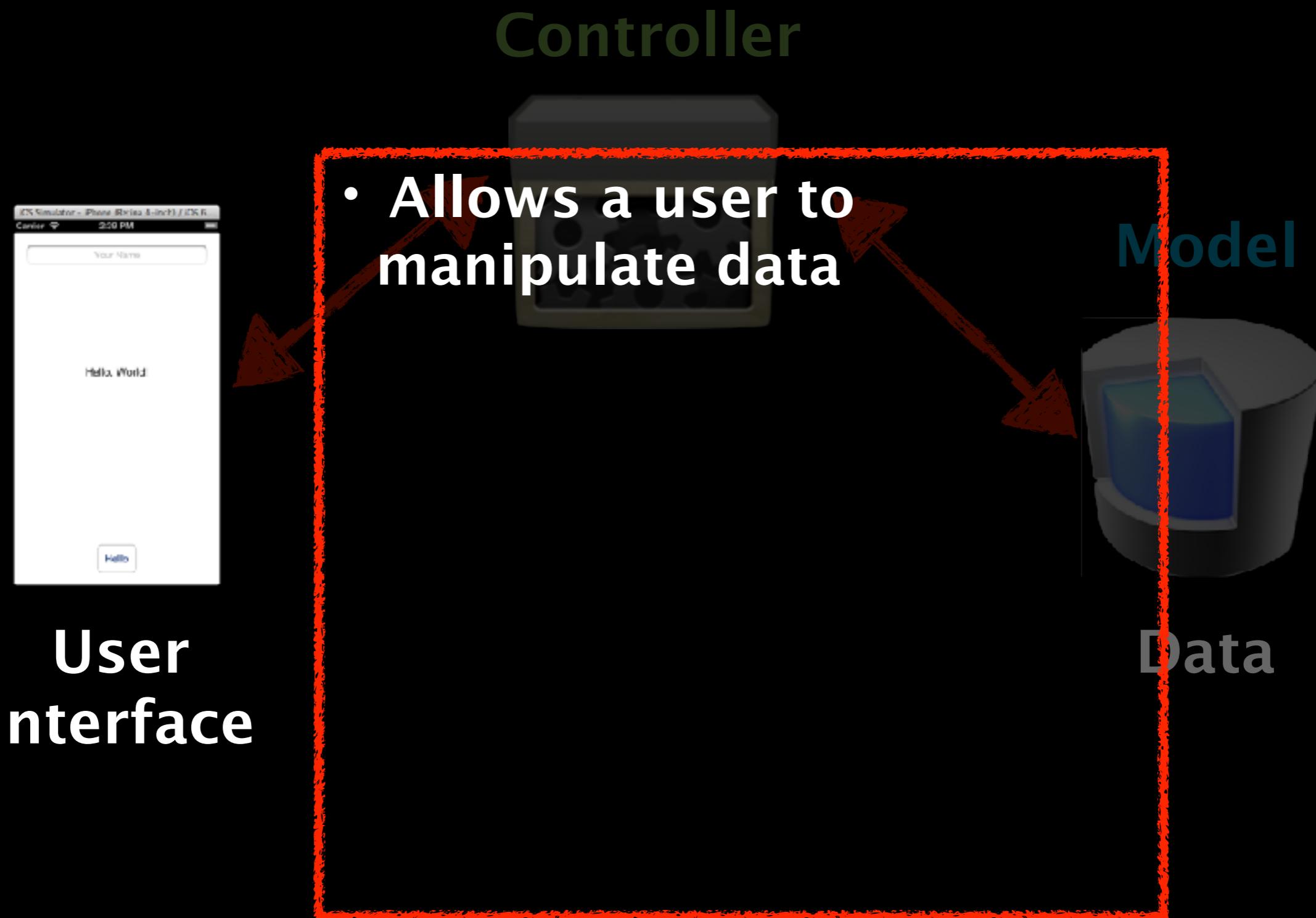
Controller



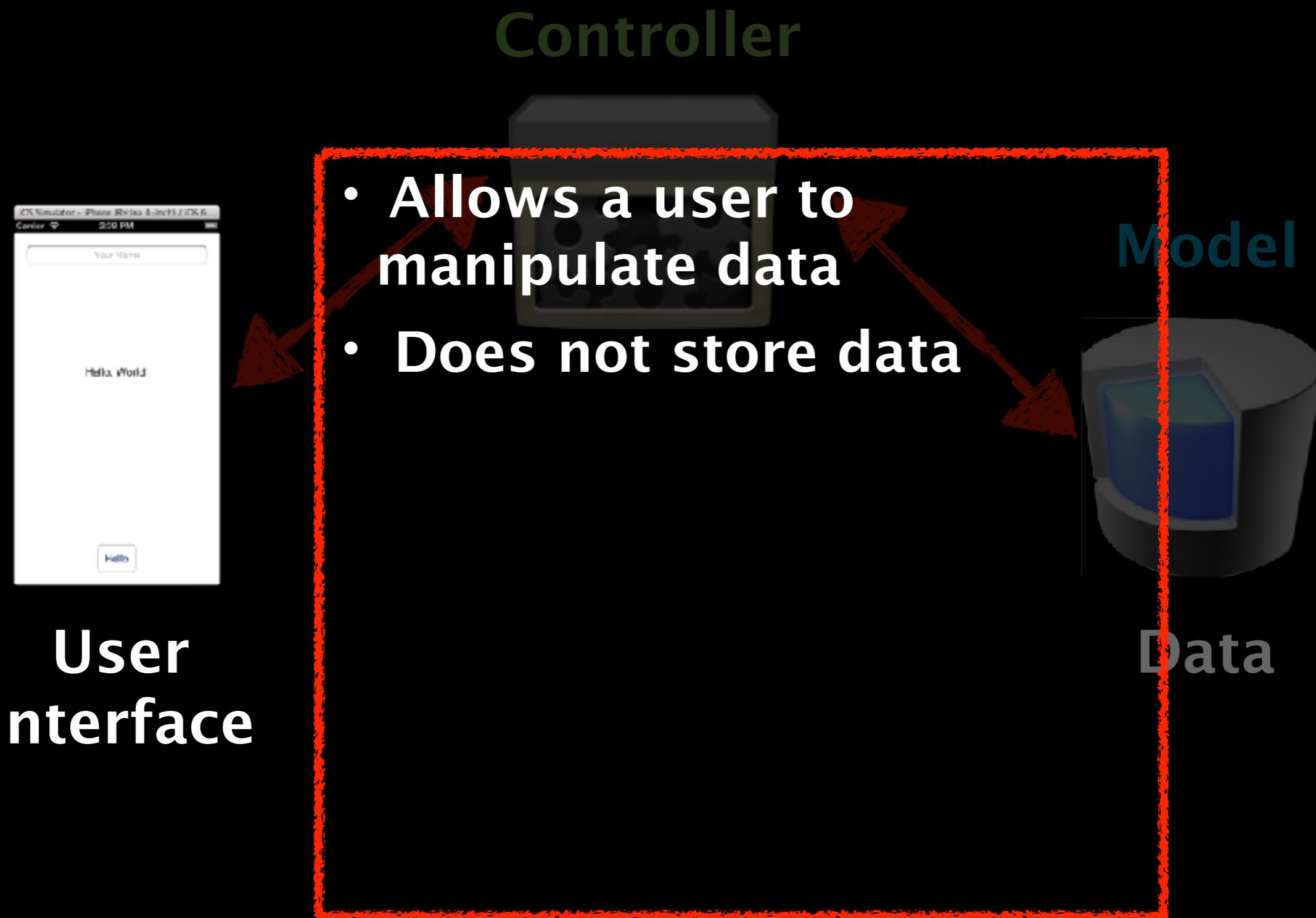
User  
Interface

Data

# Views



# Views



# Views



# Views

- View hierarchy, geometry
- UIKit - built-in controls and elements
- Auto-layout - how views are laid out
- View animations/transitions



# Model

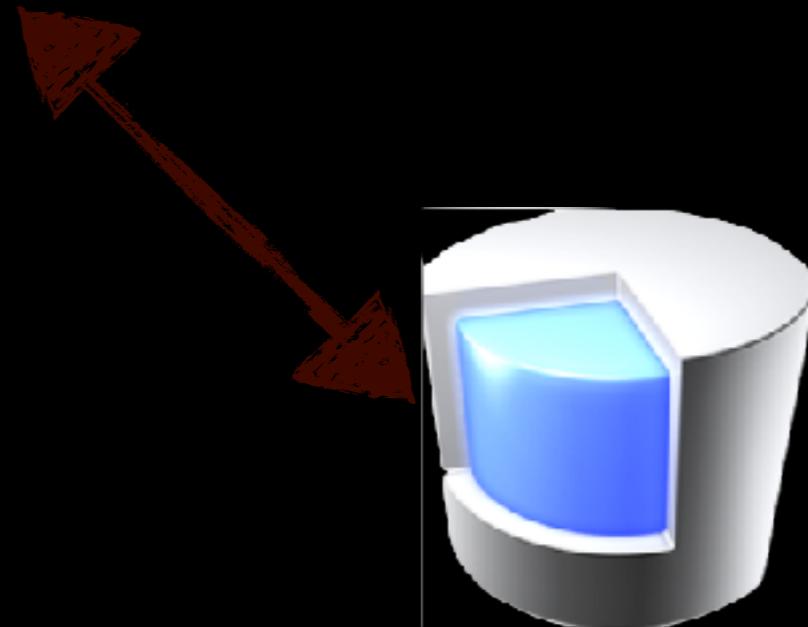
## Controller

### View



User  
Interface

Data

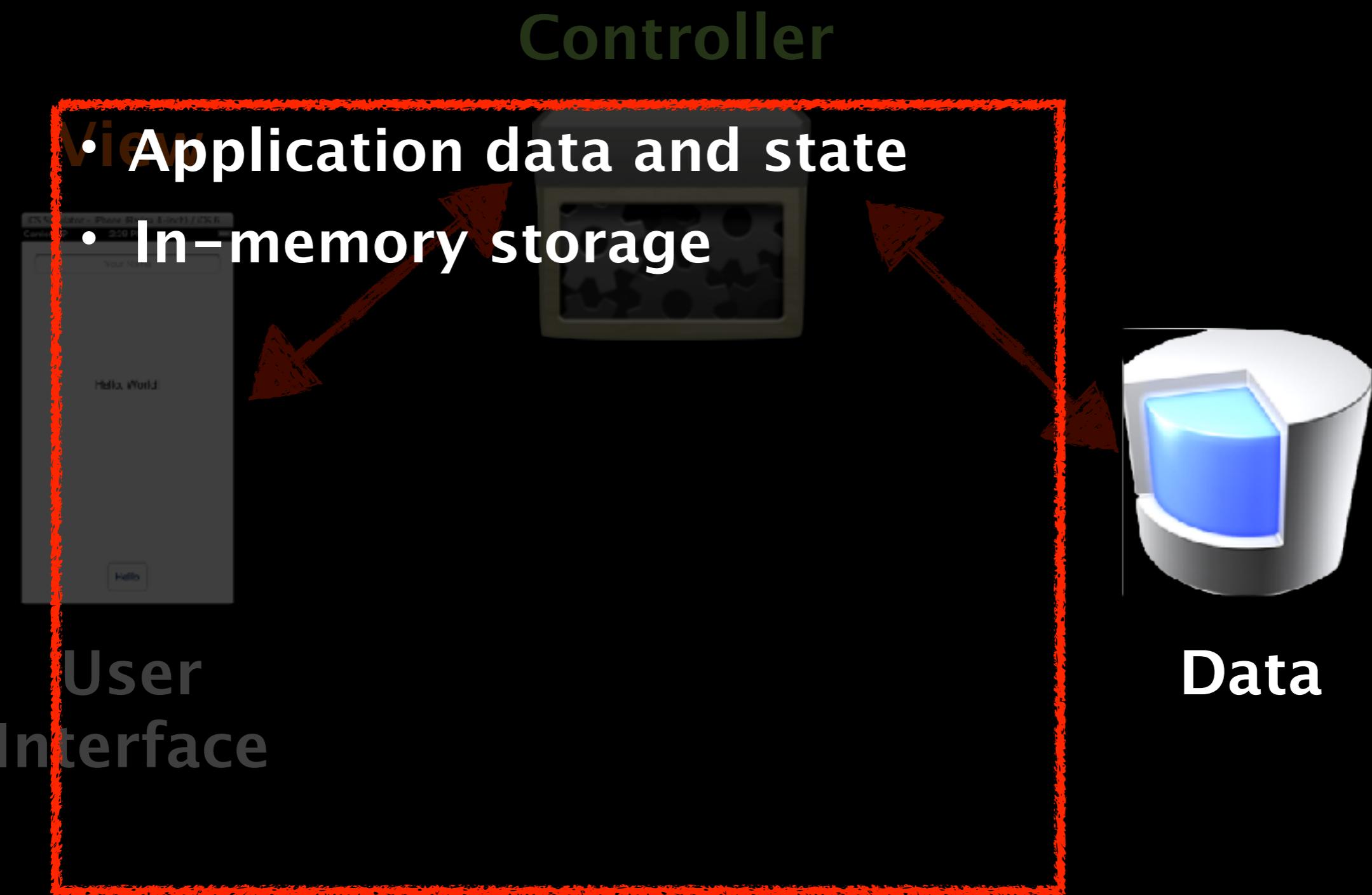


# Model

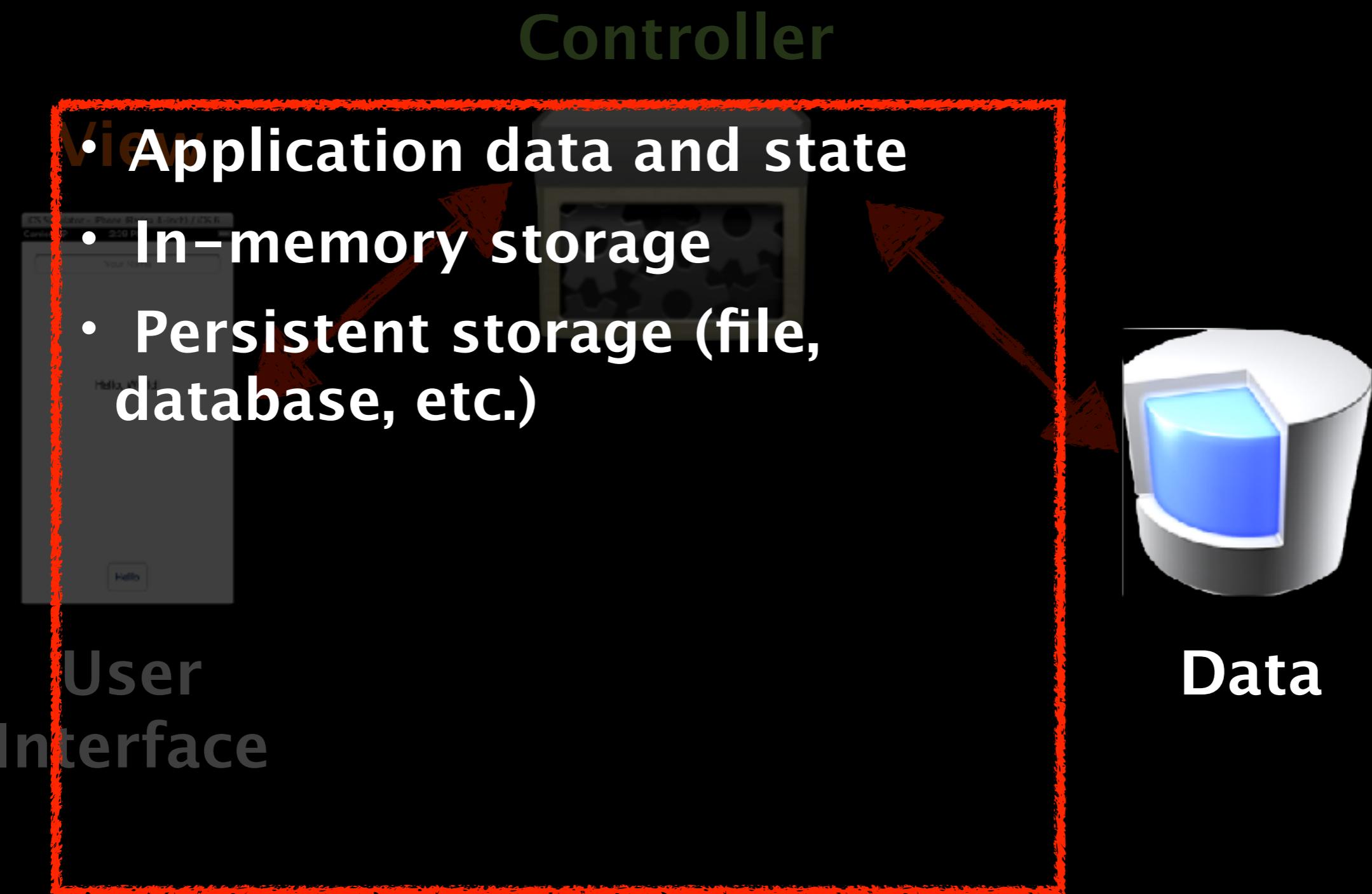
## Controller



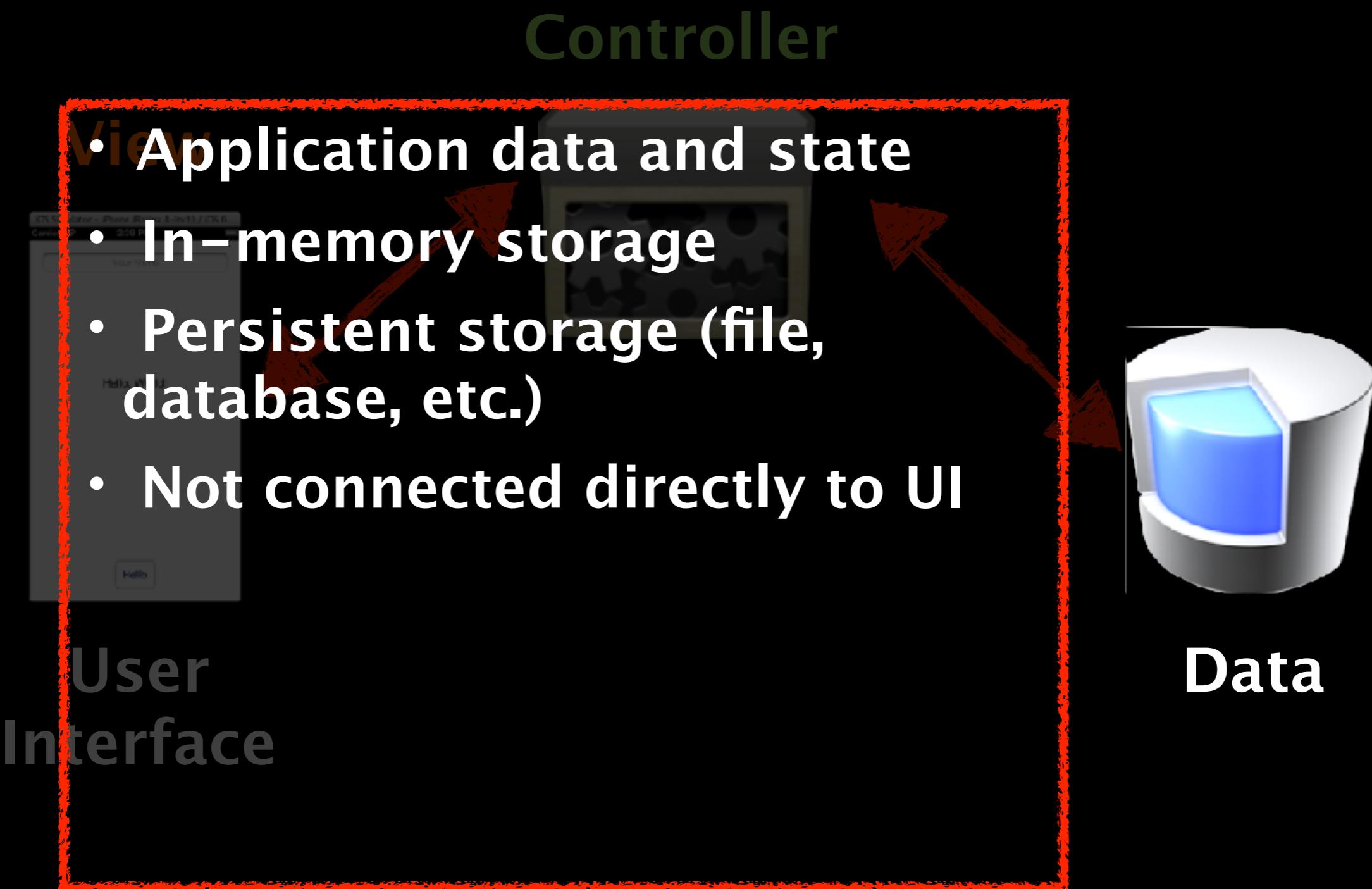
# Model



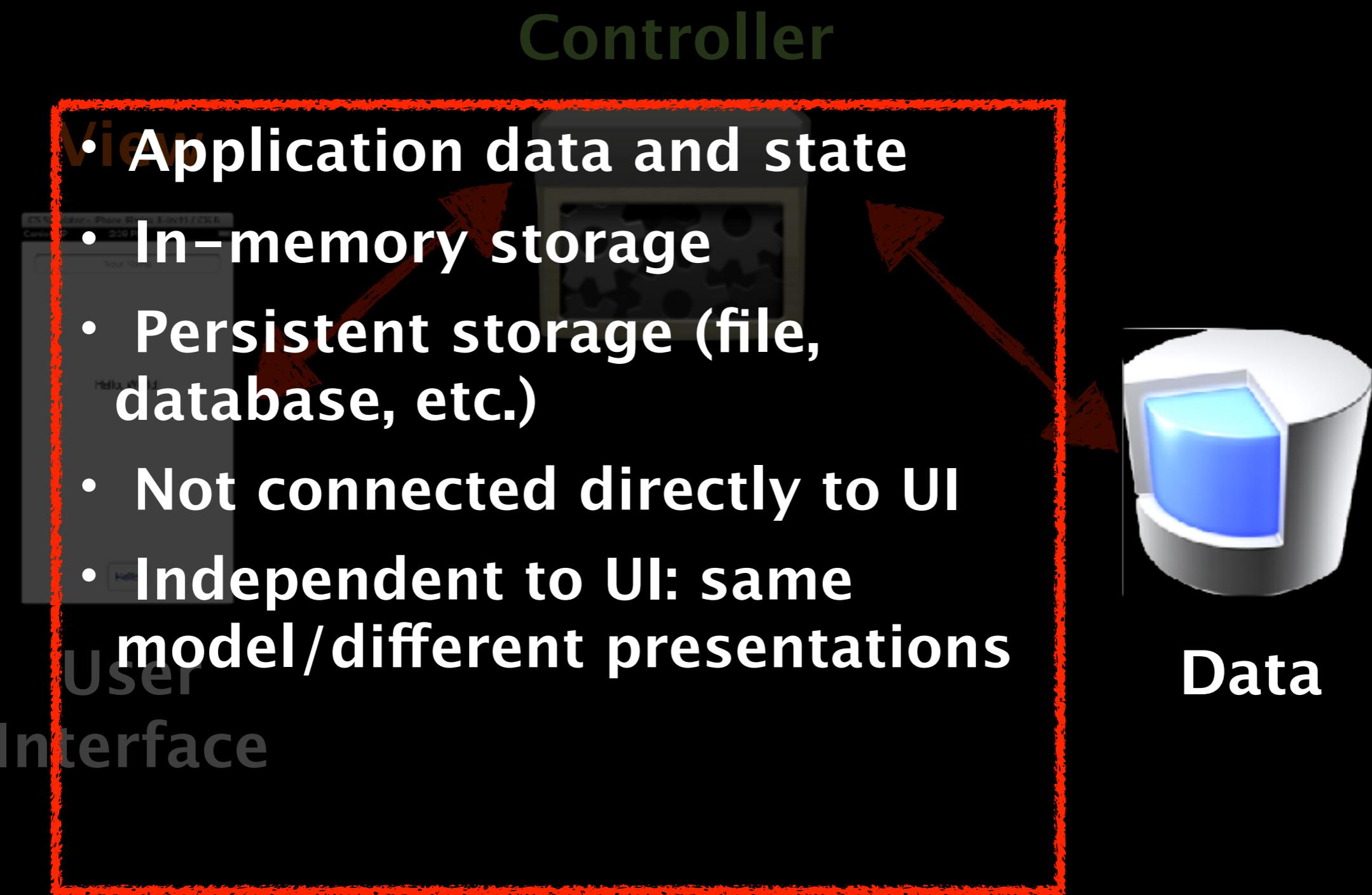
# Model



# Model



# Model



# Model

# Model

- Property list - a formatted file

# Model

- Property list - a formatted file
- Object archiving - store data in a binary format

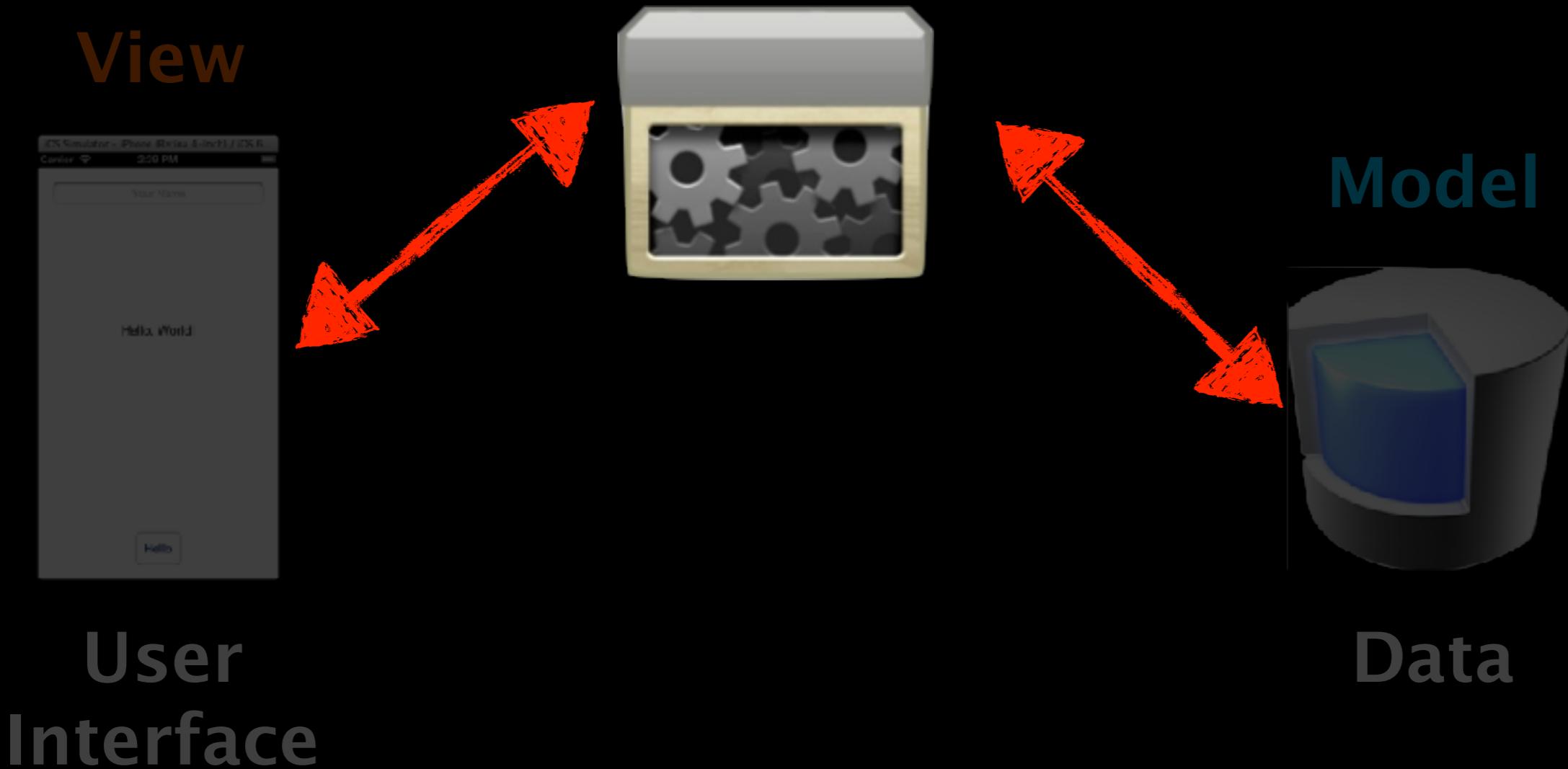
# Model

- Property list - a formatted file
- Object archiving - store data in a binary format
- Core data - store data in a database

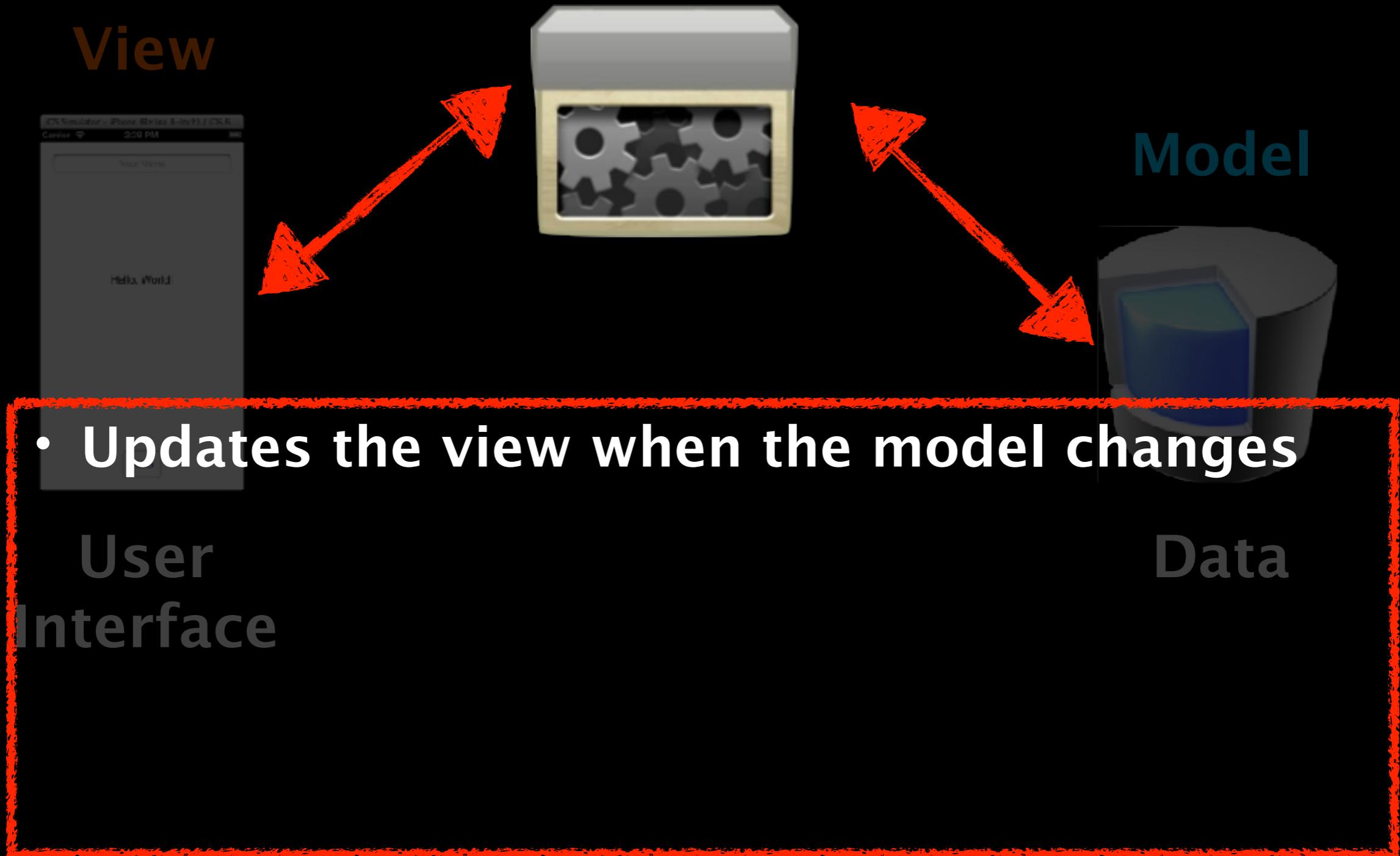
# Model

- Property list - a formatted file
- Object archiving - store data in a binary format
- Core data - store data in a database
- User defaults - store data in system “Settings”

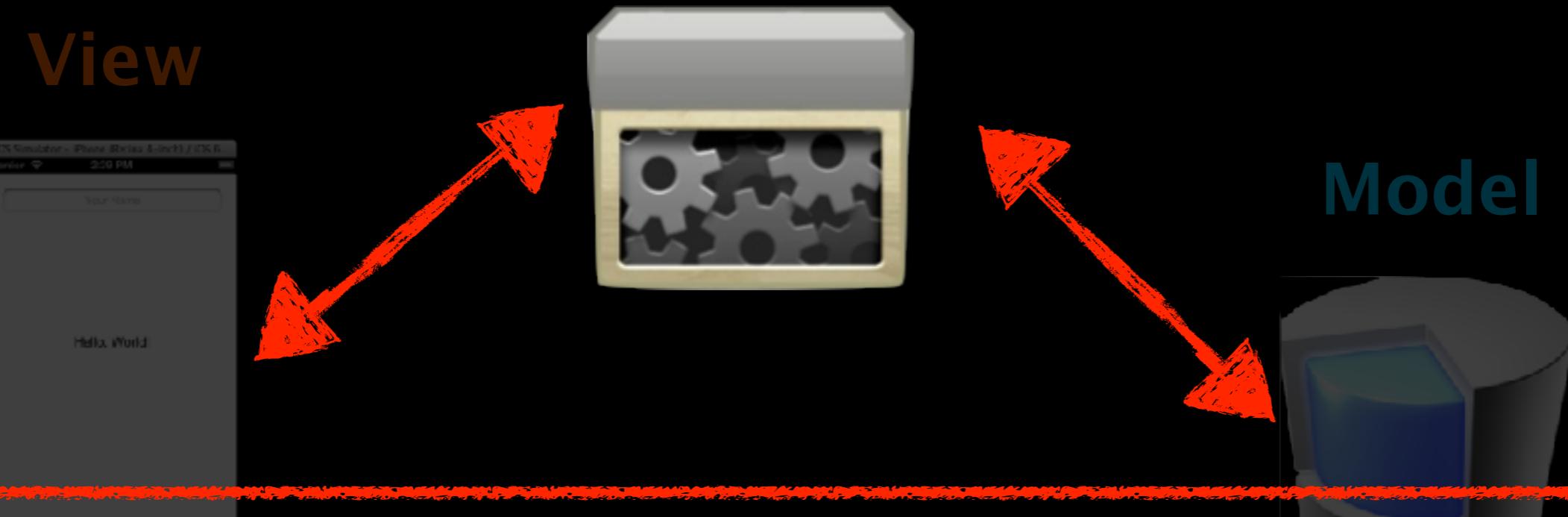
# View Controllers



# View Controllers



# View Controllers



- Updates the view when the model changes
- Updates the model when the user manipulates the view

User  
Interface

Data

# View Controllers



- Updates the view when the model changes
- Updates the model when the user manipulates the view
- Typically where the application logic lives

User  
Interface

Data

# View Controllers

# View Controllers

- View controllers manage views

# View Controllers

- View controllers manage views
- **Content** view controllers display data

# View Controllers

- View controllers manage views
- **Content** view controllers display data
- **Container** view controllers arrange the contents of other view controllers

# Outlets

# Outlets

view



# Outlets

view

Button

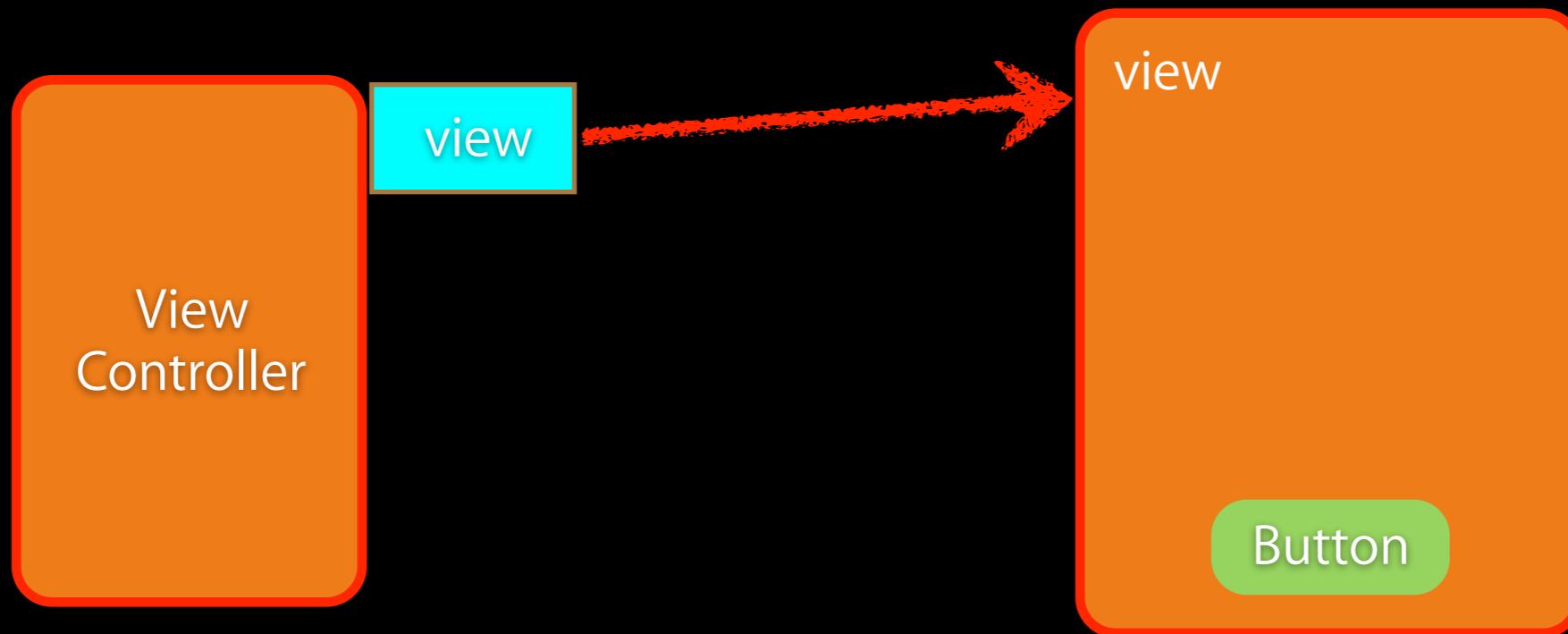
# Outlets



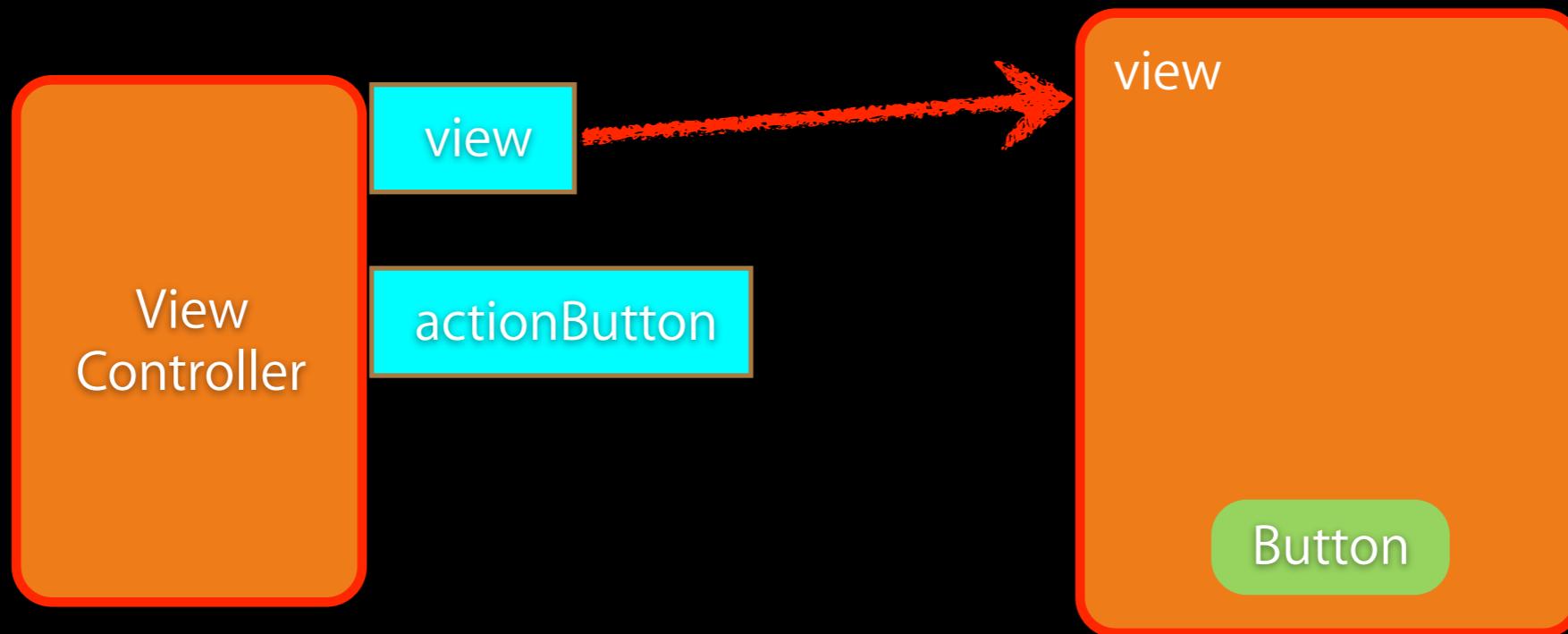
# Outlets



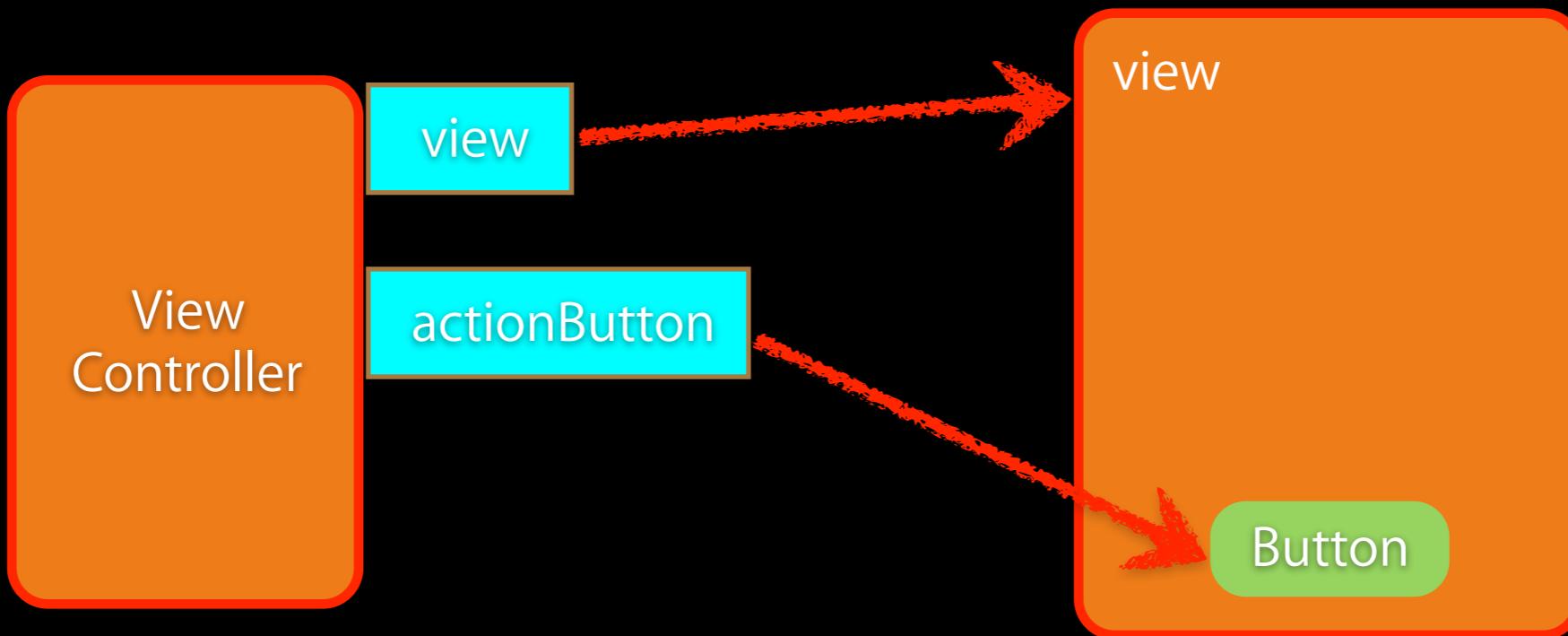
# Outlets



# Outlets

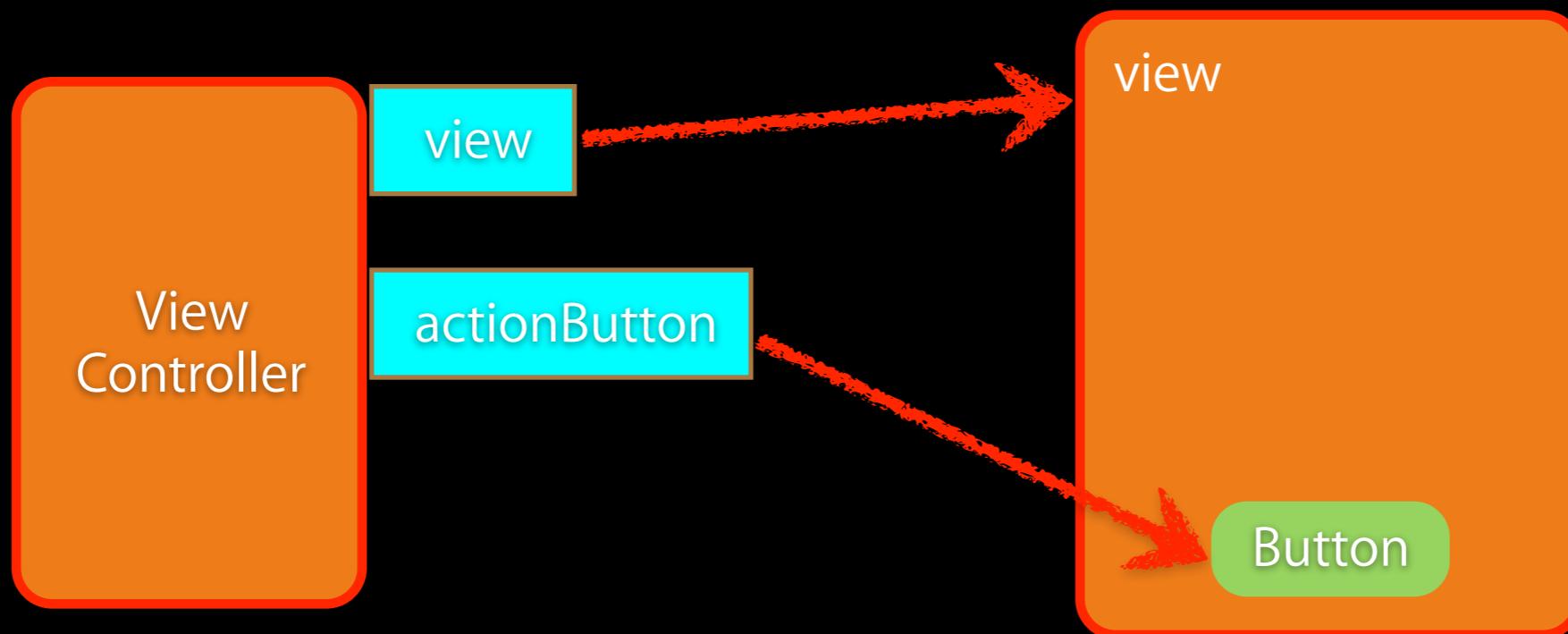


# Outlets



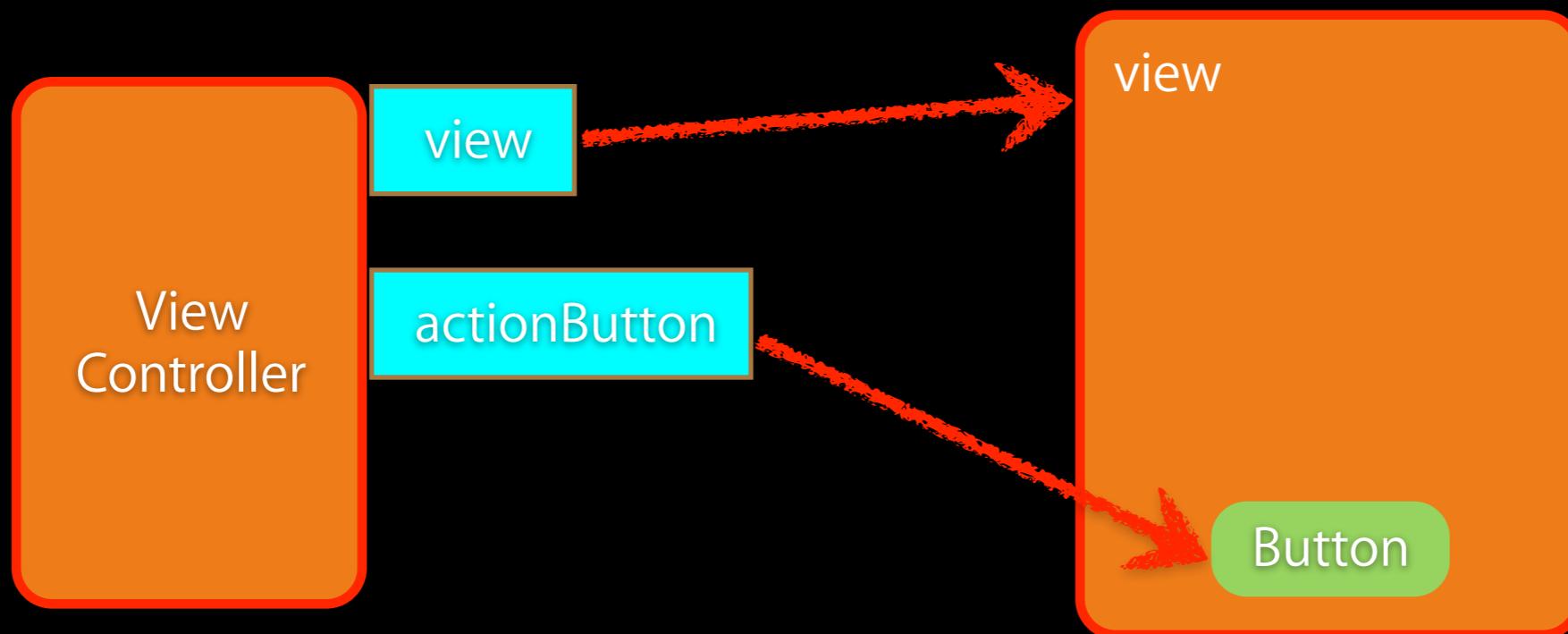
# Outlets

- View controllers use outlets (instance variables) to refer to view components



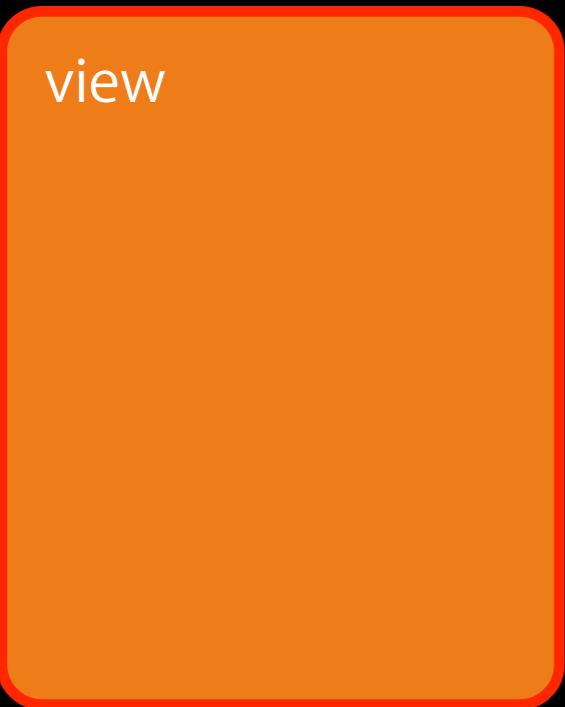
# Outlets

- View controllers use outlets (instance variables) to refer to view components



# Target-Action Design Pattern

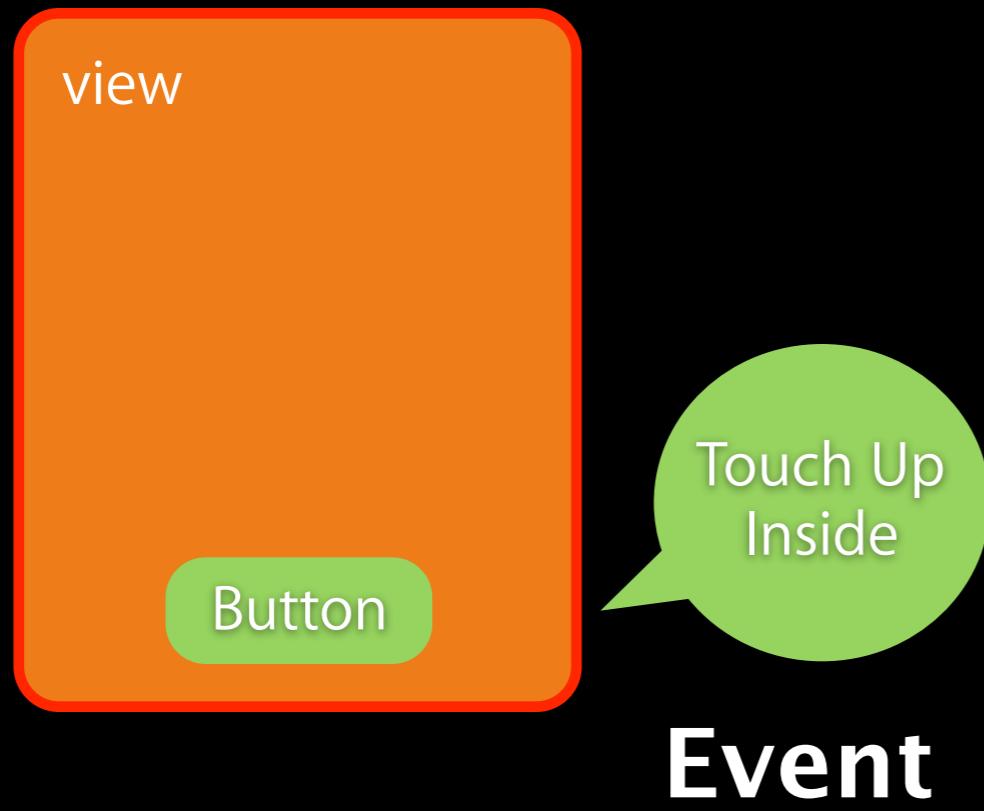
# Target-Action Design Pattern



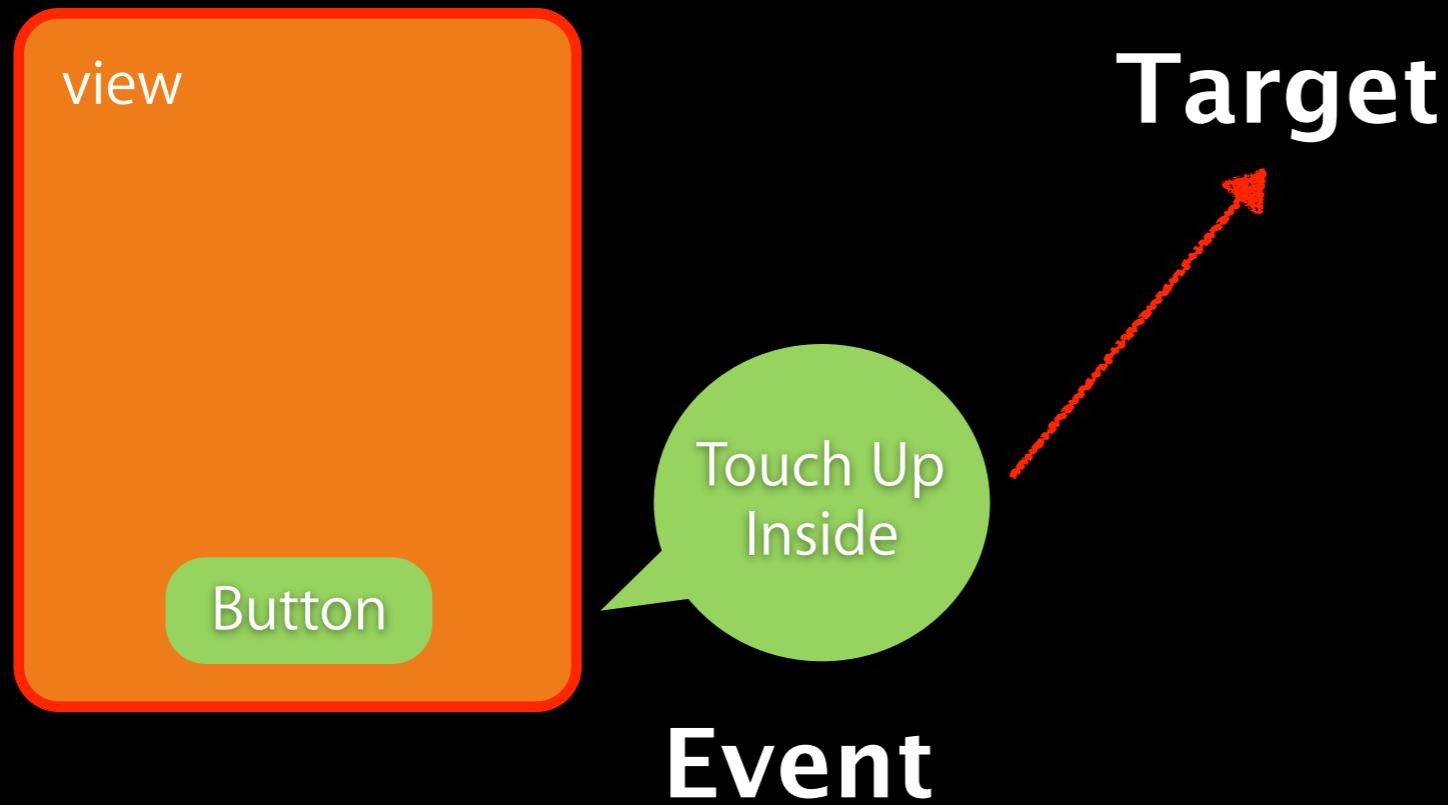
# Target-Action Design Pattern



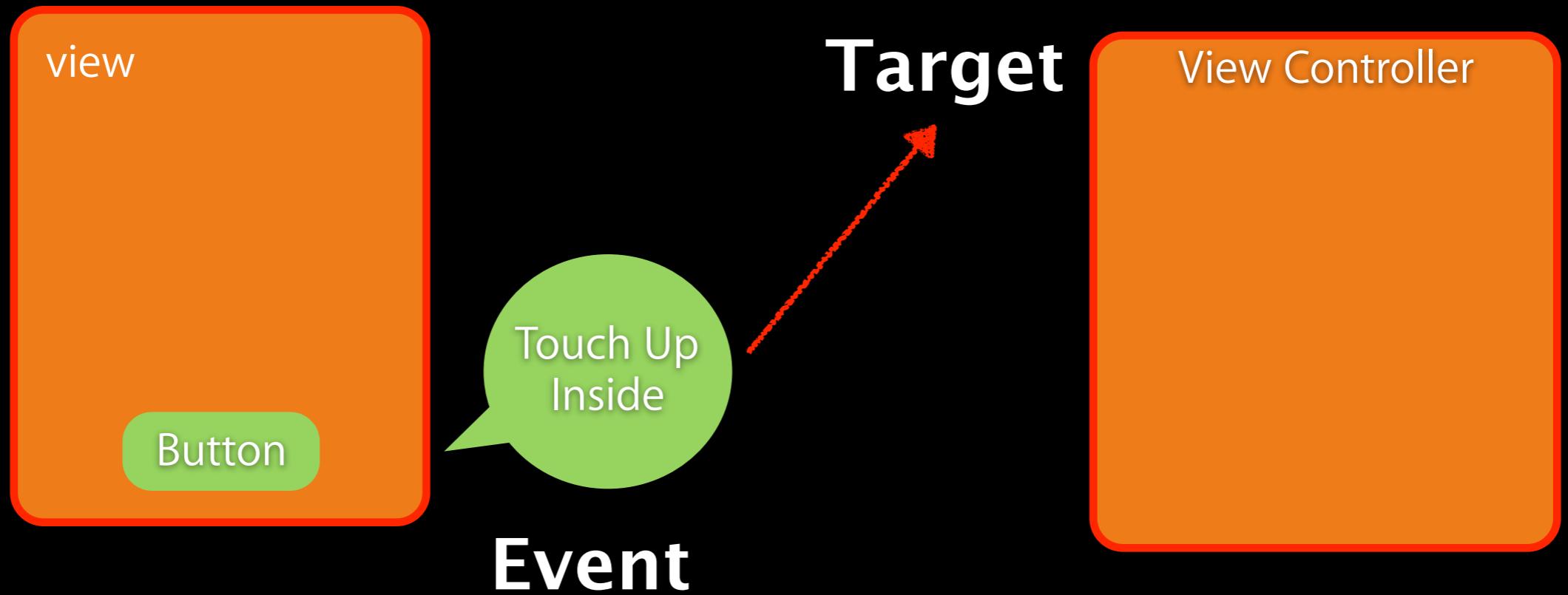
# Target-Action Design Pattern



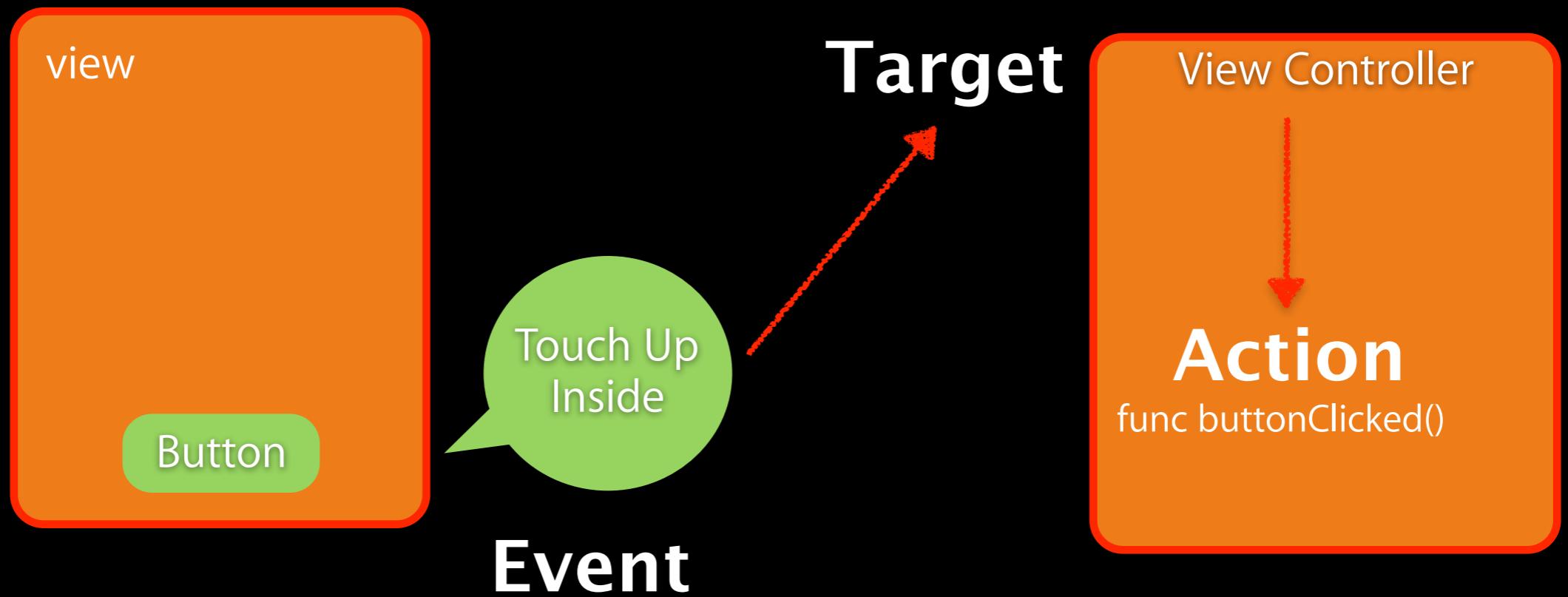
# Target-Action Design Pattern



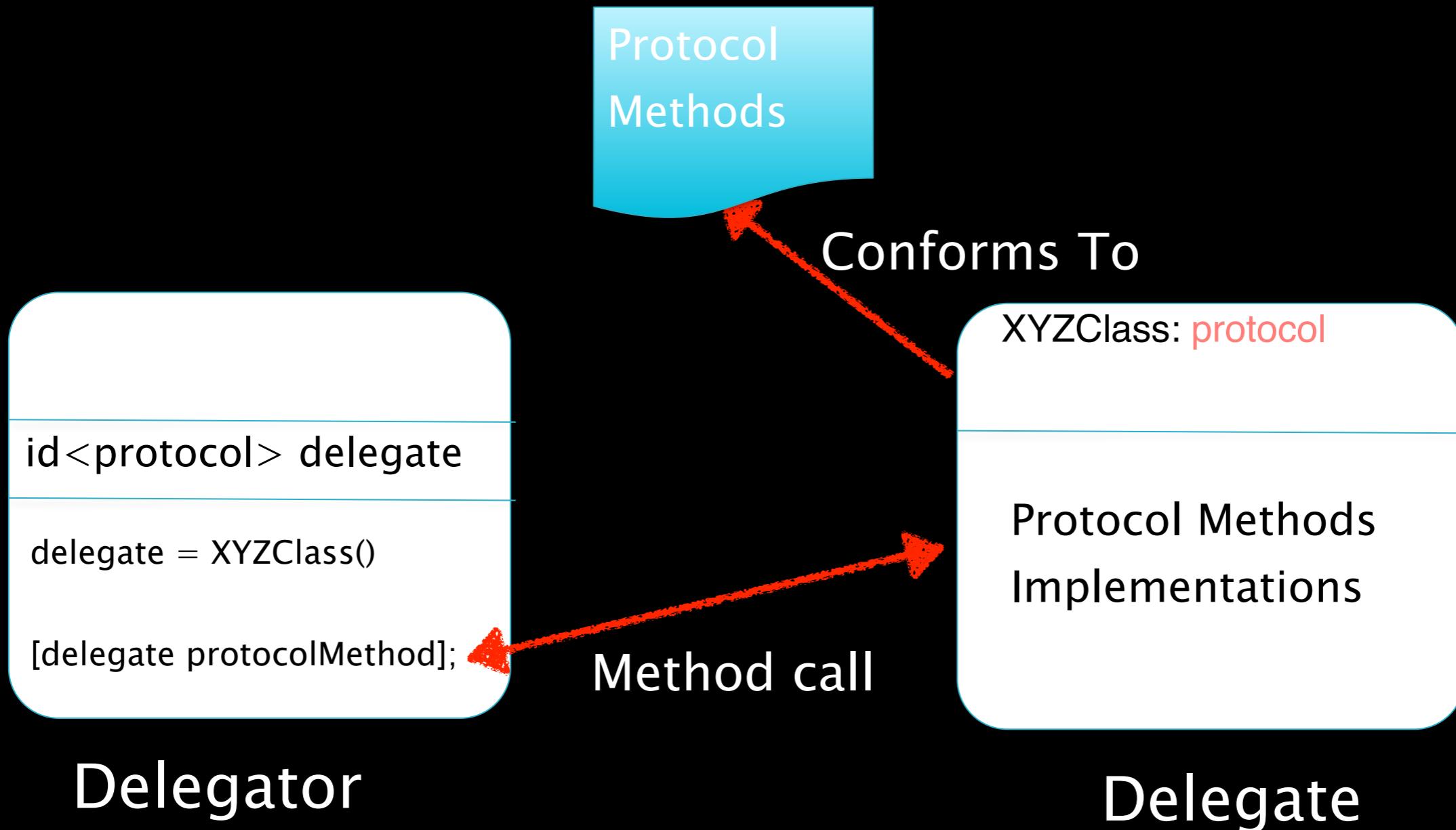
# Target-Action Design Pattern



# Target-Action Design Pattern



# Delegation Design Pattern



# Xcode

# Project



# Project

- Source files



# Project

- Source files
- Targets (e.g., executables)



# Project

- Source files
- Targets (e.g., executables)
- Schemes - how to build targets or perform actions



# Project

- Source files
- Targets (e.g., executables)
- Schemes - how to build targets or perform actions
- Build configurations - build setting variants



# The Project Setting

The screenshot shows the 'Info' tab of the Xcode Project Settings. The left sidebar lists 'PROJECT' (HelloWorld) and 'TARGETS' (HelloWorld). The main area displays the following configuration:

- Deployment Target:** iOS Deployment Target is set to 10.2.
- Configurations:**
  - Name: Based on Configuration File
  - Debug: No Configurations Set
  - Release: No Configurations Set
- Localizations:**
  - Language: English — Development Language
  - Resources: 2 Files Localized
- Internationalization:** A checked checkbox for 'Use Base Internationalization'.

# Build Configurations



# Build Configurations

- Build setting variants



# Build Configurations

- Build setting variants
- Project-level

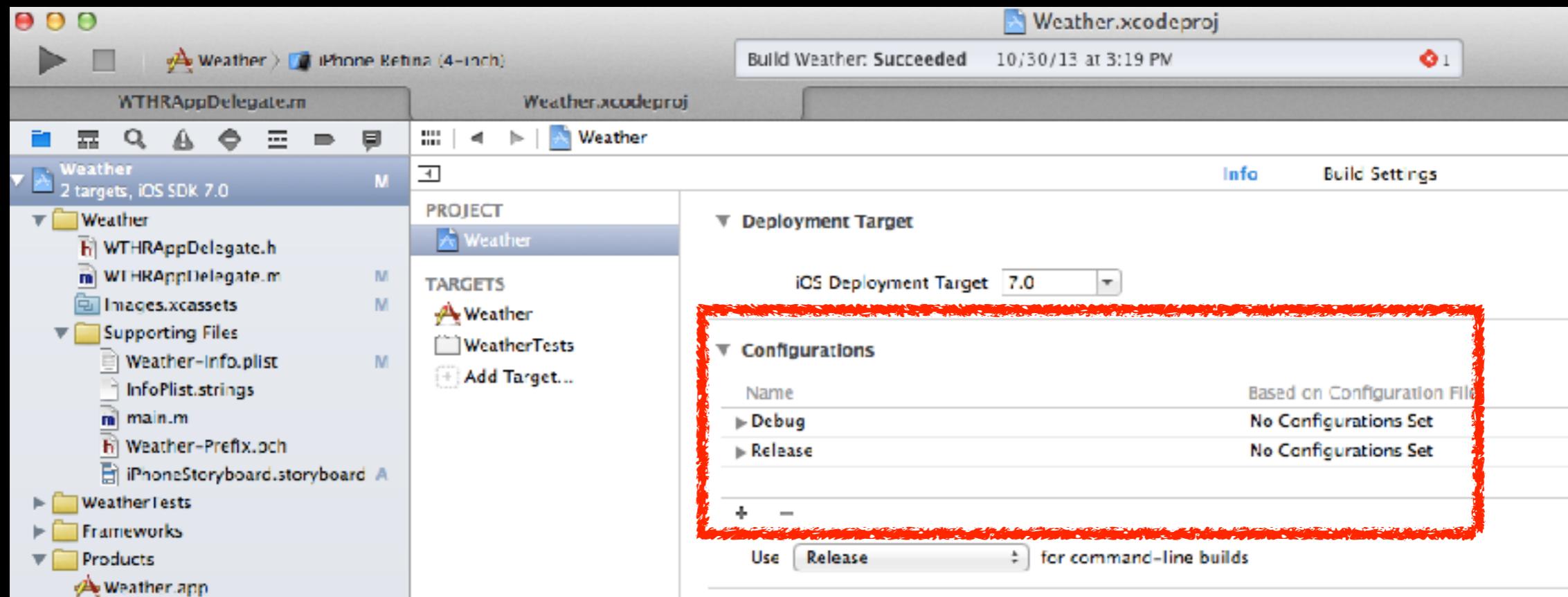


# Build Configurations

- Build setting variants
- Project-level
- Default configurations:  
Debug and Release



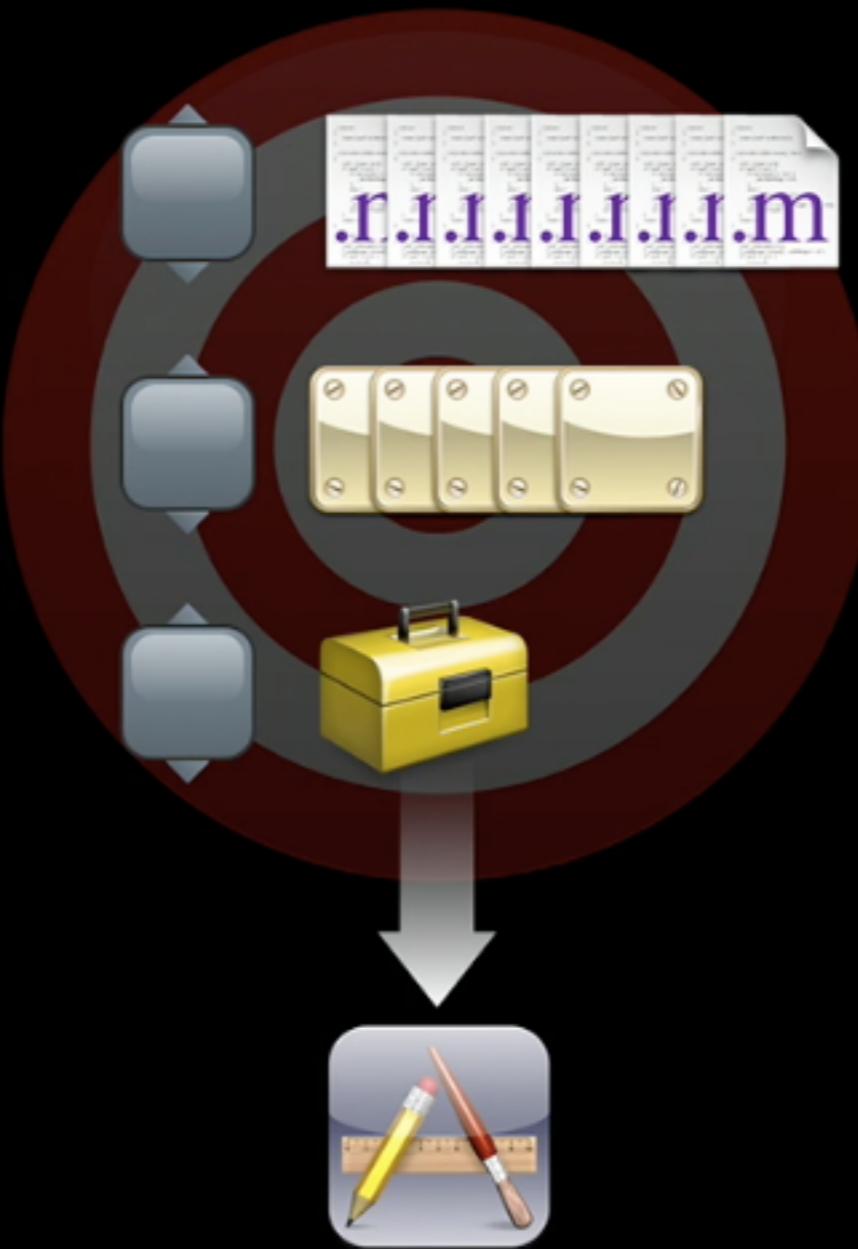
# Build Configurations



# Target – executable

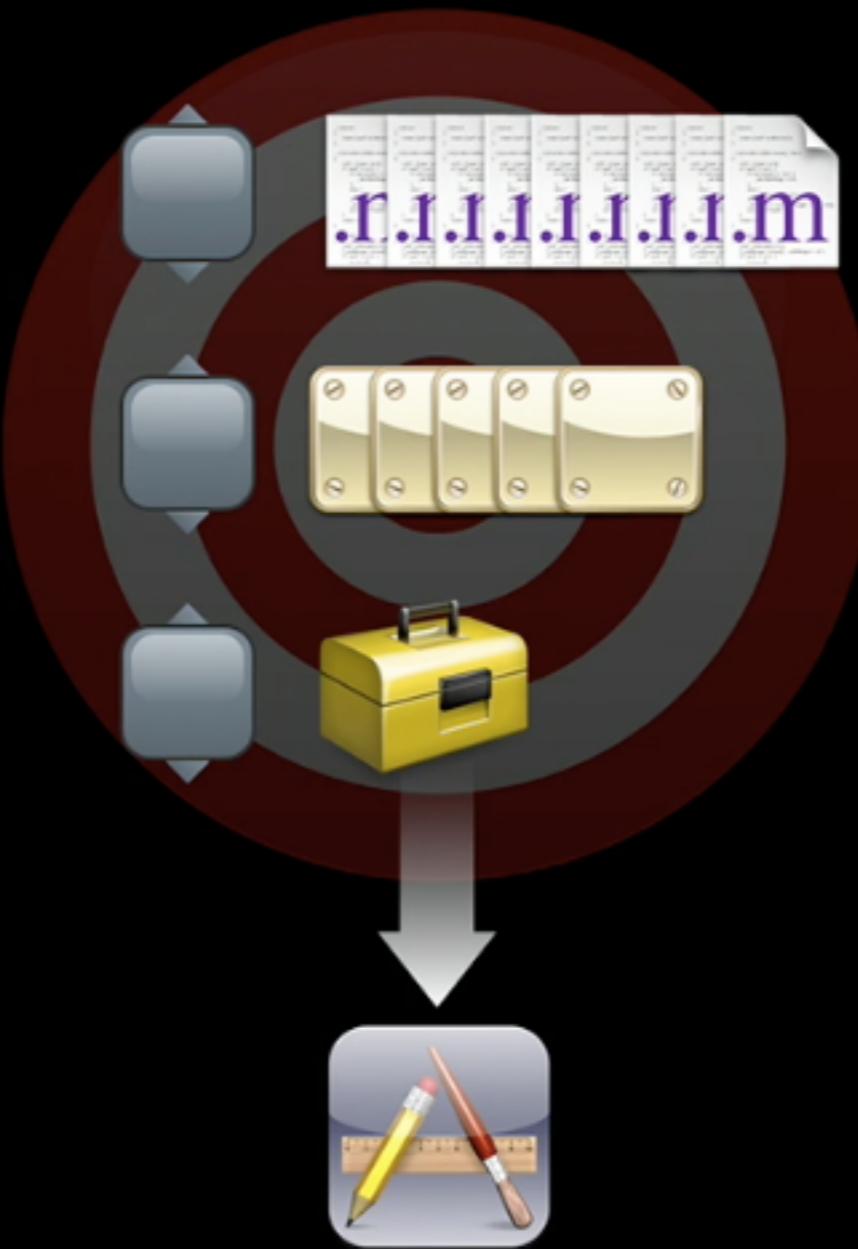
# Target – executable

- Some or all source/resource files



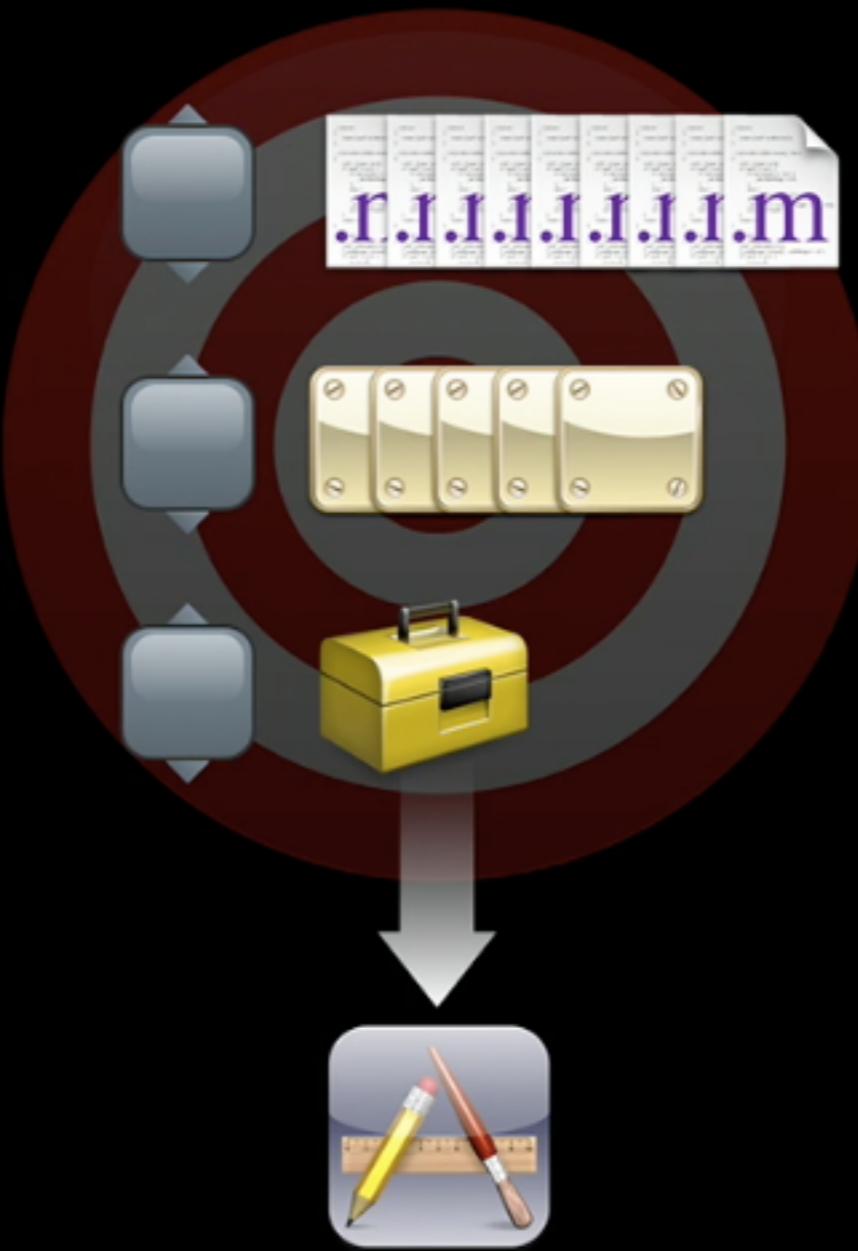
# Target — executable

- Some or all source/resource files
- Build phases - the high-level sequence of steps



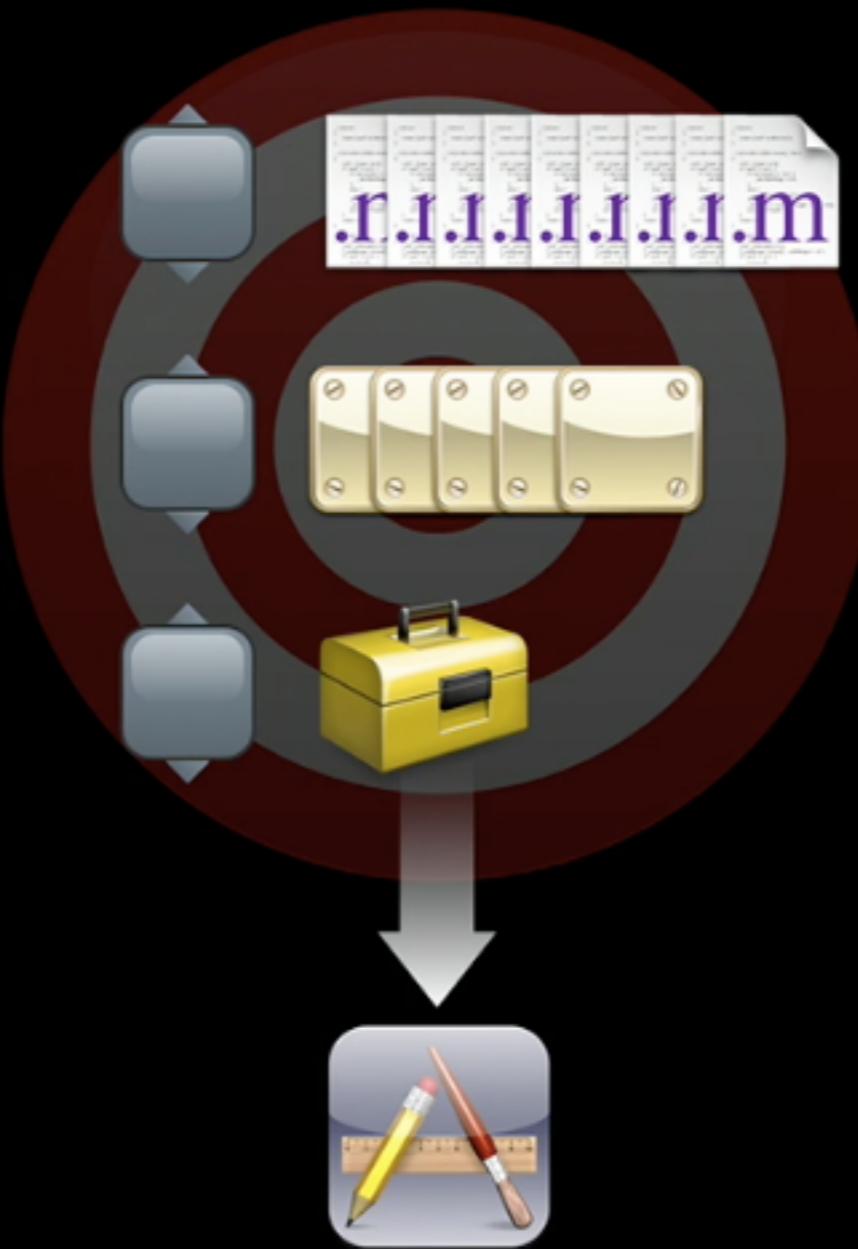
# Target — executable

- Some or all source/resource files
- Build phases - the high-level sequence of steps
- Build settings - how the build is done



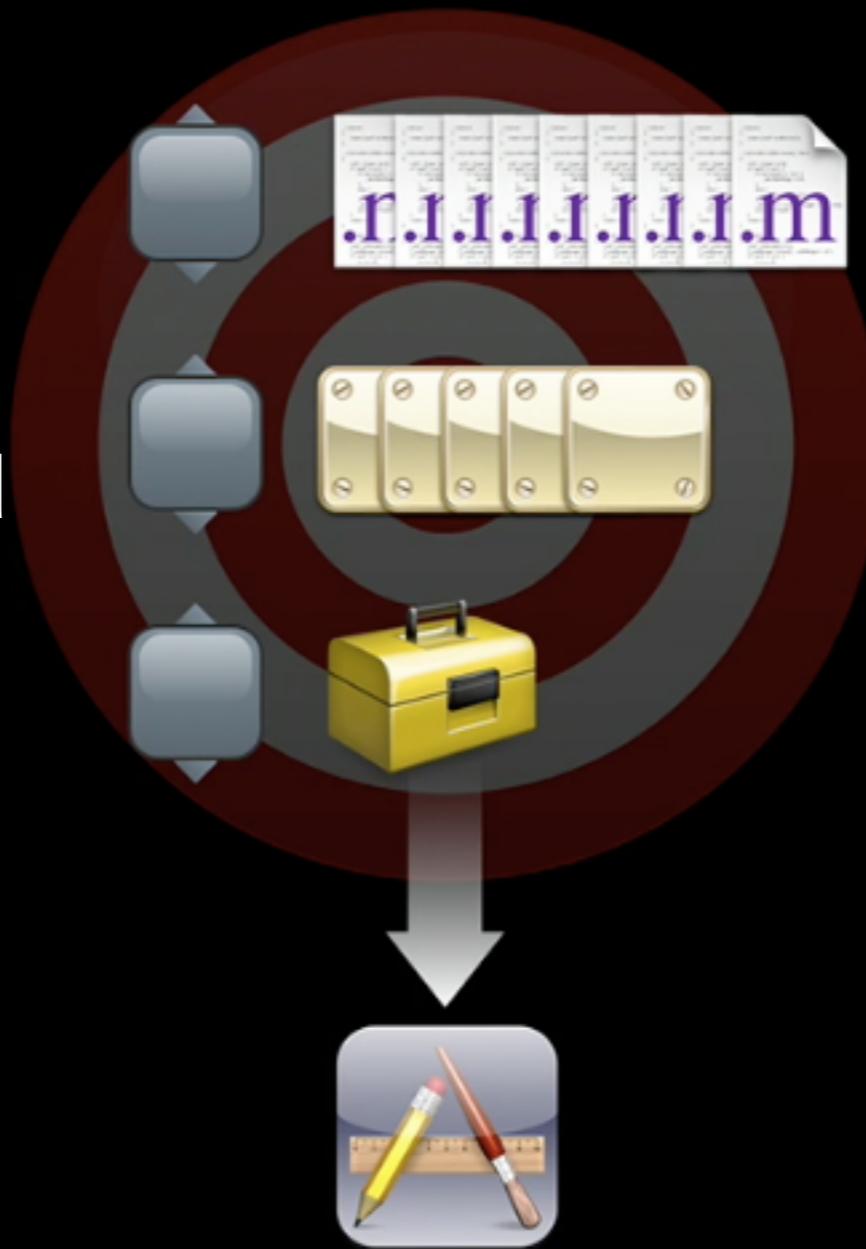
# Target – executable

- Some or all source/resource files
- Build phases - the high-level sequence of steps
- Build settings - how the build is done
- Build rules - how to handle each file/file type

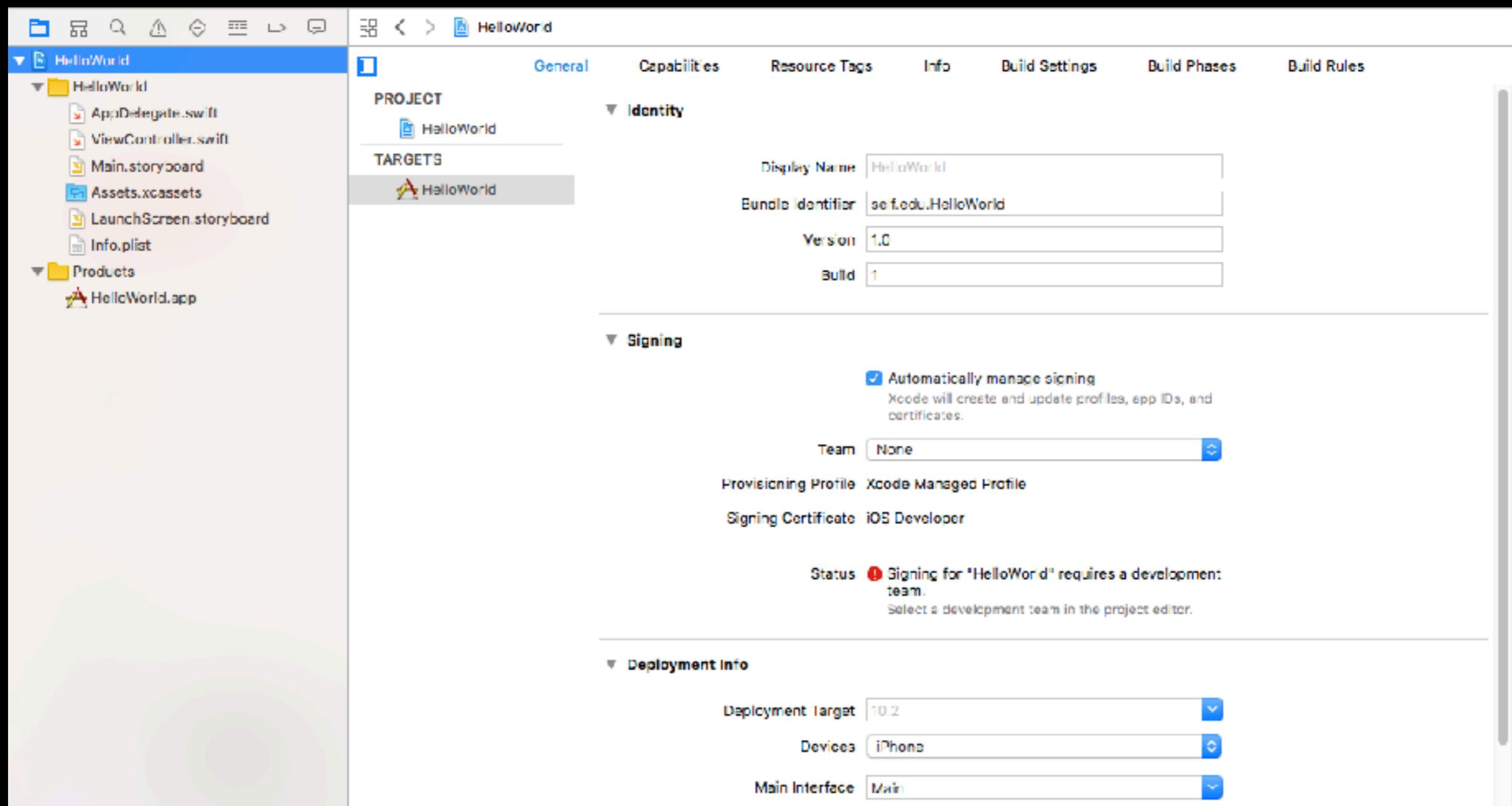


# Target — executable

- Some or all source/resource files
- Build phases - the high-level sequence of steps
- Build settings - how the build is done
- Build rules - how to handle each file/file type
- Can depend on one or more other targets - automatically built



# The Target Setting



# Build Settings Window

The screenshot shows the Xcode Build Settings window for a project named "Weather". The window is organized into sections: General, Capabilities, Info, Build Settings (selected), Build Phases, and Build Rules. The Build Settings section is further divided into sub-sections: Architectures, Build Locations, and Build Options.

**Architectures:**

Setting	Resolved	Weather	iOS Default
Architectures	Standard	Standard	Standard
Base SDK	Latest iOS (iOS 7.0)	Latest iOS (iOS 7.0)	iOS 7.0
Build Active Architecture Only	<Multiple values>	<Multiple values>	No
Debug	Yes	Yes	No
Release	No	No	No
Supported Platforms	iOS	iOS	iOS
Valid Architectures	arm64 armv7 armv7s	arm64 armv7 armv7s	arm64 armv7 armv7s

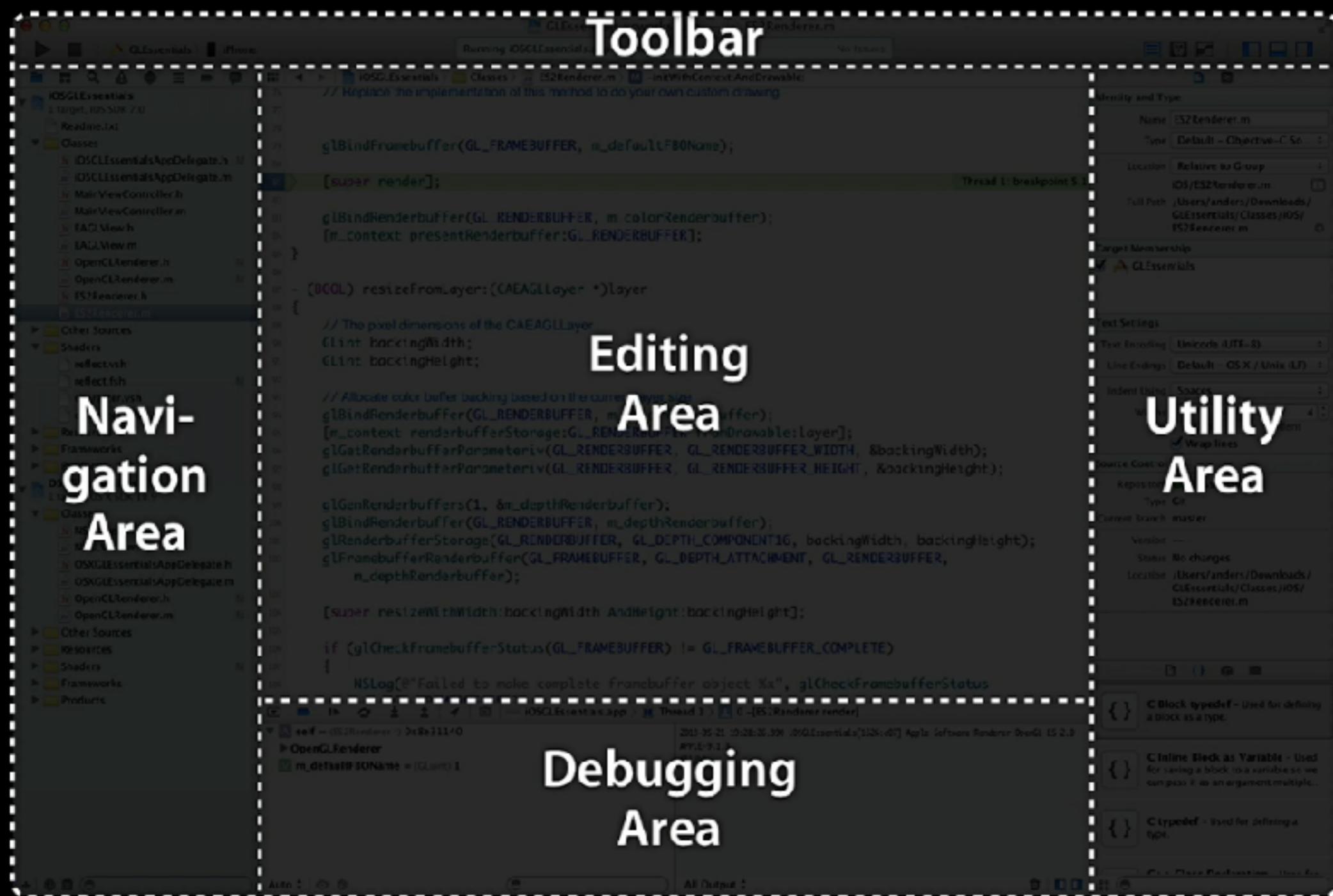
**Build Locations:**

Setting	Resolved	Weather	iOS Default
Build Products Path	build	build	build
Intermediate Build Files Path	build	build	build
Per-configuration Build Products Path	<Multiple values>	<Multiple values>	build
Debug	build/Debug-iphoneos	build/Debug-iphoneos	build
Release	build/Release-iphoneos	build/Release-iphoneos	build
Per-configuration Intermediate Build Files Path	<Multiple values>	<Multiple values>	build/.build
Debug	build/Weather.build/De...	build/Weather.build/De...	build/.build
Release	build/Weather.build/Re...	build/Weather.build/Re...	build/.build
Precompiled Headers Cache Path	/var/folders/x2/3q9z4...	/var/folders/x2/3q9z4...	/var/folders/x2/3q9z4...

**Build Options:**

Setting	Resolved	Weather	iOS Default
Build Variants	normal	normal	normal
Compiler for C/C++/Objective-C	Default compiler (Ap...)	DWARF with dSYM File	DWARF with dSYM File
Debug Information Format	DWARF with dSYM File	DWARF with dSYM File	DWARF with dSYM File
Generate Profiling Code	No	No	No
Precompiled Header Uses Files From Build Directory	Yes	Yes	Yes
Scan All Source Files for Includes	No	No	No
Validate Built Product	<Multiple values>	<Multiple values>	<Multiple values>
Debug	No	No	No
Release	Yes	Yes	Yes

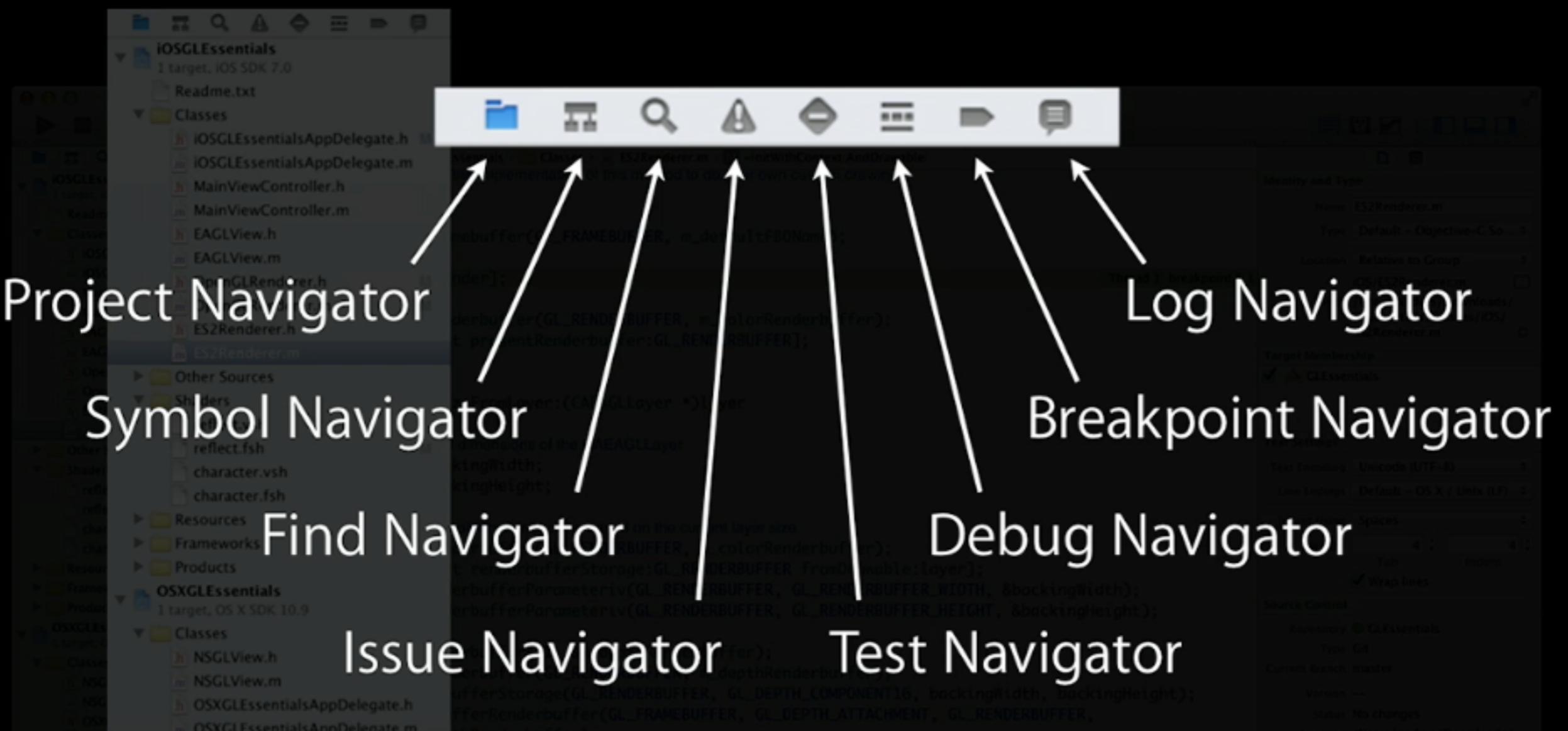
# The Xcode Layout



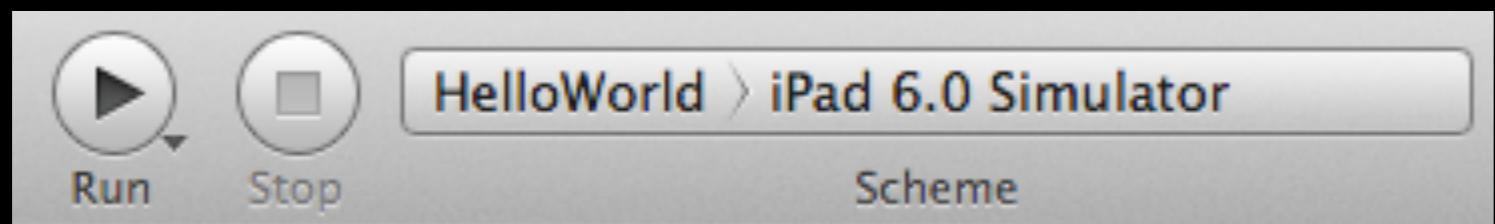
# The “Layout Selector”



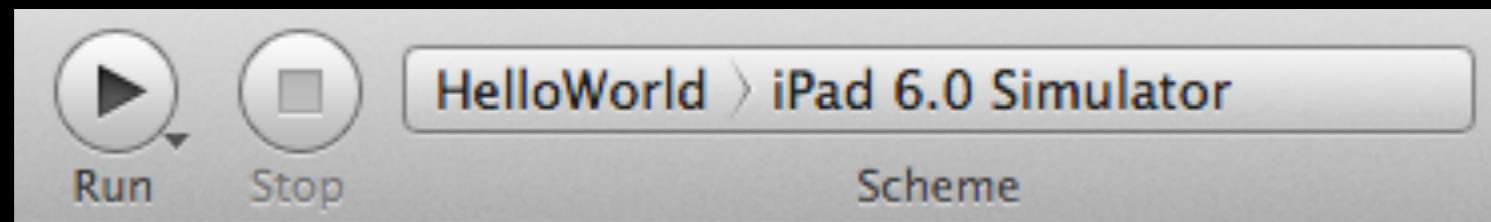
# The Navigation Selector



# Build and Run

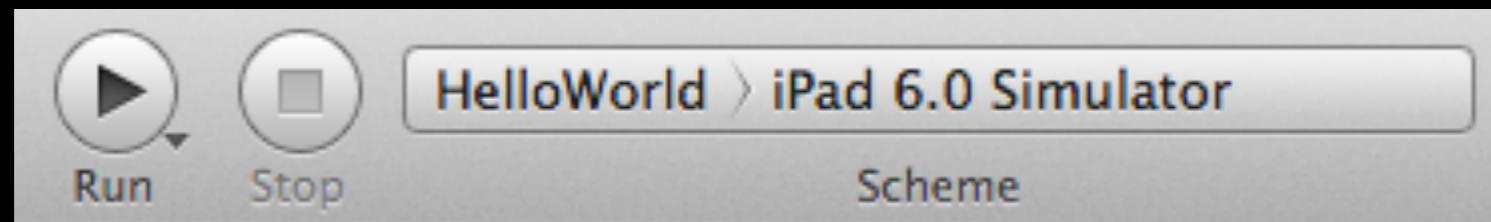


# Build and Run



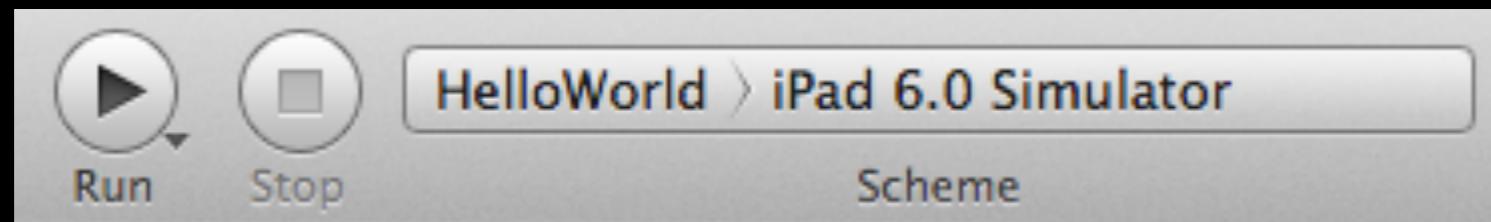
- Build a project - ⌘B

# Build and Run



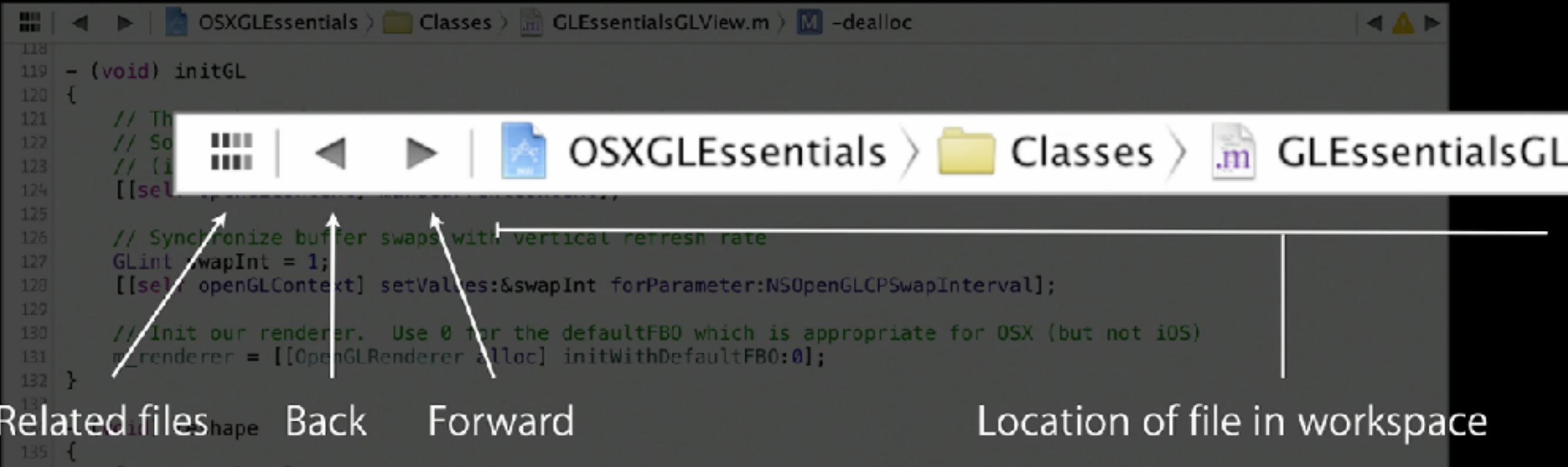
- Build a project - ⌘B
- Run a project - ⌘R

# Build and Run



- Build a project - ⌘B
- Run a project - ⌘R
- Select a testing device

# Editor Jump Bar



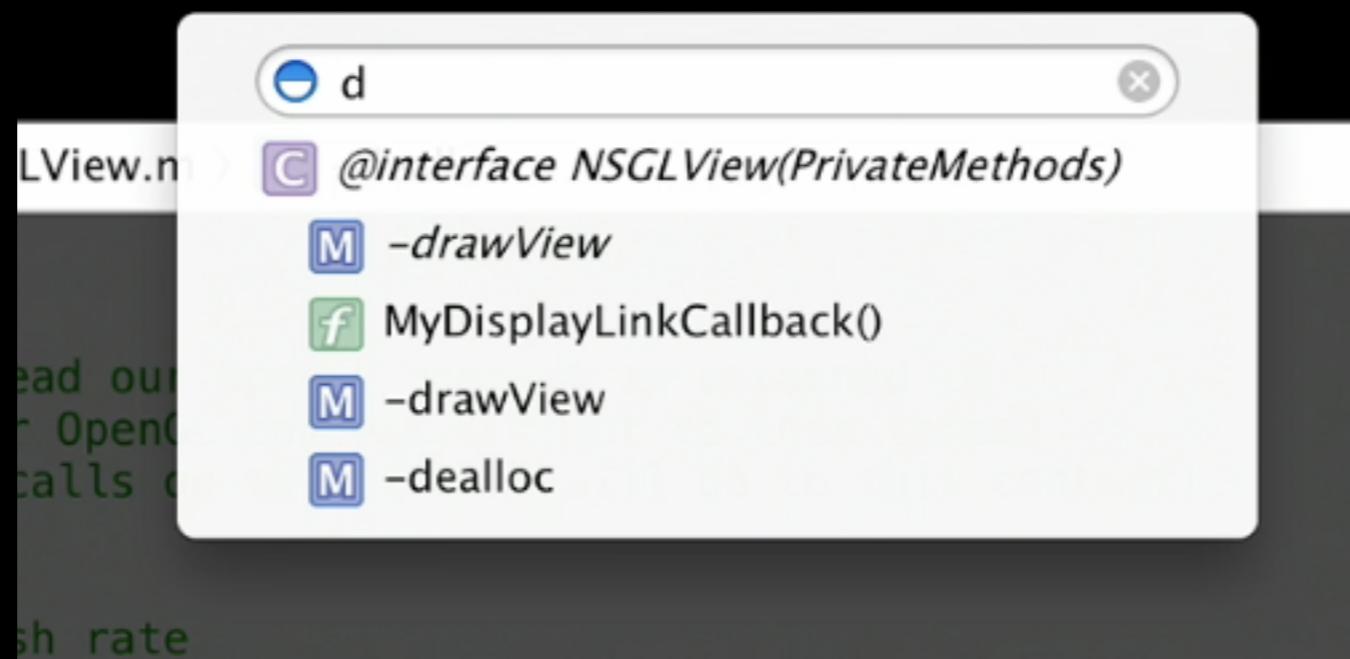
OSXGLEssentials > Classes > GLEssentialsGLView.m -dealloc

```
118
119 - (void) initGL
120 {
121     // Th
122     // So
123     // (i
124     [[sel
125
126     // Synchronize buffer swaps with vertical refresh rate
127     GLint swapInt = 1;
128     [[self openGLContext] setValues:&swapInt forParameter:NSSwapInterval];
129
130     // Init our renderer.  Use 0 for the defaultFBO which is appropriate for OSX (but not iOS)
131     m_renderer = [[OpenGLRenderer alloc] initWithDefaultFBO:0];
132 }
133
134 {
135 }
```

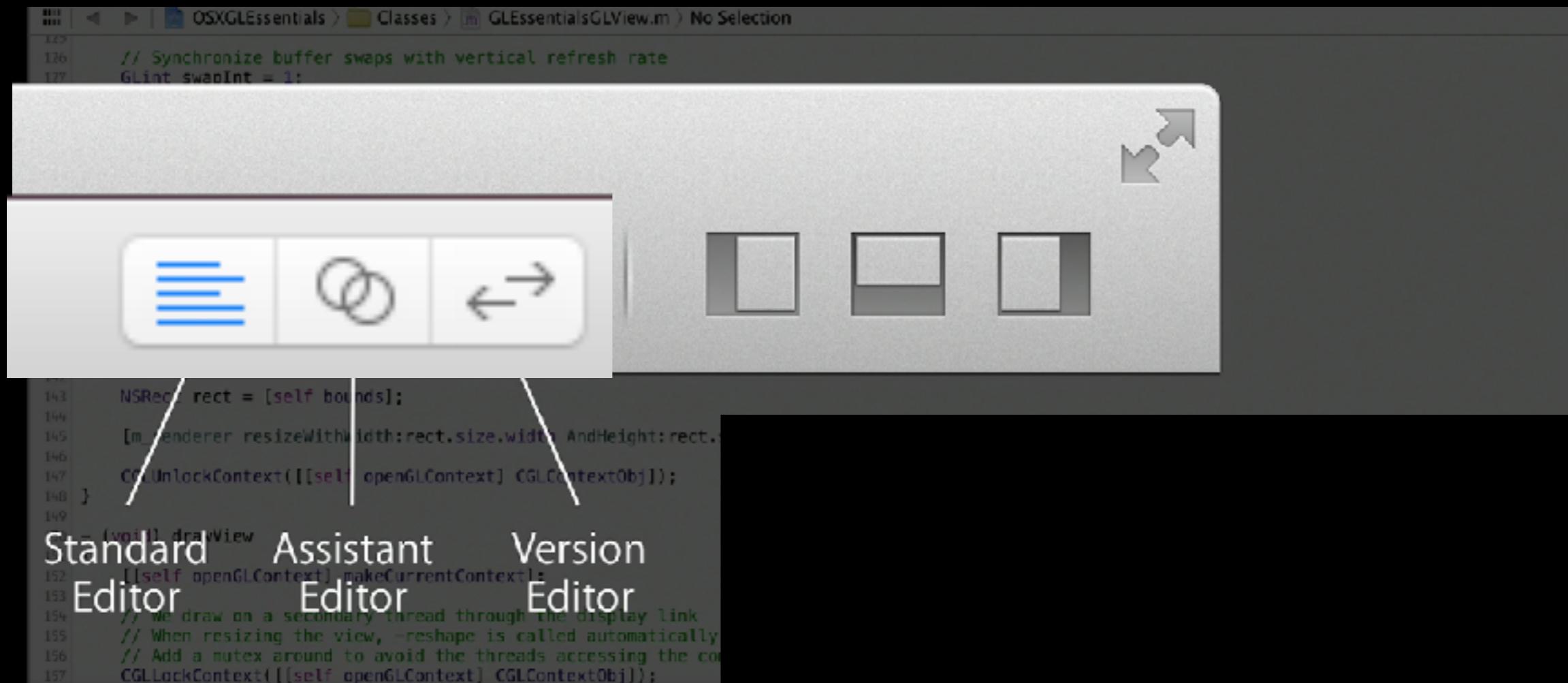
Related files Back Forward

Location of file in workspace

# Editor Jump Bar – Typeahead



# Editor Tool Bar



# Assistant Editor

# Assistant Editor

- Opened automatically by Xcode

# Assistant Editor

- Opened automatically by Xcode
- Option + click — opened manually

# Assistant Editor

- Opened automatically by Xcode
- Option + click — opened manually
  - Test callers

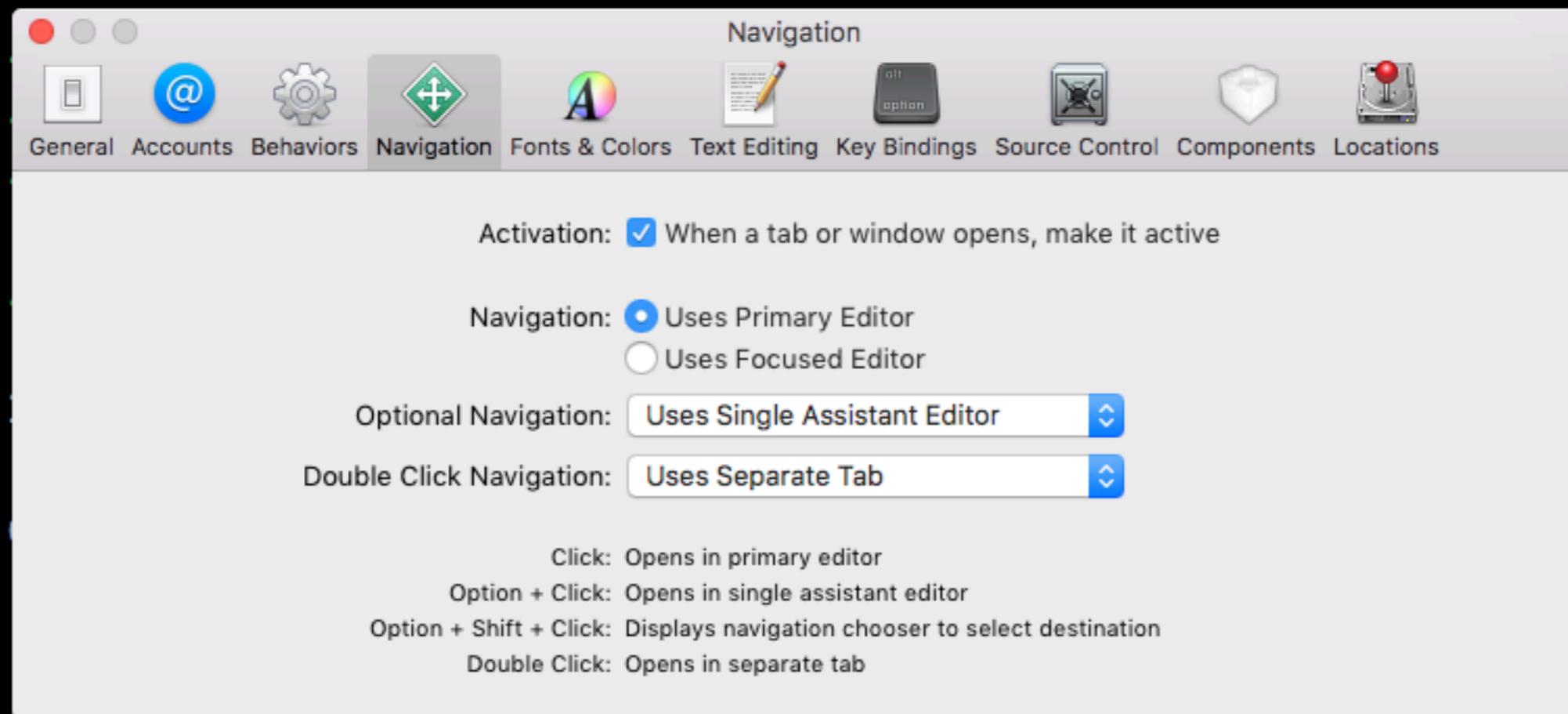
# Assistant Editor

- Opened automatically by Xcode
- Option + click — opened manually
  - Test callers
  - Previews

# Assistant Editor

- Opened automatically by Xcode
- Option + click — opened manually
  - Test callers
  - Previews
- Combined with other operations

# Modify Navigation Behavior



# Open Quickly

cmd

shift

O

File or Symbol

# Xcode Preferences

# Xcode Preferences

- Change fonts

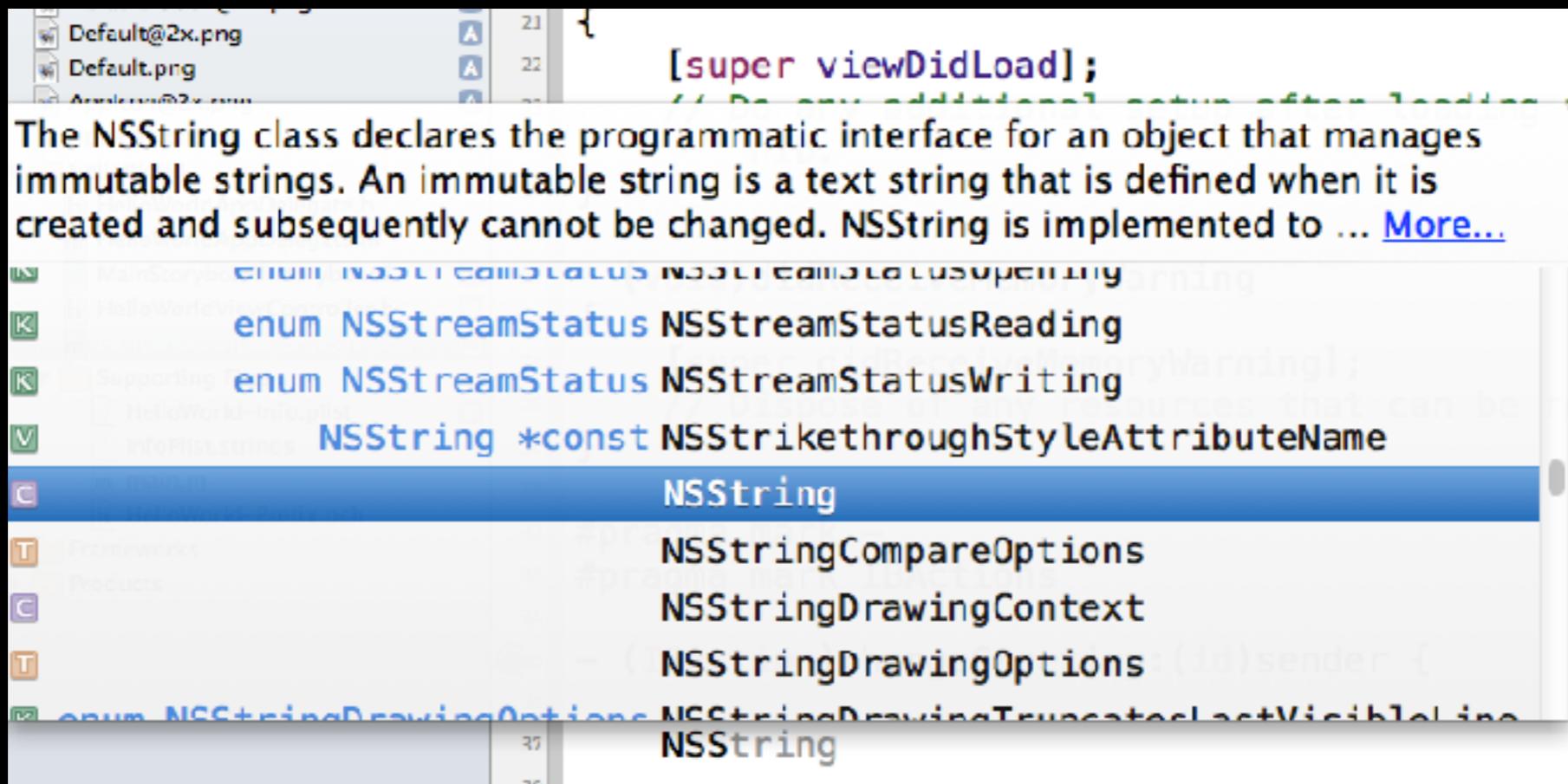
# Xcode Preferences

- Change fonts
- Download docs and tools

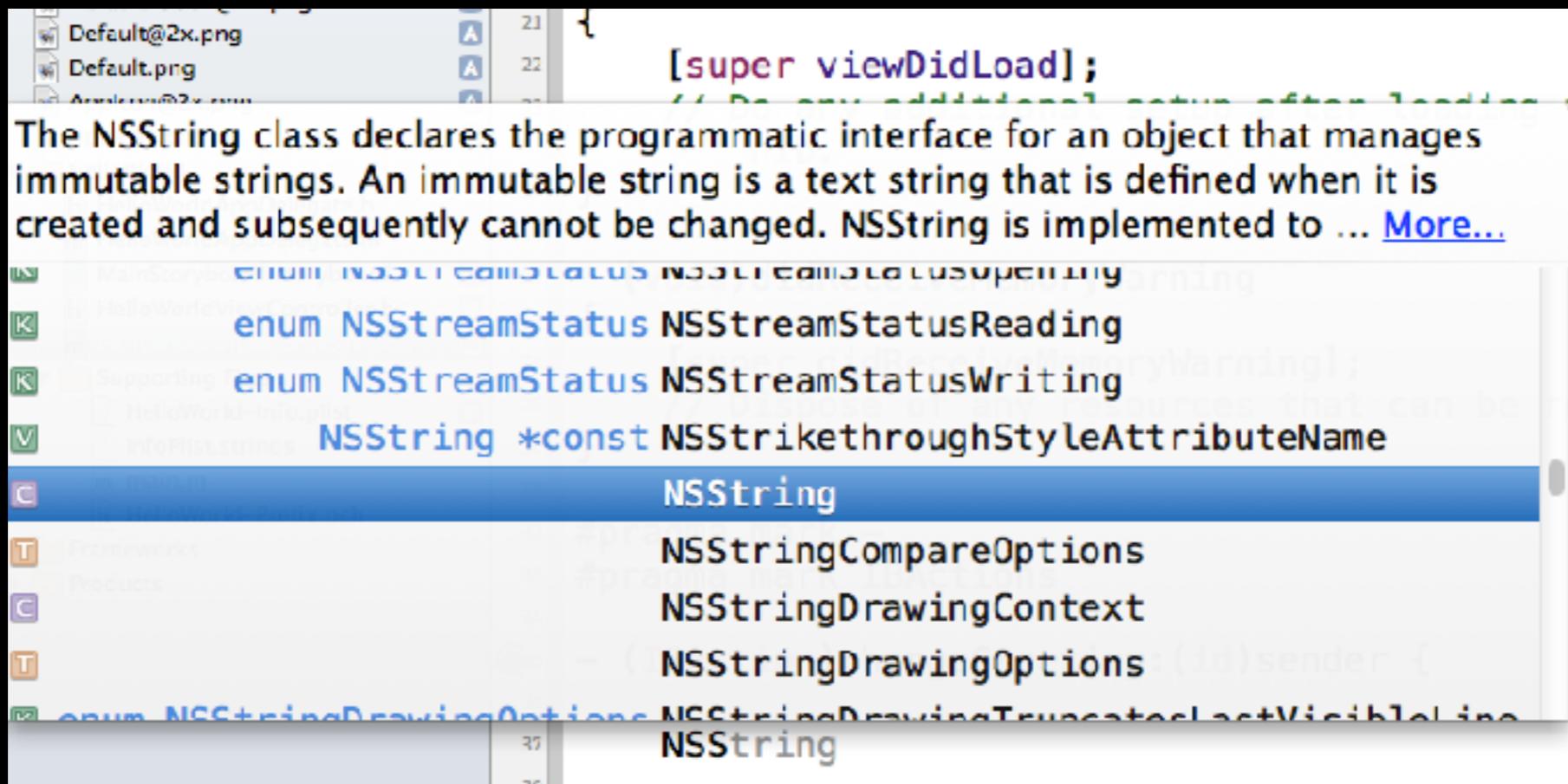
# Xcode Preferences

- Change fonts
- Download docs and tools
- Add line number

# Auto-completion

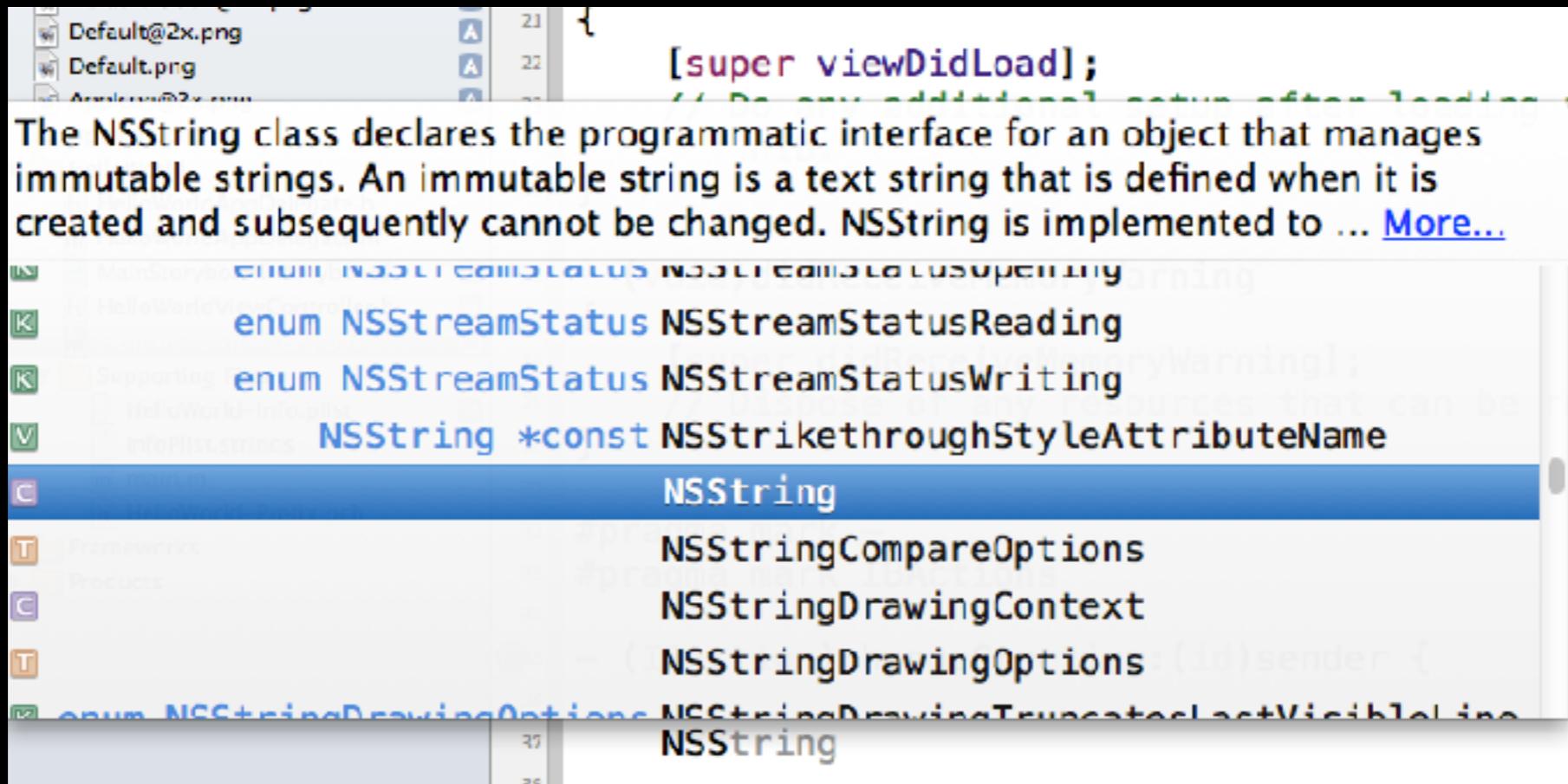


# Auto-completion



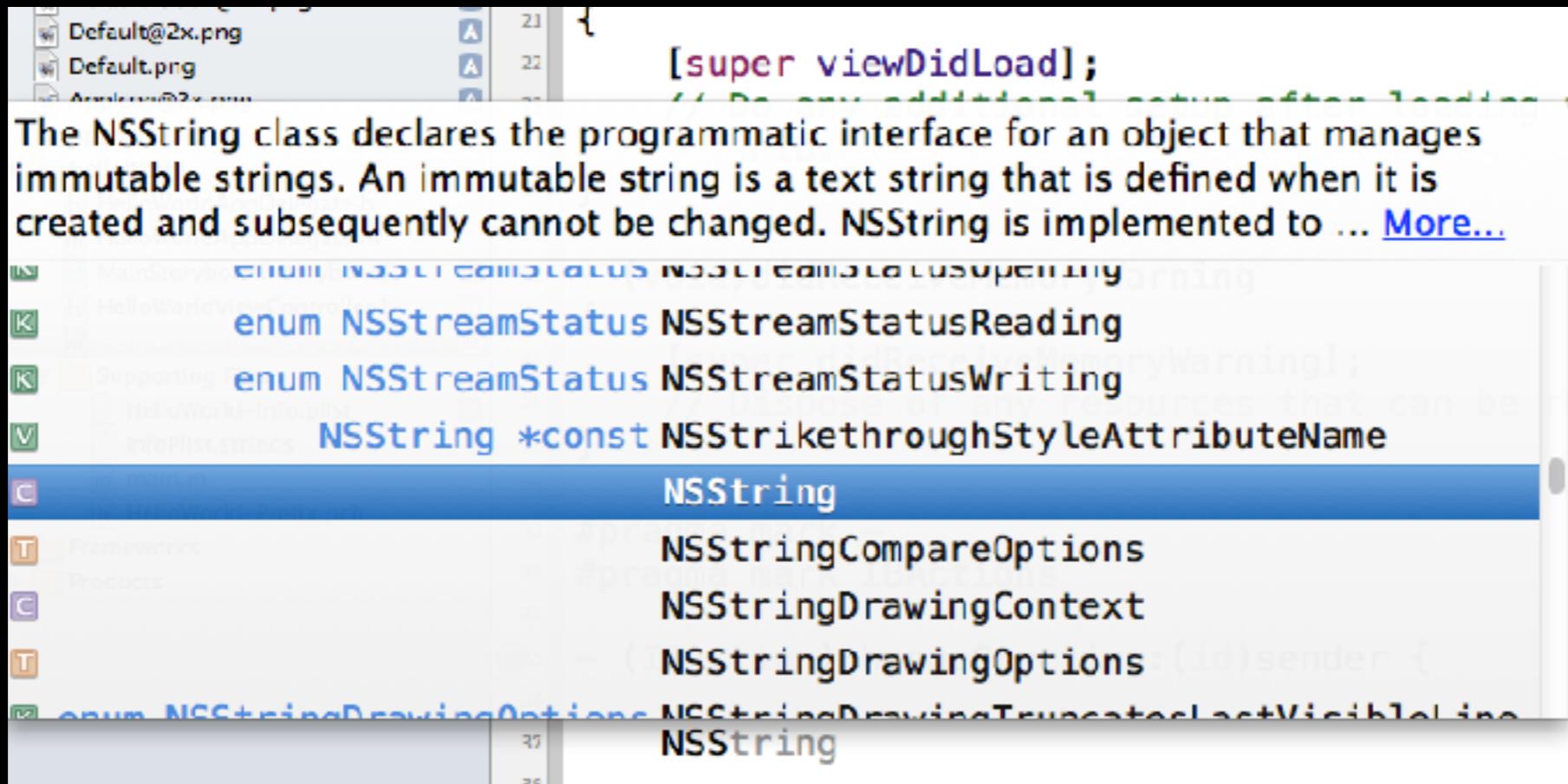
- tab or return to accept auto-completion

# Auto-completion



- tab or return to accept auto-completion
- ESC to cancel the auto-completion

# Auto-completion



- tab or return to accept auto-completion
- ESC to cancel the auto-completion
- ctrl+space to reenter the auto-completion

# Quick Help Inspector

Click any Symbol

The screenshot shows the Xcode interface with the "Quick Help" inspector open for the `NSString` class. The code editor on the left contains Objective-C code, and the quick help panel on the right provides detailed information about the class.

**Quick Help**

**Description:** The `NSString` class declares the programmatic interface for an object that manages immutable strings. An immutable string is a text string that is defined when it is created and subsequently cannot be changed. `NSString` is implemented to represent an array of Unicode characters, in other words, a text string.

**Availability:** iOS (2.0 and later)

**Declared In:** `NSString.h`

**Reference:** [NSString Class Reference](#)

**Guides:** [Property List Programming Guide](#), [String Programming Guide](#)

**Sample Code:** [AdvancedURLConnections](#), [BonjourWeb](#), [CoreTextPageViewer](#), [MVCNetworking](#), [Tableview Fundamentals for iOS](#)

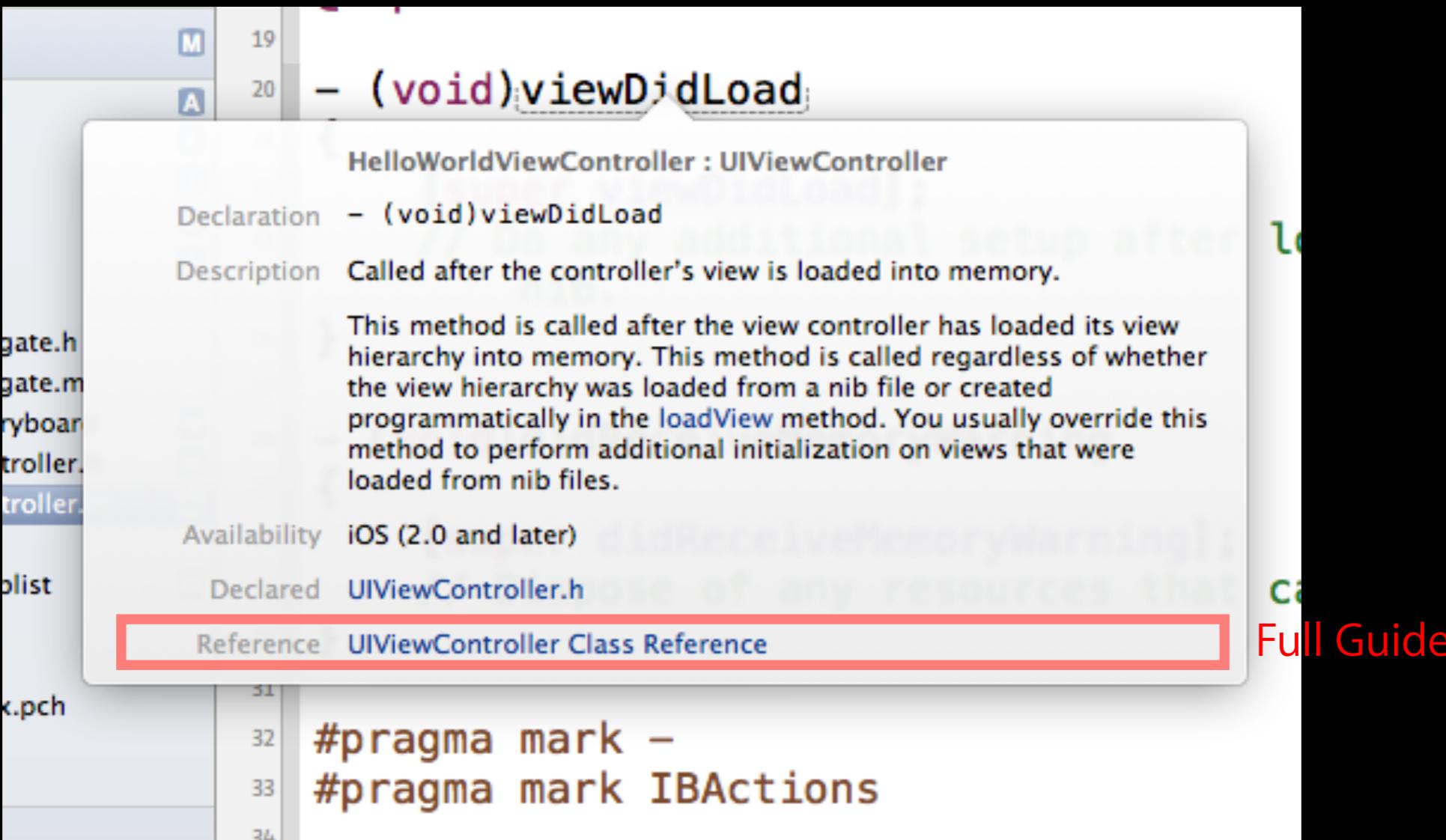
**Related Topics:**

- [Headers](#)
- [API reference](#)
- [Programming guides](#)
- [Sample code](#)

```
866     _mServerDomain = [[NSString
867         alloc] initWithString:
868             SETTINGS_IMPERO_SERVER];
869
870     }
871
872     else {
873         _mServerDomain = [[NSString
874             alloc] initWithString:self.
875             mAppInfo.HotelInfo.Domain];
876
877     }
878
879     return _mServerDomain;
880 }
881
882
883 #pragma mark Context Operations
884
885
886 - (void)restoreObjectContext
887 }
```

# In-Place Quick Help

Option + click any Symbol



Command + click : go to method definition

# Documentation Window

The screenshot shows the Xcode Documentation Window with the title bar "Search documentation". On the left, there's a sidebar with tabs for "Swift", "Objective-C", and "JavaScript". The main content area contains a paragraph of text followed by two code snippets.

In this example you start by constructing an `NSFetchRequest` that describes the data you want returned. This example does not add any requirements to that data other than the type of entity being returned. You then call `executeFetchRequest:error:` on the `NSManagedObjectContext` and pass in the request along with a pointer to an error.

**OBJECTIVE-C**

```
1 NSManagedObjectContext *moc = ...;
2 NSFetchedResultsController *request = [NSFetchRequest fetchRequestWithEntityName:@"Employee"];
3
4 NSError *error = nil;
5 NSArray *results = [moc executeFetchRequest:request error:&error];
6 if (!results) {
7     NSLog(@"%@", error.localizedDescription);
8     abort();
9 }
```

**SWIFT**

```
1 let moc = managedObjectContext
2 let employeesFetch = NSFetchedResultsController(entityName: "Employee")
3
4 do {
5     let fetchedEmployees = try moc.executeFetchRequest(employeesFetch) as! [Employee]
6 } catch {
7     fatalError("Failed to fetch employees: \(error)")
8 }
```

The `executeFetchRequest:error:` method has two possible results. It will either return an `NSArray` object with zero or more objects, or it will return `nil`. If `nil` is returned, you have received an error from Core Data and need to respond to it. If the array exists, you receive possible results for the request even though the `NSArray` may be empty. An empty `NSArray` indicates that there were no records found.

Feedback

# Find/Replace Texts in a File



# Find/Replace Texts in a File



- ⌘F - open the “in-file find” bar

# Find/Replace Texts in a File



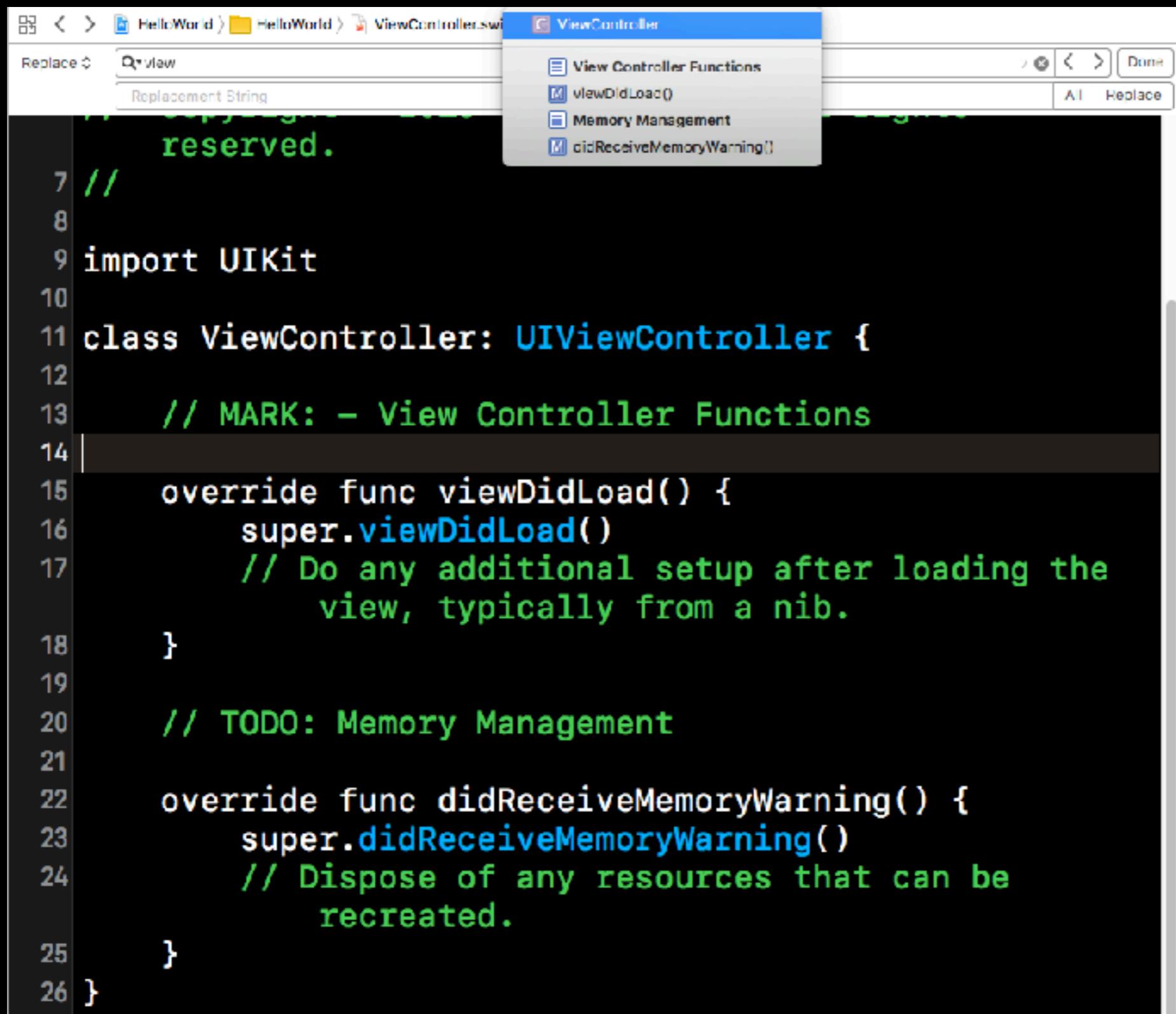
- ⌘F - open the “in-file find” bar
- ⌘G - find next

# Find/Replace Texts in a File



- ⌘F - open the “in-file find” bar
- ⌘G - find next
- “in-file” replace

# Organizing the Function Pop-up



The screenshot shows the Xcode interface with a code editor displaying Swift code for a `ViewController`. A function pop-up menu is open at the top right, listing several functions categorized under `ViewController`:

- View Controller Functions
- `viewDidLoad()`
- Memory Management
- `didReceiveMemoryWarning()`

The code in the editor includes:

```
1 // This class is reserved.
2
3 // MARK: - View Controller Functions
4
5 override func viewDidLoad() {
6     super.viewDidLoad()
7     // Do any additional setup after loading the
8     // view, typically from a nib.
9 }
10
11 // TODO: Memory Management
12
13 override func didReceiveMemoryWarning() {
14     super.didReceiveMemoryWarning()
15     // Dispose of any resources that can be
16     // recreated.
17 }
```

# Debugging in Xcode

# Debugging in Xcode

- Set a breakpoint

# Debugging in Xcode

- Set a breakpoint
- Remove a breakpoint

# Debugging in Xcode

- Set a breakpoint
- Remove a breakpoint
- Disable a breakpoint

# Debugging in Xcode

- Set a breakpoint
- Remove a breakpoint
- Disable a breakpoint
- Step over/Step into/Step out/Continue

# Debugging in Xcode

- Set a breakpoint
- Remove a breakpoint
- Disable a breakpoint
- Step over/Step into/Step out/Continue
- Browse the variable values

# Debugging in Xcode

- Set a breakpoint
- Remove a breakpoint
- Disable a breakpoint
- Step over/Step into/Step out/Continue
- Browse the variable values
- Browse the stack trace

# Debugging in Xcode

- Set a breakpoint
- Remove a breakpoint
- Disable a breakpoint
- Step over/Step into/Step out/Continue
- Browse the variable values
- Browse the stack trace
- Print out object structures

# Debugging in Xcode

- Set a breakpoint
- Remove a breakpoint
- Disable a breakpoint
- Step over/Step into/Step out/Continue
- Browse the variable values
- Browse the stack trace
- Print out object structures
- Modify object values

# Data Tips

```
_unfilteredImageSize = [size;
```

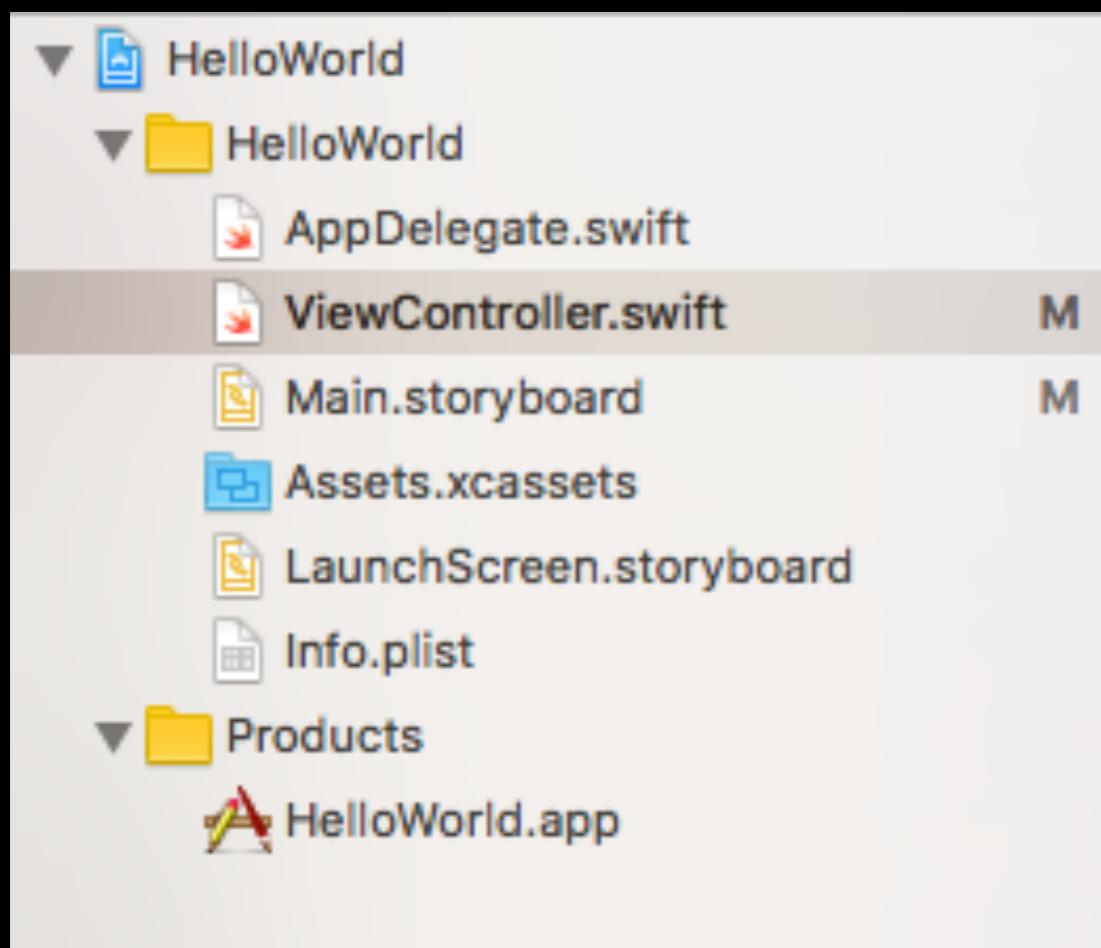
▶ (width=1216, height=811)  

# LLDB

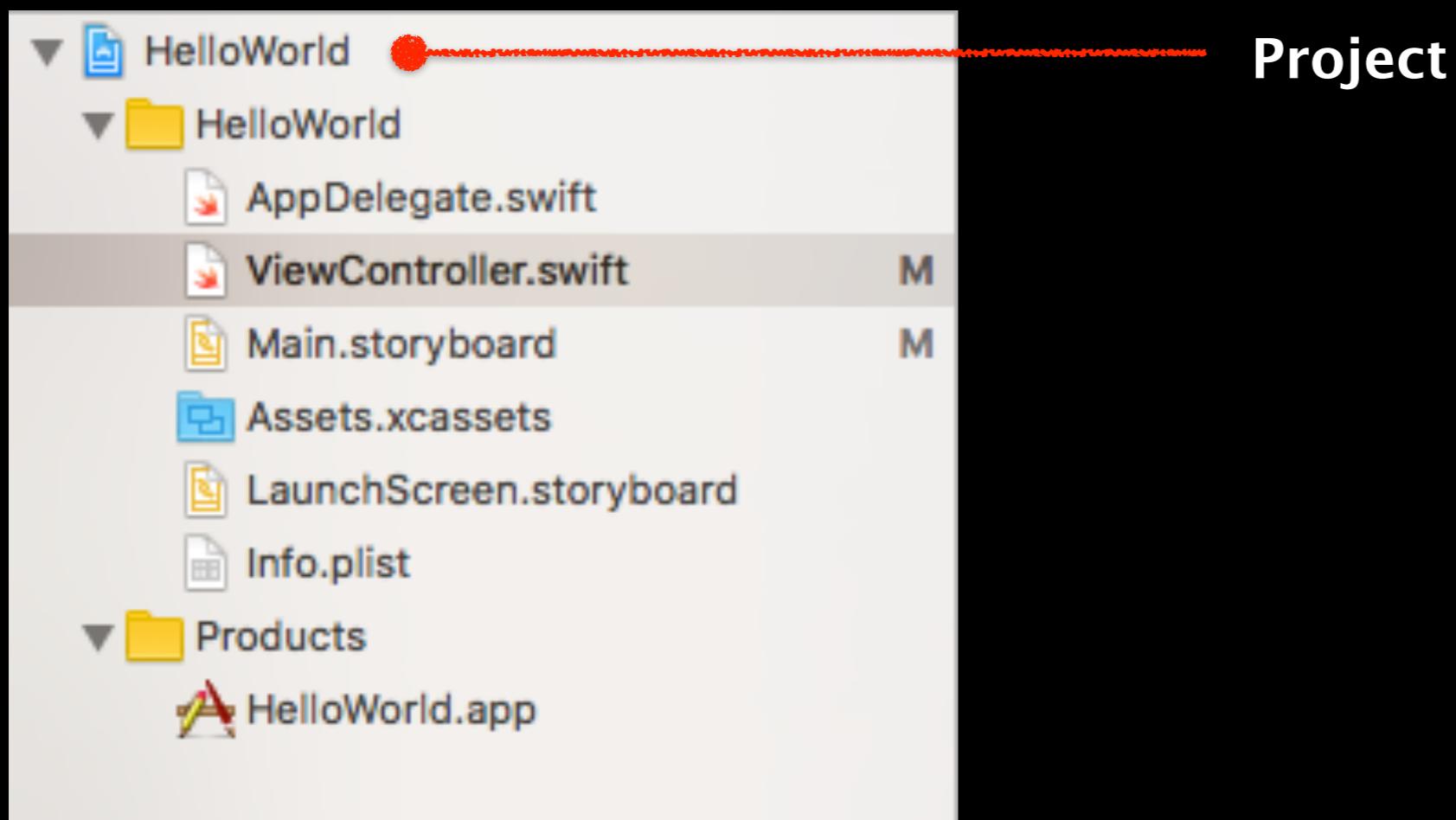
```
(lldb) po self.nameTextField
(UITextField *) $1 = 0x07176100 <UITextField: 0x7176100; frame =
(20 20; 280 30); text = 'Patrick'; clipsToBounds = YES; opaque =
NO; autoresizingMask = TM+BM; gestureRecognizers = <NSArray: 0x717c740>;
layer = <CALayer: 0x71762a0>>
(lldb) po self.userName
(NSString *) $2 = 0x00000000 <nil>
(lldb) po self.userName
(NSString *) $3 = 0x0767ff30 Patrick
(lldb) p self.userName = @""Wang"
(NSString *) $4 = 0x071986d0 @"Wang"
(lldb) po nameString
(NSString *) $5 = 0x071986d0 Wang
```

# iOS Application Overview

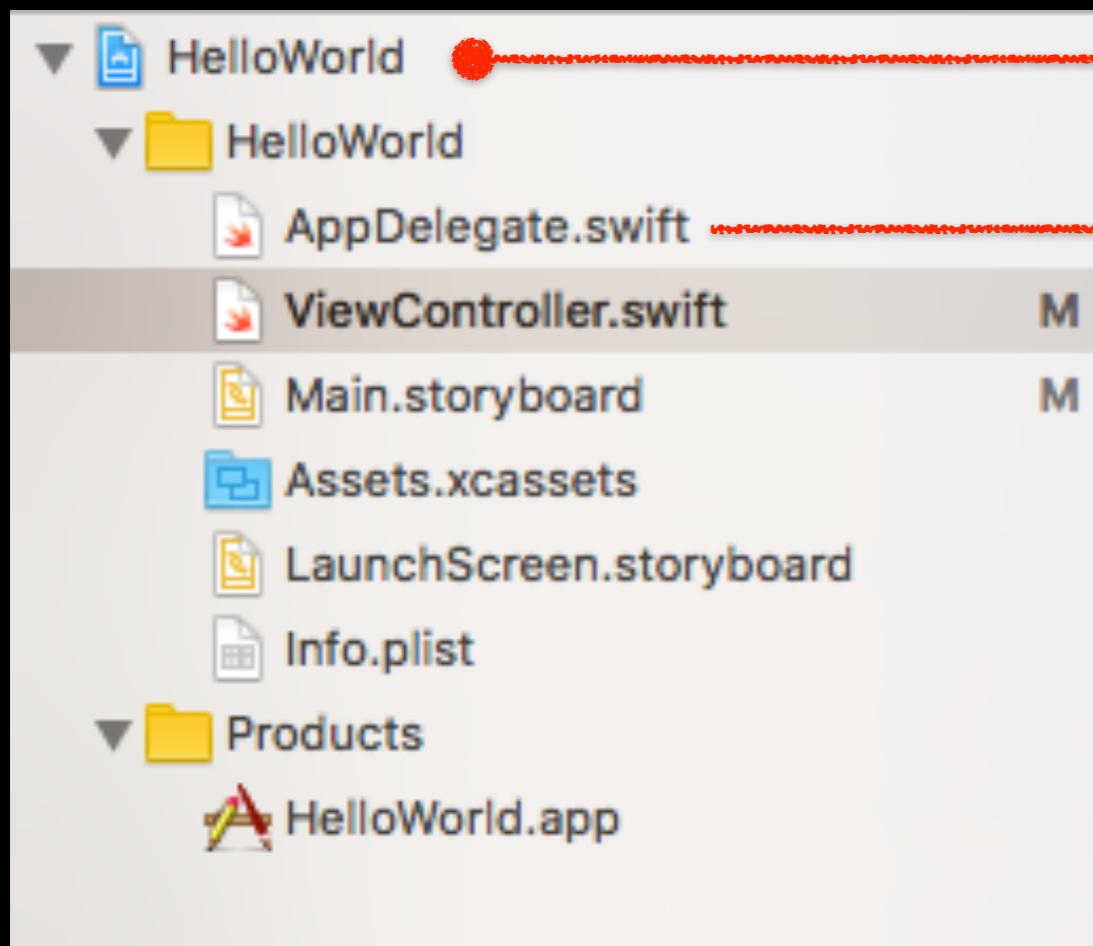
# The Anatomy of an iOS Project



# The Anatomy of an iOS Project



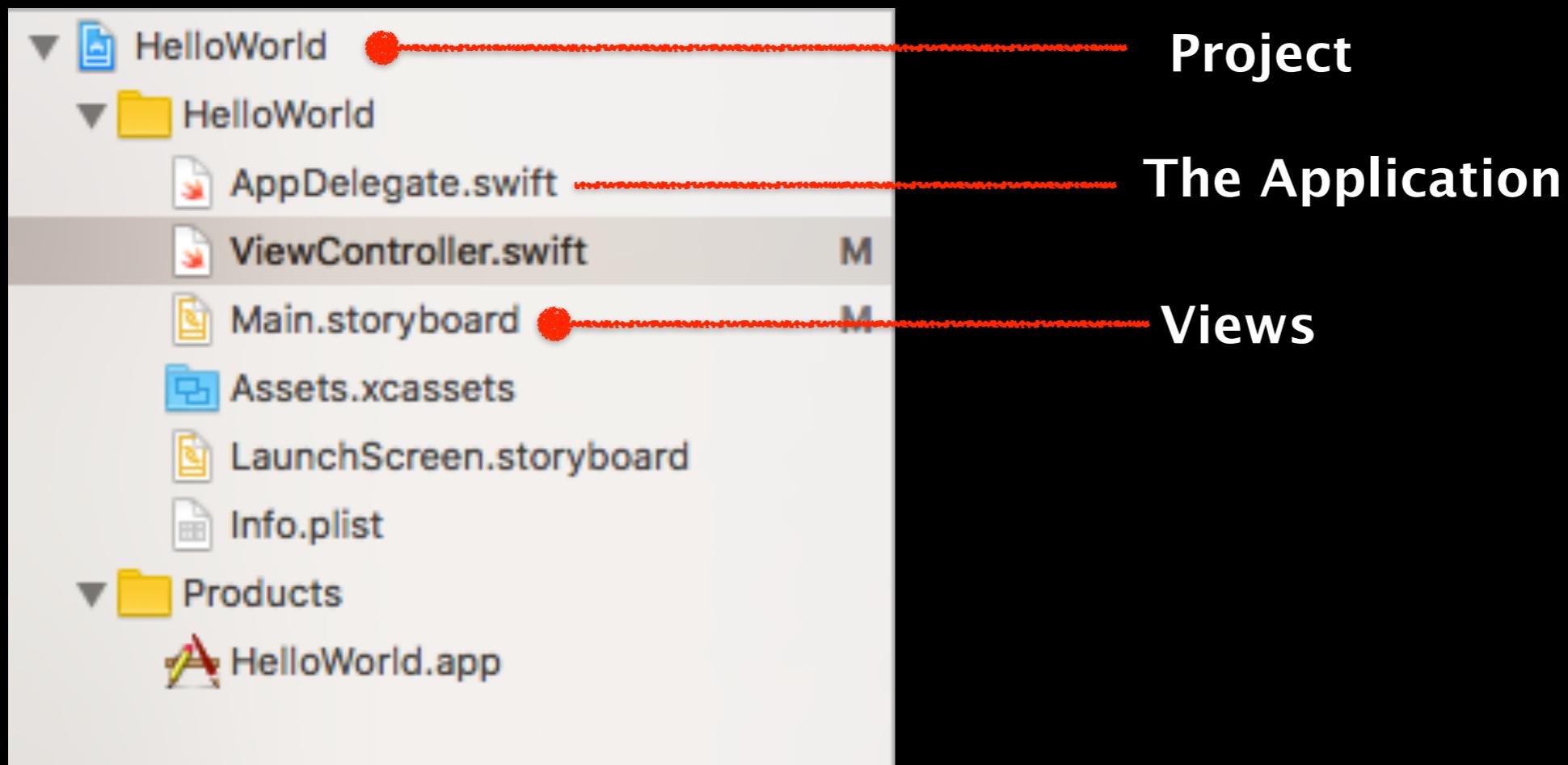
# The Anatomy of an iOS Project



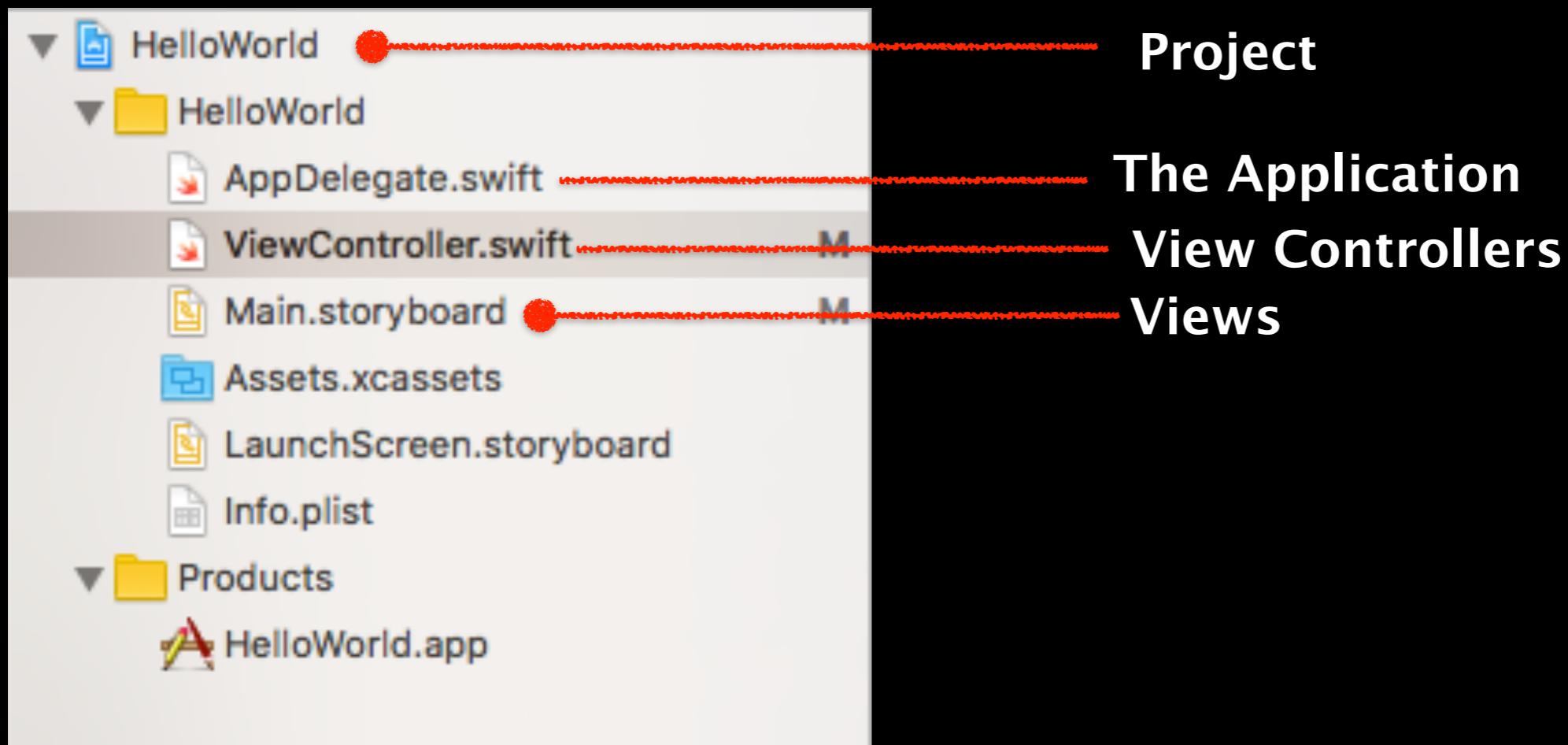
Project

The Application

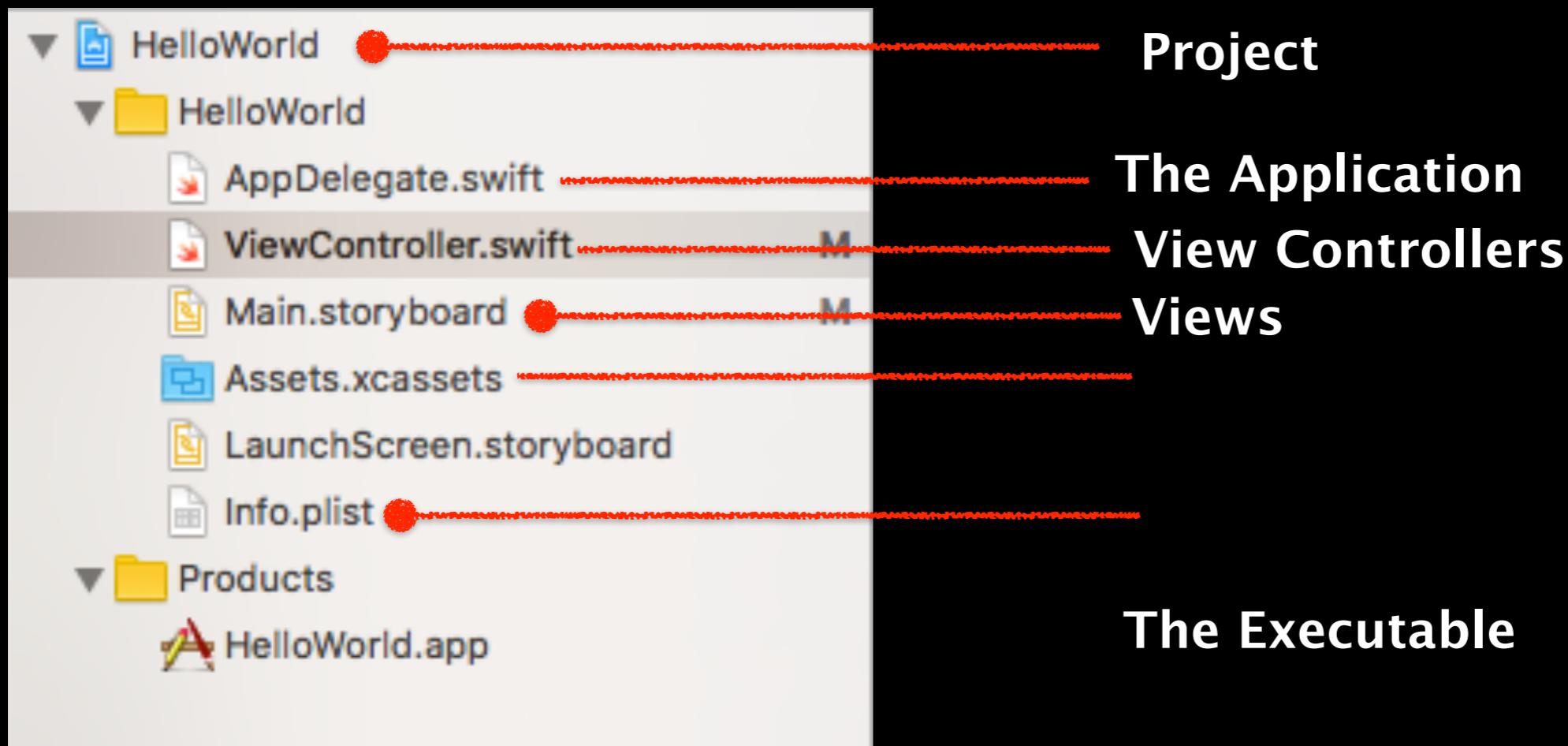
# The Anatomy of an iOS Project



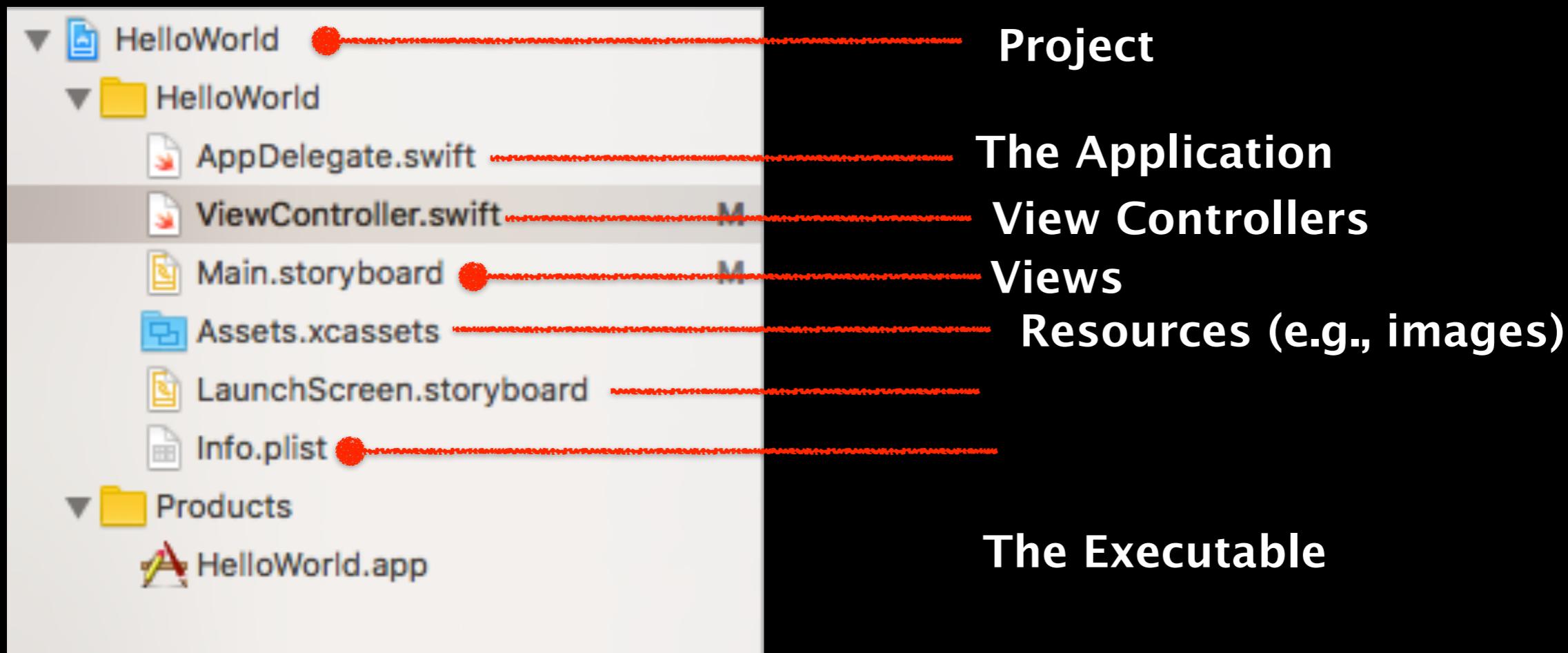
# The Anatomy of an iOS Project



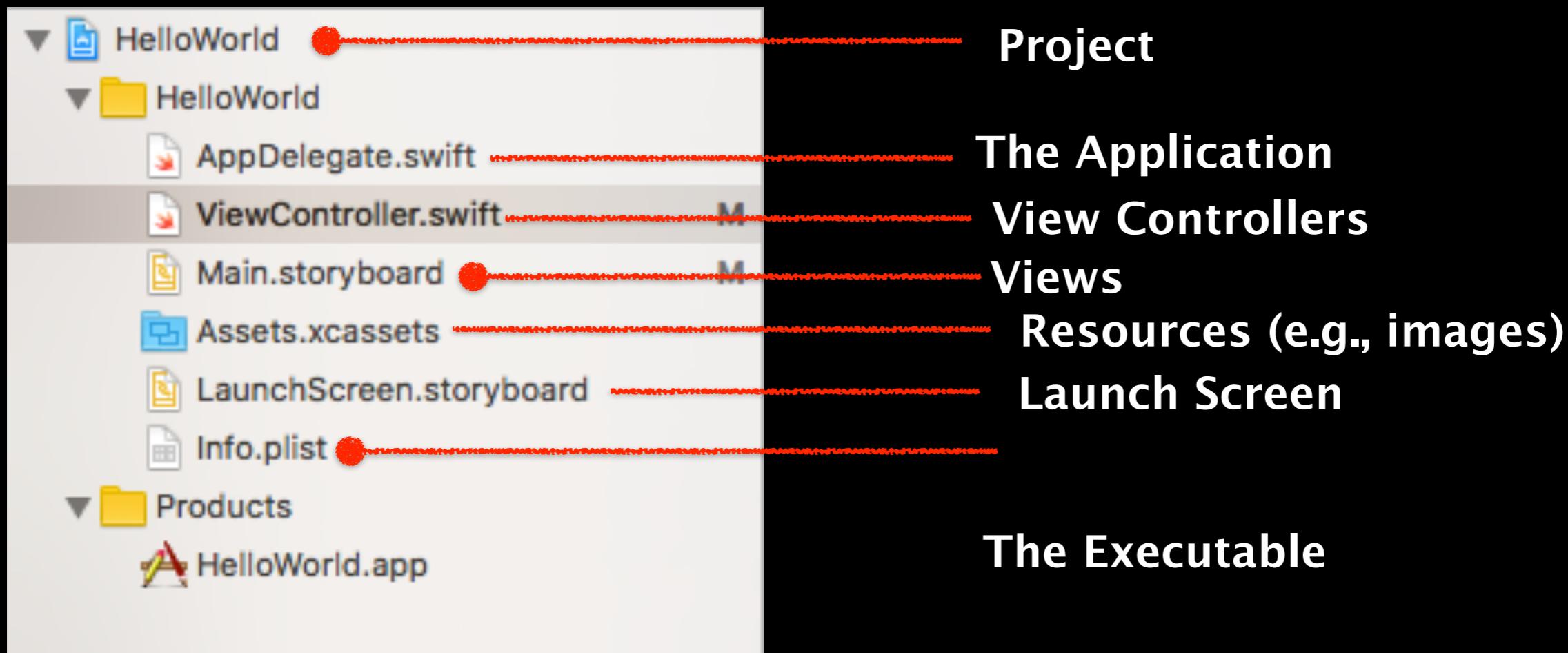
# The Anatomy of an iOS Project



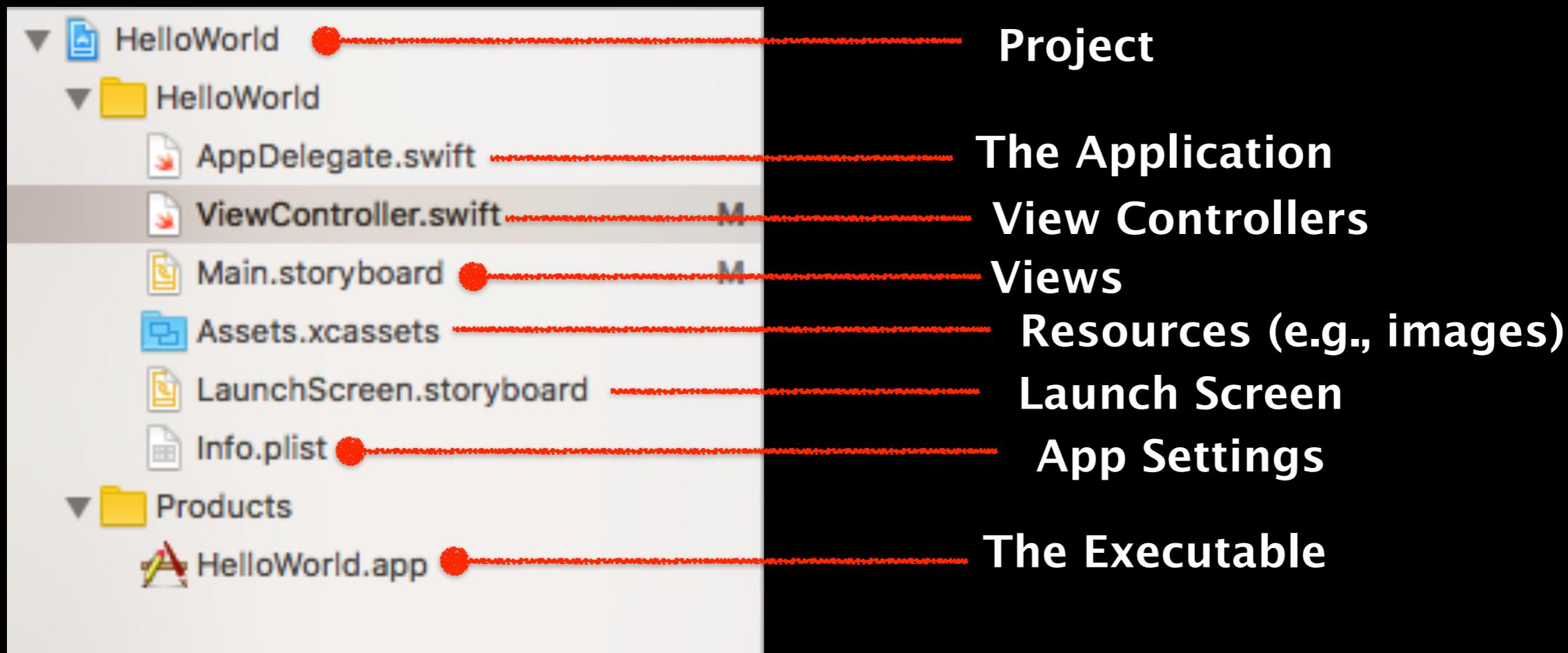
# The Anatomy of an iOS Project



# The Anatomy of an iOS Project



# The Anatomy of an iOS Project



# The Application Entry Point

- **@UIApplicationMain**: a class with this attribute is the application delegate that defines application-specific behavior
- Is named **xxxxAppDelegate**
- Supply a **main.swift** if not using this attributes

```
//  
// AppDelegate.swift  
// HelloWorld  
//  
// Created by Yanping Zhao on 11/20/15.  
// Copyright © 2015 Yanping Zhao. All rights reserved.  
  
import UIKit  
  
@UIApplicationMain  
class AppDelegate: UIResponder, UIApplicationDelegate {  
  
    var window: UIWindow?  
}
```

# The Launch Screen

# The Launch Screen

- iOS displays the launch screen instantly when the user starts an app

# The Launch Screen

- iOS displays the launch screen instantly when the user starts an app

# The Launch Screen

- iOS displays the launch screen instantly when the user starts an app
- When the app is ready to use, iOS replaces the launch image with your app's first screen

# The Launch Screen

- iOS displays the launch screen instantly when the user starts an app
- When the app is ready to use, iOS replaces the launch image with your app's first screen

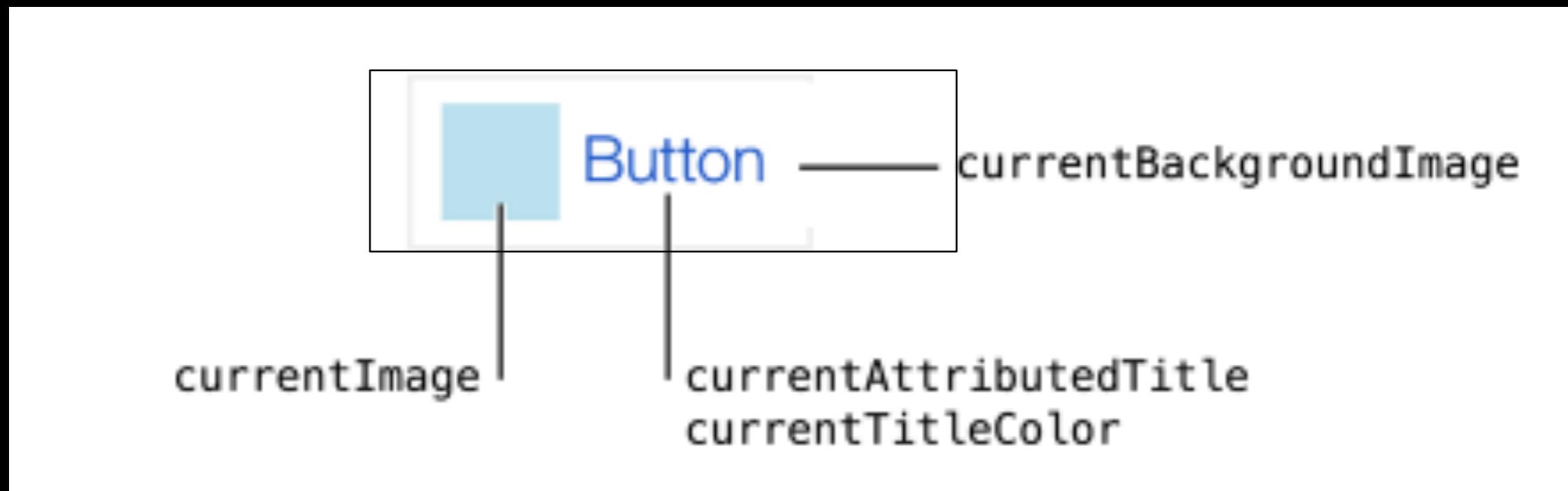
# The Launch Screen

- iOS displays the launch screen instantly when the user starts an app
- When the app is ready to use, iOS replaces the launch image with your app's first screen
- The launch image should look very similar to the first screen of your app, but without the app-specific info

# UIKit

# UIButton

## Appearance



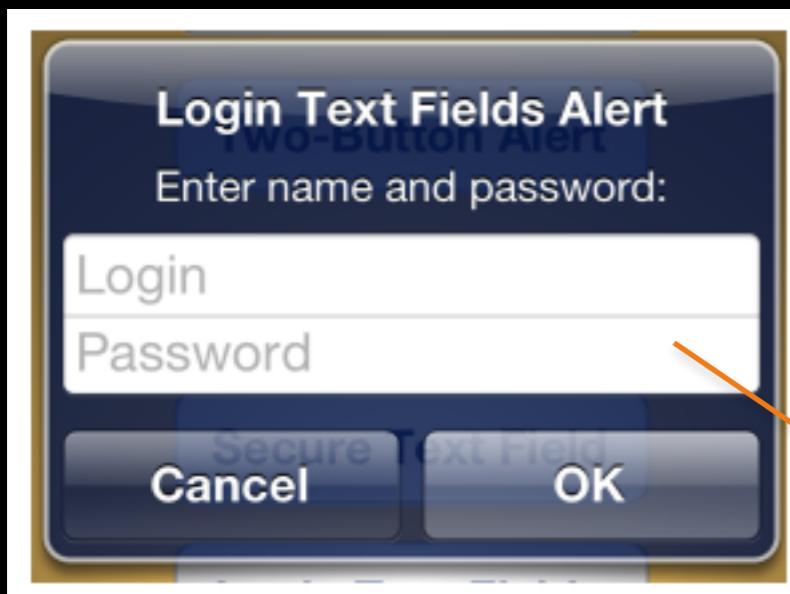
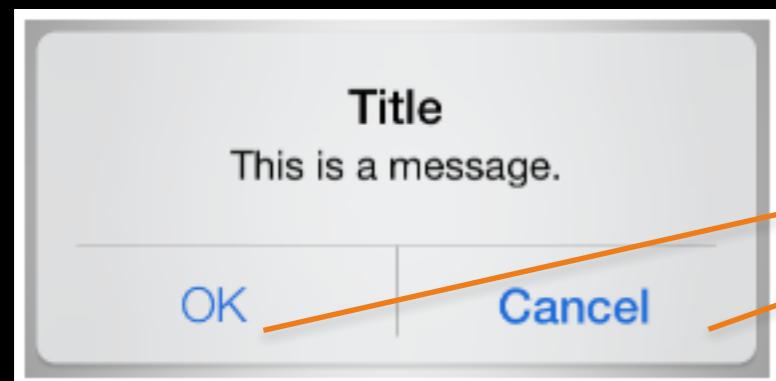
# UIButton

## State

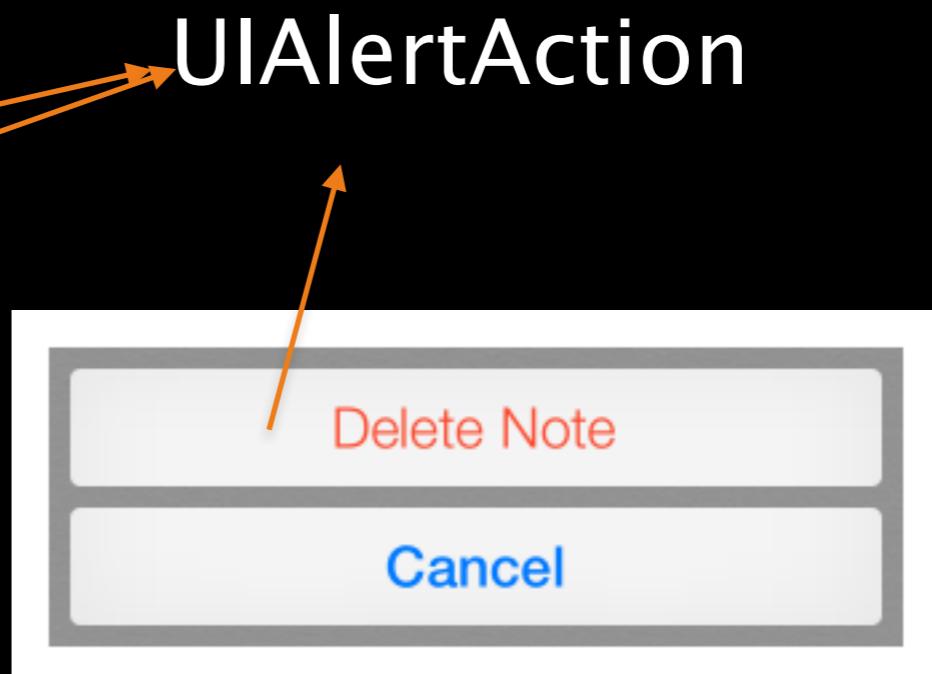
State	Description
normal	the button is enabled the user is not interacting with it
highlighted	when the button is tapped
selected	the button is selected has no effect on behavior or appearance
focused	the button has focus for Apple TV remote
disabled	the button does not interact with the user

# UIAlertcontroller

## Style



.alert



.actionsheet

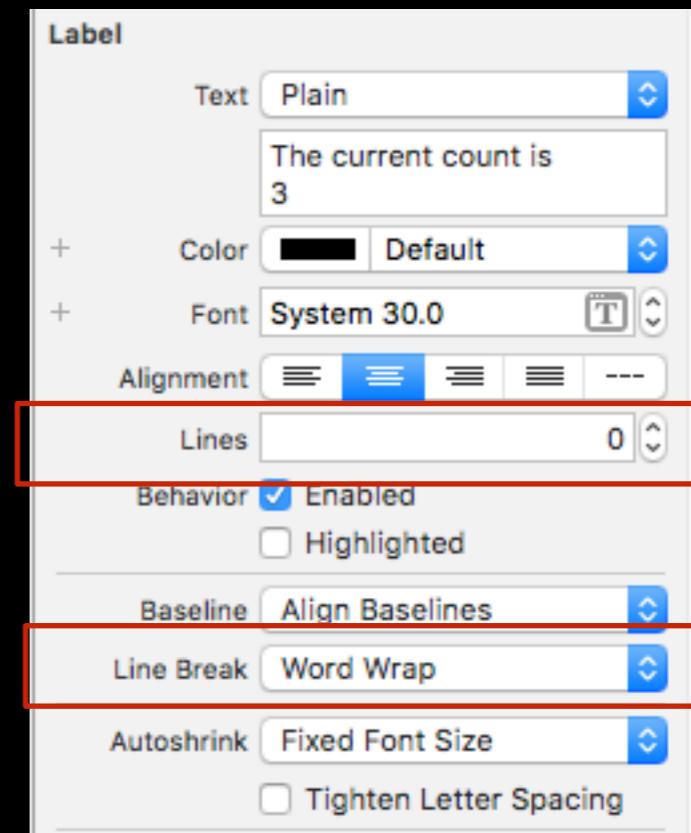
UITextField

UIAlertAction

# UILabel

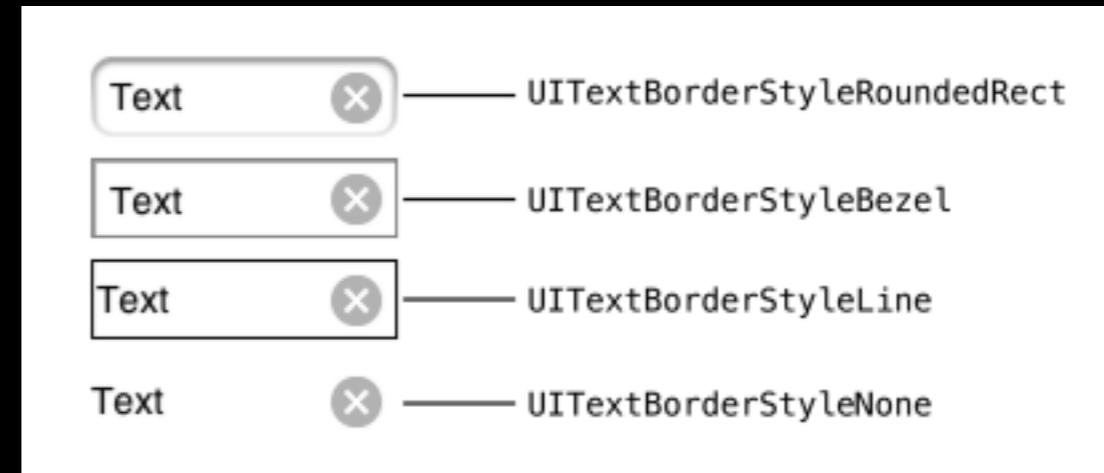
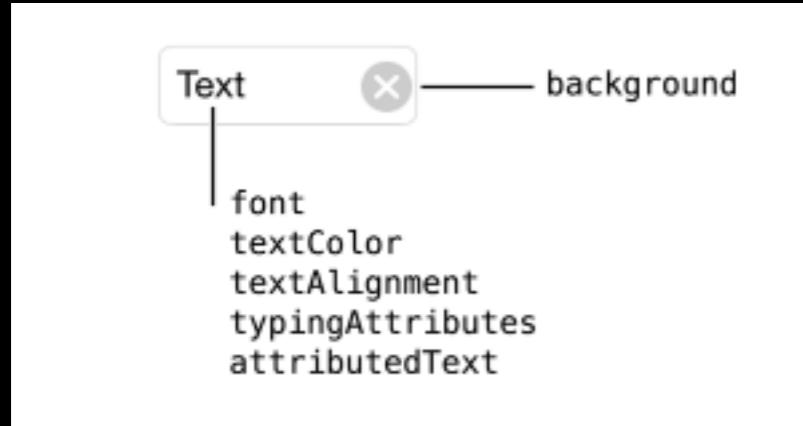
Displays multiple lines of static text

The volume of the ringer and alerts can be adjusted using the volume buttons.



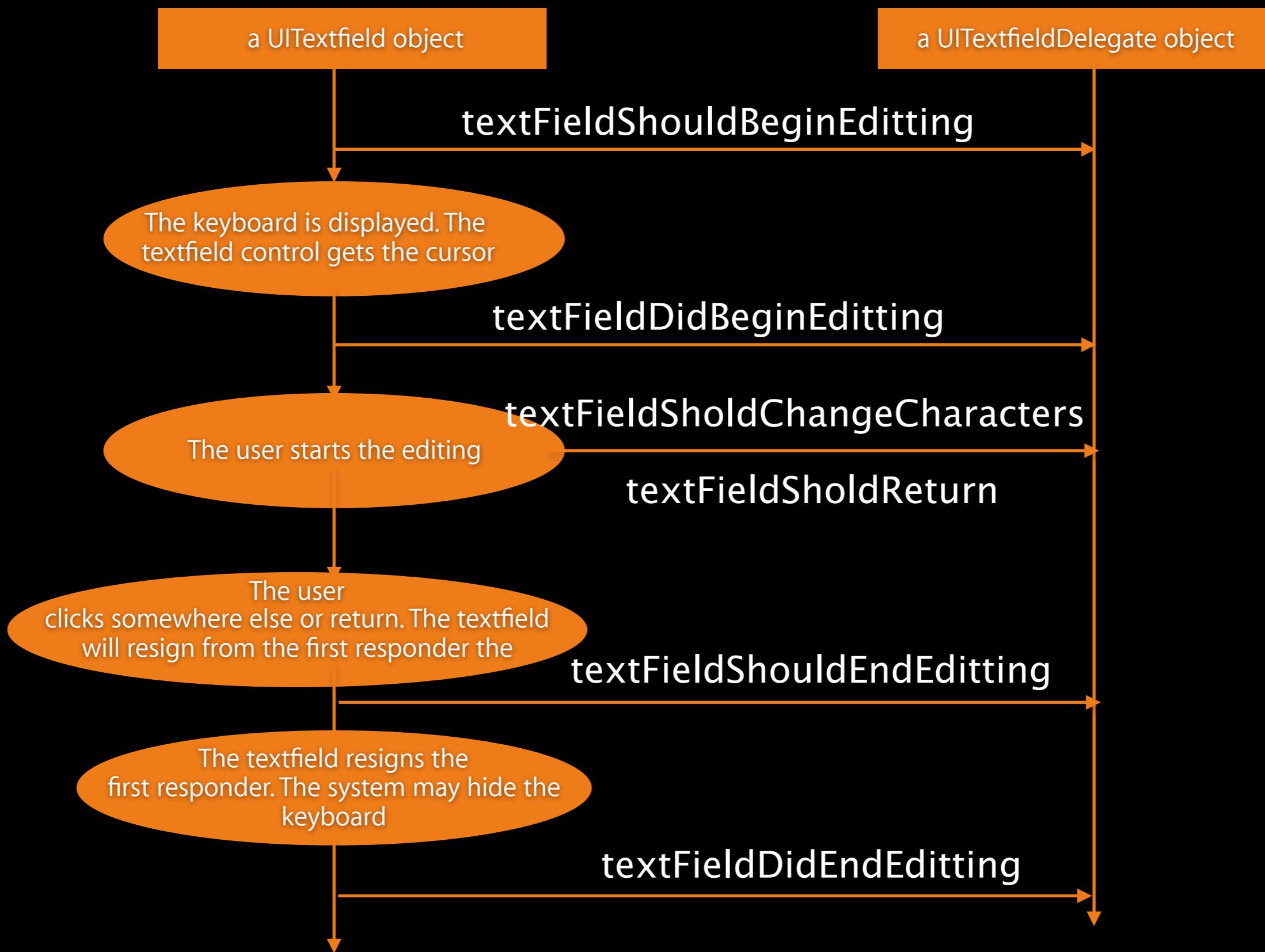
# UITextField

Displays an editable text area



The UITextFieldDelegate protocol

# The UITextFieldDelegate Protocol



- “UIKit User Interface Catalog” — <https://developer.apple.com/library/prerelease/content/documentation/UserExperience/Conceptual/UIKitUICatalog/>
- Sample code — “UIKit Catalog (iOS)”