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?

iOS

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

iOS Devices — iPhones

iPhone 7 Plus
iPhone 6x Plus

5.5in
Retina

1080x1920
px

iPhone 7
iPhone 6x

4.7in
Retina

750x1334
px

iPhone SE
iPhone 5x

4in
Retina

640x1136
px

iPhone 4x

3.5in
Retina

640x960
px

iPhone 3x
iPhone

3.5in

320x480
px

iOS Devices — iPads

iPad Pro

iPad Pro
iPad Air 1/2
iPad 3/4

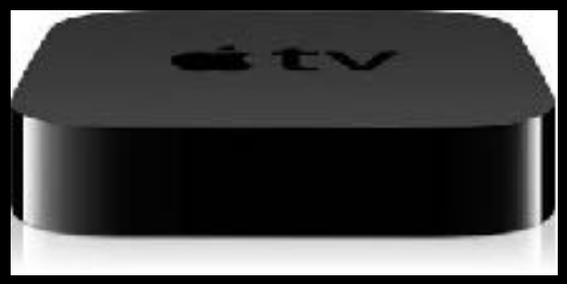
iPad 1/2

iPad Mini
2/3/4

iPad Mini

12.9in Retina	9.7in Retina	9.7in	7.9in Retina	7.9in
2048x2732 px	1536x2048 px	1536x2048 px	768x1024 px	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

iPhone OS 1.0

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

iPhone OS 2.0

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

iOS 6.0

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

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

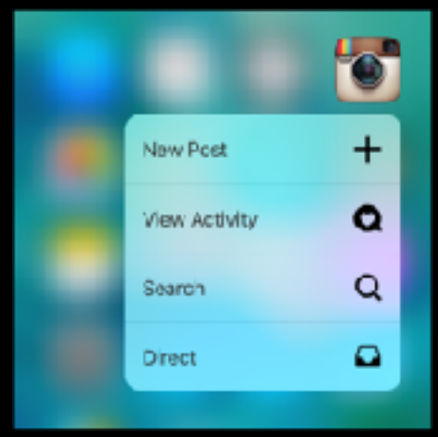
Metal

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

tvOS

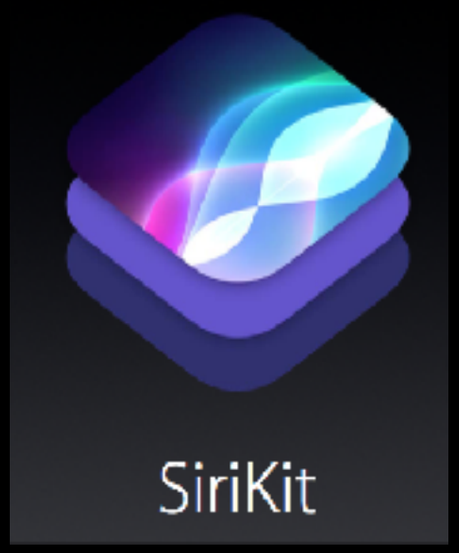
A First Water-Resistent iPhone

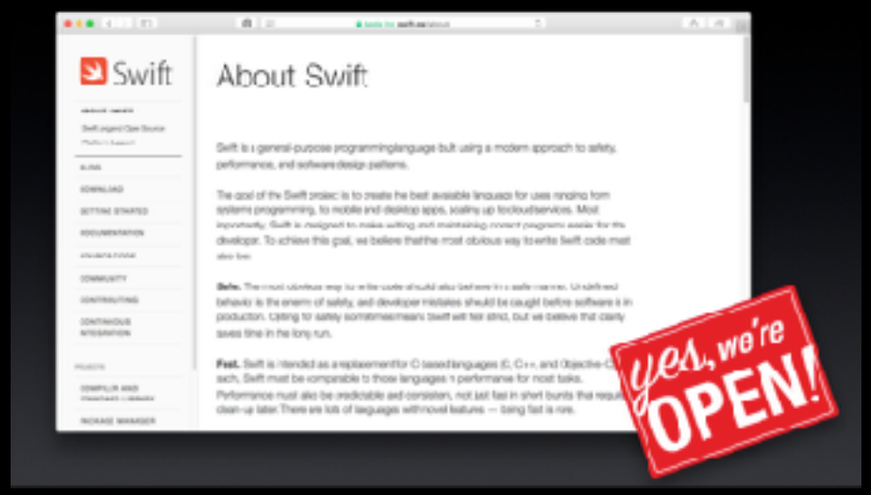
On September 7 2016, Apple announced 4.7-inch iPhone 7 and 5.5-inch iPhone 7 Plus, Apple Watch Serials 2

iOS 10 —Opening Swift, Siri, iMessage

In **WWDC 2016, Apple announced iOS 10.0.**

iMessage Apps





Class Schedule

Class Schedule

	Schedule
1	A Single-View App
2	A Table-based App with Navigation
3	
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

	Weight	Due Day
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

Reference

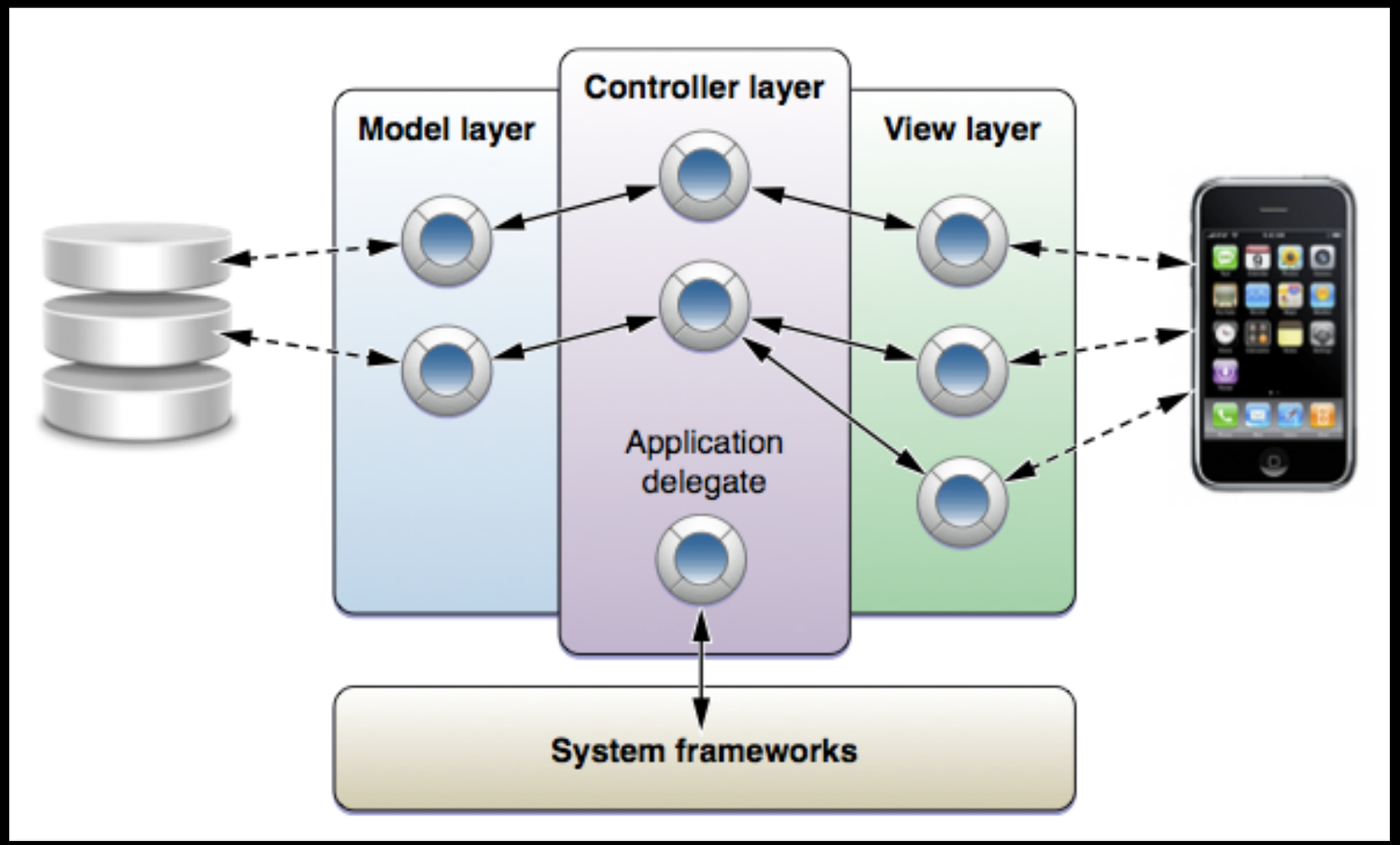
- iOS Developer Library (http://developer.apple.com/library)
- swift.org
  - The Swift Programming Language (Swift 3.0.1)

Class FTP Site

- ftp://202.205.101.168/
  - user name: zyp_s, no password
  - submit assignments, download class materials

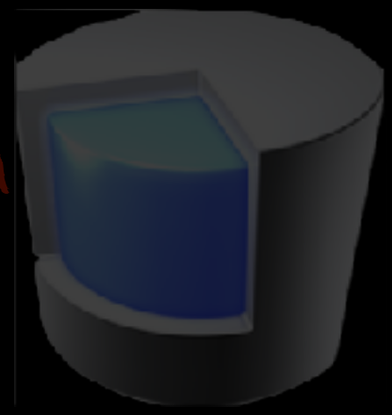
iOS Design Patterns

Model-View-Controller Design Pattern



Views

Controller



Model

User Interface

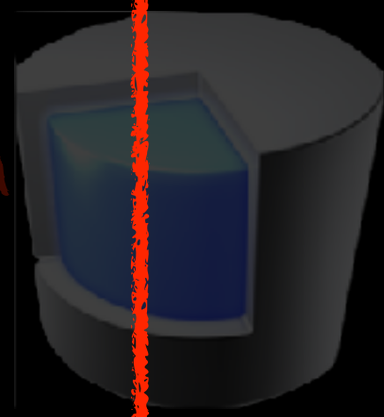
Data

Views

Controller

- Allows a user to manipulate data

User Interface



Model

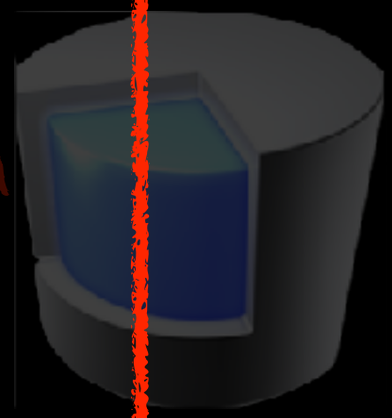
Data

Views

Controller

- Allows a user to manipulate data
- Does not store data

User Interface



Model

Data

Views


iPad
9:41 AM
Locations
Golden Gate Bridge
Checklist
Share
The Golden Gate Bridge is a suspension bridge spanning the Golden Gate, the opening of the San Francisco Bay into the Pacific Ocean. As part of both U.S. Route 101 and California State Route 1, the structure links the city of San Francisco, on the northern tip of the San Francisco Peninsula, to Marin County. It is one of the most internationally recognized symbols of San Francisco, California, and the United States. It has been declared one of the modern Wonders of the World by the American Society of Civil Engineers. The Frommers travel guide considers the Golden Gate Bridge "possibly the most beautiful, certainly the most photographed, bridge in the world".
LENGTH 8,981 ft (2,737.4 m), about 1.7 mi
WIDTH 90 ft (27.4 m)
HEIGHT 746 ft (227.4 m)
LONGEST SPAN 4,200 ft (1,280.2 m)
Golden Gate Bridge
101

Views

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

Model

Controller

View

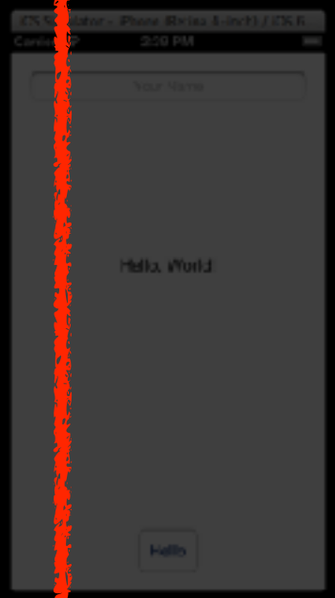
User Interface

Data

Model

Controller

- Application data and state



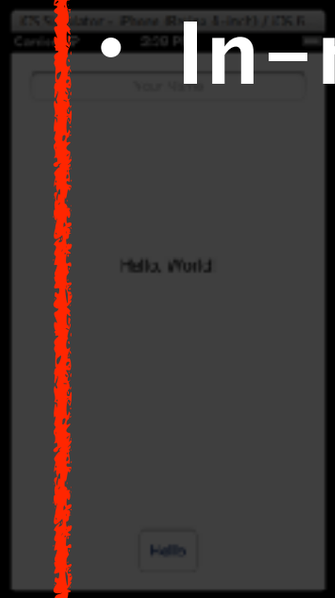
User Interface

Data

Model

Controller

- Application data and state
- In-memory storage



User Interface

Data

Model

Controller

View

- Application data and state
- In-memory storage
- Persistent storage (file, database, etc.)

User Interface

Data

Model

Controller

- Application data and state
- In-memory storage
- Persistent storage (file, database, etc.)
- Not connected directly to UI

Data

User Interface

Model

Controller

View

User Interface

- Application data and state
- In-memory storage
- Persistent storage (file, database, etc.)
- Not connected directly to UI
- Independent to UI: same model/different presentations

Data

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

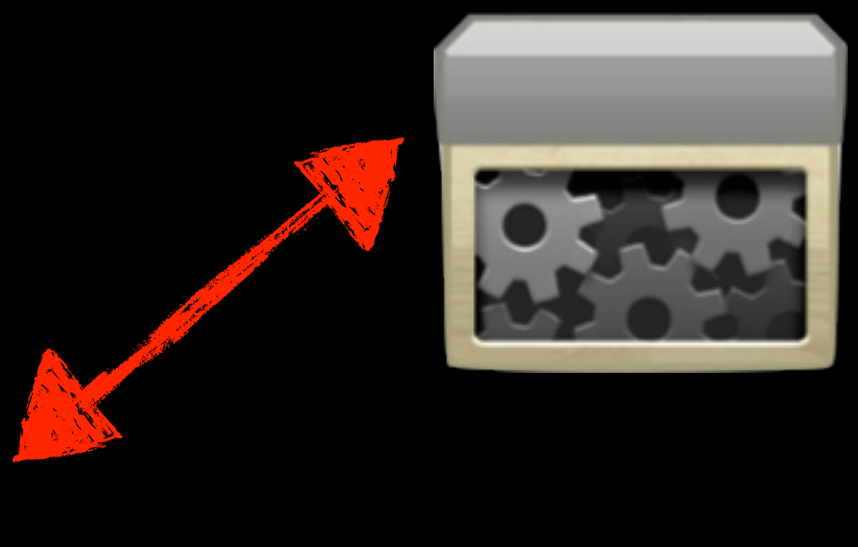
Model

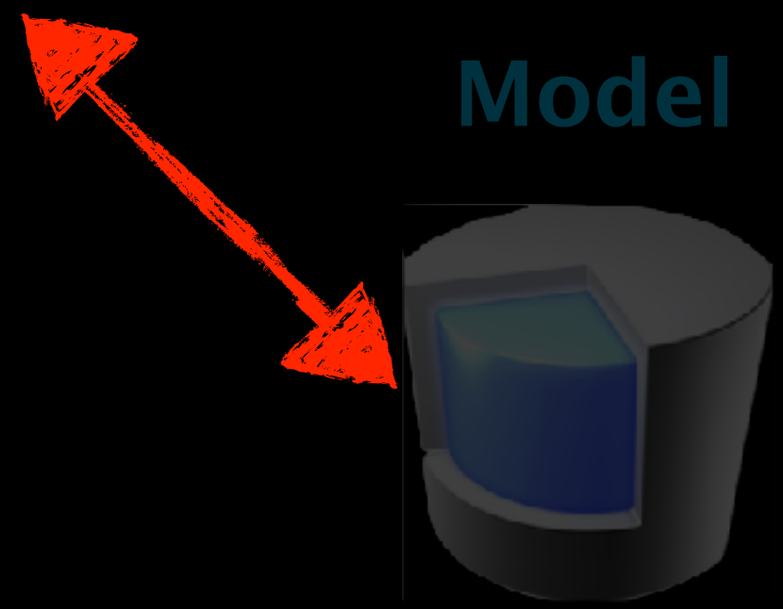
- Property list - a formatted file
- Object archiving - store data in a binary format
- Core data - store data in a database
- User defaults - store data in system “Settings”

View Controllers

View

User Interface



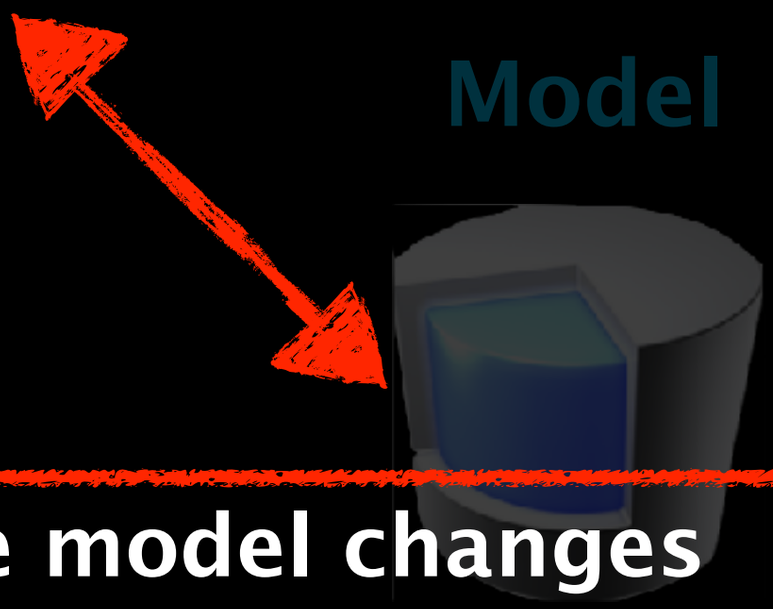


Model

Data

View Controllers

View



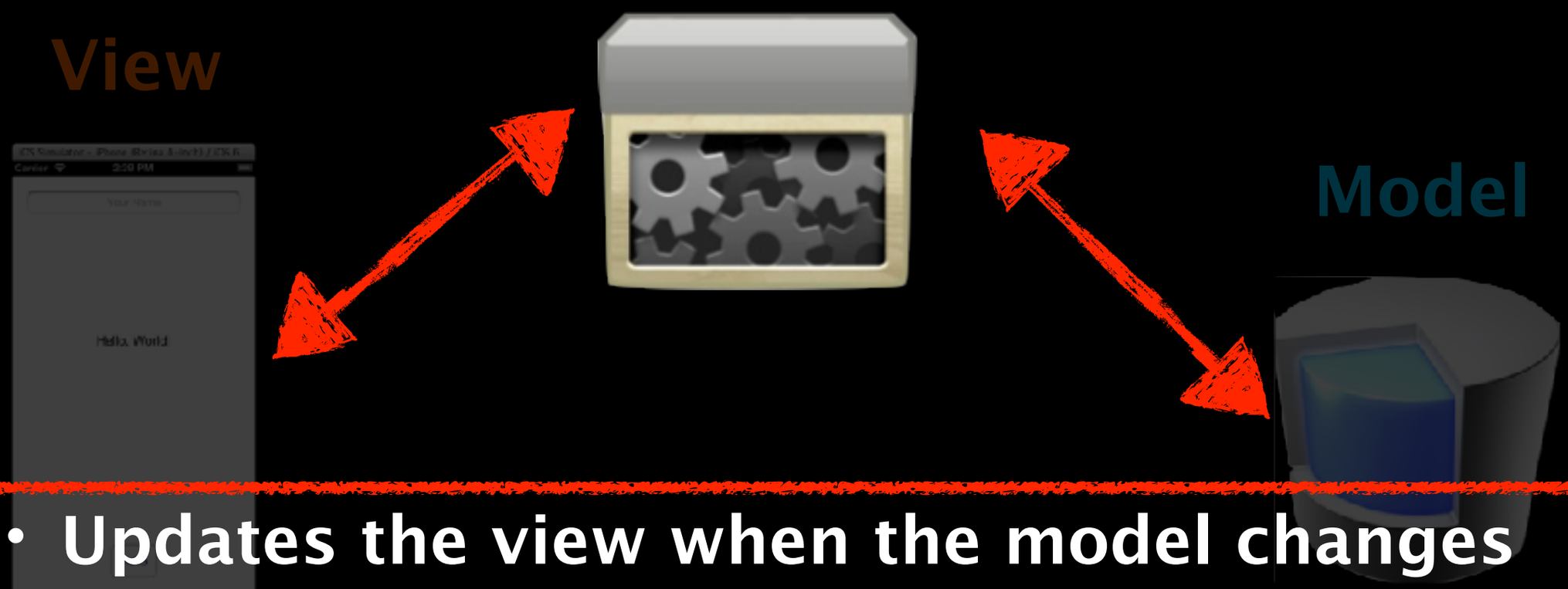
Model

- Updates the view when the model changes

User Interface

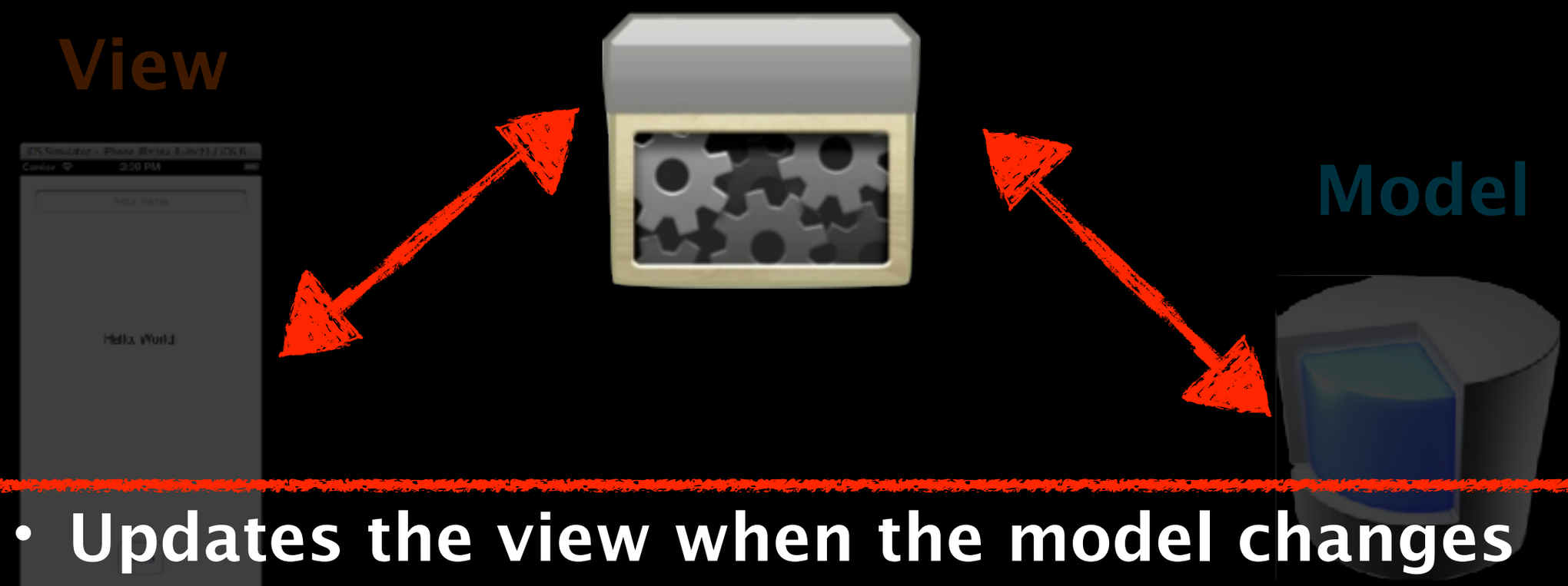
Data

View Controllers



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

View Controllers



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

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

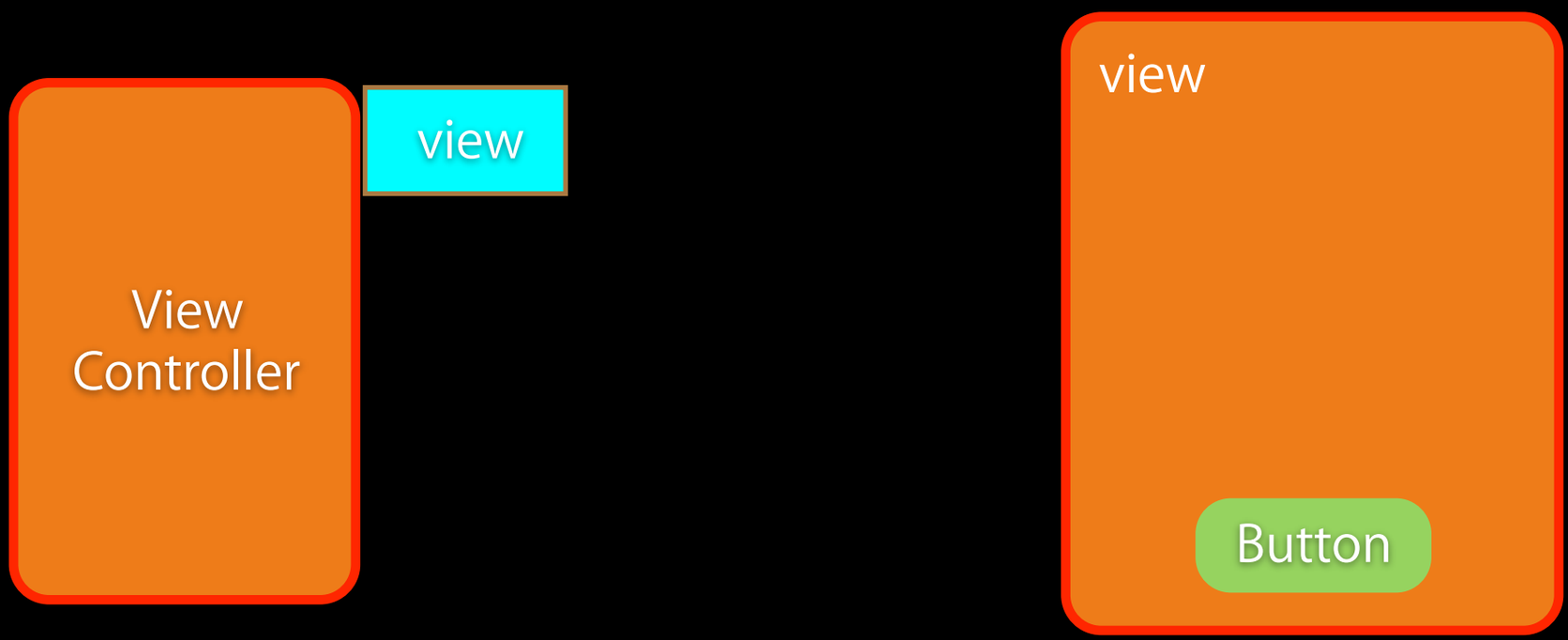
Outlets

Outlets

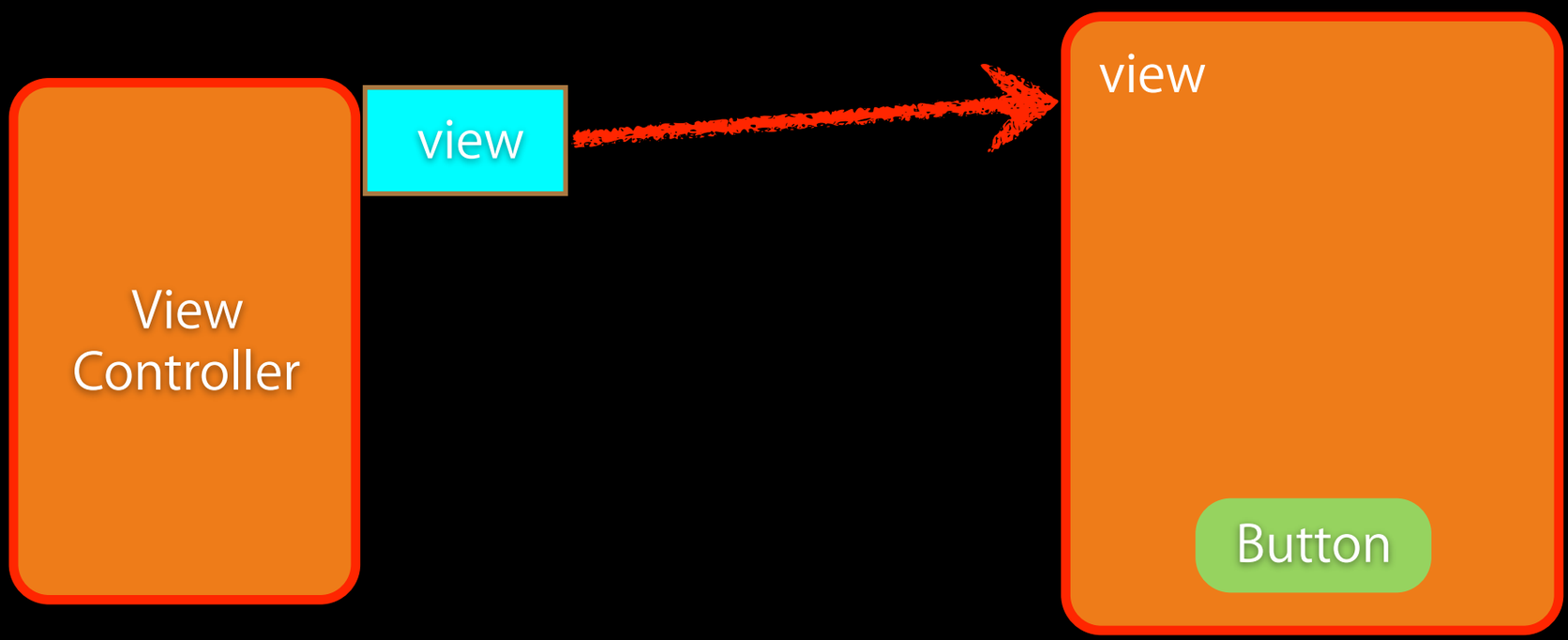
Outlets

Outlets

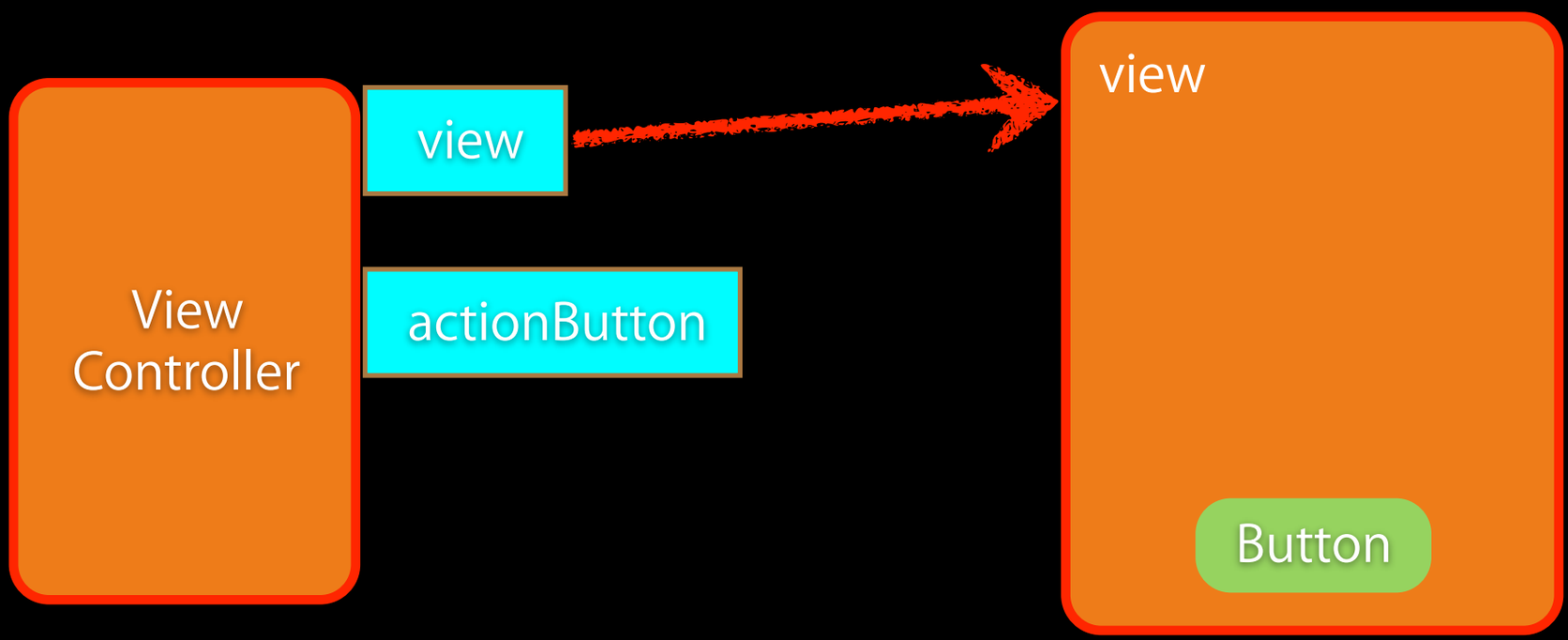
Outlets



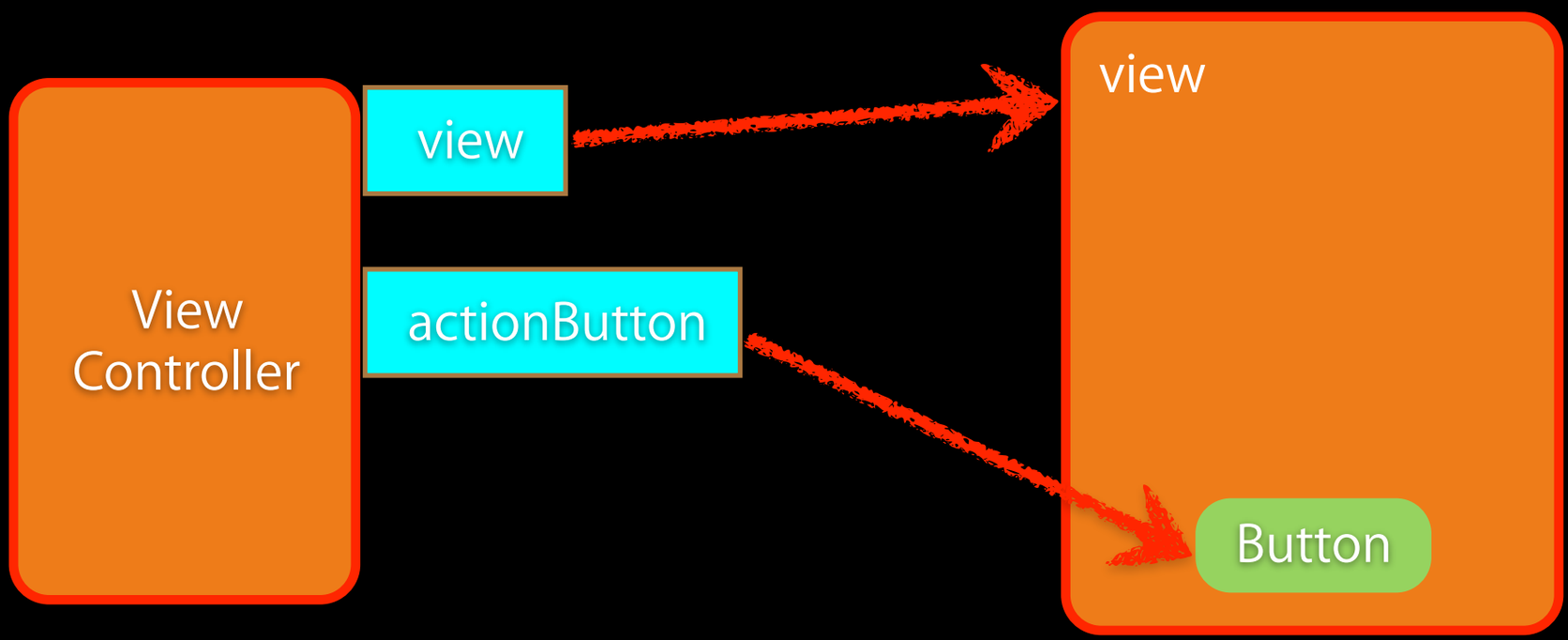
Outlets



Outlets

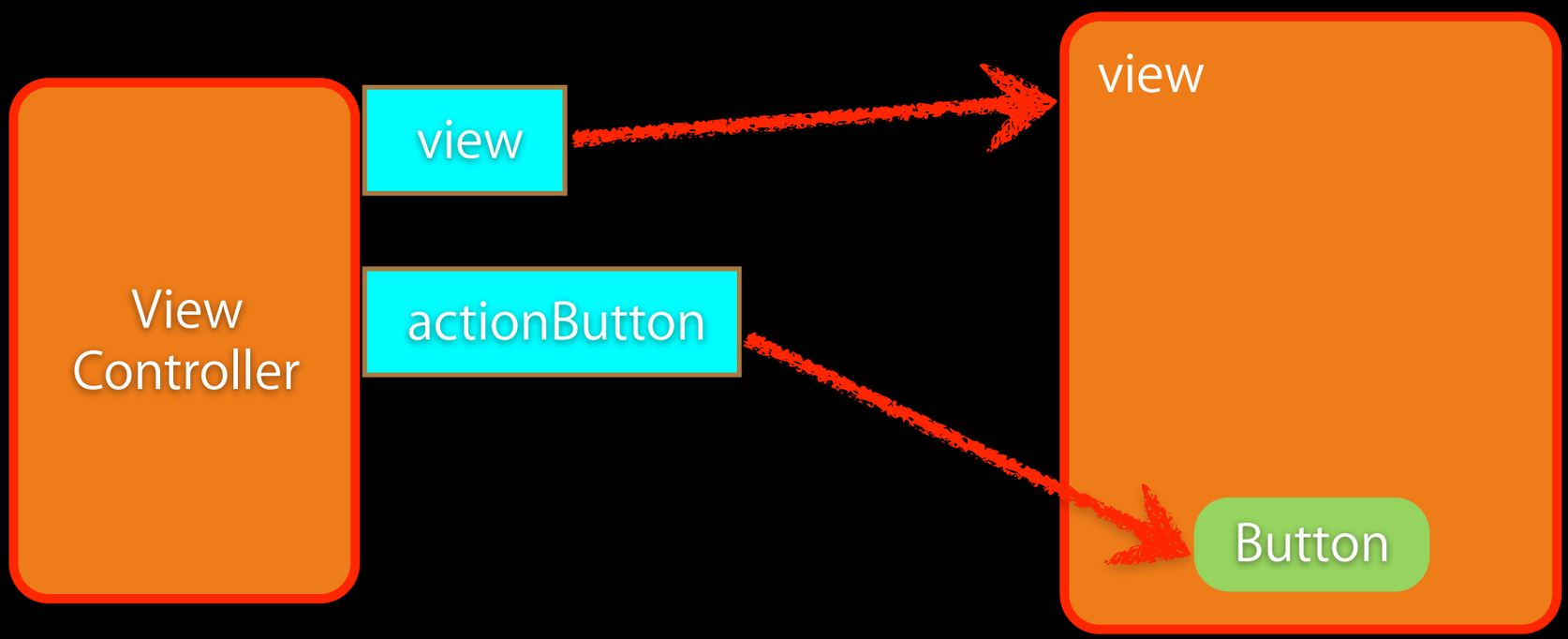


Outlets



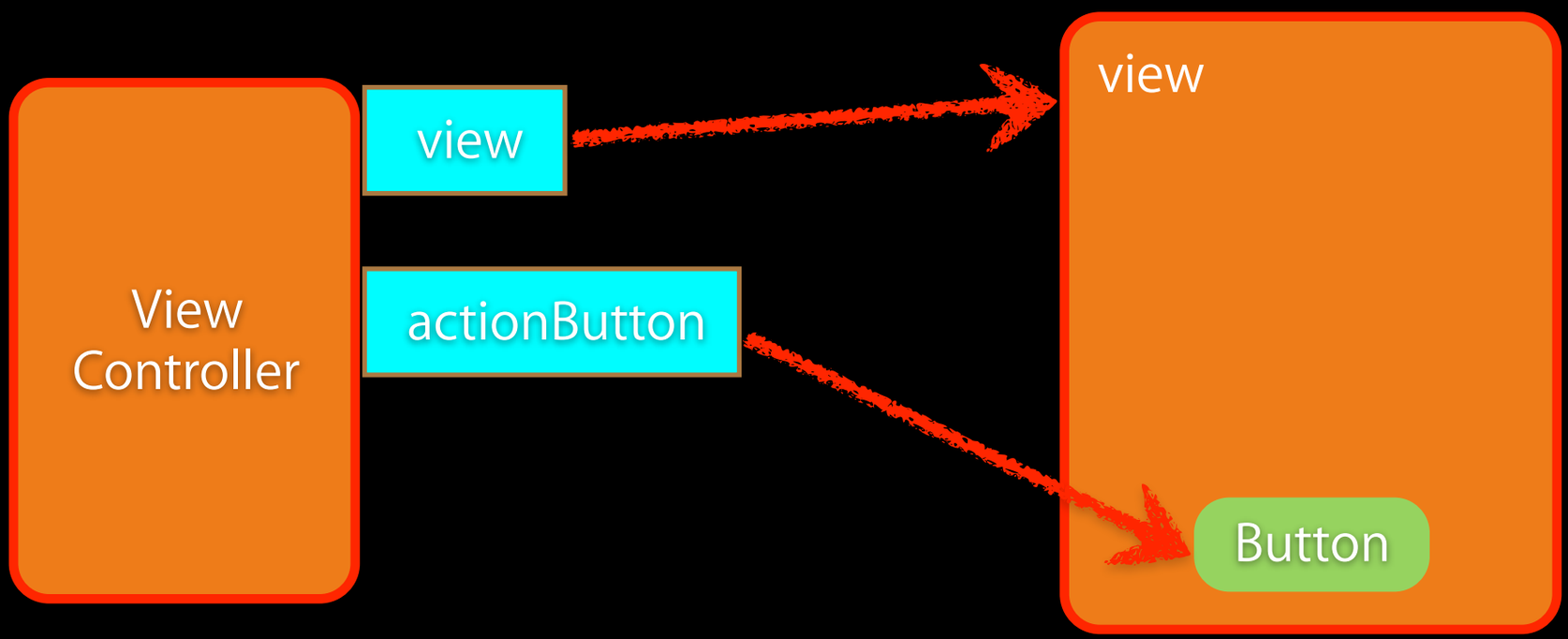
Outlets

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



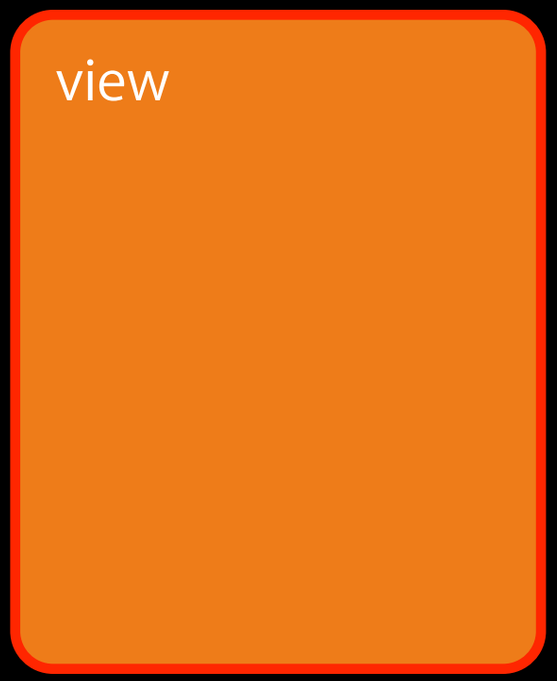
Outlets

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

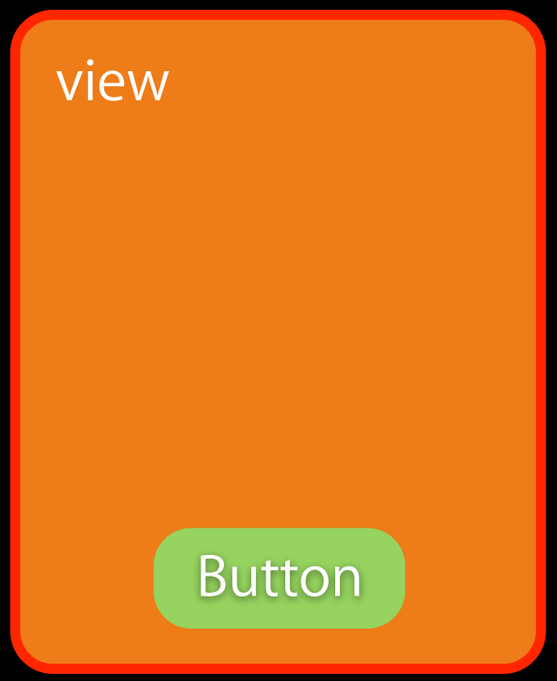


Target-Action Design Pattern

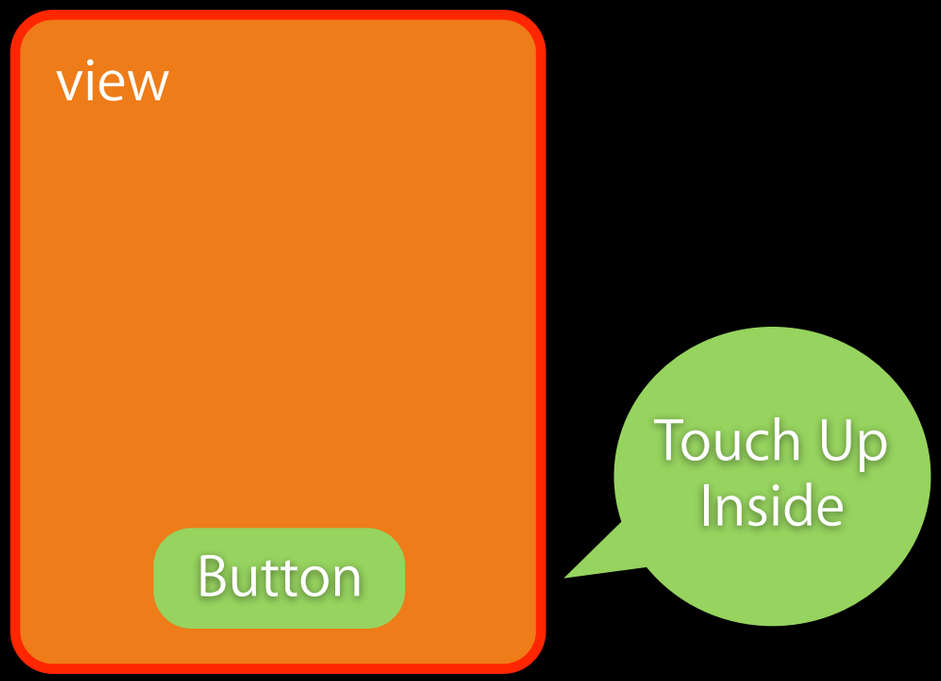
Target-Action Design Pattern



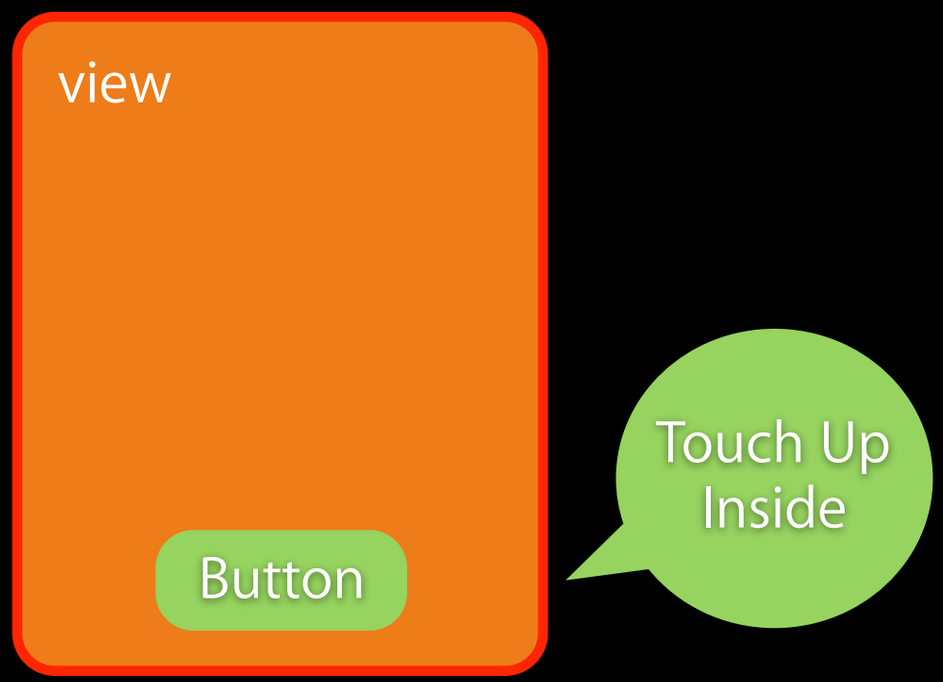
Target-Action Design Pattern



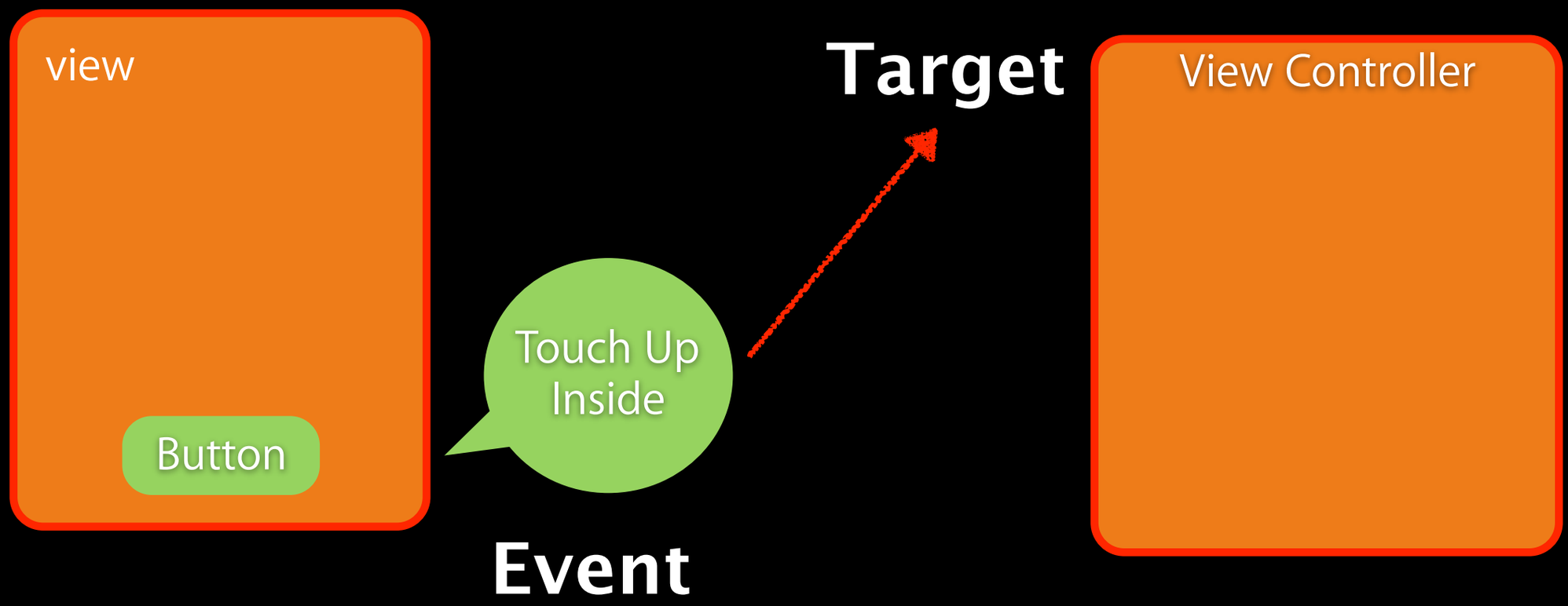
Target-Action Design Pattern



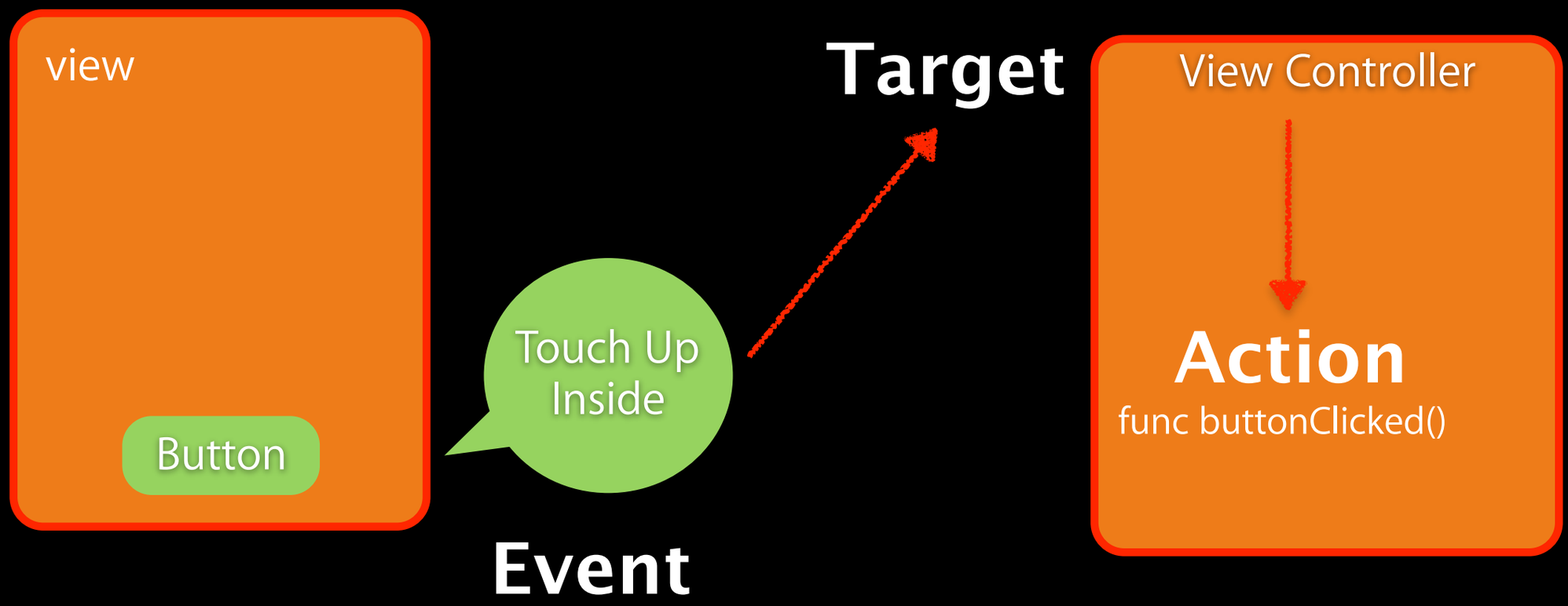
Target-Action Design Pattern



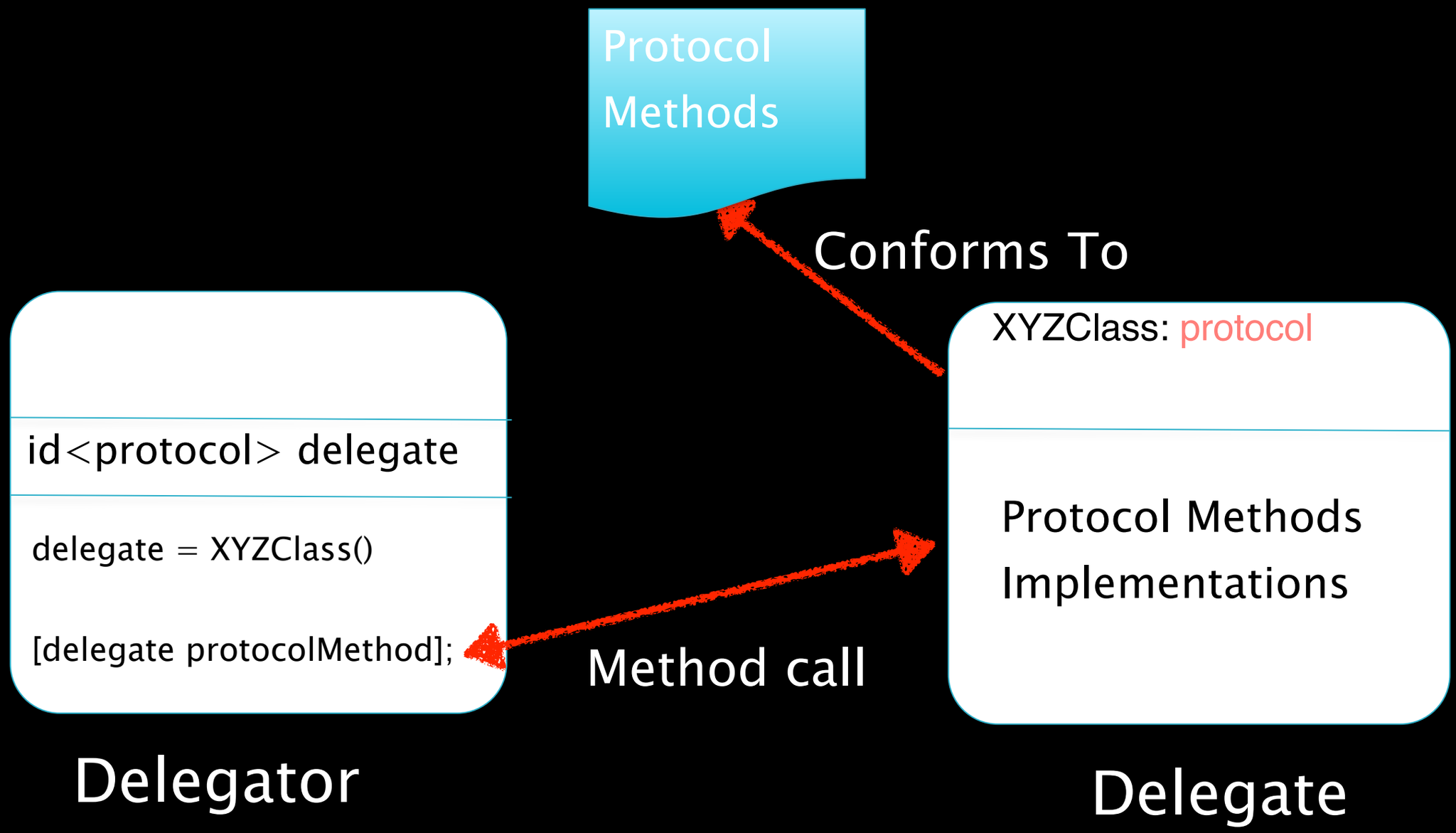
Target-Action Design Pattern



Target-Action Design Pattern

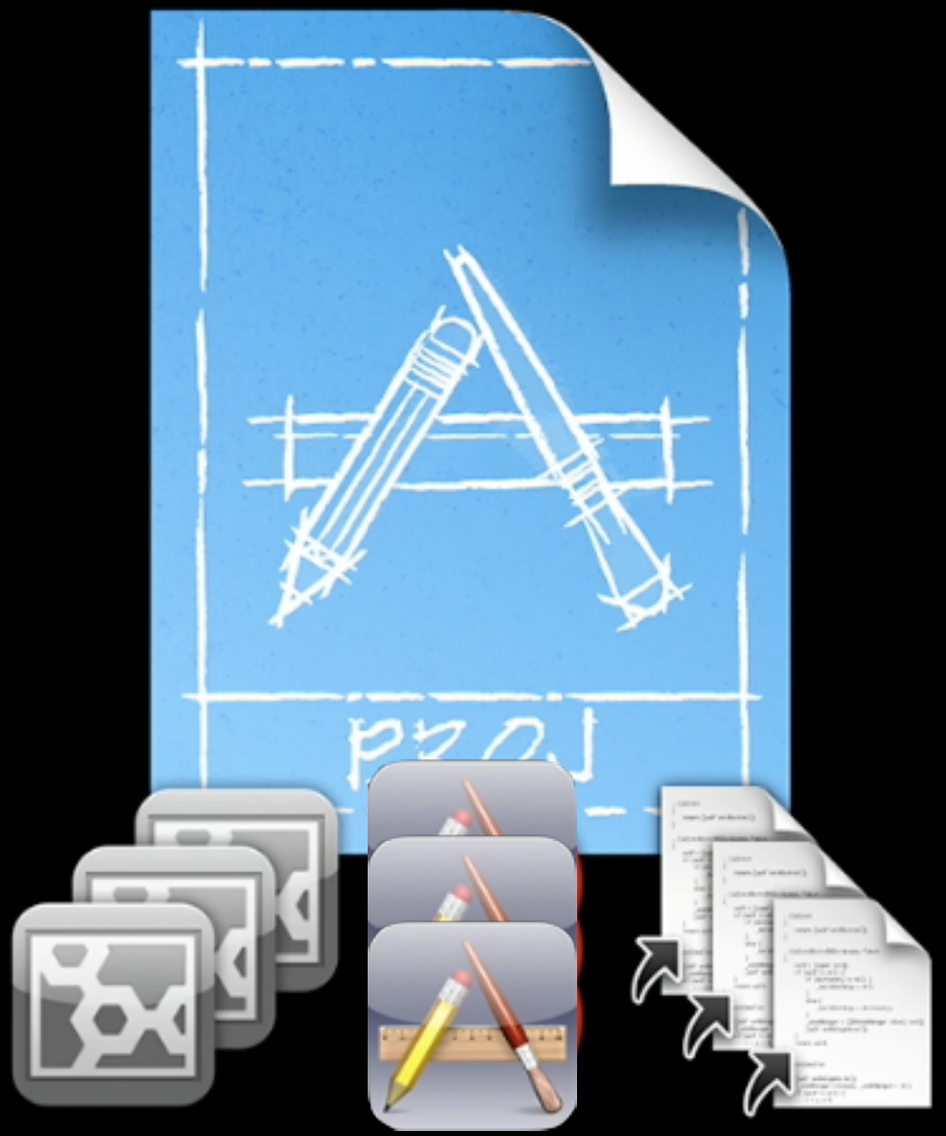


Delegation Design Pattern



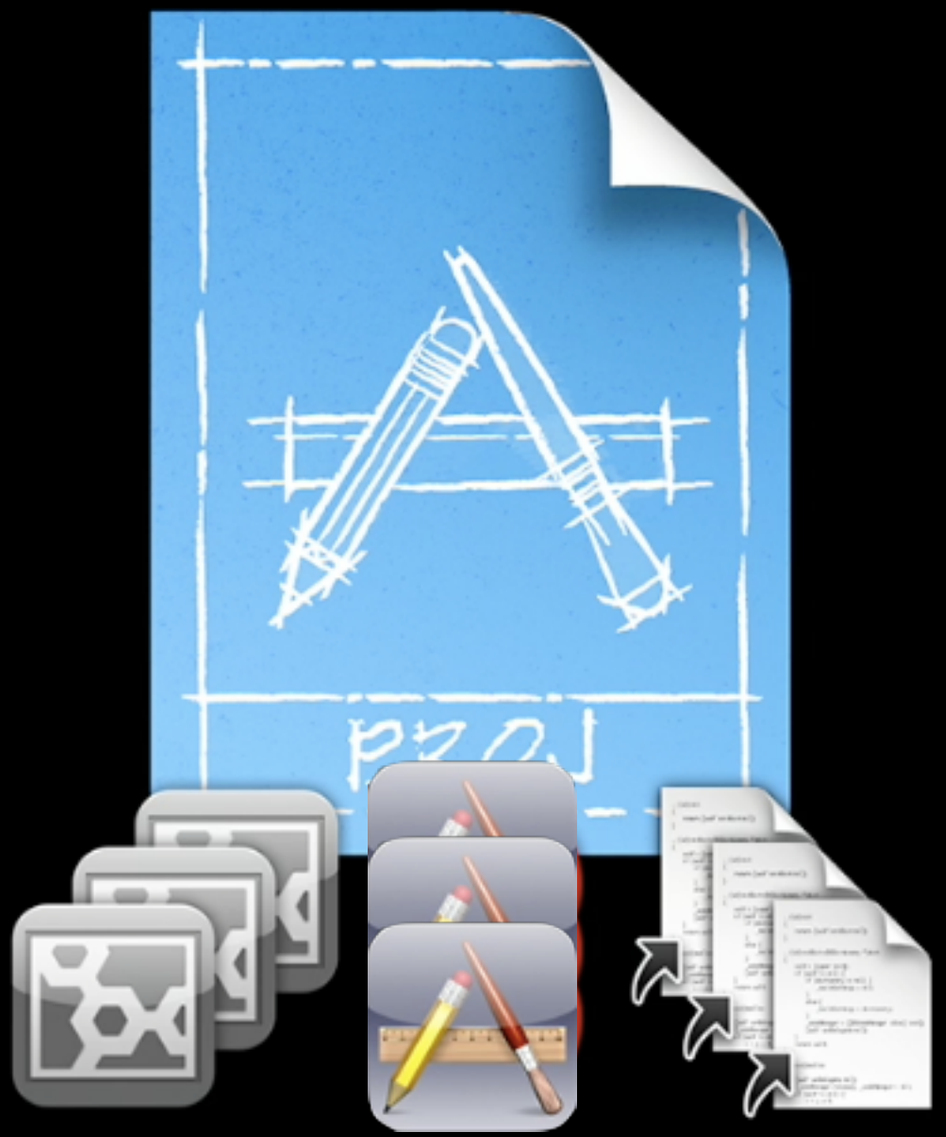
Xcode

Project



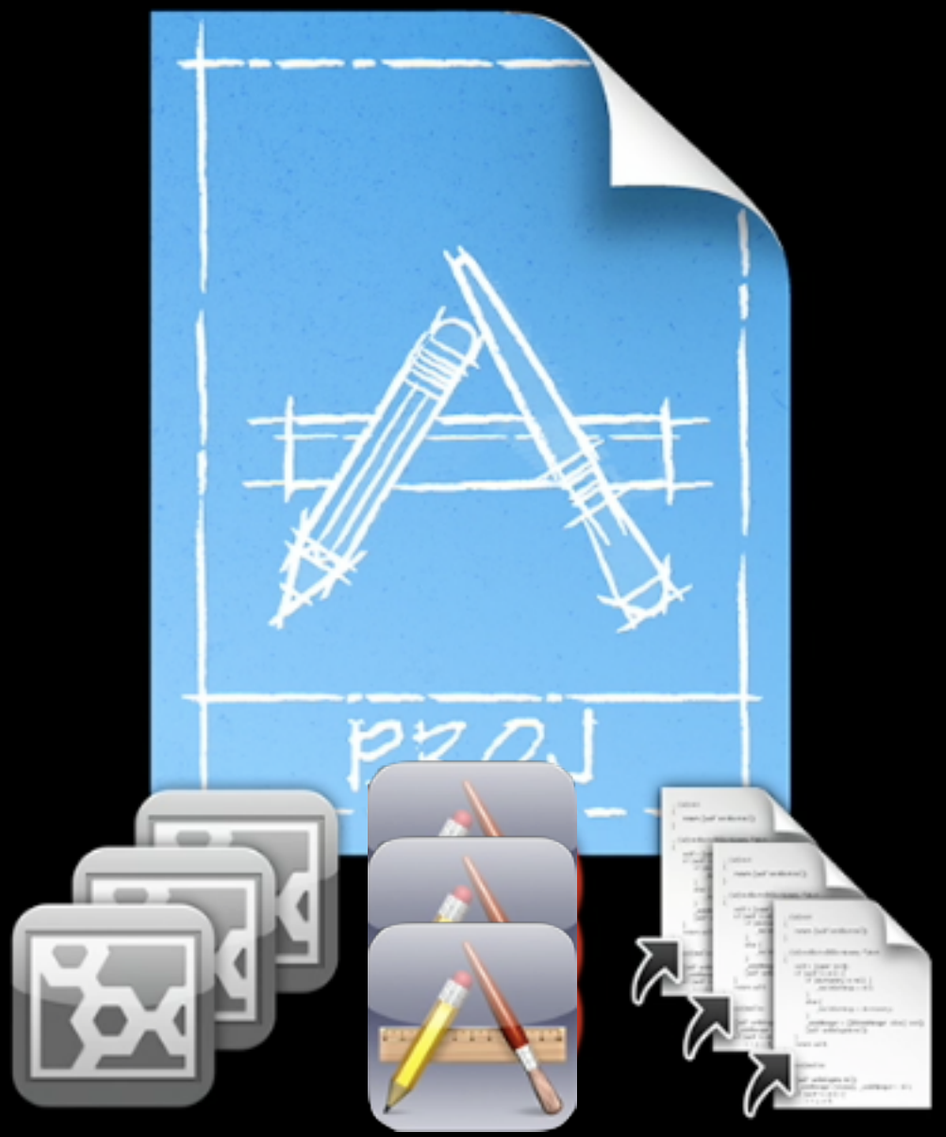
Project

- Source files



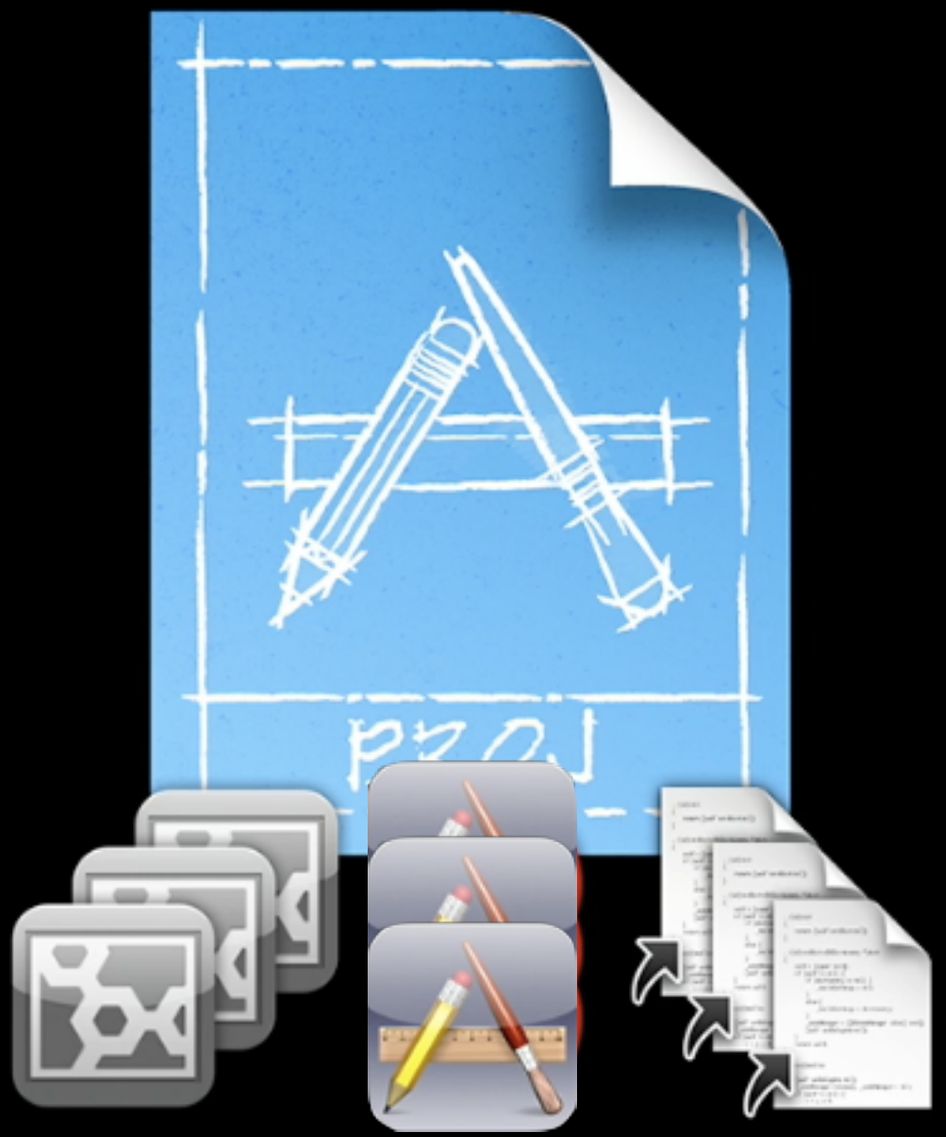
Project

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



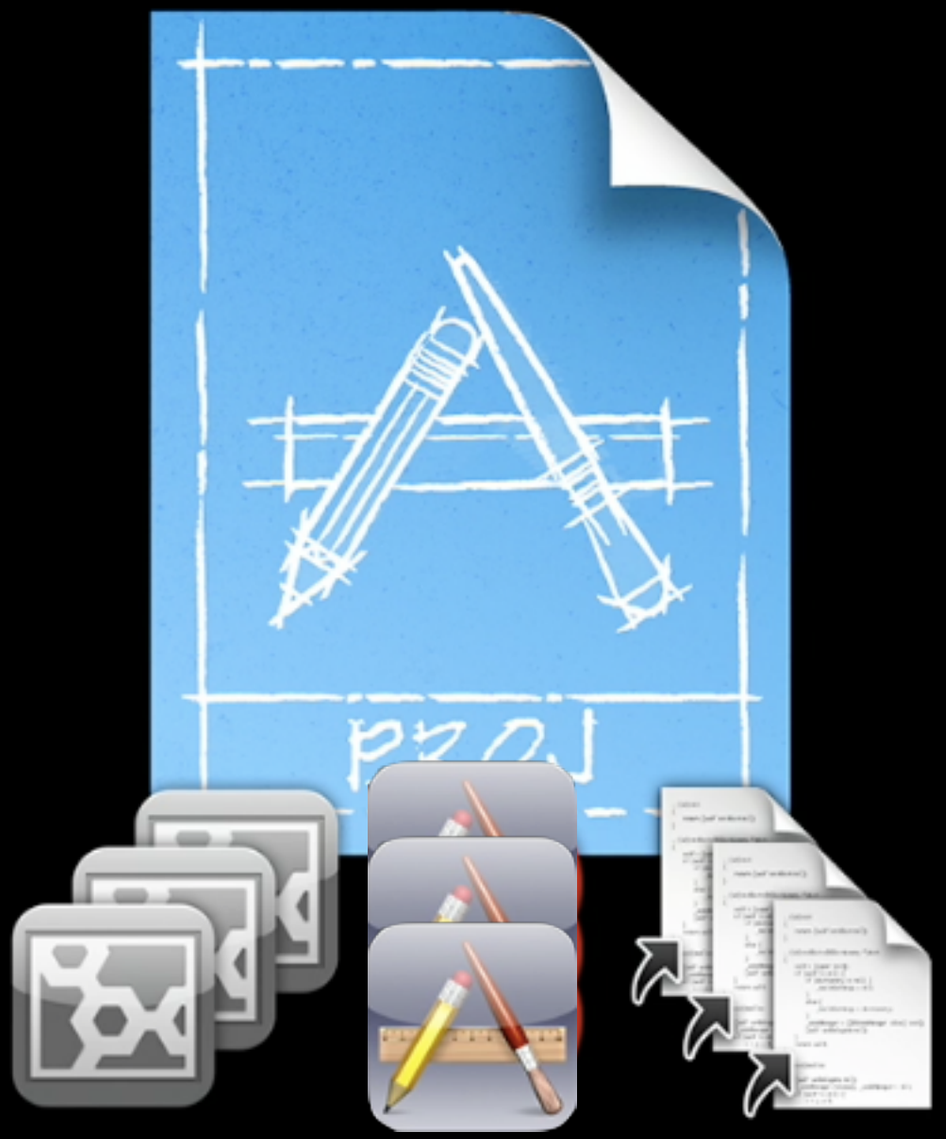
Project

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

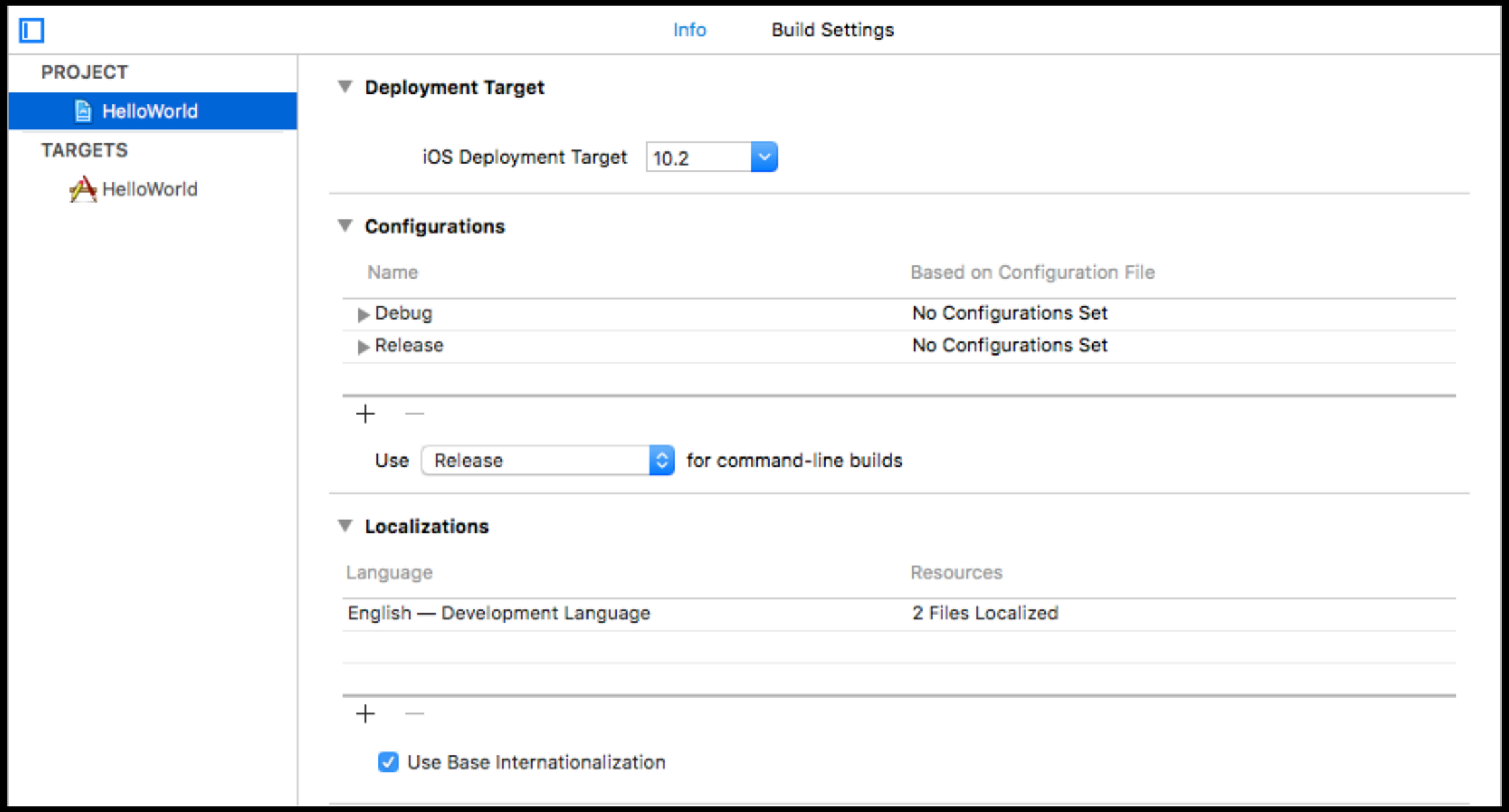


Project

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



The Project Setting



Build Configurations

Build Configurations

- Build setting variants

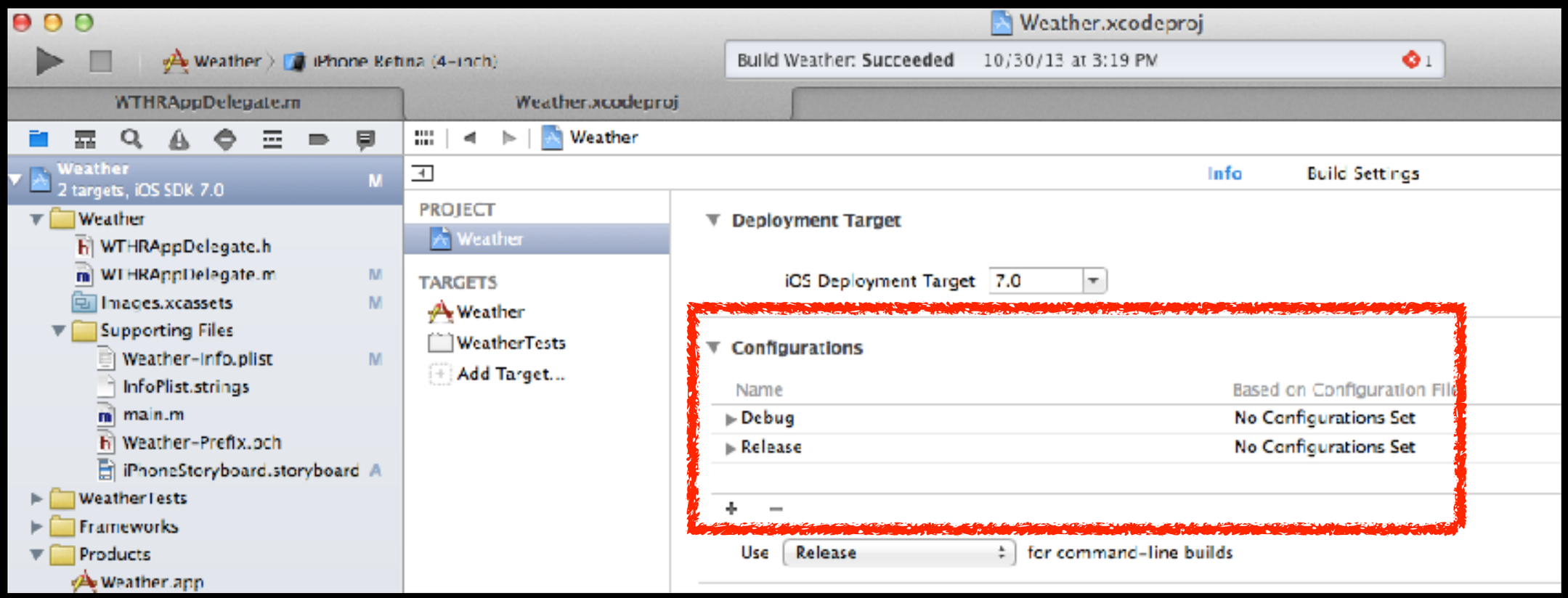
Build Configurations

- Build setting variants
- Project-level

Build Configurations

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

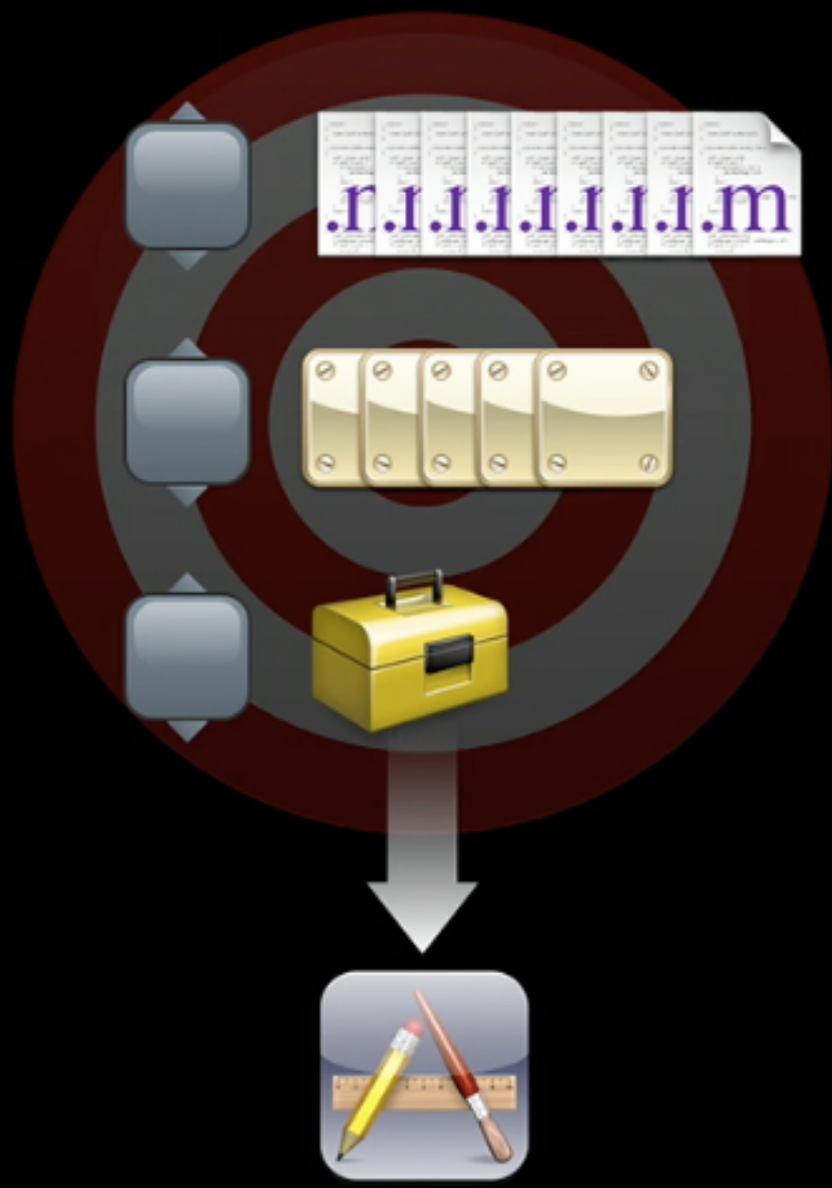
Build Configurations



Target — excutable

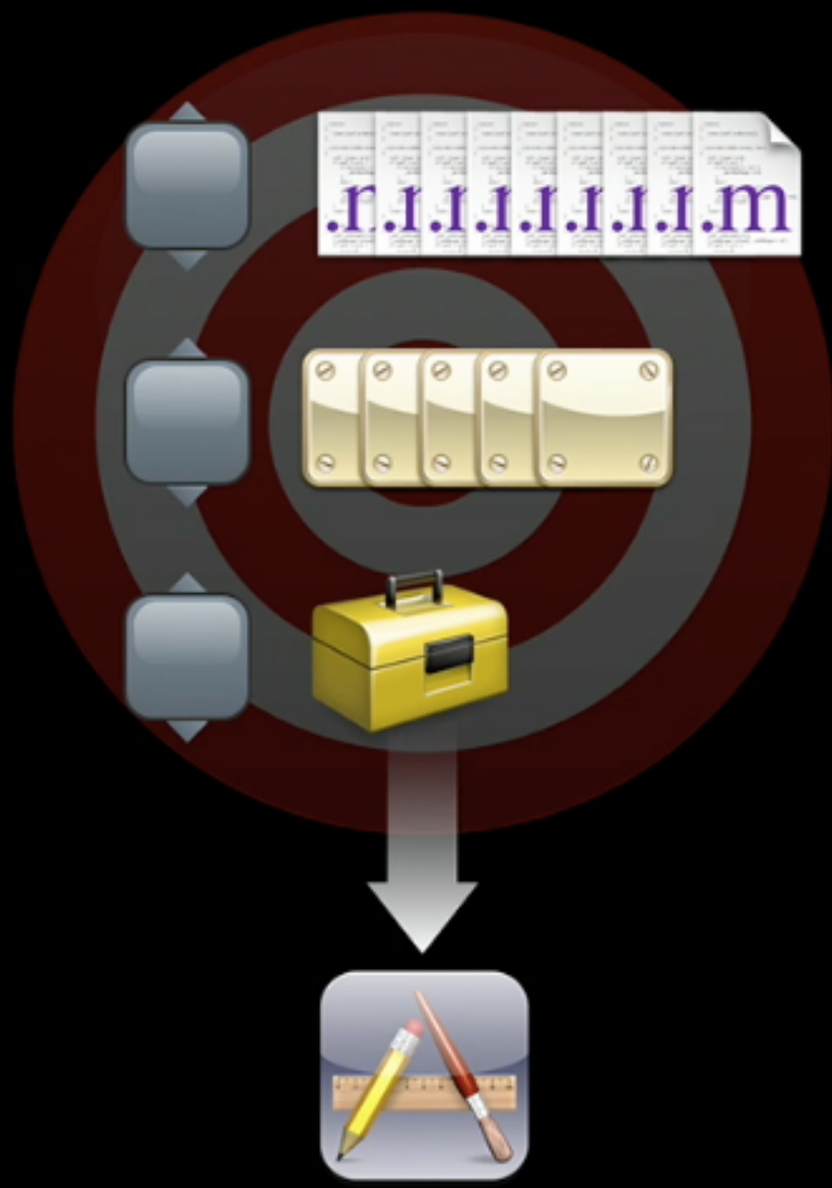
Target — excutable

- Some or all source/resource files



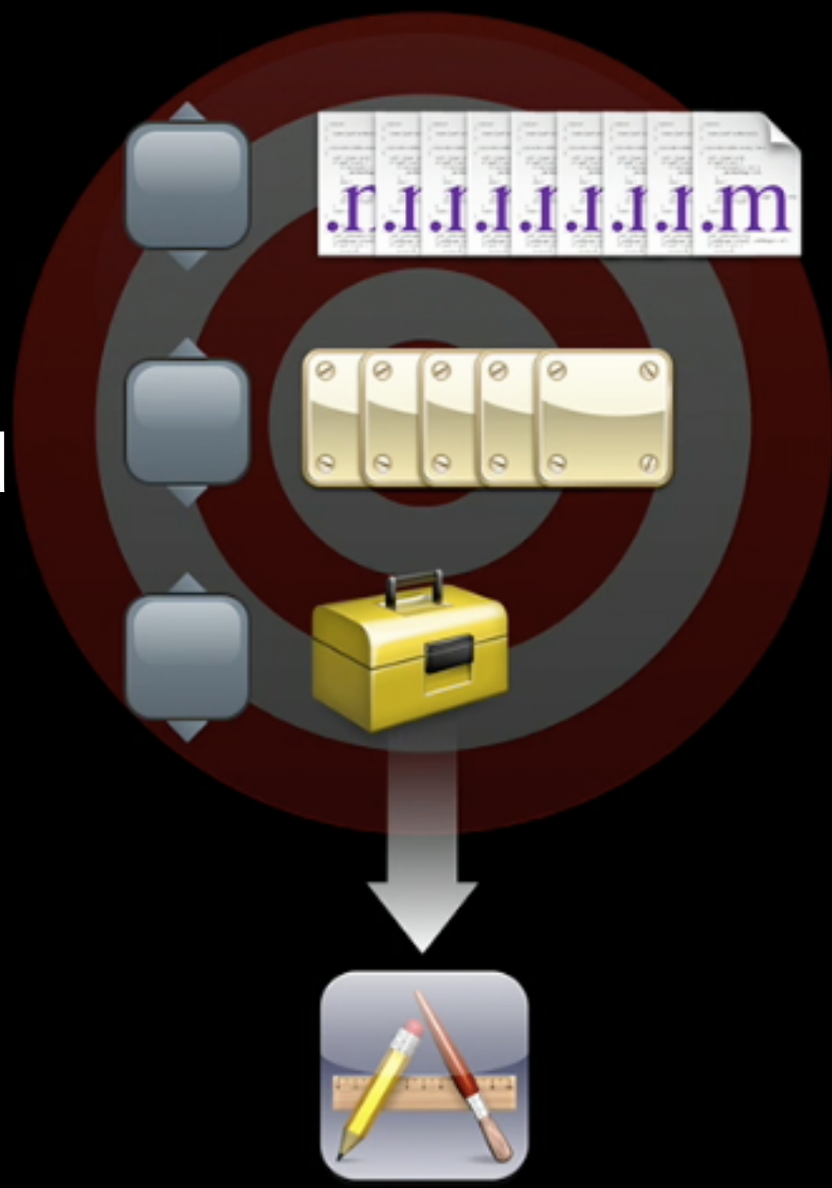
Target — excutable

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



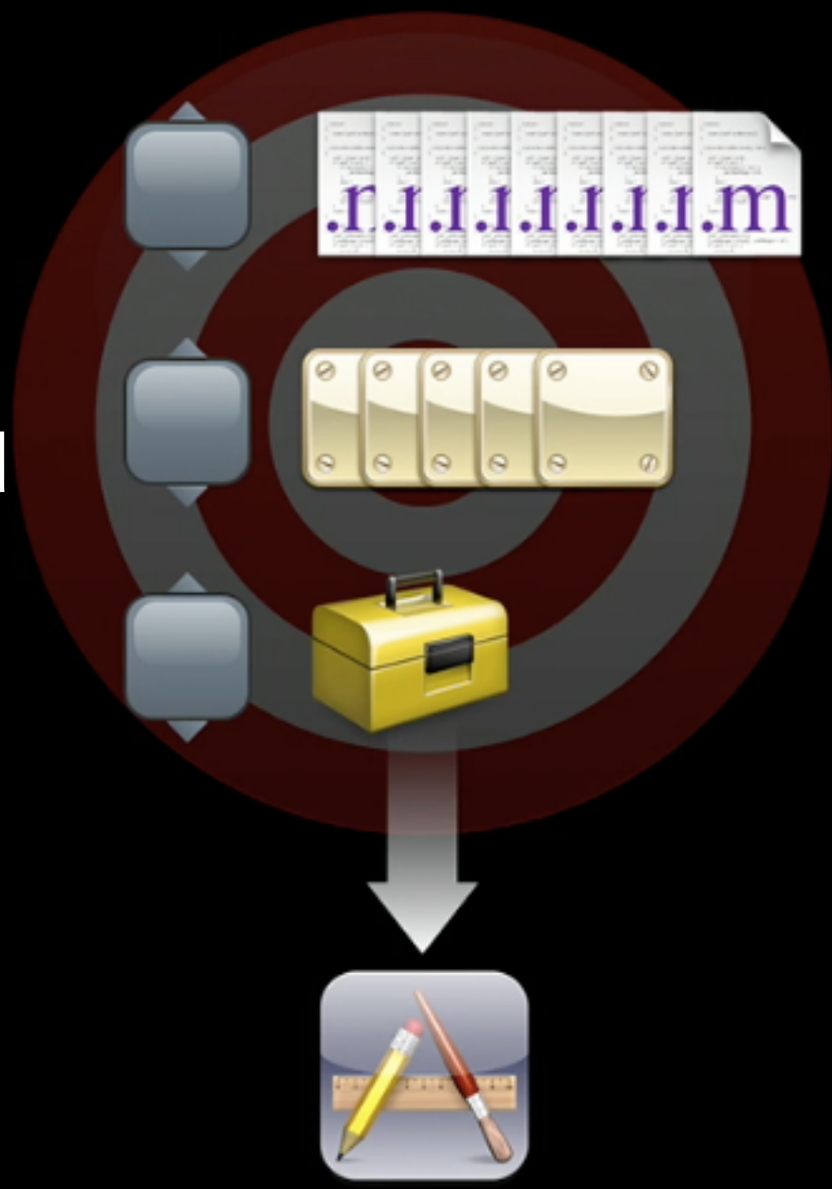
Target — excutable

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



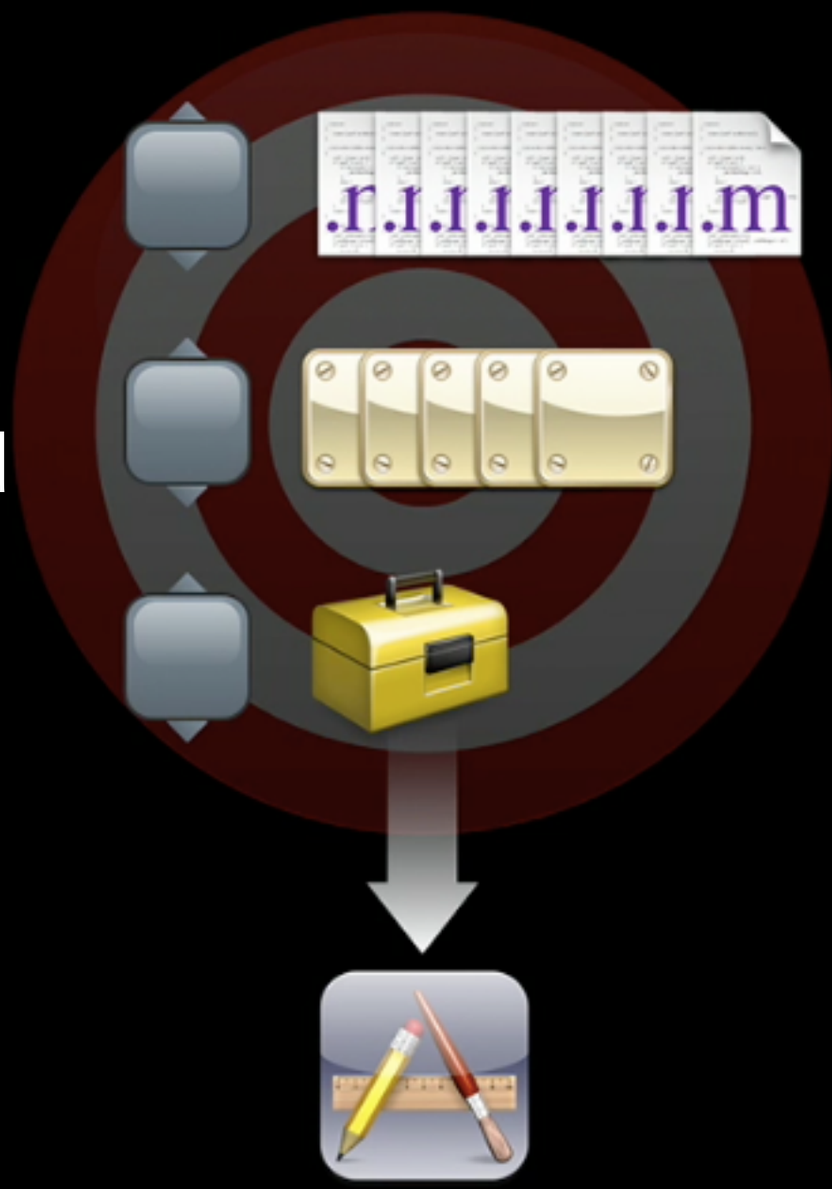
Target — excutable

- 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

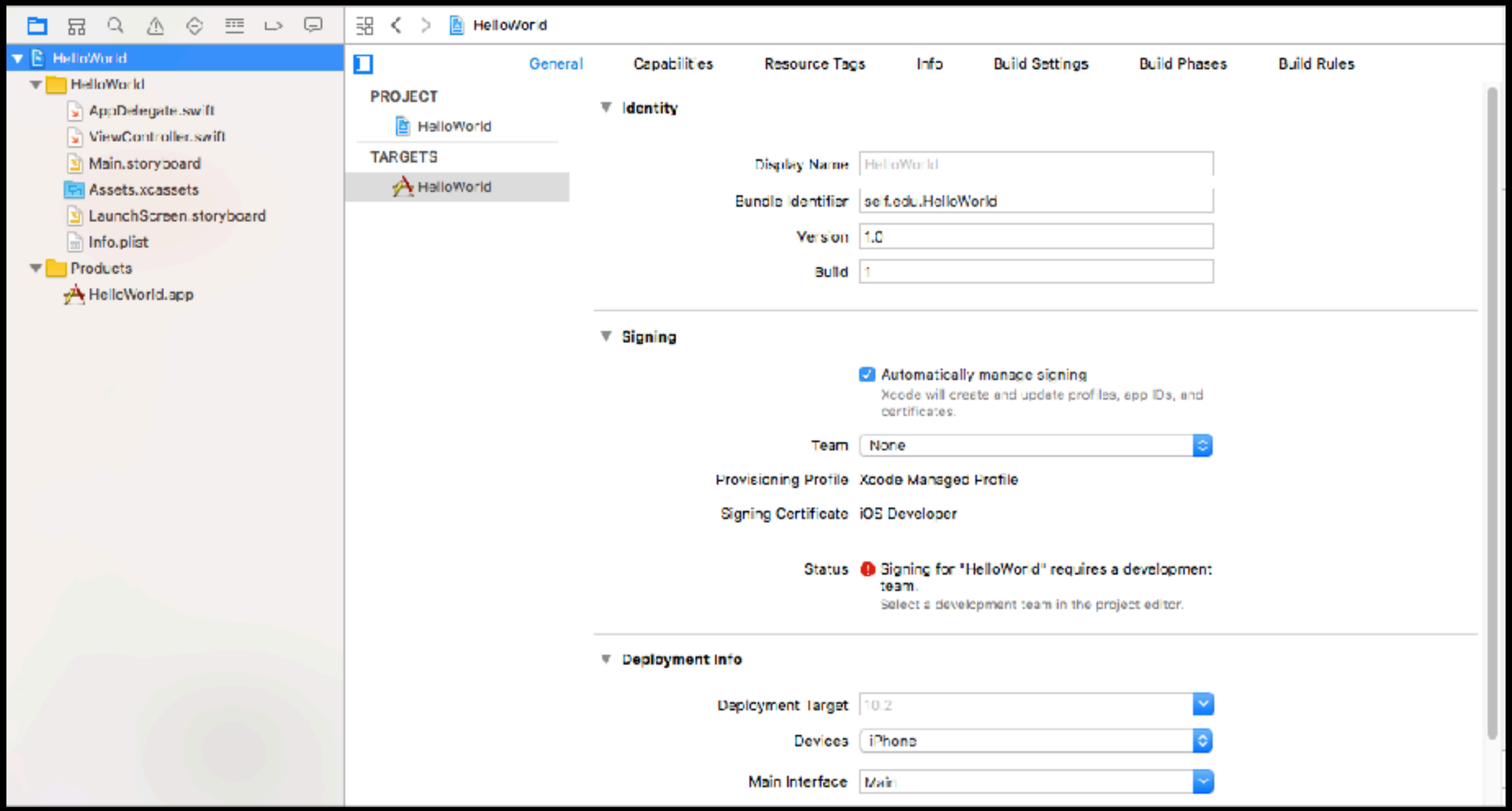


Target — excutable

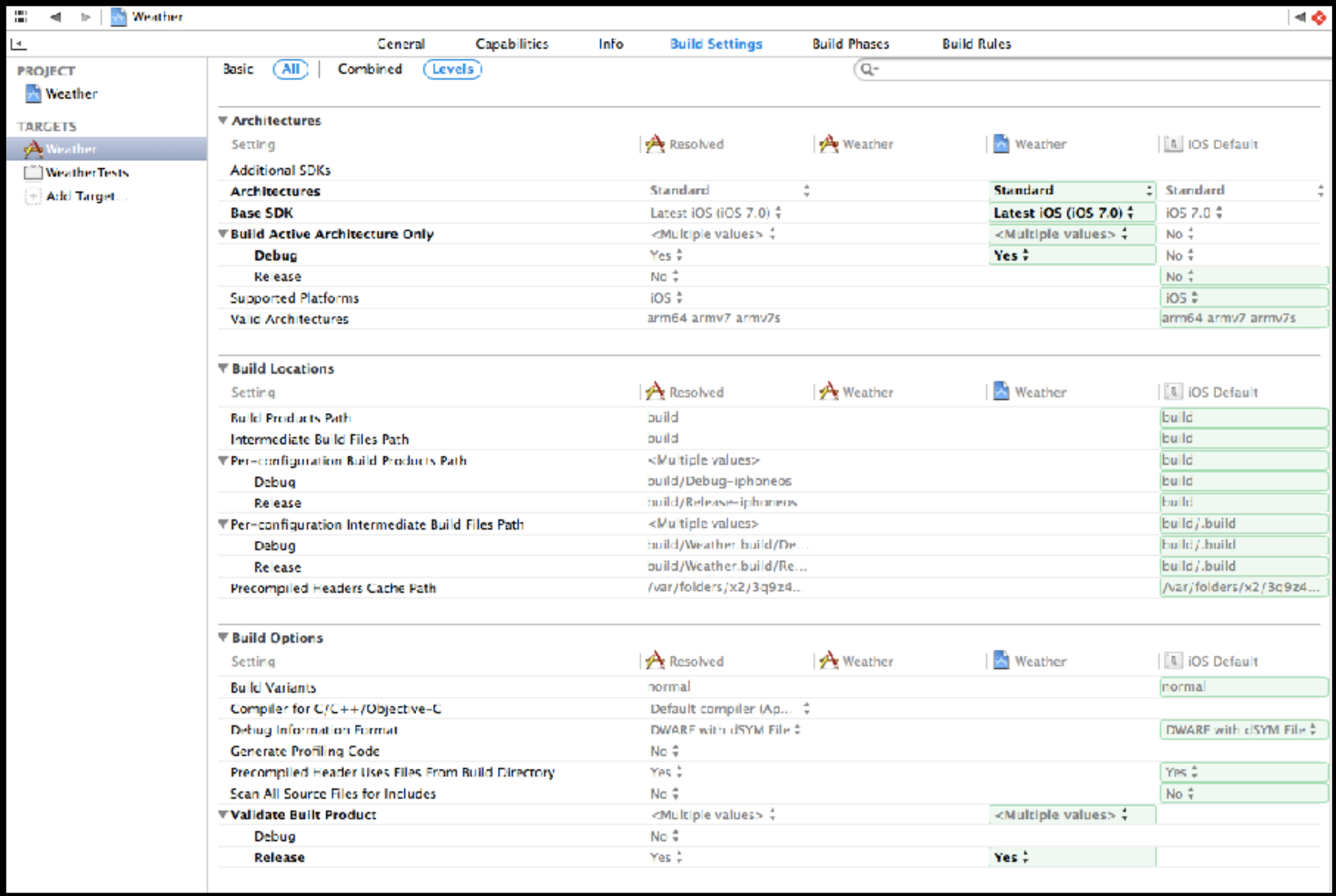
- Some or all source/resource files
- Build phases - the high-level sequence of steps
- Build settings - how the build is done
- Build rules - how to handle each file/file type
- Can depend on one or more other targets - automatically built



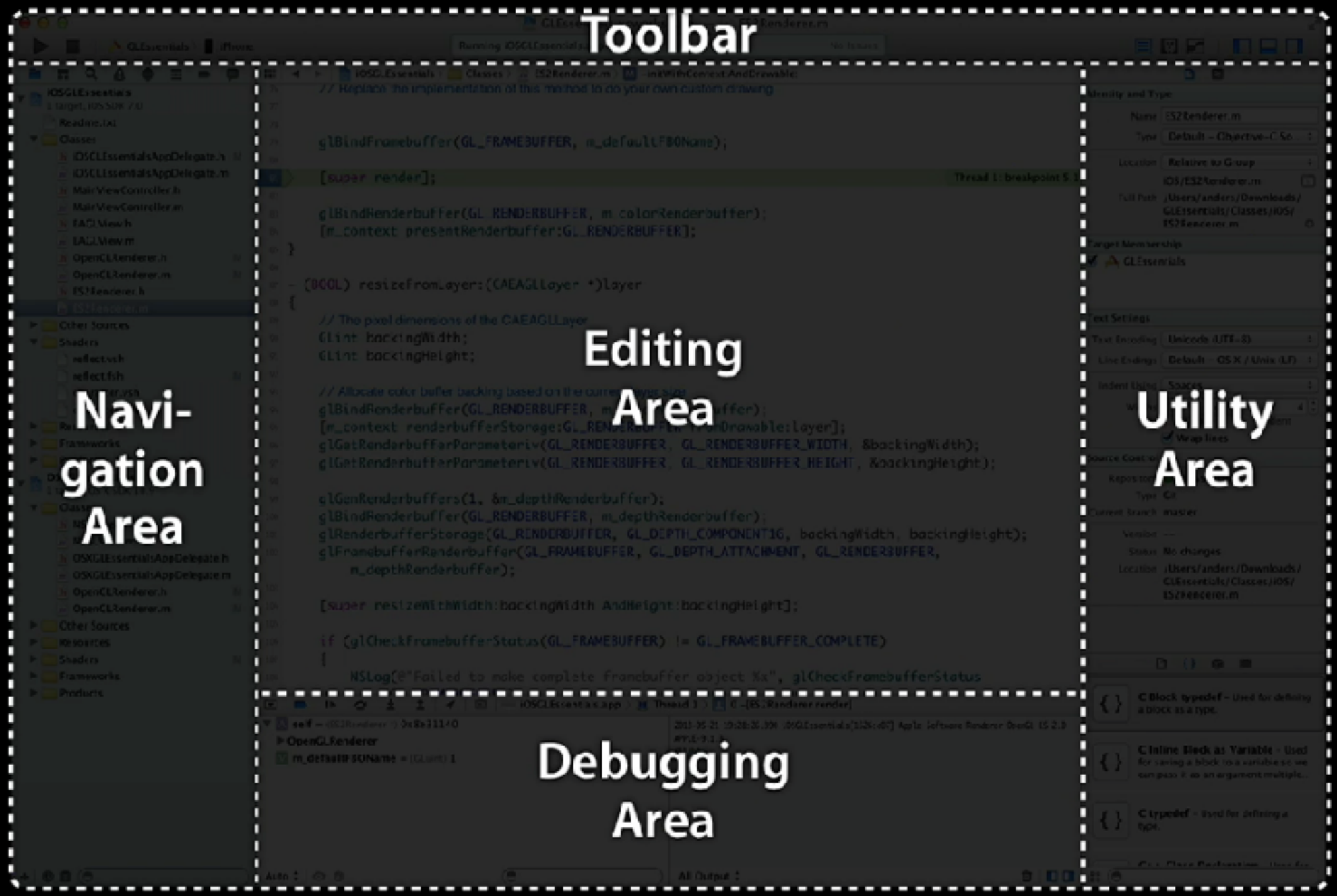
The Target Setting



Build Settings Window

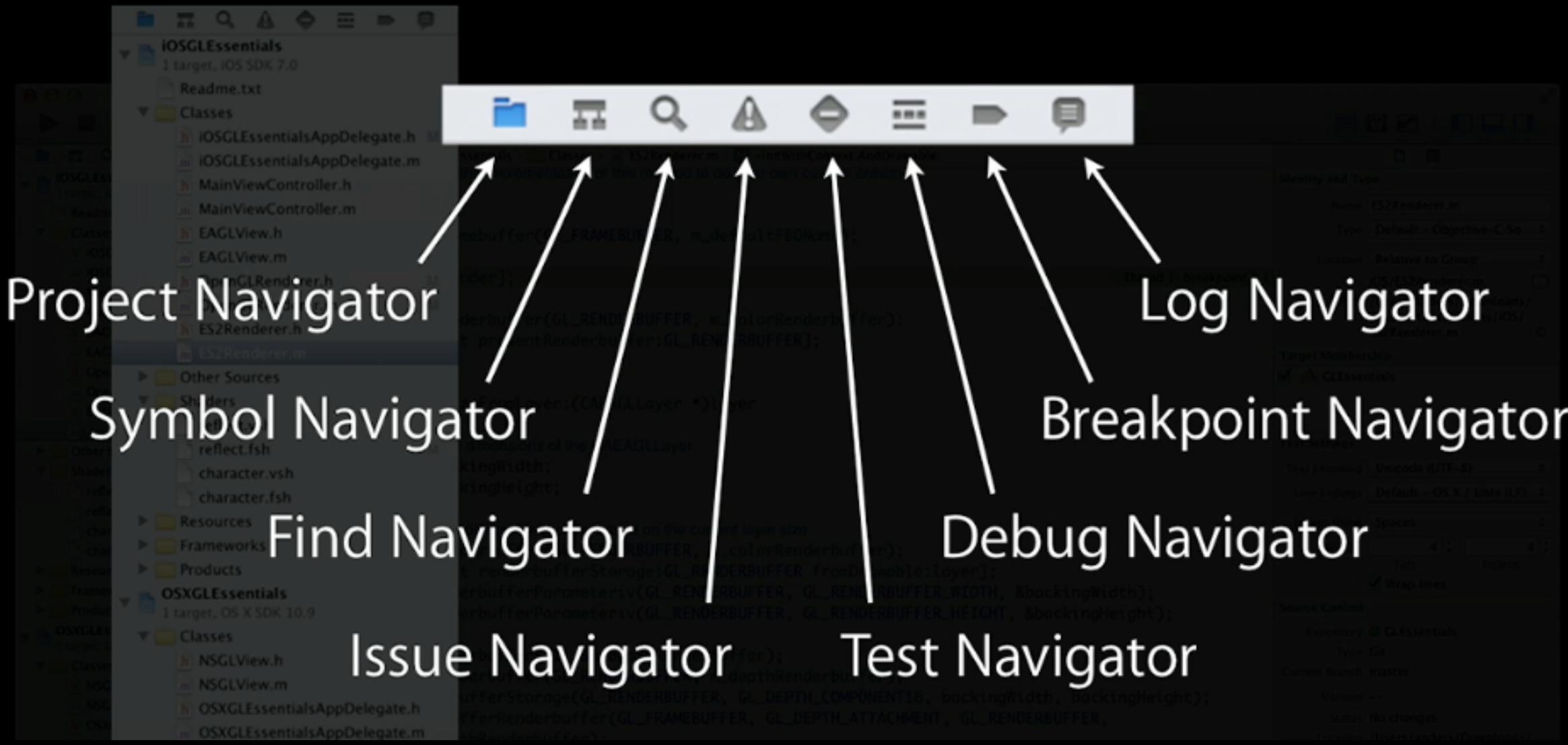


The Xcode Layout

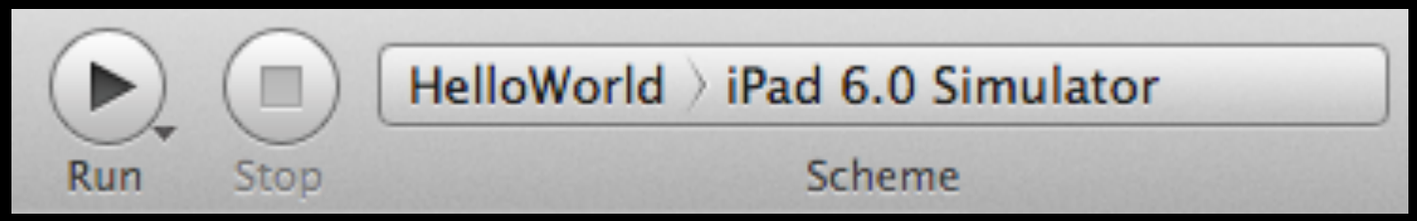


The “Layout Selector”

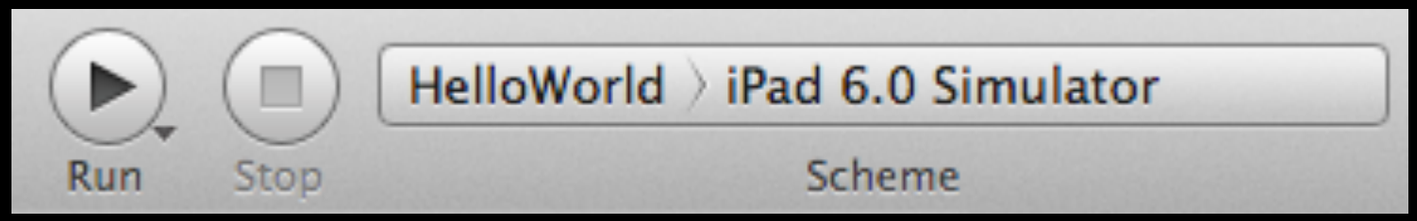
The Navigation Selector



Build and Run

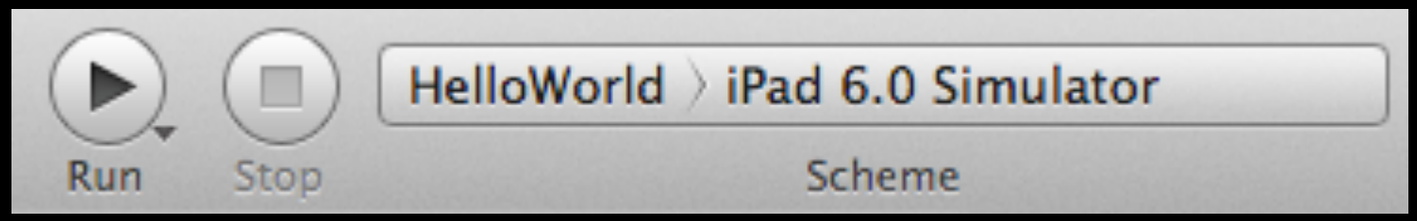


Build and Run



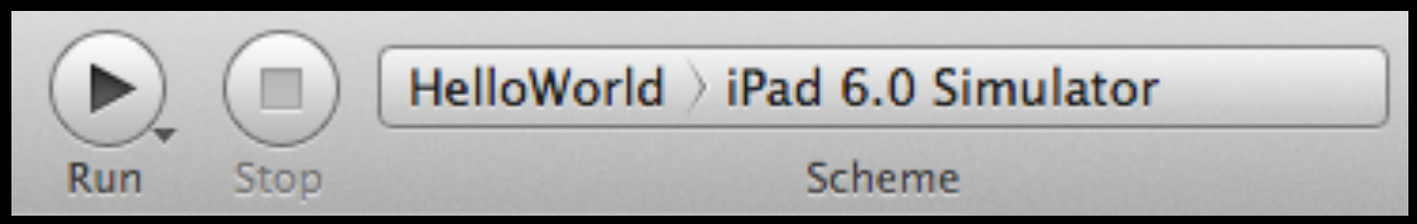
- Build a project - ⌘B

Build and Run



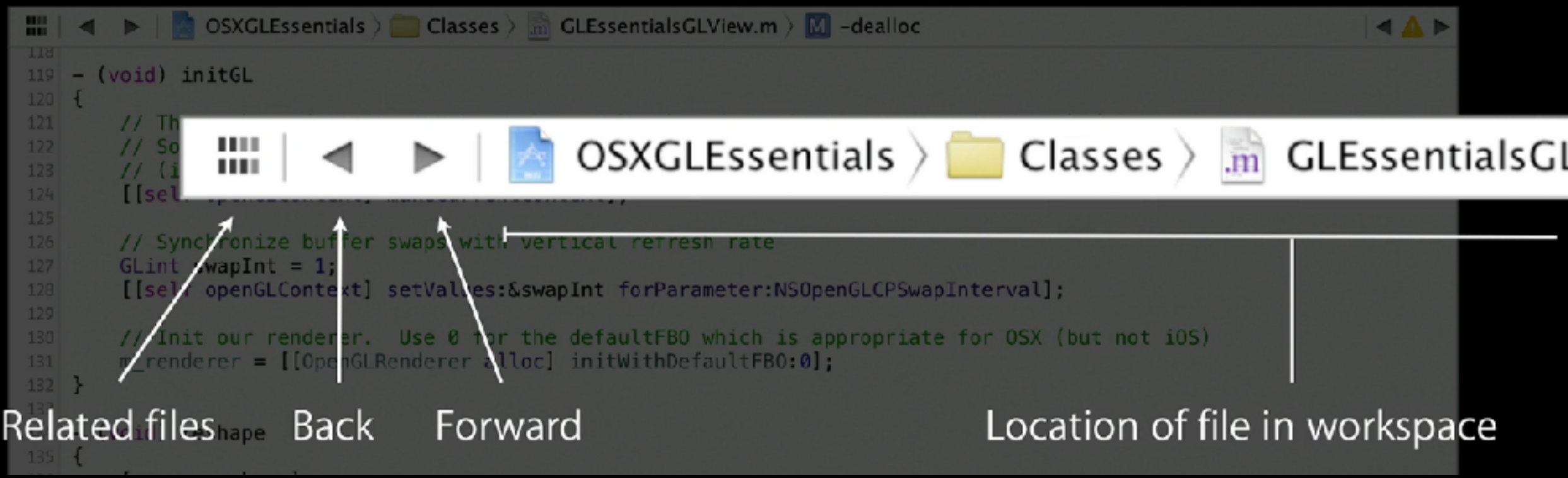
- Build a project - ⌘B
- Run a project - ⌘R

Build and Run

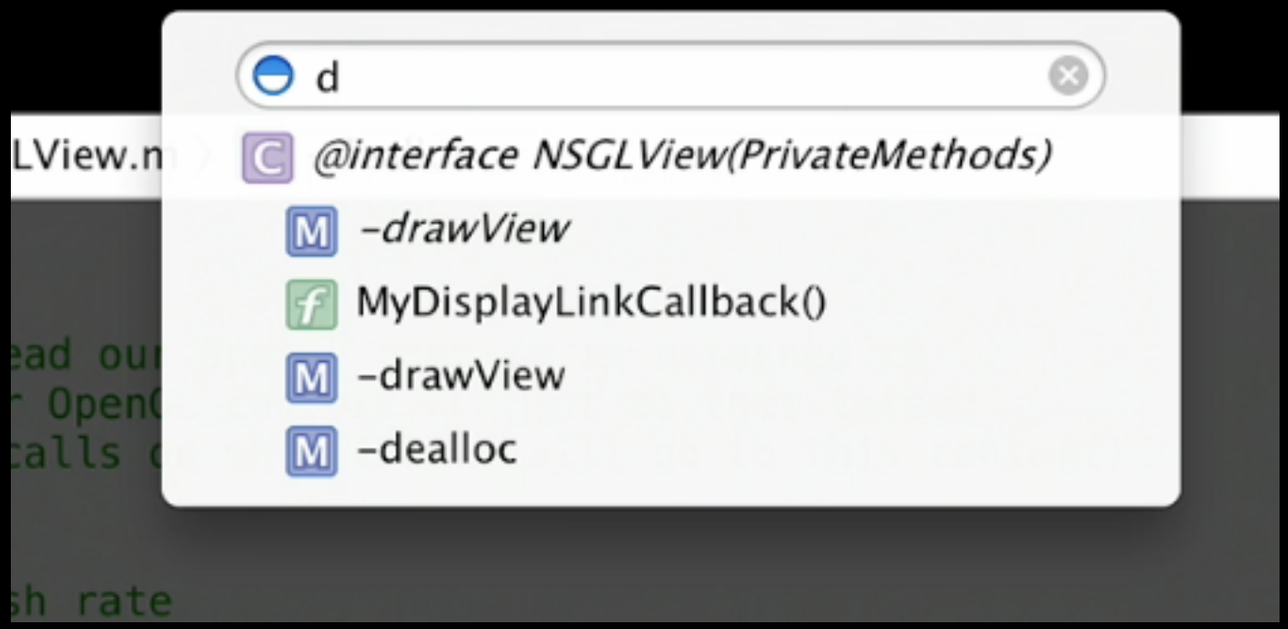


- Build a project - ⌘B
- Run a project - ⌘R
- Select a testing device

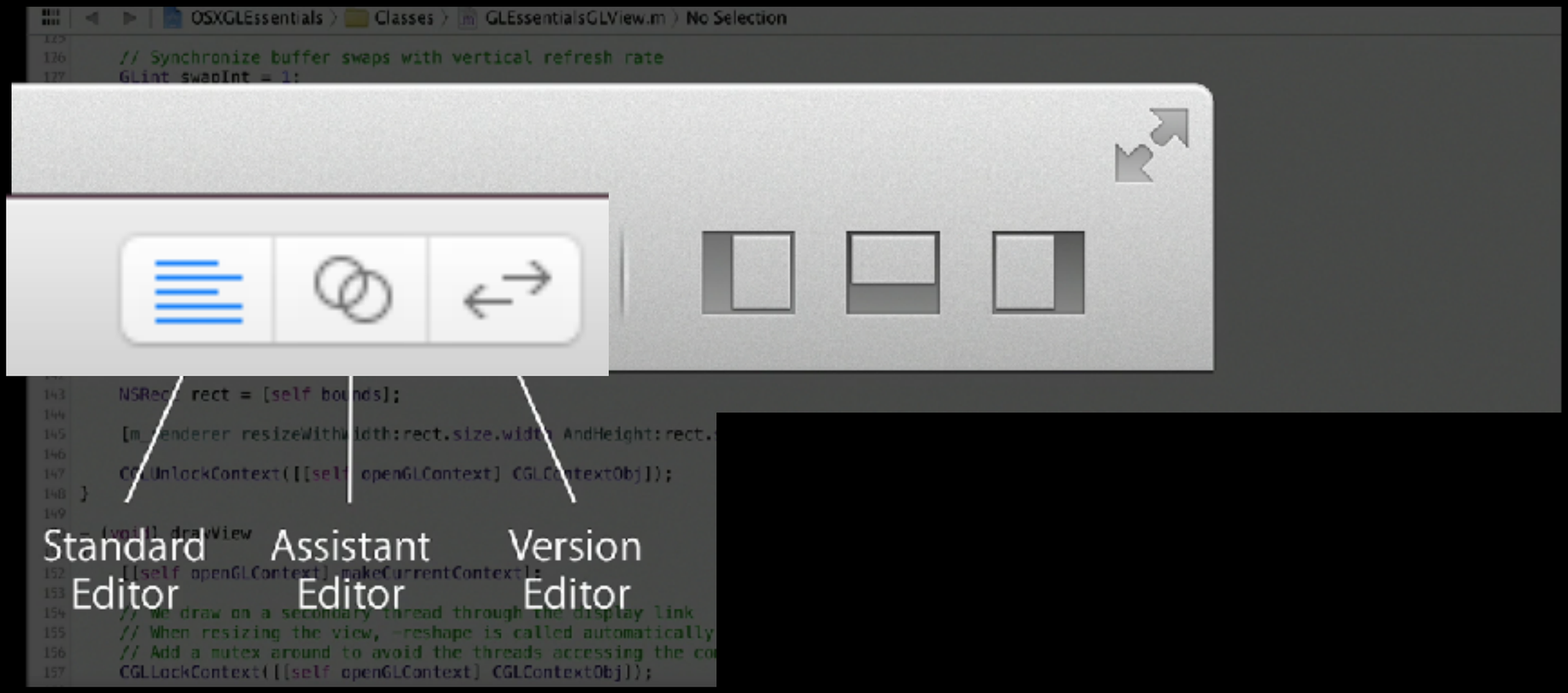
Editor Jump Bar



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

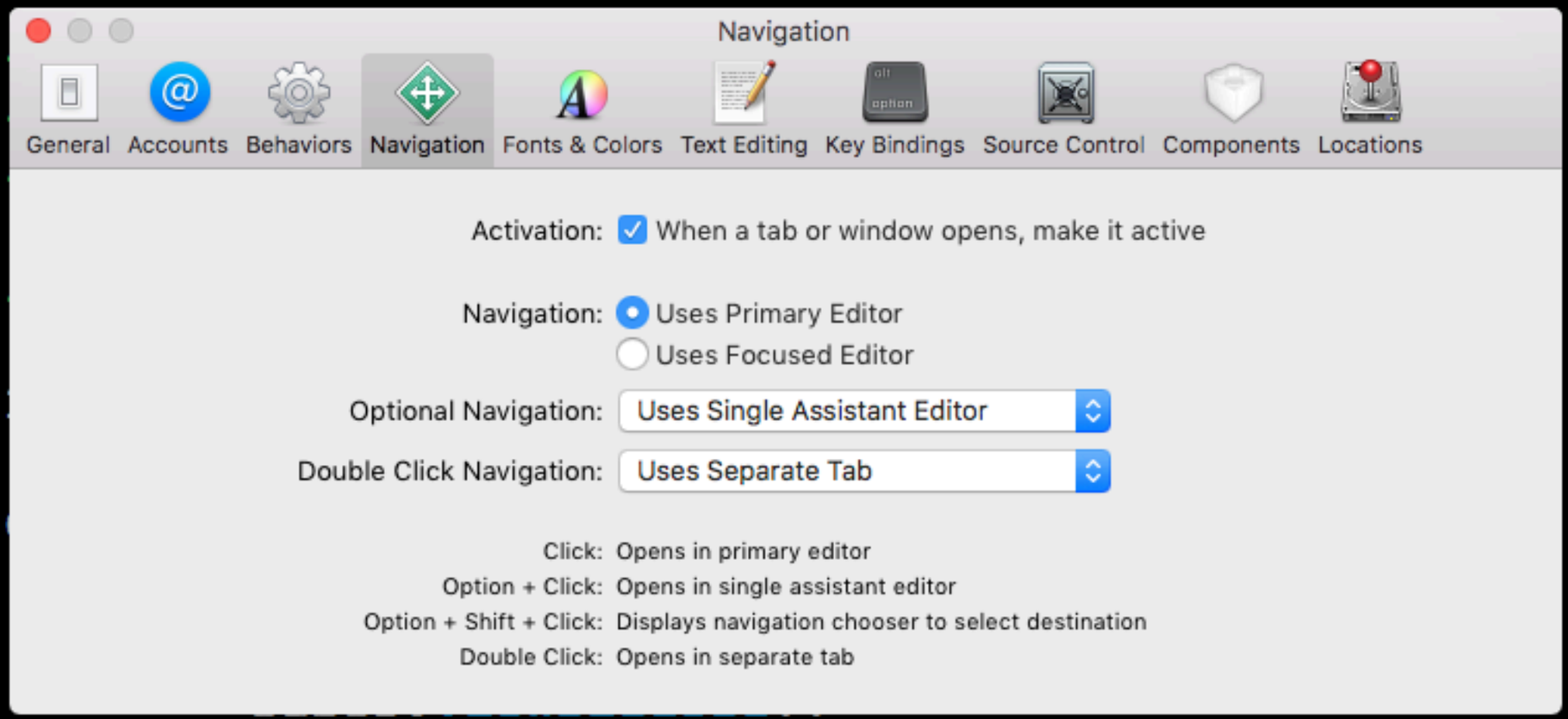
Assistant Editor

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

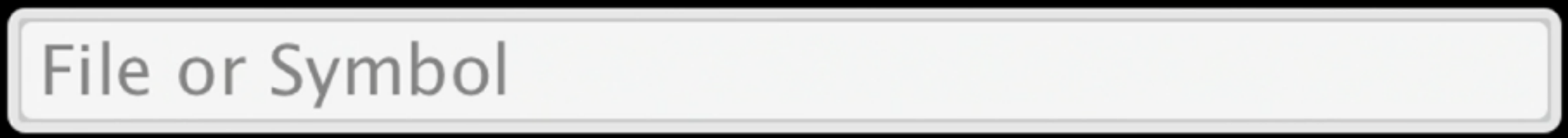
Assistant Editor

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

Modify Navigation Behavior



Open Quickly



Xcode Preferences

Xcode Preferences

- Change fonts

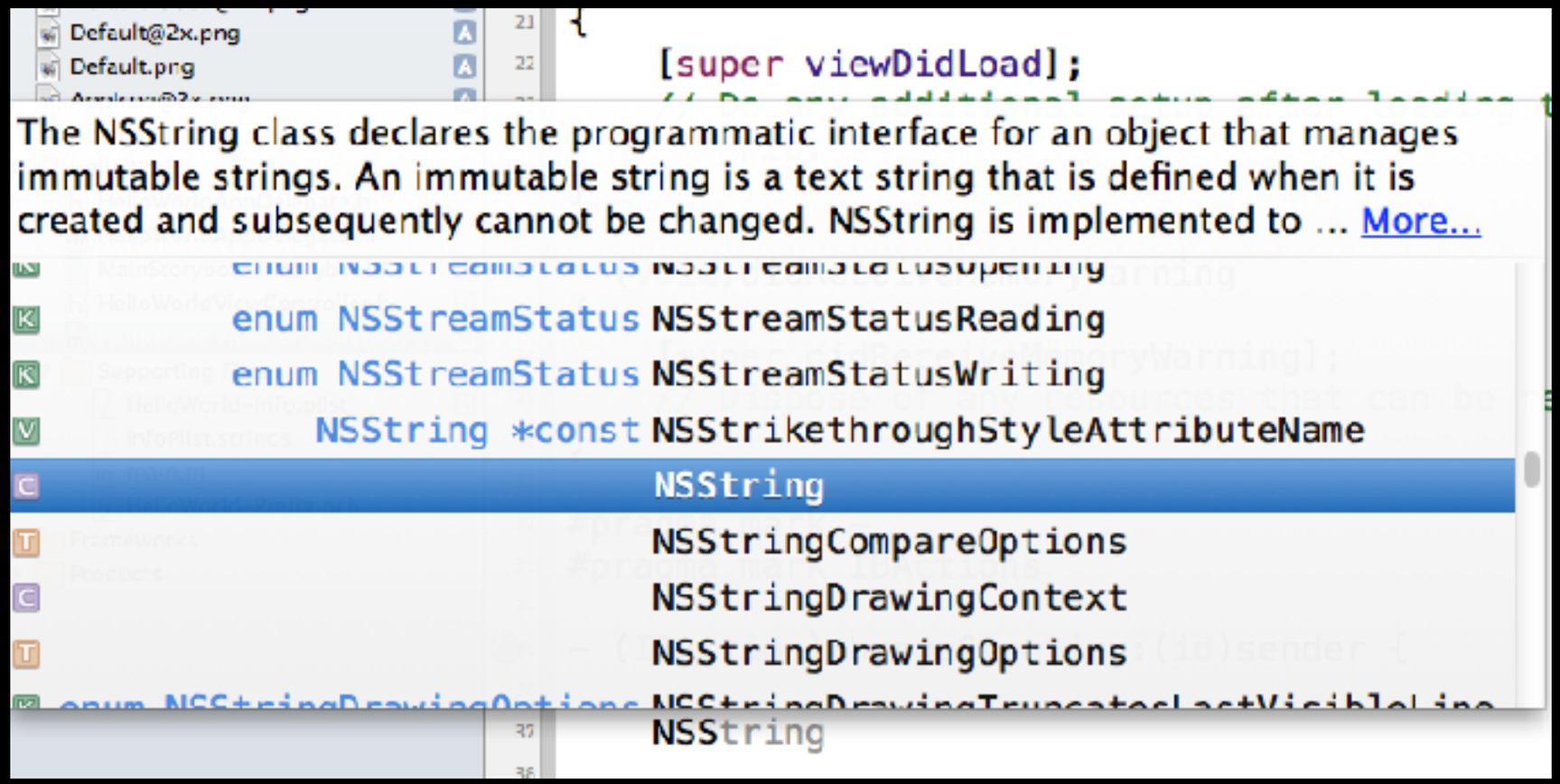
Xcode Preferences

- Change fonts
- Download docs and tools

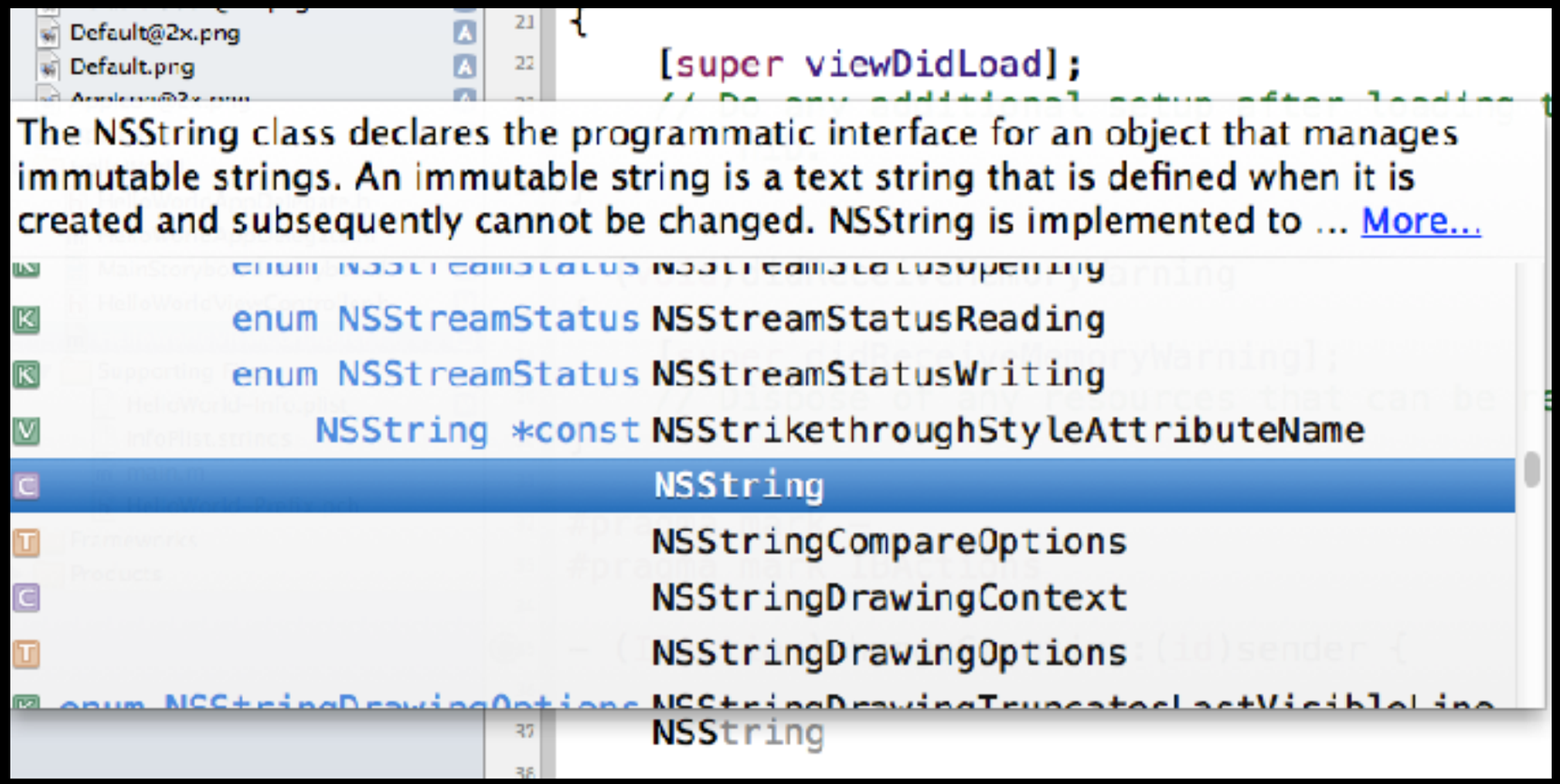
Xcode Preferences

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

Auto-completion

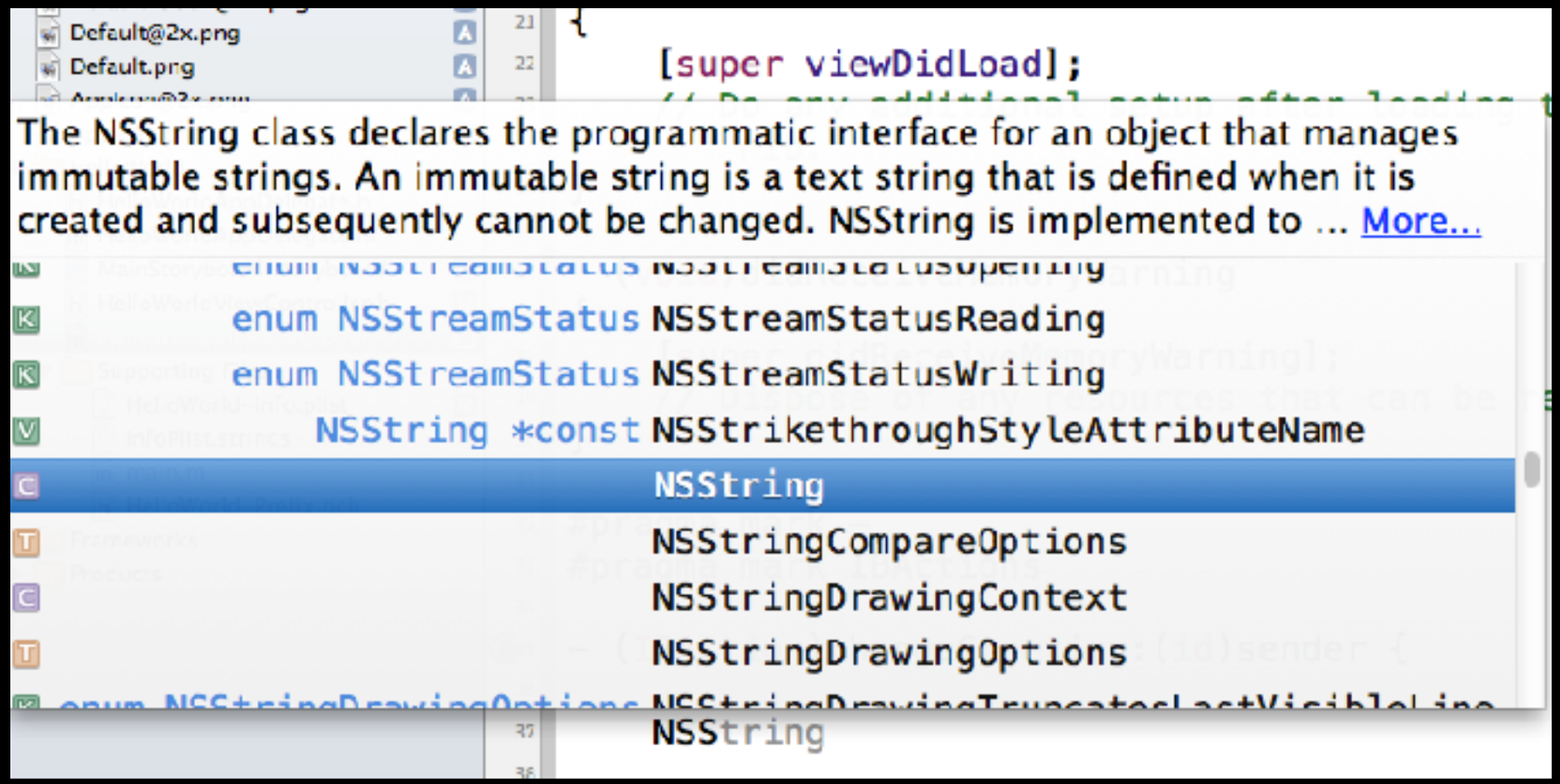


Auto-completion



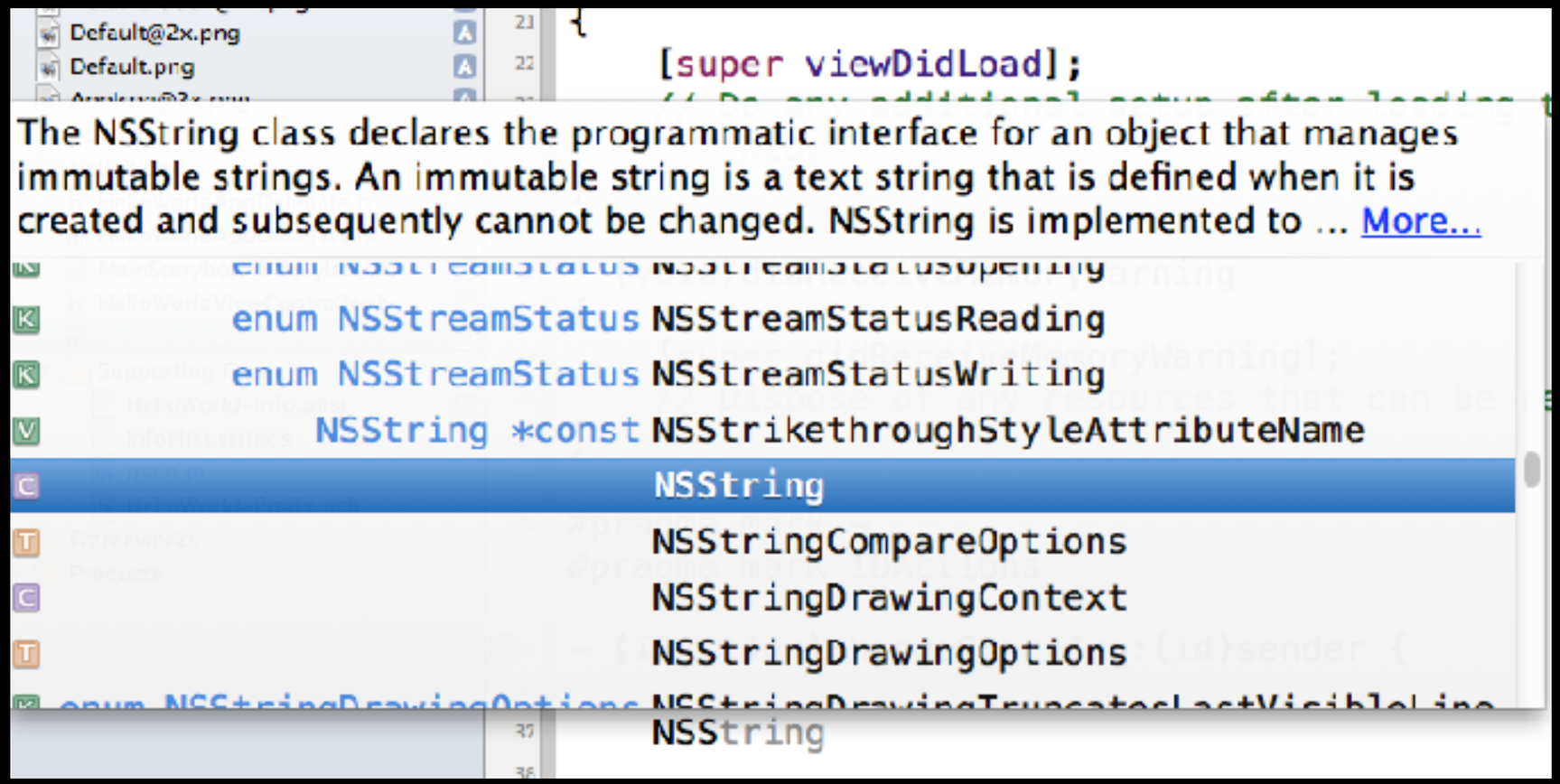
- tab or return to accept auto-completion

Auto-completion



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

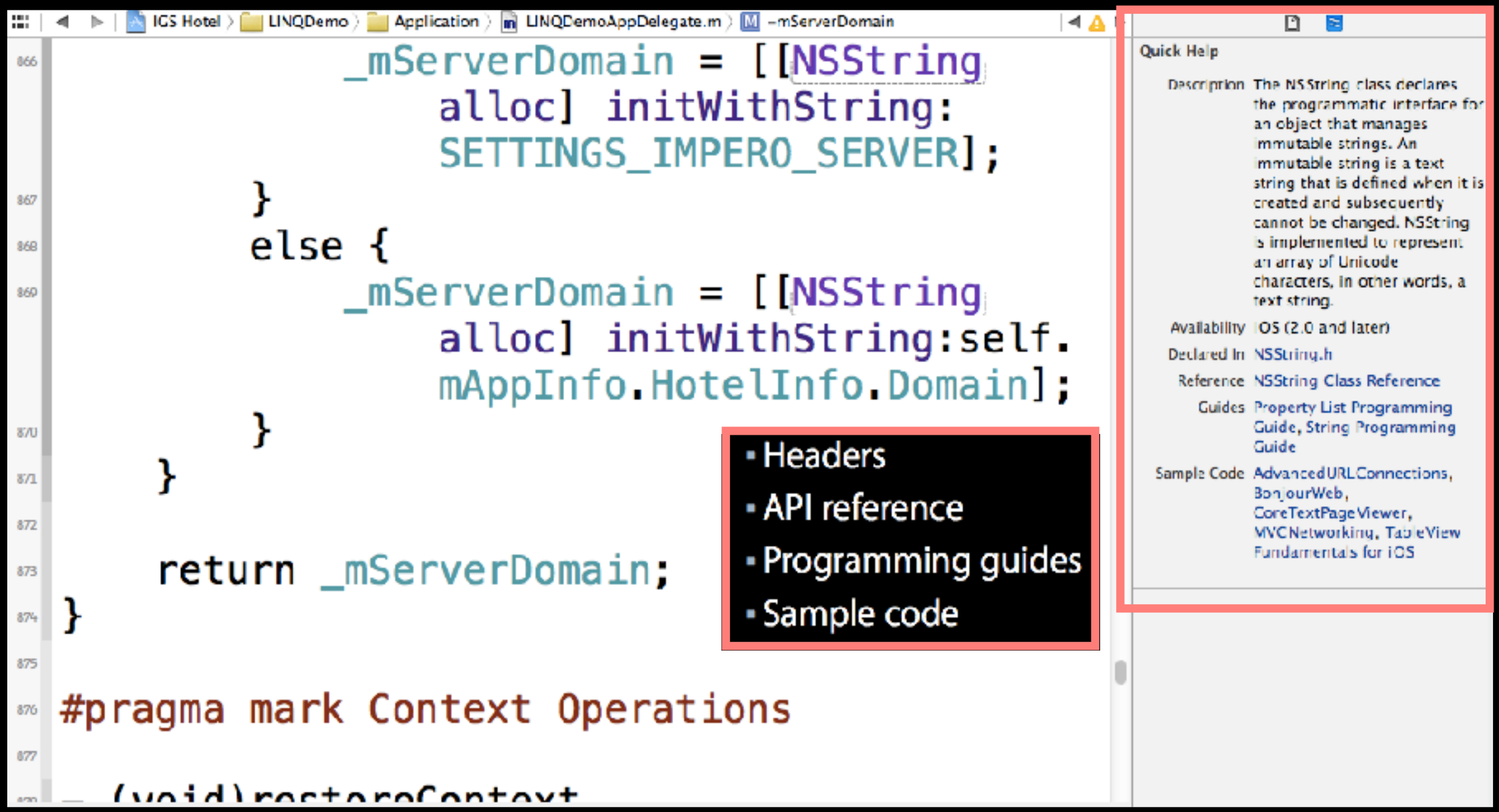
Auto-completion



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

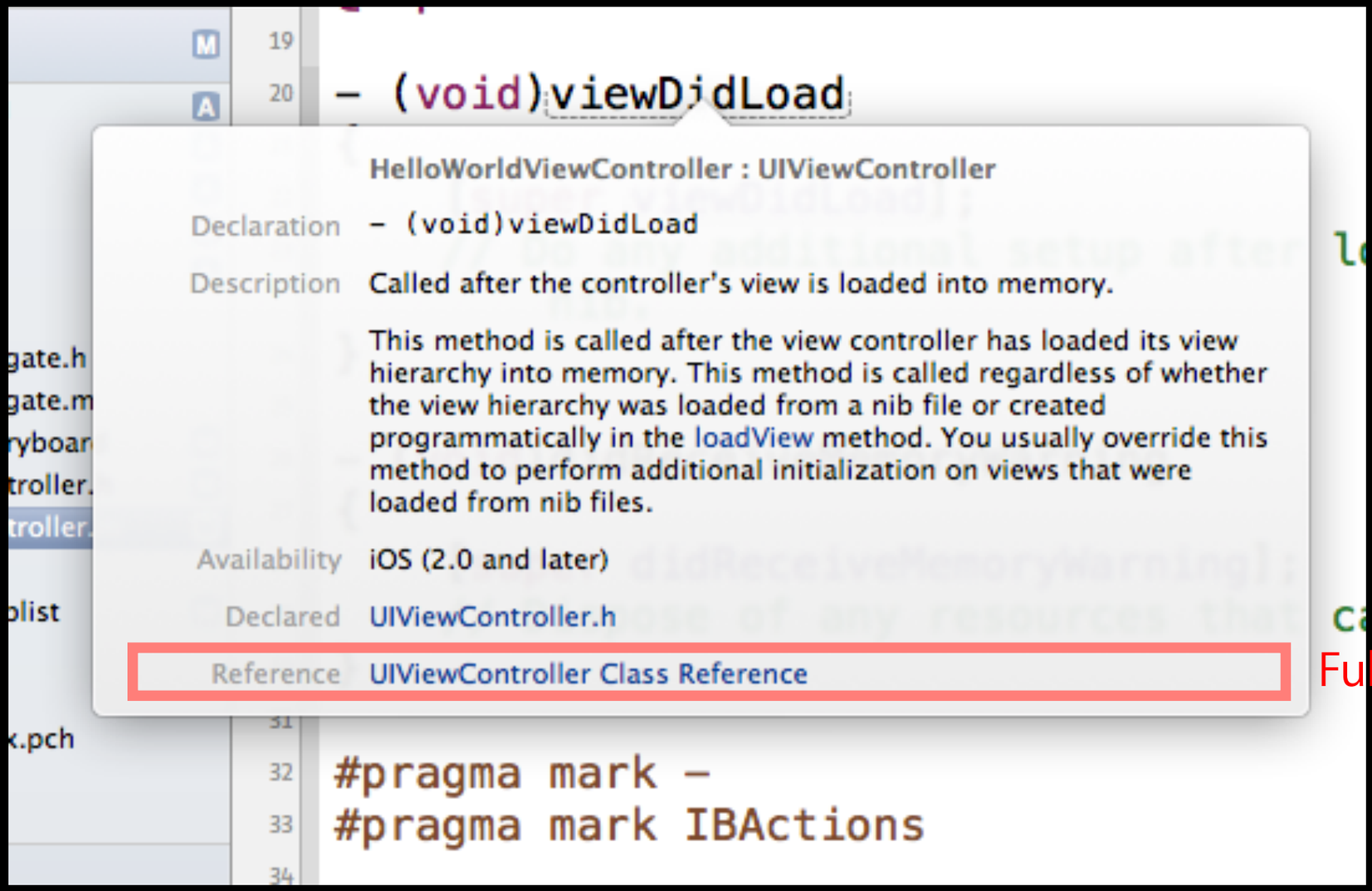
Quick Help Inspector

Click any Symbol



In-Place Quick Help

Option + click any Symbol



Command + click : go to method definition

Documentation Window

Find/Replace Texts in a File

Find/Replace Texts in a File

- ⌘F - open the “in-file find” bar

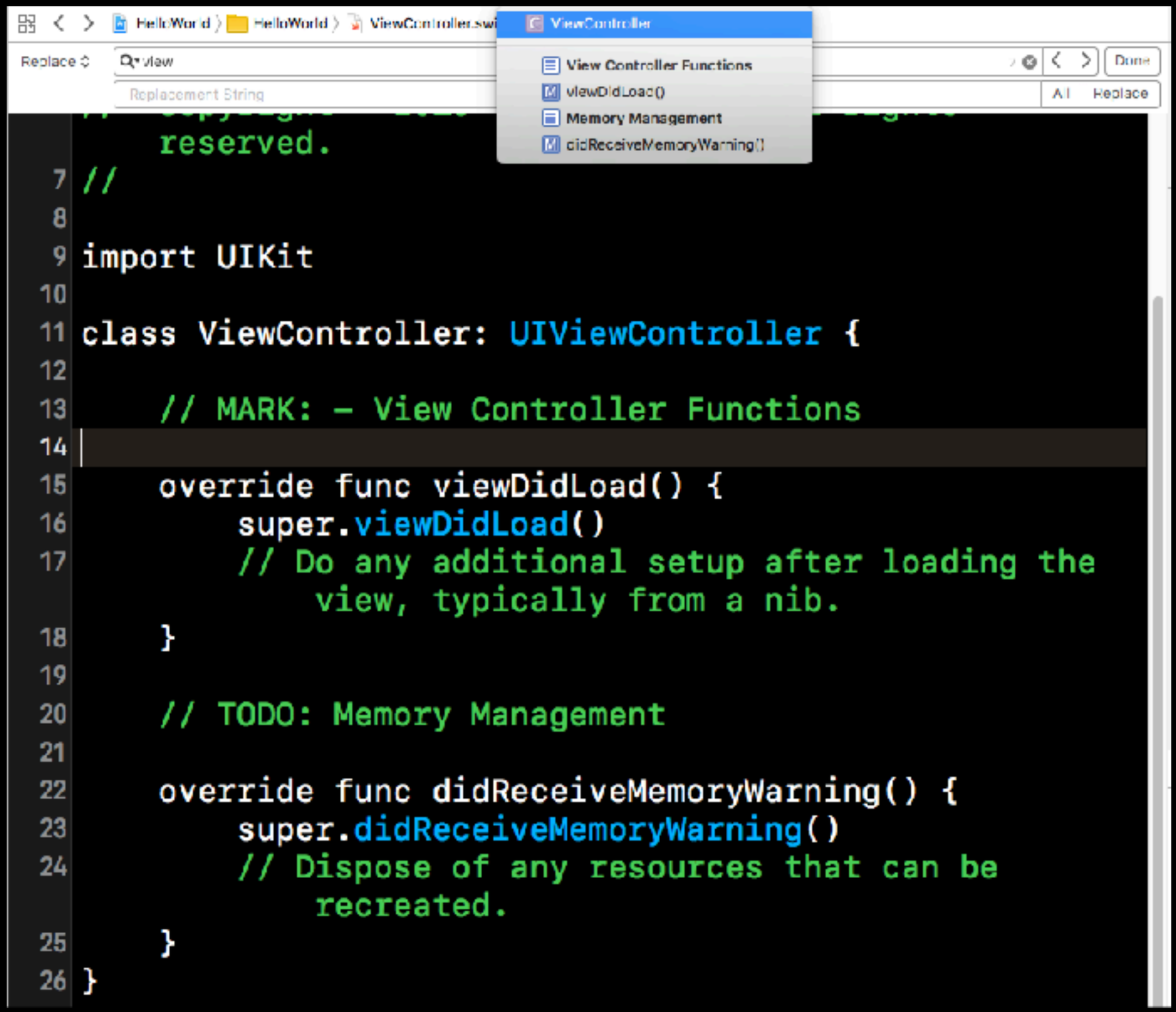
Find/Replace Texts in a File

- ⌘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



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

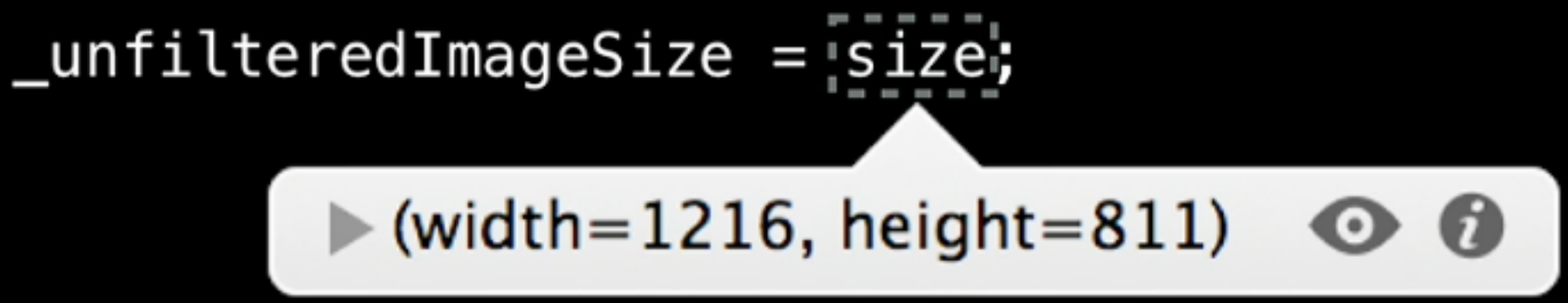
Debugging in Xcode

- 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

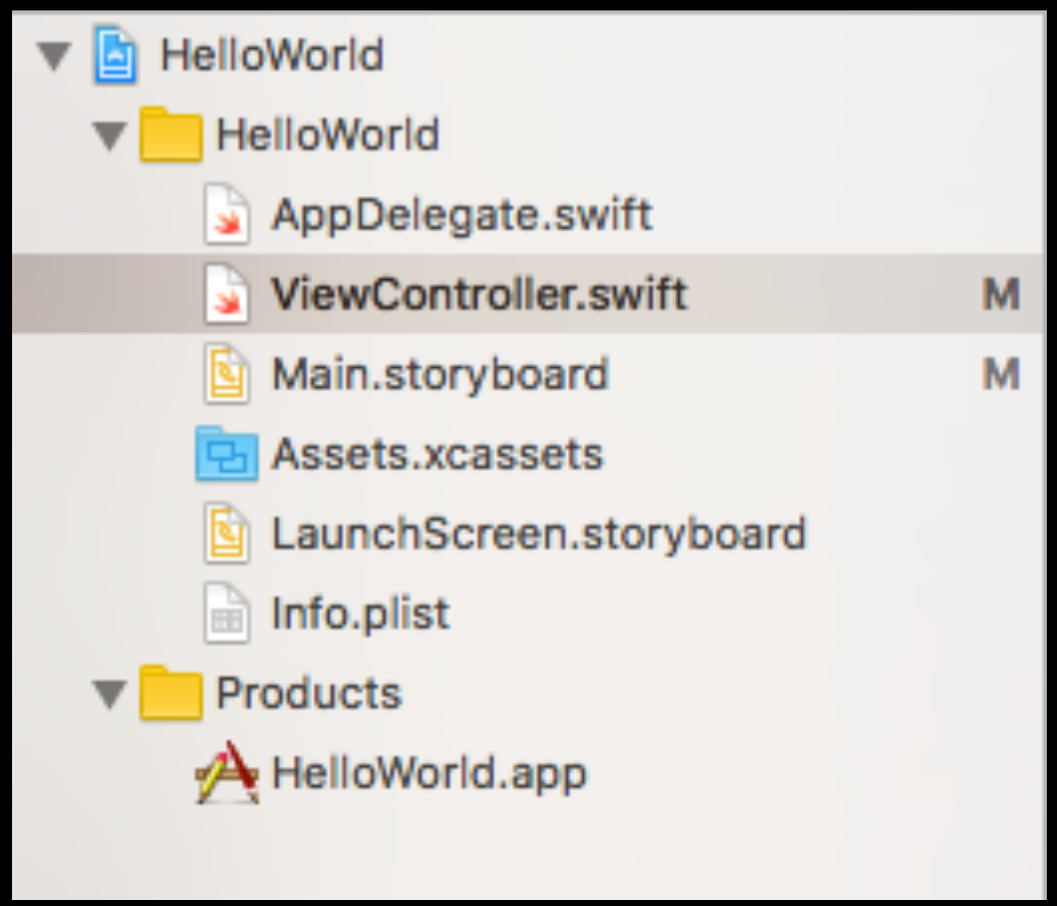


LLDB

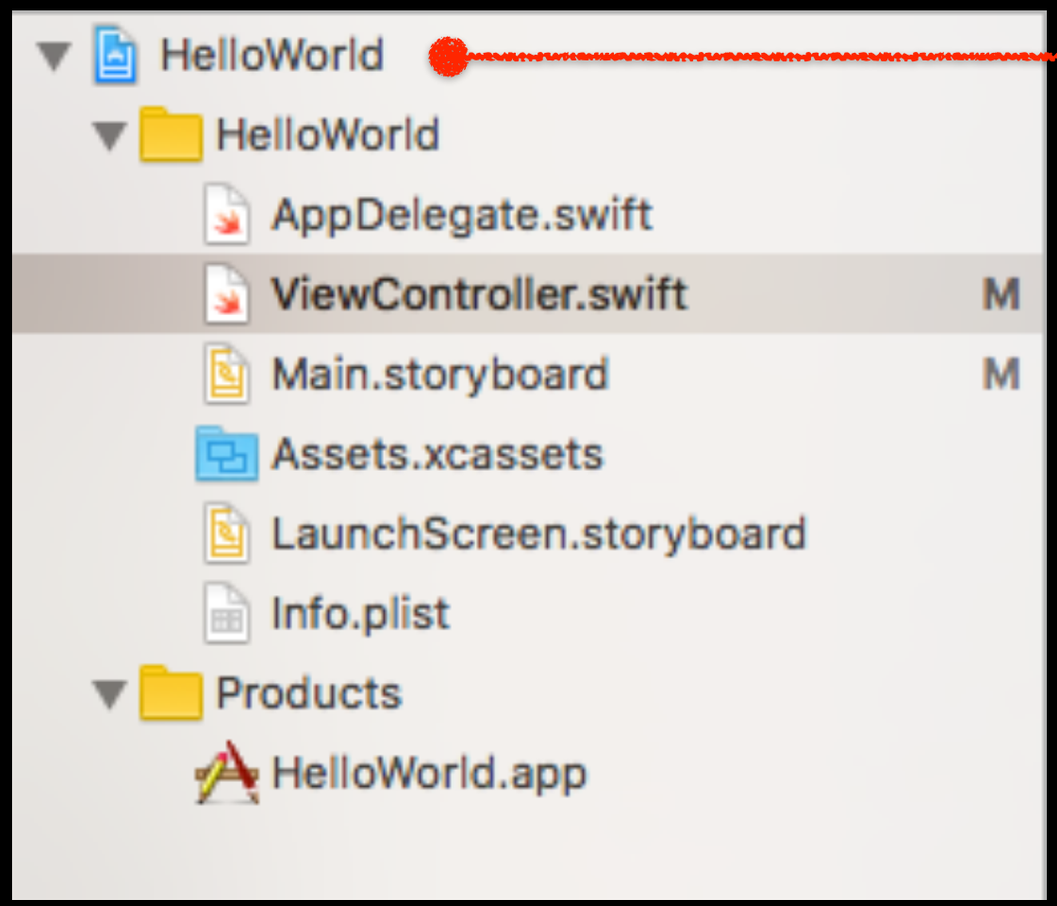
```
(lldb) po self.nameTextField
(UITextField *) $1 = 0x07176100 <UITextField: 0x7176100; frame =
(20 20; 280 30); text = 'Patrick'; clipsToBounds = YES; opaque =
NO; autoresize = TM+BM; gestureRecognizers = <NSArray: 0x717c740>;
layer = <CALayer: 0x71762a0>>
(lldb) po self.userName
(NSString *) $2 = 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

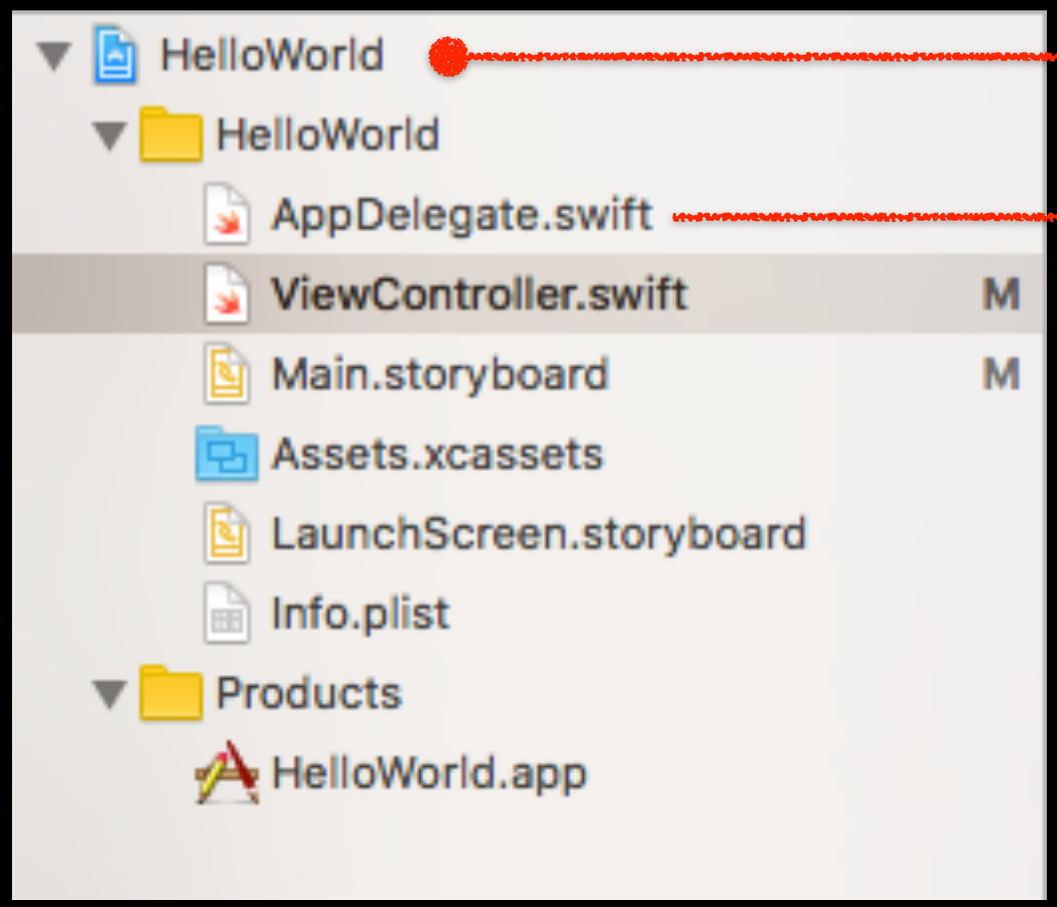


The Anatomy of an iOS Project



Project

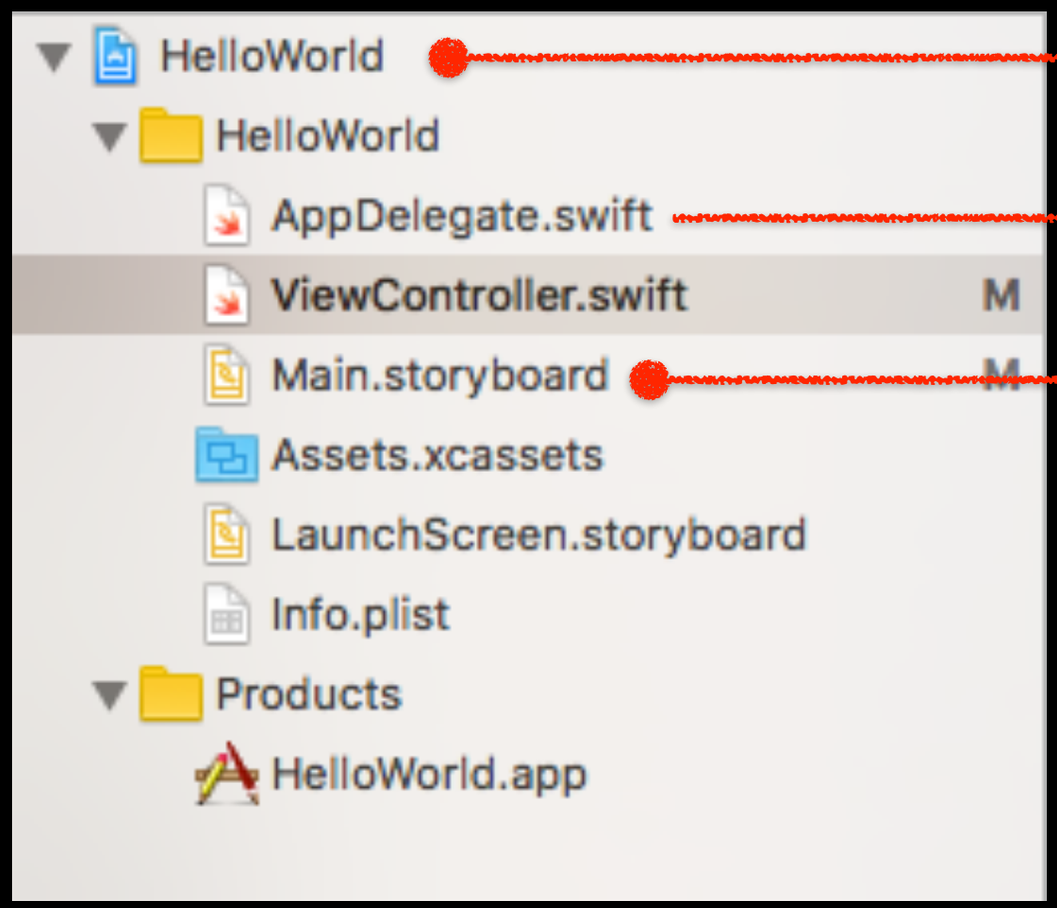
The Anatomy of an iOS Project



Project

The Application

The Anatomy of an iOS Project

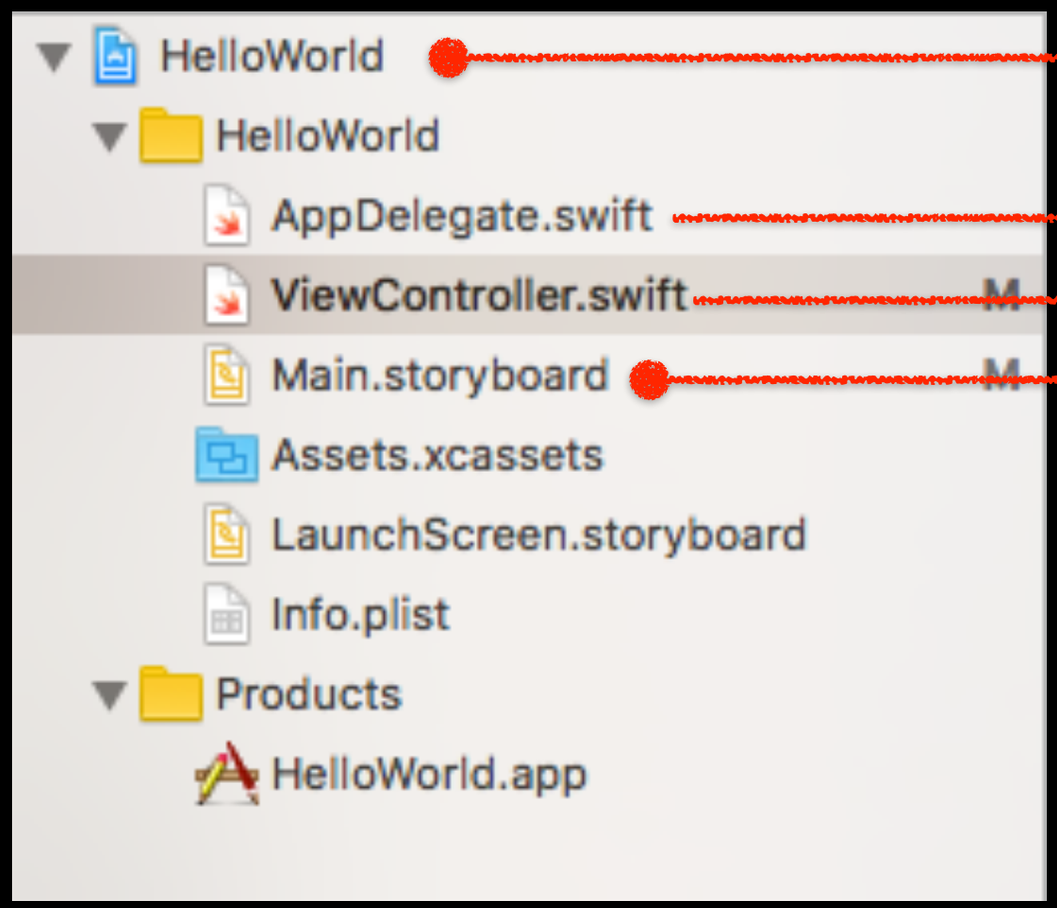


Project

The Application

Views

The Anatomy of an iOS Project



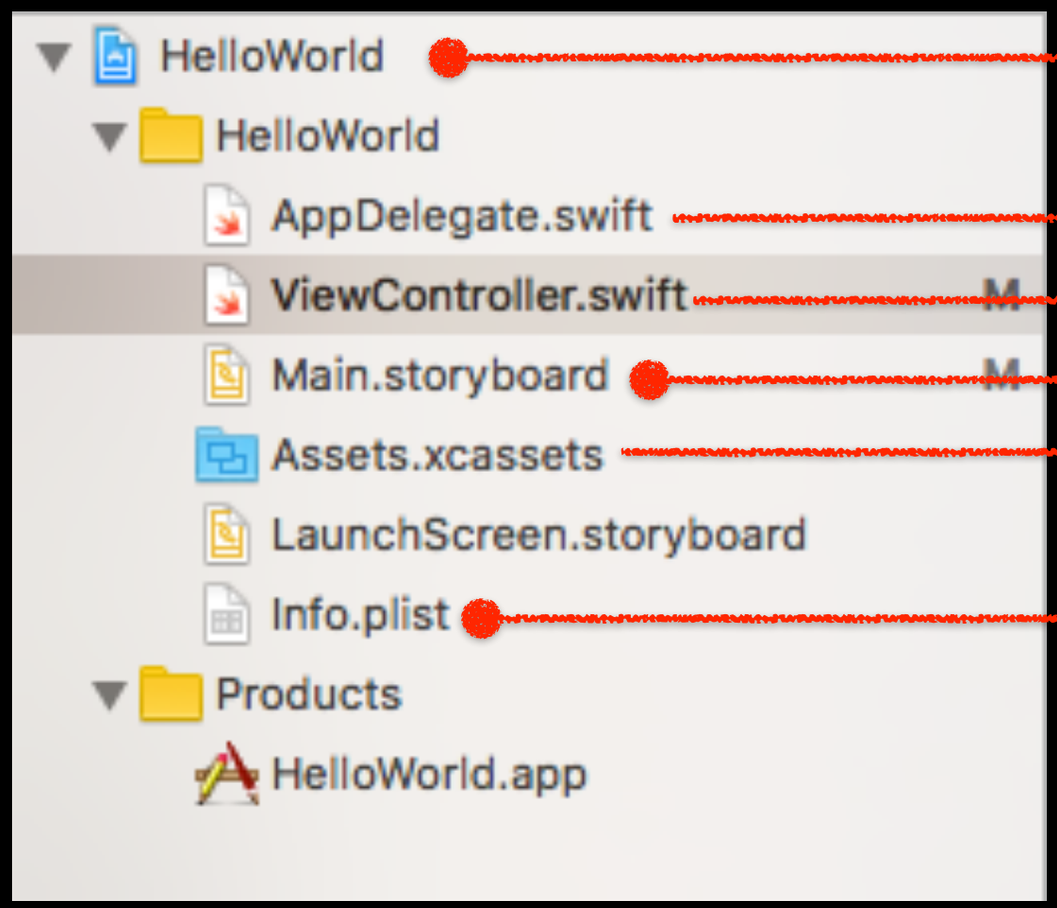
Project

The Application

View Controllers

Views

The Anatomy of an iOS Project



Project

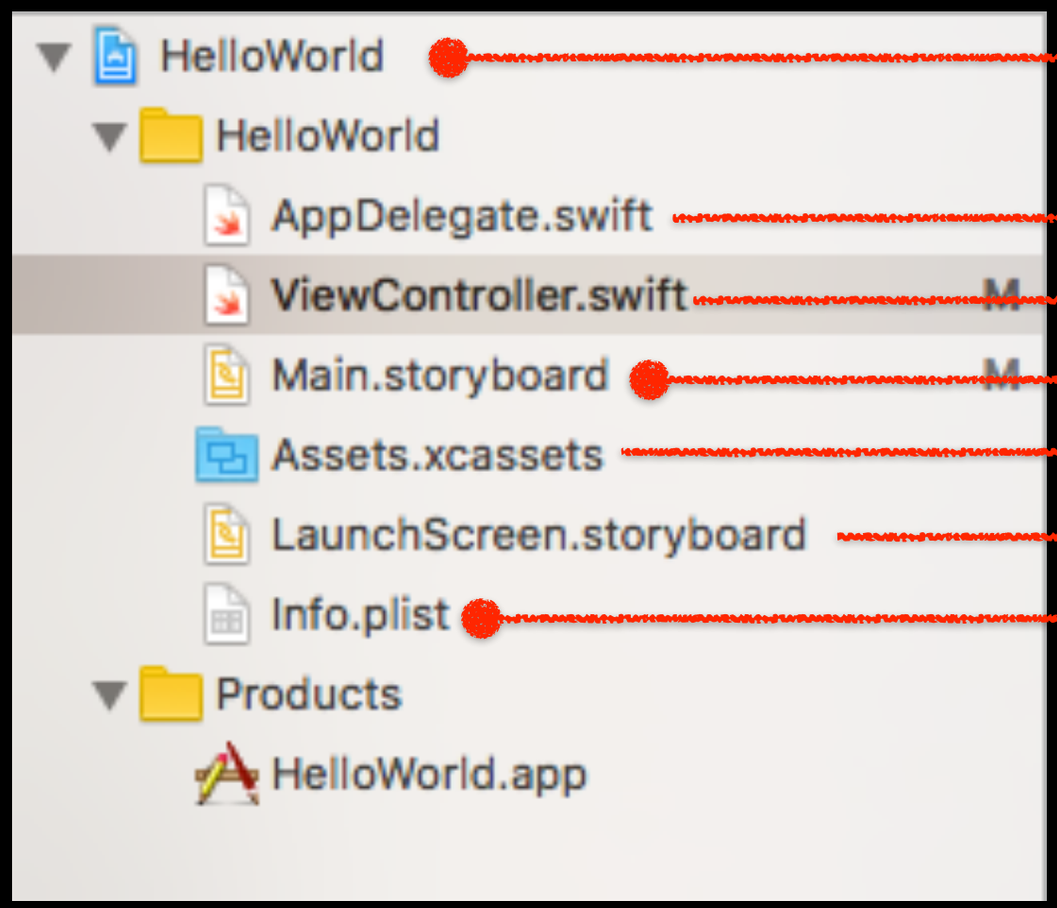
The Application

View Controllers

Views

The Executable

The Anatomy of an iOS Project



Project

The Application

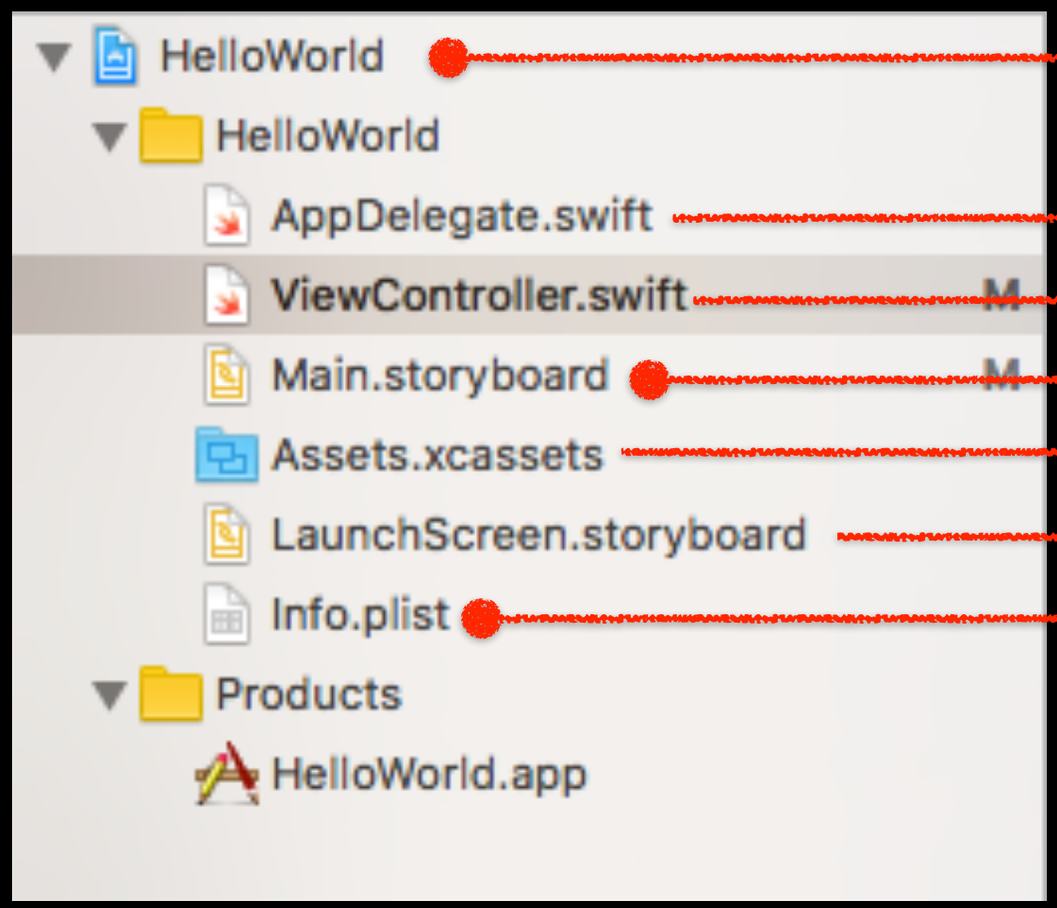
View Controllers

Views

Resources (e.g., images)

The Executable

The Anatomy of an iOS Project



Project

The Application

View Controllers

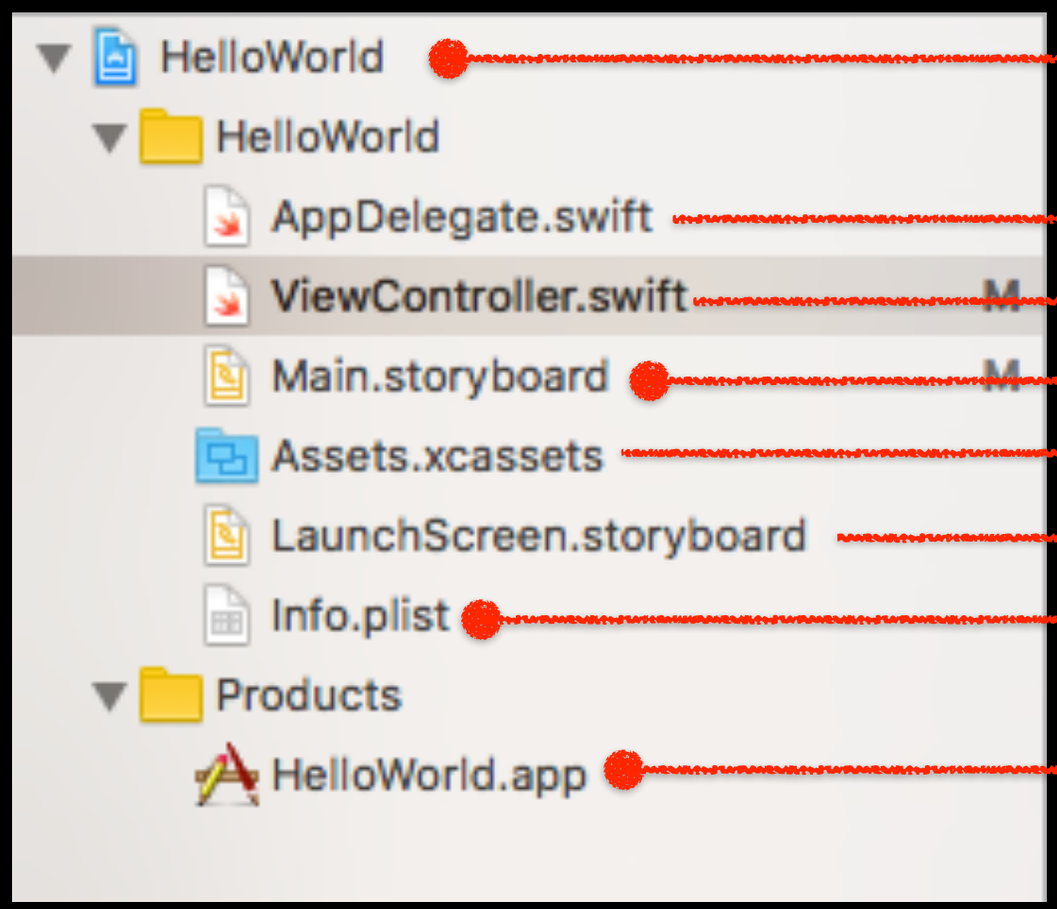
Views

Resources (e.g., images)

Launch Screen

The Executable

The Anatomy of an iOS Project



Project

The Application

View Controllers

Views

Resources (e.g., images)

Launch Screen

App Settings

The Executable

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

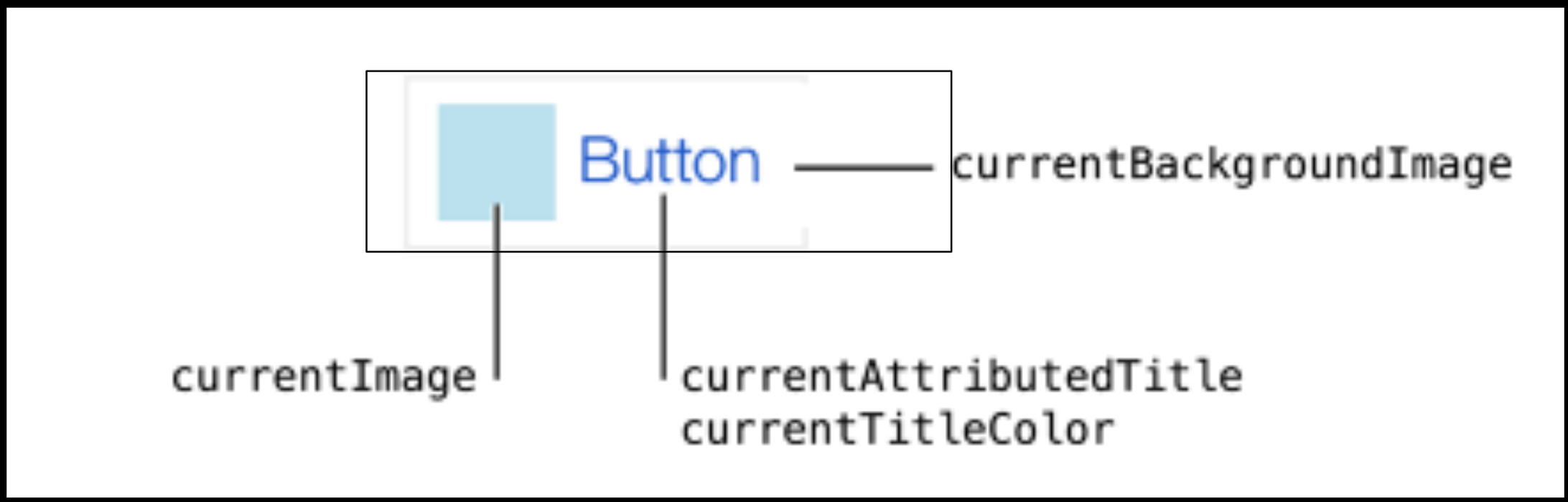
- When the app is ready to use, iOS replaces the launch image with your app's first screen

- The launch image should look very similar to the first screen of your app, but without the app-specific info

UIKit

UIButton

Appearance



UIButton

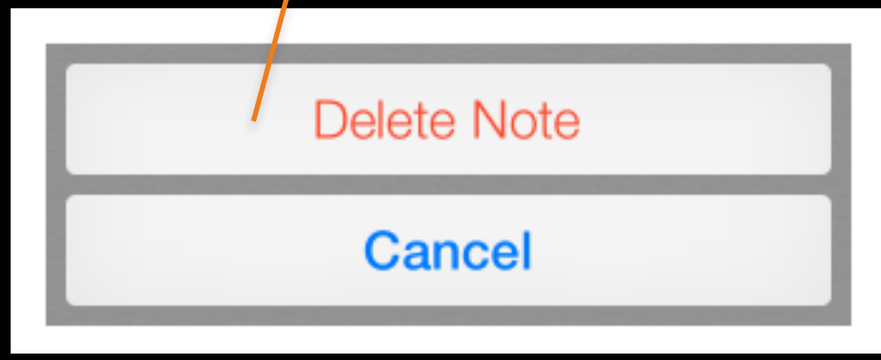
State

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

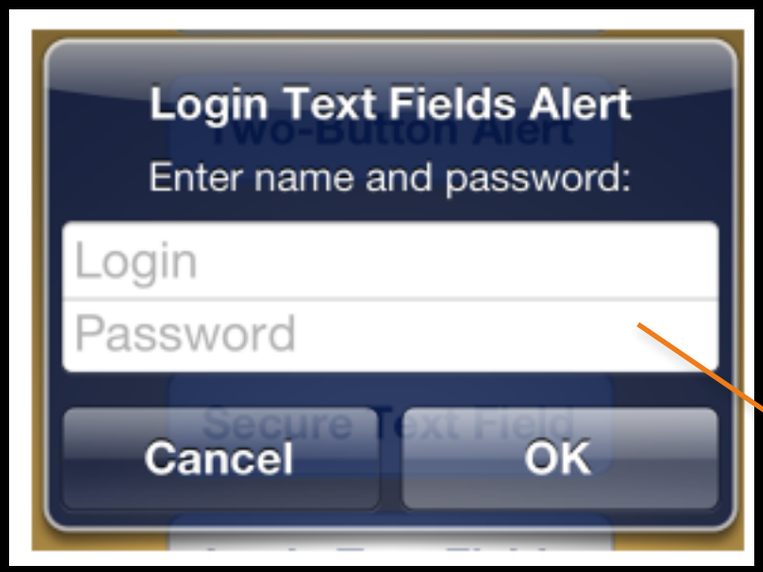
UIAlertcontroller

Style

UIAlertAction



.actionsheet



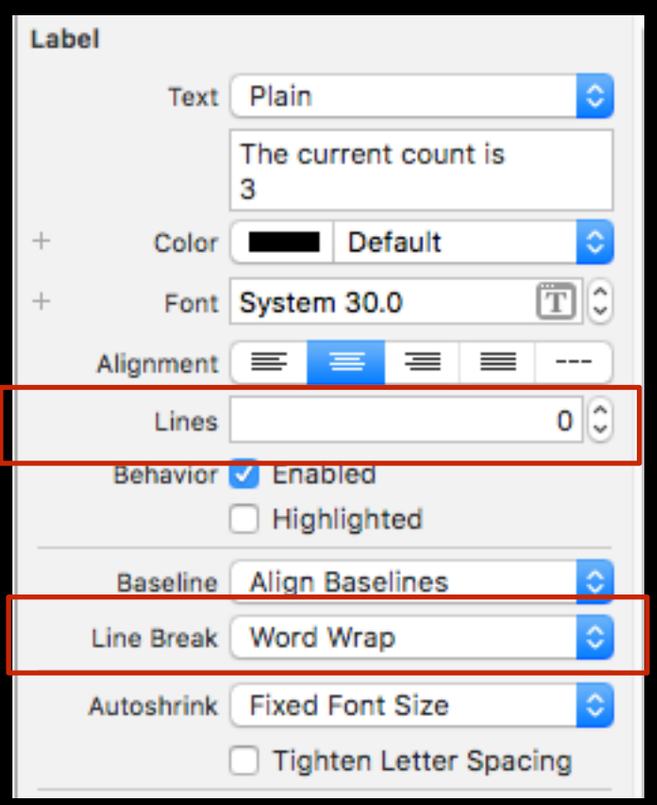
.alert

UITextField

UILabel

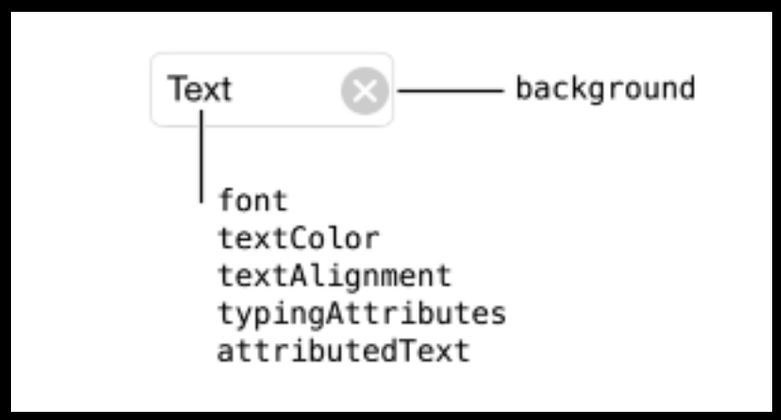
Displays multiple lines of static text

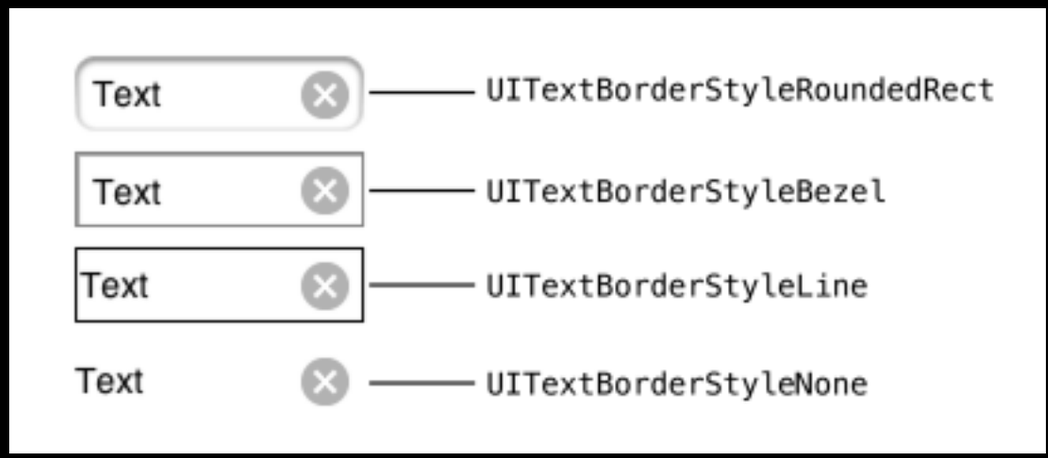
The volume of the ringer and alerts can be adjusted using the volume buttons.



UITextField

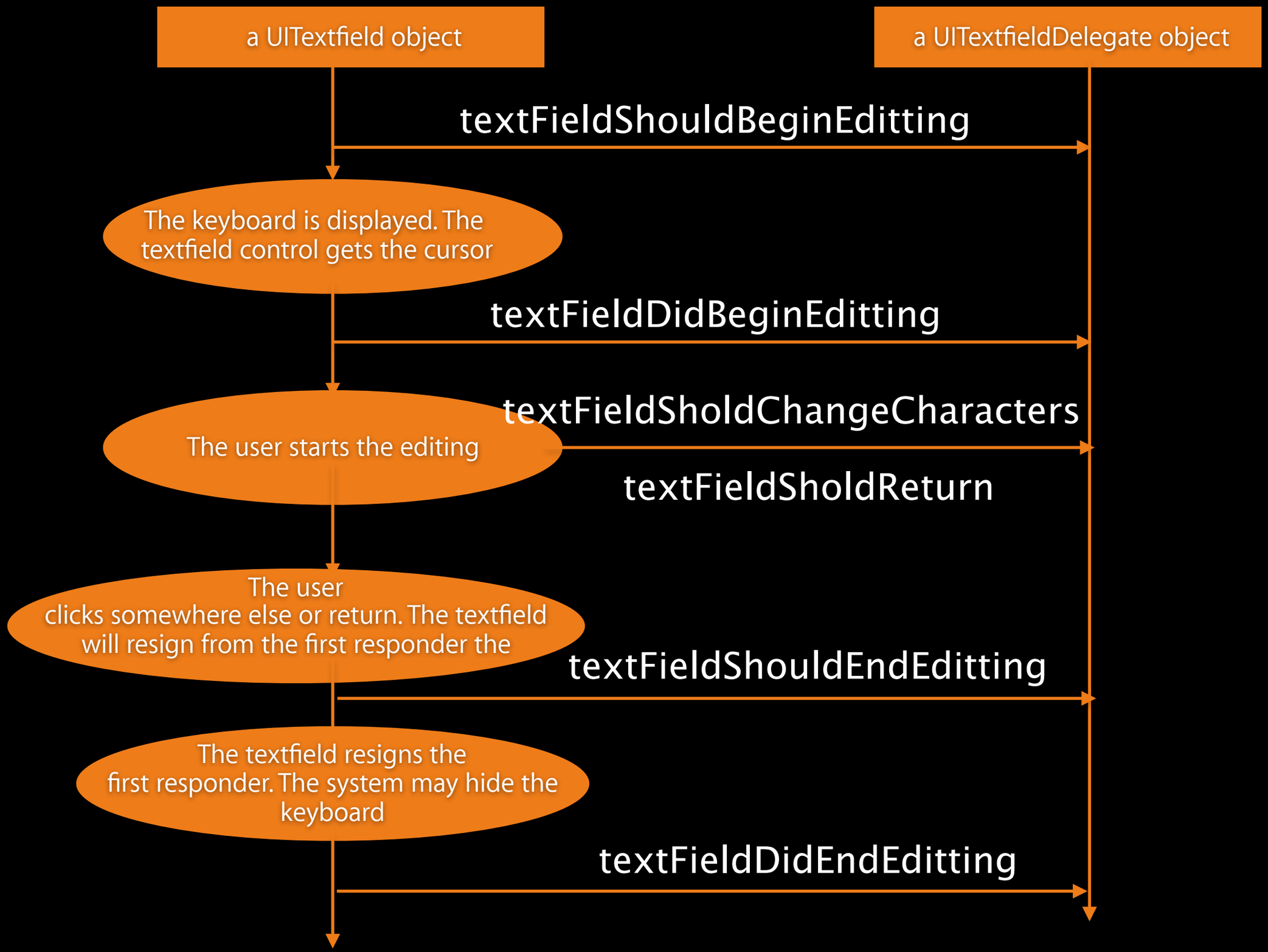
Displays an editable text area





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)”