iOS Tutorial

James Dempsey
http://jamesdempsey.net

Tapas Software
http://tapas-software.net

CocoaConf PDX
October 25, 2012

Getting Started

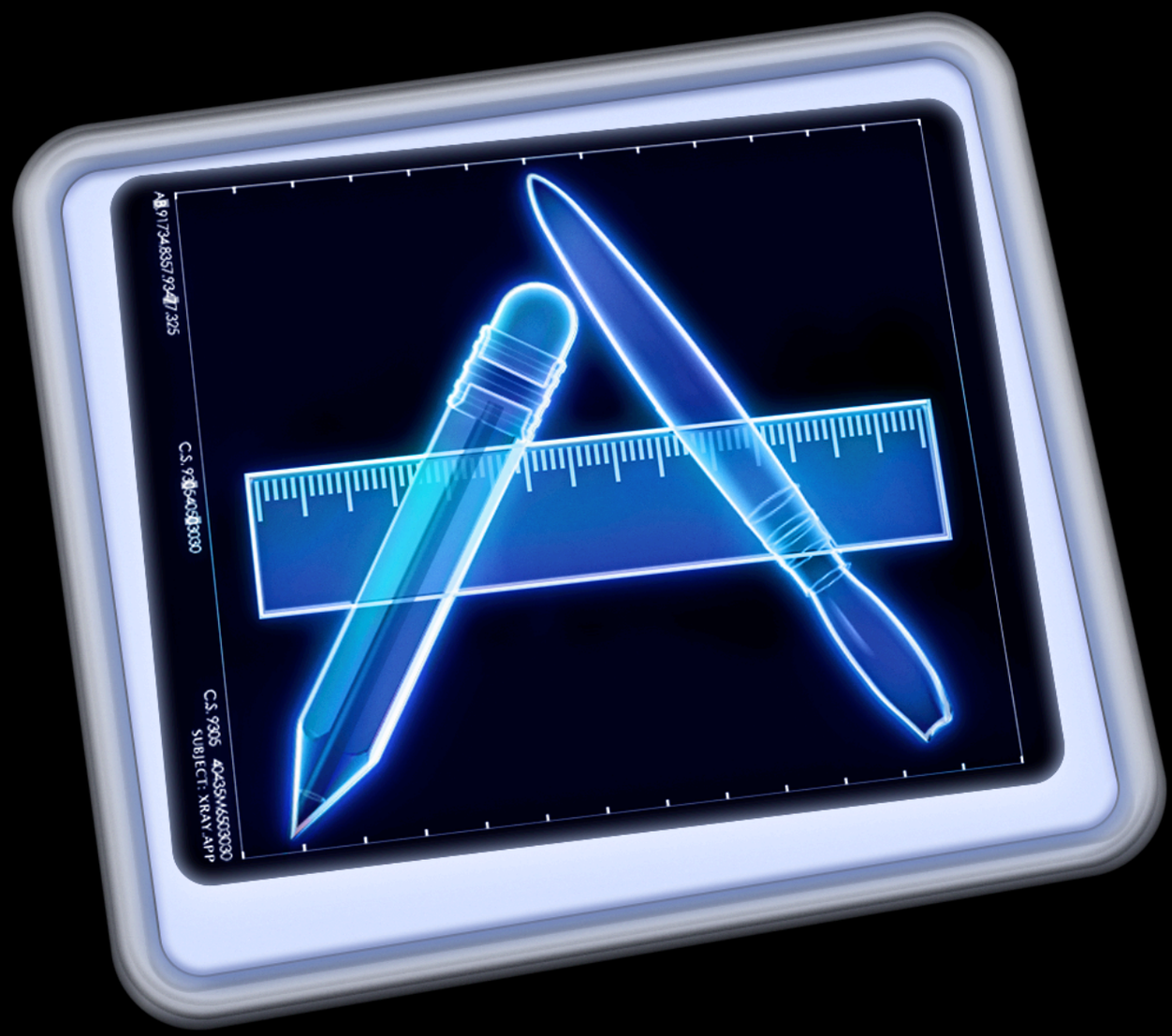
Tooling up and quick success

Xcode

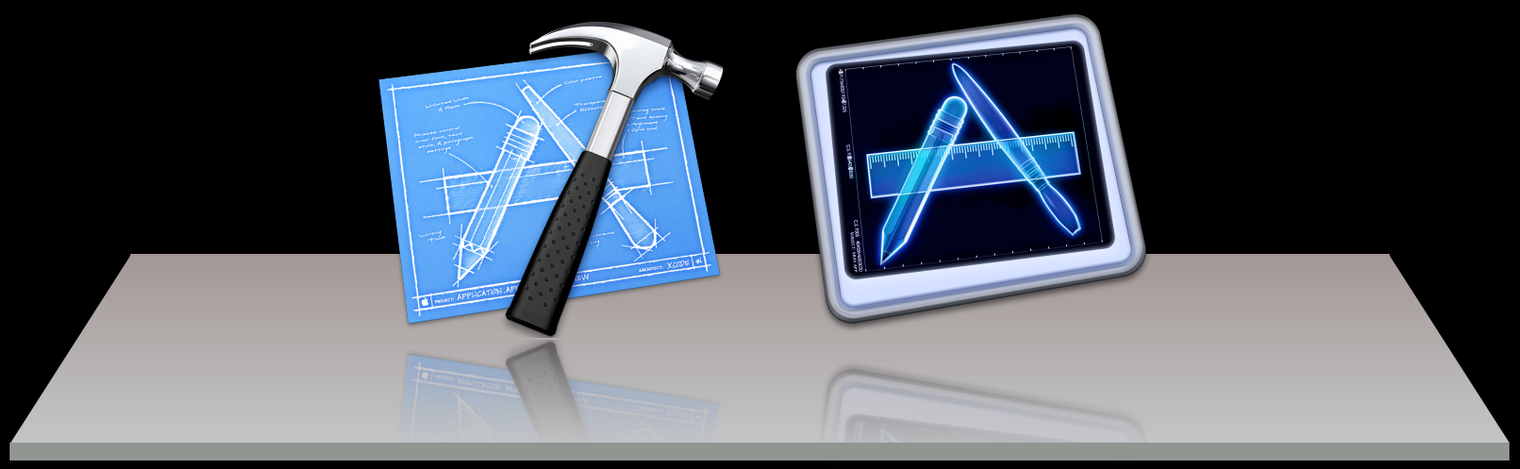

Unlimited Undo & Redo
Color palette
Precise control over font, text style, & paragraph settings
Writing tools
Text spacing
Alignment
PROJECT:
ARCHITECT: XCODE

Mac App Store

Instruments



Keep Your Tools in the Dock



Choose a template for your new project:

- iOS
  - Application
  - Framework & Library
  - Other
- Mac OS X
  - Application
  - Framework & Library
  - Application Plug-in
  - System Plug-in
  - Other

Master-Detail Application

OpenGL Game

Page-Based Application

Single View Application

Tabbed Application

Utility Application

Empty Application

Single View Application

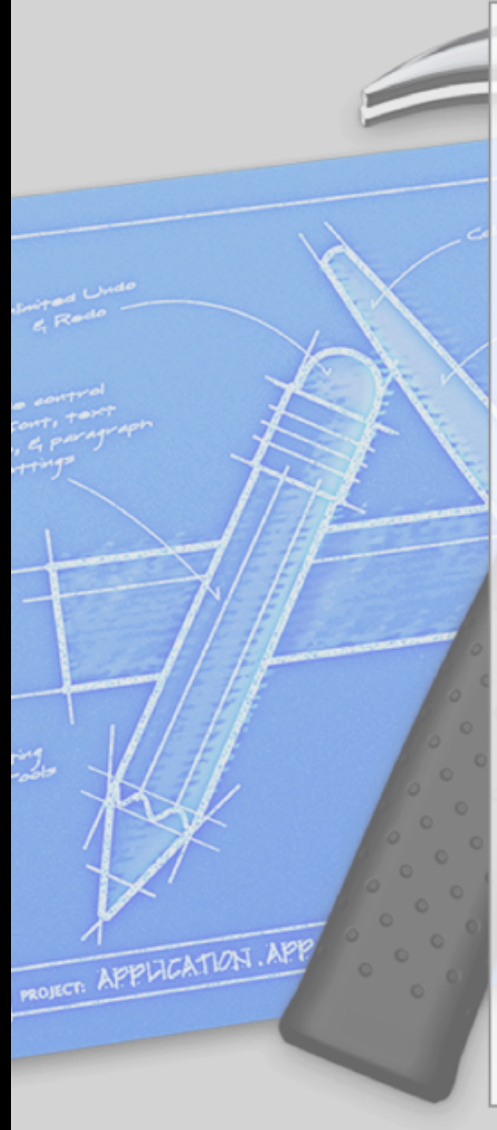
This template provides a starting point for an application that uses a single view. It provides a view controller to manage the view, and a storyboard or nib file that contains the view.

Cancel

Previous

Next

Choose options for your new project:



Product Name	Tweetings
Organization Name	Good Fun Software
Company Identifier	com.good-fun-software
Bundle Identifier	com.good-fun-software.Tweetings
Class Prefix	GFS
Devices	iPhone

- [x] Use Storyboards
- [x] Use Automatic Reference Counting
- [ ] Include Unit Tests

Cancel Previous Next

DrawingImages.xcodeproj — GFSViewController.m

Run | Stop | DrawingImages 〉 iOS Device (Scheme) | Breakpoints | Build Succeeded | 2/8/12 at 7:29 AM | Editor | View | Organizer

DrawingImages 〉 DrawingImages 〉 GFSViewController.m 〉 -thumbnailImage

DrawingImages
1 target, iOS SDK 5.1

- DrawingImages
  - phillip.jpg
  - GFSApp...egate.h
  - GFSApp...gate.m
  - MainS...board
  - GFSVie...troller.h
  - GFSVi...ller.m
  - Supp...g Files
- Frameworks
- Products

```
16 @end
17
18 @implementation GFSViewController
19
20 @synthesize imageView = _imageView;
21
22 -(UIImage *)thumbnailImage {
23   UIImage *image = nil;
24   NSString *path = [[NSBundle mainBundle] pathForResource:@"phillip" ofType:@"jpg"];
25   NSURL *url = [NSURL fileURLWithPath:path];
26   CGImageSourceRef imageSource = CGImageSourceCreateWithURL((__bridge CFURLRef)url,
27                                                             NULL);
28   if(NULL == imageSource) {
29     NSLog(@"failed to image source for %@", path);
30   } else {
31     NSNumber *maxWidth = [NSNumber numberWithInteger:1024];
32     NSNumber *always = [NSNumber numberWithBool:YES];
33     NSNumber *transform = [NSNumber numberWithBool:YES];
34     NSDictionary *dictionary = [NSDictionary dictionaryWithObjectsAndKeys:
35                                 maxWidth, (__bridge NSString *)kCGImageSourceThumbnailMaxPixelSize,
36                                 always, (__bridge NSString *)kCGImageSourceCreateThumbnailFromImageAlways,
37                                 transform, (__bridge NSString *)kCGImageSourceCreateThumbnailWithTransform,
38                                 nil];
39     CGImageRef cgImage = CGImageSourceCreateThumbnailAtIndex(imageSource, 0,
40                                                              (__bridge CFDictionaryRef)dictionary);
41     image = [UIImage imageWithCGImage:cgImage];
42     CFRelease(imageSource);
43     CGImageRelease(cgImage);
44   }
45   return image;
46 }
47
48 - (void)viewDidLoad {
49   [super viewDidLoad];
50   self.imageView.image = [self thumbnailImage];
51 }
```

No Selection

Auto | All Output | Clear

Quick Help

Name: dictionaryWithObjectsAndKeys:

Declaration: + (id)dictionaryWithObjectsAndKeys: (id)firstObject , ...

Availability: iOS (2.0 and later)

Abstract: Creates and returns a dictionary containing entries constructed from the specified set of values and keys.

Parameters:
firstObject: The first value to add to the new dictionary.
...: First the key for firstObject, then a null-terminated list of alternating values and keys.

Declared In: NSDictionary.h

Reference: NSDictionary Class Reference

Related API: dictionary, dictionaryWithContentsOfFile:, dictionaryWithContentsOfURL:, dictionaryWithDictionary:, dictionaryWithObject:forKey:, dictionaryWithObjects:forKeys:, dictionaryWithObjects:forKeys:count:,

Object Library

Push Button – Intercepts mouse-down events and sends an action message to a target object when it's clicked or...

Gradient Button – Intercepts mouse-down events and sends an action message to a target object when it's...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

DrawingImages.xcodeproj — GFSViewController.m

Run | Stop | DrawingImages 〉 iOS Device (Scheme) | Breakpoints | Build Succeeded | 2/8/12 at 7:29 AM | Editor | View | Organizer

DrawingImages 〉 DrawingImages 〉 GFSViewController.m 〉 -thumbnailImage

DrawingImages
1 target, iOS SDK 5.1

- DrawingImages
  - phillip.jpg
  - GFSApp...egate.h
  - GFSApp...gate.m
  - MainS...board
  - GFSVie...troller.h
  - GFSVi...ller.m
  - Supp...g Files
- Frameworks
- Products

```
16 @end
17
18 @implementation GFSViewController
19
20 @synthesize imageView = _imageView;
21
22 -(UIImage *)thumbnailImage {
23   UIImage *image = nil;
24   NSString *path = [[NSBundle mainBundle] pathForResource:@"phillip" ofType:@"jpg"];
25   NSURL *url = [NSURL fileURLWithPath:path];
26   CGImageSourceRef imageSource = CGImageSourceCreateWithURL((__bridge CFURLRef)url,
27                                                             NULL);
28   if(NULL == imageSource) {
29     NSLog(@"failed to image source for %@", path);
30   } else {
31     NSNumber *maxWidth = [NSNumber numberWithInteger:1024];
32     NSNumber *always = [NSNumber numberWithBool:YES];
33     NSNumber *transform = [NSNumber numberWithBool:YES];
34     NSDictionary *dictionary = [NSDictionary dictionaryWithObjectsAndKeys:
35                                 maxWidth, (__bridge NSString *)kCGImageSourceThumbnailMaxPixelSize,
36                                 always, (__bridge NSString *)kCGImageSourceCreateThumbnailFromImageAlways,
37                                 transform, (__bridge NSString *)kCGImageSourceCreateThumbnailWithTransform,
38                                 nil];
39     CGImageRef cgImage = CGImageSourceCreateThumbnailAtIndex(imageSource, 0,
40                                                              (__bridge CFDictionaryRef)dictionary);
41     image = [UIImage imageWithCGImage:cgImage];
42     CFRelease(imageSource);
43     CGImageRelease(cgImage);
44   }
45   return image;
46 }
47
48 - (void)viewDidLoad {
49   [super viewDidLoad];
50   self.imageView.image = [self thumbnailImage];
51 }
```

No Selection

Auto | All Output | Clear

Quick Help

Name: dictionaryWithObjectsAndKeys:

Declaration: + (id)dictionaryWithObjectsAndKeys: (id)firstObject , ...

Availability: iOS (2.0 and later)

Abstract: Creates and returns a dictionary containing entries constructed from the specified set of values and keys.

Parameters:
firstObject: The first value to add to the new dictionary.
...: First the key for firstObject, then a null-terminated list of alternating values and keys.

Declared In: NSDictionary.h

Reference: NSDictionary Class Reference

Related API: dictionary, dictionaryWithContentsOfFile:, dictionaryWithContentsOfURL:, dictionaryWithDictionary:, dictionaryWithObject:forKey:, dictionaryWithObjects:forKeys:, dictionaryWithObjects:forKeys:count:,

Object Library

Push Button – Intercepts mouse-down events and sends an action message to a target object when it's clicked or...

Gradient Button – Intercepts mouse-down events and sends an action message to a target object when it's...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

DrawingImages.xcodeproj — GFSViewController.m

Build **Succeeded** | 2/8/12 at 7:29 AM

Run Stop DrawingImages › iOS Device Scheme Breakpoints Editor View Organizer

DrawingImages › DrawingImages › GFSViewController.m › -thumbnailImage

DrawingImages
1 target, iOS SDK 5.1

- DrawingImages
  - phillip.jpg
  - GFSApp...egate.h
  - GFSApp...gate.m
  - MainS...board
  - GFSVie...troller.h
  - GFSVi...ller.m
  - Supp...g Files
- Frameworks
- Products

```
@end

@implementation GFSViewController

@synthesize imageView = _imageView;

-(UIImage *)thumbnailImage {
  UIImage *image = nil;
  NSString *path = [[NSBundle mainBundle] pathForResource:@"phillip" ofType:@"jpg"];
  NSURL *url = [NSURL fileURLWithPath:path];
  CGImageSourceRef imageSource = CGImageSourceCreateWithURL((__bridge CFURLRef)url,
                                                            NULL);
  if(NULL == imageSource) {
    NSLog(@"failed to image source for %@", path);
  } else {
    NSNumber *maxWidth = [NSNumber numberWithInteger:1024];
    NSNumber *always = [NSNumber numberWithBool:YES];
    NSNumber *transform = [NSNumber numberWithBool:YES];
    NSDictionary *dictionary = [NSDictionary dictionaryWithObjectsAndKeys:
                                maxWidth, (__bridge NSString *)kCGImageSourceThumbnailMaxPixelSize,
                                always, (__bridge NSString *)kCGImageSourceCreateThumbnailFromImageAlways,
                                transform, (__bridge NSString *)kCGImageSourceCreateThumbnailWithTransform,
                                nil];
    CGImageRef cgImage = CGImageSourceCreateThumbnailAtIndex(imageSource, 0,
                                                             (__bridge CFDictionaryRef)dictionary);
    image = [UIImage imageWithCGImage:cgImage];
    CFRelease(imageSource);
    CGImageRelease(cgImage);
  }
  return image;
}

- (void)viewDidLoad {
  [super viewDidLoad];
  self.imageView.image = [self thumbnailImage];
}
```

No Selection

Auto | All Output | Clear

Quick Help

Name: dictionaryWithObjectsAndKeys:

Declaration: + (id)dictionaryWithObjectsAndKeys: (id)firstObject , ...

Availability: iOS (2.0 and later)

Abstract: Creates and returns a dictionary containing entries constructed from the specified set of values and keys.

Parameters:
firstObject: The first value to add to the new dictionary.
...: First the key for firstObject, then a null-terminated list of alternating values and keys.

Declared In: NSDictionary.h

Reference: NSDictionary Class Reference

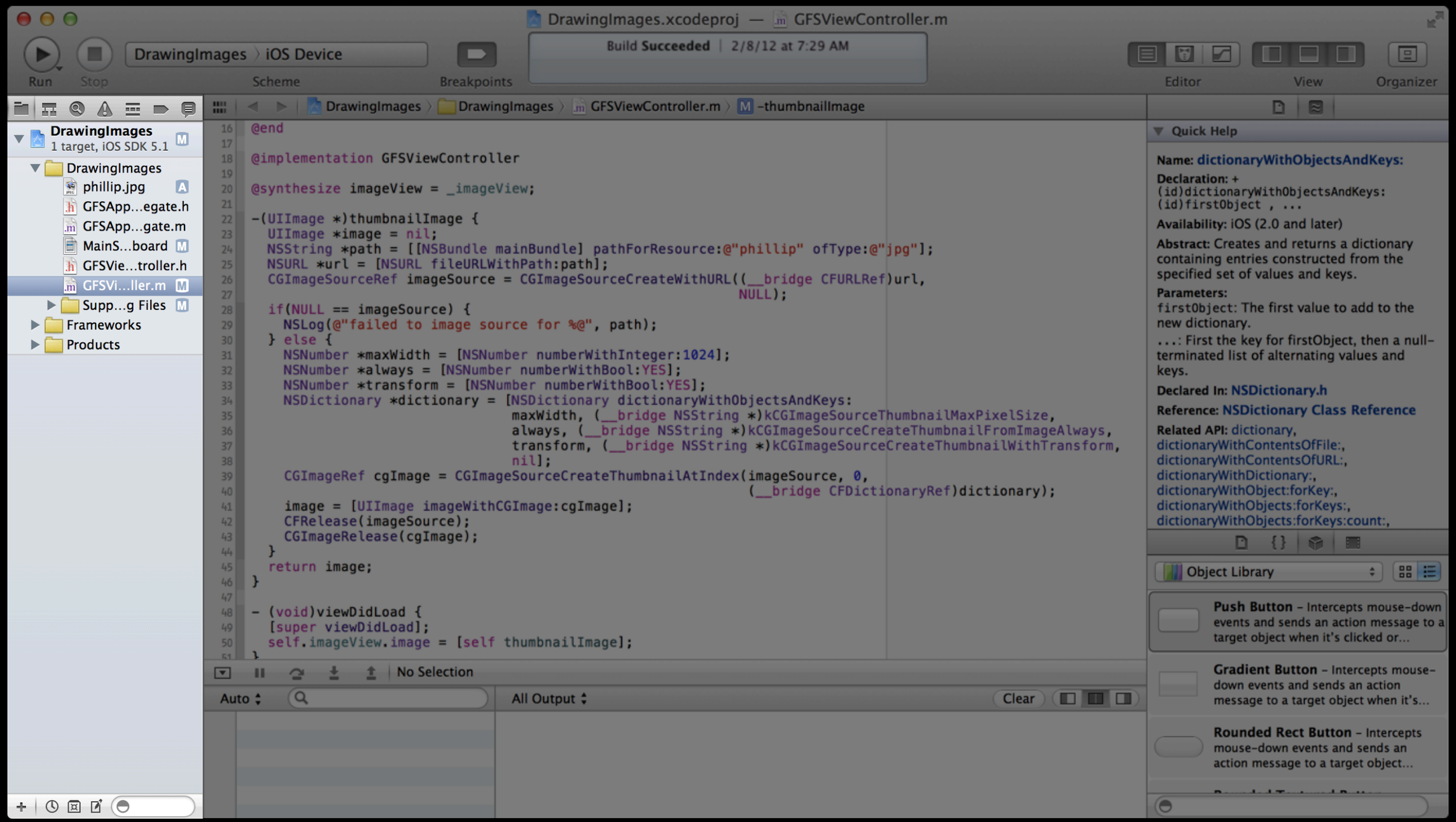
Related API: dictionary, dictionaryWithContentsOfFile:, dictionaryWithContentsOfURL:, dictionaryWithDictionary:, dictionaryWithObject:forKey:, dictionaryWithObjects:forKeys:, dictionaryWithObjects:forKeys:count:,

Object Library

Push Button – Intercepts mouse-down events and sends an action message to a target object when it's clicked or...

Gradient Button – Intercepts mouse-down events and sends an action message to a target object when it's...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...


DrawingImages.xcodeproj — GFSViewController.m
Build Succeeded | 2/8/12 at 7:29 AM
Run
Stop
DrawingImages 〉 iOS Device
Scheme
Breakpoints
Editor
View
Organizer
DrawingImages 〉 DrawingImages 〉 GFSViewController.m 〉 -thumbnailImage
DrawingImages
1 target, iOS SDK 5.1
DrawingImages
phillip.jpg
GFSApp...egate.h
GFSApp...gate.m
MainS...board
GFSVie...troller.h
GFSVi...ller.m
Supp...g Files
Frameworks
Products
@end
@implementation GFSViewController
@synthesize imageView = _imageView;
-(UIImage *)thumbnailImage {
UIImage *image = nil;
NSString *path = [[NSBundle mainBundle] pathForResource:@"phillip" ofType:@"jpg"];
NSURL *url = [NSURL fileURLWithPath:path];
CGImageSourceRef imageSource = CGImageSourceCreateWithURL((__bridge CFURLRef)url,
NULL);
if(NULL == imageSource) {
NSLog(@"failed to image source for %@", path);
} else {
NSNumber *maxWidth = [NSNumber numberWithInteger:1024];
NSNumber *always = [NSNumber numberWithBool:YES];
NSNumber *transform = [NSNumber numberWithBool:YES];
NSDictionary *dictionary = [NSDictionary dictionaryWithObjectsAndKeys:
maxWidth, (__bridge NSString *)kCGImageSourceThumbnailMaxPixelSize,
always, (__bridge NSString *)kCGImageSourceCreateThumbnailFromImageAlways,
transform, (__bridge NSString *)kCGImageSourceCreateThumbnailWithTransform,
nil];
CGImageRef cgImage = CGImageSourceCreateThumbnailAtIndex(imageSource, 0,
(__bridge CFDictionaryRef)dictionary);
image = [UIImage imageWithCGImage:cgImage];
CFRelease(imageSource);
CGImageRelease(cgImage);
}
return image;
}
- (void)viewDidLoad {
[super viewDidLoad];
self.imageView.image = [self thumbnailImage];
}
No Selection
Auto
All Output
Clear
Quick Help
Name: dictionaryWithObjectsAndKeys:
Declaration: + (id)dictionaryWithObjectsAndKeys: (id)firstObject , ...
Availability: iOS (2.0 and later)
Abstract: Creates and returns a dictionary containing entries constructed from the specified set of values and keys.
Parameters:
firstObject: The first value to add to the new dictionary.
...: First the key for firstObject, then a null-terminated list of alternating values and keys.
Declared In: NSDictionary.h
Reference: NSDictionary Class Reference
Related API: dictionary, dictionaryWithContentsOfFile:, dictionaryWithContentsOfURL:, dictionaryWithDictionary:, dictionaryWithObject:forKey:, dictionaryWithObjects:forKeys:, dictionaryWithObjects:forKeys:count:,
Object Library
Push Button – Intercepts mouse-down events and sends an action message to a target object when it's clicked or...
Gradient Button – Intercepts mouse-down events and sends an action message to a target object when it's...
Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

DrawingImages.xcodeproj — GFSViewController.m

Build Succeeded | 2/8/12 at 7:29 AM

Run Stop Scheme Breakpoints Editor View Organizer

DrawingImages 〉 DrawingImages 〉 GFSViewController.m 〉 -thumbnailImage

DrawingImages
1 target, iOS SDK 5.1

- DrawingImages
  - phillip.jpg
  - GFSApp...egate.h
  - GFSApp...gate.m
  - MainS...board
  - GFSVie...troller.h
  - GFSVi...ller.m
  - Supp...g Files
- Frameworks
- Products

```
16 @end
17
18 @implementation GFSViewController
19
20 @synthesize imageView = _imageView;
21
22 -(UIImage *)thumbnailImage {
23   UIImage *image = nil;
24   NSString *path = [[NSBundle mainBundle] pathForResource:@"phillip" ofType:@"jpg"];
25   NSURL *url = [NSURL fileURLWithPath:path];
26   CGImageSourceRef imageSource = CGImageSourceCreateWithURL((__bridge CFURLRef)url,
27                                                             NULL);
28   if(NULL == imageSource) {
29     NSLog(@"failed to image source for %@", path);
30   } else {
31     NSNumber *maxWidth = [NSNumber numberWithInteger:1024];
32     NSNumber *always = [NSNumber numberWithBool:YES];
33     NSNumber *transform = [NSNumber numberWithBool:YES];
34     NSDictionary *dictionary = [NSDictionary dictionaryWithObjectsAndKeys:
35                                 maxWidth, (__bridge NSString *)kCGImageSourceThumbnailMaxPixelSize,
36                                 always, (__bridge NSString *)kCGImageSourceCreateThumbnailFromImageAlways,
37                                 transform, (__bridge NSString *)kCGImageSourceCreateThumbnailWithTransform,
38                                 nil];
39     CGImageRef cgImage = CGImageSourceCreateThumbnailAtIndex(imageSource, 0,
40                                                              (__bridge CFDictionaryRef)dictionary);
41     image = [UIImage imageWithCGImage:cgImage];
42     CFRelease(imageSource);
43     CGImageRelease(cgImage);
44   }
45   return image;
46 }
47
48 - (void)viewDidLoad {
49   [super viewDidLoad];
50   self.imageView.image = [self thumbnailImage];
51 }
```

No Selection

Auto | All Output | Clear

Quick Help

Name: dictionaryWithObjectsAndKeys:

Declaration: + (id)dictionaryWithObjectsAndKeys: (id)firstObject , ...

Availability: iOS (2.0 and later)

Abstract: Creates and returns a dictionary containing entries constructed from the specified set of values and keys.

Parameters:
firstObject: The first value to add to the new dictionary.
...: First the key for firstObject, then a null-terminated list of alternating values and keys.

Declared In: NSDictionary.h

Reference: NSDictionary Class Reference

Related API: dictionary, dictionaryWithContentsOfFile:, dictionaryWithContentsOfURL:, dictionaryWithDictionary:, dictionaryWithObject:forKey:, dictionaryWithObjects:forKeys:, dictionaryWithObjects:forKeys:count:,

Object Library

Push Button – Intercepts mouse-down events and sends an action message to a target object when it's clicked or...

Gradient Button – Intercepts mouse-down events and sends an action message to a target object when it's...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

DrawingImages.xcodeproj — GFSViewController.m

Run | Stop | DrawingImages › iOS Device (Scheme) | Breakpoints | Build Succeeded | 2/8/12 at 7:29 AM | Editor | View | Organizer

DrawingImages › DrawingImages › GFSViewController.m › -thumbnailImage

DrawingImages
1 target, iOS SDK 5.1

- DrawingImages
  - phillip.jpg
  - GFSApp...egate.h
  - GFSApp...gate.m
  - MainS...board
  - GFSVie...troller.h
  - GFSVi...ller.m
  - Supp...g Files
- Frameworks
- Products

```
16 @end
17
18 @implementation GFSViewController
19
20 @synthesize imageView = _imageView;
21
22 -(UIImage *)thumbnailImage {
23   UIImage *image = nil;
24   NSString *path = [[NSBundle mainBundle] pathForResource:@"phillip" ofType:@"jpg"];
25   NSURL *url = [NSURL fileURLWithPath:path];
26   CGImageSourceRef imageSource = CGImageSourceCreateWithURL((__bridge CFURLRef)url,
27                                                             NULL);
28   if(NULL == imageSource) {
29     NSLog(@"failed to image source for %@", path);
30   } else {
31     NSNumber *maxWidth = [NSNumber numberWithInteger:1024];
32     NSNumber *always = [NSNumber numberWithBool:YES];
33     NSNumber *transform = [NSNumber numberWithBool:YES];
34     NSDictionary *dictionary = [NSDictionary dictionaryWithObjectsAndKeys:
35                                 maxWidth, (__bridge NSString *)kCGImageSourceThumbnailMaxPixelSize,
36                                 always, (__bridge NSString *)kCGImageSourceCreateThumbnailFromImageAlways,
37                                 transform, (__bridge NSString *)kCGImageSourceCreateThumbnailWithTransform,
38                                 nil];
39     CGImageRef cgImage = CGImageSourceCreateThumbnailAtIndex(imageSource, 0,
40                                                              (__bridge CFDictionaryRef)dictionary);
41     image = [UIImage imageWithCGImage:cgImage];
42     CFRelease(imageSource);
43     CGImageRelease(cgImage);
44   }
45   return image;
46 }
47
48 - (void)viewDidLoad {
49   [super viewDidLoad];
50   self.imageView.image = [self thumbnailImage];
51 }
```

No Selection

Auto | All Output | Clear

Quick Help

Name: dictionaryWithObjectsAndKeys:

Declaration: + (id)dictionaryWithObjectsAndKeys: (id)firstObject , ...

Availability: iOS (2.0 and later)

Abstract: Creates and returns a dictionary containing entries constructed from the specified set of values and keys.

Parameters:

`firstObject`: The first value to add to the new dictionary.

`...`: First the key for firstObject, then a null-terminated list of alternating values and keys.

Declared In: NSDictionary.h

Reference: NSDictionary Class Reference

Related API: dictionary, dictionaryWithContentsOfFile:, dictionaryWithContentsOfURL:, dictionaryWithDictionary:, dictionaryWithObject:forKey:, dictionaryWithObjects:forKeys:, dictionaryWithObjects:forKeys:count:,

Object Library

- **Push Button** – Intercepts mouse-down events and sends an action message to a target object when it's clicked or...
- **Gradient Button** – Intercepts mouse-down events and sends an action message to a target object when it's...
- **Rounded Rect Button** – Intercepts mouse-down events and sends an action message to a target object...

DrawingImages.xcodeproj — GFSViewController.m

Run | Stop | DrawingImages 〉 iOS Device (Scheme) | Breakpoints | Build Succeeded | 2/8/12 at 7:29 AM | Editor | View | Organizer

DrawingImages 〉 DrawingImages 〉 GFSViewController.m 〉 -thumbnailImage

DrawingImages
1 target, iOS SDK 5.1

- DrawingImages
  - phillip.jpg
  - GFSApp...egate.h
  - GFSApp...gate.m
  - MainS...board
  - GFSVie...troller.h
  - GFSVi...ller.m
  - Supp...g Files
- Frameworks
- Products

```
16 @end
17
18 @implementation GFSViewController
19
20 @synthesize imageView = _imageView;
21
22 -(UIImage *)thumbnailImage {
23   UIImage *image = nil;
24   NSString *path = [[NSBundle mainBundle] pathForResource:@"phillip" ofType:@"jpg"];
25   NSURL *url = [NSURL fileURLWithPath:path];
26   CGImageSourceRef imageSource = CGImageSourceCreateWithURL((__bridge CFURLRef)url,
27                                                             NULL);
28   if(NULL == imageSource) {
29     NSLog(@"failed to image source for %@", path);
30   } else {
31     NSNumber *maxWidth = [NSNumber numberWithInteger:1024];
32     NSNumber *always = [NSNumber numberWithBool:YES];
33     NSNumber *transform = [NSNumber numberWithBool:YES];
34     NSDictionary *dictionary = [NSDictionary dictionaryWithObjectsAndKeys:
35                                 maxWidth, (__bridge NSString *)kCGImageSourceThumbnailMaxPixelSize,
36                                 always, (__bridge NSString *)kCGImageSourceCreateThumbnailFromImageAlways,
37                                 transform, (__bridge NSString *)kCGImageSourceCreateThumbnailWithTransform,
38                                 nil];
39     CGImageRef cgImage = CGImageSourceCreateThumbnailAtIndex(imageSource, 0,
40                                                              (__bridge CFDictionaryRef)dictionary);
41     image = [UIImage imageWithCGImage:cgImage];
42     CFRelease(imageSource);
43     CGImageRelease(cgImage);
44   }
45   return image;
46 }
47
48 - (void)viewDidLoad {
49   [super viewDidLoad];
50   self.imageView.image = [self thumbnailImage];
51 }
```

No Selection

Auto | All Output | Clear

Quick Help

Name: dictionaryWithObjectsAndKeys:

Declaration: + (id)dictionaryWithObjectsAndKeys: (id)firstObject , ...

Availability: iOS (2.0 and later)

Abstract: Creates and returns a dictionary containing entries constructed from the specified set of values and keys.

Parameters:
firstObject: The first value to add to the new dictionary.
...: First the key for firstObject, then a null-terminated list of alternating values and keys.

Declared In: NSDictionary.h

Reference: NSDictionary Class Reference

Related API: dictionary, dictionaryWithContentsOfFile:, dictionaryWithContentsOfURL:, dictionaryWithDictionary:, dictionaryWithObject:forKey:, dictionaryWithObjects:forKeys:, dictionaryWithObjects:forKeys:count:,

Object Library

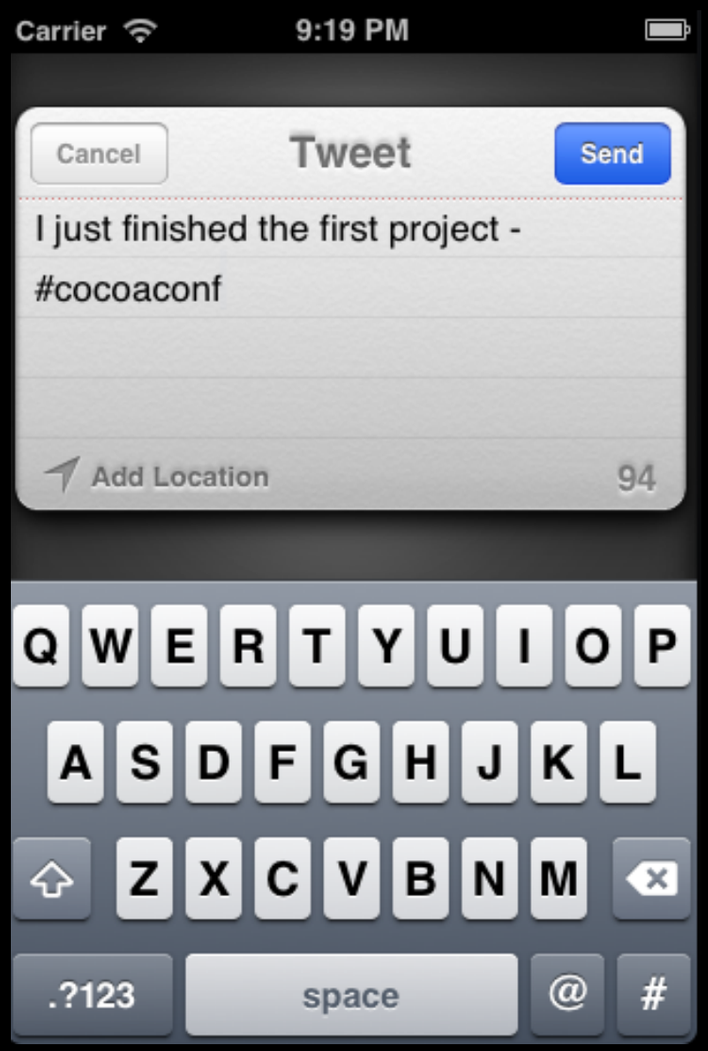
Push Button – Intercepts mouse-down events and sends an action message to a target object when it's clicked or...

Gradient Button – Intercepts mouse-down events and sends an action message to a target object when it's...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

Let’s Build!


Carrier
9:19 PM
Tweet!


Carrier
9:19 PM
Cancel
Tweet
Send
I just finished the first project -
#cocoaconf
Add Location
94
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
.?123
space
@
#


Carrier
5:48 AM
twitter
Sign in
Tweets
CocoaConf @cocoaconf 9h
Last reminder for the early birds: There are only two hours left to get the #cocoaconf discount! cocoaconf.com/register
CocoaConf @cocoaconf 17h
@irregulara Columbus was our first event (August, 2011). We will definitely be back! cocoaconf.com/blog/columbus-…
CocoaConf @cocoaconf 19h
Looks like most of the Early Birds got their tickets! Those of you that want to pay more will be able to tomorrow. :-) cocoaconf.com/register
4

Choose a template for your new project

iOS
- Application
- Framework & Library
- Other

OS X
- Application
- Framework & Library
- Application Plug-in
- System Plug-in
- Other

Master-Detail Application

OpenGL Game

Page-Based Application

Single View Application

Tabbed Application

Utility Application

Empty Application

Single View Application

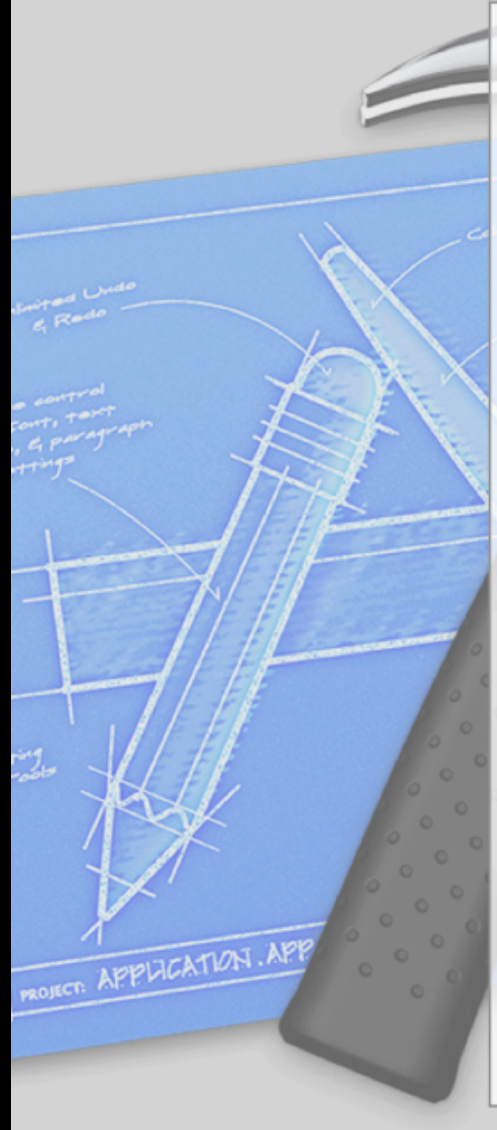
This template provides a starting point for an application that uses a single view. It provides a view controller to manage the view, and a storyboard or nib file that contains the view.

Cancel

Previous

Next

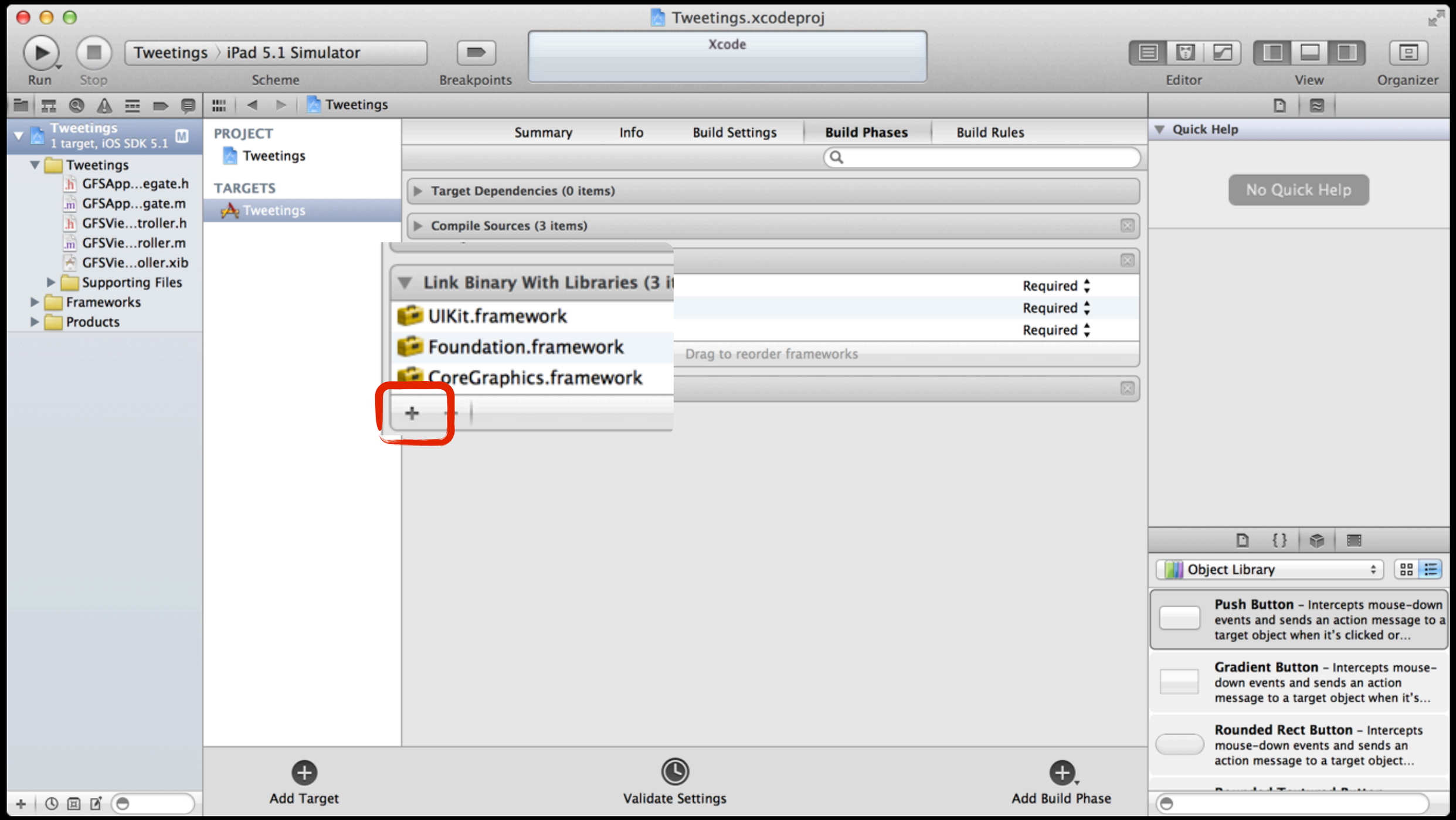
Choose options for your new project:



Product Name	Tweetings
Organization Name	Good Fun Software
Company Identifier	com.good-fun-software
Bundle Identifier	com.good-fun-software.Tweetings
Class Prefix	GFS
Devices	iPhone

- [x] Use Storyboards
- [x] Use Automatic Reference Counting
- [ ] Include Unit Tests

Cancel Previous Next


Tweetings.xcodeproj
Xcode
Run
Stop
Tweetings 〉 iPad 5.1 Simulator
Scheme
Breakpoints
Editor
View
Organizer
Tweetings
1 target, iOS SDK 5.1
Tweetings
GFSApp...egate.h
GFSApp...gate.m
GFSVie...troller.h
GFSVie...roller.m
GFSVie...oller.xib
Supporting Files
Frameworks
Products
PROJECT
Tweetings
TARGETS
Tweetings
Summary
Info
Build Settings
Build Phases
Build Rules
Target Dependencies (0 items)
Compile Sources (3 items)
Link Binary With Libraries (3 i
UIKit.framework
Foundation.framework
CoreGraphics.framework
Required
Required
Required
Drag to reorder frameworks
Quick Help
No Quick Help
Object Library
Push Button – Intercepts mouse-down events and sends an action message to a target object when it's clicked or...
Gradient Button – Intercepts mouse-down events and sends an action message to a target object when it's...
Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...
Add Target
Validate Settings
Add Build Phase

Choose frameworks and libraries to add:

Soc

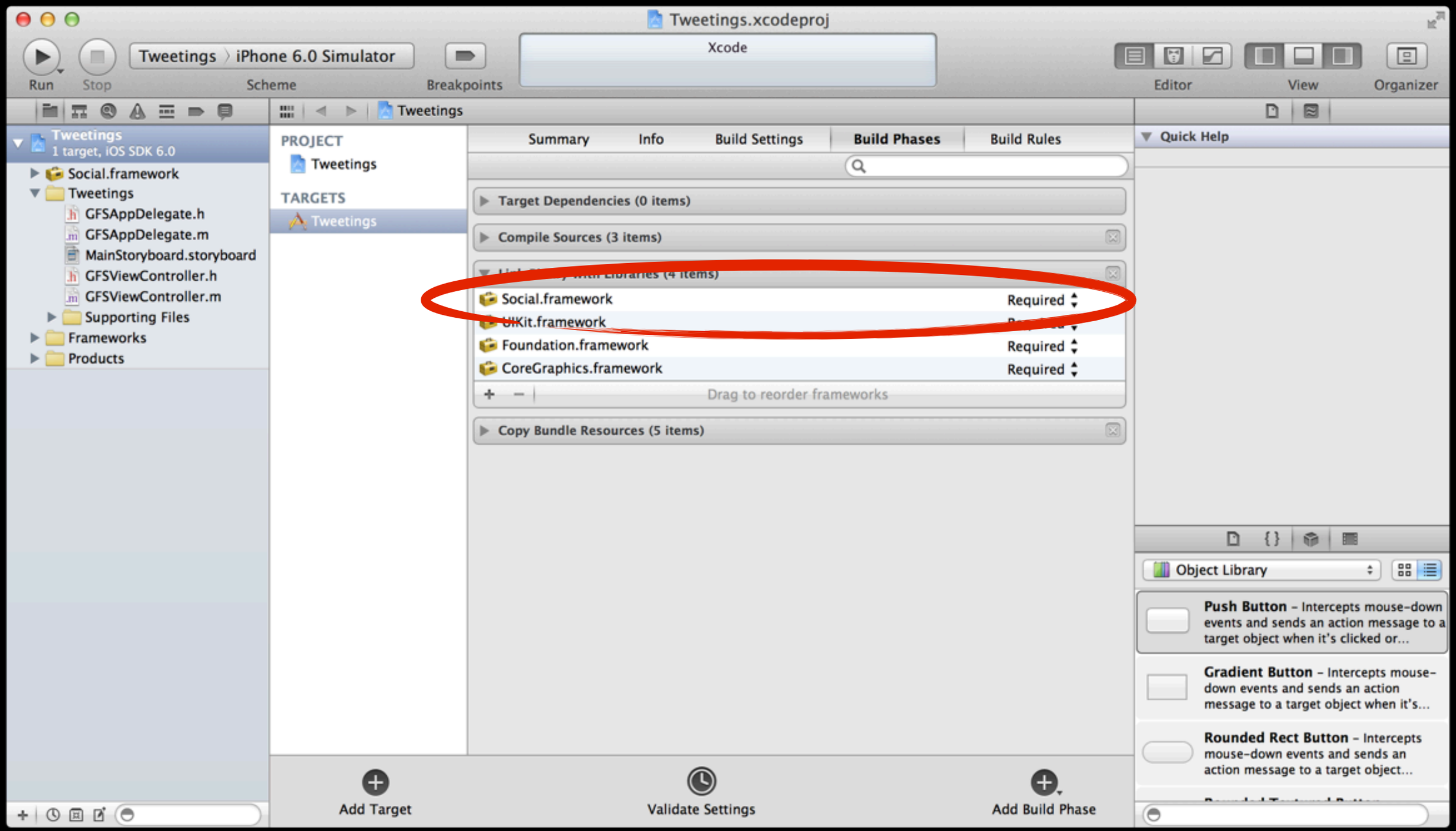
iOS 6.0

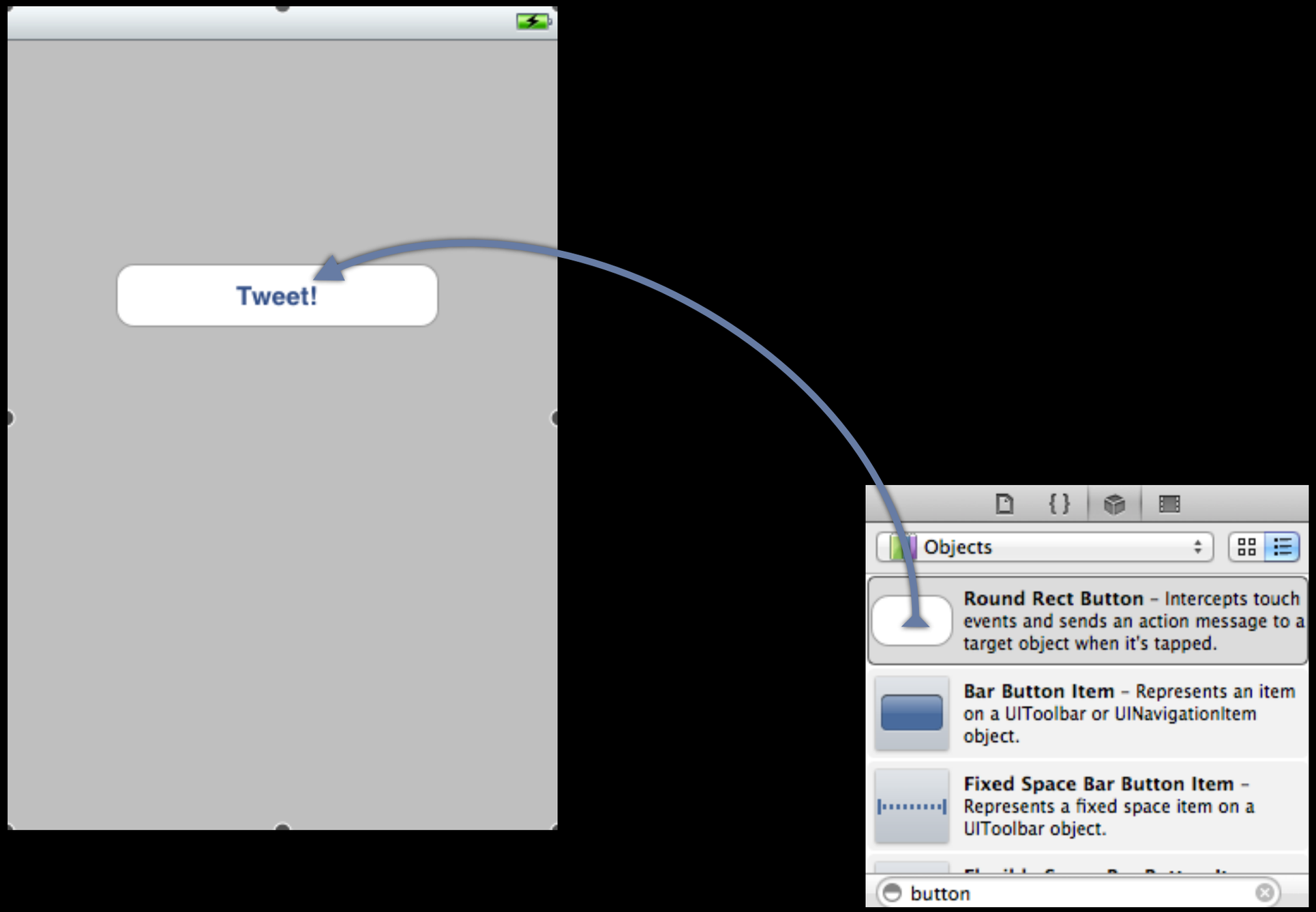
Social.framework

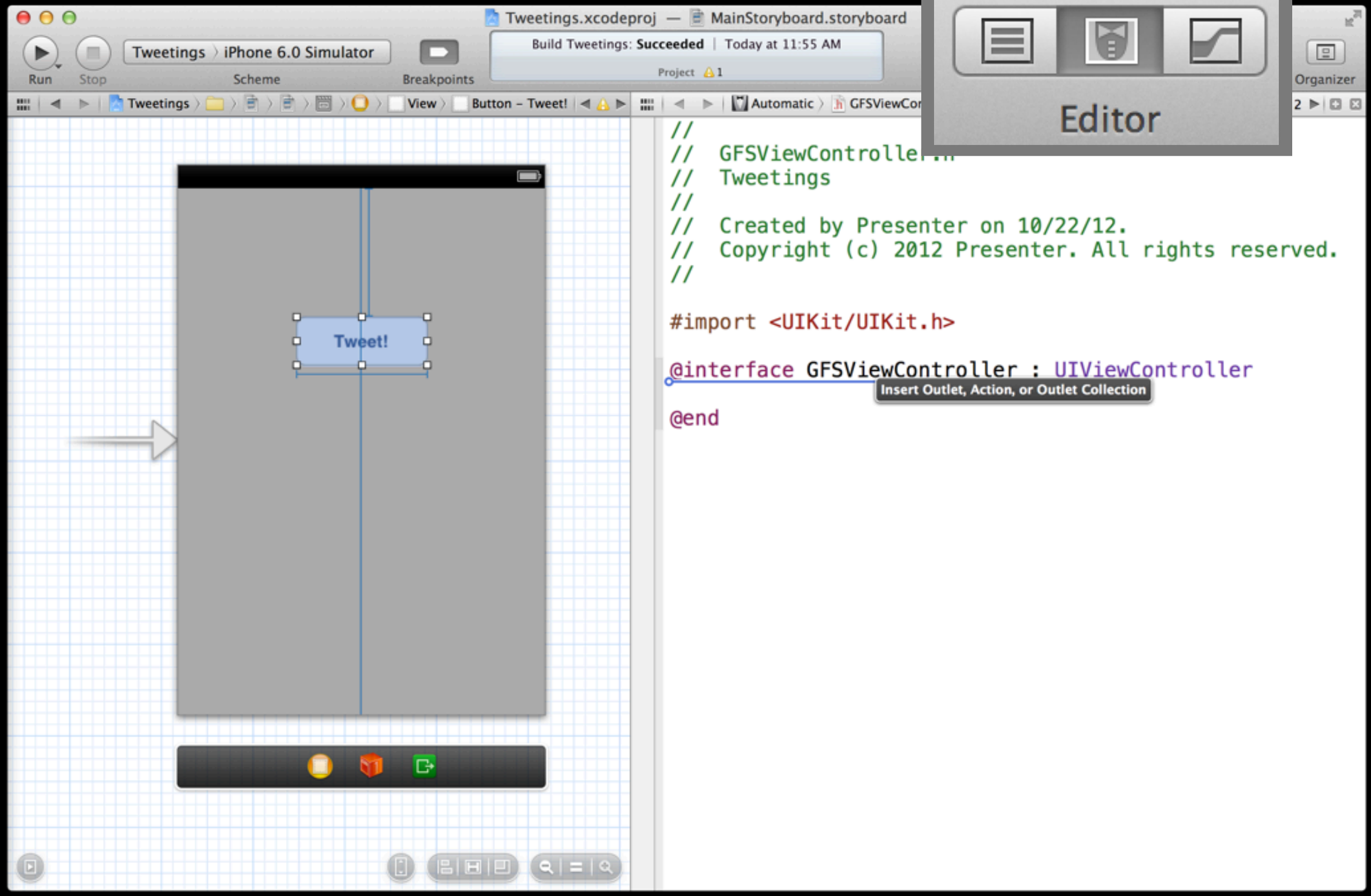
Add Other…

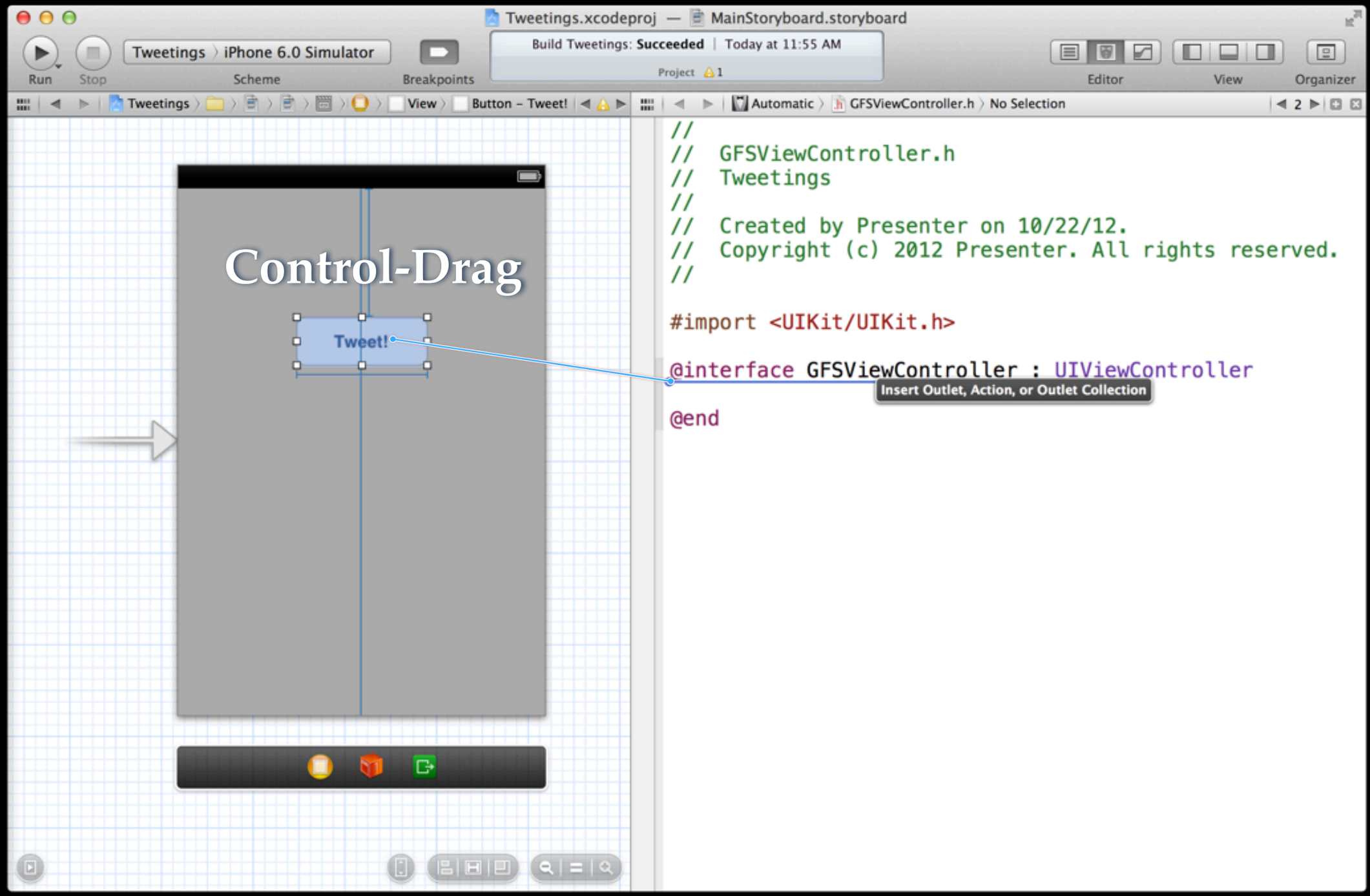
Cancel

Add


Tweetings.xcodeproj
Run
Stop
Tweetings 〉 iPhone 6.0 Simulator
Scheme
Breakpoints
Xcode
Editor
View
Organizer
Tweetings
Tweetings
1 target, iOS SDK 6.0
Social.framework
Tweetings
GFSAppDelegate.h
GFSAppDelegate.m
MainStoryboard.storyboard
GFSViewController.h
GFSViewController.m
Supporting Files
Frameworks
Products
PROJECT
Tweetings
TARGETS
Tweetings
Summary
Info
Build Settings
Build Phases
Build Rules
Target Dependencies (0 items)
Compile Sources (3 items)
Social.framework
Required
UIKit.framework
Foundation.framework
Required
CoreGraphics.framework
Required
Drag to reorder frameworks
Copy Bundle Resources (5 items)
Quick Help
Object Library
Push Button – Intercepts mouse-down events and sends an action message to a target object when it's clicked or...
Gradient Button – Intercepts mouse-down events and sends an action message to a target object when it's...
Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...
Add Target
Validate Settings
Add Build Phase


Tweet!
Objects
Round Rect Button – Intercepts touch events and sends an action message to a target object when it's tapped.
Bar Button Item – Represents an item on a UIToolbar or UINavigationItem object.
Fixed Space Bar Button Item – Represents a fixed space item on a UIToolbar object.
button


Tweetings.xcodeproj — MainStoryboard.storyboard
Run
Stop
Tweetings 〉 iPhone 6.0 Simulator
Scheme
Breakpoints
Build Tweetings: Succeeded | Today at 11:55 AM
Project 1
Organizer
Editor
Tweetings 〉 View 〉 Button – Tweet!
Automatic 〉 GFSViewCon
Tweet!
//
// GFSViewControlle
// Tweetings
//
// Created by Presenter on 10/22/12.
// Copyright (c) 2012 Presenter. All rights reserved.
//
#import <UIKit/UIKit.h>
@interface GFSViewController : UIViewController
Insert Outlet, Action, or Outlet Collection
@end


Tweetings.xcodeproj — MainStoryboard.storyboard
Run
Stop
Tweetings 〉 iPhone 6.0 Simulator
Scheme
Breakpoints
Build Tweetings: Succeeded | Today at 11:55 AM
Project 1
Editor
View
Organizer
Tweetings 〉 View 〉 Button – Tweet!
Automatic 〉 GFSViewController.h 〉 No Selection
Control-Drag
Tweet!
//
// GFSViewController.h
// Tweetings
//
// Created by Presenter on 10/22/12.
// Copyright (c) 2012 Presenter. All rights reserved.
//
#import <UIKit/UIKit.h>
@interface GFSViewController : UIViewController
Insert Outlet, Action, or Outlet Collection
@end

Tweetings.xcodeproj — MainStoryboard.storyboard

Run | Stop | Tweetings 〉 iPhone 6.0 Simulator (Scheme) | Breakpoints | Build Tweetings: **Succeeded** | Today at 11:55 AM | Project ⚠1 | Editor | View | Organizer

Tweetings 〉 View 〉 Button – Tweet!

Tweet!

Connection: Action
Object: View Controller
Name: showTweetSheet:
Type: id
Event: Touch Up Inside
Arguments: Sender
Cancel | Connect

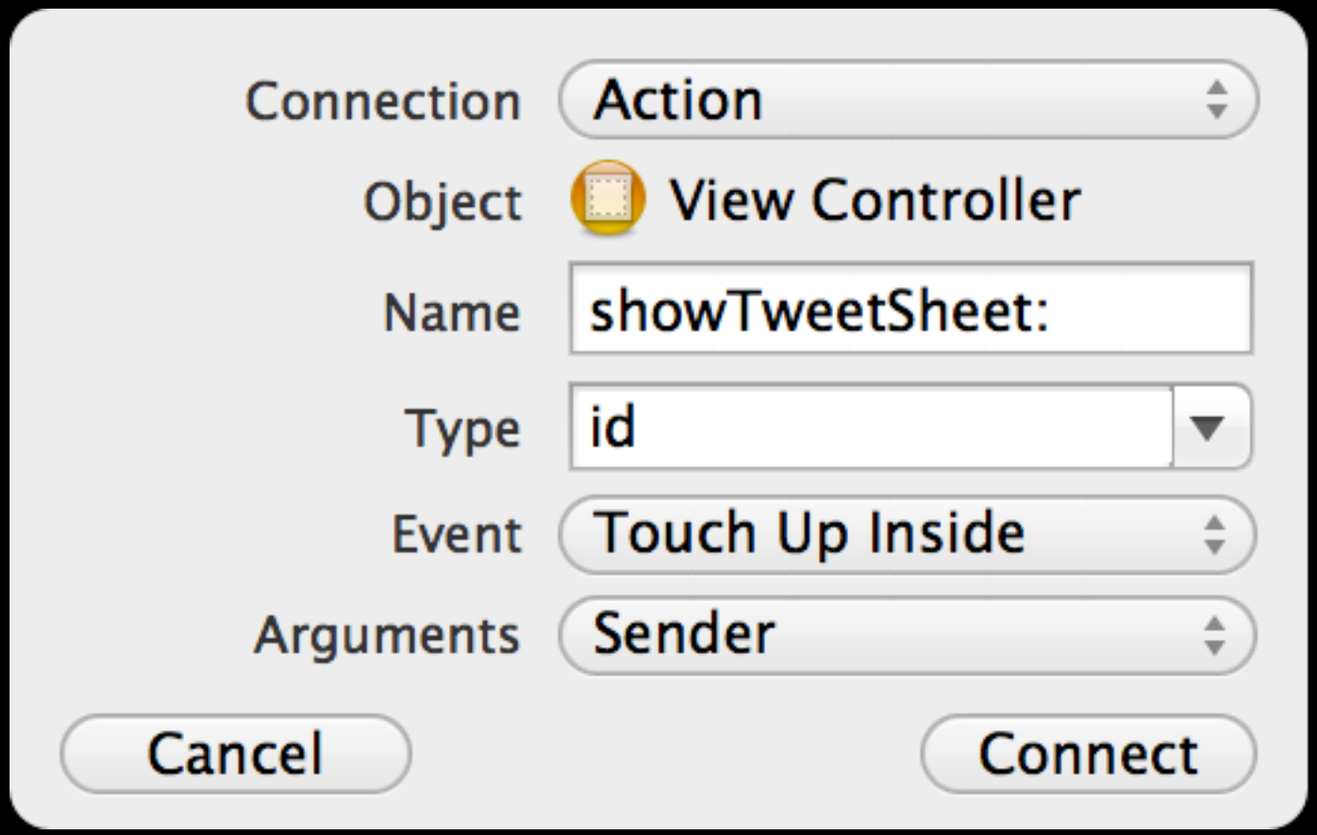
Automatic 〉 GFSViewController.h 〉 No Selection

```
//
//  GFSViewController.h
//  Tweetings
//
//  Created by Presenter on 10/22/12.
//  Copyright (c) 2012 Presenter. All rights reserved.
//

#import <UIKit/UIKit.h>

@interface GFSViewController : UIViewController

@end
```

Insert Outlet, Action, or Outlet Collection

Add Action

showTweetSheet:

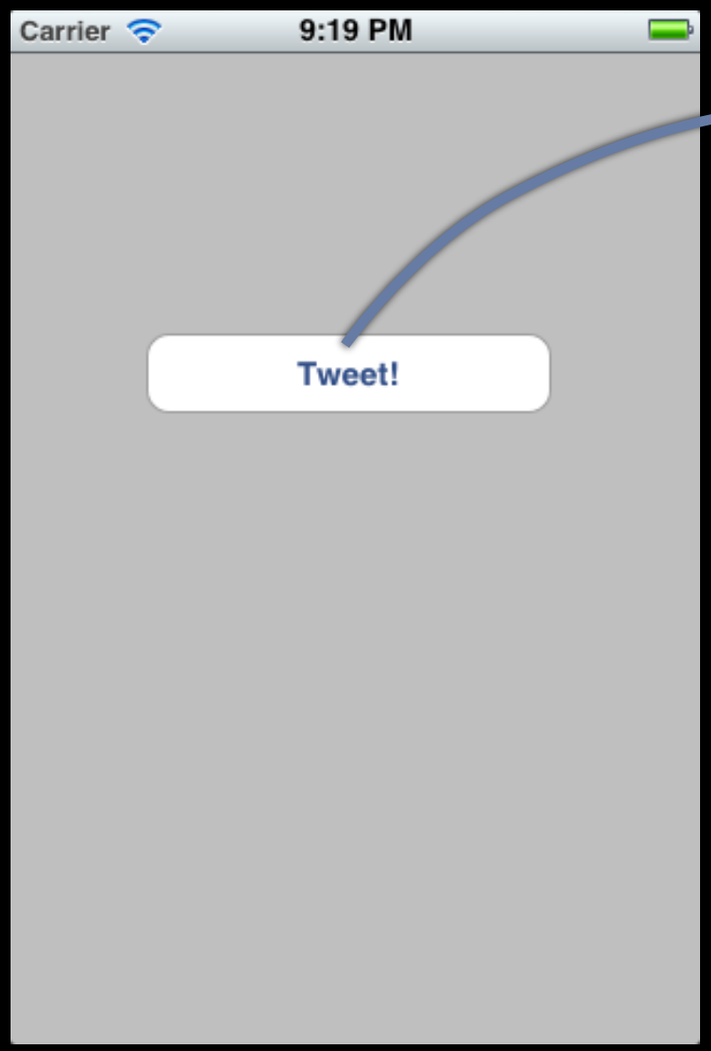


```objc
#import "GFSViewController.h"
#import <Social/Social.h>
...
@implementation GFSViewController
...
- (IBAction)showTweetSheet:(id)sender {

    SLComposeViewController *tweetVC =
            [SLComposeViewController
                composeViewControllerForServiceType:SLServiceTypeTwitter];

    tweetVC.initialText = @"I just finished the first project-#cocoaconf";

    [self presentViewController:tweetVC animated:YES completion:NULL];
}

...
@end
```

GFSViewController.m

Click Button - Invoke Code

Like a callback - only more explicit




```
-(IBAction)showTweetSheet:(id)sender {
  SLComposeViewController *tweetVC =[SLCo
  [tweetVC setInitialText:@"I just finish
  [self presentViewController:tweetVC ani
}
```

Live Code - Watch Me

Your Turn

- New Project—Single View Application
- Named ‘Tweetings’—Use Storyboards and ARC
- Add Social Framework
- Code the `showTweetSheet:` method
- Build Interface and Connect
- Run

Tweetings_01

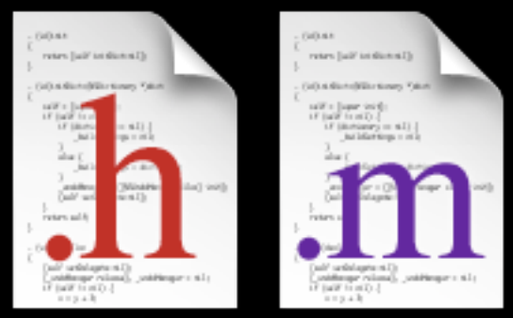
Recap

Tweetings_01

A lot to take in

Tools

Frameworks

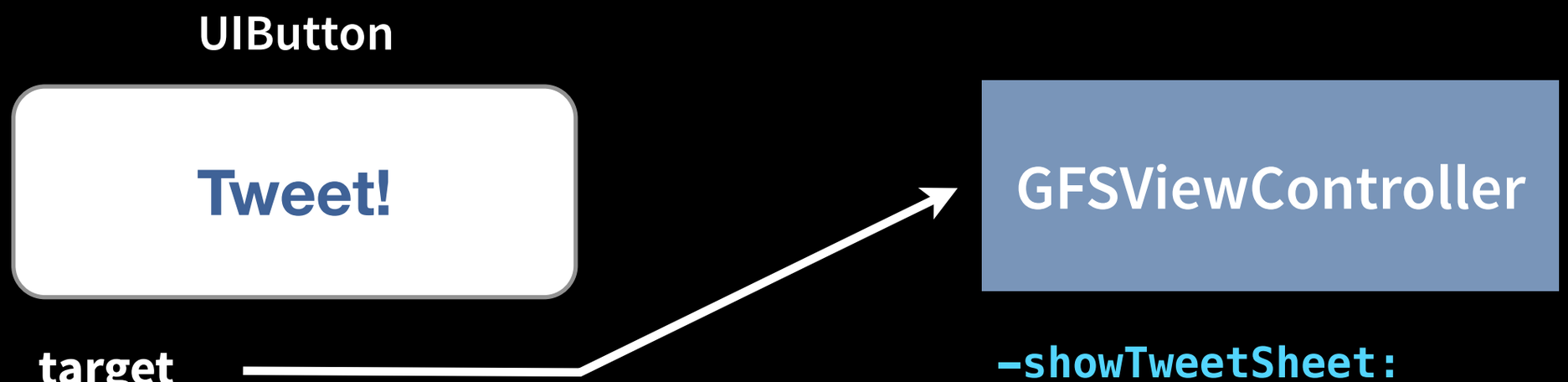


Language

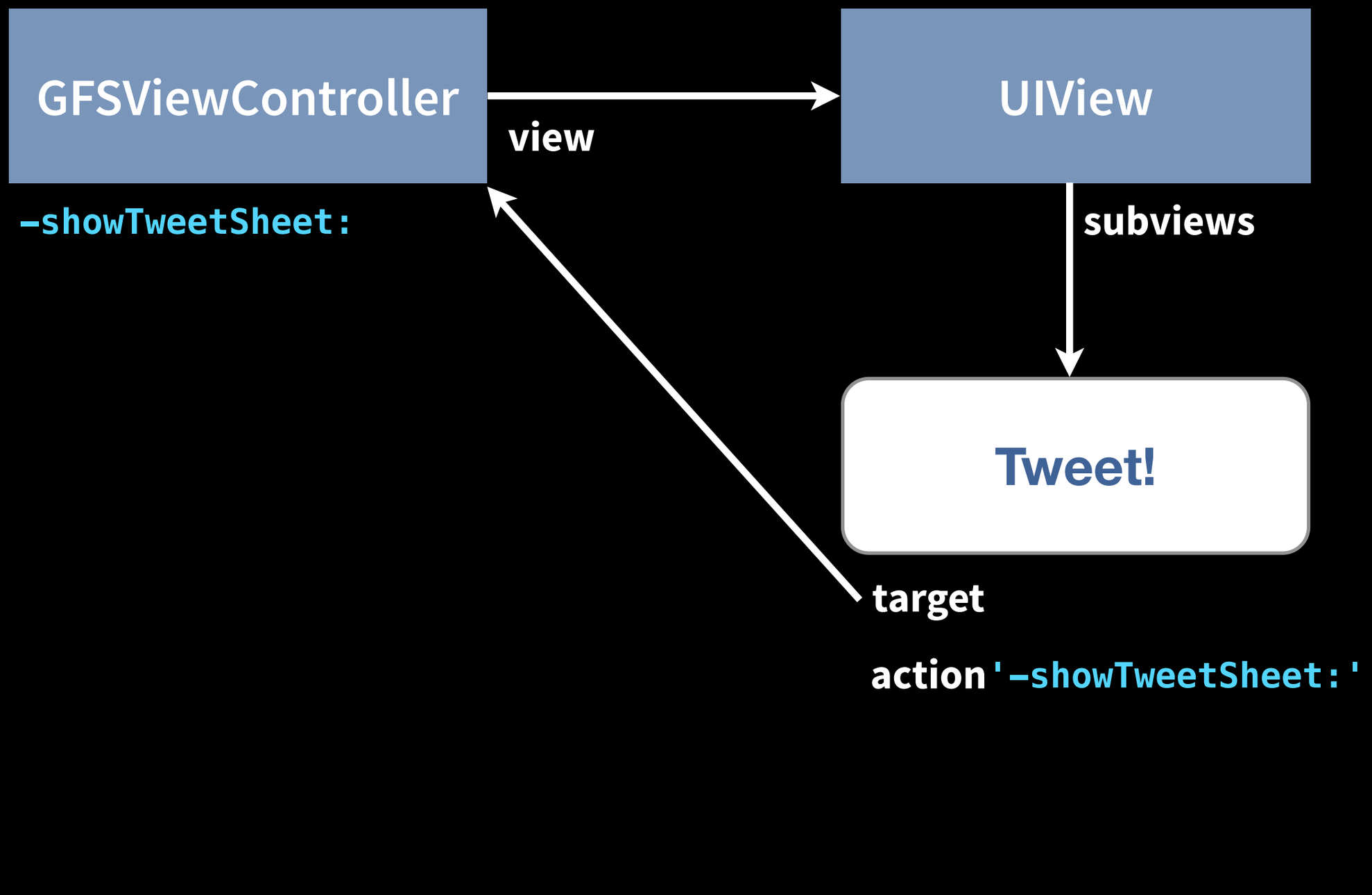
Building a network of objects

Target-Action

The view triggers action in the controller



Where we are now



View The Tweet

UIButton

View Tweets

target

action '-loadTweets:'

GFSViewController

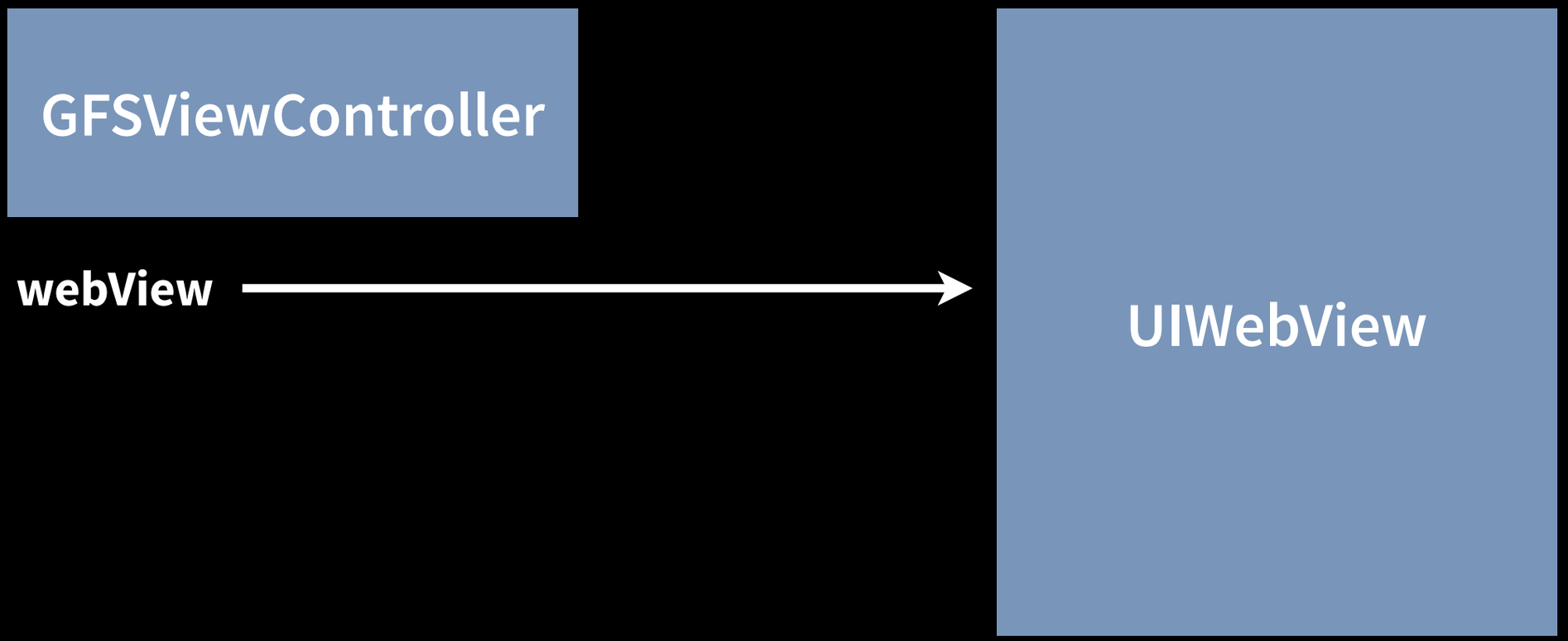
-(IBAction)loadTweets:

webView

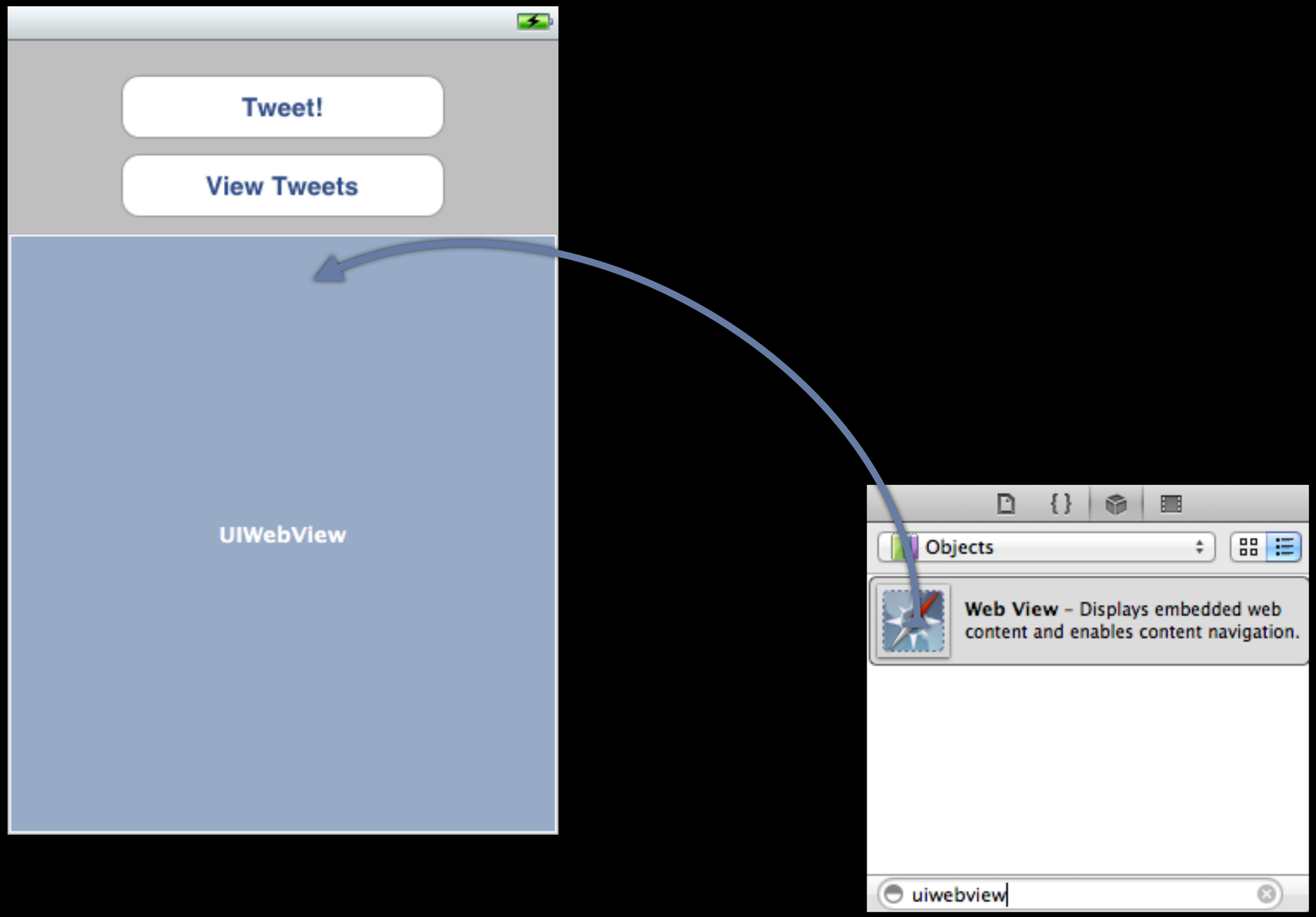
UIWebView

Outlets

The controller needs access to the view



Update the UI



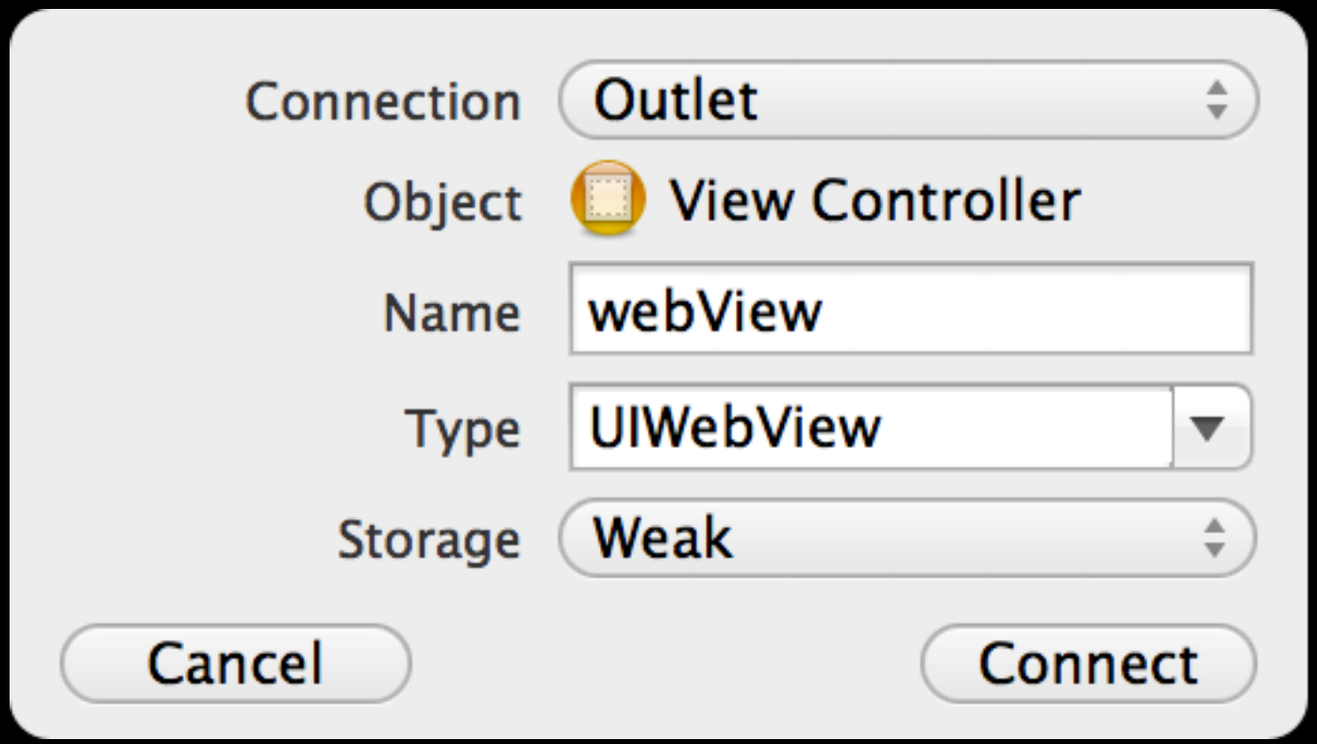
Two Ways To Make A Connection

- The Two-Step Method:
    1. Add the action / outlet in code
    2. Drag the connection in Interface Builder

- The One-Step Method:
    1. Drag from the interface to create the code and connect

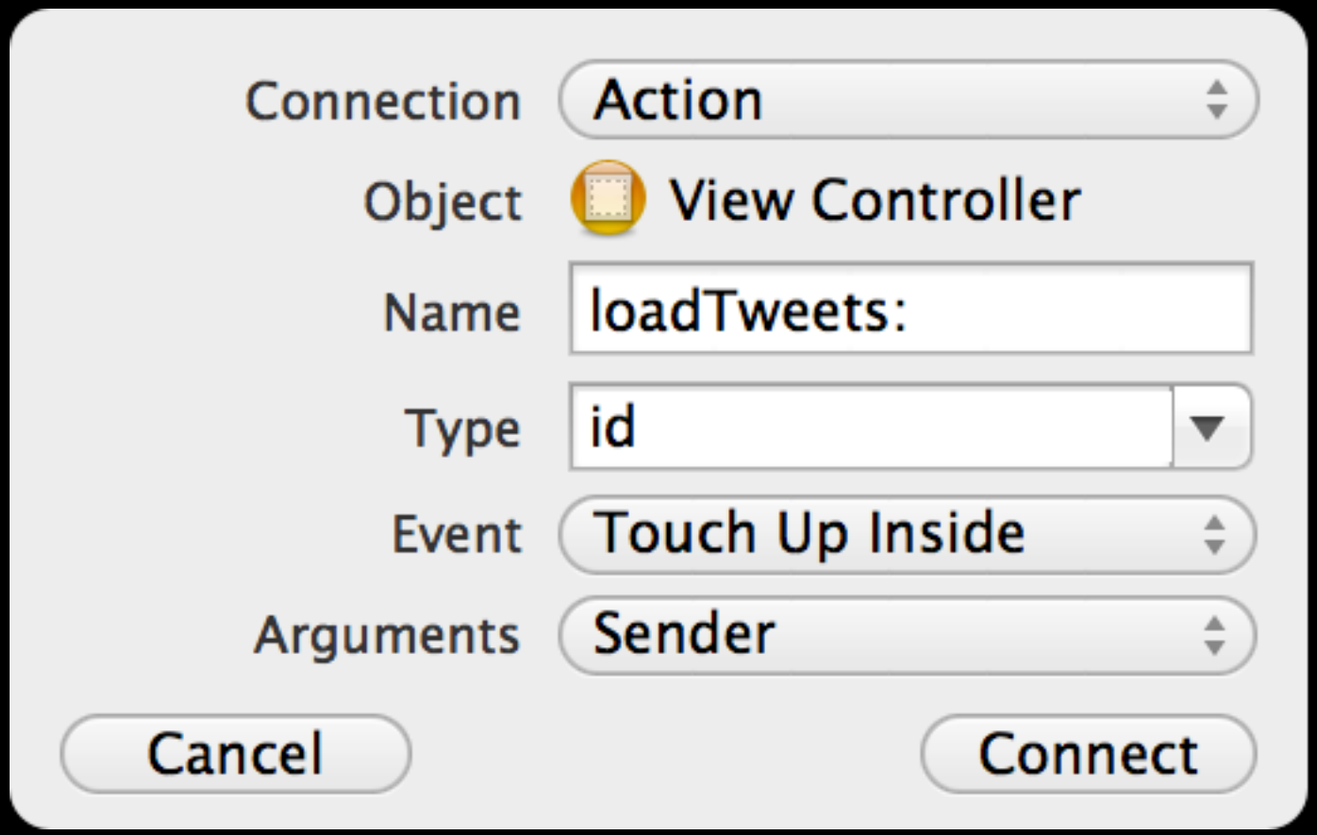
Add Outlet

webView



Add Action

loadTweets:



Loading Tweets

```
- (IBAction)loadTweets:(id)sender {
  NSURL *url = [NSURL URLWithString:@"http://twitter.com/TWITTER_HANDLE"];
  NSURLRequest *request = [NSURLRequest requestWithURL:url];
  [self.webView loadRequest:request];
}
```

GFSViewController.m

A reference to the web view

```
#import <UIKit/UIKit.h>


@interface GFSViewController : UIViewController

- (IBAction)showTweetSheet:(id)sender;
- (IBAction)loadTweets:(id)sender;

@property (nonatomic, weak) IBOutlet UIWebView *webView;

@end
```

GFSViewController.h

Live Code - Watch Me

Your Turn

- Add a UIWebView
- Add an IBOutlet property and make the connection
- Add code to load tweets
- Add a button and make the connection
- Run

Tweetings_01 ⟶ Tweetings_02

Objective-C

Objective-C

- A superset of C to add object-oriented features
- Inspired by SmallTalk
- Uses the character ‘@’ as a compiler directive

```
@interface

@implementation

@property

@"Literal Strings"

@32 // Literal number objects
```

Objective-C Source Files

A public interface, a private implementation

Header

Implementation

Getting to know Obj-C headers

Using UIWebView.h as an example

```
@class NSURLRequest;

@interface UIWebView : UIView

- (void)loadRequest:(NSURLRequest *)request;

@property(nonatomic,readonly,strong) NSURLRequest *request;

@end
```

Getting to know Obj-C headers

@interface declares the definition of a class

```
@class NSURLRequest;

@interface UIWebView : UIView

- (void)loadRequest:(NSURLRequest *)request;

@property(nonatomic,readonly,strong) NSURLRequest *request;

@end
```

Getting to know Obj-C headers

@class tells compiler this class is defined elsewhere

```
@class NSURLRequest;

@interface UIWebView : UIView

- (void)loadRequest:(NSURLRequest *)request;

@property(nonatomic,readonly,strong) NSURLRequest *request;

@end
```

Getting to know Obj-C headers

Method declarations

```
@class NSURLRequest;

@interface UIWebView : UIView

- (void)loadRequest:(NSURLRequest *)request;

@property(nonatomic,readonly,strong) NSURLRequest *request;

@end
```

Method Declarations

```
- (void)loadRequest:(NSURLRequest *)request;
```

Return type

method name

parameter type

parameter

Sending messages

- UIWebView.h

```
@interface UIWebView : UIView

- (void)loadRequest:(NSURLRequest *)request;

@end
```

- In use

```
UIWebView *webView = ... // assume this exists
NSURLRequest *request = ... // assume this exists

[webView loadRequest: request];
```

Objects and Messages

Objects are nouns, messages are verbs

The brackets mean you are sending a message

```
[self.webView loadRequest:request];
```

The receiver

The message

A parameter

Multiple Parameters

A colon per parameter—order is fixed

- UIViewController.h

```
@interface UIViewController : UIResponder

- (void)presentViewController:(UIViewController *)vc
                     animated: (BOOL)flag
                   completion:(void (^)(void))completion;

@end
```

- In use

```
[self presentViewController:tweetVC
                   animated:YES
                 completion:NULL];
```

Class methods

- UIImage.h

```
@interface UIImage : NSObject

+ (UIImage *)imageNamed:(NSString *)name;

- (void)drawAtPoint:(CGPoint)point;

@end
```

- In use

```
UIImage *image = [UIImage imageNamed:@”Cookies”];

[image drawAtPoint:CGPointMake(100.0, 100.0)];
```

Instance methods

- UIImage.h

```
@interface UIImage : NSObject

+ (UIImage *)imageNamed:(NSString *)name;

- (void)drawAtPoint:(CGPoint)point;

@end
```

- In use

```
UIImage *image = [UIImage imageNamed:@”Cookies”];

[image drawAtPoint:CGPointMake(100.0, 100.0)];
```

Creating new objects

- NSObject.h

```
@interface NSObject

+ (id)alloc;

- (id)init;

@end
```

- In use

```
MyObject *object = [[MyObject alloc] init];
```

Creating new objects

- NSURL.h

```
@interface NSURL : NSObject

- (id)initWithString:(NSString *)URLString;

+ (id)URLWithString:(NSString *)URLString;

@end
```

- In use

```
NSURL *appleURL = [[NSURL alloc] initWithString:@ "http://www.apple.com"];

NSURL *twitterURL = [NSURL URLWithString:@"http://www.twitter.com"];
```

Creating new objects

- NSURL.h

```
@interface NSURL : NSObject

- (id)initWithString:(NSString *)URLString;

+ (id)URLWithString:(NSString *)URLString;

@end
```

- In use

```
NSURL *appleURL = [[NSURL alloc] initWithString:@ "http://www.apple.com"];

NSURL *twitterURL = [NSURL URLWithString:@"http://www.twitter.com"];
```

Contrast with other languages

- Other languages often similar to C function calls

```
NSURL.URLWithString("http://twitter.com/");

webview.loadRequest(request);

this.presentViewController(viewController, YES, NULL);
```

- Objective-C adopts a Small Talk-style approach

```
[NSURL URLWithString:@"http://www.twitter.com"];

[webView loadRequest:request];

[self presentViewController:tweetVC
                   animated:YES
                 completion:NULL];
```

Getting to know Obj-C headers

Property declarations

```
@class NSURLRequest;

@interface UIWebView : UIView

- (void)loadRequest:(NSURLRequest *)request;

@property(nonatomic,readonly,strong) NSURLRequest *request;

@end
```

Properties

```
@property (strong, nonatomic) UIWindow *window;
```

Property Compiler Directive

Property Attributes

Type

Name

Accessing Properties

- UIWebView.h

```
@interface UIWebView : UIView

@property(nonatomic,readonly,strong) NSURLRequest *request;

@end
```

- In use

```
NSURLRequest *aRequest = webView.request;
```

Accessing Properties

- GFSViewController.h

```
@interface GFSViewController : UIViewController

@property (weak, nonatomic) IBOutlet UIWebView *webView;

@end
```

- In use

```
UIWebView *wv = self.webView;

[self.webView loadRequest:request];

self.webView = [[UIWebView alloc] initWithFrame:CGZeroRect()];
```

Properties as outlets

Extra keyword for the benefit of Interface Builder

```
@property (nonatomic, weak) IBOutlet UIWebView *webView;
```

Property Compiler Directive

Property Attributes

Outlet Identifier

Type

Name

Writing a new class

Define the interface, then implement it

Header

Implementation

Declare the class in the .h file

@interface

```
#import <UIKit/UIKit.h>


@interface GFSViewController : UIViewController

@end
```

GFSViewController.h

Implement in the .m file

Import the interface header file

```
#import "GFSViewController.h"


@implementation GFSViewController

@end
```

GFSViewController.m

Adding methods

The interface makes a promise

```
#import <UIKit/UIKit.h>


@interface GFSViewController : UIViewController

- (void)loadTweets:(id)sender;

@end
```

GFSViewController.h

Adding methods

The implementation delivers

```
#import "GFSViewController.h"



@implementation GFSViewController

- (void)loadTweets {
  NSURL *url = [NSURL URLWithString:@"http://twitter.com"];
  NSURLRequest *request = [NSURLRequest requestWithURL:url];
  [self.webView loadRequest:request];
}

@end
```

GFSViewController.m

Adding properties

A property is a piece of accessible state

```
#import "GFSViewController.h"


@interface GFSViewController

@property (nonatomic, weak) IBOutlet UIWebView *webView;

@end
```

GFSViewController.m

Adding properties

A property declaration implies a number of things

- This declaration:

```
@property (nonatomic, weak) IBOutlet UIWebView *webView;
```

- By default, directs the compiler to do the following:
  - Generate accessor methods for property
    - `- (UIWebView *)webView;`
    - `- (void)setWebView:(UIWebView *)view;`
  - Allocate storage for the property

GFSViewController.m

Implementing properties

- By default, the compiler handles everything

- Can customize what the compiler does in .m file
  - @synthesize
  - @dynamic

- Can write your own accessor methods
  - Often when providing a value dynamically

- Property attributes control what the compiler does

Property Attributes

	Memory Management	Concurrency	Access
Defaults*	strong	atomic	readwrite
	weak	nonatomic	readonly
	copy		

*strong with Automatic Reference Counting, weak with manual)

Using properties and methods

- You can use dot syntax to access properties

```
UIWebView *aWebView = self.webView;
self.webView = aWebView;
```

- Use square brackets to send messages

```
[viewController sendTweet];
```

- Dot syntax is just a shortcut for the underlying messages

```
UIWebView *aWebView = [self webView];
[self setWebView:aWebView];
```

- You can nest messages and combine the two forms

```
[self.webView loadRequest:request];
```

Class Extension

Allows a private interface in the implementation file

```
#import "GFSViewController.h"

@interface GFSViewController ()

@end
```

```
@implementation GFSViewController

- (void)loadTweets {
  NSURL *url = [NSURL URLWithString:@"http://twitter.com"];
  NSURLRequest *request = [NSURLRequest requestWithURL:url];
   [self.webView loadRequest:request];
}

@end
```

GFSViewController.m

Class Extension

Outlets are good candidates for class extensions

```
#import "GFSViewController.h"

@interface GFSViewController ()

@property (nonatomic, weak) IBOutlet UIWebView *webView;

@end


@implementation GFSViewController

- (void)loadTweets {
  NSURL *url = [NSURL URLWithString:@"http://twitter.com"];
  NSURLRequest *request = [NSURLRequest requestWithURL:url];
  [self.webView loadRequest:request];
}

@end
```

GFSViewController.m

Foundation Classes

- NSString
- NSNumber
- NSDate
- NSURL

NSString

- All the strings in your app

- Provides powerful functionality
  - Unicode support
  - Combining and Dividing
  - Searching and Replacing

- String literals are NSStrings not C strings
  - `@"This is a string literal"`

NSNumber

- An object wrapper for C number types
- Use regular C numbers for calculation

- Number literals
  - @45
  - @90.6
  - @YES

NSDate

- Represents a single point in time
- Use with NSCalendar for date calculations
- Use with NSDateFormatter for string representation

- Getting the current date

```
NSDate *now = [NSDate date];
```

NSURL

- Represents a URL
- Used both for network and local file resources

- Creating from a string

```
NSURL *url = [NSURL URLWithString:@"http:www.apple.com"];
```

More in Foundation

- Key Value Coding
- Key Value Observing
- Serialization
- File Handling
- Operations and Threads
- Bundles

Logging with NSLog

“Cave Man Debugging”

```
NSLog(@"My hover craft is full of eels!");

NSString *character = @"Bob";
NSLog(@"%@'s hover craft is full of eels!", character);

GFSRecipe *recipe = [[GFSRecipe alloc] init];
recipe.name = @"Cookie";
recipe.preparationTime = [NSNumber numberWithInt:15];
recipe.directions = @"mix that stuff with the other stuff";
NSLog(@"The recipe: %@", recipe);
```

-description

String description of an object

```
- (NSString *)description {
  return [NSString stringWithFormat:@"%@ - %@ - %@",
     self.name,
          self.preparationTime,
          [[self.directions substringToIndex:10]
                            stringByAppendingString:@"..."]];
}
```

Application Delegate

App delegate is called when app is finished launching

```
- (BOOL)application:(UIApplication *)application
     didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {

    // Override point for customization after application launch.

    return YES;
}
```

GFSAppDelegate.m

Live Code - Watch Me

Your Turn

- New project—Recipes
- Use ARC and Storyboards
- Create a class named ‘Recipe’
- Add three properties
- Create an instance in `application:didFinishLaunchingWithOptions:`
- Log it to the console

More Objective-C

App Delegate Revisited

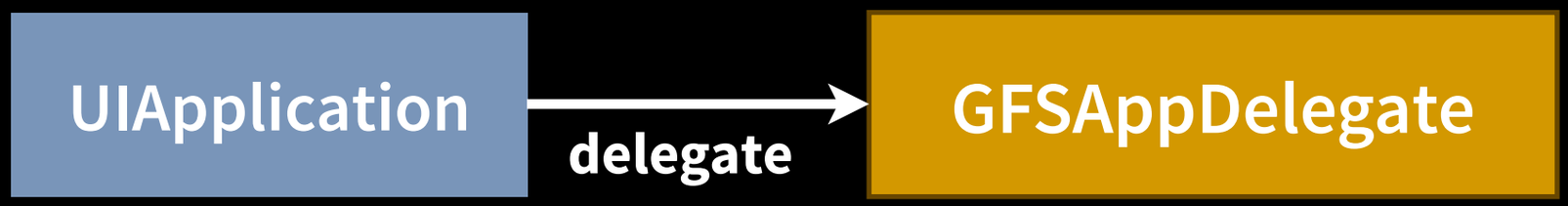
A delegate works together with another object

- Main object calls out to its delegate at set control points
  - Opportunity to add behavior

    `-application:didFinishLaunchingWithOptions:`
  - Opportunity to modify default behavior

    `-application:shouldSaveApplicationState:`

- Delegate methods are declared as a protocol



Objective-C Protocols

Defined coordination between objects

- A named list of method declarations
- Methods can be marked as required or optional
- A class can conform to a protocol by
  - Declaring it conforms to the protocol
  - Implementing all required methods
  - Implementing optional methods, if desired

The Protocol Definition

Just a list of methods

```
@protocol UIApplicationDelegate <NSObject>

@optional

-(BOOL)application:(UIApplication *)app
      willFinishLaunchingWithOptions:(NSDictionary *)options;

-(BOOL)application:(UIApplication *)app
       didFinishLaunchingWithOptions:(NSDictionary *)options;

-(void)applicationDidBecomeActive:(UIApplication *)app;

-(void)applicationDidResignActive:(UIApplication *)app;

-(void)applicationDidEnterBackground:(UIApplication *)app;

@end
```

Conforming to a protocol

Declare conformance in the interface declaration

```
@interface GFSAppDelegate : UIResponder <UIApplicationDelegate>

...

@end
```

- Multiple protocols can be declared

```
@interface MyClass : NSObject <NSCoding, NSCopying>

...

@end
```

Conforming to a protocol

Implement required methods in implementation

- Delegate methods are optional, implement what you need

```
@implementation GFSAppDelegate

- (BOOL)application:(UIApplication *)application
    didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    // Override point for customization after application launch.
    return YES;
}

@end
```

- Other protocols have required methods

Using protocol as a type

Compiler will check conformance to protocol

- As a property

```
@interface UIApplication : UIResponder

@property(nonatomic,weak) id<UIApplicationDelegate> delegate;

@end
```

- In method signatures

```
- (id<UIApplicationDelegate>)delegate;

- (void)setDelegate:(id<UIApplicationDelegate>)delegate;
```

Foundation Collections

- NSArray
- NSDictionary
- NSSet

NSArray

- Ordered list of objects

```
NSArray *recipes = [NSArray arrayWithObjects:cookie, bread, stew, nil];

NSArray *recipes = @[cookie, bread, stew];


for(GFSRecipe *recipe in recipes) {
  NSLog(@"recipe = %@", recipe);
}


NSInteger arrayCount = [recipes count];

for(int i = 0; i < arrayCount; i++) {

  GFSRecipe *recipe = [recipes objectAtIndex:i];

  NSLog(@"recipe = %@", recipe);
}
```

NSDictionary

- Values stored by key
- Keys are unique

```
NSDictionary *recipes = [NSDictionary dictionaryWithObjectsAndKeys:
                                                    cookie, @"cookie",
                                                    bread, @"bread",
                                                    stew, @"stew", nil];

NSArray *recipes = @{@"cookie": cookie, @"bread": bread, @"stew": stew};

for(NSString *key in recipes) {

  GFSRecipe *recipe = [recipes objectForKey:key];

  NSLog(@"recipe = %@", recipe);
}
```

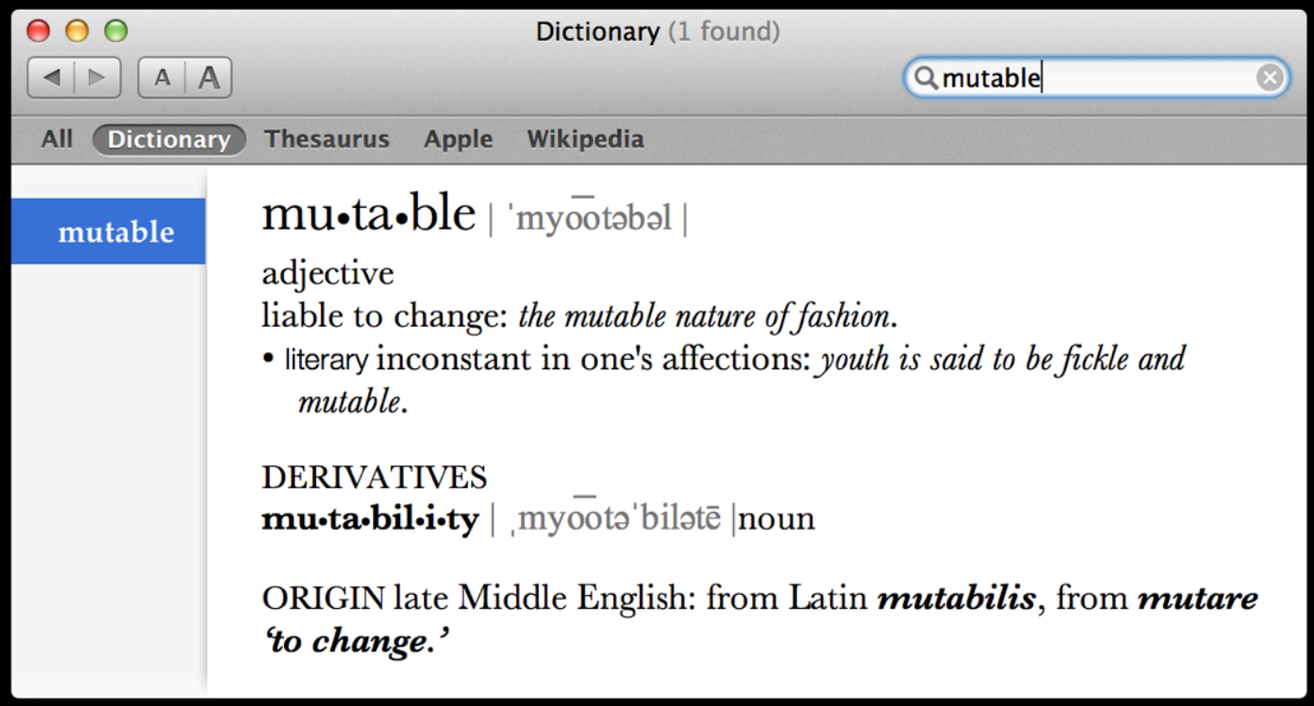
NSSet

- Unordered collection of unique objects

```
NSSet *recipes = [NSSet setWithObjects:cookie, bread, stew, nil];

for(GFSRecipe *recipe in recipes) {
  NSLog(@"recipe = %@", recipe);
}
```

Immutable with a Mutable Subclass

Mutable is fancy-talk for ‘can be changed’




```
@interface NSMutableArray : NSArray

@interface NSMutableDictionary : NSDictionary

@interface NSMutableSet : NSSet
```

Mutable Collections

```
NSMutableArray *recipes = [NSMutableArray array];
[recipes addObject:cookie];
[recipes removeObjectAtIndex:0];
[recipes insertObject:pie atIndex:0];
```

```
NSMutableDictionary *recipes = [NSMutableDictionary dictionary];
[recipes setObject:cookie forKey:@"cookie"];
[recipes removeObjectForKey:@"cookie"];
```

```
NSMutableSet *recipes = [NSMutableSet set];
[recipes addObject:cookie];
[recipes removeObject:cookie];
```

Live Code - Watch Me

Your Turn

- Create three recipes
- Add them to an array, a dictionary, and a set
- Log the collections to the console
- Change the collections to mutable
- Remove one object from each
- Log the modified collection

Build the Recipes App

Model View Controller

Model

Controller

View

Model View Controller

Model

Controller

View

App Data
App Logic

Model View Controller

Model

Controller

View

User Visible
Events

Model View Controller

Model

Controller

View

Intermediary
Conduit
“Glue”

Model View Controller

Interface boundaries encapsulate behavior

Application Definition Statement

“A concise, concrete declaration of an app’s main purpose and its intended audience.”

Apple: Human Interface Guidelines

Example: Photos on iOS

Main Purpose: Photos is an easy to use digital **photo editing, organizing, and sharing application** for casual and amateur photographers.

Intended Audience: Photos is an easy to use digital photo editing, organizing, and sharing application **for casual and amateur photographers.**

Our Example: Recipes

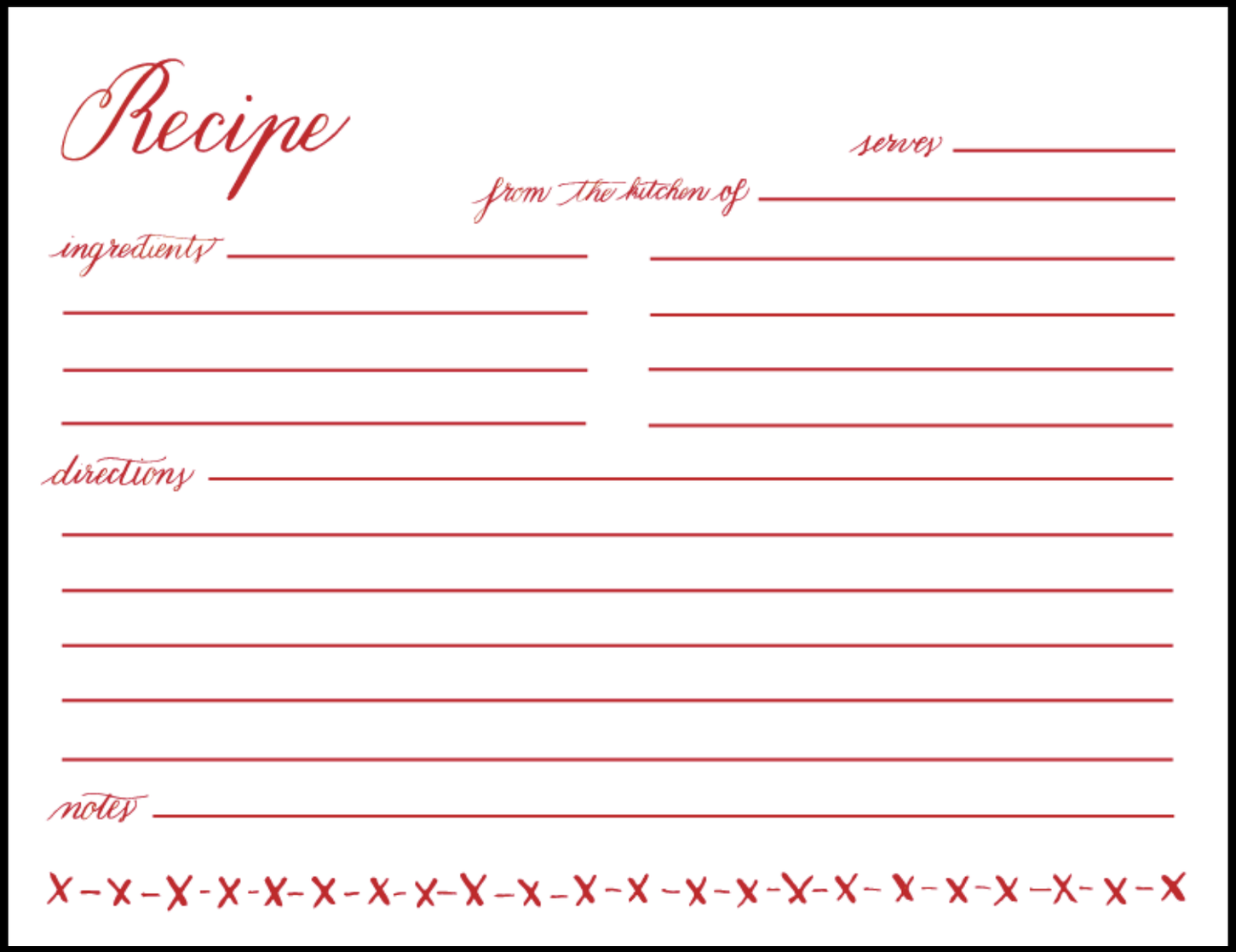
Main Purpose: Recipes is a **simple recipe editing application for casual and amateur cooks.**

Intended Audience: Recipes is a simple recipe editing application **for casual and amateur cooks.**

Recipes

http://jdorganizer.blogspot.com/2011/02/5-recipe-boxes-to-jazz-up-any-kitchen.html

Model



http://aviedesigns.blogspot.com/2010/07/mamies-recipe-card-free-download.html

Model

```
#import <Foundation/Foundation.h>

@interface GFSRecipe : NSObject

@property(nonatomic, copy) NSString *name;
@property(nonatomic, copy) NSString *directions;
@property(nonatomic, copy) NSNumber *preparationTime;
@property(nonatomic, strong) UIImage *image;

@end
```

GFSRecipe.h

Model

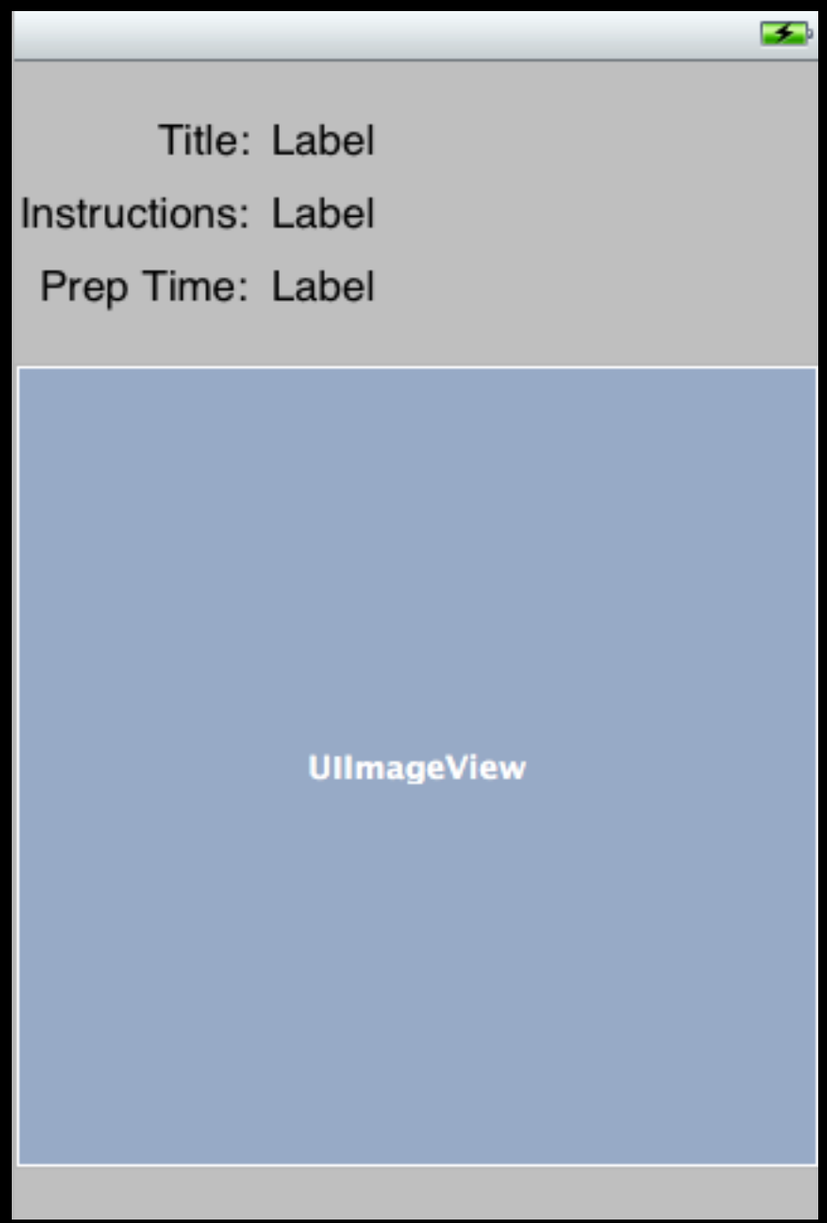
```
#import "GFSRecipe.h"

@implementation GFSRecipe

- (NSString *)description {
    return [NSString stringWithFormat:@"Recipe: name = %@",
                                                    self.name];
}

@end
```

GFSRecipe.m

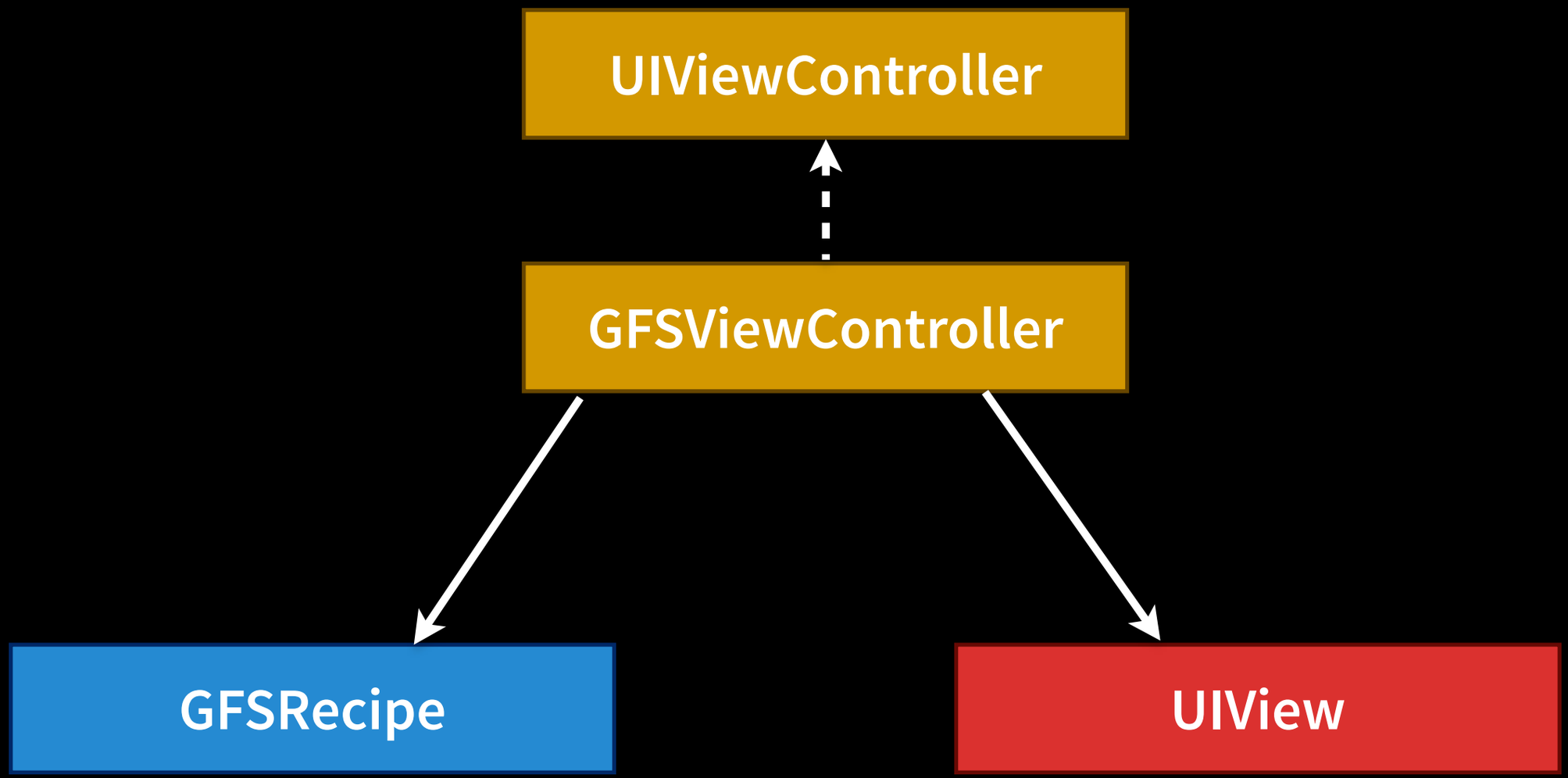
Model

A collection of recipes

View


Title: Label
Instructions: Label
Prep Time: Label
UIImageView

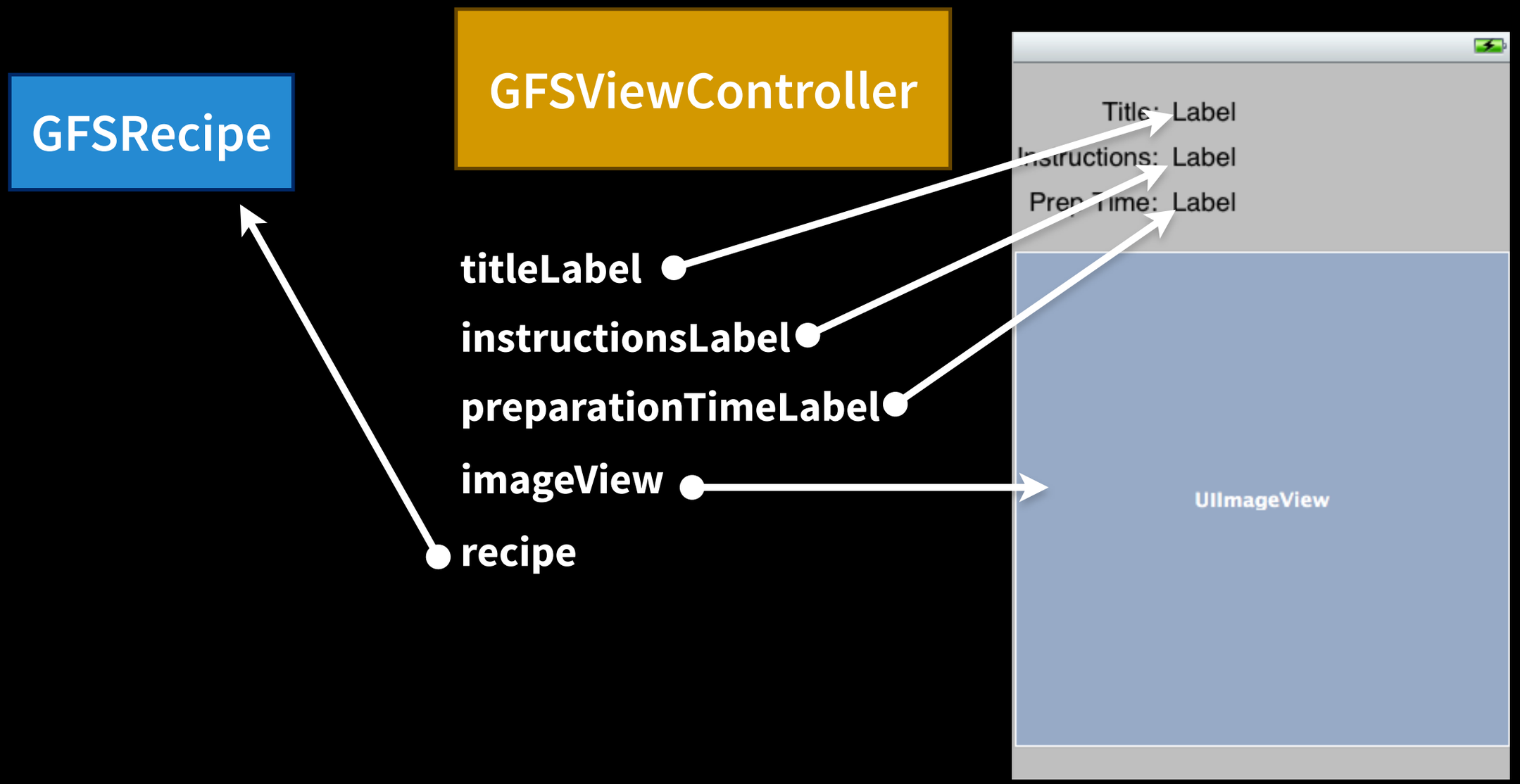
View Controller



View Controllers

- Manages a view and its subviews
  - On iPhone think “one screen worth of content”
  - On iPad think “one split pane / popover worth of content”
- Navigation
- Orientation
- Memory/Resource Management

Model View Controller



Controller

```
#import <UIKit/UIKit.h>
#import "GFSRecipe.h"

@interface GFSViewController : UIViewController

@property(nonatomic, weak) IBOutlet UILabel *nameLabel;
@property(nonatomic, weak) IBOutlet UILabel *directionsLabel;
@property(nonatomic, weak) IBOutlet UILabel *preparationTimeLabel;
@property(nonatomic, weak) IBOutlet UIImageView *imageView;

@property(nonatomic, strong) GFSRecipe *recipe;

@end
```

GFSViewController.h

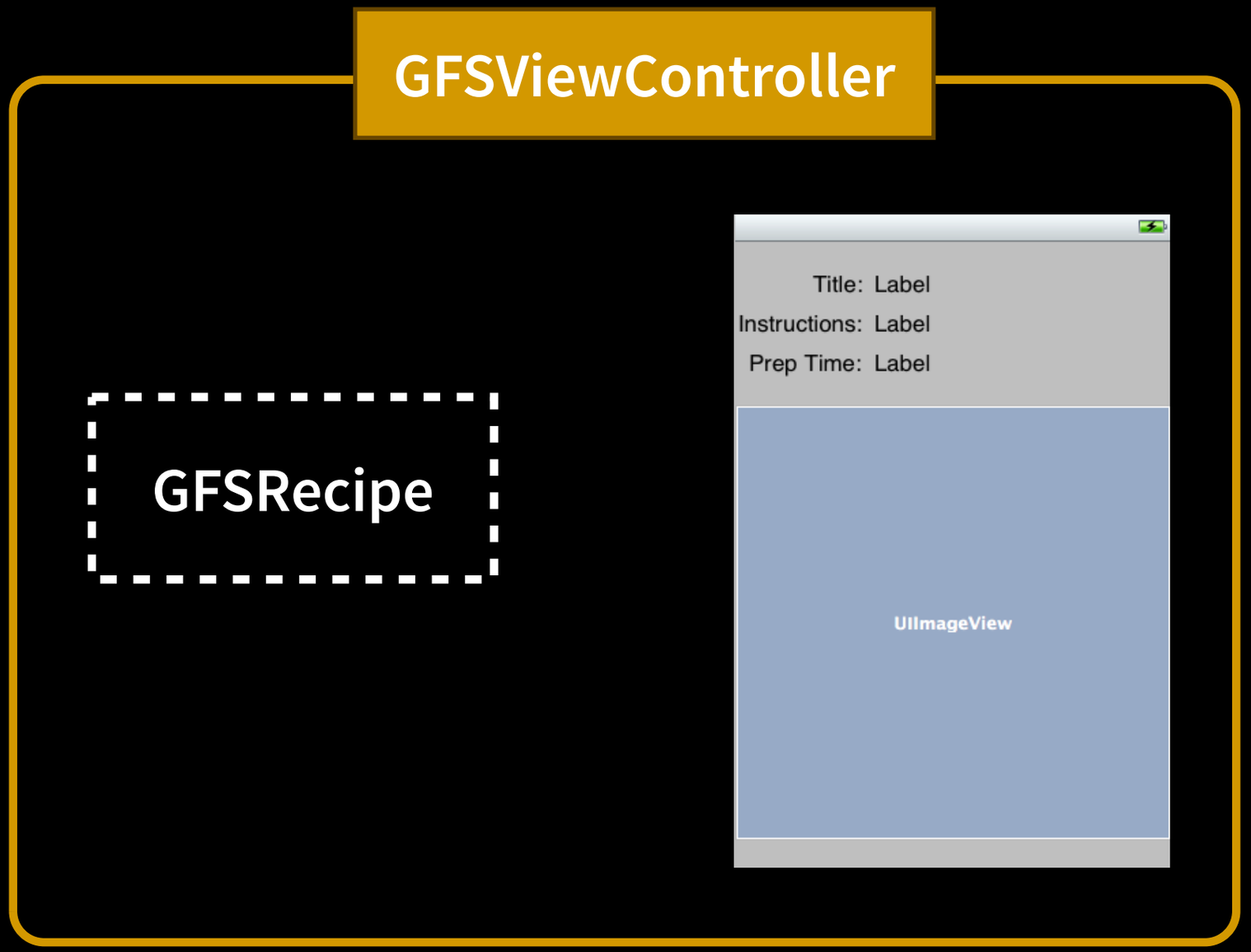
Controller

```
- (void)viewDidLoad {
  [super viewDidLoad];
  self.nameLabel.text = self.recipe.name;
  self.directionsLabel.text =  self.recipe.directions;

  self.preparationTimeLabel.text =  self.recipe.preparationTime;

  if(nil != self.recipe.image) {
    self.imageView.image = self.recipe.image;
  }

}
```

GFSViewController.h

Encapsulation

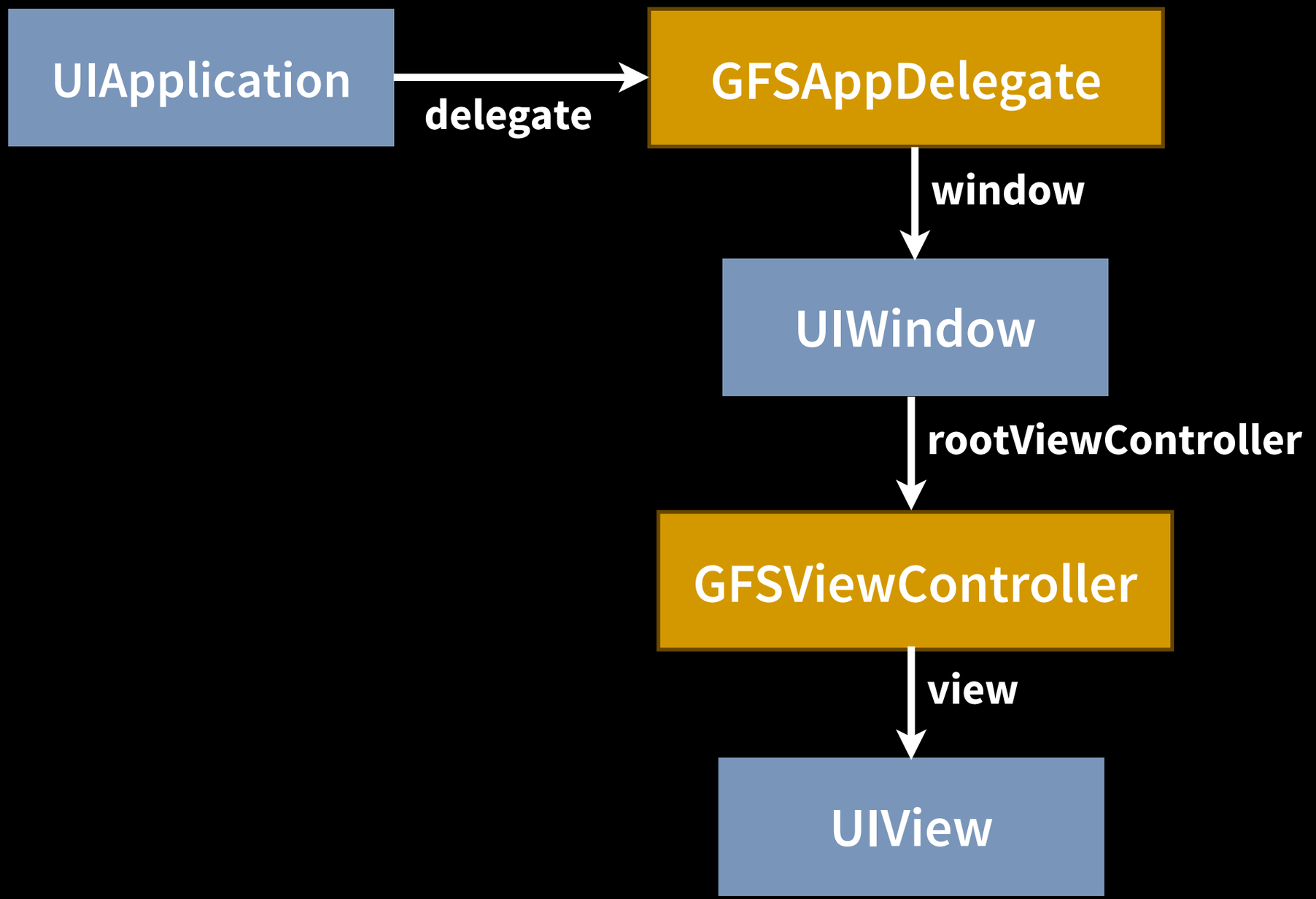
A view controller that can handle a single recipe



Where do we get the recipe?

The app delegate will provide it

One of the key objects created on app launch



Application Delegate

```
- (BOOL)application:(UIApplication *)application
            didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {

  NSString *directions = @"Put the flour and other dry ingredients in a bowl, \
  stir in the eggs until evenly moist. Add chocolate chips and stir in untileven. \
  Place tablespoon sized portions on greased cookie sheet and bake at 350° for \
  6 minutes.";

  GFSRecipe *recipe = [[GFSRecipe alloc] init];
  recipe.title = @"Chocolate Chip Cookies";
  recipe.preparationTime = @15;
  recipe.directions = directions;
  recipe.image = [UIImage imageNamed:@"cookies.jpg"];


  GFSViewController *viewController =
                          (GFSViewController *)self.window.rootViewController;
  viewController.recipe = recipe;

  return YES;
}

@end
```

GFSAppDelegate.m

Live Code - Watch Me

Your Turn

- Create the interface to display a recipe's info
- Add outlets to view controller and make connections
- Add recipe property to view controller
- In App Delegate, create a recipe and hand it to view controller
- Run the app

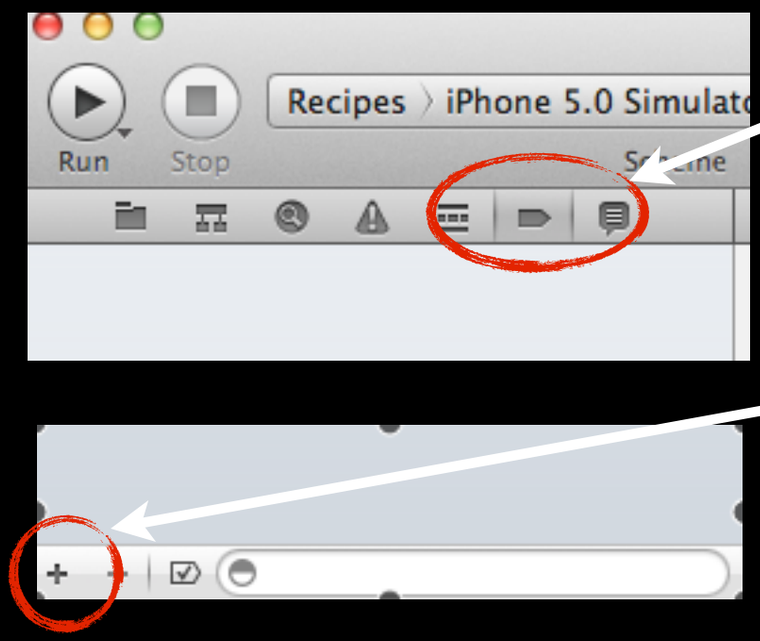
- Note: You can comment out, delete, or keep the practice code already in the App Delegate.

Xcode Debugger

For debugging, but also good for exploring

Figuring it out for yourself

Symbolic breakpoints



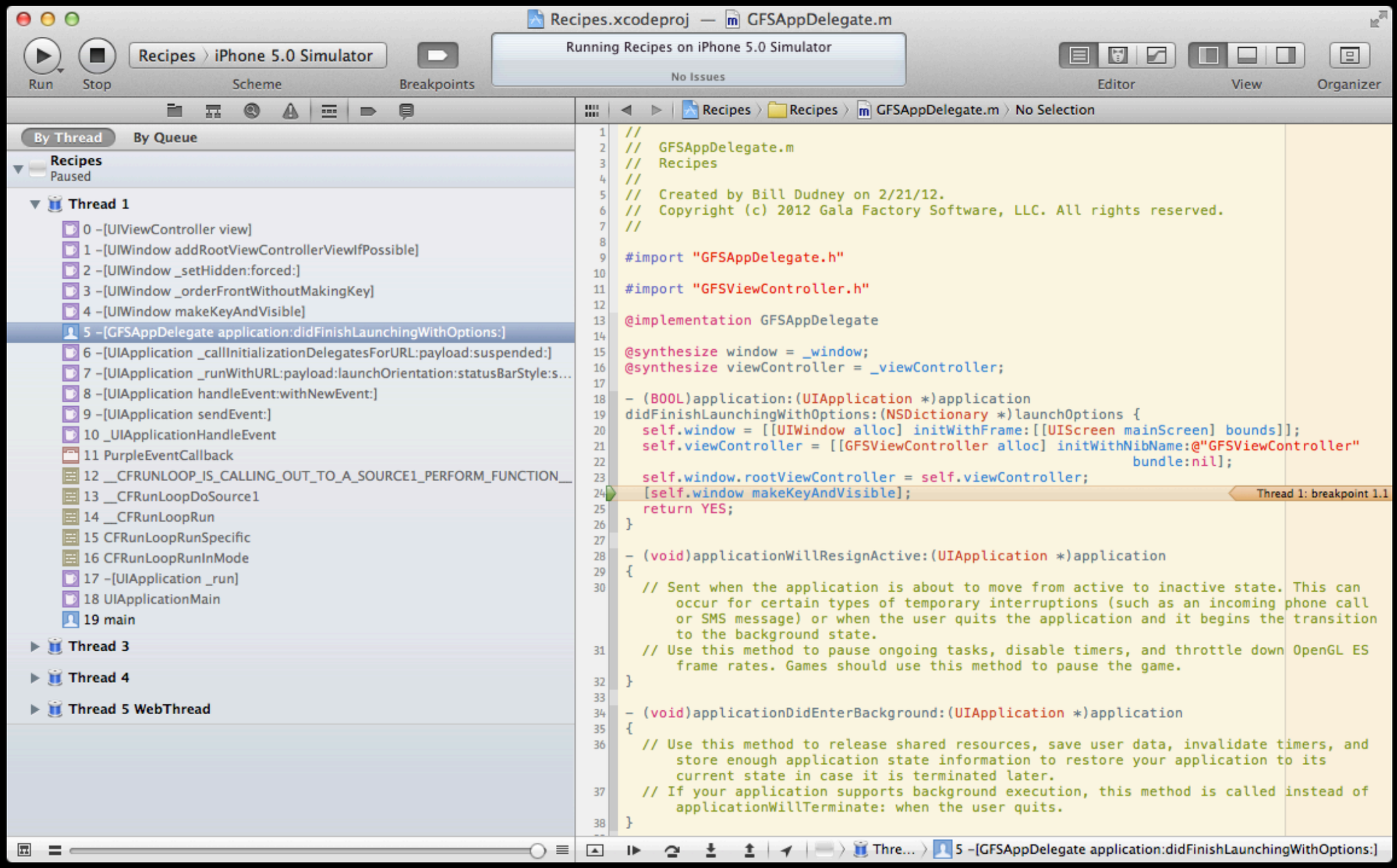
Breakpoint Navigator

Add Symbolic Breakpoint

-[UIViewController view]

Figuring it out for yourself

Symbolic breakpoints

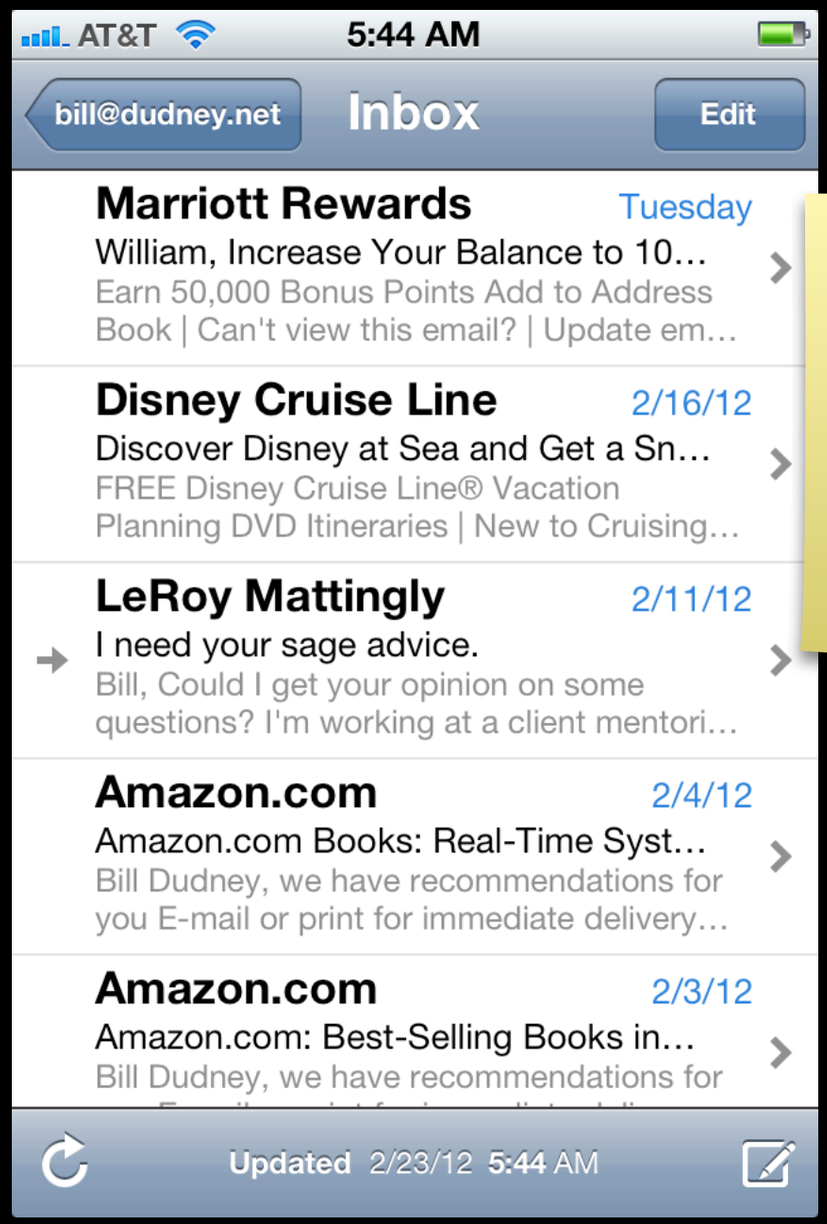


Watch Me

Table Views

Table views

An iOS app workhorse



Take your own screenshots

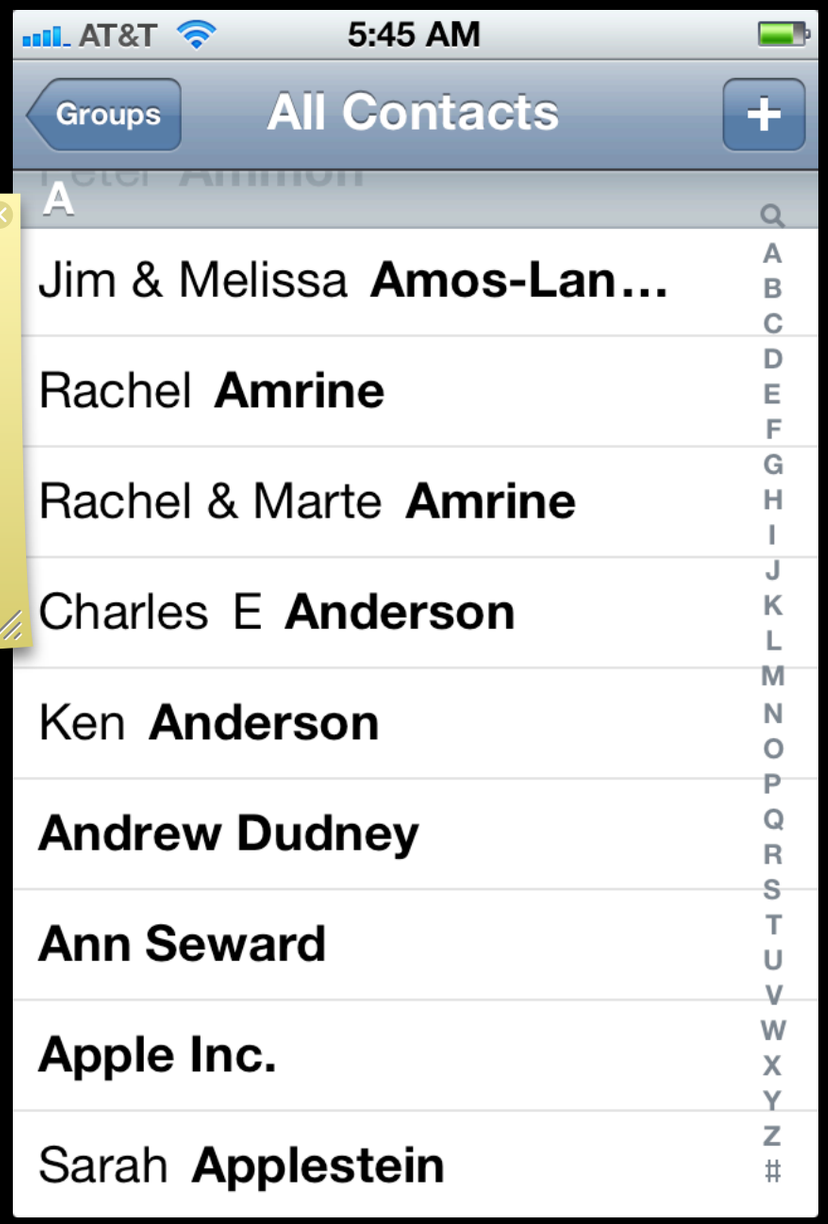
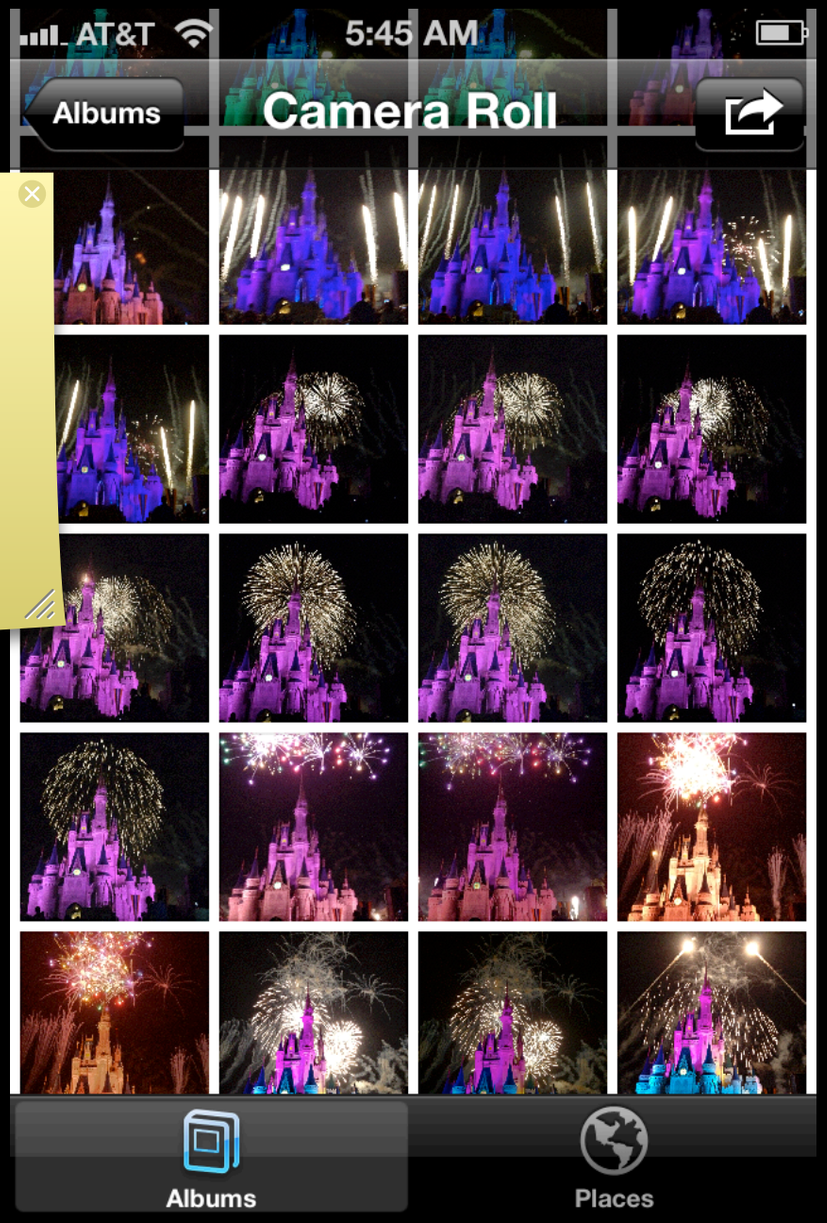


Table views

An iOS app workhorse

Take your own screenshots



Anatomy of a table view

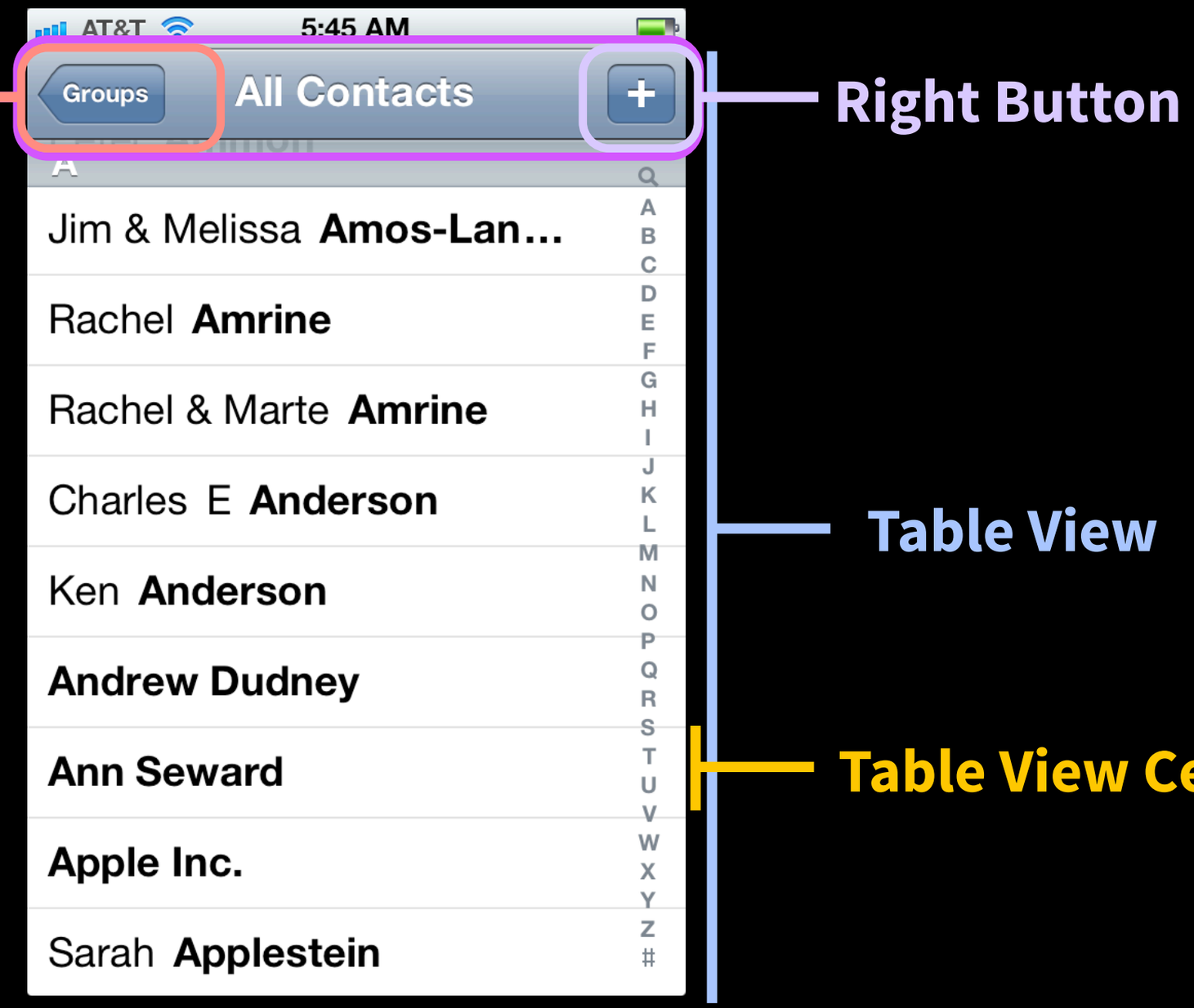


Table views display a variety of data

How can it be so flexible?

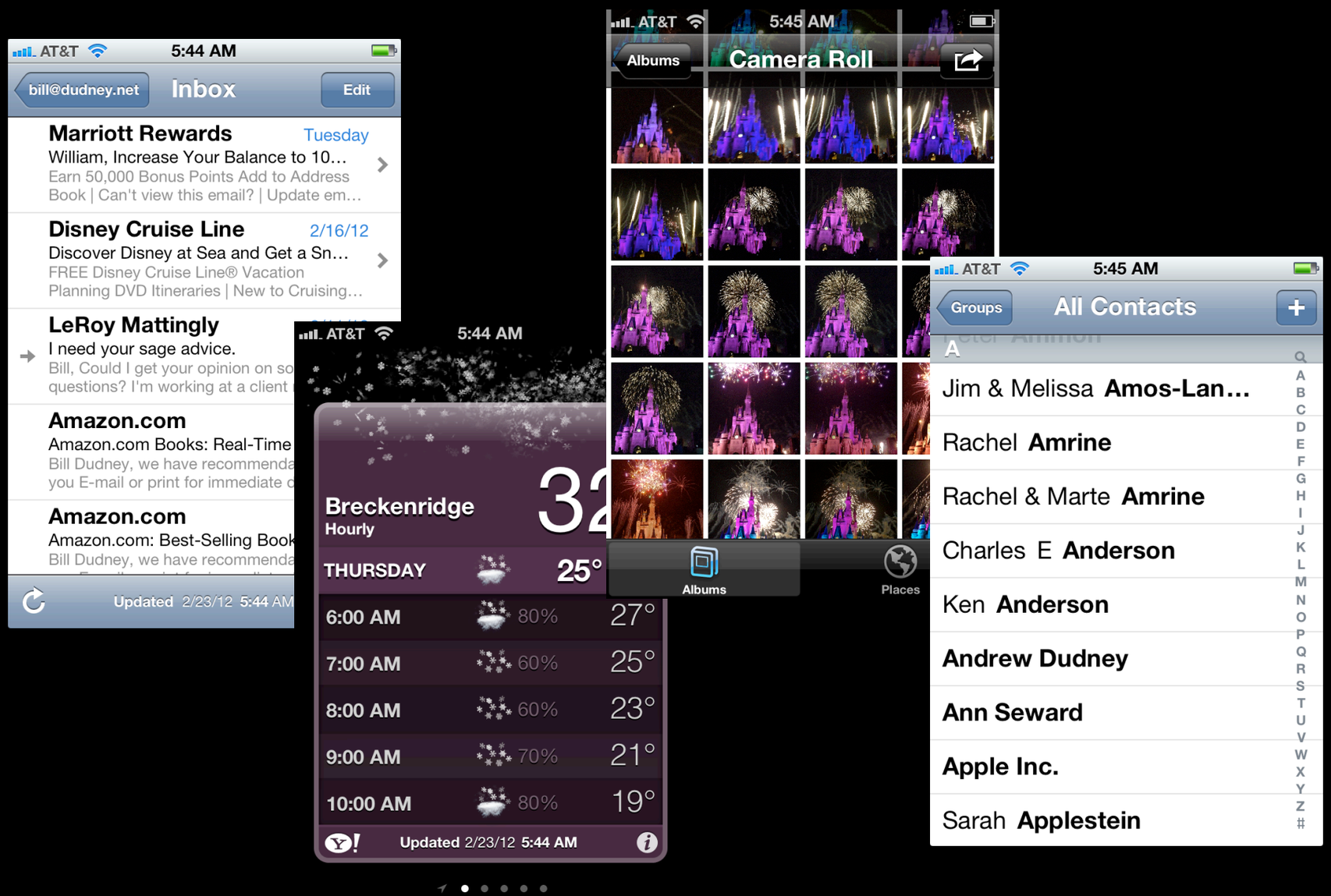
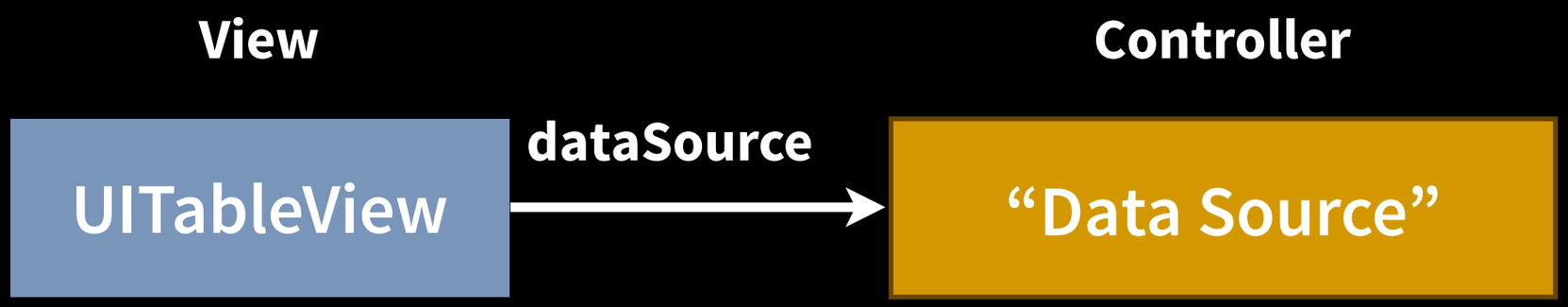


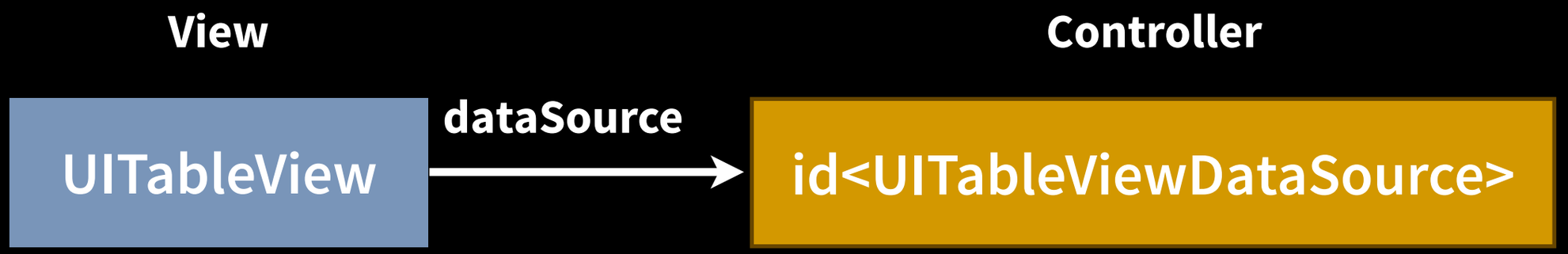
Table view works with a data source

Responsibility split between view and controller



Data source protocol defines interaction

Methods the table view expects



UITableViewDataSource

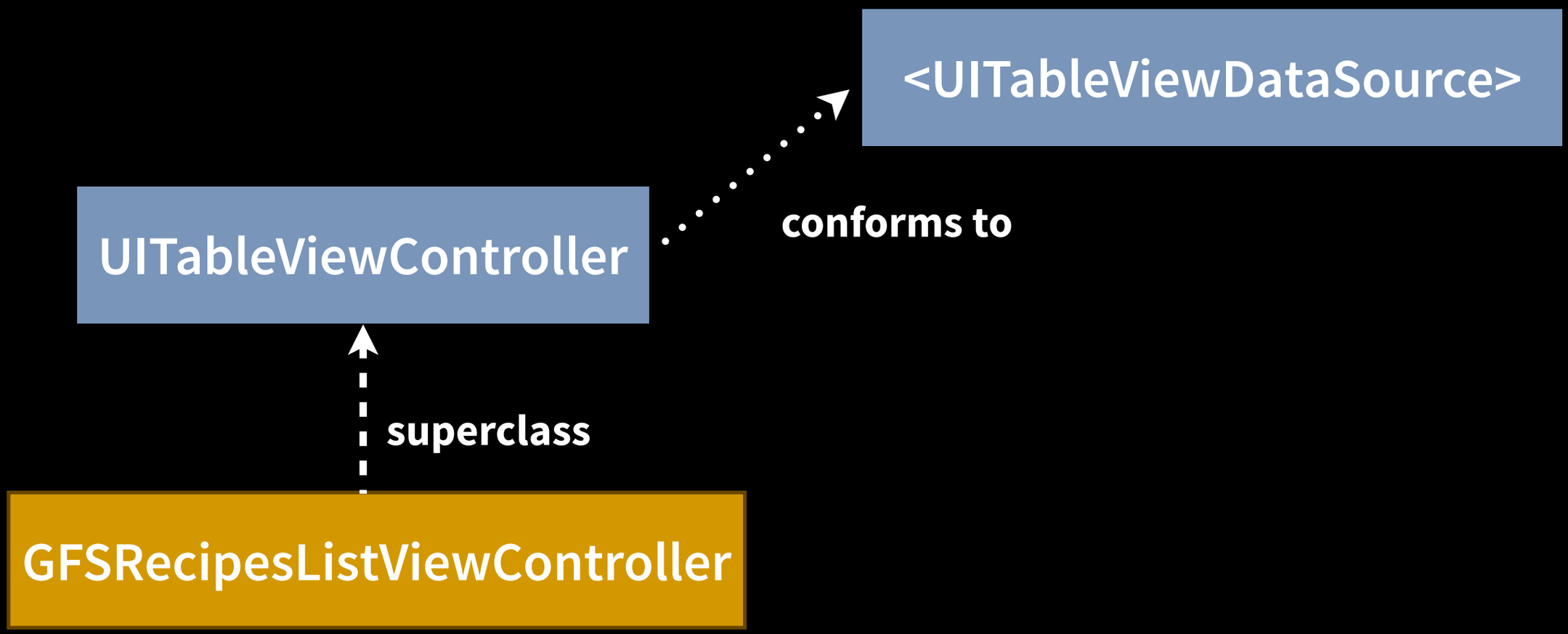
```
@protocol UITableViewDataSource <NSObject>

@required

- (NSInteger)tableView:(UITableView *)tableView
 numberOfRowsInSection:(NSInteger)section;

- (UITableViewCell *)tableView:(UITableView *)tableView
         cellForRowAtIndexPath:(NSIndexPath *)indexPath;

@optional

- (NSInteger)numberOfSectionsInTableView:(UITableView *)tv;

...

@end
```

UITableViewDataSource

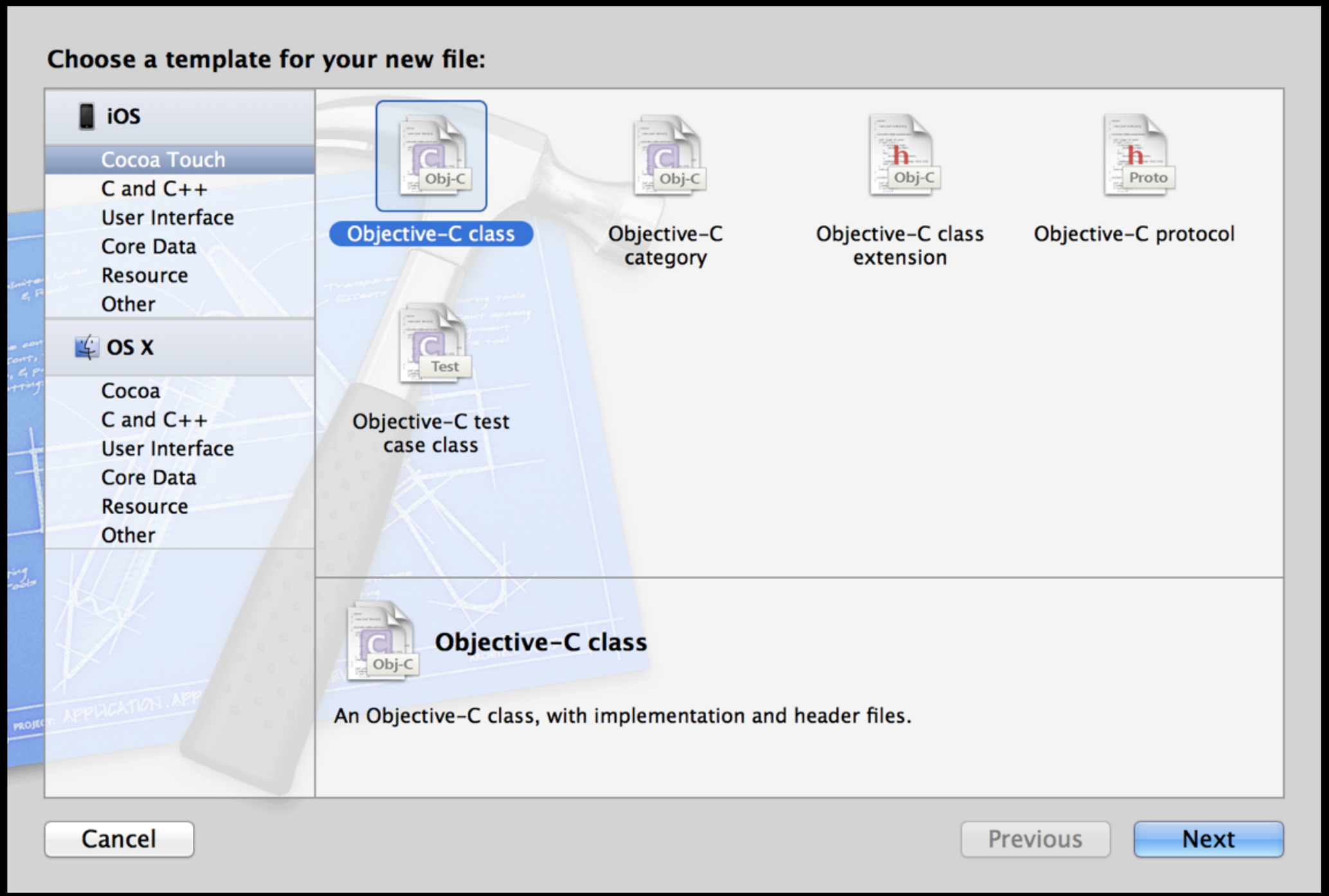
```
@protocol UITableViewDataSource <NSObject>

@required

- (NSInteger)tableView:(UITableView *)tableView
 numberOfRowsInSection:(NSInteger)section;

- (UITableViewCell *)tableView:(UITableView *)tableView
         cellForRowAtIndexPath:(NSIndexPath *)indexPath;

@optional

- (NSInteger)numberOfSectionsInTableView:(UITableView *)tv;

...

@end
```

UITableViewController

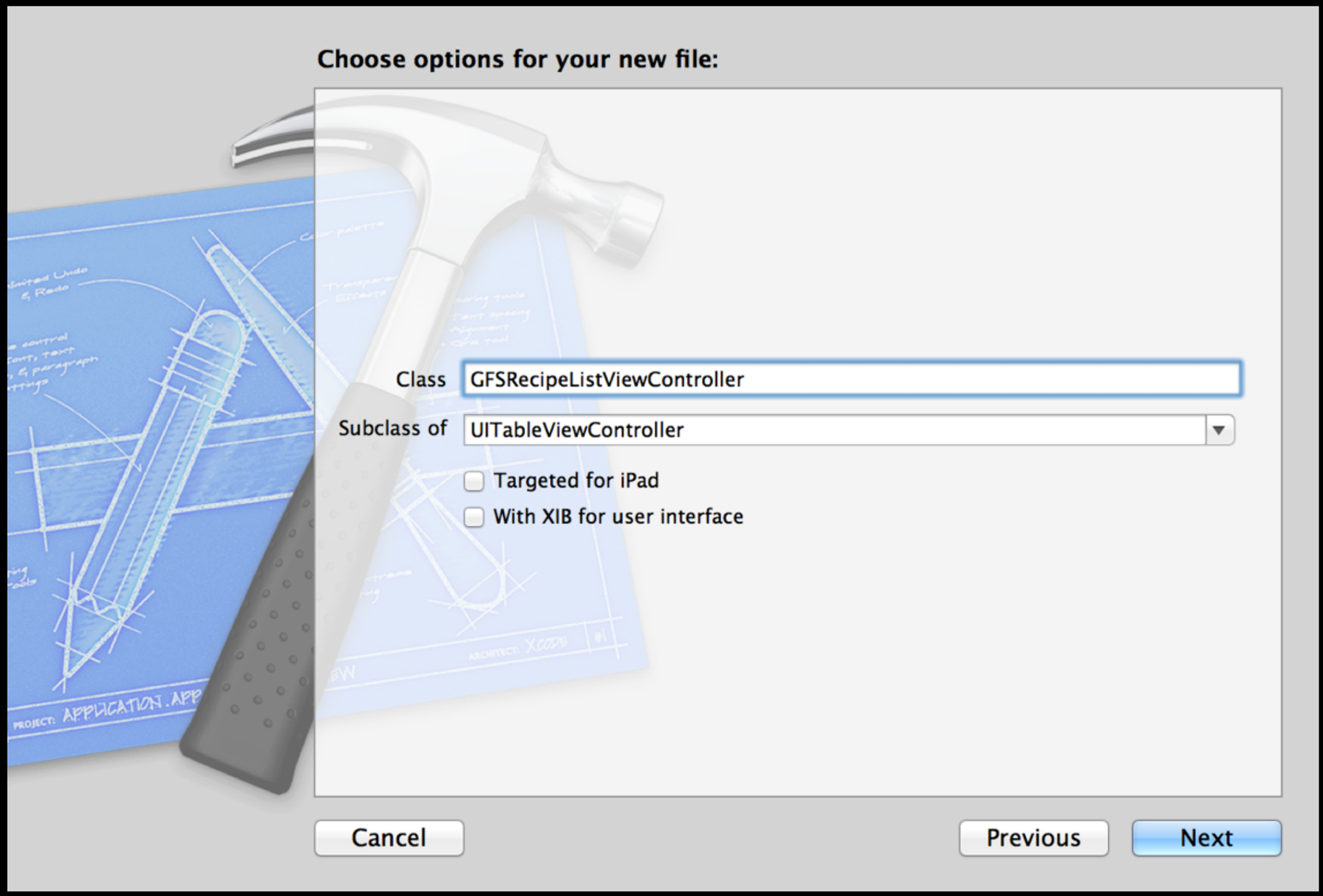
View controller subclass designed for table views



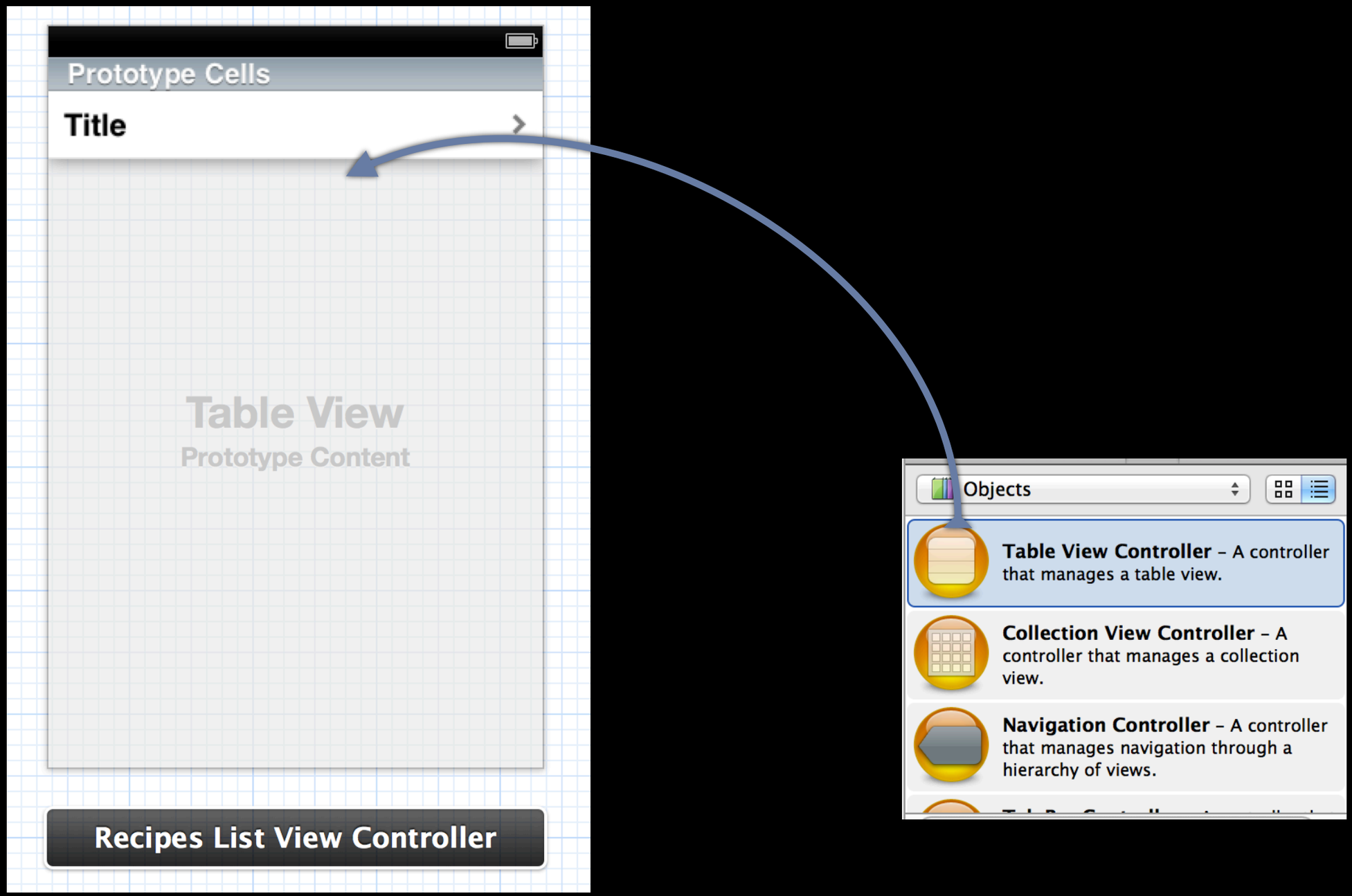
Let's Build


Choose a template for your new file:
iOS
Cocoa Touch
C and C++
User Interface
Core Data
Resource
Other
OS X
Cocoa
C and C++
User Interface
Core Data
Resource
Other
Objective-C class
Objective-C category
Objective-C class extension
Objective-C protocol
Objective-C test case class
Objective-C class
An Objective-C class, with implementation and header files.
Cancel
Previous
Next

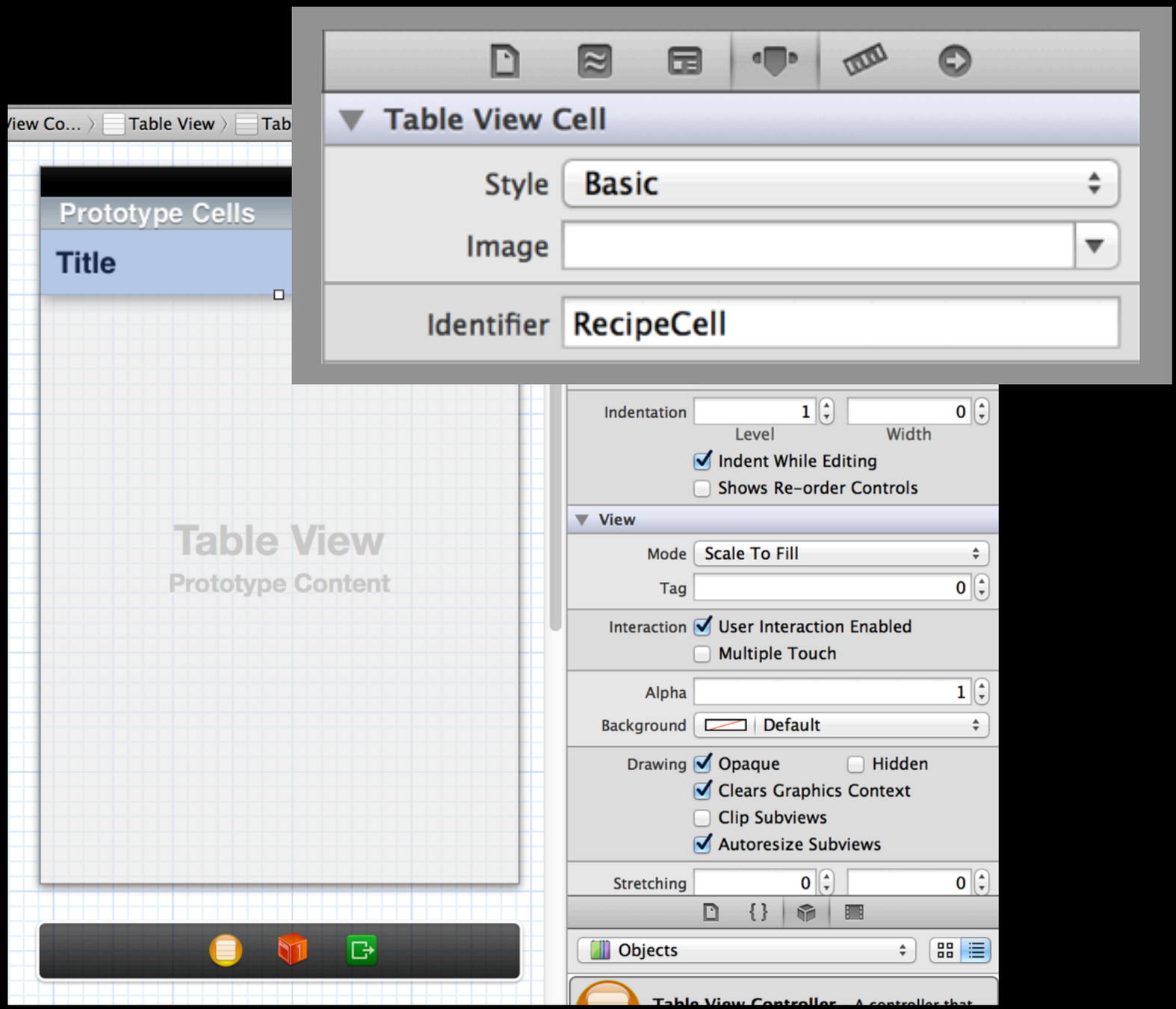
UITableViewController subclass



Add Table View Controller to Storyboard

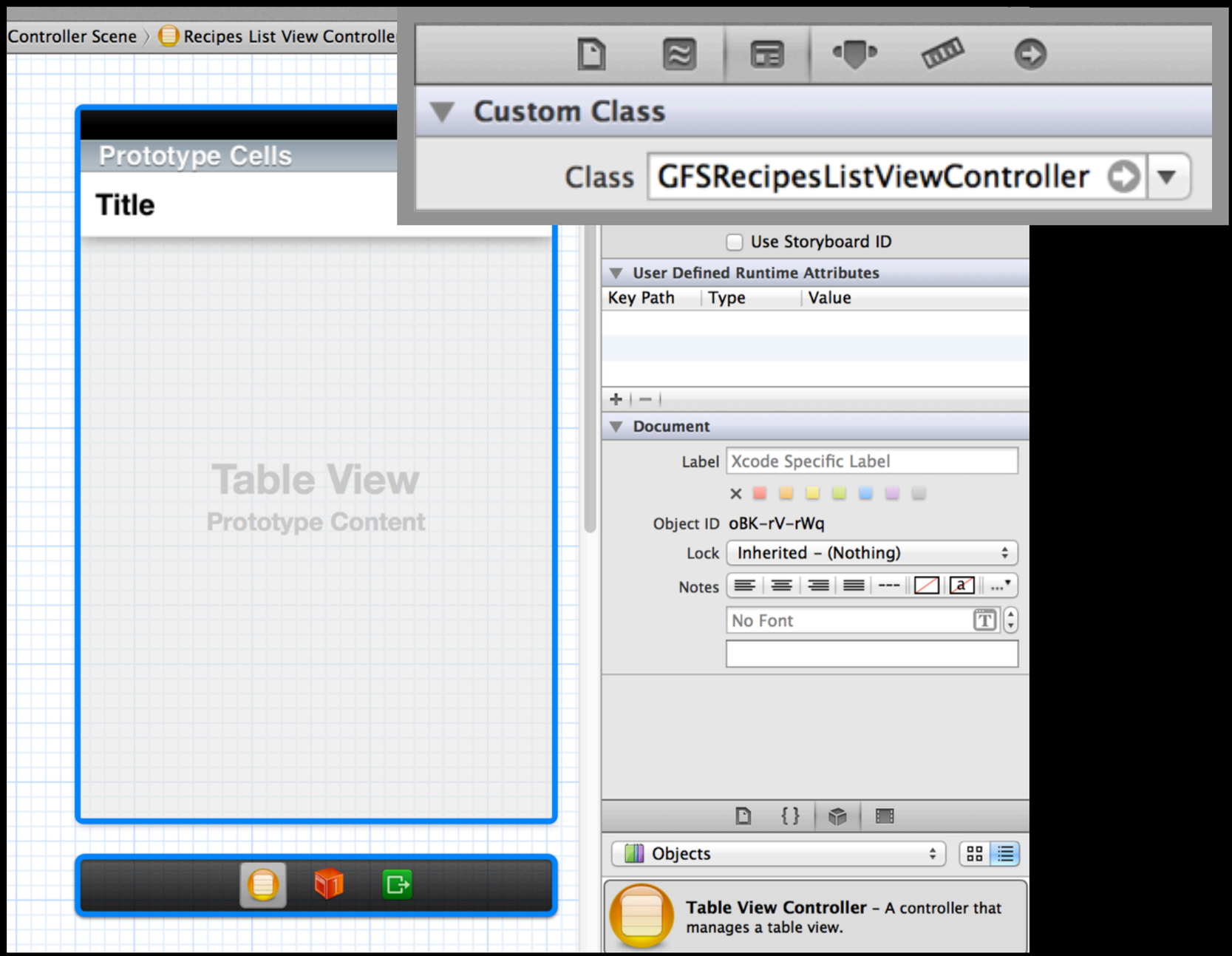


Set Up Prototype Cell



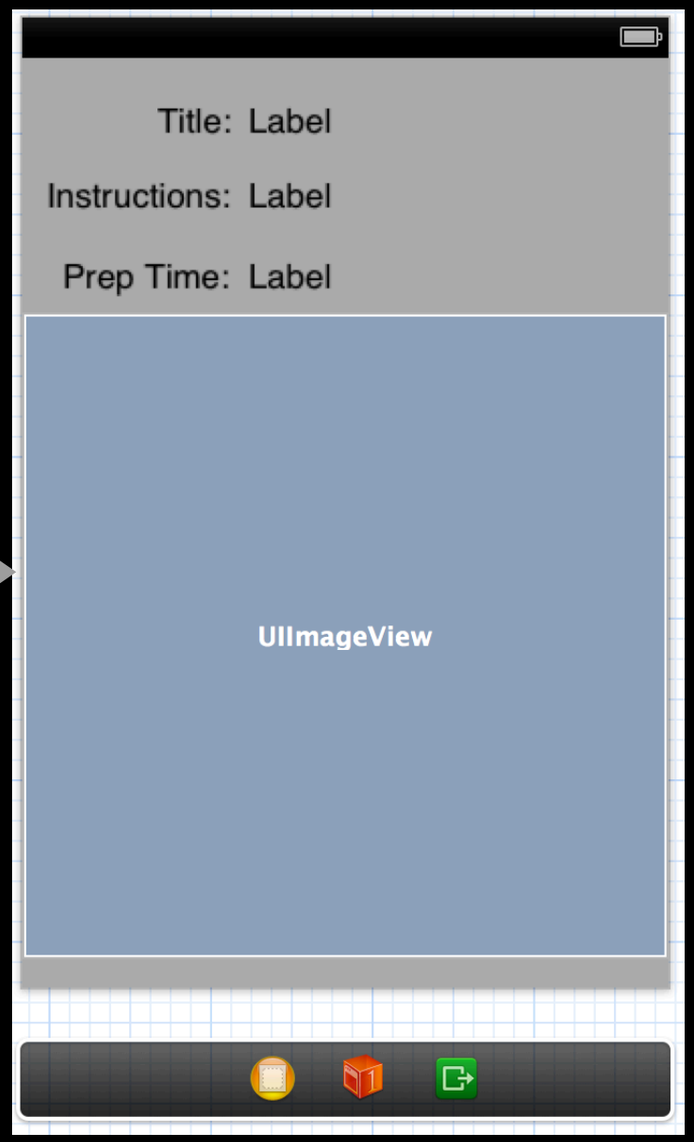
Set Custom Class

Use our custom table view controller class



Set Initial View Controller

Drag the arrow to indicate first scene on app launch



Template Table View Controller Code

```
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView {
#warning Potentially incomplete method implementation.
    // Return the number of sections.
    return 0;
}

- (NSInteger)tableView:(UITableView *)tableView
 numberOfRowsInSection:(NSInteger)section {
#warning Incomplete method implementation.
    // Return the number of rows in the section.
    return 0;
}

- (UITableViewCell *)tableView:(UITableView *)tableView
         cellForRowAtIndexPath:(NSIndexPath *)indexPath {
    static NSString *CellIdentifier = @"Cell";
    UITableViewCell *cell =
              [tableView dequeueReusableCellWithIdentifier:CellIdentifier
                                                forIndexPath:indexPath];

    // Configure the cell...

    return cell;
}
```

GFSRecipesListViewController.m

Recipes List View Controller

```
@interface GFSRecipeListViewController : UITableViewController

@property(nonatomic, strong) NSArray *recipes;

@end
```

GFSRecipesListViewController.h

Recipes List View Controller

```
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView {
    return 1;
}
```

GFSRecipesListViewController.m

Recipes List View Controller

```
- (NSInteger)tableView:(UITableView *)tableView
 numberOfRowsInSection:(NSInteger)section {

    return [self.recipes count];

}
```

GFSRecipesListViewController.m

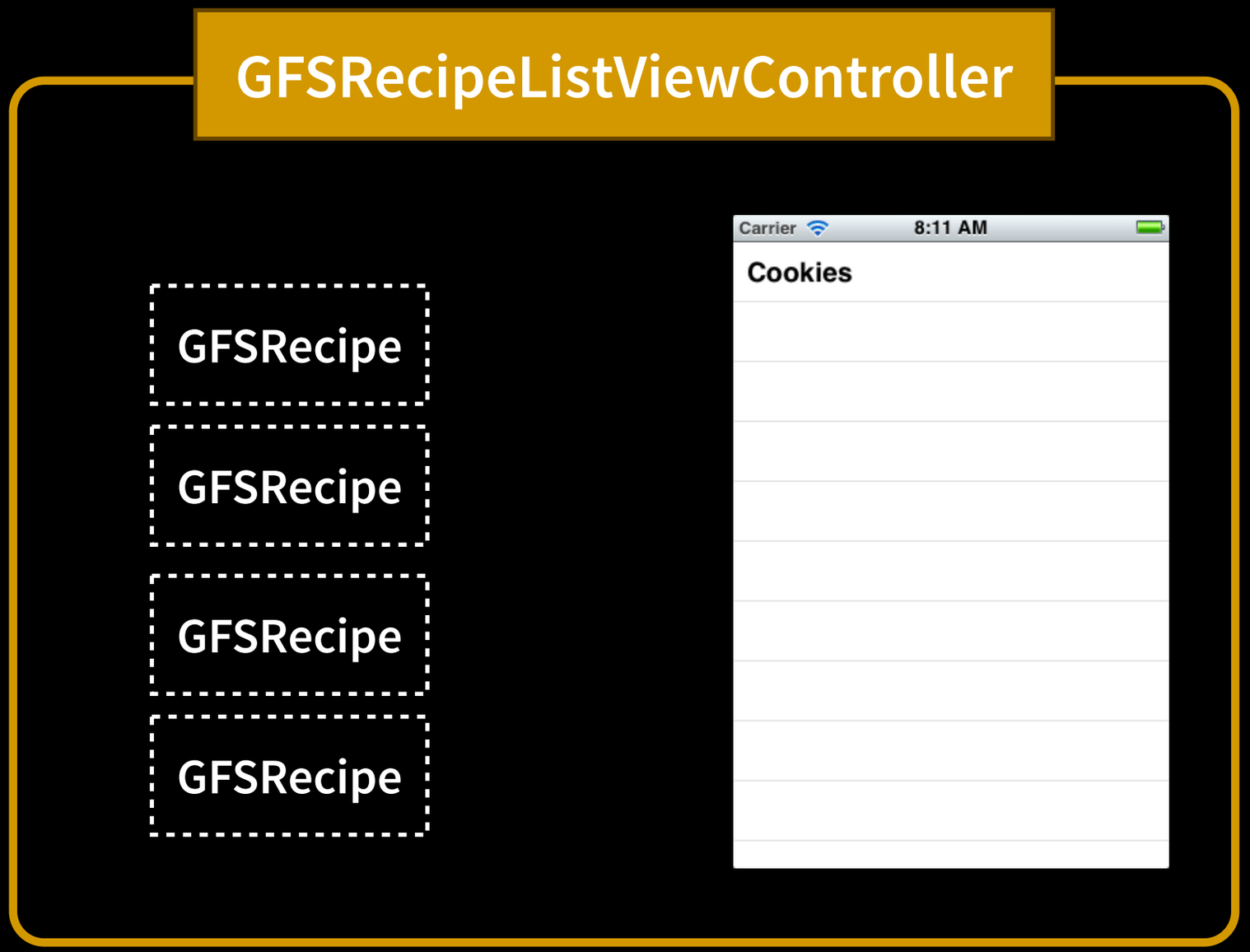
Recipes List View Controller

```
- (UITableViewCell *)tableView:(UITableView *)tableView
        cellForRowAtIndexPath:(NSIndexPath *)indexPath {

  static NSString *CellIdentifier = @"RecipeCell";

   UITableViewCell *cell =
              [tableView dequeueReusableCellWithIdentifier:CellIdentifier
                                              forIndexPath:indexPath];

  UITableViewCell *recipe = [self.recipes objectAtIndex:indexPath.row];
  cell.textLabel.text = recipe.name;

  return cell;
}
```

GFSRecipesListViewController.m

Encapsulation

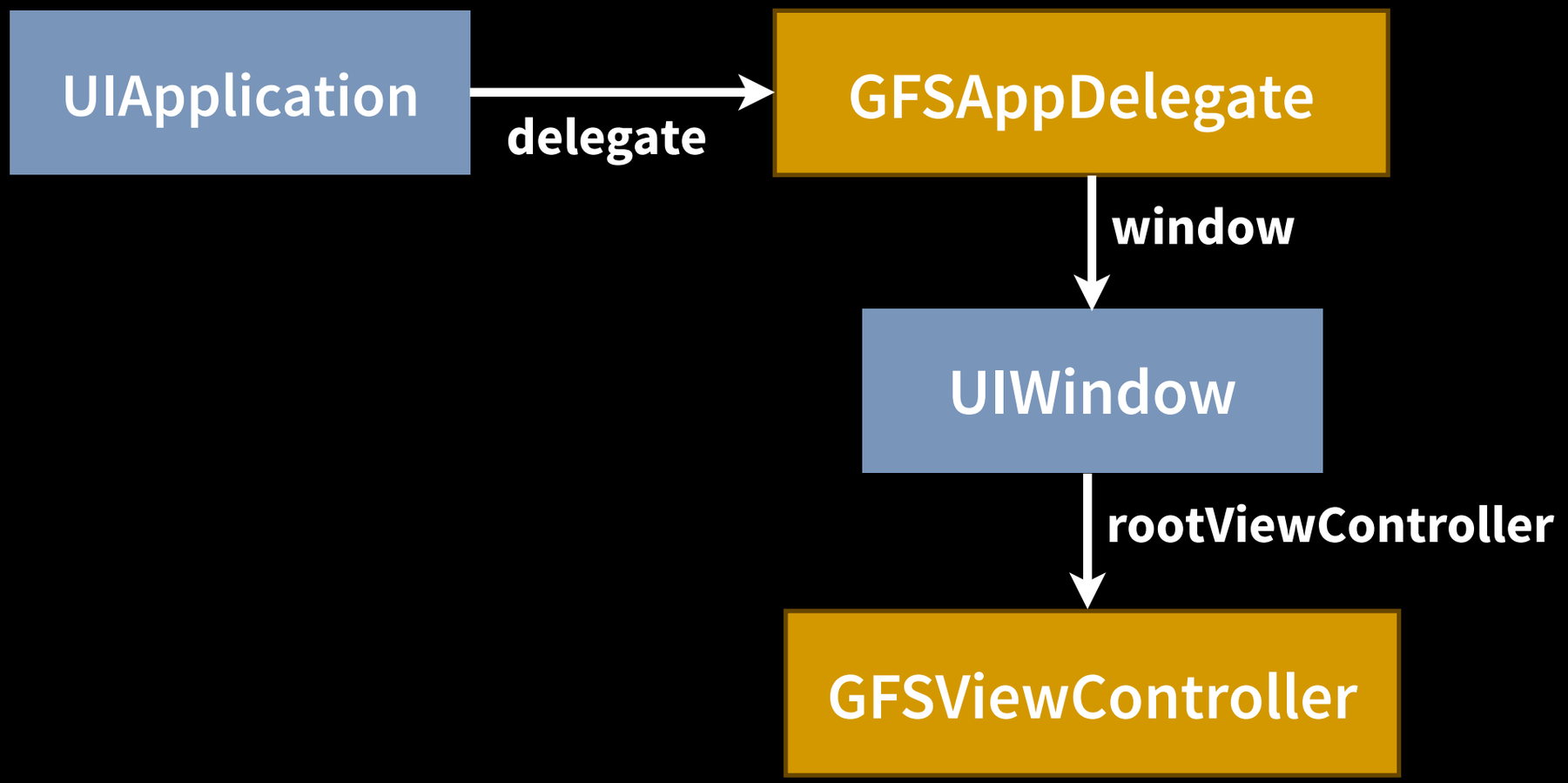
A view controller that can handle a list of recipes



Where do we get the recipe list?

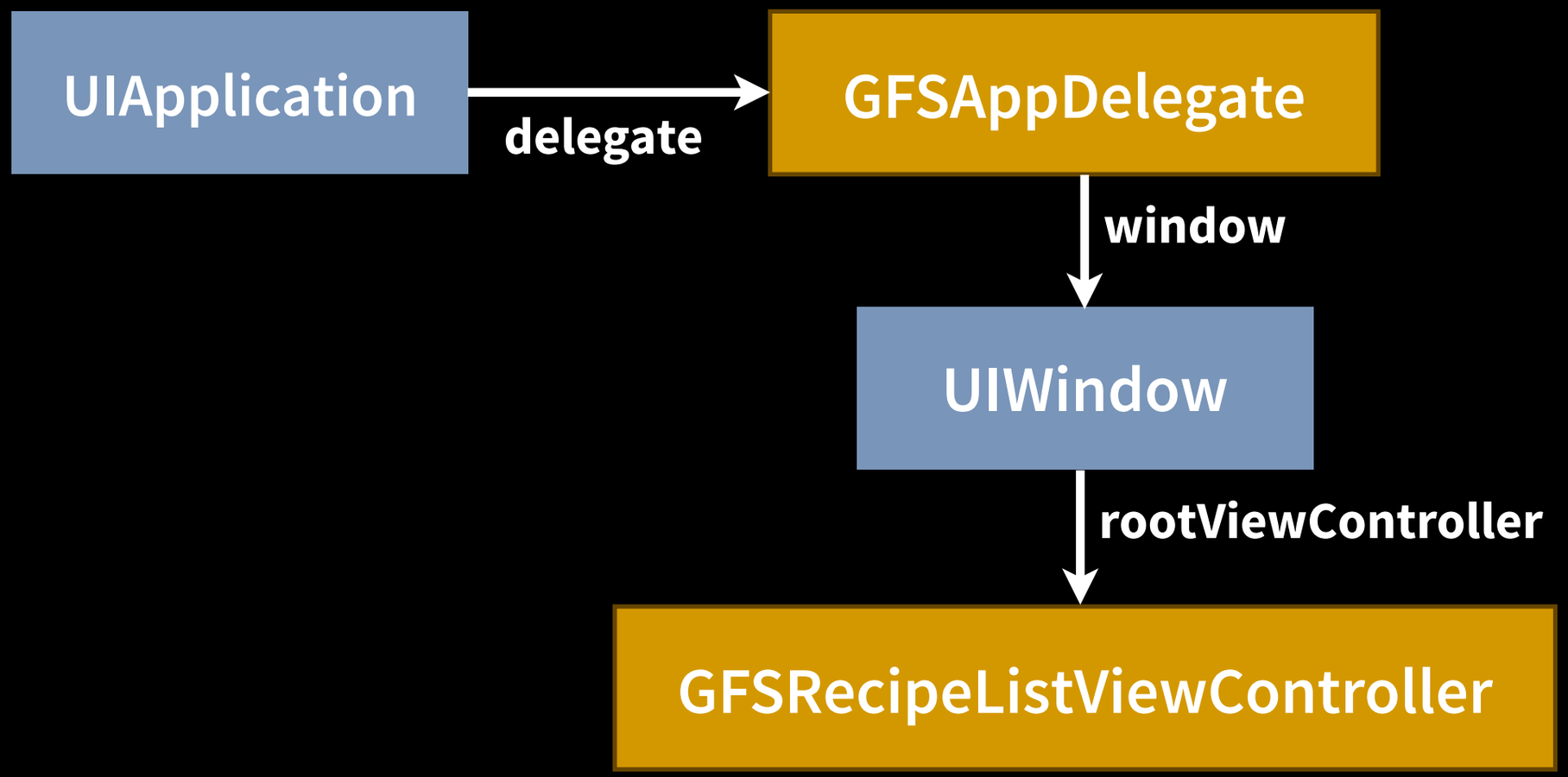
Objects revisited

We've changed our rootViewController



Objects revisited

We've changed our rootViewController



Making Recipes

```
- (NSArray *)makeRecipes {

  NSMutableArray *recipes = [NSMutableArray array];

  GFSRecipe *recipe = [[GFSRecipe alloc] init];
  recipe.title = @"Cookies";
  recipe.instructions = @"mix the stuff with the other st
  recipe.preparationTime = @15;

  [recipes addObject:recipe];

  return [recipes copy];
}
```

Get Image Name Right

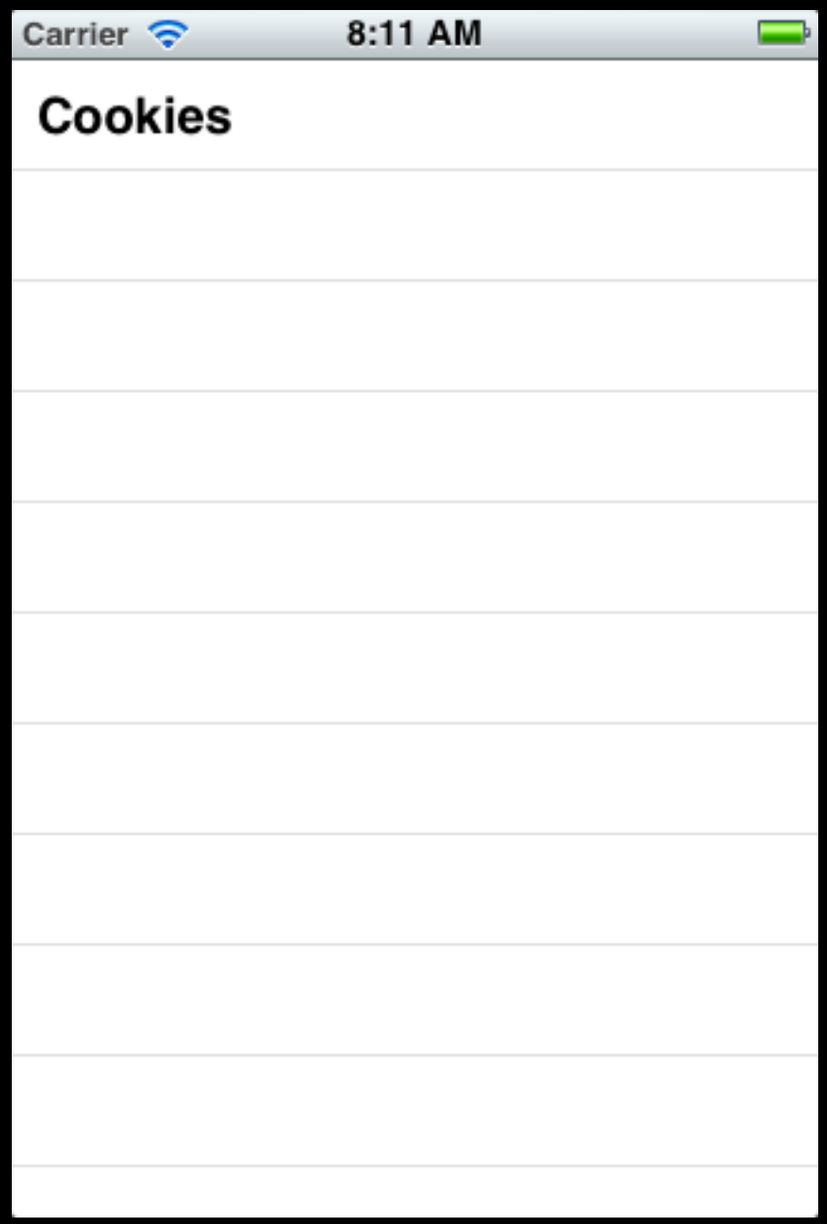
GFSAppDelegate.m

Application Delegate

```
- (BOOL)application:(UIApplication *)application
            didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {

  GFSRecipeListViewController *viewController =
                  (GFSRecipeListViewController *)self.window.rootViewController;

  viewController.recipes = [self makeRecipes];

  return YES;
}

@end
```

GFSAppDelegate.m

Results



Live Code - Watch Me

Your Turn

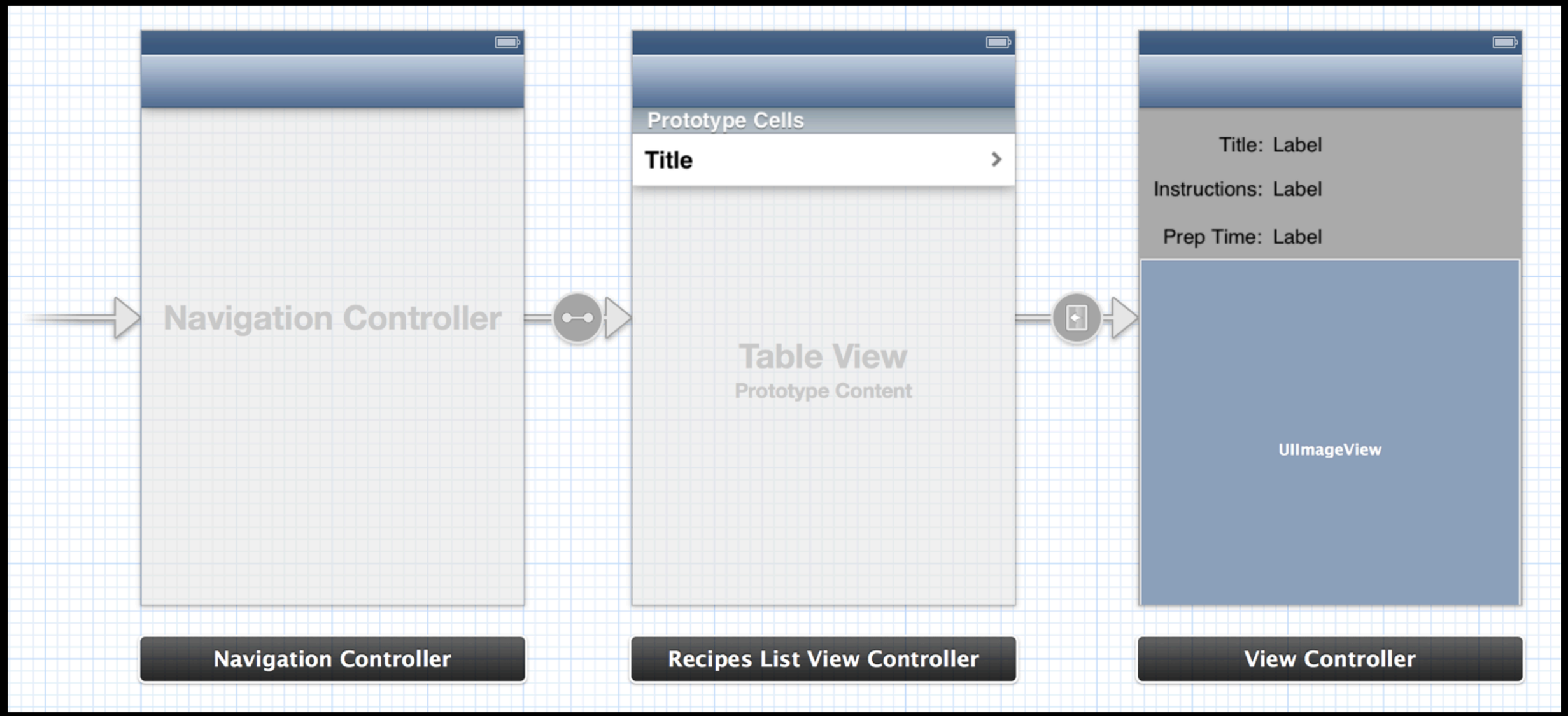
- Create the UITableViewController subclass
- Add table view controller in storyboard and set subclass
- Change the initial view controller in storyboard
- Edit table view controller subclass to handle array of recipes
- In app delegate, move recipe creation to separate method
- Put the array of recipes into the table view controller
- Run

Recipes_01 ⟶ Recipes_02

Storyboards and Navigation

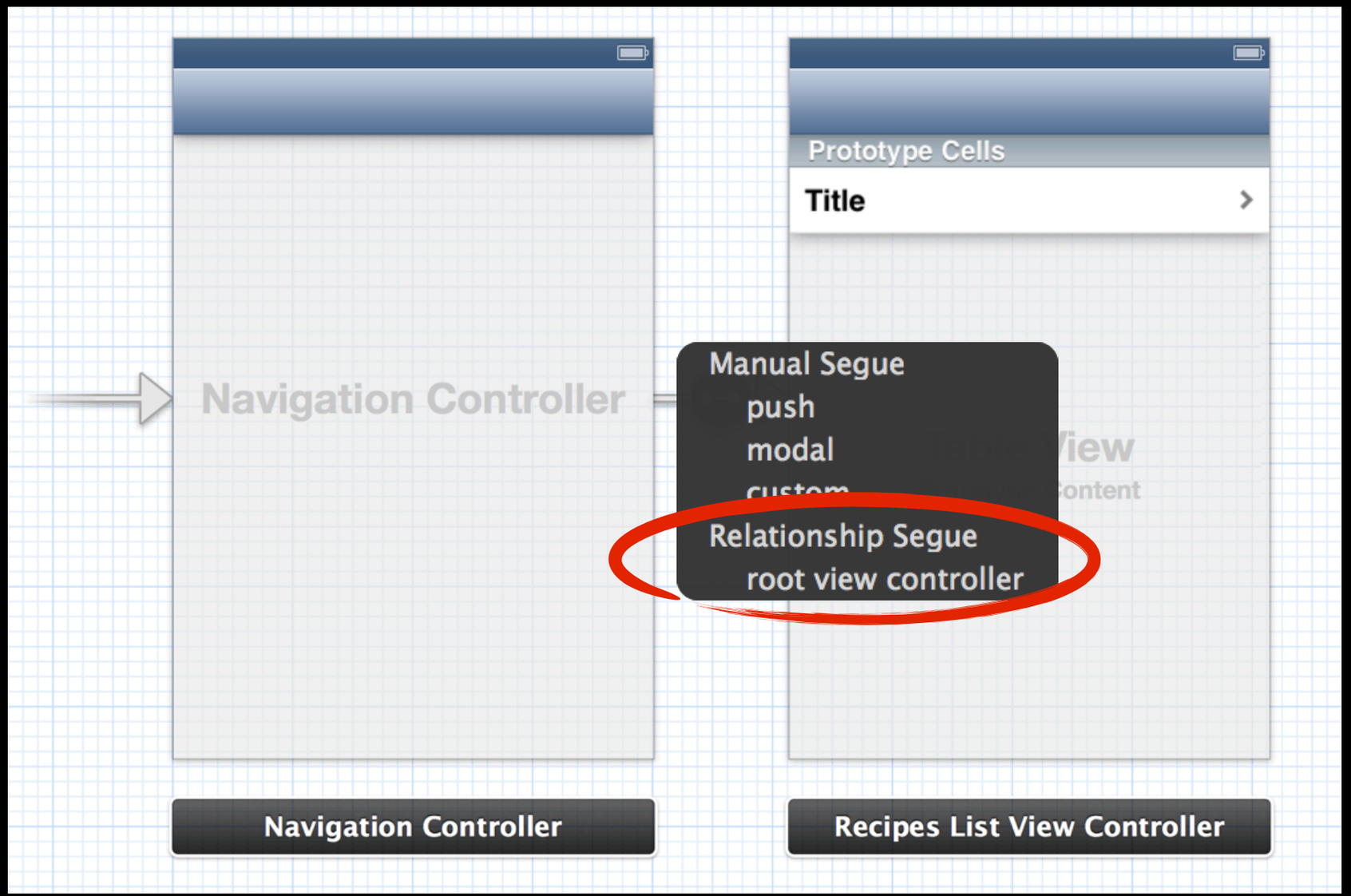
Storyboards

Scenes and Segues



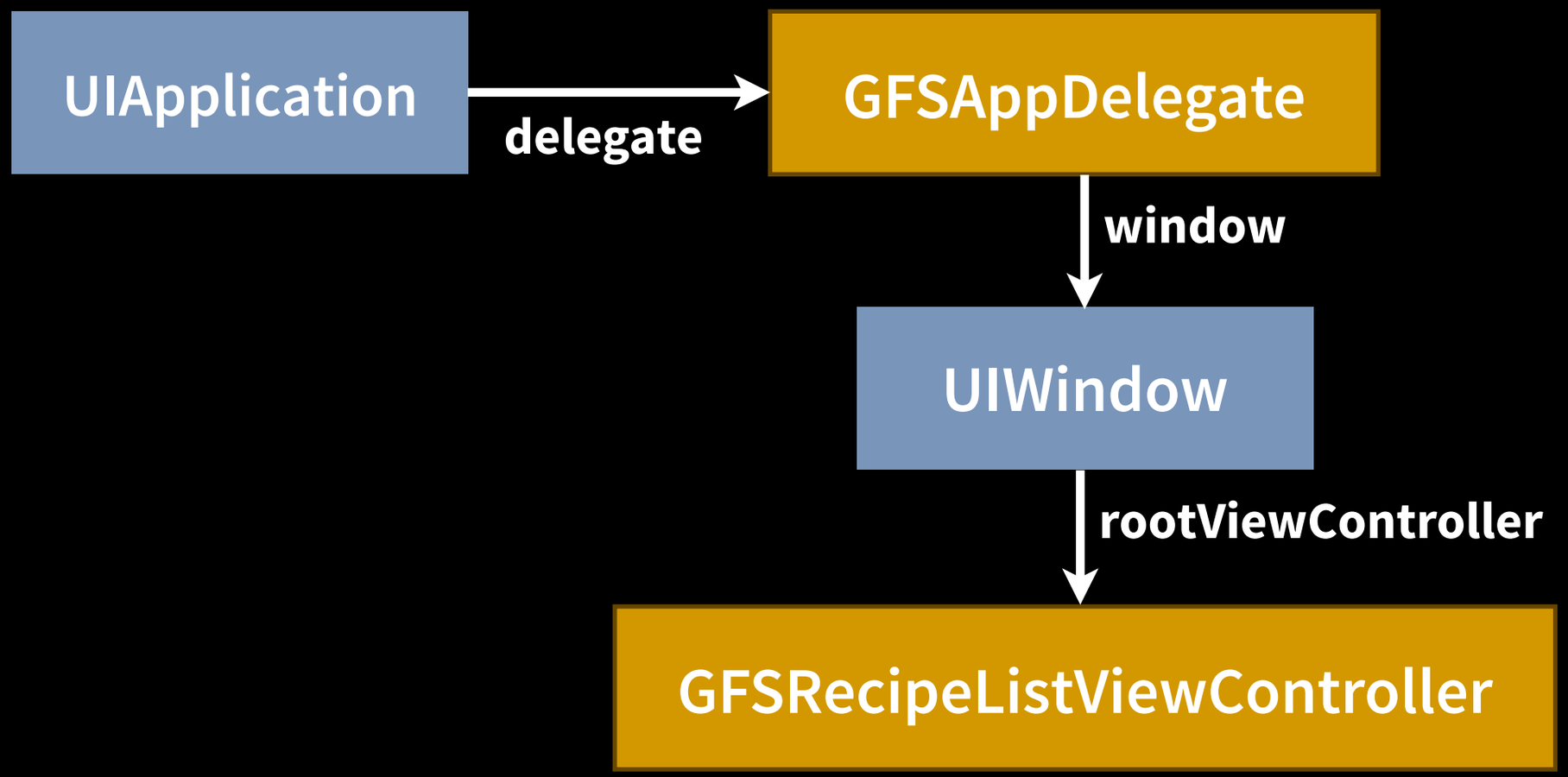
Add Navigation Controller

Control-Drag to the Table View



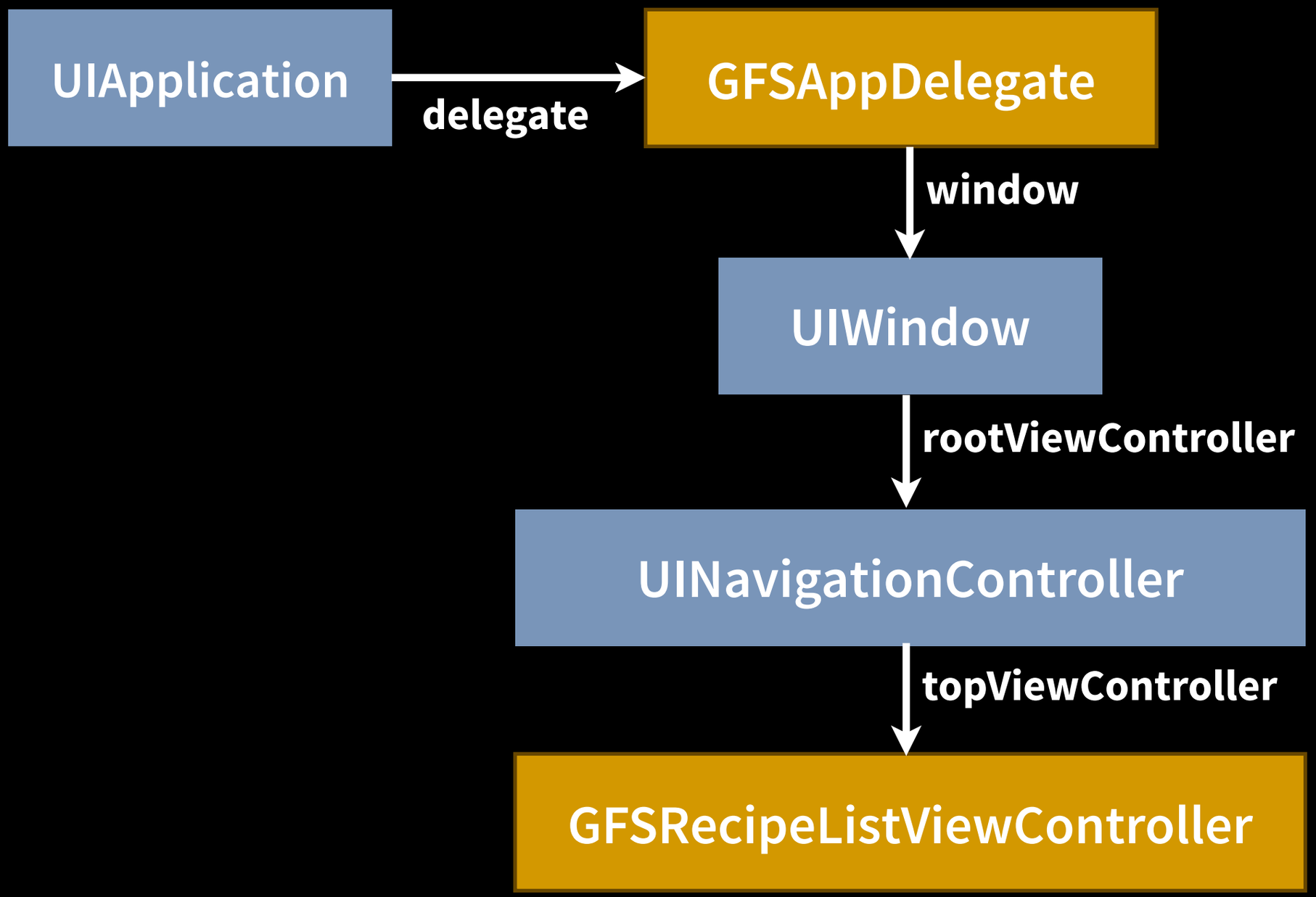
Objects revisited

We've changed our rootViewController again



Objects revisited

We've changed our rootViewController again



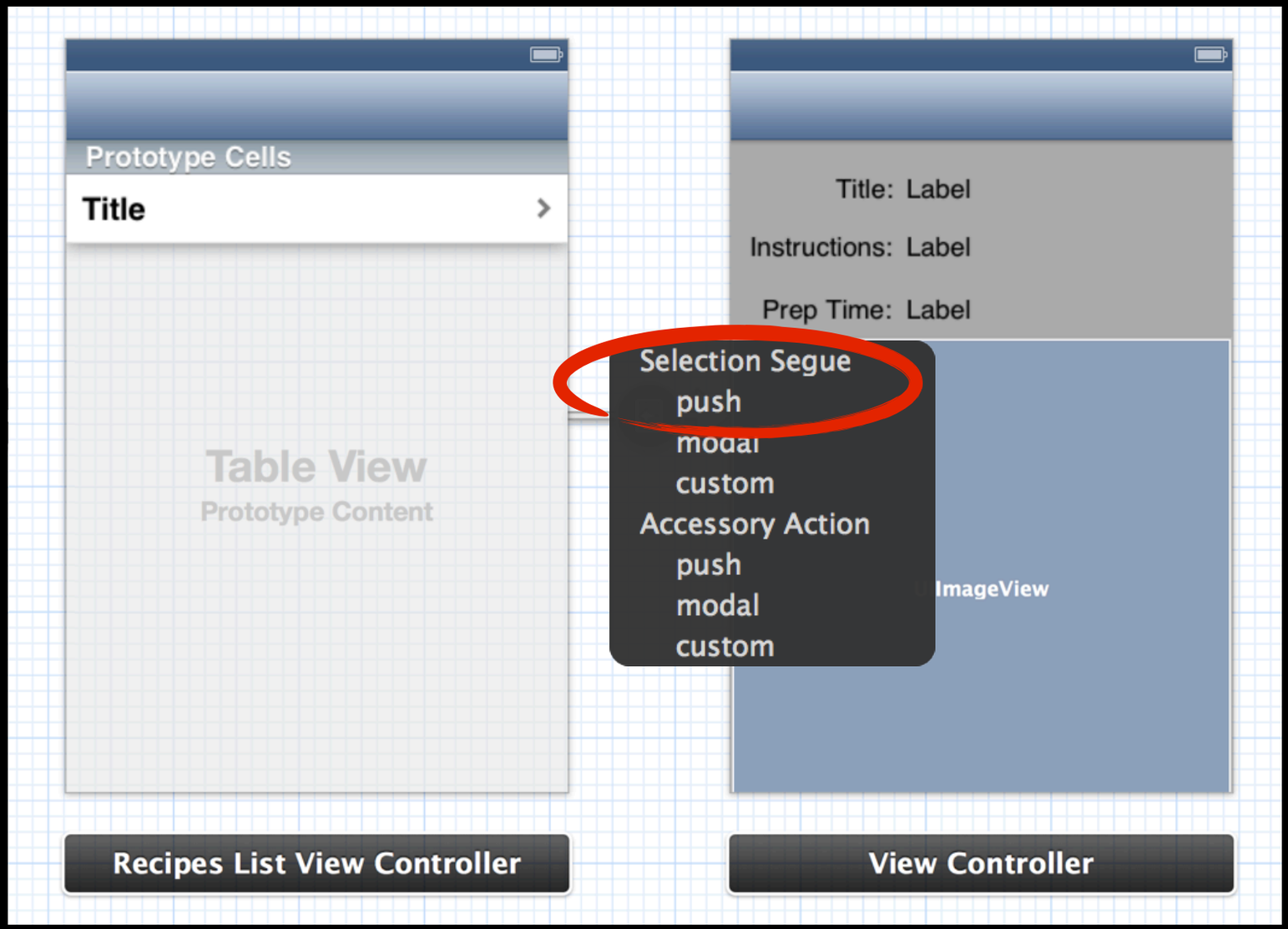
Application Delegate

```
- (BOOL)application:(UIApplication *)application
            didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {

  UINavigationController *navController =
                  (UINavigationController *)self.window.rootViewController;

  GFSRecipeListViewController *viewController =
                 (GFSRecipeListViewController *) navController.topViewController;

  viewController.recipes = [self makeRecipes];

  return YES;
}

@end
```

GFSAppDelegate.m

Connect the Detail View

Control-Drag to the detail view



Prepare for Segue

Pass the selected object to the detail view controller

```
- (void)prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender {

  NSIndexPath *indexPath = [self.tableView indexPathForCell:sender];

  GFSRecipe *recipe = [self.recipes objectAtIndex:indexPath.row];

  [segue.destinationViewController setRecipe:recipe];
}
```

GFSAppDelegate.m

Live Code - Watch Me

Your Turn

- Navigation controller to storyboard
- Delete the table view scene that is automatically created
- Set the nav controller as initial view controller
- Control-Drag from nav controller to table view controller
- Update app delegate, the nav controller is now the root
- Control-Drag from table view cell to detail controller
  - Be sure to choose ‘Selection Segue’
- Implement prepareForSegue:sender: method
- Run

Recipes_02 ⟶ Recipes_03

Encapsulation

Recipes Data Source

```
@class GFSRecipe;

@protocol GFSRecipesDataSource <

- (NSInteger)recipeCount;
- (GFSRecipe *)recipeAtIndex:(NS
- (void)deleteRecipeAtIndex:(NSI

@end
```

Could use a graphic here to explain what we are doing

GFSRecipesDataSource.h

Recipes List

```
#pragma mark Recipe List Data Source Methods

- (NSInteger)recipeCount {
  return [self.recipes count];
}

- (GFSRecipe *)recipeAtIndex:(NSInteger)index {
  return [self.recipes objectAtIndex:index];
}

- (void)deleteRecipeAtIndex:(NSInteger)index {
  [self.recipes removeObjectAtIndex:index];
}
```

GFSRecipesList.m

App Delegate

```
- (BOOL)application:(UIApplication *)application
didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {

  UINavigationController *navVC =
     (UINavigationController *)self.window.rootViewController;

  GFSRecipesListViewController *vc =
         (GFSRecipesListViewController *)
                                  [navVC topViewController];

  vc.dataSource = [[GFSRecipesList alloc] init];

  return YES;

}
```

GFSAppDelegate.m

Table View Controller

```
#import <UIKit/UIKit.h>
#import "GFSRecipesDataSource.h"


@interface GFSRecipeListViewController : UITableViewController

@property(nonatomic, strong) NSArray *recipes;

@property(nonatomic, strong) id <GFSRecipesDataSource> dataSource;

@end
```

GFSRecipesListViewController.h

Table View Controller

```
- (NSInteger)tableView:(UITableView *)tableView
 numberOfRowsInSection:(NSInteger)section {
  return [self.dataSource recipeCount];
}

- (UITableViewCell *)tableView:(UITableView *)tableView
         cellForRowAtIndexPath:(NSIndexPath *)indexPath {
  static NSString *CellIdentifier = @"Cell";
  UITableViewCell *cell = [tableView

dequeueReusableCellWithIdentifier:CellIdentifier];
  cell.textLabel.text = [[self.dataSource recipeAtIndex:indexPath.row]
name];

  return cell;
}

- (void)prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender {
  NSIndexPath *indexPath = [self.tableView indexPathForCell:sender];
  GFSRecipe *recipe = [self.dataSource recipeAtIndex:indexPath.row];
  [[segue destinationViewController] setRecipe:recipe];
}
```

GFSRecipesListViewController.m

Live Code - Watch Me

Your Turn

- Define the data source protocol
- Create recipe list class and implement data source protocol
- Use the data source protocol in the table view controller
- Create recipe list in app delegate and pass to table view controller

Recipes_03 ⟶ Recipes_04

Auto-Layout

If we have time

Old-school layout

- Set the frame rectangle of each view
- Use autoresizing ‘springs and struts’
- Do all but very simple layout in code

Auto Layout

- Constraint-based layout ‘rules’
- Frames are determined by constraints
- Very powerful layout, often with no code at all
- You define constraints for each view
  - Automatically in Interface Builder
  - Explicitly in Interface Builder or code

Watch Me

iOS Tutorial

James Dempsey
http://jamesdempsey.net

Tapas Software
http://tapas-software.net