GENERAL ASSEMBLY

IOS DEVELOPMENT WORKSHOP

Slides and Code Samples on Github

http://bit.ly/2e75RU2

Jeffrey Bergier
iOS Developer, Topology Eyewear

LEARNING OBJECTIVES

- Explain relevant history and trends in iOS development.
- Identify key skills leveraged by iOS developers.
- Survey the common tools used within the iOS ecosystem.
- Apply key concepts and skills to build your own basic iOS application.
- Create a custom learning plan to help you continue to build fundamental iOS development skills after this workshop.

PRE-WORK

PRE-WORK REVIEW

- Bring a Mac laptop with Xcode installed. Macs are required to create apps for the iOS ecosystem.
- Please note: you may need to update your OS in order to install the latest version of Xcode.

OPENING

GENERAL ASSEMBLY

JEFFREY BERGIER

iOS Developer @ Topology Eyewear
TA @ General Assembly

UX Designer @ Riverbed (4 years)
Teacher @ MobileBridge

@jeffburg

jeffburg.com

WaterMe

Plant Watering Reminders

Gratuity

The Simple Tip Calculator

ABOUT YOU

- Before we dive in, let's talk a bit about you!

- Name:
- What brings you to GA?
  - Current activities:
  - Goals:
- Fun fact?

OUR EXPECTATIONS

- You're ready to take charge of your learning experience.
- You’ve brought a Mac laptop and optional testing device.
- You’ve downloaded & installed the latest version of Xcode.
- You're curious and excited about iOS development!

THE BIG PICTURE

- What we'll cover:
  - iOS ecosystem
  - Common iOS Developer tools
  - App building principles
  - How to build a sample app

THE BIG PICTURE

- Why this topic matters:
  - App development is a huge part of modern business
  - Most people interact with brands & data using mobile devices

- Why this topic rocks:
  - iOS development offers a fun and rewarding career path!

WHAT IS IOS?

WHAT IS IOS?

- iOS is the operating system that powers the iPhone and iPad.
- It's a close cousin of macOS (until recently "Mac OS X"), tvOS, and watchOS.
- Learning the basics of iOS is a way not only to write apps for the Apple App Store, but to learn about building apps for other Apple technologies and devices.
- Programming for iOS means learning about Xcode, building user interfaces, and writing code.

WHAT IS SWIFT?

- Swift is the new language of choice for iOS development.
- While Objective-C is over 30 years old, Swift is more concise and approachable than Objective-C.
- Here are two code snippets as a comparison:

INTRODUCTION

Objective-C

```
#import <Foundation/Foundation.h>
@interface User : NSObject

@property (nonatomic, strong) NSString* firstName;
@property (nonatomic, strong) NSString* lastName;
-(instancetype)initWithFirstName: (NSString *)firstName
lastName: (NSString *)lastName;

@end


#import "User.h"
@implementation User

-(instancetype)initWithFirstName: (NSString *)firstName
lastName: (NSString *)lastName
{
    self = [super init];
    if (self) {
        self.firstName = firstName;
        self.lastName = lastName;
    }
    return self;
}

@end
```

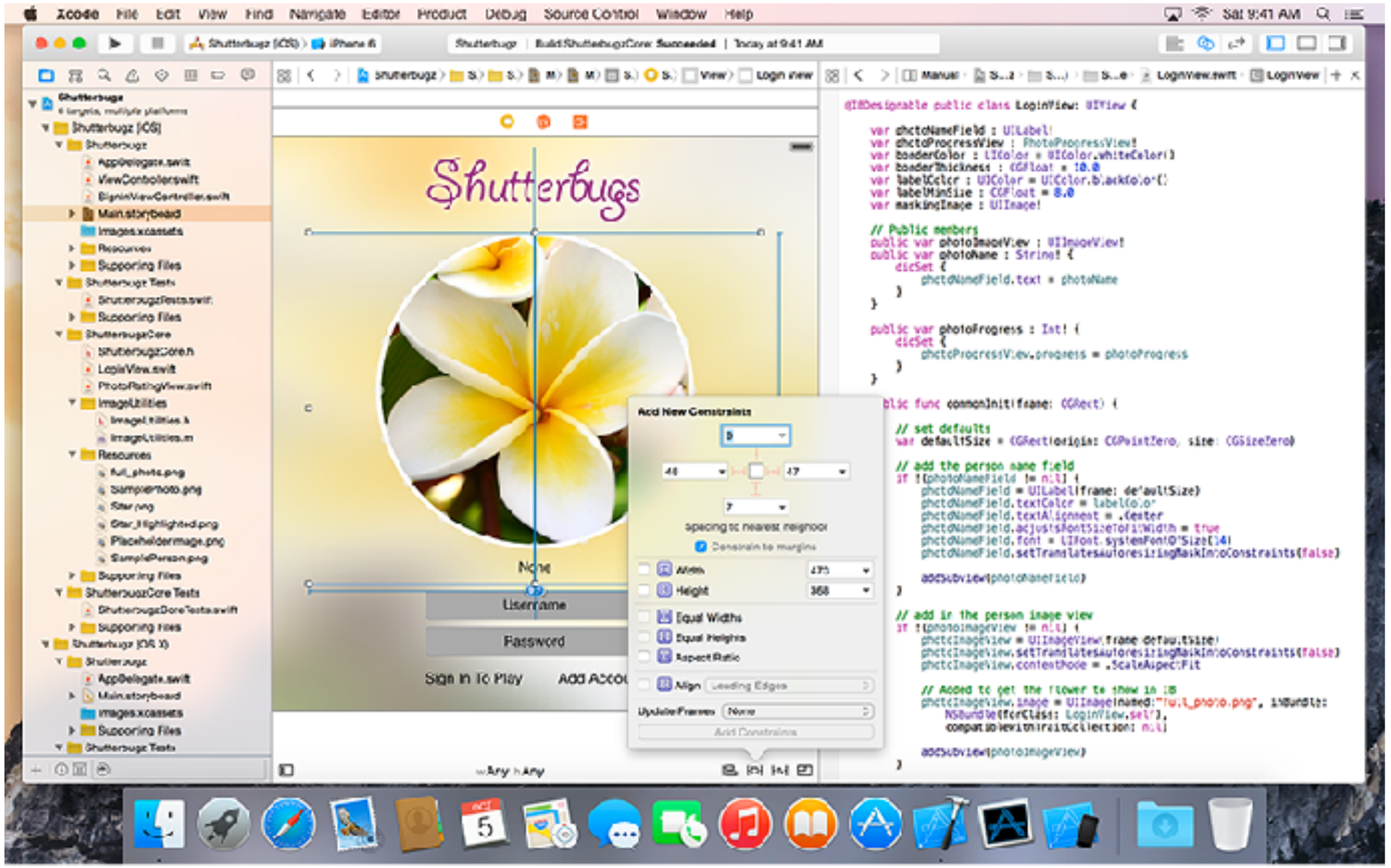
Swift

```
class User {

    var firstName: String
    var lastName: String

    init(firstName: String, lastName: String) {
        self.firstName = firstName
        self.lastName = lastName
    }
}
```

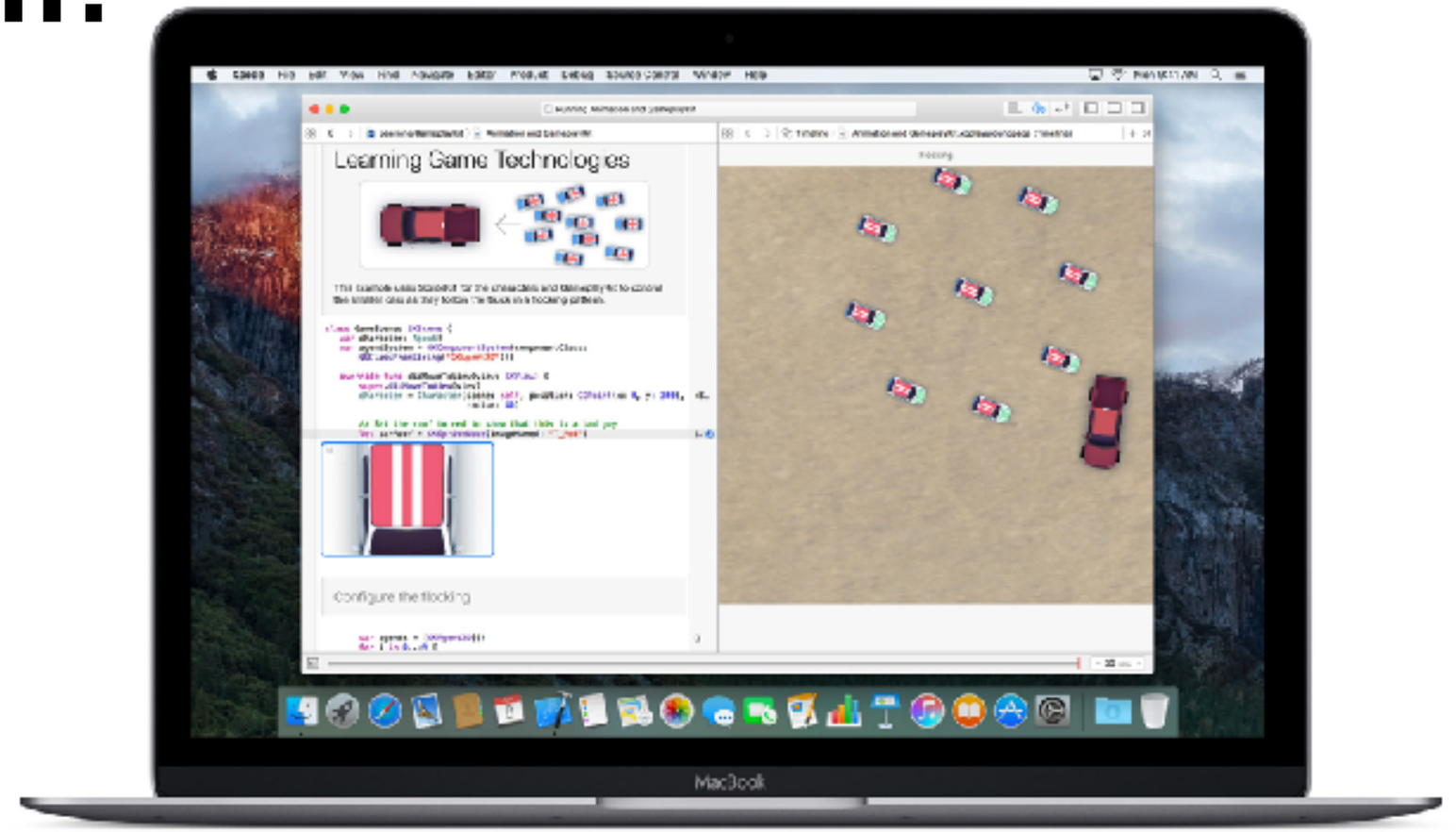
THE IOS DEVELOPER ECOSYSTEM

INTRODUCTION

XCODE



INTERFACE BUILDERS, PLAYGROUNDS, SIMULATORS, OH MY!



THE ECOSYSTEM

- Operating system (iOS) - Talks to the device. We build apps for the OS.
- Frameworks - Provided by Apple to make building apps easier. e.g. UIKit provides buttons, toolbars, ways to include images, etc.
- Third-party libraries and packages - Code from other developers that makes your life easier.

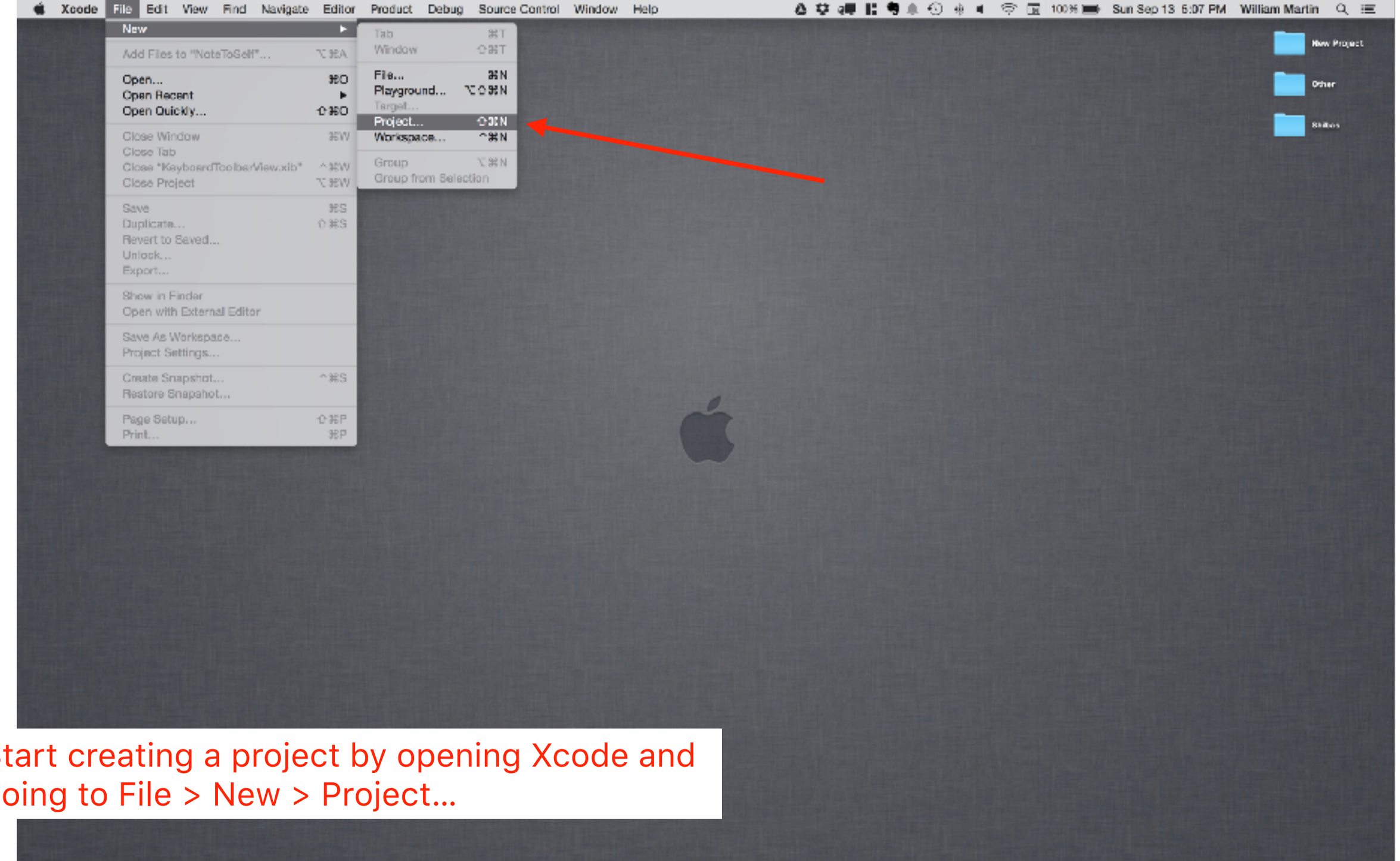
WHAT IS COCOA TOUCH?

- Cocoa Touch contains key “frameworks” for building iOS apps. These frameworks are pre-built mechanisms for the appearance and behavior of iPhone apps.
- They also provide the basic app infrastructure and support for key technologies such as multitasking, touch-based input, push notifications, and many system services.
- When designing your apps, you should investigate the technologies here first to see if they meet your needs.
- From the Apple docs here.

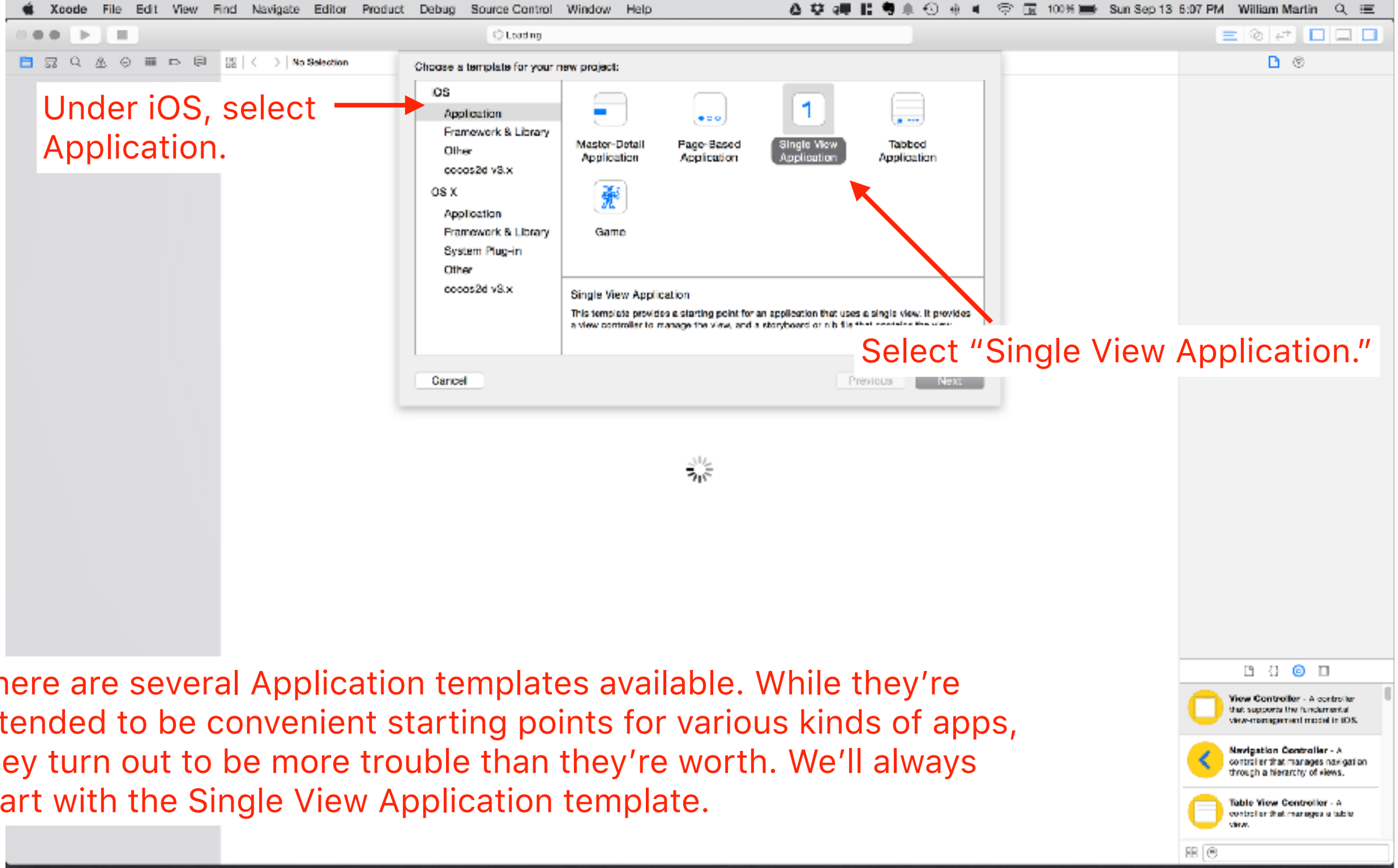
WHAT GOES INTO AN APP?

- User Interfaces - Built by composing "Views" together, provided by the UIKit framework.
- Code - Written in Swift.
- Tests - Code that ensures that the app code works well.
- Assets - Images, videos, sound files, etc.

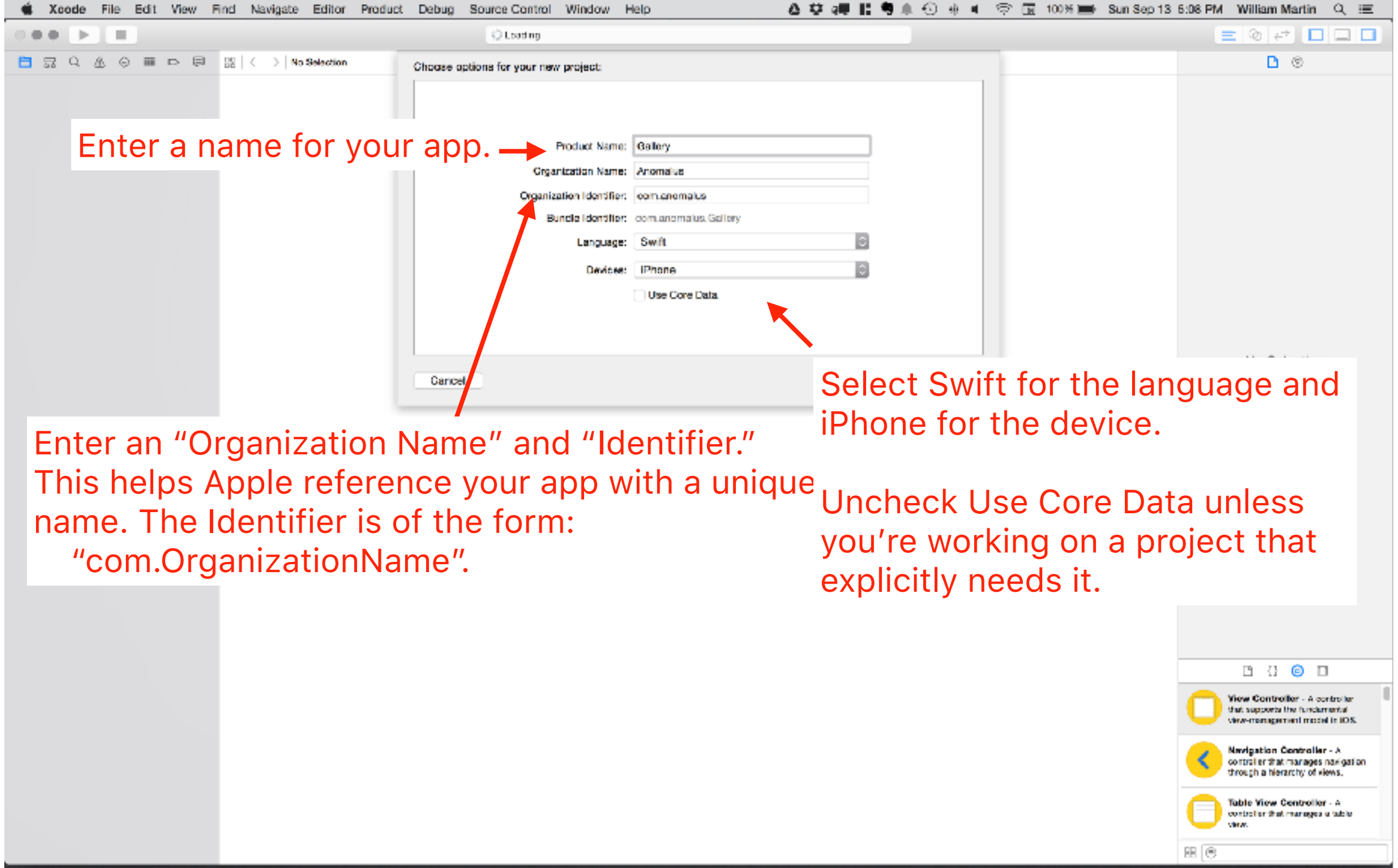
XCODE PROJECT

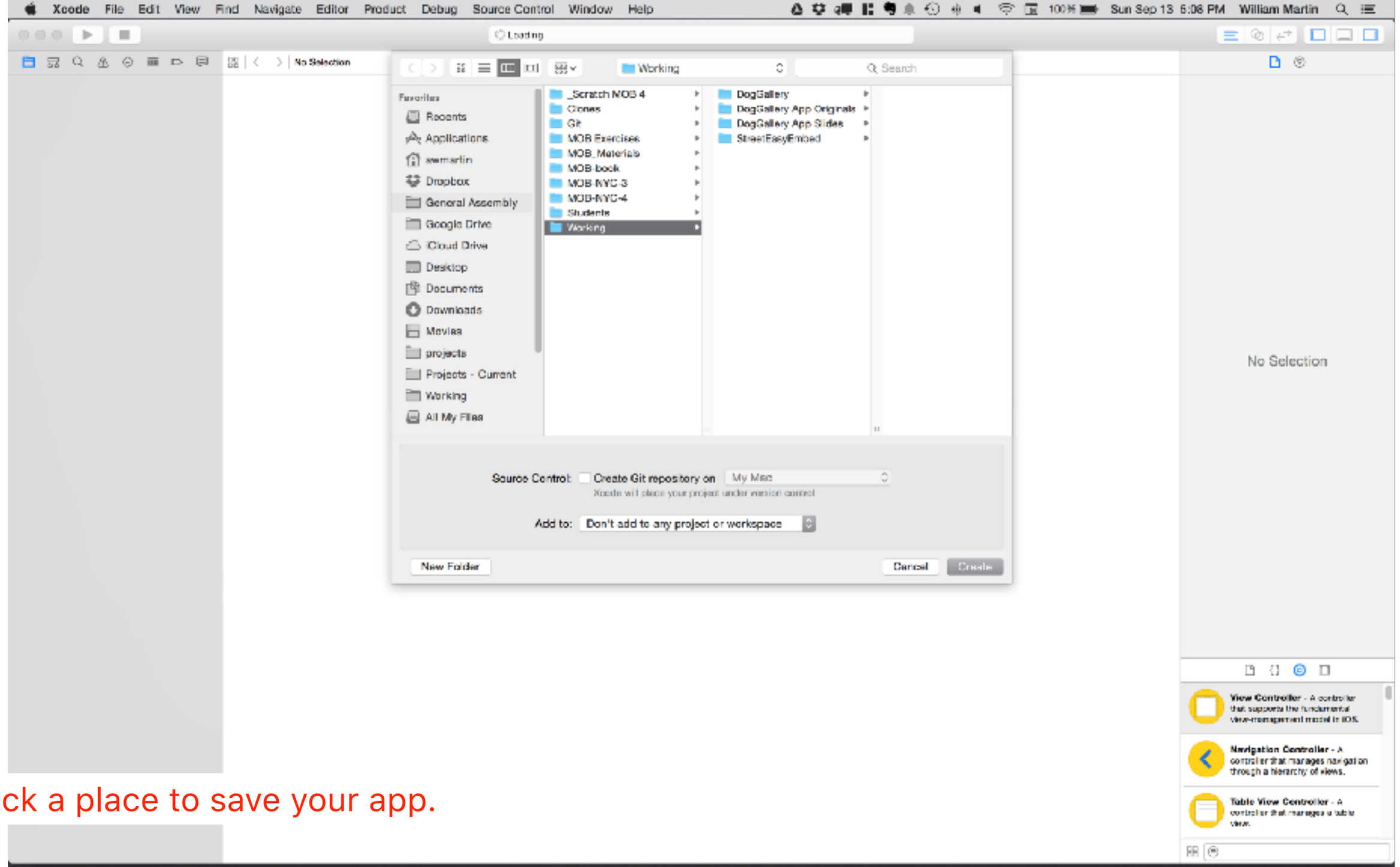


Start creating a project by opening Xcode and going to File > New > Project...

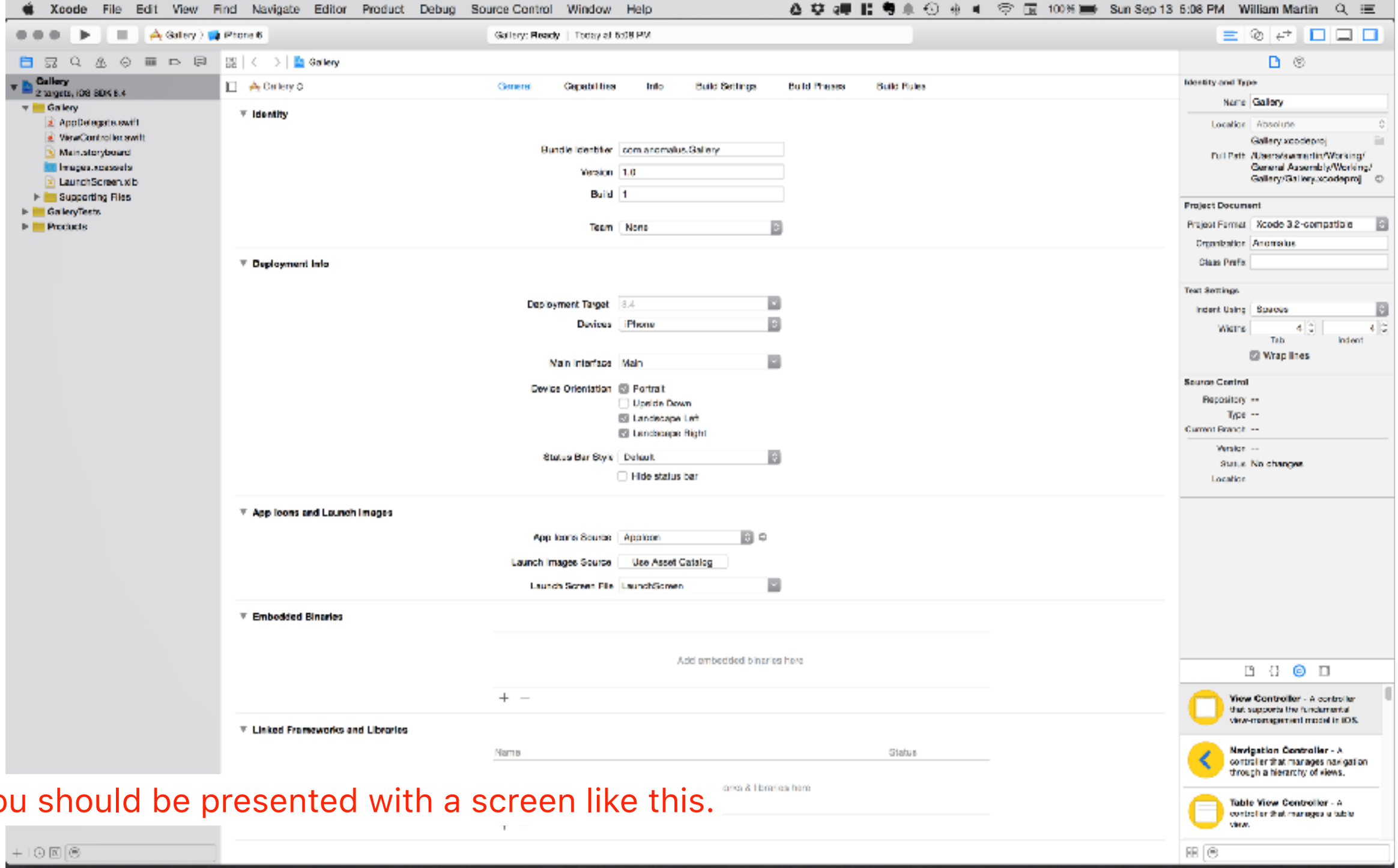


There are several Application templates available. While they’re intended to be convenient starting points for various kinds of apps, they turn out to be more trouble than they’re worth. We’ll always start with the Single View Application template.


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Sun Sep 13 6:06 PM William Martin
Loading
No Selection
Choose options for your new project:
Product Name: Gallery
Organization Name: Aromatus
Organization Identifier: com.aromatus
Bundle Identifier: com.aromatus.Gallery
Language: Swift
Devices: iPhone
Use Core Data
Cancel
Enter a name for your app.
Enter an “Organization Name” and “Identifier.” This helps Apple reference your app with a unique name. The Identifier is of the form:
“com.OrganizationName”.
Select Swift for the language and iPhone for the device.
Uncheck Use Core Data unless you’re working on a project that explicitly needs it.
View Controller - A controller that supports the fundamental view-management model in iOS.
Navigation Controller - A controller that manages navigation through a hierarchy of views.
Table View Controller - A controller that manages a table view.



Pick a place to save your app.



You should be presented with a screen like this.

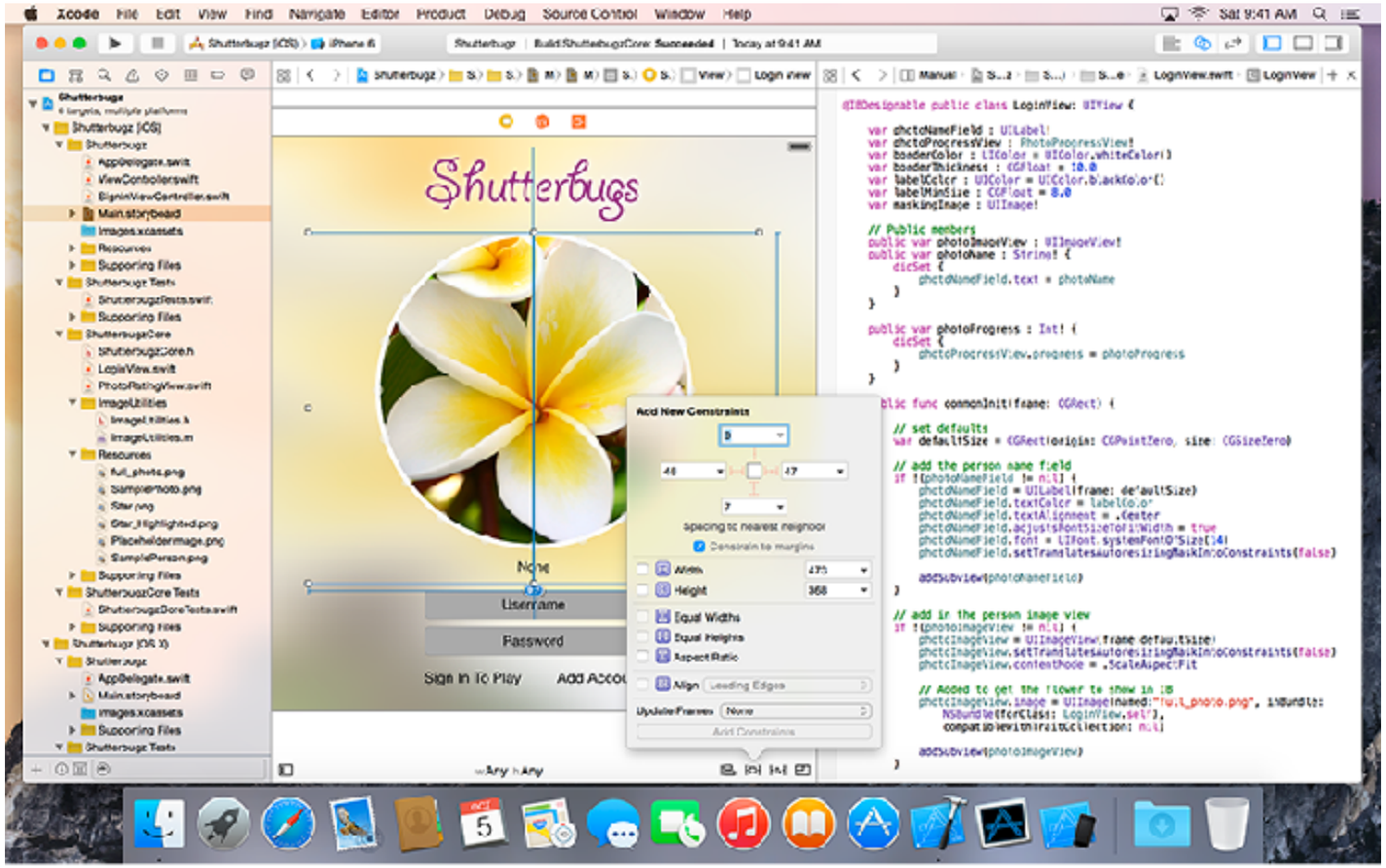
GETTING STARTED

XCODE TOUR

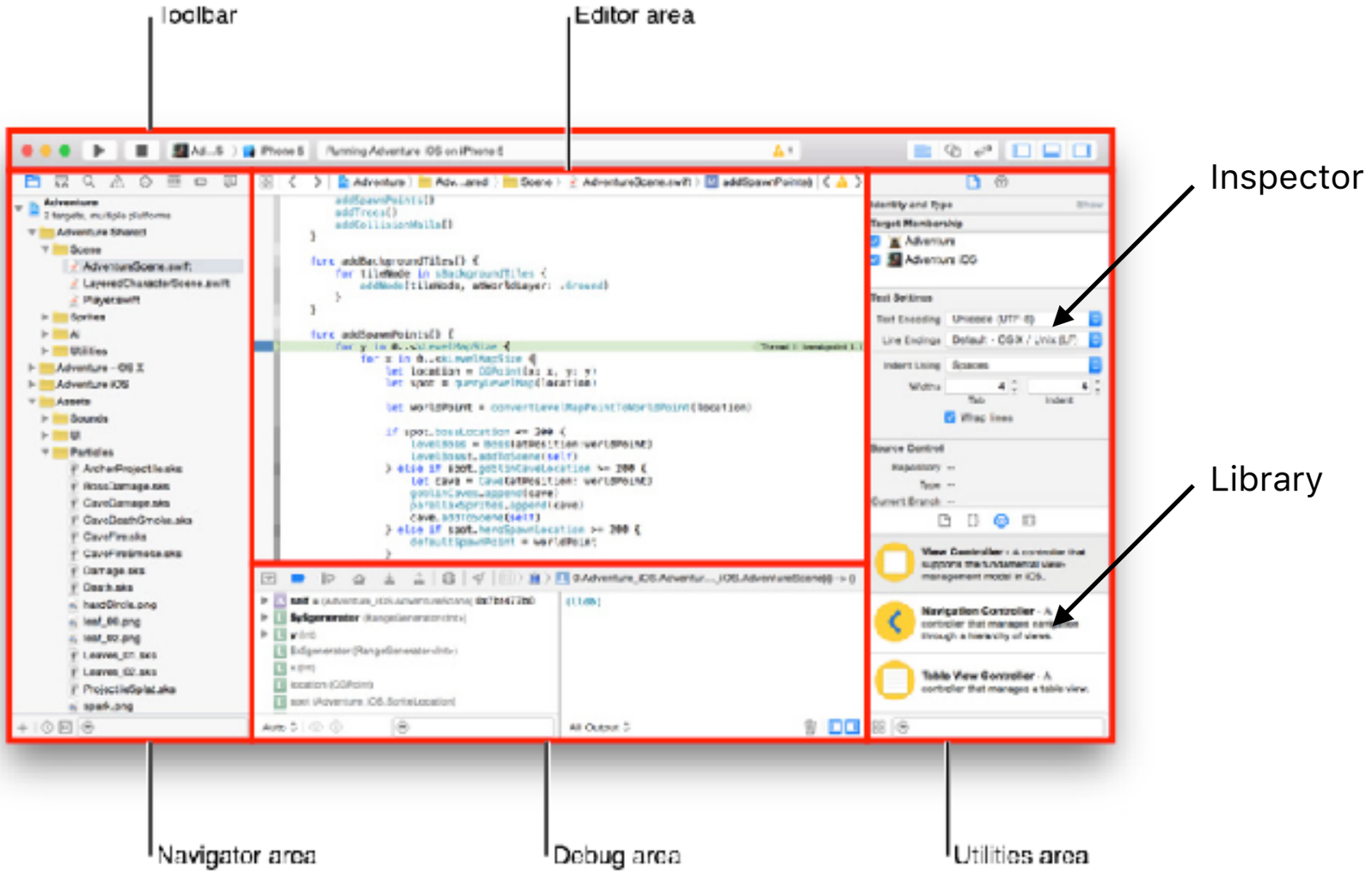
DEV WORKFLOW OVERVIEW

- Launch Xcode
- Create new project
- Briefly discuss the different project templates
- Add user interface elements to project
- Change user interface element properties
- Build / run the app / test it
- Iterate
- Publish

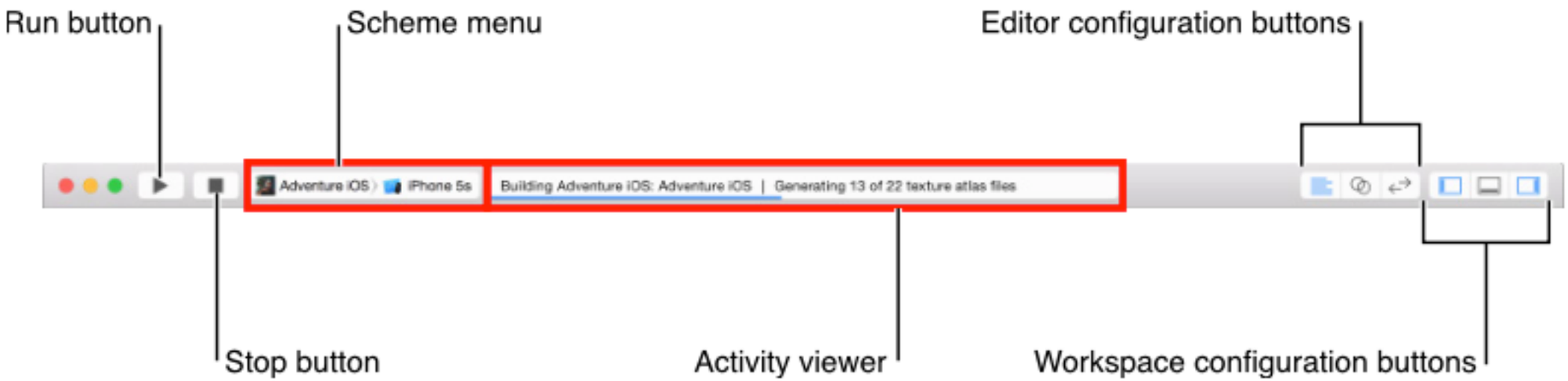
WHAT IS XCODE?



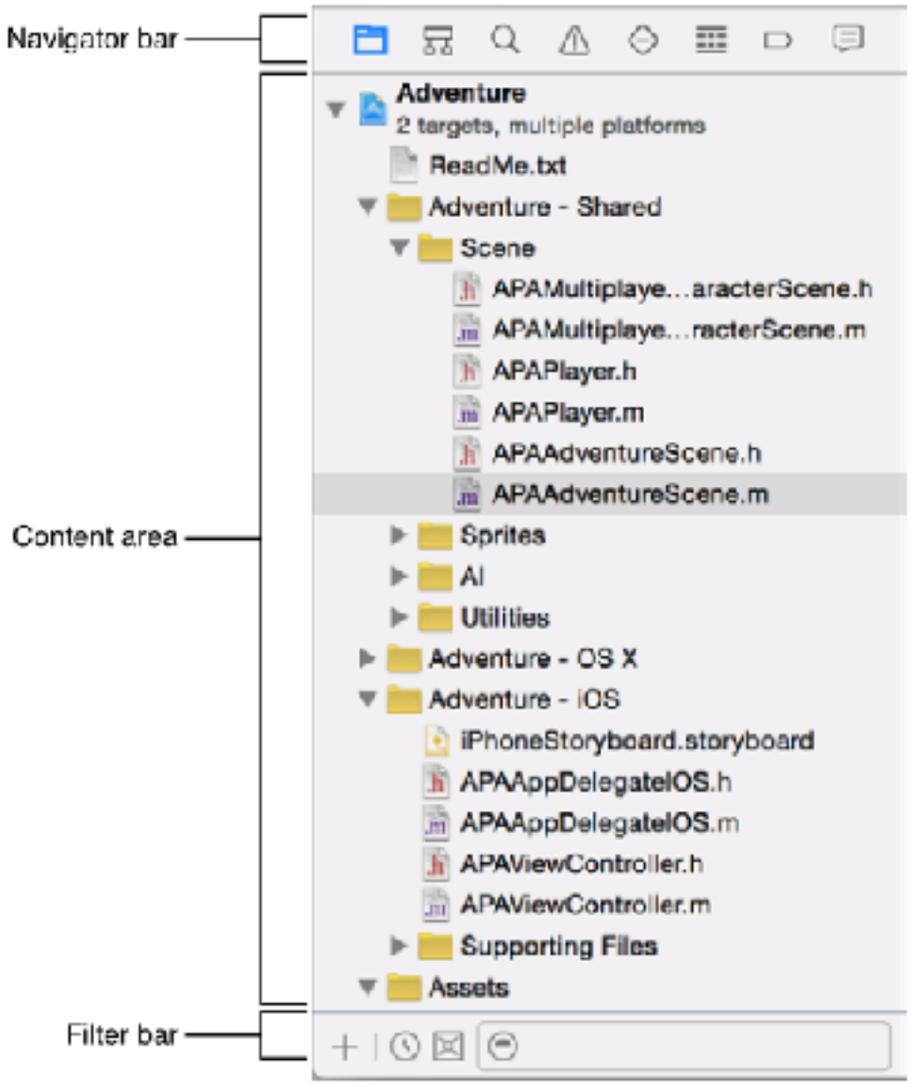
NAVIGATING XCODE



WORKSPACE TOOLBAR

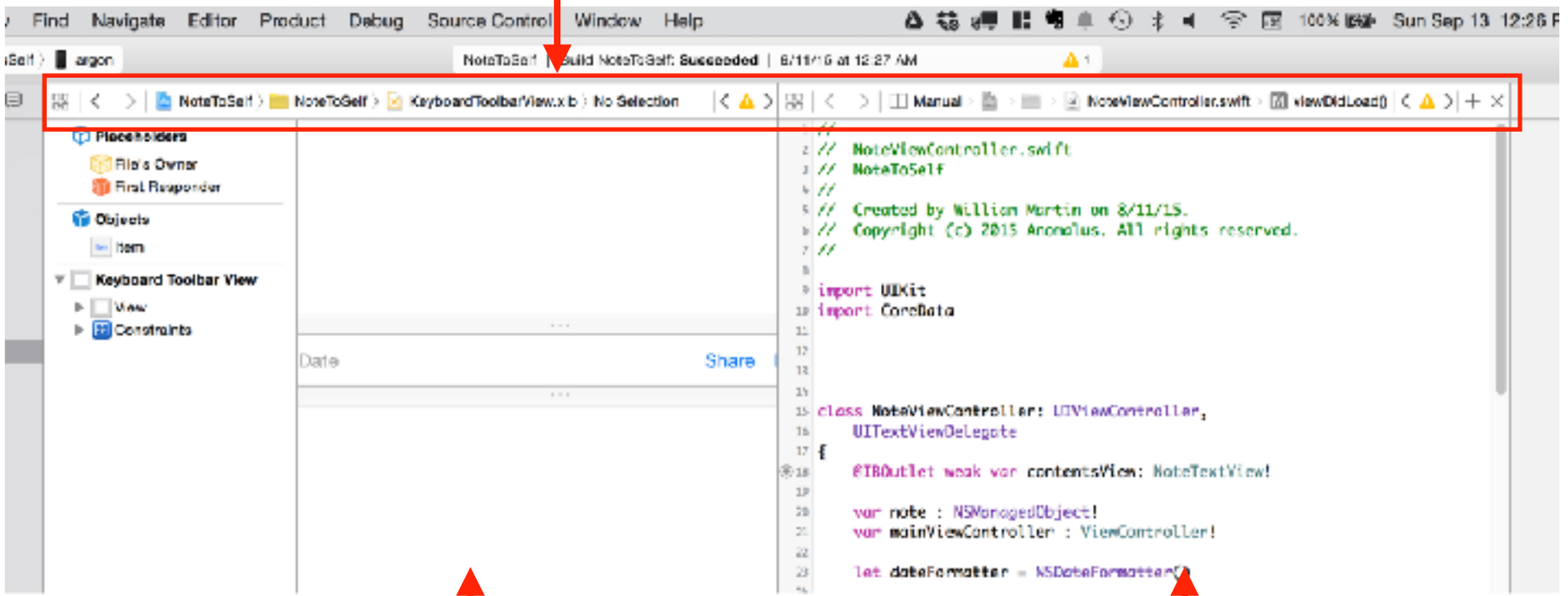


NAVIGATOR AREA



JUMP BAR AND EDITOR PANES

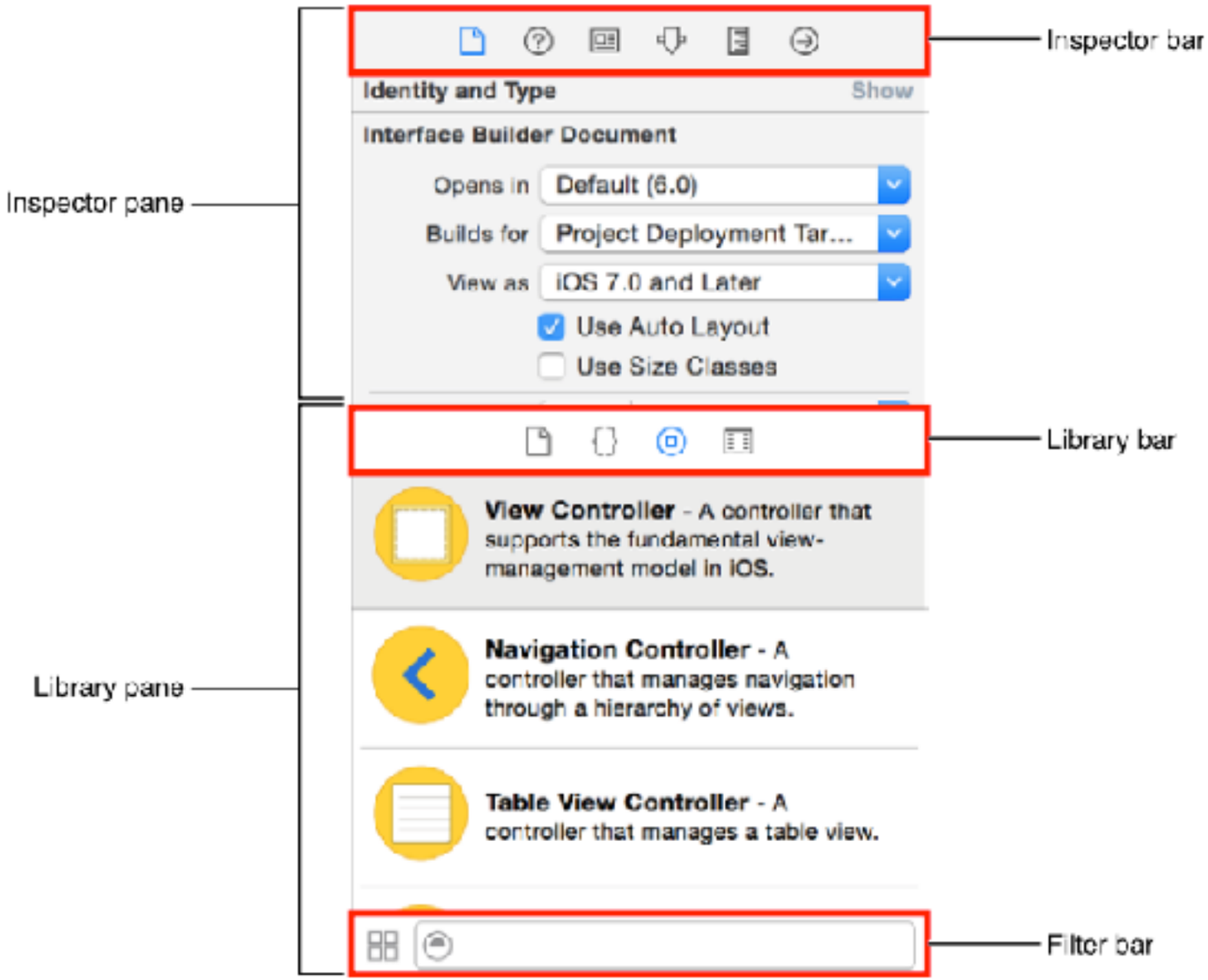
Jump bars



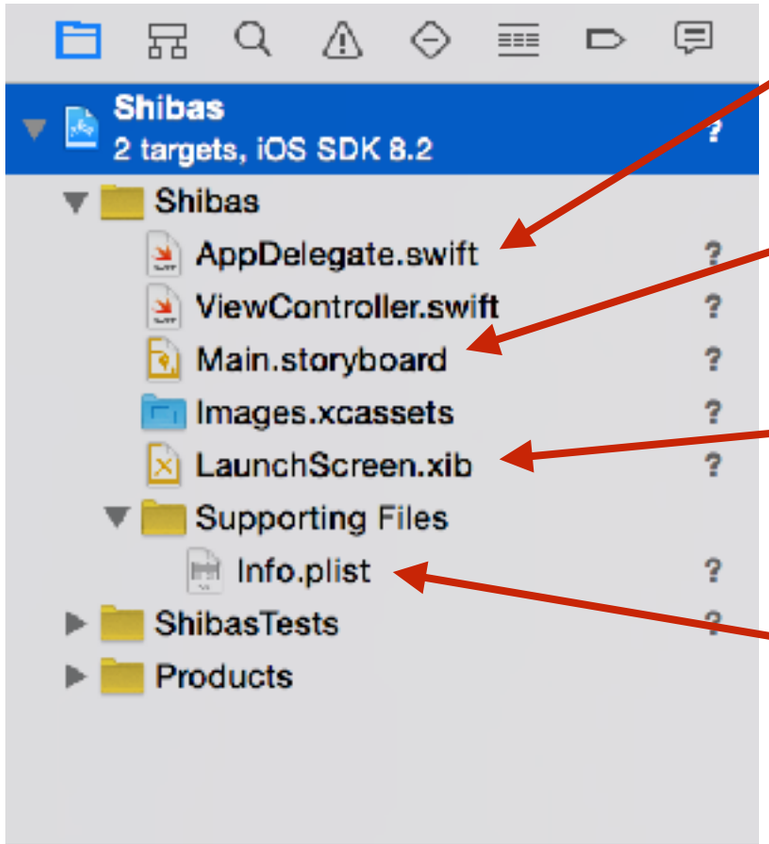
Editor pane

Assistant editor pane

UTILITIES AREA



FILE TYPES



- .swift
  Swift source code file
- .storyboard
  an Interface Builder “storyboard” file
- .xib
  an Interface Builder “NIB” file
- .plist
  a “property list”

RUNNING YOUR APP

RUNNING YOUR APP

- Run app on simulator (Cmd + R or click Play)
- BONUS: Outline the steps for deploying an app to device

RUNNING AN APP IN THE IOS SIMULATOR

- Select iOS version in toolbar area
- Select “Build and then Run” in toolbar area (⌘R)

Note

- iPad apps only run on iPad simulator
- iPhone and universal apps run on both iPad and iPhone simulators

NAVIGATING THE IOS SIMULATOR

- To run Simulator without running a project, select:
  Xcode -> Open Developer Tool -> iOS Simulator
- To select the “Home” button on simulator press ⌘ + ⇧ + H.

GETTING THINGS ON THE SCREEN

GETTING VIEWS ON SCREEN

- To start understanding iOS apps, we'll first tackle 'views' on the screen
- Almost everything we see on screen is a view.
- There are lots of kinds of views:
  - Buttons, labels, tables, images, etc
- There are several ways to lay things out on screen, we'll cover these later in class
  - Until then, our views may look a little misaligned.

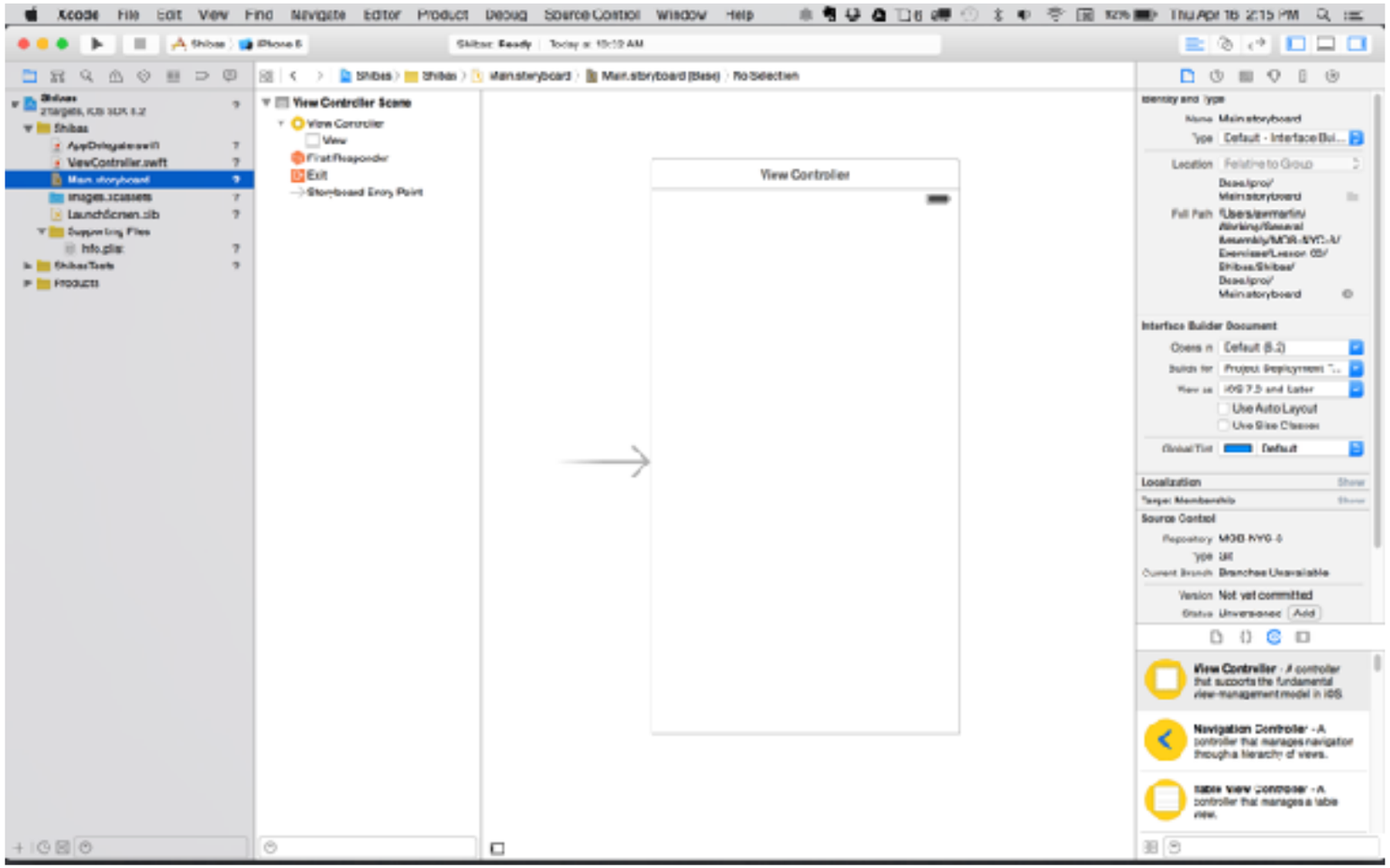
VIEWS ON THE WHITEBOARD

BUILDING USER INTERFACES

HOW DOES XCODE ENABLE YOU TO BUILD UIS?

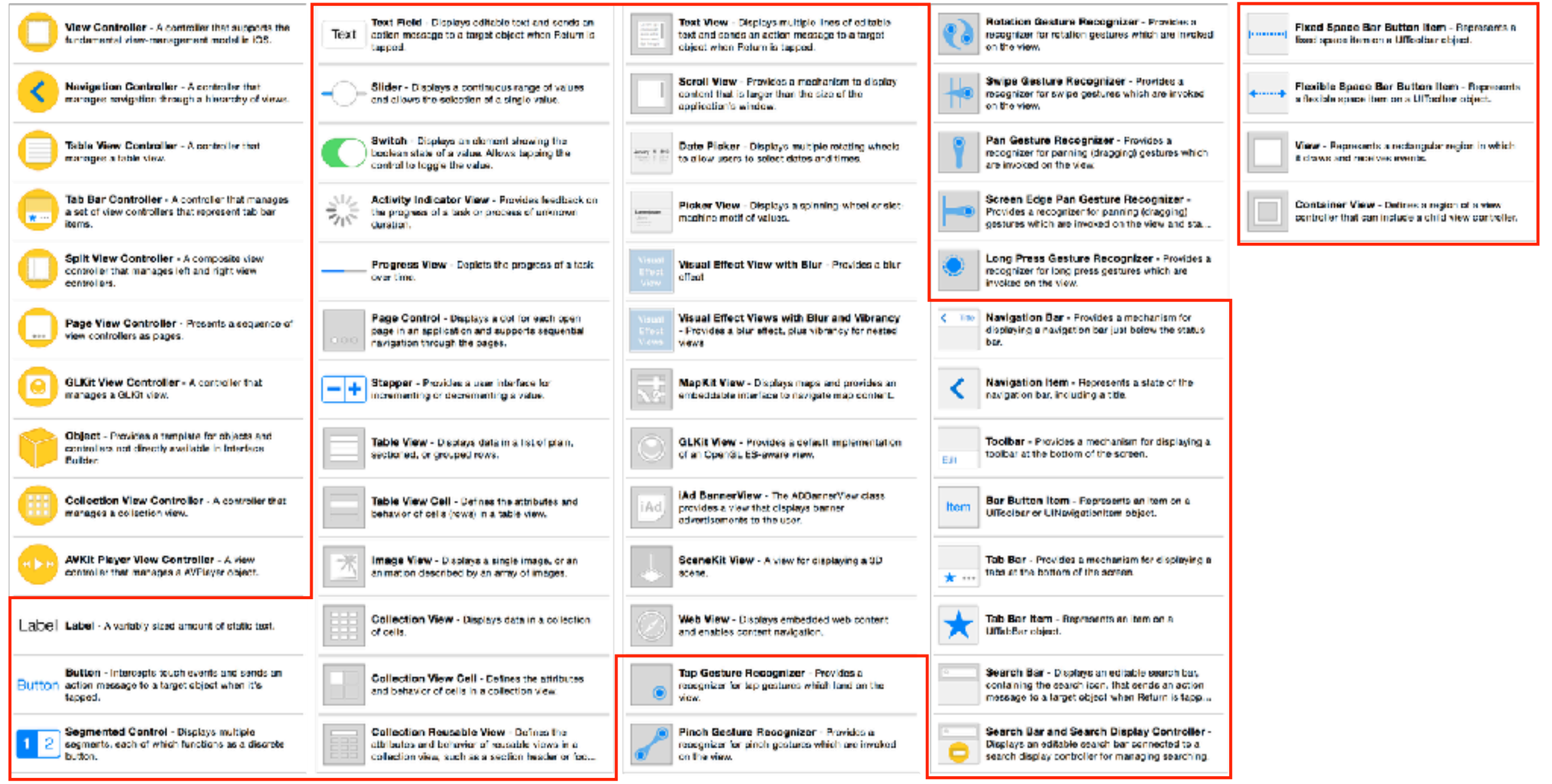
- A tool called "Interface Builder" that enables you to build "Storyboards."
- A Storyboard is a file that enables you to compose:
  - multiple Scenes,
  - the transitions between them, and
  - their respective Views
- all in a single place.

INTERFACE BUILDER



MORE ON VIEWS

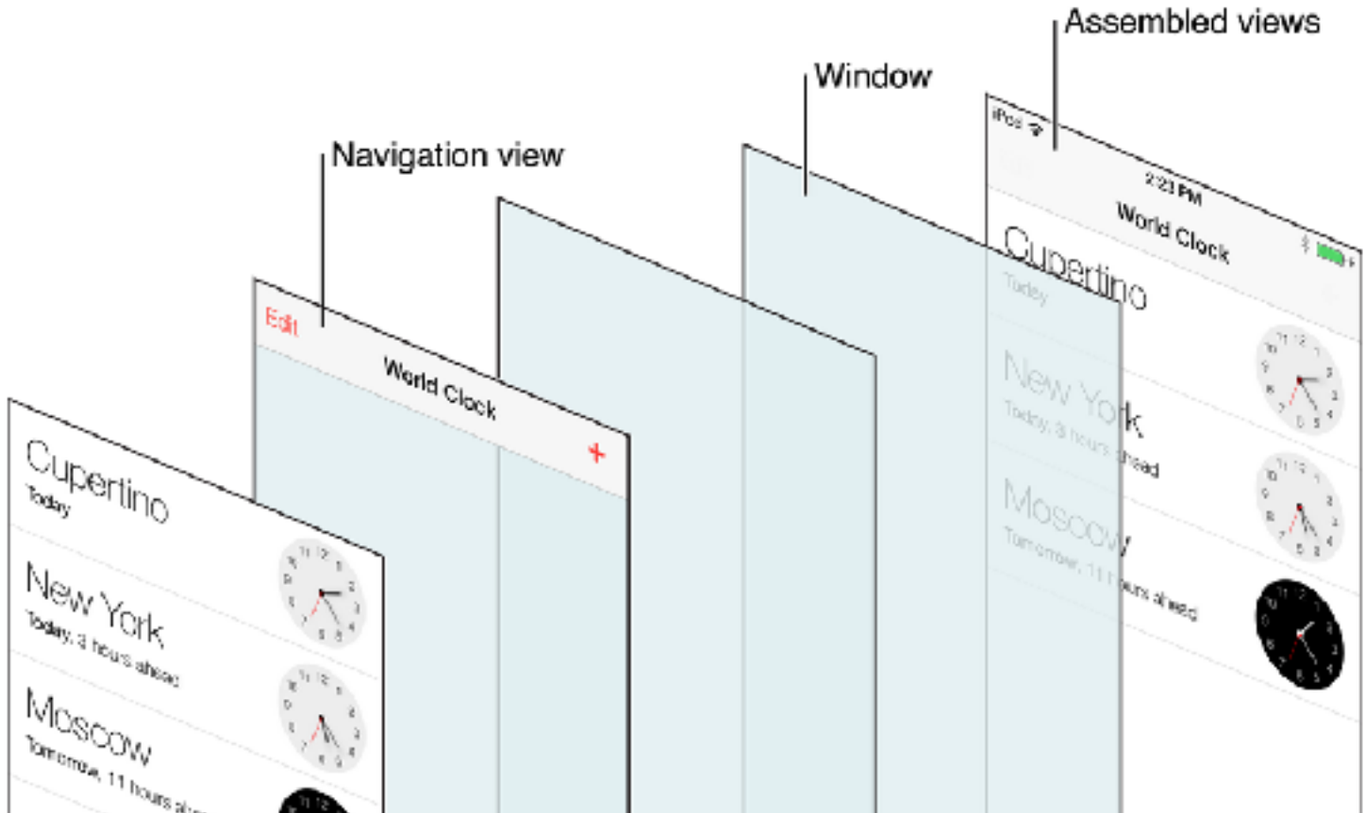
- A View is a UI element, typically rectangular, drawn at some location (or “position”) with a size.
- Views can “contain” other views (i.e. “subviews”).
- Most views have some level of interactivity (e.g. scrolling, detecting taps or other gestures, etc.).
- You can also develop your own Views.


View Controller - A controller that supports the fundamental view-management model in iOS.
Navigation Controller - A controller that manages navigation through a hierarchy of views.
Table View Controller - A controller that manages a table view.
Tab Bar Controller - A controller that manages a set of view controllers that represent tab bar items.
Split View Controller - A composite view controller that manages left and right view controllers.
Page View Controller - Presents a sequence of view controllers as pages.
GLKit View Controller - A controller that manages a GLKit view.
Object - Provides a template for objects and controllers not directly available in Interface Builder.
Collection View Controller - A controller that manages a collection view.
AVKit Player View Controller - A view controller that manages a AVPlayer object.
Label Label - A variably sized amount of static text.
Button Button - Intercepts touch events and sends an action message to a target object when it's tapped.
1 2 Segmented Control - Displays multiple segments, each of which functions as a discrete button.
Text Text Field - Displays editable text and sends an action message to a target object when Return is tapped.
Slider - Displays a continuous range of values and allows the selection of a single value.
Switch - Displays an element showing the boolean state of a value. Allows tapping the control to toggle the value.
Activity Indicator View - Provides feedback on the progress of a task or process of unknown duration.
Progress View - Depicts the progress of a task over time.
Page Control - Displays a dot for each open page in an application and supports sequential navigation through the pages.
Stepper - Provides a user interface for incrementing or decrementing a value.
Table View - Displays data in a list of plain, sectioned, or grouped rows.
Table View Cell - Defines the attributes and behavior of cells (rows) in a table view.
Image View - Displays a single image, or an animation described by an array of images.
Collection View - Displays data in a collection of cells.
Collection View Cell - Defines the attributes and behavior of cells in a collection view.
Collection Reusable View - Defines the attributes and behavior of reusable views in a collection view, such as a section header or foo...
Text View - Displays multiple lines of editable text and sends an action message to a target object when Return is tapped.
Scroll View - Provides a mechanism to display content that is larger than the size of the application's window.
Date Picker - Displays multiple rotating wheels to allow users to select dates and times.
Picker View - Displays a spinning wheel or slot machine motif of values.
Visual Effect View with Blur - Provides a blur effect
Visual Effect Views with Blur and Vibrancy - Provides a blur effect, plus vibrancy for nested views
MapKit View - Displays maps and provides an embeddable interface to navigate map content.
GLKit View - Provides a default implementation of an OpenGL ES-aware view.
iAd BannerView - The ADBannerView class provides a view that displays banner advertisements to the user.
SceneKit View - A view for displaying a 3D scene.
Web View - Displays embedded web content and enables content navigation.
Tap Gesture Recognizer - Provides a recognizer for tap gestures which land on the view.
Pinch Gesture Recognizer - Provides a recognizer for pinch gestures which are invoked on the view.
Rotation Gesture Recognizer - Provides a recognizer for rotation gestures which are invoked on the view.
Swipe Gesture Recognizer - Provides a recognizer for swipe gestures which are invoked on the view.
Pan Gesture Recognizer - Provides a recognizer for panning (dragging) gestures which are invoked on the view.
Screen Edge Pan Gesture Recognizer - Provides a recognizer for panning (dragging) gestures which are invoked on the view and sta...
Long Press Gesture Recognizer - Provides a recognizer for long press gestures which are invoked on the view.
Navigation Bar - Provides a mechanism for displaying a navigation bar just below the status bar.
Navigation Item - Represents a state of the navigation bar, including a title.
Edit Toolbar - Provides a mechanism for displaying a toolbar at the bottom of the screen.
Item Bar Button Item - Represents an item on a UIToolbar or UINavigationItem object.
Tab Bar - Provides a mechanism for displaying a tabs at the bottom of the screen.
Tab Bar Item - Represents an item on a UITabBar object.
Search Bar - Displays an editable search bar, containing the search icon, that sends an action message to a target object when Return is tapp...
Search Bar and Search Display Controller - Displays an editable search bar connected to a search display controller for managing searching.
Fixed Space Bar Button Item - Represents a fixed space item on a UIToolbar object.
Flexible Space Bar Button Item - Represents a flexible space item on a UIToolbar object.
View - Represents a rectangular region in which it draws and receives events.
Container View - Defines a region of a view controller that can include a child view controller.

MORE ON VIEWS

UIKit User Interface Catalog

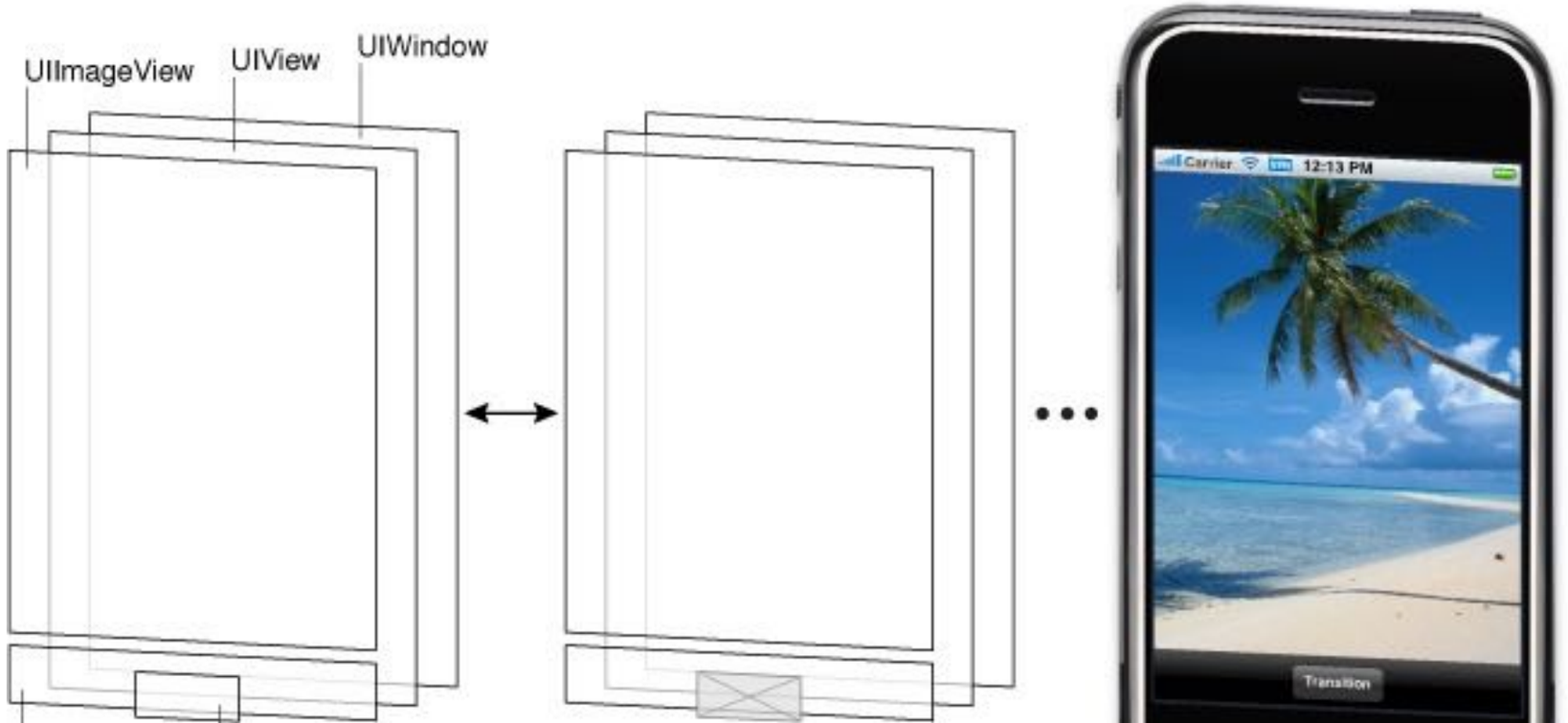
https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/UIKitUICatalog/index.html



MORE ON VIEWS

View Programming Guide for iOS

https://developer.apple.com/library/ios/documentation/WindowsViews/Conceptual/ViewPG_iPhoneOS/Introduction/Introduction.html



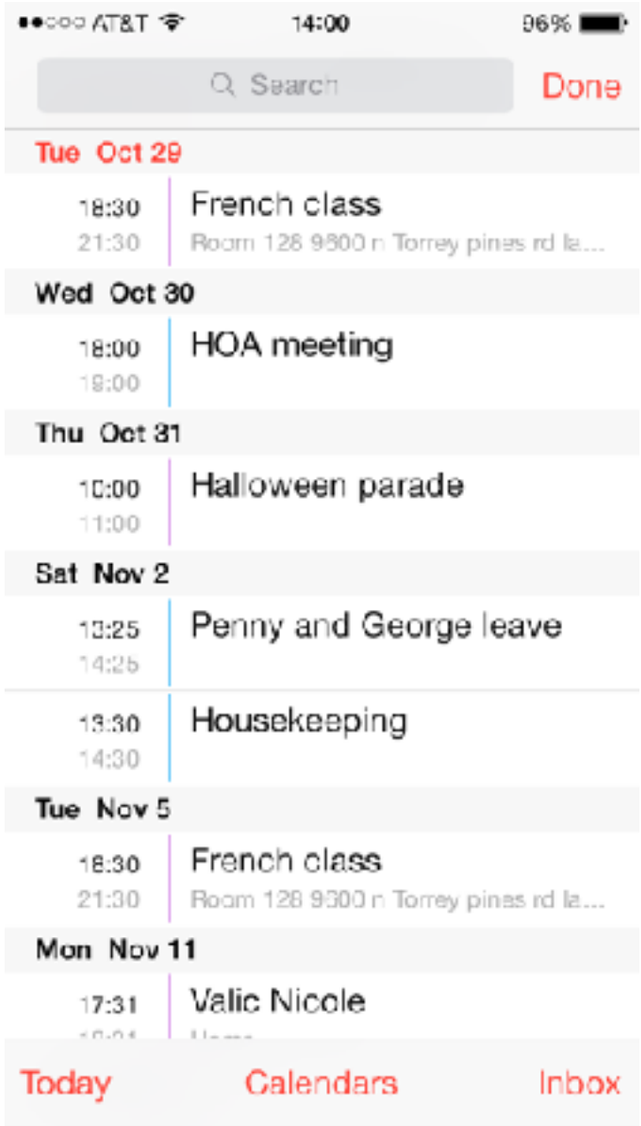
HOW-TO

PRACTICE: ADDING VIEWS

IOS UI BASICS

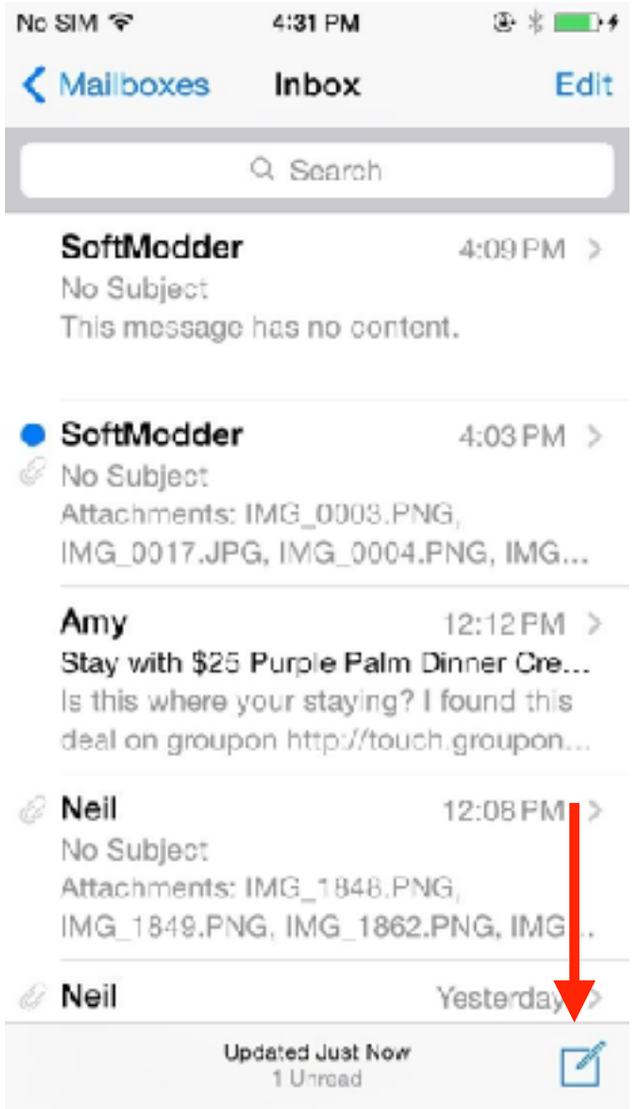
LISTS

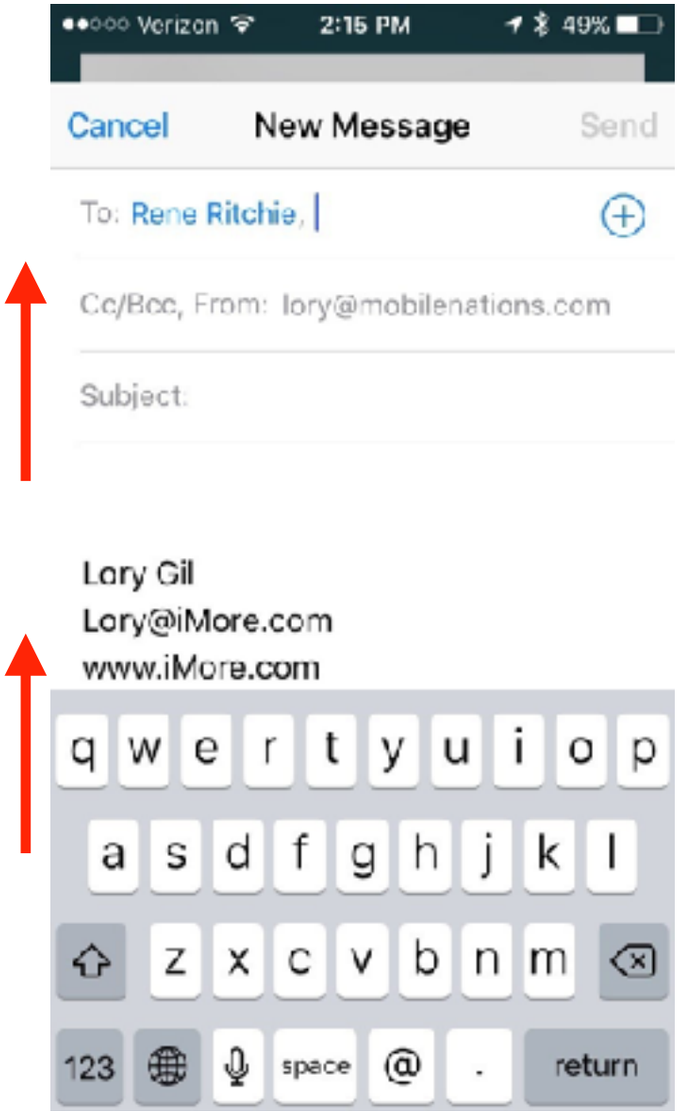
- TableView / CollectionView



USER FLOW

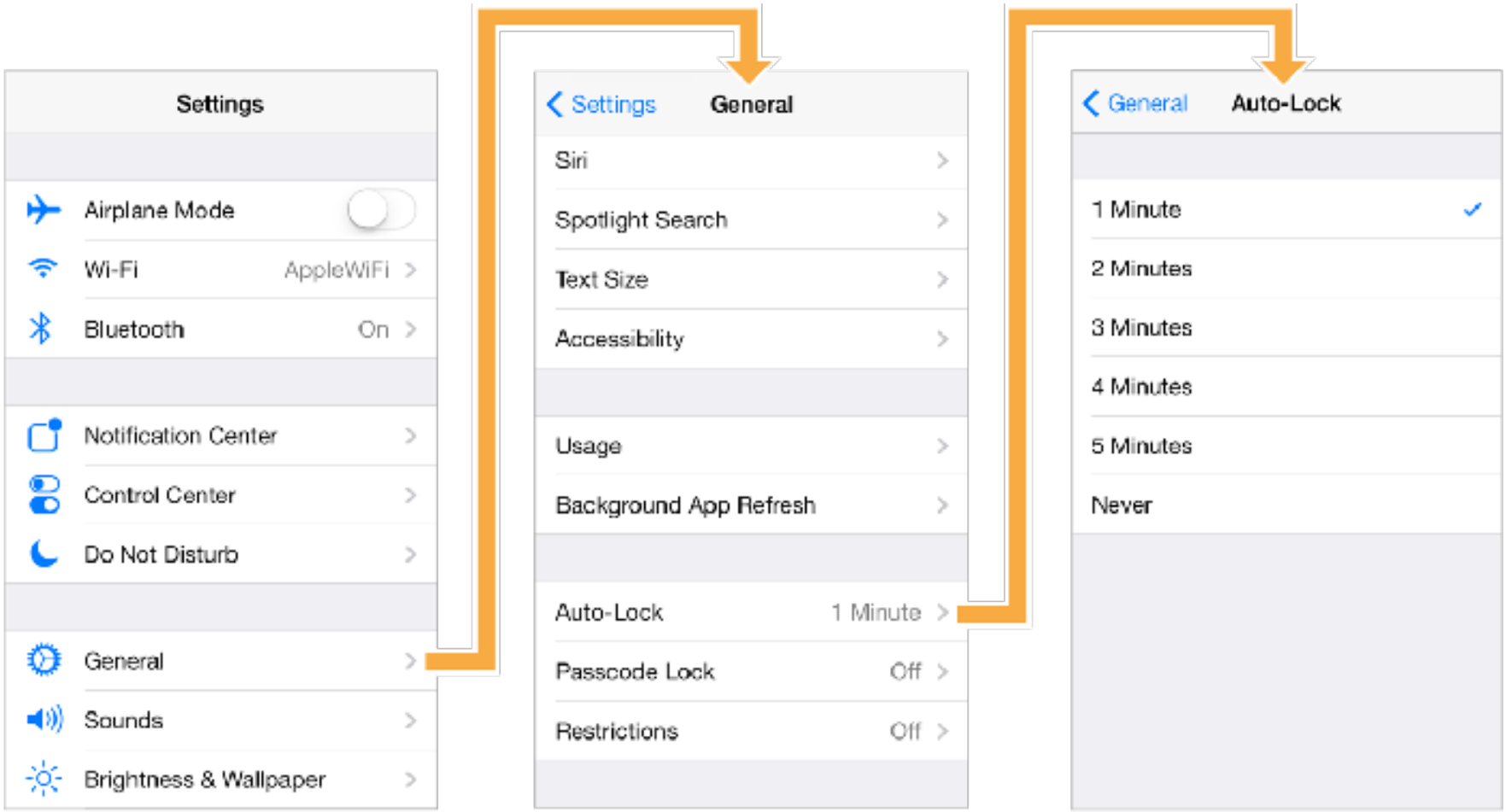
- Modal





USER FLOW

- Push/Pop



STORYBOARDS

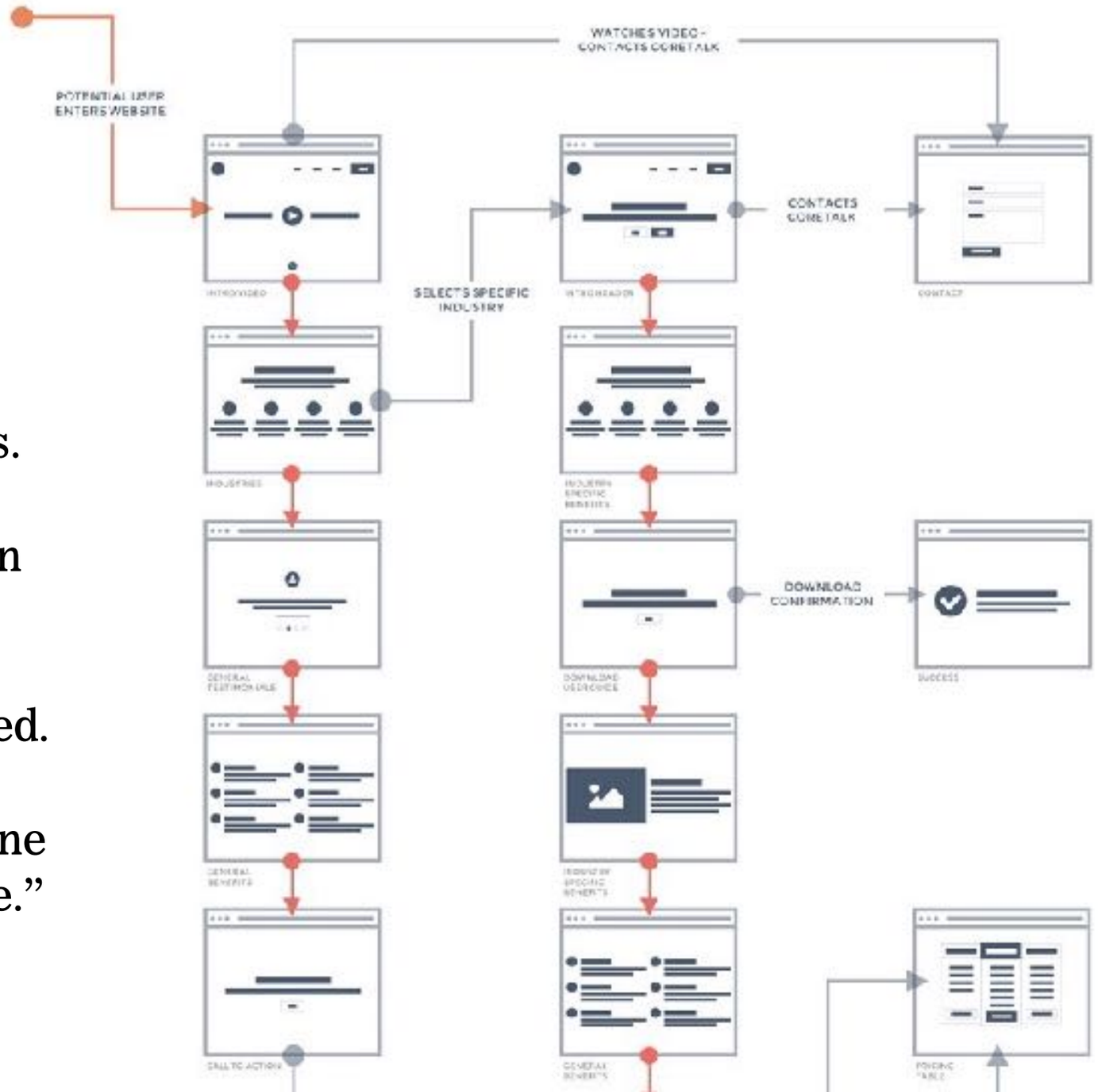
USER INTERFACES

WEBSITE SITEMAPS

Web and UX designers draw sitemap flows that show the structure of a site and describe the “flow” between pages.

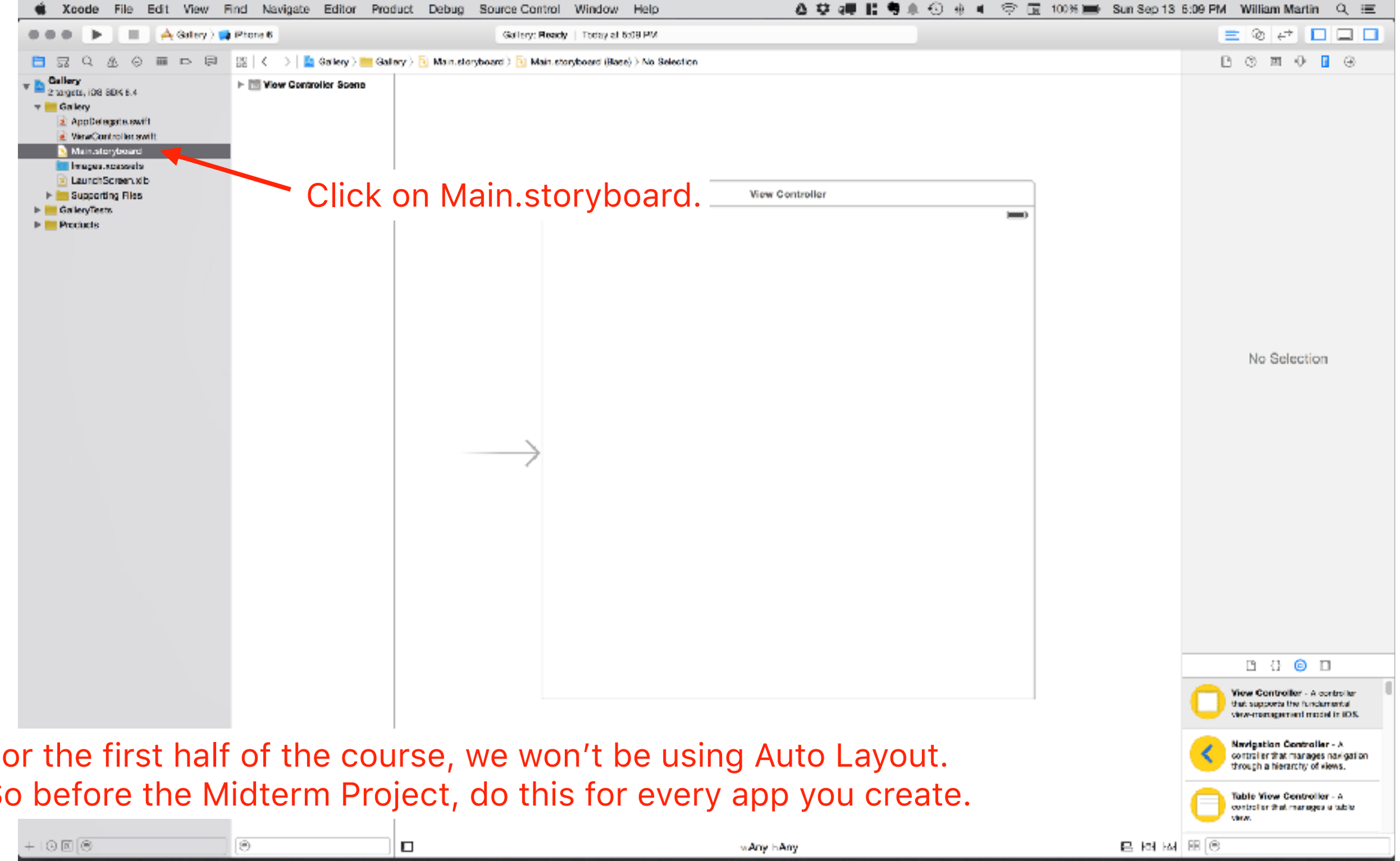
Storyboards are like this. They show an app’s screens (here called “Scenes”, each of which correlates to a View Controller), and how they are connected.

Each line of flow (or transition) from one Scene to the another is called a “Segue.”

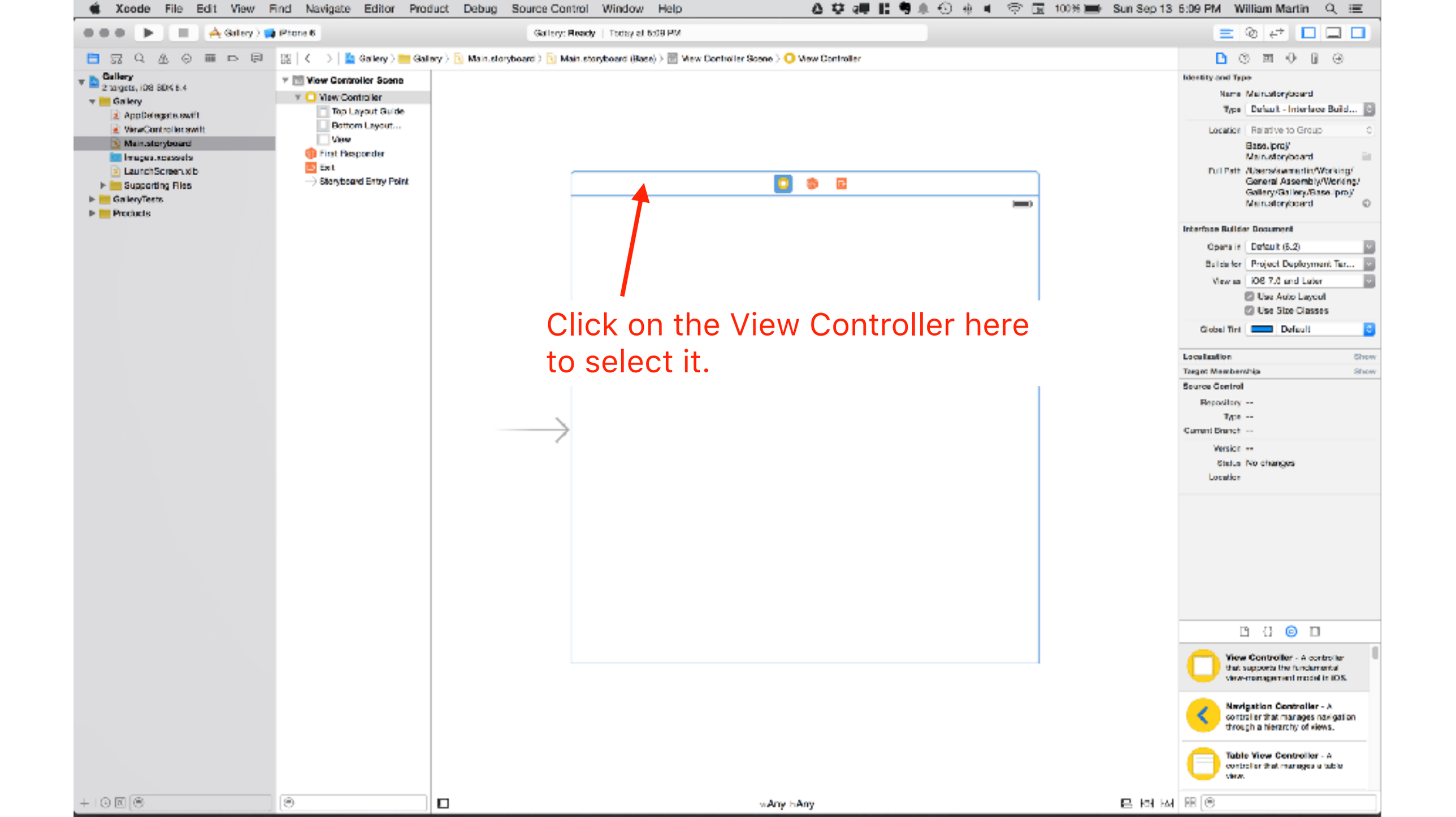


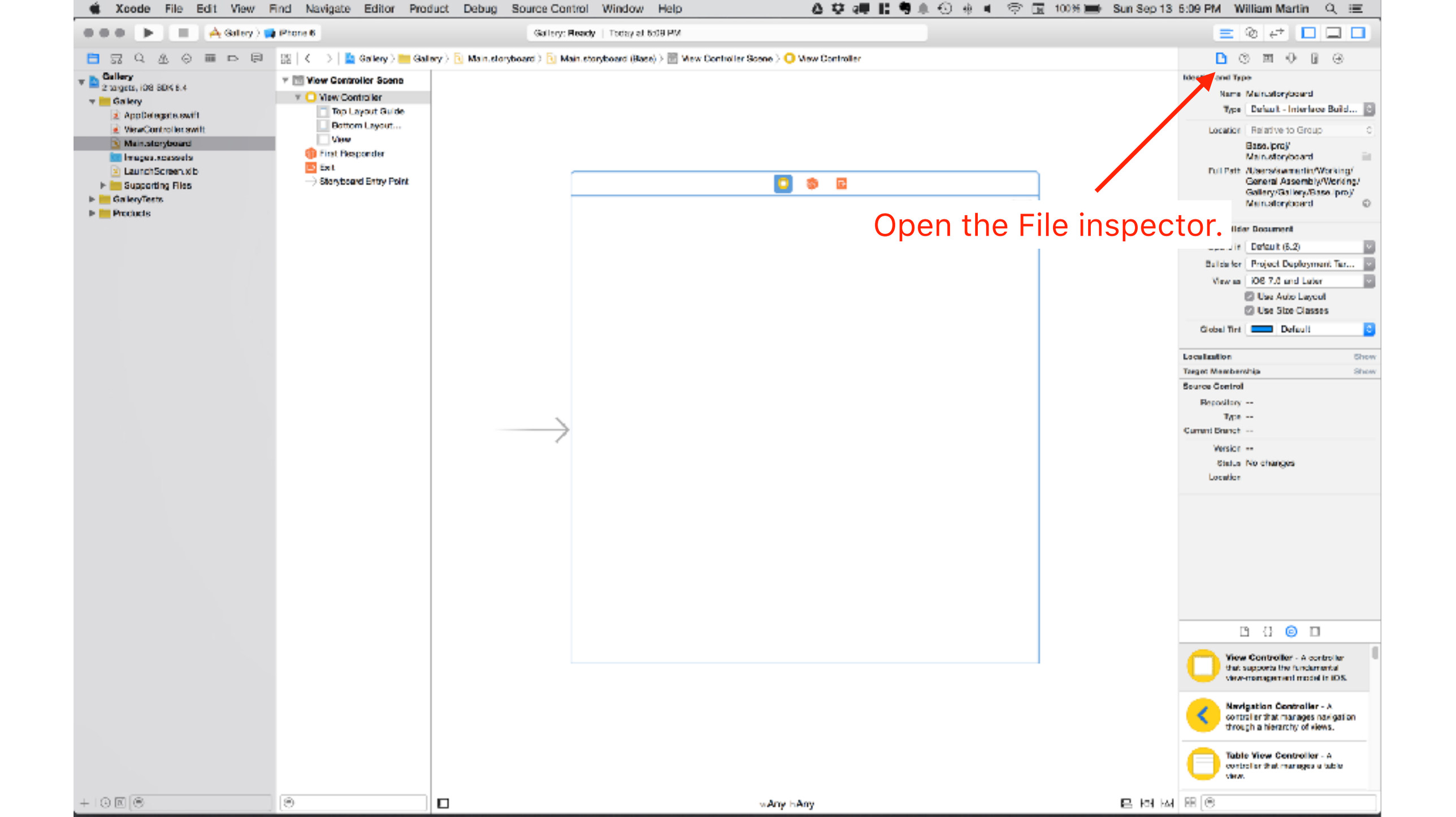
SO, WHAT ARE STORYBOARDS?

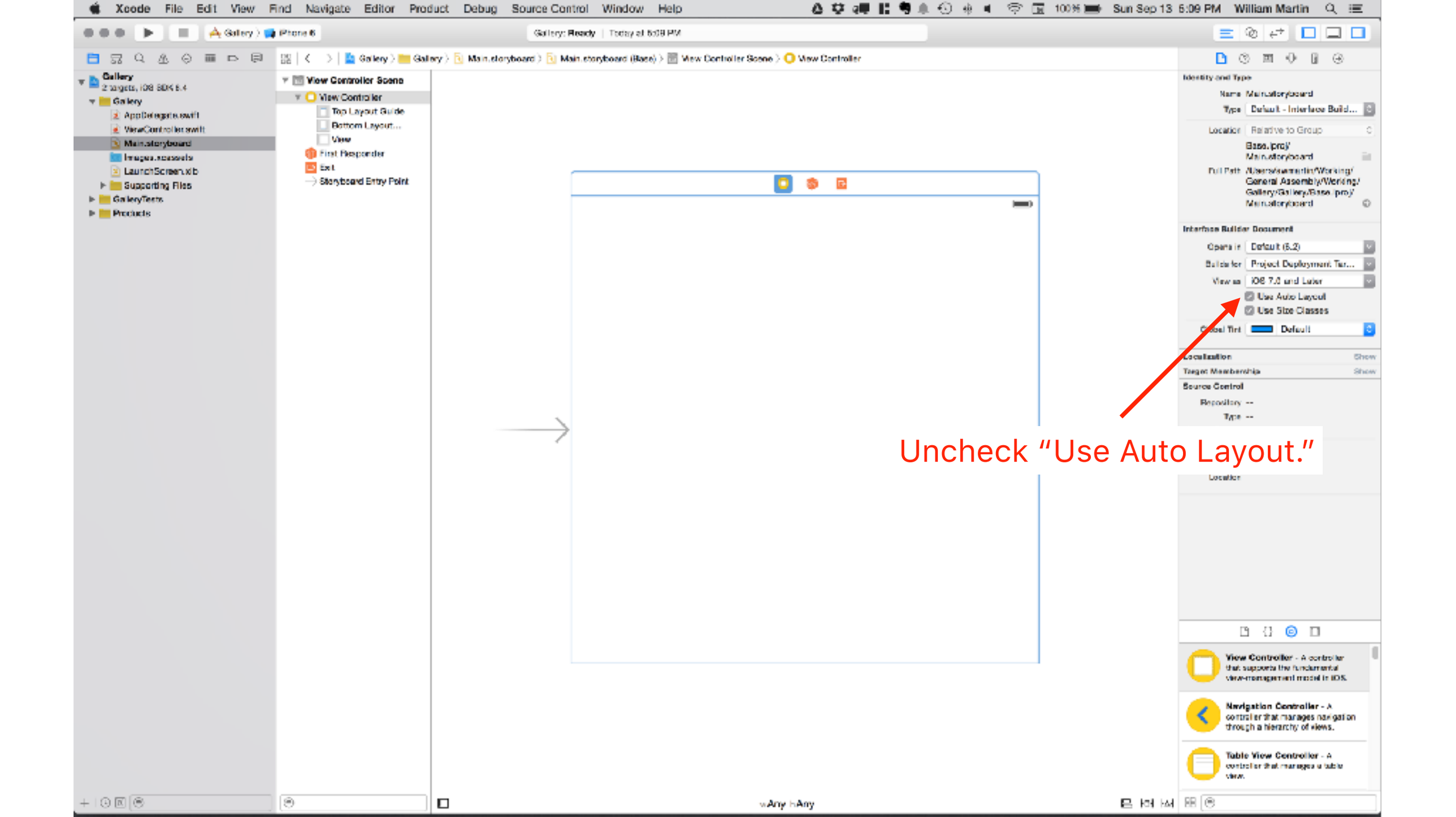
- Storyboards can serve as the scaffolding for the various screens of an app.
- By themselves, they can be deployed as a way to prototype apps without code.
- Unlike sitemaps, however, Storyboards aren’t passive descriptions, they are active documents that we can use to author a significant part of our apps.

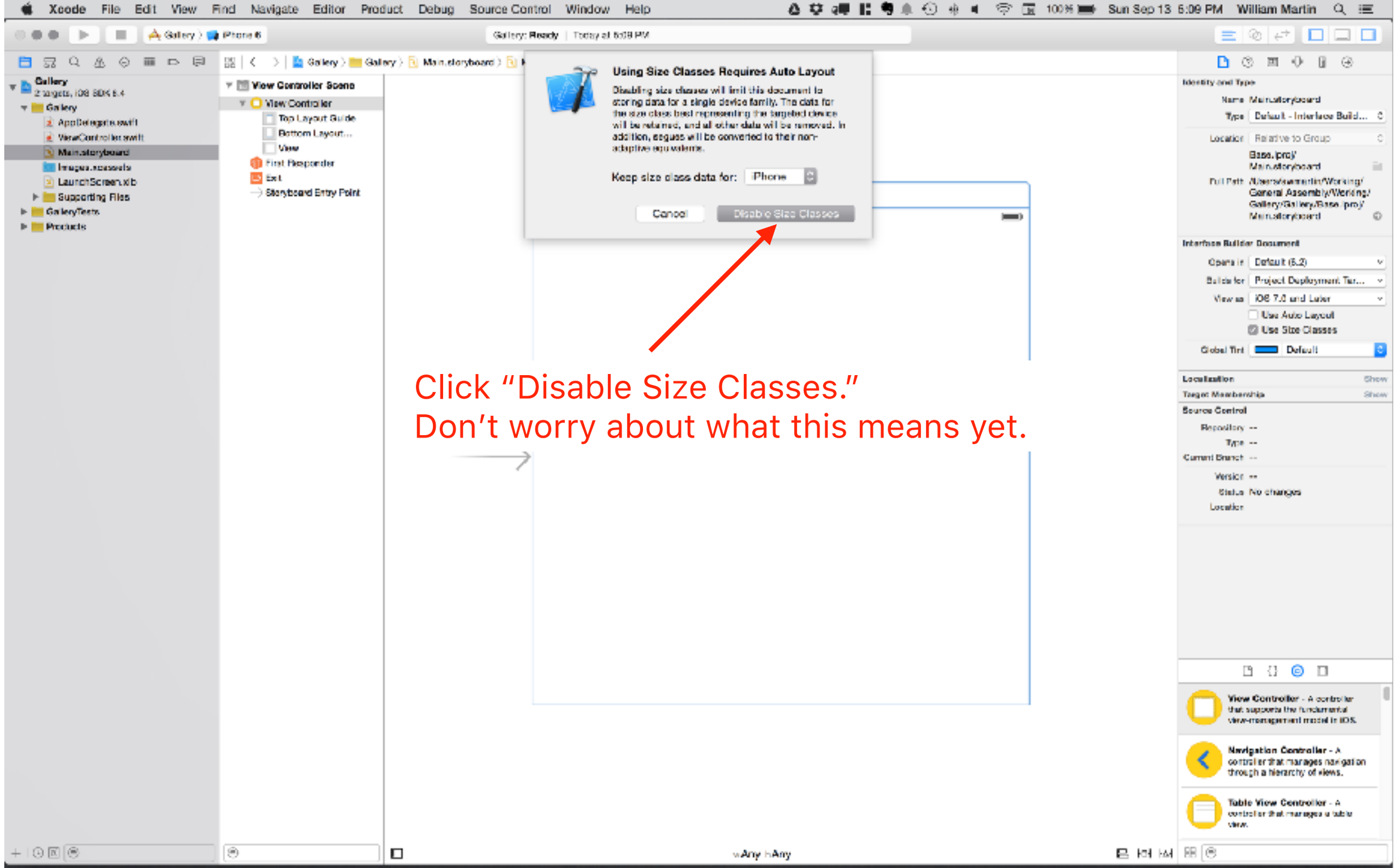


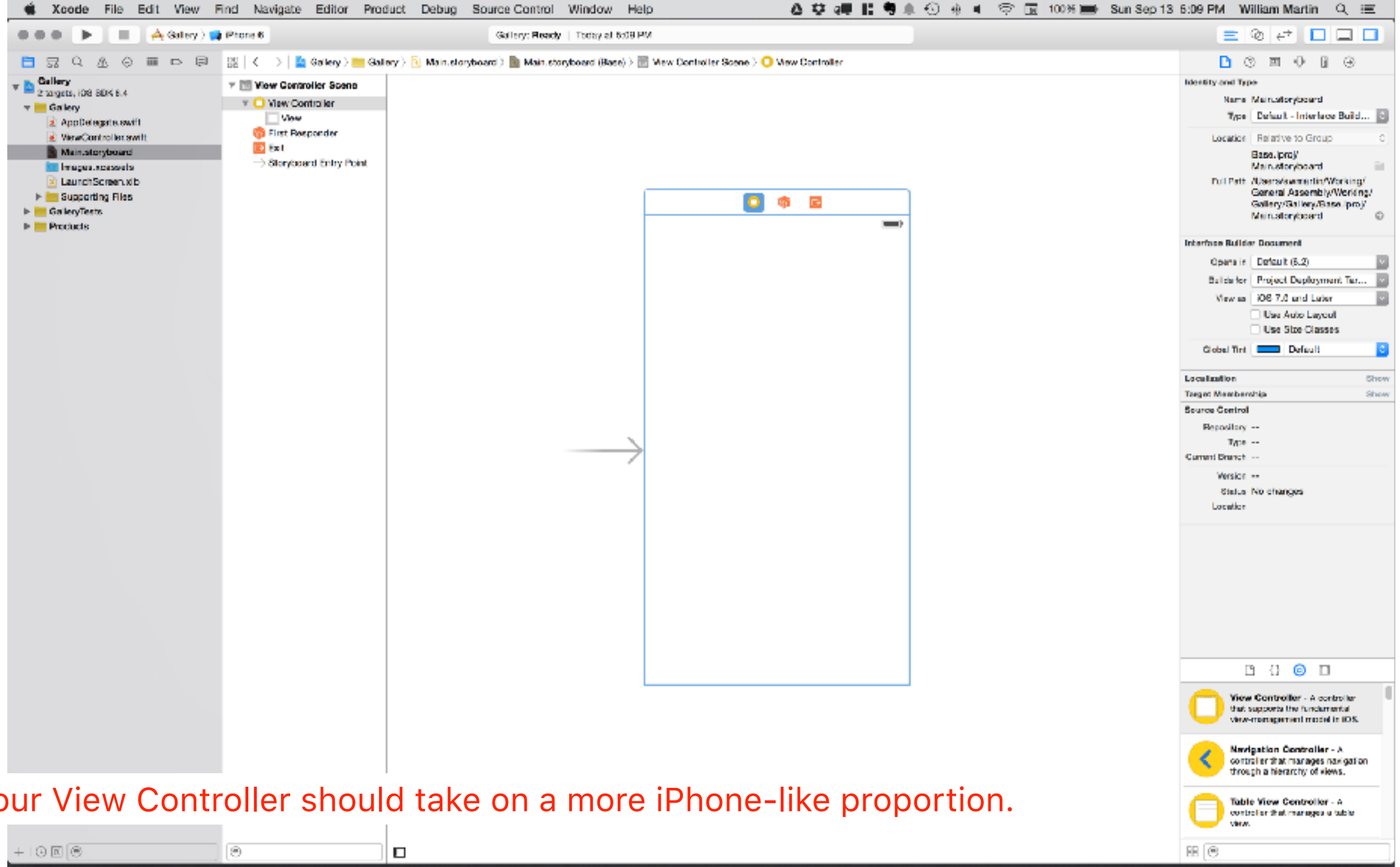
For the first half of the course, we won't be using Auto Layout.
So before the Midterm Project, do this for every app you create.


Click on the View Controller here
to select it.


Open the File inspector.


Uncheck “Use Auto Layout.”

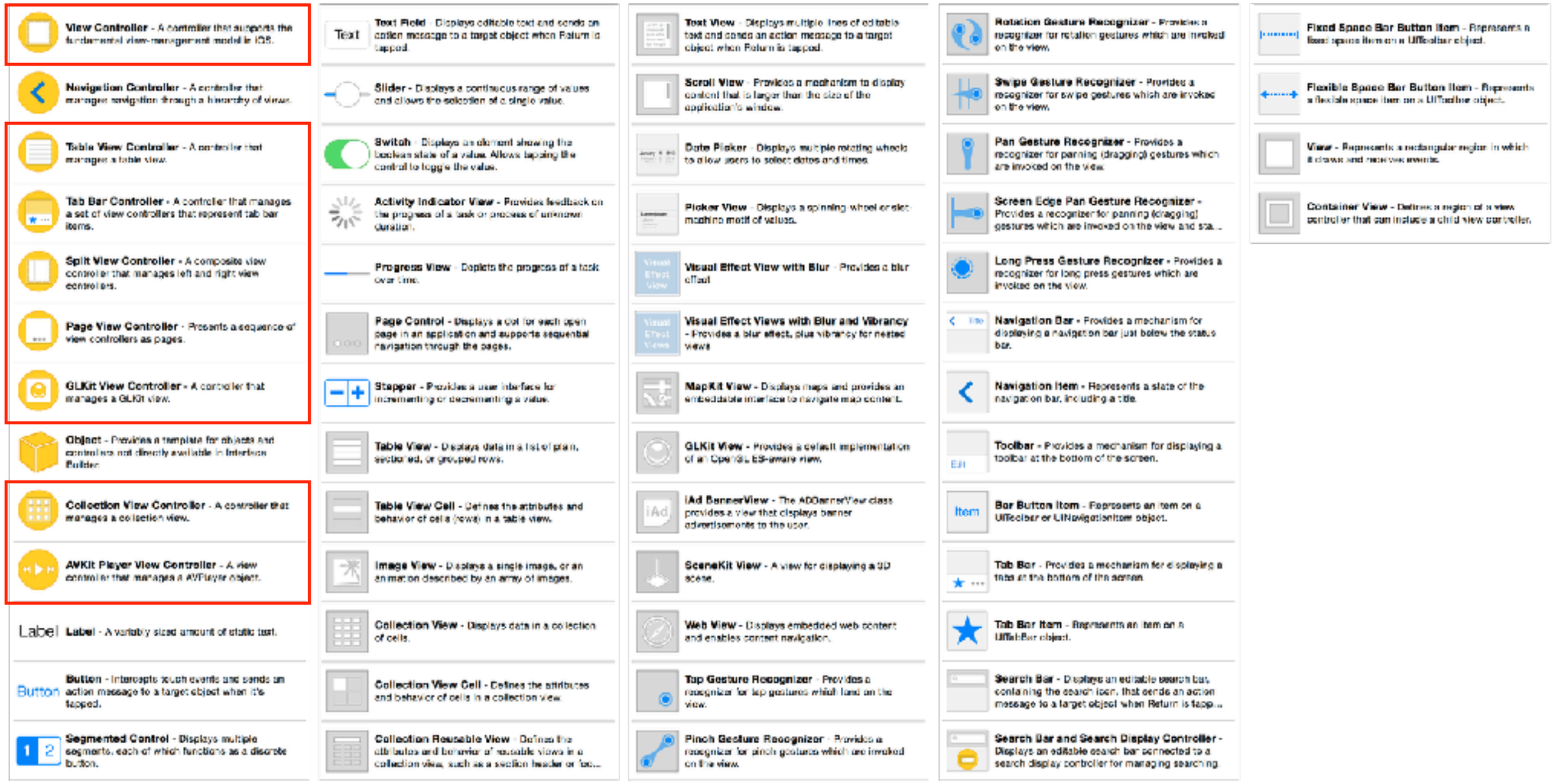

Using Size Classes Requires Auto Layout
Disabling size classes will limit this document to storing data for a single device family. The data for the size class best representing the targeted device will be retained, and all other data will be removed. In addition, segues will be converted to their non-adaptive equivalents.
Keep size class data for:
Phone
Cancel
Disable Size Classes
Click “Disable Size Classes.”
Don’t worry about what this means yet.



Your View Controller should take on a more iPhone-like proportion.

VIEW CONTROLLERS

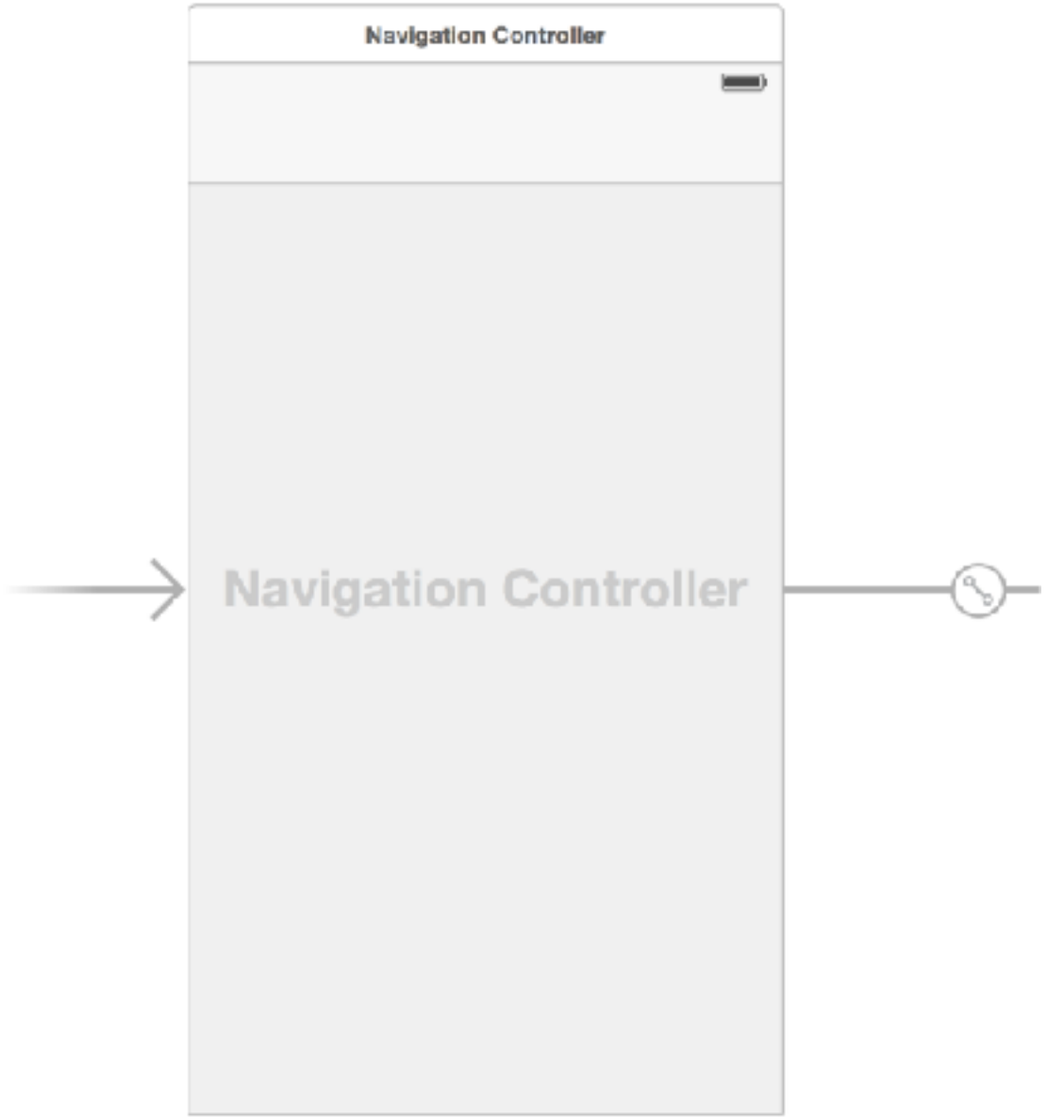
- Each screen of our app is made up of at least one View Controller and at least one View.
- In Storyboards a “Scene” is roughly equivalent to a View Controller.
- A View Controller “controls” a set of Views.


View Controller - A controller that supports the fundamental view-management model in iOS.
Navigation Controller - A controller that manages navigation through a hierarchy of views.
Table View Controller - A controller that manages a table view.
Tab Bar Controller - A controller that manages a set of view controllers that represent tab bar items.
Split View Controller - A composite view controller that manages left and right view controllers.
Page View Controller - Presents a sequence of view controllers as pages.
GLKit View Controller - A controller that manages a GLKit view.
Object - Provides a template for objects and controllers not directly available in Interface Builder.
Collection View Controller - A controller that manages a collection view.
AVKit Player View Controller - A view controller that manages a AVPlayer object.
Label Label - A variably sized amount of static text.
Button Button - Intercepts touch events and sends an action message to a target object when it's tapped.
1 2 Segmented Control - Displays multiple segments, each of which functions as a discrete button.
Text Text Field - Displays editable text and sends an action message to a target object when Return is tapped.
Slider - Displays a continuous range of values and allows the selection of a single value.
Switch - Displays an element showing the boolean state of a value. Allows tapping the control to toggle the value.
Activity Indicator View - Provides feedback on the progress of a task or process of unknown duration.
Progress View - Depicts the progress of a task over time.
Page Control - Displays a dot for each open page in an application and supports sequential navigation through the pages.
Stepper - Provides a user interface for incrementing or decrementing a value.
Table View - Displays data in a list of plain, sectioned, or grouped rows.
Table View Cell - Defines the attributes and behavior of cells (rows) in a table view.
Image View - Displays a single image, or an animation described by an array of images.
Collection View - Displays data in a collection of cells.
Collection View Cell - Defines the attributes and behavior of cells in a collection view.
Collection Reusable View - Defines the attributes and behavior of reusable views in a collection view, such as a section header or foo...
Text View - Displays multiple lines of editable text and sends an action message to a target object when Return is tapped.
Scroll View - Provides a mechanism to display content that is larger than the size of the application's window.
Date Picker - Displays multiple rotating wheels to allow users to select dates and times.
Picker View - Displays a spinning wheel or slot machine motif of values.
Visual Effect View with Blur - Provides a blur effect
Visual Effect Views with Blur and Vibrancy - Provides a blur effect, plus vibrancy for nested views
MapKit View - Displays maps and provides an embeddable interface to navigate map content.
GLKit View - Provides a default implementation of an OpenGL ES-aware view.
iAd BannerView - The ADBannerView class provides a view that displays banner advertisements to the user.
SceneKit View - A view for displaying a 3D scene.
Web View - Displays embedded web content and enables content navigation.
Tap Gesture Recognizer - Provides a recognizer for tap gestures which can be on the view.
Pinch Gesture Recognizer - Provides a recognizer for pinch gestures which are invoked on the view.
Rotation Gesture Recognizer - Provides a recognizer for rotation gestures which are invoked on the view.
Swipe Gesture Recognizer - Provides a recognizer for swipe gestures which are invoked on the view.
Pan Gesture Recognizer - Provides a recognizer for panning (dragging) gestures which are invoked on the view.
Screen Edge Pan Gesture Recognizer - Provides a recognizer for panning (dragging) gestures which are invoked on the view and sta...
Long Press Gesture Recognizer - Provides a recognizer for long press gestures which are invoked on the view.
Navigation Bar - Provides a mechanism for displaying a navigation bar just below the status bar.
Navigation Item - Represents a state of the navigation bar, including a title.
Edit Toolbar - Provides a mechanism for displaying a toolbar at the bottom of the screen.
Item Bar Button Item - Represents an item on a UIToolbar or UINavigationItem object.
Tab Bar - Provides a mechanism for displaying a tabs at the bottom of the screen.
Tab Bar Item - Represents an item on a UITabBar object.
Search Bar - Displays an editable search bar, containing the search icon, that sends an action message to a target object when Return is tapp...
Search Bar and Search Display Controller - Displays an editable search bar connected to a search display controller for managing searching.
Fixed Space Bar Button Item - Represents a fixed space item on a UIToolbar object.
Flexible Space Bar Button Item - Represents a flexible space item on a UIToolbar object.
View - Represents a rectangular region in which it draws and receives events.
Container View - Defines a region of a view controller that can include a child view controller.

NAVIGATION CONTROLLERS

- Navigation Controllers manage several View Controllers.
- One View Controller is “presented” to the user at a time.
- They organize the View Controllers in a browser-history-like fashion (think forward and back buttons).
- By default, they place Navigation Bars on all the View Controllers they manage.

NAVIGATION CONTROLLERS



View Controller - A controller that supports the fundamental view-management model in iOS.

Navigation Controller - A controller that manages navigation through a hierarchy of views.

Table View Controller - A controller that manages a table view.

Tab Bar Controller - A controller that manages a set of view controllers that represent tab bar items.

Split View Controller - A composite view controller that manages left and right view controllers.

Page View Controller - Presents a sequence of view controllers as pages.

GLKit View Controller - A controller that manages a GLKit view.

Object - Provides a template for objects and controllers not directly available in Interface Builder.

Collection View Controller - A controller that manages a collection view.

AVKit Player View Controller - A view controller that manages a AVPlayer object.

Label **Label** - A variably sized amount of static text.

Button **Button** - Intercepts touch events and sends an action message to a target object when it's tapped.

1 2 **Segmented Control** - Displays multiple segments, each of which functions as a discrete button.

Text **Text Field** - Displays editable text and sends an action message to a target object when Return is tapped.

Slider - Displays a continuous range of values and allows the selection of a single value.

Switch - Displays an element showing the boolean state of a value. Allows tapping the control to toggle the value.

Activity Indicator View - Provides feedback on the progress of a task or process of unknown duration.

Progress View - Depicts the progress of a task over time.

Page Control - Displays a dot for each open page in an application and supports sequential navigation through the pages.

Stepper - Provides a user interface for incrementing or decrementing a value.

Table View - Displays data in a list of plain, sectioned, or grouped rows.

Table View Cell - Defines the attributes and behavior of cells (rows) in a table view.

Image View - Displays a single image, or an animation described by an array of images.

Collection View - Displays data in a collection of cells.

Collection View Cell - Defines the attributes and behavior of cells in a collection view.

Collection Reusable View - Defines the attributes and behavior of reusable views in a collection view, such as a section header or foo...

Text View - Displays multiple lines of editable text and sends an action message to a target object when Return is tapped.

Scroll View - Provides a mechanism to display content that is larger than the size of the application's window.

Date Picker - Displays multiple rotating wheels to allow users to select dates and times.

Picker View - Displays a spinning wheel or slot machine motif of values.

Visual Effect View **Visual Effect View with Blur** - Provides a blur effect

Visual Effect Views **Visual Effect Views with Blur and Vibrancy** - Provides a blur effect, plus vibrancy for nested views

MapKit View - Displays maps and provides an embeddable interface to navigate map content.

GLKit View - Provides a default implementation of an OpenGL ES-aware view.

iAd **iAd BannerView** - The ADBannerView class provides a view that displays banner advertisements to the user.

SceneKit View - A view for displaying a 3D scene.

Web View - Displays embedded web content and enables content navigation.

Tap Gesture Recognizer - Provides a recognizer for tap gestures which can be on the view.

Pinch Gesture Recognizer - Provides a recognizer for pinch gestures which are invoked on the view.

Rotation Gesture Recognizer - Provides a recognizer for rotation gestures which are invoked on the view.

Swipe Gesture Recognizer - Provides a recognizer for swipe gestures which are invoked on the view.

Pan Gesture Recognizer - Provides a recognizer for panning (dragging) gestures which are invoked on the view.

Screen Edge Pan Gesture Recognizer - Provides a recognizer for panning (dragging) gestures which are invoked on the view and sta...

Long Press Gesture Recognizer - Provides a recognizer for long press gestures which are invoked on the view.

Navigation Bar - Provides a mechanism for displaying a navigation bar just below the status bar.

Navigation Item - Represents a state of the navigation bar, including a title.

Edit **Toolbar** - Provides a mechanism for displaying a toolbar at the bottom of the screen.

Item **Bar Button Item** - Represents an item on a UIToolbar or UINavigationItem object.

Tab Bar - Provides a mechanism for displaying a tabs at the bottom of the screen.

Tab Bar Item - Represents an item on a UITabBar object.

Search Bar - Displays an editable search bar, containing the search icon, that sends an action message to a target object when Return is tapp...

Search Bar and Search Display Controller - Displays an editable search bar connected to a search display controller for managing searching.

Fixed Space Bar Button Item - Represents a fixed space item on a UIToolbar object.

Flexible Space Bar Button Item - Represents a flexible space item on a UIToolbar object.

View - Represents a rectangular region in which it draws and receives events.

Container View - Defines a region of a view controller that can include a child view controller.

SEGUES

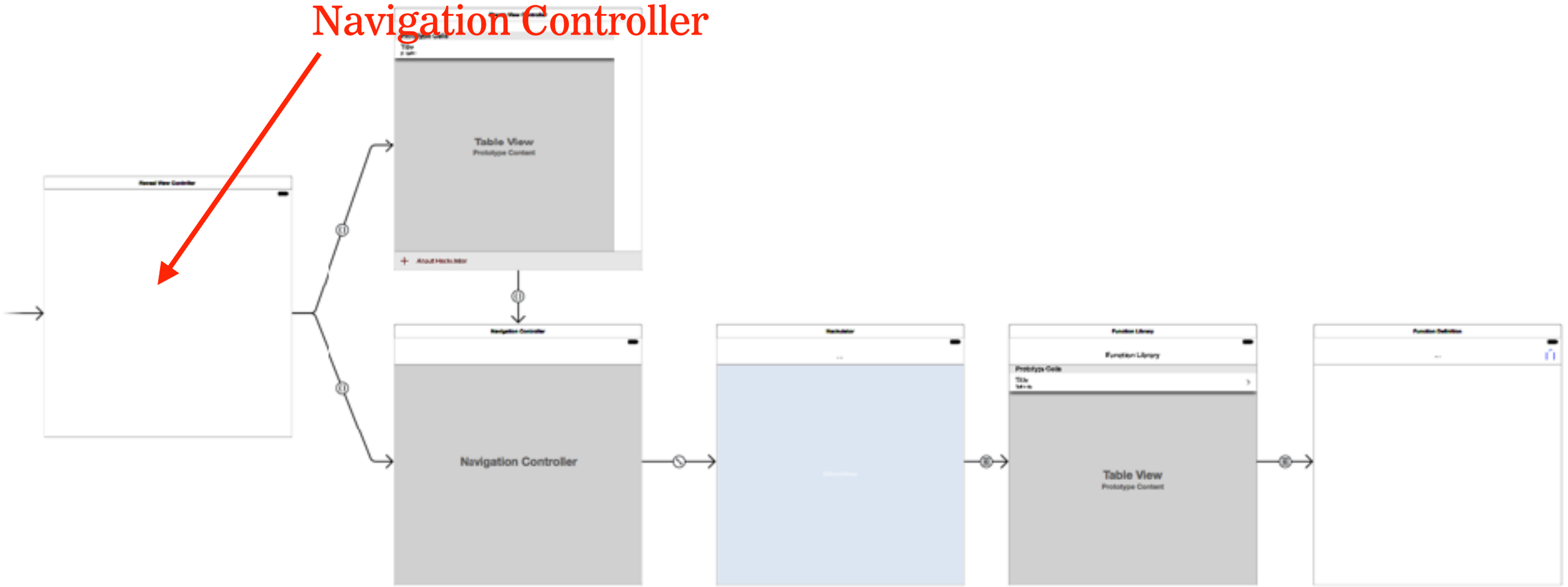
- Navigation Controllers enable us to link multiple Scenes (View Controllers) together with a very typical mobile navigation paradigm.
- They represent transitions between many Scenes. These transitions are encoded by “Segues”.

NAVIGATION CONTROLLERS

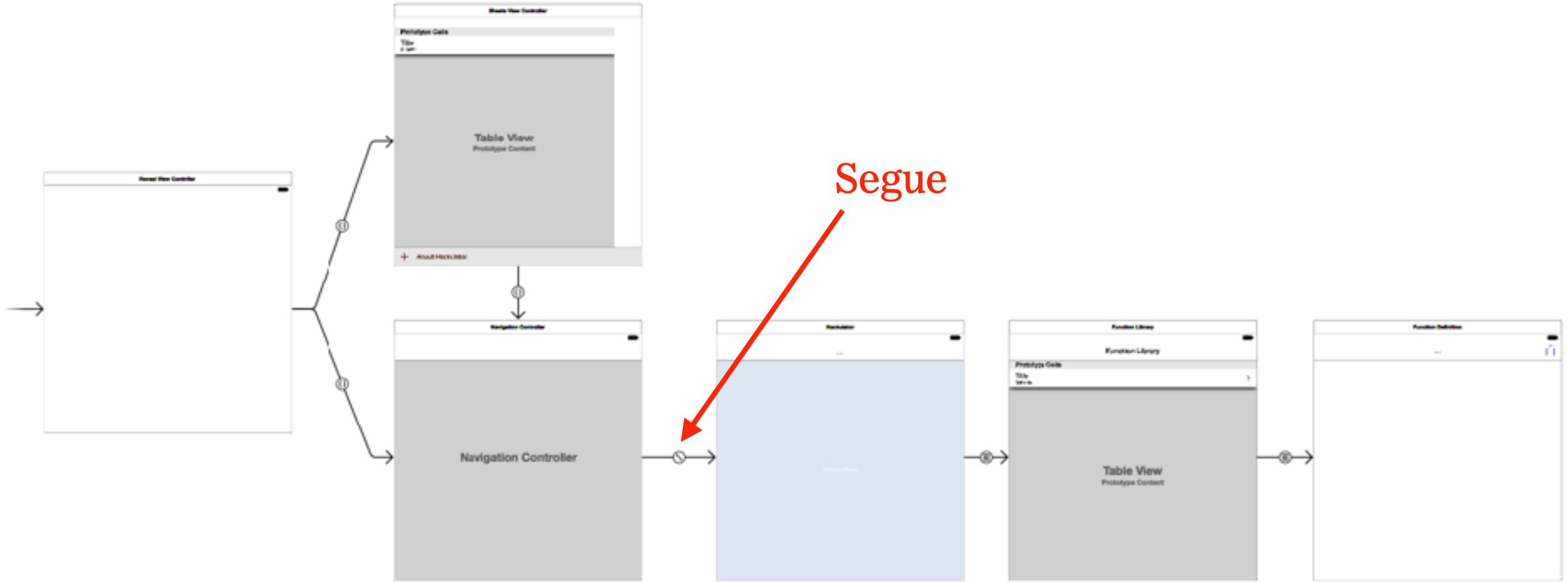
Some terms:

- “Initial View Controller” – The View Controller the app will load when an app is launched.
- “Root View Controller” – The View Controller that a Navigation Controller will first load when it is created.

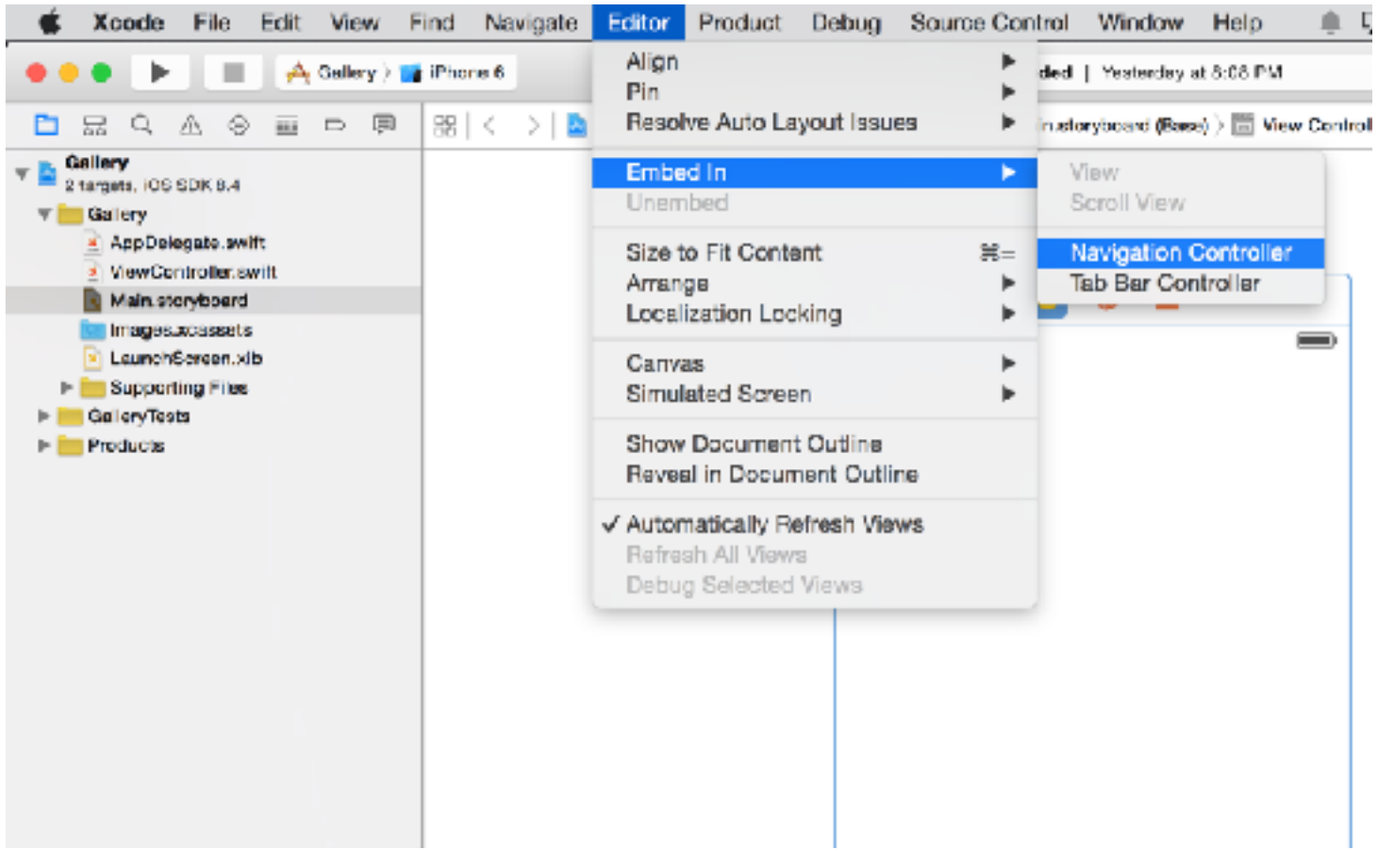
SIMPLE STORYBOARD EXAMPLE



SIMPLE STORYBOARD EXAMPLE



NAVIGATION CONTROLLERS



SEGUES

- A “Segue” is essentially a transition between two View Controllers.
- In code, we can use them to transfer information from one View Controller to another, like going from a list of emails to a single view of just one email.
- There are several types…

Name	Interface Builder Symbol	Description
Show		Present the content in the detail or master area depending on the content of the screen. If the app is displaying a master and detail view, the content is pushed onto the detail area. If the app is only displaying the master or the detail, the content is pushed on top of the current view controller stack.
Show Detail		Present the content in the detail area. If the app is displaying a master and detail view, the new content replaces the current detail. If the app is only displaying the master or the detail, the content replaces the top of the current view controller stack.
Present Modally		Present the content modally. There are options to choose a presentation style (`UIModalPresentationStyle`) and a transition style (`UIModalTransitionStyle`).
Present as Popover		Present the content as a popover anchored to an existing view. There is an option to specify the possible directions of the arrow shown on one edge of the popover view (`UIPopoverArrowDirection`). There is also an option to specify the anchor view.
Custom		A custom segue enabling you to write your own behaviors.
Push (Deprecated)		Present the content by pushing it onto the current stack of view controllers.
Modal (Deprecated)		Present the content modally on top of the existing screen. The options are the same as Present Modally.
Popover (Deprecated)		Present the content as a popover. The options are the same as Present as Popover.

USER FLOW

- Storyboards can reflect the structure of an app's user interface.
- Consider different ways of structuring a notes app.

PRACTICE MAKING SEGUES

CONCLUSION

REVIEW & RECAP

- In this workshop, we've covered the following topics:
  - What is iOS development? What does the ecosystem look like?
  - What are some common skills and tools used in iOS development?
  - How to navigate and use Xcode
  - I have an app idea; where do I begin?

LEARNING PLAN

Evaluate your skills! How confident are you with:

- Programming fundamentals
- User Interface design & UX principles
- Mobile networks
- Prototyping and testing

WHAT SHOULD YOU DO NEXT?

Want to be an iOS developer? Try working on these:

- Explore basic programming foundations
- Get familiar with Swift Syntax
- Practice app development on your own projects!!

WANT MORE?

General Assembly offers courses in iOS Development!

Check out our:

- iOS Development Immersive Program

ADDITIONAL RESOURCES

BOOKS

- Functional Swift by obj.io
- Big Nerd Ranch Guide, 4th Ed
- Swift for Beginners
- iOS Fundamentals
- Stack Overflow :)
- raywenderlich.com

ONLINE

- Ray Wenderlich's online tutorials (link)
- Apple's Developer Docs (obviously) (link)
- Curated List of "awesome" iOS Resources (link)
- App coda for beginners (link)
- Weekly iOS "goodies" online newsletter (link)
- Learn iOS thread on Quora (link)

Q&A

EXIT TICKETS

DON’T FORGET TO FILL OUT YOUR EXIT TICKET