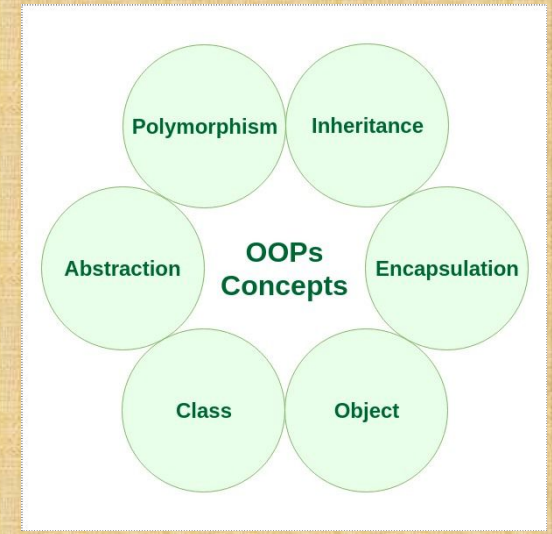
iOS Programming

Lecture 7

Recap

Object Oriented Programming

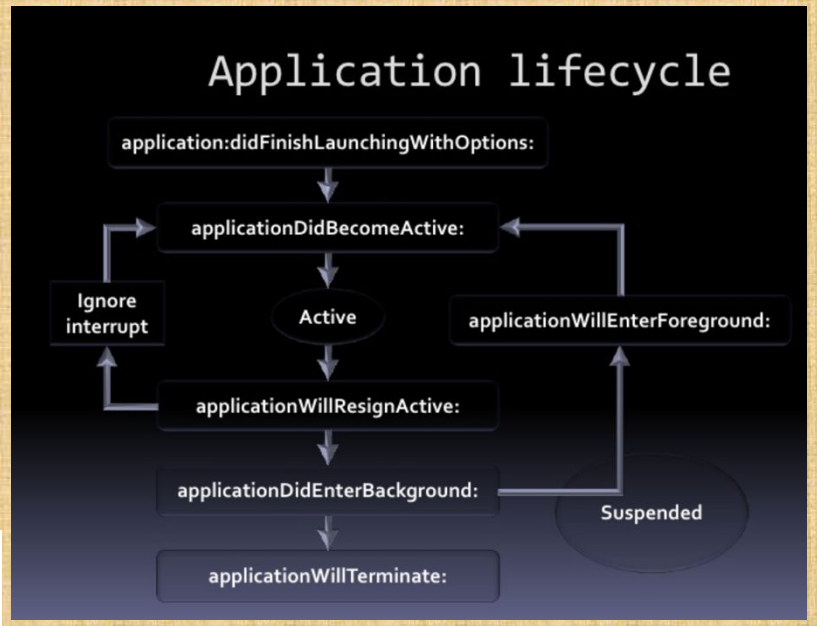


Classes

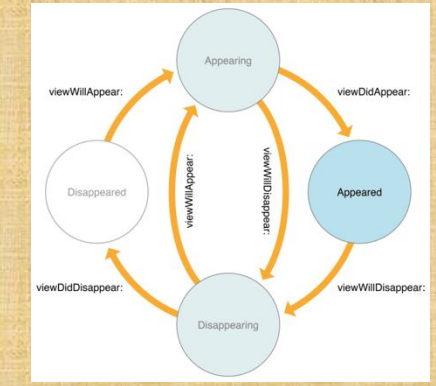
Protocols

Today

XCode Basics

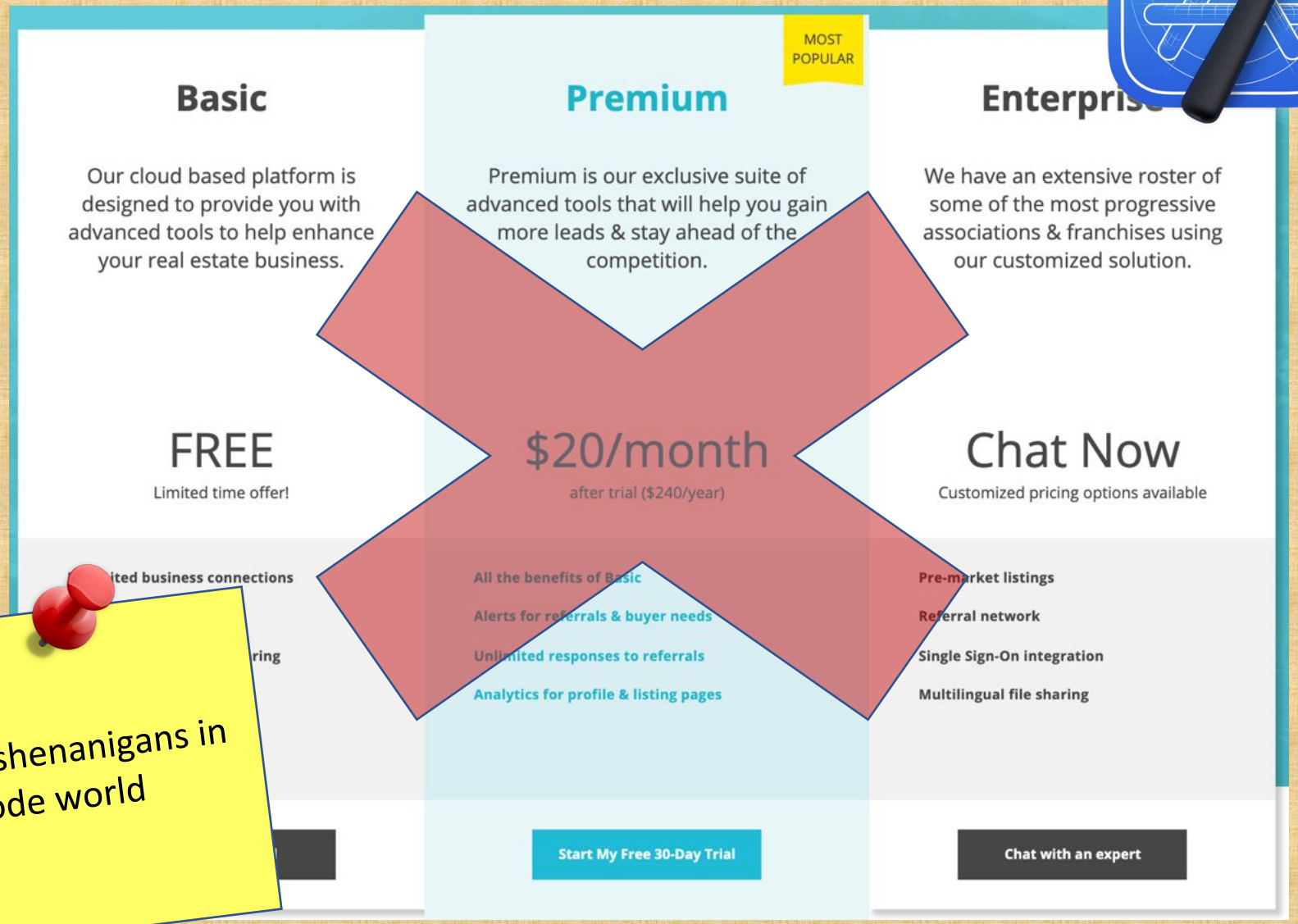


Application Life Cycle



Controller Life Cycle

XCode Basics



No such shenanigans in Xcode world

XCode Basics

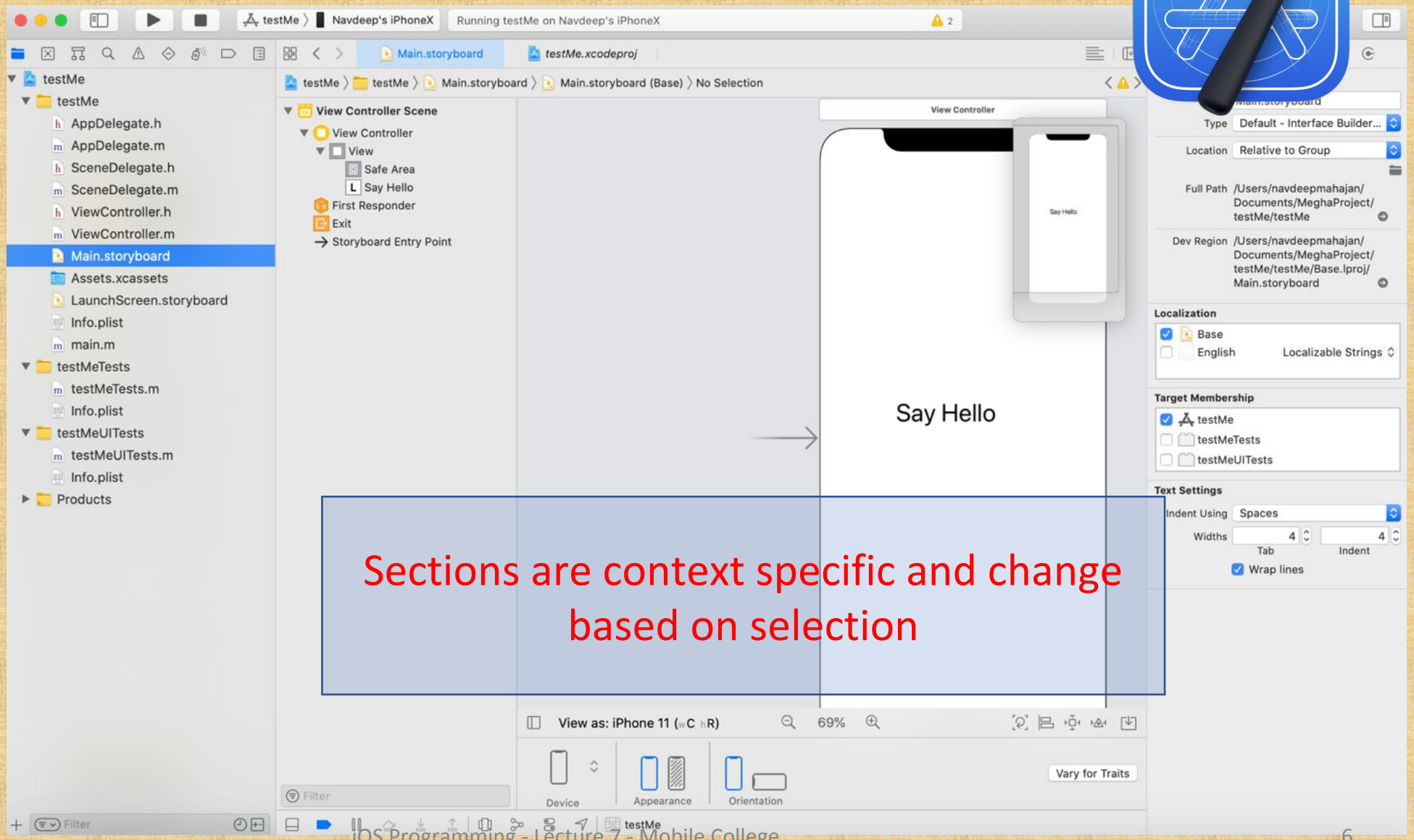
Simulator & Devices

Navigator

Editor

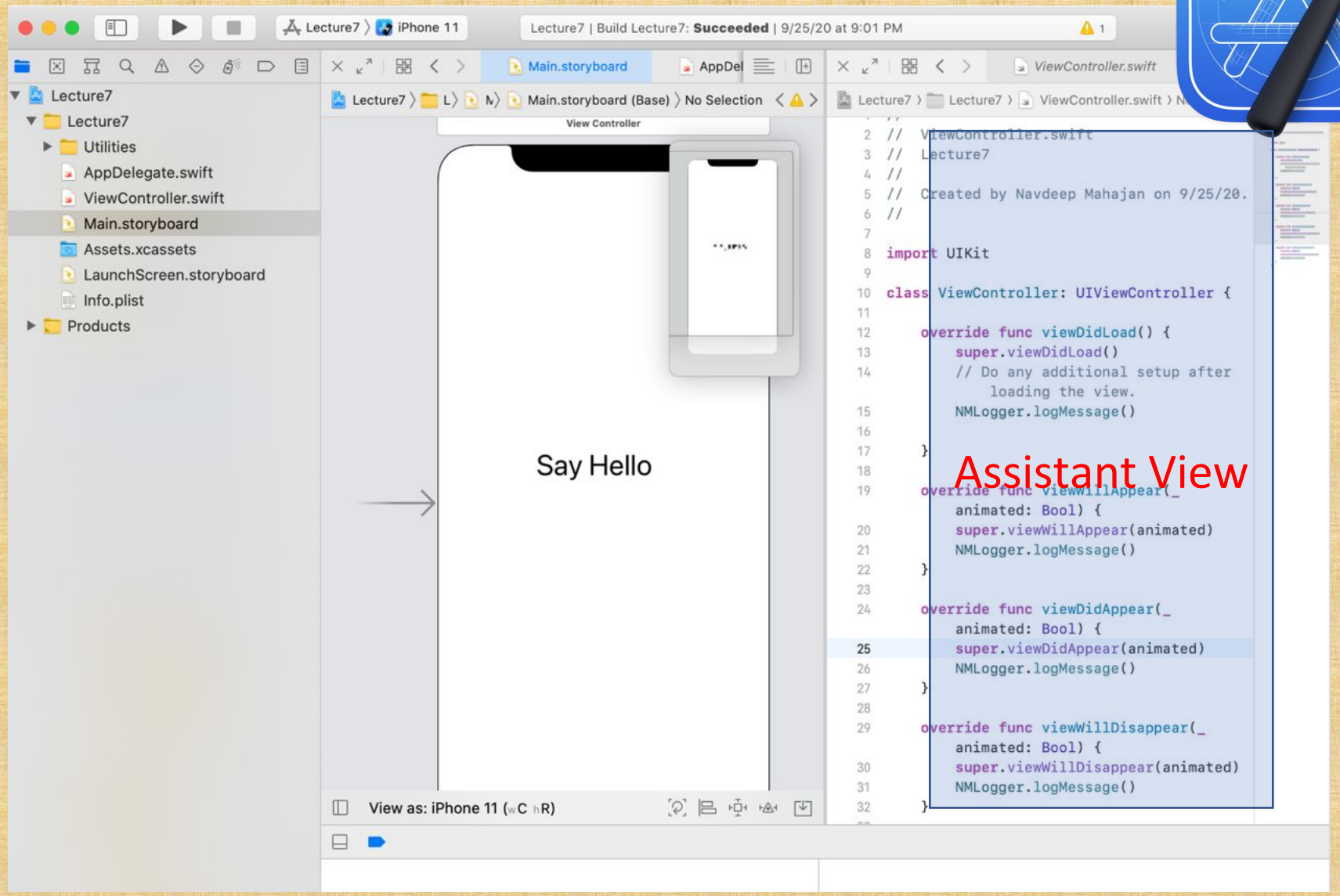
Inspectors

XCode Basics

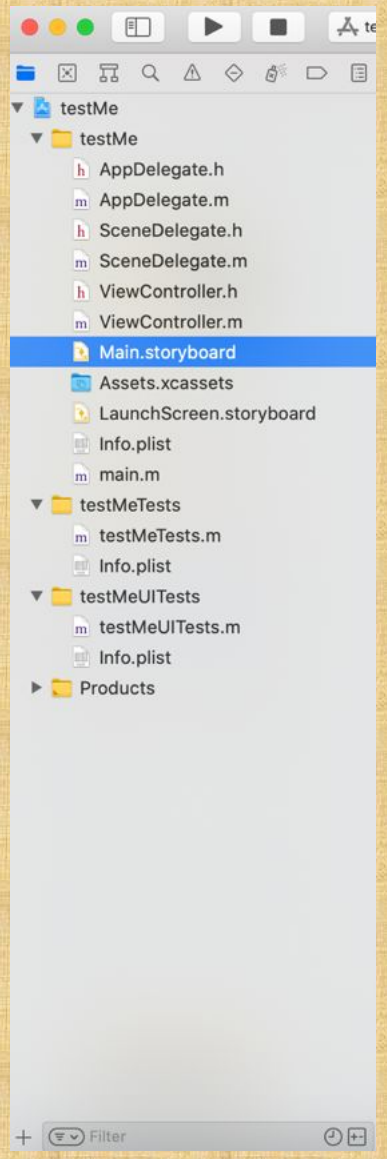


Sections are context specific and change based on selection

XCode Basics



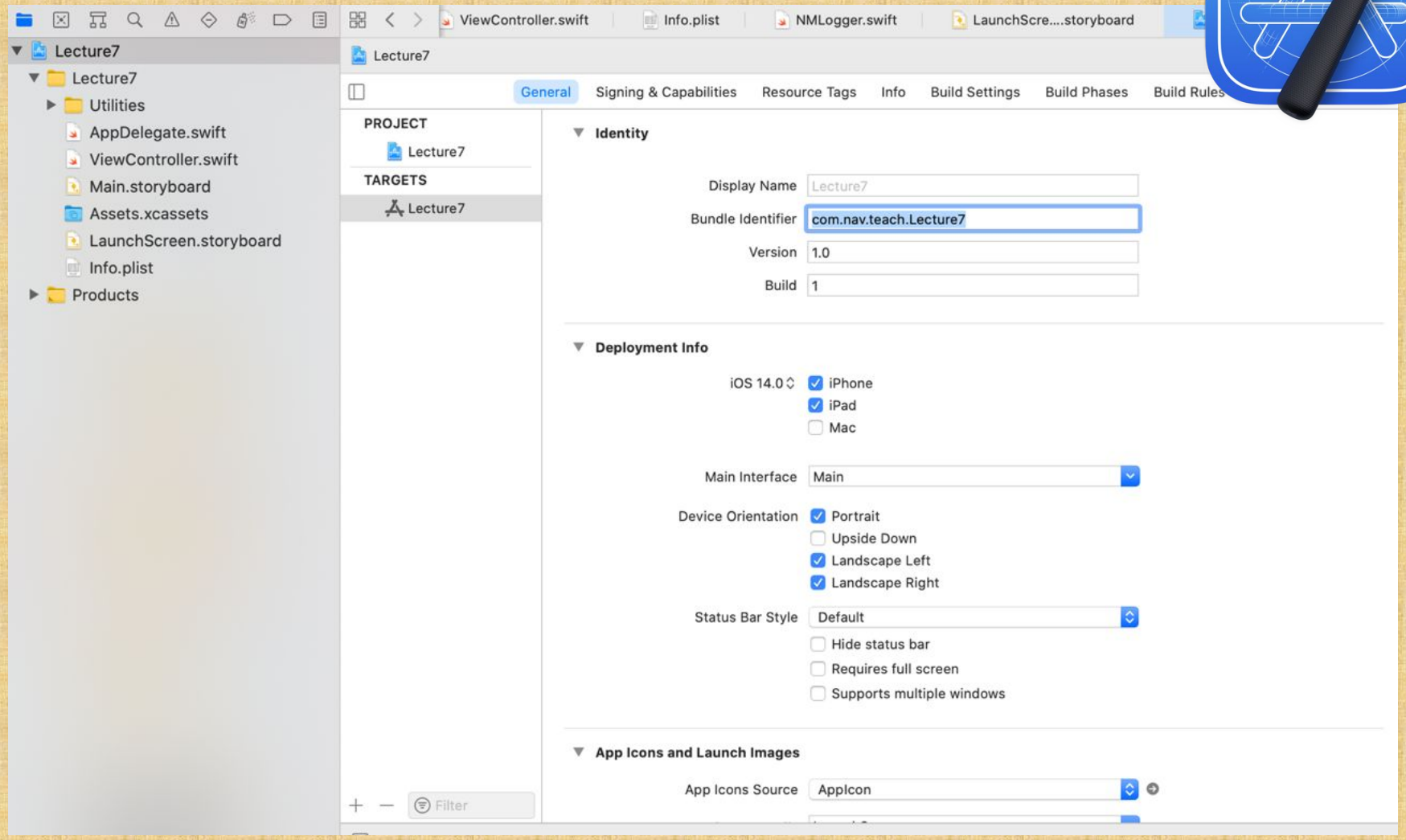
XCode – File Structure



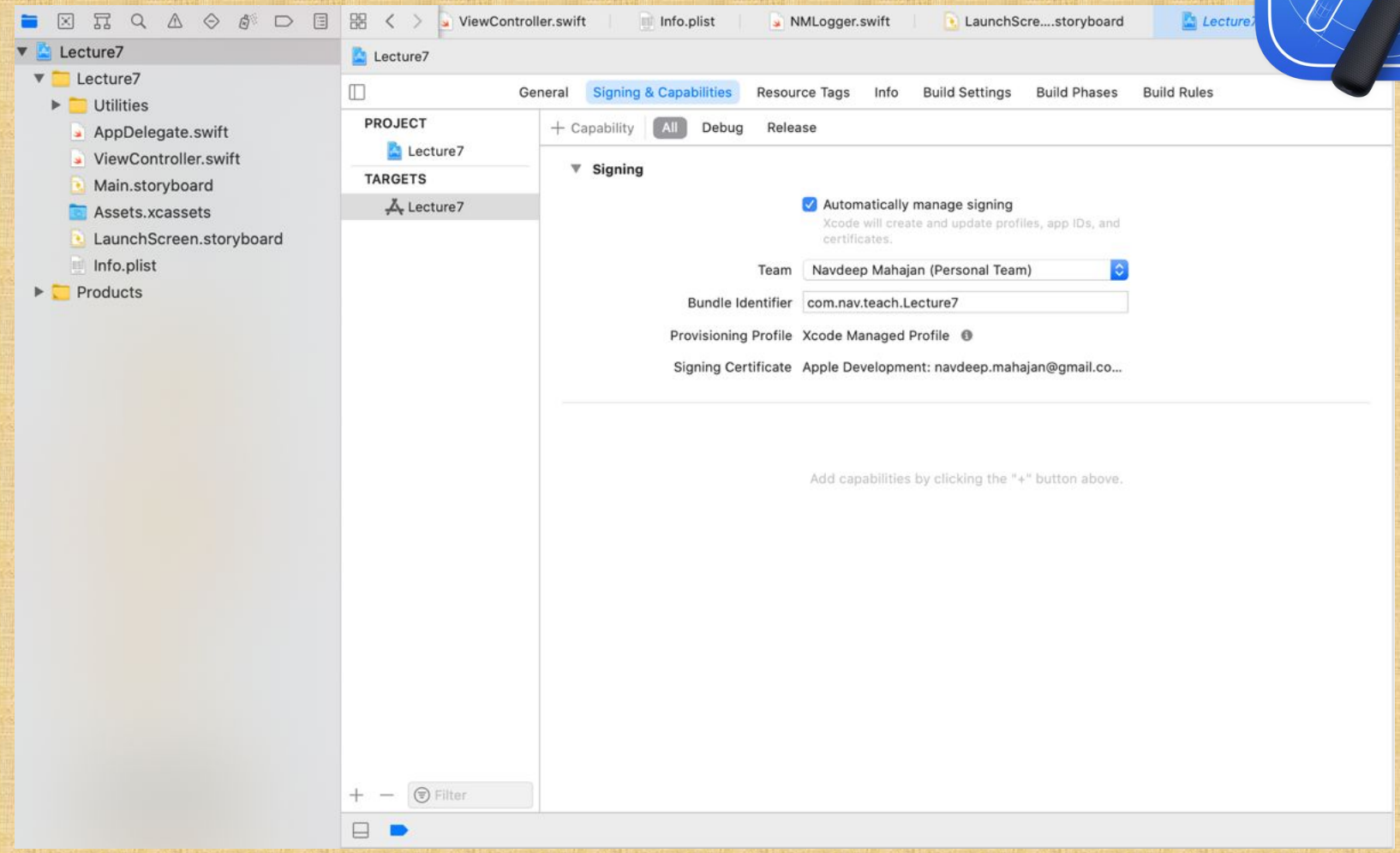
Let's start by inspecting the files:

- App Delegate
- Story Boards
- View Controllers
- Info.plist
- Test groups
- Products Dir

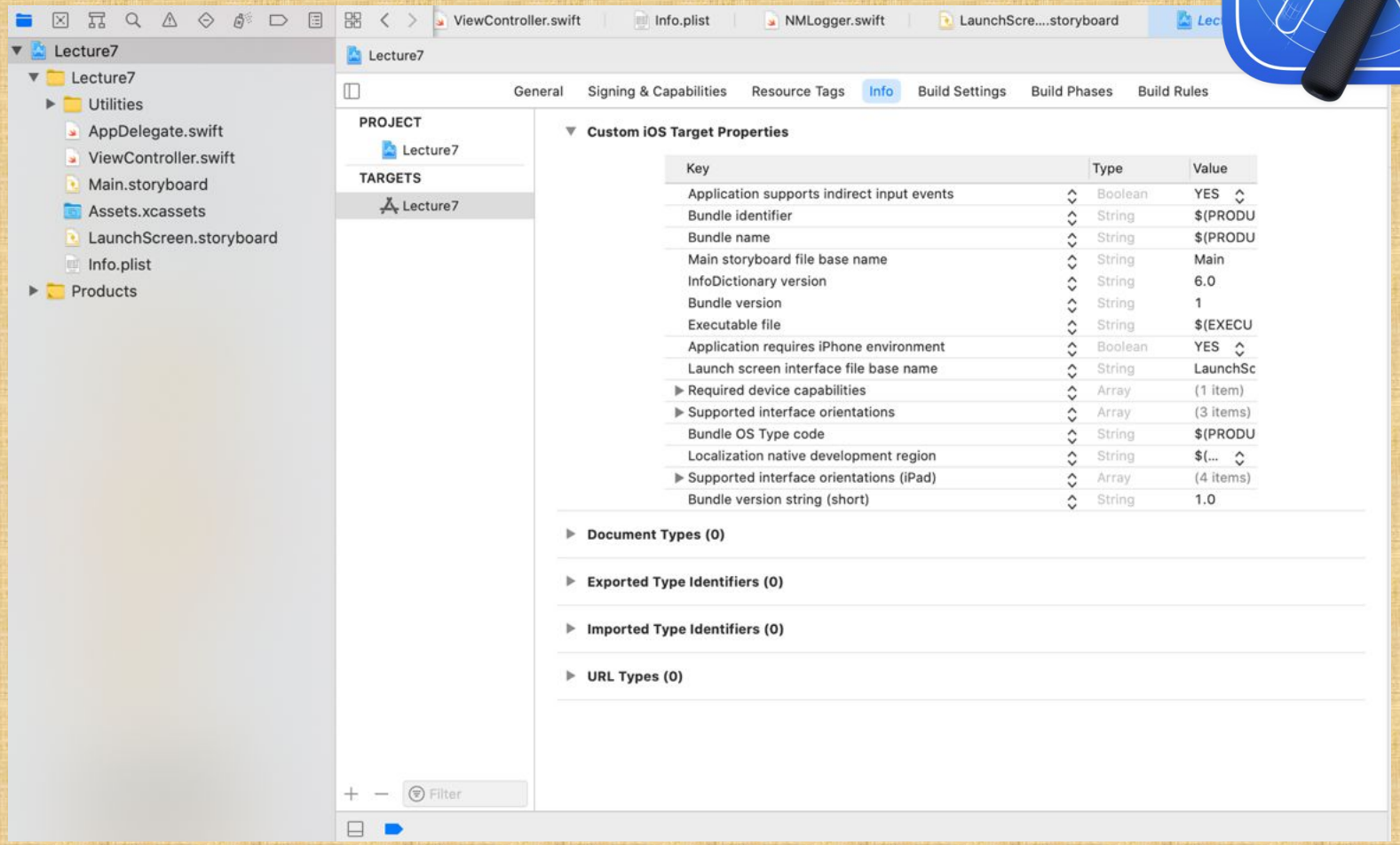
XCode – General



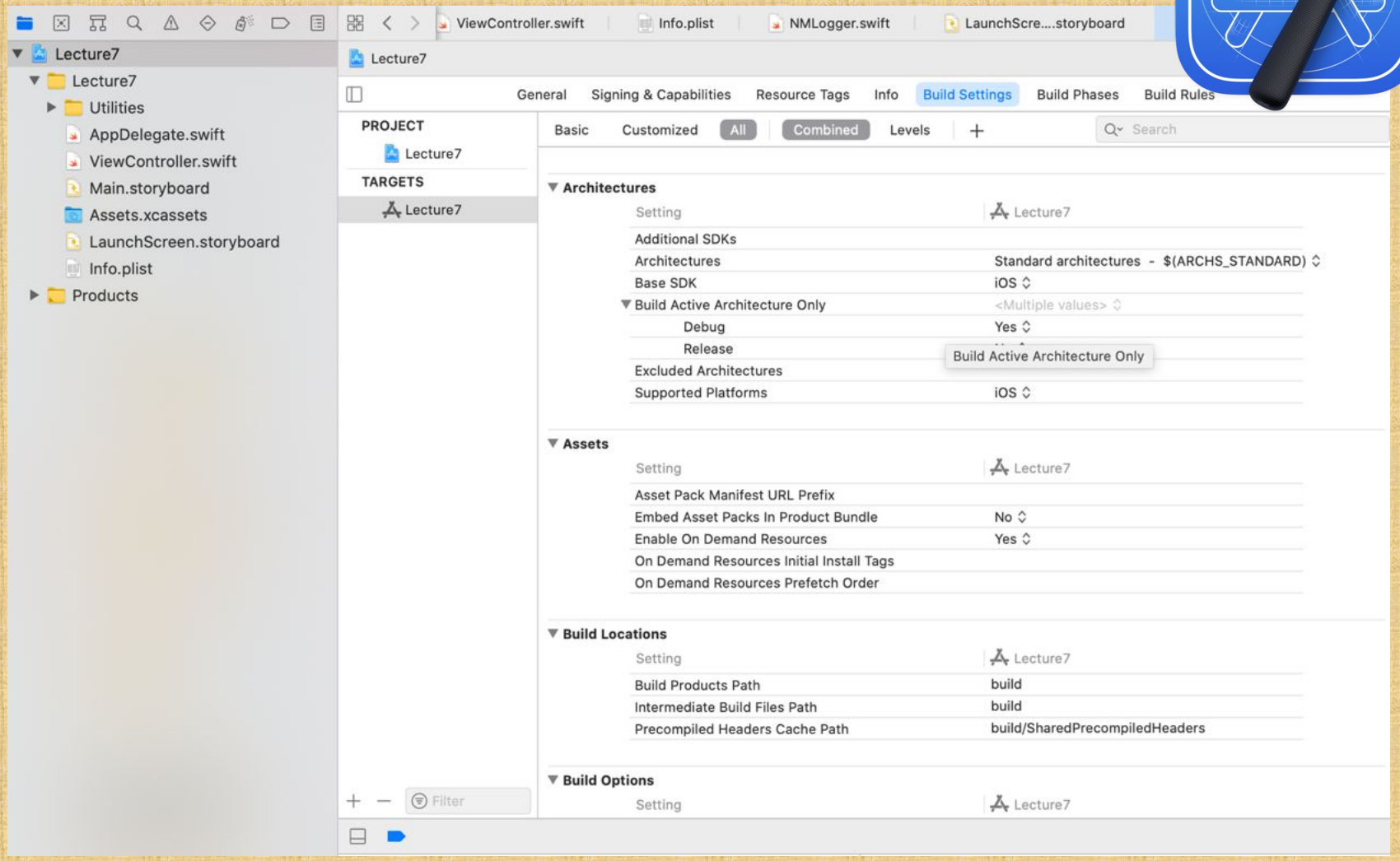
XCode – Signing & Capabilities



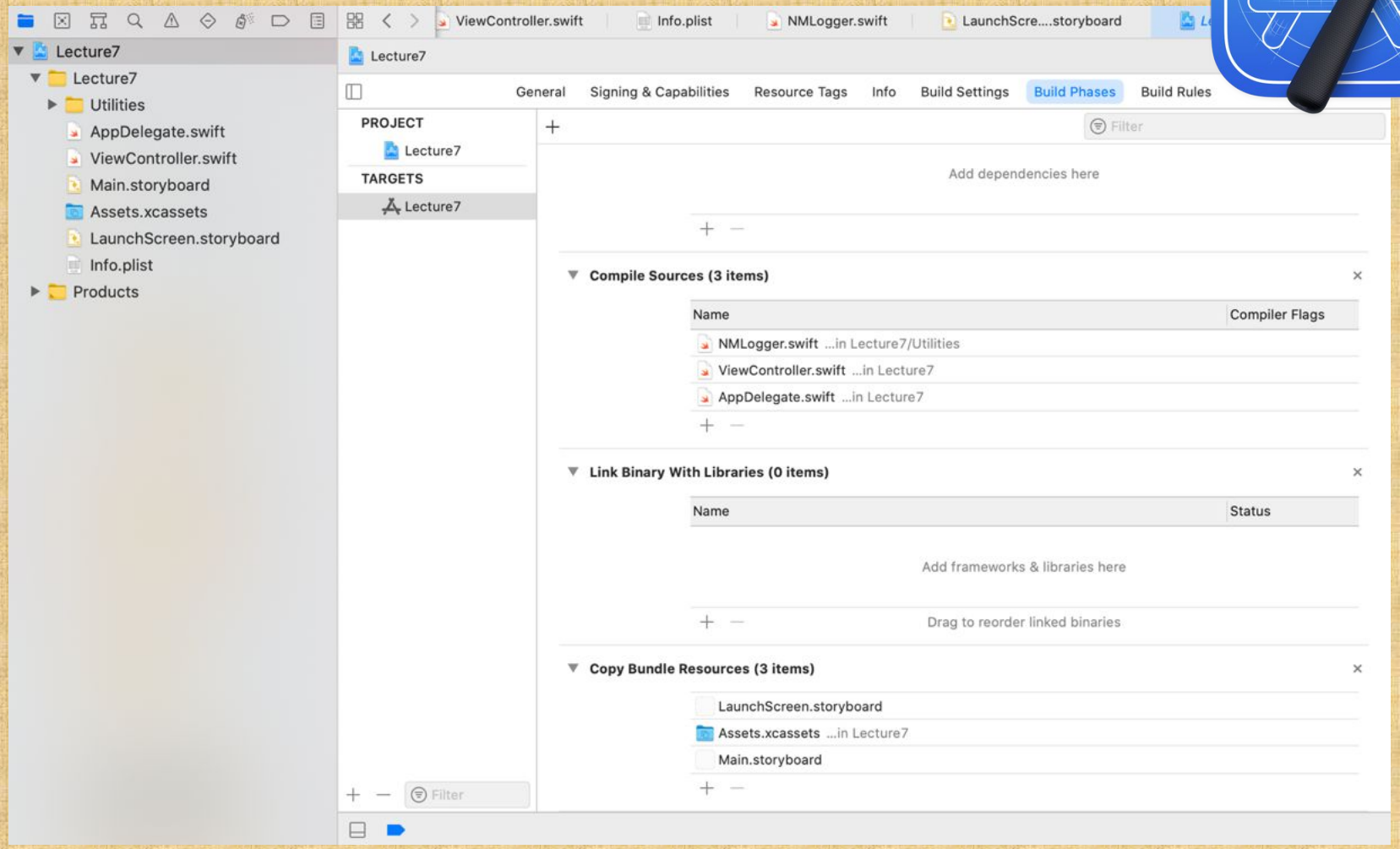
XCode – Info


ViewController.swift
Info.plist
NMLogger.swift
LaunchScre....storyboard
Lecture7
Lecture7
Utilities
AppDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Products
Lecture7
General
Signing & Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
PROJECT
Lecture7
TARGETS
Lecture7
Custom iOS Target Properties
Key
Type
Value
Application supports indirect input events
Boolean
YES
Bundle identifier
String
$(PRODU
Bundle name
String
$(PRODU
Main storyboard file base name
String
Main
InfoDictionary version
String
6.0
Bundle version
String
1
Executable file
String
$(EXECU
Application requires iPhone environment
Boolean
YES
Launch screen interface file base name
String
LaunchSc
Required device capabilities
Array
(1 item)
Supported interface orientations
Array
(3 items)
Bundle OS Type code
String
$(PRODU
Localization native development region
String
$(...
Supported interface orientations (iPad)
Array
(4 items)
Bundle version string (short)
String
1.0
Document Types (0)
Exported Type Identifiers (0)
Imported Type Identifiers (0)
URL Types (0)
Filter

XCode – Build Settings



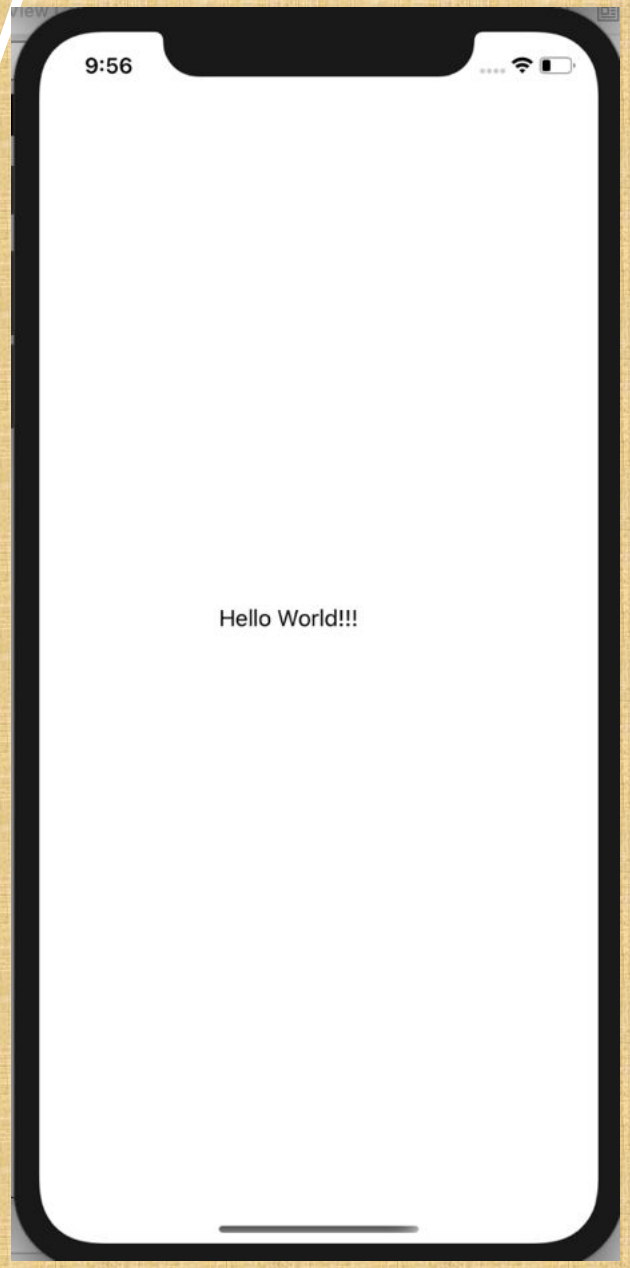
XCode – Build Phases



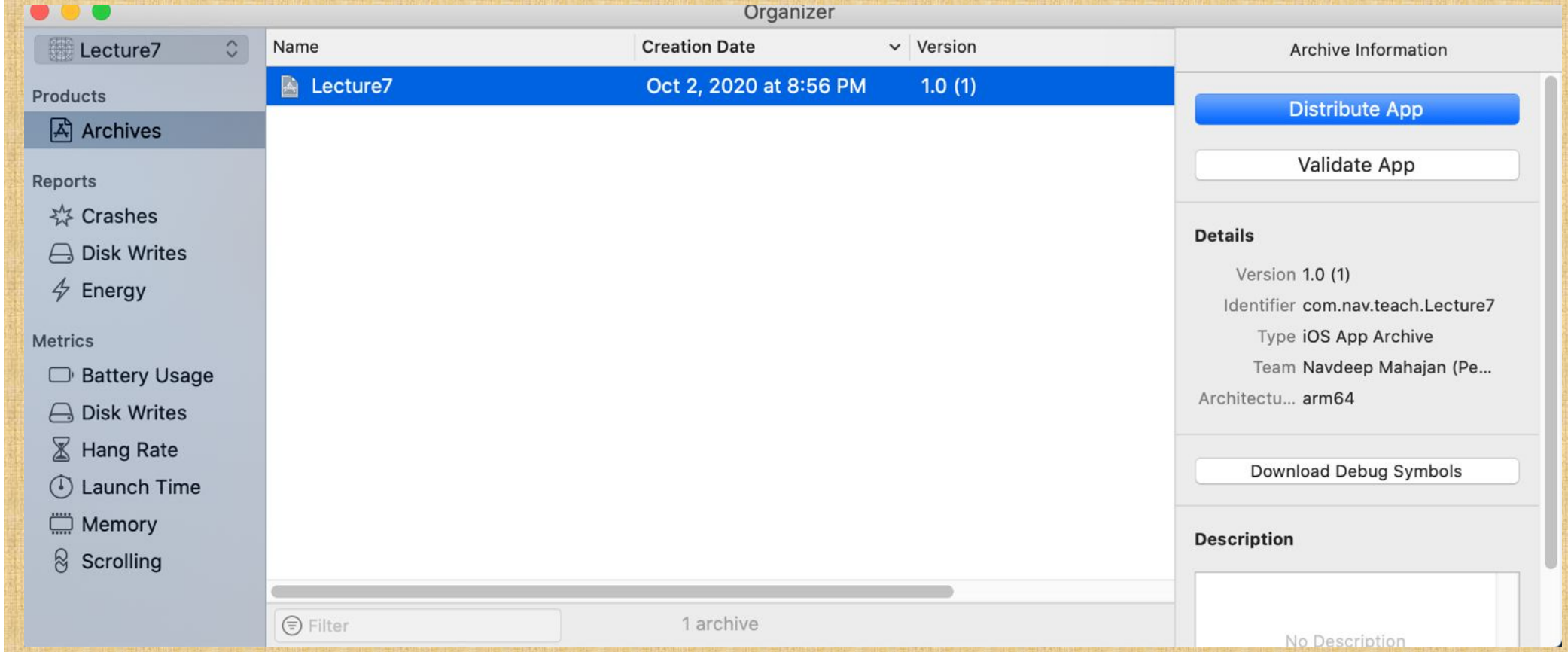
Sample App – Let’s Deploy

We ran the Sample App on Simulator in the past.

Now, Let’s see what it takes to deploy on the device.



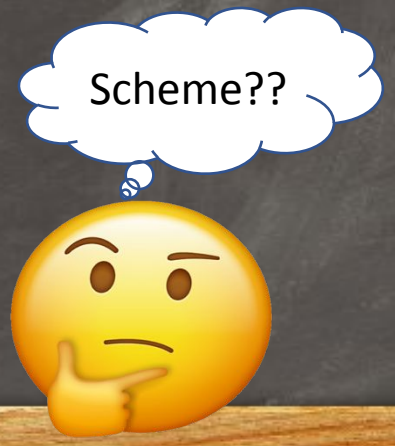
Sample App – Distribute



Sample App – Create IPA

Step 1: Create the Archive:

```
xcodebuild -project Lecture7.xcodeproj -scheme Lecture7
-archivePath
"/Users/navdeepmahajan/Documents/Projects/Teaching/iOS
Programming/Code/Lecture7/Lecture7/Lecture7/build/Lecture7.x
carchive" archive
```



Sample App – Create IPA

Step 2: List the options for exporting the archive

ExportOptions.plist

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
        <key>compileBitcode</key>
        <true/>
        <key>method</key>
        <string>development</string>
        <key>signingStyle</key>
        <string>automatic</string>
        <key>stripSwiftSymbols</key>
        <true/>
        <key>teamID</key>
        <string>8789SU3L3V</string>
        <key>thinning</key>
        <string>&lt;none&gt;</string>
</dict>
</plist>
```

Sample App – Export IPA

Step 3: Export IPA

```
xcodebuild -exportArchive -archivePath
"/Users/navdeepmahajan/Documents/Projects/Teaching/iOS
Programming/Code/Lecture7/Lecture7/Lecture7/build/Lecture7.x
carchive" -exportPath
"/Users/navdeepmahajan/Documents/Projects/Teaching/iOS
Programming/Code/Lecture7/Lecture7/Lecture7/build/"
-exportOptionsPlist
"/Users/navdeepmahajan/Documents/Projects/Teaching/iOS
Programming/Code/Lecture7/Lecture7/Lecture7/build/Lecture7.p
list"
```

Sample App – Let’s Dissect IPA

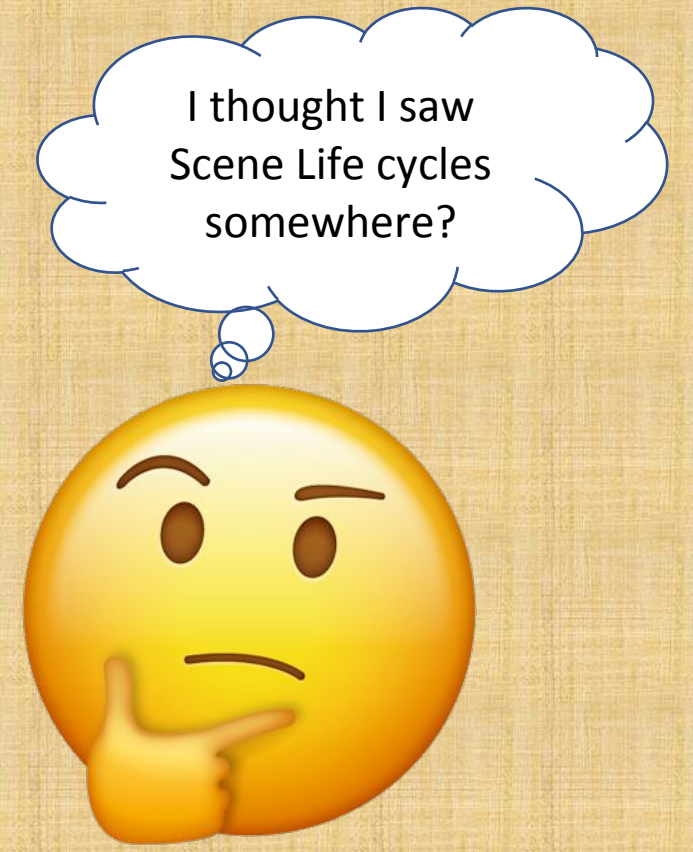
Life cycle

Definition of life cycle
1: the series of stages in form and functional activity through which an organism passes between successive recurrences of a specified primary stage

2: LIFE HISTORY sense 2

3: a series of stages through which something (such as an individual, culture, or manufactured product) passes during its lifetime

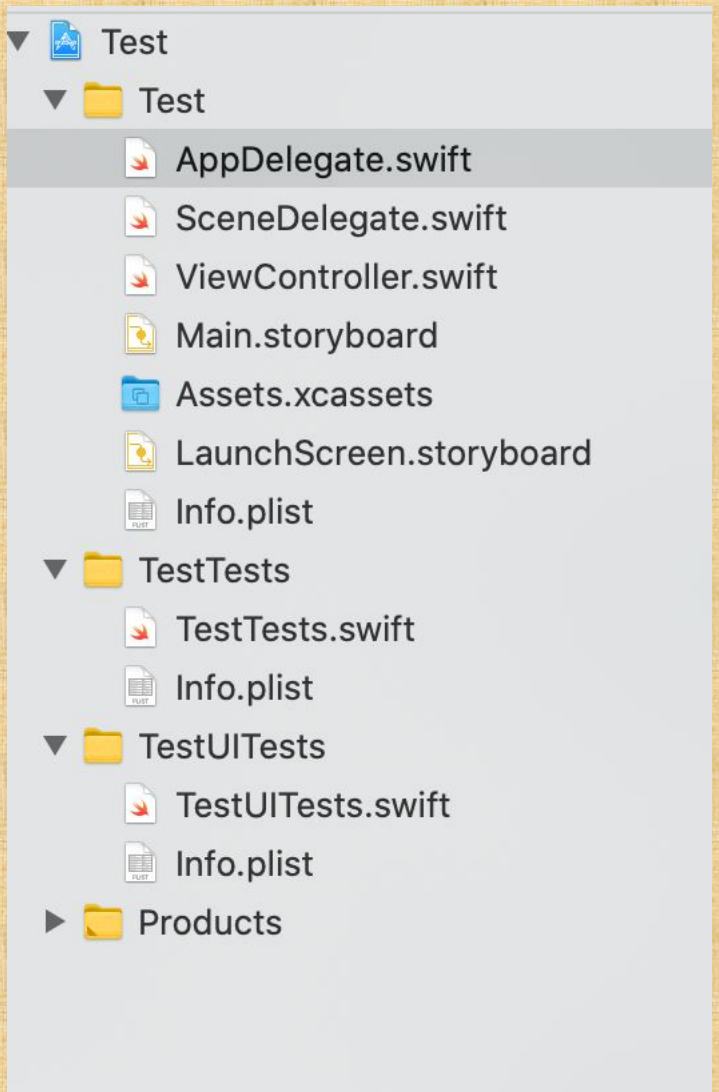
Application Life cycle

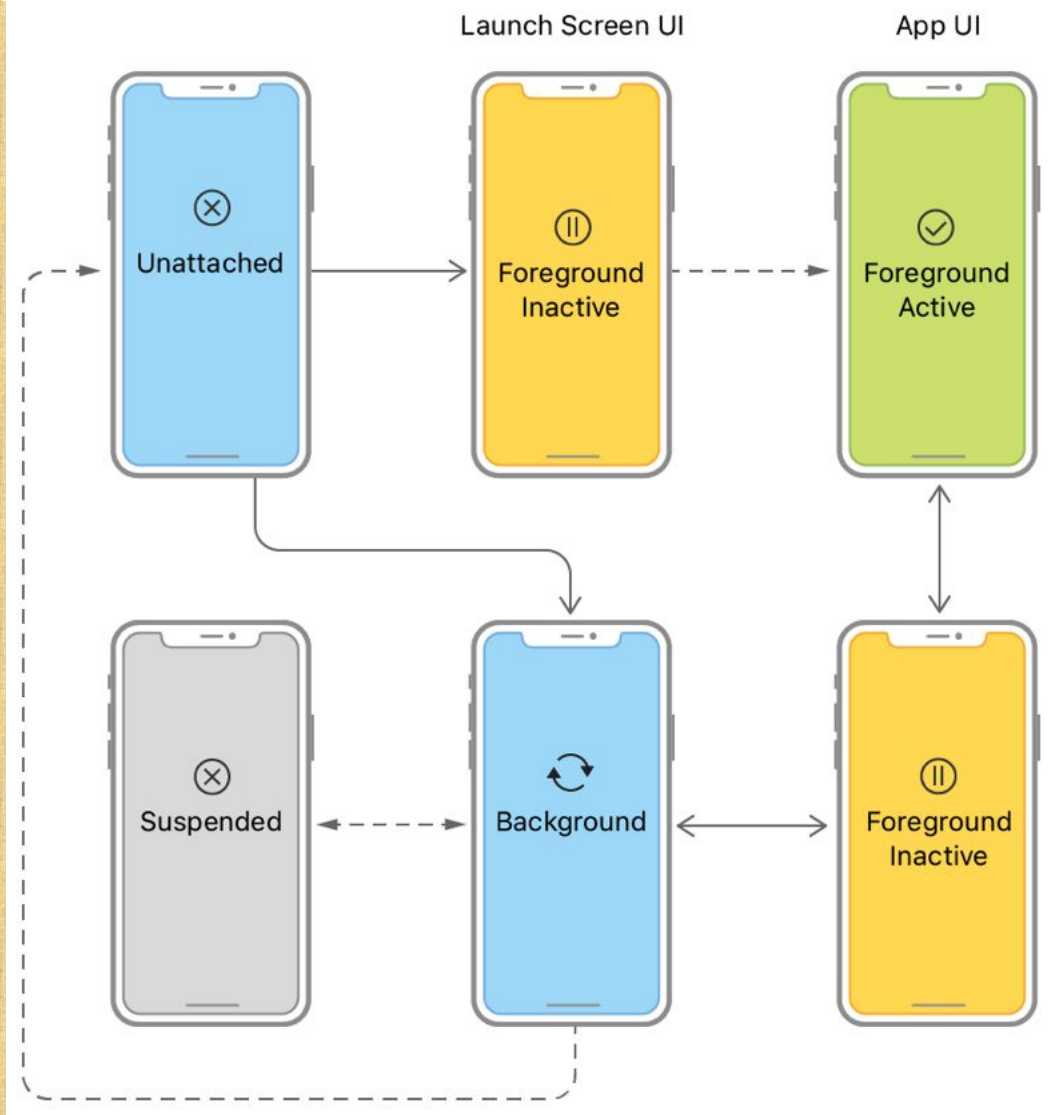


and you are right….all new apps starting iOS 13 supports scenes, by default UISceneDelegate is used instead of UIApplicationDelegate.

The basics remain the same as you will see and we will cover both.

Application Life cycle





Application Lifecycle – On launch

1

```
func application(_ application: UIApplication,
    willFinishLaunchingWithOptions launchOptions:
    [UIApplication.LaunchOptionsKey: Any]?) -> Bool
```

2

```
func application(_ application: UIApplication,
    didFinishLaunchingWithOptions launchOptions:
    [UIApplication.LaunchOptionsKey: Any]?) -> Bool
```

3

```
func applicationDidBecomeActive(_ application:
    UIApplication)
```

Application Lifecycle – App goes to background

1

```
func applicationWillResignActive(_ application:
  UIApplication)
```

2

```
func applicationDidEnterBackground(_
  application: UIApplication
```

Application Lifecycle – App comes to foreground

1

```
func applicationWillEnterForeground(_
    application: UIApplication)
```

2

```
func applicationDidBecomeActive(_ application:
    UIApplication)
```

Application Lifecycle – App kill

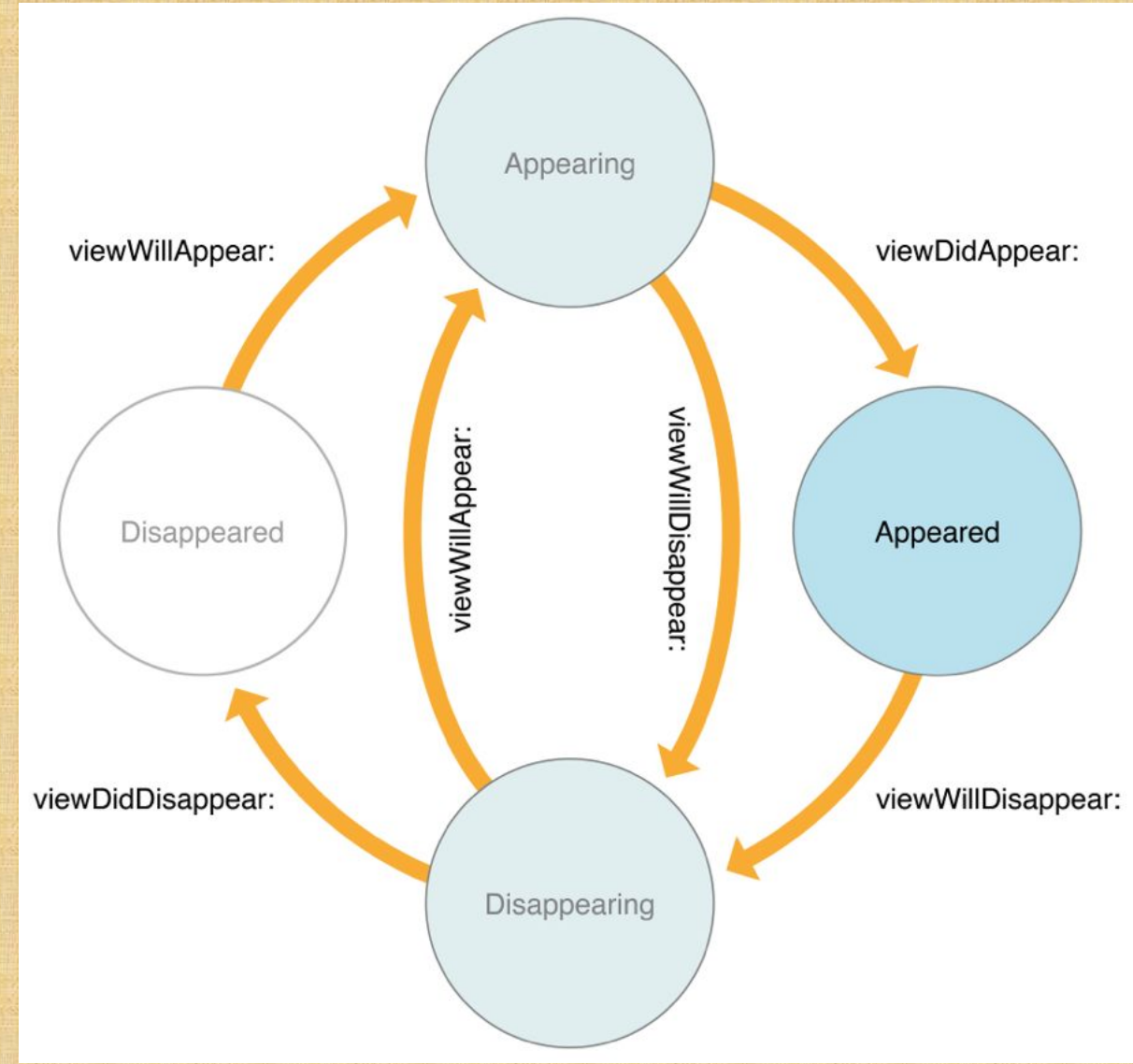
1. **func** applicationWillResignActive(_ application: UIApplication)
2. **func** applicationDidEnterBackground(_ application: UIApplication
3. **func** applicationWillTerminate(_ application: UIApplication)

Application Life cycle

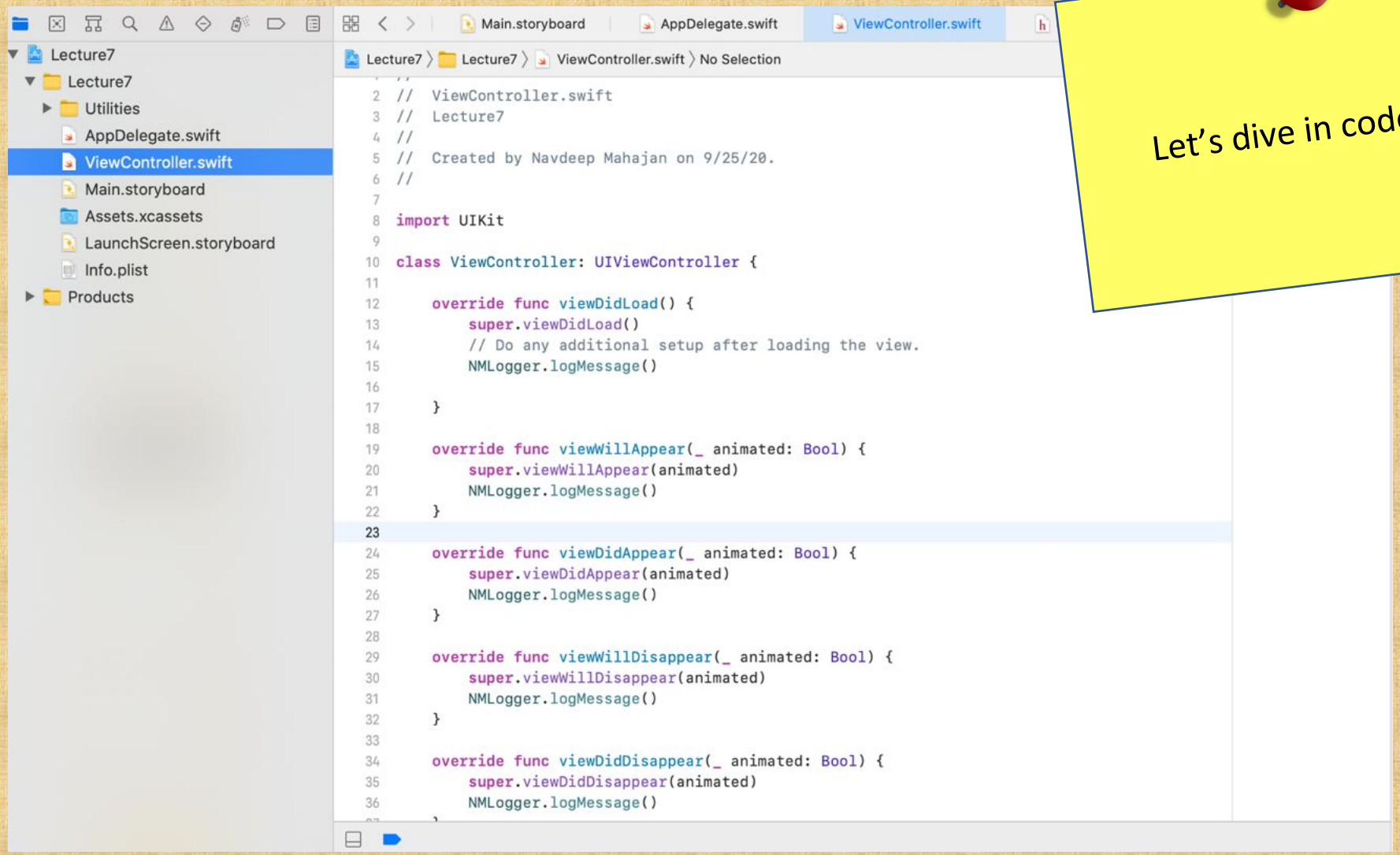
Do the same exercise with Scene Delegates

Notice and record the life cycles when App Delegate and Scene Delegate both are involved

ViewController Life cycle



ViewController Life cycle



Let’s dive in code

Parting Notes

Review:

- Application Life Cycle
- Controller Life Cycles

We want to make sure we are solid on these concepts, otherwise we will end up spending way more time debugging later.

Simple Exercise:

Build a Single Screen App and side deploy to your device if you have one, otherwise run it on the simulator. The App should use:

- Label
- Text Input
- Button Action