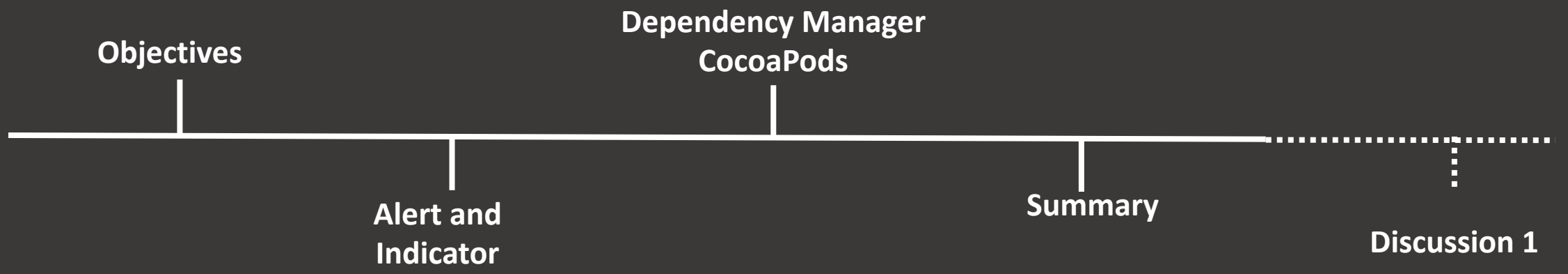
Mobile Applications Development 2 (MAD2)

Diploma in IT

Teaching Team:
Mr Ng Poh Oon
Mr Charles Keck

NGEE ANN
SCHOOL OF INFOCOMM TECHNOLOGY

Chapter 6
Alert View and CocoaPods



Objectives

To be able to understand:

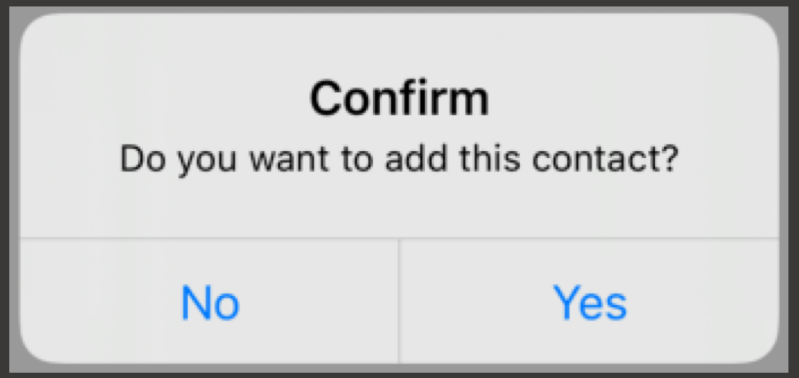
- Alert View with UIAlertController
  - Alert style and ActionSheet style
- **Indicator with UIActivityIndicator view**
- Dependancy Manager, Libraries & Frameworks : CocoaPods

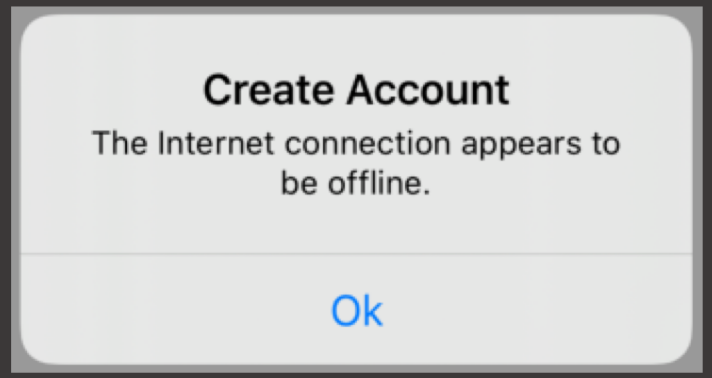
Alert View

MAD2 Oct 2019

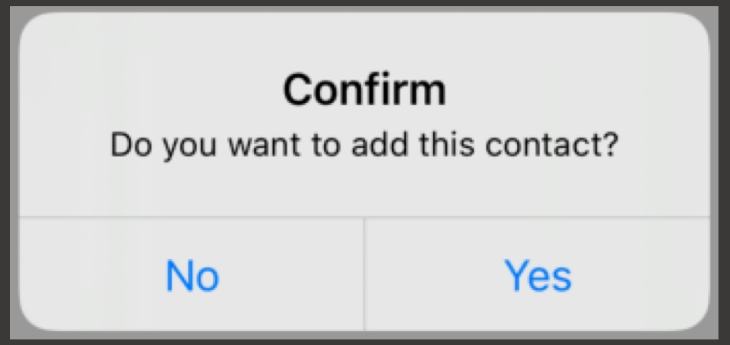
Alert View with UIAlertController

- Used to display an alert message to the user
- Helps to provide good user experience
  - Visual feedback
  - Confirmation for user actions
  - Data validation message
- Configurable buttons





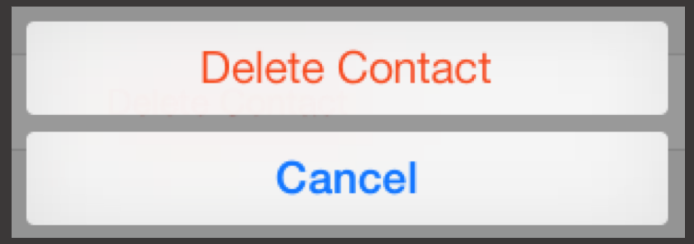
Creating Alert View

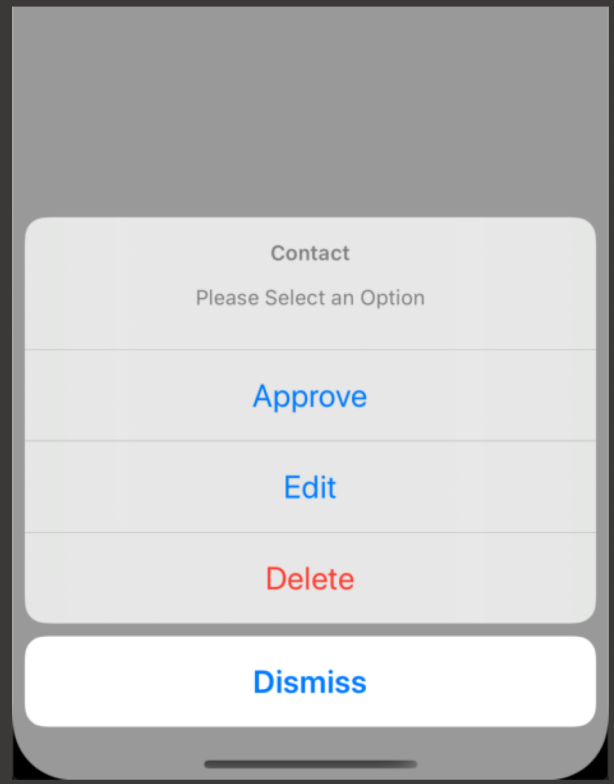



```
@IBAction func showAlert(_ sender: Any) {

        let alertView = UIAlertController(title: "Confirm",
                                          message: "Do you want to add this contact?",
                                          preferredStyle: UIAlertController.Style.alert)

        alertView.addAction(UIAlertAction(title: "No",
                                          style: UIAlertAction.Style.default,
                                          handler: { _ in //to be discussed
        }))
        alertView.addAction(UIAlertAction(title: "Yes",
                                          style: UIAlertAction.Style.default,
                                          handler: { _ in //to be discussed

        }))
        self.present(alertView, animated: true, completion: nil)
    }
```

ActionSheet with UIAlertController

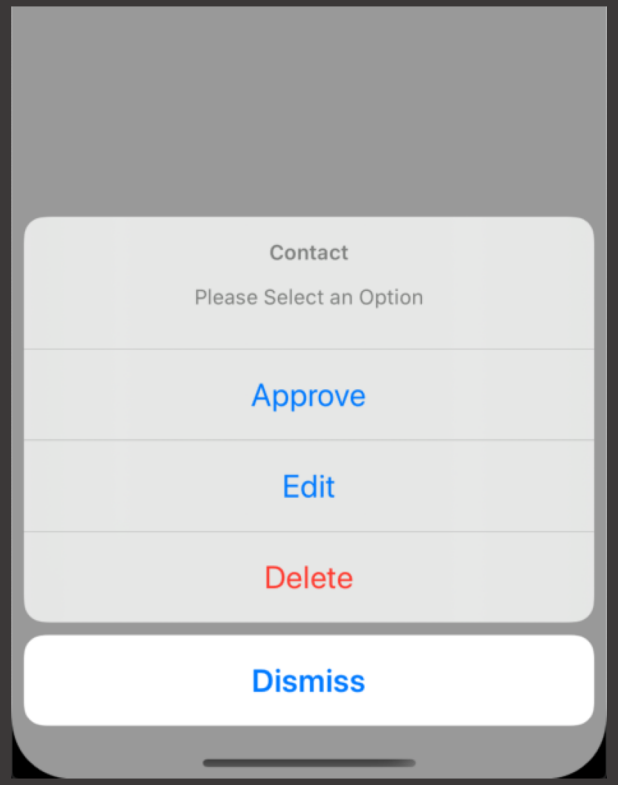
- Like the Alert view, this is a dialog to present the user with a set of alternatives for how to proceed with a given task
- Contains an optional title and one or more buttons





Creating ActionSheet

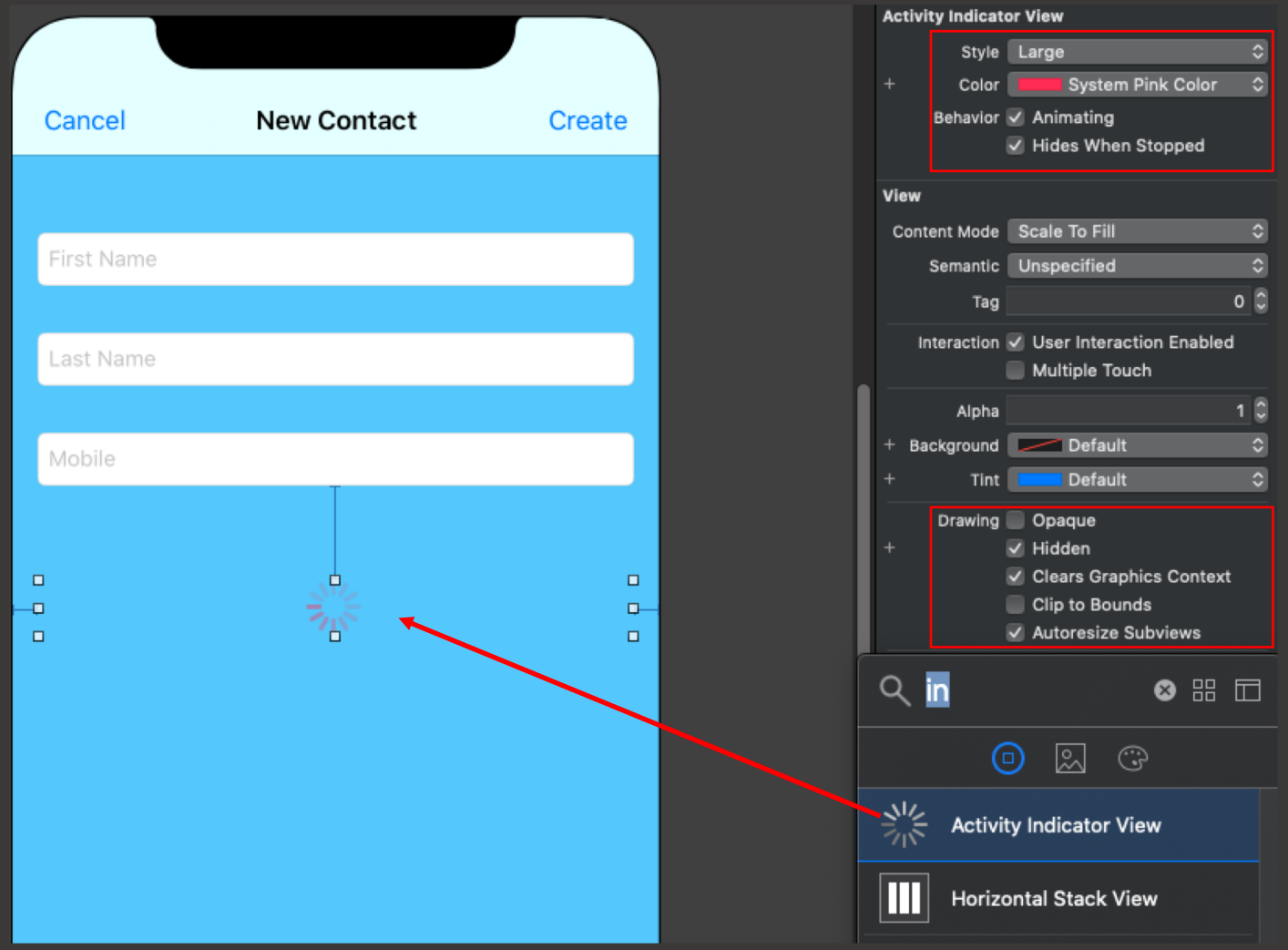
```
@IBAction func showAction(_ sender: Any) {
        let alertAction = UIAlertController(title: "Contact",
                                            message: "Please Select an Option",
                                            preferredStyle: .actionSheet)
        alertAction.addAction(UIAlertAction(title: "Approve",
                                            style: .default,
                                            handler: { (_) in print("User click Approve button")
        }))
        alertAction.addAction(UIAlertAction(title: "Edit",
                                            style: .default,
                                            handler: { (_) in print("User click Edit button")
        }))
        alertAction.addAction(UIAlertAction(title: "Delete",
                                            style: .destructive,
                                            handler: { (_) in print("User click Delete button")
        }))
        alertAction.addAction(UIAlertAction(title: "Dismiss",
                                            style: .cancel,
                                            handler: { (_) in print("User click Dismiss button")
        }))
        self.present(alertAction, animated: true, completion: {
            print("completion block")
        })
    }
```



UIActivityIndicator

- Visual feedback to inform user to “Please wait”
- Tells user something is happening and that the app is still running
- Creates a positive user experience

UIActivityIndicator

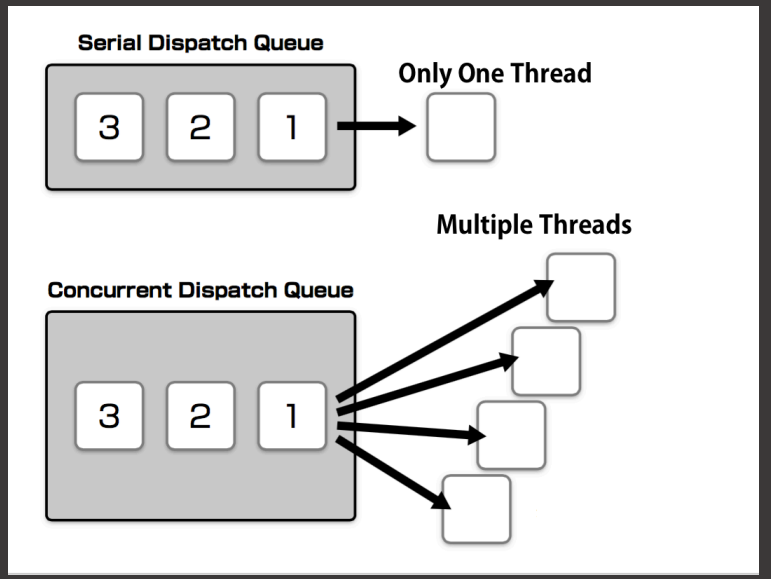


UIActivityIndicatorView

```
@IBOutlet weak var activityIndicator: UIActivityIndicatorView!

    @IBOutlet func onClicked(_ sender: Any) {
        activityIndicator.isHidden = false
        activityIndicator.startAnimating()

        DispatchQueue.global(qos: .userInitiated).async {
            self.testThread()
        }
    }
    func testThread(){
        for i in 1...3 {
            Thread.sleep(forTimeInterval: 3.0) //Sleep for 3 seconds
            print("\(i)")
        }
        DispatchQueue.main.async {
            self.onCompletion()
        }
    }

    func onCompletion(){
        activityIndicator.stopAnimating()
        activityIndicator.isHidden = true
    }
```

DispatchQueue

- Asynchronous operation (not in syllabus)
  - Managing the execution of tasks serially or concurrently on main thread or on background thread



https://developer.apple.com/documentation/dispatch/dispatchqueue

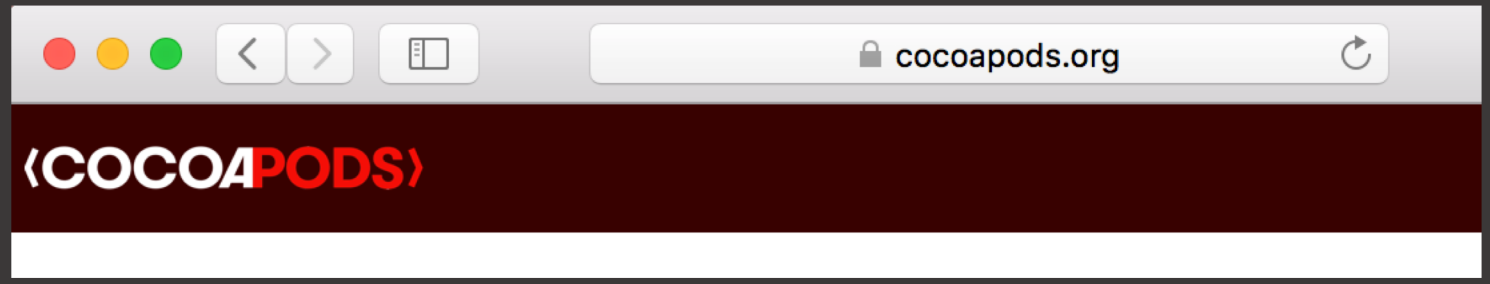
https://medium.com/@michaellong/how-to-chain-api-calls-using-swift-5s-new-result-type-and-gcd-56025b51033c

Libraries & Frameworks
CocoaPods

MAD2 Oct 2019

What is CocoaPods?

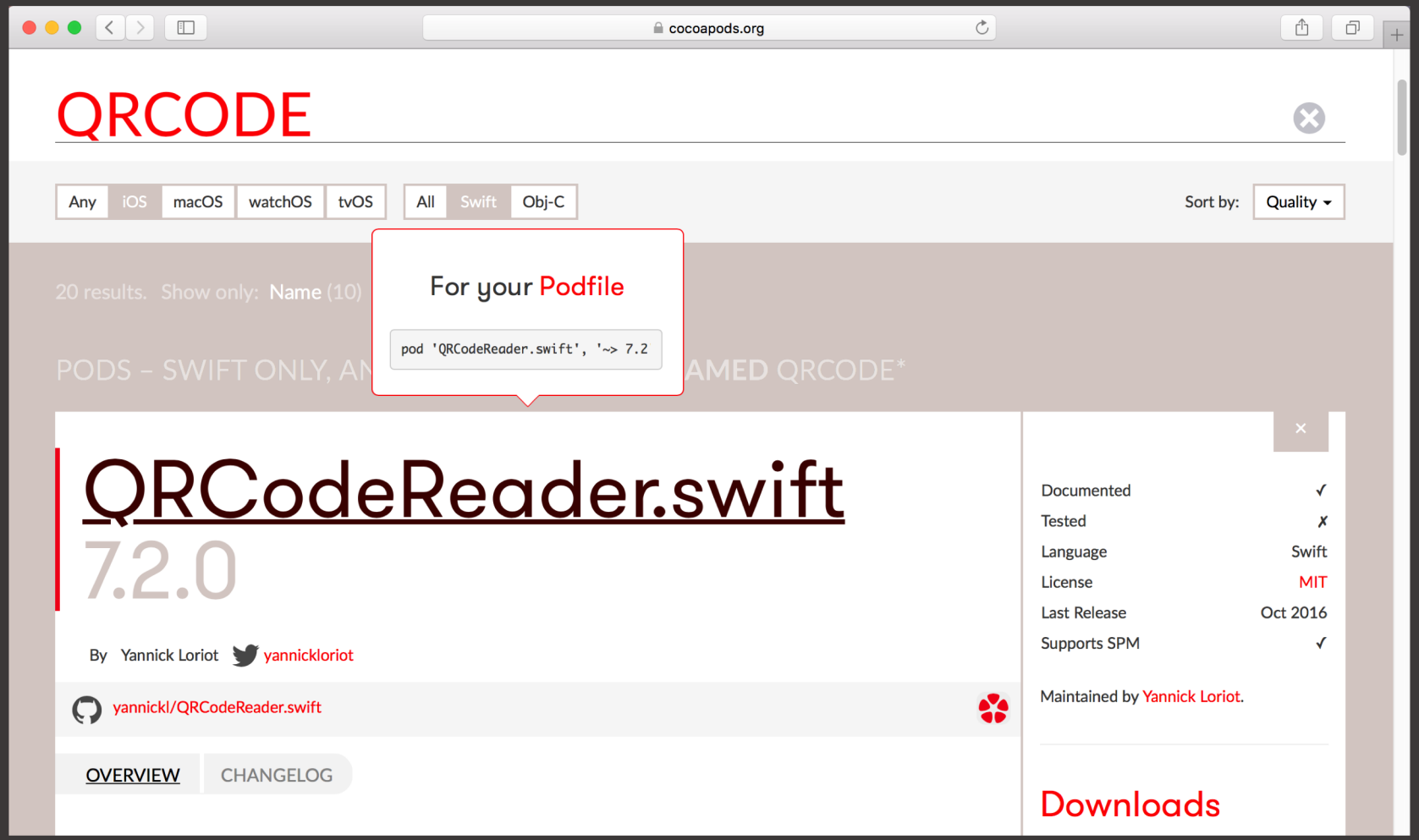
- CocoaPods is a dependency manager for Swift and Objective-C Cocoa projects.
- It has over 53 thousand libraries and is used in over 3 million apps.



Setting up

- https://www.youtube.com/watch?v=iEAjvNRdZa0

Search


cocoapods.org
QRCODE
Any
iOS
macOS
watchOS
tvOS
All
Swift
Obj-C
Sort by:
Quality
20 results. Show only: Name (10)
For your Podfile
pod 'QRCodeReader.swift', '~> 7.2
QRCodeReader.swift
7.2.0
By Yannick Loriot
yannickloriot
yannickl/QRCodeReader.swift
OVERVIEW
CHANGELOG
Documented ✓
Tested ✗
Language Swift
License MIT
Last Release Oct 2016
Supports SPM ✓
Maintained by Yannick Loriot.
Downloads

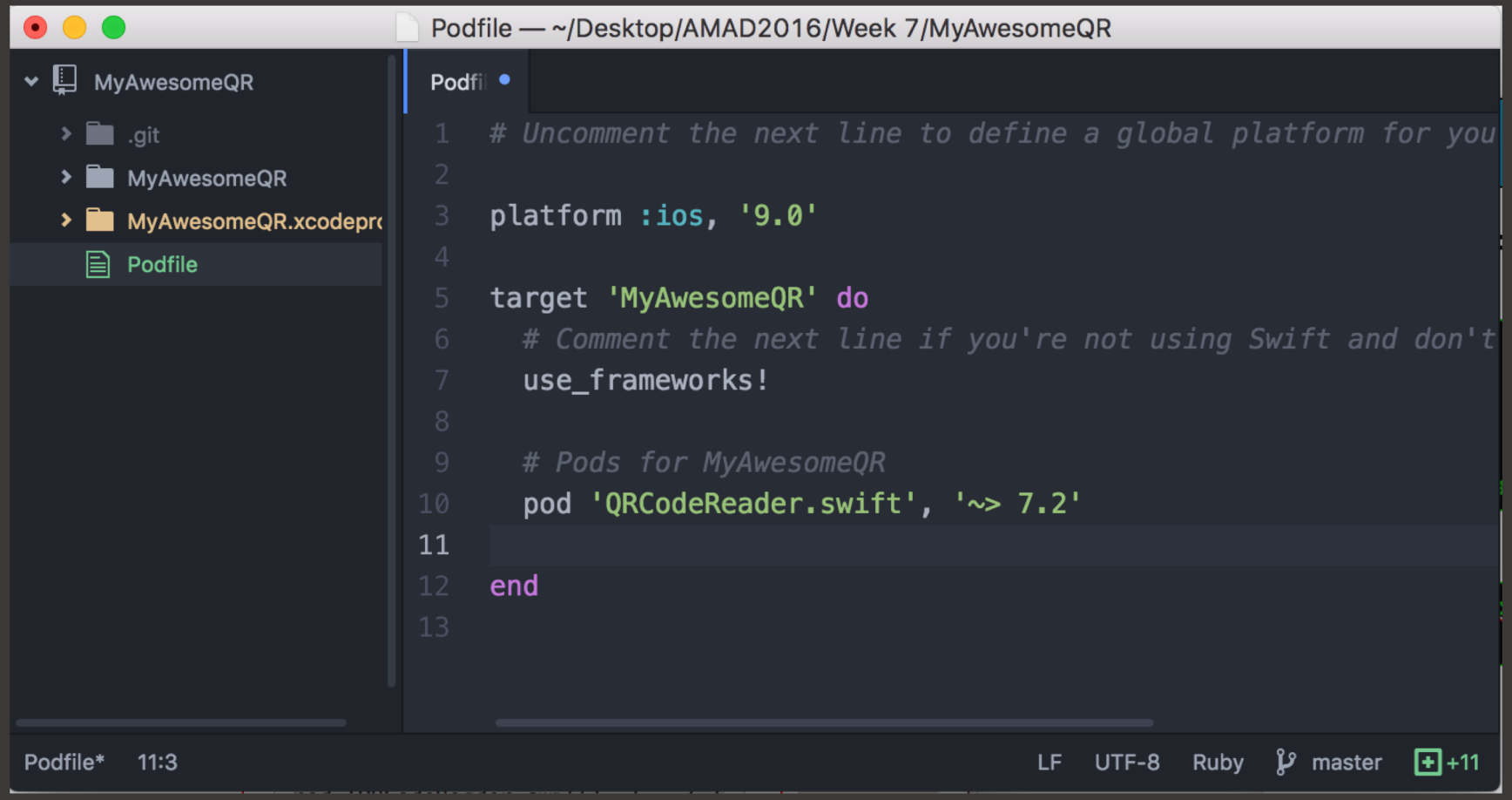
Steps

- Use of “Ruby” GEM ← A dependency manager

```
~> cd /usr/local/bin

~> sudo gem update --system

~> sudo gem install –n /usr/local/bin cocoapods

~> pod setup
```

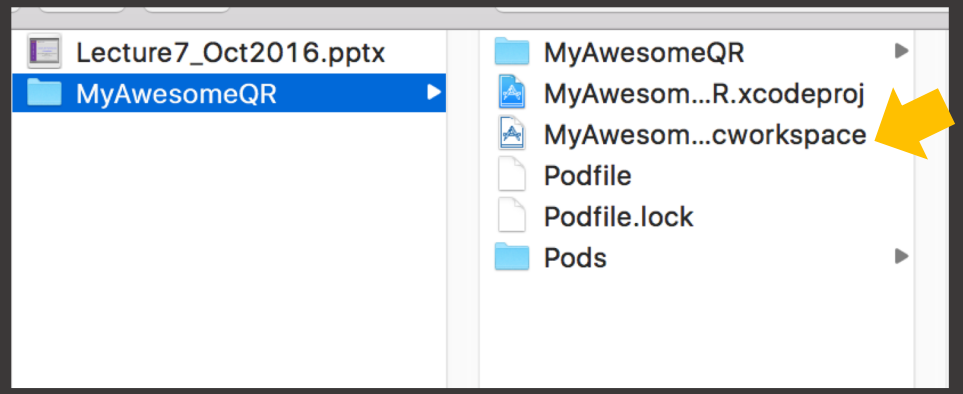
```
MyProject$ pod init
```

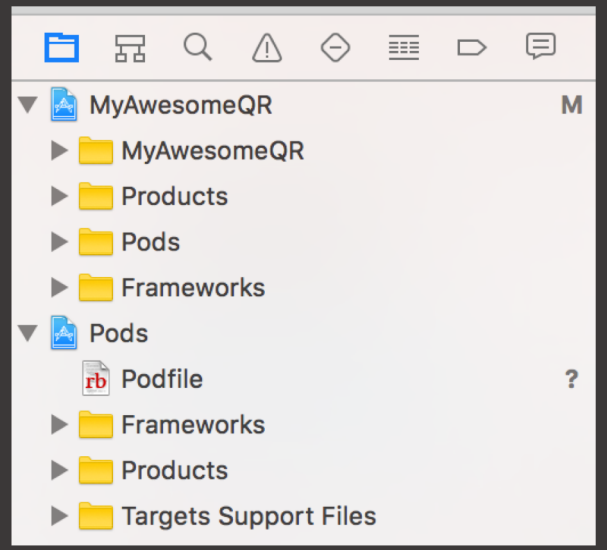
Atom or any text editor



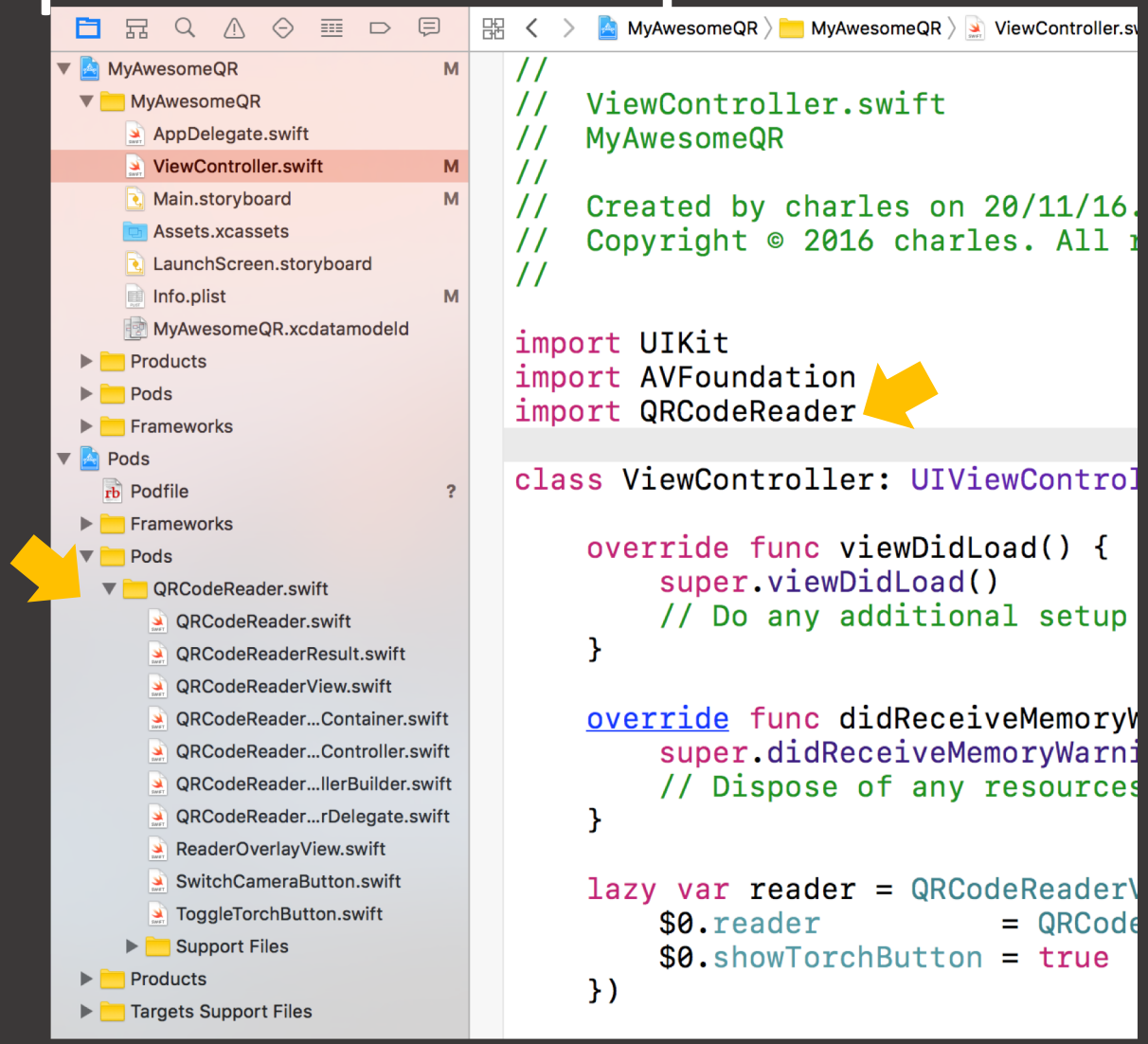
Steps

```
MyProject$ pod install

Analyzing dependencies
Downloading dependencies
Generating Pods project
```





Opening up the xcworkspace



Alternative Dependency Manager

- Carthage
  - Carthage is another simple dependency manager for the Cocoa application. It downloads and build the dependencies but will not change Project file Or Xcode project build setting like CocoaPods. We have to manually drag '. framework' binaries to "Linked Frameworks and Libraries".

```
$ brew install carthage
```

https://github.com/Carthage/Carthage

Carthage

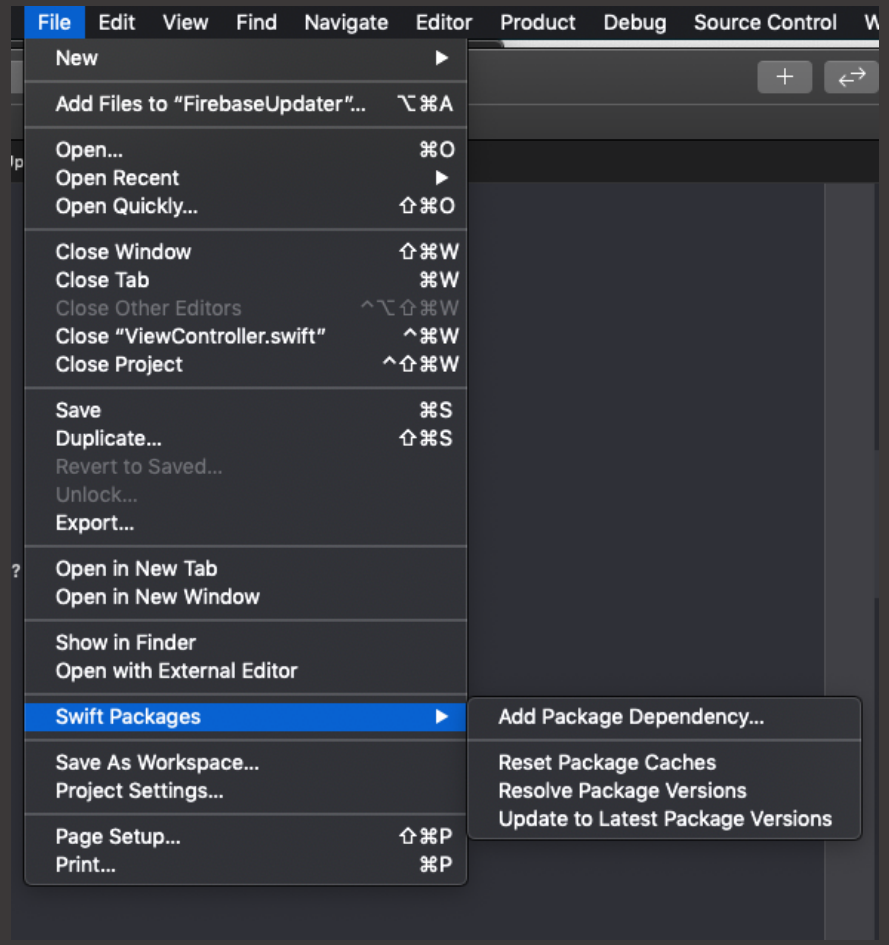
- Pros
  - Carthage won't touch your Xcode settings or project files. It just download and build the dependencies so you have proper control on what you are doing.
  - Decentralised
  - Supports submodules
- Cons
  - Unstable and Slow
  - Small Community, not many contributors
  - Lot of manual steps to perform on Xcode to get everything setup

Alternative Dependency Manager

- Swift Package Manager
  - The Swift Package Manager is a tool for managing the distribution of Swift code. It's integrated with the Swift build system to automate the process of downloading, compiling, and linking dependencies.
  - It's Official Package Manager for Swift
- Swift Package Manager is
  - Command Line Based tool (before Xcode 11)
  - Being Cross-Platform, Swift Package Manager doesn't need Xcode to create package
  - It's decentralised
  - Swift Package Manager is open-source and source code is available on Github

```
$ swift build –- version
Apple Swift Package Manager – Swift 4.2.0 (swiftpm-14460.2)
```

Xcode 11 Swift Package Manager



https://wwdcbysundell.com/2019/xcode-swiftpm-first-look/

Summary

Understand Alert View with UIAlertController

Understand the Dependancy Manager concepts, Libraries & Frameworks : CocoaPods