

Towards Modular App

Demystify the process

Rimesh Jotaniya
<https://recursiveswift.com>

Rimesh Jotaniya

Independent iOS developer

- 7 years working on iOS app development
- 5 years worked as a solo iOS developer
- Traveling 
- Trekking 
- Swimming 



Starting your career as iOS developer



Intern / Junior iOS developer in a team

- Focus on specific task
- Get mentorship
- Learn collaboration and team work
- Follow predefined architecture



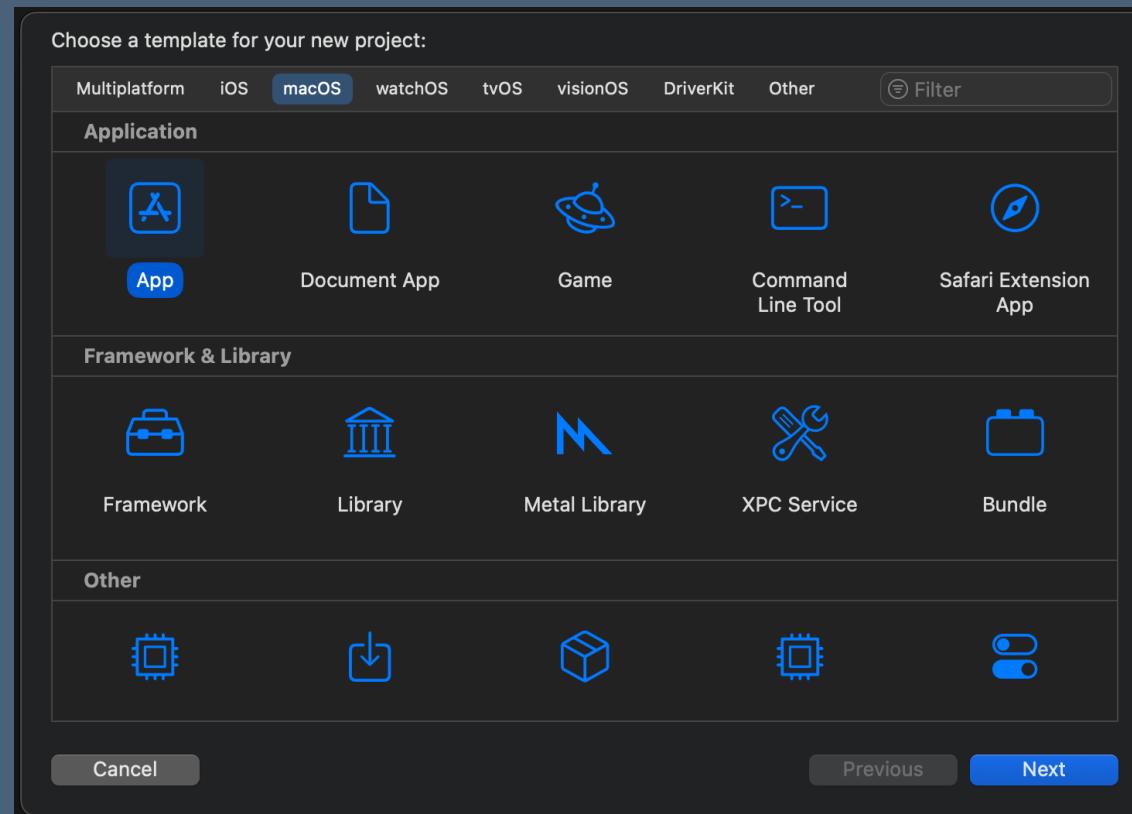
Solo iOS developer

- Build everything from scratch
- Learning from YouTube and courses
- Ownership of the work
- Monolithic architecture

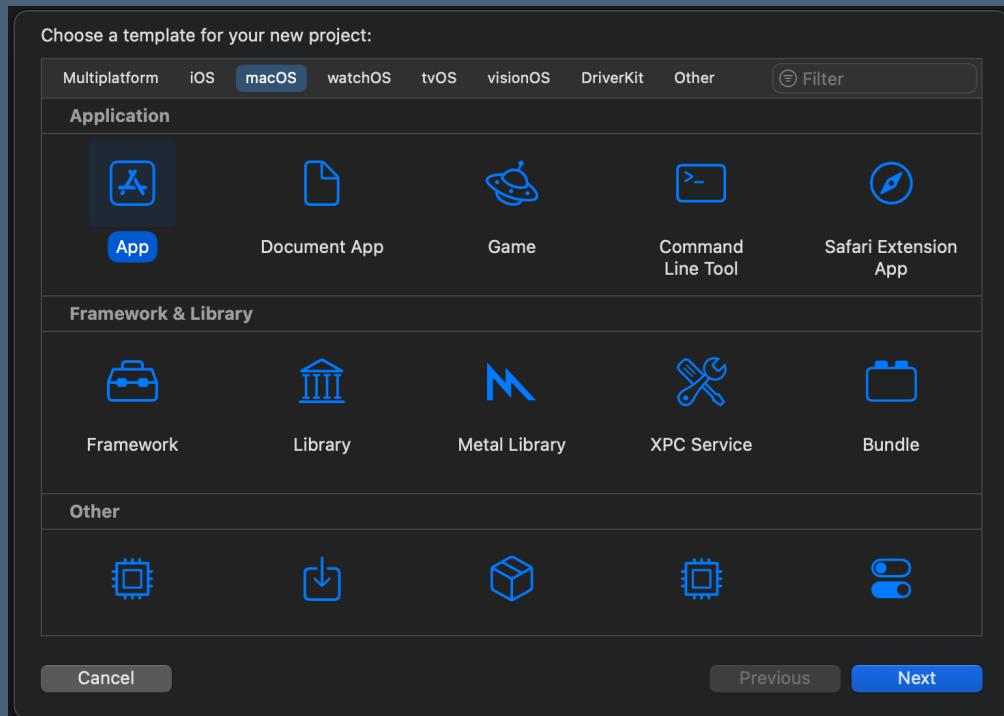
Convenient, but at what cost?

- Build time
- Inability to reuse code
- Inefficient collaboration between teams
- Debugging issues

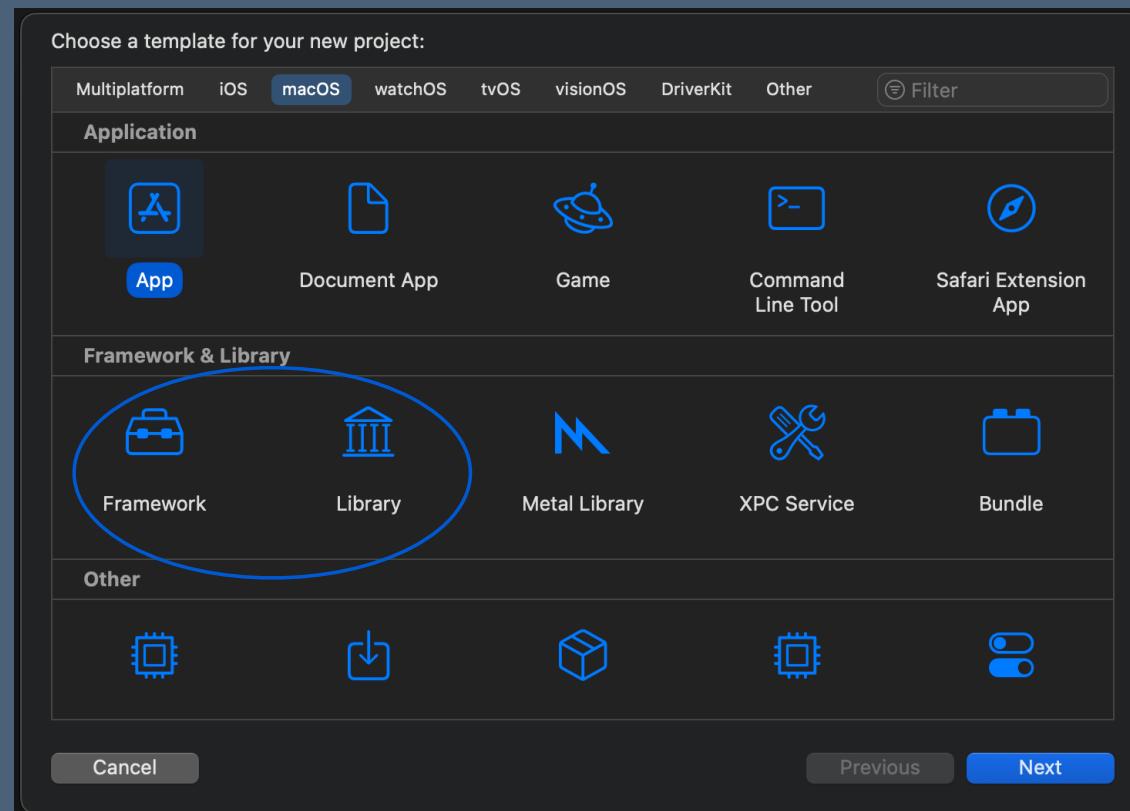
Starting a new project



Starting a new project



Starting a new project



It's 2025

It's 2025



Google



deepseek

Framework

Define a Framework in context of iOS development

In the context of iOS development, a **framework** is a pre-built collection of code, resources, and tools that developers can use to build apps more efficiently. It encapsulates a set of related functionalities, allowing developers to leverage existing solutions rather than writing everything from scratch.

Apple provides many built-in frameworks for iOS development, such as **UIKit** for user interfaces, **CoreData** for local data storage, **CoreML** for machine learning, and **ARKit** for augmented reality.

Library

Define a Library in context of iOS development

In the context of iOS development, a **library** is a collection of precompiled code that provides reusable functions and routines that developers can integrate into their apps to avoid reinventing the wheel. Unlike frameworks, which are more comprehensive and often contain both code and resources (like images, storyboards, etc.), a library is typically a more focused set of functions that handle specific tasks.

SwiftyJSON is a library that simplifies JSON parsing, SDWebImage is a library for image downloading and caching, and Lottie is a library for displaying JSON-based animations in iOS development.

Framework - Apple

UIKit, SwiftUI, CoreData, CoreML



Library - Third party devs

SDWebImage, Alamofire, SnapKit

“NO”

Does Apple provide any Library?

Framework

Core ML

Integrate machine learning models into your app.

iOS 11.0+ | iPadOS 11.0+ | Mac Catalyst 13.0+ | macOS 10.13+ | tvOS 11.0+ | visionOS 1.0+ | watchOS 4.0+

Your app

Vision

Natural Language

Speech

Sound Analysis

Core ML

Accelerate and BNNS

Metal Performance Shaders

Accelerate / BNNS

API Collection

BNNS

Implement and run neural networks for training and inference.

Overview

The Accelerate framework's BNNS library is a collection of functions that you use to construct neural networks for training and inference. BNNS provides routines optimized for high performance and low energy consumption across all Apple platforms.



Does third party devs provide Framework?



Compose Multiplatform

Develop stunning shared UIs for Android, iOS, desktop, and web.



Declarative framework for sharing UIs across multiple platforms. Based on Kotlin Multiplatform and Jetpack Compose.

Developed by JetBrains

Anyone can create
Framework and Library.

Adding Awesomeness

```
import SwiftUI
import Awesomeness
```

No such module 'Awesomeness'

```
struct AwesomeView: View {
    var body: some View {
        Text("Awesome View")
    }
}

#Preview {
    AwesomeView()
}
```



Module?

Module?

Swift organizes code into *modules*. Each module specifies a namespace and enforces access controls on which parts of that code can be used outside of the module.

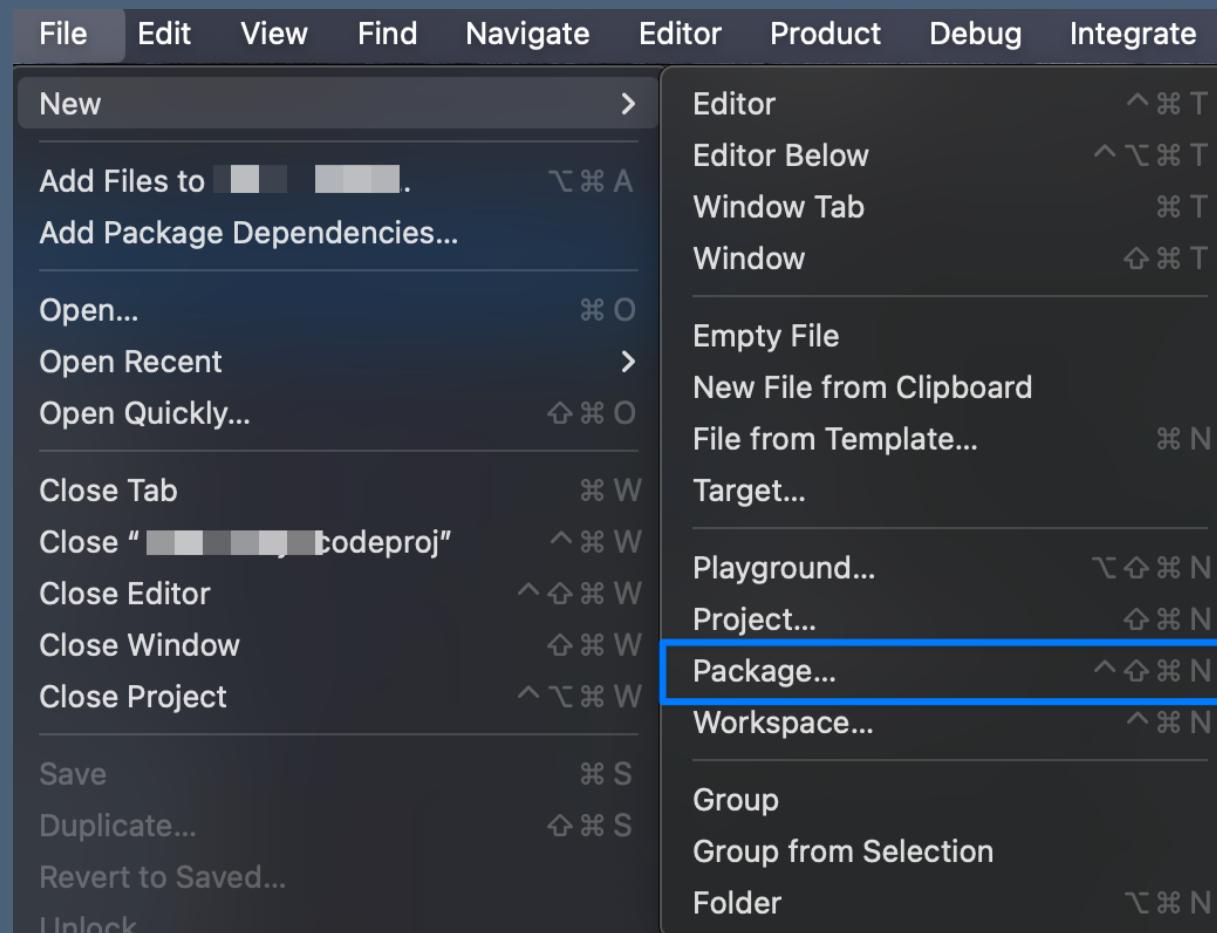
- swift.org

Whether it's Framework or Library, Swift deals in Modules.

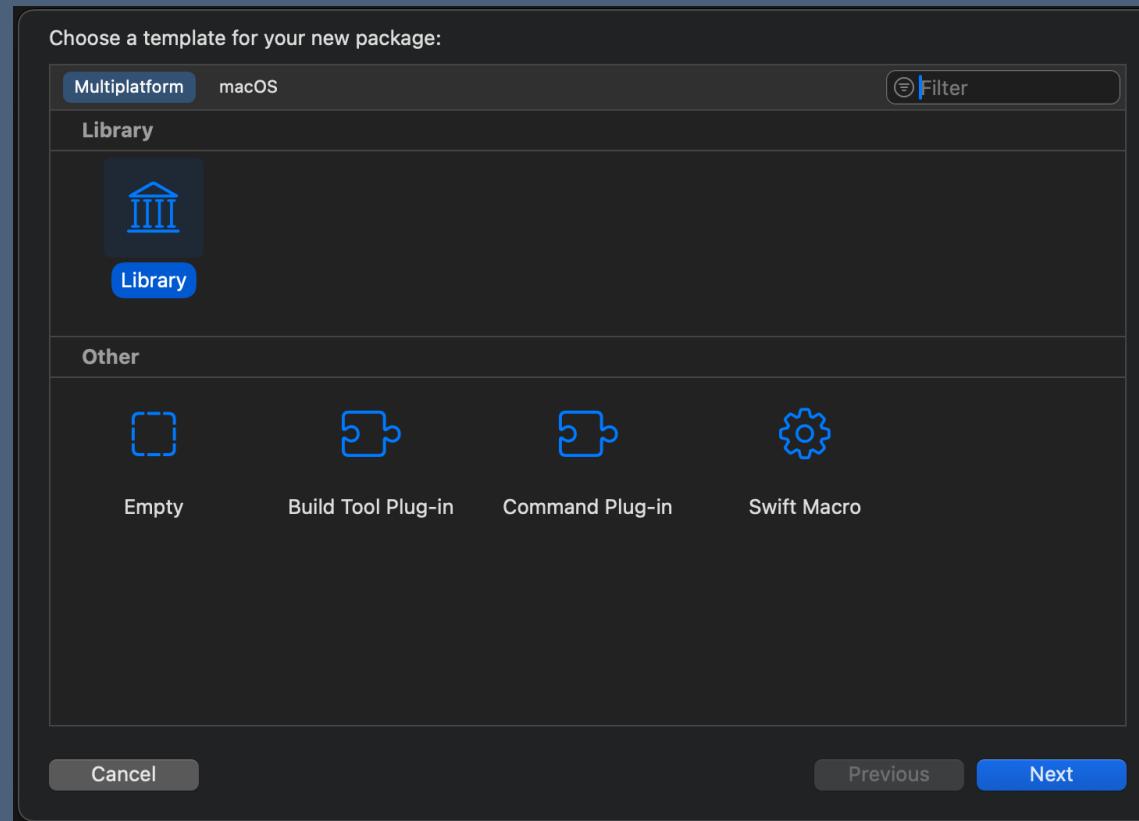
How to create Module?

- Manually creating Framework / Library
- Swift Package Manager
- Using CocoaPods
- Other third party solutions

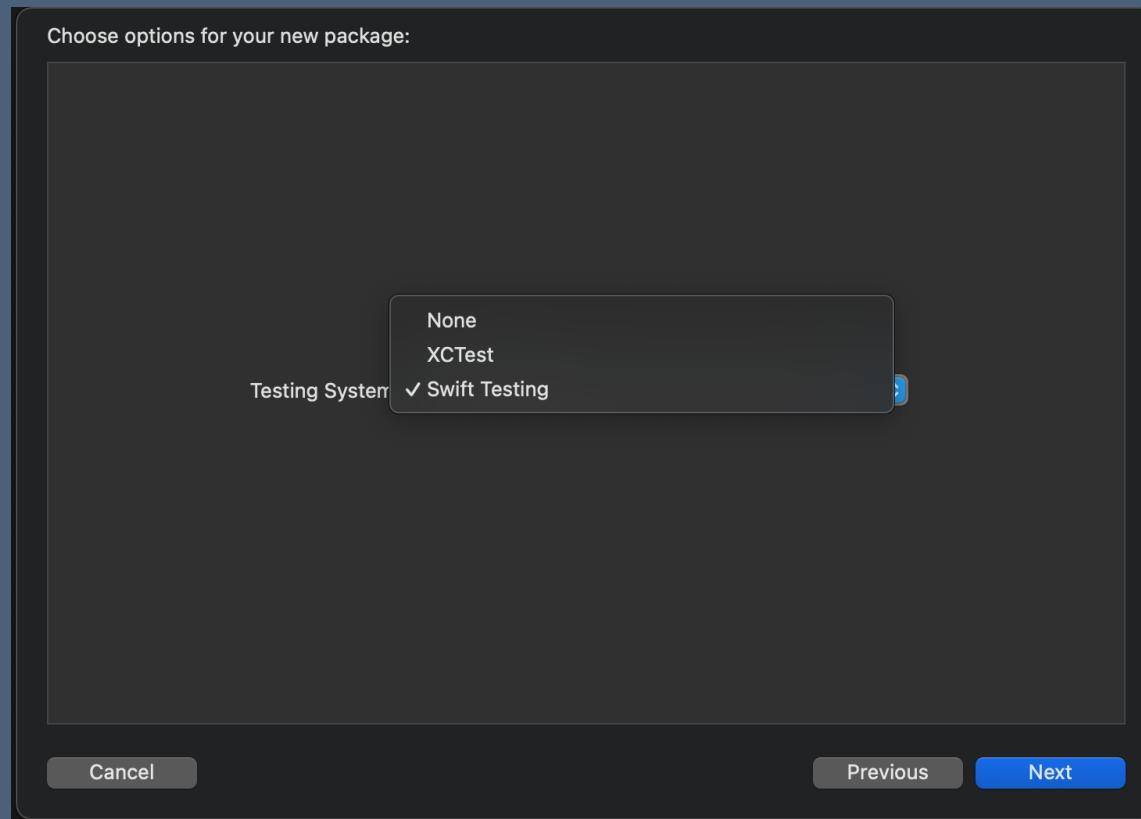
Create a new package



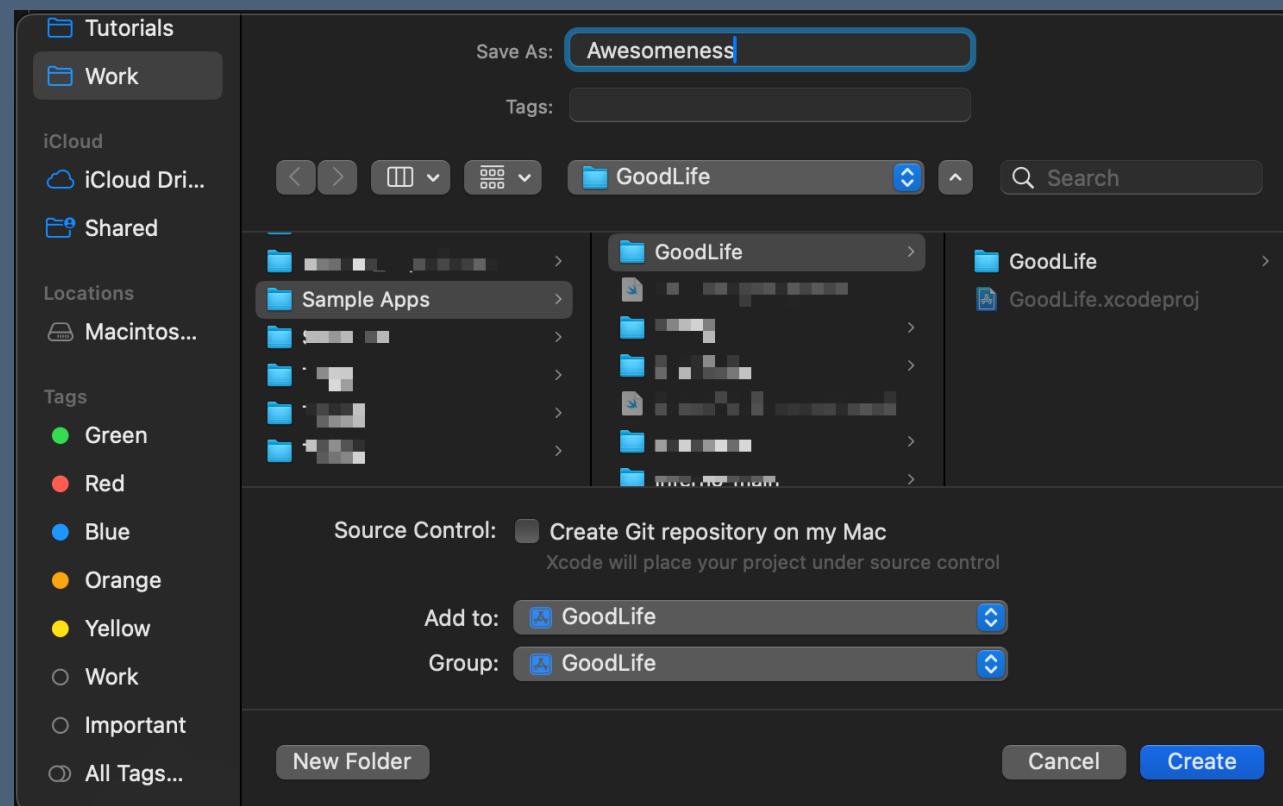
Create a new package



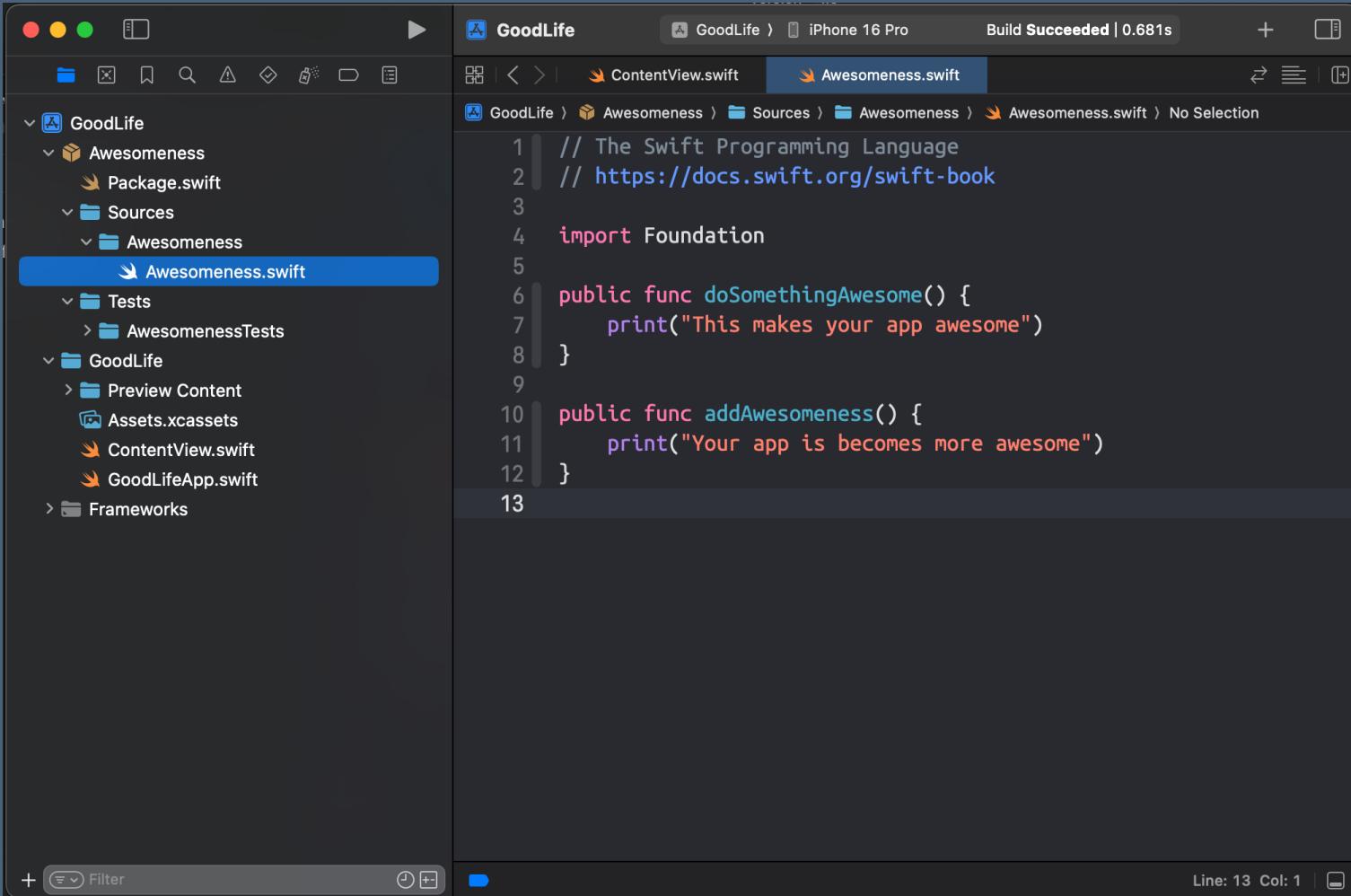
Create a new package



Create a new package



Add your code to package



The screenshot shows the Xcode interface with a dark theme. On the left is the Project Navigator, displaying a file tree for a project named "GoodLife". The "Awesomeness" folder under "Sources" contains a file named "Awesomeness.swift", which is currently selected and highlighted with a blue background. The main area is the Code Editor, showing the contents of "Awesomeness.swift". The code is as follows:

```
// The Swift Programming Language
// https://docs.swift.org/swift-book

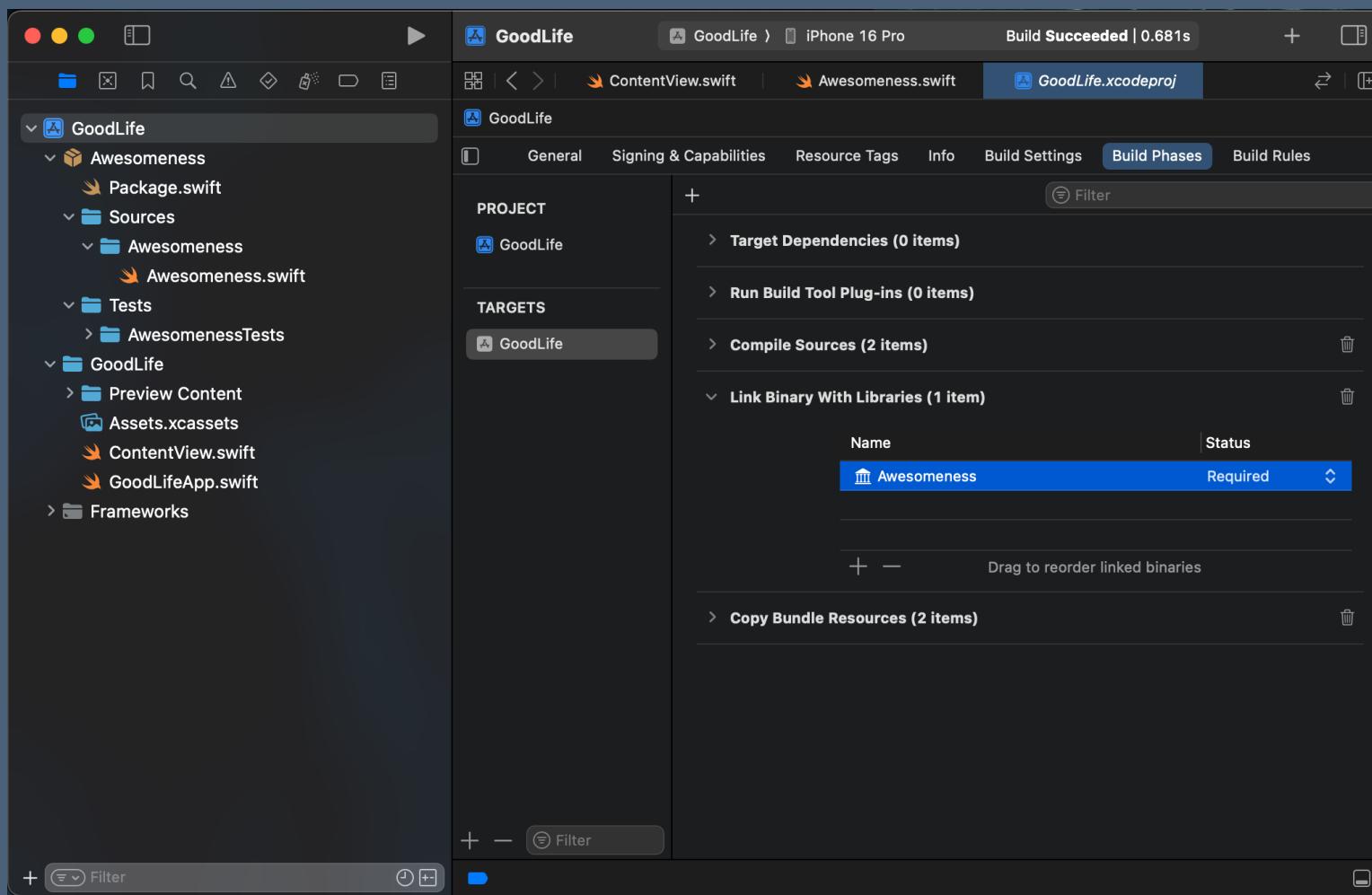
import Foundation

public func doSomethingAwesome() {
    print("This makes your app awesome")
}

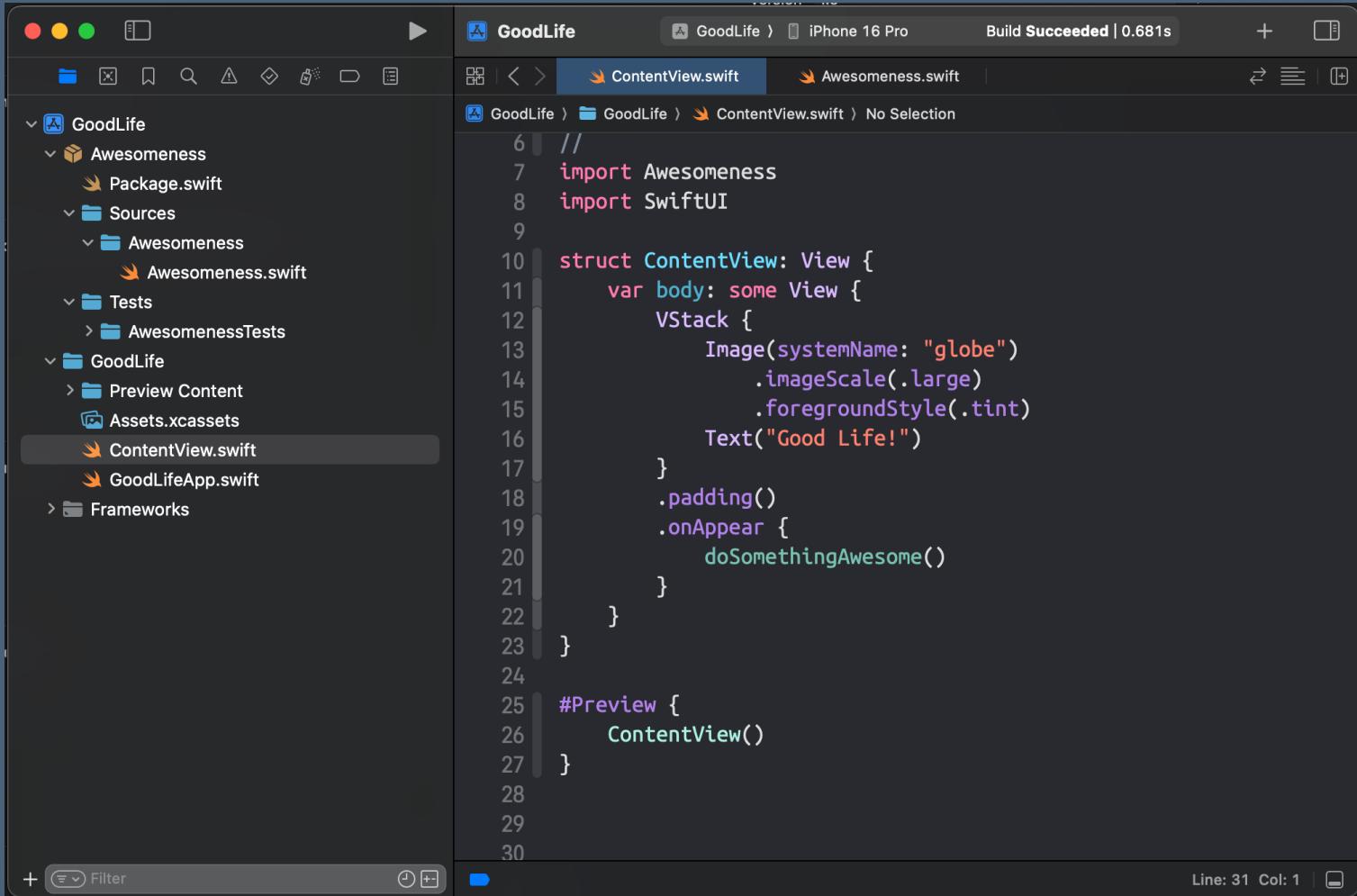
public func addAwesomeness() {
    print("Your app is becomes more awesome")
}
```

The status bar at the bottom right indicates "Line: 13 Col: 1".

Add the library



Using the package in your project



The screenshot shows the Xcode interface with a project named "GoodLife". The left sidebar displays the project structure:

- GoodLife (Project root)
- ↳ Awesomeness (Swift Package)
 - Package.swift
 - Sources
 - ↳ Awesomeness (Folder)
 - Awesomeness.swift
 - Tests
 - ↳ AwesomenessTests (Folder)
 - GoodLife
 - Preview Content
 - Assets.xcassets
 - ContentView.swift (selected)
 - GoodLifeApp.swift
 - Frameworks

The main editor window shows the ContentView.swift file:

```
//  
import Awesomeness  
import SwiftUI  
  
struct ContentView: View {  
    var body: some View {  
        VStack {  
            Image(systemName: "globe")  
                .imageScale(.large)  
                .foregroundStyle(.tint)  
            Text("Good Life!")  
        }  
        .padding()  
        .onAppear {  
            doSomethingAwesome()  
        }  
    }  
}  
  
#Preview {  
    ContentView()  
}
```

The status bar at the bottom right indicates "Line: 31 Col: 1".

Modular architecture demystified

- What - Networking, Analytics, UIComponents
- Why - Reduced built time, isolated tests per module, improved collaboration
- When - You can always start simple
- More Modules != Better Modularization

Where to go from here?

- Version management
- Dependency graph resolution
- Binary distribution
- Static and Dynamic linking
- Sharing your package
- Create your first package

Questions?

Thank you!