



Mastering SwiftPM

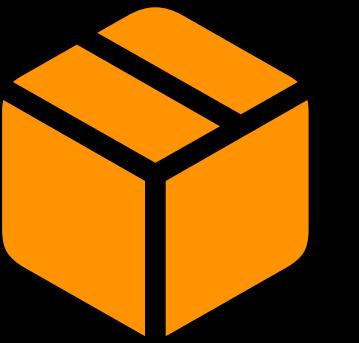


Swift Package manager

Ratnesh Jain



Mastering SwiftPM



Today

- Access Modifiers
- Package Manifest
- Package Manager Vocabulary
- Dependencies
- New Feature using SwiftPM
- Integration with Xcode Project
- Sharing Resources
- Dealing with Non-SPM external Dependency



Access Modifiers



Access Modifiers

Allow what should be visible/readable/sub-classable/overridable

- Public
- Private
- FilePrivate
- Open
- Internal



Mastering SwiftPM

Today

- Access Modifiers
- Package Manifest
- Package Manager Vocabulary
- Dependencies
- New Feature using SwiftPM
- Integration with Xcode Project
- Sharing Resources
- Dealing with Non-SPM external Dependency



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Package Manifest

Package.swift

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    products: [
        // Products define the executables and libraries a package produces, making them visible to other packages.
        .library(
            name: "MasteringSwiftPM",
            targets: ["MasteringSwiftPM"]),
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module or a test suite.
        // Targets can depend on other targets in this package and products from dependencies.
        .target(
            name: "MasteringSwiftPM"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["MasteringSwiftPM"]),
    ]
)
```



Mastering SwiftPM

Today

- Access Modifiers
- Package Manifest
- Package Manager Vocabulary
- Dependencies
- New Feature using SwiftPM
- Integration with Xcode Project
- Sharing Resources
- Dealing with Non-SPM external Dependency



Package Manager Vocabulary

Terminology used by Swift Package Manager

```
import SwiftUI

public struct FetchingView: View {
    public var body: some View {
        VStack {
            ProgressView()
            Text("Loading")
        }
    }
}
```



Package Manager Vocabulary

Directory Structure

```
import SwiftUI

public struct FetchingView: View {
    public var body: some View {
        VStack {
            ProgressView()
            Text("Loading")
        }
    }
}
```

FetchingView.swift



Package Manager Vocabulary

Directory Structure

```
import SwiftUI

public struct FetchingView: View {
    public var body: some View {
        VStack {
            ProgressView()
            Text("Loading")
        }
    }
}
```

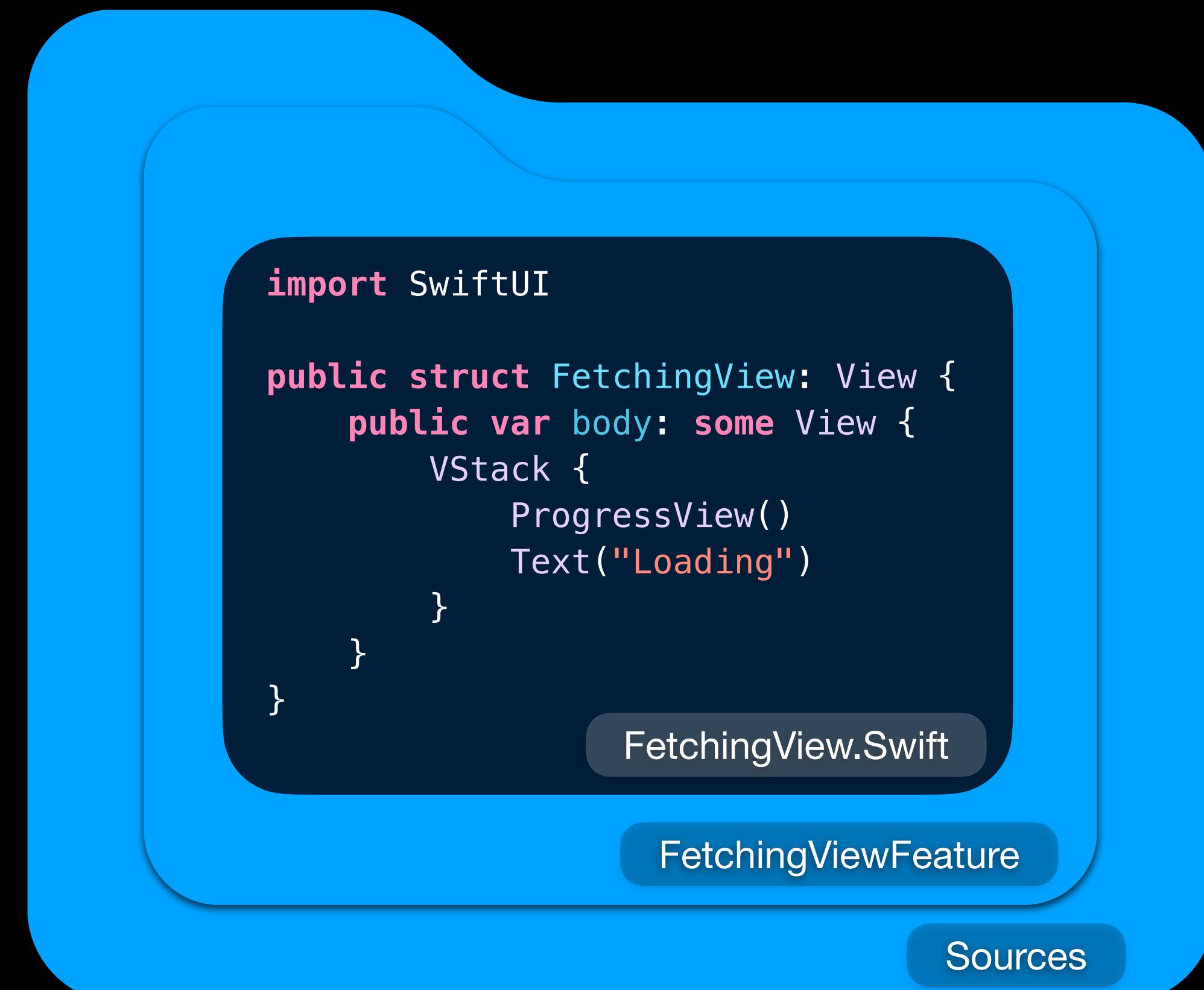
FetchingView.swift

FetchingViewFeature



Package Manager Vocabulary

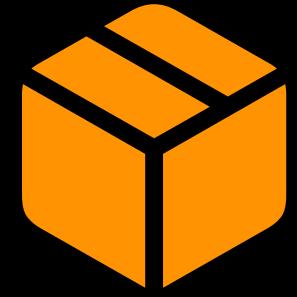
Directory Structure







Mastering SwiftPM



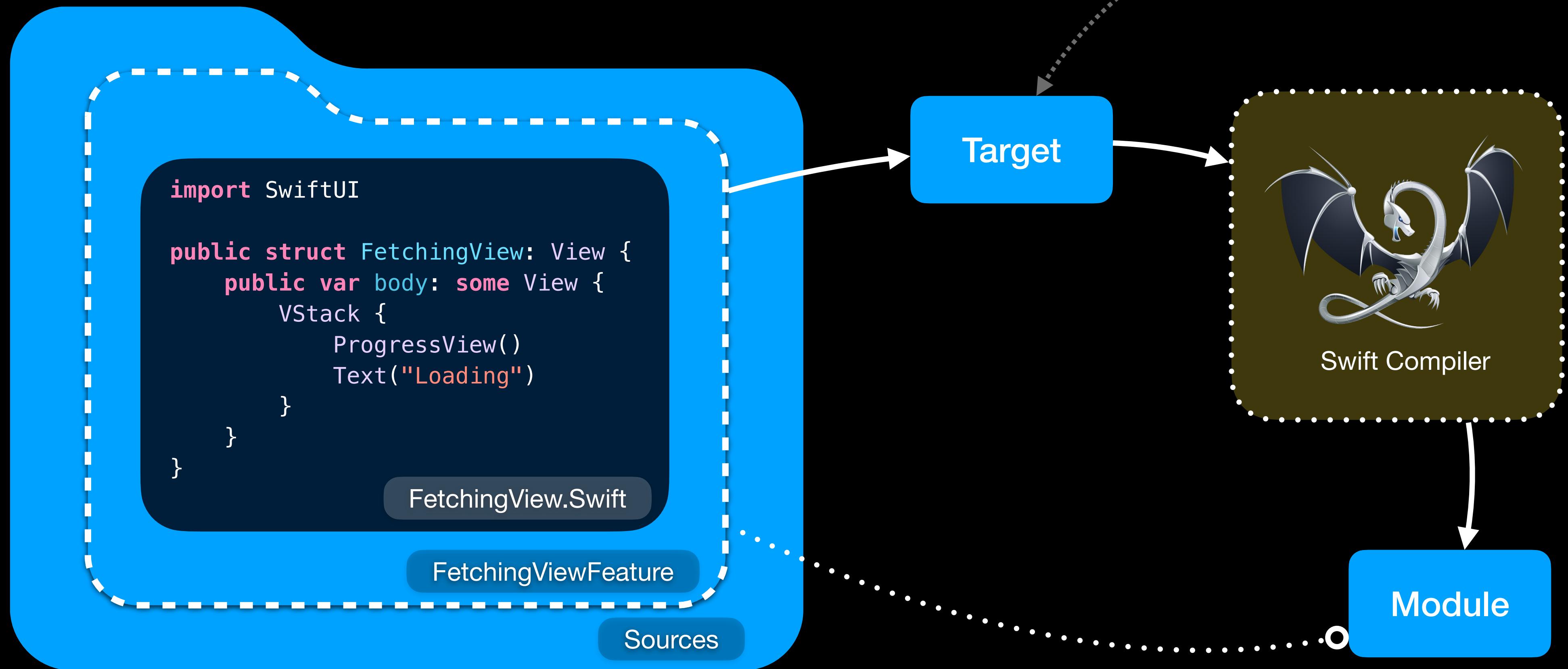
```
import SwiftUI

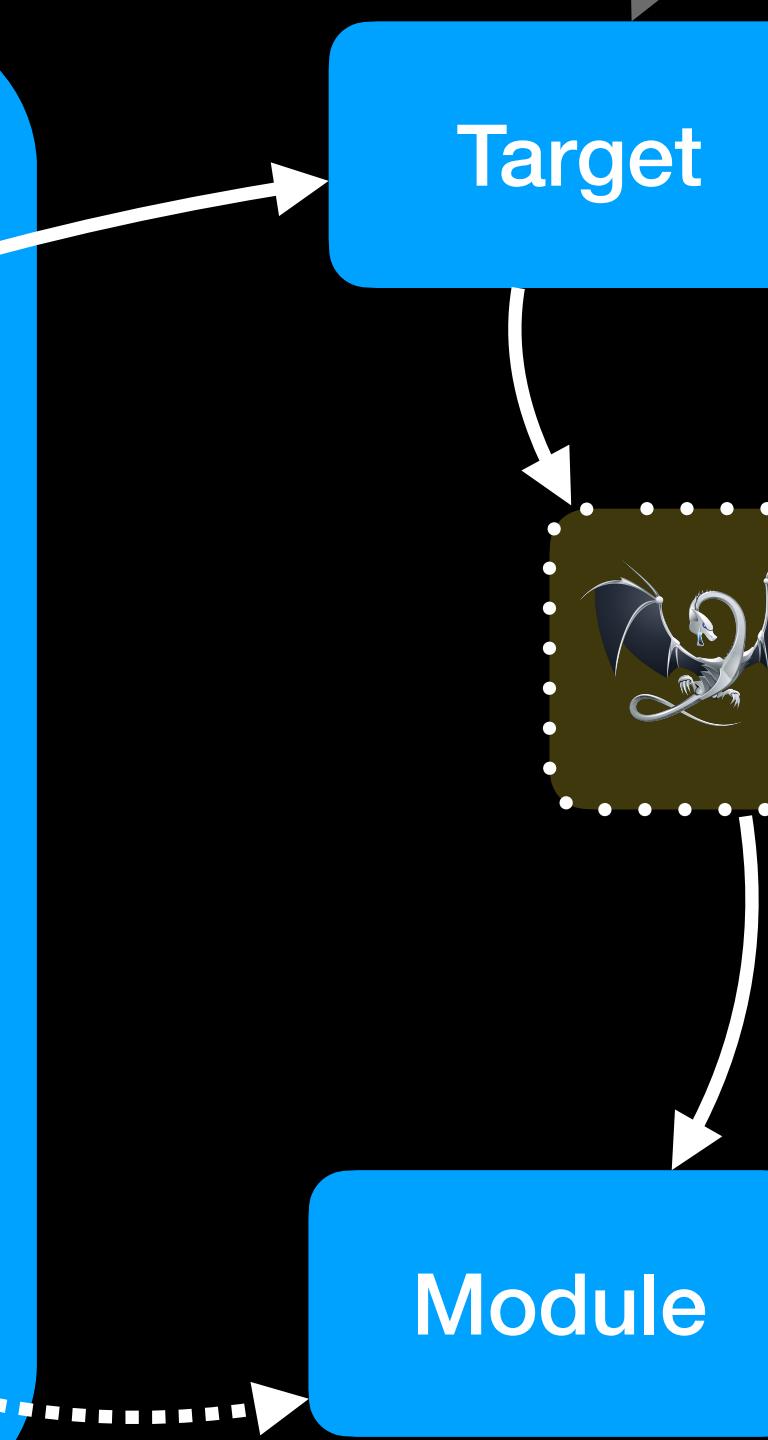
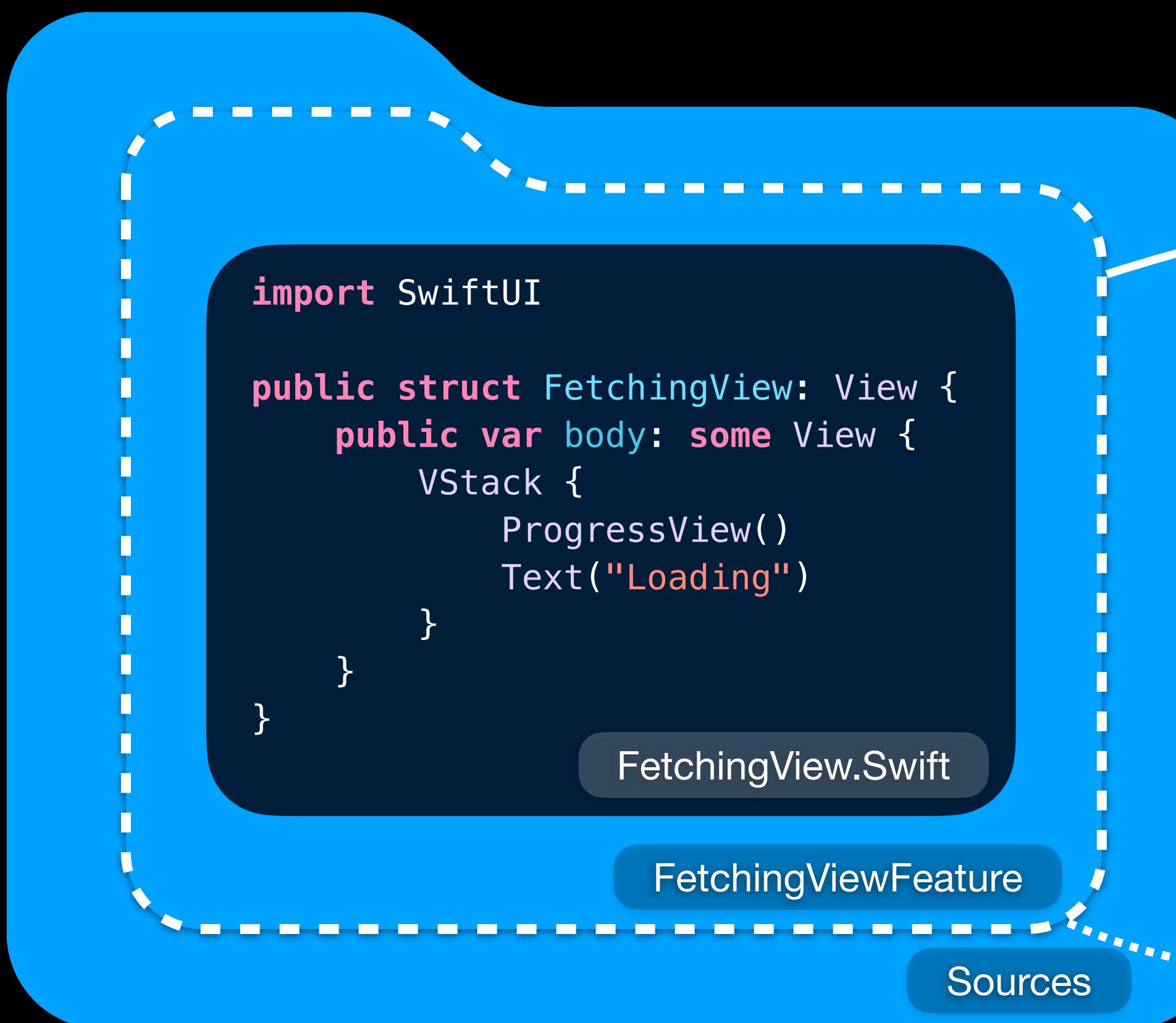
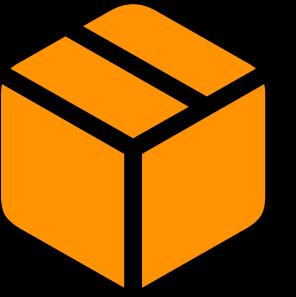
public struct FetchingView: View {
    public var body: some View {
        VStack {
            ProgressView()
            Text("Loading")
        }
    }
}
```

FetchingView.swift

FetchingViewFeature

Sources

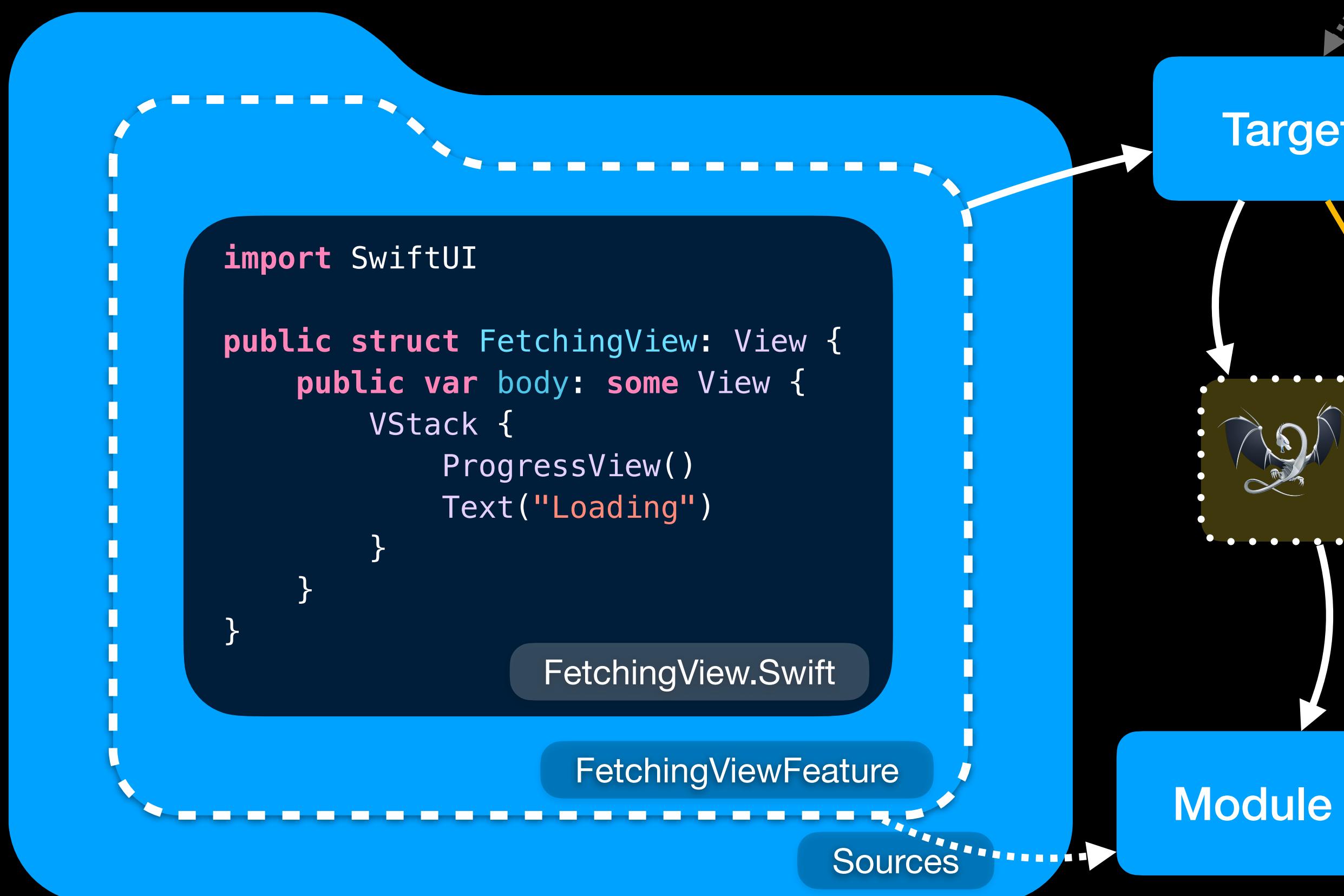
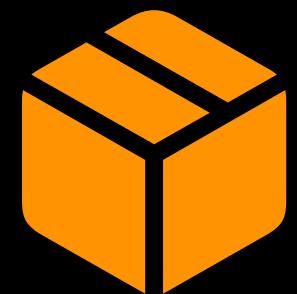




```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required by the package.

import PackageDescription

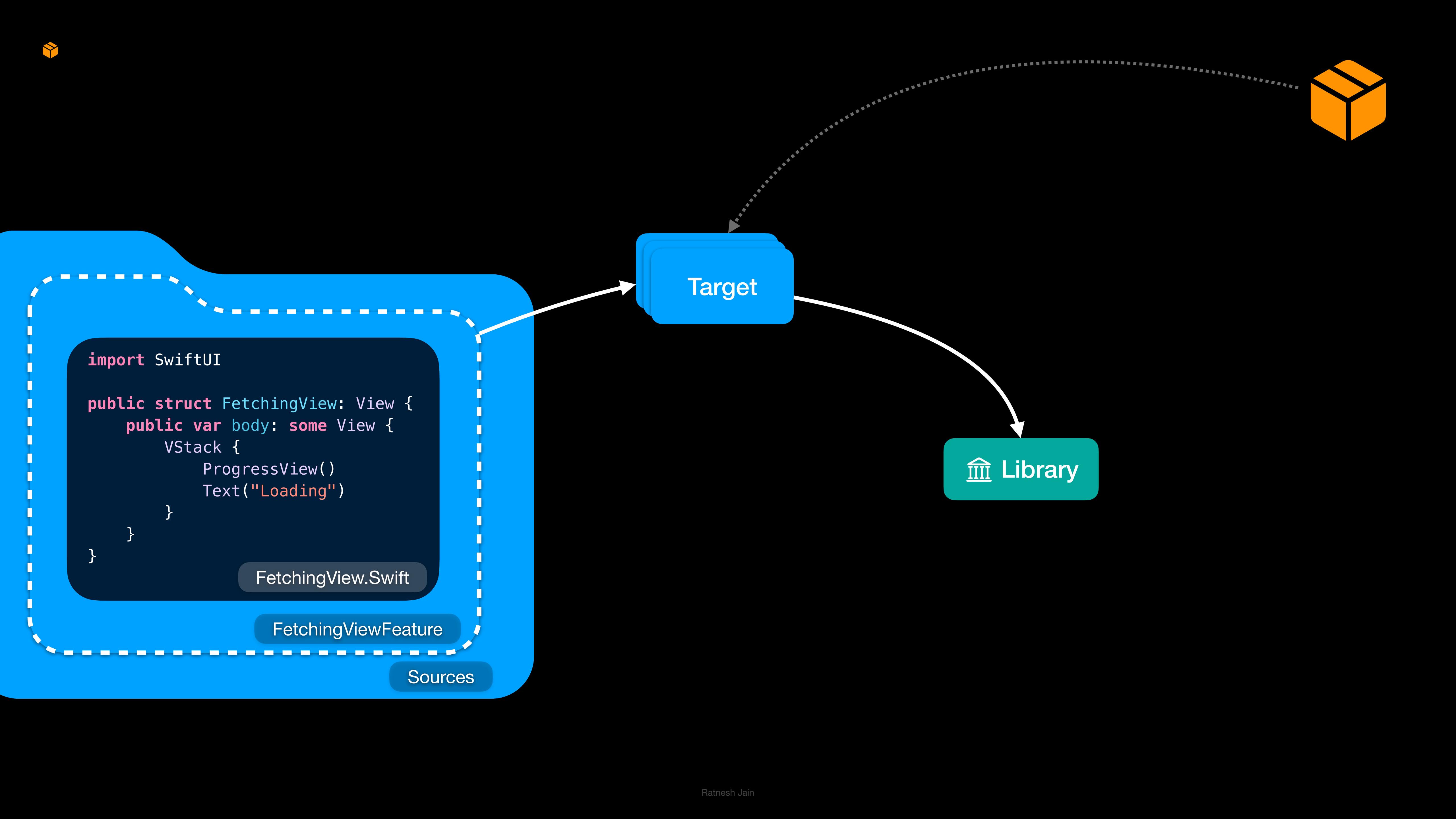
let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

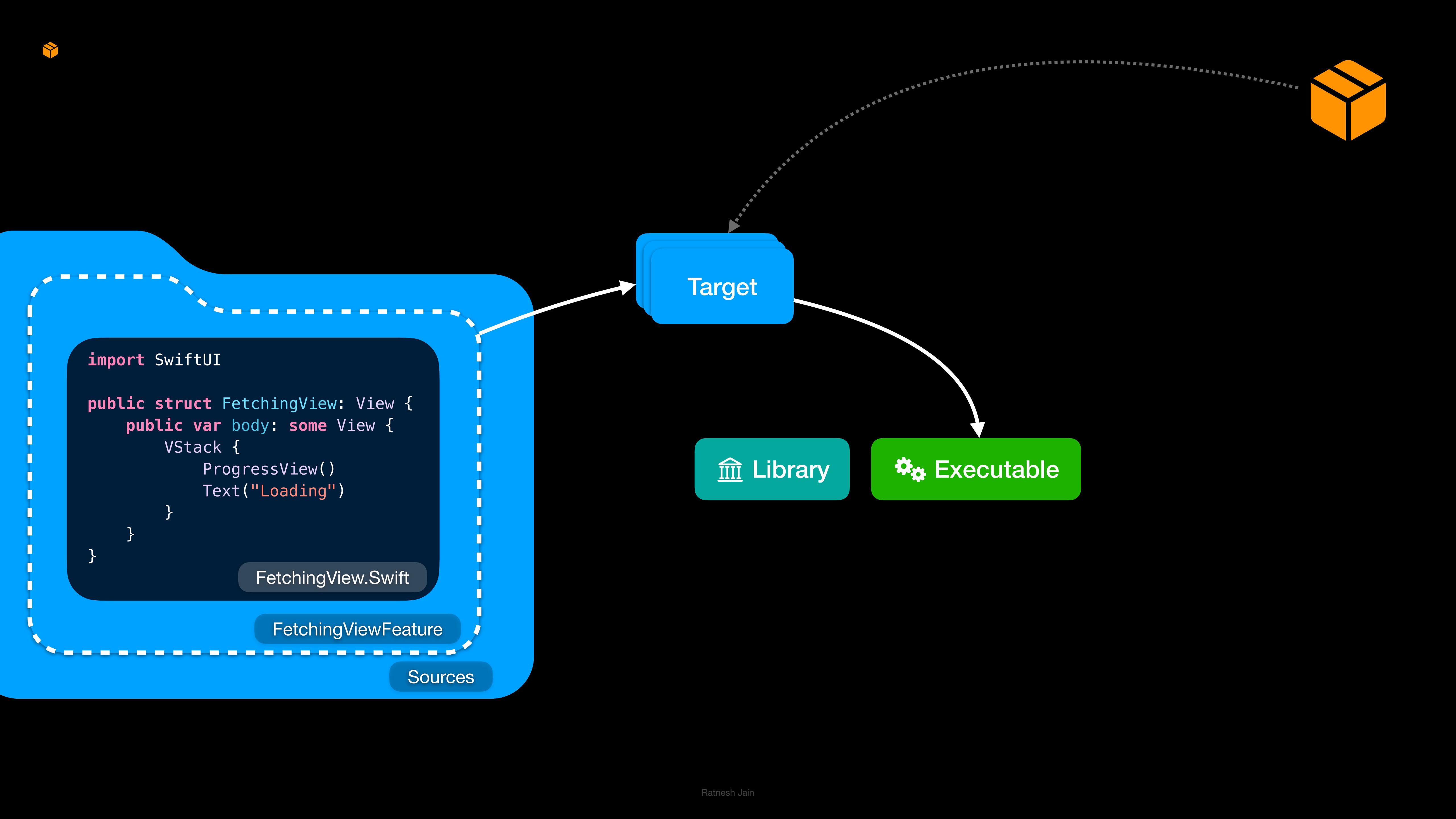


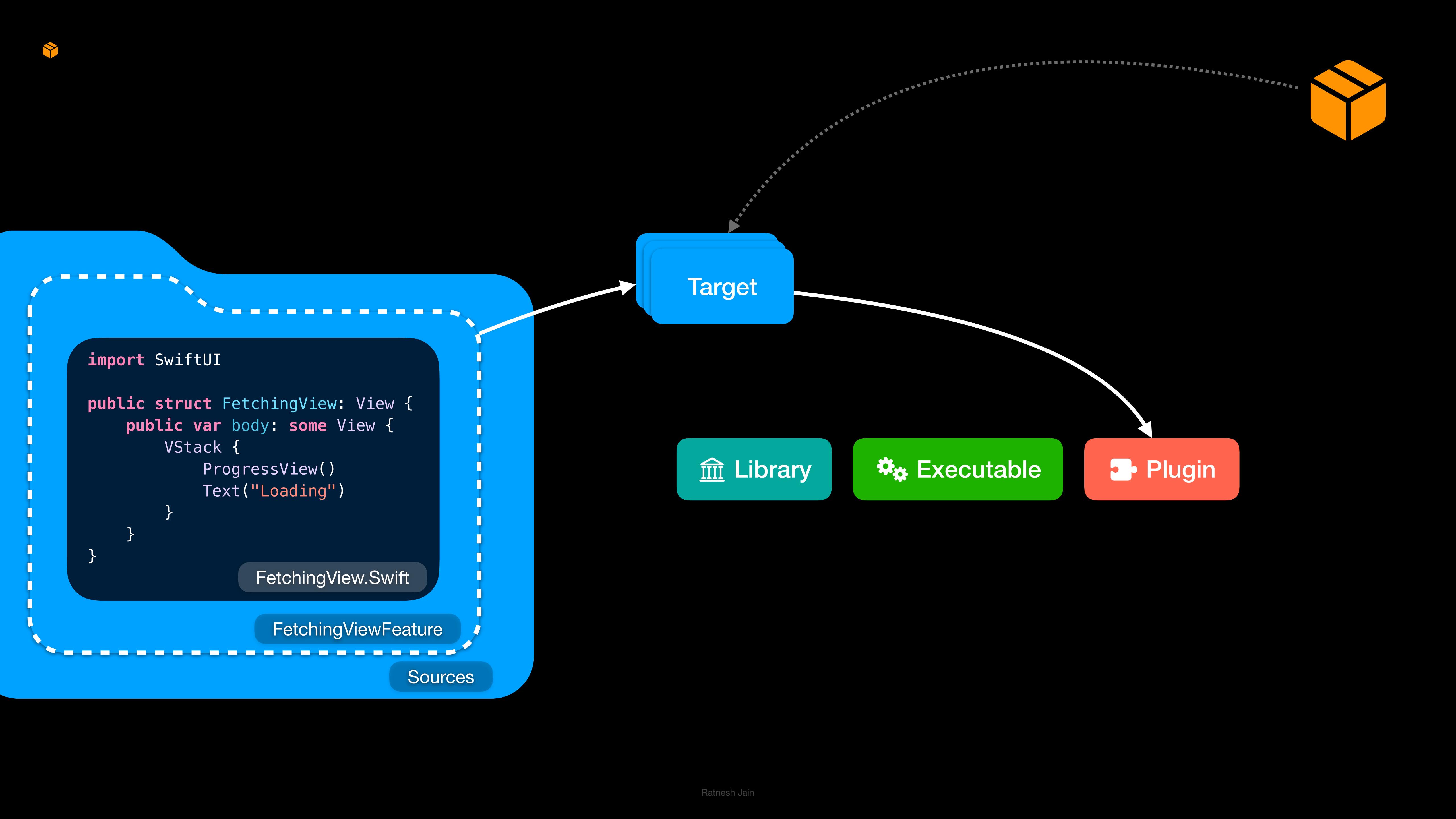
```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required by the package.

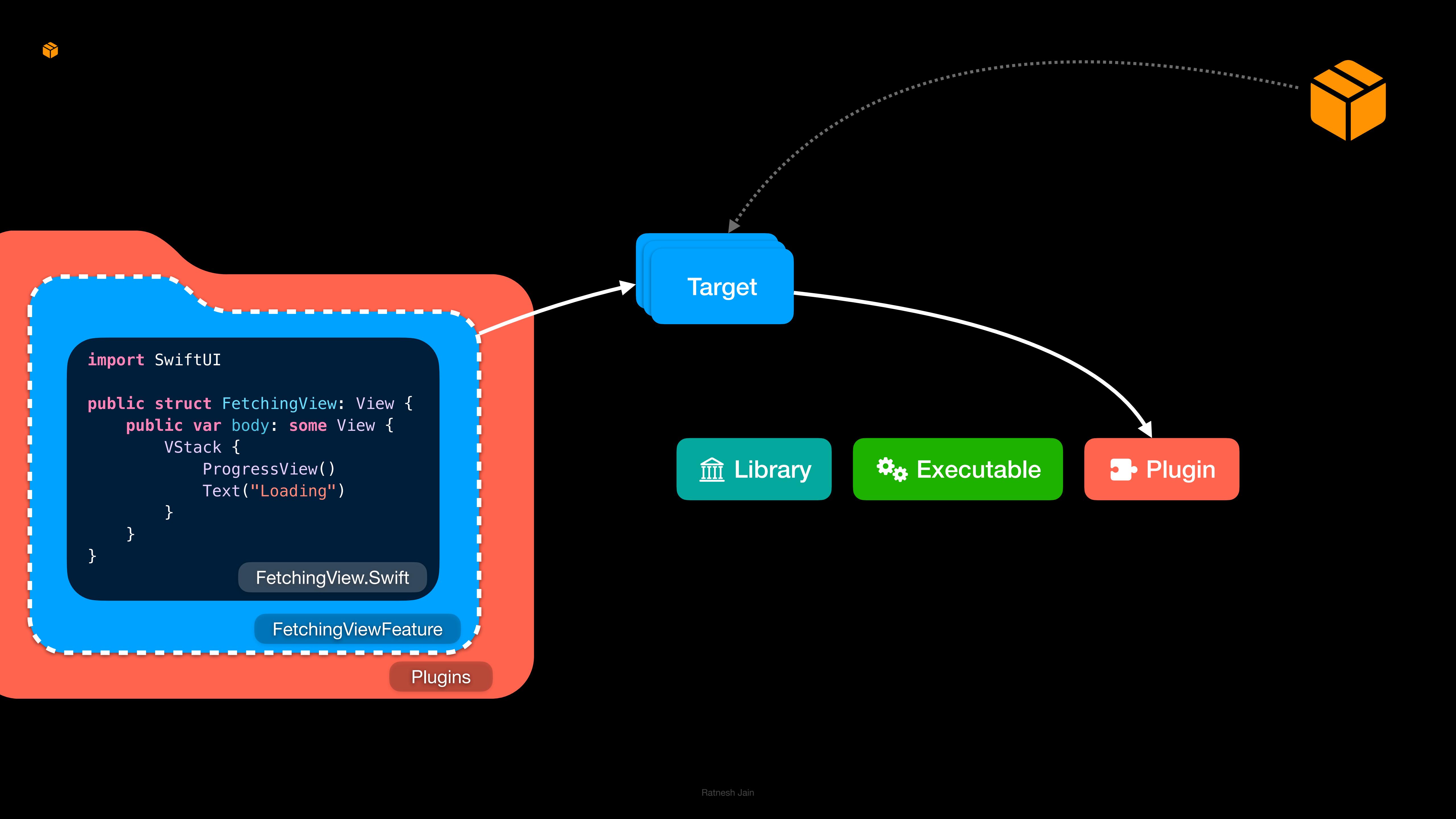
import PackageDescription

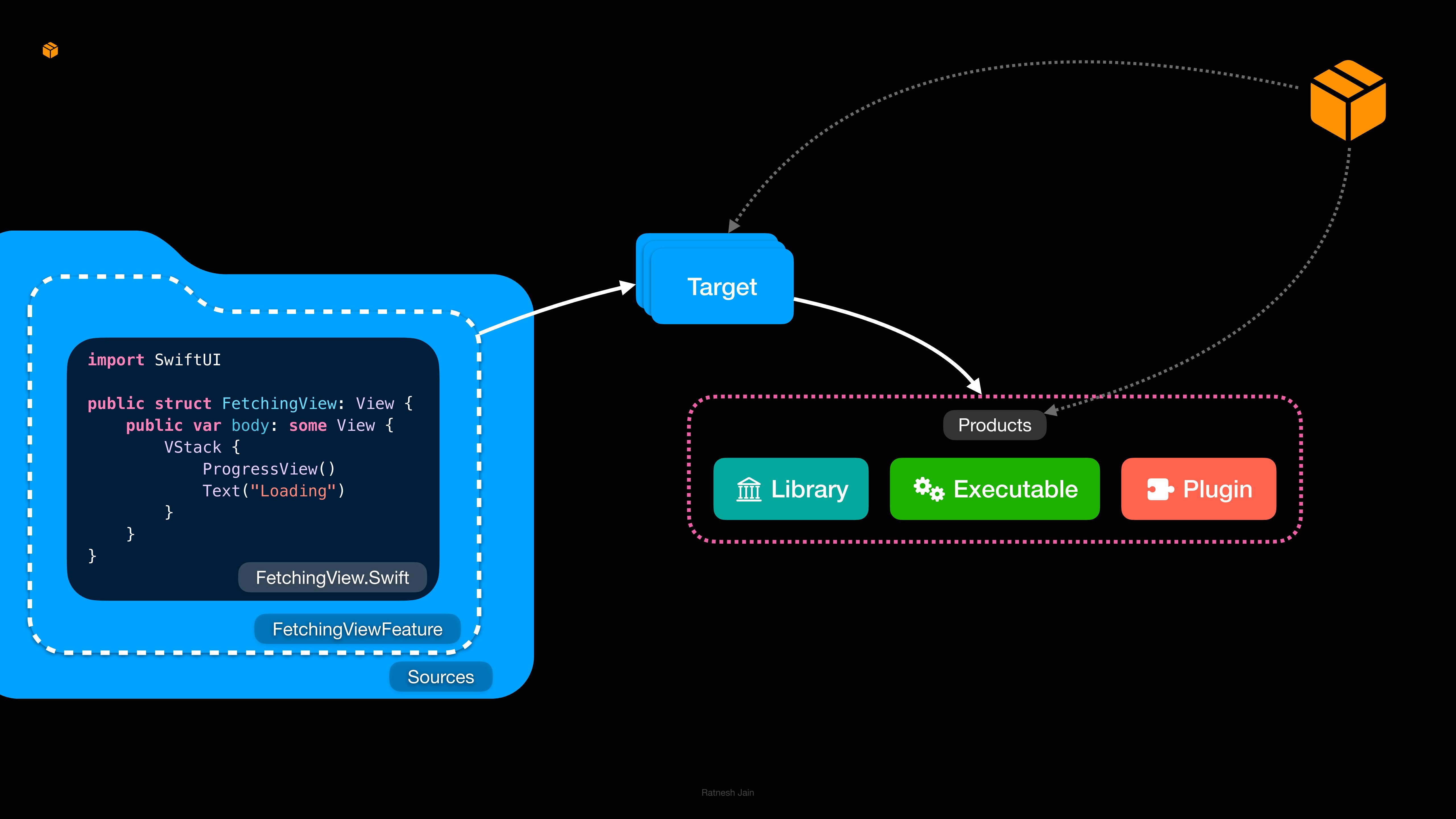
let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
        ],
    targets: [
        .target(
            name: "FetchingViewFeature"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

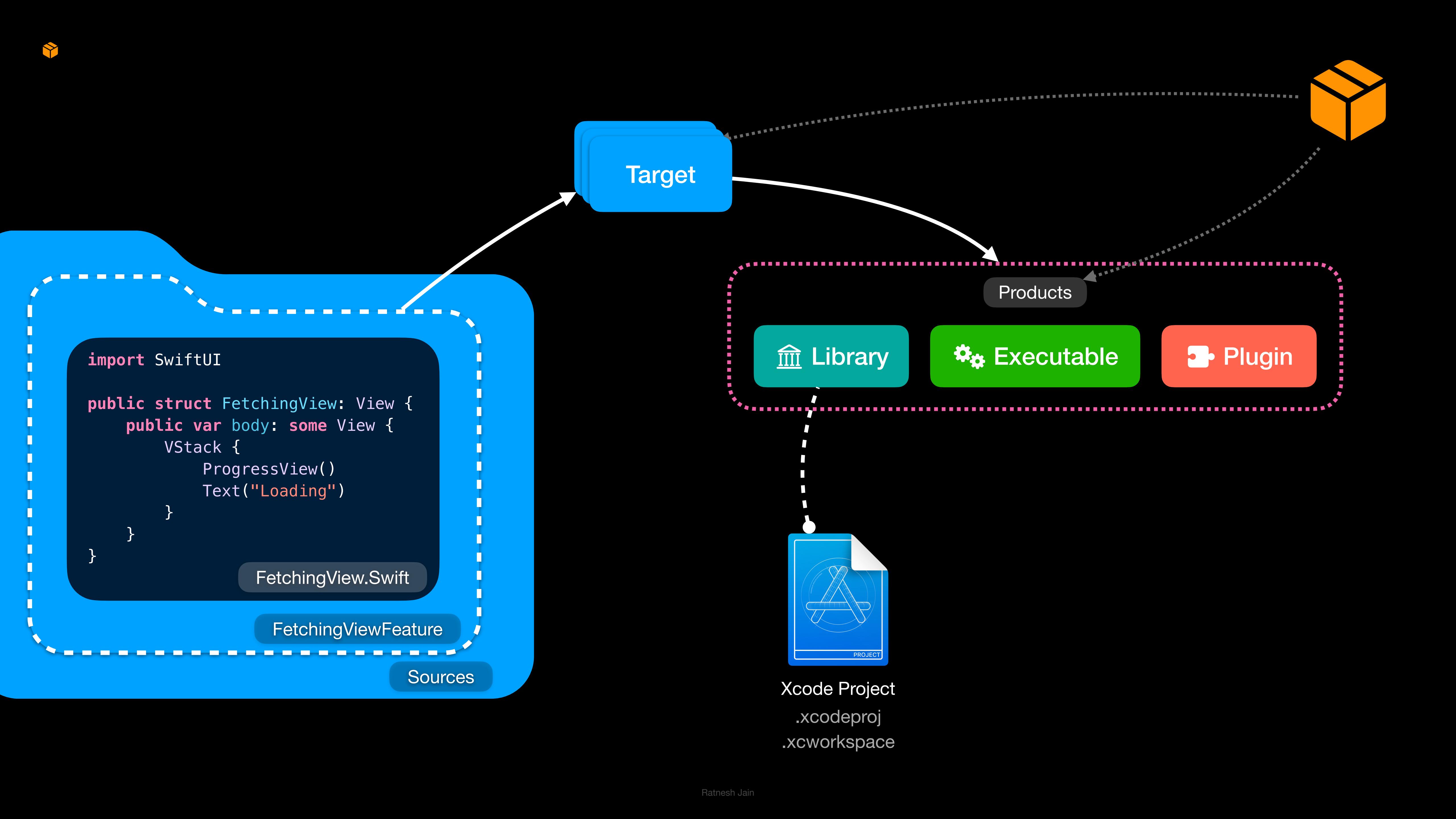


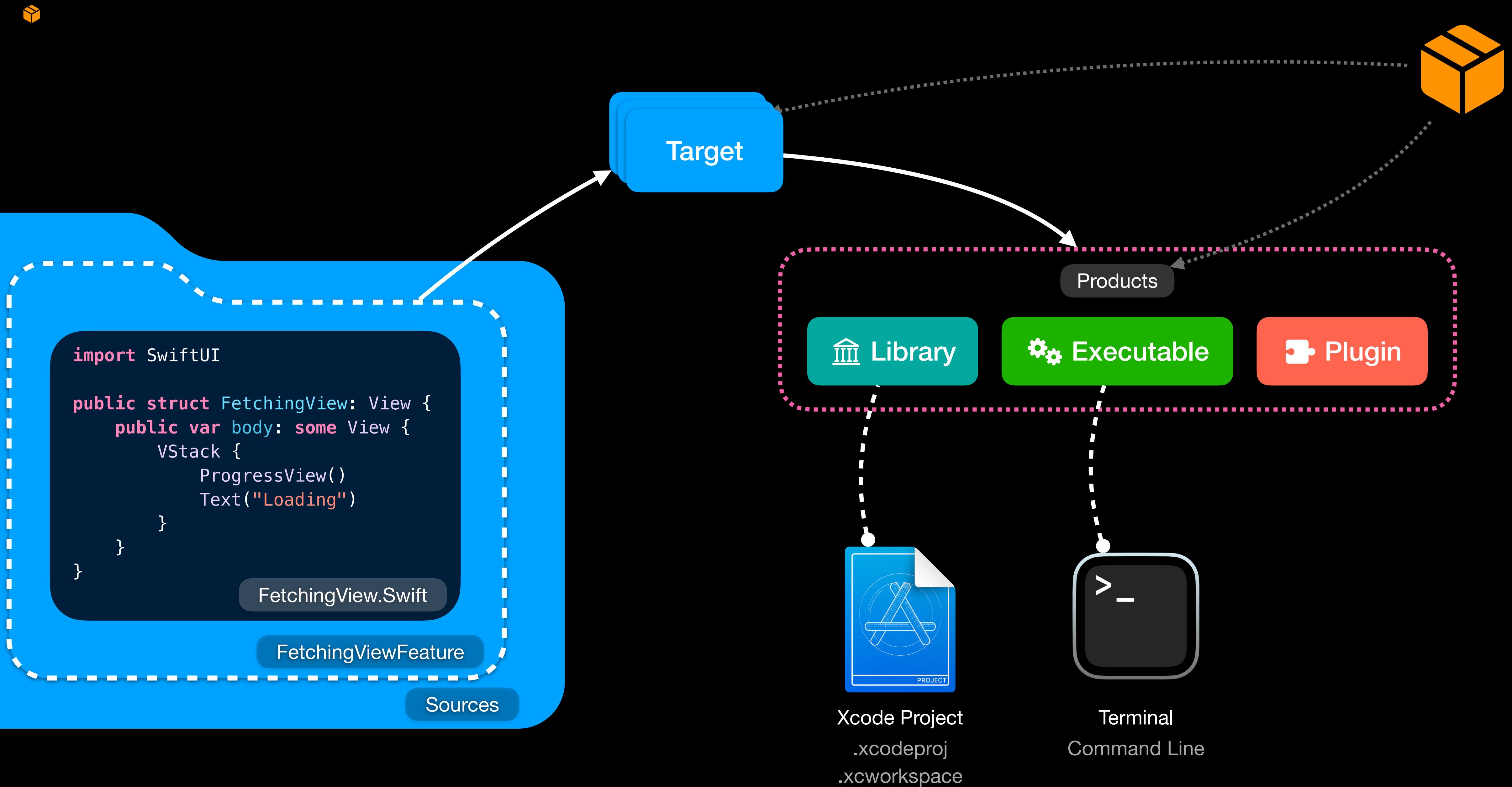


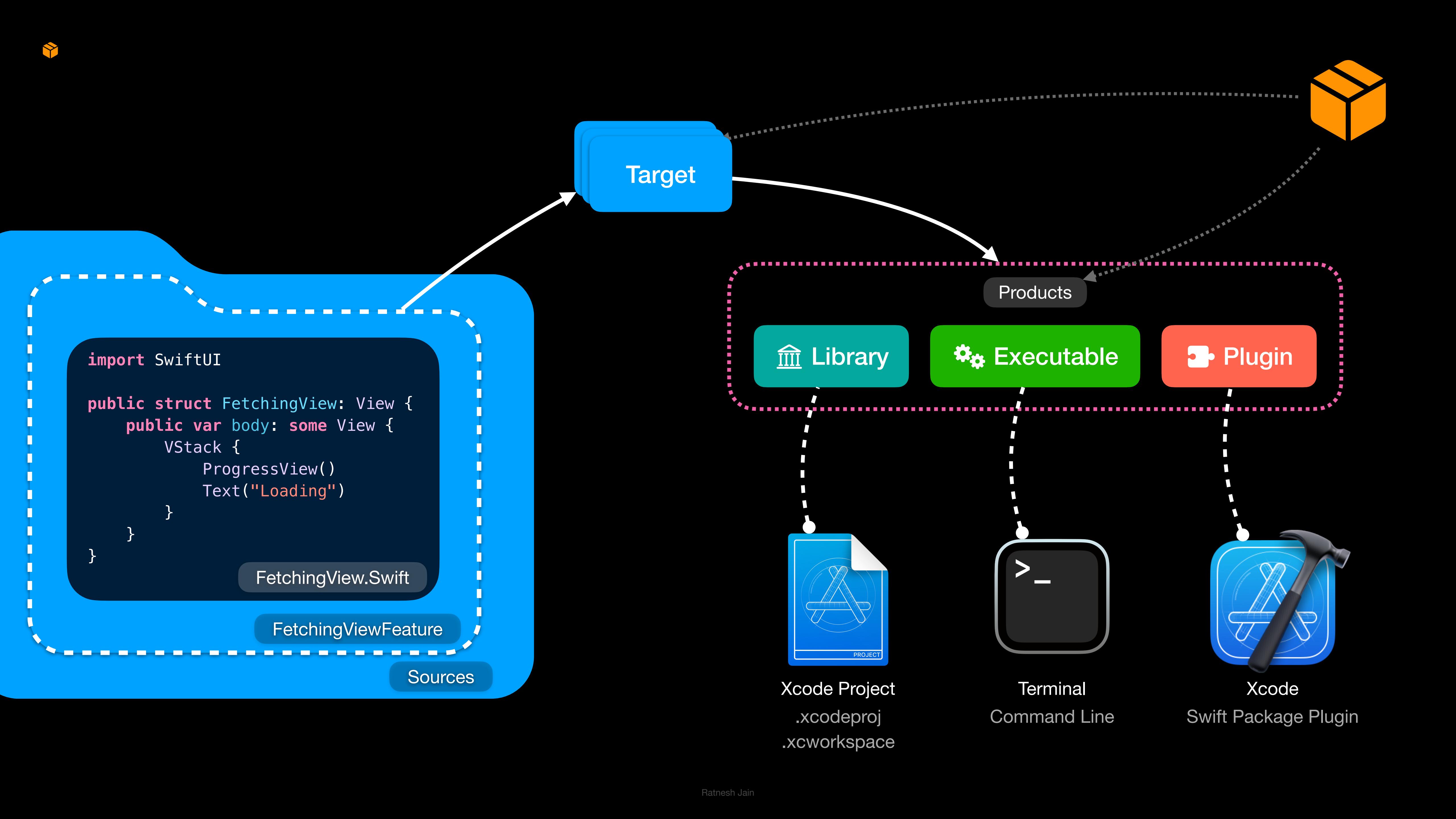


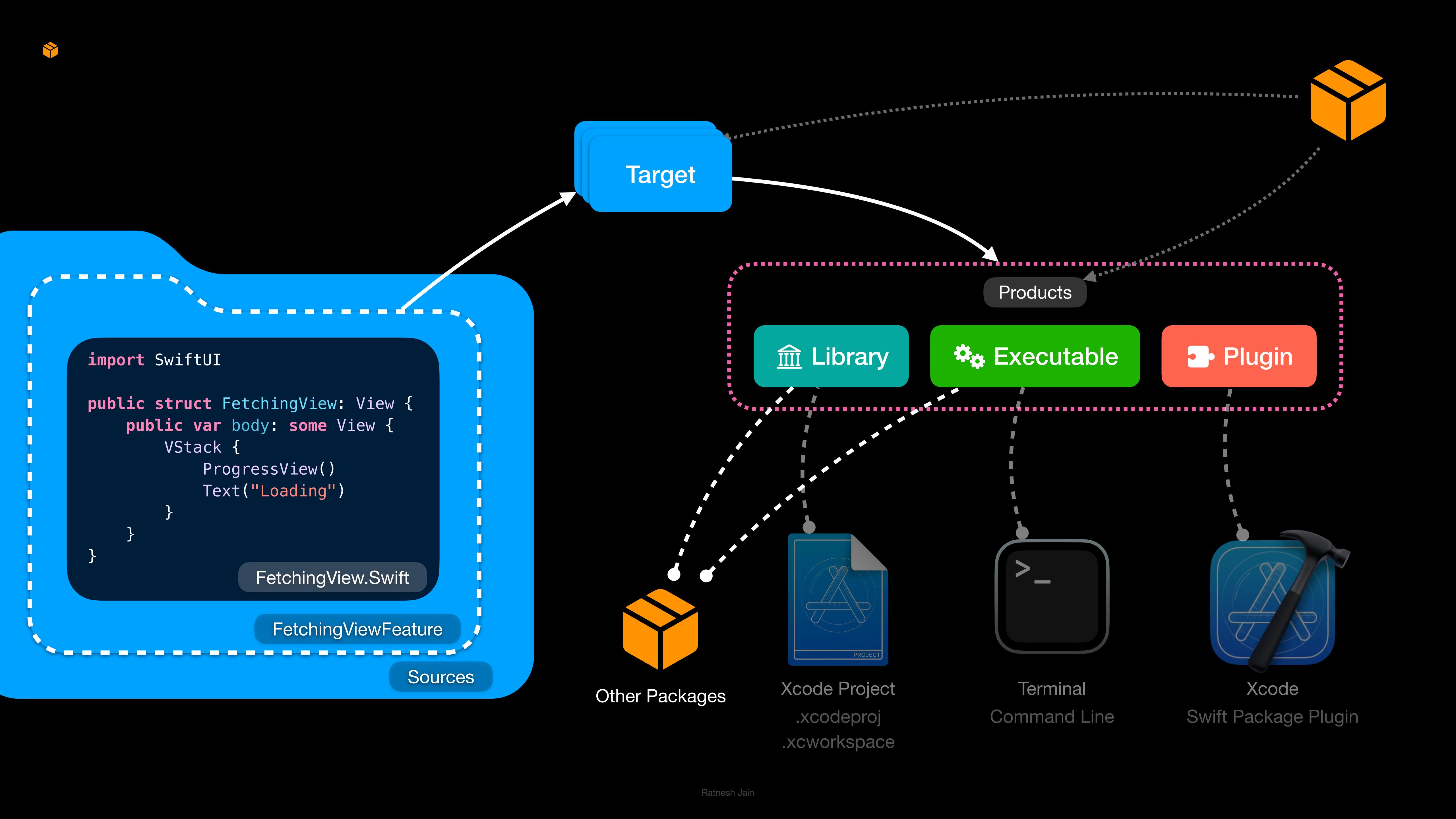














Mastering SwiftPM

Today

- Access Modifiers
- Package Manifest
- Package Manager Vocabulary
- Dependencies
- New Feature using SwiftPM
- Integration with Xcode Project
- Sharing Resources
- Dealing with Non-SPM external Dependency



Dependencies

```
import SwiftUI

public struct FetchingView: View {
    public var body: some View {
        VStack {
            ProgressView()
            Text("Loading")
        }
    }
}
```



Dependencies

```
import SwiftUI
import Alamofire

public struct FetchingView: View {
    public var body: some View {
        VStack {
            ProgressView()
            Text("Loading")
        }
    }
}
```



Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0"))
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v12)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0"))
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

Where to get the VALID
Dependency URL and Version
number?

Private < > github.com

Product Solutions Resources Open Source Enterprise Pricing Sign in Sign up

Alamofire / Alamofire Public

<> Code Issues 39 Pull requests 9 Discussions Actions Projects Security Insights

master ▾ 22 Branches 100 Tags <> Code ▾

Author	Commit Message	Date
lamtrinhdev	Update links for repositories moved to the swiftlang.org on GitHub	9b5bd5b · last month
.github	Bug: Pass queue to responseStreamDecodable (#3879)	2 months ago
.swiftpm/xcode/package.xcworkspace	Add More OSes to Testing Matrix (#3545)	2 years ago
Alamofire.xcodeproj	Prepare 5.9.1 (#3857)	6 months ago
Alamofire.xcworkspace	Fix watchOS Memory Leaks (#3110)	4 years ago
Documentation	Fix a trivial typo in AdvancedUsage.md (#3791)	10 months ago
Example	Refactor Project Structure, Break Request Types Out Into ...	8 months ago
Resources	Add watchOS Testing (#3449)	3 years ago
Source	Bug: Pass queue to responseStreamDecodable (#3879)	2 months ago
Tests	Bug: Pass queue to responseStreamDecodable (#3879)	2 months ago
docs	Update links for repositories moved to the swiftlang.org on GitHub	last month
watchOS Example	Refactor Project Structure, Break Request Types Out Into ...	8 months ago
.dockerignore	Linux and Windows Support (#3446)	3 years ago
.gitignore	Alamofire 5 Migration Guide (#3061)	4 years ago
.jazzy.yaml	Fix dash download (#2258x2)	7 years ago
.ruby-gemset	[PR #2250] Added Jazzy docs for the release to work with	7 years ago
.ruby-version	Add CocoaPods PrivacyInfo Integration (#3839)	7 months ago
.spi.yml	Refactor Project Structure, Break Request Types Out Into ...	8 months ago

About

Elegant HTTP Networking in Swift

swift networking xcode cocoapods
carthage swift-package-manager
alamofire request response
urlsession certificate-pinning
public-key-pinning urlrequest
httpurlresponse parameter-encoding

Readme MIT license Activity Custom properties 41k stars 1.1k watching 7.5k forks Report repository

Releases 99

5.9.1 Latest on Mar 31 + 98 releases

Sponsor this project

jshier Jon Shier
 Alamofire **Alamofire**

Private < >  github.com  ⌂

5 of 6 matches Begins with < > Done

 README  MIT license

- [Alamofire 4.0 Migration Guide](#)
- [Alamofire 3.0 Migration Guide](#)
- [Alamofire 2.0 Migration Guide](#)

Communication

- If you **need help with making network requests** using Alamofire, use [Stack Overflow](#) and tag `alamofire`.
- If you need to **find or understand an API**, check [our documentation](#) or [Apple's documentation for URLSession](#), on top of which Alamofire is built.
- If you need **help with an Alamofire feature**, use [our forum on swift.org](#).
- If you'd like to **discuss Alamofire best practices**, use [our forum on swift.org](#).
- If you'd like to **discuss a feature request**, use [our forum on swift.org](#).
- If you **found a bug**, open an issue here on GitHub and follow the guide. The more detail the better!

Installation

Swift Package Manager

The [Swift Package Manager](#) is a tool for automating the distribution of Swift code and is integrated into the `swift` compiler.

Once you have your Swift package set up, adding Alamofire as a dependency is as easy as adding it to the `dependencies` value of your `Package.swift` or the Package list in Xcode.

```
dependencies: [
    .package(url: "https://github.com/Alamofire/Alamofire.git", .upToNextMajor(from: "5.9.1")
]
```

Normally you'll want to depend on the `Alamofire` target:

```
.product(name: "Alamofire", package: "Alamofire")
```

But if you want to force Alamofire to be dynamically linked (do not do this unless you're sure you need it), you can depend on the `AlamofireDynamic` target:

```
.product(name: "AlamofireDynamic", package: "Alamofire")
```

Installation

Swift Package Manager

The [Swift Package Manager](#) is a tool for automating the distribution of Swift code and is integrated into the `swift` compiler.

Once you have your Swift package set up, adding Alamofire as a dependency is as easy as adding it to the `dependencies` value of your `Package.swift` or the Package list in Xcode.

```
dependencies: [
    .package(url: "https://github.com/Alamofire/Alamofire.git", .upToNextMajor(from: "5.9.1")
]
```

Normally you'll want to depend on the `Alamofire` target:

```
.product(name: "Alamofire", package: "Alamofire")
```

But if you want to force Alamofire to be dynamically linked (do not do this unless you're sure you need it), you can depend on the `AlamofireDynamic` target:

```
.product(name: "AlamofireDynamic", package: "Alamofire")
```

Installation

Swift Package Manager

The [Swift Package Manager](#) is a tool for automating the distribution of Swift code and is integrated into the `swift` compiler.

Once you have your Swift package set up, adding Alamofire as a dependency is as easy as adding it to the `dependencies` value of your `Package.swift` or the Package list in Xcode.

```
dependencies: [
    .package(url: "https://github.com/Alamofire/Alamofire.git", .upToNextMajor(from: "5.9.1"))
]
```



Normally you'll want to depend on the `Alamofire` target:

```
.product(name: "Alamofire", package: "Alamofire")
```



But if you want to force Alamofire to be dynamically linked (do not do this unless you're sure you need it), you can depend on the `AlamofireDynamic` target:

```
.product(name: "AlamofireDynamic", package: "Alamofire")
```



Private < >  github.com  

5 of 6 matches Begins with   package Manager  < > Done

README MIT license

- [Alamofire 4.0 Migration Guide](#)
- [Alamofire 3.0 Migration Guide](#)
- [Alamofire 2.0 Migration Guide](#)

Communication

- If you **need help with making network requests** using Alamofire, use [Stack Overflow](#) and tag `alamofire`.
- If you need to **find or understand an API**, check [our documentation](#) or [Apple's documentation for URLSession](#), on top of which Alamofire is built.
- If you need **help with an Alamofire feature**, use [our forum on swift.org](#).
- If you'd like to **discuss Alamofire best practices**, use [our forum on swift.org](#).
- If you'd like to **discuss a feature request**, use [our forum on swift.org](#).
- If you **found a bug**, open an issue here on GitHub and follow the guide. The more detail the better!

Installation

Swift Package Manager

The [Swift Package Manager](#) is a tool for automating the distribution of Swift code and is integrated into the `swift` compiler.

Once you have your Swift package set up, adding Alamofire as a dependency is as easy as adding it to the `dependencies` value of your `Package.swift` or the Package list in Xcode.

```
dependencies: [
    .package(url: "https://github.com/Alamofire/Alamofire.git", .upToNextMajor(from: "5.9.1"))
]
```

Normally you'll want to depend on the `Alamofire` target:

```
.product(name: "Alamofire", package: "Alamofire")
```

But if you want to force Alamofire to be dynamically linked (do not do this unless you're sure you need it), you can depend on the `AlamofireDynamic` target:

```
.product(name: "AlamofireDynamic", package: "Alamofire")
```

Private < >  github.com  ⌂

5 of 6 matches Begins with 🔍 package Manager  < > Done

 README  MIT license

- [Alamofire 4.0 Migration Guide](#)
- [Alamofire 3.0 Migration Guide](#)
- [Alamofire 2.0 Migration Guide](#)

Communication

- If you **need help with making network requests** using Alamofire, use [Stack Overflow](#) and tag `alamofire`.
- If you need to **find or understand an API**, check [our documentation](#) or [Apple's documentation for URLSession](#), on top of which Alamofire is built.
- If you need **help with an Alamofire feature**, use [our forum on swift.org](#).
- If you'd like to **discuss Alamofire best practices**, use [our forum on swift.org](#).
- If you'd like to **discuss a feature request**, use [our forum on swift.org](#).
- If you **found a bug**, open an issue here on GitHub and follow the guide. The more detail the better!

Installation

Swift Package Manager

The [Swift Package Manager](#) is a tool for automating the distribution of Swift code and is integrated into the `swift` compiler.

Once you have your Swift package set up, adding Alamofire as a dependency is as easy as adding it to the `dependencies` value of your `Package.swift` or the Package list in Xcode.

```
dependencies: [
    .package(url: "https://github.com/Alamofire/Alamofire.git", .upToNextMajor(from: "5.9.1")
]
```

Normally you'll want to depend on the `Alamofire` target:

```
.product(name: "Alamofire", package: "Alamofire")
```

But if you want to force Alamofire to be dynamically linked (do not do this unless you're sure you need it), you can depend on the `AlamofireDynamic` target:

```
.product(name: "AlamofireDynamic", package: "Alamofire")
```

Private < > github.com  

 Product Solutions Resources Open Source Enterprise Pricing / Sign in Sign up

 [ratnesh-jain / AssetPluginLibrary](#) Public  Notifications  Fork 0  Star 1

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)

 main   1 Branch  5 Tags [Code](#) ▾

 ratnesh-jain Update README.md 545c11f · last week  6 Commits

 .swiftpm/xcode	Initial commit	10 months ago
 Plugins/AssetResourcePlugin	Add MainActor annotation.	3 weeks ago
 Tests/AssetLibraryTests	Initial commit	10 months ago
 .gitignore	Initial commit	10 months ago
 Package.swift	Initial commit	10 months ago
 README.md	Update README.md	last week

 README

AssetLibrary

For usage details please visit this [story](#).

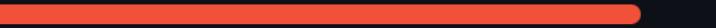
 © 2024 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)

About
No description, website, or topics provided.

 Readme
 Activity
 1 star
 1 watching
 0 forks
[Report repository](#)

Releases
 5 tags

Packages
No packages published

Languages
  Swift 100.0%

Private < >  github.com  

Product Solutions Resources Open Source Enterprise Pricing

Search or jump to... / Sign in Sign up

ratnesh-jain / AssetPluginLibrary Public

Code Issues Pull requests Actions Projects Security Insights

main 1 Branch 5 Tags Go to file Code About

No description, website, or topics provided.

ratnesh-jain Update README.md 545c11f · last week 6 Commits

.swiftlint.yml .gitignore Package.swift Readme.md Tests/AssetLibraryTests

Initial commit 10 months ago Initial commit 10 months ago Initial commit 10 months ago

5 tags Packages Languages

1 watching 0 forks Report repository Releases

Some Doesn't have swift package usage details in the README file

AssetLibrary For usage details please visit this story.

© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information



Plugins/AssetResourcePlugin

Add



Tests/AssetLibraryTests

Initia



.gitignore

Initia



Package.swift

Initia



README.md

Upda



README



Plugins/AssetResourcePlugin

Add



Tests/AssetLibraryTests

Initia



.gitignore

Initia



Package.swift

Initia



README.md

Upda



README

Private < > github.com  

 Product Solutions Resources Open Source Enterprise Pricing / Sign in Sign up

 [ratnesh-jain / AssetPluginLibrary](#) Public  Notifications  Fork 0  Star 1

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)

 main   1 Branch  5 Tags 

ratnesh-jain	Update README.md	545c11f · last week	6 Commits
.swiftpm/xcode	Initial commit	10 months ago	
Plugins/AssetResourcePlugin	Add MainActor annotation.	3 weeks ago	
Tests/AssetLibraryTests	Initial commit	10 months ago	
.gitignore	Initial commit	10 months ago	
Package.swift	Initial commit	10 months ago	
README.md	Update README.md	last week	

 README

AssetLibrary

For usage details please visit this [story](#).

 © 2024 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)

About
No description, website, or topics provided.

 Readme
 Activity
 1 star
 1 watching
 0 forks
[Report repository](#)

Releases
 5 tags

Packages
No packages published

Languages
  Swift 100.0%

Activity

☆ 1 star

👁 1 watching

🍴 0 forks

[Report repository](#)

Releases

🏷 5 tags

Packages

No packages published

Languages

Private < > github.com [Sign in](#) [Sign up](#)

ratnesh-jain / AssetPluginLibrary

Code Issues Pull requests Actions Projects Security Insights

main · 1 Branch · 5 Tags

ratnesh-jain Update README.md · last week · 6 Commits

.swiftpm/xcode	Initial commit	10 months ago
Plugins/AssetResourcePlugin	Add MainActor annotation.	3 weeks ago
Tests/AssetLibraryTests	Initial commit	10 months ago
.gitignore	Initial commit	10 months ago
Package.swift	Initial commit	10 months ago
README.md	Update README.md	last week

README

AssetLibrary

For usage details please visit this [story](#).

Go to file · Code

Notifications Fork 0 Star 1

About

No description, website, or topics provided.

Readme · Activity · 1 star · 1 watching · 0 forks · Report repository

Releases

5 tags

Packages

No packages published

Languages

Swift 100.0%

© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

Private < >

github.com

Product Solutions Resources Open Source Enterprise Pricing

Search or jump to... / Sign in Sign up

ratnesh-jain / AssetPluginLibrary Public Notifications Fork 0 Star 1

Code Issues Pull requests Actions Projects Security Insights

Releases Tags

Tags

0.0.5 · 3 weeks ago · 9351d10 · zip tar.gz

0.0.4 · on Jun 8 · d724beb · zip tar.gz

0.0.3 · on Jun 7 · d43ccbd · zip tar.gz

0.0.2 · on Jan 15 · 2299c13 · zip tar.gz

0.0.1 · on Nov 16, 2023 · 1d779c1 · zip tar.gz

© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

[Releases](#)[Tags](#)

Tags

0.0.5 [...](#)

 3 weeks ago  9351d10  zip  tar.gz

0.0.4 [...](#)

 on Jun 8  d724beb  zip  tar.gz

0.0.3 [...](#)

 on Jun 7  d43ccbd  zip  tar.gz

0.0.2 [...](#)

 on Jan 15  2299c13  zip  tar.gz

0.0.1 [...](#)

 on Nov 16, 2023  1d779c1  zip  tar.gz

Private < >

github.com

Product Solutions Resources Open Source Enterprise Pricing

Search or jump to... / Sign in Sign up

ratnesh-jain / AssetPluginLibrary Public Notifications Fork 0 Star 1

Code Issues Pull requests Actions Projects Security Insights

Releases Tags

Tags

0.0.5 ··· 3 weeks ago -O- 9351d10 zip tar.gz

0.0.4 ··· on Jun 8 -O- d724beb zip tar.gz

0.0.3 ··· on Jun 7 -O- d43ccbd zip tar.gz

0.0.2 ··· on Jan 15 -O- 2299c13 zip tar.gz

0.0.1 ··· on Nov 16, 2023 -O- 1d779c1 zip tar.gz

© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

Private < >

https://github.com/ratnesh-jain/AssetPluginLibrary

Favourites

Show All ▾

Search or jump to... / Sign in Sign up

Notifications Fork 0 Star 1

Apple SVG Shaper fo... iCloud Google Twitter Wikipedia

Yahoo! News Popular OS X Maverick... 219_s-d_whats... Rambo.- codes | G...

Frequently Visited

LinkedIn SwiftUI Basics an... Google Meet – ptt-nxji-jqj Others > Content... Bag – Apple (IN)

TopHire App Store Connect groue/ GRDB.sw... Swift Talk – objc.io objc.io Work On Projects...

Privacy Report

AssetLibrary

For usage details please visit this [story](#).

Packages

No packages published

Languages

Swift 100.0%

© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information



Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
```



<https://github.com/ratnesh-jain/AssetPluginLibrary>

The screenshot shows the Xcode Asset Catalog Editor interface. On the left, there's a sidebar with 'Source' and 'Enterprise' dropdowns, and 'Projects' and a search bar. The main area is titled 'Favourites' and contains several icons:

- An Apple logo icon labeled 'Apple'.
- A square icon containing the letters 'zz' with a beach scene at the bottom, labeled 'SVG Shaper fo...'.
A note below it says: 'This asset is a placeholder for a vector-based image. It is recommended to use the SVG Shaper tool to generate a vector-based image from a raster image.'
- An Apple logo icon labeled 'iCloud'.
- A Google logo icon labeled 'Google'.
- A Twitter logo icon labeled 'Twitter'.
- A Wikipedia logo icon labeled 'Wiki'.



Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/RatneshJain/AssetPluginLibrary",
            .upToNextMajor(from: "5.0.0"))
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

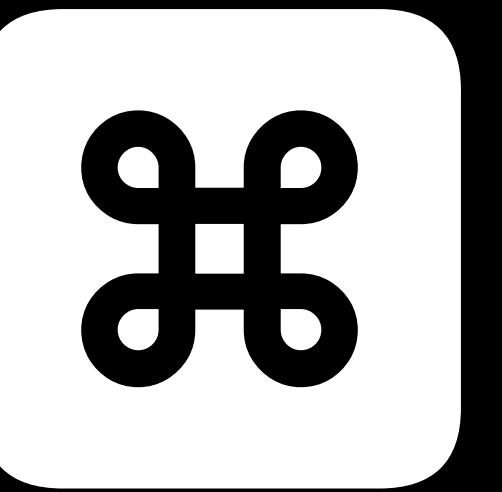


Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"))
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



Save

The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure with a selected "Package" item under "MasteringSwiftPM".
- Editor:** Displays the contents of the "Package.swift" file.
- Toolbar:** Includes standard Xcode icons for file operations like New, Open, Save, and Print.
- Top Bar:** Shows the project name "MasteringSwiftPM", the target device "iPhone 15 Pro (17.5)", and a status message "Reloading Package 'masteringswiftpm'".
- Bottom Bar:** Includes a "Filter" button, a small green icon, and status information "Line: 24 Col: 10".

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)

name: "MasteringSwiftPMTests",
dependencies: ["FetchingViewFeature"]),
]
```

Ratnesh Jain



Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(name: "MasteringSwiftPM", type: .dynamic)
    ],
    dependencies: [
        .package(url: "https://github.com/pointfreeco/swift-concurrency", from: "1.0.0")
    ]
)
```

MasteringSwiftPM > iPhone 15 Pro (17.5)

Reloading Package 'masteringswiftpm'



on

5.10

ion declares the minimum version of Swift required to build this pac

on

ftPM",

17)1.



Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(name: "MasteringSwiftPM", type: .dynamic)
    ],
    dependencies: [
        .package(url: "https://github.com/pointfreeco/swift-concurrency", from: "1.0.0")
    ]
)
```

MasteringSwiftPM > iPhone 15 Pro (17.5)

Reloading Package 'masteringswiftpm'



on

5.10

ion declares the minimum version of Swift required to build this pac

on

ftPM",

17)1.

The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure with a selected "Package" item.
- Editor:** Displays the `Package.swift` file content.
- Document Outline:** Shows the package structure: `MasteringSwiftPM > Package > No Selection`.
- Top Bar:** Shows the project name "MasteringSwiftPM", the target device "iPhone 15 Pro (17.5)", and a status message "Reloading Package 'masteringswiftpm'".
- Bottom Bar:** Includes a "Filter" button and status indicators for line 24 and column 10.

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"))
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)

name: "MasteringSwiftPMTests",
dependencies: ["FetchingViewFeature"]),
]
```

Ratnesh Jain

The screenshot shows the Xcode interface with the project 'MasteringSwiftPM' open. The left sidebar displays the project structure, including 'Sources' containing 'FetchingViewFeature' and 'Tests'. The main editor area shows the 'Package.swift' file with the following content:

```
1 // swift-tools-version: 5.10
2 // The swift-tools-version declares the minimum version of Swift required to build this package.
3
4 import PackageDescription
5
6 let package = Package(
7     name: "MasteringSwiftPM",
8     platforms: [.iOS(.v17)],
9     products: [
10         .library(
11             name: "FetchingViewFeature",
12             targets: ["FetchingViewFeature"]),
13     ],
14     dependencies: [
15         .package(
16             url: "https://github.com/Alamofire/Alamofire.git",
17             .upToNextMajor(from: "5.0.0")
18         ),
19         .package(
20             url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
21             .upToNextMajor(
22                 from: "0.0.5"
23             )
24         ),
25     ],
26     targets: [
27         .target(
28             name: "FetchingViewFeature"
29         ),
30         .testTarget(
31             name: "MasteringSwiftPMTests",
32             dependencies: ["FetchingViewFeature"]),
33     ]
34 )
35
36
37 name: "MasteringSwiftPMTests",
38 dependencies: ["FetchingViewFeature"]),
39 ]
```

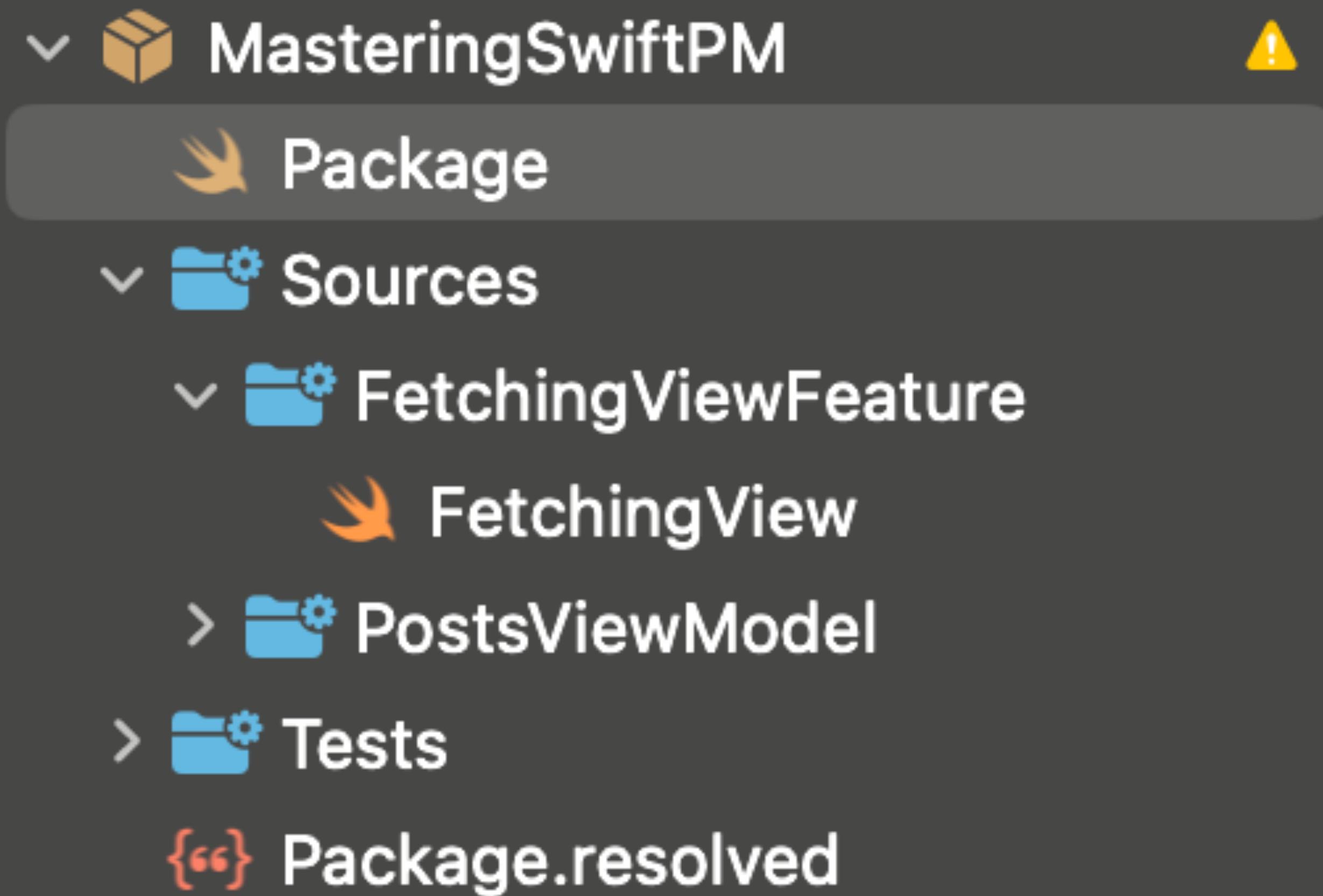
The status bar at the bottom right indicates 'Line: 24 Col: 10'. The top right corner shows 'Build Succeeded | 02/09/24 at 3:30 PM'.



Depen

```
// swift-tools-version: 5.3
// The swift-tools-version
// dependency identifier
import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v13)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: []
        )
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.9.1")
        ),
        .package(
            url: "https://github.com/AssetPluginLibrary/AssetPluginLibrary.git",
            .upToNextMajor(from: "0.0.5")
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"
        ),
        .testTarget(
            name: "FetchingViewFeatureTests"
        )
    ]
)
```



Package Dependencies

- > **Alamofire** 5.9.1
- > **AssetPluginLibrary** 0.0.5

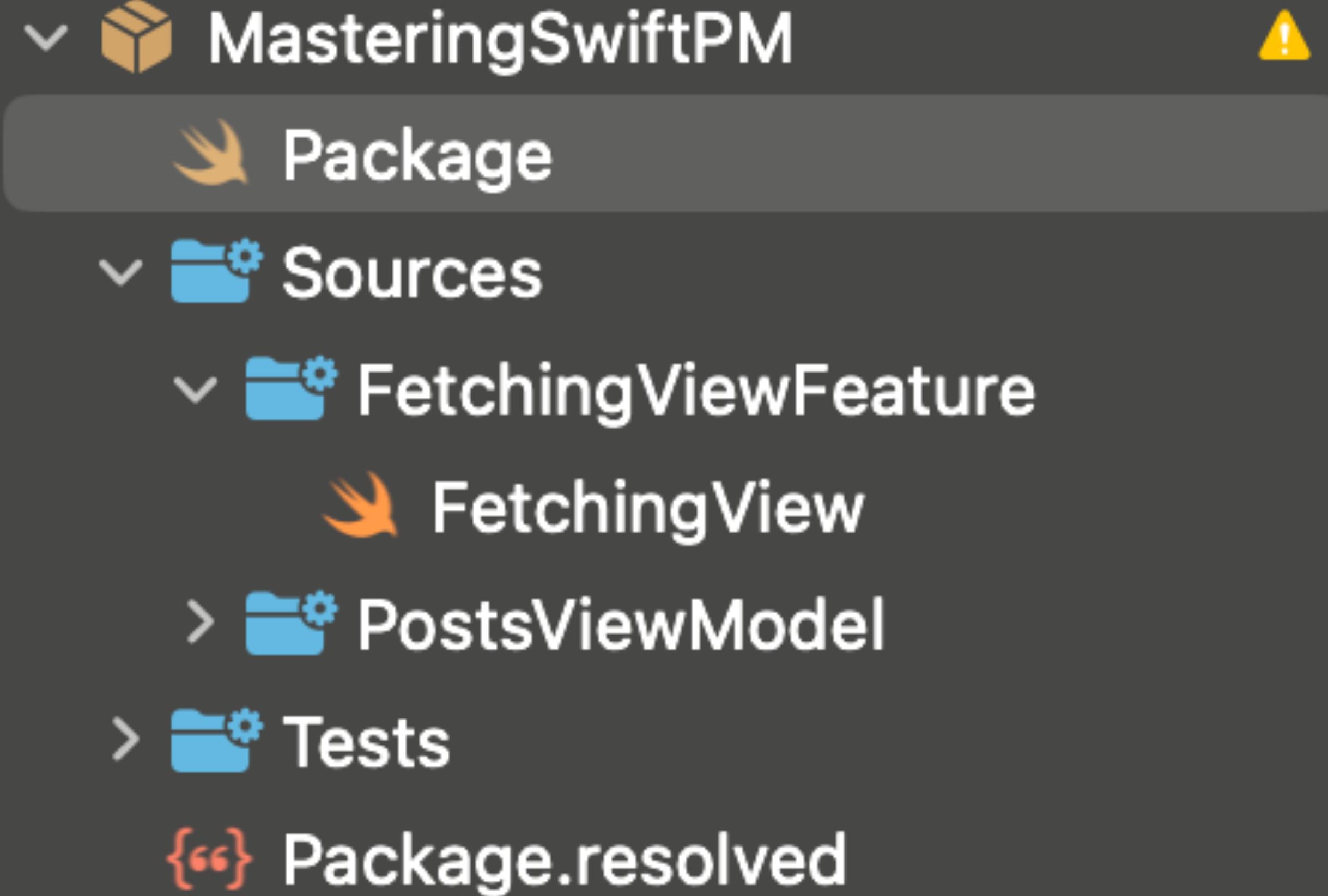
```
1 // swift-tools-version: 5.3
2 // The swift-tools-version
3
4 import PackageDescription
5
6 let package = Package(
7     name: "MasteringSwiftPM",
8     platforms: [.iOS(.v13)],
9     products: [
10         .library(
11             name: "FetchingViewFeature",
12             targets: []
13         )
14     ],
15     dependencies: [
16         .package(
17             url: "https://github.com/Alamofire/Alamofire.git",
18             .upToNextMajor(from: "5.9.1")
19         ),
20         .package(
21             url: "https://github.com/AssetPluginLibrary/AssetPluginLibrary.git",
22             .upToNextMajor(from: "0.0.5")
23         )
24     ],
25     targets: [
26         .target(
27             name: "FetchingViewFeature"
28         ),
29         .testTarget(
30             name: "FetchingViewFeatureTests"
31         )
32     ]
33 )
```



Depen

```
// swift-tools-version: 5.3
// The swift-tools-version
// dependency identifier
import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v13)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: []
        )
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.9.1")
        ),
        .package(
            url: "https://github.com/AssetPluginLibrary/AssetPluginLibrary.git",
            .upToNextMajor(from: "0.0.5")
        )
    ],
    targets: [
        .target(
            name: "FetchingView"
        ),
        .testTarget(
            name: "FetchingViewTests"
        )
    ]
)
```



```
1 // swift-tools-version: 5.3
2 // The swift-tools-version
3
4 import PackageDescription
5
6 let package = Package(
7     name: "MasteringSwiftPM",
8     platforms: [.iOS(.v13)],
9     products: [
10         .library(
11             name: "FetchingViewFeature",
12             targets: []
13         )
14     ],
15     dependencies: [
16         .package(
17             url: "https://github.com/Alamofire/Alamofire.git",
18             .upToNextMajor(from: "5.9.1")
19         ),
20         .package(
21             url: "https://github.com/AssetPluginLibrary/AssetPluginLibrary.git",
22             .upToNextMajor(from: "0.0.5")
23         )
24     ],
25     targets: [
26         .target(
27             name: "FetchingView"
28         ),
29         .testTarget(
30             name: "FetchingViewTests"
31         )
32     ]
33 )
```



Masterin



✓ MasteringSwiftPM



Package

✓ Sources

✓ FetchingViewFeature

FetchingView

> PostsViewModel

1 /

2 /

3

4

5

6

I



Masterin



▼ MasteringSwiftPM **Don't panic,**
Its Important message

▼ Sources **Just READ it first**

▼ FetchingViewFeature

FetchingView

> PostsViewModel



1

2

3

4

5

6

i



Masterin



✓ MasteringSwiftPM

Package **Otherwise**

✓ Sources

✓ FetchingViewFeature

FetchingView

> PostsViewModel



1 /
2 /
3 /
4 /
5 /
6 /

☰ | < > ❖ MasteringSwiftPM › Sources › FetchingViewFeature › FetchingView › No Selection

< ✖ > ⌂ ⌂ ⌂

```
1 // The Swift Programming Language
2 // https://docs.swift.org/swift-book
3
4 import SwiftUI
5 import Alamofire
6
7 public struct FetchingView: View {
8     public var body: some View {
9         VStack {
10             ProgressView()
11             Text("Loading")
12         }
13     }
14 }
15
```

ⓘ No such module 'Alamofire'

The screenshot shows the Xcode interface with the project 'MasteringSwiftPM' open. The left sidebar displays the project structure, including 'Sources' containing 'FetchingViewFeature' and 'Tests'. The main editor area shows the 'Package.swift' file with the following content:

```
1 // swift-tools-version: 5.10
2 // The swift-tools-version declares the minimum version of Swift required to build this package.
3
4 import PackageDescription
5
6 let package = Package(
7     name: "MasteringSwiftPM",
8     platforms: [.iOS(.v17)],
9     products: [
10         .library(
11             name: "FetchingViewFeature",
12             targets: ["FetchingViewFeature"]),
13     ],
14     dependencies: [
15         .package(
16             url: "https://github.com/Alamofire/Alamofire.git",
17             .upToNextMajor(from: "5.0.0")
18         ),
19         .package(
20             url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
21             .upToNextMajor(
22                 from: "0.0.5"
23             )
24         ),
25     ],
26     targets: [
27         .target(
28             name: "FetchingViewFeature"
29         ),
30         .testTarget(
31             name: "MasteringSwiftPMTests",
32             dependencies: ["FetchingViewFeature"]),
33     ]
34 )
35
36
37 name: "MasteringSwiftPMTests",
38 dependencies: ["FetchingViewFeature"]),
39 ]
```

The status bar at the bottom right indicates 'Line: 24 Col: 10'. The top right corner shows 'Build Succeeded | 02/09/24 at 3:30 PM'.



MasteringSwiftPM

MasteringSwiftPM > iPhone 15 Pro (17.5)

Build Succeeded | 02/09/24 at 3:30 PM | 1

MasteringSwiftPM 1 issue

⚠️ dependency 'alamofire' is not used by any target

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)

name: "MasteringSwiftPMTests",
dependencies: ["FetchingViewFeature"]),
]
```

Line: 24 Col: 10

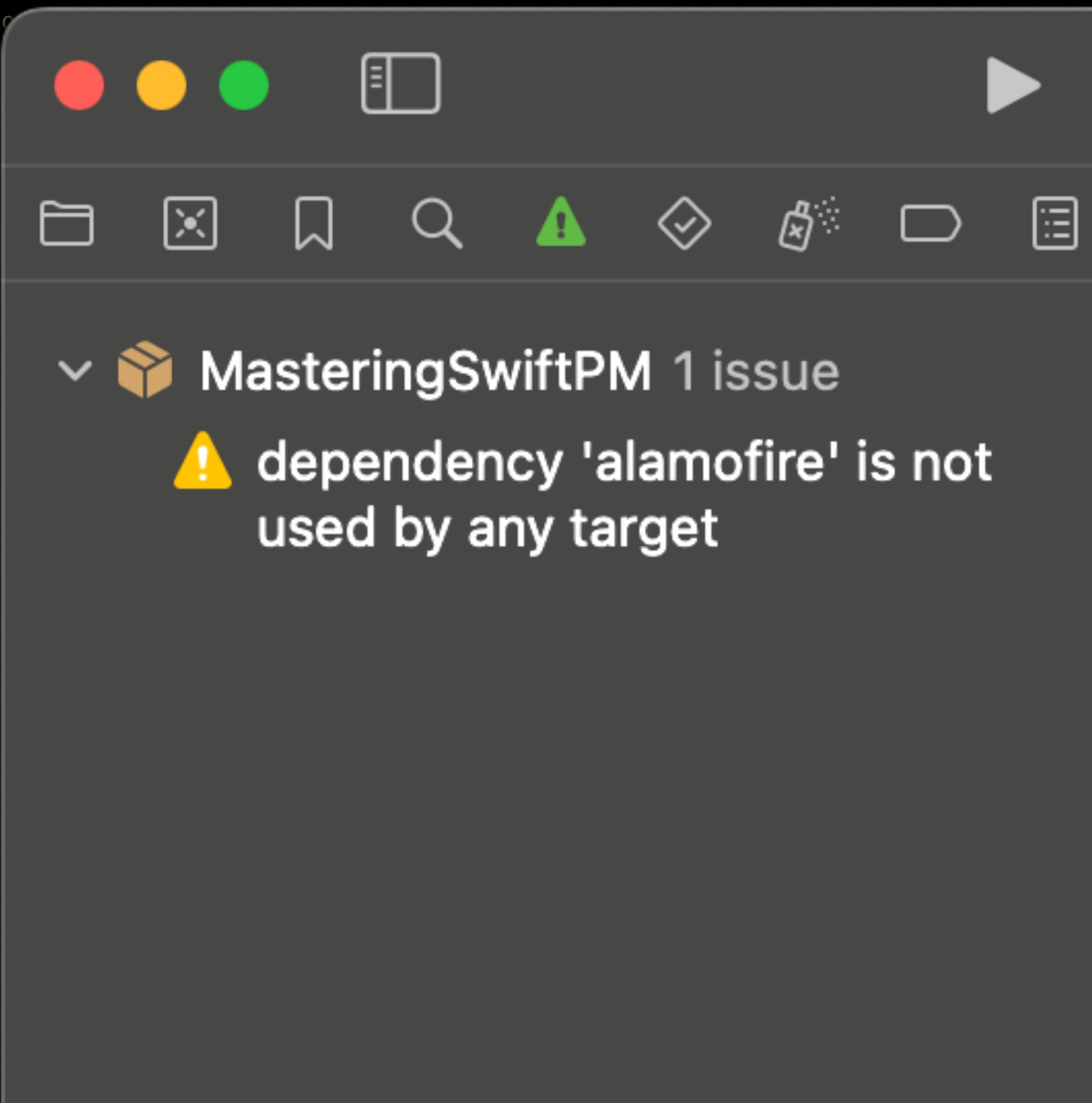


Dependencies

```
// swift-tools-version: 5.3
// The swift-tools-version specifies the minimum version of Swift required by the package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS],
    products: [
        .library(
            name: "MasteringSwiftPM",
            targets: []
        )
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            upToNextMajorVersion: true
        ),
        .package(
            url: "https://github.com/dfd/dfd.git",
            upToNextMajorVersion: true,
            from: "0.1.0"
        )
    ],
    targets: [
        .target(
            name: "MasteringSwiftPM"
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: [
                .target(name: "MasteringSwiftPM")
            ]
        )
    ]
)
```



The screenshot shows the Xcode interface with a dark theme. At the top, there are three circular status indicators (red, yellow, green) followed by a file icon. To the right is a play button icon. Below the toolbar are several icons: a folder, a delete, a bookmark, a search, a warning (yellow triangle), a checkmark, a target, and a list. The main area displays a package tree. A warning icon is shown next to the 'MasteringSwiftPM' node, indicating one issue. The details of the issue are listed below: '⚠️ dependency 'alamofire' is not used by any target'.

```
1 // swift-tools-version: 5.3
2 // The swift-tools-version specifies the minimum version of Swift required by the package.
3
4 import PackageDescription
5
6 let package = Package(
7     name: "MasteringSwiftPM",
8     platforms: [.iOS]
```



Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

```
📦 // swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

```
📦 // swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



URL Component

`https://github.com/Alamofire/Alamofire.git`

URL Scheme

`http://`

`ws://`

`ftp://`

`itms-apps://`

`file://`

`tel://`

`wss://`

`ssh://`

`itms://`

`mailto://`

`intent://`

`sms://`

`myapp://`



URL Component

`https://github.com/Alamofire/Alamofire.git`

Host

github.com

gitlab.com

bitbucket.com

Self hosted



URL Component

<https://github.com/Alamofire/Alamofire.git>

Author/Organisation

<https://github.com/ratnesh-jain/AssetPluginLibrary>

<https://github.com/hackiftekhar/IQKeyboardManager>

<https://github.com/SDWebImage/SDWebImageSwiftUI>

<https://github.com/pointfreeco/swift-composable-architecture>



URL Component

`https://github.com/Alamofire/Alamofire.git`

Repository Name

`https://github.com/ratnesh-jain/AssetPluginLibrary`

`https://github.com/hackiftekhar/IQKeyboardManager`

`https://github.com/SDWebImage/SDWebImageSwiftUI`

`https://github.com/pointfreeco/swift-composable-architecture`

```
📦 // swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

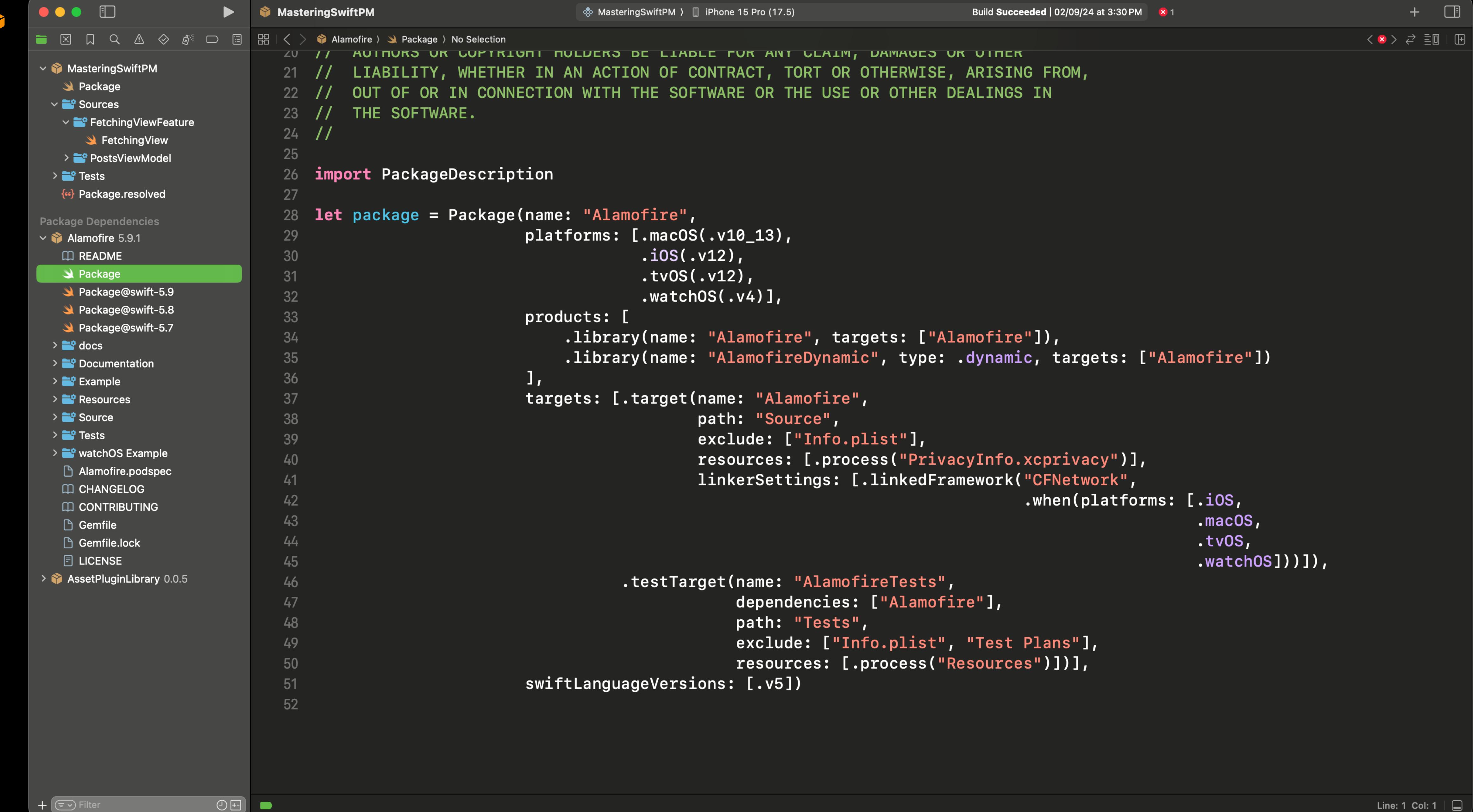
let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

MasteringSwiftPM

MasteringSwiftPM > iPhone 15 Pro (17.5) Build Succeeded | 02/09/24 at 3:30 PM x 1

Alamofire > README > No Selection

```
1 ! [Alamofire: Elegant Networking in
      Swift](https://raw.githubusercontent.com/Alamofire/Alamofire/master/Resources/AlamofireLogo.png)
2
3 [!
4   [Swift](https://img.shields.io/badge/Swift-5.7_5.8_5.9-orange?style=flat-square)](https://img.shields
      .io/badge/Swift-5.7_5.8_5.9-Orange?style=flat-square)
5 [!
6   [Platforms](https://img.shields
      .io/badge/Platforms-macOS_iOS_tvOS_watchOS_visionOS_Linux_Windows_Android-yellowgreen?style=flat-square]
      (https://img.shields
      .io/badge/Platforms-macOS_iOS_tvOS_watchOS_vision_OS_Linux_Windows_Android-Green?style=flat-square)
7 [![
8   [CocoaPods
     Compatible](https://img.shields.io/cocoapods/v/Alamofire.svg?style=flat-square)](https://img.shields
      .io/cocoapods/v/Alamofire.svg)
9 [![
10  [Carthage
     Compatible](https://img.shields.io/badge/Carthage-compatible-4BC51D.svg?style=flat-square)](https://github
      .com/Carthage/Carthage)
11 [![
12  [Swift Package
     Manager](https://img.shields.io/badge/Swift_Package_Manager-compatible-orange?style=flat-square)](https://img
      .shields.io/badge/Swift_Package_Manager-compatible-orange?style=flat-square)
13 [![
14  [Swift
     Forums](https://img.shields.io/badge/Swift_Forum-Alamofire-orange?style=flat-square)](https://forums.swift
      .org/c/related-projects/alamofire/37)
15
16
17
18
19
20
```



The screenshot shows the Xcode interface with the project "MasteringSwiftPM" open. The file "Alamofire.package" is selected in the sidebar under "Package Dependencies". The code editor displays the package description for Alamofire 5.9.1.

```
// AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
// LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
// OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
// THE SOFTWARE.

import PackageDescription

let package = Package(name: "Alamofire",
    platforms: [.macOS(.v10_13),
        .iOS(.v12),
        .tvOS(.v12),
        .watchOS(.v4)],
    products: [
        .library(name: "Alamofire", targets: ["Alamofire"]),
        .library(name: "AlamofireDynamic", type: .dynamic, targets: ["Alamofire"])
    ],
    targets: [.target(name: "Alamofire",
        path: "Source",
        exclude: ["Info.plist"],
        resources: [.process("PrivacyInfo.xcprivacy")],
        linkerSettings: [.linkedFramework("CFNetwork",
            .when(platforms: [.iOS,
                .macOS,
                .tvOS,
                .watchOS]))],
        .testTarget(name: "AlamofireTests",
            dependencies: ["Alamofire"],
            path: "Tests",
            exclude: ["Info.plist", "Test Plans"],
            resources: [.process("Resources")])],
        swiftLanguageVersions: [.v5])

```

Alamofire > Package > No Selection

```
20 // AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
21 // LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM
22 // OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
23 // THE SOFTWARE.
24 //
25
26 import PackageDescription
27
28 let package = Package(name: "Alamofire",
29                         platforms: [.macOS(.v10_13),
30                                     .iOS(.v12),
31                                     .tvOS(.v12),
32                                     .watchOS(.v4)],
33                         products: [
34                             .library(name: "Alamofire", targets: ["Alamofire"]),
35                             .library(name: "AlamofireDynamic", type: .dynamic, tar
36                         ],
37                         targets: [.target(name: "Alamofire",
38                                         path: "Source",
39                                         exclude: ["Info.plist"]),
40                                         resources: [.process("PrivacyInfo.xcpriv"])]
```

Alamofire > Package > No Selection

```
20 // AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
21 // LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM
22 // OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
23 // THE SOFTWARE.
24 //
25
26 import PackageDescription
27
28 let package = Package(name: "Alamofire",
29                         platforms: [.macOS(.v10_13),
30                                     .iOS(.v12),
31                                     .tvOS(.v12),
32                                     .watchOS(.v4)],
33                         products: [
34                             .library(name: "Alamofire", targets: ["Alamofire"]),
35                             .library(name: "AlamofireDynamic", type: .dynamic, tar
36                         ],
37                         targets: [.target(name: "Alamofire",
38                                         path: "Source",
39                                         exclude: ["Info.plist"]),
40                                         resources: [.process("PrivacyInfo.xcpriv"])]
```

Alamofire > Package > No Selection

```
20 // AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
21 // LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM
22 // OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
23 // THE SOFTWARE.
24 //
25
26 import PackageDescription
27
28 let package = Package(name: "Alamofire",
29                         platforms: [.macOS(.v10_13),
30                                     .iOS(.v12),
31                                     .tvOS(.v12),
32                                     .watchOS(.v4)],
33                         products: [
34                             .library(name: "Alamofire", targets: ["Alamofire"]),
35                             .library(name: "AlamofireDynamic", type: .dynamic, tar
36                         ],
37                         targets: [.target(name: "Alamofire",
38                                         path: "Source",
39                                         exclude: ["Info.plist"]),
40                                         resources: [.process("PrivacyInfo.xcpriv"])]
```

I OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
E SOFTWARE.

PackageDescription

```
ckage = Package(name: "Alamofire",
  platforms: [.macOS(.v10_13),
    .iOS(.v12),
    .tvOS(.v12),
    .watchOS(.v4)],
  products: [
    .library(name: "Alamofire", targets: ["Alamofire"]),
    .library(name: "AlamofireDynamic", type: .dynamic, targets: ["Alamo"]),
  ],
  targets: [.target(name: "Alamofire",
    path: "Source",
    exclude: ["Info.plist"],
    resources: [.process("PrivacyInfo.xcprivacy")]),
    linkerSettings: [.linkedFramework("CFNetwork",
      .when(platforms: [
        .platform(.macOS, .v10_13),
        .platform(.iOS, .v12),
        .platform(.tvOS, .v12),
        .platform(.watchOS, .v4)
      ])])
  ]
}
```

I OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
E SOFTWARE.

PackageDescription

```
ckage = Package(name: "Alamofire",
  platforms: [.macOS(.v10_13),
    .iOS(.v12),
    .tvOS(.v12),
    .watchOS(.v4)],
  products: [
    .library(name: "Alamofire", targets: ["Alamofire"]),
    .library(name: "AlamofireDynamic", type: .dynamic, targets: ["Alamo
  ],
  targets: [.target(name: "Alamofire",
    path: "Source",
    exclude: ["Info.plist"],
    resources: [.process("PrivacyInfo.xcprivacy")]),
    linkerSettings: [.linkedFramework("CFNetwork",
      .when(platforms: [
        .iO
      ]))
  ]
}
```

cription

```
package(name: "Alamofire",
    platforms: [.macOS(.v10_13),
        .iOS(.v12),
        .tvOS(.v12),
        .watchOS(.v4)],
    products: [
        .library(name: "Alamofire", targets: ["Alamofire"]),
        .library(name: "AlamofireDynamic", type: .dynamic, targets: ["Alamofire"])
    ],
    targets: [.target(name: "Alamofire",
        path: "Source",
        exclude: ["Info.plist"],
        resources: [.process("PrivacyInfo.xcprivacy")],
        linkerSettings: [.linkedFramework("CFNetwork",
            .when(platforms: [.iOS,
                .macOS,
                .tvOS,
                .watchOS]))]
```

cription

```
package(name: "Alamofire",
    platforms: [.macOS(.v10_13),
        .iOS(.v12),
        .tvOS(.v12),
        .watchOS(.v4)],
    products: [
        .library(name: "Alamofire", targets: ["Alamofire"]),
        .library(name: "AlamofireDynamic", type: .dynamic, targets: ["Alamofire"])
    ],
    targets: [.target(name: "Alamofire",
        path: "Source",
        exclude: ["Info.plist"],
        resources: [.process("PrivacyInfo.xcprivacy")],
        linkerSettings: [.linkedFramework("CFNetwork",
            .when(platforms: [.iOS,
                .macOS,
                .tvOS,
                .watchOS]))]
```



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

Target Dependency

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

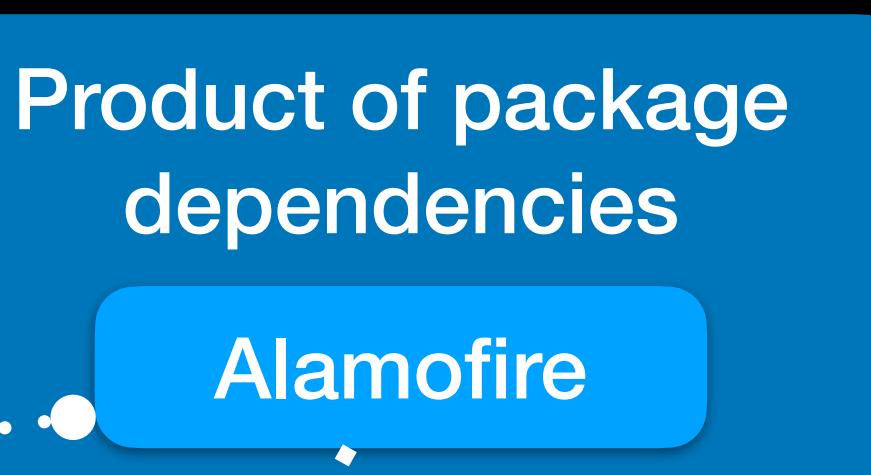


Package Dependencies

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

  
import PackageDescription  
  
let package = Package(  
    name: "MasteringSwiftPM",  
    platforms: [.iOS(.v17)],  
    products: [  
        .library(  
            name: "FetchingViewFeature",  
            targets: ["FetchingViewFeature"]),  
    ],  
    dependencies: [  
        .package(  
            url: "https://github.com/Alamofire/Alamofire.git",  
            .upToNextMajor(from: "5.0.0")  
        ),  
        .package(  
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",  
            .upToNextMajor(  
                from: "0.0.5"  
            )  
        )  
    ],  
    targets: [  
        .target(name: "Resources"),  
        .target(  
            name: "FetchingViewFeature",  
            dependencies: [  
                .product(name: "Alamofire", package: "Alamofire"),  
                "Resources"  
            ]  
        ),  
        .testTarget(  
            name: "MasteringSwiftPMTests",  
            dependencies: ["FetchingViewFeature"])  
    ]  
)
```

Product of package
dependencies

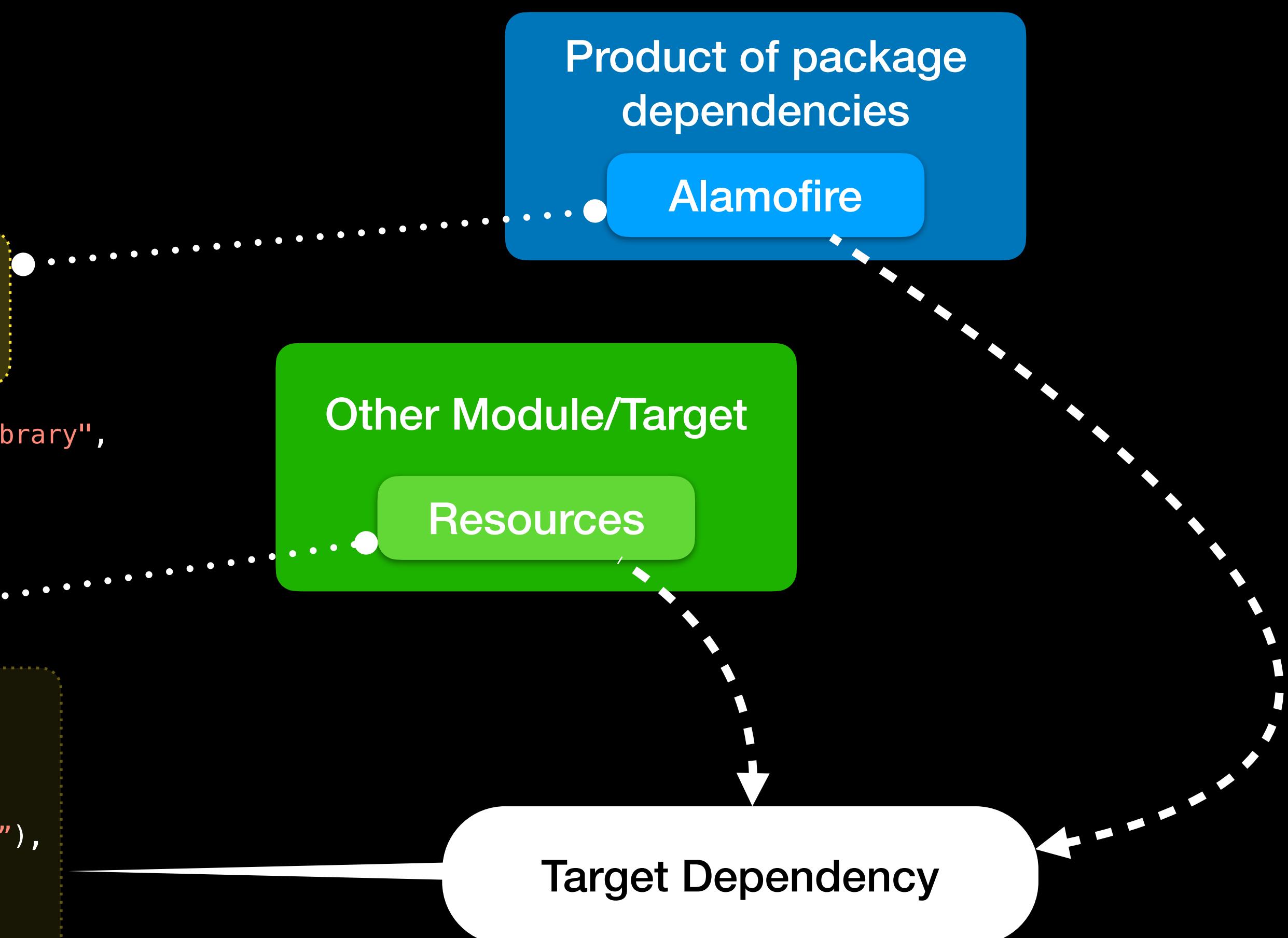
Alamofire

Target Dependency

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(name: "Resources"),
        .target(
            name: "FetchingViewFeature",
            dependencies: [
                .product(name: "Alamofire", package: "Alamofire"),
                "Resources"
            ]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"])
    ]
)
```



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

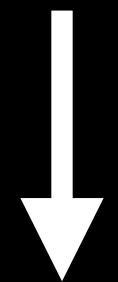


Dependencies



Source Code + Dependencies

Source Code + Dependencies





Source Code + Dependencies

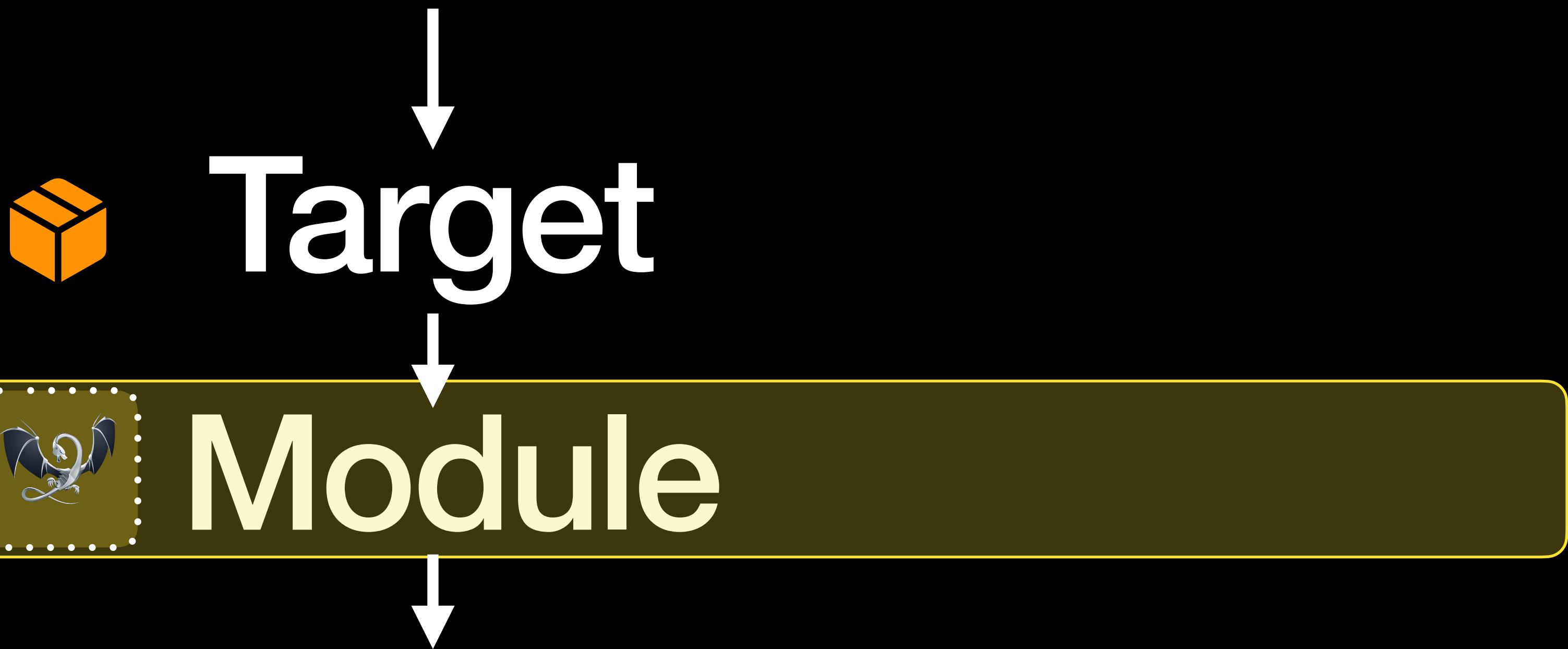


Target



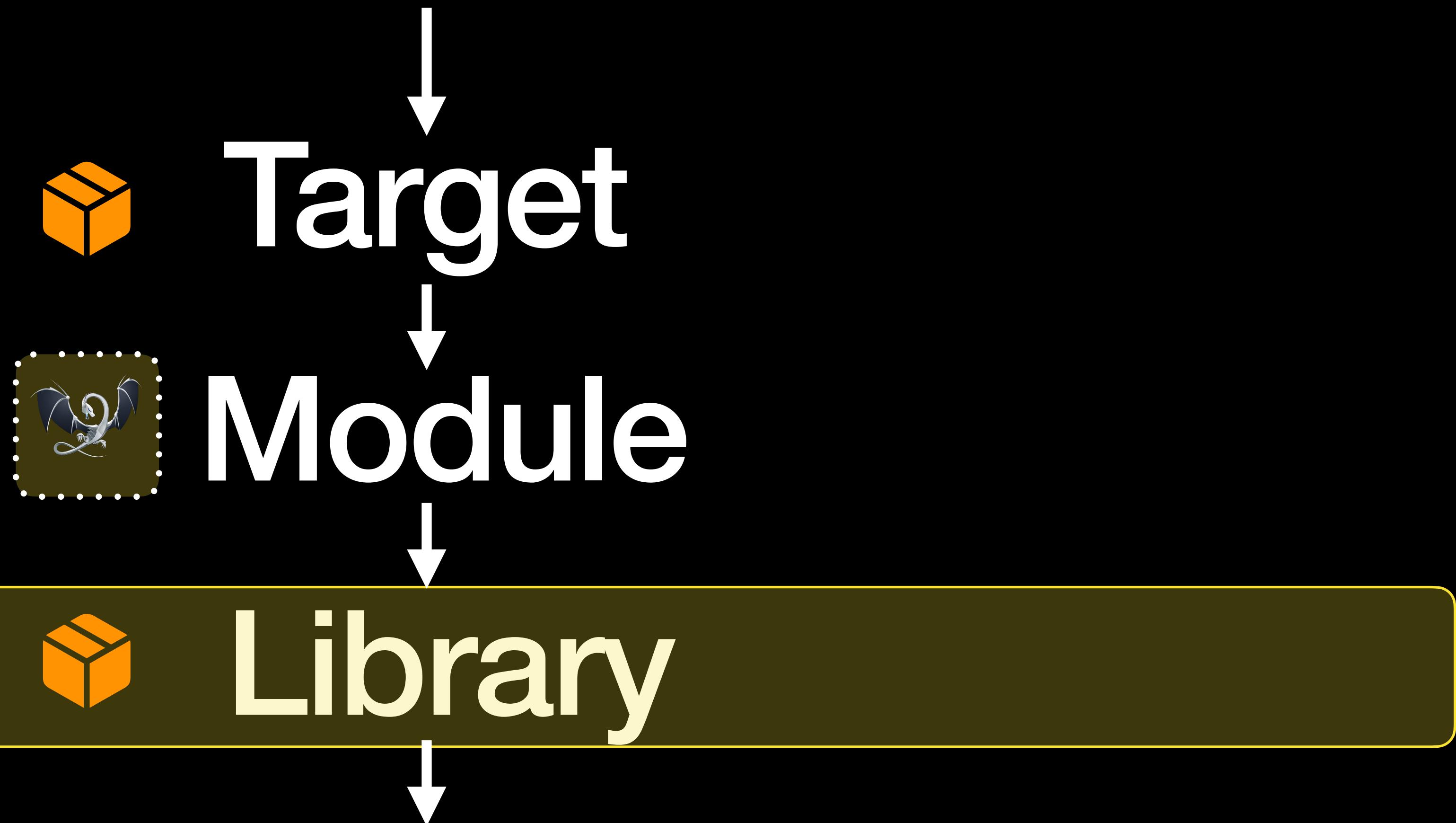


Source Code + Dependencies



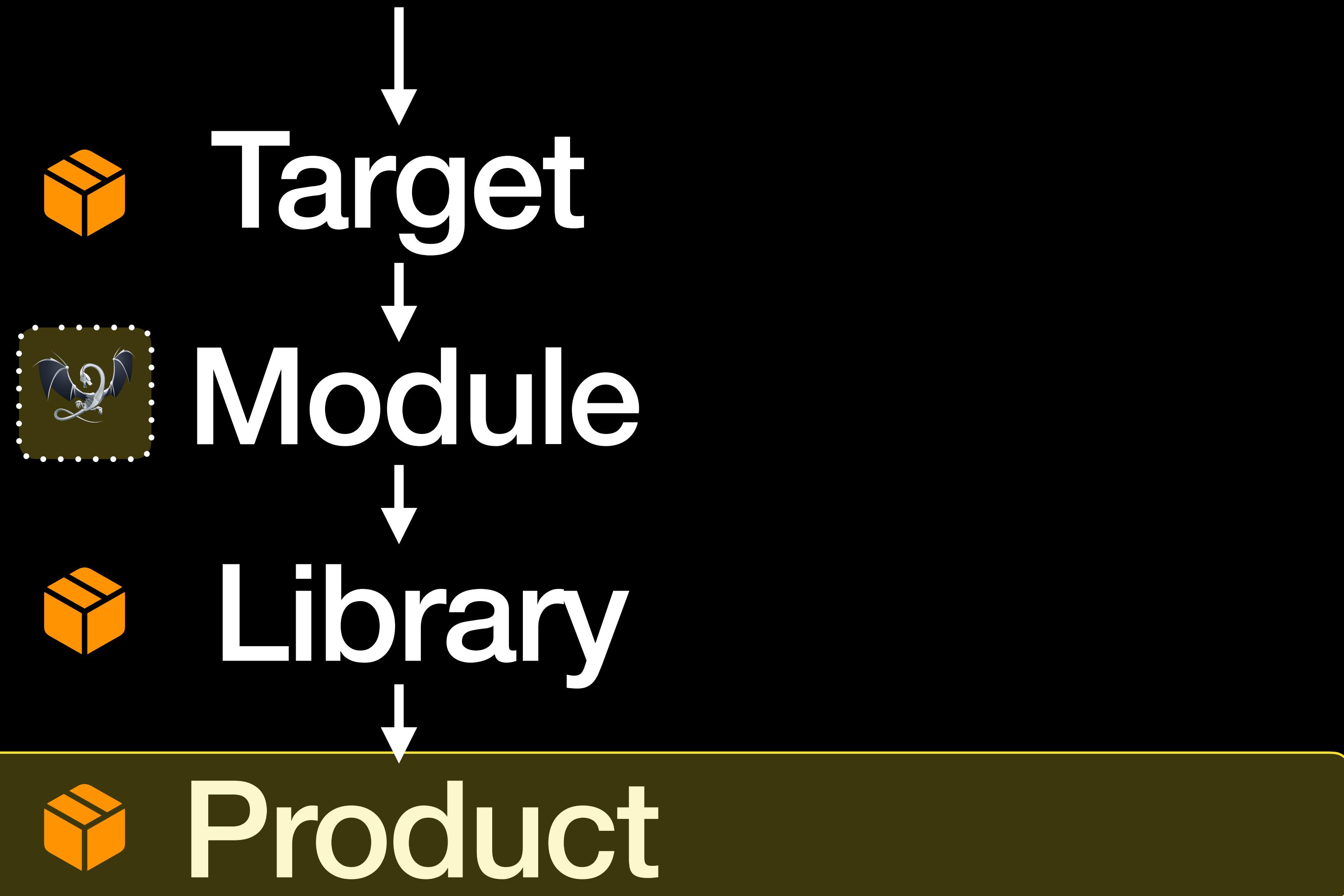


Source Code + Dependencies





Source Code + Dependencies





Mastering SwiftPM

Today

- Access Modifiers
- Package Manifest
- Package Manager Vocabulary
- Dependencies
- New Feature using SwiftPM
- Integration with Xcode Project
- Sharing Resources
- Dealing with Non-SPM external Dependency



New Feature using SwiftPM





Mastering SwiftPM

Today

- Access Modifiers
- Package Manifest
- Package Manager Vocabulary
- Dependencies
- New Feature using SwiftPM
- Integration with Xcode Project
- Sharing Resources
- Dealing with Non-SPM external Dependency



Integration with Xcode Project

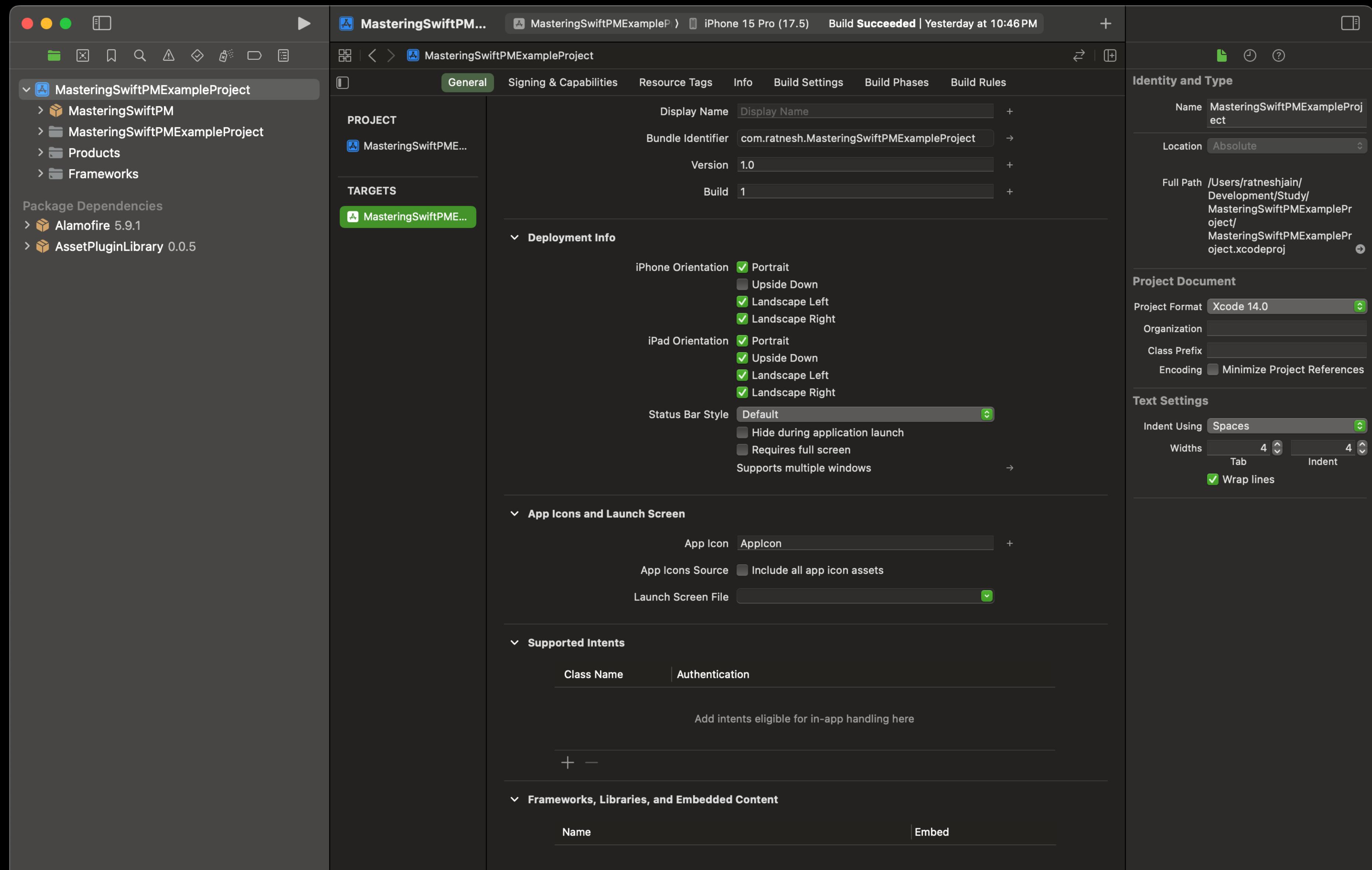
```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



Integration with Xcode Project



The screenshot shows the Xcode project settings for "MasteringSwiftPMExampleProject". The "General" tab is selected. Key details include:

- Display Name:** MasteringSwiftPMExampleProject
- Bundle Identifier:** com.ratnesh.MasteringSwiftPMExampleProject
- Version:** 1.0
- Build:** 1
- Deployment Info:** iPhone Orientation includes Portrait, Landscape Left, and Landscape Right; iPad Orientation includes Portrait, Upside Down, Landscape Left, and Landscape Right.
- Status Bar Style:** Default
- App Icons and Launch Screen:** App Icon is set to AppIcon, and the Launch Screen File is blank.
- Supported Intents:** A single intent named "Authentication" is listed under Class Name.
- Frameworks, Libraries, and Embedded Content:** No frameworks or libraries are listed.

The sidebar on the left lists the project structure, including "MasteringSwiftPMExampleProject", "MasteringSwiftPM", "Products", "Frameworks", and package dependencies like Alamofire 5.9.1 and AssetPluginLibrary 0.0.5.



Integration with Xcode Project

The screenshot shows the Xcode interface with the project `MasteringSwiftPMExampleProject` selected in the left sidebar. The sidebar also lists `MasteringSwiftPM`, `Products`, and `Frameworks`. Under `Package Dependencies`, it shows `Alamofire 5.9.1` and `AssetPluginLibrary 0.0.5`. The main area displays the `General` tab of the project settings, which includes sections for `PROJECT` and `TARGETS`. The `PROJECT` section shows the target `MasteringSwiftPME...`. The `TARGETS` section shows the target `MasteringSwiftPME...` with a green bar highlighting it. Below the targets, there is a collapsed section labeled `Deployment Info`. The bottom right corner of the screenshot shows the text `iPhone O` and `iPad O`.



Integration with Xcode Project

The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure. The target "MasteringSwiftPMExampleProject" is expanded, revealing its subfolders: "MasteringSwiftPM" (highlighted with a yellow box), "MasteringSwiftPMExampleProject", "Products", and "Frameworks".
- Package Dependencies:** Shows the dependencies of the project:
 - > **Alamofire 5.9.1**
 - > **AssetPluginLibrary 0.0.5**
- General Tab:** The active tab in the Xcode settings sidebar. It displays the project name "MasteringSwiftPMExampleProject" and the target "MasteringSwiftPMExampleProject".
- Targets:** Shows the target "MasteringSwiftPMExampleProject" (highlighted with a green box).
- Deployment Info:** A collapsed section under Targets.
- Devices:** Shows two devices: "iPhone 0" and "iPad 0".



Integration with Xcode Project

The screenshot shows the Xcode interface with the project "MasteringSwiftPMExampleProject" selected. The General tab is active, displaying various configuration options:

- PROJECT**:
 - Display Name: MasteringSwiftPMExampleProject
 - Bundle Identifier: com.ratnesh.MasteringSwiftPMExampleProject
 - Version: 1.0
 - Build: 1
- TARGETS**:
 - MasteringSwiftPMExampleProject (selected)
- Deployment Info**:
 - iPhone Orientation**: Portrait, Landscape Left, Landscape Right
 - iPad Orientation**: Portrait, Landscape Left, Landscape Right
 - Status Bar Style**: Default, Hide during application launch, Requires full screen
 - Supports multiple windows
- App Icons and Launch Screen**:
 - App Icon: AppIcon
 - App Icons Source: Include all app icon assets
 - Launch Screen File: (dropdown menu)
- Supported Intents**:
 - Class Name: Authentication
 - Add intents eligible for in-app handling here
- Frameworks, Libraries, and Embedded Content**: (empty table)

The sidebar on the left lists package dependencies: Alamofire 5.9.1 and AssetPluginLibrary 0.0.5.



Integration with Xcode Project

The screenshot shows the Xcode interface with the project "MasteringSwiftPMExampleProject" selected. The General tab is active, displaying various configuration options:

- PROJECT**:
 - Display Name: MasteringSwiftPMExampleProject
 - Bundle Identifier: com.ratnesh.MasteringSwiftPMExampleProject
 - Version: 1.0
 - Build: 1
- TARGETS**:
 - MasteringSwiftPMExampleProject (selected, highlighted with a green border)
- Deployment Info**:
 - iPhone Orientation**: Portrait, Landscape Left, Landscape Right
 - iPad Orientation**: Portrait, Landscape Left, Landscape Right
 - Status Bar Style**: Default, Hide during application launch, Requires full screen
 - Supports multiple windows
- App Icons and Launch Screen**:
 - App Icon: AppIcon
 - App Icons Source: Include all app icon assets
 - Launch Screen File: (dropdown menu)
- Supported Intents**:
 - Class Name: Authentication
 - Add intents eligible for in-app handling here
- Frameworks, Libraries, and Embedded Content**: (empty table)

Identity and Type sidebar:

- Name: MasteringSwiftPMExampleProject
- Location: Absolute
- Full Path: /Users/ratneshjain/Development/Study/MasteringSwiftPMExampleProject/MasteringSwiftPMExampleProject.xcodeproj

Project Document sidebar:

- Project Format: Xcode 14.0
- Organization: (empty)
- Class Prefix: (empty)
- Encoding: Minimize Project References

Text Settings sidebar:

- Indent Using: Spaces
- Widths: Tab 4, Indent 4
- Wrap lines: checked

▼ App Icons and Launch Screen

App Icon **Applcon**

App Icons Source Include all app icon assets

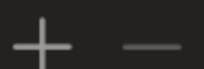
Launch Screen File 

▼ Supported Intents

Class Name

Authentication

Add intents eligible for in-app handling here

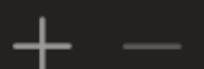


▼ Frameworks, Libraries, and Embedded Content

Name

Embed

Add frameworks, libraries, and embedded content here



▼ Development Assets

MasteringSwiftPMEExampleProject/Preview Content

+ - 



▼ App Icons and Launch Screen

App Icon **Applcon**

App Icons Source Include all app icon assets

Launch Screen File 

▼ Supported Intents

Class Name

Authentication

Add intents eligible for in-app handling here



-

▼ Frameworks, Libraries, and Embedded Content

Name

Embed

Add frameworks, libraries, and embedded content here



▼ Development Assets

MasteringSwiftPMEExampleProject/Preview Content

+ - 





Integrating SwiftPM into Xcode

The screenshot shows the Xcode interface with the project "MasteringSwiftPMExampleProject" open. The "General" tab is selected in the top navigation bar. In the "Identity and Type" section, the bundle identifier is set to "com.ratnesh.MasteringSwiftPMExampleProject". The "Deployment Info" section shows iPhone orientations: Portrait, Landscape Left, and Landscape Right are checked. A modal dialog titled "Choose frameworks and libraries to add:" is displayed, listing "Workspace" and "Apple SDKs" sections. The "Apple SDKs" section contains a list of frameworks: Accelerate.framework, Accessibility.framework, Accounts.framework, ACIPCBTLib.tbd, ActivityKit.framework, AdAttributionKit.framework, AddressBook.framework, AddressBookUI.framework, AdServices.framework, AdSupport.framework, and AppClip.framework. At the bottom of the modal are "Add Other..." and "Cancel" buttons, and an "Add" button is highlighted. The main workspace shows the "Frameworks, Libraries, and Embedded Content" section with a placeholder message: "Add frameworks, libraries, and embedded content here".

Choose frameworks and libraries to add:

▼  Workspace▼  MasteringSwiftPM

FetchingViewFeature

▼  Apple SDKs Accelerate.framework Accessibility.framework Accounts.framework ACIPCBTLib.tbd ActivityKit.framework AdAttributionKit.framework AddressBook.framework AddressBookUI.framework AdServices.framework AdSupport.framework AppClip.framework

▼ Frameworks, Libraries, and Embedded Content

Name

Embed

Choose frameworks and libraries to add:

- ▼  Workspace
 - ▼  MasteringSwiftPM
 -  FetchingViewFeature
- ▼  Apple SDKs
 -  Accelerate.framework
 -  Accessibility.framework
 -  Accounts.framework
 -  ACIPCBTLib.tbd
 -  ActivityKit.framework
 -  AdAttributionKit.framework
 -  AddressBook.framework
 -  AddressBookUI.framework
 -  AdServices.framework
 -  AdSupport.framework
 -  AppClip.framework



▼ Frameworks, Libraries, and Embedded Content

Name

Embed



Integrating SwiftPM into Xcode

MasteringSwiftPM... MasteringSwiftPMEExampleProject iPhone 15 Pro (17.5) Build Succeeded | Yesterday at 10:46PM +

MasteringSwiftPMEExampleProject

General Signing & Capabilities Resource Tags Info Build Settings Build Phases Build Rules

PROJECT

MasteringSwiftPME... MasteringSwiftPME...

TARGETS

MasteringSwiftPME...

Deployment Info

iPhone Orientation: Portrait Upside Down Landscape Left Landscape Right

iPad Orientation: Portrait Upside Down Landscape Left Landscape Right

Status Bar Style: Default Hide during application launch Requires full screen

Supports multiple windows: →

App Icons and Launch Screen

App Icon: AppIcon +

App Icons Source: Include all app icon assets

Launch Screen File: →

Supported Intents

Class Name: Authentication

Add intents eligible for in-app handling here

+ -

Frameworks, Libraries, and Embedded Content

Name	Embed
FetchingViewFeature	

+ -

Development Assets

MasteringSwiftPMEExampleProject/Preview Content

+ -

Identity and Type

Name: MasteringSwiftPMEExampleProject

Location: Absolute

Full Path: /Users/ratneshjain/Development/Study/MasteringSwiftPMEExampleProject/MasteringSwiftPMEExampleProject.xcodeproj

Project Document

Project Format: Xcode 14.0

Organization:

Class Prefix:

Encoding: Minimize Project References

Text Settings

Indent Using: Spaces

Widths: 4 Tab 4 Indent

Wrap lines

+ Filter

Filter

⌄ Supported Intents

Class Name

Authentication

Add intents eligible for in-app handling here



⌄ Frameworks, Libraries, and Embedded Content

Name

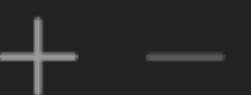
Embed

FetchingViewFeature



⌄ Development Assets

MasteringSwiftPMExampleProject/Preview Content



Filter





Integration with Xcode Project

The screenshot shows the Xcode interface with a project named "MasteringSwiftPMExampleProject". The ContentView.swift file is open in the editor. A completion popover is displayed over the code, showing suggestions for "FetchingViewFeature". The code itself contains basic SwiftUI code for displaying a globe icon and some text.

```
1 // ContentView.swift
2 // MasteringSwiftPMExampleProject
3 // Created by Ratnesh Jain on 06/09/24.
4 //
5 import SwiftUI
6 import Fet|
7
8 str FetchingViewFeature
9 FetchingViewFeature: Module
10
11 VStack {
12     Image(systemName: "globe")
13         .imageScale(.large)
14         .foregroundStyle(.tint)
15     Text("Hello, world!")
16 }
17
18 .padding()
19
20 }
21 }
```

Identity and Type

- Name: ContentView.swift
- Type: Default - Swift Source
- Location: Relative to Group
- ContentView.swift
- Full Path: /Users/ratneshjain/Development/Study/MasteringSwiftPMExampleProject/MasteringSwiftPMExampleProject/ContentView.swift

On Demand Resource Tags

- Only resources are taggable

Target Membership

- MasteringSwiftPMExampleProject

Text Settings

- Text Encoding: No Explicit Encoding
- Line Endings: No Explicit Line Endings
- Indent Using: Spaces
- Widths: 4 Tab 4 Indent
- Wrap lines



Mastering SwiftPM

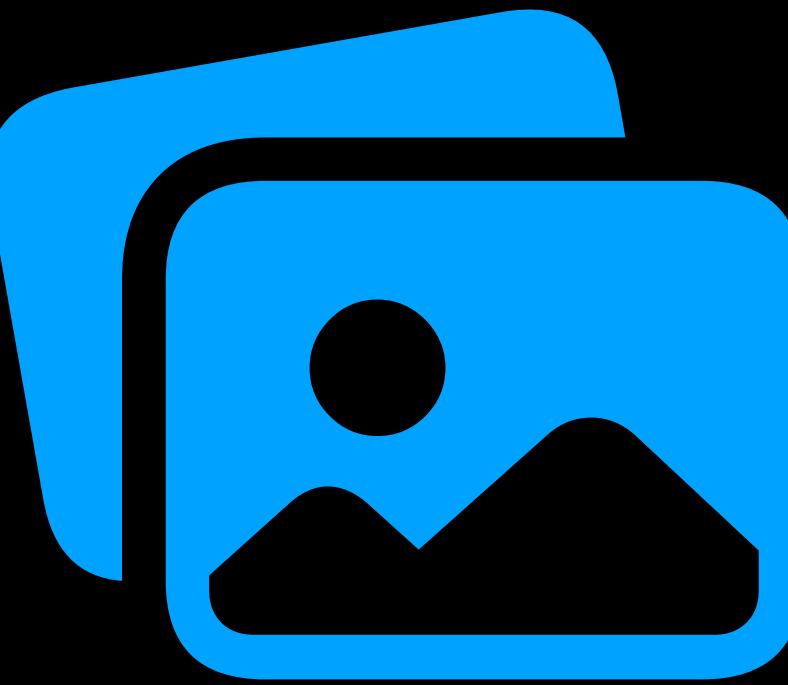
Today

- Access Modifiers
- Package Manifest
- Package Manager Vocabulary
- Dependencies
- New Feature using SwiftPM
- Integration with Xcode Project
- Sharing Resources
- Dealing with Non-SPM external Dependency



Sharing Resources With Other Modules

- Assets
 - Images
 - Colors
 - Fonts
- Core Data
- Other Resources



Assets.xcassets



Sharing Resources

Assets



- Create New Folder
- Name it “Resources”

The screenshot shows the Xcode interface. On the left, the Project Navigator displays a project named "MasteringSwiftPM". Inside the "Sources" folder, there are "FetchingViewFeature" and "PostsViewModel" subfolders, and a newly created "Resources" folder which is highlighted with a green bar. In the center, the Editor pane shows the code for "FetchingView.swift". The code imports SwiftUI and Alamofire, defines a public struct FetchingView, and returns a VStack containing a ProgressView and a Text("Loading") view.

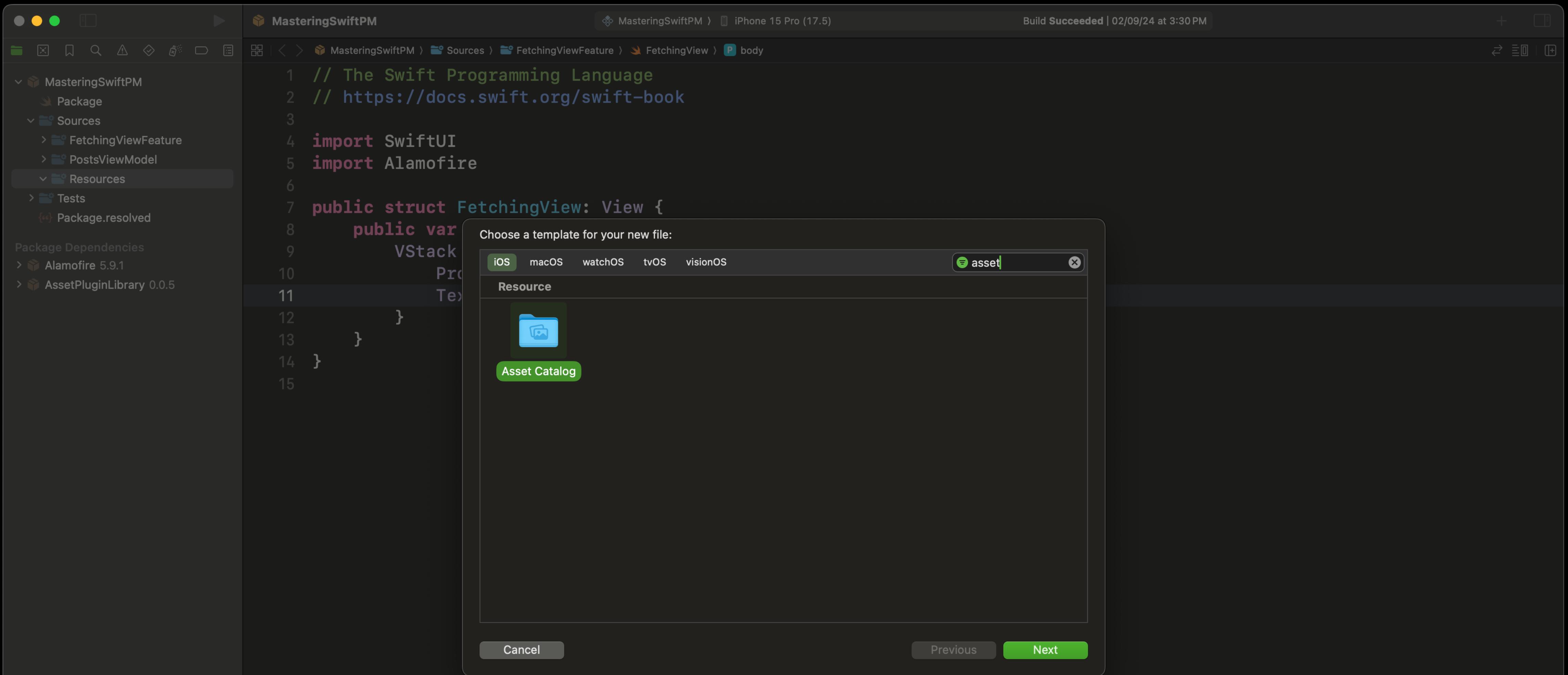
```
1 // The Swift Programming Language
2 // https://docs.swift.org/swift-book
3
4 import SwiftUI
5 import Alamofire
6
7 public struct FetchingView: View {
8     public var body: some View {
9         VStack {
10             ProgressView()
11             Text("Loading")
12         }
13     }
14 }
15
```



Sharing Resources

Assets 

- File → New File • Search for asset



The screenshot shows the Xcode interface with a dark theme. In the center, a modal dialog titled "Choose a template for your new file:" is open. It lists several platform options: iOS (selected), macOS, watchOS, tvOS, and visionOS. Below the platforms, there is a search bar containing the text "asset". Under the search bar, there is a section labeled "Resource" which contains an icon of a folder with a document and the text "Asset Catalog". At the bottom of the modal, there are "Cancel", "Previous", and "Next" buttons.

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

import SwiftUI
import Alamofire

public struct FetchingView: View {
    public var body: some View {
        VStack {
            Text("Hello, world!")
        }
    }
}
```

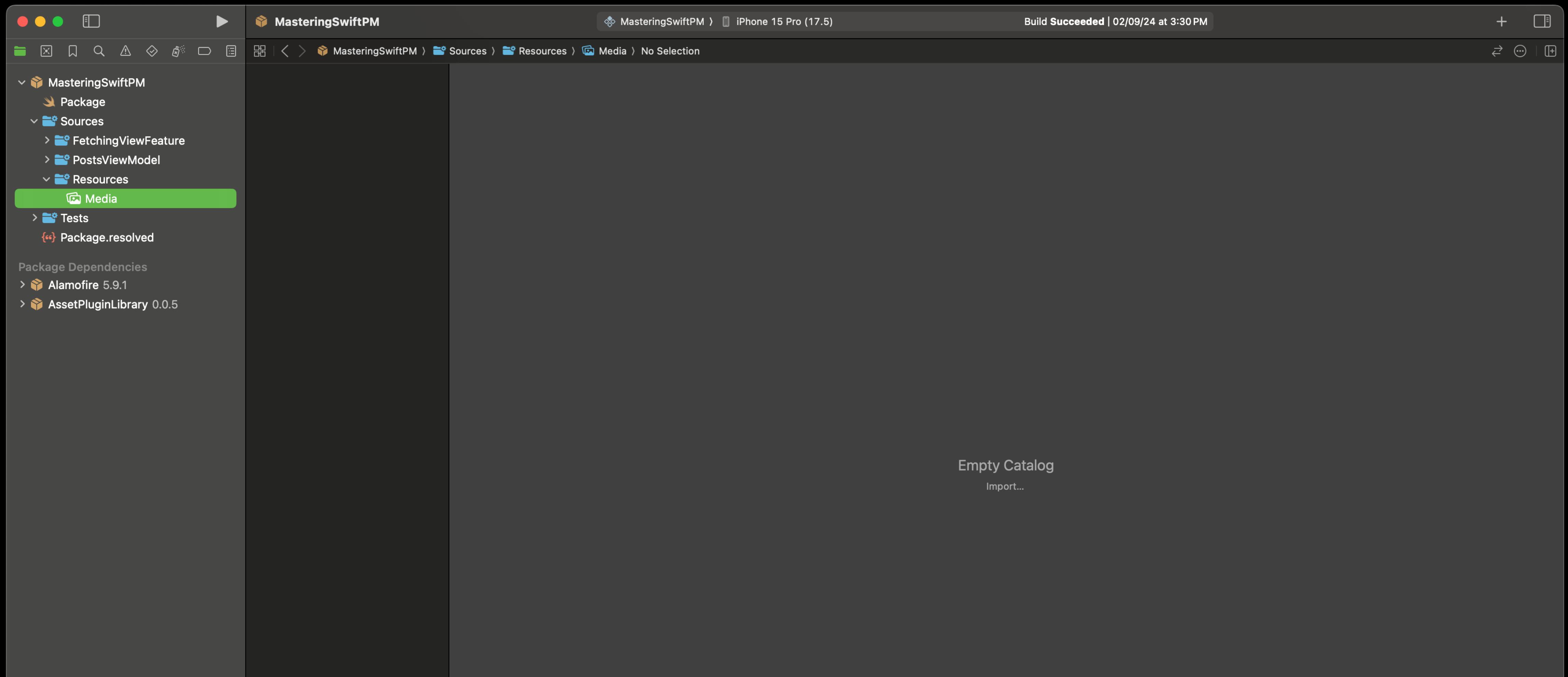


Sharing Resources

Assets



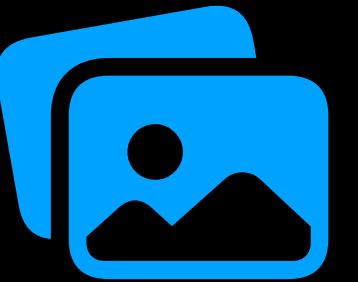
- Will place images here





Sharing Resources

Assets



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [ ... ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



Sharing Resources

Assets



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [ ... ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(name: "Resources"),
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



Sharing Resources

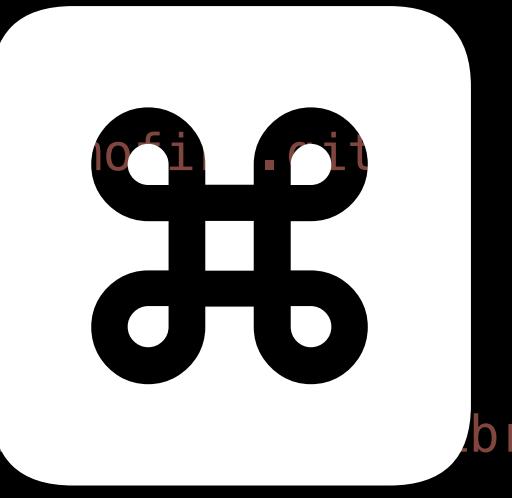
Assets



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [ ... ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/NetworkingLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(name: "Resources"),
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



Save

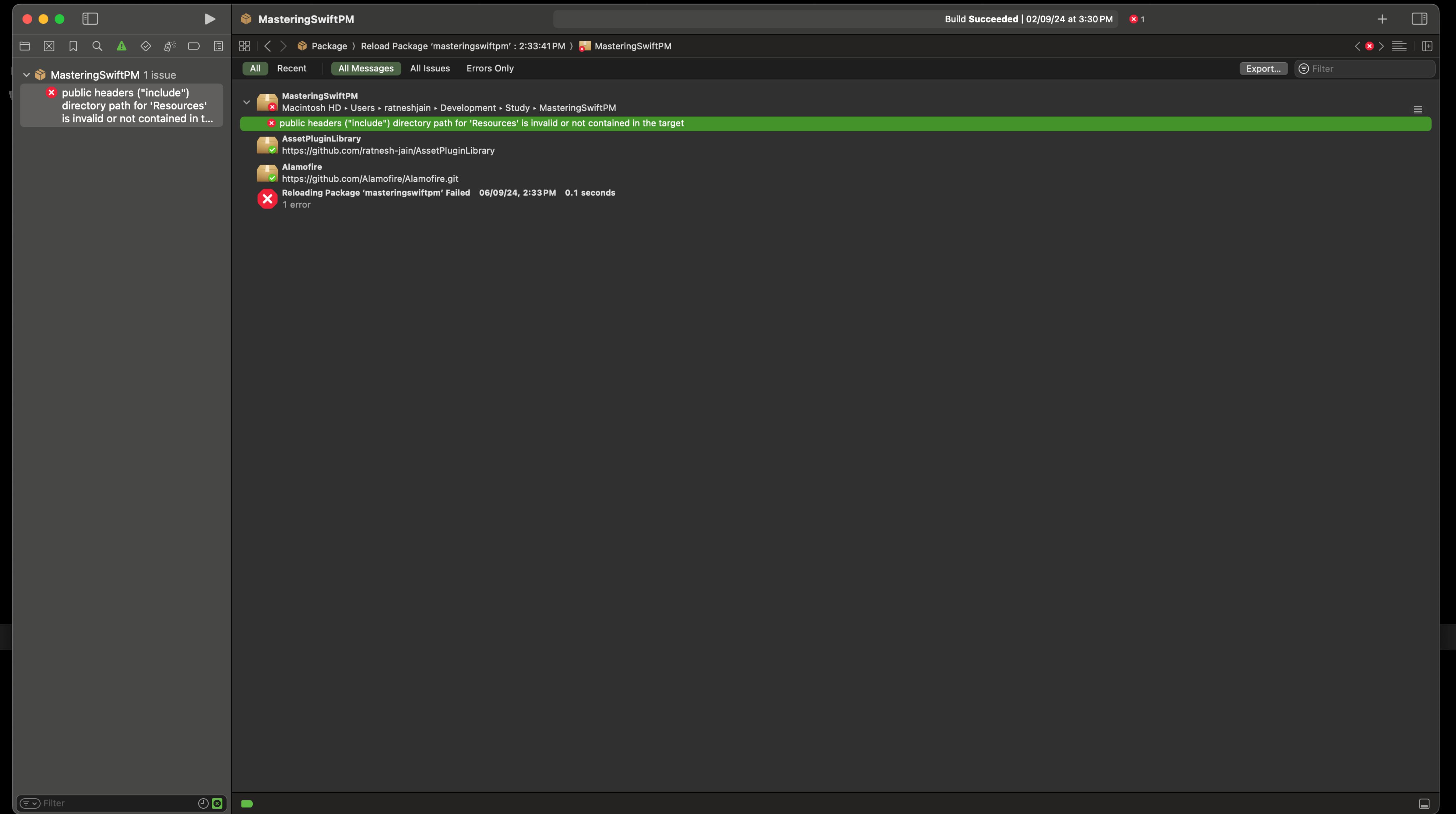
The screenshot shows the Xcode interface with the following details:

- Title Bar:** MasteringSwiftPM
- Project Navigator:** Shows the project structure with a selected "Package" item.
- File Navigator:** Shows package dependencies: Alamofire 5.9.1 and AssetPluginLibrary 0.0.5.
- Editor:** Displays the `Package.swift` file content.
- Toolbar:** Includes standard Xcode icons for file operations.
- Status Bar:** Reloading Package 'masteringswiftpm' (green circle), Line: 27 Col: 36.

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [ ... ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(name: "Resources"),
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"])
    ]
)
```



MasteringSwiftPM

Package > Reload Package 'masteringswiftpm' : 2:33:41PM > MasteringSwiftPM

All Recent

All Messages

All Issues

Errors Only

MasteringSwiftPM



Macintosh HD ▶ Users ▶ ratneshjain ▶ Development ▶ Study ▶ MasteringSwiftPM

✖ public headers ("include") directory path for 'Resources' is invalid or not contained in the target

AssetPluginLibrary



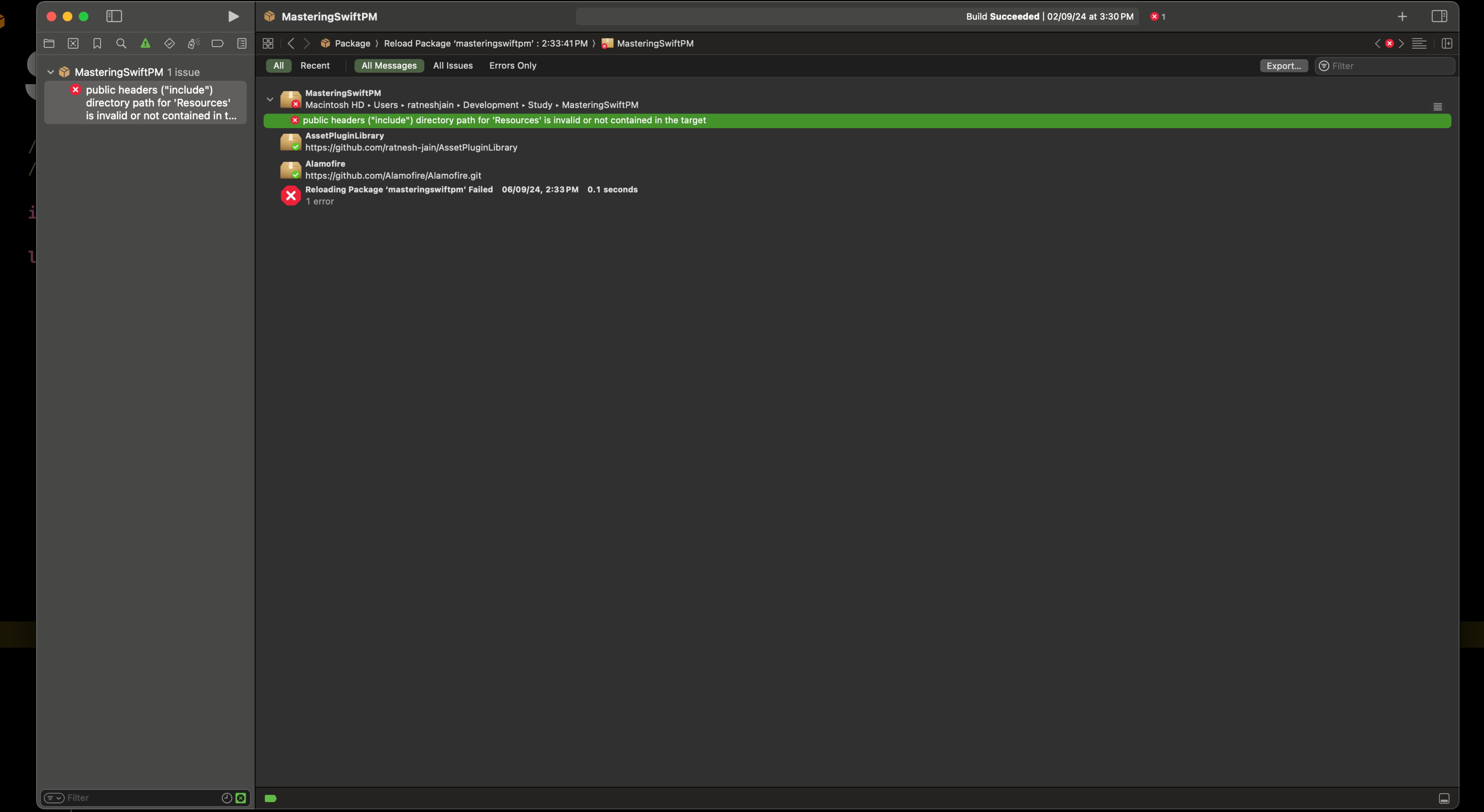
<https://github.com/ratnesh-jain/AssetPluginLibrary>

Alamofire



<https://github.com/Alamofire/Alamofire.git>

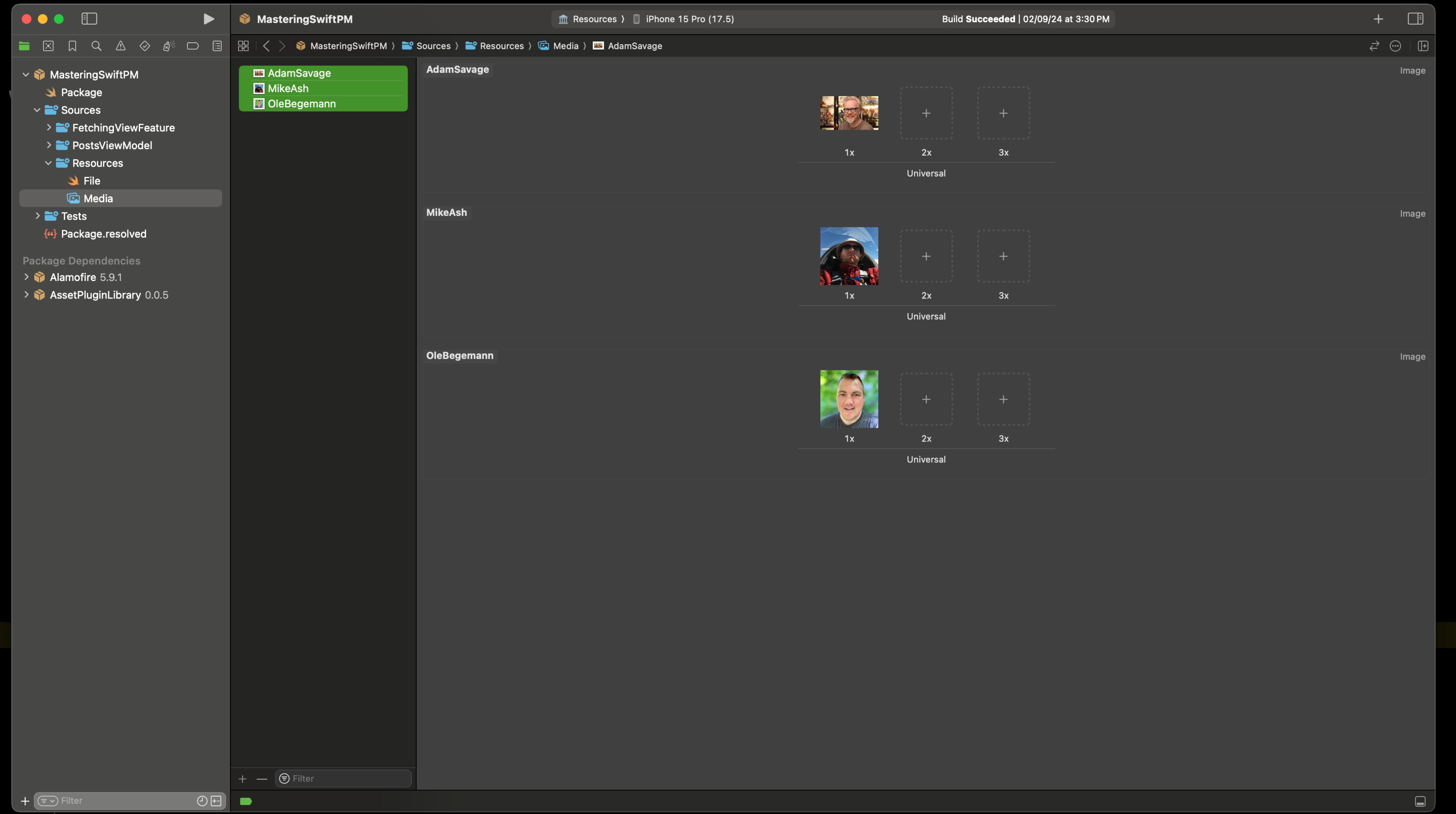
✖ **Reloading Package 'masteringswiftpm' Failed 06/09/24, 2:33 PM 0.1 seconds**
1 error



The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure for "MasteringSwiftPM". The "File" item under "Resources" is selected, highlighted with a green bar.
- File Navigator:** Shows the current file path: "MasteringSwiftPM > Sources > Resources > File > No Selection".
- Editor:** Displays the content of "File.swift".

```
1 //  
2 // File.swift  
3 //  
4 //  
5 // Created by Ratnesh Jain on 06/09/24.  
6 //  
7  
8 import Foundation  
9
```
- Utilities:** Shows package dependencies:
 - > Alamofire 5.9.1
 - > AssetPluginLibrary 0.0.5
- Bottom Bar:** Includes buttons for "Filter", "Search", and "Line: 1 Col: 1".





Sharing Resources

```
import Foundation
import SwiftUI

struct InfluencersView: View {
    var body: some View {
        List {
            Image(.adamSavage)
            Image(.mikeAsh)
            Image(.oleBegemann)
        }
    }
}

#Preview {
    InfluencersView()
}
```

Assets





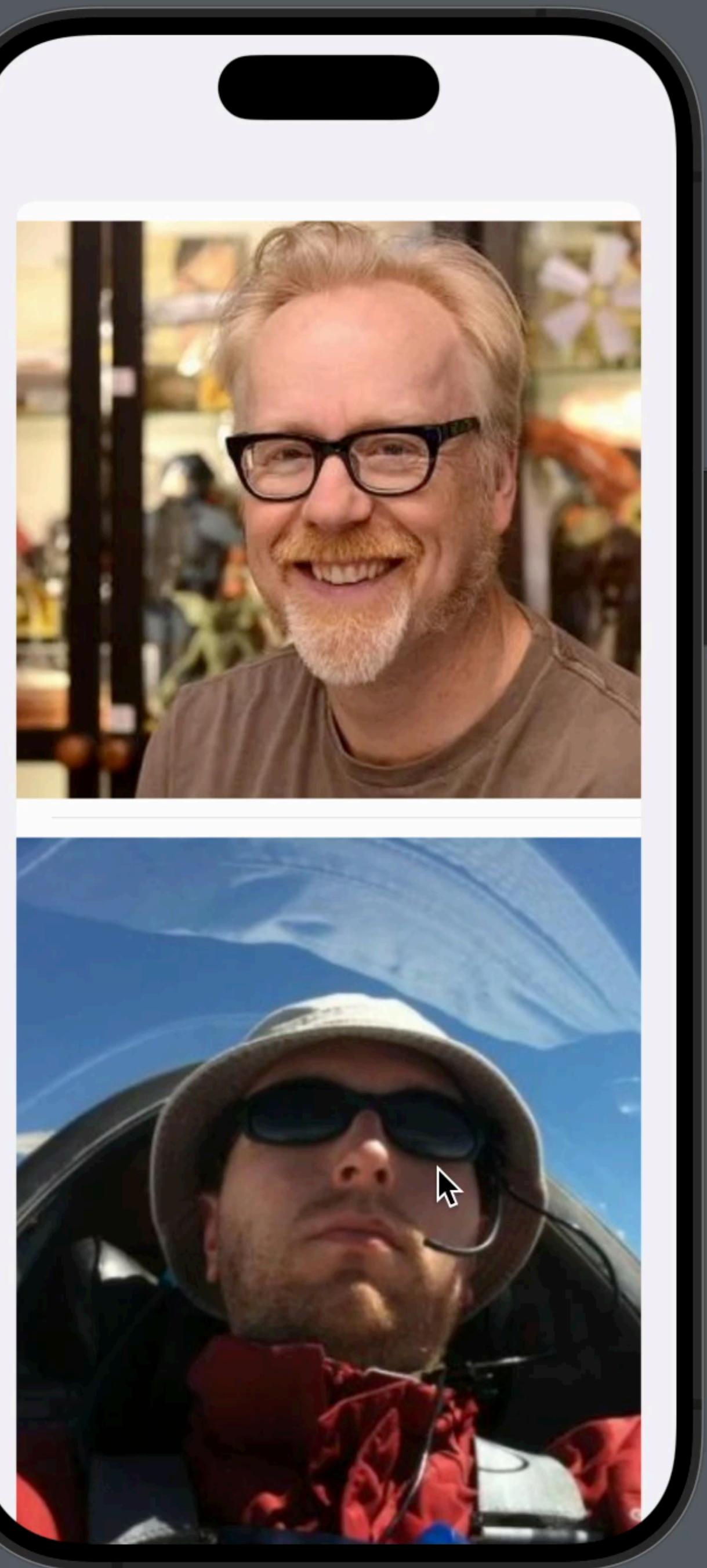
Sharing Resources

```
import Foundation
import SwiftUI

struct InfluencersView: View {
    var body: some View {
        List {
            Image(.adamSavage)
            Image(.mikeAsh)
            Image(.)
        }
    }
}

#Preview {
    InfluencersView()
}
```

M adamSavage
M mikeAsh
M oleBegemann
M actions
M add
M checkmark
M remove
M strokedCheckmark
M zero
adamSavage: ImageResource
The "AdamSavage" asset catalog image resource.



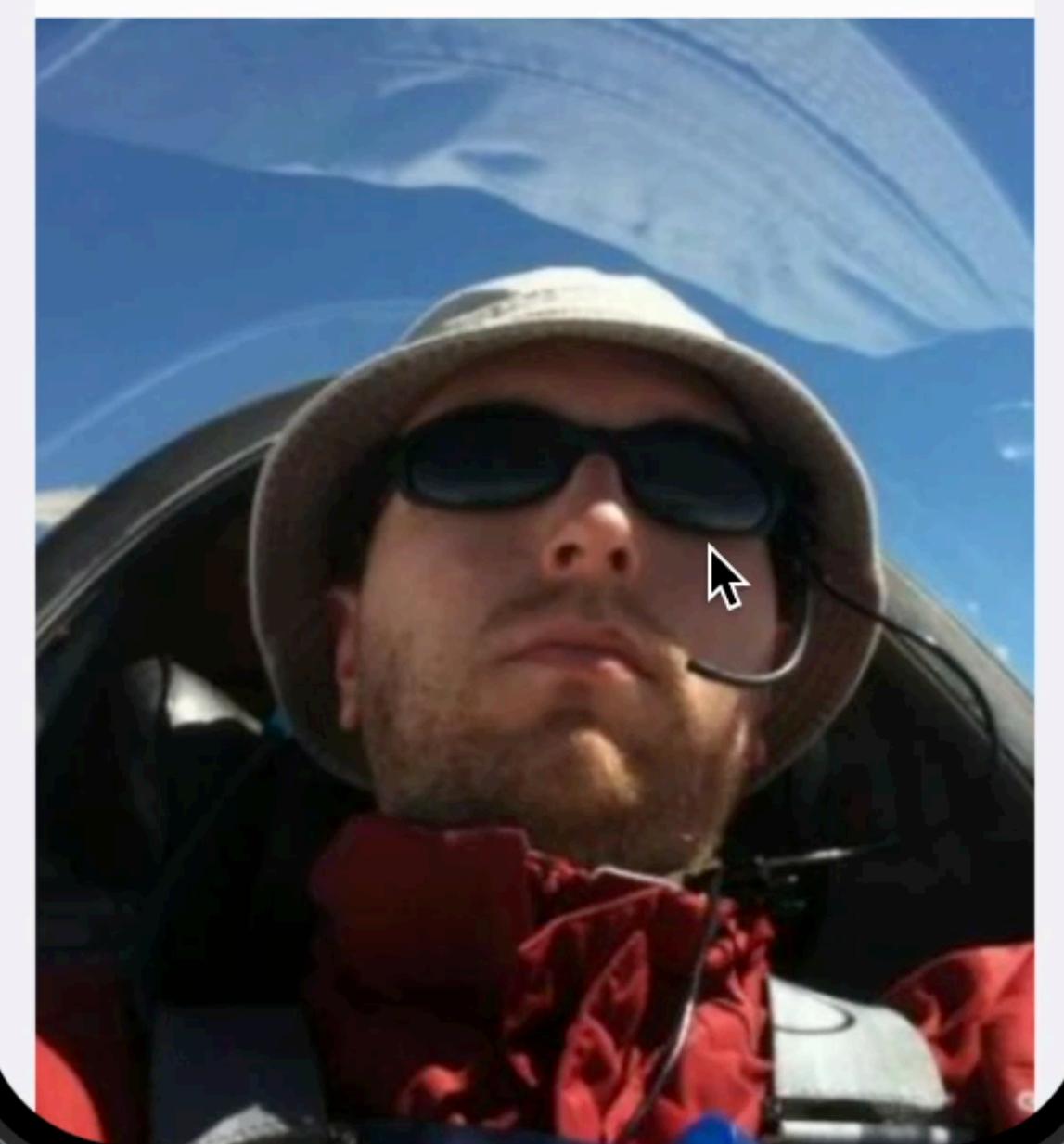


Sharing Resources

```
import Foundation
import SwiftUI

struct InfluencersView: View {
    var body: some View {
        List {
            Image(.adamSavage)
            Image(.mikeAsh)
            Image(.oleBegemann)
        }
    }
}

#Preview {
    InfluencersView()
}
```



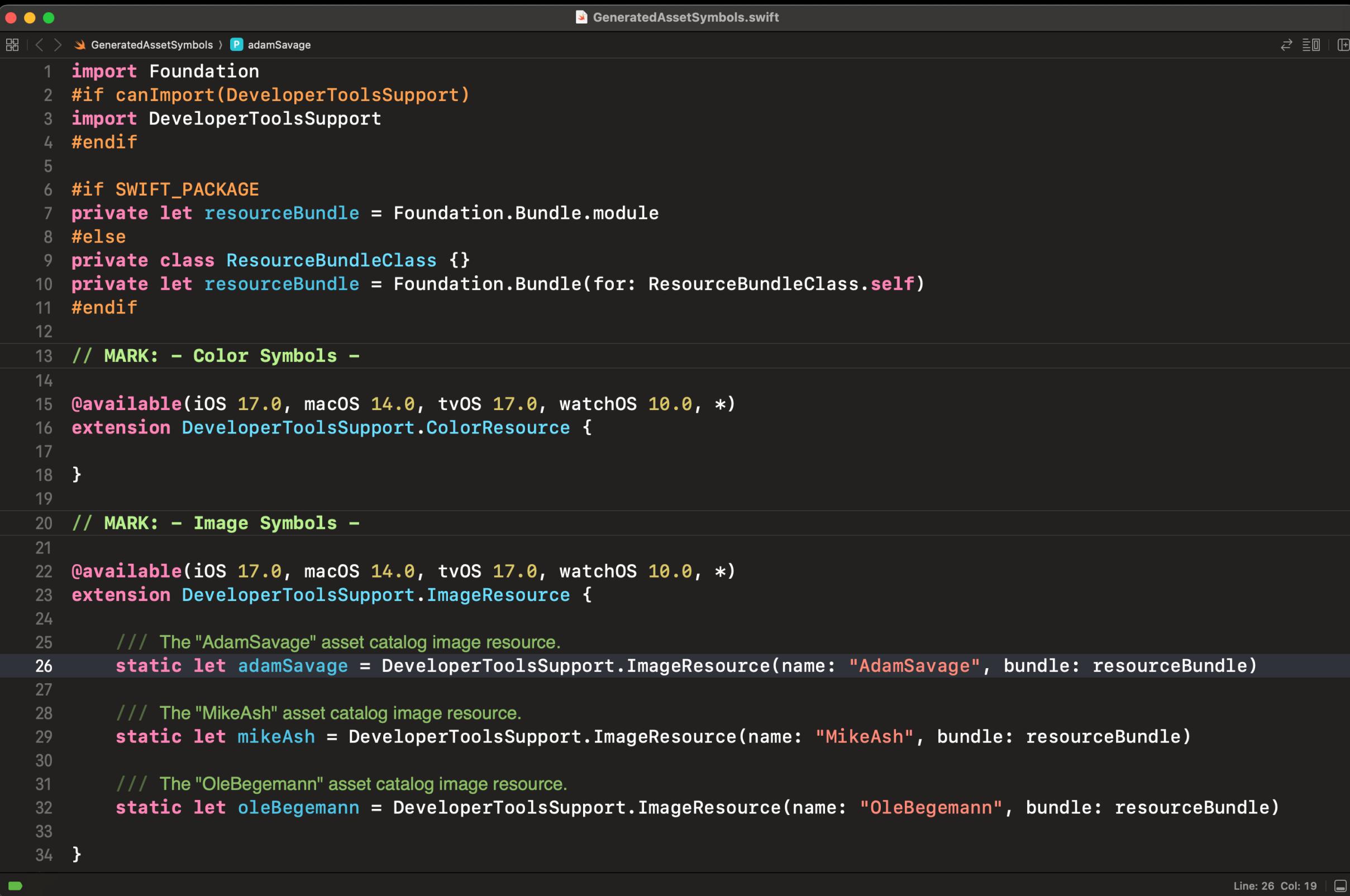


Sharing Resources

Assets



DerivedData/MasteringSwiftPM-bgofwlsbcmvvlaocnlrajdewtnq/Build/Intermediates.noindex/Previews/iphonesimulator/Resources/Intermediates.noindex/MasteringSwiftPM.build/Debug-iphonesimulator/Resources.build/DerivedSources/GeneratedAssetSymbols.swift



The screenshot shows a dark-themed Xcode code editor window. The title bar reads "GeneratedAssetSymbols.swift". The file content is a Swift script that imports Foundation and DeveloperToolsSupport, and defines static let variables for three asset catalog image resources: "AdamSavage", "MikeAsh", and "OleBegemann". The code is as follows:

```
1 import Foundation
2 #if canImport(DeveloperToolsSupport)
3 import DeveloperToolsSupport
4 #endif
5
6 #if SWIFT_PACKAGE
7 private let resourceBundle = Foundation.Bundle.module
8 #else
9 private class ResourceBundleClass {}
10 private let resourceBundle = Foundation.Bundle(for: ResourceBundleClass.self)
11 #endif
12
13 // MARK: - Color Symbols -
14
15 @available(iOS 17.0, macOS 14.0, tvOS 17.0, watchOS 10.0, *)
16 extension DeveloperToolsSupport.ColorResource {
17 }
18
19 // MARK: - Image Symbols -
20
21 @available(iOS 17.0, macOS 14.0, tvOS 17.0, watchOS 10.0, *)
22 extension DeveloperToolsSupport.ImageResource {
23
24     /// The "AdamSavage" asset catalog image resource.
25     static let adamSavage = DeveloperToolsSupport.ImageResource(name: "AdamSavage", bundle: resourceBundle)
26
27     /// The "MikeAsh" asset catalog image resource.
28     static let mikeAsh = DeveloperToolsSupport.ImageResource(name: "MikeAsh", bundle: resourceBundle)
29
30     /// The "OleBegemann" asset catalog image resource.
31     static let oleBegemann = DeveloperToolsSupport.ImageResource(name: "OleBegemann", bundle: resourceBundle)
32
33 }
34
```

Line: 26 Col: 19

```
4 #endif
5
6 #if SWIFT_PACKAGE
7 private let resourceBundle = Foundation.Bundle.module
8 #else
9 private class ResourceBundleClass {}
10 private let resourceBundle = Foundation.Bundle(for: ResourceBundleClass.self)
11 #endif
12
13 // MARK: - Color Symbols -
14
15 @available(iOS 17.0, macOS 14.0, tvOS 17.0, watchOS 10.0, *)
16 extension DeveloperToolsSupport.ColorResource {
17
18 }
19
20 // MARK: - Image Symbols -
21
22 @available(iOS 17.0, macOS 14.0, tvOS 17.0, watchOS 10.0, *)
23 extension DeveloperToolsSupport.ImageResource {
24
25     /// The "AdamSavage" asset catalog image resource.
26     static let adamSavage = DeveloperToolsSupport.ImageResource(name: "AdamSavage", bundle: resourceBundle)
27
28     /// The "MikeAsh" asset catalog image resource.
29     static let mikeAsh = DeveloperToolsSupport.ImageResource(name: "MikeAsh", bundle: resourceBundle)
30
31     /// The "OleBegemann" asset catalog image resource.
32     static let oleBegemann = DeveloperToolsSupport.ImageResource(name: "OleBegemann", bundle: resourceBundle)
33
34 }
```

```
4 #endif
5
6 #if SWIFT_PACKAGE
7 private let resourceBundle = Foundation.Bundle.module
8 #else
9 private class ResourceBundleClass {}
10 private let resourceBundle = Foundation.Bundle(for: ResourceBundleClass.self)
11 #endif
12
13 // MARK: - Color Symbols -
14
15 @available(iOS 17.0, macOS 14.0, tvOS 17.0, watchOS 10.0, *)
16 extension DeveloperToolsSupport.ImageResource {
17 }
18
19
20 // MARK: - Image Symbols -
21 @available(iOS 17.0, macOS 14.0, tvOS 17.0, watchOS 10.0, *)
22 extension DeveloperToolsSupport.ImageResource {
23
24     /// The "AdamSavage" asset catalog image resource.
25     static let adamSavage = DeveloperToolsSupport.ImageResource(name: "AdamSavage", bundle: resourceBundle)
26
27
28     /// The "MikeAsh" asset catalog image resource.
29     static let mikeAsh = DeveloperToolsSupport.ImageResource(name: "MikeAsh", bundle: resourceBundle)
30
31     /// The "OleBegemann" asset catalog image resource.
32     static let oleBegemann = DeveloperToolsSupport.ImageResource(name: "OleBegemann", bundle: resourceBundle)
33
34 }
```

Auto generated Image resource does not mark “Public” access control



Sharing Resources

Assets



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



Sharing Resources

Assets



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

The screenshot shows the Xcode interface with the project "MasteringSwiftPM" open. The left sidebar displays the project structure, including "Sources" containing "FetchingViewFeature" and "PostsViewModel", and "Resources" containing "File" and "Media". The "Tests" folder and "Package.resolved" file are also listed. The "Package Dependencies" section shows "Alamofire 5.9.1" and "AssetPluginLibrary 0.0.5". The "Package" item in the sidebar is highlighted with a green background.

The main editor area shows the `Package.swift` file for the `AssetPluginLibrary` package. The code defines a package named "AssetPluginLibrary" with platforms .iOS(.v13), .macOS(.v12), products including a plugin for "Generate Asset Resources", dependencies (none listed), and targets including a command-line plugin for generating asset resources.

```
// swift-tools-version: 5.6
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "AssetPluginLibrary",
    platforms: [
        .iOS(.v13), .macOS(.v12)],
    products: [
        .plugin(name: "AssetResourcePlugin", targets: ["Generate Asset Resources"])
    ],
    dependencies: [
        // Dependencies declare other packages that this package depends on.
        // .package(url: /* package url */, from: "1.0.0"),
    ],
    targets: [
        // Targets are the basic building blocks of a package. A target can define a module or a test suite.
        // Targets can depend on other targets in this package, and on products in packages this package depends on.
        .plugin(
            name: "Generate Asset Resources",
            capability: .command(
                intent: .custom(verb: "Generates Assets Resource", description: "Generates assets resource constants"),
                permissions: [.writeToPackageDirectory(reason: "Creates a constant files")]
            ),
            path: "Plugins/AssetResourcePlugin"
        )
    ]
)
```

At the bottom of the editor, there is a partial view of another file showing "dependencies: ["FetchingViewFeature"],]". The status bar at the bottom right indicates "Line: 1 Col: 1".

The screenshot shows the Xcode interface with the project 'MasteringSwiftPM' open. The left sidebar displays the project structure, including 'Sources' (FetchingViewFeature, PostsViewModel), 'Resources' (File, Media), and 'Tests'. The 'Package' item in the sidebar is highlighted with a green bar. The main editor area shows the 'Package.swift' file content:

```
// swift-tools-version: 5.6
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "AssetPluginLibrary",
    platforms: [
        .iOS(.v13), .macOS(.v12)],
    products: [
        .plugin(name: "AssetResourcePlugin", targets: ["Generate Asset Resources"]),
    ],
    dependencies: [
        // Dependencies declare other packages that this package depends on.
        // .package(url: /* package url */, from: "1.0.0"),
    ],
    targets: [
        // Targets are the basic building blocks of a package. A target can define a module or a test suite.
        // Targets can depend on other targets in this package, and on products in packages this package depends on.
        .plugin(
            name: "Generate Asset Resources",
            capability: .command(
                intent: .custom(verb: "Generates Assets Resource", description: "Generates assets resource constants"),
                permissions: [.writeToPackageDirectory(reason: "Creates a constant files")]
            ),
            path: "Plugins/AssetResourcePlugin"
        )
    ]
)
```

A yellow oval-shaped highlight surrounds the 'products' section of the code. The status bar at the bottom right indicates 'Line: 1 Col: 1'.

The screenshot shows the Xcode interface with the project 'MasteringSwiftPM' open. The left sidebar displays the project structure, including 'Sources' (FetchingViewFeature, PostsViewModel), 'Resources' (File, Media), and 'Tests'. The 'Package' item in the sidebar is highlighted with a green bar. The main editor area shows the 'Package.swift' file content:

```
// swift-tools-version: 5.6
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "AssetPluginLibrary",
    platforms: [
        .iOS(.v13), .macOS(.v12)],
    products: [
        .plugin(name: "AssetResourcePlugin", targets: ["Generate Asset Resources"]),
    ],
    dependencies: [
        // Dependencies declare other packages that this package depends on.
        // .package(url: /* package url */, from: "1.0.0"),
    ],
    targets: [
        // Targets are the basic building blocks of a package. A target can define a module or a test suite.
        // Targets can depend on other targets in this package, and on products in packages this package depends on.
        .plugin(
            name: "Generate Asset Resources",
            capability: .command(
                intent: .custom(verb: "Generates Assets Resource", description: "Generates assets resource constants"),
                permissions: [.writeToPackageDirectory(reason: "Creates a constant files")]
            ),
            path: "Plugins/AssetResourcePlugin"
        )
    ]
)
```

A yellow oval highlights the line `.plugin(name: "AssetResourcePlugin", targets: ["Generate Asset Resources"])`. The status bar at the bottom right shows 'Line: 1 Col: 1'.

The screenshot shows the Xcode interface with the project 'MasteringSwiftPM' selected in the top bar. The main editor area displays the `Package.swift` file. A context menu is open over the file, with the 'Generate Asset Resources' option highlighted.

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
        .library(
            name: "Resources",
            targets: ["Resources"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"))
    ],
    targets: [
        .target(name: "Resources"),
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```

The code in the `Package.swift` file defines a Swift package named 'MasteringSwiftPM'. It specifies the minimum Swift version required (5.10), lists two product libraries ('FetchingViewFeature' and 'Resources'), and includes dependencies for 'Alamofire' and 'AssetPluginLibrary'. The 'FetchingViewFeature' library has a dependency on 'Alamofire'. The 'MasteringSwiftPMTests' test target depends on the 'FetchingViewFeature' library.

MasteringSwiftPM

MasteringSwiftPM > iPhone 15 Pro (17.5)

Build Succeeded | 02/09/24 at 3:30 PM

MasteringSwiftPM > Package > No Selection

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]
        ),
        .dependencies: [
            .package(
                url: "https://github.com/Alamofire/Alamofire.git",
                .upToNextMajor(from: "5.0.0")
            ),
            .package(
                url: "https://github.com/Alamofire/AlamofireImage.git",
                .upToNextMajor(from: "5.0.0")
            )
        ]
    ],
    targets: [
        .target(name: "Resources"),
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"])
    ]
)
```

Generate Asset Resources:

Select one or more inputs to pass to the 'Generate Asset Resources' command.

< MasteringSwiftPM

- FetchingViewFeature
- MasteringSwiftPMTests

Arguments

Cancel Run

Line: 12 Col: 47



Sharing Resources

Assets



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")
        ),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"
            )
        )
    ],
    targets: [
        .target(
            name: "FetchingViewFeature"
        ),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: ["FetchingViewFeature"]),
    ]
)
```



Sharing Resources

Assets



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
        .library(
            name: "Resources",
            targets: ["Resources"])
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5"))
    ]
),
targets: [
    .target(name: "Resources"),
    .target(
        name: "FetchingViewFeature",
        dependencies: [.product(name: "Alamofire", package: "Alamofire")])
]
```

MasteringSwiftPM

MasteringSwiftPM > iPhone 15 Pro (17.5)

Build Succeeded | 02/09/24 at 3:30 PM

MasteringSwiftPM > Package > No Selection

```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
        .library(
            name: "Resources",
            targets: ["Resources"]),
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.9.1")),
        .package(
            url: "https://github.com/Alamofire/AlamofireImage.git",
            .upToNextMajor(from: "5.0.0")),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")])
    ],
    targets: [
        .target(name: "Resources"),
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")]),
        .testTarget(
            name: "MasteringSwiftPMTests",
            dependencies: [.product(name: "AlamofireImage", package: "AlamofireImage")])
    ]
)

name: "FetchingViewFeature",
dependencies: [.product(name: "Alamofire", package: "Alamofire")]
```

Generate Asset Resources:

Select one or more inputs to pass to the 'Generate Asset Resources' command.

- MasteringSwiftPM
 - FetchingViewFeature
 - MasteringSwiftPMTests
 - Resources**

Arguments

Cancel Run

Line: 27 Col: 10



Sharing Resources

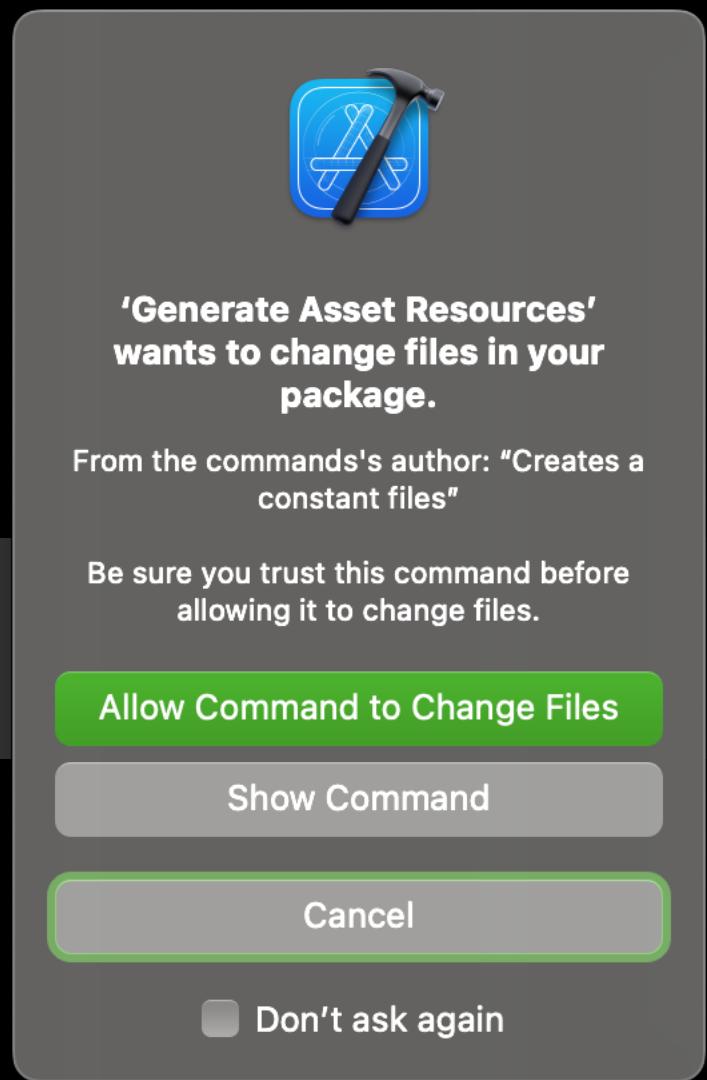
Assets



```
// swift-tools-version: 5.10
// The swift-tools-version declares the minimum version of Swift required to build this package.

import PackageDescription

let package = Package(
    name: "MasteringSwiftPM",
    platforms: [.iOS(.v17)],
    products: [
        .library(
            name: "FetchingViewFeature",
            targets: ["FetchingViewFeature"]),
        .library(
            name: "Resources",
            targets: ["Resources"])
    ],
    dependencies: [
        .package(
            url: "https://github.com/Alamofire/Alamofire.git",
            .upToNextMajor(from: "5.0.0")),
        .package(
            url: "https://github.com/ratnesh-jain/AssetPluginLibrary",
            .upToNextMajor(
                from: "0.0.5")
        )
    ],
    targets: [
        .target(name: "Resources"),
        .target(
            name: "FetchingViewFeature",
            dependencies: [.product(name: "Alamofire", package: "Alamofire")])
    ]
)
```



The screenshot shows the Xcode interface with the project "MasteringSwiftPM" open. The AssetResource file is selected in the sidebar, highlighted with a green bar. The main editor area displays the following Swift code:

```
27
28 extension ImageResource {
29     public init(asset: ImageAssetResource) {
30         self.init(name: asset.name, bundle: asset.bundle)
31     }
32 }
33
34 @available(iOS 13.0, *)
35 extension UIImage {
36     public convenience init(asset: ImageAssetResource) {
37         self.init(named: asset.name, in: asset.bundle, with: nil)!
38     }
39 }
40
41 #if canImport(SwiftUI)
42 import SwiftUI
43
44 @available(iOS 13.0, *)
45 extension Image {
46     public init(asset: ImageAssetResource) {
47         self.init(asset.name, bundle: asset.bundle)
48     }
49 }
50 #endif
51
52 @MainActor public extension ImageAssetResource {
53     /// The "OleBegemann" asset catalog image resource.
54     static let oleBegemann = Self.init(name: "OleBegemann", bundle: resourceBundle)
55
56     /// The "AdamSavage" asset catalog image resource.
57     static let adamSavage = Self.init(name: "AdamSavage", bundle: resourceBundle)
58
59     /// The "MikeAsh" asset catalog image resource.
60     static let mikeAsh = Self.init(name: "MikeAsh", bundle: resourceBundle)
61
62 }
63
```

At the bottom of the editor, there is a note: "Ratnesh Jain". The status bar at the bottom right shows "11 characters".

The screenshot shows the Xcode interface with the project "MasteringSwiftPM" open. The file "AssetResource.swift" is selected in the left sidebar, which is highlighted with a green bar. The code editor displays the following Swift code:

```
27
28 extension ImageResource {
29     public init(asset: ImageAssetResource) {
30         self.init(name: asset.name, bundle: asset.bundle)
31     }
32 }
33
34 @available(iOS 13.0, *)
35 extension UIImage {
36     public convenience init(asset: ImageAssetResource) {
37         self.init(named: asset.name, in: asset.bundle, with: nil)!
38     }
39 }
40
41 #if canImport(SwiftUI)
42 import SwiftUI
43
44 @available(iOS 13.0, *)
45 extension Image {
46     public init(asset: ImageAssetResource) {
47         self.init(asset.name, bundle: asset.bundle)
48     }
49 }
50 #endif
51
52 @MainActor public extension ImageAssetResource {
53     /// The "OleBegemann" asset catalog image resource.
54     static let oleBegemann = Self.init(name: "OleBegemann", bundle: resourceBundle)
55
56     /// The "AdamSavage" asset catalog image resource.
57     static let adamSavage = Self.init(name: "AdamSavage", bundle: resourceBundle)
58
59     /// The "MikeAsh" asset catalog image resource.
60     static let mikeAsh = Self.init(name: "MikeAsh", bundle: resourceBundle)
61
62 }
63
```

A yellow box highlights the line `static let oleBegemann = Self.init(name: "OleBegemann", bundle: resourceBundle)`. The status bar at the bottom shows the file name "AssetResource.swift", the device "iPhone 15 Pro (17.5)", and the build status "Build Succeeded | 02/09/24 at 3:30 PM". The bottom right corner indicates "11 characters" and has a small icon.

The screenshot shows the Xcode interface with the project "MasteringSwiftPM" open. The file "AssetResource.swift" is selected in the left sidebar, which is highlighted with a green bar. The code editor displays the following Swift code:

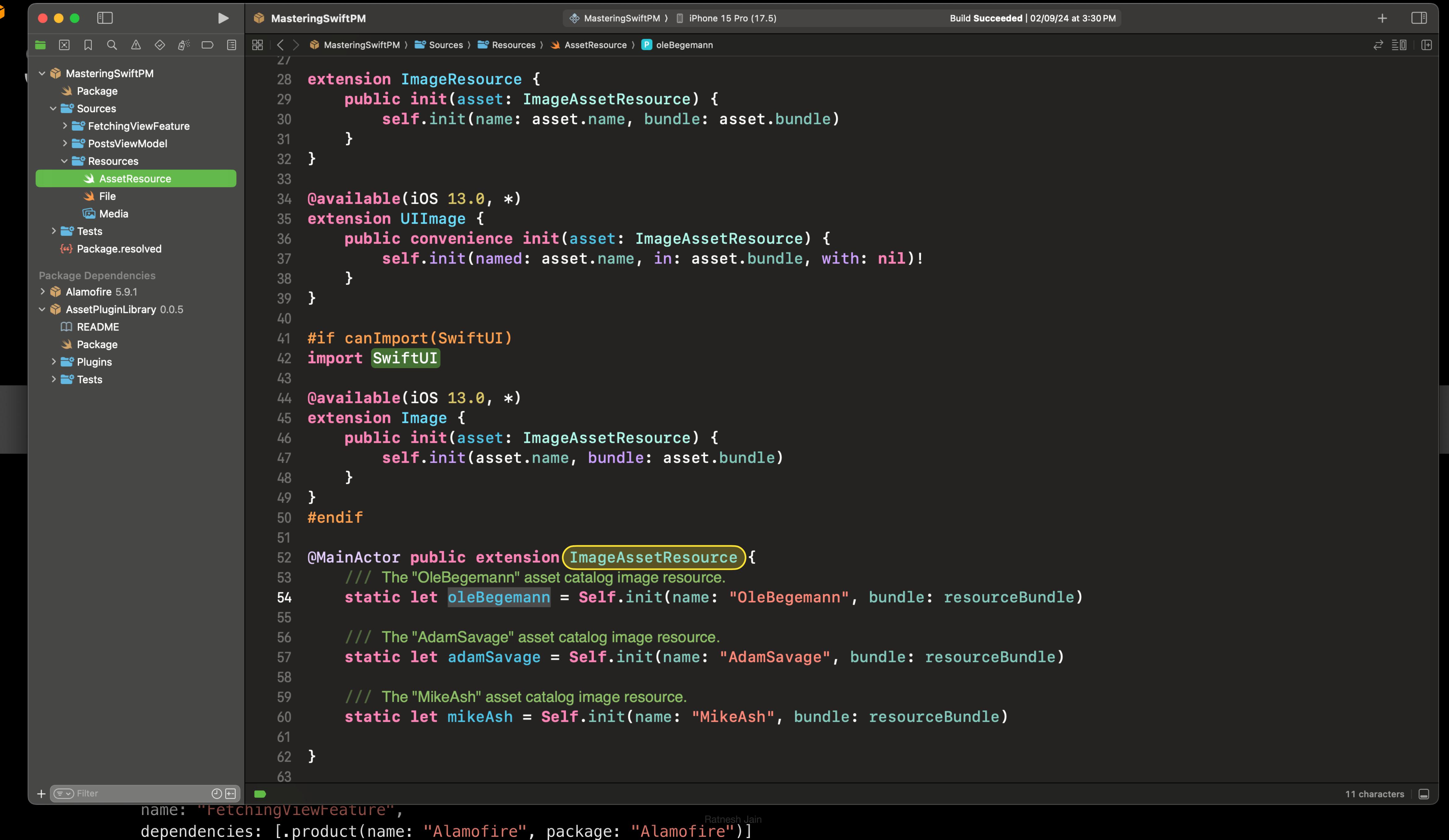
```
27
28 extension ImageResource {
29     public init(asset: ImageAssetResource) {
30         self.init(name: asset.name, bundle: asset.bundle)
31     }
32 }
33
34 @available(iOS 13.0, *)
35 extension UIImage {
36     public convenience init(asset: ImageAssetResource) {
37         self.init(named: asset.name, in: asset.bundle, with: nil)!
38     }
39 }
40
41 #if canImport(SwiftUI)
42 import SwiftUI
43
44 @available(iOS 13.0, *)
45 extension Image {
46     public init(asset: ImageAssetResource) {
47         self.init(asset.name, bundle: asset.bundle)
48     }
49 }
50 #endif
51
52 @MainActor public extension ImageAssetResource {
53     /// The "OleBegemann" asset catalog image resource.
54     static let oleBegemann = Self.init(name: "OleBegemann", bundle: resourceBundle)
55
56     /// The "AdamSavage" asset catalog image resource.
57     static let adamSavage = Self.init(name: "AdamSavage", bundle: resourceBundle)
58
59     /// The "MikeAsh" asset catalog image resource.
60     static let mikeAsh = Self.init(name: "MikeAsh", bundle: resourceBundle)
61
62 }
63
```

The variable "oleBegemann" at line 54 and the variable "adamSavage" at line 57 are highlighted with yellow circles. The status bar at the bottom right shows "Ratnesh Jain".

The screenshot shows the Xcode interface with the project "MasteringSwiftPM" selected. The AssetResource file is open in the main editor area. The code implements extensions for ImageResource, UIImage, and Image to initialize them from asset catalog resources. It includes conditional imports for SwiftUI and uses @MainActor. Three static let statements define OleBegemann, AdamSavage, and MikeAsh resources. The mikeAsh variable is highlighted with a yellow oval.

```
27
28 extension ImageResource {
29     public init(asset: ImageAssetResource) {
30         self.init(name: asset.name, bundle: asset.bundle)
31     }
32 }
33
34 @available(iOS 13.0, *)
35 extension UIImage {
36     public convenience init(asset: ImageAssetResource) {
37         self.init(named: asset.name, in: asset.bundle, with: nil)!
38     }
39 }
40
41 #if canImport(SwiftUI)
42 import SwiftUI
43
44 @available(iOS 13.0, *)
45 extension Image {
46     public init(asset: ImageAssetResource) {
47         self.init(asset.name, bundle: asset.bundle)
48     }
49 }
50 #endif
51
52 @MainActor public extension ImageAssetResource {
53     /// The "OleBegemann" asset catalog image resource.
54     static let oleBegemann = Self.init(name: "OleBegemann", bundle: resourceBundle)
55
56     /// The "AdamSavage" asset catalog image resource.
57     static let adamSavage = Self.init(name: "AdamSavage", bundle: resourceBundle)
58
59     /// The "MikeAsh" asset catalog image resource.
60     static let mikeAsh = Self.init(name: "MikeAsh", bundle: resourceBundle)
61
62 }
63
```

name: "FetchingViewFeature",
dependencies: [.product(name: "Alamofire", package: "Alamofire")]



The screenshot shows the Xcode interface with the project "MasteringSwiftPM" selected. The AssetResource file is open in the main editor area. The code implements extensions for ImageResource, UIImage, and Image to initialize them from an asset catalog. The oleBegemann asset is highlighted with a yellow oval.

```
27
28 extension ImageResource {
29     public init(asset: ImageAssetResource) {
30         self.init(name: asset.name, bundle: asset.bundle)
31     }
32 }
33
34 @available(iOS 13.0, *)
35 extension UIImage {
36     public convenience init(asset: ImageAssetResource) {
37         self.init(named: asset.name, in: asset.bundle, with: nil)!
38     }
39 }
40
41 #if canImport(SwiftUI)
42 import SwiftUI
43
44 @available(iOS 13.0, *)
45 extension Image {
46     public init(asset: ImageAssetResource) {
47         self.init(asset.name, bundle: asset.bundle)
48     }
49 }
50 #endif
51
52 @MainActor public extension ImageAssetResource {
53     /// The "OleBegemann" asset catalog image resource.
54     static let oleBegemann = Self.init(name: "OleBegemann", bundle: resourceBundle)
55
56     /// The "AdamSavage" asset catalog image resource.
57     static let adamSavage = Self.init(name: "AdamSavage", bundle: resourceBundle)
58
59     /// The "MikeAsh" asset catalog image resource.
60     static let mikeAsh = Self.init(name: "MikeAsh", bundle: resourceBundle)
61
62 }
63
```

At the bottom of the editor, there is a note: "name: "FetchingViewFeature", dependencies: [.product(name: "Alamofire", package: "Alamofire")]" and a small note "Ratnesh Jain".

The screenshot shows the Xcode interface with the project "MasteringSwiftPM" selected. The AssetResource.swift file is open in the main editor area. The code implements extensions for ImageResource, UIImage, and Image to handle image assets from a catalog. A specific asset, "oleBegemann", is highlighted with a yellow oval. The status bar at the top right indicates "Build Succeeded | 02/09/24 at 3:30 PM".

```
27
28 extension ImageResource {
29     public init(asset: ImageAssetResource) {
30         self.init(name: asset.name, bundle: asset.bundle)
31     }
32 }
33
34 @available(iOS 13.0, *)
35 extension UIImage {
36     public convenience init(asset: ImageAssetResource) {
37         self.init(named: asset.name, in: asset.bundle, with: nil)!
38     }
39 }
40
41 #if canImport(SwiftUI)
42 import SwiftUI
43
44 @available(iOS 13.0, *)
45 extension Image {
46     public init(asset: ImageAssetResource) {
47         self.init(asset.name, bundle: asset.bundle)
48     }
49 }
50 #endif
51
52 @MainActor public extension ImageAssetResource {
53     /// The "OleBegemann" asset catalog image resource.
54     static let oleBegemann = Self.init(name: "OleBegemann", bundle: resourceBundle)
55
56     /// The "AdamSavage" asset catalog image resource.
57     static let adamSavage = Self.init(name: "AdamSavage", bundle: resourceBundle)
58
59     /// The "MikeAsh" asset catalog image resource.
60     static let mikeAsh = Self.init(name: "MikeAsh", bundle: resourceBundle)
61
62 }
63
```

At the bottom of the editor, there is a note: "name: \"FetchingViewFeature\", dependencies: [.product(name: \"Alamofire\", package: \"Alamofire\")]" and a small note "Ratnesh Jain".

MasteringSwiftPM

MasteringSwiftPM > iPhone 15 Pro (17.5)

Build Succeeded | 02/09/24 at 3:30 PM

```
27
28 extension ImageResource {
29     public init(asset: ImageAssetResource) {
30         self.init(name: asset.name, bundle: asset.bundle)
31     }
32 }
33
34 @available(iOS 13.0, *)
35 extension UIImage {
36     public convenience init(asset: ImageAssetResource) {
37         self.init(named: asset.name, in: asset.bundle, with: nil)!
38     }
39 }
40
41 #if canImport(SwiftUI)
42 import SwiftUI
43
44 @available(iOS 13.0, *)
45 extension Image {
46     public init(asset: ImageAssetResource) {
47         self.init(asset.name, bundle: asset.bundle)
48     }
49 }
50 #endif
51
52 @MainActor public extension ImageAssetResource {
53     /// The "OleBegemann" asset catalog image resource.
54     static let oleBegemann = Self.init(name: "OleBegemann", bundle: resourceBundle)
55
56     /// The "AdamSavage" asset catalog image resource.
57     static let adamSavage = Self.init(name: "AdamSavage", bundle: resourceBundle)
58
59     /// The "MikeAsh" asset catalog image resource.
60     static let mikeAsh = Self.init(name: "MikeAsh", bundle: resourceBundle)
61
62 }
63
```

+ Filter

name: "FetchingViewFeature",

dependencies: [.product(name: "Alamofire", package: "Alamofire")]

Ratnesh Jain

The screenshot shows the Xcode IDE interface with the following details:

- Project Navigator:** Shows the project structure for "MasteringSwiftPM". The "AssetResource" file is selected and highlighted with a green background.
- Editor:** Displays the Swift code for "AssetResource". The code defines extensions for `ImageResource`, `UIImage`, and `Image` to initialize from asset catalog resources. It includes imports for `SwiftUI` and `Alamofire`. A specific static let `oleBegemann` is defined for the "OleBegemann" asset.
- Build Bar:** Shows "Build Succeeded" at 02/09/24 at 3:30 PM.
- Bottom Bar:** Includes a "Filter" button, a circular progress bar, and the text "11 characters".

```
27
28 extension ImageResource {
29     public init(asset: ImageAssetResource) {
30         self.init(name: asset.name, bundle: asset.bundle)
31     }
32 }
33
34 @available(iOS 13.0, *)
35 extension UIImage {
36     public convenience init(asset: ImageAssetResource) {
37         self.init(named: asset.name, in: asset.bundle, with: nil)!
38     }
39 }
40
41 #if canImport(SwiftUI)
42 import SwiftUI
43
44 @available(iOS 13.0, *)
45 extension Image {
46     public init(asset: ImageAssetResource) {
47         self.init(asset.name, bundle: asset.bundle)
48     }
49 }
50 #endif
51
52 @MainActor public extension ImageAssetResource {
53     /// The "OleBegemann" asset catalog image resource.
54     static let oleBegemann = Self.init(name: "OleBegemann", bundle: resourceBundle)
55
56     /// The "AdamSavage" asset catalog image resource.
57     static let adamSavage = Self.init(name: "AdamSavage", bundle: resourceBundle)
58
59     /// The "MikeAsh" asset catalog image resource.
60     static let mikeAsh = Self.init(name: "MikeAsh", bundle: resourceBundle)
61
62 }
63
```

name: "FetchingViewFeature",
dependencies: [.product(name: "Alamofire", package: "Alamofire")]

Ratnesh Jain

The screenshot shows the Xcode IDE interface with the following details:

- Project Navigator:** Shows the project structure for "MasteringSwiftPM". The "AssetResource" file under the "Resources" folder is selected and highlighted in green.
- Editor:** Displays the Swift code for "AssetResource". The code defines extensions for `ImageResource`, `UIImage`, and `Image` to initialize them from an `ImageAssetResource`. It also includes static let declarations for three asset catalog resources: `oleBegemann`, `adamSavage`, and `mikeAsh`.
- Build Bar:** Shows "Build Succeeded | 02/09/24 at 3:30 PM".
- Bottom Bar:** Includes a "Filter" button, a circular progress bar, and the text "11 characters".

```
27
28 extension ImageResource {
29     public init(asset: ImageAssetResource) {
30         self.init(name: asset.name, bundle: asset.bundle)
31     }
32 }
33
34 @available(iOS 13.0, *)
35 extension UIImage {
36     public convenience init(asset: ImageAssetResource) {
37         self.init(named: asset.name, in: asset.bundle, with: nil)!
38     }
39 }
40
41 #if canImport(SwiftUI)
42 import SwiftUI
43
44 @available(iOS 13.0, *)
45 extension Image {
46     public init(asset: ImageAssetResource) {
47         self.init(asset.name, bundle: asset.bundle)
48     }
49 }
50#endif
51
52 @MainActor public extension ImageAssetResource {
53     /// The "OleBegemann" asset catalog image resource.
54     static let oleBegemann = Self.init(name: "OleBegemann", bundle: resourceBundle)
55
56     /// The "AdamSavage" asset catalog image resource.
57     static let adamSavage = Self.init(name: "AdamSavage", bundle: resourceBundle)
58
59     /// The "MikeAsh" asset catalog image resource.
60     static let mikeAsh = Self.init(name: "MikeAsh", bundle: resourceBundle)
61
62 }
63
```

name: "FetchingViewFeature",
dependencies: [.product(name: "Alamofire", package: "Alamofire")]

Ratnesh Jain



Sharing Resources With Other Modules

- Assets
- Images
- Colors
- Fonts
- Core Data
- Other Resources



Font.ttf



Sharing Resources

Fonts



The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure for "MasteringSwiftPM". The "Playwrite" folder is selected and highlighted in green.
- File Navigator:** Shows package dependencies:
 - Alamofire 5.9.1
 - AssetPluginLibrary 0.0.5
 - README
 - Package
 - Plugins
 - Tests
- Search Bar:** Displays the path: MasteringSwiftPM > Sources > Fonts > Playwrite > No Selection.
- Build Status:** Build Succeeded | 02/09/24 at 3:30 PM
- Preview Area:** Displays a preview of the shared font "Playwrite". It shows three rows of text:
 - Uppercase letters: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
 - Lowercase letters: a b c d e f g h i j k l m n o p q r s t u v w x y z
 - Numbers: 1 2 3 4 5 6 7 8 9 0



Sharing Resources

Fonts



- Add fonts file to the target directory
- Declare explicit intent for the .ttf font files
 - Creates a Module bundle instance
- Register the font
 - Using `CTFontManagerRegisterGraphicsFont`
 - Using `CTFontManagerRegisterFontsForURL`
- Get the font family name
- Create (typed*) font weights to use with



Sharing Resources



```
extension UIFont {
    static func registerFont(bundle: Bundle, fontName: String, fontExtension: String) -> Bool {
        guard let fontURL = bundle.url(forResource: fontName, withExtension: fontExtension) else {
            print("Couldn't find font \(fontName)")
            return false
        }

        // NOTE: -
        // Previously we were using `CTFontManagerRegisterGraphicsFont` which takes CTFont.
        // but this was not working in the WidgetKit extension.
        // So as per the this thread,
        // https://developer.apple.com/forums/thread/659332
        // Apple engineer suggested to use `CTFontManagerRegisterFontsForURL`.
        // Here scope like `CTFontManagerScope.persistent` is not supported in this method.
        // `CTFontManagerScope.none` should used for un-registering method only.
        // `CTFontManagerScope.process` works for both App and WidgetKit extension Target.

        var error: Unmanaged<CFError>?
        let success = CTFontManagerRegisterFontsForURL(fontURL as CFURL, .process, &error)
        guard success else {
            print(
                """
                Error registering font: \(fontName). Maybe it was already registered.\n
                \(error.map { "\($0.takeUnretainedValue().localizedDescription)" } ?? "")"
                """
            )
            return false
        }

        return true
    }
}
```



Sharing Resources



```
extension UIFont {
    // Should be called from test env.
    static func unregisterFont(bundle: Bundle, fontName: String, fontExtension: String) -> Bool {
        guard let fontURL = bundle.url(forResource: fontName, withExtension: fontExtension) else {
            print("Couldn't find font \(fontName)")
            return false
        }
        guard let fontDataProvider = CGDataProvider(url: fontURL as CFURL) else {
            print("Couldn't load data from the font \(fontName)")
            return false
        }
        guard let font = CGFont(fontDataProvider) else {
            print("Couldn't create font from data")
            return false
        }

        var error: Unmanaged<CFError>?
        let success = CTFontManagerUnregisterGraphicsFont(font, &error)
        guard success else {
            print(
                """
                Error unregistering font: \(fontName). Maybe it was already registered.\n
                \(error.map { "\($0.takeUnretainedValue().localizedDescription)" } ?? "")\n
                """
            )
            return false
        }
        return true
    }
}
```



Sharing Resources



```
extension UIFont {  
    static func allFontNames() -> [String] {  
        let familyNames = UIFont.familyNames.sorted()  
        return familyNames.flatMap { familyName in  
            let fontName = UIFont.fontNames(forFamilyName: familyName)  
            return fontName  
        }  
    }  
}
```



Sharing Resources



```
extension UIFont {  
    static func allFontNames() -> [String] {  
        let familyNames = UIFont.familyNames.sorted()  
        return familyNames.flatMap { familyName in  
            let fontName = UIFont.fontNames(forFamilyName: familyName)  
            return fontName  
        }  
    }  
}
```

["AcademyEngravedLetPlain", "AlNile", "AlNile-Bold", "AmericanTypewriter", "AmericanTypewriter-Light", "AmericanTypewriter-Semibold", "AmericanTypewriter-Bold", "AmericanTypewriter-Condensed", "AmericanTypewriter-CondensedLight", "AmericanTypewriter-CondensedBold", "AppleColorEmoji", "AppleSDGothicNeo-Regular", "AppleSDGothicNeo-Thin", "AppleSDGothicNeo-UltraLight", "AppleSDGothicNeo-Light", "AppleSDGothicNeo-Medium", "AppleSDGothicNeo-SemiBold", "AppleSDGothicNeo-Bold", "AppleSymbols", "ArialMT", "Arial-ItalicMT", "Arial-BoldItalicMT", "ArialHebrew", "ArialHebrew-Light", "ArialHebrew-Bold", "ArialRoundedMTBold", "Avenir-Book", "Avenir-Roman", "Avenir-BookOblique", "Avenir-Oblique", "Avenir-Light", "Avenir-LightOblique", "Avenir-Medium", "Avenir-MediumOblique", "Avenir-Heavy", "Avenir-HeavyOblique", "Avenir-Black", "Avenir-BlackOblique", "AvenirNext-Regular", "AvenirNext-Italic", "AvenirNext-UltraLight", "AvenirNext-UltraLightItalic", "AvenirNext-Medium", "AvenirNext-MediumItalic", "AvenirNext-DemiBold", "AvenirNext-DemiBoldItalic", "AvenirNext-Bold", "AvenirNext-BoldItalic", "AvenirNext-Heavy", "AvenirNext-HeavyItalic", "AvenirNextCondensed-Regular", "AvenirNextCondensed-Italic", "AvenirNextCondensed-UltraLight", "AvenirNextCondensed-UltraLightItalic", "AvenirNextCondensed-Medium", "AvenirNextCondensed-MediumItalic", "AvenirNextCondensed-DemiBold", "AvenirNextCondensed-DemiBoldItalic", "AvenirNextCondensed-Bold", "AvenirNextCondensed-BoldItalic", "AvenirNextCondensed-Heavy", "AvenirNextCondensed-HeavyItalic", "Baskerville", "Baskerville-Italic", "Baskerville-SemiBold", "Baskerville-SemiBoldItalic", "Baskerville-Bold", "Baskerville-BoldItalic", "BodoniSvtyTwoITCTT-Book", "BodoniSvtyTwoITCTT-BookIta", "BodoniSvtyTwoITCTT-Bold", "BodoniSvtyTwoOSITCTT-Book", "BodoniSvtyTwoOSITCTT-BookIt", "BodoniSvtyTwoOSITCTT-Bold", "BodoniSvtyTwoSCITCTT-Book", "BodoniOrnamentsITCTT", "BradleyHandITCTT-Bold", "ChalkboardSE-Regular", "ChalkboardSE-Light", "ChalkboardSE-Bold", "Chalkduster", "Charter-Roman", "Charter-Italic", "Charter-Bold", "Charter-BoldItalic", "Charter-Black", "Charter-BlackItalic", "Cochin", "Cochin-Italic", "Cochin-Bold", "Cochin-BoldItalic", "Copperplate", "Copperplate-Light", "Copperplate-Bold", "CourierNewPSMT", "CourierNewPS-ItalicMT", "CourierNewPS-BoldMT", "CourierNewPS-BoldItalicMT", "DINAlternate-Bold", "DINCondensed-Bold", "Damascus", "DamascusLight", "DamascusMedium", "DamascusSemiBold", "DamascusBold", "DevanagariSangamMN", "DevanagariSangamMN-Bold", "Didot", "Didot-Italic", "Didot-Bold", "EuphemiaUCAS", "EuphemiaUCAS-Italic", "EuphemiaUCAS-Bold", "Farah", "Futura-Medium", "Futura-MediumItalic", "Futura-Bold", "Futura-CondensedMedium", "Futura-CondensedExtraBold", "Galvji", "Galvji-Bold", "GeezaPro", "GeezaPro-Bold", "Georgia", "Georgia-Italic", "Georgia-Bold", "Georgia-BoldItalic", "GillSans", "GillSans-Italic", "GillSans-Light", "GillSans-LightItalic", "GillSans-SemiBold", "GillSans-SemiBoldItalic", "GillSans-Bold", "GillSans-BoldItalic", "GillSans-UltraBold", "GranthaSangamMN-Regular", "GranthaSangamMN-Bold", "Helvetica", "Helvetica-Oblique", "Helvetica-Light", "Helvetica-LightOblique", "Helvetica-Bold", "Helvetica-BoldOblique", "HelveticaNeue", "HelveticaNeue-Italic", "HelveticaNeue-UltraLight", "HelveticaNeue-UltraLightItalic", "HelveticaNeue-Thin", "HelveticaNeue-ThinItalic", "HelveticaNeue-Light", "HelveticaNeue-LightItalic", "HelveticaNeue-Medium", "HelveticaNeue-MediumItalic", "HelveticaNeue-Bold", "HelveticaNeue-BoldItalic", "HelveticaNeue-CondensedBold", "HelveticaNeue-CondensedBlack", "HiraMaruProN-W4", "HiraMinProN-W3", "HiraMinProN-W6", "HiraginoSans-W3", "HiraginoSans-W4", "HiraginoSans-W5", "HiraginoSans-W6", "HiraginoSans-W7", "HiraginoSans-W8", "HoeflerText-Regular", "HoeflerText-Italic", "HoeflerText-Black", "HoeflerText-BlackItalic", "Impact", "Kailasa", "Kailasa-Bold", "Kefa-Regular", "KhmerSangamMN", "KohinoorBangla-Regular", "KohinoorBangla-Light", "KohinoorBangla-Semibold", "KohinoorDevanagari-Regular", "KohinoorDevanagari-Light", "KohinoorDevanagari-Semibold", "KohinoorGujarati-Regular", "KohinoorGujarati-Light", "KohinoorGujarati-Bold", "KohinoorTelugu-Regular", "KohinoorTelugu-Light", "KohinoorTelugu-Medium", "LaoSangamMN", "MalayalamSangamMN", "MalayalamSangamMN-Bold", "MarkerFelt-Thin", "MarkerFelt-Wide", "Menlo-Regular", "Menlo-Italic", "Menlo-Bold", "Menlo-BoldItalic", "DiwanMishafi", "MuktaMahee-Regular", "MuktaMahee-Light", "MuktaMahee-Bold", "MyanmarSangamMN", "MyanmarSangamMN-Bold", "Noteworthy-Light", "Noteworthy-Bold", "NotoNastaliqUrdu", "NotoNastaliqUrdu-Bold", "NotoSansKannada-Regular", "NotoSansKannada-Light", "NotoSansKannada-Bold", "NotoSansMyanmar-Regular", "NotoSansMyanmar-Light", "NotoSansMyanmar-Bold", "NotoSansOriya", "NotoSansOriya-Bold", "NotoSansSyriac-Regular", "NotoSansSyriac-Regular_Thin", "NotoSansSyriac-Regular_ExtraLight", "NotoSansSyriac-Regular_Light", "NotoSansSyriac-Regular_Medium", "NotoSansSyriac-Regular_Semibold", "NotoSansSyriac-Regular_Bold", "NotoSansSyriac-Regular_ExtraBold", "NotoSansSyriac-Regular_Black", "Optima-Regular", "Optima-Italic", "Optima-Bold", "Optima-BoldItalic", "Optima-ExtraBlack", "Palatino-Roman", "Palatino-Italic", "Palatino-Bold", "Palatino-BoldItalic", "Papyrus", "Papyrus-Condensed", "PartyLetPlain", "PingFangHK-Regular", "PingFangHK-Ultralight", "PingFangHK-Thin", "PingFangHK-Light", "PingFangHK-Medium", "PingFangHK-Semibold", "PingFangM0-Regular", "PingFangM0-Ultralight", "PingFangM0-Thin", "PingFangM0-Light", "PingFangM0-Medium", "PingFangM0-Semibold", "PingFangSC-Regular", "PingFangSC-Ultralight", "PingFangSC-Thin", "PingFangSC-Light", "PingFangSC-Medium", "PingFangSC-Semibold", "PingFangTC-Regular", "PingFangTC-Ultralight", "PingFangTC-Thin", "PingFangTC-Light", "PingFangTC-Medium", "PingFangTC-Semibold", "PlaywriteCU-Regular", "PlaywriteCU-Regular_Thin", "PlaywriteCU-Regular_ExtraLight", "PlaywriteCU-Regular_Light", "Rockwell-Regular", "Rockwell-Italic", "Rockwell-Bold", "Rockwell-BoldItalic", "STIXTwoMath-Regular", "STIXTwoText", "STIXTwoText-Italic", "STIXTwoText_Medium", "STIXTwoText-Italic_Medium-Italic", "STIXTwoText_SemiBold", "STIXTwoText-Italic_SemiBold-Italic", "STIXTwoText_Bold", "STIXTwoText-Italic_Bold-Italic", "SavoyeLetPlain", "SinhalaSangamMN", "SinhalaSangamMN-Bold", "SnellRoundhand", "SnellRoundhand-Bold", "SnellRoundhand-Black", "Symbol", "TamilSangamMN", "TamilSangamMN-Bold", "Thonburi", "Thonburi-Light", "Thonburi-Bold", "TimesNewRomanPSMT", "TimesNewRomanPS-ItalicMT", "TimesNewRomanPS-BoldMT", "TimesNewRomanPS-BoldItalicMT", "TrebuchetMS", "TrebuchetMS-Italic", "TrebuchetMS-Bold", "Trebuchet-BoldItalic", "Verdana", "Verdana-Italic", "Verdana-Bold", "Verdana-BoldItalic", "ZapfDingbatsITC", "Zapfino"]



CondensedExtraBold", "Galvji", "Galvji-Bold", "GeezaPro", "GeezaPro-Bold", "Georgia", "Georgia-Italic", "Georgia-Bold", "Georgia-BoldItalic", "GillSans", "GillSans-Italic", "GillSans-Light", "GillSans-LightItalic", "GillSans-SemiBold", "GillSans-SemiBoldItalic", "GillSans-Bold", "GillSans-BoldItalic", "GillSans-UltraBold", "GranthaSangamMN-Regular", "GranthaSangamMN-Bold", "Helvetica", "Helvetica-Oblique", "Helvetica-Light", "Helvetica-LightOblique", "Helvetica-Bold", "Helvetica-BoldOblique", "HelveticaNeue", "HelveticaNeue-Italic", "HelveticaNeue-UltraLight", "HelveticaNeue-UltraLightItalic", "HelveticaNeue-Thin", "HelveticaNeue-ThinItalic", "HelveticaNeue-Light", "HelveticaNeue-LightItalic", "HelveticaNeue-Medium", "HelveticaNeue-MediumItalic", "HelveticaNeue-Bold", "HelveticaNeue-BoldItalic", "HelveticaNeue-CondensedBold", "HelveticaNeue-CondensedBlack", "HiraMaruProN-W4", "HiraMinProN-W3", "HiraMinProN-W6", "HiraginoSans-W3", "HiraginoSans-W4", "HiraginoSans-W5", "HiraginoSans-W6", "HiraginoSans-W7", "HiraginoSans-W8", "HoeflerText-Regular", "HoeflerText-Italic", "HoeflerText-Black", "HoeflerText-BlackItalic", "Impact", "Kailasa", "Kailasa-Bold", "Kefa-Regular", "KhmerSangamMN", "KohinoorBangla-Regular", "KohinoorBangla-Light", "KohinoorBangla-Semibold", "KohinoorDevanagari-Regular", "KohinoorDevanagari-Light", "KohinoorDevanagari-Semibold", "KohinoorGujarati-Regular", "KohinoorGujarati-Light", "KohinoorGujarati-Bold", "KohinoorTelugu-Regular", "KohinoorTelugu-Light", "KohinoorTelugu-Medium", "LaoSangamMN", "MalayalamSangamMN", "MalayalamSangamMN-Bold", "MarkerFelt-Thin", "MarkerFelt-Wide", "Menlo-Regular", "Menlo-Italic", "Menlo-Bold", "Menlo-BoldItalic", "DiwanMishafi", "MuktaMahee-Regular", "MuktaMahee-Light", "MuktaMahee-Bold", "MyanmarSangamMN", "MyanmarSangamMN-Bold", "Noteworthy-Light", "Noteworthy-Bold", "NotoNastaliqUrdu", "NotoNastaliqUrdu-Bold", "NotoSansKannada-Regular", "NotoSansKannada-Light", "NotoSansKannada-Bold", "NotoSansMyanmar-Regular", "NotoSansMyanmar-Light", "NotoSansMyanmar-Bold", "NotoSansOriya", "NotoSansOriya-Bold", "NotoSansSyriac-Regular", "NotoSansSyriac-Regular_Thin", "NotoSansSyriac-Regular_ExtraLight", "NotoSansSyriac-Regular_Light", "NotoSansSyriac-Regular_Medium", "NotoSansSyriac-Regular_SemiBold", "NotoSansSyriac-Regular_Bold", "NotoSansSyriac-Regular_ExtraBold", "NotoSansSyriac-Regular_Black", "Optima-Regular", "Optima-Italic", "Optima-Bold", "Optima-BoldItalic", "Optima-ExtraBlack", "Palatino-Roman", "Palatino-Italic", "Palatino-Bold", "Palatino-BoldItalic", "Papyrus", "Papyrus-Condensed", "PartyLetPlain", "PingFangHK-Regular", "PingFangHK-Ultralight", "PingFangHK-Thin", "PingFangHK-Light", "PingFangHK-Medium", "PingFangHK-Semibold", "PingFangMO-Regular", "PingFangMO-Ultralight", "PingFangMO-Thin", "PingFangMO-Light", "PingFangMO-Medium", "PingFangMO-Semibold", "PingFangSC-Regular", "PingFangSC-Ultralight", "PingFangSC-Thin", "PingFangSC-Light", "PingFangSC-Medium", "PingFangSC-Semibold", "PingFangTC-Regular", "PingFangTC-Ultralight", "PingFangTC-Thin", "PingFangTC-Light", "PingFangTC-Medium", "PingFangTC-Semibold", "PlaywriteCU-Regular", "PlaywriteCU-Regular_Thin", "PlaywriteCU-Regular_ExtraLight", "PlaywriteCU-Regular_Light", "Rockwell-Regular", "Rockwell-Italic", "Rockwell-Bold", "Rockwell-BoldItalic", "STIXTwoMath-Regular", "STIXTwoText", "STIXTwoText-Italic", "STIXTwoText_Medium", "STIXTwoText-Medium-Italic", "STIXTwoText_SemiBold", "STIXTwoText-Italic_SemiBold-Italic", "STIXTwoText-Bold", "STIXTwoText-Italic_Bold-Italic", "SavoyeLetPlain", "SinhalaSangamMN", "SinhalaSangamMN-Bold", "SnellRoundhand", "SnellRoundhand-Bold", "SnellRoundhand-Black", "Symbol", "TamilSangamMN", "TamilSangamMN-Bold", "Thonburi", "Thonburi-Light", "Thonburi-Bold", "TimesNewRomanPSMT", "TimesNewRomanPS-ItalicMT", "TimesNewRomanPS-BoldMT", "TimesNewRomanPS-BoldItalicMT", "TrebuchetMS", "TrebuchetMS-Italic", "TrebuchetMS-Bold", "Trebuchet-BoldItalic", "Verdana", "Verdana-Italic", "Verdana-Bold", "Verdana-BoldItalic", "ZapfDingbatsITC", "Zapfino"]



Sharing Resources



```
extension UIFont {  
    static func allFontNames() -> [String] {  
        let familyNames = UIFont.familyNames.sorted()  
        return familyNames.flatMap { familyName in  
            let fontName = UIFont.fontNames(forFamilyName: familyName)  
            return fontName  
        }  
    }  
}
```

PlaywriteCU-Regular
PlaywriteCU-Regular_Thin
PlaywriteCU-Regular_ExtraLight
PlaywriteCU-Regular_Light



Sharing Resources



```
public enum AppFont: CaseIterable {
    case playWrite

    var fontName: String {
        switch self { case .playWrite: "Playwrite" }
    }

    var fontExtension: String {
        switch self { case .playWrite: ".ttf" }
    }

    public enum PlayWriteWeight: String {
        case regular
        case thin
        case extraLight
        case light

        var name: String {
            switch self {
                case .regular:
                    "PlaywriteCU-Regular"
                case .thin:
                    "PlaywriteCU-Regular_Thin"
                case .extraLight:
                    "PlaywriteCU-Regular_ExtraLight"
                case .light:
                    "PlaywriteCU-Regular_Light"
            }
        }
    }
}
```



Sharing Resources



```
extension UIFont {
    public static func register(_ font: AppFont) -> Bool {
        Self.registerFont(bundle: .module, fontName: font.fontName, fontExtension: font.fontExtension)
    }

    public static func registerAll() {
        AppFont.allCases.forEach {
            _ = Self.register($0)
        }
    }
}
```



Sharing Resources

```
import SwiftUI
import Alamofire
import Fonts

public struct FetchingView: View {
    public var body: some View {
        VStack {
            ProgressView()
            Text("Loading...")
                .font(.playwrite(weight: .light, size: 20))
        }
    }
}

#Preview {
    _ = UIFont.register(.playWrite)
    return FetchingView()
}
```





Sharing Resources With Other Modules

- Assets
- Images
- Colors
- Fonts
- Core Data
- Other Resources



CoreData



Sharing Resources

Core Data



- New directory (folder) in **Sources** directory
- Empty **.swift** file to make SwiftPM happy
- New core Data Model (**.xcdatamodeld**) file
- Restart **Xcode** to add new entities
- Add few **Entities**
- Set codegen to “**Class Definition**”
- Set module to “**Global namespace**”
- Use swift file to Create a **Persistent Container** instance



Sharing Resources

Core Data



Screenshot of the Xcode Core Data editor showing the configuration of the `Author` entity.

Project Navigator: Shows the project structure with `MasteringSwiftPMExampleProject`, `MasteringSwiftPM`, `Sources`, `PersistentModels`, and `Model`.

Editor Area:

- Entity:** `Author` (highlighted in green).
- Attributes:**
 - `id`: UUID
 - `name`: String
- Relationships:**
 - `books`: Destination: Book, Inverse: `authors`
- Fetched Properties:** None listed.

Right-hand pane (Entity details):

- Entity:** Name: `Author`, Abstract Entity: No, Parent Entity: No Parent Entity.
- Class:** Name: `Author`, Module: Global namespace, Codegen: Class Definition.
- Constraints:** No Content.
- Spotlight:** Display Name: Expression.
- User Info:** Key: Value.
- Versioning:** Hash Modifier: Version Hash Modifier, Renaming ID: Renaming Identifier.



Sharing Resources

Core Data



Screenshot of the Xcode Core Data editor showing the Book entity configuration.

Project Navigator: Shows the project structure with the Model file selected.

Entity List: Shows the **Book** entity selected.

Attributes:

Attribute	Type
S desc	String
N edition	Integer 16
I id	UUID
D publishedAt	Date
S title	String

Relationships:

Relationship	Destination	Inverse
O authors	Author	books

Fetched Properties: None listed.

Entity Configuration (Right Panel):

- Entity:** Name: Book, Abstract Entity: No, Parent Entity: No Parent Entity.
- Class:** Name: Book, Module: Global namespace, Codegen: Class Definition.
- Constraints:** No Content.
- Spotlight:** Display Name: Expression.
- User Info:** Key: Value.
- Versioning:** Hash Modifier: Version Hash Modifier, Renaming ID: Renaming Identifier.



Sharing Resources

PersistentModelStore.swift



```
open class PersistentContainer: NSPersistentContainer {

}

public enum PersistentModelStore {
    public static let container: PersistentContainer = {
        guard let modelURL = Bundle.module.url(forResource: "PersistentModels", withExtension: "momd") else { fatalError() }
        guard let model = NSManagedObjectModel(contentsOf: modelURL) else { fatalError() }
        let container = PersistentContainer(name:"PersistentModels", managedObjectModel: model)
        container.loadPersistentStores(completionHandler: { (storeDescription, error) in
            if let error = error as NSError? {
                print("Unresolved error \(error), \(error.userInfo)")
            }
        })
        return container
    }()
}
```



Sharing Resources

BookListViewModel.swift



```
import Foundation
import PersistentModels
import CoreData
import Combine

@MainActor @Observable public class BookListViewModel {
    var bookLists: [Book] = []
    private var cancellables: Set = []

    public init(bookLists: [Book] = []) {
        self.bookLists = bookLists
        performFetchRequest()
        observeChanges()
    }

    func observeChanges() {
        NotificationCenter.default
            .publisher(for: NSManagedObjectContext.didChangeObjectsNotification)
            .receive(on: DispatchQueue.main)
            .sink { [weak self] notification in
                if (notification.userInfo?[NSInsertedObjectsKey] as? Set<NSManagedObject>)?.contains(where: {$0.entity.name == "Book"}) ?? false {
                    self?.performFetchRequest()
                }
            }
            .store(in: &cancellables)
    }

    private func performFetchRequest() {
```



```
        if (notification.userInfo?[NSInsertedObjectsKey] as? Set<NSManagedObject>)?.contains(where: {$0.entity.name == "Book"}) ?? false {
            self?.performFetchRequest()
        }
    }
    .store(in: &cancelables)
}

private func performFetchRequest() {
    let request = Book.fetchRequest()
    let context = PersistentModelStore.container.viewContext
    do {
        let items = try context.fetch(request)
        self.bookLists = items
    } catch {
        print(error)
    }
}

func createNew() throws {
    let context = PersistentModelStore.container.viewContext

    let author = Author(context: context)
    author.id = UUID()
    author.name = "The Author"

    let book = Book(context: context)
    book.id = UUID()
    book.title = "The Book"
    book.desc = "Authorization Book"
    book.edition = 1
    book.publishedAt = Date()
    book.authors = author

    try context.save()
}
```



Sharing Resources With Other Modules

- Assets
- Images
- Colors
- Fonts
- Core Data
- Other Resources



CoreData



Sharing Resources

Other Resources

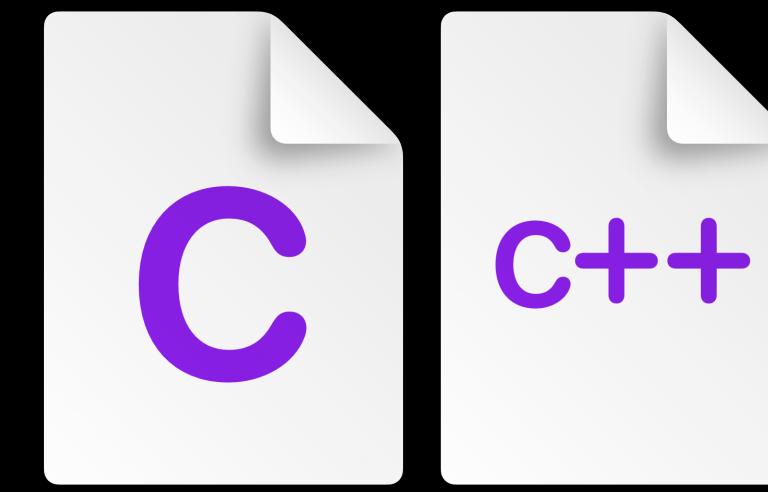




Sharing Resources Supported in SPM



SWIFT



C++



PNG



STORYBOARD



STRINGS



Sharing Resources



Mastering SwiftPM

Today

- Access Modifiers
- Package Manifest
- Package Manager Vocabulary
- Dependencies
- New Feature using SwiftPM
- Integration with Xcode Project
- Sharing Resources
- Dealing with Non-SPM external Dependency



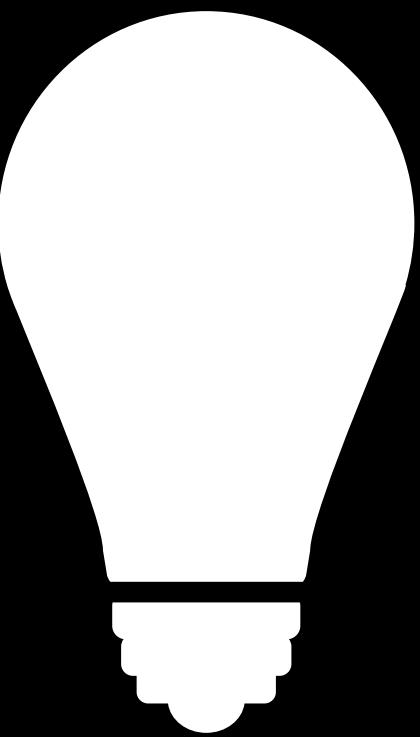
Dealing with Non-SPM external Dependency

- Cocoapods
- Carthage
- Frameworks/SDKs



Dealing with Non-SPM external Dependency

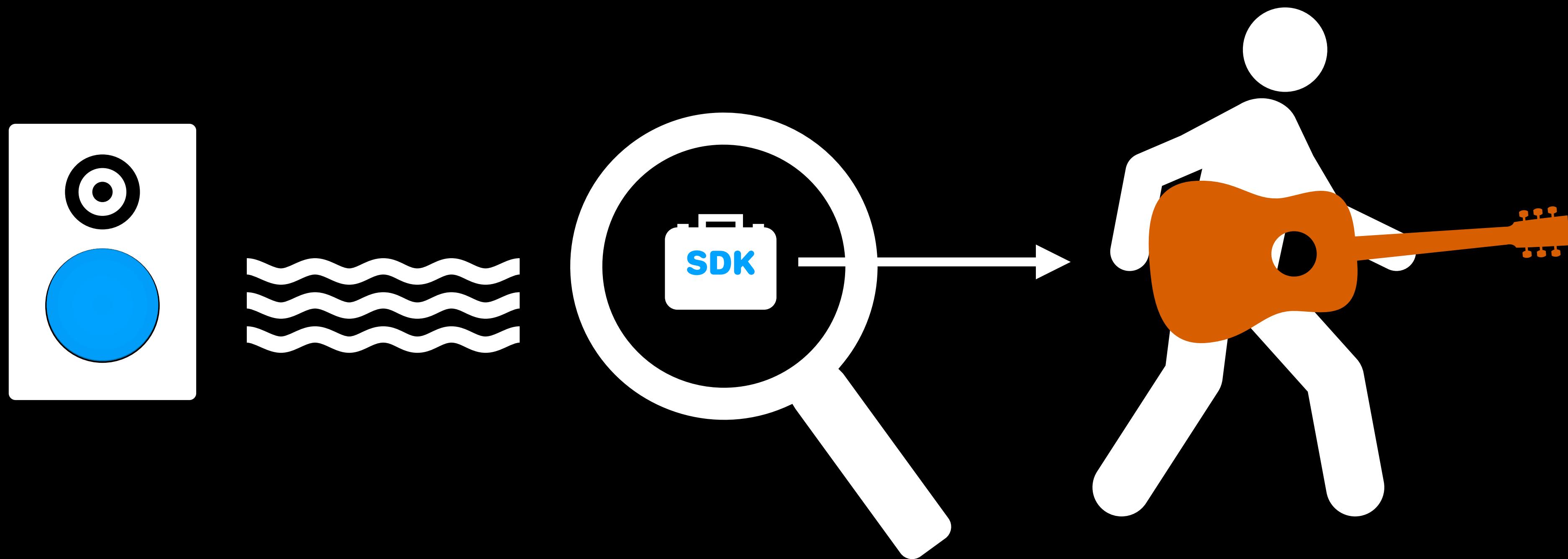
- Understand the feature
- Study the APIs of the dependency
- Study the API with respect to feature
- Use abstraction for APIs
- Protocol or Protocol Witness





Dealing with Non-SPM external Dependency

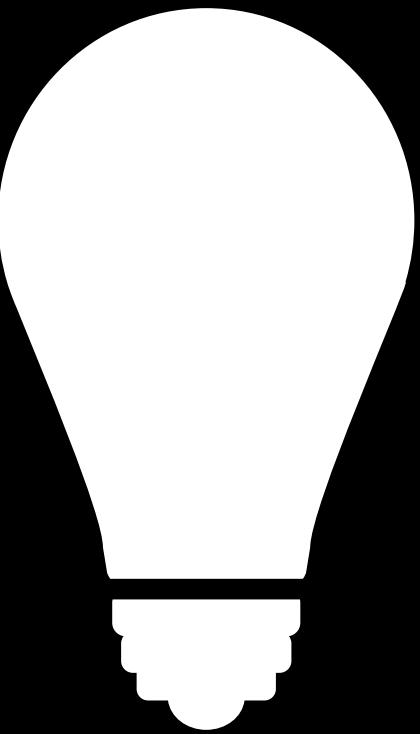
Understand the Feature





Dealing with Non-SPM external Dependency

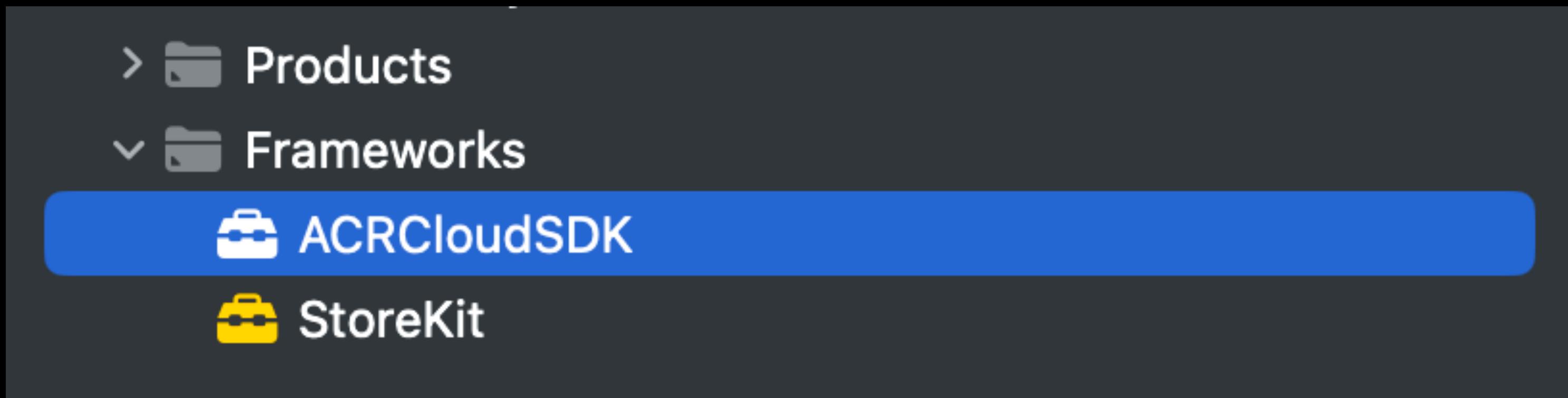
- Understand the feature
- Study the APIs of the dependency
- Study the API with respect to feature
- Use abstraction for APIs
 - Protocol or Protocol Witness
- Use Dependency injection or environment





Dealing with Non-SPM external Dependency

Understand API of Dependency



App-Bridging-Header.h

```
1 //  
2 // Use this file to import your target's public headers that you would like to expose to Swift.  
3 //  
4  
5 #import "ACRCloudConfig.h"  
6 #import "ACRCloudRecognition.h"  
7
```



Dealing with Non-SPM external Dependency

Understand API of Dependency

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

@interface ACRCloudRecognition : NSObject

-(id)initWithConfig:(ACRCloudConfig*)config;

-(void)startPreRecord:(NSInteger)recordTime;
-(void)stopPreRecord;

-(void)startRecordRec;

-(void)stopRecordRec;

-(void)stopRecordAndRec;

//only support RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 8000 Hz
-(NSString*)recognize:(char*)buffer len:(int)len;

//only support RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 8000 Hz
-(NSString*)recognize:(NSData*)pcm_data;

-(void)recognize_fp:(NSData*)fingerprint
    resultBlock:(ACRCloudResultBlock)resultBlock;

-(NSString*)recognize_fp:(NSData*)fingerprint;
-(NSString*)recognize_hum_fp:(NSData*)fingerprint;

//only support RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 8000 Hz
+(NSData*)get_fingerprint:(char*)pcm len:(int)len;
+(NSData*)get_fingerprint:(NSData*)pcm;
+(NSData*)get_hum_fingerprint:(NSData*)pcm;
+(NSData*)get_fingerprint:(char*)pcm;
```



```
//only support RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 8000 Hz
-(NSString*)recognize:(char*)buffer len:(int)len;

//only support RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 8000 Hz
-(NSString*)recognize:(NSData*)pcm_data;

-(void)recognize_fp:(NSData*)fingerprint
    resultBlock:(ACRCloudResultBlock)resultBlock;

-(NSString*)recognize_fp:(NSData*)fingerprint;
-(NSString*)recognize_hum_fp:(NSData*)fingerprint;

//only support RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 8000 Hz
+(NSData*)get_fingerprint:(char*)pcm len:(int)len;
+(NSData*)get_fingerprint:(NSData*)pcm;
+(NSData*)get_hum_fingerprint:(NSData*)pcm;

+(NSData*) get_fingerprint:(char*)pcm
    len:(unsigned)len
    sampleRate:(unsigned)sampleRate
    nChannel:(short)nChannel;
+(NSData*)get_fingerprint:(NSData*)pcm
    sampleRate:(unsigned)sampleRate
    nChannel:(short)nChannel;

//convert pcm/wav to mono 8000HZ
+(NSData*) resample:(NSData*)pcm
    sampleRate:(unsigned)sampleRate
    nChannel:(short)nChannel;

+(NSData*) resample:(char*)pcm
    len:(unsigned)len
    sampleRate:(unsigned)sampleRate
    nChannel:(short)nChannel;

+(NSData*) resample_bit32:(char*)pcm
    len:(unsigned)bytes
    sampleRate:(unsigned)sampleRate
    nChannel:(short)nChannel
    isFloat:(bool)isFloat;

+(NSString*) version;
@end
```



Dealing with Non-SPM external Dependency

Understand API of Dependency

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

@interface ACRCloudRecognition : NSObject

-(id)initWithConfig:(ACRCloudConfig*)config;

-(void)startPreRecord:(NSInteger)recordTime;
-(void)stopPreRecord;

-(void)startRecordRec;

-(void)stopRecordRec;

-(void)stopRecordAndRec;

//only support RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 8000 Hz
-(NSString*)recognize:(char*)buffer len:(int)len;

//only support RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 8000 Hz
-(NSString*)recognize:(NSData*)pcm_data;

-(void)recognize_fp:(NSData*)fingerprint
    resultBlock:(ACRCloudResultBlock)resultBlock;

-(NSString*)recognize_fp:(NSData*)fingerprint;
-(NSString*)recognize_hum_fp:(NSData*)fingerprint;

//only support RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 8000 Hz
+(NSData*)get_fingerprint:(char*)pcm len:(int)len;
+(NSData*)get_fingerprint:(NSData*)pcm;
+(NSData*)get_hum_fingerprint:(NSData*)pcm;
+(NSData*)get_fingerprint:(char*)pcm;
```



Dealing with Non-SPM external Dependency

Study the API with respect to feature

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

@interface ACRCLOUDRecognition : NSObject

-(id)initWithConfig:(ACRCloudConfig*)config;

-(void)startPreRecord:(NSInteger)recordTime;
-(void)stopPreRecord;

-(void)startRecordRec;

-(void)stopRecordRec;

-(void)stopRecordAndRec;

//only support RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 8000 Hz
-(NSString*)recognize:(char*)buffer len:(int)len;

//only support RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 8000 Hz
-(NSString*)recognize:(NSData*)pcm_data;

-(void)recognize_fp:(NSData*)fingerprint
    resultBlock:(ACRCloudResultBlock)resultBlock;

-(NSString*)recognize_fp:(NSData*)fingerprint;
-(NSString*)recognize_hum_fp:(NSData*)fingerprint;
```



Dealing with Non-SPM external Dependency

Use Abstraction (Protocol)

Inside the feature (SPM) module

```
import Foundation
import Models

public protocol SongRecogniser {
    init(configuration: Configuration)
    func start()
    func stop()
    func statusStream() -> AsyncThrowingStream<String, Error>
    func songRecognitionResultStream() -> AsyncThrowingStream<Models.MediaRecognitionResult, Error>
    func isRecognisingStream() -> AsyncStream<Bool>
}
```



Dealing with Non-SPM external Dependency

Use Abstraction (Protocol)

Inside the feature (SPM) module

```
import Foundation
import Models

public protocol SongRecogniser {
    init(configuration: Configuration)
    func start()
    func stop()
    func statusStream() -> AsyncThrowingStream<String, Error>
    func songRecognitionResultStream() -> AsyncThrowingStream<Models.MediaRecognitionResult, Error>
    func isRecognisingStream() -> AsyncStream<Bool>
}

public struct SongRecogniserDefault: SongRecogniser {
    public init(configuration: Configuration) {}
    public func start() {}
    public func stop() {}
    public func statusStream() -> AsyncThrowingStream<String, Error> {
        AsyncThrowingStream { _ in }
    }
    public func songRecognitionResultStream() -> AsyncThrowingStream<MediaRecognitionResult, Error> {
        AsyncThrowingStream { continuation in
            continuation.finish()
        }
    }
    public func isRecognisingStream() -> AsyncStream<Bool> {
        AsyncStream {
            $0.finish()
        }
    }
}
```



Dealing with Non-SPM external Dependency Conforming to Protocol

Inside the App Target

```
public class SongRecognitionManager: SongRecogniser {
    var client: ACRCLOUDRecognition
    var configuration: Configuration
    var states: ((String?) -> Void)?
    let config = ACRCLOUDConfig()
    private var isRecognising: CurrentValueSubject<Bool, Never> = .init(false)

    @Dependency(\.logger["SongRecognitionManager"]) private var logger
    static let jsonDecoder = JSONDecoder()

    public enum SongRecognitionError: Swift.Error, LocalizedError {
        case message(String)

        public var errorDescription: String? {
            switch self {
            case .message(let message):
                return message
            }
        }
    }

    required public init(configuration: Configuration) {
        self.configuration = configuration
        config.accessKey = configuration.accessKey
        config.accessSecret = configuration.accessSecret
        config.host = configuration.host
        config.protocol = configuration.protocol
        config.recMode = configuration.recognitionMode.recMode

        if (config.recMode == rec_mode_local || config.recMode == rec_mode_both) {
            config.homedir = Bundle.main.resourcePath!.appending("/acrcld_local_db");
        }
    }
}
```

Ratnesh Jain



```
    AsyncThrowingStream { (continuation: AsyncThrowingStream<Models.MediaRecognitionResult, Error>.Continuation) in
        @Dependency(\.logger["SongRecognitionResultStream"]) var logger
        self.config.resultBlock = { result, type in
            logger.debug("Music recognition Result: \(result ?? "NO-RESULT")")
            let type = RecognitionResultType(type: type)
            logger.debug("Music Recognition Type: \(\"\\(type)\"")")

            guard let data = result?.data(using: .utf8) else {
                continuation.finish(throwing: SongRecognitionError.message("Can not get the result."))
                return
            }
            do {
                let message = try Self.jsonDecoder.decode(Models.MediaRecognitionResult.self, from: data)
                logger.debug("Decoded recognition result: \(\"\\(message)\"")")
                continuation.yield(message)
                continuation.finish()
            } catch {
                print("Error decoding MediaRecognitionResult: \(error)")
                continuation.finish(throwing: SongRecognitionError.message("Could not decode result"))
            }
        }

        continuation.onTermination = { _ in
            self.stop()
        }
    }
}

public func isRecognisingStream() -> AsyncStream<Bool> {
    AsyncStream { [weak self] continuation in
        guard let self else {
            continuation.finish()
            return
        }
        let token = self.isRecognising.sink { state in
            continuation.yield(state)
        }
        continuation.onTermination = { @Sendable _ in
            token.cancel()
        }
    }
}
```



Dealing with Non-SPM external Dependency

Use Dependency Injection or environment override

Inside the App Target

```
@main
struct RecognApp: App {

    let store: StoreOf<AppFeature> = .init(initialState: .init()) {
        AppFeature()
    }

    var body: some Scene {
        WindowGroup {
            AppView(store: store)
        }
    }
}
```



Dealing with Non-SPM external Dependency

Use Dependency Injection

Inside the App Target

```
@main
struct RecognApp: App {

    let store: StoreOf<AppFeature> = .init(initialState: .init()) {
        @Dependency(\.config) var config

        let manager = SongRecognitionManager(configuration: .init(
            accessKey: config.acrCloud.hummingAccessKey,
            accessSecret: config.acrCloud.hummingSecretKey,
            host: config.acrCloud.host,
            recognitionMode: .remote
        ))
        AppFeature()
    }

    var body: some Scene {
        WindowGroup {
            AppView(store: store)
        }
    }
}
```



Dealing with Non-SPM external Dependency

Use Dependency Injection

Inside the App Target

```
@main
struct RecognApp: App {

    let store: StoreOf<AppFeature> = .init(initialState: .init()) {
        @Dependency(\.config) var config

        let manager = SongRecognitionManager(configuration: .init(
            accessKey: config.acrCloud.hummingAccessKey,
            accessSecret: config.acrCloud.hummingSecretKey,
            host: config.acrCloud.host,
            recognitionMode: .remote
        ))
        AppFeature()
            .transformDependency(\.songRecogniserClient) { dependency in
                dependency.set(manager)
            }
    }

    var body: some Scene {
        WindowGroup {
            AppView(store: store)
        }
    }
}
```



Dealing with Non-SPM external Dependency

Use Environment override

Inside the App Target

```
@main
struct RecognApp: App {
    @Dependency(\.config) var
    let manager = SongRecognitionManager(configuration: .init(
        accessKey: config.acrCloud.hummingAccessKey,
        accessSecret: config.acrCloud.hummingSecretKey,
        host: config.acrCloud.host,
        recognitionMode: .remote
    ))
    var body: some Scene {
        WindowGroup {
            AppView()
        }
    }
}
```



Dealing with Non-SPM external Dependency

Use Environment override

Inside the App Target

```
@main
struct RecognApp: App {
    @Dependency(\.config) var
    let manager = SongRecognitionManager(configuration: .init(
        accessKey: config.acrCloud.hummingAccessKey,
        accessSecret: config.acrCloud.hummingSecretKey,
        host: config.acrCloud.host,
        recognitionMode: .remote
    ))
    var body: some Scene {
        WindowGroup {
            AppView()
                .environmentObject(manager)
        }
    }
}
```



Mastering SwiftPM

Today

- Access Modifiers
- Package Manifest
- Package Manager Vocabulary
- Dependencies
- New Feature using SwiftPM
- Integration with Xcode Project
- Sharing Resources
- Dealing with Non-SPM external Dependency



Getting to know Swift Package Manager WWDC18

<https://developer.apple.com/wwdc18/411>

Creating Swift Package WWDC19

<https://developer.apple.com/wwdc19/410>

Adopting Swift Packages in Xcode WWDC19

<https://developer.apple.com/wwdc19/408>

Swift Packages: Resources & Localization WWDC20

<https://developer.apple.com/wwdc20/10169>

Creating Swift Package Plugin WWDC22

<https://developer.apple.com/wwdc22/110401>



Swift Package Manager Dependency Owners

<https://alejandromp.com/blog/swift-package-manager-dependency-owners/>

Adding Core Data to a Swift Package

<https://ishabazz.dev/blog/2020/7/5/using-core-data-with-swift-package-manager>

Localizing Swift Package

<https://developer.apple.com/documentation/xcode/localizing-package-resources>

SwiftSettings

<https://developer.apple.com/documentation/packagedescription/swiftsetting>



Mastering SwiftPM

Swift Package manager

Ratnesh Jain





Thank You

