#WWDC19

Creating Swift Packages

Boris Buegling, Developer Tools Engineer
Ankit Aggarwal, Developer Tools Engineer

Creating local packages

Publishing packages

Package manifest API

Editing packages

Open source project

Creating local packages

Publishing packages

Package manifest API

Editing packages

Open source project

Share Code

Within your workspace

With your team

With the open source community

Creating Local Packages

Local Swift Packages

Can be thought of as sub-projects in a Xcode workspace

Local Swift Packages

Can be thought of as sub-projects in a Xcode workspace

Platform independent

Local Swift Packages

Can be thought of as sub-projects in a Xcode workspace

Platform independent

Great for refactoring out reusable code

Local Swift Packages

Can be thought of as sub-projects in a Xcode workspace

Platform independent

Great for refactoring out reusable code

Not versioned

Local Swift Packages

Can be thought of as sub-projects in a Xcode workspace

Platform independent

Great for refactoring out reusable code

Not versioned

Can publish and share with others later

Demo

Creating local packages

Publishing Packages

Semantic Versioning

semver.org

1.2.4

Major Version
Breaking changes

1.2.4

Minor Version
Compatible additions

1.2.4

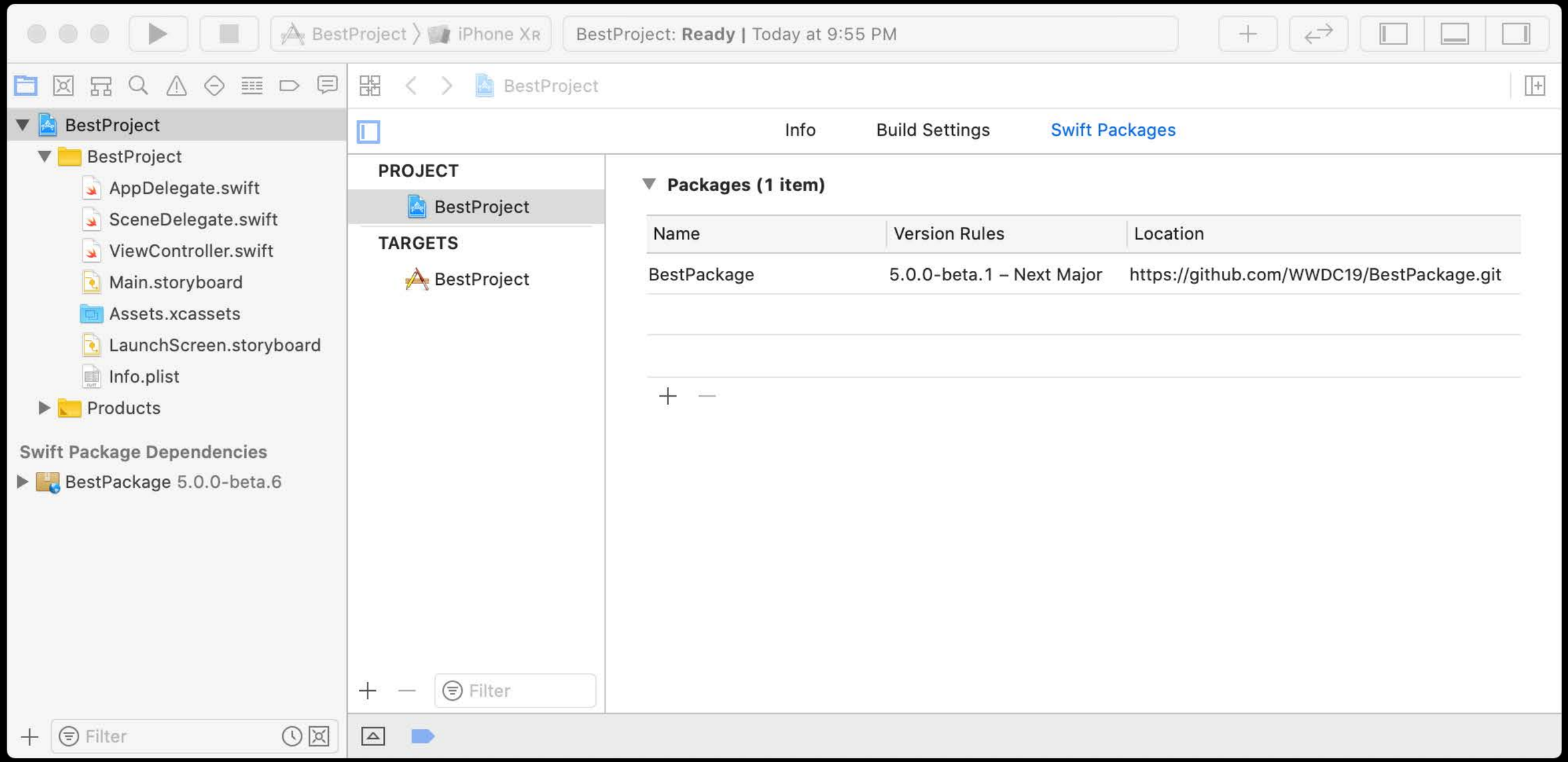
Patch Version
Bugfixes

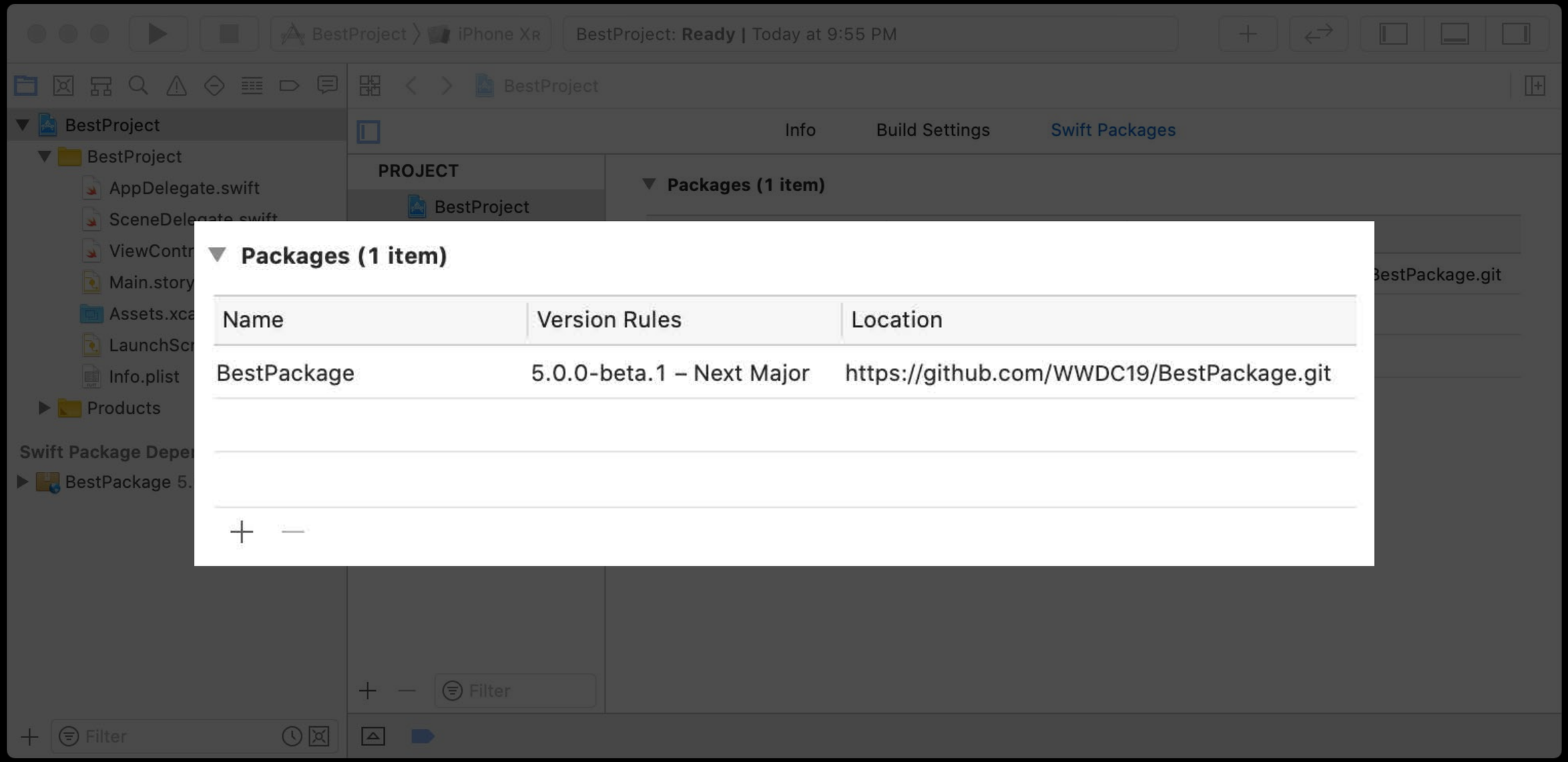
0.x.y

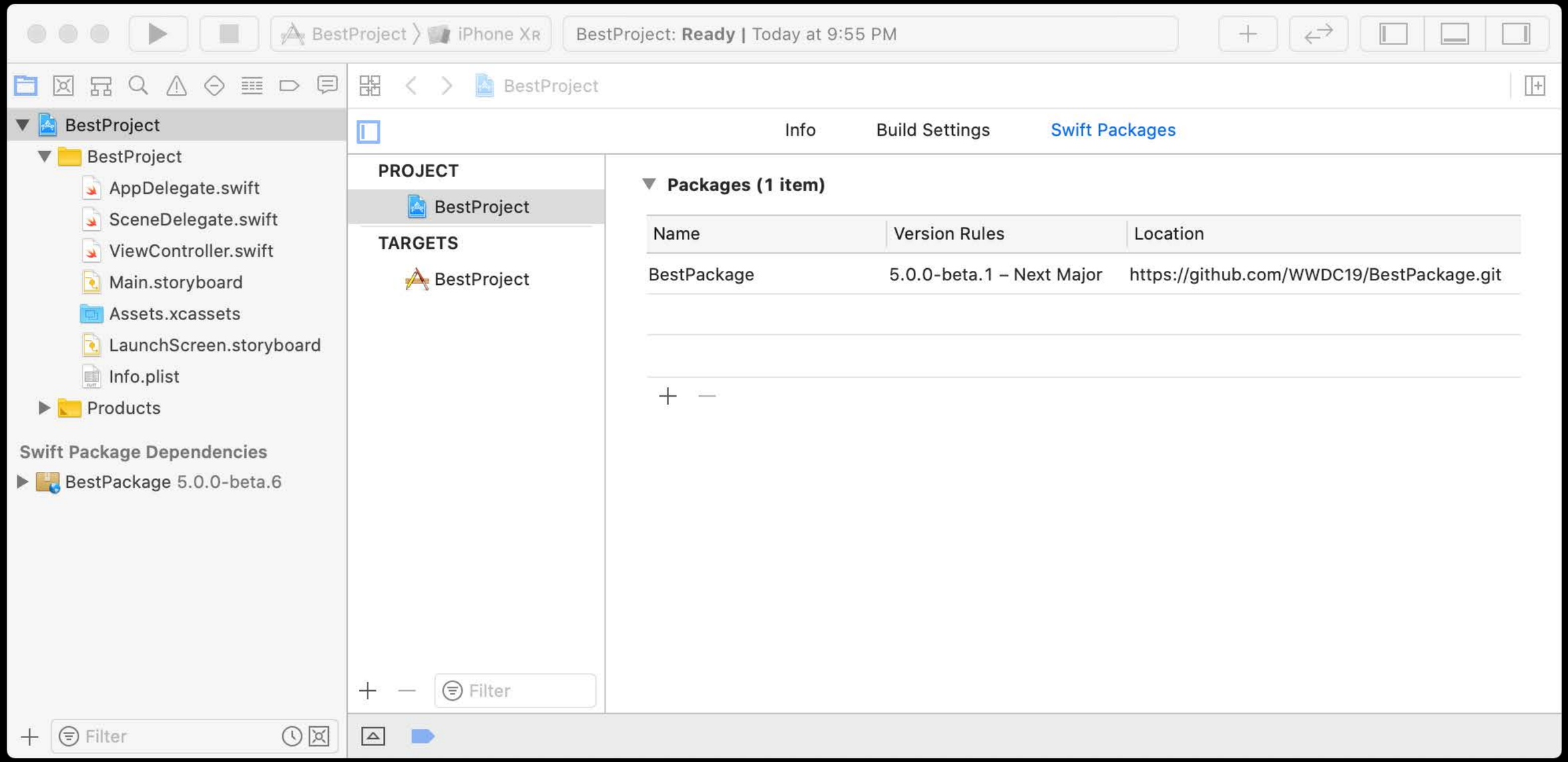
Initial Development

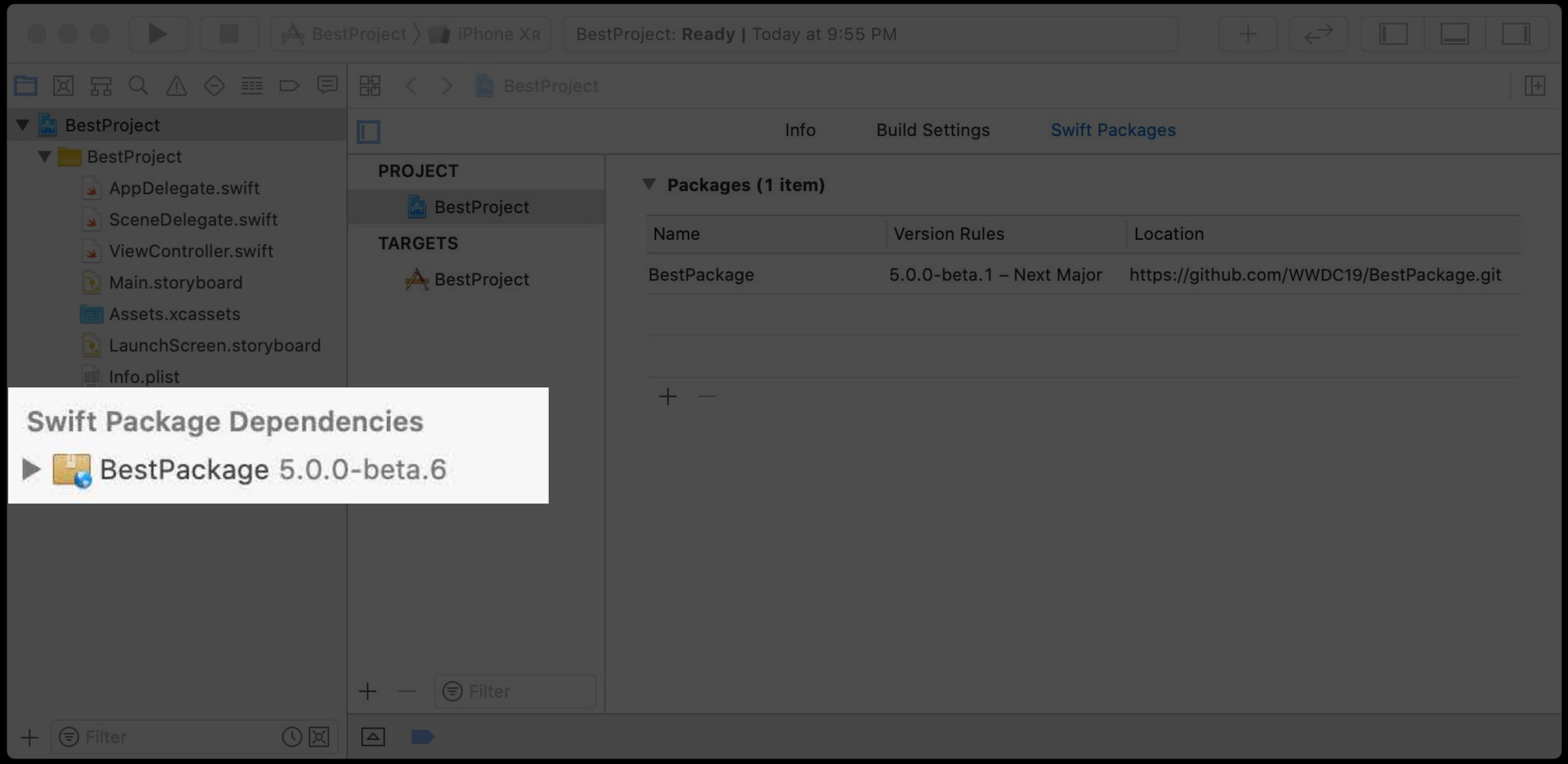
2.0.0-alpha.1

Prerelease Version


BestProject 〉 iPhone XR
BestProject: Ready | Today at 9:55 PM
BestProject
BestProject
BestProject
AppDelegate.swift
SceneDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Products
Swift Package Dependencies
BestPackage 5.0.0-beta.6
Filter
BestProject
Info
Build Settings
Swift Packages
PROJECT
BestProject
TARGETS
BestProject
Filter
Packages (1 item)
Name
Version Rules
Location
BestPackage
5.0.0-beta.1 – Next Major
https://github.com/WWDC19/BestPackage.git


BestProject
iPhone XR
BestProject: Ready | Today at 9:55 PM
BestProject
Info
Build Settings
Swift Packages
PROJECT
BestProject
Packages (1 item)
BestProject
AppDelegate.swift
Products
Filter
Packages (1 item)
Name
Version Rules
Location
BestPackage
5.0.0-beta.1 – Next Major
https://github.com/WWDC19/BestPackage.git


BestProject 〉 iPhone XR
BestProject: Ready | Today at 9:55 PM
BestProject
BestProject
BestProject
AppDelegate.swift
SceneDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Products
Swift Package Dependencies
BestPackage 5.0.0-beta.6
Filter
BestProject
Info
Build Settings
Swift Packages
PROJECT
BestProject
TARGETS
BestProject
Filter
Packages (1 item)
Name
Version Rules
Location
BestPackage
5.0.0-beta.1 – Next Major
https://github.com/WWDC19/BestPackage.git


BestProject 〉 iPhone XR
BestProject: Ready | Today at 9:55 PM
BestProject
BestProject
BestProject
AppDelegate.swift
SceneDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Info
Build Settings
Swift Packages
PROJECT
BestProject
TARGETS
BestProject
Packages (1 item)
Name
Version Rules
Location
BestPackage
5.0.0-beta.1 – Next Major
https://github.com/WWDC19/BestPackage.git
Swift Package Dependencies
BestPackage 5.0.0-beta.6
Filter
Filter

Demo

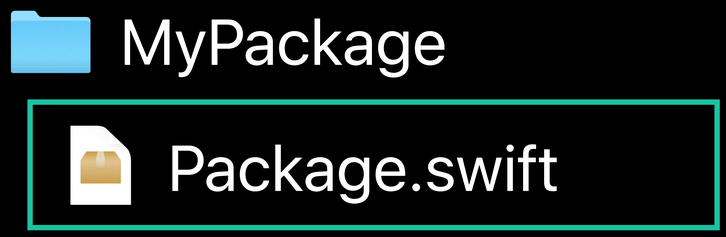
Publishing packages

Package Manifest API

Ankit Aggarwal, Developer Tools Engineer

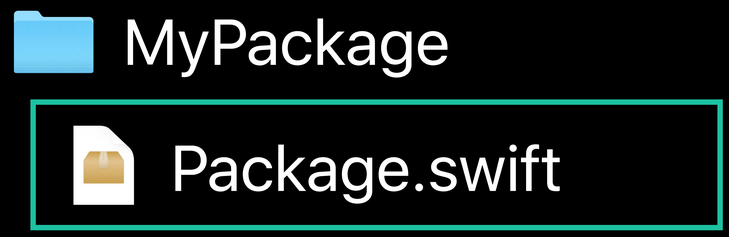
Package Manifest File

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage"
)
```



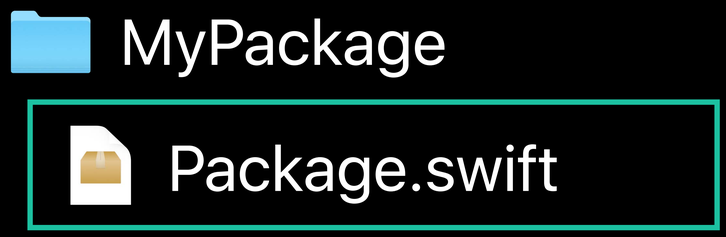
Package Manifest File

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage"
)
```



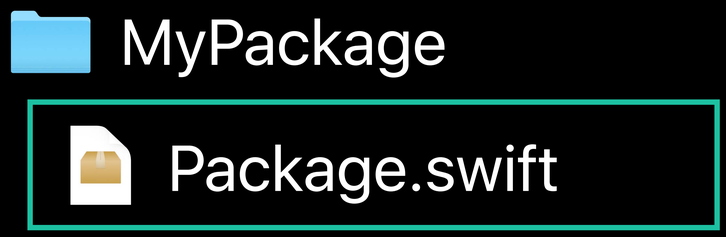
Package Manifest File

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage"
)
```



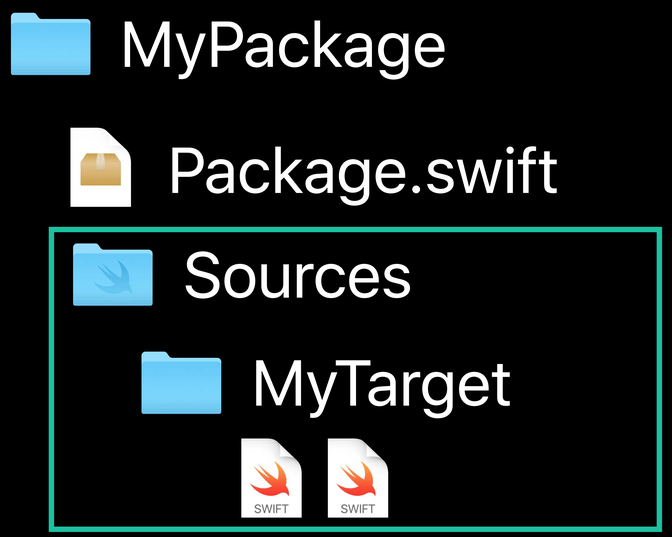
Package Manifest File

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage"
)
```



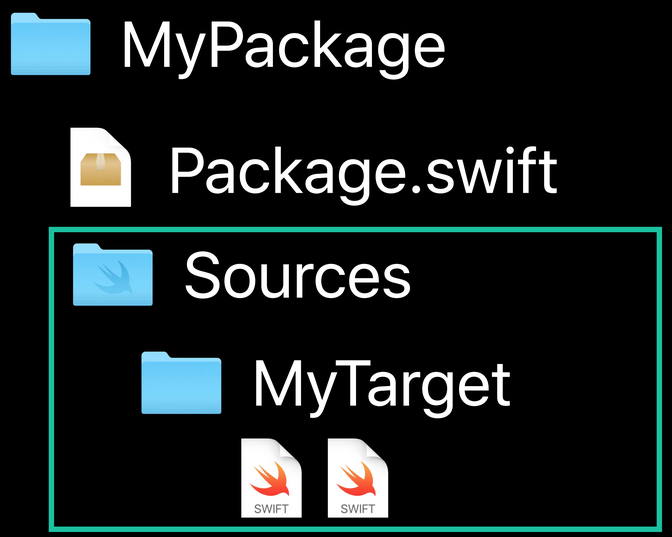
Configuring Targets

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage",
    targets: [
        .target(name: "MyTarget"),
    ]
)
```



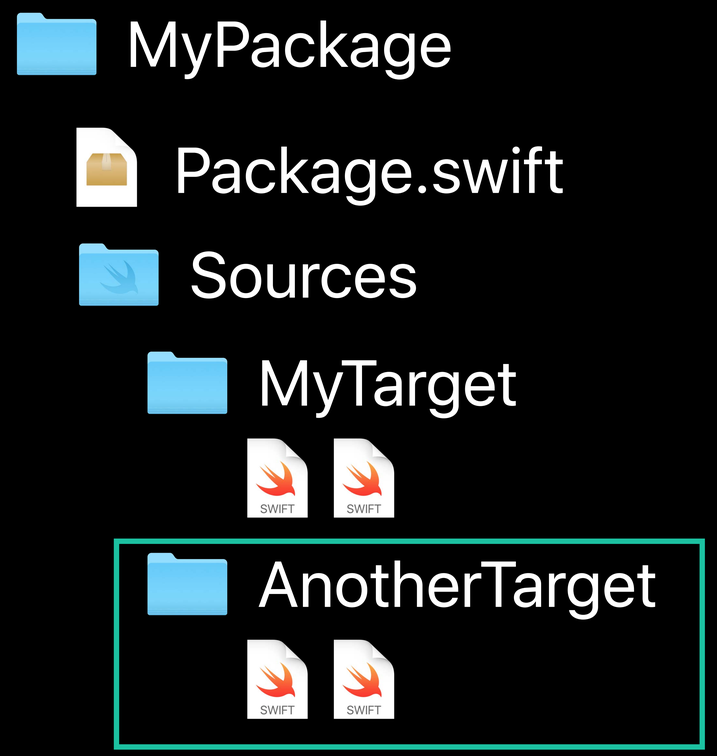
Configuring Targets

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage",
    targets: [
        .target(name: "MyTarget"),
    ]
)
```



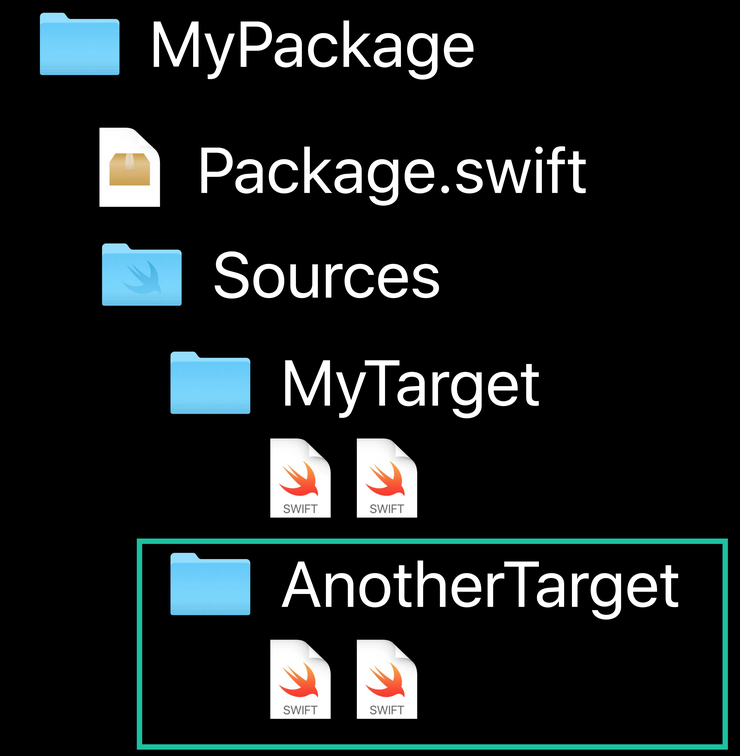
Configuring Targets

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage",
    targets: [
        .target(name: "MyTarget"),
        .target(name: "AnotherTarget"),
    ]
)
```



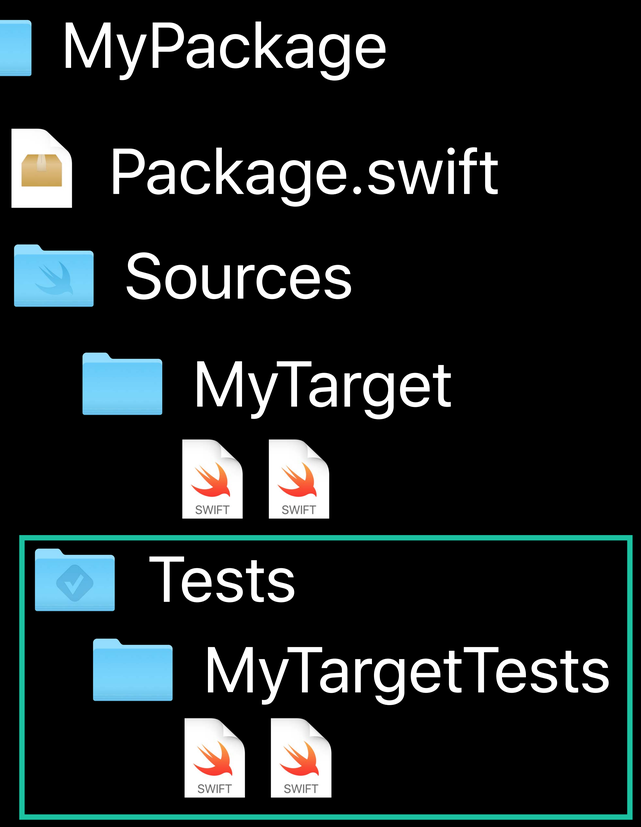
Configuring Targets

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage",
    targets: [
        .target(name: "MyTarget"),
        .target(name: "AnotherTarget"),
    ]
)
```



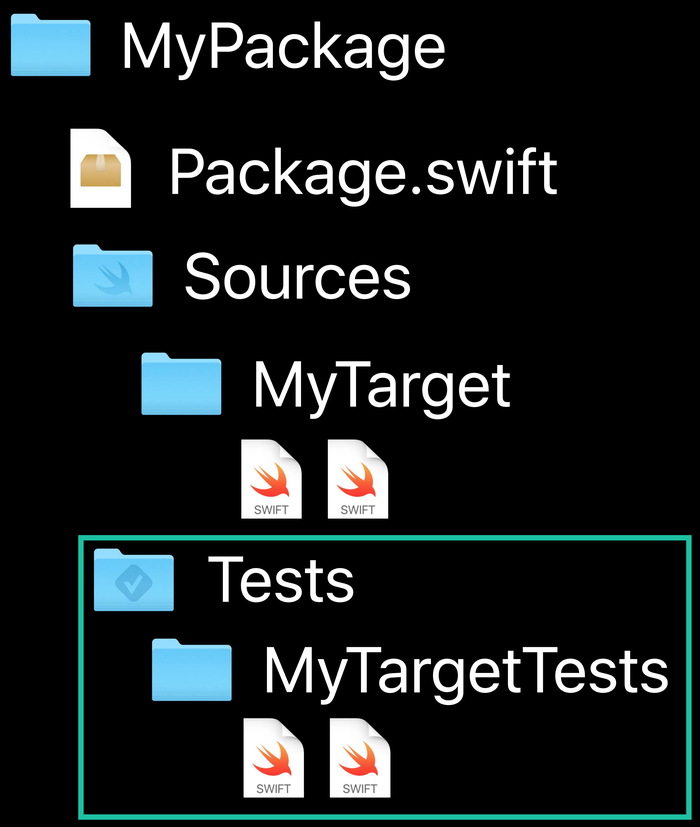
Configuring Test Targets

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage",
    targets: [
        .target(name: "MyTarget"),
        .testTarget(name: "MyTargetTests"),
    ]
)
```



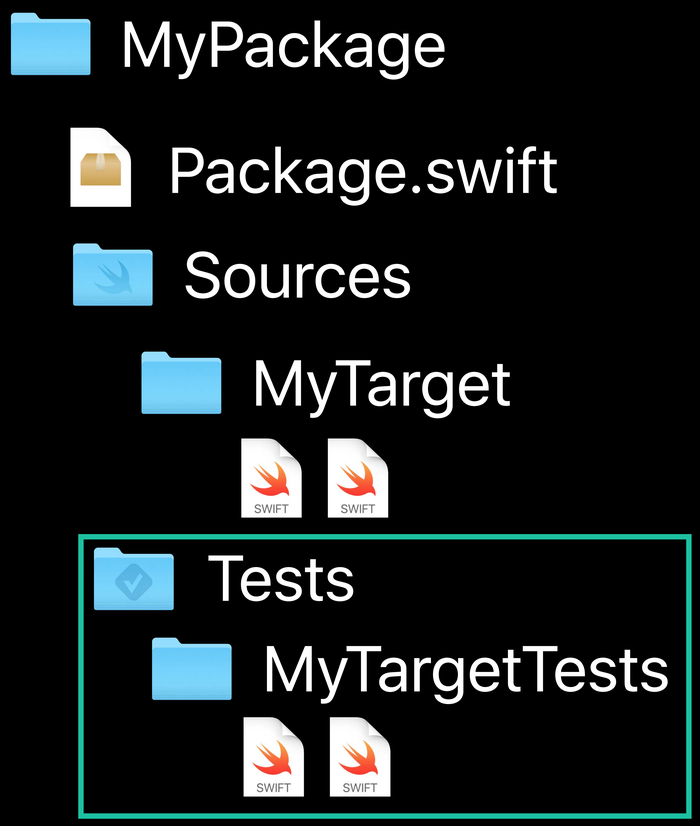
Configuring Test Targets

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage",
    targets: [
        .target(name: "MyTarget"),
        .testTarget(name: "MyTargetTests"),
    ]
)
```



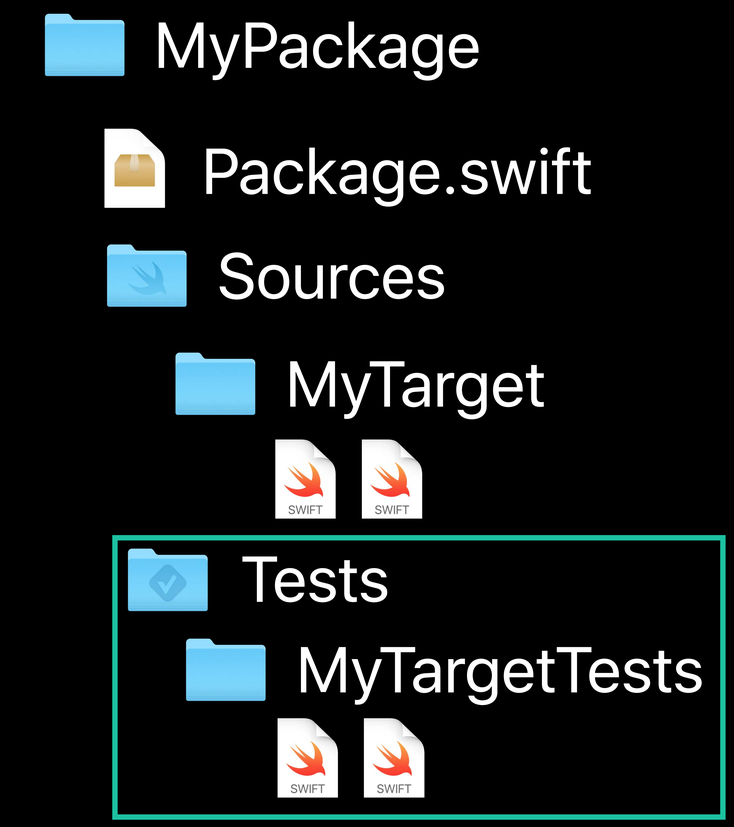
Configuring Test Targets

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage",
    targets: [
        .target(name: "MyTarget"),
        .testTarget(
            name: "MyTargetTests",
            dependencies: ["MyTarget"]
        ),
    ]
)
```



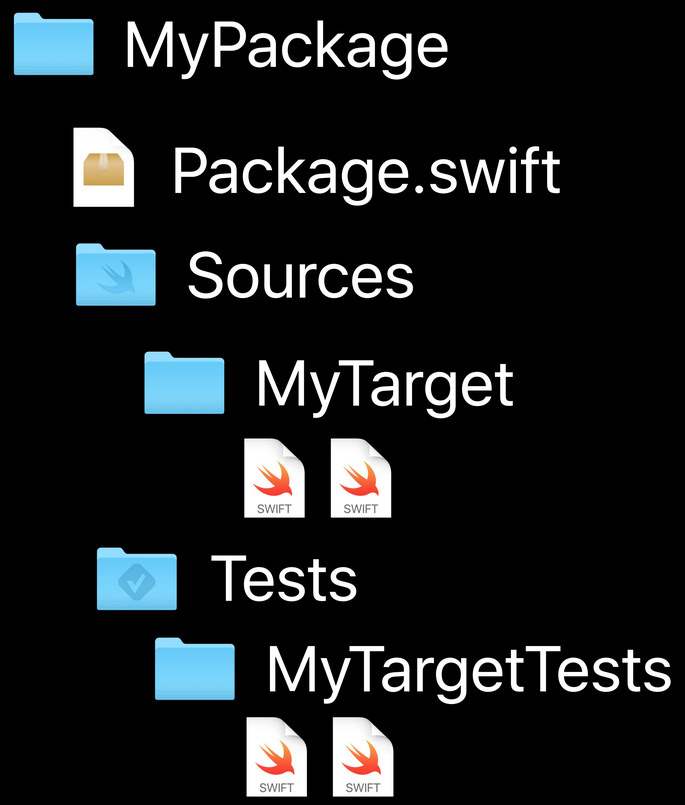
Configuring Test Targets

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage",
    targets: [
        .target(name: "MyTarget"),
        .testTarget(
            name: "MyTargetTests",
            dependencies: ["MyTarget"]
        ),
    ]
)
```



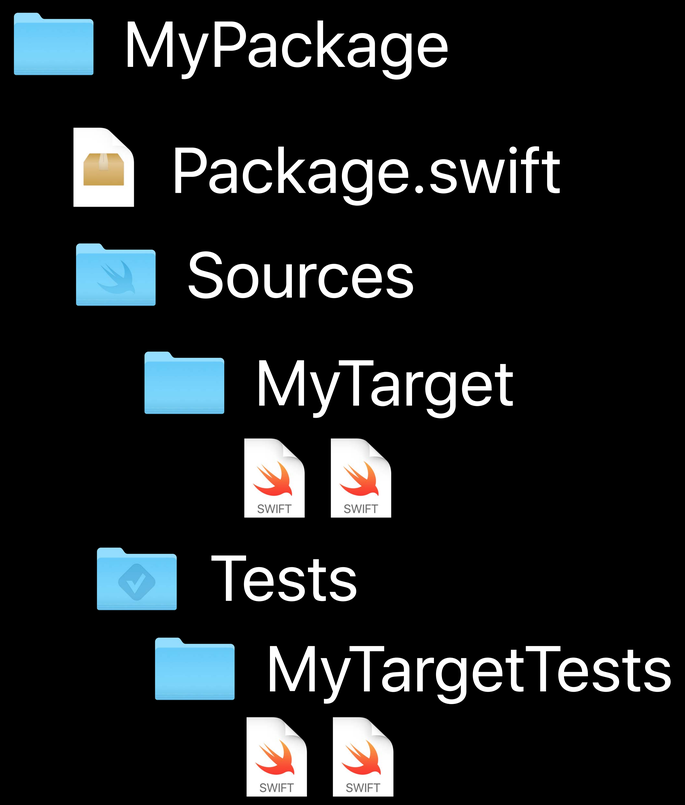
Configuring Products

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage",
    products: [
        .library(
            name: "MyProduct",
            targets: ["MyTarget"]
        ),
    ],
    targets: [
        .target(name: "MyTarget"),
         ...
    ]
)
```



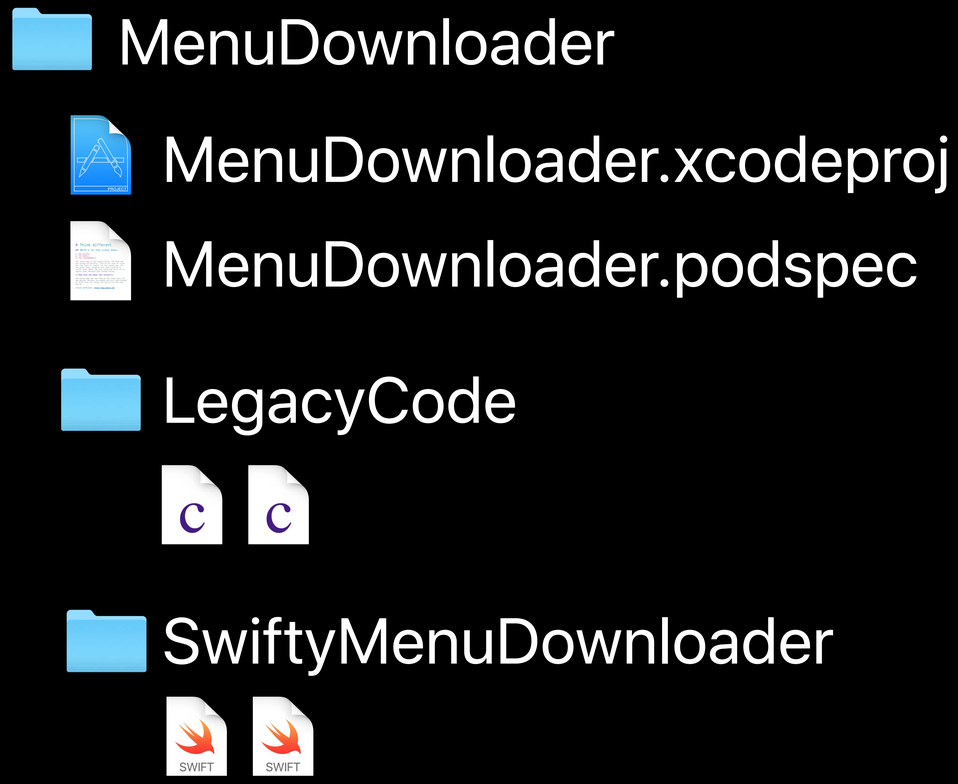
Configuring Products

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage",
    products: [
        .library(
            name: "MyProduct",
            targets: ["MyTarget"]
        ),
    ],
    targets: [
        .target(name: "MyTarget"),
         ...
    ]
)
```

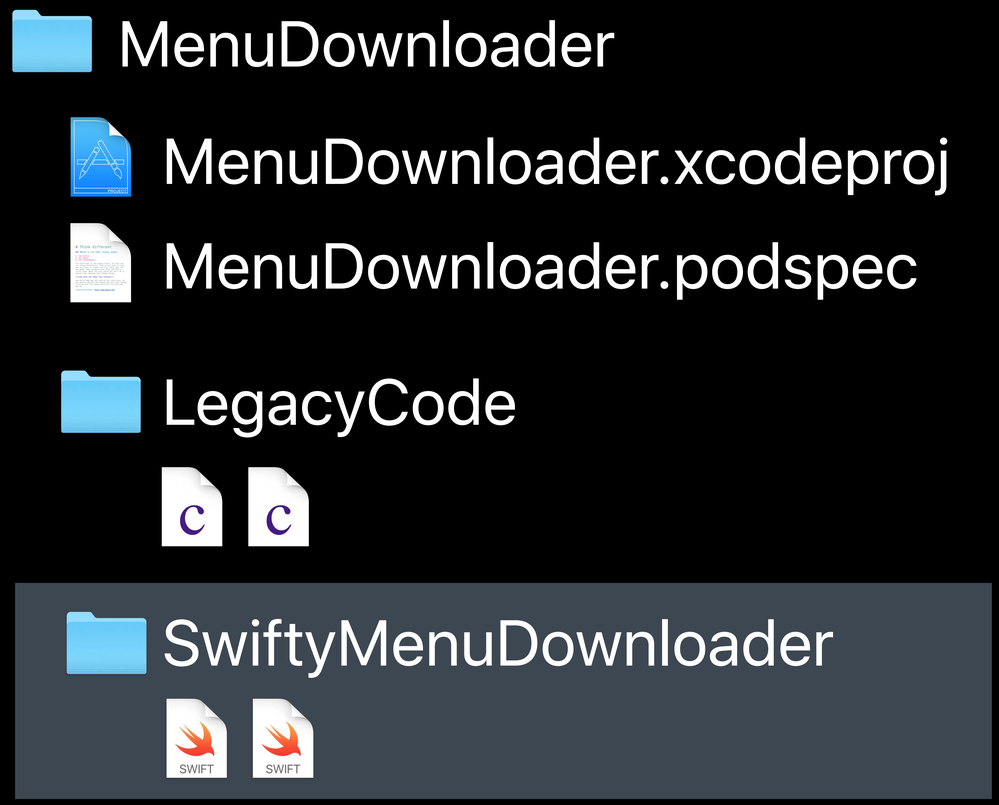


Adding support in existing projects

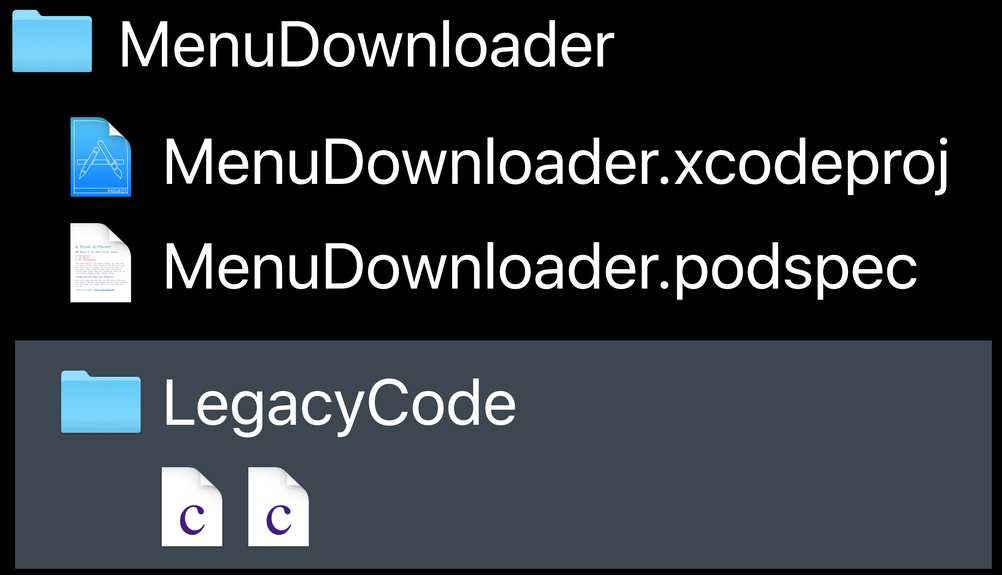
Example — MenuDownloader



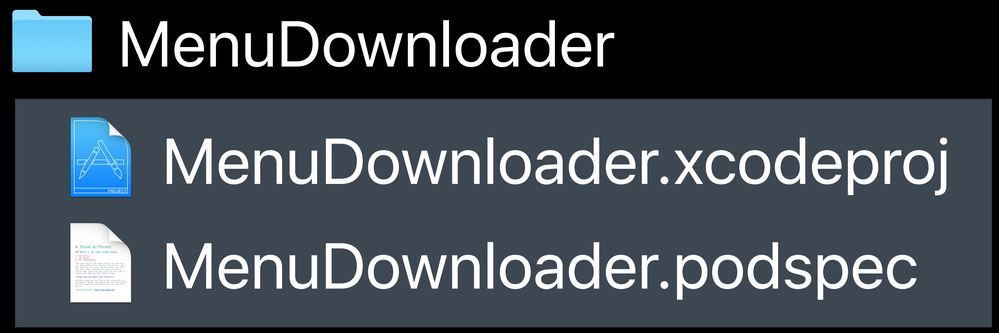
Example — MenuDownloader



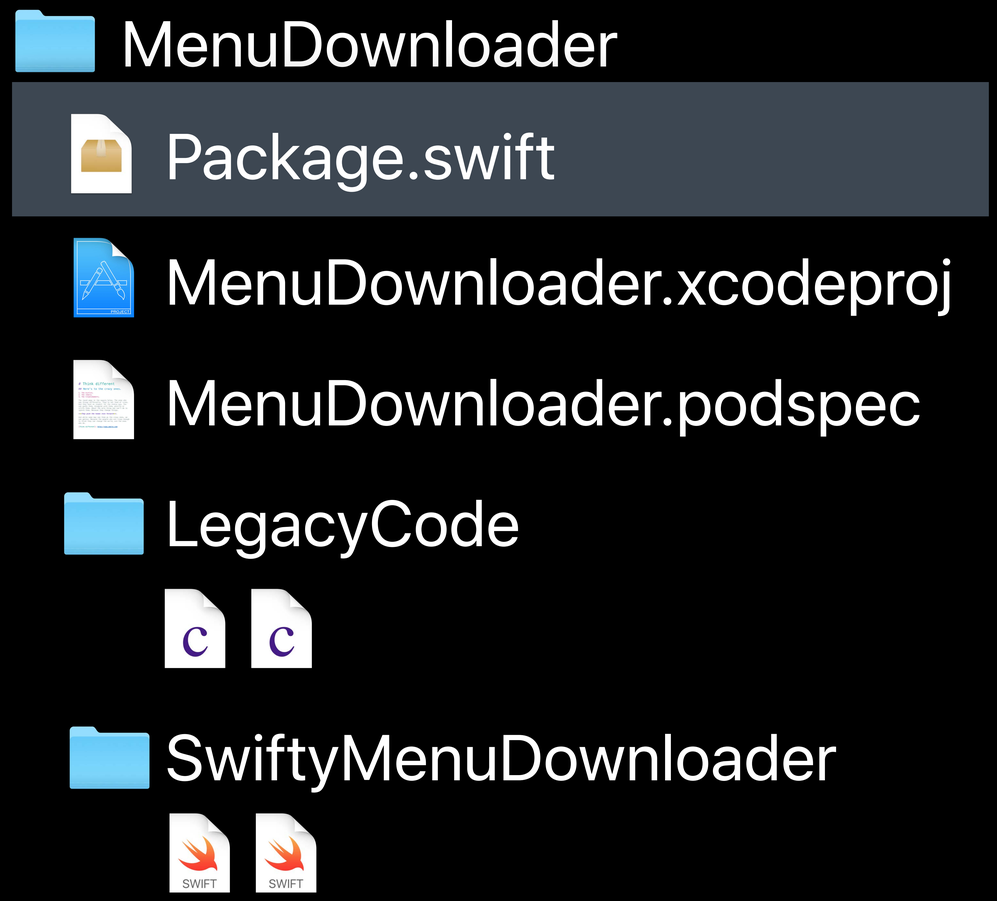
Example — MenuDownloader



Example — MenuDownloader

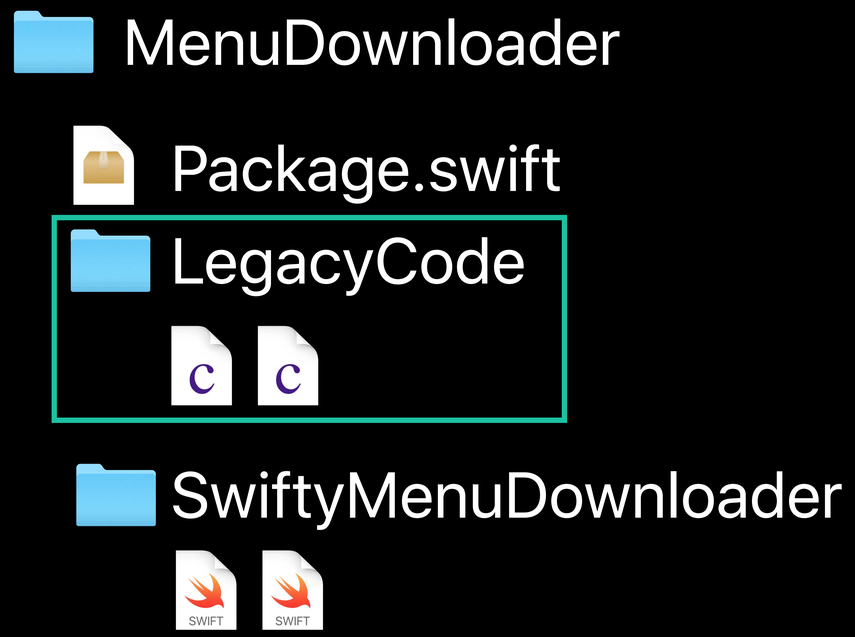


Example — MenuDownloader



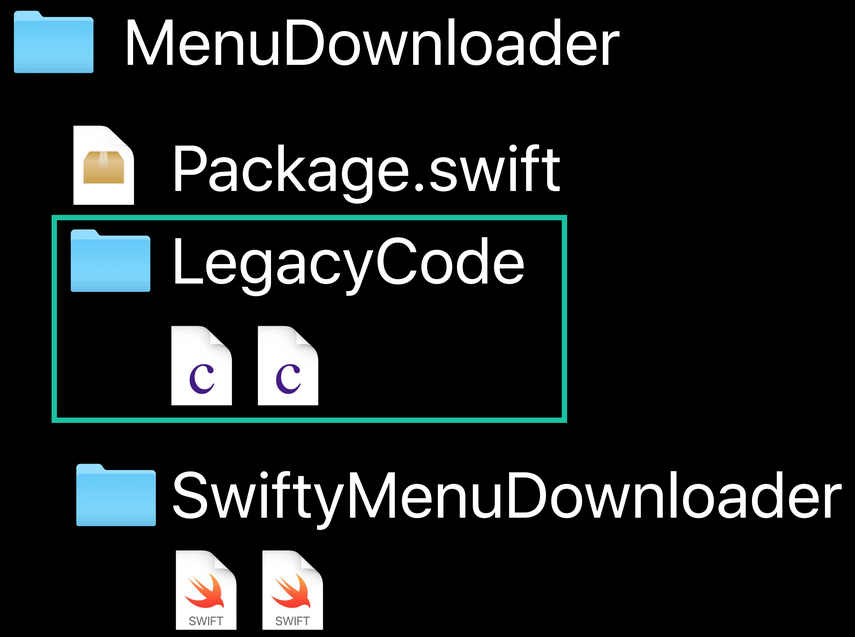
Configuring Targets

```
targets: [
    .target(
        name: "LegacyMenuDownloader",
        path: "LegacyCode",
        cSettings: [
          .define("SHOW_SECRET_MENU", to: "1")
        ]
    ),
]
```



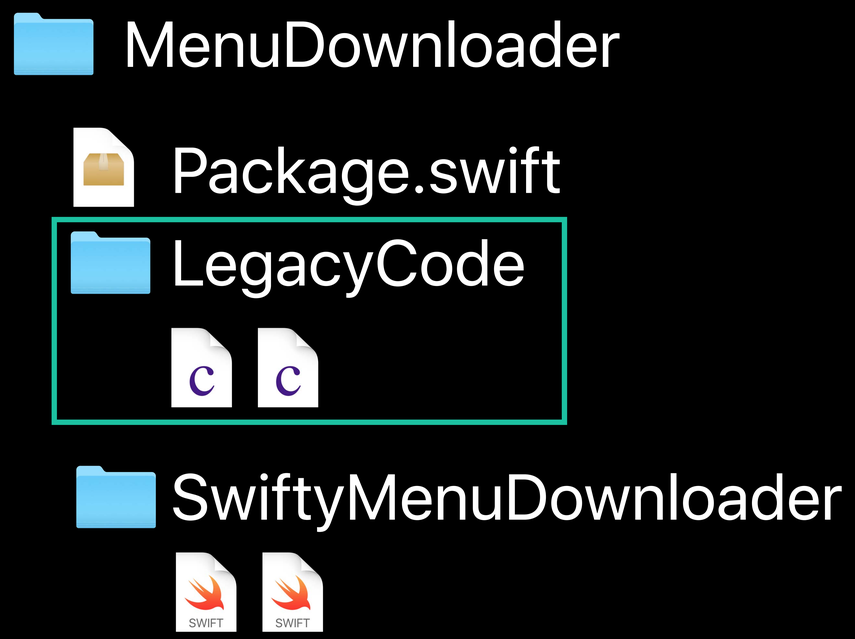
Configuring Targets

```
targets: [
    .target(
        name: "LegacyMenuDownloader",
        path: "LegacyCode",
        cSettings: [
          .define("SHOW_SECRET_MENU", to: "1")
        ]
    ),
]
```



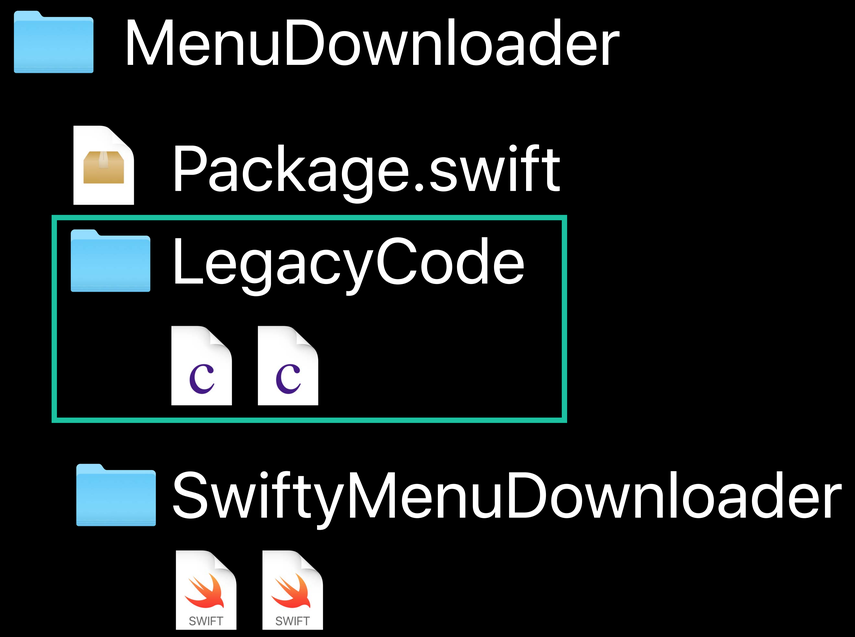
Configuring Targets

```
targets: [
    .target(
        name: "LegacyMenuDownloader",
        path: "LegacyCode",
        cSettings: [
          .define("SHOW_SECRET_MENU", to: "1")
        ]
    ),
]
```



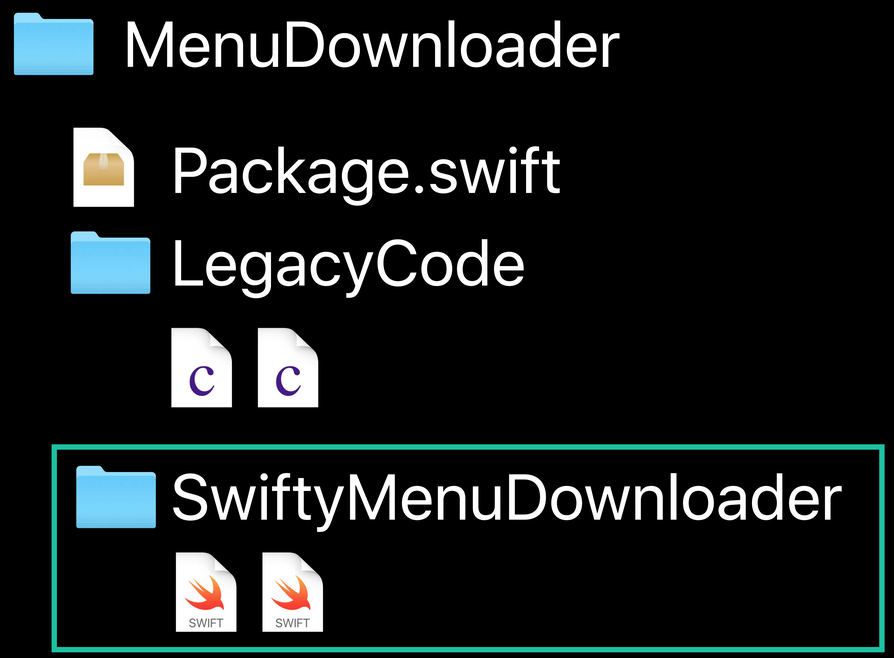
Configuring Targets

```
targets: [
    .target(
        name: "LegacyMenuDownloader",
        path: "LegacyCode",
        cSettings: [
          .define("SHOW_SECRET_MENU", to: "1")
        ]
    ),
]
```



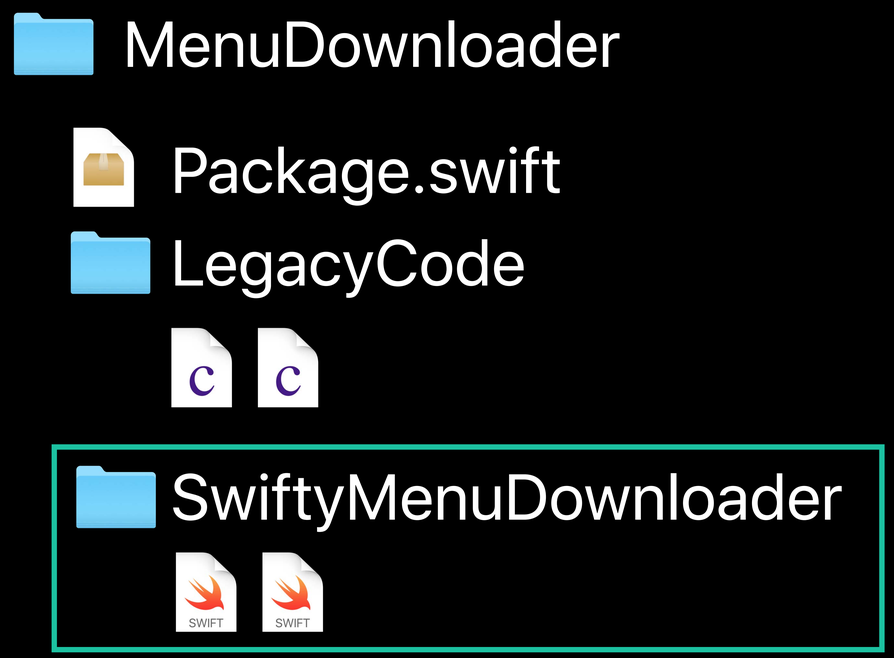
Configuring Targets

```
targets: [
        ...
    .target(
        name: "SwiftyMenuDownloader",
        path: "SwiftyMenuDownloader",
        dependencies: ["LegacyMenuDownloader"]
    ),
]
```



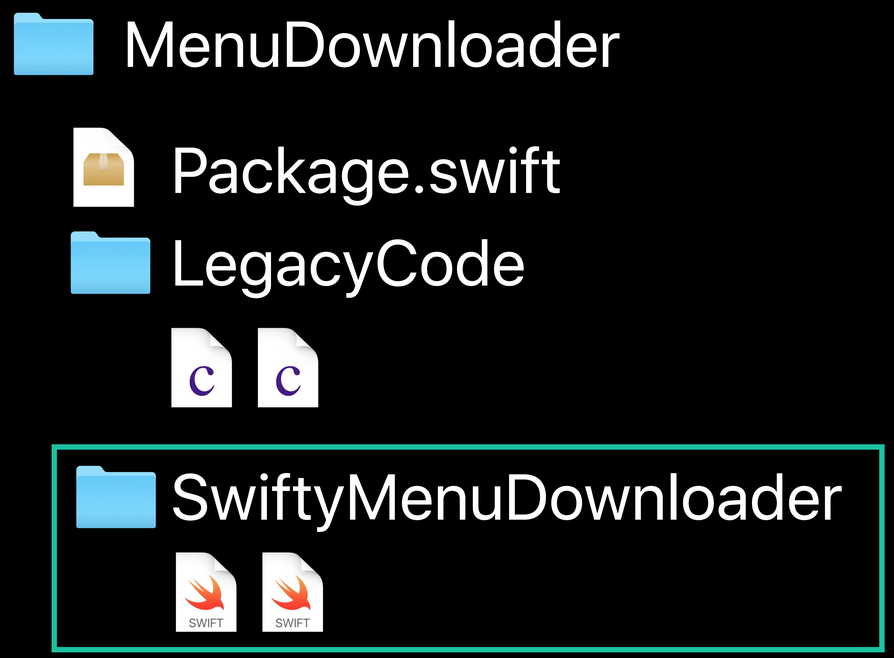
Configuring Targets

```
targets: [
        ...
    .target(
        name: "SwiftyMenuDownloader",
        path: "SwiftyMenuDownloader",
        dependencies: ["LegacyMenuDownloader"]
    ),
]
```



Configuring Targets

```
targets: [
        ...
    .target(
        name: "SwiftyMenuDownloader",
        path: "SwiftyMenuDownloader",
        dependencies: ["LegacyMenuDownloader"]
    ),
]
```



```
// Configuring Library Products
          ...
products: [
    .library(
        name: "SwiftyMenuDownloader",
        targets: ["SwiftyMenuDownloader"]
    ),
    .library(
        name: "LegacyMenuDownloader",
        type: .dynamic,
        targets: ["LegacyMenuDownloader"]
    ),
]
```

```
// Configuring Library Products
        ...
products: [
    .library(
        name: "SwiftyMenuDownloader",
        targets: ["SwiftyMenuDownloader"]
    ),
    .library(
        name: "LegacyMenuDownloader",
        type: .dynamic,
        targets: ["LegacyMenuDownloader"]
    ),
]
```

```
// Configuring Library Products
        ...
products: [
    .library(
        name: "SwiftyMenuDownloader",
        targets: ["SwiftyMenuDownloader"]
    ),
    .library(
        name: "LegacyMenuDownloader",
        type: .dynamic,
        targets: ["LegacyMenuDownloader"]
    ),
]
```

```
// Configuring Library Products
        ...
products: [
    .library(
        name: "SwiftyMenuDownloader",
        targets: ["SwiftyMenuDownloader"]
    ),
    .library(
        name: "LegacyMenuDownloader",
        type: .dynamic,
        targets: ["LegacyMenuDownloader"]
    ),
]
```

Package Dependencies

```
// Configuring Package Dependencies


let package = Package(
    name: "MenuDownloader",
    products: [
        ...
    ],
    dependencies: [
      .package(url: "https://github.com/jpsim/Yams", .upToNextMajor(from: "2.0.0")),
    ],
    targets: [
        ...
    ]
)
```

```
// Configuring Package Dependencies


let package = Package(
    name: "MenuDownloader",
    products: [
        ...
    ],
    dependencies: [
      .package(url: "https://github.com/jpsim/Yams", .upToNextMajor(from: "2.0.0")),
    ],
    targets: [
        ...
    ]
)
```

```
// Configuring Package Dependencies


let package = Package(
    name: "MenuDownloader",
    products: [
        ...
    ],
    dependencies: [
      .package(url: "https://github.com/jpsim/Yams", .upToNextMajor(from: "2.0.0")),
    ],
    targets: [
        ...
    ]
)
```

```
// Configuring Package Dependencies
// A version starting from 2.0.0 up to but not including the next major version (3.0.0)

let package = Package(
    name: "MenuDownloader",
    products: [
        ...
    ],
    dependencies: [
      .package(url: "https://github.com/jpsim/Yams", .upToNextMajor(from: "2.0.0")),
    ],
    targets: [
        ...
    ]
)
```

```
// Configuring Package Dependencies
// A version starting from 2.0.0 up to but not including the next major version (3.0.0)

let package = Package(
    name: "MenuDownloader",
    products: [
        ...
    ],
    dependencies: [
      .package(url: "https://github.com/jpsim/Yams", from: "2.0.0"),
    ],
    targets: [
        ...
    ]
)
```

```
// Configuring Package Dependencies
// A version starting from 2.0.0 up to but not including the next major version (3.0.0)

let package = Package(
    name: "MenuDownloader",
    products: [
        ...
    ],
    dependencies: [
      .package(url: "https://github.com/jpsim/Yams", from: "2.0.0"),
    ],
    targets: [
        ...
    ]
)
```

```
// Configuring Package Dependencies



    // Version-based requirements.
    .package(url: "https://github.com/jpsim/Yams", from: "2.0.0")
    .package(url: "https://github.com/jpsim/Yams", .upToNextMajor(from: "2.0.0"))

    .package(url: "https://github.com/jpsim/Yams", .upToNextMinor(from: "2.0.0"))

    .package(url: "https://github.com/jpsim/Yams", .exact("2.0.0"))
```

```
// Configuring Package Dependencies


    // Version-based requirements.
    .package(url: "https://github.com/jpsim/Yams", from: "2.0.0")
    .package(url: "https://github.com/jpsim/Yams", .upToNextMajor(from: "2.0.0"))

    .package(url: "https://github.com/jpsim/Yams", .upToNextMinor(from: "2.0.0"))

    .package(url: "https://github.com/jpsim/Yams", .exact("2.0.0"))
```

```
// Configuring Package Dependencies



    // Version-based requirements.
    .package(url: "https://github.com/jpsim/Yams", from: "2.0.0")
    .package(url: "https://github.com/jpsim/Yams", .upToNextMajor(from: "2.0.0"))

    .package(url: "https://github.com/jpsim/Yams", .upToNextMinor(from: "2.0.0"))

    .package(url: "https://github.com/jpsim/Yams", .exact("2.0.0"))
```

```
// Configuring Package Dependencies



    // Version-based requirements.
    .package(url: "https://github.com/jpsim/Yams", from: "2.0.0")
    .package(url: "https://github.com/jpsim/Yams", .upToNextMajor(from: "2.0.0"))

    .package(url: "https://github.com/jpsim/Yams", .upToNextMinor(from: "2.0.0"))

    .package(url: "https://github.com/jpsim/Yams", .exact("2.0.0"))
```

```
// Configuring Package Dependencies


    // Branch-based requirement.
    .package(url: "https://github.com/jpsim/Yams", .branch("master"))

    // Revision-based requirement.
    .package(url: "https://github.com/jpsim/Yams", .revision("85cfe06"))
```

```
// Configuring Package Dependencies


    // Branch-based requirement.
    .package(url: "https://github.com/jpsim/Yams", .branch("master"))

    // Revision-based requirement.
    .package(url: "https://github.com/jpsim/Yams", .revision("85cfe06"))
```

```
// Configuring Package Dependencies


    // Branch-based requirement.
    .package(url: "https://github.com/jpsim/Yams", .branch("master"))

    // Revision-based requirement.
    .package(url: "https://github.com/jpsim/Yams", .revision("85cfe06"))
```

```
// Configuring Package Dependencies
        ...
targets: [
          ...
    .target(
        name: "SwiftyMenuDownloader",
        dependencies: ["LegacyMenuDownloader", "Yams"]
    ),
         ...
]
```

Swift Tools Version

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MenuDownloader",
        ...
)
```

Swift Tools Version

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MenuDownloader",
        ...
)
```

Manifest API version

Swift Tools Version

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MenuDownloader",
        ...
)
```

Manifest API version

Dependency resolution

Swift Tools Version

```
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MenuDownloader",
        ...
)
```

Manifest API version

Dependency resolution

Minimum compiler version

Platform Availability

macOS iOS tvOS watchOS Linux

```
#if os(Linux)
// Code specific to Linux
#endif

#if canImport(Network)
// Code specific to platforms where Network framework is available
#endif
```

Platform Availability

macOS	iOS	tvOS	watchOS	Linux
10.10	8.0	9.0	2.0	

```
#if os(Linux)
// Code specific to Linux
#endif

#if canImport(Network)
// Code specific to platforms where Network framework is available
#endif
```

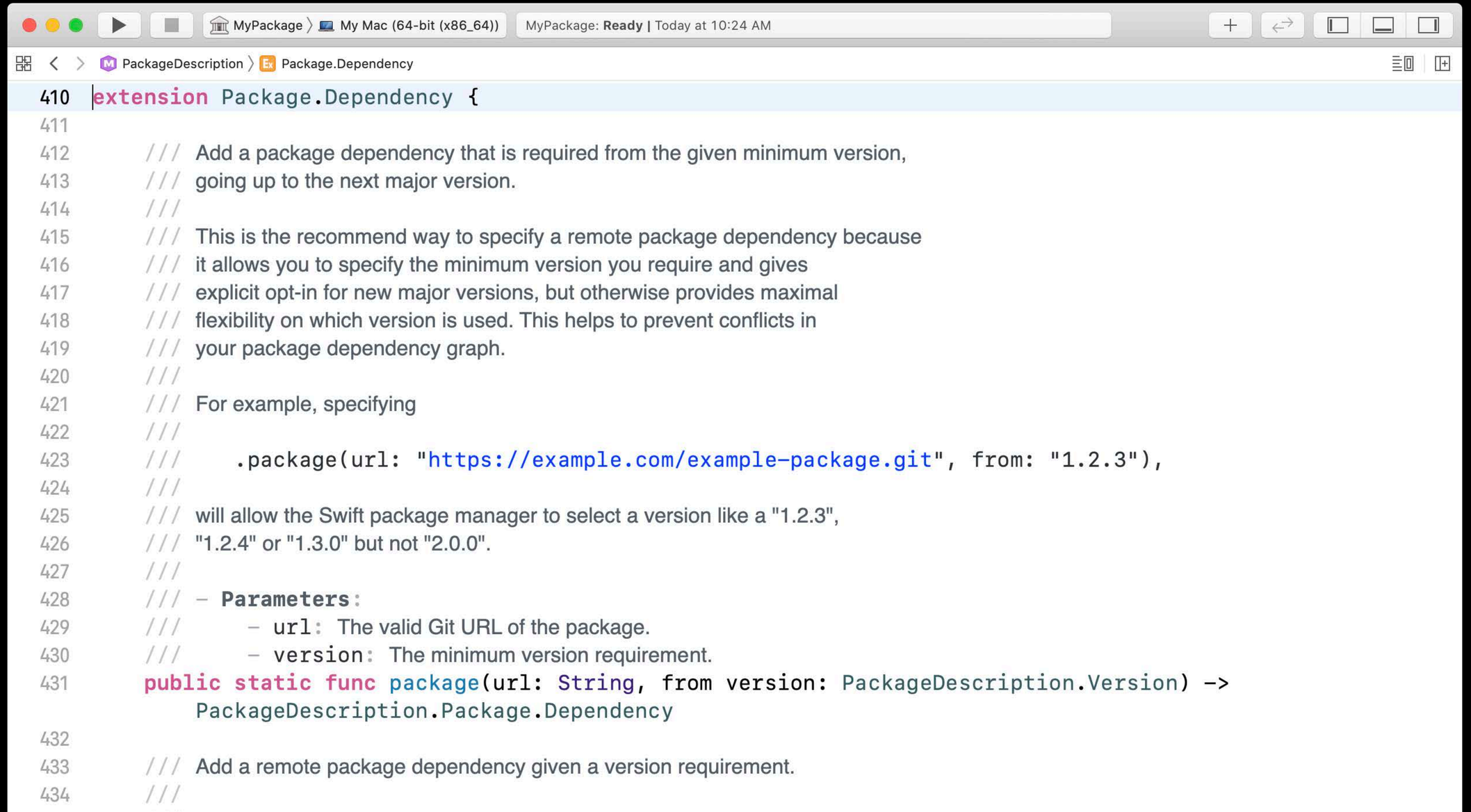
```
// Configuring Minimum Deployment Target Version

let package = Package(
    name: "MyPackage",
    platforms: [
        .macOS(.v10_15), .iOS(.v13),
    ],
    products: [
         ...
    ],
    dependencies: [
        ...
    ],
    targets: [
         ...
    ]
)
```

```
// Configuring Minimum Deployment Target Version

let package = Package(
    name: "MyPackage",
    platforms: [
        .macOS(.v10_15), .iOS(.v13),
    ],
    products: [
         ...
    ],
    dependencies: [
        ...
    ],
    targets: [
         ...
    ]
)
```

```
// Configuring Minimum Deployment Target Version

let package = Package(
    name: "MyPackage",
    platforms: [
        .macOS(.v10_15), .iOS(.v13),
    ],
    products: [
         ...
    ],
    dependencies: [
        ...
    ],
    targets: [
         ...
    ]
)
```

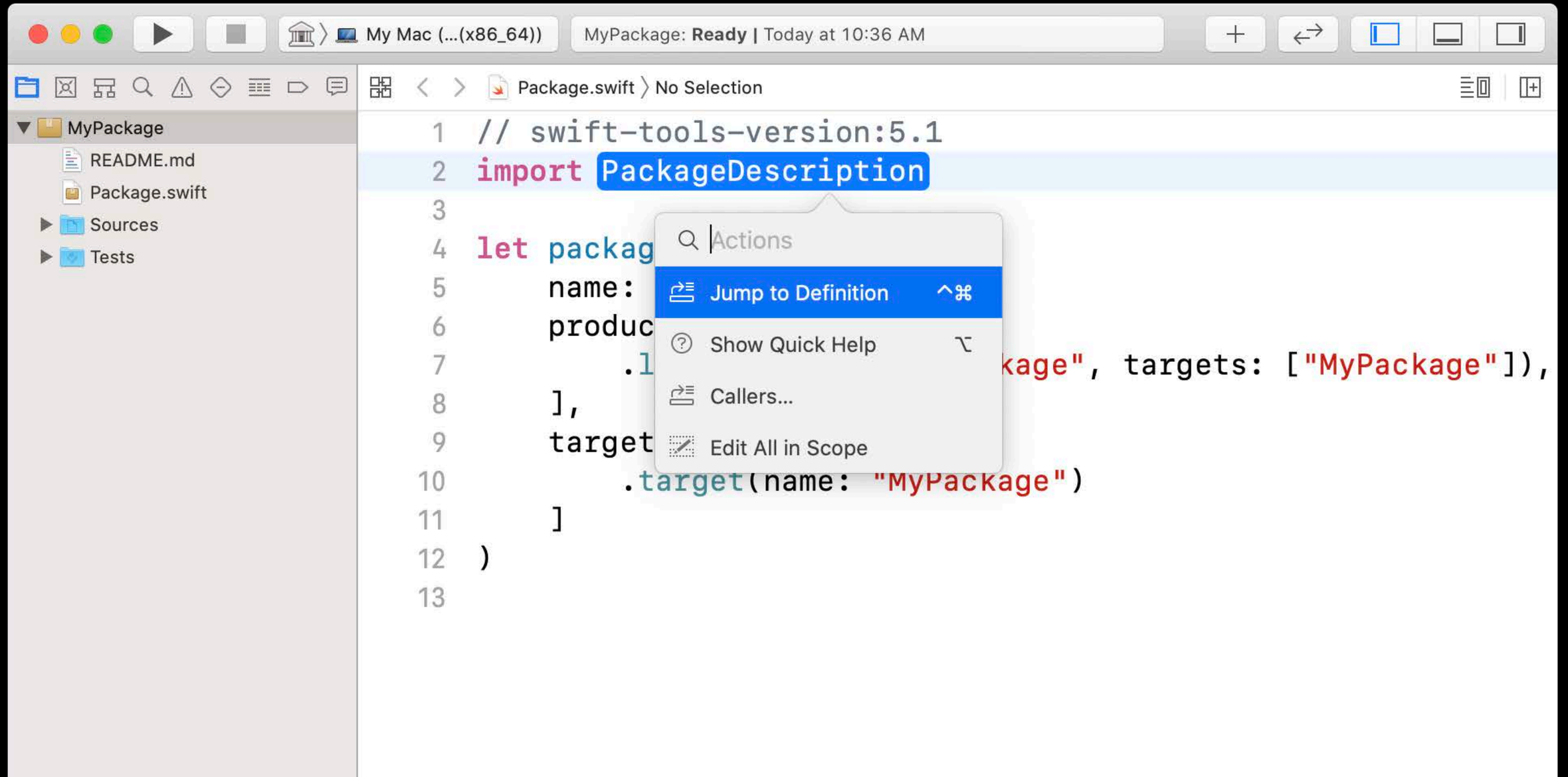
```
// Configuring Minimum Deployment Target Version

let package = Package(
    name: "MyPackage",
    platforms: [
        .macOS("10.15"), .iOS("13"),
    ],
    products: [
         ...
    ],
    dependencies: [
        ...
    ],
    targets: [
         ...
    ]
)
```

Documentation - Generated Module Interface




```
410 extension Package.Dependency {
411
412     /// Add a package dependency that is required from the given minimum version,
413     /// going up to the next major version.
414     ///
415     /// This is the recommend way to specify a remote package dependency because
416     /// it allows you to specify the minimum version you require and gives
417     /// explicit opt-in for new major versions, but otherwise provides maximal
418     /// flexibility on which version is used. This helps to prevent conflicts in
419     /// your package dependency graph.
420     ///
421     /// For example, specifying
422     ///
423     ///     .package(url: "https://example.com/example-package.git", from: "1.2.3"),
424     ///
425     /// will allow the Swift package manager to select a version like a "1.2.3",
426     /// "1.2.4" or "1.3.0" but not "2.0.0".
427     ///
428     /// - Parameters:
429     ///     - url: The valid Git URL of the package.
430     ///     - version: The minimum version requirement.
431     public static func package(url: String, from version: PackageDescription.Version) ->
        PackageDescription.Package.Dependency
432
433     /// Add a remote package dependency given a version requirement.
434     ///
```

Documentation - Generated Module Interface



Editing Packages

Boris Buegling, Developer Tools Engineer

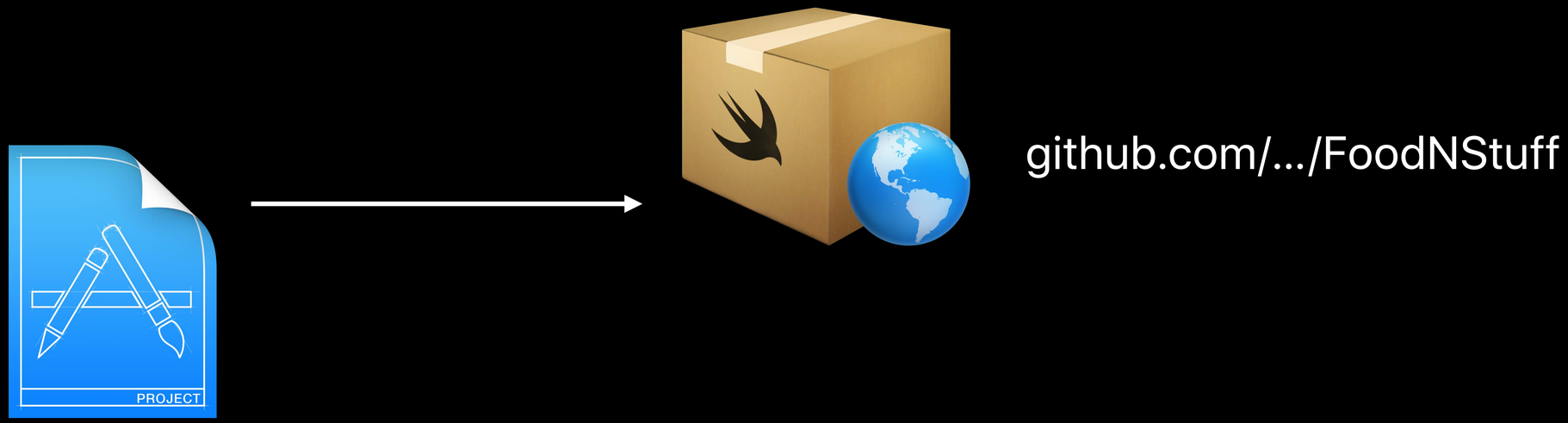
Editing Packages

Local packages can be edited

Same is true for standalone packages

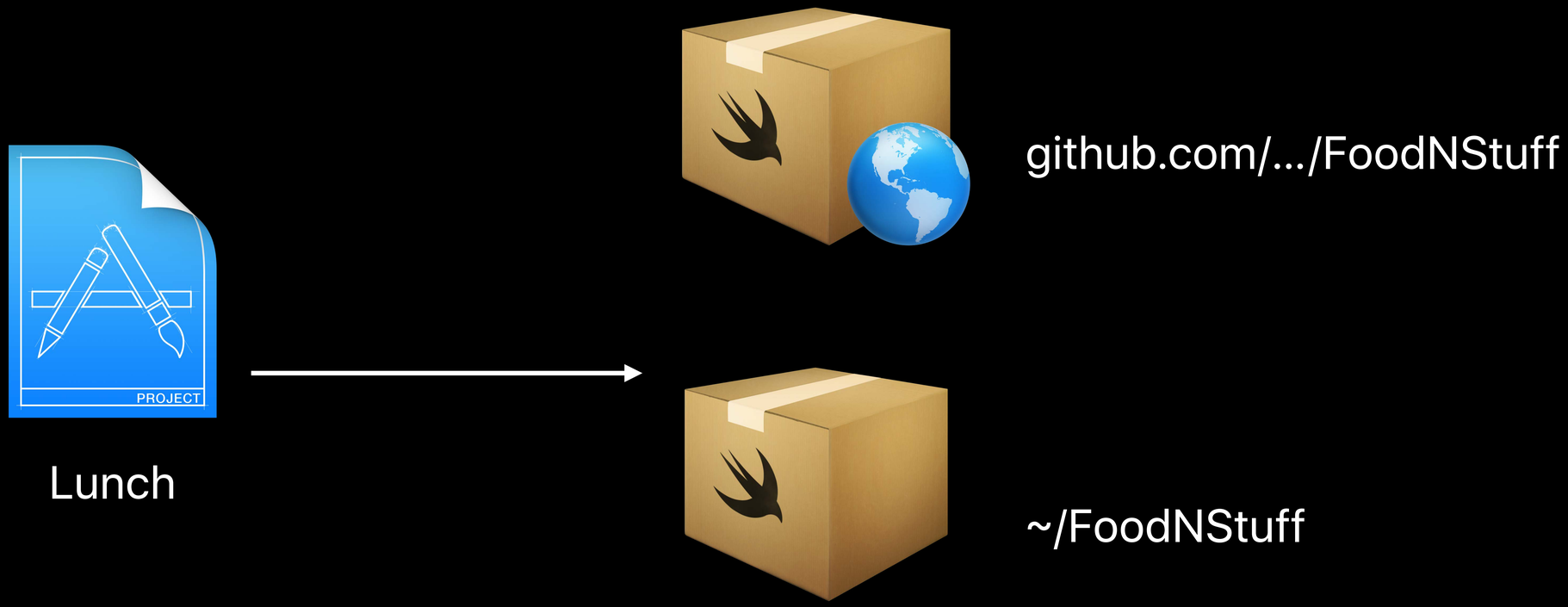
Package dependencies are locked

Overriding Dependencies



Lunch

Overriding Dependencies



Demo

Editing packages

Overriding can be used to
edit packages you do not own.

Open Source Project

Swift Package Manager

A Cross-Platform Build System for Swift

macOS

Linux

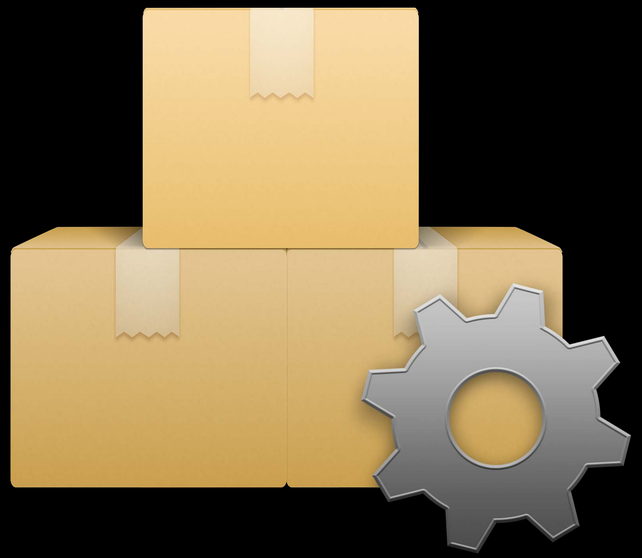
Future Platforms

SwiftPM Commands

SwiftPM Commands

xcodebuild

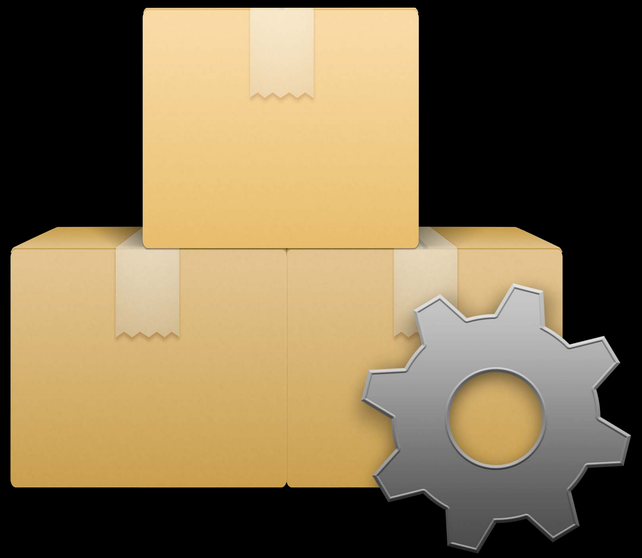
Swift Package Support in Xcode



libSwiftPM

Xcode

Swift Package Support in Xcode

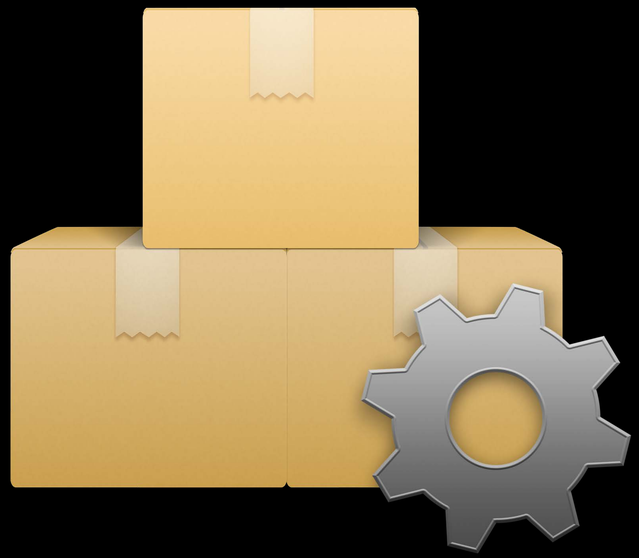


libSwiftPM

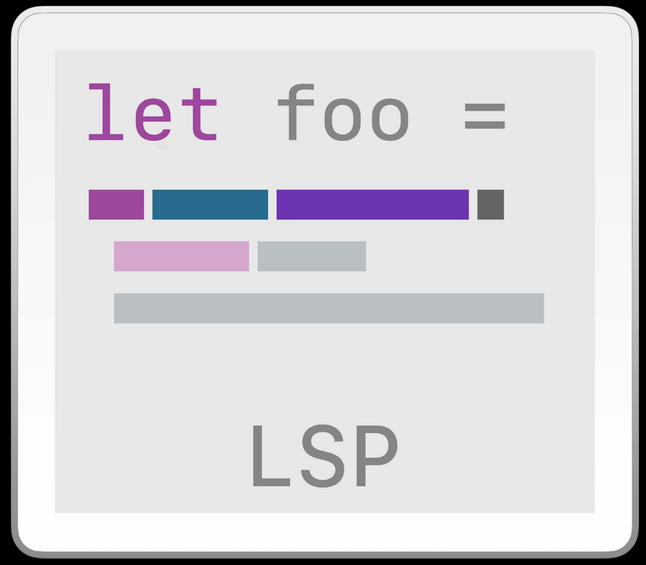
+

Xcode

SourceKit-LSP

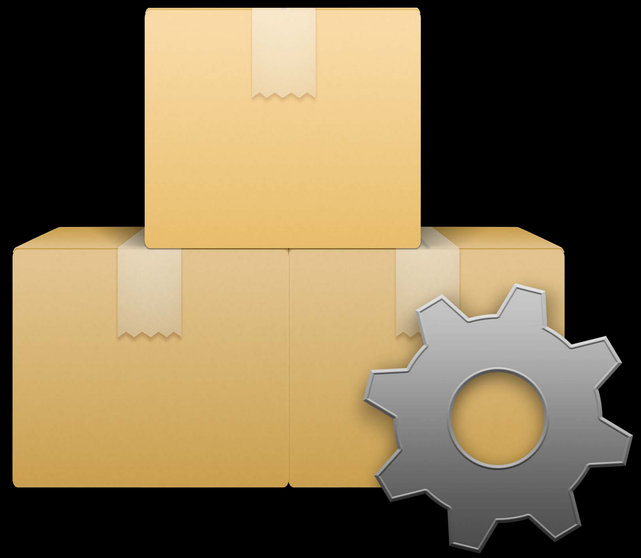


libSwiftPM



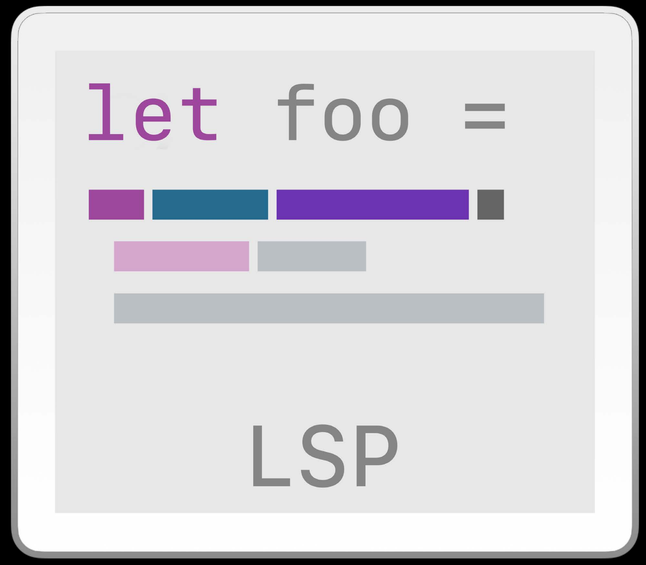
SourceKit-LSP

SourceKit-LSP

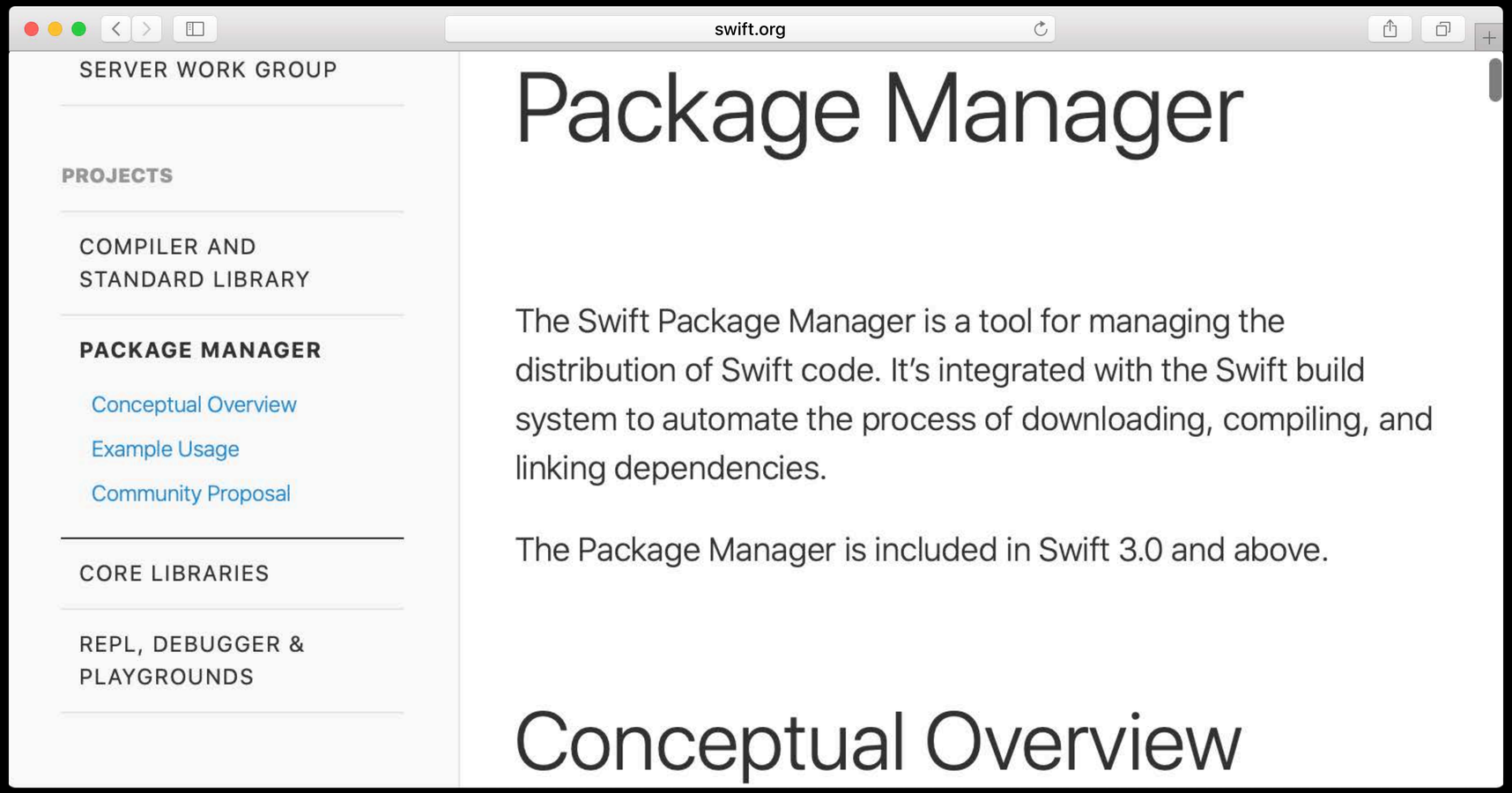


libSwiftPM

+

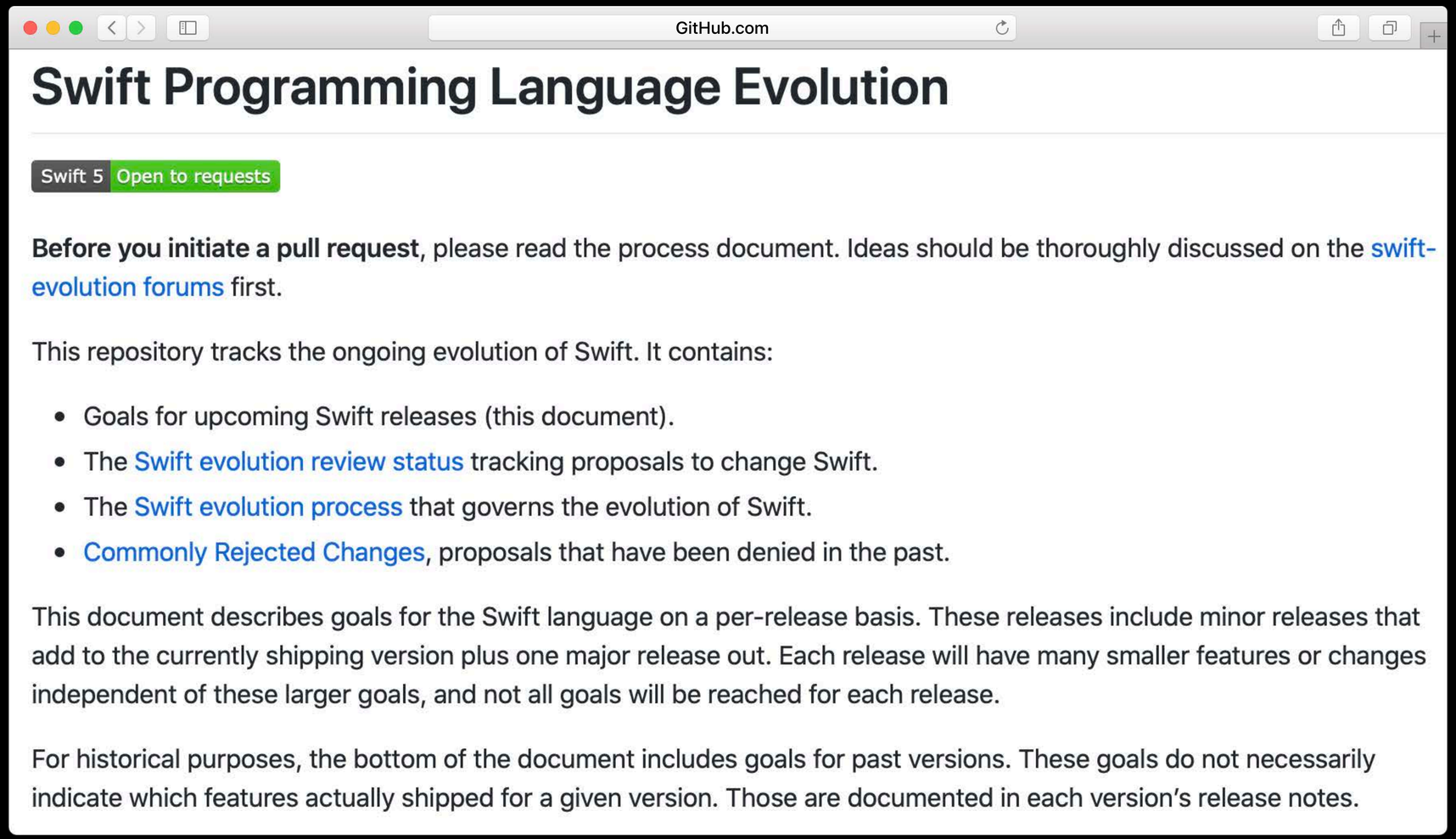


SourceKit-LSP



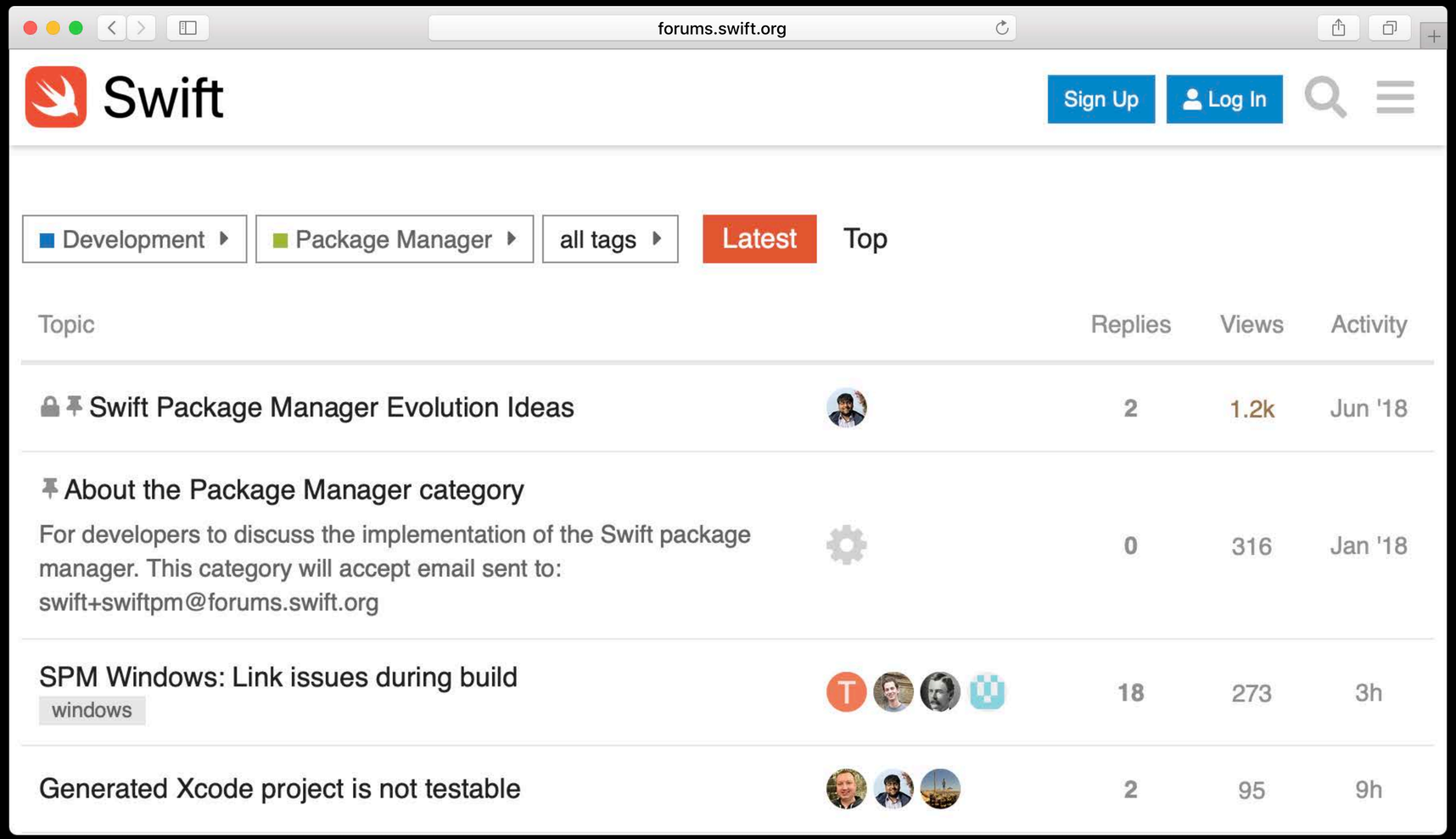
swift.org/package-manager/

Swift Evolution



github.com/apple/swift-evolution

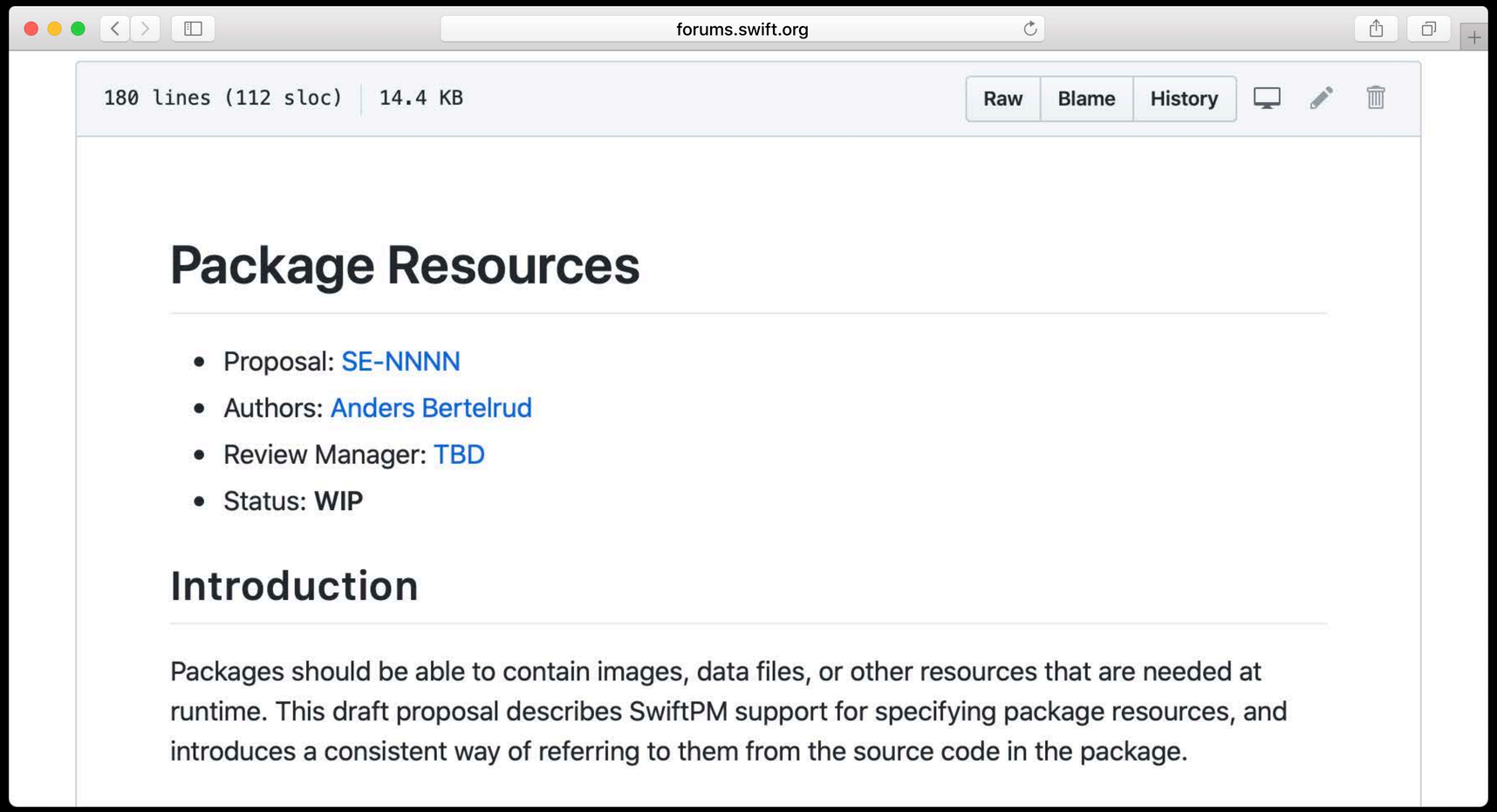
Swift Forums



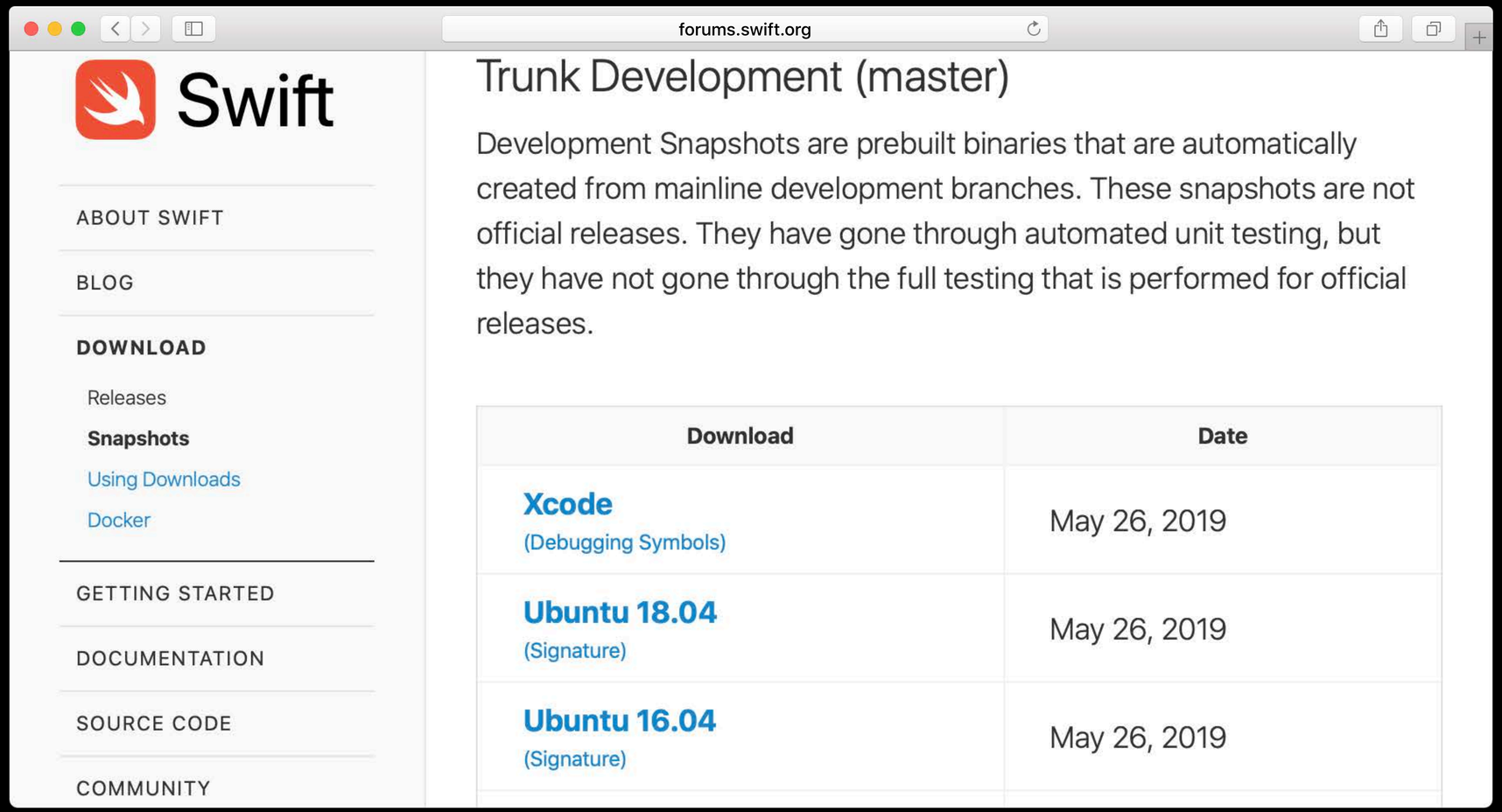
forums.swift.org/c/development/SwiftPM

Package Resources Are Not Supported

Package Resources



Trunk Snapshots



swift.org/download/#snapshots

Final Takeaway

More Information

developer.apple.com/wwdc19/410

Swift Packages Lab	Thursday, 12:00
Swift Packages Lab	Friday, 12:00


WWDC19