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
Initial Development
<table>
<thead>
<tr>
<th>Name</th>
<th>Version Rules</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>BestPackage</td>
<td>5.0.0-beta.1 – Next Major</td>
<td><a href="https://github.com/WWDC19/BestPackage.git">https://github.com/WWDC19/BestPackage.git</a></td>
</tr>
</tbody>
</table>
Swift Package Dependencies

- BestPackage 5.0.0-beta.6
Demo
Publishing packages
Package Manifest API

Ankit Aggarwal, Developer Tools Engineer
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage"
)
// swift-tools-version:5.1

import PackageDescription

let package = Package(
    name: "MyPackage"
)
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage"
)
// swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MyPackage"
)
import PackageDescription

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

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

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

```swift
import PackageDescription

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

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

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

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

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

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

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

```swift
// 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
// 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
// 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

- MenuDownloader
  - MenuDownloader.xcodeproj
  - MenuDownloader.podspec

- LegacyCode
  - C

- SwiftyMenuDownloader
Example — MenuDownloader

- MenuDownloader
  - MenuDownloader.xcodeproj
  - MenuDownloader.podspec
- LegacyCode
  - Swift
  - Swift
- SwiftyMenuDownloader
  - Swift
  - Swift
Example — MenuDownloader

- MenuDownloader
  - MenuDownloader.xcodeproj
  - MenuDownloader.podspec
- LegacyCode
  - LegacyCode
- SwiftyMenuDownloader
Example — MenuDownloader

- MenuDownloader
  - MenuDownloader.xcodeproj
  - MenuDownloader.podspec

- LegacyCode
  - C
  - C

- SwiftyMenuDownloader
  - Swift
  - Go
Example — MenuDownloader

- MenuDownloader
  - Package.swift
  - MenuDownloader.xcodeproj
  - MenuDownloader.podspec
- LegacyCode
  - LegacyCode
- SwiftyMenuDownloader
  - Swift
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")
        ]
    ),
]

MenuDownloader

Package.swift

LegacyCode

SwiftyMenuDownloader
Configuring Targets

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

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: [
        ...
    ]
)
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:5.1
import PackageDescription

let package = Package(
    name: "MenuDownloader",
    ...
    )
/ swift-tools-version:5.1
import PackageDescription

let package = Package(
    name: "MenuDownloader",
    ...
)
// swift-tools-version:5.1
import PackageDescription

let package = Package(
  name: "MenuDownloader",
  ...
)
// swift-tools-version:5.1

import PackageDescription

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

<table>
<thead>
<tr>
<th>macOS</th>
<th>iOS</th>
<th>tvOS</th>
<th>watchOS</th>
<th>Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.10</td>
<td>8.0</td>
<td>9.0</td>
<td>2.0</td>
<td></td>
</tr>
</tbody>
</table>

```c
#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: [
        ...
    ]
)
```swift
extension Package.Dependency {

    /// Add a package dependency that is required from the given minimum version,
    /// going up to the next major version.
    ///
    /// This is the recommend way to specify a remote package dependency because
    /// it allows you to specify the minimum version you require and gives
    /// explicit opt-in for new major versions, but otherwise provides maximal
    /// flexibility on which version is used. This helps to prevent conflicts in
    /// your package dependency graph.
    ///
    /// For example, specifying
    ///
    ///     .package(url: "https://example.com/example-package.git", from: "1.2.3"),
    ///
    /// will allow the Swift package manager to select a version like a "1.2.3",
    /// "1.2.4" or "1.3.0" but not "2.0.0".
    ///
    /// - Parameters:
    ///   - url: The valid Git URL of the package.
    ///   - version: The minimum version requirement.

    public static func package(url: String, from version: PackageDescription.Version) ->
    PackageDescription.Package.Dependency

    /// Add a remote package dependency given a version requirement.
```
Documentation - Generated Module Interface

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

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

Local packages can be edited
Same is true for standalone packages
Package dependencies are locked
Overriding Dependencies

github.com/.../FoodNStuff
Overriding Dependencies

github.com/.../FoodNStuff

~/FoodNStuff
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

$ swift build
$ swift run
$ swift test
$ swift package
xcodebuild
Swift Package Support in Xcode

libSwiftPM

Xcode
Swift Package Support in Xcode

libSwiftPM + Xcode
SourceKit-LSP

libSwiftPM

let foo = LSP

SourceKit-LSP
The Swift Package Manager is a tool for managing the distribution of Swift code. It's integrated with the Swift build system to automate the process of downloading, compiling, and linking dependencies.

The Package Manager is included in Swift 3.0 and above.
Before you initiate a pull request, please read the process document. Ideas should be thoroughly discussed on the swift-evolution forums first.

This repository tracks the ongoing evolution of Swift. It contains:

- Goals for upcoming Swift releases (this document).
- The Swift evolution review status tracking proposals to change Swift.
- The Swift evolution process that governs the evolution of Swift.
- Commonly Rejected Changes, proposals that have been denied in the past.

This document describes goals for the Swift language on a per-release basis. These releases include minor releases that add to the currently shipping version plus one major release out. Each release will have many smaller features or changes independent of these larger goals, and not all goals will be reached for each release.

For historical purposes, the bottom of the document includes goals for past versions. These goals do not necessarily indicate which features actually shipped for a given version. Those are documented in each version's release notes.

github.com/apple/swift-evolution
Swift Forums

forums.swift.org/c/development/SwiftPM
Package Resources Are Not Supported
Package Resources

- Proposal: SE-NNNN
- Authors: Anders Bertelrud
- Review Manager: TBD
- Status: WIP

Introduction

Packages should be able to contain images, data files, or other resources that are needed at runtime. This draft proposal describes SwiftPM support for specifying package resources, and introduces a consistent way of referring to them from the source code in the package.
Trunk Snapshots

Trunk Development (master)

Development Snapshots are prebuilt binaries that are automatically created from mainline development branches. These snapshots are not official releases. They have gone through automated unit testing, but they have not gone through the full testing that is performed for official releases.

<table>
<thead>
<tr>
<th>Download</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xcode (Debugging Symbols)</td>
<td>May 26, 2019</td>
</tr>
<tr>
<td>Ubuntu 18.04 (Signature)</td>
<td>May 26, 2019</td>
</tr>
<tr>
<td>Ubuntu 16.04 (Signature)</td>
<td>May 26, 2019</td>
</tr>
</tbody>
</table>

swift.org/download/#snapshots
Final Takeaway
More Information

developer.apple.com/wwdc19/410

<table>
<thead>
<tr>
<th>Swift Packages Lab</th>
<th>Thursday, 12:00</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swift Packages Lab</td>
<td>Friday, 12:00</td>
</tr>
</tbody>
</table>