

#WWDC18

Getting to Know Swift Package Manager

Session 411

Rick Ballard, SwiftPM Release Manager
Boris Buegling, Developer Tools Engineer

Why a package manager for Swift?

How to use it

The design of SwiftPM

Evolution ideas

Open source process

Why a package manager for Swift?

How to use it

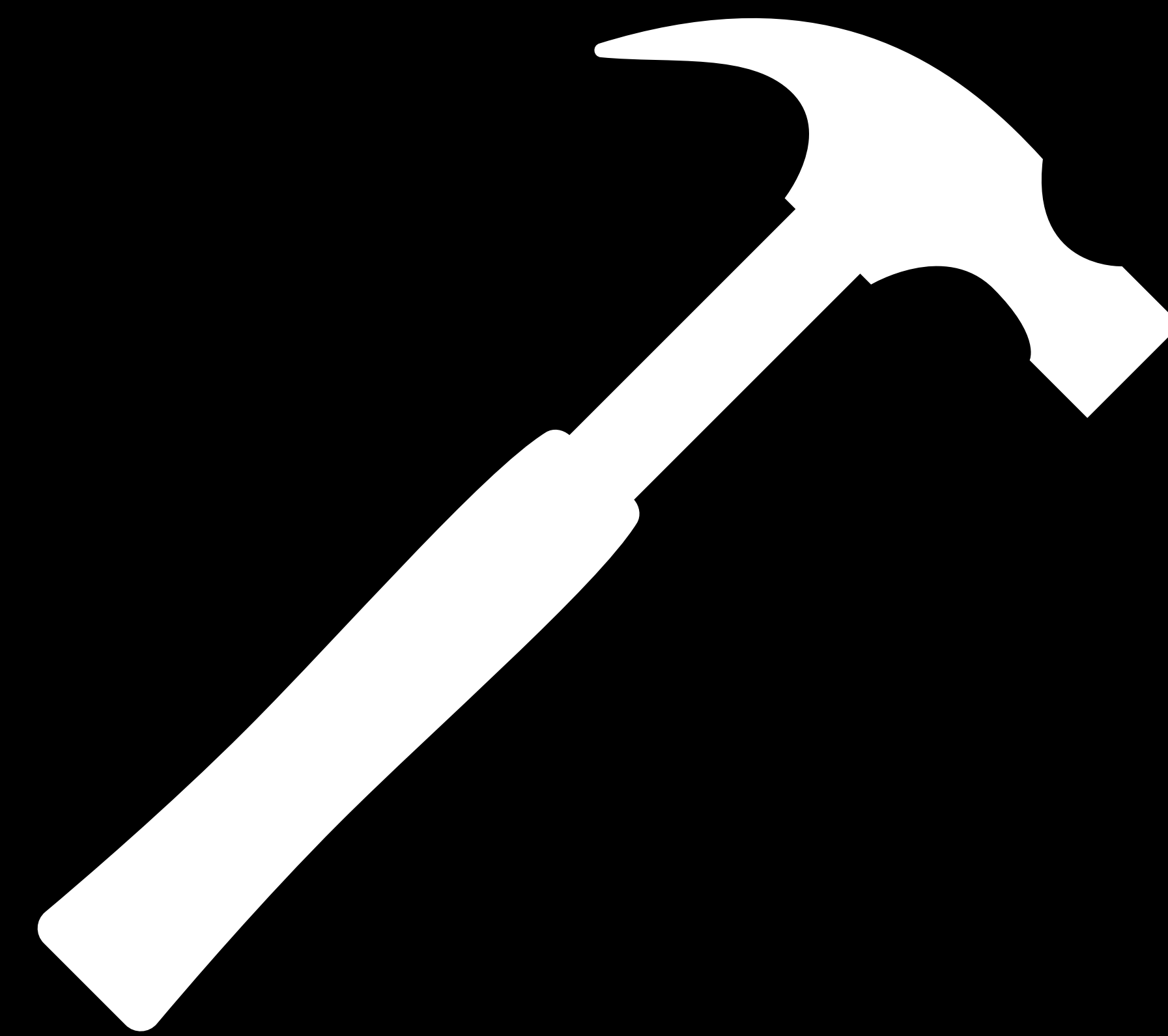
The design of SwiftPM

Evolution ideas

Open source process

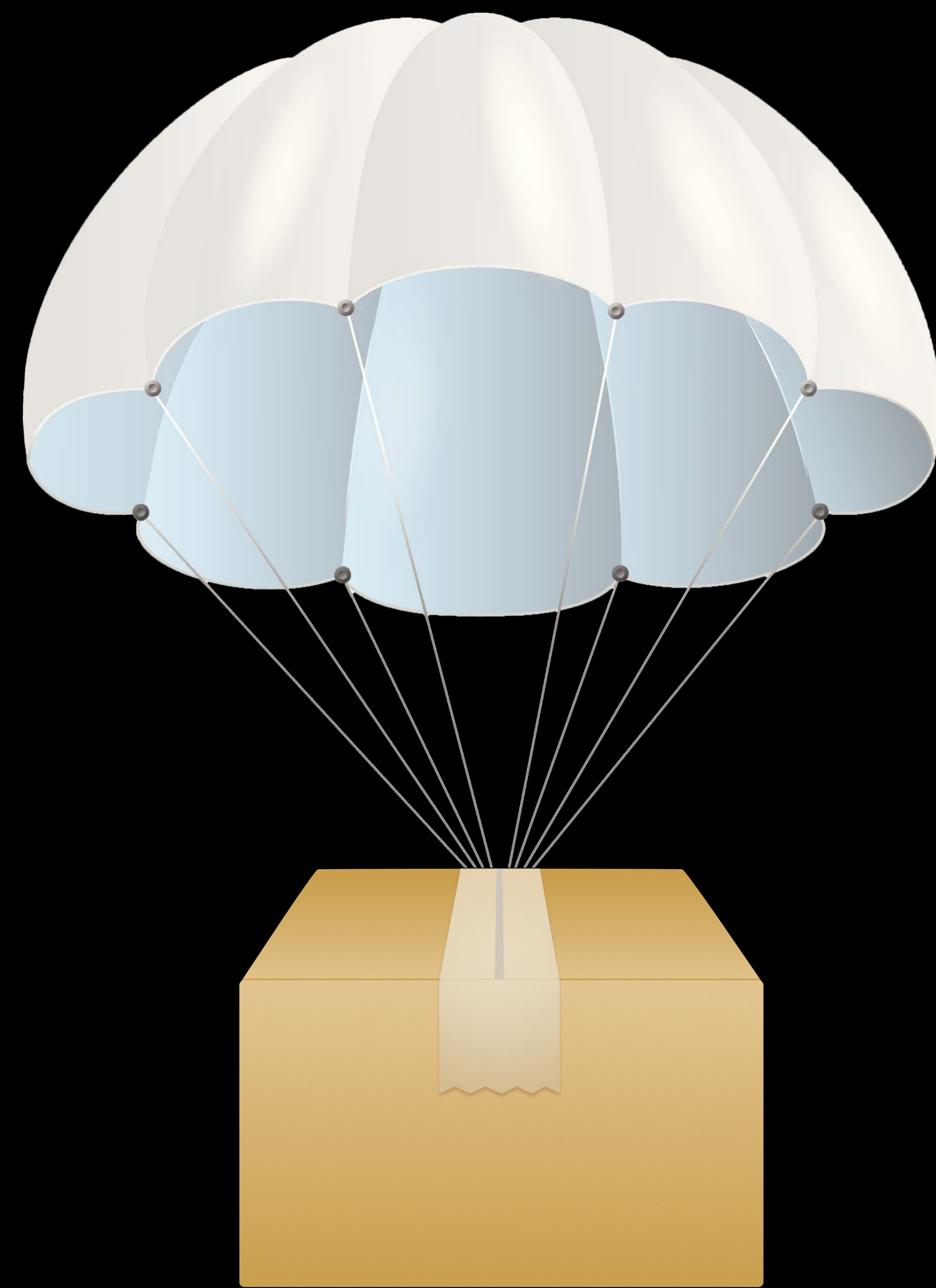
A Cross-Platform Build System for Swift

A Cross-Platform Build System for Swift



Canonical Package Management Tool

Canonical Package Management Tool



mac
OS

Ubuntu

Future
Platforms

Code Reuse Beyond the Core Libraries



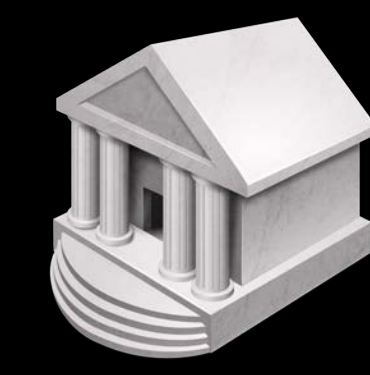
Foundation



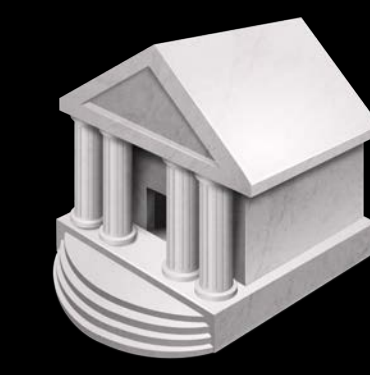
Dispatch



XCTest

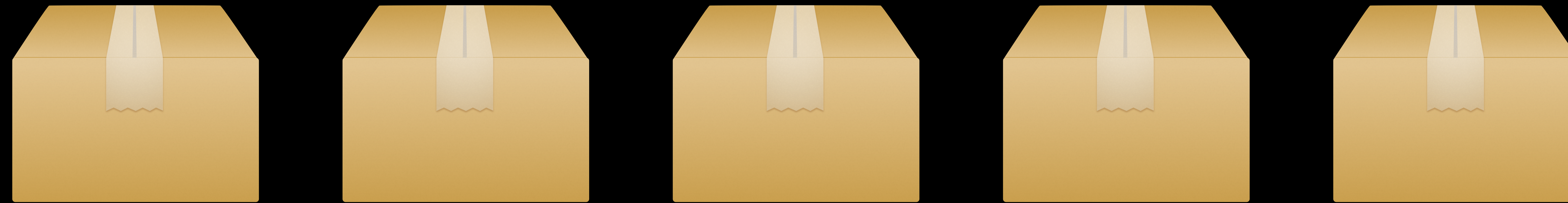


stdlib



system

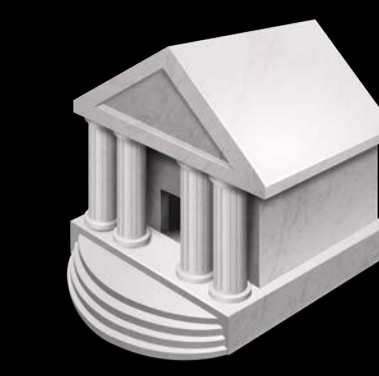
Code Reuse Beyond the Core Libraries



Foundation



Dispatch



XCTest



stdlib



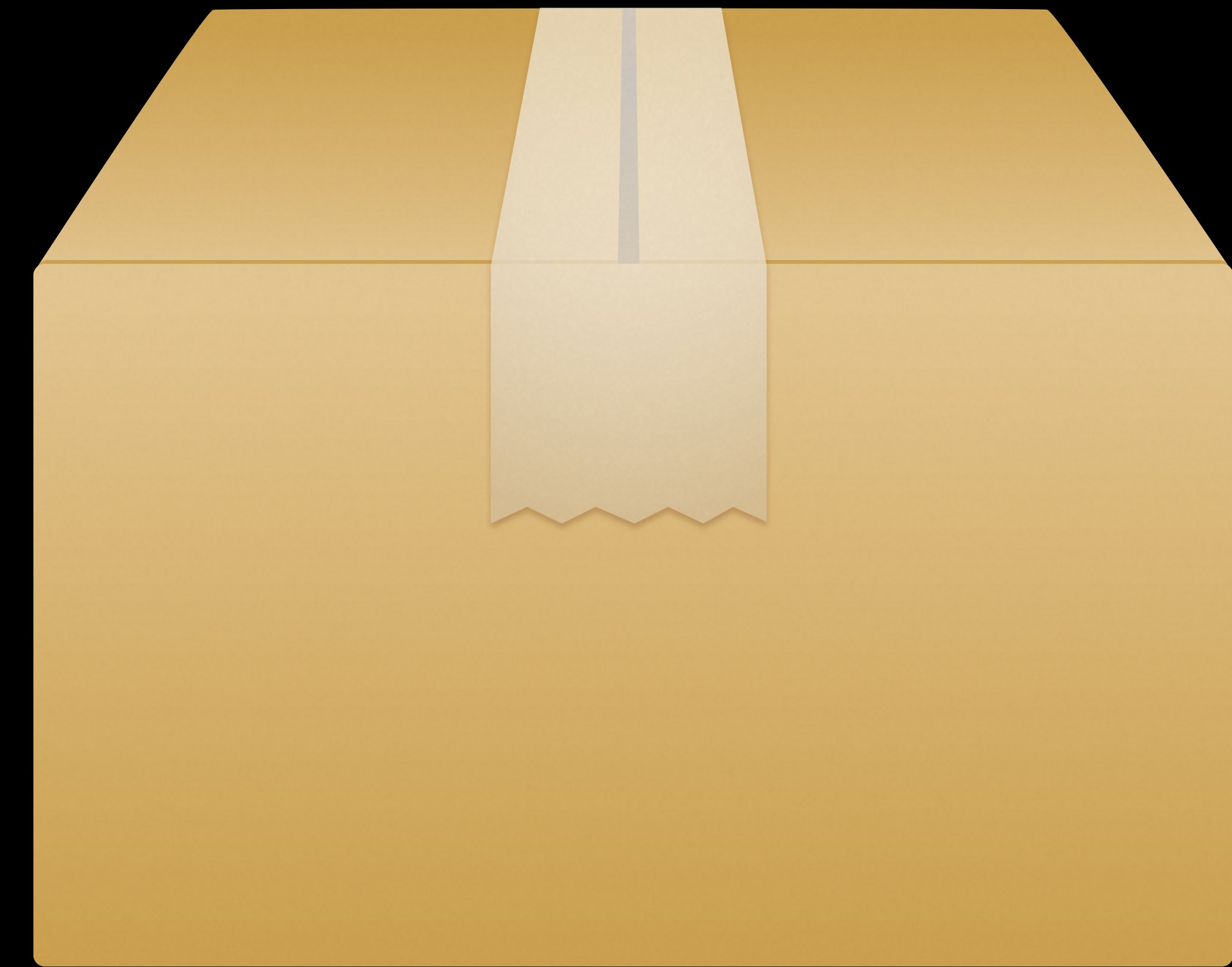
system

Take Full Advantage of the Power of Swift

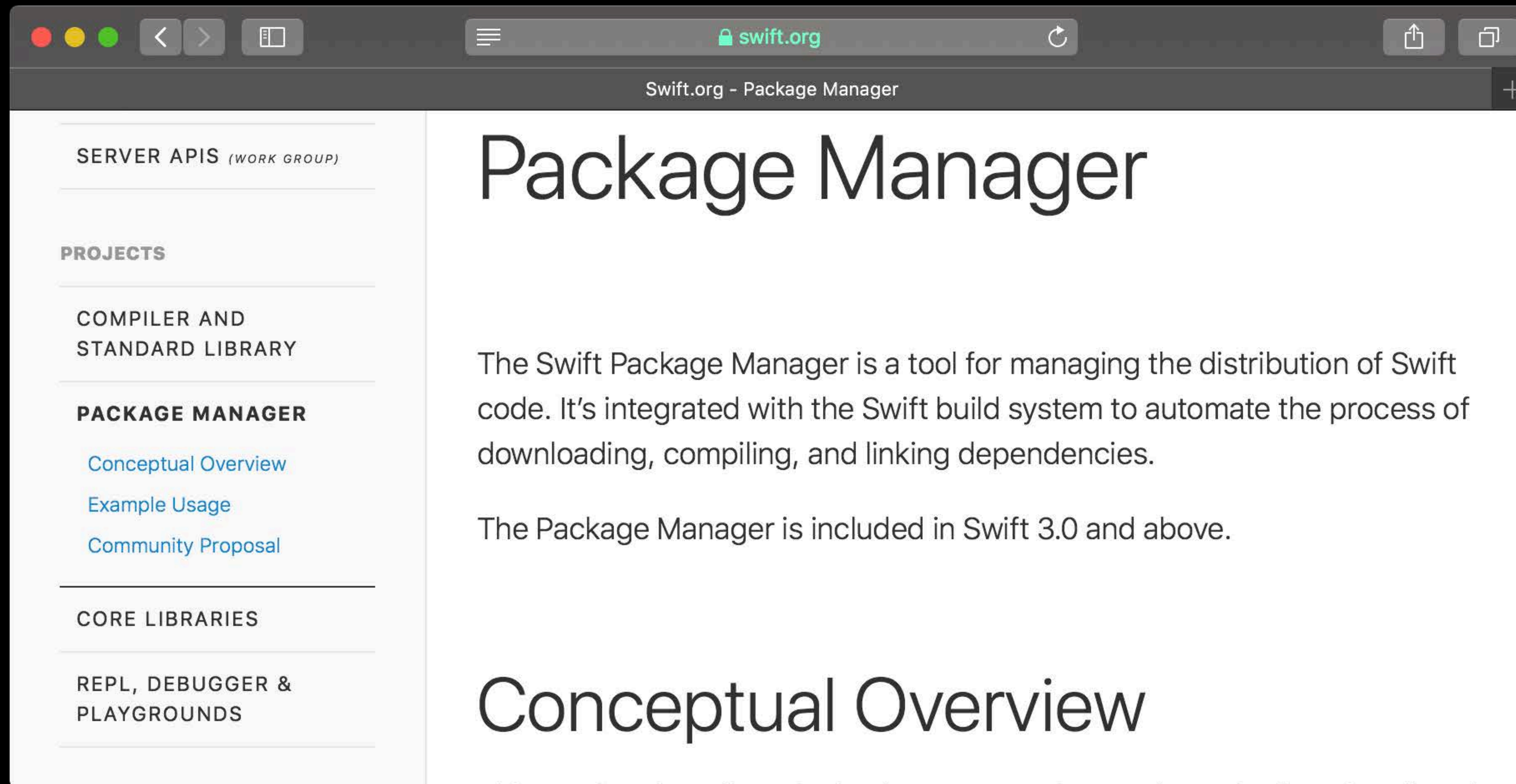
Take Full Advantage of the Power of Swift



+

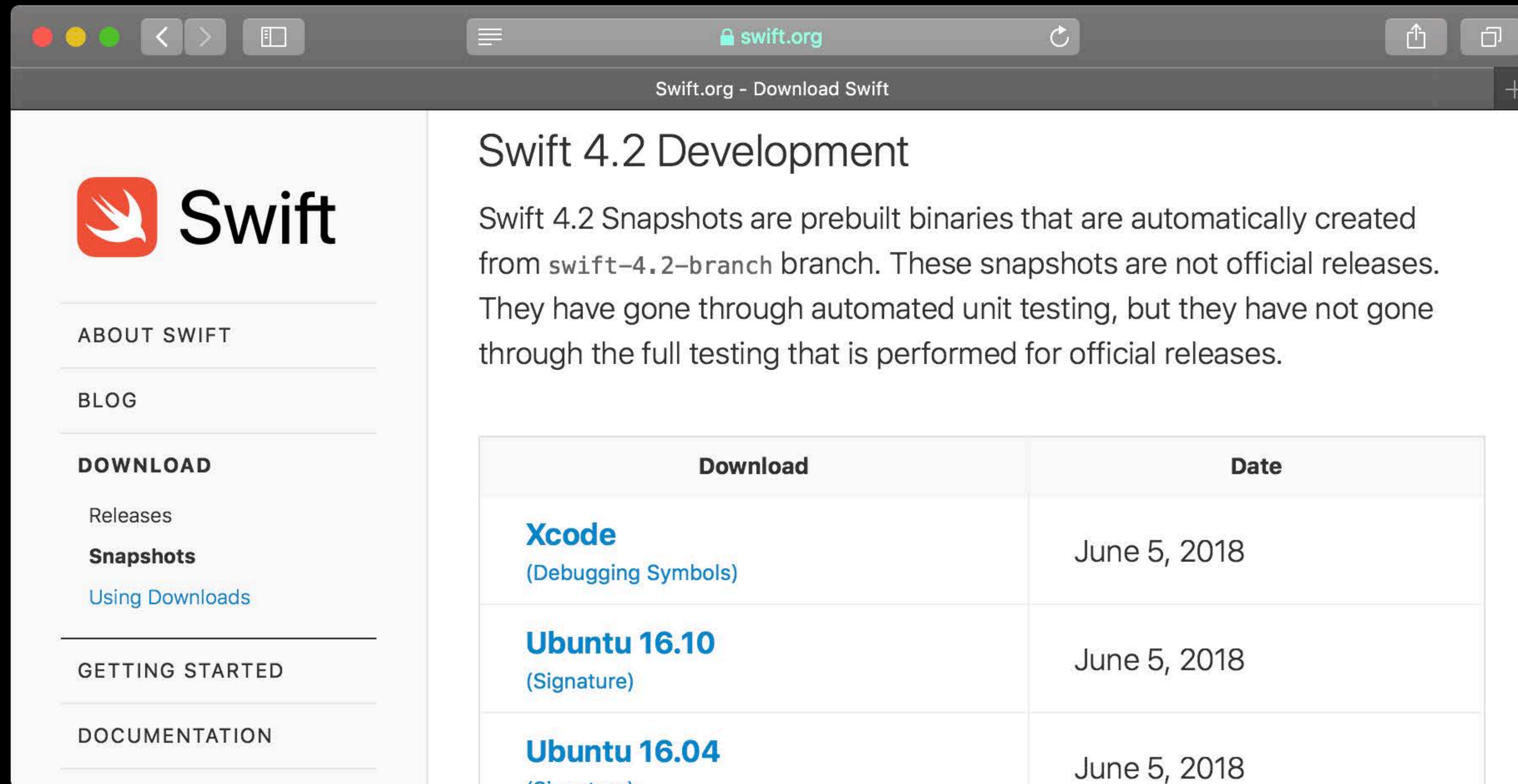


Part of Swift Open Source Project



<https://swift.org/package-manager/>

Included in Swift Toolchains



The screenshot shows a web browser window with the URL swift.org. The page title is "Swift.org - Download Swift". The main content area is titled "Swift 4.2 Development" and contains the following text:

Swift 4.2 Snapshots are prebuilt binaries that are automatically created from `swift-4.2-branch` branch. 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.

Below the text is a table with two columns: "Download" and "Date".

Download	Date
Xcode (Debugging Symbols)	June 5, 2018
Ubuntu 16.10 (Signature)	June 5, 2018
Ubuntu 16.04 (Signature)	June 5, 2018

The left sidebar contains navigation links: ABOUT SWIFT, BLOG, DOWNLOAD (with sub-links for Releases, **Snapshots**, and Using Downloads), GETTING STARTED, and DOCUMENTATION.

<https://swift.org/download/>

Included with Xcode



Why a package manager for Swift?

How to use it

The design of SwiftPM

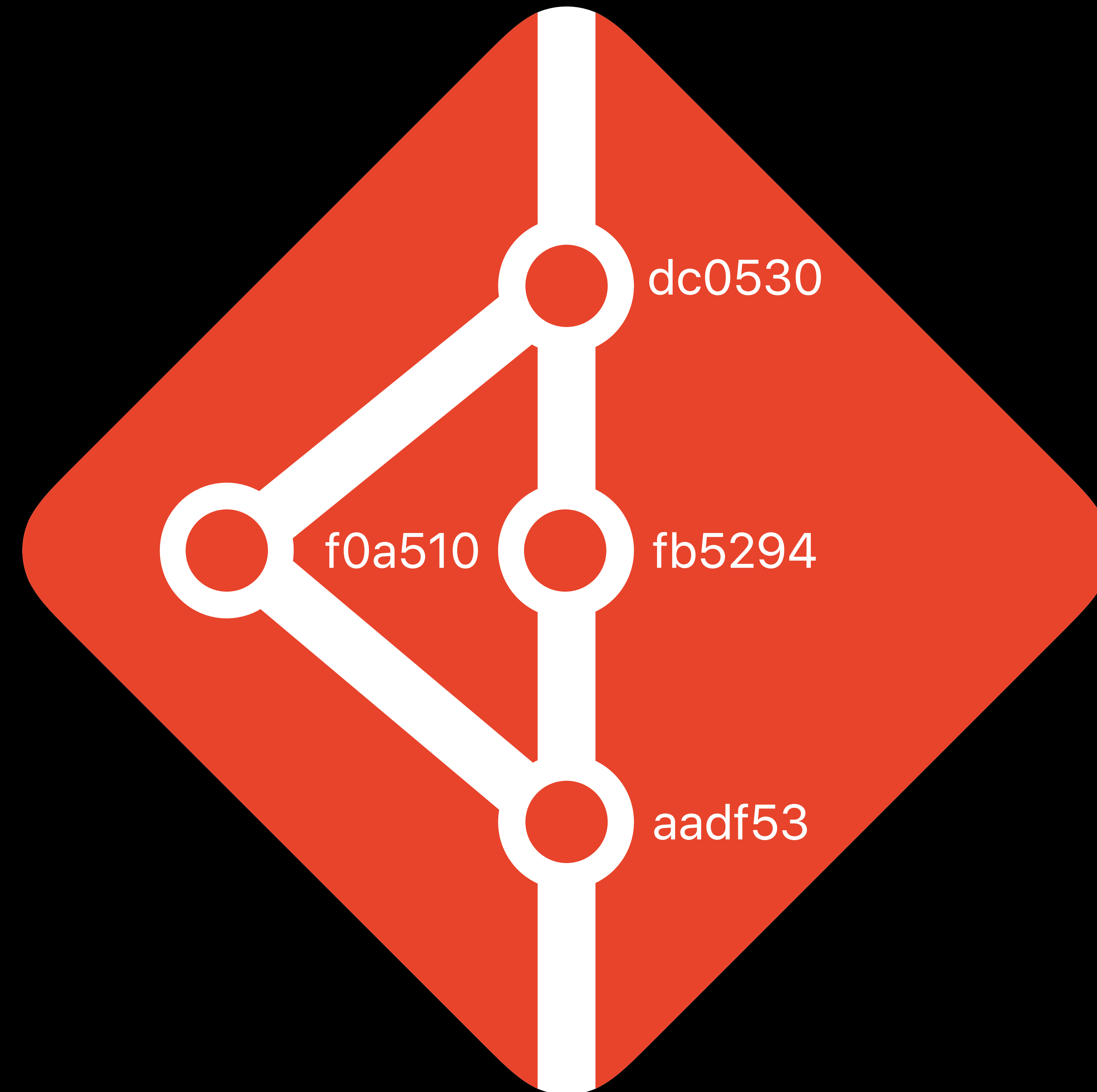
Evolution ideas

Open source process

SwiftPM Commands

```
demo — -bash — 53x16
$ swift build
$ swift run
$ swift test
$ swift package
```

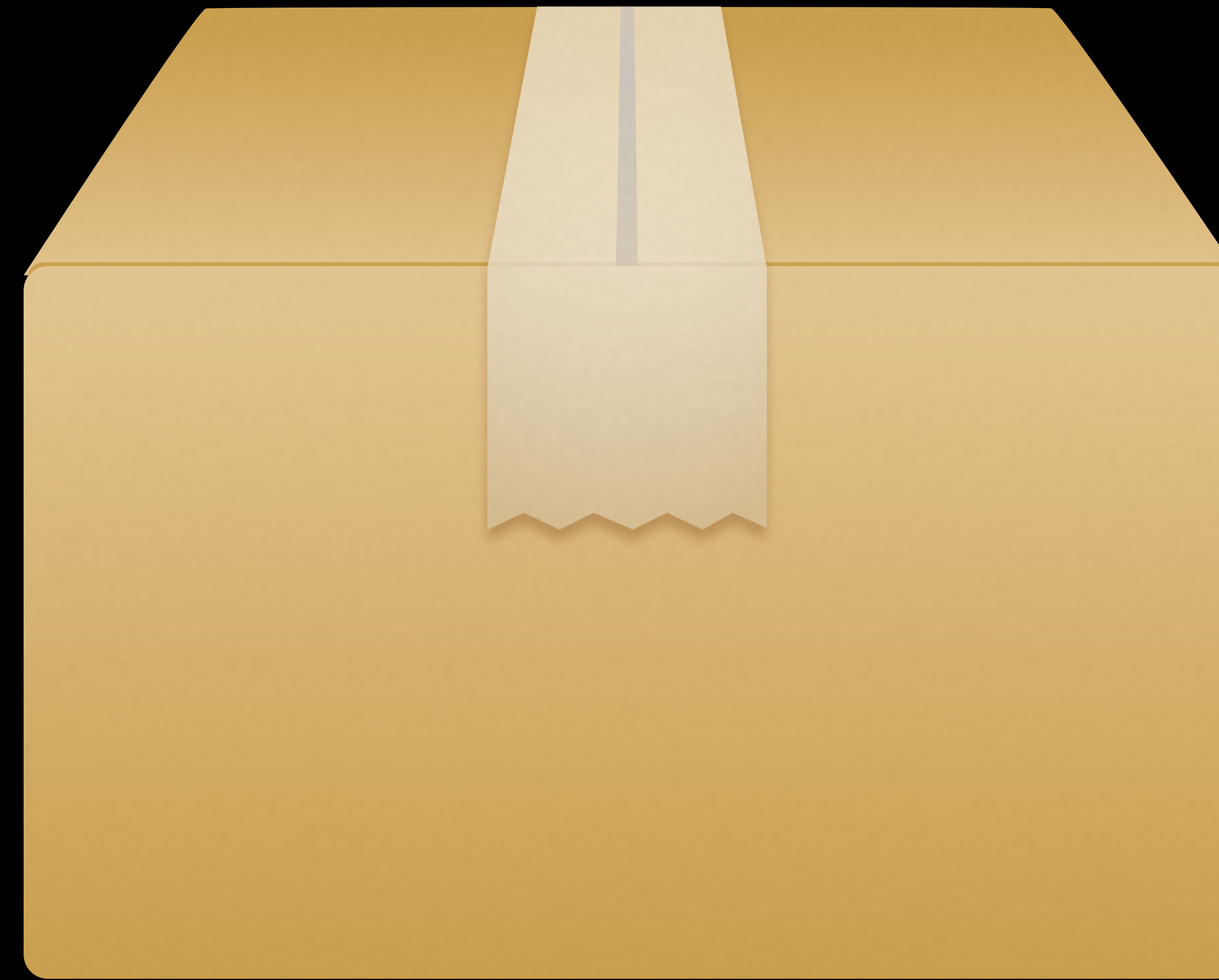

Packages Use Git



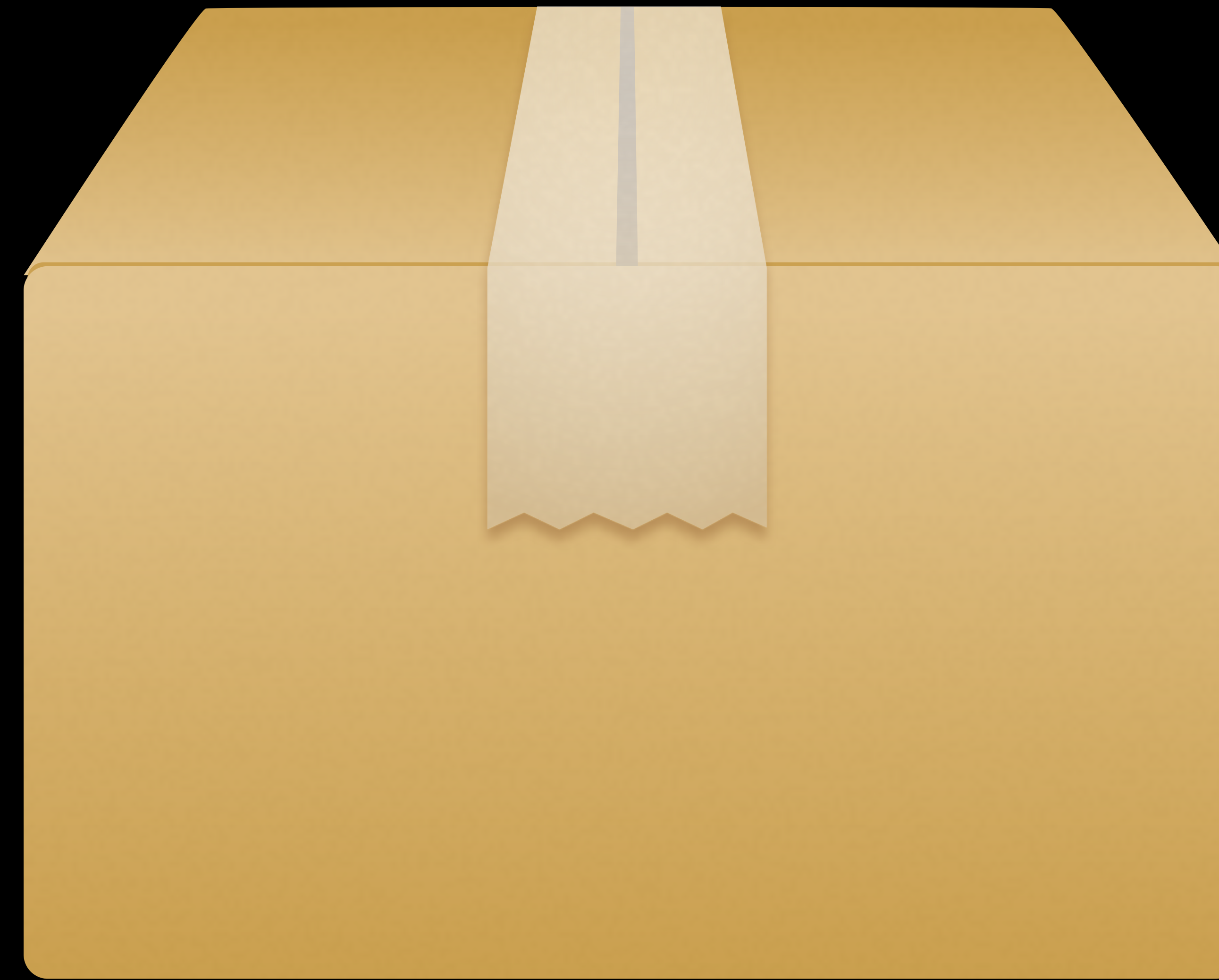
Demo

Creating your first package

Anatomy of a Package



Anatomy of a Package

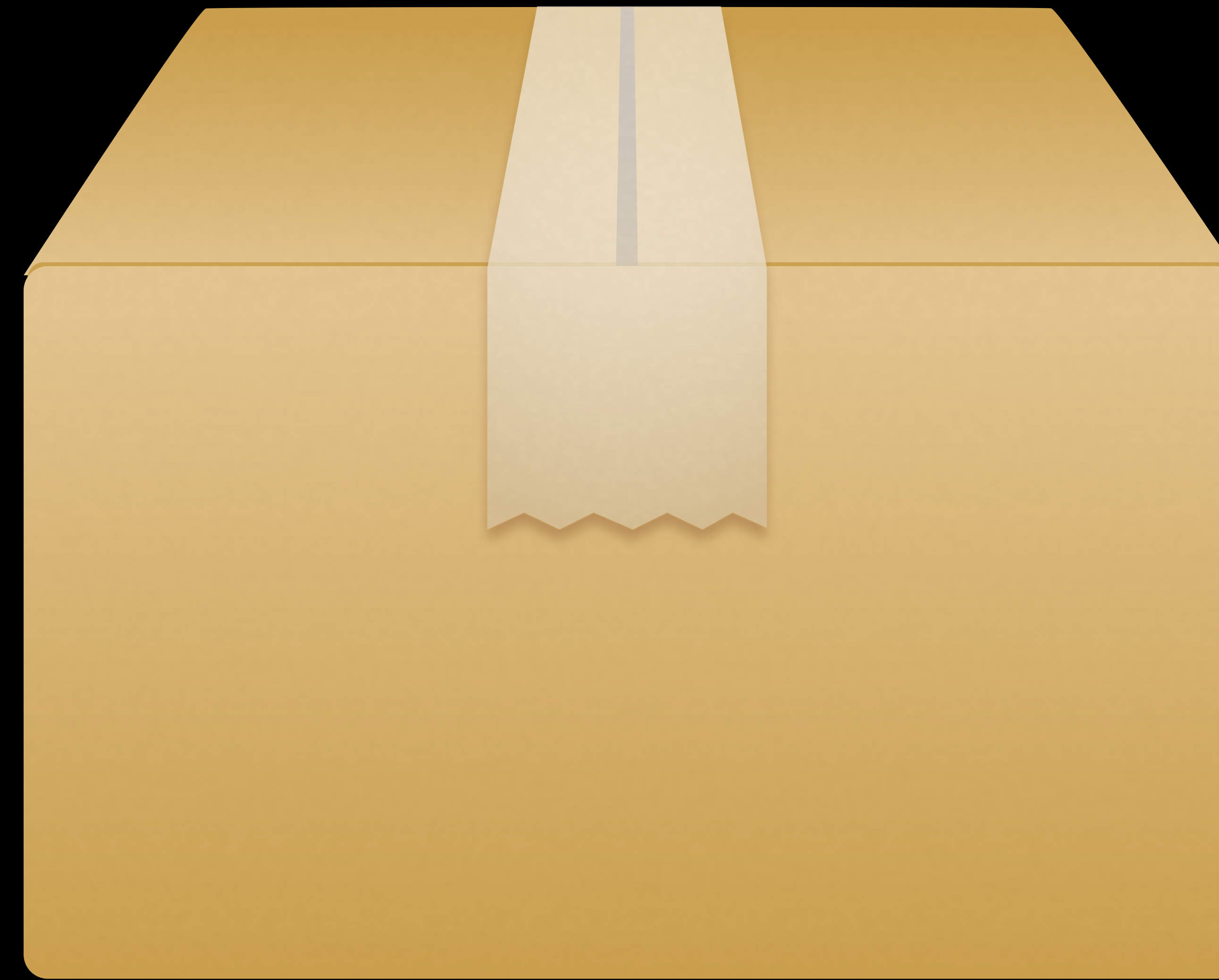


Dependencies

Anatomy of a Package



Targets

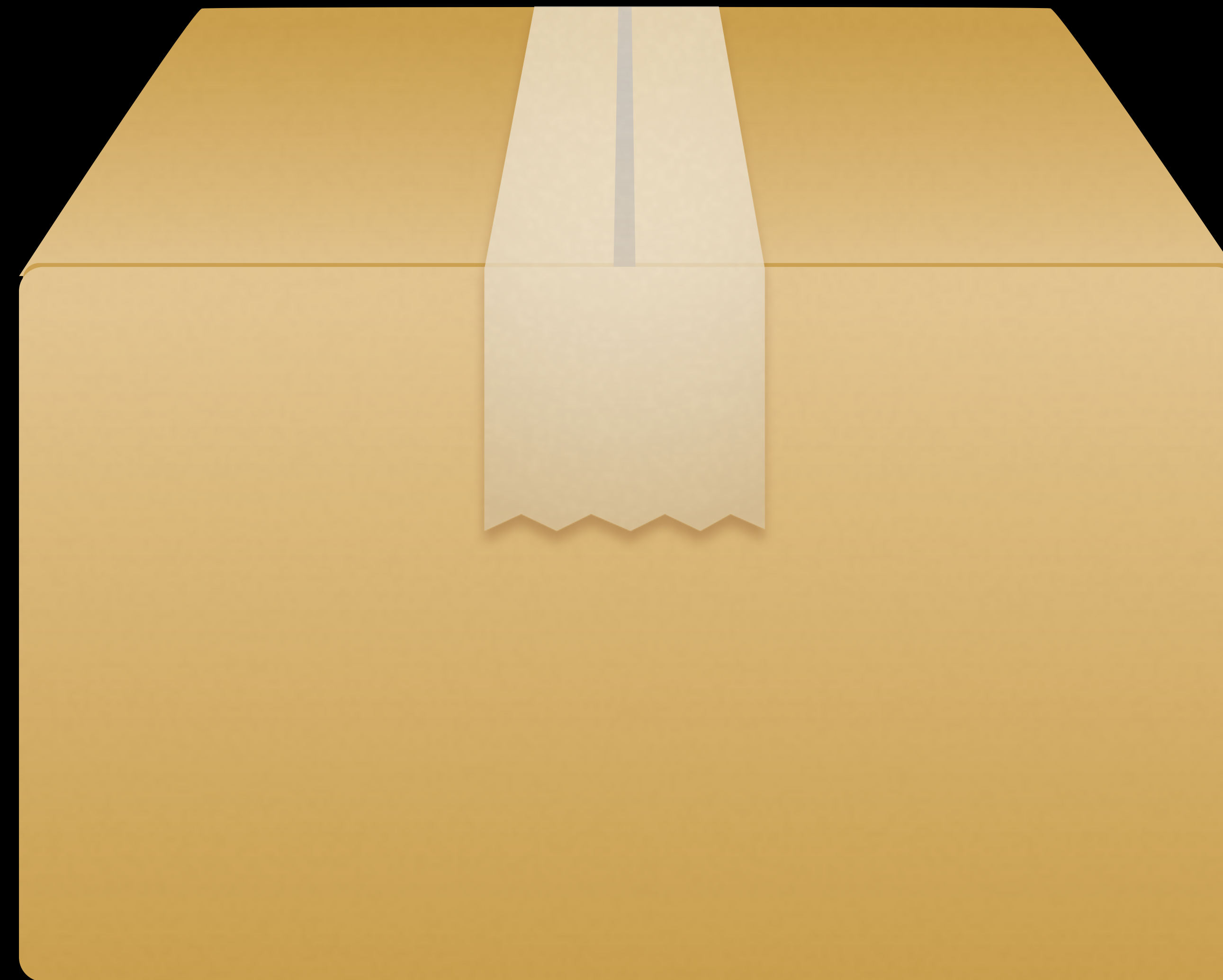


Dependencies

Anatomy of a Package



Targets

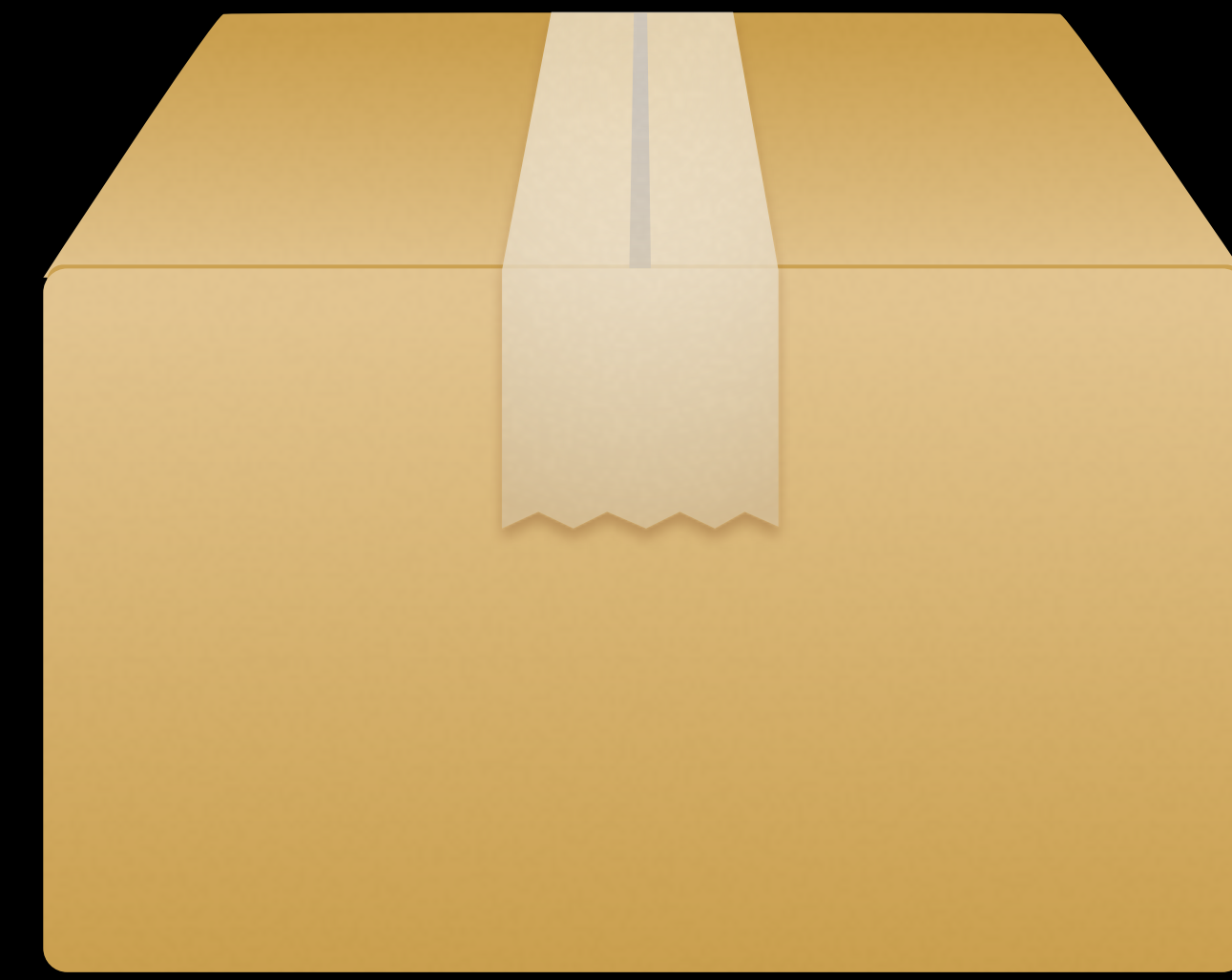


Products

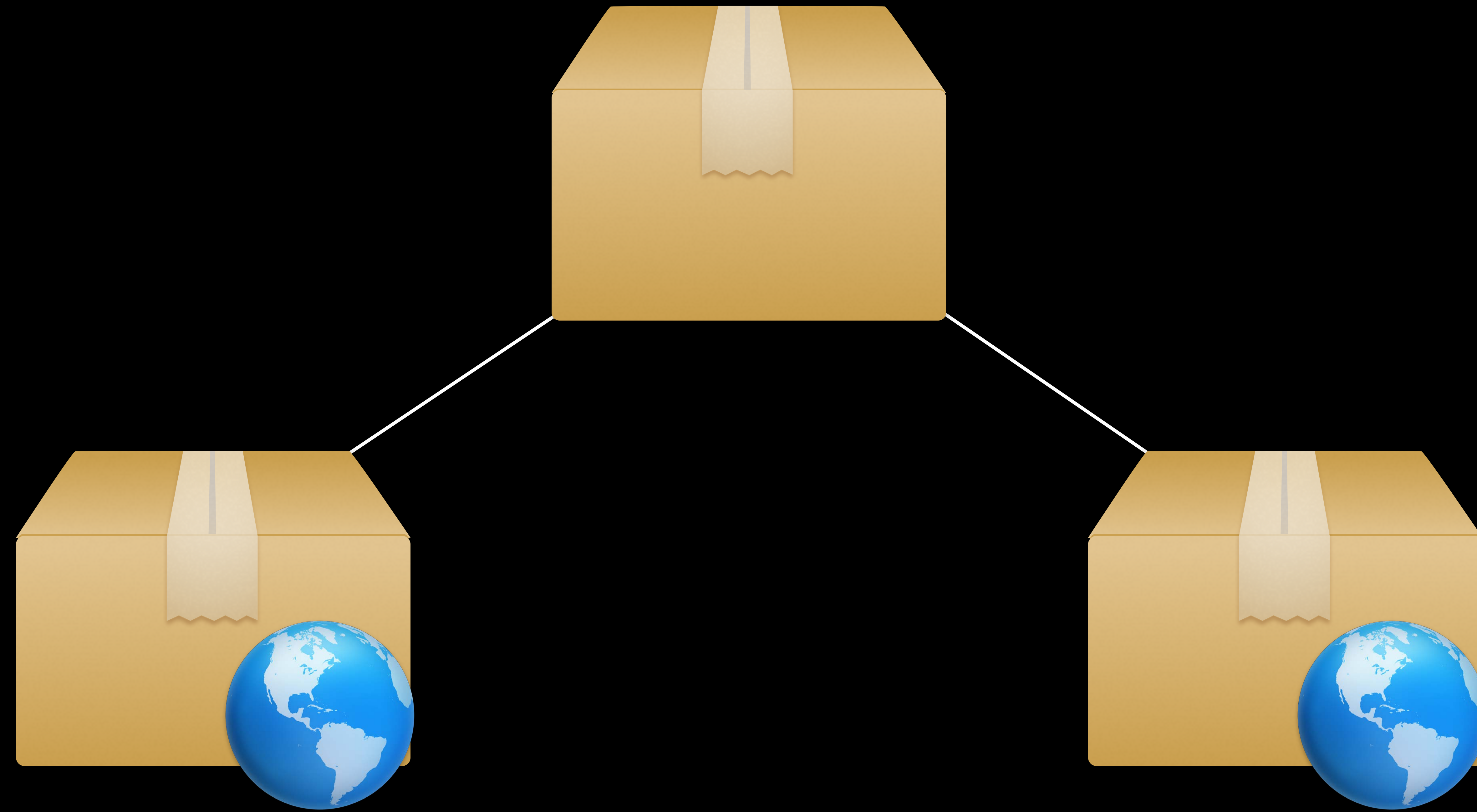


Dependencies

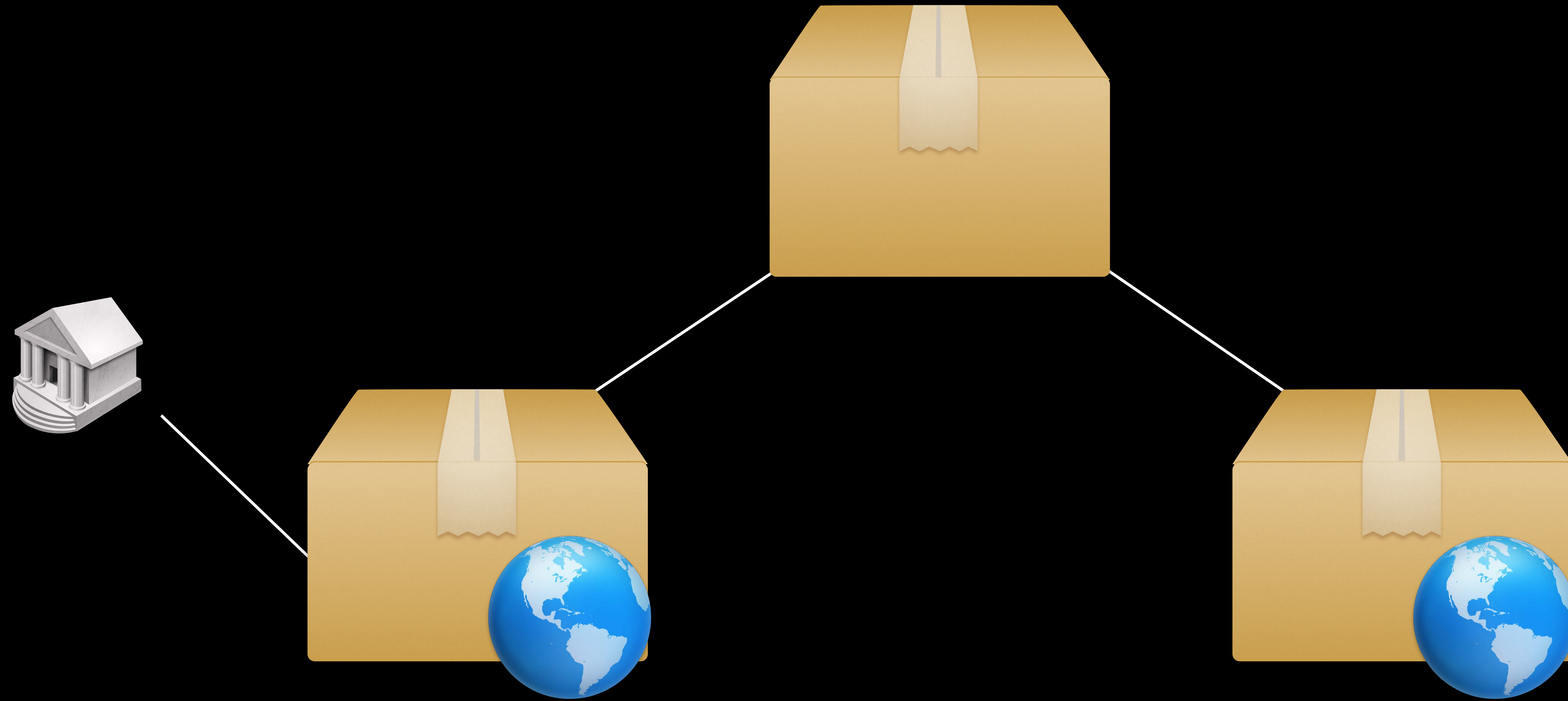
Dependencies



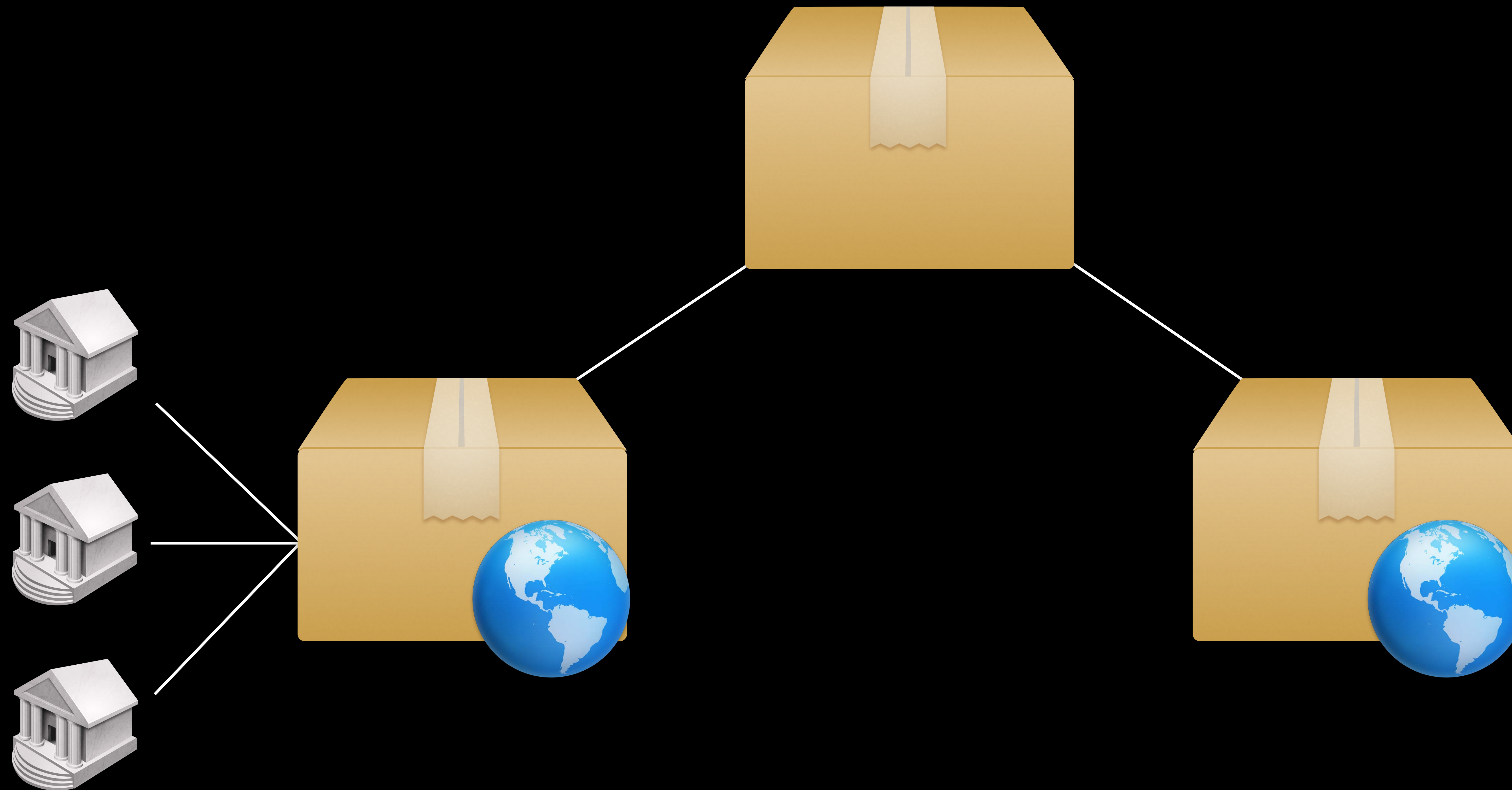
Dependencies



Dependencies



Dependencies



```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

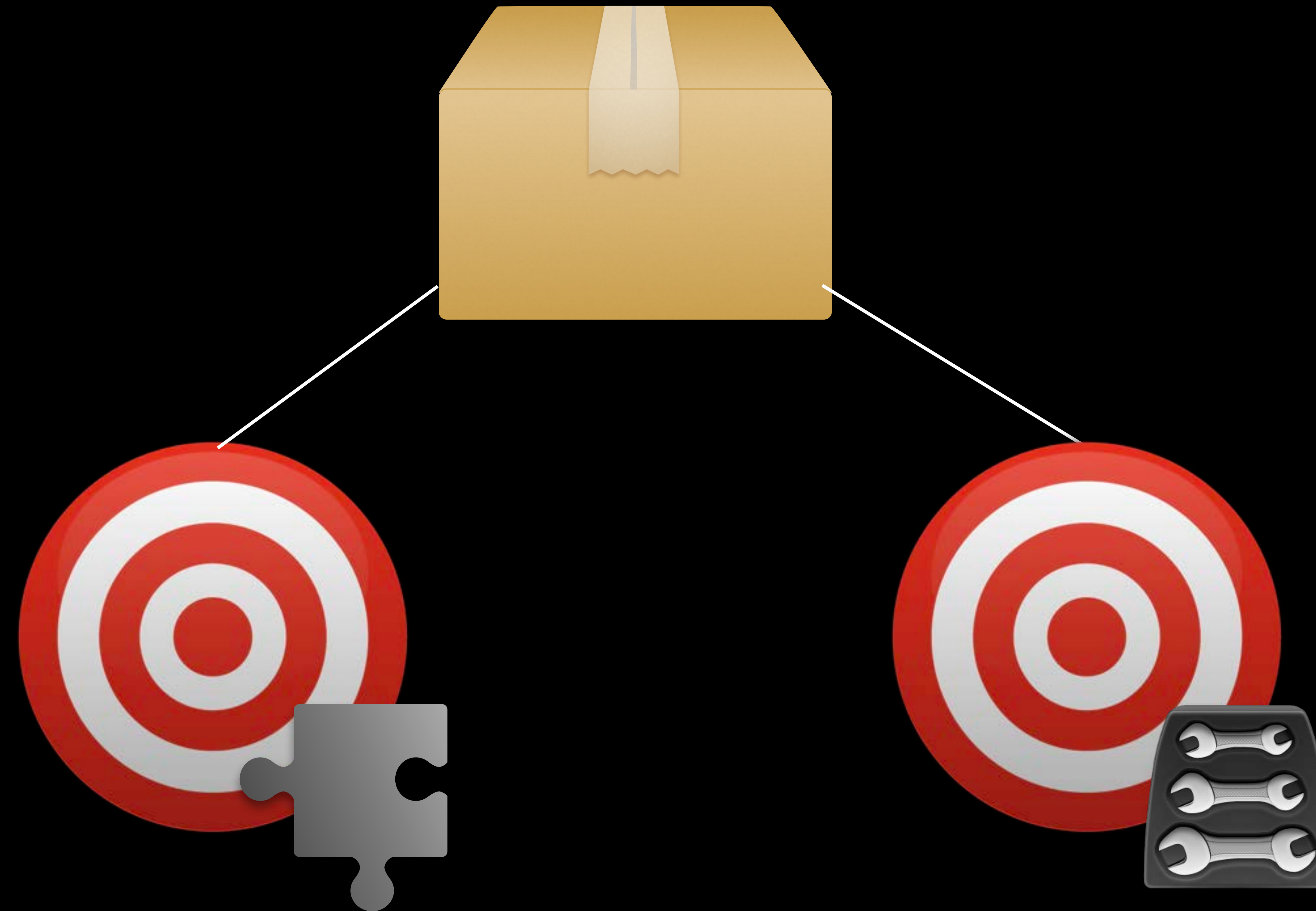
```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

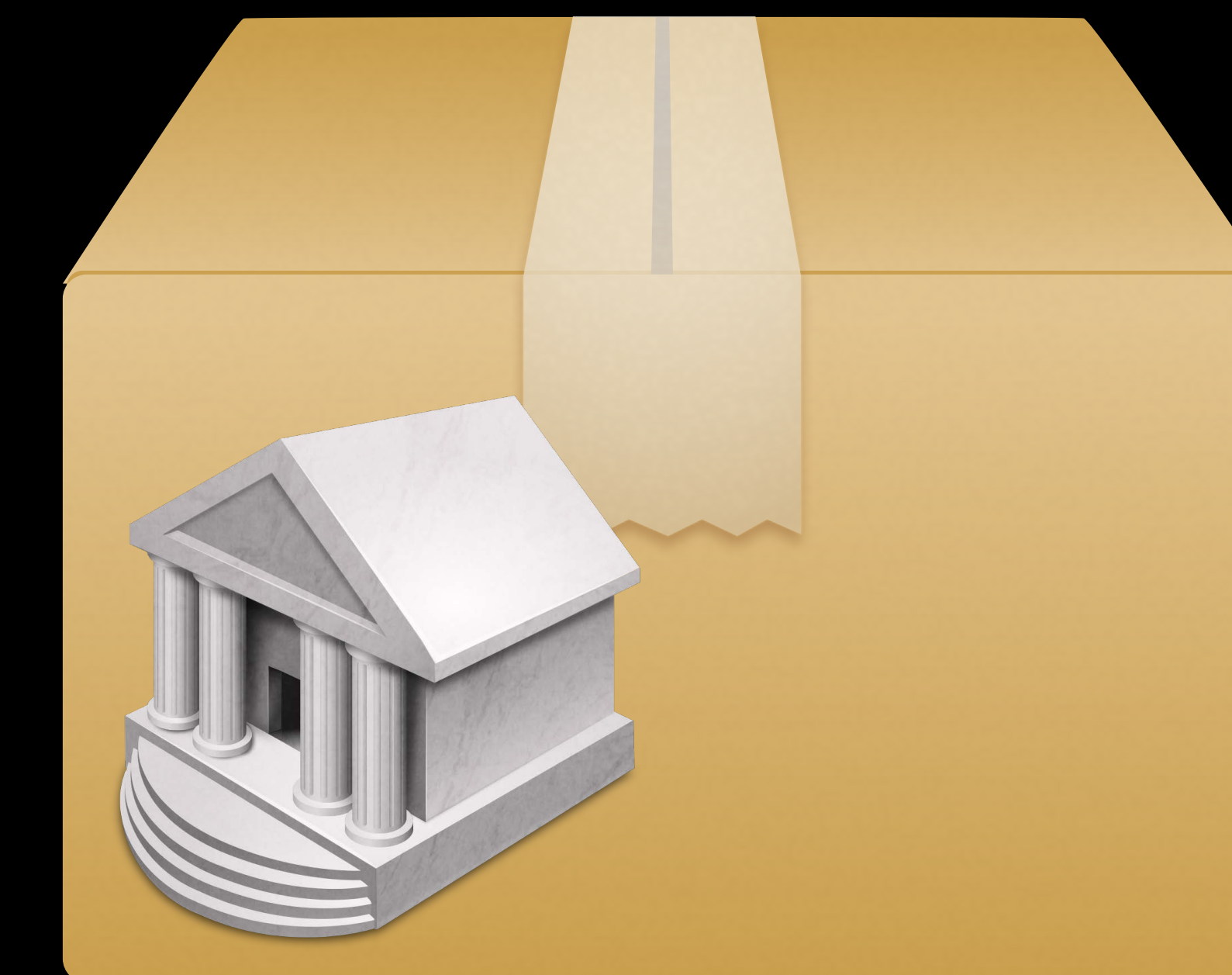
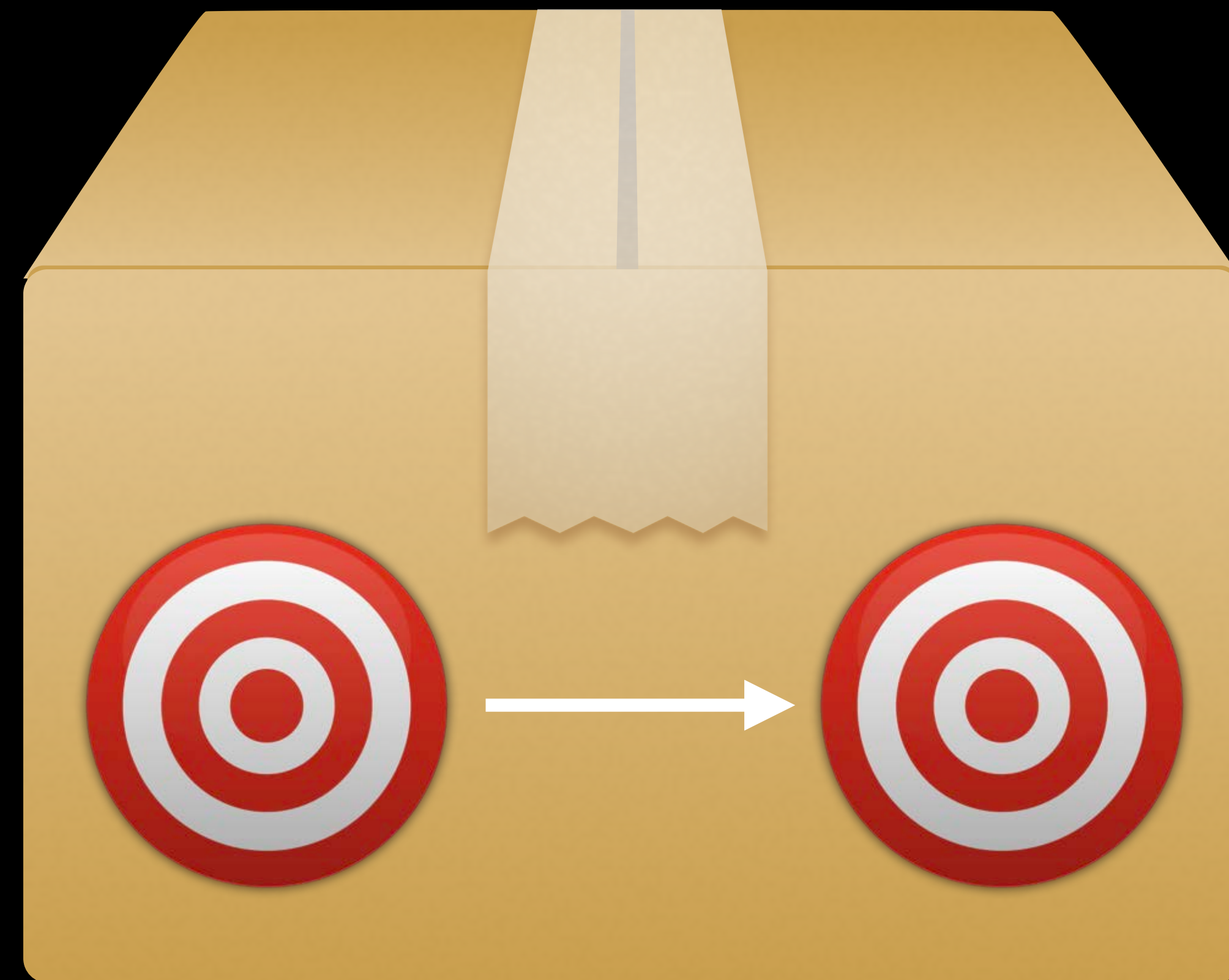
```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

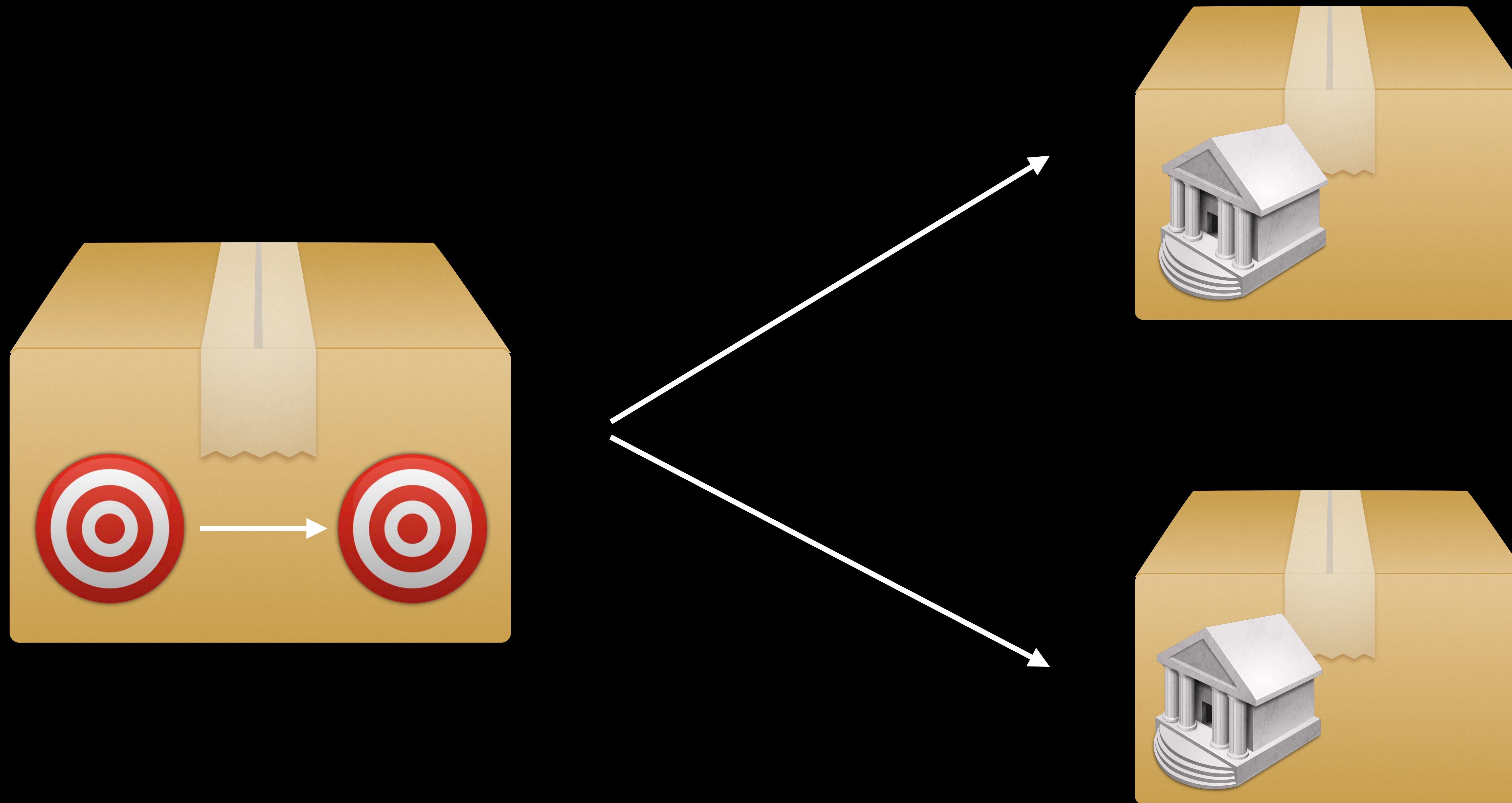
Targets



Targets Depend on Other Targets and Products



Targets Depend on Other Targets and Products



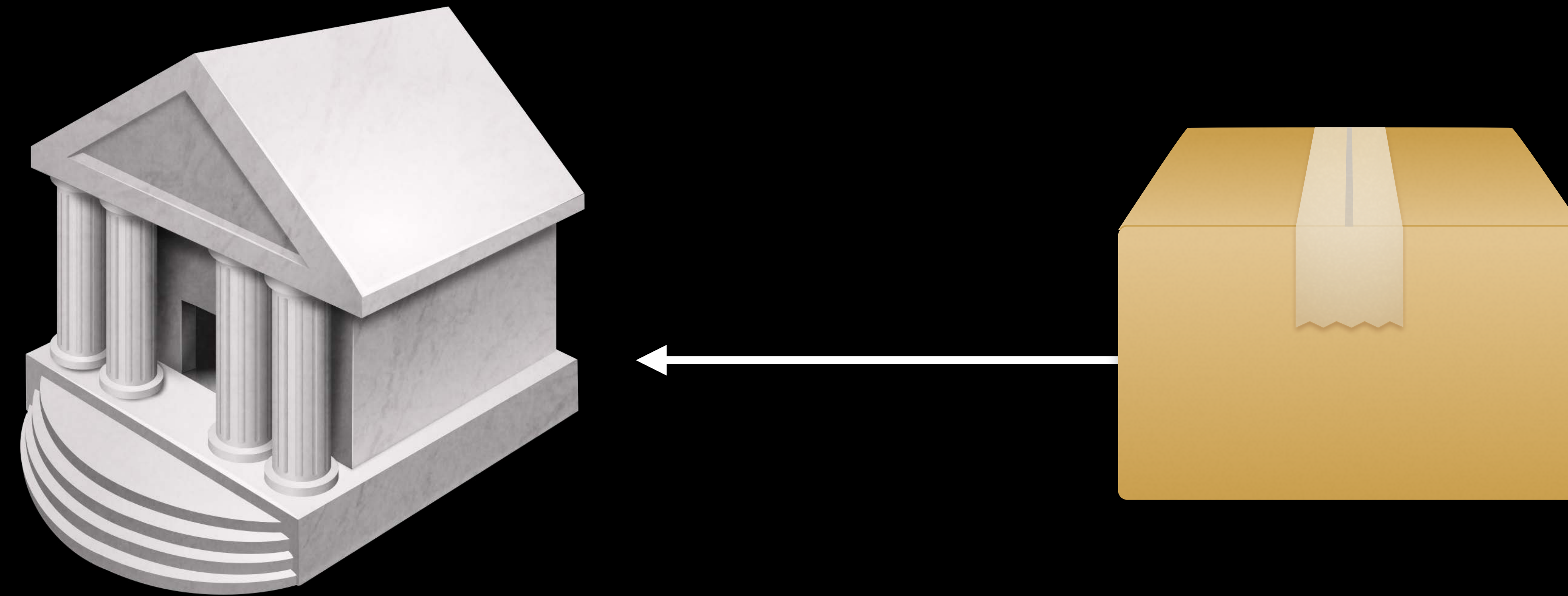
Products



Products



Packages Provide Products



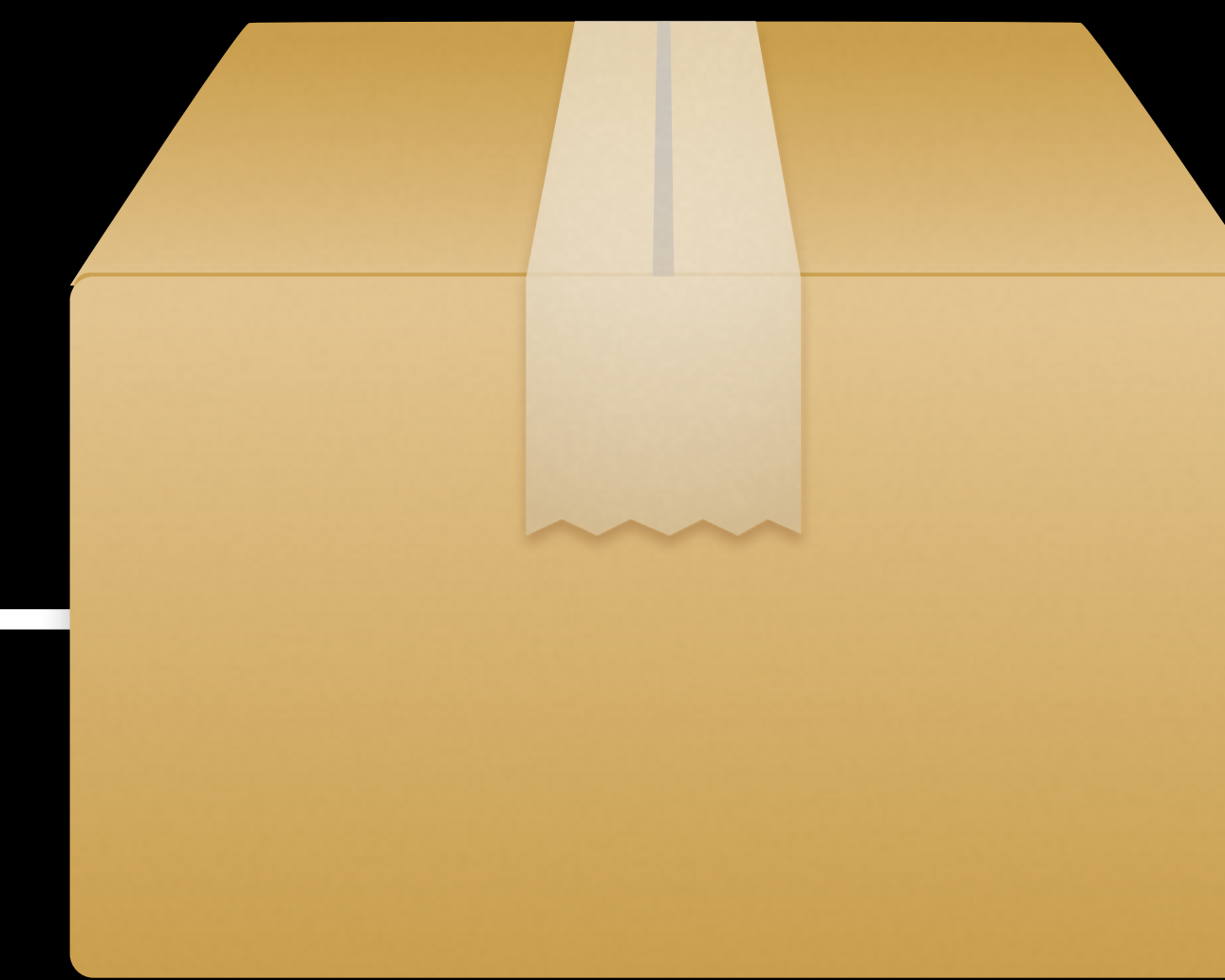
Packages Provide Products

Linkage:

Automatic

Static

Dynamic



```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```



```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```

```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "dealer",
    products: [
        .library(name: "libdealer", targets: ["libdealer"]),
        .executable(name: "dealer", targets: ["dealer"]),
    ],
    dependencies: [
        .package(url: "git@github.com:apple/example-package-deckofplayingcards", from: "3.0.0")
    ],
    targets: [
        .target(name: "libdealer", dependencies: ["DeckOfPlayingCards"]),
        .target(name: "dealer", dependencies: ["libdealer"]),
        .testTarget(name: "dealerTests", dependencies: ["libdealer", "dealer"]),
    ]
)
```


Demo

Adding dependencies

Why a package manager for Swift?

How to use It

The design of SwiftPM

Evolution ideas

Open source process

Following Swift's Philosophy

Safe

Fast

Expressive



Following Swift's Philosophy

Safe: isolated build environment

Fast

Expressive



Following Swift's Philosophy

Safe: isolated build environment

Fast: scalable to large dependency graphs

Expressive



Following Swift's Philosophy

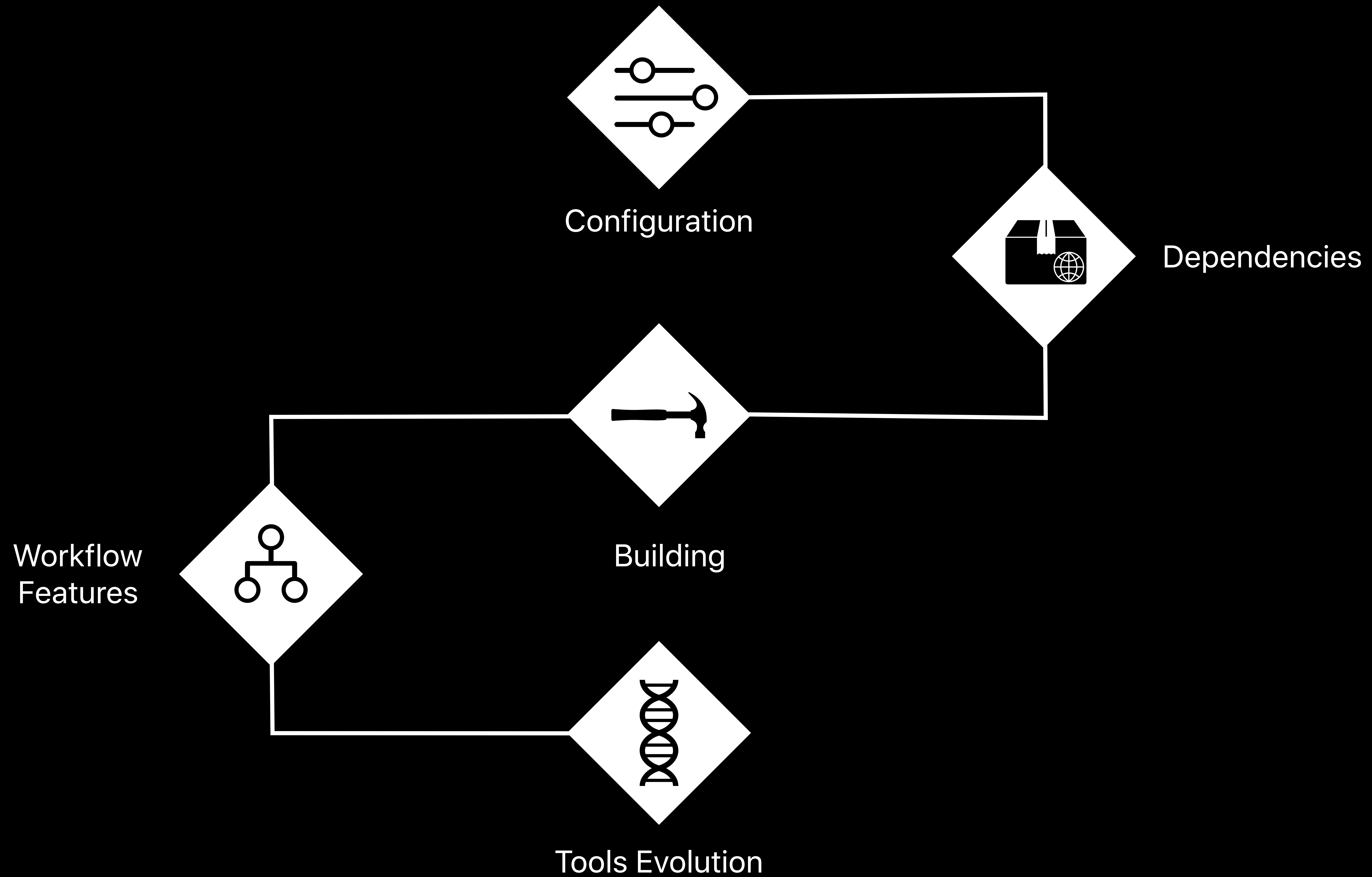
Safe: isolated build environment

Fast: scalable to large dependency graphs

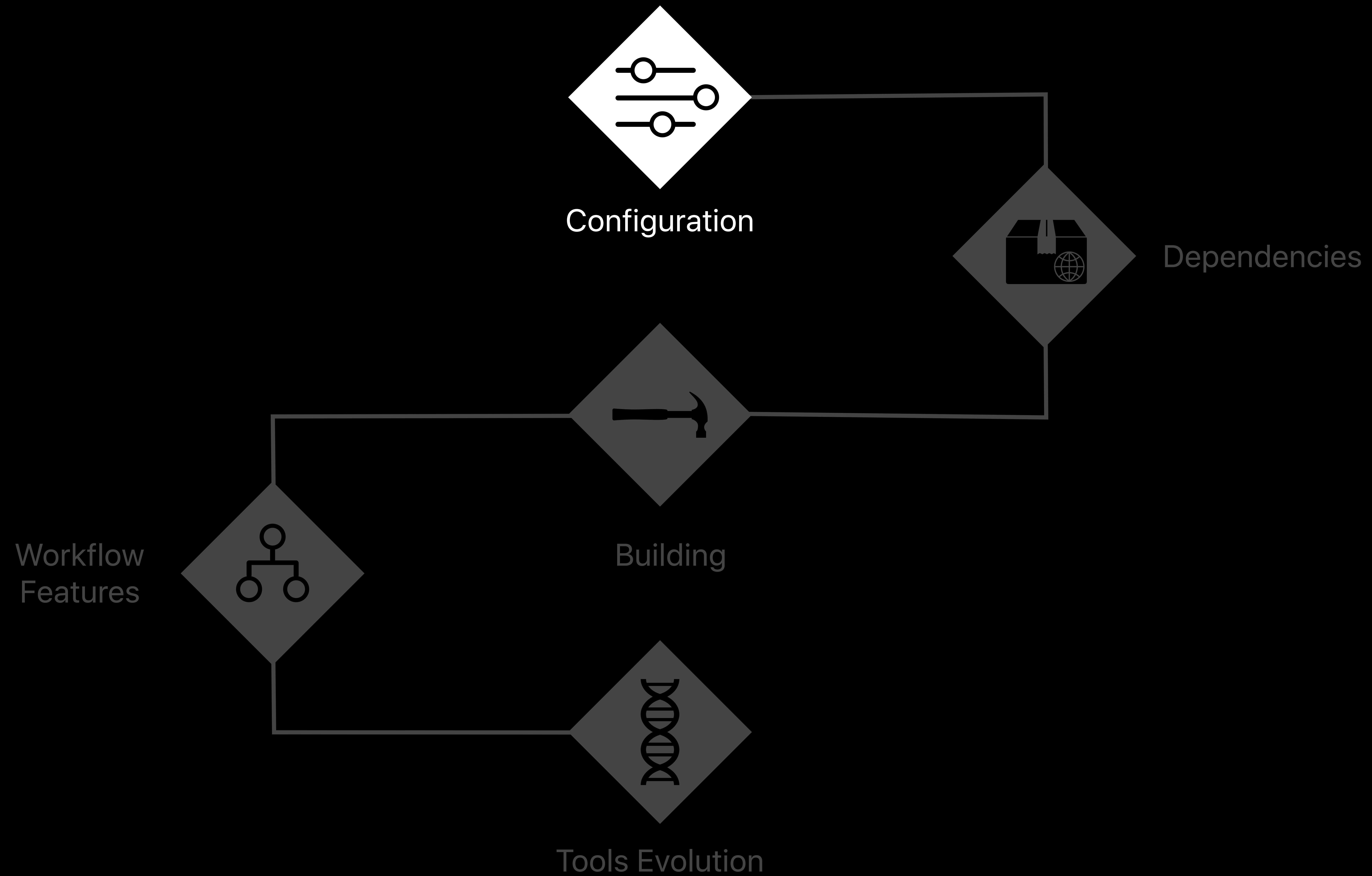
Expressive: Swift language manifest format



The Design of SwiftPM



The Design of SwiftPM

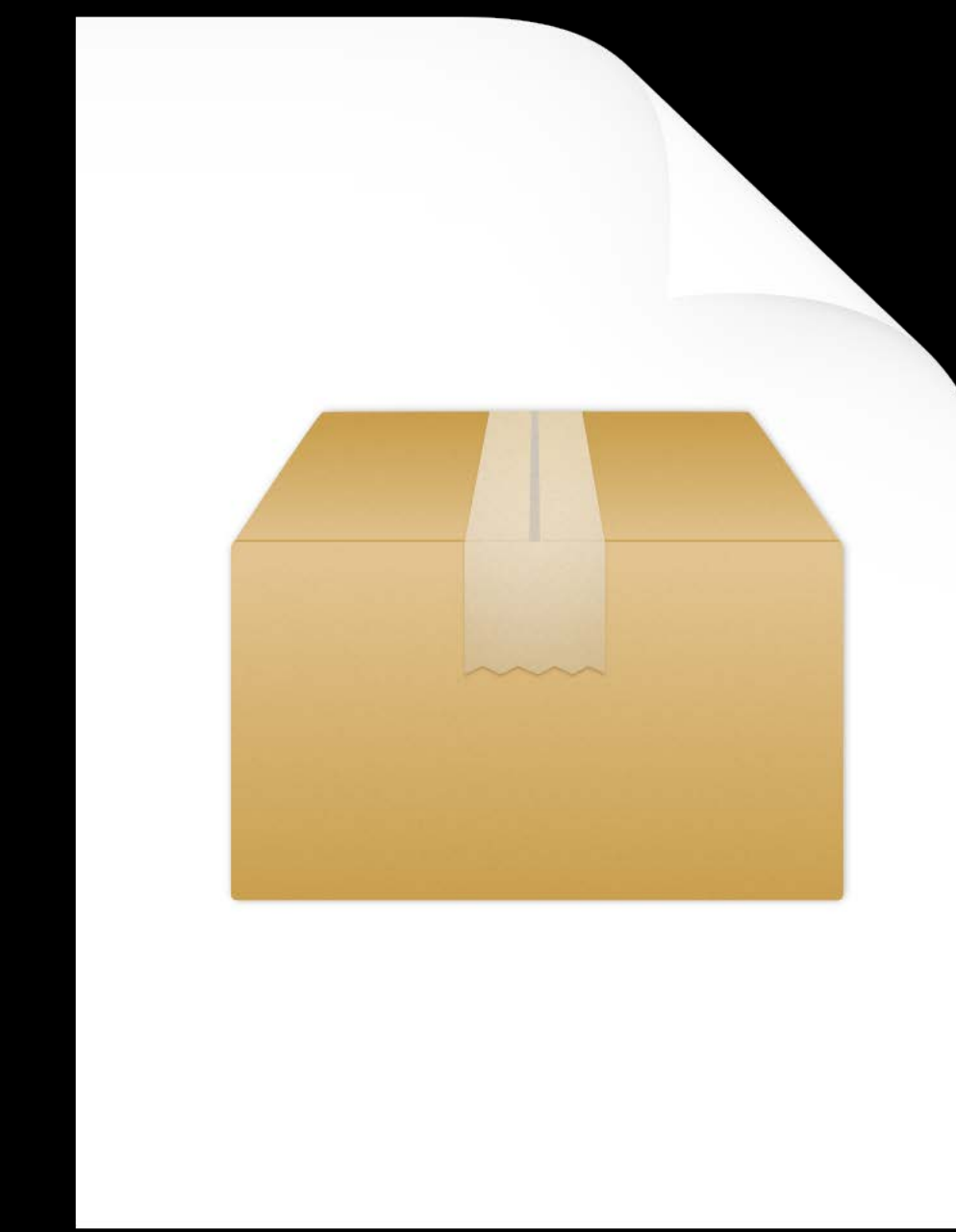


Swift Language Manifest Format

Easy to understand

Follows Swift API design guidelines


Supported by existing Swift tools



Prefer Declarative Syntax


```
// swift-tools-version:4.2
import PackageDescription

let name = generateName()
let package = Package(
    name: name,
    targets: [
        .target(
            name: name,
            dependencies: []),
        .testTarget(
            name: "\(name)Tests",
            dependencies: [.target(name: name)])
    ])
```



```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
    name: "A",
    targets: [
        .target(
            name: "A",
            dependencies: []),
        .testTarget(
            name: "ATests",
            dependencies: ["A"]),
    ]
)
```



Specifying Package Source Files

 Sources

 dealer

 main.swift

 libdealer

 libdealer.swift

Convention versus Configuration

 Sources

 dealer

 main.swift

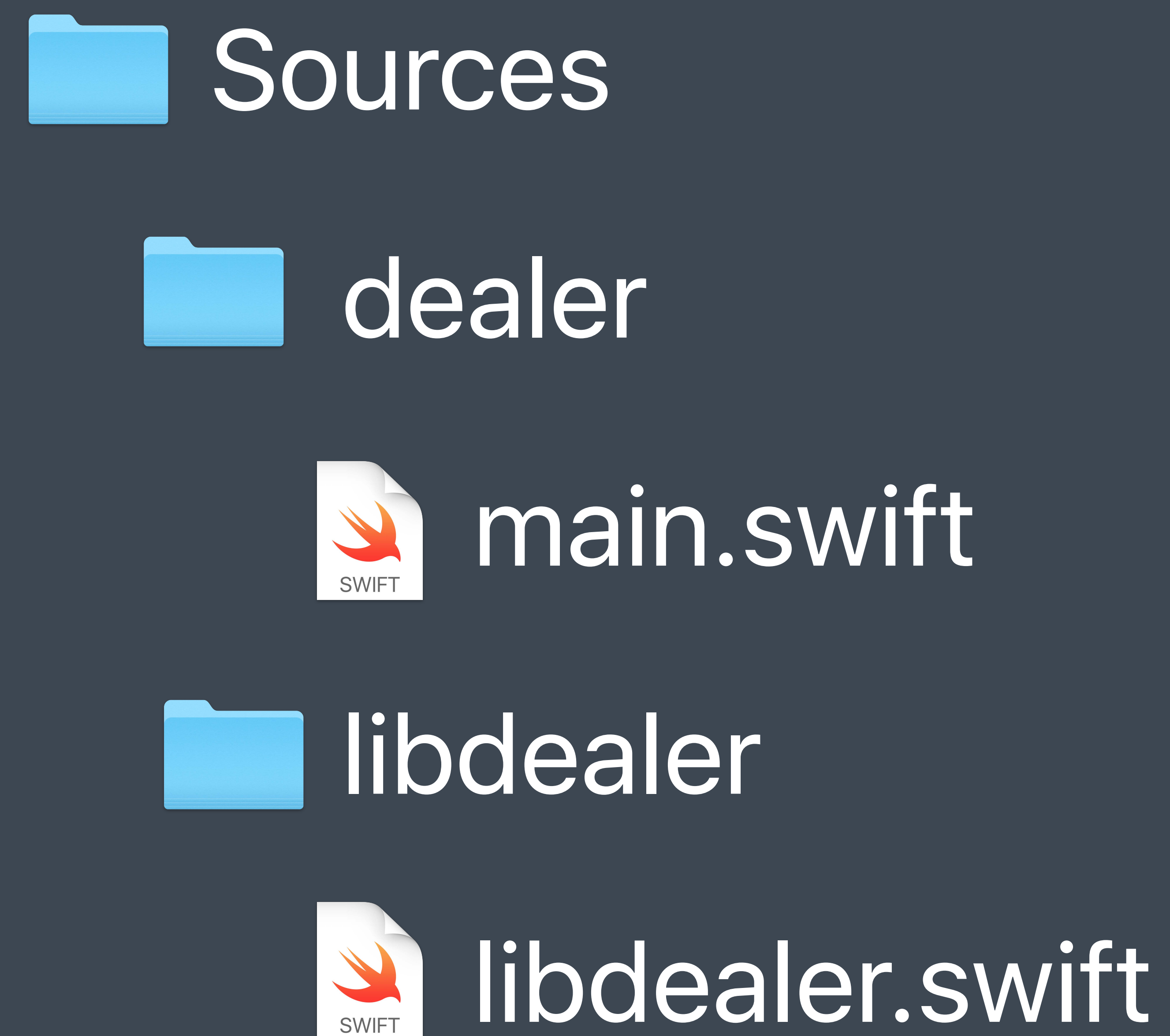
 libdealer

 libdealer.swift

```
...
products: [
    .library(name: "libdealer",
              targets: ["libdealer"]),
...
targets: [
    .target(name: "libdealer",
             dependencies: ["DeckOfPlayingCards"]),
...

```

Convention versus Configuration



```
...
products: [
    .library(name: "libdealer",
             targets: ["libdealer"]),
    ...
targets: [
    .target(name: "libdealer",
            dependencies: ["DeckOfPlayingCards"]),
    ...

```

Convention versus Configuration

 Sources

 dealer

 main.swift

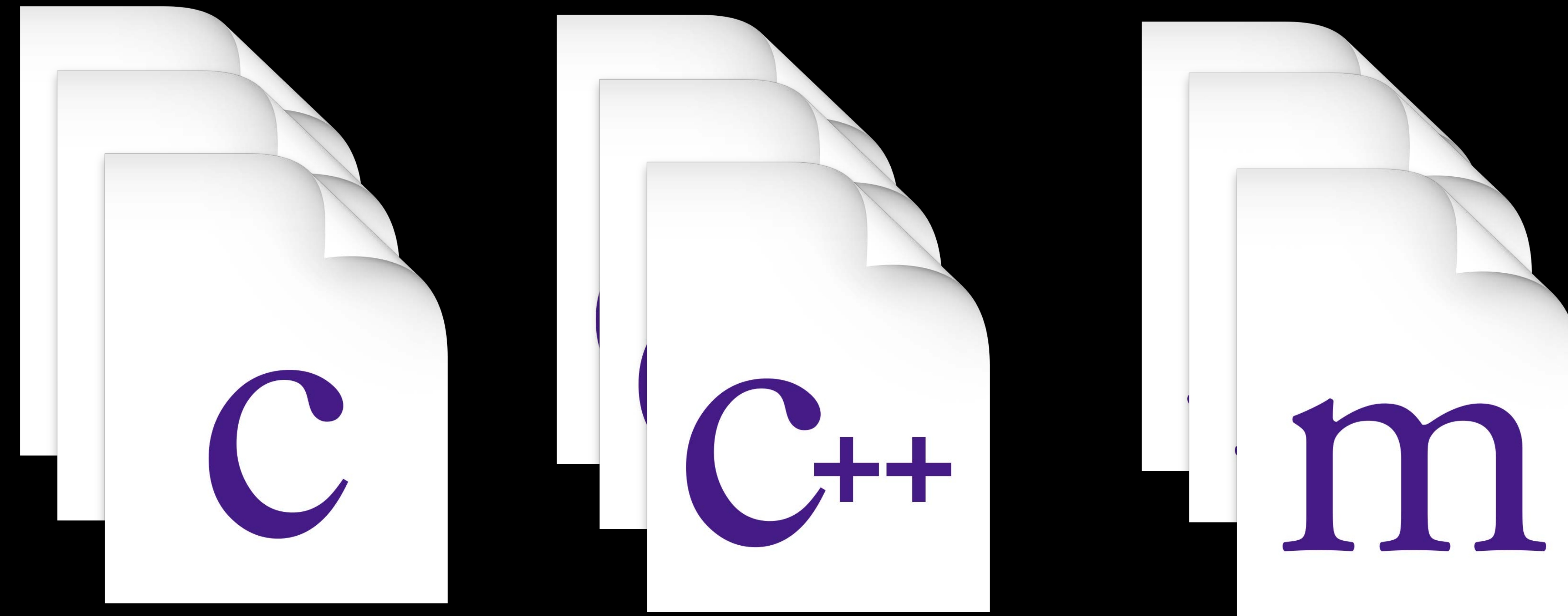
 libdealer

 libdealer.swift

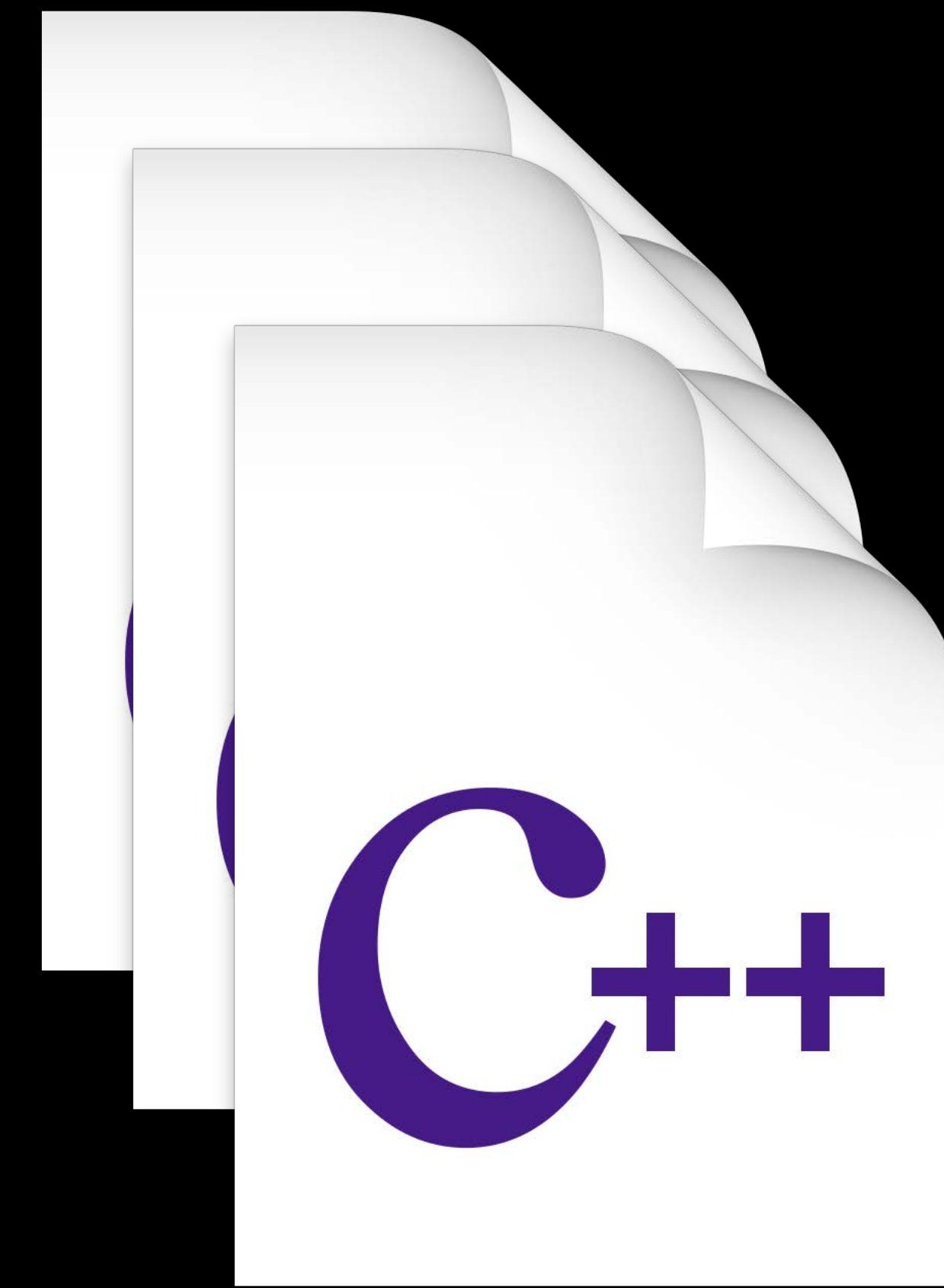
```
...
products: [
    .library(name: "libdealer",
              targets: ["libdealer"]),
...
targets: [
    .target(name: "libdealer",
            dependencies: ["DeckOfPlayingCards"]),
...

```

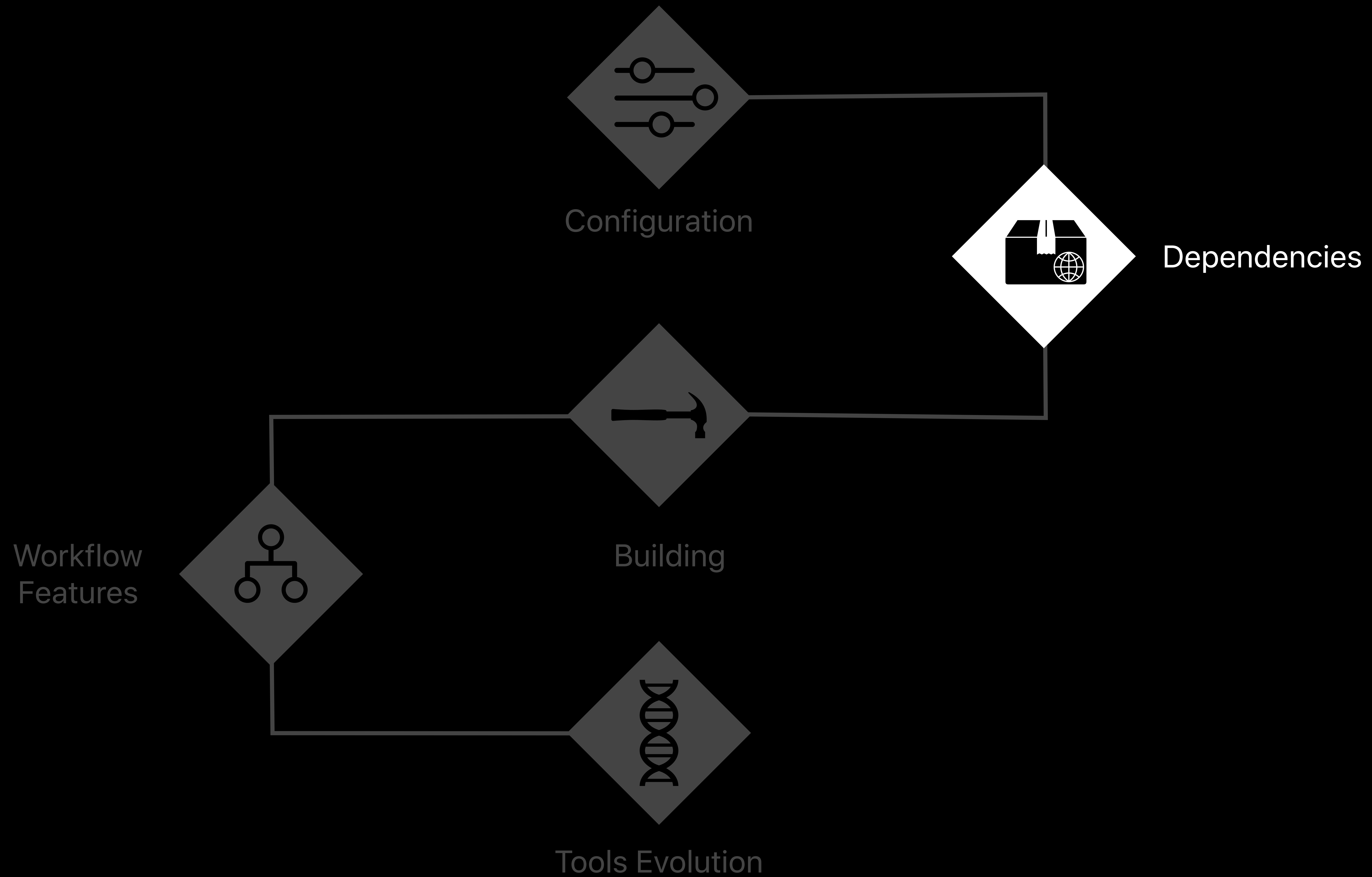
Support for Building Other Languages



Support for Building Other Languages



The Design of SwiftPM



Semantic Versioning

semver.org

1.2.4

Major Version

Breaking changes

1.2.4

Minor Version

Compatible additions

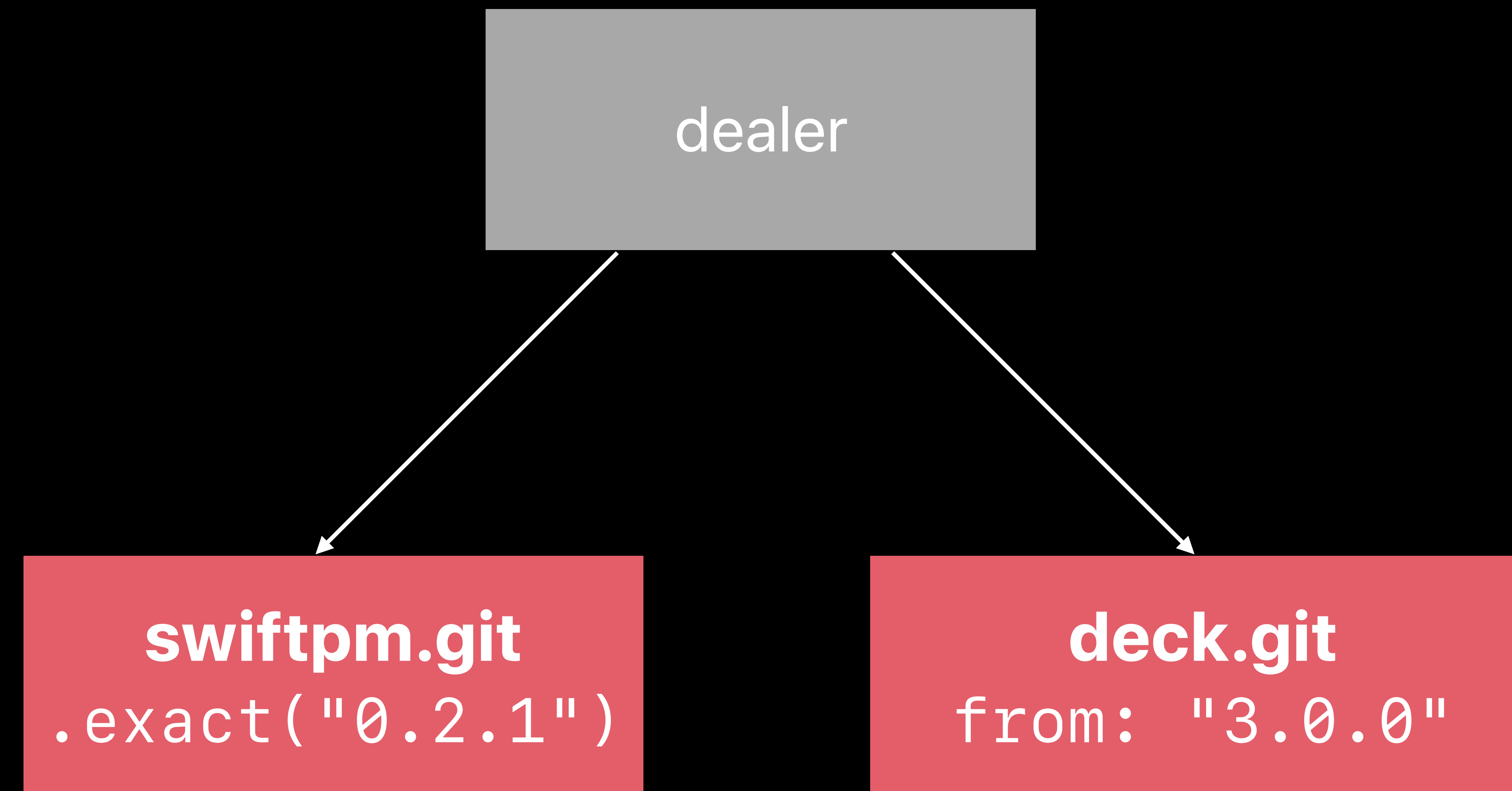
1.2.4

Patch Version

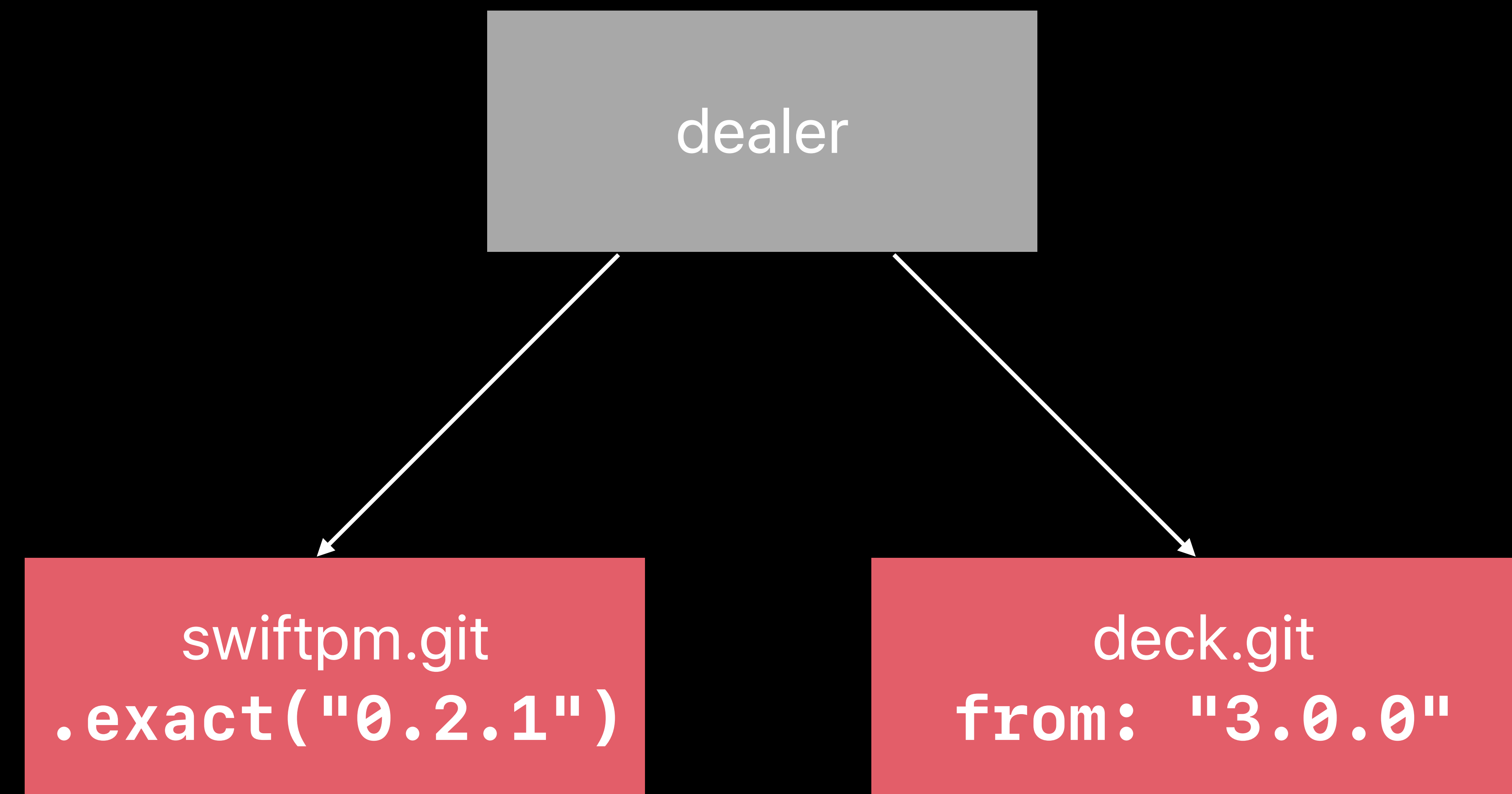
Bug fixes

Dependency Resolution

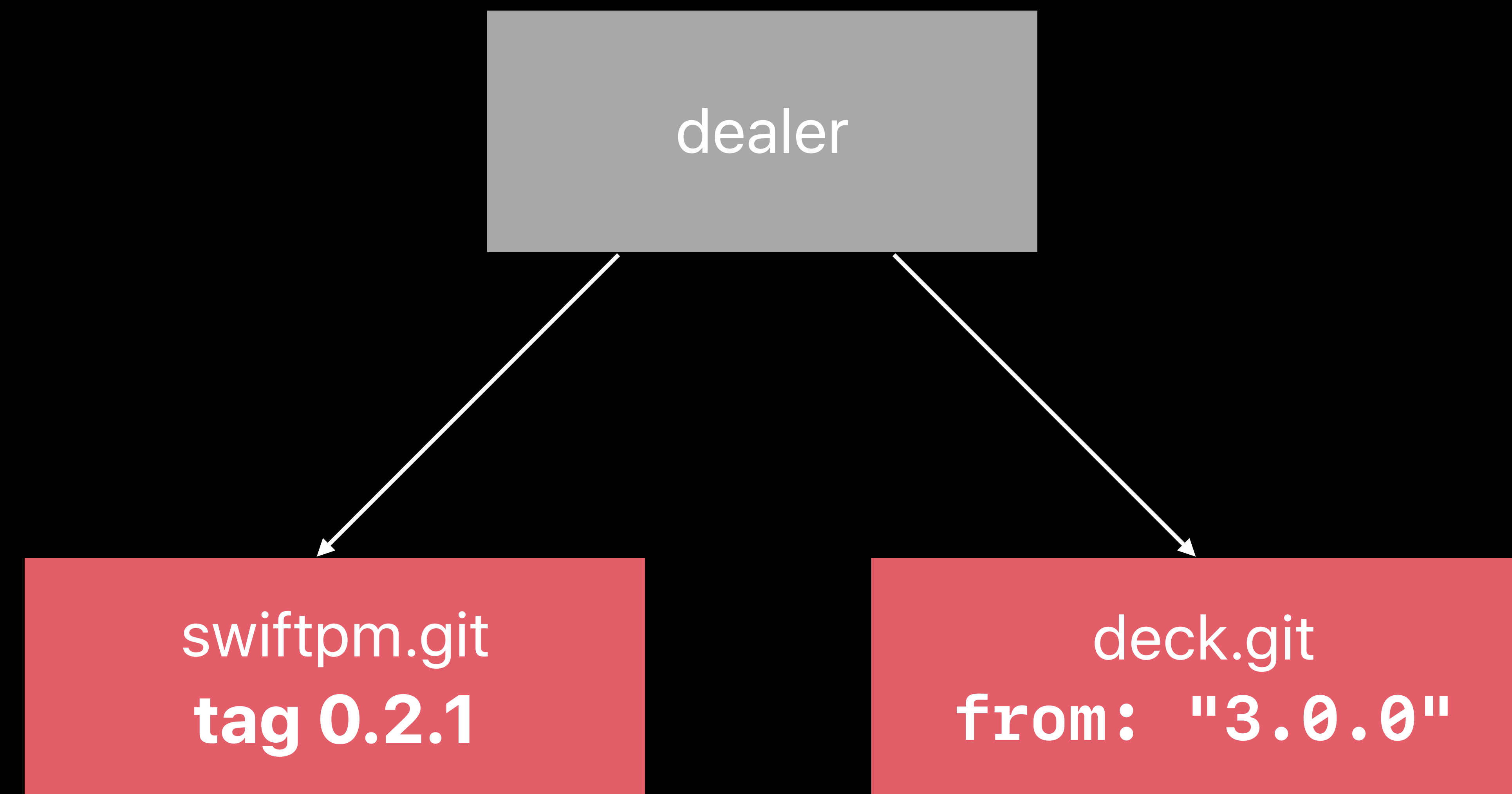
Direct Dependencies



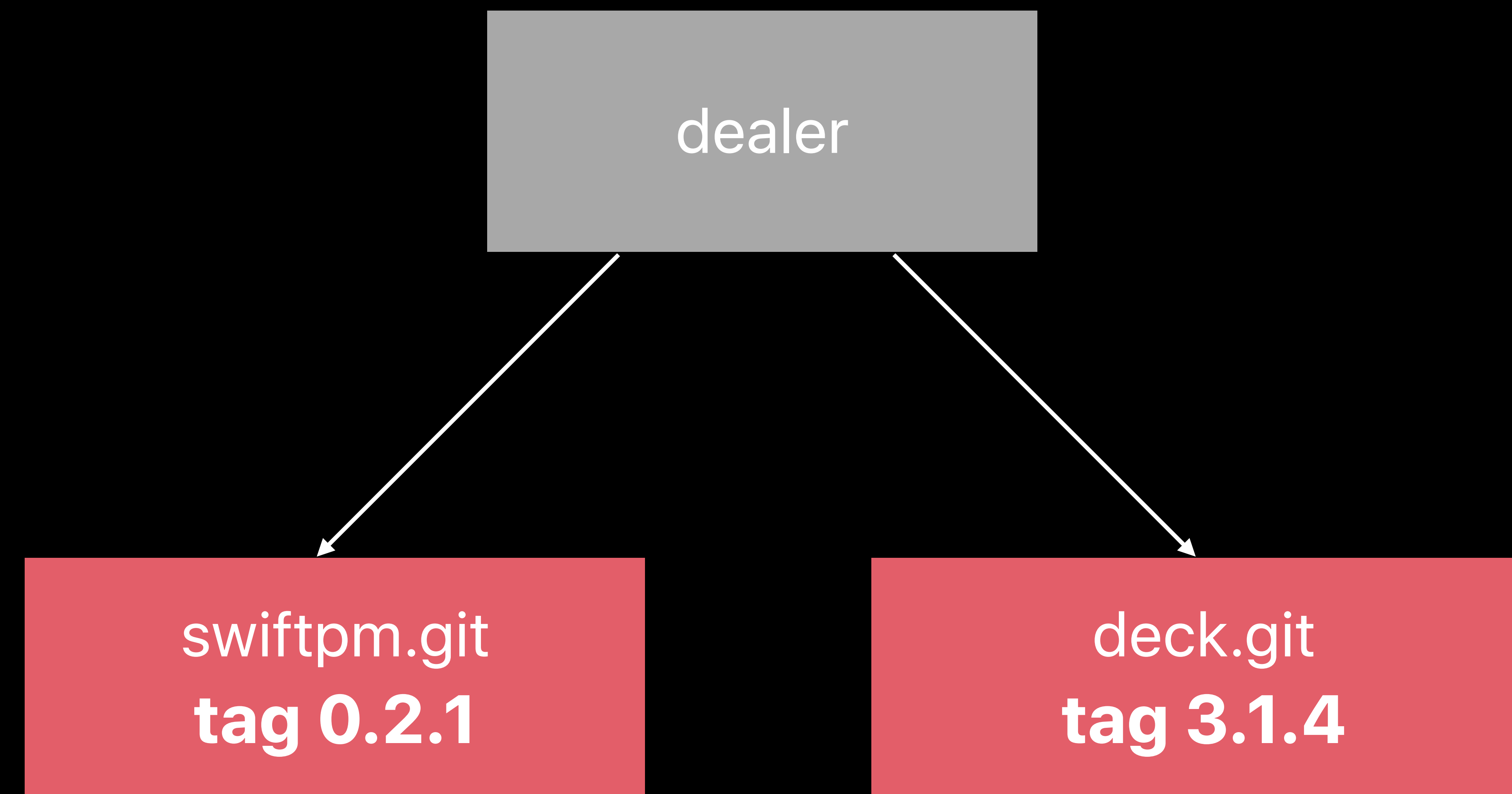
Resolving Versions



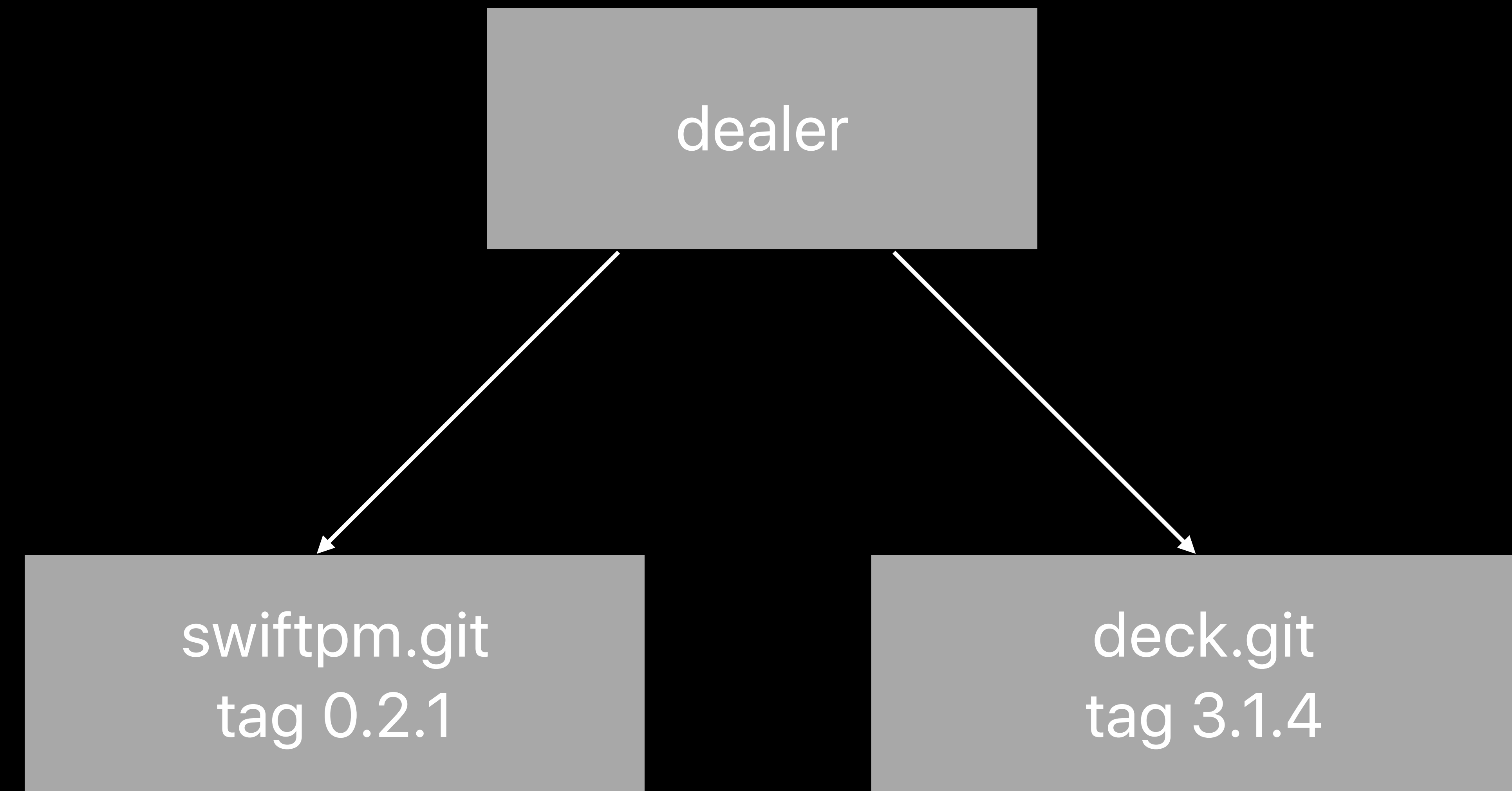
Resolving Versions



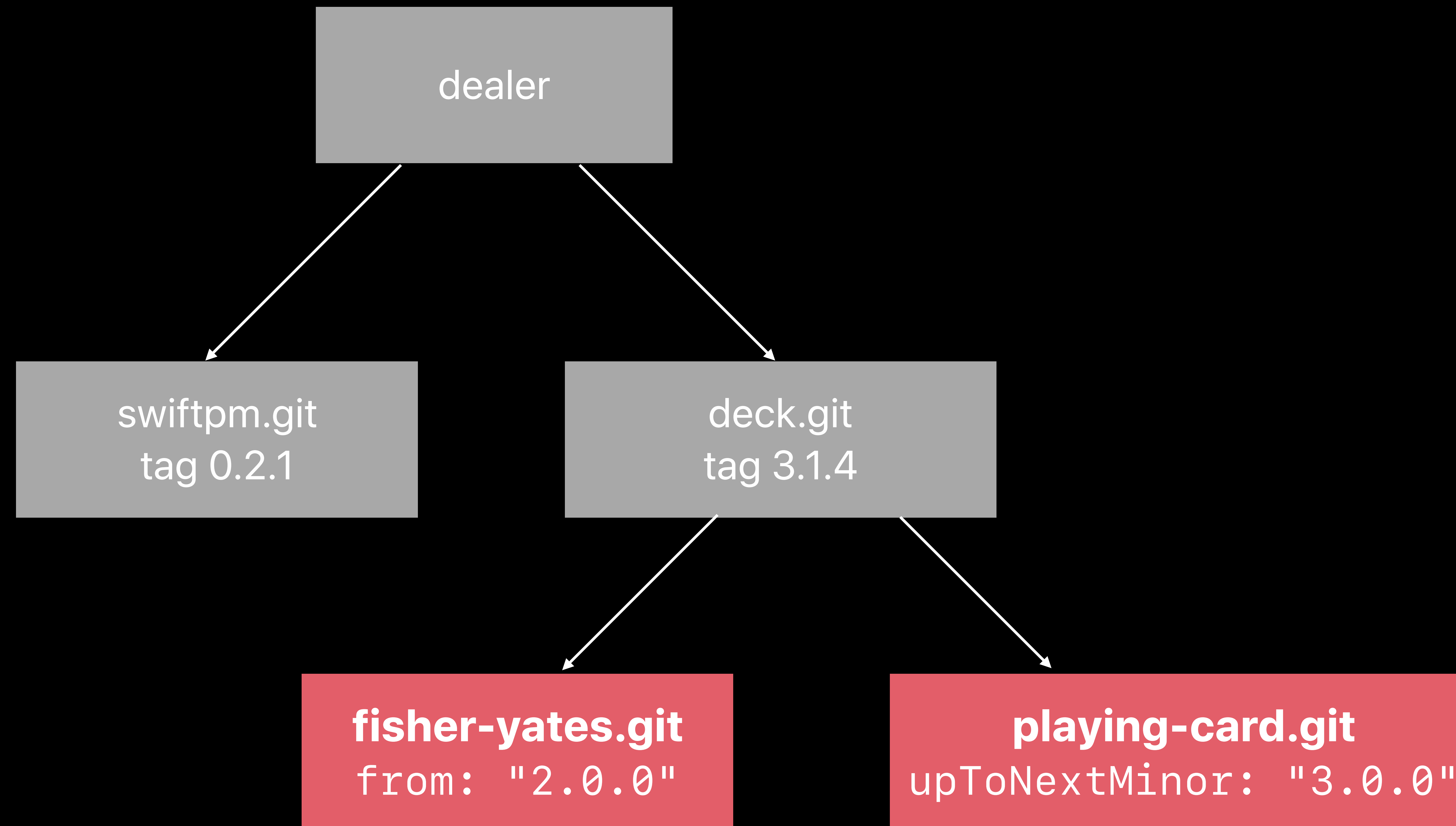
Resolving Versions



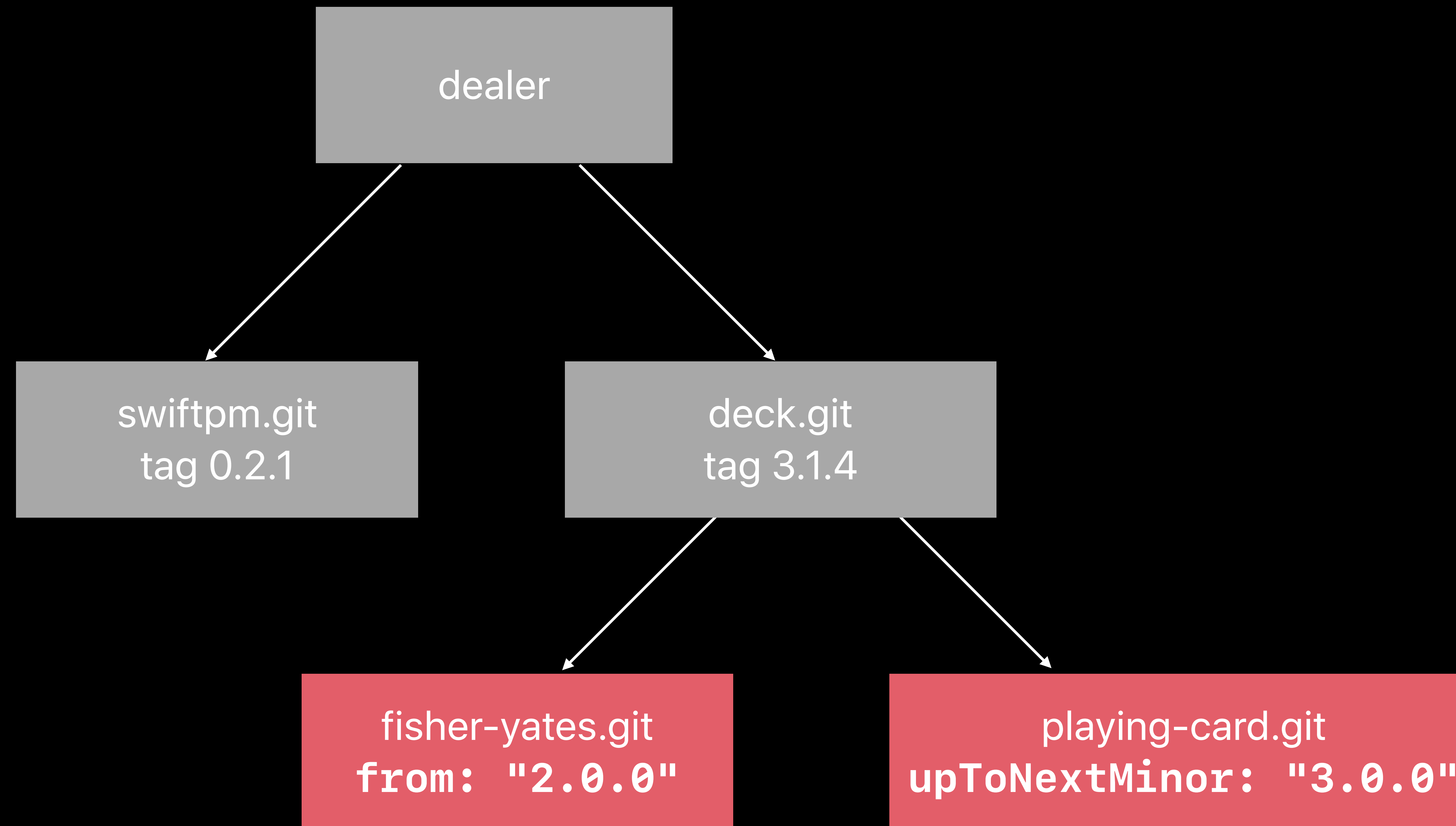
Transitive Dependencies



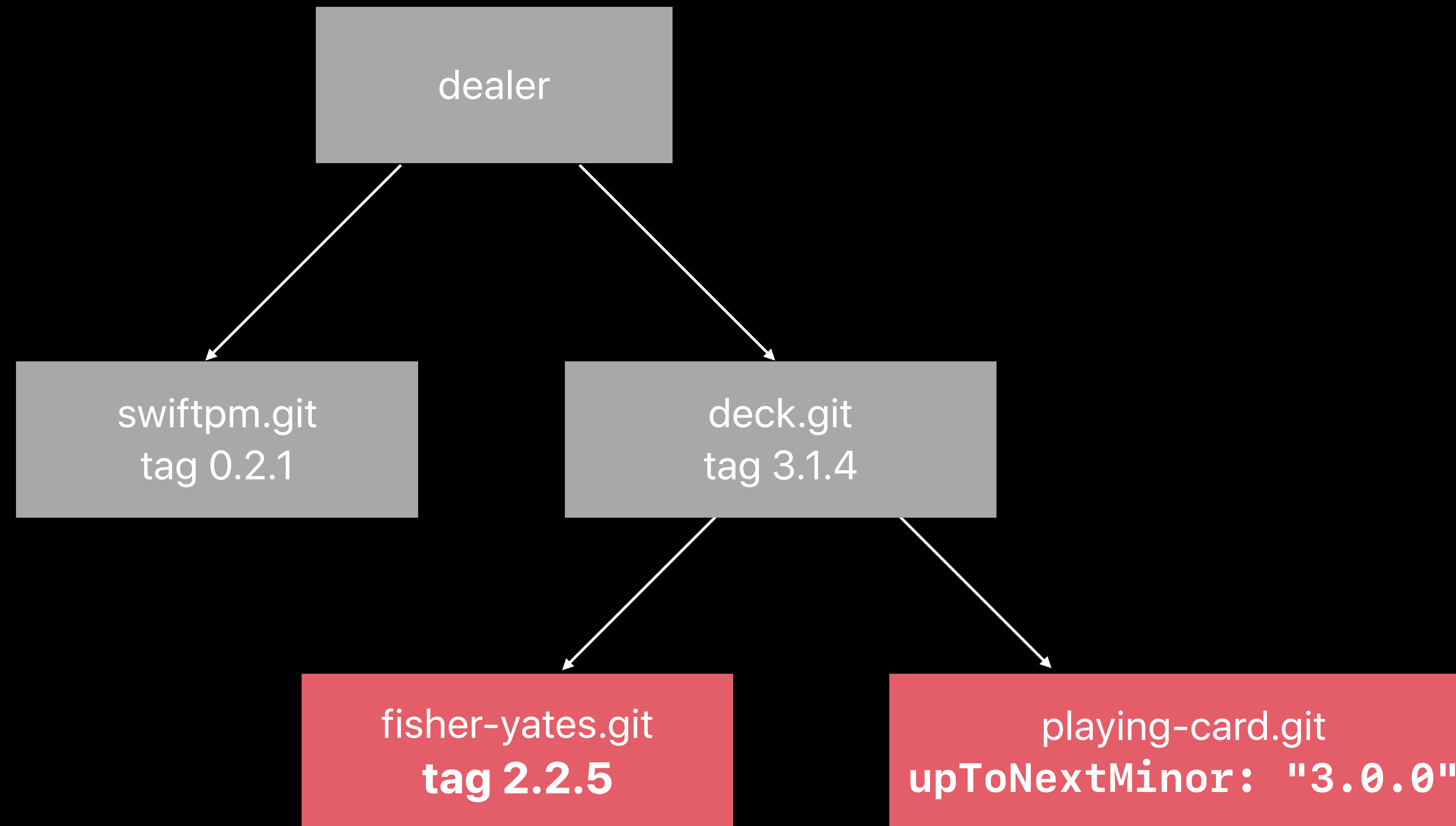
Transitive Dependencies



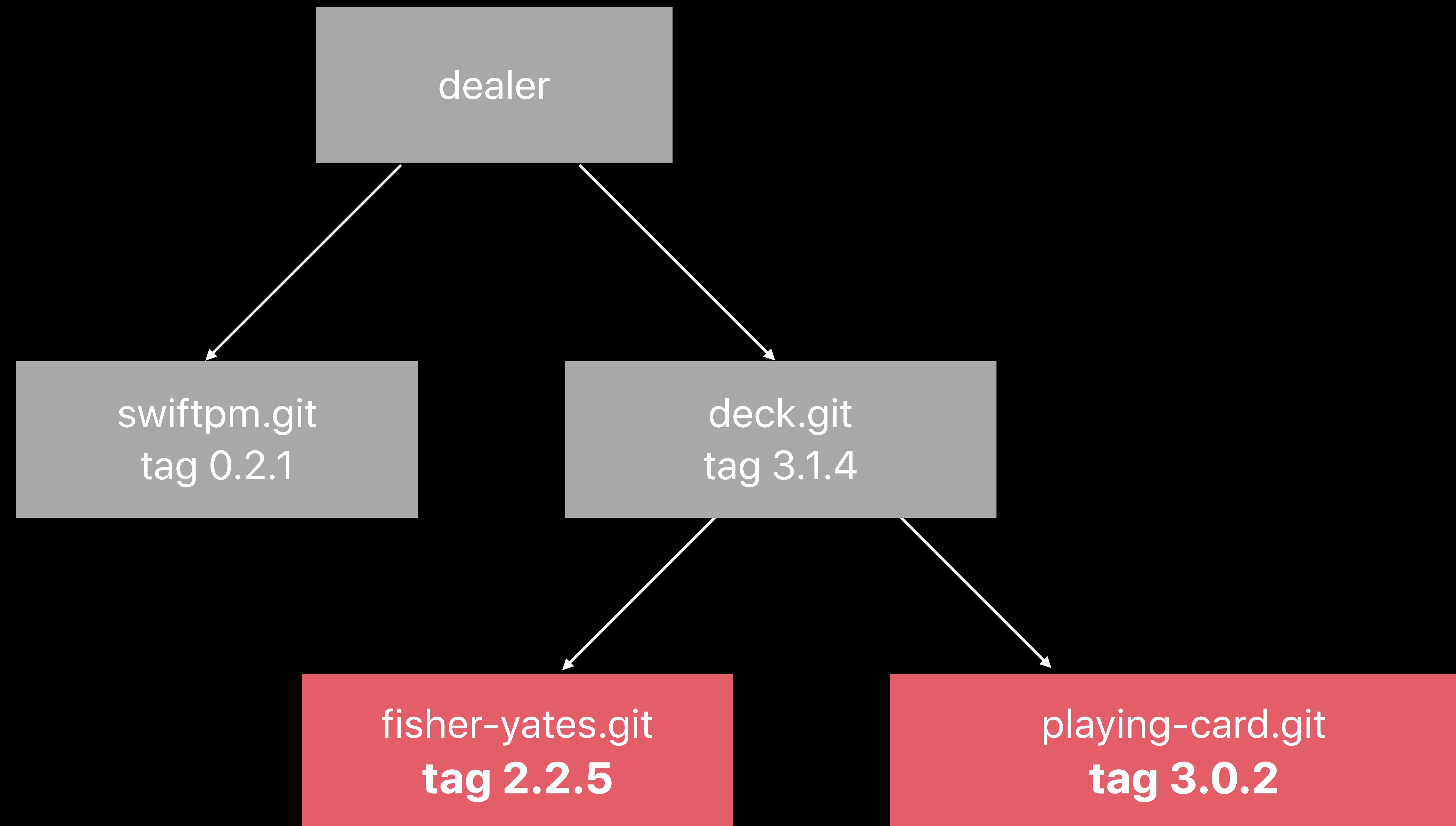
Transitive Dependencies



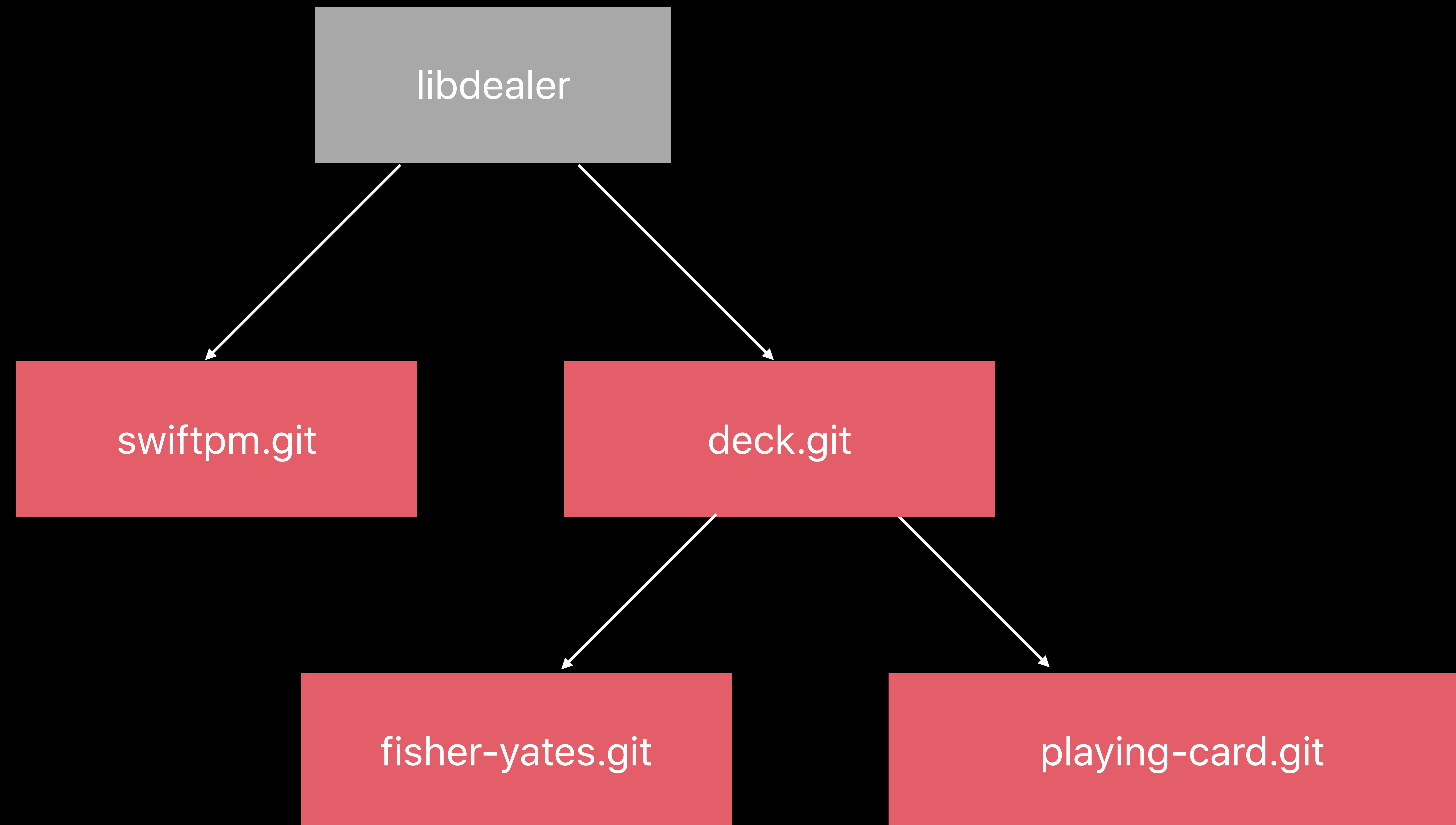
Transitive Dependencies



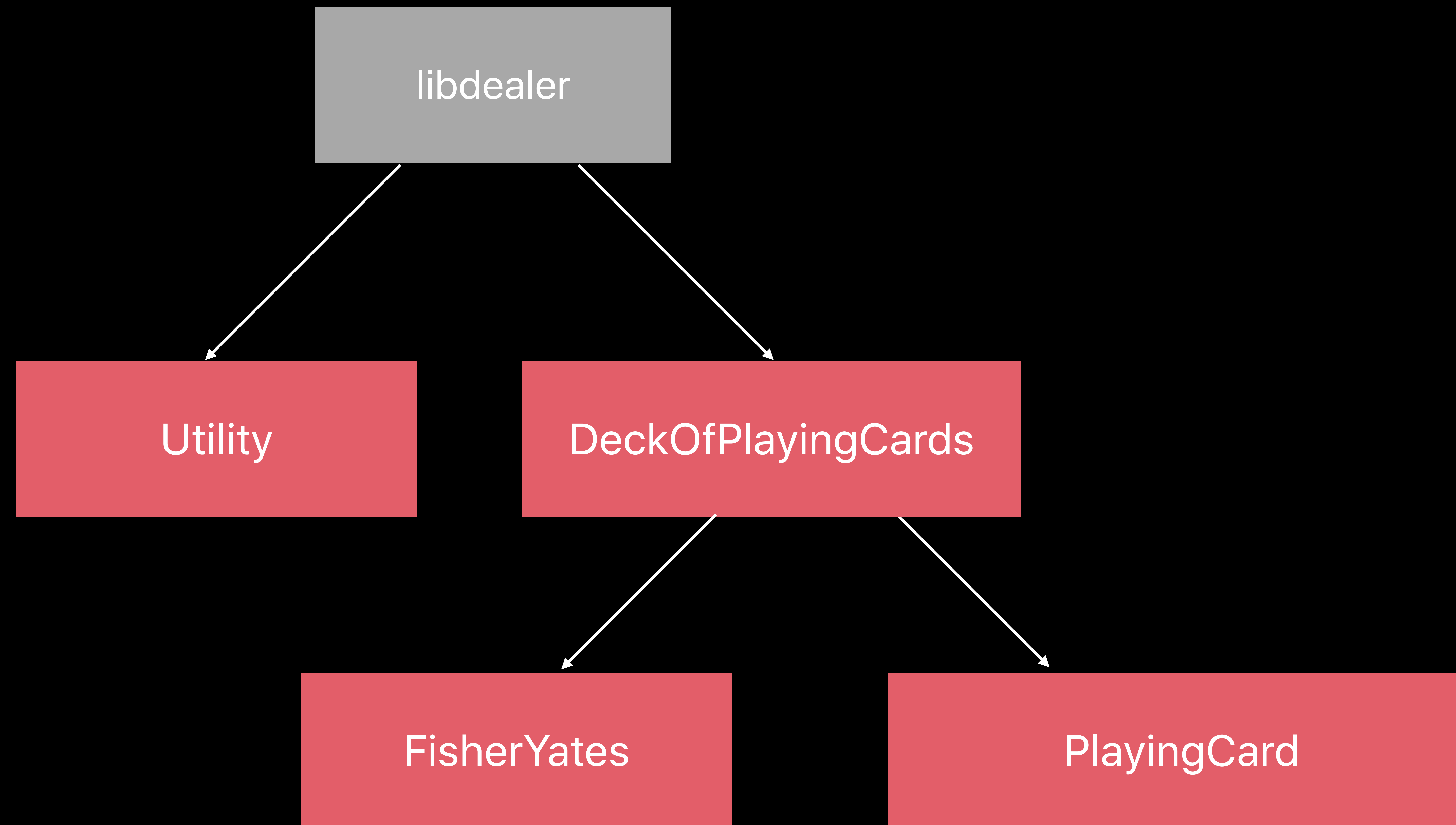
Transitive Dependencies



Resolving Products



Resolving Products



Package.resolved Resolved Versions File

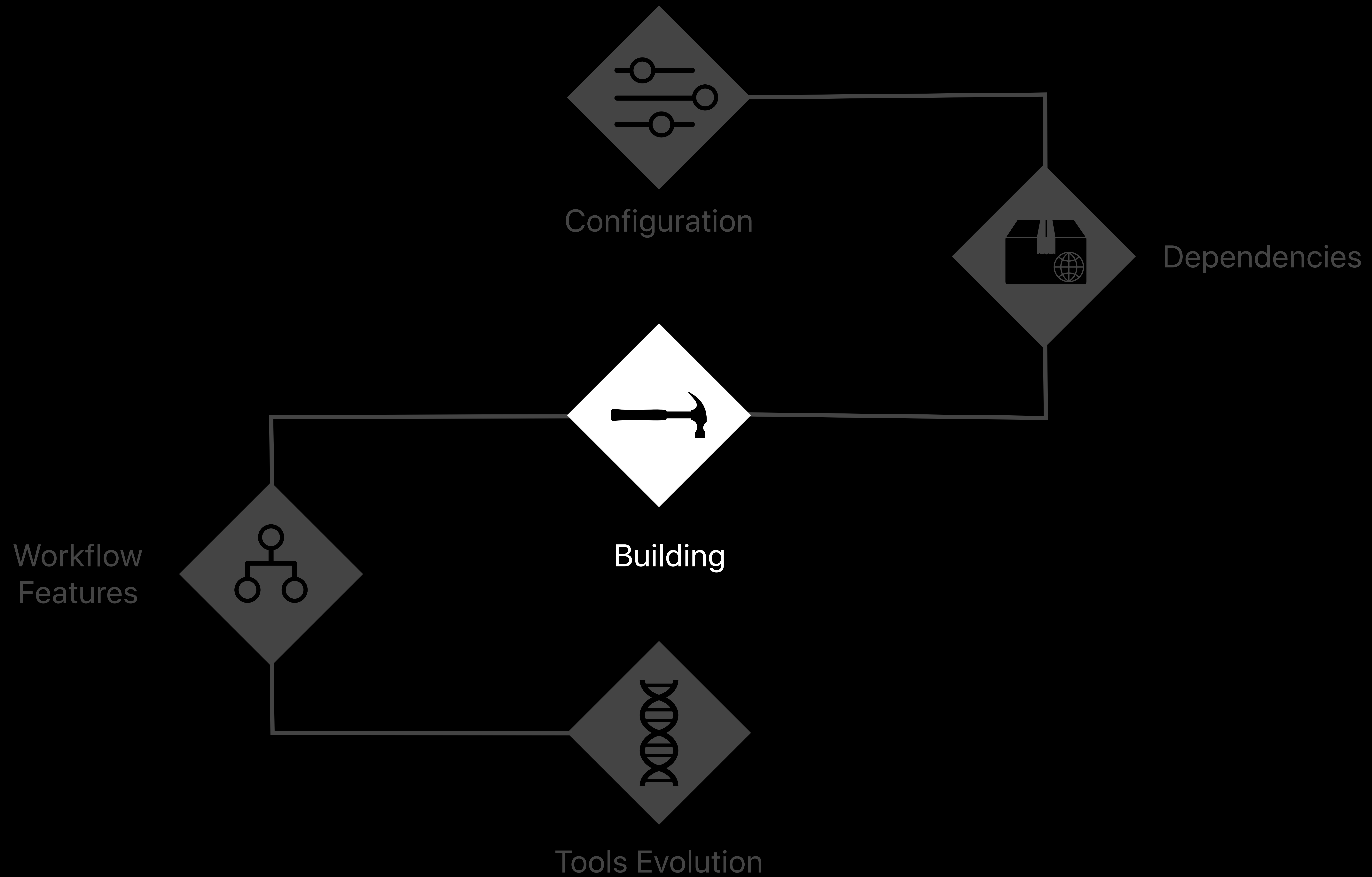
Records resolved package versions

Can be shared for dependable build results

Easy to update with new version resolution

Only used from the top-level package

The Design of SwiftPM



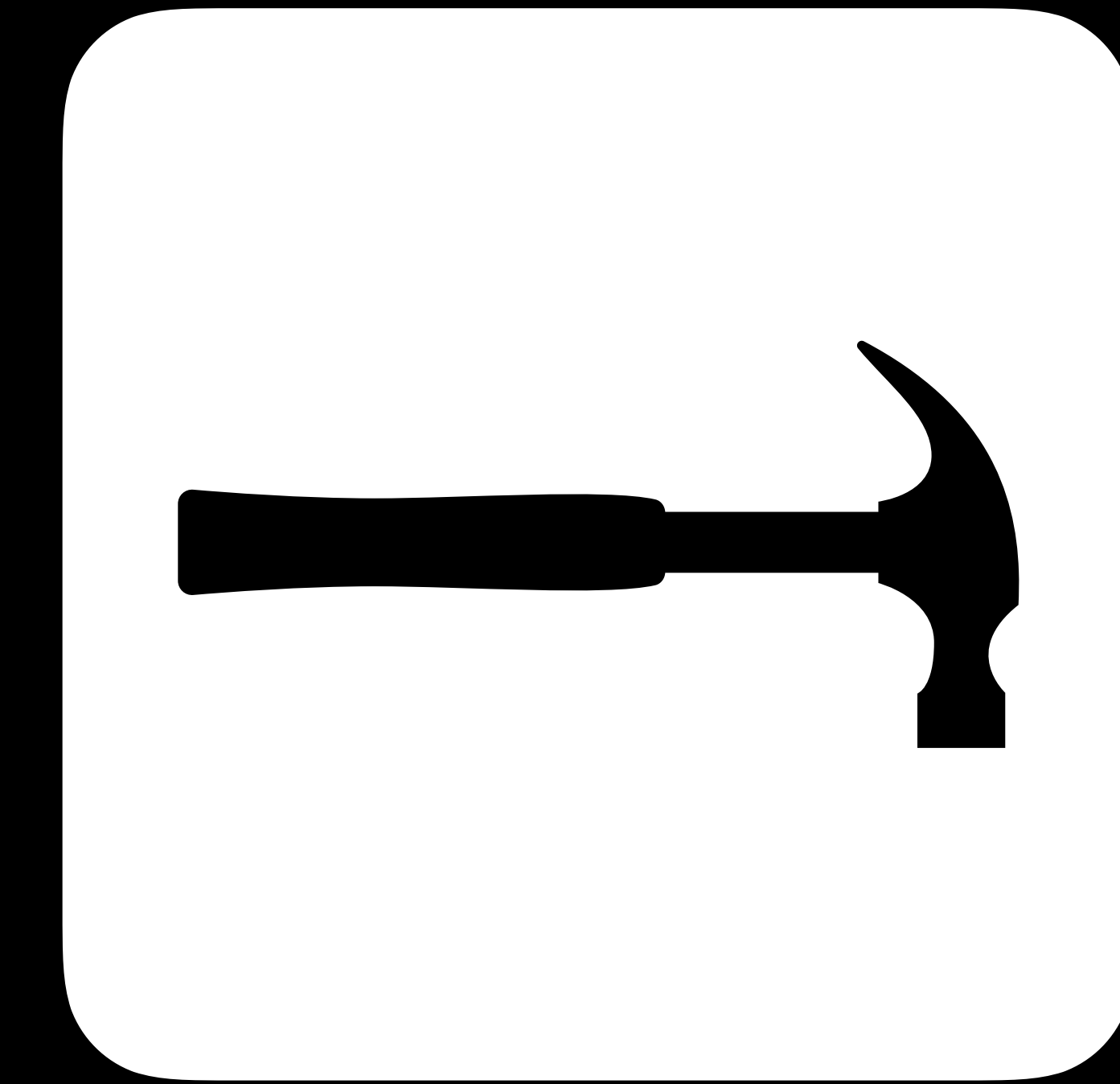
llbuild

SwiftPM's build execution engine

Provides fast and correct incremental builds

Also used by Xcode's new build system

Part of the Swift open source project

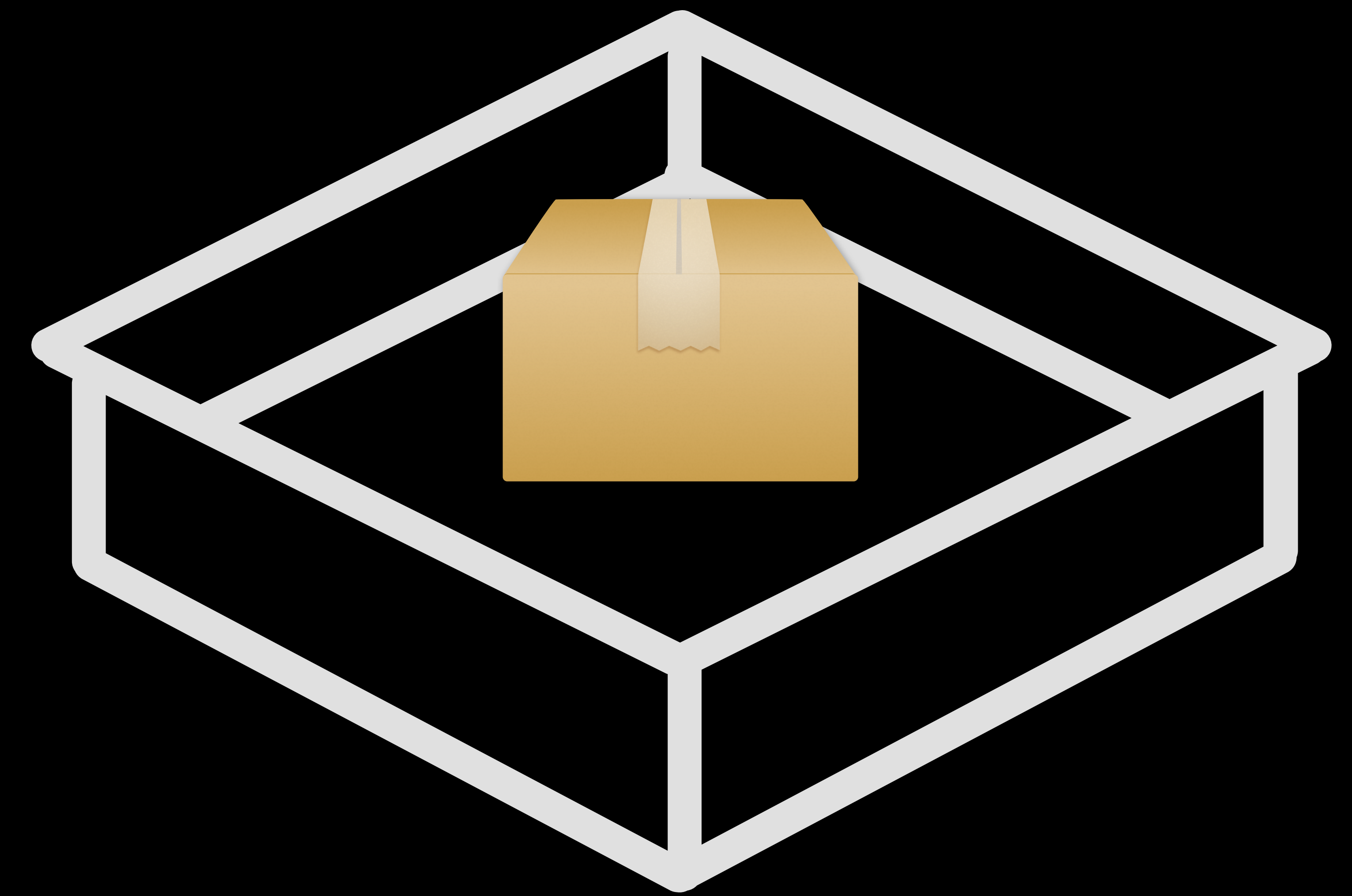


Build Environment Isolation

SwiftPM builds packages in isolation

Builds are sandboxed

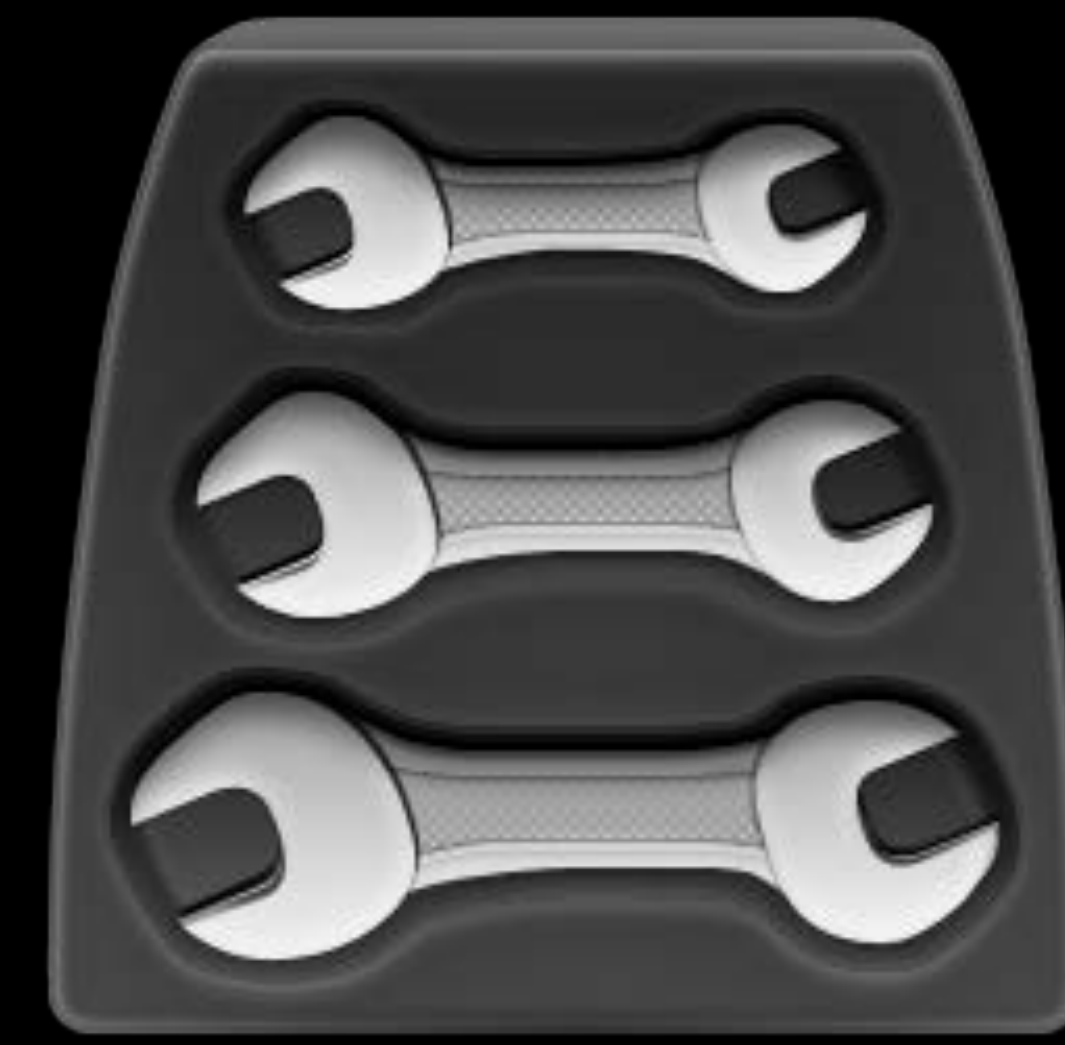
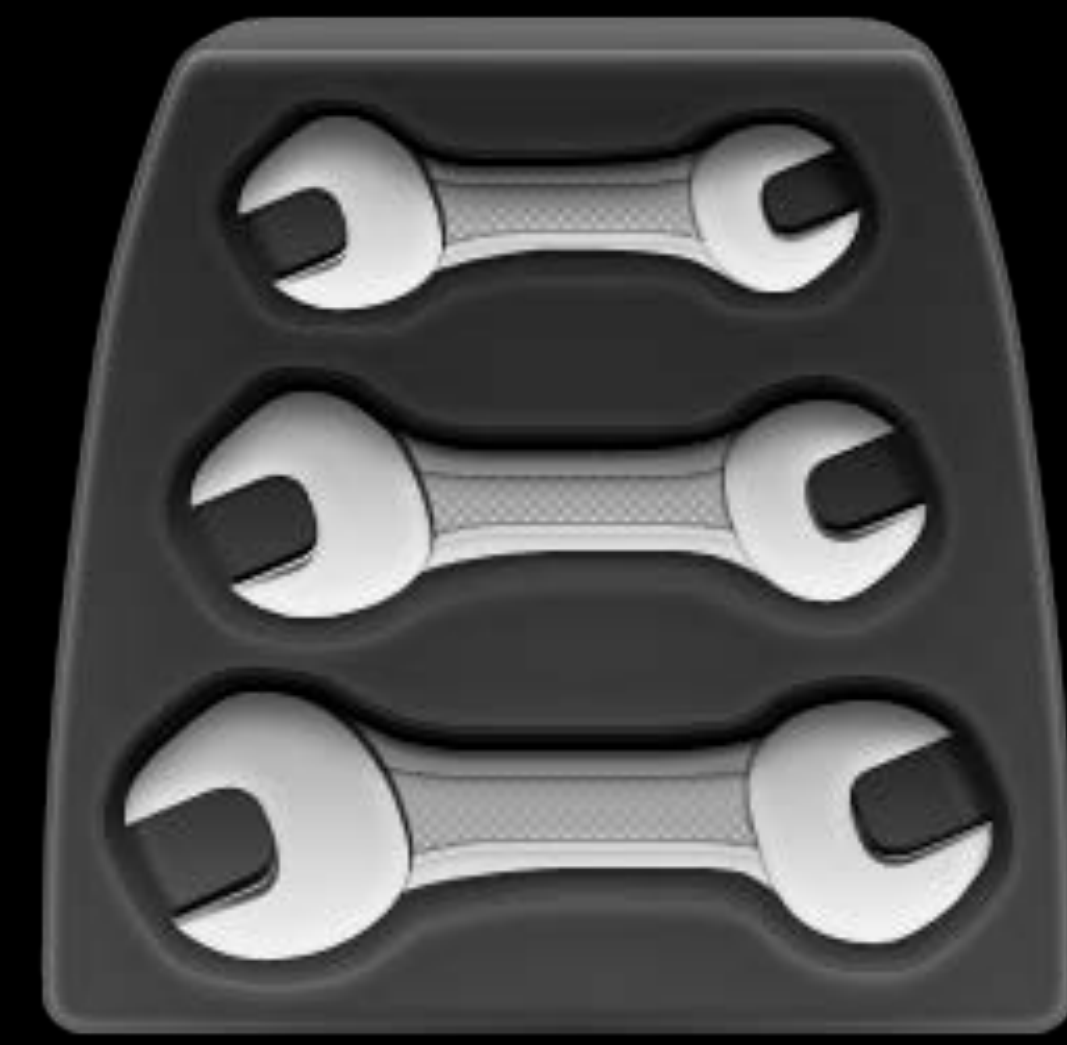
No arbitrary commands or shell scripts



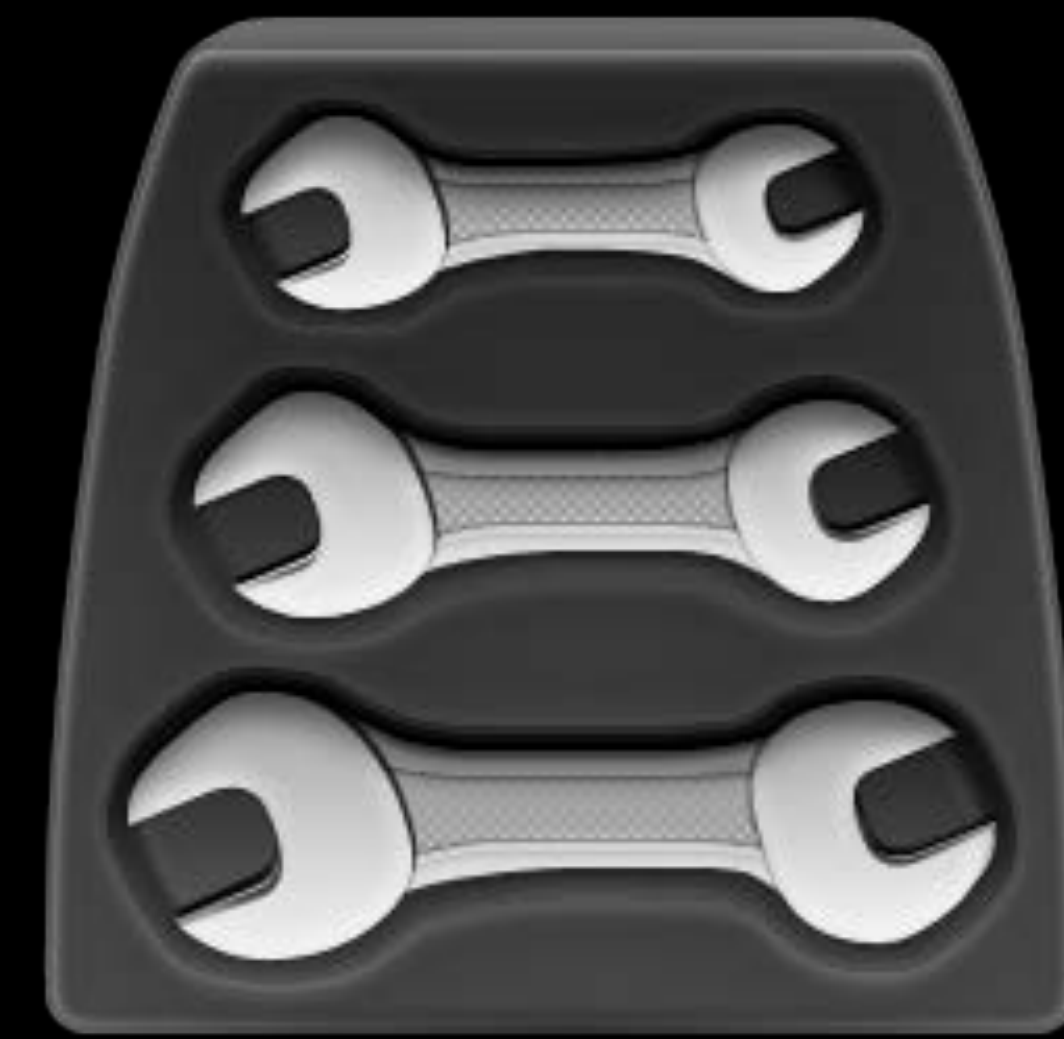
Testing



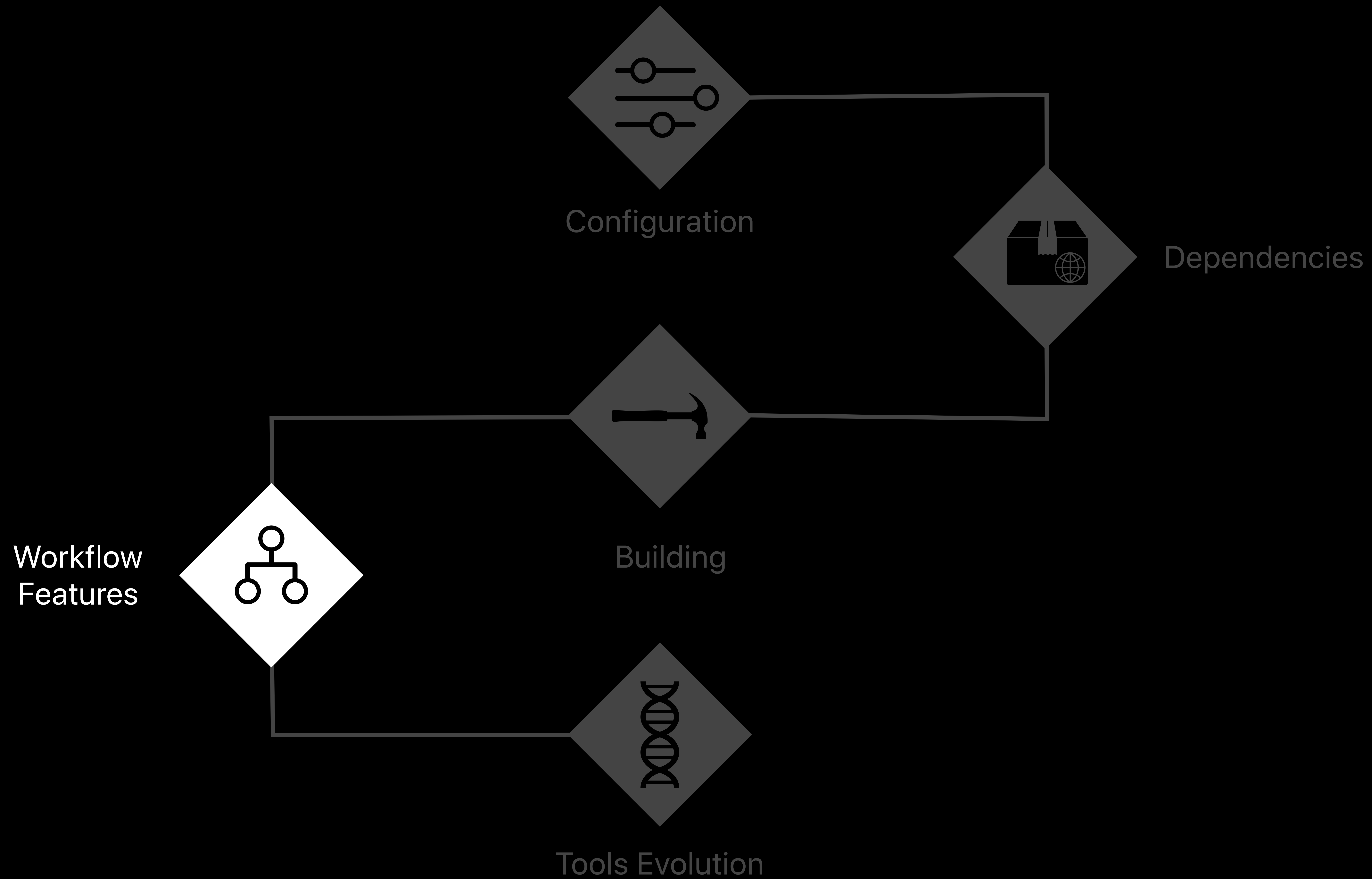
Parallel Testing



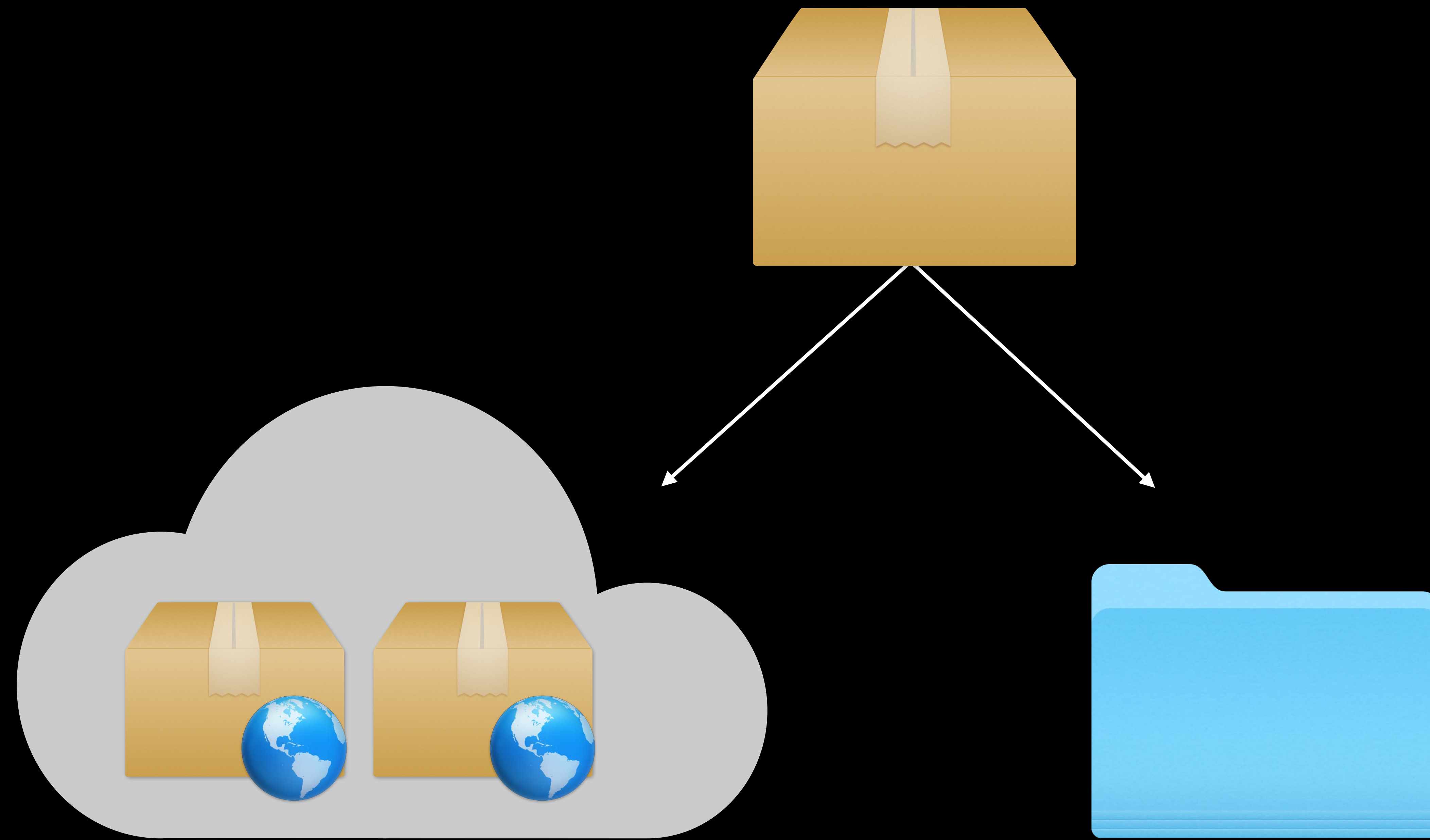
Test Filtering



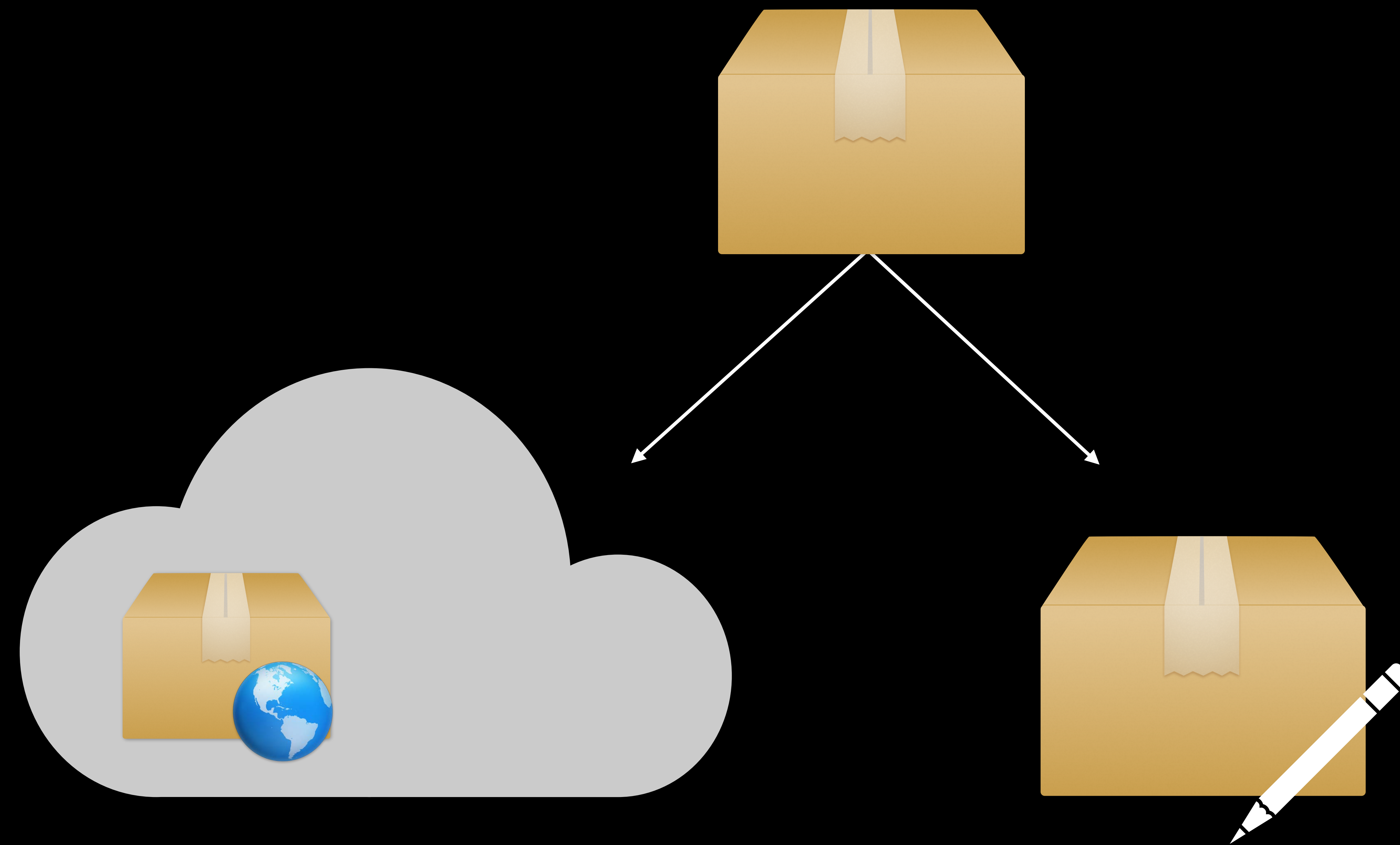
The Design of SwiftPM



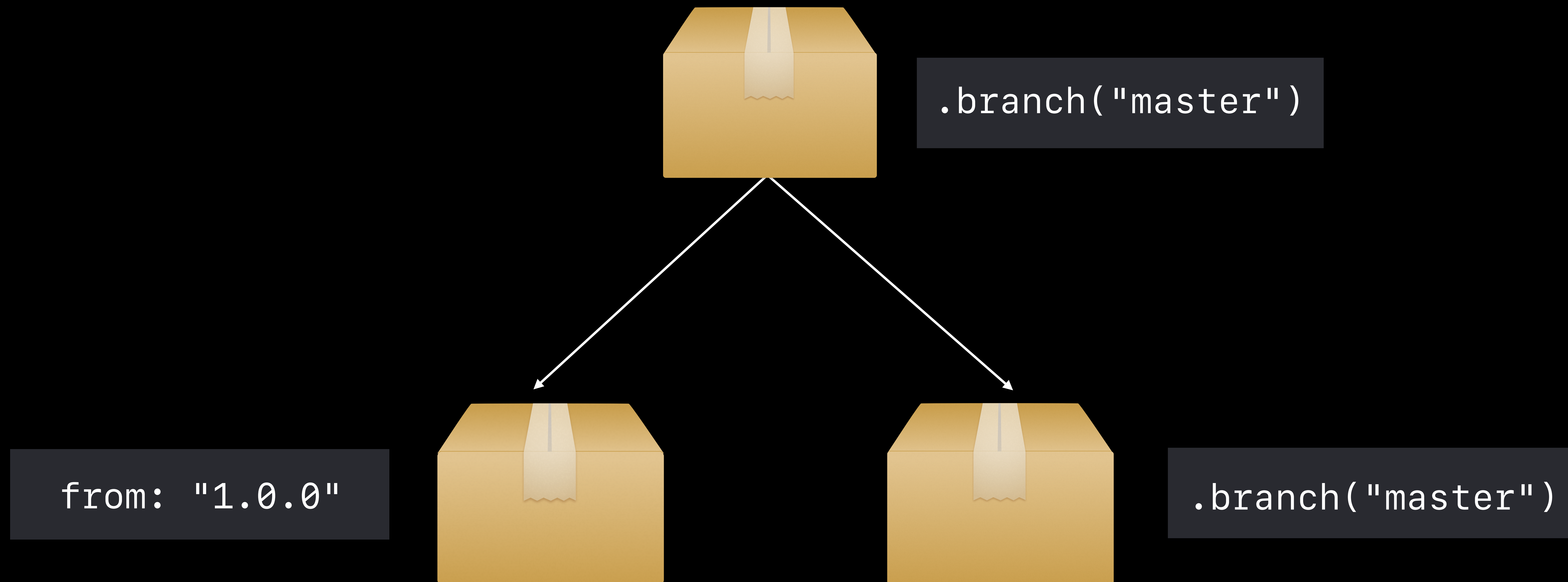
Edit Mode



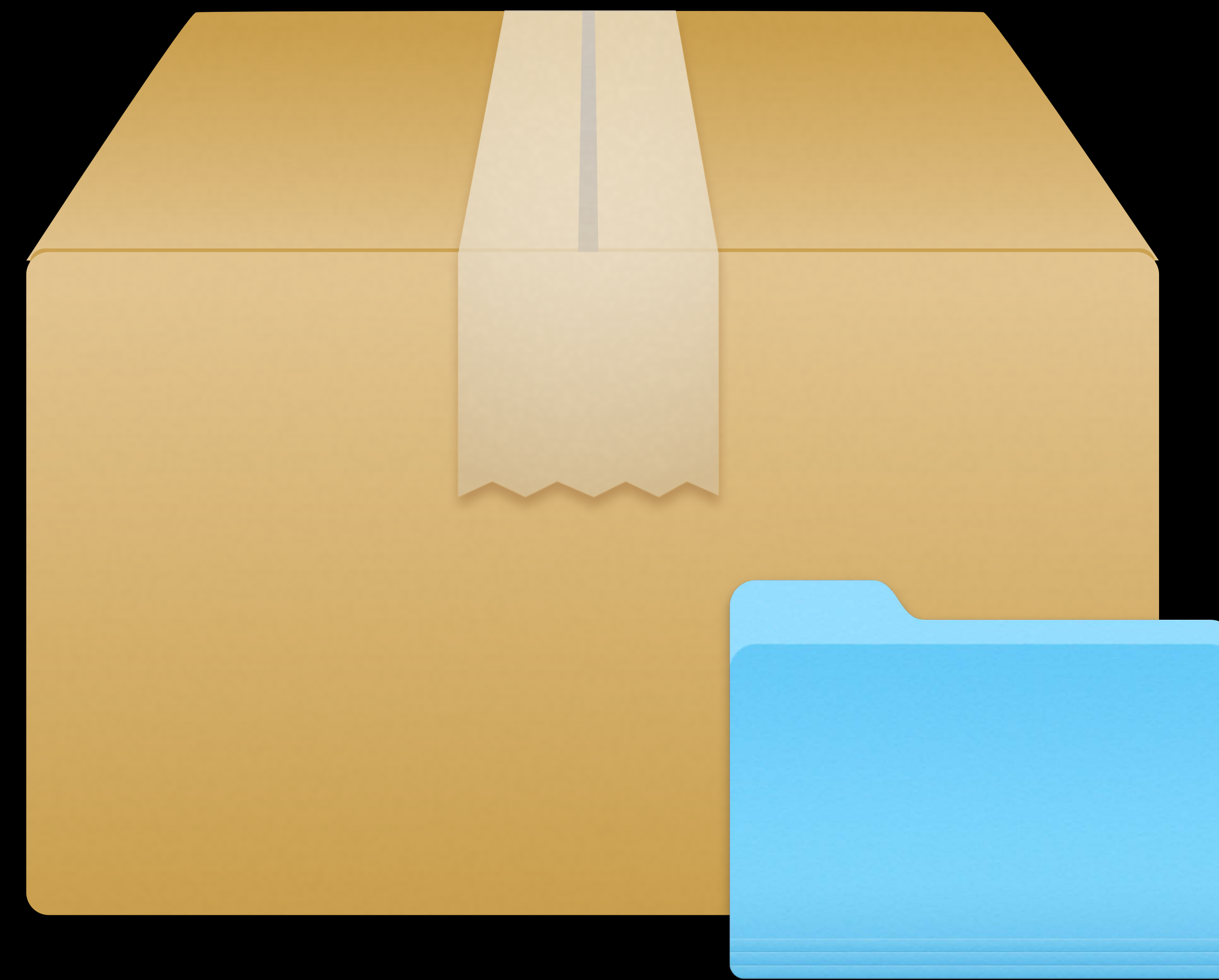
Edit Mode



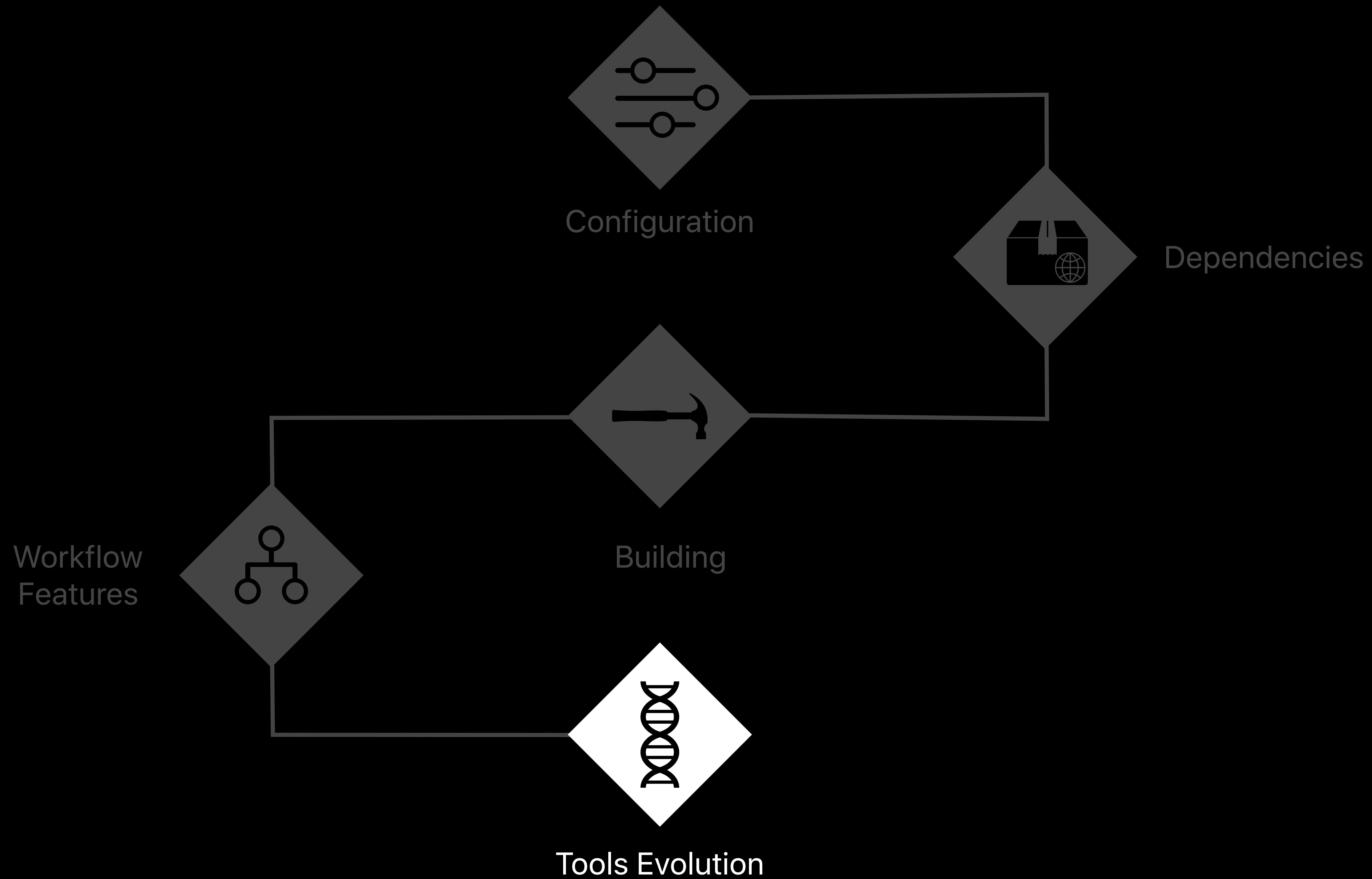
Branch Dependencies



Local Package Dependencies



The Design of SwiftPM

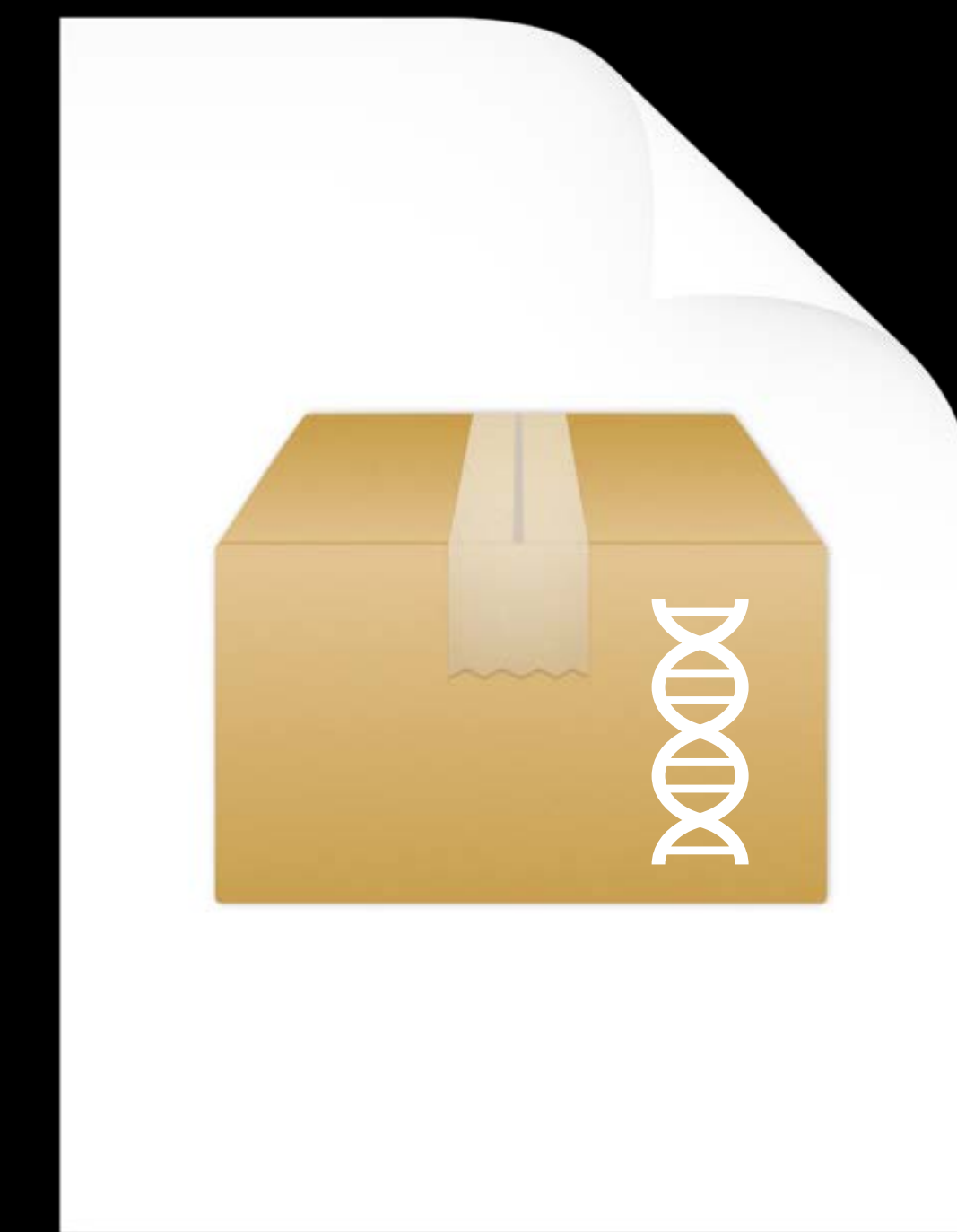


Package.swift Manifest API Evolution

Package API can be updated with each new Swift version

Previous API is still available

Allows using new Swift tools without updating the manifest



Swift Tools Version

```
// swift-tools-version:4.2
import PackageDescription

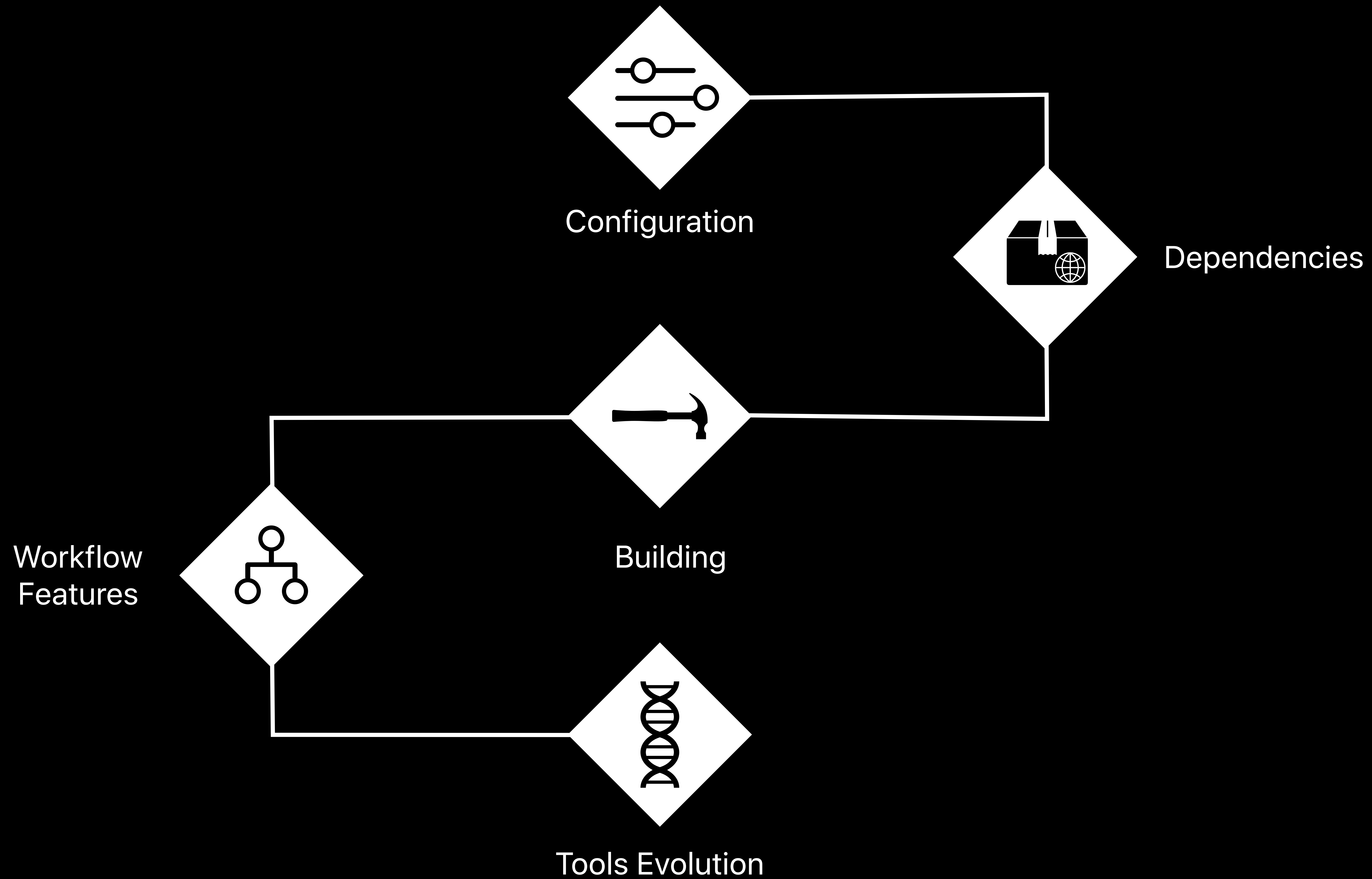
let package = Package(..., swiftLanguageVersions: [.v4_2, .v4])
```


Swift Language Version

```
// swift-tools-version:4.2
import PackageDescription

let package = Package(..., swiftLanguageVersions: [.v4_2, .v4])
```

The Design of SwiftPM



Why a package manager for Swift?

How to use it

The design of SwiftPM

Evolution ideas

Open source process

Open Evolution Process

Themes

Great integration with other tools

Publish and deploy

Support complex packages

Find, manage, and trust packages

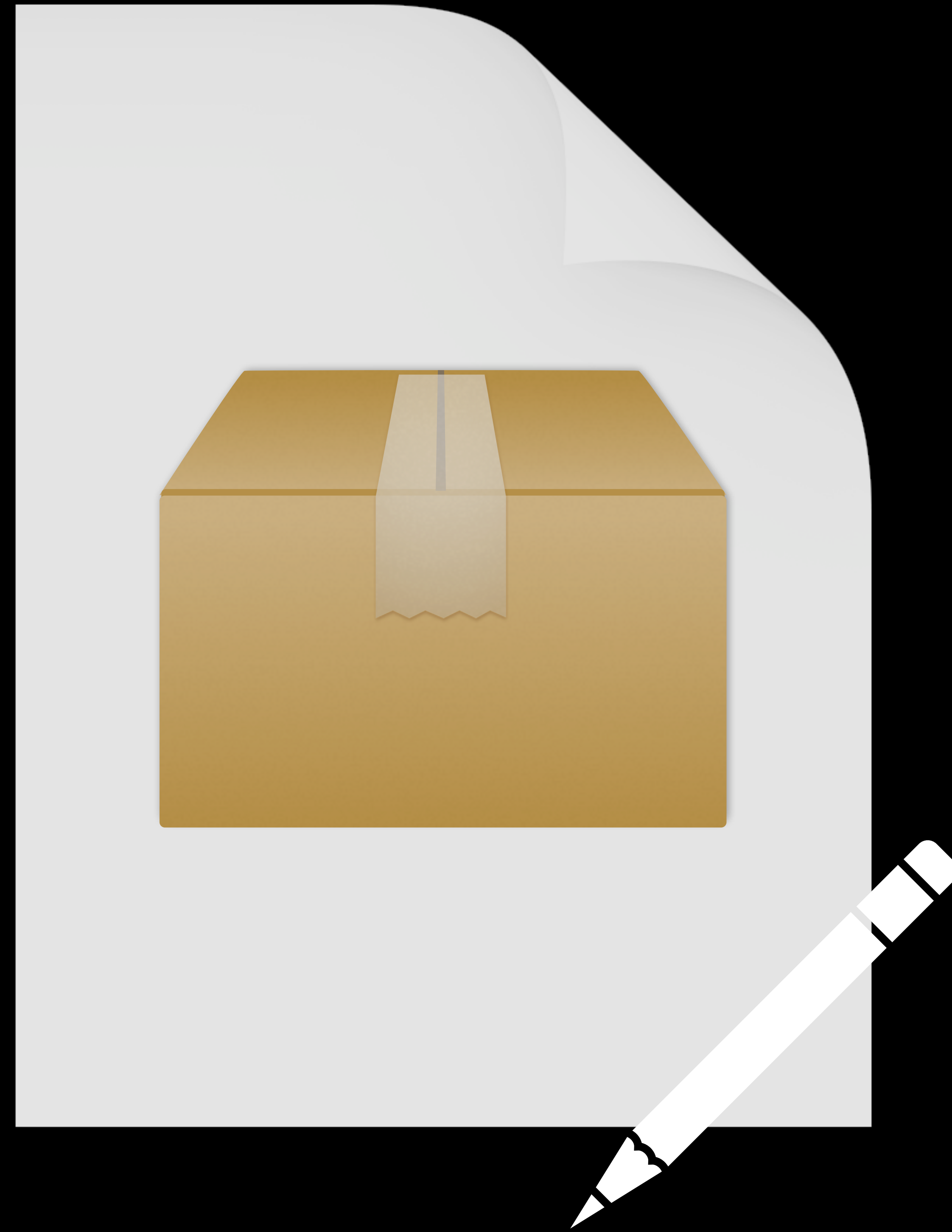
Great Integration with Other Tools

libSwiftPM Available



SwiftPM support in
developer tools is encouraged!

Idea: Machine-Editable Package.swift



Idea: Machine-Editable Package.swift



+



libSyntax

Idea: Machine-Editable Package.swift

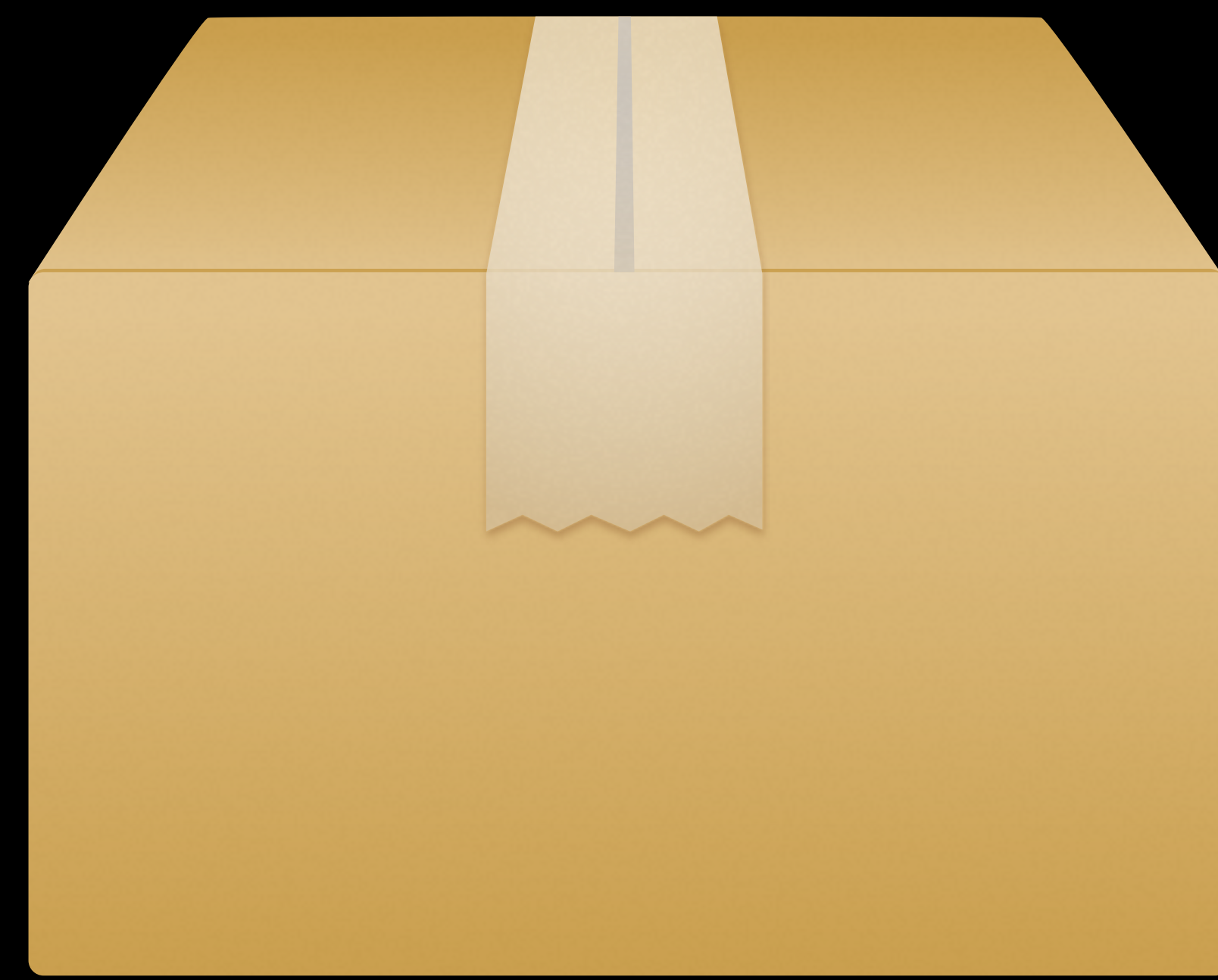
```
let package = Package(  
    name: "Networking",  
    products: [  
        .library(name: "Networking", targets: ["Networking"])  
    ],  
    targets: [  
        .target(  
            name: "Networking",  
            dependencies: []),  
    ]  
)
```

Idea: Machine-Editable Package.swift

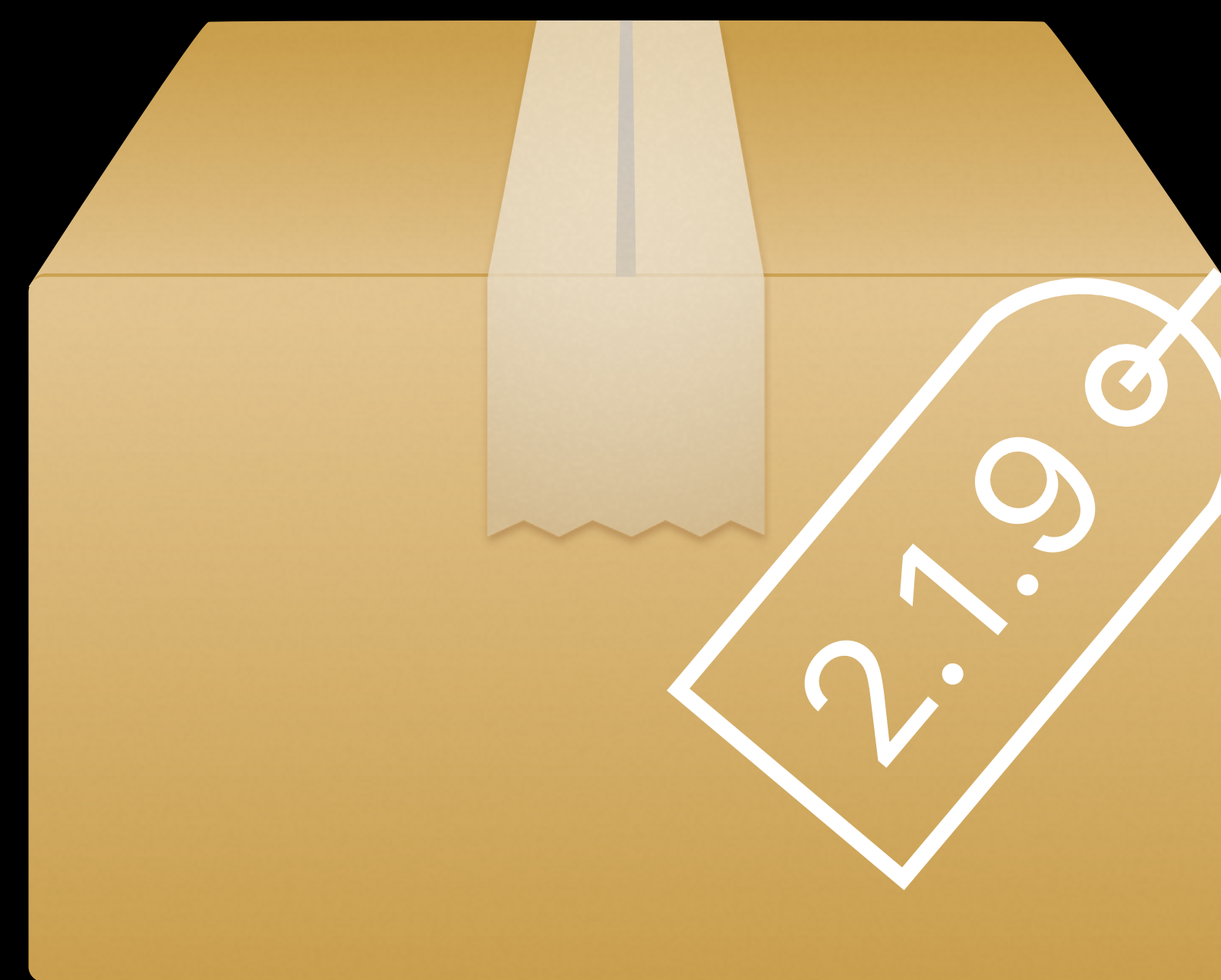
```
let package = Package(  
    name: "Networking",  
    products: [  
        .library(name: "Networking", targets: ["Networking"])  
    ],  
    targets: [  
        .target(  
            name: "Networking",  
            dependencies: ["NetworkCore"]),  
        .target(  
            name: "NetworkCore",  
            dependencies: []),  
    ]  
)
```

Publish and Deploy

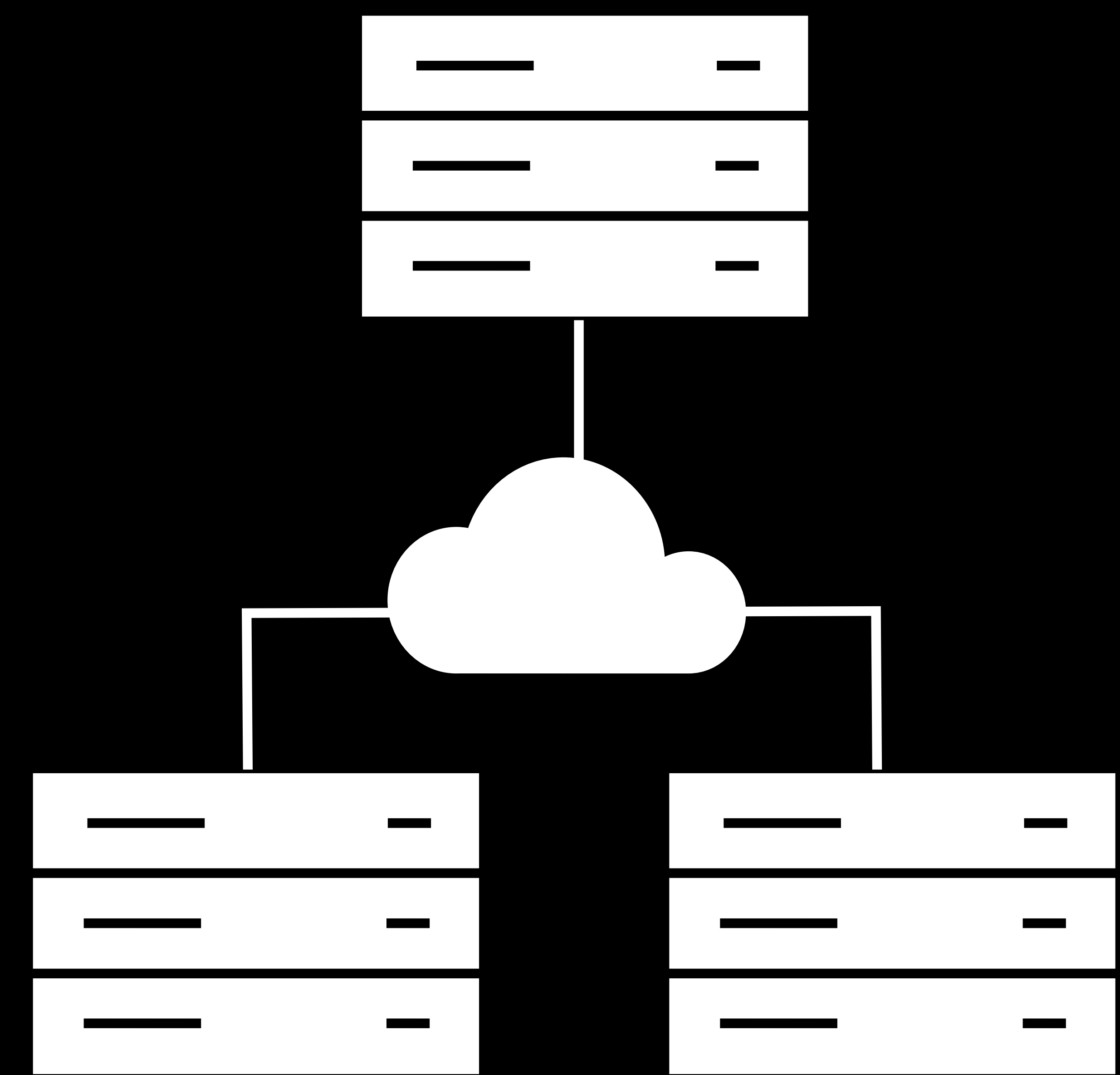
Idea: Tagging and Publication Support



\$ git tag 2.1.9



\$ git push origin 2.1.9



Idea: Automatic Semantic Versioning

```
public func findPerson(  
  firstName: String)  
  -> Contact
```



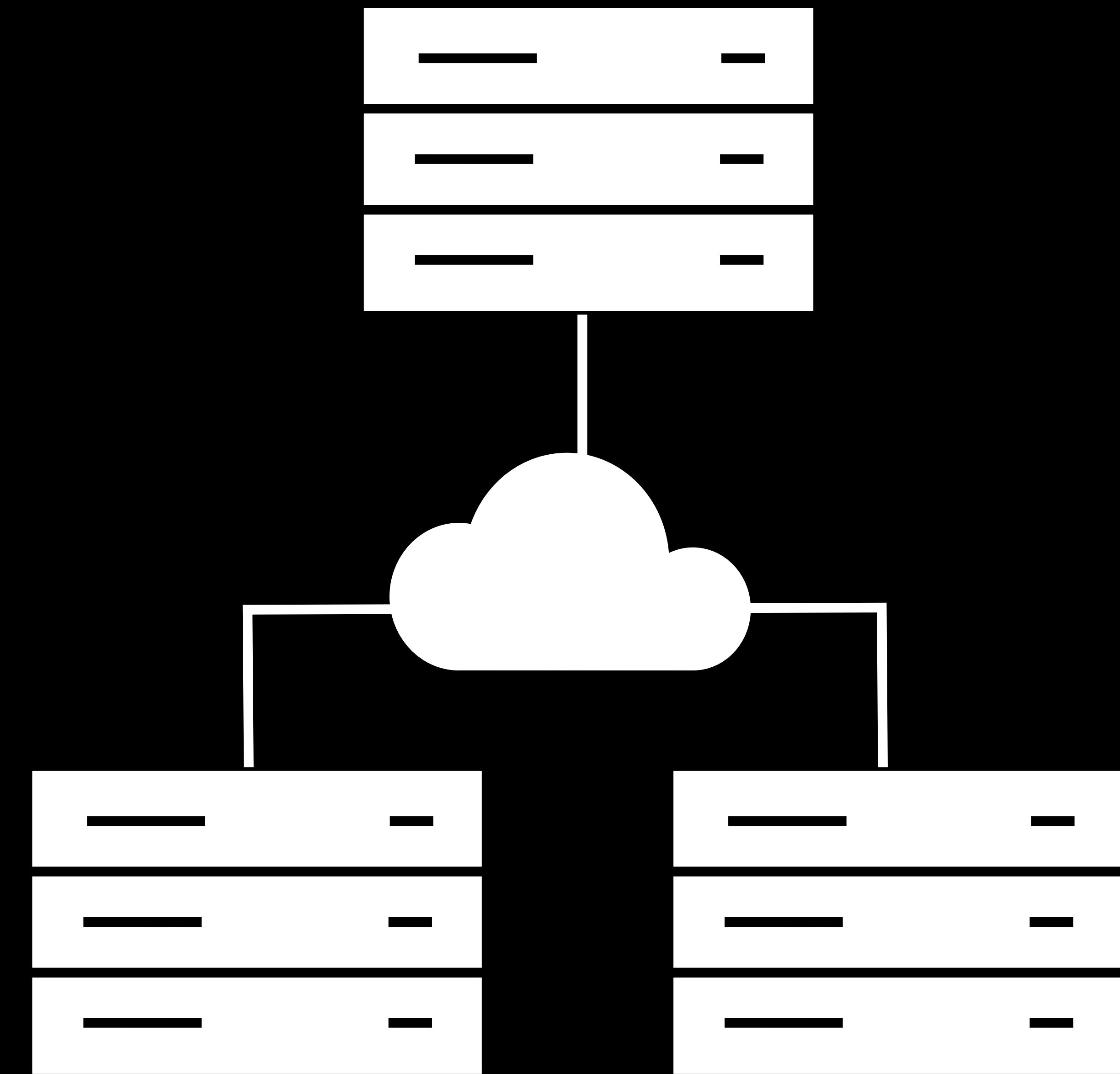
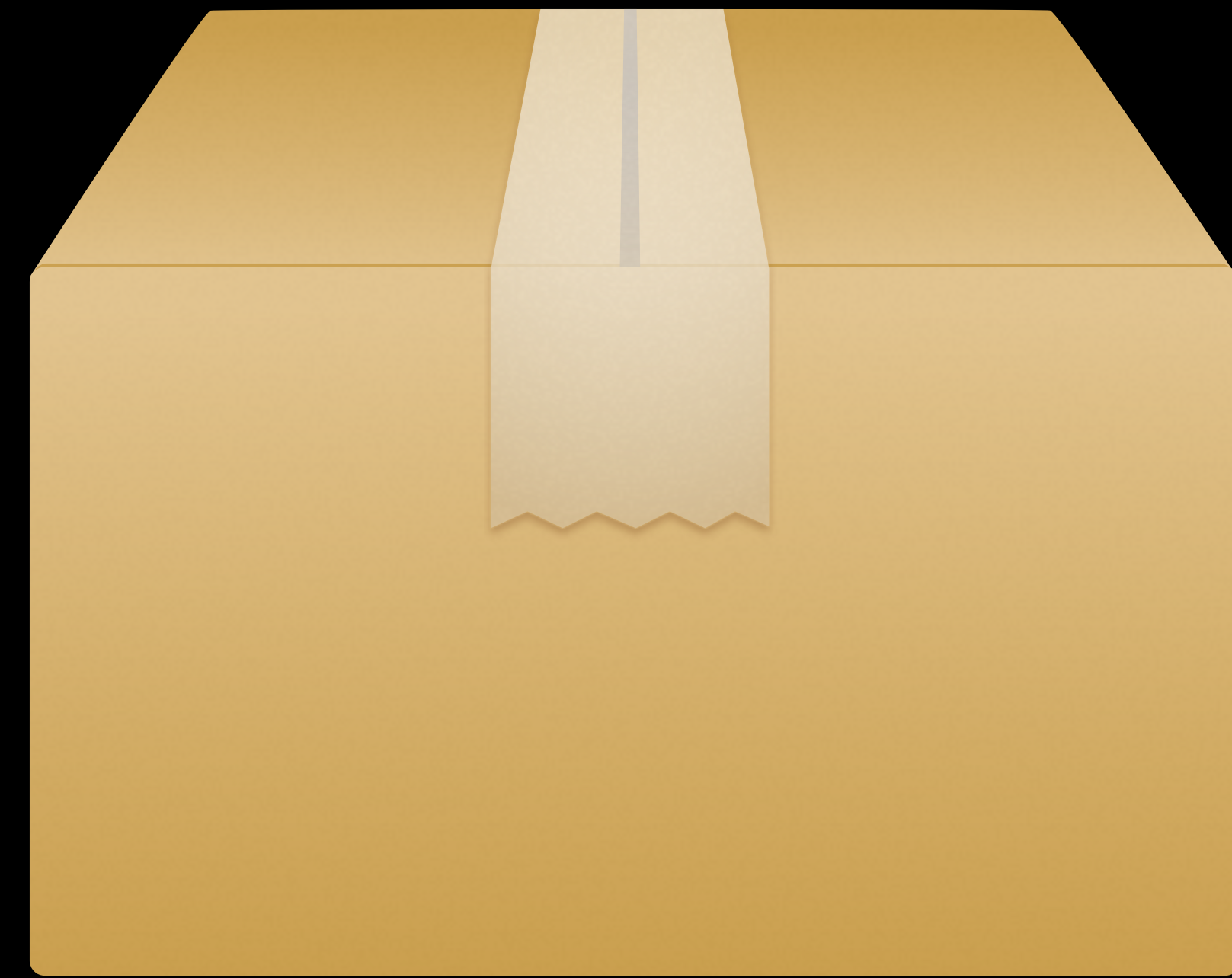
```
public func findPerson(  
  firstName: String,  
  lastName: String)  
  -> Contact
```

1.6.0

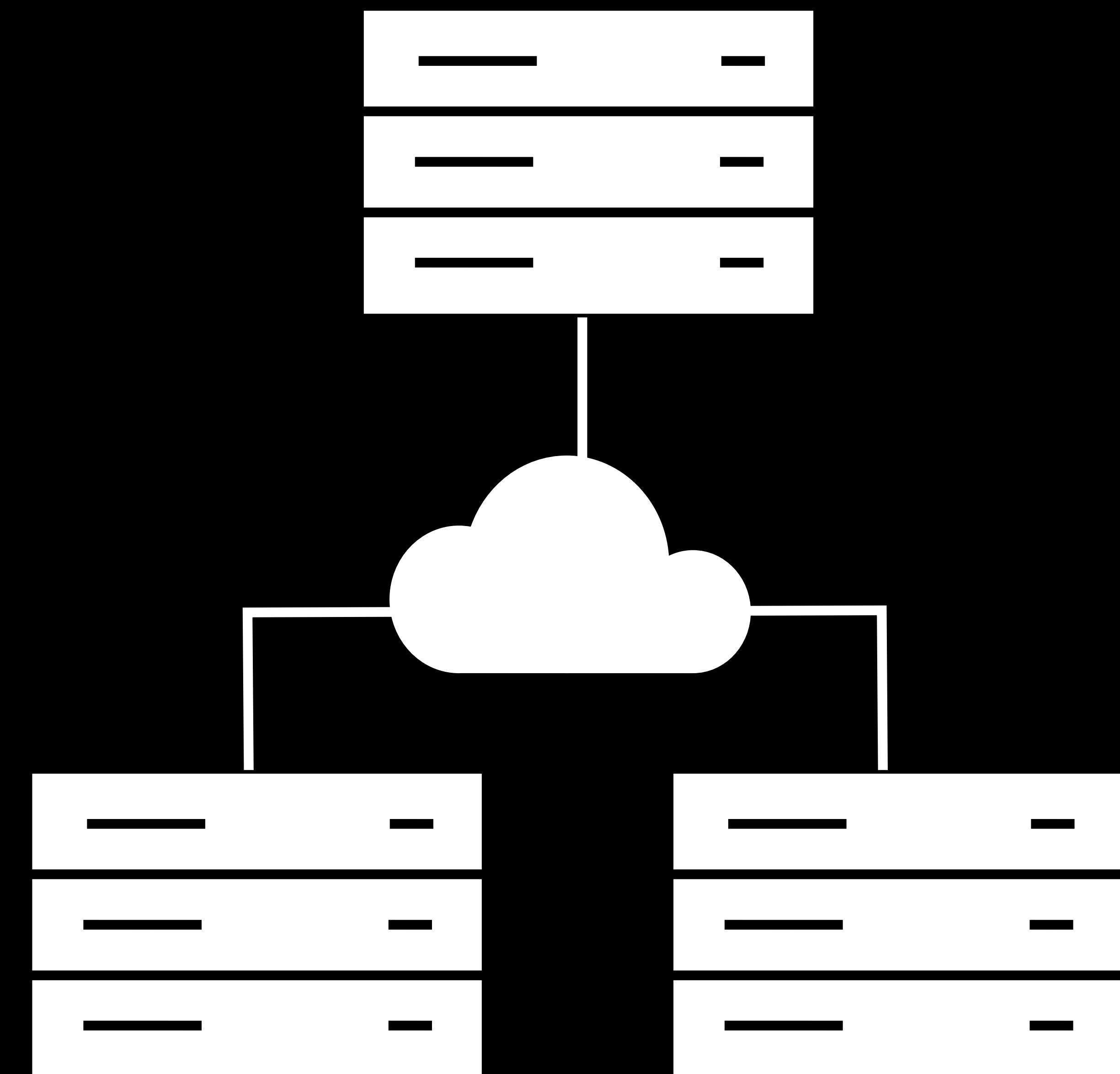
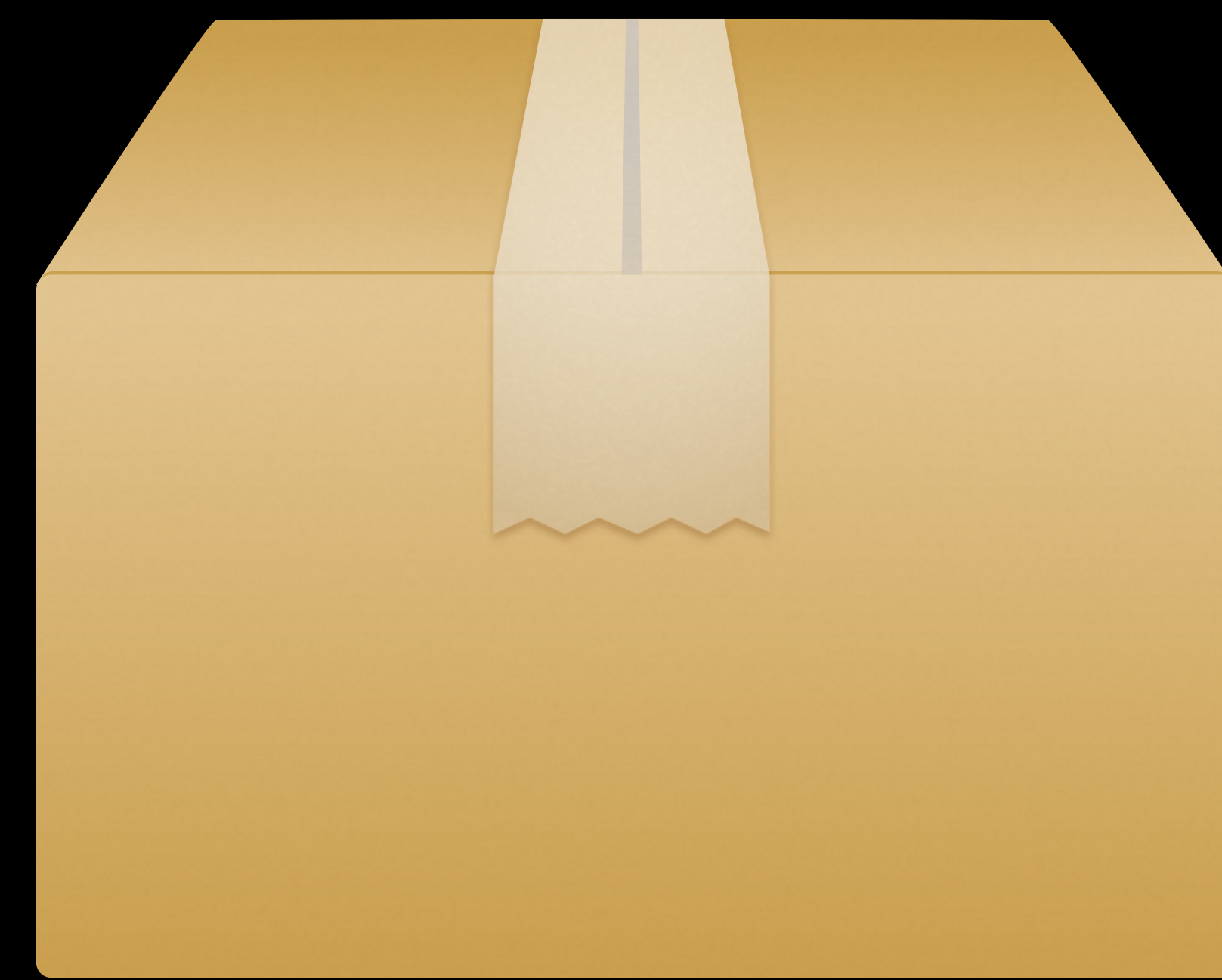


2.0.0

Idea: Deployment Automation

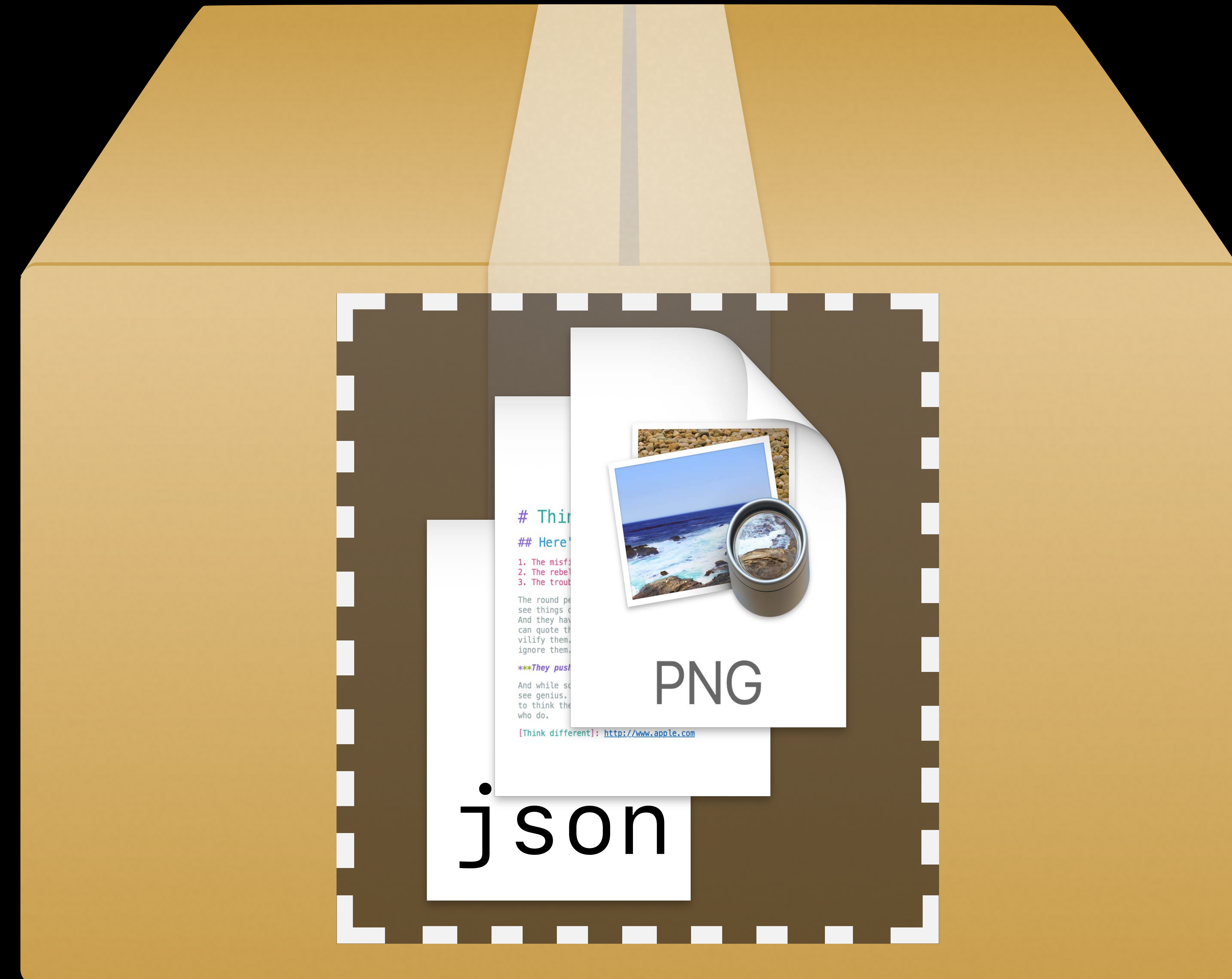


Idea: Deployment Automation



Support Complex Packages

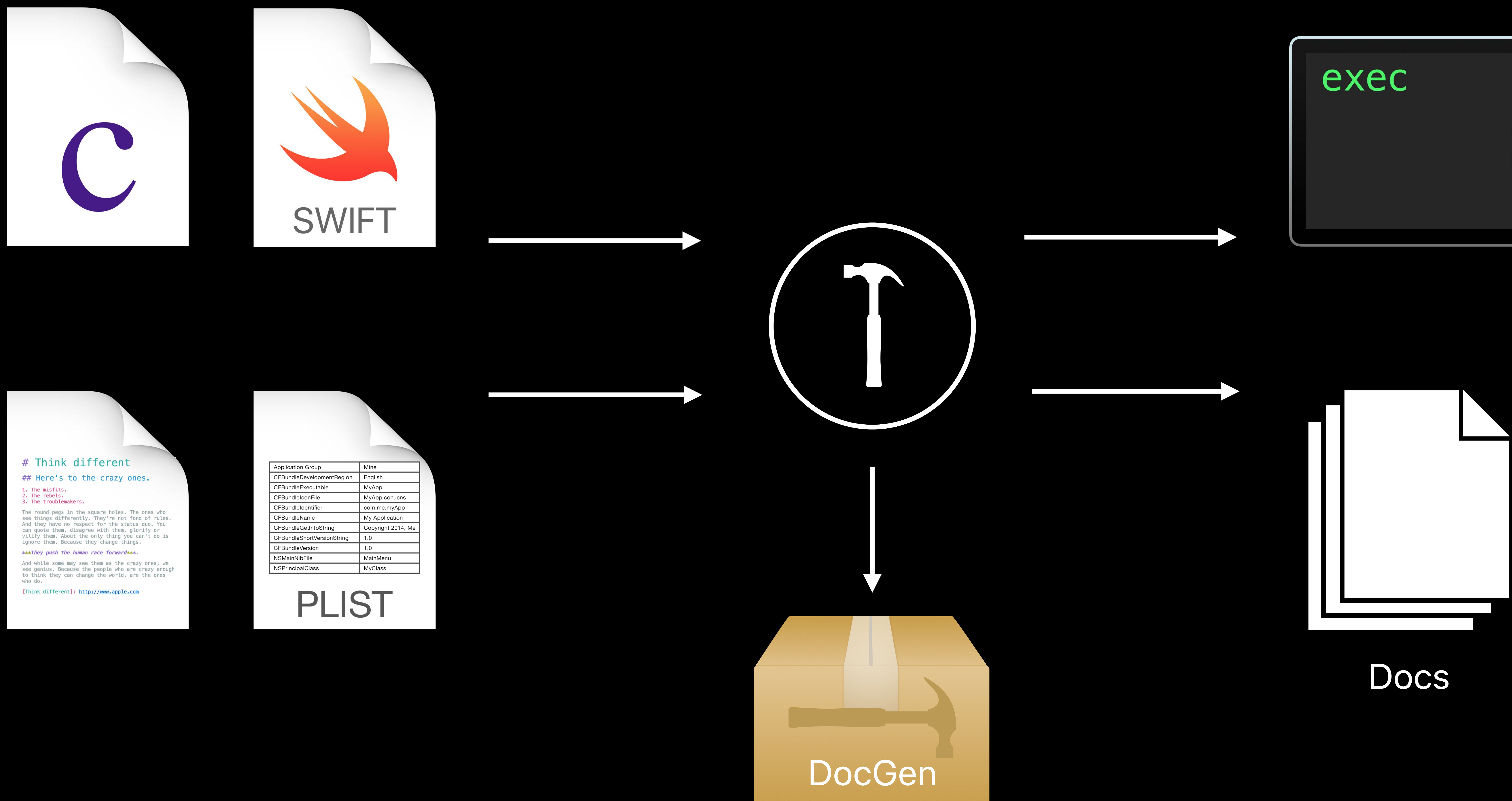
Idea: Resources



Idea: Build Settings



Idea: Extensible Build Tools



Trust, Manage, and Find Packages

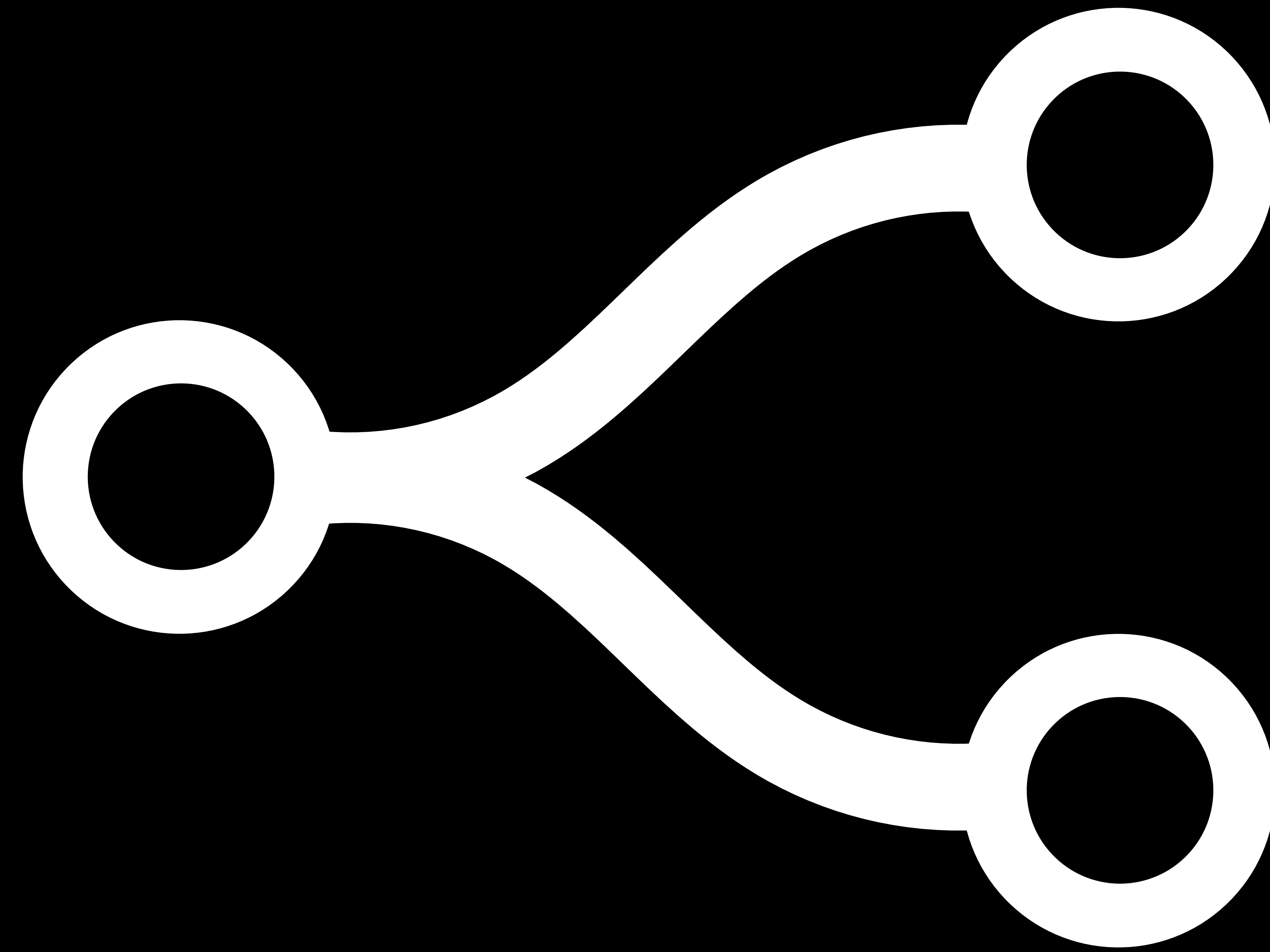
Idea: Package Content Verification



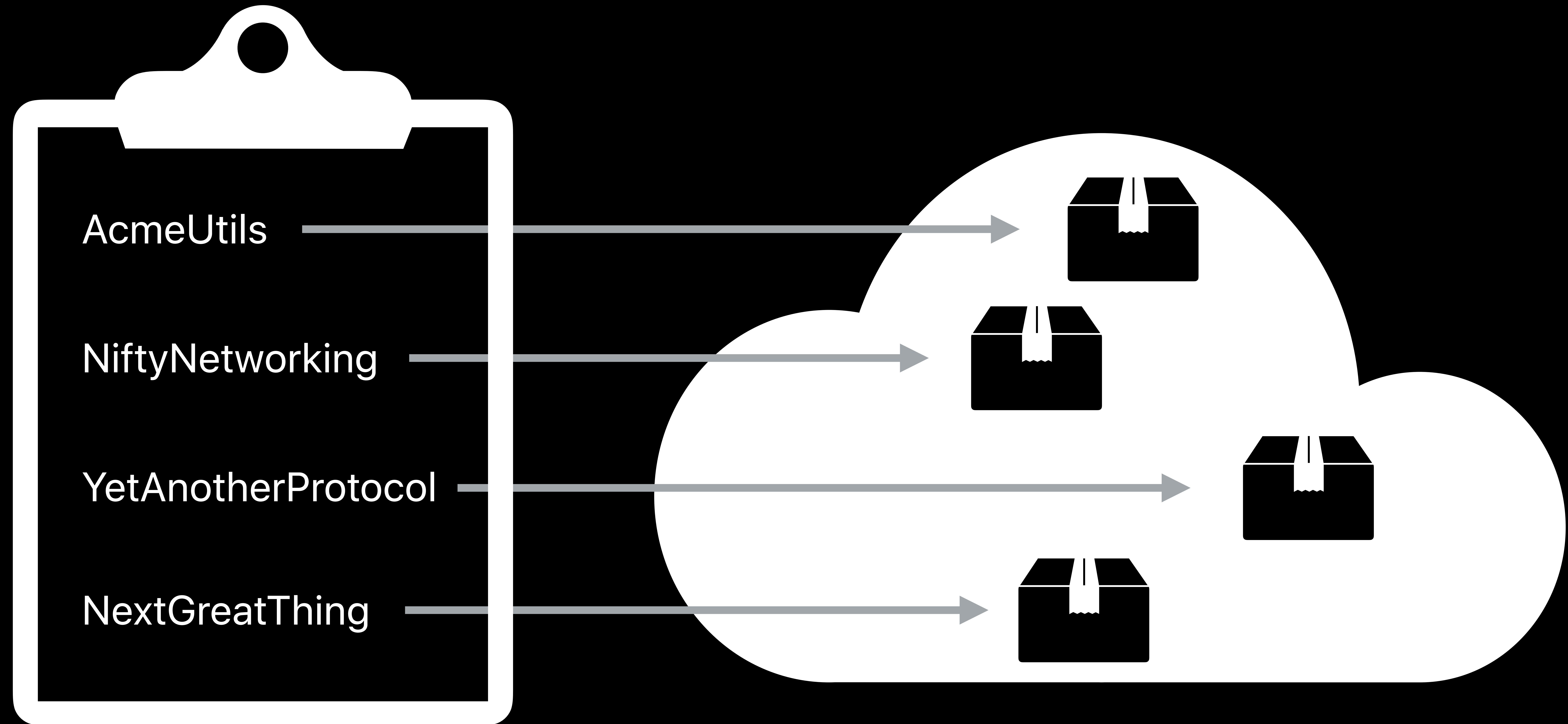
Idea: Cross-Platform Sandboxing



Idea: Fork Support



Idea: Package Index



Deployment Automation

Verify Expected Package Content

Build Settings

Extensible Build Tools

Cross-Platform Sandboxing

Your Contributions

Resources

Fork Support

Automatic SemVer

Machine-Editable Package.swift

Tag & Publish

Package Index

Leverage libSwiftPM

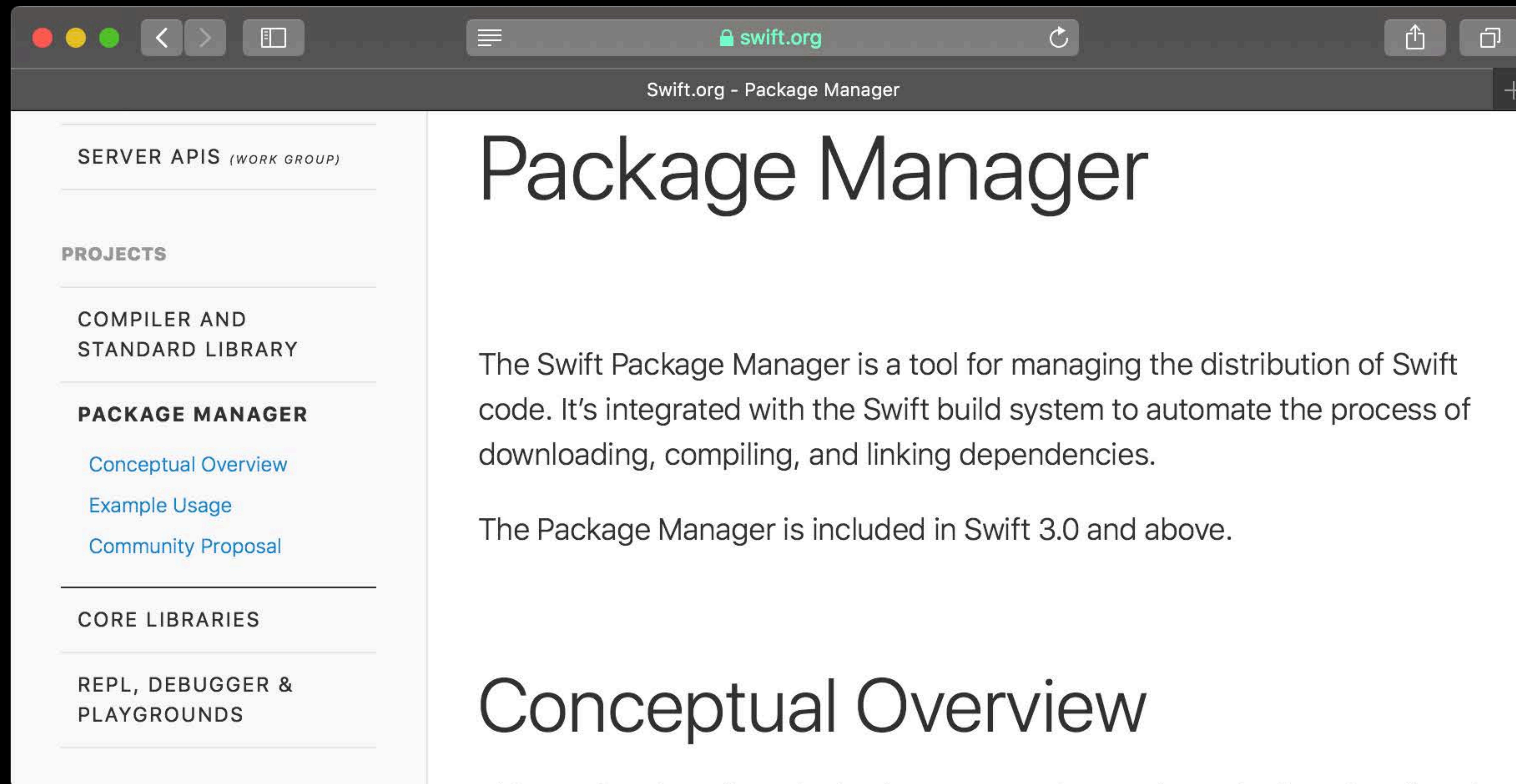
Why a package manager for Swift?

How to use it

The design of SwiftPM

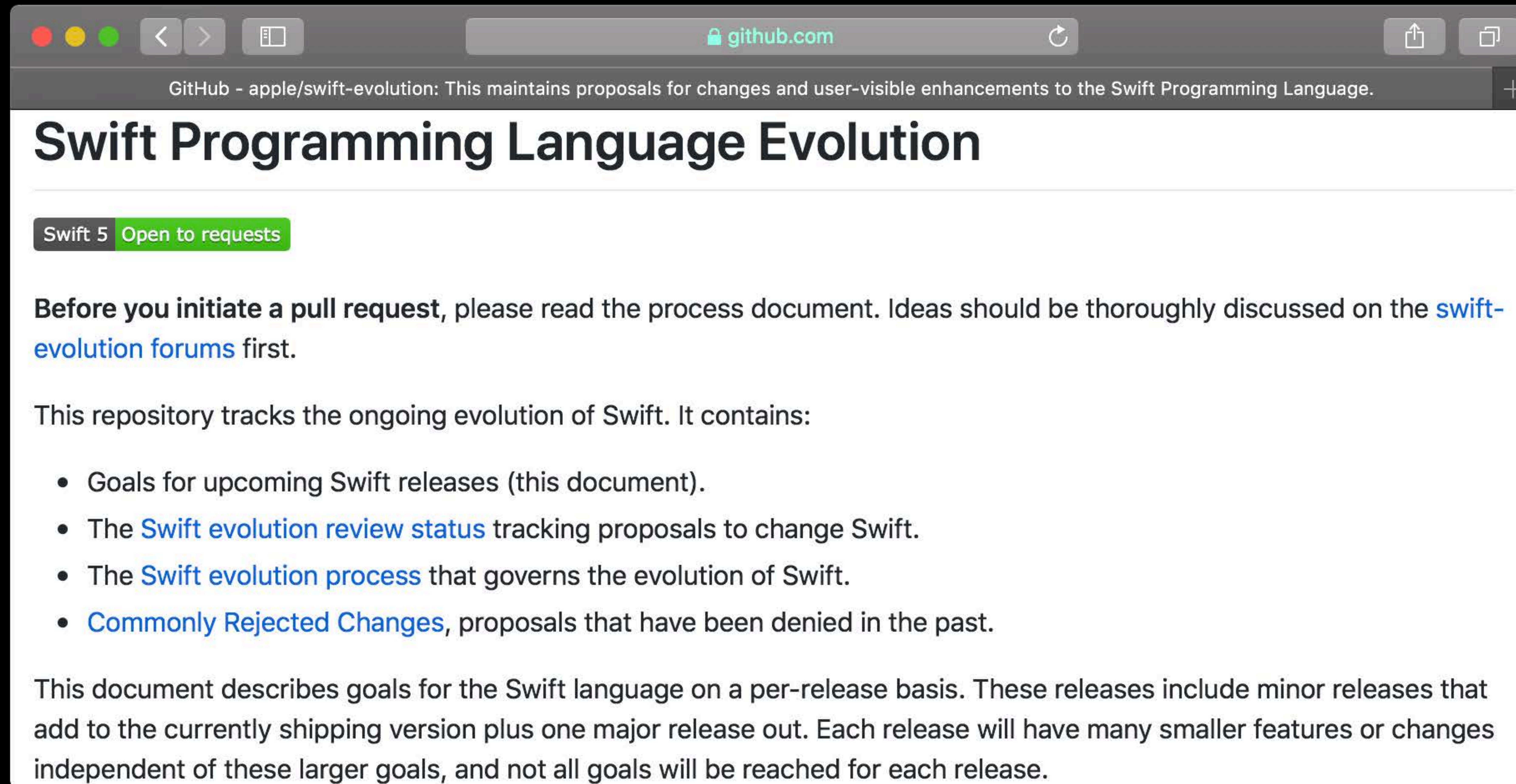
Evolution ideas

Open source process



<https://swift.org/package-manager/>

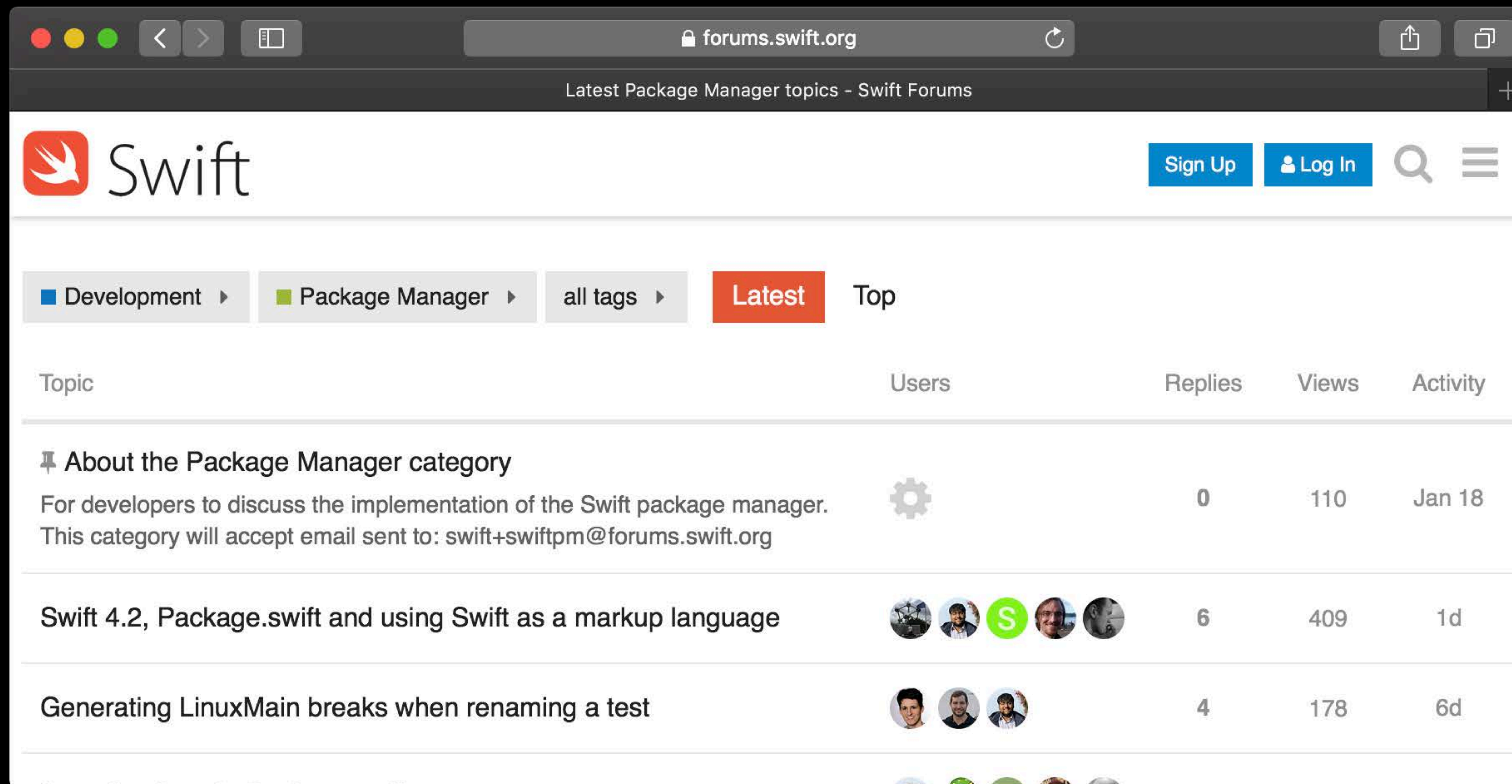
Swift Evolution



The screenshot shows a web browser window with the address bar displaying 'github.com'. The page title is 'GitHub - apple/swift-evolution: This maintains proposals for changes and user-visible enhancements to the Swift Programming Language.' The main heading is 'Swift Programming Language Evolution'. Below the heading, there is a green button labeled 'Swift 5 Open to requests'. The text on the page reads: 'Before you initiate a pull request, please read the process document. Ideas should be thoroughly discussed on the [swift-evolution forums](#) first.' This is followed by a paragraph: 'This repository tracks the ongoing evolution of Swift. It contains:' and a bulleted list: '• 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.', and '• [Commonly Rejected Changes](#), proposals that have been denied in the past.' The final paragraph states: '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.'

<https://github.com/apple/swift-evolution>

Swift Forums



The screenshot shows a web browser window with the URL `forums.swift.org`. The page title is "Latest Package Manager topics - Swift Forums". The Swift logo is in the top left, and "Sign Up" and "Log In" buttons are in the top right. Below the navigation bar, there are tabs for "Development", "Package Manager", "all tags", "Latest", and "Top". The "Latest" tab is selected. The main content area displays a list of forum topics with columns for "Topic", "Users", "Replies", "Views", and "Activity".

Topic	Users	Replies	Views	Activity
⚙ About the Package Manager category For developers to discuss the implementation of the Swift package manager. This category will accept email sent to: <code>swift+swiftpm@forums.swift.org</code>		0	110	Jan 18
Swift 4.2, Package.swift and using Swift as a markup language		6	409	1d
Generating LinuxMain breaks when renaming a test		4	178	6d

<https://forums.swift.org/c/development/SwiftPM>

Swift Bug Tracker

The screenshot shows a web browser window with the URL `bugs.swift.org`. The page title is "[SR-4186] Support enforcing minimum tools version based on PackageDescription API used - Swift". The navigation bar includes the Swift logo, "Dashboards", "Projects", "Issues", a search bar, and a "Log In" link. The main content area features a search filter section with "Swift" selected, "Type: All", "Status: All", "Assignee: All", and a search box containing "Contains text". Below this, filters for "Resolution: Unresolved" and "Component: Package Manager" are visible. A list of search results is shown on the left, with SR-4186 selected. The details for SR-4186 are displayed in the center, including its type (New Feature), status (OPEN), priority (Medium), resolution (Unresolved), component (Package Manager), and radar URL. The description states: "We should support enforcing a minimum swift-tools-version based on the newest PackageDescription API in use." On the right, the "People" section lists the assignee (Ankit Aggarwal) and reporter (Rick Ballard), along with voting and watching options.

Search Export Tools

Swift Type: All Status: All Assignee: All More Advanced ⌵

Resolution: Unresolved ✕ Component: Package Manager ✕

+ SR-4186
[Support enforcing minimum tool...](#)

SR-4084
[Automate setting development b...](#)

SR-4045
[Increase test coverage of null bu...](#)

+ SR-4402
[Possible Workspace API improv...](#)

SR-4394

⌵ 1 2 3 4 ⌵

Details

Type: **+ New Feature**

Status: **OPEN**

Priority: **↑ Medium**

Resolution: **Unresolved**

Component/s: **Package Manager**

Labels: **None**

Radar URL: **<rdar://problem/40205471>**

Description

We should support enforcing a minimum swift-tools-version based on the newest PackageDescription API in use.

People

Assignee: **Ankit Aggarwal**

Reporter: **Rick Ballard**

Votes: **0** [Vote for this issue](#)

Watchers: **2** [Start watching this issue](#)

Dates

<https://bugs.swift.org>

Swift Bug Tracker—Starter Bugs

The screenshot shows a web browser window at bugs.swift.org. The page title is "[SR-1402] Check dependency graph for possible collisions - Swift". The navigation bar includes "Swift", "Dashboards", "Projects", "Issues", a search bar, and "Log In".

The search filters are set to: Swift, Type: All, Status: All, Assignee: All, Contains text, More, Advanced. The selected filters are: Resolution: Unresolved, Component: Package Manager, Label: StarterBug.

Search Results:

- [SR-1402](#): Check dependency graph for po...
- [SR-6978](#): SwiftPM ignores vX.X.X style ta...
- [SR-7559](#): Building swiftpm requires rsync, ...
- [SR-4329](#): SwiftPM should have a feature t...
- [SR-7279](#)

Details for SR-1402:

- Type: Bug
- Status: OPEN
- Priority: Medium
- Resolution: Unresolved
- Component/s: Package Manager
- Labels: StarterBug
- Radar URL: rdar://problem/40206312

Description:

if an external dependency has a module name same as a module in root package the root package module is not compiled at all. This should

People:

- Assignee: Unassigned
- Reporter: Ankit Aggarwal
- Votes: 0 Vote for this issue
- Watchers: 4 Start watching this issue

Dates:

<https://bugs.swift.org>

Swift Bug Tracker—Starter Bugs

The screenshot shows the Swift Bug Tracker interface. The browser address bar displays `bugs.swift.org`. The page title is "[SR-1402] Check dependency graph for possible collisions - Swift". The navigation menu includes "Swift", "Dashboards", "Projects", and "Issues". A search bar is present with a "Search" button and a "Log In" link. The main content area shows a search filter for "Label: StarterBug" highlighted in a white box. Below the filter, a list of bugs is shown, with the first one, SR-1402, selected. The details for SR-1402 are displayed in a table-like format:

Type:	Bug
Status:	OPEN
Priority:	Medium
Resolution:	Unresolved
Component/s:	Package Manager
Labels:	StarterBug
Radar URL:	rdar://problem/40206312

The description for SR-1402 is: "if an external dependency has a module name same as a module in root package the root package module is not compiled at all. This should". The "People" section shows the assignee as "Unassigned" and the reporter as "Ankit Aggarwal". There are 0 votes and 4 watchers for this issue.

<https://bugs.swift.org>

Swift Bug Tracker—Starter Bugs

The screenshot shows a web browser window at bugs.swift.org. The page title is "[SR-1402] Check dependency graph for possible collisions - Swift". The navigation bar includes "Swift", "Dashboards", "Projects", "Issues", a search bar, and "Log In".

The search filters are set to: Swift, Type: All, Status: All, Assignee: All, Contains text, More, Advanced. The selected filters are: Resolution: Unresolved, Component: Package Manager, Label: StarterBug.

The search results list several issues, with SR-1402 selected. The details for SR-1402 are:

- Type: Bug
- Status: OPEN
- Priority: Medium
- Resolution: Unresolved
- Component/s: Package Manager
- Labels: StarterBug
- Radar URL: rdar://problem/40206312

The description for SR-1402 is: "if an external dependency has a module name same as a module in root package the root package module is not compiled at all. This should".

The "People" section shows: Assignee: Unassigned, Reporter: Ankit Aggarwal, Votes: 0 (Vote for this issue), Watchers: 4 (Start watching this issue).

<https://bugs.swift.org>

Swift Continuous Integration



ci.swift.org

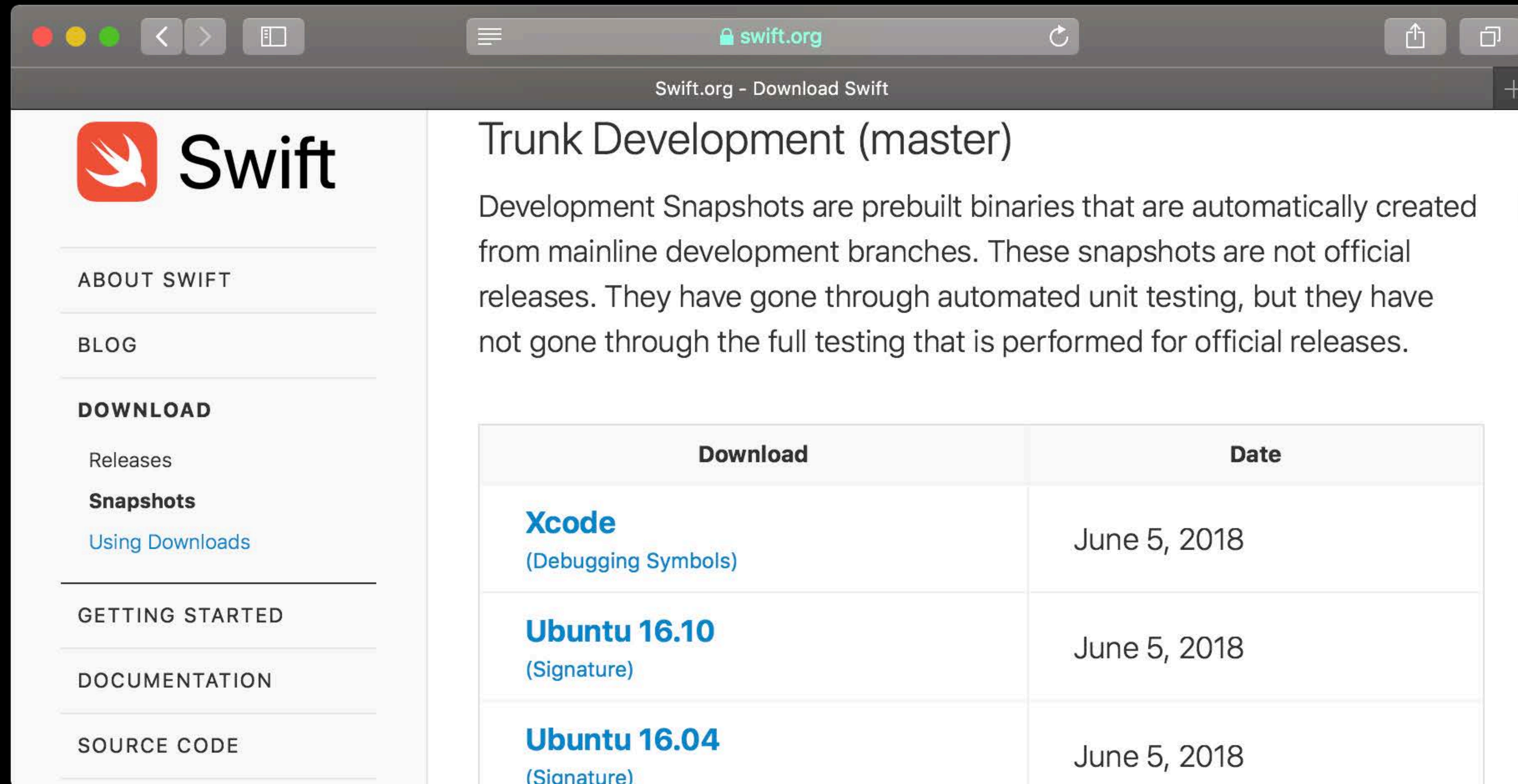
Pull Request [Jenkins]

Jenkins Pull Request ENABLE AUTO REFRESH

Trigger (swift 4.2) #160	Swift Package Manager Request Test Linux	13 days - #16	3 mo 13 days - #8	1 hr 18 min
9 OSS - Swift Package Trigger (master) #1752	Swift Package Manager Request Test Linux (smoke test)	4 days 20 hr - #321	13 days - #315	29 min
10 Idle	Swift Package Manager Request Test OS X	13 days - #16	3 mo 13 days - #7	33 min
macOS-05	Swift Package Manager Request Test OS X (smoke test)	4 days 20 hr - #311	1 mo 13 days - #290	51 min
1 1. OSS - Swift (Tools Opt+Assert, Stdlib Opt+DebInfo+Assert, Test Simulator) - OS X (master) #857	Swift Package Manager Test and Merge	2 mo 23 days - #1	N/A	52 min
macOS-06	Swift Pull Request ASAN OS X	4 days 23 hr - #33	2 mo 19 days - #28	4 hr 21 min
1 1. OSS - Swift (Tools	Swift Pull Request Benchmark OS X	6 days 2 hr - #238	18 days - #225	6 hr 22 min

@swift-ci please test

Trunk Snapshots



The screenshot shows the Swift.org website with the following content:

- Navigation Menu:**
 - ABOUT SWIFT
 - BLOG
 - DOWNLOAD**
 - Releases
 - Snapshots**
 - [Using Downloads](#)
 - GETTING STARTED
 - DOCUMENTATION
 - SOURCE CODE
- Main Content:**
 - ## 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.

Download	Date
Xcode (Debugging Symbols)	June 5, 2018
Ubuntu 16.10 (Signature)	June 5, 2018
Ubuntu 16.04 (Signature)	June 5, 2018

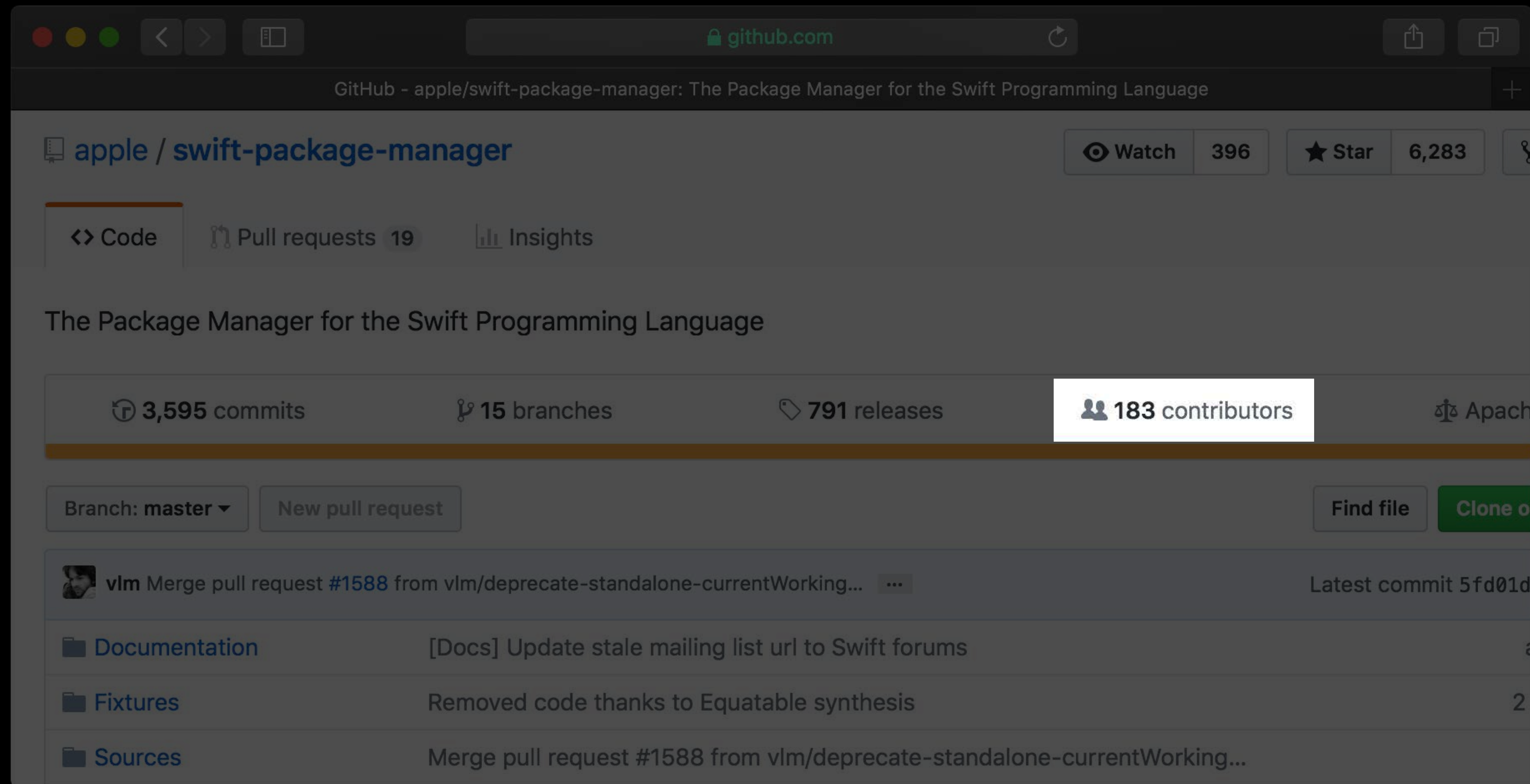
<https://swift.org/download/#snapshots>

Growing Community

The screenshot shows the GitHub repository page for `apple/swift-package-manager`. The browser address bar shows `github.com`. The repository name is `apple / swift-package-manager`. The page features several statistics: 396 Watchers, 6,283 Stars, 3,595 commits, 15 branches, 791 releases, and 183 contributors. The Apache License logo is also visible. The current branch is `master`. A recent pull request by `vlm` is highlighted, titled "Merge pull request #1588 from vlm/deprecate-standalone-currentWorking...". Below the pull request, a list of files is shown:

File	Commit Message
Documentation	[Docs] Update stale mailing list url to Swift forums
Fixtures	Removed code thanks to Equatable synthesis
Sources	Merge pull request #1588 from vlm/deprecate-standalone-currentWorking...

Growing Community



The screenshot shows the GitHub repository page for 'apple/swift-package-manager'. The repository is described as 'The Package Manager for the Swift Programming Language'. It has 3,595 commits, 15 branches, 791 releases, and 183 contributors. The '183 contributors' text is highlighted with a white box. The page also shows a pull request by user 'vlm' and a file tree with folders like 'Documentation', 'Fixtures', and 'Sources'.

github.com

GitHub - apple/swift-package-manager: The Package Manager for the Swift Programming Language

apple / swift-package-manager

Watch 396 Star 6,283

Code Pull requests 19 Insights

The Package Manager for the Swift Programming Language

3,595 commits 15 branches 791 releases **183 contributors** Apache

Branch: master New pull request Find file Clone or

vlm Merge pull request #1588 from vlm/deprecate-standalone-currentWorking... Latest commit 5fd01d

Documentation [Docs] Update stale mailing list url to Swift forums

Fixtures Removed code thanks to Equatable synthesis 2

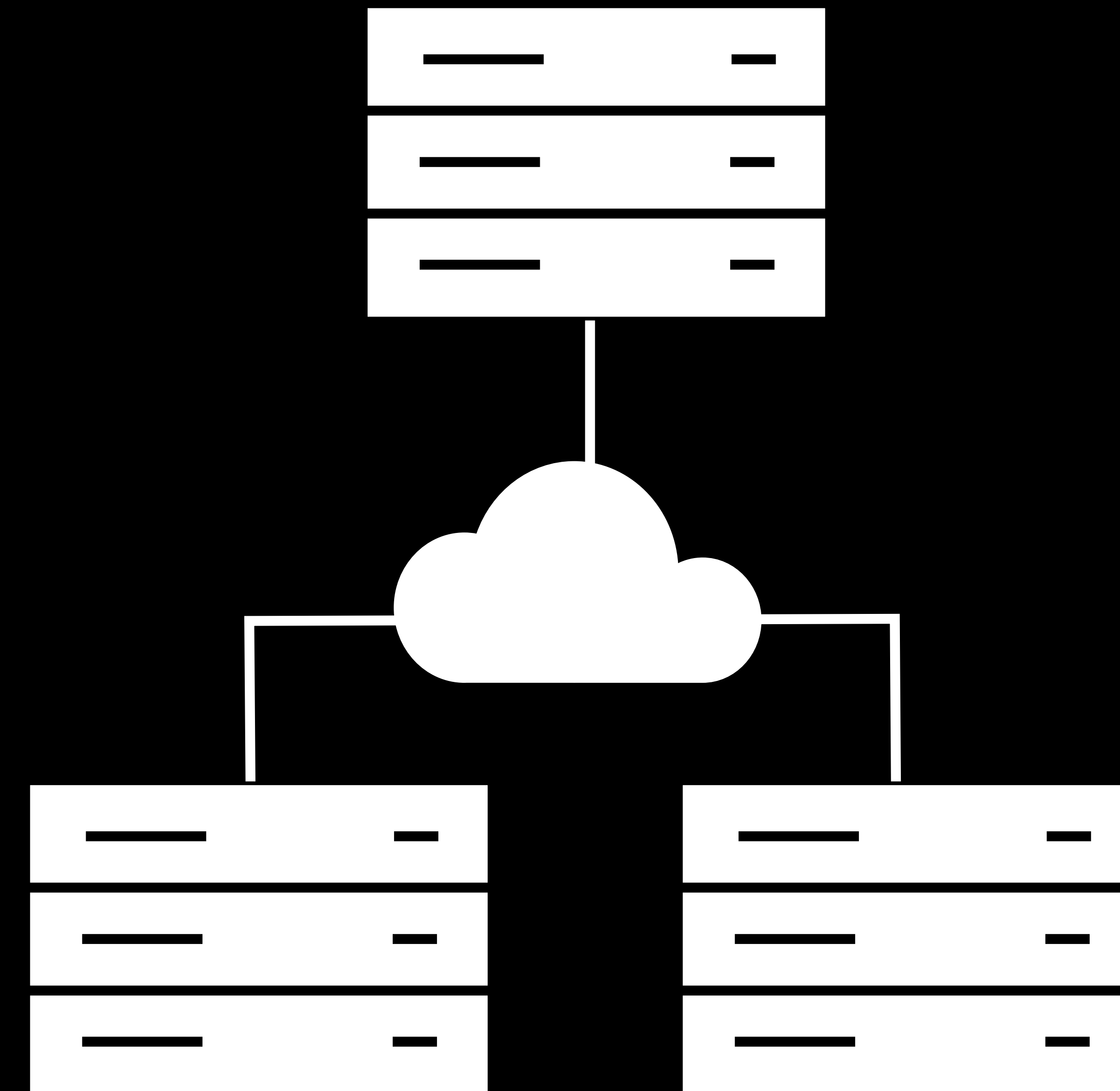
Sources Merge pull request #1588 from vlm/deprecate-standalone-currentWorking...

Server-Side Swift

Server-Side Swift



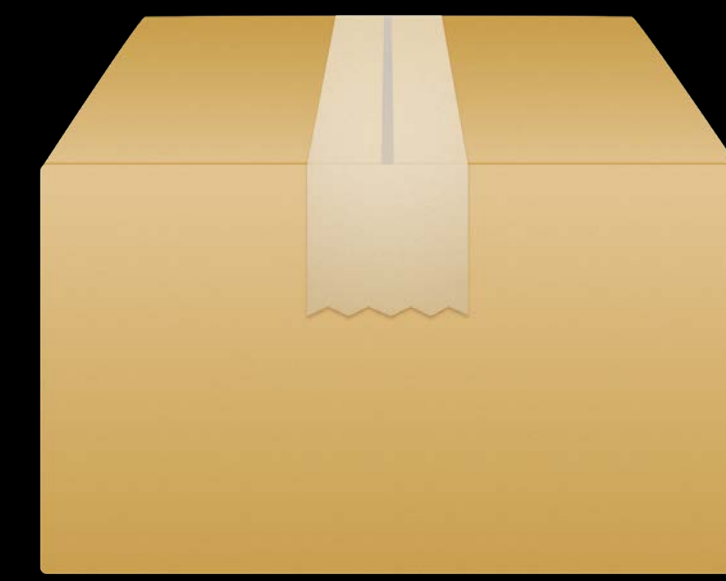
+



Command-Line Utilities



```
exec
```



\$ swift package init

More Information

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

Swift Open Hours

Technology Lab 10

Friday 3:00PM

 **WWDC18**