



# Первое смузи из Swift Package Registry

Евгений Рыжов  
Senior iOS Developer, Ozon



# Кому полезно

И каким образом

```
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (5m 24.15s)
swift package purge-cache; \
swift package reset; \
swift package resolve

Fetching ios_common/SnapKit.git
Fetching ios_common/KeychainAccess.git
Fetching ios_common/AppsFlyerFramework
Fetching sxm_ios/moya.git
Fetching ios_common/lottie-ios.git
Fetching ios_common/Alamofire.git
Fetched ios_common/SnapKit.git from cache (5.7
0s)
Fetching ios_common/needle.git
Fetched ios_common/needle.git from cache (126.
93s)
Fetched ios_common/AppsFlyerFramework from cac
he (313.69s)
Fetched ios_common/Alamofire.git from cache (3
13.69s)
Fetched ios_common/lottie-ios.git from cache (
313.69s)
Fetched sxm_ios/moya.git from cache (313.69s)
Fetched ios_common/KeychainAccess.git from cac
he (313.69s)
Computing version for ios_common/SnapKit.git
Computed ios_common/SnapKit.git at 5.0.1 (1.19
s)
Computing version for sxm_ios/moya.git
Computed sxm_ios/moya.git at 15.0.2 (0.49s)
Computing version for ios_common/lottie-ios.gi
t
Computed ios_common/lottie-ios.git at 4.4.2 (0
.55s)
```



# Кому полезно

И каким образом

```
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (38.449s)
swift package purge-cache; \
swift package reset; \
swift package resolve

Computing version for external-snapshot.snapkit
Computed external-snapshot.snapkit at 5.0.1 (0.73s)
Computing version for external-snapshot.moya
Computed external-snapshot.moya at 15.0.3 (0.46s)
Computing version for external-snapshot.lottie-ios
Computed external-snapshot.lottie-ios at 4.4.2 (0.56s)
Computing version for external-snapshot.appsflyerframework
Computed external-snapshot.appsflyerframework at 6.12.2 (0.52s)
Computing version for external-snapshot.alamofire
Computed external-snapshot.alamofire at 5.6.2 (0.53s)
Computing version for external-snapshot.keychainaccess
Computed external-snapshot.keychainaccess at 4.2.2 (0.55s)
Computing version for external-snapshot.needle
Computed external-snapshot.needle at 0.24.0 (0.56s)
Computing version for external-snapshot.rxswift
Computed external-snapshot.rxswift at 6.2.0 (0.53s)
Computing version for external-snapshot.reactiveswift
Computed external-snapshot.reactiveswift at 6.6.1 (0.45s)
Fetching external-snapshot.SnapKit
warning: 'external-snapshot.SnapKit': external-snapshot.SnapKit versi
on 5.0.1 source archive from
                is not signed
Fetched external-snapshot.SnapKit from cache (0.23s)
Fetching external-snapshot.KeychainAccess
warning: 'external-snapshot.KeychainAccess': external-snapshot.Keycha
inAccess version 4.2.2 source archive from
                is not signed
Fetched external-snapshot.KeychainAccess from cache (0.22s)
Fetching external-snapshot.reactiveswift
warning: 'external-snapshot.reactiveswift': external-snapshot.reactiv
```





# Кому полезно

И каким образом



## Account suspended

Access to your account has been suspended due to a violation of our [Terms of Service](#).

Please [contact support](#) for more information.

[Terms](#) [Privacy](#) [Security](#) [Contact GitHub](#)





# Кому полезно

И каким образом



# Обо мне

Живу в платформе три года без перерывов

В основном работаю с билд системой и менеджерами зависимостей

Не люблю менеджеры зависимостей



# AGENDA

**Swift Package и Registry**



Swift Registry Service и как с ней работать

В чём бенефиты и где нюансы

Заключение. Как оно?





01



Swift Package  
и Registry

# Что за Swift Package?

Все же уже знают, да?

- SPM — уже привычный для iOS разработчиков инструмент, которым можно и нужно пользоваться
- Минута молчания в честь падших в виде подов



# Что за Swift Package?

Все же уже знают, да?

swift-package-manager / Documentation / README.md 



 ashokm Fix broken link to 'Swift Package Registry Service Specification' (#7107 

ef871c2 · 9 months ago  History

Preview

Code

Blame

106 lines (62 loc) · 8.61 KB · 

Raw



## Swift Package Manager

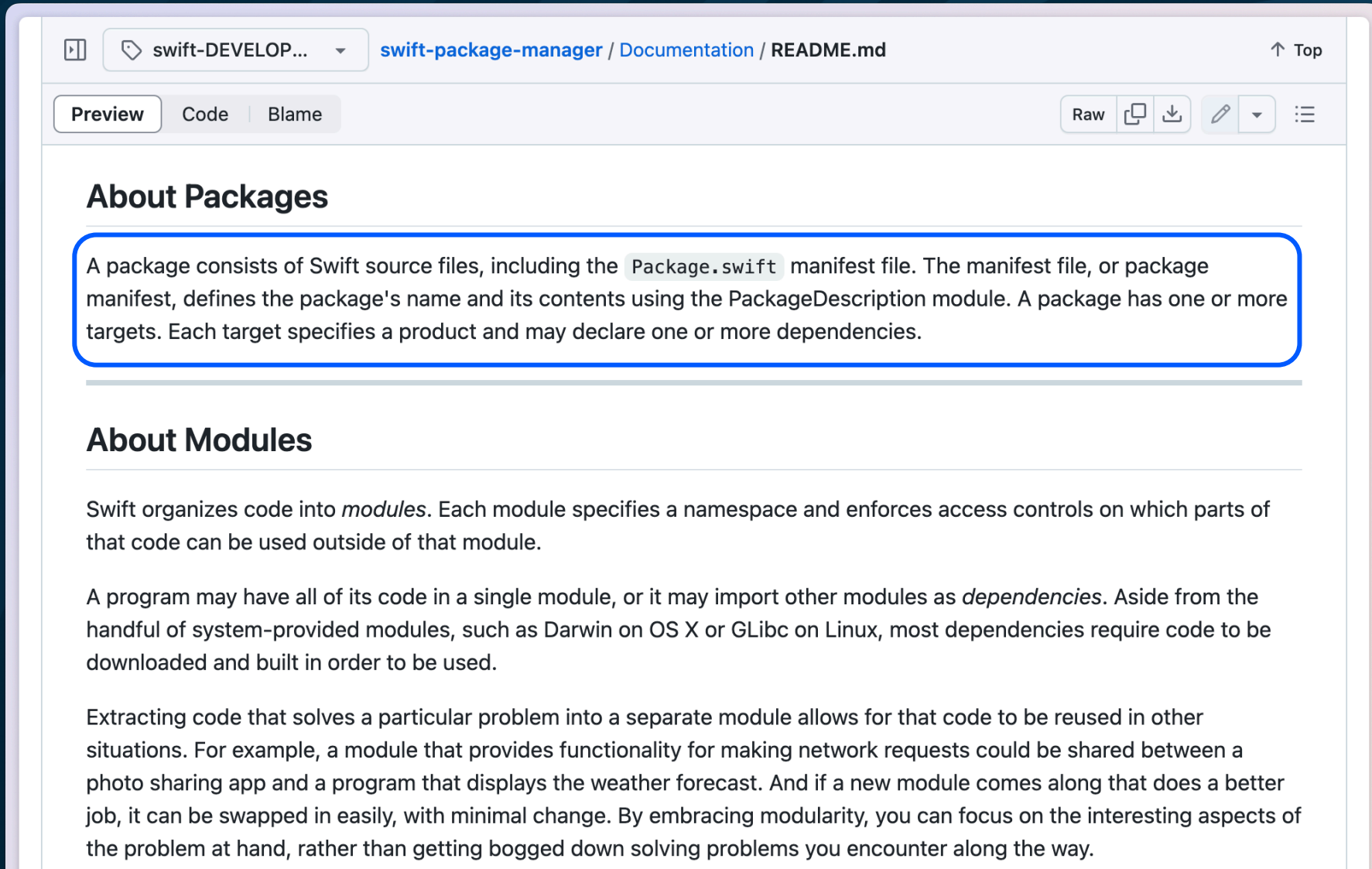
The Swift Package Manager is a tool for managing distribution of source code, aimed at making it easy to share your code and reuse others' code. The tool directly addresses the challenges of **compiling and linking Swift packages, managing dependencies,** versioning, and **supporting flexible distribution and collaboration** models.

We've designed the system to make it really easy to share packages on services like GitHub, but packages are also great for private personal development, sharing code within a team, or at any other granularity.



# Что за Swift Package?

Все же уже знают, да?



The screenshot shows a web browser window displaying the documentation for the Swift Package Manager. The browser's address bar shows the URL `swift-DEVELOP... / swift-package-manager / Documentation / README.md`. The page has a navigation bar with tabs for 'Preview', 'Code', and 'Blame', and a utility bar with buttons for 'Raw', 'Copy', 'Download', 'Edit', and a menu icon. The main content area features a section titled 'About Packages' which is highlighted with a blue border. Below it is a section titled 'About Modules'.

## About Packages

A package consists of Swift source files, including the `Package.swift` manifest file. The manifest file, or package manifest, defines the package's name and its contents using the `PackageDescription` module. A package has one or more targets. Each target specifies a product and may declare one or more dependencies.

## About Modules

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

A program may have all of its code in a single module, or it may import other modules as *dependencies*. Aside from the handful of system-provided modules, such as Darwin on OS X or Glibc on Linux, most dependencies require code to be downloaded and built in order to be used.

Extracting code that solves a particular problem into a separate module allows for that code to be reused in other situations. For example, a module that provides functionality for making network requests could be shared between a photo sharing app and a program that displays the weather forecast. And if a new module comes along that does a better job, it can be swapped in easily, with minimal change. By embracing modularity, you can focus on the interesting aspects of the problem at hand, rather than getting bogged down solving problems you encounter along the way.

# Что за Swift Package?

Какие-то зависимости и ресурсы. Фу

```
1 import PackageDescription
2
3 let package = Package(
4     name: "MyPackage",
5     dependencies: [
6         .package(
7             url: "https://github.com/apple/example-package-playingcard.git",
8             from: "0.1.0"
9         ),
10    ]
11 )
```

# За что Git?

И в чём его недостатки

```
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (5m 24.15s)
swift package purge-cache; \
swift package reset; \
swift package resolve

Fetching ios_common/SnapKit.git
Fetching ios_common/KeychainAccess.git
Fetching ios_common/AppsFlyerFramework
Fetching sxm_ios/moya.git
Fetching ios_common/lottie-ios.git
Fetching ios_common/Alamofire.git
Fetched ios_common/SnapKit.git from cache (5.7
0s)
Fetching ios_common/needle.git
Fetched ios_common/needle.git from cache (126.
93s)
Fetched ios_common/AppsFlyerFramework from cac
he (313.69s)
Fetched ios_common/Alamofire.git from cache (3
13.69s)
Fetched ios_common/lottie-ios.git from cache (
313.69s)
Fetched sxm_ios/moya.git from cache (313.69s)
Fetched ios_common/KeychainAccess.git from cac
he (313.69s)
Computing version for ios_common/SnapKit.git
Computed ios_common/SnapKit.git at 5.0.1 (1.19
s)
Computing version for sxm_ios/moya.git
Computed sxm_ios/moya.git at 15.0.2 (0.49s)
Computing version for ios_common/lottie-ios.gi
t
Computed ios_common/lottie-ios.git at 4.4.2 (0
.55s)
```





# За что Git?

И в чём его недостатки

```
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (5m 24.15s)
swift package purge-cache; \
swift package reset; \
swift package resolve

Fetching ios_common/SnapKit.git
Fetching ios_common/KeychainAccess.git
Fetching ios_common/AppsFlyerFramework
Fetching sxm_ios/moya.git
Fetching ios_common/lottie-ios.git
Fetching ios_common/Alamofire.git
Fetched ios_common/SnapKit.git from cache (5.7
0s)
Fetching ios_common/needle.git
Fetched ios_common/needle.git from cache (126.
93s)
Fetched ios_common/AppsFlyerFramework from cac
he (313.69s)
Fetched ios_common/Alamofire.git from cache (3
13.69s)
Fetched ios_common/lottie-ios.git from cache (
313.69s)
Fetched sxm_ios/moya.git from cache (313.69s)
Fetched ios_common/KeychainAccess.git from cac
he (313.69s)
Computing version for ios_common/SnapKit.git
Computed ios_common/SnapKit.git at 5.0.1 (1.19
s)
Computing version for sxm_ios/moya.git
Computed sxm_ios/moya.git at 15.0.2 (0.49s)
Computing version for ios_common/lottie-ios.gi
t
Computed ios_common/lottie-ios.git at 4.4.2 (0
.55s)
```



# За что Git?

И в чём его недостатки

```
1 .package(  
2   url: "https://github.com/Moya/Moya.git",  
3   exact: "15.0.3"  
4 ),  
5 // ... depends on  
6 .package(  
7   url: "https://github.com/Alamofire/Alamofire.git",  
8   exact: "5.6.2"  
9 ),
```



# За что Git?

И в чём его недостатки

```
1 .package(  
2     url: "https://github.com/Moya/Moya.git",  
3     exact: "15.0.3"  
4 ),  
5 // ... depends on  
6 .package(  
7     url: "https://github.com/Alamofire/Alamofire.git",  
8     exact: "5.6.2"  
9 ),
```

```
1 public struct GitRepositoryProvider: RepositoryProvider {  
2  
3     // ...  
4  
5     public func fetch(  
6         repository: RepositorySpecifier,  
7         to path: Basics.AbsolutePath,  
8         progressHandler: FetchProgress.Handler? = nil  
9     ) throws {  
10        try self.clone(  
11            repository,  
12            repository.location.gitURL,  
13            path.pathString,  
14            ["--mirror"],  
15            progress: progressHandler  
16        )  
17    }  
18 }
```



# За что Git?

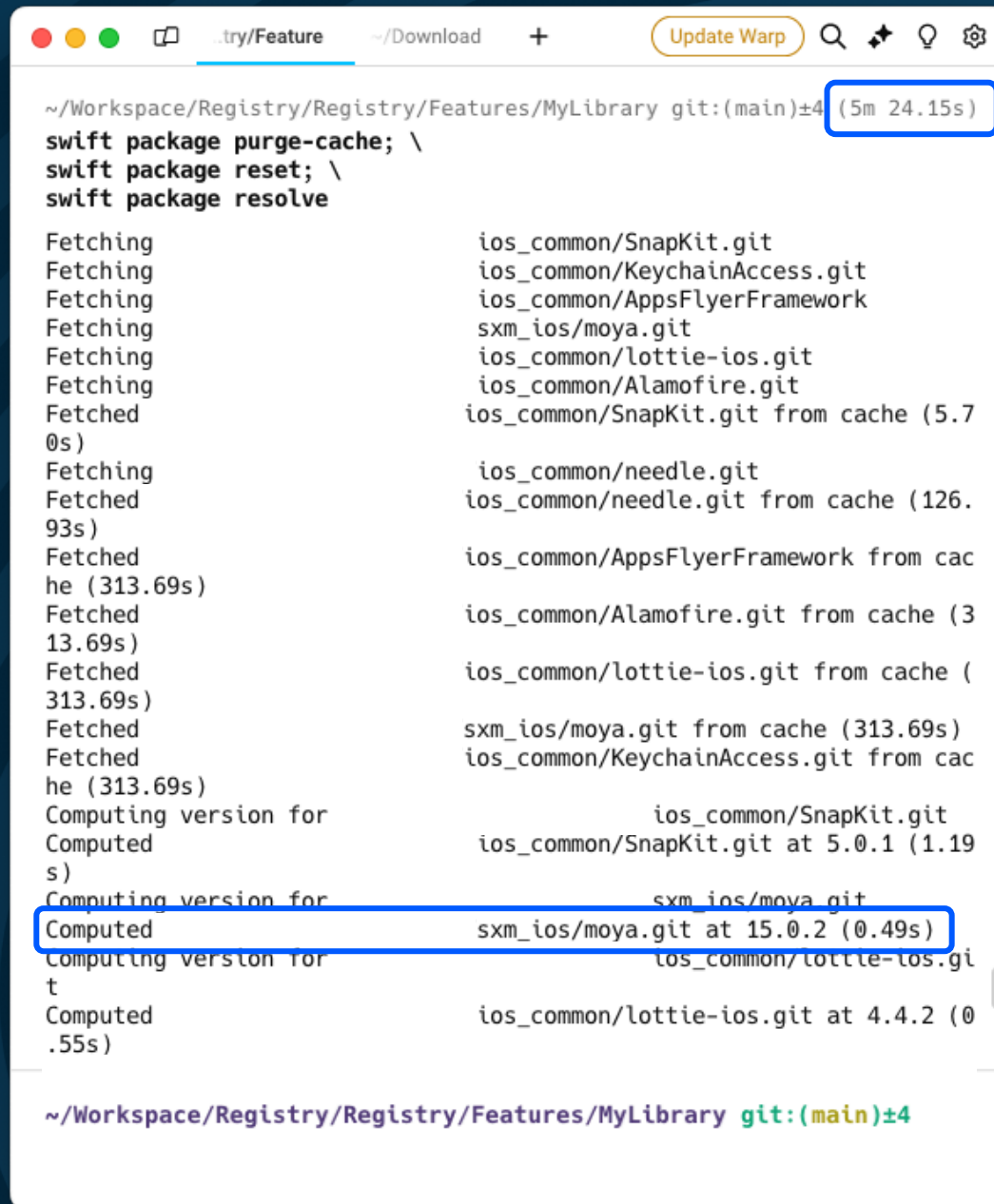
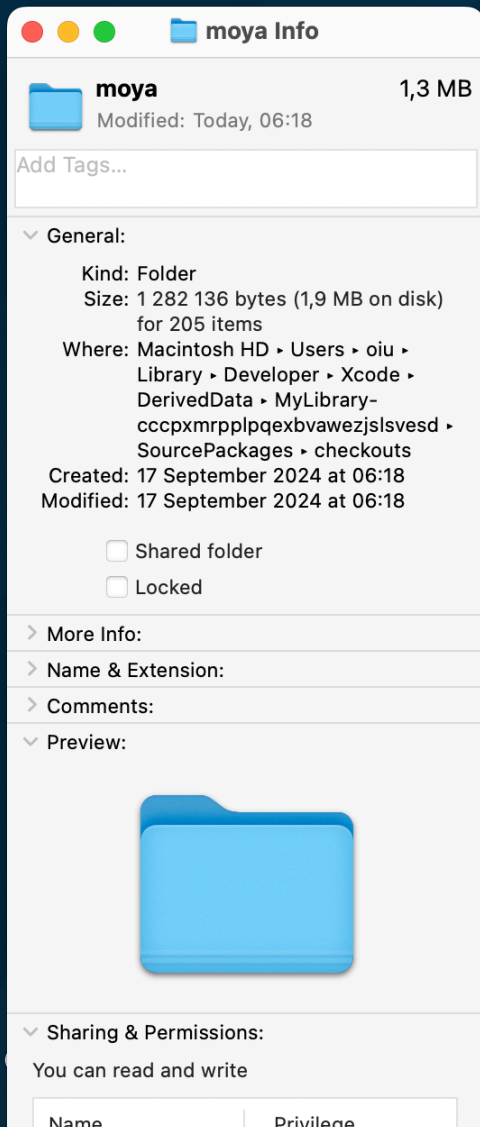
И в чём его недостатки

```
1 .package(  
2     url: "https://github.com/Moya/Moya.git",  
3     exact: "15.0.3"  
4 ),  
5 // ... depends on  
6 .package(  
7     url: "https://github.com/Alamofire/Alamofire.git",  
8     exact: "5.6.2"  
9 ),
```

```
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (5m 24.15s)  
swift package purge-cache; \  
swift package reset; \  
swift package resolve  
  
Fetching ios_common/SnapKit.git  
Fetching ios_common/KeychainAccess.git  
Fetching ios_common/AppsFlyerFramework  
Fetching sxm_ios/moya.git  
Fetching ios_common/lottie-ios.git  
Fetching ios_common/Alamofire.git  
Fetched ios_common/SnapKit.git from cache (5.7  
0s)  
Fetching ios_common/needle.git  
Fetched ios_common/needle.git from cache (126.  
93s)  
Fetched ios_common/AppsFlyerFramework from cac  
he (313.69s)  
Fetched ios_common/Alamofire.git from cache (3  
13.69s)  
Fetched ios_common/lottie-ios.git from cache (  
313.69s)  
Fetched sxm_ios/moya.git from cache (313.69s)  
Fetched ios_common/KeychainAccess.git from cac  
he (313.69s)  
Computing version for ios_common/SnapKit.git  
Computed ios_common/SnapKit.git at 5.0.1 (1.19  
s)  
Computing version for sxm_ios/moya git  
Computed sxm_ios/moya.git at 15.0.2 (0.49s)  
Computing version for ios_common/lottie-ios.gi  
t  
Computed ios_common/lottie-ios.git at 4.4.2 (0  
.55s)  
  
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4
```

# За что Git?

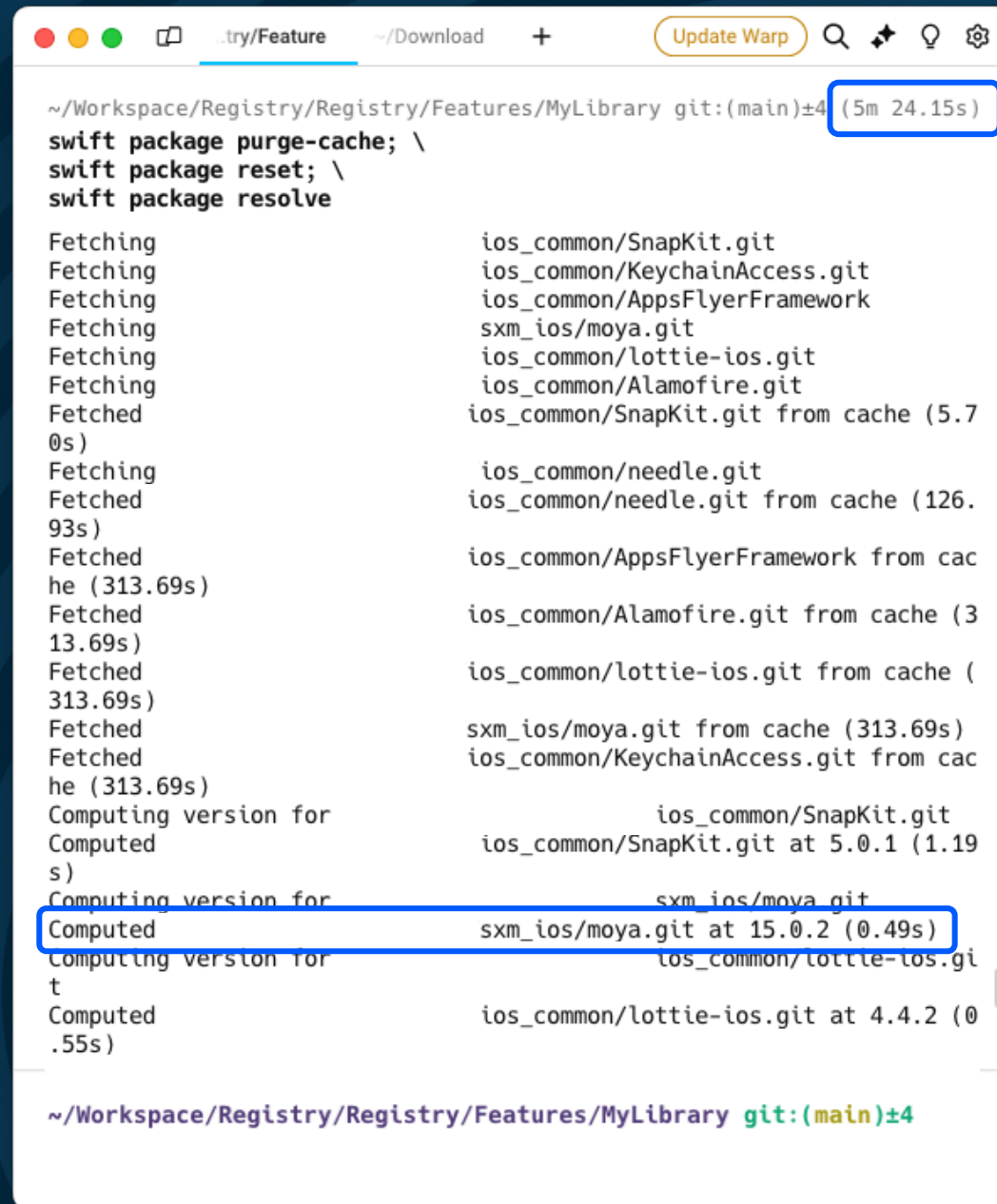
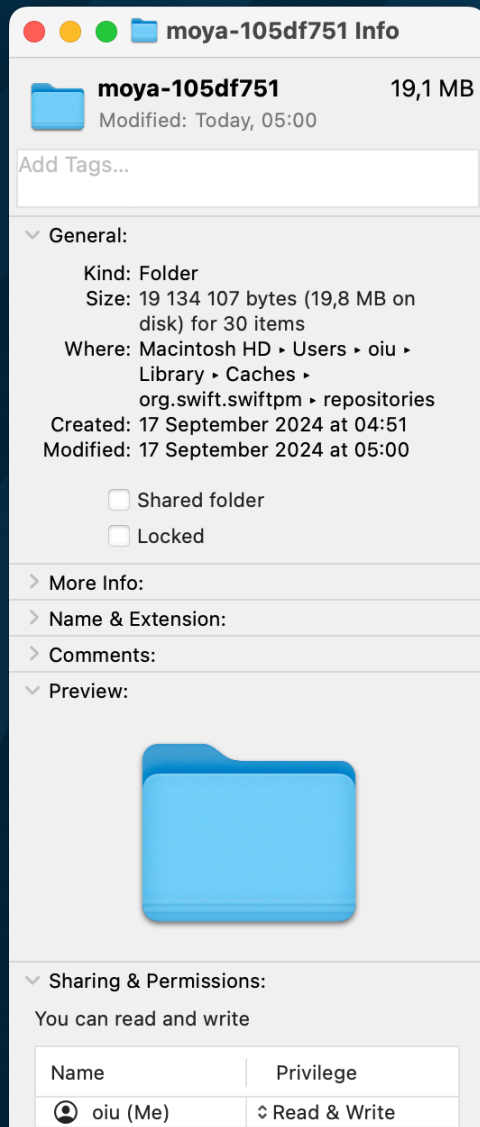
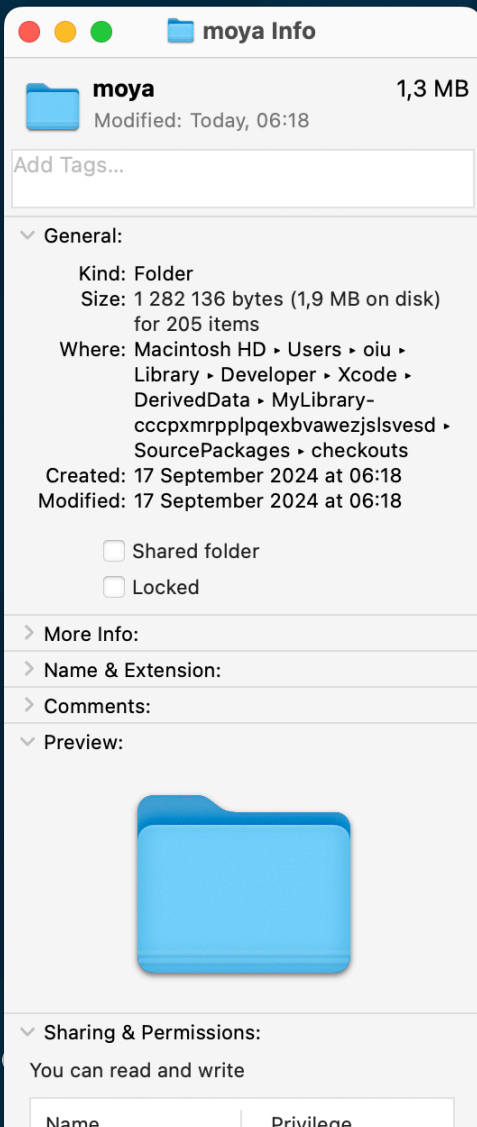
И в чём его недостатки





# За что Git?

И в чём его недостатки



# За что Git?

И в чём его недостатки

**moya Info**

**moya** 1,3 MB  
Modified: Today, 06:18

Add Tags...

General:

- Kind: Folder
- Size: 1 282 136 bytes (1,9 MB on disk) for 205 items
- Where: Macintosh HD > Users > oiu > Library > Developer > Xcode > DerivedData > MyLibrary-cccpxmrrpplpqexbvawezjslsvsd > SourcePackages > checkouts
- Created: 17 September 2024 at 06:18
- Modified: 17 September 2024 at 06:18


Shared folder:  Shared folder,  Locked

More Info:

Name & Extension:

Comments:

Preview:



Sharing & Permissions:

You can read and write

Name	Privilege
oiu (Me)	Read & Write

**moya-105df751 Info**

**moya-105df751** 19,1 MB  
Modified: Today, 05:00

Add Tags...

General:

- Kind: Folder
- Size: 19 134 107 bytes (19,8 MB on disk) for 30 items
- Where: Macintosh HD > Users > oiu > Library > Caches > org.swift.swiftpm > repositories
- Created: 17 September 2024 at 04:51
- Modified: 17 September 2024 at 05:00


Shared folder:  Shared folder,  Locked

More Info:

Name & Extension:

Comments:

Preview:



Sharing & Permissions:

You can read and write

Name	Privilege
oiu (Me)	Read & Write

**moya**

Alamofire, AppsFlyerFramework, KeychainAccess, lottie-ios, **moya**, needle, SnapKit

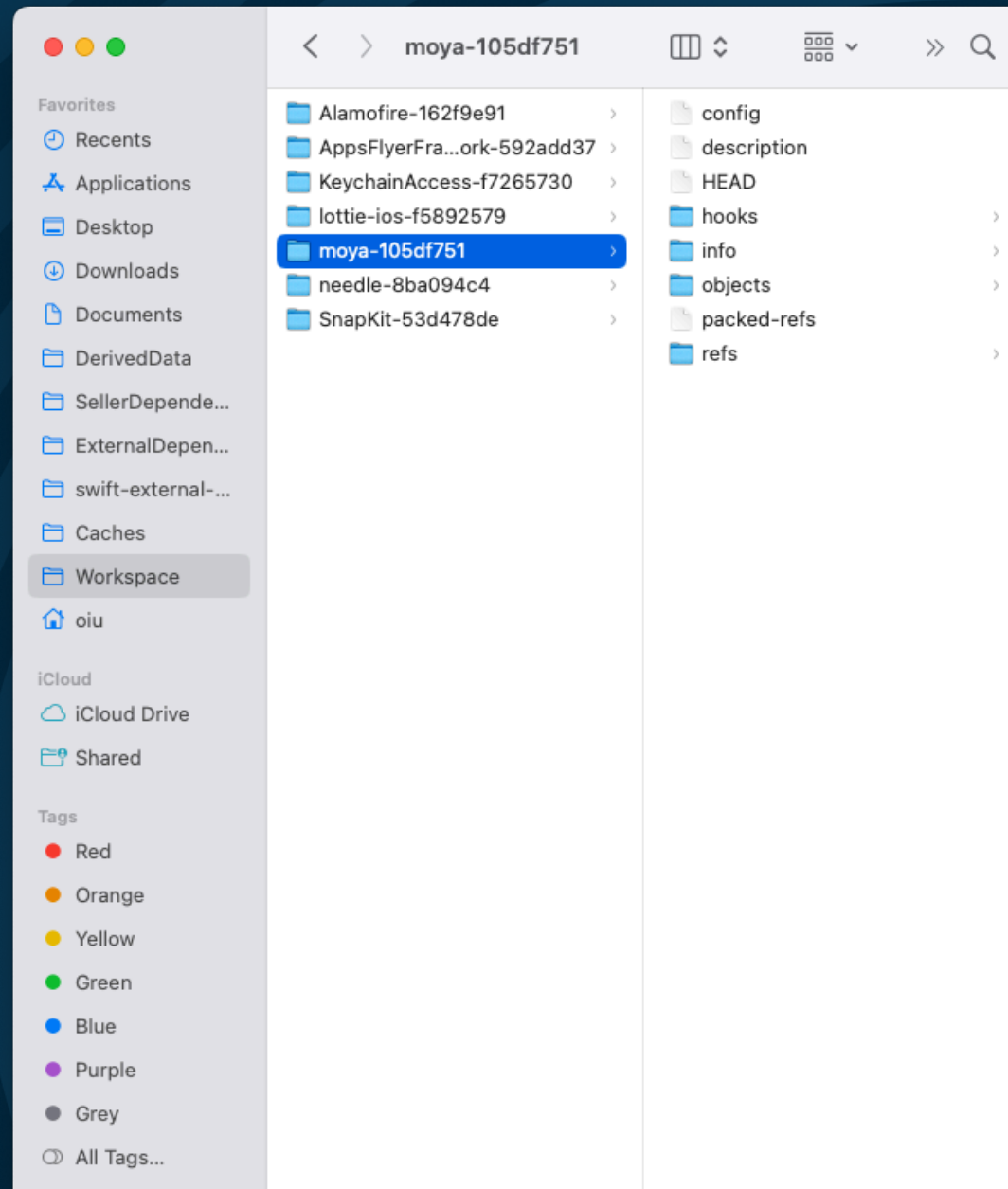
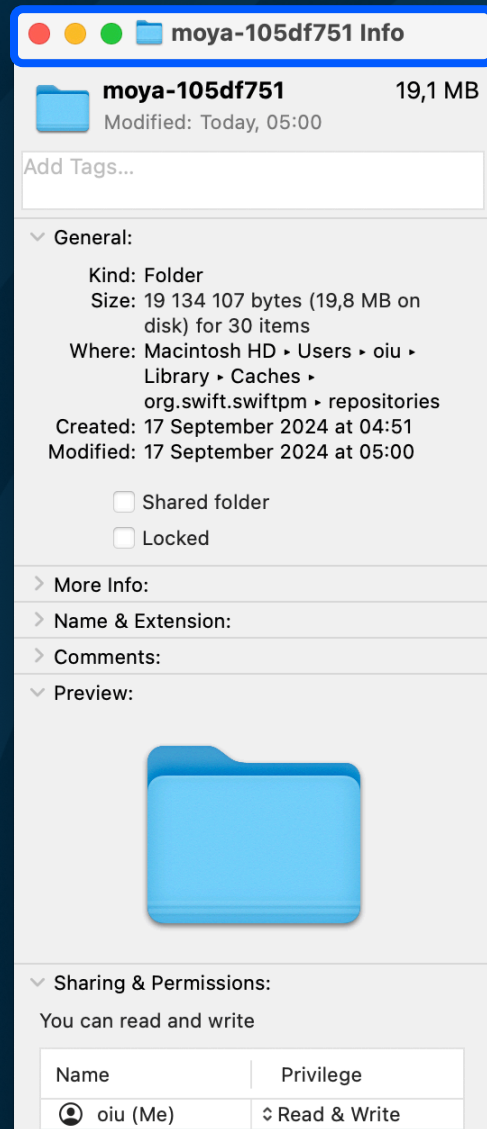
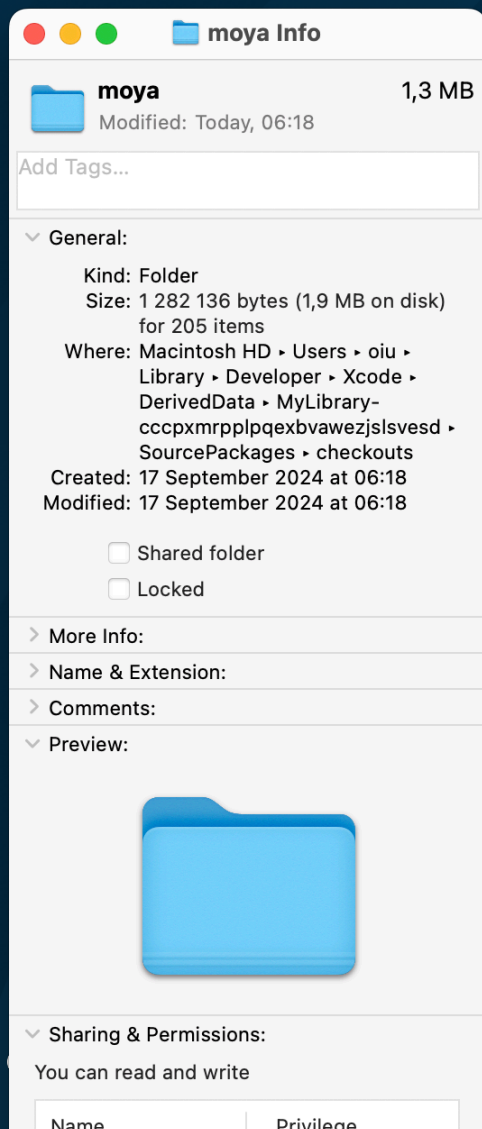
.circleci, .git, .gitattributes, .github, .gitignore, .swiftlint.yml, Cartfile, Cartfile.private, Cartfile.resolved, Changelog.md, Code of Conduct\_CN.md, Code of Conduct.md, codecov.yml, Contributing.md, Dangerfile.swift, docs, docs\_CN, Examples, Gemfile, Gemfile.lock, License.md, Moya.podspec, Moya.xcodeproj, Package.resolved, **Package.swift**, Rakefile, Readme\_CN.md, **Readme.md**, scripts, **Sources**, Tests, Vision\_CN.md

Tags: Red, Orange, Yellow, Green, Blue, Purple, Grey, All Tags...



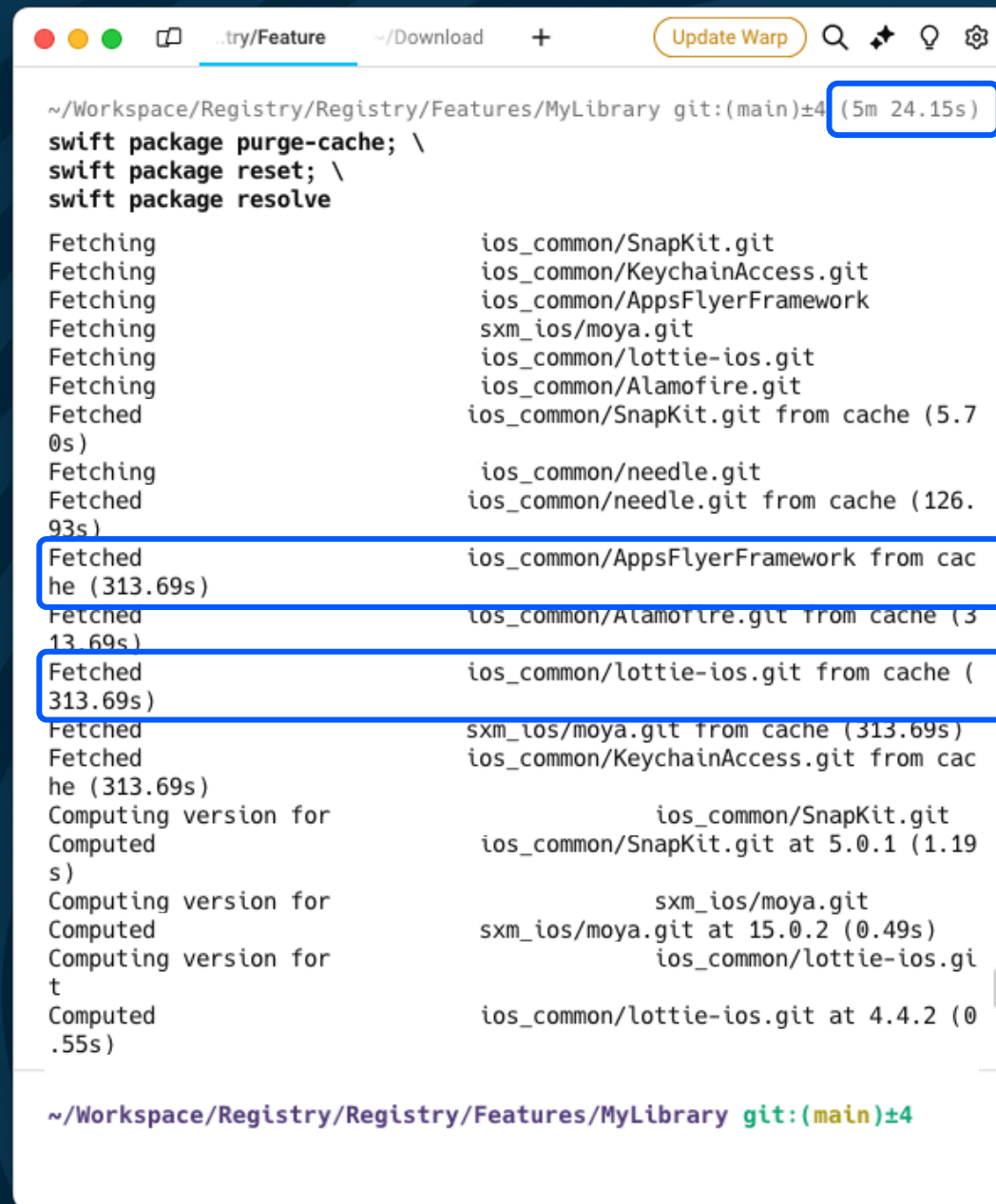
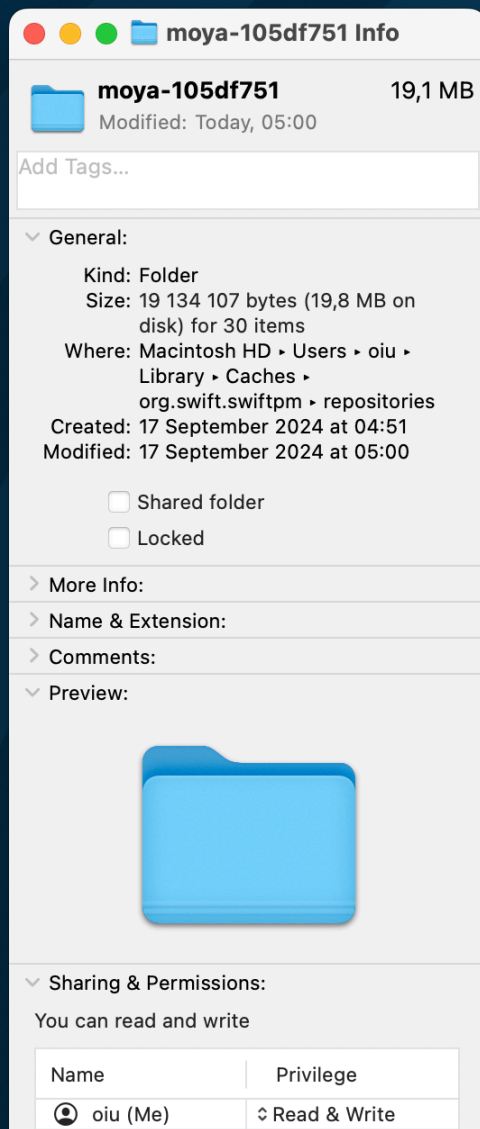
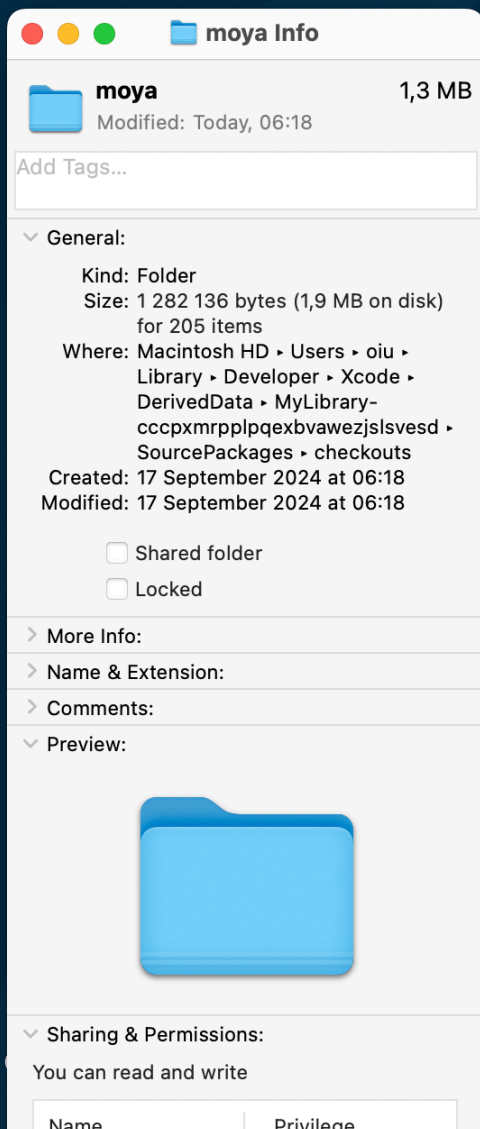
# За что Git?

И в чём его недостатки



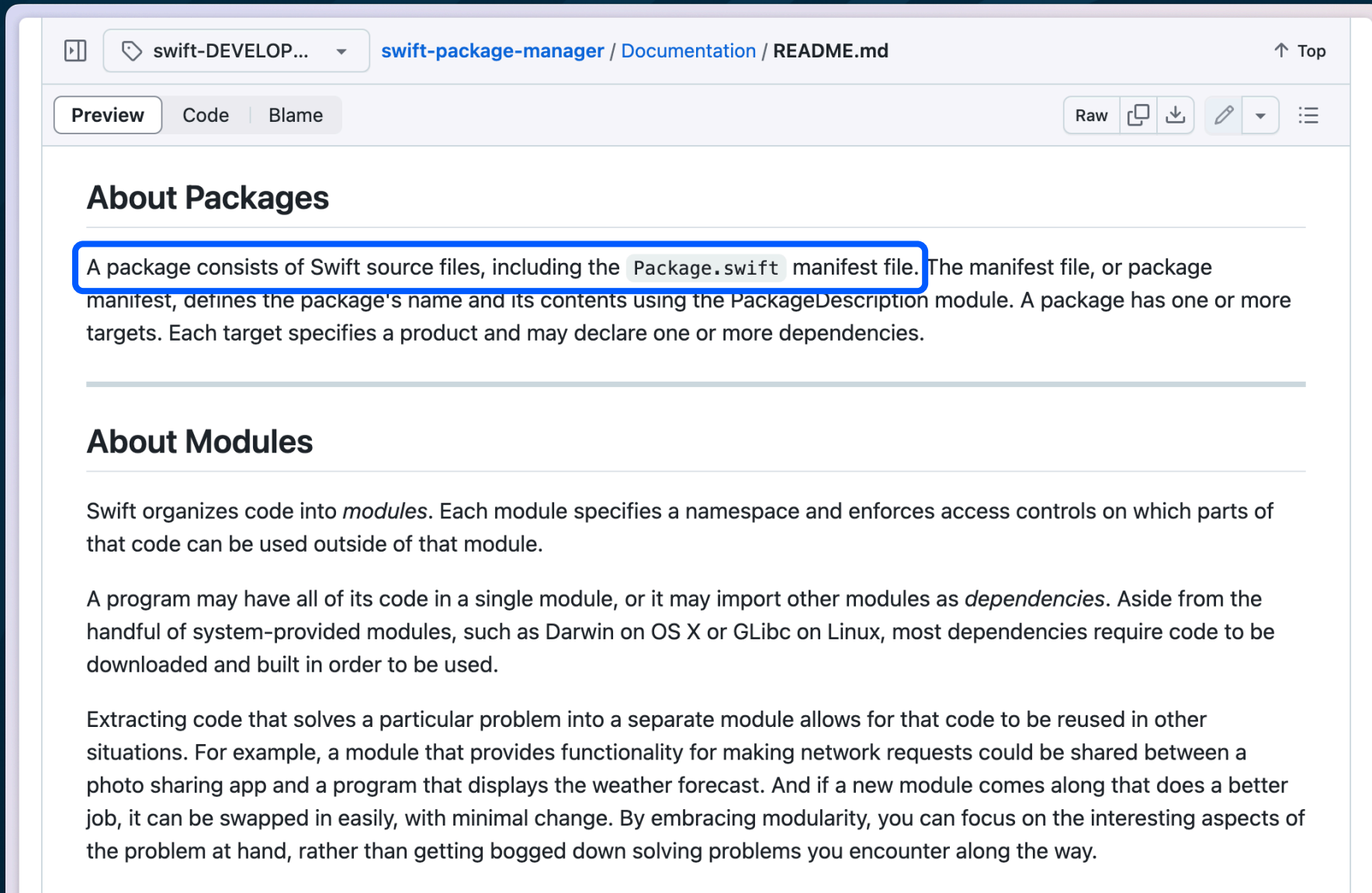
# За что Git?

И в чём его недостатки



# Откуда Git?

И за что



swift-DEVELOP... swift-package-manager / Documentation / README.md

Preview Code Blame Raw Copy Download Edit

## About Packages

A package consists of Swift source files, including the `Package.swift` manifest file. The manifest file, or package manifest, defines the package's name and its contents using the `PackageDescription` module. A package has one or more targets. Each target specifies a product and may declare one or more dependencies.

## About Modules

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

A program may have all of its code in a single module, or it may import other modules as *dependencies*. Aside from the handful of system-provided modules, such as Darwin on OS X or Glibc on Linux, most dependencies require code to be downloaded and built in order to be used.

Extracting code that solves a particular problem into a separate module allows for that code to be reused in other situations. For example, a module that provides functionality for making network requests could be shared between a photo sharing app and a program that displays the weather forecast. And if a new module comes along that does a better job, it can be swapped in easily, with minimal change. By embracing modularity, you can focus on the interesting aspects of the problem at hand, rather than getting bogged down solving problems you encounter along the way.



# Откуда Git?

И за что

0.1.0 swift-package-manager / Documentation / Usage.md ↑ Top

Preview Code Blame Raw Copy Download Edit

## Create a Package

Simply put: a package is a git repository with semantically versioned tags, that contains Swift sources and a `Package.swift` manifest file at its root.

### Create a library package

A library package contains code which other packages can use and depend on. To get started, create a directory and run `swift package init` command:

```
$ mkdir MyPackage
$ cd MyPackage
$ swift package init # or swift package init --type library
$ swift build
$ swift test
```

This will create the directory structure needed for a library package with a target and the corresponding test target to write unit tests. A library package can contain multiple targets as explained in [Target Format Reference](#).

### Create an executable package

SwiftPM can create native binary which can be executed from command line. To get started:



# Как избавиться от Git?

Что за Registry Service?

Preview Code Blame

Raw Copy Download Edit

## Package Registry Service

- Proposal: [SE-0292](#)
- Authors: [Bryan Clark](#), [Whitney Imura](#), [Mattt Zmuda](#)
- Review Manager: [Tom Doron](#)
- Status: **Implemented (Swift 5.7)**
- Implementation: [apple/swift-package-manager#3023](#)
- Decision Notes: [Rationale](#)
- Review: [1](#) [2](#) [3](#) [Amendment](#)
- Previous Revision: [1](#) [2](#) [3](#)

### Introduction

Swift Package Manager downloads dependencies using Git. Our proposal defines a standard web service interface that it can also use to download dependencies from a package registry.

Swift-evolution thread: [Swift Package Registry Service](#)

### Motivation

A package dependency is currently specified by a URL to its source repository. When Swift Package Manager builds a project for the first time, it clones the Git repository for each dependency and attempts to resolve the version requirements

AppsFlyerFramework-59...

**AppsFlyerFramework...** 659,8 MB  
Modified: Today, 04:41

Add tags...

General:

Kind: Folder  
Size: 659 779 826 bytes (670,9 MB on disk) for 29 items  
Where: Macintosh HD • Users • iou • Library • Caches • org.swift.swiftpm • repositories  
Created: 17 September 2024 at 04:36  
Modified: 17 September 2024 at 04:41


Shared folder  
 Locked

More Info:

Name & Extension:

Comments:

Preview:



Sharing & Permissions:

You can read and write

Name	Privilege
iou (Me)	Read & Write
staff	Read only
everyone	Read only

02



# Swift Registry Service

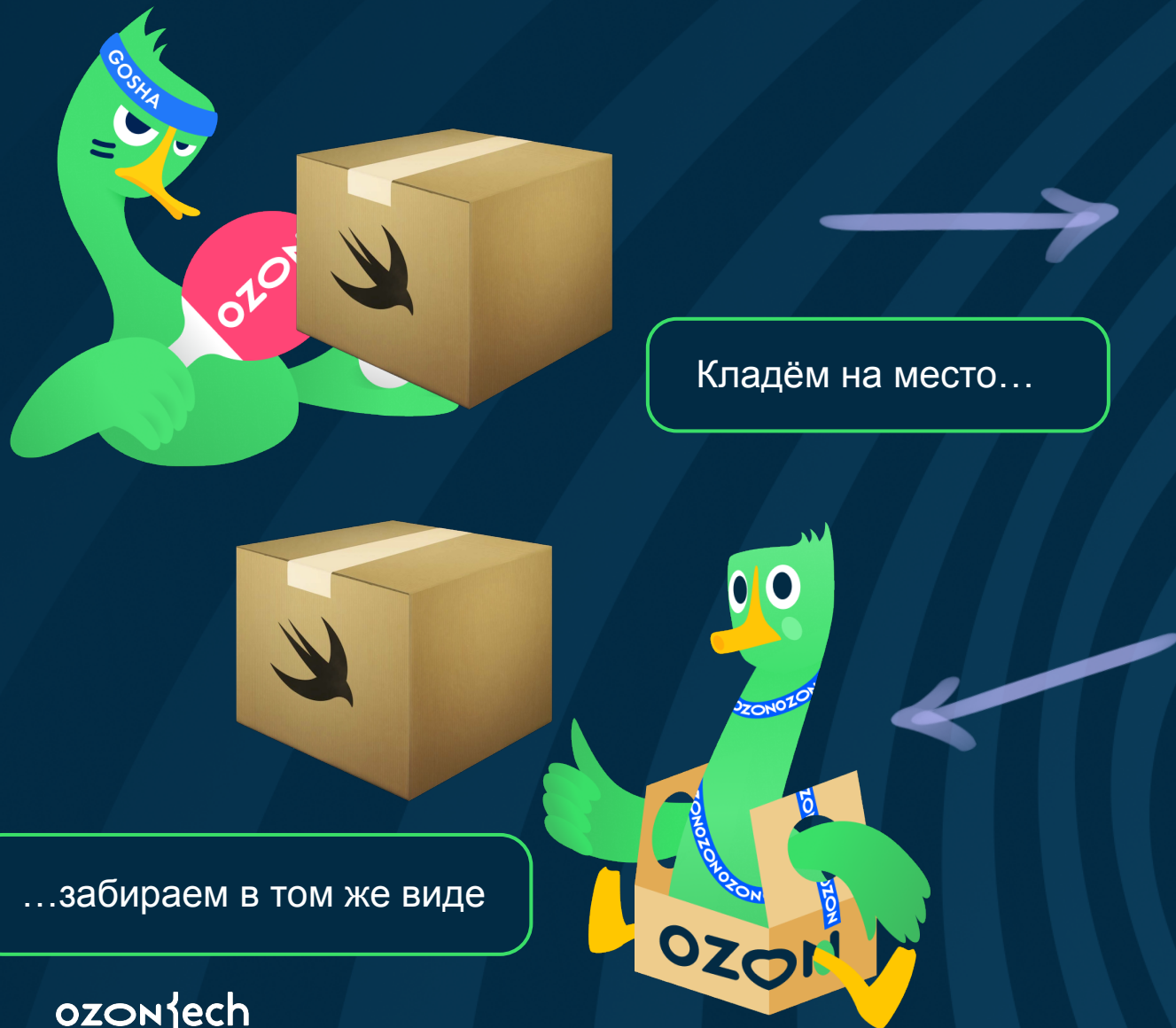
И как с ней работать





# Что за Registry

В общем понимании



ozon{tech





# Что за Registry

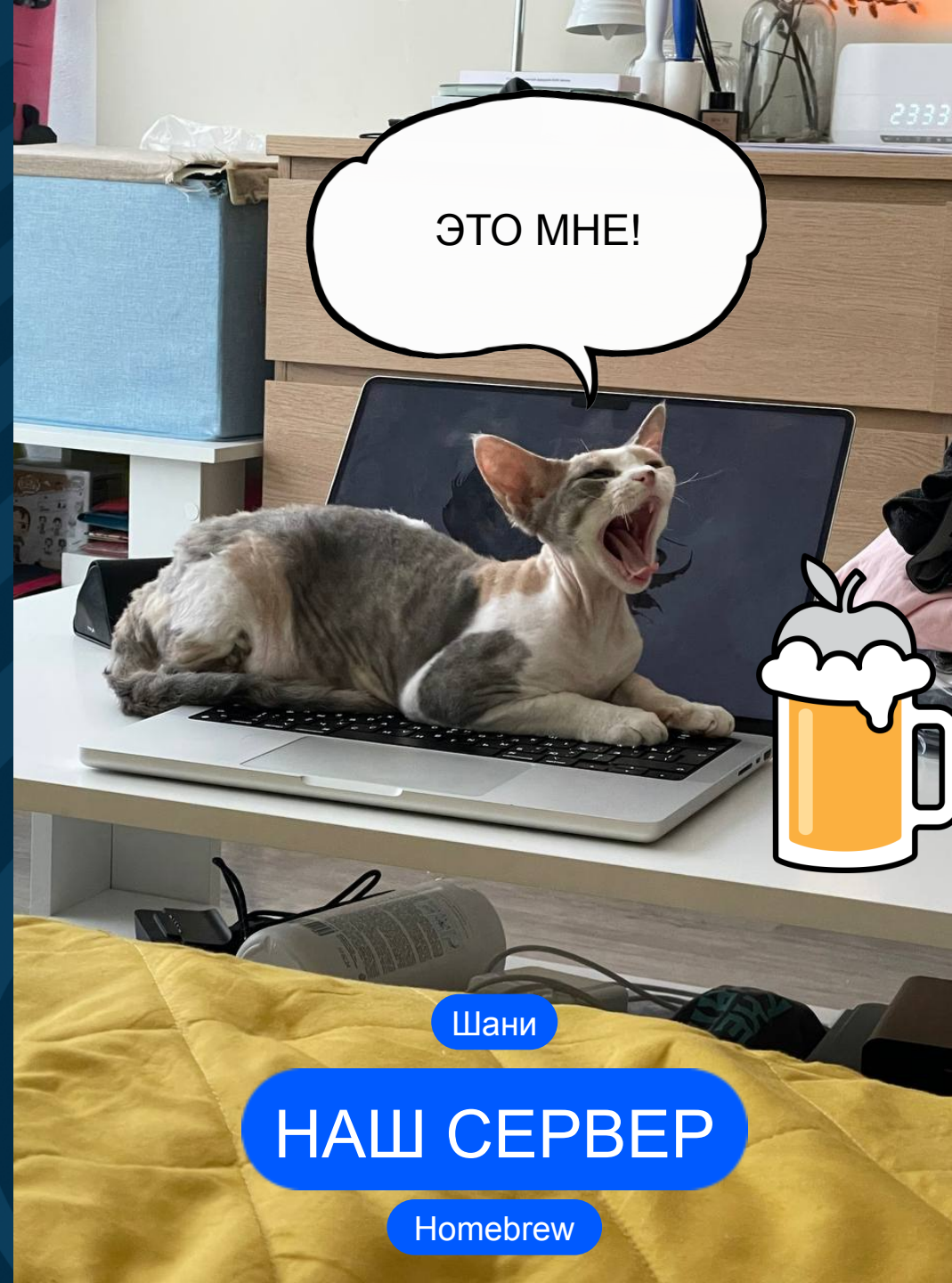
В общем понимании



Кладём на место...



```
1 $ brew create foo.com/foo-1.0.tgz
2 Created /opt/homebrew/Library/Taps/homebrew/homebrew-
  core/Formula/foo.rb
```



Шани

НАШ СЕРВЕР

Homebrew

# Что за Registry

В общем понимании



```
1 $ brew install foo
```



...забираем в том же виде

ozon{tech

ЛАДНО, ЗАБИРАЙ



Шани

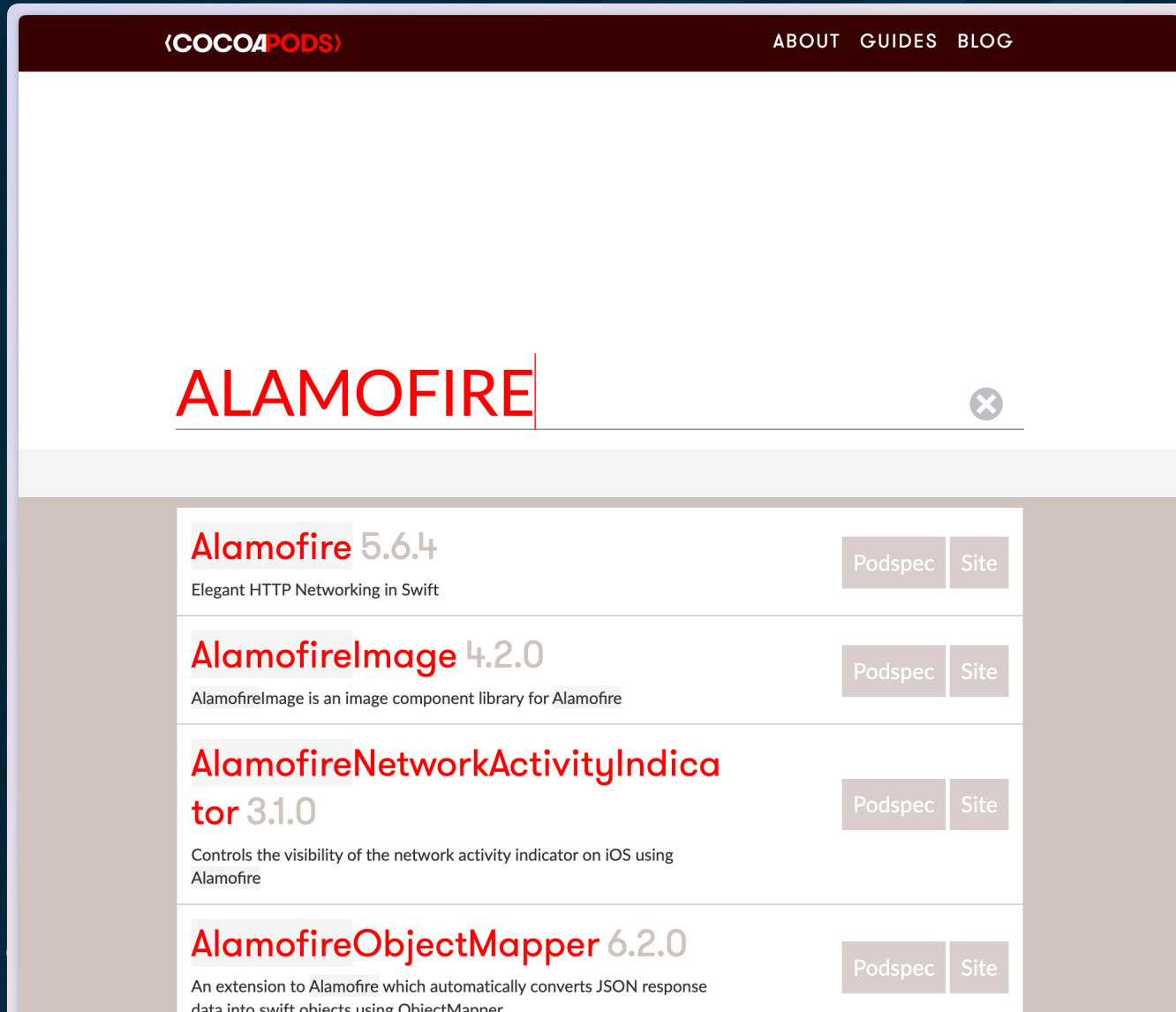
НАШ СЕРВЕР

Homebrew



# Что за Registry

В понимании iOS



The screenshot shows the CocoaPods website interface. At the top, there is a navigation bar with the CocoaPods logo and links for 'ABOUT', 'GUIDES', and 'BLOG'. Below the navigation bar, a search bar contains the text 'ALAMOFIRE'. The search results are displayed as a list of four items, each with a title, version number, description, and links for 'Podspec' and 'Site'.

Package Name	Version	Description	Podspec	Site
Alamofire	5.6.4	Elegant HTTP Networking in Swift	<a href="#">Podspec</a>	<a href="#">Site</a>
AlamofireImage	4.2.0	AlamofireImage is an image component library for Alamofire	<a href="#">Podspec</a>	<a href="#">Site</a>
AlamofireNetworkActivityIndicator	3.1.0	Controls the visibility of the network activity indicator on iOS using Alamofire	<a href="#">Podspec</a>	<a href="#">Site</a>
AlamofireObjectMapper	6.2.0	An extension to Alamofire which automatically converts JSON response data into swift objects using ObjectMapper	<a href="#">Podspec</a>	<a href="#">Site</a>



```
1 platform :ios, '8.0'  
2 use_frameworks!  
3  
4 target 'MyApp' do  
5   pod 'Alamofire', '~> 2.6'  
6 end
```



# Как он получился?

И почему вы о нём не слышите

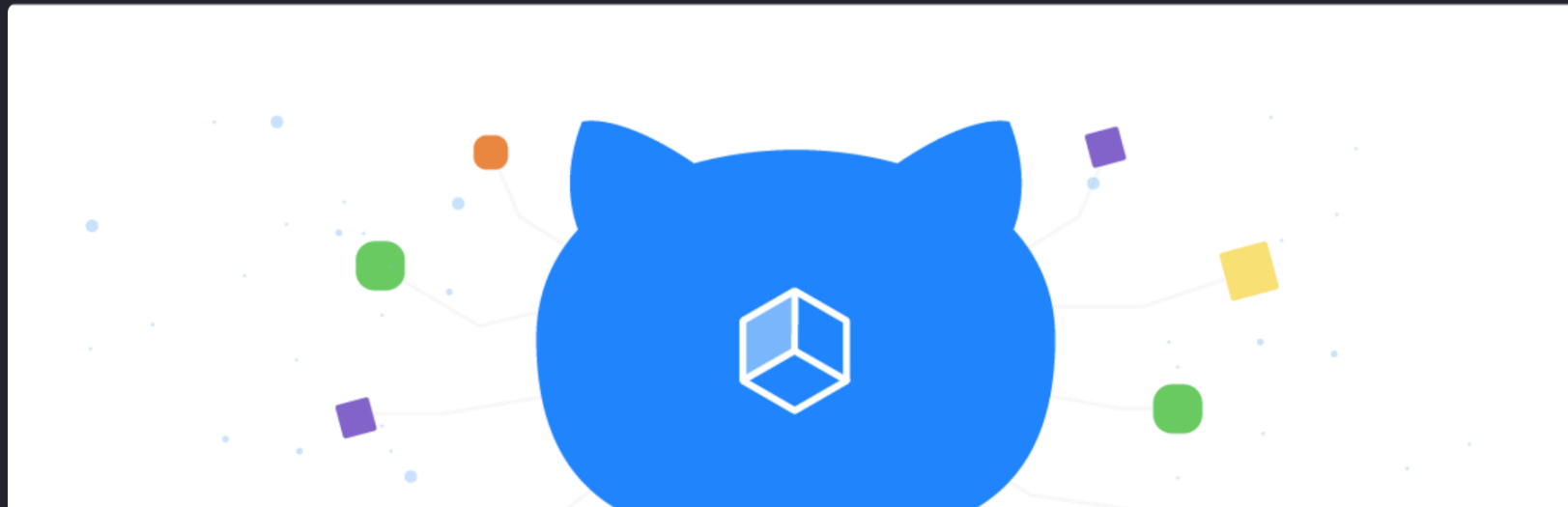
 / Blog



[Home](#) / [News & insights](#) / [Product](#)

## Introducing GitHub Package Registry

With GitHub Package Registry your packages are at home with their code—sign up for the limited beta to try it out.



# Как он получился?

И почему вы о нём не слышите

The screenshot shows a forum post on the Swift website. The post is titled "GitHub / Swift Package Management Service" and is categorized under "Development" and "Package Manager". The author is Bryan Clark, who is highlighted with a blue box around his profile picture and name. His bio states: "I'm a Director of Product at GitHub. I work on the Package Registry and Open Source." The post content discusses the development of a Swift Package Management Service, mentioning that they plan to open source their implementation (written in Go) and that they've learned from Rust, Go, and others. The post is dated Nov 5, 2019, and has 16 replies.

**Swift**

Topics

- swift.org
- My Posts
- More
- Categories
  - Announcements
  - Evolution
  - Development
  - Server
  - Using Swift
  - Related Projects
  - Community Showc...
- All categories
- Tags
  - concurrency
  - foundation

## GitHub / Swift Package Management Service

Development Package Manager `packagemanager`

Nov 2019 Nov 2019

**clarkbw** Bryan Clark 5 posts in topic 16 v 2019

I'm a Director of Product at GitHub. I work on the Package Registry and Open Source.

[twitter.com/clarkbw](https://twitter.com/clarkbw) Victoria, BC Canada

Posted Nov 5, 2019 Joined Oct 30, 2019 Read 1h

and other elements required or in addition to the basics. We plan to open source our implementation (written in Go) such that others can contribute back to it and progress can be made in the open.

We've learned from Rust, Go, and many others when thinking about package management services; and we intend to take the best of those worlds into this process.

Nov 2019

What are your thoughts here?

# Как он получился?

И почему вы о нём не слышите

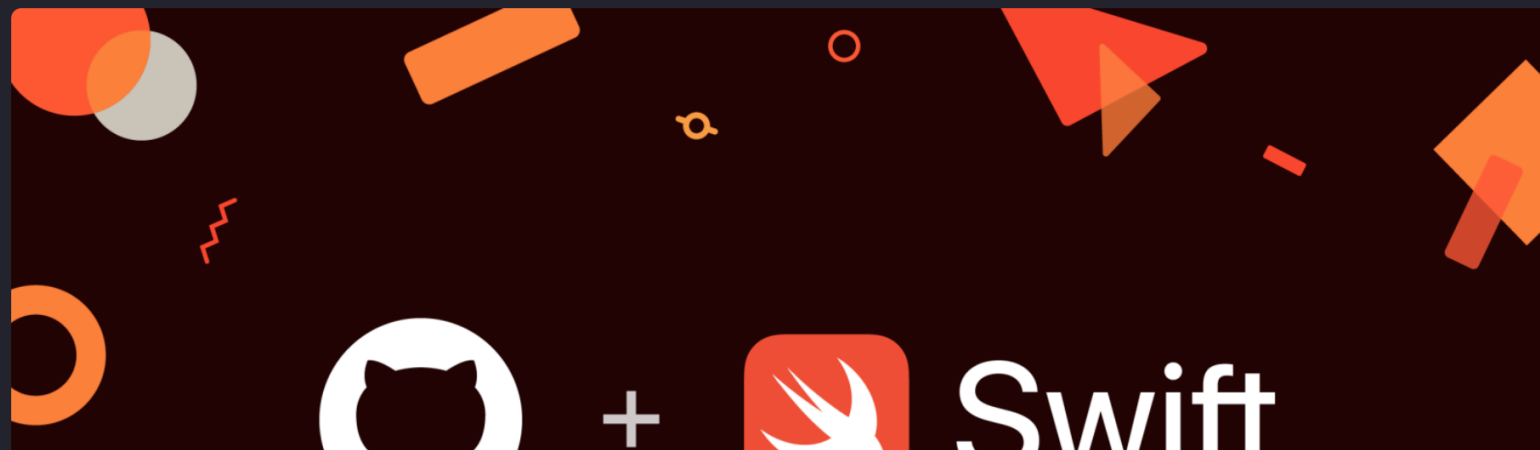
 / Blog



Home / News & insights / Product

## GitHub Package Registry will support Swift packages

Today we're excited to announce that we'll be adding support for Swift packages to GitHub Package Registry. Swift packages make it easy to share your libraries and source code across your projects and with the Swift community.





# Как он получился?

И почему вы о нём не слышите

The screenshot shows the Swift website interface. At the top left is the Swift logo and a menu icon. To the right are 'Sign Up' and 'Log In' buttons, and a search icon. A left sidebar contains navigation options: Topics (swift.org, More), Categories (Announcements, Evolution, Development, Server, Using Swift, Related Projects, Community Showc...), All categories, and Tags (concurrency, foundation, linux, packagemanager). The main content area features the title 'Swift Package Registry Service' with tags for Evolution, Pitches, and packagemanager. A post by user 'mattt' from June 2020 is displayed, with a progress indicator showing 1/94 posts from June 2020 to Dec 2022. The post text discusses the announcement by Apple and GitHub regarding Swift package registry support, mentions Bryan Clark's thread, and details the author's work on defining a draft specification for the service.

Swift

Sign Up Log In

Topics

swift.org

More

Categories

Announcements

Evolution

Development

Server

Using Swift

Related Projects

Community Showc...

All categories

Tags

concurrency

foundation

linux

packagemanager

## Swift Package Registry Service

Evolution Pitches packagemanager

**M** mattt Jun 2020

**Hello!**

Last June, Apple and GitHub announced that the [GitHub Package Registry will support Swift packages](#) <sup>699</sup>. A few months ago, [Bryan Clark](#) <sup>74</sup> started a [thread](#) <sup>79</sup> to gather ideas about a standard package registry API that could be implemented by anyone, not only GitHub.

Over the past few weeks, I've had the privilege to work with Bryan, as well as [Whitney Imura](#) <sup>110</sup> and other great folks at GitHub, to define a draft specification for a Swift package registry service. I'm thrilled to share it with you all today.

Before diving into the proposal itself, I wanted to make two points at the outset, to help frame our discussion:

- This proposal defines an open standard for Swift package registry services. GitHub will implement these APIs as part of its [GitHub Packages](#) <sup>168</sup> offering, and anyone else will be able to create their own registry as well by implementing these endpoints.
- This proposal defines the package registry API *only*. Swift Package

Jun 2020

1/94

Jun 2020

Dec 2022

# Какие проблемы решает?

И откуда эти проблемы

Удобный способ распространения

- ``swift package publish`` вместо ``git push``

Эффективность Git`а

- Консистентность
- Эффективность скачки и кэширования
- Эффективность просчёта версий

Общий индекс

- Поиск пакетов и их версий

ДА ПОЙДЁТ))



# Как убрали git

API Сервиса

Preview Code Blame Raw Copy Download Edit More

## Package Registry Service

- Proposal: [SE-0292](#)
- Authors: [Bryan Clark](#), [Whitney Imura](#), [Matt Zmuda](#)
- Review Manager: [Tom Doron](#)
- Status: **Implemented (Swift 5.7)**
- Implementation: [apple/swift-package-manager#3023](#)
- Decision Notes: [Rationale](#)
- Review: [1](#) [2](#) [3](#) [Amendment](#)
- Previous Revision: [1](#) [2](#) [3](#)

### Introduction

Swift Package Manager downloads dependencies using Git. Our proposal defines a standard web service interface that it can also use to download dependencies from a package registry.

Swift-evolution thread: [Swift Package Registry Service](#)

### [↗](#) Motivation

A package dependency is currently specified by a URL to its source repository. When Swift Package Manager builds a



# Как убрали git

API Сервиса

The screenshot shows a GitHub file viewer for the file `swift-evolution / proposals / 0292-package-registry-service.md`. The interface includes a breadcrumb trail, a "main" branch selector, and navigation tabs for "Preview", "Code", and "Blame". On the right, there are icons for "Raw", copy, download, edit, and a menu. The main content area contains a paragraph describing REST API endpoints, followed by a table with columns for Method, Path, and Description. Below the table, there is another paragraph mentioning a formal specification and OpenAPI document.

A package registry service implements the following REST API endpoints for listing releases for a package, fetching information about a release, and downloading the source archive for a release:

Method	Path	Description
GET	<code>/ {scope} / {name}</code>	List package releases
GET	<code>/ {scope} / {name} / {version}</code>	Fetch metadata for a package release
GET	<code>/ {scope} / {name} / {version} / Package.swift {?swift-version}</code>	Fetch manifest for a package release
GET	<code>/ {scope} / {name} / {version}.zip</code>	Download source archive for a package release
GET	<code>/ identifiers {?url}</code>	Lookup package identifiers registered for a URL

A formal specification for the package registry interface is provided [alongside this proposal](#). In addition, an OpenAPI (v3) document and a reference implementation written in Swift are provided for the convenience of developers interested in building their own package registry.

# Как убрали git

## API Сервиса

main swift-evolution / proposals / 0292-package-registry-service.md ↑ Top

Preview Code Blame Raw Copy Download Edit More

### New `PackageDescription` API

The `Package.Dependency` type adds the following static method:

```
extension Package.Dependency {  
    /// Adds a dependency on a package with the specified identifier  
    /// that uses the provided version requirement.  
    public static func package(  
        id: String,  
        _ requirement: Package.Dependency.VersionBasedRequirement  
    ) -> Package.Dependency  
}
```

These methods may be called in the `dependencies` field of a package manifest to declare one or more dependencies by their respective package identifier.

```
dependencies: [  
    .package(id: "mona.LinkedList", .upToNextMinor(from: "1.1.0")),  
    .package(id: "mona.RegEx", .exact("2.0.0"))  
]
```

A package dependency declared with an identifier using this method may only specify a version-based requirement. `Package.Dependency.VersionBasedRequirement` is a new type that provides the same interface as `Package.Dependency.Requirement` for version-based requirements, but excluding branch-based and commit-based requirements.



# Как работает сервис

А как вкачать?

`https://шани-емае.py/`

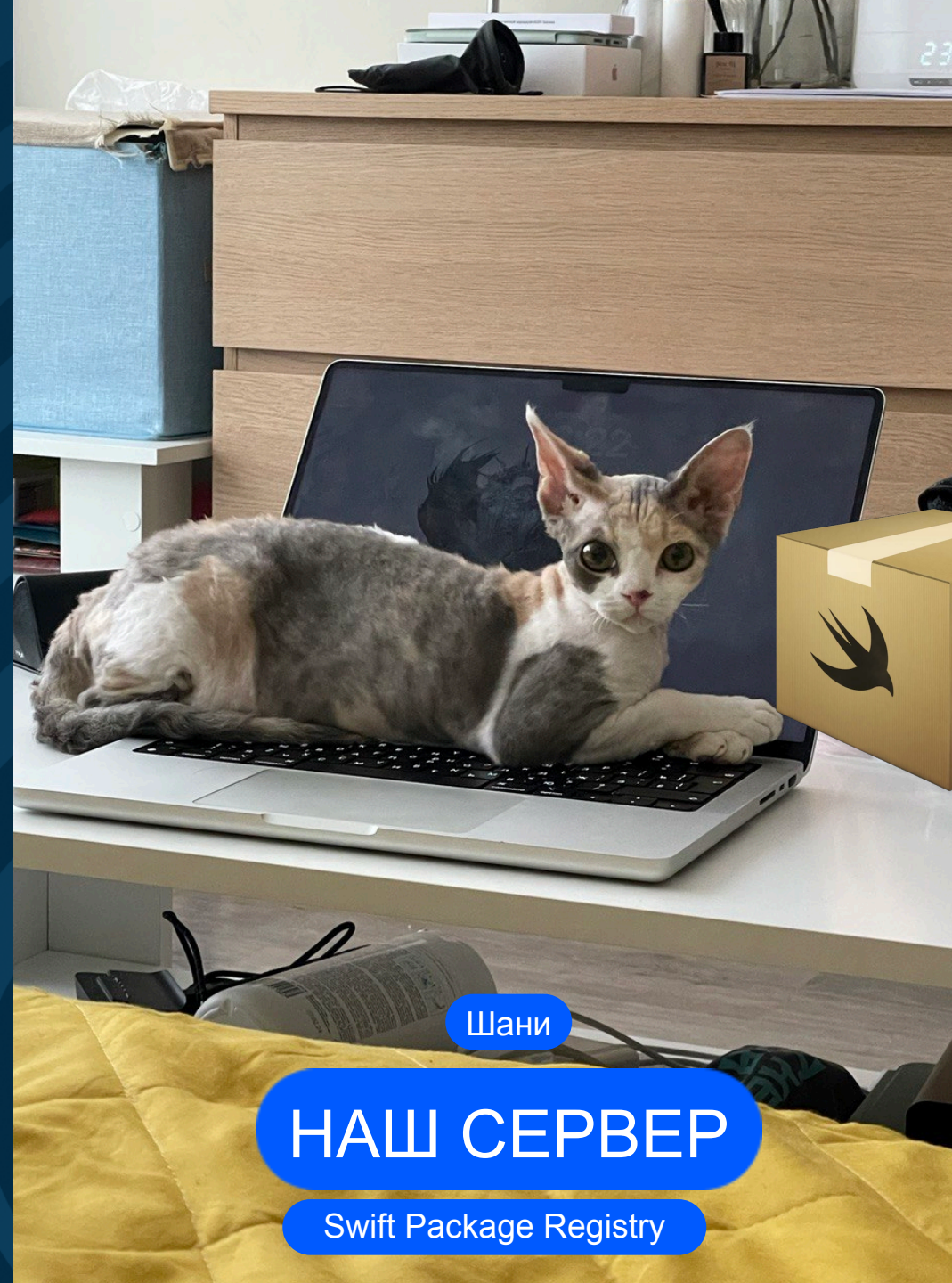


`leet-scope/leet-library/leet-library-1.33.7.zip`

[6]

PUT

`/ {scope} / {name} / {version}`



Шани

НАШ СЕРВЕР

Swift Package Registry



# Как работает сервис

А как вкачать?

<https://шани-емае.py/>



[leet-scope/leet-library/leet-library-1.33.7.zip](#)

[6]

PUT

`/ {scope} / {name} / {version}`

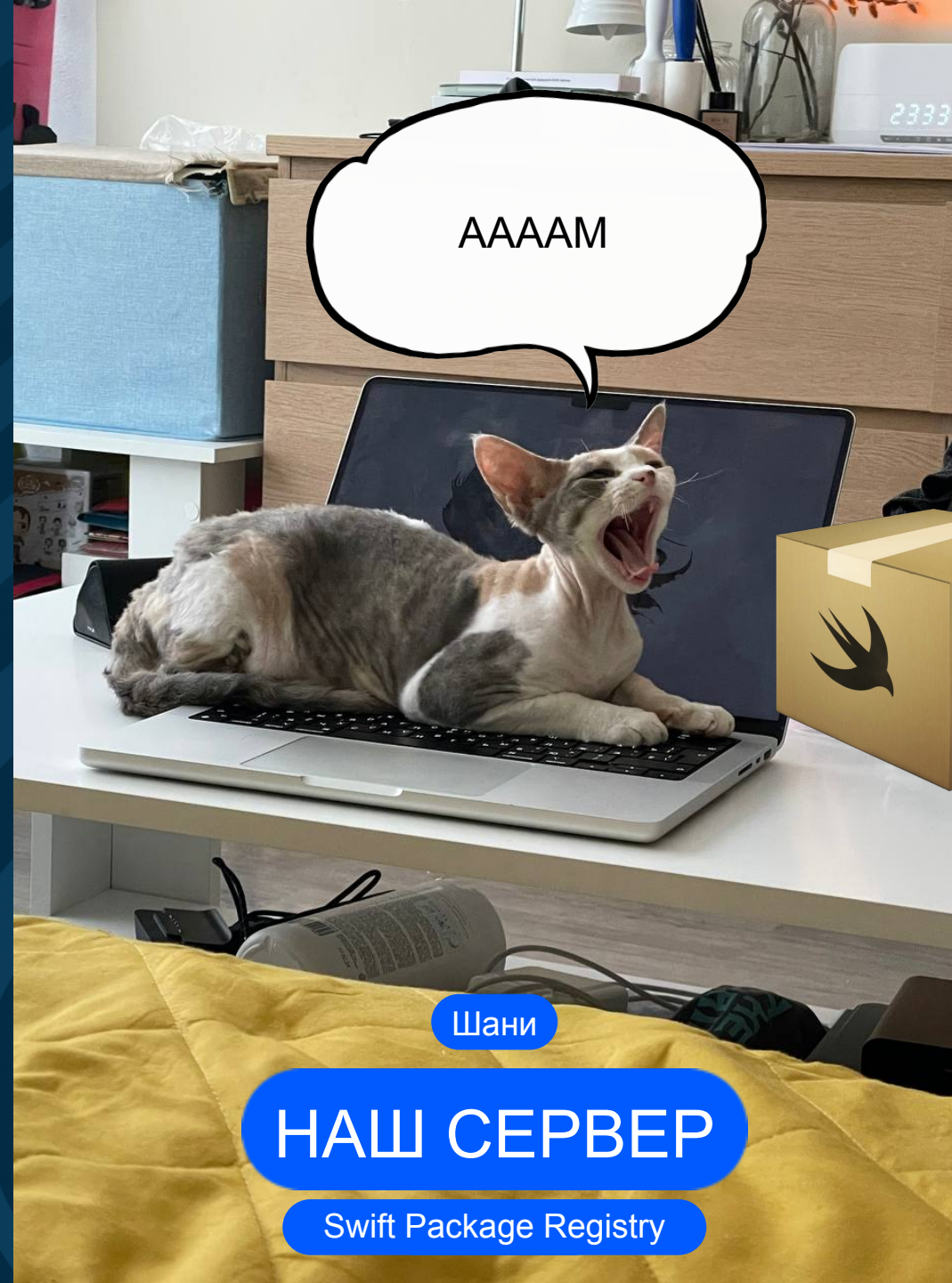


[шани/leet-scope/Leet-library-1.33.7.zip](#)

Шани

НАШ СЕРВЕР

Swift Package Registry



# Как работает сервис

А как вкачать?

```
~/Workspace/wrap git:(main)±7 (0.18s)
swift package archive-source --help
OVERVIEW: Create a source archive for the package
USAGE: swift package archive-source [--output <output>]
OPTIONS:
  -o, --output <output>  The absolute or relative path for the
                          generated source archive
  --version              Show the version.
  -h, -help, --help     Show help information.

~/Workspace/wrap git:(main)±7
```

=

```
GIT-ARCHIVE(1)          Git Manual          GIT-ARCHIVE(1)
NAME
    git-archive - Create an archive of files from a named tree
SYNOPSIS
    git archive [--format=<fmt>] [--list] [--prefix=<prefix>/] [<extra>]
                [-o <file> | --output=<file>] [--worktree-attributes]
                [--remote=<repo> [--exec=<git-upload-archive>]] <tree-ish>
                [<path>...]
DESCRIPTION
    Creates an archive of the specified format containing the tree structure for the named tree, and writes it out to the standard output. If <prefix> is specified it is prepended to the filenames in the archive.
    git archive behaves differently when given a tree ID versus when given a commit ID or tag ID. In the first case the
```



# Как работает сервис

А как скачать?

`https://шани-емае.py/`



`leet-scope/leet-library/leet-library-1.33.7.zip`

GET

`/{{scope}}/{{name}}/{{version}}.zip`



Шани

НАШ СЕРВЕР



# Как работает сервис

А как скачать?

<https://шани-емае.py/>



[leet-scope/leet-library/leet-library-1.33.7.zip](#)

GET

`/ {scope} / {name} / {version} .zip`



build/registry/Leet-library  
1.33.7.zip



ozon{tech



Шани

НАШ СЕРВЕР

Swift Package Registry

# Как работает сервис

Как использовать у себя?

```
1 import PackageDescription
2
3 let package = Package(
4     name: "MyPackage",
5     dependencies: [
6         .package(
7             url: "https://github.com/Moya/Moya.git",
8             from: "15.0.0"
9         ),
10     ]
11 )
```

```
1 import PackageDescription
2
3 let package = Package(
4     name: "MyPackage",
5     dependencies: [
6         .package(
7             id: "test.Moya",
8             from: "15.0.0"
9         ),
10     ]
11 )
```

# Как работает сервис

Как использовать у себя?

```
~/Workspace/swift-external-dependencies
swift package-registry set --help
OVERVIEW: Set a custom registry
USAGE: swift package-registry set [--global] [--scope <scope>] <url>
ARGUMENTS:
  <url>          The registry URL
OPTIONS:
  --global      Apply settings to all projects for this user
  --scope <scope> Associate the registry with a given scope
  --version     Show the version.
  -h, -help, --help Show help information.

~/Workspace/swift-external-dependencies
```

```
1 // registries.json
2 {
3   "authentication" : {
4
5   },
6   "registries" : {
7     "leet-scope" : {
8       "supportsAvailability" : false,
9       "url" : "sanya.org/"
10    },
11  },
12  "version" : 1
13 }
```

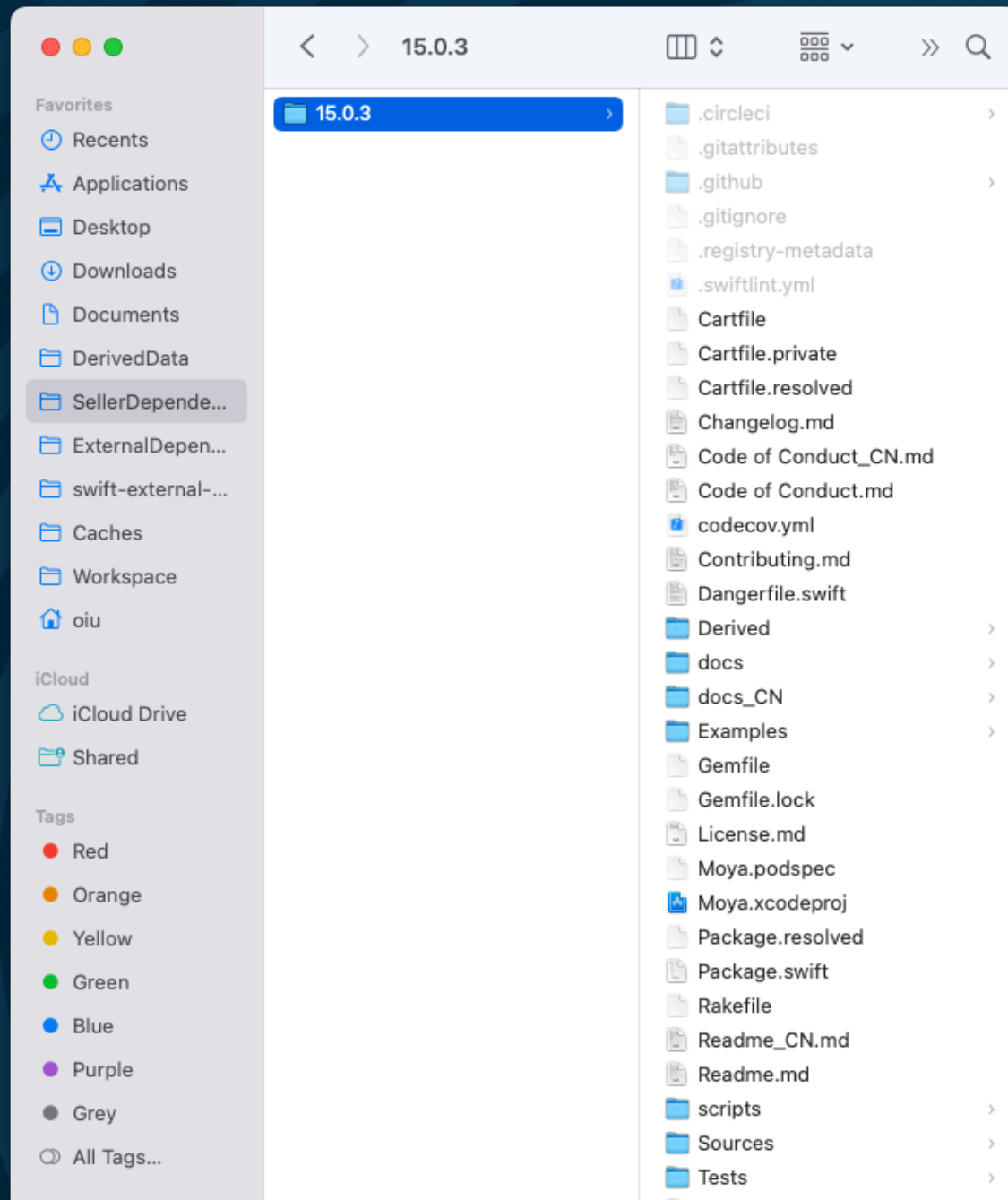


# Как работает сервис

А как выглядит на тачке?



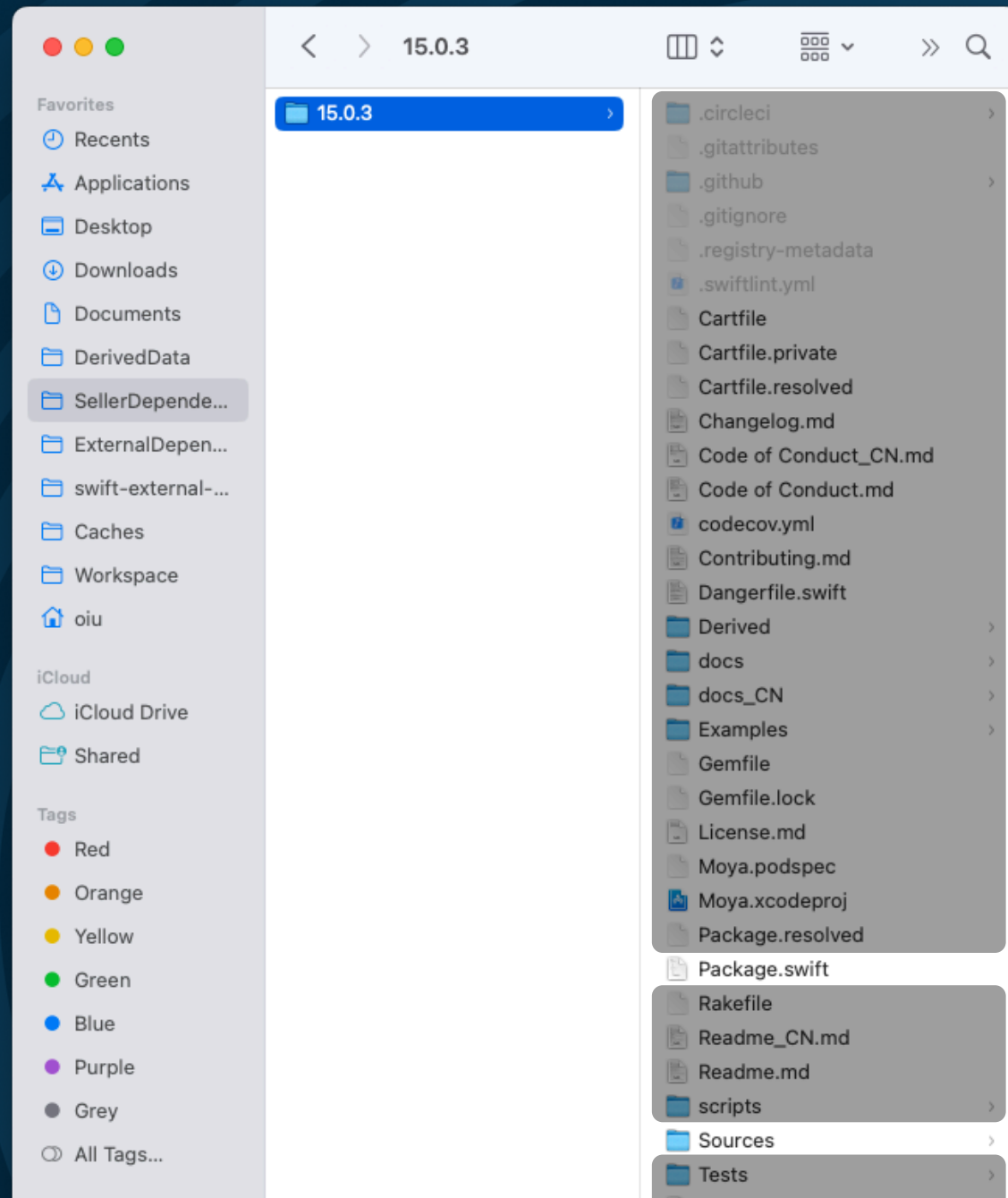
```
1 public struct RegistryClient: RepositoryProvider {
2
3     // ...
4
5     func _downloadSourceArchive( // <- строка 967
6         registry: Registry,
7         package: PackageIdentity.RegistryIdentity,
8         version: Version,
9         destinationPath: AbsolutePath,
10        progressHandler: ((_ bytesReceived: Int64, _
11        totalBytes: Int64?) -> Void)?,
12        timeout: DispatchTimeInterval?,
13        fileSystem: FileSystem,
14        observabilityScope: ObservabilityScope,
15        callbackQueue: DispatchQueue,
16        completion: @escaping (Result<Void, Error>) -> Void
17    ) {
18        // ... скачка, валидации чексумм и подписей
19    } // <- строка 1200
20 }
```



# Как работает сервис

А как выглядит на тачке?

```
1 public struct RegistryClient: RepositoryProvider {
2
3     // ...
4
5     func _downloadSourceArchive( // <- строка 967
6         registry: Registry,
7         package: PackageIdentity.RegistryIdentity,
8         version: Version,
9         destinationPath: AbsolutePath,
10        progressHandler: ((_ bytesReceived: Int64, _
11        totalBytes: Int64?) -> Void)?,
12        timeout: DispatchTimeInterval?,
13        fileSystem: FileSystem,
14        observabilityScope: ObservabilityScope,
15        callbackQueue: DispatchQueue,
16        completion: @escaping (Result<Void, Error>) -> Void
17    ) {
18        // ... скачка, валидации чексумм и подписей
19    } // <- строка 1200
20 }
```



03



В чём бенефиты  
И где нюансы



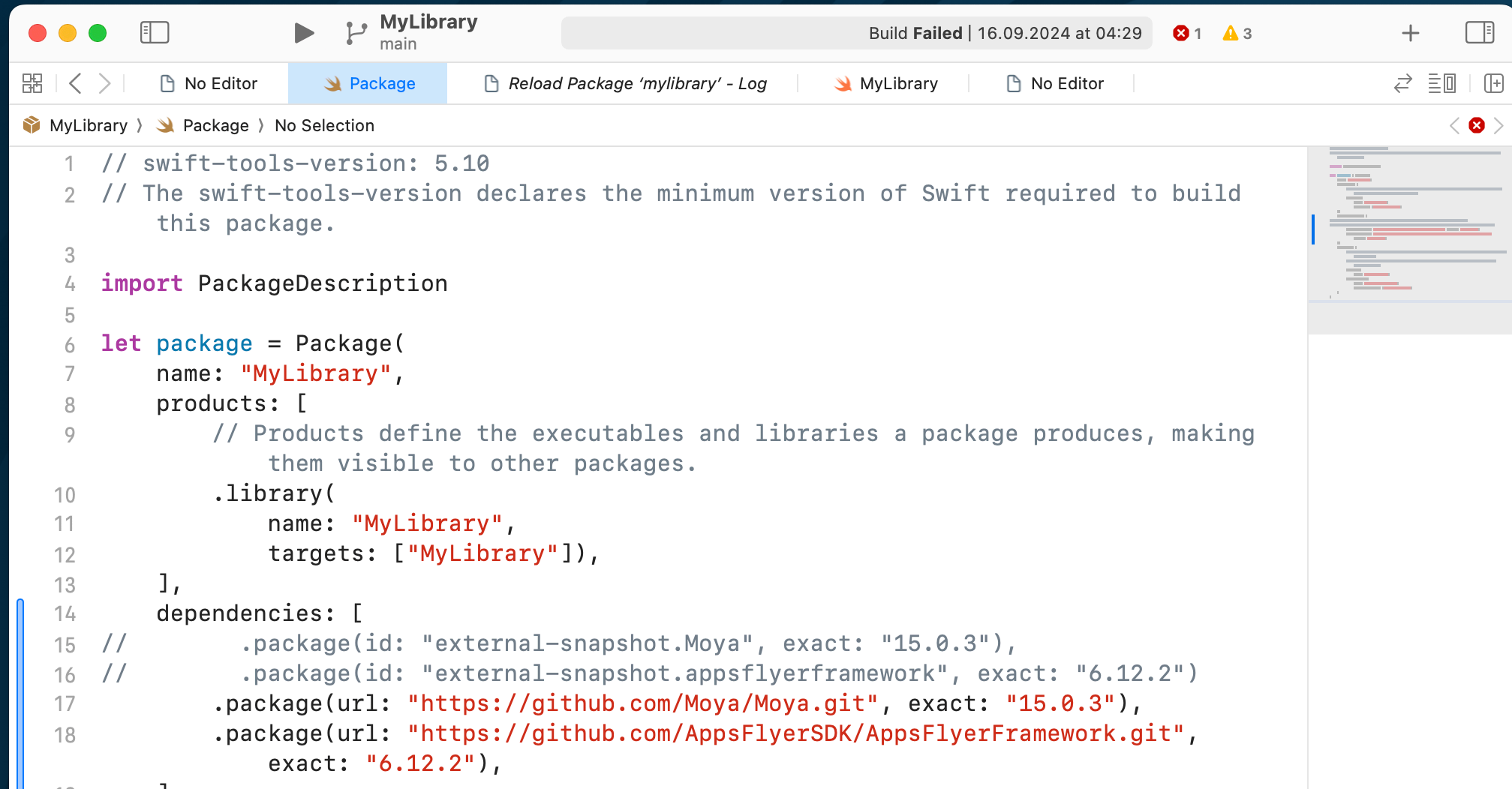


# Что получилось в итоге?

	SPM	SPR
Скорость	-	+
Эксплуатация и обслуживание	+	-

# Почему скорость

Прямое сравнение



The screenshot shows an IDE window for a project named "MyLibrary". The title bar indicates a "Build Failed" status at 16.09.2024 at 04:29, with 1 error and 3 warnings. The editor displays the Package.swift file with the following content:

```
1 // swift-tools-version: 5.10
2 // The swift-tools-version declares the minimum version of Swift required to build
   this package.
3
4 import PackageDescription
5
6 let package = Package(
7     name: "MyLibrary",
8     products: [
9         // Products define the executables and libraries a package produces, making
   them visible to other packages.
10        .library(
11            name: "MyLibrary",
12            targets: ["MyLibrary"]),
13    ],
14    dependencies: [
15        // .package(id: "external-snapshot.Moya", exact: "15.0.3"),
16        // .package(id: "external-snapshot.appsflyerframework", exact: "6.12.2")
17        .package(url: "https://github.com/Moya/Moya.git", exact: "15.0.3"),
18        .package(url: "https://github.com/AppsFlyerSDK/AppsFlyerFramework.git",
   exact: "6.12.2"),
19    ]
20 }
```

# Почему скорость

## Прямое сравнение

```
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (6m 11.63s)
swift package purge-cache; \
swift package reset; \
swift package resolve

Fetched https://github.com/ReactiveCocoa/ReactiveSwift.git from cache (22.43s)
Fetched https://github.com/ReactiveX/RxSwift.git from cache (22.84s)
Computing version for https://github.com/ReactiveX/RxSwift.git
Computed https://github.com/ReactiveX/RxSwift.git at 6.7.1 (0.53s)
Computing version for https://github.com/ReactiveCocoa/ReactiveSwift.git
Computed https://github.com/ReactiveCocoa/ReactiveSwift.git at 6.7.0 (0.75s)
Computing version for https://github.com/Alamofire/Alamofire.git
Computed https://github.com/Alamofire/Alamofire.git at 5.9.1 (0.42s)
Creating working copy for https://github.com/AppsFlyerSDK/AppsFlyerFramework.git
Working copy of https://github.com/AppsFlyerSDK/AppsFlyerFramework.git resolved at 6.12.2
Creating working copy for https://github.com/ReactiveCocoa/ReactiveSwift.git
Working copy of https://github.com/ReactiveCocoa/ReactiveSwift.git resolved at 6.7.0
Creating working copy for https://github.com/Moya/Moya.git
Working copy of https://github.com/Moya/Moya.git resolved at 15.0.3
Creating working copy for https://github.com/Alamofire/Alamofire.git
Working copy of https://github.com/Alamofire/Alamofire.git resolved at 5.9.1
Creating working copy for https://github.com/ReactiveX/RxSwift.git
Working copy of https://github.com/ReactiveX/RxSwift.git resolved at 6.7.1
Fetching binary artifact https://github.com/AppsFlyerSDK/AppsFlyerFramework/releases/download/6.12.2/AppsFlyerLib.xcframework.zip from cache
```

```
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (0.016s)
ls

~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (9.212s)
swift package purge-cache; \
swift package reset; \
swift package resolve

Fetching external-snapshot.rxswift
warning: 'external-snapshot.rxswift': external-snapshot.rxswift version 6.2.0 source archive from
is not signed
Fetched external-snapshot.rxswift from cache (1.42s)
Fetching external-snapshot.Moya
warning: 'external-snapshot.Moya': external-snapshot.Moya version 15.0.3 source archive from
is not signed
Fetched external-snapshot.Moya from cache (0.23s)
Fetching external-snapshot.appsflyerframework
warning: 'external-snapshot.appsflyerframework': external-snapshot.appsflyerframework version 6.12.2 source archive from
is not signed
Fetched external-snapshot.appsflyerframework from cache (0.07s)
Fetching external-snapshot.alamofire
warning: 'external-snapshot.alamofire': external-snapshot.alamofire version 5.6.2 source archive from
is not signed
Fetched external-snapshot.alamofire from cache (0.89s)
Fetching external-snapshot.reactiveswift
warning: 'external-snapshot.reactiveswift': external-snapshot.reactiveswift version 6.6.1 source archive from
is not signed
Fetched external-snapshot.reactiveswift from cache (0.25s)
```



# Почему скорость

## Прямое сравнение

```
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±3 (28.65s)
swift package resolve
Computed https://github.com/moya/moya.git at 15.0.3 (0.02s)
Fetching https://github.com/ReactiveCocoa/ReactiveSwift.git from cache
Fetching https://github.com/ReactiveX/RxSwift.git from cache
Fetching https://github.com/Alamofire/Alamofire.git from cache
Fetched https://github.com/Alamofire/Alamofire.git from cache (1.54s)
Fetched https://github.com/ReactiveCocoa/ReactiveSwift.git from cache (1.88s)
Fetched https://github.com/ReactiveX/RxSwift.git from cache (2.21s)
Computing version for https://github.com/ReactiveX/RxSwift.git
Computed https://github.com/ReactiveX/RxSwift.git at 6.7.1 (0.02s)
Computing version for https://github.com/ReactiveCocoa/ReactiveSwift.g
it
Computed https://github.com/ReactiveCocoa/ReactiveSwift.git at 6.7.0 (
0.02s)
Computing version for https://github.com/Alamofire/Alamofire.git
Computed https://github.com/Alamofire/Alamofire.git at 5.9.1 (0.03s)
Creating working copy for https://github.com/Moya/Moya.git
Working copy of https://github.com/Moya/Moya.git resolved at 15.0.3
Creating working copy for https://github.com/ReactiveX/RxSwift.git
Working copy of https://github.com/ReactiveX/RxSwift.git resolved at 6
.7.1
Creating working copy for https://github.com/AppsFlyerSDK/AppsFlyerFra
mework.git
Working copy of https://github.com/AppsFlyerSDK/AppsFlyerFramework.git
resolved at 6.12.2
Creating working copy for https://github.com/Alamofire/Alamofire.git
Working copy of https://github.com/Alamofire/Alamofire.git resolved at
5.9.1
Creating working copy for https://github.com/ReactiveCocoa/ReactiveSwi
ft.git
```

```
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (0.016s)
ls

~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (9.212s)
swift package purge-cache; \
swift package reset; \
swift package resolve

Fetching external-snapshot.rxswift
warning: 'external-snapshot.rxswift': external-snapshot.rxswift versio
n 6.2.0 source archive from
is not signed
Fetched external-snapshot.rxswift from cache (1.42s)
Fetching external-snapshot.Moya
warning: 'external-snapshot.Moya': external-snapshot.Moya version 15.0
.3 source archive from
is not signed
Fetched external-snapshot.Moya from cache (0.23s)
Fetching external-snapshot.appsflyerframework
warning: 'external-snapshot.appsflyerframework': external-snapshot.app
sflyerframework version 6.12.2 source archive from
is not signed
Fetched external-snapshot.appsflyerframework from cache (0.07s)
Fetching external-snapshot.alamofire
warning: 'external-snapshot.alamofire': external-snapshot.alamofire ve
rsion 5.6.2 source archive from
is not signed
Fetched external-snapshot.alamofire from cache (0.89s)
Fetching external-snapshot.reactiveswift
warning: 'external-snapshot.reactiveswift': external-snapshot.reacti
ve swift version 6.6.1 source archive from
is not signed
Fetched external-snapshot.reactiveswift from cache (0.25s)
```

# Почему скорость

Что у SPM?

- Чекауты могут быть долгими на пустой кэш
- Одни чекауты задерживают другие (простройка графа)
  - Сначала загружаем пакет и разбираем его манифест, а уже потом переходим к его зависимостям
- Кэш работает нестабильно (и его много)

```
swift-syntax
https://github.com/apple/swift-syntax.git
! Fetching from https://github.com/apple/swift-syntax.git 199.7 seconds
! skipping cache due to an error: https://github.com/apple/swift-syntax.git: An unknown error occurred. http parser error: stream ended at an unexpected time (-1)
```

# Почему скорость

Что у SPR?

- Есть отдельная ручка на запрос всех версий
- Есть отдельная ручка на архив с самим пакетом
- Кэши сильно легче и не протухают (просчёт через md5)





# Почему эксплуатация

У SPM

- Типичная жизнь с гитом
  - Иногда постреливающие креды
  - Внешний репозиторий может быть недоступен
  - Тег, на который мы завязались, может быть снесён
- В остальном — все привыкли



# Почему эксплуатация

У SPR

- Нужно предоставлять конфиги `swift package-registry set`
- Скорее всего — предоставлять зеркала с гитовых реп
- ... есть нюанс)



# Почему эксплуатация

## А где брать registry зависимости?

The screenshot shows the Xcode interface with a Swift Package Manifest file open. The file content is as follows:

```
1 // swift-tools-version:5.0
2
3 import PackageDescription
4
5 let package = Package(
6     name: "Moya",
7     platforms: [
8         .macOS(.v10_12),
9         .iOS(.v10),
10        .tvOS(.v10),
11        .watchOS(.v3)
12    ],
13    products: [
14        .library(name: "Moya", targets: ["Moya"]),
15        .library(name: "ReactiveMoya", targets: ["ReactiveMoya"]),
16        .library(name: "RxMoya", targets: ["RxMoya"])
17    ],
18    dependencies: [
19        .package(url: "https://github.com/Alamofire/Alamofire.git",
20                .upToNextMajor(from: "5.0.0")),
21        .package(url: "https://github.com/Moya/ReactiveSwift.git",
22                .upToNextMajor(from: "6.1.0")),
23        .package(url: "https://github.com/ReactiveX/RxSwift.git",
24                .upToNextMajor(from: "5.0.0")),
25        .package(url: "https://github.com/Quick/Quick.git", .upToNextMajor(from:
26                "2.0.0")), // dev
27        .package(url: "https://github.com/Quick/Nimble.git", .upToNextMajor(from:
28                "8.0.0")), // dev
29        .package(url: "https://github.com/AlSoftware/OHHTTPStubs.git",
30                .upToNextMajor(from: "9.0.0")), // dev
31        .package(url: "https://github.com/shibapm/Rocket", .upToNextMajor(from:
32                "1.0.0")) // dev
33    ],
34    targets: [
35        .target(name: "Moya", dependencies: ["Alamofire"]),
36    ]
37)
```

The dependencies section (lines 19-32) is highlighted with a blue box. The right sidebar shows "No Selection".



# Почему эксплуатация

## А где брать registry зависимости?

The screenshot shows the Xcode Registry application interface. On the left is a sidebar with a project tree and package dependencies. The main area displays the contents of a Package.swift file for a package named 'Moya'. The code defines the package name, supported platforms, products, and dependencies. A blue box highlights the 'dependencies' array in the code. Several warnings are visible in the editor, indicating that certain platform version ranges are deprecated.

```
1 // swift-tools-version:5.10
2
3 import PackageDescription
4
5 let package = Package(
6     name: "Moya",
7     platforms: [
8         .macOS(.v10_12),
9         .iOS(.v10),
10        .tvOS(.v10),
11        .watchOS(.v3)
12    ],
13    products: [
14        .library(name: "Moya", targets: ["Moya"]),
15        .library(name: "ReactiveMoya", targets: ["ReactiveMoya"]),
16        .library(name: "RxMoya", targets: ["RxMoya"])
17    ],
18    dependencies: [
19        .package(id: "test.Alamofire", from: "5.0.0"),
20        .package(id: "test.ReactiveSwift", from: "6.1.0"),
21        .package(id: "test.RxSwift", from: "5.0.0"),
22        .package(id: "test.Quick", from: "2.0.0"),
23        .package(id: "test.Nimble", from: "8.0.0"),
24        .package(id: "test.OHHTTPStubs", from: "9.0.0"),
25        .package(id: "test.Rocket", from: "1.0.0"),
26    ],
27    targets: [
28        .target(name: "Moya", dependencies: [.product(name: "Alamofire", package:
29            "test.Alamofire")]),
30        .target(name: "ReactiveMoya", dependencies: ["Moya", .product(name:
31            "ReactiveSwift", package: "test.ReactiveSwift")]),
32        .target(name: "RxMoya", dependencies: ["Moya", .product(name: "RxSwift",
33            package: "test.RxSwift")]),
34        .testTarget(name: "MoyaTests", dependencies: []) // dev
35    ]
36 ]
```

Warnings shown in the editor:

- 'v10\_12' is deprecated: macOS 10.13 is the oldest supported version
- 'v10' is deprecated: iOS 12.0 is the oldest supported version
- 'v10' is deprecated: tvOS 12.0 is the oldest supported version
- 'v3' is deprecated: watchOS 4.0 is the oldest supported version

Package Dependencies:

- Alamofire 5.0.0
- Moya 14.0.1
  - Readme
  - Package
  - docs
  - docs\_CN
  - Examples
  - scripts
  - Sources
  - Tests
  - web
  - Cartfile
  - Cartfile.private
  - Cartfile.resolved
  - Changelog
  - Code of Conduct\_CN
  - Code of Conduct
  - codecov
  - Contributing
  - Dangerfile

Right sidebar: No Selection

# Почему эксплуатация

## Версионирование транзитивных зависимостей



```
1 .package(  
2   url: "https://github.com/Moya/Moya.git",  
3   exact: "15.0.3"  
4 ),  
5 // ... depends on  
6 .package(  
7   url: "https://github.com/Alamofire/Alamofire.git",  
8   exact: "5.6.2"  
9 ),
```



```
1 .package(  
2   url: "https://github.com/Moya/Moya.git",  
3   exact: "15.0.3"  
4 ),  
5 // ... depends on  
6 .package(  
7   url: "https://github.com/Alamofire/Alamofire.git",  
8   branch: "update/xcode-13.3"  
9 ),
```

# Почему эксплуатация

## Версионирование транзитивных зависимостей



```
1 .package(  
2   id: "external-snapshot.moya",  
3   exact: "15.0.3"  
4 ),  
5 // ... depends on  
6 .package(  
7   id: "external-snapshot.alamofire",  
8   exact: "5.6.2"  
9 ),
```



```
1 .package(  
2   id: "external-snapshot.moya",  
3   exact: "15.0.3"  
4 ),  
5 // ... depends on  
6 .package(  
7   id: "external-snapshot.alamofire",  
8   exact: "5.9.1-custom-rc"  
9 ),
```



# Почему эксплуатация

## Версионирование транзитивных зависимостей

```
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (2.08s)
swift package purge-cache; \
swift package reset; \
swift package resolve

Computing version for external-snapshot.snapkit
Computed external-snapshot.snapkit at 5.0.1 (0.41s)
Computing version for external-snapshot.moya
error: Dependencies could not be resolved because root depends on 'external-snapshot.moya' 15.0.3.
'external-snapshot.moya' 15.0.3 cannot be used because 'external-snapshot.moya' 15.0.3 depends on 'external-snapshot.alamofire' 5.0.0..<6.0.0 and root depends on 'external-snapshot.alamofire' 5.9.1-custom-rc
.
```

```
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (0.036s)
clear
```

```
1 .package(
2     id: "external-snapshot.moya",
3     exact: "15.0.3"
4 ),
5 // ... depends on
6 .package(
7     id: "external-snapshot.alamofire",
8     exact: "5.9.1-custom-rc"
9 ),
```

# Почему эксплуатация

## Версионирование транзитивных зависимостей

```
~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (2.08s)
swift package purge-cache; \
swift package reset; \
swift package resolve

Computing version for external-snapshot.snapkit
Computed external-snapshot.snapkit at 5.0.1 (0.41s)
Computing version for external-snapshot.moya
error: Dependencies could not be resolved because root depends on 'external-snapshot.moya' 15.0.3.
'external-snapshot.moya' 15.0.3 cannot be used because 'external-snapshot.moya' 15.0.3 depends on 'external-snapshot.alamofire' 5.0.0..<6.0.0 and root depends on 'external-snapshot.alamofire' 5.9.1-custom-rc
.

~/Workspace/Registry/Registry/Features/MyLibrary git:(main)±4 (0.036s)
clear
```

Version 5.9.1-custom-rc ✓

### 5.9.1-custom-rc

Given the version you entered:

- The next **major** release will be 6.0.0
- The next **premajor** release will be 6.0.0-0
- The next **minor** release will be 5.10.0
- The next **preminor** release will be 5.10.0-0
- The next **patch** release will be 5.9.1
- The next **prepatch** release will be 5.9.2-0
- The next **prerelease** release will be 5.9.1-custom-rc.0

# Почему обслуживание

SPM + SPR

- За исключением стандартных требований к работе с SPM (контроль подключений и источников), нас всё устраивает, пока сторонние Git-репозитории, с которых мы забираем нужные зависимости, работают стабильно
- По registry:
  - Для использования registry нужны свои серверы со своей имплементацией спеки, с которой сможет «общаться Xcode», а также мониторинг конфигураций (проставление скоупов) — это требует денежных средств
  - Плюсы: больше контроля. Минусы: больше контроля



# Почему обслуживание

Нюансы перевода проектов с SPM -> SPR

**Registry** — не самая стабильная штука при работе с обычными готовыми пакетами

- Постреливает кэш из-за identity пакетов, которые раньше грузились с git

Если вы релизились только в git, а потом резко начали релизиться только в registry — **у вас проблемы**

- При переводе проектов и обновлении версий часто приходится опираться на зеркала, иначе будем ловить дубликаты identity
- Проект может неправильно разрешить зависимости: **если нужной версии нет в registry, SPM автоматически подтянет пакет с соответствующей identity из git. Однако этот пакет может не подходить по ABI. Резолв пройдёт успешно, но сборка будет падать**

# Как можно использовать registry?

## Безопасность и стабильность

1. Так как в registry пакеты приходится выгружать — их в любом случае надо как-то обрабатывать
2. Раз уж пакеты приходится трогать (автоматикой или вручную), их можно проверять на уязвимости в безопасности
3. Перепроверка и валидация фингерпринтов и подписей
4. Выгружая все пакеты на локальный ресурс, можно в конце концов полностью уйти от использования внешних сервисов (например, GitHub)
5. Можно контролировать доступ к любым библиотекам, если все клиенты смотрят только на registry (dependency whitelist)

# Как можно использовать registry?

swift-external-dependencies

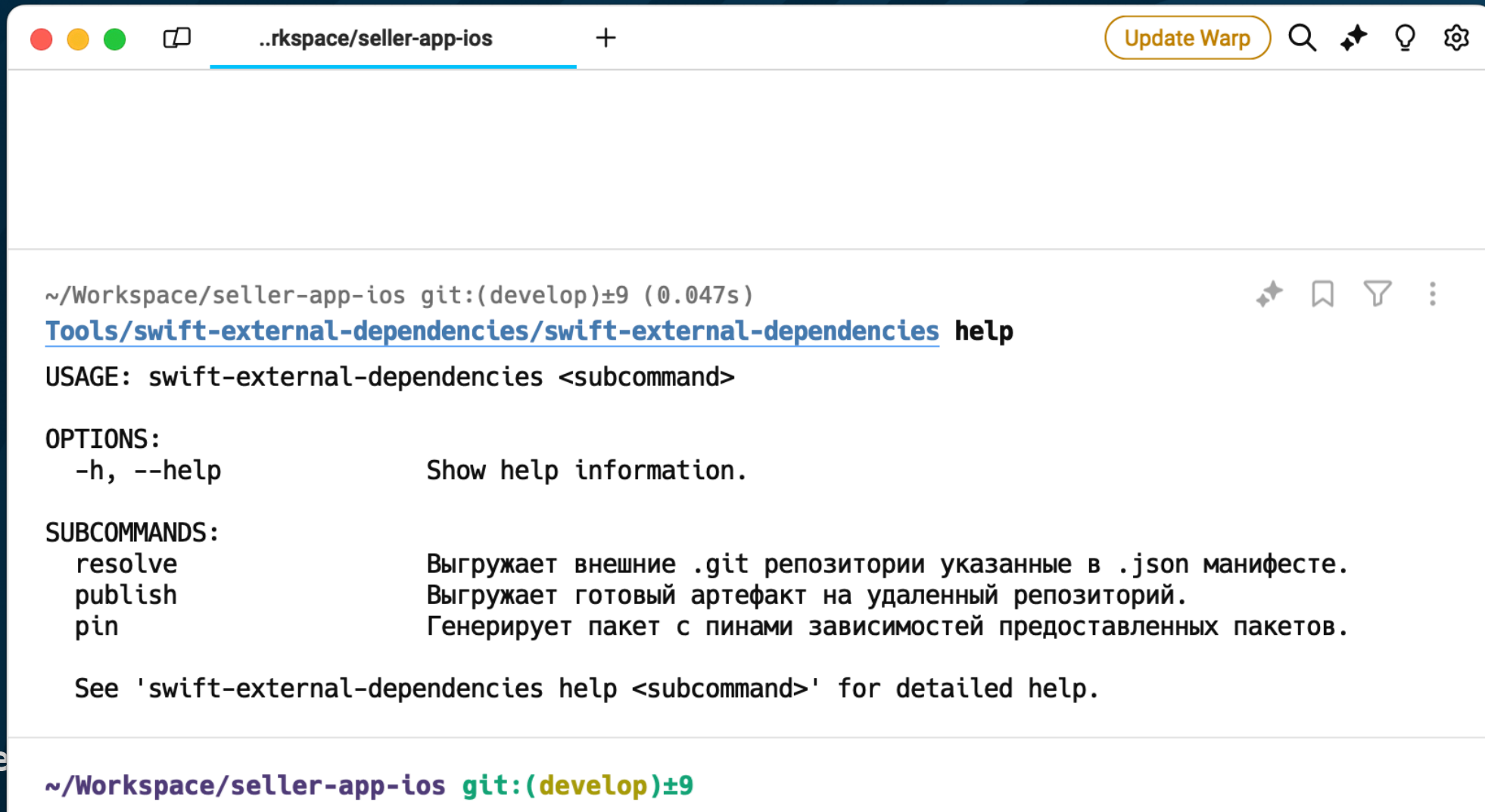
swift-external-dependencies и воркфлоу разработчика

- В репозитории лежит .json файл, который является апрунутым белым листом внешних зависимостей (spm пакетов)
- Разработчик коммитит в репозиторий изменение списка зависимостей и получает апрувы от безопасников
- После апрува и мержа, внешний репозиторий и его транзитивные зависимости выгружаются в нашу registry



# Как можно использовать registry?

swift-external-dependencies



```
~/Workspace/seller-app-ios git:(develop)±9 (0.047s)
Tools/swift-external-dependencies/swift-external-dependencies help
USAGE: swift-external-dependencies <subcommand>

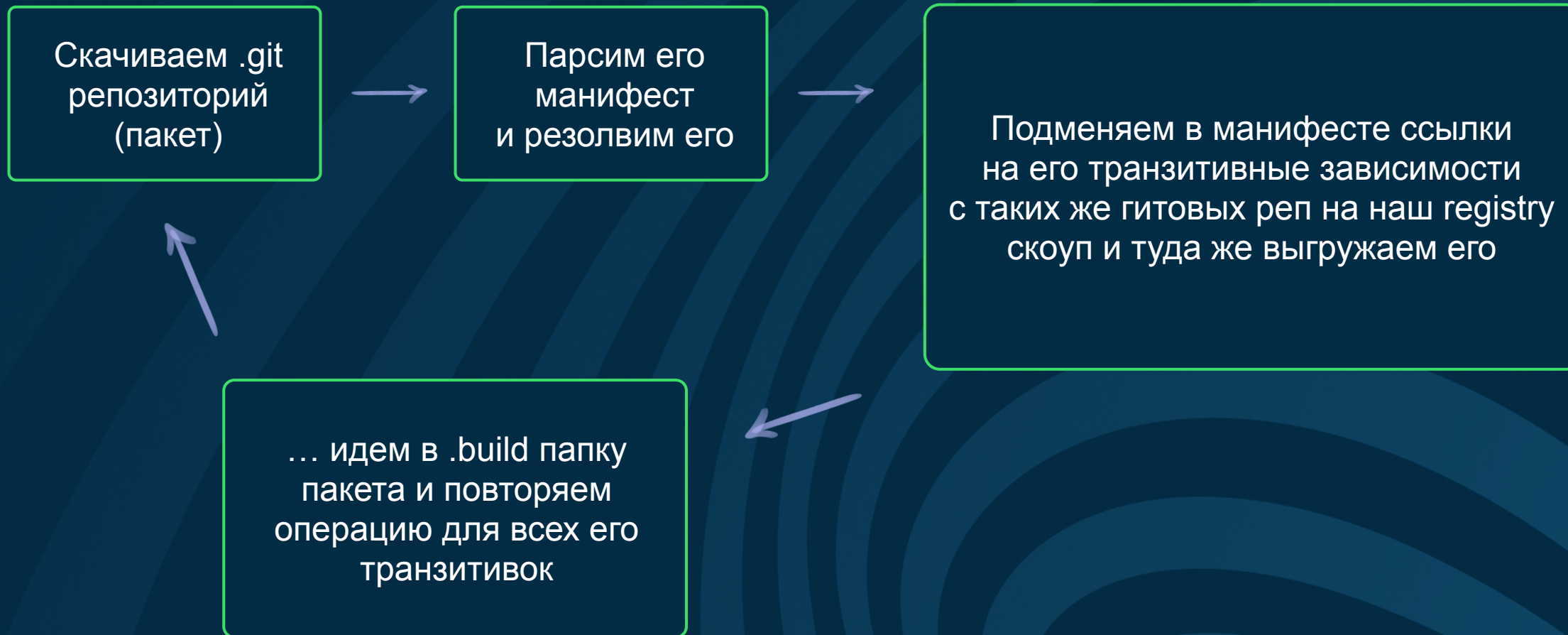
OPTIONS:
  -h, --help          Show help information.

SUBCOMMANDS:
  resolve             Выгружает внешние .git репозитории указанные в .json манифесте.
  publish             Выгружает готовый артефакт на удаленный репозиторий.
  pin                 Генерирует пакет с пинами зависимостей предоставленных пакетов.

See 'swift-external-dependencies help <subcommand>' for detailed help.
```

# Как можно использовать registry?

swift-external-dependencies

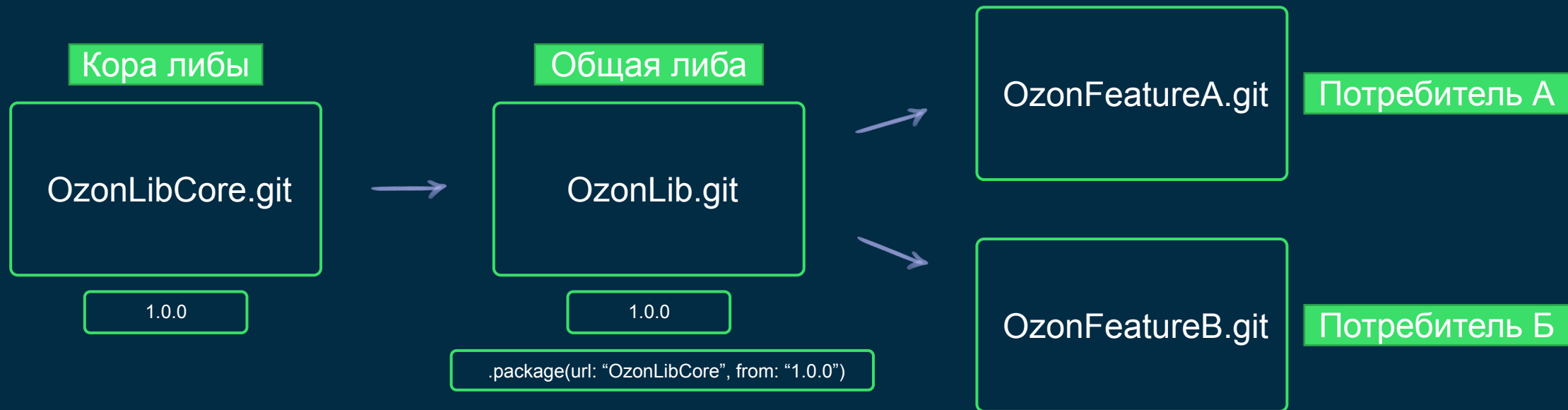


# Как можно использовать registry?

Монорепа для внутренних библиотек и инструментов

Мы построили монорепа на основе registry — хотите так же?

- Зачем нужна?



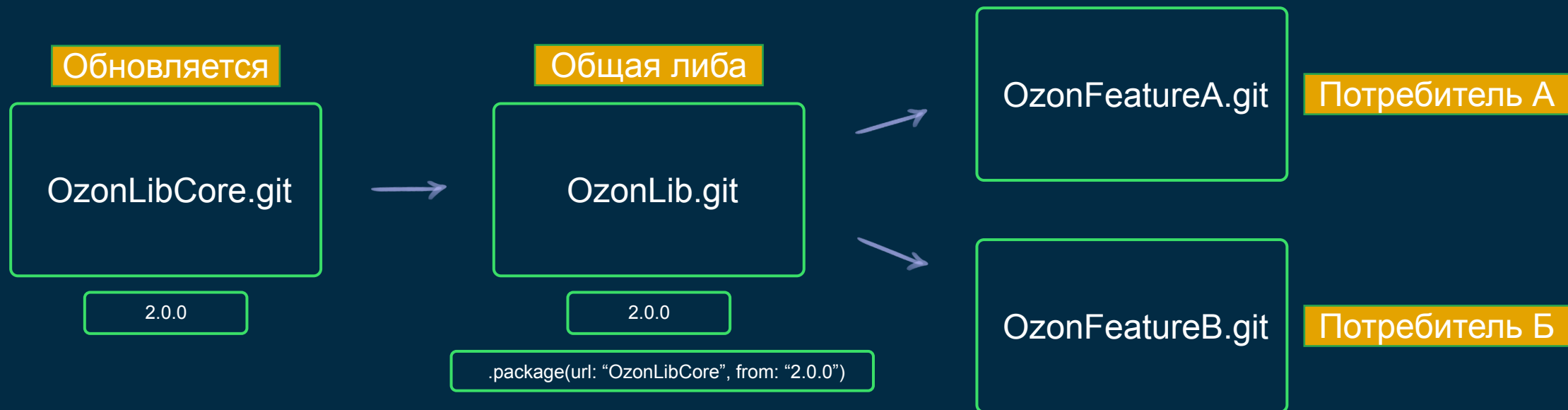


# Как можно использовать registry?

Монорепа для внутренних библиотек и инструментов

Мы построили монорепа на основе registry — хотите так же?

- Зачем нужна?

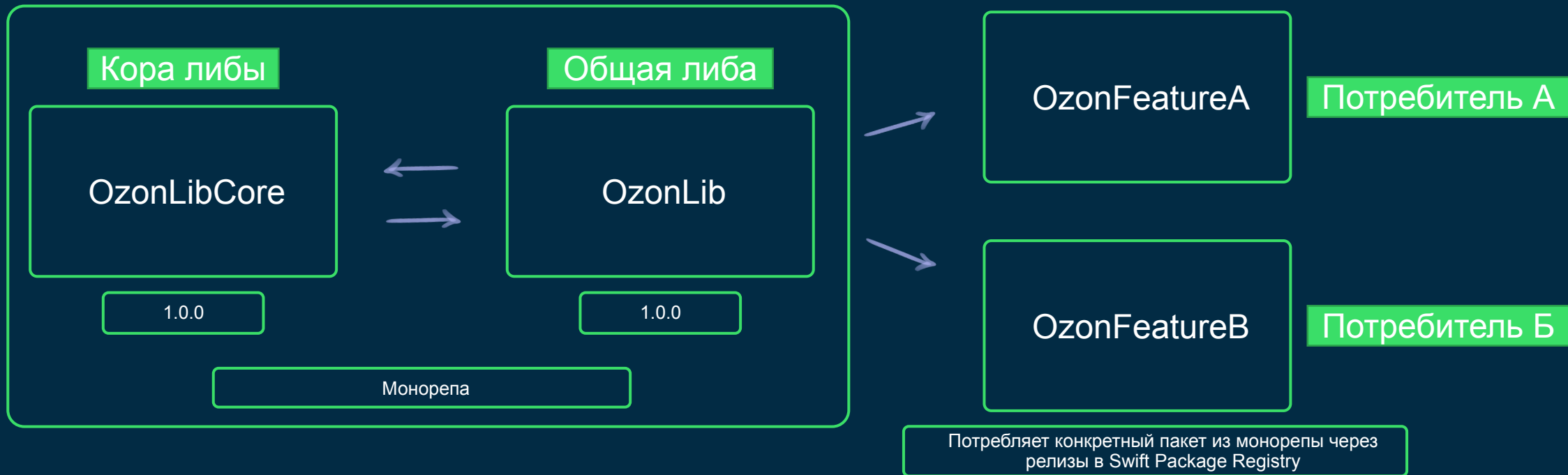


# Как можно использовать registry?

Монорепа для внутренних библиотек и инструментов

Мы построили монорепо на основе registry — хотите так же?

- Зачем нужна?



# Как можно использовать registry?

Монорепа для внутренних библиотек и инструментов

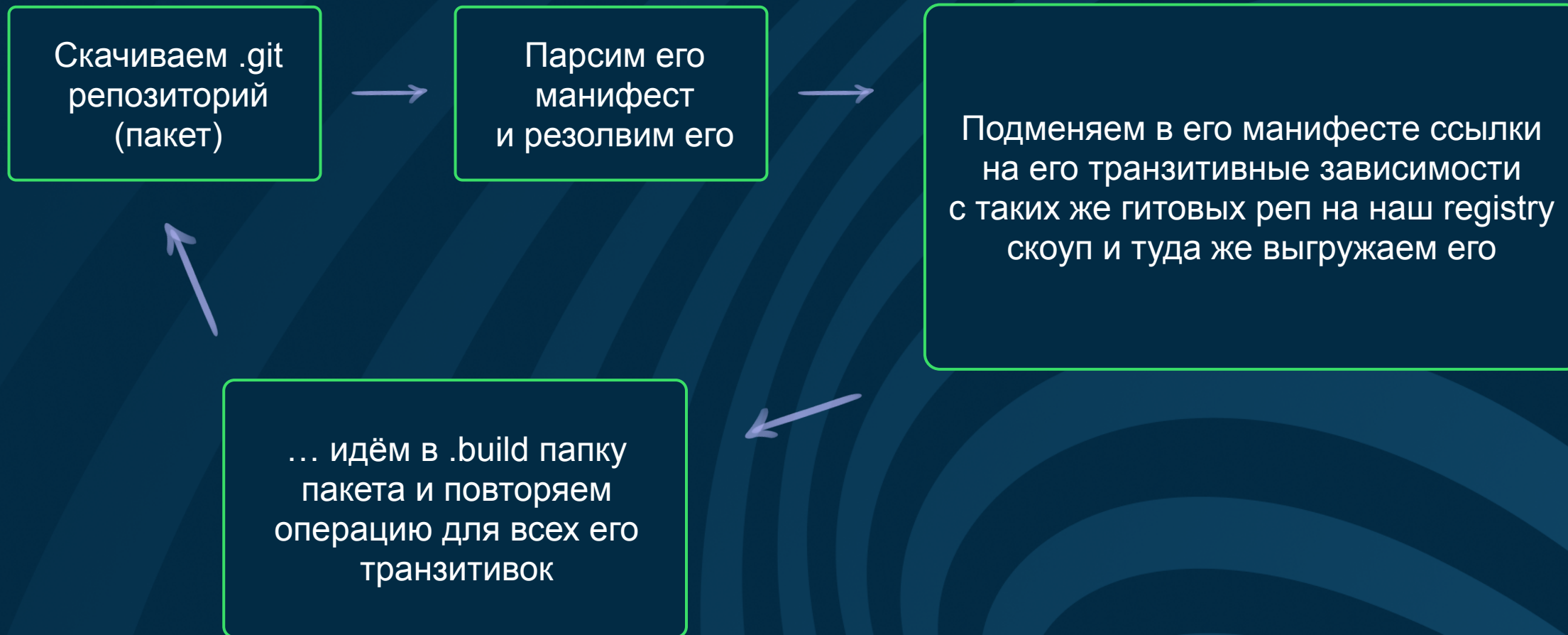
Мы построили монорепу на основе registry — хотите так же?

- Монорепа с разбойными релизами
- Работаем со всеми исходниками, а на релизы выгружаем отдельный архив конкретного пакета из монорепы (сильно меньше *dependency hell* при релизе нескольких пакетов)
- Без монорепы невозможно (либо с костылями)
- Попробуйте как-нибудь положить пакет в пакет



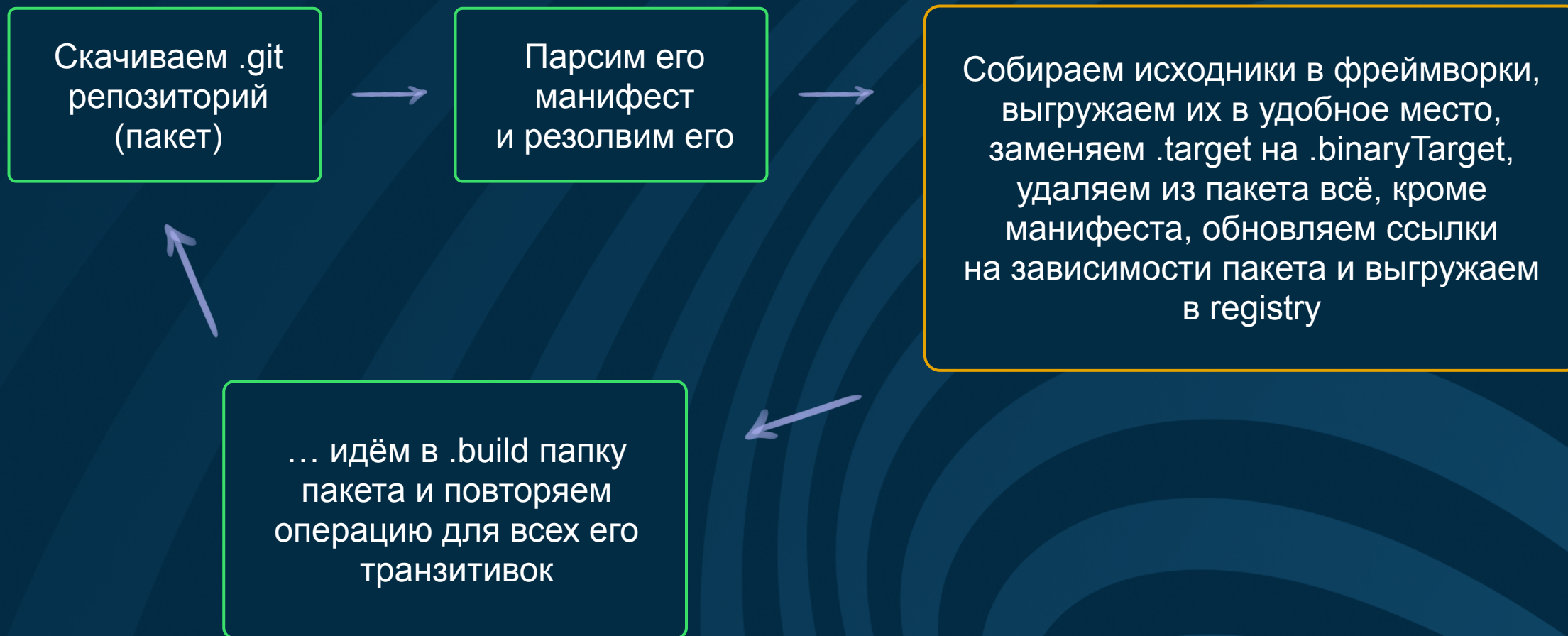
# Как можно использовать registry?

Система кэширования



# Как можно использовать registry?

Система кеширования



# Как можно использовать registry?

Скорость

Система кэширования на пакетах — а зачем?

- Такого просто нет (привет Tuist)
- Никакой обязанности на стороне клиента (проекта) ((Привет Tuist))
- Простая подмена сурсов на бинари / наоборот через зеркала на нужные скоупы
- Можно и на гите, но с registry сильно проще



04



# Заключение

Как оно?



# Заключение

## Общее впечатление

- SPM уже, так или иначе, стал стандартом разработки
  - Registry позволяет решить часть проблем SPM
- Registry позволяет решить проблемы не только самого SPM, но и реальные нужды вашего конкретного проекта
- Registry быстрее



# Заключение

Субъективное впечатление

- Переход на registry в текущем её состоянии должен происходить **ТОЧНО** не за цифры
- Если переходите (особенно не SPM -> SPR, а например Pods -> SPM) — готовьтесь к страданиям





# Полезные материалы

Локальная  
имплементация Swift  
Package Registry



JFrog (Artifactory)  
Documentation



Swift Registry Service  
Open-api





# Спасибо за внимание, вопросы?

Евгений Рыжов  
Senior iOS Developer, Ozon

