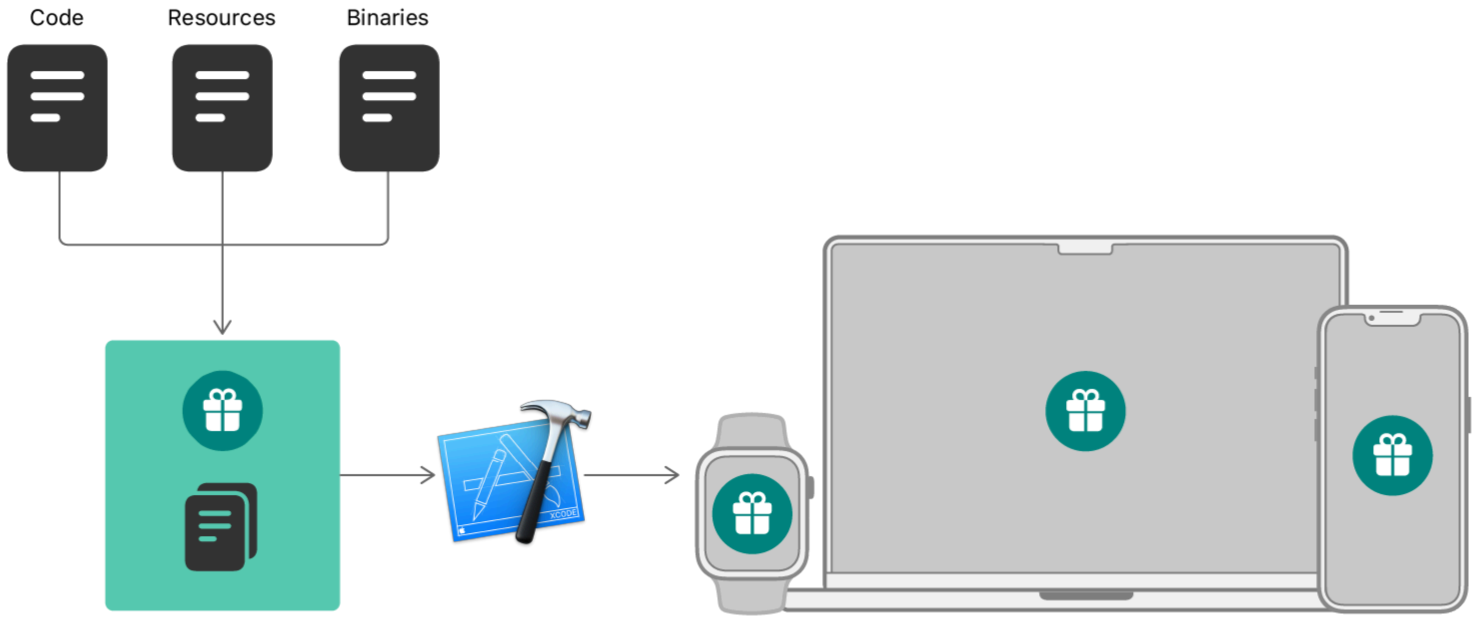
Swift packages

Create reusable code, organize it in a lightweight way, and share it across Xcode projects and with other developers.

Overview



Swift packages are reusable components of Swift, Objective-C, Objective-C++, C, or C++ code that developers can use in their projects. They bundle source files, binaries, and resources in a way that's easy to use in your app's project.

Xcode supports creating and publishing Swift packages, as well as adding, removing, and managing package dependencies. Its support for Swift packages is built on top of the open source Swift Package Manager project.

To learn more about the API you use in your package manifest, see `Package`. To learn more about the Swift Package Manager, see Swift.org and the open source Swift Package Manager repository.

Topics

Package dependencies

Adding package dependencies to your app

Integrate package dependencies to share code between projects, or leverage code from other developers.

Identifying binary dependencies

Find out if a package dependency references a binary and verify the binary’s authenticity.

Editing a package dependency as a local package

Override a package dependency and edit its content by adding it as a local package.

Package creation

Creating a standalone Swift package with Xcode

Bundle executable or shareable code into a standalone Swift package.

Bundling resources with a Swift package

Add resource files to your Swift package and access them in your code.

Localizing package resources

Ensure that your Swift package provides localized resources for many locales.

Distributing binary frameworks as Swift packages

Make binaries available to other developers by creating Swift packages that include one or more XCFrameworks.

Developing a Swift package in tandem with an app

Add your published Swift package as a local package to your app’s project and develop the package and the app in tandem.

Organizing your code with local packages

Simplify maintenance, promote modularity, and encourage reuse by organizing your app’s code into local Swift packages.

PackageDescription

Create reusable code, organize it in a lightweight way, and share it across your projects and with other developers.

Package distribution

Publishing a Swift package with Xcode

Publish your Swift package privately, or share it globally with other developers.

Continuous integration

Building Swift packages or apps that use them in continuous integration workflows

Build Swift packages with an existing continuous integration setup and prepare apps that consume package dependencies within an existing CI pipeline.

See Also

Code

Source Editor

Edit your source files, locate issues, and make necessary changes using the Source Editor.

Bundles and frameworks

Organize code and resources in bundles and frameworks.