Building and testing Swift

In this article

You can create a continuous integration (CI) workflow to build and test your Swift project.

> **Note:** GitHub-hosted runners are not currently supported on GitHub Enterprise Server. You can see more information about planned future support on the [GitHub public roadmap](#).

Introduction 🔗

This guide shows you how to build and test a Swift package.

GitHub-hosted runners have a tools cache with preinstalled software, and the Ubuntu and macOS runners include the dependencies for building Swift packages. For a full list of up-to-date software and the preinstalled versions of Swift and Xcode, see "[About GitHub-hosted runners](#)."

Prerequisites 🔗

You should already be familiar with YAML syntax and how it's used with GitHub Actions. For more information, see "[Workflow syntax for GitHub Actions](#)."

We recommend that you have a basic understanding of Swift packages. For more information, see "[Swift Packages](#)" in the Apple developer documentation.

Using the Swift starter workflow 🔗

GitHub provides a Swift starter workflow that should work for most Swift projects, and this guide includes examples that show you how to customize this starter workflow. For more information, see the [Swift starter workflow](#).

To get started quickly, add the starter workflow to the `.github/workflows` directory of your repository.

YAML

```
name: Swift

on: [push]

jobs:
  build:

    runs-on: macos-latest

    steps:
      - uses: actions/checkout@v2
      - name: Build
        run: swift build
      - name: Run tests
        run: swift test
```

Specifying a Swift version ⧉

To use a specific preinstalled version of Swift on a GitHub-hosted runner, use the `swift-actions/setup-swift` action. This action finds a specific version of Swift from the tools cache on the runner and adds the necessary binaries to `PATH`. These changes will persist for the remainder of a job. For more information, see the [swift-actions/setup-swift](#) action.

If you are using a self-hosted runner, you must install your desired Swift versions and add them to `PATH`.

The examples below demonstrate using the `swift-actions/setup-swift` action.

Using multiple Swift versions ⧉

You can configure your job to use multiple versions of Swift in a matrix.

YAML

```
# This workflow uses actions that are not certified by GitHub.
# They are provided by a third-party and are governed by
# separate terms of service, privacy policy, and support
# documentation.

# GitHub recommends pinning actions to a commit SHA.
# To get a newer version, you will need to update the SHA.
# You can also reference a tag or branch, but the action may change without warning

name: Swift

on: [push]

jobs:
  build:
    name: Swift ${{ matrix.swift }} on ${{ matrix.os }}
    strategy:
      matrix:
        os: [ubuntu-latest, macos-latest]
        swift: ["5.2", "5.3"]
    runs-on: ${{ matrix.os }}
    steps:
      - uses: swift-actions/setup-swift@65540b95f51493d65f5e59e97dcef9629ddf11bf
        with:
          swift-version: ${{ matrix.swift }}
      - uses: actions/checkout@v2
      - name: Build
        run: swift build
```

```
    - name: Run tests
      run: swift test
```

Using a single specific Swift version 🔗

You can configure your job to use a single specific version of Swift, such as `5.3.3`.

YAML

```yaml
steps:
  - uses: swift-actions/setup-swift@65540b95f51493d65f5e59e97dcef9629ddf11bf
    with:
      swift-version: "5.3.3"
  - name: Get swift version
    run: swift --version # Swift 5.3.3
```

Building and testing your code 🔗

You can use the same commands that you use locally to build and test your code using Swift. This example demonstrates how to use `swift build` and `swift test` in a job:

YAML

```yaml
steps:
  - uses: actions/checkout@v2
  - uses: swift-actions/setup-swift@65540b95f51493d65f5e59e97dcef9629ddf11bf
    with:
      swift-version: "5.3.3"
  - name: Build
    run: swift build
  - name: Run tests
    run: swift test
```

Legal

Terms   Privacy   Status   Pricing   Expert services   Blog