Swift Pathway

Get started with this easy-to-navigate collection of videos, documentation, and tools to build great apps and games.

A Swift overview
Learn the basics
Structure your code
Built for Swift
Go further

A Swift overview

The Swift programming language is approachable, safe, fast, and powerful. It's also backed by an extensive open source community that has one goal — to make Swift the best general-purpose programming language in the world.

Developed by Apple and announced at the Worldwide Developer Conference in 2014, Swift was designed to be a safe, approachable, high performance general purpose programming language. These goals are achieved through various features including Swift's modern and expressive syntax, type-safety system, and interoperability with C, C++, and Objective-C code.

In 2015 Apple announced that Swift would be released under an open source license, allowing developers outside of Apple to contribute to the language's growth and development. This announcement helped Swift grow and evolve in the coming years, and it quickly became the preferred language for developing for Apple platforms.

Collaboration between Apple and the open source community has also enabled Swift to expand beyond Apple, with added support for different tools and platforms. This expansion into more and more use cases promotes a growing community of diverse developers and contributors all solving different problems, which in turn benefits all Swift language users. Overall, Swift has become a powerful and flexible programming language that developers can use to create modern apps for Apple platforms and beyond.

To learn more about Swift's open source community, read the Community Overview on swift.org.

Learn the basics

Now that you've covered a brief overview of Swift's history and community, it's time to take a look at the features of Swift. Although this pathway is focused on developing for Apple platforms, the foundations here apply to writing Swift code on any platform.

To get started with Swift, you can begin by reading "A Swift Tour" or watching the video session of the same name. These resources give an overview of the features and syntax of Swift, and will act as your introduction to the Swift programming language guide. The guide is the definitive source of information on Swift and all of its features, and is a great reference manual to come back to as you continue on your learning journey. In this pathway there are callouts to specific chapters in the guide that are especially helpful to read when first approaching Swift.

A Swift Tour: Explore Swift's features and design

A Swift Tour (Article)

The Swift programming language guide

After taking a tour of Swift's features and design philosophy, you can dive a little deeper into each of the topics that were covered. Check out The Basics for a further look into the types of data you'll work with in Swift, from constants and variables, to more advanced types like tuples and optional types. Next, visit Collection Types and explore the three primary ways to store collections of values in Swift: arrays, sets, and dictionaries. You'll start with an overview of each type, followed by examples of different ways to create, access, and modify the values they contain.

The Basics

Type Safety and Type Inference

Collection Types

Then you can explore more of the unique ways that you can manage control flow when developing with Swift. For example, pattern matching capabilities provide powerful and expressive ways to construct your code path, while Swift's handling of optional types can reduce the risk of errors at runtime. Swift also provides a lightweight approach to error handling that integrates seamlessly with its control flow mechanisms. And once you begin to break down and organize your control flow, you'll discover some of the notable features of functions within Swift, and how they contribute to your code's overall safety and performance.

Control flow

Functions

Structure your code

Understanding the building blocks available to you when structuring your code and modeling your data is essential to ensuring your app's success. An organized codebase will improve readability, ensure consistency, allow for easier testing, and remove future frustrations for you and any collaborators.

To begin, read an overview of value and reference types to learn more about how these different types behave. Understanding these behaviors is crucial to picking the right models for each situation. Then, spend some time learning more about three fundamental constructs for organizing your data — Structures, classes, and enumerations (enums). Though all three are used to organize data, it's important to grasp the differences in their behaviors and use cases to ensure the efficacy of your data model. You can also read *Choosing between structures and classes* to explore best practices and gain a better understanding for how and when to use a specific model.

Value and reference types

Structures and classes

Choosing between structures and classes

Enumerations

Built for Swift

If you're looking to jump directly into designing and building apps for Apple platforms, we have multiple frameworks designed specifically to harness the power and safety of Swift. SwiftUI and SwiftData offer an even more approachable way to build stunning and powerful apps, all while writing far less code.

SwiftUI

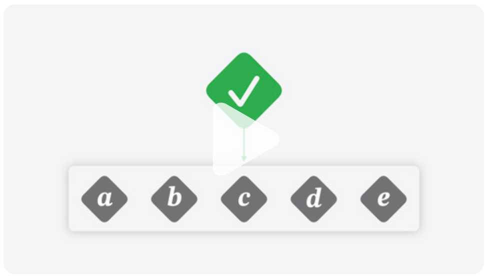
With SwiftUI, you have everything you need to jump right into developing great apps for Apple platforms. The approachable declarative syntax allows you to write and understand code more easily, while live previews in Xcode empower you to iterate rapidly while you view your changes in real time. To get started, you can dive into the SwiftUI pathway and experience how quickly you can bring your ideas to life.

SwiftUI pathway

Swift Testing

Built for Swift from the ground up, Swift Testing is a new framework with expressive APIs that make it easy to write tests. Swift Testing uses macros like `#expect` to capture complex expressions and provide rich, detailed output on your test results. It also includes features like *parameterization* to easily run the same test over a series of values, and *tagging* to selectively run tests based on specified criteria. And test results really shine in the Xcode 16 test navigator.

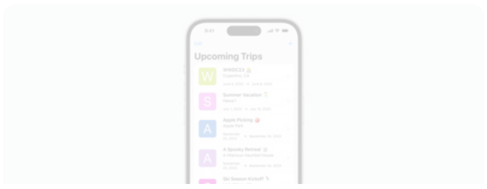
SwiftUI Essentials



Go further with Swift Testing

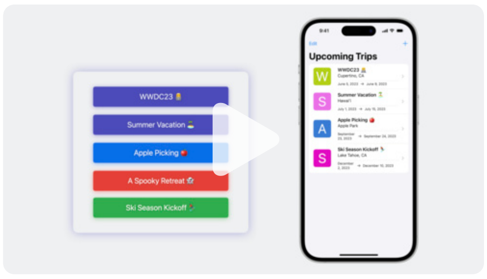
SwiftData

SwiftData is Apple's framework for data modeling and management. Learn how you can utilize this framework to help you persist data in your app, and create a clear and efficient data model. You can also find resources on how to migrate your current app(s) to SwiftData, and tutorials to get started with data modeling.

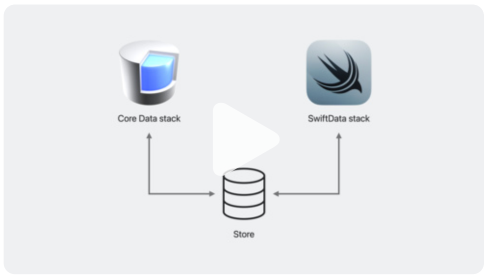


Meet SwiftData

Model your schema with SwiftData



Dive deeper into SwiftData



Migrate to SwiftData

Related documentation

[Tutorial: Welcome to data modeling]

Go further

You've covered a lot — From uncovering how Swift handles common data types, to learning about unique control flow capabilities, and exploring helpful resources for modeling complex data structures. The concepts and skills you've learned lay a strong foundation for developing with Swift. Beyond these foundations, Swift has even more features that will help you make your codebase more flexible, manageable, and powerful. Explore the latest updates to Swift, as well as some advanced features to take your app to the next level.

Protocols

Swift protocols define functionality and characteristics that multiple types can adopt, and can make your codebase more flexible, modular, and reusable.

Protocols

Design protocol interfaces in Swift

Adopting common protocols

Generics

Swift generics allow you to write flexible and reusable code that can operate on different types, without first specifying those types. This flexibility can reduce duplications in your codebase, which can improve clarity, performance, and overall ease of maintenance.

Generics

Embrace Swift generics

Concurrency

Swift concurrency provides you with powerful tools for writing asynchronous and concurrent code that can improve the performance and responsiveness of your app. And the Swift 6 language mode makes concurrent programming dramatically easier by diagnosing data races at compile time, helping you find and fix bugs faster.

- Concurrency
- Meet async/await in Swift
- Explore structured concurrency in Swift
- Visualize and optimize Swift concurrency
- Migrate your app to Swift 6

Macros

Macros allow you to reduce time spent writing repetitive code and adopt complex features more easily, and you will undoubtedly encounter them when developing with Swift. Learn how you can integrate macros into your codebase to make it more powerful and expressive.

- Expand on Swift macros
- Applying macros
- Write Swift macros

Embedded Swift

Embedded Swift brings the power and safety of Swift to constrained environments such as microcontrollers, using a tiny memory footprint with no runtime.

- Go small with Embedded Swift

Run Swift on your server

Swift is a great choice for writing modern, efficient, and secure server-side code. With features like memory safety and interoperability, along with expanding ecosystem support, Swift allows developers to build robust services.

- Learn more
- Get started with Swift on Server
- Explore the Swift on Server ecosystem

Discover more

Pathways

Explore more Apple platforms and technologies.

Learn more ›

Meet with Apple

Join us for sessions, workshops, labs, and appointments — tailored for you.

Learn more ›

Apple Developer Centers

The home for in-person events and activities around the world.

Learn more ›

WWDC highlights

Apple's biggest event of the year for developers.

Learn more ›

Developer Swift Get Started

Platforms

iOS
iPadOS
macOS
tvOS
visionOS
watchOS

Tools

Swift
SwiftUI
Swift Playground
TestFlight
Xcode
Xcode Cloud
SF Symbols

Topics & Technologies

Accessibility
Accessories
App Extensions
App Store
Audio & Video
Augmented Reality
Design
Distribution
Education
Fonts
Games
Health & Fitness
In-App Purchase
Localization
Maps & Location
Machine Learning
Open Source
Security
Safari & Web

Resources

Documentation
Tutorials
Downloads
Forums
Videos

Support

Support Articles
Contact Us
Bug Reporting
System Status

Account

Apple Developer
App Store Connect
Certificates, IDs, & Profiles
Feedback Assistant

Programs

Apple Developer Program
Apple Developer Enterprise Program
App Store Small Business Program
MFi Program
News Partner Program
Video Partner Program
Security Bounty Program
Security Research Device Program

Events

Meet with Apple
Apple Developer Centers
App Store Awards
Apple Design Awards
Apple Developer Academies
WWDC

Get the Apple Developer app.

Light Dark Auto

Terms of Use | Privacy Policy | Agreements and Guidelines

English ˅