



How To Learn Swift .org

# Swift

## Structured Learning



**A focussed plan for structured learning.**

Learn faster, build a stronger foundation, get a job, learn UI, learn more about apps and build a more profitable career in iOS by structuring your learning.



## What to Learn and in What Order?

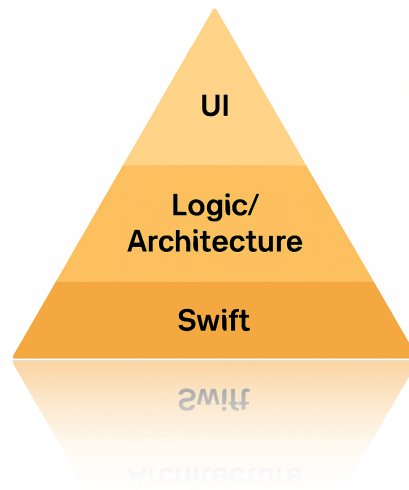
Swift → Architecture → UI

Step 1: **Swift Language Concepts**

Step 2: Architecture & Design Patterns

Step 3: UI + Project Settings

👉 Learn Swift now. Learn UI later. 👉



## Layer Your Learning

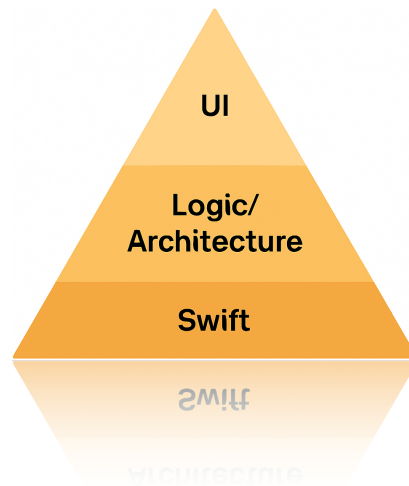
Swift → Architecture → UI

In today's world of mobile app development, the first thing many aspiring developers are drawn to is the user interface. And with good reason—SwiftUI makes it incredibly easy to build beautiful screens with very little code. The instant visual feedback and the elegant syntax give a strong sense of progress and creativity. But there's a downside to this rapid gratification: **it can mask a lack of deeper understanding.**

When you start learning Swift with the UI first, you may find yourself wondering why your view isn't updating, why the app crashes when you navigate between screens, or why passing data feels so confusing. These frustrations often stem from a missing foundation—not knowing how Swift works under the hood. The language has powerful features built into it that, when understood properly, can make your code safer, more scalable, and easier to maintain.

This document is about helping you bridge the gap. Whether you're coming from a background in design, are switching from another programming language, or are just starting your journey with Swift, we'll guide you through why the language itself—not just the UI framework—is the key to mastering iOS development.





## Learn Swift (as a Language) First

Let's talk about what makes Swift different from many other programming languages. Swift emphasizes safety and clarity. Optionals force you to think about the presence or absence of values. Value types (like struct) help you write predictable code that avoids side effects. Protocol-oriented programming allows you to model behavior in flexible and testable ways.

If you ignore these principles and dive straight into building screens, your learning will stall the moment you try to do anything even slightly complex. You might be able to copy code from tutorials, but you won't be able to adapt it to your own ideas. That's where true learning happens—when you're no longer copying and pasting, but creating and debugging with intention.

Spend time learning the basics of the language:

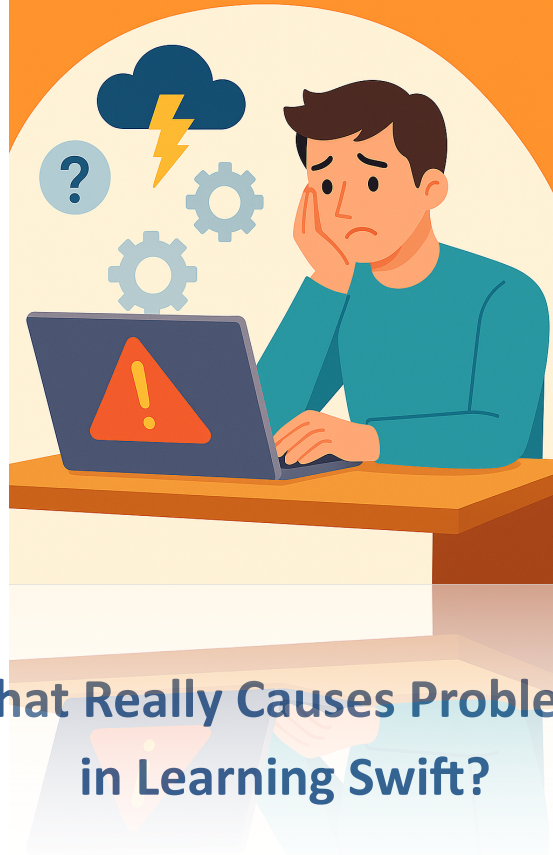
- How does Swift handle memory? What is ARC?
- What's the difference between a class and a struct, and when

should you use each?

- How do closures work, and what makes escaping closures dangerous?
- How can you use enum to model app state in a clean way?

The more deeply you understand these tools, the more capable and confident you'll become as a developer. And Swift is a fantastic language to grow with—it's powerful, expressive, and designed with developer happiness in mind.

## What really causes problems in learning Swift?



## What Really Causes Problems in Learning Swift?

You've probably seen beginner-friendly SwiftUI courses that jump right into building an app. And to be clear—that's not always bad. Motivation and momentum are important. It's rewarding to build something that looks real.

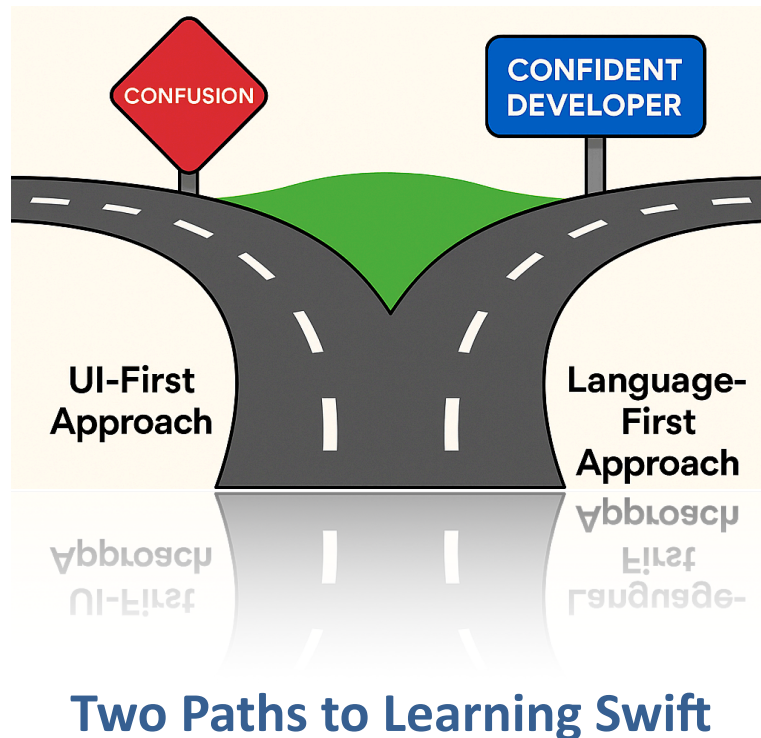
But the UI-first approach comes with risks. It can give you a false sense of security. You might build something that appears functional, but **you don't truly understand the code driving it**. When you try to modify it—add logic, handle errors, or organize your code—it begins to fall apart.

This leads to frustration and burnout. It's one of the main reasons people give up learning Swift. They feel like they've been making progress, but then suddenly everything becomes opaque and intimidating. It's not their fault—they've just skipped the foundation.

A better approach is to start small and go deep. Build a command-line app that fetches data. Create a calculator with input validation. Learn to decode JSON and map it into Swift types before you ever worry about displaying it.

Once you have this solid base, SwiftUI becomes a joy to work with.

You're no longer guessing—you're applying skills you truly understand.



Learning how to build apps means writing computer code, constructing UI (user interface) ontop of that code and bundling all of this together into an app which must be configured correctly.

Starting with how to build an app means you'll be forced to learn the last things first and the first things last. 🧑 🧑 🧑

Employers will employ you to primarily write computer code so make sure you have a good understanding of the language before attending a job interview that will quiz you on it!

If you are starting to learn a musical instrument perhaps **don't book a spot on stage with the orchestra when you don't even know all the chords!** It might be rather embaessing for you!

Think of this statement as you are applying for a job in a competitive market!

## A Better Approach

### A Language-First Approach

Learning Swift should be like learning to play an instrument. You don't start by writing a symphony—you begin with scales, exercises, and simple songs. **These fundamentals build muscle memory and understanding that you can rely on later.**

In the same way, your Swift journey should begin with focused mini-projects:

- Use optionals to build a login form that gracefully handles empty fields
- Use enums to manage game state in a Rock-Paper-Scissors game
- Use structs to model data from a weather API
- Use protocols to decouple your view models from your views

These may not seem exciting at first glance—but they're powerful. You'll learn how Swift thinks. You'll make mistakes in a safe environment. You'll debug, iterate, and improve. And when you finally build that complex app, you'll be ready.

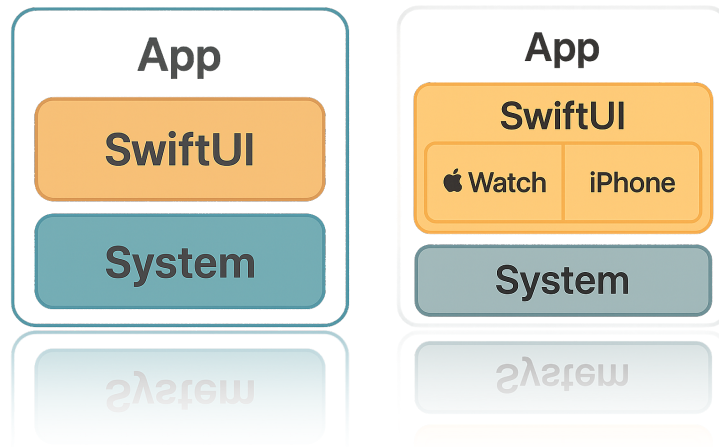


## What Really Causes Problems in Learning Swift?

👉 When you're struggling to write code due to a lack of understanding of the language

The solution is easy; better understand the Swift language.

Swift is at the foundation of your iOS career and learning it properly provides stability and confidence creating better developers.



## App Sourcecode

Each app has its sourcecode and this source of code when compiled together (using Xcode) generates a .app file. This is your app to then be submitted to the AppStore.

The source of code is simply a collection of .swift files that contain the classes, structs and enums that will collectively result in the behaviour programmed for your application.

As you can clearly see from the diagram above, each app source code is seperated into two layers; System and SwiftUI.

**System** is the “layer” (build your systems in layers) that contains the actualy system itself. Therefore, we could remove the UI layer altogether and the system would still function. The UI (User Interface) is a reflection of the current state of the system.

**SwiftUI** is the “layer” that represents the system. This is the UI (User Interface). It contains absolutely no logic what-so-ever



about the system and instead, it displays values from the system and triggers functions made available to the UI.

## UI Separation

To best understand this, think about building an Apple Watch app and an iPhone app. Both apps (two apps) are generated from one source code. This source code will contain just the one layer for the system (the code is unchanged) and two separate isolated entities for each separate device that will interact with the system.



The knowledge and habits you gain from learning Swift will carry over into every programming challenge you face - because you'll have the knowledge to understand it.

Don't settle for surface-level knowledge. Learn the language. **Master the fundamentals. Practice small, then build big.**

If you do, you'll go from being someone who uses Swift to someone who truly knows Swift. And that's where the real opportunities begin.





How To Learn Swift .org

Structured Learning of Swift

Copyright [HowToLearnSwift.org](https://HowToLearnSwift.org)

All rights reserved [HowToLearnSwift.org](https://HowToLearnSwift.org)