

iOS Development with Swift & SwiftUI

Build, test and ship modern iPhone apps with Swift and SwiftUI

Mobile App Development · intermediate · Live online, instructor-led

This syllabus is a draft awaiting instructor review and may change.

A fourteen-week live course in native iOS: Swift, SwiftUI, UINavigationController, Observation, async/await and actors, URLSession and Codable, SwiftData, Keychain, MVVM, testing, accessibility, privacy, Instruments and TestFlight.

Length and schedule

84 hours of live teaching, on either schedule:

- Weekday Intensive (Mon–Fri): about 17 weeks before holidays
- Weekend Evening (Sat & Sun evenings): about 21 weeks before holidays

Fees per student, whole course

- Group: Rs. 14,999
- Duo: Rs. 24,999
- One-to-One: Rs. 44,999

Launch fee — GST treatment to be confirmed.

What you will be able to do

- Write safe, expressive Swift using optionals, value types, protocols, generics and error handling
- Build adaptive, accessible interfaces with SwiftUI
- Manage navigation with UINavigationController and state with @State and @Observable
- Load data with async/await, actors, URLSession and Codable
- Persist data with SwiftData and protect secrets in the Keychain
- Structure apps with MVVM and test them with Swift Testing, XCTest and UI tests
- Handle permissions and privacy manifests, and profile with Instruments
- Sign an app, distribute it through TestFlight and prepare it for App Store review

Before you start

- Basic programming in any language: variables, conditions, loops, functions and simple classes
- No previous Swift or iOS experience required
- A Mac capable of running the current version of Xcode is required: Xcode runs only on macOS, so Windows and Linux laptops cannot be used. Check Apple's published Xcode system requirements before enrolling; 16 GB RAM and about 50 GB of free disk space are recommended
- An iPhone for on-device testing is recommended but not required; the Simulator is used in class
- A free Apple Account is enough for most of the course; publishing to TestFlight needs a paid Apple Developer Program membership, which is optional and paid directly to Apple

Tools

Xcode, Swift, SwiftUI, iOS Simulator, Swift Package Manager, SwiftData, Swift Testing and XCTest,

Instruments, Accessibility Inspector, App Store Connect and TestFlight, Git and GitHub

Syllabus

Module 1: Swift language foundations

Swift syntax, optionals and the difference between value and reference types.

1. Xcode, playgrounds and Swift basics (2 h)
2. Optionals (2 h)
3. Value and reference types (2 h)
4. Swift practice and doubt-clearing (2 h)

Module 2: Swift in depth

Protocols, generics, closures and error handling — the building blocks of Swift APIs.

5. Protocols and extensions (2 h)
6. Generics and closures (2 h)
7. Error handling (2 h)
8. Swift review and doubt-clearing (2 h)

Module 3: SwiftUI and state

Declarative UI with SwiftUI, from layout to observable state.

9. Views, modifiers and layout (2 h)
10. Lists, forms and controls (2 h)
11. Local state with @State and @Binding (2 h)
12. Observable models and the environment (2 h)
13. SwiftUI lab and doubt-clearing (2 h)

Module 4: App lifecycle and navigation

How an iOS app starts and moves between screens.

14. The app lifecycle (2 h)
15. Navigation with UINavigationController (2 h)
16. Tabs, sheets and deep links (2 h)
17. Navigation review and doubt-clearing (2 h)

Module 5: Swift concurrency and networking

Load remote data safely with async/await, actors, URLSession and Codable.

18. async/await and tasks (2 h)
19. Actors and data-race safety (2 h)
20. Networking with URLSession and Codable (2 h)
21. Robust network code (2 h)
22. Networking lab and doubt-clearing (2 h)

Module 6: Persistence and architecture

Store data on device, protect secrets and organise the app with MVVM.

23. Persistence with SwiftData (2 h)
24. Keychain and secure storage (2 h)
25. MVVM and dependency injection (2 h)
26. Architecture review and doubt-clearing (2 h)

Module 7: Testing iOS apps

Prove the app works with unit tests and UI tests.

27. Unit testing with Swift Testing and XCTest (2 h)
28. Testing async code and dependencies (2 h)
29. UI tests with XCUITest (2 h)
30. Testing review and doubt-clearing (2 h)

Module 8: Accessibility, privacy and performance

Make the app usable by everyone, respectful of user data and fast.

31. Accessibility in SwiftUI (2 h)
32. Permissions and privacy manifests (2 h)
33. Performance with Instruments (2 h)
34. Quality review and doubt-clearing (2 h)

Module 9: Signing, TestFlight and App Store preparation

Take an app from Xcode to testers and prepare it for review.

35. Code signing and provisioning (2 h)
36. Archiving and TestFlight (2 h)
37. App Store review preparation (2 h)
38. Release review and doubt-clearing (2 h)

Module 10: Capstone: sports facility booking app

Plan, build and present a complete iOS app from a written brief.

39. Capstone brief and architecture planning (2 h)
40. Capstone build clinic (2 h)
41. Capstone peer review (2 h)
42. Final demos and final assessment (2 h)

Final project

Build and prepare for release an iPhone app for booking slots at a community sports facility. You receive a written brief and a REST API for facilities, available slots and bookings. You deliver:- A Swift and SwiftUI app with facility browsing, a slot picker, booking and cancellation, and a “My bookings” screen, using UINavigationController with value-based navigation and a TabView.- MVVM with @Observable models, dependencies injected through protocols, and clear loading, empty and error states with retry.- Networking with URLSession, async/await and Codable, with cancellation handled correctly and shared state protected by an actor.- SwiftData storage so upcoming bookings are available offline, and the user's session token stored in the Keychain.- Unit tests with Swift Testing for view models and services, and UI tests for the booking journey.- An accessibility pass covering VoiceOver labels, Dynamic Type and contrast, and a calendar-permission flow with a clear purpose string and a complete privacy manifest.- An Instruments note showing one performance issue you measured and fixed.- A signed build archived in Xcode with an App Store review checklist, privacy details and draft metadata; learners with a developer membership also upload it to TestFlight. Reviewed live in the final week: you demo the app in the Simulator or on a device, walk through the architecture and answer questions on your design choices.

Certificate

- At least 80% attendance
- All assignments submitted

- Projects passed
- Final assessment passed (60%)

A ZORK Academy course completion certificate; paying for the course does not by itself earn it.

Next step

Book a free consultation: <https://zorkacademy.online/consultation?course=ios-development>

We help you prepare for work; we do not guarantee jobs, placements or salaries.