

**iOS Development Using Swift and Xcode**

**Course Number:** MBL-134  
**Duration:** 5 days

**Overview**

This Introduction to iOS Development training course teaches attendees how to build iOS native applications for iPhone and iPad using the Swift programming language and Apple's Xcode IDE.

**Note:** This class is taught using the latest version of Swift.

**Prerequisites**

Students should have:

* Experience with an object-oriented programming language such as Java, Objective-C, C#, or C++
* Familiarity with using macOS and an iOS-based device

**Materials**

All iOS training students receive a printed handout that contains all of the material presented as well as step-by-step instructions for each lab exercise.

**Software Needed on Each Student PC**

* Mac running the current or immediately previous version of macOS, with 8 GB RAM or more
* The latest version of Xcode (available for free from the Apple App Store)

**Objectives**

* Develop a thorough understanding of the Swift programming language
* Understand iOS application development architecture
* Gain familiarity with Xcode and other Apple development tools
* Design app UIs using storyboards
* Build a network connected app
* Use best practices for targeting both iPhone and iPad devices

**Outline**

* Intro
  + Swift Versions
  + swift.org
  + Xcode
* Native vs Cross-Platform
* Structure of an App
  + iOS Templates
  + View Controllers
  + Views
  + Storyboards
  + Icons
  + Object Library
  + SwiftUI
  + App Delegate
  + Scene Delegate
* Swift vs JavaScript
* Tour of Xcode
  + Target Settings
  + Navigators
  + Editor Area
  + Inspectors
* Debugging
  + Breakpoints
  + Debug Area
* Classes
  + Single Inheritance
  + Properties: let/var
  + Initializers/deinit
  + Functions
* Structs
  + Similarities to Classes
  + Differences to Classes
* More on Properties
  + wilSet/didSet Observers
  + Computed
* OOP in Swift
* Simulator
* UI Design
  + Controls (Label, Button, Picker, Slider, Stepper, etc.)
  + Views
  + Outlets
  + Actions
  + TextField
* UI Design Comparisons
* Collections
  + Array
  + Set
  + Tuples
  + Dictionary
  + String
  + Operators
* Control Flow
  + Ranges
  + Loops
  + Switch
* UI Design: Auto-Layout
  + Constraints
  + Variants
  + Stack Views
* Constraints in Code
* Swift UI
* Thinking Swifty
  + Error/Throw
  + do/try/catch
  + Enums
  + Optionals
  + Operators & Types
  + Guard
  + Defer
  + Access Control
  + Typealias
* Unit Testing
  + Unit Tests
  + Performance Tests
  + UI Tests
* Patterns
  + Extensions
  + Protocols
  + Delegate
* TableView
  + Table
  + Cells
  + Delegate/Datasource
* More TableView
  + Delete
  + Editing Actions
  + Swipe Actions
* UI Design: Advanced TableView
  + Refresh Control
  + TableView Controller
  + Custom Cells
* Controllers
  + Navigation
  + Segues
  + Tab Bar
* Data
  + Data Class
  + Files
  + UserDefaults
* Closures
  + Higher Order Functions
  + Closures
  + Function Types
* Server Communication with URLSession
* JSON & Codable
  + Decodable
  + Encodable
  + CodingKeys
* UI Interaction
  + Touches
  + Gestures
  + Animation
* Notifications
  + Local
  + Push
  + Handling Notifications
  + NotificationCenter
* Threading
* Frameworks
  + CoreData
  + CoreLocation
* WebView
  + WebKit
  + SafariKit
* App Dev Considerations
  + App States
  + Git
  + Pods
  + Icons
* Localization
* Deployment
* Additional Topics (Optional)
  + Lazy properties/Collections
  + Generics
  + Input Accessory
  + Collection View
  + Image Access
  + Contacts
  + Email
  + Calls
  + SMS
  + LocalAuth/Biometric Authentication
  + Accessibility
  + ObjC/Swift Interoperability
  + Unit Test expectations
  + MVC vs MVVM
  + Protocol Oriented Programming
  + Functional Programming
* Conclusion