iOS 8 Application Development for iPhone and iPad Using Objective-C
(5 Days)

Class Overview
This iOS Development using Objective-C training class teaches how to build iOS 8 native applications for iPhone and iPad using Objective-C and Apple’s Cocoa Touch framework.

Class Goals
- Understand iOS application development architecture.
- Gain familiarity with Xcode 6 and other Apple development tools.
- Learn proper use of techniques and patterns in Objective-C.
- Design app UIs using storyboards.
- Explore techniques for custom drawing and animation.
- Store data locally using Core Data and SQLite.
- Build apps that communicate with web services.
- Learn about iOS security services and sandboxing.
- Use best practices for targeting both iPhone and iPad devices.

Prerequisites
Experience in the following areas is required:
- Substantial previous programming experience using an object-oriented C-based language such as C#, Java, or C++.

Courses that can help you meet these prerequisites: Objective-C Training for iOS Development

Class Outline

<table>
<thead>
<tr>
<th>Introduction</th>
<th>Asset Management</th>
</tr>
</thead>
<tbody>
<tr>
<td>iPhone and iPad Device Anatomy</td>
<td>XCTest Testing Framework</td>
</tr>
<tr>
<td>iOS Architecture and SDK Frameworks</td>
<td>Continuous Integration and Bots</td>
</tr>
<tr>
<td>iOS and SDK Version Compatibility</td>
<td>Automatic Configuration</td>
</tr>
<tr>
<td>Apple iOS Developer Program</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>New APIs and Service in iOS 8</th>
<th>Objective-C for Experienced Programmers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Touch ID</td>
<td>Classes, Objects, and Methods</td>
</tr>
<tr>
<td>PhotoKit</td>
<td>Declared Properties</td>
</tr>
<tr>
<td>HealthKit</td>
<td>Memory Management</td>
</tr>
<tr>
<td>HomeKit</td>
<td>Automatic Reference Counting (ARC)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Xcode 6</th>
<th>Categories and Extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tour of the IDE</td>
<td>Formal and Informal Protocols</td>
</tr>
<tr>
<td>Templates, Projects, and Workspaces</td>
<td>Blocks</td>
</tr>
<tr>
<td>Creating a New Project</td>
<td></td>
</tr>
<tr>
<td>LLVM and LLDB</td>
<td></td>
</tr>
<tr>
<td>Debug Gauges</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Application Patterns and Architecture</th>
<th>Model View Controller (MVC)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IBOutlets and IBActions</td>
</tr>
<tr>
<td></td>
<td>Subclassing and Delegation</td>
</tr>
</tbody>
</table>
Views and Windows
The View Hierarchy
- Containers
- Controls
- Text and Web Views
- Navigation View and Tab Bars
- Alert Views and Action Sheets
- Controlling Rotation Behavior
- View Autosizing
- Autolayout

Storyboards
Adding Scenes
- Segues
- Transitions
- Using in a Tab Bar Application

Table Views
Static and Dynamic Table Views
- Delegates and DataSources
- Table View Styles
- Custom Cells

Navigation Based Applications
Adding the Root View Controller
- Creating the Navigation Controller
- Controlling the Stack Navigation Programmatically

UIPickerView and UIDatePicker
Designing the UI
- Coding for the Data Picker
- Hiding the Keyboard
- Memory Management

Directories and Files
NSFileManager, NSFileHandle, and NSData
- Problems Solved by ADO.NET Entity Framework
- Pathnames in Objective-C
- Working with Directories
- Working with Files
- Reading and Writing from a File
- iCloud
- Key-Value Data
- Archiving

CloudKit

Authentication
- Private and Public Databases
- Structured and Asset Storage

Working with Data
SQLite Integration
- Using SQLite Directly
- Overview of Core Data
- Managed Objects
- Persistent Store Coordinator
- Entity Descriptions
- Retrieving and Modifying Data

Multitouch, Taps, and Gestures
The Responder Chain
- Touch Notification Methods
- Enabling Multitouch on the View
- Gesture Motions
- Gesture Recognizers

Drawing
Core Graphics and Quartz 2D
- Lines, Paths, and Shapes

Animation
- Core Animation Blocks
- Animation Curves
- Transformations
- SpriteKit
- SceneKit
- Metal

Multitasking
Application States
- Background Execution
- Background App Refresh
- State Restoration

Notifications
Local Notifications
- Push Notifications

Core Location Framework
Location Accuracy
- Obtaining Location Information
- Calculating Distances
MapKit Framework and MKMapView

Concurrency
- Grand Central Dispatch (GCD)
  - Serial and Concurrent Queues
  - Main Dispatch Queue
  - Completion Blocks
  - Operation Queues

Networking
- Reachability
  - Synchronous Downloads
  - Asynchronous Downloads
  - Handling Timeouts
- Sending HTTP GET and POST Requests
- Parsing JSON
- Parsing XML
- AirDrop

Handoff
- Interactions
  - App Framework Support
  - Implementing Handoff
  - Continuation Streams
  - Best Practices

Targeting Multiple Devices
- iPhone vs. iPad
  - Universal Apps
  - Multiple SDK Support
  - Detecting Device Capabilities
  - Supporting Multiple iOS Versions

Localization

Resources
- Language and Region
- NSLocale
- Text
- Dates
- Numbers

App Extensions
- Extension Types
  - Creating an Extension
  - Common Scenarios

Running on a Physical Device
- Development Certificates
  - Assigning Devices
  - Creating an App ID
  - Provisioning Profiles
  - Running

Performance and Power Optimization
- Measuring Performance
  - Instruments
  - Responsiveness
  - Memory Usage, Spikes, and Leaks
  - Networking and Power

Deployment
- Icons and Launch Images
  - Distribution Certificates
  - Distribution Provisioning Profiles
  - Archiving an Application
  - App Store Distribution
  - AdHoc and Enterprise Distribution
  - iTunes Connect