KD-40539

Build iOS Apps with C# and .NET using the Xamarin Tools for Visual Studio

www.ked.com.mx

Av. Revolución No. 374 Col. San Pedro de los Pinos, C.P. 03800, México, CDMX. Tel.: 55 52 78 55 60

About this course.

During this course, C# .NET developers will learn how to build iOS applications using Visual Studio and Xamarin. Topics include the basics of mobile development, storing, retrieving, and displaying data in iOS, and building a responsive, native UI using the Xamarin.iOS designer to publish an app in the App store.

Length.

5 Days.

Audience profile.

This course is intended for developers with at least basic C# knowledge with 6-12 months of .NET experience. Students should own and/or have used an iOS device and be familiar with the paradigms of the platform.

Hardware/Software:

- A Mac running the latest OS, with Visual Studio for Mac and XCode installed.
- For Windows development (optional): a Windows 10 PC with Visual Studio 2017 and the Mobile development with .NET workload installed
- An iOS device is highly recommended.

Prerequisites.

Before attending this course, students must have:

- Basic knowledge of C# and six months of experience developing using .NET.

At course completion.

After completing this course, students will be able to:

- Build iOS applications with C# .NET using the Xamarin tools for Visual Studio.
- Design multiscreen applications using the Storyboard designer and choose the appropriate navigation for larger applications.
- Load and visualize data including accessing REST services and local SQLite databases.
- Customize the visual presentation for collections of data using the ListView.

Exam.

None.

Course outline.

Module 1: Getting Started with Xamarin (XAM101).

By the end of this module, you will be able to choose the Xamarin approach that is right for you. You will also set up your development machine and run an app on Windows, the iOS simulator, and an Android emulator.

- Define the state of mobile technology today.
- Discuss mobile app trends.
- Identify approaches to development.
- Discover the Xamarin Approach.
- Set up your development environment.

Labs: Building Cross Platform Applications with Xamarin.

- Demonstration - Cross Platform Applications with Xamarin.
- Demonstration – View a Xamarin.Forms Project.
- Setup Validation – Validate your development environment ready.

Module 2: Introduction to Xamarin.iOS (IOS101).

This module takes you through the entire development process. You will create a new project in Visual Studio, code the user interface and behavior, build the project, and test it on an iPhone simulator. As you build your app, you will learn several iOS design patterns such as Model-View-Controller, protocols, and delegates.

- Introduce the development tools.
- (De)constructing the application.
- Add views and behavior.

Labs: Create a Tip Calculator for iOS in Xamarin.

- Group Exercise: Create and run your first iOS application.
- Add a root view controller to the Tip Calculator.
- Create the UI for the Tip Calculator.
- Add logic to your Tip Calculator.

Module 3: Using the Xamarin.iOS Designer (IOS102).

Use the Xamarin.iOS Storyboard Designer to design a responsive, multi-screen application.

- Create a single screen application.
- Describe and use Auto Layout.
- Interact with designer-defined views programmatically.
- Navigate between view controllers.

Labs: Use the iOS Designer to create a multi-screen application.

- Create the UI for a single view application.
- Add constraints to the fireworks app.
- Add a second screen to your app and code a button to navigate to it.
- Add segues to define the navigation.

Module 4: TableViews in iOS (IOS110).

During this module you will learn to use the UITableView to display a collection in your Xamarin.iOS app. You will use the built-in styles to visualize your data, learn to handle item-selection events and navigate the app to a new page. Finally, you will see how to configure cell reuse to reduce memory usage and boost performance.

- Explore Table Views.
- Utilize built-in cell styles.
- Add selection behavior.
- Implement cell reuse.

Labs: Use a Table View, populate it with data and perform basic customizations.

- Add a Table View to an application.
- Populate a Table View.

- Use built-in cell styles.
- Use the accessory styles and row selection.
- Implement cell reuse.

Module 5: Customizing Table Views (IOS115).

During this module you will define a custom cell manually in code and graphically using the UI Designer. You will also see how to display grouped data with headers/footers that delimit each group and an index to make the groups easy to navigate.

- Customize table view cells in code.
- Customize table view cells in the designer.
- Group data in the table view.

Labs: Customize table cells.

- Customize a default table cell.
- Create a custom table view cell in code.
- Create a prototype table view cell using the designer.
- Create static cells in the designer.
- Create a grouped table with an index.

Module 6: Consuming REST-based Web Services (XAM150).

In this module, you will learn to consume REST-based web services with HttpClient using both the managed networking stack and the platform-specific handlers that boost performance. You will also learn some common strategies for dealing with the unique challenges that mobile devices face when communicating over the network.

- Obtain the device's network capabilities.
- Introduce REST.
- Consume REST services with Xamarin.

Labs: Use, consume and communicate with REST services.

- Determine the network connectivity.
- Communicate with a Book Service.
- Demonstration: Leverage the native platform network stack.

Module 7: SQLite and Mobile Data (XAM160).

During this module you will learn to identify the proper location for your database file and how to insert, update, retrieve, and delete data efficiently using asynchronous I/O calls.

- Choose a data storage strategy.
- Store data locally with SQLite.
- Use SQLite asynchronously.

Labs: Utilize SQLite to store and access data.

- Determine your database file path.
- Add SQLite.Net to your projects.
- Access a SQLite database with SQLite.Net.
- Access a SQLite database using asynchronous methods.

Module 8: Navigation Patterns (IOS205).

This course shows you how to implement three iOS navigation patterns: stack, tab, and master-detail. It also includes some guidelines to help you decide which pattern is appropriate for your app.

- Progress through pages of data with stack-based navigation.
- Show different views of related data with tab navigation.
- Display hierarchical relationships with master/detail navigation.

Labs: Implement Stack, Tab and Master/Detail navigation.

- Implement Stack Navigation.
- Implement Tab Navigation.
- Implement Master/Detail.

Module 9: Backgrounding Modes and File Transfers in iOS (IOS210).

This module shows you how to register for notifications as your app transitions between the foreground and background and how to use the Finite-length Task API to ensure you have enough time to save application state.

- Understand the iOS Backgrounding Model.
- Work with Finite-Length Tasks.

Labs: Ensure important operations complete by allowing them to run in the background.

- Integrate a Finite-Length Task with the application lifecycle.
- Code a cancellable Finite-Length Task.

Module 10: Editing TableViews (IOS215).

This module shows you how to register for notifications as your app transitions between the foreground and background and how to use the Finite-length Task API to ensure you have enough time to save application state.

- Work with built-in editing operations.
- Add support for modern editing operations.
- Integrate a search bar.

Labs: Create a Table view with advanced features.

- Turn on editing features on a Table View.
- Committing editing operations.
- Add support for swipe-gesture edit actions.
- Add support for the "pull-to-refresh" gesture.
- Add search support to a Table View Controller.

Module 11: Maps and Location (IOS230).

In this module you will use MKMapView to integrate a map into your UI. You will determine the user's location, set the view port of the map, and add annotations to the display.

- Explore iOS mapping options.
- Use location data.
- Adjust the viewport of the map.
- Add annotations to the map.

Labs: Create an iOS map with markers, viewports and annotations.

- Group Exercise: Add a map and set the properties.
- Show the device's current location.
- Changing the map's viewport with the camera.
- Add an annotation to your map.

Module 12: Touch and Gestures (IOS240).

In this module, you will use low-level touch events to create a multi-touch drawing application. You will also detect standard gestures to translate, rotate, and zoom an image.

- Respond to touch events.
- Handle multi-touch events.
- Utilize gestures.

Labs: Implementing Automation.

- Group Exercise: Override touch methods.
- Drag and snap.
- Create the Xam Paint app.
- Add pan gestures.
- Add Multiple gestures to add scale and rotation.

Module 13: AutoLayout in Xamarin.IOS (IOS300).

This module shows you how to use Constraints and Size Classes to define a flexible UI. iOS uses your code and the runtime device's screen properties to calculate the size and position of your views.

- Create adaptive UIs using the iOS Designer.
- Create and update constraints programmatically.
- Animate constraint changes.
- Use Size Classes to customize your UI for different screen sizes.

Labs: Create a responsive UI using Size Classes and constraints.

- Use Auto Layout in the Designer.
- Update Constraints Programmatically.
- Animate Constraints.
- Update your UI based on the Size Class.

Module 14: Diagnosing Memory Issues (XAM370).

In this module you will learn how memory leaks happen in managed memory even with a sophisticated garbage collector and how to discover and fix them. It then shows you several memory issues that are specific to Xamarin.iOS.

- Identify and fix memory leaks in your code.
- Recognize and fix Xamarin.iOS specific memory problems.
- Recognize and fix Xamarin.Android specific memory problems.

Labs: Track down and correct memory leaks in your Xamarin.Forms app.

- Using the Xamarin Profiler to monitor allocations.
- Finding and fixing delegate reference leaks.

- Identifying and breaking strong reference cycles.
- Watch out for peer promotions.
- Show ListView memory and performance with a custom adapter.

Module 15: Preparing for Publishing (XAM220).

Prepare your app for release through the App Store, Google Play, and Windows Marketplace.

- Getting ready to publish your app.
- Understanding publishing styles.
- Publishing to a store.

Module 16: iOS Publishing (IOS220).

Build your app and submit it to the App Store.

- Prepare an application for publishing.
- Sign an application.
- Publish an app to the App Store.

Labs: Implementing Automation.

- Configuring Automation accounts.
- Creating runbooks.