Workshop	
Name	**Developing Android & IOS apps with Xamarin**
Learning Objective	This course Leverage your skills in Microsoft Visual Studio and C# to build cross-platformapps that run on both Android and iOS. This course shows you everything you need to get started using Xamarin's integration with Visual Studio to create cross-platformC# apps. It covers the list of tools you need, their setup, architecture, and their use. This course will also take you on an in-depth tour of all of the major features of Xcode. Special attention will be given to those skills that can help you become a more productive Xcode developer.
Duration	**3Days**
Participants' Entry Profile	To get the most out of the course, you should be somewhat familiar with C# Language & .NET Framework
Training Methodology	**The workshop will follow Synergetics methodology of** • Concept Visualization • Active Experimentation • Application Development The workshop will be 100% Hands-On with each participant having access to system during the session.
Setup Requirements	
Hardware and Software Requirements	Participant's as well as Trainer's Machine are required to have : **Hardware / Software** • Intel® Core™2 Duo Processor • 4 GB RAM minimum • 100 GB free in the hard disk • Internet connection • OS: Windows 7 • Web Server: IIS 7 **Development tools** • Visual Studio 2012 Ultimatewith Xamarin • XCode 5 IDE
Training Lab Requirements	Whiteboard 6 feet by 4 feet (minimum) Whiteboard markers – Red, Blue, Green, Black Video Projector (1024 X 768 resolutions)

Course Content

Day 1

- Architecture and Setup
  - Understanding the Challenge
  - . NET on Android/iOS
  - Bringing .NET to Android/iOS
  - Android Tool Experience
  - iOS Tool Experience
  - Development Philosophy
  - Solution Organization

- Creating The Android App
  - Create the Android Project
  - Layout the User Interface
  - Tie User Interface to Code
  - Run the App on the Emulator
  - Add Image to Layout
  - Add Image Handling Code

- iOS Development
  - The iOS Development Experience
  - Installation Requirements
  - Installing Xamarin on OS X
  - Visual Studio/Xamarin.iOS Connectivity
  - Configuring Connectivity
  - Verifying Setup

- Creating the iOS App
  - Add the iOS Project
  - Add UI Designer Support
  - Add Controls to the UI
  - Add Properties for the Controls
  - Tie the User Interface to Code
  - Add Image to Layout
  - Add Images and Image Handling Code

- Creating the Shared Code Library
  - Creating a Portable Class Library
  - Add Basic Data Representation
  - Add Data Management
  - Create Cursor-Like Behavior
  - Connect to Android User Interface
  - Connect to iOS User Interface
  - Providing Visual Feedback from the Library

- Cross-platform Image Management
  - Shared Image Management in the iOS App
  - Shared Image Management in the Android App

- Translating Image Names to Resource Identifiers with Reflection
- Reducing Image Translation Overhead

Day 2

➢ Swipe-Navigation in Android
- Direct Navigation
- Moving to Swipe-Navigation
- Add Random Access
- Create the Fragment
- Support Package and Namespace
- Add Support Library as a Xamarin Component
- Implement FragmentStatePagerAdapter
- Create the New Activity Layout
- Create the Swipe-Navigation Activity

➢ Swipe-Navigation in iOS
- Moving from Direct to Swipe Navigation
- Adding the UIPageViewController
- Creating the UIViewController
- Displaying the first Page
- Adding Navigation Events
- Adding Position Awareness to UIViewController
- Handling Navigation Events
- Using Swipe Navigation
- Adding page Turning effects

➢ Working with iOS Protocols
- What are Protocols
- Hides Protocol Use
- Swipe Navigation with Delegates vs. Protocol
- The Challenge of Protocols in C#
- Implementing Protocols in Xamarin
- Implication of Protocols as Classes
- Implementing Swipe Navigation w/ Protocol

➢ Android Master/Detail Navigation
- Android Master/Detail Experience
- Creating a Master List
- Working with an ArrayAdapter
- Creating a Custom List Adapter
- Implementing List Adapter GetView
- Connecting the Custom List Adapter
- Starting the Detail view
- Passing Intent Extras

➢ Android Navigation Drawer
- Architecture with Navigation Drawer
- Adding Navigation Drawer to the Layout
- Managing Master and Detail Data Together
- Populating the Navigation Drawer Choices

- Customizing the Navigation Drawer List Appearance
- Managing List Selections in the Navigation Drawer
- Making the FragmentStatePagerAdapter Updatable
- Updating the Displayed Details

Day 3

iOS Master/Detail Navigation

- iOS Master/Detail Experience
- iOS UI Options
- Creating a Master Table View Controller
- Creating a Table View Data Source
- Connecting Table View and Data Source
- Adding a Navigation Controller
- Displaying the Detail View Controller
- Passing the Master Selection to the Detail View Controller
- Fixing Project Version Issues
- Adjust Detail View Controller Layout

Introduction to Xcode 5

- What is Xcode
- Compiler
- Interface Builder
- A lap Around Xcode
- Xcode Review
- Organizer
- Simulator
- Instruments

Workspaces and Projects

- Workspaces
- Projects
- Project Navigator
- Adding new Files
- Adding Existing Files
- Snapshots and Source Control

Writing and Navigating Code

- Opening Files
- Navigating Windows
- Finding Code
- Editors and Editor Features
- Navigating Code

Building and Debugging

- Workspaces and Projects
- Build Configurations, Targets and Schemes
- Debugging
- Breakpoints
- Advanced Breakpoints

- Using Interface Builder
  - What is Interface Builder?
  - Interface Builder Canvas
  - Connections and Actions
  - Localization and Storyboarding

- Xcode Organizer
  - Documentation
  - Devices
  - Repositories
  - Projects
  - Archives
  - Distribution
  - AdHoc Deployment