iOS
application development

tutorialspoint
SIMPLYEASYLEARNING

About the Tutorial

iOS is a mobile operating system developed and distributed by Apple Inc. It was originally released in 2007 for the iPhone, iPod Touch, and Apple TV. iOS is derived from OS X, with which it shares the Darwin foundation. iOS is Apple's mobile version of the OS X operating system used in Apple computers.

Audience

This tutorial has been designed for software programmers with a need to understand the iPhone and iPad application development on iOS using Objective C programming.

Prerequisites

Before proceeding with this tutorial, you should have a basic understanding of Computer Programming terminologies. A basic understanding of any of the programming languages, especially Objective C programming language, will help you learn the concepts of iOS programming faster.

Copyright & Disclaimer

Table of Contents

About the Tutorial i

Audience i

Prerequisites i

Copyright & Disclaimer i

Table of Contents ii

1. IOS – GETTING STARTED 1

General Overview 1

Registering as an Apple Developer 2

Apple iOS Developer Program 3

2. IOS – ENVIRONMENT SETUP 5

iOS – Xcode Installation 5

Interface Builder 6

iOS Simulator 6

3. IOS – OBJECTIVE C 8

Interface and Implementation 8

Object Creation 8

Methods 8

Important Data Types in Objective C 9

Printing Logs 10

Control Structures 10

Properties 10

Categories 10

Arrays 11

Dictionary 11

4. IOS – FIRST IPHONE APPLICATION 12

Creating the First App **12**

Code of the First iOS Application **17**

AppDelegate.h **17**

AppDelegate.m **18**

ViewController.h **20**

ViewController.m **21**

5. IOS – ACTIONS AND OUTLETS 22

Actions and Outlets – Steps Involved **22**

6. IOS – DELEGATES 29

Example for Delegate **29**

Steps in Creating a Delegate **29**

7. IOS – UI ELEMENTS 36

What UI Elements are? **36**

How to Add UI Elements? **36**

Our Focus **36**

Our Approach **36**

List of UI Elements **36**

Text Fields **38**

Input Types – TextFields **44**

Buttons **48**

Label **52**

Toolbar **53**

Status Bar **55**

Navigation Bar **57**

Tab Bar 61

Image View 63

Scroll View 65

Table View 68

Split View 74

Text View 84

View Transition 87

Pickers 93

Switches 97

Sliders 99

Alerts 101

Icons 104

8. IOS – ACCELEROMETER 105

Accelerometer – Steps Involved 105

9. IOS – UNIVERSAL APPLICATIONS 107

Universal Application – Steps Involved 107

10. IOS – CAMERA MANAGEMENT 113

Camera Management – Steps Involved 113

11. IOS – LOCATION HANDLING 116

Location Handling – Steps Involved 116

12. IOS – SQLITE DATABASE 121

Steps Involved 121

13. IOS – SENDING EMAIL 130

Steps Involved 130

14. IOS – AUDIO AND VIDEO 134

Steps Involved 134

15. IOS – FILE HANDLING 137

16. IOS – ACCESSING MAPS 139

Steps Involved 139

17. IOS – IN-APP PURCHASE 144

Steps Involved 144

18. IOS – IAD INTEGRATION 152

iAd Integration – Steps Involved 152

19. IOS – GAMEKIT 155

Steps Involved 155

20. IOS – STORYBOARDS 162

Steps Involved 162

21. IOS – AUTO-LAYOUTS 167

22. IOS – TWITTER AND FACEBOOK 173

Steps Involved 173

23. IOS – MEMORY MANAGEMENT 177

Memory Management Issues 177

Memory Management Rules 177

Handling Memory in ARC 177

Memory Management Tools 177

Steps for Analyzing Memory Allocations 177

24. IOS – APPLICATION DEBUGGING 180

Selecting a Debugger 180

How to Find Coding Errors? 180

Set Breakpoints 180

Exception Breakpoint 182

1. iOS – Getting Started

General Overview

iOS, which was previously called iPhone OS, is a mobile operating system developed by Apple Inc. Its first release was in 2007, which included iPhone and iPod Touch. iPad (1st Generation) was released in April 2010 and iPad Mini was released in November 2012.

The iOS devices get evolved quite frequently and from experience, we find that at least one version of iPhone and iPad is launched every year. Now, we have iphone5 launched which has its predecessors starting from iPhone, iPhone 3gs, iPhone 4, iPhone 4s. Similarly, iPad has evolved from iPad (1st Generation) to iPad (4th Generation) and an additional iPad Mini version.

The iOS SDK has evolved from 1.0 to 6.0. iOS 6.0, the latest SDK is the only officially supported version in Xcode 4.5 and higher. We have a rich Apple documentation and we can find which methods and libraries can be used based on our deployment target. In the current version of Xcode, we'll be able to choose between deployment targets of iOS 4.3, 5.0 and 6.0.

The power of iOS can be felt with some of the following features provided as a part of the device.

- Maps
- Siri
- Facebook and Twitter
- Multi-Touch
- Accelerometer
- GPS
- High end processor
- Camera
- Safari
- Powerful APIs
- Game center
- In-App Purchase
- Reminders
- Wide Range of gestures

The number of users using iPhone/iPad has increased a great deal. This creates the opportunity for developers to make money by creating applications for iPhone and iPad the Apple's App Store.

For some one new to iOS, Apple has designed an application store where the user can buy apps developed for their iOS devices. A developer can create both free and paid apps to App Store. To develop applications and distribute to the store, the developer will require to register with iOS developer program which costs $99 a year and a Mac with Mountain Lion or higher for its development with latest Xcode.

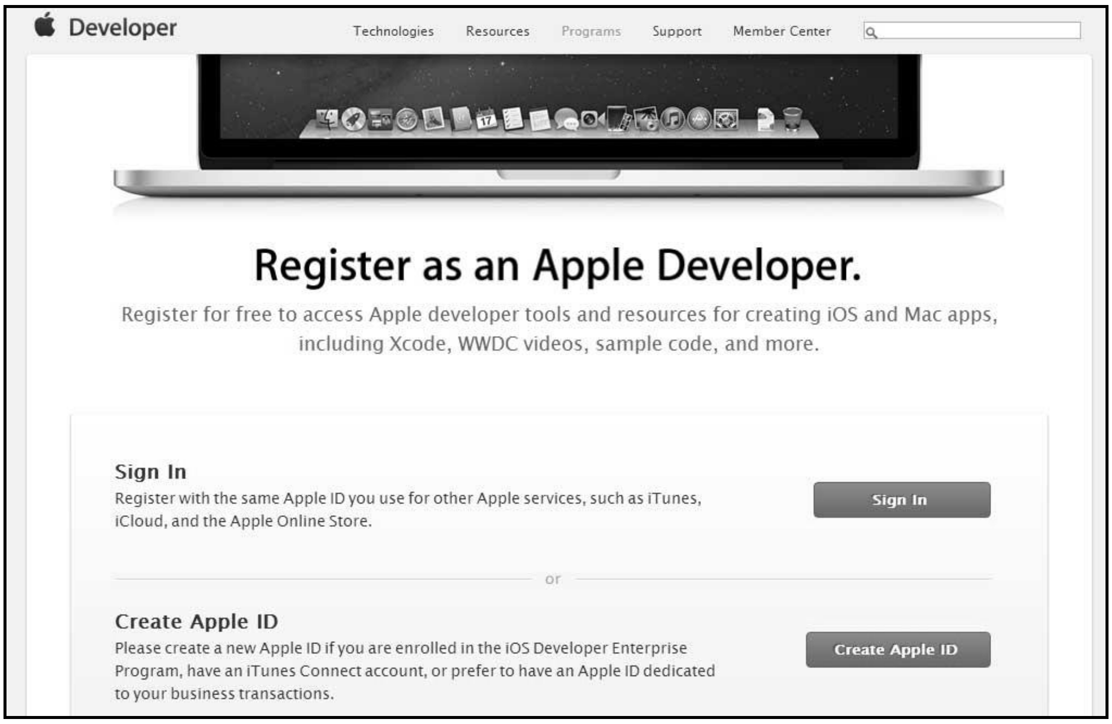
Registering as an Apple Developer

An Apple ID is most necessary if you are having any Apple device and being a developer, you definitely need it. It's free and hence, no issues in having one. The benefits of having an Apple account are as follows:

- Access to development tools.
- Worldwide Developers Conference (WWDC) videos.
- Can join iOS developer program teams when invited.

To register an Apple account, follow the steps given below:

1. Click the link (https://developer.apple.com/programs/register/) and select "Create Apple ID".



Register as an Apple Developer.

Register for free to access Apple developer tools and resources for creating iOS and Mac apps, including Xcode, WWDC videos, sample code, and more.

Sign In

Register with the same Apple ID you use for other Apple services, such as iTunes, iCloud, and the Apple Online Store.

Create Apple ID

Please create a new Apple ID if you are enrolled in the iOS Developer Enterprise Program, have an iTunes Connect account, or prefer to have an Apple ID dedicated to your business transactions.

2. Provide the necessary information, which is self explanatory as given in the page.
3. Verify your account with your email verification and the account becomes active.
4. Now you will be able to download the developer tools like Xcode, which is packaged with iOS simulator and iOS SDK, and other developer resources.

Apple iOS Developer Program

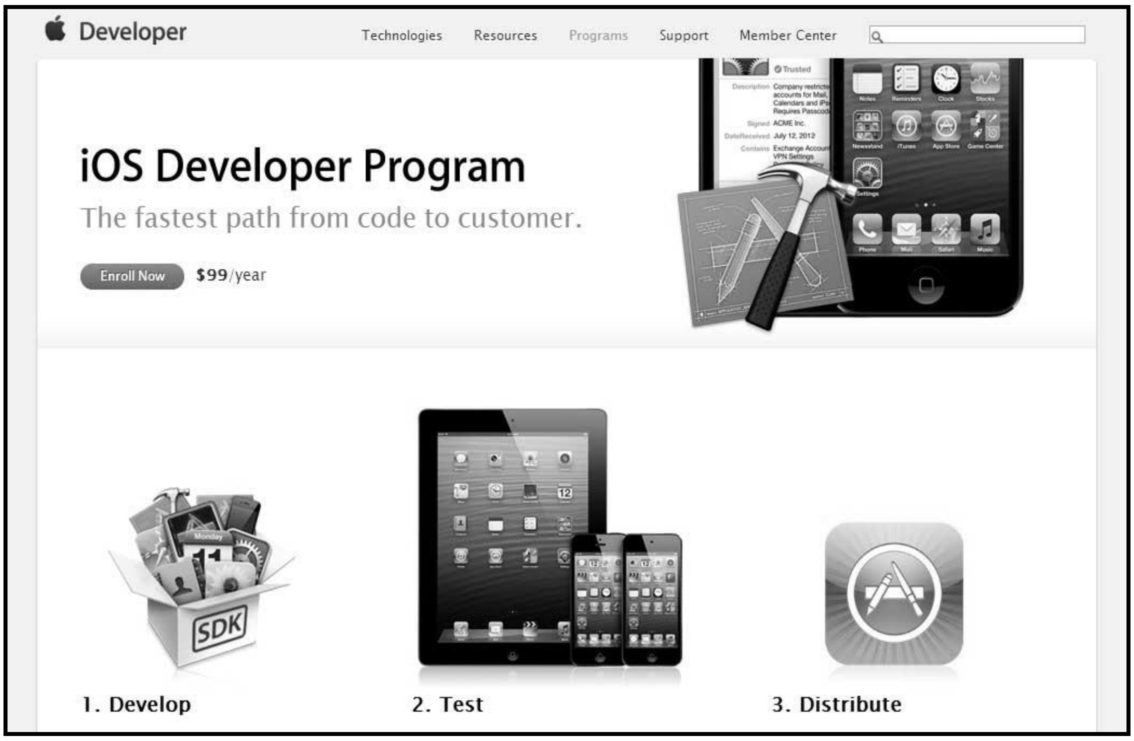
The first question that would arise to a new developer is – Why should I register for an iOS developer program? The answer is quite simple; Apple always focuses on providing quality applications to its user. If there was no registration fee, there could be a possibility of junk apps being uploaded that could cause problems for the app review team of Apple.

The benefits of joining the iOS developer program are as follows:

- Run the apps you develop on the real iOS device.
- Distribute the apps to the app store.
- Get access to the developer previews.

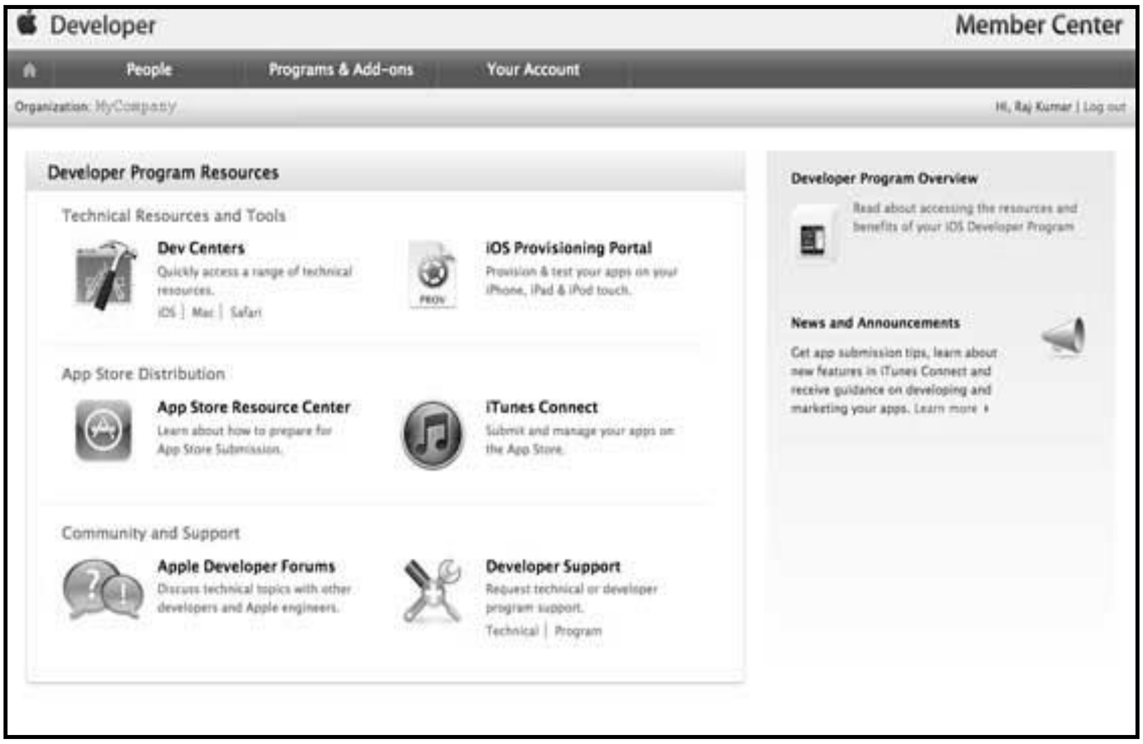
The steps to join the iOS developer program are as follows:

1. To register, click on the link - (https://developer.apple.com/programs/ios/).



2. Click on Enroll Now in the page that is displayed.
3. You can either sign in to your existing apple account (if you have one) or create a new Apple ID.
4. Thereafter, you have to select between Individual and Company accounts. Use company account if there will be more than one developer in your team. In individual account, you can't add members.
5. After entering the personal information (for those who newly registers), you can purchase and activate the program by paying with the help of your credit card (only accepted mode of payment).

6. Now you will get access to developer resources by selecting the member center option in the page.



Developer
Member Center

People | Programs & Add-ons | Your Account

Organization: MyCompany
Hi, Raj Kumar | Log out

Developer Program Resources

Technical Resources and Tools

Dev Centers
Quickly access a range of technical resources.
iOS | Mac | Safari

iOS Provisioning Portal
Provision & test your apps on your iPhone, iPad & iPod touch.

App Store Distribution

App Store Resource Center
Learn about how to prepare for App Store Submission.

iTunes Connect
Submit and manage your apps on the App Store.

Community and Support

Apple Developer Forums
Discuss technical topics with other developers and Apple engineers.

Developer Support
Request technical or developer program support.
Technical | Program

Developer Program Overview
Read about accessing the resources and benefits of your iOS Developer Program

News and Announcements
Get app submission tips, learn about new features in iTunes Connect and receive guidance on developing and marketing your apps. Learn more ▸

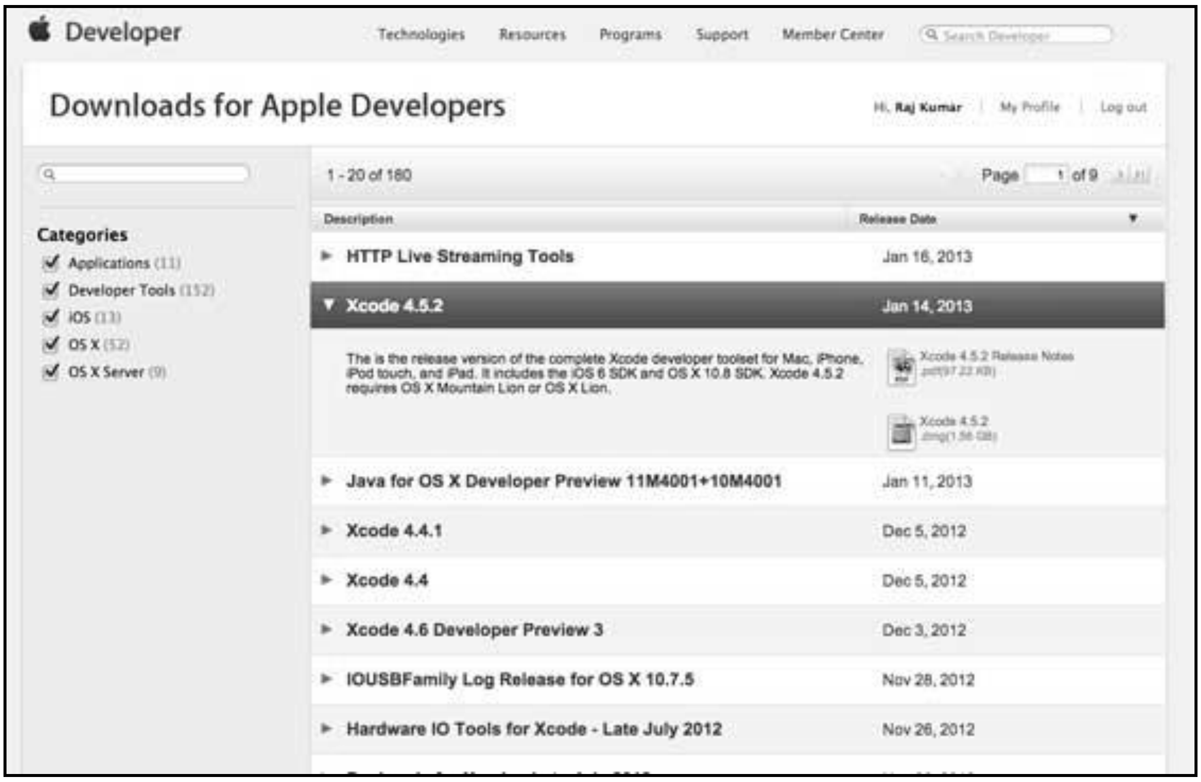
7. Here you will be able to do the following:

   - Create provisioning profiles.
   - Manage your team and devices.
   - Managing application to app store through iTunes Connect.
   - Get forum and technical support.

2. iOS – Environment Setup

iOS – Xcode Installation

1. Download the latest version of Xcode from (https://developer.apple.com/downloads/)



Developer — Technologies — Resources — Programs — Support — Member Center — Search Developer

Downloads for Apple Developers

Hi, Raj Kumar | My Profile | Log out

1 - 20 of 160 — Page 1 of 9

Categories
- ✓ Applications (11)
- ✓ Developer Tools (152)
- ✓ iOS (13)
- ✓ OS X (52)
- ✓ OS X Server (9)

Description	Release Date
► HTTP Live Streaming Tools	Jan 16, 2013
▼ Xcode 4.5.2	Jan 14, 2013
The is the release version of the complete Xcode developer toolset for Mac, iPhone, iPod touch, and iPad. It includes the iOS 6 SDK and OS X 10.8 SDK. Xcode 4.5.2 requires OS X Mountain Lion or OS X Lion.	Xcode 4.5.2 Release Notes pdf(97.22 KB); Xcode 4.5.2 dmg(1.56 GB)
► Java for OS X Developer Preview 11M4001+10M4001	Jan 11, 2013
► Xcode 4.4.1	Dec 5, 2012
► Xcode 4.4	Dec 5, 2012
► Xcode 4.6 Developer Preview 3	Dec 3, 2012
► IOUSBFamily Log Release for OS X 10.7.5	Nov 28, 2012
► Hardware IO Tools for Xcode - Late July 2012	Nov 26, 2012

2. Double click the Xcode dmg file.

3. You will find a device mounted and opened.

4. There will be two items in the window that's displayed namely, Xcode application and the Application folder's shortcut.

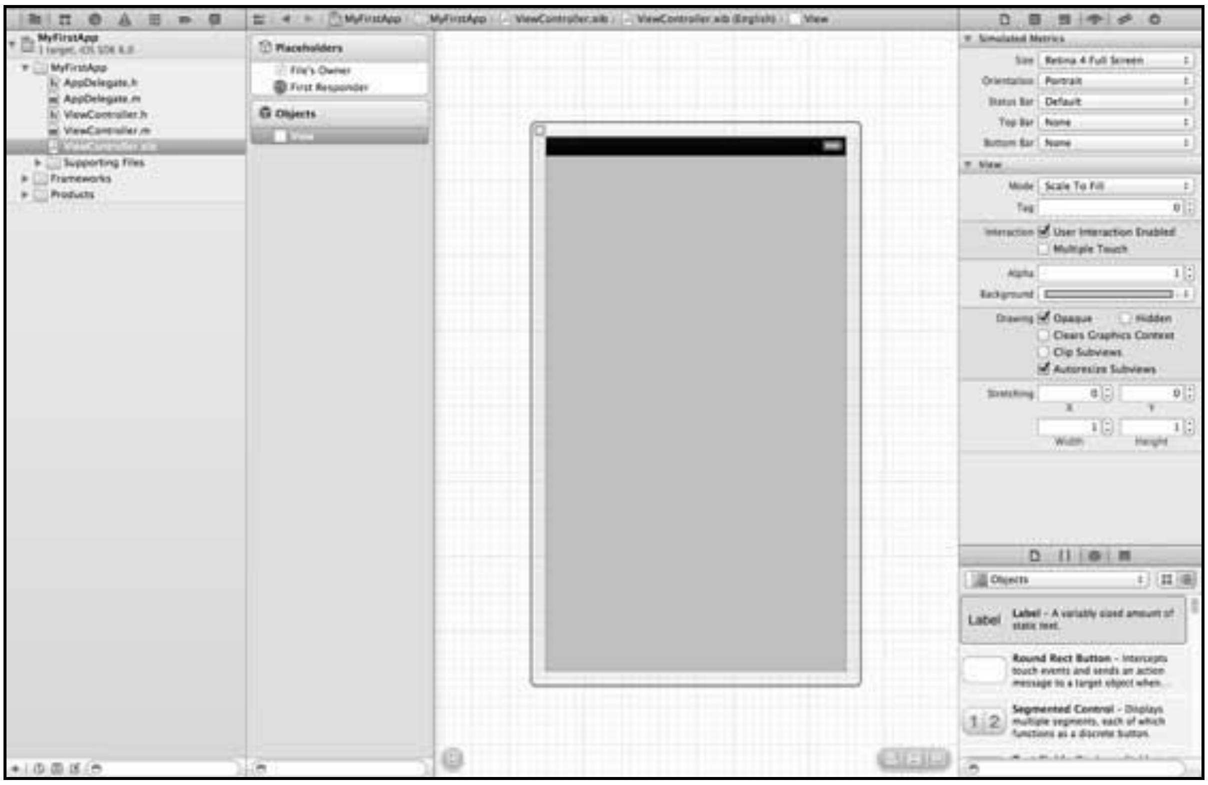
5. Drag the Xcode to application and it will be copied to your applications.

6. Now Xcode will be available as a part of other applications from which you can select and run.

You also have another option of downloading Xcode from the Mac App store and then install following the step-by-step procedure given on the screen.

Interface Builder

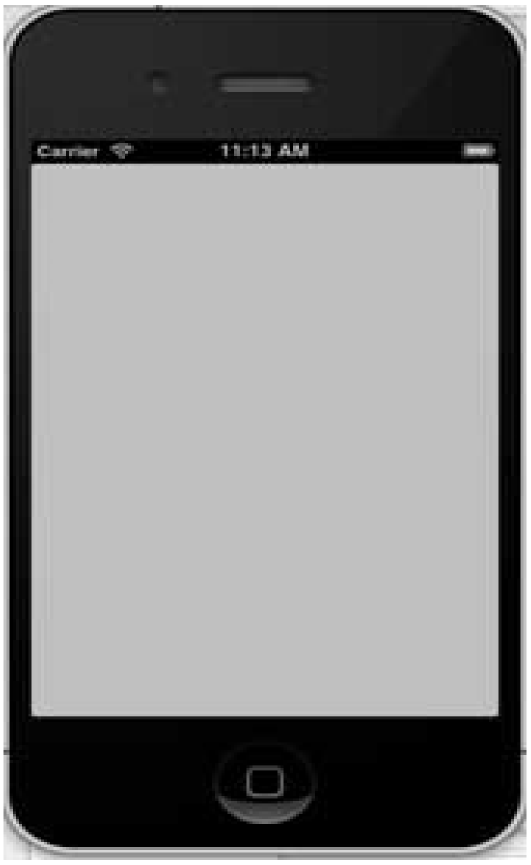
Interface builder is the tool that enables easy creation of UI interface. You have a rich set of UI elements that is developed for use. You just have to drag and drop into your UI view. We'll learn about adding UI elements, creating outlets and actions for the UI elements in the upcoming pages.



You have objects library at the right bottom that consists the entire necessary UI element. The user interface is often referred as **xibs**, which is its file extension. Each of the xibs is linked to a corresponding view controller.

iOS Simulator

An iOS simulator actually consists of two types of devices, namely iPhone and iPad with their different versions. iPhone versions include iPhone (normal), iPhone Retina, iPhone 5. iPad has iPad and iPad Retina. A screenshot of an iPhone simulator is displayed below.



You can simulate location in an iOS simulator for playing around with latitude and longitude effects of the app. You can also simulate memory warning and in-call status in the simulator. You can use the simulator for most purposes, however you cannot test device features like accelerometer. So, you might always need an iOS device to test all the scenarios of an application thoroughly.

3. iOS – Objective C

The language used in iOS development is objective C. It is an object-oriented language and hence, it would be easy for those who have some background in object-oriented programming languages.

Interface and Implementation

In Objective C, the file where the declaration of class is done is called the **interface file** and the file where the class is defined is called the **implementation file**.

A simple interface file **MyClass.h** would look like the following:

```
@interface MyClass:NSObject{
// class variable declared here
}
// class properties declared here
// class methods and instance methods declared here
@end
```

The implementation file **MyClass.m** would be as follows:

```
@implementation MyClass
// class methods defined here
@end
```

Object Creation

Object creation is done as follows:

```
MyClass  *objectName = [[MyClass alloc]init] ;
```

Methods

Method is declared in Objective C as follows:

```
-(returnType)methodName:(typeName) variable1 :(typeName)variable2;
```

An example is shown below.

```
-(void)calculateAreaForRectangleWithLength:(CGfloat)length
andBreadth:(CGfloat)breadth;
```

You might be wondering what the **andBreadth** string is for; actually it’s an optional string, which helps us read and understand the method easily, especially at the time of calling. To call this method in the same class, we use the following statement:

```
[self calculateAreaForRectangleWithLength:30 andBreadth:20];
```

As said above, the use of andBreadth helps us understand that breadth is 20. Self is used to specify that it's a class method.

Class Methods

Class methods can be accessed directly without creating objects for the class. They don't have any variables and objects associated with it. An example is shown below.

```
+(void)simpleClassMethod;
```

It can be accessed by using the class name (let's assume the class name as MyClass) as follows:

```
[MyClass simpleClassMethod];
```

Instance Methods

Instance methods can be accessed only after creating an object for the class. Memory is allocated to the instance variables. An example instance method is shown below.

```
-(void)simpleInstanceMethod;
```

It can be accessed after creating an object for the class as follows:

```
MyClass  *objectName = [[MyClass alloc]init] ;
[objectName simpleInstanceMethod];
```

Important Data Types in Objective C

S.N.	Data Type
1	**NSString** It is used for representing a string.
2	**CGfloat** It is used for representing a floating point value (normal float is also allowed but it's better to use CGfloat).
3	**NSInteger**

	It is used for representing integer.
4	**BOOL** It is used for representing Boolean (YES or NO are BOOL types allowed).

Printing Logs

NSLog - used for printing a statement. It will be printed in the device logs and debug console in release and debug modes respectively. For example,

```
NSlog(@"");
```

Control Structures

Most of the control structures are same as in C and C++, except for a few additions like for in statement.

Properties

For an external class to access the class, variable properties are used. For example,

```
@property(nonatomic , strong) NSString *myString;
```

Accessing Properties

You can use dot operator to access properties. To access the above property, we will do the following.

```
self.myString = @"Test";
```

You can also use the set method as follows:

```
[self setMyString:@"Test"];
```

Categories

Categories are used to add methods to the existing classes. By this way, we can add method to classes for which we don't have even implementation files where the actual class is defined. A sample category for our class is as follows:

```
@interface MyClass(customAdditions)
- (void)sampleCategoryMethod;
@end
```

```
@implementation MyClass(categoryAdditions)


-(void)sampleCategoryMethod{
   NSLog(@"Just a test category");
}
```

Arrays

NSMutableArray and NSArray are the array classes used in objective C. As the name suggests, the former is mutable and the latter is immutable. An example is shown below.

```
NSMutableArray *aMutableArray = [[NSMutableArray alloc]init];
[anArray addObject:@"firstobject"];
NSArray *aImmutableArray = [[NSArray alloc]
initWithObjects:@"firstObject",nil];
```

Dictionary

NSMutableDictionary and NSDictionary are the dictionary classes used in objective C. As the name suggests, the former is mutable and the latter is immutable. An example is shown below.

```
NSMutableDictionary*aMutableDictionary = [[NSMutableArray alloc]init];
[aMutableDictionary setObject:@"firstobject" forKey:@"aKey"];
NSDictionary*aImmutableDictionary= [[NSDictionary
alloc]initWithObjects:[NSArray arrayWithObjects:
@"firstObject",nil] forKeys:[ NSArray arrayWithObjects:@"aKey"]];
```