iOS Programming Basic: How Does tl Hello World App Work?

april 14, 2012 by simon ng 84 comments

Tweet g+1 10

I hope you enjoy the first iOS programming tutorial and already created your first app. Before we move onto the next tutorial and build a more complex app, let's step back and have a closer look at the Hello World app. It'll be good for you to understand some of the Objective-C syntax and the inner workings of the app.

So far you follow the step-by-step guide to build the Hello World app. But as you go through the tutorial, you may come across these questions:

What are those .xib, .h and .m file?

What are those "ugly" code inside "showMessage"? What do they mean?!

What actually happens after you taps the "Hello World" button? How does the button t "showMessage" action?

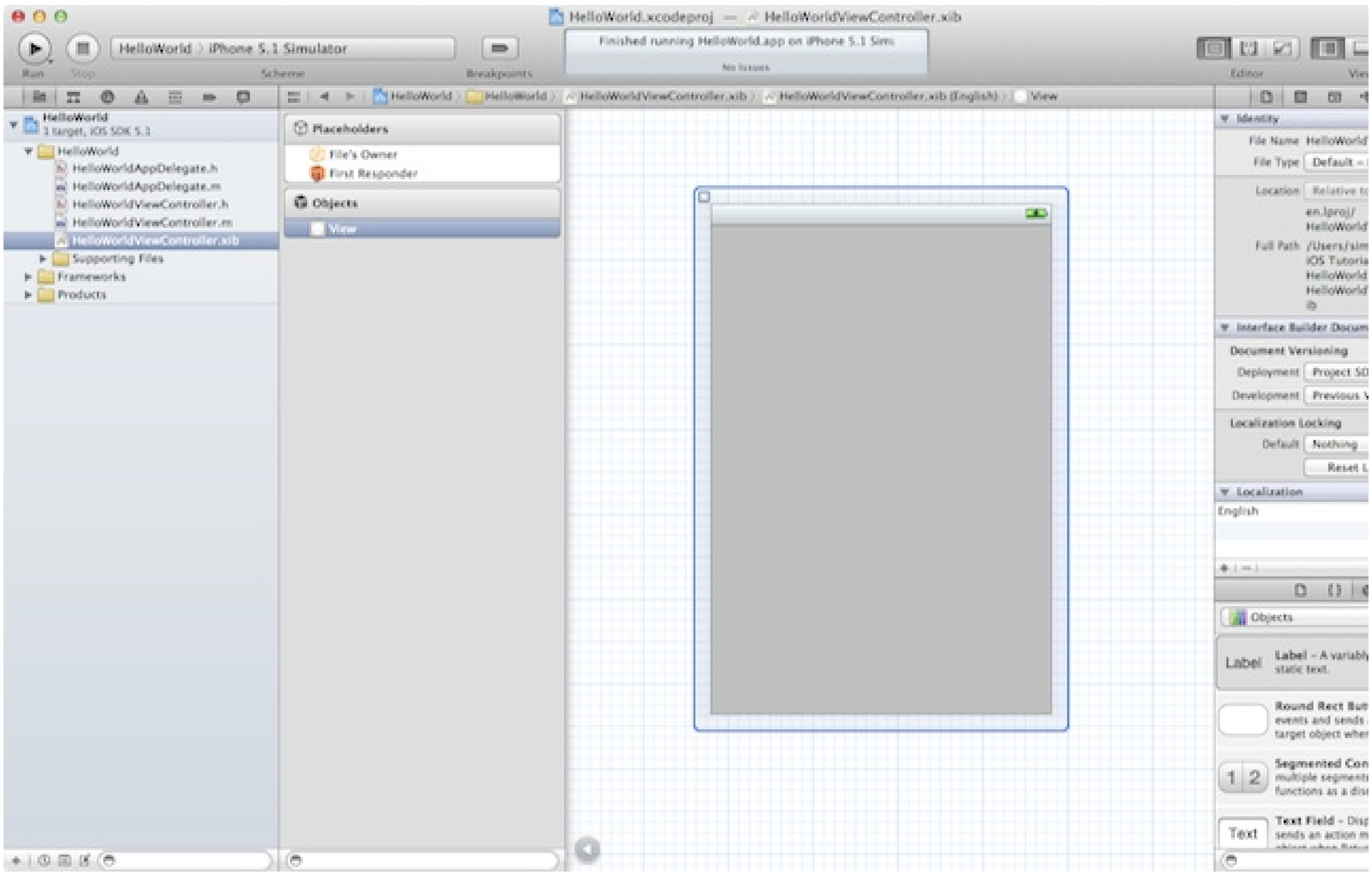
How does the "Run" button in Xcode work?

I want you to focus on exploring the Xcode environment so I didn't explain any of the above previous post. Yet it's essential for every developer to understand the inner details behind t and grasp the basic concept of iOS programming. For some technical concepts, they may b to understand particularly you have no programming background. Don't worry, however. T start. As you move on and write more code in later tutorials, you'll get better understanding programming. Just try your best to learn as much as possible.

Interface Builder, Header and Implementation Files

First, what are those .xib, .h and .m files? This is a very good question raised by one of the re Under the Project Navigator, you should find three main types of files – .xib, .h and .m. (If yo the "Supporting Files" folder, you'll find other file types such as plist and framework. But for forget about them first. We'll talk about them later.)

.xib – For files with .xib extension, they're Interface Builder files that store the application's interface (UI). As you click on the .xib file, Xcode automatically switches to the Interface Bui to edit the UI of the app via drag-and-drop.


HelloWorld.xcodeproj — HelloWorldViewController.xib
HelloWorld › iPhone 5.1 Simulator
Finished running HelloWorld.app on iPhone 5.1 Sim
No Issues
HelloWorld
HelloWorldAppDelegate.h
HelloWorldAppDelegate.m
HelloWorldViewController.h
HelloWorldViewController.m
HelloWorldViewController.xib
Supporting Files
Frameworks
Products
Placeholders
File's Owner
First Responder
Objects
View
Identity
Localization
Objects
Label

Interface Builder in Xcode

.h and .m – Files with .h extension refers to the header files while those with .m extension
implementation files. Like most of the programming languages, the source code of Objectiv
divided into two parts: *interface* and *implementation*.

Well, to put in analogy that you can better understand both terms, let's consider a TV remot
convenient to control the volume of a TV set wirelessly with a remote. To increase the spea
you press the "Volume +" button. To switch channel, you simply key in the channel number
you. Do you know what happens behind the scene when pressing the "Volume" button? Pr
believe most of us don't know how the remote communicates with the TV set and controls t
volume. What we just know is, that button is used for changing the volume. In this example,
that interacts with you is the "interface" and the inner detail which is hidden behind the bu
referred as the "implementation".

Now you should have a better idea about interface and implementation. Let's go back to the
Objective-C, interfaces of a class are organized in ".h" file. We use the syntax "@interface" t
the interface of a class. Take a look at the HelloWorldViewController.h, which is the header

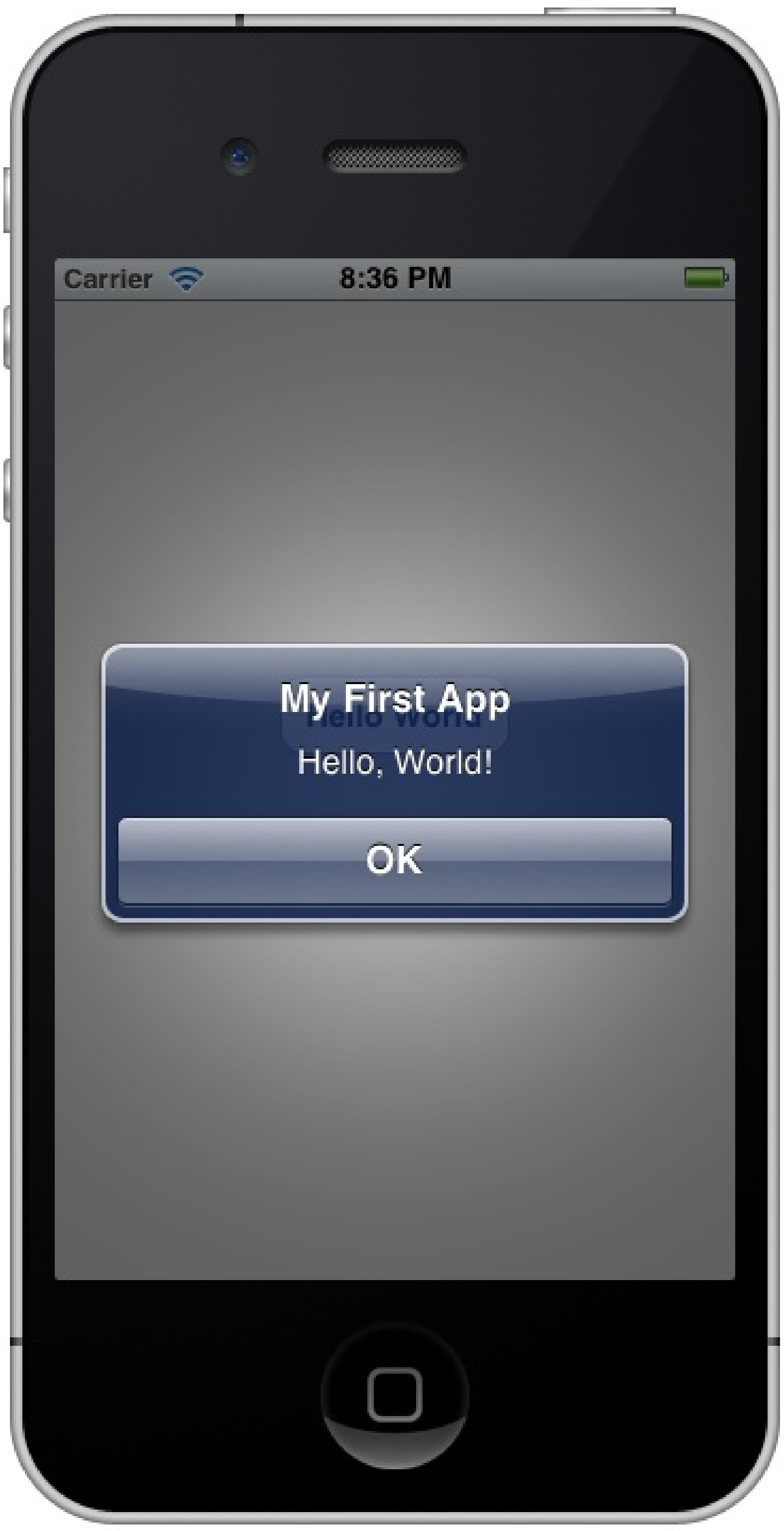
```
@interface HelloWorldViewController : UIViewController

-(IBAction)showMessage;

@end
```

It starts with "@interface" followed by HelloWorldViewController, which is the class name. I
declares a "showMessage" action, which is known as a method call.

Like the "Volume" button, apparently we do not know how the "showMessage" action work
know it's used to display a message on screen. The actual implementation is put in the
HelloWorldViewController.m, the implementation file:

```
@implementation HelloWorldViewController

// I've removed other methods for better reading. Focus on the showMessage method fi

- (IBAction)showMessage
{
    UIAlertView *helloWorldAlert = [[UIAlertView alloc]
                                   initWithTitle:@"My First App" message:@"Hello, World!" delegate:nil c

    // Display the Hello World Message
    [helloWorldAlert show];
}

@end
```

As you can see from the above, you use the declaration "@implementation" to declare an implementation. Inside the "showMessage", it's the actual code defined to display the alert screen. You may not understand every single line of code inside the "showMessage" metho it creates an UIAlertView with "My First App" as the title and "Hello, World" as the message. calls up the "show" method and request iOS to display the pop-up message on screen.



My First App

Hello, World!

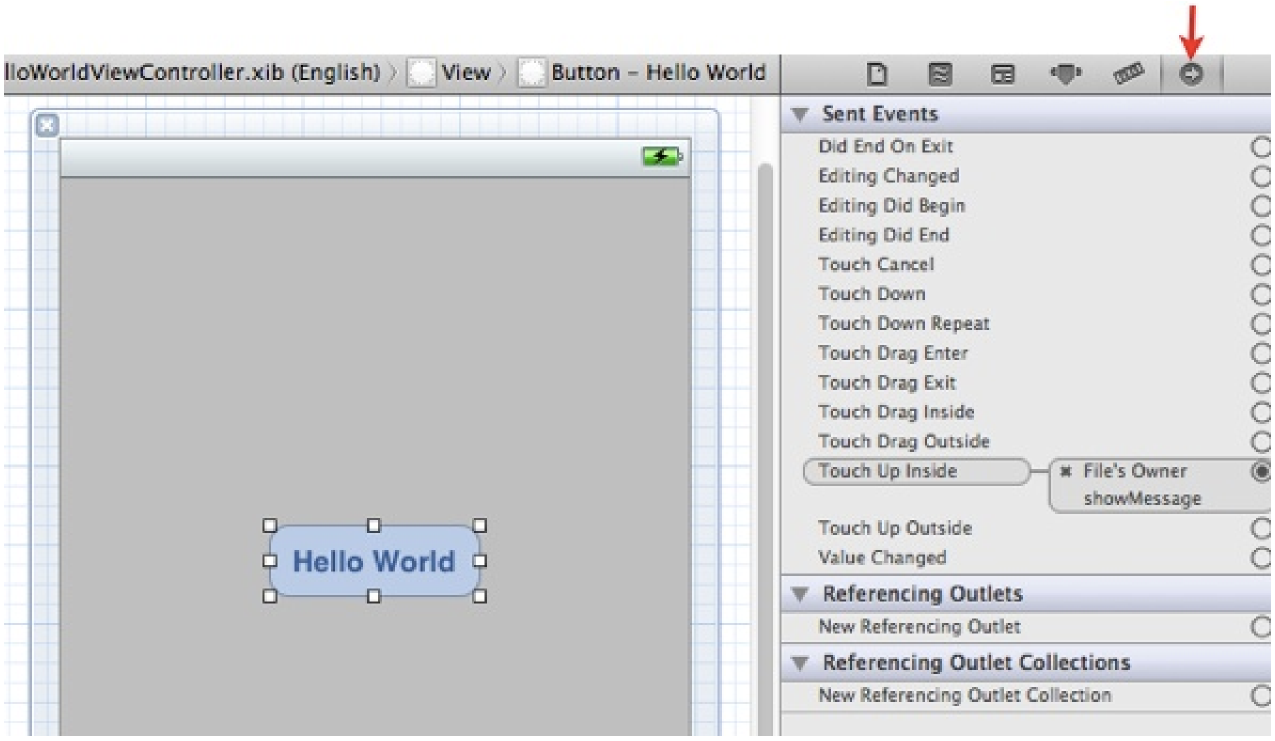
Hello World App

It's vital for you to understand the concept of interface and implementation. If you have any
feel free to raise your question at our forum.

Behind the Touch and Tap

What actually happened after tapping the "Hello World" button? How does the "Hello Worl
invoke the "showMessage" method to display the "Hello World" message?

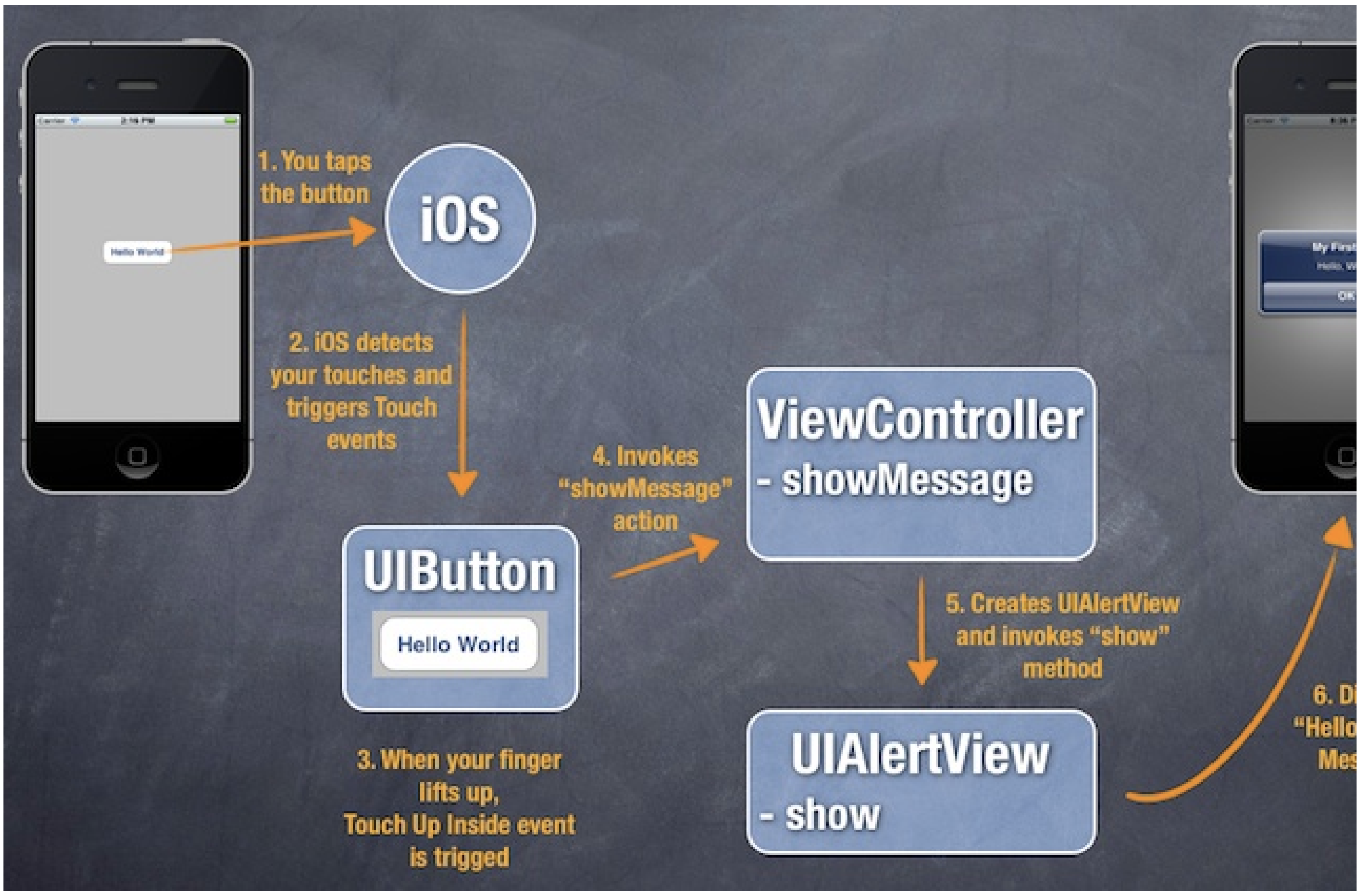
Recalled that you established a connection between the "Hello World" button and the "send
action in Interface Builder. Try opening up the "HelloWorldViewController.xib" again and se
"Hello World" button. Click the "Sent Events" button in the Utility area to open the Sent Eve



The Sent Events section shows you all connections between events and actions. As you can
above figure, the "Touch Up Inside" event is connected to the "showMessage" action. In iOS
event-driven. The control/object (e.g. UIButton) listens for certain events (e.g. touches and
When the event is triggered, the object calls up the preset action that associates with the e

In our Hello World app, when users lift up the finger inside the button, the "Touch Up Insid
triggered. Consequently, it calls up the "showMessage" action to display the "Hello World" r

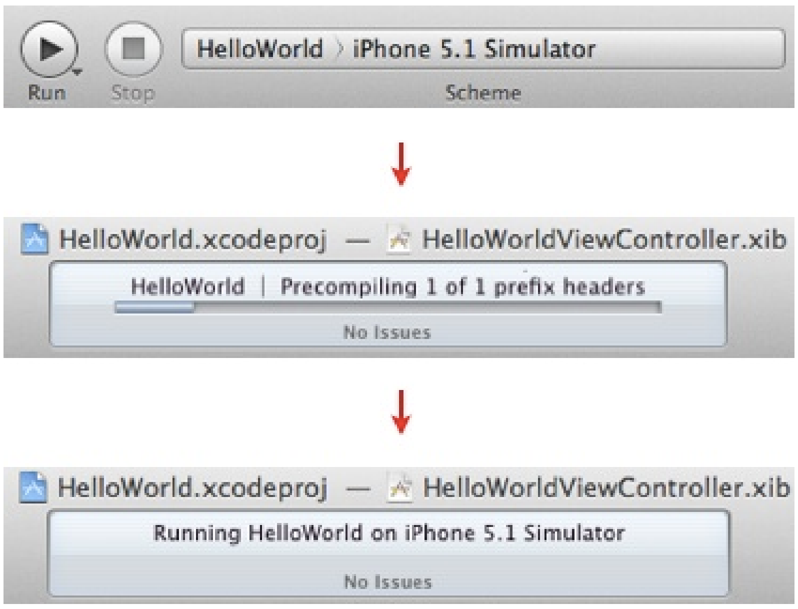
The below illustration sums up the event flow and what I have just described.


1. You taps the button
iOS
2. iOS detects your touches and triggers Touch events
UIButton
Hello World
3. When your finger lifts up, Touch Up Inside event is trigged
4. Invokes "showMessage" action
ViewController - showMessage
5. Creates UIAlertView and invokes "show" method
UIAlertView - show
6. Di "Hello Mes

Event and Message Flow of Hello World App

Behind the Scene of the "Run" Button

When you click the "Run" button, Xcode automatically launches the Simulator and runs you
what happens behind the scene? As a programmer, you have to look into the entire process


Run
Stop
HelloWorld › iPhone 5.1 Simulator
Scheme
HelloWorld.xcodeproj — HelloWorldViewController.xib
HelloWorld | Precompiling 1 of 1 prefix headers
No Issues
HelloWorld.xcodeproj — HelloWorldViewController.xib
Running HelloWorld on iPhone 5.1 Simulator
No Issues

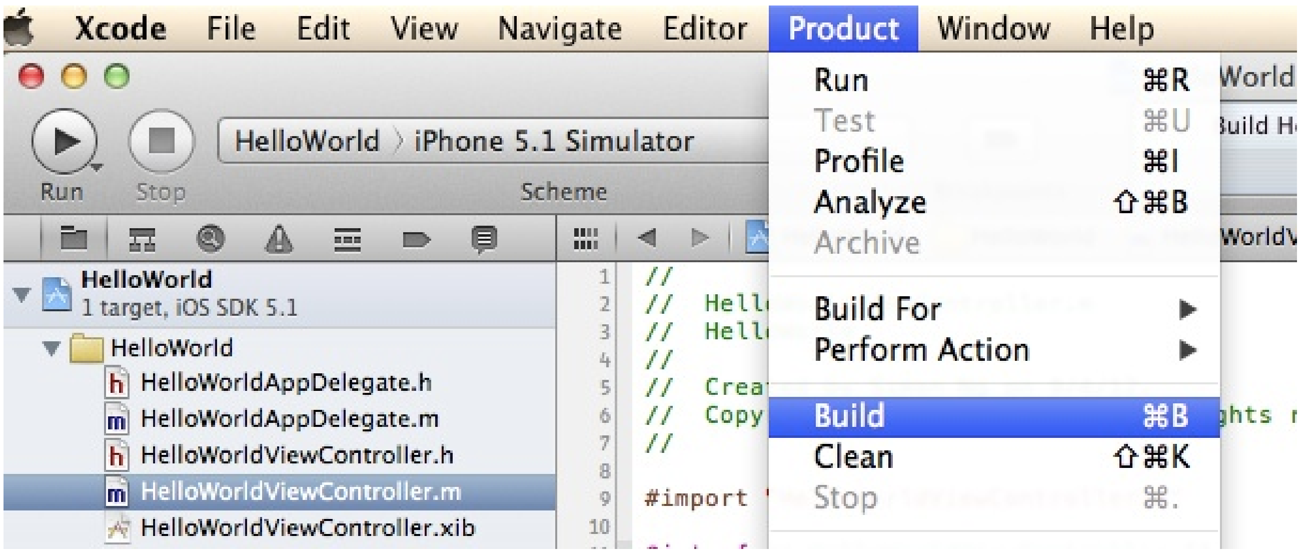
The entire process can be broken into three phases: compile, package and run.

Compile – You probably think iOS understands Objective-C code. In reality, iOS only reads

code. The Objective-C code is only for you, the programmer to write and read. To make iO
understand the source code of the app, we have to go through a translation process to tran
Objective-C code into machine code. This process is referred as “compile”. Xcode already c
a built-in compiler to compile the source code.

Package – Other than source code, an app usually contains resource files such as images, t
xib files, etc. All these resources are packaged to make up the final app.

We used to refer these two processes as the “build” process.



Run – This actually launches the Simulator and loads your app.

Got Questions?

I try my best to explain how Hello World app actually works. As a beginner without prior pro
experience, it’s not easy to understand all the concepts we just discussed. So don’t hesitate
questions at our forum. I, as well as, other experienced members are eager to help.

As always, leave me comment to share your thought about the tutorial. Comments are alway
😃

You May Like These:

iOS Programming 101: Customize UITableView and UITableViewCell Background using Storyboard	iOS Programming with iCloud: An Introduction	How to Create a Simple RSS Reader App	How To Fetch and Parse JSON Using iOS SDK

Tweet

filed under: course, tutorials tagged with: ios programming, ios sdk, objective c, xcode

Get Free Chapters of Our AppCoda Book

The Learn iOS 7 Programming from Scratch is a 400-page eBook written for beginners little or even no programming experience. It is based on the tutorials of our popular programming course. The book starts with the basics and walks you through the proces create iOS apps using iOS 7 SDK and Xcode 5. Want to learn more? Check it out here ar two free chapters.

Build Your Own Game and Monetize on the Side

ALSO ON APPCODA

iOS Programming Tutorial: Creating a Universal App
2 comments

How To Implement Search Bar in iOS 7 Us
Storyboard 16 comments

Understanding Multipeer Connectivity Framework in iOS 7 – Part 1 13 comments

Using Text Kit to Manage Text in Your iOS
7 comments

78 Comments Appcoda

Sort by Best Share

Join the discussion…

Nidhi • a year ago
Just loved the details you have given with the screenshots and I can very confidently say this
tutorial to start with.

Good work dude. Cheers!
27 • Reply • Share ›

Shailesh • a year ago
Superb tutorial :) You have explained very nice way. Thank you very much :)
9 • Reply • Share ›

phuctrantuan • 11 months ago
it's very easy to understand. Thank you so much.
6 • Reply • Share ›

JW • a year ago
Wow! This is a great beginner tutorial. Simple and get to the point with screenshot. My I.Q. v
point! I am going to read the rest of the tutorials. Thank you for taking the time creating this
5 • Reply • Share ›

Ram • a year ago
amazing tutorial
4 • Reply • Share ›

bob • a year ago

Very helpful tutorial Thanks
Make more

3 ⌃ | ⌄ • Reply • Share ›

Cool Tim ➦ bob • 2 months ago

Yes! Make more!

⌃ | ⌄ • Reply • Share ›

Mulie • 5 months ago

Thank you very much for your kind contributions and for doing your best in all lessons!

1 ⌃ | ⌄ • Reply • Share ›

joseph • 5 months ago

super thanks

1 ⌃ | ⌄ • Reply • Share ›

Ste • 6 months ago

Thanks so much for this great tutorial. I'm a complete newbie to this and I'm really excited at
accessible iOS app development can be. It wouldn't be without your help so I would like to th
putting your time and effort into helping people like me realise their dreams of writing their
don't even care if my ideas are not original, I am just happy to have learnt these skills and to
on my phone which I can say I have coded (with help of course :)). Thanks

1 ⌃ | ⌄ • Reply • Share ›

Gabriel • 6 months ago

Very good tutorial and explanation!!!! Thanks a lot!!!

1 ⌃ | ⌄ • Reply • Share ›

khanhtungna • a year ago

Very helpful guide. Thank so much!

1 ⌃ | ⌄ • Reply • Share ›

R. George • a year ago

This tutorial is really helpful, just what I need to get into iOS programming. Thanks so much

1 ⌃ | ⌄ • Reply • Share ›

obloodyhell • 7 months ago

OK, anyone want to tell me how to END the app? I mean, it seems remarkably bad behavior t
app that doesn't just shut itself off when you are done with it -- no, I don't want to hit the XC
and chop it off at the legs -- I want to tell the APP it is done INSIDE the app, and that it can

No one seems to think this is something a beginner ought to be doing... lovely.

⌃ | ⌄ • Reply • Share ›

Chris ➦ obloodyhell • 6 months ago

This is not an oversight. It is also not something that ANY developer, beginner or othe
to be doing.

iOS apps, by design, don't have an exit() or die() call that the developer explicitly mak

iOS apps continue to run in the background until iOS determines that the OS is runni resources. Then the app is ended by the OS. One can place some calls in the ApplicationWillTerminate (as of iOS 4; I assume it's still called this) hook to take care minute QUICK tasks (e.g. saving prefs/data to a .plist, etc.). If you take too long to co tasks, the app will be closed anyway.

If you want to close an app as a user, you can leave the app, double-home-button clic swipe up on the app (iOS 7+) or long-press and click the X that overlays the app (< i

As far as programmatically closing the app, this doesn't fall to the developer within th lifecycle.

1 · Reply · Share ›

Sanjeev Reddy · 4 days ago

Really Awesome .Really from scratch.I enjoyed it.

Reply · Share ›

fizza · 10 days ago

Just awesome explanation!!!

Reply · Share ›

Mathew A · 20 days ago

great tutorial...but in what case do i have to use the .xib or storyboard ? i don't understand th between them

Reply · Share ›

Rubim Shrestha · a month ago

great tutorial.... (Y)

Reply · Share ›

smilealgernon · a month ago

Hello,i have a question.I find the xib is xml format， I was wondering how does xcode deal w file when compiled. sorry about my english...

Reply · Share ›

Tam · 2 months ago

The best tutorial Ive ever seen

Reply · Share ›

Bhavin Kansagara · 2 months ago

Best Explained with the image of entire process happening behind the events...Really appreciated..Thanks

Reply · Share ›

Vaishali Modi · 2 months ago

Nice tutorial for beginners. Thanks.. :)

Reply • Share ›

maikytec • 2 months ago

Excelent...

Reply • Share ›

Nikola KneeJah Markovic • 2 months ago

Best. Class. EVER!

Reply • Share ›

waye • 3 months ago

I can not find any tutorials better than yours!

Reply • Share ›

Bailey • 4 months ago

Best Tutorial I have found, and its free!

Reply • Share ›

Vinicius de Paula • 4 months ago

Excellent teaching, congratulations!!! ;)

Reply • Share ›

shilpa • 4 months ago

u are just awesome... thank u so much..:) :*

Reply • Share ›

Saran • 5 months ago

As a beginner,this tutorial helped me a lot.Thank U

Reply • Share ›

GrayChen • 5 months ago

I like ur tutorial.

Reply • Share ›

Maulik • 5 months ago

what is the extension of file/package it creates once the "build" is finished... i mean is it .exe f

Reply • Share ›

Arnab Bose • 5 months ago

Thanks for your tutorial.It's very easy to understand. Thank you so much.

Reply • Share ›

Nick D • 5 months ago

Great tutorial, thanks a lot!

Reply • Share ›

ROONEY • 6 months ago
Thanks a lot, Simon. You make it so simple, and easy to learn.
Hoping to see more how-to articles in future.
• Reply • Share ›

Kumar Subramani • 6 months ago
Thanks Simon..
• Reply • Share ›

CEMSOFT SOFTWARE • 6 months ago
thank you ; very useful tutorial for me.
• Reply • Share ›

Salophone • 6 months ago
awesome tutorial. Detail and clear explaination.. thanks.
• Reply • Share ›

Louie • 6 months ago
Great Interface and Implementation explanation. I like the level of detail you are teaching, n
and not to few. A great kick starter.
• Reply • Share ›

helemi • 6 months ago
great job
• Reply • Share ›

Pat • 6 months ago
First xcode tut i have been able to follow and get a working app out of it. so kudos!
• Reply • Share ›

MTagb • 7 months ago
I have been working with iOS dev for over a year, but loved this one.
• Reply • Share ›

raj • 7 months ago
amazing , want some more examples ..
• Reply • Share ›

gyesy • 7 months ago
Thanks alot!!!
• Reply • Share ›

Hanne • 7 months ago
Thank you, this is very helpful :)
• Reply • Share ›

sonnt • 8 months ago

sonnt · 8 months ago

This tutorial is so helpful!
In x-code 4.5, did .xib file was replaced by .storyboard file?

· Reply · Share ›

Simon Ng Mod ➦ sonnt · 8 months ago

It doesn't replace .xib. You are free to choose from Interface Builder or Storyboard. If
learn more about Storyboard, you can check out this tutorial:
http://www.appcoda.com/use-sto...

· Reply · Share ›

Yogesh · 8 months ago

Now after referring to dozens of online tutorials I can say, I found what I was looking for :) T
the awesome explanation and sharing knowledge.

· Reply · Share ›

Carlos · 8 months ago

I would love to see this on XCode 4.6

· Reply · Share ›

Sandip Pund · 9 months ago

Amazing Tutorial for Beginners.....Good Work...!

Search this website… Search

Sign Up for Our Free Tutorials

Enter your email address Subscribe

Recent Discussions

Core Data

Display 4 different lists of events on the same tableview controller

How To Handle Row Selection in UITableView

Search Bar in iOS 7 Using Storyboard

Slide-out side bar

RoundImage View

making tables

login and register on my app

Search bar

Reusable View Having Various Components Like image, labels and 5 buttons

Recent Posts

First Time App Developer Success Stories Part 2: From Zero iOS Programming Experience to Launching Their First Apps

iOS Programming 101: How To Create Circular Profile Picture and Rounded Corner Image

Background Transfer Service in iOS 7 SDK: How To Download File in Background

First Time App Developer Success Stories Part 1: From Zero iOS Programming Experience to Launching Their First Apps

Adding Animated Effects to iOS App Using UIKit Dynamics

Connect With Us

About AppCoda

About

Tutorials

Course

Book

Game Kit

Marketplace

Forum

Contact

Latest Comments

凯峰 费 on How To Create UIPageViewController Using Storyboard

Raushan Kumar on Background Transfer Service in iOS 7 SDK: How To Download File in Background

Torres Ho on Adding Local Notifications in Your iOS App

Raj ambi on Understanding Multipeer Connectivity Framework in iOS 7 – Part 1 iOS Programming 101: How To Create Circular Profile … – AppCoda | MUSCLE CODER on iOS Programming 101: How To Create Circular Profile Picture and Rounded Corner Image

Connect With Us

Email Newsletter

Sign up to receive free tutorial and to hear what's going on with AppCoda!

Enter your email address Subscribe