iOS 开发 60 分钟入门

本文面向已有其它语言（如 Java，C，PHP，Javascript）编程经验的 iOS 开发初学者，初衷在于让我的同事一小时内了解如何开始开发 iOS App，学习目标包括：

- 能使用 Xcode IDE、模拟器
- 能修改、调试已有 iOS App
- 能在已有应用内创建新模块
- 能创建新应用
- 能发布应用到 App Store

本文不包含任何高级的 iOS 开发知识，已学会 iOS 开发的同学不要看，看完这篇文章学会了的同学也不用再看了。

不仅是学习一门新语言

有过脚本开发经验的人（如 Javascript，PHP，Shell）在刚开始学习 iOS 开发的时候，会觉得 iOS 开发的学习曲线比脚本语言要高，是的，这种感觉是对的。因为学 iOS 开发，不仅是学习一门新语言，它包括：

- 一门语言：Objective-C
- 一个框架：Cocoa Touch
- 一个 IDE：Xcode

初学脚本语言通常不会来绘制图形界面、与人交互，iOS 如果不做图形界面，像脚本语言一样处理文本操作数据库，就没啥意思了。

所以，过去我写别的新手入门教程，通常都是写《XXX 入门 15 分钟教程》，而 iOS 就要花数倍的时间来写了。

环境准备

做 iOS 开发一定要有苹果的软件环境：Mac OS 操作系统、Objective-C 编译器、设备模拟器等，开发工具倒不一定要用 Xcode，只要是个源代码编辑工具就行（vim 都行，只是没 Xcode 那么多功能）。

Mac OS

拥有 Mac OS 环境最简单的方法是找一台苹果电脑，包括 iMac, MacBook Pro, MacBook Air, Mac Mini，但不包括苹果的移动设备（iPod Touch, iPhone, iPad, iPad Mini，它们运行的是 iOS 系统，不是 Mac OS），苹果电脑在出厂

的时候就会预装 Mac OS，目前最新版本是 Mac OS X 10.8，主流的版本还有 Mac OS X 10.6、Max OS X 10.7。

如果囊中羞涩，可以借一台，或者上淘宝买个二手的。

黑苹果

提到 iOS 开发入门，似乎没办法不说黑苹果。所谓黑苹果，就是把 Mac OS 改造后安装在非苹果的硬件上，这是违反 DMCA 法案的，黑苹果的更多资料，可以在维基上找到

苹果电脑价格高，国内软件开发者生存压力大，所以黑苹果在国内也有一些真实的存在，国外当然也有啦。

黑苹果基本可以胜任 iOS 开发，但有一些问题：

- 安装黑苹果是非法的
- 个人行为苹果公司一般不会追究，但会遭同行的鄙视
- 黑苹果超级难装，挑硬件。即使完全相同的型号，相同的批次，也有可能 A 机器装上了，B 机器装不上
- 黑苹果系统多少都存在一些使用上的问题，像驱动 Bug 啦、待机恢复蓝屏啦、上网浏览有问题啦
- 黑苹果不能随意升级，可能升级一次 safari 就导致整个系统崩溃了

上面这些虽然不会直接影响 Xcode 写代码、模拟器测试，但写着写着想上网查个东西的时候，safari 不能翻页，确实挺影响心情的。所以，钱包允许的前提下，还是搞个苹果电脑省心一些。

Xcode 和 模拟器

Xcode 可以在苹果官网免费下载：Xcode 下载地址

安装 Xcode 时会自动安装 iOS SDK 和模拟器。

这么强大的 IDE 居然是免费的，还是挺让人开心的。

从改一个现成的应用开始吧

学一门新软件开发技能，能够第一时间做出一个可运行的产品非常重要，有助于给自己正面激励，我上大学的时候，有很多次想学一门新语言，往往花了半个月，还沉浸在数据类型和语法字典里，连第一个 Hello World 都没做出来。

这一次，就让我们从改一个现成的应用开始吧。

下载

首先，我们从苹果开发者中心下载一个示例代码回来。我选了ToolBarSearch。

在本文档的末尾，还有一些其它的网址可以下载开源 iOS 产品或者代码段，但我试了一下，还是 Apple Sample Code 最容易成功。

下载回来的 zip 文件最好保存在"下载"或者"文稿"目录里，因为在 Mac OS 10.8 以前，有些目录（例如/var/private/tmp）在 Finder 中是看不到的，要通过 Finder 的“前往 > 前往文件夹”功能才能进入。

打开

有三种方式可以打开一个 iOS Project

双击 project 文件

打开 Finder，进入刚刚下载解压的 ToolBarSearch 目录，找到 ToolBarSearch.Xcodeproj 文件，双击之，Xcode 会自动启动，并打开这个项目

在 Xcode 里选择 Project 打开

- 在 Xcode 没启动的情况下（如果 Xcode 已经启动了，就先按 Command Q 退出），启动 Xcode，会弹出“Welcome to Xcode”的欢迎页，点击左下角的“Open Other”按钮，找到 ToolBarSearch 目录，双击 ToolBarSearch 目录，或者双击 ToolBarSearch.Xcodeproj 文件都可以
- 如果 Xcode 处于打开状态，可以点击其菜单栏的 File -> Open，或者 File -> Open Recent，然后再选择要打开的项目

通过命令行打开

在 Mac OS 10.8 以前，有些目录（例如/var/private/tmp），在 Finder 和 Xcode 的 File > Open 对话框中，点击鼠标是找不到的，这时候就要通过命令行终端来打开了。

打开终端，执行：

```
cd /ToolBarSearch 的父目录/ToolBarSearch
open -a Xcode
```

open -a 是 mac os 的系统命令，除了 iOS 项目，别的项目也可以这样打开。

运行刚下载的应用

点击 Xcode 左上角的 Run 按钮（或者同时按下 Comman 和 R 键），Xcode 会编译源码并在模拟器中运行这个应用。

编译成功会在屏幕上淡淡地显示“Build Succeeded”。反之，失败就显示“Build Failed”且不启动模拟器。

修改

在模拟器上看到“Performed search using…”了吧，下面我们改掉它。

- 在 Xcode 左上角的 Run 按钮下方，有一排小按钮，从左到右第三个是一个放大镜图标，鼠标移上去会显示“Show the Search Navigator”，点一下它，打开搜索界面，在它下方出现的 Find 输入框中输入“performed”
- 搜索结果只有一条：ToolbarSearchViewController.m，点文件名下方被高亮的“Performed”字串，右侧代码编辑区会自动打开这个文件，并滚动屏幕，使包含“Performed”的这一行出现在编辑区的中间。
- 修改双引号里的字串，随便改成啥，然后按“Command S”保存。

当然，这些操作，你也可以在终端下通过 grep 和 vim 完成。

运行修改后的应用

按 Command R 运行，看看，是不是看到效果啦？

是的，修改一个应用就这么简单。

Objective-C

Objective-C 是苹果应用软件（包括苹果电脑上的 Mac OS App 和移动设备上的 iOS App）的开发语言。它是一种面向对象的编程语言。

苹果公司还提供了一个软件，叫 Interface Builder，简称 IB，用于可视化的界面制作，就像用 Dreamweaver 做网页，或者像 Visual Basic 做桌面软件一样。后来 IB 就整合进了 Xcode，成了 Xcode 的一部分。这篇文档不讲 IB，只讲 Objective-C，因为：

- 基本上，每一本讲 iOS 开发的书（纸质书、电子书），都有大量的截图一步一步教如何用 IB 开发 iOS 应用，而讲 Objective-C 开发应用的书却没有那么多。
- IB 可以用来直观方便地画界面、设置控件属性、建立代码与控件的联系，但后台的业务逻辑和数据处理仍然要靠 Objective-C，可见，不管用不用 IB，Objective-C 都是绕不过去的。

C 的超集

Objective-C 扩展了 ANSI C，是 C 的超集，也就是说：

- 任何 C 源程序，不经修改，即可通过 Objective-C 编译器成功编译
- Objective-C 源程序中可以直接使用任何 C 语言代码

除了面向对象有语法是 SmallTalk 风格的（下面会讲到），其它非面向对象的语法、数据类型，与 C 完全相同，所以本文就不再赘述。 来看一个经典的 Hello World 示例吧：

```
#import <Foundation/Foundation.h>
int main(int argc, char *argv[]){
    @autoreleasepool{
        NSLog(@"Hello World!");
    }
    return 0;
}
```

是不是仿佛穿越回了大一学习 C 语言的时代，看起来和 C 几乎没有区别，是吧？是的，因为还没用到它的面向对象特性，哈哈！

SmallTalk 的消息传递语法风格

Objective-C 的面向对象语法源自 SmallTalk，消息传递（Message Passing）风格。在源码风格方面，这是它与 C Family 语言（包括 C/C++、Java、PHP）差别最大的地方。

在 Java、C++世界，我们调用一个对象的某方法，在 Objective-C 里，这称作给类型发送一个消息，这可不仅仅是文字游戏，他们的技术细节也是不同的。

在 Java 里，对象和方法关系非常严格，一个方法必须属于一个类/对象，否则编译是要报错的。而在 Objective-C 里，类型和消息的关系比较松散，消息处理到运行时（runtime）才会动态决定，给类型发送一个它无法处理的消息，也只会抛出一个异常，而不会挂掉。

```
[obj undefinedMethod];
```

在代码里调用没定义的方法（这是 Java 世界的习惯说法啊，专业的叫法是，给 obj 对象传递它无法处理的消息），Xcode 会警告，但编译能成功，运行的时候会出错。它会输出这样一个错误：

```
Terminating app due to uncaught exception 'NSInvalidArgumentException', reason: '-
[NSObject undefinedMethod]: unrecognized selector sent to instance 0x8871710'
```

类似 Java 的 OOP 概念

Objective-C 中一些面向对象的概念，也可以在 Java 中找到类似的实现（只能说是类似，不是完全相同），我的读者基本都是 Java 和 PHP 程序员，我会在下文中尽量用 Java 的概念来类比。

GoogleCode 上有人整理了 Java 和 Objective-C 的概念、数据类型对应表，参见这里

字符串

Objective-C 里有字符串是由双引号包裹，并在引号前加一个@符号，例如：

```
title = @"Hello";
if(title == @"hello") {}
```

PHP 程序员要注意，在这里不能用单引号，即使只有一个字符也不能用。Objective-C 与 Java、C 一样，双引号表示字符串。

函数调用

前文述及，不涉及面向对象时，它和 C 是完全一样的。以下是几个函数调用的示例：

不带参数

```
startedBlock();
```

带参数

```
NSLog(@"decrypted string: %@", str);
CGRectMake(0,0,0,0);
```

传递消息给类/实例方法

不带参数

```
[obj method];
```

对应的 Java 版本

```
obj.method();
```

带一个参数：

```
[counter increase:1];
```

对应的 Java 版本

```
counter.increase(1);
```

带多个参数

对 C Family 程序员来说，这是最难接受的，最反人类的：

```
- (void) setColorToRed: (float)red Green: (float)green Blue:(float)blue {...} //定义方法
[myObj setColorToRed: 1.0 Green: 0.8 Blue: 0.2]; //调用方法
```

对应的 Java 版

```
public void setColorToRedGreenBlue(float red, float green, float blue) {...}
myObj.setColorToRedGreenBlue(1.0, 0.8, 0.2);
```

消息嵌套

```
UINavigationBar *bar = [[[UINavigationBar alloc] init] autorelease];
```

对应的 Java 版

```
UINavigationBar bar = UINavigationBar.alloc().init().autorelease();//Java 没有指针，所以星号去掉了
```

类

接口和实现

Objective-C 的类分为接口定义和实现两个部分。接口定义（Interface）放在头文件中，文件扩展名是.h，实现（implementation）放在实现文件中，文件扩展名是.m（也有.mm 的扩展名，表示 Objective-C 和 C++混编的代码）。

接口定义也可以写在.m 文件中，但最好不要这么干

需要注意的是，与 Objective-C 的 interface 概念最接近的是 C 和 C++里的头文件，它与 implementation 是成双成对出现的，作用是声明类的成员变量和方法。它与 Java 的 interface 概念完全不同：

- Objective-C 里，interface 有且只有一个实现，Java 的 interface 可以有 0-N 个实现
- Objective-C 里，interface 可以定义成员属性，Java 里不可以

在 Objective-C 里，和 Java 的 Interface 概念相似的是 Protocol，下文会讲到。

请看示例：

Interface

```
@interface MyClass {
    int memberVar1;
    id  memberVar2;
}
```

```
-(return_type) instance_method1;
-(return_type) instance_method2: (int) p1;
-(return_type) instance_method3: (int) p1 andPar: (int) p2;
@end
```

Implementation

```
@implementation MyClass {
    int memberVar3;
}

-(return_type) instance_method1 {
    ....
}
-(return_type) instance_method2: (int) p1 {
    ....
}
-(return_type) instance_method3: (int) p1 andPar: (int) p2 {
    ....
}
@end
```

接口和实现以@interface、@implementation 开头，都以@end 结束。“@”符号在 Objective-C 中是个很神奇的符号。

冒号也是方法名的一部分，method 和 method:是两个不同的方法名，不是 overload，第二个带参数。

上述代码对应的 Java 版：

```
public class MyClass {
    protected int memberVar1;
    protected pointer memberVar2;
    private int memberVar3;

    public (return_type) instance_method1() {
        ....
    }

    public (return_type) instance_method2(int p1) {
        ....
    }

    public (return_type) instance_method3andPar(int p1, int p2) {
```

```
        ....
    }
}
```

私有方法和公开方法

写在.h 头文件里的方法都是公开的，Objective-C 里没有私有方法的概念（没有你说个蛋啊，哈哈哈哈）。

官方并没有提到 Objective-C 怎么实现私有方法，我查阅了 stackoverflow，统一的答案是，要实现私有方法的效果只能借助 Category，不过，根据我的测试，即使采用了 Category，也不能阻止外部的代码调用这个“私有方法”，只是 Xcode 不支持“私有方法”的自动完成，并会有警告提示，运行的时候，还是会成功的。各位看官知道有这么回事就可以了，这里不深讲。

变量和属性

类方法和实例方法

类方法

类方法就是 Java、PHP 里的 Static Method，不用实例化就能调。类方法有一个加号前缀。 示例：

类定义

```
@interface MyClass
    +(void) sayHello;
@end

@implementation MyClass

+(void) sayHello {
    NSLog(@"Hello, World");
}
@end
```

使用

```
[MyClass sayHello];
```

实例方法

实例方法就是 Java、PHP 里的普通方法，必须实例化才能调。实例方法有一个减号前缀。 示例：

类定义

```
@interface MyClass : NSObject
-(void) sayHello;
@end

@implementation MyClass

-(void) sayHello {
    NSLog(@"Hello, World");
}
@end
```

使用

```
mycls = [MyClass new];
[mycls sayHello];
```

Selector

selector 就是一个方法指针，类似 PHP 里的动态方法名：

```
<?php
class Hello {
    public function sayHello() {}

    public function test() {
        $fun_name = "sayHello";
        $this->$fun_name();
    }
}
```

在 Objective-C 里，selector 主要用来做两类事情：

绑定控件触发的动作

```
@implementation DemoViewController
- (void)downButtonPressed:(id)sender {//响应“按钮被按下事件”的方法
    UIButton *button = (UIButton*)sender;
    [button setSelected:YES];
}

- (void)drawAnButton {
    UIButton *btn = [UIButton buttonWithType:UIButtonTypeCustom];
    btn.frame = _frame;
    btn.tag = 1;
```

```
    btn.backgroundColor = [UIColor clearColor];
    [btn addTarget: self
         action: @selector(downButtonPressed:)
         forControlEvents: UIControlEventTouchUpInside];//当这个按钮被按下时，触发
downButtonPressed:方法
}
@end
```

延时异步执行

```
@implementation ETHotDealViewController
- (void)viewDidLoad {

    //获取数据源
    HotDealDataSource *ds = [[HotDealDataSource alloc]init];
    [ds reload];
    _items = ds.items;

    [self performSelector: @selector(refreshTable)
          withObject: self
          afterDelay: 0.5];//延迟 0.5 秒调用 refreshTable 方法
}

-(void)refreshTable
{
    [self.tableView reloadData];
}
@end
```

这个例子中，获取数据源是通过 ASIHTTP 组件异步调用服务端 HTTP 接口，refreshTable 要用到数据源返回回来的数据，如果不延迟 0.5 秒，就会立刻执行，执行的时候数据还在路上呢，页面就要变空白了。

继承

继承是写在 Interface 定义里面的。语法为：子类名在左，父类名在右，中间用冒号分隔。 示例：

```
@interface MyClass : NSObject
@end
```

对应的 Java 版本是：

```
public class MyClass extends NSObject {
}
```

协议（**Protocol**）

就是 Java、PHP 里的 Interface。

协议的定义

协议的定义用@protocol 关键字：

```
@protocol Printable
    -(void)print:(NSString)str;
@end
```

对应的 Java 版本是：

```
publilc interface Printable {
    public void print(String str);
}
```

协议的继承

协议本身也可以继承别的协议：

```
@protocol Printable <NSObject>
    -(void)print:(NSString)str;
@end
```

对应的 Java 版本：

```
public interface Printable extends NSObject {
    public void print (String str);
}
```

可选方法

协议可以包含可选方法，顾名思义，可选方法可以不被类实现：

```
@protocol Printable
@optional
    -(void)print:(NSString)str;
@end
```

加了@optional 关键字，一个类在 implements 这个协议时，便可以不实现 print:方法。

Java 里没有类似的实现，除了 Collection 里会有一些方法带有 optional 的注释，但 Collection 是个特例。

协议的实现

一个类实现某些协议是写在 Interface 定义里面的。语法为：协议名用尖括号包裹，多个协议名用逗号隔开，协议写在父类的右边（如果没有父类就直接写在子类右边）。

示例：

```
@interface  class MyClass : NSObject <Printable, Drawable>
@end
```

Printable, Drawablw 就是两个协议。

对应的 Java 版本是：

```
public class MyClass extends NSObject implements Printable, Drawable {
}
```

分类（Category）

分类可以给一个已经存在的类增加方法，而不用去改它的源码。Java 和 PHP 中都没有类似的特性。

比如说，NSObject 是一个 Objective-C 内置的系统类，我们想给它增加 toJson 方法，就像这样：

头文件：NSObject+Json.h

```
@interface NSObject (Json)
    -(NSString)toJson;
@end
```

实现文件：NSObject+Json.m

```
@implementation NSObject (Json)
    -(NSString)toJson {
        //...
    }
@end
```

使用的时候，只要包含 NSObject+Json.h，实例化 NSObject 类，就可以使用 toJson 方法了：

```
import "NSObject+Json.h"
@implatementation XYZController
    -(void)test {
```

```
        NSObject *obj = [[NSObject alloc]init];
        NSString *str = [obj toJson];
    }
@end
```

当然了，NSObject 本来的那些方法依然还是可以用的，什么都没变，只是多了个 toJson 方法。看起来是不是和继承没太多差别呢（除了使用的时候实例化的是 NSObject，而不是 JsonObject）？再看一个继承实现不了的例子：

头文件：NSObject+Json+XML.h

```
@interface NSObject (Json)
    -(NSString)toJson;
@end

@interface NSObject (XML)
    -(NSString)toXML;
@end
```

实现文件：NSObject+Json+XML.m

```
@implementation NSObject (Json)
    -(NSString)toJson {
        //...
    }
@end

@implementation NSObject (XML)
    -(NSString)toXML {
        //...
    }
@end
```

使用：

```
import "NSObject+Json+XML.h"
@implatementation XYZController
    -(void)test {
        NSObject *obj = [[NSObject alloc]init];
        NSString *json = [obj toJson];
        NSString *xml = [obj toXML];
    }
@end
```

Cocoa Touch

Cocoa 是 Mac OS App 的开发框架，Cocoa Touch 是 iOS 开发用的框架，Cocoa Touch 和 Cocoa 大部分是一样的，只是 Cocoa Touch 多了一些移动设备特有的东西，如：触摸屏、加速度传感器、GPS 定位。Cocoa 中多任务、多窗口的特性，在 Cocoa Touch 中也是没有的（或者跟 Cocoa 不完全一样的）。

就像学了 Java 语言还要再学一些 Spring、Hibernate、Struts（或者其它类似的 Java 类库）才能开始做 J2EE 应用一样，学过 Objective-C 语言之后，也要再学习 Cocoa Touch 框架才能顺利地开发 iOS 应用。

最常用设计模式之 Delegate

Cocoa Touch 大量使用 Delegate（委派）设计模式。

常用控件：按钮、文本块、图片、输入框

TableView

WebView

导航条

Xcode

运行

快捷键：Comman R

搜索

搜索文本

搜索文件

新建文件/目录

推荐在 Finder 中新建好的再添加进来

断点

模拟器和真机测试

模拟器测试

在 Xcode 中打开你的项目，在 Xcode 顶部工具栏的 Stop 按钮（Run 按钮右边那个黑色正方形按钮）右边，有个下拉菜单，显示着 “ToolBarSearch >

iPhone 5.0 Simulator”（即 你的应用英文名 > 当前选中的调试 ），点击这个下拉菜单，选中 iPhone 5.0 Simulator（这里的 5.0 是指 iOS 版本，不是 iPhone5 的意思，如果你的项目是 iPad 应用，请选 iPad 5.0 Simulator），再按“Run”按钮，Xcode 就会自动把当前正在编辑开发的应用编译并安装到模拟器上。

在模拟器上操作时，如果执行过程中遇到了你在 Xcode 里设置的断点，模拟器会暂停运行，并将当前活动窗口切换回 Xcode，供你调试。

在 Xcode 里增加或者取消了断点，不需要重新编译和安装应用即可生效。

切换被模拟的设备

模拟器的“硬件”菜单，可以选择想要模拟什么设备，有 iPad、iPhone 可选。

- Retina：表示视网膜屏，iPhone(Retina)代表 iPhone4，iPhone4S
- 4-Inch：表示 4 英寸的 iPhone，iPhone(Retina 4-Inch)就是 iPhone 5

切换模拟的 iOS 版本

在模拟器的“版本”菜单，可以选择要模拟什么版本的 iOS。设备和版本是彼此独立的，iPhone 4S 可以有 5.0，5.1，6.1 几种 iOS 版本，当然了，iPhone 5 不可能有 4.3 的 iOS 版本。

触摸屏

用鼠标点击（不区分左右键）模拟器上的 iPhone、iPad 屏幕，就是在模拟用手指触摸 iPhone，iPad 的屏幕，可以实现一些触摸效果比如：

- 鼠标单击 等于 手指轻触
- 鼠标长按 等于 手指长按（例如你可以在模拟器上长按应用 icon 调出删除应用的确认框）
- 鼠标按住拖动 等于 手指拖动
- 双击和单击模拟器的 Home 键也等于双击和单击真机的 Home 键

多指手势

多指手势比较复杂，在白苹果笔记本上可以模拟简单的双指手势，白苹果的触控板天然支持多指触摸，但要定位到模拟器的区域再响应多指手势就需要借助一些额外的键啦：

- 按住 Option 键，再用两个手指去操作触摸板，可模拟双指拖动、旋转
- 按住 Option+Shift，可模拟双指合拢

输入法和键盘

输入中文

手机上特有的输入法（比如九宫格输入法）不能模拟。模拟器默认的 iOS 软键盘只有英文输入，在测试应用的时候，我们要用到中文，有两个办法：

- 使用剪贴板，在 Mac OS 里复制，再到模拟器运行的应用中的输入框上长按鼠标（模拟手指长按）3 秒以上，等弹出“粘贴”的时候选择之，即可。
- 在模拟器里，按 Home 键，找到 Setting 那个 App icon（不是 Mac OS 顶部的模拟器菜单啊，那里没有 Setting 的），打开被模拟 iOS 设备的设置，依次点击”General - Keyboard - International Keyboards - Add New Keyboard…”，加个中文键盘，以后就可以使用被模拟 iOS 设备软件盘输入中文了，跟在 iPhone/iPad 真机上一样。

使用 Mac 电脑的键盘

如果要输入大量文本，使用模拟器里的软键盘效率太低，这时候可以使用物理键盘，方法是：在 Mac OS 顶部的模拟器菜单栏，点击”硬件”菜单，勾选下拉菜单中的“模拟硬件键盘”。以后再用模拟器运行 iOS 应用时，点击 iOS 应用中的输入框，软键盘就不弹出来了，可直接使用 Mac 电脑的物理键盘输入。

注意：

- 模拟器中的 iOS 接管了物理键盘输入，所以，调用的是模拟器 iOS 的输入法，不是你的 Mac 电脑的输入法。打个比方，你的 Mac OS 装的是搜狗五笔，模拟器中 iOS 加了个拼音输入法（Add New Keyboard），那么，在 iOS 应用中输入中文会调用拼音输入法。
- 要切换模拟器中 iOS 的中英文输入法，也只能按 iOS 设备软键盘上的小地球图标，按 Mac 电脑上的 Command+空格键是不行的。

地理位置

但 Mac 电脑没有定位用的硬件（GPS）和软件基础，因此模拟器不能自动获得当前的地理位置，不能用模拟器测试定位功能。（注意，虽然 WiFi 也可以独立定位——iPad WiFi 版没有 GPS 也可以定位，但 Mac 电脑的 WiFi 不具备定位相关的软件）

要在模拟器里测试依赖地理位置的功能（如”我附近的 xx”），可以手工指定一个经纬度给模拟器，方法：在 Mac 电脑顶部的模拟器菜单，点击”调试 - 位置 - 自定位置”，会弹出一个对话框，在弹出的框内填入经纬度即可。

如何获得经纬度？ 上谷歌地图（ditu.google.cn），在地图上找到你想要的位置（比如你想知道杭州大厦的位置，就在通过搜索框找到杭州大厦），点击右键，选择“这儿是什么”，搜索框中就会出现这个位置的经纬度了，前面是纬度，后面是经度。咱们天朝的版图，都是北纬和东经。

摄像头

Mac 电脑有摄像头，但 Mac OS 没有设计 API 给 iOS 模拟器调用，所以，不能用模拟器测试对焦闪光灯等功能。

要在模拟器上测试依赖照片的功能，可以在代码里做一个 workaround，即当代码检测到摄像头不可用时，弹出一个照片选择器，让测试人员从相册里选择一幅照片，来进行后续的操作（如照片美化、人脸识别、条码扫描）。

真机测试

模拟器能验证你开发的 iOS 应用的大部分功能，但有些 Mac 设备上不具备的硬件，模拟器是不能模拟的。前文提到了一个绕过这些限制的办法，但获取当前位置、拍照、加速度感应这些是模拟不了的，一款应用发布给消费者之前，必须要在真实设备上验证过。

将未提交 App Store 审核通过的应用安装到 iOS 设备上测试，有三种办法：

- 加入苹果的 Developer Program，成为付费会员，有了这个付费会员资格，就可以直接在 Xcode 中点击”Run”将刚刚改过的代码编译打包安装到开发测试用的 iOS 设备上。在 iOS 真机上操作被测试的程序能激活 Xcode 中设置的断点。
- 越狱 iOS 设备。将 iPhone 和 iPad 越狱后，可以通过 SSH 直接上传 Xcode 编译好的 ipa 包（一个 iOS App 本质上就是一个 ipa 包）。
- 越狱的 iOS 设备，配合破解过的 Xcode，甚至可以实现和付费开发者计划一样的功能：在 Xcode 上点击”Run”，就自动编译安装到 iOS 设备上去运行了
- 企业部署方案。就像阿里巴巴的轩辕剑一样，用 iPhone/iPad 访问这个网址，点击里面的轩辕剑链接就可以安装轩辕剑这个应用了。

破解 Xcode 是违法行为（越狱是合法的），而且挑版本挑得厉害，不是所有 Xcode 版本都能破解，也不是所有 Xcode 的破解版都能和越狱的 iOS 配合好。越狱+SSH 上传跟企业部署一样效率低（部署效率低，无法激活 Xcode 中的断点），只能用于 QA 验收，不适合开发自测。综上所述，最适合开发实时测试的就是第一个正规途径了。下面重点讲这个：

拥有一个开发者账号

苹果的 Developer Program 分为个人开发者和公司开发者，分别是每年 99 美元和每年 299 美元，分别可以注册 100 台和 500 台苹果测试设备。这个台数限制在一个付费年度内不会清空，比如说，2013 年 4 月 1 日付费成功的，付费会员资格在 2014 年 3 月 31 日之前有效，这期间，注册一台就少一个名额，哪怕这个设备注册进来用了之后一分钟马上又删掉了，减少的这个名额也不会回来。

在交钱之前，最好问一下，周围的同事，有没有已经交了钱的。如果有，你只需要注册一个免费的 Apple ID（就是你在 App Store 安装软件用的 Apple ID），请他发个邀请邮件给你，把你的 Apple ID 加入他的团队就可以了，苹果会认为你们两个人是一个团队的，你们分别用自己的账号，共享 100 台设备的限额，这是合法的。

安装证书和私钥

证书

不想看下面各种点击各种页面跳转的直接用浏览器访问证书管理，你要登录你就用 Apple ID 登录（前提是交过钱，或者找交了钱的人把你加入团队了）。

不嫌烦，或者想知道下次没我这个文档的时候怎么进证书管理吗？按这个步骤操作：

- 进入 苹果开发者中心
- 点击 iOS Dev Center
- 点蓝色“Login”按钮，用你的 Apple ID 登录，登录成功会跳到 开发者首页
- 点击右上角的 iOS Provisioning Portal（别找了，直接 Command F 搜索多好）
- 点左侧菜单栏里的 Certificates

页面上有一个“Your Certificate”区域，下方有个 Download 圆角按钮，这是你的个人证书，下载下来。再下面一行，有一句“If you do not have the WWDR intermediate certificate installed, click here to download now”，这个是苹果的公共证书，也下下来。

双击下载回来的证书，装证书时，会提示你输入密码，这是【钥匙串访问工具】在问你要你的 Mac OS 账号开机密码（相当于 linux 里面的 sudo），不是 Apple ID 的密码，不要搞错了。

安装私钥

如果你是和其它同事公用的账号，让他给你一个私钥即可，就是一个扩展名为 p12 的文件，双击之，钥匙串访问会自动出来，需要你输入一个密码，这个密码问给你 p12 文件的人要，不是你的 Mac OS 系统开机密码，也不是你的 Apple ID 密码。

将设备注册到 Provisioning Portal

- 打开 Xcode，从 Xcode 的 Window 菜单中找到 Organizer，打开之（Shift Command 2）。
- 把 iOS 设备连上电脑，Organizer 会自动识别出你的设备，并显示在左侧边栏。

- 在 Organizer 左侧边栏找到你的设备，右键，点击“Add Device to Provisioning Portal”，然后等 Organizer 提示你操作成功即可。（选中设备后，右边设备详情区域会显示一个按钮“Use for Development”，点它也可以）。

到 iOS 真机上运行测试版程序

回到 Xcode 主界面，在 Stop 按钮（Run 按钮右边那个黑色正方形按钮）右边，有个下拉菜单，显示着 “ToolBarSearch > iPhone 5.0 Simulator” （即 你的应用英文名 > 当前选中的调试 ），点击这个下拉菜单，选中你的真机设备名，再按“Run”按钮，Xcode 就会自动把当前正在编辑开发的应用编译并安装到真机上测试啦！

发布到 App Store

打 IPA 包

IPA 包本质上是一个 ZIP 压缩包，只不过它有着特殊的目录结构，扩展名是 ipa，制作方法如下：

- 在 Xcode 中 Build 项目，快捷键 Command B
- 在左侧项目导航器中，展开 Products 文件夹，找到你要打包的应用，你的应用名.app，右键，选择 show in finder
- 到 Finder 中 Copy 这个.app 目录（选中，按 Command C），复制到一个你新建的名为 Payload（区分大小写）的文件夹中
- 找到你的应用 Logo，即一个 512 * 512 像素的 PNG 文件，copy 到与 Payload 一起（与 Payload 并列，不要放进 Payload 了），并重命名为 iTunesArtwork（区分大小写，没有扩展名）
- 将 Payload 目录、ItunesArtwork 文件打成一个 zip 包，并更改扩展名为 ipa
- 双击这个 ipa 文件，会用 iTunes 打开，如果打开成功，且在 iTunes 里有应用 Logo 显示，就成功了

批量自动打包

除 App Store 外，还有许多其它的 iOS 应用市场（如 91 助手，同步推等等），如果一个应用需要发布到很多个应用市场，且他们的代码略有不同（比如说，统计代码不同），按上述方法手工修改源码再打包，再还原，比较容易出错。好消息是，Xcode 是有命令行的，我们可以写一个 shell 脚本，先用 se 自动修改源码，再调用 Xcode 的命令行来编译以得到 your——app.app 目录，最后调用 zip、mv 等命令把上一个章节讲的 ipa 打包动作自动执行。

阅读应用代码

从头新建一个应用：Hello World

其它

代码里的控件尺寸

iOS App 里的控件尺寸和字体大小都是指 Point，Retina 设备（iPhone 4，4S，5；the new Pad）和非 Retina 设备（iPhone 3GS，iPad，iPad 2）的 Point 数是一样的，尽管 iPhone 4 的分辨率是 3GS 的 2 倍。比如说，10point 在 Retina 设备里是 20 pixel，在非 Retina 设备（iPhone 3G）上则是 10 pixel。

项目成员间交流时，应使用 Point，不要使用 pixel。

SVN 操作含有@符号的文件

iOS 应用中经常出现 xxxx@2x.png 这样的文件名,它们是给 retina 设备用的高分辨率大图，用 svn 命令行操作它们的时候会被@符号干扰,解决方案是在 svn 命令末尾加上一个@符号,如:

```
svn del icon@2x.png@
svn info Default@2x.png@
```

如果一次移动了几十个 png 文件再 svn commit 的,可以用 shell 批处理:

```
svn status | awk '($1=="!"){print $2}' | grep -v @ | xargs svn del
```

上面这个命令是将文件名不包含@符号的,且已经不在硬盘上的文件从 svn version controll 中删掉

```
for file in `svn status | awk '($1=="!"){print $2}' `; do svn del $file"@"; done
```

上面这个命令是将文件名包含@符号的,且已经不在硬盘上的文件从 svn version controll 中删掉

svn add 同上，如法炮制即可.

Xcode 中的代码结构与操作系统上的文件系统并不一致

推荐在 Finder 里建好目录再到 Xcode 的 Project Navigator 中点击“Add Files to”添加到项目中

iPhone 5 适配

iPhone 5 与之前的 iPhone 不一样，采用了 4 寸 Retina 屏，所以它的 Point 数变成了 320 * 568 points

开源代码

- Apple 官方的 Sample Code
- 维基百科上的开源 iOS App
- iOS Opensource --Domain Parking 了，以前可以下载 Twitter 和 Wordpress 客户端的
- code 4 app
- UI 4 app， code4app 的姐妹站

Objective-C 教程

- Apple 官方教程
- Cocoa Dev Center
- 维基上的 Objective-C 语言简介 --中文，十分钟可读完，推荐