让不懂编程的人爱上iPhone开发(2013秋iOS7版)-第1篇

博客：blog.sina.com.cn/eseedo

微博：eseedo

微信：iseedo

Email: eseedo@gmail.com

说明：本系列教程仅针对入门新手，高手勿入！时间是最宝贵的财富，只选择自己需要的信息才是王道！

主要素材和示例项目来源：http://www.raywenderlich.com/store

需要特别强调的是,iOS7和Xcode5相比iOS之前的版本和Xcode4有巨大的差异，所以这系列教程只好重新编写。

特别说明：不适合程序猿出身的童鞋！目的是让产品策划，设计或管理人员了解iPhone开发的基础知识，便于今后和项目组沟通，而不是以成为编程高手为学习目的。

适合看本系列教程的对象：

1.从未学过编程，或者对Objective-C语言一无所知，但要懂一些常用的英语单词

2.从未学过iPhone/iPad开发

3.喜欢苹果，充满想象力，喜欢创造，同时也愿意承受学习的压力，愿意投入时间和精力

主要内容：

1. 编程入门
2. iOS入门
3. 游戏开发入门

如果你对iOS开发已具备丰富的经验，请不要在此浪费时间！

简介：

人天生就喜欢游戏，所以我们要开发的第一款应用不会是苍白无力的”Hello World”，而是一个小游戏，名为Bull’s Eye(拖拖看)。虽然这个游戏非常简单，但如果你从未接触过编程，可能还是会遇到一些困难。但是不要担心，即便你第一遍接触这些新概念的时候还有些含糊不清，但我们会在整个系列的教程中不断重复，直到它们成功的进入你的潜意识，甚至在梦中都不会忘记~

需要提醒大家的是，学习一门语言或工具的最好方式是练习和实践。因此，对于初学者来说，千万不要只是看过一遍了事，而应该自己手动敲入所有的代码，甚至故意修改其中的代码，刻意制造一些bug，然后想办法解决。而在学完本教程之后，要立即开始实战，同时多看苹果官方的示例代码和Github里的示例。不要害怕麻烦和错误，在解决麻烦和修正错误的过程中，你能更深入的领会为何要这样做，而不仅仅是简单的copy和paste。

前面说了，本系列教程是针对完全的菜鸟来设计的。也就是说，哪怕你是个完全不懂编程的文科生，我们也有信心让你爱上iPhone开发。当然，如果你懂一点编程知识，学习起来会快很多。如果你是从PHP或Java语言转过来的，可能会被Objective-C的低级特性和奇怪的语法吓倒。但别害怕，大多数编程语言的工作原理是类似的，它们的相似之处要远远大于不同之处。只要你学过任何一门其它语言，那么对Objective-C就会很快入手。当然，对于从C++，甚至汇编语言转过来的开发者，会发现Objective-C要相对简单一些。

对当前的主流开发语言难度排个序，大致如下（从最难到最简单）：

机器语言 >汇编> C++ >Objective-C > C, Lisp,Prolog>C# > Java > Python > PHP > Javascript, ActionScript,Ruby

正如刚才所提到的，对于汇编以下难度的语言，只要真正学懂一门，其它的是一通百通，毕竟真正的程序猿和攻城师很少只会一门开发语言的。只懂一门开发语言能活到现在的要吗是某个方面的顶级专家，要吗就是走了技术转管理的路线。

在我们的教程中，不会也不可能教你学习所有和iPhone,iPad开发的知识。iOS SDK（开发工具包）非常庞大，除了苹果的官方技术文档，市面上没有任何一个教材可以涵盖iOS开发的全部内容。我们只会教你了解Objective-C和iOS开发所需具备的核心基础。一旦你掌握了建筑技术，可以自己去探索iOS开发的其它细节。

除了Objective-C语言和iOS开发工具包的相关知识，我们最重要的目的是让你学会程序猿的思维方式。一旦你具备了这种思维方式，可以完成任何编程任务，不管是游戏，工具，网络应用还是其它你能想到的东西。作为一个程序猿，需要思考解决各种计算问题，并创

造性的想出解决方案。一旦掌握了解决问题的方法，不论多复杂的问题都可以解决。这才是本系列教程的终极目的，让不懂编程的人爱上开发！

我可以百分百保证的是，你在学习的过程中一定会遇到各种问题。程序代码中会出现无数莫名其妙的bug，让你不知所措。但即便是一个拥有20年以上编程经验的程序猿，也会经常遇到这样的问题。我们只是人类，而人类的大脑在处理复杂计算问题的时候总会出错的。不要害怕出错，但我们会提供一些思维工具，教会你如何填平自己挖的坑。

在我身边有很多人学习iPhone开发的方式是:

从大量的博客和网站中拷贝粘贴代码，而完全不理解这些代码的工作原理，以及该如何将这些代码嵌入到自己的项目之中。从网络中寻找解决方案是一种高效的工作方式，但你必须真正的理解这些代码的作用，才能举一反三。

在本系列教程中，我们从一开始就会学习如何构建真正的应用，而不是所谓的baby应用，或是仅仅为了学习目的而设计的简单示例。我们会详细解释其中的每一步操作，并附上丰富的图片帮助大家来理解。

通过这些步骤，你将在制作这些有趣应用的同时逐渐掌握编程的思维和技能。当你最终学完本系列教程后，应该已经掌握了Objective-C和iOS开发工具包的精髓。更重要的是，你应该学会了如何用程序猿的思维方式来编程和解决问题，并真正开始制作属于自己的应用。对此，我有百分之一千的信心！

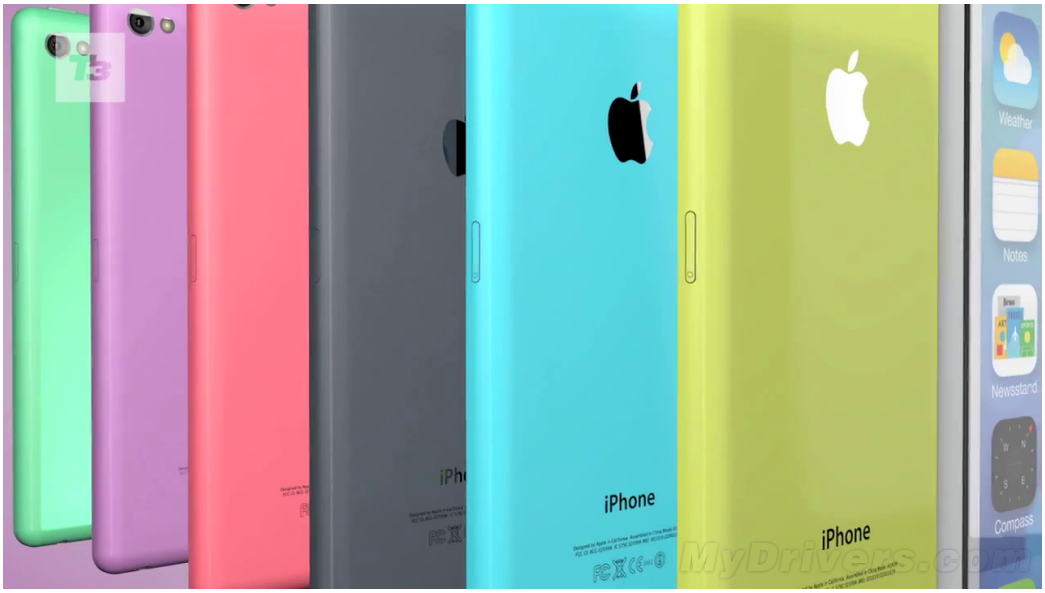
那还等什么，让我们就此开始吧！

iOS6, iOS7...

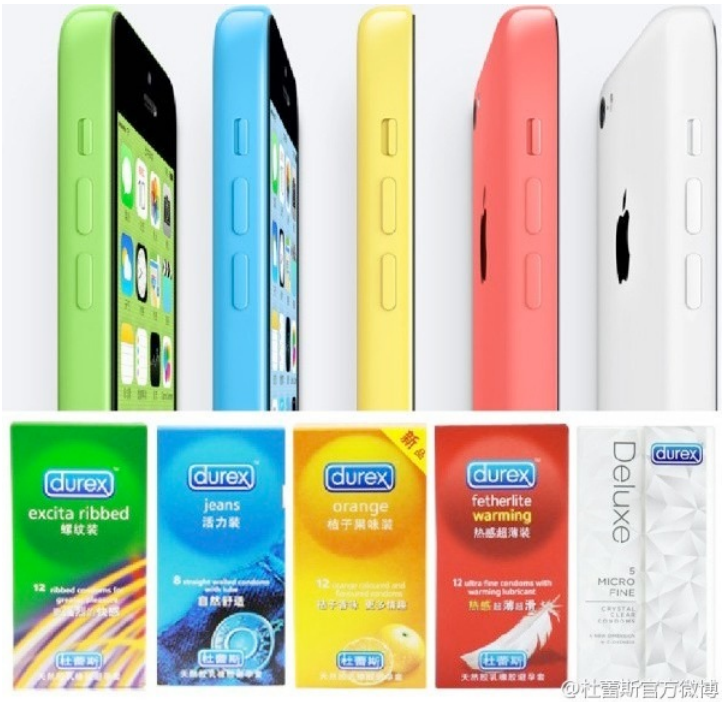
时光飞逝，世事无常。自2007年1月Macworld上乔帮主那一次惊天地泣鬼神的演讲至今，竟然已经6年了！6年前，Nokia藐视群雄，Motorola和三星争斗不休，众多国产品牌手机和山寨手机还在华强北幸福的收割着打工者腰包里不多的毛爷爷。6年后，Nokia亏损连连，被Elop的木马计成功收入微软旗下，Motorola被Google收入帐下，当年的手机三雄竟然只剩下三星还活得风生水起。苹果帝国好不容易占据半壁江山，却不幸遭遇王者的离去，新的Cook船长迟迟拿不出创新之作，iPhone5S和iPhone5C的发布在让苹果股价狂跌的时候，同时也让人概叹天才企业家对于一个公司的重要性。。。。

轻松一刻:

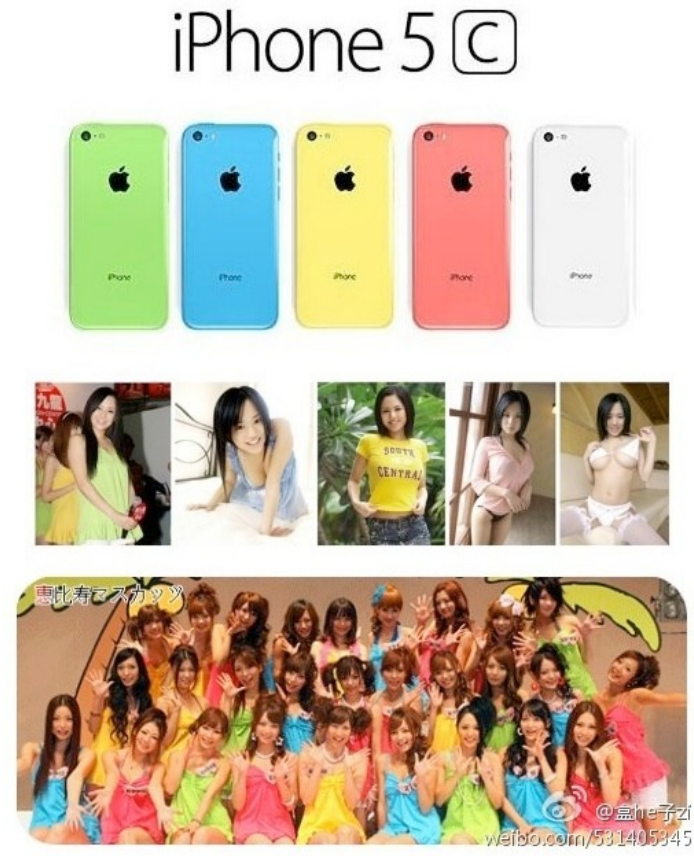
厨子Cook眼中的iPhone5C



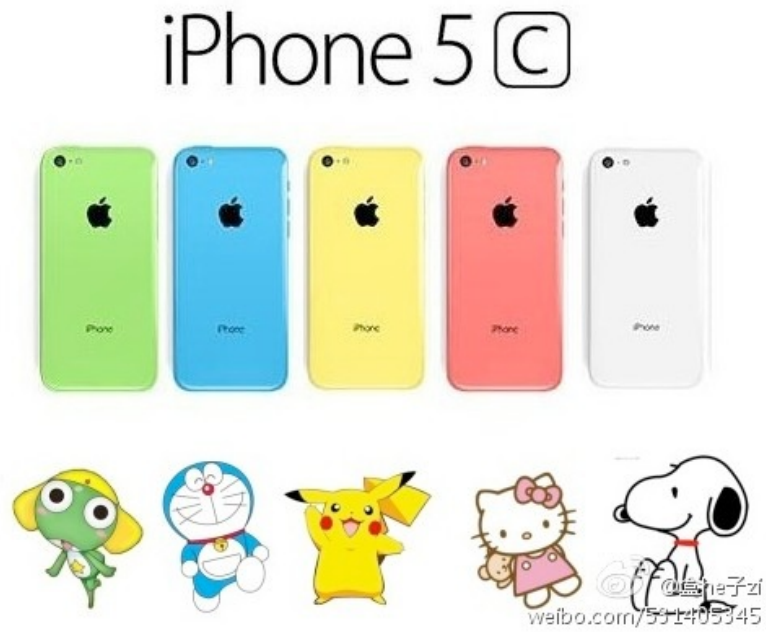
糕富帅眼中的iPhone 5C



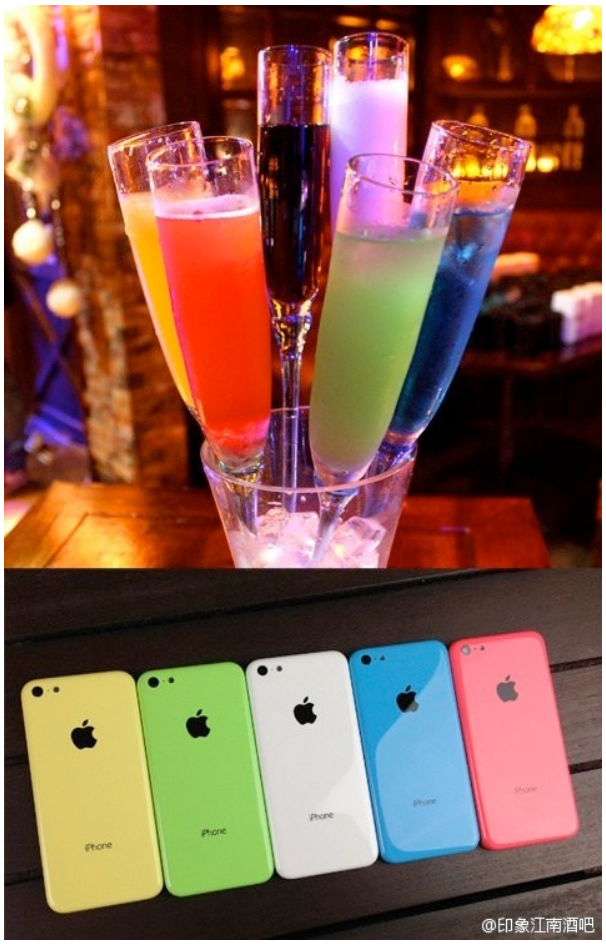
屌丝眼中的iPhone 5C



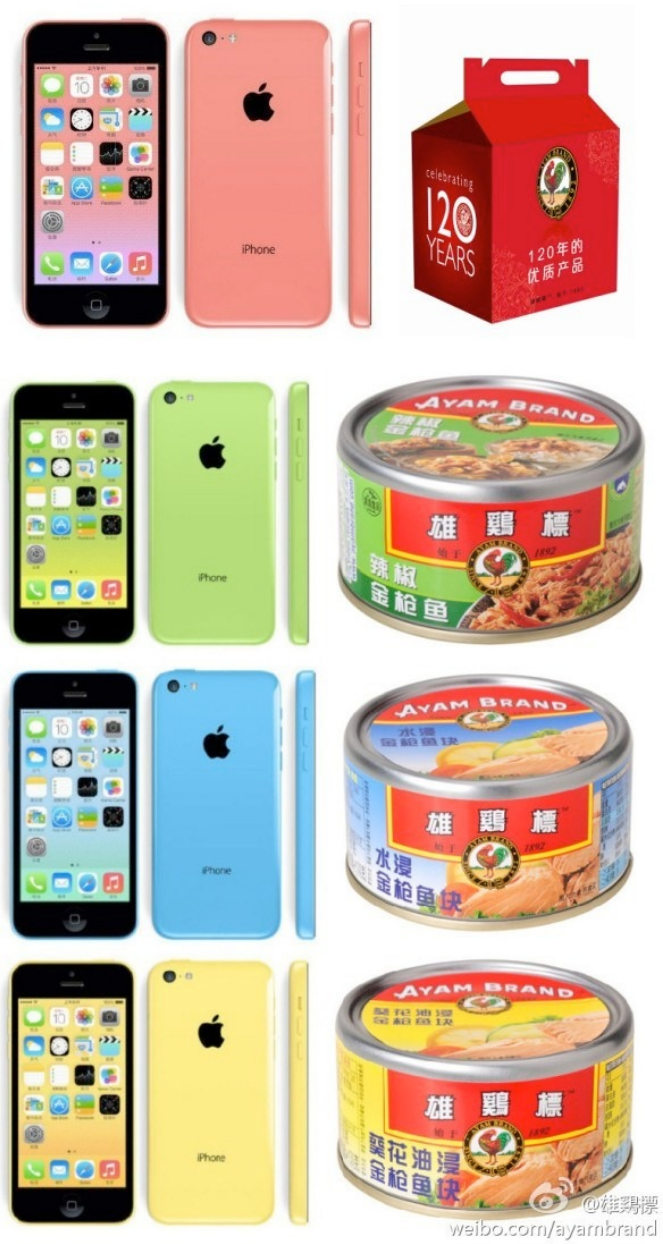
萌妹子眼中的iPhone5C



夜店公主眼中的iPhone5C



吃货中的iPhone5C

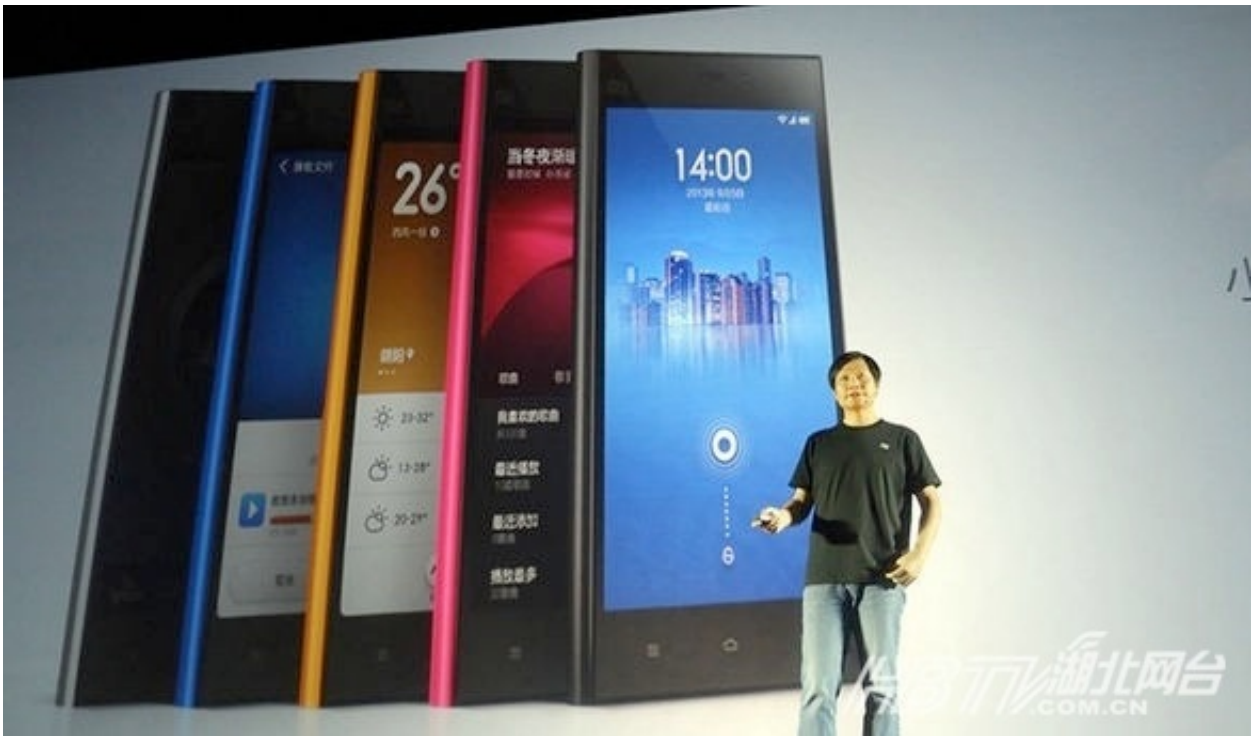


好了，开个玩笑而已，吐槽到此结束。

或许我们对苹果的诸多吐槽，也是因为对它又爱又恨的缘故吧。

如果将移动互联网市场比作三国，那么苹果如同曹魏，可惜雄才大略的曹操已去；Google阵营如同江东猛虎，进退自如，实力雄厚；而微软（加收购的诺基亚）则如同偏居一隅的蜀汉，死守住最后的一块阵地。如果不是微软CEO老鲍的辞职，这场三国演义几乎就要成了绝代双骄的对决。iOS和Android的地位类似当年Windows和Linux系统，而windows phone则几乎成了当年桌面时代的Mac系统。当年硬件和软件厂商各行其是的局面被如今软硬件一体化的大势所替代。

当然，这只是全球市场的盛况。但是别忘了，最近Google负责Android的高级副总裁加入了我朝的小米，这一场战火的主战场也逐渐从合众国蔓延到了天国。



打住打住，还是谈谈iOS7吧。

其实上面要说的是，移动互联网行业的发展速度太快了，iPhone操作系统到现在已经到了第七代iOS7。本系列教程将完全基于iOS7，并支持其中的一些新特性，比如Automatic Reference Counting(ARC)，Storyboard。在iOS5之前，iPhone应用开发中必须考虑手动管理内存，而iOS5新引入的ARC会解决这个老大难问题。

既然是学习一门全新的工具，自然要从最新的版本学起，因此，iOS7是我们的第一选择。

有舍才有得

学习iPhone开发不但可以作为一门兴趣爱好，同样还可以带来不错的收益（如果你能给用户带来不错的产品，或者找到一份提供给力薪水的工作~）。但天下没有白吃的午餐，进行iOS开发也是要花钱的。以下是你需要投资的：

1.一台设备：

iPhone,iPad,iPad mini或iPod touch中的任一种。只用虚拟机永远没法学会真正的开发。当然，为了支持最新的iOS系统，就不要买太老型号的设备了，比如iPhone3GS之类的。

2.一台使用Intel内核处理器的Mac电脑：

需要安装Mac OS X 10.8（Mountain Lion）操作系统或者即将推出的最新的Mac OS X 10.9(Mavericks)。建议电脑的内存在4G以上，否则你会很痛苦的。。。

有人说可以用虚拟机在PC上开发，我的建议是，宁可买一台二手的MAC，也不要用PC开发。否则，你会遇到各种莫名其妙的问题，而且对提高你的编程思维没有任何帮助。

3.一个付费的iOS开发者账号

有了这种账号，你也可以学习Tiny Wings展翅高飞了。

如何申请付费的iOS开发者账号

考虑到篇幅，本系列教程不会专门讲这个，大家可以参考cocoachina中的相关文章（http://www.cocoachina.com/bbs/read.php?tid=13372&fpage=2）。

强大的Xcode

Xcode是开发iPhone应用的主要工具。Xcode带有一个文本编辑器，可以让你敲入自己的代码，同时还有一个可视化的工具由来设计应用的用户界面。Xcode可以将你编写的源代码编译成可执行的应用，并在模拟器(Simulator)或设备上进行测试。同时，Xcode还带有一个debugger(调试器)，用于帮助你发现代码中的错误（很遗憾，目前它还没法自动帮你修复bug，这一天的到来还需要更给力的人工智能)

.

有两种方式下载最新版的Xcode，一种是注册后从iOS 开发者网站下载（developer.apple.com)，需要登录。而另一种则是从Mac App Store中下载。

本系列教程用的Xcode版本是最新的Xcode 5，而iOS版本是7.0GM。

关于计算机语言

语言是一种沟通工具。很多时候我们以为iPhone只是一部手机，其实它的内核是一个非常先进的微型计算机，只是同时具备打电话的功能而已。和其它计算机一样，iPhone是通过数字电路的0，1指令来工作的。如果我们编写软件在iPhone上运行，就必须把源代码翻译成计算机可以理解的0，1指令。

几十年前，人们不得不使用0，1指令和计算机直接交流。而随着汇编和高级语言的出现，大多数的编程语言变得更接近于日常生活所使用的英语。这样一来，人们更容易理解编程语言的使用。但同时也需要将人类可以理解的语言翻译成计算机可以理解的0，1指令。

举例而言，计算机内部会使用以下的语言：（不要关注其中的细节，你现在还看不懂）：

```
Ltmp96:
    .cfi_def_cfa_register %ebp
    pushl   %esi
    subl    $36, %esp
Ltmp97:
    .cfi_offset %esi, -12
    calll   L7$pb
L7$pb:
    popl    %eax
    movl    16(%ebp), %ecx
    movl    12(%ebp), %edx
    movl    8(%ebp), %esi
    movl    %esi, -8(%ebp)
    movl    %edx, -12(%ebp)
    movl    %ecx, (%esp)
    movl    %eax, -24(%ebp)
    calll   _objc_retain
    movl    %eax, -16(%ebp)
    .loc    1 161 2 prologue_end
Ltmp98:
    movl    -16(%ebp), %eax
    movl    -24(%ebp), %ecx
    movl    L_OBJC_SELECTOR_REFERENCES_51-L7$pb(%ecx), %edx
    movl    %eax, (%esp)
    movl    %edx, 4(%esp)
    calll   _objc_msgSend_fpret
    fstps   -20(%ebp)
    movss   -20(%ebp), %xmm0
    movss   %xmm0, (%esp)
    calll   _lroundf
```

事实上，计算机真正看到的指令如下：

```
00011001010011110100100011001111100101
00101000100111101011011100111010110100 1
01010001110011111010111011000011100011 0
10010000011100010100110100111110011001 11
```

上面的movl和calll指令只是为了方便人类理解。但即便如此，对我个人来说这种语言还是令人望而生畏。今天的编程语言是下面这样的（先不要深入细节，看看而已）：

```
void HandleMidiEvent(char byte1, char byte2, char byte3, int deltaFrames)
```

```
{
char command =(byte1 & 0xf0);

if(command == MIDI_NOTE_ON && byte3 !=0)
{
      PlayNote(byte2 + transpose, velocityCurve[byte3]/ 127.0f, deltaFrames);
}
elseif((command == MIDI_NOTE_OFF)
       ||  (command == MIDI_NOTE_ON && byte3 ==0))
{
      StopNote(byte2 + transpose, velocityCurve[byte3]/ 127.0f, deltaFrames);
}
elseif(command == MIDI_CONTROL_CHANGE)
{
if(data2 ==64)
          DamperPedal(data3, deltaFrames);
elseif(data2 == 0x7e || data2 == 0x7b)
          AllNotesOff(deltaFrames);
}
}
```

看到这里或许你才有点感觉了。即便你没有任何编程经验，但只要懂英语，就大概能判断出上面代码的意思。以上代码是从一个音效同步工具的程序中截取的。它使用C语言编写，这门语言是上世纪60年代开发的，人们用它开发了著名的Unix操作系统（今天所有操作系统的鼻祖，包括Windows,Mac,Linux）。当然，iOS的内核也是基于Unix系统的。

用来开发iOS应用的语言被称为Objective-C，它是标准C语言的扩展。使用Objective-C可以完成C语言所能完成的任何工作。同时它还添加了很多有用的特性，比如最重要的面向对象编程（Objective-Oriented）。Objective-C在前些年可谓门庭冷落，无人问津，除了铁杆的Mac粉丝，几乎濒临灭绝。但随着2007年那一次伟大的iPhone产品发布后之后，几乎要被历史遗忘的Objective-C语言再次进入人们的视线，甚至成为今的主流开发语言。Objective-C是2011和2012年的年度编程语言No.1（2009,2010年是年度编程语言的No.2)。

最新的2013年8月编程语言排行榜

Position Aug 2013	Position Aug 2012	Delta in Position	Programming Language	Ratings Aug 2013	Delta Aug 2012	Status
1	2	↑	Java	15.978%	-0.37%	A
2	1	↓	C	15.974%	-2.96%	A
3	4	↑	C++	9.371%	+0.04%	A
4	3	↓	Objective-C	8.082%	-1.46%	A
5	6	↑	PHP	6.694%	+1.17%	A
6	5	↓	C#	6.117%	-0.47%	A
7	7	=	(Visual) Basic	3.873%	-1.46%	A
8	8	=	Python	3.603%	-0.27%	A
9	11	↑↑	JavaScript	2.093%	+0.73%	A
10	10	=	Ruby	2.067%	+0.38%	A
11	9	↓↓	Perl	2.041%	-0.23%	A
12	15	↑↑↑	Transact-SQL	1.393%	+0.54%	A
13	14	↑	Visual Basic .NET	1.320%	+0.44%	A
14	12	↓↓	Delphi/Object Pascal	0.918%	-0.09%	A--
15	20	↑↑↑↑↑	MATLAB	0.841%	+0.31%	A--
16	13	↓↓↓	Lisp	0.752%	-0.22%	A
17	19	↑↑	PL/SQL	0.751%	+0.14%	A
18	16	↓↓	Pascal	0.620%	-0.17%	A-
19	23	↑↑↑↑	Assembly	0.616%	+0.11%	B
20	22	↑↑	SAS	0.580%	+0.06%	B

由于排名算法的调整，从2013年8月开始Java跃居第一，而 Objective-C从第三的位置被C++语言赶到第四。

这里不得不提到C++语言，事实上C++和Objective-C语言几乎是同时出现的。和Objective-C语言的简洁不同，C++语言几乎包含了所有可能的特性。作为一门编程语言，它非常强大，且执行效率超高。事实上，所有的操作系统，以及大量的网络游戏，主机游戏和PC游戏，游戏引擎都会使用C++来开发。C++的问题在于，对于一个新手来说，它异常复杂，包括了基本语言结构，面向对象开发和模板、标准库等诸多内容。学习C++还是颇有难度的，仅次于汇编语言。不过C++11（2011年的新标准）这一C++的最新版本在很多方面做了大的改进，相信会让这门“古老”而又强大的编程语言更加褶褶生辉。

在进行iOS应用或游戏开发的时候，我们可以混合使用C,C++和Objective-C（简称为Objective-C++）。此外，在WWDC 2013中，官方还特别介绍了如何在原生应用中嵌入Javascript。

总之，对于iPhone应用开发来说，最主要接触的语言是Objective-C，偶尔也会用到C++和C，以及javascript等脚本语言。

对于iPhone游戏开发来说，由于Cocos2d-x引擎的迅速普及，C++的使用频率也大大增加。

考虑到本教程的很多读者从未接触过任何编程语言，这里对其它几个主流语言的特点和作用稍微说明一下：

1.Java语言是当今最普遍使用的开发语言，它简单易学（相对C++,C和Objective-C），且跨平台性非常强，对网络开发的支持令人称赞。很多企业使用Java语言来开发商业相关的网络应用。此外，Java语言也是开发Android应用的必备工具。

2.C语言是几个主流开发语言(Java,C++,C#,Objective-C)的根基所在。常有人说，学好C语言，其它的语言就会一通百通。因此对硬件底层性能的支持超强，它的主要应用领域是嵌入式开发、游戏引擎开发等偏底层的部分。

3.PHP语言主要用于开发网络应用（特别是web服务器端，也就是用户不可见的部分，如结合MySQL进行后台数据传输处理等），相对其它几门语言，它非常容易上手。但它的局限性在于除了web应用，对其它应用的开发力不从心。

4.Javascript语言主要用于开发Web前端（也就是用户可见的部分），随着HTML5技术的兴起，Javascript语言必将是未来三到五年的主流Web开发工具。

5.C#语言是微软为了对抗Java语言的强势而自行开发的一种编程语言。它和Java一样简单易学（同样是相对的），但只能支持微软的平台。闻名业界的.NET就是C#语言的最佳搭配。关于C#有个更新点，目前最火爆的移动平台3D游戏开发引擎Unity3D主要支持C#和javascript开发，而windows手机平台的卷土重来也让C#有了新的机会。但随着微软在移动互联网领域的式微，C#的地位和前几年比起来大有下降。

6.Python,Ruby,Perl同PHP语言的作用类似，属于脚本语言，对于开发网络应用非常高效。其中Python和另一种脚本语言Lua还常在游戏中作为脚本语言使用。

7.Go语言，一门全新的系统级语言，由Google开发，于2009年发布。虽然它的历史非常短暂，但根据目前的发展来看,Go语言有望在未来十年成为一款成功的系统级语言。Go语言功能强大，可以替代C++

8.Basic（Visual Basic)语言，曾经风骚一时，若干年前很多编程入门课程必教的开发语言。其学习曲线非常平缓，易于上手，但实际项目中用到的不是很多。

9.SQL语言，这是目前最重要的关系数据库操作语言，其影响已经超出数据库领域，在很多其它领域得到采用，比如人工智能领域的数据检索，软件开发工具中嵌入SQL的语言等。SQL语言是一种交互式查询语言，允许用户直接查询存储数据，但它并不是完整的程序语言，没有DO或FOR类似的循环语句，但可以嵌入到另一种语言中，通过接口发送到数据库管理系统。

10.汇编语言，虽然现在是高级编程语言的天下，但性能超强的直接面向硬件的汇编语言仍然在嵌入式开发领域占据着一席之地。只是汇编语言和硬件本身的关联很大，所以普及性一般。

11.LISP语言，一种相对冷门的函数式编程语言，其长处在于超强的运算能力。如今在人工智能领域和CAD绘图软件中仍有大量的支持者。

12.Erlang语言，一个结构化，动态类型编程语言，内建并行计算支持。起初是由爱立信专门为通信应用设计的，比如控制交换机或变换协议等，非常适合于构建分布式并行计算系统。

其它语言相对来说比较冷僻，或者曾经热门但如今使用的人很少，用不到的时候可以不管。

关于这些编程语言，我目前正在写一个《编程语言的故事》系列，将用故事的形式，按照历史发展的脉络将一些主要编程语言的发展做一些简要介绍，请关注。

科普知识到此结束，我们不打算对Objective-C语言的特性做详细的介绍，不然很可能5分钟不到你就睡着了。我们将在创建项目的过程中一步步解释你所遇到的语言。包括什么是变量，什么是对象，如何调用方法（发送信息）等等。

当然，如果你需要一本随时可以查询的工具书，我们强烈推荐这本书给你：

Stephen G. Kochan编写的<Programming in Objective-C>

好了，有了这么多的基础做铺垫，我们可以进入正式的开发了！