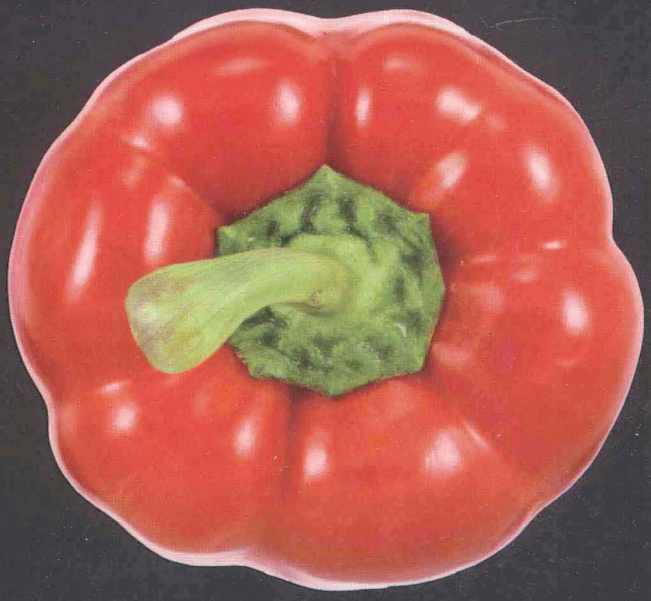
移动开发经典丛书

学会运用流行速度最快的语言进行编程



Objective-C 开发经典教程

Beginning Objective-C

[美] James Dovey
Ash Furrow 著
冯宝隆 于鹏飞 译

Apress®

清华大学出版社

移动开发经典丛书

Objective-C 开发经典教程

[美] James Dovey
Ash Furrow 著
冯宝隆 于鹏飞 译

清华大学出版社
北 京

James Dovey, Ash Furrow
Beginning Objective-C
EISBN：978-1-4302-4368-7

北京市版权局著作权合同登记号　图字：01-2013-5118

图书在版编目(CIP)数据

Objective-C 开发经典教程/(美) 达维(Dovey, J.)，(美) 弗罗(Furrow, A.) 著；冯宝隆，于鹏飞 译.
—北京：清华大学出版社，2014
(移动开发经典丛书)
书名原文：Beginning Objective-C

ISBN 978-7-302-34667-8

Ⅰ. ①O…　Ⅱ. ①达…　②弗…　③冯…　④于…　Ⅲ. ①C 语言—程序设计—教材　Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字(2013)第 291226 号

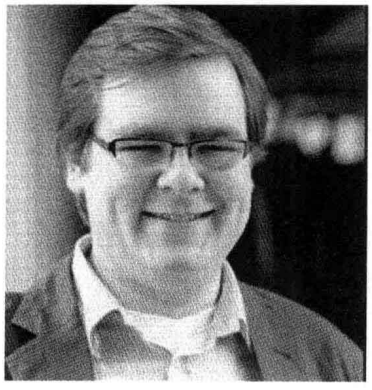
责任编辑：王　军　杨信明
装帧设计：牛艳敏
责任校对：邱晓玉
责任印制：沈　露

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：清华大学印刷厂
装 订 者：三河市溧源装订厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　张**：21.5　　**字　数**：523 千字
版　　次：2014 年 1 月第 1 版　　**印　次**：2014 年 1 月第1 次印刷
印　　数：1～4000
定　　价：59.80 元

产品编号：054094-01

作 者 简 介

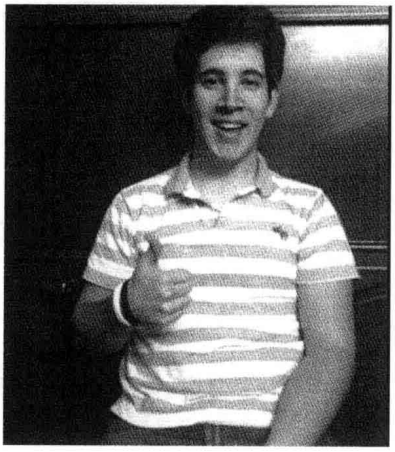
James Dovey 已经独立为 Macintosh(以及之后的 iOS)编写了 12 年软件。他出生于英国，在加拿大多伦多柯保公司(Kobo)工作，在那里一直担任公司 iOS 应用程序首席架构师，但最近他担任出版业和各种标准委员会的联系人，并且在办公室里扛着一个上面写着“实现 ePub 3”的大棒子(实际上这个大棒子看起来像是某种马里特木锤——读者可自行用 Google 搜索它)。作为一个黑客(这是个问题吗？请问我们能把它当作一个问题吗？)，他是很多开源项目的建立者，这些开源项目包括 AQGridView，该项目是原始的 iOS 表格视图控件；AQXMLParser，iPhone 最好的基于事件的 XML 解析器；以及原始的 Apple TV 第三方开发工具。他还开发了 Outpost，即最初的 iPhone Basecamp 客户端，还建立了基于 Apple TV 的数字信号系统。本书是他的第一本书，但他希望将来能出更多的书。



Ash Furrow 从 iOS 2 开始就在编写 iOS 应用程序。在完成他的学士学位时，他开发了用于地方选举的 iOS 应用程序并在新不伦瑞克大学教授 iOS 开发。他还开发了几个自己的应用程序(在 App Store 上销售)并发布开源项目。在 2011 年，他为了在 500px 工作移居到多伦多并开发了现在广为流行的 iOS 应用程序。

当前，Ash 是 500px iOS 组的首席开发者。他还喜欢发推特、写博客和摄影。

技术审校者简介



Felipe Laso Marsetti 是自学成才的 iOS 软件开发者，他当前作为系统工程师受雇于 Lextech Global Services。尽管过去使用过很多语言，但他认为没有比开发 iPhone 和 iPad 项目更让他高兴的事情了。Felipe 有超过两年的 iOS 专业经验。他喜欢在 http://iFe.li 上写博客，在 www.raywenderlich.com 上撰写 iOS 教程和文章，以及担任 Objective-C 和 iOS 相关书籍的技术审校者。Felipe 的推特账号为@Airjordan12345，Facebook 账号为其名字，App.net 账号为@iFeli。在他不工作或编程时，他喜欢阅读、学习新语言和技术、看体育比赛、烹饪或弹吉他和拉小提琴。

致　谢

如果不是在 2009 年苹果世界大会上偶遇 Jeff LaMarche，这一切就不会发生，之后他在那一年的苹果全球开发者大会上把我介绍给了 Apress 的 Clay Andres。Apress 大家庭的作者和编辑给了我很大的帮助和鼓励，尤其是 Felipe Laso Marsetti，他在确保本书中有价值信息的正确性方面提供了无价的帮助，还有编辑 Katie Sullivan、Douglas Pundick 与 Steve Anglin，他们尤其应该得到奖励，因为在过去的一年里他们忍受了我道格拉斯·亚当斯式的应对最后期限的方法。

——James Dovey

我得到了很多帮助，包括在我撰写本书内容时和获得一个职位时得到的帮助。在该职位上我得到了足够的经验，从而能帮助编写这本书。没有人能只靠自己获得成功，每个人都会在他们的前进道路上得到帮助。实在有太多朋友、老师和导师需要感谢。我专门把我的想法讲给 Jason Brennan 和 Paddy O'Brien 听，把我的文章给他们看，他们在帮助我完善写作方面总是能提供无价的帮助，感谢他们的敏锐眼光。

在我撰写这本书期间，我的妻子给了我绝对的支持，她容忍我在深夜和周末继续工作，没有她本书就不能得以顺利付梓。

——Ash Furrow

目　录

第1章

Objective-C 入门

Objective-C 编程语言有一段很长的历史，但在过去的大多数时间里，Objective-C 只是一种一直被冷落在边缘地带的潜力级语言。iPhone 引入它之后，迅速给它带来了声誉(或者骂名)。2012 年 1 月，TIOBE 官方宣布 Objective-C 获得了 2011 年的 TIOBE 编程语言奖。这个奖是颁发给在之前 12 个月中使用量增长最快的语言的。对于 Objective-C 而言，2011 年它的使用量从 TIOBE 排行榜的第 8 位快速提升至第 5 位。你可以在图 1-1 中看到它快速、迅猛的上升趋势。

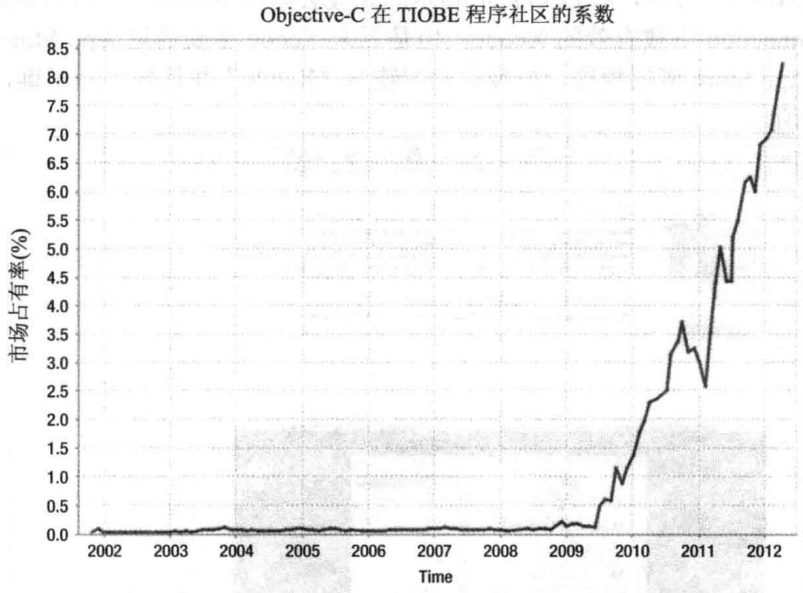


图 1-1　2002 年 1 月至 2012 年 1 月 Objective-C 的使用趋势

Objective-C 编程语言诞生于 20 世纪 80 年代初期，由 StepStone 公司的 Brad Cox 和 Tom Love 创建。它的设计目的是，使用 C 语言将 Smalltalk(由 Xerox PARC 公司于 20 世纪 70 年代创建)的面向对象编程思想带给全世界使用 C 编程语言实现的软件系统。1988 年，Steve

Jobs(对，就是那个大家都知道的 Steve Jobs)授权在 NeXT 操作系统中使用来自 StepStone 的 Objective-C 语言和运行时。NeXT 也在 GCC 中嵌入了 Objective-C 编译器，并且开发了 FoundationKit 和 ApplicationKit 框架，这些工作直接为 NeXTstep 操作系统的编程环境打下了基础。虽然 NeXT 电脑并没有席卷全球，但它使用 Objective-C 构建的开发环境已经在软件产业圈内得到了广泛的好评。在 20 世纪 90 年代中期，这个操作系统最终发展成为 OpenStep 标准，被 NeXT 公司和 Sun Microsystems 公司所使用。

1997 年，正在为下一代操作系统寻找一个坚实基础的苹果公司收购了 NeXT 公司。NeXTstep 操作系统从那时起成为 Max OS X 的基石，并于 2001 年早期发布了第一个 Max OS X 的商业版本。当旧的 Mac OS 系统可以兼容 AppKit 和 Foundation 库(从那时起就在市场上被称作 Cocoa)之后，这两个库构成了 OS X 上新的编程环境的核心。NeXT 公司的编程工具——Project Builder 和 Interface Builder——可以在 Mac OS X 上免费下载，但直至 2008 年，它才有了支持 iPhone SDK 的版本。从此，众多程序员开始积极为这个让人激动的新设备编写程序，Objective-C 也开始了它的腾飞之路。

在本章你将要学习如何使用 Xcode 编程环境创建一个简单的 Mac 应用程序，包括实现 UI 布局与用户交互。你将会接触到一些 Objective-C 语言本身的细节，如关键字、结构体、Objective-C 程序的格式，以及语言本身提供的一些功能。

1.1 Xcode

为 Mac 和 iPhone 编写程序主要使用苹果公司的免费工具集，该工具集主要扩展了 Xcode 集成开发环境(IDE)。过去，存放 OS X 副本的光盘会附带 Xcode，也可以通过前往 Apple Developer Connection 下载有效的 Xcode。但是近来 Xcode 主要通过 App Store 获取。打开你的 Mac 的 App Store 应用程序，在搜索栏中输入“Xcode”并且按下回车键，就可以找到它，如图 1-2 所示。

图 1-2　最新版本的 Xcode 可以在 Mac App Store 上免费下载

单击下载它，等待一段时间后就可以在 Applications 文件夹中找到可用的 Xcode 了。

与其他同名的 IDE 应用程序相比，Xcode 附带了很多东西。它包含了很多有用的调试和配置功能，而且可以选择下载 GCC 和 LLVM 编译套件的命令行版本。你将找到的现有工具中包含如下这些工具：

- Instruments：该工具可以为你的应用程序生成详细的运行时配置信息——这很可能是所有工具中对 Max 或 iOS 开发者最有用的工具。
- Dashcode：一个 HTML 和 JavaScript 编辑器，旨在帮助你轻松构建 Dashboard 组件和 Safari 插件。
- Quartz Composer：这个应用程序允许你使用无代码拼接组装技术创建复杂的图形转换、过滤器和动画。
- OpenGL Apps：为了使用 OpenGL(以及 iOS 上的 OpenGL ES)进行工作，该应用程序提供了完整的应用程序套件。在该套件中可以找到配置文件、性能监视器、着色器生成器，以及 OpenGL 驱动监视器。
- Network Link Conditioner：对于基于网络的软件工程师而言，一个梦想已经变为现实，这个方便的小工具可以让你模拟不同网络配置的主机。默认情况下，它模拟最常遇到的环境。你也可以设定带宽、丢包率、延迟和 DNS 延迟，创建自己的网络环境。想要测试你的 iOS 应用程序在 Wi-Fi 网络极边缘地带时会如何处理吗？使用这个小工具会使其变得完美且简单。

以上这些只是我们最喜爱的工具中的一部分，但它决不是一个详尽的清单。你将在本书的后文中看到很多更加令人印象深刻的 Xcode 工具背后的技术。

1.2　创建你的第一个项目

第一次打开 Xcode 时，会看到 Xcode 的欢迎界面。下面的步骤能引导你创建一个新的项目。

(1) 单击名为 Create a new Xcode project 的按钮，然后选择要创建项目的类型。

(2) 在 Mac OS X 选项中，选择 Application，然后在主窗格内选择 Cocoa Application 图标。

(3) 单击 Next 按钮，会出现一些定义项目的选项，输入图 1-3 所示的细节信息，然后再次单击 Next 按钮并且选择项目存放的位置。

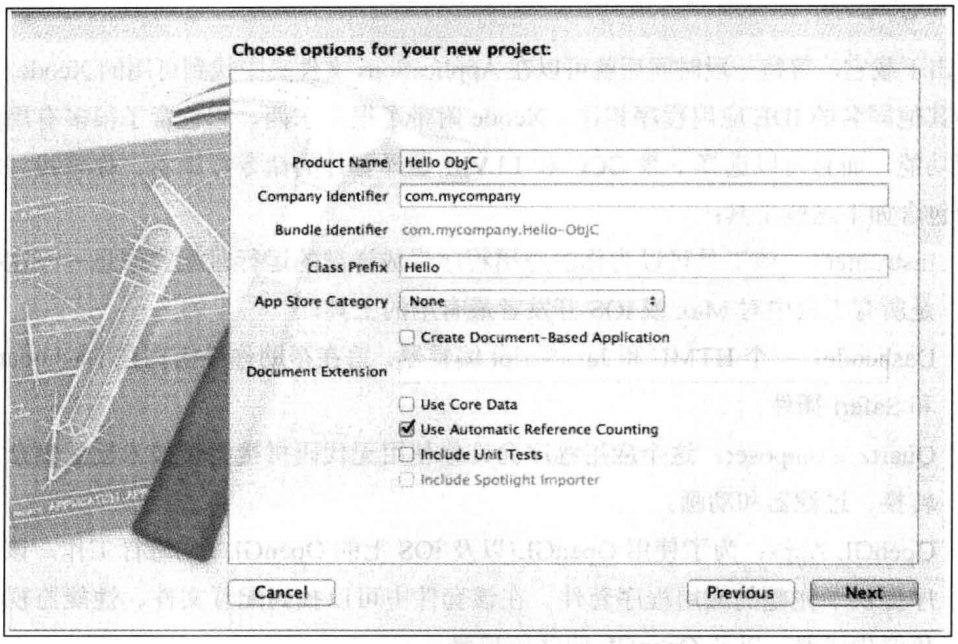


图 1-3　你的第一个项目的选项

浏览一下 Xcode 和这个新项目的大致布局。在窗口的左边可以看到导航栏，如图1-4 所示。在此可以浏览项目的源代码文件、资源、库和输出文件。这个导航栏也能让你浏览项目的类层次结构，在整个项目中执行查找和替换工作，以及浏览编译日志。

Xcode 窗口中间的窗格是编辑区。可以在此编写代码以及使用你的用户界面资源。

右边是实用工具窗格。上侧则会根据当前在编辑器窗格的焦点的内容而敏感地显示不同的选项卡。下侧是分隔板面板，可以从这里拖出用户界面元素、基于模板的新文件、代码片段以及媒体。也可以在这里添加自己的模板和代码片段。

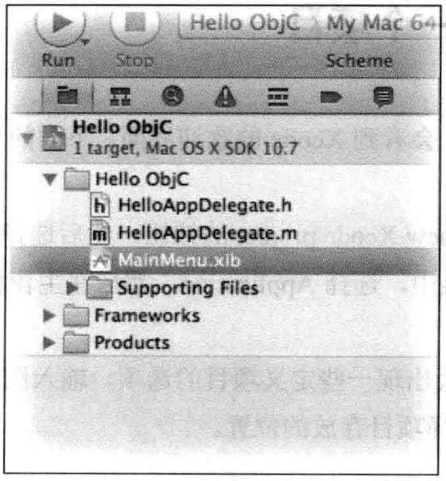


图 1-4　Xcode 导航栏窗格

1.2.1　应用程序模板

Cocoa Application 模板已经为你生成了很多信息。实际上，在这里你已经有了一个功能完善的应用程序。在导航栏选择浏览器选项卡(最左侧的选项)并且查看 Hello ObjC 文件夹中的内容，会看到一些主要的源文件和用户界面定义文件(一个.xib 文件)。这里还有一个 Supporting Files 文件夹，它包含了应用程序的 main.m 文件，该文件是整个应用程序自身的入口，同时作为前缀部分自动添加到项目中每个文件的开头。你还能看到 Hello ObjC-Info.plist 文件，它包含了应用程序的元数据；还有 InfoPlist.strings 文件，它保存了 plist 文件中的本地化版本的数据。通常不需要直接改动这些文件，因为 Info.plist 一般通过目标编辑器进行编辑，本书会在后面的章节中对此进行介绍。

在此你可能需要改动的一个文件是 Credits.rtf。这个文件的内容会显示在应用程序的 About 对话框中。而且因为它是一个.rtf 文件，所以你能随心所欲地设计它。这些内容将会显示在 About 对话框下可滚动的多行文本框中。

再向下是 Frameworks 文件夹。它包含了一个你的应用程序运行所基于的所有框架和动态库的列表。需要注意的是，这不是一个自动管理的列表：需要在使用框架(framework)和库时自己将它们添加到项目中。最后，Products 文件夹包含了一个指向编译过的应用程序的引用。现在它的名字呈红色显示，那是因为你还没有构建它。

单击 HelloAppDelegate.h，可以在编辑器窗格中看到它的内容。现在它看起来有点空，就像程序清单 1-1 显示的那样。这段代码声明了一个类的结构和接口，在这个例子中这个类的名称是 HelloAppDelegate。这段代码通知系统该类执行所有定义在名为 NSApplication-Delegate 的协议中的必要方法，并且该类有一个名为 window 的属性。在第 2 章中将介绍这个语法的细节部分，至于现在，你只需要相信它是按照我们的期望工作的。

程序清单 1-1　HelloAppDelegate.h

```
#import <Cocoa/Cocoa.h>

@interface HelloAppDelegate : NSObject <NSApplicationDelegate>
@property (assign) IBOutlet NSWindow *window;
@end
```

接下来介绍执行文件，参见程序清单 1-2。现在它的内容同样很简单。在定义 HelloApp-Delegate 类的实现的一些分隔符中，可以看到一个名为@synthesize 的指令，它看起来与你刚刚看到的 window 属性有关。事实正是如此，这个指令可以让 Objective-C 编译器生成获取(get)和设置(set)window 属性的方法，为你节省了自己编写这些方法所需的时间。它同时指定了用来存储该属性的实例成员变量的名称为_window。编译器也会为你创建该成员变量，这也为你节省了显式编写它的时间。

程序清单 1-2　HelloAppDelegate.m

```
#import "HelloAppDelegate.h"

@implementation HelloAppDelegate
@synthesize window = _window;
```

```
- (void)applicationDidFinishLaunching:(NSNotification *)aNotification
{
    // Insert code here to initialize your application
}
@end
```

1.2.2 界面生成器

如果选择了 MainMenu.xib 文件，编辑区会切换至界面生成器模式。之所以命名为“界面生成器”，是因为生成用户界面的任务直至最近才被改为一个名为“界面生成器”的单独应用域。你能看到如图 1-5 所示的界面生成器。

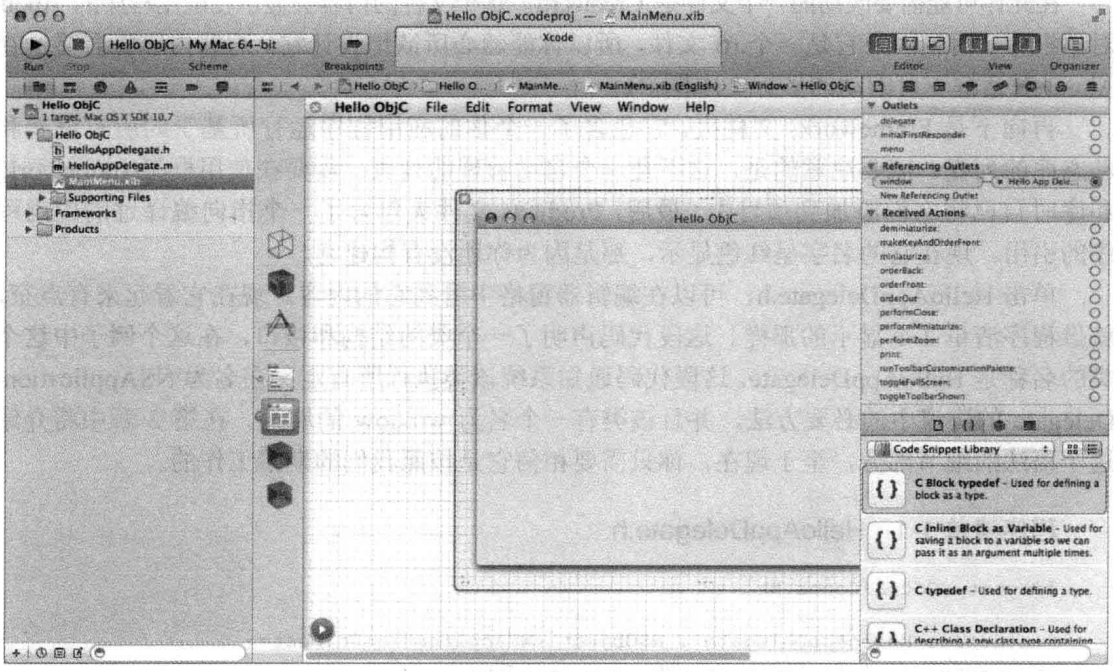


图 1-5 界面生成器

在图 1-5 中，可以看到应用程序的菜单。编译器的左下方是文档缩略图。这个界面文档中的所有对象都会在此列举。顶部是一些“推断(inferred)”对象，它们会出现在所有(或者是几乎所有)的.xib 文档中。分割线的下方是已经被显式添加到.nib 文件中的对象。列表中的第二项是应用程序的窗口。选择后可以让它出现在编辑器中。

现在你已经选中了它，Xcode 窗口右侧的实用工具窗格上方出现了很多选项卡。亲自点点这些选项卡看看会显示什么吧。将鼠标悬停在选项卡的小图标上，就会显示一个工具提示，告诉你这个选项卡的名称。

界面生成器具有很多强大的幕后智能特性，使你可以通过用户的输入和动态反馈构建优秀的应用程序。更重要的是，只需要给项目添加三行代码就能完成这件事情。

1.2.3　用户界面控件

首先，需要给用户提供一个可供输入的位置。可以从实用工具窗格下半部分的对象面板中取出控件。在图 1-6 中能看到你将用到的所有控件。

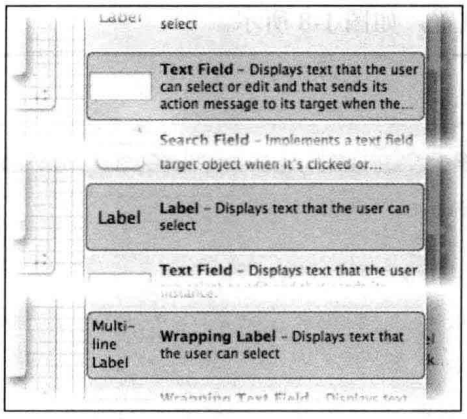


图 1-6　Text Field、Label 和 Multiline Label 控件

(1) 在实用工具窗格的下方，选择从右侧数的第二个选项卡(图标是个盒子)，切换到用户界面 Object Palette。

(2) 向下拉 Object Library 弹出式菜单，并且选择 Controls，将面板的内容限制为此时只显示标准控件(见图 1-6)。

(3) 需要找到的第一个控件是 Text Field。略微向下滚动就能找到它。

(4) 现在将该行从面板直接拖动到编辑器的窗口。你会发现这样做能使它变成一个真正的文本框。

(5) 将它向上拖动到窗口内容区域的右上角，会出现蓝色的引导线协助你定位。就把它放在那里，如图 1-7 所示。

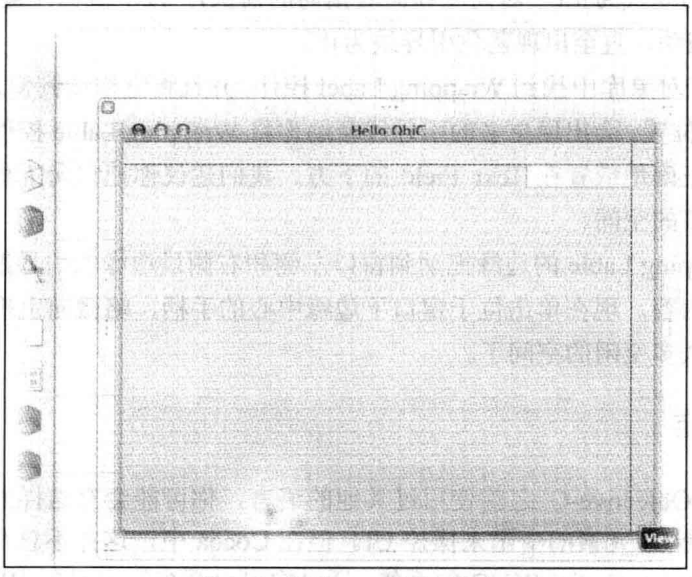


图 1-7　放置 Text Field 控件

(6) 接下来，需要找到 Label 控件(可以认为它是一个没有特殊背景色的、不可编辑的 Text Field 控件)。

(7) 将 Label 拖动到界面左上角，但需要注意的是，在把它拖动到左上角的过程中，划过刚放置的 Text Field 控件的底部和其内部基线时也会出现蓝色引导线。后者就是你要找的地方。将 Label 放在该位置，如图 1-8 所示。

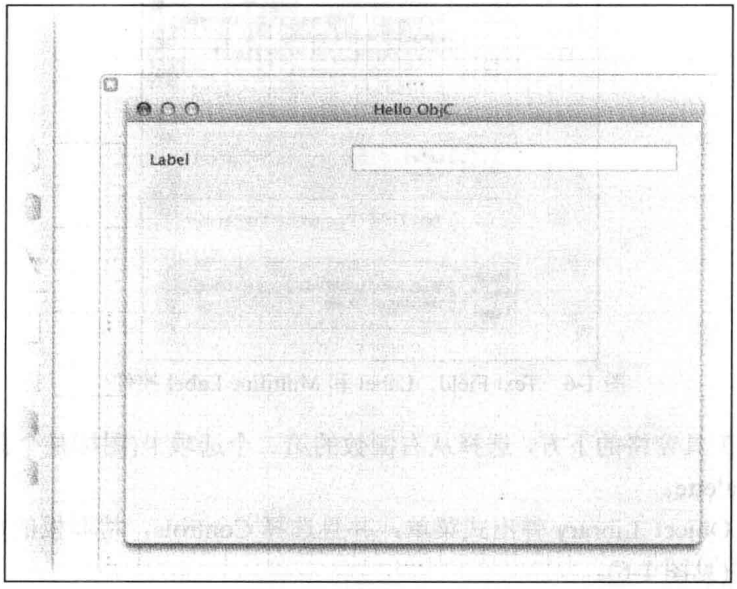


图 1-8 放置 Label 控件

(8) 双击 Label 可以编辑它的内容。输入“Your Name:”，然后按下回车键保存这个改动。现在同时按下⌘和=，让 Label 的大小变得跟它的文本大小一致。

(9) 选中 Text Field，并且将鼠标光标向 Text Field 的左侧边缘处移动，直到光标变为可以重定义尺寸的模式为止(一对向左和向右指向的箭头)。按下鼠标左键并且将 Text Field 的边缘向 Label 拖动，直至出现蓝色引导线为止。

(10) 最后，在对象库中找到 Wrapping Label 控件，并且将它拖动到窗口的中间，比 Text Field 略微偏下的位置。会出现更多的引导线帮助你将 Wrapping Lable 控件移动到窗口横向中心的位置并且正确地放置在 Text Field 的下方。我们建议你把它向下移动得远一些，使它周围可以有更多的空间。

(11) 将 Warpping Lable 的边缘拖动到窗口左侧和右侧最边缘的引导线位置，这可以使文本显示较多的内容。现在单击位于窗口下边缘中心的手柄，略微向上拖动来缩小窗口，这样这里就没有太多空闲的空间了。

1.2.4 界面绑定

如果在学习 Objective-C 之前使用过其他的语言，你可能会有这样的想法：通过挂钩(hooking)引用可变 UI 元素的变量来操控 UI。但在 Cocoa 中，这并不总是必要的。相反有一个被称为 Key-Value Coding(KVC)的系统，通过它可以观察指定对象中的指定值，该值通