

## <<Objective-C基础教程>>

### 图书基本信息

书名：<<Objective-C基础教程>>

13位ISBN编号：9787115314581

10位ISBN编号：7115314586

出版时间：2013-5

出版时间：人民邮电出版社

作者：[美] Scott Knaster,[美] Waqar Malik,[美] Mark Dalrymple

译者：周庆成

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## <<Objective-C基础教程>>

### 内容概要

《图灵程序设计丛书:Objective-C基础教程(第2版)》结合理论知识与示例程序,全面而系统地介绍了Objective—C编程的相关内容,包括Objective—C在C的基础上引入的特性、Cocoa工具包的功能及框架,以及继承、复合、源文件组织等众多重要的面向对象编程技术。附录中还介绍了如何从其他语言过渡到Objective—C。

## <<Objective-C基础教程>>

### 作者简介

Scott Knaster Mac开发界传奇人物，曾就职于苹果公司，帮助开发人员编写早期平台的Mac软件。目前在谷歌工作，负责维护Google Mac Blog。

著有多部程序员必读的经典著作，如How to Write Macintosh Software、Hacking Mac OS X Tiger和Macintosh Programming Secrets。

Waqar Malik UNIX迷，Macintosh控。

早期曾就职于苹果公司，帮助开发Cocoa和Unix。

目前创立了技术咨询公司Crimson Research，从事Cocoa应用开发工作。

Mark Dalrymple资深Mac和Unix程序员，有20多年开发经验，曾开发过跨平台工具包、因特网发布工具、高性能Web服务器和终端用户桌面应用。

另外，他还与人合著过Core Mac OS X and Unix Programming和Advanced Mac OS X Programming（《Mac OS X高级编程权威指南》）。

## 书籍目录

第1章 启程 1 1.1 预备知识 1 1.2 历史 1 1.3 内容简介 2 1.4 准备工作 3 1.5 小结 5 第2章 对C的扩展 6 2.1 最简单的Objective—C程序 6 2.2 解构HelloObjective—C程序 10 2.2.1 #import语句 10 2.2.2 框架 11 2.2.3 NSLog ( ) 和@“字符串” 11 2.3 布尔类型 14 2.4 小结 18 第3章 面向对象编程的基础知识 19 3.1 间接 19 3.1.1 变量与间接 20 3.1.2 使用文件名的间接 22 3.2 在面向对象编程中使用间接 28 3.2.1 过程式编程 28 3.2.2 实现面向对象编程 34 3.3 有关术语 37 3.4 Objective—C语言中的OOP 38 3.4.1 @interface部分 38 3.4.2 @implementation部分 41 3.4.3 实例化对象 43 3.4.4 扩展Shapes—Object程序 45 3.5 小结 47 第4章 继承 48 4.1 为何使用继承 48 4.2 继承的语法格式 51 4.3 继承的工作机制 53 4.3.1 方法调度 54 4.3.2 实例变量 55 4.4 重写方法 57 4.5 小结 59 第5章 复合 61 5.1 什么是复合 61 5.2 自定义NSLog ( ) 62 5.3 存取方法 65 5.3.1 设置engine属性的存取方法 66 5.3.2 设置tires属性的存取方法 67 5.3.3 Car类代码的其他变化 68 5.4 扩展CarParts程序 69 5.5 复合还是继承 70 5.6 小结 71 第6章 源文件组织 72 6.1 拆分接口和实现 72 6.2 拆分Car程序 75 6.3 使用跨文件依赖关系 77 6.3.1 重新编译须知 78 6.3.2 让汽车跑一会儿 79 6.3.3 导入和继承 81 6.4 小结 82 第7章 深入了解Xcode 84 7.1 窗口布局一览 84 7.2 改变公司名称 85 7.3 使用编辑器的技巧 86 7.4 在Xcode的帮助下编写代码 87 7.4.1 首行缩进 ( 美观排版 ) 88 7.4.2 代码自动完成 88 7.4.3 括号配对 90 7.4.4 批量编辑 91 7.4.5 代码导航 94 7.4.6 集中精力 96 7.4.7 使用导航条 97 7.4.8 获取信息 98 7.5 调试 101 7.5.1 暴力测试 101 7.5.2 Xcode的调试器 101 7.5.3 精巧的调试符号 101 7.5.4 开始调试 101 7.5.5 检查程序 104 7.6 备忘录 105 7.7 小结 106 第8章 FoundationKit介绍 107 8.1 稳固的Foundation 107 8.2 使用项目样本代码 107 8.3 一些有用的数据类型 108 8.3.1 范围 108 8.3.2 几何数据类型 109 8.4 字符串 109 8.4.1 创建字符串 110 8.4.2 类方法 110 8.4.3 关于大小 111 8.4.4 字符串比较 111 8.4.5 不区分大小写的比较 112 8.4.6 字符串内是否还包含别的字符串 113 8.4.7 可变性 114 8.5 集合大家族 115 8.5.1 NSArray 115 8.5.2 可变数组 119 8.5.3 枚举 120 8.5.4 快速枚举 121 8.5.5 NSDictionary 122 8.5.6 请不要乱来 124 8.6 其他数值 124 8.6.1 NSNumber 124 8.6.2 NSValue 125 8.6.3 NSNull 126 8.7 示例：查找文件 126 8.8 小结 130 第9章 内存管理 131 9.1 对象生命周期 131 9.1.1 引用计数 132 9.1.2 对象所有权 134 9.1.3 访问方法中的保留和释放 134 9.1.4 自动释放 136 9.1.5 所有对象放入池中 136 9.1.6 自动释放池的销毁时间 137 9.1.7 自动释放池的工作流程 138 9.2 Cocoa的内存管理规则 140 9.2.1 临时对象 141 9.2.2 拥有对象 141 9.2.3 垃圾回收 143 9.2.4 自动引用计数 144 9.3 异常 154 9.3.1 与异常有关的关键字 155 9.3.2 捕捉不同类型的异常 156 9.3.3 抛出异常 156 9.3.4 异常也需要内存管理 157 9.3.5 异常和自动释放池 158 9.4 小结 159 第10章 对象初始化 160 10.1 分配对象 160 10.1.1 初始化对象 160 10.1.2 编写初始化方法 161 10.1.3 初始化时要做些什么 162 10.2 便利初始化函数 163 10.3 更多部件改进 164 10.3.1 Tire类的初始化 165 10.3.2 更新main ( ) 函数 166 10.3.3 清理Car类 168 10.4 Car类的内存清理 ( 垃圾回收方式和ARC方式 ) 171 10.5 指定初始化函数 172 10.5.1 子类化问题 173 10.5.2 Tire类的初始化函数改进后的版本 175 10.5.3 添加AllWeatherRadial类的初始化函数 175 10.6 初始化函数规则 176 10.7 小结 176 第11章 属性 177 11.1 使用属性值 177 11.1.1 简化接口代码 178 11.1.2 简化实现代码 179 11.1.3 点表达式的妙用 182 11.2 属性扩展 183 11.2.1 名称的使用 186 11.2.2 只读属性 188 11.2.3 自己动手有时更好 189 11.2.4 特性不是万能的 189 11.3 小结 189 第12章 类别 191 12.1 创建类别 191 12.1.1 开始创建类别 192 12.1.2 @interface部分 193 12.1.3 @implementation部分 194 12.1.4 类别的缺陷 195 12.1.5 类别的优势 196 12.1.6 类扩展 196 12.2 利用类别分散实现代码 197 12.3 通过类别创建前向引用 200 12.4 非正式协议和委托类别 201 12.4.1 iTunesFinder项目 202 12.4.2 委托和类别 204 12.4.3 响应选择器 205 12.4.4 选择器的其他应用 206 12.5 小结 206 第13章 协议 207 13.1 正式协议 207 13.1.1 声明协议 207 13.1.2 采用协议 208 13.1.3 实现协议 209 13.2 复制 209 13.2.1 复制Engine 210 13.2.2 复制Tire 211 13.2.3 复制Car 212 13.2.4 协议和数据类型 215 13.3 Objective—C2.0的新特性 215 13.4 委托方法 216 13.5 小结 218 第14章 代码块和并发性 219 14.1 代码块 219 14.1.1 代码块和函数指针 219 14.1.2 Objective—C变量 223 14.2 并发性 224 14.2.1 同步 224 14.2.2 队列也要内存管理 227 14.2.3 操作队列 229 14.3 小结 231 第15章 AppKit简介 232 15.1 构建项目 232 15.2 创建委托文件的@interface部分 234 15.3 InterfaceBuilder 235 15.4 设计用户界面 236 15.5 创建连接 239 15.5.1 连接输出口 ( IBOutlet ) 239 15.5.2 连接操作 ( IBAction ) 240 15.6 应用程序委托的实现 242 15.7 小结 244 第16章 UIKit简介 245 16.1 视图控制器 249 16.2 小结 263 第17章 文件加载与保存 264 17.1 属性列表 264 17.1.1 NSDate 264 17.1.2 NSData 265 17.1.3 写入和读取属性列表 266 17.1.4 修改对象类型 267 17.2 编码

<<Objective-C基础教程>>

对象 268 17.3 小结 273 第18章 键/值编码 274 18.1 入门项目 274 18.2 KVC简介 276 18.3 键路径 277 18.4 整体操作 278 18.4.1 休息一下 279 18.4.2 快速运算 282 18.5 批处理 284 18.6 nil仍然可用 285 18.7 处理未定义的键 286 18.8 小结 287 第19章 使用静态分析器 288 19.1 静态工作 288 19.1.1 开始分析 288 19.1.2 协助分析器 292 19.1.3 了解更多 293 19.2 小结 295 第20章 NSPredicate 296 20.1 创建谓词 296 20.2 计算谓词 297 20.3 数组过滤器 298 20.4 格式说明符 299 20.5 运算符 300 20.5.1 比较和逻辑运算符 300 20.5.2 数组运算符 301 20.6 有SELF就足够了 302 20.7 字符串运算符 304 20.8 LIKE运算符 304 20.9 结语 305 附录 从其他语言转向Objective—C 306 索引 314

## &lt;&lt;Objective-C基础教程&gt;&gt;

## 章节摘录

版权页：插图：不仅可以将一个类的实现代码分散到多个不同的源文件中，还可以分散到多个不同的框架中。

NSString是Foundation框架中的一个类，包含了许多面向数据的类，如字符串、数字和集合。

所有的视觉组件（如窗口、颜色和绘图等）都包含在AppKit和UIKit中。

虽然NSString类是在Foundation框架中声明的，但是AppKit中有一个Nsstring的类别，称为NSStringDrawing，该类别允许你向字符串发送绘图消息。

在绘制一个字符串时，该方法会将字符串的文本渲染到屏幕上。

由于这是一种图形功能，所以它是AppKit框架中的方法。

不过Nsstring又是Foundation框架的对象。

cocoa的设计人员通过类别使数据功能放在Foundation框架中，而绘图功能放在AppKit框架中。

作为编程人员，我们直接使用NSString类即可，通常不必关心某个方法来自何处。

12.3通过类别创建前向引用 前面提到过，Cocoa没有任何真正的私有方法。

如果你知道对象支持的某个方法的名称，即使该对象所在的类的接口中没有声明该方法，你也可以调用它。

不过，编译器会尽力提供帮助。

如果编译器发现你调用对象的某个方法，但是却没有找到该方法的声明或定义，那么它将发出这样的错误提示：warning: 'CategoryThin9 - may not respond to—setThing4: '。

通常情况下，这种错误提示是有益的，因为它将有助于你捕捉许多输入错误。

不过，如果你的某些方法的实现使用了在类的@interface部分未列出的方法，编译器的这种警惕性会带来一些问题。

对于为什么不想在类的@interface部分列出自己的全部方法，有许多充分的理由。

这些方法可能是纯粹的实现细节，你可能将根据方法的名称来确定要使用哪个方法。

但是不管怎样，如果你在使用自己的方法之前没有声明它们，编译器就会提出警告。

而修复掉所有的编译器警告是一种良好的习惯，那么该如何去做呢？

如果能够先定义一个方法然后再使用它，编译器将会找到你的方法定义，因而不会产生警告。

但是，如果不方便这样做，或者你使用的是另一个类中尚未发布的方法，那么就需要采取其他措施。

通过类别来救急只要在类别中声明一个方法，编译器就会表示：“好了，该方法已经存在，如果遇到编程人员使用该方法，我不会提出警告。

”实际上，如果你不想实现这个方法的话，放着不管就可以了。

常用的策略是将类别置于实现文件的最前面。

假设car类有一个名为rotateTires的方法。

我们可以使用另一个名为moveTireFromPosition: toPosition:的方法来实现rotateTires方法，以交换两个位置的轮胎。

第二个方法是实现细节，而不是将其放在car类的公共接口中。

通过在类别中声明moveTireFromPosition: toPosition:方法，rotateTires方法可以使用它，但不会引发编译器产生警告。

## <<Objective-C基础教程>>

### 媒体关注与评论

“读过Scott Knaster书的人都知道，他的书从来不会让你失望，这本书也不例外。它介绍了Objective-C语言的基础知识和最新特性，生动有趣。

”——Bob Boonstra2. “我是Scott Knaster的老读者了，25年来他撰写了很多畅销Mac编程书，这本一如既往地棒。

”——John A. Vink

## <<Objective-C基础教程>>

### 编辑推荐

权威解读Objective-C和Cocoa特性.全面涵盖最新技术和新增工具.iPhone、iPad、Mac开发必备.Objective-C是一门面向对象、通用、高级、强大的编程语言。

它有着优雅的编程环境，并发扬了C语言的优秀特性，是苹果的iOS和OS X操作系统的主要编程语言。全面系统地讲述了Objective-C的基础知识和面向对象编程的重要概念，结合实例介绍了Cocoa工具包的优秀特性及框架，以及继承、复合、对象初始化、类别、协议、内存管理和源文件组织等重要编程技术，教你如何针对iOS或OS X用户界面编写出优秀的应用程序。

另外，本书第2版新增内容有：1．Objecitve-C最新特性：代码块、ARC、类扩展；2．新增工具Clang静态分析器GCD；3．如何使用UIKit框架开发精致的iOS应用程序；4．如何使用最新版本的Xcode。无论你是初次接触Objective-C和Cocoa，还是已有丰富的C语言、C++或者Java编程经验，本书都能让你轻松过渡并熟练掌握Objective-C！

<<Objective-C基础教程>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>