TURING 图灵程序设计丛书 移动开发系列

Apress®

- 解析iOS设计模式的开山之作
- 优化Objective-C编程实践的必修宝典
- 由此迈入移动开发高手行列

Pro Objective-C Design Patterns for iOS

Objective-C编程之道

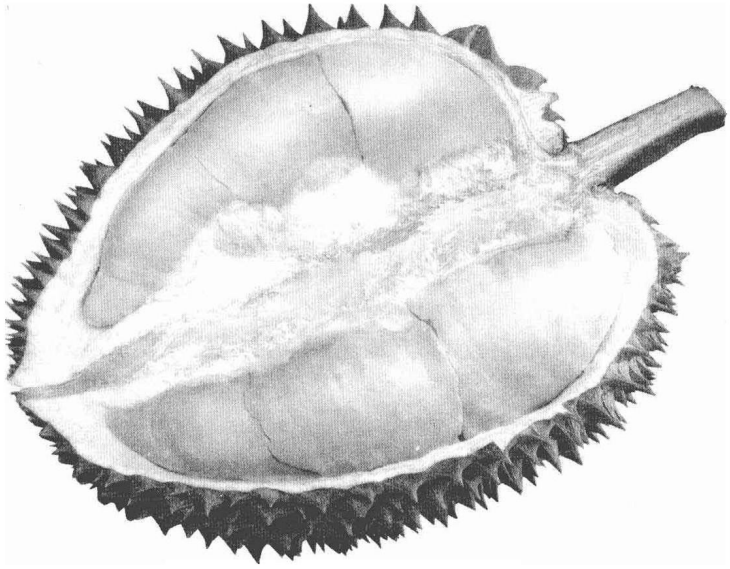
[美] Carlo Chung 著

刘 威 译

人民邮电出版社

POSTS & TELECOM PRESS

TURING 图灵程序设计丛书 移动开发系列



Pro Objective-C Design Patterns for iOS

Objective-C编程之道
iOS设计模式解析

[美] Carlo Chung 著
刘威 译

人民邮电出版社
北京

图书在版编目（CIP）数据

Objective-C编程之道 : iOS设计模式解析 / (美)钟冠贤著 ; 刘威译. -- 北京 : 人民邮电出版社, 2011.11
（图灵程序设计丛书）
书名原文: Pro Objective-C Design Patterns for iOS
ISBN 978-7-115-26586-9

Ⅰ. ①O… Ⅱ. ①钟… ②刘… Ⅲ. ①C语言－程序设计 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2011)第209817号

内 容 提 要

本书是基于iOS的软件开发指南。书中应用GoF的经典设计模式，介绍了如何在代码中应用创建型模式、结构型模式和行为模式，如何设计模式以巩固应用程序，并通过设计模式实例介绍MVC在Cocoa Touch框架中的工作方式。

本书适用于那些已经具备Objective-C基础、想利用设计模式来提高软件开发效率的中高级iOS开发人员。

图灵程序设计丛书

Objective-C编程之道：iOS设计模式解析

◆ 著 [美] Carlo Chung
译 刘 威
责任编辑 傅志红
执行编辑 罗词亮

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子邮件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
北京鑫正大印刷有限公司印刷

◆ 开本：800×1000 1/16
印张：19.25
字数：455千字 2011年11月第1版
印数：1－4 000册 2011年11月北京第1次印刷
著作权合同登记号 图字：01-2011-2955号
ISBN 978-7-115-26586-9

定价：59.00元

读者服务热线：(010)51095186转604 印装质量热线：(010)67129223
反盗版热线：(010)67171154

版权声明

致　　谢

“你知道吗，这本书真的很难写。”这是我和Apress出版社的一位策划编辑第一次通电话时他对我的提醒。出于他对我的信任，我在2010年夏天开始了这次写作历程。

感谢协调编辑Corbin Collins，他让整个团队保持专注。感谢项目编辑Douglas Pundick，感谢他在编辑审查阶段给我的所有提升本书的建议。感谢策划编辑Michelle Lowman的支持和耐心。感谢技术编审James Bucanek直到印刷之前最后一刻还提出了一些批评意见。感谢文字编辑Mary Ann Fugate和Mary Behr，是她们完美的文字编辑工作让这本书精彩纷呈。感谢所有了不起的Apress人——没有他们的帮助就不可能有这本书。

我要感谢Michael Fredrickson和Sreenivasa Busam，他们以其专业的视角对本书作了审阅。最后，非常感谢Mike Hambleton，感谢他审阅书稿时所体现的技术写作专业性以及细致和耐心的态度。

前　言

拥有超过20万个应用(且每秒都在增加)可供用户随意下载,苹果公司的应用商店(App Store)影响着各行各业。苹果公司广告语“There's an app for that”(总有一款应用可以做这个)的效应绝对不容忽视。不管你信不信,在这4个月里,我是坐在沙发上用iPad完成了本书的大部分内容。

每天都有更多的iOS开发者想跟随潮流,用下一个杀手级应用发家致富。截至本书写作时,全世界已有超过5万名iOS开发者,而且这一数字还在迅速增长。如果你真的有意从事iOS开发,并想通过好的软件设计原则让开发工作更加高效,那么你应该读这本书。

身为iOS开发者,我了解开发应用程序的痛苦与收获。学习新的编程语言绝非易事。最终,我们学会了并开始开发应用程序。在考虑Cocoa Touch框架时,即使是经验丰富的开发者也很容易被它优美的设计与结构所打动。这种优美来自设计者的深思熟虑,通过把各种为人熟知(或不为人知)的设计模式应用到框架的各种基础结构之中,他们为你我这样的第三方开发者提供了很好的可扩展性与灵活性。框架的大部分在一遍又一遍地重用相同的模式,后来添加到框架的新元素可以很容易地被其他应用程序开发者理解,不用再经过一次艰难的学习。

理解Cocoa Touch框架中使用的模式只是第一步。如果不花时间进行项目的设计,有了杀手级应用的想法马上就开始编码,那么随着后来为它添加更多的功能,它就会变成一个巨大的“樟脑球”。这样,最糟糕的情况是,它变得无法管理,而你(或团队中其他开发者)则根本无法理解你的代码。最终你将花更多的时间去修改代码缺陷,而不能专注于新的改进。

要充分利用Cocoa Touch框架,应该对设计模式有深刻理解,并在实现中进行恰当应用。本书受到Erich Gamma、Richard Helm、Ralph Johnson和John Vlissides(常被称作GoF)的经典著作*Design Patterns:Elements of Reusable Object-Oriented Software*[①]的启发。本书的每一章,模式的详细讲解中都引用了该书的定义。

今天,我们理解了在很多软件项目中使用设计模式的重要性。Cocoa Touch框架的大部分是用Objective-C写的,本书写作时,市面上还没有讲解如何用Objective-C语言实现设计模式的书。本书旨在成为该主题的第一本权威指南,引导大家了解该如何在iOS平台上以Objective-C语言实现Cocoa Touch开发所要用到的传统设计模式。

作为iOS开发者,你肯定想让开发更容易,是吧?如果不了解应用程序软件设计的最佳做法,开发过程将会困难重重,甚至最终毫无结果。而且,总的来说,想要跟上App Store或软件市场快

---

① 以下简称《设计模式》。其中文版《设计模式:可复用面向对象软件的基础》于2004年由机械工业出版社出版。——编者注

节奏的变化，重用那些开发好并经过验证的模块非常关键。

用一点儿耐心来理解本书的内容，尽量在实际项目中应用这些模式，读者很快就能体验到设计模式的好处。

我为本书建了一个网站，www.objective-c-design-patterns.com。上面有与本书相关的一些其他信息。也欢迎大家登录网站，分享在项目中使用设计模式的成功事例以及遇到的困难。本书的源代码可从www.apress.com[①]下载。

祝编码愉快！

Carlo Chung
钟冠贤
2011年3月

① 读者也可以在图灵社区（iturıng.com.cn）的本书页面下载。——编者注

目　录

第三部分 接口适配

第四部分 对象去耦

第五部分 抽象集合

第六部分 行为扩展

第七部分 算法封装

第八部分 性能与对象访问

第九部分 对象状态

Part 1 第一部分

设计模式初体验

本部分内容

第1章 你好，设计模式

几乎每一本讲计算机编程的书开篇都是以“你好，世界!”（Hello, World!）一章来介绍主题。由于这是一本关于设计模式的书，我们就从“你好，设计模式”开始吧。

既然你已经拿起了这本书，对于面向对象编程中设计模式的概念，你也许已经熟悉。设计模式是有用的抽象化工具，用于解决工程和建筑等其他领域的设计问题。出于同样的目的，软件开发领域借用了这一概念。设计模式是一个对象或类的设计模板，用于解决特定领域经常发生的问题。

本章介绍设计模式的简史，以及设计模式与Cocoa Touch技术之间的关系。不仅讨论了影响设计的若干问题，还介绍了本书用到的对象图示法，以及本书如何安排设计模式。

1.1 这是一本什么书

本书写给想通过使用设计模式以使软件开发更高效、更有趣的专业人员和有抱负的iOS开发人员。本书的目的在于，展示如何将设计模式在iOS的应用开发中付诸实践。我会集中讨论各种设计模式对于Cocoa Touch框架及其相关技术的适用性。

尽管本书介绍的有些原则和概念可能也适用于Cocoa Touch的老大哥——Mac OS X的Cocoa，但是不保证它们完全适用于完整版的Cocoa。另外，本书还可以用做Objective-C设计模式的快速参考手册。

你将会学到：

- 各种设计模式的基本概念；
- 在各种设计场景，如何将设计模式应用到代码中；
- 设计模式如何增强应用程序。

本书的网站是www.objective-c-design-patterns.com或www.objectivecdesignpatterns.com。欢迎分享你在项目中使用设计模式的成功事例以及遇到的困难。

书中的源代码可从www.apress.com下载。

1.2 开始前的准备

同其他讲iOS应用程序开发的书一样，你需要Xcode和iOS SDK来运行本书中的示例代码。本

书的示例项目的编译和测试使用的是Xcode 3.2.5和iOS SDK 4.2，应该也可以使用更高版本。有许多免费或付费的非Mac OS X平台的工具可以开发iOS应用程序，但是不保证本书中的示例程序在这些平台也能正常运行。

本书也涉及了iOS应用程序设计的一些内容。你可能需要下载或购买一些软件工具来帮助构建线框图和UI布局，用以练习或设计实际的应用程序。本书中的线框图是用OmniGraffle（www.omnigroup.com/products/omnigraffle/）制作的。该网站还提供了各种免费模板（stencil）和线框模板的下载。

本书中大量使用了类和对象图，但是可用的类和对象的建模工具软件却很少。本书写作的时候，多数建模软件基于标准图示法，而这些标准图示法只适用于C++、Java和C#等其他面向对象编程语言。Objective-C有一些特殊功能，比如范畴（category）和扩展（匿名范畴），难以用标准图示法来表达。因此，为了表达这些特殊的功能，我新造了若干图示法。在本章稍后的1.7节中将予以介绍。可以用你的绘图软件根据这些新造的图示法来绘制Objective-C的类和对象。希望适用于Objective-C的标准化对象建模图示法尽早面世。

1.3 预备知识

本书属于专业丛书，所以并不是“iOS开发基础”或“Objective-C 24小时入门”。读者应具有iOS SDK（Cocoa Touch框架）的基础知识。此外，读者也应对Objective-C编程语言有足够的了解。否则，将无法理解本书中的很多观点和高级技巧。

尽管本书主要面向中高级开发人员，但读者并不需要对设计模式有深入的了解就能理解设计模式的概念。即便你已经在工作中接触过一些设计模式，你仍可从本书获益。介绍模式的所有章节都借助打比方，让读者对设计模式能有透彻的理解。而且，不易理解之处会有提示和说明。

准备好了吗？我们开始吧！

1.4 似曾相识的设计

身为开发人员，你可能有过这样的感受：“我以前解决过这个问题，但不记得具体是在哪里、怎样解决的。”经常会有这样的事儿，要是你重复做着特定类型的项目更是如此。比如，数据库的应用程序都有存储和检索数据的数据库访问功能。你要是记录下问题的细节和解决方法，就可以复用这些方法，而不是次次从零开始。

有关设计中最常见的似曾相识的情况以及解决方法，最早作出权威性论述和分类的是《设计模式》（Erich Gamma、Richard Helm、Ralph Johnson和John Vlissides著，Addison- Wesley Professional出版社，1994年）。该书将设计面向对象软件的经验总结为可以有效使用的设计模式。有趣的是，Cocoa（Touch）框架的前身NEXTSTEP的设计中采用了很多漂亮、可复用的面向对象软件设计模式，而这对激发GoF（该书作者为人熟知的绰号）的灵感起到了重要作用。

根据《设计模式》一书，设计模式是对定制来解决特定场景下一般设计问题的类和相互通信的对象的描述。

简而言之，设计模式是为特定场景下的问题而定制的解决方案。特定场景指问题所在的重复出现的场景。问题指特定环境下你想达成的目标。同样的问题在不同的环境下会有不同的限制和挑战。定制的解决方案是指在特定环境下克服了问题的限制条件而达成目标的一种设计。

设计模式是经时间证明为有效的，对特定面向对象设计问题主要方面的一种抽象，体现了面向对象设计的重要思想。有些设计原则影响着设计模式。这些原则是构建可复用、可维护的面向对象应用程序的经验法则，比如，“优先使用对象组合而不是类继承”和“针对接口编程而不是针对实现编程”。

例如，通过给程序的变动部分定义接口而对其封装和隔离，这些部分的变动就独立于程序的其他部分，因为它们不依赖于任何细节。以后就可以变更或扩展这些可变的部分而不影响程序的其他部分。程序将因此能够更灵活而可靠地进行变更，因为我们消除了部分与部分之间的依赖关系并减少了耦合。这些益处使得设计模式对于可复用软件的编写非常重要。

程序（包括其中的对象和类），如果在设计中使用了设计模式，将来就更易于复用与扩展，更易于变更。而且，基于设计模式的程序会更加简洁而高效，因为达成同样目的所需的代码行会更少。

1.5　设计模式的起源——模型、视图和控制器

模型-视图-控制器（MVC）设计模式及其变体至少在Smalltalk诞生初期就已经出现了。这个设计模式是Cocoa Touch中很多机制和技术的基础。

在MVC设计模式中，对象在应用程序中被分为三组，分别扮演模型、视图和控制器。MVC模式也定义了对象之间跨越其角色的抽象边界的通信方式。MVC对Cocoa Touch应用程序设计起了重要作用。应用程序设计的一个主要步骤是决定对象或类应该属于这三组中的哪一组。如果应用程序的MVC划分得清晰，使用Cocoa Touch框架中的任何技术都会相对容易。

这个模式本身不是独立的模式，而是由几个其他基本模式组成的复合模式。这些基本模式将在本书介绍模式的章节中详细介绍。

相比于非MVC的应用程序，MVC的应用程序中的对象更加易于扩展和复用，因为其接口通常会定义得更好。而且，许多Cocoa Touch技术和架构是建立在MVC之上的，要求应用程序和对象服从于给MVC的角色设定的规则。

后面几节将阐述MVC中的各个角色在架构中如何发挥其作用。

1.5.1　在模型对象中封装数据和基本行为

模型对象维护应用程序的数据，并定义操作数据的特定逻辑。模型对象可以复用，因为它表示的知识适用于特定的问题领域。例如，模型对象可以表示复杂的数据结构，对应于用户在屏幕上所画的图形，或者仅仅表示待办事项应用程序中的一条待办事项。

只要加载的是包含有应用程序永久信息的数据，就应将其放入模型对象。理想状况下，模型对象同用于对其进行显示和编辑的用户界面之间不应有任何直接的关联。

1.5.2 使用视图对象向用户展示信息

视图对象可以响应用户操作，并懂得如何将自己展现在屏幕上。视图对象通常从应用程序的模型对象获取数据用以展示。它可以跟一个模型对象的部分、整体或者多个模型对象合作。通常，用户可以通过它修改数据。

虽然视图对象和模型对象之间关系密切，但是在MVC应用程序中它们之间没有耦合。除非因性能原因（比如视图需要对数据进行缓存），不应将视图用于存储它所展示的数据。

因为视图对象可以与许多不同的模型对象合作，所以它们往往可在不同应用程序之间复用并保持一致。UIKit框架提供了各种类型的视图类，可复用于我们的应用程序。

1.5.3 用控制器对象联系起模型和视图

控制器对象就像视图对象和模型对象的中间人。作为中间人或协调人，它建立起沟通渠道，使视图得以知晓模型的变更而给予响应。

除了协调作用之外，控制器对象还可以为应用程序执行其他操作，比如为应用程序管理其他对象的生命周期，进行设置和协调任务。

举例来说，用户通过操作视图对象（比如在文本框中输入）得到的值，可以传给控制器对象。控制器对象也可以让视图对象根据此用户操作改变其外观或行为，比如禁用某个文本输入框。

依照所需的设计，控制器对象可设计为可复用的或不可复用的（具体的[①]）。

1.5.4 作为复合设计模式的 MVC

MVC本身并不是最基本的设计模式，它包含了若干更加基本的设计模式，这些模式将在本书的模式章节中阐述。在MVC中，基本设计模式相互配合，确定了各功能之间的协作，这是MVC应用程序的特性。

Cocoa(Touch)的MVC用到的模式有：组合(Composite)、命令(Command)、中介者(Mediator)、策略（Strategy）和观察者（Observer）。

- 组合（第13章）——视图对象之间以协作的方式构成一个视图层次体系，其中既可以有复合视图（比如表格视图），也可以有独立视图（比如文本框或按钮）。每个层次的每个视图节点都可以响应用户的操作并把自己绘制到屏幕上。
- 命令（第20章）——这是一种“目标–动作”机制，视图对象可以推迟其他对象（比如控制器）的执行，让其他对象等到发生了某些事件后再执行。这一机制构成了命令模式。
- 中介者（第11章）——控制器对象起着中间人的作用，而这个中间人则采用了中介者模式，它构成了在模型和视图对象之间传递数据的双向通道。应用程序的控制器对象将模型的变更传达给视图对象。

① 原文为concrete。concrete class指既不继承其他类也不被其他类继承，没有虚函数的类。——译者注

- 策略（第19章）——控制器可以是视图对象的一个“策略”。视图对象将自身隔离，以期维持其作为数据展示器的唯一职责，而将一切应用程序特有的界面行为的决定委派给它的“策略”对象（即控制器）。
- 观察者（第12章）——模型对象向它所关注的控制器等对象发出内部状态变化的通知。

图1-1展示了虚构场景下这些模式是如何协同工作的。

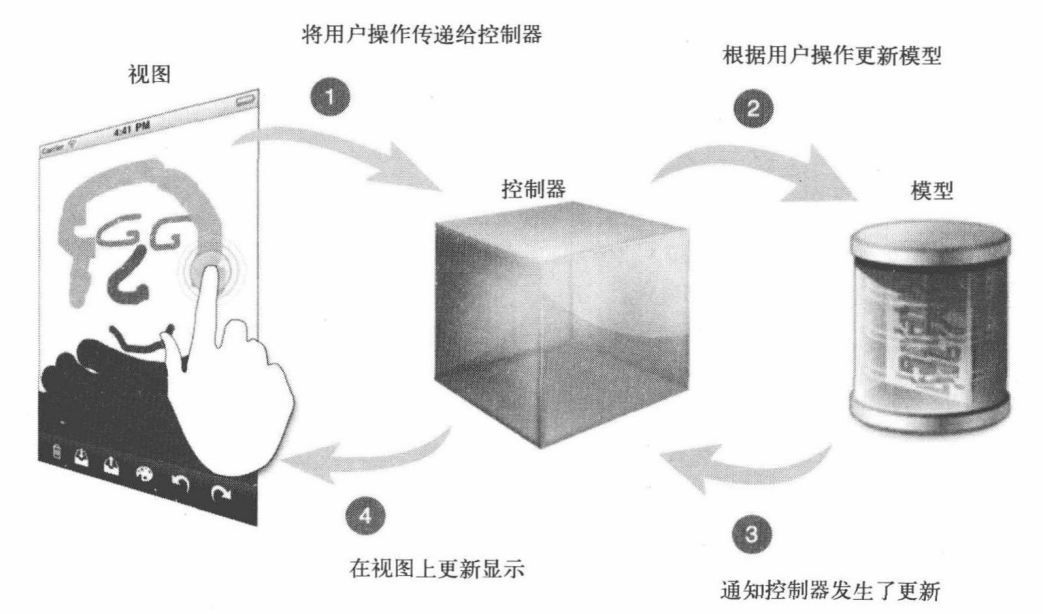


图1-1　展示模型、视图和控制器作为一组不同的实体如何交互的示意图

在图1-1中，

(1) 用户在画布视图上用手指触摸或拖动，产生一个触摸事件。被触摸的实际视图（图层）就在视图组合中的某个层次上。画布（视图）将触摸消息传达给视图控制器。

(2) 控制器对象接收到触摸事件及其相关信息，然后应用策略来变更模型的状态，必要时请求视图对象根据此事件更新其行为或外观。

(3) 每当变更发生并已反映到模型对象，模型对象就会通知所有已注册的观察者对象，如控制器。

(4) 控制器就像一个协调人，它将变更了的数据从模型传递给视图，以便视图可以相应地更新其外观。

1.6　影响设计的几个问题

设计模式肯定会从许多方面影响系统设计。但是有一些设计原则也会影响设计。有些原则针对一般的软件设计，而有些原则是针对Objective-C和Cocoa Touch的。我将在以下各节进行讨论。