

<<Cocoa设计模式>>

图书基本信息

书名：<<Cocoa设计模式>>

13位ISBN编号：9787111317401

10位ISBN编号：7111317408

出版时间：2010年11月

出版时间：机械工业出版社

作者：Erik M. Buck,Donald A. Yacktman

页数：336

译者：陈宗斌,孔祥波

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>

<<Cocoa设计模式>>

前言

Apple的Cocoa软件所包含的多数技术从1988年开始就已经投入商业应用，虽然Cocoa可能不够完善，但仍然具有革命性意义。

它已在市场上得到广泛应用，如NEXTSTEP、OPENSTEP、Rhapsody和Yellow Box等都应用了该技术。Cocoa由一系列可重用的软件框架组成，包含用于构建Mac OS X桌面和手机应用程序的对象和相关资源。

近年来，Apple对Cocoa进行了显著的扩展并添加了新的软件开发工具，从而在Cocoa已经提供的高水平性能的基础上进一步提高了程序员的效率。

很多程序员在第一次使用此框架时，面对宽泛和复杂的Cocoa通常会不知所措。

Cocoa包含大量的功能部件，但在一致性方面表现却十分出色，这要归功于在Cocoa的设计中应用了模式。

了解这些模式才能更有效地使用框架，并在编写自己的应用程序时作为指导。

本书介绍了Cocoa框架中用到的面向对象的设计模式。

设计模式不是Cocoa的专利，它在很多可重用的软件库和软件开发环境中均有所应用。

设计模式能识别那些重复出现的软件问题和解决它们的最佳做法。

本书的主要目的是提供有关设计的专业知识和Cocoa的基本原理，掌握了这些知识，你就可以在自己的软件中高效地重用那些可靠的正确模式，即使你用的并非Cocoa。

<<Cocoa设计模式>>

内容概要

对于cocoa框架的庞大和复杂，mac和iphone开发者常常不知所措。

虽然cocoa看起来很庞大，但是一旦理解了它所使用的面向对象的模式，你就会发现它极其出色，也非常简单可靠。

本书首先介绍了所有模式的起源，即“模型—视图—控制”模式，它是所有mac和iphone开发的核心，因此从一开始就牢固掌握它非常重要。

本书列出了将在cocoa中用到的主要设计模式，包括：基础模式，如枚举器、访问器和两阶段创建；增强模式，如单例模式、委托模式和响应链；隐藏复杂性的模式，如程序包、类集群、代理和转发，以及控制器。

这些还不是全部！

本书精心列出了28种设计模式，并包含一些依然有效的示例代码。

本书还讲解了core data模型、appkit视图，以及绑定和控制器。

本书利用objective-c和cocoa框架的基础，清晰地定义了每种模式能解决的问题，所有mac和iphone开发者都应该学习。

<<Cocoa设计模式>>

作者简介

Erik M. Buck于1993年成立了EMB & Associates公司，并且通过利用后来成为Apple的Cocoa框架的NeXT/Apple软件技术，把公司建设成太空和娱乐软件业中的领导者。

Buck先生的工作还涉及建筑业、给8中学生讲授科学、展览油画肖像，以及开发可替代燃料汽车。Buck先生在2002年出售了

<<Cocoa设计模式>>

书籍目录

译者序 序 前言 作者简介 第一部分 一种可控制一切的模式 第1章 模型-视图-控制器 1.1 cocoa中的mvc 1.2 小结 第2章 分析和应用mvc 2.1 非mvc设计 2.2 mvc设计 2.3 小结 第二部分 基础模式 第3章 两阶段创建 3.1 动机 3.2 解决方案 3.3 cocoa中的示例 3.4 后果 第4章 模板方法 4.1 动机 4.2 解决方案 4.3 cocoa中的示例 4.4 后果 第5章 动态创建 5.1 动机 5.2 解决方案 5.3 cocoa中的示例 5.4 后果 第6章 类别 6.1 动机 6.2 解决方案 6.3 cocoa中的示例 6.4 后果 第7章 匿名类型和异类容器 7.1 动机 7.2 解决方案 7.3 cocoa中的示例 7.4 后果 第8章 枚举器 8.1 动机 8.2 解决方案 8.3 cocoa中的示例 8.4 后果 第9章 执行选择器和延迟执行 9.1 动机 9.2 解决方案 9.3 cocoa中的示例 9.4 后果 第10章 访问器 10.1 动机 10.2 解决方案 10.3 cocoa中的示例 10.4 后果 第11章 归档和解档 11.1 动机 11.2 解决方案 11.3 cocoa中的示例 11.4 后果 第12章 复制 第三部分 主要通过解耦来变得更强大的模式 第13章 单例 第14章 通知 第15章 委托 第16章 层次结构 第17章 插座变量、目标和动作 第18章 响应者链 第19章 联合存储 第20章 调用 第21章 原型 第22章 享元 第23章 装饰器 第四部分 主要用于隐藏复杂性的模式 第24章 包 第25章 类簇 第26章 外观 第27章 代理和转发 第28章 管理者 第29章 控制器 第五部分 模式应用的实用工具 第30章 核心数据模型 第31章 应用程序工具箱视图 第32章 绑定和控制器 附录 资源

<<Cocoa设计模式>>

章节摘录

插图：第一部分一种可控制一切的模式第1章模型-视图-控制器模型-视图-控制器(MVC)是世界上最古老、最成功的可复用软件设计模式之一。

它最初出现在20世纪70年代的Smalltalk编程语言中。

MVC定义了Cocoa框架的总体结构。

它是一种高级别的模式，能将多个协作对象的大型群组划分为独立的子系统：模型、视图和控制器。

分析通用应用程序的功能和行为，有助于理解子系统在MVC模式中担当的角色。

多数应用程序存储信息、检索信息，并将信息呈现给用户，使用户能够编辑或操纵信息。

在面向对象的应用程序中，信息不只是字节，对象会将信息与使用该信息的方法封装在一起。

应用程序中的每个对象都应该符合且仅符合以下子系统之一：
· 模型。

模型子系统由为应用程序提供独特功能和信息存储的对象组成。

模型包含处理应用程序数据的所有规则。

模型是使应用程序体现其价值的关键子系统。

保证模型子系统的独立而不依赖于视图或者控制器子系统非常关键。

· 视图。

视图子系统用于展示从模型中收集的信息，并为用户提供与此信息交互的方式。

理解视图的关键是要知道总是有大量视图存在。

例如，可能有图形用户界面视图、打印报表视图、命令行视图、基于Web的视图以及脚本语言视图，它们都与同一个模型交互。

· 控制器。

控制器的目的是解除模型和视图之间的耦合。

用户与视图交互的结果是向控制器子系统发出请求，还可能会进一步请求改变模型中的信息。

控制器还要处理数据的转换和格式，以呈现给用户。

例如，模型中可能以米存储数据，但根据用户的偏好，控制器可能要将数据转换为英尺。

模型可能将对象存储在无序集合中，但控制器可能会先为这些对象排序，然后再转到视图中展示给用户。

<<Cocoa设计模式>>

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>