PROGRAMMER TO PROGRAMMER™

Professional PHP Design Patterns

PHP设计模式

(美) Aaron Saray 著
梁志敏 蔡建 译

wrox™
清华大学出版社

PHP 设计模式

(美) Aaron Saray 著

梁志敏 蔡建 译

清华大学出版社

北 京

Aaron Saray
Professional PHP Design Patterns
EISBN：978-0-470-49670-1

北京市版权局著作权合同登记号　图字：01-2009-7477

图书在版编目(CIP)数据

PHP 设计模式 /(美) 萨莱 (Saray,A.) 著；梁志敏，蔡建 译.—北京：清华大学出版社，2010.6
书名原文：Professional PHP Design Patterns
ISBN 978-7-302-22843-1
Ⅰ. P…　Ⅱ. ①萨…　②梁…　③蔡…　Ⅲ. PHP 语言—程序设计　Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2010)第 095093 号

责任编辑：王　军　于　平
装帧设计：孔祥丰
责任校对：胡雁翎
责任印制：李红英
出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**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：三河市新茂装订有限公司
经　　销：全国新华书店
开　　本：185×230　**印　张**：17　**字　数**：350 千字
版　　次：2010 年 6 月第 1 版　　**印　　次**：2010 年 6 月第1 次印刷
印　　数：1～4000
定　　价：36.00 元

产品编号：034809-01

作 者 简 介

在 Aaron Saray 8 岁的时候，他接触到一台没有永久存储器的二手 Commodore 64 家用计算机，这使他开始着迷于计算机科学，并且了解了许多不同的语言和计算机。在 2001 年，Aaron 最终决定选择使用 PHP 语言。从那时开始，他坚持学习其他多种 Web 语言(如 HTML、CSS 和 JavaScript)，同时不断充实自己的 PHP 专业知识。在其从业过程中，Aaron 曾经为 Internet Service Provider(ISP)设计和维护过各种 Web 站点工具，为一家大型牙科保险公司的客户创建过基于 Web 的账户管理工具，还为基于 Internet 连接的 Point of Sales 系统开发过后台管理 Web 站点。在成为 Zend 认证工程师后，Aaron 开始应用 Web 开放源代码软件来创办运营自己的 Web 开发公司。在 http://www.aaronsaray.com 站点上，他一直都在发布开放源代码软件，并且不断更新以 PHP 内容为主的博客。

前　言

PHP 是目前的主流编程技术。我们可以看到大量 PHP 网站以及大量有关 PHP 的工作机会，还可以看到许多大型公司都使用这种开放源代码语言来支持其业务。这种最初毫不起眼的开放源代码语言如今已广泛植根于整个业界之内。PHP 如今已得到了人们的广泛重视，诸如 IBM 和 Microsoft 这样的公司都已支持这种企业级语言。PHP 友好地融合了许多新的观念和思想，其中最值得关注的是通过更健壮的、更灵活的和更经济的部署来开发 PHP 应用程序。与此同时，许多资深的编程人员也在 PHP 中引入了若干重要的概念，本书侧重的就是其中一个主要的概念：设计模式。

0.1　本书的适用范围

在确定本书所适用的读者对象时，作者非常难以取舍。是为刚接触 PHP 及其功能和概念的初级编码员而编写，还是为具有多年工作经验的编程人员而编写？是应当为了解 PHP 面向对象功能的专业读者而编写吗？本书是否需要继续包含对 PHP4 的支持？最后一个问题比较容易回答："当然，不再支持 PHP 4。"然而，考虑到 PHP 4 仍然被广泛部署，开发人员仍然在使用 PHP 4 创建新的功能，因此并不能轻易地给出这个答案。为了帮助更广泛的读者使用 PHP 实现设计模式，而不是仅仅作为 PHP 语言参考材料，本书采用了下列指导原则来确定适用的读者。

本书读者对象：

- 必须完全了解 PHP 语言，或者至少收藏过 http://php.net 网址。因为某些示例可能使用编程新手之前从未遇到过的函数。
- 必须大致掌握 PHP 中所使用的面向对象编程(Object Oriented Programming，OOP)技术。中级 OOP 编程人员会发现第 2 章中对 PHP 高级 OO 功能的探讨是非常有价值的。
- 必须使用 PHP 5 或更高版本，从而具有可用的面向对象编程功能的完整集合，并且能够执行示例和案例分析代码。
- 应当熟悉统一建模语言(Unified Modeling Language，UML)。

简单来说，对于在构建交互式应用程序方面具有一定经验(至少曾经建立过一个博客)的编程人员来说，本书中使用的示例和概念是极为有价值的。如果以前只使用 PHP 完成过简单的工作(如构建主题模板或联系表单)，那么读者会发现阅读与模式相关的章节是较为困难的。

0.2 本书的结构

本书分为 3 大部分：引言章节、参考章节以及案例分析章节。这几个部分具有不同的侧重点。

0.2.1 引言章节

第 1 章不仅对设计模式进行了简要的介绍，而且说明了在 PHP 中使用这些设计模式的要求。全世界才华横溢的 PHP 编程人员总是渴望学习新的知识。本章的目的在于：将 PHP 编程人员的视野范围从只基于 PHP 的概念扩展至体系结构更健全的设计模式领域。

第 2 章侧重于介绍一些工具，这些工具在 PHP 中能够用于构建各种设计模式概念的基础。通过回顾 PHP 的中级和高级 OOP 功能、标准 PHP 库以及现有的开放源代码 PHP 架构，本章将帮助读者更深入地理解 PHP 与设计模式。

0.2.2 参考章节

参考章节是本书的中间章节，也是设计模式最基本的部分。这些章节可以分为 4 个主要部分：名称、问题与解决方案、UML 图以及一个简单的面向对象的代码示例。上述内容基本上覆盖了设计模式的主要功能部分，同时不至于过分冗长。

0.2.3 案例分析

本书的最后一个部分是一个深入的案例分析，包括项目和计划的详细说明、对可用模式的分析以及逐步应用这些模式的方式。

1. 功能分析

通常，当您获得一系列规范说明时，它们并不是最终的版本。在最初查看这些说明时，您应当对具体的体系结构有大致的了解。此时，您会希望了解项目的需求，以便确定它是一个仅仅应用一次的实例，还是一个可扩展的项目。需要在将来实现哪些功能？假如您不是这方面的专家，那么可能需要根据从业务分析中了解到的具体问题来寻求答案。

在研究案例时，您会收到客户提供的规范说明。本书将全面讲述查看规范说明、提出问题和理清思路的整个过程。最后，我们将提供一个已更新的规范说明文档。

2. 模式分析

在开发任何项目时，必须首先进入分析阶段。我遇到过太多这样的例子：在项目开始阶段，许多编程人员要么漫不经心，要么得意忘形，从而导致最终完成的项目往往不尽如人意。此时，应该重新认真查看规范说明以确定自己的工作计划。

3. 逐步生成代码

案例分析中的这个环节有些偏离本书设定的目标。该部分包含基于 UML 图的详细代码示例。本书将从模式层次分步骤阐明构建应用程序各部分的整个思考过程。然而，我们的重点并非针对语言具体功能的分析。因此，中级编程人员有时可能需要参考 PHP 手册。

生成全部代码后，您应当回顾自己的应用程序以及编码过程中的所有选择，从而确保没有比当前模式更适合的模式。设计模式并不意味着必须坚持严格的规则，但是必须针对具体的应用程序建立构造块和框架。为了创建体系结构更健全的代码库，我们完全可以变换项目中的设计模式。

0.3 使用本书的要求

因为本书的一个优点是更注重概念(而非应用)，所以使用本书的要求非常简单，如下所示：

- Windows 或 Linux 操作系统
- PHP 5.2 或更高版本
- MySQL 5.0 或更高版本

如果不满足上述要求，大部分样本代码仍然能够正常运行。然而，运行本书的最后一个案例分析时，必须满足这些要求。

0.4 源代码

学习本书中的示例时，读者既可以手动输入所有的代码，也可以使用本书附带的源代码文件。本书使用的所有源代码都可以从 http://www.wrox.com 和 http://www.tupwk.com.cn 上下载。登录到该站点，使用 Search 工具或使用书名列表就可以找到本书。随后，单击本书细目页面上的 Download Code 链接就可以获得所有源代码。

因为许多图书的书名都很相似，所以通过 ISBN 找到本书是最简单的查找方式。本书的 ISBN 是 978-0-470-49670-1。

下载代码之后，只需用自己喜欢的解压缩软件对它进行解压缩即可。此外，读者也可以进入 http://www.wrox.com/dynamic/books/download.aspx 上的 Wrox 代码下载主页，查看本书和其他 Wrox 图书的所有可用代码。

0.5 勘误表

请给 wkservice@vip.163.com 发电子邮件，我们就会检查您的反馈信息，如果是正确的，我们将在本书的后续版本中采用。

尽管我们竭尽全力来保证文章或代码中不出现错误，但错误总是难免的。如果在本书中发现了错误(例如拼写错误或代码错误)，请及时告知我们，我们将不胜感激。勘误表可以让其他读者避免被误导，当然，还有助于提供更高质量的信息。

要在网站上找到本书的勘误表，读者可以登录 http://www.wrox.com 站点，通过 Search 工具或书名列表查找到本书，然后在本书的细目页面上单击 Book Errata 链接。在这个页面上，读者可以查看到已提交的和 Wrox 编辑已发布的所有勘误项。完整的图书列表还应当包括每本书的勘误表，网址是 http://www.wrox.com/misc-pages/booklist.shtml。

如果在 Book Errata 页面没有找到您所发现的错误，那么请进入 http://www.wrox.com/contact/techsupport.shtml 页面并填写完相应的表格，从而向我们发送您的意见。我们会阅读您发送的信息，如果是正确的，那么我们将在本书的后续版本中加以采用。

p2p.wrox.com

为了便于作者和读者之间的讨论，读者可以在 http://p2p.wrox.com 站点上加入 P2P 论坛。该论坛是一个基于 Web 的系统，用于发布与 Wrox 图书相关的信息和相关技术，并且支持与其他读者和技术用户的交流。这个论坛提供了订阅功能，一旦论坛上出现新贴，就会向您发送您所选择的主题。Wrox 作者、编辑、其他业界专家以及读者都会在这个论坛上进行讨论。

在 http://p2p.wrox.com 站点上有许多不同的论坛，这不仅能够帮助读者阅读本书，而且读者在开发自己的应用程序时也可以从这个论坛中获益。如果想加入这个论坛，那么需要执行下面的步骤：

(1) 进入 http://p2p.wrox.com 站点，单击 Register 链接。

(2) 阅读使用条款，单击 Agree 按钮。

(3) 提供加入论坛所需的必要信息与愿意提供的可选信息，单击 Submit 按钮。

(4) 随后用户会接收到一封电子邮件，其中的信息描述了如何验证账户以及完成加入过程。

注意：

不加入 P2P 也可以阅读论坛上的信息，但只有加入论坛后，用户才能够发布自己的信息。

加入论坛后，用户就可以发布新的信息，回复其他用户的贴子。用户可以随时在 Web 上阅读信息。如果希望某个论坛给自己发送新信息，那么可以在论坛列表中单击该论坛对应的 Subscribe to this Forum 图标。

要想了解如何使用 Wrox P2P 的更多信息，请确认阅读 P2P FAQ，从而了解论坛软件的工作原理以及许多针对 P2P 和 Wrox 图书的常见问题解答。要阅读 FAQ，可以单击任意 P2P 页面上的 FAQ 链接。

目　录

第Ⅰ部分　初识设计模式与 PHP

第 1 章

理解设计模式

通常，如果拿到一本书并阅读自己不熟悉的主题 5 页以上，就会使读者觉得不安。超过 5 页就可能让读者放弃阅读、情绪烦躁，甚至高声抱怨太困难了！虽然本章的篇幅不止 5 页，但是请读者一定不要放弃阅读。在某种程度上，术语“设计模式(Design Pattern)”只是一个新奇的名称，它并非特别复杂。本章的亮点在于采用了读者可能已知的和经常使用的知识，并且将其提炼为更简明的定义。让我们开始了解到底什么是设计模式。

1.1 什么是设计模式

下面给出的 Steve 的故事有助于描述现实生活环境中的设计模式，我希望读者从未听过这个故事。

1.1.1 一个普通的示例

Steve 在一家大型保险公司工作，他最近的工作是开发一个通过 Web 界面将客户信息显示给电话客服中心代表的软件。Steve 设计了一个复杂的系统，这个系统允许电话客服中心代表搜索某个客户、输入电话记录、更新客户保险金额信息以及处理付款。系统的安装非常顺利，几乎没有出现在产品环境中安装新软件时经常遇到的麻烦。Steve 非常高兴，倍感轻松，接下来似乎应当坐下来惬意地品尝一下咖啡了。

很快，保险公司最新的投资规模达到了以前的 3 倍。不仅 Steve 被叫回为电话客服中心软件提供新的可扩展性和增强处理，而且公司急切希望在其站点上为新招揽的客户提供某些新的功能。Steve 所在的部门为此也增加了两名新开发人员：Andy 和 Jason。

公司副总裁提出的要求是：在客户完成成功和安全的登录后，公司的站点应当允许客户自己进行相关支付操作。此外，系统应当显示客户呼入电话客服中心的次数。最后，系统应当显示客户账户对电话客服中心软件进行的所有改动的审计日志。

Steve 知道：简单地更新电话客服中心软件就能够提供审计日志，随后复制并改进代码就可以使用支付处理功能。但是，新加入的编程人员需要尽快投入到新系统的开发工作中。Steve 的老板希望两人接手 Steve 最熟悉的项目部分。考虑到 Steve 是经验最丰富的顶尖 PHP 编程人员，老板需要 Steve 尽快完成公司站点的其他部分，然后再投入到新加入编程人员对电话客服中心软件的审计功能的修改工作中。最后，Steve 还要负责为用户登录界面的新支付处理部分提供挂钩程序。

Steve 先前编写的代码不算太糟糕，但是 Jason 似乎要花费很长的时间才能理解这些代码并完成将支付处理部分移植入公司站点的工作。因此，Jason 决定采用自己的方法来编写代码，以便更快地完成任务。Jason 向 Steve 报告了自己的想法，随即按照这个思路着手开展工作。Andy 同样也忙忙碌碌，因为他刚获得计算机科学专业的硕士学位，所以没有太多经验应对时而混乱的代码。

经过加班加点的工作，这个团队终于成功地完成了对新代码的更改工作。Andy 认为整个开发还可以采用更好的体系结构。Steve 觉得：其他编程人员如果只是复制和粘贴他原先设计的代码，那么开发进度会更快；Jason 和 Andy 只需要进行一些改进，问题就会迎刃而解。Jason 则迫切想解开自己的困惑：为什么某些功能在代码中的不同部分所实现的方式不同。

随着公司 Web 站点的访问者越来越多，系统的性能开始变得越来越差。因此，Steve 的老板建议开发团队花时间对代码进行优化。

Jason 发现他为支付处理操作编写的方法与 Steve 编写的方法几乎完全相同。因此，Jason 将这些方法组合入一个类中。Steve 则开始查看他为电话客服中心站点编写的身份验证代码与他为公司站点的用户登录编写的类之间的共性。Andy 意识到他们创建的所有 PHP 页面的顶部都具有一组相同的函数调用。因此，他创建了一个引导类型类，以便将所有这些函数调用放在同一个位置，进而减少了代码的重复。

作为局外人，我们可以客观地看到很多问题。Steve 的代码可以按照自己的方式从通用性中受益。Andy 在软件设计方面所受的正规教育使他有时会对 PHP 完成任务的能力以及体系结构提出质疑。Jason 无法轻易地理解 Steve 设计的支付系统，因此他选择自己从头开始创建，从而导致代码的重复。最后，经过对软件的分析，这个开发团队开始在看上去混乱的代码库中发现了各种模式。自此，开发团队开始转向处理设计模式。

1.1.2　设计模式能够解决相同的问题

在上面的示例中，Steve 团队的错误实际上与设计模式概念的第一个重要部分有关。在软件开发中，该团队并未明确地创建模式。模式往往是通过实际环境中的实践和应用发现的。将支付应用系统和引导类型调用合并入特定类就属于在编程过程中标识模式的示例。

有人曾经说过，所有的音乐片段都能找到出处。如今，创作新的音乐只是将特定的音

符按照不同的节奏和速度进行重新排列。除了某些实现重大突破的特例之外，通常的软件开发也具有类似的情况。相同的问题会不断地重复出现，并且需要通用的解决方法。因此，对于这些相同的问题来说，设计模式是一种可重用的解决方案。

任何提及设计模式的书籍都离不开著名的“四人组(Gang of Four)”：Erich Gamma、Richard Helm、Ralph Johnson 和 John Vlissides，这 4 个人是最早介绍设计模式的书籍的作者。在自己的领域从事长期工作之后，他们开始注意到不同开发项目中出现的设计的特定模式。这 4 个人将这些想法汇集到一起，从而形成了最早的设计模式概念。通过将特定模式标识为用于未来开发的模板，他们能够将设计模式表达为易于理解的参照基准，以便人们能更好地理解大型和复杂的编程概念。

设计模式涵盖了从接口设计到体系结构的许多方面，甚至包括市场营销和度量。本书主要关注使用面向对象编程方法构造开发语言。

软件设计中的问题由下列 3 部分组成：

- what：要考虑的相关业务和功能需求。
- how：如何使用特定设计满足上述需求。
- work：实际的实现，或者说如何投入应用和实践。

设计模式正好对应该过程的 how 部分，因此本书阐述了如何解决这些问题以及成功解决问题所需的部分实现工作。可以将 PHP 描绘为解决问题所采用的载体。一旦知道了软件需要完成的功能，就能够设计如何完成这些功能，从而通过更少的重构更容易地实现所有功能。

我们已经提到，本书的重点并不是对 PHP 语言的掌握以及对其复杂性的理解，而是在于描述通用的、经过时间检验的一系列方法及其与 PHP 的联系。

从前面的示例中可以看出，模式当然是从软件开发阶段开始的。不过，如果拥有涉及现有模式的完整资料，那么就能够更快地计划体系结构和做出更好的选择。此外，来自不同软件领域的编程人员能够识别该模式，并且只需将其改写为特定的语言类型。应用程序应采用一组清晰的模式，这有助于团队新成员理解项目，从而避免浪费太多时间。

1.1.3　设计模式无所不在

我们已经介绍过，Steve 的团队能够理解其软件中的基本模式以及创建可重用项。现在我们也能够进行自己的软件开发。有多少次使用自己的用户类创建相同的用户登录和身份验证系统？是否在希望的位置使用了 db( )函数？这些都是如何使用模式的示例。

在偏爱的 PEAR 或其他架构库中，能够发现更详细的和更接近于基础模式的示例。例如，使用 PEAR DB 是应用设计模式(特别是工厂方法)的示例。Zend Framework 也会使用各种不同的模式(如单元素模式和适配器模式)。