PHP设计模式

phpw

目　录

PHP设计模式

:-: # PHP设计模式*

(美) Aaron Saray 著

梁志敏 蔡建 译

:-: 

:-: 清华大学出版社

:-:

北京

作者简介

作者简介

在Aaron Saray 8岁的时候，他接触到了一台没有永久存储器的二手Commodore64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、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的帐户管理工具，还为基于Internet连接的Point of sales系统开发过后台是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为主的博客。

前言

前言

PHP是目前的主流编程技术。我们可以看到大量的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 本书适用的范围

在确定本书所适用的读者对像时，作者非常难于取舍。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看工作经验的编程人员而编写？是应当为了解PHP面向对像功以折专业读者而编写吗？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PHP4。” 然而，考虑到PHP4仍然被广泛部署，开发人员仍然在使用PHP4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事实上适用的读者。

0.1本书读者对像：

- 必须完全了解PHP语言，或者至少收藏过http://www.php.net 的网址。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编程人员会发现第二间中对PHP高级OO功能的探讨是非常有价值的。
- 必须使用PHP5或更高版本从而具有可用的面向对像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

本书分三大部分：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

第一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概念扩展至体系结构更健全的设计模式领域。

第二章侧重于介绍一些工具，这些工具是在PHP中构建各种设计模式的基础。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，也是设计模式是基本的部分。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像的代码示例。上述内容基本上覆盖了设计模式的主要功能部分，同时也不至于过分冗长。

0.2.3案例分析

本书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，你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，以便确定它是一个公公应用一次的实例，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。

在研究案例的时候，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，必须先进入分析阶段。我们遇到太多这样的例子：在项目开始的阶段，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高本书设定的目标。该部分包括基于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，你应当回顾自己的应用程序以及编码过程中所有的选择，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支持严格的规则，但是必须针对具体的应用程序建立构选块和框架。为了创建体系结构更建全的代码库，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），所以本书的要求非常简单，如下所示：

- Windows 或 linux操作系统

- PHP5.2或更高版本
- MySQL5.0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

学习本书的示例时，读者既可以手动输入所有代码，也可以使用本书附带的源代码文件。本书使用所有源代码都可以从http://www.wrox.com 和 http://www.rupwk.com.cn 上下载。登录到该站点，从Search工具可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。

下载代码后，只需用自己喜欢的解压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，查看本书的其他Wrox图书的所有可用代码。

第I部分 初识设计模式与PHP

第1章理解设计模式

第一章 理解设计模式

通常，如查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！虽然本间的篇幅不止5页，但是请读者一定不要放弃阅读。

在某种程序上，术语“设计模式（Design Pattern）”只是一个新奇的名称，他并非特别复杂。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，并具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

1.1什么是设计模式

下面我们给出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，他最近的工作是一开发一个通过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服中心代表的软件。Stevey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和客户提供某些新的功能。Steve所在的部门也为此增加了两名开发人员：Andy和Jason。

公司副总提出的要求是：在客户完成成功和安全的登录后，公司的站应当允许客户自己进行相关的支付操作。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 显示客户帐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糟糕，但是Jason似乎要花费很长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，以便于更快地完成任务。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实现的方式不同。

随差公司Web站的访问量越来越多，系统的性能开始变的越来越差。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

写入一个类中。Steve则开始查看他为电话客服中心站点编写的身份证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，以便将所有这些耿耿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在软件设计方法所受到的正规教育使他有是会对PHP完成任务的能力以及体系结构提出质疑。Jason无法累世易地理解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，所有有音乐片断都能找到出处。如今，创作新的音乐只是将特定的音符按照不同的顺序和速度进行重新的排列。除了某些实现重大突确的特例之外，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,这四个人是最早介绍设计模式的书籍和作者。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像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：

1. what:要考虑的相关业务和功能需求。
2. how:如何使用特定设计满足上述需求。
3. workk：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和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描绘为解决问题所彩用的载体。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语言的掌握以及对期复杂性的理解，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，如果摇篮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，并具只需将期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设计模式无处不在

我们已经介绍过，Steve的团队能够理解期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一切，有多少次使用自己的有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（特是工厂方法的示例）。zend framework也会使用各种不同模式（如单无素模式和适配器模式）。

1.14设计模式的公共部分

为了描述设计模式，“四人组”首先提出了一个文档标准，并且在著作中将该文档标准应用于所介绍的每种模式。“四人组”之后PHP相关书箱的甩有作者都复制了这种严格的格式，并且不断地普及该文档形式。因为我觉得这方面的相关介绍非常多，并且只是结构问题，所以本书不会再做过多描述。这里只是简单介绍改线 个模式的文档的四个主要部分：

1. 名称
2. 问题与解决方案
3. 统一建模语言（Unified Modeling Language,UML）图以及代码示例

1.名称

在设计模式中，名称实际上比您最初想像的更重要。对于向项目和其他模式解释特定模式的行业和关系来说，适当的描述性命名约定大有帮助。

在本单的示例中，我们注意到Jason向Steve提及自己准备重写支付处理系统的一部分代码。余粮高级编程人员，Steve也许不会认可Jason采用的言法，但是无疑会建议在重写代码体系机构中使用某些模式。因此，整个团队都必须了解支付系弘的基础概念，而Jason要负责具体的实现。

2.问题与解决方案

本单前面曾经介绍过，设计模式是为了使用通用的解决方案来解决相同的问题而出现的。这部分说明覆盖了项目中的主要问题或各种问题，并且表明为何特定的设计模式是其中一个较好的解决方案。

读者也许注意到，本书并没有使用“最佳解决方案”的说法，其原因在于没有人能够给出这样的结论。即命名您发现所选择的设计模工是最适合于特定的问题，但是为了更理想地将期应用于具体的项目，您仍需要准备进行一定的改进工作。

3.UML图

UML图说明了设计模式的通用结构。在某些情况下，为了说明设计模式的其他实现或者以更易于理解的分段形式阐明复杂的概念，可能需要生成多个UML图。

> 什么是UML
>
> 统一建模语言（Unified Modeling Language,UML）图应当是编程工具库中的主要组成部分。UML是对操作、对像和用例进行图形编程的标准方式。在使用PHP构建复杂的软件时，UML图有助于清楚地表达您的设计。要最快地了解UML的最新信息，读者可以访问Wikipedia站点上的 http://wikipedia.org/wiki/Unified_Modeling_Language 页面。

读者可能注意到，我们能够宽松地根据通用模式图给出用于为具体项目生成UML图的构建块。当然，与救命相对，方法名、类名和属性是多样公的，并且更加复杂。

4.代码示例

具有实际经验的PHP编程人员非常喜欢设计模式的最后一个主要部分：代码示例。这些相对简单的救命伤脑筋PHP代码表达设计模式的根念。本书介绍基于PHP的设计模式，其优点在于不必了解采用其他语言编写的模式示例 。侧重于企业级设计模式的其他书籍都是采用由Java或C语言编写的示例，对于使用单一语言的编程人员来说，示例效果稍有影响。

本书会不断地重申：代码示例只是例子，他们不能即插即用。代码示例可能不包含错误日志 记录或处理、审计或完全安全的编程技术。这并非表明我不欣赏高质量的、全安的编程（之前与我同一团队的编程人员可以证明我是一个注重细节的人），而是代码示例可以很好地说明我解释的概念。

1.2设计模式未涵盖的内容

通过介绍设计模式并未涵盖的内容来进行阐述设计模式也十分重要。到上目前为至，读者可能已经注意到本书为设计模式给出的定义涉及了相当大的范围。

1.2.1设计模式并非即插即用

对于具体的项目来说，设计模式并非简单的“即插即用”式的解决方案。只是一套为了解决php程序开发过程中维护和代码复用性的思路或方案，而非解决方案。

1.2.2设计模式是可维护的，但并非总是最有效的

在应用程序的开发中，设计模式并不总是提供最佳的效率和速度。设计模式的目的是帮助我们以便于重复和重用的的方式设计解决方案。这意味着：设计模式可能不是专门适合于特定的环境，但是却具有更好的代码可维护性和可理解性。

1.2.3设计模式是重构的必经之路但并非最终目标

设计模式是重构方法的重要途径，但不应当是最终目标。我们赞成使用一组非常详细的设计模式体系结构规范来启动一个项目，但是并不希望为了应用模式而套用某种模式。

1.3设计模式的相关论证

为了不使读者由于语言特性的不同而混淆核心概念，过去的大多数设计模式示例都非常简单和偏于理论化。过去学习过设计模式或者面向对象编程的读者会非常熟悉常见的正方形、圆形和椭圆形对象示例。

有关设计模式的书籍在其示例中大量使用简单的对象(如正方形和人)进行讨论。完美主义者认为应当尽可能详细地阐述设计模式的概念和实际应用，并且给出最简单的示例，从而不必分心于设计模式的真正实现。在数学课上，由于额外信息而讨厌应用题的人就属于这种完美主义者。以我的经验，未经过正规培训的PHP编程人员情愿看到更详尽的、采用代码形式展现的概念示例。在工作之初，这些编程人员可能通过复制和粘贴代码学到很多知识。

本书介绍的设计模式包含了说明模式的小规模和中等规模PHP代码示例。这种两阶段方式将模式的实际概念解释(适用于需要了解特定结构的人)与基于示例的模式论证(适用于更喜欢亲自动手的初学者)组合在一起。

1.4在PHP中使用设计模式的原因

PHP支持可供企业使用的引擎，这便于初学者的学习。通过在己有的HTML文档中插入若千行代码，我们就可以进入PHP的世界。简单地将文件扩展名.html修改为.php，添加一些代码片段，将文件部署至PHP服务器，您就成为一名真正的PHP编程人员了。在Zend认证工程师(ZendCertifiedEngineer，ZCE)认证出现之前，PHP编程人员不必具有高超的技能。即使成为了ZCE,PHP编程人员仍然可能缺乏开发可供企业使用的、体系健全的应用软件所需的某些基木知识。

如果觉得本章开始部分介绍的示例还不够令人鼓舞，那么我们可以告诉您：许多商用级播放器都是使用PHP开发的。因为PHP较为简陋，所以不像主要的企业级编程语言那样引人注口。然而，Zend的努力工作以及大型Internet公司(例如Yahoo!和Amazon)对PHP的广泛采用说明了PHP是可供企业使用的。随着企业级软件需求的引入，企业级方法也相应地不断增加。

目前，PHP己经能支持基于本书所介绍各种概念的大景构建块。在使用PHP3或PHP/FI的时期，应用这些类型的设计模式可能非常困难或者无法实现。不要曲解我的意思，设计模式是以语言形式存在的，对于某些人来说，本书及本书中的示例可能毫无用处。

1.5本章小结

本章使用了一个普通的编程示例讨论了在日常编程工作中流行的设计模式。通过扩展对模式的理解，读者可以将这个示例与实际的设计模式关联在一起。通过查看设计模式所涉及的领域以及未涵盖的内容，我们为设计模式提供了一个更为简明的定义。最后，本章指出PHP支持构建基本设计模式，并且提到PHP在某些大型企业中的地位，从而说明了在PHP中使用设计模式的原因。

既然已经理解了设计模式，下面就开始介绍对PHP使用人有帮助的、目前可用的.工具。

第2章使用现有的工具

第1章详细阐述了什么是设计模式，本章将介绍目前的工具集并且查看已使用这些工具的场合。

第1章简单地提及了常见的PHP架构和库(例如PEAR和ZendFramework),本章将深
入介绍这些架构和库，从而使读者更详细地了解广泛使用的各种设计模式。此外，读者还将接触到一个以前就可能知道的架构。

PHP5引入了一个新的、称为SPL的类与函数标准集。在进行简单介绍后，本章会详细讨论为参考章节创建设计模式代码示例时有用的功能。

最后，本章还将介绍EclipsePDTIDE中有助于创建和复制设计模式的功能。

2.1已有架构中的模式

最普通的PHP架构能够成功，原因在于以下几点：细致的体系结构、可维护性与可扩展性。这也正是在最初体系结构中恰当使用设计模式的全部目标。本节将针对每种架构给出数个模式使用示例，其目的在于证明这些模式在日常编程中越来越普遍，从而使读者不会过分畏惧学习参考章节的内容。如果读者对这些架构中某种设计模式的示例感兴趣，那么可以翻阅相应的参考章节以进一步了解相关的详细信息。

设计模式只是用于构造程序的模板，重申这一观点非常重要。并非每个基于设计模式的体系结构都是某种模式的目标，并且也不会严格遵循该模式的手册规范。读者将会频繁看到：类是使用多个设计模式创建的，或者需将基本模式进行较大调整才能使其适用于特定的环境。

2.1.1PEAR中的设计模式

PEAR是一个较早的PHP扩展库。PEAR HPExtensionandApplicationRepository

2.2PHP标准库

2.3使用具有模式的EclipsePDT

2.4本章小结

第II部分 参考内容

第3章适配器模式

第4章建造者模式

第5章数据访问对象模式

第6章装饰器模式

第7章委托模式

第8章外观模式

第9章工厂模式

第10章解释器模式

第11章迭代器模式

第12章中介者模式

第13章观察者模式

第14章原型模式

第15章代理模式

第16章单元模式

第17章策略模式

第18章模板模式

第19章访问者模式

第III部分 PHP设计案例分析

第20章需求分析