

TURING

图灵程序设计丛书

Apress®

PHP Objects, Patterns, and Practice
Fifth Edition

深入PHP

面向对象、模式与实践

(第5版)

[英] 马特·赞德斯彻 著 杨文轩 译

- 前Yahoo!公司PHP高级开发人员20余年企业级Web开发经验总结
- 用通俗易懂的方式讲解复杂的面向对象编程原则和模式



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS

TURING

图灵程序设计丛书

PHP Objects, Patterns, and Practice

Fifth Edition

深入PHP

面向对象、模式与实践

(第5版)

[英] 马特·赞德斯彻 著 杨文轩 译

人民邮电出版社

北 京

图书在版编目 (C I P) 数据

深入PHP：面向对象、模式与实践：第5版 / (英)
马特·赞德斯彻 (Matt Zandstra) 著；杨文轩译. --
北京：人民邮电出版社，2019.6
(图灵程序设计丛书)
ISBN 978-7-115-51233-8

I. ①深… II. ①马… ②杨… III. ①PHP语言—程序
设计 IV. ①TP312.8

中国版本图书馆CIP数据核字(2019)第094215号

内 容 提 要

本书是 PHP 经典图书升级版，它既是一本关于面向对象设计与编程的书，也是一本关于如何使用工具管理 PHP 代码（从协作到部署）的书。书中讲解了 PHP 的新特性，例如匿名类、标量参数提示和返回值类型。第 5 版重写了 Composer 和 Packagist 库的相关内容，并增加了关于 Git 版本控制的篇幅。示例代码全面更新，符合 PSR-1 和 PSR-2 标准。阅读本书能够帮你构建实现既定目标且易于协同开发的系统，并让你的代码优雅、简洁且易于理解。

本书适合中、高级 PHP 程序员，以及任何对 PHP 感兴趣的 Web 开发人员阅读。

-
- ◆ 著 [英] 马特·赞德斯彻
译 杨文轩
责任编辑 温 雪
责任印制 周昇亮
 - ◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路11号
邮编 100164 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：32.5
字数：768千字 2019年6月第1版
印数：1-3 500册 2019年6月北京第1次印刷
著作权合同登记号 图字：01-2018-5947号
-

定价：129.00元

读者服务热线：(010)51095183转600 印装质量热线：(010)81055316

反盗版热线：(010)81055315

广告经营许可证：京东工商广登字 20170147 号

站在巨人的肩上
Standing on Shoulders of Giants



iTuring.cn

站在巨人的肩上
Standing on Shoulders of Giants



iTuring.cn

版 权 声 明

Original English language edition, entitled *PHP Objects, Patterns, and Practice, Fifth Edition* by Matt Zandstra, published by Apress, 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705 USA.

Copyright © 2016 by Matt Zandstra. Simplified Chinese-language edition copyright © 2019 by Posts & Telecom Press. All rights reserved.

本书中文简体字版由 Apress L.P. 授权人民邮电出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

献给 Louise，你是我的至爱。

前言

在我初次构思本书时，PHP 中的面向对象设计还是一个非常深奥的话题。不过在这几年，我不仅看到了 PHP 作为面向对象语言正在势不可挡地崛起，还看到了框架的演进。框架的作用当然毋庸置疑。它们负责管理许多（现如今应该是绝大多数）Web 应用程序的内容以及整合。不过就本书的主题而言，更重要的是它们都实践了本书所探讨的设计原则。

然而如同所有有用的 API 一样，对于开发者而言，这里也暗藏着一种危险。开发者害怕他们被置于用户区，当出现问题的时候，他们就不得不等待千里之外的某位大师来修复 bug，或按照他的意愿加入新特性。这时从某种意义上说，开发者距离“流放”仅有一步之遥：他们所做的工作都必须基于框架中的“高级魔术”般的特性，而且都只不过是为一个强大的、不可知的基础设施添加一些小装饰。

虽然我要重新发明轮子，但是我的论点主旨并非是我们都应该抛弃框架并从头开始构建 MVC 应用程序（至少并非总是如此）。相反，作为开发人员，我们应该理解框架能够解决的问题，以及它们解决问题时所采用的策略。我们应当不仅仅能够从功能上评估框架，还能够从框架开发者所做的设计决策方面评估框架，以及判断框架的实现质量。当判定框架适用于项目时，接下来就可以基于框架来开发自己的应用程序了。随着开发的深入，我们还需要编译自己的可复用代码库。

我希望本书能够帮助 PHP 开发者以设计为导向实现平台与代码库，并希望本书中所讲述的概念工具能够确实帮助到 PHP 开发者。^①

^① 可访问本书图灵社区页面（<http://www.it-ebooks.com.cn/book/1352>）下载示例代码并提交中文版勘误。——编者注

致 谢

和前几版一样，第 5 版的问世同样得到了许多人的帮助。但是，我也必须一如既往地回顾一下写书的契机。在布莱顿的一次演讲中，我试着提炼出了书中的一些基本概念，那时我们刚开始对 PHP 5 的无限未来感到惊奇。感谢 Andy Budd 主持了那场演讲，感谢充满活力的布莱顿开发者社区。还要感谢当时在场的 Jessey White-Cinis，是她帮助我联系上了 Apress 出版社的 Martin Streicher。

再一次感谢为我提供巨大支持、反馈和鼓励的 Apress 团队。很荣幸，我能够从他们的专业精神中有所收益。

在计划第 5 版时，我一口答应会更新所有的代码，使之遵守最新的编码风格规范。我很荣幸我的朋友兼同事 Paul Tregoin 担任了第 5 版的技术评审员。他不仅让我兑现了自己的承诺，而且修正了代码中许多违反标准的地方，超预期地完成了他的工作。此外，Paul 的知识储备、洞察力以及对细节的关注也对第 5 版大有裨益。非常感谢 Paul！

感谢我的妻子 Louise 以及我的孩子 Holly 和 Jake，我爱你们。是你们让我在写书之余能够享受我所渴望的家庭时光。感谢 Steven Metsker 允许我使用 PHP 重新实现了他在其著作 *Building Parsers with Java* 中提到的解析器 API 的简化版本。

我总是边听音乐边写作。在编写本书之前的版本时，我常听 John Peel 的节目。他是伟大的 DJ，也是地下音乐和 eclectic 风的捍卫者。第 5 版的内容大部分是伴随着循环播放的 BBC Radio 3 当代音乐节目 *Late Junction* 写成的。感谢这些音乐让我激情常在。

目 录

第一部分 对象

第 1 章 PHP: 设计和管理..... 2

- 1.1 问题..... 2
- 1.2 PHP 与其他语言..... 3
- 1.3 关于本书..... 5
 - 1.3.1 对象..... 5
 - 1.3.2 模式..... 6
 - 1.3.3 实践..... 6
 - 1.3.4 第 5 版新增内容..... 7
- 1.4 小结..... 8

第 2 章 PHP 和对象..... 9

- 2.1 PHP 对象的偶然成功..... 9
 - 2.1.1 起源: PHP/FI..... 9
 - 2.1.2 语法糖: PHP 3..... 9
 - 2.1.3 一场静悄悄的革命: PHP 4..... 10
 - 2.1.4 拥抱变化: PHP 5..... 11
 - 2.1.5 迎头追赶: PHP 7..... 12
- 2.2 拥护和顾虑: 关于对象的争辩..... 12
- 2.3 小结..... 13

第 3 章 对象基础..... 14

- 3.1 类和对象..... 14
 - 3.1.1 第一个类..... 14
 - 3.1.2 一个 (或两个) 对象..... 15
- 3.2 设置类中的属性..... 16
- 3.3 使用方法..... 18
- 3.4 参数和类型..... 21
 - 3.4.1 基本类型..... 21

- 3.4.2 接受提示: 对象类型..... 24
- 3.5 继承..... 28
 - 3.5.1 继承问题..... 28
 - 3.5.2 使用继承..... 33
 - 3.5.3 public、private 和 protected:
管理类的访问..... 38
- 3.6 小结..... 43

第 4 章 高级特性..... 44

- 4.1 静态方法和属性..... 44
- 4.2 常量属性..... 47
- 4.3 抽象类..... 48
- 4.4 接口..... 50
- 4.5 trait..... 52
 - 4.5.1 trait 可以解决的问题..... 52
 - 4.5.2 定义和使用 trait..... 53
 - 4.5.3 使用多个 trait..... 54
 - 4.5.4 组合使用 trait 与接口..... 55
 - 4.5.5 通过 insteadof 管理方法名
冲突..... 56
 - 4.5.6 使用别名重写 trait 的方法..... 57
 - 4.5.7 在 trait 中使用静态方法..... 58
 - 4.5.8 访问宿主类的属性..... 59
 - 4.5.9 在 trait 中定义抽象方法..... 59
 - 4.5.10 改变 trait 中方法的访问权限..... 60
- 4.6 延迟静态绑定: static 关键字..... 61
- 4.7 错误处理..... 65
- 4.8 final 类和方法..... 73
- 4.9 内部错误类..... 74
- 4.10 使用拦截器..... 75

4.11 定义析构方法	81	6.6 忘记细节	133
4.12 使用__clone()复制对象	82	6.7 四个方向标	133
4.13 定义对象的字符串值	85	6.7.1 代码重复	134
4.14 回调、匿名函数和闭包	86	6.7.2 类知道太多	134
4.15 匿名类	90	6.7.3 万能的类	134
4.16 小结	92	6.7.4 条件语句	134
第5章 对象工具	93	6.8 UML	134
5.1 PHP 和包	93	6.8.1 类图	135
5.1.1 PHP 包和命名空间	93	6.8.2 序列图	140
5.1.2 自动加载	101	6.9 小结	142
5.2 类函数和对象函数	105		
5.2.1 查找类	106		
5.2.2 检查对象或类	106		
5.2.3 得到指向类的完全限定的字符串引用	107		
5.2.4 检查方法	108		
5.2.5 检查类属性	110		
5.2.6 检查继承	110		
5.2.7 方法调用	111		
5.3 反射 API	112		
5.3.1 入门	112		
5.3.2 是时候大干一场了	113		
5.3.3 检查类	115		
5.3.4 检查方法	117		
5.3.5 检查方法参数	118		
5.3.6 使用反射 API	120		
5.4 小结	123		
第6章 对象与设计	124		
6.1 定义代码设计	124		
6.2 面向对象编程与面向过程编程	125		
6.2.1 职责	129		
6.2.2 内聚	129		
6.2.3 耦合	129		
6.2.4 正交	129		
6.3 选择类	130		
6.4 多态	131		
6.5 封装	132		
		第二部分 模式	
		第7章 什么是设计模式，为什么要使用设计模式	144
		7.1 什么是设计模式	144
		7.2 设计模式概要	146
		7.2.1 名称	146
		7.2.2 问题	146
		7.2.3 解决方案	147
		7.2.4 效果	147
		7.3 《设计模式》的格式	147
		7.4 为什么使用设计模式	148
		7.4.1 设计模式定义了问题	148
		7.4.2 设计模式定义了解决方案	148
		7.4.3 设计模式与编程语言无关	148
		7.4.4 模式定义了一组词汇	148
		7.4.5 模式是经过测试的	149
		7.4.6 模式为协作而设计	149
		7.4.7 设计模式促进优秀设计	149
		7.4.8 流行的框架都使用了设计模式	150
		7.5 PHP 与设计模式	150
		7.6 小结	150
		第8章 一些模式原则	151
		8.1 模式的启示	151
		8.2 组合与继承	152
		8.2.1 问题	152

8.2.2 使用组合	155	第 10 章 使面向对象编程更加灵活的 模式	192
8.3 解耦	157	10.1 构造可灵活创建对象的类	192
8.3.1 问题	157	10.2 组合模式	192
8.3.2 解耦	159	10.2.1 问题	193
8.4 针对接口编程, 而不是针对实现 编程	161	10.2.2 实现	195
8.5 概念在变化	162	10.2.3 效果	199
8.6 不要盲从模式	162	10.2.4 组合模式小结	202
8.7 模式	163	10.3 装饰器模式	202
8.7.1 用于生成对象的模式	163	10.3.1 问题	202
8.7.2 用于组织对象和类的模式	163	10.3.2 实现	205
8.7.3 面向任务的模式	163	10.3.3 效果	209
8.7.4 企业设计模式	163	10.4 外观模式	209
8.7.5 数据库模式	163	10.4.1 问题	209
8.8 小结	163	10.4.2 实现	211
第 9 章 生成对象	164	10.4.3 效果	211
9.1 生成对象的问题和解决方案	164	10.5 小结	212
9.2 单例模式	168	第 11 章 执行及描述任务	213
9.2.1 问题	169	11.1 解释器模式	213
9.2.2 实现	169	11.1.1 问题	213
9.2.3 效果	171	11.1.2 实现	214
9.3 工厂方法模式	172	11.1.3 解释器模式的问题	222
9.3.1 问题	172	11.2 策略模式	222
9.3.2 实现	175	11.2.1 问题	222
9.3.3 效果	177	11.2.2 实现	223
9.4 抽象工厂模式	177	11.3 观察者模式	227
9.4.1 问题	177	11.4 访问者模式	235
9.4.2 实现	178	11.4.1 问题	235
9.4.3 效果	180	11.4.2 实现	236
9.5 原型模式	181	11.4.3 访问者模式的问题	241
9.5.1 问题	182	11.5 命令模式	242
9.5.2 实现	183	11.5.1 问题	242
9.6 推向边缘: 服务定位器	186	11.5.2 实现	242
9.7 完全隔离: 依赖注入	187	11.6 空对象模式	247
9.7.1 问题	187	11.6.1 问题	247
9.7.2 实现	188	11.6.2 实现	249
9.7.3 效果	191	11.7 小结	251
9.8 小结	191		

第 12 章 企业设计模式	252	13.6.1 问题	326
12.1 架构概述	252	13.6.2 实现	326
12.1.1 模式	252	13.6.3 效果	327
12.1.2 应用与分层	253	13.7 标识对象	329
12.2 企业架构外的基础模式	255	13.7.1 问题	329
12.2.1 注册表	255	13.7.2 实现	330
12.2.2 实现	256	13.7.3 效果	335
12.2.3 效果	260	13.8 选择工厂与更新工厂模式	335
12.3 表示层	260	13.8.1 问题	336
12.3.1 前端控制器	261	13.8.2 实现	336
12.3.2 应用控制器	271	13.8.3 效果	340
12.3.3 页面控制器	283	13.9 现在映射器中还剩下什么	340
12.3.4 模板视图和视图助手	288	13.10 小结	342
12.4 业务逻辑层	291		
12.4.1 事务脚本	291	第三部分 实践	
12.4.2 领域模型	295	第 14 章 优秀（以及糟糕）的实践	346
12.5 小结	298	14.1 超越代码	346
第 13 章 数据库设计模式	299	14.2 借轮子	347
13.1 数据层	299	14.3 合作愉快	348
13.2 数据映射器	299	14.4 为代码插上翅膀	349
13.2.1 问题	300	14.5 标准	350
13.2.2 实现	300	14.6 Vagrant	350
13.2.3 效果	313	14.7 测试	351
13.3 标识映射	315	14.8 持续集成	351
13.3.1 问题	315	14.9 小结	352
13.3.2 实现	315	第 15 章 PHP 标准	353
13.3.3 效果	318	15.1 为什么需要标准	353
13.4 工作单元	319	15.2 什么是 PSR	354
13.4.1 问题	319	15.2.1 为什么选择 PSR	354
13.4.2 实现	319	15.2.2 哪些人需要 PSR	355
13.4.3 效果	323	15.3 编码风格	355
13.5 延迟加载	323	15.3.1 PSR-1 基础编码规范	356
13.5.1 问题	323	15.3.2 PSR-2 编码风格规范	358
13.5.2 实现	324	15.3.3 检查和修改代码	360
13.5.3 效果	326	15.4 PSR-4 自动加载规范	362
13.6 领域对象工厂	326	15.5 小结	365

第 16 章 通过 Composer 使用和创建组件.....366

- 16.1 什么是 Composer.....366
- 16.2 安装 Composer.....367
- 16.3 安装一个(组)包.....367
 - 16.3.1 通过命令行安装包.....368
 - 16.3.2 版本.....368
 - 16.3.3 require-dev 元素.....369
- 16.4 Composer 与自动加载.....370
- 16.5 创建自己的包.....371
 - 16.5.1 添加包信息.....371
 - 16.5.2 平台软件包.....372
- 16.6 通过 Packagist 分发包.....373
- 16.7 私有包.....376
- 16.8 小结.....377

第 17 章 用 Git 进行版本控制.....378

- 17.1 为什么进行版本控制.....378
- 17.2 安装 Git.....379
- 17.3 使用在线 Git 代码库.....380
- 17.4 配置 Git 服务器.....382
- 17.5 启动项目.....384
- 17.6 更新与提交.....387
- 17.7 文件和目录的添加与移除.....390
 - 17.7.1 添加文件.....390
 - 17.7.2 删除文件.....390
 - 17.7.3 添加目录.....391
 - 17.7.4 删除目录.....391
- 17.8 标记一次发布.....392
- 17.9 创建分支.....393
- 17.10 小结.....398

第 18 章 使用 PHPUnit 进行测试.....399

- 18.1 功能测试与单元测试.....399
- 18.2 手动测试.....400
- 18.3 引入 PHPUnit.....402
 - 18.3.1 创建测试用例.....402
 - 18.3.2 断言方法.....405
 - 18.3.3 测试异常.....406

- 18.3.4 运行测试套件.....407
- 18.3.5 约束.....407
- 18.3.6 mock 和 stub.....409
- 18.3.7 失败是成功之母.....412
- 18.4 编写 Web 测试.....415
 - 18.4.1 为测试重构 Web 应用.....415
 - 18.4.2 简单的 Web 测试.....417
 - 18.4.3 引入 Selenium.....419
- 18.5 警告.....424
- 18.6 小结.....426

第 19 章 使用 Phing 进行自动化构建.....427

- 19.1 Phing 是什么.....427
- 19.2 获取和安装 Phing.....428
- 19.3 编写构建文档.....428
 - 19.3.1 目标.....430
 - 19.3.2 属性.....432
 - 19.3.3 类型.....438
 - 19.3.4 任务.....443
- 19.4 小结.....446

第 20 章 Vagrant.....447

- 20.1 问题.....447
- 20.2 设置.....448
- 20.3 挂载本地目录到 Vagrant 镜像.....450
- 20.4 配置.....451
 - 20.4.1 设置 Web 服务器.....452
 - 20.4.2 设置 MySQL.....452
 - 20.4.3 配置主机名.....453
- 20.5 结束语.....455
- 20.6 小结.....455

第 21 章 持续集成.....456

- 21.1 什么是持续集成.....456
 - 21.1.1 准备一个持续集成项目.....458
 - 21.1.2 安装 Jenkins 插件.....467
 - 21.1.3 设置 Git 公钥.....468
 - 21.1.4 创建新项目.....469
 - 21.1.5 运行第一次构建.....472

21.1.6 配置报告	472	22.2.2 模式与设计原则	480
21.1.7 触发构建	474	22.3 实践	482
21.2 小结	476	22.3.1 测试	482
第 22 章 对象、模式和实践	477	22.3.2 标准	483
22.1 对象	477	22.3.3 版本控制	483
22.1.1 选择	478	22.3.4 自动构建	483
22.1.2 封装与委托	478	22.3.5 持续集成	484
22.1.3 解耦	478	22.3.6 我们遗漏了什么	484
22.1.4 可复用性	479	22.4 小结	485
22.1.5 美学	479	附录 A 参考文献	486
22.2 模式	479	附录 B 一个简单的解析器	488
22.2.1 模式给我们带来了什么	480		

第一部分

对 象

- 第 1 章 PHP: 设计和管理
- 第 2 章 PHP 和对象
- 第 3 章 对象基础
- 第 4 章 高级特性
- 第 5 章 对象工具
- 第 6 章 对象与设计

PHP 5.0 发布于 2004 年 7 月。该版本对 PHP 进行了多项重大改进，其中最重要的恐怕是加强了对面向对象编程的支持。这激发了 PHP 社区对对象和设计的兴趣。事实上，这是自 PHP 4 首次让 PHP 走上了面向对象编程的道路后迈出的又一大步。

本章将讲解面向对象编程可以满足哪些需求，并对模式和相关实践的演变进行简要总结。本章还会概述全书要讨论的主题，包括以下内容。

- 灾难的衍化：项目走向失败。
- 设计与 PHP：面向对象技术如何扎根于 PHP 社区。
- 本书：对象、模式、实践。

1.1 问题

问题来了，PHP 太易于使用了，它吸引你使用它实现自己的想法，并回报给你非常棒的结果。你可以直接在网页中编写大量代码，因为 PHP 本身就支持这种功能。你可以在文件中编写工具函数（例如访问数据库的代码），然后这些文件会被各个网页引用。不知不觉中，你的 Web 应用程序已经可以正常工作了。

不过，你正走在通往毁灭的道路上。你当然不会意识到这个问题，因为你的网站看起来非常棒。它运行得很顺畅，你的客户很高兴，你的用户也愿意在这个网站上购物。

但当你进入下一个开发阶段时，麻烦就来了。现在你有了更大的团队、更多的用户以及更充足的预算。但事情在毫无征兆的情况下每况愈下，你的项目就仿佛中了毒一样。

虽然代码有一点复杂，但对你来说并不难理解。不过，新来的程序员得费尽心思才能理解代码。要完全发挥出他作为团队一员的作用，所需的时间远比你想象的长。

预计只需一天就可以完成的一个简单修改，实际可能需要三天，因为你发现你需要修改 20 个或者更多的网页。

一位程序员修改的代码覆盖了你之前对同一段代码所做的重要修改。而这个问题直到三天后，当你修改你的本地代码副本时才暴露出来。你需要一天时间来解决这个问题，而这会妨碍正在修改这个文件的第三位程序员的工作。

随着用户数量的增加，你需要将程序迁移到一台新的服务器上。整个项目需要手动部署，而