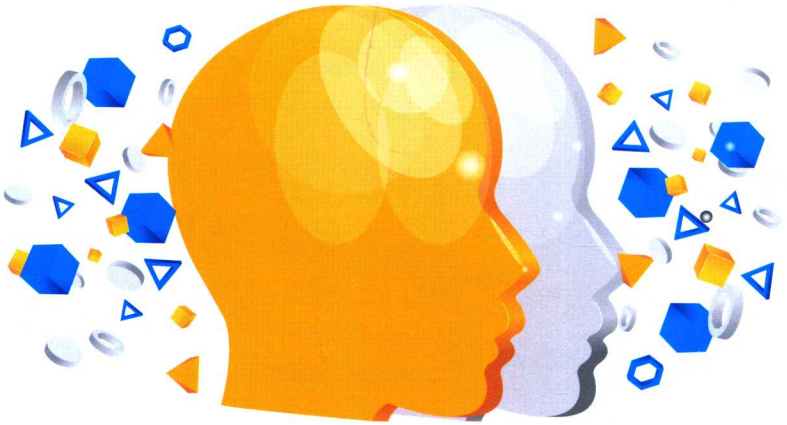
系统：概念、流程、技巧一网打尽
全面：全方位、全流程提升面向对象技术能力
扎实：书中的每一部分都是作者十余年开发经验的总结

Broadview®
www.broadview.com.cn



面向对象葵花宝典

思想、技巧与实践

李运华 编著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

面向对象葵花宝典
思想、技巧与实践

李运华 编著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

本书系统地讲述了面向对象技术的相关内容，包括面向对象的基本概念、面向对象开发的流程、面向对象的各种技巧，以及如何应用面向对象思想进行架构设计。在讲述相关知识或技术的时候，除了从“是什么”这个角度进行介绍外，更加着重于从“为什么”和“如何用”这两个角度进行剖析，力争让读者做到“知其然，并知其所以然”，从而达到在实践中既能正确又能优秀地应用面向对象的相关技术和技巧。

本书的内容涵盖广泛，无论读者已有的水平如何，都能够从书中获益。

- 如果你是刚入门的程序员，书中有详尽的和通俗易懂的概念和方法介绍，有完整的面向对象开发流程，让你能够快速掌握面向对象开发的基本技巧，从容不迫地完成开发任务；
- 如果你有了一定经验，能够熟练应用各种面向对象技术和技巧，但却没有深入地去探索，书中有独辟蹊径的解读，能够让你“知其然，并知其所以然”；
- 如果你已经成为了面向对象的大牛程序员，但苦于不能继续提升，书中有“面向对象架构设计”的点石成金术，让你从程序员秒变架构师。

图书在版编目（CIP）数据

面向对象葵花宝典：思想、技巧与实践 / 李运华编著. —北京：电子工业出版社，2015.12
ISBN 978-7-121-27473-2

Ⅰ. ①面… Ⅱ. ①李… Ⅲ. ①面向对象语言－程序设计 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字（2015）第 258098 号

策划编辑：陈晓猛
责任编辑：葛 娜
印 刷：三河市双峰印刷装订有限公司
装 订：三河市双峰印刷装订有限公司
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编 100036
开 本：787×980 1/16 印张：23.75 字数：460 千字
版 次：2015 年 12 月第 1 版
印 次：2015 年 12 月第 1 次印刷
印 数：3000 册 定价：69.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

我最早接触“面向对象”这个词是在大学的选修课“C++语言”这门课程上，当时的教材分为两部分，前面一部分是讲 C 语言编程，后面一部分是讲 C++的面向对象特性。当时觉得“面向对象”这个词很有意思，但由于教材和教学的原因，当时对“面向对象”的理解和认识很粗浅，认为面向对象就是写 class。

工作后开始真正地在业务开发中应用面向对象技术，最初也只是按照写 class 的方式去应用面向对象，但越来越发现不对劲，自己的疑惑也越来越多。例如：

为什么要这样写 class，不能那样写 class，反正最后都能完成功能啊?

为什么我说要设计这个 class，你说要设计那个 class，标准是什么?

客户需求都是用自然语言描述的，根本没有类和继承这些面向对象的概念，那这些设计又是如何得出的呢? 是拍脑袋还是靠天才的创造，抑或是有章可循?

如何判断面向对象的设计做得好还是不好呢?

带着这些问题，我开始真正地去探索面向对象的世界，于是一头扎进去，发现了一个精彩但又纷繁复杂的世界，各种各样的技术扑面而来，让人目不暇接——设计模式、SOLID 设计原则、UML、Java、C++……费了九牛二虎之力，好不容易把这些啃完了，我自信满满地以为掌握了面向对象的精髓，于是又准备到业务开发中大展拳脚，稍微一实践后却发现自己只是掌握了一大堆的技术名词，例如“封装”、“继承”、“工厂模式”、“开闭原则”，但还是“知其然，不知其所以然”，在实际开发过程中依然不能很好地应用。

除此之外，不管采用“瀑布流程”，还是“敏捷”流程，软件开发流程都可以简单地分为需求分析、系统设计、编码实现、测试部署这几个阶段。在这些不同的阶段中，面向对象起什么作用呢? 如何在这些不同的阶段中应用面向对象技术呢? 我曾经带着这些疑问去问大虾大牛，但没人能够清楚地回答；也曾经到各种面向对象书籍中探索，但没有找到确切答案。

虽然没有直接找到答案，但在这个思考和探索的过程中，也吸收到了更多的知识、技能和思想。随着我对面向对象思想和技术的理解逐渐加深，加上我在工作中不断地实践，很多问题我自己慢慢地竟然想通了，同时也形成了一套完整的面向对象方法论。

在这个过程中，有很多疑惑，甚至有很多痛苦，投入也很大，是对技术的热爱支撑着我一直探索下去，但我相信大部分程序员没有这么大的兴趣去探索，也没有那么多的时间可以投入，所以我就写成了这本书，让更多的程序员更好地掌握面向对象的思想和技巧，享受程序人生，实现自己的梦想！

本书内容

本书内容主要分为 4 部分。

面向对象基础：通过对面向对象的历史、发展，与面向过程的对比等相关背景知识的介绍，让读者对面向对象有一个更完整的认识；并深入地阐述了面向对象的各种概念，让读者“知其然，并知其所以然”。

面向对象实战：通过一个实例，完整地介绍了面向对象相关技术如何在软件开发流程中落地，整个面向对象的开发流程一环扣一环，步步为营，让读者避免“拍脑袋”、“头脑风暴”式的开发方式。

面向对象技巧：对“内聚耦合”、“设计模式”、“设计原则”、“UML”等最常见的面向对象技术进行了深入和别具一格的阐述，让读者不但知道“what”（是什么），还能知道“why”（为什么）和“how”（如何用）。

面向对象架构设计：讲解了面向对象思想在架构设计中的应用，只要掌握了面向对象的思想，程序员也可以秒变架构师。

由于面向对象相关的知识和技术纷繁复杂，在一本书里面浓缩这么多的知识，对个人而言是一个很大的挑战，加上很多资料都是来自于网络和英文文档，其中错误与疏漏在所难免，希望读者批评指正。

致谢

在本书写作过程中参考了大量的网络资料，包括维基百科、百度百科等，向这些促进知识传播的网络平台致以诚挚的谢意。

个人在探索的过程中，阅读了大量的面向对象相关的书籍和文档，向这些促进面向对象技术发展的前辈致以诚挚的谢意。

特别感谢我就职的优视科技公司（UCWEB），良好的技术氛围、宽松的工作环境、不断发展和成长的业务，为本书的写作创造了良好的外部环境。

衷心感谢父母和妻子，在本书的写作过程中承担了更多的家庭琐事，给予我最大的支持。

目　录

第 1 部分　面向对象基础

第 1 章　面向对象概述

面向对象是目前最流行的一种程序设计和实现思想，无论你是从事企业级开发、互联网应用开发，还是手机软件开发，都会使用到面向对象的技术；在主流的编程语言中，C++、Java、C#、PHP、Python 等都是支持面向对象的语言；在编程排行榜前十的语言中，面向对象的编程语言能够稳定占据 7～8 席……所有的这些现象，都展示了面向对象的流行程度和受欢迎程度。如果你是一个程序员，但却不懂面向对象，那么真的可能被别人认为是“程序猿”了，因为这样的程序员既像“猿”一样原始，又像“猿”一样稀少。

1.1　程序设计思想的发展

史前时代：面向机器

最早的程序设计是采用机器语言来编写的，直接使用二进制码来表示机器能够识别和执行的指令和数据。简单来说，就是直接编写 0 和 1 的序列来代表程序语言。例如，使用 0000 代表加载（LOAD），使用 0001 代表存储（STORE）等。

机器语言由机器直接执行，速度快，但一个很明显的缺点就是：写起来实在是太困难了，一旦发现自己写错了，改起来更头疼！这样直接导致程序的编写效率十分低下，编写程序花费的时间往往是实际运行时间的几十倍或几百倍。

有一个关于机器语言和比尔·盖茨的笑话，是说比尔·盖茨拿着绣花针在一张光盘上戳，把 Windows 给戳出来了！但如果真的让你去戳，不要说 Windows，连一个简单的“Hello world”都要让人戳到眼睛冒烟！

由于机器语言实在是太难编写了，于是就发展出了汇编语言。汇编语言亦称符号语言，用助记符代替机器指令的操作码，用地址符号（Symbol）或标号（Label）代替指令或操作数的地址。由于汇编语言是采用助记符来编写程序的，比用机器语言的二进制代码编程要

方便些，在一定程度上简化了编程过程。例如，使用 LOAD 来代替 0000，使用 STORE 来代替 0001。

即使汇编语言相比机器语言提升了可读性，但其本质上还是一种面向机器的语言，需要编写者精确掌握 CPU 指令、寄存器、段地址等底层硬件的细节，编写同样困难，也很容易出错。相信很多计算机专业毕业的学生，至今都会对学校的汇编课程中的练习程序心有余悸。

脱离机器第一步：面向过程

在通常情况下，面向机器的语言被认为是一种“低级语言”，为了解决面向机器的语言存在的问题，计算机科学的前辈们又创建了面向过程的语言。面向过程的语言被认为是一种“高级语言”。相比面向机器的语言来说，面向过程的语言已经不再关注机器本身的操作指令、存储等方面，而是关注如何一步一步地解决具体的问题，即解决问题的过程，这应该也是面向过程说法的来由。

相比面向机器的思想来说，面向过程是一次思想上的飞跃，将程序员从复杂的机器操作和运行的细节中解放出来，转而关注具体需要解决的问题；面向过程的语言也不再需要和具体的机器绑定，从而具备了移植性和通用性；面向过程的语言本身也更加容易编写和维护。这些因素叠加起来，大大减轻了程序员的负担，提升了程序员的工作效率，从而促进了软件行业的快速发展。

典型的面向过程的语言有：COBOL、FORTRAN、BASIC、C 语言等。

第一次软件危机：结构化程序设计

主要参考：http://zh.wikipedia.org/wiki/%E8%BD%AF%E4%BB%B6%E5%8D%B1% E6%9C%BA

随着计算机硬件的飞速发展，以及应用的复杂度越来越高，软件规模越来越大，原有的程序开发方式已经越来越不能满足需求了。20 世纪 60 年代中期爆发了第一次软件危机，典型的表现有软件质量低下、项目无法如期完成、项目严重超支等，因为软件而导致的重大事故时有发生。例如，1963 年美国的水手一号火箭发射失败事故（http://en.wikipedia.org/wiki/Mariner_1），就是因为一行 FORTRAN 代码错误导致的。

最典型的软件危机的例子莫过于 IBM 的 System/360 的操作系统开发。佛瑞德·布鲁克斯（Frederick P. Brooks, Jr.）作为项目主管，率领 2000 多个程序员夜以继日地工作，共

计花费了 5000 人一年的工作量，写出将近 100 万行的源码，总共投入 5 亿美元，是美国的“曼哈顿”原子弹计划投入的 1/4。尽管投入如此巨大，但项目进度却一再延迟，软件质量也得不到保障。布鲁克斯后来基于这个项目经验而总结的《人月神话》一书，成了史上最畅销的软件工程书籍。

为了解决问题，在 1968 年、1969 年连续召开了两次著名的 NATO 会议，会议正式创造了“软件危机”一词，并提出了针对性的解决方法“软件工程”。虽然“软件工程”提出之后也曾被视为软件领域的银弹，但后来事实证明，软件工程同样无法解决软件危机。

差不多在同一时间，“结构化程序设计”作为另外一种解决软件危机的方案被提出来了。Edsger Dijkstra 于 1968 年发表了著名的《GOTO 有害论》的论文，引起了长达数年的论战，并由此产生了结构化程序设计方法。同时，第一个结构化的程序语言 Pascal 也在此时诞生，并迅速流行起来。

结构化程序设计的主要特点是抛弃 goto 语句，采取“自顶向下、逐步细化、模块化”的指导思想。结构化程序设计本质上还是一种面向过程的设计思想，但通过“自顶向下、逐步细化、模块化”的方法，将软件的复杂度控制在一定范围内，从而从整体上降低了软件开发的复杂度。结构化程序设计方法成为了 20 世纪 70 年代软件开发的潮流。

科学研究证明，人脑存在人类短期记忆，一般一次只能记住 5～9 个事物，这就是著名的 7±2 原理。结构化程序设计是面向过程设计思想的一个改进，使得软件开发更加符合人类思维的 7±2 特点。

第二次软件危机：面向对象程序设计

主要参考：http://www.parallellabs.com/2010/04/01/the-third-software-crisis/

结构化编程的风靡在一定程度上缓解了软件危机，然而好景不长，随着硬件的快速发展，业务需求越来越复杂，以及编程应用领域越来越广泛，第二次软件危机很快就到来了。

第二次软件危机的根本原因还是在于软件生产力远远跟不上硬件和业务的发展，相比第一次软件危机主要体现在“复杂性”方面，第二次软件危机主要体现在“可扩展性”、“可维护性”上面。传统的面向过程（包括结构化程序设计）方法已经越来越不能适应快速多变的业务需求，软件领域迫切希望找到新的银弹来解决软件危机，在这种背景下，面向对象的思想开始流行起来。

面向对象的思想并不是在第二次软件危机后才出现的，早在 1967 年的 Simula 语言中