

/THEORY/IN/PRACTICE

代码之美

Beautiful Code

Leading Programmers Explain How They Think

O'REILLY®



机械工业出版社
China Machine Press



Andy Oram & Greg Wilson 编

BC Group 译

代码之美

Andy Oram & Greg Wilson 编
BC Group 译

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Taipei • Tokyo

O'Reilly Media, Inc. 授权机械工业出版社出版

机械工业出版社

图书在版编目 (CIP) 数据

代码之美 / (美) 奥莱姆 (Oram, A.), (美) 维尔森 (Wilson, G.) 编; BC Group 译. —北京: 机械工业出版社, 2008.9

书名原文: Beautiful Code

ISBN 978-7-111-25133-0

I. 代… II. ①奥… ②维… ③B… III. 代码—程序设计 IV. TP311.11

中国版本图书馆 CIP 数据核字 (2008) 第 144657 号

北京市版权局著作权合同登记

图字: 01-2007-4189 号

Copyright©2007 by O'Reilly Media, Inc.

Simplified Chinese Edition, jointly published by O'Reilly Media, Inc. and China Machine Press, 2008. Authorized translation of the English edition, 2007 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

英文原版由 O'Reilly Media, Inc. 出版 2007。

简体中文版由机械工业出版社出版 2008。英文原版的翻译得到 O'Reilly Media, Inc. 的授权。此简体中文版的出版和销售得到出版权和销售权的所有者——O'Reilly Media, Inc. 的许可。

版权所有, 未得书面许可, 本书的任何部分和全部不得以任何形式重制。

本书法律顾问 北京市展达律师事务所

书 名 / 代码之美

书 号 / ISBN 978-7-111-25133-0

责任编辑 / 杨庆燕 周茂辉

封面设计 / Randy Comer, 张健

出版发行 / 机械工业出版社

地 址 / 北京市西城区百万庄大街 22 号 (邮政编码 100037)

经 销 / 新华书店北京发行所发行

印 刷 / 北京牛山世兴印刷厂

开 本 / 178 毫米 × 233 毫米 39.75 印张

版 次 / 2009 年 1 月第 1 版第 1 次印刷

定 价 / 99.00 元

凡购本书, 如有倒页、脱页、缺页, 由本社发行部调换

O'Reilly Media, Inc. 介绍

为了满足读者对网络 and 软件技术知识的迫切需求，世界著名计算机图书出版机构 O'Reilly Media, Inc. 授权机械工业出版社，翻译出版一批该公司久负盛名的英文经典技术专著。

O'Reilly Media, Inc. 是世界上在 Unix、X、Internet 和其他开放系统图书领域具有领导地位的出版公司，同时是联机出版的先锋。

从最畅销的《The Whole Internet User's Guide & Catalog》（被纽约公共图书馆评为二十世纪最重要的 50 本书之一）到 GNN（最早的 Internet 门户和商业网站），再到 WebSite（第一个桌面 PC 的 Web 服务器软件），O'Reilly Media, Inc. 一直处于 Internet 发展的最前沿。

许多书店的反馈表明，O'Reilly Media, Inc. 是最稳定的计算机图书出版商——每一本书都一版再版。与大多数计算机图书出版商相比，O'Reilly Media, Inc. 具有深厚的计算机专业背景，这使得 O'Reilly Media, Inc. 形成了一个非常不同于其他出版商的出版方针。O'Reilly Media, Inc. 所有的编辑人员以前都是程序员，或者是顶尖级的技术专家。O'Reilly Media, Inc. 还有许多固定的作者群体——他们本身是相关领域的技术专家、咨询专家，而现在编写著作，O'Reilly Media, Inc. 依靠他们及时地推出图书。因为 O'Reilly Media, Inc. 紧密地与计算机业界联系着，所以 O'Reilly Media, Inc. 知道市场上真正需要什么图书。

推荐序一

重新擦亮思考的火花

《程序员》杂志技术主编 孟岩

《代码之美》已经成为经典。关于它本身的赞美之辞已经不少了，不过到底从这本书里该读些什么，我倒是有些思考。

20世纪90年代初期，当时正在加州大学埃尔文分校攻读博士学位的Douglas Schmidt在观察了他所参与的软件项目开发实践之后，得出一个结论，即未来的软件开发将越来越多地体现为整合（integration），而不是传统意义上的编程（programming）。换言之，被称为“软件开发者”的这个人群，将越来越明显地分化：一部分人开发核心构件和基础平台，而更多地人将主要是配置和整合现有构件以满足客户的需求，类似现代汽车、机床和家用电器制造业的产业格局即将到来。面对这一前景，博士生Schmidt一方面写文章对于其进步意义大加赞扬，另一方面毫不犹豫地投入到核心构件及平台的开发阵营中去。他很清楚，在这样一种分工体系中，由于软件整合产业很难出现垄断局面，因此大多数利润总是被截留在上游，人当然要往高处走，整合化是好事，但他老兄宁可让别人来推动这个好事。

事实上，软件产业中大多数预测都被历史的发展无情地抛到垃圾堆里了，然而Schmidt博士生的这个预测却惊人的准确，其后十几年软件的发展完美地印证了他当年的判断。因此，他本人基于这一预测所选择的人生道路也一帆风顺。如今已经是教授的Douglas

Schmidt 先后创造了 ACE、TAO、CIAO 等一系列分布式计算基础件，先后主导了美国学界和国防领域内若干重大科研与实际开发项目，成为世人公认的分布式计算架构领导者。

抛开他个人的辉煌不说，“整合化”趋势实际上已经深刻地改变了世界软件工业的面貌，从而也影响了身为晚进者的我们的命运。如今大部分的程序员实际上是在整合与配置现有资源以满足需求，而不是真正意义上的“编程”。这当然是一件好事，整合同样需要深刻的洞察力和创新精神，优秀的整合者与天才的程序员一样不可多得，甚至更加罕见。然而我们也不能不承认，大多数整合性的工作是机械的，简单的，重复的，欠缺创意的，深入的思考往往不必要。因此，在这个整合为王的时代里，思考的精神在钝化。更有甚者，互联网和搜索引擎的出现大大加速了这种钝化，几乎所有的问题都有人解决并且张贴在互联网上了，因此独自思考和解决问题已经成了不必要的、降低效率的行为，不但不时髦，而且不经济。软件开发迅速成为一个强调搜索和短期记忆力的技能，我想这是 50 年前第一代程序员们做梦也没有想到的。

老实讲，就整体而言，我仍然认为这是一种进步。任何一个产业的成熟，无不伴随着分工的明晰、技能的简化和从业门槛的降低。与少数人享受思考乐趣的需求相比，大多数人享受便宜而无处不在的软件服务的需求显然远为重要。但是，对于身处软件行业中的个体来说，思考力的削弱和丧失却是不折不扣的悲剧。这一点不必过多解释，几年来苦苦寻找自己核心竞争力的开发者们都知道我说的是什么意思。长期以来对中国开发者社群的近距离观察使我确信，尽管作为一个产业，中国软件一直享受着比较快的成长，但是总体而言，中国的软件开发者越来越迷惘、焦躁和不自信。这一情况当然是由多种原因导致的，但开发者们每念及此，多抱怨体制、产业、市场等身外之物，实在也有失偏颇。平心而论，这几年中国软件技术界的生存环境还是有了很大改善，对于那些真正出类拔萃的程序员来说，过上一种充实自信的生活并不困难。摆在每一个个体面前的主要问题还是在于能否出类拔萃，而这就需要重新找回思考的能力。具备强悍思考能力的人，也就具备强悍的解决问题的能力，而这样的开发者永远都是产业中的稀缺资源。

我认为这正是《代码之美》这本书的一个重要价值。合作的诸位大师级作者，给我们一个很好的机会，让我们能够一边阅读，一边思考，找回深思熟虑的智慧火花。这本书里所讲的每一个问题，可以说都是程序员在工作中会遇到或者至少会擦边的问题，既没有故弄玄虚的文字游戏，也没有携带了领域知识的私货，只有朴实而实际的一个个问题。虽然不是以提问的方式给出，但在整个阅读的过程中，我们还是能够找到很多机会与大师互动，不断地发现问题和解决问题。我在阅读中经常感到，看上去一个很简单的问题，却被这些大师们一层一层挖掘得如此深入，到最后阶段不由得令人感到战栗和震撼。看着这些智慧的光芒，我们不但可以领略大师之所以称为大师的秘密，而且也认识到思考

的真谛。因此，千万不要像看小说一样一带而过，那样会错过本书95%的价值！我们不是要阅读这些文字，而是要与文字背后的大师交流学习，一点一点把自己的心得记下来，对于作者提出的新问题，先自己思考，直接写程序尝试，争取跟上大师的思路，甚至可能需要反复几遍，才能真正读通这本书。这样的精力不会是白费的，读者应当认识到，当我们拥有这本书的时候，我们获得了怎样宝贵的机会，可以在相对比较短的时间里有效地提升自己的思考能力。这是一个机会，也是一次考验，我绝对相信，通过这次考验的读者，会在思考和解决问题的能力上有一个大的进步。

我希望自己能够以这样的态度读这本了不起的书，以此文与其他读者朋友共勉之。

推荐序二

编程的艺术

CSDN 副总经理

《梦断代码》译者

韩磊

学术界有一种叫“论文集”的东西，她把许多人的论文整合到一起出版，让读者能够在一本书的篇幅之内，了解某个特定领域的研究状况，是有效的知识传播手段之一。技术界，类似的出版物却是凤毛麟角，的确是一种遗憾！

《代码之美》就是这样一本书。她将 38 位大牛人的技术文章汇集到一起，讲述作者们认为“最漂亮的代码”，其涉及应用领域虽广，而代码之美却一以贯之。如果我们承认编程是一门艺术——具有高度创造性和人类智慧参与的活动，不是艺术是什么？——那么，这 33 篇文章恰恰体现了这门艺术的最高境界。

别担心！大牛们可不是坐而论道，也没有写什么常人不可索解的奥义，文章主题之朴实无华，比如“查找”，比如“分布式编程”，比如“Linux 内核驱动模型”……几乎要让人们以为是不知道什么人编写的大学教材呢。这貌似普通的三个主题，作者分别是 XML 创始人之一 Tim Bray、Google Map/Reduce 架构发明人 Jeff Dean 和 Linux 内核维护者 Greg Kroah-Hartman ——吓死人的阵容。其余文章也都类似，小题目中见大手笔。

我深信这帮大牛接受约稿、写这种“小”文章，的确是出于对编程的热爱，出于对“漂亮代码”的不懈追求。所谓“漂亮代码”，意思远超“规范、好看”，更多地体现出逻辑、

推荐序三

享受代码之美

Thought Works 中国咨询师

《重构》译者

熊节

“希望写出漂亮代码的开发者可以向艺术家们学习一些东西。画家常常放下手中的画笔，然后远离画布一段距离，围着它转一转，翘起脑袋，斜着看看，再从不同的角度看看，在不同的光线下看看。在寻求美的过程中，他们需要设计这样一些视角并使它们融为一体。如果你的画布是个集成开发环境 (IDE)，而你的媒介就是代码，想一想，你如何做到离开画布一段距离，用挑剔的眼光从不同的视角来审视你的作品？——这将使你成为一个更优秀的程序员，并帮你写出漂亮的代码。”

写这段话的 Alberto Savoia 在他的文章里真的没有讲什么令人敬畏的高新技术或是大架构，他讲的是每个计算机系的大二学生都熟悉的二分查找。所以 Savoia 真的只是在讲如何写出漂亮的代码，所以才选择了这么一个所有人都清楚得不能再清楚的例子。你会觉得这种事情都是些不谙世事的小程序员才会热衷于干的吧？可这位 Savoia 却是从 Google 离职以后开创了 Agitar Software 公司 (<http://www.agitar.com/>) 的不折不扣的创业者。有意思吗？一个胡须花白、在这个行业里厮混了数十年、拥有自己公司的老者，还在乐此不疲地谈论“漂亮的代码”。

《代码之美》就是由 33 篇像这样有意思的文章组成的。像 Brian Kernighan、Tim Bray、Charles Petzold、Douglas Schmidt、Yukihiro Matsumoto 这样的名字，你甚至很难想

象他们会同时出现在同一本书上。或许也只有“漂亮的代码”这样的话题才能激起他们共同的兴趣。于是就有了这本了不起的书：从正则表达式匹配器到图像处理，从通信到基因排序，这些世界上最优秀的程序员毫不吝啬地向读者展示：不论面对什么问题、使用什么语言，代码的美感都是始终存在的，而且这种美感应该是程序员毕其一生不懈追寻的。

作为《重构》的译者，不时有人会问我一些关于重构的问题，其中一个问题让我最为难：为什么要这样做？真的，如果不是要修改代码，也不是要添加功能，为什么要把这段代码抽取出来呢？让每个方法都保持5行以内的长度到底有什么好处呢？这种时候与其说是有什么利弊权衡，毋宁说就是为了让代码“更漂亮”。当然了，在大部分时间里，软件开发是一项集合了科学、工程和服务的工作，但至少在我们的内心深处——它多少还有那么一点艺术的成分。除了完成任务以外让自己手上的代码更具美感，也算是对自己作为程序员梦想的小小坚持吧。

所以，既然你已经拿起了这本书，就暂时放开那些功利的目标吧——别误会，这不是一本没用的书，通过阅读这些“大师”们的编程心得，对自己的能力提升就算不能立竿见影至少也有潜移默化之效。但那也只是装珍珠的盒子而已。在一个安静的周末，给自己泡上一杯清茶，跟着38位顶尖高手畅游在代码世界，在他们的指引下赏遍代码之美，这才是作为一个程序员最大的享受呢。

推荐序四

不仅仅是代码

InfoQ 中文站总编辑

<http://www.infoq.com/cn>

霍泰稳

在翻阅完机械工业出版社华章分社快递给我的厚厚初稿后，我不由自主地赞叹“不仅仅是代码”。掩卷沉思，深感本书带给人的震撼远非一个序言所能概括。让我们先来简单看几个这本震撼之作的震撼作者们：

Tim Bray, 1995 年启动了最早的公共网页搜索引擎之一，1996 至 1999 年间与他人共同发明了 XML 1.0；

Bryan Cantrill, 《华尔街日报》2006 年度最高创新奖获奖产品 DTrace 的作者之一；

Douglas Crockford, JSON 数据格式的发明者；

Jeff Dean, Google 网页抓取、索引、查询服务以及广告系统的主力开发者；

Yvkihiro “Matz”, Ruby 语言的发明者；

Sanjay Ghemawat, Google Fellow, 设计并实现了 Google 的分布式存储系统、文本索引系统及性能分析工具等；

.....

太多了，一句话，都是牛人。想一想，在你的编程生涯中有那么多牛人给你传道授业解惑，那是一种什么样的感觉？是不是在为没有早些看到此书而遗憾，又或为今日能读到此书而庆幸？无论如何，我的感觉是这样的。

但如果你认为此书仅是对代码片段的解剖，并分析哪个算法最美而已，那可是大大低估了这本书的价值，同时也误解了编辑的初衷。在 Ronald Mak 介绍 NASA 火星漫步者任务中的高可靠企业系统时，他总结说，与小型程序不同的是，大型应用程序的漂亮性并不一定只存在于优美的算法中。对于 NASA 的协同信息系统 CIP 来说，漂亮性在于它的面向服务架构实现以及大量简单却经过仔细挑选的组件。而本书的策划编辑 Greg Wilson 在整理书稿的过程中，也体味到在不同的地方我们可以看到代码不同的漂亮性，有些漂亮性存在于手工精心打造软件的细微之处，这儿可以理解为代码段，而另外有些漂亮性则蕴藏在大局之中——那些使程序能够持续发展的架构，或者用来构造程序的技术。也就是说，本书不仅剖析了如何撰写美丽的代码片段，还告诉你如何设计美丽的架构。

那么除了美丽的代码、算法以及架构外，还有其他地方能体现代码的漂亮性吗？现在人们对软件开发行业的关注或者尊崇已经大不如从前，程序员也从以前的高薪一族沦为“软件蓝领”或者“IT 民工”，伴随而来更严重的变化是编程的神圣感也在逐步缺失。当一份工作在我们的眼中只是一个糊口的工具时，你很难投之于激情和梦想，更谈不上最后会取得什么成就。让我们看一下大名鼎鼎的互联网隐私服务 Neomailbox 的首席开发人员 Ashish Gulhati 对“程序员”是如何定义的吧：作为代码的编写者，在很大程度上是程序员的责任感使得我们编写的代码不仅在设计和实现上是漂亮的，同时还使代码所带来的结果在社会的环境中是漂亮的。这就是我认为自由的代码非常优美的原因。它把计算机技术置于一个最庄严神圣的用途：保护人权和人的生命。

我无意成为一个高尚道德的说教者，只是在本书阅读过程中，逐渐认识到：会编写优秀的代码，会设计优秀的架构，有敢于担当的社会责任心，是一件多么令人骄傲和让人尊敬的事情。

推荐序五

等度的流明

知名技术专家

《大道至简》作者

周爱民

我上一次印象深刻的美的体验，大概已经是在十年之前了，那只是在午后睡醒，面对窗外的一棵大梧桐树时的感觉。不过这并不是说我这十年来都只看到了丑的事物，而是说我已经忘了去观察既已存在的美。

直到我拿到这本《代码之美》，我忽然地回到了那种仰望着星光闪烁的夜空，或低头沉思于一两句大家文字的日子里。那时我既不是在思考，也不是在分析，更不是在解释，而只是在感受自然的、文字的，或将自然蕴于文字之中的美。

有一本书开启了一个时代，我们如今仍然在这个时代之中而不知觉于这本书的深远影响，那是三位图灵奖得主合著的《结构程序设计》（注1）。其中 Dijkstra 将人“理解一个程序的种种思维方法”归为三种：枚举、数学归纳和抽象。

注1：《结构程序设计》出版于1972年，作者分别是1972、1980、2001年图灵奖得主 Edsger W.Dijkstra、C.Anthony R.hoare 和 Dahl。

显然 Wirth 先生更为深层地看到了程序的本质，他说“程序=算法+数据结构”（注2）。他揭示了这样一个事实：一个未知的、无序的世界是不可能实现“程序”的，于是我们抽象它——使它成为结构，或者对象，或者网，或者某个相对规则的事物。然后，我们再着之以“算法”。

《代码之美》这本书，38位大师，在33章的内容中详细讨论了代码中抽象的过程、算法的过程和编程的过程。显然地，这些正是程序中最深刻的美，如同花之蕊，叶之脉，以及维系花蕊叶脉的美的那些汁液。这种对美的触及，使在我面前闪耀着与前两本书等度的流明。

三

“只有在不仅没有任何功能可以添加，而且也没有任何功能可以删除的情况下，设计师才能够认为自己的工作已臻完美。”（注3）然而编程的过程呢？我们最初只是想实现一个功能。但为了实现它，我们写了一段功能代码、一段测试代码、一段功能代码的配置代码，一段功能代码的配置代码的测试代码……

如此往复不休。

我们回到原始的问题，原本只是要做一个“实现某项功能”的代码，我们却为何把代码做到了“往复不休”的绝境？

或者你做的事情并不完美，但是你应该知道所谓完美的终极。代码要不停的测试，以及为测试代码再写测试代码，这一过程也不是美的。或许你认为它“必须”，但你应知道它终究不美。

四

大师们也并没有创造完美的能力，他们只是在一步步地进行着。在这本书里，Adam Kolawa告诉你的，Lincoln Stein告诉你的，以及Elliote Rusty Harold等告诉你的，就是那经年累月地或亦步亦趋地进行过程，和那个“终极完美”的定义。

这只是过程和隐于过程中对美的追求。而“美”是什么，还是在你的心底。你心中原本就没有美的感受，如何写得出美的代码？所以代码写到烂处，写到心胸滞涩处，便不如

注2： 《算法+数据结构=程序》出版于1975年，作者Pascal之父Niklaus Wirth，是1984年图灵奖得主。

注3： 出自Antoine de SaintExupery，法国战士与文学家。本书第3章中Jon Bentley引用。

寻一清静所在，捧《代码之美》一册，回顾一下，品味一下，吐故纳新一下了。

看得多远，取决于你站得多高；要想成为他们，你得先知道他们。

这就是我的建议了。

代码之美

陈昊

www.umi-china.com

陈昊

陈昊，1989年毕业于清华大学，现为清华大学计算机系教授，长期从事计算机系统架构、操作系统、数据库、网络、云计算、大数据、人工智能等领域的研究和教学工作。著有《代码之美》、《代码之美》、《代码之美》等书籍。

陈昊，1989年毕业于清华大学，现为清华大学计算机系教授，长期从事计算机系统架构、操作系统、数据库、网络、云计算、大数据、人工智能等领域的研究和教学工作。著有《代码之美》、《代码之美》、《代码之美》等书籍。

陈昊，1989年毕业于清华大学，现为清华大学计算机系教授，长期从事计算机系统架构、操作系统、数据库、网络、云计算、大数据、人工智能等领域的研究和教学工作。著有《代码之美》、《代码之美》、《代码之美》等书籍。

陈昊，1989年毕业于清华大学，现为清华大学计算机系教授，长期从事计算机系统架构、操作系统、数据库、网络、云计算、大数据、人工智能等领域的研究和教学工作。著有《代码之美》、《代码之美》、《代码之美》等书籍。

推荐序六

向大师学习美

UMLChina 首席专家

WWW.UMLChina.com

潘加宇

我1989年参加高考时，总分120分的语文才考了71分，这最弱一环差点造成致命打击。当时，我最害怕的是写作文，记得当年的题目是写一封信，在右江盆地高温的教室里，我写得汗都滴在试卷上。没想到，后来看的東西多了，“不会作诗也会吟”了，而且居然也能接到杂志和出版社的约稿，写一些文章和序言之类。

翻开本书第29章，Ruby之父Yukihiro Matsumoto（松本行弘）说：Treating Code As an Essay。写代码如同写文章一般，多看多研究大师的作品，才能够信手拈来，写出美丽的代码。本书就是33位大师的倾情之作。

本书并不限于讨论某一种语言的技巧，你能看到大师们使用现在流行的Java、Ruby，也能看到历史悠久的Fortran。讨论的领域从Linux内核，到NASA火星探测器、ERP系统，让我们从不同角度来体会代码之美。

美除了让人欣赏，还能带来金钱。因北京奥运的成功，博尔特的速度之美估计价值上千万美元。美丽的软件也一样。一些软件公司，公司人少且稳定，多年来专注于做某一个小领域里的软件，对软件的打磨可谓是精雕细琢，美丽软件带来的利润自然也很可观。比起那些靠低人力成本、低价格在市场上打拼的“程序员民工”公司，他们要活得滋润得多，安全得多。