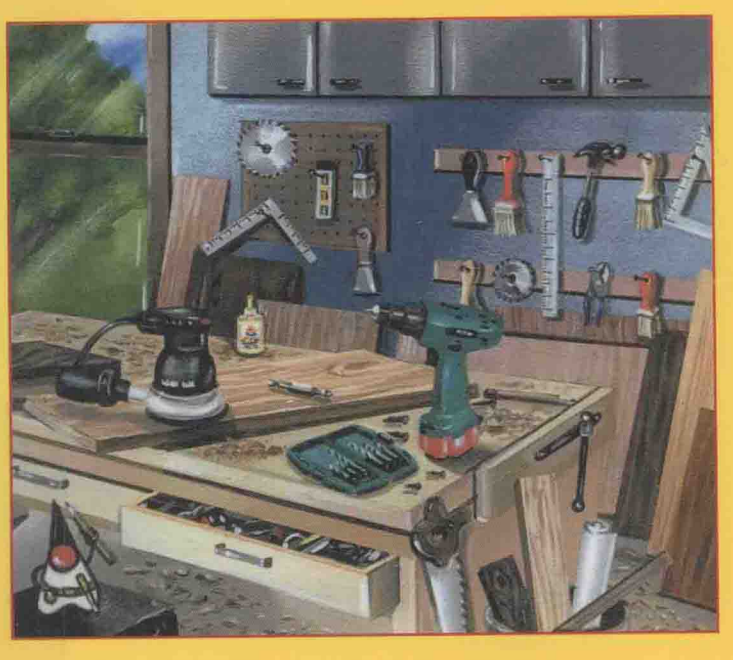
HZ BOOKS

Sun公司核心技术丛书

Effective Java中文版第2版

Effective Java Second Edition



（美）Joshua Bloch 著
杨春花 俞黎敏 译

“我很希望10年前就拥有这本书。可能有人认为我不需要任何Java方面的书籍，但是我需要这本书。”

Java之父James Gosling

Sun microsystems

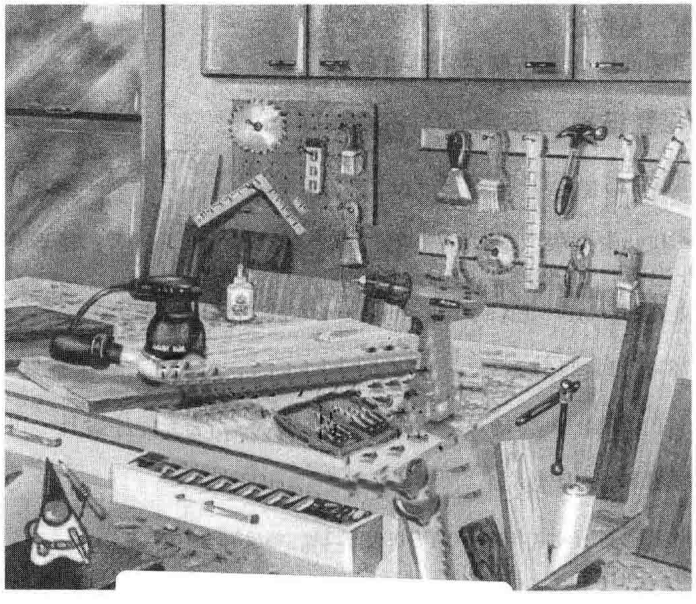
机械工业出版社
China Machine Press

Sun 公司核心技术丛书

Effective Java中文版第2版

Effective Java Second Edition

（美）Joshua Bloch 著
杨春花 俞黎敏 译



机械工业出版社
China Machine Press

本书介绍了在Java编程中78条极具实用价值的经验规则，这些经验规则涵盖了大多数开发人员每天所面临的问题的解决方案。通过对Java平台设计专家所使用的技术的全面描述，揭示了应该做什么，不应该做什么才能产生清晰、健壮和高效的代码。第2版反映了Java 5中最重要的变化，并删去了过时的内容。

本书中的每条规则都以简短、独立的小文章形式出现，并通过示例代码加以进一步说明。本书内容全面，结构清晰，讲解详细。可作为技术人员的参考用书。

本书版权登记号：图字：01-2008-2445

图书在版编目（CIP）数据

Effective Java中文版　第2版 /（美）布洛克（Bloch, J.）著；杨春花，俞黎敏译. —北京：机械工业出版社，2009.1（2013.12重印）

书名原文：Effective Java Program Language Guide, 2E

ISBN 978-7-111-25583-3

Ⅰ. E…　Ⅱ①布…　②杨…　③俞…　Ⅲ. JAVA语言－程序设计　Ⅳ. TP312

中国版本图书馆CIP数据核字（2008）第178021号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）

责任编辑：陈佳媛

北京市荣盛彩色印刷有限公司印刷

2013年12月第2版第12次印刷

186mm × 240mm · 19印张

标准书号：ISBN 978-7-111-25583-3

定价：52.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

译者序

Java从诞生到日趋完善，经过了不断的发展壮大，目前全世界拥有了成千上万的Java开发人员。如何编写出更清晰、更正确、更健壮且更易于重用的代码，是大家所追求的目标之一。作为经典Jolt获奖作品的新版书，它已经进行了彻底的更新，涵盖了自第1版之后所引入的Java SE 5和Java SE 6的新特性。作者探索了新的设计模式和语言习惯用法，介绍了如何充分利用从泛型到枚举、从注解到自动装箱的各种特性。本书的作者Joshua Bloch曾经是Sun公司的杰出工程师，带领团队设计和实现过无数的Java平台特性，包括JDK 5.0语言增强版和获奖的Java Collections Framework。他也是Jolt奖的获得者，现在担任Google公司的首席Java架构师。他为我们带来了共78条程序员必备的经验法则：针对你每天都会遇到的编程问题提出了最有效、最实用的解决方案。

书中的每一章都包含几个“条目”，以简洁的形式呈现，自成独立的短文，它们提出了具体的建议、对于Java平台精妙之处的独到见解，并提供优秀的代码范例。每个条目的综合描述和解释都阐明了应该怎么做、不应该怎么做，以及为什么。通过贯穿全书透彻的技术剖析与完整的示例代码，仔细研读并加以理解与实践，必定会从中受益匪浅。书中介绍的示例代码清晰易懂，也可以作为日常工作的参考指南。

适合人群

本书不是针对初学者的，读者至少需要熟悉Java程序设计语言。如果你连equals()、toString()、hashCode()都还不了解的话，建议先去看些优秀的Java入门书籍之后再来阅读本书。如果你现在已经在Java开发方面有了一定的经验，而且想更加深入地了解Java编程语言，成为一名更优秀、更高效的Java开发人员，那么，建议你用心地研读本书。

内容形式

本书分为11章共78个条目，涵盖了Java 5.0 / 6.0的种种技术要点。与第1版相比，本书删除了“C语言结构的替代”一章，增加了Java 5所引入的“泛型”、“枚举和注解”各一章。数量上从57个条目发展到了78个，不仅增加了23个条目，并对原来的所有资料都进行了全面的修改，删去了一些已经过时的条目。但是，各章节没有严格的前后顺序关系，你可以随意选

择感兴趣的章节进行阅读。当然，如果你想马上知道第2版究竟有哪些变化，可以参阅附录中第2版与第1版详细的对照情况。

本书重点讲述了Java 5所引入的全新的泛型、枚举、注解、自动装箱、for-each循环、可变参数、并发机制，还包括对象、类、类库、方法和序列化这些经典主题的全新技术和最佳实践，如何避免Java编程语言中常被误解的细微之处：陷阱和缺陷，并重点关注Java语言本身和最基本的类库：java.lang、java.util，以及一些扩展：java.util.concurrent和java.io等等。

章节简介

第2章阐述何时以及如何创建对象，何时以及如何避免创建对象，如何确保它们能够被适时地销毁，以及如何管理销毁之前必须进行的所有清除动作。

第3章阐述对于所有对象都通用的方法，你会从中获知对equals、hashCode、toString、clone和finalize相当深入的分析，从而避免今后在这些问题上再次犯错。

第4章阐述作为Java程序设计语言的核心以及Java语言的基本抽象单元（类和接口），在使用上的一些指导原则，帮助你更好地利用这些元素，设计出更加有用、健壮和灵活的类和接口。

第5和第6章中分别阐述在Java 1.5发行版本中新增加的泛型（Generic）以及枚举和注解的最佳实践，教你如何最大限度地享有这些优势，又能使整个过程尽可能地简单化。

第7章讨论方法设计的几个方面：如何处理参数和返回值，如何设计方法签名，如何为方法编写文档。从而在可用性、健壮性和灵活性上有进一步的提升。

第8章主要讨论Java语言的具体细节，讨论了局部变量的处理、控制结构、类库的使用、各种数据类型的用法，以及两种不是由语言本身提供的机制（reflection和native method，反射机制和本地方法）的用法。并讨论了优化和命名惯例。

第9章阐述如何充分发挥异常的优点，可以提高程序的可读性、可靠性和可维护性，以及减少使用不当所带来的负面影响。并提供了一些关于有效使用异常的指导原则。

第10章阐述如何帮助你编写出清晰、正确、文档组织良好的并发程序。

第11章阐述序列化方面的技术，并且有一项值得特别提及的特性，就是序列化代理（serialization proxy）模式，它可以帮助你避免对象序列化的许多缺陷。

举个例子，就序列化技术来讲，HTTP会话状态为什么可以被缓存？RMI的异常为什么可以从服务器端传递到客户端呢？GUI组件为什么可以被发送、保存和恢复呢？是因为它们实现了Serializable接口吗？如果超类没有提供一个可访问的无参构造器，它的子类可以被序列化

吗？当一个实例采用默认的序列化形式，并且给某些域标记为transient，那么当实例反序列化回来后，这些标志为transient域的值各是些什么呢？……这些问题如果你现在不能马上回答，或者不能很确定，没有关系，仔细阅读本书，你会对它们有更深入与透彻的理解。

技术范围

虽然本书是讨论更深层次的Java开发技术，讲述的内容深入，涉及面又相当广泛，但是它并没有涉及图形用户界面编程、企业级API以及移动设备方面的技术，不过在各个章节与条目中会不时地讨论到其他相关的类库。

这是一本分享经验与指引你避免走弯路的经典著作，针对如何编写高效、设计优良的程序提出了最实用、最权威的指导方针，是Java开发人员案头上的一本不可或缺的参考书。

本书由我组织进行翻译，第1章到第8章由杨春花负责，我负责前言、附录以及第9章到第11章的翻译，并负责本书所有章节的全面审校。参与翻译和审校的还有：荣浩、邱庆举、万国辉、陆志平、姜法有、王琳、林仪明、凌家亮、李勇、师文丽、刘传飞、王建旭、程旭文、罗兴、翟育明、黄华，在此深表感谢。

虽然我们在翻译过程中竭力追求信、达、雅，但限于自身水平，也许仍有不足，还望各位读者不吝指正。关于本书的翻译和翻译时采用的术语表以及相关的技术讨论大家可以访问我的博客http://blog.csdn.net/YuLimin，也可以发邮件到YuLimin @ 163.com与我交流。

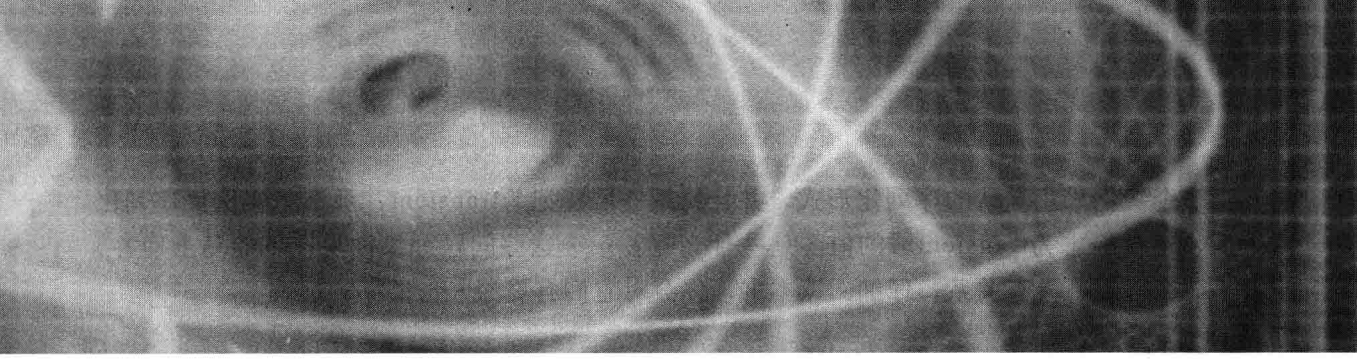
在这里，我要感谢在翻译过程中一起讨论并帮助我的朋友们，他们是：崔毅，郑晖，左轻侯，郭晓刚，满江红开放技术研究组织创始人曹晓钢，Spring中文站创始人杨戈（Yanger），SpringSide创始人肖桦（江南白衣）和来自宝岛台湾的李日贵（jini）、林康司（koji）、林信良（caterpillar），还有责任编辑陈佳媛也为本书出版做了大量工作，在此再次深表感谢。

快乐分享，实践出真知，最后，祝大家能够像我一样在阅读中享受本书带来的乐趣！

Read a bit and take it out, then come back read some more.

俞黎敏

2008年11月



序

如果有一个同事这样对你说，“我的配偶今天晚上在家里制造了一顿不同寻常的晚餐，你愿意来参加吗？”（Spouse of me this night today manufactures the unusual meal in a home. You will join?）这时候你脑子里可能会浮现起三件事情：第一，满脑子的疑惑；第二，英语肯定不是这位同事的母语；第三，同事是在邀请你参加他的家庭晚宴。

如果你曾经学习过第二种语言，并且尝试过在课堂之外使用这种语言，你就该知道有三件事情是必须掌握的：这门语言的结构如何（语法），如何命名你想谈论的事物（词汇），以及如何以惯用和高效的方式来表达日常的事物（用法）。在课堂上大多只涉及前面两点，当你使出浑身解数想让对方明白你的意思时，常常会发现当地人对你的表述忍俊不禁。

程序设计语言也是如此。你需要理解语言的核心：它是面向算法的，还是面向函数的，或者是面向对象的？你需要知道词汇表：标准类库提供了哪些数据结构、操作和功能（Facility）？你还需要熟悉如何用习惯和高效的方式来构建代码。关于程序设计语言的书籍通常只是涉及前面两点，或者只是蜻蜓点水般地介绍一下用法。也许是因为前面两点比较容易编写。语法和词汇是语言本身固有的特性，但是，用法则反映了使用这门语言的群体的特征。

例如，Java程序设计语言是一门支持单继承的面向对象程序设计语言，在每个方法的内部，它也支持命令式的（面向语句的，Statement-Oriented）编码风格。Java类库提供了对图形显示、网络、分布式计算和安全性的支持。但是，如何把这门语言以最佳的方式运用到实践中呢？

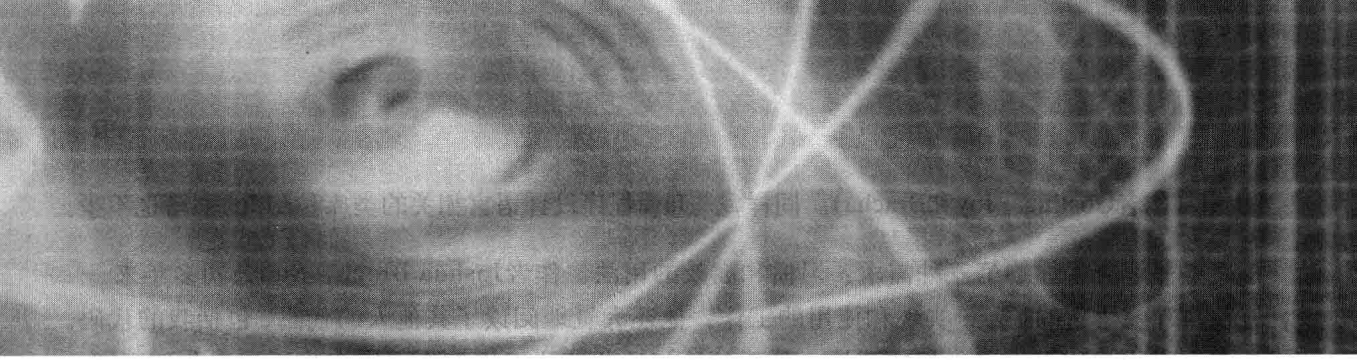
还有一点：程序与口语中的句子以及大多数书籍和杂志都不同，它会随着时间的推移而发生变化。仅仅编写出能够有效地工作并且能够被别人理解的代码往往是不够的，我们还必须把代码组织成易于修改的形式。针对某个任务可能会有10种不同的编码方法，而在这10种方法中，有7种方法是笨拙的、低效的或者是难以理解的。而在剩下的3种编码方法中，哪一种会是最接近该任务的下一年度发行版本的代码呢？

目前有大量的书籍可以供你学习Java程序设计语言的语法，包括《The Java Programming Language》[Arnold05]（作者Arnold、Gosling和Holmes），以及《The Java Language Specification》

[JLS]（作者Gosling、Joy和Bracha）。同样，与Java程序设计语言相关的类库和API的书籍也不少。

本书解决了你的第三种需求：习惯和高效的用法。作者Joshua Bloch在Sun公司多年来一直从事Java语言的扩展、实现和使用的工作；他还大量地阅读了其他人的代码，包括我的代码。他在本书中提出了许多很好的建议，他系统地把这些建议组织起来，旨在告诉读者如何更好地构造代码以便它们能工作得更好，也便于其他人能够理解这些代码，便于将来对代码进行修改和改善的时候不至于那么头疼。甚至，你的程序也会因此而变得更加令人愉悦、更加优美和雅致。

Guy L. Steele Jr.
Burlington, Massachusetts
2001年4月



前　言

自从我于2001年写了本书的第1版之后，Java平台又发生了很多变化，是该出第2版的时候了。Java 5中最为重要的变化是增加了泛型、枚举类型、注解、自动装箱和for-each循环。其次是增加了新的并发类库：java.util.concurrent。我和Gilad Bracha一起，有幸带领团队设计了最新的语言特性。我还有幸参加了设计和开发并发类库的团队，这个团队由Doug Lea领导。

Java平台中另一个大的变化在于广泛采用了现代的IDE（Integrated Development Environment），例如Eclipse、IntelliJ IDEA和NetBeans，以及静态分析工具的IDE，如FindBugs。虽然我还未参与到这部分工作，但已经从中受益匪浅，并且很清楚它们对Java开发体验所带来的影响。

2004年，我离开Sun公司到了Google公司工作，但在过去的4年中，我仍然继续参与Java平台的开发，在Google公司和JCP（Java Community Process）的大力帮助下，继续并发和集合API的开发。我还有幸利用Java平台去开发供Google内部使用的类库。现在我了解了作为一名用户的感受。

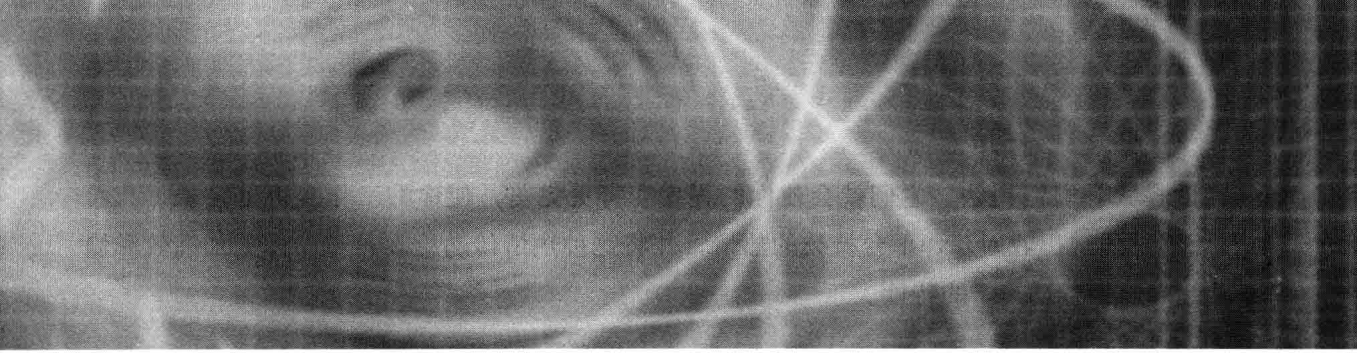
我在2001年编写第1版的时候，主要目的是与读者分享我的经验，便于让大家能够避免我所走过的弯路，使大家更容易成功。新版仍然大量采用来自Java平台类库的真实范例。

第1版所带来的反应远远超出了我最大的预期。我在收集所有新的资料以使本书保持最新时，尽可能地保持了资料的真实。毫无疑问，本书的篇幅肯定会增加，从57个条目发展到了78个。我不仅增加了23个条目，并且修改了原来的所有资料，并删去了一些已经过时的条目。在附录中，你可以看到本书中的内容与第1版的内容的对照情况。

在第1版的前言中我说过：Java程序设计语言和它的类库非常有益于代码质量和效率的提高，并且使得用Java进行编码成为一种乐趣。Java 5和6发行版本中的变化是好事，也使得Java平台日趋完善。现在这个平台比2001年的要大得多，也复杂得多，但是一旦掌握了使用

新特性的模式和习惯用法，它们就会使你的程序变得更完美，使你的工作变得更轻松。我希望第2版能够体现出我对Java平台持续的热情，并将这种热情传递给你，帮助你更加高效和愉快地使用Java平台及其新的特性。

Joshua Bloch
San Jose, California
2008年4月



致　谢

我要感谢本书第1版的读者给予本书如此热情的好评，感谢他们将书中的理念铭记在心，感谢他们让我知道该书给他们以及他们的工作带来了怎样积极的影响。我感谢许多教授在教学中采用了本书，感谢许多开发团队应用了本书。

我要感谢Addison-Wesley的整个团队，感谢他们的诚恳、专业、耐心，以及压力之下所体现出来的从容。编辑Greg Doench自始至终保持镇定自若：他是一名优秀的编辑，同时也是一位完美的绅士。产品经理Julie Nahil具备了产品经理应该具备的一切：勤奋、敏捷、训练有素，且待人和气。编审Barbara Wood一丝不苟，富有鉴赏能力。

我有幸再一次得到了所能想到的最佳审核团队的支持，我真诚地感谢他们中的每一位。核心团队负责审核每一个章节，他们包括：Lexi Baugher、Cindy Bloch、Beth Bottos、Joe Bowbeer、Brian Goetz、Tim Halloran、Brian Kernighan、Rob Konigsberg、Tim Peierls、Bill Pugh、Yoshiki Shibata、Peter Stout、Peter Weinberger以及Frank Yellin。其他审核人员包括：Pablo Bellver、Dan Bloch、Dan Bornstein、Kevin Bourrillion、Martin Buchholz、Joe Darcy、Neal Gafter、Laurence Gonsalves、Aaron Greenhouse、Barry Hayes、Peter Jones、Angelika Langer、Doug Lea、Bob Lee、Jeremy Manson、Tom May、Mike McCloskey、Andriy Tereshchenko以及Paul Tyma。这些审核人员再次提出了大量的建议，使本书得到了极大的改善，也让我避免了诸多尴尬。剩下的任何错误都是我自己的责任。

我要特别感谢Doug Lea和Tim Peierls，他们成了书中许多理念的倡导者。Doug和Tim为本书毫不吝惜地奉献了他们的时间和学识。

我要感谢我在Google公司的经理Prabha Krishna，感谢她持续不断的支持和鼓励。

最后，我要感谢我的妻子Cindy Bloch，她鼓励我写作，阅读了初稿中的每个条目，用Framemaker帮我排版，为我编写索引，在我写作的时候一直对我十分宽容。

目　录

第1章

引　言

本书的目标是帮助读者最有效地使用Java程序设计语言及其基本类库：java.lang、java.util，在某种程度上还包括java.util.concurrent和java.io。本书也会不时地讨论到其他的类库，但是没有涉及图形用户界面编程、企业级API以及移动设备相关的类库。

本书共包含78个条目，每个条目讨论一条规则。这些规则反映了最有经验的优秀程序员在实践中常用的一些有益做法。本书以一种比较自由的方式将这些条目组织成10章，每一章都涉及软件设计的一个主要方面。本书并不一定要按部就班地从头读到尾，因为每个条目都有一定程度的独立性。这些条目相互之间交叉引用，因此你可以很容易地在书中找到自己需要的内容。

Java 5（发行版本1.5）中增加了许多新特性。本书中大多数条目都以一定的方式用到了这些特性。表1-1列出了这些特性所在的主要章节或条目。

表1-1　新增特性所在章节或条目

特　性	所在章节或条目	特　性	所在章节或条目
泛型	第5章	自动装箱	第40、49条
枚举	第30～34条	varargs	第42条
注解	第35～37条	静态导入	第19条
for-each循环	第46条	java.util.concurrent	第68、69条

大多数条目都通过程序示例进行说明。本书一个突出的特点是，包含了许多代码示例，这些例子说明了许多设计模式（Design Pattern）和习惯用法（Idiom）。当需要参考设计模式领域的标准参考书[Gamma 95]时，还为这些设计模式和习惯用法提供了交叉引用。

许多条目都包含有一个或多个应该在实践中避免的程序示例。像这样的例子，有时候也叫做“反模式（Antipattern）”，在注释中清楚地标注为“//Never do this!”。对于每种情况，条目中都解释了为什么此例不好，并提出了另外的解决方法。

本书并不是针对初学者的：本书假设读者已经熟悉Java程序设计语言。如果你还没有做到，请考虑先参阅一本很好的Java入门书籍[Arnold05, Sestoft05]。本书的目标是适用于任何具有实际Java工作经验的程序员，对于高级程序员，也应该能够提供一些发人深思的东西。

本书中大多数规则都源于少数几条基本的原则。清晰性和简洁性最为重要：模块的用户永远也不应该被模块的行为所迷惑（那样就不清晰了）；模块要尽可能小，但又不能太小［本书中使用的术语模块（Module），是指任何可重用的软件组件，从单个方法，到包含多个包的复杂系统，都可以是一个模块］。代码应该被重用，而不是被拷贝。模块之间的依赖性应该尽可能地降到最小。错误应该尽早被检测出来，最好是在编译时刻。

虽然本书中的规则不会百分之百地适用于任何时刻和任何场合，但是，它们确实体现了绝大多数情况下的最佳程序设计实践。你不应该盲目地遵从这些规则，但是，你应该只在偶尔的情况下，有了充分理由之后才去打破这些规则。同大多数学科一样，学习编程艺术首先要学会基本的规则，然后才能知道什么时候可以打破这些规则。

本书大部分内容都不是讨论性能的，而是关心如何编写出清晰、正确、可用、健壮、灵活和可维护的程序来。如果你能够做到这一点的话，那么要想获得所需要的性能往往就相对比较简单了（见第55条）。有些条目确实谈到了性能问题，甚至有的还提供了性能指标。但是，在提及这些指标的时候，也会出现“在我的机器上”这样的话，所以，你最好把这些指标视同近似值。

有必要提及的是，我的机器是一台过时的家用电脑，主机为2.2 GHz双核AMD Opteron 170，2G内存，在Microsoft Windows XP Professional SP2操作系统平台上运行Sun 1.6_05发行版本的Java SE Development Kit（JDK）。这个JDK有两台虚拟机：Java HotSpot Client和Server VM。性能指标是在Server VM上测量的。

讨论Java程序设计语言及其类库特性的时候，有时候必须要指明具体的发行版本。为了简单起见，本书使用了*工程版本号*（engineering version number），而不是正式的发行名称。表1-2列出了发行名称与工程版本号之间的对应关系。

表1-2 Java的工程版本号

正式发行名称	工程版本号
JDK 1.1.x / JRE 1.1.x	1.1
Java 2 Platform, Standard Edition, v 1.2	1.2
Java 2 Platform, Standard Edition, v 1.3	1.3
Java 2 Platform, Standard Edition, v 1.4	1.4
Java 2 Platform, Standard Edition, v 5.0	1.5
Java Platform, Standard Edition 6	1.6

尽管这些例子都很完整，但是它们注重可读性更甚于注重完整性。它们直接使用了java.util和java.io包中的类。为了编译这些示例程序，可能需要在程序中加上一行或者多行这样的import语句：

```
import java.util.*;
import java.util.concurrent.*;
import java.io.*;
```

其他代码示例中也有类似被省略的情况。但是，在本书的Web站点：http://java.sun.com/docs/books/effective，提供了每个示例的完整版本，你可以直接编译和运行这些示例。

本书采用的大部分技术术语都与《*The Java Language Specification, Third Edition*》[JLS]⊖相同。有一些术语则值得特别提出来。Java语言支持四种类型：接口（interface）、类（class）、数组（array）和基本类型（primitive）。前三种类型通常被称为引用类型（reference type），类实例和数组是对象（object），而基本类型的值则不是对象。类的成员（member）由它的域（field）、方法（method）、成员类（member class）和成员接口（member interface）组成。方法的签名（signature）由它的名称和所有参数类型组成；签名不包括它的返回类型。

本书也使用了一些与《The Java Language Specification》不同的术语。与《The Java Language Specification》不同的是，本书用术语“继承（inheritance）”作为“子类化（subclassing）”的同义词。本书不再使用“接口继承”这种说法，而是简单地说，一个类实现（implement）了一个接口，或者一个接口扩展（extend）了另一个接口。为了描述“在没有指定访问级别的情况下所使用的访问级别”，本书使用了描述性的术语“包级私有（package-private）”，而不是如[JLS, 6.6.1]中所使用的技术性术语“缺省访问（default access）级别”。

本书也使用了一些在《The Java Language Specification》中没有定义的技术术语。术语“导出的API（exported API）”，或者简单地说API，是指类、接口、构造器（constructor）、成员和序列化形式（serialized form），程序员通过它们可以访问类、接口或者包。（术语API是Application Programming Interface的简写，这里之所以使用API而不用接口（interface），是为了不与Java语言中的interface类型相混淆）。使用API编写程序的程序员被称为该API的用户（user），在类的实现中使用了API的类被称为该API的客户（client）。

类、接口、构造器、成员以及序列化形式被统称为API元素（API element）。导出的API由所有可在定义该API的包之外访问的API元素组成。任何客户端都可以使用这些API元素，而API的创建者则负责支持这些API元素。Javadoc工具类在默认操作模式下也正是为这些元素生成文档，这绝非偶然。不严格地讲，一个包的导出的API是由该包中的每个公有（public）类或者接口中所有公有的或者受保护的（protected）成员和构造器组成。

⊖ 该书影印版《Java语言规范》由机械工业出版社引进出版，书号是：7-111-18839。——编辑注

第2章

创建和销毁对象

本章的主题是创建和销毁对象：何时以及如何创建对象，何时以及如何避免创建对象，如何确保它们能够适时地销毁，以及如何管理对象销毁之前必须进行的各种清理动作。

第1条：考虑用静态工厂方法代替构造器

对于类而言，为了让客户端获取它自身的一个实例，最常用的方法就是提供一个公有的构造器。还有一种方法，也应该在每个程序员的工具箱中占有一席之地。类可以提供一个公有的*静态工厂方法*（static factory method），它只是一个返回类的实例的静态方法。下面是一个来自Boolean（基本类型boolean的包装类）的简单示例。这个方法将boolean基本类型值转换成了一个Boolean对象引用：

```
public static Boolean valueOf(boolean b) {
    return b ? Boolean.TRUE : Boolean.FALSE;
}
```

注意，静态工厂方法与设计模式[Gamma95, p.107]中的工厂方法模式不同。本条目中所指的静态工厂方法并不直接对应于设计模式中的工厂方法。

类可以通过静态工厂方法来提供它的客户端，而不是通过构造器。提供静态工厂方法而不是公有的构造器，这样做具有几大优势。

静态工厂方法与构造器不同的第一大优势在于，它们有名称。如果构造器的参数本身没有确切地描述正被返回的对象，那么具有适当名称的静态工厂会更容易使用，产生的客户端代码也更易于阅读。例如，构造器BigInteger（int, int, Random）返回的BigInteger可能为素数，如果用名为BigInteger.probablePrime的静态工厂方法来表示，显然更为清楚。（1.4的发行版本中最终增加了这个方法。）

一个类只能有一个带有指定签名的构造器。编程人员通常知道如何避开这一限制：通过提