

第1版广受好评，本次加入**泛型、Ajax、线程池、Java多进程**归并
内容更完善，是提升程序性能的不二之选

Java Optimize Programming

Java 优化编程

林胜利 王坤茹 编著

(第2版)



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
<http://www.phei.com.cn>

Java技术大系

Java 优化编程 (第2版)

林胜利 王坤茹 编著

贾兆高 封面设计

北京航空航天大学出版社

北京市西城区百万庄大街24号

邮政编码 100039

北京市西城区百万庄大街24号

电话 010-88254888

电报挂号 010-88254888

网址 www.bjup.com.cn

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书通过丰富、完整、富有代表性的实例，展示了如何提升 Java 应用性能，并且给出了优化前与优化后的 Java 应用程序的性能差别，以实际的实例与数字告诉你，为什么不可以这么做，应该怎么做，深入分析了影响 Java 应用程序性能的根本原因。本书不是教你怎样使用 Java 语言开发应用程序，而是教你怎样才能开发出更高效、更优秀的 Java 应用程序。书中每一个例子都经过了作者严格的验证。

本书适合于所有想编写更高效、完美 Java 应用程序的开发人员阅读。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目 (CIP) 数据

Java 优化编程 / 林胜利, 王坤茹编著. —2 版. —北京: 电子工业出版社, 2007.7
(Java 技术大系)

ISBN 978-7-121-04564-6

I. J… II. ①林… ②王… III. JAVA 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2007) 第 082651 号

责任编辑: 高洪霞

印 刷: 北京市通州大中印刷厂

装 订: 三河市鹏成印业有限公司

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: 787×1092 1/16 印张: 26.25 字数: 643 千字

印 次: 2007 年 7 月第 1 次印刷

印 数: 6000 册 定价: 49.00 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 zltts@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

写作背景

Java 语言一经问世，就打破以往 C++ 在面向对象开发界一统天下的格局。Java 语言是一门非常纯正的面向对象语言，它已经被广泛地应用到电信应用系统与企业级系统软件的开发领域中。在目前从事软件开发的程序员中，应用 Java 语言进行开发的占了非常大的比例，这主要是因为 Java 语言的跨平台性与强大的功能。但是大部分开发人员对 Java 语言只是知道如何应用，至于怎样才能应用好，怎样才能提高 Java 语言开发应用程序的效率，却知之甚少。只有知道如何更好地运用一门优秀的语言，才能充分发挥它的潜在功用，开发出优质的软件。一门语言好比一个工具，如果使用者只知道如何简单地使用它，那么即使这个工具非常优秀，这个使用者也只能用这个工具创造出平庸的作品。对 Java 语言来讲亦是如此。

写作目的

正是考虑到以上原因，编写本书的目的不是教读者如何使用 Java 语言，也不是向读者介绍 Java 虚拟机的深奥理论，而是通过大量的知识点与开发实例结合作者多年的开发经验，向读者介绍如何提升用 Java 语言所开发的应用程序的效率与性能，目的是告诉那些已经具备 Java 开发基础知识的开发人员与涉猎 Java 开发领域不久的开发人员，怎样才能写出优秀的大师级的 Java 代码，从而帮助他们快速成为优秀的 Java 程序员。

本书特色

本书虽然重点讲解了如何提升 Java 应用程序性能的相关理论，但是本书不是一本侧重理论的图书。过于强调理论的图书，通常内容晦涩，容易让读者读后有种云里雾里的感觉，很难体会理解。本书通过丰富、完整、更具代表性的实例，展示了如何提升应用程序性能。本书不但告诉读者如何提升应用程序的性能，并且给出了这样做与不这样做开发的 Java 应用程序的性能差别，以实际的实例与数字告诉读者，为什么不可以这么做，应该怎么做，并且告诉读者影响 Java 应用程序性能的根本原因是什么，这样可以使读者能更深刻地理解书中所涉及的知识点，从而达到深刻理解、熟练运用的目的。

本书没有教读者怎样使用 Java 语言进行开发，而是教读者怎样才能开发出更高效、更优秀的 Java 应用程序，这是本书的最大特点。举个例子来说，本书没有具体讲解如何运用 Java 核心类来进行开发，因为这些知识很容易找到相关的技术资料，也容易被开发人员掌握。本书重点讲解的是影响 Java 应用程序性能的核心类的特点，以及在开发应用

程序时，它们所造成的性能瓶颈，如何可以摆脱这个性能瓶颈，或者给出打破这个瓶颈的可行方法，引导读者掌握合理使用这些核心类，编制性能最优化的应用程序。书中所涉及的每一个知识点几乎都是 Java 开发中的技术要点，并在讲解这些知识点时，给出如何避免进入某些技术误区的方法与措施。

本书非常注重内容的充实性与实用性，希望读者通过阅读本书，能获取更多实用的 Java 程序设计技术知识。书中的每一章都包含较多的知识点，通常情况下每一个知识点是一节，在每一章的结束部分都有一个小结，小结的目的是总结回顾这一章中最重要的知识点，以加深读者对本章所学知识的印象。本书在对每一个知识点的讲解过程中都附有详细的例子，每个例子都具有代表性，并且经过了笔者严格验证，相信读者在掌握本书所涵盖的知识之后一定能开发出具备最优性能的 Java 应用程序。

本书大部分知识点都是作者多年开发经验的结晶，衷心地希望它们能帮助读者在 Java 开发的职业生涯中少走弯路，养成良好的编程习惯，尽早成为 Java 开发专家。

本书第一版得到读者的广泛好评，作者紧跟 Java 技术的发展步伐对第一版的内容做了必要的调整，对过时的技术做了必要的删减，对精要技术做了进一步的完善，并在此基础上追加了目前较为前卫的优化技术如泛型，Ajax，线程池，Java 多进程归并等，使本书的品质得到了较大的提升。

本书结构

本书共分为 15 章。由于本书并不是一本系统讲解 Java 语言的程序设计的图书，它着重介绍如何优化 Java 应用开发的关键技术的图书，因此，本书的内容是并列的，但是在写作过程中，笔者还是做了必要的调整与统筹，尽量做到由浅入深地编排本书的内容。下面简单介绍一下本书各章的内容安排。

第 1 章 Java 程序设计风格。本章详细地讲解了有关 Java 程序设计风格的技术知识，通过对本章知识的学习、掌握，只要读者能够坚持其中介绍的编写规则，仅从代码风格上来讲，读者就能够具备编写大师级 Java 代码的能力。

第 2 章 内存管理。本章介绍了 Java 内存管理的相关知识，虽然在 Java 程序设计中的内存是由 JVM 控制管理的，但并不是说开发人员无法通过改善应用程序或通过其他途径对内存的使用进行优化，本章通过对相关知识点的讲解，可以使读者对 Java 内存的使用了如指掌，运用自如，对 Java 的内存管理机制有系统的认识。在对知识点的讲解过程中，笔者给出了选择与放弃的规则，这样读者就可以在实际的开发过程中，避免由于不合理地使用 Java 内存而导致系统性能下降或者系统崩溃等情况的发生，并且可以权衡考虑在特定的场景下选用哪种合理的使用内存的技术。

第 3 章 表达式、语句与保留字。本章讲解了 Java 语言中有关表达式、语句与保留字的知识，通过对本章内容的学习，读者会发现在日常的应用程序开发中有很多值得注意的地方，却往往被忽略了。结合本章所讲解的知识，相信读者能在很多方面不断地完善、优化读者的应用程序。

第 4 章 Java 核心类与性能优化。本章讲解了 Java 语言中的部分核心类与开发应

用软件性能的知识。通过对本章的学习,相信读者在应用 Java 核心类做应用开发时,就有了取舍的依据及提高应用性能的方法。

第 5 章 JNI 程序设计与性能优化。本章讲解了 JNI 技术架构,JNI 技术中数据类型与处理方法,JNI 数学计算与性能,以及如何处理好 JNI 技术程序设计中的中文问题等相关技术知识。通过对本章的学习,读者能够对 JNI 技术建立一个完整概念,并且掌握其技术要点,知道在什么地方,什么时候应用 JNI 技术,以及怎样更好地应用 JNI 技术,开发优质应用软件。

第 6 章 类与接口。本章讲解了 Java 语言程序设计中有类与接口的相关知识。通过对本章的学习,读者能够对面向对象程序设计中的两个重要概念——类与接口——有更高层次上的认识。经历了从过去的知道到现在的精通这一过程,读者能有一个质的飞越。

第 7 章 JSP 与 Servlet 性能优化。本章分为两部分内容:第一部分讲解如何提升 JSP 应用性能,第二部分讲解如何提升 Servlet 应用性能,通过对本章的学习,读者能够进一步认识到开发良好的 JSP 与 Servlet 应用软件需要多方面综合考虑,并可以掌握提高 JSP 与 Servlet 应用性能的技术要点。

第 8 章 开发高性能的 EJB 应用。本章首先讲解了有关如何提高 EJB 应用性能的相关技术知识,接着讲解了如何优化 Session Bean 性能,最后讲解了 EJB 的结合应用法则与提高 EJB 应用性能的其他途径等内容。通过对本章的学习,读者能够进一步加深对 EJB 技术的认识,成为一名精通 EJB 技术的高手。

第 9 章 JMS 性能优化。本章讲解了有关 JMS 类型及其各自适用场合,如何优化 JMS 虚拟通道及如何优化消息发送与接收过程,如何选择合适的消息类型,JMS 服务器的选择及 JMS 技术的相关经验等内容。通过对本章的学习,读者能够进一步加深对 JMS 技术的了解,在实际的应用开发中,运用本章讲解的技术知识,开发更加高效的 JMS 应用。

第 10 章 JDBC 与性能优化。本章讲解了有关 JDBC 性能优化的技术知识,其中包括 JDBC 编程的步骤,选择最优的驱动程序优化应用性能,如何优化 JDBC 连接,如何优化 Statement 对象,如何优化 ResultSet 对象,以及如何优化 SQL 查询操作大数据字段的读取方法及 JDBC 编程的相关经验等内容。通过对本章的学习,读者能够更好地应用 JDBC 技术,掌握提升 JDBC 应用的方法,开发高性能的 JDBC 应用。

第 11 章 软件结构、设计模式与性能优化。本章首先对设计模式做了简要的介绍,接着讲解了常用模式的 Java 实现与结构优化的技术课题,以及 J2EE 中的模式与性能的技术知识。通过对本章的学习,读者可以在实际的应用开发中,熟练地使用书中介绍的模式来解决实际的问题,并且根据实际的需要创建新的模式。

第 12 章 Java 多线程技术与应用性能优化。本章主要讲解了有关 Java 多线程技术与应用性能优化的相关技术知识。通过对本章的学习,读者会对 Java 多线程技术与如何通过多线程技术与线程池技术来优化应用性能有一个更为深刻的认识。

第 13 章 Java 泛型与应用优化。本章主要讲解了有关 Java 泛型与应用优化的相关技术知识。通过对本章的学习,读者能够通过应用泛型技术编写更为安全高效的 Java 应用。

第 14 章 Ajax 技术与 Web 应用性能优化。本章主要讲解了有关 Ajax 技术与 Web 应用性能优化的相关技术知识。通过对本章的学习,读者可以基本掌握 Ajax 技术,同时能够运用该技术来优化 Web 应用部分场景实现的性能。

第 15 章 其他优化话题。本章主要讲解了 Java 编程中一些相对零散的优化技术。这些技术虽然没有被纳入某一个具体方面，比较零散，但是在实际的应用开发中也是非常有用的。灵活适当地运用这些技术同样可以在很大程度上改善应用性能。

附录 A Together 工具的使用简介。在附录 A 中，介绍软件开发、建模、发布工具 Together，这是一款非常不错的软件工具，尤其是在 UML 建模、文档生成等方面更是强于其他的同类产品。本附录主要介绍如何通过 Together 创建工程，如何完成工程建模，如何引用标准模式，创建工程模块等内容。

附录 B J2SE 5.0 的新特性与性能的提升。在附录 B 中，介绍 J2SE 的新特性，尤其是介绍 J2SE 5.0 在性能与使用简易方面有所提升的内容。该部分内容取材于 Sun 公司的 J2SE 5.0 的相关技术文档。

附录 C 编排代码的精美工具 JxBeauty。在附录 C 中，向读者介绍了一款非常精致的编排 Java 代码的工具 JxBeauty，并告诉读者如何使用该工具来编排、美化读者的 Java 代码，创建具良好风格的 Java 源程序。

读者对象

如果读者是接触 Java 语言不久的程序员，通过对本书的学习，读者可以缩短成为一名 Java 高手的周期；如果读者是一个熟练应用 Java 语言的程序员，但是对于 Java 语言开发应用程序时的性能考虑不够深入，或者没有系统、全面的认识，那么通过掌握本书中所讲解的知识，可以弥补读者这方面的不足，帮助读者在已有的基础上编写更高效、完美的 Java 应用程序。

致谢

作为本书的第一作者，首先感谢我的妻子给予我的极大支持与鼓励。在本书的写作过程中，赛贝斯软件（中国）公司软件研发部的吴昊、连旦晖、王健、赫虹等几位工程师，以及西门子电子（中国）公司研发中心的王立冬高级工程师，给予了许多建设性的建议，使本书所涵盖的知识更加全面、系统。另外刘华军、陈铁路、刘文彬、徐军华、毕晓冬、李艳崑、宋双锐、吕健、李乐、于永梅、路宗强、赵沛星、薛殿桥、吴晓华、郑英、刘志明、舍攀、杨杨、王坤敬、李海燕、吴雅莉、程彬、王坤强、刘佳伟等同志参与了本书部分章节的校对与编排工作，进一步修正、完善了本书的内容，在此一并致谢！

与编著者联系

书中若有不当之处，敬请读者批评指正。编者的 E-mail 地址是：lsl_wkr@yeah.net，欢迎读者来信指正，谢谢。

林胜利
2007 年 2 月

目 录

第 1 章 Java 程序设计风格	1
1.1 Java 文件名与文件组织结构	3
1.2 Java 文件注释头	3
1.3 包的声明与引用	5
1.4 类与接口的声明	6
1.5 Java 源文件编排格式	7
代码行长度与折行规则	7
1.6 程序注释	9
1.7 变量的声明初始化与放置	11
1.7.1 变量声明	11
1.7.2 变量初始化	12
1.7.3 变量放置	12
1.8 Java 程序语句编写规则	12
1.8.1 简单语句	13
1.8.2 复合语句	13
1.9 空格与空行的应用规则	17
1.9.1 空格的应用规则	17
1.9.2 空行的应用规则	18
1.10 方法、变量与常量的命名规则	19
1.10.1 方法的命名规则	19
1.10.2 变量的命名规则	19
1.10.3 常量的命名规则	20
1.11 Java 编程实践	20
1.11.1 访问实例与类中变量的规则	20
1.11.2 引用类中的静态变量与方法的规则	20
1.11.3 变量赋值规则	21
1.11.4 综合规则	21
小结	22
第 2 章 内存管理	23
2.1 垃圾回收	24
堆内存	26
2.2 JVM 中对象的生命周期	27

2.2.1	创建阶段	27
2.2.2	应用阶段	29
2.2.3	不可视阶段	32
2.2.4	不可到达阶段	33
2.2.5	可收集阶段、终结阶段与释放阶段	33
2.3	Java 中的析构方法 finalize	33
2.4	数组的创建	37
2.5	共享静态变量存储空间	37
2.6	对象重用与 GC	39
2.7	瞬间值	42
2.8	不要提前创建对象	43
2.9	JVM 内存参数调优	43
2.10	Java 程序设计中有内存管理的其他经验	48
小结		49
第 3 章	表达式、语句与保留字	50
3.1	表达式	51
3.1.1	括号规则	51
3.1.2	简单规则	52
3.1.3	单一意图规则	52
3.1.4	方法返回值比较规则	53
3.1.5	字符串比较规则	54
3.2	Java 语言中的保留字	55
3.2.1	静态的 (static)	56
3.2.2	超类 (super)	58
3.2.3	最终的 (final)	58
3.2.4	同步 (synchronized)	64
3.2.5	实例识别 (instanceof)	65
3.3	判断语句与循环语句	67
3.3.1	判断语句	67
3.3.2	循环语句	68
3.4	正则表达式	73
小结		74
第 4 章	Java 核心类与性能优化	75
4.1	散列表类与性能优化	76
4.1.1	线程同步散列表类	76
4.1.2	设置 ArrayList 初始化容量	80
4.1.3	ArrayList 与 LinkedList	81

4.2 String 类与性能优化	84
4.2.1 字符串累加与性能优化	84
4.2.2 字符串的 length()方法与性能优化	86
4.2.3 toCharArray()方法与性能优化	87
4.2.4 字符串转化为数字	88
4.3 系统 I/O 类	89
4.3.1 Java 语言中输入/输出流	89
4.3.2 通过系统缓冲流类提高 I/O 操作效率	91
4.3.3 通过自定义缓冲区提高 I/O 操作效率	93
4.3.4 通过压缩流提高 I/O 操作效率	97
4.3.5 通过非阻塞 I/O 优化应用性能	100
4.4 其他	104
4.4.1 数据格式化与性能优化	104
4.4.2 获取文件信息与性能优化	106
小结	107
第 5 章 JNI 程序设计与性能优化	108
5.1 JNI 技术架构	109
5.2 创建带有本地方法的 Java 应用	110
5.3 创建 C 端代码	111
5.3.1 创建 C 端代码头文件	111
5.3.2 创建 C 端代码主文件	113
5.4 JNI 技术中数据类型与处理方法	114
5.4.1 JNI 技术中的本地数据类型	115
5.4.2 访问 JNI 本地数据类型的方法	117
5.4.3 在 JNI 本地方法中访问数组	120
5.4.4 JNI 中的主要方法	123
5.5 JNI 中的重要技术	125
5.5.1 局部引用与全局引用	125
5.5.2 处理本地方法引起的 Java 错误	128
5.5.3 线程与本地方法	129
5.6 JNI 数学计算与性能优化	130
5.7 处理好 JNI 中的中文问题	131
小结	133
第 6 章 类与接口	134
6.1 类的构造器	135
6.1.1 构造器编写规则	136

6.2	类的继承规则	141
6.2.1	单线继承规则	142
6.2.2	包内部继承规则	143
6.2.3	逻辑包含继承规则	144
6.3	抽象类与接口	146
6.4	继承与组合的应用时机	148
6.5	接口与抽象类的应用时机	151
6.6	内部类	153
6.7	与性能相关的建议与经验	155
	小结	155
第 7 章	JSP 与 Servlet 性能优化	156
7.1	提升 JSP 应用性能	157
7.1.1	优化 jspInit() 方法	157
7.1.2	通过优化 _jspService() 方法提高系统性能	161
7.1.3	JSP 高级知识	167
7.2	提升 SERVLET 应用性能	168
7.2.1	提高 Servlet 应用性能的七个方法	169
7.2.2	合理缓冲静态数据与动态数据	170
7.2.3	改善 Servlet 应用性能的方法	171
7.2.4	Filter Servlet 与 Listener Servlet	172
	小结	177
第 8 章	开发高性能的 EJB 应用	178
8.1	采用 EJB 技术的必要性	179
8.1.1	EJB 技术的优势特性	179
8.1.2	EJB 技术体系具有清晰的架构层次	179
8.1.3	EJB 与传统 Bean 相比的性能优势	181
8.2	EJB 的类型	182
8.2.1	EJB 的生命周期	183
8.2.2	三种 EJB 的特点与适用场合	187
8.2.3	本地 EJB 与远程 EJB 的性能比较	189
8.2.4	有状态会话 EJB 与 HttpSession	190
8.2.5	EJB 3.0 规范中的 EJB	191
8.3	优化无状态会话 EJB 性能	194
8.3.1	如何控制无状态会话 EJB 的生命周期	195
8.3.2	通过调节无状态会话 EJB 实例池的大小来优化系统性能	196
8.3.3	无状态会话 EJB 资源的缓冲与释放	197

8.4 优化有状态会话 EJB 性能	197
8.4.1 控制有状态会话 EJB 生命周期	197
8.4.2 优化有状态会话 EJB 的主要途径	198
8.5 优化实体 EJB 的性能	200
8.5.1 如何控制实体 EJB 的生命周期	200
8.5.2 通过调节实体 EJB 实例池的大小来优化系统性能	201
8.5.3 控制好实体 EJB 中的事务	202
8.5.4 提高实体 EJB 应用性能的其他知识	207
8.6 优化消息 EJB 性能	210
8.6.1 如何控制消息 EJB 的生命周期	210
8.6.2 如何缓存释放系统资源	212
8.7 几种 EJB 的结合应用规则	212
8.8 提高 EJB 应用性能的其他途径	214
小结	215
第 9 章 JMS 性能优化	216
9.1 JMS 消息收发模式及其各自适用场合	218
9.2 发送与接收 JMS 消息	221
9.3 优化 JMS 中的会话对象	224
9.4 优化连接对象	226
9.5 优化消息目的地 Destination 及消息生产者与消费者	228
9.6 优化消息对象及合理使用事务机制	233
9.7 影响 JMS 性能的其他因素	234
小结	237
第 10 章 JDBC 与性能优化	238
10.1 选择最优的驱动程序优化应用性能	240
10.2 优化 JDBC 连接	242
10.2.1 设置合适的预取行值	242
10.2.2 采用连接池技术	244
10.2.3 合理应用事务	245
10.2.4 选择合适的事务隔离层与及时关闭连接	247
10.3 优化 Statement 对象	248
10.4 优化 ResultSet 对象	252
10.4.1 通过缓冲数据行提高系统性能	252
10.4.2 通过设置合适的处理数据行的方向提高系统性能	253
10.4.3 通过采用合适的 getXXX() 方法提高系统性能	254
小结	255

第 11 章 软件结构、设计模式与性能优化	256
11.1 模式简介	257
11.2 常用模式的 Java 实现与结构优化	258
11.2.1 单态/单件模式的结构优化	259
11.2.2 抽象工厂模式的结构优化	261
11.2.3 适配器模式的结构优化	266
11.2.4 代理模式的结构优化	271
11.2.5 桥接模式的结构优化	273
11.2.6 命令模式的结构优化	275
11.2.7 观察者模式的结构优化	278
11.2.8 职责链模式的结构优化	281
11.2.9 中介者模式的结构优化	285
11.2.10 访问者模式的结构优化	288
11.2.11 任务分配中心模式的结构优化	291
11.3 J2EE 中的模式与性能优化	296
11.3.1 服务定位模式与性能优化	297
11.3.2 会话门面模式与性能优化	300
11.3.3 消息门面模式与性能优化	302
11.3.4 返回值打包模式与性能优化	303
11.3.5 返回值包装工厂模式与性能优化	305
11.3.6 值列表处理器模式与性能优化	306
11.3.7 复合实体模式与性能优化	308
小结	309
第 12 章 Java 多线程技术与应用性能优化	310
12.1 Java 多线程技术	311
12.1.1 进程与线程	311
12.1.2 线程的生命周期	311
12.2 并行任务与性能	313
12.2.1 并行任务与多线程	313
12.2.2 并行任务与死锁	313
12.3 线程池技术与应用性能优化	315
12.3.1 线程池	315
12.3.2 调优线程池的尺寸	315
12.4 通过线程池技术优化套接字网络编程	316
小结	322

第 13 章	Java 泛型与应用优化	323
13.1	认识泛型	324
13.1.1	使用泛型的收益	324
13.1.2	泛型与 JDK 5.0 中的集合类	325
13.2	使用泛型	327
13.2.1	创建支持泛型的类	327
13.2.2	泛型的自动解包装与自动包装的功能	328
13.2.4	限制泛型中类型参数的范围	329
小结		330
第 14 章	Ajax 技术与 Web 应用性能优化	331
14.1	了解 Ajax	332
14.2	通过 Ajax 技术改善 Web 应用性能	332
14.2.1	Ajax 技术实现	332
14.2.2	Ajax 技术性能优化实例	335
小结		340
第 15 章	其他优化话题	341
15.1	用 WeakHashMap 屏蔽内存泄漏	342
15.2	优化 Java 应用大小	344
15.3	通过 RandomAccess 接口优化迭代列表	346
15.4	合并 Java 中的多进程与系统优化	351
小结		356
附录 A	Together 工具的使用简介	357
附录 B	J2SE 5.0 的新特性与性能的提升	370
附录 C	编排代码的精美工具 JxBeauty	399

第 1 章

Java 程序设计风格

本章要点

本章讲解有关 Java 程序设计风格的技术知识，学习、掌握了本章知识后，仅从代码风格上来讲，读者将具备编写大师级 Java 代码的能力。知识点包括：

- Java 文件名与文件组织结构
- Java 文件注释头
- 包的声明与引用
- 类与接口的声明
- Java 源文件编排格式
- Java 程序注释
- 变量的声明初始化与放置
- Java 程序语句编写规则
- 空格与空行的应用规则
- 方法，变量与常量的命名规则
- 在 Java 编程实践中的三个规则

很多时候,开发人员都不太注意自己的程序风格,因此总给人一种“幼稚”的感觉。他们把精力完全放在了程序功能的实现上,当然,程序设计主要目的就是实现软件需求中所描述的用户需求,这本来是无可厚非的,但是不得不提醒开发人员的是,好的程序设计风格同样也非常重要,它会使你的开发生涯焕然一新,使你的开发工作走向“成熟”,令你受益无穷。

从近期目标看是应该先着眼于功能的实现,这也符合大多数管理者的意图,但是只能解一时之渴,而不思长远之计,确不可取。一个杂乱无序的程序让人看后有一种不解其意、心绪繁乱的感觉,由此而会抱怨、怀疑开发人员的技术素质与程序设计修为的低下。如果一个庞大的系统,所有程序代码的风格均是如此,我想这个系统能转交其他开发团队的可能性,就几乎没有了,即使是原班人马近时还可以读懂程序,并可修改程序,如若假以时日,恐怕连自己都会有一种茫然无措、无从下手的感觉。造成这个结果的罪魁祸首就是开发人员没有养成良好的程序设计风格,我想这不仅对开发人员有百害而无一益,对管理人员和企业而言更是损失惨重,结果是残局难以收拾。因为在通常情况下软件公司的人员流动是较为普遍、频繁的。如果软件系统因为无法继续维护开发,而半途夭折,将会给企业带来多大的损失是不言而喻的。即使在花费巨大的精力的前提下,系统得以继续开发,企业也因此付出了极大的人力、物力。因此,通常来说优秀的开发人员一般都有良好的程序设计风格,优秀的管理人员会在自己的开发团队中灌输养成良好程序设计风格的意识,优秀的软件企业都有自己一套健全、成熟的程序设计风格规范文档,用以规范开发人员编写代码的风格。

软件开发所采用的语言是多种多样的,目前较流行的程序设计语言有 Java、C、C++、C#、Basic 等,每一种语言都有自己的程序设计风格与规范,我们常常会在阅读一些高手所设计的程序代码时享受到因体会、感受到良好的代码风格所带给人一种舒服的感觉,同时也能感觉到此代码作者的水平。本书是介绍 Java 程序设计技术的,因此,我们只对 Java 程序设计的风格做一下讨论。

从事 Java 程序设计已有很长的时间了,在这之前我热衷于 C 与 C++ 程序设计,但是自从接触 Java 语言后,有一种相识恨晚的感觉,说得有些缠绵,不过当用 Java 开发一个软件产品哪怕是编写一个小的软件模块时都会有一种极强的成就感,会像欣赏一件完美的艺术品一样品味它曼妙的格式,Java 灵活的对象引用及对象方法的调用方式可以令你极尽串联引用之能事(如:obj.getArray()[i].getName().toString()),可谓“完美、神俊”。下面我们就来看下面一段曲线玲珑的代码:

```
... ..
if (elem == null) {
    for (int i = 0; i < size; i++)
        if (elementData[i]==null)
            return i;
} else if (elem == obj ) {
    if (maxIndex > 100) {
        for (int i = 0; i < size; i++)
            if (elem.equals(elementData[i]))
                return i;
    }
    else
        obj.getArray()[i].getName().toString();
}
```

```
    }  
    }  
    return -1;  
    .....
```

这段代码不长，结构也相对简单，却能体会出 Java 代码的绰约风姿。错落有致的代码确实会给人一种心旷神怡、赏心悦目的感觉。啰唆了这么多，读者可能有些不耐烦了，下面就详细地介绍在进行 Java 程序设计时应该注意的书写格式，应用好这些格式同样也能写出大师级的代码。

1.1 Java 文件名与文件组织结构

Java 文件名由实意文件名+后缀组成，后缀名因类型的不同而不同。如表 1-1 所示。

表 1-1 后缀名的意义

文件类型	后 缀 名
Java 源文件	.java
Java 字节码文件	.class

一般来说 Java 源文件的结构是这样的，每一节代码之间是由一个空行分割开的，并且每一节都有相应的注释，通常来讲一个 java 源文件不应该超过 2 000 行，否则就是视为类“臃肿”，这种情况在程序设计时，应该尽量避免。

在 Java 源文件中应该包含一个单一的公共类（class）或接口（interface），当私有类、私有接口与公共类有关联时，你可以把它们分到公共类的源文件中，这个公共类或公共接口，应该是这个源文件的第一个类或接口。

每一个 Java 源文件一般由下面的顺序构成：

- （1）文件注释头
- （2）包名（package）
- （3）引入（import）声明
- （4）类（class）或接口（interface）的声明部分

1.2 Java 文件注释头

Java 类文件注释头是用来描述该类功能及其特点，以及相关开发信息的，如该类的关联类（通常情况下不描述 Java 系统核心类如 java.util.Vector, java.lang.Thread 等）、开发公司或单位、版权、作者、代码审定人该类所支持的 JDK 版本、该类版本、开发日期、最后更改日期、修改人、复审人等信息，下面就是一个 Java 类文件注释头：

```
/*  
 * *****  
 * 该类功能及其特点的描述（例如：该类是用来……）  
 *  
 */
```