计 算 机 科 学 丛 书

Java编程思想

Thinking in JAVA

(美) Bruce Eckel 著 京京工作室 译

机械工业出版社 China Machine Press

Prentice Hall

计算机科学丛书

Java 编程思想

（美）Bruce Eckel 著

京京工作室 译

机械工业出版社
China Machine Press

PRENTICE HALL

本书指导读者利用Java语言进行面向对象的程序设计，详细介绍了Java的基本语法及它的高级特性(网络编程、高级面向对象能力、多线程)，系统地讲述了Java的高级理论，形象地阐述了面向对象基本理论。书中包括320个有用的Java程序，15 000余行代码，这些短小精悍的程序示例有助于读者理解含义模糊的概念。

本书适用于进行万维网编程和网页设计的程序员和计算机爱好者，也可作为大专院校相关专业的教学参考书。

本书版权登记号：图字：01-98-1434

图书在版编目(CIP)数据

Java 编程思想/(美) 埃克尔 (Eckel. B.) 著；京京工作室译. —北京：机械工业出版社，1999.4
(计算机科学丛书)
书名原文：Thinking in Java
ISBN 7-111-07064-X

Ⅰ.J… Ⅱ.①埃…②京… Ⅲ.Java 语言－程序设计 Ⅳ.TP312

中国版本图书馆CIP数据核字 (1999) 第04615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：吴 怡
北京昌平第二印刷厂印刷 · 新华书店北京发行所发行
1999年4月第1版 · 2000年10月第4次印刷
787mm × 1092mm 1/16 · 42.75印张
印数：11 001-16 000册
定价：60.00元

前　言

同人类任何语言一样，Java为我们提供了一种表达思想的方式。如操作得当，同其他方式相比，随着问题变得愈大和愈复杂，这种表达方式的方便性和灵活性会显露无遗。

不可将Java简单想象成一系列特性的集合；如孤立地看，有些特性是没有任何意义的。只有在考虑“设计”、而不是考虑简单的编码时，才能真正体会到Java的强大。为了按这种方式理解Java，首先必须掌握它与编程的一些基本概念。本书讨论了编程问题，它们为何会成为问题以及Java用以解决它们的方法。所以，我对每一章的解释都建立在如何用语言解决一种特定类型的问题基础上。按这种方式，我希望引导你一步一步地进入Java的世界，使其最终成为你使用的最自然的一种语言。

贯穿本书，我试图在你的大脑里建立一个模型，或者说一个“知识结构”。这样可加深对语言的理解。若遇到难解之处，应学会把它填入这个模型的对应地方，然后自行演绎出答案。事实上，学习任何语言时，脑海里有一个现成的知识结构往往会起到事半功倍的效果。

1. 前提

本书假定读者对编程多少有些熟悉。知道程序是一系列语句的集合，知道子程序／函数／宏是什么，知道像“if”这样的控制语句，也知道像“while”这样的循环结构。注意，这些东西在大量语言里都是类似的。假如你学过一种宏语言，或者用过Perl之类的工具，那么它们的基本概念并没有什么区别。总之，只要能习惯基本的编程概念，就可顺利阅读本书。当然，C/C++程序员在阅读时能得到更多的方便。但是，即使不熟悉C，一样不要把自己排除在外(尽管以后的学习要付出更多的努力)。我会讲述面向对象编程的概念，以及Java的基本控制机制，所以不用担心自己会打不好基础。况且，你需要学习的第一类知识就会涉及到基本的流程控制语句。

尽管经常都会谈及C和C++语言的一些特性，但并没有打算使它们成为内部参考，而是想帮助所有程序员都能正确地看待那两种语言。毕竟，Java是从它们那里衍生出来的。我将试着尽可能地简化这些引用和参考，并合理地解释一名非C/C++程序员通常不太熟悉的内容。

2. Java的学习

在我第一本书《Using C++》(Osborne/McGraw-Hill于1989年出版)面市的几乎同时，我开始教授那种语言。讲授程序设计语言已成为我的专业。自1989年以来，我在世界各地见过许多昏昏欲睡、茫然、不解的面孔。开始在室内面向较少的一组人授课以后，我从作业中发现了一些特别的问题。即使那些上课面带会心的微笑或者频频点头的学生，对许多问题也存在认识上的混淆。在过去几年间的“软件开发会议”上，由我主持C++分组讨论会(现在变成了Java讨论会)。有的演讲人试图在很短的时间内向听众灌输过多的主题。所以到最后，尽管听众的水平都还可以，而且提供的材料也很充足，但仍然失去了一部分听众。这可能是由于问得太多了，但由于我是那些采取传统授课方式的人之一，所以很想使每个人都能跟上讲课进度。

有段时间，我编制了大量教学简报。经过不断的试验和修订(或称“反复”，这是在Java程序设计中非常有用的一项技术)，最后成功地在一门课程中集成了从我的教学经验中总结出来的所有东西——我在很长一段时间里都在使用。其中由一系列离散的、易于消化的小步骤组成，而且每个小课程结束后都有一些适当的练习。我已在Java公开研讨会上公布了这一课程，大家

可到http://www.BruceEckel.com了解详情(对研讨会的介绍也以CD-ROM的形式提供，具体信息可在同样的Web站点找到)。

从每一次研讨会收到的反馈都帮助我修改及重新制订学习材料的重心，直到我最后认为它成为一个完善的教学载体为止。但本书并不仅仅是一本教科书——我尝试在其中装入尽可能多的信息，并按照主题进行了有序的分类。无论如何，这本书的主要宗旨是为那些独立学习的人士服务，他们正准备深入一门新的程序设计语言，而没有太大的可能参加此类专业研讨会。

3. 目标

就像我的另一本书《Thinking in C++》(中译本《C++编程思想》机械工业出版社出版)一样，这本书面向语言的教授进行了良好的结构与组织。特别是，我的目标是建立一套有序的机制，可帮助我在自己的研讨会上更好地进行语言教学。在我思考书中的一章时，实际上是在想如何教好一堂课。我的目标是得到一系列规模适中的教学模块，可以在合理的时间内教完。随后是一些精心挑选的练习，可以在课堂上立即完成。

在这本书中，我想达到的目标如下:

1) 每一次都将教学内容向前推进一小步，便于读者在继续后面的学习前消化前面的内容。

2) 采用的示例尽可能简短。当然，这样做有时会妨碍我解决“现实世界”的问题。但我同时也发现；对那些新手来说，如果他们能理解每一个细节，那么一般会产生更大的学习兴趣；而假如他们一开始就被要解决的问题的深度和广度所震慑，那么一般都不会收到很好的学习效果。另外，在实际教学过程中，对能够摘录的代码数量是有严格限制的。另一方面，这样做无疑会使有些人批评我采用了“不真实的例子”，但只要能起到良好的效果，我宁愿接受这一指责。

3) 要揭示的特性按照我精心挑选的顺序依次出场，而且尽可能符合读者的思想历程。当然，我不可能永远都做到这一点；在那些情况下，会给出一段简要的声明，指出这个问题。

4) 只把我认为有助于理解语言的东西介绍给读者，而不是把我所知道的一切东西都抖出来，这并非藏私。我认为信息的重要程度存在一个合理的层次。有些情况是95%的程序员都永远不必了解的。如强行学习，只会干扰他们的正常思维，从而加深语言在他们面前表现出来的难度。以C语言为例，假如你能记住运算符优先次序表(我从来记不住)，那么就可以写出更“聪明”的代码。但再深入想一层，那也会困扰代码的读者／维护者。所以忘了那些次序吧，在拿不准的时候加上括号即可。

5) 每一节都有明确的学习重点，所以教学时间(以及练习的间隔时间)非常短。这样做不仅能保持读者思想的活跃，也能使问题更容易理解，对自己的学习产生更大的信心。

6) 提供一个坚实的基础，使读者能充分理解问题，以便更容易地转向一些更加困难的课程和书籍。

4. 联机文档

由Sun微系统公司提供的Java语言和库(可免费下载)配套提供了电子版的用户帮助手册，可用Web浏览器阅读。此外，由其他厂商开发的几乎所有类似产品都有一套等价的文档系统。而目前出版的与Java有关的几乎所有书籍都重复了这份文档。所以你要么已经拥有了它，要么需要下载。所以除非特别必要，否则本书不会重复那份文档的内容。因为一般地说，用Web浏览器查找与类有关的资料比在书中查找方便得多(电子版的东西更新也快)。只有在需要对文档进行补充，以便你能理解一个特定的例子时，本书才会提供有关类的一些附加说明。

5. 章节

本书在设计时认真考虑了人们学习Java语言的方式。在我授课时，学生们的反映有效地帮助了我认识哪些部分是比较困难的，须特别留意。我也曾经一次讲述了太多的问题，但得到的

教训是：假如包括了大量新特性，就需要对它们全部做出解释，而这特别容易加剧学生们的混淆。因此，我做大量努力，使这本书一次尽可能地少涉及一些问题。

因此，我在书中的目标是让每一章都讲述一种语言特性，或者只讲述少数几个相互关联的特性。这样一来，读者在转向下一主题时，就能更容易地消化前面学到的知识。

下面列出对本书各章的简要说明，它们与我实际进行的课堂教学是对应的。

第1章：对象入门

这一章是对面向对象的程序设计(OOP)的一个综述，其中包括对“什么是对象”之类的基本问题的回答，并讲述了接口与实现，抽象与封装，消息与函数，继承与合成以及非常重要的多形性的概念。这一章会向大家提出一些对象创建的基本问题，比如构建器、对象存在于何处、创建好后把它们置于什么地方以及魔术般的垃圾收集器(能够清除不再需要的对象)。要介绍的另一些问题还包括通过违例实现的错误控制机制、反应灵敏的用户界面的多线程处理以及连网和因特网等等。大家也会从中了解到是什么使得Java如此特别，它为什么取得了这么大的成功，以及与面向对象的分析与设计有关的问题。

第2章：一切都是对象

本章将大家带到可以着手写自己的第一个Java程序的地方，所以必须对一些基本概念做出解释，其中包括：对象“句柄”的概念；怎样创建一个对象；对基本数据类型和数组的介绍；作用域以及垃圾收集器清除对象的方式；如何将Java中的所有东西都归为一种新数据类型(类)，以及如何创建自己的类；函数、自变量以及返回值；名字的可见度以及使用来自其他库的组件；static关键字；注释和嵌入文档等等。

第3章：控制程序流程

本章开始介绍起源于C和C++而由Java继承的所有运算符。除此以外，还要学习运算符中一些不易使人注意的问题，以及涉及造型、升迁及优先次序的问题。随后讲述基本的流程控制及选择运算，这些几乎是所有程序设计语言都具有的特性：用if-else实现选择；用for和while实现循环；用break和continue以及Java的标签式break和contiune(它们被认为是Java中“不见的goto”)退出循环；用switch实现另一种形式的选择。尽管这些与C和C++中见到的有一定的共同性，但多少存在一些区别。除此以外，所有示例都是完整的Java示例，能使大家很快地熟悉Java的外观。

第4章：初始化和清除

本章开始介绍构建器，它的作用是担保初始化的正确实现。对构建器的定义要涉及函数过载的概念(因为可能同时有几个构建器)。随后要讨论的是清除过程，它并不肯定如想象的那么简单。用完一个对象后，通常可以不必管它，垃圾收集器会自动介入，释放由它占据的内存。这里详细探讨了垃圾收集器及它的一些特点。在这一章的最后，我们将更贴近地观察初始化过程：自动成员初始化、指定成员初始化、初始化的顺序、static(静态)初始化以及数组初始化等等。

第5章：隐藏实施过程

本章要探讨将代码封装到一起的方式，以及在库的其他部分隐藏时，为什么仍有一部分处于暴露状态。首先要讨论的是package和import关键字，它们的作用是进行文件级的封装(打包)操作，并允许我们构建由类构成的库(类库)。此时也会谈到目录路径和文件名的问题。本章剩下的部分将讨论public、private及protected三个关键字，“友好”访问的概念，以及各种场合下不同访问控制级的意义。

第6章：类再生

继承的概念是几乎所有OOP语言中都占有重要的地位。它是对现有类加以利用，并为其添

加新功能的一种有效途径(同时可以修改它，这是第7章的主题)。通过继承来重复使用原有的代码时(再生)，一般需要保持“基础类”不变，只是将这儿或那儿的东西串联起来，以达到预期的效果。然而，继承并不是在现有类基础上制造新类的唯一手段。通过“合成”，亦可将一个对象嵌入新类。在这一章中，大家将学习在Java中重复使用代码的这两种方法，以及具体如何运用。

第7章：多形性

若由你自己来干，可能要花9个月的时间才能发现和理解多形性的问题，这一特性实际是OOP一个重要的基础。通过一些小的、简单的例子，读者可知道如何通过继承来创建一系列类型，并通过它们共有的基础类对那个系列中的对象进行操作。通过Java的多形性概念，同一系列中的所有对象都具有共通性。这意味着，我们编写的代码不必再依赖特定的类型信息。这使程序更易扩展，包容力也更强。由此，程序的构建和代码的维护可以变得更方便，付出的代价也会更低。此外，Java还通过“接口”提供了设置再生关系的第三种途径。这里所说的“接口”是对对象物理“接口”一种纯粹的抽象。一旦理解了多形性的概念，接口的含义就很容易解释了。本章也向大家介绍了Java 1.1的“内部类”。

第8章：对象的容纳

对一个非常简单的程序来说，它可能只拥有一个固定数量的对象，而且对象的“生存时间”或者“存在时间”是已知的。但是通常，我们的程序会在不确定的时间创建新对象，只有在程序运行时才可了解到它们的详情。此外，除非进入运行期，否则无法知道所需对象的数量，甚至无法得知它们的确切类型。为解决这个常见的程序设计问题，我们需要拥有一种能力，可在任何时间、任何地点创建任何数量的对象。本章的宗旨便是探讨在使用对象的同时用来容纳它们的一些Java工具：从简单的数组到复杂的集合(数据结构)，如Vector和Hashtable等。最后，我们还会深入讨论新型和改进过的Java 1.2集合库。

第9章：违例差错控制

Java最基本的设计宗旨之一便是组织错误的代码不会真的运行起来。编译器会尽可能地捕获问题。但某些情况下，除非进入运行期，否则问题是不会被发现的。这些问题要么属于编程错误，要么是一些自然的出错状况，它们只有在作为程序正常运行的一部分时才会成立。Java为此提供了“违例控制”机制，用于控制程序运行时产生的一切问题。这一章将解释try、catch、throw、throws及finally等关键字在Java中的工作原理，并讲述什么时候应当“掷”出违例，以及在捕获到违例后该采取什么操作。此外，大家还会学习Java的一些标准违例，如何构建自己的违例，违例发生在构建器中怎么办，以及违例控制器如何定位，等等。

第10章：Java IO系统

理论上，我们可将任何程序分割为三个部分：输入、处理和输出。这意味着IO(输入／输出)是所有程序最为关键的部分。在这一章中，大家将学习Java为此提供的各种类，如何用它们读写文件、内存块以及控制台等。将着重强调“老”IO和Java 1.1的“新”IO。除此之外，本节还要探讨如何获取一个对象、对其进行“流式”加工(使其能置入磁盘或通过网络传送)以及重新构建它等等。这些操作在Java的1.1版中都可以自动完成。另外，我们也要讨论Java 1.1的压缩库，它将用在Java的归档文件格式(JAR)中。

第11章：运行期类型鉴定

若只有指向基础类的一个句柄，Java的运行期类型标识(RTTI)使我们能获知一个对象的准确类型是什么。一般情况下，我们需要有意忽略一个对象的准确类型，让Java的动态绑定机制(多形性)为那一类型实现正确的行为。但在某些场合下，对于只有一个基础句柄的对象，我们

仍然特别有必要了解它的准确类型是什么。拥有这个资料后，通常可以更有效地执行一次特殊情况下的操作。本章将解释RTTI的用途、如何使用以及在适当的时候如何放弃它。此外，Java 1.1的“反射”特性也会在这里得到介绍。

第12章：传递和返回对象

由于我们在Java中同对象沟通的唯一途径是“句柄”，所以将对象传递到一个函数里以及从那个函数返回一个对象的概念就非常有趣了。本章将解释在函数中进出时，什么才是为了管理对象需要了解的。同时也会讲述String(字串)类的概念，它用一种不同的方式解决了同样的问题。

第13章：创建窗口和程序片

Java配套提供了“抽象Window工具包”(AWT)。这实际是一系列类的集合，能以一种可移植的形式解决视窗操纵问题。这些窗口化程序既可以程序片的形式出现，亦可作为独立的应用程序使用。本章将向大家介绍AWT以及网上程序片的创建过程。我们也会探讨AWT的优缺点以及Java 1.1在GUI方面的一些改进。同时，也会在这里强调重要的“Java Beans”技术。Java Beans是创建“快速应用开发”(RAD)程序构造工具的重要基础。我们最后介绍Java 1.2的“Swing”库——它使Java的UI组件得到了显著的改善。

第14章：多线程

Java提供了一套内建的机制，可提供对多个并发子任务的支持，我们称其为“线程”。这线程均在单一的程序内运行。除非机器安装了多个处理器，否则这就是多个子任务的唯一运行方式。尽管还有别的许多重要用途，但在打算创建一个反应灵敏的用户界面时，多线程的运用显得尤其重要。举个例子来说，在采用了多线程技术后，尽管当时还有别的任务在执行，但用户仍然可以毫无阻碍地按下一个按钮，或者键入一些文字。本章将对Java的多线程处理机制进行探讨，并介绍相关的语法。

第15章：网络编程

开始编写网络应用时，就会发现所有Java特性和库仿佛早已串联到了一起。本章将探讨如何通过因特网通信，以及Java用以辅助此类编程的一些类。此外，这里也展示了如何创建一个Java程序片，令其同一个“通用网关接口”(CGI)程序通信；揭示了如何用C++编写CGI程序；也讲述了与Java 1.1的“Java数据库连接”(JDBC)和“远程方法调用”(RMI)有关的问题。

第16章：设计范式

本章将讨论非常重要但同时也是非传统的“范式”程序设计概念。大家会学习设计进展过程的一个例子。首先是最初的方案，然后经历各种程序逻辑，将方案不断改革为更恰当的设计。通过整个过程的学习，大家可体会到使设计思想逐渐变得清晰起来的一种途径。

第17章：项目

本章包括一系列项目，它们要么以本书前面讲述的内容为基础，要么对以前各章进行一番扩展。这些项目显然是书中最复杂的，它们有效地演示了新技术和类库的应用。

有些主题似乎不适于放到本书的核心位置，但我发现有必要在教学时讨论它们，这些主题都放入了本书的附录中。

附录A：使用非Java代码

一个完全能够移植的Java程序，肯定存在一些严重的缺陷：速度太慢，而且不能访问与具体平台有关的服务。若事先知道程序要在什么平台上使用，就可考虑将一些操作变成“固有方法”，从而显著加快执行速度。这些“固有方法”实际是一些特殊的函数，以另一种程序设计语言写成(目前仅支持C/C++)。Java还可通过另一些途径提供对非Java代码的支持，其中包括CORBA。本附录将详细介绍这些特性，以便大家能够创建一些简单的例子，同非Java代码打交道。

附录B: C++和Java的对比

对一个C++程序员，他应该已经掌握了面向对象程序设计的基本概念，而且Java语法对他来说无疑是非常眼熟的。这一点是明显的，因为Java本身就是从C++衍生而来。但是，C++和Java之间的确存在一些显著的差异。这些差异意味着Java在C++基础上做出的重大改进。一旦理解了这些差异，就能理解为什么说Java是一种杰出的语言。这一附录便是为这个目的设立的，它讲述了使Java与C++明显有别的一些重要特性。

附录C: Java编程规则

本附录提供了大量建议，帮助大家进行低级程序设计和代码编写。

附录D: 性能

通过这个附录的学习，大家可发现自己的Java程序中存在的瓶颈，并可有效地改善执行速度。

附录E: 关于垃圾收集的一些话

这个附录讲述了用于实现垃圾收集的操作和方法。

附录F: 推荐读物

列出我觉得特别有用的一系列Java参考书。

6. 练习

为巩固对新知识的掌握，我发现简单的练习特别有用。读者在每一章结束时都能找到一系列练习。

大多数练习都很简单，在合理的时间内可以完成。如将本书作为教材，可考虑在课堂内完成。老师要注意观察，确定所有学生都已消化了讲授的内容。有些练习要难些，它们是为那些有兴趣深入的读者准备的。大多数练习都可在较短时间内做完，有效地检测和加深你的知识。有些题目具有挑战性，但都不会太麻烦。事实上，练习中碰到的问题在实际应用中也会经常碰到。

7. 多媒体CD-ROM

本书配套提供了一张多媒体CD-ROM，可单独购买及使用。它与其他计算机书籍的普通配套CD不同，那些CD通常仅包含书中用到的源码(本书的源码可从www.BruceEckel.com免费下载)。本CD-ROM是一个独立的产品，包含了一周“Hads-OnJava”培训课程的全部内容。这是一个由Bruce Eckel讲授的、时间在15小时以上的课程，含500张以上的演示幻灯片。该课程建立在这本书的基础上，所以是非常理想的一个配套产品。

CD-ROM包含了本书的两个版本:

1) 本书一个可打印的版本，与下载版本完全一致。

2) 为方便读者在屏幕上阅读和索引，CD-ROM提供了一个独特的超链接版本。这些超链接包括:

- 230个章、节和小标题链接
- 3600个索引链接

CD-ROM包含600MB以上的数据。我相信它已对“物超所值”做了崭新的定义。

CD-ROM包含本书打印版的所有东西，另外还有来自五天快速入门课程的全部材料。我相信它建立了一个新的书刊品质评定标准。

若想单独购买此CD-ROM，只能从Web站点www.BruceEckel.com处直接订购。

8. 源代码

本书所有源码都作为保留版权的免费软件提供，可以独立软件包的形式获得，亦可从http://www.BruceEckel.com下载。为保证大家获得的是最新版本，我用这个正式站点发行代码

及本书电子版。亦可在其他站点找到电子书和源码的镜像版(有些站点已在http://www.BruceEckel.com 处列出)。但无论如何，都应检查正式站点，确定镜像版确实是最新的版本。可在课堂和其他教育场所发布这些代码。

版权的主要目标是保证源码得到正确的引用，并防止在未经许可的情况下，在印刷材料中发布代码。通常，只要源码获得了正确的引用，在大多数媒体中使用本书的示例都没有什么问题。

在每个源码文件中，都能发现下述版本声明文字:

```
//////////////////////////////////////////////
// Copyright (c) Bruce Eckel, 1998
// Source code file from the book "Thinking in Java"
// All rights reserved EXCEPT as allowed by the
// following statements: You can freely use this file
// for your own work (personal or commercial),
// including modifications and distribution in
// executable form only. Permission is granted to use
// this file in classroom situations, including its
// use in presentation materials, as long as the book
// "Thinking in Java" is cited as the source.
// Except in classroom situations, you cannot copy
// and distribute this code; instead, the sole
// distribution point is http://www.BruceEckel.com
// (and official mirror sites) where it is
// freely available. You cannot remove this
// copyright and notice. You cannot distribute
// modified versions of the source code in this
// package. You cannot use this file in printed
// media without the express permission of the
// author. Bruce Eckel makes no representation about
// the suitability of this software for any purpose.
// It is provided "as is" without express or implied
// warranty of any kind, including any implied
// warranty of merchantability, fitness for a
// particular purpose or non-infringement. The entire
// risk as to the quality and performance of the
// software is with you. Bruce Eckel and the
// publisher shall not be liable for any damages
// suffered by you or any third party as a result of
// using or distributing software. In no event will
// Bruce Eckel or the publisher be liable for any
// lost revenue, profit, or data, or for direct,
// indirect, special, consequential, incidental, or
// punitive damages, however caused and regardless of
// the theory of liability, arising out of the use of
// or inability to use software, even if Bruce Eckel
// and the publisher have been advised of the
// possibility of such damages. Should the software
// prove defective, you assume the cost of all
// necessary servicing, repair, or correction. If you
// think you've found an error, please email all
// modified files with clearly commented changes to:
// Bruce@EckelObjects.com. (Please use the same
// address for non-code errors found in the book.)
//////////////////////////////////////////////
```

可在自己的开发项目中使用代码，并可在课堂上引用(包括学习材料)。但要确定版权声明在每个源文件中得到了保留。

9. 编码样式

对于本书的示例，我采用了一种特定的编码样式。该样式得到了大多数Java开发环境的支持。该样式问世已有几年，最早起源于Bjarne Stroustrup先生在《The C++ Programming Language》(Addison-Wesley 1991年出版，第2版)里采用的样式。由于代码样式目前是个敏感问题，极易招致数小时的激烈辩论，所以我在这里只想指出，自己并不打算通过这些示例建立一种样式标准。之所以采用这些样式，完全出于我自己的考虑。由于Java是一种形式非常自由的编程语言，所以读者完全可以根据自己的感觉选用适合的编码样式。

本书的程序是由字处理程序包括在正文中的，它们直接取自编译好的文件。所以，本书印刷的代码文件应能正常工作，不会造成编译器错误。会造成编译错误的代码已经用注释//!标出。因此很容易发现，也很容易用自动方式进行测试。读者发现并向作者报告的错误首先会在发行的源码中改正，然后在本书的更新版中校订(所有更新都会在Web站点http:// www.BruceEckel.com处出现)。

10. Java版本

尽管我用几家厂商的Java开发平台对本书的代码进行了测试，但在判断代码行为是否正确时，却通常以Sun公司的Java开发平台为准。

当你读到本书时，Sun应已发行了Java的三个重要版本：1.0、1.1及1.2(Sun声称每9个月就会发布一个主要更新版本)。就我看，1.1版对Java语言进行了显著改进，完全应标记成2.0版(由于1.1已做出了如此大的修改，真不敢想象2.0版会出现什么变化)。然而，它的1.2版看起来最终将Java推向一个全盛时期，特别是其中考虑到了用户界面工具。

本书主要讨论1.0和1.1版，也涉及1.2版的部分内容。但有些时候，新方法明显优于老方法。此时，我会明显偏向新方法，通常教给大家更好的方法，而完全忽略老方法。然而，有的新方法要以老方法为基础，所以不可避免地要从老方法入手。这一特点尤以AWT为甚，因为那里不仅存在数量众多的老式Java 1.0代码，有的平台仍然只支持Java 1.0。我会尽量指出哪些特性是哪个版本特有的。

大家会注意到，我并未使用子版本号，比如1.1.1。到本书完稿为止，Sun公司发布的最后一个1.0版是1.02；而1.1的最后版本是1.1.5(Java 1.2仍在β测试)。在这本书中，我只会提到Java 1.0、Java 1.1及Java 1.2，避免由于子版本编号过多造成的键入和印刷错误。

11. 课程和培训

我的公司提供了一个五日制的公共培训课程，以本书的内容为基础。每章内容都代表着一堂课，并附有相应的课后练习，以便巩固学到的知识。一些辅助用的幻灯片可在本书的配套光盘上找到，最大限度地方便各位读者。欲了解更多的情况，请访问：

```
http://www.BruceEckel.com
```

或发函至：

```
Bruce@EckelObjects.com
```

我的公司也提供了咨询服务，指导客户完成整个开发过程——特别是你的单位首次接触Java开发的时候。

12. 错误

无论作者花多大精力来避免，错误总是从意想不到的地方冒出来。如果你认为自己发现了一个错误，请在源文件(可在http://www.BruceEckel.com处找到)里指出有可能是错误的地方，

填好我们提供的表单。将你推荐的纠错方法通过电子函件发给Bruce@EckelObjects.com。经适当的核对与处理，Web站点的电子版及本书的下一个印刷版本会做出相应的改正。具体格式如下:

1) 在主题行(Subject)写上“TIJ Correction”(去掉引号)，以便你的函件进入对应的目录。

2) 在函件正文，采用下述形式:

find: 在这里写一个单行字串，以便我们搜索错误所在的地方

Comment: 在这里可写多行批注正文，最好以“here’s how I think it shoud read”开头

###

其中，“###”指出批注正文的结束。这样一来，我自己设计的一个纠错工具就能对原始正文来一次“搜索”，而你建议的纠错方法会在随后的一个窗口中弹出。

若希望在本书的下一版本中添加什么内容，或对书中的练习题有什么意见，也欢迎你指出。我们感谢你的所有意见。

目　录