本书如此生动有趣，
以致于我想亲自编
写一个Java支程序！
-SCOTT McNEALY
Chairman,CEO,
and President
Sun Microsystems,Inc.

OSBORNE

BONUS
http://www.osborne.com
Free Code on Web Page

JAVA 使用手册

[美] Patrick Naughton 著
●谢小兵　于春燕　译　●邓召义　审校

JAVA HANDBOOK

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

JAVA 使用手册

The Java Handbook

JAVA 使用手册

[美] Patrick Naughton 著
谢小兵 于春燕 译
邓召义 审校

電子工業出版社
Publishing House of Electronics Industry

内 容 提 要

由 Sun Microsystems 推出的革命性的编程环境 Java 使 World Wide Web 进入一个全新的交互领域。具有强大功能的动画、多媒体和交互能力的 Java 是一种灵活的、面向对象的 Internet 编程语言。

本书共分三部分 20 章，从面向对象的基本概念入手，深入浅出地介绍了 Java 语言结构、Java 的类库、多线程、网络和图形编程等方面的内容，并充分利用实例，讲述了创建交互式 Web 页面所需的技巧。

本书语言流畅、内容翔实、分析透彻，是一本适合广大计算机爱好者的优秀读物。

The Java Handbook

JAVA 使用手册

[美] Patrick Naughton 著

谢小兵 于春燕 译

邓召义 审校

责任编辑 应月燕

*

电子工业出版社出版(北京市万寿路)

电子工业出版社发行 各地新华书店经销

北京大中印刷厂印刷

*

开本：787×1092 毫米 1/16 印张：17.25 字数：430 千字

1996 年 10 月第一版 1996 年 10 月第一次印刷

印数：6000 册 定价：28 元

ISBN 7-5053-3841-2/TP・1654

著作权合同登记号 图字：01-1995-445

前　言

Java 是一种 Internet 的编程语言——至少在本书中会有这种感觉，Java 实际上是C++的一种变形，可用它来创建安全的、可移植的、面向对象的、多线程的、交互式的程序。如果你不想编写这类程序，那么就不能从这本书学到什么。严格地说，Java 与 Internet 有关联是因为被大家称为"killerapp"的第一个用 Java 语言编写的非常有趣的应用程序是一个交互式的 Web 浏览器。

Java 正受到史无前例的关注。公众被 Java 与平台无关的编程环境所吸引。这种新的编程语言概念——即用它开发的程序可在多种平台上运行——是对那些以专有接口来维持市场占有率的系统开发商的公开挑战。这引起关注编程语言的人们对它的极大兴趣。

不仅如此，这本书不是有关 Java 的广告，而是讲述一种全新的编程环境，它可以解决传统语言中存在的许多问题。Java 是由一群永不满足的工程师开发出来的，目的是作为一种最终的编程语言，而结果是，目的促进了手段，手段也变成了结果。

Java 是一种非常有趣的编程语言。你可以不用费多大周折就能很快开发出程序并得到满意的结果。Java 的编译器能准确、有效地报告错误以利修改。比C++更简洁的语法不仅减轻了负担而且在用过之后还使人倍感新奇。许多从C++转到 Java 的程序员都再也不愿意回到C++。

尽管本书中的实例所采用的屏幕图和格式均来自 Windows 95 PC 机上，但所有程序均可在支持 Java 的平台上正确运行——今天，这些平台包括 Solaris、Irix、Linix、HPUX、OSF、Windows 95、Windows NT、OS/2 和 Macintosh。

本书将首先介绍面向对象编程技术的基本知识——这是对所有想用 Java 编程的人必须具有的预备知识。然后将花大量篇幅涵盖不需要太多面向对象技术的低级编程技术。这主要是为了帮助读者中的非专业程序员。一旦学会如何编写 Java 的类，我们就可以快速地掌握 Java 的类库。紧跟其后，我们将剖析几个有趣的支程序，以便观察如何拼装一个实用的 Java 程序。

本书的结构

本书共分三部分：Java 语言、Java 类库和实际的 Java 支程序。

第一部分"Java 语言"由 8 章组成。这 8 章通过许多实例详细介绍了整个 Java 的语法。

第 1 章介绍创建 Java 的由来以及构成这个革命性的编程环境的重要特性。

第 2 章介绍面向对象编程的基本概念以及与C++的详细比较。

第 3 章逐行分析第一个 Java 程序，在这里将学会一个可运行的 Java 程序的基本组成。

第 4 章论述 Java 的类型系统，它用来定义数据如何声明和保存。尽管这章大部分与 C 和 C++重复，但重要的不同之处均加以说明。

第 5 章介绍运算符。这一章内容也与 C 和C++参考手册中的相关内容很相似，但详细指出了几个细小的不同之处。

第 6 章介绍控制流，也是与 C 雷同的最后一章。本章中的许多小实例程序涵盖了分支迭代和跳转。

第 7 章进一步讨论 Java 的面向对象编程。本章用实例详细介绍了第 2 章中提及的理论概念。

第 8 章介绍包和接口——两个 C++不具备的 Java 概念。这个增加允许程序员管理类名空间并实现多继承的动态链接。

第二部分详细介绍标准的 Java 开发包中的所有类。

第 9 章讨论串操作。字符串操作是许多程序的核心;Java 的 String 类具有许多特性,这一章将详细讨论它们。

第 10 章介绍异常处理。Java 采用一种非常简明的方法管理异常。它彻底有别于大部分较旧式的难于维护和调试的程序系统。

第 11 章介绍线程和同步。多线程编程是 Java 最优秀的语言特性之一。多数的程序系统都不具备多线程能力,因此,它们难于使用和调试,更难于同步。

第 12 章介绍许多 Java 的数据结构,如栈、向量和枚举。同时本章还介绍了一些系统接口工具如数学函数及时间/日期访问函数。

第 13 章介绍输入/输出。Java 将 I/O 当作一组流类。这样,各种形式的输入/输出具有统一的接口。

第 14 章介绍网络。由于 Java 是服务于 Internet 的编程语言。因此网络类就尤其重要。本章介绍在 Internet 上创建客户/服务器程序的全部要素。作为超值回报,HTTP 服务程序也在本章中加以说明。

第 15 章介绍支程序和图形——它们使 Java 名扬四海。在本章中,讨论支程序和整个图形的 API,其中包括字体和着色图元。

第 16 章介绍抽象窗口工具集。这个工具集是一个重要的类库,用以实现窗口、菜单、按钮和滚动条等的跨平台抽象。本章详细介绍抽象窗口工具集的当前情况以及通过许多实例程序说明其 API。

第 17 章介绍图象。许多 Java 的动画支程序使用图象。本章全面介绍装载、动画和操纵图象的所有类。

第三部分介绍三个真实的 Java 支程序。

第 18 章介绍 Impression 支程序。SGI 公司的 Paul Haeberli 将其获专利的图形技术变成一个 Java 的支程序,使它成为网络上的一个有趣的支程序。使用这个支程序可使扫描的图象更具有真实感和震撼感。

第 19 章介绍 DynaDraw 支程序,在 Paul 的图形技术的引导下,并将它的另一个 SGI 程序移植到 Java。DynaDraw 模拟现实中的毛笔,可在屏幕上画出书法笔划。

第 20 章介绍 Magnet Poetry 支程序,它基于一个很好的想法:冰箱上贴有字条的小磁铁可以按任何方式排列。

附录 A 是一个简短的介绍,告诉读者如何将 Java 开发包下载并安装到机器上。几个有趣的开发环境和调试程序也在附录中说明。

最后是结尾部分"Java 的漫长发展史"。在这里介绍了我个人围绕着这个优秀的软件环境所遇到困惑和所做的努力。

目　录

第一部分　Java 语言

第二部分 Java 类库

第三部分 Java 支程序

第一部分　Java 语言

第 1 章　Java 的革命

对一个作者而言，去宣传任何新的革命性技术都是非常危险的。由于大多数技术的生命期都很短，今年的革命性技术常常成为明年的笑柄。大家还记得个人数字助理和手写识别吗？十九世纪早期的古老技术真实地改变了我们的生活。经过消费者短暂的抢购之后，个人数字助理最终还是消失了。把钱用在一项新技术如 Java 上的风险就好像你正自鸣得意而突然又来了一个更好的事情。有一件事在工业界是明确的：有许多优秀的人才正在做着或多或少相同的但具有相互竞争的事情。所以什么才能创造革命性技术？它不仅要更快，更小和更便宜，它还要影响我们做事的根基，它必须以一种简单的办法去解决根本性的问题。

现在我已经以极大热情预言一项正确的技术。我确信 Java 是近 20 年来计算机软件环境中的最有意义的进步之一。和超文本标注语言 HTML（一种在 World Wide Web 上实现静态文本的语言）一样重要，Java 是现今 Internet 上的热门话题，有三个根本性的因素使 Java 作为根本性的技术不同于现今已存在的语言，第一，它具有让任何人使用支程序的能力。支程序是一个小巧、安全、动态、跨平台、活跃，网络化的应用程序，这是有史以来的第一次。和 HTML 的任何其他方面一样，支程序可以很容易地根据客户的实际需要进行安全地配置和分布。第二，Java 是一种强有力的编程语言，它将面向对象的设计用一种简单和熟悉的语法根植在增强的、易于使用的环境中。这种语言允许广大的程序员去创建新的组件和新的支程序。第三，Java 具有一组丰富的对象类，使程序员可以对许多公共的系统功能如窗口操作、网络和输入/输出进行简明的抽象处理。这些类的核心是它们为各种公用系统接口提供了跨平台的抽象。让我们在以后更进一步地研究它们吧！

1.1　Java 支程序

现在，任何熟悉 HTML 并知道如何将带有他们家猫的照片的主页放到 Internet 上的用户都会使用 Java 的支程序。一个支程序实际上是一个小的 Java 程序，它动态地从网中下载，就像一幅图象、一个声音文件或一段录像剪辑。最重要的区别是支程序是一个智能化的的程序，而不仅是一个动画或多媒体文件。换句话说，支程序是一个能响应用户输入并动态改变的程序，而不是一个只会重复播放动画或声音的程序。

动态的多媒体 web 页面吸引了 Java 的大部分早期支持者。由于这些用户并不完全理解它更革命性的方面，Java 常被当作下载画面和与 Web 客户进行简单交互的一个常规技术。传统的多媒体公司，如 Adobe 和 MacroMedia 告诉用户他们的产品和 Java 具有相同的特性，面向

对象的软件公司，如 kaleida，Taligent 和 NeXT 都说他们的对象环境和 Java 在每一点上都同样创新，Microsoft 则声称他们以 10 年前的旧技术获得了胜利。他们中没有谁真正进一步了解大家将要学会和发现的 Java 编程环境的真实能力。

在图 1-1 中，股市行情收集支程序不仅将股票数据从右向左显示，并以客观存在实时方式从网络服务器上检索股票数据，及对数据进行算术运算，然后动态地以图形方式显示出来。

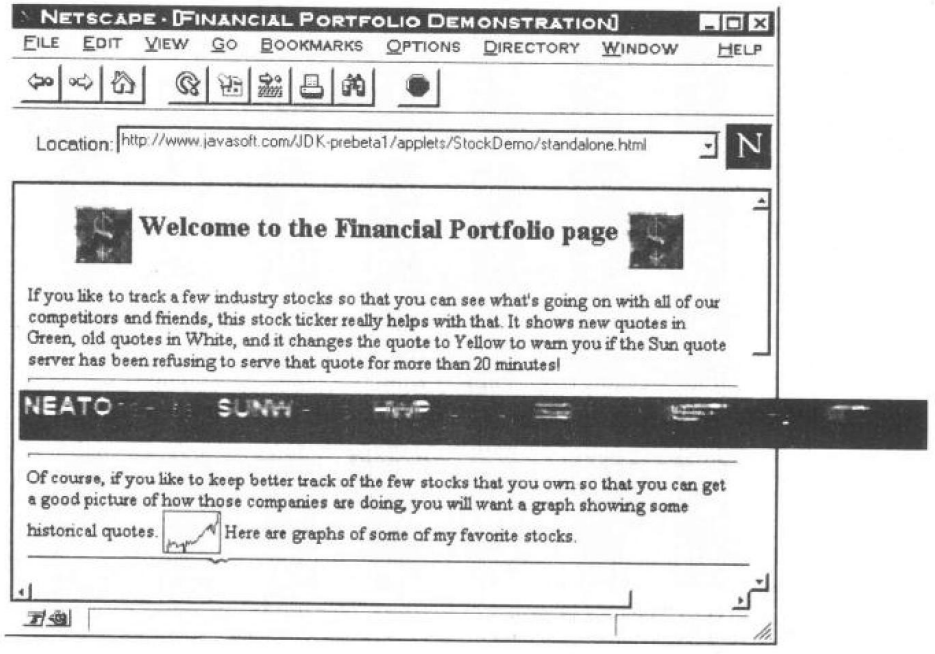


图 1-1 股市行情收集支程序

在图 1-2 中，我的好友以及 HotJava 的合作者 Jonathan Payne 的头像在重力的影响下漫游，反弹和摩擦。基本的动画软件，如 MacroMedia Shockwave 不能实现这种复杂的模拟。如果运行平台具有完整的编程环境，就可以实现各种交互控制。

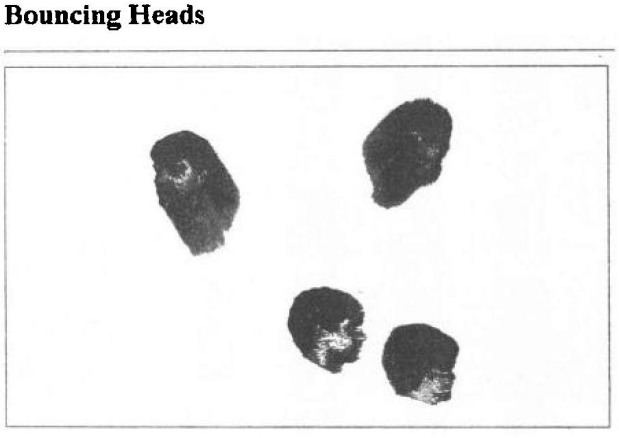
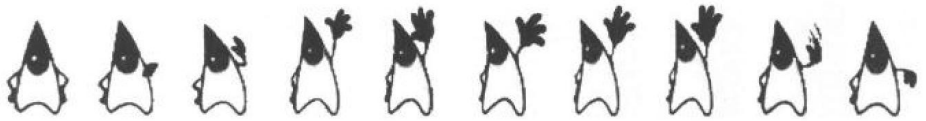


图 1-2 弹跳的头像

由于商标权的限制，Jonathan 的头像取代了我在 1992 年时喝的可乐罐图象。这是第一批

用 Java 语言编写的其中一个程序，其中由我和早期的 Java 开发人员 Chris Warth 编写的大部分代码至今没变。



Java 组的吉祥物 Duke™，最早是为个人数字助理原型研究小组设计的，它是个人数字助理的一个视觉表示，当你打开个人数字助理时，首先就会看到小 Duke 在向你善意地招手。这个图象安装在 Java 的动画支程序中，在 HotJava 中运行支程序可以看到它。

1.2　革命性的编程语言

Java 允许非程序员使用这些优秀的支程序，但是如何去创建呢？在屏幕上看到的每个活动的东西实际上都是用 Java 语言编写的对象。对象所表现的每一个动作都封装在 Java 类中。这些类用来定义对象。这样，Duke 动画中的每一帧都是一个对象，而所有的帧作为一个整体又是另一个对象。使 Duke 招手又是一个对象。在较复杂的弹跳头像支程序中，Jon 的漫游头像和招手的 Duke 支程序几乎一样。只不过描述它的类进行了增强，以使它能响应具有模拟重力和摩擦效果的对象。

在开发 Java 之前，为 Java 确定了符合现实需要的前提，它们是：

- 简单而又强大
- 安全
- 面向对象
- 可靠
- 交互式
- 结构无关
- 解释性并且高性能
- 易学

既使你一行 Java 程序也不编写，知道这些特性还是很有益处的。因为这些语言特性已溶在支程序的能力之中。

1.2.1　简单而又强大

一旦学会面向对象编程的基本概念，就可以很快学会 Java 编程。为了写出有效且满意的程序，你只需明白几个基本概念。Java 力图不增加什么超级的特点。有许多编程系统津津乐道，说它们可以有许多方法来做同样一件事情。如果一种语言过分暴露系统的内在工作机制，那么你就可以自由地以任何方式做任何事情。这确实对编程专家提供了很好的帮助，但这种自由也带来了复杂性。在 Java 中，对一个特定的任务，只有几个简单的方法。

这种简明的风格经常导致出无效和低廉的“描述语言”，但 Java 不是一种描述语言。描述语言限制了用户的创新，它假定用户所需的行为都封装在内置的对象中并且很少需要说明。Java 则以一种明晰的面向对象的方法帮助用户表达他们的想法，而不需将所有底层系统的内

部实现危险地暴露出来。

1.2.2 安全

现在对涉及到网上安全的问题，公众有很多的议论，特别是进入 Internet 之后，人人都认为在 Internet 上从事商业交易就像将你的信用卡号写在公共气车上一样不安全。病毒和特洛伊木马无处不在。甚至字处理文件都可能带有病毒，除非先用“word Prank”宏——一种扫描 Word 6.0 文档病毒的宏——对刚得到的电子邮件进行扫描，否则不要轻易打开邮件。这些问题的大部分都是由于那些老式的系统在设计时没有考虑 Internet 的安全性而产生的。有可能用户也能读到一些语言如“Safe-Tcl”和“Safe-Visual Basic”，它们在系统上增加一层保护层，这看起来像将语言弄成残废使其不再有用和可信。

Java 的一个核心设计原则是安全性。Java 从来没有任何不安全因素需要去掩盖。大多数引起系统“死机”的因素在 Java 程序中都不可能出现由于 Java 程序不能够调用全局函数并且不能随意访问系统资源，Java 采用自己独特的控制方法是其它系统不能访问的。

1.2.3 面向对象

令人吃惊的是，有许多老的程序语言也号称自己是面向对象的。这个词就和今天的“可上 Internet”一样到处都是。然而 Java 是从头开始设计的，它并不从任何其它的语言中派生，也不与任何一种语言向上兼容。

在白纸上可画最美的图画。因此，人们为 Java 选择了一条清晰的、实用的面向对象途径。通过对最近几十年面向对象软件环境认真借鉴——Java 的创造者们在两种观念中保持了平衡：即一切皆对象和万事不求人。Java 中的对象模型简单而又易扩充，而同时数字和其它简单类型作为高性能的非对象也被保存下来。

许多面向对象系统采用了僵便的、难于管理的对象层次树结构或采用完全的动态对象模型系统，但却损失了性能和可理解性。而 Java 又在简单类机制和动态接口模型中保持了平衡。熟悉面向对象的编程风格是学习如何用 Java 编程的最严重障碍，而不是一般的障碍。下面的章节将详细讨论对象和类。

1.2.4 强健性

也许大家认为在 Internet 上强调强健性是一种奢侈。我们都在 Internet 上运行过各种工具的测试版，为什么要关心一种强健的编程语言呢？难道这不是航天飞机程序员所关心的吗？其实不然，Java 严格限制程序员在几个关键范围内以便程序员在程序开发过程中尽早发现错误。同时，在 Java 中不用担心在别的语言中会出现的大量编程错误。Java 在用户输入代码的时候自动进行检查，而当运行程序时，它已经没有错误了。许多在运行时难于重复的不可跟踪的错误在 Java 中就根本不可能产生。在各种情况下程序均按预期方式执行是 Java 又一个主要特点。

今天的大多数程序执行失败无外乎两个原因：内存管理或异常。这是两个非常重要的问题，因为如果不妥善地处理内存管理和异常将不可能编写出一个实用的程序。在传统的环境中内存管理是一个难缠的任务。程序员必须跟踪所有的内存分配并确保不再使用时将其释放回系统。程序员经常忘记将分配的内存释放，或者更糟，他们试图释放其它部分的代码还在使用

的内存空间。在传统的环境中异常情况经常发生在除零溢出或文件未找到的时候，而且必须用复杂的、难以理解的结构加以管理。Java 利用一种称为“垃圾回收”的高级内存管理和集成的面向对象的异常处理机制彻底解决了这两个问题。这种无须关注琐事的机制极大地提高程序员的创造力。

由于 Java 在类型和声明上非常严格，因此，通常的错误在编译时都能发现，与那些必须在运行时才能发现动态错误的程序相比自然节省了许多时间。虽然 Java 在灵活性上比有些语言逊色，但这种付出是值得的。

1.2.5　交互式

Java 的设计目的是为了满足交互式网络化编程的现实需要，很多系统满足一个需求都很困难，更别说同时满足交互式和网络化这两个需求。Java 有几个很先进的特性，允许程序员编写的程序一次能处理很多任务，而同时又能跟踪事件的发生和什么时间发生。同时做多项任务取决于各部分同步的能力。如果没有有效工具，做这种事情会让程序员发疯。Java 具有多线程同步能力，使得构造交互式系统易如反掌。Java 的易于使用的多线程允许程序员去考虑需要实现的程序行为，而不用去处理行为之间的“事件循环”。

1.2.6　结构无关

比 PC 和 Mac 之间的战争更重要的是代码的生命期和可移植性。今天写的程序，不一定保证明天还能运行，甚至在同一台机器上是如此。操作系统的升级，处理器的升级以及核心系统资源的改变都会令你的程序不再能工作。Java 的设计者在 Java 语言和运行系统上做出了几个重要的决定，以达到“程序只要写一次，就可在任何时候任何地方永远地工作下去”。你只要多考虑如何编写一个好程序，Java 会让它们在 Macintosh、PC、UNIX 以及任何将来的平台上运行。

1.2.7　解释性并且高性能

Java 实现这种不寻常的跨平台技术是通过将代码编译成一种被称为 Java bytecode 的中间格式来达到的。这种中间格式可被任何具有 Java 运行器的系统解释执行。以前的系统在解决跨平台的问题上遇到了一个很大的问题：性能。其它平台无关的系统都是解释系统，如 BASIC、Tcl 和 perl，但它们性能受到了损失。而 Java 却可以在很低级的 CPU 上运行得很好。尽管 Java 是解释性的，但 Java 的 bytecode 经过精心设计，因此可直接翻译成机器的本地代码以实现高性能。Java 的“即时”优化没有损失平台无关系统的任何优点。“高性能、跨平台”已不再是个梦想。

注意：Java 的编译器将源代码编译成 bytecode。取名叫 bytecode 是因为 Java 程序在编译后变成一串虚拟机的执行指令字节。我们说虚拟机，是因为有些本地编译语言如 C++将代码编译成某种机器的具体的真实指令。例如 Microsoft C++环境将程序编译成可在 Intel x86 芯片中直接执行的一串指令流。而 Java 解释器则根据具体机器的不同，逐条检查每个指令，然后相应改变虚拟机的状态。

1.2.8 易学

所有的这些特性交织起来构成了一个优秀的编程语言。尽管 Java 比描述性语言要复杂得多，但还是比那些五脏俱全的编程语言要简单的多。在编程的每一环节，都会感受到没有错误的鼓舞。语言的特性鼓励程序员采用一种优秀的编程风格以自然的方式去编写程序。由于对象系统具有强制性和简易性，你会很快熟悉面向对象的编程风格，并从中获益非浅。

1.3 丰富的对象环境

Java 环境不仅是一个编程环境。其中有许多现实世界中要处理的抽象类。这些类丰富了 Internet 上的 Java。能使成千上万的开发人员在网络上开发 Java 对象的关键原因是因为 Java 有一组能不断满足他们需要的公共类。对我来说，用 Java 编程能充分领略游戏的乐趣，但这还不足以使一个新语言获得成功。我认为 Java 内置抽象类以实现真正跨平台是该语言成功的关键。我确信人们还将继续使用 C++和 Objective-C，因为它们还有可能在大多数系统上运行的类库。有些环境如 MFC/COM、NeXTStep、Motif 和 OpenDoc 对与之兼容的平台仍然很有用，但 Internet 却是今天最重要的平台。

1.3.1 Internet 类

Internet 是分别由 Microsoft/Intel PCs、Apple Macintoshes 和 UNIX 工作站共同组成的，它们在全世界上站点上分别占有 60%、23%和 17%的份额，这些系统在处理网络、窗口、图形和输入/输出上都有自己独特的环境，没有谁具有处理 Internet 协议的内置类库。Java 类库通过提供一个单一的 Internet 协议模型解决了这个问题。

1. *TCP/IP 网络*

处理 Internet 的基本协议封装在几个简单的类中。ftp、http、nntp、smtp 以及低级网络接口和命名接口也包括其中。这样使得程序员无须关心网络协议底层细节就可以完成许多复杂的网络功能。Java 的简易性是显而易见的：一旦复杂和负担被化解，Internet 协议就变成简单易用的 Java 对象。事实上，如果你知道 Internet 协议的底层细节，就可以很容易地用 Java 实现客户或服务器。在第 14 章介绍了一个简单 Java 程序，它可以实现基本的 Web http 服务程序功能。

2. *WWW 和 HTML*

Java 是为满足交互式地获取嵌入在 HTML 页上的信息而设计的。由于 Java 的成功取决于 World Wide Web 以及 Java 的支程序要嵌入到扩充的 HTML 页中，因此 Java 的类特别适合支持这些协议和格式。而 Java 中的基本串操作类又适合字符串处理，这正是基于 HTML 程序的一部分工作。

3. *发布程序*

Java 的类可以很容易地通过网络传送。由于文件系统和网络流具有单一的接口，因此通

过网络装载一个类就显得很平常。这就是为什么 Java 支程序可以在 web 浏览器中被装载。将一个程序发布或进行局部升级也很方便。这样就解决了长期以来令人烦恼的版本控制问题。现在甚至可以升级许多系统赖以生存的核心类而不用破坏现有的类。Java 解决了 C++中"易碎的基本类"问题也可算是 Java 最重要的特点之一。

1.3.2 核心类

Java 从开始而不是事后就从满足网络化、面向对象、多线程系统的需求而设计的。由于 Java 是一个逐步扩充的系统，因此有许多 Java 环境需要的核心类。许多核心语言概念在 Java 中被当作类实现，如串操作、多线程和异常处理。Java 编译器用的许多数据结构被当作工具类进行抽象并提供给系统的其它部分。Java 在运行时输入/输出操作被抽象成 I/O 流类。下面我们要在这里简要介绍一下这些方面的内容，详细的内容将在后面的章节中说明。

1. 语言

在最底层，Java 需要一组类才能工作。这些类提供了对 Java 语言的支持。支持正文处理的操作封装在 String 类中。所有对字符串的操作都通过 String 类进行。拼接、查找和字符串替代操作都封装在这个类中。多线程由 Thread 类处理。使用 Thread 类，可以直接操纵处理状态，如优先级、运行、睡眠或挂起。异常处理由 Exception 和 Error 类负责。必须认真处理异常，因为 Java 强化所有声明的方法所产生的异常。许多基本的函数都可能引发异常，因此必须在代码中仔细处理这些异常。三角函数如 sin、cos、tag，在传统语言中一般为缺省的全局函数，在 Java 中被封装到 Math 类。

刚开始也许觉得这些抽象有点别扭，但以后会欣赏这种将各种功能分门别类组织到类中的方法。你还会注意到这种方法对将要运行其上的平台提供了一种简单的对象。

2. 工具

Java 的工具包具有一个完备的复杂数据结构和相应的方法。这对实现复杂的支程序和应用程序是无价的。公共数据结构如 hash 表、栈和可变数组以简明的类加以提供(如果你现在不知道这些数据结构是什么也没有关系)。经过精心设计的可视化接口以简便的方式实现动态的对象并将内部状态显示在屏幕上。这种模型一视图一控制器结构首次在 Xerox PARC 上得以实现。访问系统底层的函数如日期和时间被归到 Date 类中。我们将在以后详细讨论它们。

3. 输入/输出

Java 采用一种单一的流模型来处理各种形式的输入/输出。无论是处理文件系统，还是网络或输入设备，用户程序只须使用掩盖了所有细节的而又简单的 InputStream 和 OutputStream 对象即可。

4. 低级网络

(Java 提供了一组由 Berkeley UNIX 派生的低级网络插座抽象类)。这个包封装了与其它 Internet 机器插座进行地址和 I/O 流操作的类。这些接口处于 Java 核心类中就足以说明 Java 与 Internet 的关系。这就是为什么 Java 能从网络上安全地装载它的类。

5. 抽象图形用户接口类

也许 Java 中最困难的类是“AWT”(抽象窗口工具集)。这些类实现了 Xerox PARC'80、Macintosh'84、X/Motif'88 和 Windows'95 的图形用户接口,而这些系统是当今桌上型计算机的常用图形系统。对窗口、滚动条、按钮、菜单等的彻底抽象才使 Java 真正成为与平台无关的系统。由于这个包里的类太多,我们将在以后的章节中加以讨论。

1.4 下一步

现在你已经明白了 Java 编程环境的优点和特性。我确信大家一定急于想编写自己的支程序。下面我们尽快开始。在编写网络上的支程序前,我还要讲一讲面向对象的编程。并且要讨论为什么 Java 比 C++更适合交互式网络编程。然后我们再花几章介绍 Java 语言,并试着使用几个简单的类。请耐心,先学会编写小的程序将更有助于今后获得支程序编程竞赛的大奖。熟练的 C++程序员可以跳过这些章节。一旦你熟悉 Java 类并可以控制程序流,那么就可以开始学习类库。我们还要分析从网上下载下来的优秀支程序源码。一旦学会这些实例,你将会变成一个令世人惊叹的 Java 支程序专家。