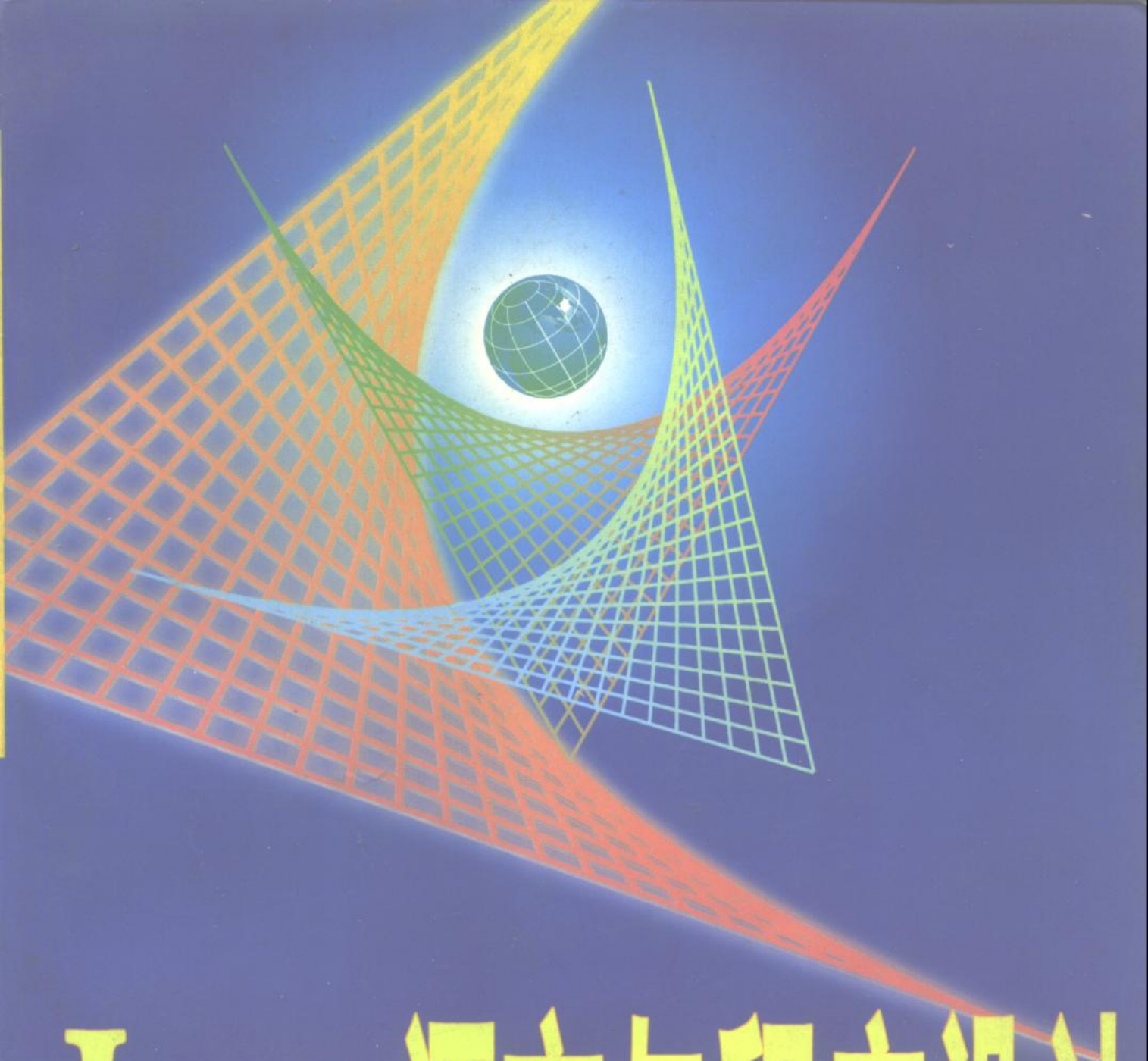




Java语言与程序设计

杨华中 贾耀炜 编著

人民邮电出版社



内容提要

本书通过丰富的实例,由浅入深地、全面系统地介绍了运用 Java 语言进行程序设计的方法和技巧。本书除介绍 Java 语言的基本概念、基本语法之外,还介绍了 Java 语言的系统资源、异常、多线程、用户界面的制作方法(包括声音、图形、图象、动画等)、网络、安全管理器、本地过程的使用方法和 Java 的开发环境。另外,本书还简要地介绍了 Java 语言在数据库方面的应用等新的进展。

本书内容全面、范例程序丰富、图象逼真、语言浅显易懂,非常适合于大中专院校的学生、Internet 用户和广大的计算机网络爱好者。

JS363 / 13

Java 语言与程序设计

◆ 杨华中 贾耀炜 编著

责任编辑 滑 玉

◆ 人民邮电出版社出版发行

北京崇文区夕照寺街 14 号

北京顺义向阳印刷厂印刷

新华书店总店北京发行所经销

◆ 开本:787×1092 1/16

印张:27.75

字数:698 千字

1997 年 2 月第 1 版

印数:1—8 000 册

1997 年 2 月北京第 1 次印刷

ISBN7-115-06387-7/TP·406

定价:38.00 元

Java 语言是由 Sun 公司于 1995 年推出的、从 C++ 语言衍生而来的一种编程语言,在推出后短短半年多时间内,Java 语言就成为了新的网络化软件的命脉,被人们誉为 Internet 和环球信息网(World Wide Web)上的“世界语”,给软件行业带来了一次革命。

Java 语言具有面向对象的各种优势,而且抛弃了 C++ 语言中那些难理解、较少使用的部分,因而易学易用;Java 语言的分散式界面使 Java 程序可以很容易地同 TCP/IP 结合,从而使跨网络文件的存取就如同存取本地文件一样方便,同时还使 Java 程序能从网络中搜索所需的软件并执行之;Java 语言的多线程界面改变了传统的环球信息网浏览器的设计模式,使实时网络交互成为现实;Java 语言的安全性机制使电脑可免遭病毒侵害、并有效防止存取未授权的数据对象。Java 语言为程序设计提供了一个十分友好的网络环境,其字节码技术使 Java 程序能适应多元化的 CPU 和操作系统结构。

随着信息社会的发展,Java 语言将越来越重要,它改变了网络信息处理的环境、应用系统及人机交互的方式,打破了传统的应用软件开发和销售的模式,实现了一个独立于硬件平台和操作系统的程序设计环境。Java 的出现为 Internet 和环球信息网开辟了崭新的时代。在 IBM 和 Microsoft 等公司的支持下,Sun 已于最近成立了 Java 专业子公司 Javasoft,该子公司致力于 Java 在手持移动电话、寻呼机和便携式信息终端等通信设备中的应用。Sun 公司还计划在今年出售 Java 微处理芯片,进一步拓展 Java 在信息社会中的各种应用。

随着我国教育科研网、邮电通信网和医学科学网的建立和迅速发展,Internet 和环球信息网已经步入寻常百姓之家,国内也出现了一批 Java 迷。我们坚信,生活在信息时代的人们对信息的需求将迅速膨胀,而 Java 将成为人们获取、处理信息的有力工具。

为此,我们结合自己的实践经验编写了本书,通过丰富的实例系统地介绍 Java 语言。如果读者在阅读完本书之后能使用 Java 语言来获取、处理各种网络信息,那么,也必定能在信息社会日益激烈的竞争中获得“回首射雕处,千里暮云平”的美好心情。这也是我们最大的欣慰。

在本书的编写过程中,曾得到清华大学电子工程系汪蕙、刘润生等教授的热心参与和大力支持,没有他们,本书将只能是镜中花月。

作 者

1996 年 8 月,于清华园

第一章 引 言	1
1.1 Java 的发展历程	1
1.2 Java 语言简介	2
1.2.1 Java 的特点	3
1.2.2 Java 开发工具包(JDK)的安装	6
1.2.3 两个简单的例子	8
第二章 Java 语言基础	11
2.1 面向对象的程序设计概念	11
2.1.1 对象(Object)	11
2.1.2 消息(Message)	13
2.1.3 类(Class)	14
2.1.4 继承(Inheritance)	15
2.2 Java 的基本语句	16
2.2.1 变量和数据类型	16
2.2.2 运算符(Operator)	18
2.2.3 表达式	20
2.2.4 流程控制语句	21
2.2.5 数组和字符串	24
2.2.6 示例	26
2.3 对象、类和构架(Interface)	29
2.3.1 对象	29
2.3.2 类	33
2.3.3 子类、父类和继承	48
2.3.4 构架	54
2.3.5 创建和使用类组	57
2.4 类 String 和类 StringBuffer	60
2.4.1 创建类 String 对象和类 StringBuffer 对象	60
2.4.2 存取过程	61
2.4.3 修改 StringBuffer 对象	63
2.4.4 String 对象与其它类型对象的相互转换	64
2.4.5 String 对象与 Java 编译器	65
2.4.6 Java 的类 String 对象为首类对象	66

2.5 程序运行环境	67
2.5.1 设置和使用属性	68
2.5.2 应用程序的命令行参数	69
2.5.3 小应用程序参数	74
2.6 系统资源	74
2.6.1 类 System 的使用方法	74
2.6.2 标准输入流、输出流	75
2.6.3 系统属性	76
2.6.4 强迫终结过程调用与无用存储单元回收	79
2.6.5 加载动态库	80
2.6.6 杂类过程	80
2.6.7 使用与系统相关的资源	83
2.7 利用异常处理错误	83
2.7.1 什么是异常	83
2.7.2 示例	88
2.7.3 Java 的捕捉或指定要求	90
2.7.4 捕捉异常	91
2.7.5 抛出异常	98
2.7.6 有关运行态异常的争论	102
2.8 线程	103
2.8.1 基本概念	103
2.8.2 线程的属性	105
2.8.3 多线程程序	122
2.8.4 小结	134
2.9 输入、输出流	135
2.9.1 概述	135
2.9.2 输入、输出流的基本用法	138
2.9.3 过滤流(Filtered Stream)	150
2.9.4 随机存取文件	156
第三章 小应用程序	161
3.1 小应用程序初步	161
3.1.1 一个简单的小应用程序 SimpleApplet	161
3.1.2 小应用程序的生命周期	163
3.1.3 与生命周期相关的过程	164
3.1.4 绘制和事件处理过程	165
3.1.5 添加用户界面(UI)元件的过程	165
3.1.6 添加小应用程序至 HTML 页面	168
3.2 小应用程序中的线程	171
3.2.1 小应用程序的线程	171

3.2.2 创建和使用线程	173
3.3 创建小应用程序用户界面	182
3.3.1 创建一个图形用户界面	182
3.3.2 播放声音	183
3.3.3 定义和使用小应用程序参数	185
3.3.4 获取系统属性	189
3.3.5 显示简短的状态字符串	190
3.3.6 在标准输出和错误流上显示诊断信息	190
3.4 与其它程序通信	191
3.4.1 向在同一页面内的其它小应用程序发送消息	191
3.4.2 与浏览器通信	197
3.4.3 使用服务器方的应用程序进行通信	200
3.5 小应用程序的能力和限制	216
3.5.1 安全性限制	216
3.5.2 小应用程序的能力	219
3.6 完成小应用程序的一般步骤	220
3.6.1 发布小应用程序之前的工作	220
3.6.2 完美的小应用程序作品	221
3.7 小结	222
第四章 创建用户界面	225
4.1 概述	226
4.1.1 常用 AWT 元件	226
4.1.2 其它的 AWT 类	230
4.1.3 元件层次、绘制与事件处理	231
4.2 元件,GUI 的构建模块	244
4.2.1 AWT 元件的基本特征	245
4.2.2 更深入的内容	280
4.3 在容器中布置元件	281
4.3.1 AWT 系统提供的布局管理器	281
4.3.2 创建用户的布局管理器	294
4.3.3 不使用布局管理器(按绝对坐标放置元件)	299
4.4 图形、文本、图象及动画制作	300
4.4.1 概述	300
4.4.2 图形与文本	301
4.4.3 图象	318
4.4.4 制作动画	341
第五章 网络和安全管理器	365
5.1 网络总览	365

5.1.1 通信协议与通信端口	365
5.1.2 有关 Java 网络的知识	367
5.2 URL 及其使用方法	368
5.2.1 什么是 URL	368
5.2.2 创建一个 URL	369
5.2.3 分析一个 URL	371
5.2.4 直接从一个 URL 读入	373
5.2.5 连接至一个 URL	375
5.2.6 对 URLConnection 进行读写操作	376
5.3 套接字(Socket)及其使用方法	379
5.3.1 什么是套接字(Socket)	380
5.3.2 编写客户程序	380
5.3.3 编写服务器程序	383
5.4 数据报(Datagram)及其使用方法	386
5.4.1 什么是数据报	386
5.4.2 编写数据报客户和服务端	387
5.5 提供自己的安全管理器	393
5.5.1 安全管理器的作用	393
5.5.2 编写安全管理器的方法	394
5.5.3 安装安全管理器的方法	396
5.5.4 决定覆盖 SecurityManager 过程	398
第六章 Java 开发工具	401
6.1 Java 编译器:javac	401
6.2 Java 解释器:java	402
6.3 Java 调试器:jdb	405
6.4 C 头文件和 Stubs 文件生成器:javah	407
6.5 Java 类文件反编译器:javap	407
6.6 Java API 文档生成器:javadoc	410
6.7 Java 小应用程序观览器:appletviewer	413
第七章 在 Java 程序中集成本地过程	415
第八章 Java 语言与数据库	421
8.1 JDBC 简介	421
8.2 JDK 1.1 预览	424
附 录	427
附录 1:Java 关键字	427
附录 2:Java 类层次结构	427
参考资料	435

第一章

引言

Java 语言(简称 Java)的诞生不仅对计算机软件产业带来了一次革命,也对人们的日常生活产生了深刻的影响。Java 的推出绝不是对 C++ 语言的升级,Java 的平台无关性使之成为 Internet 上获取、处理信息的最佳程序设计语言。可以预计,Java 将不仅是未来新型操作系统的核心,也是环球信息网(World Wide Web,简称 WWW)上的“世界语”。Java 所带来的革命意味着计算机已成为 Internet 和环球信息网的外设,Java 营造了一个优者生存的开放环境。回顾 Internet 和环球信息网的发展历程,我们不难发现,网络已经成为一个以指数级增长的电子生态系统,而 Java 则扮演着生命领域中 DNA 的角色。因此,Java 当之无愧地被纽约时报评为 1995 年的十大科技成果之一,这也是计算机产业界唯一入选的成果。难怪美国一位著名的经济学家对 Java 给予了极高的评价,并呼吁将 Java 作为一项重大发明载入科技史册。

1.1 Java 的发展历程

在九十年代的计算机产业中,Microsoft 的视窗系统和 Netscape 公司的网络浏览器都曾经名噪一时,而 Java 则在顷刻之间使它们黯然失色。人们已经看到了 Java 今天的巨大成就,同时也在勾画其明日的辉煌。但是,Java 的发展却经历了一个曲折的历程。

Java 诞生于 1995 年,是美国卡内基—梅隆(Carnegie Mellon)的大学本科毕业生 James Gosling 的杰作,但 Java 的发展历程可以追溯至 1990 年。当时 Sun 公司计划开发一套名为 Oak 的程序设计语言,其目的是用于消费类电子和家用电器的微处理器。经过开发人员 18 个月

的努力,Oak 在 1992 年的秋天有了很大的发展,并开始了他们的绿色计划,包括绿色操作系统、绿色用户界面和绿色硬件平台等。后来,由于商标命名上的原因,Oak 被更名为 Java。Sun 公司的绿色计划推出了一种称为“*7”的硬件,一种类似于 PDA 的器件,当时被 Gosling 称为“掌上遥控器”。“*7”模型成型之前,绿色计划已数易其稿,Sun 公司为了从绿色计划中取得经济效益,于 1993 年初成立了专门的绿色小组,即当时的 FirstPerson 公司,Gosling 是绿色小组的负责人。由于“*7”的外形较小,因此,绿色小组花费了巨大的心血,使代码的小体积和高效率完美地统一起来,这些工作是日后 Java 能如一轮红日冉冉升起的基础。

绿色小组的工作是有成效的,在技术上取得了许多突破,并以其先进的技术投标 Time-Warner 的设备顶盒(set-top box)项目。尽管被告知他们拥有最好的技术,Sun 公司还是未能中标。后来,Gosling 将未中标的原因归结为商业政治。直到 1994 年初,FirstPerson 公司还致力于设备顶盒的研究,但是,由于交互电视并未形成主流市场,FirstPerson 公司只好解散。而 Java 也几乎随着绿色计划的解体而胎死腹中。

正当原 FirstPerson 公司的大部分人员投入到 Sun Interactive 并从事数字视频服务器的工作时,环球信息网已经出现,并于 1994 年中期变得庞大起来,Gosling 立即意识到他们可以利用 Java 的成果建立一个真正好的网络浏览器。该浏览器具有结构中立、实时、安全可靠等特点。在 Gosling、Naughton、Jonathan 等人的共同努力下,于 1994 年秋推出了 WebRunner,这是一个由 Java 语言开发的网络浏览器,是 HotJava 的早期实体,这个原型系统展示了 Java 可能带来的广阔的市场前景,并进一步增添了 Sun 公司总裁 Scott McNealy 对 Java 所抱的坚定的信念,为 Java 进军环球信息网奠定了基础。

Sun 公司于 1995 年 5 月 23 日发布了网络浏览器 HotJava,它立即在环球信息网上激起千层波澜,并在社会各界引起了轰动效应,使人们意识到一场软件革命即将到来。

从 Java 的发展历程可以看到,是环球信息网使 Java 脱胎换骨;而 Java 的出现又为 Internet 和环球信息网开辟了一个崭新的时代。

在 Java 推出的短短一年多的时间内,在全球范围类掀起了一股强劲的 Java 热潮。美国、日本和欧洲诸国相继出现了许多 Java 增值开发商,诸如蓝色巨人 IBM、软件霸主 Microsoft 等都竞相购买 Java 的使用权。当然,神州大地也不例外,在 Cernet 和 ChinaNet 上 Java 一词已被炒得炙手可热,在高校中出现了许多 Java 迷。在香港地区,1996 年 4 月就举行了全岛范围的 Java 杯比赛,并在计算机界出现了学习 Java 的热潮。

总之,Java 的诞生必将对整个计算机产业产生深远的影响,对传统的计算机模型提出新的挑战,并不断改变人们的日常生活模式。当然,Java 也为生活在信息社会中的每一位成员带来了千载难逢的机遇。

1.2 Java 语言简介

Java 是 Internet 上最佳的程序设计语言,是新型的网络化软件的命脉,Java 改变了网络信息处理的环境、应用系统及人机交互的方式,打破了传统的应用软件开发和销售的模式;Java 实现了一个独立于硬件平台和操作系统的程序设计环境,能使 Internet 和环球信息网发挥新的巨大潜力。

1.2.1 Java 的特点

Java 语言的优点主要表现在:简单、面向对象、自动的内存管理、分布式、稳定性、安全性、解释执行、结构中立、可移植性、高效能、多线程、可靠的异常模型和具有动态特性等方面。

一、简单

Sun 公司开发 Java 的目的是希望建立一种真正简单的系统,它允许使用者轻易地设计程序而不需要像现今的标准程序一样接受深奥的训练。目前大多数程序设计师都采用 C 语言,而面向对象程序设计师则大多使用 C++ 语言。Java 与 C++ 语言非常相近,这使得 Java 具有面向对象的语言优势,也更容易被人们理解。Java 删除了 C++ 语言中许多极少被使用、不容易理解和令人混淆的 C 语言功能,因为这些功能在使用过程中往往只能带来麻烦而非效益。被删除的 C++ 语言功能主要包括运算符重载(operator overloading)、多重继承(inheritance)以及广泛的自动强迫同型(automatic coercions)。Java 语言也提供重载函数,不过它重载的对象是函数(method)而非变量或运算符。

Java 所具有的自动内存管理机制也大大简化了 Java 程序设计。

二、面向对象

“面向对象”是被计算机产业界过度渲染的术语之一。不过,面向对象是一项很强的设计功能,因为它促成明确的接口定义,并允许建立可重复使用的软件模块。

简单地说,面向对象设计是一种以数据(对象)及其接口为重心的程序设计技术。以木匠制作椅子为例:一位“面向对象”的木匠最重视的是他想制作的椅子,其次才是他用来制做椅子的工具。相反,一位“非面向对象”的木匠则把心思放在工具上。面向对象设计也是定义程序模块如何“即插即用”的机制。

Java 的面向对象设施实质上就是 C++ 语言,并包含 Objective C 的一些延伸。Java 提供了简单的类机制和动态的构架模型。对象中封装了它的状态变量和过程(又称方法),很好地实现了模块化和信息隐藏;而类则提供了一类对象的原型,通过继承和重载机制,子类可以使用或重新定义父类或超类所提供过程,从而既实现了代码的复用,又提供了一种动态的解决方案。

三、自动的内存管理

Java 的自动无用存储单元收集(auto garbage collection)实现了内存的自动管理,因此简化了 Java 程序工作,但同时也让系统变得复杂功能。储存管理(storage management)是使 C 与 C++ 应用程序变得复杂的常见的一项原因。Java 语言的自动无用存储单元收集功能(周期性地释放未被使用的内存)不仅简化了程序设计工作,而且能大幅度减少小错误(bugs)数量,提高 Java 程序的稳定性和可靠性。

四、分布式

Java 拥有广泛的例程库(routine library),能轻易地处理 TCP/IP 协议,例如 HTTP 与 FTP 等。这使得在 Java 中比在 C 或 C++ 中更容易创建网络连接。Java 应用程序可以借助通

用资源定位器 URL,在网络中开启和存取对象,而且,这类操作和存取本地(local)文件系统一样简单。

五、稳定性好

Java 强调在设计初期即检查可能存在的问题,其后则执行动态(runtime)检查,并排除容易出现错误的条件。

强类型(strongly typed)语言(例如 C++ 语言)的优点是允许在编译时进行深入的检查,以便提前发现错误。不幸的是,C++ 承袭了 C 在编译检查(compile-time checking)时的一些漏洞;C 的检查较为松散,特别是在方法或过程声明方面。在 Java 中要求声明方法或过程,但并不支持 C 风格的隐性声明。

链接器(linker)链接类型系统并重复执行许多已由编译器完成的类型检查,以避免出现版本不匹配问题。

Java 与 C/C++ 之间最大的不同点之一在于 Java 拥有一种指针(pointer)模型,能排除发生内存被覆盖和毁损数据的可能性。Java 不采用指针算术法,而是提供真正的阵列。这允许程序执行下标检查;再者,它也不可能发生借由对象类型转换将一个任意整数转成指针的情形。

另外,Java 的自动内存管理功能在简化程序设计的同时,还极大地减少了 Java 程序的动态运行错误(bug),增加了程序的稳定性。因为,Java 程序员很少需要处理内存问题;Java 中也不存在指针,Java 程序也就不可能意外覆盖一片内存缓冲区;同时,Java 程序也不可能非法访问内存。

六、安全性高

Java 的设计目的是提供一个用于网络/分布式的运算环境。为此,Java 非常强调安全性,以确保建立无病毒、且不会被病毒侵入的系统。Java 的验证技术是以公钥(public-key)加密法为基础,而且从环境变量、类加载器、文件系统、网络资源和名字空间等诸多方面施加了安全管理措施。

另外,由于 Java 不支持指针操作,所有对内存的访问都必须通过对象的实例变量来实现,从而有效地防止了程序原采用诸如“特洛伊木马”等手段来访问对象的私有成员的欺骗行为,增加了安全可靠。

七、解释执行

Java 解释器(interpreter)可以直接在任何已移植解释器的机器上解释、执行 Java 字节代码,不需存储。再者,由于其链接比较倾向于逐步增量与轻量过程,因此开发程序的速度更快。

由于编译期间的信息属于字节代码流的一部份,这不仅使链接过程更加简单,而且,Java 程序可以在运行期间携带更多的信息,便于排除 Java 程序在执行过程中所发生的各种错误。

八、结构中立性

Java 是网络上的“世界语”,Java 应用程序能够在网络上任何地方执行,其编译器会产生一种具备结构中立性(architecture neutral)的目标文件格式。编译后的程序码可以在提供 Java 运行系统的多种不同处理器上面执行。

这不仅对网络应用很有帮助,而且也很适合单一系统软件流通。在目前的个人电脑市场

上,应用程序开发者必须为他们的程序分别编写 IBM PC 和 Apple Macintosh 兼容的版本。现在,PC 市场正分散成许多 CPU 结构(透过 Windows/NT),而 Apple 则从 68000 转向 PowerPC,这些事实使得我们几乎不可能采用以前的手段设计出能在所有平台上执行的软件。而 Java 的结构中立性允许同一版本的应用程序在所有平台执行,因此,Java 可以将目前软件行业中从事软件移植工作的 85% 的劳动力解放出来,让他们从事别的更具创造性的程序设计工作。

九、可移植性

结构中立性是确保程序可移植的最重要部份,不过,还必须具备很多别的条件。和 C 语言、C++ 语言不同,Java 规范中并无任何依赖于特定结构的陈述存在。它指定基本数据类型的大小,以及其算术运算元的执行行为。例如,"int" 代表一个有符号的二进制补码 32 比特整数,而"float"代表一个 32 比特 IEEE 754 浮点数。这些选择在今天的环境很适用,因为几乎所有 CPU 都具备这些特性。Java 的程序库定义了一些可移植的程序接口。例如,它包括一个抽象的 Windows 类,并且提供了类在 Unix、Windows 和 Macintosh 平台上的实现。另外,Java 系统本身具备相当好的移植性。编译器以 Java 写成,而 runtime 程序使用 ANSI C,并有一个实质上与 POSIX 相容的移植疆界(portability boundary)。Java 的这些特征使之具有很好的可移植性。

十、高性能

Java 比 BASIC、Lisp、TCL 和 Smalltalk 等解释执行语言具有更高的性能,因为,Java 字节码的设计使之能够很容易地转换为特定 CPU 的机器码,从而到达较高的性能。

十一、多线程

Java 的多线程(multithreading)机制使程序可以并行运行。Java 还拥有一组复杂的同步化基本单元,它们是以广泛使用的 C. A. R. Hoare 监视器与条件变量图为基础的。同步机制保证了对共享数据的正确操作。多线程使程序设计者可以用不同的线程分别实现各种不同的行为,而不需要采用全局的事件循环机制,因此,使用 Java 语言可以非常轻松地实现网络上的实时交互行为。

十二、可靠的异常模型

Java 的异常模型允许接管所有不正常的程序结束,并继续执行未出错的部分。Java 的异常模型使程序的主流逻辑变得更加清晰明了,并能简化错误处理工作。

十三、动态性

Java 是一种比 C 或 C++ 更具动态特性的语言。它在设计上强调为进化中的运算环境提供支援。可以在 Java 的类库中加入新的过程和实例变量而不会影响用户程序的执行。Java 的动态性还表现在它通过构架来支持多重继承,因而比严格的类继承更为灵活,也更利于扩展。

对象类有一种运行(runtime)表示法:它有一个称为 Class 的类,其内容包括运行类(runtime class)定义。在 C 或 C++ 程序中,如果有一个指针指向一个对象但又不知道该对象的类型,那么你就没有办法找出它。然而,在 Java 中,根据运行类的信息辨认出对象的类型是很容易

易的。因为,在编译和运行时都会检查数据类型转换,所以你在 Java 中可以信任这种转换。相反,C 与 C++ 语言的编译器则只是想当然地相信你已做了正确的处理。

1.2.2 Java 开发工具包(JDK)的安装

Java 开发工具可以从 `ftp.javasoft.com` 节点取得。

一、UNIX 用户

JDK 1.0.2 的 Solaris 版本仅支持 Sun Solaris 2.3、2.4、2.5 基于 SPARC 的机器,以及 Solaris 2.5 x86 机型机器,你可以从 Internet 上取得相应的压缩的软件包:

`JDK-1.0.2-Solaris2-sparc.tar.Z`

或 `JDK-1.0.2-Solaris2-x86.tar.Z1`

1. 解压缩

将下载得到的压缩软件包进行解压缩。你需要执行下述命令:

```
/usr/bin/zcat JDK-1.0.2-Solaris2-sparc.tar.Z | tar xf -
```

该命令将创建一个目录 `java/`,其中包含几个目录和文件,包括 `src.zip` 和 `lib/classes.zip`。注意:一定不要解压缩 `classes.zip` 文件。但如果你对 Java 核心类的原始代码感兴趣的话,可以将 `src.zip` 解压缩,并将文件移至你所希望的目录下。

2. 更新环境变量

将环境变量 `CLASSPATH` 设置为正确的类路径。对于 C-Shell 用户,你需要执行命令:

```
setenv CLASSPATH . : $HOME/java/lib/classes.zip
```

以确保使用的是 JDK 1.0.2 版本正确的类。对于标准 Shell 用户,你需要采用 `set` 和 `export` 来设置环境变量,这方面的内容请参阅 UNIX 的用户手册。

3. 测试

将 `java/bin` 加到你的路径中,然后进入一个包含 `.html` 文件的目录,例如:

```
cd java/demo/TicTacToe
```

并按下述方式运行 `appletviewer`,

```
appletviewer example1.html
```

二、Windows 用户

JDK 1.0.2 的 Windows 版本仅支持 Microsoft Windows NT/intel 和 Windows 95。你可以从 Internet 上下载下述的压缩软件包:

`JDK-1.0.2-win32-x86.exe`

1. 解压缩

将压缩软件包移至将要安装 Java 的目录下,由于该压缩软件包为一个自解的程序,所以直接执行就可以了。解压缩后,将创建 java 父目录以及相应的子目录和文件,包括 src.zip 和 lib\classes.zip。注意:一定不要解压缩 classes.zip 文件。但如果你想看 JDK 库的源代码,可以解压缩 src.zip,但你应该用一个可以处理长文件名的 unzip 程序进行解压缩。

2. 更新环境变量

将 java\bin 目录加入到路径中,并更新 CLASSPATH 环境变量,例如:

```
set CLASSPATH=.;c:\java\lib\classes.zip
```

3. 测试

进入一个包含 .html 文件的目录,例如:

```
cd java\demo\TicTacToe
```

然后,按下述方式运行 appletviewer,

```
appletviewer example1.html
```

4. 排错

(1) 如果在运行中发生下述错误:

```
net.socketException: errno=10047
```

或者

```
Unsupported version of Windows Socket API
```

请检查你安装的是哪一种 TCP/IP 驱动程序。appletviewer 只支持 Windows 95 附带的微软公司的 TCP/IP 驱动程序。如果你使用的是第三方的驱动程序,当你想通过网络加载小应用程序的时候,应该更换至本地内置的微软公司的 TCP/IP 驱动程序。

(2) 如果 appletviewer 不加载小应用程序,你可以作以下尝试:

① 设置下列环境变量:

```
set HOMEDRIVE=C:  
set HOMEPATH=\\
```

然后在同一 DOS 窗口中重新启动 appletviewer。

② 设置环境变量:

```
set HOME=C:\\
```

然后在同一 DOS 窗口中重新启动 appletviewer。

如果上述努力均不奏效,你可以尝试:

```
java -verbose sun.applet.AppletViewer
```

它将列出已被加载的类的名称。从该输出中,你可以看到小应用程序浏览器试图加载哪个类、以及从哪儿加载。如果你发现某个类加载失败,请检查确认这个类已经存在、而且没有被损坏。

1.2.3 两个简单的例子

使用 Java 语言可以编写两种类型的程序:应用程序(Application)和小应用程序(Applet)。

应用程序是可以独立运行的程序,如 HotJava 浏览器;小应用程序类似于应用程序,但它不能独立运行,必须遵循一套约定,运行在 Java 使能的浏览器上。象学习其它语言一样,我们也从“Hello World!”程序开始,分别介绍这两种类型程序的一个示例,从而建立一个对 Java 语言的感性认识。

一、“Hello World!”应用程序

1. 创建 Java 源程序

用文本编辑器创建一个名为 HelloWorldApp.java 的文本文件,下面给出该文件的完整内容,请读者务必注意其中的大小写。

```
/* * 文档注释内容
 * The HelloWorldApp class implements an application that
 * simply displays "Hello World!" to the standard output.
 */

public class HelloWorldApp {
    public static void main(String[] args) { //定义 main() 过程
        System.out.println("Hello World!");
    }
}
```

2. 编译 Java 源文件

UNIX 用户和 Windows 95/NT(在 DOS Shell 下)用户都一样,均在命令行键入以下命令(请注意大小写):

```
javac HelloWorldApp.java
```

如果编译成功,将产生一个名为 HelloWorldApp.class 的文件。如果编译失败,请你仔细检查源程序及编译命令。注意:Java 是区分大小写的。

3. 运行应用程序

运行 java 命令来解释执行应用程序 HelloWorldApp,注意:java 命令的参数是运行的类的名称,而不是文件名,所以,对 UNIX 用户和 Windows 95/NT(在 DOS Shell 下)用户都一样,均在命令行键入以下命令:

```
java HelloWorldApp
```

你将能看见“Hello World!”显示在标准输出(屏幕或运行该程序的窗口)上。

二、“Hello World!”小应用程序

编写“Hello World!”小应用程序的过程很类似于“Hello World!”应用程序。

1. 创建 Java 源程序

文件名为 HelloWorldApplet.java:

```
import java.applet.Applet;  
import java.awt.Graphics;  
  
public class HelloWorldApplet extends Applet {  
    public void paint(Graphics g) {  
        g.drawString("Hello World!", 50, 25);  
    }  
}
```

2. 编译 Java 源程序

同“Hello World!”应用程序的编译过程一样。如果成功,将产生一个名为 HelloWorldApplet.class 的文件。

3. 创建包含小应用程序的 HTML 文件

在包含 HelloWorldApplet.class 文件的同一目录下创建一个名为 HelloWorld.html 的文本文件。

```
<html>  
<head>  
<title> A Simple Applet </title>  
</head>  
<body>  
  
Here is the output of my first Applet:  
<applet code=HelloWorldApplet.class width=150 height=25>  
</applet>  
</body>  
</html>
```

有关 HTML 的知识请参阅其它有关书籍。

4. 加载 HTML 文件

加载 HTML 文件至一个可运行 Java 小应用程序的浏览器(如 Netscape Navigator)或小应用程序观览器(如 JDK 提供的 appletviewer)中。为了使应用程序加载 HTML 文件,常常需

要将 HTML 文件的位置告诉应用程序,例如:

```
file:/home/jiayw/javas/HelloWorld.html
```

UNIX 用户和 Windows 95/NT(在 DOS Shell 下)用户均可键入以下命令加载 HTML 文件:

```
appletviewer file:/home/jiayw/javas/HelloWorld.html
```

实际上,如果你在 HTML 文件所在的目录下使用 appletviewer 加载 HTML 文件,可以键入下列命令进行加载:

```
appletviewer HelloWorld.html
```

如果上述步骤均成功,将会在浏览器窗口看到如下的内容:

Here is the output of my first Applet: Hello World!

图 1.1 是使用 appletviewer 加载 HTML 文件后,HelloWorld.html 的运行情况。

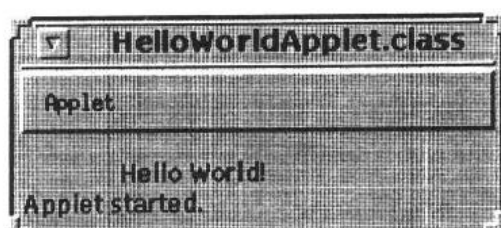


图 1.1 HelloWorld 小应用程序的运行情况