

高职高专院校“十一五”规划教材

Java 程序设计 基础教程

主编 高洪志 邓琨



哈尔滨工业大学出版社
HARBIN INSTITUTE OF TECHNOLOGY PRESS

高职高专院校“十一五”规划教材

Java 程序设计基础教程

主 编	高洪志	邓 琨	
副主编	曲开宇	侯占军	吕志峰
主 审	李 东	左晓英	

哈尔滨工业大学出版社

内 容 提 要

本书的编写针对高职高专院校及应用型本科院校的教学特点,突出了应用技能培养的需要,同时兼顾了计算机等级考试的知识点。对于 Java 的基本语法到最高级特性(面向对象的概念、多线程、输入输出、Applet、JDBC、JSP 等)进行了细致的描述,深入浅出,阐述了 Java 编程原理。本书紧密结合实际,循序渐进,总结优化了 Java 编程经验。每章都配有大量习题、典型实例及编程项目,生动详细讲解了 Java 编程思想和编程语法。简单明了,重点突出,使读者能够边学边练,激发学习兴趣。

本书共分十二章,主要内容为 Java 编程环境、Java 基本程序结构、对象和类、继承与多态、异常处理、输入输出、多线程、Applet 编程、使用 JDBC 创建数据库的连接、JSP 技术基础及 JSP 实例等内容。可作为高职高专及应用型本科院校面向对象程序设计语言及 Java 语言的授课教材,也可供 Java 程序员参考使用,同时也可作为计算机等级考试的辅助教材。

图书在版编目(CIP)数据

Java 程序设计基础教程/高洪志,邓琨主编. —哈尔滨:

哈尔滨工业大学出版社,2008.1

ISBN 978-7-5603-2255-1

I . J… II . ①高…②邓… III . JAVA 语言-程序设计-教材 IV . TP312

中国版本图书馆 CIP 数据核字(2008)第 004474 号

策划编辑 李艳文 房俊春

责任编辑 李广鑫

封面设计 卞秉利

出版发行 哈尔滨工业大学出版社

社 址 哈尔滨市南岗区复华四道街 10 号 邮编 150006

传 真 0451-86414749

网 址 <http://hitpress.hit.edu.cn>

印 刷 黑龙江省地质测绘印制中心印刷厂

开 本 787mm×1092mm 1/16 印张 18.5 字数 420 千字

版 次 2008 年 2 月第 1 版 2008 年 2 月第 1 次印刷

书 号 ISBN 978-7-5603-2255-1

定 价 28.00 元

(如因印装质量问题影响阅读,我社负责调换)

高职高专院校“十一五”规划教材编委会

主 任 姜宗胜

副主任 解学祖 裴 杭 王明海 李长碌 南文杰

许淑云 袁正友 金绍珍 邹德奎 竺培国

单振东 金 武 刘国范 张学哲 张季中

王 鹏 杨会春 许 涛

委 员 (按姓氏笔画为序)

孔祥春 王 海 王小树 王国娟 王树军

王首义 王晓平 王鹏华 左晓英 石云峰

吕 强 许洪军 何蕴争 吴文庆 吴柏英

张 岐 张大龙 张志勇 张德江 李 军

李 青 李传鸿 李艳文 肖春艳 陈亚玲

孟 鹏 武金艳 金忠伟 祖国强 郝世峰

钟啸剑 徐翠娟 贾晓芳 郭志强 郭宝清

郭翌飞 高洪志 勒恩丽 崔玉波 常健斌

符啸威 隋晓明 程显林 葛骥文 解晨光

訾世庆 靳 敏 樊任军 薛永三

前言

Preface

目前,能够适应高职高专教育的技能型教材建设仍落后于职业教育的发展。为此,哈尔滨工业大学出版社组织具有丰富教学经验的教学专家和来自生产第一线的工程师,共同研发并编写了《高职高专院校“十一·五”规划教材》系列丛书,该系列丛书以“任务驱动式”和“项目策划式”教学法作为编写主线,将计算机技能教育与各类计算机技能认证紧密结合,知识教育与计算机技能考试认证相适应。在内容组织方面以“必须、实用”为本,以“够用、适度”为纲,删繁就简,打破原教课书过于追求系统性、完整性的旧框框。重点突出实践动手技能、职业岗位技能、创新技能和解决实际问题技能的培养,强化职业技能训练。针对各专业不同的教学需要,在广度优先的基础上保证所“必须”的深度,在“够用”的理论基础上,更注重应用技术技能的培养与训练。

本系列图书的主要特点为:

- 教材中理论知识浅显易懂,实践内容丰富且比重较大。
- 教材中有丰富鲜活的案例,以任务驱动为主线。
- 学科教育与认证培训相结合。

自从1995年SUN公司推出了Java语言以来,Java因其具有纯面向对象、高性能、分布式及稳定性好等优点而迅速风靡世界,尤其是Java的跨平台性、安全性等特点使得Java非常便于开发Internet/Intranet应用程序。SUN公司在推出Java的同时,也推出了Java的开发工具JDK,其中包括Java编译器、Java解释器等开发程序时需要的工具。

虽然在程序执行效率上Java确实不如C和C++,但是在网络日益普及的今天,Java具有它独特的魅力,它的作用不再局限于程序的开发,更为重要的是它提供了一个网络解决方案,Java的内涵也不再局限于语言本身,还包括Java软件技术、Java虚拟机、Java嵌入技术等从软件到硬件的一系列概念。同时Java技术的应用范围不仅仅局限于信息领域,而且涉及电子、信息综合服务、科学计算、软件生产等很多领域,有着广阔的应用前景,受到了各类用户的肯定和支持。

本书介绍了Java编程的基础知识。书中采用分步骤的教学方法,安排了许多示例、编程项目。本书不需要读者具备编程经验,从最基本的基础知识,诸如如何编译并运行一个Java程序开始讲起。接下来讨论了Java的每一个关键字,还介绍了Java的一些最重要的高级功能,如多线程编程和创建Applet。最终读者将会牢固地掌握Java编程精髓。

本书结合了编者多年的Java教学经验,具有较好的可操作性、实验性和先进性。本书可作为应用型本科院校及高职院校教材,也可以作为培训教材使用。本书通俗易懂,简单明了,重点突出,并用各种事例来阐明一些难于理解或者易混淆的概念。读者可以边学边练,逐步加深和完善对Java核心技术的理解。

本书由高洪志和邓琨策划编写,陈艳成编写了第1、2章,侯占军、梁兴柱编写了第3、8、9章,高洪志编写了第4、5章,曲开宇、吕志峰编写了第6、7、11章,邓琨编写了第10、12章,另外参加编写和整理的还有孙立家等,全书最后由高洪志统稿。本书在编写过程中得到了哈尔滨工业大学李东教授、王首义教授、左晓英教授及王建一副教授、郭翌飞副教授的大力支持和帮助,在此表示诚挚感谢。

在本书的编写过程中尽管尽了最大努力来避免错误的发生,但限于水平和时间,书中疏漏在所难免,恳请各位专家、读者批评指正,意见和建议可发邮件至ghzhi@263.net。

编 者

2008年1月

CONTENTS 目录

第1章 引言	1
1.1 Java 语言的特点	1
1.1.1 简单性	1
1.1.2 面向对象	2
1.1.3 分布式	3
1.1.4 强健性	3
1.1.5 安全性	4
1.1.6 跨平台	4
1.1.7 可移植性	6
1.1.8 相对较高效率	6
1.1.9 多线程	6
1.1.10 动态可扩充性	8
1.2 对 Java 语言的误解	8
1.2.1 Java 是 HTML 的扩充	8
1.2.2 Java 将成为所有平台的统一编程语言	8
1.2.3 Java 是解释型的,它对于特殊平台的重要应用太慢了	8
1.2.4 所有的 Java 程序都在 Web 页中运行	9
1.2.5 Java 消除了 CGI 编程的需要	9
本章小结	9
习题	9
第2章 Java 编程环境	10
2.1 JDK	11
2.2 JBuilder	11
2.2.1 基于 JDK1.1	12
2.2.2 BeansExpress 组件重用可大大提高开发效率	12
2.2.3 可伸缩的数据库结构	12
2.2.4 实用的可视化开发工具	12
2.2.5 JBuilder OpenTools API	13
2.2.6 多层数据库链接	13
2.2.7 面向不同用户的多种版本	13
本章小结	13

习题	14
第3章 Java 基本程序结构	15
3.1 Java 程序的构成	15
3.1.1 程序构成	15
3.1.2 标识符	16
3.1.3 注释	16
3.1.4 分隔符	17
3.2 数据类型	17
3.2.1 整型	18
3.2.2 浮点型	18
3.2.3 布尔型	19
3.2.4 字符型	19
3.2.5 引用类型	20
3.2.6 数据类型转换	21
3.3 常量和变量	21
3.3.1 变量	21
3.3.2 常量	23
3.4 运算符与表达式	23
3.4.1 算术运算符	23
3.4.2 关系运算符	25
3.4.3 逻辑运算符	27
3.4.4 位运算符	28
3.4.5 赋值运算符	30
3.4.6 其他运算符	31
3.4.7 运算符的优先级和结合性	32
3.5 流程控制语句	32
3.5.1 选择语句	32
3.5.2 循环语句	36
3.5.3 break 和 continue 语句	40
3.6 数组	42
3.6.1 一维数组	43
3.6.2 二维数组	45
本章小结	49
习题	49
第4章 对象和类	54
4.1 面向对象的基本概念	54
4.2 类的定义	57
4.3 类的成员方法	58

4.4	对象的创建及回收	60
4.5	对象的使用	61
4.6	构造方法	61
4.7	包	63
4.8	访问控制	64
4.9	this 关键字	67
4.10	方法重载	69
4.11	static 关键字	73
4.12	设计类的原则	74
4.13	类之间的关系	75
4.14	嵌套类	76
	本章小结	78
	习题	78
第 5 章	继承与多态	81
5.1	继承性	81
5.1.1	子类的创建	81
5.1.2	成员变量与方法的继承	83
5.1.3	super 关键字的使用	84
5.1.4	多态性	85
5.1.5	终止类	88
5.1.6	类对象之间的类型转换	88
5.2	抽象类	89
5.3	接口	91
5.3.1	接口的定义	92
5.3.2	接口的实现和使用	92
5.3.3	接口中的变量	95
5.3.4	接口与抽象类	96
	本章小结	96
	习题	97
第 6 章	异常处理	105
6.1	异常处理基础	105
6.1.1	什么是异常	105
6.1.2	如何捕获异常	107
6.2	异常抛出	115
6.2.1	throw 语句的使用	115
6.2.2	throws 语句的使用	117
6.3	如何创建自己的异常	122
6.3.1	创建自己的异常类	122

6.3.2 Throwable 类	126
本章小结	127
习题	127
第7章 输入输出	130
7.1 概述	130
7.1.1 流的概念	130
7.1.2 I/O 流类概述	130
7.2 文件读写	132
7.2.1 写文本文件	132
7.2.2 读文本文件	134
7.2.3 写二进制文件	135
7.2.4 读二进制文件	136
7.2.5 随机文件读写	137
本章小结	139
习题	140
第8章 多线程	142
8.1 Java 线程的概念	142
8.2 Thread 类简介	143
8.3 线程的创建	144
8.3 线程的生命周期	147
8.4 线程的调度与优先级	150
8.5 控制线程	154
8.6 线程的同步	156
本章小结	163
习题	163
第9章 Applet 编程	168
9.1 Applet 基础	168
9.1.1 一个简单的 Applet 程序	168
9.1.2 Applet 与 HTML	169
9.1.3 Applet 类及 Applet 程序的生命周期	170
9.2 Applet 应用编程	173
9.2.1 播放声音	173
9.2.2 图像	173
9.2.3 网页中的动画	175
9.3 Applet 的事件处理	180
9.3.1 鼠标操作	180
9.3.2 键盘操作	182
本章小结	184

习题	184
第 10 章 使用 JDBC 创建数据库连接	186
10.1 创建数据库连接	186
10.1.1 加载驱动程序	186
10.1.2 建立连接	187
10.1.3 处理连接选项	188
10.1.4 获得数据源信息	189
10.1.5 关闭连接	189
10.2 处理查询	190
10.2.1 一般查询	190
10.2.2 参数查询	191
10.2.3 存储过程	193
10.2.4 处理查询参数	194
10.3 数据库操作	196
10.3.1 检索结果集	196
10.3.2 数据库更新操作	198
10.4 访问数据库元信息	200
10.5 Java 接口	211
10.5.1 java.sql 包	211
10.5.2 java.sql.CallableStatement	214
10.5.3 java.sql.Connection	217
本章小结	221
习题	221
第 11 章 JSP 技术基础	222
11.1 JSP 技术概述	222
11.1.1 什么是 JSP	223
11.1.2 JSP 的技术原理	223
11.1.3 JSP 的运行环境配置	226
11.2 JSP 语法	230
11.2.1 变量和方法的声明	230
11.2.2 程序段	230
11.2.3 表达式	231
11.2.4 JSP 中的注释	232
11.2.5 JSP 指令标签	233
11.3 JSP 内置对象	236
11.3.1 response 对象	236
11.3.2 request 对象	237
11.3.3 session 对象	238

11.3.4 out 对象	238
11.3.5 application 对象	239
11.4 JSP 标准动作	239
11.4.1 jsp:include 动作	239
11.4.2 jsp:forward 动作	240
11.4.3 jsp:plugin 动作	241
11.4.4 使用 JavaBean	243
本章小结	247
习题	247
第 12 章 JSP 实例	248
12.1 网上投票举例	248
12.2 留言板	260
本章小结	271
习题	272
附录 Java 编程规则	273
参考文献	281



引言

1.1 Java 语言的特点

作为一种计算机语言,Java 相当优秀,最为显著的优点是它与平台无关。Java 依靠它的运行库(Run Time Library)获得了以往任何一种语言都没有的平台无关性。同样的代码可以不用改动就可在 Windows、Solaris、Unix 等各种软硬件平台上运行。这种平台无关性对于开发 Internet 应用程序来说是非常重要的。

另外一个显著的优点是 Java 的类 C++ 语法。Java 从 C++ 发展而来,对于当今世界上众多的 C++ 程序员来说,Java 显得并不陌生。Java 是一种面向对象的编程语言,甚至比 C++ 更为“面向对象”。目前面向对象技术已经取代早期的结构化程序设计方法而成为计算机界的标准技术,因为事实证明面向对象技术处理复杂问题的优势远非其他方法所能及。

Java 并不仅仅是 C++ 语言的一个变种,它们在某些本质问题上有根本的不同:

(1)Java 比 C++ 程序可靠性更高。有人曾估计每 50 行 C++ 程序中至少有一个 BUG。姑且不去讨论这个数字是否夸张,但是任何一个 C++ 程序员都不得不承认 C++ 语言在提供强大的功能的同时也提高了程序含 BUG 的可能性。Java 语言通过改变语言的特性大大提高了程序的可靠性。

(2)Java 语言不需要程序对内存进行分配和回收。在 Java 语言中,内存的分配和回收都是自动进行的,程序员无须考虑内存碎片的问题。

(3)Java 语言中没有指针的概念,引入了真正的数组。不同于 C++ 中利用指针实现的“伪数组”,Java 引入了真正的数组,同时将容易造成麻烦的指针从语言中去掉,这将有防止在 C++ 程序中常见的因为数组操作越界等指针操作而对系统数据进行非法读写带来的不安全问题。

(4)Java 用接口(Interface)技术取代 C++ 程序中的多继承性。接口与多继承有同样的功能,但是省却了多继承在实现和维护上的复杂性。

1.1.1 简单性

用 Java 来设计应用系统时,程序员不需要非常专业的训练,并且这种语言应该和当前所流行的技术一致,因此,虽然发现 C++ 并不是这样的语言,但 Java 还是被设计成为尽可能和 C++ 类似,只有这样 Java 才易于理解。将 C++ 中那些很少使用、难于理解以及令人迷惑的东西去掉形成 Java,这些被去掉的 C++ 特性带来的好处远不如它们可能造成的痛苦多。

Java 与 C++ 极为相似,但却简单得多。Java 的语法实际上可以看做是 C++ 语法的一



个净化版。高级编程语言的所有特性中,不是绝对需要的都已删去了。例如,Java 没有运算符重载(Operator Overload)、头文件(Headerfiles)、预处理、指针运算、结构、联合、模板及隐式类型变换。但是对于 C++ 中其他的一些并不太好的语法特色,Java 并没有作改进,如 switch 语句。总的说来,如果懂得一点 C、C++ 或 Pascal,就会很快驾驭 Java,从 C++ 转变为 Java 的编程风格并不太难。相比于 VB 而言,Java 并不简单,习惯 VB 的程序员会觉得 Java 有一些很奇怪的语法。Java 与 VB 更主要的差异在 VB 的可视化编程环境上,在 VB 中很多控件对象可以简单地通过拖放相应的工具实现,但在 Java 中这一切都需要通过界面编程实现。

作为一种面向对象的语言来说,Java 是相当简单的。一方面是由于它净化了 C++ 的语法,另外一方面原因在于它的小巧。

1.1.2 面向对象

面向对象其实是现实世界模型的自然延伸。现实世界中任何实体都可以看做是对象。对象之间通过消息相互作用。另外,现实世界中任何实体都归属于某类事物,任何对象都是某一类事物的实例。如果说传统的过程式编程语言是以过程为中心以算法为驱动的话,面向对象的编程语言则是以对象为中心以消息为驱动。用公式表示,过程式编程语言为:程序 = 算法 + 数据;面向对象编程语言为:程序 = 对象 + 消息。

简单地说,面向对象的设计方法是一种主要关注数据以及与之有关的接口(操作)的设计技巧。打个比方,一个使用“面向对象”方法的木匠首先关注的是他要做的椅子,然后才是考虑应该用什么工具来做这把椅子,相反,一个使用“面向过程”的木匠首先考虑应该用什么工具来做这把椅子。Java 的面向对象性和 C++ 非常类似。

面向对象编程工具应该支持四个概念:封装、多态性、继承性和动态联编,Java 也不例外。现实世界中的对象均有属性和行为,映射到计算机程序上,属性表示对象的数据,行为表示对象的方法(其作用是处理数据或同外界交互)。所谓封装,就是用一个自主式框架把对象的数据和方法联在一起形成一个整体。可以说,对象是支持封装的手段,是封装的基本单位。Java 语言的封装性较强,因为 Java 无全程变量,在 Java 中绝大部分成员是对象,只有简单的数字类型、字符类型和布尔类型除外。而对于这些类型,Java 也提供了相应的对象类型以便与其他对象交互操作。

多态性就是多种表现形式,具体来说,可以用一个对外接口,多个内在实现“方法”表示。举一个例子,计算机中的堆栈可以存储各种格式的数据,包括整型、浮点或字符。不管存储的是何种数据,堆栈的算法实现是一样的。针对不同的数据类型,编程人员不必手工选择,只须使用统一接口名,系统可自动选择。运算符重载一直被认为是一种优秀的多态机制体现,但使用它程序会变得难以理解,所以 Java 最后还是把它取消了。

继承性是指一个对象直接使用另一对象的属性和方法。事实上,很多实体都有继承的含义。例如,把汽车看成一个实体,它可以分成多个子实体,如卡车、公共汽车等。这些子实体都具有汽车的特性,因此,汽车是它们的“父亲”,而这些子实体则是汽车的“孩子”。Java 提供给用户一系列类(Class),Java 的类有层次结构,子类可以继承父类的属性和方法。与另一些面向对象编程语言不同,Java 只支持单一继承。

动态联编指的是一旦对象生成以后,要使用这个对象时只须简单地把信息传递给它,不



再需要去参考对象当初设计,这体现了 Java 设计具有的灵活性。

从这个四方面来衡量现有的编程语言或工具,有很多并不十分完善,Visual Basic 缺乏数据的封装性。相比之下,Java 在四个方面都可以做得很好。Java 的面向对象性与 C++ 语言非常相似,二者最主要的区别在于多层继承性方面,Java 在处理多层继承性方面有更好的解决办法。

近 30 年来,面向对象已经向世人证明了它的价值,现在的程序员如果还不掌握面向对象编程思想的话,那么在与别人交流或合作时将会感到非常的不便。

1.1.3 分布式

分布式包括数据分布和操作分布。数据分布是指数据可以分散在网络的不同主机上,操作分布是指把一个计算分散在不同主机上处理。

Java 支持 WWW 客户机/服务器计算模式,因此,它支持这两种分布性。对于前者,Java 提供了一个叫做 URL 的对象,利用这个对象,可以打开并访问具有相同 URL 地址上的对象,访问方式与访问本地文件系统相同。对于后者,Java 的 Applet 小应用可以从服务器下载到客户端,即部分计算在客户端进行,提高系统执行效率。

Java 有着丰富的类库,可以很方便地编写 TCP/IP 协议(如 HTTP、FTP 等)上的应用程序。Java 应用程序可以由网络通过 URL 调用远程对象,与操作本地文件系统没有区别。

Java 提供了一整套网络类库,开发人员可以利用类库进行网络程序设计,方便地实现 Java 的分布式特性。Java 语言的网路功能非常强大,而且很简单,如果曾使用其他的语言编写过网络应用程序的话,就会惊讶于用 Java 生成一个 Socket 竟然这样简单。运用 Java 编写 CGI 脚本也很简单。

1.1.4 强健性

Java 最初的设计是应用于电子类消费产品,因此要求较高的可靠性。Java 虽然源于 C++,但它消除了许多 C++ 的不可靠因素,可以避免许多编程错误。

设计 Java 语言的另外一个出发点是用 Java 语言编程必须是可靠的。Java 非常强调对错误的检查,包括早期对可能问题的检查,后期(运行时)对错误的动态检查以及消除可能产生错误的情况,其中 Java 在强健性上与 C++ 最大的区别在于 Java 使用一个指针模型,不同于 C++ 的指针,Java 不会产生对内存的非正常操作或者造成数据的混乱。

首先,Java 是强类型的语言,要求显示的方法声明,这保证了编译器可以发现方法调用错误,保证程序更加可靠。Java 的编译器会指出许多程序潜在的错误,这些错误在其他的语言中可能要到运行时才会显示为错误(甚至在运行时都不会显示出错)。

其次,Java 不支持指针,这杜绝了对内存的非法访问。关于指针问题,VB 的程序员可能感觉不到 Java 的优点,因为在 Basic 语言中不需要使用指针,也用不着担心内存的问题(当然也不能轻易地实现那些需要指针的复杂的数据结构),C 或 C++ 语言的程序员对此深有体会,C 语言要用指针处理字符串、数组、对象甚至文件。而稍有不慎,指针就有可能造成系统的紊乱。Java 综合了 C 与 Basic 的优点:第一,Java 减少了对指针的依赖——C 语言中需要用指针实现的数据结构很多在 Java 中不再要用指针了,如字符串、数组等。第二,Java 仍



然可以利用安全的指针来实现复杂的数据结构,如链表等。Java 的指针之所以安全在于它不会访问一个非法的指针或造成内存的分配错误。第三,Java 的自动单元收集避免了内存丢失等动态内存分配导致的问题。第四,Java 解释器运行时实施检查,可以发现数组和字符串访问的越界。第五,Java 提供了异常处理机制,程序员可以把一组错误代码放在一个地方,这样可以简化错误处理任务并便于恢复。

1.1.5 安全性

一种编程语言,尤其是网络编程语言,与跨平台性随之而来的还有另一大特性:安全性。在网络上,如果网络语言不十分可靠,极有可能被人有意或无意地利用而造成恶果:也许文件会在硬盘上奇怪地消失;也许发出去的文件又莫名其妙地反复存在自己的服务器上;也许内存忽然不足,以至运行的程序也越来越慢,甚至突然中断;也许秘密在不知不觉中被人窃取;更有甚者,系统无端地崩溃。

Java 被设计成为易于在网络/分布环境下运行,因此在安全性上作了很多的考虑,Java 可以避免计算机病毒的侵害,也能够防止对系统的意外损害。

作为网络上的应用程序,安全性是至关重要的。为防止利用 Java 程序攻击其他人的系统,Java 主要的编译和运行时,系统中采取了防护措施:编译程序保证源代码不违反安全规则;运行时系统校验字节代码,类装入程序保证类不侵犯名字空间和访问限制的范围;运行时进行内存分配以使潜在的破坏者无法预知内存的布局,对文件的访问加以限制(当从服务器传过来的 Java 程序被像 Netscape Navigator 这样的客户端程序激活时,它不能对本地的文件进行读或写)。这其中,字节代码校验是一个关键部分。Java 字节码的一个特性是它的可解析性强,能够对其进行分析并对其进行以后的行为做出推理,这是 Java 安全系统的基础之一。任何代码在执行以前,都要经过一系列严格而复杂的测试。通过测试,可以保证校验字节代码不非法进行类型转换(如伪造指针),不以非法形式访问对象,操作数堆栈不上溢或下溢,不错误地使用参数、变量和类型等。总之,Java 的编译和运行环境为 Java 代码筑起了两道坚实的防护屏障。

1.1.6 跨平台

所谓跨平台,就是指应用程序与计算机系统硬件结构的无关性,更直观地说就是用 Java 编写的应用程序不用修改就可不同的软硬件平台上同样运行。跨平台包括源代码级和目标代码级。C 和 C++ 具有一定程序的源代码级与平台无关,表明用 C 或 C++ 写的应用程序不用修改只须重新编译就可以在不同平台上运行。

跨平台在 Internet 上显得尤其重要。在 Internet 上,存在着各种各样的终端、各种各样的系统(Windows95、WindowsNT、Unix、Macintosh、Solaris 等),倘若一种编程语言希望在 Internet 上畅通无阻,必然要在跨平台性上提出较高的要求。抛开网络不谈,在软件界中跨平台性也一直是程序员为之头痛的事情。一种平台上编译执行的软件移植到另一种平台,就需要重新编译,再加上各种平台和操作系统规则不尽一致,极易出错。因此 Java 的跨平台性具有深远意义。首先,它使得编程人员梦寐以求的事情(开发一次,任意运行)变成事实,这将大大加快和促进软件产品的开发。其次 Java 的平台无关性正好迎合了“网络计算机”思想。如



果大量常用的应用软件(如字处理软件等)都用 Java 重新编写,并且放在某个 Internet 服务器上,那么作为网络客户端的用户将不需要占用大量的空间安装软件,他们只需要一个 Java 解释器。目前,已有数家公司开始使用这种新型的计算机模式构筑自己的企业信息系统。

Java 并不是第一种跨平台的高级语言,早在二十多年前,UCSD Pascal 系统就已经号称“跨平台”了。Java 通过半编译半解释的方式成功地解决了平台独立性的问题。它并不像 C 或 C++ 一样把源代码直接编译成可供机器执行的低层的机器语言,而是生成一种介于源码和机器码之间的字节码(Byte Code,以 Class 为扩展名,又称中介码)。事实上,它的源码和字节码分别跨平台,这样无论是何种机型,何种操作系统,只要它装有 Java 解释器,Java 程序都可以很好地运行。

虽然这种半编译半解释的思维方式并非 Java 所独有,但相比之下,Java 更趋成熟,Java 的设计者在设计字节码时充分考虑了当今常用的计算机体系结构,使得字节码能在大多数的机器上正常运行,并且从字节码到本地机器码的翻译过程也很简单。

当然,字节码的出现必然影响到速度问题,这也是 Java 发展进程中的一大障碍。有资料表明,由于多了一道“解释器”的手续,Java 比起 C 和 C++ 来要慢 50%~200% 左右。显然,在速度要求比较高的场合,Java 就有些力不从心了。目前有两种方法可使 Java 更快一些:在 Java 程序中链接原生码(Native Code,也叫做本地方法代码),或者把 Java 字节码转换为原生码。但是如果采用这些方法,将失去 Java 的跨平台等诸多特性。

SUN 公司并未停留在 Java 软件方式的虚拟机 JVM(Java Virtual Machine)上,它开发 Java 芯片(1997 年 12 月 SUN 在北京发布了它的 Java 芯片),希望以微处理器的形式从硬件上解释 Java 以获得更快的速度。

除了字节码,Java 还定义了一整套严格的语言规范,每一部分的数据类型和变量初值都是确定的。这就从根本上消除了跨平台时由于系统的不同而造成的危险。例如,在 C 语言中变量不设初值,下列语句虽然在警告后仍执行,但在不同的平台上其结果无法预知:

```
int var1, var2;
```

```
var2 = var1 * 5;
```

而 Java 即使没有明确的程序定义,但它已经事先缺省地给各种变量赋予了初值,所以,上述语言合法,并仍能预测结果, $var2 = 0$, $var1 = 0$ 。

但尽管 Java 在语法上、在虚拟机上煞费苦心,但跨平台性仿佛并未尽善尽美,归纳起来大概有 3 点:

(1) Java Applet 不继承用户声明的浏览器的特性。比如说,当在主页(Hompage)上设计 Applets 时,力图使 Applet 中的字体与它周围的静态文本相吻合。如果使用与设计者相同的浏览器观看时结果很好,但倘若在别的平台上或使用其他的浏览器浏览时,也许多少会有一点差别。

(2) Java 中的 AWT(Abstract Window Toolkit)要依靠原有的窗口系统操作它的 UI 单元:在 Unix 上用 X/motif,在 Macintosh 上用 Toolbox,在 Windows 上用 Windows 等。但每一种窗口对自己的单元都有不同的规定。所以其外形不尽相同,这对于文件是不利的。

(3) 因为 Java 是跨平台的,所以它在执行时所需要的任何东西都必须也是跨平台的。如果用 SUN 公司的 JDBC 封装编写了一个数据库,你必须保证每一台机器在使用数据库时