

Java

语言基础

王艳玲 主编



21世纪高职高专计算机科学与技术应用专业系列教材



机械工业出版社
CHINA MACHINE PRESS



21 世纪高职高专计算机科学与技术专业系列教材

Java 语言基础

王艳玲 主编



机械工业出版社

本书从面向对象的基本概念和 Java Application 及 Java Applet 的结构入手,按照循序渐进、深入浅出的原则,讲述 Java 语言的主要特点、基础知识及其面向对象的编程方法。介绍了对象和类、包和接口的使用方法、异常处理机制、常用系统类、图形用户界面、多媒体编程、事件处理机制、多线程技术及网络编程等 Java 技术。

全书选材适当、结构合理、内容新颖,注重实用。每章的最后都配有小结和习题,以指导读者深入地学习。书中的所有例题都在 j2sdk1.4.1 环境下调试通过。本书既可作为高职高专院校的教材,又可作为对 Java 语言和面向对象编程技术感兴趣读者的自学用书。

图书在版编目(CIP)数据

Java 语言基础/王艳玲主编. —北京:机械工业出版社, 2003.7

(21 世纪高职高专计算机科学与技术专业系列教材)

ISBN 7-111-12451-0

I. J... II. 王... III. Java 语言—程序设计—高等学校: 技术学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2003)第 049393 号

机械工业出版社(北京市百万庄大街 22 号 邮政编码 100037)

策 划: 胡毓坚

责任编辑: 田 梅

责任印制: 路 琳

北京市樱花印刷厂印刷·新华书店北京发行所发行

2003 年 7 月第 1 版·第 1 次印刷

787mm×1092mm 1/16·16.25 印张·399 千字

0001—5000 册

定价: 23.00 元

凡购本图书,如有缺页、倒页、脱页,由本社发行部调换
本社购书热线电话(010) 68993821、88379646

封面无防伪标均为盗版

出版说明

新世纪对高职高专教育提出了新的目标和要求,高职高专教育面临新一轮的改革和发展。为了进一步推进高职高专的教育,培养 21 世纪与我国现代化建设相适应的,具有较宽厚的文化基础底蕴,并在生产、管理、服务岗位第一线的技术型应用型人才。机械工业出版社与高职高专计算机科学与应用专业教材编委会联合组织了全国近百所院校的一线骨干教师,在交流、研讨的基础上,根据国家教育部的精神,以及高职高专教学改革的新思路、新突破、新经验、新举措,编写了这套“21 世纪高职高专计算机科学与应用专业系列教材”。目前已出版了 2 轮,近 30 种教材。随着教改的深入,新技术的出现,新一轮的高职高专教材将陆续出版。

第一轮教材更明确高职高专学生培养的定位,更强化学生实践能力和创新意识的培养,更反映现代高等职业技术教育的理念、方法和手段,更注重培养第一线的技术应用型人才。新的教材是将高职高专院校教学改革力度比较大,内容新颖,注重能力,体现创新的教材,或者各院校急需使用,适合社会经济发展新课题的教材列入选题规划,进行修编或新编。力求体现“定位准确、注重能力、内容创新、结构合理和叙述通俗”的编写特色。新教材是由个人申报,经各院校推荐,编委会会同专家评选,出版社立项出版的。

望各高职高专院校积极选用本套系列教材,及时提出修改意见,不断提高教材的编写质量。

高职高专计算机科学与应用专业教材编委会
机械工业出版社

21 世纪高职高专计算机

科学与应用专业教材编委会名单

主 任 周智文

副主任 周岳山 詹红军 林 东 王协瑞 李传义

委 员 赵佩华 陈付贵 吕何新 朱连庆 陶书中

刘瑞新 鲁 辉 王德年 马 伟 于恩普

谢 川 姜国忠 汪赵强 龚小勇 马林艺

王 泰 陶 洪 余先锋 陈丽敏 翟社平

赵增敏 王养生 赵国玲 卫振林 顾 伟

总策划 胡毓坚

前 言

因特网的互联性、开放性和共享信息的模式，打破了传统信息的传播方式，改变了人们的工作和生活方式。随着社会信息化程度的提高，社会生活中依赖于计算机解决的问题越来越多，也越来越复杂，面向对象的技术正在逐步取代传统的面向过程的程序设计方法，成为计算机应用开发领域的主流方向。

诞生于 20 世纪 90 年代初的 Java 语言是面向对象技术成功应用的著名范例，它是一种解释型的、面向对象的程序设计语言。它所具有的与平台无关的特性、较高的可靠性和安全机制以及内嵌的网络支持等特色，特别适合于 Internet 网上应用程序的开发。Java 语言已成为当今编写网络程序的首选工具之一。

本书所针对的读者是高职高专的学生及其他对 Java 语言和面向对象编程技术感兴趣的读者。本书内容遵循教学规律，按照易学、易懂、易操作、易掌握的原则，结合 Java 语言的特点，由浅入深、难易结合、循序渐进地介绍了 Java 语言及其程序设计的方法。本书无论是从教材内容的编排上，还是从例题、习题的选择上都保证了一定的实用性、新颖性和系统性。书中的所有例题都在 j2sdk1.4.1 环境下调试通过。

全书共分 11 章。第 1 章介绍了 Java 语言的发展和特点，两种类型的 Java 程序与工作机制，Java 开发工具 j2sdk1.4.1 的使用；第 2 章介绍了 Java 语言的基本组成、数据类型、运算符及数组的定义及使用方法；第 3 章介绍了 Java 各种流程控制语句；第 4 章介绍了面向对象程序设计的方法，包括类、对象及方法的概念及使用；第 5 章介绍了类的继承与多态特性及接口、包的概念及使用；第 6 章介绍了异常的概念及 Java 的异常处理机制；第 7 章介绍了 Java 常用的系统类，包括 Applet 类、字符串类、数学函数类、日期类、随机数类及向量类等的使用方法；第 8 章介绍了如何使用 Swing 建立图形用户界面，并介绍了 Java 事件处理机制；第 9 章介绍了 Java 的多媒体技术，包括如何绘制图形、显示图像，如何设计动画和播放声音等；第 10 章介绍了多线程程序设计方法，包括线程的概念、线程的创建与管理方法；第 11 章介绍了 Java 网络编程技术。

本书第 1~4 章由王艳玲编写，第 5~7 章由朱尧花编写，第 8~10 章由刘艳青编写，第 11 章由郑金刚编写。全书由王艳玲主编。在本书的编写过程中，得到了山东省电子工业学校各级领导的大力支持和协助，并提出了许多宝贵意见，谨此表示由衷的感谢。

由于编者水平有限，书中难免存在一些错误和疏漏，恳请各位专家和读者批评指正。

目 录

出版说明	2.4.7 字符串运算符	33
前言	2.4.8 其他运算符	33
第1章 概述	2.4.9 运算符优先级	34
1.1 Java语言的发展和特点	2.4.10 表达式	34
1.1.1 Java的起源和发展	2.5 数组	35
1.1.2 Java语言的特点	2.5.1 一维数组	36
1.2 Java程序举例	2.5.2 一维数组的引用	38
1.2.1 Java应用程序举例	2.5.3 多维数组	38
1.2.2 Java小程序举例	2.5.4 数组的复制	40
1.2.3 Java程序的结构	2.6 小结	41
1.3 Java程序开发工具与运行环境	2.7 习题	41
1.3.1 Java程序开发工具	第3章 流程控制	43
1.3.2 JDK的下载与安装	3.1 选择控制结构	43
1.3.3 Java应用程序的开发及运行过程	3.1.1 if语句引导的选择结构	43
1.4 小结	3.1.2 if...else语句引导的选择结构	44
1.5 习题	3.1.3 switch语句引导的多分支选择结构	45
第2章 Java的语法基础	3.2 循环控制结构	47
2.1 Java语言的基本组成	3.2.1 for循环语句	48
2.1.1 标识符	3.2.2 while循环语句	50
2.1.2 保留字	3.2.3 do...while语句	51
2.1.3 分隔符	3.2.4 循环嵌套	52
2.2 简单的输入输出	3.3 跳转控制语句	54
2.2.1 系统标准输出	3.3.1 break语句	54
2.2.2 系统标准输入	3.3.2 continue语句	55
2.3 数据类型	3.3.3 return语句	56
2.3.1 常量与变量	3.4 小结	57
2.3.2 基本数据类型	3.5 习题	57
2.4 运算符与表达式	第4章 Java面向对象编程	59
2.4.1 算术运算符	4.1 方法	59
2.4.2 关系运算符	4.1.1 方法的概念及作用	59
2.4.3 条件运算符	4.1.2 变量的作用域	61
2.4.4 逻辑运算符	4.1.3 方法的调用及参数传递	63
2.4.5 位运算符	4.1.4 方法的递归调用	65
2.4.6 赋值运算符	4.1.5 方法的重载	65
	4.2 类	66

4.2.1	Java 的类特性	66	抛出异常	109
4.2.2	Java 类的定义	67	6.2.2	异常的捕捉
4.2.3	类对象	69	6.2.3	finally 语句
4.2.4	类和成员的修饰符	69	6.3	创建自己的异常
4.2.5	构造方法	71	6.4	小结
4.3	对象	72	6.5	习题
4.3.1	对象的性质	72	第 7 章	常用系统类
4.3.2	对象的创建	73	7.1	Applet 类
4.3.3	对象的使用	74	7.1.1	Applet 概述
4.3.4	对象的清除	77	7.1.2	Applet 的主要方法
4.4	小结	77	7.1.3	Applet 的标签属性
4.5	习题	77	7.1.4	Applet 和 Application 的 合并运行
第 5 章	继承与多态	79	7.1.5	HTML 和 Applet 参数传送
5.1	类成员的继承	79	7.2	字符串类
5.1.1	父类和子类	79	7.2.1	字符串类概述
5.1.2	成员变量的继承和隐藏	81	7.2.2	字符串类的常用方法
5.1.3	成员方法的覆盖	82	7.2.3	字符串类的应用
5.1.4	this 和 super 的使用	84	7.3	其他常用类
5.2	类的多态性	85	7.3.1	数学函数类
5.2.1	成员方法的重载	86	7.3.2	日期类
5.2.2	构造方法的重载	89	7.3.3	随机数类
5.3	接口	90	7.3.4	向量类
5.3.1	接口的定义	90	7.4	小结
5.3.2	接口的实现	91	7.5	习题
5.3.3	接口的继承	93	第 8 章	Java 的图形用户界面
5.3.4	接口类型的使用	94	8.1	Java 的用户事件处理
5.4	包	97	8.1.1	事件处理机制
5.4.1	包的创建	97	8.1.2	AWT 事件分类
5.4.2	包的引用	98	8.1.3	适配器类和匿名类
5.4.3	Java 的系统程序包	99	8.1.4	键盘事件举例
5.5	小结	100	8.2	基本用户界面
5.6	习题	101	8.2.1	标签 (JLabel)
第 6 章	异常处理	103	8.2.2	按钮 (JButton)
6.1	异常概述	103	8.2.3	选项框
6.1.1	异常的概念	103	8.2.4	组合框 (JComboBox)
6.1.2	异常的层次结构	105	8.2.5	列表框 (JList)
6.1.3	Java 定义的标准异常类	106	8.2.6	文本框 (JTextField)
6.2	Java 的异常处理机制	109	8.2.7	文本区 (JTextArea)
6.2.1	利用 throw 和 throws 语句			

8.2.8	滚动条 (JScrollBar)	166
8.3	高级用户界面	167
8.3.1	颜色和字体	167
8.3.2	面板	170
8.3.3	窗口、框架、对话框	173
8.3.4	菜单	180
8.3.5	布局管理器	183
8.4	小结	195
8.5	习题	195
第9章	Java 的多媒体技术	197
9.1	多媒体技术概述	197
9.2	图形处理	197
9.2.1	基本图形	198
9.2.2	文字输出	203
9.2.3	颜色与绘图模式控制	204
9.3	图像处理	205
9.3.1	图像的种类	205
9.3.2	图像的显示	206
9.3.3	图像的生成	209
9.3.4	图像效果处理	210
9.4	动画处理	213
9.4.1	动画原理	213
9.4.2	图像的动画	213
9.4.3	文本的动画	214
9.5	声音处理	216
9.5.1	声音文件的加载	216
9.5.2	在 Applet 中播放声音	216
9.5.3	在 Application 中播放声音	218
9.6	小结	219

9.7	习题	219
第10章	线程	220
10.1	线程的基本概念	220
10.1.1	程序与进程	220
10.1.2	进程与线程	220
10.1.3	线程与多线程	221
10.2	线程的创建与使用方法	221
10.2.1	线程的生命周期	221
10.2.2	创建线程	222
10.2.3	在 Applet 中使用线程	225
10.3	线程的管理	230
10.3.1	线程的同步	230
10.3.2	线程的优先级	233
10.4	小结	234
10.5	习题	234
第11章	Java 网络程序设计	236
11.1	Java 网络基础知识	236
11.2	利用 URL 实现网络通信	238
11.2.1	什么是 URL	238
11.2.2	创建 URL 对象	238
11.2.3	利用 URL 读取服务器文件	239
11.2.4	利用 URLConnection 和服务器交互	240
11.3	服务器程序的实现	243
11.4	客户程序的实现	245
11.5	小结	247
11.6	习题	247
参考文献		248

第1章 概述

1.1 Java 语言的发展和特点

Java 语言是一种面向对象的、解释型的、多线程的、交互式的、与平台无关的程序设计语言。Java 语言诞生于 20 世纪 90 年代初，它的与平台无关的特性、高可靠性和安全机制以及内嵌的网络支持等特色，特别适合于 Internet 应用程序的开发。Java 语言已成为当前编写网络程序的首选工具之一。

1.1.1 Java 的起源和发展

Java 语言的开发始于 1991 年，它来自于美国 Sun 公司的一个叫 Green 的项目，其前身是 Sun 公司为电视、电冰箱等家用消费电子产品开发的一种智能化家电语言 Oak（橡树）。Oak 是一种用于网络的精巧而安全的语言，Sun 公司曾依此投标一个交互式电视项目，但结果被竞争对手打败。随后 Oak 项目组成员发现可以利用 Oak 创造动态页面，便对 Oak 进行了大规模的改造，推出了新一代的面向对象的程序设计语言即 Java。

程序设计语言的取名一般有两种方式，即取英文单词的缩写词或其他有意义的关键词，如 BASIC 是 Beginner All Symbolic Instruction Code 的缩写，C++ 是 C 语言的改进版本等。Java 的取名与传统的取缩写词或其他有意义的关键词的做法不同，它只是 Java 的开发者在喝一种名叫 Java（爪哇）的咖啡时产生的灵感。1993 年 WWW 出现后，Java 就成为 Web 编程的有力工具，并且随着 WWW 的迅速发展，Java 语言的开发逐渐流行起来。如今，Java 本身已从一种程序设计语言上升为一类技术，成为网络编程及电子商务系统开发不可缺少的有力工具和开发平台。

Java 作为一种革命性的软件开发技术，已得到了工业界的普遍认可。包括 IBM、Apple、DEC、Adobe、HP、Oracel 及 Microsoft 在内的越来越多的计算机产业的大公司都购买了 Java 的许可证，众多的软件开发商也开发出了支持 Java 的软件产品。同时，随着 Java 技术的日益成熟，一些消费电器产品也采用了 Java 技术，如最近一段时间被炒得沸沸扬扬的 Java 手机就是一个很好的例子。随着 Java 在网络应用方面的进一步增强，Java 的类库也在不断丰富，其性能不断提高，应用领域得到了进一步的拓展。

1.1.2 Java 语言的特点

作为一种程序设计语言，Java 语言简单易用、面向对象、不依赖于机器的结构、具有可移植性、健壮性和安全性，并且提供了多线程的机制，具有很高的性能。其次，Java 语言最大限度地利用了网络，Java 的小应用程序（Applet）可在网络上传输而不受 CPU 和网络环境的限制。另外，Java 还提供了丰富的类库，使程序设计者可以很方便地建立自己的系统。Java 语言已成为网络编程和电子商务系统开发不可缺少的有力工具，在计算机和网络技术高

度发展的今天占据着关键性的地位。其主要特点如下:

1. 简单性

Java 语言是一种面向对象的语言,它通过提供最基本的方法来完成指定的任务。程序设计者只需理解一些基本的概念,就可以用 Java 语言编写出适合于各种情况的应用程序。Java 语言的风格与 C++ 十分接近,但又摒弃了 C++ 中的头文件、预处理器、指针、内存管理等一些不常用又容易引起程序错误的特性,使得 Java 比 C++ 更容易学习,其程序的可读性更强。

2. 面向对象

面向对象的程序设计是当今程序设计的主流。面向对象是现实世界模型的自然延伸,自然界中的任何事物都可以看做对象。对象可以是看得见、摸得着的有形的实体,如汽车、教室等;也可以是表示人或组织所起的作用,如驾车、教学等;还可以表示事件,如开演唱会、看足球赛等;更可以表示规则,如约束条件、规章制度等。

面向对象程序设计 (Object Oriented Programming) 又简称 OOP,它具有数据封装、继承和方法重载三个方面的特性。所谓封装是指将数据分成私有数据、保护数据、公用数据等,达到数据保护的目的。所谓继承是指有继承的关系后,子类就可继承父类的数据或方法,从而达到数据重复使用的目的。方法重载是指同样名称的方法,其程序代码不同。例如,假设父类中有一个方法名叫 `dispaly()`,用于显示员工的出生年月等信息;子类中也有一个方法名叫 `dispaly()`,用于显示员工的电话号码数据。这两种方法的名称都叫 `dispaly()`,但它们的作用不同。子类中的 `dispaly()` 将覆盖掉父类中的 `dispaly()` 方法,称为方法的重载。关于 OOP 的这三个特性将在本书后面的有关章节中详细介绍。

Java 语言是一种纯粹的面向对象的程序设计语言,除了基本的数值型、布尔型和字符型数据以外,其余的结构都是对象。Java 的程序设计集中于对象及其接口,它提供了简单的类机制以及动态的接口模型。对象中封装了它的状态变量以及相应的方法,实现了模块化的结构和信息的隐藏;而类则提供了一类对象的原型,并且通过继承机制,子类可以使用父类所提供的方法,实现了代码的重复使用。与其他程序设计语言相比较,Java 是一种更严谨的、更标准的和具有更高可靠性的面向对象的程序设计语言。

3. 分布式

Java 语言之所以成为当前编写网络程序的首选工具,一个非常重要的原因就在于它是一种分布式的语言。这里所说的分布式主要是指数据分布和操作分布。数据分布是指数据可分散在网络中的不同主机上;操作分布是指把一个操作处理分散在不同的主机上进行。另外,Java 还包含一个支持 HTTP 和 FTP 等基于 TCP/IP 协议的内容广泛的网络类库,可支持网络上的应用程序,程序设计者可利用 Java 丰富的网络类库进行网络程序的开发。因此,Java 应用程序可凭借 URL 地址打开并访问网络上的对象,其访问方式与访问本地文件系统几乎完全相同。

4. 结构中立性

为了使 Java 程序能在任意类型的系统上运行,Java 源程序被编译成一种高层次的与机器无关的 Byte-Code (字节码) 格式语言。这些字节码指令需要由 Java 运行系统中的解释器来解释执行,只要机器中安装了 Java 运行系统,Java 解释器得到字节码后,对它进行转换,Java 程序就可在任意的处理器上运行。

5. 稳定性

为保证系统的稳定性, Java 在编译和运行程序时, 都要对可能出现的问题进行检查, 以避免错误的产生。Java 系统通过提供自动垃圾收集的方法来进行内存管理, 以防止程序员在管理内存时产生的错误。通过面向对象的异常处理机制, 提示出可能出现但未被处理的异常, 帮助程序员正确地进行选择以防止系统的崩溃。Java 在编译时还可捕获类型声明中的许多常见错误, 防止动态运行时不匹配问题的出现。另外, Java 不支持指针, 一切对内存的访问都必须通过对象的实例变量来实现, 这样就防止了病毒程序使用欺骗手段访问对象的私有成员, 同时也避免了指针操作中容易产生的错误, 从而提高了系统的安全性。

6. 解释执行

按照程序的执行方式, 程序设计语言分为编译执行的语言和解释执行的语言。编译执行是在编写完程序后, 通过特定的工具软件将源代码经过目标代码转换成机器代码(即可执行程序), 然后直接交给操作系统运行。这类程序设计语言的编译器与机器之间存在一定的依赖性, 在一个系统上编译的程序到另一个系统上不一定能够运行。C 语言就属于编译执行的程序设计语言。解释执行是程序读入一句执行一句, 不需要编译连接, 这样的语言与操作系统的相关性较小。Java 属于解释执行的程序设计语言。Java 解释器直接对 Java 字节码进行解释执行, 字节码本身携带了许多编译信息, 使得连接过程更加简单。

7. 可移植性

由于 Java 是解释执行的语言, 因此只需提供字节码, 就可在不同的平台上执行。另外, Java 所提供的基本数据类型的大小不会因为系统的不同而变化。正是因为 Java 具有与体系结构无关的特性, Java 应用程序可以在配备了 Java 运行环境的任何计算机系统上运行。因此, Java 应用软件能够方便地被移植到网络上的不同机器中。同时, Java 的类库中也实现了与不同平台的接口, 使这些类库可以移植。另外, Java 编译器也是由 Java 语言实现的, 这使得 Java 系统本身也具有可移植性。

8. 高性能

和其他解释执行的程序设计语言如 BASIC 不同, Java 字节码的设计能够很容易地直接转换成对应于特定 CPU 的机器码, 从而具有较高的性能。

9. 多线程

线程是一个大进程里分出来的小的独立的进程。Java 提供的多线程功能使得在一个程序里可同时执行多个小任务。例如 Web 浏览器的用户在翻页的同时可收听音乐, 而后台浏览器还可同时下载图像。Java 语言是一种多线程语言, 提供了一套复杂的线程同步机制, 可同时对不同的任务进行管理。多线程机制使应用程序能够并行执行, 而且同步机制保证了对共享数据的正确操作。通过使用多线程, 程序设计者可以分别用不同的线程完成特定的行为, 而不需要采用全局的事件循环机制, 这样就很容易实现网络上的实时交互行为。

10. 动态性

Java 语言适用于变化的环境, 是一种动态的程序设计语言。Java 的动态特性是其面向对象设计方法的发展, 它允许程序动态地装入运行过程中所需要的类。

从上面的特性可以看出, Java 确实是一种适合于 Internet 和分布式环境的技术。在本书后面的各个章节中, 通过对 Java 语言各个方面的介绍来印证上述的基本特点。

1.2 Java 程序举例

Java 编程语言可用来创建任何常规编程语言所能创建的应用程序。按照 Java 程序的结构和运行环境的不同,Java 程序分为两种,即 Java 应用程序 (Java Application) 和 Java 小应用程序 (Java Applet)。

1.2.1 Java 应用程序举例

Java 应用程序 (Java Application) 是一种独立的完整的程序,由 Java 解释器加以解释就可独立运行,它不需要任何 Web 浏览器来执行。Java Application 是一种典型的通用程序;可运行于任何具备 Java 运行环境的设备中。下面是 Java Application 的简单例子。

【例 1-1】简单的 Java Application。

```
public class First {  
    public static void main (String args[]) {  
        System.out.println("This is my first Java Application!");  
    }  
}
```

本程序的作用是在标准输出设备 (显示器) 上输出下面一行信息:

This is my first Java Application!

程序中,首先用保留字 `class` 来声明一个新的类,其类名为 `First`,它是一个 `public` 公共类,表明所有的程序都可使用此类。整个类定义由大括号 `{ }` 括起来。在该类中定义了一个 `main ()` 方法,其中 `public` 表示访问权限,指明所有的类都可以使用这一方法;`static` 指明该方法是一个类方法,它可以通过类名直接调用;`void` 则指明 `main ()` 方法不返回任何值。

对于一个 Java 应用程序来说,`main ()` 方法是必须的,而且必须按照上述的格式来定义。Java 解释器在没有生成任何实例的情况下,以 `main ()` 作为入口来执行程序。Java 程序中可以定义多个类,但是最多只能有一个公共类 (`public` 类),每个类中可以定义多个方法,但 `main ()` 方法也只能有一个,作为程序的入口。`main ()` 方法定义中,括号 `()` 中的 `String args[]` 是传递给 `main ()` 方法的参数,参数名为 `args`,它是类 `String` 的一个实例,参数可以为 0 个或多个,每个参数用“类名 参数名”来指定,多个参数间用逗号分隔。

在上面的例子中,在 `main ()` 方法的方法体 (大括号) 中,只有 `System.out.println ("This is my first Java Application!");` 一条语句,它用来将括号中的字符串输出到系统标准输出设备 (显示器) 上。

首先把该程序保存到一个名为 `First.java` 的文件中。需要注意的是,由于 Java 解释器要求公共类必须放在与其同名的文件中,所以文件名必须和类名相同,在使用时尤其要注意字母的大小写。然后输入 `javac First.java` 命令对它进行编译,编译的结果是生成字节码文件 `First.class`。最后执行 `java First` 命令用 Java 解释器来运行该字节码文件,得到程序的运行结果。

1.2.2 Java 小程序举例

Java Applet 是一种储存于 WWW 服务器上的用 Java 语言编写的小程序，是 Java 语言使用最多的程序，也是 Java 语言快速推广与流行的重要因素。Java Applet 是动态的、安全的、跨平台的网络应用程序。Applet 被嵌入到 Web 页中用来产生特殊的页面效果。

Applet 从网络上进行传输，然后在支持 Java 的浏览器中运行。当用户在客户端的 Web 浏览器中访问嵌入了 Applet 资源的 Web 页时，指定的 Applet 字节码将下载到用户的 Web 浏览器执行。为了减少下载时间，提高下载速度，Applet 程序一般比较小。由于 Java 语言的安全机制，用户一旦载入 Applet，就可以放心地用来生成多媒体的用户界面或完成复杂的计算而不必担心病毒的入侵。虽然 Applet 可以和图象、声音、动画等一样从网络上下载，但它并不同于这些多媒体的文件格式，它可以接收用户的输入，动态地进行改变，而不仅仅是动画的显示和声音的播放，也就是说 Applet 具有较强的交互性。由于 Applet 是在用户的计算机上执行的，因此它的执行速度不受网络带宽或者 Modem 存取速度的限制，这样用户就可以更好地欣赏网页上 Applet 产生的多媒体效果。

【例 1-2】Java Applet 的简单实例。

```
import java.awt.*;
import java.applet.*;

public class HelloWorldApplet extends Applet {
    public void paint(Graphics g) {
        g.drawString ("Hello World!",20,20);
    }
}
```

这是一个简单的 Applet（小应用程序）。程序中，首先用 import 语句引入 java.awt 包和 java.applet 类库，使得该程序可以使用这些包和类库中所定义的类，它类似于 C 语言中的 #include 语句。然后声明一个公共类 HelloWorldApplet，用 extends 指明它是 Applet 的子类。在类中，重写父类 Applet 的 paint() 方法，其中参数 g 为 Graphics 类，它表明当前所画的内容。在 paint() 方法中，调用 g 的方法 drawString()，在坐标 (20,20) 处输出字符串“Hello World!”，其中坐标是用像素点来表示的。

这个程序中没有 main() 方法，这是 Applet 与应用程序 Application（如【例 1-1】）的明显区别。为了运行该程序，首先也要把它放在文件 HelloWorldApplet.java 中，然后用 javac HelloWorldApplet.java 对它进行编译，得到字节码文件 HelloWorldApplet.class。

由于 Applet 中没有 main() 方法作为 Java 解释器的入口，必须编写 HTML 文件，将该 Applet 嵌入其中，然后用 appletviewer 来运行，或在支持 Java 的浏览器上运行。下面简要介绍 HTML 语言。

超文本标记语言 HTML (Hyper Text Markup Language) 是一种与平台无关的描述性语言，其源代码由浏览器解释执行，可用任何一种文本编辑器编写 HTML 语言的代码。在 HTML 中不区分大小写字母，其显著的特点是在文本中加入标签对：“<”和“>”。常用的标签对如下：

<HTML>和</HTML>标签对：所有的 HTML 文件都以<HTML>开始、以</HTML>结束。

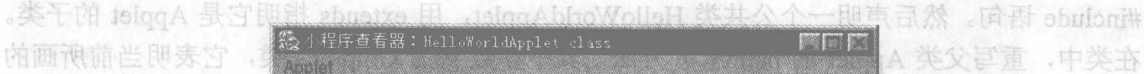
<BODY>和<BODY>主体标签对:为 HTML 文件的正文部分和基础,是网页的可视内容。

需要说明的是,在HTML文件中嵌入Java的小程序时,必须使用<APPLET>和</APPLET>标签对。

只用于浏览，特别对某些特殊 Modern 普通宽带网受不受限制时因出，因时土林真长
<HTML>
HEAD: HEAD: 果教本教条的土气 telnet 土页网赏知此致更以可德

中用 `<APPLET>` 标记来启动 `HelloWorldApplet`，把这个 HTML 文件存入 `2.HTML`，HTML 文件名可以与类名不相同。`Applet` 可在下面几种环境下运行：

输入 Appletviewer Example2.HTML 命令运行该程序，这时屏幕上弹出一个窗口，其中显示 Hello World!，显示结果如图 1-1 所示。



World" 其中坐科最显来点示的。

如果机器上安装了支持 Java 的浏览器,可直接使用浏览器调用这个 HTML 文件。打开

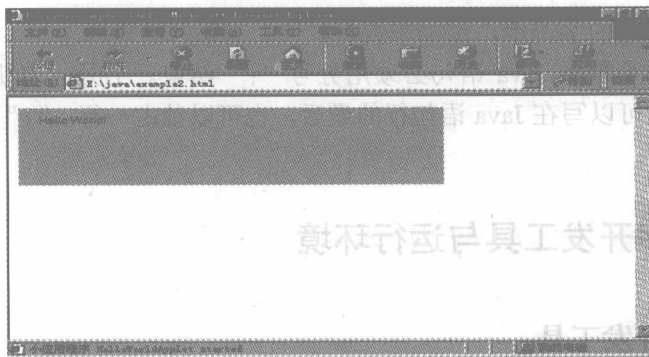


图 1-2 【例 1-2】的运行结果

1.2.3 Java 程序的结构

从上面的例子中可以看出，Java 程序是由类构成的。一个完整的 Java 程序可以包含一个或多个类，这些类可能是用户创建的或包含在 Java 类库内的。在用 Java 编译器编译时，每个类会产生与自己类名相同的一个.class 文件。

Java 程序包含类的声明及类定义体。在类的定义中，应包含类变量的声明和类中方法的实现两部分内容。在所有的方法中最特殊的方法是 main () 方法。Java 应用程序必须在主类中定义 main () 方法，main () 方法是程序运行的起点。而 Java Applet 中的任何一个类都不含有 main () 方法，所以 Java Applet 只能用支持 Java 的浏览器来运行。通过下面的例子来说明 Java 程序的书写规范。

【例 1-3】简单的 Java 应用程序。

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

在程序的第 1 行用关键字 class 定义了一个 HelloWorld 类，每个 Java 程序都是这样开头的。类名是一个字母开头的包含字母、数字、下划线 (_) 和 \$ 的字符串，类名必须与文件名相同。在上面的例子中包含一个 public 类型的名为 HelloWorld 的类，程序文件名必须命名为 HelloWorld.java。需要特别注意的是 Java 语言是区分大小写字母的语言。

在程序的第 1 行~第 5 行用一对大括号 “{” 和 “}” 定义类体。第 2 行包含了 4 个关键字：public 、static 、void 、main。public 说明 main () 方法可以被任何类和对象所共享；static 说明 main () 方法为静态方法；void 说明 main () 方法不返回任何值；main 是被定义的方法的名字。在关键字 main 后面的 () 中的内容是 main 方法的参数列表。

在程序的第 2 行和第 4 行用 “{” 和 “}” 定义了 main () 方法的方法体。程序的第 3 行是一条简单的可执行语句 System.out.println ("Hello World!"); 表示 main () 方法的具体内容是将引号内的字符串 Hello World! 显示在屏幕上。其中 System 是系统内部定义的一个类；out 是 System 子类中的对象；println 是 out 对象的一个方法，其作用是向系统的标准输

出设备输出其形参指定的字符串，用回车换行。

特别需要注意的是每条 Java 语句必须用分号“;”结束。另外，表明类体和方法体的大括号“{”和“}”既可以写在 Java 语句行的最后，也可以独占一行。“{”还可以写在 Java 语句行的前面。

1.3 Java 程序开发工具与运行环境

1.3.1 Java 程序开发工具

一个软件的开发过程包括：设计、编码、测试、调试、生成文档、维护和升级等若干个步骤。一个集成的软件开发环境是一套程序的集合，它能够方便软件的开发。目前，可用于开发 Java 程序的工具很多。常用的有 Sun 公司的 JDK，Microsoft 公司的 Visual J++，IBM 公司的 Visual Age，Borland 公司的 Jbuilder，Symantec 公司的 Visual Cafe 等。JDK 是免费的，其他的开发工具由于商业上的原因，与纯 Java 有一定的不兼容。作为初学者来说，最好还是使用 JDK。本书中所有的例子使用的都是 JDK。JDK 是 Java Development Kit 的缩写，即 Java 开发工具包。截止到 2003 年 1 月，美国 Sun 公司提供下载的 JDK 标准版软件最新正式版本为 j2sdk-1.4.1。不同的操作系统使用不同的开发版本。本书主要介绍 Windows95/98 版本。

1.3.2 JDK 的下载与安装

Java 不仅提供了一个丰富的语言和运行环境，而且还提供了一个免费的 Java 开发工具集（Java Development Kits，简称 JDK）。编程人员和最终用户可以利用这些工具来开发 Java 程序。美国 Sun 公司免费提供的 Java 开发工具（JDK）是命令行方式的，是许多 Java 专家最初使用的开发环境，尽管许多编程人员已经使用第三方的开发工具，但 JDK 仍被当作 Java 开发的重要工具。另外还有 Java WorkShop，是开发 Java 程序的集成环境。

可从网址 <http://www.javasoft.com> 处下载 JDK 开发工具 j2sdk-1.4.1。下载完毕后，运行 j2sdk-1.4.1-win.exe 就可进行开发工具的安装，安装时可指定安装盘和目录，也可安装到默认的路径中。本书中的所有例题都是在 j2sdk-1.4.1 下调试的。

JDK 的编程环境由一系列文件组成，主要包括：bin 目录、demo 目录、include 目录、jre 目录、lib 目录及 readme 文件等。安装 JDK 后产生的目录结构如图 1-3 所示。

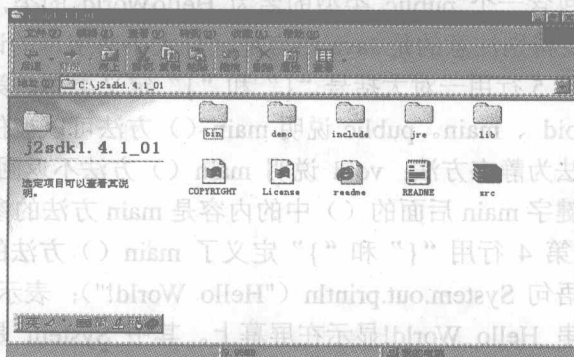


图 1-3 Windows 下的 JDK 安装目录