

高等学校规划教材

计算机类

Java 程序设计教程

JAVA CHENGXU SHEJI JIAOCHENG

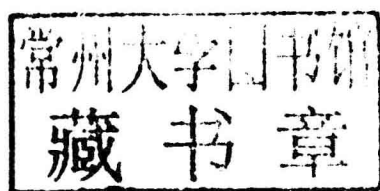
陈 蕴 主 编 ◀



北京师范大学出版集团
BEIJING NORMAL UNIVERSITY PUBLISHING GROUP
安徽大学出版社

Java 程序设计教程

陈 蕴 主编



北京师范大学出版集团
BEIJING NORMAL UNIVERSITY PUBLISHING GROUP
安徽大学出版社

图书在版编目(CIP)数据

Java 程序设计教程/陈蕴主编. —合肥:安徽大学出版社, 2013. 2

ISBN 978-7-5664-0337-7

I. ①J… II. ①陈… III. ①JAVA 语言—程序设计—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字(2011)第 219614 号

Java 程序设计教程

陈蕴 主编

出版发行: 北京师范大学出版集团
安徽大学出版社
(安徽省合肥市肥西路 3 号 邮编 230039)
www.bnupg.com.cn
www.ahupress.com.cn

经 销: 全国新华书店
印 刷: 中国科学技术大学印刷厂
开 本: 184mm×260mm
印 张: 20
字 数: 470 千字
版 次: 2013 年 2 月第 1 版
印 次: 2013 年 2 月第 1 次印刷
定 价: 36.00 元

ISBN 978-7-5664-0337-7

策划编辑: 蒋 芳
责任编辑: 蒋 芳
责任校对: 程中业

装帧设计: 李 军
美术编辑: 李 军
责任印制: 赵明炎

版权所有 侵权必究

反盗版、侵权举报电话: 0551-65106311

外埠邮购电话: 0551-65107716

本书如有印装质量问题, 请与印制管理部联系调换。

印制管理部电话: 0551-65106311

前言

Foreword Foreword Foreword Foreword

Java 程序设计语言是随着互联网的发展而被推广应用的,目前已成为国内外最为广泛使用的程序设计语言之一,并逐步成为计算机应用开发领域的主流语言。

为顺应社会的需求,推动计算机类教学的改革与实践,加快对 Java 程序设计语言的应用普及与提高,北京师范大学出版集团安徽大学出版社精心组织具有多年 Java 编程经历和丰富教学经验的教师集体编写了这本 Java 语言程序设计教材。

本书从 Java 语言的易懂易学特点出发,介绍 Java 面向对象的程序设计要点、难点和应用实例,力求结构严谨,由浅入深、循序渐进。在介绍基本概念和方法的同时,配有精心设计的实例,使读者能够更快地理解、应用该语言的各个功能,培养读者运用 Java 语言进行编程的实践能力。

全书共分 10 章。第 1、2、3 章主要介绍了 Java 程序设计中的基本概念,对初学者而言,掌握 Java 语言最基本的概念知识和重要的面向对象概念,对面向对象程序设计的思想在 Java 中的应用会有比较完整的认识。第 4 章介绍了 Java 中的一些常用工具,学习如何通过这些工具类对日期、时间、向量、堆栈等进行处理的基本方法。第 5 章介绍了小应用程序和用 awt、swing 组件进行图形用户界面设计的方法,学习并能设计出专业化图形用户界面。第 6、7、8 章分别介绍了 Java 特有的异常处理机制、多线程设计以及输入输出流的实现方法,为设计出完整、实用的程序提供保证。第 9、10 章讨论了 Java 在网络和数据库方面的应用,体现 Java 的实用性,提高读者学习 Java 的兴趣。读者在使用本书时要多上机练习,才能更好、更快地掌握使用 Java 语言进行程序设计的方法。

本书由陈蕴、刘冬冬、黄晓梅、李钧、张步忠、赵靖、钟娟编写,陈蕴教授通审、修改定稿。在本书的编写过程中,作者尽可能将同行和自己多年积累的经验、体会融入到教材的每个章节中,并在李梅、李镜平、蒋芳等多位同志共同努力下,使本书得以和读者见面。但是终因作者水平有限,书中不妥之处在所难免,竭诚期望读者多提宝贵意见。电子邮箱:chenyun_aju@163.com。在此对本书的出版给以支持和帮助的同行和朋友表示感谢!

本书可作为高等院校计算机及其相关专业的教材,也可作为高等院校程序设计公共选修课的教材。

陈 蕴

2012 年 8 月

目 录

Catalog Catalog Catalog Catalog

第 1 章 Java 概述 (1)

1.1 Java 语言简介	(1)
1.1.1 Java 语言的发展	(1)
1.1.2 Java 的工作原理	(3)
1.2 编写 Java 程序	(5)
1.2.1 建立 Java 开发环境	(6)
1.2.2 Java 程序规范	(12)
1.2.3 接触 Java 程序	(17)
1.3 小结	(26)
习题 1	(26)

第 2 章 Java 语言基础 (27)

2.1 Java 的关键字、标识符、各种数据类型	(27)
2.1.1 关键字	(27)
2.1.2 标识符	(28)
2.1.3 数据类型	(28)
2.2 Java 的各种运算符和表达式	(30)
2.2.1 算术运算符	(30)
2.2.2 关系运算符	(30)
2.2.3 位运算符	(31)
2.2.4 其他运算符	(31)



2.2.5 运算符的优先级与结合性	(32)
2.3 类型转换	(33)
2.4 语句	(34)
2.4.1 表达式语句	(35)
2.4.2 复合语句	(35)
2.4.3 选择语句	(35)
2.4.4 switch 开关语句	(38)
2.4.5 循环语句	(39)
2.4.6 跳转语句	(41)
2.5 数组	(45)
2.5.1 数组声明	(45)
2.5.2 创建数组空间	(45)
2.5.3 数组元素的引用及初始化	(46)
2.6 字符和字符串的处理	(46)
2.6.1 字符	(46)
2.6.2 字符串	(47)
2.7 小结	(51)
习题 2	(51)

第 3 章 面向对象的 Java

(53)

3.1 介绍类	(53)
3.1.1 类基础	(53)
3.1.2 声明对象	(57)
3.1.3 给对象引用变量赋值	(58)
3.1.4 方法	(59)
3.1.5 构造方法	(64)
3.1.6 this 关键字	(67)
3.1.7 垃圾回收	(68)
3.1.8 finalize() 方法	(68)
3.1.9 一个堆栈类	(69)

3.2 类的高级话题	(71)
3.2.1 方法重载	(71)
3.2.2 对象作为参数	(77)
3.2.3 参数是如何传递的	(80)
3.2.4 返回对象	(82)
3.2.5 递归	(83)
3.2.6 访问控制	(85)
3.2.7 理解 static	(88)
3.2.8 介绍 final	(90)
3.2.9 重新温习数组	(91)
3.2.10 介绍嵌套类和内部类	(93)
3.2.11 探索 String 类	(96)
3.2.12 命令行参数	(98)
3.3 类间继承	(99)
3.3.1 继承的基础	(99)
3.3.2 使用 super	(105)
3.3.3 创建多级类层次	(110)
3.3.4 何时调用构造方法	(114)
3.3.5 方法重载	(115)
3.3.6 动态方法调度	(118)
3.3.7 抽象类	(122)
3.3.8 继承中使用 final	(124)
3.3.9 Object 类	(126)
3.4 包和接口	(126)
3.4.1 包	(126)
3.4.2 访问保护	(129)
3.4.3 引入包	(132)
3.4.4 关键字 interface	(134)
3.4.5 接口可以扩展	(144)
3.5 小结	(145)
习题 3	(145)

第 4 章 常用工具类

(146)

4.1 Data 类	(146)
4.2 Calendar 类	(148)
4.3 Math 类	(149)
4.4 向量类 Vector	(150)
4.5 栈类 Stack	(153)
4.6 哈希表类 Hashtable	(154)
4.7 小结	(156)
习题 4	(156)

第 5 章 图形用户界面

(157)

5.1 AWT 实现用户界面	(157)
5.1.1 java.awt 包结构	(158)
5.1.2 容器类和组件类	(158)
5.1.3 常用的容器组件类	(159)
5.1.4 布局管理类	(167)
5.1.5 AWT 事件处理	(174)
5.1.6 常用 AWT 组件类	(183)
5.2 Swing 实现用户界面	(196)
5.2.1 Swing 简介	(196)
5.2.2 Swing 组件和容器	(198)
5.2.3 选择界面风格	(203)
5.3 用户界面在 Applet 中的实现	(203)
5.3.1 Applet 的基本工作原理	(203)
5.3.2 Applet 小程序中实现用户界面	(206)
5.4 小结	(209)
习题 5	(209)

第 6 章 异常处理

(211)

6.1 异常与异常类	(211)
6.1.1 异常	(211)
6.1.2 异常类	(212)
6.2 捕获和处理异常	(214)
6.2.1 捕获异常	(215)
6.2.2 抛出异常	(217)
6.3 自定义异常类	(218)
6.3.1 自定义异常	(218)
6.3.2 举例分析	(219)
6.4 小结	(220)
习题 6	(220)

第 7 章 Java 多线程机制

(221)

7.1 Java 中的线程	(221)
7.1.1 线程的概念	(221)
7.1.2 线程的结构	(222)
7.1.3 线程的状态	(222)
7.2 创建线程	(223)
7.2.1 Thread 类	(223)
7.2.2 Runnable 接口	(225)
7.3 线程控制	(227)
7.3.1 线程调度	(227)
7.3.2 结束线程	(227)
7.3.3 线程挂起	(228)
7.3.4 终止线程	(228)
7.3.5 测试线程状态	(228)
7.4 线程同步	(228)
7.4.1 synchronized 关键字	(228)

7.4.2 wait、notify 和 notifyAll	(229)
7.4.3 同步实例	(230)
7.5 小结	(234)
习题 7	(234)

第 8 章 流式输入输出与文件处理 (235)

8.1 流和流类的层次结构	(235)
8.1.1 流的概念	(235)
8.1.2 流类的层次结构	(236)
8.2 标准输入/输出流	(237)
8.2.1 标准输入流	(237)
8.2.2 标准输出流	(238)
8.3 InputStream 类	(239)
8.4 OutputStream 类	(240)
8.5 Reader 类和 Writer 类	(241)
8.6 File 类	(245)
8.7 文件的输入和输出	(247)
8.8 管道流	(249)
8.9 过滤流	(250)
8.10 小结	(253)
习题 8	(254)

第 9 章 Java 网络编程 (255)

9.1 网络相关概念	(255)
9.1.1 TCP/IP 协议	(255)
9.1.2 IP 地址	(255)
9.1.3 通信端口	(256)
9.1.4 套接字	(256)
9.1.5 URL	(256)

9.1.6	Java 网络通信	(257)
9.2	URL 类	(257)
9.1.1	构造方法	(257)
9.2.2	常用方法	(258)
9.3	URLConnection 类	(259)
9.4	InetAddress 类	(262)
9.4.1	InetAddress 对象创建方法	(262)
9.4.2	常用方法	(263)
9.5	Socket 编程	(263)
9.5.1	创建套接字对象	(264)
9.5.2	通信过程	(265)
9.5.3	实例	(266)
9.6	UDP 编程	(268)
9.6.1	DatagramPacket 类	(269)
9.6.2	DatagramSocket 类	(269)
9.6.3	数据收发流程	(270)
9.6.4	实例程序	(272)
9.7	数据广播	(274)
9.8	小结	(276)
习题 9		(277)

第 10 章 Java 数据库操作

(278)

10.1	数据库访问接口	(278)
10.1.1	ODBC	(278)
10.1.2	JDBC	(279)
10.2	JDBC 体系结构	(279)
10.2.1	JDBC 结构	(279)
10.2.2	JDBC 驱动程序类型	(281)
10.2.3	JDBC API	(282)

10.3 数据库连接过程	(283)
10.3.1 JDBC URL	(283)
10.3.2 数据库操作流程	(284)
10.3.3 数据库连接流程	(284)
10.4 JDBC-ODBC 桥	(287)
10.5 数据管理	(289)
10.5.1 数据查询	(290)
10.5.2 增加记录	(292)
10.5.3 修改记录	(293)
10.5.4 删除数据	(294)
10.5.5 事务处理	(294)
10.5.6 滚动查询	(295)
10.6 综合实例	(297)
10.7 小结	(306)
习题 10	(306)

参考文献	(307)
-------------------	-------

第 1 章 Java 概述

Java 是一门优秀的面向对象编程语言,具有“一次写成,处处运行”的特点,特别适合于 Internet 的应用开发,因此 Java 已成为网络时代重要的编程语言之一。本章首先介绍 Java 的产生,然后介绍 Java 语言的特点、工作原理、虚拟机和三种开发平台,接下来介绍 Java 的开发环境,即 JDK 的安装和配置,并以实例的方式介绍使用记事本和 JCreator 开发 Java 程序的方法。

本章主要学习目标是:

- 了解 Java 的特点。
- 理解 Java 的编程规范。
- 掌握使用 Java 帮助文档。
- 掌握 JDK 的安装和配置。
- 熟练掌握 Java 应用程序和小应用程序的编写。

1.1 Java 语言简介

Java 是一种既可面向对象又可跨平台的语言。Java 语言产生于 C++ 语言之后,是完全面向对象的编程语言。Java 语言充分吸取 C++ 语言的优点,采用程序员所熟悉的 C 和 C++ 语言的许多语法,同时又去掉了 C++ 语言中指针、内存申请和释放等影响程序健壮性的部分。可以说,Java 语言是站在 C++ 语言这个“巨人”的肩膀上前进的。本节的主要内容是介绍 Java 语言的产生、地位,Java 虚拟机及其特点,以及 Java 的三种平台的主要构成。

1.1.1 Java 语言的发展

1. Java 语言的产生

Java 语言来源于 Sun Microsystems 公司的 Green 项目,该项目最初的目的是为家用消费电子产品开发一个分布式代码系统,以使用户将 E-mail 发送给电冰箱、电视机、烤箱等家用电器,对它们进行控制,并和它们进行信息交换。在项目研制初始阶段,项目组成员准备采用 C++ 语言开发该系统,但是 C++ 语言遇到了前所未有的挑战,因为它太复杂而且安全性差。所以最后项目组成员基于 C++ 语言开发了一种新的语言 Oak(橡树),这就是 Java 语言的前身。Oak 是一种用于网络的精巧而又安全的语言,Sun 公司曾依此投标了一个交互式电视节目,但结果被 SGI 打败。正当 Oak 无家可归之时,Mark Andreessen 开发的 Mosaic 和 Netscape 启发了 Oak 项目组成员,他们用此技术编制了 HotJava 浏览器,并得到了 Sun 公司首席执行官 Scott McNealy 的支持,开启了 Java 进军 Internet 的契机。1994 年,Internet 的迅猛发展和 WWW 的快速增长,促进了 Java 语言研制的进展,使得它逐渐成为 Internet 上受欢迎的开发与编程语言。1995 年 5 月,Sun 公司正式发布 Java 语言的第一

个办公版本。

作为新一代的面向对象的程序设计语言,Java 备受青睐。一些著名的计算机公司纷纷购买了 Java 语言的使用权,如 IBM、Netscape、Novell、Apple、DEC、SGI、Oracle 等,甚至包括最不情愿的 Microsoft。

Java 语言被美国著名杂志 PC Magazine 评为 1995 年十大优秀科技产品(计算机类仅此一项入选)。微软公司总裁 Bill Gates 不无感慨地说:“Java 是长时间以来最卓越的程序设计语言。”Sun 公司的总裁 Scott McNealy 认为:“Java 为 Internet 和 WWW 开辟了一个崭新的时代。”万维网(WWW)的创始人 Berners-Lee 说:“计算机事业发展的下一个浪潮就是 Java,并且将很快会发生。”甚至有人预言:“Java 将是网络上的‘世界语’,今后所有的用其他语言编写的软件统统都要用 Java 语言来改写。”

2. Java 语言在网络时代的地位

Java 语言特别适合于 Internet 应用程序开发。一时间“连 Internet,用 Java 编程”,成为技术人员的一种时尚。虽然新闻界的报导可能言过其实,但 Java 作为软件开发的一种革命性的技术,其地位已被确立,这表现在以下几个方面:

(1)计算机产业的许多大公司购买了 Java 的许可证,包括 IBM、Apple、DEC、Adobe、HP、Oracle、Toshiba,以及 Microsoft。这一点说明,Java 已得到了业界的认可。

(2)众多的软件开发商都支持 Java 的软件产品。今天是以网络为中心的计算机时代,不支持 HTML 和 Java 的应用程序的应用范围只能限于同质的环境(相同的硬件平台)。

(3)Intranet 成为企业信息系统最佳的解决方案,而其中 Java 将发挥不可替代的作用。Intranet 的目的是把 Internet 用于企业内部的信息系统,它的优点表现在:便宜,易于使用和管理。用户不管使用何种类型的机器和操作系统,界面都是统一的浏览器,而数据库、Web 页面、应用程序(用 Java 编写的 Applet)则存在 WWW 服务器上。因此无论是开发人员,还是管理人员,或是用户都可以受益于该解决方案。

Java 语言正在不断发展和完善,Sun 公司是主要的发展推动者。较通用的编译环境有 JDK(Java Develop Kit)与 JWS(Java Workshop),还有很多其他公司开发 Java 语言的编译器与集成环境(例如:JCreator、Eclipse 集成开发环境)。现在用 Java 编程和用 C++ 编程一样方便。

3. 三种平台简介

1999 年,Sun 公司根据应用的领域不同把 Java 技术划归为三个平台,当时分别称为 J2SE、J2EE 和 J2ME,现在改名为 Java SE、Java EE 和 Java ME。

(1)Java SE:Java SE 是 Java 平台标准版(Java Platform Standard Edition)的简称,用于开发和部署桌面、低端服务器以及实时环境中的 Java 应用程序。Java SE 包含那些构成 Java 语言核心的类,比如数据库连接、接口定义、输入输出、网络编程等;同时,Java SE 为 Java EE 提供了基础。

(2)Java EE:Java EE 是 Java 平台企业版(Java Platform Enterprise Edition)的简称,用于开发便于组装、健壮、可扩展、安全的服务器端 Java 应用。Java EE 建立于 Java SE 之上,具有 Web 服务、组件模型以及通信 API 等特性,这些为面向服务的架构(SOA)以及开发 Web 2.0 应用提供了支持。Java EE 基于 Java SE,此外新加了企业应用所需的类库。

(3)Java ME:Java ME 是 Java 微版(Java Platform Micro Edition)的简称,是一个技术

和规范的集合,它为移动设备(包括消费类产品、嵌入式设备、高级移动设备等)提供了基于 Java 环境的开发与应用平台。Java ME 目前分为两类配置:一类是面向小型移动设备的 CLDC(Connected Limited Device Configuration);另一类是面向功能更强大的移动设备如智能手机和机顶盒,称为 CDC(Connected Device Configuration)。Java ME 有自己的类库,其中 CLDC 使用的是被称为 KVM 的专用的 Java 虚拟机。

1.1.2 Java 的工作原理

Java 是一种跨平台的语言,同时也是既编译又解释的语言。Java 可以运行在网络中不同的计算机上,但运行的计算机上必须装有相应的 Java 解释器。这种解释器称为 Java 虚拟机(JVM)。用 Java 编写程序,其运行的流程大致如图 1-1 所示。

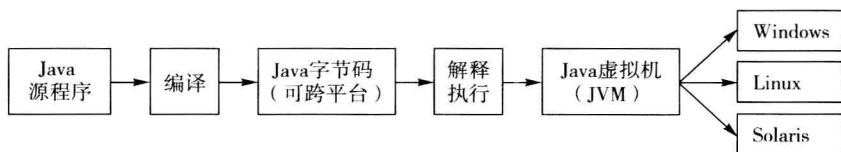


图 1-1 Java 程序运行流程

1. Java 虚拟机

Java 主要靠 Java 虚拟机(JVM)来实现平台无关性。JVM 是一个想象中的机器,在实际的计算机上通过软件模拟来实现。JVM 有自己想象中的硬件,如处理器、堆栈、寄存器等;还具有相应的指令系统,它安装在具体操作系统之上。Java 语言的一个非常重要的特点就是与平台的无关性,而使用 JVM 是实现这一特点的基础与关键。一般的高级语言如果要在不同的平台上运行,至少需要编译成不同的目标代码;而引入 JVM 后,Java 语言在不同平台上运行时不需要重新编译。Java 语言使用 JVM 屏蔽了与具体平台相关的信息,使得 Java 语言编译程序只需生成在 JVM 上运行的目标代码(字节码),就可以在多种平台上不加修改地运行。JVM 在执行字节码时,把字节码解释成具体平台上的机器指令执行。另外,由于 Java 采用的是基于 IEEE 标准的数据类型,则通过 JVM 保证了数据类型的一致性,也确保了 Java 的平台无关性。JVM 是 Java 语言底层实现的基础,对 Java 语言感兴趣的人都应对 JVM 有个大概的了解。这有助于理解 Java 语言的一些性质,也有助于使用 Java 语言。对于要在特定平台上实现 JVM 的软件人员、Java 语言的编译器作者以及要用硬件芯片实现 JVM 的人来说,则必须深刻理解 JVM 的规范。另外,如果想扩展 Java 语言,或是把其他语言编译成 Java 语言的字节码,也需要深入地了解 JVM。

JVM 在计算机系统上的位置如图 1-2 所示。

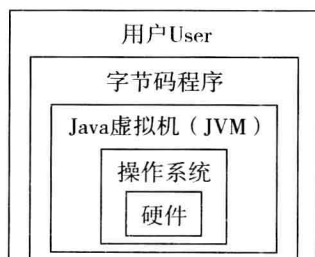


图 1-2 JVM 在计算机中的位置

2. Java 语言的特点

Java 是一门各方面性能都很好的编程语言,它的基本特点是简单、面向对象、分布式、健壮、结构中立、安全、可移植、高效解释执行、高性能、多线程、动态及自动垃圾回收机制。Java 十分适合在 Internet 环境上开发应用系统,可以制作大部分网络应用程序系统,而且与当今流行的 WWW 浏览器结合得很好。

(1) 简单。Java 最初是对家用电器进行集成控制而设计的一种语言,它通过提供最基本的方法来完成指定的任务。因此,只需理解一些基本的概念,就可以用它编写出适合于各种情况的应用程序。Java 语言的简单性主要体现在以下三个方面。

① Java 的风格类似于 C++,因而 C++ 程序员很容易接受。从某种意义上讲,Java 语言是 C 及 C++ 语言的一个变种,因此, C++ 程序员可以很快就掌握 Java 编程技术。

② Java 摒弃了 C++ 中容易引发程序错误的地方,如指针。另外,Java 略去了运算符重载、多重继承等模糊的概念,并且通过自动回收无用信息,大大简化了程序设计者的内存管理工作。

③ Java 提供了丰富的类库。

(2) 面向对象。面向对象可以说是 Java 最重要的特性。Java 语言的设计完全是面向对象的,它不支持类似 C 语言那样的面向过程的程序设计技术。Java 支持代码继承及重用。单从面向对象的特性来看,Java 尤其适用于分布式计算环境。

(3) 分布式。Java 是面向网络的语言。Java 包括一个支持 HTTP 和 FTP 等基于 TCP/IP 协议的子库,通过提供的类库可以处理 TCP/IP 协议,因此,Java 应用程序可凭借 URL 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。Java 的语法特性使我们很容易地为分布环境尤其是 Internet 提供动态内容。

(4) 健壮。Java 在编译和运行程序时,都要对可能出现的问题进行检查,以消除错误的产生。它提供自动无用信息收集来进行内存管理,防止程序员在管理内存时产生错误。在编译时,Java 通过集成的面向对象的异常(Exception)处理机制,揭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。另外,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。

(5) 结构中立。Java 将它的程序编译成一种结构中立的中间文件格式。只要有 Java 运行系统的机器都能执行这种中间代码。Java 源程序被编译成一种高层次的与机器无关的字节代码文件,这种中间代码在虚拟机上运行,由机器相关的运行调试器实现执行。

(6) 安全。用于网络、分布环境下的 Java 必须要防止病毒的入侵。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,这样就防止程序员使用特洛伊木马等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。另外,当 Java 用来创建浏览器时,语言功能和浏览器本身提供的功能结合起来,使它更安全。

(7) 可移植。同体系结构无关的特性使得 Java 应用程序可以在配备了 Java 解释器和运行环境的任何计算机系统上运行,这成为 Java 应用软件便于移植的良好基础。但仅仅如此还不够。如果基本数据类型设计依赖于具体实现,也将为程序的移植带来很大不便。例如在 Windows 3.1 中整数(Integer)为 16 bit,在 Windows 95 中整数为 32 bit,在 DEC Alpha 中整数为 64 bit,在 Intel 486 中整数为 32 bit。通过定义独立于平台的基本数据类型及其运算,使 Java 数据得以在任何硬件平台上保持一致。Java 语言的基本数据类型及其表



示方式如表 1-1 所示。

表 1-1 Java 语言的基本数据类型及其表示方式

数据类型	长度 (bit)
byte	8bit
short	16bit
int	32bit
long	64bit
float	32bit
double	64bit
char	16bit

在 Java 语言中,数据类型都是依据以上标准具体实现的。因为几乎目前使用的所有 CPU 都能支持以上数据类型 8~64 位整数格式的补码运算和单/双精度浮点运算。

(8) 高效解释执行。Java 解释器(运行系统)能直接运行目标代码指令。链接程序通常比编译程序所需资源少,使得连接过程更加简单,所以程序员可以在创建源程序上花上更多精力。

(9) 高性能。如果解释器速度不慢,Java 可以在运行时直接将目标代码翻译成机器指令。Sun 用直接解释器一秒钟内可调用 30 万个过程。翻译目标代码的速度与 C/C++ 的性能没什么区别。

(10) 多线程。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。线程——有时也称小进程——是一个大进程里分出来的小的独立的进程。因为 Java 实现的多线程技术,所以比 C 和 C++ 更健壮。多线程带来的更大的好处是更好的交互性能和实时控制性能。当然实时控制性能还取决于系统本身(UNIX、Windows、Macintosh 等),在开发难易程度和性能上都比单线程要好。任何用过当前浏览器的人,都感觉为调用一幅图片而等待是一件很烦恼的事情。在 Java 里,可用一个单线程来调一幅图片,同时可以访问 HTML 里的其他信息而不必等它。

(11) 动态。Java 的设计使它适合于一个不断发展的环境。在类库中可以自由地加入新的方法和实例变量而不会影响用户程序的执行。另外,Java 通过接口来支持多重继承,使之比严格的类继承具有更灵活的方式和扩展性。

(12) 自动垃圾回收机制。Java 语言中一个显著的特点就是引入了垃圾回收机制,使 C++ 程序员最感头疼的内存管理的问题迎刃而解,它使得 Java 程序员在编写程序的时候不再需要考虑内存管理。由于有个垃圾回收机制,Java 中的对象不再有“作用域”的概念,只有对象的引用才有“作用域”。垃圾回收可以有效地防止内存泄露。垃圾回收器通常是作为一个单独的低级别的线程运行,不可预知的情况下对内存堆中已经死亡的或者长时间没有使用的对象进行清除和回收。程序员不能实时地调用垃圾回收器对某个对象或所有对象进行垃圾回收。

1.2 编写 Java 程序

编写 Java 程序之前,计算机上必须安装 Java SDK(Java Software Develop Kit),该软件