



高等学校计算机专业“十一五”规划教材

Java 程序设计教程

赵 莉 杨国梁 孙喆喆 徐 飞 编著
张白一 主审



西安电子科技大学出版社
<http://www.xduph.com>

高等学校计算机专业“十一五”规划教材

Java 程序设计教程

赵 莉 杨国梁 孙喁喁 徐 飞 编著

张白一 主审

西安电子科技大学出版社

2009

内 容 简 介

本书全面介绍了 Java 语言程序设计方法, 主要内容包括 Java 语言基础知识、面向对象基本理论知识与编程技术、数组与字符串、常用系统类的使用、图形用户界面设计、网络编程、异常处理、多线程与并发编程、套接字和网络、输入/输出和文件操作、数据库编程等。书中以实例的形式解释了 Java 面向对象编程思想, 介绍了 Java 编程技巧。本书注重循序渐进、由浅入深的原则, 每一个理论概念都通过详尽的实例演示加以说明。

本书适合作为高等院校计算机及相关专业的 Java 程序设计或面向对象程序设计课程的教材, 也适合专业技术人员参考。

★本书配有电子教案, 需要者可登录出版社网站, 免费下载。

图书在版编目(CIP)数据

Java 程序设计教程 / 赵莉等编著. —西安: 西安电子科技大学出版社, 2009. 8

高等学校计算机专业“十一五”规划教材

ISBN 978 - 7 - 5606 - 2313 - 9

I. J… II. 赵… III. Java 语言—程序设计—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2009)第 119132 号

策 划 臧延新

责任编辑 南 景 臧延新

出版发行 西安电子科技大学出版社(西安市太白南路 2 号)

电 话 (029)88242885 88201467 邮 编 710071

网 址 www.xduph.com 电子邮箱 xdupfb001@163.com

经 销 新华书店

印刷单位 西安文化彩印厂

版 次 2009 年 8 月第 1 版 2009 年 8 月第 1 次印刷

开 本 787 毫米×1092 毫米 1/16 印 张 17.375

字 数 405 千字

印 数 1~4000 册

定 价 25.00 元

ISBN 978 - 7 - 5606 - 2313 - 9/TP · 1172

XDUP 2605001-1

*** 如有印装问题可调换 ***

本社图书封面为激光防伪覆膜, 谨防盗版。

高等学校计算机专业“十一五”规划教材

编审专家委员会

主 任: 马建峰 (西安电子科技大学计算机学院院长, 教授)

副主任: 赵祥模 (长安大学信息工程学院院长, 教授)

余日泰 (杭州电子科技大学计算机学院副院长, 副教授)

委 员: (按姓氏笔画排列)

王忠民 (西安邮电学院计算机系副主任, 教授)

王培东 (哈尔滨理工大学计算机与控制学院院长, 教授)

石美红 (西安工程大学计算机科学与技术系主任, 教授)

纪 震 (深圳大学软件学院院长, 教授)

刘卫光 (中原工学院计算机学院副院长, 教授)

陈 以 (桂林电子科技大学计算机与控制学院副院长, 副教授)

张尤赛 (江苏科技大学电子信息学院副院长, 教授)

邵定宏 (南京工业大学信息科学与工程学院副院长, 教授)

张秀虹 (青岛理工大学计算机工程学院副院长, 教授)

张焕君 (沈阳理工大学信息科学与工程学院副院长, 副教授)

张瑞林 (浙江理工大学信息电子学院副院长, 教授)

李敬兆 (安徽理工大学计算机科学与技术学院院长, 教授)

范 勇 (西南科技大学计算机学院副院长, 副教授)

陈庆奎 (上海理工大学计算机学院副院长, 教授)

周维真 (北京信息科技大学计算机学院副院长, 教授)

徐 苏 (南昌大学计算机系主任, 教授)

姚全珠 (西安理工大学计算机学院副院长, 教授)

徐国伟 (天津工业大学计算机技术与自动化学院副院长, 副教授)

容晓峰 (西安工业大学计算机学院副院长, 副教授)

龚尚福 (西安科技大学计算机系主任, 教授)

策 划: 臧延新 云立实

杨 璠 陈 婷

前 言

Java 是一种广泛使用的网络编程语言。首先，作为一种程序设计语言，它简单、面向对象、不依赖于机器的结构，具有可移植性、鲁棒性、安全性，提供了并发的机制，并且具有很高的性能；其次，它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 和环境的限制。另外，Java 还提供了丰富的类库，使程序员可以很方便地建立自己的系统。Java 语言最主要的特点就是“Write once, run anywhere”，这句话一直是 Java 程序员的精神指南，也是 Java 语言深得程序员喜爱的原因之一。

本书循序渐进，由浅入深，在编写过程中，语言组织符合学生的思维习惯，对于重点问题通过大量的实例图文并茂地加以阐述，力求做到通俗易懂、言简意赅。书中的每个实例均上机调试过，便于读者通过实际上机操作进一步理解 Java 的原理、功能、作用。本书内容全面，在章节编排上做了精心设计和安排。首先对 Java 的基本语法进行介绍，在此基础上，将面向对象的基本概念、理论结合 Java 以具体示例形式进一步说明，使得基础理论的学习清晰明了，学生学习时也能事半功倍，为后续课程的学习打下良好的基础；其次，对 Java 高级程序设计进行介绍，包含了异常处理、图形用户界面、多线程、网络编程、数据库应用等方面。在每一章节的编排中，均结合具体示例加以阐述。各章具体内容介绍如下：

第 1 章全面介绍了 Java 语言的基本情况，介绍 JDK 的安装与设置，并对可扩展的免费开源代码的集成开发环境 Eclipse 平台的安装使用进行了简单介绍。

第 2 章对 Java 语法进行了详细介绍，这是学习 Java 必须掌握的基本内容。

第 3 章是 Java 面向对象编程的内容，深入浅出地介绍了面向对象编程的核心：对象、类、子类、继承、多态等基本概念。

第 4 章对常用对象的数组与字符串进行了详细介绍。

第 5 章介绍了常用系统类，如何掌握更多的 Java 类库和方法，并对接口及多继承进行了详细说明。

第 6 章首先介绍了 Java 的异常处理机制以及如何实现这种机制，然后介绍了怎样利用 Java 提供的异常类处理异常，以及如何定义新的异常类。

第 7 章介绍了 AWT 事件处理机制、布局管理器及常见 Swing 组件的使用方法，讲述基本的图形、图像处理功能以及简单动画的生成方法。

第 8 章主要介绍了 Java 语言的流输入/输出处理机制和文件的处理机制。输入/输出是计算机最基本的操作。Java 提供了丰富的输入/输出流操作类，功能强大。

第 9 章介绍了多线程技术，它是 Java 的主要特色之一，线程是 Java 中的重要内容，利用多线程编程具有一定的复杂性。

第 10 章介绍了基于 java.net 包进行网络通信程序设计，主要介绍了 URL 网络编程中常

用的 URL 类和 URLConnection 类, 重点介绍了用于编写客户端/服务器程序的 Socket 通信。

第 11 章介绍了四种类型的 JDBC 驱动程序以及按照 JDBC 编程步骤实现 JDBC 在数据库访问中的应用, 通过一个具体实例讲解了 JDBC 在数据库连接中的实际应用。

本书适合作为高等院校计算机及相关专业的 Java 程序设计或面向对象程序设计课程的教材, 也适合专业技术人员参考。

本书在编写过程中, 参考了大量文献、相关著作以及网络上的最新资料, 在此表示衷心的感谢!

编 者

2009 年 5 月

目 录

第 1 章 绪论	1
1.1 Java 发展简史	1
1.2 Java 的特点	2
1.3 Java 开发环境	5
1.3.1 JDK 的安装与设置	5
1.3.2 运行 Java 程序	8
1.3.3 Eclipse 介绍	11
本章小结	12
习题	12
第 2 章 Java 语言基础	13
2.1 标识符与关键字	13
2.2 注释和分隔符	13
2.2.1 注释	13
2.2.2 分隔符	14
2.3 基本数据类型	14
2.4 变量与常量	16
2.4.1 变量	16
2.4.2 常量	17
2.4.3 整型数据	17
2.4.4 浮点型数据	17
2.4.5 字符型数据	18
2.4.6 布尔型数据	19
2.4.7 各类数值型数据间的混合运算	19
2.5 运算符与表达式	20
2.5.1 算术运算符	20
2.5.2 关系运算符	24
2.5.3 逻辑运算符	25
2.5.4 位运算符	26
2.6 程序基本结构	28
2.6.1 选择语句	29
2.6.2 循环语句	34
2.6.3 跳转语句	39

本章小结	43
习题	43
第3章 Java 面向对象程序设计	45
3.1 面向对象程序设计基础	45
3.1.1 结构化程序设计	45
3.1.2 面向对象程序设计方法及特征	46
3.2 类和对象	50
3.2.1 类的定义	50
3.2.2 类成员	51
3.2.3 类的使用	52
3.2.4 数据成员和成员方法的使用	54
3.3 类的封装性	55
3.3.1 类的构造方法	56
3.3.2 this 和 super 引用	57
3.3.3 类成员的访问权限修饰符	60
3.3.4 实例成员与类成员	62
3.4 类的继承性	64
3.5 类的多态性	65
3.5.1 覆盖和重载	66
3.5.2 运行时多态	70
3.6 静态修饰符、静态数据成员和静态成员方法	73
3.6.1 静态修饰符	73
3.6.2 静态数据成员	74
3.6.3 静态常量	76
3.6.4 静态成员方法	76
3.7 抽象类和最终类	77
3.7.1 抽象类	77
3.7.2 最终类	79
本章小结	79
习题	80
第4章 数组与字符串	83
4.1 数组	83
4.1.1 一维数组	83
4.1.2 二维数组	86
4.1.3 不规则数组	89
4.1.4 数组实用类 Arrays	89
4.2 字符串	92
4.2.1 String 类	92
4.2.2 StringBuffer 类	97

4.2.3 String 类与 StringBuffer 类的比较	100
本章小结	100
习题	101
第 5 章 接口、内部类和包	103
5.1 接口	103
5.1.1 接口定义	103
5.1.2 实现接口	104
5.2 内部类	107
5.2.1 成员类	108
5.2.2 静态内部类	109
5.2.3 局部内部类	110
5.2.4 匿名内部类	111
5.3 包	112
5.3.1 包的创建	112
5.3.2 包的引入	114
5.3.3 设置包的路径	114
5.3.4 包的作用	115
5.4 Java API 基础	115
5.4.1 Java 提供的包概述	115
5.4.2 java.lang 语言包	116
5.4.3 java.util 实用包	118
本章小结	121
习题	121
第 6 章 异常处理	122
6.1 异常的概念与异常处理	122
6.1.1 程序错误、发现时刻及错误处理原则	122
6.1.2 Java 的错误和异常	123
6.2 Java 的异常处理	124
6.2.1 Java 中的 Throwable 类	124
6.2.2 try-catch-finally 语句	125
6.2.3 throw 和 throws 语句	129
6.3 自定义异常处理类	131
6.4 应用举例	133
本章小结	136
习题	136
第 7 章 图形用户界面	137
7.1 Java 图形用户界面	137
7.1.1 AWT 组件概述	137
7.1.2 Applet 程序	138

7.2 事件处理	138
7.2.1 AWT 的委托事件模型	139
7.2.2 AWTEvent 事件类	142
7.2.3 事件监听器接口和事件适配器类	144
7.3 布局管理	150
7.3.1 BorderLayout 布局管理器	151
7.3.2 CardLayout 布局管理器	152
7.3.3 FlowLayout 布局管理器	154
7.3.4 GridLayout 布局管理器	156
7.3.5 GridBagLayout 布局管理器	158
7.4 Swing 组件	161
7.4.1 Swing 组件关系概述	161
7.4.2 窗口组件	162
7.4.3 文本显示和编辑组件	164
7.4.4 按钮组件	166
7.4.5 列表框和组合框组件	167
7.4.6 菜单组件	170
7.5 图形处理	174
7.5.1 绘图类	174
7.5.2 绘图方法	175
本章小结	177
习题	178
第 8 章 输入/输出流和文件操作	180
8.1 流和文件的概念	180
8.1.1 操作系统中的文件和目录概念	180
8.1.2 流的概念	181
8.1.3 Java 的输入/输出流概述	181
8.2 字节输入/输出流类	183
8.2.1 概述	183
8.2.2 Java 的标准输入/输出	183
8.2.3 文件输入/输出流类	184
8.2.4 数据字节输入/输出流类	188
8.2.5 对象输入/输出流类	190
8.2.6 缓冲字节流	193
8.3 字符流类	194
8.3.1 字符输入/输出流抽象类	194
8.3.2 文件字符输入/输出流类	195
8.3.3 字符输入/输出流类	196
8.3.4 字符缓冲流类	200

8.4 文件操作类	201
8.4.1 文件类	201
8.4.2 文件过滤器接口	203
8.4.3 随机存取文件类	205
本章小结	206
习题	206
第 9 章 多线程	207
9.1 Java 线程模型	207
9.1.1 线程	207
9.1.2 线程与进程的关系	208
9.1.3 多线程并发编程	208
9.2 基本线程的编写	209
9.2.1 主线程	211
9.2.2 获得新线程的两种方式	213
9.2.3 选择合适的方法	214
9.2.4 Thread 类的相关方法	214
9.2.5 Runnable 接口简述	216
9.2.6 线程的生命周期	218
9.2.7 多线程及其优先级	220
9.3 线程同步	223
9.3.1 使用 synchronized 同步线程	223
9.3.2 使用 ReentrantLock 和 ReentrantReadWriteLock 类同步线程	227
9.4 线程组	227
本章小结	228
习题	229
第 10 章 网络通信	230
10.1 概述	230
10.2 使用 URL 访问网络资源	231
10.2.1 URL 类	232
10.2.2 URLConnection 类	234
10.3 Socket 通信	236
10.3.1 Socket 通信原理	237
10.3.2 Java 的 Socket 通信实现	239
本章小结	245
习题	245
第 11 章 数据库编程	246
11.1 JDBC 技术简介	246
11.2 JDBC 的结构	247
11.2.1 JDBC API	247

11.2.2 JDBC Driver API.....	248
11.3 JDBC 驱动程序的类型.....	248
11.4 JDBC 在数据库访问中的应用.....	249
11.4.1 建立与数据库的连接	249
11.4.2 执行查询语句	251
11.4.3 更新数据库操作	255
11.4.4 事务	256
11.5 JDBC 综合应用示例.....	256
11.5.1 建立数据库	256
11.5.2 建立数据源	257
11.5.3 数据库操作程序	260
本章小结	264
习题	264
参考文献.....	266

第1章 绪 论

Java 语言是面向对象的程序设计语言，集安全性、简单性、易用性和跨平台性于一身，不仅可以解决传统的程序设计问题，更主要的是它与互联网密切相关，特别适合于网络环境下编程使用。

1.1 Java 发展简史

Java 是 Sun 公司的产品，而 Sun 公司一直是以经营工作站而闻名的。在相当长的时期中，PC 越来越强的功能对 Sun 工作站构成了很大压力，于是 Sun 公司企图通过开拓电子消费类产品市场减轻这种压力。为此，1991 年，Sun Microsystems 公司的 James Gosling 等开发者开发了一个称为“Green”的项目。这个项目的目的是开发一个分布式系统架构，并使其能在像交互式电视、烤面包箱等消费类电子产品的运行平台上执行。因为这些设备没有很强的处理能力和大容量的内存，所以这个语言必须非常小并且能够生成非常紧凑的代码。另外，由于不同厂商可能选择不同的 CPU，因此这个语言不能够限定在单一的体系结构下。当时，不管是 C 和 C++ 这样的主流语言，还是其他的绝大多数语言都只能对特定目标进行编译。为一种类型的 CPU 芯片编译 C++ 程序就需要一个以该 CPU 为目标的 C++ 编译器。另外，C++ 中的指针操作功能尽管很强大，但操作不慎就会引起问题，使程序出现错误。因此，Green 小组的开发者将 C++ 语言进行简化，去掉指针操作、运算符重载、多重继承等复杂机制，开发出一种新的语言。这是一种解释执行的语言，在每个芯片上装上一个该语言的虚拟机即可运行。Green 小组成员根据公司楼外的一棵橡树而将这种语言命名为“Oak”，这就是 Java 语言的前身。后来发现 Oak 已是另一个公司的注册商标，才改名为 Java (Java 是太平洋上一个盛产咖啡的岛屿的名字)。

到了 1994 年，随着 Internet 的迅猛发展，万维网的应用日益广泛，Gosling 决定用 Java 开发一个实时性较高、可靠安全、有交互功能的新的 Web 浏览器，它不依赖于任何硬件平台和软件平台。新的浏览器称为 HotJava，于 1995 年 5 月 23 日发表后立即在业界引起了巨大的轰动，Java 的地位也随之而得到肯定，这一天也就被 IT 界视为 Java 的生日。一些著名的计算机公司纷纷购买了 Java 语言的使用权，如 Microsoft、IBM、Netscape、Novell、Apple、DEC、SGI 等，因此，Java 语言被美国的著名杂志《PC Magazine》评为 1995 年十大优秀科技产品之一（计算机类仅此一项入选）。随后大量出现了用 Java 编写的软件产品，并受到了工业界的重视与好评，认为“Java 是 20 世纪 80 年代以来计算机界的一件大事”。

微软总裁比尔·盖茨在悄悄地观察了一段时间后，不无感慨地说：“Java 是长时间以来

最卓越的程序设计语言”，并确定微软整个软件开发的战略从 PC 单机时代向着以网络为中心的计算机时代转移，而购买 Java 则是他的重大战略决策的实施部署。因此，Java 的诞生对整个计算机产业产生了深远的影响，对传统的计算模型提出了新的挑战。现今，Java 语言已广泛应用于企业、个人终端、移动通信等众多领域。

1.2 Java 的特点

Java 是一种简单、面向对象、不依赖于机器的结构，具有可移植性、鲁棒性、安全性，提供了并发的机制，并且具有很高的性能；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 和环境的限制；另外，Java 还提供了丰富的类库，使程序员可以很方便地建立自己的系统。Java 语言最主要的特点就是“Write once, run anywhere”，这句话一直是 Java 程序员的精神指南，也是 Java 语言深得程序员喜爱的原因之一。

1. 简单性

Java 语言通过提供最基本的方法来完成指定的任务。程序员只需要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 摒弃了 C++ 中容易引发程序错误的机制(如指针和内存管理)，略去了运算符重载、多重继承等模糊而且很少用到的概念，并且通过实现自动垃圾回收机制，大大简化了程序员的内存管理工作。另外，Java 也适合于在小型机上运行，它的基本解释器及类的支持只有 40 KB 左右，加上标准类库和线程的支持也只有 215 KB 左右。

2. 面向对象

面向对象是当前软件开发的先进技术和重要方法。面向对象的方法基于信息隐藏和数据抽象类型的概念，利用类和对象的机制将数据和方法封装在一起，通过统一的接口与外界交互，通过类的继承机制实现代码重用。Java 语言只支持单继承，但它却支持多接口。Java 语言还支持方法重载和动态调用。总之，Java 语言支持面向对象方法中的三个基本特性：封装性、继承性和多态性。面向对象方法反映了客观世界中现实的实体在程序中的独立性和继承性，有利于提高程序的可维护性和可重用性，还有利于提高软件的开发效率和程序的可管理性。与面向对象的 C++ 语言相比，C++ 为了兼容 C 语言，不得不保留一些面向过程的成分，而 Java 语言去掉了其中非面向对象的成分，是一个完全面向对象的程序设计语言。

3. 体系结构中立、可移植

与其他语言相比，用 Java 语言编写的程序可移植性比较高。Java 语言为了保证可移植性采用了下述机制：

(1) Java 语言规定同一种数据类型在各种不同的实现中，必须占据相同的内存空间。例如，short 型为 16 位，int 型为 32 位，long 型为 64 位，它们与硬件平台无关。由于 Java 语言在数据类型的空间大小方面采用了统一标准，因此保证了其程序的平台独立性。

(2) Java 程序的最终实现需要经过编译和解释两个步骤。Java 语言的编译器生成的可执行代码称为字节码，该字节码可以在提供 Java 虚拟机(JVM, Java Virtual Machine)的任何一

个系统上解释运行,与任何硬件平台无关。由于 Sun 公司规定的 JVM 规范没有涉及任何硬件平台,因此只要根据 JVM 规范创建的平台便可以实现 Java 程序。JVM 是 Java 与平台无关的关键,在 JVM 上有一个 Java 解释器用来解释 Java 编译器编译后的字节码。Java 程序员在编写完软件后,可通过 Java 编译器将 Java 源程序编译为 JVM 的字节码。任何一台计算机只要安装了 JVM,就可以运行这个程序,而不管这种字节码是在什么平台上生成的(如图 1-1 所示)。正因为如此,Java 程序才具有“一次编写,到处运行”的特点。Java 语言采用的这种先编译后解释的方法是以牺牲执行速度来换取与平台无关的,从而提高了可移植性。

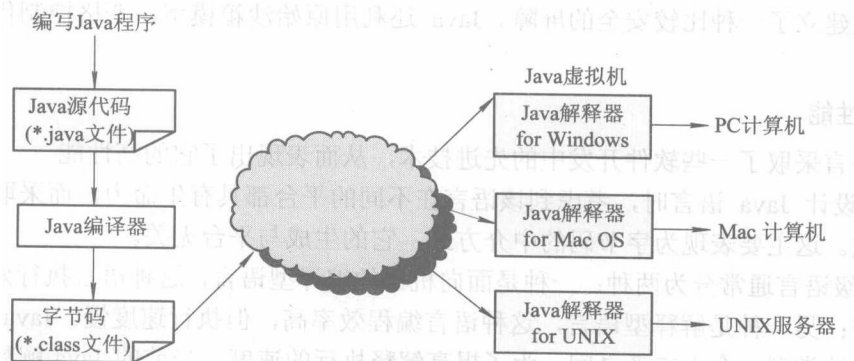


图 1-1 Java 程序的编译和解释

4. 鲁棒性

鲁棒性反映了程序的可靠性。Java 几个内置的特性使程序的可靠性得到改进:

- (1) Java 是强类型语言。编译器和类载入器保证所有方法调用的正确性,防止隐式类版本的兼容性。
- (2) Java 没有指针,不可能引用内存指针搞乱内存或数组越界访问。
- (3) Java 进行自动内存回收,程序员无法意外释放内存,不需要判断应该在何处释放内存。
- (4) Java 在编译和运行时,都要对可能出现的问题进行检查,以消除错误的产生。另外,在编译的时候它还可揭示出可能出现但尚未被处理的异常,以防止系统的崩溃。

5. 安全性

作为 Web 编程语言,Java 具有强大的安全结构和安全策略。代码在编译和实际运行过程中都会接受一层层的安全检查,这样可以防止恶意程序或病毒的入侵。Java 安全性最重要的保证来自字节码检验器。在一个运行平台上运行的字节码是由编译器编译得到的,这类字节码通常符合安全性规定,但是,也有一些较大型的软件从其他地方引入代码,另外,来自网络的病毒也可能会生成字节码,这两类代码都不是编译器产生的,所以,并不能保证是否符合安全性规定。为此,Java 运行系统中设置了一个字节码检验器,在字节码进行解释执行前,字节码检验器先通过一个规则验证程序,对每个代码段作安全检测,这样,通过在字节码装载过程中使用检验器,就确保了指令中参数类型的正确性、对象域访问的合理性以及操作数的边界检查(例如数组边界的自动检查)。

另外,内存布局由 JVM 决定,并依赖于 Java 运行时系统和 JVM 所在宿主机平台的特

性实现内存管理的自动化。一方面,Java 不允许程序员强制性地使用内存指针来访问内存,也就是说,在 Java 程序中,内存分配是透明的,程序员没有对内存分配的决定权,这种将内存分配权交给底层具体运行系统的机制减少了许多内存出错的可能;另一方面,在 Java 程序中,编译器也没有对内存布局的决定权,编译后的字节码中保留了表示内存的符号引用信息,但不使用具体的数字来指定内存的位置,也就是说,Java 把内存分配权交给运行平台,因此,最终的内存布局是由具体运行系统所在软硬件平台决定的;此外,类装载器还为安全性提供了一定的措施,类装载器在装载字节码时,将本地的类组成一个统一的空间,而将外来的类组成另一个空间,这样,可以使后者不能以任何方式对前者进行操作,即为本地类建立了一种比较安全的屏障。Java 还利用原始沙箱模型,严格控制代码的访问权限。

6. 高性能

Java 语言采取了一些软件开发中的先进技术,从而表现出了它的高性能。

(1) 在设计 Java 语言时,考虑到该语言在不同的平台都具有生命力,而采取了一种中性结构方式。这主要表现为字节码的中介方式,它的生成与平台无关。

(2) 高级语言通常分为两种:一种是面向机器的编译型语言,这种语言执行效率高,但可移植性差;另一种是解释型语言,这种语言编程效率高,但执行速度慢。Java 语言是先编译后解释的类型,介于二者之间。为了提高解释执行的速度,当前的 Java 解释器普遍采用了即时编译技术,该技术把字节码转换成对应于特定 CPU 的机器码并缓存,当再次运行该段字节码时,可直接采用已经缓存的机器码执行,避免了每次执行时都要解释执行,从而大大提高了程序的执行效率。

(3) 操作系统两个重要的概念是进程和线程。进程(process)的特点是在执行过程中拥有自己独立的内存空间和系统资源,不同进程的内存数据和状态是彼此独立的。线程(thread)在执行过程中共享一块内存空间和一组系统资源,线程之间可以直接进行数据交换。因此,线程比进程的开销要小。多个线程并行执行,类似于多个 CPU 在同时运行。例如,有两个线程同时工作,一个线程在执行某种复杂的运算,而另一个线程可与用户实现交互。可见,多线程可提高程序的运行效率。Java 语言真正支持多线程。它通过同步关键字(synchronized)来保证多个线程不会同时访问一个被说明为同步的方法,从而使得某些关键操作不被打断。

7. 动态性

Java 的动态性是其面向对象程序设计的延伸。这种特性使得 Java 程序能够适应不断变化的执行环境。类库的使用是最明显的一个例子。用 C++ 编写的应用程序经常会用到各种类库,不仅仅是基础类库,很多时候还需要一些从第三方厂商处购买的类库。销售应用程序时,类库有时是单独出售的。这样就会导致一个问题:类库一旦升级,用这些类库编写的应用程序就必须重新编译,并且重新发送到用户手中,否则就无法利用升级后类库的新增功能。

Java 的“滞后联编”避免了这个问题。“滞后联编”机制使得 Java 完全利用了面向对象编程模式的优点。Java 程序的基本组成单元为类,这些类是在运行过程中动态装载的。因此,Java 可以在分布式环境中动态地维护应用程序及其支持类库之间的一致性。这样,对于 Java 而言,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到原来使用该类

的程序的运行。其支持的类库升级之后,相应的应用程序不必重新编译,也一样可以利用升级后类库的新增功能。

除此之外,Java 的动态性还体现在其对动态数据类型和动态协议的支持上。利用一种特殊的 Applet,即内容句柄,程序员可以很方便地使 HotJava 支持新的数据类型。类似地,通过编写协议句柄,可以使 HotJava 支持新的、自定义的传输协议。

1.3 Java 开发环境

要进行 Java 的开发,必须首先建立起 Java 的运行环境。有了 Java 运行环境,就可以利用任何文本编辑工具编写 Java 源程序,再使用 Java 编译程序对源程序进行编译,之后就可以用解释程序来运行了。

1998 年 12 月, Sun 公司发布了 Java 2 平台和 JDK 1.2,这是 Java 发展史上的一个里程碑。1999 年 6 月, Sun 公司重新组织了 Java 平台的集成方法,并将企业级 Java 应用平台作为发展方向。如今,Java 家族已经有三个主要成员:

- (1) J2SE(Java 2 Standard Edition)是用于工作站、PC 机的 Java 标准平台;
- (2) J2EE(Java 2 Enterprise Edition)是可扩展的企业级 Java 应用平台;
- (3) J2ME(Java 2 Micro Edition)是嵌入式电子设备的 Java 应用平台。

利用 Java 可以开发 Java 小程序(Java Applet)、Java 应用程序(Java Application)、服务器端小程序(Servlet)和 JSP 程序(Java Server Page)。Applet 是嵌入在 HTML 文件中的 Java 程序,相当于嵌入在页面之中的脚本。Applet 的大小和复杂性是没有限制的,但由于 Internet 网速的限制,通常 Applet 会很小。对于 Java 开发工具(JDK)而言,应用程序可以理解为从命令行运行的程序。Java 应用程序在最简单的环境中,它的惟一外部输入就是在启动应用程序时所使用的命令行参数。Servlet 和 JSP 都主要工作在服务器端,为 HTTP 服务提供动态的处理。所不同的是,Servlet 是 Java 程序,而 JSP 是在 HTML 文件里嵌入了 Java 代码。

1.3.1 JDK 的安装与设置

1. JDK 简介

JDK(Java Development Kit)即 Java 软件开发工具包,与 J2SDK(Java 2 Software Development Kit)的含义通常是一样的,是 Java 的开发环境。由于人们对早期的版本简称为 JDK,因此现在往往还将 J2SDK 简称为 JDK。一般初学 Java 时都选用 JDK 作为开发环境,而其他的集成开发环境都是在 JDK 的基础上建立的,也就是说,如果没有 JDK,其他集成开发环境是无法工作的。Sun 公司的 J2SDK 是免费的工具,可以到 Sun 公司网站或提供相关下载的网站下载。不同的版本适合不同的操作系统,用户可以根据自己所用的操作系统下载相应的 J2SDK 版本。本书以 Windows XP 操作系统为例进行运行环境的搭建,所使用的是 JDK 1.5。

2. JDK 的安装

首先要下载 JDK 开发工具,用户可以从 <http://java.sun.com> 下载最新的 JDK 开发工具。目