

高职高专计算机教育规划教材

Java 程序设计

刘海军 主编 陈素羨 李英杰 副主编 杨学全 主审



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

高职高专计算机教育规划教材

Java 程序设计

	刘海军	主 编
陈素美	李英杰	副主编
程治国	张红强	参 编
	杨学全	主 审

内 容 简 介

Java 作为一种纯面向对象的程序语言以其安全性、健壮性与平台无关性等特点,成为当前最流行的网络编程语言。本书将 Java 语言的基本特性与面向对象的编程思想有机结合,侧重程序设计的基础知识、基本语法与实例相结合,培养读者面向对象编程思想的思维方式和解决实际问题的能力。

本书共分为 12 章,主要内容包括 Java 的基本语法、面向对象思想、GUI 界面设计、输入/输出、数据库连接、多线程编程、网络程序开发等。

全书内容丰富、讲解深入浅出、系统性和实例性较强,是高职高专学生的首选教材,本书也可作为 Java 程序开发人员学习和应用的参考书。

图书在版编目(CIP)数据

Java 程序设计/刘海军主编. —北京:中国铁道出版社, 2006. 8

高职高专计算机教育规划教材

ISBN 7-113-07039-6

I. J... II. 刘... 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(2006)第 100640 号

书 名: Java 程序设计

作 者: 刘海军 等

出版发行: 中国铁道出版社(100054, 北京市宣武区右安门西街 8 号)

策划编辑: 严晓舟 秦绪好

责任编辑: 苏 茜 翟玉峰

特邀编辑: 贺 军

封面设计: 付 巍

封面制作: 白 雪

责任校对: 黄园园

印 刷: 河北省遵化市胶印厂

开 本: 787×1092 1/16 印张: 17.25 字数: 400 千

版 本: 2006 年 8 月第 1 版 2006 年 8 月第 1 次印刷

书 号: ISBN 7-113-07039-6/TP·1791

定 价: 23.00 元

版权所有 侵权必究

本书封面贴有中国铁道出版社激光防伪标签,无标签者不得销售

凡购买铁道版的图书,如有缺页、倒页、脱页者,请与本社计算机图书批销部调换。

高职高专计算机教育规划教材

顾 问：冯博琴

主 任：张晓云

副主任：陈建铎 李伟华 王海春 范启岭

王 津 杨俊清 孟繁增 崔永红

委 员：（按姓氏字母先后为序）

白延丽 董少明 韩文智 韩银锋 黄伟敏

李培金 李秀疆 刘省贤 刘喜勋 梅创社

沈久福 王 可 王 坤 吴晓葵 熊永福

杨卫社 杨学全 张 勇 张 宇 钟生海

我国经济建设和发展取得了举世瞩目的成就,随着经济建设发展的需求,教育事业也得到了发展,特别是我国高职高专教育实现了跨越式发展。依据教育部公布的教育统计年报,2000年全国高等院校共有1813所,到2005年全国普通高等院校和成人高等院校共有2273所,增长25%;2000年普通高等院校共招本科、高职(专科)学生464.21万人,校均规模达5289人,2005年达504.46万人,校均规模达7666人,增长8%;2005年全国各类高等教育总规模超过2300万人,高等教育入学率达到21%。

十六大报告指出,本世纪头20年经济建设和改革的主要任务是:完善社会主义市场经济体制,推动经济结构战略性调整,基本实现工业化,大力推进信息化,加快现代化建设。坚持以信息化带动工业化,以工业化促进信息化,走出一条科技含量高、经济效益好、资源消耗低、环境污染少、人力资源优势得到充分发挥的新型工业化路子。形成以高新技术产业为先导、基础产业和制造业为支撑、服务业全面发展的产业格局。高职高专的专业建设格局也依据该思路不断调整。例如:陕西省2005年高职高专院校中,专业排名前10位的统计中,有44所学校开设了“计算机应用技术”课程,排名第一;有26所学校开设了“计算机网络技术”课程,排名第五;有24所学校开设了“计算机信息管理”课程,排名第六。2005年,这3个专业的毕业生数共计5199人。除了信息产业外,具有信息化知识的复合人才也是信息产业人才需求的另一部分,随着社会信息化的程度日趋提高,需要一大批既懂得计算机技术,又懂得经营管理的信息化人才;随着网络技术的发展,网络管理、网络安全、网页制作等方面的人才需求也将日趋增加。

高校扩招、用人需求扩大、专业建设不断发展壮大,优质教学资源短缺成为高职高专教育的热点问题。作为信息化基础的计算机技术及应用方向的教学体系与课程建设,在教学思想、教学方法、教学手段不断改革的过程中,积累了大量的可推广的经验。一套好的教材是优质教师队伍通过长时间教学实践积累的产物,是教学改革经验与成果的有效推广载体与手段,是教学改革经验与成果推广应用的有效途径,因此,教材建设工作是整个高职高专教育教学工作中的重要组成部分,积极推动教材建设工作是解决优质教学资源短缺、实现优质资源共享的有效方式。中国铁道出版社正是认识到了计算机技术教育发展信息化建设的关联,因而积极推广教学改革经验与成果,协助高职高专院校实现优质资源共享,并为此推出了计算机“高职高专计算机教育规划教材”丛书。

本丛书本着以服务为宗旨，以就业为导向，面向社会、面向市场、面向职业岗位能力，积极围绕职业岗位人才需求的总目标和职业能力需求，根据不同课程在课程体系中的地位及不同作用，采取不同的教学即教材编写方法。如以知识讲授为主体的围绕问题中心的教学和教材编写；以基础能力训练为核心的围绕基础训练任务的教学和教材编写；以岗位综合能力训练为核心的以任务为中心的教学和教材编写等。

国家兴盛，人才为本；人才培养，教育为本。信息化是我国加快实现工业化和现代化的必然选择，高职高专教育应抓住机遇，乘势而上，培养数以千万计的高技术应用专门人才，为经济建设发展提供保障，以求在贯彻和服务于全面落实科学发展观的过程中能更好的发展。

张 晓 云

2005 年 11 月 7 日, 全国职业教育工作会议在北京召开。会上提出大力发展职业教育, 是推进我国工业化、现代化的迫切需要, 是促进社会就业和解决“三农”问题的重要途径, 也是完善现代国民教育体系的必然要求。这次全国职教工作会议的召开必将有力地推动职业教育的快速发展。

高等职业教育是我国高等教育的重要组成部分。近年来, 高等职业教育异军突起, 迅速发展。2004 年, 全国普通高等学校毕业生 280 万人, 其中高等职业教育毕业生 147 万人, 占总数的 52.5%, 本科毕业生占 47.5%。因此, 加强高等职业教育教学改革, 提高人才培养质量就显得格外重要。以课程改革为重点的教学改革, 是培养生产、建设、管理、服务等第一线具备综合职业能力和全面素质的高级实用人才的关键; 以技术应用能力为中心, 以就业为导向构建专业理论教学、实践教学、职业素质拓展课程体系是专业教学改革的重要任务。

《JAVA 程序设计》是一门完全面向对象的程序设计语言, 适用于计算机应用类专业或非计算机专业的程序设计教学, 是软件工程、信息系统开发等课程的前驱课程。本教材结合作者多年的教学实践经验, 以职业技术能力培养为主线, 采用任务驱动模式, 对内容进行了精心地组织和安排。

本书由刘海军主编, 陈素羨、李英杰任副主编, 程治国、张红强、李淑娣任参编, 其中第 1、2 章由刘海军编写, 第 3~7 章由陈素羨编写, 第 8、9 章由李英杰编写, 第 10 章由张红强编写, 第 11 章由李淑娣编写, 第 12 章由程治国编写。全书由刘海军老师统稿, 杨学全老师主审, 陈素羨、李英杰校对。

由于作者水平所限, 书中难免会有疏漏之处, 还望各位同行和读者提出宝贵意见。

编者

2006 年 8 月

目录

CONTENTS

第 1 章 Java 语言入门	1
1.1 Java 的发展史	1
1.1.1 Java 的起源	1
1.1.2 Java 的发展	2
1.1.3 Java 的成熟	2
1.2 Java 的特点	2
1.3 Java 的运行环境	4
1.3.1 Java 虚拟机	4
1.3.2 Java 程序的运行机制	5
1.4 搭建基本的 Java 开发环境	5
1.4.1 J2SDK 的安装	5
1.4.2 设置环境变量	7
1.4.3 J2SDK 开发工具包	8
1.5 Java 程序的开发过程	10
1.5.1 Java 应用程序的开发过程	10
1.5.2 Java 小应用程序的开发过程	12
本章小结	13
习题	14
第 2 章 Java 的基本语法	15
2.1 标识符和关键字	15
2.1.1 标识符	15
2.1.2 关键字	16
2.2 基本数据类型与常量	16
2.2.1 整型与常量	16
2.2.2 浮点型与常量	17
2.2.3 字符型与字符型常量	18
2.2.4 布尔型	18
2.3 变量	18
2.4 运算符与表达式	20
2.4.1 算术运算符和算术表达式	20
2.4.2 关系运算符和关系表达式	21
2.4.3 逻辑运算符和逻辑表达式	22
2.4.4 条件运算符	22
2.4.5 赋值运算符和赋值表达式	23
2.4.6 运算符的优先级及结合性	24
2.5 语句和流程控制	25

2.5.1 语句.....	25
2.5.2 选择结构.....	26
2.5.3 循环结构.....	29
2.5.4 跳转语句.....	31
本章小结.....	32
习题	32
第3章 Java 与面向对象	34
3.1 面向对象程序设计	34
3.1.1 面向对象的引入	34
3.1.2 面向对象的基本特征	35
3.1.3 面向对象的基本概念	36
3.2 类与对象.....	36
3.2.1 类的定义.....	36
3.2.2 对象.....	39
3.3 包.....	41
3.3.1 创建自定义包.....	41
3.3.2 包的引用.....	42
3.3.3 Java 的系统包.....	43
3.3.4 Java 程序的结构.....	44
3.4 类的继承.....	48
3.4.1 创建子类.....	48
3.4.2 super 与 this 的使用.....	49
3.5 接口.....	51
3.5.1 接口的定义.....	51
3.5.2 接口的声明.....	51
3.5.3 接口的实现.....	52
3.6 类的多态.....	53
3.6.1 方法重载.....	53
3.6.2 方法重写.....	54
本章小结.....	55
习题	55
第4章 数组与字符串.....	57
4.1 一维数组的创建与使用	57
4.1.1 一维数组的创建.....	57
4.1.2 一维数组的初始化.....	58
4.1.3 一维数组的基本使用	59
4.1.4 数组的常用方法.....	61
4.2 多维数组.....	63
4.2.1 二维数组的创建.....	63
4.2.2 二维数组的初始化.....	64

4.2.3 二维数组的引用	64
4.3 字符串	65
4.3.1 字符串的创建	66
4.3.2 字符串的常用方法	67
4.3.3 可变字符串 StringBuffer	72
本章小结	73
习题	73
第 5 章 JBuilder 2006 开发环境简介	75
5.1 JBuilder 简介	75
5.2 JBuilder 集成开发环境	76
5.3 从 JDK 过渡到 JBuilder	78
5.3.1 创建工程	78
5.3.2 用 JBuilder 开发 Java 应用程序	80
5.3.3 用 JBuilder 开发 Java 小应用程序	83
5.3.4 移植 Java 程序	85
本章小结	88
习题	88
第 6 章 Java 的图形用户界面	89
6.1 GUI 支持的包简述	89
6.2 容器、组件及布局的关系	91
6.3 AWT 工具集	92
6.3.1 窗口	92
6.3.2 文本框和文本区	93
6.3.3 按钮	96
6.3.4 标签	96
6.3.5 面板 Panel 类	98
6.4 布局管理器	99
6.4.1 FlowLayout 流布局管理器	99
6.4.2 BorderLayout 边框布局管理器	100
6.4.3 GridLayout 网格布局管理器	101
6.4.4 CardLayout 卡片布局管理器	102
6.4.5 null 布局	104
6.5 事件处理机制	105
6.5.1 Java 中的事件类	107
6.5.2 事件处理方法	108
6.5.3 使用内部类实现事件处理	111
6.5.4 使用匿名类实现事件处理	112
6.6 Swing 组件的应用	113
6.6.1 常用容器组件	113
6.6.2 标签	116

6.6.3 文本框.....	116
6.6.4 按钮.....	118
6.6.5 选择框.....	120
6.6.6 列表框与组合框.....	123
6.6.7 滑块.....	126
6.6.8 菜单.....	128
6.6.9 对话框.....	132
本章小结.....	137
习题.....	137
第 7 章 Java 与图形、图像	138
7.1 Java 与图形.....	138
7.1.1 绘制直线.....	138
7.1.2 绘制矩形.....	139
7.1.3 绘制椭圆.....	140
7.1.4 绘制圆弧.....	141
7.1.5 绘制多边形.....	141
7.1.6 填充图形.....	142
7.1.7 绘制文字.....	143
7.1.8 在应用程序中绘制图形.....	147
7.2 Java 与图像.....	148
本章小结.....	150
习题.....	150
第 8 章 Java 的数据库连接	152
8.1 JDBC 概述.....	152
8.1.1 JDBC 的发展.....	152
8.1.2 JDBC 与 ODBC 的比较.....	153
8.1.3 JDBC 的构成.....	154
8.2 JDBC 应用程序接口简介.....	155
8.2.1 JDBC 的驱动程序管理器——DriverManager 类.....	155
8.2.2 JDBC 与数据库的连接——Connection 接口.....	156
8.2.3 执行普通 SQL 语句——Statement 接口.....	156
8.2.4 数据结果集——ResultSet 接口.....	157
8.3 利用 JDBC 访问数据库实例.....	157
8.3.1 通过 JDBC-ODBC 桥连接来访问数据库.....	157
8.3.2 利用本地协议纯 Java 驱动程序连接数据库.....	161
8.4 异常处理机制.....	162
8.4.1 异常的相关概念.....	163
8.4.2 异常处理方法.....	163
本章小结.....	167
习题.....	167

第 9 章 Java 的输入/输出	169
9.1 输入/输出流概述	169
9.2 字节流类	171
9.3 字符流类	175
9.4 标准输入/输出类	178
9.5 其他常用流类	179
9.5.1 缓冲流	179
9.5.2 数据流	180
9.5.3 随机存取文件流	182
9.6 文件操作	183
本章小结	185
习题	185
第 10 章 Java 的多线程编程	187
10.1 多线程概述	187
10.1.1 进程与线程	187
10.1.2 线程的状态与生命周期	188
10.2 线程的创建	189
10.2.1 Thread 类简介	189
10.2.2 线程的实现方法	190
10.3 线程的同步	193
10.4 线程的死锁	194
本章小结	194
习题	195
第 11 章 Java 的网络编程	196
11.1 网络通信基本原理	196
11.2 TCP 与套接字编程	198
11.2.1 套接字的概念	198
11.2.2 Socket 类	198
11.2.3 ServerSocket 类	199
11.2.4 InetAddress 类	201
11.3 UDP 编程	202
本章小结	208
习题	208
第 12 章 学生成绩管理系统综合实训	210
12.1 系统分析	210
12.2 系统设计	210
12.3 系统实现	214
参考答案	261
参考文献	263

第 1 章 \ Java 语言入门

建议学时: 4 学时

总体要求:

- 了解 Java 的发展史
- 熟练掌握 Java 开发环境的使用
- 掌握两种 Java 应用程序的开发过程

核心技能点:

- Java 开发环境的使用
- Java 两种应用程序的编写、编译和运行

扩展技能点:

- 了解 Java 运行环境

相关知识:

- Java 的发展史
- Java 的特点

学习重点:

- 熟练掌握 Java 开发环境的配置
- 掌握两种 Java 应用程序的开发过程

Java 是面向对象的编程语言,也是当今流行的网络编程语言。Java 的出现引起了软件开发的变革,它为迅速发展的 IT 业增添了新的活力。近几年来, Sun 公司推出了 J2ME、J2SE、J2EE 三大平台,显现了 Java 语言强大的应用能力。

本章主要介绍 Java 的发展史,Java 的基本特点及用途,Java 程序开发环境的设置方法,Java 程序的运行方法,为后续各章的学习做好准备。

1.1 Java 的发展史

1.1.1 Java 的起源

Java 的历史可以追溯到 1991 年 Sun 公司的 James Gosling 等开发者开发的一个称为“Green”的项目。这个项目的目的是为家用消费性电子产品(如有线电视转换盒)开发一个分布式代码系统。由于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,所以这个语言不能够限定在单一的体系结构下。最初,它的开发者想用 C++ 语言来开发,但是,由于芯片的种类各不相同,程序需要进行多次编译。另外,C++ 中的指针操作功能很强大,一旦操作不慎就会引起问题,使程序出现错误。因此,它的开发者将 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。它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。开始,这种新的语言被称为“Oak”。后来,发现“Oak”已经是一门存在的计算机语言,于是改名为“Java”。

1.1.2 Java 的发展

体系结构中立的需要是 Java 诞生的源动力,而 Java 的成功更要归功于 Internet 的发展。随着 Internet 和 Web 的出现,关于可移植语言的老问题又被提了出来。Internet 由不同的、分布的系统组成,其中包括各种类型的计算机、操作系统。尽管许多类型的平台都可以与 Internet 连接,但用户希望它们能够运行同样的程序。后来人们想到了 Java 语言,它是解释型执行语言,只要在每个机器上装上它的“虚拟机”,Java 语言就可以运行。于是 Java 语言得到了快速的发展。目前,Java 语言已经是网络上最流行的开发语言之一。

1.1.3 Java 的成熟

Java 语言正在逐渐成熟起来,它的类已经近千个。Java 中的类无所不含,而且还可以通过第三方购买,这为 Java 语言提供了良好的发展前景。另外,它还是跨平台的语言。因此,许多软、硬件开发商争先恐后地想乘上 Java 语言的快车,在他们的产品中都声称支持 Java 语言,而且 Sun 公司也正努力开发 Java 芯片,这一切都极大地促进了 Java 语言的发展。

1.2 Java 的特点

Java 之所以发展得如此迅速,源于它的一系列优点和特性。Java 设计者将 Java 设计成了简单、可移植性、面向对象、解释型、分布式、高性能、健壮性、多线程、安全、动态及体系结构中立的一种计算机程序设计语言。

1. 简单

Java 语言易学易用。Java 语言的风格类似于 C++语言,但要比 C++简单。它舍弃了 C++中一些不常使用、难于理解、易于混淆的特性,如指针操作、多继承等,并增加了自动垃圾回收功能,用于回收不再使用的内存区域。这不但有利于程序的编写,而且大大减少了由于内存分配而引发的各种问题。

另外,简单还体现在小巧上。Java 的解释器、系统类库和运行系统都比较小巧,既能够在小型机上运行,也可以从网络上下载到 PC 上运行。

2. 面向对象

面向对象技术较好地解决了当前软件开发过程中各种面向过程语言不能处理的问题。这些问题包括开发规模的扩大、升级加快、维护难度加大等。面向对象编程按照人们的思维方式建立问题空间模型,利用类和对象的机制将数据及其操作封装在一起,通过统一的接口与外界交互。用 Java 语言编写的任何程序都具有面向对象的特性。

3. 分布式

Java 是网络编程语言,它的网络能力强大而且易于使用。其类库提供用于 HTTP 和 FTP 传送信息的方法。Java 程序能够通过 URL 很方便地打开和访问网络上的资源,其便利程度如同访问本地文件一样。另外,Java 的 Applet 小应用程序可以从服务器下载到客户端,部分操作可以在客户端进行,从而提高了系统的执行效率。

4. 解释型和高性能

Java 是解释型的语言,它编译后并不生成特定的 CPU 机器代码,而是生成一种被称为

“字节码”的目标代码。字节码可以在提供“Java 虚拟机 (Java Virtual Machine, JVM)”的任何一种系统上解释执行。

众所周知,解释型语言的执行速度比编译型语言要慢。为了解决这个问题,在许多平台上,有另一种编译方式——即时编译 (JIT)。它可以一次把字节码编译为本地代码,缓存得到的结果,在需要的时候被重新调用,从而使 Java 的运行速度接近于普通的编译型语言的运行速度。

5. 健壮性

Java 在编译和运行程序时,都要对可能出现的问题进行检查,以消除错误的产生。Java 的内存管理机制减少了内存出错的可能性。它提供的自动垃圾收集机制防止了编程人员在管理内存时容易产生的错误。Java 提供了异常处理机制,在编译时 Java 将提示可能出现但未被处理的异常,帮助编程人员正确地进行判断,以防止系统崩溃。另外,例行检查可以帮助编程人员检查出许多开发早期出现的问题。

6. 安全

由于 Java 语言主要用于网络应用程序开发,所以对安全性有很高的要求。如果没有安全保证,用户运行从网络上下载的 Java 程序将十分危险。Java 通过自己的安全机制防止了病毒程序的产生和下载程序对本地系统的威胁破坏。当 Java 字节码进入解释器时,必须经过字节码校验器的检查;然后,Java 解释器将决定程序中类的内存布局;最后,类装载器负责把来自网络的类装载到单独的内存区域,避免应用程序之间相互干扰破坏。除此之外,客户端用户还可以限制从网络上装载的类访问某些文件系统。Java 安全机制禁止 Java 程序进行的一些操作:如禁止运行时堆栈溢出、禁止在自己的处理空间外破坏内存、禁止通过安全控制类装载器来读写本地文件,就像 Web 浏览器禁止此类操作一样。所有的这些特点都表明 Java 的确是当前最安全的编程语言之一。

7. 体系结构中立

Java 程序设计者的目标是“一次编写,处处运行”。Java 编译器通过生成与特定计算机体系结构无关的字节码指令来实现这个目标。而且,字节码文件既可以容易地在当今大多数通用计算机体系结构上解释执行,也能够运行时很容易地被编译成本地机器代码。

8. 可移植性

结构中立构成了程序可移植性的基础。Java 语言的基本数据类型长度采用了固定长度,消除了代码移植时让人头疼的问题。另外,Java 的类库提供了可移植性的接口,如类库中有一个抽象类 Windows,并给出了在 UNIX、Windows 和 Macintosh 等系统上的不同实现方法。

9. 多线程

设计 Java 的目标之一是为了满足用户对创建交互网络程序的需要。为此,Java 支持多线程编程。用 Java 语言编写的多线程应用程序可以同时运行多个程序段,从而有更好的交互性和实时性。编写一个多线程程序要比编写一个单线程程序困难得多。而 Java 提供了一套复杂地线程同步化机制,使得编程人员可以很方便地使用这一机制编写出健壮的多线程程序。

10. 动态

Java 的动态性使得它能够适应环境发展的需求。其类库可以随意地增加新的方法及实例变量,而客户端可以不做任何更改。另外,在 Java 中确定实时类型信息是简单直接的。

1.3 Java 的运行环境

Java 程序运行于 Java 平台之上，Java 平台运行于 Windows、Linux、Solaris 等操作系统之上。Java 平台由 Java 虚拟机（JVM）和 Java 应用程序接口（Application Programming Interface, API）组成。Java 虚拟机屏蔽了不同操作系统之间的差异，Java API 为编程人员提供统一的编程接口。Java 平台把 Java 程序从硬件依赖中分离出来，从而实现了 Java 的“一次编写，处处运行”的特性。

Java 的运行环境如图 1-1 所示。这样的执行架构，大致可以分成以下 4 层：

- 第 1 层：Java 字节码 (*.class 文件)
- 第 2 层：Java 平台（包括 Java 核心 API 和 Java 虚拟机）
- 第 3 层：操作系统
- 第 4 层：硬件平台（计算机硬件、嵌入式硬件系统等）

硬件平台执行上层的操作系统，操作系统再执行上一层的 Java 虚拟机，Java 虚拟机解释并执行 Java 字节码文件。

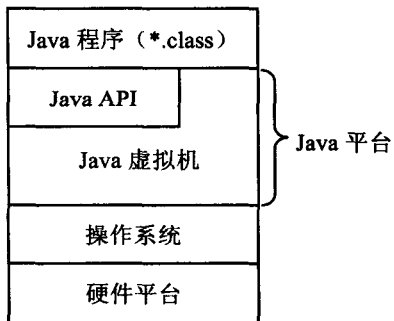


图 1-1 Java 平台的组成

1.3.1 Java 虚拟机

Java 平台由 Java 虚拟机和 Java 应用程序接口组成。

所谓 Java 虚拟机，实际上也是运行在操作系统上的一种程序，它可以解释 Java 的字节码文件，并调用操作系统的系统函数执行 Java 字节码。其原理如图 1-2 所示。

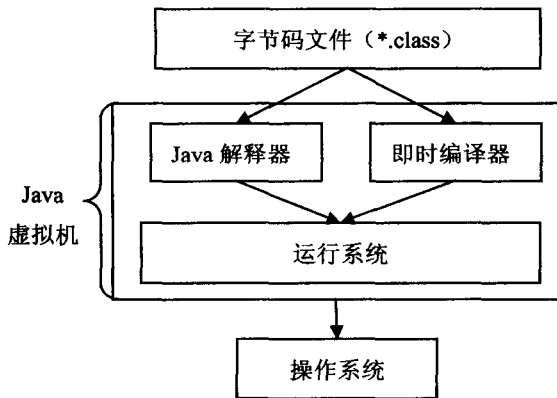


图 1-2 Java 虚拟机

Java 源程序首先由编译器编译成 Java 语言特有的字节代码，然后由 Java 虚拟机去解释执行。Java 字节码是一种二进制文件，但不能直接在操作系统上运行，它可看作是虚拟机的机器码。

Java 虚拟机是一种利用软件方法实现的抽象化的计算机，它把字节码文件与各操作系统和硬件平台分开，使 Java 程序独立于平台。

1.3.2 Java 程序的运行机制

Java 源程序通过编译器将源文件 (*.java) 编译成 Java 字节码文件。然后, Java 字节码文件通过 Java 虚拟机中的 Java 解释器边解释边执行。另外, 它也可以通过 Java 虚拟机中的 Java 即时编译器编译, 只需编译一次便可执行。

Java 程序的运行机制如图 1-3 所示。

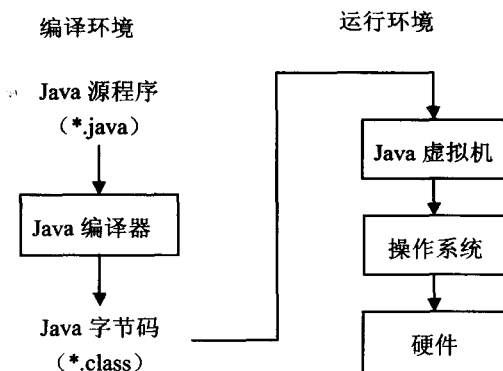


图 1-3 Java 程序的运行机制

1.4 搭建基本的 Java 开发环境

构建基本的 Java 开发环境是开发 Java 程序的前提和基础。

Sun 公司提供了一个免费的 Java 软件开发工具包 JDK (Java Development Kit), 该工具包包含了编译、运行及调试 Java 程序所需要的工具, 此外还提供了大量的基础类库供编写程序使用, 它是开发 Java 程序的基础。Sun 公司将 JDK1.2 以后版本通称为 Java 2。如后来推出的 1.3、1.4 及 1.5 等版本都属于 Java 2 范畴。现在 JDK 通常又称为 J2SDK (Java 2 Software Development Kit)。

Sun 公司为不同的操作系统平台, 如 Windows、UNIX、Linux 等提供了相应的 Java 开发包的安装程序。用户可到 Sun 公司站点 <http://java.sun.com> 下载最新的适应于相应操作系统的开发包。由于 J2SDK 的更新速度很快, 通常每三四个月就会有新版本推出, 每推出一个新版本都会对前一个版本有一些改进, 同时也减少了 Bug 的存在。因此, 安装最新版本的 J2SDK 是很有必要的。本书中使用 Windows 操作系统环境下的 Java 开发包。书中的实例程序均在 1.5.0.02 版本下运行通过, 所使用的操作系统为 Windows 2000 专业版。

本节主要介绍如何安装 J2SDK、创建 Java 标准平台及建立 Java 平台所需的环境。所有的 Java 程序是在 Java 运行环境中运行的, Java 运行环境是建立在 Java 平台之上的, 它同 Java 平台一同创建。

1.4.1 J2SDK 的安装

(1) 双击开发包安装程序, 出现如图 1-4 所示的开始安装界面。随后, 安装程序弹出