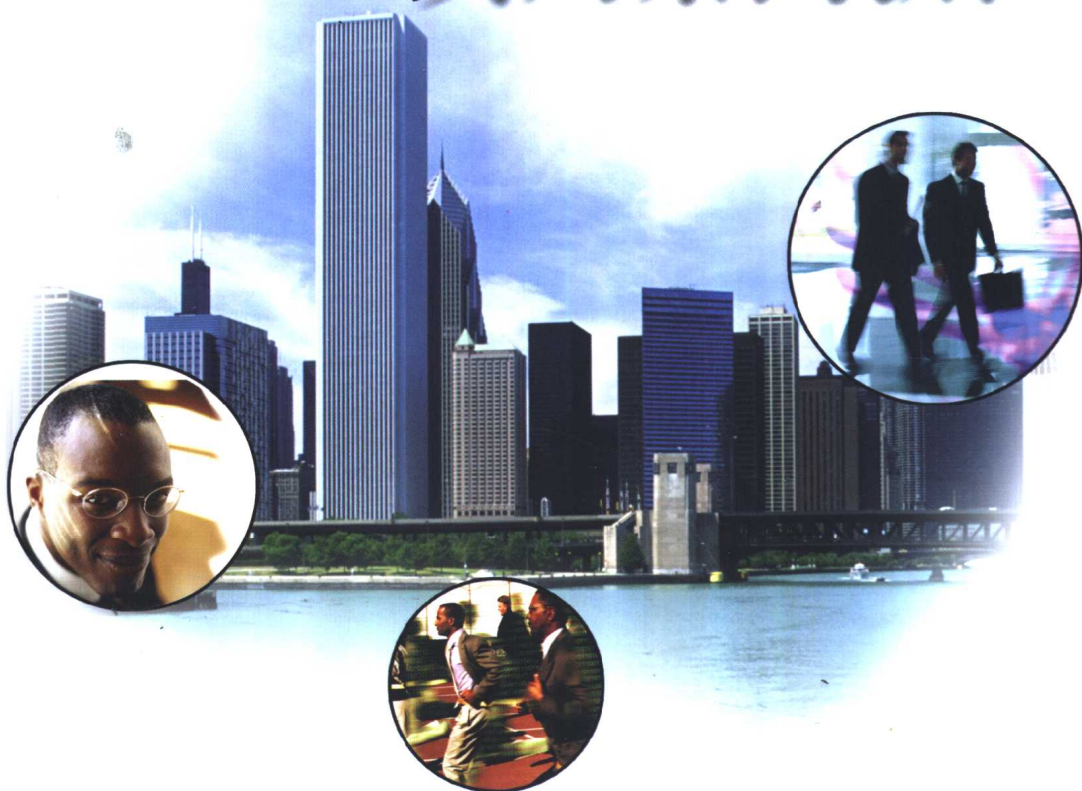


IT 培训 标准 系列 教材

软件程序员系列

# Java

## 实用案例教程



新东方IT教育 教材研发室  
张利国 编著



清华大学出版社  
<http://www.tup.tsinghua.edu.cn>



IT 培训标准系列教材 软件程序员系列

# Java 实用案例教程

新东方 IT 教育 教材研发室

张利国 编著

清华大学出版社

北 京

## 内 容 简 介

本书系统地介绍了 Java 语言基础知识、面向对象编程(OOP)和一些重要的高级特性,如内部类、事件处理机制、多线程、JDBC 等,并附有一套具有代表性的 Sun 认证 Java 程序员(SCJP)考试模拟试题及深入分析。本书结构完整、论述生动、深入浅出;围绕各知识点提供了丰富的例程和综合实用案例,清楚地展示了基于 Java 技术进行实用性应用软件开发的全过程;适合作为高校及社会培训机构教材,以及各类 Java 初学者的自学参考书。本书全部源代码可在新东方网站 [www.getjob.com.cn](http://www.getjob.com.cn) 上下载。

版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

### 图书在版编目(CIP)数据

Java 实用案例教程/张利国编著. —北京:清华大学出版社,2003.10

(IT 培训标准系列教材. 软件程序员系列)

ISBN 7-302-07050-4

I. J... II. 张... III. JAVA 语言—程序设计—技术培训—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2003)第 070081 号

出 版 者: 清华大学出版社

<http://www.tup.com.cn>

社总机: 010 - 62770175

地 址: 北京清华大学学研大厦

邮 编: 100084

客户服务: 010 - 62776969

组稿编辑: 郑寅堃

文稿编辑: 林庆嘉

封面设计: 丛 琳

印 刷 者: 北京鑫丰华彩印有限公司

发 行 者: 新华书店总店北京发行所 \ 清华大学出版社出版发行

开 本: 185 × 260 印张: 20 字数: 481 千字

版 次: 2003 年 10 月第 1 版 2003 年 10 月第 1 次印刷

书 号: ISBN 7-302-07050-4/TP·5182

印 数: 1 ~ 3000

定 价: 34.00 元

# 丛书特点

本丛书是为人们掌握 IT 实用技能,适应 IT 实际工作的需要出版的。丛书贯彻以下宗旨:

## 1. IT 职业技能与认证兼顾

- 面向 IT 职位需求选材,重点在四大流行 IT 职位上,包括电脑美术设计师、网络工程师、软件程序员、数据库管理员。为了让读者更加容易了解 IT 职位的需求,搞清楚每个职位所需具备的实用技能,丛书按照职位划分系列,每个系列面向一个职位,每一本图书针对职位内的一种 IT 技能,独立成册。
- 大部分图书都是由新东方 IT 教育资深任课教师编写,符合教师授课需要和学员学习要求;与新东方面向职位的就业课程密切结合,读者也可以作为自学教材和认证培训的教材。

## 2. 合理的知识结构顺序与阶梯式案例教学完美结合

- 按照学习规律安排全书写作顺序,从而让读者易于把握住技术的整体框架和学习的脉络。
- 采用新东方阶梯式案例教学法组织案例写作。丛书的例子分为“示例”和“案例”两种类型,通过“示例”理解技术点,从而掌握软件技术;通过“案例”掌握 IT 实用技能,达到胜任 IT 职位要求的目的。通过阶梯式案例的写法,可使读者实现三个方面能力的转变:
  - 技术深度上的转变,从技术入门到职业技能的提高;
  - 专业化的转变,由非专业到专业;
  - 能力上的转变,真正掌握 IT 技能的学习方法。

## 3. IT 职位的技能需求更新与软件版本更新

软件版本会更新,而软件的应用变化则相对缓慢,IT 职位的需求很难发生根本性变化,当然,软件版本更新也会促进应用和职位技能的新需求。丛书立足于 IT 职位,让图书的软件版本跟着 IT 职位需要走,综合考虑 IT 职位需求与软件版本这两方面因素来更新图书的版本。

最后,希望本丛书能够让广大读者从中获益。

# 总 序

新东方教育集团董事长 俞敏洪

当今社会是一个需要技能的社会,新东方最初的主要方向就是培训学生英语方面的技能,因为良好的英语技能对于工作、前途都起着非常重要的作用。在过去的几年里,新东方的英语培训取得了长足的发展和辉煌的成就,每年培训几十万学生,并且在全世界建立了十几所分校。大量的新东方学员经过新东方培训后取得了出国深造的资格。但是在世界上除了英语技能,还有许多其他的技能,对提高人们的生活水平和质量起着更加重要的作用,电脑技能就是其中之一。在现代化的技术世界中,电脑已经成为人们生活的一个重要内容。有知识的人离不开英语,但是更离不开电脑,电脑已经成为人们日常学习和工作必不可少的一部分。不可否认,电脑的发展使这个世界更加简单化、条理化、明了化。

新东方一直把为人们谋取更好的工作、更好的生活质量作为目标,所以在英语培训取得了一定的成就以后,新东方一直在思考是否能在电脑方面为学员提供一点帮助,能够像英语一样,经过一定的培训也能让大家获得更好的电脑技能。出于这样的思考,新东方于 1999 年创立了电脑培训部。经过几年的发展,电脑培训部由原来只有 20 多台电脑的小小培训部发展到了现在有数百台电脑的培训中心。在发展过程中,电脑培训部得到了微软、联想、Adobe、Macromedia 等世界著名电脑公司的培训授权。迄今为止,新东方电脑培训已经为社会培训了上万名学员,为众多公司及企事业输送了大量的电脑专业人才,有些人经过新东方电脑培训部的培训,在国外的电脑公司也找到了理想的工作。

提到新东方电脑培训部,首先必须提到的是我的朋友、新东方董事会成员、新东方电脑培训部的创始人王强老师。王强自 1996 年回国创业,力推美语思维教学法,使新东方成千上万的学生在口语方面取得了长足的进步。但王强老师实际上还是一位电脑专家,来新东方以前是美国贝尔传讯研究所的软件工程师,曾在贝尔获得过优秀软件工程师的称号。回国后王强老师尽管转入英语教学领域,但对电脑的热情始终不减,力推电脑培训部的建设,终于在 1999 年使新东方电脑培训部得以成立。

另外一位我要介绍的人是新东方电脑培训部的第一任主任周怀军老师。周怀军在来电脑培训部负责具体业务之前,已经移民加拿大,并在加拿大一家计算机公司找到了一份收入不错的工作。但是他有感于新东方的创业精神,有感于自己想要干一番事业的理念,毅然放弃了在加拿大的职位,放弃了在加拿大和家人团聚的机会,从电脑培训部成立的第一天干起,四年中一直忘我地工作,奠定了电脑培训部发展的基础,确立了电脑培训部的发展思路。正是由于周怀军踏实肯干,对什么事情都勤勤恳恳、认真负责的精神,才使新东方电脑培训部由最初的几十台计算机的规模发展到了现在的几百台计算机的培训中心,学生也由几十人发展到了上万人。由于种种原因,周怀军已经从新东方出去自己创业,我对他的创业十分支持并祝愿他取得更大的成功。当初我就是走上了独立创业的道路,才有了新东方的今天。同时,周怀军依然是新东方电脑培训部的股东,是很好的朋友,对新东方有着真诚的热爱。

2002年,新东方电脑培训部升级为新东方电脑培训公司,现任公司总经理吴雷老师是我不得不说的另外一位新东方重要人物。吴雷毕业于北京联合大学,是一所不算有名的大学。但吴雷是一位非常聪慧、能干而又吃苦耐劳的人,并且有着很好的商业判断能力。在中国的商业圈凭着自己的智慧取得了很好的成就后,他于1998年进入了世界著名的耶鲁大学商学院深造。1999年,他与新东方另一位著名老师钱永强一起回到新东方,帮助新东方参与了和联想公司合作的谈判,和钱永强一起创建了新东方教育在线TOL24.COM,正式成为了新东方管理团队的重要成员。2001年,吴雷老师回到耶鲁去完成在商学院未竟的学业,2002年以优异成绩毕业。本来吴雷可以留在北美工作,取得比在国内更丰厚的报酬,但出于对新东方的热爱和对国内商机的独到眼光使他毫不犹豫地回到了中国,再次加入到了新东方火热的事业之中。

经过几年的发展,新东方电脑培训积累了大量的理论和实际操作经验。周怀军老师在两年前就开始把电脑部积累的这些经验编写成了书,把电脑方面从低级到高级各个领域方面的经验汇集成册,我感到非常高兴。新东方在英语方面已经出版了很多的图书,电脑书籍的出版填补了新东方图书出版的一个空白,必将对大家尤其是广大电脑爱好者起到强大而明确的指导作用。如今,吴雷老师更是雄心勃勃,要把新东方的电脑培训和新东方的电脑图书推上一个更高的世界舞台,我感到非常兴奋。也为新东方整个教育集团以胡敏总裁为首的新一代管理团队的战略眼光和雄心壮志而深感欣慰。同时,我要向清华大学出版社表示衷心的感谢。清华大学出版社是全国著名的出版社,敢于与新东方这样的民间机构合作,表明了清华大学出版社的心胸与远见。

每个人都在刻画着自己的人生轨迹,每个人都在寻找着自己的奋斗目标,如果一个人连自己的人生目标都不知道,连自己这辈子应该干些什么都不清楚,那生活就会变得平庸和无聊。一个人明确了自己的人生目标,再用英语和电脑把自己武装起来,那就如虎添翼,在社会中成功的机会就会更高,成功的速度也会更快。英语是使你走向世界的必不可少的工具,电脑是使你管理现代化和接受新思想必不可少的工具。我衷心希望所有看到这篇序言的人,都能够在生活和学习中努力掌握这两项技能,让它们一同伴随着你向人生的终极目标前进!

2003年3月

# 目 录

|                         |    |
|-------------------------|----|
| 第 1 章 概述                | 1  |
| 1.1 Java 语言简介           | 1  |
| 1.1.1 Java 的产生及其发展      | 1  |
| 1.1.2 Java 语言的特点        | 1  |
| 1.2 Java 虚拟机            | 2  |
| 1.2.1 高级编程语言的程序运行机制     | 2  |
| 1.2.2 Java 虚拟机的作用及其特点   | 3  |
| 1.3 Java 运行环境           | 4  |
| 1.3.1 Java 运行环境简介       | 4  |
| 1.3.2 垃圾收集              | 4  |
| 1.3.3 Java 运行环境的安装和配置   | 5  |
| 1.4 Java 程序举例           | 6  |
| 第 2 章 面向对象的程序设计初步       | 9  |
| 2.1 面向对象的分析和设计简介        | 9  |
| 2.2 类和对象                | 10 |
| 2.2.1 类是对象的蓝图           | 10 |
| 2.2.2 类的声明              | 11 |
| 2.2.3 属性声明              | 12 |
| 2.2.4 方法声明              | 12 |
| 2.2.5 创建对象及访问对象成员       | 12 |
| 2.2.6 信息的隐藏和封装          | 13 |
| 2.3 构造方法                | 14 |
| 2.3.1 构造方法的声明           | 14 |
| 2.3.2 默认的构造方法           | 15 |
| 2.4 Java 源文件的层次结构       | 16 |
| 第 3 章 标识符 关键字 数据类型      | 17 |
| 3.1 注释及分隔符              | 17 |
| 3.1.1 注释                | 17 |
| 案例 3-1 javadoc 文档化工具的使用 | 18 |
| 3.1.2 Java 分隔符          | 19 |
| 3.2 标识符和关键字             | 20 |
| 3.2.1 标识符               | 20 |
| 3.2.2 Java 关键字          | 21 |

|       |                           |    |
|-------|---------------------------|----|
| 3.3   | Java 数据类型 .....           | 21 |
| 3.3.1 | Java 基本数据类型 .....         | 22 |
| 3.3.2 | Java 引用类型 .....           | 24 |
| 3.4   | 值传递 .....                 | 27 |
| 3.5   | 关键字 this .....            | 31 |
| 3.6   | Java 语言编码惯例 .....         | 32 |
| 3.6.1 | 命名惯例 .....                | 32 |
| 3.6.2 | 源程序排版惯例 .....             | 32 |
| 第 4 章 | 运算符 表达式和流程控制 .....        | 34 |
| 4.1   | 变量 .....                  | 34 |
| 4.1.1 | 变量及其作用域 .....             | 34 |
| 4.1.2 | 变量的初始化 .....              | 35 |
| 4.2   | 运算符 .....                 | 36 |
| 4.2.1 | 算术运算符 .....               | 36 |
| 4.2.2 | 关系运算符 .....               | 38 |
| 4.2.3 | 逻辑运算符 .....               | 38 |
| 4.2.4 | 位运算符 .....                | 39 |
| 4.2.5 | 赋值运算符 .....               | 40 |
| 4.2.6 | 字符串连接操作符 + ; .....        | 41 |
| 4.3   | 表达式 .....                 | 42 |
| 4.4   | 分支语句 .....                | 42 |
| 4.4.1 | if-else 语句 .....          | 43 |
| 4.4.2 | switch 语句 .....           | 44 |
| 4.4.3 | 条件运算符 ?: .....            | 45 |
| 4.5   | 循环语句 .....                | 46 |
| 4.5.1 | for 循环语句 .....            | 47 |
| 4.5.2 | while 循环语句 .....          | 48 |
| 4.5.3 | do/while 循环语句 .....       | 48 |
| 4.6   | 特殊的循环流控制语句 .....          | 49 |
| 4.6.1 | break[label] 语句 .....     | 49 |
| 4.6.2 | continue [label] 语句 ..... | 50 |
| 第 5 章 | 数组 .....                  | 52 |
| 5.1   | 一维数组 .....                | 52 |
| 5.1.1 | 一维数组的声明 .....             | 52 |
| 5.1.2 | 一维数组的创建和初始化 .....         | 53 |
| 5.1.3 | 数组元素的引用 .....             | 54 |
| 5.2   | 多维数组 .....                | 55 |
| 5.2.1 | 二维数组 .....                | 56 |

|                              |            |
|------------------------------|------------|
| 5.2.2 二维数组的使用 .....          | 57         |
| 5.3 数组复制 .....               | 58         |
| 案例 5-1 整型数据进制转换 .....        | 59         |
| <b>第 6 章 数据结构基础 .....</b>    | <b>62</b>  |
| 6.1 数据排序算法 .....             | 62         |
| 6.1.1 冒泡法排序 .....            | 62         |
| 6.1.2 选择法排序 .....            | 63         |
| 6.1.3 插入法排序 .....            | 64         |
| 6.2 链表 .....                 | 66         |
| 6.2.1 单链表 .....              | 66         |
| 6.2.2 链表操作 .....             | 67         |
| 6.3 堆栈 .....                 | 69         |
| 案例 6-1 四则表达式求值 .....         | 71         |
| <b>第 7 章 面向对象的程序设计 .....</b> | <b>78</b>  |
| 7.1 包(package) .....         | 78         |
| 7.1.1 package 语句 .....       | 79         |
| 7.1.2 import 语句 .....        | 80         |
| 7.1.3 JDK 中主要的包 .....        | 82         |
| 7.2 类的继承及相关机制 .....          | 82         |
| 7.2.1 类的继承 .....             | 82         |
| 7.2.2 访问控制 .....             | 85         |
| 7.2.3 方法的重写 .....            | 86         |
| 7.2.4 多态性 .....              | 89         |
| 7.2.5 对象造型 .....             | 91         |
| 7.2.6 构造方法的重载和继承 .....       | 93         |
| 7.2.7 Object 类 .....         | 97         |
| 7.3 关键字 static 和 final ..... | 100        |
| 7.3.1 关键字 static .....       | 100        |
| 7.3.2 Singleton 设计模式 .....   | 104        |
| 7.3.3 关键字 final .....        | 104        |
| 7.4 抽象类 .....                | 105        |
| 7.5 接口 .....                 | 107        |
| 7.6 内部类 .....                | 109        |
| <b>第 8 章 Java 异常处理 .....</b> | <b>114</b> |
| 8.1 异常的概念及分类 .....           | 114        |
| 8.1.1 什么是异常 .....            | 114        |
| 8.1.2 Java 异常分类 .....        | 115        |

|               |                                                  |            |
|---------------|--------------------------------------------------|------------|
| 8.2           | Java 异常处理机制 .....                                | 116        |
| 8.2.1         | 捕获异常 .....                                       | 116        |
| 8.2.2         | 声明抛弃异常 .....                                     | 119        |
| 8.2.3         | 人工抛出异常 .....                                     | 120        |
| 8.3           | 用户自定义异常 .....                                    | 121        |
| <b>第 9 章</b>  | <b>Java 控制台应用程序设计 .....</b>                      | <b>122</b> |
| 9.1           | 命令行参数和系统属性 .....                                 | 122        |
| 9.1.1         | Java 命令行参数 .....                                 | 122        |
| 9.1.2         | 系统属性 .....                                       | 123        |
| 9.2           | 数据输入/输出 .....                                    | 124        |
| 9.2.1         | 标准 I/O 操作 .....                                  | 124        |
| 9.2.2         | 文件 I/O 操作 .....                                  | 125        |
| 9.3           | JDK 常用类介绍 .....                                  | 128        |
| 9.3.1         | Math 类 .....                                     | 128        |
| 9.3.2         | String 类 .....                                   | 129        |
| 9.3.3         | StringBuffer 类 .....                             | 131        |
| 9.4           | Collection API 简介 .....                          | 132        |
| 9.5           | Deprecated API .....                             | 135        |
|               | 案例 9-1 猜数程序 .....                                | 136        |
| <b>第 10 章</b> | <b>高级 I/O 流编程 .....</b>                          | <b>139</b> |
| 10.1          | Java I/O 基本原理 .....                              | 139        |
| 10.2          | 基本 I/O 类型 .....                                  | 140        |
| 10.2.1        | InputStream .....                                | 140        |
| 10.2.2        | OutputStream .....                               | 141        |
| 10.2.3        | Reader .....                                     | 142        |
| 10.2.4        | Writer .....                                     | 142        |
| 10.3          | I/O 流的使用 .....                                   | 143        |
| 10.3.1        | FileInputStream/FileOutputStream .....           | 143        |
| 10.3.2        | DataInputStream/DataOutputStream .....           | 144        |
| 10.3.3        | ObjectInputStream/ObjectOutputStream .....       | 146        |
| 10.3.4        | ByteArrayInputStream/ByteArrayOutputStream ..... | 148        |
| 10.3.5        | PipedInputStream/PipedOutputStream .....         | 148        |
| 10.3.6        | FileReader/FileWriter .....                      | 150        |
| 10.3.7        | BufferedReader/BufferedWriter .....              | 151        |
| 10.3.8        | InputStreamReader/OutputStreamWriter .....       | 152        |
| 10.3.9        | PrintStream/PrintWriter .....                    | 152        |

|                                     |     |
|-------------------------------------|-----|
| <b>第 11 章 Java GUI 应用程序开发</b> ..... | 153 |
| 11.1 Java GUI 设计 .....              | 153 |
| 11.1.1 抽象窗口工具集(AWT) .....           | 153 |
| 11.1.2 Java 组件和容器 .....             | 154 |
| 11.1.3 布局管理器 .....                  | 156 |
| 11.2 Java GUI 事件处理机制 .....          | 162 |
| 11.2.1 Java 事件和事件处理 .....           | 162 |
| 11.2.2 多重监听器 .....                  | 165 |
| 11.2.3 事件适配器 .....                  | 167 |
| 11.2.4 内部类和匿名类在 Java 事件处理中的应用 ..... | 169 |
| 11.3 Java GUI 应用程序开发 .....          | 173 |
| 11.3.1 AWT 组件综述 .....               | 173 |
| 11.3.2 AWT 菜单 .....                 | 175 |
| 11.3.3 颜色和字体 .....                  | 177 |
| 案例 11-1 文本文件阅读器开发 .....             | 178 |
| <b>第 12 章 线程</b> .....              | 183 |
| 12.1 线程基本概念 .....                   | 183 |
| 12.1.1 什么是线程 .....                  | 183 |
| 12.1.2 多线程 .....                    | 185 |
| 12.1.3 创建线程的第二种方式 .....             | 186 |
| 12.2 线程状态控制 .....                   | 187 |
| 12.2.1 线程状态 .....                   | 187 |
| 12.2.2 线程调度 .....                   | 190 |
| 12.3 线程的同步 .....                    | 192 |
| 12.3.1 临界资源问题 .....                 | 192 |
| 12.3.2 互斥锁 .....                    | 194 |
| 12.3.3 wait()和 notify()方法 .....     | 195 |
| 12.3.4 生产者 - 消费者问题 .....            | 196 |
| 案例 12-1 考试系统设计与实现 .....             | 198 |
| <b>第 13 章 网络编程</b> .....            | 211 |
| 13.1 网络基础知识简介 .....                 | 211 |
| 13.1.1 什么是计算机网络 .....               | 211 |
| 13.1.2 网络通信协议及接口 .....              | 212 |
| 13.2 URL .....                      | 213 |
| 13.3 Socket 编程 .....                | 214 |
| <b>第 14 章 Java Applet</b> .....     | 218 |
| 14.1 Applet 简介 .....                | 218 |

---

|               |                                 |            |
|---------------|---------------------------------|------------|
| 14.1.1        | 什么是 Applet .....                | 218        |
| 14.1.2        | Applet 运行机制 .....               | 219        |
| 14.1.3        | Applet 安全机制 .....               | 220        |
| 14.2          | Applet 插件标记 .....               | 221        |
| 14.3          | Applet 绘图 .....                 | 224        |
|               | 案例 14-1 Java 绘图板设计与实现 .....     | 226        |
| 14.4          | 多媒体支持 .....                     | 233        |
|               | 14.4.1 图像处理 .....               | 234        |
|               | 14.4.2 声音处理 .....               | 236        |
|               | 14.4.3 Application 的多媒体支持 ..... | 239        |
| 14.5          | JAR 文件 .....                    | 241        |
| <b>第 15 章</b> | <b>Java 数据库连接 .....</b>         | <b>244</b> |
| 15.1          | 数据库简介 .....                     | 244        |
| 15.2          | JDBC .....                      | 245        |
|               | 15.2.1 什么是 JDBC .....           | 245        |
|               | 15.2.2 JDBC 的基本应用 .....         | 246        |
|               | 15.2.3 预处理语句和存储过程 .....         | 250        |
| 15.3          | JDBC-ODBC .....                 | 251        |
|               | 15.3.1 设置 ODBC 数据源 .....        | 252        |
|               | 15.3.2 JDBC-ODBC 编程 .....       | 253        |
|               | 案例 15-1 电子书店设计与实现 .....         | 257        |
| <b>附录 1</b>   | <b>SCJP 模拟试题 .....</b>          | <b>271</b> |
| <b>附录 2</b>   | <b>SCJP 试题分析 .....</b>          | <b>295</b> |

# 第 1 章 概 述

## 本章要点

Java 语言简介

Java 技术核心机制

Java 运行环境的安装和配置

开发并运行简单的 Java 应用程序

## 1.1 Java 语言简介

### 1.1.1 Java 的产生及其发展

Java 是由美国 Sun 公司 20 世纪 90 年代中期正式推出的一种高级编程语言,采用面向对象的编程技术。Java 语言的前身是 Sun 公司开发的一种基于 C++ 的编程语言(Oak),其最初的开发目标是提供一种适合于连网家用电器程序开发的新型语言,因此要求它运行所需资源尽量地少,并能得到多种不同厂家的硬件和操作系统的支持。

因其具有以跨平台性为代表的多种卓越特性,Java 语言一经问世就引起计算机软件行业的革命,受到业界的普遍关注和支持,引起世界范围内的学习和使用 Java 的热潮,并以其迅猛的势头发展至今。

迄今为止,Java 语言已作为一门综合性技术在多个领域得到发展和应用,包括:

- JSP/Servlet——用于 Client/Server 模式下提供动态网站(网页)服务功能。
- Java Bean——用 Java 语言开发可重用、跨平台的软件组件技术。
- EJB(Enterprise JavaBean)——用于开发和部署多层结构的、分布式的、面向对象的 Java 应用系统的跨平台的构建体系结构。
- 嵌入式应用环境——将 Java 运行环境固化到硬件(芯片)中,真正实现 Java 程序在小型电子产品上的自由运行,提供增强功能。
- Java + XML——适用于基于 Web 的数据挖掘的黄金组合。

### 1.1.2 Java 语言的特点

#### 面向对象

和 C++ 等目前主程序程序设计语言类似,Java 采用完全的面向对象技术,Java 程序本身就类(class)定义的形式存在,能很好地实现数据的封装和程序的模块化,更大程度上支持代码的复用。

## 跨平台性(可移执性)

Java 语言针对网络编程而开发,其最主要特点是跨平台(操作系统)性。为使 Java 程序能够在网络上各种不同的计算机和操作系统上正常运行而不需进行改造,Java 语言提供了一种独特的编译和解释执行机制来满足此需求,真正实现了 Java 程序“Write once, run any-time, run anywhere!”。

## 分布性

Java 程序可以方便地进行网络通信,包括普通的数据传输和程序间的协作、远程方法调用(RMI, Remote Method Invocation)等高级应用,实现网络环境中的分布式计算。

## 解释执行

Java 源程序首先由编译器编译成与平台无关的中间产物——字节码文件,然后在运行时由解释器对其进行解释执行。针对不同的操作系统,由 Sun 或其他厂商提供不同的解释器,以实现跨平台性,并简化 Java 程序的执行过程。

## 多线程

Java 语言的多线程性能是指可以在一个程序内部,将此应用程序分解为多个更小的执行单位——线程,对每个线程进行独立的运行控制,各线程之间还具备数据通信、协作的机制。多线程的优点是能使程序具备更好的交互性能和实时控制性能,提高程序运行效率。例如,在 Java 程序运行过程中,当一个线程在访问网络资源,因数据传输速率过低而导致阻塞时,就可以让处于空闲状态的 CPU(Central Processing Unit, 中央处理器)自动服务于其他的线程。

## 在 C++ 基础上做了改进

有一种说法是:Java 语言是 C++ 的改进版,虽不准确但确有其道理。Java 开发较晚,因而得以在以 C++ 为代表各种高级编程语言的基础上发展,引入和发扬其优点,而去除不合理的部分。

Java 语言采用和 C \ C++ 类似的部分语法,因为这些语法被实践证明是适用、有效的;去除了 C \ C++ 中影响程序的可读性、安全性、健壮性及结构合理性的一些成分——全局变量、无条件转移语句、指针、显式内存管理、无节制的数据类型转换、头文件、结构和联合、预处理等功能。

# 1.2 Java 虚拟机

## 1.2.1 高级编程语言的程序运行机制

计算机高级编程语言按其程序的执行方式可分为两种类型:编译型和解释型。

## 编译型

编译型语言是指使用专门的编译器、针对特定平台(操作系统)将某种高级语言源程序一次性“翻译”成可被该平台硬件运行的机器码(包括指令和数据),并包装成该平台的操作系统所能识别和运行的格式。这一过程称为“编译”。最后生成的程序(可执行文件)可以脱离开发环境在特定平台上独立执行。

有时编译过程结束之后,还可能需要对其他编译好的目标代码模块进行连接,即组装两个以上的目标代码模块生成最终的可执行文件,以实现较低层次上的代码重用。

编译型高级语言的程序执行时效率较高,因为它针对特定的平台一次性编译成机器码且可以脱离开发环境而独立运行。但编译后生成的目标码文件也就无法再移植到不同的平台上,如要移植则必须修改源程序,或至少针对不同的平台,采用不同的编译器进行重新编译。现有的多数高级语言,如 FORTRAN、C、C++、Pascal、LISP 等都是编译型的。

## 解释型

解释型语言是指使用专门的解释器将某种高级语言源程序逐条解释成特定平台的机器码指令并立即执行,解释一句执行一句,这类似于会场中的“同声翻译”,而不进行整体性的编译和连接处理。解释型语言相当于把编译型语言相对独立的编译和执行过程混合到一起,而且每一次执行时都要重复进行“编译”,因而执行的效率较低,且不能脱离解释器独自执行。但只要针对不同平台提供相应的解释器,就能方便地实现源程序级的移植,但这是以牺牲程序的执行效率为代价的。

一般来说,程序的可移植性和执行效率存在互斥关系,此消则彼长,难以同时达到最优。

Java 语言根据自身的实际需要采用了一种灵活的机制——编译型和解释型的结合。先采用通用的 Java 编译器将 Java 源程序编译成为仍然是与平台无关的中间产物——字节码文件,然后采用一种虚拟计算机技术,称为 Java 虚拟机(JVM——Java Virtual Machine),运行于特定的平台之上,再由 JVM 对字节码文件进行解释执行。

### 1.2.2 Java 虚拟机的作用及其特点

为实现跨平台性,Java 引进的 Java 虚拟机技术是针对不同的平台,完全由软件模拟或由软件和硬件共同模拟,实现一台虚拟的计算机。Sun 公司制定的 Java 虚拟机规范在技术上规定了 JVM 的统一标准,具体定义了 JVM 的下述细节:

- 指令集(中央处理器[CPU])
- 寄存器
- 类文件格式
- 栈
- 垃圾收集堆
- 存储区

制定这些标准的目的是为了提供一种统一的编译 Java 源代码的规范,最终实现 Java 软件的平台无关性。因为编译生成的字节码文件是针对 JVM 的,是平台无关的。接下来,在

任何不同的时间或场合下,都可以在 JVM 上采用 Java 的运行时解释器解释执行该字节码文件,也就是由 JVM 将字节码指令解释为特定平台的机器码指令并执行。在这一过程中,JVM 起到翻译的作用。针对不同平台,开发不同的 JVM 就好比多个翻译将一份某种语言的作品分别翻译成不同的语言版本,达到信息交流的目的,如图 1-1 所示。

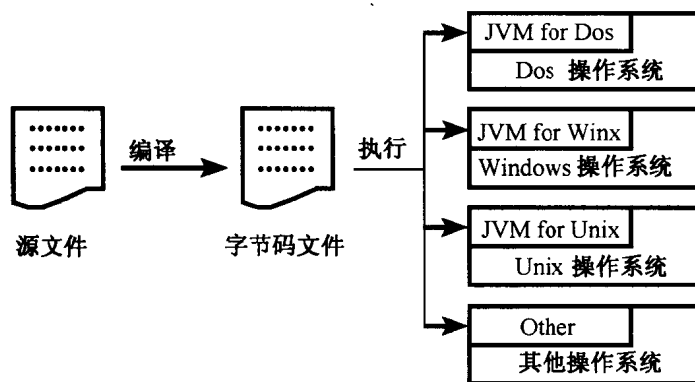


图 1-1 Java 程序运行机制

## 1.3 Java 运行环境

### 1.3.1 Java 运行环境简介

要运行 Java 程序,光有 Java 虚拟机是不够的,因为 JVM 只相当于一台计算机的硬件部分,而计算机光有主板、CPU、内存、硬盘等硬件是不能运行的,还必须配备所需的软件(操作系统等),才能真正进行计算等操作。对于 JVM 来说,所需要的能使之运行的软件部分就是运行时解释器(Runtime Interpreter)。

JVM 和 Java 运行时解释器合在一起构成了 Java 运行环境,简称 JRE(Java Runtime Environment)。JRE 才是“硬件”、“软件”都具备的真正意义上的虚拟计算机。

为了能运行 Java 程序,通常要在用户的计算机上安装 Java 运行环境,有些应用软件本身内嵌了小型的 Java 运行环境,因此也可以直接运行 Java 程序。

JRE 主要担负下述三大任务:

- 加载代码——由类加载器执行
- 校验代码——由字节码校验器执行
- 执行代码——由运行时解释器执行

### 1.3.2 垃圾收集

和 C \ C++ 语言不同,Java 语言不允许程序员直接控制内存空间的使用。内存空间的分配和回收都是由 JRE 负责在后台自动进行的,尤其是无用内存空间的回收操作(garbage

collection,也称垃圾回收),只能由运行环境提供的一个超级线程进行监测和控制,一般是在CPU空闲或内存空间不足时自动进行垃圾回收,而程序员无法精确控制垃圾回收的时机和顺序等。

这种机制解除了程序员管理内存空间的责任,可以避免因内存使用不当,如忘记回收无用内存空间而导致内存泄露等问题。

### 1.3.3 Java 运行环境的安装和配置

在开发第一个Java程序之前,还需要做一些准备工作,就是要在计算机上安装并配置Java运行环境。Sun公司开发的Java开发工具集(JDK——Java Development Kits)提供了编译、运行Java程序所需的各种资源,包括Java编译器、JVM、Java运行时解释器以及常用的Java应用程序编程接口API(Application Programming Interface,常用Java类库)。

运行环境的安装非常简单,目前国内业界使用的主流JDK版本是中文JDK 1.3和英文JDK 1.4,可以到Sun公司网站<http://java.sun.com>免费下载相关资源。只要选择并运行适用于特定操作系统的JDK安装程序(可执行程序),再按照提示指定安装路径即可。系统将自动创建一个名为jdk1.3的子目录,在此目录下面保存了各种资源文件。

运行环境的配置主要是指环境变量Path、CLASSPATH的配置,下面简要介绍它们的作用及在Windows平台下的设置方法。

Path——外部命令搜索路径。标明某个可执行文件的保存位置后,系统都可以通过Path这一环境变量记录的路径找到它,进而实现在任何路径下都能运行该文件。

CLASSPATH——类资源位置搜索路径。标明定义过的Java类的字节码文件可能的保存位置,在其他的类中使用该类时,运行环境能够找到并引入该类。

#### Win2000/NT 下环境变量设置方法

“开始”->“设置”->“控制面板”->“系统”->“高级”->“环境变量”->“(用户变量)新建”,在弹出的对话框中加入新的环境变量,如表1-1所示。

表 1-1 环境变量

| 变量名       | 变量值                                                        |
|-----------|------------------------------------------------------------|
| Path      | C: \ jdk1.3 \ bin                                          |
| CLASSPATH | .;C: \ jdk1.3 \ lib \ tools.jar;C: \ jdk1.3 \ lib \ dt.jar |

其中,假定JDK安装在C:盘根目录下,“C: \ jdk1.3”代表JDK的实际安装路径。新建的环境变量也可以加入到系统变量中,这样以任何用户登录本机都可以使用此环境变量,否则只有以该特定用户的身份登录时才能使用。

#### Win98 下 JDK 环境变量设置方法

修改系统盘根目录下的Autoexec.bat文件:用普通文本编辑器(如notepad)打开该文件后进行修改。例如:假定在Win98操作系统中JDK安装在C:盘,可以在文件最后追加下述两行语句,然后保存: