

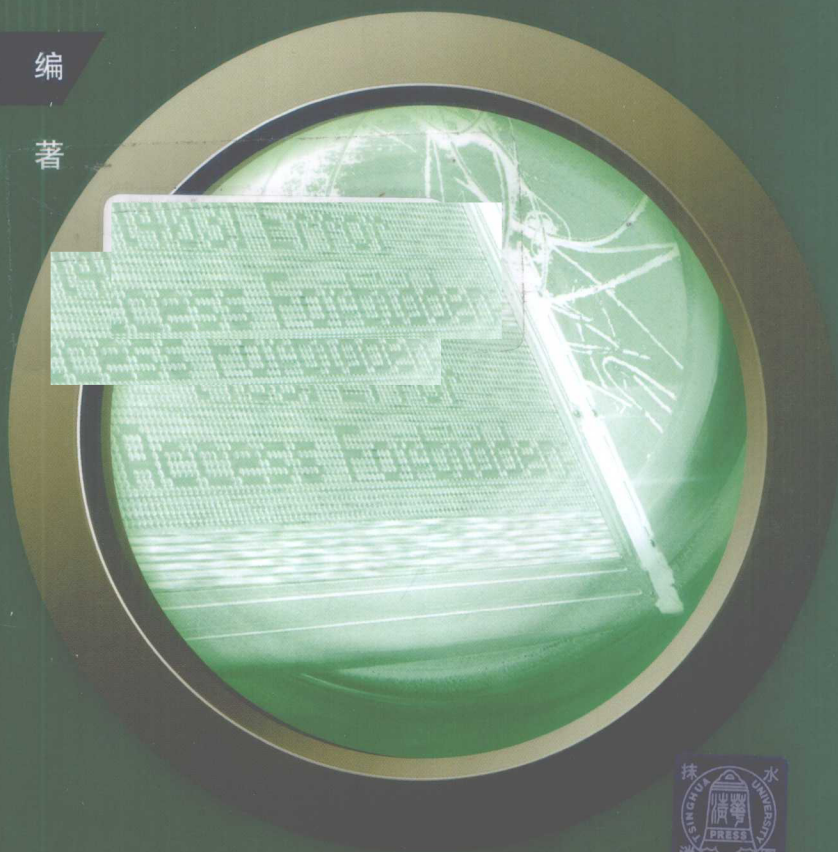


高等院校计算机应用技术系列教材

# Java程序设计 基础教程

赵卓君 主 编

代俊雅 魏志军  
夏力前 张权范 编 著



清华大学出版社 • 北京交通大学出版社

高等院校计算机应用技术系列教材

# Java 程序设计基础教程

赵卓君 主编

代俊雅 魏志军 夏力前 张权范 编著

清华大学出版社 (CIP) 数据

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

## 内 容 简 介

Java 已是目前世界最流行的高级编程语言之一。自诞生以来,Java 迅速成为开发互联网应用程序首选的编程语言。本书特针对普通高等院校和独立学院计算机专业 Java 教学而编写,介绍 Java 的起源、Java 的语法、Java 中面向对象的实现、异常和断言、字符串、输入输出和集合框架。第二部高级教程(稍后出版)则对 Java 的泛型、Applet 小程序、网络媒体通信、线程、GUI 编程、JDBC 数据库连接等都作了深入讲解。整套书应用了大量完整的程序案例来辅助说明。本书基本涵盖 Sun 公司 SCJP 认证要考核的知识点。

本书既可作为普通高等院校和独立学院计算机本科专业的 Java 教材,也可作为 Sun 公司 Sun Certified Java Programmer (SCJP) 认证考试的辅导用书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13501256678 13801310933

## 图书在版编目(CIP)数据

Java 程序设计基础教程 / 赵卓君主编; 代俊雅等编著. —北京: 清华大学出版社; 北京交通大学出版社, 2010.8

(高等院校计算机应用技术系列教材)

ISBN 978-7-5121-0286-6

I. J… II. ①赵… ②代… 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 (2010) 第 167576 号

责任编辑: 谭文芳

出版发行: 清华大学出版社 邮编: 100084 电话: 010-62776969 <http://www.tup.com.cn>

北京交通大学出版社 邮编: 100044 电话: 010-51686414 <http://press.bjtu.edu.cn>

印刷者: 北京东光印刷厂

经 销: 全国新华书店

开 本: 185×260 印张: 18.25 字数: 407 千字

版 次: 2010 年 9 月第 1 版 2010 年 9 月第 1 次印刷

书 号: ISBN 978-7-5121-0286-6/TP·614

印 数: 1~4 000 册 定价: 29.00 元

本书如有质量问题, 请向北京交通大学出版社质监组反映。对您的意见和批评, 我们表示欢迎和感谢。

投诉电话: 010-51686043, 51686008; 传真: 010-62225406; E-mail: [press@bjtu.edu.cn](mailto:press@bjtu.edu.cn)。

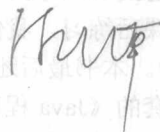
## 言 序 前

Java 程序设计语言是一种完全面向对象且与平台无关的高级编程语言。在面向对象技术越来越成为主流技术，网络应用编程的需求越来越扩大的今天和未来，Java 已经并也无疑将成为未来的主流程序设计语言之一。

在长时间的教学工作中，尤其在独立学院中，我们发现许多学生对于面向对象技术和面向对象程序设计语言的学习和应用颇感困难。无论他们是否曾经学习过其他程序设计语言，他们对于抽象与建模的掌握能力较差。因此，在 Java 程序设计教学中，结合大量案例，训练学生的抽象和建模能力，并应用 Java 语言进行实现显得十分重要。

本书的几位作者长期从事 Java 语言的教学工作，并且参与过许多基于 Java 和 J2EE 平台的实际软件工程项目开发，并具有丰富的教学和实际应用经验。作者使用通俗易懂的语言，结合大量完整的案例阐述 Java 语言的思想及应用基础知识，还设计了实验和课后练习题，并提供所有案例的源代码、上机实验和课后练习答案。

本书既可作为普通高等院校尤其是独立学院 Java 语言的教材，也可作为一般 Java 语言初学者的自学参考书。相信本书的出版将有助于 Java 语言的教学和初学者的自学，也希望作者注意收集使用者的意见，并在使用过程中进一步优化该书的内容与结构。



林国璋 教授

北京理工大学珠海学院

计算机学院院长

2010 年 7 月

## 前言

Java 是一个完全面向对象的高级编程语言。从 1995 年问世以来逐步被 IT 界所接受。Java 语言拥有丰富的类库，具有与平台无关、简单安全等特点，又解决了互联网上的大型应用问题，一出现就被广泛应用在多个领域。

市面上关于 Java 书的数量用多如“恒河数沙”来形容是一点也不夸张的。我们之所以编写这套教材倒不是为了“凑热闹”。近年来，几乎所有普通高等院校和独立学院的计算机本科专业都开设了 Java 程序设计语言这门课。很多学生在已学过 C 语言程序设计有了一定程序设计基础之后再学 Java 语言却仍然遇到不少困难。有些学生甚至反映 Java 语言难学，不如 C# 容易入门、较易掌握。我们在几年的高校 Java 程序设计语言的实际教学中也遇到了类似的问题。在不断地解决这些问题的过程中我们积累了一定的经验，深知对于难以理解的知识点在理论课上需要深入浅出地阐述，在实验课上要细心地引导学生，从实际动手中让学生进一步理解知识点。

本书用通俗易懂的语言来讲解 Java 语言的基础知识，并用大量完整的案例来辅助说明这些知识点。本书共分为 7 章，详细介绍 Java 语言的基础知识，包括 Java 语言的起源、Java 语言的基本语法、面向对象的实现、异常和断言、java.lang 包和字符串、java.util 包和集合框架、java.io 包和输入输出等 Java 基础知识。在编写中，我们尽量保证 Java 知识体系的完整性，也尽量考虑到 Java 教学中的逻辑顺序。为了让教师和学生更好地使用本书，本书每章都安排了实验、课后练习，我们可提供本书的所有案例源代码、上机实验和课后练习答案（请与出版社联系）。本书最后还附有综合复习题以便读者自我练习。关于 Java 语言的高级知识将在与本书配套的《Java 程序设计高级教程》中讲解。

参与本书编写的共有 5 位老师，分别是：赵卓君、代俊雅、魏志军、张权范、夏力前，均来自北京理工大学珠海学院计算机学院。赵卓君负责本书前言、第 1 章 Java 概述和第 4 章异常和断言的编写；张权范负责第 2 章 Java 语法基础的编写；夏力前负责第 3 章面向对象的实现的编写；代俊雅负责第 5 章 java.lang 包和字符串及第 6 章 java.util 包和集合框架的编写；魏志军负责第 7 章 java.io 包和输入输出的编写。本书由赵卓君主编，其余编者的署名按作者姓名的拼音排序，排名不分先后。本书的整合和校对工作由赵卓君完成。

参与本书编写的老师平时都有繁重的教学任务，但仍然坚持完成本书的编写。在这里向所有参与本书编写的老师表示衷心的感谢。没有大家的努力，就没有这本书。另外，在写作过程中也获得很多其他老师的帮助和支持，尤其是北京理工大学珠海学院计算机学院院长林国璋教授的大力支持。林院长还亲自参与本书的审核，给予了很多宝贵的建议，并为该书撰序，在这里特别鸣谢。

尽管我们已经尽了最大的努力编写本书，但是仍然难免有错误和疏漏之处。因此，在这里恳请所有的读者不吝指出，以便我们修改。再次感谢所有读者的鼓励和支持。

编者

2010 年 7 月

# 目 录

|                      |    |
|----------------------|----|
| 第1章 Java 概述          | 1  |
| 简介                   | 1  |
| 1.1 Java 的发展简介       | 1  |
| 1.2 Java 的特点         | 2  |
| 1.3 开发工具包 JDK 及其配置   | 5  |
| 1.4 编写简单的 Java 程序    | 9  |
| 1.4.1 一个最简单的 Java 程序 | 9  |
| 1.4.2 编译和运行 Java 程序  | 10 |
| 1.5 Java 的开发环境       | 11 |
| 1.6 JCreator 的使用     | 12 |
| 总结                   | 14 |
| 上机实验                 | 14 |
| 课后练习                 | 15 |
| 第2章 Java 语法基础        | 16 |
| 简介                   | 16 |
| 2.1 标识符              | 16 |
| 2.1.1 标识符的定义         | 16 |
| 2.1.2 标识符的组成规则       | 17 |
| 2.1.3 分隔符            | 17 |
| 2.2 关键字              | 18 |
| 2.3 数据类型、变量与常量       | 18 |
| 2.3.1 数据类型           | 18 |
| 2.3.2 基本数据类型         | 18 |
| 2.3.3 复合数据类型         | 19 |
| 2.3.4 类型转换           | 20 |
| 2.3.5 常量             | 22 |
| 2.3.6 变量             | 23 |
| 2.4 Unicode          | 25 |
| 2.4.1 汉字字符集          | 25 |
| 2.4.2 Unicode 的定义与特点 | 25 |
| 2.5 运算符              | 26 |
| 2.5.1 算术运算符          | 26 |
| 2.5.2 关系运算符          | 28 |

|       |                     |    |
|-------|---------------------|----|
| 2.5.3 | 逻辑运算符               | 29 |
| 2.5.4 | 位运算符                | 30 |
| 2.5.5 | 位移运算符               | 32 |
| 2.5.6 | 条件运算符               | 34 |
| 2.5.7 | 运算符的优先级与结合性         | 34 |
| 2.6   | 表达式 (Expression)    | 35 |
| 2.6.1 | 算术表达式               | 35 |
| 2.6.2 | 关系表达式               | 36 |
| 2.6.3 | 逻辑表达式               | 36 |
| 2.6.4 | 赋值表达式               | 37 |
| 2.6.5 | 复合赋值运算表达式           | 37 |
| 2.7   | 流程控制                | 37 |
| 2.7.1 | 条件选择语句              | 37 |
| 2.7.2 | switch...case 多分支语句 | 41 |
| 2.7.3 | 循环控制语句              | 44 |
| 2.8   | 数组                  | 51 |
| 2.8.1 | 数组的概念、特征与分类         | 51 |
| 2.8.2 | 一维数组                | 51 |
| 2.8.3 | 多维数组                | 57 |
| 2.9   | 命令行参数               | 59 |
| 2.9.1 | 命令行参数的定义与使用         | 59 |
| 2.9.2 | 命令行参数的转换            | 60 |
|       | 总结                  | 61 |
|       | 上机实验                | 61 |
|       | 课后练习                | 61 |
| 第3章   | 面向对象的实现             | 63 |
| 3.1   | 简介                  | 63 |
| 3.1   | 类与对象                | 63 |
| 3.1.1 | 对象及对象的引用            | 64 |
| 3.1.2 | 类与对象的关系             | 65 |
| 3.1.3 | 抽象                  | 66 |
| 3.1.4 | 封装                  | 67 |
| 3.2   | 在 Java 中实现类         | 68 |
| 3.2.1 | 类中的数据成员             | 68 |
| 3.2.2 | 类中的方法               | 71 |
| 3.2.3 | Java 中类的声明          | 78 |
| 3.2.4 | 使用类的对象实例            | 81 |
| 3.2.5 | 类的初始化与垃圾回收          | 82 |
| 3.2.6 | this 关键字            | 88 |

|       |                  |     |
|-------|------------------|-----|
| 3.3   | 访问修饰符和隐藏实现       | 90  |
| 3.3.1 | 包                | 90  |
| 3.3.2 | 访问控制修饰符          | 92  |
| 3.3.3 | 再谈封装             | 95  |
| 3.4   | 继承               | 97  |
| 3.4.1 | 软件复用与继承的概念       | 98  |
| 3.4.2 | 继承的实现: 父类与子类     | 99  |
| 3.4.3 | 继承中的初始化与收尾       | 102 |
| 3.4.4 | 继承中成员访问和继承的根类    | 105 |
| 3.5   | 多态性              | 107 |
| 3.5.1 | 方法重载 (编译时多态)     | 109 |
| 3.5.2 | 方法重定义 (运行时多态)    | 112 |
| 3.6   | 方法修饰符            | 116 |
| 3.6.1 | static 修饰符       | 116 |
| 3.6.2 | final 修饰符        | 121 |
| 3.6.3 | abstract 修饰符     | 126 |
| 3.7   | 接口               | 127 |
| 3.7.1 | 接口的定义            | 128 |
| 3.7.2 | 接口的实现            | 129 |
| 3.7.3 | 引用接口             | 132 |
| 3.7.4 | 接口的继承            | 133 |
| 3.8   | 嵌套类              | 135 |
| 3.8.1 | 成员类              | 136 |
| 3.8.2 | 局部类              | 138 |
| 3.8.3 | 匿名类              | 139 |
| 3.9   | Java 中的枚举类型      | 140 |
|       | 总结               | 143 |
|       | 上机实验             | 144 |
|       | 课后练习             | 145 |
| 第4章   | 异常和断言            | 153 |
|       | 简介               | 153 |
| 4.1   | 异常               | 153 |
| 4.2   | 异常处理             | 155 |
| 4.2.1 | try...catch 块    | 155 |
| 4.2.2 | finally 块        | 158 |
| 4.2.3 | 多重 catch 块       | 159 |
| 4.2.4 | 嵌套 try...catch 块 | 163 |
| 4.2.5 | 使用 throw 显式引发异常  | 165 |
| 4.2.6 | 使用 throws 回避异常   | 166 |

|     |              |                               |            |
|-----|--------------|-------------------------------|------------|
| 09  | 4.3          | 用户自定义异常 .....                 | 167        |
| 09  | 4.4          | 检验异常和非检验异常 .....              | 168        |
| 59  | 4.5          | 断言 .....                      | 169        |
| 20  |              | 总结 .....                      | 172        |
| 79  |              | 上机实验 .....                    | 172        |
| 80  |              | 课后练习 .....                    | 173        |
|     | <b>第 5 章</b> | <b>java.lang 包和字符串 .....</b>  | <b>177</b> |
| 501 |              | 简介 .....                      | 177        |
| 201 | 5.1          | Object 类 .....                | 177        |
| 701 | 5.2          | 包装类 .....                     | 181        |
| 901 | 5.2.1        | 数值类型包装类 .....                 | 181        |
| 511 | 5.2.2        | Boolean 类 .....               | 183        |
| 911 | 5.2.3        | Character 类 .....             | 183        |
| 911 | 5.3          | Math 类 .....                  | 185        |
| 151 | 5.4          | System 类 .....                | 186        |
| 951 | 5.5          | 字符串 .....                     | 188        |
| 751 | 5.5.1        | String 类 .....                | 188        |
| 851 | 5.5.2        | StringBuffer 类 .....          | 194        |
| 951 |              | 总结 .....                      | 196        |
| 551 |              | 上机实验 .....                    | 196        |
| 851 |              | 课后练习 .....                    | 197        |
|     | <b>第 6 章</b> | <b>java.util 包和集合框架 .....</b> | <b>199</b> |
| 951 |              | 简介 .....                      | 199        |
| 851 | 6.1          | 日期与时间类 .....                  | 199        |
| 951 | 6.1.1        | Date 类 .....                  | 199        |
| 951 | 6.1.2        | 日历类 Calendar .....            | 200        |
| 841 | 6.2          | 随机数类 Random .....             | 203        |
| 941 | 6.3          | 集合框架 .....                    | 205        |
| 841 | 6.3.1        | Collection 接口 .....           | 205        |
| 821 | 6.3.2        | Iterator 接口 .....             | 206        |
| 821 | 6.3.3        | List 接口 .....                 | 207        |
| 821 | 6.3.4        | 向量类 Vector .....              | 208        |
| 821 | 6.3.5        | 栈类 Stack .....                | 210        |
| 821 | 6.3.6        | 数组列表类 ArrayList .....         | 211        |
| 821 | 6.3.7        | 链表类 LinkedList .....          | 212        |
| 951 | 6.3.8        | Set 接口 .....                  | 213        |
| 801 | 6.3.9        | HashSet 类 .....               | 213        |
| 201 | 6.3.10       | TreeSet 类 .....               | 214        |
| 901 | 6.3.11       | Map 接口 .....                  | 215        |

|        |                                               |     |
|--------|-----------------------------------------------|-----|
| 6.3.12 | HashMap                                       | 216 |
| 6.3.13 | TreeMap                                       | 217 |
| 6.3.14 | Collections 类                                 | 218 |
| 总结     |                                               | 220 |
| 上机实验   |                                               | 220 |
| 课后练习   |                                               | 220 |
| 第 7 章  | java.io 包和输入输出                                | 222 |
| 简介     |                                               | 222 |
| 7.1    | File 类                                        | 222 |
| 7.1.1  | File 类构造方法                                    | 223 |
| 7.1.2  | File 类常用方法                                    | 223 |
| 7.2    | 流                                             | 225 |
| 7.2.1  | 流的概念                                          | 225 |
| 7.2.2  | 字节流                                           | 226 |
| 7.2.3  | 字符流                                           | 227 |
| 7.3    | 使用字节流进行文件读/写                                  | 227 |
| 7.3.1  | InputStream 类和 OutputStream 类                 | 227 |
| 7.3.2  | FileInputStream 类和 FileOutputStream 类         | 231 |
| 7.4    | 使用字符流进行文件读/写                                  | 234 |
| 7.4.1  | 基类 Reader 和 Writer 类                          | 234 |
| 7.4.2  | FileReader 类和 FileWriter 类                    | 237 |
| 7.4.3  | InputStreamReader 类和 OutputStreamWriter 类     | 239 |
| 7.4.4  | BufferedReader 类和 BufferedWriter 类            | 242 |
| 7.4.5  | PrintWriter 类                                 | 245 |
| 7.5    | 过滤流                                           | 246 |
| 7.5.1  | FilterInputStream 类和 FilterOutputStream 类     | 247 |
| 7.5.2  | DataInputStream 类和 DataOutputStream 类         | 247 |
| 7.5.3  | BufferedInputStream 类和 BufferedOutputStream 类 | 251 |
| 7.5.4  | LineNumberInputStream 类                       | 253 |
| 7.5.5  | PushbackInputStream 类                         | 255 |
| 总结     |                                               | 256 |
| 上机实验   |                                               | 257 |
| 课后练习   |                                               | 257 |
| 附录 A   | 综合复习题                                         | 259 |
| 参考文献   |                                               | 282 |

# 第1章 Java 概述

## 本章要点:

- ☒ Java 的起源
- ☒ Java 的特点
- ☒ Java 开发环境的配置
- ☒ 编写和运行第一个 Java 程序

## 简介

Java 是一门发展非常迅速的编程语言,从 1995 年 Java 语言诞生开始时只有少数几个公司知道,到现在书店里关于 Java 的书多得数不胜数,互联网上关于 Java 的资料更是遍布几乎所有的 IT 网站。目前参加 Java 认证考试的人也越来越多。

Evans 数据公司 2007 年 12 月的调查数据表明,Java 语言的使用者已在所有编程语言中占据首位。软件强国印度更是公开发表声明,声称其国民生产总值的 2% 与 Java 有关。小小的 Java 语言成了撬动软件产业的力量之源。

Sun 公司中国区首席培训官张瓚认为:“来自美国的消息表明,至少在未来八年内,不会出现可以对 Java 构成威胁的语言。Java 语言广泛应用于各个方面,你们脖子上挂的员工卡就包含了 Java 技术,而宇宙飞船的内部控制程序也是用 Java 开发的。”

由 IT168 网站和 ITPUB&IXPUB 论坛主办的 2007 年 IT 技术应用趋势大调查的结果显示,45.3% 的开发者首选的计算机语言是 Java;其次是 C 语言,比例为 19.1%;其他计算机语言使用比例均低于 9%。

Java 已是目前世界最流行的高级编程语言之一。自诞生以来,Java 迅速成为开发互联网应用程序首选的编程语言。那么 Java 是如何起源的?它具有什么特点,为什么会迅速流行和具有如此的魅力呢?我们又如何开始编写第一个 Java 程序呢?这些都是本章将要讨论的。

## 1.1 Java 的发展简介

20 世纪 90 年代初期, Sun Microsystems 公司的 James Gosling (Java 之父) 等人致力于开发一种计算机语言。当时开发人员所在的办公室窗外有一棵大大的橡树,因此他们给这个语言命名为 Oak 语言 (这是 Java 语言的前身)。Oak 语言其最初的目的是为家电消费电子产品开发一个分布式系统。这样用户可以把电子邮件发给电冰箱、电视机等家用电器,对它们进行控制。当时, C 语言已经很难满足人们的这一愿望。因为 C 语言总是针对特定的芯片,将源程序编译为机器码,该机器码的运行就与特定的芯片指令有关。在其他不同类型的

芯片（如不同类型、不同厂商的电子产品的芯片）上可能无法运行或出现运行错误，甚至可能引起设备的毁坏等灾难性后果。解决这个问题需要一门独立于特定芯片的语言。在这个需求下，Java 语言诞生了。

Java 命名的来由也有一段广为人知的故事。有一天，几位 Sun 公司的 Java 成员组人员讨论给这门新语言取什么名字时，他们当时正在喝着爪哇岛（Java）产的咖啡。有一个人建议说就叫 Java 怎么样？他的提议得到了其他人的赞同，于是 Java 这个名字就这样定了下来，而 Java 的图标正是一杯冒着热气的咖啡。

Java 不仅适合开发大型的桌面程序，而且特别适合开发网络通信应用程序，已经成为目前技术开发中最常用的一种编程语言之一。企业的解决方案正在从客户-服务器（C/S）架构转换到浏览器-服务器（B/S）架构。在传统的 C/S 架构中，要针对不同的机器类型和操作系统类型编写不同的应用程序，开发难度大，而且难于维护。而在 B/S 架构中，终端用户的界面统一为 Internet 浏览器，这其中 Java 起了不可替代的巨大作用。

Java 最初的出现是为了解决 Internet 上的大型应用问题，因此 Java 和 Internet 之间有着千丝万缕的联系。随着 Internet 的迅速发展，Web 的应用日益广泛，Java 语言也得到了迅速发展，而 Java 语言的发展反过来也进一步推动了 Internet 的发展。

1999 年 Sun 公司将企业级应用平台作为 Java 语言的发展方向，包含 J2SE、J2EE 和 J2ME 三个平台。J2SE 是 Java 平台标准版，主要应用于桌面程序和 Java 小应用程序开发；J2EE 主要用于企业级开发和大型网站的开发；J2ME 主要用于手机等移动设备程序的开发。2004 年 Sun 公司发布了 JDK1.5。同时，Sun 公司将版本号 1.5 改为 5.0。JDK1.6 版本也就是 JDK6.0。

## 1.2 Java 的特点

Java 出现后得到迅猛的推广，它和 Internet 的完美结合、平台无关性、解释执行、安全可靠等特点引起了计算机业界的广泛关注。要想学好 Java 语言，当然首先需要了解它的特点。下面介绍 Java 语言的特点。

### 1. 简单性

Java 语言的语法规则和 C 语言非常相似，只有很少一部分不同于 C 语言，并且 Java 还舍弃了 C 语言中复杂的数据类型（如指针和结构体），因此很容易入门和掌握。Java 面向对象的编程思想是从 C++ 继承过来的，但又对其进行了改善。例如：使用接口来取代 C++ 中的多重继承；提供了内存垃圾自动回收机制等，大大简化了人工管理内存的工作。总之，Java 是一门易学易用的高级编程语言。它的这个特点，相信读者在学了 Java 之后会有深刻的体会。

### 2. 可靠性和安全性

Java 从源代码到最终运行经历了一次编译和一次解释，每次都进行检查，比其他只进行一次编译检查的编程语言具有更高的可靠性和安全性。Java 的可靠性和安全性具体体现在以下几点。

Java 是一种强数据类型的编程语言，它要求显式的变量和方法声明，不支持 C 语言

隐式声明的特性,这保证了编译器可以发现变量方法的使用错误,保证程序在运行时更加可靠。

☞ Java 不支持指针数据类型,加上它自动内存垃圾回收的特点,防止了非法访问内存及破坏数据的可能性。

☞ Java 虚拟机 (Java Virtual Machine, JVM) 中有一个字节码校验器,对 Java 源代码编译之后产生的字节码文件进行第二次检查,可以发现数组和字符串访问的越界。

☞ Java 语言提供了异常处理机制,对程序运行过程出现的异常可以进行捕获和处理,安全终止程序的运行并释放占用的内存空间,防止程序因意外突然中断没有释放内存空间。

上述几种机制结合起来,使得 Java 成为一种安全可靠的编程语言。

### 3. 面向对象

面向对象的编程思想旨在计算机程序中来模拟现实世界中的概念。现实世界中任何实体都可以看作是对象;任何实体都可归属于某类事物;任何对象都是某类事物的一个具体的实例;任何对象之间是通过消息相互作用的。如果说传统的面向过程的编程语言(如 C 语言)是以过程为中心、以算法为驱动的话,面向对象的编程语言则是以对象为中心、以消息为驱动的。用公式来表示,面向过程的编程语言可表示为“程序=算法+数据”;面向对象的编程语言可表示为:“程序=对象+消息”。

Java 是一种完全面向对象的编程语言,因此它具有面向对象编程语言都具有的封装、继承和多态三大特点。所谓封装,就是把对象的数据和方法封装成一个整体。Java 语言的基本封装单元是“类”(class)。类定义了该类对象应该具有的形式,指定了对象的数据和操作数据的方法。继承是指一个对象直接使用另一个对象的属性和方法。Java 的类具有层次结构,子类具有父类的属性和方法。与另外一些面向对象编程语言不同的是,Java 只支持单一继承。多态性是面向对象程序设计的又一个重要手段。具体来说,它是指在一个程序中出现多个同名的方法。运行时,根据不同情况执行不同版本,从而实现不同的功能。Java 提供了方法重载和方法重写两种多态形式。

### 4. 平台无关和解释执行

Java 语言的一个非常重要的特点就是平台无关性。它是指用 Java 编写的应用程序编译后不用修改就可在不同的操作系统平台上运行。Java 之所以能平台无关,主要是依靠 Java 虚拟机 (JVM) 来实现的。JVM 是一种抽象机器,它附着在具体操作系统之上,本身具有一套虚拟机器指令,并有自己的栈、寄存器组等。

Java 编程人员在编写完 Java 程序后,Java 编译器将 Java 源代码文件编译后生成字节码文件(一种与操作系统无关的二进制文件)。字节码文件通过 JVM 里的类加载器加载后,经过字节码校验,由解释器解释成当前计算机的操作系统能够识别的目标代码并最终运行。图 1-1 展示了 Java 程序从编译到最后运行的完整过程。

Java 的平台无关性具有深远的意义。首先,它将编程人员梦寐以求的事情(开发一次软件在任意平台上运行)变成了事实。这将大大加快和促进软件产品的开发。其次,Java 的平台无关性正好迎合了“网络计算机”思想。如果常用的应用软件(如字处理软件等)都是用 Java 编写的,并且放在某个 Internet 服务器上,那么用户每当需要使用某种应用软件

时, 只需下载该软件的字节代码即可, 运行结果也可以发回服务器。目前, 已有数家公司开始使用这种新型的计算模式构筑自己的企业信息系统。

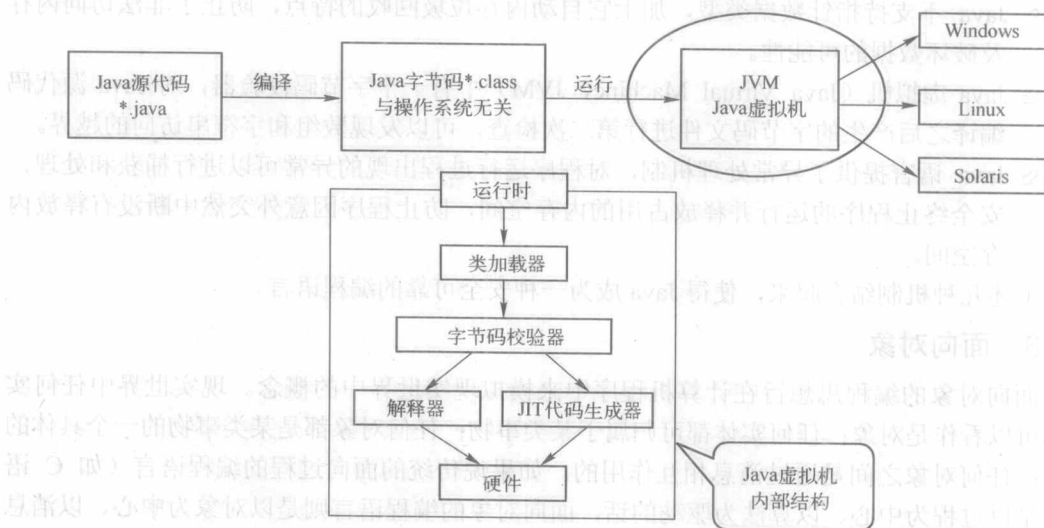


图 1-1 Java 程序编译、解释、运行过程示意图

## 5. 分布式

分布式包括数据分布和操作分布。数据分布是指把数据可以分散在网络的不同主机上, 操作分布是指把一个计算分散在不同主机上处理。

Java 支持 WWW 客户-服务器计算模式。因此, 它支持这两种分布式。对于数据分布, Java 提供了一个叫作 URL 的对象, 它可以打开并访问具有相同 URL 地址的对象, 访问方式与访问本地文件系统相同。对于操作分布, Java 的 Applet 小程序可以从服务器下载到客户端, 即部分计算在客户端进行, 提高系统执行效率。

## 6. 多线程

线程是操作系统的一种新概念, 它又被称作轻量进程, 是分配内存资源最小的单位。多线程是指程序在同一时间执行多个任务的功能。C 和 C++采用单线程体系结构, 而 Java 却提供了多线程支持。

Java 在两方面支持多线程。一方面, Java 环境本身就是多线程的。系统线程运行负责必要的无用内存空间回收, 系统维护等系统级操作; 另一方面, Java 语言内置多线程控制, 可以大大简化多线程应用程序开发。Java 提供了一个 Thread 类负责启动、运行、终止线程, 并可检查线程状态。Java 还可以对线程实行并发控制。利用 Java 的多线程编程接口, 开发人员可以方便地写出支持多线程的应用程序, 提高程序执行效率。必须注意的是, Java 的多线程支持在一定程度上受运行时操作系统的限制。如果操作系统本身不支持多线程, Java 的多线程特性可能就表现不出来。

## 1.3 开发工具包 JDK 及其配置

要想用 Java 语言编写程序,需要安装 Java 开发工具集 JDK。可以到 Sun 公司的网站(<http://www.javasoft.com>)上免费下载 JDK。本书使用的 JDK 版本是 JDK1.6 (目前最新版本是 JDK1.7)。JDK1.6 默认安装的路径是“\Program Files\Java”文件夹。JDK 下面的 bin 文件夹里有两个重要的 exe 文件,分别是 javac.exe 和 java.exe。javac.exe 是用来编译 Java 源程序的,它把 Java 源程序文件 (\*.java) 编译成字节码文件 (\*.class),而 java.exe 则是用来执行字节码文件的。

安装 JDK1.6 非常简单,双击安装文件弹出安装向导对话框(见图 1-2)。

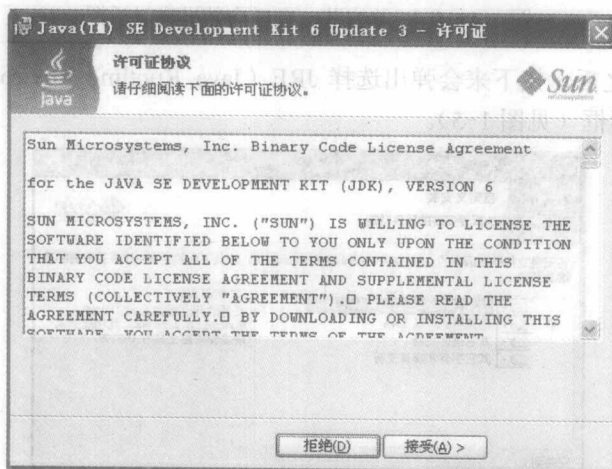


图 1-2 JDK1.6 安装: 安装向导对话框

单击“接受”按钮,接受许可协议,出现选择 JDK 安装路径对话框(见图 1-3)。

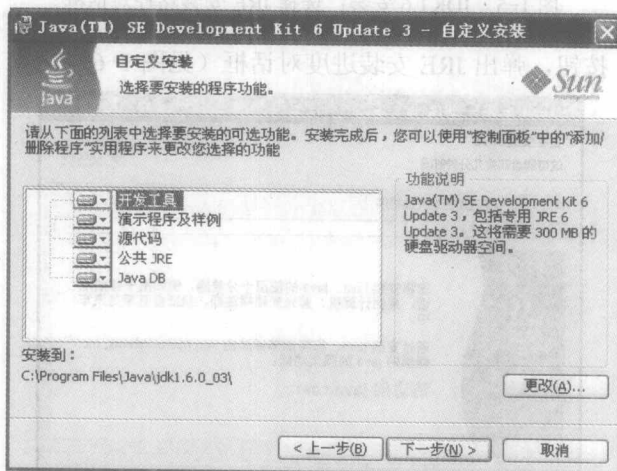


图 1-3 JDK1.6 安装: 选择 JDK 安装路径对话框

单击“更改”按钮可以选择 JDK 安装的路径，若使用默认路径则单击“下一步”按钮，出现 JDK 安装进度对话框（见图 1-4）。

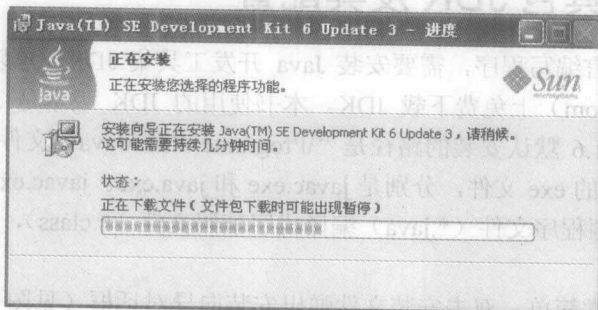


图 1-4 JDK1.6 安装：JDK 安装进度对话框

JDK 安装完毕之后，接下来会弹出选择 JRE（Java Runtime Environment，Java 运行时环境）安装路径对话框（见图 1-5）。

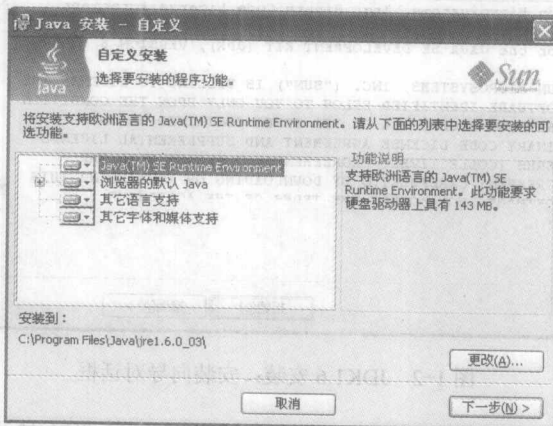


图 1-5 JDK1.6 安装：选择 JRE 安装路径对话框

单击“下一步”按钮，弹出 JRE 安装进度对话框（见图 1-6）。



图 1-6 JDK1.6 安装：JRE 安装进度对话框

JRE 安装完毕后，弹出 JDK 安装完成对话框（见图 1-7）。

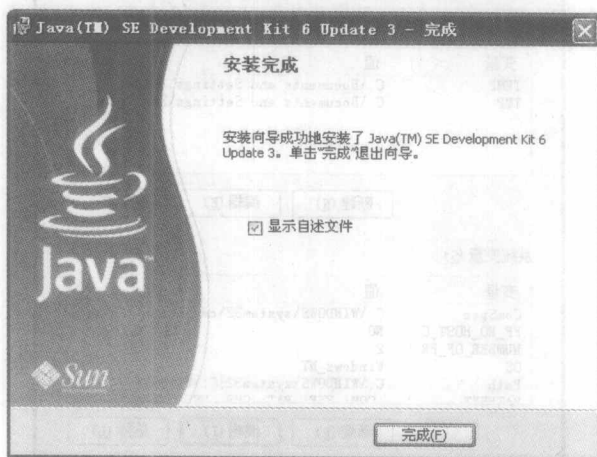


图 1-7 JDK1.6 安装：JDK 安装完成对话框

现在检视系统盘下的“C:\Program Files\Java”文件夹，会看到两个子文件夹：jdk1.6.0\_03 和 jre1.6.0\_03（见图 1-8）。

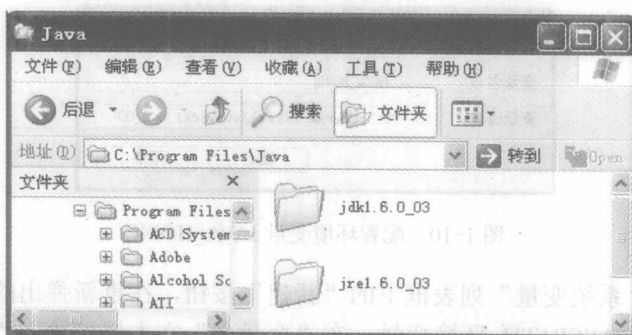


图 1-8 JDK1.6 安装：JDK 安装完成后

JDK 安装完毕之后，它的 3 个重要可执行文件（javac.exe、java.exe 和 javadoc.exe）都在其安装目录下，即系统盘下的“C:\Program Files\Java\jdk1.6.0\_03\bin”文件夹里。如果要在 DOS 命令提示符下使用这 3 个可执行文件，还需要配置 3 个环境变量：JAVA\_HOME、CLASSPATH 和 PATH。具体配置步骤如下。

① 鼠标右击桌面上的“我的电脑”，在弹出的快捷菜单中选择“属性”，打开“属性”对话框，单击“高级”选项卡，选择下面的“环境变量”按钮，弹出 JDK 运行“环境变量”对话框（见图 1-9）。

② 单击“系统变量”列表框下的“新建”按钮，在弹出的“新建系统变量”对话框里，首先配置 JAVA\_HOME 环境变量。在“变量名”文本框里输入“JAVA\_HOME”，然后在“变量值”文本框里输入“C:\Program Files\Java\jdk1.6.0\_03”，然后单击“确定”按钮（见图 1-10）。