21 世纪高职高专规划教材·计算机类

Java 程序设计

主　编　王　唯

副主编　姚　嵩　张洪民

北京理工大学出版社
BEIJING INSTITUTE OF TECHNOLOGY PRESS

内 容 简 介

Java 自问世以来，以其独特的优势迅速风靡了计算机世界。经过数年的发展，它已日益显现出巨大的优势和潜力，成为当今主流的编程语言。本书通过对 Java 语言的全面介绍，使读者学会运用面向对象方法分析和解决实际问题的能力。书中包含大量精心设计并调试通过的编程实例，方便初学者使用。本书共分 10 章，内容包括 Java 语言概述、Java 基本语法、面向对象编程、接口和包、异常、输入与输出、Java 网络编程和数据库编程、使用图形用户界面和创建窗口、Java 多媒体设计、Java 线程。

本书语言通畅，示例丰富，针对所阐述的理论列举了比较典型的实例，便于读者学习、掌握。本书可作为高等职业技术教育教材，也可供从事软件开发以及相关领域的工程技术人员自学使用。

图书在版编目（CIP）数据

Java 程序设计/王唯主编. —北京：北京理工大学出版社，2007.7
21 世纪高职高专规划教材·计算机类
ISBN 978-7-5640-1156-7

Ⅰ. J…　Ⅱ. 王…　Ⅲ. JAVA 语言-程序设计-高等学校：技术学校-教材　Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2007）第 066907 号

出版发行／北京理工大学出版社
社　　址／北京市海淀区中关村南大街 5 号
邮　　编／100081
电　　话／（010）68914775（办公室）　68944990（批销中心）　68911084（读者服务部）
网　　址／http：//www.bitpress.com.cn
经　　销／全国各地新华书店
印　　刷／北京国马印刷厂
开　　本／787 毫米×1092 毫米　1/16
印　　张／16.5
字　　数／382 千字
版　　次／2007 年 7 月第 1 版　　2007 年 7 月第 1 次印刷
印　　数／1～4000 册　　责任校对／张　宏
定　　价／25.00 元　　责任印制／吴皓云

前　言

Java 语言是美国 Sun 公司开发的面向对象程序设计语言（Object Oriented Programming Language），自发布以来取得了巨大的成功。Java 语言具有面向对象和网络编程的优点，经过数年的发展，已日益显现出巨大的优势和潜力，它不仅成为当今主流的编程语言，而且逐步向各个应用领域发展，掀起了计算机软件开发的新浪潮。Java 语言的面向对象、平台无关、安全机制、高可靠性、多线程和内嵌的网络支持等特性，使其成为 21 世纪开发应用程序的首选工具。

本书定位于高等职业技术院校的学生和从事软件开发以及相关领域的工程技术人员，培养读者利用面向对象的技术分析和解决实际问题的能力，指导读者在较短的时间内学会利用最先进的 Java 工具开发软件产品，顺应网络时代对人才的新需求。本书在难易程度上遵循由浅入深、循序渐进的原则；在写作风格上突出其实用性，将复杂的面向对象理论融于具体的实例之中。

全书共分为 10 章，第 1～2 章介绍 Java 语言基础，第 3～4 章介绍 Java 面向对象技术中的对象、类、封装、继承、多态、接口及包等重要内容，第 5 章介绍 Java 强大的异常处理功能，第 6 章介绍 Java 的输入、输出操作功能，第 7 章介绍数据库编程和网络编程，第 8～9 章介绍图形用户界面的设计技术及多媒体编程技术，第 10 章介绍 Java 的多线程技术。

本书语言流畅，示例丰富，针对所阐述的理论列举了比较典型的实例，便于读者学习使用。

本书作者都是从事计算机教学和科研的教师，全书由王唯主编，负责总体构思，并确定章节框架和写作内容，姚嵩、张洪民作为副主编并编写了重点章节，沈楠、李金勇、肖汉夫参与编写。本书在编写过程中自始至终得到了北京理工大学出版社的大力支持，同时也参考了许多学者的研究成果，在此一并表示感谢。

教材中的每章实例和习题都已经调试通过，并且在源文件包（下载地址：www.bitpress.com.cn）中附有全部实例的源代码，可供读者查看。

由于时间仓促，作者水平有限，书中难免有不妥之处，恳请读者批评指正。

目　　录

第 1 章　Java 语言概述

本章将介绍 Java 的发展简史及特点，引导读者构建 Java 基本开发环境，学会编写、编译以及运行简单的 Java 应用程序和 Java Applet，旨在让未曾接触过 Java 编程语言的读者快速入门。

1.1　Java 简介

Java 由于其与生俱来的诸多优点，目前已经在各行各业得到了广泛应用。到处都在讨论 Java，但是 Java 究竟是什么呢?概括说来，和一般编程语言的不同之处在于：Java 不仅是一种面向对象的高级编程语言，它还是一个平台（Platform），应用 Java 更易于开发出高效、安全、稳定以及跨平台的应用程序。目前 Java 还处于快速发展阶段，新的特性和应用仍在不断涌现。本小节将对 Java 的发展历史以及特点进行简要介绍。

1.1.1　Java 语言的发展简史

Java 语言是美国 Sun 公司开发的面向对象的程序设计语言。由于其具有面向对象、高性能、跨平台、安全性高、分布式、支持多线程等优良特性，因而逐步成为网络应用的主流开发语言，被人们誉为“网络上的世界语”。Java 彻底改变了应用软件的开发模式，带来了自微型计算机以来的又一次技术革命，为迅速发展的信息世界增添了新的活力。

Java 语言的产生最早可以追溯到 1991 年，当时 WWW 还没有正式出现，最初是 Sun 公司为了从传统起家的工作站市场进一步扩展到消费性电子产品市场。1991 年 4 月 8 日，Sun 公司成立了 Green 小组，其领导人是 Sun 公司的一位杰出的工程师 James Gosling。该小组的研究方向是能够建立分布式系统结构，同时将软件上的各种新技术移植到消费性电子产品上。在研究过程中，Gosling 发现消费类电子产品的用户不同于工作站用户，他们不关心 CPU 的型号，而只需要一个建立在标准基础之上的、与硬件平台无关的一系列可选方案。为了做到平台无关性，一开始 Gosling 试着从 C++入手，增加了 C++编译器的功能，但不久便发现 C++无法满足需要，而且还存在巨大的安全隐患。例如指针的使用、没有数组越界的检查、缺乏自动的内存管理、过于灵活的数据类型转换等。最后，Green 小组决定重新开发一套全新的语言和系统，于是 Java 的前身——Oak 出现了。Oak 最初是一种用来给蜂窝电话及遥控器之类的设备编程的语言，可以在需要时下载，这样就可以取代以往的设备出厂前的预先编程工作。当设备新增功能时客户可以立即用上，而不用再将设备送回工厂。1993 年，Sun 公司采用此技术制作了遥控器样机，当时尽管令人鼓舞，但并没有得到销售商的广泛支持。这样，这个在技术上很成功的产品在市场上却遇到了挫折。

1994 年，Internet 和 WWW 服务在全球如火如荼地发展起来，Green 小组开始认识到它的可下载技术可以用到 Web 上。于是，Sun 公司决定将 Oak 技术用于 WWW 服务，但首先需要有一个支持它的浏览器。Oak 小组决定在已有工作的基础上自己研制一种新的浏览器。同时，将这种语言改为 Java。Java 是咖啡的俚语，而 Java 语言也的确像咖啡一样可口、令人耳目一新。

在当时，WWW 服务还只是静态的，缺少交互性，只是一些静态的图像和文本。在 Web 页上也的确出现了一些诸如简单的绘画程序的 CGI 脚本，但实际上并没有交互性；客户端的请求还需要送回服务器，给服务器增加了额外的负担。如果程序能够下载并在客户端的浏览器上运行，那么服务器的负担就会减轻，从而能够提供更多的文档服务。这种浏览器就是 Java 小组当时要建立的浏览器。1995 年 5 月，名为 HotJava 的浏览器面市后，引起了巨大的轰动。HotJava 是第一个具有自动装载和运行 Java 程序的浏览器。HotJava 在 WWW 上使人对 Java 产生了很大兴趣，许多公司，如 Netscape 公司在 1995 年 9 月发布了其广为流传的著名的 WWW 浏览器软件 Navigator2.0，该浏览器提供了对 Java 程序的支持。其后，Microsoft，IBM，Oracle，Apple，DEC，Adobe，Silicon Graphics，HP 等大公司纷纷从 Sun 公司处得到了 Java 使用的许可证，以便把 Java 技术应用到他们自己的产品中。1995 年 12 月，Symantec 公司推出了第一个 Windows 平台上的 Java 开发工具 Symantec Cafe；1996 年初，Sun 公司也推出了免费的 Java 编程环境 JDK 1.0。1997 年 11 月，以 Sun 公司为代表提出的 Java 规范被国际标准化组织批准。另外，众多的软件开发商也开发了许多支持 Java 的软件产品。例如，Borland 公司的基于 Java 的快速应用程序开发环境 Latte、Metrowerks 公司和 Natural Intelligence 公司分别开发的基于 Macintosh 的 Java 开发工具、Sun 公司的 Java 开发环境 Java Workshop、Microsoft 公司开发的系列 Java 产品。数据库厂商如 Illustra，Sybase，Versant，Oracle 等都致力于开发支持 HTML 和 Java 的 CGI(Common Gateway Interface)。在以网络为中心的计算时代，不支持 HTML 和 Java 就意味着应用程序的使用范围只能局限于同质的环境。Java 已经成为网络世界中不可或缺的重要技术之一。

严格说来，Java 技术不只是包括 Java 语言，但 Java 语言确实是整个 Java 技术的根源和基础。

1.1.2 Java 语言的特点

Java 是一种简单、安全、容易使用、面向对象、可移植、高性能、多线程的语言。

1. 简单性

Java 语言简单高效，基本 Java 系统（编译器和解释器）所占空间不足 250KB。由于 Java 最初是为了对家用电器进行集成控制而设计的，因而具备简单明了的特征。

如上所述，Java 从 C++演变而来，保留了 C++的许多优点，去除了 C++中易产生错误的功能，简化了内存管理，减轻了程序员进行内存管理的负担。

2. 面向对象

面向对象技术是现代软件工业的一次革新，提高了软件的模块化程度和重复使用率，缩

短了软件开发时间，降低了开发成本。在 Java 之前虽然已经有面向对象的程序设计语言问世，但有些如 C++并不是完全的面向对象，而是面向过程和面向对象的混合体。Java 则是完全面向对象的程序设计语言。

3. 安全性

Java 是可以用在网络及分布式环境下的网络程序设计语言。在网络环境下，语言的安全性变得更为重要。Java 提供了许多安全机制来保证其使用上的安全性。

4. 平台独立

平台独立指程序不受操作平台的限制，可以应用在各种平台上。Java 源程序经过编译后生成字节码文件，而字节码与具体的计算机无关。只要计算机安装了能解释执行字节码的 Java 虚拟机 JVM（Java Virtual Machine），就可以执行字节码文件，从而实现了 Java 的平台独立性。

为了理解 Java 语言的平台独立性，在此对 Java 虚拟机做一下简单介绍。大部分高级语言的源程序必须经过编译或解释程序翻译成机器语言，才能在计算机上执行。例如 C、C++等属于编译式语言，而 Basic 和 Lisp 属于解释型语言。

然而，Java 程序却比较特殊，它必须先经过编译，再经过解释过程才能执行。通过编译器，Java 语言源程序转换成与平台无关的中间编码，Java 称之为字节码（Byte Codes）。字节码再经过解释器的解释，转换为机器码，便可在计算机上运行。

任何可以运行 Java 字节码的程序都可以看成是 Java 的虚拟机，如浏览器和 Java 的开发工具等都可以看成是 JVM 的一部分。

字节码的最大好处是可跨平台运行，也就是说字节码使“编写一次，到处运行”的梦想成真。当将 Java 的源程序用任何一种 Java 编译器编译成字节码后，便可运行在任何含有 JVM 的平台上，无论是 Windows，Mac OS 还是 UNIX 均可。

这种跨平台特性是 Java 得到快速普及的主要原因之一。

5. 多线程

Java 具备内建的多线程功能，可以将一个程序的不同程序段设置为不同的线程，使各线程并发、独立地执行，提高系统的运行效率。

6. 网络功能

Java 能从全球的网络资源获取所需信息，如数据文件、影像文件、声音文件等，并对所得信息进行处理，所以说 Java 是一种网络语言。

7. 执行效率

Java 的字节码需要经过 Java 虚拟机 JVM 解释成机器码才能执行，所以速度上较慢。但是随着 JVM 技术的进步，使得其执行速度直逼 C 与 C++。

1.2 Java 程序的开发过程

Java 程序的开发过程如图 1–1 所示。

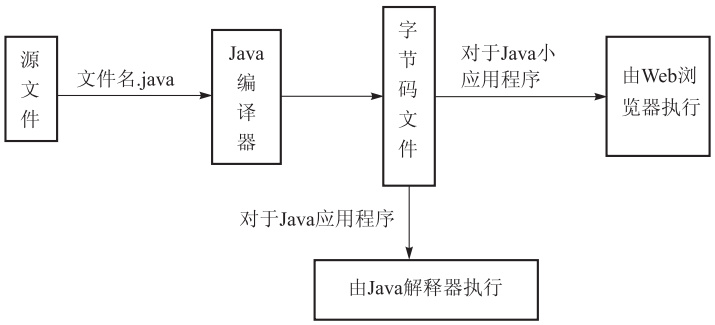

图 1–1 Java 程序的开发过程

本书中的所有程序均已调试通过。

（1）编写源文件：使用一个文字编辑器，如 Edit 或记事本，来编写源文件。不能使用 Word 编辑器，因它含有不可见字符。将编写好的源文件保存起来，源文件的扩展名必须是.java。

（2）编译 Java 源程序：使用 Java 编译器（javac.exe）编译源文件得到字节码文件。

（3）运行 Java 程序：Java 程序分为两类——Java 应用程序和 Java 小应用程序，Java 应用程序必须通过 Java 解释器（java.exe）来解释执行其字节码文件；Java 小应用程序必须通过支持 Java 标准的浏览器来解释执行。后面将讲解怎样使用解释器和浏览器来运行程序，使用普遍的 Netscape Navigator 和 Microsoft Explorer 都完全支持 Java。

1.3 一个简单的 Java 程序

1. 编写源程序

```
public  class  Hello
{
  public static void main (String  args[])
  {
    System.out.println("你好，很高兴学习 Java");
  }
}
```

注意：Java 源程序中语句所涉及的小括号及标点符号都是英文状态下输入的括号和标点符号，比如程序中"你好，很高兴学习 Java"的引号必须是英文状态下的引号，而字符串里面

的符号不受限制。

程序说明如下。

（1）一个 Java 源程序是由若干个类组成的。如果学过 C 语言，就会知道一个 C 源程序是由若干个函数组成的。上面的这个 Java 应用程序简单到只有一个类，类的名字是由我们起的，叫 Hello。

（2）class 是 Java 的关键字，用来定义类。public 也是关键字，说明 Hello 是一个 public 类（我们将会系统地学习类的定义和使用）。第一个大括号和最后一个大括号以及它们之间的内容叫做类体。

（3）public static void main(String args[])是类体中的一个方法，之后的两个大括号以及之间的内容叫做方法体。一个 Java 应用程序必须且只有一个类含有 main 方法，这个类称为应用程序的主类。main 也是关键字。public、static 和 void 分别是对 main 方法的说明。在一个 Java 应用程序中 main 方法必须被说明为 public static void。

（4）String args[]的作用是声明一个字符串类型的数组 args[]（注意 Sting 的第一个字母是大写的），它是 main 方法的参数。main 方法是程序开始执行的位置。

现在将源文件保存到 C:\1000\中并命名为 Hello.java。注意不可写成 hello.java，因为 Java 语言是区分大小写的。源文件的命名规则是这样的，如果源文件中有多个类，那么只能有一个类是 public 类，如果有一个类是 public 类，那么源文件的名字必须与这个类的名字完全相同，扩展名是.java。如果源文件没有 public 类，那么源文件的名字只要和某个类的名字相向，并且扩展名是.java 就可以了。

2. 编译

创建了 Hello.java 这个源文件后，就要使用 Java 编译器对其进行编译：

```
C:\1000\javac Hello.java
```

编译完成后生成一个 Hello.class 文件，该文件称为字节码文件。这个字节码文件 Hello.class 将被存放在与源文件相同的目录中。

如果 Java 源程序中包含了多个类，那么用编译器 javac 编译完源文件将生成多个扩展名为.class 的文件，每个文件中只存放一个类的字节码，其文件名与该类的名字相同。这些字节码文件将被存放在与源文件相同的目录中。

如果对源文件进行了修改，那么必须重新编译，生成新的字节码文件。

注：如果在安装时没有另外指定目录的话。javac.exe 和 java .exe 被存放在 C:\jdk1.4.2\bin 下，如果想在任何目录下都能使用编译器和解释器，那么应在 DOS 命令提示符下运行下列命令：C:\>path C:\jdk1.4.2\bin 或将 path C:\jdk1.4.2\bin 放到 autoexec.bat 文件中。

3. 运行

使用 Java 解释器（java.exe）运行这个应用程序：

```
C:\1000\>java Hello
```

注意：当 Java 应用程序中有多个类时，java 命令后的类名必须是包含了 main()方法的那个类的名字。

1.4 一个简单的 Java 应用程序（Java Applet）

1. 编写源程序

```
import  java.applet.*;
import  java.awt.*;
public  class  boy  extends  Applet
{
  public void paint(Graphics g)
  {
     g.setColor(Color.red);
     g.drawString("我一边喝着咖啡，一边学习 Java 呢",5,10);
     g.setColor(Color.blue);
     g.drawString("我学得很认真，尽管我的对面坐着一个美丽的女孩",5,30);
  }
}
```

一个 Java Applet 也是由若干个类组成的，一个 Java Applet 不再需要 main 方法，但必须有一个类扩展了 Applet 类，即它是 Applet 类的子类。我们把这个类叫做这个 Java Applet 的主类，Java Applet 的主类必须是 public 的。Applet 类是系统提供的类。当保存上面的源文件时，必须将其命名为 boy.java。假设保存 boy.java 在 C:\1000\目录下。

注：上述源程序中使用了 import 语句，这是因为要使用系统提供给的 Applet 类，Applet 类在包 Java.applet 中。包 java.applet 中有很多类，Java 语言把一些类放在一起叫做一个包，这里 Java.applet 是一个包的包名，关于包后面还会讲解。如果不使用 import 语句，则主类必须写成：public boy extends java.applet.Applet。Graphics 也是包 java.awt 中的一个类。

2. 编译

```
C:\1000\>javac boy.java
```

编译成功后，在文件夹 1000 下会生成一个 boy.class 文件。如果源文件有多个类，将生成多个.class 文件，都和源文件放在同一文件夹里。

3. 运行

Java Applet 必须由浏览器来运行，因此必须编写一个超文本文件（含有 Applet 标记的 Web 页），通知浏览器运行这个 Java Applet。

下面是一个最简单的一个 HTML 文件，通知浏览器运行 Java Applet，使用记事本编辑如下：

```
<applet code=boy.class    height=300    width=300>
</applet>
```

超文本中的标记<applet…>和</applet>通知浏览器运行一个 Java Applet，code 通知浏览器运行哪个 Java Applet。code 的“=”后面是主类的字节码文件，当然这个字节码文件的扩展名是.class，而它的名字和源文件的名字是相同的。Width、height 属性规定了这个 Java Applet 的宽度和高度，单位是像素。要想让浏览器运行一个 Java Applet，<applet…></applet>标记中的 code，height，width 都是必需的。另外还有一些可选的项，如：vspace 用于设置小程序与其周围对象的垂直距离；hspace 用于设置水平距离等。

现在把上面编辑的文件 boy.html（扩展名必须是.html，名字不必是 boy，可以是一个自己喜欢的名字）保存在 C:\1000 目录下，即和 boy.class 在同一目录下。如果不是这样，则必须在文件 boy.html 中增加选项 codebase，来指定小程序中的.class 文件所在的目录。

注：（1）也可以使用 JDK 提供的 Applet Viewer 来调试小程序。如，在 DOS 命令行执行：

```
C:\1000\appletviewer  boy.html
```

（2）g.drawString(“我一边喝着咖啡，一边学 Java 呢”，5，10)的作用是在程序中画字符串。数字 5 和 10 规定了字符串的位置，即从距小程序左面 5 个像素、距上面 10 像素的位置开始从左到右画字符串“我一边喝着咖啡，一边学 Java 呢”。

1.5　开发环境

1.5.1　JDK 简介

Sun 公司在推出 Java 语言的同时，推出了 Java 的一系列开发工具，例如 JDK（Java Development Kit）。JDK 是可以从网上免费下载的 Java 开发工具集。随后，一些著名的公司也相继推出了自己的 Java 开发工具，例如 Microsoft 公司的 Visual J++、Borland 公司的 JBuilder、IBM 公司的 Visual Age for Java 等。下面简单介绍 JDK 开发工具。

JDK 提供了丰富的开发工具，包括 Java 编译器、Java 解释器、Java 小程序浏览器、Java 类分解器、C 文件生成器、Java 调试器等。

1. Java 编译器

Java 编译器（javac.exe）本身就是一个用 Java 语言编写的应用程序，与其他的编译语言不同，它的作用是把 Java 源程序（.java 文件）编译生成一个与程序运行平台无关的字节码文件（.class 文件）。

Java 编译器的编译格式为

```
javac[options]  source_file
```

其中，source_file 是扩展名为.java 的源程序，options 为选项，选择参数如下。

（1）classpath<路径；路径；…>——指定在编译过程中查找出的某个类定义搜索路径表，路径间用“；”分隔。

（2）deprecation——显示在 JDK 中禁止使用的旧版方法。

（3）d<目录；目录；…>——指定类文件存放的路径，目录间用“；”分隔。

（4）g——为调试器生成附加信息，这是默认方式。

（5）nowarm——不显示警告错误信息。

（6）verbase——代码优化。

（7）debug——设置允许调用 Java 调试器 jdb。

2. Java 解释器

Java 源程序文件经过编译后生成的字节码文件是不能直接在机器上运行的，需要经过 Java 解释器（java.exe）的解释执行。调用格式为

```
java[interpreter options]  classname
```

其中，classname 是扩展名为.class 的类名，即需要解释执行的程序，并且在此类中必须包含一个 main()方法，程序的执行正是从 main()方法开始的；interpreter options 为选项部分，选择参数如下。

（1）classpath<路径；路径：…>——设置定义类的搜索路径。

（2）cs，checksource——检查类加载后，类文件和源程序间的一致性。

（3）debug——设置允许调用的 Java 调试器 jdb。

（4）ms initmem [k/m]——设置初始内存池容量，单位为 KB 或 MB。

（5）mx maxmem[k/m]——设置最大内存池容量，单位为 KB 或 MB。

（6）noasyncgc——设置自动收集无用空间的动作。

（7）noverify——关闭类文件检查。

（8）verify——检查类文件的所有代码。

（9）verifyremote——对类加载器加载的类代码检查，为默认方式。

（10）oss stacksize[k/m]——设置每个线程代码的长度，单位为 KB 或 MB。

（11）help——使用帮助信息。

（12）ss stacksize[k/m]——设置初始内存池容量，单位为 KB 或 MB。

（13）v，verbose——显示类文件装载信息。

（14）verbosege——显示自动收集无用空间动作的信息。

（15）version——版本信息。

实质上，Java 解释器是字节码解释器，只要指定一个类文件名，它就会自动装载程序需要的类文件，通过位查机制确定一个类是否合法，确保解释执行的字节码不会破坏 Java 语言的约定。

Java 解释器能够自动在.zip 文件中查找所需要的类，只是这里的 zip 文件是未压缩的。如果把长文件名的类打包在 zip 文件中，可以满足不支持长文件名系统对长文件名类访问的需要。

如果用 main()方法或其他方法创建了线程，Java 解释器将一直运行完 main()方法及所有的线程再退出；否则，执行完 main()方法后即退出终止。

3. Java 类分解器

Java 类分解器 javap 用于对类文件进行反汇编，分解类的组成单元。调用格式为

```
javap[options]classname[classname…]
```

其中，options 为选项表，选择参数如下。