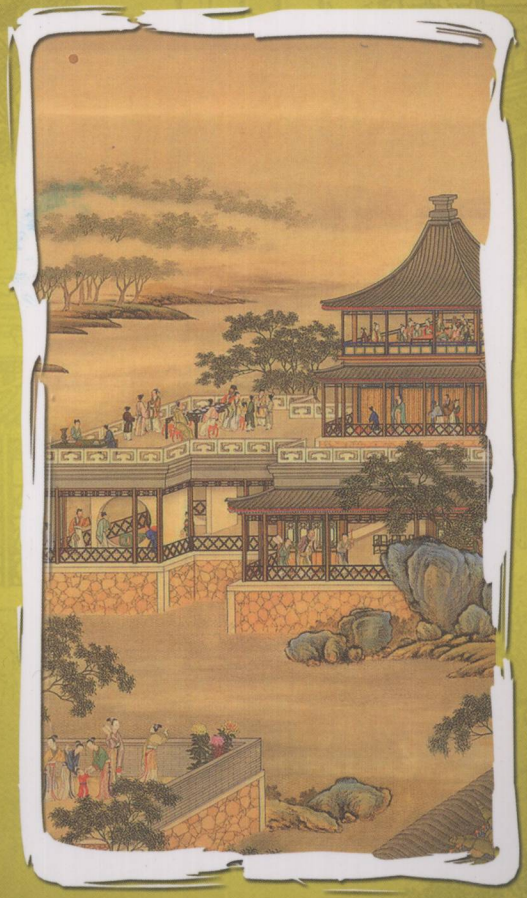
TURING 高等院校计算机教材系列

Java语言程序设计

徐保民 陈旭东 李春艳 编著

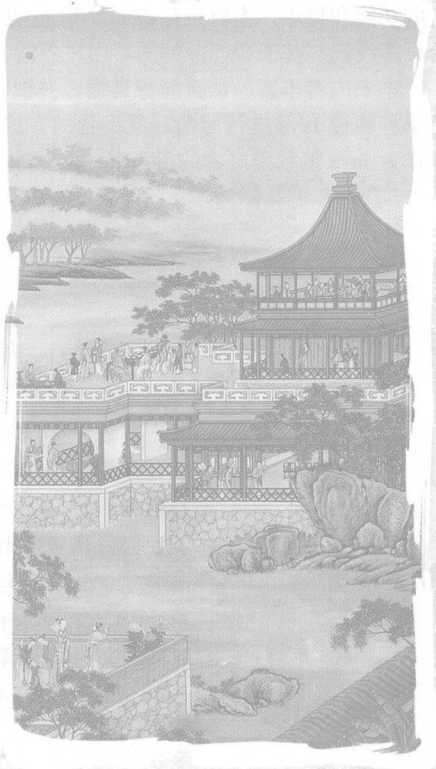


人民邮电出版社
POSTS & TELECOM PRESS

TURING 高等院校计算机教材系列

Java语言程序设计

徐保民　陈旭东　李春艳　编著



人民邮电出版社
北　京

图书在版编目（CIP）数据

Java语言程序设计 / 徐保民，陈旭东，李春艳编著.
—北京：人民邮电出版社，2009.8
（高等院校计算机教材系列）
ISBN 978-7-115-19966-9

Ⅰ. J… Ⅱ. ①徐…②陈…③李… 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099330号

内 容 提 要

Java是当今最流行的程序设计语言之一。本书以Java最新版本Java SE 6.0为基础，全面介绍Java语言的功能和特点，主要包括Java语言基础知识、面向对象编程技术、异常处理、输入/输出流、泛型、集合、多线程、基于Swing的图形用户界面、网络编程等内容。

本书内容丰富、语言简练易懂，并辅以大量的示例，即使没有程序设计语言基础的读者，也可以轻松地掌握通过Java进行面向对象编程的方法。本书可作为高等院校计算机或相关专业的Java语言教材，也可作为Java编程爱好者的参考书。

高等院校计算机教材系列

Java语言程序设计

◆ 编　　著　徐保民　陈旭东　李春艳
责任编辑　杨海玲
执行编辑　马晓燕

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街14号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京顺义振华印刷厂印刷

◆ 开本：800×1000　1/16
印张：20
字数：498千字　　2009年8月第1版
印数：1－3 000册　　2009年8月北京第1次印刷

ISBN 978-7-115-19966-9/TP

定价：39.00元

读者服务热线：(010)51095186　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前　言

Java是一种纯面向对象的程序设计语言。它由C++发展而来，充分吸取C++语言的优点，同时摈弃了C++中诸如指针、内存申请和释放等影响程序健壮性的特性。可以说，Java语言是站在C++语言这个“巨人”的肩膀上发展起来的。

目前，Java语言已经真正成长为一门被广泛使用的程序设计语言，而且已成为软件开发者应当掌握的一门基础语言，因为很多新的技术领域都涉及Java语言。

现在市面上有关Java的书不少，但是适合于课堂教学的书却不多。结合多年的教学经验和工程实践基础，作者力图使本教程具有如下特点。

(1) 内容新而全。“新”体现在以Java SE 6.0为基础来讲解Java语言的功能和特点，“全”体现在对一般教材没有涉及到的字符串常量池、断言和日志、执行器、Annotation注释及反射机制等都有较详细的论述。

(2) 螺旋式推进。对新概念的引入和讲解循序渐进、逐步展开，确保读者能够掌握每一阶段所讨论的内容。

(3) 浅显易懂。通过翔实的示例展现Java语言的编程思想，使读者在较短的时间内掌握较多的知识。

(4) 强调实用性。既有就事论事而编写的短小程序，也有贯穿全文的综合的、易于理解的一个共享白板工具的实现，特别适合读者动手练习。所有示例程序都在Java SE 6.0环境下调试运行通过，读者可以直接参照使用。

全书包括12章和4个附录。

- 第1章概述了Java发展历史、特点和运行机制，并对一个虽然简单但五脏俱全的“Hello World!”程序进行了详细分析。
- 第2章主要对Java语言的基本成分进行介绍，包括数据类型、变量和常量、运算符、表达式、流程控制等。它们是利用Java语言进行程序设计的基础。
- 第3章以Java的面向对象编程核心概念——类和对象为线索，循序渐进地对方法、成员变量、修饰符、this关键字、方法重载、包等概念和它们的用途进行了详细的讲解。同时还就JAR文件、与字符串处理有关的类及数组的概念和使用进行了介绍。
- 第4章主要讨论了面向对象程序设计的两个重要特征——继承和多态在Java中的具体实践。对Java语言的最终类、抽象类、Object类、接口、内部类等概念进行了比较详细的讨论。
- 第5章集中讨论了Java的异常处理机制和软件调试的日志及断言机制，这是Java语言为防

止程序设计中的错误提供的全新解决方案。

- 第6章在介绍Java的I/O包的整体架构的基础上，选择性地介绍了几个常用I/O处理类的特点及使用。
- 第7章重点讨论JDK5.0引入的、旨在提高代码安全性和重用性的泛型。主要通过几个实例讨论带有泛型的类、泛型方法、有界类型参数、类型擦除等重要概念。
- 第8章就Java中常用的集合接口Set（集）、List（列表）和Map（映射）的功能、特点及其相关实现进行介绍。本章内容对于简化程序开发工作和优化程序大有裨益。
- 第9章主要讲解Java多线程的基本概念和应用方法，以及线程同步、线程管理、执行器等内容。
- 第10章主要讲解基于Swing的GUI编程的基本思想和方法，包括常用Swing容器、布局管理、事件处理等内容。
- 第11章重点介绍常用的Swing用户界面组件，并对每一个组件都给出了具体的应用示例，以帮助读者更好地掌握图形用户界面的设计方法。
- 第12章讨论Java网络编程。内容包括网络编程基础知识，使用URL方式进行网络连接的方法，基于TCP和UDP的网络编程等。
- 附录A~附录D分别对Java开发平台，文档注释、Annotation注释及反射机制，Java事件类，JComponent类常用方法进行了简单介绍。

全书第1章～第9章由徐保民编写，第10章～第12章及附录A～附录D由李春艳编写。全书由陈旭东负责统稿和定稿。

本书的出版得到了人民邮电出版社图灵公司的大力支持，北京交通大学计算机与信息技术学院、软件学院相关课程的老师也为本书的编写提出了不少的建议，在此一并表示深深的谢意。

由于作者水平所限，难免会存在很多不足和错误，恳请各位读者不吝赐教，作者的电子邮箱为xubaomin@yahoo.com.cn。

目　录

第1章 Java概述

在计算机编程技术发展史上，Java是发展最快、普及最快的技术之一。在问世后短短几年时间内，Java技术就以其独特的魅力引起了许多程序员的关注。

本章主要介绍Java的发展历史、特点和Java应用程序的执行步骤，并以一个简单的Java应用程序为例，对Java应用程序的基本构成元素进行了较详细的讨论。可以说，本章是你开始Java学习之旅的基石。

1.1 Java发展历史

Java的发展历史可以追溯到1991年4月，它来自于美国Sun公司[①]的一个由James Gosling博士领导的名为“绿色”的项目。Java的前身是“绿色”项目组的技术人员为交互式电视和电冰箱等家用消费类电子产品所开发的一种程序语言Oak（橡树）。Oak语言的特点是：在执行程序前，先生成“中间代码”，在任何一种设备上只要安装特定的解释器，该设备就可以运行“中间代码”。1992年，Sun公司利用这种技术参与一个交互式电视项目的招标，结果被竞争对手打败。在随后的一段时间内，Oak语言基本上无任何发展。

1993年，随着WWW的出现以及Mosaic浏览器（美国伊利诺伊大学国家超级计算应用中心开发）和Netscape浏览器（Netscape公司开发）的先后问世，James Gosling博士意识到WWW需要一个不依赖于任何硬件平台和软件平台，并具有较高实时性、可靠性和交互性的浏览器。于是“绿色”项目组的技术人员对Oak进行改造，推出了新一代面向对象的程序设计语言Java。同时由Naughton和Jonathan Payne负责使用Java语言开发一个全新的Web浏览器“Web Runner”。到1994年秋天，他们完成了该浏览器的开发工作，随后将其更名为“HotJava”，并于1995年5月23日发布。“HotJava”在产业界引起了巨大的轰动。Sun公司也绝没想到本想用于消费类电子产品开发的程序语言却率先在网络中得到广泛应用。这真是应了“有心栽花花不成，无心插柳柳成荫”的格言。由于Oak这个商标已被他人注册，Sun公司的工程师们便以他们常享用的咖啡（Java）来重新命名。自此，James Gosling博士也多了一个“Java之父”的称号。

1996年，Sun公司成立Javasoft分公司，专门负责Java事宜。同时，正式发布Java开发者版本JDK 1.0。该版本包括运行Java程序所需的运行环境JRE（Java Runtime Environment）和开发Java

① Sun公司目前已被Oracle公司收购。——编者注

程序的开发环境JDK（Java Development Kits）两部分。在JDK 1.0时代，除用于开发图形用户界面的AWT类库外，其他类库并不完整，即JDK 1.0并不能完全适用于应用程序的开发。

1997年JDK 1.1发布。相对于旧版本而言，JDK 1.1的最大亮点是推出Java即时编译器JIT（Just-In-Time），该项技术使得Java程序的执行效率有了很大的提高。

1998年，标志着Java 2平台诞生的JDK 1.2版本发布；该版本引入委托事件处理模型。为了方便市场推广，Sun公司将Java改名为Java 2，同时发布JSP/Servlet、EJB规范，并将Java 2平台划分为J2EE（Java 2 Platform Enterprise Edition，Java企业版）、J2SE（Java 2 Platform Standard Edition，Java标准版）和J2ME（Java 2 Platform Micro Edition，Java微缩版）3大块。这表明Java开始向企业、桌面应用和移动设备应用3大领域挺进。自此，Java才真正成为现代软件开发工具中的利器。

随后，Sun公司于2000年发布JDK 1.3。该版本的主要改进表现在一些类库上（如数学运算、新的Timer API等）的变化、在命名目录接口方面增加了一些域名系统的支持以及增加了Java本地接口的支持，这使得Java可以访问本地资源，支持XML以及使用新的Hotspot虚拟机代替传统的Java虚拟机。并推出Linux和Solaris版的JDK。

2002年JDK 1.4发布。该版本主要对Hotspot虚拟机的锁机制进行了改进，使得JDK 1.4的性能有了质的飞跃。同时增加了断言功能。

2004年发布JDK 5.0。该版本的主题是易用，它不仅增加了诸如泛型、增强型`for`语句、可变参数、Annotation注释、自动拆包和装包等功能，同时推出了EJB3.0规范和Java服务器界面（Java Server Faces，JSF）编程规范。JSF类似于ASP.NET的服务端控件，通过它可以快速地构建复杂的JSP（Java Server Pages）界面。为了暗示该版本较以前版本有较大的改动，Sun公司将版本号1.5改为5.0。同时将J2SE1.5改名为J2SE5.0（Java 2 Platform Standard Edition 5.0）。

2006年发布Java SE 6.0。该版本主要提供了若干实用和方便的功能，如脚本、Web Service、XML、编译器API等。Java SE 6.0是专为Vista而设计的，它在Vista上将会拥有更好的性能。

按照对类库的支持程度来划分，JDK家族已经是拥有Java EE、Java SE、Java ME和Java Card（Java卡）4个各具特色成员的大家庭了，如图1-1所示。如果从纵向的角度来看所有的Java平台的划分，则如图1-2所示。

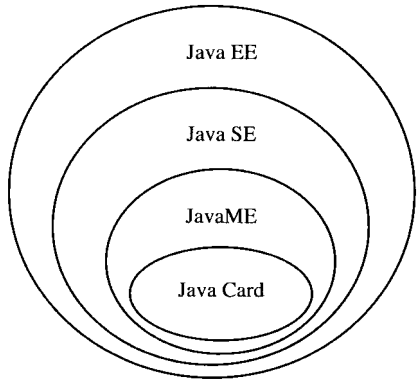


图1-1　Java家族的划分

- Java EE主要用于开发分布式的、基于Web的应用程序。它本身是一个开放的标准，任何软件厂商都可以推出符合其标准的产品。Java EE将逐步发展成为可以与微软的.NET相对抗的网络计算平台。Java EE的类库除了具备Java语言的所有特性外，还包含针对企业计算的各种编程接口和规范，如JavaBean、JSP、RMI（Remote Method Invoke）、EJB（Enterprise Java Beans）、JMS（Java Message Service）、JIDL（Java Interface Definition Language）、JTA（Java Transaction API）等。
- Java SE的主要目的是，为台式机和工作站提供一个开发和运行常见桌面应用程序的平台。其类库包含了Java语言的所有特性，是标准的开发工具包。这也是本教程所要讨论的主要

内容。

- Java ME主要是面向消费电子产品，为消费电子产品提供一个运行Java的平台。它使得Java程序能够在手机、机顶盒、PDA等产品上运行。由于受到设备内存和处理器的限制，其类库比较小，在对Java SE的类库作了一些剪裁的基础上增加了若干新的特性。目前，已有越来越多的手持设备能运行Java ME应用程序，如Java游戏、股票相关程序、记事程序等。
- Java Card主要是针对智能芯片和IC卡等设备而定制的最小Java子集。它只支持核心类库`java.lang.*`及`boolean`和`byte`两种基本数据类型。它对资源需求极小，可运行在不支持图形用户界面和网络的设备上。

从图1-2可以看出，要实现所谓的“一次编写，到处运行”（Write once, run anywhere）的目标，不管是哪种开发平台都必须包括虚拟机、基本的类库及应用程序所需的类库。

图1-2 从纵向的角度来看Java家族的划分

1.2 Java的特点

Sun公司的“Java白皮书”中对Java的定义是：Java是一种简单、面向对象、分布式、解释执行、健壮、安全、结构中立、可移植、高效率、多线程及动态的语言。从此定义可以看出Java的主要特点有如下几条。

(1) 结构简单。Java最初是为消费类电子产品所设计的一种程序语言，因此它必须简单明了。其简单性主要体现在以下两点。

- 尽管Java是在C++的基础上开发的，它继承了C++的许多特性，但同时也放弃掉C++语言中许多繁琐、难以理解和不安全的内容（如指针、运算符重载、多重继承等）。并且通过实现自动垃圾收集，大大简化了程序设计者的内存管理工作。
- JDK提供了丰富的类库，对具有C++编程经验的程序员来讲，无需经过长时间训练即可掌握它。

(2) 面向对象。Java语言是完全面向对象的。它提供了简单的类机制和动态接口模型，使开发者把设计集中于对象和接口。对象中封装了数据成员和成员方法，实现了数据封装和信息隐藏。通过继承机制，子类可以使用父类所提供的方法，实现了代码的复用。为了简单起见，Java只支持类之间的单继承，但支持接口之间的多重继承，支持类与接口之间的实现机制。

(3) 分布式。Java语言支持Internet应用的开发。针对TCP/IP的不同层次，Java提供了3大类库。其中，Java程序可以通过URL打开并访问网络上的对象，其访问方式与访问本地文件系统几乎完全相同。

(4) 健壮性。因为Java最初的设计目的是应用于电子类家庭消费产品，所以它对程序的健壮性要求较高。Java的强类型机制、异常处理、垃圾自动收集等都是Java程序健壮性的重要保证。比如Java的垃圾收集器可自动收集闲置对象占用的内存；Java的异常处理机制能有效地解决程序出现的异常情况；Java通过类型检查、null（空）值检测、数组边界检测等方法，保证程序中有可能出现异常的地方尽量在编译期间被暴露出来。

(5) 安全性。Java主要用于网络环境，为此它采取了一系列的安全机制来保证系统的安全性。比如Java要求每种类型都要严格定义。每个变量、表达式都有类型。Java编译器对所有的数值传递、表达式和参数都要进行类型相容性检查，确保类型兼容。在编译期间，Java编译器并不分配内存，而是推迟到运行时由解释器决定，这样编程人员就无法通过指针访问内存；在运行期间，Java的运行环境提供有类装载器、字节码校验器、运行时内存布局和文件的访问限制等安全保障机制。

(6) 结构中立。Java的设计目标是支持网络应用程序。而网络是由包括各种类型的CPU和操作系统的不同系统构成。为了使Java程序在网络上任何地方都能执行，Java平台将Java源程序编译为结构中立的字节码格式文件（称为.class文件），然后可以在实现这个Java平台的任何系统中运行该字节码格式文件。

(7) 可移植。使用Java开发的程序是可移植的。结构中立是确保程序可移植的必要条件，此外还需很多其他条件的配合。比如Java严格规定了其基本数据类型的长度，Java编译器自身也是用Java语言实现等。

(8) 高效率。与常见的解释执行语言（例如Python、TCL）不同，Java语言既是一种编译型的语言，同时也是一种解释型的语言。编译就是把Java源程序转化为.class文件。解释指的是Java虚拟机通过解释.class文件完成对.class文件的执行。Java的字节码格式是针对机器码的转换设计的，因此，从字节码到机器码的转换非常简单。自动的寄存器分配与编译器对字节码的一些优化可使之生成高质量的代码。Java虚拟机的改进和Java即时编译技术的出现，也使得Java的执行速度有了更大的提高。传统的Java运行环境的执行过程是每收到一条字节码指令后，将其解释成机器码并立刻执行该机器码，然后丢弃掉该机器码。而即时编译器会把常执行的部分先解释好并保存在内存中，当以后遇到该指令时，直接从内存中取出它的机器码并执行，这样就大大提高了Java的运行效率。

(9) 多线程。多线程是指在一个程序中可以同时执行多个简单任务。Java是第一个在语言级提供内置多线程支持的高级语言。它提供了实现多线程的简便方法，并拥有一组高度复杂的同步机制，大大简化了实现多线程的工作。

(10) 动态性。Java语言的设计目标之一是适用于动态变化的环境。这主要表现在：Java程序所需要的类能动态地被载入到运行环境，也可以通过网络来载入它所需要的类；Java中的类有一个运行时刻的表示，通过该表示能进行运行时刻的类型检查等工作；Java类库中可以自由地加入新的成员方法和数据成员，而不会影响用户程序的执行等。

1.3 Java平台

Java不仅是一种面向对象的编程语言，它还是一个开发平台。它给程序员提供了许多工具，比如编译器、解释器、文档生成工具、文件打包工具等。

平台是指程序在其中运行的硬件或软件环境。Java平台与现存的大多数平台的不同之处在于：它是一种能运行在其他硬件平台上的纯软件平台。

Java平台由Java虚拟机、Java应用程序编程接口（Application Programming Interface，API）和各种工具构成，其中Java虚拟机是Java平台的基础。

1. Java虚拟机

Java虚拟机是Java语言底层实现的基础，对Java虚拟机有个大概的了解有助于理解Java语言的一些性质，也有助于更好的使用Java语言。

Java虚拟机是实现Java程序与底层操作系统和硬件无关的关键。不同的操作系统都会有一个针对本操作系统平台的特定Java虚拟机。这样，同一个.class文件便可以在不同的操作系统上执行，Java虚拟机负责屏蔽各平台之间的差异，这正是Java能够做到跨平台的根本所在。

Java虚拟机是一个可由软件模拟也可由硬件来实现的一个想象中的计算机。它拥有自己想象中的硬件，如指令系统、寄存器、存储区、栈等。

Java指令系统是以Java语言的实现为目的设计的，其中包含了用于调用方法和监视多线程的指令。同其他计算机的指令系统一样。Java指令也是由操作码和操作数两部分组成。操作码为8位二进制数，用于指定一条指令操作的性质。操作数紧随操作码之后，其长度根据具体需要而不同。

与微处理器中的专用寄存器类似，Java虚拟机的寄存器用于保存Java虚拟机的运行状态。它们分别是Java程序计数器、指向操作数栈顶端的指针、指向当前执行方法的运行环境的指针、指向当前执行方法的局部变量区中的第一个变量的指针。

Java虚拟机的存储区有常量池和方法区两类。常量池用于存储类名、方法名、变量名以及字符串常量。方法区则用于存储Java方法的字节码。

Java虚拟机的栈用于存放方法的调用状态。当Java虚拟机得到一个Java字节码应用程序后，便为该代码中类的每一个方法创建一个栈，以保存该方法的局部变量、运行环境等状态信息。

Java虚拟机的堆主要用于存储Java类的对象所需的空间。Java解释器负责为类对象在堆内分配空间，同时它还监督对象所占用的内存区域的使用情况。一旦对象使用完毕，便将其回收到堆中。

2. Java API

Java提供了庞大且功能强大的API，如字符串处理、数据输入/输出、网络操作等。人们可以使用这些API作为基础来进行程序开发，而无需重复开发功能相同或类似的类库。事实上，在Java的学习过程中，更多的是要学习如何使用Java API开发程序的技能。

3. 各种工具

这些工具包括：用来编译、运行、监视、调试应用程序以及建立应用程序文档的开发工具，如javac、java、jconsole等；用来创建复杂的图形用户界面的开发包，如Swing等。

1.4　第一个 Java 程序

学习任何一门新的编程语言，传统的做法都是用著名的“Hello, World!”程序作为第一个例子，通过它来讲解利用该语言所编写的程序的基本构成。这里也使用一个具有简单输入功能的“Hello, World!”程序来说明Java应用程序的基本构成，以便让初学者能迅速了解Java应用程序最基本的一些特征。

1.4.1　建立 Java 源程序

要建立一个Java程序，首先要创建Java程序的源代码，即一个由符合Java规范的各种语句构成的文本文件。Java源程序的编写可以使用任何一种文本编辑器，比如UltraEdit、Notepad、Wordpad等，然后只要把编辑好的文件存成扩展名为.java的文件即可。当然了，也可以用集成开发环境（如Eclipse、NetBeans等）编写Java源程序。有关Eclipse、NetBeans及编译运行例1-1的JDK的安装和使用方法请参考附录A。

例1-1是本书的第一个Java应用程序的源程序。打开记事本，在记事本中输入例1-1所示内容，完成后将其保存于目录D:\java_book\chap01下，并命名为HelloWorld.java。该程序的功能是首先在屏幕上显示出“Hello, World!”字符串信息，并等待用户输入，用户输入后在屏幕上显示输入的信息。

例1-1　第一个Java应用程序

```
import java.util.Scanner;
/*
本程序功能是：首先输出"Hello, World!"，然后等待用户输入，
并显示用户输入的信息。
*/
public class HelloWorld {
    /**
     * 方法main()是Java应用序的唯一入口
     * @param args 输入参数
     * @exception 没有异常抛出
     */
    public static void main( String[] args ) {
        System.out.println( "Hello, World!" );  //输出"Hello, World!"
        /*printf和scanner是J2SE5.0的新功能*/
        Scanner scanner = new Scanner(System.in);
        System.out.print("请输入您的名字：");
        System.out.printf("%s%s 这是您的第%d个Java程序！\n", "您好！",scanner.next(),1);
    }
}
```

在编写程序时需注意区分大小写，Java语言是严格区分大小写的，即同一字母的大小写具有不同的含义。

1.4.2　编译和运行 Java 程序

1. 编译源程序

在Windows XP的开始菜单中执行“运行”命令，在如图1-3所示的“运行”对话框中键入“cmd”

命令，点“确定”可进入命令行界面。

使用“cd”命令，进入待编译文件所在目录D:\java_book\chap01。在DOS命令提示符下输入如下命令，并按回车键。

```
D:\java_book\chap01> javac HelloWorld.java
```

如果命令提示符窗口没有出现任何提示信息，则说明Java源程序已经编译成功，并在当前目录D:\java_book\chap01下产生一个扩展名为.class的文件，该文件为Java源程序编译后所产生的HelloWorld.class文件。由于.class文件不是可以直接运行的机器代码，因此不能直接在操作系统环境下执行它。但是它可以在Java虚拟机内执行。

上面所用的命令javac是Sun公司发布的JDK中所提供的Java编译器的名字，它的作用是将Java源程序编译成Java虚拟机能够解释执行的.class文件。

2. 执行程序

在DOS命令提示符下输入如下命令，并按回车键。

```
D:\java_book\chap01>java HelloWorld
```

如果程序没有错误的话，屏幕上则显示如下信息。

```
Hello, World!
请输入您的名字：XBM
您好！XBM 这是您的第1个Java程序！
```

其中，“请输入您的名字：”后面的字符XBM是用户从键盘输入的内容。如图1-4所示。

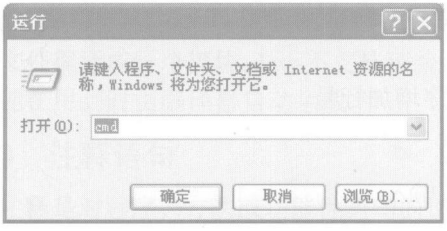


图1-3 “运行”命令对话框

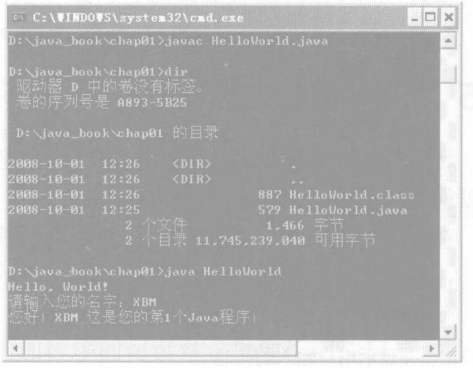


图1-4 在命令行界面中编译和执行Java程序

命令java是Sun公司发布的JDK中所提供的Java解释器的名字，其作用是调用Java虚拟机将.class文件解释成它所在计算机能识别的二进制文件并执行。

1.4.3 常见问题及解决方法

编译和运行Java程序有以下几个常见问题。

(1) javac不是有效内部命令。主要原因是：没有安装JDK，或安装了JDK但环境变量Path的路径设置不对。