



普通高等教育“十三五”规划教材
计算机专业系列



Java面向对象 程序设计

主编 程细柱 戴经国



电子科技大学出版社

图书在版编目 (CIP) 数据

Java 面向对象程序设计 / 程细柱, 戴经国主编. —
成都: 电子科技大学出版社, 2016. 11
ISBN 978 - 7 - 5647 - 4063 - 4

I. ①J… II. ①程… ②戴… III. ①JAVA 语言 - 程序
设计 IV. ①TP312.8

中国版本图书馆 CIP 数据核字 (2016) 第 297908 号

内 容 简 介

本书从面向对象的思想出发系统地介绍 Java 程序设计的基础知识、开发环境与开发技术。全书共分 11 章, 内容包括 Java 发展史与开发环境、Java 的基本语法、Java 类和对象与封装性、Java 继承性与多态性、异常处理机制、数组与字符串与集合类、多线程机制、Swing 容器与布局管理器、事件处理与常用组件、IO 流与文件操作、Java 网络编程。本书所有知识都结合具体实例进行介绍, 力求详略得当。各实例代码经过精心调试、编译和运行。每章后附上上机实验指导和相关习题, 供读者实践练习。另外还提供了课程资源包, 资源包中提供有本书所有实例、上机指导、相关案例的源代码, 电子课件 PPT 等内容, 资源包可在 csdn 下载。

Java 面向对象程序设计

程细柱 戴经国 主编

出 版: 电子科技大学出版社 (成都市一环路东一段 159 号电子信息产业大厦 邮编: 610051)

策划编辑: 张 鹏

责任编辑: 张 鹏

主 页: www.uestcp.com.cn

电子邮箱: uestcp@uestcp.com.cn

发 行: 新华书店经销

印 刷: 北京市彩虹印刷有限责任公司

成品尺寸: 185mm × 260mm 印张: 17.5 字数: 393 千字

版 次: 2016 年 11 月第 1 版

印 次: 2016 年 11 月第 1 次印刷

书 号: ISBN 978 - 7 - 5647 - 4063 - 4

定 价: 36.80 元

■ 版权所有 侵权必究 ■

◆ 本社发行部电话: 028 - 83202463; 销售电话: 010 - 61705319

◆ 本书如有缺页、破损、装订错误, 请寄回印刷厂调换。

前言

随着网络技术的发展，Java 已是业界最流行的编程语言之一。由于其具有平台无关性、可移植性、面向对象、运行稳定、安全性好、支持多线程、支持数据库编程、支持网络编程等许多优良特性，所以成为目前使用最为广泛的编程语言。其作为面向对象语言已经受到越来越多企业的青睐，作为一种网络语言，其自身也发展迅速。

现在许多大学已经将《Java 面向对象程序设计》作为计算机科学技术专业、软件工程专业、网络技术专业、通信工程专业的一门专业基础课。本书正是为了培养以上人才而编写的，全书共分 11 章，采用“任务驱动”教学方法，重视编程训练，做到理论与实践相结合，每章包括：教学目标、重点内容、基本原理、编程实例、归纳总结、实验指导、单元习题等七方面的内容。各章节的内容如下：

第 1 章 Java 发展史与开发环境。主要介绍 Java 语言的发展史、Java 语言的特点、Java 运行环境的安装与配置、Java 项目的打包与发布。

第 2 章 Java 的基本语法。主要介绍 Java 符号集、基本数据类型、运算符与表达式、流程控制语句。

第 3 章 Java 类、对象与封装性。主要介绍面向对象的基本思想、Java 类的定义、数据成员和成员方法、对象的声明与创建、Java 的封装性。

第 4 章 Java 继承性与多态性。主要介绍类的继承性、类的多态性、终极类、抽象类、接口、Java 包。

第 5 章 Java 异常处理机制。主要介绍异常的概念、异常的层次结构、异常处理机制、用户自定义异常。

第 6 章 数组、字符串与集合类。主要介绍一维数组和多维数组、字符串、Java 集合框架、Collection 接口、List 接口、Set 接口、Map 接口。

第 7 章 多线程机制。主要介绍进程和线程的概念、Thread 类、Runnable 接口、线程的创建方法、线程的状态、线程的调度与同步。

第 8 章 Swing 容器与布局管理器。主要介绍 AWT 和 Swing 的区别、Swing 的常见容器、Java 的布局管理器。

第 9 章 事件处理与常用组件。主要介绍 Java 的事件处理机制、常见的事件类和事件监听接口、适配器、Swing 常用组件。

第 10 章 IO 流与文件操作。主要介绍数据流的概念、常见的字节流与字符流、文件对象、对象的串行化。

第 11 章 Java 网络编程。主要介绍网络基础知识、网络地址相关类、TCP 网络编程、UDP 网络编程。

本书由浅入深地讲解了 Java 语言的编程知识，重点突出、案例充分，适合作为各大专院校、职业院校和培训学校的相关专业的教材，也适合作为软件开发入门者的自学用书和参考书。

通过对本教材的认真学习后，学生应该能够了解 Java 语言的主要特性、掌握 Java 语言的运行环境、理解面向对象的编程技术、学会使用异常处理机制、明白数组与字符串及集合类的使用、掌握输入/输出流及文件操作、理解事件处理机制、掌握图形用户界面的设计、学会开发多线程应用程序、理解 TCP/IP 及 UDP 协议、掌握 Java 网络编程。

本书除了第 6 章是由周月鹏老师编写，其余 1 至 5 章和 7 至 11 章共十章全部由本人一人编写。另外在编写过程中，戴经国教授主审了初稿，刘霖老师也提供了很好的建议，在此表示感谢！虽然在编写本书的过程中倾注了大量心血，但难免存在疏漏和错误之处，恳请广大专家与读者批评指正，本人不胜感谢，本人的 email 是：cxz973@qq.com。

本书免费提供的电子教案和源代码等相关教学资源可在网址 <http://download.csdn.net/user/cflynn> 处下载。

程细柱
2016 年 10 月

目 录

第 1 章 Java 发展史与开发环境	1
1.1 Java 语言的产生与发展	1
1.2 Java 语言的特点	2
1.3 Java 语言实现的机制	4
1.4 Java 运行环境的安装与配置	5
1.5 第一个 Java 应用程序	7
1.6 开发工具 Eclipse	9
1.7 Java 项目的打包与发布	13
实验 1 Java 运行环境的安装与测试	16
习 题	16
第 2 章 Java 的基本语法	21
2.1 Java 符号集	21
2.2 Java 的基本数据类型	23
2.3 运算符与表达式	30
2.4 流程控制语句	35
实验 2 Java 基本语句练习	44
习 题	44
第 3 章 Java 类、对象与封装性	54
3.1 面向对象基本思想	54
3.2 Java 类与对象	56
3.3 数据成员	62
3.4 成员方法	63



3.5	static 关键字	67
3.6	Java 的封装性	70
	实验3 Java 类、对象与封装性	72
	习 题	73
第4章	Java 继承性与多态性	80
4.1	类的继承性	80
4.2	类的多态性	82
4.3	final 关键字	85
4.4	抽象类与接口	87
4.5	Java 包	91
	实验4 Java 继承性与多态性	93
	习 题	94
第5章	Java 异常处理机制	102
5.1	异常概念	102
5.2	异常的层次结构	103
5.3	异常处理	106
5.4	用户自定义异常	112
	实验5 Java 的异常处理	113
	习 题	114
第6章	数组、字符串与集合类	117
6.1	数组	117
6.2	字符串	123
6.3	Java 集合框架	128
6.4	Collection 接口	129
6.5	List 接口	131
6.6	Set 接口	134
6.7	Map 接口	135
	实验6 数组、字符串和集合类的应用	137

习 题	137
第 7 章 多线程机制	138
7.1 多线程概述	138
7.2 线程的创建	139
7.3 线程的状态	142
7.4 线程的调度	145
7.5 线程的同步	147
实验 7 多线程的编程	149
习 题	149
第 8 章 Swing 容器与布局管理器	154
8.1 AWT 和 Swing 的区别	154
8.2 GUI 设计概述	155
8.3 Swing 容器	156
8.4 布局管理器	163
实验 8 图形用户界面设计	177
习 题	177
第 9 章 事件处理机制与常用组件	182
9.1 事件处理机制	182
9.2 事件类	187
9.3 事件监听接口	188
9.4 适配器	191
9.5 Swing 常用组件	192
实验 9 Java 常用组件与事件处理	221
习 题	222
第 10 章 IO 流与文件操作	227
10.1 数据流的概念	227
10.2 字节流	228
10.3 字符流	233



10.4 文件对象	236
10.5 对象的串行化 (Serialization)	241
实验 10 输入/输出流与文件存取	242
习 题	242
第 11 章 Java 网络编程	251
11.1 网络基础知识	251
11.2 网络地址相关类	254
11.3 TCP 网络编程	258
11.4 UDP 网络编程	266
实验 11 Java 网络编程	271
习 题	271

第 1 章 Java 发展史与开发环境

本章教学目标

- ◆ 掌握 Java 开发环境的安装与配置；
- ◆ 熟悉 Java 项目的开发过程；
- ◆ 了解 Java 语言的特点与发展史。

本章重点内容

- ◆ Java 语言的特点与工作原理；
- ◆ Java 语言实现的机制；
- ◆ 运行环境的安装与配置；
- ◆ Java 项目的设计、调试、打包与发布。

1.1 Java 语言的产生与发展

Java 诞生于 Sun 公司的一个叫做 Green 的项目。1991 年初，美国加州的 Sun Microsystem 公司（以下简称 Sun 公司）成立了一个以 James Gosling 为首的、名为 Green 的项目研发小组，其目标是试图开发一个面向家用电器市场的软件产品。项目研发小组首先从改写 C++ 语言的编译器着手。但是，他们很快便意识到：在家用电器中使用的处理器没有统一的标准，消费电子产品种类繁多，即使是同一类消费电子产品所采用的处理芯片和操作系统也不相同。为了简化开发工作，他们需要设计一个与平台无关的开发环境，而 C++ 太复杂，且安全性差，显然无法胜任，因此 Green 项目开发组的成员决定自行开发一种精巧而且安全的新的语言，并将该语言命名为 Oak（橡树），传说将其命名为 Oak 是因为在他们设计该语言的结构时，窗外的一棵橡树（Oak）映入了他们的眼帘。但后来又不得不放弃这个名字，因为较早的一种语言也用了这个名字，有一次研发小组的成员在公司附近的一家咖啡厅喝咖啡时产生灵感，因为咖啡的原产地是 Java，所以开发小组为这个新语言取了一个新名称——Java（爪哇），直到现在“一杯飘香的咖啡”仍然作为该语言的标志，目前 Java 这杯咖啡已经飘香在世界各地。

尽管这个原型非常成功，但在竞争激烈的家用电器市场上却败给了竞争对手，因为

Java 被束之高阁。直到 1993 年 7 月，伊利诺斯大学的 NCSA 推出了一个在 Internet 上广为流行的 WWW 浏览器 Mosaic1.0 版。然而，这时的 WWW 页面虽然内容丰富，可以实现声、图、文并茂，但它却是静态的，若想增强 WWW 的动感，需要通过一种机制来使它具有动态性。其解决方案显然是嵌入一种既安全可靠，又非常简练的语言，Oak 完全满足这一要求。1994 年，Sun 公司的创始人之一 Bill Joy 的介入，使 Oak 成为 Java 而得以走红。1995 年，美国 Sun Microsystems 公司正式向 IT 业界推出了 Java 语言，该语言具有安全、跨平台、面向对象、简单、适用于网络等显著特点，当时以 web 为主要形式的互联网正在迅猛发展，Java 语言的出现迅速引起所有程序员和软件公司的极大关注，程序员们纷纷尝试用 Java 语言编写网络应用程序，并利用网络把程序发布到世界各地进行运行。IBM、Oracle、微软、Netscape、Apple、SGI 等大公司纷纷与 Sun Microsystems 公司签订合同，获得授权使用 Java 平台技术。2004 年 9 月 30 日，J2SE1.5 发布，成为 Java 语言发展史的一个里程碑，为了表示该版本的重要性 J2SE1.5 更名为 Java SE5.0。2005 年 6 月，JavaOne 大会召开，Sun 公司发布了 Java SE6。2009 年 4 月 20 日，甲骨文（Oracle）公司以 74 亿美元的价格收购 Sun 公司，随后 Java 社区的运营也交由甲骨文公司管理，2011 年 7 月，甲骨文发布 Java SE7，2014 年 3 月，甲骨文发布 Java SE8。

1.2 Java 语言的特点

Java 到底有哪些主要特点呢？

1. 面向对象

面向对象其实是现实世界模型的自然延伸，现实世界中任何实体都可以看做是对象。对象包含属性和方法，对象的说明用属性表达，而通过使用方法来操作这个对象。如果说传统的过程式编程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。用公式表示，过程式编程语言为：程序 = 算法 + 数据；面向对象编程语言为：程序 = 对象 + 消息。Java 是一种纯面向对象的语言，支持封装性、继承性和多态性等三个特点，在 Java 中无全程变量，Java 中绝大部分成员是对象，只有简单数字类型、字符类型和布尔类型除外。但对于这些类型，Java 也提供了相应的对象类型以便与其他对象交互操作。

2. 跨平台性

跨平台是指软件可以不受计算机硬件和操作系统的约束而在任意计算机环境下正常运行。这是软件发展的趋势和编程人员追求的目标。之所以这样说，是因为计算机硬件的种类繁多，操作系统也各不相同，不同的用户和公司有自己不同的计算机环境偏好，而软件为了能在这些不同的环境里正常运行，就需要独立于这些平台。Java 语言自带的虚拟机很好地实现了跨平台性，它提供了一个字节码到底层硬件平台及操作系统的屏障，使得 Java 源代码经过编译后生成二进制的字节码是与平台无关的，但可被 Java 虚拟机解释执行。

3. 可移植性

由于 Java 的平台无关性使得 Java 程序经过一次编译后可以移植到别的任意平台上解

释执行。Java 编译器编译生成的字节码文件与体系结构无关，任何种类的计算机只要安装了 Java 虚拟机就可以运行字节码文件。程序员不需要编写各种版本的程序以在不同的处理器上运行，减少了程序员的负担，提高了开发效率。

4. 安全性

安全性可以分为四个层面，即语言级安全性、编译时安全性、运行时安全性、可执行代码安全性。语言级安全性指 Java 的数据结构是完整的对象，这些封装过的数据类型具有安全性。编译时要进行 Java 语言和语义的检查，保证每个变量对应一个相应的值，编译后生成 Java 类。运行时 Java 类需要类加载器载入，并经由字节码校验器校验之后才可以运行。Java 类在网络上使用时，对它的权限进行了设置，保证了被访问用户的安全性。在 Java 程序中没有指针、没有多重继承机制、没有全局变量、没有 `#include` 和 `#define` 等预处理器、不能采用地址计算的方法通过指针访问内存单元，这些大大减少了错误发生的可能性。Java 的数组也并非用指针实现，这样就可以在检查中避免数组越界的发生。另外，Java 提供内存自动回收机制，使得程序员不必费心管理内存，使程序设计更加简单，同时大大减少了出错的可能，这也增加了 Java 的鲁棒性。

5. 并发性

Java 内置了多线程技术，定义了一些类和方法等来建立和管理用户定义的多线程。多线程特别有利于在程序中实现并发任务，Java 提供线程类 `Thread` 实现了多线程的并发机制。然而，程序的并发执行必定会出现多个线程互斥访问临界资源的局面，因而并发系统解决的关键就是对临界资源的管理和分配问题，而在进行临界资源分配时有两方面需要考虑，即安全性和公平性，Java 提供了相关的解决方法。

6. 分布式

Java 在构建一个分布式系统平台有如下优势：

- 1) 核心库中有网络特性包，支持 TCP/IP UDP 等，非堵塞的 IO 等等；
- 2) 核心库中支持丰富的数据结构，一个分布式系统需要很多不同类型的数据结构，而 Java 有各种集合支持；
- 3) 可在分布式系统中完成一致的吞吐量，Java 支持多线程方面是强大的；
- 4) 很早支持对象序列化到字节，在一个分布式系统发送复杂数据是一件方便的事情；
- 5) Java 有很多 API，非常广泛；
- 6) Java 的性能是难以置信的，多线程、垃圾回收、主流网络 IO、并发磁盘 IO、各种弱引用等等。

7. 简单易用

Java 的简单性首先体现在精简的系统上，力图用最小的系统实现足够多的功能；对硬件的要求不高，在小型的计算机上也可以良好的运行。Java 语言采用了 C 语言中的大部分语法，熟悉 C 语言的程序员会发现 Java 语言在语法上与 C 语言极其相似。另外，Java 源代码的书写不拘泥于特定的环境，可以用记事本、文本编辑器等编辑软件来实现，然后将源文件进行编译，编译通过后可直接运行，通过调试则可得到想要的结果。



1.3 Java 语言实现的机制

1.3.1 Java 虚拟机

Java 语言的执行模式是半编译和半解释型。Java 编写好的程序首先由编译器转换为标准字节代码，然后由 Java 虚拟机去解释执行。字节代码也是一种二进制文件，但不能直接在操作系统上运行，它可看做虚拟机的机器码。虚拟机把字节代码程序与各操作系统和硬件分开，使 Java 程序独立于平台。虚拟机可以用软件实现，也可用硬件实现，但在无线技术中，都用硬件实现。

虚拟机（VM）的执行过程有 3 个特点：

- 1) 多线程；
- 2) 动态连接；
- 3) 异常处理。

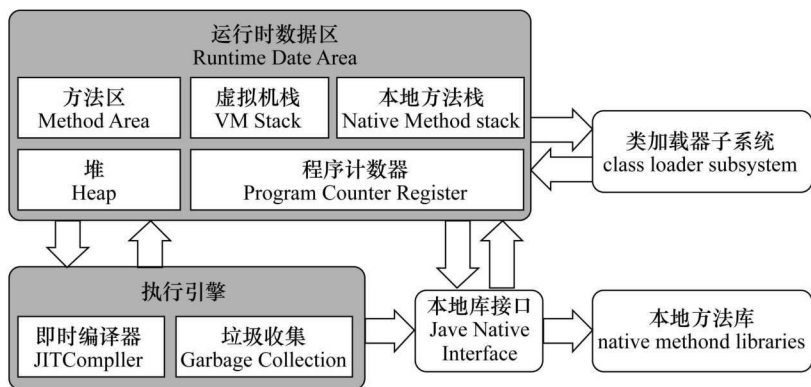


图 1-1 Java 虚拟机机理

1.3.2 垃圾回收机制

在 C++ 语言中程序员需要在编写程序的时候把不再使用的对象内存释放掉，如果不及及时进行无用内存的回收，就会导致内存泄漏，进而导致系统崩溃。但是这种人为的管理内存释放的方法却往往由于程序员的疏忽而致使内存无法回收，同时也增加了程序员的工作量。

而在 Java 语言中，所有事物都封装在类中，需要时创建类的实例（对象）来处理，这种动态的实例都存储在内存堆中。为了充分利用资源，Java 有一个系统级的线程，对内存的使用进行跟踪，该线程可以在系统空闲时对不用的内存进行回收，使程序员从繁忙的内存管理中解放出来。

1.3.3 代码安全检测

Java 程序的安全性体现在多个层次上：

- 1) 在编译层, 有语法检查;
- 2) 在平台层, 通过配置策略, 可设定访问资源域, 而无需区分本地或远程;
- 3) 在字节码的解释执行层, 也需要经过三个步骤, 首先由类装载机 (class loader) 负责把类文件 (.class 文件) 加载到 Java 虚拟机中, 在此过程需要检验该类文件是否符合类文件规范; 其次字节码校验器 (bytecode verifier) 检查该类文件的代码中是否存在着某些非法操作, 例如 applet 程序中存在针对本机文件系统的操作; 如果字节码校验器检验通过, 由 Java 解释器负责把该类文件解释成为机器码进行执行。Java 虚拟机采用的是“沙箱”运行模式, 即把 Java 程序的代码和数据都限制在一定内存空间里执行, 不允许程序访问该内存空间外的内存, 如果是 applet 程序, 还不允许访问客户端机器的文件系统。

1.4 Java 运行环境的安装与配置

1.4.1 什么是 Java JDK?

Sun 公司为所有的 Java 程序员提供了一套免费的 Java 开发和运行环境, 取名为 JDK (Java Development Kit)。自从 Java 推出以来, JDK 已经成为使用最广泛的 Java SDK。JDK 是整个 Java 的核心, 包括了 Java 运行环境、Java 工具和 Java 基础类库。JDK 是学好 Java 的第一步。从 SUN 的 JDK5.0 开始, 提供了泛型等非常实用的功能, 其版本也不断更新, 运行效率得到了非常大的提高。

1. JDK 包含的基本组件有

Javac: 编译器, 将源程序转成字节码;

Java: 运行编译后的 java 程序 (.class 后缀的);

appletviewer: 小程序浏览器, 一种执行 HTML 文件上的 Java 小程序的 Java 浏览器;

Jar: 打包工具, 将相关的类文件打包成一个文件;

Javadoc: 文档生成器, 从源码注释中提取文档;

Jdb: debugger, 查错工具;

Javah: 产生可以调用 Java 过程的 C 过程, 或建立能被 Java 程序调用的 C 过程的头文件;

Javap: Java 反汇编器, 显示编译类文件中的可访问功能和数据, 同时显示字节代码含义;

Jconsole: Java 进行系统调试和监控的工具。

2. Java 常用的包

java.lang: 这个是系统的基础类, 比如 String 等都是这里面的, 这个包是唯一一个可以不用引入 (import) 就可以使用的包;

java.io: 这里面是所有输入输出有关的类, 比如文件操作等;

java.nio: 为了完善 io 包中的功能, 提高 io 包中性能而写的一个新包, 例如 NIO 非堵塞应用;

Java.net: 这里面是与网络有关的类, 比如 URL, URLConnection 等;

java.util: 这个是系统辅助类, 特别是集合类 Collection, List, Map 等;
java.sql: 这个是数据库操作的类, Connection, Statement, ResultSet 等;
javax.servlet: 这个是 JSP, Servlet 等使用到的类。

1.4.2 建立 Java 开发环境

1. JDK 的下载与安装

Sun 公司为 Java 程序员提供的免费 Java 开发和运行环境可以从 <http://www.oracle.com> 上进行下载。现在最新的 Windows-平台的有 jdk-8u77-windows-i586.exe, 安装过程比较简单, 下载后双击该文件按向导进行安装就可以了。安装的时候可以选择安装到任意的硬盘驱动器上, 例如安装到 C:\Program Files\Java\jdk1.8.0_40\ 目录下。通常在 JDK 目录下有 bin、jre、lib、include、demo 等子目录, 其中 bin 目录保存了 Javac、Java、appletviewer 等命令文件, jre 保存的是 Java 的运行环境, lib 目录保存了 Java 的类库文件, include 目录保存了本地方法, demo 目录保存了许多 Java 的例子。

2. 环境配置

在安装完 JDK 之后, 必须配置类路径 classpath 和环境变量 path, JDK 才能够正常运行。

对于 Win2000、WinXP 系统, 使用鼠标右击“我的电脑”→“属性”→“高级”→“环境变量”, 对于 Vista、Win7、Win8 系统, 使用鼠标右击“计算机”→“属性”→左侧“高级系统设置”→“高级”→“环境变量”打开对话框(图 1-2)。



图 1-2 环境变量设置对话框

在“环境变量”的“系统变量”下面选择“新建”后，打开对话窗口，在变量名中输入 JAVA_ HOME，在变量值中输入 C: \ Program Files \ Java \ jdk1.8.0_73，如图 1-3。

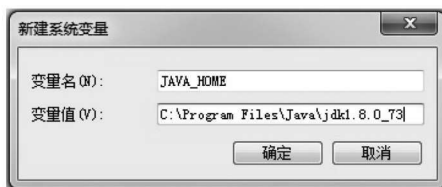


图 1-3 classpath 设置对话框

再次新建系统变量，在变量名中输入 CLASSPATH，在变量值中输入 % JAVA_ HOME% \ lib \ dt. jar; % JAVA_ HOME% \ lib \ tools. jar，最后在系统变量列表中单击“PATH”→“编辑”，如图 1-4 所示。值域最后面输入：

; % JAVA_ HOME% \ bin; % JAVA_ HOME% \ jre \ bin;

注意变量值部分不能出现空格。

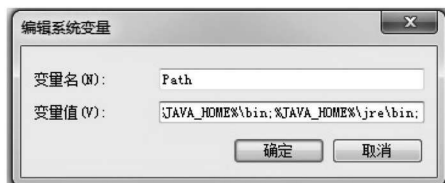


图 1-4 PATH 设置对话框

配置完成后可在 DOS 环境下输入 java -version 命令测试 JVM 是否安装配置正确，如图 1-5 显示的结果说明配置成功。



图 1-5 测试 JVM

1.5 第一个 Java 应用程序

1. 编写 Java 源程序

我们用记事本来编辑一个 HelloWorld 程序作为本书编程的开始，然后把编辑好的文件存成纯文本的 HelloWorld. Java 文件。当然也可以用一些集成开发环境，例如 Borland 公司

的 JBuilder, IBM 公司的 Eclipse, Sun 公司的 NetBeans 等, 在后面章节将介绍其中一种。

【例 1-1】 HelloWorld. Java

```
public class HelloWorld
{
    public static void main(String args[])
    {
        System.out.println("大家好,这是第一个 Java 程序!");
    }
}
```

例 1-1 是一个简单的 Java 程序, 现简单分析其结构。

- class 为关键字, 定义一个类。
- public 是访问控制修饰符, 表示一个公有类, 其他所有的类也可以访问这个类的对象。
- HelloWorld 为类的名称, 由程序员自己定义。
- 类名之后要写一对大括号, 类中的数据成员 (变量)、方法成员放入其中。
- public static void main() 用于定义一个方法 main(), 它是 Java 程序的执行入口, 它也是以 “{}” 开始和结束。main 方法的参数是一个字符串数组 args, 虽然在本程序中没有用到, 但是必须列出来。
- System.out.println (“大家好, 这是第一个 Java 程序!”) 是 main 方法中的一条语句, 它是调用 System 类的 out 对象的 println() 方法, 向标准输出 (屏幕) 送一字符串信息并回车换行。

2. 编译 Java 源程序

安装了 JDK 程序开发包后, 就可以用其中的 “Javac.exe” 命令编译 HelloWorld. Java 源程序了。方法是单击【开始】菜单, 在【运行】窗口输入 “cmd” 按【确定】按钮进入 DOS 状态, 然后在 “命令提示符” 下通过 cd 命令找到 Java 源文件所在的目录, 在命令行中输入 Javac HelloWorld. Java 命令编译, 编译成功将生成 HelloWorld. class 类文件, 如图 1-6 所示:

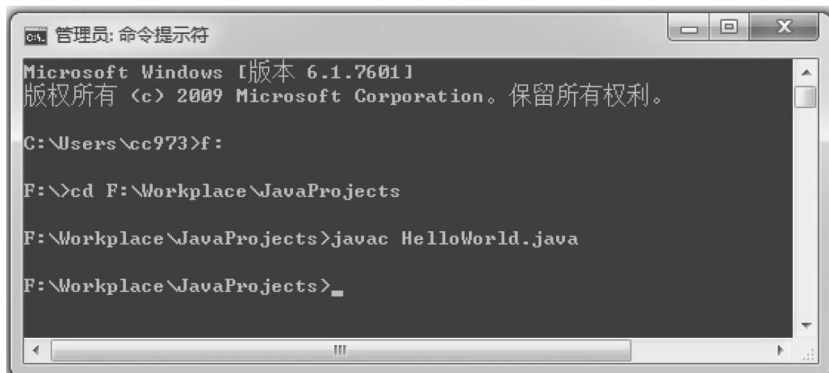


图 1-6 编译结果

3. 运行 Java 程序

当编译结束以后，Java 源文件中的每一个类都会生成相应的 .class 文件，例如上图中生成了一个 HelloWorld.class 文件，现在就可以在“命令提示符”下输入：Java HelloWorld 进行执行，该程序的运行结果是直接在命令行下输出“大家好，这是第一个 Java 程序！”，如图 1-7 所示：

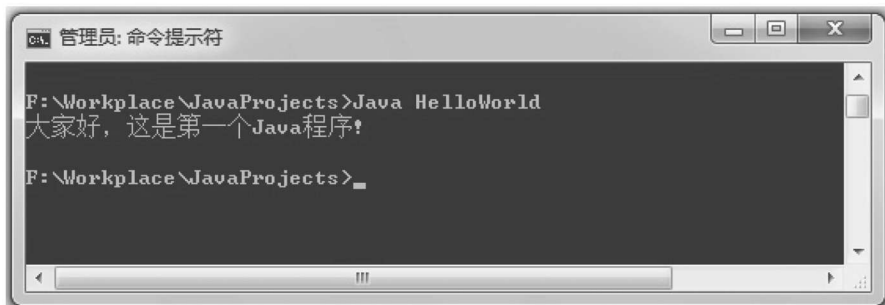


图 1-7 运行结果

几点注意事项：

- 1) Java 源文件的后缀是 .java，其文件名应与包含 main() 方法的 public 类名完全一致，且区分字母的大小写。
- 2) 运行时输入“Java 文件名”即可，不能输入 .class 后缀。

1.6 开发工具 Eclipse

Eclipse 是著名的跨平台的自由集成开发环境（IDE）。最初主要用来 Java 语言开发，通过安装不同的插件 Eclipse 可以支持不同的计算机语言，比如 C++ 和 Python 等开发工具。Eclipse 的本身只是一个框架平台，但是众多插件的支持使得 Eclipse 拥有其他功能相对固定的 IDE 软件很难具有的灵活性。许多软件开发商以 Eclipse 为框架开发自己的 IDE。

Eclipse 主要由 Eclipse 项目、Eclipse 工具项目和 Eclipse 技术项目三个项目组成，具体包括四个部分组成——Eclipse Platform、JDT、CDT 和 PDE。JDT 支持 Java 开发、CDT 支持 C 开发、PDE 用来支持插件开发，Eclipse Platform 则是一个开放的可扩展 IDE，提供了一个通用的开发平台。它提供建造块和构造并运行集成软件开发工具的基础。Eclipse Platform 允许工具建造者独立开发与他人工具无缝集成的工具从而无须分辨一个工具功能在哪里结束，而另一个工具功能在哪里开始。

Eclipse 最初由 OTI 和 IBM 两家公司的 IDE 产品开发组创建，起始于 1999 年 4 月。IBM 提供了最初的 Eclipse 代码基础，包括 Platform、JDT 和 PDE。Eclipse 项目 IBM 发起，围绕着 Eclipse 项目已经发展成为了一个庞大的 Eclipse 联盟，有 150 多家软件公司参与到 Eclipse 项目中，其中包括 Borland、Rational Software、Red Hat 及 Sybase 等。Eclipse 是一个