

实例讲解
实训强化
培养技能
面向就业

全国高等职业教育计算机类规划教材·实例与实训教程系列

Java编程技术基础

◎ 刘勇军 孙璐 主编

◎ 陈虹君 罗国涛 吴雪琴 副主编



- ◆ 最新开发环境：采用最新的、主流的开发环境JDK6+MyEclipse7
- ◆ 讲解简洁精练：知识分布合理，讲解由浅入深，做到实用、够用
- ◆ 案例丰富实用：案例丰富、实用，体现形式多样，注重对知识的综合运用
- ◆ 实训练题精选：实训和习题丰富，紧扣知识点，做到对知识的及时巩固，注重实践



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY <http://www.phei.com.cn>

全国高等职业教育计算机类规划教材·实例与实训教程系列

Java 编程技术基础

刘勇军 孙璐 主编

陈虹君 罗国涛 吴雪琴 副主编

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书以工程为导向,强化实训和案例教学。通过案例与工程的训练,加深对理论知识的理解,掌握这些知识后能够开发完整的 Java 项目。

全书共 13 章,内容包括 Java 开发环境介绍、Java 语言基础、面向对象基础、数组、常用类与集合、图形用户界面、异常处理、输入/输出流、线程、数据库应用、Applet 与 Java 网络编程等。本书结构清晰,知识点分布合理。每章都有与本章知识点结合紧密的案例以及相应实训操作和习题。本书案例非常丰富,体现的形式多样。在各章后有对本章知识应用的小案例。全书提供 3 个完整的项目:贪吃蛇游戏、学生成绩管理系统和公共聊天室程序;其中,贪吃蛇游戏项目按知识点体现在相应章节后面,并对知识的应用思路有较详细的介绍。

本书可作为大专院校计算机专业的教材,也适合作为 Java 培训教材。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有,侵权必究。

图书在版编目(CIP)数据

Java 编程技术基础 / 刘勇军, 孙璐主编. —北京: 电子工业出版社, 2012.1

全国高等职业教育计算机类规划教材·实例与实训教程系列

ISBN 978-7-121-15180-4

I. ①J… II. ①刘… ②孙… III. ①JAVA 语言—程序设计—高等职业教育—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字(2011)第 238554 号

策划编辑: 程超群

责任编辑: 程超群 文字编辑: 王艳萍

印 刷: 三河市鑫金马印装有限公司

装 订:

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×1 092 1/16 印张: 20.5 字数: 523.2 千字

印 次: 2012 年 1 月第 1 次印刷

印 数: 3 000 册 定价: 36.00 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zlts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

前 言

Java 语言以其面向对象、平台无关性、安全性等特征而得到越来越多人的认可和使用。目前，Java 作为一种非常流行的编程语言，在各高校和培训机构都有开设。出版的教材也比较多，但大部分理论性较强，学生在学习过程中感到枯燥、困难，因此我们编写了本书。

全书共 13 章，小案例比较多，综合性较强的案例有 3 个，每章后有相应的案例、实训和习题。本书注重编程能力，即动手能力的培养，同时也强调对章节知识的及时巩固。

第 1 章介绍 Java 的集成开发环境：JDK6 与 MyEclipse7。第 2 章介绍贪吃蛇游戏项目的整体情况，而具体开发步骤将通过学习各章相应知识点后进行具体应用的方式来体现。第 3 章介绍 Java 语言的语法和语句等基础知识。第 4 章介绍面向对象的知识：类、对象、继承、抽象、接口等。第 5 章介绍数组和字符串类、日期类、随机数类等常用类，还介绍了集合。第 6 章介绍异常的处理。第 7 章介绍输入/输出流。第 8 章介绍 Applet 的图像类和画图等。第 9 章介绍 AWT、Swing 等图像组件以及事件的处理等。第 10 章介绍线程。第 11 章介绍网络编程。第 12 章介绍 Java 连接数据库的知识，同时在本章后有一个综合性较强的数据库连接项目：学生成绩管理系统。第 13 章介绍公共聊天室程序项目的开发过程。

另外，本书的 3 个综合项目侧重各不同：贪吃蛇游戏项目侧重 Java 游戏的开发，学生成绩管理系统侧重 JDBC 编程应用，公共聊天室程序项目侧重基于 Socket 的网络编程应用。这 3 个项目做到了对本书知识的全覆盖。

本书由刘勇军、孙璐主编，陈虹君、罗国涛、吴雪琴副主编。其中刘勇军编写了第 11 章和第 13 章，孙璐编写了第 4 章、第 6 章、第 12 章，陈虹君编写了第 2 章、第 7 章、第 8 章、第 10 章，罗国涛编写了第 1 章、第 5 章、第 9 章，吴雪琴编写了第 3 章。各章后的贪吃蛇项目程序由陈虹君编写。

本书适合高职高专院校、Java 培训机构使用。

由于编者水平有限，错误和不足之处在所难免，恳请广大读者和同仁批评指正。

编 者

目 录

第 1 章 Java 开发环境	(1)
1.1 Java 语言简介	(1)
1.2 Java 程序举例	(3)
1.2.1 Java 应用程序	(3)
1.2.2 Java 小应用程序 (Applet)	(4)
1.3 Java 开发环境的搭建	(5)
1.3.1 JDK 的下载与安装	(5)
1.3.2 环境变量的配置	(6)
1.3.3 Java 程序的编译与运行	(7)
1.4 MyEclipse 集成开发环境	(8)
1.4.1 MyEclipse 7.0 环境配置	(8)
1.4.2 MyEclipse 下开发 Java 工程	(11)
1.5 实训操作	(13)
第 2 章 贪吃蛇游戏项目概述	(14)
2.1 项目概述	(14)
2.2 解决方案	(14)
2.2.1 工程	(15)
2.2.2 类说明	(15)
2.3 游戏说明与运行效果	(15)
2.3.1 游戏说明	(15)
2.3.2 运行部分效果	(16)
2.4 打包与运行	(17)
第 3 章 Java 语言基础	(21)
3.1 标识符、关键字和注释	(21)
3.1.1 标识符	(21)
3.1.2 关键字	(21)
3.1.3 注释	(22)
3.2 数据类型	(22)
3.2.1 数据类型的划分	(22)
3.2.2 变量和常量	(24)
3.2.3 类型转换	(26)
3.3 运算符和表达式	(27)
3.3.1 运算符	(27)
3.3.2 表达式	(29)
3.4 Java 流控制语句	(30)
3.4.1 分支语句	(30)

3.4.2	循环语句	(34)
3.4.3	跳转控制语句	(35)
3.5	案例：输出九九乘法表程序	(37)
3.6	实训操作	(37)
	习题	(38)
第4章	面向对象基础	(40)
4.1	面向对象基础概述	(40)
4.1.1	类与对象概述	(41)
4.1.2	封装和数据隐藏	(42)
4.1.3	继承和重用	(42)
4.1.4	多态性	(42)
4.1.5	方法与消息概述	(43)
4.2	类的定义及成员变量初始化	(43)
4.3	创建对象	(44)
4.3.1	对象的生命周期	(44)
4.3.2	创建对象	(45)
4.3.3	对象的使用	(45)
4.3.4	对象的释放	(47)
4.4	类成员的定义	(47)
4.4.1	成员的修饰符	(48)
4.4.2	常量的定义和最终方法	(50)
4.4.3	成员方法的重载——多态性体现	(52)
4.5	构造方法	(55)
4.5.1	构造方法的定义	(55)
4.5.2	带参数的构造方法	(57)
4.5.3	构造方法重载	(58)
4.6	this 的用法	(60)
4.7	静态成员	(63)
4.7.1	静态成员的特性	(63)
4.7.2	静态成员的使用	(64)
4.7.3	静态成员变量	(64)
4.7.4	静态成员方法	(65)
4.8	继承	(66)
4.8.1	继承的定义	(66)
4.8.2	继承说明	(67)
4.8.3	成员变量的隐藏	(69)
4.8.4	方法的覆盖	(71)
4.8.5	super 关键字	(73)
4.8.6	子类构造方法	(74)
4.9	抽象类	(76)

4.10	接口	(79)
4.10.1	接口的定义	(80)
4.10.2	接口的使用	(81)
4.11	包	(83)
4.11.1	包的创建	(83)
4.11.2	包的引用	(84)
4.12	内部类与匿名类	(84)
4.12.1	内部类	(85)
4.12.2	匿名类	(86)
4.13	案例：本章知识在贪吃蛇项目中的应用	(87)
4.14	实训操作	(96)
	习题	(101)
第5章	数组、常用类与集合	(110)
5.1	数组	(110)
5.1.1	一维数组	(110)
5.1.2	多维数组	(113)
5.2	字符串类	(115)
5.2.1	String 类及其函数	(115)
5.2.2	StringBuffer 类及其函数	(120)
5.3	Math 类	(121)
5.4	Random 类	(122)
5.5	数字类	(123)
5.5.1	包装类	(123)
5.5.2	数字类与字符串类型间转换的常用方法	(125)
5.6	日期类	(126)
5.7	集合	(130)
5.7.1	集合概述	(130)
5.7.2	Collection	(130)
5.7.3	List	(132)
5.7.4	Set	(136)
5.8	案例	(140)
5.8.1	案例一：输出杨辉三角形	(140)
5.8.2	案例二：简易学生管理系统	(141)
5.8.3	案例三：本章知识在贪吃蛇项目中的应用	(145)
5.9	实训操作	(146)
	习题	(146)
第6章	异常处理	(148)
6.1	异常处理的概念和 Java 异常处理的体系结构	(148)
6.2	异常处理	(150)
6.2.1	try-catch-finally 语句	(150)

6.2.2	throw 语句	(152)
6.2.3	throws 子句	(154)
6.3	自定义异常	(155)
6.4	案例：算数运算中的异常处理	(156)
6.5	实训操作	(158)
习题		(159)
第7章	输入/输出流	(162)
7.1	输入/输出概述	(162)
7.2	标准输入/输出流	(162)
7.3	文件类	(163)
7.4	字节输入/输出流	(165)
7.5	字符输入/输出流	(167)
7.6	缓冲流	(169)
7.7	随机存取文件	(171)
7.8	案例：内容的输入与存放	(172)
7.9	实训操作	(173)
习题		(174)
第8章	Applet	(175)
8.1	Applet 概述	(175)
8.2	Applet 的生命周期	(176)
8.3	paint()方法与 Graphics 类	(177)
8.4	案例：本章知识在贪吃蛇项目中的应用	(179)
8.5	实训操作	(183)
习题		(183)
第9章	图形用户界面	(185)
9.1	AWT	(185)
9.1.1	AWT 概述	(185)
9.1.2	组件和容器	(186)
9.2	布局设计	(189)
9.2.1	FlowLayout 布局	(189)
9.2.2	BorderLayout 布局	(190)
9.2.3	CardLayout 布局	(191)
9.2.4	GridLayout 布局	(192)
9.3	Swing	(194)
9.4	事件处理	(195)
9.4.1	事件处理机制	(196)
9.4.2	事件处理方法	(196)
9.4.3	典型事件处理	(199)
9.5	案例	(204)
9.5.1	案例一：简单计算器的实现	(204)

9.5.2	案例二：简单记事本的实现	(216)
9.5.3	案例三：本章知识在贪吃蛇项目中的应用	(221)
9.6	实训操作	(226)
习题	(226)
第 10 章	线程	(228)
10.1	线程概述	(228)
10.2	线程的创建与启动	(228)
10.3	线程的状态与常用方法	(231)
10.3.1	线程的状态	(231)
10.3.2	线程常用方法	(231)
10.4	线程优先级和调度	(233)
10.5	案例	(234)
10.5.1	案例一：模拟球的弹跳	(234)
10.5.2	案例二：本章知识在贪吃蛇项目中的应用	(239)
10.6	实训操作	(246)
习题	(246)
第 11 章	Java 网络编程	(247)
11.1	IP 地址与 InetAddress 类	(247)
11.1.1	IP 地址简介	(247)
11.1.2	InetAddress 类	(247)
11.2	基于 URL 的 Java 网络编程	(248)
11.2.1	统一资源定位符 URL	(248)
11.2.2	URL 组成	(249)
11.2.3	URL 创建	(249)
11.2.4	URL 解析	(250)
11.2.5	从 URL 读取 WWW 网络资源	(251)
11.3	基于 Socket 的 Java 网络编程	(252)
11.3.1	Socket 通信	(252)
11.3.2	Socket 通信机制	(252)
11.3.3	创建 Socket	(253)
11.3.4	客户端 Socket	(253)
11.3.5	服务器端的 ServerSocket	(253)
11.3.6	打开输入/输出流	(254)
11.3.7	关闭 Socket	(254)
11.4	案例：简易通信程序设计	(255)
11.5	实训操作	(257)
习题	(262)
第 12 章	数据库应用	(263)
12.1	JDBC	(263)
12.1.1	JDBC 概述	(263)

12.1.2	JDBC API	(264)
12.1.3	Java 访问数据库的步骤	(265)
12.2	查询记录	(267)
12.2.1	根据条件查询记录	(267)
12.2.2	对查询的记录进行排序输出	(269)
12.2.3	通配符查询	(272)
12.2.4	prepareStatement()方法的应用	(274)
12.3	更新记录	(276)
12.4	添加记录	(280)
12.5	删除记录	(283)
12.6	案例: 学生成绩管理系统	(285)
12.6.1	数据库的相关操作	(285)
12.6.2	主界面设计	(290)
12.6.3	查询模块设计	(291)
12.6.4	添加记录模块设计	(293)
12.6.5	修改记录模块设计	(295)
12.6.6	删除记录模块设计	(298)
12.7	实训操作	(298)
	习题	(299)
第 13 章	综合案例	(300)
13.1	公共聊天室程序说明	(300)
13.2	公共聊天室程序设计	(300)
13.2.1	登录流程	(300)
13.2.2	注册流程	(301)
13.2.3	聊天流程	(302)
13.2.4	聊天服务器	(303)
13.3	案例: 公共聊天室应用程序	(304)
13.3.1	数据处理实现	(304)
13.3.2	登录实现	(309)
13.3.3	注册实现	(310)
13.3.4	聊天实现	(312)
13.3.5	聊天服务器实现	(315)

第 1 章 Java 开发环境

本章要点:

- Java 语言简介
- Java 程序举例
- Java 开发环境搭建
- MyEclipse 集成开发工具使用

Java 是由 Sun 公司于 1991 年开发的新一代计算机高级编程语言, 该语言与平台无关且移植性很强, 在很多领域被广泛使用, 如 Java 程序可以在便携式计算机、电视、电话、手机和其他大型设备上运行。本章主要内容包括 Java 语言简介、开发环境的配置以及开发工具的配置与使用。

1.1 Java 语言简介

1. Java 语言的产生

Java 语言产生的原因是为家电类电子产品开发一个分布式代码系统。它由 Java 之父詹姆斯·戈士林博士设计。作为 Sun 研究院院士, 詹姆斯·戈士林亲手设计了 Java 语言, 并完成了 Java 技术的原始编译器和虚拟机 (Virtual Machine)。Java 最初的名字是 Oak, 在 1995 年被重命名为 Java。

Java 发展速度很快。Sun 公司于 1995 年 3 月发布 Java, Java 语言诞生。1996 年 1 月, JDK1.0 发布。次年 2 月, JDK1.1 发布。1998 年 12 月, JDK1.2 发布, 这是 Java 语言的里程碑, Java 也首次被划分为 J2SE、J2EE、J2ME 三个平台。不久 Sun 公司将 Java 改称 Java 2。2004 年 10 月, JDK1.5 发布。2006 年 6 月, JDK1.6 发布, 同时 Java 的各版本去掉 2 的称号, J2EE 改称 Java EE, J2SE 改称 Java SE, J2ME 改称 Java ME。

Java 是一种通过解释方式来执行的语言, 其语法规则和 C++ 类似。同时, Java 也是一种跨平台的程序设计语言, 用 Java 语言编写的程序可以运行在任何平台和设备上。例如在 Mac 苹果系统、各种微处理器硬件平台以及 Windows、OS/2、UNIX 等系统平台, 真正实现“一次编写, 到处运行”。Java 非常适合于企业网络和 Internet 环境, 由于提高了程序的可靠性与安全性, 因而成为 Internet 中最有影响力、最受欢迎的编程语言之一。

Java 语言编写的程序既是编译型的, 又是解释型的。程序代码经过编译之后转换成 Java 字节码, Java 虚拟机 (Java Virtual Machine) 对字节码进行解释和运行。编译只进行一次, 而解释在每次运行程序时都会进行。编译后的字节码采用一种针对虚拟机优化过的机器码形式保存, 然后在计算机上运行。Java 语言程序代码的编译和运行过程如图 1.1 所示。

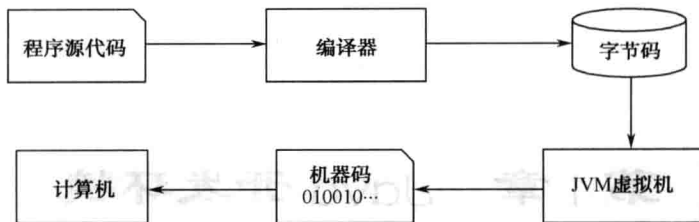


图 1.1 Java 程序的编译和运行过程

2. Java 语言的特点

Java 是一种应用比较广泛的网络编程语言，它面向对象、独立于平台运行且具有分布性、安全性、支持多线程等特性。具体特点如下：

(1) 面向对象。

Java 语言跟 C++、C# 一样，都是属于面向对象编程的语言。它提供类、接口、继承和重载等内容。与其他语言不一样，它只支持类之间的单继承，但接口之间可以实现多继承，通过继承实现了代码复用，从而提高开发效率。在类中还支持方法的重载与覆盖。总之，Java 语言是一种面向对象编程的高级语言。

(2) 简单性。

Java 语言使用简单。它的语法与 C 语言、C++ 语言很接近，使得大多数学过 C 语言与 C++ 语言的读者很容易上手学习 Java 语言。另一方面，Java 丢弃了 C++ 中很少使用的、很难理解的特性，如自动强制类型转换、操作符重载和多继承。Java 语言不再使用比较难且容易出错的指针，并提供了自动垃圾回收机制，使得程序员不必为内存管理而担忧，从而使得编程变得十分简单。

(3) 分布性。

Java 语言分布式特性主要是通过网络编程实现的。在网络编程中提供了网络访问接口 (java.net)，它提供了用于网络应用编程的 Socket、ServerSocket、URL 等类库。其中 URL 对象使用户能打开并访问具有相同 URL 地址上的对象，访问方式与访问本地文件系统相同。

(4) 健壮性。

在程序开发过程中，经常会遇到各种各样的异常，如编译时异常、运行时异常、内存异常等。面对各种各样的问题，Java 是如何解决的呢？Java 的强类型机制、异常处理、自动垃圾回收机制等可以解决上述问题。在实际应用过程中，它丢弃了指针的应用并进行安全检查，从而使得 Java 更具健壮性。

(5) 安全性。

Java 主要用于网络编程开发，因此对安全性有较高的要求。在 Internet 上如何防止恶意代码攻击是首先要解决的问题。在 Java 执行过程中，当 Java 字节码进入解释器时，首先要通过字节码校验器进行检查，没有问题后，Java 解释器将决定程序中类的内存布局。然后，类装载器负责将来自网络的类装载到单独的内存区域以防替代本地的同名类，从而保证代码以及数据的安全。

(6) 解释性。

Java 程序在运行时，首先通过编译器对 Java 程序语法进行检查，如果程序没有语法错误，将生成字节码文件，然后 Java 解释器再对这些字节码文件进行解释执行，将载入的类转换成机器能够直接执行的二进制代码。因此 Java 语言是一种解释性语言。

(7) 多线程。

在实际应用开发过程中，很多应用都会用到多线程。Java 在两方面支持多线程：一方面，Java 环境本身就是多线程，它们的主要作用是进行垃圾回收、系统维护等系统级操作；另一方面，Java 语言内置多线程控制，它提供了一个类 Thread，由它负责启动、终止线程，并可检查线程状态。利用 Java 的多线程编程接口，开发人员可以方便地写出很多支持多线程的应用程序，从而提高程序执行效率。

(8) 动态性。

Java 语言设计的主要目的是要适应动态变化的环境。Java 程序需要的类能够动态地被载入到运行环境，同时也可以通过网络来载入所需要的类以适应新的环境要求，从而使得程序开发更加灵活有效。

1.2 Java 程序举例

在 Java 程序开发应用过程中，Java 程序可分为两大类：一种是 Java 应用程序，另一种是 Java 小应用程序。现在以具体案例讲解两个程序的使用，通过这两个案例使读者对 Java 程序有一个初步的认识。

1.2.1 Java 应用程序

Java 程序可以用文本编辑器（如 Windows 的记事本）编写，也可以用开发工具进行编写。现在在记事本中新建一个 Java 程序，编写如下代码。代码编辑完成后，将该文件保存为 HelloWorld.java。

```
/** 第一个比较简单的 HelloWorld 程序 */
public class HelloWorld {
    public static void main(String args[]) {
        System.out.println("Hello World!");
    }
}
```

下面对上述程序进行简单说明：

(1) 程序第 1 行代码“/**”到“*/”为注释语句，不会被解释器执行。

(2) 程序中第 2 行定义了一个名字为“HelloWorld”的类，类的标志符为 class。public 为访问控制符，表示该类为公共类，该类的名称必须与文件的名称相同，包括大小写。每个编译单元最多只能有一个 public 类（也可没有），否则编译时会报错。

(3) 程序第 3 行中有一个方法 main()。在 Java 中，main()方法是 Java 应用程序的入口方法，程序运行时，第一个执行的方法就是 main()方法。这个方法和其他的方法有很大的不同，Java 要求该方法的名字必须是 main，修饰符必须是 public static void，方法必须接收一个字符串数组类型的参数（String args[]）。Java 程序中可以定义多个类，每个类可以定义多个方法，但最多只有一个类为公共类，且 main()方法也只能有一个。

(4) 程序第 4 行使用了 Java API 完成字符串的输出功能。System.out 为标准输出流对象，println()是此对象中的一个成员方法，其功能是输出括号中的字符串或其他类型的数据并换行。与该方法类似的是 print()方法，但是 print()方法输出内容后不会换行。

1.2.2 Java 小应用程序 (Applet)

Java 小应用程序即 Applet，该程序运行于浏览器上，可以生成生动美丽的页面，进行友好的人机交互，同时还能处理图像、声音、动画等多媒体数据。Applet 在 Java 的成长过程中起到了不可估量的作用，直到今天 Applet 依然是 Java 程序设计最吸引人的地方之一。现在以一个简单的案例讲解 Applet 程序的使用。打开记事本，编写 Applet 程序代码如下。

```
import java.awt.*;
import java.applet.*;
public class HelloWorld extends Applet
{
    public void paint(Graphics g)
    {
        g.drawString("Hello World!",5,35);
    }
}
```

将上述代码保存为 HelloWorld.java。现对该代码解释如下：

(1) 第 1~2 行代码采用了 import 命令，将 awt 和 applet 包中的类引入该类中，便于该类调用包中类的属性和方法。该命令具体怎么使用，后边章节会讲到。

(2) 第 3 行代码中的类 HelloWorld 继承了父类 Applet，这时 HelloWorld 拥有了 Applet 特性。继承是什么意思，后续章节会进行详细讲解。

(3) 第 5 行代码为 paint() 方法，该方法返回类型为 void，即该方法没有返回值。该方法中有一个参数“Graphics g”，g 为图像类 Graphics 对象，主要用于向浏览器输出图像信息。比如“g.drawString("Hello World!",5,35);”表示将“Hello World!”字符串输出到浏览器中。注意：小应用程序必须含有 paint() 方法。

由于小应用程序 Applet 的执行过程比较复杂，现写出后面具体的执行步骤。

第一步已经完成源代码的编写，现在第二步需要对源代码进行编译，在 DOS（命令提示符）下的操作为：

```
javac HelloWorld.java /*javac 命令会将 HelloWorld.java 生成 class 文件*/
```

这时将会生成 HelloWorld.class 字节码文件，便于后面 HTML 文件引用该字节码文件。

第三步，新建 HelloWorld.html 文件，代码如下：

```
<HTML>
<TITLE>HelloWorld! Applet</TITLE>
<APPLET
CODE="HelloWorld.class"
WIDTH=200
HEIGHT=100>
</APPLET>
</HTML>
```

从 HTML 文件可以看出，代码中嵌套了<APPLET>标签，在<APPLET>标签中有一个 CODE 属性，该属性主要引用字节码文件 HelloWorld.class。

第四步, 执行 HelloWorld.html 文件。如果用 appletviewer 运行 HelloWorld.html, 需在 DOS (命令提示符) 下输入如下的命令行:

```
appletviewer HelloWorld.html /*该语句调用了 appletviewer 命令, 表示直接运行 HelloWorld.html 中的<APPLET>程序*/
```

如果用浏览器运行 HelloWorld.html, 在浏览器的地址栏中输入 HTML 文件的 URL 地址即可。

这两种方式运行后都会在浏览器中显示:

```
Hello World!
```

1.3 Java 开发环境的搭建

在上一节中讲解了 Java 应用程序和 Java 小应用程序。如果想要运行这些程序, 必须首先进行环境配置。具体 JDK 的安装和环境变量的配置介绍如下。

1.3.1 JDK 的下载与安装

Sun 公司提供了自己的一套 Java 开发环境, 通常称为 JDK (Java Development Kit), 又称为 J2SDK。为了让客户和开发者在现有 Java 投资和 OpenJDK 参考实施的基础上获得构建和创新的机会, 甲骨文和 IBM 两家公司在 OpenJDK 社区进行合作, 共同开发技术领先的开源 Java 环境。现在 JDK 的较新版本是 JDK1.7, 但在实际应用中使用的是 JDK1.6 稳定版本。本教材使用的即为 JDK1.6。

可以通过网络 (<http://www.oracle.com/technetwork/java/javase/downloads/index.html>) 下载稳定版本 JDK1.6。进入下载地址后, 选择下载界面, 如图 1.2 所示, 单击“Download”按钮进行下载。

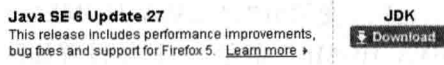


图 1.2 下载 JDK

下载完成后, 双击可执行文件 jdk-6u27-windows-i586.exe, 按照提示完成安装, 这里的安装路径选择为“C:\Program Files\Java\jdk1.6.0_27\”, 如图 1.3 所示。

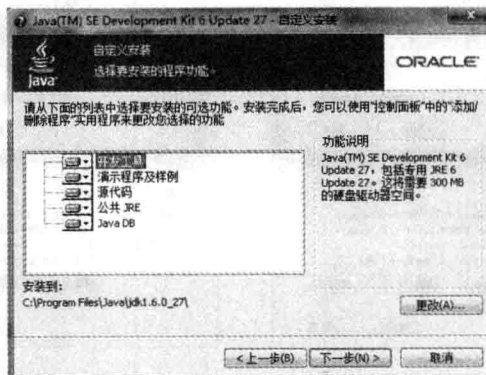


图 1.3 安装 JDK

直接单击图 1.3 中的“下一步”按钮完成 JRE 的安装后即可完成 JDK 的安装。

1.3.2 环境变量的配置

JDK 安装完成后还需要进行环境变量配置。在 Windows 系统桌面中，右击“我的电脑”图标，从弹出的快捷菜单中选择“属性”，在弹出窗口单击“高级”选项卡里面的“环境变量”按钮，如图 1.4 所示。在新打开的界面中系统变量需要设置三个属性。在没安装过 JDK 的电脑中“PATH”属性是本来存在的。

(1) 如图 1.5 所示，单击“新建”按钮，弹出“新建用户变量”对话框，在“变量名”中输入“JAVA_HOME”，顾名思义就是 Java 的安装路径，然后在“变量值”中输入 1.3.1 小节中的安装路径“C:\Program Files\Java\jdk1.6.0_27”。注意：变量值后边不要添加分号“;”。

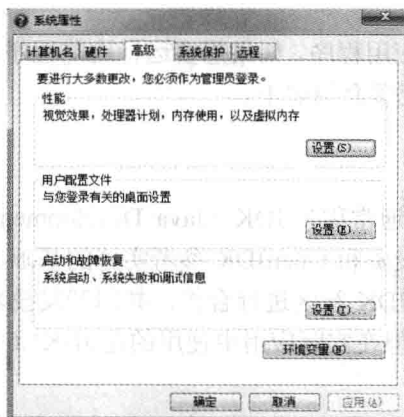


图 1.4 环境变量设置



图 1.5 JAVA_HOME 变量设置

(2) 在系统变量里选择变量“PATH”，单击“编辑”按钮。PATH 的含义就是系统在任何路径下都可以识别 Java 命令。添加变量值“;%JAVA_HOME%\bin;”（其中“% JAVA_HOME %”的含义是刚才设置的“JAVA_HOME”的值）。具体设置如图 1.6 所示。

(3) 选择变量“classpath”，单击“编辑”按钮，弹出窗口如图 1.7 所示。在“变量值”中填写“.;% JAVA_HOME %\lib;”（要加圆点“.”，表示当前路径）。该变量的含义为 Java 加载类（bin 或 lib）的路径，只有类在 classpath 中，Java 命令才能识别。



图 1.6 PATH 变量设置



图 1.7 classpath 变量设置

环境变量配置完成后，验证是否安装成功。单击“开始”菜单→“运行”，输入“cmd”，进入命令行界面，输入“java -version”，如果安装成功，则系统显示“java version "1.6.0_27"”（不同版本号则显示不同），如图 1.8 所示。

```
C:\Users\Administrator>java -version
java version "1.6.0_27"
Java(TM) SE Runtime Environment (build 1.6.0_27-b07)
Java HotSpot(TM) Client VM (build 20.2-b06, mixed mode, sharing)
```

图 1.8 JDK 测试

JDK 安装成功后，会生成很多目录，具体介绍如下：

- bin 目录：该目录表示存放可执行文件（如 java.exe、javac.exe、appletviewer.exe 等）。
- lib 目录：该目录表示开发工具使用的归档包文件。
- jre 目录：该目录表示 Java 运行时环境的根目录，包含 Java 虚拟机（JVM）、运行时的类包和 Java 应用启动器，但不包含开发环境中的开发工具。
- include 目录：该目录包含 C 语言头文件，支持 Java 本地接口与 Java 虚拟机调试程序接口的本地编程技术，用于支持 native-code 库使用 JVM Debugger 接口。
- demo 目录：该目录存放 JDK 自带的案例。
- jre/bin 目录：该目录存放 JRE 执行文件及 DLL 库，与 jdk6.0/bin 目录相同。
- jre/lib 目录：该目录存放 Java APP 运行时类库、默认参数设置和资源文件。
- sample 目录：该目录是 Sun 配带的帮助开发者学习的 Java 案例。
- JavaDB 目录：该目录是 Java 增加的新目录，主要与数据库有关。

1.3.3 Java 程序的编译与运行

Java 程序写好后，Java 程序是如何进行编译与运行的呢？这是通过 Java 虚拟机（JVM）来实现的。JVM 是一个虚构出来的计算机，是通过在实际的计算机上仿真模拟各种计算机功能来实现的。它有自己的硬件架构，如处理器、寄存器、堆栈等，还具有相应的指令系统。Java 程序编写完成后，首先通过 javac 命令进行编译，将文件后缀名为.java 的程序编译成 JVM 可执行的字节码文件即.class 文件。其次是执行字节码文件，该过程由解释器来完成。解释器的执行过程分为代码的装入、代码的校验和代码的执行。代码装入工作由类装载器完成，它主要负责查找和导入.class 文件。字节码校验器的主要作用是对代码进行校验，如果校验没有什么问题就开始执行代码。这就是 Java 虚拟机处理 Java 程序的原理。

下面讲解 Java 程序的编译与运行步骤。

- (1) 用编辑器如 Windows 记事本或者开发工具编写源程序，并将源程序命名为.java 文件。
- (2) 在 DOS（命令提示符）下运行 javac 命令，主要作用是将.java 程序生成字节码文件。格式为：javac ×××.java。
- (3) 执行第二步后，如果代码没有问题则生成×××.class 文件。然后运行“java ×××”来解释执行 Java 程序（×××为 Java 文件名）。

对于 1.2.1 小节中的 HelloWorld.java 应用程序，运行过程如下：

- (1) 在 DOS（命令提示符）下运行“javac HelloWorld.java”，如果程序没有错误，则生成 HelloWorld.class 文件，界面如图 1.9 所示。
- (2) 在 DOS 下运行“java HelloWorld”，则输出“Hello World!”，界面如图 1.10 所示。