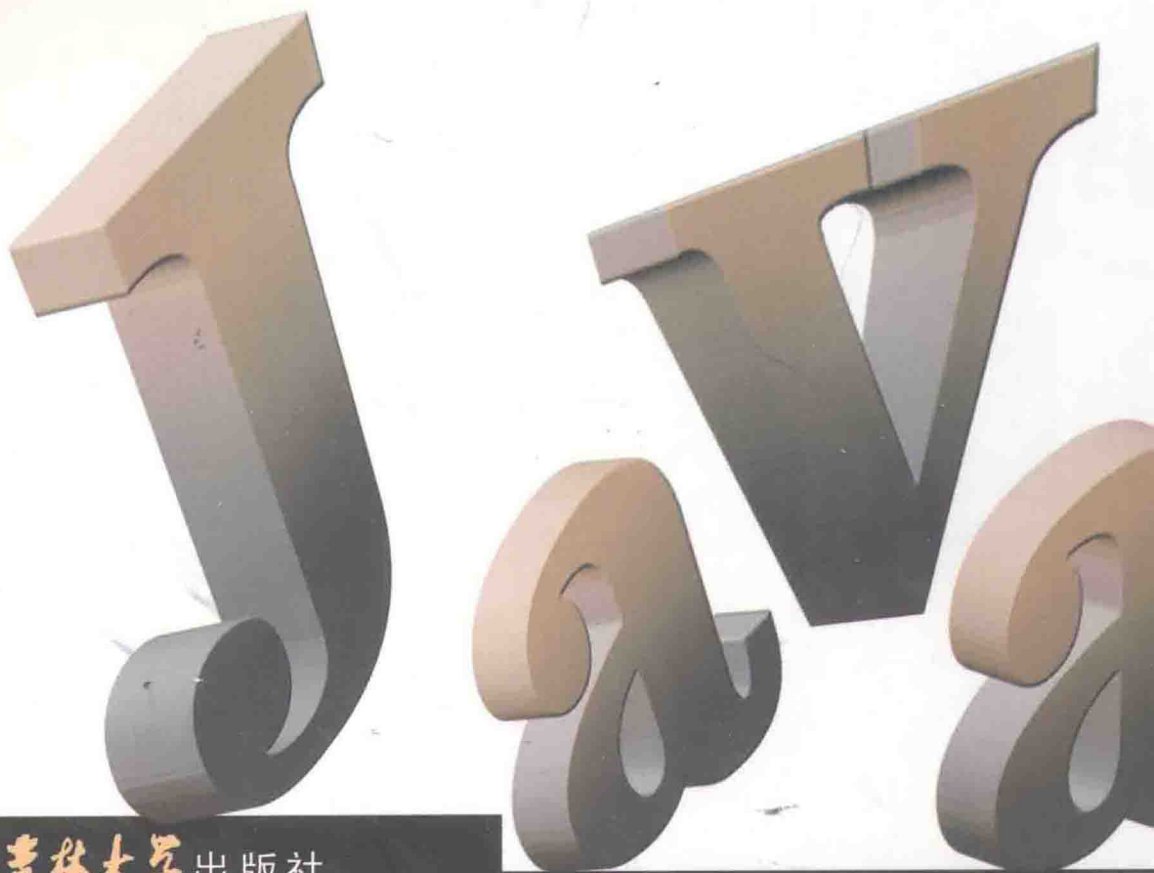


面向“十二五”高等教育规划教材

Java程序设计

陈显亭 徐学禹 刘跃娟 主编



 吉林大学出版社
JILIN UNIVERSITY PRESS

面向“十二五”高等教育规划教材

JAVA 程序设计

主 编 陈显亭 徐学禹
刘跃娟

副主编 王烨兴
参 编 贾晓飞

吉林大学出版社

图书在版编目(CIP)数据

JAVA 程序设计 / 陈显亭, 徐学禹, 刘跃娟主编. --
长春: 吉林大学出版社, 2010. 11
ISBN 978-7-5601-4976-9

I. ①J… II. ①陈… ②徐… ③刘… III. ①
JAVA 语言—程序设计—高等学校—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字(2010)第 229586 号

书 名: JAVA 程序设计

作 者: 陈显亭 徐学禹 刘跃娟 主编

责任编辑、责任校对: 刘冠宏 卢婵

吉林大学出版社出版、发行

开本: 787×1092 毫米 1/16

印张: 20 字数: 500 千字

ISBN 978-7-5601-4976-9

封面设计: 科发教材出版中心
北京广达印刷有限公司 印刷

2011 年元月 第 1 版

2011 年元月 第 1 次印刷

定价: 34.00 元

版权所有 翻印必究

社址: 长春市明德路 421 号 邮编: 130021

发行部电话: 0431-88499826

网址: <http://www.jlup.com.cn>

E-mail: jlup@mail.jlu.edu.cn

前 言

Java 程序设计语言是由 Sun 公司提出并开发的面向对象的跨平台编程语言。随着 Internet 和 Intranet 的迅速发展和以网络为中心的计算机的普及,客观上需要一种独立于平台、代码可移植的程序设计语言,Java 语言在技术上很好地满足了这一发展需求,它的平台无关性使 Java 成为分布式环境的最好语言,并且得到了迅速发展和广泛应用。

本书为 Java 语言编程的教材和参考用书,围绕 Java 语言本身的特色展开介绍。主要讲解了 Java 语言的基础内容和编程的基本思路,在面向对象的设计思想和技巧上作了深入思索和探讨,利用 Java 语言逐个讲解关于面向对象的知识点,使读者不仅能够学习到 Java 本身,而且最终掌握面向对象程序设计思想。

全书共 13 章,内容循序渐进,建议安排 64 学时来完成本课程(32 学时理论,32 学时上机)。

第 1 章介绍了 Java 的发展历史及 Java 语言的特性,讲解 JDK 的安装及配置方法,并介绍使用 Java 语言编写 Java 应用程序和 Java 小程序的基本方法。

第 2 章讲解 Java 语言基础,主要介绍 Java 程序设计语言的基本数据类型、运算符、表达式以及数组。

第 3 章介绍 Java 程序中的流程控制,包括选择结构、循环结构以及与程序转移有关的其他控制语句。

第 4 章主要介绍利用 Java 程序设计语言实现面向对象的抽象性和封装性的基本方法,包括类的定义、对象的构造、访问权限以及方法重载等。

第 5 章主要介绍利用 Java 程序设计语言实现面向对象的高级特性的基本方法,包括继承、多态、接口、包以及嵌套类等。

第 6 章学习异常的概念、Java 异常处理机制,包括 try-catch、多重 catch、try-catch-finally 几种异常流程控制的使用,throw、throws 关键字的使用和如何自定义异常类及使用自定义异常。

第 7 章主要介绍字符串处理,包括 String 类、StringBuffer 类、StringTokenizer 类的使用。

第 8 章讨论 Java 的多线程编程,包括线程的概念,如何在程序中实现多线程以及线程的同步、互斥及联合。

第 9 章介绍 Swing GUI 技术,包括 Swing 容器、布局管理器、Swing 常用组件以及 Java 事件处理机制。

第 10 章学习 Java 的 I/O 技术,包括字节流、字符流、数据流、对象流的读写以及对象序列化等。

第 11 章讲解 Java 的网络编程,介绍使用 Java 实现 URL 通信、Socket 通信和数据报通信的基本网络编程方法。

第12章主要介绍泛型程序设计的相关知识以及常用的Java对象容器接口及使用方法。

第13章主要介绍利用Java语言访问数据库的基本实现方式。

本书由陈显亭、徐学禹、刘跃娟任主编,王烨兴任副主编。具体分工如下:陈显亭(吉林电子信息职业技术学院)编写了第一、第二、第六、第九章;徐学禹(沈阳职业技术学院)编写了第四、第五、第十二章;刘跃娟(哈尔滨医科大学)编写了第三、第七、第十三章;王烨兴(齐齐哈尔医学院)编写了第十、第十一章。参编的还有贾晓飞(吉林电子信息职业技术学院)编写了第八章。

由于本书涉及内容广泛,加之作者水平有限、时间紧张,书稿虽几经修改,仍难免存在错误和不足,敬请读者给予批评指正。

作 者

2011年元月

目 录

第 1 章 Java 语言概述	(1)
1.1 Java 语言简介	(1)
1.2 Java 运行环境与开发环境	(2)
1.2.1 Java 三种平台简介	(2)
1.2.2 Java SDK 开发环境	(3)
1.3 一个 Java 程序的开发过程	(4)
1.3.1 Java 程序的开发过程	(4)
1.3.2 一个简单的 Java 应用程序的开发	(5)
1.3.3 一个简单的 Java 小应用程序的开发	(6)
1.4 Java 开发工具简介	(6)
小 结	(7)
习 题	(7)
第 2 章 Java 语言基础	(8)
2.1 基本数据类型与变量	(8)
2.1.1 标识符	(8)
2.1.2 保留字	(8)
2.1.3 空白与注释	(9)
2.1.4 基本数据类型	(9)
2.1.5 常量	(10)
2.1.6 变量	(11)
2.2 运算符与表达式	(12)
2.2.1 算术运算	(12)
2.2.2 关系运算	(13)
2.2.3 逻辑运算	(13)
2.2.4 赋值运算	(13)
2.2.5 位运算	(14)
2.2.6 数据类型转换	(14)
2.2.7 运算符的优先级	(16)
2.3 数 组	(17)
2.3.1 一维数组	(17)
2.3.2 二维数组	(18)
小 结	(20)

习 题	(21)
第 3 章 Java 语言流程控制语句	(23)
3.1 分支语句	(23)
3.1.1 if 语句	(23)
3.1.2 条件运算符	(26)
3.1.3 switch 语句	(26)
3.2 循环语句	(30)
3.2.1 while 语句	(30)
3.2.2 do-while 语句	(31)
3.2.3 for 语句	(31)
3.2.4 增强的 for 语句	(33)
3.2.5 循环嵌套	(34)
3.3 跳转语句	(34)
3.3.1 break 语句	(34)
3.3.2 continue 语句	(35)
3.3.3 返回语句 return	(35)
3.4 循环控制要点及循环语句比较	(36)
小 结	(38)
习 题	(39)
第 4 章 Java 面向对象编程基础	(40)
4.1 面向对象技术基础	(40)
4.1.1 面向对象的基本概念	(40)
4.1.2 面向对象的基本特征	(40)
4.2 Java 中的类	(41)
4.2.1 类的定义	(41)
4.2.2 类及其成员的访问控制符	(42)
4.2.3 数据成员的声明	(42)
4.2.4 成员方法的声明	(43)
4.3 对 象	(45)
4.3.1 对象的创建与使用	(45)
4.3.2 构造方法	(47)
4.4 访问权限	(51)
4.4.1 public 修饰符	(51)
4.4.2 private 修饰符	(51)
4.5 方法的重载	(52)
4.6 static、final 修饰符	(54)
4.6.1 static 修饰符	(54)
4.6.2 final 修饰符	(56)
4.7 参数的传递	(57)

4.7.1	方法参数的传递	(57)
4.7.2	命令行参数的使用	(58)
小 结		(59)
习 题		(59)
第 5 章	Java 面向对象编程进阶	(62)
5.1	继 承	(62)
5.1.1	Java 语言的继承语法	(62)
5.1.2	继承与成员访问控制	(65)
5.1.3	java.lang.Object 类介绍	(67)
5.1.4	成员变量的隐藏和方法的重写	(68)
5.1.5	super 与 this	(70)
5.1.6	对象的上转型对象	(72)
5.2	多态性	(73)
5.3	最终类最终方法与抽象类抽象方法	(78)
5.4	接 口	(82)
5.4.1	接口声明	(82)
5.4.2	接口与类	(83)
5.4.3	使用接口的优点	(88)
5.5	包	(88)
5.5.1	package 语句	(88)
5.5.2	import 语句	(89)
5.5.3	编译和运行包	(90)
5.5.4	包级访问控制	(91)
5.6	嵌套类	(93)
5.6.1	内部类	(94)
5.6.2	局部内部类	(95)
5.6.3	匿名内部类	(96)
5.6.4	静态嵌套类	(97)
小 结		(99)
习 题		(99)
第 6 章	异 常	(101)
6.1	异常类	(102)
6.1.1	Java 异常类的类层次	(102)
6.1.2	运行时异常与非运行时异常	(102)
6.1.3	创建用户异常类	(103)
6.2	异常的捕获与处理	(104)
6.2.1	Java 异常处理机制	(104)
6.2.2	try-catch-finally	(104)
6.2.3	抛出异常	(108)

小 结	(111)
习 题	(111)
第 7 章 字符串处理	(112)
7.1 String 类	(112)
7.1.1 创建 String 对象	(112)
7.1.2 String 类的常用方法	(114)
7.1.3 其他数据类型与字符串的相互转换	(118)
7.2 StringBuffer 类	(120)
7.2.1 创建 StringBuffer 对象	(120)
7.2.2 StringBuffer 类的常用方法	(120)
7.3 StringTokenizer 类	(123)
小 结	(124)
习 题	(124)
第 8 章 线 程	(126)
8.1 线程的概念	(126)
8.2 线程的控制与调度	(127)
8.2.1 创建线程	(127)
8.2.2 线程生命周期	(131)
8.2.3 线程的优先级	(132)
8.2.4 线程的调度	(132)
8.2.5 守护线程	(135)
8.3 线程的互斥	(136)
8.4 线程的同步	(140)
8.5 线程联合	(143)
小 结	(144)
习 题	(145)
第 9 章 图形用户界面	(146)
9.1 Java 程序的图形用户界面	(146)
9.1.1 Java 图形用户界面的组成	(146)
9.1.2 AWT 和 Swing 组件	(147)
9.1.3 Java 的事件处理模式	(148)
9.2 Swing 容器	(149)
9.2.1 顶层容器	(149)
9.2.2 中间容器	(154)
9.3 布局管理器	(156)
9.3.1 FlowLayout 布局管理器	(158)
9.3.2 BorderLayout 布局管理器	(159)
9.3.3 GridLayout 布局管理器	(161)
9.4 Swing 组件	(162)

9.4.1	标签.....	(163)
9.4.2	按钮.....	(166)
9.4.3	复选按钮.....	(168)
9.4.4	单选按钮.....	(171)
9.4.5	文本框.....	(174)
9.4.6	组合框.....	(177)
9.5	Java 事件类及处理	(179)
9.5.1	Java 事件类	(179)
9.5.2	窗口事件的处理.....	(181)
9.5.3	键盘事件的处理.....	(187)
9.5.4	鼠标事件的处理.....	(189)
9.5.5	语义事件的处理.....	(191)
小 结		(196)
习 题		(196)
第 10 章	Java 的输入/输出	(198)
10.1	File 类	(198)
10.2	字节流.....	(201)
10.2.1	FileInputStream 类	(201)
10.2.2	FileOutputStream 类.....	(203)
10.3	字符流.....	(204)
10.4	RandomAccessFile 类	(207)
10.5	数据流.....	(209)
10.6	对象流与序列化.....	(212)
小 结		(214)
习 题		(215)
第 11 章	网络通信	(216)
11.1	URL 通信	(216)
11.1.1	URL 通信基础	(216)
11.1.2	URL 类	(217)
11.1.3	通过字节流读取 WWW 资源	(218)
11.1.4	使用 HttpURLConnection 实现双向通信	(219)
11.2	Socket 通信	(220)
11.2.1	InetAddress 类	(222)
11.2.2	Socket 类与 ServerSocket 类	(223)
11.2.3	多线程的客户/服务器程序	(227)
11.2.4	URL 通信与 Socket 通信的区别	(231)
11.3	数据报通信.....	(231)
11.3.1	DatagramSocket 类	(232)
11.3.2	DatagramPacket 类	(232)

11.3.3	UDP 通信的示例程序	(233)
小 结		(235)
习 题		(236)
第 12 章	泛型程序设计与对象容器	(237)
12.1	泛型程序设计	(237)
12.1.1	泛型类的定义与使用	(238)
12.1.2	对象包装器	(240)
12.2	Java 对象容器	(241)
12.2.1	Java 对象容器接口	(241)
12.2.2	Java 的对象容器类	(243)
12.2.3	支持对象容器的辅助工具	(245)
12.3	迭代器	(245)
12.3.1	迭代器设计模式	(245)
12.3.2	迭代器的基本用法	(247)
12.3.3	列表专用迭代器	(247)
12.4	类集容器	(250)
12.4.1	类集容器接口	(250)
12.4.2	列表容器	(251)
12.4.3	集合容器	(253)
12.5	映射容器	(256)
12.5.1	映射容器接口	(256)
12.5.2	映射容器类	(257)
小 结		(258)
习 题		(259)
第 13 章	JDBC 与数据库应用	(260)
13.1	建立使用 JDBC 的运行环境	(260)
13.2	使用 JDBC—ODBC 桥接器访问数据库	(262)
13.3	Java 与 SQL 间的数据类型映射	(268)
13.4	修改、添加、删除记录	(269)
13.5	使用数据表的元信息	(271)
13.6	JDBC 编程实例	(274)
小 结		(281)
习 题		(282)
附录 A	Java 语言编码规范	(283)
附录 B	设计模式初步	(296)
参考文献		(308)

第 1 章 Java 语言概述

面向对象的编程语言使程序能够比较直观地反映客观世界的本来面目,并且使软件开发人员能够运用人类认识事物所采用的一般思维方法进行软件开发,这是当今计算机领域中软件开发的主流技术。所有面向对象的程序设计语言都支持对象、类、消息、封装、继承、多态等诸多概念,而这些概念是人们在进行软件开发、程序设计的过程中逐渐提出来的。

美国加州 Sun Microsystem 公司于 1995 年正式推出纯面向对象(object-oriented,简称 OO)的 Java 程序设计语言,具有面向对象、与平台无关、安全、稳定和多线程等优良特性,是目前软件设计中极为健壮的编程语言。由于它很好地解决了网络编程语言中的诸多问题,因此一经推出便受到了计算机界的普遍欢迎和接受,并得到了广泛的应用和发展,成为目前最为流行的面向对象程序设计语言之一。

1.1 Java 语言简介

Java 语言的出现是源于对平台无关语言的需要,希望这种语言能编写出嵌入各种家用电器等设备的芯片上,且易于维护的程序。但是,人们发现当时的编程语言,比如 C、C++ 等都有一个共同的缺点,那就是只能对特定的 CPU 芯片进行编译。这样,一旦电器设备更换了芯片就不能保证程序正确运行,就可能需要修改程序并针对新的芯片重新进行编译。1990 年 Sun 公司成立了由 James Gosling 领导的开发小组,开始致力于开发一种可移植的、跨平台的语言,该语言能生成正确运行于各种操作系统、各种 CPU 芯片上的代码。因此,Gosling 决定自行开发一种新的语言,并将该语言命名为 Oak(橡树),这便是 Java 语言的前身。

Java 语言的快速发展得益于 Internet 和 Web 的出现。1993 年 7 月,伊利诺伊大学的 NCSA 推出了一个在 Internet 上广为流行的 WWW 浏览器 Mosaic 1.0 版。然而,这时的 WWW 页面虽然内容丰富,比如可以实现声、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,同时又非常简练的语言,这时的 Oak 完全能满足这一要求。到 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。

由于 Java 确实是一种分布式、安全性高、内部包含的编译器非常小、适合网络开发环境的语言,因而一经发布,立即得到包括 Netscape 在内的各 WWW 厂商的广泛支持。工业界一致认为:“Java 是(20 世纪)80 年代以来计算机界的一件大事。”微软总裁 Bill Gates 认为:“Java 是长期以来最卓越的程序设计语言。”Java 已成为现在最流行的面向对象的网络编程语言。

Java 具有简单、面向对象、可移植、分布式、多线程、安全、动态等特点。

(1)简单:Java 语言产生于 C++ 语言之后,是完全面向对象的编程语言,充分吸取了 C++ 语言的优点,采用了程序员所熟悉的 C 和 C++ 语言的许多语法,同时又去掉了 C 语言

中指针、内存申请和释放等影响程序健壮性的部分,并提供自动垃圾回收机制以简化内存管理。因此,可以说 Java 语言是站在 C++ 语言这个“巨人的肩膀上”前进的。

(2)面向对象:Java 是纯面向对象的程序设计语言,支持封装性、多态性和继承性。本书将在第 4 章开始详细地讨论 Java 语言的面向对象特性。

(3)可移植:Java 语言的可移植性来源于它的平台无关性,这是 Java 语言最大的优势。Java 将源程序编译成字节码(二进制代码),这种字节码通过 Java 解释器来解释执行。任何一台机器,只要配备了 Java 解释器就可以运行 Java 字节码,而不管这种字节码是在何种平台上生成的。这不仅使开发的 Java 源代码是可移植的,甚至源代码经过编译之后形成的二进制代码——字节码——也同样是可移植的。因为 Java 源程序编译生成的字节码不是直接在操作系统平台上运行,而是在一个称为 Java 虚拟机(JVM)的平台上运行。利用 JVM 把 Java 字节码跟具体的软硬件平台分隔开来,这就保证在任何机器上编译的 Java 字节码文件都能在不同的机器上运行。

(4)分布式:基于 Java 开发的应用程序可以借助 URL 开启和存取网络对象,实现了数据的分布性;同时,Java 语言编写的 Applet 可以从服务器下载到客户端执行,服务器的部分计算因而能被转移到客户端进行,实现了操作的分布性。Java 还提供了一整套网络类库,开发人员可以利用类库进行网络程序设计,进而实现了 Java 语言的分布式特性。

(5)多线程:首先,Java 环境本身就是多线程的;其次,Java 语言内置对多线程的控制,可以大大简化多线程应用程序开发。实际上多线程使人产生多个任务在同时执行的错觉,因为,目前的计算机的处理器在同一时刻只能执行一个线程,但处理器可以在不同的线程之间快速地切换,由于处理器速度非常快,远远超过了人接收信息的速度,所以给人的感觉好像多个任务在同时执行。C++ 没有内置的多线程机制,因此必须调用操作系统的多线程功能来进行多线程程序的设计。

(6)安全:指针和释放内存等原 C++ 功能在 Java 中被删除,从而避免了非法内存操作。比如,当你准备从网络上下载一个程序时,你最大的担心是程序中含有恶意的代码,比如试图读取或删除本地机上的一些重要文件,甚至该程序是一个病毒程序等。当你使用支持 Java 的浏览器时,你可以放心地运行 Java 的小应用程序 Java Applet 而不必担心病毒的感染和恶意的企图,因为 Java 小应用程序将限制在 Java 运行环境中,不允许它访问计算机的其他部分。

(7)动态:Java 的动态特性是其面向对象设计方法的发展,它允许程序动态地装入运行过程中所需要类,这就使得 Java 可以在分布环境中动态地维护程序及类库,而不像 C++ 那样,每当其类库升级之后,相应的程序都必须修改,重新编译。

1.2 Java 运行环境与开发环境

1.2.1 Java 三种平台简介

Java 运行平台主要分为下列三个版本:

(1)J2SE:称为 Java 标准版或 Java 标准平台。J2SE 提供了标准的 SDK 开发平台。利用该平台可以开发 Java 桌面应用程序和低端的服务器应用程序,也可以开发 Java Applet 程序。

(2)J2EE:称为 Java 企业版或 Java 企业平台。J2EE 可以构建企业级的服务应用,J2EE

平台包含了 J2SE 平台,并增加了附加类库,以便支持目录管理、交易管理和企业级消息处理等功能。

(3)J2ME:称为 Java 微型版或 Java 小型平台。J2ME 是一种很小的 Java 运行环境,用于嵌入式的消费产品中,如移动电话、掌上电脑或其他无线设备等。

1.2.2 Java SDK 开发环境

1. 安装 Java Development Kit(JDK)

本书基于 J2SE 的 SDK1.5 版来学习 Java,可以登录到 Sun 的网站(<http://java.sun.com/>)免费下载 SDK1.5(jdk-1_5_0-windows-i586.exe)。安装的时候可以选择安装到任意的硬盘驱动器上,如安装到 D:\jdk1.5.0 目录下。正确安装后,在 JDK 目录下有 bin、demo、lib、jre 等子目录,如图 1-1 所示,其中 bin 目录保存了 javac、java、appletviewer 等命令文件,demo 目录保存了许多 Java 的例子,lib 目录保存了 Java 的类库文件,jre 目录保存的是 Java 运行时的环境(JRE)。在安装 SDK 的同时,计算机就安装上了 Java 运行环境平台。

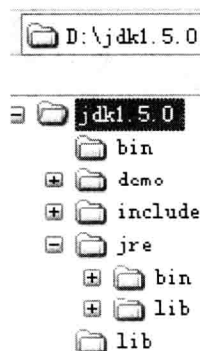


图 1-1 SDK 目录结构

2. 环境变量的设置

设置环境变量的目的是为了能够正常使用所安装的 JDK 开发包。通常,我们需要设置三个环境变量:JAVA_HOME、PATH 和 CLASSPATH。

(1)JAVA_HOME:该环境变量的值就是 Java 所在的目录,一些 Java 版的软件和一些 Java 的工具需要用到该变量,设置 PATH 和 CLASSPATH 的时候,也可以使用该变量以方便设置。

(2)PATH:该环境变量指定一个路径列表,用于搜索可执行文件。执行一个可执行文件时,如果该文件不能在当前路径下找到,则依次寻找 PATH 中的每一个路径,直至找到待执行的文件,或者找完 PATH 中的路径也不能找到,则报错。Java 的编译命令(javac)、执行命令(java)和一些工具命令(javadoc、jdb 等)都在其安装路径下的 bin 目录中。

(3)CLASSPATH:该环境变量也指定一个路径列表,是用于搜索 Java 编译或者运行时需要用到的类。在 CLASSPATH 列表中除了可以包含路径外,还可以包含 jar 文件。Java 查找类时会把这个 jar 文件当做一个目录来进行查找。通常,我们需要把 JDK 安装路径下的 jre\lib\rt.jar 包含在 CLASSPATH 中。

设置环境变量有以下三种方法:

(1)在系统特性中设置 PATH 和 CLASSPATH。

在 Windows 2000、Windows 2003、Windows XP 系统中,右击“我的电脑”,在弹出的快捷菜单中,选择“属性”命令,弹出“系统特性”对话框,再单击该对话框中的“高级”选项,然后单击“环境变量”按钮,添加如下的环境变量:

变量名:JAVA_HOME

变量值:d:\jdk1.5.0(其内容应根据 JDK 安装目录变化)

变量名:PATH

变量值:.;%JAVA_HOME%\bin;(其内容应根据 JDK 安装目录变化)

变量名:CLASSPATH

变量值:.;%JAVA_HOME%\jre\lib\rt.jar;(其内容应根据实际情况变化)

(2)在 MS-DOS 命令行窗口设置。

也可以在 MS-DOS 命令行键入下列命令后,回车确认。例如:

```
set JAVA_HOME=d:\jdk1.5.0;
```

```
set PATH=.; %JAVA_HOME%\bin;
```

```
set CLASSPATH = .;%JAVA_HOME%\jre\lib\rt.jar;
```

注意:这种方式设置的环境变量只对本 DOS 窗口有效。

(3)修改系统自动批处理文件。

用记事本编辑 Autoexec.bat 文件,将上面 MS-DOS 命令行窗口输入的语句加入即可。

注意:path 及 classpath 环境变量设置中的“.”是对应用程序的当前目录。

1.3 一个 Java 程序的开发过程

1.3.1 Java 程序的开发过程

Java 程序分成三类:Application(应用程序)、Applet(小程序)和 servlet(服务器端小程序)。应用程序在计算机中单独运行,而小程序只能嵌在 HTML 网页中由浏览器执行。servlet 是运行在服务器端的小程序,它可以处理客户传来的请求,然后将执行结果传给客户端。无论哪种 Java 程序,其基本开发过程都如图 1-2 所示。

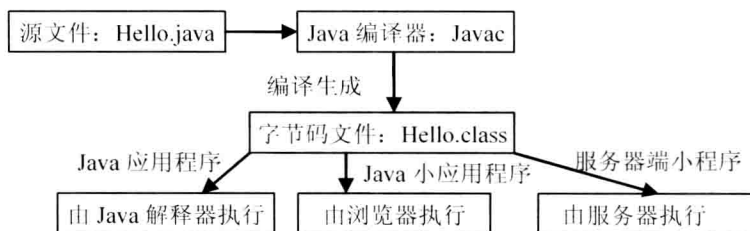


图 1-2 Java 程序的开发过程

(1)编写源文件:使用一个文字编辑器(如 UltraEdit 或记事本)来编写源文件。不可以使用 Word 或者写字板等带格式控制的编辑器,因为它们含有不可见字符。然后将编好的源文件以扩展名.java 保存起来。

(2)编译 Java 源程序:使用 javac.exe 编译源文件得到字节码文件,其扩展名为.class。

(3)运行 Java 程序:如前所述,Java 程序分为三类:Java 应用程序和 Java 小应用程序(Applet)、servlet(服务器端小程序),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,而 servlet 必须在应用服务器中执行。下面对 Java 应用程序和小应用程序的开发过程进行简单介绍。

1.3.2 一个简单的 Java 应用程序的开发

1. 编写源文件

```
public class HelloWorld
{
    public static void main(String args[])
    {
        System.out.println("Hello World!");
    }
}
```

注意:

(1)一个 Java 源程序是由若干个类组成的。上面的这个 Java 应用程序简单到只有一个类,类的名字是叫 HelloWorld。

(2)class 是 Java 的关键字,用来定义一个类。public 也是关键字,说明 Hello 是一个 public 类。第一个大括号和最后一个大括号及它们之间的内容叫做类体。我们将会在后面章节中系统地学习类的定义和使用。

(3)public static void main(String args[])是类体中的一个方法,main 方法是程序开始执行的位置。在一个 Java 应用程序中 main 方法必须被说明为 public static void。

(4)String args[]声明一个字符串类型的数组 args[] (注意 String 的第一个字母是大写的,它是 main 方法的参数,以后会学习怎样使用这个参数)。

(5)将源文件命名为 HelloWorld.java。源文件的命名规则是这样的:如果源文件中有多类,那么只能有一个类是 public 类;如果有一个类是 public 类,那么源文件的名称必须与这个 public 类的名称完全相同(大小也必须相同),扩展名是 .java。如果源文件没有 public 类,那么源文件的名称只要和某个类的名称相同并且扩展名是 .java 就可以了。

2. 编译

当创建了 HelloWorld.java 这个源文件后,就要使用 java 编译器 javac.exe 对其进行编译:

```
javac Hello.java
```

编译完成后在当前目录下生成一个 HelloWorld.class 文件,该文件称为字节码文件。

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

3. 执行

Java 应用程序必须通过 Java 虚拟机中的 java 解释器(java.exe)来解释执行其字节码文件。Java 应用程序总是从主类的 main 方法开始执行。因此,必须按如下方式运行 Java 应用程序:

```
java HelloWorld
```

MS-DOS 命令行窗口将显示如下信息:

Hello World!

注意:当 Java 应用程序中有多个类时,运行 java.exe 时,跟在其后的类名必须是包含了 main 方法的那个类的名字,即主类的名字。

1.3.3 一个简单的 Java 小应用程序的开发

Java 小应用程序的建立及运行可分为下述四个步骤:

(1)建立 Java 小应用程序源程序文件。

```
import java.awt.*;  
import java.applet.*;  
public class HelloWorldApplet extends Applet  
{  
    public void paint(Graphics g)  
    {  
        g.drawString("Hello World!",25,25);  
    }  
}
```

Java 小应用程序不再需要 main 方法,但必须继承自 Applet 类,即它是 Applet 类的子类。

上述源程序中我们使用了 import 语句,这是因为我们要使用系统提供给我们的 Applet 类。Applet 类在包 java.applet 中。关于包以后还会讲解。

(2)利用 Java 编译器(Javac)编译该 Java Applet。产生.class 字节码文件。

```
javac HelloWorldApplet.java
```

(3)建立一个 HTML 文件,在其中嵌入 Java 字节码文件。

在记事本中建立名为 Test.html 的 HTML 文件,并输入下面两行语句。

```
<APPLET CODE="HelloWorld.class" width=150 height=100>  
</APPLET>
```

(4)用 WWW 浏览器或 appletviewer 装入 HTML 文件,使该 Applet 运行。图 1-3 是用 appletviewer test.html 的运行结果。



图 1-3 HelloWorldApplet 运行结果

1.4 Java 开发工具简介

Java 程序的开发,首先需要用到文本编辑器来编辑源文件,最简单的是 Windows 系统自带的记事本。不过,建议大家还是安装一个专门的编辑工具,常用的有 UltraEdit,当前的版本是 16.0,其下载地址是:<http://www.ultraedit.com/>。

除了纯文本编辑器外,Java 也有许多集成开发环境,我们给大家推荐两款免费下载的集成开发环境,如 Gel 和 Eclipse。