



高等学校精品规划教材

Java面向对象程序设计 (第二版)

主 编 赵生慧

副主编 刘 奎 陈业斌 何爱华



中国水利水电出版社
www.waterpub.com.cn

21 世纪高等学校精品规划教材

Java 面向对象程序设计

（第二版）

主 编 赵生慧

副主编 刘 奎 陈业斌 何爱华



中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

Java 语言是当今面向对象程序设计语言的代表之一。本书循序渐进地介绍了 Java 语言程序设计基础, 面向对象程序设计, 图形用户界面设计, 异常处理, applet 小程序设计, I/O 流及 Java 高级编程等。

本书由 13 章组成。第 1 章介绍了 Java 的特点及 Java 程序的开发环境。第 2 章讨论了 Java 中的基本数据类型和运算表达式。第 3 章介绍了类与对象的概念及简单应用。第 4 章进一步讨论了面向对象的继承与多态性。第 5 章说明了程序控制结构及相应的语句。第 6 章则对数组的创建及应用展开了讨论。第 7 章介绍了系统包和常用类以及它们的应用。第 8 章是接口与抽象类的说明与应用, 第 9 章讨论了异常处理的方法。第 10 章详细介绍了图形用户界面及其设计。第 11 章则讨论了 Java 的输入与输出流。第 12 章对 applet 程序设计作了介绍。第 13 章讨论 Java 的高级编程。

本书应用性强, 讲解清晰透彻, 每一章均由引例开始, 配备了丰富的例题与习题, 不仅适合作为大专院校计算机公共课程和专业课程的 Java 语言入门教材, 也可供专业程序设计人员使用。

本书配有电子教案, 读者可以到中国水利水电出版社或万水书苑网站免费下载, 网址:
<http://www.waterpub.com.cn/softdown/>或 <http://www.wsbookshow.com>。

图书在版编目 (C I P) 数据

Java 面向对象程序设计 / 赵生慧主编. -- 2 版. --
北京: 中国水利水电出版社, 2010. 8
21 世纪高等学校精品规划教材
ISBN 978-7-5084-7650-6

I. ①J… II. ①赵… III. ①JAVA 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 121409 号

策划编辑: 雷顺加 责任编辑: 李炎 加工编辑: 李冰 封面设计: 李 佳

书 名	21 世纪高等学校精品规划教材 Java 面向对象程序设计 (第二版)
作 者	主 编 赵生慧 副主编 刘 奎 陈业斌 何爱华
出版发行	中国水利水电出版社 (北京市海淀区玉渊潭南路 1 号 D 座 100038) 网址: www.waterpub.com.cn E-mail: mchannel@263.net (万水) sales@waterpub.com.cn 电话: (010) 68367658 (营销中心)、82562819 (万水)
经 售	全国各地新华书店和相关出版物销售网点
排 版	北京万水电子信息有限公司
印 刷	北京蓝空印刷厂
规 格	184mm×260mm 16 开本 18.25 印张 443 千字
版 次	2007 年 7 月第 1 版 2010 年 8 月第 2 版 2010 年 8 月第 6 次印刷
印 数	20001—24000 册
定 价	30.00 元

凡购买我社图书, 如有缺页、倒页、脱页的, 本社营销中心负责调换
版权所有·侵权必究

前 言

Java 是一种优秀的面向对象程序设计语言，其平台无关性以及 Internet 应用的支持等多种特点使得其成为当今程序设计语言的代表。更重要的是，它已经有了相当广泛的市场基础，几乎成为软件开发人员及程序员的必备技术。在网络编程、数据库编程、移动通信开发等方面，Java 语言均占有优势。同时，由于 Java 与 Internet 的天然联系，使得开发基于 Java 的 Web 应用变得非常轻松。

根据社会对应用型人才的需求定位，要求高等学校在人才培养时注重教学内容的实用性和应用性，要不断改革教学目标、内容及教学方法，加强对应用能力及学习方法的培养。本书作为一门实用性很强的课程，也突出了其应用性和方法性。

本书采用循序渐进的方法，理论与实践相结合，既有基本的理论介绍，又注重技术的应用及实践。对类和对象这些关键的概念，逐步引入，并结合实际生活中的应用设计案例。对一些抽象的概念，例如接口、抽象类和输入输出流等，均从相关概念延伸出，结合具体例题解释。教材中的引例和实例大部分给出了解题思路和程序分析，以帮助读者理解程序。这种编写方式有利于初学者掌握程序设计流程和编程思想，也能为读者的后续课程学习打下基础。

本书由 13 章组成。第 1 章介绍了 Java 的特点及 Java 程序的开发环境。第 2 章讨论了 Java 中的基本数据类型和运算表达式。第 3 章介绍了类与对象的概念及简单应用。第 4 章进一步讨论了面向对象的继承与多态性。第 5 章说明了程序控制结构及相应的语句。第 6 章则对数组的创建及应用展开了讨论。第 7 章介绍了系统包和常用类以及它们的应用。第 8 章是接口与抽象类的说明与应用，第 9 章讨论了异常处理的方法。第 10 章详细介绍了图形用户界面及其设计。第 11 章则讨论了 Java 的输入与输出流。第 12 章对 applet 程序设计作了介绍。第 13 章讨论 Java 的高级编程。

本书将出版配套的实验指导书，为教学与自学提供方便。在实验指导书中，给出了本书的大部分习题参考解答，且包含与本教材配套的光盘，光盘中给出了本书所有的程序代码、辅助教学系统及教学 PPT 文档。应用辅助教学系统，读者可以从题库中抽取习题进行练习，学习和巩固知识。

参加本书编写的人员均为长期从事计算机教学的教师及技术开发人员，有丰富的 Java 应用及教学经验。本书第 1、2 章由赵生慧编写，第 3、4 章由陈业斌编写，第 5、6 章由何爱华编写，第 7、8、10 章由刘奎编写，第 9 章由陈平编写，第 11 章由马骏编写，第 12、13 章由黄晓玲编写。全书由赵生慧统稿。编写过程中，得到了陈桂林、计成超、陈海宝、李跃民、杨传健等各位教师的帮助。

由于作者水平所限，书中难免存在一些缺点和错误，期待广大读者批评指正。

作 者
2010 年 6 月

目 录

前言

第1章 了解 Java	1
1.1 Java 概述	1
1.1.1 Java 的起源	1
1.1.2 Java 平台的构成	2
1.1.3 Java 的特点	2
1.2 Java 开发环境与应用程序举例	4
1.2.1 Java 程序开发环境	4
1.2.2 Java 应用程序结构	6
1.2.3 Java 应用程序开发过程	8
1.3 Java Applet 程序举例	9
1.3.1 Java Applet 的开发过程	10
1.3.2 Applet 程序组成	11
1.4 面向对象程序设计基础	12
1.4.1 对象和类	12
1.4.2 面向对象特性	14
1.4.3 面向对象程序设计过程	14
习题一	15
第2章 基本数据类型与运算	17
2.1 引例	17
2.2 标识符	19
2.2.1 标识符的分类	19
2.2.2 关键字	20
2.3 Java 基本数据类型	20
2.3.1 数值型	20
2.3.2 字符型	21
2.3.3 布尔型	21
2.4 常量、变量与赋值	21
2.4.1 常量	22
2.4.2 变量	23
2.4.3 赋值语句	24
2.4.4 类型转换	25
2.5 运算符与表达式	26
2.5.1 算术运算符和算术表达式	26

2.5.2 关系运算和逻辑运算	27
2.6 其他运算符	28
2.6.1 条件运算符	28
2.6.2 位运算符	29
2.6.3 赋值运算符和赋值表达式	29
2.6.4 运算符优先级	30
2.7 实例	32
习题二	34
第3章 类与对象	37
3.1 引例	37
3.2 类	38
3.2.1 类的声明	39
3.2.2 类体的构成	39
3.3 对象	44
3.3.1 对象的创建	44
3.3.2 对象的使用	45
3.3.3 垃圾对象的回收	47
3.4 成员变量	47
3.4.1 实例变量和类变量	47
3.4.2 常量	48
3.5 成员方法	49
3.5.1 实例方法和类方法	49
3.5.2 方法中的参数传递	50
3.6 关键字 this	53
3.7 内部类	54
3.8 自定义包	56
3.8.1 创建包	56
3.8.2 使用包	56
3.9 访问权限	58
3.9.1 类与构造方法的访问权限	58
3.9.2 成员变量和成员方法的访问权限	58
3.10 泛型类	60
3.11 实例	61

习题三	65	6.2.3 数组的赋值及引用	107
第4章 类的继承和多态	68	6.3 字符数组	108
4.1 引例	68	6.3.1 字符数组的声明和创建	109
4.2 继承	70	6.3.2 字符串与字符数组	109
4.2.1 继承的概念	70	6.4 对象数组	110
4.2.2 子类与父类	70	6.5 多维数组	111
4.2.3 子类的继承性	71	6.5.1 二维数组的声明、创建和初始化	111
4.3 创建子类对象	74	6.5.2 不规则数组	112
4.4 关键字 super	76	6.6 ArrayList 类	113
4.5 final 修饰符	78	6.7 实例	114
4.6 多态性	79	习题六	116
4.6.1 多态的定义与作用	79	第7章 系统包与常用类	118
4.6.2 方法的重载	79	7.1 Java 系统包	118
4.6.3 上转型对象	80	7.2 Java 常用类	119
4.6.4 方法的覆盖	81	7.2.1 基本数据类型类	120
4.7 实例	82	7.2.2 String 类	121
习题四	85	7.2.3 StringBuffer 类	125
第5章 基本控制结构与实现	88	7.2.4 System 类	126
5.1 引例	88	7.2.5 Math 类	127
5.2 选择语句	90	7.2.6 Random 类	129
5.2.1 单分支 if 语句	90	7.2.7 日期类	130
5.2.2 双分支 if...else 语句	91	7.2.8 Vector 类	131
5.2.3 嵌套 if 语句	93	7.3 实例	133
5.2.4 switch 语句	93	习题七	134
5.3 循环语句	94	第8章 接口与抽象类	136
5.3.1 while 循环	94	8.1 引例	136
5.3.2 do-while 循环	96	8.2 接口	137
5.3.3 for 循环	97	8.2.1 定义接口	138
5.4 控制转移语句	98	8.2.2 接口实现	138
5.4.1 break 语句	98	8.3 集合接口	139
5.4.2 continue 语句	99	8.3.1 Collection 接口	140
5.4.3 return 语句	99	8.3.2 List 接口	140
5.5 实例	100	8.3.3 Set 接口	141
习题五	102	8.3.4 Iterator 接口	142
第6章 使用数组	105	8.4 抽象类	143
6.1 引例	105	8.5 实例	144
6.2 声明及创建数组	106	习题八	147
6.2.1 声明数组	106	第9章 异常处理	149
6.2.2 创建数组	106	9.1 引例	149

9.2 异常和异常类	150	11.2.3 窗口 JFrame	193
9.3 异常处理	151	11.3 Swing 基本组件	195
9.3.1 一个异常处理示例	151	11.3.1 JLabel 组件	195
9.3.2 捕获异常	152	11.3.2 JButton 组件	196
9.3.3 抛出异常	153	11.3.3 JTextField 与 JTextArea 组件	197
9.4 自定义异常类	156	11.3.4 JCheckBox 组件	201
9.5 实例	157	11.3.5 JRadioButton 组件	202
习题九	158	11.4 布局管理器	203
第 10 章 Java 的输入与输出流	160	11.4.1 FlowLayout	204
10.1 引例	160	11.4.2 BorderLayout	205
10.2 流	161	11.4.3 GridLayout	206
10.3 标准输入/输出流	162	11.4.4 CardLayout	207
10.4 文件访问	163	11.4.5 GridBagLayout	207
10.5 字节流	165	11.4.6 容器的嵌套	208
10.5.1 InputStream 和 OutputStream 类	165	11.5 事件驱动设计	209
10.5.2 FileInputStream 类和 FileOutputStream 类	166	11.5.1 委托事件模型	209
10.5.3 BufferedInputStream 类和 BufferedOutputStream 类	169	11.5.2 事件类	211
10.5.4 DataInputStream 类和 DataOutputStream 类	171	11.5.3 ActionEvent 类	212
10.6 字符流	172	11.5.4 KeyEvent 类	214
10.6.1 Reader 类和 Writer 类	173	11.5.5 MouseEvent 类	217
10.6.2 FileReader 类和 FileWriter 类	173	11.5.6 WindowEvent 类	219
10.6.3 BufferedReader 类和 BufferedWriter 类	174	11.6 Swing 其他组件	221
10.7 随机读写文件	175	11.6.1 JList 组件	221
10.8 对象串行化	177	11.6.2 JComboBox 组件	223
10.9 实例	179	11.6.3 JScrollBar 组件	225
习题十	182	11.6.4 JDialog 组件	227
第 11 章 编写图形用户界面程序	183	11.6.5 JOptionPane 组件	229
11.1 GUI 概述	183	11.6.6 菜单的设计	231
11.1.1 GUI 组成元素分类	184	11.7 实例	234
11.1.2 AWT 和 Swing 介绍	184	习题十一	238
11.1.3 颜色、字体和图形	185	第 12 章 applet 程序设计	239
11.1.4 使用 AWT 创建图形用户界面	188	12.1 引例	239
11.2 Swing 容器	190	12.2 applet 的基本工作原理	241
11.2.1 简单的窗口对象示例	190	12.2.1 applet 的工作机制	241
11.2.2 Swing 窗口对象	191	12.2.2 applet 的生命周期	242
		12.3 applet 的创建和运行	246
		12.3.1 applet 的程序框架	246
		12.3.2 applet 与 HTML 文件的配合	247
		12.3.3 applet 的执行	249

12.3.4	applet 与 application	250	13.2	数据库编程	266
12.4	applet 与多媒体技术	253	13.2.1	JDBC 概述	266
12.5	实例	255	13.2.2	使用 JDBC 进行数据库开发	267
	习题十二	256	13.3	网络编程	273
第 13 章	Java 高级编程	257	13.3.1	URL 编程	273
13.1	多线程程序设计	257	13.3.2	基于 TCP/IP 协议的 Socket 编程	275
13.1.1	进程与线程	257	13.3.3	基于 UDP 协议的 Socket 编程	279
13.1.2	多线程定义	258		习题十三	281
13.1.3	多线程的实现方法	259	参考文献		282
13.1.4	多线程同步与控制	262			

第 1 章 了解 Java



本章内容

- Java 的起源、发展和主要特点。
- Java 的基本开发环境。
- Java 应用程序的基本结构与组成元素。
- 编辑、编译及运行 Java 应用程序的基本方法与过程。
- Java 小程序的组成与运行。
- 类与对象的概念，面向对象的基本特性，以及面向对象程序设计的基本过程。



学习目标

- 了解 Java 的技术背景以及 Internet 的发展对 Java 的推动作用。
- 理解平台无关性及可移植性的含义。
- 掌握编写具有简单输出功能的 Java 应用程序的方法。
- 熟悉 JDK 的安装以及编辑、编译、运行 Java 应用程序的方法。
- 理解类、对象、属性、方法等面向对象的基本概念。
- 了解封装、继承及多态等面向对象的基本特性。

1.1 Java 概述

随着程序设计技术的改革及计算环境的改变，Java 语言也得到了迅速发展。一方面，C++ 等面向对象程序设计语言的发展为 Java 提供了技术基础。事实上，Java 也大量地继承了 C 及 C++ 的成果，并增加了体现程序设计技术发展状态的功能。另一方面，网络及 Internet 的发展，对程序设计语言提出了新的要求，为 Java 的发展注入了强大的动力。正是 Internet 的快速发展和普及，改变了传统的计算模式，促进了 Java 的普及与流行。

1.1.1 Java 的起源

Java 是由 Sun Microsystems 的 James Gosling 所领导的开发小组设计的。最初的版本是 1991 年的 Oak（橡树），其目标是设计独立于平台且能够嵌入到不同的消费类电子产品的程序。所谓独立于平台，是指通过该语言生成的代码可以在不同体系结构的 CPU 上运行，也可以在异构的操作系统环境下运行。

随着 Internet 及 WWW 的发展，人们发现 Web 也需要在不同的环境、不同的平台上进行程序的移植，这个变化导致了 Oak 的转型及 Java 的诞生。1995 年，Sun 公司的技术人员对 Oak

进行了修改,用于开发 Internet 应用程序,并将其命名为 Java。今天,Java 技术的多功能性、有效性、平台的可移植性以及安全性已经使它成为网络计算领域最完美的技术之一。通过 Java 可以:

- 在一个平台上编写软件,然后在另一个平台上运行。
- 创建在 Web 浏览器中运行的程序。
- 开发适用于联机论坛、即时通讯、在线投票之类的以 HTML 或者 XML 格式处理的服务器端应用程序。
- 开发基于面向服务体系结构的应用程序,并将应用程序封装为服务向用户发布。
- 为移动电话、远程处理器、低成本的消费产品以及任何具有数字核心的设备编写强大而高效的嵌入式应用程序。

1.1.2 Java 平台的构成

Java 平台包括核心 JVM 以及 Java API。

1. JVM

JVM (Java 虚拟机, Java Virtual Machine) 是一种纯软件形式的计算机,Java 虚拟机由具体的硬件平台及相应的 Java 解释器组成,解释器的功能是将字节码翻译成目标机器上的机器语言。

2. Java 平台的三种版本

从诞生到现在,Java 经历了许多变化,1998 年 Sun Microsystems 发布了 JDK1.2,从此称之为 Java 2 平台,1999 年开始,Java 2 被分成 J2SE (Java 2 Platform, Standard Edition)、J2EE (Java 2 Platform, Enterprise Edition) 和 J2ME (Java Platform, Micro Edition) 三种平台,2005 年,Java 的三种版本被相应地更名为:Java SE、Java EE 及 Java ME。

● Java SE

Java SE 允许开发和部署在桌面、服务器、嵌入式环境和实时环境中使用的 Java 应用程序。Java SE 是基于 Java 跨平台技术和强有力的安全模块而开发的,其最新的特征和功能极大地提高了 Java 语言的伸缩性、灵活性、适用性以及可靠性。

● Java EE

Java EE 是 Sun 公司针对 Internet 环境下企业级应用推出的一种全新概念的模型,比传统的互联网应用程序模型更有优势,适合于开发服务器端应用程序或者大型 ERP 系统等。Java EE 也是一组规范集,其中的每一个规范规定了 Java 技术应该如何提供一种类型的功能,为应用开发与企业应用集成定义了数目众多的应用编程接口 (API) 和多种编程模型。

● Java ME

Java ME 可以使用在各种各样的消费电子产品上,例如,智能卡、手机、PDA、电视机顶盒等方面。Java ME 也提供了 Java 语言一贯的特性——跨平台和安全网络传输。随着 3G 及嵌入式芯片技术的发展,基于 Java 的移动式、嵌入式应用将会越来越广泛。

以上三种版本中,作为桌面环境下应用开发工具的主要是 Java SE,它是 Java EE 和 Java ME 的基础。关于 Java 的最新发展,有兴趣的读者可以参考 Java 网站 <http://java.sun.com>。

1.2.3 Java 的特点

Java 的迅速发展和广泛流行要归功于它所具有的基本特点。Sun 公司在 Java 语言白皮书

中将 Java 的特点归纳为, Java 是简单的、面向对象的、分布式的、解释型的、健壮的、安全的、结构中立的、可移植的、高效的、多线程的、动态的等。下面对其中的主要特点进行简要的解释。

1. Java 是简单的

Java 的简单性是指其使用与学习时, 与同为面向对象程序设计语言的 C++相比, Java 要简单一些。在 C++中有许多容易混淆的概念, 或者被 Java 抛弃, 或者以一种更容易理解的方式实现。例如, 在 Java 中不再有指针的概念。

2. Java 是面向对象的

传统的程序设计语言是面向过程的, 编程时关注的是程序的控制结构、数据的处理过程等。基于对象的编程更加符合人的思维模式, 编程时只需关注应用程序的数据和处理数据的方法, 并将数据及其处理方法封装在对象之中, 使得编写程序更加容易, 效率也更高。

3. Java 是结构中立的

Java 推出之初, 最能吸引编程人员的就是其体系结构中立 (architecture-neutral) 的特点, 又称为平台无关性, 它是指 Java 应用程序与体系结构无关, 不用修改就可以在不同体系结构的硬件平台上运行。实际运行时, 首先将 Java 源代码编译成字节码 (bytecode)。字节码是结构中立的, 可以运行在 Java 虚拟机上, 与具体的体系结构无关, 如图 1-1 所示。字节码文件可以在任何具有 Java 解释器的平台 (即 Java 虚拟机) 上运行。

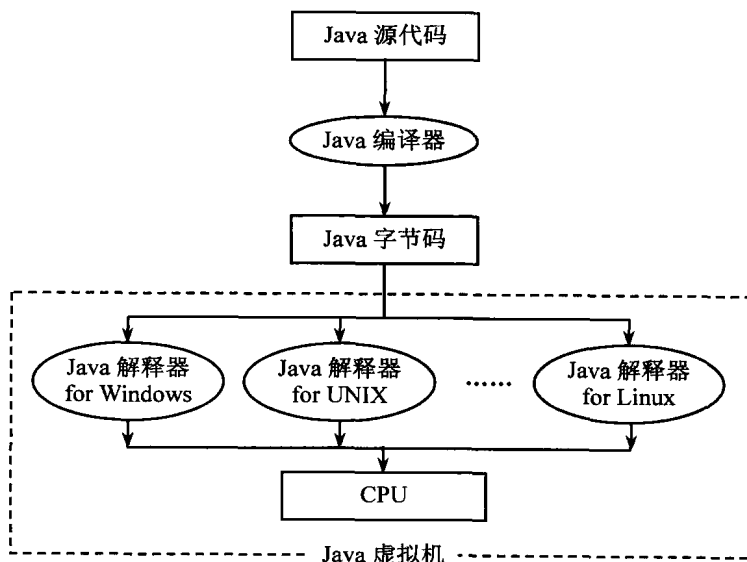


图 1-1 Java 解释器及其平台无关性

4. Java 是安全的

Internet 的安全总是让用户不放心。作为 Internet 程序设计语言, Java 用于网络与分布式环境。因为 Java 执行多层安全机制用以保护系统不受恶意程序破坏, 所以当下载并运行一个 Java applet 时, 它不会损坏本地系统。

5. Java 是可移植的

可移植性是与平台无关性联系在一起的, Java 程序不必重新编译就可以在任何平台上运

行,从而具有很强的可移植性。

6. Java 是开放的

除了以上特点外,Java 的开放性也是促进其快速发展的重要因素。由于 Sun 采取了开放源码策略,在带动 Java 及相关开发技术迅速发展的同时,也使得基于 Java 的开源软件技术成为一种软件开发模式。

1.2 Java 开发环境与应用程序举例

Java 程序分为两类,Java application (应用程序)和 Java applet (小程序)。应用程序是独立的程序,与其他的高级语言程序相同,能够在任何具有 Java 解释器的计算机上运行。applet 是一种特殊的 Java 程序,通常称之为小程序,它可以在兼容 Java 的 Web 浏览器中直接运行,适合开发 Web 程序。

1.2.1 Java 程序开发环境

程序设计语言都有严格的使用规则(语法规则),编写程序时必须遵守这些规则,否则就不能被正确的编译并运行。不同的程序设计语言有不同的开发环境,其语法规则也有差异。在开始学习或者使用 Java 之前,要先准备好基本的环境。

1. Java 开发工具包的来源

一个 Java 程序的开发要经过编辑、编译和运行三个过程。Java 程序的编辑可以使用任何一个文本编辑器,编译与运行则通过 Sun 公司提供的 Java 开发工具箱 JDK (Java Development Toolkit) 进行。JDK 是一个简单的命令行工具集,包括软件库、编译 Java 源程序的编译器、执行字节码的解释器以及其他的一些实用工具。Sun 公司一直在升级 JDK,并免费提供给用户,最新的 JDK 版本请参看 Sun 公司网站。JDK 软件包中包括 JRE (Java Runtime Environment) 以及用于编译调试 Java 程序的命令行界面的开发工具。JRE 在运行 Java 程序时是必需的,可以单独安装,JDK 软件包可以从 Sun 公司网站<http://java.sun.com>下载。

2. 安装 JDK

软件下载后需要安装,基本安装过程如下例所示。

例 1.1 下载并安装最新版本的 JDK。

① 访问<http://java.sun.com>, 下载 JDK (以 jdk-6u20-windows-i586.exe 为例) 软件包并将其存储在硬盘上。

② 双击文件名,开始安装过程。首先在屏幕上显示软件许可证协议,选择接受协议条款继续下面的安装过程。

③ 选择安装的项目及安装位置。可以选择的安装项目包括开发工具、演示程序及样例、源代码及公共 JRE,如图 1-2 所示。开发工具是必须安装的;演示程序及样例提供了一些示例,为用户编写程序提供参考;源代码是 API 的代码,可以帮助我们了解使用的 API 是如何编写及如何工作的;JRE 也是运行 Java 程序必需的;因此建议这 4 项全部安装。

默认的安装位置是“C:\Program Files\Java\jdk1.6.0\”。如果要改变安装位置,单击“更改”按钮重新设置,例如,将 JDK 安装到 C:\jdk1.6。

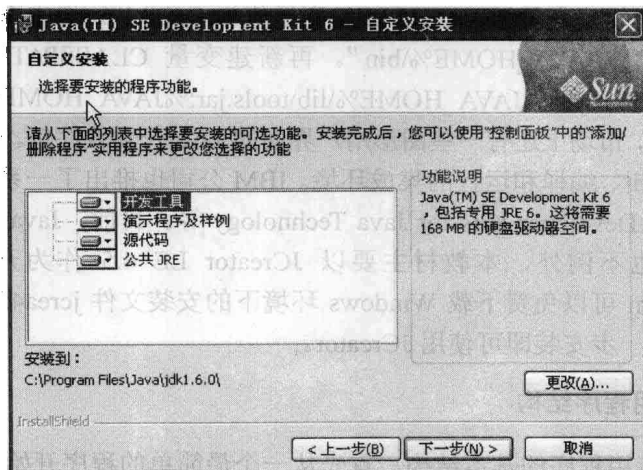


图 1-2 选择 JDK 安装组件

上述安装过程完成后，接着安装公用 JRE，根据屏幕提示选择即可。

3. JDK 环境变量设置

JDK 提供的是命令行用户界面，为了保证在用户工作目录下能够正常调用 Java 编译器，需要在操作系统中进行路径设置。如果使用图形用户界面的集成开发环境，在安装软件时会自动设置环境变量，这一步可以省略。

例 1.2 在 Windows 中设置环境变量，以保证 Java 编译器的正常工作。

分析：设置环境变量就是在 Path 变量值中加入 JDK 的路径。新建一个变量 JAVA_HOME，变量值为 JDK 的具体安装位置；再新建变量 CLASSPATH 设置类路径，使 Java 程序可以方便地调用 JDK 中的类。具体操作方法如下：

- ① 右击“我的电脑”，选择“属性”，打开“我的电脑”对话框。
- ② 在显示的对话框中选择“高级”选项卡，单击“环境变量”按钮，屏幕显示如图 1-3 所示的环境变量对话框。

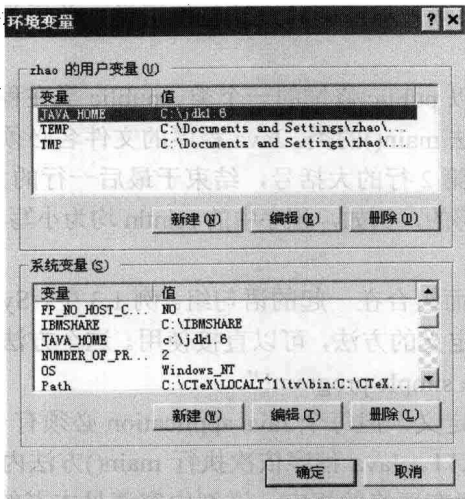


图 1-3 设置路径

③ 在系统变量中新建变量 `JAVA_HOME`, 变量值为 `C:\jdk1.6`。选中系统变量中的 `Path` 项, 在其变量值中增加 “;%`JAVA_HOMECLASSPATH`, 将其变量值设为 “;%`JAVA_HOMEJAVA_HOMEJAVA_HOME`

除了 Sun 的 JDK, 市场上还有一些图形用户界面的 Java 开发软件包。例如 Eclipse、JCreator 和 JBuilder 等具有编辑、编译和运行的集成环境。IBM 公司也推出了一系列的 Java 开发工具, 主要有 WebSphere 及 Developer Kits for Java Technology 等。目前, Java 的开发常使用集成开发环境, 在教学中也不例外, 本教材主要以 JCreator LE 4.5 作为开发环境。通过网址 <http://www.jcreator.com> 可以免费下载 Windows 环境下的安装文件 `jcrea450_setup.exe`。双击该文件, 根据提示一步一步安装即可使用 JCreator。

1.2.2 Java 应用程序结构

为了理解 Java 应用程序的基本结构, 首先从一个最简单的程序开始讨论, 该程序的功能是在屏幕上显示信息 “This is a simple program!”。

例 1.3 一个简单的 Java 应用程序。

```
//文件名 Simple.java
public class Simple
{
    public static void main(String[] args)
    {
        System.out.println("This is a simple program!");
    }
}
```

一般情况下, 一个 Java 应用程序由类、对象与方法等若干部分组成。下面结合本程序对这些组成部分进行简单的解释。

1. 类

类是面向对象程序设计中的基本概念, 也是 Java 应用程序的基本组成单位。所有 Java 程序都是由一个或者多个类组成的, 一个 Java 程序至少包含一个类, 子程序都包含在类定义的块内。要编写 Java 程序, 必须能够理解并设计与使用类。关于类的进一步讨论将在本书的第 3 章与第 4 章展开。

例 1.3 中, Simple 就是以 `public` 修饰的一个类。`public` 类在程序中最多只有一个, 又被称为程序的主类, 即含有主方法 `main()` 的类。Java 程序的文件名必须是这里的 `public` 修饰的主类名。本例中, 类定义开始于第 2 行的大括号, 结束于最后一行的大括号。

注意, Java 是大小写敏感的, 例如, 程序中的 `println` 均为小写, 如果写成 `Println` 就会出错。

2. 方法与 `main()` 方法

方法是为执行一个操作而组合在一起的语句组。例 1.3 中, `System.out.println` 是一个方法, 其中的 `println()` 是 Java 预先定义的方法, 可以直接使用。这个方法的目的是在屏幕上输出字符串, 本例中是输出 “This is a simple program!”。

方法也可以由用户自己定义, 但每个 Java application 必须有一个用户声明的 `main()` 方法, 用来表示 Java 程序的执行入口。Java 程序依次执行 `main()` 方法内的每一条语句, 直到方法的结束。`main()` 方法的定义格式有严格的规定, 详细内容请见本书的第 3 章。

3. 标识符与关键字

编写程序时使用的各种单词或者字符串被称为标识符。关键字是程序设计语言中具有特殊意义的一组标识符，它只能按照预先定义好的方式使用，不能用作其他目的。在 Simple 程序中，出现的标识符有 public、class、Simple、static、void、main、String、args、System、out 和 println，其中 public、class、static、void 是关键字。例如在程序的第 1 行中，当编译器看到 class 时，就知道其后的标识符 Simple 是一个类的名字。关于标识符及关键字的详细描述请参考第 2 章。

有些特定的关键字又被称为修饰符，Java 使用它们来指定数据、方法和类的属性与使用方式。程序中的 public 和 static 就是修饰符。

4. 语句

一条语句表示一个操作，也可以表示一系列操作。程序中，println("This is a simple program!");就是一条语句，在 Java 中，语句用分号“;”结束。

5. 块

将程序中的一些成份组合起来，就构成一个块（block）。在 Java 中，块使用“{”及“}”表示其开始与结束。每个类都有一个组合该类的属性和方法的方法块，每个方法也有一个组合该方法语句的方法块。块也可以嵌套，即一个块可以放到另一个块内，图 1-4 表示了这个关系。

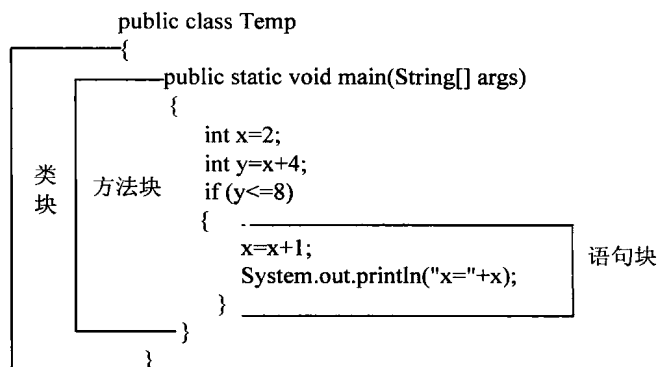


图 1-4 块的示例

6. 注释

为了方便阅读和理解程序，通常的做法是在程序中增加适当的注释。注释语句都是不执行的，编译器在编译源程序时会将其忽略。注释是程序的重要组成部分，一个具有良好风格的程序必须要有清晰而具体的注释。

在 Java 中，注释有两种格式。单行注释用两个斜杠（//）作引导；多行注释用/*和*/将注释的内容括起来。此外，Java 还支持一种称为文档注释的特殊注释，它以/**开头，以*/结尾，主要用于描述类、数据和方法。还可以通过 JDK 的 javadoc 命令转为 HTML 文件，具体请访问 <http://java.sun.com/products/j2se/javadoc/index.html>。

Java 规范中并没有就程序的书写格式提出明确的要求，但为了增加程序的可读性，设计具有良好风格的程序，建议在书写程序时采用缩进格式，即按照程序的层次，下一个层次比上一个层次后退两格。

1.2.3 Java 应用程序开发过程

一个 Java 程序的开发过程如图 1-5 所示，主要包括：编辑、编译和运行。这个过程是反复的，不管是在创建源代码，还是在编译或者运行时，只要有错误，就必须通过修改程序源代码以纠正错误，然后再重新编译或者运行。

例 1.4 请在计算机系统中创建例 1.3 中的源程序文件，并编译运行。

1. 编辑源程序

编辑源代码可以使用任何一个文本编辑器，但源代码文件的文件名必须与程序中定义的公共（public）类的类名相同，扩展名必须是“.java”。使用 JCreator 编辑例 1.3 中的源代码文件，命名为 Simple.java。

打开 JCreator 应用程序，然后选择 File→New，进入图 1-6 所示界面，在图 1-6 中，单击 File Type，选择 Java Classes，然后选择 Java Class，单击 Next 后，可以直接输入文件名 Simple 而无需输入扩展名，若未确定文件类型，则需要输入全名 Simple.java。同时，要求选择存放路径。完成后，在文本编辑窗口输入源代码，结果如图 1-7 所示。

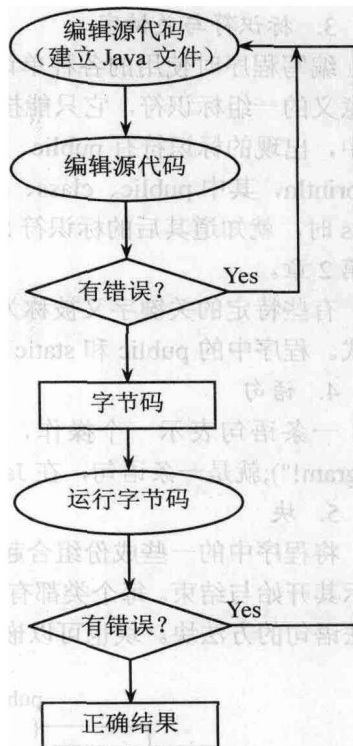


图 1-5 Java 程序开发过程

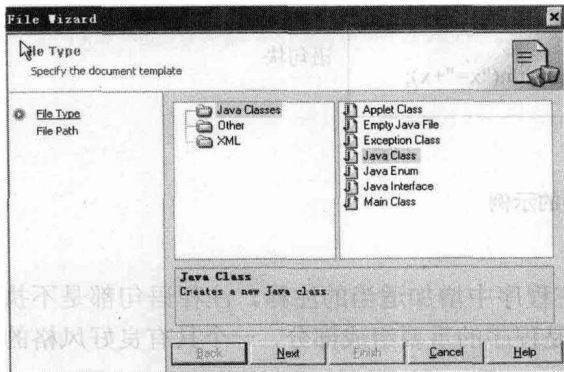


图 1-6 选择文件类型

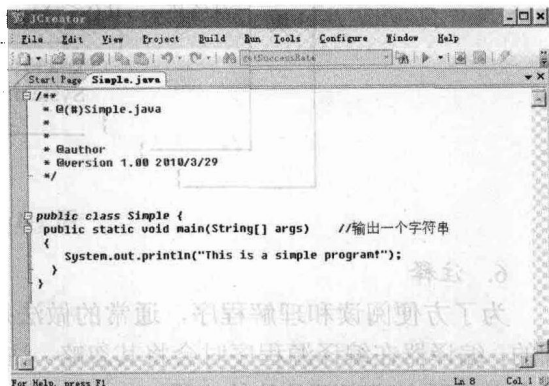


图 1-7 编辑窗口

2. 编译源程序

对一个程序进行编译是指通过编译器 javac.exe 将 Java 源代码文件翻译成字节码（byte code）文件。JCreator 中也集成了编译器，单击菜单 Build→Build File，即对文件进行编译。如果源程序代码没有语法错误，编译器就会生成一个名为 Simple.class 的文件，这个文件称为字节码文件。如果有编译错误，系统会给出一些提示，需要对源代码进行修改，并再次进行编译。

3. 运行字节码文件

运行 Java 程序实际上就是运行字节码文件，在任何一个平台上，只要安装了 Java 解释器 `java.exe`，就可以运行字节码。在 JCreator 中运行字节码文件的方法是，单击菜单的 `Run→Run File`，运行结果如图 1-8 所示。

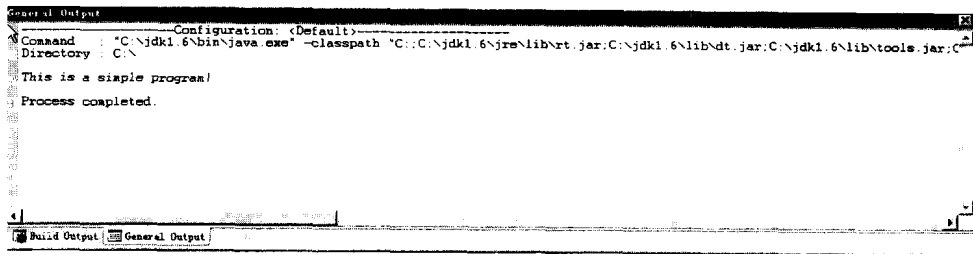


图 1-8 Simple.java 运行结果

在运行字节码时，可能出现运行错误，也可能出现结果不正确的现象，这时需要重新检查并修改源代码文件，将修改后的文件再次编译并运行。

1.3 Java applet 程序举例

Java applet 不是独立的程序，它与 application 既有明显的差异，也有一些共同特征。例如，每个 application 都有一个由 Java 解释器调用的 `main()` 方法，而 applet 则不需要，因为它是直接在浏览器下运行的。下面的例子是一个 applet 程序，它的功能也是显示 “This is a simple applet!”。

例 1.5 一个简单的 applet。

```
//文件名 SimpleApplet.java
import java.awt.Graphics;
public class SimpleApplet extends java.applet.Applet
{
    public void paint(Graphics g)
    {
        g.drawString("This is a simple applet!",15,15);
    }
}
```

这个程序的功能是在浏览器中显示字符信息 “This is a simple applet!”。但是它不能直接运行，需要将它编译成字节码文件，再通过 HTML 文档从浏览器中访问。

Java applet 的运行还需要图形环境，包括文本信息在内的任何内容都要在图形模式下显示。程序中的 `drawString()` 是 Java 标准类 `java.awt.Graphics` 中提供的方法，其功能是在图形环境中显示文本。`import` 语句就是导入 `java.awt.Graphics` 类。通过导入，在例题程序中可以直接使用类 `java.awt.Graphics` 中的方法而不必重新编写程序代码。如果没有这个导入语句，就不能在程序中调用 `drawString()` 方法，这是软件可重用性的一个例子，也是面向对象程序的一大优点。关于 `drawString()` 方法的应用参见第 11 章 `Graphics` 类的介绍。

Java applet 和 Java application 一样也需要经过编译才能运行，但它们的具体执行环境并不一样，下面对 applet 的开发过程进行简单的讨论。