

普.通.高.等.学.校
计算机教育“十二五”规划教材

Java 语言 程序设计

(第2版)

JAVA PROGRAMMING
(2nd edition)

辛运伟 饶一梅 ◆ 编著



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS

普.通.高.等.学.校
计算机教育“十二五”规划教材

Java 语言 程序设计

(第 2 版)

JAVA PROGRAMMING
(2nd edition)

辛运伟 饶一梅 ◆ 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 / 辛运伟, 饶一梅编著. -- 2版

— 北京: 人民邮电出版社, 2015. 12

普通高等学校计算机教育“十二五”规划教材

ISBN 978-7-115-40581-4

I. ①J… II. ①辛… ②饶… 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(2015)第231889号

内 容 提 要

本书是学习 Java 语言的入门读物。全书从 Java 语言的基本特点入手, 全面介绍了 Java 语言的基本概念和编程方法, 同时深入介绍了 Java 的高级特性。本书共分为 12 章, 涉及 Java 中的基本数据类型、基本语法、类的概念及特性、异常处理、用户界面设计、小应用程序、输入/输出操作及线程等内容。

本书内容详尽, 并配合大量示例, 在每章的最后均列出若干习题, 供读者参考。

本书可作为教材供普通高等院校计算机专业的本科生选用, 也可供初学者选用, 还可作为专业人员的参考书。

◆ 编 著 辛运伟 饶一梅

责任编辑 武恩玉

责任印制 沈 蓉 彭志环

◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路 11 号

邮编 100164 电子邮件 315@ptpress.com.cn

网址 <http://www.ptpress.com.cn>

北京昌平百善印刷厂印刷

◆ 开本: 787×1092 1/16

印张: 17

2015 年 12 月第 2 版

字数: 447 千字

2015 年 12 月北京第 1 次印刷

定价: 39.80 元

读者服务热线: (010)81055256 印装质量热线: (010)81055316

反盗版热线: (010)81055315

第 2 版前言

Java 自 1995 年问世以来,始终保持其优良特性,目前已经成为应用范围最广泛的开发语言之一。从桌面办公应用到网络数据库应用,从 PC 到嵌入式移动平台,从 Java Applet 到架构庞大的 J2EE 企业级解决方案,都有 Java 语言的用武之地。这在程序设计语言发展历史上是非常罕见的。

Java 在电子商务、企业级应用开发、网站建设及游戏编程领域非常活跃。学习 Java 语言的学生可选择的就业方向包括 JSP 网站开发、Java 游戏开发、Java 桌面程序设计等。包括电信、银行及保险等行业内,都有使用 Java 语言实现的系统在运行,为企业服务提供支持。

本书是在前一版的基础上修订而成的,主要的修订内容集中在以下几个方面。

(1) Java 语言的原创始者 Sun 公司已被 Oracle 公司收购,所以相关的资料网站已经改变。同时,Java 这几年也推出了更新版本,下载安装 JDK 的过程有所变化。本书全面修订了这部分的内容。

(2) 增加了类的一些例子,在最新的 Java 版本下重新调试了示例代码。规范了类中成员变量访问权限的定义,并相应地增加了访问方法。

(3) 数据流的介绍在第 10 章,这已是书的较后部分了。但在前面若干章的学习过程中,会经常遇到输入/输出问题。为了方便调试程序,在第 4 章增加了输入/输出的简单介绍。

(4) 在第 5 章介绍接口时增加了 Java 提供的 Comparable 接口的介绍,用于对象之间的比较操作。

编者在编写本书过程中,注重汲取国内外优秀教材的特点,结合编者在教学过程中的感受及学生们的反映,精选示例和程序,从 Java 语言的基本特点入手,循序渐进地向读者介绍 Java 语言,同时配合介绍编程的思想和方法。书中的程序均提供了较完整的代码,并在 JDK8.0 下完成了调试。读者可以在自己的机器上尝试运行,得到执行结果。JDK8.0 是目前最新的版本,随着 Java 的不断完善,会有更新的版本问世,读者可随时关注相关网站。

作为教材,在每章的最后列出若干习题,供读者参考。教师可根据各学校的具体课时安排,全部或选讲书中的内容。

在本书的编写过程中,得到了南开大学信息技术科学学院卢桂章教授、陈有祺教授、刘璟教授、周玉龙教授等的亲切关怀和悉心指导,在此一并表示深深的感谢。我们也非常感谢出版社为我们提供的这次机会,感谢选择本书作为教材的教师们和同学们,也感谢读者您在众多的 Java 参考书中选中了本书。

本书由辛运玮、饶一梅共同编写。其中,辛运玮编写了第 1 章至第 7 章,饶一梅编写了第 8 章至第 12 章。最后由辛运玮对全书内容进行审校。由于编者的水平有限,书中难免有错误和不妥之处,恳请广大读者特别是同行专家们批评指正。您的任何意见或建议,都是我们继续修改本书的动力。衷心感谢您的指导!

编 者

2015 年 5 月于南开园

第 1 章 Java 语言简介.....1

1.1 Java 语言简介.....1

1.1.1 Java 语言的问世.....1

1.1.2 Java 语言的组成.....1

1.2 开发环境的安装.....2

1.3 一个简单的 Java 应用程序.....6

1.3.1 Java 应用程序示例.....6

1.3.2 使用 Java 核心 API 文档.....9

习题.....11

第 2 章 面向对象程序设计技术.....13

2.1 面向对象程序设计技术的基本

概念.....13

2.1.1 什么是面向对象程序设计方法.....13

2.1.2 什么是类和对象.....14

2.1.3 面向对象的重要特性.....16

2.2 Java 与 C++ 的 OOP 能力比较.....19

习题.....20

第 3 章 标识符和基本数据类型.....22

3.1 Java 的基本语法单位.....22

3.1.1 空白、注释及语句.....22

3.1.2 关键字.....24

3.1.3 标识符.....24

3.2 Java 编码体例.....25

3.3 Java 的基本数据类型.....26

3.3.1 变量和常量.....26

3.3.2 基本数据类型.....27

3.3.3 变量的说明和赋值.....33

习题.....34

第 4 章 表达式和流程控制语句.....36

4.1 表达式.....36

4.1.1 操作数和运算符.....36

4.1.2 表达式的提升和类型转换.....47

4.2 流程控制语句.....50

4.2.1 表达式语句.....50

4.2.2 块.....52

4.2.3 分支语句.....53

4.2.4 循环语句.....60

4.2.5 break 与 continue 语句.....63

4.3 简单的输入/输出.....65

4.3.1 Scanner 类.....65

4.3.2 NumberFormat 类和 DecimalFormat

类.....67

习题.....68

第 5 章 类与对象.....72

5.1 类的定义与对象的创建.....72

5.1.1 类的定义格式.....72

5.1.2 对象的创建和初始化.....75

5.2 构造方法.....76

5.2.1 构造方法及其重载.....77

5.2.2 默认的构造方法.....80

5.3 定义方法.....81

5.3.1 方法定义格式.....81

5.3.2 按值传递.....85

5.4 类的继承.....86

5.4.1 继承的定义.....87

5.4.2 多态性与转换对象.....89

5.5 继续讨论 Java 的关键字.....91

5.5.1 static.....91

5.5.2 final 和 abstract.....94

5.5.3 this 和 super.....99

5.6 方法重写.....101

5.6.1 方法重写.....102

5.6.2 应用重写的规则.....104

5.6.3 调用父类构造方法.....105

5.7 接口.....106

5.7.1 多重继承中的二义性.....106

5.7.2 接口的定义.....106

5.7.3 接口的实现	107	8.3 布局管理器	180
5.7.4 Comparable 接口	110	8.3.1 常用的布局管理器	181
5.8 Java 包	111	8.3.2 其他的布局管理器	184
5.8.1 Java 包的概念	111	8.4 界面设计的细节	188
5.8.2 import 语句	112	8.4.1 控制组件外观	188
5.9 内部类	114	8.4.2 提示工具和助记符	190
5.9.1 内部类的概念	114	8.5 事件处理	190
5.9.2 匿名类	117	8.5.1 事件简述	190
习题	118	8.5.2 组件的事件处理	194
第 6 章 数组、容器和字符串	125	8.5.3 事件的种类	208
6.1 数组	125	8.5.4 事件适配器	212
6.1.1 数组说明和初始化	125	习题	214
6.1.2 数组的使用	133	第 9 章 Java Applet	216
6.2 容器和字符串类	139	9.1 编写 Applet	216
6.2.1 容器	139	9.1.1 小程序示例	216
6.2.2 字符串类型	140	9.1.2 小程序设计过程	217
习题	145	9.2 小程序中使用的方法	217
第 7 章 Java 语言中的异常	148	9.2.1 基本方法	218
7.1 异常示例	148	9.2.2 用于显示 Applet 的方法	218
7.2 异常处理	152	9.3 HTML 文档	220
7.2.1 异常处理相关语句	152	9.3.1 <applet>标记	220
7.2.2 公共异常	155	9.3.2 Applet 参数的读取	221
7.3 抛出语句	157	9.3.3 Applet 与 URL	222
7.4 创建自己的异常	159	9.4 在 Applet 中的多媒体处理	223
习题	160	9.4.1 在 Applet 中显示图像	223
第 8 章 图形用户界面设计	164	9.4.2 在 Applet 中播放声音	224
8.1 GUI 元素	164	9.5 Applet 的事件处理	224
8.1.1 AWT 包与 Swing 包	165	习题	226
8.1.2 组件、容器及内容窗格	166	第 10 章 Java 数据流	228
8.2 Swing 组件	168	10.1 数据流的基本概念	228
8.2.1 按钮	168	10.1.1 输入数据流	229
8.2.2 标签	171	10.1.2 输出数据流	229
8.2.3 组合框	173	10.2 基本字节数据流类	229
8.2.4 文本组件	175	10.2.1 文件数据流	229
8.2.5 列表	175	10.2.2 过滤器数据流	231
8.2.6 菜单组件	178	10.3 基本字符流	232
8.2.7 对话框	179	10.4 文件的处理	237
		10.4.1 File 类	237

10.4.2 随机访问文件	239
习题	239
第 11 章 线程	241
11.1 线程和多线程	241
11.1.1 线程的概念	241
11.1.2 线程的结构	242
11.2 线程的状态	242
11.3 创建线程	243
11.3.1 继承 Thread 类	243
11.3.2 实现 Runnable 接口	245
11.4 线程的控制	247
11.4.1 线程的启动	247
11.4.2 线程的调度	248
11.4.3 挂起线程	249
11.4.4 线程间的通信	250
习题	251

第 12 章 Java 的网络功能	254
12.1 概述	254
12.1.1 基本概念介绍	254
12.1.2 使用 InetAddress	255
12.2 统一资源定位器	256
12.2.1 URL 的概念	256
12.2.2 URL 的创建	257
12.2.3 与 URL 相关的异常	257
12.2.4 获取 URL 对象属性	257
12.2.5 读入 URL 数据	258
12.3 Socket 接口	259
12.3.1 Socket 的基本概念	261
12.3.2 Socket 通信的基本步骤	262
12.3.3 Socket 通信的程序设计	262
习题	264
参考文献	265

第 1 章

Java 语言简介

程序设计语言是程序员编制程序完成某个任务的必备工具之一。自计算机问世以来,出现过很多的编程语言,Java 属于高级程序设计语言范畴,是最具代表性的语言之一。本章简要介绍 Java 语言的产生背景,并以一个简单程序为例,介绍执行程序时的关键步骤。

1.1 Java 语言简介

Java 语言是目前主流的高级程序设计语言之一,越来越多的应用是用 Java 开发的。

1.1.1 Java 语言的问世

1991 年,美国 Sun Microsystems 公司的 Jame Gosling、Bill Joe 等人为在电视、烤箱等家用消费类电子产品上进行交互式操作而开发了 Oak 语言,之后 Sun 的开发人员又将 Oak 改名为 Java。在 1995 年 Sun 向公众推出 Java 时曾引起业界的巨大轰动,互联网的出现又给 Java 带来了勃勃生机。Java 语言面向网络应用,其类库不断丰富,性能不断提高,应用领域也不断拓展,已成为当今最通用、最流行的软件开发语言之一,是许多专业人员首选的开发语言。

1.1.2 Java 语言的组成

Java 语言是一种全新的高级程序设计语言,它代表了一种新的计算模式。它既是开发环境,又是应用环境。Java 语言包含了 4 个方面:面向对象的程序设计语言、Java 虚拟机 (Java Virtual Machine, JVM)、与机器无关的二进制格式类文件及完整的软件开发包。

1. 面向对象的程序设计语言

Java 是一种面向对象的程序设计语言。它所有的语句都包含在类的定义中。这样的设计方式代表了程序设计的主流趋势。它的语句格式非常类似于 C++ 语言风格,所以熟悉 C++ 语言的程序员可以很快地过渡到 Java 语言。另外,Java 又是非常严格的强类型语言,它去掉了 C++ 语言中容易出错的语法成分。例如,Java 中没有指针、结构和类型定义等概念,不再有全局变量,没有 `#include` 和 `#define` 等预处理器,也没有类的多重继承机制。C++ 语言中经常使用的释放空间的语句,在 Java 中也交给系统自动处理。这些改变使得设计 Java 程序时更加简单,对内存的控制也更加得心应手。Java 的系统对程序进行严格的类型检查,所以程序运行起来更加安全和稳定。Java 还有一个显著的特点,那就是它的解释器只占用很少的内存,适合在绝大多数类型的机器上运行。

使用 Java 语言便于开发网络应用,它内置了 TCP/IP、HTTP、FTP 等协议类库。Java 程序在

语言定义阶段、字节码检查阶段及程序执行阶段进行的三级代码安全检查机制，对参数类型匹配、对象访问权限、内存回收、Java 小应用程序的正确使用等都进行了严格的检查和控制，可以有效地防止非法代码的侵入，阻止对内存的越权访问，能够避免病毒的危害。

2. Java 虚拟机

使用高级程序设计语言编写的代码并不能直接在计算机上运行，代码需要经过编译过程、链接过程等的特殊处理，生成可执行文件，也就是机器能够识别的机器码，然后才能正常运行。在 Java 中，这个处理过程由 Java 虚拟机（Java Virtual Machine, JVM）来执行。JVM 将 Java 源程序转为字节码（bytecode）文件。字节码文件又称为类文件，文件扩展名为.class。之后 JVM 读取类文件来执行。JVM 的实现叫作 Java 运行时系统或运行时环境（Runtime Environment），简称为运行时。

Java 虚拟机是运行 Java 程序必不可少的机制。Java 虚拟机规范中给出了 JVM 的定义：JVM 是在一台真正的机器上用软件方式实现的一台假想机。JVM 的某些指令很像真正的 CPU 指令，包括算术运算、流控制和数组元素访问等。JVM 的具体实现包括指令集（等价于 CPU 的指令集）、寄存器组、类文件格式、栈、垃圾收集堆、内存区。JVM 既可以使用软件方式实现，也可以使用硬件方式实现。

可以将 JVM 看作是编译后的 Java 程序和硬件系统之间的接口，JVM 屏蔽了硬件平台的差异性，正因为有了这个机制，Java 做到了“一次编写，到处运行”，具备了平台无关性的特性。这才是 JVM 的奇妙之处。

3. 类文件格式

计算机所用的芯片不同，安装的操作系统也会有所不同，因此同一种软件产品往往只能安装并运行在某一类或某几种计算机上，这就是我们通常所说的平台依赖性。程序员在编写软件时，也会根据所运行机器的独特性，在代码一级做特殊处理。所以程序移植时非常困难，这是困扰编程人员多年的大问题。Java 语言很好地解决了这个问题。

在 Java 中，编译后得到的文件称为类文件。由于有 JVM 的存在，这个类文件可以采取统一的格式而不依赖于硬件平台。同时，由于 Java 是强类型语言，因而程序中所有类型的长度都是固定的，不会依系统的不同而有所变化。凡是与机器相关的部分都交给 JVM 去处理，而在代码一级是完全不需要考虑机器平台差异的，所以，Java 程序是与平台无关的。

4. 丰富的 API 文档和类库

Java 为用户提供了详尽的应用程序编程接口（Application Programming Interface, API）说明文档。Java 开发工具包（Java SE Development Kit, JDK）中的类库包罗万象，程序员的开发工作可以在一个较高的层次上展开。这也正是 Java 受欢迎的重要原因之一。

1.2 开发环境的安装

要编写一个 Java 应用程序，必须先安装并配置好开发环境。最基本的开发环境包括一个编辑软件及开发 Java 程序必需的 JDK 工具包。编辑软件可以使用机器上现有的任何一个文本编辑器，如记事本、写字板、UltraEdit 等。读者也可以安装集成开发环境（Integrated Development Environment, IDE）。比较著名的 IDE 有 Java WorkShop、Jbuilder（Borland 公司）、Visual J++（Microsoft 公司）、Visual Age for Java（IBM 公司）及 Eclipse 和 Eclipse 的加强版 WSAD。但建议读者在熟悉 JDK 之后再慢慢过渡到使用 IDE，毕竟 JDK 是基础。当编写大型程序时，使用 IDE

编程会更加方便快捷。

这里我们只介绍最基本的 JDK 的安装。

1. 文件下载

JDK 是软件开发工具包, 其中含有编写和运行 Java 程序的所有工具, 我们要用到的工具包括以下几种。

- **javac**: Java 语言编译器, 用来将 java 程序编译成字节码。它读入 Java 源程序, 输出结果为 Java 字节码文件, 即类文件。

- **java**: Java 字节码解释器, 它执行 Java 编译器生成的字节码文件。这是面向 Java 程序的一个独立运行系统。

- **javaAppletViewer**: 小应用程序浏览工具, 用于测试并运行 Java 小应用程序。

- **jdb**: Java 调试器, 用于调试 Java 程序。

- **javap**: 反编译, 将类文件还原回方法和变量。

- **javadoc**: 文档生成器, 创建 HTML 文件。

JDK 还包括 Java 类库 (包括 I/O 类库、用户界面类库、网络类库等)。

运行 Java 程序时, 一定要先安装 JDK, 并正确设置系统的 PATH 和 CLASSPATH 环境变量, 这样系统才能找到 javac (用于编译程序的命令) 和 java (用于执行程序的命令) 所在的目录。

2010 年, Java 最初的发布者 Sun 公司被 Oracle 公司收购, 相关的文档转移至 Oracle 公司相关的网站上。目前最新的 Java 版本是 Java SE Development Kit 8u31, 可从网址 <http://www.oracle.com/technetwork/java/javase/downloads/index.html> 下载, 如图 1-1 所示。



图 1-1 文档下载网页

Java 语言的参考文档是 Java Platform Standard Edition 8 Documentation, 可从网址 <http://docs.oracle.com/javase/8/docs/api/index.html> 在线访问。也可以在图 1-1 所示的界面中, 向下拖动滚动条查找到 Java SE 8 Documentation, 单击 download 按钮下载到本机上, 下载的文件是一个压缩文件, 文件名是 jdk-8u31-docs-all。解压到本机后就可以使用了。

2. 安装 JDK

我们以 Windows 环境为例, 介绍 JDK 的安装过程。

从图 1-1 所示的网站下载的 JDK 文件名是 jdk-8u31-windows-x64 (适用于 64 位操作系统), 将它保存到自己的机器上。下载成功后双击该文件, 启动安装程序, 如图 1-2 所示。

读者可以选择将 JDK 安装在系统设定的默认目录中，也可以安装在自己设定的安装目录中。这两者没有本质差别。



图 1-2 开始安装 JDK



图 1-3 设置安装目录

在图 1-3 所示的窗口中直接单击“下一步”按钮，则 JDK 将安装在默认的安装目录 C:\Program Files\Java\jdk1.8.0_31 中，如图 1-4 所示。如果在单击“下一步”按钮之前，先单击“更改”按钮，则可选择你自己的安装目录，如 E:\Java，如图 1-5 所示。

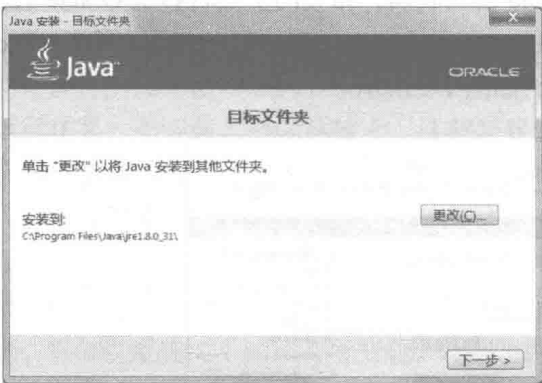


图 1-4 安装在默认目录中

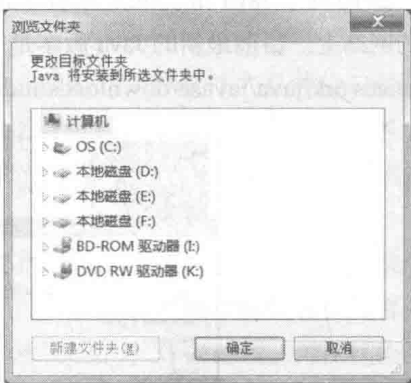


图 1-5 选择安装目录

安装目录选好后，进入安装阶段，如图 1-6 所示。安装完成，将弹出结束界面，如图 1-7 所示。至此，JDK 安装完毕。

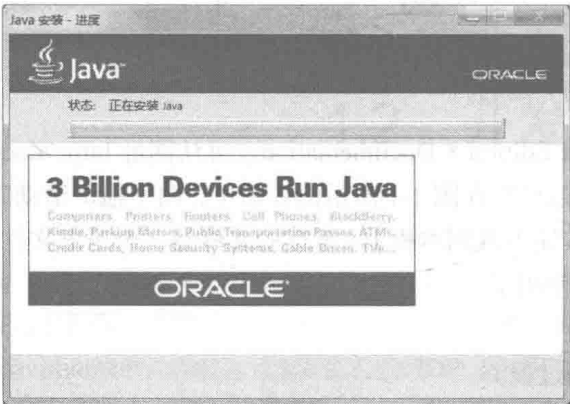


图 1-6 安装进程中



图 1-7 安装完成

安装 JDK 后，我们来检验是否安装成功。

进入系统命令提示符下，输入 java-version，应该显示目前系统中安装的 Java 版本号，如图 1-8 所示。如果能看到这样的信息，恭喜，说明 JDK 已正确安装到你的机器上了。

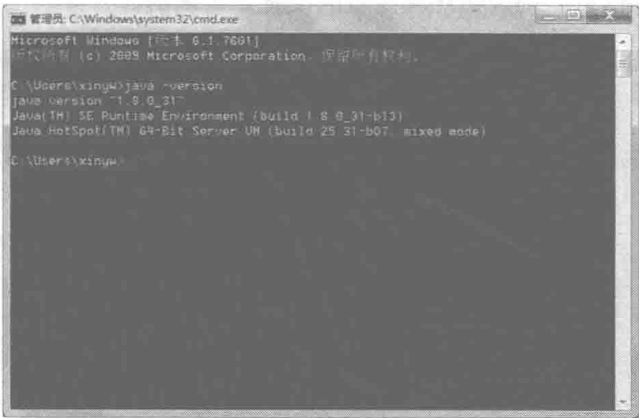


图 1-8 查询 Java 版本号

3. 设置环境变量

JDK 安装成功后，在能使用 Java 编写程序之前还需要进行一步重要的工作，就是设置环境变量。

在桌面上右键单击“计算机”→“属性”→“高级系统设置”→“环境变量”，将弹出图 1-9 所示的窗口。

在“用户变量”中，单击“新建”按钮，将弹出“新建用户变量”窗口，如图 1-10 所示。在“变量名”文本框中输入“JAVA_HOME”，在“变量值”文本框中输入 JDK 的安装路径（系统默认的安装目录或是你自己设定的安装目录），单击“确定”按钮，则设定完毕。



图 1-9 设置环境变量窗口

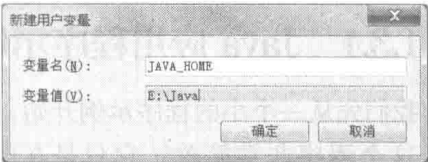


图 1-10 新建用户变量窗口

接下来，在“系统变量”中，选择已有的变量 Path（如果不存在，则新建该变量），单击“编辑”按钮，在“编辑系统变量”窗口中，修改它的值。这个窗口类似于图 1-10 所示的窗口。在“变量值”文本框的最后，添加“;%JAVA_HOME%\jre\bin;%JAVA_HOME%\bin;”，单击“确定”按钮。

接下来, 仍在“系统变量”中, 新建 CLASSPATH 变量。在“变量值”文本框中输入“.;%JAVA_HOME%\lib\dt.jar;%JAVA_HOME%\lib\tools.jar;”。

好了，环境变量已经设置完毕，现在来检验一下。在图 1-8 所示的命令提示符下，输入“javac”，如果显示图 1-11 所示的信息，则表明环境变量的设置是正确的。



图 1-11 检查环境变量的设置

1.3 一个简单的 Java 应用程序

本书主要介绍两种类型的 Java 程序。一类称为 Java 应用程序 (Java Application)，通过 Java 解释器控制，在命令行下使用 `javac/java` 命令来运行；另一类是 Java 小应用程序 (Java Applet)，或称为 Java 小程序。Java 小程序不能独立运行，而是被嵌入 Web 页面中，由 Java 兼容浏览器控制执行。如果计算机上安装了常见的那些 IDE 环境，则这两种类型的程序都可以在 IDE 中运行。

本书的前几章先介绍 Java 应用程序，第 9 章介绍小应用程序。除非特别指明，本书中“程序”一词是指应用程序。

1.3.1 Java 应用程序示例

我们先从一个小的程序示例开始，了解程序的编写、运行过程。这个程序非常简单，它只是在屏幕上显示字符串“Hello World!”，包含的是最基本的语句。

使用文本编辑器编写 Java 程序，然后保存文件，源文件的文件扩展名为 .java。在命令行下使用 javac 命令编译这个程序。如果程序正确，会生成类文件，类文件的文件扩展名为 .class。执行程序时，使用 java 命令。程序从创建到执行的过程如图 1-12 所示。

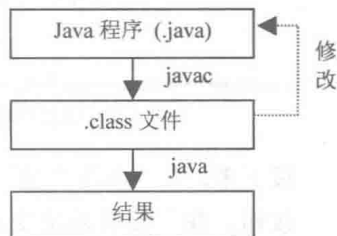


图 1-12 Java 程序的创建、执行过程

1. 程序的创建

可以使用文本编辑器来编写程序，如 Windows 系统中的记事本、写字板等。

在文本编辑器中输入程序 1-1 所示的内容（忽略第一列的数字序号），创建一个文件，文件名为 HelloWorldApp.java。注意保存文件时，不要带有任何的排版信息。例如，使用记事本时，不要保存成文本文档的形式，因为这类文件的扩展名默认情况下是.txt。

Java 是大小写敏感的语言，所以输入程序及为程序命名时都要注意字符的大小写。

Java 是完全的面向对象的语言，编写程序的过程就是定义类的过程。程序 1-1 中定义了一个公有类 HelloWorldApp，因此保存这个程序的文件名为 HelloWorldApp.java，可以将文件放到当前工作目录下。

程序 1-1 一个基本的 Java 应用程序。

```

1 //
2 // 简单的应用程序 HelloWorldApp
3 //
4 public class HelloWorldApp
5 {   public static void main (String args[])
6     {   System.out.println ("Hello World!");
7     }
8 }
```

程序共有 8 行，前 3 行是注释行，对程序进行必要的解释和描述；后 5 行定义了一个公有类 HelloWorldApp，其中只有一个方法 main()。方法中仅含有一条可执行的语句，其功能是在屏幕上打印一个字符串。

程序 1-1 很短，只包含了必要的语句。一个完整的程序大致有以下的内容。

- package 语句：每个文件中只能有 0 个或 1 个该语句，必须放在文件最前面的地方。
- import 语句：语句个数不限，必须放在所有类定义之前。
- 类定义：类有公有类及非公有类。每个文件中，公有类最多有一个，非公有类的个数没有限制。
- 接口定义：每个文件包含的接口定义的个数没有限制。

一个程序中必须至少有一个公有类，包含公有类的这个文件的文件名要与公有类的类名一致，所以每个文件中最多只能有一个公有类。如果一个程序中需要定义多个公有类，则这些公有类必须分散放置在多个文件中。当然一个文件中可以仅包含非公有类。

package 语句与 import 语句都是比较特殊的语句，必须放到所有类定义的前面，且 package 语句需要放到最前面。

例 1-1 语句序列示例。

正确的语句序列：

```

package Transportation;
import java.awt.Graphics;
import java.applet.Applet;
```

错误的语句序列：

```

import java.awt.Graphics;
import java.applet.Applet;
package Transportation;    // 在包说明语句之前含有其他语句
```

错误的语句序列：

```
package Transportation;
package House;
import java.applet.Applet;    // 含有两个包说明语句
```

2. 程序的编译

源文件保存后，使用命令行命令对程序进行编译。Java 的编译程序是 `javac.exe`。编译程序 1-1 的命令如下：

```
javac HelloWorldApp.java
```

如果编译器没有返回任何错误信息，则表示编译成功，源文件正确。编译后，在工作目录下会生成一个新文件 `HelloWorldApp.class`。这是二进制格式的字节码文件。

3. 程序的运行

编译后生成的类文件就可以执行了。Java 采用的是解释执行方式，它的解释器是 `java.exe`，JVM 通过解释器解释执行类文件。执行程序 1-1 的命令如下：

```
java HelloWorldApp
```

执行命令后，会在屏幕上看到一行信息，即“Hello World!”。

4. 程序的解释

下面逐行解释一下程序 1-1 中的代码。

程序的前 3 行是注释行，注释是程序附加的内容，对程序的运行不起任何作用，只用来对程序进行解释说明，帮助读程序的人理解程序。

从第 4 行开始至第 8 行结束是程序的执行语句行，这几行定义了一个完整的公有类，类的名字是 `HelloWorldApp`。类的内容写在一对花括号中。

类中定义了一个方法 `main()`，从第 5 行开始到第 7 行结束。这个方法称为程序的主函数，是执行程序时的主入口，地位非常特殊。凡是应用程序都需要一个 `main()` 方法。`main()` 方法前面有 3 个修饰符，不能缺少，但次序可以有小的变化。例如，可以写成如下形式：

```
static public void
```

这 3 个修饰符有各自独特的含义。`public` 表示方法 `main()` 是公有方法，可以被任何方法访问。`static` 表示 `main()` 方法是静态的，可用在类 `HelloWorldApp` 中，不需要通过该类的实例来调用。如果方法不是静态的，则必须先创建类的实例，然后才能调用实例的方法。有关类和实例的内容请参看本书后面的相关章节。`void` 指明 `main()` 方法不返回任何值。这很重要，因为 Java 要进行严格的类型检查，包括对调用方法所返回的类型和方法说明的类型之间的匹配度检查。如果方法没有返回值，则必须说明为 `void`，不可省略。如果方法有返回值，则以返回值类型替换 `void`。

`main()` 方法中还有一个重要的量，即参数列表 `args[]`。它用来接受命令行参数，即动态传给程序的参数。运行程序时，命令行参数列在程序名之后，以空格为分界。系统将这些参数依次放到 `args` 数组中。该数组各元素是 `String` 类型的，数组的名字可由程序员自己定义，如 `myargs`。数组的下标从 0 开始计数。

下面我们把程序 1-1 稍做修改，让它接受一个命令行参数，并将参数的内容打印出来。

程序 1-2 处理命令行参数。

```
//
// 简单的应用程序 HelloWorldApp
//
public class HelloWorldApp
{
    public static void main (String args[])
    {
        System.out.println ("Hello World!");
        if (args.length!=0) System.out.println("Hello " + args[0]+ "!");
    }
}
```

执行程序 1-2 时，我们给定一个参数，相应的命令如下：

```
java HelloWorldApp earthman
```

屏幕上显示的结果如下：

```
Hello World!
```

```
Hello earthman!
```

这里我们只带了一个命令行参数，系统将这个参数放到 `args[0]` 中。`args.length` 表示命令行参数的个数，运行程序时，系统自动判定命令行参数的个数，并进行赋值。本例中，`args.length` 的值为 1。

程序 1-1 中 `main()` 方法内唯一的可执行语句是输出语句，将一个字符串显示在屏幕上。它调用的是系统标准输出流中的方法。其中 `System` 是系统包 `java.lang` 中的一个类，该类中有成员变量 `out`，这是标准输出流，用来显示信息；`println` 方法接受一个字符串参数，并将其输出到标准输出流中。这行语句反映了类名、对象名和方法调用之间的关系。

5. 程序的调试

程序编写后，难免会存在错误，往往都需要一个调试过程。如果编译或运行时出现错误，则需要按照错误内容提示进一步修改程序，并重新进行编译。找出并修正程序中的错误，重新编译、运行，直到得到正确的结果。

编译阶段发现的错误称为编译时错误，运行阶段发现的错误称为运行时错误。一般来讲，编译时错误很多都是语法方面的问题，如某个标识符拼写错误、括号不匹配或是实例使用的方式不正确等。而运行时错误更多的是逻辑方面的问题，如整除运算时除数为零。当程序中有语法错误时，语法检查就可以提示程序员错误的位置及错误的类型，所以这类错误比较容易发现并修正。运行时错误不是简单的语法问题，在语法检查阶段往往发现不了，必须等到运行时才能发现。

1.3.2 使用 Java 核心 API 文档

JDK 文档中有许多 HTML 文件，这些是 JDK 配套的应用程序编程接口文档，可使用浏览器查看。API 提供了写 Java 程序时会用到的类集合，用来帮助程序员开发自己的类、Applet 和应用程序。程序员使用最多的应该是 Java 核心 API。

核心 API 文档是按层设计的，以主页方式提供给用户。主页按照链接列出包的所有内容。如果选定了一个具体的包，则页面中将列出作为包成员的所有内容。每个类对应为一个链接，选择这个链接将提供该类的信息页。每个类文档有相同的格式，不过，根据具体类的不同，有些内容项可能没有。Java 核心 API 中共有 39 个大包，有的大包中还有或多或少的小包，每个包中都有

个数不等的类和接口，类及接口中又含有若干属性。API 文档中依次列出各类及各接口的相应内容。

Java 提供的内容如此之多，读者不可能在读了一两本 Java 教科书之后就全部掌握。在实际编程时，API 是不可缺少的工具。实际上，Java 正是因其丰富的 API 才获得了如此巨大的成功。

1.2 小节中下载的压缩文件 jdk-8u31-docs-all 就是这个 API 文档。解开压缩文件，单击 api 目录下的 index 文件，会在浏览器中显示 API 的初始页面，如图 1-13 所示。

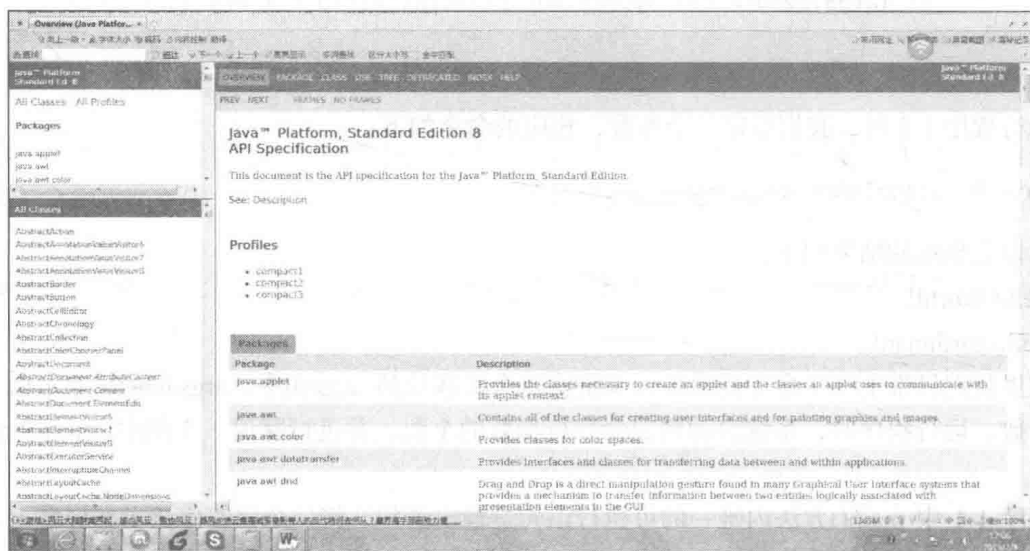


图 1-13 API 文档首页

类文档窗口分为三部分：左上部分显示 Java JDK 中提供的所有包的信息，选中某个包后，将在左下部分显示这个包中所有接口及类的信息。例如，选择查看 java.lang 包，在左下窗口内将显示与这个包相关的内容。

选中包中的某个类，可以在右面看到这个类的具体内容，主要包括类层次结构、类及其一般目的的说明、成员变量表、构造函数表、方法表、变量详细说明表及每一个变量使用目的的详细描述、构造方法的详细说明及进一步的描述、方法的详细说明及进一步的描述。例如，我们选中 Integer 类，则右侧窗口部分将显示 java.lang 中 Integer 类的所有接口及类的内容，如图 1-14 所示。



图 1-14 Integer 类的介绍