



“十三五”普通高等教育本科规划教材

# Java 语言程序设计 实用教程

王素琴 主 编  
周长玉 彭 文 副主编



中国电力出版社  
CHINA ELECTRIC POWER PRESS

扫一扫

看微课



“十三五”普通高等教育本科规划教材

# Java 语言程序设计 实用教程

主 编 王素琴

副主编 周长玉 彭 文

编 写 张智源 韩立涛 刘谕齐 施文豪



中国电力出版社  
CHINA ELECTRIC POWER PRESS

## 内 容 提 要

本书为“十三五”普通高等教育本科规划教材，在全面介绍 Java 语言语法知识的同时，注重对面向对象程序设计思想、Java 核心机制与基本原理的阐述，主要内容包括 Java 语言概述、Java 语言基础、类与对象、异常处理、基于 Swing 的图形用户界面设计、输入/输出流、数据库编程、多线程编程和网络编程。

本书特别注重提高读者运用 Java 语言和面向对象技术解决实际问题的能力。书中给出了大量经过调试运行的程序实例及适量的课后习题，便于读者学习和自测。本书在重点章节设置了二维码，读者可扫描观看相关教学视频或动画。

本书可作为普通高等院校计算机及相关专业 Java 语言程序设计课程的教材，也可供从事计算机工作的技术人员学习或参考。

## 图书在版编目（CIP）数据

Java 语言程序设计实用教程/王素琴主编. —北京：中国电力出版社，2017.2

“十三五”普通高等教育本科规划教材

ISBN 978-7-5198-0230-1

I. ①J… II. ①王… III. ①JAVA 语言—程序设计—高等学校—教材 IV. ①TP312.8

中国版本图书馆 CIP 数据核字（2017）第 003055 号

中国电力出版社出版、发行

（北京市东城区北京站西街 19 号 100005 <http://www.cepp.sgcc.com.cn>）

北京市同江印刷厂印刷

各地新华书店经售

\*

2017 年 2 月第一版 2017 年 2 月北京第一次印刷

787 毫米×1092 毫米 16 开本 19 印张 465 千字

定价 38.00 元



## 敬告读者

本书封底贴有防伪标签，刮开涂层可查询真伪  
本书如有印装质量问题，我社发行部负责退换

版权专有 翻印必究

# 前言

面向对象软件开发方法已经成为计算机应用开发领域的主流技术,它从现实世界客观存在的事物(即对象)出发来构造软件系统,并在其中尽可能运用人类的自然思维方式。采用面向对象方法开发的软件系统具有容易理解、稳定性好、可重用性高等优点。

Java 语言是面向对象程序设计语言的成功典范,自 1995 年诞生以来,短短几年就成为软件开发领域最常用、最重要的语言之一,广泛应用于 Web 应用、移动应用及云计算平台的开发中。

本书在内容的编排上做了精心的设计,注重理论性、实用性和先进性的统一。在准确、深入地介绍 Java 语言基本语法知识的同时,将实用性强的应用程序穿插在理论讲述中。另外,结合开发应用程序的需要,本书还详细阐述了数据库应用程序开发、多线程编程及网络程序设计技术,并配以精心设计的案例及程序。通过本书的学习,读者不仅能够掌握 Java 语言的语法知识,了解面向对象程序设计的基本方法,而且能够提升开发实际应用程序的能力。同时,作为最活跃的程序设计语言之一,Java 语言一直在发展、演化中。本书对 Java 语言的常用新特性进行了详细介绍,包括 JDK 7 中引入的异常的多重捕获及自动资源管理, JDK 8 中引入的函数式接口及 lambda 表达式的使用等。

本书在内容的阐述上自成体系,通俗易懂,从问题的引入到问题的解决,体现了由浅入深、循序渐进的原则。由于在数据库、多线程及网络编程等章节的学习上需要用到数据库、操作系统和计算机网络等课程的基础知识,考虑到 Java 语言课程的开设可能早于这些专业课,因此在各章中加入了专业基础知识的介绍,便于读者的理解和掌握。各章都配有丰富的例题,较复杂的例题都有详细的分析过程和运行结果的说明。各章后面配有多种类型的习题,知识点覆盖全面,便于读者复习和自测。本书在重点章节设置了二维码,读者可扫描观看教学视频或动画。

下面简要介绍本书的主要内容与教学安排:

第 1 章 Java 语言概述,主要介绍 Java 语言的发展历史、语言特点、平台构成,并以一个简单的程序为例来说明 Java 程序的开发过程及使用的开发工具。

第 2 章 Java 语言基础,介绍 Java 语言的基础知识,包括标识符、数据类型、变量、运算符、表达式、流程控制、数组、字符串和输入/输出等。

第 3 章 类与对象,系统介绍 Java 语言中面向对象程序设计的基本概念和基本方法,重点是封装、继承和多态三大特性的实现过程。

第 4 章 异常处理,介绍异常的概念、异常类、捕获异常、声明异常、异常处理机制及自定义异常类等。

第 5 章 基于 Swing 的图形用户界面设计,主要介绍 Java 图形用户界面设计的基本原理、常用的组件、布局管理器和事件处理机制等。

第 6 章 输入/输出流,介绍流的基本概念、I/O 类的体系、文件流、缓冲流、数据流、对象流、桥接流等。

第7章 数据库编程，首先介绍 Java 数据库连接应用编程接口 JDBC 的相关概念及结构化查询语言 SQL，然后详细阐述了使用 JDBC 技术开发数据库应用程序的基本方法和过程。

第8章 多线程编程，首先介绍 Java 多线程机制的基本概念，然后重点阐述了线程的创建、调度、同步控制及线程之间的通信等。

第9章 网络编程，首先介绍网络编程相关的基本概念，然后进一步介绍如何编写连接网络服务的 Java 程序，重点介绍基于连接的 Socket 网络通信程序设计。

第1章～第6章是 Java 基础篇，第7章～第9章是 Java 应用篇，在教学中可根据实际情况选用。

本书第1章～第3章由彭文编写，第4章～第6章由王素琴编写，第7章～第9章由周长玉编写。高宇豆、王金睿、张智源、韩立涛、刘谕齐和施文豪参与了内容的校对、例题和习题的编写及程序的调试工作。

限于作者水平，书中难免存在疏漏之处，欢迎各位同行和广大读者批评指正。

编 者

2016年12月

# 目 录

## 前言

|                   |    |
|-------------------|----|
| 第1章 Java语言概述      | 1  |
| 1.1 Java语言的发展历史   | 1  |
| 1.2 Java语言的运行原理   | 1  |
| 1.3 Java语言的特点     | 2  |
| 1.4 Java语言平台      | 3  |
| 1.5 Java程序开发      | 4  |
| 1.6 Eclipse的安装与使用 | 6  |
| 本章小结              | 8  |
| 习题                | 8  |
| 第2章 Java语言基础      | 9  |
| 2.1 标识符与数据类型      | 9  |
| 2.2 运算符与表达式       | 14 |
| 2.3 流程控制          | 18 |
| 2.4 数组            | 27 |
| 2.5 字符串           | 34 |
| 2.6 输入/输出         | 38 |
| 本章小结              | 41 |
| 习题                | 41 |
| 第3章 类与对象          | 42 |
| 3.1 面向对象基本概念      | 42 |
| 3.2 类的封装          | 43 |
| 3.3 类的继承          | 59 |
| 3.4 类的多态          | 64 |
| 3.5 抽象类           | 70 |
| 3.6 接口            | 72 |
| 3.7 内部类           | 76 |
| 3.8 Java语言中的常用类   | 79 |
| 本章小结              | 91 |
| 习题                | 92 |
| 第4章 异常处理          | 94 |
| 4.1 异常的概念         | 94 |

|              |                                 |            |
|--------------|---------------------------------|------------|
| 4.2          | 异常类及异常处理方式 .....                | 95         |
| 4.3          | 捕获异常 .....                      | 97         |
| 4.4          | 声明异常 .....                      | 103        |
| 4.5          | Java 异常处理机制 .....               | 107        |
| 4.6          | 人为抛出异常 .....                    | 108        |
| 4.7          | 自定义异常类 .....                    | 109        |
|              | 本章小结 .....                      | 112        |
|              | 习题 .....                        | 112        |
| <b>第 5 章</b> | <b>基于 Swing 的图形用户界面设计 .....</b> | <b>114</b> |
| 5.1          | Java 图形用户界面基础 .....             | 114        |
| 5.2          | 常用的组件与容器 .....                  | 116        |
| 5.3          | 布局管理器 .....                     | 124        |
| 5.4          | 事件处理 .....                      | 134        |
| 5.5          | 选项类组件 .....                     | 150        |
| 5.6          | 对话框 .....                       | 159        |
| 5.7          | 菜单 .....                        | 162        |
| 5.8          | 工具栏 JToolBar .....              | 167        |
| 5.9          | 表格 JTable .....                 | 170        |
| 5.10         | 树 JTree .....                   | 178        |
|              | 本章小结 .....                      | 182        |
|              | 习题 .....                        | 183        |
| <b>第 6 章</b> | <b>输入/输出流 .....</b>             | <b>185</b> |
| 6.1          | 流的概念 .....                      | 185        |
| 6.2          | I/O 类体系 .....                   | 186        |
| 6.3          | 文件流 .....                       | 190        |
| 6.4          | 实体流和装饰流 .....                   | 203        |
| 6.5          | 缓冲流 .....                       | 203        |
| 6.6          | 数据流 .....                       | 206        |
| 6.7          | 对象流与对象序列化 .....                 | 210        |
| 6.8          | 标准输入/输出 .....                   | 216        |
| 6.9          | 桥接流 .....                       | 217        |
| 6.10         | 流的关闭 .....                      | 218        |
|              | 本章小结 .....                      | 220        |
|              | 习题 .....                        | 220        |
| <b>第 7 章</b> | <b>数据库编程 .....</b>              | <b>222</b> |
| 7.1          | 数据库概念及 SQL 语句 .....             | 222        |
| 7.2          | Java 访问数据库技术 .....              | 229        |
| 7.3          | Java 数据库应用程序开发 .....            | 242        |
|              | 本章小结 .....                      | 247        |

|                          |     |
|--------------------------|-----|
| 习题 .....                 | 248 |
| <b>第 8 章 多线程编程</b> ..... | 249 |
| 8.1 线程的概念 .....          | 249 |
| 8.2 实现多线程的方式 .....       | 253 |
| 8.3 线程控制的基本方法 .....      | 258 |
| 8.4 线程的同步机制 .....        | 263 |
| 8.5 线程之间的通信 .....        | 269 |
| 8.6 定时器类 Timer 的应用 ..... | 274 |
| 本章小结 .....               | 276 |
| 习题 .....                 | 276 |
| <b>第 9 章 网络编程</b> .....  | 278 |
| 9.1 网络通信基础 .....         | 278 |
| 9.2 Socket 通信机制 .....    | 282 |
| 本章小结 .....               | 294 |
| 习题 .....                 | 294 |
| <b>参考文献</b> .....        | 296 |

扫一扫 观看视频

二维码 总码





## 第1章 Java 语言概述

Java 是一种可以编写跨平台应用程序的面向对象程序设计语言，在软件开发领域得到了广泛应用。本章主要介绍 Java 语言的发展历史、语言特点和平台构成，并以一个简单的程序为例来说明 Java 程序的开发过程及常用的开发工具。

### 1.1 Java 语言的发展历史

1995 年 5 月 23 日，Sun 公司在 SunWorld 大会上第一次公开展示 Java 语言。Java 这个词是位于印度尼西亚的爪哇岛的英文名称，该岛因盛产咖啡而闻名，所以 Java 语言的标识是一杯正冒着热气的咖啡。

1996 年初，Sun 公司发布了 Java 语言的第 1 个版本 Java 1.0，但很快人们就意识到它不能用来进行真正的应用开发。后来的 Java 1.1 弥补了其中的大部分缺陷，改进了反射能力，并为图形用户界面（GUI）编程增加了新的事件处理模型。

1998 年 12 月 Java 1.2 版本发布，这个版本取代了早期玩具式的 GUI，图形工具箱更加精细且具有可伸缩性，更加接近“一次编写，随处运行”的目标。Java 1.2 发布三天后，Sun 公司将其命名为“Java 2 标准版软件开发工具箱 1.2 版”。

标准版的 1.3 和 1.4 版本继续对 Java 2 进行改进，扩展了标准类库，提高了系统性能，修改了部分缺陷。

Java 5.0 版是自 1.1 版以来第一个对 Java 语言做出重大修改的版本（这一版本原来被命名为 1.5 版，在 2004 年的 JavaOne 会议之后，版本数字升为 5.0）。这个版本添加了泛型类型、for each 循环、自动装箱和元数据等新的语言特性。

2006 年末，Sun 公司发布了 Java 6，改进了系统性能，增加了类库。Oracle 公司于 2009 年收购了 Sun 公司，并于 2011 年发布了 Java 的新版本 Java 7，其中做了较多的改进。2014 年 3 月 Java 8 面世，这一版本最大的改进是新增了 Lambda 表达式。

### 1.2 Java 语言的运行原理

Java 语言比较特殊，它既是编译型语言又是解释型语言。Java 的源程序代码必须通过编译器将其编译为 Java 字节码（byte code）后才能运行，所以称之为编译型语言。但 Java 字节码不能直接运行，只能在 Java 虚拟机（具体说明见 1.4.1 节）环境中被解释执行，因此称之为解释型语言。一个 Java 程序的运行过程如图 1-1 所示。



图 1-1 Java 程序的运行过程

Java 程序是由虚拟机负责解释执行，并非是操作系统，这样做的优点是可以实现程序的跨平台运行。也就是说，一段 Java 程序编译后的字节码可以在装有不同操作系统的计算机上运行，只要每台计算机上装有对应的 Java 虚拟机即可，如图 1-2 所示。这种方式使得 Java 语言解决了在不同操作系统编译时产生不同机器代码的问题，大大降低了程序开发和维护的成本。

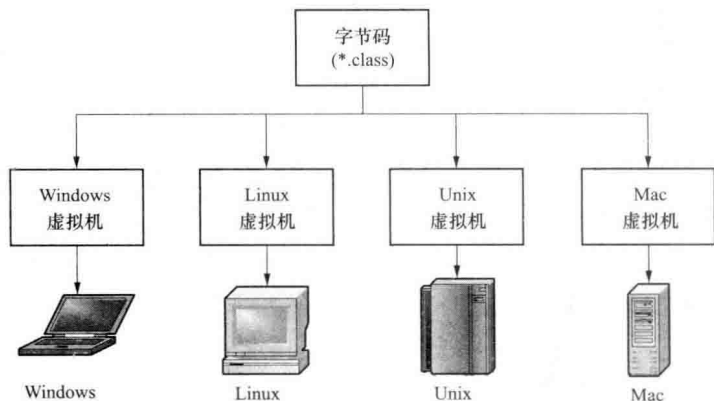


图 1-2 Java 程序的跨平台运行原理

### 1.3 Java 语言的特点

与其他语言（如 C++、Delphi、C#等）相比，Java 语言有着突出的特点，使其受到广泛的关注，主要体现如下：

#### 1. 简单性

Java 语言是一种相对简单的编程语言，其基本语法与 C++ 语言极为相似，如常用的条件语句、循环语句和控制语句等。但 Java 语言摒弃了 C++ 语言中不易理解的部分，减少了编程的复杂性，例如去掉了指针变量、结构体、运算符重载、多重继承等复杂特性。

#### 2. 面向对象

Java 语言是一种纯粹的面向对象程序设计语言，除了基本数据类型以外，都是对象，程序代码以类的形式组织，由类来定义对象的各种属性和行为。Java 语言支持继承机制，减少了程序设计的复杂性。

#### 3. 平台无关性

Java 语言经编译后生成与计算机硬件结构无关的字节码，这些字节码不依赖于任何硬件平台和操作系统。Java 程序在运行时，需要由一个解释程序对生成的字节码解释执行。这体现了 Java 语言的平台无关性，使得 Java 程序可以在任何平台上运行，如 MS-DOS、Windows、Unix 等，因此具有很强的移植性。

#### 4. 安全性

在网络环境中，安全性是个不容忽视的问题。Java 语言在安全性方面引入了实时内存分配及布局来防止程序员直接修改物理内存布局。通过字节码验证器对字节码进行检验，防止网络病毒及其他非法代码侵入。此外，Java 语言还采用了专门的异常处理机制对程序运行过

程中遇到的异常事件进行捕获和处理，如输入错误、内存空间不足、程序异常中止等。

### 5. 分布式

分布式包括数据分布和操作分布。数据分布是指数据可以分散在不同主机之上，而操作分布是指把一个计算任务分散到不同主机上进行处理。对于数据分布，Java 语言提供 URL 对象访问 URL 地址上的资源。对于操作分布，Java 语言提供了用于网络应用编程的类库，开发人员可以进行网络程序设计。此外，Java 语言的远程方法调用（RMI）机制也是开发分布式应用的重要手段。

### 6. 多线程机制

Java 语言支持多线程机制，Java 程序能够并行处理多项任务，可以大大提高程序的执行效率。

### 7. 内存管理机制

Java 语言采用了自动垃圾回收机制进行内存的管理，它可以自动、安全地回收不再使用的内存块。程序员在编程时无需担心内存的管理问题，从而使 Java 程序的编写变得简单，同时也减少了内存管理方面出错的可能性。

## 1.4 Java 语言平台

为了使软件开发人员、服务提供商和设备生产商可以针对特定的市场进行开发，Sun 公司将 Java 划分为三个技术平台，分别为 Java SE、Java EE 和 Java ME。

Java SE（Java Platform Standard Edition，以前称为 J2SE），为开发普通桌面和商务应用程序提供解决方案。Java SE 是三个平台中最核心的部分，包含了 Java 最核心的类库，如集合、流、数据库连接及网络编程等。

Java EE（Java Platform Enterprise Edition，以前称为 J2EE），为开发企业级应用程序提供解决方案。Java EE 是在 Java SE 的基础上构建的，用于开发、装配及部署企业级应用程序，主要包括 JSP、Servlet、JavaBean、EJB 和 Web Service 等技术。

Java ME（Java Platform Micro Edition，以前称为 J2ME），为在移动设备和嵌入式设备上运行的应用程序提供解决方案。Java ME 包括灵活的用户界面、健壮的安全模型、许多内置的网络协议及对可以动态下载的联网和离线应用程序的丰富支持。

Java 语言平台由 Java 虚拟机（Java Virtual Machine, JVM）和 Java 应用编程接口（Java Application Programming Interface, API）两部分组成，它为 Java 源程序的编写、编译和运行提供了完善的环境。

### 1.4.1 Java 虚拟机

Java 虚拟机是运行 Java 程序的虚拟计算机，它解释 Java 字节码并执行代码要完成的操作。Java 虚拟机与计算机硬件和操作系统相关，根据不同的操作系统开发相应的版本。

Java 虚拟机由五部分组成，包括一组指令集、一组寄存器、一个栈、一个无用单元收集堆和一个方法区域。这五部分是 Java 虚拟机的逻辑组成，不依赖于任何实现技术或组织方式，但他们的功能必须在真实计算机上以某种方式实现。

Java 虚拟机的指令包含一个单字节的操作符，用于指定要执行的操作，还有 0 个或多个操作数，提供操作所需的参数或数据。有些指令没有操作数，仅由一个单字节的操作符构成。

Java 虚拟机的寄存器用于保存计算机的运行状态,与微处理器中的某些专用寄存器类似。Java 虚拟机的寄存器有四种,包括 Java 程序计数器、指向操作数栈顶端的指针、指向当前执行方法的执行环境指针和指向当前执行方法的局部变量区第一个变量的指针。

Java 虚拟机的栈有三个区域:局部变量区、运行环境区、操作数区。

Java 虚拟机的堆是一个运行时数据区,类的实例(对象)从中分配空间。Java 语言具有无用单元收集能力,它不给程序员显式释放对象的能力。

Java 虚拟机的方法区域与传统语言中的编译后代码及 Unix 进程中的正文段类似。它保存方法代码(编译后的 Java 代码)和符号表。在当前的 Java 实现中,方法代码不包含在无用单元收集堆中,但计划在将来的版本中实现。

#### 1.4.2 Java API

Java API 是 Java 语言提供的类和接口的集合,目的是为应用程序与开发人员提供一套访问主机系统资源的标准方法,但又无需访问源码或理解内部工作机制细节。Java API 以包的形式组织,表 1-1 列出了 Java API 中常用包的名称及包含内容。

运行 Java 程序时,虚拟机装载程序所使用的 Java API 类文件。所有被装载的类文件和已经装载的动态库共同组成了在 Java 虚拟机上运行的整个程序。

表 1-1 Java API 中常用包的名称及包含内容

| 包名称         | 包 含 内 容           | 包名称         | 包 含 内 容            |
|-------------|-------------------|-------------|--------------------|
| java.applet | 创建 Applet 需要的类和接口 | java.rmi    | 远程方法调用的类和接口        |
| java.awt    | 图形用户界面的类和接口       | java.sql    | 访问数据库的类和接口         |
| java.io     | 输入/输出流的类和接口       | java.text   | 处理文本、日期、数字和消息的类和接口 |
| java.lang   | Java 的基础类和接口      | java.util   | 包含日期、集合等实用的类和接口    |
| java.math   | 与数学计算相关的类和接口      | javax.swing | 轻量级用户界面的类和接口       |
| java.net    | 网络应用的类和接口         |             |                    |

### 1.5 Java 程序开发

为了方便用户开发 Java 程序, Sun 公司提供了一套 Java 开发工具包(Java Development Kit, JDK),它是整个 Java 语言的核心,其中包括 Java 编译器、Java 运行工具、Java 文档生成工具和 Java 打包工具等。此外, Sun 公司还提供了 Java 程序运行时的环境工具 JRE(Java Runtime Environment),Java 虚拟机就包含在其中。为了简化操作流程,在 JDK 中自带了 JRE,这样开发人员只需要在计算机上安装 JDK 即可,不需要专门安装 JRE 工具。

#### 1.5.1 JDK 下载与安装

##### 1. JDK 的下载

在浏览器录入以下网址,进入下载页面,其中有针对不同操作系统的 JDK 版本,选择适用于 Windows 操作系统的最新版本下载即可,本书中的代码是基于 JDK 8 版本编写。

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

##### 2. JDK 的安装

双击已下载好的 JDK 文件,按照安装提示逐步操作,选择默认路径依次安装 JDK 和 JRE

文件。

### 3. 安装文件说明

JDK 安装完毕后, 在安装目录会出现文件夹“JDK 1.8.0\_74”(不同 JDK 版本安装后的文件夹名称不同), 其中包含的内容有:

- (1) bin 目录: 存放 JDK 开发工具的可执行文件。
- (2) lib 目录: 存放开发工具需要的附加类库和支持文件。
- (3) jre 目录: 存放 Java 虚拟机, 可支持执行 Java 语言编写的程序。
- (4) demo 目录: 存放带有源代码的 Java 编程示例。
- (5) include 目录: 存放 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。
- (6) src.zip 压缩文件: 存放组成 Java2 核心 API 的所有类的 Java 编程语言源文件。

### 1.5.2 Java 程序的编写

JDK 安装完毕后, 便可以开始 Java 程序的开发, 主要分为编写源程序、编译、调试和运行等步骤。

打开 Windows 操作系统中的记事本程序, 录入 [例 1-1] 中的代码, 将文件命名为“HelloWorld.java”并保存在 E 盘根目录下。

**【例 1-1】** 输出一行文字“第一个 Java 程序”。

```
/*
 *该程序的功能是在标准输出端
 *打印出一行文字"第一个 Java 程序"
 */
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("第一个 Java 程序");
    }
}
```

各行代码的含义如下:

- (1) 第 1~4 行代码为 Java 的注释语句, 这里采用的是多行注释。
- (2) 第 5 行的 class 是 Java 关键字, 用于定义一个类, public 表示类的访问权限。在 Java 中, 所有的代码都需要写在类中。HelloWorld 是类名, 关键字 class 和类名之间需要用空格、制表符或换行符等空白字符分隔。类名后面要写一对花括号, 定义了这个类的范围, 花括号中的内容也称为类体。Java 规定标记为 public 的类名必须与文件名一致, 所以该程序的文件名必须是“HelloWorld.java”(注意: 大小写敏感)。
- (3) 第 6 行定义了一个 main()方法, 这是 Java 程序的执行入口。根据 Java 语言规范, main()方法必须声明为 public。
- (4) 第 7 行是一条打印语句, 作用是在输出端显示文本“第一个 Java 程序”。其中 System 类是一个特殊的类, 它表示当前运行的系统, out 是 System 类的成员变量, 是标准输出流对象。

在编写程序时, 代码中出现的括号、分号、双引号等符号必须采用英文半角格式, 否则程序会出错。另外, 为了增加程序可读性, 在编写 Java 代码时, 大家都会遵循一定的规范:

(1) 类名：首字母大写，多个单词合成一个类名时，每个单词的首字母要大写，如 HelloWorld。

(2) 接口名：命名规则与类名相同。

(3) 方法名：常由多个单词合成，第一个单词首字母小写，其后每个单词的首字母都要大写，如 showMessageDialog()。

(4) 变量名：一般为名词，第一个单词全小写，其后每个单词首字母大写，如 rowName。

(5) 常量名：基本数据类型的常量名为全大写，如果是由多个单词构成，可以用下划线隔开，如 PERSON\_COUNT。

### 1.5.3 Java 程序的编译与运行（微课 1）

Java 源程序文件编写完毕后，打开 DOS 窗口对程序进行编译与运行。Windows 7 及以下



操作系统，单击“开始”菜单按钮，在“搜索程序和文件”框中输入“cmd”，再按“回车”键即可打开一个 DOS 窗口。Windows 8 系统可在键盘上同时按下 windows+r 的组合键，在弹出窗口中输入“cmd”，然后再按“回车”键，打开 DOS 窗口。

为了操作方便，首先将源文件所在的目录（文件夹）设为当前目录，[例 1-1] 中的源文件在 E 盘根目录下，则需要在 DOS 提示符下直接输入“E:”，单击“回车”键，当前 DOS 提示符变为 E:，如图 1-3 所示。如果源文件不是在 E 盘根目录，而是在 E:\JavaCode 目录下，则首先在 DOS 提示符下输入“E:”，按“回车”键之后，再录入“cd E:\JavaCode”，按“回车”键之后，目录切换成功。

在图 1-3 所示的目录下，输入命令“javac HelloWorld.java”，按“回车”键之后开始编译源文件。一旦编译成功，会在当前目录下生成 HelloWorld.class 文件。Java 程序中的每个类被编译后都会生成一个后缀名为 class 的字节码文件。

运行 Java 程序也相对简单，在上述操作之后，输入“java HelloWorld”命令，按“回车”键之后，在界面上显示文本“第一个 Java 程序”。需要注意的是，录入的命令中，不需要加上“.class”，否则会出现错误。

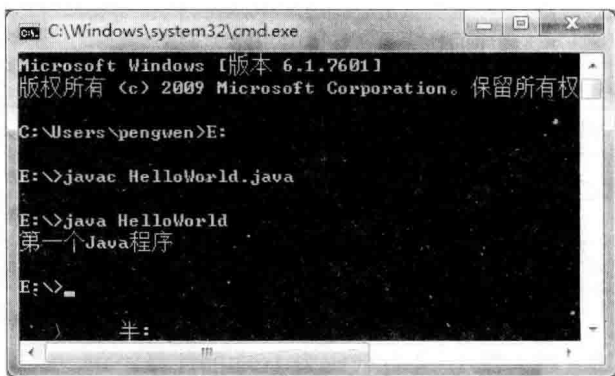


图 1-3 DOS 窗口

## 1.6 Eclipse 的安装与使用

在记事本中编写 Java 程序效率较低，目前市面上出现若干针对 Java 语言的开发平台，其中 Eclipse 是目前最常用的平台之一，它是一个开放源代码的、可扩展的集成开发环境。

### 1.6.1 Eclipse 的下载与安装

#### 1. Eclipse 的下载

Eclipse 的下载网址为 <http://www.eclipse.org/downloads/eclipse-packages/>，在下载页面中列出了不同版本的 Eclipse 的下载选项。需要下载的是 Eclipse IDE for Java Developers，根据

操作系统的不同选择 32 位或 64 位版本，点击下载即可，本书中下载的是 Windows64 bit 版本。

## 2. Eclipse 的安装

双击下载后的文件 eclipse.exe，即可打开 Eclipse。在安装过程中会出现“选择 Workspace”的界面，也就是项目代码存放的位置，本书中工作空间设在“D:\Java”文件夹下。

### 1.6.2 Eclipse 的使用（微课 2）

通过 Eclipse 可以完成 Java 程序的编写、编译、调试和运行，主要步骤如下。



#### 1. 建立项目

选择菜单栏中的 File→New→Java project 这一选项，在弹出界面的 Project name 栏中输入项目名，例如“Test”，其他选项都是默认的即可，直到单击 Finish 创建项目完毕。

名为“Test”的项目建立成功，界面如图 1-4 所示，在 Test 节点下出现了 src 节点。与此同时，在工作空间中出现“Test”文件夹，其中包含目录 src、bin、lib 和.settings。src 目录中存放在该项目下编写的 Java 源代码，bin 目录存放编译后的字节码文件，lib 存放与该项目相关的包，.settings 目录存放与配置相关的文件。

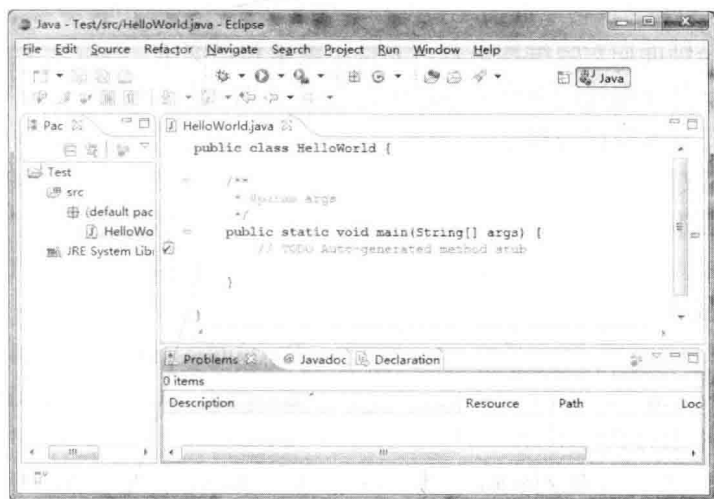


图 1-4 Eclipse 建立工程界面

#### 2. 创建 Java 程序

选择项目 Test 下的 src，单击右键，选择 New→class，输入类名，这里输入了类名“Hello Word”，选中 public static void main(String[] args)，单击 Finish 按钮，Java 程序文件创建成功。

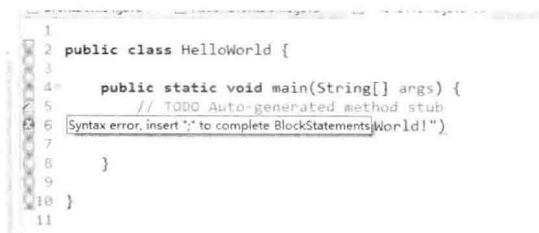


图 1-5 编写 Java 程序

#### 3. 编写、运行程序

Eclipse 采用即时编译机制，在输入代码的过程中，如果某一行语句有错误，就会在该行的左侧显示一个“红叉”，将鼠标移至该红叉位置，会显示错误提示信息，如图 1-5 所示。

程序编写完毕后,单击工具栏上的  (run) 按钮,即可运行程序。



## 本章小结

本章介绍了 Java 语言的发展历史、Java 语言的运行原理及平台构成和 Java 语言的特点。通过编写了一个简单的 Java 程序,详细说明了 Java 程序的编译及运行过程。最后,介绍了 Java 语言编程中常用的 Eclipse 软件的安装与使用。

通过本章的学习,读者对 Java 语言有了初步的认识,了解了 Java 程序的开发过程,更多的 Java 语言知识将在后续章节中讲解。

## 习 题



### 一、单项选择题

1. Java 程序经过编译后生成的文件的后缀是\_\_\_\_\_。  
A. obj                      B. exe                      C. class                      D. java
2. 下面关于 Java 语言特点的描述中,错误的是\_\_\_\_\_。  
A. Java 是纯面向对象编程语言,支持单继承和多继承  
B. Java 支持分布式的网络应用,可访问网络上的其他对象  
C. Java 支持多线程编程  
D. Java 程序与平台无关、可移植性好
3. Java 语言的核心包中,提供编程应用的基本类和接口的包是\_\_\_\_\_。  
A. java.util                B. java.lang                C. java.applet                D. java.rmi
4. 下面的 main() 方法原型,正确的是\_\_\_\_\_。  
A. void main()                      B. public static void main()  
C. public main(String args[])                      D. public static void main(String args[])
5. Java 语言中编译程序的命令是\_\_\_\_\_。  
A. java                      B. javadoc                      C. javac                      D. javah

### 二、简答题

1. 简述 Java 语言的特点。
2. Java 语言的三个技术平台分别是什么?每个平台的适用范围是什么?
3. 简述 Java 虚拟机的主要功能。
4. 简述 Java 程序的开发步骤。
5. 通过查找资料,简述目前 Java 程序开发的主要工具有哪些?



## 第2章 Java 语言基础

学习编程语言与学习英语一样，都需要从字母、单词和语法等基础知识开始。本章将介绍 Java 语言的基础知识，包括标识符、数据类型、变量、运算符、表达式、流程控制、数组、字符串和输入、输出等。

### 2.1 标识符与数据类型

在编写 Java 程序之前，首先需要了解 Java 程序能使用哪些有效字符序列及能处理哪些类型的数据，这就是 Java 语言的标识符与数据类型。

#### 2.1.1 标识符

Java 程序中使用的各种对象，如变量、方法、类、数组等都要有名字，这些名字称为标识符 (Identifier)。标识符由编程者指定，但必须遵循一定的语法规则。Java 语言中的标识符必须满足以下条件：

- (1) 标识符只能包含字母、数字、下划线 ( `_` ) 和美元符号 ( `$` )；
- (2) 标识符必须以字母、下划线和美元符号开头。

标识符的命名除了必须满足上述条件之外，还应使其尽量体现明确的含义，也就是要做到“见名知意”，以提高程序的可读性。例如，`name`、`id`、`width` 等标识符要比 `n1`、`i2`、`w3` 等标识符更直观，尽管它们都是合法的。

另外，Java 语言的标识符是大小写敏感的，即 `Name` 和 `name`、`Person` 和 `person` 是两个不同的标识符，在使用时要特别注意这一点。

在所有合法的标识符中，有一些标识符被 Java 语言赋予特定含义，不允许用户对其重新定义，这些标识符称为保留字 (Reserved Word)。而在所有保留字中又有一些标识符对 Java 的编译器有特殊的意义，它们用来表示一种数据类型，或者表示程序的结构等，称之为关键字 (Keyword)。

Java 语言中共有 53 个保留字，其中 50 个为关键字 (`goto` 和 `const` 目前尚未使用)，另外 3 个保留字是值，分别为 `true`、`false` 和 `null`。Java 语言定义的保留字如表 2-1 所示。

表 2-1

Java 语言定义的保留字

|                       |                        |                        |                          |                       |                          |                         |                           |                           |
|-----------------------|------------------------|------------------------|--------------------------|-----------------------|--------------------------|-------------------------|---------------------------|---------------------------|
| <code>abstract</code> | <code>assert</code>    | <code>boolean</code>   | <code>break</code>       | <code>byte</code>     | <code>case</code>        | <code>catch</code>      | <code>char</code>         | <code>class</code>        |
| <code>const</code>    | <code>continue</code>  | <code>default</code>   | <code>do</code>          | <code>double</code>   | <code>else</code>        | <code>enum</code>       | <code>extends</code>      | <i><code>false</code></i> |
| <code>final</code>    | <code>finally</code>   | <code>float</code>     | <code>for</code>         | <code>goto</code>     | <code>if</code>          | <code>implements</code> | <code>import</code>       | <code>instanceof</code>   |
| <code>int</code>      | <code>interface</code> | <code>long</code>      | <code>native</code>      | <code>new</code>      | <i><code>null</code></i> | <code>package</code>    | <code>private</code>      | <code>protected</code>    |
| <code>public</code>   | <code>return</code>    | <code>short</code>     | <code>static</code>      | <code>strictfp</code> | <code>super</code>       | <code>switch</code>     | <code>synchronized</code> | <code>this</code>         |
| <code>throw</code>    | <code>throws</code>    | <code>transient</code> | <i><code>true</code></i> | <code>try</code>      | <code>void</code>        | <code>volatile</code>   | <code>while</code>        |                           |