高等院校计算机应用系列教材

Java 程序设计教程

(第三版)(微课版)

林巧民　主　编
张苏伟　姜　玻　张台国　副主编

清華大學出版社
北　京

内 容 简 介

本书以 Java 语言为基础，详细介绍计算机语言的结构化编程和面向对象编程。全书共分 12 章，主要内容包括 Java 入门、Java 编程基础、Java 程序基本结构、方法与数组、类和对象、继承、多态与接口、字符串和常用库类、多线程和异常机制、图形用户界面、Java I/O、Java 游戏开发基础，以及药店药品管理系统开发实例。如果说结构化编程的特征是方法，那么面向对象编程的体现就是类的设计和使用，全书对这两种不同的程序设计思想都做了充分介绍。此外，每章的最后都配有思考练习，习题有选择题、填空题、简答题、编程题等多种类型，选择题、填空题和简答题有助于读者对所学知识的理解和掌握，编程题则可以提高读者的动手和实践能力。

本书结构清晰、内容翔实，可作为高等院校相关专业的教材，也可作为从事软件开发工作的专业技术人员的参考书。

本书配套的电子课件、实例源程序和习题答案可以到 http://www.tupwk.com.cn/downpage 网站下载，也可以扫描前言中的二维码下载。读者扫描前言中的视频二维码可以观看视频进行学习。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。
版权所有，侵权必究。举报：010-62782989，beiqinquan@tup.tsinghua.edu.cn

图书在版编目(CIP)数据

Java 程序设计教程：微课版 / 林巧民主编. —3 版. —北京：清华大学出版社，2022.5
高等院校计算机应用系列教材
ISBN 978-7-302-60574-4

Ⅰ. ①J… Ⅱ. ①林… 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(2022)第 064248 号

责任编辑：胡辰浩
封面设计：高娟妮
版式设计：孔祥峰
责任校对：马遥遥
责任印制：宋 林

出版发行：清华大学出版社
网 址：http://www.tup.com.cn，http://www.wqbook.com
地 址：北京清华大学学研大厦 A 座 邮 编：100084
社 总 机：010-83470000 邮 购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：天津安泰印刷有限公司
经 销：全国新华书店
开 本：185mm×260mm **印 张**：21.5 **字 数**：551 千字
版 次：2008 年 9 月第 1 版 2022 年 6 月第 3 版 **印 次**：2022 年 6 月第 1 次印刷
定 价：79.00 元

产品编号：095595-01

前　言

Java 语言自面世以来，一直受到大学生和广大软件研发人员的青睐。目前，许多高校已改变先讲授 Pascal 语言或 C 语言，再让学生选修 Java 语言的惯例，而开始尝试让学生在大学低年级就学习 Java 语言。还有不少高校甚至对非计算机专业的大一新生开设了 Java 课程。但目前，市面上大多数的 Java 教程在讲述面向对象技术时几乎都忽视了对 Java 语言基础的介绍，片面追求技术的新、奇、特，无法满足编程初学者的入门需要。

本书旨在突破市面上大多数 Java 教材的局限，尝试用一种语言来充分阐述两种编程思想，即结构化程序设计和面向对象程序设计，以满足普通初学者的需要。事实上，结构化程序设计是面向对象程序设计的基础，面向对象程序的基本组成还是结构化程序。面向对象程序设计引入了类的概念，使得程序设计人员可以站在设计类(而不是方法)的高度，对程序进行设计和实现，同时必须重视结构化程序设计基本功的锻炼，因为类的设计恰恰是建立在结构化程序设计的基础之上的。因此，本书以 Java 语言为工具，从结构化程序设计和面向对象程序设计两种不同编程思想的角度，分别对 Java 编程的相关基础知识予以介绍，希望能对广大编程爱好者尤其是初学者有所裨益。

全书共分 12 章，各章的主要内容如下。

第 1 章是 Java 入门，简要介绍 Java 的诞生、Java 语言的特点、Java 开发工具以及具体的开发步骤等。

第 2 章是 Java 编程基础，主要介绍 Java 的基本数据类型、赋值语句、条件表达式、运算等。

第 3 章是 Java 程序基本结构，详细介绍程序的 3 种基本流程结构：顺序结构、分支结构和循环结构。

第 4 章是方法与数组，主要介绍方法的概念与定义、方法的调用、变量的作用域、数组以及数组与方法的关系等。

第 5 章是类和对象，详细介绍类的概念和定义、对象的创建与使用、访问控制符和包等。

第 6 章是继承、多态与接口，详细介绍继承与多态技术、抽象类和接口等知识。

第 7 章是字符串和常用库类，主要介绍 Java 提供的 String、StringBuffer 类和部分常用库类。

第 8 章是多线程和异常机制，详细介绍线程的概念、创建、生命周期及状态、线程同步、优先级和调度等；还对 Java 的异常机制做了简要介绍。

第 9 章是图形用户界面，详细介绍 AWT 组件集中的常用组件，包括容器类组件、布局类组件、普通组件以及事件处理机制等。此外，本章最后还简要介绍 Swing 组件集。

第 10 章是 Java I/O，即 Java 输入输出，详细介绍 Java 输入输出流的概念、字节流类、字符流类、File 类以及 RandomAccessFile 类等。

第 11 章是 Java 游戏开发基础，介绍游戏编程的相关知识，包括图形环境的坐标体系、图形图像的绘制、各种坐标变换、动画的生成和动画闪烁的消除等。

第 12 章是药品管理系统开发实例，以 SQL Server 为数据库，详细介绍基于 Java 的药店药品管理系统的开发过程，完成了药店药品管理系统的基本页面布局、数据库存储以及部件的应用，实现了基本的药品信息存储、购进药品、出售药品、药品保质期预警、进货价格曲线图等功能。

本书在编写过程中力求做到概念清楚、由浅入深、通俗易懂、论述详尽、实例丰富，以方便读者自学。全书内容具有较强的实用性。

本书由林巧民、张苏伟、姜玻和张台国共同编著，燕城成、周斌、吕民军、黄业林和沈慧琳制作了部分微课视频。由于作者水平所限，书中难免会有不足之处，敬请广大同行和读者给予批评和指正。我们的邮箱是 992116@qq.com，电话是 010-62796045。

本书配套的电子课件、实例源程序、习题答案可以到 http://www.tupwk.com.cn/downpage 网站下载，也可以扫描下方的二维码下载。读者扫描下方的视频二维码可以观看视频进行学习。

学习资源

视频

作者

2022 年 1 月

目　录

第 1 章　Java 入门 …… 1
1.1　概述 …… 1
1.1.1　Java语言的诞生 …… 2
1.1.2　Java语言的特点 …… 3
1.1.3　Java与其他编程语言间的关系 …… 5
1.2　Java开发环境配置 …… 6
1.2.1　软件安装 …… 6
1.2.2　环境变量配置 …… 7
1.3　第一个Java程序 …… 9
1.3.1　Java程序的结构 …… 9
1.3.2　编译运行 …… 10
1.3.3　中文问题 …… 13
1.4　Java开发工具 …… 13
1.5　小结 …… 15
1.6　思考练习 …… 15

第 2 章　Java 编程基础 …… 17
2.1　引言 …… 17
2.1.1　符号 …… 17
2.1.2　分隔符 …… 18
2.1.3　常量 …… 19
2.1.4　变量 …… 20
2.1.5　final变量 …… 21
2.1.6　变量类型转换 …… 21
2.2　基本数据类型 …… 22
2.2.1　布尔型 …… 22
2.2.2　整型 …… 23
2.2.3　浮点型 …… 25
2.2.4　字符型 …… 25
2.3　程序语句 …… 26
2.3.1　赋值语句 …… 26
2.3.2　条件表达式 …… 27
2.3.3　运算 …… 28
2.4　举例 …… 30
2.5　小结 …… 31
2.6　思考练习 …… 31

第 3 章　Java 程序基本结构 …… 32
3.1　复合语句 …… 32
3.2　顺序结构 …… 33
3.3　分支结构 …… 37
3.3.1　单分支if语句 …… 37
3.3.2　双分支if-else语句 …… 40
3.3.3　分支结构嵌套 …… 43
3.3.4　switch语句 …… 48
3.4　循环结构 …… 51
3.4.1　while语句 …… 51
3.4.2　do-while语句 …… 54
3.4.3　for语句 …… 56
3.4.4　循环嵌套 …… 57
3.4.5　跳转语句 …… 59
3.5　小结 …… 62
3.6　思考练习 …… 62

第 4 章　方法与数组 …… 66
4.1　方法的概念和定义 …… 66
4.2　方法的调用 …… 68
4.2.1　调用方式 …… 69
4.2.2　参数传递 …… 71
4.2.3　返回值 …… 72
4.2.4　方法嵌套及递归 …… 73
4.3　变量作用域 …… 78
4.4　数组 …… 80

4.4.1 数组的概念 80
4.4.2 数组的声明和创建 80
4.4.3 数组的应用举例 83
4.5 数组与方法 88
4.6 小结 89
4.7 思考练习 89

第 5 章 类和对象 92
5.1 引言 92
5.2 类 93
5.2.1 类声明 95
5.2.2 类体 96
5.2.3 成员变量 96
5.2.4 成员方法 98
5.2.5 方法重载 101
5.2.6 构造方法 102
5.2.7 main()方法 103
5.2.8 finalize()方法 103
5.3 对象 104
5.3.1 对象的创建 104
5.3.2 对象的使用 106
5.3.3 对象的清除 108
5.4 访问控制符 108
5.4.1 类的访问控制符 109
5.4.2 对类成员的访问控制 109
5.5 包 112
5.5.1 包的创建 113
5.5.2 import语句 115
5.6 小结 118
5.7 思考练习 118

第 6 章 继承、多态与接口 121
6.1 继承与多态 121
6.1.1 子类、父类与继承机制 121
6.1.2 Java的继承 122
6.1.3 多态性 126
6.2 抽象类和接口 130
6.2.1 抽象类 130
6.2.2 接口 131
6.3 其他 135
6.3.1 final关键字 135
6.3.2 实例成员和类成员 136
6.3.3 类java.lang.Object 140
6.3.4 内部类 142
6.4 小结 144
6.5 思考练习 144

第 7 章 字符串和常用库类 147
7.1 字符串的创建 147
7.1.1 创建String类型的字符串 147
7.1.2 创建StringBuffer类型的字符串 148
7.2 String类型字符串的操作 149
7.3 StringBuffer类型字符串的操作 158
7.3.1 字符串操作 158
7.3.2 字符分析器 163
7.3.3 main()方法 164
7.4 常用库类 164
7.4.1 包装类 164
7.4.2 数字处理类 168
7.4.3 时间日期类 169
7.4.4 System类 171
7.4.5 Scanner类 172
7.5 小结 173
7.6 思考练习 173

第 8 章 多线程和异常机制 175
8.1 多线程 175
8.2 多线程的创建 176
8.2.1 Thread子类创建线程 176
8.2.2 使用Runnable接口 178
8.3 线程的生命期及其状态 179
8.3.1 线程的状态 179
8.3.2 与线程状态有关的Thread类方法 180
8.4 线程的同步 184
8.4.1 方法同步 185
8.4.2 对象同步 187
8.4.3 饿死和死锁 188
8.5 线程的优先级和调度 189
8.5.1 线程的优先级 189
8.5.2 线程的调度 189

8.6 守护线程……190
8.7 线程组……192
8.8 异常机制……194
8.8.1 异常示例……195
8.8.2 异常抛出和处理……196
8.8.3 异常类……200
8.9 小结……204
8.10 思考练习……204

第 9 章 图形用户界面……206
9.1 概述……206
9.2 AWT组件集……207
9.2.1 容器类组件……208
9.2.2 布局类组件……208
9.2.3 普通组件……218
9.2.4 事件处理……228
9.3 Swing组件集简介……239
9.4 小结……245
9.5 思考练习……245

第 10 章 Java I/O……246
10.1 引言……246
10.2 流的概念……246
10.2.1 标准输出……247
10.2.2 标准输入……249
10.3 字节流……253
10.3.1 InputStream……253
10.3.2 OutputStream……259
10.4 字符流……263
10.4.1 Reader……263
10.4.2 Writer……266
10.5 文件……272
10.5.1 File类……272
10.5.2 RandomAccessFile类……274
10.6 小结……277
10.7 思考练习……277

第 11 章 Java 游戏开发基础……279
11.1 概述……279
11.2 绘制2D图形图像……279
11.2.1 坐标体系……279
11.2.2 绘制图形……280
11.2.3 绘制图像……281
11.3 图形图像的坐标变换……284
11.3.1 使用Graphics2D类进行坐标变换……284
11.3.2 使用AffineTransform类进行坐标变换……288
11.4 生成动画……291
11.5 消除动画闪烁……294
11.6 小结……297
11.7 思考练习……297

第 12 章 药店药品管理系统开发……298
12.1 概述……298
12.2 需求分析……298
12.3 系统设计……299
12.3.1 开发环境……299
12.3.2 系统功能结构……299
12.3.3 系统业务流程……300
12.4 模块的划分与设计……301
12.5 系统数据库与数据表设计……301
12.5.1 数据库分析……301
12.5.2 数据表分析……302
12.6 工具类设计……303
12.6.1 基于SQL Server的数据库连接工具类……304
12.6.2 表格标签提供类……306
12.7 数据更新模块的设计与实现……307
12.7.1 药品编码表模块……307
12.7.2 药品编码表录入模块……310
12.7.3 药品信息表模块……312
12.7.4 药品信息表录入模块……315
12.8 数据查询模块的设计与实现……316
12.8.1 药品有效期查询模块……317
12.8.2 药品库存量查询模块……318
12.8.3 进货价格查询模块……320
12.8.4 进货预测查询模块……325
12.9 系统主菜单的设计与实现……327

12.10 登录模块的设计与实现……328
12.11 系统部署……329
12.12 小结……330
12.13 思考练习……330
参考文献……331
附录 ASCII 码表……332

第1章

Java入门

本章学习目标：

- 了解 Java 语言的历史和特点
- 理解 Java 与其他编程语言的关系
- 掌握 Java 语言开发环境的配置
- 掌握 Java 程序的基本结构及其编译运行过程
- 了解流行的 Java 语言集成开发环境

1.1 概述

Java 是由美国 Sun 公司(现已被 Oracle 公司收购)开发的支持面向对象程序设计的计算机语言。它的最大优势在于借助虚拟机机制实现了跨平台特性，即实现所谓的“一次编译，随处运行”，使代码的移植工作变得不再复杂。也正因为如此，Java 语言迅速流行起来，成为深受广大开发者喜爱的编程语言之一。目前，随着 Java ME、Java SE 和 Java EE 的发展，Java 已经不仅仅是一门计算机开发语言了——以 Java 为核心，开发者已经拓展出了一系列先进的技术。

从应用的角度来看，Microsoft、IBM、DEC、Adobe、SiliconGraphics、HP、Toshiba、Netscape 和 Apple 等大公司均已购买了 Java 许可证，Microsoft 还在其 IE 浏览器中增加了对 Java 的支持。通过 Java 编写的应用程序可以在各种主流平台上运行。另外，对于开发者，众多软件开发商都设计了许多支持 Java 开发的 IDE(Integrated Development Environment，集成开发环境)工具软件，如免费开源的 Java IDE：Eclipse(推荐使用)、Genuitec 公司的 MyEclipse、JetBrains 公司的 Intellij IDEA(推荐使用)、计算机 Java 二级考试指定的 NetBeans、Oracle 公司的 JDeveloper、Sourceforge 上的开源 Dr Java 以及以教育为目的的开源 BlueJ 等。Java 编程语言有 4 种平台：Java 标准版(Java SE)、Java 企业版(Java EE)、Java 微型版(Java ME)和 JavaFX。所有 Java 都由 Java 虚拟机(VM)和应用程序编程接口(API)组成。Java 虚拟机是一个用于特定硬件和软件平台的程序，它运行 Java 技术应用程序。API 是一组软件组件，可用于创建其他软件组件或应用程序。每个 Java 平台都提供了一个虚拟机和一个 API，这使得为该平台编写的应用程序可以在具有 Java 编程语言所有优点的任何兼容系统上运行。

- **Java SE**：当大多数人想到 Java 编程语言时，他们会想到 Java SE API。Java SE 的 API 提供了 Java 编程语言的核心功能。它定义了从 Java 编程语言的基本类型和对象到用于

网络、安全、数据库访问、图形用户界面(GUI)开发和 XML 解析的高级类的所有内容。除了核心 API 之外，Java SE 平台还包括虚拟机、开发工具、部署技术，以及 Java 技术应用程序中常用的其他类库和工具包。

- **Java EE**：Java EE 平台构建在 Java SE 平台之上。Java EE 平台为开发和运行大规模、多层、可扩展、可靠和安全的网络应用程序提供了 API 和运行时环境。
- **Java ME**：Java ME 平台提供了一个 API 和一个小型虚拟机，用于在小型设备(如手机)上运行 Java 编程语言应用程序。该 API 是 Java SE API 的一个子集，是对小型设备应用程序开发有用的特殊类库。Java ME 应用程序通常是 Java EE 平台服务的客户端。
- **JavaFX**：JavaFX 是一个使用轻量级用户界面 API 创建富 Internet 应用程序的平台。JavaFX 应用程序使用硬件加速的图形和媒体引擎来利用更高性能的客户端和现代外观，以及用于连接到网络数据源的高级 API。JavaFX 应用程序可以是 Java EE 平台服务的客户端。

Java SE、Java EE、Java ME 和 JavaFX 是 Java 针对不同的应用领域而提供的不同服务，即提供不同类型的类库。初学者一般可以从 Java SE 入手学习 Java 语言。Java SE 是一个优秀的开发库，开发者可以基于这一环境创建功能丰富的交互式应用程序，并且可以把这些应用部署到不同的平台上。本书将以目前最新的 Java SE Development Kit 17.0.1 版本(JDK 17)为例进行讲述。该版本是 Oracle 公司推出的 LTS 长期支持版，但对于初学者的学习而言，其实 JDK 11 或 JDK 8 等版本也是可以的。Java SE Development Kit 17 提供了不少新特性和增强功能。不过对于大多数项目而言，往往需要更改代码才能利用到这些新变化，但性能除外——开发者只需要升级 JDK 版本，就能免费获得性能提升。Java SE 是多种不同风格软件的开发基础，例如，客户端 Java 小程序和应用程序，以及独立的服务器应用程序等。同时，Java SE 也是 Java ME 和 Java EE 的基础。事实上，大部分非企业级软件几乎都是在 Java SE 上开发部署的。首先，多数开发者选择 Java SE 进行应用软件的开发；其次，Java SE 和 Java EE 是兼容的，企业版是在标准版基础上进行扩充的，在 Java SE 上开发的软件可以在企业版平台无缝运行；最后，通常情况下手机及嵌入式设备上的应用开发和调试也是在 Java SE 环境中完成的。

1.1.1 Java 语言的诞生

早在 1990 年 12 月，Patrick Naughton、Mike Sheridan 和 James Gosling 成立了一个名为 Green Team 的小组(此即 Sun 公司的前身)。该小组的主要目标是发展一种分散式系统架构，使其能在消费级电子产品平台上执行，如 PDA(个人数字助理)、手机、家用电器等。1992 年 9 月 3 日，Green Team 发布了一款名为 Star Seven(*7)的机器。它有点像早期的 PDA 产品，不过有着比 PDA 更强大的功能，如无线通信(Wireless Network)、5 寸彩色 LCD、PCMCIA 界面等。

Java 语言的前身 Oak 就是在那个时候诞生的，其主要目的是用来撰写在 Star 7 上运行的应用程序。为什么取名 Oak 呢？因为在 James Gosling 办公室的窗外，正好有一棵橡胶树(Oak)，于是顺手就取了这个名字。Java 所提供的一些特性，在 Oak 中就已经具备了，例如安全性、网络通信、面向对象、垃圾收集(Garbage Collection)、多线程等。Oak 已经是一个相当优秀的程序设计语言，为什么 Oak 会改名为 Java 呢？因为当 Oak 要去注册商标时，发现已经有另外一家公司先使用了 Oak 这个名字。既然 Oak 不能用，那要取什么新名字呢？工程师们边喝咖啡边讨

论着，看看手上的咖啡，灵机一动，就叫 Java 好了。Java 是印度尼西亚爪哇岛的英文名称，因盛产咖啡而闻名。就这样，它就变成了业界所熟知的 Java 语言了。

1995 年 5 月 23 日，JDK(Java Development Kits) 1.0 版本正式对外发布，它标志着 Java 语言的正式诞生。2014 年 3 月 18 日，Oracle 正式发布了 Java SE 8。Java 8 是一个非常流行的版本。之后，Java 11 和 Java 17(均为长期支持版本)相继推出。

1.1.2　Java 语言的特点

Java 语言之所以流行并得到广泛应用，这和它的优秀特性是分不开的。

1. 平台独立性

平台独立性意味着 Java 程序可以在任何支持 Java 语言的平台上运行。Java 虚拟机独立于所有其他软硬件而运行。因此，不管操作系统是 Windows、Linux、UNIX 还是 macOS，也不管计算机是大型机、小型机还是普通 PC，甚至是 PDA 或手机、智能家电，Java 程序都能运行。当然，在这些平台上都要装有相应版本的 JVM(Java 虚拟机)，即运行平台必须支持 Java 语言。

现在，绝大部分手机都支持 Java，很多手机应用也采用 Java 开发。这样，任何支持 Java 的手机都能运行这些应用，这就是平台独立性所带来的好处，如图 1-1 所示。

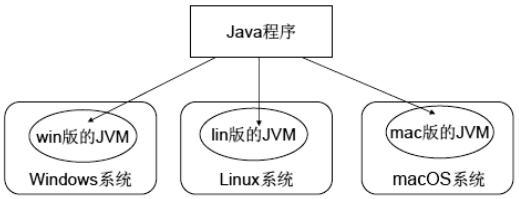

图 1-1　Java 应用程序可以跨平台运行

平台独立性保证了软件的可移植性，而软件的可移植性是软件投资在未来的保证。运用 Java 开发的软件，保证了程序在将来移植时的便捷性。其他语言，如 C 和 C++也宣称其是可移植的，但其移植工作仍然需要花费大量的时间和精力来适配不同的平台和硬件。

2. 安全性

目前，用 Java 语言开发的应用程序很多都属于网络应用程序，因此对软件的安全性有很高的要求。如果没有安全保证，用户运行从网络上下载的 Java 应用程序将会是十分危险的行为。Java 语言通过一系列安全措施，在很大程度上避免了病毒程序的产生和网络程序对本地系统的破坏，具体体现在如下几点。

(1) 去除指针这种数据类型，简化了编程，避免了程序对内存可能的非法访问。

(2) Java 是一种强类型的程序设计语言，要求显式的声明。这样，可以保证编译器提前发现程序错误，提高程序的可靠性。

(3) 垃圾自动回收机制。这一机制可让程序员从烦琐的内存管理工作中解脱出来，专注于程序逻辑本身的开发。更重要的是，通过这种内存自动回收机制，可以很好地保证内存管理的正确性，避免出现“内存泄漏”等问题。

(4) Java 语言提供了异常处理机制。

(5) Java 程序运行时，解释器会对其进行数组和字符串等的越界检查，确保程序的安全执行。运行时堆栈溢出是被 Java 所禁止的(它是蠕虫等病毒常用的袭击手段)。

3. 多线程

在以前的 DOS 时代，人们一次只能运行一个程序，运行完一个程序后才能运行另一个。后来出现了视窗系统，如 Windows 之后，人们可以同时运行几个程序，并且可以在各个运行的程序之间进行切换，如一边听音乐一边编辑 Word 文档。这时的操作系统引入了进程的概念，每个运行中的程序都是一个进程。再后来，为了提高程序的并发性，又引入了线程的技术。

线程也称为轻量级的进程。进程是系统分配资源的基本单位，而线程则是 CPU 调度执行的基本单位。一个进程可以只有一个线程，也可以有多个线程。在很多情况下，开发多线程程序是很有必要的。例如，在早期单线程程序时代，安装程序在开始执行安装操作后，就只能一路安装下去，而现在的软件安装程序一般都提供了“取消”操作，允许安装者在安装过程中随时取消安装，这是安装程序“多线程”化后的一个表现。

多线程的目的就是分散总程序的执行调度，让子程序们“同时”并发地执行。这里的“同时”只是为了强调 CPU 执行各个子程序的速度很快，从宏观上看，像是同时在执行。如果要实现真正的并发同时，就要借助多核甚至多处理器，如现在广泛流行的四核、八核 CPU。另外，随着程序规模的扩大以及对效率的重视，在线程之后又出现了纤程技术。纤程对线程又做了进一步细分，成为 CPU 调度的基本单位，使程序员在设计并发程序时可以更加灵活。

Java 是支持多线程程序开发的，它提供的 Thread 类，负责线程的启动运行、终止等，并可测试线程状态。后面章节会有关于多线程的专门介绍。

4. 网络化

在网络环境中，程序可以在本地或远程机器上执行。Java 程序可以通过网络来打开和访问对象，就像访问本地系统一样简单。Java 语言提供了丰富的类库，保证其可以在 HTTP、FTP 和 TCP/IP 协议中良好运行。

5. 面向对象

随着软件业的蓬勃发展，面向对象程序设计方法已经流行开来，出现了很多面向对象的程序设计语言，如 C++、SmallTalk 和 Ruby 等。Java 也属于面向对象程序设计语言。简单地说，面向对象主要是通过引入类的概念，使得原本面向过程的程序设计方法有了质的飞跃。类中不仅包含数据部分，而且还包含操作方法。这个囊括了数据和方法的类，成为面向对象程序设计中最关键的要素。可以说，所有功能的实现都是围绕类展开的。同样，面向对象技术的特征也是由类体现出来的。面向对象技术主要有三大特征。

(1) 封闭性。类定义的一般形式如下：

```
class 类名
{
    类的细节
};
```

其中，“类的细节”以类的形式封装起来了，该细节就是类的成员。它可以是数据，也可以是操作这些数据的方法(在面向过程程序设计中被称为函数)。当这些数据和方法的访问权限被设置为私有后，它们就不能被其他对象从外部访问，像是被隐藏了起来。而对外部只暴露那些访问权限被设置为公有的成员。

(2) 继承性。类是可继承的，就像遗产一样。一个已经设计好的类，可以被其他类继承，并扩充数据和方法部分，成为功能更为强大的新类。这样可以大大提高代码的可复用性，提高程序的开发效率，同时也能降低系统复杂性，提升代码的可读性。

(3) 多态性。在传统的面向过程程序设计中，函数一般是不能重名的。即便功能类似而只是处理的数据类型不同的函数也需要不同的函数名，这就大大增加了程序的复杂性和维护的难度。而在面向对象程序设计中允许同名方法的存在，只是形式参数不同，执行程序时，根据实际参数的不同，自动调用对应的(同名)方法。多态性是面向对象技术的三大特征之一。

1.1.3　Java 与其他编程语言间的关系

程序开发语言可以分为 4 代：机器语言、汇编语言、高级语言和面向对象程序设计语言。机器语言指机器最终执行时所能识别的二进制 0、1 序列，任何其他语言编写的程序最终都要转换为相应的机器语言才能运行。在电子计算机刚刚诞生的一小段时间内，人们只能用 0、1 序列进行编程，难度可想而知。后来为了提高编程效率，引入了英文助记符，这样就出现了汇编语言。汇编语言的出现，大大提升了代码的编写速度，同时也使代码的可读性和可维护性大大提高。直到今天，仍然有人在用汇编语言进行编程，这主要是为了保证程序性能的缘故。系统底层应用，如一些硬件驱动程序的编写，使用汇编语言编程，其执行效率一般会更高。但是，汇编语言仍然不容易被快速理解，对程序员水平有很高的要求，这就限制了其他领域的科技工作者们利用计算机编程辅助工作的普及。为了推广计算机编程，使之成为社会各行各业的一种工具，这就需要开发语法简单、编写容易的编程语言。比尔 • 盖茨的第一桶金据说就是从这个需求中赚到的。他在大学时代成功移植了 BASIC 语言，并将其出售给 IBM 公司，预装到 IBM 的计算机中。BASIC 语言是由达特茅斯学院院长 John G. Kemeny 与数学系教师 Thomas E. Kurtz 共同研制出来的。1964 年 BASIC 语言正式发布；1975 年比尔 • 盖茨把它移植到计算机上。BASIC 语言本来是为校园的大学生们创造的高级语言，目的是使大学生容易使用计算机，虽然初期的 BASIC 语言功能弱、语句少，只有 14 条语句(后来发展到 17 条语句)，但由于 BASIC 在当时比较容易学习，它很快从校园走向社会，成为初学者学习计算机程序设计的首选语言。除了 BASIC，还有很多其他高级语言，如 Pascal、FORTRAN、C 等陆续面世。随着软件行业的不断发展，软件的规模变得越来越大，迫切需要更高效的编程语言——面向对象程序设计语言，于是 Java、C++、Visual Basic 和 Delphi 等应运而生。除此之外，世界上还有很多其他编程语言，只是它们不是很流行，所以不为人们所熟知而已。事实上，每一种流行的开发语言也都有其优缺点：C 语言适合开发系统程序，很多操作系统及驱动程序都是用 C 语言编写的；FORTRAN 适合用来进行数值计算；Pascal 语言结构严谨，适合作为教学语言；Visual Basic 和 Delphi 适合用来快速开发中小型可视化应用软件；C++适合开发大型系统级应用程序；Java 适合开发跨平台的应用程序。

总之，每种语言各有其特色，至于选用什么语言作为开发工具，要依据具体的开发需求。没有最好的，只有相对合适的。不少开发任务可能需要同时使用几种语言共同来完成。本书主要面向没有任何编程基础的初学者学习 Java 语言而编写。下面就开始简单的 Java 学习之旅吧！

1.2 Java 开发环境配置

如果只是想运行 Java 编写好的程序，则只需下载 Java 运行时库 JRE(Java Runtime Environment)即可。但如果要用 Java 语言进行开发、编译和运行 Java 程序，则需要下载并安装 JDK(Java Development Kits)。

1.2.1 软件安装

从 Oracle 的官方网站(https://www.oracle.com/java/technologies/downloads/)上可以下载所需要的 JDK 安装包。下载时，读者需要根据自己计算机的系统，如 Linux、macOS 或 Windows，选择相应的 JDK。目前可以下载的最新版本是 Java SE Development Kit 17.0.1(JDK 17)。如果想下载老版本的 Java，可以在打开的页面中单击 Java archive 查找。下面以编者的 Windows 10 64 位系统为例，从上述网址中单击 x64 Installer 的链接 https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.exe，下载安装包 jdk-17_windows-x64_bin.exe。该文件是一个可执行程序，双击即可进行安装。安装程序首先收集一些信息，用于安装的选择，如设置安装路径，然后才开始复制文件、设置环境变量等。安装过程中，只需按照提示操作即可。假设安装路径为默认的 C:\Program Files\Java\jdk-17.0.1，安装完毕后，切换到该目录，可以发现有如下一些子目录。

(1) bin 文件夹：该文件夹包含 Java 编译器(javac.exe)、解释器(java.exe)等 Java 命令，如图 1-2 所示。

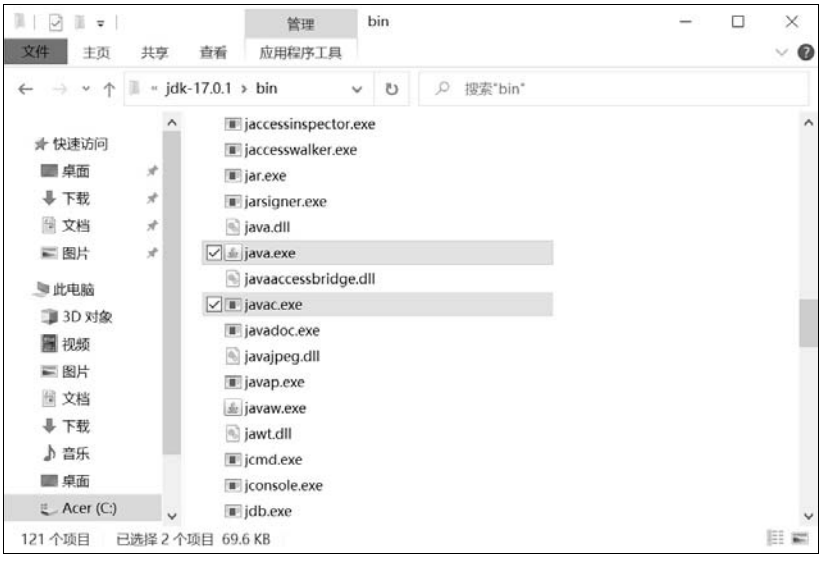

图 1-2 Java 安装目录中的 bin 文件夹

(2) conf 文件夹：存放配置文件，可配置 Java 访问权限，密码。

(3) include 文件夹：头文件，支持源代码编辑。

(4) jmods 文件夹：存放 JDK 的各种模块。

(5) legal 文件夹：存放 Java 及各类模块的 license。

(6) lib 文件夹：存放 JDK 使用的类库。

注意目录 C:\Program Files\Common Files\Oracle\Java\javapath，这个目录中提供了编译器(javac.exe)、解释器(java.exe) 等，它们是后面编译运行程序时需要用到的，而且这个目录的路径在安装 JDK 过程中会被自动设置到操作系统的环境(系统)变量 path 中。这样，在进入 cmd 命令行窗口后系统就可以自动找到它们，便于用户直接调用。

1.2.2　环境变量配置

由于程序开发过程中需要经常用到各种头文件、库文件和编译器文件等，一种比较方便的做法是把这些文件对应的路径添加到系统环境变量中，以方便开发环境找到所需要的文件。

不同的操作系统，其设置系统环境变量的方法也各不相同，这里我们以 Windows 10 操作系统为例，设置环境变量的具体操作如下。

(1) 右击“此电脑”或“我的电脑”图标，从弹出的快捷菜单中选择“属性”命令，单击左上角的“高级系统设置”文字链接，打开“系统属性”对话框，选择“高级”选项卡，如图 1-3 所示。

(2) 从打开的“系统属性”对话框的“高级”选项卡里，单击“环境变量”按钮，打开“环境变量”对话框，如图 1-4 所示。

图 1-3　“系统属性”对话框

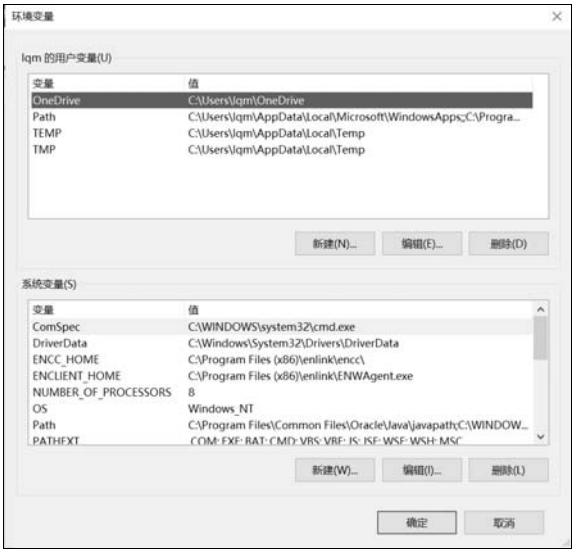

图 1-4　“环境变量”对话框

(3) 在“环境变量”对话框中，“××的用户变量”选项组中的内容是用户个人的环境变量，而“系统变量”列表框中的内容是系统环境变量。它们的区别是：用户变量只对本用户有效；而系统变量则对任何用户均有效。设置用户变量或系统变量后均需重新启动 cmd 命令窗口才能生效。一般情况下，只需要配置两个环境变量：Path 和 classpath。

(4) 观察一下，Path 环境变量在用户变量和系统变量中均存在，双击系统变量中的 Path 环境变量，弹出如图 1-5 所示的“编辑环境变量”对话框。可以发现，路径 C:\Program Files\Common Files\Oracle\Java\javapath 在安装 JDK 时已经被自动添加，此时打开命令行窗口，输入 javac 命令，然后按回车键，将有如图 1-6 所示的信息，该信息表明 JDK 已经安装成功。

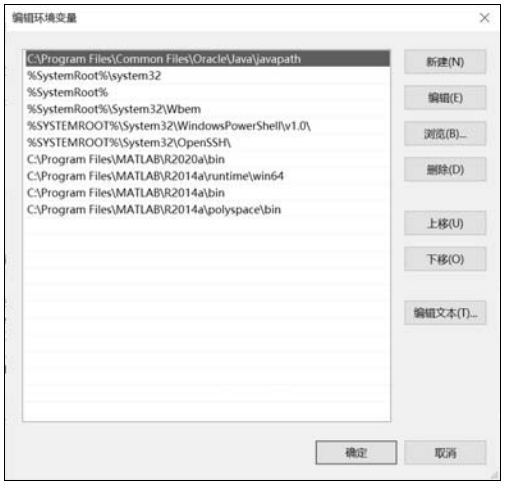

图 1-5 “编辑环境变量”对话框

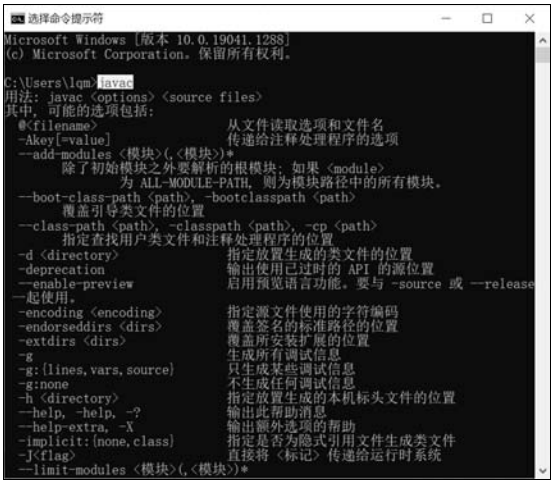

图 1-6 JDK 安装好后可直接调用 javac 命令

(5) classpath 环境变量则需要新建一个，在用户变量或系统变量中新建均可，并将变量值设置为 C:\Program Files\Java\jdk-17.0.1\lib 这个路径。此外，当运行读者自己编写的 Java 程序时，一般还需要将相应的工作目录(即存放 Java 源程序及编译过的字节码文件的目录)添加到 classpath 变量值中，以便程序运行时系统能自动找到读者所编写 Java 源代码(*.java)编译后生成的字节码文件(*.class)。这一点一定要格外注意，很多人在初学 Java 时会忘记，导致程序运行失败。

设置完 classpath 环境变量后，在屏幕左下角的搜索框中输入 cmd，然后按回车键，在打开的命令行窗口中输入 set 命令，验证刚才的设置是否成功，如图 1-7 所示。

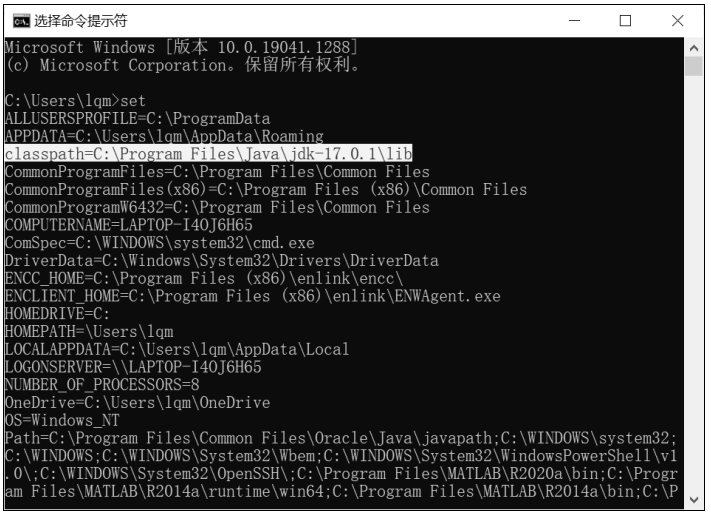

图 1-7 通过 set 命令查看环境变量设置情况

1.3 第一个 Java 程序

1.3.1 Java 程序的结构

运用 Java 编写的程序有两种类型：Java 应用程序(Java Application)和 Java 小应用程序(Java Applet)。虽然二者的语法是完全一样的，但后者需要客户端浏览器的支持才能运行，并且运行前须将其嵌入 HTML 文件的<applet>和</applet>标签对中。当用户浏览该 HTML 页面时，首先从服务器端下载 Java 小应用程序，然后客户端已安装的 Java 虚拟机解释并执行该程序。由于 Java 小应用程序安全性差，JDK 从很早的版本中就将其删除了，不再支持其开发，因此，这里只对 Java 应用程序的结构进行介绍。下面看一个简单的 Java 程序。

```
public class Hello {
    public static void main(String args[])
    {
        System.out.println("Hello, welcome to Java programming.");
    }
}
```

Java 源程序是以文本文件的形式存放的，文件扩展名必须为 java。可将上面的程序代码保存为 Hello.java 文件。这里有一个非常细小但一定要注意的问题：文件名必须与(主)类名一致，包括字母大小写也要一致，因为 Java 语言是大小写敏感的。通常在定义类时，类名的第一个字母都大写。在正确编辑以上代码后，保存时应确保文件名正确，否则后面将不能通过编译，更无法运行。

所有的 Java 语句都必须以英文的分号“;”结束。编辑程序时千万注意不要输入中文的“；”，因为中文 “；” 不能被编译器识别。此外，既然 Java 语言是大小写敏感的，编辑程序时还应该注意区分关键字及标识符中的大小写字母。

下面通过图 1-8 对该程序的结构做简要介绍。

图 1-8　第一个 Java 程序

上述程序中，首先用关键字 class 声明一个新类，类名为 Hello，它是一个公共类(public)。整个类的定义由一对大括号{}括起来。在该类中定义了一个 main()方法。其中，public 表示访问权限，指明所有类都可以调用(使用)这一方法；static 指明该方法是一个静态类方法，它可以通过类名被直接调用；void 则指明 main()方法不返回任何值。对于一个应用程序来说，main()方法是必需的，而且必须按照如上格式定义。Java 解释器在没有生成任何实例的情况下，以 main()方法作为入口来执行程序。一个 Java 源文件中可以定义多个类，每个类中也可以定义多

个方法，但最多只能有一个公共类，main()方法也只能有一个。

在main()方法的定义中，括号()中的String args[ ]是传递给main()方法的参数，参数名为args，它是类String的一个实例，参数可以为0个或多个。每个参数用“类名 参数名”来指定，多个参数之间用逗号分隔。在main()方法的实现(大括号中)部分，只有一条语句：“System.out.println("Hello, welcome to Java programming.");”，它用来实现字符串的输出。这条语句与C语言中的printf 语句和C++中std::cout＜＜语句的功能相同。

在图1-8中，除了类名的定义和唯一的一条语句外，其他部分都可以被看成编写程序时的模板，但要注意字母大小写问题和大括号的配对。

简单Java程序的模板如下：

```
public class 类名 {
    public static void main(String args[])
    {
        //程序代码
    }
}
```

提示：

- 类名后面的大括号对标识着类定义的开始和结束，而main()方法后面的大括号对则标识着方法体的开始和结束。Java 程序中的大括号都是成对出现的，因而在输入左大括号后，最好也把右大括号写上，这样可以避免遗漏；否则，可能会给程序编译、调试带来不便。初学者经常犯此类错误，花了很多时间查错，最后发现原来是大括号不配对。
- 一般将类名的首字母大写，而变量名则以小写字母开头，变量名由多个单词组成时，第一个单词后边的每个单词首字母均大写。
- 程序中应适当使用空格符和空白行对程序语句元素进行间隔，增加程序的可读性。一般在方法体中，将整个方法体的内容部分缩进，这样可以使程序的结构清晰，一目了然。编译器会自动忽略这些间隔用的空格符和空白行。也就是说，它们仅起到提高程序可读性的作用，而不对程序的编译和运行产生任何影响。
- 在编辑程序时，最好一条语句占一行。虽然 Java 允许一条长语句分开写在几行中，但前提是不能从标识符或字符串的中间进行分割。另外，文件名与 public 类名在拼写和大小写上必须保持一致。
- 一个 Java 应用程序有且仅有一个 main()方法，它是程序的执行入口。除了 main()方法外，程序还可以有其他方法，后面章节将会详细介绍。

1.3.2 编译运行

设置好环境变量并编辑好Java程序源代码后，就可以在命令行模式下进行编译和运行了。下面以上述的第一个Java程序为例，说明编译过程。假定程序Hello.java存放在“C:\工作目录”文件夹中，如图1-9所示。

图 1-9　存放 Java 程序的目录

打开命令行窗口，输入“javac Hello.java”命令，试图对源程序进行编译操作，出现如图 1-10 所示的错误提示。

图 1-10　找不到源程序错误

通过图 1-10 可以看到，由于找不到源程序，编译出错。解决的办法是切换到源程序所在的工作目录，然后再运行“javac Hello.java”命令，如图 1-11 所示。

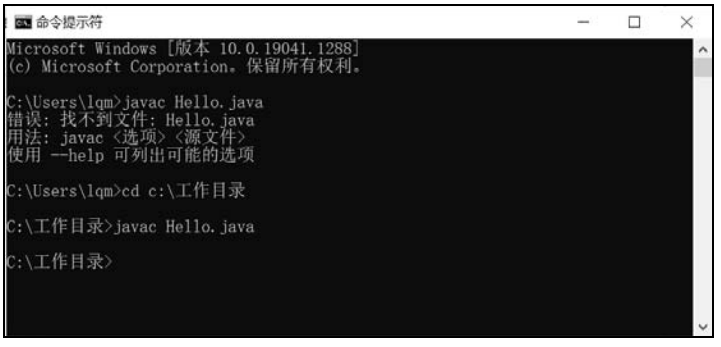

图 1-11　编译 Java 源程序

此时，源程序编译成功。系统自动在工作目录下生成一个字节码文件 Hello.class。这是一个二进制格式的文件，供解释器运行时使用。

对于初学者而言，程序一般都不太可能一次编写成功。因此，当试图编译带错误(如语法错误)的源程序时，系统不会生成二进制的字节码文件，而是在命令行窗口中用“^”符号将可能出错的地方标示出来，并给出适当的信息提示程序员去改正。图 1-12 所示就是一种编译失败的情形。

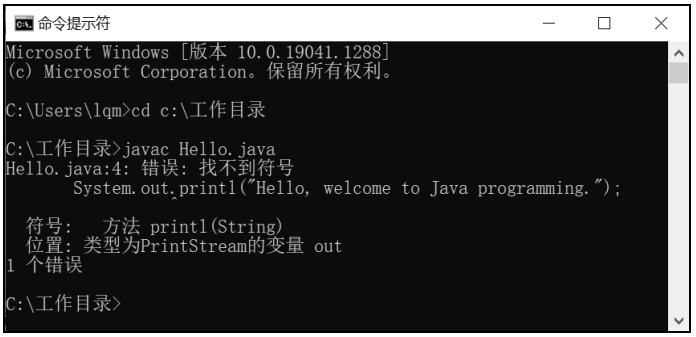

图 1-12　编译失败

图 1-12 中的出错信息提示方法名 printl 不能被识别。原因是编辑源程序时漏掉了 println 最后面的字符 n。有些时候，程序前面的一个错误会导致后面出现一系列的连锁错误。因此，当编译程序出现非常多的错误时，不要慌张，应从第一个错误处开始纠正。

编译成功后就可以执行该程序了。运行该 Java 程序的命令是“java Hello”。这里要注意，java 命令和字节码文件名(不含扩展名.class)之间至少要有一个空格隔开。然后按回车键，如图 1-13 所示。

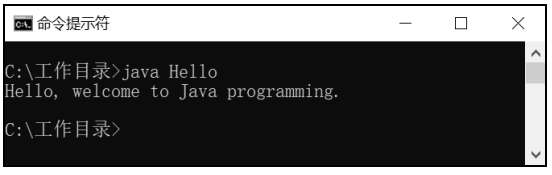

图 1-13　执行 Java 字节码程序

程序中仅有一条 System.out.println()输出语句，输出内容为“Hello, welcome to Java programming.”。图 1-13 中显示了该字符串的原样输出。

有些初学者可能会碰到这样的情形，上次编译运行成功的程序，后来再运行却失败了，如图 1-14 所示。而程序一点也没改动！这是为什么呢？

图 1-14　程序执行失败

在图 1-14 中，试图运行 Hello 字节码文件，却失败了。细心的读者可能会发现，这次执行命令的路径变成了“C:\Users\lqm>”，与原来“C:\工作目录>”不一样。原来的路径保证可以找到本路径下的字节码文件，而现在当前路径不一样了，就找不到了，因此系统提示**错误：找不到或无法加载主类 Hello**。解决上述问题的有效办法是将工作目录的路径添加到 classpath 环境变量中，这样，不管当前路径是什么，就都能找到相应的 class 字节码文件。

提示：

- 源程序文件名的扩展名必须为.java，这点初学者须牢记。有些人直接通过 Windows 系统的右键进行文本文件的创建，然后对该文本文件进行重命名，但是他的 Windows 系

统可能是不显示已知文件类型的扩展名的。例如，本来想命名为 Hello.java，却由于对 Windows 系统不熟悉，实际的文件名为 Hello.java.txt，其中 txt 扩展名被系统隐藏了。对于这种情况，可以通过修改 Windows 系统的配置来解决。在窗口中单击“查看”命令，选中“文件扩展名”复选框，如图 1-15 箭头所示，再对文件扩展名进行修改即可，如图 1-16 所示。

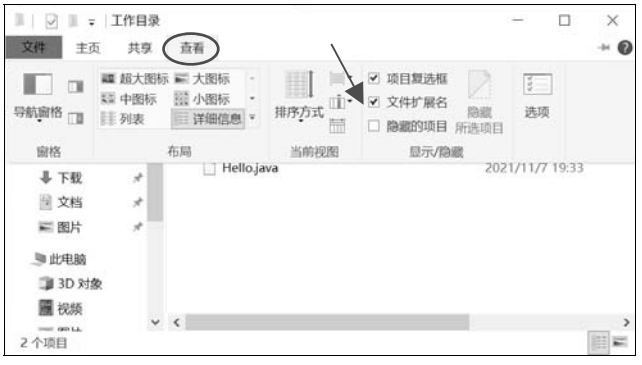

图 1-15　选中“文件扩展名”复选框

图 1-16　修改扩展名为 java

- 开发 Java 程序时，必须安装 JDK，而只运行 Java 程序时，用户只需要安装 JRE(Java Run-time Environment，Java 运行时环境)即可。安装 JDK 后，JRE 也就自动安装上了，较低版本的 JDK 安装完，会有一个单独的 jre 目录，而 JDK11 以上则没有。
- 编译型语言 C/C++直接可以编译成操作系统可以识别的可执行文件，而 Java 只编译成 Java 自己的可执行文件格式(.class 字节码文件)，.class 文件在执行时还需要 Java 虚拟机对.class 文件中的代码一句一句地进行解释、再执行。
- Java 虚拟机负责对字节码进行解释，对于不同运行平台，有不同的虚拟机。正是虚拟机屏蔽了底层平台的差异性，才实现了“一次编译，随处运行”。

1.3.3　中文问题

前面 Hello 实例代码中，若“System.out.println("Hello, welcome to Java programming.");”语句改写为“System.out.println("嗨, welcome to Java programming.");”，即程序代码中出现中文，这时再编译会报错：编码 GBK 的不可映射字符(0xA8)。解决办法有两种：一种是采用“javac -encoding UTF-8 Hello.java”命令来编译，即添加“-encoding UTF-8”告知编译器源文件编码格式；另一种是将源文件另存为 ANSI 编码格式，即可成功编译。

1.4　Java 开发工具

编写 Java 源程序的工具软件有很多，只要是能编辑纯文本的工具都可以，如 Windows 自带的 Notepad(记事本)或 Notepad++(免费)、UltraEidt 和 EditPlus 等商业软件。然而，如上一节所述，编辑完的源代码还需要用户自己执行编译、运行等操作，一旦遇到编译错误或执行结果错误，调试也是一件很麻烦的事情。因此，Java 开发人员一般会使用一些 IDE 工具软件来编写和调试程序，以提高开发效率、缩短开发周期。下面介绍一些比较流行的 IDE 工具软件及其特点，

如当下最流行的两款工具 Eclipse 和 IDEA 等。

1. Eclipse IDE

Eclipse 是一个著名的跨平台自由集成开发环境，它可扩展、开放源代码、免费，深受开发人员和软件开发公司的喜爱，是目前最流行的 Java 集成开发环境。图 1-17 是 Eclipse 官网的下载界面。

图 1-17　Eclipse 官网的下载界面

2. MyEclipse

MyEclipse 是对 Eclipse IDE 的扩展，增加了许多功能。相对于 Eclipse 来说，MyEclipse 更像是将 Eclipse 中好多插件都集成起来了，性能方面也有更好的考虑，不过这款 IDE 是收费的。因为 MyEclipse 属于收费软件，所以大公司很少使用。

3. Intellij IDEA

Intellij IDEA 简称 IDEA，具有美观、高效等众多特点。IDEA 是 JetBrains 公司的产品，这家公司总部位于捷克共和国的首都布拉格，开发人员以严谨著称的东欧程序员为主。它的旗舰版本还支持 HTML、CSS、PHP、MySQL、Python 等。免费版只支持 Java 等少数语言。据传它有“最智慧的 Java IDE”之称，它能帮助开发人员拿出最具有创造性的解决方案，它的 Smart Code Completion 和 On-the-fly Code Analysis 功能等可以提高开发人员的工作效率。IDEA 是目前最流行的商业 Java 集成开发环境，对社区版、学生以及开源项目开发者免费。

4. NetBeans

NetBeans 是业界第一款支持创新型 Java 开发的开放源代码 IDE，可以方便地在 Windows、macOS、Linux 和 Solaris 中运行。它是计算机 Java 二级考试指定软件，但相对于 IDEA 和 Eclipse，其市场还是偏小的。

5. JDeveloper

Oracle 公司的 JDeveloper 是一款涵盖整个开发生命周期的开源式免费 IDE。JDeveloper 可以轻松地与 Oracle 应用开发框架(Oracle ADF)相集成，并通过提供可视化和声明性的开发方法，来简化应用程序开发。除了 Java，它也可以被用于开发 JavaScript、PHP、SQL 和 XML 等应用程序。

6. Dr Java

Dr Java 是一个在 Sourceforge(全球最大的开源软件平台)上不断完善的轻量级 Java IDE。Dr Java 为初学者精心设计，采用 Swing 工具包进行开发，有很漂亮的界面。

7. BlueJ

BlueJ 是以教育为目的的开源 Java IDE。BlueJ 常用于小型软件项目的开发，它有很基础的用户界面，因此初学者可以很好地使用它。值得一提的是，BlueJ 以干净的可视化的方式提供了类和对象的显示，这对于那些以前没有接触过这一类知识的初学者是一个很棒的体验。

此外，jCreator、jGrasp、Greenfoot、Codenvy 和 Xcode 等，都可以用来开发 Java。Java 初学者们如初入大观园的刘姥姥，看花了眼，不知该如何选择，其实这些工具各有所长，没有绝对完美的，读者不妨多试用下。但要记住的是，它们仅仅是集成开发环境，在这些环境中，有一样东西是共同的，那就是 JDK(Java Development Kits)。JDK 是整个 Java 的核心，包括了 Java 运行时环境(Java Runtime Environment)、一些 Java 工具和 Java 的基础类库等，所有的集成开发环境都需要 JDK。对于初学者的建议是：通过“JDK+记事本”的模式来熟悉 Java 简单程序的开发，即用记事本编写代码，然后利用 JDK 编译、运行它，这种开发方式虽然简陋，但不失为初学 Java 语言的可选途径。学习一段时间后，建议改用 Notepad++、UltraEidt 或 EditPlus 等高级记事本类编辑软件，可提高效率。熟悉以上软件后，应试用不同的 IDE，挑选主流的、顺手的 IDE，作为后续开发的利器。

1.5 小结

本章对 Java 语言的相关内容做了初步介绍，让读者对 Java 的历史和特点有所了解。通过一个简单的 Java 应用程序，读者应当掌握 Java 的开发环境配置、程序编写、编译运行等步骤。

下面对 Java 应用程序的开发步骤做一下归纳，主要步骤如下。

(1) 下载 JDK 软件并安装。

(2) 配置相应的环境变量(path 和 classpath)，注意，安装采用压缩包时才需配置 path。

(3) 编写 Java 源程序(使用文本编辑器或集成开发环境)。

(4) 编译 Java 源程序，得到字节码文件(javac *.java)。

(5) 执行字节码文件(java 字节码文件名)。

1.6 思考练习

1. Java 语言有哪些主要特点？
2. Java 编程语言有哪四种不同的平台？
3. Java 应用程序的开发步骤有哪些？

4. 什么是环境变量？设置环境变量的主要目的是什么？

5. 试着编写一个简单的Java应用程序，实现在屏幕上输出“Welcome to Java”字符串，并对该程序进行编译和运行。

6. 编写一个 Java 应用程序，实现分行显示字符串“Welcome to Java”中的 3 个单词。