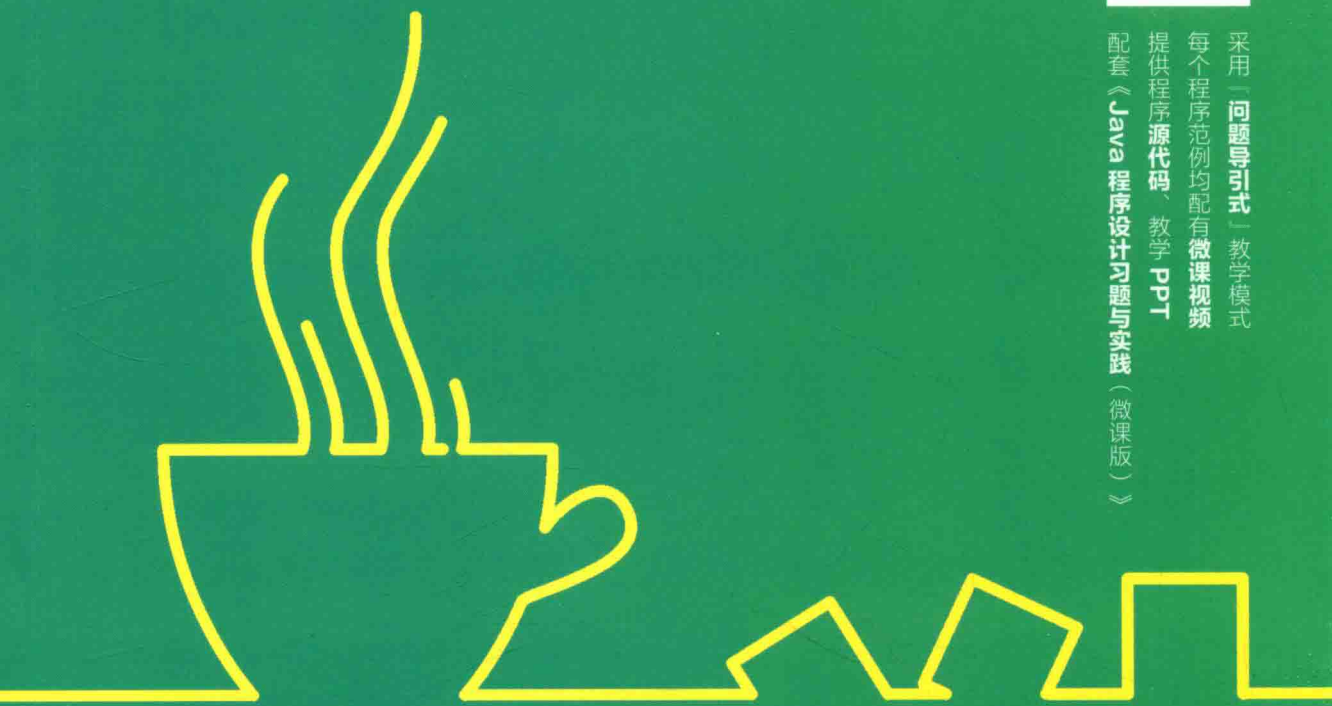
21世纪高等教育计算机规划教材

微课示例

采用『**问题导引式**』教学模式
每个程序范例均配有**微课视频**
提供程序**源代码**、教学 **PPT**
配套《**Java 程序设计习题与实践**（微课版）》



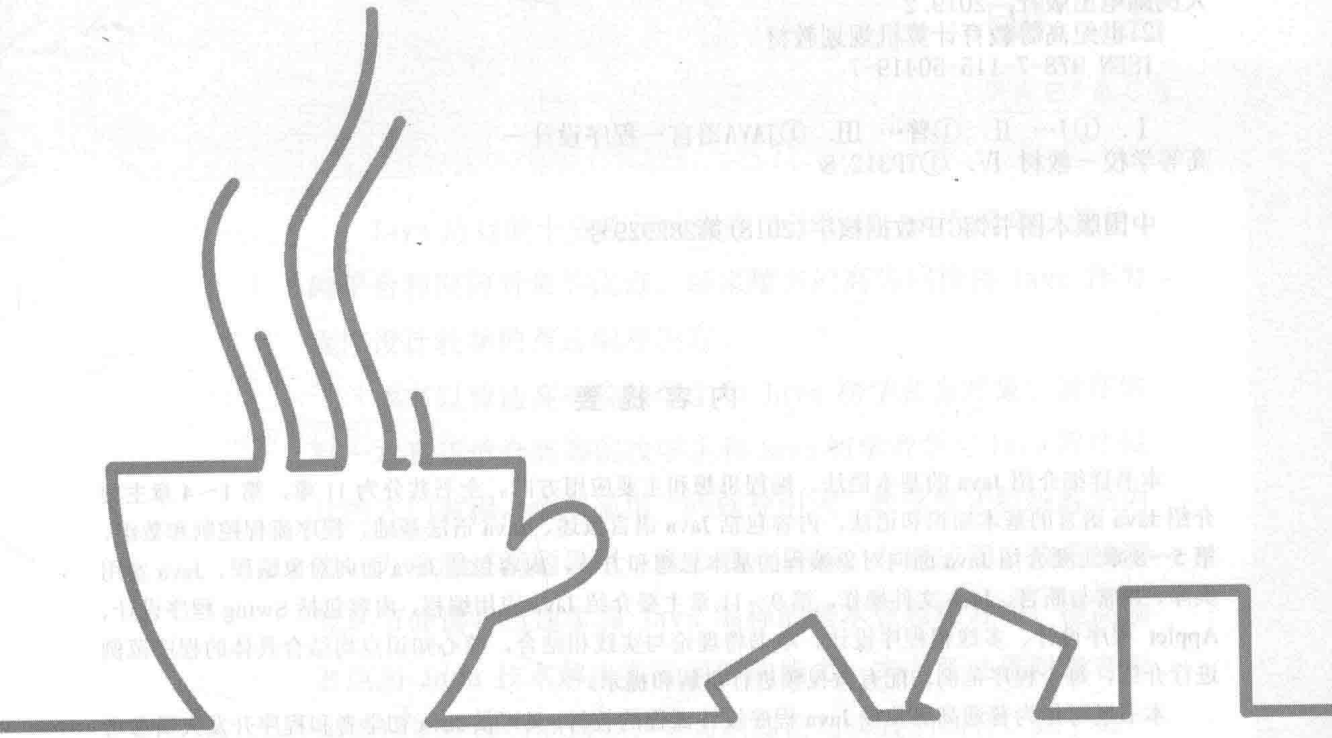
Java程序设计

微课版

普运伟 / **主编**
田春瑾 王樱子 / **副主编**
胡钰 柳翠寅 刘领兵 付湘琼 / **参编**

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等教育计算机规划教材



Java程序设计

微课版

普运伟 / 主编

田春瑾 王樱子 / 副主编

胡钰 柳翠寅 刘领兵 付湘琼 / 参编

人民邮电出版社

北京

图书在版编目（CIP）数据

Java程序设计 : 微课版 / 普运伟主编. -- 北京 : 人民邮电出版社, 2019.2
21世纪高等教育计算机规划教材
ISBN 978-7-115-50419-7

Ⅰ. ①J… Ⅱ. ①普… 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(2018)第282929号

内 容 提 要

本书详细介绍 Java 的基本语法、编程思想和主要应用方向。全书共分为 11 章，第 1～4 章主要介绍 Java 语言的基本知识和语法，内容包括 Java 语言概述、Java 语法基础、程序流程控制和数组。第 5～8 章主要介绍 Java 面向对象编程的基本思想和方法，内容包括 Java 面向对象编程、Java 实用类库、异常与断言、Java 文件操作。第 9～11 章主要介绍 Java 应用编程，内容包括 Swing 程序设计、Applet 程序设计、多线程程序设计。本书将理论与实践相结合，核心知识点均结合具体的程序范例进行介绍，每个程序范例均配有微视频进行讲解和提示。

本书既可作为普通高等院校 Java 程序设计课程的教材，又可供 Java 初学者和程序开发人员参考使用。

◆ 主　　编　普运伟
副 主 编　田春瑾　王樱子
责任编辑　刘海溧
责任印制　焦志炜

◆ 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
大厂聚鑫印刷有限责任公司印刷

◆ 开本：787×1092　1/16
印张：17　　2019 年 2 月第 1 版
字数：425 千字　　2019 年 2 月河北第 1 次印刷

定价：44.80 元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字 20170147 号

前 言

PREFACE

Java 是目前十分流行的程序设计语言，具有简单、健壮、跨平台和面向对象等优点。越来越多的高等院校将 Java 作为程序设计教学的首选编程语言。

本书以普通高等院校学生和 Java 初学者为对象，旨在编写一本真正适合高等院校学生和 Java 初学者学习 Java 程序设计的入门教程。全书采用“问题导引式”教学模式组织教学内容，力求通过问题导引、核心要点讲解、专题应用、效果检测等环节使读者迅速掌握 Java 编程的基本思想和方法，提高读者应用 Java 技术解决实际问题的能力。本书既注重对读者程序设计思维能力的培养，又强调理论与实践的结合。对于每一个核心知识点，本书都结合具体的程序范例进行介绍，每个程序范例均给出了问题分析、程序代码、运行结果和程序说明，并配有精心设计的微视频进一步讲解和提示。同时，每章专门设置“专题应用”，力求拓展各章内容并使读者迅速提高程序设计能力。此外，本书内容注重由浅入深、循序渐进，编写力求简洁明了、通俗易懂，以助读者理解和掌握 Java 面向对象编程的基本思想以及 Java 技术的主要应用方向和编程方法。

每章后面配有自测与思考，以便读者及时检测学习效果。配套出版的由田春瑾主编的《Java 程序设计习题与实践（微课版）》可供学习者巩固知识和上机实践使用。

本书由普运伟担任主编并编写第 1 章和第 5 章，田春瑾担任副主编并编写第 3 章和第 9 章，王樱子担任副主编并编写第 4 章和第 8 章，胡钰编写第 2 章，柳翠寅编写第 6 章，刘领兵编写第 7 章，付湘琼编写第 10 章和第 11 章。全书由普运伟负责统稿和定稿工作。

本书得到昆明理工大学特色精品系列教材建设项目立项支持。本书在编写过程中，得到了昆明理工大学教务处、编者所在部门广大教师的关心和大力支持，在此对他们表示衷心的感谢！

编者

2018 年 11 月

目 录

CONTENTS

第 1 章 Java 语言概述 1

1.1 初识 Java 技术 2
1.1.1 Java 发展历程 2
1.1.2 Java 技术平台 3
1.1.3 Java 语言的特点 3
1.2 理解 JVM、JRE 和 JDK 4
1.2.1 Java 程序的运行机制 4
1.2.2 JRE 5
1.2.3 Java 开发环境 5
1.3 准备 Java 开发环境 6
1.3.1 JDK 的下载、安装和配置 6
1.3.2 常见的 Java 开发工具 6
1.4 编写第一个 Java 程序 7
1.4.1 Java 程序的编辑 8
1.4.2 Java 程序的编译 9
1.4.3 Java 程序的运行 9
1.5 Java 程序的结构和语法规范 10
1.5.1 进一步认识 Java 程序 10
1.5.2 标识符和关键字 12
1.5.3 程序注释 13
1.5.4 对 Java 程序的再次说明 14
1.6 专题应用：为 Java 程序输入数据 14
自测与思考 18

第 2 章 Java 语法基础 21

2.1 基本数据类型 22
2.2 变量与常量 23
2.2.1 变量 23
2.2.2 常量 24
2.3 基本数据类型变量的赋值 25
2.3.1 整型变量的赋值 25
2.3.2 浮点型变量的赋值 27
2.3.3 字符型变量的赋值 27
2.3.4 字符串变量的赋值 29
2.3.5 布尔型变量的赋值 29
2.3.6 基本数据类型变量的默认值 30
2.4 表达式与运算符 30
2.4.1 表达式 30
2.4.2 运算符 31
2.4.3 运算符的优先级 35
2.5 扩展表达式和类型转换 36
2.5.1 扩展表达式 36
2.5.2 表达式的数据类型转换 36
2.6 专题应用：数据的随机产生与高效计算 38
自测与思考 42

第 3 章 程序流程控制 45

3.1 典型程序结构 46
3.2 选择结构 47
3.2.1 if 语句 47
3.2.2 switch 语句 50
3.3 循环结构 53

3.3.1 for 语句 53
3.3.2 while 语句 55
3.3.3 do-while 语句 57
3.3.4 嵌套循环 58
3.4 控制跳转语句 60
3.4.1 标号语句 60
3.4.2 continue 语句 60
3.4.3 break 语句 61
3.5 专题应用：典型流程控制算法 62
自测与思考 65
第 4 章 数组 69
4.1 数组的引入 70
4.1.1 引入数组的必要性 70
4.1.2 数组的概念 70
4.2 一维数组 71
4.2.1 一维数组的定义 71
4.2.2 一维数组的长度 74
4.2.3 创建一维数组的方法 74
4.2.4 一维数组应用举例 75
4.3 二维数组 77
4.3.1 声明二维数组变量 78
4.3.2 创建二维数组 78
4.3.3 二维数组的赋值与使用 79
4.3.4 二维数组的长度 79
4.3.5 非矩阵型二维数组 80
4.3.6 二维数组应用举例 82
4.4 多维数组 84
4.5 专题应用：数组元素的排序 84
自测与思考 87
第 5 章 Java 面向对象编程 90
5.1 面向对象程序设计概述 91
5.1.1 程序设计方法的发展 91
5.1.2 面向对象程序设计的特点 91
5.2 类和对象 92
5.2.1 定义类 93
5.2.2 成员变量 94
5.2.3 成员方法 95
5.2.4 创建、使用和销毁对象 96
5.2.5 方法中的参数传递 99
5.2.6 成员变量、局部变量和方法参数的区别 101
5.3 构造方法 101
5.3.1 构造方法的定义 102
5.3.2 对象的生成过程 104
5.3.3 this 关键字 105
5.4 类的继承 107
5.4.1 继承的概念 107
5.4.2 Java 继承的实现 108
5.4.3 访问权限修饰符 110
5.4.4 构造方法与继承 111
5.4.5 super 关键字 113
5.4.6 Object 类 113
5.5 类的多态 114
5.5.1 多态的概念 114
5.5.2 方法重载 114
5.5.3 方法覆盖 115
5.5.4 向上转型和动态绑定 117
5.6 final 关键字 119
5.6.1 终极变量 119
5.6.2 终极方法 121
5.6.3 终极类 121
5.7 static 关键字 122
5.7.1 静态变量 122
5.7.2 静态方法 124
5.8 抽象类 124
5.8.1 抽象方法 125
5.8.2 抽象类的定义及应用 125

5.9 接口 127
5.9.1 定义接口 127
5.9.2 实现接口 128
5.10 内部类 130
5.10.1 内部类的定义及访问 130
5.10.2 匿名内部类 132
5.11 专题应用：多类设计 133
自测与思考 136
第 6 章 Java 实用类库 139
6.1 Java 包及核心 API 140
6.1.1 包的概念和作用 140
6.1.2 创建包 140
6.1.3 引用包中的类 142
6.1.4 常用的 Java 类库 143
6.2 String 类和 StringBuffer 类 144
6.2.1 String 类 145
6.2.2 StringBuffer 类 148
6.3 集合接口与集合类 150
6.3.1 集合接口与相关实现类 150
6.3.2 常见集合类的用法 153
6.3.3 泛型集合 155
6.4 专题应用：开发一个应用项目的方法 157
自测与思考 160
第 7 章 异常与断言 162
7.1 异常 163
7.1.1 Java 异常机制 163
7.1.2 try-catch 语句 163
7.1.3 异常类的继承 165
7.1.4 Exception 异常 167
7.1.5 try-catch-finally 和 try-with-resource 结构 167
7.2 断言 169
7.2.1 断言的基本语法 169
7.2.2 断言在单元测试中的应用 171
7.3 专题应用：账户存款管理 172
自测与思考 175
第 8 章 Java 文件操作 177
8.1 File 类 178
8.1.1 创建文件对象 178
8.1.2 常用文件操作 178
8.2 文本文件的输入和输出 181
8.2.1 抽象字符流 181
8.2.2 文件字符流 183
8.2.3 缓冲字符流 184
8.3 字节文件的输入和输出 186
8.3.1 抽象字节流 186
8.3.2 文件字节流 188
8.4 数据流和对象流 189
8.4.1 数据流 189
8.4.2 对象流 191
8.5 专题应用：记录式文件的读写 192
自测与思考 195
第 9 章 Swing 程序设计 198
9.1 GUI 程序设计简介 199
9.2 Swing 容器 200
9.2.1 JFrame 容器 201
9.2.2 JPanel 容器 203
9.3 布局管理器 204
9.3.1 FlowLayout 布局管理器 205

9.3.2 BorderLayout 布局管理器 206
9.3.3 GridLayout 布局管理器 207
9.3.4 绝对定位 209
9.4 Java 事件处理 209
9.4.1 事件模型 209
9.4.2 事件类和事件监听器 211
9.4.3 事件适配器 218
9.4.4 事件监听器的实现方式 219
9.5 常用 Swing 组件 219
9.5.1 标签 220
9.5.2 按钮 221
9.5.3 文本组件 222
9.5.4 单选按钮和复选框 224
9.5.5 列表框 225
9.5.6 组合框 227
9.5.7 对话框 228
9.5.8 菜单 229
9.6 专题应用：GUI 的设计与实现 230
自测与思考 233

第 10 章 Applet 程序设计 236

10.1 Applet 简介 237
10.1.1 编写并运行第一个 Applet 程序 237
10.1.2 Applet 程序的执行流程与生命周期 238
10.1.3 Applet 类和 JApplet 类 240
10.1.4 Applet 程序的安全性 240
10.2 Applet 程序开发过程 240
10.2.1 使用 NetBeans 创建 Applet 程序 241
10.2.2 将 Applet 程序嵌入网页中 242
10.3 利用 Applet 程序展示多媒体 242
10.3.1 图形绘制 242
10.3.2 图像处理 243
10.3.3 声音输出 244
10.4 专题应用：图片轮换 245
自测与思考 247

第 11 章 多线程程序设计 249

11.1 线程的概念 250
11.1.1 程序与进程 250
11.1.2 进程与线程 250
11.1.3 Java 的多线程机制 250
11.1.4 线程状态和生命周期 251
11.1.5 线程调度与优先级 252
11.2 多线程程序的编写 252
11.2.1 继承 Thread 类 253
11.2.2 实现 Runnable 接口 254
11.3 线程同步、死锁与合并 255
11.3.1 线程同步 255
11.3.2 线程死锁 257
11.3.3 线程合并 257
11.4 专题应用：龟兔赛跑 258
自测与思考 260

参考文献 263

Chapter 1

第 1 章

Java 语言概述

Java 是目前非常流行的程序设计语言。本章首先介绍 Java 的发展历程、技术平台与语言特点，然后介绍 Java 程序的运行机制、运行环境、开发环境、开发过程以及程序结构和规范，最后专题介绍为 Java 程序输入数据的方法。

本章导学

- ✧ 了解 Java 的发展历程、技术平台和语言特点
- ✧ 理解 Java 程序的运行机制
- ✧ 掌握 Java 程序的编辑、编译和运行过程
- ✧ 掌握 Java 程序的基本结构和语法规范
- ✧ 熟悉常见的为 Java 程序输入数据的方法

1.1 初识 Java 技术

在智能手机、智能家电、网络服务等各种应用领域，无处不见 Java 技术的身影。Java 究竟是一种什么样的技术，为什么具有如此大的魅力？本节将通过 Java 的发展历程、简介 Java 技术平台以及分析 Java 语言的特点，为你揭示答案。

1.1.1 Java 发展历程

Java 语言的诞生既充满神奇色彩，又是顺应时代发展的产物。1990 年末，Sun 公司启动了 Green 项目，旨在将各种家用电子产品（如电视机、微波炉等）和消费性数字产品（如手机、PDA 等）连接起来以形成一个智能化的分布式服务系统。但当 Green 项目组成员在实现此类系统时，却发现当时主流的编程语言 C、C++均无法满足要求，因为他们面对的是型号各异的芯片。于是，Green 项目负责人 James Gosling（被尊称为 Java 之父）在 C++的基础上开发了一种全新的语言，并根据办公室窗外的橡树将这种语言命名为 Oak。但由于 Oak 这个名字已被注册，Green 项目组成员在边喝咖啡边讨论新名字的过程中，最终决定将这种全新的语言取名为 Java。实际上，Java 是印度尼西亚一个盛产咖啡的岛屿，中文名为爪哇。Green 项目组将这种全新的语言命名为 Java，寓意不仅在于 Java 语言给世人带来的是一种咖啡般的美好感觉，还在于使用 Java 进行编程就是一种如同喝咖啡般的美妙享受！

然而当时市场对消费电子智能化分布式服务系统的需求并不像 Sun 公司设想的那样乐观，Green 项目处于一种市场前景惨淡甚至面临夭折的尴尬境地。恰在这时，Internet 的兴起给 Java 带来了生机。James Gosling 意识到 Java 的结构中立特性正好适合开发运行于不同计算机平台上的各种网络软件。在网络浏览器 Mosaic 的启发下，Green 项目组采用 Java 语言开发了 HotJava 浏览器，并推出 Java Applet 技术，使互联网网页互动技术达到前所未有的高度，并据此确立了 Java 网络编程的地位。

1995 年 5 月 23 日，Sun 公司发布了 Java 语言和 HotJava 浏览器，并在 1996 年 1 月正式推出 Java 开发工具包（Java Development Kit，JDK）1.0 版本，这标志着一个新的网络计算时代的到来。在之后的 10 多年中，Java 语言随互联网的高速发展而不断演化，其功能越来越强大。尤其是 1998 年 12 月发布的 1.2 版本，将 Java 划分为 J2EE、J2SE 和 J2ME 3 个版本，以分别满足企业级、桌面级和移动设备应用开发的需要。2004 年 9 月，J2SE 1.5 正式发布，在该版本中，Sun 公司为 Java 增添了许多新的重要功能，为突出这些功能，Java 1.5 版本直接更名为 J2SE 5.0。从 Java SE 6.0 开始，Sun 公司取消了 Java 各种版本中的数字“2”。现在，Java 的 3 个技术分支被分别命名为 Java EE、Java SE 和 Java ME。

2010 年，Oracle 收购 Sun 公司，Java 随之开始了新的历程。2011 年 7 月 28 日，Oracle 公司发布 Java SE 7.0，并于 2014 年 3 月发布 Java SE 8.0，这说明 Java 的更新与改进始终没有停滞。尤其自 2017 年 9 月 21 日发布 Java SE 9.0 以来，Oracle 将 Java 版本的发布计划调整为每半年一次。有关 Java 各主要版本的发布时间如图 1-1 所示。

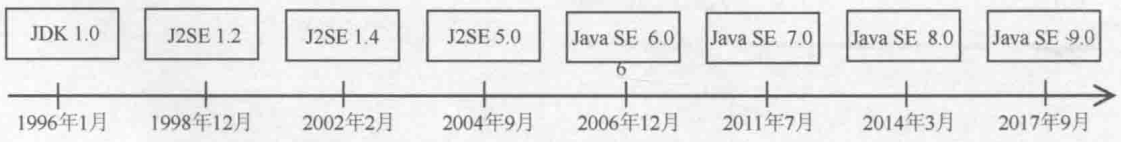


图 1-1 Java 各主要版本的发布时间

1.1.2 Java 技术平台

经过 20 多年的发展，Java 已从最初的网络互动展示技术扩展到各类应用。Java 已不再是一门单纯的程序设计语言，而是包含 Java 语言、各种应用编程接口（Application Program Interface，API）链接库、Java Server Page、Java Servlet、Java RMI、JavaBeans、JavaFX、Java Web Start 等众多技术的业界解决问题的平台。根据 Java 的主流应用方向，Java 包含以下 3 个主要的技术分支。

1. Java SE

Java SE 全称为 Java Standard Edition，即 Java 标准版。Java SE 包含 Java 语言核心和一些 Java 标准 API。Java SE 可用于开发图形用户界面（Graphical User Interface，GUI）、Java Applet、网络应用、数据库应用等传统桌面级应用程序。Java SE 是 Java 技术各类应用平台的基础和核心，也是本书重点讨论的对象。

2. Java EE

Java EE 全称为 Java Enterprise Edition，即 Java 企业版。Java EE 是在 Java SE 的基础上，通过引入 Java Server Page、Java Servlet、Java Mail、JavaBeans 等技术构建的一个多层次、分布式、以组件和 Web 为基础的企业级应用解决方案。

3. Java ME

Java ME 全称为 Java Micro Edition，即 Java 微型版。Java ME 主要用于开发智能手机、PDA 等消费性电子产品或嵌入式系统中广为使用的各类应用程序，如手机 App、Java 小游戏、记事程序以及其他各种控制程序。

1.1.3 Java 语言的特点

Java 语言是整个 Java 技术框架的基础，它具有简单、健壮、面向对象、跨平台、分布式、多线程、动态、API 类库丰富且功能强大等优良特性，这里仅就其中几个关键特性进行简要介绍。

1. 简单

Java 语言源于 C++，因此其语法规则和 C++类似。但 Java 语言对 C++进行了简化和提高，去除了指针和多重继承等复杂概念，转而通过对象引用和接口实现相似功能。同时，Java 通过提供自动垃圾回收机制实现对内存的自动管理，大大降低了程序员管理内存的负担。

2. 面向对象

面向对象是当今主流的程序设计方法。Java 是一种完全面向对象的程序设计语言，其支持面向对象的 3 大显著特征——封装、继承和多态。在 Java 程序中，所有信息和操作都被封装在类和对象中，类和类之间通过单重继承实现代码复用，并通过方法重载、覆盖、动态绑定等进一步提高代码复用的效率和灵活性。因此，使用 Java 语言可以开发各类非常复杂的应用系统。

3. 分布式

Java 是针对 Internet 的分布式环境而设计的。在 Java 中，内置了 TCP/IP、HTTP、FTP 等协议类库，用户可以很方便地通过 URL 访问网络上的对象，访问方式就像访问本地文件系统一样。同时，Java 还提供 Applet、Java Server Page 和 Servlet 等技术来丰富网页效果，实现动态页面，构建网络服务器功能。因此，Java 非常适合用于开发以网络为中心的各种分布式应用。

4. 跨平台

跨平台是 Java 语言的重要特性。Java 程序（*.java 文件）先经 Java 编译器被翻译成一种特殊的二进制文件，即字节码文件（*.class 文件），然后字节码文件由安装在各运行平台上的 Java 虚拟机（Java Virtual Machine，JVM）解释执行。对于不同的运行平台（如 Windows、Linux 等），所安装的 JVM 并不一样，但这些 JVM 都可以将同一个字节码文件正确翻译成本机上可执行的代码。可见，正是 Java 独特的“虚拟机”运行机制，使得字节码文件可以独立于具体的机器设备。同时，为了保证程序中使用的数据在各种硬件平台上保持一致性，Java 定义了独立于平台的基本数据类型和运算。Java 语言的跨平台特性保证了 Java 程序良好的可移植性，从而真正实现“一次编写，到处运行（Write Once，Run Anywhere）”的宏伟目标。

5. API 类库丰富且功能强大

无论是传统的桌面级应用，还是 Java EE 企业级应用及 Java ME 移动应用，Java 语言均提供了丰富且功能强大的各种 API 类库。有了这些 API 支撑，Java 开发就像“搭积木”式的编程，将已经提前准备好的各种组件进行合理组装，便可构建功能强大的各种应用程序。

1.2 理解 JVM、JRE 和 JDK

Java 语言的跨平台特性主要得益于 JVM 运行机制。JVM 究竟是如何成就 Java 程序的平台无关性的？与 JVM 密切相关的 Java 运行环境（Java Runtime Environment，JRE）和 JDK 又是怎样的概念，它们在 Java 程序开发和运行中扮演什么样的角色？理清 JVM、JRE、JDK 的区别与联系，是 Java 程序设计开发必须解决的首要问题。

1.2.1 Java 程序的运行机制

Java 程序是与平台无关的程序，可以实现“一次编译，到处运行”。这其中的奥秘正在于 JVM 的使用。图 1-2 给出了 Java 程序的执行过程。

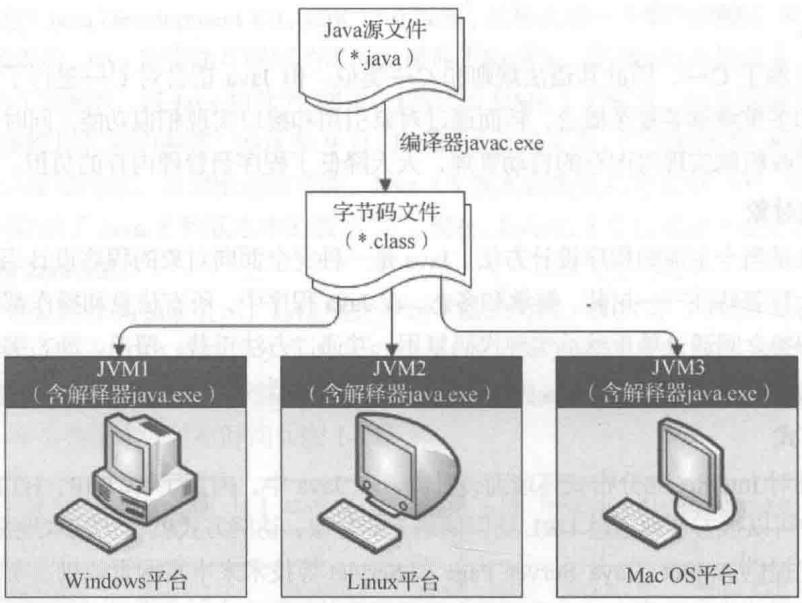


图 1-2 Java 程序的执行过程

从图 1-2 可见，Java 源程序的扩展名为“.java”，经过编译器（javac.exe）被翻译成扩展名为“.class”的字节码文件。字节码文件又称为类文件，它是一种独立于具体平台的、特殊的二进制文件。字节码文件并不直接面对具体的机器平台，而是面对 JVM，由 JVM 中的解释器（java.exe）负责解释转换成特定系统的机器代码并执行。这样，对于同一个字节码文件，便可实现在不同的平台上运行。要注意的是，每一种平台上的 JVM 和解释器都不相同，这些安装在本地机器上的 JVM，负责为字节码文件提供统一的虚拟运行平台，并和底层实际硬件进行沟通。JVM 如同一个“本地翻译”，可以将字节码文件的内容翻译为本地计算机能够理解的机器语言内容。

不难看出，JVM 是可运行 Java 字节码文件的“虚拟计算机”，它相当于给各种实际运行平台（Windows、Linux、Mac OS 等）包装上 Java 虚拟操作系统，Java 字节码文件便可直接运行在此虚拟平台上。因此，对于特定的计算机和各种消费电子产品，只要按照 JVM 规范实现了设备相应的 JVM，便可保证 Java 字节码文件在该设备上运行。

1.2.2 JRE

JRE 是执行 Java 程序必备的各种要素的集合。从 1.2.1 节的介绍可知，JVM 为 Java 程序的运行提供了一个虚拟的平台，Java 字节码文件正是在 JVM 的支持下才得以运行的。从这个意义上说，JVM 是 JRE 的基础和核心。除此之外，运行 Java 程序还需要别的东西吗？

其实，现代程序设计已经摆脱了“从零开始”的时代。可以设想，我们要设计一个类似 Microsoft Word 的应用程序，如果每一个按钮和菜单项都需要从头开始编写代码，那将是多么麻烦的事情！软件厂商通过提供各种已经设计好的程序组件和大量的 API 类库解决了这个问题，实现了程序代码的复用，提高了编程效率。就 Java SE 而言，Sun、Oracle 以及其他 Java 厂商已经为大家准备好了种类丰富、功能强大的 API 类库，如基本语言和工具库（java.lang 和 java.util）、输入和输出库（java.io）、窗口程序工具库（java.awt 和 javax.swing）等，直接调用这些类库中的组件，便可快速进行程序开发。在 JVM 将字节码文件转换为本地可执行代码的过程中，为了使其能够理解 Java 程序中使用的各种 API，JRE 还具备 Java SE API 类库。

此外，为了能够将设计好的各种 Java 程序方便、快速地安装和应用到客户端，JRE 还具备软件部署技术。JRE 的组成如图 1-3 所示。

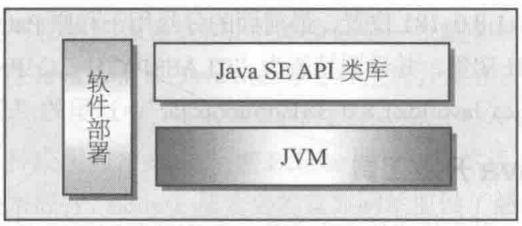


图 1-3 JRE 的组成

1.2.3 Java 开发环境

JRE 是运行 Java 程序的必备环境，而不是开发 Java 程序的环境。如果仅仅只运行 Java 程序，即只想使自己的机器能够理解并执行 Java 字节码文件，则只需安装 JRE 即可。Oracle 官方网站提供了单独的 JRE 下载。如果除了运行 Java 程序，还需要编写和测试 Java 程序，则需要 JDK 的支持。JDK 除了包括 JRE，还包括许多开发和测试 Java 程序的实用工具。例如，用于将 Java 源

程序翻译成字节码文件的编译器 javac.exe，用于解释执行字节码文件的解释器 java.exe，用于测试 Java Applet 小程序运行效果的 appletviewer.exe，用于依据程序说明格式生成 HTML 帮助文档的 javadoc.exe，用于将程序进行打包发布的 jar.exe 等。

因此，学习 Java 程序设计的第一步便是下载和安装 JDK，并对 JDK 进行合适的配置，以使计算机满足编写、编译和运行 Java 程序的要求。

1.3 准备 Java 开发环境

厘清了 JVM、JRE 和 JDK 彼此之间的关系后，接下来便可以搭建 Java 开发环境，为开始编写 Java 程序做好准备。Java 程序的编写与测试要做好哪些准备工作呢？常用于开发 Java 程序的工具软件又有哪些？下面进行具体介绍。

1.3.1 JDK 的下载、安装和配置

Java 程序的编译和运行离不开 JDK 的支持，因此，准备 Java 开发环境的第一步便是下载、安装和配置 JDK。这个过程本身并不复杂，详细过程可参阅配套实践教材的第二部分——Java 开发环境及程序调试。在这里，仅简要解释一下为什么要对 JDK 进行配置，以助读者理解 Java 程序的编译和运行过程。

JDK 的配置主要包括系统环境变量 Path 和类变量 CLASSPATH 的配置。其中，Path 变量用于设置 Windows 系统的默认搜索路径，即告诉操作系统默认情况下到什么地方去寻找编译器 javac.exe 和解释器 java.exe 等。显然，如果系统在 Path 指定的路径信息中找不到 javac.exe 和 java.exe 等，编写好的 java 程序便不能被成功编译和运行。另外，CLASSPATH 变量用于设置 JVM 搜索字节码文件的默认搜索路径，即告诉 JVM 默认情况下到什么地方去寻找字节码文件。在实际的 Java 项目中，一个主程序在编译或运行时通常会调用其他的字节码文件，或使用 Java SE API 类库中的某些字节码文件。有了 CLASSPATH 变量，JVM 便会根据该变量的设置去查找所需要的字节码文件，从而保证程序的正常编译和执行。

通常，Path 和 CLASSPATH 变量应做如下配置。

（1）在原 Path 变量后添加“;C:\Program Files\Java\jdk1.8.0_181\bin”（这里假设 JDK 被安装到 C:\Program Files\Java\jdk1.8.0_181 位置，最前面的分号用于和原 Path 路径列表进行分隔）。

（2）创建 CLASSPATH 变量，并设置其值为“CLASSPATH=.;C:\Program Files\Java\jdk1.8.0_181\lib\dt.jar;C:\Program Files\Java\jdk1.8.0_181\lib\tools.jar”，这里的“.”表示当前目录。

1.3.2 常见的 Java 开发工具

Java 源程序本质上是一种无格式的文本文件。在安装和配置好 JDK 后，便可使用 Windows 记事本之类的简易编辑器编写 Java 源程序，然后在命令行方式下使用 javac.exe 和 java.exe 分别对编写好的源程序进行编译和执行。但这样的程序开发方式效率极低、不易于查错，且要求用户对 DOS 操作系统的命令行方式有一定的了解。

为了提高 Java 程序的开发效率，简化程序设计和调试过程，人们开发了一些方便的工具软件和专门的集成开发环境（Integrated Development Environment，IDE），如 UltraEdit、NetBeans 和 Eclipse 等。

1. UltraEdit

从严格意义上讲，UltraEdit 并不能算是一个专门的 Java 开发工具软件。实际上，UltraEdit 是一个功能超强且灵活方便的文本编辑器，主要用于取代记事本程序，实现对各种文本文件（*.txt）、系统配置文件（*.conf）和十六进制文件的编辑与修改。但值得一提的是，UltraEdit 内置了对 C/C++、C#、Java、JSP、PHP、HTML 和 XML 等主流设计语言的支持，可以对这些语言的源代码文件进行很好的语法着色，以清晰地区分代码的各个组成部分。同时，通过对 UltraEdit 进行相应的配置，可直接在图形用户界面下编译和运行程序。正因为这些优点，许多人将 UltraEdit 作为常用的源代码编辑器。

本书编者也建议初学者开始学习 Java 编程时使用该工具软件（如学习本书第 1～4 章时），以利于理解和掌握 Java 程序的结构，熟悉 Java 程序关键词和常用 API 类库名称的准确写法。对于如何配置该软件，使之能够适于编辑、编译和运行 Java 程序，可参阅配套实践教材的第二部分。

2. NetBeans

NetBeans 最早源于 Sun 公司于 2000 年创建的开源代码计划，旨在向程序员提供一个一流的 Java 程序集成开发环境。利用 NetBeans，可以非常方便地进行 Java 程序的编辑、编译、调试和运行，还可实现清晰、直观的项目管理，以及灵活、高效的 Java 应用程序部署。

NetBeans 的主要特性包括语法着色、代码完成与提示、代码折叠、代码错误提示等，这些特性极大地提高了用户开发 Java 程序的效率。

NetBeans 支持多种语言和多种主流平台，且通过插件支持，还支持 C/C++、PHP、Ruby 和 Ajax 等程序的开发，这是开源软件共同的显著特征。目前，NetBeans 由开源组织 NetBeans 社区负责管理并提供技术交流，通过专门的社区网站可下载最新的 NetBeans 版本，并得到各种最新资讯。本书编者建议读者从第 5 章开始，尝试使用 NetBeans IDE 进行 Java 程序的开发。

3. Eclipse

Eclipse 是另外一个较为出色的开源代码集成开发环境，是在 IBM 公司早期开发工具 Visual Age for Java 的基础上发展起来的非常优秀的 Java 开发工具。就功能而言，Eclipse 和 NetBeans 差不多，同样具备语法着色、代码完成与提示等特性，并提供了方便、高效的项目管理等典型功能。

Eclipse 是一个成熟的可扩展的体系结构。它本身只是一个框架和一组服务，通过安装和集成各种各样的插件，可支持 Java Web 和 Java ME 等各种主流开发和应用，并可作为其他语言的开发工具。随着 Java 技术的广泛应用，许多著名的 IT 企业纷纷加入 Eclipse 架构的开发中，并发布了数量众多的支持各种应用的各式插件，使 Eclipse 的发展非常迅速。目前，Eclipse 由 Eclipse 基金会负责管理并提供技术服务，Eclipse 基金会的官方网站提供了最新的软件和插件下载及技术支持。

1.4 编写第一个 Java 程序

了解了 Java 程序的运行机制，并安装、配置好 Java 开发环境，接下来便可亲自体验开发 Java 程序的整个过程。

Java 程序的开发一般需经历编辑、编译和运行 3 个步骤。本节将通过一个简单的范例，说明设计 Java 程序的一般过程和方法。

1.4.1 Java 程序的编辑

Java 程序的编辑指利用 Java 开发工具创建 Java 源文件，并将其保存为扩展名为“.java”的文件。

【程序 1-1】第一个 Java 程序。

问题分析

首先，应按配套实践教材中的方法，安装 UltraEdit 并配置 UltraEdit 的菜单命令和工具栏。之后，在打开的编辑窗口中输入 Java 程序。需要注意的是，由于 Java 语言严格区分大小写，因此输入时要特别注意每个单词的大小写，且所有标点符号均为英文标点。程序输入完毕，可通过“文件→保存”命令或工具栏上的“保存”按钮将程序保存为 HelloJavaDemo.java 文件。这里，也要注意文件名中字母的大小写。编辑结束后，本例的结果如图 1-4 所示。

程序代码

```
01 //我的第一个 Java 程序，文件名为 HelloJavaDemo.java
02 public class HelloJavaDemo {
03     public static void main(String args[]) {
04         System.out.println("Hello,Java!");  //屏幕输出
05     }
06 }
```

程序 1-1 解析

运行结果

Hello,Java!

程序说明

（1）01，02，…，06 是为了说明程序而添加的行号，并非 Java 程序的组成部分，因此输入程序时不必输入。本书的所有范例程序均采用这种形式，以后不再说明。

（2）Java 程序总是从定义一个类开始的。在本例中，定义了名称为 HelloJavaDemo 的类，其内容从类名后的左花括号“{”开始，直到对应的右花括号“}”结束，即 02～06 行。Java 是一种纯粹的面向对象编程语言，“类（class）”和“对象（object）”是其核心概念，本书将在第 5 章进行详述。这里只需知道，定义类时采用 class 关键字，类还可以采用类似 public 的关键字进行描述，表示这个类是一个“公有”的类，可以在整个程序中进行访问。所谓关键字，指 Java 语言中固有的词汇，具有特别的含义，不能再作他用。同时，需要说明的是，当一个类被声明为 public 类时，对应的文件名必须和该类的名称一样。例如，在本例中，HelloJavaDemo 为公有类，则文件名只能为 HelloJavaDemo.java。显然，一个 Java 文件中最多只能有一个类被修饰为 public，否则文件将无法命名。

（3）一个可执行的 Java 程序，总包含且只能含有一个 main 方法。main 方法是程序运行的入口点，由 JVM 负责查找并运行。如果找不到该方法，程序将无法单独运行。“方法（method）”是 Java 语言中另一个重要的概念，用于描述一类对象在某方面的行为和功能。方法被定义在类中，含有 main 方法的类通常称为主类。定义方法时，也采用从左花括号“{”开始，直到对应的“}”结束，如本例的 03～05 行。作为程序起点的 main 方法，采用关键字 public、static 和 void 修饰，表明该方法是公有的、静态的和没有返回值的。也即，JVM 可以在没有对象被创建的情况下调用该方法，且方法执行完毕不会返回任何值。读者现在也许还不能较好地理解这些修饰词的含义，但随着对 Java 程序设计学习的深入，逐渐都能理解。这里，只需要知道“public static void