高等院校信息技术规划教材

Java语言程序设计实践教程

孙　涛　主编

清华大学出版社

高等院校信息技术规划教材

Java语言程序设计实践教程

孙　涛 主编
孙　涛 褚燕华 王丽颖 编著

清华大学出版社
北京

内 容 简 介

Java 程序设计语言是由 Sun 公司开发的面向对象的语言。它既可以开发一般的桌面程序，又可以用于 Web 编程，编译跨平台、跨语言的代码，简单易学，功能强大，并越来越受到人们的青睐。

本书以面向对象程序设计的基本概念为起点，由浅入深和循序渐进地介绍 Java 语言程序设计的基本概念和方法。本书是一本实践教程，注重结合实例讲解关键的技术和知识点，叙述详细，通俗易懂。全书共分 11 章，内容包括 Java 语言概论、Java 程序设计基础、数组与字符串、类与对象、接口与 Java API 基础、异常处理、输入输出流、图形用户界面、多线程、数据库连接、网络编程。

本书所有案例均已在 Eclipse 环境下调试通过，便于学习与教学。

图书在版编目（CIP）数据

Java 语言程序设计实践教程/孙涛主编. —北京：清华大学出版社，2012.1
（高等院校信息技术规划教材）
ISBN 978-7-302-27253-3

Ⅰ. ①J… Ⅱ. ①孙… 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(2011)第 226815 号

责任编辑：袁勤勇　赵晓宁
责任校对：时翠兰
责任印制：何　芊

出版发行：清华大学出版社　　地　　址：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　邮　　编：100084
社　总　机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62795954，jsjjc@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：三河市李旗庄少明印装厂
经　　销：全国新华书店
开　　本：185×260　　印　张：18.75　　字　数：436 千字
版　　次：2012 年 1 月第 1 版　　印　次：2012 年 1 月第 1 次印刷
印　　数：1～3000
定　　价：29.00 元

产品编号：043886-01

前言 foreword

面向对象程序设计方法是目前最广泛使用的程序设计方法。Java语言作为一种优秀的面向对象程序设计语言，具有跨平台、分布性、高性能、可移植等优点，已成为软件开发领域中的主流技术，被广泛用于网络应用程序以及各种电气设备的嵌入系统的开发。

本书是作者在总结了多年的开发经验与教学经验的基础上编写的。该书是定位于普通高等学校计算机科学与技术及相近专业的本科生的Java语言程序设计课程教材，也可作为相关专业研究生的Java语言程序设计课程教材，还可以作为业余程序设计人员的参考教材使用。

本书以“基础性，理论性，实践性，实用性”为编写宗旨，主要有如下特色：

1. 注重基础，内容全面

本书全面地介绍了Java语言程序设计的基本知识和运行机制，主要包括Java开发运行环境的介绍、Java语言基础、数组与字符串、类与对象、接口与Java API、异常处理、输入输出流、图形程序设计、多线程、数据库连接、网络编程。这些内容构成了Java语言程序设计基础必备的知识单元，帮助学习者建立牢固扎实的理论基础。

2. 案例丰富，理论联系实际，提高实践性，体现实用性

程序设计课程是理论与实践并重的课程。教学中不仅要注重学生对于基础知识的掌握，更要注重培养学生程序设计的基本技能，帮助学生建立程序设计的基本思维模式。本书采用理论基础案例与综合实践案例相结合的方式，在巩固学生理论知识的基础上，提高学生程序设计的实践能力和运用语言解决实际问题的能力。同时，具有实用性的案例，将极大地提高学生的学习兴趣，促进教学效果的提高。

全书共分11章。第1章主要介绍Java的特点，背景以及开发环境的使用。第2章主要介绍了Java的常量与变量，运算符和控制语句等基本语法要素。第3章讲解了数组与字符串的语法环节，介绍了数组与字符串的实用技术。第4章讲解了类与对象、封装、继

承、多态、抽象类以及包的使用。这一章是面向对象程序设计的重要基础环节。第 5 章介绍了接口的定义及使用方法。第 6 章讲解了 Java 的异常处理机制以及异常处理过程，同时还介绍了 Java API 文档的查阅和使用方法。第 7 章主要讲解了 Java 的输入输出流技术。第 8 章讲解了图形程序设计，主要介绍了图形界面设计中的常用组件的添加，容器的使用和事件的处理机制。第 9 章讲解了多线程技术，帮助学生掌握线程的含义以及多线程程序设计方法。第 10 章讲解了数据库连接技术，介绍了怎样通过 JDBC 操作数据库。第 11 章讲解了 Java 网络编程中的一些重要技术，涉及 URL，TCP Socket，UDP 等重要内容。

本书由孙涛任主编，第 1 和第 2 章由王丽颖编写。第 3～第 6 章由孙涛编写。第 7～第 11 章由褚燕华编写。由于作者水平有限，对于书中存在的不妥之处，敬请读者朋友批评指正。

本教材获得 2011 年度内蒙古科技大学教材基金资助。

本书声明：本书示例中涉及的人名等数据信息均为虚拟，与实际无关。

编　者

2011 年 9 月

目录 Contents

第1章 chapter 1

Java语言概论

本章大纲

本章教学目标

- 了解Java语言特点；
- 熟悉Java的开发环境；
- 掌握Java程序的开发流程；
- 掌握Java程序框架。

本章教学难点

- Java语言的特点；
- Java程序的开发流程。

1.1 知识点讲解

1.1.1 Java的发展和特点

1. 程序设计的基本概念

人们要使用计算机，就是用计算机处理各种不同的问题，为了实现问题的求解，需要把问题和求解步骤告诉计算机，并要求计算机返回计算结果。把要解决的问题变成计算机能够接受且能执行的一组指令，即编排一系列计算机解题步骤，这就是程序。完整地说，所谓程序，就是为完成某个任务而设计的，由有限步骤所组成的一个有机的指令序列。简单讲，程序就是指令的序列。程序设计语言就是编制程序的工具，人与计算机之间通过程序设计语言进行通信。

同人类的自然语言一样，程序设计语言目前多达上千种，常用的也有几十种，仅就程序设计语言如何进行分类，就有很多不同的讨论结果。多数人认为程序设计语言分为三大类：面向机器的语言、面向问题的语言和面向对象的语言，接下来就使用这个分类标准对程序设计语言进行介绍。

1）面向机器的语言

计算机只能识别0和1二进制代码组成的机器指令。机器语言是0和1二进制代码

的机器指令的集合，也就是由硬件所能实现的语义序列集合。面向机器的程序设计语言是计算机能够直接识别的语言，就是使用机器指令编写的程序。

因为指令都是 0 和 1 的一组序列，使用机器语言编程极端困难，于是将机器指令、寄存器、存储器地址等物理概念使用人类容易理解的助记符号表示，这就是汇编语言。汇编语言与机器语言相比较，具有较好的可读性和可操作性。由于汇编语言是助记符语言，一条汇编指令对应一条机器语言指令，因此用汇编语言编写的程序运行效率高。

2）面向问题的语言

面向问题的语言比汇编语言和机器语言更类似于正常的人类语言，通常将其称为高级语言。如果这样说，汇编语言是用语言中的单词编写程序，那高级语言就是将单词使用某种语法组合成语句，用语句编写程序。因此，高级语言更容易表示出复杂的逻辑关系，程序的编写更加容易，可以使用这些程序设计语言开发一些规模较大或较复杂的程序。然而，计算机只能识别机器语言，所有的程序在计算机运行之前，都必须由编译器翻译成机器语言，高级语言编写的程序中，一条语句对应着多条机器指令，与用汇编语言编写的程序比较起来，高级语言编写的程序可能运行的时间更长，占用的内存更多。

在这个时期，问题求解和算法是程序设计的重点。程序设计的目标是尽量小的存储空间和尽量快的运行时间，并不特别注意算法与数据结构的依赖关系，算法和数据处于相对独立的状态。当时，程序设计中引入了模块化的概念。模块化的思想是将应用程序划分为若干个规模适度的模块，每个模块独立命名、独立编译。模块化带来了模块内信息隐藏的重要概念，模块中的变量可以与其他模块中的变量同名而不至于产生混淆；模块间通过全局变量共享数据；模块间通过外部变量、公共变量实现通信等。另外，研究证明，仅用顺序、分支、循环三种基本控制结构即能构造出任何单入口、单出口的程序块。由于块与块可以嵌套，用基本控制结构组织的程序最终可以表示为一系列顺序串联的块的形式。这些研究结果与模块化的概念相结合，形成了采用自顶向下、逐步求精的设计步骤和单入口、单出口基本控制结构的结构化程序设计方法。Pascal 语言和 C 语言都是结构化程序设计语言，同时它们也是过程化程序设计语言。

过程的概念来自子程序。由于子程序是相对于其调用程序而言的，不如用“过程”描述实现算法的程序块更为准确和贴切。过程是过程化语言构建程序的基本构件。结构化程序设计的研究中有一个重要的结论，如图 1-1 所示，程序表示为程序＝算法＋数据结构。程序设计时先考虑解决问题的算法，再设计合适的数据结构使算法得以实现。它强调算法和数据结构之间的独立性，体现了一种以数据结构为中心的观念。数据库技术和数据库管理语言也体现了程序设计以数据结构为中心的这种特点。这种以数据结构为中心的缺点是整个程序中许多重要的过程和函数的实现依赖于一个或几个关键的数据结构，如果关键数据结构中的数据有所改变，则会影响到整个系统，许多过程和函数必须修改或是重写，显然，这是不利于程序的维护和扩展的。

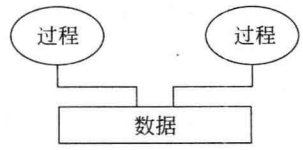

图 1-1　面向过程的程序设计

3）面向对象的程序设计

面向对象的程序设计方法继承了结构化、过程化、模块化等方法的所有积极成分，创

造性地引入了“对象”的概念。如图 1-2 所示，对象是构建程序的基础，是由数据结构和对数据结构的操作(方法)封装而成的一个整体。封装使得算法和数据结构的关系成为相互依存的关系，而不再是过程化程序设计中算法对数据结构的单方依赖关系。在面向对象的软件系统中，以“对象＝数据结构＋算法，程序＝对象＋对象＋…＋对象”取代了“程序＝数据结构＋算法”的传统程序设计模式，是对传统的程序设计方法的颠覆，引发了程序设计观念的革命。

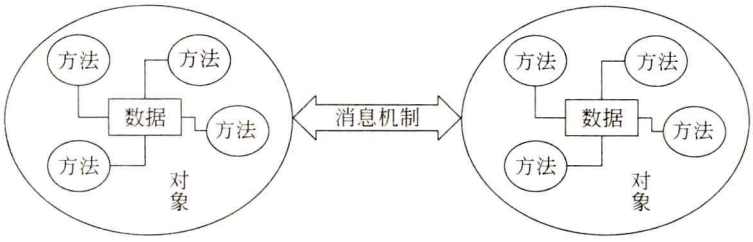

图 1-2 面向对象的程序设计

2. 面向对象程序设计语言的发展

面向对象方法采用数据抽象与隐藏、层次结构体系、动态绑定等概念和措施，提供一种模拟人类认知方式的软件系统建模方法，带来了系统的安全性、可扩充性、代码重用、易维护等特性。

20 世纪 60 年代，由挪威计算中心 Ole-Johan Dahl 和 Kyisten Nygaard 主持开发的 Simula 67 被公认为是面向对象语言的先驱。

20 世纪 70 年代，由美国国防部资助开发的 ADA 语言，可以支持抽象数据类型，在面向对象语言发展中占有一席之地。

20 世纪 70～80 年代，美国 Xerox 公司 Palo Alto 研究中心(PARC)的 Alan Kay、Adale Goldberg 和 Dan Ingans 等人主持开发的 Smalltalk 语言正式使用了“面向对象”这个术语。它的问世标志着面向对象程序设计方法的正式形成。

20 世纪 80 年代中期，Simular67、Smalltalk_80 等语言仅仅局限在学术界小范围内使用。面向对象语言对计算平台的特殊要求使这些语言难以被软件开发商和程序员所接受。贝尔实验室的 Bjarne Stroustrup 及其研究小组在当时最受欢迎的 C 语言基础上开发的 C++ 语言，引入了对面向对象概念的支持。

C++ 语言以其与 C 语言兼容、高运行效率等特性，使面向对象程序设计技术受到软件界的广泛关注，使面向对象技术进入一个全面发展的时期。尽管 C++ 只能算作一种混合式面向对象语言，也就是说 C++ 既可以进行面向过程的结构化的程序设计，也可以采用面向对象的程序设计方法。也正是由于 C++ 的这个特点，大量 C 程序员通过 C++ 的帮助迅速掌握了面向对象的概念和方法，全面促进了面向对象技术的应用。

Java 语言是由 Sun Microsystem 公司推出的一种纯面向对象语言。Java 从 C++ 语言中继承了大量的语言成分，抛弃了 C++ 语言中容易引起问题的功能，将面向对象、平台无关性、稳定性、安全性等优点集于一身，提供了一个良好的程序设计环境，因而成为适合于面向对象的程序设计语言。

3. Java 的发展

1991 年 4 月，Sun 公司的 James Gosling 领导的绿色计划(Green Project)开始重点发展一种分布式系统结构，使其能够在各种消费性电子产品上运行。Green 项目组的成员在开始的时候使用 C++ 语言来完成这个项目。但是，很快发现 C++ 语言存在很多不足，有必要研发一种新的语言来替代它，这种语言最初被命名为 Oak，17 个月之后，整个系统完成，这个系统是更类似于机顶盒式的操作系统。不过，在当时市场不成熟的情况下，他们的项目没有获得成功。1995 年，互联网在世界上蓬勃发展，Sun 公司发现 Oak 语言所具有的跨平台、面向对象、安全性高等特点非常符合互联网的需要，于是对 Oak 进行了一些小规模的改造。终于在 1995 年 3 月 23 日，Java 诞生了。

Sun 公司虽然在 1995 年推出了 Java，这时的 Java 只是一种程序设计语言，要进行复杂的应用程序的开发，必须要有一个强大的开发库的支持。Sun 在 1996 年 1 月 23 日发布了 JDK 1.0。这个版本包括运行环境(即 JRE)和开发环境(即 JDK)两部分。运行环境包括了核心 API，集成 API，用户界面 API，发布技术和 Java 虚拟机(JVM)5 个部分。而开发环境还包括编译 Java 程序的编译器(即 javac)。在 JDK1.0 版本中，除了用于开发图形用户界面的 API(AWT)外，其他的库并不完整。

Sun 公司在 1996 年早期发布了 Java 第一版。人们认识到 Java 1.0 并不适合做真正的应用开发。它的后续版本 Java 1.1 很快填补了前者的不足，极大地提高了 Java 的软件开发能力并为 GUI 编程增加了新的事件模型。

1998 年 12 月 Java 1.2 版本发布，Sun 公司将其称为 Java2 标准版软件开发工具箱 1.2 版(J2SE SDK 1.2)，以区分和以前版本的明显不同。除了 Java 的“标准版”J2SE 之外，Sun 公司还推出了两种其他的版本：用于移动电话等嵌入式设备的“微型版”——J2ME；用于服务器端处理的“企业版”——J2EE。J2ME 主要是为消费性产品而设计的，如 PDA、手机等，这类消费产品的一个非常重要的特点是内存较小。J2SE 主要是针对 PC、笔记本计算机而设计的。J2EE 主要是针对企业级的、服务器端的高端应用而设计的。本书使用的是标准版——J2SE。

4. Java 的特点

Sun 公司的“Java 白皮书”对 Java 做了如下定义：Java 是一种简单的、面向对象的、分布式的、解释执行的、健壮的、安全的、结构中立的、可移植的、高效率的、多线程的和动态的语言。

Sun 公司对 Java 的定义充分展示了 Java 的如下几个特点。

1) 简单

Java 是一种简单的语言。Java 在 C/C++ 语言的基础上开发，继承了 C/C++ 语言的许多特性，使用了许多与 C/C++ 语言同样的语言结构。但是又把 C/C++ 语言中一般程序员很少使用的、容易出错的特征去掉了。Java 略去了运算符重载、多重继承等概念；不支持 goto 语句，代之以提供 break 和 continue 语句以及异常处理。Java 不使用主文件，它免去了预处理程序。因为 Java 是面向对象的，所以 C 语言中的结构 struct 和 union 已被删去。Java 还通过自动无用单元收集(垃圾回收机制)，大大简化了程序员的内存管理

工作，使程序员不必为内存的管理问题而烦恼。

最重要的简化是Java不使用指针。指针在C/C++语言的编程中是使用最灵活也是最容易出错的部分。Java没有结构，数组和字符串都是对象，所以不需要指针。从而把程序员从“指针悬空”、“无效指针引用”和“存储泄露(leak)”这些困扰中解脱出来。

另外，JDK还提供了丰富的基础类库，具有C/C++语言编程经验的程序员都会对这些基础类库很熟悉，无须经过长时间训练即可掌握它。

2）面向对象

面向对象可以说是Java最重要的特性。Java语言的设计完全是面向对象的，不支持类似C语言那样的面向过程的程序设计技术。Java将待解决的现实问题转换成一组分离的对象，这些对象彼此之间可以进行信息交互。一个对象包含了对应实体应有的信息以及访问和改变这些信息的方法。通过这种设计方式，设计出来的程序更易于改进、扩展、维护和重用。Java语言提供类、接口和继承等概念。Java语言只支持类之间的单继承，但支持接口之间的多继承，并支持类与接口间的实现机制。Java语言全面支持动态绑定，而C++语言只对虚函数使用动态绑定。

3）分布式

Java是一种分布式的语言。传统的基于客户端/服务器(C/S)架构的程序，是采用客户端向服务器提出服务请求，服务器再根据要求执行适当的程序并将结果返回的方式，所以，服务器负荷较重。Java采用Java虚拟机架构，可将许多工作直接交由终端处理，因此，数据可以被分布式处理。此外，Java类库的运用，大大减轻了网络传输的负荷。Java类库包含了支持HTTP和FTP等基于TCP/IP协议的子库。Java应用程序可凭借URL打开并访问网络上的对象，其访问方式与访问本地文件系统几乎完全相同。

4）编译和解释性

Java是高效解释执行的语言。前面已经提到高级语言编写的程序必须转换为机器语言才能在计算机上执行。但是，不同的计算机系统所使用的机器语言不同。为了实现“一次编译，随处运行”(Write Once，Run Anywhere)的目标，Java程序在编译时并不直接编译成特定的机器语言程序，而是编译成与系统无关的“字节码”(Byte Code)，由Java虚拟机(Java Virtual Machine，JVM)执行。任何系统只要安装了Java虚拟机后，就可以执行Java程序。

JVM能直接在任何机器上执行，为字节码提供运行环境。当JVM解释执行Java程序时，Java实时编译器(Just-In-Time，JIT)会将字节码翻译成目标平台对应的机器语言的指令代码。实际上Java的前半阶段是编译执行，后半阶段是解释执行。

很多程序设计语言尝试解决跨平台问题的方案对性能要求都很高。其他解释执行的语言系统，如BASIC，有无法克服的性能缺陷。但是，Java却可以在非常低档的CPU上顺畅运行。这是因为JVM能够直接使用JIT编译技术将字节码转换成高性能的本机代码。事实上，随着JIT编译器技术的发展，Java程序的运行速度已接近于C++语言。因而，“高效且跨平台”对Java来说已不再矛盾。

5）健壮性

Java是健壮的语言。为了更好地理解Java的健壮性，先讨论一下传统编程环境下程

序设计失败的一个主要原因：内存管理错误和误操作引起的异常或运行时异常。

在传统的编程环境下，C/C++ 语言中，内存管理是一项困难的工作。例如，必须手工分配、释放所有的动态内存。如果忘记释放原来分配的内存，或释放了其他程序正在使用的内存时，系统就会出错。同时，在传统的编程环境下，异常情况可能经常由“除数为零”、“Null 指针操作”、“文件未找到”等原因引起，必须编写一组语句进行处理。

在 Java 程序设计中，Java 通过自行管理内存分配和释放的方法，从根本上消除了有关内存的问题。Java 提供垃圾收集器，可自动收集闲置对象占用的内存。Java 提供面向对象的异常处理机制来解决异常处理的问题。通过类型检查、Null 指针检测、数组边界检测等方法，在程序开发早期就会发现程序的错误。

6）安全

Java 是安全的网络编程语言。Java 提供了一系列的安全机制以防恶意代码攻击，确保系统安全。Java 的安全机制分为多级，包括 Java 语言本身的安全性设计以及严格的编译检查、运行检查和网络接口级的安全检查。

7）结构中立

Java 是结构中立的语言。Java 的设计目标是要支持网络应用。一般而言，网络是由许多不同的系统构成，包括各种不同的 CPU 与操作系统。为了让 Java 应用程序能够在网络上任何地方执行，其编译器会产生一种具备结构中立性的对象文件格式，即 Java 字节码文件。Java 字节码可在任何安装了 Java 虚拟机的平台上运行。

8）可移植

Java 开发的程序具有可移植性。结构中立是确保程序可移植的必要条件，此外还需要很多其他条件的配合。Java 语言规范中没有任何“同具体实现相关”的内容，解决了所有可能会影响到 Java 可移植性方面的问题。例如，不同的操作系统和 CPU 对数据类型及长度都作了不同的定义，在 Windows 3.1 中整型数(integer)为 16 位，在 Windows 95 中整型数为 32 位，这给程序的可移植性带来了一定的困难，Java 通过定义独立于平台的基本数据类型及其运算，使数据在任何硬件平台上保持一致。

9）高效率

Java 是高效率的语言。每一次的版本更新，Java 在性能上均做出了改进。在历经数个版本变更后，Java 已经拥有与 C/C++ 语言同样甚至更好的运行性能。Java 可以在运行时直接将目标代码翻译成机器指令。使用 JVM 在 1 秒钟内可调用 300 000 个过程，这已经同 C/C++ 语言不相上下。

10）多线程

Java 是支持多线程的语言。多线程是一种应用程序设计方法。线程是从进程里分出来的、更小的、独立的进程，使得在一个程序里可同时执行多个小任务。多线程带来的好处是具有更好的交互性能和实时控制性能。但采用传统的程序设计语言(如 C/C++ 语言)实现多线程非常困难。Java 实现了多线程技术，提供了一些简便地实现多线程的方法，并拥有一套高复杂性的同步机制。

11）动态

Java 语言具有动态特性。Java 动态特性是其面向对象设计方法的扩展，允许程序动

态地装入运行过程中所需的类，这是C++语言进行面向对象程序设计所无法实现的。在C++语言程序设计过程中，每当在类中增加一个实例变量或一种成员函数后，引用该类的所有子类都必须重新编译；否则将导致程序出错。Java采取如下措施来解决此类问题。

Java编译器不是将对实例变量和成员函数的引用编译为数值引用，而是将符号引用信息在字节码中保存后传递给解释器，再由解释器在完成动态连接类后，将符号引用信息转换为数据偏移量。存储器生成的对象不在编译过程中决定，而是延迟到运行时由解释器确定。这样，对类中变量和方法进行更新时就不至于影响现存的代码。解释执行字节码时，这种符号信息的查找和转换过程仅在一个新的名字出现时才进行一次，随后代码便可以全速执行。在运行时确定引用的好处是可以使用已被更新的类，而不必担心会影响原有的代码。如果程序连接了网络中另一系统的某一类，该类的所有者也可以自由地对该类进行更新，而不会使任何引用该类的程序发生错误。

Java还简化了使用一个升级的或全新的程序的方法。如果系统运行Java程序时遇到了不知如何处理的程序，Java能自动下载所需要的功能程序。

因此，Java是一种比C/C++语言更具动态特性的语言，在设计上强调为运行中的运算环境提供动态支持。Java在运行时为模块与模块之间建立连接，并能够更直接地运用面向对象设计体系。程序库可以自由地增加新方法和实例变量，而不会对用户产生任何影响。

1.1.2 Java的核心特征

1. Java虚拟机

Java语言的一个非常重要的特点就是平台无关性。使用Java虚拟机是实现这一特点的关键。高级语言程序如果要在不同的平台上运行，往往需要编译成不同的目标代码。而引入Java虚拟机后，Java语言在不同平台上运行时不需要重新编译。Java语言使用Java虚拟机屏蔽了与具体平台相关的信息，即在计算机和编译程序之间加入了一个抽象的虚拟的机器。这台虚拟的机器在任何平台上都提供给编译程序一个共同的接口。编译程序只需要面向虚拟机，生成虚拟机能够理解的代码，然后由解释器来将虚拟机代码转换为特定系统的机器码执行。Java语言编译程序生成能够在Java虚拟机上运行的目标代码(字节码)，Java虚拟机在执行字节码时，把字节码解释成具体平台上的机器指令执行。

Java工作流程如图1-3所示，先用文本编辑器创建与编辑Java源程序，Java源程序用java作为扩展名。源程序完成后，就可以进行编译。需要使用Java编译器将源程序翻译成Java虚拟机能够识别的指令集合，并以字节码的形式保存在文件中，字节码文件以class作为扩展名。最后，Java解释器，读取字节码，取出指令并且翻译成计算机能执行的代码，完成运行过程。字节码运行的平台是Java虚拟机，只要计算机上安装有Java虚拟机，不论采用哪种操作系统，硬件配置如何，产生的运行结果都相同。

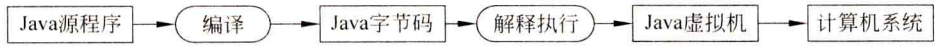

图1-3 Java的工作流程

2. 垃圾回收机制

在程序的执行过程中，部分内存在使用过后就处于废弃状态，如果不及时进行回收，就会导致内存泄露，甚至导致系统崩溃。在 C++ 语言中对象所占用的内存空间一直被占用，直到在程序运行结束后释放或者由程序员进行内存回收，程序员需要在编写程序时将不再使用的对象所占有的内存释放，但是这种人为的管理内存释放的方法却往往由于程序员的疏忽而导致内存无法回收，同时也增加了程序员的工作量。而在 Java 运行环境中，这个工作由垃圾回收器来负责。当没有对象引用指向原先分配给某个对象的内存时，该内存便成为垃圾，Java 提供了一个系统级的线程，也就是垃圾回收器，专门跟踪内存的使用情况，定期检测不再使用的内存并释放该内存空间，同时垃圾收集器还可以整理内存碎片。由于创建对象和垃圾收集器释放丢弃对象所占的内存空间，内存会出现碎片。整理出的内存会分配给新的对象。垃圾回收线程在 Java 程序的生命周期内自动执行，避免了内存的泄露，也减轻了程序员的工作量。

3. 代码安全

Java 语言是强类型语言，要求变量的使用要严格符合定义，所有变量都必须先定义后使用。因此，程序中使用的每个变量和表达式都有确定的类型。并且，Java 编译器对所有的表达式和参数都要进行类型相容性的检查，以确保类型是匹配的，如果类型不匹配就会出现语法错误，这些语法错误必须全部被纠正后才能运行程序。此外，Java 对内存访问进行了严格的限制。Java 编译器在编译期间并不分配内存，而是在运行期间由解释器分配内存。这样，编程人员就无法通过指针非法访问内存。在 Java 运行系统看来，任何代码都不可信赖，因此必须将代码提交字节码检验器进行检查。字节码检验器将代码传送给一个简单的规则检验程序，进行如下检测：

① 不存在伪造的指针。

② 未违反访问权限。

③ 严格遵循对象规范来访问对象。

通过语言的内在安全机制，再加上对生成代码的运行检查，Java 建立了一套严密的安全体系。

Java 提供了一种面向分布式计算的编程环境。Java 的运行系统内置了可预防病毒和防止破坏文件的保护机制。在网络环境下，Java 禁止未授权的代码更改 Java 的可执行代码。在网络接口级，Java 网络软件包提供了处理各种网络协议（如 FTP、HTTP 和 Telnet 等）的用户接口，用户可按自己的需求来设置网络访问权限。此外，Sun 公司承诺，Java 的未来版本将采用公开密钥法以及其他加密技术，核实从网络上传输过来的代码的源主机及该代码的完整性。

1.1.3 Java 的工作平台 Java Develop Kit

1. 在 Sun 公司的网站下载 JDK

在浏览器的地址栏输入网址 java. sun. com，进入 Sun 公司网站的主页，在右侧 Software Downloads 栏下单击 Java SE 项，如图 1-4 所示。