普通高等教育“十一五”国家级规划教材

中国高等学校信息管理与信息系统专业规划教材

Java程序设计基础（第5版）

陈国君 主编

根据教育部管理科学与工程类学科专业教学指导委员会主持鉴定的《中国高等院校信息系统学科课程体系》组织编写

与美国ACM和IEEE/CS Computing Curricula 2005同步

清华大学出版社

普通高等教育“十一五”国家级规划教材

中国高等学校信息管理与信息系统专业规划教材

Java程序设计基础（第5版）

陈国君 主编

陈磊 邹林达 李梅生 刘洋 鲜征征 陆寄远 编著

清华大学出版社

内 容 简 介

Java是近年来最流行的计算机程序设计语言。本书全面系统地介绍Java语言的特点及应用技术，内容上以Java的基础程序设计、面向对象程序设计和事件处理为三大主线，采用浅显易懂的语言和丰富简单的实例，完整地介绍了Java面向对象程序设计的重点和难点。本书共分18章，其中第1～5章介绍程序设计基础；第6～11章介绍面向对象程序设计；第12章介绍泛型和容器类；第13章和第14章介绍界面设计和事件处理；第15章介绍绘图程序设计；第16章介绍Applet程序设计，第17章介绍Java数据库编程，第18章介绍Java网络编程。

本教材在取材上特别注意教材的体系，其特色是结构合理、概念清楚、思路清晰、突出重点、分解难点、循序渐进、通俗易懂。尤其在结构上特别注重前后内容的连贯性，力求抓住关键、突出重点、分解难点，体现“理论性、实用性、技术性”三者相结合的编写特色。对每个知识点不但能告诉读者要怎么做，而且还要告诉读者这样做的原因和道理。

本书既可作为高等院校计算机及其相关专业的教学用书，也可作为各学校程序设计公共选修课的教材，同时也可用作职业教育的培训用书和Java初学者的入门教材或为具有一定Java编程经验的开发人员学习使用。

图书在版编目(CIP)数据

Java程序设计基础/陈国君主编. --5版. --北京：清华大学出版社，2015
中国高等学校信息管理与信息系统专业规划教材
ISBN 978-7-302-39402-0

Ⅰ. ①J… Ⅱ. ①陈… 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035085号

责任编辑：刘向威　李　晔
封面设计：常雪影
责任校对：时翠兰
责任印制：李红英

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：010-62770175　　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
课 件 下 载：http://www.tup.com.cn，010-62795954

印 装 者：北京密云胶印厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：28　　**字　　数**：680千字
版　　次：2006年1月第1版　2015年5月第5版　　**印　　次**：2015年5月第1次印刷
印　　数：1～2000
定　　价：49.00元

产品编号：063142-01

前言

本教材自出版以来一直受到广大读者的好评，市场反映非常热烈。尤其是本教材的第3版被“中国书刊发行业协会”评为2011年度全行业优秀畅销教材后，虽已多次印刷，但均已售罄。为了能适应科学技术的发展和计算机教学的需要，清华大学出版社和本书作者在征求广大读者意见和建议的基础上，决定修订再版，以便更好地满足广大读者的需求。本版在总结了以前版本的经验基础上，根据读者的建议增加了泛型、容器类与数据库编程等内容，使得该版教材在体系结构、内容组织、语言表达等方面都更加完善。该版中的所有例题完全采用Swing GUI组件重新编写，每个例题都突出一个编程的知识点。除保持了由浅入深、循序渐进的优点外，还对教学过程中学生和教师遇到的问题进行了详细的讲解，彰显了突出重点、分解难点的特色，使学生对学习Java编程产生兴趣，而兴趣又成了学习Java语言的动力，使学生在学习的乐趣中掌握Java的基本编程技巧。这种良性循环归功于教材内容的精选和组织结构的合理性，衷心希望本教材能成为广大读者的良师益友。本教材正是由于不断优化的知识体系，通俗易懂的讲解方式，对知识点的透彻分析和灵活实用的举例而深受读者的欢迎，这也是催生该书再版的主要原因。由于Java技术的内容丰富、结构复杂，所以从中抽出基本的内容，并能以通俗的方式介绍给读者并非易事，所以本教材难免存在不尽人意的地方，因此希望广大读者继续能对本教材提出合理化建议，使本教材更加完善。由于计算机技术发展得很快，加之作者水平有限，书中难免有不足之处，欢迎广大读者斧正。

书中所有例题全部在JDK 7环境下编译通过并运行。

本版教材由陈国君、陈磊、邹林达、李梅生、刘洋、鲜征征、陆寄远共同修改完成。

本教材的再版，得到了清华大学出版社的大力支持，在此本书全体作者对清华大学出版社的大力支持，尤其是对索梅编审的热心关注、建议与指导表示衷心的感谢！

作　者

2015年1月

目录

第 1 章　Java 语言概述

本章主要内容：

- Java 语言的特点。
- Java 源文件(.java)与 Java 字节码文件(.class)。
- Java 应用程序和 Java 小程序的主类。
- Java 虚拟机。
- Java 程序的种类和结构。
- Java 应用程序和 Java 小程序的差异。

Java 语言是一种简单易用、完全面向对象、与平台无关、安全可靠、主要面向 Internet 的开发工具。

1.1　Java 语言的诞生与发展

Java 语言诞生于 20 世纪 90 年代初期，从它正式问世以来，其快速发展已经让整个 Web 世界发生了翻天覆地的变化。

Java 语言的前身是 Sun Microsystems 公司(Sun 公司于 2009 年 4 月被 Oracle 公司收购)开发的一种用于智能化家电的名为 Oak(橡树)的语言，它的基础是当时最为流行的 C 和 C++语言。但是，由于一些非技术上的原因，Oak 语言并没有得到迅速的推广。直到 1993 年，WWW(万维网)迅速发展，Sun 公司发现可以利用 Oak 语言的技术来创造含有动态内容的 WWW 网页，于是已受人冷落了的 Oak 语言又被重新的开发和改造，并将改造后的 Oak 语言改名为 Java 语言，Java 是太平洋上的一个盛产咖啡的岛屿的名字。终于，在 1995 年，Java 这个被定位于网络应用的程序设计语言被正式推出。

由于 Java 语言功能强大，其问世后不久，即被业界广泛接受，于是 IBM、Apple、DEC、Adobe、HP、Oracle、Toshiba、Netscape 和 Microsoft 等大公司均购买了 Java 语言的许可证。Microsoft 还从其 Web 浏览器 Internet Explorer 3.0 版起开始增加了对 Java 语言的支持。同时，众多的软件开发商也开发了许多支持 Java 的产品。在目前以网络为中心的计算机时代，不支持 HTML 和 Java 语言，就意味着应用程序的应用范围只能限于同质的环境。

随着 Java Servlet 的推出，Java 语言极大地推动了电子商务的发展。Java Server Page (JSP)技术的推出，更是让 Java 语言成为基于 Web 应用程序的首选开发工具。Internet 的普及和迅猛发展，以及 Web 技术的不断渗透，使得 Java 语言在现代社会的经济发展和科学

研究中,占据了越来越重要的地位。

1.2 Java 语言的特点

Java 语言是一种跨平台、适合于分布式计算环境的面向对象编程语言。它具有的特点很多,如简单性、面向对象、分布式、解释型、可靠性、安全性、平台无关性、可移植性、高性能、多线程、动态性等。下面介绍 Java 语言的几个重要特性。

1. 简单易学

Java 语言虽然衍生自 C++,与 C++相比 Java 是一个完全面向对象的编程语言。出于安全性和稳定性的考虑,Java 去掉了 C/C++支持的三个不易理解和掌握的数据类型:指针(pointer)、联合体(unions)和结构体(structs)。这样做的目的是用户不能通过 Java 程序直接访问内存地址,从而保证了程序更高的安全性。而 C/C++中联合体和结构体的功能,完全可以在 Java 中用类及类的属性等面向对象的方法来实现,这不但更加合理规范,而且降低了学习难度。

2. 面向对象

Java 语言最吸引人之处,就在于它是一种以对象为中心、以消息为驱动的面向对象的编程语言。面向对象的语言都支持三个概念:封装、继承和多态,Java 语言也是如此。

1) 封装

所谓封装,就是指利用抽象数据类型将数据和基于数据的操作封装在一起,数据被保护在抽象数据类型的内部,系统的其他部分只有通过封装在数据外面的被授权的操作,才能够与这个抽象数据类型交互。

2) 继承

继承是指一个对象直接使用另一个对象的属性和方法。Java 语言给用户提供了一系列的类,并且 Java 语言的类很有层次结构,子类可以继承父类的属性和方法。Java 语言只支持单一继承,这样就大大降低了复杂度,但在 Java 语言中,可以通过接口来实现多重继承。

3) 多态

多态是指一个程序中同名的多个不同方法共存的情况,即一个对外接口,多个内在实现方法。面向对象的程序中多态的情况有多种,可以通过子类对父类方法的覆盖实现多态,也可以利用重载在同一个类中定义多个同名的不同方法来实现多态。多态的特点使得它们不需了解对方的具体细节,就可以很好地共同工作。这个优点,对程序的设计、开发和维护都有很大的好处。

3. 平台无关性

Java 是与平台无关的语言,这是指使用 Java 语言编写的应用程序不用修改就可在不同的软硬件平台上运行。

平台无关有两种:源代码级和目标代码级。C 和 C++语言具有一定程度的源代码级平台无关,即用 C 和 C++语言编写的应用程序不用修改只需重新编译就可以在不同平台上运行。Java 语言是靠 Java 虚拟机(JVM)在目标代码级实现平台无关性的,可以说,JVM 是 Java 平台无关的基础。(关于 JVM 的使用,在 1.4 节介绍)

4. 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网络的不同主机上;

操作分布是指把一个计算分散在不同的主机上处理。Java语言支持WWW客户机/服务器计算模式，因此，它支持这两种分布性。对于数据分布，Java语言提供了一个称作URL的对象，利用这个对象，可以打开并访问URL地址上的对象，访问方式与访问本地文件系统相同。对于操作分布，Java的小程序(Applet)可以从服务器下载到客户端，将部分计算在客户端进行，提高系统执行效率。同时，Java语言提供了一整套网络类库，开发人员可以利用类库进行网络程序设计，方便地实现Java语言的分布式特性。

5. 可靠性

Java语言具有很高的可靠性。首先，Java语言是强类型的语言，要求显式的方法说明，这就保证了编译器可以发现方法的调用错误，保证了程序更加可靠；其次，Java语言不支持指针，这就避免了对内存的非法访问；第三，Java语言的自动单元回收功能防止了内存丢失等动态内存分配导致的问题；第四，Java解释器运行时实施检查，可以发现数组和字符串访问的越界；最后，Java语言提供了异常处理机制，可以把一组错误的代码放在一个地方，这样可以简化错误处理任务，便于恢复。

6. 安全性

Java是一种主要用于网络应用程序开发的语言，因此，对安全性要有较高的要求。如果没有安全保证，用户从网络上下载程序执行就会非常危险。

Java语言具有较高的安全性，它通过自己的安全机制防止了病毒程序的产生和下载程序对本地系统的威胁破坏。当Java字节码进入解释器时，首先必须经过字节码校验器的检查；其次，Java解释器将决定程序中类的内存布局；再次，类装载器负责把来自网络的类装载到单独的内存区域，避免应用程序之间相互干扰破坏；最后，客户端用户还可以限制从网络上装载的类只能访问某些文件系统。综合了上述几种机制，使得Java成为了安全的编程语言。

7. 支持多线程

线程是比进程更小的可并发执行的单位。C++语言没有内置的多线程机制，因此必须调用操作系统的多线程功能来进行多线程程序设计。而Java语言却提供了多线程支持。Java语言在两个方面支持多线程：一方面，Java环境本身就是多线程的，若干个系统线程运行，负责必要的无用单元回收、系统维护等系统级操作；另一方面，Java语言内置多线程机制，可以大大简化多线程应用程序开发。同时，Java语言的线程还包括一组同步原语，这些原语负责对线程实行并发控制。利用Java语言的多线程编程接口，开发人员可以方便地写出支持多线程的应用程序，提高程序执行效率。但需要注意的是，Java语言的多线程在一定程度上受到运行时支持平台的限制。

8. 支持网络编程

Java语言通过它所提供的类库可以处理TCP/IP协议，用户可以通过URL地址在网络上很方便地访问其他对象。Java的小程序(Applet)是动态、安全、跨平台的网络应用程序。Java的小程序嵌入在HTML文档中，通过主页发布到Internet。网络用户访问服务器的小程序时，这些小程序从网络上进行传输，然后在支持Java的浏览器中运行。

9. 编译与解释并存

用Java语言编写的程序称为源文件(扩展名为.java的文件)，源文件是不能被计算机执行的。要想使程序得以运行，必须利用编译器(不同的计算机语言有不同的编译器)对源

文件进行编译,编译器将源文件编译(即翻译)成计算机能懂的语言。Java 提供的编译器并不是把源文件编译成二进制码,而是将其编译成一种独立于机器平台的中间代码,这种中间代码被称为字节码(即扩展名为.class 的文件)。字节码可以被 Java 解释器所执行,由解释器将字节码再翻译成二进制码,使程序得以运行。字节码非常类似于机器指令,但字节码与具体机器是无关的,并不能在具体的平台上执行,而要通过 Java 运行系统中的解释器来解释执行。也就是说,Java 程序的运行要经过两个步骤来完成:首先是由编译器将 Java 源程序编译成字节码文件,然后再由 Java 运行系统解释执行字节码文件,这就是所谓的编译与解释并存。当然从本质上说,Java 语言属于解释型的高级程序设计语言,但 Java 语言通过字节码的方式,又在一定程度上解决了传统解释型语言执行效率低的问题,同时又保留了解释型语言可移植的特点。所以 Java 程序运行时比较高效。而且,由于字节码并不专对一种特定的机器,因此,Java 程序无须重新编译便可在多种不同的计算机上运行。

1.3 Java 技术简介

目前 Java 技术主要包括三个方面。

1. Java SE(Java Platform Standard Edition)

以前的版本称为 J2SE,是 Java 平台的标准版,是用于工作站、PC 的 Java 标准平台。它体现了 Sun 公司的开放精神,被称为是"互联网上的世界语"。

2. Java ME(Java Platform Micro Edition)

以前的版本称为 J2ME,是 Java 平台的精简版,是致力于消费产品和嵌入式设备的最佳解决方案。Java ME 是移动商务最佳的应用典范,不论是无线通讯、手机、PDA 等小型电子装置,均可采用 Java ME 作为开发工具及应用平台。它提供了 HTTP 等高级 Internet 协议,可以使移动电话能以 Client/Server 方式直接访问 Internet 的全部信息,不同的 Client 访问不同的文件,此外还能访问本地存储区,提供最高效率的无线交流。

3. Java EE(Java Platform Enterprise Edition)

以前的版本称为 J2EE,是 Java 平台的企业版,它是以企业为环境而开发应用程序的解决方案。它提供了企业 e-Business 架构及 Web Services 服务,其优越的跨平台能力与开放的标准,深受广大企业用户的喜爱。目前它已经成为开发商创建电子商务应用的事实标准。

1.4 Java 虚拟机

大部分的计算机语言程序都必须先经过编译(compile)或解释(interpret)的操作后,才能在计算机上运行,例如 C/C++等是属于编译型的语言,而 Basic 与 Lisp 等则是属于解释型的语言。然而,Java 程序(.java 文件)却比较特殊,它必须先经过编译的过程,然后再利用解释的方式来运行。通过编译器(compiler),Java 程序会被转成与平台无关(platform-independent)的机器码,Java 称之为"字节码"(byte-codes),字节码文件的扩展名为.class。通过 Java 的解释器(interpreter)便可解释并运行 Java 的字节码。图 1.1 说明了 Java 程序的执行过程。

字节码是 Java 虚拟机(Java Virtual Machine,JVM)的指令组,和 CPU 上的微指令码很相似。Java 语言编译成字节码后文件尺寸较小,便于网络传输。

图 1.1 Java 程序的运行过程：先编译，后解释

字节码最大的好处是可跨平台运行，即 Java 的字节码可以编写一次，到处运行。用户使用任何一种 Java 编译器将 Java 源程序(.java)编译成字节码文件(.class)后，无论使用哪种操作系统，都可以在含有 JVM 的平台上运行。这种跨越平台的特性，也是让 Java 语言急速普及的原因之一。

任何一种可以运行 Java 字节码的软件均可看成是 Java 的“虚拟机”(JVM)，如浏览器与 Java 的开发工具等皆可视为一部 JVM。很自然的，可以把 Java 的字节码看成是 JVM 上所运行的机器码(machine code)，即 JVM 中的解释器负责将字节码解释成本地的机器码。所以从底层上看，JVM 就是以 Java 字节码为指令组的“软 CPU”。也就是说，JVM 是可运行 Java 字节码的假想计算机。它的作用类似于 Windows 操作系统，只不过在 Windows 上运行的是.exe 文件，而在 JVM 上运行的是 Java 字节码文件，也就是扩展名为.class 的文件。JVM 其实就是一个字节码解释器。

1.5 Java 程序种类和结构

使用 Java 语言可以编写两种类型的程序：Application(应用程序)和 Applet(小程序)。这两种程序的开发原理是相同的，但是在运行环境和计算结构上却有着显著的不同。

应用程序是从命令行运行的程序，它可以在 Java 平台上独立运行，通常称之为 Java 应用程序。Java 应用程序是独立完整的程序，在命令行调用独立的解释器软件即可运行。另外，Java 应用程序的主类必须包含有一个定义为 public static void main(String[] args)的主方法，这个方法是 Java 应用程序的标志，同时也是 Java 应用程序执行的入口点，也就是说在应用程序中包含有 main()方法的类一定是主类，但主类并不一定要求是 public 类。

小程序是嵌入在 HTML(超文本标记语言)文档中的 Java 程序，需要搭配浏览器来运行，因此称为小程序。由此可见，当运行一个 Java 小程序时，同时还要为它编写一个 HTML 文件，然后在 WWW 浏览器中运行这个 HTML 文件，就可以激活浏览器中的 Java 解释器。另外，也可以调用一些能够模拟浏览器环境并执行 Java 小程序的软件来直接运行 Java 小程序。由于浏览器受安全控制的限制，所以 Java 小程序一般使用模拟浏览器环境的软件来执行。

Java 小程序与 Java 应用程序之间存在着很多不同之处，具体如下：

首先，小程序和应用程序之间的技术差别在于运行环境。Java 应用程序运行在最简单的环境中，它的唯一外部输入就是命令行参数；而小程序则需要来自 Web 浏览器的大量信息，它是内嵌在 HTML 文件里，在 WWW 浏览器这个特定环境下运行的，它需要知道何时启动，何时放入浏览器窗口，在何处、何时激活、关闭等。

其次，由于小程序和应用程序的执行环境不同，它们的最低要求也不同。在应用方面，WWW 使小程序的发布十分便利，因此小程序更适合在 Internet 上的使用；相反，非网络系

统和内存较小的系统更适合使用 Java 应用程序。

再次,Java 小程序可以直接利用浏览器或 AppletViewer 提供的图形用户界面,而 Java 应用程序则必须另外书写专用代码来营建自己的图形界面。

最后,小程序的主类(程序执行的入口点)必须是一个继承自系统类 JApplet 或 Applet 的子类,且该类必须是 public 类;而 Java 应用程序的主类,必须是包含有主方法 main() 的类。

小程序的编写方式与应用程序类似,因此只要熟悉了 Java 应用程序的编写方式,很快就能学会编写小程序。

一个复杂的程序可以由一个或多个 Java 源文件构成,每个文件中可以有多个类定义。下面的程序是一个一般的 Java 应用程序文件(关于小程序的结构将在第 16 章详细讨论)。

说明:为了便于对程序代码的解释,在每行之前加一个标号,它们并不是程序代码的一部分。

```
1  package ch01;                                    //定义该程序属于 ch01 包
2  import java.io.*;                                //导入 java.io 类库中的所有类
3  public class App1_1                              //定义类: App1_1
4  {
5    public static void main(String[] args)
6    {
7      char c = '';
8      System.out.print("请输入一个字符: ");
9      try{
10        c = (char)System.in.read();
11     }catch(IOException s){ }
12     System.out.println("您输入的字符是: " + c);
13   }
14 }
```

从这个程序可以看出,一般的 Java 源程序文件由以下三部分组成:

- package 语句(0 句或 1 句)。
- import 语句(0 句或多句)。
- 类定义(1 个或多个类定义)。

其中,package 语句表示本程序所属的包。它只能有一个或者没有。如果有,必须放在最前面。如果没有,表示本程序属于默认包。

import 语句表示引入其他类库中的类,以便使用。import 语句可以有 0 或多个,它必须放在类定义的前面。

类定义是 Java 源程序的主要部分,每个文件中可以定义若干个类。

Java 程序中定义类使用关键字 class,每个类的定义由类头定义和类体定义两部分组成。类体部分用来定义属性和方法这两种类的成员,其中方法类似于其他高级语言中的函数,而属性则类似于变量。类头部分除了声明类名之外,还可以说明类的继承特性,当一个类被定义为是另一个已经存在的类(称为父类)的子类时,它就可以从其父类中继承一些已定义好的类成员,而不必自己重复编码。

在类体中通常有两种组成成分：一种是域，包括变量、常量、对象数组等独立的实体；另一种是方法，类似于函数的代码单元块，这两种组成成分通称为类的成员。在上面的例子中，类 Appl_1 中只有一个类成员，即第 5 行定义的方法 main()。用来标志方法头的是方法名后面的一对小括号，小括号里面是该方法使用的形式参数，方法名前面的 public 用来说明这个方法属性的修饰符，其具体语法规定将在第 6 章中介绍。方法体部分由若干以分号";"结尾的语句组成，并由一对大括号{}括起来，在方法体内部不能再定义其他的方法。

同其他高级语言一样，语句是构成 Java 程序的基本单位之一。每一条 Java 语句都由分号";"结束，其构成应该符合 Java 语言的语法规则。类和方法中的所有语句应该用一对大括号{}括起来。除 package 及 import 语句之外，其他执行具体操作的语句，都只能存在于类的大括号之中。

比语句更小的语言单位是表达式、变量、常量和关键字等，Java 的语句就是由它们构成的。其中，声明变量与常量的关键字是 Java 语言语法规定的保留字，用户程序定义的常量和变量的取名不能与保留字相同。

Java 源程序的书写格式比较自由，如语句之间可以换行，也可以不换行，但养成一种良好的书写习惯比较重要。

注意：Java 是严格区分大小写的语言。书写时，大小写不能混淆。

一个程序中可以有多个类，但只能有一个类是主类。在 Java 应用程序中，这个主类是指包含 main()方法的类。在 Java 小程序中，这个主类是一个继承自系统类 JApplet 或 Applet 的子类。应用程序的主类不一定要求是 public 类，但小程序的主类一定要求是 public 类。主类是 Java 程序执行的入口点。

本章小结

1. Java 程序设计语言于 1995 年诞生，它是由美国加州的 Sun 计算机公司推出的，是一种能够跨平台使用的程序设计语言。

2. Java 分为标准版、企业版与精简版。Java 的标准版简称为 Java SE，企业版简称 Java EE，而精简版则简称为 Java ME。

3. Java 程序比较特殊，它必须先经过编译的过程，然后再利用解释的方式来执行。即首先要将源程序(.java 文件)通过编译器将其转换成与平台无关的字节码(.class 文件)，然后再通过解释器来解释执行字节码。

4. 字节码(byte-codes)最大的好处是可跨平台执行，可让程序"编写一次，到处运行(write once，run anywhere)"的梦想成真。

5. Java 程序可分为两种：一种是 Java application 称为 Java 应用程序；另一种是 Java applet 称为 Java 小程序。Java 应用程序是指可以在 Java 平台上独立运行的一种程序；而 Java 小程序则是内嵌在 html 文件里，需要在浏览器的支持下才能运行。

6. 无论是应用程序还是小程序都必须有一个主类，主类是程序执行的起始点，应用程序的主类是包含有 main()方法的类，但应用程序的主类并不一定要求是 public 类；小程序的主类必须是一个继承自系统类 JApplet 或 Applet 的子类，且该类必须是 public 类。