计算机教育核心课程教材

Java程序设计实用教程

主 编 刘甫迎 谢 春 徐 虹

科学出版社
www.sciencep.com

●计算机教育核心课程教材

Java 程序设计实用教程

主 编 刘甫迎 谢 春 徐 虹

科学出版社
北 京

内 容 简 介

Java 程序设计语言是由 Sun Microsystems 公司开发的一种面向对象的语言，它不仅适用于开发一般的商业程序，而且可以用于开发基于 Web 的互联网交互程序。Java 可以编译成跨平台、跨语言的代码。它去掉了 C 语言中的指针和多继承，简单易学且功能强。所以 Java 越来越多地得到人们的青睐。

本书包括 10 章和两个附录，详述了 Java 语言的概念、Java 的数据类型、运算符与表达式、结构化程序设计及算法、数组和结构、面向对象程序设计(类、继承、重载、事件)、可视化应用程序设计、Java 的文件操作、Applet、Java 的数据库操作等。本书内容翔实，突出了对实际编程能力的培养。

本书有实例、实验指导书、习题和教学大纲，便于学习与教学，可作为高等院校及软件学院的教材，也可作为软件开发人员的参考用书。

图书在版编目(CIP)数据

Java 程序设计实用教程/刘甫迎，谢春，徐虹主编. —北京：科学出版社，2005

(计算机教育核心课程教材)

ISBN 7-03-016212-9

Ⅰ. J… Ⅱ. ①刘… ②谢… ③徐… Ⅲ. JAVA 语言—程序设计—教材 Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2005)第 100018 号

责任编辑：余 丁 / 责任校对：包志虹
责任印制：安春生 / 封面设计：陈 敬

科学出版社 出版
北京东黄城根北街16号
邮政编码：100717
http://www.sciencep.com

丽源印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2005 年 9 月第 一 版 开本：787×1092 1/16
2005 年 9 月第一次印刷 印张：22 3/4
印数：1—4 000 字数：520 000

定价：28.00 元

(如有印装质量问题，我社负责调换〈新欣〉)

前　言

Java程序设计语言是由Sun Microsystems公司开发的一种面向对象的语言，它不仅适用于开发一般的商业程序，而且可以用于开发基于Web的互联网交互程序。近年来，Java程序设计语言之所以如此流行是因为它具有安全特性和平台无关性的优势，这意味着用Java程序设计语言编写的程序可以在任何一个平台上运行。Java可以编译成跨平台、跨语言的代码。它去掉了C语言中的指针和多继承，简单易学且功能强。它越来越多地得到人们的青睐。

目前，国内高校该类课程的教学内容一般比较偏重于编程语言理论的教学，大多采用传统的教学模式，结果导致学生的编程设计和应用能力不强。笔者认为学习此课程主要应掌握实践动手能力，分析问题、解决问题的能力。相关院校很有必要改变课程的教学内容和教学模式，用先进的教学理念和方法培养一流人才。本教材就是在此改革思路的指导下编写的，其特点是：

1. 把课程教学目标纳入该专业技能培养目标中，目录具体明确，学生学习兴趣大。例如对“计算机软件”等专业来说，Java程序设计课程是该专业的一门专业基础课，本书将本课程分解为结构化程序设计、面向对象程序设计、C/S模式编程、可视化（Visual）编程等能力模块进行教学。

2. 本书一开始便是用Java的面向对象的程序来展开的，便于学生们学习。

3. 选材上，讲述的理论“以必需够用为度”，减轻学生负担。另外，书中例子使用了许多经典算法，弥补了有些读者未学数据结构的不足。

4. 突出实践动手能力和实用性，突出案例（各章有例子以及综合实例等），配有实验指导书、习题、教学大纲等，便于学习与教学，力图使学生在学习本书后便能编写应用程序和系统。

本书可作为高等院校及软件学院的教材，也可作为从事软件开发和应用的人员的参考用书。本书由刘甫迎教授、谢春、徐虹老师主编。刘甫迎编写第1章、第2章、第5章、第6章和第10章；徐虹编写第7章以及第3、4、9章的实例；谢春编写第3章、第4章、第8章和第9章。本书由刘甫迎统稿。

由于作者水平有限，不妥之处在所难免，请读者批评指正。

目　录

第 1 章　使用 Java 编写第一个程序

本章主要介绍如何编写、修改和运行一个 Java 程序以及怎样使用数据。

1.1　创建一个程序

1.1.1　程序设计

一个计算机程序只不过是一组人为编写的用来告诉计算机做什么的指令集合。计算机是由很多小的开关组成的电路构建而成的，所以可以顺着下面的命令行来编写计算机程序：

第一个开关处于开状态
第二个开关处于关状态
第三个开关处于关状态
第四个开关处于开状态

程序可以为几千个开关像这样不断地写下去，这种模式的程序是用机器语言编写的，机器语言是最基本的电路语言。用这种语言编写程序的问题在于：在编写任何一个有价值的任务时需要明白开关当前的状态以及当程序不能按照预期效果运行的时候，如何去发现那些出错的开关。另外，不同的计算机开关的数量和位置是不同的，这就意味着需要为每种想在它上面运行程序的机器定制一种机器语言。

幸运的是，由于高级程序设计语言的发展，编程已经变得更容易了。高级程序设计语言提供了像“read”，“write”或者“add”等易于理解的术语，取代了完成这些任务所需的开关的一系列组合。高级程序设计语言还允许定义直观的计算机存储单元的名字，像“hoursWorked”，“rateOfPay”，而不是必须记住那些值的存储单元（开关号）。

每一种高级程序设计语言都有它自己的语法，或是语言的规则。例如：取决于特定的高级语言，可能会用动词“print”或“write”来产生输出。所有的语言都有一个特定的有限词汇表以及使用那个词汇表的一组特定的规则。程序员使用一个叫编译器的计算机程序把他用高级语言编写的语句翻译为机器代码。每当程序员错误使用编程语言时，编译器就会发出一个错误提示信息；接着，程序员纠正这个错误并且尝试再次编译该程序。学习像 Java，C++或 COBOL 这样的程序设计语言时，实际上是在学习那种语言的词汇表和语法规则。

除了要学习一种特殊语言的正确语法之外，程序员还必须理解计算机程序设计逻辑。任何一个程序背后的逻辑都包括了以正确的顺序执行各种各样的语句和方法来产生所期望得到的结果。比如：你可能能在一个乐器上弹奏优秀的乐章，但是如果你不能按正确的顺序来弹奏它们（当期望 F 高调时却弹成 B 降半音），那么你就不能弹出悦耳的

音乐。类似地，你可能能够正确地使用计算机语言的语法，但是程序的逻辑结构不正确。逻辑错误的例子包括了当你打算把两个值相除时却把它们相乘了，或者在获得适当的输入之前就产生了输出。

1.1.2 面向对象程序设计

编写计算机程序有两种常用的方法：面向过程的程序设计和面向对象的程序设计。

面向过程的程序设计包括用你的程序语言建立存放值的存储单元并且编写对那些值进行运算的一系列步骤或操作。计算机存储单元被称为变量，因为它们所保存的值可以变化。例如：某公司的工资程序中有变量 rateOfPay，这个变量存放的存储器单元在不同时间内可以有不同的值（公司中每个员工对应不同的值）。当执行工资程序时，对存储在 rateOfPay 中的值可对应多种操作，——比如，从输入设备中输入该值，与表示时间的变量相乘，在打印纸上输出。为方便起见，一个计算机程序的各个操作通常被组合成逻辑单元，称为过程。比如，把确定个人所得税得四到五步比较和计算可以合成一个过程，称为 caculateFederalWithholding。面向过程的程序定义了可变的存储单元，然后调用或引用一些过程对这些单元中的值进行输入、操作和输出。一个面向过程的程序通常包括成百上千的变量和过程调用。

面向对象程序设计是面向过程程序设计的一种扩展，在编写程序时采用的方法有一些不同。用面向对象的方法考虑问题首先把程序元素看成是与现实世界中的具体对象相似的对象，然后对这些对象进行操作以得到期望的结果。编写面向对象的程序包括创建对象和创建使用这些对象的应用程序。

如果你曾经使用过命令行操作系统（如 DOS）的计算机，也使用过图形用户界面的操作系统（如 Windows）的计算机，那么你已经了解了面向过程的程序和面向对象的程序区别。如果想将几个文件从软盘复制到硬盘，可以在提示符或命令行键入命令，或者是在一个图形环境下用鼠标来完成这个任务。两者的区别在于是否用一系列的命令按顺序移动三个文件，或者是拖动表示文件的图标从屏幕上一个位置移到另一个位置，这如同在办公室里将文件从一个档案柜拿到另一个档案柜一样。两者之中的任何一种操作系统都能移动相同的三个文件，但图形用户界面系统允许你如同对现实世界中的文件复制品那样对文件进行操作。换句话说，图形用户界面系统允许把文件当作对象处理。

在现实世界和面向对象程序设计中的对象都由状态和方法组成。一个对象的状态也称为属性。比如，汽车有一些属性是制造单位、型号、生产日期和购买价格，另一些属性包括汽车目前是否在行驶、车速和是否干净。所有汽车都有相同的属性，当然这些属性值不会相同。类似的，你的狗有品种、名字、年龄、杂色是否流行等属性。你的红色 Chevrolet 汽车是所有汽车类中的一个实例。你的名叫 Goldie 的金黄色的狗是所有狗类中的一个实例。把某些项目看成是类的实例允许你把类的常识运用到类的个体成员中，这些对象的特定实例继承了一般化类的属性。如果你的朋友购买了一辆汽车，你知道了它的型号，如果你的朋友喂了一只狗，你知道了狗的品种。你可能不知道你朋友汽车的速度或他的狗的颜色的精确内容或当前的状态，但你确实知道这一类汽车和这类狗的属性。同样，在图形用户操作环境中，你希望每一个组件都有特殊

的、一致的属性，如相同的菜单条和标题栏，因为每一个组件都继承了图形用户界面组件通用类的成员属性。

注意：按规定，程序员使用 Java 程序语言定义类名以大写字母开头。因此定义一个汽车的属性和方法的类就可能被命名为 Automobile，并且狗类可能就被命名为 Dog。然而，生成一个可行的程序并不要求遵循这个规定。

除属性外，对象能使用方法来完成任务。例如：汽车能够向前行驶和向后倒车、能装满汽油或被清洗，这些都是改变它们某些属性的方法。方法是为了弄清某些属性，比如汽车目前的速度和汽油缸当前的状态。类似地，一只狗能走或者跑，吃东西和洗澡，并且也有判断这只狗有多饿的一些方法。图形用户界面操作系统组件能被最大化、最小化和拖动。与面向过程的程序一样，面向对象程序有变量（属性）和过程（方法），但是这些属性和方法是被封装在对象中的。封装是一种技术，把对象属性整合成一个互相依赖的单元，从而作为一个不可分割的整体使用。程序员有时把封装看成是一个“黑盒”，或者可以使用的设备，而不用考虑其内部的装置。

如果一个对象的方法编写好了，用户可以不知道方法是如何执行的这些底层的细节。在这种情况下，用户必须简单理解方法与对象间的接口或交互作用。例如：如果你能把汽车的汽油加满，是因为你知道汽油泵塞和汽车缸之间的接口如何打开，而不必了解泵如何工作或者汽油缸实际装在汽车的哪个部位。如果你能读取速度表，而显示的数字是如何计算的就不重要了。事实上，如果生产了更先进、精确的速度测定设备，并安装到你的汽车中，只要它与原来的设备接口相同，你不必了解或关心它如何操作。同样的道理可应用到面向对象程序设计中的创建对象上。

1.1.3　Java 程序设计语言

Java 程序设计语言是由 Sun Microsystems 公司开发的一种面向对象的语言，它不仅用于开发一般的商业程序，而且可以用于开发基于万维网的互联网交互程序。近年来，Java 程序设计语言之所以如此流行是因为它的以下优势：安全特性和平台无关性，这意味着用 Java 程序设计语言编写的程序可以在任何一个平台上运行。运行 Java 程序的机器只需要有一个称为解释器的特殊程序来为主机翻译程序。相比之下，当使用其他的程序设计语言时，软件的开发商通常要生产同一产品的多种版本（DOS、Windows3.1、Windows95 和 Windows98 等版本），这样所有的用户才能使用这个程序。有了 Java 程序语言，一个程序版本可以在所有这些平台上运行。用 Java 编写的程序被编译成 Java 虚拟机代码，称为字节代码。编译后的字节代码最后在执行程序的机器上被解释。任何编译后的程序可以在带有 Java 语言解释器的任何机器上运行。

注意：为了简化，在本书中“Java 程序”和“Java 程序设计语言编程”这两个术语可交换使用。

Java 程序设计语言的另一大优点是它比许多其他的面向对象程序语言使用起来更简单。Java 程序设计语言模仿了 C++程序设计语言。尽管两种语言在首次推出时都认为不容易于阅读或理解，但 Java 语言去除了一些在 C++理解上最难的特性。

注意：对于 C++程序员来说，指针和多重继承是两个最难掌握的特性。Java 程序

设计语言除去了 C++语言中一些令人感到麻烦的特性。

1.1.4 开始写程序

乍一看，即使是最简单的 Java 程序都包括了一定数量的易混淆的语法。就拿下面这个简单的程序来说。这个程序只有 7 行，并且它的任务只是在屏幕上显示“First Java program”。

```
public class First
{
   public static void main(String[]args)
   {
    System. out. println("First Java program");
   }
}
```

在这个程序中实际起作用的语句是 System. out. println（" First Java program"）；所有的 Java 程序语句都以分号结束。

文本“First Java program”是一个照原样输出字符串。也就是说，它输出的是与输入完全相同的一组字符。在 Java 中任何照原样输出的字符串都输出的是双引号之间的字符。

，“First Java program”这个字符串出现在括号中是因为这个字符串是传给一个方法的参数，并且传给方法的参数都是出现在括号中。参数是由一个方法要完成它的任务所需的信息组成的。例如：你可能与一个出售体育用品的公司签了一个订单。处理一个订单就是由一组标准过程组成的一个方法。然而，每一个订单都要求如下的信息——例如：你要订购的产品的编号和数量——这些信息就可以作为订单的参数。如果你订购了两个编号为 5432 的产品，你肯定期望得到与你订购 1000 个编号为 9008 的产品不同的结果。同样地，如果输入参数“Happy Holidays”到一个过程，肯定期望得到与输入" First Java program"不同的结果。

在 System. out. println（" first Java program"）；这个语句中，" first Java program" 所传给的方法被称为 println（）。Println（）这个方法主要是在屏幕上打印一行输出，让光标显示在下一行上，然后为其他的输出做准备。

注意：方法的名字后面常常跟有括号，就像在 println（）中，所以可以以此来区别变量名和方法名。

在 System. out. println（" First Java program"）；这个语句中，out 是一个对象。out 这个对象代表屏幕。包括打印等几个方法对 out 对象来说都是可利用的。当然，并不是所有的对象都有一个 println（）方法（例如：你不能在一个键盘或你的汽车或你的狗上打印），但是 Java 平台的创建者认为你会频繁地想在屏幕上显示输出。因此 out 这个对象被创建并被赋予了名为 println（）的方法。在这部分中，将创建你自己的对象并为它们赋予方法。

注意：print（）方法与 println（）方法非常相似。使用 println（），信息打印后，

光标将换行显示。使用 print ()，光标不会换行，它仍然停留在与输出相同的一行。

在 System. out. println (" First Java program")；这个语句中，System 是一个类。因此该类定义了一组相似的 System 对象的属性，就像 Dog 类定义 dog 对象的属性一样。System 的一个对象就是 out（你可能可以猜到另一个对象是 in，并且它代表的是一个输入设备。）

注意：Java 程序语言是非常敏感的——名叫 System 的类是与名叫 system，SYSTEM 或者 sYsTeM 完全不同的类。

在 System. out. println (" First Java program")；这个语句中的句点是用于分隔类名、对象名以及方法名的。可以在 Java 程序中重复使用这种类、对象、方法的格式。

例 1.1　打印字符串“First Java program”这个语句被嵌入下列程序中。

```
public class First
{
 public static void main(String[]args)
 {
   System. out. println("First Java program");
 }
}
```

在 Java 程序中使用的一切都是类的一部分。书写 public class First 时，定义了一个名为 First 的类。可以用任何一个你所需要的名字或标识符定义一个 Java 类，只要它满足下列要求：

① 一个类名必须以字母表中的字母（它包括任何一个非英文字符，例如：α 或 π），下划线或美元符号开头。

② 类名中只能含有字母，数字，下划线或美元符号。

③ 类名中不能与 Java 语言中保留的关键字相同，例如：public 或 class（保留的关键字列在表 1.1 中。）

④ 类名中不能是任何一个下列值：true，false 或 null。

注意：Java 程序语言是基于统一的字符集编码标准，它是字符表示的国际标准。Letter 这个术语表示英语字符，阿拉伯字符，希腊字符以及其他字母表。本章的 B 部分将提供更多的关于 Unicode（统一的字符集编码标准）的信息。

表 1.1　Java 程序语言保留关键字

abstract	float	private
boolean	for	protected
break	future	public
byte	generic	rest
byvalue	goto	return
case	if	short
cast	implements	static
catch	import	super
char	inner	switch

续表

class	instanceof	synchronized
const	int	this
continue	interface	throw
default	long	throws
do	native	transient
double	new	try
else	null	var
extends	operator	void
final	outer	volatile
finally	package	while

Java 程序语言中类名一般用大写字母开头，并且为了提高可阅读性而根据需要使用一些其他的大写字母。表 1.2 中列出了一些 Java 程序有效的和常用的类名。

表 1.2 在 Java 程序设计语言中的一些有效的类名

类 名	描 述
Employee	用一个大写字母开头
UnderGradStudent	用一个大写字母开头，不含空格，用大写字母强调每个新词的首字母
InventoryItem	用一个大写字母开头，不含空格，用大写字母强调第二个词的首字母
Budget2001	用一个大写字母开头，不含空格

表 1.3 列出了一些有效的类名，但是不常用。

表 1.3 Java 程序设计语言中的一些不规则的类名

类 名	描 述
employee	用一个小写字母开头
Undergradstudent	新词的首字母没有用大写字母表示，阅读起来很困难
Inventory _ Item	下划线不是常用在指示新词上
BUDGET2001	所有都是大写字母

注意：你应该遵循为 Java 程序设计语言已经建立的规则，以便你的程序能很容易被其他程序员看懂。本书使用的就是已经建立的 Java 程序规则。

表 1.4 列出了 Java 程序设计语言中不合法的类名。

表 1.4 Java 程序设计语言中不合法的类名

类 名	描 述
an employee	空格字符是非法的
Inventory Item	空格字符是非法的
class	class 是保留字
2001Budget	类名不能以数字开头
phone#	# 号是不允许在类名中出现的

在例 1.1 中，public class First 这行包含了关键字 class，标识 First 为一个类。保留字 public 是一个访问权限。一个访问权限定义了一个类能被访问的情形。public 是最自由的访问形式。将在第二章中学习 public 和其他访问方式。

将所有类的内容放入花括号 {} 中。一个类能够包含任意数量的数据成员以及方

法。在例 1.1 中，First 类在 {} 中仅仅包含一个方法。这个方法名是 main ()，并且 main () 方法包含它自己的括弧和 println () 打印语句。

注意：通常，在 Java 程序中空格是任选的。空格是空格键，制表符和回车键的任意组合。然而在标识符或者关键字中不能使用空格。在字与字，行与行之间可以通过输入空格键、制表符或回车键来插入空白，因为编译器将忽略这些额外的空格，可以在程序代码中加入一些空格，使它更容易阅读。

在 Java 程序中，对每一个花括号的开始符“{”，都必须有一个花括号的结束符“}”与之对应。花括号的开始符“{”和花括号的结束符“}”的位置对编译器来说是不重要的。例如：下面的这种方式与图 1-1 的执行结果是一样的。仅有的不同就是方法组织的不同。通常，代码放在成对地垂直排列的“{”和“}”中更易于阅读。应该仔细输入代码，以便它容易阅读。

```
public static void main(String[]args)
  {System. out. println("First Java program");}
```

main () 方法是相当复杂的。当你学完这本书的时候，就会很清楚在这个方法中使用的每一个术语的含义和目的；现在只是做个简要的介绍。

在 public static void main (String [] args) 中，public 是一个访问权限，就像定义 First 类一样，在英语中，单词 static 的意思是几乎不变的，或者是固定的。在 Java 程序设计语言中，保留关键字 static 也是不变的意思，并且指出为 First 类创造的每一个成员都有一个统一的不变的。在 Java 程序设计语言中，static 还隐含了唯一性。对于 First 类只有一个 main () 方法将被存储在计算机的存储器中。当然，其他类最终也会有它们自己的不同的 main () 方法。

在英语中，void 的意思是空的。当关键字 void 用于 main () 的方法头中时，它并不意味着这个 main () 方法是空的，而是，当 main () 方法被调用时，它不会返回任何值。这也不意味着 main () 方法不产生输出——事实上，它是要产生输出的。main ()方法不会返回任何值到其他任何使用它的方法中。在第二章中，将学到更多的关于返回值的知识。

所有的 Java 应用程序都必须包括一个名为 main () 的方法，并且大多数的 Java 应用程序都还有一些另外的方法。执行一个 Java 应用程序时，编译器总是先执行main () 方法。

在 public static void main (String [] args) 中，可能已经注意到在括弧之间的那些内容，(String [] args) 必须作为参数传到 main () 方法，正如字符串“First Java program”是一个传到 println () 方法的参数。String 代表一个 Java 类，它能用于代表字符串。标识符 args 被用来保存被传到 main () 方法的任意字符串。main () 方法可以用那些参数做一些事情，例如：打印它们，但是在例 1.1 中的 main () 方法实际上没有用 args 标识符。然而，必须在 main () 方法中放入一个标识符。这个标识符不需要被命名为 args——它可以是任意合法的 Java 标识符——但是通常使用 args。

注意：在 main () 的方法头中引用字符串类时，方括弧指示的是字符串对象的一个数组。在第四章中将学到更多的关于数组和字符串类的知识。

在例 1.1 中出现的那个简单的程序有许多需要记住的东西。然而，现在，可以把例 1.2 的这个程序作为一个框架来使用，可以用任意想执行的语句来替换/ * * * * * /行。

例 1.2 输出程序框架。

```
public class First
{
 public static void main(String[]args)
{
  /*****/
}
}
```

既然已经了解了 Java 程序语言的基本框架，就可以在一个文本编辑器中输入你的第一个 Java 程序了。程序员的传统是：用任何语言写第一个程序时都是产生一个“Hello，World!”作为输出。创造这样一个程序可以用任何文本编辑器，比如 Notepad，WordPad，或者其他任何的字处理程序。

书写第一个 Java 程序：

① 启动任何一个文本编辑器（比如 Notepad，WordPad 或者其他任何的字处理程序），如果需要的话，打开一个新文件（Notepad 用来书写程序是最简单的）。

② 输入类的文件头 public class Hello。在这个例子中，类名是 Hello，可以为这个类使用任何你想使用的有效的名字。如果选择 Hello，就必须用 Hello 来引用这个类，而不是 hello，因为 Java 程序语言是要区分大小写的。

③ 按一次回车键，输入｛，再按一次回车键，输入｝。就可以在花括号中添加 main（）方法了。虽然没有被要求，但是最好把每一个花括号单独列一行。这样会使你的代码更易于阅读。

④ 如例 1.3 所示，在花括号中加入 main（）函数文件头，然后为 main（）方法输入一对花括弧。

例 1.3 Hello 类的 main（）函数框架。

```
Public class Hello
{
  public static void main(String[]args)
  {


  }
}
```

然后在 main（）函数括号中加入语句产生输出“Hello，World!”

⑤ 如例 1.4，添加一个 println（）语句到 main（）方法中。

例 1.4 为 Hello 类完成 main（）方法。

```
public class Hello
{
```