

教育部 高职高专 规划教材
高职高专 计算机系列 规划教材



Java语言程序设计 (第3版)

陆 迟 编著

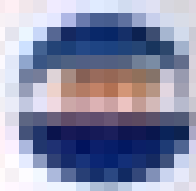


电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>

教育部高等学校计算机专业类
教学指导委员会推荐教材



Java 语言程序设计

(第3版)

邵维刚 编

清华大学出版社

高职高专计算机系列规划教材

Java 语言程序设计

(第3版)

陆 迟 编著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内容简介

本书以 Java SDK 最新版 6.0 (亦称为 1.6.0 版) 为基础, 从面向对象的基本概念和 Java 应用程序及 Java 小程序结构、示例入手, 由浅入深、循序渐进地讲述 Java 语言的主要特征、基础知识及其面向对象的特性和编程方法, 介绍了 Java 的流处理、多线程编程、GUI 编程和多媒体编程等 Java 的实用技术。

全书内容丰富、选材适当、结构合理、论述清楚, 注重实用性。所有例题均用 Java 软件开发工具 JDK 6.0 调试通过。本书既可用做普通高等院校应用型本科、高职高专、成人高等院校计算机及相关专业的教材, 又可作为计算机技术爱好者自学 Java 语言的参考书, 对于需要了解 Java 基本知识的人员, 也可从本书获得所要的参考信息。

未经许可, 不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有, 侵权必究。

图书在版编目 (CIP) 数据

Java 语言程序设计/陆迟编著. —3 版. —北京: 电子工业出版社, 2010.12

高职高专计算机系列规划教材

ISBN 978-7-121-11959-0

I. ①J… II. ①陆… III. ①JAVA 语言—程序设计—高等学校: 技术学校—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字 (2010) 第 194739 号

策划编辑: 吕 迈

责任编辑: 吕 迈

印 刷: 北京市天竺颖华印刷厂

装 订: 三河市鑫金马印装有限公司

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×1092 1/16 印张: 17 字数: 435 千字

印 次: 2010 年 12 月第 1 次印刷

印 数: 4 000 册 定价: 28.50 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zltz@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

前 言

Java 作为在 Internet 上最流行的编程语言，其发展非常迅速。从 1995 年诞生以来，经过短短十多年的发展，如今，它已不仅仅是一门语言，而已发展为一门技术，包括 Java 的芯片技术、Java 的编译技术、Java 的数据库连接技术、基于 Java 的信息家电的联网技术、企业信息服务的综合求解方案技术，等等。

Java 语言作为一种优秀的面向对象的程序设计语言，具有平台无关性、安全机制、高可靠性和内嵌的网络支持等特点。由于 Java 语言的这些特点，使得 Java 语言成为当前网络应用程序编写的首选工具之一。还有人预言，不久的将来，全世界 90% 的程序代码将用 Java 语言重写和改写。Java 语言和技术的大量使用也促使 Java 语言本身不断发展。截止到现在，Java 语言的发明者美国 Sun 公司推出了 Java 开发工具（JDK）的最新版本是第 6 版，供使用者免费下载使用。本书也应用了 JDK 6 的部分新功能。

本书是面向应用型本科学生、高职专科学生及其他对 Java 语言和面向对象程序编程技术感兴趣的读者的。书中对内容的编排、取舍和例题、习题的选择，保证了一定的新颖性和深度、广度，在编写方法上注意遵循由浅入深、循序渐进、难点分散的原则。学习本书前应对计算机有一定的认识，最好会 DOS、Windows 等系统的基础操作。可以将本书作为第一门程序设计语言来学习，获得编写程序的初步经验。对有其他高级语言编程经验的读者，学习本书也会感到得心应手，从中领略 Java 语言面向对象、易学易用的特点。

本书第 1 章概要介绍面向对象的一些基本概念，对两种 Java 程序进行了介绍，并简述了 Java 程序的结构和开发过程。第 2 章和第 3 章介绍 Java 语言编程的基础知识，包括了一般程序设计语言的大部分内容，有基本数据类型、常量、变量、运算符、表达式和流程控制语句、方法（函数）等，还介绍了 Java 语言的异常处理，初步涉及面向对象的程序设计技术。第 4 章介绍 Java 语言的一维数组、二维数组定义和使用。第 5 章介绍面向对象设计的主要内容，包括类、继承、接口和包等。第 6 章介绍 Java 语言的字符串，包括大量的字符串处理方法和 main() 方法参数的使用等。第 7 章介绍 Java 语言的流处理，包括面向字节和面向字符的两种流。第 8 章介绍 Java Applet，同时也介绍了相关的 HTML 和图形界面的 AWT 绘图等内容。第 9 章介绍 Java GUI 程序设计，包括布局、观感、Swing 组件和事件处理等。第 10 章介绍 Java 的多线程程序设计，包括线程的概念、多线程的实现方法、线程的生命周期、线程的控制、线程的异步和同步等。第 11 章介绍 Java 语言的多媒体技术，包括在 Applet 和 Application 中显示图像、制作动画和播放声音等。第 12 章介绍 Java 数据库连接（JDBC）技术的知识，简述了数据库连接的两种方法，并给出了程序实例，对初次接触用 Java 来进行数据库应用程序开发的读者，可起到一个引导入门的作用。第 13 章介绍一个简单实用的 Java 应用程序的例子：Java 支持的图像文件的演示和声音文件的播放的 Java 应用程序。它应用了 GUI 的程序设计技术，包括多种 Swing 组件、事件处理、文件打开对话框、信息对话框等，并介绍了如何用 jar 工具软件来组织和对应用程序的类和资源文件压缩打包并执行的方法。第 14 章介绍本课程的实验内容与安排。共安排了八个实验，供需要者

选用。

计算机语言的学习应是课堂学习和上机实验的有机结合，特别要重视上机实验的环节。优秀的软件开发人员都有大量上机编程的经验，从实践中可学到很多书本上没有的东西。读者除在学校安排固定实验时间之外，还要利用更多的其他实验机会。只要注重实验，边学边练，Java 语言程序设计入门是不难的。

本书的出版与所有帮助作者完成本书的领导、同事和家人的支持是分不开的，在此表示衷心感谢。特别要感谢浙江大学陈海燕老师审阅了全书，出版社编辑和江汉大学有关老师大力的支持和帮助。虽然作者在编写本书时尽了最大努力，但因时间紧迫，书中仍难免疏漏错误，欢迎各位读者、专家批评指正。

作 者
2010 年 9 月

目 录

第 1 章 Java 语言概述	1
1.1 Java 语言发展和特点	1
1.1.1 Java 语言的发展	1
1.1.2 Java 语言的特点	1
1.1.3 Java 程序的工作机制	2
1.2 面向对象程序设计	2
1.2.1 传统与面向对象程序设计语言	3
1.2.2 对象的性质	4
1.3 Java 程序举例	5
1.3.1 Java 应用程序举例	5
1.3.2 Java 小程序举例	6
1.3.3 Java 程序结构	7
1.4 Java 程序开发工具与开发方法	7
1.4.1 JDK	7
1.4.2 JCreator Pro 简介	10
1.5 习题	12
第 2 章 Java 语言基础	13
2.1 标识符、保留字和分隔符	13
2.1.1 标识符	13
2.1.2 保留字	13
2.1.3 分隔符	14
2.2 数据类型	14
2.2.1 数据类型概述	14
2.2.2 常量与变量	15
2.3 基本类型	15
2.3.1 整型数据	15
2.3.2 实型（浮点型）数据	16
2.3.3 字符型数据	17
2.3.4 布尔型数据	18
2.3.5 字符串数据	18
2.3.6 默认初始值	19
2.4 运算符	20
2.4.1 算术运算符	20
2.4.2 关系运算符	22
2.4.3 布尔运算符	22

2.4.4	位运算符	24
2.4.5	赋值运算符	25
2.4.6	条件运算符	26
2.4.7	字符串运算符	26
2.5	常用 Java 数值计算方法	27
2.6	表达式	28
2.6.1	类型转换	28
2.6.2	优先级	30
2.6.3	结合性	31
2.7	包装类	31
2.8	输入/输出初步	32
2.8.1	标准输入/输出	32
2.8.2	对话框输入/输出	35
2.9	习题	36
第 3 章	流程控制	38
3.1	语句和块	38
3.1.1	语句	38
3.1.2	块	39
3.2	分支语句	39
3.2.1	if - else 语句	39
3.2.2	switch 语句	42
3.3	循环语句	43
3.3.1	while 语句	44
3.3.2	do - while 语句	44
3.3.3	for 语句	45
3.3.4	循环嵌套	47
3.4	标号和其他流程控制语句	48
3.4.1	标号	48
3.4.2	break 语句	49
3.4.3	continue 语句	50
3.4.4	return 语句	51
3.5	方法的使用	51
3.5.1	方法的定义与调用	52
3.5.2	方法调用中的数据传送	53
3.5.3	方法和变量的作用域	54
3.5.4	方法的嵌套和递归调用	57
3.5.5	方法的重载	58
3.6	异常处理	59
3.6.1	异常概述	59
3.6.2	异常处理	61

3.7	习题	67
第 4 章	数组	71
4.1	一维数组	71
4.1.1	一维数组的定义	71
4.1.2	一维数组的引用	73
4.2	多维数组	76
4.2.1	二维数组的定义	76
4.2.2	二维数组的引用	78
4.3	数组作为方法参数和返回值	79
4.4	数组操作的常用方法	80
4.5	数组应用举例	82
4.6	习题	84
第 5 章	面向对象程序设计	87
5.1	类 (class)	87
5.1.1	类定义	87
5.1.2	类对象	88
5.1.3	构造方法	90
5.1.4	类和成员的修饰符	92
5.1.5	类的使用	94
5.2	类的继承 (inheritance)	97
5.2.1	类继承的实现	98
5.2.2	抽象类和抽象方法	100
5.2.3	类对象之间的类型转换	101
5.3	接口 (interface)	101
5.3.1	接口的定义和实现	102
5.3.2	接口的继承和组合	103
5.3.3	接口的多态	104
5.3.4	接口类型的使用	105
5.4	包 (package)	105
5.4.1	创建包	105
5.4.2	使用包	106
5.4.3	类及类成员的访问权限	107
5.4.4	Java 的应用程序接口 (API)	108
5.5	习题	109
第 6 章	字符串处理	111
6.1	String 类字符串	111
6.1.1	String 类字符串的定义	111
6.1.2	String 类的常用方法	112
6.2	StringBuffer 类字符串	117
6.2.1	StringBuffer 类字符串的定义	117

6.2.2	StringBuffer 类的常用方法	118
6.3	main 方法的参数	121
6.4	字符串应用举例	123
6.5	习题	124
第 7 章	输入/输出处理	126
7.1	输入/输出概述	126
7.1.1	输入/输出流概念	126
7.1.2	输入/输出类	126
7.1.3	标准输入/输出	128
7.2	文件的顺序访问	129
7.2.1	字节流 (InputStream 类和 OutputStream 类)	130
7.2.2	字符流 (Reader 类和 Writer 类)	136
7.2.3	应用 Scanner 类读取顺序文本文件	139
7.3	文件的随机访问	140
7.4	目录和文件管理	142
7.4.1	目录管理	143
7.4.2	文件管理	143
*7.5	其他常用流处理	145
7.5.1	管道流	145
7.5.2	内存的访问	145
7.5.3	顺序流	146
7.6	习题	147
第 8 章	Java 小程序	148
8.1	Java 小程序概述	148
8.1.1	Java 小程序的特点	148
8.1.2	HTML 语言	148
8.1.3	在 HTML 文件中嵌入小程序	149
8.2	Java 小程序的创建和执行	151
8.2.1	Applet 的类层次及框架结构	151
8.2.2	Applet 的执行	152
8.2.3	Applet 的主要方法及生命周期	152
8.2.4	小程序和应用程序	154
8.2.5	HTML 参数传送	155
8.3	Java 小程序的 AWT 绘制	157
8.3.1	AWT 绘制基础	157
8.3.2	应用字体	158
8.3.3	应用颜色	159
8.3.4	绘制图形	161
8.4	小程序的通信	163
8.4.1	同页小程序间的通信	163

8.4.2	小程序与浏览器之间的通信	164
8.5	Java 小程序的应用	165
8.5.1	访问 WWW 资源	165
8.5.2	访问网络资源	166
8.6	习题	167
第 9 章	图形用户界面 (GUI) 设计	169
9.1	图形用户界面 (GUI) 设计概述	169
9.1.1	GUI 支持包和简单 GUI 程序例	169
9.1.2	容器、组件、布局和观感	172
9.1.3	事件处理	174
9.2	布局管理器	176
9.3	常用 Swing 组件	180
9.3.1	容器组件	180
9.3.2	按钮 (JButton)	183
9.3.3	标签 (JLabel)	186
9.3.4	复选框 (JCheckBox)	187
9.3.5	单选按钮 (JRadioButton)	188
9.3.6	文本框	190
9.3.7	列表框 (JList)	194
9.3.8	组合框 (JComboBox)	196
9.3.9	滚动条 (JSlider)	197
9.3.10	菜单	199
9.3.11	对话框	201
9.4	鼠标和键盘事件	205
9.4.1	鼠标事件	205
9.4.2	键盘事件	206
9.5	习题	208
第 10 章	Java 的多线程	209
10.1	多线程概述	209
10.1.1	多线程的概念	209
10.1.2	Java 对多线程的支持	210
10.1.3	线程的状态和生命周期	211
10.1.4	线程的调度和优先级	212
10.2	多线程的实现方法与控制	212
10.2.1	多线程的实现方法	212
10.2.2	多线程的控制	217
10.3	多线程的互斥	217
10.4	多线程的同步	218
10.5	多线程的应用	221
10.6	习题	222

第 11 章	Java 多媒体技术	224
11.1	图像	224
11.2	动画	227
11.3	声音	232
11.4	习题	235
第 12 章	JDBC 数据库应用程序设计	236
12.1	JDBC 概述	236
12.2	JDBC 数据库操作实例	243
12.3	使用 JDK 内置的 Java DB	250
12.4	习题	252
第 13 章	Java 应用实例	254
13.1	程序设计	254
13.2	程序组装与打包	257
第 14 章	Java 实验指导	258
实验一	Java 语言环境和简单程序设计	258
实验二	流程控制语句和数组程序设计	259
实验三	类和对象程序设计	259
实验四	输入/输出程序设计	260
实验五	Java Applet 设计	260
实验六	图形用户界面设计	260
实验七	Java 线程和多媒体程序设计	261
实验八	JDBC 连接数据库程序设计	261

第 1 章 Java 语言概述

1.1 Java 语言发展和特点

1.1.1 Java 语言的发展

1990 年, 美国 Sun 公司的 James Gosling、Bill Joe 等人, 为在电视、控制烤箱等家用消费类电子产品上进行交互式操作而开发了一种与平台无关、可靠性强、小而灵活的编程语言, 但当时并没有引起人们的注意。直到 1994 年下半年, Internet 的迅猛发展, 环球信息网 WWW 的快速增长, 人们发现 Java 这种中性平台及可靠性强的语言恰恰就是全球信息网在等待的语言。Java 的开发人员基于网络对 Java 进行了一系列的改进, 融合了 C 和 C++ 等语言的优点, 形成了现在这套与众不同的面向对象的通用程序设计语言。

Java 的原名叫 Oak (橡树), 但在申请注册商标时, 发现 Oak 已经有人用了。在想了一系列名字后, 最终, 使用了提议者在喝一杯 Java 咖啡时无意提到的 Java。

Java (JDK 1.0) 正式发表于 1995 年 5 月。Java 的“Write Once, Run Anywhere (一次编写, 到处运行)”口号使得 Java 一出现就引起广泛的注意, 用 Java 开发的软件可以不用修改或重新编译而直接应用于多种不同平台的计算机上。Java 语言的众多优点使得它逐渐成为 Internet 上受欢迎的开发与编程语言。Java 的诞生对传统的计算模型提出了新的挑战。

Java 语言有着广泛的应用前景, 可以从以下几个方面来考虑其应用。

- (1) 所有面向对象的应用开发, 包括面向对象的事件描述、处理、综合等;
- (2) 计算过程的可视化、可操作化的软件的开发;
- (3) 动态画面的设计, 包括图形图像的调用;
- (4) 交互操作的设计 (选择交互、定向交互、控制流程等);
- (5) Internet 的系统管理功能模块的设计, 包括 Web 页面的动态设计、管理和交互操作设计等;
- (6) Intranet (企业内部网) 上的软件开发 (直接面向企业内部用户的软件);
- (7) 与各类数据库进行连接并进行各种操作的实现;
- (8) 其他应用类型的开发设计。

目前, 针对不同领域的软件开发, Java 有三个平台可供选择:

Java SE: Java 标准版, 可用于开发桌面应用和一般的网络应用。

Java EE: Java 企业版, 可用于开发一般和复杂的分布式的综合网络应用。

Java ME: Java 微型版。可用于开发可用资源较少的个人数字助理、手机等终端设备上的应用。

1.1.2 Java 语言的特点

Java 是一个迅速发展的网络编程语言, 它是一种新的计算概念。

首先，作为一种程序设计语言，它简单、面向对象、分布式、解释执行、不依赖于机器的结构、具有可移植性、鲁棒性、安全性、并且是多线程的、动态的，具有很高的性能。

其次，Java 最大限度地利用了网络。一种嵌入 HTML 语言中，通过 Web 页发布到 Internet，称为 Java 小程序 (Applet) 的 Java 程序，可在网络上运行而不受 CPU 和环境的限制。一旦载入小程序，就可以放心地来生成多媒体的用户界面或完成复杂的计算而不必担心病毒的入侵。小程序不仅可以和图像、声音、动画等一样从网络上下载，它还可以进行人机交互，动态地改变图像、动画的显示和声音的播放。

另外，Java 还提供了丰富的类库，以满足网络化、多线程、面向对象系统的需要，使程序设计者可以很方便地建立自己的系统。

1.1.3 Java 程序的工作机制

学习 Java 语言，有必要了解 Java 的工作机制，这将更有助于理解 Java 语言的特点。

对于运行在 Internet 上的网络应用程序，需要有良好的可移植性。因为 Internet 是由各种各样不同类型的终端、服务器和 PC 等硬件设备组成的，而且在这些设备上运行的软件系统也是多种多样的，所以 Internet 上的网络应用程序应该具有在各种不同的软硬件平台上均可正常工作的能力。Java 的工作机制使得它具有了这样的能力。

Java 的工作机制是这样的：编程人员首先编写好源代码，然后经编译生成一种二进制的中间码，称为字节码 (byte code)，最后再通过运行与操作系统平台环境相应的一种称为 Java 解释器的运行机构来执行编译生成的字节码。虽然不同的平台环境需要有各自相应的解释器，但是任何一个平台上的解释器，对于一段 Java 程序的字节码来说却是相同的，它们对 Java 字节码呈现出完全相同的面貌。也就是说，Java 的运行机制是利用解释器来隐藏网络上平台环境的差异性的。由此可见，Java 实现了二进制代码级的可移植性，在网络上实现了跨平台的特性。

Java 的解释器和运行时系统又称为“Java 虚拟机 (JVM, Java Virtual Machine)”，是驻留于计算机内存的虚拟计算机或逻辑计算机，实际上是一段负责解释执行 Java 字节码的程序。JVM 能够从字节码流中读取指令并解释指令的含义，每条指令都含有一个特殊的操作码，JVM 能够识别并执行它。从这个意义上说，Java 可以被称为是一种“解释型”的高级语言。

若 Java 解释器是一个独立的应用程序，并可以在操作系统下直接启动，那么它解释执行的程序被称为“Java 应用程序 (Application)”；若 Java 解释器包含在一个 WWW 的客户端浏览器内部，使得这个浏览器能够解释字节码程序，则这种浏览器能够自动执行的 Java 程序被称为“Java 小程序 (Applet)”。这两种程序从程序结构到运行机理都不相同，Java 应用程序多在本地或服务器上运行，而 Java 小程序则只能嵌入到网页中运行。

无论是 Java 应用程序还是 Java 小程序，其程序源代码文件都以 .java 为文件扩展名，源代码文件编译后产生的二进制字节码文件以 .class 为文件扩展名。

1.2 面向对象程序设计

Java 语言是一种面向对象 (OO, Object Oriented) 的程序设计语言。无论是 Java 应用程序还是 Java 小程序，它们都是以类为基础构建的。在认识 Java 程序前，这里先介绍一些

关于面向对象程序设计的概念。

1.2.1 传统与面向对象程序设计语言

传统的面向过程的程序设计方法从解决问题的每一个步骤入手，较适合于解决规模较小的问题。如广为流传的传统程序设计语言 BASIC、C 等采用面向过程的程序设计模型，但是由于这类语言本身几乎没有支持代码重用的语言结构，并且缺乏统一的接口，使得当程序的规模达到一定程度时，程序员很难控制其复杂性。面向对象的程序设计方法则按照现实世界的特点来管理复杂的事物，把它们抽象为对象，具有自己的状态和行为，通过对消息的反应来完成一定的任务。

面向对象程序设计（OOP）是当今计算机领域最流行的程序设计方法。这里的“对象”是什么含义呢？借用一个现实世界中“车”的例子来说明对象的含义。

在现实生活中，人们理解的“车”有各种各样的种类，如手推车、自行车、摩托车、汽车等。从各种车可归纳出它们的共性，如车有车轮、重量、颜色等，是汽车还有车速、耗油量等。这些是从“车”这类事物抽象出来的共性，也即所谓的数据、数据成员或属性。车不仅有这些静态的数据，还有很多与这些数据有关的动作和行为，如车的启动、加速、刹车和修理等，这就是所谓的代码、成员函数或方法。将上述车的数据和动作代码组合起来，就得到一个车类（class）。

在 Java 语言中定义一个关于车的类，一般的形式为：

```
class 车 { // 定义一个车类
    // 车的数据成员定义
    车轮数;
    车的颜色;
    车的重量;
    车速;
    ...
    // 车的成员方法定义
    启动();
    加速();
    刹车();
    修理();
    ...
}
```

有了抽象的车类后，一辆实际的车，如一辆自行车、一辆汽车等，就是车类的一个对象（或称实例）。对象是一个实体，而不像车是一个抽象概念。类是一类事物共性的反映，而对象是一类事物中的一个，是个性的反映。每个对象都有与其他对象不完全一样的特性。如您和我的自行车虽然都是自行车，但二者的颜色、重量等就不可能完全一致。

通过抽象来处理复杂事物的方法可从现实世界对应到计算机程序设计中来。传统的算法程序可以抽象成各种要处理的对象，一个对象就是数据和相关的方法的集合，其中，数据表明对象的状态和属性，方法表明对象所具有的行为、操作和运算功能，即类和对象将需要处理的数据和对这些数据进行的各种操作封装起来，当需要的时候，就可以向对象发送消息，使得对象能够调用这些方法进行相应的动作。这就是面向对象编程的基础。与人类理解复杂事物的方式一样，面向对象的概念构成了 Java 语言的核心。

1.2.2 对象的性质

面向对象程序设计技术具有封装、继承、多态三个主要特性。

1. 封装性

操纵汽车时，不用去考虑汽车内部各个零件如何运作的细节，而只需根据汽车可能的行为使用相应的方法即可。实际上，面向对象的程序设计实现了对象的封装，使用户不必关心对象的行为是如何实现这样一些细节的。从最基本的角度看，任何程序都包含两个部分：数据和代码。在传统的代码模型中，数据在内存中进行分配并由子程序或函数代码来处理。而面向对象设计的核心一环是将处理数据的代码、数据的声明和存储封装在一起。

可以把封装想像为一个将代码和数据包起来的保护膜。这个保护膜定义了对象的行为，并且保护代码和数据不被任何其他代码任意访问，即一个对象中的数据和代码相对于程序的其他部分是不可见的，它能防止那些不希望的交互和非法的访问。

Java 封装的基本单元是类，即 Java 程序的基本元素是类。用户可以创建自己的类，它是一组具有行为和结构的对象的一种抽象。对象是相关类的具体实例，是将类作为模子造出的一个翻版。因此，有时也称对象为类的实例。

封装的目的是为了减少复杂性，因此，类具有一套隐藏复杂性的机制。类中的每个数据和方法可以被定义为公共或私有。类的公共部分可以让外界的用户知道或必须知道，公共的数据和方法是类与外部的接口，程序的其他部分通过这个接口使用类的功能。而定义成私有的数据和方法则不能被类以外的其他代码访问。

借助于类的封装性，便可以将数据和方法像部件一样用于各种程序，而不必了解和记忆其内部细节。改变类的其他部分不会对整个程序产生预料之外的影响，只要保持类接口不变，它的内部工作方式可以随意改动。

需要改变对象的数据或状态，或需要进行对象之间的交互时，面向对象的程序设计方法提供消息机制。例如，要使汽车加速，必须发给它一个消息，告诉它进行何种动作（这里是加速）以及实现这种动作所要的参数（这里是需要达到的速度等）。这里要指明消息的接收者、接收对象应采用的方法、方法所需要的参数。同时，接收消息的对象在执行相应的方法后，可能会给发送消息的对象返回一些信息（如加速后，汽车的车速表上会出现已经达到的速度等）。

由于任何一个对象的所有行为都可以用方法来描述，通过消息机制就可以完全实现对象之间的交互，同时，处于不同处理过程甚至不同主机的对象间都可以通过消息实现交互。

2. 继承性

人们通常都会将世界看成相互关联的可划分层次的各种对象，如车、汽车和轿车。这里汽车是车的继承，而轿车又是汽车的继承。下一层次继承了上一层次的所有特性。一个多层次的继承关系构成了一个类树结构。

在已经定义了车类后，再定义一个汽车类，若再一次声明车轮数、颜色、重量等车类已定义的数据和方法，那么效率就太低了，其实只需定义汽车与一般意义下的车的不同即可。例如，增加定义一下汽车的车门数、座位数等，而继承车类中已定义过的数据和方法。这样可使得程序代码得到充分复用，结构更加清晰易懂，程序的可维护性、逻辑性就更强。

在面向对象的程序设计中，继承是指在已有类的基础上建立一个新类。新类自动拥有

父类的所有元素：数据成员和成员方法，然后再根据需要添加新任务所需的数据成员和成员方法。合理使用继承可以减少很多的重复劳动。若类实现了一个特别的功能，那么由该类继承的新类（派生类）就可以重复使用这些功能，而不再需要重新编程。对 Java 的内置类可以创建派生类，也可以对自己创建的类建立派生类。

一个不由任何类派生的类称为基类；一个派生类的最近的上层类叫做该类的父类；从某一类派生出来的类叫做该类的子类。通过父类和子类，实现了类的层次，可以从最一般的类开始，逐步特殊化，定义一系列的子类。子类的层次并不是越多越好，因为，若层次太多，还不如重新创建一个新类。

一个类从派生它的基类到它自身可能要经过好几个层次。类不仅能继承其父类的所有方法和实例变量，而且还能继承从它的基类开始到它自身之间经过的所有层次上的类的方法和实例变量。通过继承也实现了代码的复用，使程序的复杂性线性地增长，而不是呈几何级数增长。

在 Java 中，继承车类派生汽车类的一般形式为：

```
class 汽车 extends 车 { ... }
```

继承和封装具有很好的合作性。若一个给定类封装了某些属性，那么它的任何子类将继承这些属性并可增加它们特有的属性。

3. 多态性

不同的对象对于相同的方法表现其不同的理解和响应。可以想象的到：对于自行车和汽车，它们都能刹车，但它们的刹车方法却是完全不同的。

而在面向对象的程序设计语言中，多态性意味着一个对象具有多个面孔。Java 通过方法重载（overload）和方法覆盖（override，也称为方法重写）来实现多态。

在一个类中声明多个名字相同但参数不同（个数或类型）的方法，称为方法重载。在子类中声明与父类中完全相同的方法称为方法覆盖。

通过方法覆盖，子类可以重新实现父类的某些方法，使其具有自己的特征。例如对于汽车类的加速方法，其子类（如赛车）中可能增加了一些新的部件来改善和提高加速性能，这时可以在赛车类中覆盖父类的加速方法。覆盖隐藏了父类的方法，使子类拥有自己的具体实现，更进一步表明了与父类相比，子类所具有的特殊性。

Java 不仅允许程序设计者自己创建类，还针对各种应用提供了大量的预定义类库，例如屏幕显示、文件访问、数学计算等方面的类库。大量的预定义类的学习是 Java 语言学习的一个重点，也是一个难点。要掌握基本概念，通过对一些典型系统类的学习，学会举一反三，触类旁通。

有了有关类、对象的概念后，下面来看一看简单的 Java 程序。

1.3 Java 程序举例

由前面介绍已知，Java 程序形式分为应用程序和小程序两种，下面通过这两种程序形式的简单例子，对它们进行介绍。

1.3.1 Java 应用程序举例

【例 1.1】输出信息为 Hello World! 的 Java 应用程序。