



普通高等教育“十五”国家级规划教材

Java

程序设计

许满武 主编

严 悍 张 琨 李千目 编



高等教育出版社

普通高等教育“十五”国家级规划教材

Java 程 序 设 计

许满武 主 编

严 悍 张 琨 李千目 编

高等教育出版社

内容提要

Java 语言自 1995 年面世以来,不断改进完善,2005 年又增加了新的语言成分。本书汲取了 James Gosling 与 Ken Arnold 等人关于 Java 设计的精华,增添了目前 Java 语言最新发展的特色,融入了语言概念与原理的剖析,还与传统编程语言 C++ 进行了比较。本书内容共分五篇。第一篇介绍面向对象程序设计。第二篇介绍 Java 的命令式成分,包括运算符、表达式和控制流语句。第三篇介绍 Java 的面向对象机制,这是本书的核心,包括类、扩展类、接口和内嵌类型,并介绍了 Java 语言的最新成分。第四篇介绍 Java 语言基础类,包括异常、字符串、线程、类型编程、废区收集和内存管理、包。第五篇介绍 J2SE 类库,主要包括输入/输出、集聚类型及其他设施,也包括用于系统编程的类库。各章后配有适量的练习题。附录中给出 Java 语言和文档注释的最新语法规范。

作为“十五”国家级规划教材,本书可作为计算机专业的本科生的教学用书和研究生的参考教材,也可作为软件开发人员的自学用书和研究人员的参考用书。

图书在版编目(CIP)数据

Java 程序设计 / 许满武主编;严悍,张琨,李千目编.

北京:高等教育出版社,2006.5

ISBN 7-04-019645-X

I. J... II. ①许...②严...③张...④李...

III. Java 语言—程序设计—高等学校—教材

IV. TP312

中国版本图书馆 CIP 数据核字 (2006) 第 064174 号

策划编辑	倪文慧	责任编辑	俞丽莎	封面设计	于文燕
版式设计	张 岚	责任校对	朱惠芳	责任印制	宋克学

出版发行 高等教育出版社
社 址 北京市西城区德外大街 4 号
邮政编码 100011
总 机 010-58581000

购书热线 010-58581118
免费咨询 800-810-0598
网 址 <http://www.hep.edu.cn>
<http://www.hep.com.cn>
网上订购 <http://www.landaco.com>
<http://www.landaco.com.cn>
畅想教育 <http://www.widedu.com>

经 销 蓝色畅想图书发行有限公司
印 刷 高等教育出版社印刷厂

开 本 787×1092 1/16
印 张 29.75
字 数 670 000

版 次 2006 年 5 月第 1 版
印 次 2006 年 5 月第 1 次印刷
定 价 36.80 元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换。

版权所有 侵权必究

物料号 19645-00

前言

Java 语言自 1995 年面世以来, 以其简明的面向对象风格和与平台无关的特性迅速得到广泛的应用。近十年来, 经应用实践与理论探讨, Java 语言以及编程技术不断发展和完善, 目前已成为主流的软件研发工具。笔者在参与了国家教育委员会《面向二十一世纪计算机类教学内容和课程体系改革研究》项目后, 深感有必要将 Java 程序设计的最新面貌介绍给读者, 并且按教学改革的要求组织了本书的内容。

对于 Java 语言的结构成分与相关编程技术, 我们在国家自然科学基金和国家高新技术研究与发展计划持续资助下, 进行着不断的原理探讨, 在本、专科和研究生教学中进行了多次的讲授, 而且就所承担的软件开发项目完成了一定规模的工程实践, 这些工作为本书的编写奠定了基础。本书汲取了 James Gosling 与 Ken Arnold 等人关于 Java 设计的精华, 在尊重其原意的基础上, 增添了最新的 J2SE 5.0 版的新特色, 融入了语言概念与原理的剖析, 还与传统的面向对象编程语言 C++ 进行了比较。

本书内容分为五篇。第一篇介绍面向对象程序设计。第二篇介绍 Java 的命令式成分, 包括运算符、表达式和控制流语句。第三篇介绍 Java 的面向对象机制, 这是本书的核心, 包括类、扩展类、接口和内嵌类型, 并介绍了 Java 语言的最新成分。第四篇介绍 Java 语言基础类, 这些类以库的方式支持语言结构, 包括异常、字符串、线程、类型编程、废区收集和内存管理、包。第五篇介绍 J2SE 类库, 主要包括输入/输出、集聚类型及常用工具, 也包括用于系统编程的相关类库, 最后简单介绍了其他类库。各章后配有适量的练习题。附录中给出了 Java 语言和文档注释的最新语法规范。

普适与自主计算是近年来计算机界两项革命性的事件, Java 技术在其中表现出旺盛的生命力, 我们将继续关注 Java 技术的最新进展。本文不足之处, 竭诚希望广大读者指正。

编 者

2006 年 4 月

目 录

第一篇 面向对象程序设计

第 1 章 面向对象程序设计的基本概念	3	2.6.2 静态域或类域	16
1.1 对象	3	2.6.3 废区收集	17
1.2 类	3	2.7 方法和形参	17
1.3 类型	4	2.7.1 方法的调用	18
1.4 封装	5	2.7.2 this 引用	19
1.4.1 信息/实现隐藏	5	2.7.3 静态方法或类方法	19
1.4.2 状态保持	6	2.8 数组	20
1.5 继承	6	2.9 字符串对象	23
1.6 多态	7	2.10 类的扩展	24
1.7 小结	8	2.10.1 调用超类的方法	26
1.8 练习题	9	2.10.2 Object 类	26
第 2 章 Java 面向对象程序设计概述	10	2.10.3 类型强制转换	27
2.1 引言	10	2.11 接口	27
2.2 变量	11	2.12 异常	29
2.3 注释	12	2.13 包	31
2.4 命名常量	12	2.14 Java 的底层结构	33
2.5 控制流	13	2.15 其他主题	33
2.6 类和对象	15	2.16 小结	33
2.6.1 创建对象	15	2.17 练习题	34

第二篇 Java 语言的命令式成分

第 3 章 标记、运算符和表达式	37	3.2.1 基本类型	39
3.1 词法元素	37	3.2.2 引用值	40
3.1.1 字符集	37	3.2.3 布尔值	40
3.1.2 标记	37	3.2.4 字符值	40
3.1.3 标识符	38	3.2.5 整型值	40
3.1.4 关键字	38	3.2.6 浮点值	41
3.2 类型及其字面值	39	3.2.7 字符串值	41

II 目 录

3.2.8 类的字面值	41	3.7 运算符的优先级和结合性	56
3.3 变量	41	3.8 表达式	57
3.3.1 域和局部变量	42	3.8.1 运算顺序	57
3.3.2 形参	42	3.8.2 表达式类型	58
3.3.3 final 变量	43	3.8.3 隐式转换	58
3.3.4 初始值	44	3.8.4 显式转换	59
3.4 数组变量	44	3.8.5 字符串转换	60
3.4.1 数组修饰符	45	3.9 访问成员	60
3.4.2 数组的数组	45	3.10 小结	61
3.4.3 数组的初始化	46	3.11 练习题	61
3.4.4 数组和类型	47	第 4 章 流程控制	64
3.5 算术运算符	48	4.1 语句和分程序	64
3.5.1 整数运算	48	4.2 if-else	65
3.5.2 浮点运算	48	4.3 switch-case-default	66
3.5.3 严格和非严格的浮点运算	49	4.4 while 和 do-while	68
3.6 普通运算符	50	4.5 for	69
3.6.1 递增与递减运算符	50	4.6 标号	71
3.6.2 关系和判等运算符	51	4.7 break	71
3.6.3 逻辑运算符	52	4.8 continue	73
3.6.4 instanceof 运算符	53	4.9 return	74
3.6.5 按位运算符	53	4.10 没有 goto 语句的原因	75
3.6.6 条件运算符?	54	4.11 小结	75
3.6.7 赋值运算符	54	4.12 练习题	76
3.6.8 字符串连接	55		

第三篇 Java 语言的面向对象机制

第 5 章 类和对象	79	5.4 创建对象	83
5.1 一个简单的类	79	5.5 构造与初始化	84
5.1.1 类成员	80	5.5.1 构造器	84
5.1.2 类修饰符	80	5.5.2 初始化块	86
5.2 域	80	5.5.3 静态初始化	87
5.2.1 域初始化	81	5.6 方法	88
5.2.2 静态域	81	5.6.1 静态方法	89
5.2.3 final 域	82	5.6.2 方法调用	89
5.3 访问控制	82	5.6.3 方法执行和返回	90

5.6.4 形参值	91	第7章 接口	133
5.6.5 用方法来控制访问	93	7.1 接口举例	133
5.7 this 引用	95	7.2 接口的说明	136
5.8 方法的名复用	95	7.2.1 接口常量	136
5.9 main 方法	97	7.2.2 接口方法	136
5.10 native 方法	98	7.2.3 接口修饰符	137
5.11 小结	99	7.3 扩展接口	137
5.12 练习题	100	7.3.1 继承和隐藏常量	137
第6章 类的扩展	102	7.3.2 继承、改写和名复用	138
6.1 扩展类举例	102	7.4 接口的使用	139
6.2 扩展类中的构造器	106	7.4.1 接口的实现	140
6.3 方法的改写和域的隐藏	109	7.4.2 实现的使用	141
6.3.1 方法的改写	109	7.5 标记接口	142
6.3.2 域的隐藏	110	7.6 何时使用接口	142
6.3.3 访问继承成员	110	7.7 小结	143
6.3.4 可访问性与改写	111	7.8 练习题	144
6.3.5 隐藏静态成员	112	第8章 内嵌类型	145
6.3.6 super 关键字	112	8.1 静态内嵌类型	145
6.4 类型兼容与类型转换	113	8.1.1 静态内嵌类	145
6.4.1 兼容性	113	8.1.2 静态内嵌接口	146
6.4.2 显式的类型强制转换	113	8.2 内含类	146
6.4.3 类型测试	114	8.2.1 访问	148
6.5 protected 的真正含义	114	8.2.2 扩展内含类	148
6.6 final 方法和类	116	8.2.3 继承、作用域和隐藏	149
6.7 抽象类和方法	118	8.3 局部内含类	150
6.8 Object 类	119	8.4 匿名内含类	152
6.9 对象的克隆	120	8.5 内嵌类型的继承	153
6.9.1 克隆策略	120	8.6 接口中的内嵌类型	154
6.9.2 正确克隆	121	8.7 内嵌类型的实现	155
6.9.3 深度克隆和浅度克隆	124	8.8 名字的含义	156
6.10 扩展类的使用	124	8.9 小结	158
6.11 扩展使用的类	125	8.10 练习题	159
6.12 单继承和多继承	129	第9章 Java 新添语言成分	160
6.13 小结	130	9.1 断言	160
6.14 练习题	131	9.1.1 断言语句	160

IV 目 录

9.1.2 断言用法	161	9.5.2 增加数据和行为	179
9.1.3 编译和运行	166	9.5.3 增加特定行为	180
9.2 类属	166	9.5.4 EnumSet 和 EnumMap	181
9.2.1 需要类属的原因	167	9.6 可变形参	182
9.2.2 简单类属	167	9.7 静态移入	184
9.2.3 类属的类型兼容问题	168	9.8 注释	184
9.2.4 通配符	168	9.8.1 注释类型说明	185
9.2.5 类属方法	171	9.8.2 给程序添加注释	185
9.3 逐元循环	173	9.8.3 注释的应用	186
9.4 自动包装/解包	175	9.9 小结	188
9.5 枚举类型	176	9.10 练习题	189
9.5.1 枚举举例	177		

第四篇 Java 语言基础类

第 10 章 异常	193	11.8.1 修改缓冲区	214
10.1 创建异常类型	193	11.8.2 提取数据	216
10.2 throw	194	11.8.3 容量管理	216
10.2.1 控制转移	195	11.9 小结	217
10.2.2 异步异常	195	11.10 练习题	217
10.3 throws 子句	196	第 12 章 线程	218
10.3.1 throws 子句和方法改写	197	12.1 线程的创建	219
10.3.2 throws 子句和本地方法	197	12.2 使用 Runnable	221
10.4 try、catch 和 finally 子句	197	12.3 同步	224
10.5 何时使用异常	201	12.3.1 同步方法	225
10.6 小结	202	12.3.2 静态同步方法	226
10.7 练习题	203	12.3.3 同步语句	227
第 11 章 字符串	204	12.3.4 同步设计	230
11.1 基本字符串操作	204	12.4 线程通信	231
11.2 字符串比较	205	12.5 wait 和 notify 的细节	233
11.3 实用功能	208	12.6 线程调度	234
11.4 建立相关字符串	209	12.7 死锁	237
11.5 字符串转换	210	12.8 结束线程的执行	239
11.6 字符串与 char 数组	211	12.8.1 取消线程	239
11.7 字符串与 byte 数组	212	12.8.2 等待线程结束	241
11.8 StringBuffer 类	213	12.9 结束应用程序的执行	243

12.10 易变性 (volatile)	243	13.2.13 包	282
12.11 线程管理、安全性与线程组	244	13.2.14 Proxy 类	282
12.12 线程与异常	247	13.3 类的加载	284
12.13 ThreadLocal 变量	249	13.3.1 ClassLoader 类	286
12.14 调试线程	250	13.3.2 类的准备	289
12.15 小结	251	13.3.3 加载相关资源	289
12.16 练习题	251	13.4 小结	291
第 13 章 类型编程	253	13.5 练习题	292
13.1 包装类	253	第 14 章 废区收集与内存管理	293
13.1.1 Void 类	255	14.1 废区收集	293
13.1.2 Boolean 类	255	14.2 一个简单模型	294
13.1.3 Character 类	256	14.3 终结	295
13.1.4 Number 类	258	14.4 与废区收集器交互	297
13.1.5 整数包装类	259	14.5 可达状态与引用对象	298
13.1.6 浮点包装类	260	14.5.1 Reference 类	298
13.2 反射	261	14.5.2 引用的强度和可达性	299
13.2.1 Class 类	262	14.5.3 引用队列	301
13.2.2 类属的反射	265	14.6 小结	303
13.2.3 命名类	268	14.7 练习题	304
13.2.4 检查类成员	269	第 15 章 包	305
13.2.5 Modifier 类	272	15.1 包的命名	305
13.2.6 Field 类	273	15.2 类型移入	306
13.2.7 Method 类	274	15.3 包的访问	307
13.2.8 创建对象与 Constructor 类	276	15.4 包的内容	311
13.2.9 访问检查与 AccessibleObject 类	278	15.5 Package 对象	311
13.2.10 诠释的反射	278	15.6 小结	313
13.2.11 枚举的反射	280	15.7 练习题	314
13.2.12 数组	281		

第五篇 J2SE 类库

第 16 章 输入/输出包	317	16.2.1 Reader 类	322
16.1 字节流	318	16.2.2 Writer 类	324
16.1.1 InputStream 输入流	319	16.2.3 字符流和标准流	325
16.1.2 OutputStream 输出流	320	16.3 转换流	325
16.2 字符流	322	16.4 流类简介	326

VI 目 录

16.4.1 同步和并发	327	17.4 Collection 接口	370
16.4.2 Filter 流	328	17.5 Set 与 SortedSet 接口	371
16.4.3 Buffered 流	330	17.5.1 HashSet 类	373
16.4.4 Piped 流	330	17.5.2 TreeSet 类	373
16.4.5 ByteArray 字节流	332	17.6 List 接口	374
16.4.6 CharArray 字符流	333	17.6.1 ArrayList 类	375
16.4.7 String 字符串	334	17.6.2 LinkedList 类	375
16.4.8 Print 流	335	17.7 Map 与 SortedMap 接口	377
16.4.9 LineNumberReader 类	336	17.7.1 HashMap 类	379
16.4.10 SequenceInputStream 类	337	17.7.2 TreeMap 类	379
16.4.11 Pushback 流	338	17.7.3 WeakHashMap 类	380
16.4.12 StreamTokenizer 类	340	17.8 包装的集聚和 Collections 类	380
16.5 数据字节流	344	17.8.1 同步包装	380
16.5.1 DataInput 和 DataOutput 接口	344	17.8.2 不可修改的包装对象	382
16.5.2 Data 流类	345	17.8.3 Collections 工具方法	382
16.6 文件处理	346	17.9 Arrays 类	385
16.6.1 File 流和 FileDescriptor	346	17.10 Iterator 的实现	385
16.6.2 RandomAccessFile 类	347	17.11 Collection 的实现	387
16.6.3 File 类	348	17.12 遗留集聚类型	391
16.6.4 FilenameFilter 与 FileFilter 接口	352	17.12.1 Enumeration 接口	392
16.7 对象序化	353	17.12.2 Vector 类	392
16.7.1 Object 字节流	353	17.12.3 Stack 类	392
16.7.2 使类可序化	354	17.12.4 Dictionary 类	392
16.7.3 序化和去序化的顺序	355	17.12.5 Hashtable 类	393
16.7.4 定制序化	356	17.13 Properties 类	393
16.7.5 对象版本	359	17.14 小结	394
16.7.6 序化域	360	17.15 练习题	395
16.7.7 Externalizable 接口	362	第 18 章 其他设施	396
16.8 IOException 异常	362	18.1 BitSet 类	396
16.9 小结	364	18.2 Observer/Observable	398
16.10 练习题	364	18.3 Random 类	401
第 17 章 集聚类型	365	18.4 StringTokenizer 类	402
17.1 集聚	365	18.5 Timer 和 TimerTask 类	403
17.2 迭代	367	18.6 Date 类	405
17.3 排序	369	18.7 Math 和 StrictMath 类	406

18.8 小结	407	20.1 java.awt——抽象窗口工具	425
18.9 练习题	407	20.2 java.applet——小应用程序	427
第 19 章 系统编程	409	20.3 java.beans——构件	428
19.1 System 类	409	20.4 java.math——数学运算	429
19.1.1 标准 I/O 流	409	20.5 java.net——网络	430
19.1.2 系统性质	410	20.6 java.rmi——远程方法调用	433
19.1.3 实用工具	413	20.7 java.security——安全工具	437
19.2 创建进程	413	20.8 java.sql/javax.sql——关系 数据库访问	437
19.2.1 Process 类	413	20.9 java.nio——新输入/输出	440
19.2.2 进程的环境变量	415	20.10 java.text——多语种支持	441
19.2.3 可移植性	416	20.11 工具子包	441
19.3 停机	416	20.11.1 归档文件——java.util.jar	441
19.3.1 停机钩子	416	20.11.2 ZIP 文件——java.util.zip	442
19.3.2 停机序列	417	20.12 javax.*——标准扩展包	442
19.3.3 停机策略	418	20.13 javax.accessibility——GUI 的 易用性	442
19.4 Runtime 类的其他功能	418	20.14 javax.naming——命名与目录 服务	443
19.4.1 加载本地代码库	418	20.15 javax.sound——声音处理	444
19.4.2 调试	419	20.16 javax.swing——Swing GUI 构件	444
19.5 安全性	419	20.17 org.omg.CORBA—— CORBA API	444
19.5.1 SecurityManager 类	419	20.18 小结	445
19.5.2 许可	420		
19.5.3 安全策略	421		
19.5.4 访问控制与特权执行	422		
19.6 小结	423		
19.7 练习题	423		
第 20 章 其他包	424		
 附录 A Java 语言语法规范	 446		
附录 B 文档注释语法规范	461		
参考文献	463		

第一篇

面向对象程序设计

程序设计描述待解问题的求解。至今已出现了多种风格的程序设计。本篇介绍面向对象程序设计的基本概念和 Java 特定的面向对象程序设计的概貌。

1

第 1 章 面向对象程序设计的基本概念

面向对象程序设计需要一些基本概念作为其理论基础。本章简要介绍对象、类、类型的基本概念，讨论面向对象程序设计的基本特性：封装性、继承性和多态性。

1.1 对象

面向对象程序设计是对应用领域中问题求解过程的内部结构的模拟。问题求解的实体单位具有“知”与“行”的能力，这样的实体单位称之为对象。对象的“知”表现为它所持有的值，对象的“行”表现为它在持有值上所进行的操作。

面向对象软件开发，就是分析问题域中这样的具有“知”与“行”能力的客观实体，通过某种规范的途径最终编制出计算机可执行的软件的过程。

面向对象软件的运行，就是它自己产生出的那些对象的生命活动。尽管多种面向对象软件的运行机制或有差异，但对于每个对象而言，总有创建、生存和消亡的过程。整个面向对象计算由一组对象合作完成，合作的手段是发送消息，以提交计算任务和获取计算结果。

综上所述，对象是面向对象程序设计系统中运行时刻的基本成分；它是“知”与“行”（即值与操作）的封装体，其中还包括与其他对象进行信息传递的机制。

1.2 类

如前所述，对象是系统中运行时刻的基本成分。那么它在程序中又是如何反映的呢？从认识论角度来分析，一种是采用抽象的途径；另一种是基于原型的途径。先就抽象途径进行分析。一般来说，系统中往往存在多个具有共同特性的对象。例如， $O_1, O_2, \dots, O_n$ 等对象全都具有特性 $P_1, P_2, \dots, P_m$ ，则具有特性 $P_1, P_2, \dots, P_m$ 的抽象对象便是 $O_1, O_2, \dots, O_n$ 等对象的一个抽象。在语言中，这样的抽象称之为“类”。换言之，类刻画了一组具有共同特性的对象。在基于类的面向对象程序设计语言中，程序由一个或多个类组成。在运行时，按需要创

4 第一篇 面向对象程序设计

建该类的个别对象（即其实例）。这里，类是静态概念，对象是动态概念。类可以看作是抽象数据类型的一种实现。这里，抽象的含义是指与表示和实现无关，主要表现在两个方面：一是指不涉及值的具体表示（只涉及值域）；二是操作与具体实现无关（只刻画操作所满足的抽象性质）。

在基于类的面向对象程序设计语言中，类的作用有两个：一是作为对象的描述机制，刻划一组对象的公共属性和行为；二是作为程序的基本单位，支持模块化设计。

结构化程序设计是基于系统功能来对系统进行分解的，因而系统的结构是与功能相关的。因划分方法具有较大的随意性，所以功能改变后，结构就要改变。在面向对象程序设计中，基于类的面向对象程序设计语言的模块化设计则是基于类的层次关系（对应客观世界中概念的包含关系）自然地分解系统，划分方法相对规范，系统结构相对稳定。

值得指出的是，除了采用抽象途径之外，还有基于原型的途径。前者简明规范，而后者更容易反映问题域对象的灵活多样，对应的程序设计语言的结构相对复杂，但基于类的面向对象程序设计语言易于学习掌握，目前已推出多种实用且有成效的程序设计语言，Java 便是重要的代表。

1.3 类型

在计算机科学和数理逻辑的文献中，类型（type）一词在不同场合可有不同含义。一般来说，在任何一个类型系统中，类型是对讨论的值所构成的论域所进行一种划分或分类。一个类型是共享某些性质的值的集合。在程序设计语言中，类型是一种重要的工具，因为类型可以采用某种方式加以检测，检测时间可以是在编译时刻，也可以在运行时刻。编译时刻的类型检测，用以保证程序中每个短语、表达式、说明等定义了具有某种类型的元素。这个类型，或者在程序正文中显式地定义，或者通过该短语、表达式被使用的方式隐式地定义。类型检测失败的程序，将被编译程序拒绝接受。

编译时刻类型检测的好处如下：

- ① 较早地发现程序中类型不相适应的错误。
- ② 确定运行时刻预期值的性质，有利于生成优化的目标代码。
- ③ 便于阅读与编制文档。

运行时刻的类型检测用于在程序运行时去阻止执行错误的计算，检测的这些错误必须是在运行时刻真正发生的。由于操作数的真正值一般直到运行的即刻才能知晓，故从必须且只须排除运行时刻真正会发生的错误角度来分析，编译时刻有可能拒绝了在运行时并不产生错误的程序，那么编译时刻类型检测的做法则是保守的。

在程序设计语言中，以可实施的操作来表示同一类型中值所共享的性质，特别是在面向对象程序设计语言中，例如 Java，对象本身为值，一个类型规定了所含对象共享的一组操作。其他常用的值，如整数 0、1、-1、2、-2…连同其上的操作，如“加”、“减”、“乘”等，便构成通常所说的整数类型，简称整型。布尔值有“真”和“假”，连同其上的操作“与”、“或”、“非”

等，便构成通常所说的布尔类型，亦简称布尔型。

Java 语言中提供了灵活丰富的类型系统，本书后续章节将给出详细的描述。

1.4 封装

封装（encapsulation）是把相关概念组成一个单元，然后可通过一个名字来引用它。

面向对象封装是把表示状态的属性和对状态的操作包装成一个对象类型，使得对状态的存取只能通过封装提供的接口进行。

一个包（package）封装了一组类型，并提供了这些类型的命名空间，即一个包中的各类型具有唯一名称。一个类的完整名称应以包名作为其前缀，如“包名.子包...类名”。包中的类可选择是否对外部可见。

一个类中封装一组属性和一组操作作为该类的成员，各成员在类的命名空间中具有唯一的名称或基调（signature），关于基调的详细内容可参见 2.7 节。

类中每个成员都具有可见性说明，以确定该成员对该类外部是否可访问。一般来说，可见性说明主要有以下三种：

- ① **private**：私有，只有本类内部可访问。
- ② **public**：公有，类外部可访问，只要类是可见的，公有成员就可访问。
- ③ **protected**：受保护，本类及子类可访问。

通常类的属性被说明为私有，而操作说明为公有，这样只能通过操作来访问、修改对象的状态，可保护关键数据免遭滥用和破坏。

从另一个角度看，一个类封装了类的成员和对象的成员。类中每个成员都要确定是否为静态（static）。静态成员即为类的成员，非静态成员是对象的成员。访问一个静态成员应以类名作为前缀，如“类名.静态成员名”；而对非静态成员的访问应先区分对象，再确定成员，如“对象名.非静态成员名”。

1.4.1 信息/实现隐藏

从对象外部来看，良好的封装能隐藏大量细节。信息/实现隐藏是使用封装将某些信息或实现方法限制在封装结构内部，以约束外部的可见性。隐藏有两种形式：信息隐藏和实现隐藏。所谓信息隐藏是使封装单元内的信息不被外界察觉，而实现隐藏是指外界不能察觉单元内的实现细节。

信息/实现隐藏是降低软件复杂度的有效手段。对象外部可把对象看作一个黑盒子，即外部知道对象可以做什么，而不知道对象如何做，也不知道对象内部如何构造。

信息/实现隐藏有以下两个主要优点：

① 设计决策局部化。在对象内私有的设计决策对系统其他部分影响很小，甚至没有影响，因此，对局部决策的修改对整个系统的影响最小，这样限制了“修改蔓延”的影响。

② 减少信息内容。因对象对外部隐藏了实现细节，这使外部用户不会受到对象内部信息特定格式的困扰，这样对象外部用户，即其他程序员，就不用干涉对象内部的事情，也防止向对象内部引入不稳定的连接。

1.4.2 状态保持

在对象封装基础上，对象具有保持自己状态的能力。当传统的过程模块（如函数、子程序、过程等）返回到调用方，只是把计算结果返回，而不能保持自己的状态。当同一个过程模块再次调用时与首次调用一样，模块对以前的状态没有记忆。

对象在自身内部封装了状态信息，并保持直到对象消亡。在对象生命周期中可被多次调用，每次状态改变都能保持记忆。而且，对象如何保持其状态是隐藏在内部的细节。外部无需知道如何处理。

面向对象封装、信息/实现隐藏及状态保持是面向对象机制的核心。

1.5 继承

继承是类与类之间存在的一种关系，它使程序员可在已有类的基础上定义和实现新类。继承是构造可复用软件构件的有效机制。

继承能有效地支持类的复用，使得当需要在系统中增加新特征时，所需的新代码最少，并且将继承和下面将介绍的多态相结合使用时，修改系统所需变动的源代码为最少。

当类 Y 继承了类 X 时（如图 1-1 所示），称 X 是基类（Java 语言中称为超类（superclass）），Y 是 X 的派生类（Java 中称为子类（subclass））。在这种情形下，Y 由两部分组成：继承部分和新增部分。继承部分是从类 X 继承得到的，新增部分是专为 Y 编写的新代码。从这个角度看，继承的作用在于避免编写大量重复的代码。

从 X 中特征到 Y 中继承部分相应特征的映射是由语言规则和程序员在 Y 中所编写的代码共同确定的。这一映射可以是一种简单的恒等变换，使得 Y 的继承部分同 X 完全一样；也可重新定义，对所继承特征进行调整。与所有语言机制一样，继承使用不当会带来严重后果，Java 语言为此做了精心的考虑。

类与类之间的继承关系反映了相应概念间的包含关系，这是因为；当类 Y 是类 X 的派生类时，Y 通过继承 X 而获得了 X 的所有特征，所以从概念上讲，X 包含 Y，或者说，Y 是一种 X。因此，继承关系可用来表示问题域中已有的抽象结构。以几何图形间的关系为例：矩形是一种特殊的多边形。这一关系很容易用继承关系来说明（如图 1-2 所示）。作为多边形的派生类，矩形又具有多边形的所有特征。另外，由于多边形是闭合图形的派生类，矩形也继承了闭合图形的特征。

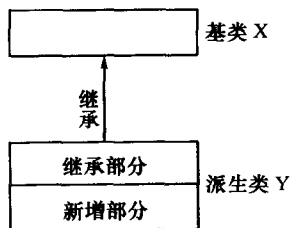


图 1-1 类 Y 继承类 X