

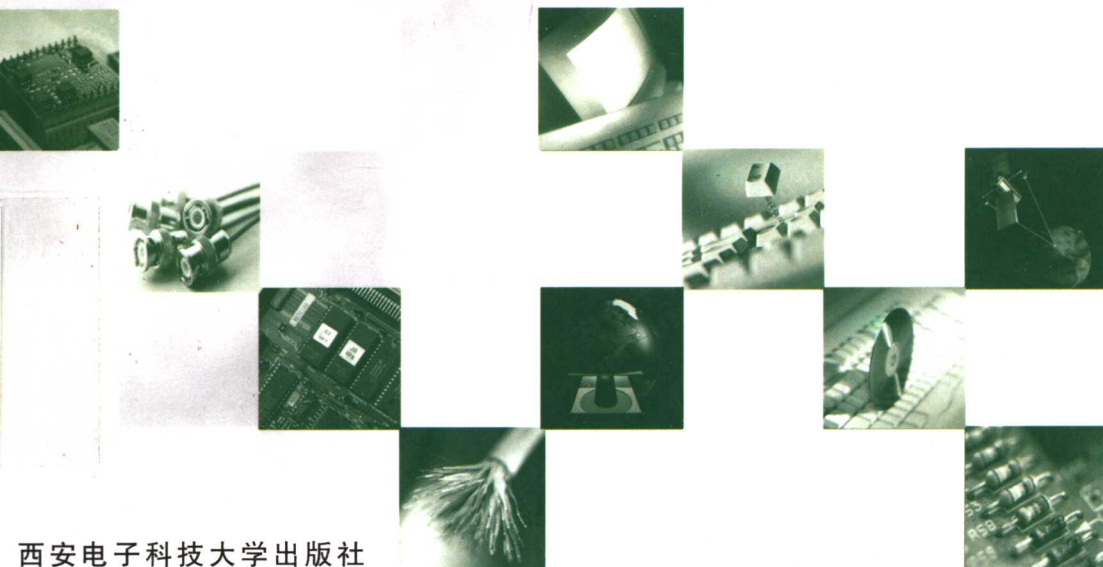
普通高等院校计算机类专业系列教材

Object-Oriented Program Design—Java

面向对象程序设计

——Java (第二版)

张白一 崔尚林 编著





普通高等院校计算机类专业系列教材

面向对象程序设计

——Java

(第二版)

Object-Oriented Program Design—Java

张白一 崔尚森 编著

西安电子科技大学出版社

2006

内 容 简 介

本书将面向对象的理论与 Java 语言程序设计技术相结合,意在培养读者正确运用面向对象的思维方法分析问题和解决问题的能力。全书共分 16 章。前 6 章主要介绍面向对象的基本理论、原理、技术方法和 Java 语言基础知识,阐述了面向对象程序设计的基本原则和特点。第 7 章介绍字符串类。这次修订新增的第 8 章介绍了在没有指针类型的 Java 语言中进行链表操作的技术。从第 9 章开始的以后各章介绍 Java 的常用标准类库及编程技巧,主要包括 GUI 设计、Swing 组件、异常处理、多线程技术、输入/输出技术、网络编程技术和 JDBC 数据库应用编程技术等。

本书可作为大专院校相关课程的教材,也可作为对面向对象编程技术和 Java 语言感兴趣的读者的自学用书。

为方便教学和实践,本书配有光盘一张,其中包括电子教案、示例程序源代码及相关工具软件。

图书在版编目(CIP)数据

面向对象程序设计:Java / 张白一等编著. —2 版. 西安:西安电子科技大学出版社, 2006.1

(普通高等院校计算机类专业系列教材)

ISBN 7-5606-1605-4

I. 面… II. 张… 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(2005)第 134030 号

责任编辑 阎 彬 云立实

出版发行 西安电子科技大学出版社(西安市太白南路 2 号)

电 话 (029)88242885 88201467 邮 编 710071

http://www.xduph.com E-mail: xdupfxb@pub.xaonline.com

经 销 新华书店

印刷单位 陕西天意印务有限责任公司

版 次 2003 年 2 月第 1 版 2006 年 1 月第 2 版 2006 年 1 月第 5 次印刷

开 本 787 毫米×1092 毫米 1/16 印 张 23.5

字 数 548 千字

印 数 22 001~26 000 册

定 价 32.00 元(含光盘)

26.00 元

ISBN 7-5606-1605-4/TP·0922

XDUP 1897002-5

如有印装问题可调换

本社图书封面为激光防伪覆膜,谨防盗版。

普通高等院校计算机类专业系列教材

编审专家委员会名单

主任委员 冯博琴

副主任委员 陈建铎 李伟华 武 波 李荣才

委 员 (按姓氏笔画排列)

巨永锋 冯德民 石美红 朱明放

何东健 陈 桦 李长河 李晋惠

李银兴 张俊兰 孟东升 赵文静

耿国华 龚尚福

项目策划 陈宇光 马乐惠

策 划 云立实 马武装 臧延新 马晓娟

电子教案 马武装

第一版前言

面向对象技术引起了程序设计方法学的一场革命，它已经替代面向过程的程序设计技术，成为当今计算机应用开发领域的主流技术。其原因主要在于面向对象技术能够比较客观地模拟现实世界，能够使软件开发人员运用人类认识事物所采用的一般思维方法进行软件开发；其次是在面向对象的程序设计中将数据与操作捆绑在一起，符合现代大规模软件开发的高可靠性和易维护性等方面的要求。计算机网络的发展，要求程序设计语言具有安全性强、可移植性好、与具体的操作平台无关等特性。Java 语言正是为满足这些要求而设计与研发的，并以网络为发展方向，随着网络的发展而兴盛。

1995 年 Java 语言刚一推出，便以其纯面向对象、平台无关性、多线程、高安全性、良好的可移植性和可扩展性等特征，受到了计算机界的普遍欢迎，并得到了广泛的应用和发展。近几年来，Java 的应用已经扩展到各个应用领域，加上各种功能配件的推陈出新，使得 Java 能够满足产品开发的需求，成为网络时代最流行的程序设计语言。利用 Java 来开发软件，具有跨平台、易整合、易扩展的优点。有人预言，不久的将来全世界 90% 的程序代码将用 Java 语言书写或改写。

本书是为大专院校的学生及其他对面向对象编程技术和 Java 语言感兴趣的读者编写的，意在培养读者正确使用面向对象的思维方法分析问题和解决问题的能力，让读者学会利用当今最先进的软件开发工具开发软件产品，以适应网络时代社会对人才的需求。根据作者多年来讲授“面向对象程序设计”及其相关课程的经验，本书在内容的取舍上作了精心的选择，确保有一定的深度和广度；在内容的编排上体现了由浅入深、循序渐进的学习规律；在写作风格上立

足于理论与实践相结合,将复杂的面向对象理论融会于众多的实例之中,使读者学会面对一个具体的问题时,能够用面向对象的思维方法分析问题,并利用面向对象的语言编写解决实际问题的计算机程序。本书可作为大专院校相关课程的教材,也可作为对面向对象编程技术和 Java 语言感兴趣的读者的自学用书。

全书共分 15 章。第 1 章介绍 Java 的特点和运行环境。第 2 章和第 3 章讲述程序设计的基本语法规则和程序流程控制。第 4 章和第 5 章全面讲述面向对象的理论和程序设计技术,包括类与对象、抽象与封装、消息、继承、多态等诸多概念及其在程序设计中的具体应用。第 6 章则介绍数组这种在各种程序设计语言中常用的数据结构在 Java 中的应用技术。从第 7 章开始介绍 Java 的常用标准类库及其编程技巧。第 7 章介绍字符串类。第 8~10 章讲述 Java 图形用户界面的设计与编程实现技术,并以最新的 Swing GUI 组件为主。第 11 章介绍 Java 的异常处理。第 12 章讲述多线程技术。第 13 章讲述程序的输入与输出技术。第 14 章和第 15 章分别介绍 Java 的网络编程技术和数据库应用编程技术。

要想很好地掌握面向对象技术,学习用 Java 程序设计语言编写出高质量的程序,先要学习其语法规则,这是编写 Java 程序的基本功;还要学习使用类库,这是提高编程效率和质量的必由之路,甚至从一定程度上来说,是否能熟练自如地掌握尽可能多的 Java 类库,决定了一个 Java 程序员编程能力的高低。此外,计算机学科是注重实践的学科,优秀的软件人员无不经过大量的上机实践的磨练,因此,读者只有在学习书本内容的同时辅以相应的实际练习,才能真正掌握书中介绍的知识和技能。

本书第 1~7 章由张白一编写,第 8~15 章由崔尚森编写。在本书的编写过程中,参考了许多相关书籍和网站,得到了许多同仁和同事的支持与帮助,在此我们一并表示感谢。首先应该感谢支持和参与本课程教学的长安大学的同学们,正是他们活跃的思维和永无止境的求知欲帮助作者发现内部试用版的错误,

鞭策作者不断改进与完善书稿。同时，特别感谢赵文静教授在百忙之中仔细审阅了本书全稿，并提出了许多宝贵的修改意见，使本书更趋完善。此外，还要衷心感谢陕西省计算机教育学会、西安电子科技大学出版社以及长安大学的领导和同事们，正是由于他们的大力支持，才使得本书与广大读者见面。

尽管书稿几经修改，但由于作者水平所限，书中难免有疏漏之处，我们热诚地欢迎各位同行和广大读者批评指正。

作 者

2002 年 8 月



第二版前言

Java 语言推出至今已有 20 个年头了，在这 20 年中，尤其是近几年来，网络编程技术的飞速发展使得 Java 语言受到了广泛的欢迎并得到迅速的发展，许多网络软件开发人员将 Java 语言作为首选的开发工具，国内许多高校也相继开设了 Java 编程方面的课程。承蒙读者厚爱，我们于 2002 年编写并出版的《面向对象程序设计——Java》一书已多次印刷，许多读者借助该书学会了 Java 编程，同时也提出了一些宝贵的意见和建议。此外，随着 Java 版本的升级换代，书中的一些内容也需要更新。为适应这种变化，我们对该书进行了修改和补充。

与第一版相比，首先，我们在开发环境上使用了运行于 Windows XP 操作系统上的最新版本 jdk-1_5_0_04 对各个章节进行了修改，对书中所附算法和示例程序全部进行了重新调试和改写，增加了注释与图解等。其次，根据 Java 2D API 的特性，对与 GUI 设计有关的章节进行了彻底的更新。第三，为了使读者对面向对象技术有一个更深入的理解，我们在本书中新增了“链表”一章，介绍了链接存储结构的概念和特点以及在没有指针类型的 Java 语言中进行链表操作的技术。

全书共分 16 章。第 1 章介绍 Java 的特点和运行环境。第 2 章和第 3 章讲述程序设计的基本语法规则和程序流程控制。第 4 章和第 5 章全面讲述面向对象的理论和程序设计技术，包括类与对象、抽象与封装、消息、继承、多态等诸多概念及其在程序设计中的具体应用。第 6 章和第 8 章介绍数组和链表这两种常用的数据结构在 Java 中的应用技术。第 7 章介绍字符串类。从第 9 章开始的以后各章介绍 Java 的常用标准类库及其编程技巧。其中，第 9~11 章以最新的 Java 2D API 和 Swing GUI 组件为主讲述 Java 图形用户界面的设计与编程实现技

术；第 12 章介绍 Java 的异常处理；第 13 章讲述多线程技术；第 14 章讲述程序的输入与输出技术；第 15 章和第 16 章分别介绍 Java 的网络编程技术和数据库应用编程技术。

本书在第一版的基础上，由张白一完成了原第 1~7 章的改写和完善工作，由崔尚森完成了原第 8~15 章(本版为第 9~16 章)的改写和完善工作，新增的第 8 章由张向一编写。

虽然这已是该书的第二版，但因为 Java 开发环境的不断发展及作者水平有限，书中仍难免有疏漏之处，我们热诚地欢迎各位同行和广大读者批评指正。对于本书的各种意见和建议可直接发送 E-mail 到 byzhang@chd.edu.cn。

作 者

2005 年 10 月



目 录

第 1 章 Java 系统环境概述	1	2.3.1 算术表达式	27
1.1 编程语言的发展	1	2.3.2 赋值表达式	31
1.1.1 机器语言	2	2.3.3 表达式语句	32
1.1.2 汇编语言	2	2.3.4 关系表达式	33
1.1.3 高级语言	2	2.3.5 逻辑表达式	34
1.1.4 面向对象的语言	3	2.3.6 位运算	35
1.1.5 面向对象语言的发展	4	2.3.7 运算符的优先级	36
1.2 网络时代的编程语言——Java	4	习题 2	36
1.2.1 Java 的产生	4		
1.2.2 Java 的特点	5	第 3 章 程序流程控制	38
1.3 Java 的开发运行环境	9	3.1 选择结构程序设计	38
1.3.1 建立 Java 2 SDK 开发环境	9	3.1.1 if 语句	38
1.3.2 Java 工具集	11	3.1.2 switch 语句	43
1.4 Java 程序的运行步骤	11	3.1.3 条件运算符	45
1.4.1 运行系统的结构及工作原理	12	3.2 循环结构程序设计	46
1.4.2 Java Application 程序的建立 及运行	12	3.2.1 while 语句	46
1.4.3 Java Applet 程序的建立及运行	14	3.2.2 do-while 语句	47
1.4.4 Java 虚拟机	16	3.2.3 for 语句	48
习题 1	17	3.2.4 for 语句头的变化与逗号 运算符	49
		3.2.5 循环语句比较	50
第 2 章 Java 语言基础	18	3.2.6 循环控制要点	51
2.1 Java 符号集	18	3.2.7 循环嵌套	53
2.1.1 标识符及其命名	18	3.3 break 和 continue 语句	55
2.1.2 关键字	19	3.3.1 break 语句	55
2.1.3 运算符	19	3.3.2 continue 语句	57
2.1.4 分隔符	20	习题 3	58
2.1.5 注释	20		
2.2 数据类型、常量与变量	20	第 4 章 类与对象	60
2.2.1 数据类型的概念	20	4.1 类与对象的概念	60
2.2.2 常量	21	4.1.1 抽象原则	60
2.2.3 变量	23	4.1.2 对象	61
2.2.4 引用类型	27	4.1.3 类	61
2.3 表达式和语句	27	4.1.4 类与对象的关系	62

4.1.5 定义类的一般格式.....	63	5.3 多态机制.....	97
4.1.6 Java 类库.....	64	5.3.1 多态的概念.....	98
4.1.7 创建对象.....	66	5.3.2 重载.....	98
4.1.8 使用对象.....	68	5.3.3 覆盖.....	99
4.1.9 对象的初始化与构造方法.....	69	5.4 继承机制.....	99
4.2 封装机制.....	71	5.4.1 继承的概念.....	99
4.2.1 封装的概念.....	71	5.4.2 继承的特征.....	100
4.2.2 类的严谨定义.....	72	5.4.3 Java 用 extends 指明继承关系.....	100
4.2.3 类修饰符.....	72	5.4.4 this 与 super.....	103
4.3 数据成员.....	75	5.4.5 构造方法的重载与继承.....	107
4.3.1 数据成员的声明.....	75	5.4.6 向方法传递对象.....	109
4.3.2 用 static 修饰的静态数据成员.....	75	5.4.7 继承与封装的关系.....	110
4.3.3 静态数据成员的初始化.....	76	5.5 抽象类、接口与包.....	111
4.3.4 用 Final 修饰的最终数据成员.....	77	5.5.1 抽象类.....	111
4.4 成员方法.....	78	5.5.2 接口.....	114
4.4.1 成员方法的分类.....	78	5.5.3 包与程序复用.....	120
4.4.2 声明成员方法的格式.....	79	习题 5.....	124
4.4.3 方法体中的局部变量.....	79		
4.4.4 成员方法的返回值.....	80	第 6 章 数组.....	126
4.4.5 形式参数与实际参数.....	81	6.1 一维数组.....	127
4.4.6 成员方法的引用方式.....	83	6.1.1 一维数组的声明.....	127
4.4.7 引用成员方法时应注意的事项.....	83	6.1.2 一维数组的初始化.....	127
4.4.8 成员方法的递归引用.....	83	6.1.3 一维数组的引用.....	128
4.4.9 用 static 修饰的静态方法.....	86	6.2 一维数组引用举例.....	129
4.4.10 数学函数类方法.....	87	6.2.1 测定数组的长度.....	129
4.4.11 用 final 修饰的最终方法.....	88	6.2.2 数组下标的灵活使用.....	130
4.4.12 用 native 修饰的本地方法.....	89	6.2.3 数组间相互赋值.....	133
习题 4.....	89	6.2.4 向成员方法传递数组元素.....	134
		6.2.5 向成员方法传递数组.....	134
第 5 章 消息、继承与多态.....	91	6.2.6 同类型不同长度的数组引用 同一个成员方法.....	136
5.1 消息.....	91	6.2.7 数组元素排序.....	137
5.1.1 消息的概念.....	91	6.2.8 对象数组.....	140
5.1.2 公有消息和私有消息.....	92	6.3 二维数组.....	142
5.1.3 特定于对象的消息.....	92	6.3.1 二维数组的声明.....	143
5.2 访问控制.....	94	6.3.2 二维数组的初始化.....	143
5.2.1 公共访问控制符 public.....	94	6.4 二维数组的引用.....	145
5.2.2 缺省访问控制符.....	95	6.4.1 测定数组的长度及数组赋值.....	145
5.2.3 私有访问控制符 private.....	96	6.4.2 数组名作为成员方法的参数.....	147
5.2.4 保护访问控制符 protected.....	97		

习题 6.....	149	8.3.6 查找单链表中的一个结点.....	178
第 7 章 字符串类	151	8.3.7 按给定条件向单链表中插入 一个结点.....	180
7.1 String 类.....	151	8.3.8 删除单链表中的一个结点.....	182
7.1.1 创建 String 对象.....	151	8.3.9 单链表基本操作示例.....	184
7.1.2 String 类的构造方法.....	152	8.4 单链表的其他操作.....	187
7.1.3 String 类的常用方法.....	154	8.4.1 修改链表中结点的数据值.....	187
7.1.4 访问字符串对象.....	154	8.4.2 链表的排序.....	188
7.1.5 字符串比较.....	156	8.4.3 链表的修改与排序应用示例.....	190
7.1.6 字符串操作.....	157	习题 8.....	191
7.1.7 将其他类型的数据转换成 字符串.....	158	第 9 章 文字与图形 GUI 设计	193
7.1.8 main 方法中的参数.....	159	9.1 GUI 设计概述.....	193
7.2 StringBuffer 类.....	160	9.1.1 图形用户界面元素分类.....	193
7.2.1 创建 StringBuffer 对象.....	160	9.1.2 Applet 的执行程序.....	194
7.2.2 StringBuffer 类的常用方法.....	160	9.1.3 JApplet 类.....	195
7.2.3 StringBuffer 类的测试缓冲区 长度的方法.....	161	9.1.4 Java 2D API.....	195
7.2.4 StringBuffer 类的 append() 方法.....	162	9.1.5 与图形文字有关的类及其 继承关系.....	196
7.2.5 StringBuffer 类的 insert() 方法.....	163	9.1.6 Java 2D API 坐标系统.....	196
7.2.6 StringBuffer 类的 setCharAt() 方法.....	164	9.1.7 三种图形对象.....	196
习题 7.....	164	9.1.8 Graphics2D 对象的属性设置.....	196
第 8 章 链表	166	9.2 绘制文字.....	197
8.1 链接存储结构的概念.....	166	9.2.1 绘制文字的成员方法.....	197
8.1.1 顺序存储结构的优缺点.....	166	9.2.2 Font 类.....	199
8.1.2 链接存储的概念.....	167	9.3 Color 类.....	201
8.1.3 链接存储结构的优缺点.....	168	9.3.1 Color 类的构造方法.....	201
8.2 链表结点类.....	169	9.3.2 Color 类的数据成员常量.....	201
8.2.1 单链表结点类.....	169	9.3.3 Color 类的成员方法.....	202
8.2.2 双链表结点类.....	170	9.3.4 应用举例.....	203
8.2.3 创建单链表结点类的应用示例.....	170	9.4 绘制形状图形.....	203
8.2.4 单链表中结点的链接方法.....	171	9.4.1 绘制几何图形的方法与步骤.....	203
8.3 单链表类及其基本操作.....	172	9.4.2 绘制线段与矩形.....	204
8.3.1 单链表类的抽象描述.....	172	9.4.3 绘制椭圆、圆及弧.....	207
8.3.2 创建一个空链表.....	173	9.4.4 绘制多边形.....	209
8.3.3 向单链表中添加结点.....	174	9.4.5 图形重叠时的色彩设置.....	210
8.3.4 遍历单链表.....	175	9.4.6 绘制剪切文字图形.....	212
8.3.5 单链表的创建与遍历示例程序.....	176	习题 9.....	214

第 10 章 常用组件 GUI 设计	216	11.1 界面布局管理	241
10.1 Swing 概述	216	11.1.1 FlowLayout.....	241
10.1.1 Swing 中常用的包.....	216	11.1.2 BorderLayout.....	242
10.1.2 常用组件的继承关系.....	217	11.1.3 CardLayout.....	243
10.2 事件响应原理.....	218	11.1.4 GridLayout	244
10.2.1 委托事件模型.....	218	11.1.5 BoxLayout	245
10.2.2 Swing 组件的事件及监听者.....	218	11.2 键盘事件(KeyEvent).....	248
10.2.3 Java.awt 事件类继承关系	221	11.3 鼠标事件(MouseEvent).....	249
10.2.4 AWT 中的事件与事件监听者	221	11.4 窗口与面板.....	251
10.3 JLabel 组件	223	11.4.1 JFrame 容器	251
10.4 JButton 组件与 JToggleButton 组件.....	223	11.4.2 窗口事件(WindowEvent)	253
10.4.1 AbstractButton 类的常用		11.4.3 JPanel 容器.....	255
成员方法	224	11.4.4 JScrollPane 容器	257
10.4.2 JButton 类的构造方法	225	11.4.5 JScrollbar 组件	258
10.4.3 JToggleButton 类的构造方法	225	11.4.6 JTabbedPane 容器	262
10.4.4 ActionEvent 事件及其响应.....	225	11.5 菜单设计	264
10.4.5 应用举例	226	11.6 对话框设计	266
10.5 JCheckBox 和 JRadioButton 组件	228	11.6.1 JOptionPane 类对话框.....	267
10.5.1 JCheckBox 类的构造方法	228	11.6.2 JDialog 类对话框.....	274
10.5.2 JradioButton 类的构造方法	228	习题 11.....	277
10.5.3 ItemEvent 事件	229		
10.5.4 应用举例	229	第 12 章 异常处理	279
10.6 JComboBox 组件.....	231	12.1 Java 的异常处理机制.....	279
10.6.1 JComboBox 类的构造方法		12.1.1 异常处理机制的结构	280
及成员方法	232	12.1.2 异常类的继承关系.....	280
10.6.2 事件响应.....	232	12.2 Java 的异常处理语句.....	283
10.7 JList 组件	233	12.2.1 try-catch-finally 语句	283
10.7.1 JList 类的构造方法及成员方法	234	12.2.2 嵌套 try-catch-finally 语句	284
10.7.2 ListSelectionEvent 事件	234	12.2.3 抛出异常的 throw 语句与	
10.8 JTextField 与 JTextArea 组件	236	throws 语句	285
10.8.1 JTextField 组件的构造方法		习题 12.....	287
及成员方法	236		
10.8.2 JTextArea 组件的构造方法		第 13 章 多线程	288
及成员方法	236	13.1 Java 中的多线程实现技术	288
10.8.3 事件处理	237	13.1.1 线程的生命周期	288
10.8.4 应用举例	238	13.1.2 Thread 类的方法.....	290
习题 10.....	239	13.1.3 通过继承 Thread 类方式	
		创建线程	292
		13.1.4 通过实现 Runnable 接口方式	
第 11 章 高级组件 GUI 设计	241		

创建线程	293	15.2.2 Socket 类与 ServerSocket 类	331
13.2 多线程的管理	295	15.2.3 流式 Socket 通信的示例程序	333
13.2.1 线程调度	295	15.2.4 URL 通信与 Socket 通信的区别	336
13.2.2 线程优先级	295	15.3 UDP 通信	337
13.2.3 线程同步	296	15.3.1 UDP 通信机制	337
13.2.4 线程组	299	15.3.2 DatagramSocket 类	338
习题 13	299	15.3.3 DatagramPacket 类	338
第 14 章 输入与输出	301	15.3.4 UDP 通信示例程序	338
14.1 基本输入/输出流类	301	习题 15	341
14.1.1 InputStream 类	301	第 16 章 JDBC 连接数据库	343
14.1.2 OutputStream 类	305	16.1 关系数据库与 SQL 语言	343
14.1.3 Reader 类和 Writer 类	309	16.1.1 关系数据库的基本概念	343
14.2 文件的输入/输出	312	16.1.2 数据定义语言	344
14.2.1 File 类	312	16.1.3 数据操纵语言	345
14.2.2 FileInputStream 类和 FileOutputStream 类	315	16.1.4 数据查询语言	345
14.2.3 字节文件输入/输出流的读/写	316	16.2 使用 JDBC 连接数据库	346
14.2.4 FileReader 类和 FileWriter 类	319	16.2.1 JDBC 结构	346
14.2.5 RandomAccessFile 类	320	16.2.2 四类 JDBC 驱动程序	347
习题 14	323	16.2.3 JDBC 编程要点	348
第 15 章 网络编程	325	16.2.4 常用的 JDBC 类与方法	348
15.1 URL 通信	325	16.2.5 安装 ODBC 驱动程序示例	352
15.1.1 URL 类	326	16.3 JDBC 编程实例	354
15.1.2 利用 URL 类访问网上资源 示例程序	327	16.3.1 创建数据表	354
15.1.3 使用 URLConnection 类访问 网上资源	329	16.3.2 向数据表中插入数据	355
15.2 Socket 通信	330	16.3.3 更新数据	356
15.2.1 Socket 的概念及通信机制	330	16.3.4 删除记录	358
		习题 16	359
		参考文献	360



第1章

Java 系统环境概述

Java 语言是美国加州 Sun Microsystem 公司于 1995 年正式推出的纯面向对象(object-oriented, 简称 OO)的程序设计语言。由于它很好地解决了网络编程语言中的诸多问题, 因此一经推出, 便受到了计算机界的普遍欢迎和接受, 并得到了广泛的应用和发展, 成为目前网络时代最为流行的程序设计语言。

面向对象的编程语言使程序能够比较直观地反映客观世界的本来面目, 并且使软件开发人员能够运用人类认识事物所采用的一般思维方法进行软件开发, 是当今计算机领域中软件开发和应用的主流技术。所有面向对象的程序设计语言都支持对象、类、消息、封装、继承、多态等诸多概念, 而这些概念是人们在进行软件开发、程序设计的过程中逐渐提出来的。因此, 要弄清面向对象及其相关概念, 就得先从程序设计语言的发展谈起。

1.1 编程语言的发展

自从 1946 年第一台电子计算机问世以来, 人们一直在探索着自然语言与计算机语言之间的映射问题。我们知道, 人类的任何思维活动都是借助于人们所熟悉的某种自然语言进行的。若希望借助计算机完成人类的一种思维活动, 就需要把用自然语言表达的东西转换成计算机能够理解和执行的形式语言, 这便是编程语言或程序设计语言。毫无疑问, 电子计算机毕竟是一种机器, 它能够理解和执行的编程语言和自然语言之间存在着较大的差距, 这种差距被人们称作“语言的鸿沟”。这一鸿沟虽不可彻底消除, 但可以使其逐渐变窄。事实上, 从计算机问世至今, 各种编程语言的发展变迁, 其目的就是为了缩小这一鸿沟。图 1.1 引自参考文献[3], 笔者稍作修改, 该图展示了从机器语言发展到面向对象的语言使“语言的鸿沟”变窄的情形。

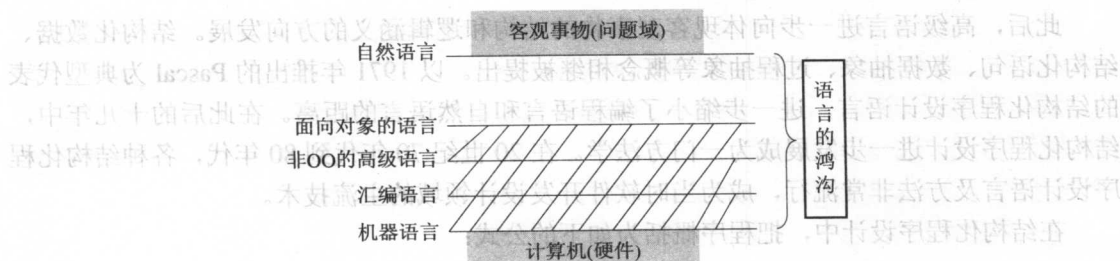


图 1.1 编程语言的发展使“语言的鸿沟”变窄



1.1.1 机器语言

电子计算机是一种机器，这种机器主要由电子元器件构成。对于电子元器件来说，最容易表达的是电流的通/断，或电位的高/低两种状态。因此，在电子计算机问世之初，人们首先想到的是用“0”和“1”两种符号来代表电路的通和断两种状态，这便是最早的编程语言——机器语言。

机器语言是计算机能够理解并直接执行的惟一语言。整个语言只包含“0”和“1”两种符号。用机器语言编写的程序，无论是它的指令、数据还是其存储地址，都是由二进制的“0”和“1”组成的。这种语言离计算机最近，机器能够直接执行它。然而，由“0”和“1”组成的二进制串没有丝毫的形象意义，因此，它离人类的思维最远，“语言的鸿沟”最宽。所以，用机器语言编写程序的效率最低，并且在编写程序时很容易发生错误。

1.1.2 汇编语言

为了克服机器语言的缺陷，人们设想用一些易于理解和记忆的符号来代替二进制码，这便是汇编语言。由于汇编语言用符号构成程序，而这些符号表示指令、数据、寄存器、地址等物理概念，因而，使用汇编语言编程在适合人类形象思维的道路上前进了一步。但是，使用汇编语言编写程序时，编程人员依然需要考虑寄存器等大量的机器细节，即汇编语言仍然是一种与具体机器硬件有关的语言，是一种面向机器的语言，因此，人们也把它称为符号化的机器语言。

1.1.3 高级语言

由于机器语言和汇编语言都离不开具体的机器指令系统，用它们编程时要求程序员必须熟悉所用计算机的硬件特性，因而，用它们编写程序的技术复杂，效率不高，且可维护性和可移植性都很差。为了从根本上摆脱语言对机器的依附，人们经过多年的潜心研究，终于在 1956 年推出了一种与具体机器指令系统无关、表达方式接近自然语言的计算机语言——FORTRAN 语言。在 FORTRAN 语言程序中，采用了具有一定涵义的数据命名和人们容易理解的执行语句，屏蔽了机器细节，使得人们在书写和阅读程序时可以联系到程序所描述的具体事物。所以，人们把这种“与具体机器指令系统无关，表达方式接近自然语言”的计算机语言称为高级语言。高级语言的出现是编程语言发展史上的一大进步，它缩小了编程语言与自然语言之间的鸿沟。

此后，高级语言进一步向体现客观事物的结构和逻辑涵义的方向发展。结构化数据、结构化语句、数据抽象、过程抽象等概念相继被提出。以 1971 年推出的 Pascal 为典型代表的结构化程序设计语言，进一步缩小了编程语言和自然语言的距离。在此后的十几年中，结构化程序设计进一步发展成为一门方法学。在 20 世纪 70 年代到 80 年代，各种结构化程序设计语言及方法非常流行，成为当时软件开发设计领域的主流技术。

在结构化程序设计中，把程序概括为如下的公式：

$$\text{程序} = \text{数据结构} + \text{算法}$$

其中，数据结构是指利用计算机的离散逻辑来量化表达需要解决的问题，而算法则研究如

何高效而快捷地组织解决问题的具体过程。可见，以结构化程序设计为代表的高级语言是一种面向数据/过程的程序设计语言，人们把这类语言也称为面向过程的语言。

面向过程的语言可以精确地用计算机所理解的逻辑来描述和表达待解问题的具体解决过程。然而，它把数据和过程分离为相互独立的实体，使程序中的数据和操作不能有效地组织成与问题域中的具体事物相对应的程序成分，所以它很难把一个具有多种相互关系的复杂事物表述清楚。程序员在编写算法时，必须时刻考虑所要处理问题的数据结构，如果数据结构发生了轻微的变化，那么对处理这些数据的算法也要做出相应的修改，甚至完全重写，否则这个算法就不可再用。因而，用这种程序设计方法编写的软件，其重用性较差。为了较好地解决软件的重用性问题，使数据与程序始终保持相容，人们又提出了面向对象的程序设计方法。

1.1.4 面向对象的语言

面向对象的编程语言(Object-Oriented Programming Language, OOPL)的设计出发点是为了能更直接地描述问题域中客观存在的事物(即对象)以及它们之间的关系。面向对象技术追求的是软件系统对现实世界的直接模拟，是将现实世界中的事物直接映射到软件系统的解空间。它希望用户最大程度地利用软件系统，花费少量的编程时间来解决需要解决的问题。

在面向对象的程序设计语言中，可以把程序描述为如下的公式：

程序 = 对象 + 消息

面向对象的语言对现实世界的直接模拟体现在下面几个方面：

(1) 对象(object)。只要我们仔细研究程序设计所面对的问题域——客观世界，就可以看到：客观世界是由一些具体的事物构成的，每个事物都具有自己的一组静态特征(属性)和一组动态特征(行为)。例如，一辆汽车有颜色、型号、马力、生产厂家等静态特征，又具有行驶、转弯、停车等动态特征。要把客观世界的这一事实映射到面向对象的程序设计语言中，则需把问题域中的事物抽象成对象，用一组数据描述该对象的静态特征(即属性，在 Java 中称之为数据成员)，用一组方法来刻画该对象的动态特征(即行为)。

(2) 类(class)。客观世界中的事物既具有特殊性又具有共同性。人类认识客观世界的基本方法之一就是事物进行分类，即根据事物的共同性把事物归结为某些类。考虑一下所有的汽车和一辆汽车之间的关系就很容易理解这一点。OOPL 很自然地用类(class)来表示一组具有相同属性和方法的对象。

(3) 继承(inheritance)。在同一类事物中，每个事物既具有同类的共同性，又具有自己的特殊性。OOPL 用父类与子类的概念来描述这一事实。在父类中描述事物的共性，通过父类派生(derive)子类的机制来体现事物的个性。考虑同类事物中每个事物的特殊性时，可由这个父类派生子类，子类可以继承父类的共同性，又具有自己的特殊性。

(4) 封装(encapsulation)。客观世界中的事物是一个独立的整体，它的许多内部实现细节是外部所不关心的。例如，对于一个只管开车的驾驶员来说，他可能根本不知道他所驾驶的这辆汽车内部用了多少根螺钉或几米导线，以及它们是怎样组装的。OOPL 用封装机制把对象的属性和方法结合为一个整体，并且屏蔽了对象的内部细节。

(5) 关联(association)。客观世界中的一个事物可能与其他事物之间存在某种行为上的联系。例如，一辆行驶中的汽车遇到红色信号灯时要刹车停止，OOPL 便通过消息连接来表示