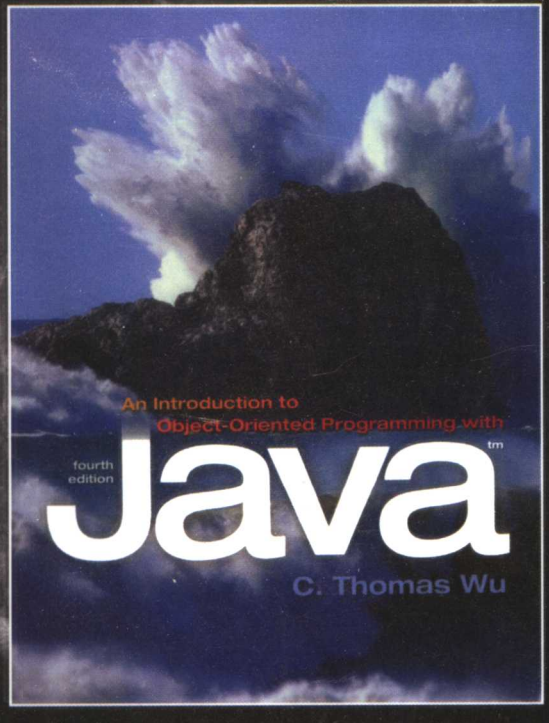
HZ BOOKS
Mc Graw Hill
计 算 机 科 学 丛 书
原书第4版

面向对象程序设计教程
（Java版）

（美）C. Thomas Wu 著　马素霞 齐林海 王素琴 谢 萍 等译



An Introduction to Object-Oriented Programming with Java
Fourth Edition

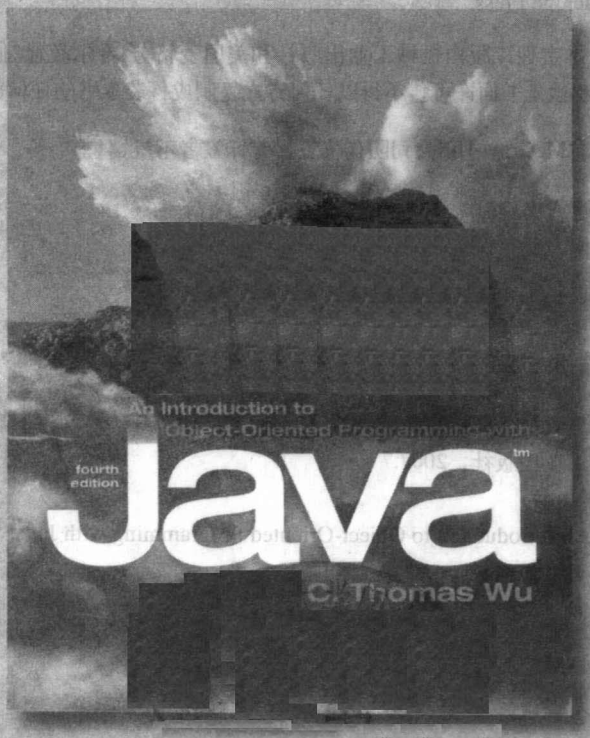
机械工业出版社
China Machine Press

计 算 机 科 学 丛 书

原书第4版

面向对象程序设计教程
(Java版)

(美) C. Thomas Wu 著　马素霞 齐林海 王素琴 谢 萍 等译



An Introduction to Object-Oriented Programming with Java

Fourth Edition

机械工业出版社
China Machine Press

本书全面详细地讲解面向对象程序设计的基本思想和编程方法，主要内容包括Java语言的介绍和使用、选择语句、重复语句、自定义类、异常与断言、字符与字符串、数组、排序和查找、文件输入与输出、继承与多态、图形用户界面与事件驱动的程序设计、递归算法。本书以面向对象的程序设计方法贯穿始终，在讲解的过程中使用了很多形象的比喻，容易学习，而不让人感到枯燥，且几乎每章都结合实例开发，基础性和实用性并重。读完本书后，读者不仅能掌握Java语言，而且能够掌握一些常见的实际问题的解决方法。

本书适合作为高等院校“Java语言程序设计”课程的教材，也适合初学者作为自学教材。

C. Thomas Wu：An Introduction to Object-Oriented Programming with Java, Fourth Edition (ISBN 0-07-294652-0).

本书版权登记号：图字：01-2006-2687

图书在版编目（CIP）数据

面向对象程序设计教程：Java版（原书第4版）/（美）武（Wu, C. T.）著；马素霞等译. -北京：机械工业出版社，2007.7

（计算机科学丛书）

书名原文：An Introduction to Object-Oriented Programming with Java, Fourth Edition

ISBN 978-7-111-21316-1

Ⅰ.面…　Ⅱ.①武…　②马…　Ⅲ.Java语言-程序设计-教材　Ⅳ.TP312

中国版本图书馆CIP数据核字（2007）第054310号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）

责任编辑：王　璐

北京京北制版厂印刷·新华书店北京发行所发行

2007年7月第1版第1次印刷

184mm×260mm · 36.5印张

定价：59.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书电话（010）68326294

译 者 序

面向对象技术是程序设计方法的一场革命。与传统的结构化技术相比，面向对象技术在解决大型复杂的问题及网络计算方面更有优势。上个世纪末以来，面向对象技术逐渐成为计算机应用领域的主流技术。诞生于1995年的Java语言是典型的面向对象程序设计语言，由于Java具有跨平台特性、安全机制出众、高可靠性及内嵌的网络支持，它已成为编写网络应用程序的首选工具之一。目前，大多数学校都在本科阶段开设Java程序设计课程，有些学校还将Java作为第一门语言课程。

本书以面向对象的程序设计方法贯穿始终，全面详细地讲解了面向对象程序设计的基本思想和编程方法，在讲解的过程中运用了很多形象的比喻，读者在学习过程中不会感到枯燥乏味，且容易理解。书中还结合大量实例进行讲解，基础性和实用性并重。通过阅读本书，读者不仅能掌握Java语言，而且能够掌握一些常见实际问题的解决方法，因此，本书是很好的Java语言及面向对象程序设计的入门书籍，适合作为大专院校“Java语言程序设计”的课程教材，也适合初学者作为自学教材。

本书的突出特点是讲解细致、深入，在讲解理论的同时，注重软件开发实践，几乎在每一章的最后都有实例开发一节。全书共分16章，读者在学习时可以按照顺序阅读，建议按顺序阅读第1章到第7章的内容，其他章节可以根据具体情况有选择性地阅读，特别是标有“*”的章节，其中第0、14、15章和3.11节、6.11节及8.6节的断言部分可作为选学内容。

本书的翻译工作主要由华北电力大学计算机系的教师完成，主要翻译人员包括马素霞(第0~1、12~15章)、齐林海(第2~5章)、谢萍 (第6~8章) 、王素琴(第9~11章)。另外，华北电力大学计算机系的钱力、王强、林天华、刘金晓、王萍萍、王飞翔、齐明也参与了本书的翻译工作。在全体翻译人员的通力协作下，本书得以顺利完成，在此我对他们表示感谢，最后本人对全书做了仔细审核与修改，尽力使译稿准确、易懂。

由于本书翻译时间有限，译稿难免存在错误和疏漏，欢迎读者批评指正。

马素霞

2007年4月

前　言

本书以面向对象程序设计方法为主线，强调适当的面向对象设计实践。学生首先学习如何使用对象，然后学习设计类。在第4版中，本书将以轻松的方式教授学生如何设计自己的类，自定义类的内容将分为两章讲解。

第4版的主要变化

在介绍本书的特点之前，首先简要介绍一下第4版的变化。第4版对自定义类及Java 5.0(Java 2 SDK 1.5)的讨论更容易理解、更深入，对GUI具有较少的依赖性。

1. 轻松地介绍自定义类

在学习面向对象程序设计时，最令学生感到困难的问题之一就是创建自定义类。大多数学生觉得使用标准类中的对象非常简单，然而当他们试着定义自己的类时，通常会陷入困惑。在第3版中，我们用一章的篇幅介绍了与自定义类相关的所有主题，而第4版则将自定义类分为两章进行讲解。我们将在第4章中使用新例子介绍自定义类的基本知识，轻松的讨论会使学生更容易理解。

2. 更深入地讨论自定义类

第7章将更深入地介绍自定义类，包括方法重载、保留字this的使用、类方法和变量，这些内容是大多数学生难以掌握的。第4版在讲解完选择和重复控制方面的传统主题后，到第7章再介绍这些高级主题，会使学生更容易理解。另外，通过使用控制结构，可以使用更详细、更实际的例子来介绍OO特性，而这些例子非常清晰地显示出对这些特性的需求。

3. Java 5.0

最新的Java 2 SDK增加了很多内容。在第4版中，我们对其中一些新增加内容作了介绍，并改进了CS1的教学。首先介绍的是`Scanner`类。在SDK 1.5之前，都是通过使用`BufferedReader`对象来完成标准的输入例程。由于`BufferedReader`抛出异常，我们必须在讲解标准的输入之前讨论异常处理，或者提供某种自定义输入类来隐藏异常处理。而使用新的`Scanner`类，我们可以讲解更为简单的输入例程，这些简单的输入例程不需要任何异常处理，`Scanner`类将在第3章介绍。接下来介绍的是`Formatter`类，这个类提供了格式化技术，它与C程序设计语言所支持的格式化技术几乎一样，我们将在第6章讲解`Formatter`类。

4. 不依赖于GUI知识

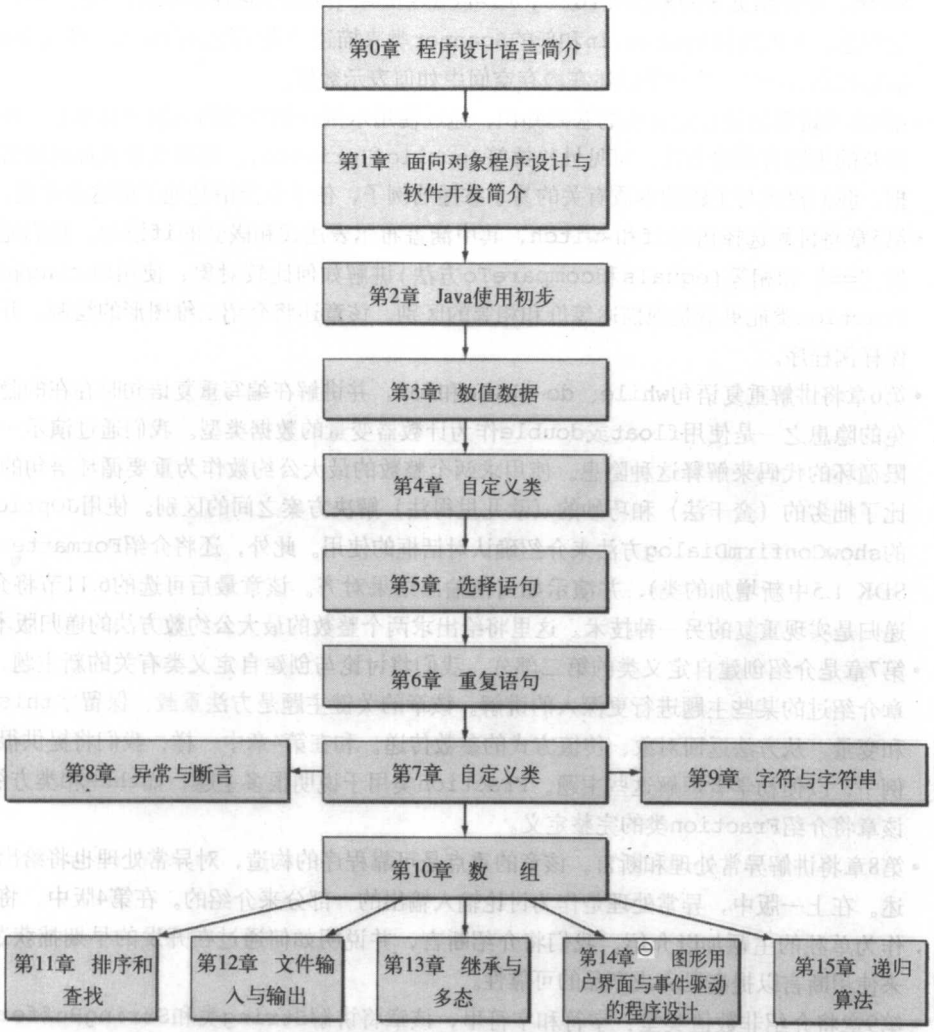
在第3版中，我们在第7章介绍了基本的GUI及事件驱动的程序设计，在第14章介绍高级GUI。在后面章节中，某些例子和实例开发需要GUI的知识。我们将这些知识合并为一章，并移到第14章，因此提供了一定的灵活性。在CS1课程中不讲授GUI的教师照样能够使用第4版。讲授GUI的教师可以有选择性地介绍GUI的主题，且可以从第2章以后就开始介绍。

本书的组织

本书共16章（即第0章~第15章），这些内容对于一学期的课程而言足够了。这些内容可以按顺序讲解，也可以不按顺序讲解。我们首先介绍各章之间的关系，然后对每一章的内容进行概括。

各章之间的关系

大部分章节应按顺序阅读，但也可以有一些变化，特别是对标有“*”的章节。第0、14、15章和3.11节、6.12节是可选的，8.6节也可以作为可选内容。下面是一个简单的各章之间的关系图。



各章内容概述

下面对每一章的内容进行简要描述。

- 第0章是“可选”章节。我们将介绍程序设计语言的背景知识。如果希望从面向对象程序设计的概念开始讲解，则可以跳过这一章，或者将该章留作课外阅读内容。
- 第1章将给出面向对象程序设计的基础知识。我们将描述面向对象程序设计的主要成分，并用UML图形符号说明每一个概念。
- 第2章将介绍Java程序设计的基础及编辑、编译和运行程序的过程。从该章介绍的第一个样例程序开始，就强调面向对象的概念。通过介绍标准类`String`、`JOptionPane`、`Date`和

⊖ 一些例子使用数组，但数组的使用并不是必需的。也可以对这些例子进行修改，使其不使用数组。该章前面部分的很多主题在第2章以后就可以尽早介绍。

SimpleDateFormat，将强化对象声明、创建及使用的概念。此外，通过使用标准类，学生可以立即开始编写实际的程序。

- 第3章将介绍变量、常量和处理数值数据的表达式。我们将讲解java.lang包中的标准类Math，并介绍更多的标准类(GregorianCalendar和DecimalFormat)来继续强化面向对象的概念。我们使用System.in和新的Scanner类来描述并说明控制台输入，使用System.out描述控制台输出。3.11节描述在内存空间中如何表示数值。
- 第4章将讲解创建自定义类的基础知识，通过使用示例介绍基础的知识使该章易于理解。该章涉及的主题有构造方法、可见性修饰符(public和private)、局部变量及如何给方法传递数据。我们提供与主题的本质有关的易于掌握的例子，使学生能清楚地了解这些主题。
- 第5章将讲解选择语句if和switch，其中涵盖布尔表达式和嵌套的if语句。我们通过使用等价“==”和相等(equals和compareTo方法)讲解如何比较对象；使用String和自定义的Fraction类能更清楚地描述等价和相等的区别。该章还将介绍二维图形的绘制，开发一个屏保样例程序。
- 第6章将讲解重复语句while、do-while和for，并讲解在编写重复语句时存在的隐患。要避免的隐患之一是使用float或double作为计数器变量的数据类型。我们通过演示一段导致无限循环的代码来解释这种隐患。使用求两个整数的最大公约数作为重要循环语句的例子，对比了拙劣的（蛮干法）和巧妙的（欧几里得法）解决方案之间的区别。使用JOptionPane类的showConfirmDialog方法来介绍确认对话框的使用。此外，还将介绍Formatter类(Java 2 SDK 1.5中新增加的类)，并演示如何使输出结果对齐。该章最后可选的6.11节将介绍递归，递归是实现重复的另一种技术。这里将给出求两个整数的最大公约数方法的递归版本。
- 第7章是介绍创建自定义类的第二部分。我们将讨论与创建自定义类有关的新主题，也对第4章介绍过的某些主题进行更深入的讲解。该章的关键主题是方法重载、保留字this、类方法和变量、从方法返回对象、传值方式的参数传递。和在第4章中一样，我们将提供很多易懂的例子，以便初学者理解这些主题。Fraction类用于说明很多主题，如this和类方法的使用。该章将介绍Fraction类的完整定义。
- 第8章将讲解异常处理和断言。该章的重点是可靠程序的构造，对异常处理也将给出详细的描述。在上一版中，异常处理是作为讨论输入输出的一部分来介绍的。在第4版中，将异常处理作为单独的主题加以介绍。我们将介绍断言，并说明如何通过在开发的早期捕获逻辑错误，来使用断言以提高已完成产品的可靠性。
- 第9章将介绍非数值类型：字符和字符串，该章将讲解String类和StringBuffer类。字符串处理的一个重要应用是模式匹配。该章描述模式匹配和正则表达式，介绍Pattern和Matcher类，以及如何将这两个类应用于模式匹配。
- 第10章将讲授数组，包括基本数据类型的数组和对象数组。在Java中，数组是引用数据类型，我们将说明如何将数组传递给方法。我们会描述如何处理二维数组，在Java中，二维数组实际上就是数组的数组。接下来将介绍列表和映射，以更通用和灵活的方式来维护数据集合。在样例程序中，将介绍java.util包中的ArrayList和HashMap类的使用。另外，还将介绍如何用映射类TreeMap来实现在第9章的样例开发程序中所使用的WorldList辅助类。
- 第11章将介绍查找和排序算法，涉及复杂性为N^2和$N \log_2 N$的排序算法。根据学生的基础，查找和排序算法的数学分析可以省略。
- 第12章将讲解文件I/O，介绍诸如File和JFileChooser等标准类，其中涵盖文件I/O的所有类型，从低级的字节I/O到高级的对象I/O。我们还将介绍如何使用文件I/O技术来实现第8章和

第9章样例开发程序中的辅助类——Dorm和FileManager类。

- 第13章将讨论继承和多态，以及如何在程序设计中有效地使用它们。针对成员可访问性和构造方法，讲解了继承的作用，以及抽象类和抽象方法的作用。
- 第14章将介绍GUI和事件驱动的程序设计。该章只涉及基于Swing的GUI组件，包括JButton、JLabel、ImageIcon、JTextField、JTextArea和与菜单有关的类。我们将描述嵌套面板和布局管理器的有效使用，在样例程序中将描述和说明鼠标事件的处理。不讲授GUI的教师可以跳过本章的所有内容，讲授GUI的教师可以从第2章以后尽早地介绍该章的开始部分。
- 第15章将讲解递归。由于我们想展示适合使用递归的例子，所以不包括任何递归算法(除了那些用于解释的例子以外)，这些算法实际上都应该编写为非递归的。

本书的特点

问题解决

实例开发程序

第2章到第13章将演示以下重要问题的解决步骤：

- 问题陈述
- 总体计划
- 设计
- 编码
- 测试

开发练习给学生提供进行实际的增量设计的机会。

带有“Bad Version”字样的代码不仅表示学生的常犯错误，而且表明这是糟糕的设计。其后会给出正确的解决方案，使学生理解为什么第一次尝试是不正确的。

面向对象方法

本书作者采用面向对象程序设计的完全沉浸式方法，从一开始就强调正确的面向对象设计实践和Java语言的使用。因此，不仅教给学生如何成长为面向对象的程序员，而且还用贯穿全书的例子来说明如何成长为程序员。

- 使用大量图来说明关键理念。
- 例子代码的编写符合良好的面向对象实践。
- 从第7章的样例代码开始使用Javadoc注释，向学生教授Java代码所使用的标准文档技术。

输入／输出

对于用户界面，本书使用最新的标准Java库，向学生介绍基于控制台的方法和图形化方法，每种方法都用多个例子加以说明。

- 在很多例子中，输入和输出都采用JOptionPane类。
- System.out用于讲解控制台输出。
- Scanner类用于向学生讲授基于控制台的输入。

教学法

设计指导 提供良好程序设计的建议。

有益的提示 给学生以提示，帮助他们记住重要概念。

建议 在学生有效的程序设计方面给学生提供建议。

你可能想知道 给学生提供有趣的信息点。

自我检测 练习在每节的最后，帮助学生测试他们对主题的理解。

教师和学生补充资料

本书的在线学习中心网址是www.mhhe.com/wu。

教师的补充资料[⊖]

- 完整的PowerPoint资料，包括讲义和图。
- 本书练习的完整答案。
- 例子库(Example Bank)——增加的例子（可以通过主题搜索）在网站的“bank”栏目中提供。
- 作业管理员／测试库(Homework Manager/Test Bank)——概念回顾问题存储在这个电子问题库中，可以指定其中的问题为考试题或家庭作业。
- 与本书配套的在线实验室(Online lab)可以用于封闭实验、开放实验，或者用于布置程序设计项目。

学生的补充资料

- 在大多数流行编译器上如何启动和运行方面，Compiler How Tos提供了指导，来帮助学生使用IDE。
- 交互测验(Interactive Quizzes)允许学生测试对所学知识的掌握程度，并立即得到反馈。
- 书中所有例子程序的源代码(Source code)。
- 快速检查练习的答案(Answer)。
- 关键术语的术语表(Glossary)。
- 与计算机科学有关的新闻链接。
- 其他主题(Additional Topic)，如swing方面的更多信息及数据结构方面的介绍。

致谢

对以下评阅者所给予的评论、建议和鼓励表示诚挚的感谢！

Wu Focus Group—Jackson Hole, WY
Elizabeth Adams, *James Madison University*
GianMario Besana, *Depaul University*
Michael Buckley, *State University of New York, Buffalo*
James Cross, *Auburn University*
Priscilla Dodds, *Georgia Perimeter College*
Christopher Eliot, *University of Massachusetts-Amherst*
Joanne Houlahan, *John Hopkins University*
Len Myers, *California Polytechnic State University, San Luis Obispo*
Hal Perkins, *University of Washington*
William Shea, *Kansas State University*
Marge Skubic, *University of Missouri, Columbia*
Bill Sverdlik, *Eastern Michigan University*
Suzanne Westbrook, *University of Arizona*

⊖ 本教辅只提供给采用本书作为教材的教师，请需要者与麦格劳-希尔公司北京代表处联系。——编者注

评阅者

Ajith, Abraham, *Oklahoma State University*
Elizabeth Adams, *James Madison University*
David L. Atkins, *University of Oregon*
GianMario Besana, *DePaul University*
Robert P. Burton, *Brigham Young University*
Michael Buckley, *State University of New York, Buffalo*
Rama Chakrapani, *Tennessee Technological University*
Teresa Cole, *Boise State University*
James Cross, *Auburn University*
Priscilla Dodds, *Georgia Perimeter College*
Kossi Delali Edoh, *Montclair State University*
Christopher Eliot, *University of Massachusetts-Amherst*
Michael Floeser, *Rochester Institute of Technology*
Joanne Houlahan, *John Hopkins University*
Michael N. Huhns, *University of South Carolina*
Eliot Jacobson, *University of California, Santa Barbara*
Martin Kendall, *Montgomery Community College*
Mike Litman, *Western Illinois University*
Len Myers, *California Polytechnic State University, San Luis Obispo*
Jun Ni, *University of Iowa*
Robert Noonan, *College of William and Mary*
Jason S. O'Neal, *Mississippi College*
Hal Perkins, *University of Washington*
Gerald Ross, *Lane Community College*
William Shea, *Kansas State University*
Jason John Schwarz, *North Carolina State University*
Marge Skubic, *University of Missouri, Columbia*
Bill Sverdlik, *Eastern Michigan University*
Peter Stanchev, *Kettering University*
Krishnaprasad Thirunarayan, *Wright State University*
David Vineyard, *Kettering University*
Suzanne Westbrook, *University of Arizona*
Melissa Wiggins, *Mississippi College*
Zhiguang Xu, *Valdosta State University*

我的故事

2001年9月，出于个人原因我改名了，由Thomas Wu教授改成现在的Thomas Otani教授。为了保持连续性，并且不使人们感到困惑，我继续用以前的名字出版本书。对我改名感兴趣的人，可以访问我的网站www.drcaffeine.com。

目　录

第0章 程序设计语言简介

0.1 程序设计语言

程序设计语言大体上可划分为三级：机器语言、汇编语言和高级语言。机器语言是CPU唯一理解的程序设计语言，每种CPU都有自己的机器语言。机器语言指令是二进制编码，一条机器指令可以将内存中的内容传到CPU寄存器，或者将两个寄存器中的数据相加。因此，我们必须提供很多机器语言指令，才能完成诸如计算20个数据的平均值这样的简单任务。比机器语言更高一级的计算机语言是汇编语言(assembly language)，允许“更高级别”的符号程序设计。汇编语言允许程序员使用符号操作码写程序，而不是写成二进制的位序列。由于CPU不能识别用汇编语言编写的程序，我们使用汇编程序(assembler)将汇编语言编写的程序翻译成机器语言程序。与用机器语言编写程序相比，用汇编语言编写程序更快，但编写复杂的程序时，速度也相当受限。

高级语言的开发使得程序员编写程序比用汇编语言更快。例如，FORTRAN(FORmula TRANslator)是以数学计算为目的的程序设计语言，允许程序员这样直接表达数学公式：`X=(Y+Z)/2`。

COBOL(COmmon Business-Oriented Language)是以商业数据处理应用为目的的程序设计语言。FORTRAN和COBOL分别于20世纪50年代末及60年代初开发，目前仍在使用。BASIC是专门为学生学习和使用而开发的简单语言，是第一个在微机上使用的高级语言。另一个著名的高级语言是Pascal，被设计为教学语言。由于CPU不能识别用高级语言编写的程序，我们必须使用编译器(compiler)将其翻译成汇编语言程序。

20世纪70年代初，在AT&T Bell实验室开发出了C程序设计语言。20世纪80年代初，继C之后又开发出了C++程序设计语言，在C++中增加了对面向对象程序设计的支持。面向对象程序设计是目前获得更广泛接受的程序设计风格，虽然面向对象程序设计的概念很早就有人提出过，但其重要性直到20世纪80年代初才被人们广泛认识到。我们在本书中使用的程序设计语言Java是最新的面向对象的程序设计语言，由Sun Microsystems公司开发。

0.2 Java

Java是一种新的面向对象语言，受到了工业界及学术界的广泛关注。Java是James Gosling和他的团队在位于美国加利福尼亚州的Sun Microsystems开发出来的，是基于C和C++开发的。最初它的名字是Oak，以Gosling办公室外面的橡树命名，但这个名字已经被使用过了，所以团队将其改名为Java。

Java通常被描述为Web程序设计语言，因为可以用Java 编写小程序(applet)，在Web浏览器中运行。也就是说，需要Web浏览器来运行Java小程序。Java小程序允许在因特网上更加动态和灵活地分发信息，这种特性本身使得Java成为了有吸引力的语言。然而，在Java中，我们并不限于编写Java小程序。我们还能够编写Java应用程序。Java应用程序(application)是完全独立的程序，运行时不需要Web浏览器。Java应用程序类似于我们使用其他程序设计语言编写的程序。在本书中，我们的重点是Java应用程序，因为我们的目的是教授面向对象程序设计的基础知识，这些基础知识可以应用于其他所有的面向对象程序设计语言。

本教材选择Java的主要原因是其清晰的设计。Java语言的设计者采用了简约方法，只包括绝对必要的特性，而去除了他们认为多余的特性。这种简约方法使得Java语言比其他面向对象程序设计语言更容易学习。Java是教授面向对象程序设计基础的理想工具。

第1章 面向对象程序设计与软件开发简介

学习目标

- 面向对象程序设计中的基本元素。
- 区分类和对象。
- 区分类方法和实例方法。
- 区分类数据值和实例数据值。
- 用面向对象程序设计中的类、对象及其他元素的图标画程序图。
- 面向对象程序设计中继承的重要性。
- 解释软件生命周期的各个阶段。

简介

在开始编写程序之前，有必要先介绍与面向对象程序设计(object-oriented programming，OOP)有关的几个基本概念。本书选择面向对象程序设计讲授程序设计的方法，本章的目的是使读者对面向对象程序设计有一个感性印象，并介绍面向对象程序设计的基本概念。在后面的学习中，可能需要参考本章的内容。

建议

对于那些具有某些编程经验的人，无论是具有面向对象的编程经验，还是非面向对象的编程经验，都会发现Java与其他的高级程序设计语言之间具有很多相似之处。这种相似会加快学习进程，但在很多情况下，表面上看起来相似，结果却完全不同。所以，请不要贸然地断定相似性。

本章的另外一个目的是介绍软件开发过程。为了能够编写程序，只具有面向对象程序组成元素方面的知识是不够的，一定要学习开发程序的过程。本章将简要介绍软件开发过程。

1.1 类和对象

在面向对象程序设计中，最重要的概念是类和对象。从广义上讲，一个对象(object)就是我们能够想象的一件事物，可以是有形的，也可以是无形的。以面向对象风格编写的程序由交互的对象组成。跟踪大学住校学生的程序，可能有很多Student、Room及Floor对象；而跟踪顾客及自行车商店投资的程序，可能有Customer、Bicycle及很多其他类型的对象。对象由数据和作用在数据上的操作组成。例如，Student对象的数据可能有姓名、性别、出生日期、家庭住址、电话号码及年龄，其操作可能是对这些数据值的赋值及更改。我们将使用图1-1所示的符号表示对象。本书使用基于工业标准——统一建模语言UML(Unified Modeling Language)的符号。我们将通过一些例子轻松地讲解UML的规则。

几乎所有重要的程序都有很多相同类型的对象。例如，在自行车商店程序中，我们可能看到很多Bicycle对象、Customer对象及其他对象。图1-2给出了名为Moto-1、Moto-2的两个Bicycle对象及一个名为John Java的Customer对象。

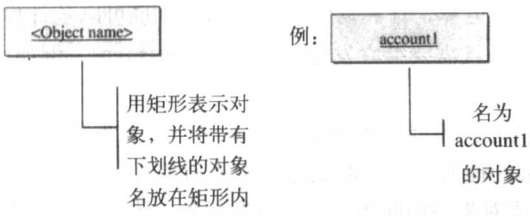


图1-1 对象的图形表示

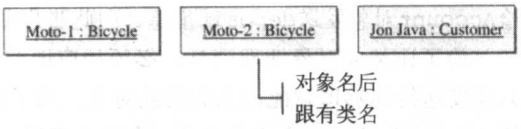


图1-2 名为Moto-1、Moto-2的两个Bicycle对象及名为John Java的一个Customer对象

可以在程序中编写创建对象的指令。为了让计算机创建对象，我们必须提供对象的定义，对象的定义称为类(class)。类是描述对象能够做什么及不能做什么的一种模型或模板。将对象称为类的实例(instance)。一个对象只能是一个类的实例，而一个类的实例属于这个类。例如，对象Moto-1和Moto-2是Bicycle类的实例。一旦定义了类，就可以根据程序的需要创建多个类对象。

提示

在创建类的实例(对象)之前，一定要先定义类。

本书使用图1-3来表示类。

1. 为Person类及两个Person对象Ms. Latte和Mr. Espresso画对象图。
2. 在创建对象之前必须定义什么？

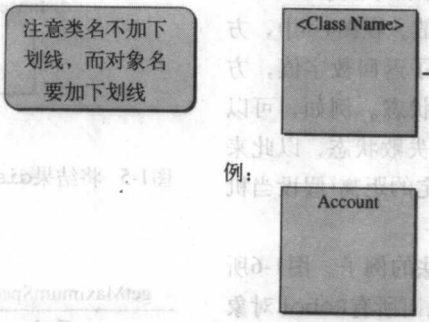


图1-3 类的图形化表示

刚接触面向对象程序设计语言的人往往搞不清楚类与对象之间的区别，为了帮助大家更好地理解这种区别，我们将类与对象比作木刻印版和用木刻印版印出的印刷品。木刻印版是一块木头，上面雕刻着印刷设计的图样，一旦有了木刻印版，想要多少印刷品就可以印多少。与此类似，一旦你拥有了类，就可以用这个类创建很多对象。另外，就像没有木刻印版就不能生产印刷品一样，不先定义类就不能创建对象。对19世纪日本艺术家Hiroshige的印刷样本感兴趣的读者可访问http://www.ibiblio.org/wm/paint/auth/hiroshige/。

另一个很形象的比喻是机器人工厂。工厂是类，从工厂中生产出来的机器人是此类的对象。为了制造出机器人(实例)，我们首先需要有工厂(类)。对移动机器人感兴趣的读者可访问http://www.ai.mit.edu/projects/mobile-robots/robots.html。

1.2 消息和方法

在编写面向对象程序时，我们必须首先定义类。当程序运行时，我们使用类和由类生成的对

象来完成任务。任务的范围可以是对两个数进行相加运算，计算助学贷款的利息，也可以是计算航天飞机的再入角。要命令类或对象执行某项任务，就需要给它发送一条消息(message)。例如，给Account对象发送deposit消息，向此账户中存款100美元。

为了让类或对象处理消息，必须相应地编写消息。不能简单地将消息发送给任意类或对象，只能发送给能够理解此消息的类或对象。为了能够处理所接收到的消息，类或对象必须拥有相应的方法(method)。一个方法就是一个指令序列，类或对象按照此指令序列执行任务。为类定义的方法称为类方法(class method)，为对象定义的方法称为实例方法(instance method)。

让我们首先看一个实例方法的例子。假设为Robot对象定义了一个称为walk的方法，此方法指示机器人行走一段指定的距离。只要定义了这个方法，我们就可以将消息walk与要行走的距离一同发送给Robot对象。传递给对象的值称为消息的参数(argument)。值得注意的是，发送给对象或类的消息名必须与方法名相同。消息的发送如图1-4所示。

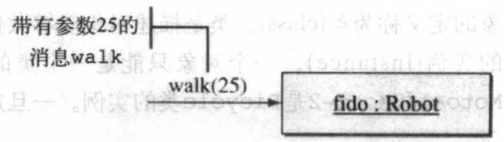


图1-4 给Robot对象发送消息walk

图1-4描述的是单向通信，即对象执行请求的操作(行走指定的距离)，但不对消息的发送者做出回应。在很多情况下，我们都需要进行双向通信，要求对象通过返回值来回应消息的发送者。例如，假设我们想知道机器人到离它最近的障碍物的距离，机器人的设计者就可以增加方法getObstacleDistance，返回期望的值。在图1-5中，方法返回值给消息的发送者。除了返回数字值，方法还能够传回所请求的操作的状态。例如，可以将方法walk定义为返回成功或失败状态，以此来表示机器人是否顺利走完了指定的距离(假设当机器人撞上障碍物时，就失败了)。

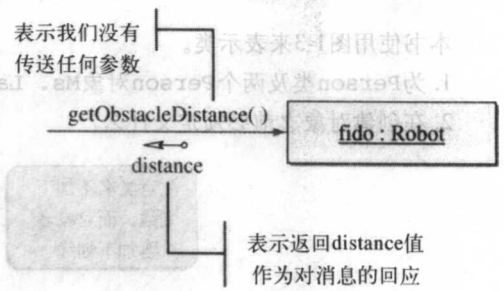


图1-5 将结果distance返回给消息的发送者

现在让我们来看一个类方法的例子。图1-6所示的方法getMaximumSpeed返回所有Robot对象的最大可能速度。方法getMaximumSpeed处理一个类的所有实例的集合信息，通常将这样的方法定义为类方法。所以我们为适合个体实例的任务定义实例方法，为适合所有实例的任务定义类方法。

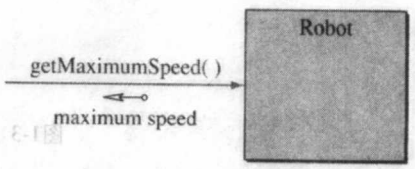


图1-6 类方法返回所有Robot对象的最大可能速度

建议

你是否犯过这样的错误：由于拨错了号码而在陌生人的电话应答机上留言？当机器说“你好！我现在不能接电话，你是否能够留言……”，你回答到“你好！是我呀！请立即到SP105。”结果是对方没有按照你说的去做，因为他不明白你说的是什么。

同样，你不能给类或对象发送消息，除非编写了处理这个消息的程序，也就是说一定要包含相应的方法。向类或对象发送消息，就可以使相应的方法执行。如果没有相应的方法，则什么都不会发生(实际上，将会出现一个错误)。

自我检测

1. Account对象具有实例方法deposit和withdraw，画出其对象图。
2. 方法getObstacleDistance是实例方法，还是类方法？

1.3 类数据值和实例数据值

假设Account对象的方法deposit给该对象的当前余额增加一定数量的金额，该对象将当前余额保存在哪里呢？记住，一个对象由数据值和方法组成。与定义类方法和实例方法类似，我们需要定义类数据值和实例数据值。例如，为Account对象定义实例数据值(instance data value) current balance 来记录当前余额。图1-7给出了具有数据值current balance的三个Account对象。注意，它们都具有相同的数据值current balance。相同类的所有对象都拥有相同的数据值集合。如图显示的那样，current balance的实际美元数量各不相同。Account对象的其他可能的实例数据值还有开户余额及帐号。

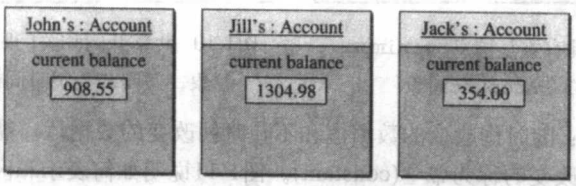


图1-7 具有相同的数据值current balance，但实际美元数量不同的3个Account对象

类数据值(class data value)用来表示被所有实例共享的信息，或表示所有实例的共同信息。例如，如果每一个账户都必须有最低余额限制（比方说100美元），我们就可以定义一个类数据值minimum balance。实例可以访问它所属的类的类数据值。图1-8显示如何表示类数据值。注意，类数据值加了下划线。由于类的对象加了下划线，类数据值对于此类的所有对象都是可访问的，所以我们也给类数据值加上下划线来表示这种关系。数据值也被称为数据成员(data members)，因为它们属于类或类的实例。

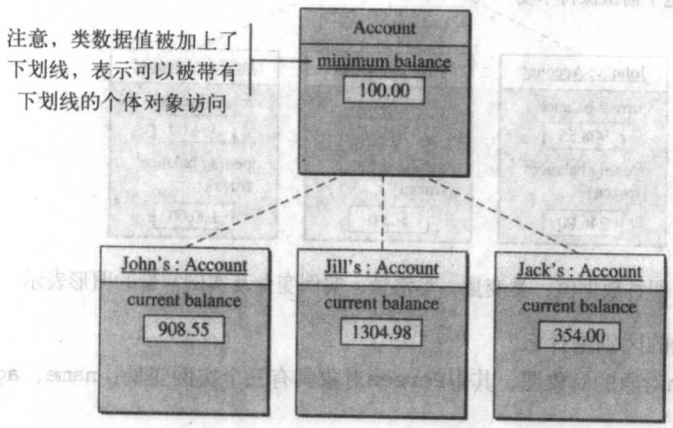


图1-8 作为类数据值存储的3个Account对象的共享信息(minimum balance = $100)

为了进一步说明类数据值的重要性，如果我们将minimum balance表示为实例数据值，会发生什么？图1-9显示三个Account对象，其current balance具有的美元数量不同，但minimum balance具有的美元数量相同。显然，minimum balance的重复存储是多余的，并且浪费空间。例如，如果银行将minimum balance提升到200美元，考虑一下会发生什么？如果有100个Account对象，则必须更新minimum balance的100个副本。通过将minimum balance定义为类数据值就可以避免这种情况。图1-10又举了一个例子，在此例中，校园中的所有自助餐厅的开门和关门时间相同。