外计算机科学经典教材

THOMSON

Java with Object-Oriented Programming

Java 面向对象程序设计

(美) Paul S. Wang 著

杜一民 赵小燕 译

清华大学出版社

Java 面向对象程序设计

(美) Paul S. Wang 著

杜一民 赵小燕 译

清华大学出版社

北京

Paul S. Wang
Java with Object-Oriented Programming

EISBN: 0-534-39276-8

981-240-702-2

北京市版权局著作权合同登记号　图字:01-2002-5742

图书在版编目(CIP)数据

Java 面向对象程序设计/(美)王保罗著；杜一民　赵小燕译. —北京：清华大学出版社，2003
书名原文：Java with Object-Oriented Programming
ISBN 7-302-06745-7
Ⅰ.J… Ⅱ.①王…②杜…③赵… Ⅲ.Java 语言—程序设计　Ⅳ.TP312
中国版本图书馆 CIP 数据核字(2003)第 046112 号

出 版 者：清华大学出版社　　**地　　址**：北京清华大学学研大厦
http://www.tup.com.cn　　**邮　　编**：100084
社 总 机：010-62770175　　**客户服务**：010-62776969

组稿编辑：曹康
文稿编辑：陈宗斌
封面设计：康博
版式设计：康博
印 刷 者：清华大学印刷厂
发 行 者：新华书店总店北京发行所
开　　本：185×260　**印张**：30.75　**字数**：787 千字
版　　次：2003 年 7 月第 1 版　2003 年 7 月第 1 次印刷
书　　号：ISBN 7-302-06745-7/TP・5028
印　　数：0001～5000
定　　价：56.00 元

前　言

自 20 世纪 90 年代以来，Java 已经逐渐发展成熟。其平台无关性、面向对象、联网功能和图形用户界面(GUI)以及线程支持使得 Java 成为许多应用程序理想的开发工具。本书将帮助您了解这些内容以及有关 Java 的其他重要方面，以便您可以充分地利用其功能。本书的合作 Web 站点为教师和学生提供了信息的更新和有用的资源。

Java 结合了面向对象设计(OOD)和面向对象编程(OOP)的概念和技术，因此，从本书中您能够同时了解 Java 语言和 OOP 技术。本书通过具有面向对象结构的实例详细演示了所包含的主题。目的是以一种简明且实际的方式提供对 Java 和 OOP 全面而详细的介绍。

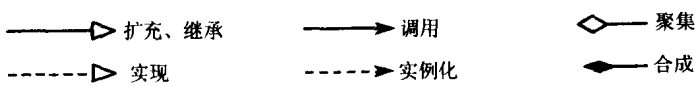
更新

Java 语言本身已经很成熟，但是支持 Java 主要功能的类库仍在不断发展。本书更新、扩充并重新组织了 1998 年出版的“Java with Object-Oriented Programming and World Wide Web Applications”一书，以便包括新内容和强调 OOP 和 OOD 技术。合作 Web 站点提供了信息的更新和实践经验，以便于您使用 Java 的最新版本。

OOP 和 OOD

OOP 是一种最有影响的现代编程范型；全面了解 OOP 是编程人员的当务之急。理论与实践相结合可使 OOP 原则具体化，而且可以提供形成 Java 构造的多种原因。许多完整的示例均演示不同的 Java 构造如何与个别的 OOP 技术相结合从而在实践中获得结果。Java 为 OOP 提供了良好的支持。但是，只使用 Java 构造并不会自动产生结构良好的面向对象程序。相反，如果不利用面向对象的观点来创建良好的设计方案，最终得到的程序将非常类似于用 Java 编写的面向过程的程序。更为糟糕的是这样的程序将很难产生类。

首先介绍 OO 概念和技术，它们与 Java 的有关内容密切相关。您将了解 OOP，随着本书的深入，您将实际使用这些概念和技术。第 13 章主要介绍 OOD 概念、方法和模式。所给出的主要示例的 OO 设计使用了统一建模语言(UML)类图，以增强面向对象的思考方式。UML 类图使用以下图形符号：



为了把它们和类相区别，使用符号<<interface>>来标记接口名称，并以斜体表示。

GUI

有了图形用户界面，最终用户就可以非常轻松地编写程序。但是 GUI 会增加程序的复杂程度。Java 基础类(Java Foundation Class，简写 JFC)提供了基本的和预定义的 GUI 组件或构件。Swing 程序包是 JFC 的重要组成部分。深入介绍 Swing 及其实际应用程序可使您初步了解 applet 的 GUI 程序(类似于独立的应用程序)。

事件驱动编程

GUI 要求采用可处理运行时事件的编程风格。而不采用指定的执行路径，事件驱动程序对无法预知其发生时间的外部事件作出反应。介绍事件驱动编程技术并在 GUI 和 applet 编程中使用此技术。

内容广泛

本书同时涵盖基本主题和高级主题，重点介绍 OOP 和实际应用程序。

- 基本主题——Java 语言的结构、类、对象、使用对象解决问题、OOP 提示、Java 程序结构、编译、执行、错误处理和调试
- OOD 和 OOP——通过继承、超类和子类、方法重写、兼容于插件的对象、多态性、抽象超类、接口、统一的公共接口计划、对象克隆、迭代器、设计模式和 Java 中的模型-视图-控制器(MVC)模式来扩充程序
- 通用编程和多态性编程——编写和使用类型独立的程序或可以使用对象层次结构的程序；Java 集合框架
- GUI——使用 Swing 构件为程序构造图形用户界面
- applet——编写和部署基于 Swing 的 applet、Java 插件
- 线程——多线程的概念、技术和应用
- 高级主题——使用 URL 和套接字联网、编写服务器端和客户端代码、远程方法调用(RMI)、多线程、了解并发编程及其所面临的问题、互斥、调度、并发活动的协作、多线程应用程序的动画制作、通过 JNI 与本机程序(C、C++、F77)连接、安全管理器、签名程序

示例用来演示概念、构造和用法，并显示如何合并 Java 功能以实现目标。但是，广泛的主题、大量的完整示例、出色的附录和完整的索引并没有占用太多的篇幅。实际上，本书并不比一般的教科书厚。

灵活使用

本书的初衷是将它作为高等院校的学期课程或研究生的学习资料。学生们应该已经了解 C 或 C++语言。不过，本书包含了对语言本身的介绍，可以通过本书激励对 C/C++知识了解得很少的那些学生加倍努力。本书适用于介绍 Java、使用 OOP 的 Java，或使用 Java 的 OOP 原理方面的有关课程。也可将它用于讲授 OOP 理论、联网、图形用户界面设计、Web 编程或并发/并行编程课程。

本书还可用于制定专业培训课程。如果课时较少，则可以根据需要适当地将第 10~12 章跳过。高级课程可以分配较多的时间来阅读第 1 章和第 2 章，并从练习中选择更加实际的编程项目。

课程结束之后，可以将本书留作有价值的参考资料。

Web 站点

为本书提供服务的 Web 站点如下：

http://sofpower.com/java

通过该站点可以找到信息更新、Web 示例、实用练习、教师笔记(PDF 格式)、文章、常见的问题解答、参考文献链接及其他资源。

在整本书中，都对概念及其用法借助示例进行了演示。我们没有炮制示例，而是尽可能使示例具有实际价值，并且能作为一个完整的程序输入到计算机中。您可以从该站点上下载完整的示例集，并遵循简单的指导来安装这些示例。

便于参考

本书以便于您快速、方便地参考的方式组织和给出内容。书中共有 11 个附录，其中包括：“Java 和 ANSI C/C++之间的主要区别”、“Java 调试器：jdb”、相互参照的类列表，以及一个全面、详尽的索引。本书对于用 Java 语言编程的开发人员也颇具参考价值。

简　介

自从 1995 年 Sun Microsystems 推出 Java 以来，Java 编程系统就以史无前例的速度在世界范围内赢得了人们的认可。其之所以能够被广泛接受应主要归功于 Java 的面向对象、平台无关性、图形用户界面支持和 Internet/Web 功能。

Java 的核心是创新的 Java 编程语言和 SDK(Software Development Kit，软件开发工具包)。通用的 Java 语言完全面向对象。通过构建软件对象来编程。面向对象编程(Object-Oriented Programming，OOP)使得构建、维护、修改和重用软件更加简单。SDK 包含一套丰富的代码库和工具。同时，该语言和 SDK 使得编程更加方便和高效。

平台无关性是 Java 成功的一个重要原因。编译成 Java 字节代码的 Java 程序可在安装有 Java 解释程序或 Java 虚拟机(Java Virtual Machine，JVM)的任何计算机上运行，JVM 可提供执行 Java 程序的运行时环境。字节代码是一种体系结构，而且独立于操作系统，因此可以一次编写和编译 Java 程序，然后在任意系统上运行。许多供应商提供的 Java 解释程序还可以将字节代码实时编译成本机代码，以便提高执行速度。使用 Java，软件开发人员可以从繁忙的移植工作中摆脱出来，从而集中精力开发能在任何供应商提供的计算机上运行的高质量的软件。

Java 组件

可以从全世界的许多公司那里获得 Java 工具和软件。其中 Sun 公司提供的 Java 包括以下组件：

- Java 编程语言——支持 OOP 且带有 I/O、字符串、多线程和异常处理的核心类库。
- Java 编译器——将 Java 源代码编译为 Java 字节代码。
- Java 虚拟机——解释和执行 Java 字节代码。
- JFC(Java 基础类)——支持事件驱动和图形用户界面(GUI)编程。
- applet 和 Servlet 支持——帮助开发 Web 客户端和服务器端程序。
- Java 文档生成器——从 Java 源文件自动生成 HTML 格式的应用程序编程接口(API)文档。
- JAR (Java Archive)——对文件打包和解包，以通过网络快速、高效地传递。
- JDBC (Java DataBase Connectivity)——将 Java 前端与现有的数据库相连。
- RMI (Remote Method Invocation)——通过网络在远程计算机上启用对 Java 对象的访问。
- Java 调试器——帮助您查找和修复 Java 程序中的错误。
- Java 反汇编程序——从编译的字节代码中提取源代码信息。
- Java 头文件生成器——创建用 C、C++或 FORTRAN 程序扩充 Java 代码所需的头文件(.h)。
- JNI (Java Native Interface)——提供 C 和 C++代码接口。

- Java 文档——包括演示、教程或主题指南、工具使用说明、API 和其他信息。可以在 Web 上访问 Java 文档。

上面所列出的大多数项目均包含在 Sun 公司发布的 SDK 中。

新的工具和软件正在快速地引入。我们将关注 Java 语言和核心类。本书通过许多示例演示了常规、GUI、applet 和联网应用程序中的 Java OOP。

applet 和 GUI 编程

Web 正日益促进 Internet 的蓬勃发展，它正在强化机关、公司和个人在 Web 上获得和发布信息的能力。Java applet 是 Java 代码的一种特殊形式，它可以提供 Web 文档中的可执行内容。Web 浏览器可以使用 Java 插件在本地计算机上执行 applet。Java applet 是向 Web 页面中添加动画、音效和用户响应交互作用的一种方式。

Java 还具有对图形用户界面和事件驱动编程的良好支持。添加面向交互作用的窗口-鼠标的图形界面即可轻松地使用应用程序。Java 基础类(JFC)可提供构件(窗口对象)、事件处理、图形绘制、图像显示和可动态选择的外观。applet 是 JFC 的一部分。

OOP 的定义

OOP 是现代编程范型的一种选择。面向对象编程的主要目的是使用软件对象构建程序。可以将对象视为一个带有其自己的数据和程序的自含式计算实体。例如，现代计算机、窗口、菜单和文件夹通常由软件对象来表示。但是可以将对象应用到多种程序中。对象可以是航班预定记录、银行账户，甚至可以是汽车引擎。引擎对象将包括描述其物理属性的数据(称为字段)和管理其内部工作方式及它与汽车中的其他相关部件(还有对象)交互作用的方式的编程(称为方法)。

工资单系统具有员工记录、记时卡、加班、病假、税款和扣除额等对象。空中交通管制系统具有跑道、班机和乘客门之类的对象。因此，在 OOP 中软件对象与应用领域中所涉及的真实对象紧密对应。这种对应关系使得对计算机程序的了解和操纵变得非常简单。相反，传统的编程可处理字节、变量、数组、索引和很难与现有的问题相关联的其他编程人工因素。另外，传统的编程主要利用一个接一个的过程(称为算法)来完成要求的任务。鉴于此，传统的编程也称为面向过程的编程。

OOP 具备的优势

大型计算机程序的构造很复杂。大型软件系统的设计、实现、验证、维护和修订过程所需的成本很高。因此找到完成这些任务的简易方式尤为重要。在这方面，OOP 具有无穷的潜力。

OOP 可提供以下主要优势：

- 简单性——因为软件对象可以对应用领域中的真实对象建模，所以减少了程序的复杂度，使得程序结构变得既清晰又简单。
- 模块性——每个对象可构成一个单独的实体，该实体的内部工作方式与系统的其他部分

相分离。

- 可修改性——对 OO(Object-Oriented，面向对象)程序中所用的数据表示法或过程进行小的更改非常简单。如果保留对象的外部行为，那么在对象中所做的更改就不会影响程序的任何其他部分。
- 可扩充性——添加新功能或响应更改操作环境可能会引入少数新对象和修改现有的某些对象。
- 灵活性——OO 程序可以很灵活地适应不同的环境，因为不需修改对象即可更改对象之间的交互作用模式。
- 可维护性——可以对对象进行单独维护，这样就可以轻松地查找和修复问题，并添加带有警告性的“铃声和哨声”。
- 可重用性——可以在不同程序中重用对象。例如，可以在需要表(已经排序)的任何程序中使用创建表的对象。因此，可以在所需的时间段内通过预制的和预先测试的组件来创建程序，从而无需从头开始创建新程序。

OOP 的有关概念

OOP 技术的突破性观念是将程序与数据结合在一起。此观念改变了数据和程序之间的传统隔离。将程序和数据包装到一起称为封装，最终得到的是一个软件对象。在 Java 中，所有的过程都被封装起来，并将它们称为方法。例如，图形用户界面系统中的一个窗口对象(图 0-1)包含该窗口的物理尺寸、屏幕上的位置、前景色和背景色、边框样式和其他相关数据。与这些数据封装在一起的是用于移动和调整窗口本身、改变其颜色、显示其文本、将其缩小成图标等的方法。用户界面程序的其他部分只会调用窗口对象(通过向该对象发送预定义的消息)来执行这些任务。窗口对象的工作就是执行相应的操作并保留其更新的内部数据。这些任务的实际执行方式和内部数据结构都不是该对象外部程序所关心的。由对象可以理解的不同种类的消息构成的公共接口完全用于定义该对象的使用方式。此公共接口是该对象的应用程序编程接口(Application Programming Interface，API)。把内部详细信息隐藏起来可以使对象变得抽象，有时将该技术称为数据抽象化。

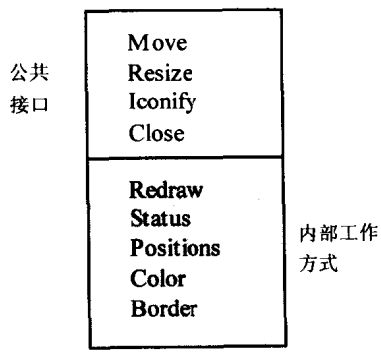


图 0-1　窗口对象

将公共接口与内部工作方式相分离并不难理解。实际上，这是日常生活中的普通做法。例

如，考虑一下银行出纳员。客户进入任意一家银行，然后使用同一组消息告诉任意一个出纳员：账号、存款、取款和余额等。每家银行或每位出纳员实际上都会保留记录或执行客户所不关心的内部任务。这些经过检验而可靠的原则简化了所有级别的业务，而且可以为组织程序带来相同的收益。当执行 OO 程序时，会创建对象、发送消息和销毁对象。这些只是允许在对象上执行的一些操作。禁止把对象中的内部(私有)数据或方法声明为公有。将对象中的私有机制与对象外部的例程相分离可以显著降低程序的复杂度。

通常情况下，需要有多个对象具有相同类型。例如，多个窗口经常会出现在工作站屏幕上。给定类型的对象通常是其定义可以指定这些对象及其公共接口的私有(内部)工作方式的类的实例。因此，在 OOP 中，要为所需的各种类型的对象定义类。类是创建特殊对象的蓝图。类定义和相应的初始值用于创建类的实例(对象)。此操作称为对象实例化。

OOP 技术通常要求使用简单的方式在其他对象的顶部构造对象。有两种主要方法：合成和继承。合成允许将现有的对象用作构建其他对象的组件。例如，计算器对象可能是算术单位对象和用户界面对象共同合成的对象。继承是 OOP 的主要特性，它允许在不更改现有类的代码的情况下对现有类进行扩充和更改。在 Java 中，此操作是通过类扩充完成的。子类可以继承其超类的代码，还可以添加自己的数据和方法。例如，图形窗口、文本窗口和终端仿真程序窗口都可以从窗口基类扩充而来。另外，支票、发票和申请书都可以从业务基类扩充而来。继承允许您从类似或相关对象之间提取共同体，还允许将 OO 软件库中的类用于许多不同或不可预知的用途。继承一个类是单一继承，继承多个类是多重继承。

此外，OOP 还支持多态性，也就是说一个程序能够操作多个不同的对象(作为可以互相交换数据的黑盒)。多态性允许创建可兼容的对象(这些对象可以互相交换)。修改和改进多态性程序是一件简单的事情，只需插入已更新的对象即可。

显然，OOP 提出了很多重要的概念。Java 提供了预先设计好的一套语言机制帮助您实现这些目标。只有通过实际编程才能充分理解这些概念的内涵。

Java 特性

由于许多原因，Java 很受欢迎，而且非常有用。其中的一些重要特性概括如下：

- 面向对象——Java 完全面向对象。Java 程序中没有任何附加功能。所有的方法都在类中定义。诸如整型和双精度型这样的基本数据类型都具有类包装。类本身就是对象，从而允许程序操纵类。
- 简单性——由于 Java 的语法与 ANSI C 和 C++的语法类似，因此学习起来非常容易。但是 Java 比 C++简单得多，而且文件尺寸小。Java 去除了头文件、预处理程序、指针运算、多重继承、运算符重载、结构体、共用体和模板。另外，Java 还可以自动执行无用单元收集，从而不需要显式进行内存管理。
- 简洁性——Java 是一种小巧型语言。其最简洁的版本可以控制小型应用。通过独立地对其他库进行打包，Java 解释程序和基类支持的文件尺寸都很小。
- 可移植性——Java 程序被编译为与体系结构无关的字节代码，而且可在安装有 Java 解释程序的任何平台上运行。Java 编译器和其他工具是用 Java 语言编写的。Java 解释程

序是用 ANSI C 语言编写的。无论如何，Java 语言规范都具备与实现方式无关的特性。

- 网络友好性——Java 具有用于网络通信、Web applet 和客户-服务器应用程序以及远程访问数据库、方法和程序的内置工具。
- GUI 支持——Java 基础类可使您简单、轻松地编写事件驱动 GUI。
- 动态增量加载和链接——在加载时动态地链接 Java 类。因此，向类中添加新的方法和数据字段不需要重新编译客户类。例如，在 C++中，修改后的头文件需要重新编译所有的客户文件。此外，应用程序还可以执行语句以查找字段和方法，然后利用它们执行相应操作。
- 国际化——用 Unicode 编写 Java 程序，Unicode 是 16 位字符编码，包括世界上广泛使用的语言的大多数字母。Java 操纵 Unicode 字符并支持本地日期/时间等，因此受到全世界的欢迎。
- 线程——Java 提供在一个 Java 程序中并发执行的多个控制流。线程允许 Java 程序一次执行多项计算任务，此功能支持事件驱动、联网和动画程序。
- 安全性——Java 安全措施包括对 Java applet 的有关限制、可重新定义的套接字实现方式和用户定义的安全管理器对象。它们使得 applet 值得信赖，而且允许应用程序履行和强制执行自定义的安全规则。

Java 中的 OOP

Java 完全面向对象。Java 程序由一个或多个类组成。可以将类组织到程序包中。Java 类通过封装数据成员(字段)和函数成员(方法)来定义软件对象。可以将成员指定为私有、受保护、程序包或公有，从而提供一种快捷的方式来定义对象的公共接口和私有域。

Java 的继承机制是类扩充，既可以实现类型关系又可以获得代码重用性。只允许单一继承。动态方法重写支持多态性，而且允许构建符合统一接口标准的可以互换的对象。Java 抽象超类可以帮助您为与插件兼容的对象集合安排统一接口并提供公共代码。

您可以在 Java 中指定接口。接口可以指定任何类为了特定用途所需的方法。例如，一个可排序的接口可以指定可排序的任何对象的行为。您可以实现所需的方法，从而使类符合接口的标准。一个类可以实现任意个接口，所以它的对象支持指定的某些行为。

通用容器是一种数据结构，您可以将任何类型的对象放置到其中。Java 集合框架提供预先设计的体系结构，以支持多种通用容器的表示法、操纵和互操作性。

联网

Internet 是使用 Internet 协议(Internet Protocol，IP)连接网络的一个全球网络。连接计算机网络的活动称为网络互联，因此被称为 Internet。Internet 可以连接世界范围内的各种组织机构——大学、政府机构、公司、图书馆、超级计算机中心、研究室、甚至还有个别家庭。Web 是 Internet 中的一大部分，它使用 HTTP(Hypertext Transfer Protocol，超文本传输协议)。Internet 上的连接数目和 Web 站点的数目很大，而且正在快速增长。

在 Web 上，资源和服务用 URL(Uniform Resource Locator，统一资源定位符)标识。Java 提供用于使用 URL 联网的有效类库，使您可以轻松访问标准的 Internet 服务。Java 还支持使用较

低级的 Internet 协议 TCP/IP 或 UDP/IP 的 Internet 套接字。

多线程

用 Fortran、C 或 C++这类语言编写的程序执行单一控制流程。这些都是单线程程序。Java 允许多个线程同时存在于同一个程序中，而且允许独立地执行这些线程。多线程功能可以带来许多好处，包括分离耗时的计算和灵活地处理事件。

使用多个线程，您可以将任务组织起来，在一个程序中并发或并行执行。为了实现功能强大的并行性，您需要克服众多与管理和协调同时发生的多项活动相关的难题。

面向对象设计

如果不利用面向对象的方法来创建设计方案，程序就可以很容易变成用 Java 语言编写的面向过程的程序。更为糟糕的是，这样的程序可能根本无法产生类。面向对象需要利用自己的方法来进行软件设计。基本的面向对象设计(Object-Oriented Design，OOD)技术包括分解方法、UML(统一建模语言)类和用例图、CRC 方法(类、职责和合作者)和设计模式。

相关 Web 站点和本书的组织

为本书服务的 Web 站点为您提供了丰富实用的资源，一个可供下载的示例软件包，能够帮助您学习开发经验，站点地址如下：

http://www.sofpower.com/java

该站点介绍了 Java，高效 OOP，applet，GUI 和 Swing 事件编程，Java 集合框架，联网，多线程，以及面向对象程序设计基础。站点内容对众多重要的主题进行了全面而深刻的阐述。

本书把对 Java 的全面介绍与 OOP 的概念和技术结合在一起。书中具有现实意义的示例不仅示范了单个特性，还演示了如何将它们有效地结合在一起。所有的示例代码都可以从站点 http://www.sofpower.com/java 下载获得。书后的附录补充说明了相关的背景和参考资料。

本书第 12 章介绍了大量高级主题，包括反射，运行时类加载，通过 javadoc 的文档生成，与本机程序(C，C++，F77)的接口，剪贴板，安全管理，签名 applet(signed applets)和 applet 间(interapplet)通信。

第 13 章概述了 OOD 基础知识，提供了一个采用 CRC 设计的计算器模拟程序，并解释了 Java 中 Swing 程序包的 MVC(模型-视图-控制器)体系结构。教师可以选择从第 13 章开始介绍 OOD 的内容。

本书适合作为高等院校高年级学生的教材，当然也适用于定制的培训课程。短期课程可以略去第 10～13 章的内容。高级课程可以加快基础知识的学习进程，尽快接触到第 13 章的内容，并选择具有实际意义的项目作为实践练习。

学习本书需要具备一定的 C 或 C++基础。对 Java 语言有所了解的读者同样也可以从本书中获得乐趣和挑战性。附录 K 提供了 Java 的基本背景知识，因此，一个积极主动的学生，即使缺少 C/C++编程经验，通过努力也能够掌握本书的内容。

目　　录