

## <<Java面向对象程序设计教程>>

### 图书基本信息

书名：<<Java面向对象程序设计教程>>

13位ISBN编号：9787302206507

10位ISBN编号：7302206503

出版时间：2009-11

出版时间：清华大学出版社

作者：李发致

页数：517

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## <<Java面向对象程序设计教程>>

### 前言

技术在发展，认识在加深，这是本书改版的主要原因。

作者认为，Java用于面向对象程序设计语言教学，JDK 1.4版本刚刚好，而JDK 1.5版本的新功能，如泛型、自动装箱、静态导入、实参个数可变等，基本上是由于减少开发人员的代码量而增加的。这些东西概念上不一定全都面向对象，理解起来不见得简单。

授课时，老师置之不理也未尝不可。

但往往同学好奇，如果老师不与时俱进，也会被同学质疑麻木不仁的。

于是从众，上课老老实实地提了，书自然也得改……当然，作者改书的原动力，还在于感觉原来关于Java语言的面向对象特性以及如何用Java完成面向对象程序设计的理解还不够深入、叙述还不太系统

。不重新组织一下、扩充一把，着实耿耿于怀，难得心安啊！

后来有了出版社之约，就开始重写了。

关于这本书我们都在演绎Java，都试图找出一条更好的学习、Java语言的途径。

本书的构思是围绕着面向对象技术和面向对象编程语言这一主题展开的，因为这是学习Java时无法回避的问题。

事实上，对象之于初学者是一个需要费点力才能跨过去的坎：引用变量怎么声明？

对象怎样创建、使用和清除？

类如何设计？

什么是符合面向对象编程规范的Java程序？

像这样一些问题，真正深究起来的确有许多值得思考的地方。

## <<Java面向对象程序设计教程>>

### 内容概要

本书围绕着面向对象技术和面向对象编程语言主题展开，注重对学习Java语言过程中的主要问题进行深入分析，讲解重点是怎样编写规范的Java程序。

书中首先介绍Java的运行机制、程序样式、基本语法；然后以面向对象程序实现为中心，讲解在Java语言环境下对象初始化、对象使用过程与继承性、多态性之间的联系，对面向对象编程规范等问题进行比较深入的梳理与分析；随后介绍Java的多线程、输入输出、集合框架与泛型、图形用户界面等专题内容。

本书在内容组织上紧扣面向对象编程，并强调编程规范的重要性，对知识点溯本求源，力求给出合理的解释；在内容编排上尽可能做到由浅入深，让读者能基本掌握知识点，然后逐渐把问题引向深入，给学有余力的读者提供一定的发展空间。

书中例子的选用与设计也尽可能典型精致，做到有的放矢。

本书的主要阅读对象是计算机及相关专业的大专院校学生，Java初、中级编程人员以及对从事面向对象编程感兴趣的读者。

## &lt;&lt;Java面向对象程序设计教程&gt;&gt;

## 书籍目录

第1章 Java导论	1.1 Java语言的特点	1.1.1 简单的面向对象程序设计语言	1.1.2 健壮安全的分布式语言	
	1.1.3 结构中立、可移植性强的解释型语言	1.1.4 高效能	1.1.5 支持多线程的动态语言	
	1.1.6 Java与几种常用语言的比较	1.2 Java执行环境	1.2.1 JVM机制Java平台的基础	
	1.2.2 自动垃圾收集	1.2.3 保护域机制与沙箱模型	1.2.4 类加载器启动引擎的点火器	
	1.3 JDK的组成与安装设置	1.3.1 Java平台的组成结构	1.3.2 SETPATH和SETCLASSPATH的作用	
	1.3.3 Java开发平台简介	1.4 典型例子及常见问题	1.4.1 编译器、解释器、小程序观察器和反编译器	
	1.4.2 简单的“HelloWorld!”例子	1.4.3 应用程序的主方法	1.4.4 小程序的运行	
	1.5 思考与练习	第2章 Java的基本语法		
	2.1 类型、值与基本变量	2.1.1 基本概念	2.1.2 基本类型	
	2.1.3 引用类型	2.1.4 类型转换	2.1.5 简化操作的一些辅助手段	
	2.2 运算符与表达式	2.2.1 算术运算符与表达式	2.2.2 关系运算符与表达式	
	2.2.3 逻辑运算符与表达式	2.2.4 位运算符与表达式	2.2.5 条件运算符与表达式	
	2.2.6 赋值运算符与表达式	2.2.7 instanceof运算符	2.2.8 运算符优先级与赋值顺序的理解	
	2.2.9 基本类型所支持的操作运算	2.3 语句	2.3.1 编组语句	
	2.3.2 表达式语句	2.3.3 选择语句	2.3.4 循环语句	
	2.3.5 迭代与递归	2.3.6 控制转移语句	2.3.7 注释语句	
	2.4 思考与练习	第3章 Java对象的生命周期		
	3.1 面向对象技术	3.1.1 面向过程与面向对象的比较	3.1.2 面向对象技术的基本原则	
	3.1.3 面向对象技术的基本要素	3.1.4 软件开发过程的面向对象技术	3.1.5 Java语言中的一些基本术语	
	3.2 对象声明	3.2.1 对象的声明方式	3.2.2 引用变量与对象的关系	
	3.3 对象的创建	3.3.1 创建方式	3.3.2 对象的初始化	
	3.3.3 继承链上相关类的加载顺序以及构造方法链的调用	3.4 对象的使用	3.4.1 使用了哪个对象	
	3.4.2 使用了哪个方法	3.4.3 使用了哪个属性	3.4.4 内部类	
	3.5 对象清除	3.5.1 垃圾自动收集机制	3.5.2 调用垃圾收集方法	
	3.5.3 finalize方法的利用	3.6 思考与练习	第4章 对象设计的Java规范	
	4.1 规范概述	4.1.1 硬约束与软约束	4.1.2 软件工程的要求	
	4.1.3 面向对象设计原则	4.2 Java语言的Object类及标准包		
	4.2.1 Object类	4.2.2 java.lang包	4.2.3 Java标准包	
	4.3 Java的异常处理机制	4.3.1 异常处理	4.3.2 核心代码与异常处理分离机制	
	4.3.3 引发异常及对异常处理的两条途径	4.3.4 Java的内置异常类	4.3.5 设计异常类	
	4.3.6 断言语句	4.3.7 对异常不作为与作为的区别	4.4 包、接口、类与方法设计	
	4.4.1 包设计	4.4.2 制作JAR文件包	4.4.3 接口设计	
	4.4.4 抽象类设计	4.4.5 类设计	4.4.6 方法设计	
	4.4.7 设计模式	4.5 Java编码的其他规范		
	4.5.1 命名规范	4.5.2 文件组织样式	4.5.3 增加程序可读性的一些建议	
	4.5.4 完整的例子	4.6 思考与练习		
	第5章 Java的线程			
	5.1 线程的概念与POSIX标准	5.1.1 现成的概念	5.1.2 线程的POSIX标准	
	5.2 多线程的Java实现	5.2.1 线程的创建方法	5.2.2 线程的状态	
	5.3 互斥线程间的同步机制	5.3.1 多线程带来的冲突问题	5.3.2 共享资源合理使用的实现	
	5.3.3 按同步协调程度划分的线程间的关系	5.4 java.util.concurrent包简介		
	5.5 思考与练习			
	第6章 Java的输入输出			
	6.1 流输入输出类的层次结构	6.1.1 流的概念	6.1.2 字节流	
	6.1.3 字符流	6.1.4 输入输出类的分类	6.1.5 标准流及其重定向	
	6.1.6 IOException及其子类	6.2 输入输出流类的应用		
	6.2.1 输入输出流类的般例子	6.2.2 典型的输入输出流类的组合应用	6.2.3 格式化输出	
	6.3 数据持久化	6.3.1 对象串行化	6.3.2 XML文件的输入输出	
	6.3.3 JDBC入门	6.4 文件类的应用		
	6.4.1 File类	6.4.2 File类应用举例	6.5 思考与练习	
	第7章 Java的集合框架与泛型			
	7.1 集合API	7.2 Collection与Iterator	7.2.1 Collection接口	
	7.2.2 迭代器Iterator	7.2.3 Iterator的使用举例	7.3 List、LinkedList与List	
	7.3.1 List接口	7.3.2 LinkedList与ArrayList类	7.3.3 List的使用举例	
	7.4 Set、SortedSet、HashSet与TreeSet	7.4.1 Set和SortedSet接口	7.4.2 HashSet、TreeSet和LinkedHashSet类	
	7.4.3 Set的使用举例	7.5 Map、SortedMap接口及其实现类	7.5.1 Map接口	
	7.5.2 SortedMap接口	7.5.3 HashMap、TreeMap和LinkedHashMap等实现类	7.5.4 Map的使用举例	
	7.6 泛型类型	7.6.1 向下转型	7.6.2 泛型的定义	
	7.6.3 泛型的使用	7.6.4 泛型的设计	7.7 思考与练习	
	第8章 Java的图形用户界面			
	8.1 JFC的组成	8.1.1 AWT	8.1.2 Swing	
	8.1.3 DragandDrop	8.1.4 Java2D	8.1.5 JavaAccessibility	
	8.2 应用程序和小程序界面图形化	8.2.1 Swing的根面板	8.2.2 小程序的典型例子	
	8.2.3 应用程序的典型例子	8.2.4 结合小程序和应用程序的典型例子		

<<Java面向对象程序设计教程>>

8.3 AWT的组件布局管理模型      8.3.1 基于策略模式的授权模型      8.3.2 布局管理器类      8.4 AWT的事件处理模型      8.4.1 基于观察者模式的授权模型      8.4.2 事件源      8.4.3 事件类      8.4.4 事件监听器  
8.4.5 创建事件监听器对象      8.5 思考与练习附录 源代码清单参考文献

章节摘录

插图：从以上的步骤看，可以发现Java虚拟机的执行机制和数控机床的执行机制非常相似。如果有一定的生活经验积累，对理解一些抽象的概念是有所帮助的。

3. 为什么要使用Java虚拟机？

由于web应用的地域分布很广，跨平台问题比以往的任何应用都突出。

高级语言如果要在不同的平台上运行，一般要把源代码移植到不同的平台上重新修改编译，最终生成不同的目标代码。

对于一个项目而言，如果要实现构建的代码在不同的平台上移植，有两条途径：一是让所有系统平台采用完全相同的标准架构，直接利用编译好的应用程序；二是面向各种异构平台修改源代码，重新编译、调试、运行。

对于第一条途径，从现状看其协调难度很大；对于第二条途径，从现状看受到的约束比较小，有实现的可能。

但是，当项目的代码量比较大时，移植过程的代码修改量往往超乎预料，让开发者头疼不已。

那么，有没有可能找到一个办法，让用户在现有的异构平台上移植代码的修改量最少呢？

答案是肯定的。

如果仔细分析，就会发现异构系统平台的底层描述确实存在一些差异，不过这些差异是有限且相对稳定的。

当采用编译型语言进行应用开发时，语言的编译环境与系统平台直接绑定，致使这些差异在各种应用中被反复频繁引用，从而引发“蝴蝶效应”，使得一个应用在异构系统平台中重新编译时不得不修改大量代码才能再编译运行。

## <<Java面向对象程序设计教程>>

### 编辑推荐

《Java面向对象程序设计教程(第2版)》：教学目标明确，注重理论与实践的结合教学方法灵活，培养学生自主学习的能力教学内容先进，强调计算机在各专业中的应用教学模式完善，提供配套的教学资源解决方案

<<Java面向对象程序设计教程>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>