

<<JAVA软件开发>>

图书基本信息

书名：<<JAVA软件开发>>

13位ISBN编号：9787030254955

10位ISBN编号：7030254953

出版时间：2009-11

出版时间：科学出版社

作者：张义

页数：361

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>

前言

Java的升级从来都不是对旧版本的彻底淘汰，而是在原有基础上进行“扬弃”的改进，并保持对旧版的兼容，让用户可以迅速适应新的特性。

自1995年以来，Java的芯片技术、数据库连接、Jini（基于Java的信息家电联网）和EnterpriseJavaBeans都已经逐渐和现代生活息息相关。它的安全、跨平台和高性能都是其他语言所无法比拟的，相信将来Java还会给我们带来越来越多的惊喜，在Java的推动下，我们的生活会更加便捷，更加丰富多彩！

本书由浅入深地对Java，编程规则及其应用进行了详细的讲解，全书由如下几个部分组成。

（1）Java及其面向对象特性。

第1章介绍了什么是Java和Java的由来；第2章介绍Java的基本语法知识；第3章介绍面向对象思想——继承、封装和多态；第4章介绍Java面向对象的设计；第5章介绍Java中类的高级特性。

（2）Java标准应用。

第6章介绍Java中的数据结构，首先使用伪码讲解，最后给出了Java的实现；第7章介绍了Java的异常处理机制；第8章介绍了输入输出系统；第9章介绍了小应用程序——Applet；第10章介绍了多线程机制；第11章介绍了基本的图形用户界面；第12章介绍了AWT的组件和事件处理机制；第13章介绍了Swing图形用户界面。

（3）Java高级应用。

第14章介绍了网络编程原理；第15章介绍了JDBC编程；第16章介绍了服务器小应用程序——servlet；第17章介绍了Struts和Hibernate；第18章简单地介绍了J2EE的概念；第19章简单地介绍了J2ME的概念。

<<JAVA软件开发>>

内容概要

《Java软件开发》探讨运用Java进行应用程序开发的指导教程，详尽地探讨了当前流行应用程序的开发工具——Java语言的核心技术。

全书共分23章。

内容主要包括4个部分，分别介绍了Java语言及其面向对象特征，Java基础应用程序开发，Java高级应用及网络应用开发，以及跨平台应用程序开发及Eclipse开发工具的使用等内容。

《Java软件开发》是广大Java软件设计、嵌入式及网络应用开发行业程序员的必备工具，亦可作为高校、社会培训班教师教材。

由于《Java软件开发》的专业性、实用性与易读性，现已被选为“IBM教育学院”、“英特尔软件学院”教育培养计划指定教材。

书籍目录

第1章 Java及Java开发环境概述1.1 Java的诞生及其影响1.2 Java的特征1.2.1简单1.2.2 面向对象1.2.3 分布式1.2.4 健壮1.2.5 体系结构中1.2.6 可移植1.2.7 解释执行1.2.8 高性能1.2.9 多线程1.2.10 动态1.3 Java5.0新特性1.4 安装Java开发工具1.4.1 JDK的取得1.4.2 安装并测试1.5 JDK开发工具包1.5.1 Javac1.5.2 Java1.5.3 Javadoc1.5.4 jar1.5.5 Javah1.5.6 Javap1.5.7 appletviewer1.5.8 idb1.5.9 native2ascii1.5.10 extcheck1.6 Java集成开发环境简介1.6.1 Eclipse发展背景1.6.2 Eclipse工作台简介第2章 Java语言基础2.1 Java关键字和标识符2.1.1 标识符2.1.2 关键字2.2 Java数据类型、常量和变量2.2.1 Java数据类型2.2.2 常量2.2.3 变量2.3 简单数据类型2.3.1 整数类型2.3.2 浮点类型2.3.3 字符类型2.3.4 布尔类型2.3.5 枚举类型2.3.6 综合举例2.3.7 自动类型转换与强制类型转换2.4 Java运算符及表达式2.4.1 Java运算符简介2.4.2 算术运算符2.4.3 关系运算符2.4.4 布尔逻辑运算符2.4.5 按位运算符2.4.6 赋值运算符2.4.7 条件运算符2.4.8 表达式及运算符优先级2.5 数组2.6 字符串2.6.1 字符串的初始化2.6.2 String和StringBuffer类2.6.3 StringBuilder类2.6.4 字符串的访问2.6.5 修改字符串2.7 Java流程控制2.7.1 条件语句2.7.2 循环语句2.7.3 转移语句第3章 面向对象思想3.1 结构化程序设计的方法3.2 面向对象的编程思想3.2.1 什么是对象3.2.2 什么是面向对象3.2.3 什么是类3.2.4 学会抽象整个世界——实体、对象和类3.2.5 面向对象方法——抽象的进步3.3 面向对象的特点3.3.1 继承3.3.2 封装3.3.3 多态性第4章 Java面向对象设计4.1 类和类的实例化4.1.1 定义类的结构4.1.2 类的实例化4.2 Java内存使用机制4.3 抽象类和接口4.3.1 抽象类4.3.2 接口4.4 命名空间与包4.4.1 包的基本概念4.4.2 自定义一个包4.4.3 源文件与类文件的管理4.5 现有类的使用4.5.1 访问权限4.5.2 使用import导入已有类4.5.3 静态导入4.5.4 类的继承和多态第5章 类的高级特性5.1 静态变量和方法5.1.1 静态变量5.1.2 静态方法5.2 常量、最终方法和最终类5.2.1 常量5.2.2 最终方法5.2.3 最终类5.3 抽象类和抽象方法的使用5.4 接口的使用5.4.1 接口的概念5.4.2 定义接口5.4.3 执行接口5.4.4 使用接口5.5 内部类的使用5.5.1 使用内部类的共同方法5.5.2 内部类5.5.3 内部类属性第6章 数据结构6.1 抽象数据类型6.2 基本数据结构6.2.1 向量6.2.2 线性表6.2.3 堆栈6.2.4 队列6.2.5 树6.2.6 图第7章 Java异常处理7.1 异常机制简述7.1.1 异常的概念7.1.2 异常的分类7.2 Java异常体系7.2.1 捕获异常7.2.2 声明异常7.2.3 抛出异常7.2.4 自定义异常第8章 Java输入 / 输出系统8.1 Java输入 / 输出体系8.2 字节流8.2.1 InputStream类8.2.2 OutputStream类8.2.3 FileInputStream类8.2.4 FileOutputStream类8.2.5 ByteArrayInputStream类8.2.6 ByteArrayOutputStream类8.2.7 管道流PipedInputStream和PipedOutputStream类8.2.8 过滤流FilterInputStream和FilterOutputStream类8.3 字符流8.3.1 Reader类8.3.2 Writer类8.3.3 FileReader类8.3.4 FileWriter类8.3.5 CharArrayReader类8.3.6 CharArrayWriter类8.3.7 PushbackReader类8.4 文件的读写操作8.5 对象序列化及其恢复8.5.1 Serializable接口8.5.2 ObjectOutputStream类8.5.3 ObjectInputStream类第9章 创建JavaApplet9.1 Applet类9.2 Applet概述9.3 Applet的使用技巧9.3.1 波浪形文字9.3.2 大小变化的文字9.3.3 星空动画9.3.4 时钟第10章 多线程10.1 多线程的概念10.1.1 多线程10.1.2 Java中的多线程10.1.3 线程组10.2 线程的创建10.2.1 通过实现：Runnable接口创建线程10.2.2 通过继承Thread类创建线程10.2.3 两种线程创建方法的比较10.3 线程的调度与控制10.3.1 线程的调度与优先级10.3.2 线程的控制10.4 线程的状态与生命周期10.5 线程的同步10.6 线程的通信10.7 线程池第11章 图形用户界面11.1 AWT及其根组件11.1.1 java.awt包11.1.2 根组件（Component）11.2 容器（Container）和组件11.3 布局管理器（LayoutManager）11.3.1 FlowLayout布局管理器11.3.2 BorderLayout布局管理器11.3.3 GridLayout布局管理器11.3.4 CardLayout布局管理器11.3.5 GridBagLayout布局管理器11.3.6 null布局管理器第12章 AWT基本组件及事件处理机制12.1 AWT基本组件12.1.1 Component类12.1.2 AWT事件模型12.2 GUI事件的处理12.2.1 AWT事件继承层次12.2.2 AWTEvent子类事件12.2.3 监听器接口12.3 几个简单组件12.3.1 按钮（Button类）12.3.2 标签（JLabel类）12.3.3 文本组件（TextField和TextArea类）12.4 使用类适配器（Adapter）进行事件处理12.5 使用匿名类进行事件处理第13章 Swing用户界面组件13.1 Swing组件库简介13.1.1 JFC和Swing13.1.2 Swing包概览13.1.3 Swing和AWT的区别13.1.4 示例程序SwingApplication13.2 Swing组件及其容器13.2.1 JComponent类13.2.2 AbstractButton及其子类13.3 JComboBox和JList组件13.4 JSlider类——滑杆13.5 JInternalFrame类第14章 网络通信程序设计14.1 Java.net包14.2 socket编程14.2.1 socket基础知识14.2.2 socket机制分析14.2.3 客户端编程14.2.4 服务器端编程14.2.5 服务器 / 客户端通信实例14.2.6 DatagramSockets编程第15章 Java数据库访问机制——JDBC15.1 JDBC介绍15.1.1 JDBC的概述15.1.2 JDBC——底层API15.1.3 JDBC的设计过程15.1.4 JDBC和ODBC的比较15.2 关

系数据库和sQL15.2.1 关系数据库15.2.2 关系数据库的应用模型15.2.3 结构化查询语言15.3 JDBC应用程序编程接口15.3.1 JDBC的类15.3.2 DriverManager15.3.3 JDBC驱动程序的类型15.4 JDBC编程基础15.4.1 JDBC访问数据库15.4.2 创建一个数据源15.4.3 数据库ORE15.4.4 建立与数据源的连接15.4.5 发送SQL语句15.4.6 处理查询结果15.5 基本JDBC应用程序15.5.1 JDBC在应用程序中的应用15.5.2 JDBC在Applet中的使用15.6 JDBC API的主要界面15.6.1 Statement15.6.2 ResultSet15.6.3 PreparedStatement15.6.4 CallableStatement15.7 事务管理15.7.1 保存点15.7.2 批量更新15.8 高级连接管理第16章 servlet16.1 servlet综述16.1.1 什么是servlet16.1.2 servlet的生命周期16.1.3 servlet与其他开发技术的比较16.1.4 servlet的应用范围16.1.5 配置servlet的开发的的环境16.2 servlet编程16.2.1 HTTP协议介绍16.2.2 简单程序servlet16.2.3 会话跟踪16.2.4 Servlet协作第17章 Struts与Hibernate入门17.1 MVC框架17.1.1 MVC模式17.1.2 基于Web应用的MVC模式17.2 Struts结构和处理流程17.3 Struts组件17.3.1 Web应用程序的配置17.3.2 控制器17.3.3 struts-config.xml文件17.3.4 Action类17.3.5 视图资源17.3.6 ActionForm17.3.7 模型组件17.4 Hibernate简介17.4.1 第一个Hibernate程序17.4.2 关联映射第18章 J2EE基础18.1 J2EE综述18.1.1 J2EE的主要特征18.1.2 J2EE的架构18.1.3 J2EE应用场景描述18.2 客户端层技术18.2.1 客户端层的问题18.2.2 客户端层的解决方案18.3 Web层技术18.3.1 Web层的目的18.3.2 Web层的解决方案18.4 EJB层技术18.4.1 EJB组件结构18.4.2 EJB层的目的18.4.3 EJB层的解决方案第19章 J2ME概述19.1 J2ME综述19.2 CLDC介绍19.2.1 CLDC类库介绍19.2.2 MIDLET介绍19.2.3 MIDlet界面19.3 CDC概述第20章 Java跨平台特性20.1 可移植性.....第21章 Java泛型程序设计第22章 Java编码规范第23章 使用Eclipse进行Java程序开发附录

章节摘录

Java解释器生成与体系结构无关的字节码指令，只要安装了Java运行时系统，Java程序就可在任意的处理器上运行。

这些字节码指令对应于Java虚拟机中的表示，Java解释器得到字节码后，对它进行转换，使之能够在不同的平台上运行。

1.2.6可移植 许多类型的计算机和操作系统都是连接到Interact上的。

要使运行于各种各样的平台上的计算机都能动态下载同一个程序，就需要有能够生成可移植性代码的方法。

与平台无关的特性，使Java程序可以方便地被移植到网络上的不同机器。

同时，Java的类库中也实现了与不同平台的接口，使这些类库可以移植。

另外，Java编译器是由Java，语言实现的，Java运行时系统由标准C实现，这使得Java系统本身也具有可移植性。

Java的可移植性取决于其中立的代码结构。

代码只需编写一次，字节码可以发布给不同的平台，不需要任何修改就可以运行。

其他语言中常见的一个移植问题就是基本数据类型（如整数、字符和浮点数）所占比特数不同。

例如，在C++中，int可以是一个16位整数，也可以是一个32位整数。

而在Java中的int，无一例外地都是32位的整数，不管它是在UNIX、OS / 2、Macintosh或者Windows上运行。

使用标准的整数定义可以避免类似上溢或下溢之类的错误，而这些错误对于基本数据类型大小（假定不一致时）是经常出现的。

1.2.7解释执行 Java编译器并不生成可执行程序的机器语言指令。

相反，它生成一种中间代码，成为字节码。

Java解释器直接对Java，字节码进行解释执行。

字节码本身携带了许多编译时信息，使得连接过程更加简单。

再者，由于其链结比较倾向于逐步增量与减量过程，因此发展程序更快、更精密。

由于编译期间的信息属于字节代码资料流的一部分，因此可以在运行期间携带更多的信息。

这正是链结器类型检查的基础，它也让程序更容易执行除错。

Java解释器和这种抽象机模型就称为Java虚拟机（Java Virtual Machine，JVM）。

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>