



Java

程序设计教程

北京科海 总策划

刘培文 主 编

走实践应用案例教学之路
培养技能型紧缺人才

21 世纪高职高专计算机教育规划教材

Java 程序设计教程

刘培文 主 编

中国人民大学出版社
• 北京 •

北京科海电子出版社
www.khp.com.cn

图书在版编目(CIP)数据

Java 程序设计教程/刘培文主编.

北京: 中国人民大学出版社, 2008

21 世纪高职高专计算机教育规划教材

ISBN 978-7-300-09874-6

I .J...

II .刘...

III.JAVA 语言—程序设计—高等学校: 技术学校—教材

IV. TP312

中国版本图书馆 CIP 数据核字 (2008) 第 165076 号

21 世纪高职高专计算机教育规划教材

Java 程序设计教程

刘培文 主编

出版发行 中国人民大学出版社 北京科海电子出版社

社 址 北京中关村大街 31 号

邮政编码 100080

北京市海淀区上地七街国际创业园 2 号楼 14 层

邮政编码 100085

电 话 (010) 82896442 62630320

网 址 <http://www.crup.com.cn>

<http://www.khp.com.cn> (科海图书服务网站)

经 销 新华书店

印 刷 北京市艺辉印刷有限公司

规 格 185mm×260mm 16 开本

版 次 2009 年 1 月第 1 版

印 张 18.25

印 次 2009 年 1 月第 1 次印刷

字 数 444 000

定 价 29.00 元

版权所有 侵权必究 印装差错 负责调换

丛书序

2006年北京科海电子出版社根据教育部的指导思想,按照高等职业教育教学大纲的要求,结合社会对各类人才的技能需求,充分考虑教师的授课特点和授课条件,组织一线骨干教师开发了“21世纪高职高专计算机教育规划教材”。3年来,本套丛书受到了高等职业院校老师的普遍好评,被几百所院校作为教材使用,其中部分教材,如《C语言程序设计教程——基于Turbo C》被一些省评为省精品课配套教材,这使我们倍感欣慰和鼓舞。

近年来,IT技术不断发展,新技术层出不穷,行业应用也在不断拓宽,因此教材的更新与完善很有必要,同时,我们也收到了很多老师的来信,他们希望本套教材能够进一步完善,更符合现代应用型高职高专的教学需求,成为新版精品课程的配套教材。在此背景下,我们针对全国各地的高职高专院校进行了大量的调研,邀请全国高职高专院校计算机相关专业的专家与名师、(国家级或省级)精品课教师、企业的技术人员,共同探讨教材的升级改版问题,经过多次研讨,我们确定了新版教材的特色:

- 强调应用,突出职业教育特色,符合教学大纲的要求。
- 在介绍必要知识的同时,适当介绍新技术、新版本,以使教材具有先进性和时代感。
- 理论学习与技能训练并重,以案例实训为主导,在掌握理论知识的同时,通过案例培养学生的操作技能,达到学以致用目的。

本丛书宗旨是,走实践应用案例教学之路,培养技能型紧缺人才。

丛书特色

☒ 先进性: 力求介绍最新的技术和方法

先进性和时代性是教材的生命,计算机与信息技术专业的教学具有更新快、内容多的特点,本丛书在体例安排和实际讲述过程中都力求介绍最新的技术(或版本)和方法,并注重拓宽学生的知识面,激发他们的学习热情和创新欲望。

☒ 理论与实践并重: 以“案例实训”为原则,强调动手能力的培养

由“理论、理论理解(或应用)辅助示例(课堂练习)、阶段性理论综合应用中案例(上机实验)、习题、大型实践性案例(课程设计)”五大部分组成。在每一章的末尾提供大量的实习题和综合练习题,目的是提高学生综合利用所学知识解决实际问题的能力。

- 理论讲解以“够用”为原则。
- 讲解基础知识时，以“案例实训”为原则，先对知识点做简要介绍，然后通过小实例来演示知识点，专注于解决问题的方法，保证读者看得懂，学得会，以最快速度融入到这个领域中来。
- 阶段性练习，用于培养学生综合应用所学内容解决实际问题的能力。
- 课程设计实践部分以“贴近实际工作需要为原则”，让学生了解社会对从业人员的真正需求，为就业铺平道路。

☑ 易教易学：创新体例，内容实用，通俗易懂

本丛书结构清晰，内容详实，布局合理，体例较好；力求把握各门课程的核心，做到通俗易懂，既便于教学的展开和教师讲授，也便于学生学习。

☑ 按国家精品课要求，不断提供教学服务

本套教材采用“课本 + 网络教学服务”的形式为师生提供各类服务，使教材建设具有实用性和前瞻性，更方便教师授课。

用书教师请致电（010）82896438或发E-mail: feedback@khp.com.cn免费获取电子教案。

我社网站（<http://www.khp.com.cn>）免费提供本套丛书相关教材的素材文件及相关教学资源。后期将向师生提供教学辅助案例、考试题库等更多的教学资源，并开设教学论坛，供师生及专业人士互动交流。

丛书组成

本套教材涵盖计算机基础、程序设计、数据库开发、网络技术、多媒体技术、计算机辅助设计及毕业设计和就业指导等诸多领域，包括：

- 计算机应用基础
- Photoshop CS3 平面设计教程
- Dreamweaver CS3 网页设计教程
- Flash CS3 动画设计教程
- 网页设计三合一教程与上机实训——Dreamweaver CS3、Fireworks CS3、Flash CS3
- 中文 3ds max 动画设计教程
- AutoCAD 辅助设计教程（2008 中文版）
- Visual Basic 程序设计教程
- Visual FoxPro 程序设计教程

- C 语言程序设计教程
- Visual C++程序设计教程
- Java 程序设计教程
- ASP.NET 程序设计教程
- SQL Server 2000 数据库原理及应用教程
- 计算机组装与维护教程
- 计算机网络应用教程
- 计算机专业毕业设计指导
- 电子商务

.....

编者寄语

如果说科学技术的飞速发展是21世纪的一个重要特征的话,那么教学改革将是21世纪教育工作不变的主题。要紧跟教学改革,不断创新,真正编写出满足新形势下教学需求的教材,还需要我们不断地努力实践、探索和完善。本丛书虽然经过细致的编写与校订,仍难免有疏漏和不足,需要不断地补充、修订和完善。我们热情欢迎使用本丛书的教师、学生和读者朋友提出宝贵的意见和建议,使之更臻成熟。

丛书编委会

2009年1月

内容提要

本书从 Java 语言的基本概念入手，较深入地讲解了面向对象程序设计的主要原理和方法，以及 Java 语言最核心的技术。全书共分为 12 章，首先介绍了 Java 语言的基本数据类型、流程控制、类和对象等基础知识；然后在深入分析面向对象设计方法的基础上，讲解了继承机制和多态性的实现与应用、异常的捕捉和处理以及集合类的定义方法；此外还讲解了多线程、图形用户界面、输入/输出流和文件操作以及数据库操作等内容，同时还提供了大量具有可实践性的程序实例、上机实战和练习题。

本书内容编排合理，整体结构完整，采取自顶向下的设计方法，突出学科方法论，强调知识体系可扩展的原则，既可作为高职高专院校计算机及相关专业的 Java 语言教材，也可作为软件开发人员的参考用书，还可以作为社会相关培训学校的培训教程。

前 言

随着计算机技术日新月异的发展以及互联网的普及,Java 语言已成为最具吸引力的程序设计语言之一。Java 语言完全面向对象,程序可读性强,具有诸多传统计算机编程语言无可比拟的优点。

本书从 Java 语言的基本概念入手,较深入地介绍了面向对象程序设计的主要原理和方法,以及 Java 语言最核心的技术。书中强调面向对象的思维方法训练,使读者能够在掌握理论的同时,培养一定的面向对象的设计和开发能力,为进行大型软件的研究、设计打下基础。书中对应各章内容还提供了涵盖相关章节知识点的上机实战,既阐明了原理和方法,又保证了一定的实用性,同时也确保了一定的广度和深度。

本书共分为 12 章,第 1 章介绍 Java 语言的特点、发展过程以及 Eclipse 集成化开发环境的使用;第 2 章介绍 Java 的基本语法知识以及程序流程控制等;第 3 章介绍面向对象程序设计的基本概念;第 4 章介绍继承、多态以及 Object 类;第 5 章介绍字符串操作;第 6 章介绍 Java 的数据结构;第 7 章介绍异常处理方面的相关知识;第 8 章详细探讨如何实现 Java 的线程以及线程同步的概念和应用;第 9 章介绍图形用户界面,包括 AWT 组件、Swing 组件和 Java 的事件处理机制等;第 10 章介绍 Java 输入/输出流和文件的操作;第 11 章介绍数据库的应用;第 12 章介绍学生管理系统的开发,巩固所学知识,强化实际应用技能。

本书内容编排合理,整体结构完整,采取自顶向下的设计方法,突出学科方法论,强调知识体系可扩展的原则,既可作为高职高专院校计算机及相关专业的 Java 语言教材,也可作为软件开发人员的参考用书,还可以作为各计算机培训学校的培训教材。

本书由刘培文主编,陈秀兰参与编写。由于时间仓促,加上编者水平有限,不足之处在所难免,恳请广大读者不吝指正。

编 者

2009 年 1 月

目 录

第1章 Java概述	1	2.4.7 instanceof运算符	32
1.1 了解Java.....	1	2.4.8 运算符优先级	32
1.1.1 Java的诞生和发展	1	2.5 控制语句	33
1.1.2 Java语言的特点和优势	3	2.5.1 块作用域	34
1.1.3 Java的核心技术	6	2.5.2 if语句	34
1.2 搭建Java开发环境	7	2.5.3 switch语句	35
1.2.1 J2SE的安装	7	2.5.4 循环控制	36
1.2.2 设置J2SE的环境变量	8	2.5.5 流程控制	38
1.2.3 编写Java程序	9	2.6 上机实战	41
1.3 Eclipse集成化开发工具	11	2.7 小结	44
1.3.1 基本概念	11	2.8 习题	45
1.3.2 安装Eclipse	12	第3章 面向对象的程序设计	47
1.3.3 使用Eclipse编写Java应用程序	13	3.1 面向对象概念	47
1.4 上机实战	14	3.1.1 面向对象介绍	47
1.5 小结	15	3.1.2 面向对象的特点	48
1.6 习题	15	3.2 对象	49
第2章 Java语言基础	17	3.2.1 对象的创建	51
2.1 标识符和关键字	17	3.2.2 对象的使用	52
2.1.1 标识符	17	3.3 类	53
2.1.2 Java关键字	18	3.3.1 类的定义	53
2.2 注释和分隔符	19	3.3.2 类的成员变量	54
2.3 数据类型、常量和变量	20	3.3.3 类的方法	58
2.3.1 数据类型	20	3.3.4 定义和使用构造方法	63
2.3.2 常量和变量	24	3.4 接口	64
2.4 运算符	25	3.4.1 接口的定义	65
2.4.1 算术运算符	25	3.4.2 接口的实现	65
2.4.2 关系运算符	26	3.4.3 接口的使用	66
2.4.3 逻辑运算符	28	3.4.4 回调	68
2.4.4 位运算符	29	3.5 内部类	70
2.4.5 复杂赋值运算符	31	3.5.1 内部类介绍	70
2.4.6 条件运算符	31	3.5.2 内部类的使用	71
		3.6 抽象类	73

3.6.1 抽象类概念	73	5.1.2 字符串比较	111
3.6.2 接口和抽象类	75	5.1.3 hashCode()方法	112
3.7 包	78	5.1.4 定位字符和子串	113
3.7.1 包的创建	78	5.1.5 抽取子串	114
3.7.2 包的导入	78	5.1.6 字符串修改	115
3.8 上机实战	79	5.1.7 String类的其他方法	115
3.9 小结	81	5.2 StringBuffer类	116
3.10 习题	81	5.2.1 StringBuffer类的构造函数	116
第4章 继续与多态	83	5.2.2 length()、capacity()、setLength() 和ensureCapacity()方法	116
4.1 继承	83	5.2.3 charAt()、setCharAt()、 getChars()和reverse()方法	117
4.1.1 属性和方法的继承	84	5.2.4 append()方法	118
4.1.2 动态绑定	85	5.2.5 插入和删除	119
4.2 继承的实现	85	5.3 StringTokenizer类	120
4.2.1 成员变量的继承和隐藏	85	5.3.1 StringTokenizer类的构造函数 ...	120
4.2.2 方法继承、方法覆盖和方法 隐藏	86	5.3.2 StringTokenizer类的常用方法 ...	120
4.2.3 super关键字	88	5.3.3 改进的StringTokenizer类： BreakIterator类	122
4.2.4 final关键字	90	5.4 使用正则表达式	124
4.3 方法和成员变量的权限控制	91	5.4.1 正则表达式	125
4.3.1 public	91	5.4.2 Pattern类	128
4.3.2 protected	92	5.4.3 Matcher类	129
4.3.3 private	94	5.4.4 操作符优先级	129
4.3.4 存取方法	95	5.4.5 正则表达式总结	130
4.4 多态	96	5.5 上机实战	132
4.4.1 多态性的概念	96	5.6 小结	134
4.4.2 覆盖实现多态性	97	5.7 习题	134
4.4.3 重载实现多态性	98	第6章 Java的数据结构	136
4.5 Object类	100	6.1 数组	136
4.5.1 toString()方法	100	6.1.1 一维数组	137
4.5.2 equals()方法	102	6.1.2 多维数组	139
4.5.3 Object与其他类型的转换	105	6.2 集合简介	141
4.6 上机实战	106	6.2.1 Java集合接口	141
4.7 小结	108	6.2.2 Java集合实现	142
4.8 习题	108	6.2.3 Java集合的发展历史	142
第5章 字符串操作	110	6.3 集合的使用	143
5.1 字符和字符串	110	6.3.1 Collection接口	143
5.1.1 String类的构造函数	110		

6.3.2 Set接口	145	8.5 上机实战	197
6.3.3 List接口	151	8.6 小结	201
6.3.4 Map接口	156	8.7 习题	202
6.4 上机实战	162	第9章 图形用户界面	204
6.5 小结	163	9.1 Java图形用户界面初步	204
6.6 习题	163	9.1.1 Java图形用户界面概述	204
第7章 异常处理	165	9.1.2 AWT图形用户界面组件概述	206
7.1 异常处理概述	165	9.2 Swing组件	209
7.2 程序错误与处理	165	9.2.1 Swing组件关系概述	209
7.2.1 程序错误、发现时刻及错误 处理原则	166	9.2.2 文本框	209
7.2.2 Java的错误和异常	166	9.2.3 按钮	210
7.3 Java的异常处理	166	9.2.4 下拉列表组件	216
7.3.1 异常类说明	167	9.3 上机实战	216
7.3.2 错误分类	168	9.4 小结	219
7.3.3 异常处理机制	169	9.5 习题	219
7.3.4 异常处理语句	170	第10章 输入/输出流和文件操作	221
7.3.5 声明异常	173	10.1 文件的概念	221
7.3.6 抛出异常	174	10.2 字节输入/输出流类	223
7.3.7 自定义异常类	176	10.2.1 字节输入流类 (InputStream类)	223
7.4 上机实战	177	10.2.2 字节输出流类 (OutputStream类)	224
7.5 小结	179	10.2.3 文件字节输入/输出流类 (FileInputStream/ FileOutputStream类)	224
7.6 习题	179	10.2.4 对象输入/输出流类 (ObjectOutputStream和 ObjectInputStream类)	226
第8章 多线程	181	10.3 字符输入/输出流类	228
8.1 多线程概述	181	10.3.1 字符输入/输出流抽象类 (Reader和Writer类)	228
8.1.1 什么是多线程	181	10.3.2 文件字符输入/输出流类 (FileReader/FileWriter类)	230
8.1.2 为何要使用多线程	182	10.3.3 字符缓冲流类 (BufferedReader类)	231
8.2 建立Java线程	182	10.4 文件操作类	232
8.2.1 计时器	182	10.4.1 文件类 (File类)	232
8.2.2 Thread类	184		
8.2.3 Runnable接口	190		
8.3 操作系统中的进程和线程	192		
8.3.1 进程	192		
8.3.2 线程	192		
8.4 线程优先级	193		
8.4.1 线程优先级和线程调度	193		
8.4.2 用户线程和守护线程	196		

10.4.2 文件过滤器接口.....	234	11.8 习题.....	256
10.4.3 随机存取文件类.....	235	第12章 综合实例.....	258
10.5 上机实战.....	237	12.1 学生管理系统.....	258
10.6 小结.....	238	12.1.1 系统应用背景.....	258
10.7 习题.....	239	12.1.2 系统的功能.....	258
第11章 Java与数据库的连接.....	241	12.2 系统设计.....	259
11.1 JDBC的概念.....	241	12.2.1 系统结构.....	259
11.2 数据库驱动程序.....	242	12.2.2 教师管理部分.....	259
11.3 JDBC-ODBC桥.....	242	12.2.3 学生管理部分.....	262
11.3.1 连接一个ODBC数据源.....	243	12.2.4 详细设计.....	264
11.3.2 读取数据.....	246	12.3 数据库设计.....	265
11.4 JDBC驱动.....	248	12.4 程序实现.....	267
11.5 Statement和PreparedStatement.....	250	12.5 实例拓展.....	279
11.6 上机实战.....	251	主要参考文献.....	280
11.7 小结.....	255		

第 1 章

Java 概述

CHAPTER 01

Java 语言是由 Sun 公司于 1995 年推出的一种编程语言,也是目前推广速度最快的程序设计语言,它采用面向对象的编程技术,功能强大且简单易学。Java 伴随着 Internet 的发展而成熟,是近 20 年来计算机软件环境中最有意义的进步之一。使用它可在不同种机器、不同种操作平台的网络环境中开发软件。Java 正在逐步成为 Internet 应用的主要开发语言,它彻底改变了应用软件的开发模式,带来了自 PC 机以来又一次技术革命,为迅速发展的信息世界增添了新的活力。



本章主要内容

- Java 的特点及发展过程
- 安装和使用 Java 开发运行环境
- Eclipse 集成化开发环境的使用

1.1 了解 Java

Java 是 Sun 公司研发的一种新型的程序设计语言。在高级语言已经非常丰富的背景下,Java 语言脱颖而出,不仅成为一门最为流行的计算机语言,而且形成一种专门的技术,这不仅有其独特的历史背景,还有其独树一帜的品质。

1.1.1 Java 的诞生和发展

Java 从诞生到现在已经有十多年时间,在这段时间里衍生了许多与 Java 相关的技术和标准。现在,我们来回顾一下 Java 的发展轨迹。

1. Java 的诞生

熟悉 Java 历史的人非常多,如果要问 Java 是哪年诞生的,也许大多数人都会回答是 1995 年(这个非常好记,因为微软公司的 Windows 95 也是在这一年发布的)。但事实上 Java 早在 20 世纪 90 年代初就开始酝酿了。

1991年4月, Sun公司的James Gosling领导的绿色计划(Green Project)开始着力发展一种分布式系统结构, 使其能够在各种消费类电子产品上运行。Green项目组的成员一开始使用C++语言来完成这个项目, 由于Green项目组的成员都具有C++背景, 因此他们首先把目光锁定在C++编译器上。Gosling首先改写了C++编译器, 但很快他就感觉到C++的不足, 需要研发一种新的语言Java来替代它。

17个月后, 整个系统完成了。这个系统是用于机顶盒的操作系统, 被命名为Oak。不过在当时市场不成熟的情况下, 他们的项目并没有获得成功, 但Java语言却得到了Sun公司总裁McNealy的赏识。

直至1994年下半年, 由于Internet的迅猛发展, 第一个万维网浏览器Mosaic诞生了。此时, 工业界对适合在网络异构环境下使用的语言有一种非常急迫的需求, James Gosling决定改变绿色计划的发展方向, 他们对Oak进行了小规模改造, 就这样, Java在1995年3月23日诞生了。Java的诞生标志着互联网时代的开始, 它能够被应用在万维网的平台上编写互动性极强的Applet程序, 因此1995年的Applet无疑带给了人们无穷的视觉和脑力震荡。

2. JDK 1.0

1995年Sun公司推出的Java还只是一种语言, 如果想开发复杂的应用程序, 则必须要有一个强大的开发库支持。因此, Sun公司在1996年1月23日发布了Java开发工具包(Java Development Kit, JDK) 1.0。这个版本包括了两部分: Java运行环境(Java Runtime Environment, JRE)和开发环境。在运行环境中包括了核心API、集成API、用户界面API、发布技术和Java虚拟机(Java Virtual Machine, JVM) 5个部分。在JDK 1.0时代, JDK除了AWT(一种用于开发图形用户界面的API)外, 其他的库并不完整。

Sun公司推出JDK 1.0后, 紧跟着在1997年2月18日发布了JDK 1.1。JDK 1.1相对于JDK 1.0最大的改进就是为JVM增加了即时编译器(Just In Time, JIT)。JIT和传统的编译器不同, 传统的编译器是编译一条指令, 运行完后再将其扔掉, 而JIT会将经常用到的指令保存在内容中, 在下次调用时就不需要再编译了。这样JDK在效率上有了非常大的提高。

3. Java 2时代

1998年12月4日, Sun发布了Java历史上最重要的一个JDK版本: JDK 1.2。这个版本标志着Java已经进入Java 2时代, 这个时期也是Java飞速发展的时期。Sun公司对Java进行了很多革命性的变化, 而这些革命性的变化一直沿用到现在, 对Java的发展形成了深远的影响。

JDK 1.2自从被分成了企业版(Java 2 Enterprise Edition, J2EE)、标准版(Java 2 Standard Edition, J2SE)和微缩版(Java 2 Micro Edition, J2ME)三大块以后, 引起市场的强烈反响。

Java 2还增加了很多新的特性, 其中最著名的当属Swing。Swing是Java的另一个图形库。它不但有各式各样先进的组件, 而且连组件风格都可替换。Swing出现后, 很快就抢了AWT的风头, 但Swing并不是为取代AWT而存在的, 事实上Swing是建立在AWT之上的。另外Java 2还在多线程、集合类和非同步类上做了大量的改进。

从JDK 1.2开始, Sun公司以平均每两年一个版本的速度推出新的JDK。2000年5月8日, Sun公司对JDK 1.2进行了重大升级, 推出了JDK 1.3。

Sun公司在JDK 1.3中同样进行了大量的改进, 主要表现在一些类库(如数学运算、

新的 Timer API 等)上,在 JNDI 接口方面增加了一些 DNS 的支持以及增加了 JNI 的支持。这使得 Java 可以访问本地资源,支持 XML 以及使用新的 Hotspot 虚拟机代替传统的虚拟机。

在 JDK 1.3 时代,相应的应用程序服务器也得到了广泛的应用,如第一个稳定版本的 Tomcat 3.x 在这一时期得到了广泛的应用,Weblogic 等商业应用服务器也逐渐被接受。

2002 年 2 月 13 日, Sun 公司发布了 JDK 历史上较为成熟的版本: JDK 1.4。

进入 21 世纪以来,曾经在 .NET 平台和 Java 平台之间发生了一次声势浩大的孰优孰劣的论战,Java 的主要问题就是性能,因此,这次 Sun 公司将主要精力放在了 Java 的性能上。在 JDK 1.4 中, Sun 公司对 Hotspot 虚拟机的锁机制进行了改进,使 JDK 1.4 的性能有了质的飞跃。同时由于 Compaq、Fujitsu、SAS、Symbian 和 IBM 等公司的参与,使 JDK 1.4 成为发展最快的一个 JDK 版本。到 JDK 1.4 为止,人们已经可以使用 Java 实现大多数的应用了。

4. JDK 的成熟期

虽然从 JDK 1.4 开始 Java 的性能有了显著的提高,但 Java 又面临着另一个问题,那就是复杂性。

虽然 Java 是纯面向对象语言,但它对一些高级语言特性(如泛型、增强的 for 语句)并不支持,而且与 Java 相关的技术,如 EJB 2.x,也由于它们的复杂性而很少有人问津。也许是 Sun 公司意识到了这一点,在 2004 年 10 月, Sun 公司发布了我们期待已久的版本: JDK 1.5,同时, Sun 公司将 JDK 1.5 改名为 J2SE 5.0。和 JDK 1.4 不同, JDK 1.4 的主题是性能,而 J2SE 5.0 的主题是易用。Sun 公司之所以将版本号 1.5 改为 5.0,就是预示着 J2SE 5.0 较以前的 JDK 版本有着很大的改进。

Sun 公司不仅为 J2SE 5.0 增加了诸如泛型、增强的 for 语句、可变数目参数、注释(Annotations)、自动拆箱(Auto-Unboxing)和装箱等功能,而且也更新了企业级规范,如通过注释等新特性改善了 EJB 的复杂性,并推出了 EJB 3.0 规范;同时又针对 JSP 的前端界面设计推出了 JSF,JSF 类似于 ASP.NET 的服务端控件,通过它可以很快地建立起复杂的 JSP 界面。

2006 年年底, Sun 公司又再接再厉地推出了 J2SE 6.0,它不仅在性能、易用性方面得到了前所未有的提高,而且还提供了对如脚本、全新的 API (Swing 和 AWT 等 API 已经被更新)等的支持。

Java 在发展的十几年时间里,经历了无数风雨,现在 Java 已经成为一种相当成熟的语言。Java 平台吸引了数百万的开发者,在网络计算遍及全球的今天,作为 Java 技术的基础,J2SE 功不可没,让我们期望 J2SE 伴随 Java 平台一路走好。

1.1.2 Java 语言的特点和优势

Sun 公司的白皮书对 Java 的定义是“Java: A Simple, object-oriented, distributed, robust, secure, architecture-neutral, portable, high-performance, multi-threaded, and dynamic language.”,即 Java 是一种简单的、面向对象的、分布式的、健壮的、安全的、体系结构中立的、可移

植的、高性能的、多线程的动态语言。这个定义充分说明了 Java 语言的特点。

1. 简单性

Java是一种简单的语言。语言的设计者尽量减小语言的规模,所以Java取消了许多语言中十分烦琐和难以理解的内容,例如C++的指针、运算符重载、类的多继承等,并且通过实现自动垃圾收集功能大大减少了程序设计者的内存管理工作。Java在外观上让大多数程序员感到很熟悉,便于学习,同时Java的编译器也很小,便于在各种类型的计算机上实现。

2. 面向对象

Java是一种面向对象的语言。这里的对象是指封装数据及其操作方法的程序实体。Java的程序设计集中于对象及其接口,Java提供了简单的类机制以及动态的接口模型,实现了模块化和信息封装。Java类可提供一类对象的原型,再通过继承机制实现代码的重用。

3. 分布式

Java是一种分布式语言。它有一套很齐全的通信及其相关功能的程序库,可以处理TCP/IP及其他协议,用户可用URL地址在网络上很方便地访问其他对象。利用Java来开发分布式的网络程序是Java的主要应用之一。

Java是面向网络的语言,是Internet上的通用语言。Java从设计上就支持网络应用,它是分布式语言,Java接口和Java类是以包(Package)的形式分组存放的。Java语言专门有一个支持网络的包Java.net,通过这个包中的类,支持各种层次的网络连接。例如,可以处理TCP/IP,用户可以通过URL地址在网络上很方便地访问其他远程对象,就如同存取一个本地(Local)文件系统一样简单。Java也以Socket类支持可靠的流(Stream)网络连接,所以可以产生分布式的客户机和服务器,这比原来的客户机/服务器方式更进了一步。

4. 健壮性

Java原来是用做编写消费类家用电子产品软件的语言,所以它是被设计成用来编写高可靠性和稳健性软件。

Java是一个强类型语言,具有扩展编译时检查潜在的类型不匹配问题的功能。Java要求用显式的方法声明,它不支持C语言风格的隐式声明。这些严格的要求保证了编译程序能捕捉到调用错误,从而保证了程序的可靠性。

5. 安全性

Java是一种安全的网络编程语言,不支持指针类型,一切对内存的访问都必须通过对象的实例来实现。这样能够防止他人使用欺骗手段访问对象的私有成员,也能够避免在指针操作中易产生的错误。此外,Java有多个层次的互锁保护措施,能有效地防止病毒侵入和蓄意的破坏行为。

6. 可移植性

Java具有很好的移植性,这主要得益于它与平台无关的特性。同时,Java的类库中也实现了与平台无关的接口,使得这些类库也能移植。而且,虽然Java编译器主要是由Java

本身来实现，但是 Java 的运行系统（虚拟机）则由标准 C 语言实现，因而整个 Java 系统都具有可移植性。

7. 解释型语言

Java 虚拟机能直接在任意计算机上解释执行 Java 字节码，因为字节码本身带有许多编译信息，使得连接过程更加简单。

Java 编译程序把 Java 源程序编译成字节码，而不是通常的汇编码或机器码，这是 Java 实现跨平台特点的关键所在。为了实际运行 Java 程序，还要使用解释程序来执行编译出来的字节码，所以从这种意义上说 Java 是一个解释型语言。实际上 Java 的前一阶段是编译，后一阶段是解释。

Java 字节码提供对计算机体系结构中立的目標文件格式，而且代码设计成可有效地传送程序到多个平台。Java 程序可以在任何实现了 Java 解释程序和运行环境的系统上运行，这个解释程序和运行环境实现了一个称为 Java 虚拟机的运行系统，简称 JVM。

8. 高效性

Java 是一种先编译后解释的语言，但和其他纯解释型语言不同，Java 字节码的设计使之能很容易地直接转换成对应于特定 CPU 的机器码或汇编码，当然这不如全编译型语言快。事实上，Java 比 C 语言写的程序执行起来平均慢 20 倍。不过要认识到，这个速度对交互式运行、图形用户界面和基于网络的应用来讲是适当的，因为在这些应用中，CPU 常常是有空闲的，它等待用户做某些事，或者等待来自网络上的程序或数据。

9. 多线程机制

Java 具有多线程机制，这使得应用程序能够并行地执行。它的同步机制也保证了对共享数据的共享操作，同时，线程具有优先级的机制，有助于分别使用不同线程完成特定行为，也提高了交互的实时响应能力。

Java 程序通过流控制来执行程序流，程序中单个顺序的流控制称为线程，多线程则指的是在单个程序中可以同时运行多个不同的线程来执行不同的任务。

多线程程序设计大大提高了程序执行效率和处理能力。对程序员来说，用 C 或 C++ 语言编写处理多线程的代码可能是很麻烦的事，而 Java 语言使具有多线程的程序设计很容易，因为 Java 对线程提供内部语言支持，Java 中的 Thread 类和 Runnable 接口使程序员很容易开发出多线程的应用程序。

如果计算机只有一个 CPU，则在给定时刻只有一个执行线程。但因为 CPU 迅速地在几个线程之间切换，从而造成了几个线程同时执行的假象，所以说单处理器系统只支持逻辑并发，而不支持物理并发。在多处理器系统上，几个线程才真正同时执行，实现物理并发。多线程程序的重要特征是：不管是否真正实现物理并发，多线程程序都支持逻辑并发。

10. 动态性

Java 比 C++ 语言更具有动态性，更能适应不断变化的环境。Java 不会因程序库某些部分的更新而需要全部重新编译程序，所以，在类库中可以自由地加入新的方法和实例变量，而不会影响用户程序的执行。并且 Java 通过接口（Interface）机制支持多重继承，使之比