



普通高等教育“十一五”国家级规划教材
(高职高专教育)

工程化程序设计 ——Java

张荣新 主 编



高等教育出版社
Higher Education Press

普通高等教育“十一五”国家级规划教材
(高职高专教育)

工程化程序设计——Java

Gongchenghua Chengxu Sheji——Java

张荣新 主编

时瑞鹏 李 勤 副主编



高等教育出版社·北京

HIGHER EDUCATION PRESS BEIJING

内容提要

本书是普通高等教育“十一五”国家级规划教材。

本书以培养读者能力为目的,以项目为中心,将知识点与项目相结合,用项目带动知识点的学习,在完成项目的同时学习相关知识,逐渐培养正确运用面向对象的思维方法分析问题和解决问题的能力。

全书共 12 章,主要内容包括:Java 开发基础知识、简单的 Java 程序、Java 编程基础知识、Java 面向对象技术、数组和字符串、异常处理与内存管理、小应用程序的设计与应用、输入与输出流的应用、多线程的应用、布局管理、事件处理以及数据库编程等。本书采用项目驱动的案例教程形式,按节进行知识点的细化和组织,将案例与知识有机地结合起来。

本书可作为应用型、技能型人才培养的各类教育相关专业的教学用书,也可供各类培训、计算机从业人员和爱好者参考使用。

图书在版编目(CIP)数据

工程化程序设计:Java / 张荣新主编. —北京:高等教育出版社,2010.1

ISBN 978-7-04-028654-0

I. 工… II. 张… III. JAVA 语言-程序设计-高等学校-教材 IV. TP312

中国版本图书馆 CIP 数据核字(2009)第 224063 号

策划编辑 杜 冰
版式设计 王 莹

责任编辑 郭福生
责任校对 刘 莉

封面设计 张志奇
责任印制 韩 刚

责任绘图 尹 莉

出版发行 高等教育出版社
社 址 北京市西城区德外大街 4 号
邮政编码 100120
总 机 010-58581000

经 销 蓝色畅想图书发行有限公司
印 刷 廊坊市文峰档案印务有限公司

开 本 787×1092 1/16
印 张 19.25
字 数 470 000

购书热线 010-58581118
咨询电话 400-810-0598
网 址 <http://www.hep.edu.cn>
<http://www.hep.com.cn>
网上订购 <http://www.landraco.com>
<http://www.landraco.com.cn>
畅想教育 <http://www.widedu.com>

版 次 2010 年 1 月第 1 版
印 次 2010 年 1 月第 1 次印刷
定 价 24.80 元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换。

版权所有 侵权必究

物料号 28654-00

前言

高等职业教育近年来得到了蓬勃的发展。项目驱动的教学方式作为目前高等职业教育课程改革的一种形式,受到了老师和学生的普遍欢迎。首先,项目驱动方式能够提高学生的学习兴趣和学习的主动性。其次,项目教学注重的是学生动手能力和实际操作能力的培养,特别适合高职层次的人才培养目标。

本书精心设计了众多案例,帮助学生更好地理解知识,掌握如何应用知识。将知识讲解融入案例中,并能很好地指导学生实践,这对于学生从实践中理解和巩固知识,培养应用能力,具有非常重要的实际意义。

本书的主要特点:

(1) 采用项目驱动的教学方式。以完成项目任务为中心,在项目实施过程中完成相关理论知识的讲解。

(2) 根据知识线索设计案例,分解相关知识点,有明确的目的和要求,针对性强。

(3) 坚持“基础为本源,实践出真知”的理念。

(4) 强调实用性,培养应用能力。

(5) 包含从 Java SE5 开始的一些新特性。

(6) 每一章的案例演练由浅入深,构成一个完整的项目,逐渐提升学生面向对象的程序设计能力。

本书的第1~3章介绍 Java 程序设计的基础知识;第4章介绍面向对象的概念、面向对象程序设计的思想、类的继承和多态、包和接口等内容;第5章介绍数组、字符串及常用方法;第6章介绍 Java 的异常处理机制及内存管理;第7章介绍小应用程序的设计与应用;第8章介绍 Java 的输入与输出的基础知识;第9章介绍多线程的基本概念和应用;第10章和第11章介绍图形用户界面设计,主要包括组件的创建和应用、布局管理、事件的处理等内容;第12章介绍数据库编程的方法。

学时分配表

序号	授课内容	学时分配	
		讲课	实践
1	Java 开发基础知识	2	2
2	简单的 Java 程序	2	2
3	Java 编程基础知识	2	2
4	Java 面向对象技术	4	4
5	数组和字符串	2	2

续表

序号	授课内容	学时分配	
		讲课	实践
6	异常处理与内存管理	2	2
7	小应用程序的设计与应用	2	2
8	输入与输出流的应用	4	4
9	多线程的应用	2	2
10	布局管理	4	4
11	事件处理	6	6
12	数据库编程	4	4
合计		36	36

本书由天津职业大学张荣新任主编。其中第1~5章由张荣新及大连交通大学刘金羽共同编写,第6章由天津职业大学李勤编写,第7~12章由天津职业大学时瑞鹏编写,最后张荣新进行统稿。在本书的编写过程中,得到了来自企业实践经验非常丰富的工程师刘彩民和温山的大力帮助和指导,在此深表感谢。

由于时间仓促,加上作者水平有限,书中难免存在不足和疏漏之处,恳请广大读者批评指正,使本书得以改进和完善。作者的联系方式:zrx-03@163.com。

作者

2008年10月

目 录

第 1 章 Java 开发基础知识

1.1 Java 的发展历史及特点	1	1.6 常用 Java 开发工具	10
1.1.1 Java 的发展历史	1	1.6.1 简介	10
1.1.2 Java 语言的特点	2	1.6.2 使用 Eclipse 进行开发	10
1.2 Java 与 C/C++ 的比较	4	1.6.3 使用 NetBeans 开发 Java 程序	14
1.3 Java 的运行机制	6	本章小结	14
1.4 Java 的不同版本	8	习题	14
1.5 Java 运行环境的建立	8		

第 2 章 简单的 Java 程序

2.1 Java 程序的开发流程	15	2.3.1 命名规范	20
2.2 一个简单的 Java 程序	16	2.3.2 增强程序的可读性	21
2.2.1 编写一个简单的 Java 应用程序	16	本章小结	22
2.2.2 编写 Java 小应用程序	19	习题	22
2.3 编程风格	20		

第 3 章 Java 编程基础知识

3.1 案例 1 两数交换	23	3.4.1 运算符	32
3.2 案例 1 相关知识	24	3.4.2 表达式	33
3.2.1 注释和关键字	24	3.4.3 应用拓展	35
3.2.2 标识符	26	3.5 案例 3 输出 1900—2008 年间的所 有闰年	35
3.2.3 数据类型	26	3.6 案例 3 相关知识	37
3.2.4 变量与常量	28	3.6.1 语句	37
3.2.5 简单数据类型的转换	29	3.6.2 应用拓展	46
3.2.6 应用拓展	30	本章小结	47
3.3 案例 2 温度转换	31	习题	47
3.4 案例 2 相关知识	32		

第 4 章 Java 面向对象技术

4.1 案例 1 学生类	49	4.2.1 面向对象技术概述	52
4.2 案例 1 相关知识	52	4.2.2 面向对象程序设计的基本概念	54

4.2.3 类的概念及定义	58	4.6.2 抽象类和抽象方法	102
4.2.4 对象	67	4.6.3 内部类	106
4.3 案例2 完善学生类	71	4.6.4 包	107
4.4 案例2 相关知识	74	4.7 拓展知识——Java 中已有的类	112
4.4.1 类的封装	74	4.7.1 Math 类	112
4.4.2 类的继承	80	4.7.2 Date 类	114
4.4.3 类的多态	90	4.7.3 GregorianCalendar 类	114
4.5 案例3 计算几何体的体积	95	本章小结	115
4.6 案例3 相关知识	98	习题	115
4.6.1 接口	98		

第5章 数组和字符串

5.1 案例1 成绩管理程序	117	5.4.2 变长字符串	131
5.2 案例1 相关知识——数组	121	5.4.3 拓展知识	134
5.3 案例2 字符串类的演示	126	本章小结	134
5.4 案例2 相关知识	128	习题	135
5.4.1 定长字符串	128		

第6章 异常处理与内存管理

6.1 案例 异常处理	137	6.2.4 自定义异常	146
6.2 案例相关知识	139	6.2.5 拓展知识——异常处理的优点	147
6.2.1 异常处理机制	139	6.3 内存的管理与回收	148
6.2.2 异常的捕获	141	本章小结	149
6.2.3 抛出异常	143	习题	149

第7章 小应用程序的设计与应用

7.1 案例1 一个简单的计算器	151	7.5 案例3 基本图形绘制	165
7.2 案例1 相关知识	156	7.6 案例3 相关知识	167
7.3 案例2 文字的打字机效果	159	本章小结	169
7.4 案例2 相关知识	161	习题	169

第8章 输入与输出流的应用

8.1 案例1 统计通过键盘输入字符的 程序	170	8.5 案例3 显示文件信息	179
8.2 案例1 相关知识	171	8.6 案例3 相关知识	181
8.3 案例2 显示文件基本信息的 程序	173	8.7 案例4 简易电话簿	184
8.4 案例2 相关知识——File 类	175	8.8 案例4 相关知识	186
		8.9 应用拓展——复制文件	188
		本章小结	194

习题	194
----------	-----

第 9 章 多线程的应用

9.1 案例 1 多线程执行	195	9.4 案例 2 相关知识	207
9.2 案例 1 相关知识——线程	197	9.4.1 线程的同步	207
9.2.1 线程的概念	197	9.4.2 线程间通信	208
9.2.2 线程的使用	198	本章小结	209
9.3 案例 2 生产者-消费者程序	203	习题	209

第 10 章 布局管理

10.1 案例 1 窗口容器	210	10.7 案例 4 GridLayout 布局 管理器	219
10.2 案例 1 相关知识	211	10.8 案例 4 相关知识	221
10.3 案例 2 FlowLayout 布局 管理器	213	10.9 案例 5 CardLayout 布局 管理器	221
10.4 案例 2 相关知识	216	10.10 案例 5 相关知识	224
10.5 案例 3 BorderLayout 布局 管理器	216	本章小结	225
10.6 案例 3 相关知识	218	习题	225

第 11 章 事件处理

11.1 案例 1 JButton 事件处理	226	11.8 案例 4 相关知识	239
11.2 案例 1 相关知识	228	11.9 案例 5 JList 事件处理	240
11.3 案例 2 JTextField 和 JPasswordField 事件处理	230	11.10 案例 5 相关知识	242
11.4 案例 2 相关知识	232	11.11 案例 6 鼠标事件处理	244
11.5 案例 3 JCheckBox 和 JRadioButton 事件处理	233	11.12 案例 6 相关知识	245
11.6 案例 3 相关知识	236	11.13 案例 7 键盘事件处理	246
11.7 案例 4 JComboBox 事件处理	237	11.14 案例 7 相关知识	248
		本章小结	249
		习题	249

第 12 章 数据库编程

12.1 案例 1 使用 JDBC-ODBC 桥实现 数据库的连接	251	12.6 案例 3 相关知识	264
12.2 案例 1 相关知识	255	12.7 案例 4 数据更新	268
12.3 案例 2 使用纯 Java JDBC 驱动程序 实现数据库的连接	257	12.8 案例 4 相关知识	274
12.4 案例 2 相关知识	258	12.9 案例 5 获取数据库的基本 信息	275
12.5 案例 3 读取数据	260	12.10 案例 5 相关知识	278
		12.11 案例 6 使用表格浏览数据	279

IV 目录

12.12 案例 6 相关知识	283	12.16 案例 8 相关知识	294
12.13 案例 7 使用预处理语句	284	本章小结	295
12.14 案例 7 相关知识	288	习题	295
12.15 案例 8 读写照片数据	288		
参考文献			296

本章深入浅出地介绍 Java 程序开发的一些基础知识,包括 Java 的特点、Java 与 C++的区别、Java 的运行机制及 Java 的不同版本,还介绍了 Java JDK 的安装与配置及常用 Java 开发工具,并给出了相关的学习建议。



学习目标

- 了解 Java 的发展历史。
- 理解 Java 的特点及其与 C/C++的区别。
- 理解 Java 的运行机制。
- 熟悉 Java 运行环境的配置及常用开发工具的使用方法。



重点和难点

- Java 的特点及其与 C/C++的区别。
- Java 的运行机制。

1.1 Java 的发展历史及特点

Java 语言是优秀的面向对象程序设计语言,具有平台独立、安全、稳定和多线程等优良特性,是极为健壮的程序设计语言。Java 语言不仅可以用来开发桌面应用程序,而且可以用来开发嵌入式应用程序和 Web 应用等企业级应用程序。Java 所具有的“一次编写,到处运行”的特点,使得 Java 已成为网络时代最重要的程序设计语言之一。

1.1.1 Java 的发展历史

Java 语言的出现源于对平台独立语言的需要,希望这种语言能编写出嵌入各种家用电器等设备的芯片上且易于维护的程序。但是,人们发现当时的编程语言,比如 C、C++等都有一个共同的缺点:只能对特定的 CPU 芯片进行编译。这样,一旦电器设备更换了芯片就不能保证程序正确运行,可能需要修改程序并针对新的芯片重新进行编译。

Java 的发展历史可以追溯到 1990 年。当时 Sun Microsystem 公司为了发展消费类电子产品而进行了一个名为 Green 的项目计划。这个计划的负责人是 James Gosling。由他领导的开发小组开始致力于发展小型、可靠、可移植、分布式、实时的嵌入式系统。项目一开始决定采用 C++,

如上所述,遇到一系列的困难。最后决定更换程序设计语言,开发了一种与平台无关的新语言,并取名为 Oak。它采用了大部分与 C++ 类似的语法,对容易引发程序错误的功能加以改进,例如,将内存管理改为由语言自己进行管理,以减少程序员的负担及可能发生的错误。

到了 1994 年, Oak 的技术已日趋成熟,这时 Internet 也开始蓬勃发展。Oak 研发小组发现它很适合作为一种 Internet 上的程序设计语言,因此开发了一个能与 Oak 配合使用的浏览器——Hotjava。实践证明, Oak 的确能用于 Internet 上的程序开发。鉴于 Oak 已经被其他产品注册使用,研发小组就以常饮用的爪哇咖啡(Java)重新命名该产品。此后,Java 的快速发展也得益于 Internet 和 Web 的发展,Internet 上有各种不同的计算机,它们可能使用完全不同的操作系统和 CPU 芯片,但仍希望运行相同的程序。Java 的出现标志着真正的分布式计算时代的到来。

1.1.2 Java 语言的特点

Java 语言具有简单、安全、易用、面向对象、平台独立、可移植、动态、多线程等特点。

1. 简单性

Java 语言简单、高效。基本 Java 系统(编译器和解释器)所占空间不足 250 KB。Java 最初是为对家用电器进行集成控制而设计的一种语言,因此它必须简单明了。Java 语言的简单性主要体现在 3 个方面:

- Java 的风格类似于 C++,因而 C++程序员初次接触 Java 语言,就会感到很熟悉。从某种意义上讲,Java 语言是 C 及 C++语言的一个变种,因此,C++程序员可以很快地掌握 Java 编程技术。
- Java 摒弃了 C++中容易引发程序错误的一些特性,如指针、结构以及内存管理等。
- Java 提供了丰富的类库,可以帮助程序员方便地开发 Java 程序。

2. 面向对象

Java 是面向对象的编程语言,它支持继承、重载、多态等面向对象的特性。Java 集中于对象及其接口,它提供了简单的类机制以及动态的接口模型。对象中封装了它的状态变量以及相应的方法,实现了模块化和信息隐藏;而类则提供了一类对象的原型,并且通过继承机制,子类可以使用父类所提供的方法,实现了代码的复用。

3. 健壮安全的分布式语言

Java 是面向网络的语言。通过它提供的类库可以处理 TCP/IP 协议,Java 应用程序可以借由 URL(Uniform Resource Locator,在 Internet 的 WWW 服务程序上用于指定信息位置的表示方法)通过网络访问各种资源,就如同访问本地文件系统一样简单。

Java 在编译和运行程序时,都要对可能出现的问题进行检查,以消除错误的产生。它提供自动垃圾回收机制来进行内存管理,防止程序员在管理内存时容易产生的错误。通过异常处理机制,在编译时,Java 提示出可能出现但未被处理的异常,帮助程序员正确地进行选择,以防止系统的崩溃。另外,Java 在编译时还可捕获类型声明中的许多常见错误,防止动态运行时不匹配问题的出现。

用于网络、分布环境下的 Java 必须防止病毒的入侵。为此,Java 非常注重安全性。Java 的验证技术是以公钥加密算法为基础。“健壮性”与“安全性”之间存在一种很强的相互作用关系。例如,Java 不支持指针,一切对内存的访问都必须通过对象的实例变量来实现,这样就防止黑客使用“特洛伊”木马等黑客工具访问对象的私有成员,同时也避免了指针操作中容易产生的错误。

4. 平台独立性

平台独立是指程序不受操作系统平台的限制,可以运行在各种平台上。Java 源程序经过编译后生成字节码(bytecode)文件,而字节码与具体的计算机无关。只要计算机安装了能解释执行字节码的 Java 虚拟机(Java Virtual Machine,JVM),就可以执行字节码文件,从而实现 Java 的平台独立性。

为了理解 Java 语言的平台独立性,在此对 Java 虚拟机做一介绍。JVM 实际上是在计算机上由软件模拟出来的一个抽象计算机,它是一种运行环境,又称为运行时系统。JVM 并不认识所谓的 Java 语言,而仅仅能够读懂字节码文件中有限的指令、符号及其他一些辅助信息。此外,JVM 运行时能够操作各个内存区域,用以实现与操作系统和硬件环境的交互。

大部分高级语言的源程序必须经过编译或解释程序翻译成机器语言,才能在计算机上执行。例如 C、C++等属于编译型语言,BASIC 和 LISP 属于解释型语言。而 Java 程序必须先经过编译,再经过解释过程才能执行。通过编译器 Java 语言源程序转换成与平台无关的中间编码——字节码。字节码再经过解释器的解释,转换为机器码,便可在计算机上运行。图 1-1 说明了 Java 程序的执行流程。

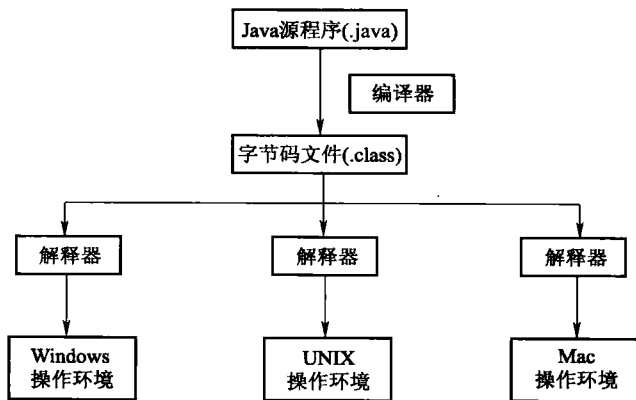


图 1-1 Java 程序执行的平台独立性

图 1-1 说明,Java 程序的执行流程为:创建 Java 源程序(.java)→经编译器编译→生成字节码文件(.class)→经解释器运行→计算机执行字节码。

5. 可移植性

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计算机上运行。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。

6. 多线程

Java 语言的一个重要特性就是在语言级支持多线程程序设计。多线程机制使应用程序能够并行执行,而且同步机制保证了对共享数据的正确操作。比方说,假如要做一张桌子,如果由一个人完成这张桌子——就好像单线程,那么需要先做桌面,做完桌面后,再做 4 个桌子腿;如果现在由 5 个人来做桌子——如同在程序中开辟 5 个线程,其中 1 个人做桌面,同时另外 4 个人分别做 4 个桌子腿,那么这两种方式效率的高低,相信大家都能区分出来。通过使用多线程,程序设计者可以分别用不同的线程完成特定的行为,而不需要采用全局的事件循环机制,这样就很容易地实现网络上的实时交互行为。

7. 动态性

Java 的动态特性是其面向对象设计方法的扩展。它允许程序动态地装入运行过程中所需要的类,这是采用 C++ 语言进行面向对象程序设计所无法实现的。

1.2 Java 与 C/C++ 的比较

首先应该清楚,Java 是由 C++ 发展而来的,它虽然保留了 C++ 的大部分内容,但语法更清晰,规模更小,更易学。它是在对多种程序设计语言进行了深入细致研究的基础上,摈弃了其他语言的不足之处,从根本上解决了 C++ 的固有缺陷,而产生的一种新的完全面向对象的语言。

Java 和 C/C++ 的相似之处多于不同之处,有 C 基础的读者,学习 Java 会更容易。相比较而言,Java 的编程环境更为简单。因篇幅所限,这里不能完全列出不同之处,仅列出一些比较显著的区别。

1. 全局变量

在 Java 程序中,不能在所有类之外定义全局变量,只能通过在一个类中定义公用、静态的变量来实现一个全局变量。例如:

```
Class GlobalVar {  
    public static global_var;  
}
```

在类 GlobalVar 中定义变量 global_var 为 public static,使得其他类可以访问和修改该变量。

Java 对全局变量进行了更好的封装。而在 C/C++ 中,依赖于不加封装的全局变量常常造成系统的崩溃。

2. 指针

指针是 C/C++ 中最灵活也是最容易产生错误的数据类型。由指针所进行的内存地址操作

常会造成不可预知的错误;另外,通过指针对某个内存地址进行显式类型转换后,可以访问一个 C++中的私有成员,从而破坏安全性,造成系统的崩溃。而 Java 无指针,并且增加了自动的内存管理功能,从而有效地防止了 C/C++语言中指针操作失误所造成的系统崩溃。因为在 Java 中无指针,没有指针的程序无法访问不属于它的内存,从而消除了 C/C++中一些常见的错误,这有利于提高 Java 程序的安全性。

3. 多重继承性

C++支持多重继承,这是 C++的一个特征,它允许从许多父类派生一个类。尽管多重继承功能很强,但使用复杂,而且会引起许多麻烦,编译程序实现它也很不容易。Java 不支持多重继承,但允许一个类继承多个接口,实现了 C++多重继承的功能,又避免了 C++的许多缺陷。

4. 内存管理

在 C 中,程序员通过库函数 `malloc()` 和 `free()` 来分配和释放内存,C++中则通过运算符 `new` 和 `delete` 来分配和释放内存。再次释放已释放的内存块或未被分配的内存块,会造成系统的崩溃;同样,忘记释放不再使用的内存块也会逐渐耗尽系统资源。而在 Java 中,所有的数据结构都是对象,通过运算符 `new` 为它们分配内存堆。通过 `new` 得到对象的处理权,而实际分配给对象的内存可能随程序运行而改变,Java 对此自动地进行管理并且进行垃圾回收,有效防止了由于程序员的误操作而导致的错误,并且更好地利用了系统资源。

5. 数据类型的支持

在 C/C++中,对于不同的平台,编译器对于简单数据类型(如 `int`、`float` 等)分别分配不同长度的字节数,例如,`int` 在 IBM PC 中为 16 位,在 VAX-11 中为 32 位,这导致了代码的不可移植性。但在 Java 中,对于这些数据类型总是分配固定长度的位数,且与具体的机器无关,如对 `int` 型,它总占 32 位,这就保证了 Java 的平台无关性。

6. 类型转换

在 C/C++中,可以通过指针进行任意的类型转换,常常带来安全隐患;而在 Java 中,运行时系统对对象的处理要进行类型相容性检查,以防止不安全的类型转换。

7. 预处理功能

Java 没有预处理器,因此不支持预处理功能。C/C++在编译过程中都有一个预编译阶段,要用到预处理器。预处理器为开发人员提供了方便,但增加了编译的复杂性。在 Java 程序中,若想使用另一个库里的类,只需使用 `import` 命令,并指定库名即可。与 C++预处理器的功能类似。

8. 结构和联合

Java 中不包含结构和联合,所有的内容都封装在类中。

9. goto 语句

goto 语句是 C/C++ 语言技术上的合法部分,但 goto 语句使程序结构变得混乱,不易理解。goto 语句主要用于无条件转移子程序和多结构分支。鉴于以上理由,Java 不提供 goto 语句,它虽然指定 goto 作为关键字,但不支持它的使用,使程序简洁、易读。

10. 提前声明

Java 中不必进行提前声明。若想在定义前使用一个类或方法,只需直接使用它即可——编译器会保证使用恰当的定义。这和 C/C++ 不同,不会碰到任何涉及提前引用的问题。

11. 头文件

Java 用包取代了命名空间。由于将所有东西都置入类中,而且采用了一种名为“封装”的机制,它能针对类名进行类似于命名空间分解的操作,所以命名的问题不再进入考虑之列。包也会在单独一个库名下收集库的组件。只需简单地导入一个包,剩下的工作会由编译器自动完成。

1.3 Java 的运行机制

1. Java 虚拟机

Java 语言是一种解释型的程序设计语言,即 Java 程序需要通过 Java 解释器的解释之后才能执行,这个解释器就是前面提到过的 Java 虚拟机的一部分。

虚拟机(virtual machine)是一种用软件模拟硬件功能实现的“虚拟”计算机。Java 语言采用虚拟机技术实现 Java 程序的跨平台运行,这就是 Java 虚拟机(Java Virtual Machine, JVM)。通常, JVM 在形式上表现为一个软件。与常见的计算机硬件结构类似,虚拟机中定义了指令集、寄存器集、类文件结构栈、垃圾回收堆、内存等,从而实现了计算机的基本框架。

与传统的解释型语言(如 BASIC 等)不同的是,Java 语言源程序(. java 文件)需要首先被编译成一种称为字节码的二进制文件(. class 文件),而不是机器码文件,然后 Java 解释器解释执行这种字节码文件。这一过程如图 1-2 所示。

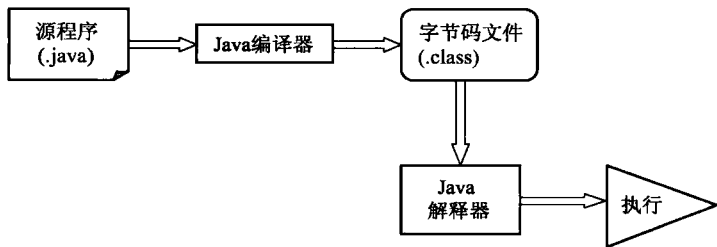


图 1-2 Java 程序的执行机制

也可以用硬件实现 Java 虚拟机。目前, Sun 公司已经设计实现了 Java 芯片, 主要应用在网络计算机上。Java 芯片的出现也会使 Java 更容易嵌入家用电器中。Java 编程人员在编写完软件后, 通过 Java 编译器将 Java 源程序编译成 JVM 的字节码。任何一台机器只要安装了 Java 解释器, 就可以运行这个程序, 而不管这种字节码是在何种开发环境中生成的。目前, JVM 已有多款软硬件平台的版本, 可以适应绝大多数的开发与应用。

Java 语言是一种网络环境的程序设计语言, 为了提高其安全、可靠性, Java 虚拟机在运行 Java 字节码程序时提供了垃圾回收机制、代码安全机制等。

2. 垃圾回收机制

垃圾回收机制是 Java 语言为了减轻编程人员在内存资源分配上的负担而设计的一种内存资源自动回收机制。在 C/C++ 语言中, 程序员不仅要通过编程为程序中的数据分配内存, 而且常常为了充分利用有限的内存资源, 通过编程回收不再使用的内存。这一过程一方面增加了程序员的编程负担, 另一方面也会因程序员的疏忽造成资源的浪费。Java 语言中的内存资源回收是自动完成的。系统在程序执行过程中跟踪分配给对象的内存资源的使用情况, 进行定期检查, 找出不再使用的内存并由系统自动回收这部分内存, 以便于再次分配。这样可以提高内存资源利用率, 同时减小系统和软件出现故障的几率, 提高软件的安全性与可靠性。

3. 代码安全机制

Java 安全模式的重点在于保护最终用户不受从网上下载的破坏性程序的干扰。为了达到这个目的, Java 在扩展其技术应用领域的同时, 也在不断改进其安全模型。Java 1.0 提供了一个专用的运行 Java 程序的沙箱(sandbox)。沙箱内可做任何事情, 但超出此边界就不能有任何操作。例如, 对于运行于网页中的小应用程序(applet), 通过限定下载代码的可执行范围, Java 安全模式可使用户免受破坏性程序的威胁。

在安全沙箱中, JVM 的一个重要组成部分是类载入程序。该程序负责输入那些定义运行程序的类和接口的二进制数据。图 1-3 是 JVM 类装载的过程。通过类载入程序, Java 系统可以确认被载入程序是否安全。

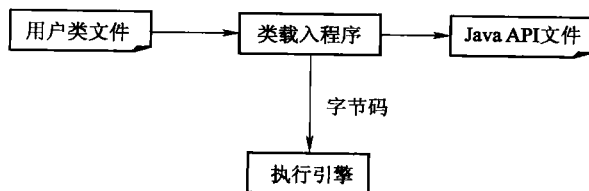


图 1-3 Java 系统的类载入过程

Java 1.1 采用的安全模型是信任模型, 用户可以有选择地授权给远程代码, 如果远程某地资源可以被信任, 那么带有此地签名的代码将被授权可以访问本地的系统资源, 其他不被信任的代码仍受限在沙箱内。Java 1.2 以后采用了域管理模型, 无论是本地代码或远程代码, 都可以通过配置来设定访问的资源, 从而更好地支持企业级应用。

1.4 Java 的不同版本

Java 具有 3 种不同的版本,各种版本适合在什么样的场合使用呢?下面将详细介绍这些版本。

Java Development Kit (JDK):JDK 是指 Java 开发工具包,是开发 Java 程序必需的软件工具。自从 Java 1.2 发布后,Java 改名为 Java 2,相应的 JDK 也根据应用领域的不同横向划分成 3 个版本,分别是 J2ME、J2SE 和 J2EE。2005 年,当 J2SE 的 JDK 版本纵向发展到 6.0 时,Sun 公司又把 3 个版本的 Java 分别更名为 Java ME、Java SE 和 Java EE。

- **Java SDK Micro Edition (Java ME)**:此版本用来开发掌上电脑、手机等移动通信设备上使用的应用程序。现阶段,不是所有移动设备都支持 Java,只有具备其运行环境的设备才能运行它。

- **Java SDK Standard Edition (Java SE)**:主要用于开发一般的台式机应用程序,平时说的 JDK 其实就是指 Java SE,本书也是围绕着它来讲述。

- **Java SDK Enterprise Edition (Java EE)**:用于开发分布式的企业级大型应用程序,其中的核心被称为 EJB (Enterprise JavaBeans)。

利用 Java 可以开发 Java 小应用程序 (Java applet)、Java 应用程序 (Java application)、服务器端小程序 (servlet) 和 JSP 程序 (JavaServer Pages) 以及其他类型的一些企业级应用程序。Applet 是嵌入在 HTML 文件中的 Java 程序,相当于嵌入在页面之中的脚本。Applet 的大小和复杂性是没有限制的,但由于 Internet 网速的限制,通常 applet 会很小。对于 Java 开发工具 (JDK) 而言,应用程序可以理解为从命令行运行的程序。Java 应用程序在最简单的环境中,它的唯一外部输入就是在启动应用程序时所使用的命令行参数。Servlet 和 JSP 都在服务器端运行,为 HTTP 服务提供动态的处理。所不同的是,servlet 是 Java 程序,而 JSP 是 HTML 文件里嵌入了 Java 代码。本书是基于 Java SE 的教程,Java SE 是开发各种类型的 Java 程序的基础。

1.5 Java 运行环境的建立

JDK (Java Development Kit) 与 Java SDK (Java Software Development Kit) 的含义通常是一样的,是 Java 的开发环境。Sun 公司的 Java SDK 是免费的工具,可以到 Sun 公司网站或提供相关下载的网站下载。目前,提供下载的 Java 标准版 (Java SE) 的最新版本为 JDK 6。

1. JDK 的下载与安装

首先,下载 JDK 6。Sun 的官方网站:<http://java.sun.com/javase/downloads/index.jsp>。

其次,下载代码编辑器,以便用于测试自己的安装和设置是否成功,可以选择 NetBeans、Eclipse 或是 UltraEdit32,当然也可以用“记事本”。

NetBeans 的下载地址:<http://netbeans.org/downloads>。