

Java 2 简明教程

(第2版)

- ◆ Java 语言概述
- ◆ 数据类型、运算符和表达式
- ◆ 控制语句
- ◆ Java 的类、接口和包
- ◆ 字符串处理
- ◆ 异常处理
- ◆ 输入与输出处理
- ◆ Java 多线程和小程序
- ◆ GUI 布局管理器
- ◆ 事件处理



皮德常 张凤林 编著



清华大学出版社

TP312
1338=2

高等院校计算机应用技术系列教材

Java 2 简明教程

(第 2 版)

皮德常 张凤林 编著

清华大学出版社

北 京

内 容 简 介

本书以 Java 2 语言为基础,详细介绍了面向对象的编程思想和方法。全书共 12 章,主要包括:Java 2 编程基础、面向对象编程原理、接口、包、字符串类 String 和 StringBuffer、异常处理、输入和输出、多线程、Java 小程序、GUI 布局管理、对象序列化、内隐类、Adapter 类和事件处理等,非常适合于 Java 初学者阅读。此外,本书还突出了 Java 语言与 C/C++ 的异同点,从而也非常适合于具有 C/C++ 编程经验,又想转向 Java 编程的读者阅读。

本书语言流畅,实例丰富,全部代码都在 JDK 5.0 运行环境下调试通过,并配有大量的习题,同时在 <http://www.tupwk.com.cn> 网站“资源下载”栏提供了该书的电子教案和程序示例源码。

本书特别适合于高等院校用作讲授 Java 2 编程语言和面向对象程序设计的教材。

版权所有,翻印必究。举报电话:010-62782989 13501256678 13801310933

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

本书防伪标签采用特殊防伪技术,用户可通过在图案表面涂抹清水,图案消失,水干后图案复现;或将表面膜揭下,放在白纸上用彩笔涂抹,图案在白纸上再现的方法识别真伪。

图书在版编目(CIP)数据

Java 2 简明教程/皮德常,张凤林 编著. —2 版. —北京:清华大学出版社,2006.7

(高等院校计算机应用技术系列教材)

ISBN 7-302-13282-8

I. J… II. ①皮… ②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(2006)第 070693 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:胡伟卷

文稿编辑:刘金喜

封面设计:王永

版式设计:康博

印刷者:北京嘉实印刷有限公司

装订者:三河市金元印装有限公司

发行者:新华书店总店北京发行所

开 本:185×260 印张:17 字数:392 千字

版 次:2006 年 7 月第 2 版 2006 年 7 月第 1 次印刷

书 号:ISBN 7-302-13282-8/TP·8377

印 数:1~5000

定 价:25.00 元

前 言

Internet 的发展极大地影响着计算机世界,同时也影响着人们的生活。目前,它已经成为全球范围最大和资源最为丰富的网络。美国的 Sun Microsystems 公司推出的 Java 语言是一种令人激动的新型语言,它具有面向对象、平台无关、可移植性强、安全、支持分布式等诸多特征,特别适合开发 Internet 程序,它能使网页中静态的图像和文本元素变得具有动感,栩栩如生。

网络是多台计算机的互联集合,Java 则是网络中通用的世界语。目前,Java 已经成为网络编程语言事实上的标准。从计算机程序设计语言的发展历史来看,Java 语言是在 C++ 语言之上推出的新一代语言,其语法与 C++ 的语法相似,但剔除了 C++ 中易于出错的成分。Java 增加了诸如内存自动回收之类的新功能,删去了 C++ 中令人费解的、不常用的成分,如运算符重载等。Java 语言自从 1995 年问世以来,很快流行于全世界,并获得了极大成功。目前,许多软件项目已选择 Java 语言作为其编程语言,特别是计算机网络方面的软件,如大型软件 ERP 有许多就是采用 Java 语言开发的。目前越来越多的人开始学习并使用 Java 语言,全世界已经掀起了一股 Java 热潮。

Java 2 的推出使 Java 的用途更加广泛,它适用于各种应用开发,尤其是网络应用和嵌入式系统开发。

本书的第一版受到了大量读者(高校教师、大学生、程序员等)的欢迎,他们在使用的过程中,给作者提出了一些很好的意见和建议,在此,作者表示深深的感谢。

本书特点

1. 抓住重要概念

Java 语言属于高等院校计算机相关专业的学生要学习的一门专业课,一般安排在 C/C++ 课程之后学习,理论教学的学时往往比较少,本科教学大多是 24 学时,专科教学是 40 学时。在如此少的学时内,教师不可能讲授完 Java 2 的所有知识,只能抓住重点内容进行介绍。针对这种情况,本书突出了 Java 与 C/C++ 的不同之处,对这些语言的共性不作详细介绍,仅仅讲授 Java 2 的核心内容,这不但便于读者学习和掌握,同时也便于教师讲解。

本书将要介绍的主要知识点如下。

- 类:类是面向对象程序设计中的重要概念。本书从第 4 章起便开始介绍类及其应用。
- 字符串:Java 将字符串作为一种对象进行处理,这与 C/C++ 不同。本书单独开辟一章讲解用于处理字符串的类 String 和 StringBuffer,并对各种处理方式的特点、性质和处理方法进行了分析。

- 异常处理：采用 Java 语言编写的系统不会因几个较小的错误而导致整个系统崩溃，因而异常处理是系统设计中不得不考虑的一个因素。
- 输入和输出处理：为了支持网络数据读写，Java 提供了许多数据输入和输出类。本书分析和讲解了其中较常用的类，并着重介绍了对象序列化的知识。
- 多线程：多线程可以使程序并行执行，从而提高对系统资源的利用率。本书重点介绍了如何设计多线程程序。
- 事件处理：事件处理是图形用户界面程序设计的核心。本书深入分析了 Java 委托事件处理思想、组件事件、鼠标事件和键盘事件，并给出了 3 个比较大的示例。

2. 舍弃次要内容

考虑到 Java 语言课程的学时有限，以及 Java 与 C/C++ 之间的关系，本书对 Java 开发工具的应用、基本数据类型、运算符、控制语句和类库，都没有作过多的讨论。例如，Java 中的基本数据类型、运算符及控制语句等内容与 C/C++ 类似，因而只是作了简单介绍；另外，由于 Java 的类库十分庞大，因此本书仅对常用的类库进行了介绍。

3. 力求培养学生的思考能力

本书就 Java 的一些实现技术进行了讨论和分析，并介绍了实现内幕。例如，作者结合自己的理解，分析了实例变量和方法的多态性问题(见第 4.9 节)；结合 String 和 StringBuffer 类，分析了“+”号操作的实现内幕(见第 6.2.3 节)。这些内容对培养学生的思考能力提供了一定的帮助，有助于培养他们勤于思考、勇于实践的能力。

4. 以最新的 J2SE 5.0 为标准

本书讲述的内容以 Sun 公司的 J2SE 5.0 为标准，所有程序均在 NetBeans 4.1 集成开发环境下运行通过。

5. 突出 Java 与 C/C++ 的异同点

如前所述，Java 课程往往安排在 C/C++ 课程之后。为了便于读者对 Java 的理解和掌握，本书在内容编排上突出了 Java 与 C/C++ 的异同点，以免读者误解 Java 的知识点。

6. 力求通俗易懂

编写本书的目的是让读者通过自学或在教师的指导下，学会运用 Java 语言的核心要素，进行面向对象的程序设计。因此，本书围绕着如何进行 Java 编程展开。为了便于读者学习，作者力求使本书的语言通俗易懂，将复杂的概念采用浅显的语言来讲述，便于读者理解和掌握。

本书的编排特点

- 每章开始均点明本章要讲解的内容和学习要求。

- 每章结束时，都进行了小结，给出了该章内容的概括性描述，并对该章的知识点进行了归纳。
- 每章安排的习题都具有很强的操作性，读者可通过计算机进行练习。
- 书中重要的内容采用黑体标记，特别重要的内容采用下面加点的方式标记。
- 本书强调程序的可读性。书中的程序全部采用统一的程序设计风格。例如，类名、方法名和变量名的定义做到“望名知义”；语句的末尾或下一句的开头放上左大括号，而右大括号自成一行，并采用缩排格式组织程序代码；此外，对程序中的语句还尽可能多地进行了注释。希望读者在编程时模仿这种程序设计风格。
- 本书强调程序的可移植性，不以某个 Java 开发工具为标准，而是以 Sun 公司提供的最新 J2SE 5.0 为标准。
- 本书包含了大量的程序示例，并给出了运行结果。凡是程序开头带有编号的，都是完整的程序，可以直接在计算机上编译运行。
- 本书采用了醒目的标记来显示知识点。这些标记包括“注意”、“警告”和“思考”等，它们穿插在全书中，能帮助读者尽快找到重要的信息。这些标记的含义如下。
 - **警告**：这是警告信息，它们往往是容易混淆的知识点。
 - **注意**：值得关注的地方，其级别略次于警告。
 - **思考**：提出问题，引导读者进行思考，以培养读者的独立思考能力。

教学支持

本书的电子教案是采用 PowerPoint 2000 制作的，可以在讲课时用多媒体投影演示，这可部分取代板书。教师不仅可以使使用本教案，还可以方便地修改和重新组织其中的内容以适应自己的教学需要。使用本教案可以减少教师备课时编写教案的工作量，以及因板书所耗费的时间和精力，从而提高单位课时内的知识含量。

我们向使用本教材的教师免费提供本书的电子教案和 116 个程序示例源码，它们的下载网址为 <http://www.tupwk.com.cn/downpage>。需要本书习题参考答案的教师请发邮件至 cwkbook@tup.tsinghua.edu.cn，邮件的主题请设为“获取《Java 2 简明教程(第 2 版)》的参考答案”。为了更好地为您服务，请在邮件中附上姓名、工作单位、地址、联系电话、主讲课程等信息。

感谢读者选择本书，欢迎对本书的内容提出批评和修改建议，作者将不胜感激。作者的联系地址如下。

电子邮件地址：dc.pi@nuaa.edu.cn

通信地址：江苏省南京市南京航空航天大学信息科学与技术学院 皮德常（收）

邮政编码：210016

作 者
2006 年 3 月

目 录

第 1 章 Java语言简介	1
1.1 Java 语言的发展	1
1.2 Java 语言的特点	2
1.2.1 简单性	3
1.2.2 面向对象	3
1.2.3 分布性	3
1.2.4 解释执行	4
1.2.5 健壮性	4
1.2.6 安全性	4
1.2.7 结构中立	5
1.2.8 可移植性	5
1.2.9 高效性	6
1.2.10 多线程	6
1.2.11 动态性	6
1.3 Java 类库的概念	6
1.4 网络浏览器	7
1.5 Java 开发工具	8
1.6 Java 程序分类	9
1.6.1 使用 NetBeans 运行 Java 应用程序	9
1.6.2 使用 NetBeans 运行 Java 小程序	10
1.7 对 Java 程序的解释	11
1.7.1 程序注释方法	11
1.7.2 对 Java 应用程序的解释	11
1.7.3 对 Java 小程序的解释	13
1.7.4 对 HTML 文件的解释	13
1.8 编写 Java 程序的风格要求	14
1.9 本章小结	14
1.10 思考和练习	15
第 2 章 数据类型、运算符和表达式	17
2.1 常量	17

2.2 变量	18
2.2.1 整型变量	19
2.2.2 字符型变量	21
2.2.3 浮点型变量	21
2.2.4 布尔型变量	22
2.2.5 对原子类型变量生存空间 的讨论	22
2.3 变量赋值问题	23
2.4 数组	24
2.4.1 一维数组	24
2.4.2 二维数组	26
2.5 Java 中的参数传递方式	27
2.6 Java 的运算符	29
2.6.1 算术运算符	29
2.6.2 关系运算符	30
2.6.3 逻辑运算符	31
2.6.4 位运算符	32
2.6.5 三元条件运算符	33
2.6.6 “+” 运算符	33
2.7 本章小结	34
2.8 思考和练习	34
第 3 章 控制语句	37
3.1 分支语句	37
3.1.1 if 语句	37
3.1.2 switch 语句	40
3.2 循环控制语句	43
3.2.1 while 语句	43
3.2.2 do-while 语句	44
3.2.3 for 语句	45
3.3 break 语句和 continue 语句	46
3.3.1 不带标号的 break 语句 和 continue 语句	46

3.3.2 带标号的 break 语句和 continue 语句.....	46	4.14 抽象类和抽象方法.....	85
3.4 本章小结.....	48	4.15 对象的类型转换.....	87
3.5 思考和练习.....	49	4.15.1 向上类型转换.....	88
第 4 章 Java 的类	51	4.15.2 向下类型转换.....	89
4.1 类与对象.....	51	4.16 访问权限限制.....	89
4.1.1 类与对象的区别.....	51	4.16.1 友元.....	90
4.1.2 Java 和 C 编程思想的区别	52	4.16.2 public 成员.....	91
4.1.3 如何定义类.....	52	4.16.3 private 成员.....	91
4.1.4 对象和引用.....	53	4.16.4 protected 成员.....	92
4.2 方法.....	55	4.17 应用程序从键盘输入 数据举例.....	93
4.3 实例变量和局部变量.....	56	4.18 本章小结.....	94
4.4 构造函数.....	58	4.19 思考和练习.....	95
4.5 方法重载.....	60	第 5 章 接口和包	97
4.6 关键字 this.....	61	5.1 接口.....	97
4.6.1 指代对象.....	62	5.1.1 接口的定义和应用.....	97
4.6.2 指代构造函数.....	64	5.1.2 接口和抽象类的异同点.....	103
4.7 继承.....	65	5.2 包.....	103
4.7.1 继承的概念.....	65	5.2.1 package 语句.....	103
4.7.2 关键字 super.....	67	5.2.2 import 语句.....	104
4.7.3 再论构造函数.....	68	5.2.3 包应用举例.....	105
4.8 方法的覆盖.....	69	5.3 本章小结.....	108
4.8.1 覆盖与重载的区别.....	70	5.4 思考和练习.....	108
4.8.2 方法的动态调用.....	72	第 6 章 字符串处理	109
4.9 一个令人迷惑的问题: 多态性不适合于继承链中的 实例变量.....	74	6.1 字符串的分类.....	109
4.10 finalize.....	75	6.2 String 类.....	109
4.11 static.....	77	6.2.1 字符串常量.....	110
4.11.1 static 变量.....	77	6.2.2 创建 String 类对象.....	111
4.11.2 static 方法.....	79	6.2.3 String 类的常用方法.....	114
4.12 关键字 final.....	81	6.2.4 Java 应用程序的 命令行参数.....	121
4.12.1 final 数据.....	81	6.3 StringBuffer 类.....	122
4.12.2 final 方法.....	82	6.3.1 创建 StringBuffer 类对象.....	122
4.12.3 final 类.....	82	6.3.2 StringBuffer 类的常用方法.....	123
4.13 组合与继承.....	83		

6.3.3 String 类中“+”操作的 技术内幕·····	127
6.4 应用举例·····	128
6.5 本章小结·····	131
6.6 思考和练习·····	131
第 7 章 异常处理·····	133
7.1 异常的层次结构·····	133
7.2 异常处理语句·····	135
7.2.1 try 和 catch 语句·····	136
7.2.2 finally 语句·····	138
7.2.3 throw 语句·····	139
7.2.4 throws 语句·····	140
7.3 自定义异常类·····	143
7.4 异常处理常用调试方法·····	144
7.5 本章小结·····	146
7.6 思考和练习·····	146
第 8 章 输入与输出处理·····	147
8.1 流的层次结构·····	147
8.2 File 类·····	148
8.3 InputStream 类和 OutputStream 类·····	150
8.3.1 InputStream 类的常用方法·····	150
8.3.2 OutputStream 类的常用方法·····	151
8.3.3 FileInputStream 类·····	151
8.3.4 FileOutputStream 类·····	153
8.3.5 DataInputStream 和 DataOutputStream 类·····	154
8.4 RandomAccessFile 类·····	156
8.5 对象流和对象序列化·····	160
8.5.1 对象流的概念·····	160
8.5.2 对象序列化·····	161
8.6 IOException 类·····	164
8.7 本章小结·····	165
8.8 思考和练习·····	165
第 9 章 多线程·····	167
9.1 Java 中的多线程的基本概念·····	167
9.2 线程类·····	168
9.2.1 多线程编程中常用的 常量和方法·····	168
9.2.2 线程的生命周期·····	169
9.2.3 创建多线程的方法·····	170
9.3 资源的协调与同步·····	174
9.3.1 线程调度模型·····	174
9.3.2 资源冲突·····	175
9.3.3 同步方法·····	177
9.4 线程间通信·····	178
9.4.1 通过封装共享变量实现 线程通信·····	178
9.4.2 通过系统方法实现 线程通信·····	180
9.5 本章小结·····	183
9.6 思考和练习·····	184
第 10 章 小程序·····	185
10.1 小程序的基本知识·····	185
10.1.1 小程序与应用程序 的区别·····	185
10.1.2 小程序标签的语法格式·····	186
10.2 小程序的生命周期·····	188
10.3 小程序常用方法·····	191
10.3.1 常用的输出方法·····	191
10.3.2 输出中的颜色控制·····	193
10.4 常用组件·····	195
10.4.1 组件和容器的关系·····	196
10.4.2 按钮·····	197
10.4.3 标签·····	198
10.4.4 文本框·····	199
10.4.5 文本域·····	201
10.4.6 选择框·····	203
10.4.7 下拉列表·····	205
10.4.8 列表·····	207

10.5	本章小结	209	12.2.1	JButton 事件处理	230
10.6	思考和练习	209	12.2.2	JTextField 和 JPassword Field 事件处理	232
第 11 章	GUI 布局管理器	211	12.2.3	JCheckBox 和 JRadio Button 事件处理	234
11.1	Swing 常用容器	211	12.2.4	JComboBox 事件处理	238
11.1.1	框架	211	12.2.5	JList 事件处理	240
11.1.2	面板	213	12.3	鼠标事件处理	242
11.2	布局管理器	215	12.4	Adapter 类	246
11.2.1	FlowLayout 布局	215	12.5	键盘事件处理	247
11.2.2	BorderLayout 布局	217	12.6	事件处理综合应用举例	250
11.2.3	GridLayout 布局	220	12.6.1	舞动的字符	250
11.2.4	CardLayout 布局	222	12.6.2	播放声音剪辑	253
11.3	本章小结	226	12.6.3	网络浏览器	255
11.4	思考和练习	226	12.7	本章小结	258
第 12 章	事件处理	229	12.8	思考和练习	258
12.1	委托事件处理模型	229	参考文献		259
12.2	组件事件处理	230			

第1章 Java语言简介

Java 语言是由美国 Sun Microsystems 公司开发的一种程序设计语言，现在已经成为 Internet 的主力开发语言。它采用了面向对象技术，具有支持分布式、安全、结构中立、可移植性强和多线程等特点，现在已经成为网络编程的首选语言。

Java 是从 C++ 发展而来的，它与 C++ 类似，但比 C++ 简单。熟悉 C++ 的读者可以很容易地掌握 Java 的语法格式，但更要注意二者的区别。

本章的学习目标：

- 了解 Java 语言的发展历程
- 掌握 Java 语言的特点
- 了解 Java 语言的开发工具
- 了解 Java 应用程序和小程序
- 掌握 Java 程序的注释方法
- 掌握 Java 程序的编写规范

1.1 Java 语言的发展

1991 年 Sun Microsystems 公司成立了一个名为 Green 的项目开发小组，负责人是 Jame Gosling，该小组主要开发面向家电的编程软件。1991 年 6 月该小组就开发了新的语言，当时命名为 Oak，后来改名为 Java。

Jame Gosling 在设计 Java 时采用了虚拟机代码(Virtual Machine Code)，即.class 文件，它通过解释器运行。如果每一台计算机上都安装一个解释器，那么这种程序就能实现与计算机的操作系统平台无关。

1994 年，Internet 的发展如火如荼，Jame Gosling 意识到它需要一种不依赖于任何硬件和软件平台的中性浏览器。事实确实如其所想，网络浏览器确实是中性的。

随着 Internet 的发展，1995 年 5 月 Sun Microsystems 公司对外正式发布 Java 1.0。目前 Sun Microsystems 公司推出的标准版 J2SE、企业版 J2EE 和移动版 J2ME，使 Java 具有更广泛的用途，它适用于各种应用开发，尤其是网络应用、嵌入式系统和移动系统。

从程序设计语言发展史看，Java 语言是建立在 C++ 语言之上的，图 1-1 显示了程序设计语言的发展历程。

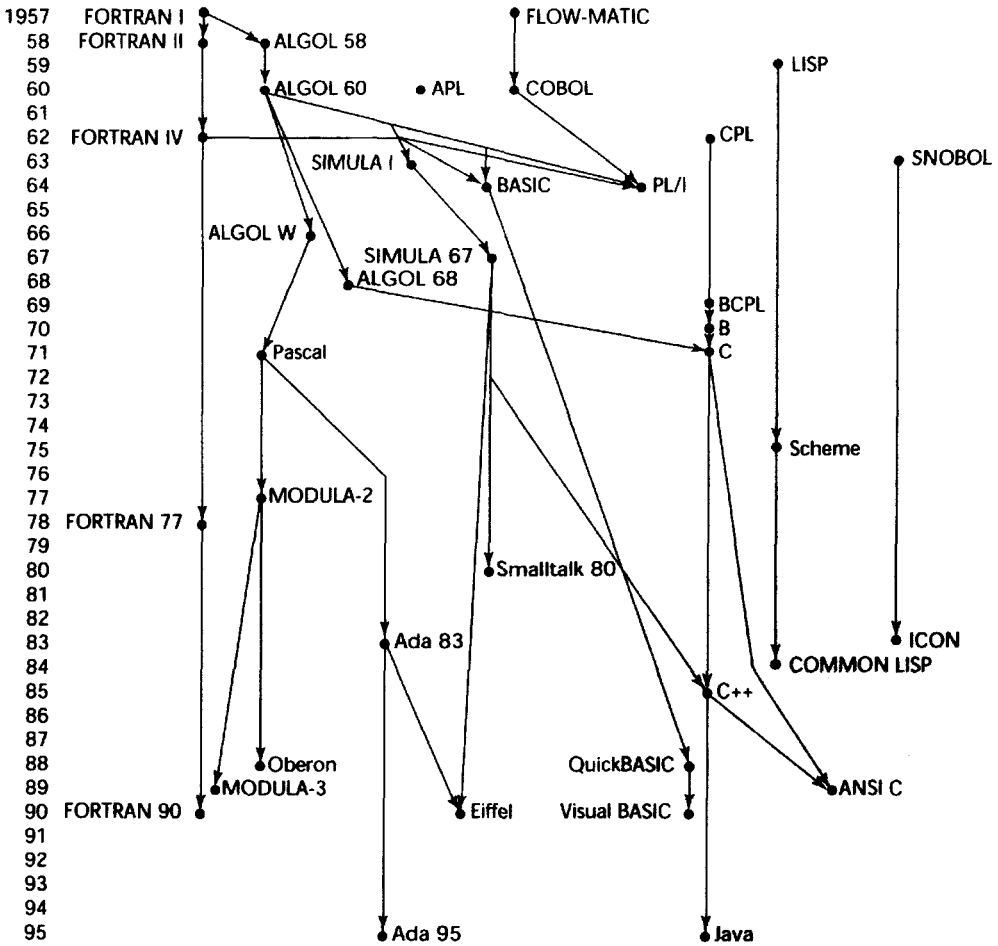


图 1-1 程序设计语言的发展历程

1.2 Java 语言的特点

Sun Microsystems 公司提供的 Java 白皮书(可从该公司网站 [http:// java.sun.com/](http://java.sun.com/)下载)中是这样描述 Java 语言的:

“Java: A simple, object-oriented, distributed, interpreted, robust, secure, architecture-neutral, portable, high-performance, multi-thread and dynamic language.”

翻译过来就是“Java 语言是一种简单的、面向对象的、分布式的、解释执行的、健壮的、安全的、结构中立的、可移植的、高效的、多线程的、动态的语言”。从此定义可以知道, Java 语言具有许多突出的特点, 下面分别给予介绍。

1.2.1 简单性

Java 语言的简单性体现在以下两个方面:

- Java 与 C/C++保持了一定的兼容性, 与 C++类似, 但比 C++简单。
- Java 取消了 C/C++中很少用的、难以理解的、容易混淆的特性。例如, Java 不支持 goto 语句, 而采用带标号的 break 和 continue 语句以及异常处理; 不支持头文件和预处理; 取消了类型自动转换、运算符重载和多继承等; 取消了结构体类型和指针类型; 取消了内存空间的动态申请与释放, 增加了内存空间的自动回收功能, 以管理废弃的内存。

Java 的缔造者之一 Bill Joy 在一场关于 Java 的演讲中说: “Java=C++ - -”。意思就是: Java 是“移除累赘和难于理解的部分之后的 C++”。因此 Java 是一个更为纯粹的、面向对象的程序设计语言。若学过 C++和 Java, 可以发现 Java 确实要比 C++简单许多。

1.2.2 面向对象

Java 支持面向对象(object-oriented, 简称 OO)的程序设计, 这与 C 语言不同, C 是一种面向函数(过程)的程序设计语言。Java 以类(class)的形式来组织程序, 并且还硬性规定: 所有类的根结点都是 Object 类, 其余的类都是其子类。

既然 Java 采用面向对象的思想进行程序设计, 显然它支持继承性, 这保证了代码复用。Java 仅支持单继承, 即一个类只能有一个父类, 这一点不同于 C++; C++支持多继承, 即一个类可以有多个父类。多继承虽然可以实现强大的功能, 但不易理解, 因此 Java 中取消了多继承, 但是 Java 提供了另一种功能: 支持多接口(interface)(在第 5 章介绍)。

运算符重载是 C++的一个特点, Java 取消了该功能, 但为了输出方便, 保留了“+”符号的重载能力, 用于连接各种类型的数据。

Java 不单是一个面向对象的程序设计语言, 除了支持对象数据类型外, 还支持一些基本数据类型, 如整型、实型、字符型和布尔型等。当然, 也不能苛求 Java, 因为这 4 种类型是程序设计中最常用的数据类型, 若将它们设计成对象类型, 使用起来未必方便。从程序设计语言原理的角度讲, 一种语言要在市场上占有一席之地, 它必定要迎合用户的口味和需求, 否则将被淘汰。

由于 Java 采用了面向对象的思想组织程序, 所以支持 OO 的 3 个基本特性, 即: 封装性、多态性和继承性(见第 4 章)。

1.2.3 分布性

Java 语言的应用程序编程接口具有支持 HTTP 和 FTP 等 TCP/IP 协议的类库, 这样, Java 应用程序可以通过 URL 地址直接访问网络上的对象, 就和访问本地对象一样。

1.2.4 解释执行

Java 程序的执行方式比较特殊,类似过去的数据库管理系统 FoxBase+的实现方式, FoxBase+是将源程序 .prg 文件编译生成 .fox 文件,然后解释执行。Java 采用的思想与此类似,即先编译后解释执行。具体顺序如下:

(1) 采用编辑器编写代码并保存。例如,采用文本编辑器编写代码,并保存为 .java 文件。

(2) Java 编译器对 .java 源文件进行编译,生成一种称为 .class 的字节码(Byte Code)文件。

(3) Java 装载器将 .class 的字节码文件装入内存。

(4) Java 字节码检验器对字节码进行安全检验,若其不违背 Java 的安全性,将继续进行,否则停止执行。

(5) Java 解释器解释执行字节码。

在第(2)步生成的 .class 文件就是字节码文件,Java 将其称为虚拟机代码。Java 这种先编译后解释运行方式的优点是:字节码是一种与平台无关的文件格式,可以在不同的平台上传输和运行。其缺点也比较明显,尽管执行速度比典型的解释程序(如 Basic)快,但比纯编译方式的语言,如 C/C++,要慢很多倍,一般是 8~20 倍。

注意:

目前最新的 Java 版本是 5.0 (即是 JDK1.5,也称 5.0) 已经和 NetBeans 4.1 捆绑,通过 NetBeans 这个集成环境可以编辑、编译、生成和运行 Java 程序。

1.2.5 健壮性

Java 语言的健壮性主要表现在两个方面,一方面是 Java 取消了指针,另一方面是 Java 引入了异常(Exception)处理机制。C/C++中的指针可以直接操作硬件地址端口,还可以修改指定内存中的内容,同时内存空间的申请和释放也必须由程序员负责。Java 取消了指针,程序只能访问有限的内存资源,并且具有严格的内存保护机制。此外,Java 还引入了动态的内存分配技术和垃圾回收功能(这是 C/C++所不具备的功能),从而减少了程序员的负担。

Java 还吸收了著名的 Ada 语言优秀的一面,引入了异常(Exception)处理机制,程序员可以编写相应的程序处理代码,不至于因几个错误而导致整个系统崩溃。

1.2.6 安全性

Java 语言的安全性主要表现为以下 4 个方面:

- 语言结构设计严谨,对象的方法和变量具有 public、protected、private 和友元这几种不同的保护机制,并且规定 final 类不能被其他类继承。

- C/C++中的指针是安全代码设计中的一大隐患, Java 取消了指针, 从而提高了系统的安全性。
- 字节码文件(即.class 文件)附带有一些安全检验信息, 字节码检验器依此信息进行安全检验, 从而可以尽早发现程序是否违背安全性原则。
- 浏览器在运行.class 文件时, 也要对其进行安全检验。

通过层层严格把关, 保证了 Java 程序的安全。

1.2.7 结构中立

在最初设计 Java 语言时, 设计师们就考虑到了多种不同的计算机平台, Java 要在不同的平台上都具有生命力, 必须采取一种中性结构, 这主要表现为以下两个方面。

- 字节码的中介方式, 与运行平台无关。
- 与 C++相比, Java 语言定义严格。例如, 在类中定义的两个数据成员:

```
class myclass {
    int a;           // 在 C++中, a 的值依赖于构造函数
    int b=1;        // 在 C/C++中, 对数据成员直接赋值是非法的
                  // 其他数据成员和方法略
}
```

在 Java 中, a 和 b 都是 4 个字节, a 的初值是 0, b 的初值是 1。与 C++比较, 这样的定义比较严格, 避免了与实现平台有关。

1.2.8 可移植性

与其他语言程序相比, 用 Java 语言编写的程序可移植性比较高, 这是由下列特性决定的。

- Java 语言定义严格, 结构中立。
- Java 提供的类库, 不论对哪一种操作系统, 如 Windows NT、UNIX 或者是 Macintosh 等都一样。
- 每种基本类型的变量所占的空间大小在 Java 中是确定不变的。它们的大小不会像其他程序设计语言那样“随运行平台而定”。例如: int 类型的变量, 不管是在何种类型的平台上, 都占 4 个字节, 而 C 语言的 int 类型变量, 在 DOS 平台上占两个字节, 但在 UNIX 平台上占 4 个字节。

上述特性保证了 Java 程序具有比较好的可移植性。当然, 这一点也不是万能的, 有时因编译器、解释器和计算机的差异, 无法保证一个 Java 程序不做任何修改即可从一台计算机直接迁移到另一台计算机。

1.2.9 高效性

为了提高 Java 程序的执行效率,其编译器先将程序编译为与机器指令非常接近的字节码。这个特点具有一定的相对性,和完全解释执行的 Basic 程序相比具有优势,但和采用编译方式执行的 C/C++程序相比则不具有优势。

1.2.10 多线程

进程和线程是操作系统中两个重要的基本概念。进程(process)在执行过程中有自己独立的内存空间和系统资源,各个进程的内存数据和状态彼此孤立,交换数据通过特定的通信机制完成,如管道。线程(thread)是在进程中产生的一种轻负荷进程(light weight process),线程在执行过程中共享一块内存空间和一组系统资源,因此线程之间可以直接进行数据交换。

Java 真正支持多线程,C/C++等语言都不支持多线程。有些读者可能采用 C++编写过多线程的程序,但 C++中的多线程能力,实际上是通过调用操作系统的多线程机制实现的。为了避免因资源冲突导致系统死锁,Java 引入了同步关键字 `synchronized`,用于指定某个方法或某个对象不能被并发执行。引入多线程显然提高了程序的工作效率。

1.2.11 动态性

Java 程序的基本构成单元是类,即 Java 程序必须写在类中,这就像 C/C++程序必须写在函数中一样。Java 的类是在运行时动态加载的,不影响程序的运行。

1.3 Java 类库的概念

Java 程序由类构成,一个 Java 程序可以写在若干个类中。用户在编写程序的过程中要充分利用系统提供的类和方法。学习 Java 语言实际上包括了两个方面:一是学习用 Java 语言编写自己所需的类,另一个是学习如何利用 Java 类库中的类和方法。这样做具有如下优点:

- 采用已有的类库编程,可以避免一些从头开始的编程工作。在软件工程中,使用现有的构件称为软件重用,这是面向对象程序设计中的一个重要思想。
- 利用类库编程可以提高程序运行的性能。类库中的类和方法都是经过严格检验的,无论是质量还是效率都比较高。
- 利用类库编程,可以提高程序的可移植性。因为这些类和方法包含在适合所有平台的 Java 版本中。

下面举例阐述 Java 和 C/C++ 的异同点。首先介绍 C 程序，对于一个 C 程序，其结构往往如下：

```
#include "stdio.h" /* 这是 C 中包含头文件的常用方法 */

void fun ()      /* 一个 C 程序往往由若干个函数构成*/
{
    .....
}

void main ()
{
    fun ();
}
```

上述 C 程序由两个函数构成，fun 和 main 函数共同构成了一个程序整体。而一个 Java 程序往往是这样的：

```
import java.awt.*; // 引用 Java 类库的方法

class myclass{     // 用户定义的第一个类
    void fun (){
        System.out.println("Hello Java!");
    }
}

public class Class1{ // 用户定义的第二个类，Java 程序必须写在类中
    public static void main(String args[]){
        myclass obj;

        obj=new myclass ();
        obj.fun();
    }
}
```

通过上述程序可以看到 Java 引用类库的方法。读者目前可能不了解该程序，随着学习的深入，将会逐渐掌握这种编程方法。

1.4 网络浏览器

Java 程序在支持 Java 的网络浏览器上才能运行，支持 Java 的网络浏览器很多，如 HotJava、Netscape Navigator 和 Microsoft 公司的 Internet Explorer 等。目前市场上的计算机系统基本上都能满足这个条件。