

J A V A

系 列 丛 书



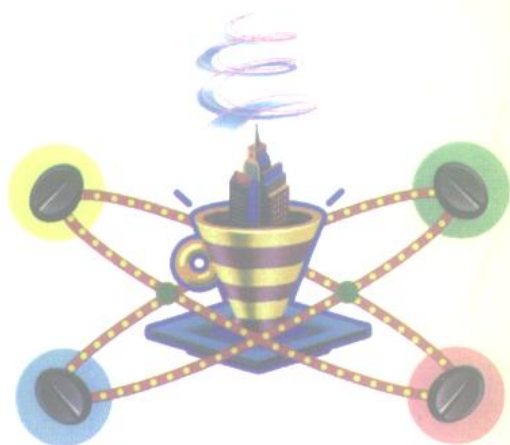
# JAVA 语言

## SQL 接口

—— JDBC 编程技术

王克宏 主编

丁 锂 孙 元 编著



清 华 大 学 出 版 社

Java 系列丛书之七

# Java 语言 SQL 接口

## —— JDBC 编程技术

王克宏      主编  
丁 锂 孙 元 编著

清华大学出版社

(京)新登字 158 号

## 内 容 简 介

JDBC(Java DataBase Connectivity)是 Java 语言与数据库互联的接口。通过 JDBC 可以使 Java 应用程序或 Java applet 访问网络上的各种数据库。本书首先介绍了 JDBC 的概念、结构和特点。然后逐步讲解了如何编写 JDBC 程序。最后给出了 JDBC API 的详细说明。附录中给出了 JDBC 的获取与安装的方法。

本书覆盖了 JDBC 的各个方面,适合于有一定的 Java 语言编程基础和数据库及 SQL 语言基础的开发人员使用。

版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无防伪标签者不得销售。

JS193/07

### 图书在版编目(CIP)数据

Java 语言 SQL 接口: JDBC 编程技术/孙元编著. —北京:清华大学出版社,1997 年 .4

(Java 系列丛书/王克宏主编)

ISBN 7-302-02489-8

I . J… II . 孙… III . Java 语言-接口程序-程序设计 IV . TP312

中国版本图书馆 CIP 数据核字(97)第 10319 号

出 版 者:清华大学出版社(北京清华大学校内,邮政编码:100084)

责任编辑:徐培忠

印 刷 者:人民文学印刷厂

发 行 者:新华书店总店北京科技发行所

开 本:787×1092 1/16 印张:10.75 字数:224 千字

版 次:1997 年 4 月第 1 版 1997 年 8 月第 2 次印刷

书 号:ISBN 7-302-02489-8/TP·1260

印 数:8001~14000

定 价:16.00 元

# 序 言

近年来 Internet 以迅猛之势发展,已成为全世界规模最大的计算机网络,网上资源丰富,为世界各国千万用户所瞩目,我国也已于 1994 年 5 月正式入网,并已有成千上万使用者上网工作,如何充分、合理地利用 Internet 的网络与信息资源,为我国社会的各方面服务,是我们应予以重视和研究的问题。

Java 的出现正迎合了 Internet 发展形势的需要,它所体现的简单、不依附于平台、面向对象、分布式、可靠性、安全性、可移植性、动态性、多线程等特性,为 Internet 的使用提供了一种良好的开发和运行环境,成为 Internet 适用、新型的编程语言。Java 出现后立即为世界各大公司所注目,纷纷购买 Java 使用权,并在剧烈的市场竞争中展开了大规模的研究与产品开发。Java 语言及其相关技术同样也引起了我国学术界、产业界和应用界的广泛关注和兴趣。因此 Java 丛书的出版发行,是符合时宜之举,必将受到大家的欢迎。

根据编著者的计划,Java 丛书将包括:《Java 语言入门》,《Java 虚拟机规范》,《Java 语言编程技术》,《Java 语言 Applet 编程技术》,《Java 语言 API 类库》,《Java 软件包的使用》,《Java 语言 SQL 接口》,《Java 语言调试技术》及《HotJava 使用指南》等十多部书,达几百万字。详细深入地介绍 Java 语言及其相关技术并在短期内出齐,为我国广大读者研究和掌握 Java 提供了一种系统而全面的文献资料,无疑是一种十分有意义的事。Java 作为新生事物,尚在不断发展完善之中,因此我希望我们对 Java 及其相关技术不只停留于被动的学习、跟踪与使用,还应以积极主动的态度,通过应用实践和深入分析研究,参与开发创新,为计算机的网络应用做出我们自己的贡献。

清华大学计算机科学与技术系王克宏教授及其知识工程科研组,在 Internet 上进行多层次的研究开发工作已有一段时间,取得了阶段性成果,并曾在不久前召开的“中国计算机学会第九次全国学术大会”的全体会议上作报告,受到与会者的欢迎。相信他们在结合自己研究工作基础上编写的这套 Java 系列丛书,必将有助于我国信息科学技术的发展。我高兴地为丛书作序并祝他们成功!

张淑祥  
1996.8

\* 本文作者为中国科学院院士、中国计算机学会理事长

# 开辟 Java 计算的新时代

Sun 公司发明的 Java 语言风靡全球。据 Forrester Research 公司对《幸福》杂志评出的世界 1000 家大型企业进行的调查,目前在它们中间已有 62%正在使用 Java 从事开发工作;有 42%已将 Java 纳入自己企业一年内的战略开发计划。这个调查结果表明,Java 已为大家所接受。

计算机自从 50 年前诞生以来,经历了几个发展时代:终端/主机计算、PC 计算、客户机/服务器计算,而现在已发展到一个新的阶段,即 Java 计算时代。这个时代的特点就是以 Java 为代表的网络计算。

Java 带来的是一场革命。这是第一个真正独立于平台的计算方案,它能充分发挥 Internet 的作用。Java 计算是实现“一个世界,一个网络”构想的关键。这样一个透明的、全球连接的和信息交换的网络,可将所有最新的计算技术、电话、出版/媒体和娱乐集成一体。过去妨碍这一构想成为现实的是计算平台不能兼容,而 Java 语言却以其许多优秀特性使之成为允许各类系统相互兼容和共享应用环境的桥梁。这使各类软件“一次写成,到处可用(write once/run anywhere)”,这样,相同的软件可在不同器件上运行,无论是 PC 机、苹果机、UNIX 计算机,还是顶置盒、PDA(个人数据助理)、移动电话,乃至智能元器件,无一例外。Java 的这一贡献将使全人类受益,因而给人们的生活方式带来极大的改变。

清华大学王克宏教授是最早研究 Java 语言的中国学者之一。在他的周围,许多热心于 Java 计算的青年学者、博士生、硕士生和本科生组成了一支朝气蓬勃、积极进取的生力军,他们思想活跃、工作勤奋、通过努力把 Java 语言的开发和应用提高到一个新的水平。据我所知,在中国的诸多大专院校和科研院所,包括北京航空航天大学、上海复旦大学,也都有许多学者致力于 Java 的研究和开发。由于他们的辛勤努力和积极探求,在 Java 平台上开发和使用的 Java 应用将给中国的信息业带来新的发展,使 Java 计算在中国进一步得到推广。

《Java 系列丛书》就是 Java 开发园中的一朵奇葩,它不消不争,以其朴实无华、实用有效的风格展示于人。我们希望由王克宏教授主编的该套丛书能赢得广大读者的喜爱,希望能有更多的中国智者和用户在 Java 计算的时代熟悉 Java,掌握 Java,利用 Java 的诸多优点,编写出更多更美的应用软件,从而造福于人类。

谨以此愿祝贺 Java 系列丛书的出版发行。

李永超

1996.10

\* 本文作者为 Sun 公司中国地区副总裁

• I •

# 前 言

本书讲解了 Java 语言与数据库互联的接口——JDBC 的基本概念、结构以及编程方法。JDBC 是 Java 语言为支持基本的 SQL 功能而提供的一个独立于特定的 DBMS 的用户接口。使用 JDBC 编写的 Java 应用程序或 Java Applet 可以实现对分布在网络上的不同的数据库进行访问。使用 JDBC 编写程序可以使程序员将精力集中于上层功能的实现,而不必关心底层的与具体的 DBMS 的连接和访问过程。

通过本书可以学习如何编写 JDBC 程序,实现与数据库的连接、读写及显示结果等操作。本书还给出了 JDBC API 的详细说明,可以作为程序员的手册来使用。

本书内容的组织结构如下:

第 1 章是 Java 语言概述,帮助读者回顾一下 Java 语言。

第 2 章是 SQL 语言概述,简单地讲解了 SQL 语言的语法及其功能。

第 3 章讲解了 JDBC 的概念、结构和特点。

第 4 章详细讲解了如何编写 JDBC 程序。

第 5 章给出了 JDBC API 中的接口的详细说明。

第 6 章给出了 JDBC API 中的类库的详细说明。

附录 A 给出了如何获取与安装 JDBC 及其驱动程序,并给出了对应不同数据库的驱动程序的列表。

附录 B 给出了 JDBC1.01 版到 JDBC1.1 版的改变。

# 目 录

|              |                               |    |
|--------------|-------------------------------|----|
| <b>第 1 章</b> | <b>Java 语言概述</b>              | 1  |
| 1.1          | Java 的背景                      | 1  |
| 1.2          | Java 的特点                      | 2  |
| 1.3          | 简单的 Java 程序                   | 5  |
| 1.4          | Java 的应用远景                    | 9  |
| <b>第 2 章</b> | <b>SQL 概述</b>                 | 10 |
| 2.1          | 关系数据库模型                       | 10 |
| 2.2          | SQL 语言简介                      | 10 |
| 2.3          | 嵌入式 SQL (Embedded SQL)        | 18 |
| <b>第 3 章</b> | <b>JDBC 概述</b>                | 20 |
| 3.1          | JDBC 的特点                      | 20 |
| 3.2          | JDBC 的两种主要接口                  | 22 |
| 3.3          | 使用方法                          | 23 |
| 3.4          | 安全性问题                         | 25 |
| <b>第 4 章</b> | <b>编写 JDBC 程序</b>             | 29 |
| 4.1          | 建立数据源                         | 29 |
| 4.2          | 与数据库建立连接                      | 35 |
| 4.3          | 执行查询语句                        | 39 |
| 4.4          | 检索结果集                         | 42 |
| 4.5          | 更新数据库操作                       | 50 |
| 4.6          | 参数的输入和输出                      | 55 |
| 4.7          | 动态数据库访问                       | 59 |
| 4.8          | JDBC 和 SQL 的数据转换              | 67 |
| 4.9          | JDBC 中的意外                     | 71 |
| 4.10         | JDBC 中的其他问题                   | 73 |
| <b>第 5 章</b> | <b>JDBC API 接口</b>            | 77 |
| 5.1          | java.sql.CallableStatement 接口 | 78 |
| 5.2          | java.sql.Connection 接口        | 81 |

|              |                                     |            |
|--------------|-------------------------------------|------------|
| 5.3          | java.sql.DatabaseMetaData 接口 .....  | 85         |
| 5.4          | java.sql.Driver 接口 .....            | 115        |
| 5.5          | java.sql.PreparedStatement 接口 ..... | 116        |
| 5.6          | java.sql.ResultSet 接口 .....         | 121        |
| 5.7          | java.sql.ResultSetMetaData 接口 ..... | 129        |
| 5.8          | java.sql.Statement 接口 .....         | 132        |
| <b>第 6 章</b> | <b>JDBC API 类库 .....</b>            | <b>136</b> |
| 6.1          | java.sql.Date 类 .....               | 136        |
| 6.2          | java.sql.DriverManager 类 .....      | 137        |
| 6.3          | java.sql.DriverPropertyInfo 类 ..... | 139        |
| 6.4          | java.sql.Numeric 类 .....            | 140        |
| 6.5          | java.sql.Time 类 .....               | 147        |
| 6.6          | java.sql.Timestamp 类 .....          | 148        |
| 6.7          | java.sql.Types 类 .....              | 149        |
| 6.8          | java.sql.DataTruncation 类 .....     | 151        |
| 6.9          | java.sql.SQLException 类 .....       | 153        |
| 6.10         | java.sql.SQLWarning 类 .....         | 154        |
| <b>附录 A</b>  | <b>JDBC 及其驱动程序的获取与安装 .....</b>      | <b>156</b> |
| A.1          | JDBC 的获取与安装 .....                   | 156        |
| A.2          | JDBC 文档资料的获得 .....                  | 157        |
| A.3          | JDBC-ODBC Bridge 的获取与安装 .....       | 157        |
| A.4          | 其他 JDBC 驱动程序的获取 .....               | 158        |
| <b>附录 B</b>  | <b>JDBC1.01 版到 1.1 版的变化 .....</b>   | <b>161</b> |



# 第 1 章 Java 语言概述

## 1.1 Java 的背景

### 1.1.1 Java 语言出现的背景

在过去的一年多,计算机界最热门的话题大概要算 INTERNET,而当前 INTERNET 上最吸引人的就是 Java 了。自从 1995 年 5 月的 SunWorld 大会上它受到公众瞩目以来,便掀起了一股对于 INTERNET 和 Java 的热潮。Java 语言以其强大的生命力吸引着软件开发人员,而 INTERNET 上蕴藏了 Java 语言各种模式的无尽的信息资源,一时间众人都在评说这一被比尔·盖茨称作“长时间以来最卓越的程序设计语言”的新一代网络计算机语言,环球信息网 WWW 的创始人 Berners-Lee 说:计算机事业发展的下一个浪潮就是 Java,并且很快会发生。因此可以说:使用 Java 已成为大势所趋!

我们知道,计算机产业正在朝着网络化的方向阔步前进,WWW(World Wide Web/环球信息网)正在成为世界上最大的信息中心。但目前 WWW 上仍多为文本、图象、声音等静态信息,Java 为其提供了极为简便且功能强大的编程接口,向开发人员提供了向 Web 增加动态,交互性内容的功能,使得 Web 页面翻开了新的一页。Java 的新的有利特性使得网络开发变得前所未有的简单易行,完善的可移植性及可靠的安全性更为其开辟了极为广阔的前景。Java 同时有许多其他特性,因此对于那些对生成可在 Internet 上运行的应用程序感兴趣的开发人员来说有着极大的吸引力。

### 1.1.2 Java 语言的深远影响

工业界不少人预言:“Java 语言的出现,将会引起一场软件革命”。

Java 是一种面向对象的语言,允许开发人员生成独立于平台的应用程序。它产生小的轻量代码,这些代码从已经存在于客户计算机的类库中取得其许多功能。因为 Java 是一种解释性语言,所以用它编写的程序可在许多硬件平台上运行而无需重新解释。它同时还有一个强大的安全模块,可以让你生成用户可下载的 applet 而无需担忧在机器上发生任何可怕的事情。同时,Java 的这些特性,对于那些想在其 Web 网点上增加高一级交互性的开发商来说,是非常有吸引力的工具。

大家也许已经发现,现在的编程工具越来越复杂。Borland C++、Visual C++,动辄几百兆,强大是非常强大了,但要掌握它们则非得花上一番功夫不可,令人望而却步。而 Java 的基本解释器只有 40KB 左右,加上标准类库和线程的支持也不过 215KB 左右,可谓短小精悍,而功能毫不逊色,对多线程到面向对象以至于多媒体均提供全面支持。

另外,传统的编程软件往往依赖于具体实现环境,一旦更换则可能有大批已有代码需要重写或重新编译。Java 在二进制码上的兼容性使得同一软件能不加修改地用于不同系统之

上,只需提供相应的 Java 解释器即可。因此,Java 程序可以非常简单地移植到网络上的不同机器上,而只有这样的程序设计语言才能满足网络上形形色色的系统的不同要求。因此,有人预言:Java 将是网络上的“世界语”,今后所有的用其他语言编写的软件统统都要用 Java 语言来改写。

## 1.2 Java 的特点

Java 之所以如此受人推崇,因其确实可称得上是新一代编程语言,具有众多优点,它简单、面向对象、可移植、与硬件无关,系统强健安全,具有很高的性能,并提供了分布性、多线程、动态性的支持。它还提供了在 INTERNET 上加入应用程序的简便方法(Java applet)。下面我们具体介绍 Java 的特点,并与 C、C++ 进行比较。

### 1.2.1 Java 语言的特点

#### (1) 简单性

Java 的简单性首先体现在精简的系统,力图用最小的系统实现足够多的功能。无需强大的硬件环境,Java 在小型的机器上便可以良好运行。

对于编程人员来说,Java 也是非常友善的,无需进行复杂深奥的钻研,只要理解一些基本的概念,就能编写出合适的应用程序。为了使大多数程序员能迅速地掌握 Java,它采取了类似 C++ 的语法,尽管这也许并不是最合适的。Java 语言基于 C++,但它除去了 C++ 中许多难得一用而又复杂不易理解的部分,并去掉了将潜在地威胁 Java 安全的模块,对我们来说它们常常是弊大于利。其中包括有运算符重载、多重继承等等,使人不必纠缠于这些原本就有些模糊的概念。所保留的只是一个简单的、面向对象的多线程语言,其特性包括一个单继承的类系,强类型,支持多线程执行,自动碎片收集,并且只采用引用(非指针)。

Java 实现了自动的垃圾收集,程序设计者不必费心管理内存,一切工作由系统完成。这不但使程序设计更加简便,同时也大大减少了出错的可能。

#### (2) 面向对象技术

和所有新一代的程序设计语言一样,Java 也采用了面向对象技术并更加彻底,所有的 Java 程序和 applet 均是对象,封装性实现了模块化和信息隐藏,继承性实现了代码的复用,使用户可以建立自己的类库。

#### (3) 分布性

通过 Java 提供的类库可以方便地处理 TCP/IP 协议,用户的应用程序可以通过 URL 地址访问网络上的其他对象,就像访问本地文件一样的简单。

#### (4) 鲁棒性

Java 在编译及运行程序时,都要进行严格的检查,以消除错误发生的可能性。作为一种强类型语言,Java 在编译和连接时都进行大量的类型检查,防止不匹配问题的发生。如果你引用一个非法类型,或执行一个非法类型操作,Java 将在解释时指出该错误。

在 Java 程序中不能采用地址计算的方法通过指针访问内存单元,这样就大大减少了错误发生的可能性,Java 的数组并非用指针实现,这样就可以在检查中避免数组越界的发生。

Java 的自动垃圾收集避免了管理内存时容易出现的一些错误。

#### **(5) 安全性**

网络上最重要的是安全问题,作为网络语言,Java 必须提供足够的安全保障,并且要防止病毒的侵蚀。Java 在运行应用程序时,严格检查其访问数据的权限,例如不允许网络上的应用程序修改本地的数据。下载到用户计算机中的字节代码在其被执行前要经过一个核实工具,一旦字节代码被核实,它便由 Java 运行时解释器来执行,该解释器通过阻止对内存的直接访问来进一步提高 Java 的安全性。同时,由于 Java 具有很强的鲁棒性,也使它具有更好的安全性,不能使用指针,应用程序便不能通过计算来非法地访问对象的私有成员。

#### **(6) 体系结构中立**

Java 是一种网络语言,而网络上充满了各种不同类型的机器和操作系统。为使 Java 程序能在网络的任何地方运行,Java 解释器生成与体系结构无关的字节码结构的文件格式,它们能在任何种类的处理器的上运行,只需提供相应的 Java 运行系统。

在单一系统上体系结构中立同样具有非常大的作用。目前,软件作者们不得不编写各种版本的程序以在不同的处理器上运行,而处理器的结构又在不断地发生变化,这使得开发出能在所有平台上工作的软件几乎是不可能的。使用 Java,则同一版本的应用程序就可以运行在所有的平台上。

#### **(7) 可移植性**

体系结构中立使得 Java 程序具有可移植性。同时,Java 的类库也具有可移植性,可以在不同平台上使用。另外,Java 的编译器由 Java 语言实现,运行器由标准 C 实现,因此 Java 本身也具有可移植性。

#### **(8) 解释执行**

Java 解释器直接对 Java 字节码解释执行,字节码本身包含了许多编译时产生的信息,使连接过程更加简单。

#### **(9) 高性能**

尽管 Java 是解释执行的,但它具有非常高的性能,在一些特定的 CPU 上,Java 字节码飞速地转换成机器码,让人几乎无法感觉到它是在解释执行的。

由于 Java 字节码格式的设计就针对着机器码的转换,实际转换时便相当简便,自动的寄存器分配与编译器对字节码的一些优化使之生成高质量的代码。

#### **(10) 多线程**

多线程使应用程序可以同时进行不同的操作,处理不同的事件。网络连接需要时间,如果采用事件循环机制则会造成长时间的等待。在多线程机制中,不同的线程处理不同的任务,他们之间互不干涉,不会由于一处等待影响其他部分,这样就可以容易地实现网络上的实时交互操作。

#### **(11) 动态性**

在许多方面,Java 比 C、C++ 更具有动态性,它被设计为适应于不断发展的环境。在类库中可以自由地加入新的方法或实例量,而不会影响使用这一类库的应用程序的执行。

### **1.2.2 Java Applet 的特点**

Java Applet 是一类特殊的 Java 程序,它嵌入在 HTML 语言中,随主页发布到

INTERNET上。利用它,用户可以非常简单地实现 INTERNET 上的程序编写,实现多媒体的用户界面或进行复杂的计算。Java 程序的简单性使其与其他语言相比大大缩短了下载时间,而 Java Applet 的结构更加简单,代码更短。Java 的可移植性保证了 Java Applet 可以在网络的任意地方运行,只要具备支持 Java 的浏览器即可。Java 的安全检查机制保证用户不会受到病毒的入侵。Java Applet 可以使 INTERNET 上的信息真正具有动态性和交互性。

所有的 Java Applet 都是 Applet 类的子类。Applet 类为用户做好了程序的框架,用户只需填入相应的函数实现即可,无需考虑诸如窗口创建,事件处理之类的问题,大大简化了编程复杂性。

### 1.2.3 Java 丰富的类库

Java 提供了大量的类以满足网络化、多线程、面向对象的要求。

- 语言包提供了字符串、多线程、例外事件处理及数学函数等支持,用它可以简单的实现 Java 程序的运行平台。
- 实用程序包支持哈希表、堆栈、可变数组、时间和日期等。
- 输入输出包用统一的“流”实现了所有格式的 I / O 操作,包括文件系统、网络、输入/输出设备等。
- 低级网络包用于实现 Socket 编程。
- 抽象图形接口包实现了不同平台的计算机的图形用户接口部件,包括窗口、菜单、滚动条、对话框等,使 Java 可以移植到不同平台的机器上。
- 网络包支持 INTERNET 的 TCP/IP 协议,提供了与 INTERNET 的接口,支持 URL 连接,WWW 的即时访问,并简化了用户/服务器模型的程序设计。

### 1.2.4 Java 与 C、C++ 的比较

Java 使用了类似与 C 和 C++ 的语法,而去除了 C 和 C++ 中许多不合理的内容,以实现其简单、鲁棒、安全等特性。

#### (1) 全局变量

Java 程序中,不能定义全局变量,只能通过类中的公用、静态的变量实现全局变量。这样便保证了更好的安全性,全局变量被封装在类中。而在 C 和 C++ 中,依赖于不加封装的全局变量常会造成系统的崩溃。

#### (2) 无条件转移指令

与 C 和 C++ 不同,Java 不支持 goto 语句,而通过例外处理语句 try,catch,final 等来代替 C 和 C++ 中的 goto 来处理遇到错误时跳转的情况,使程序更加结构化并提高了程序的可读性。

#### (3) 指针

指针是 C 和 C++ 中最灵活,但也是最容易出错的数据类型。以指针进行内存操作常造成不可预知的错误。而且,通过指针对内存地址进行显式类型转换后,可以访问类的私有成员,破坏了安全性。Java 中,程序员不能进行任何指针操作。同时,数组在 Java 用类来实现,良好地解决了数组越界这一 C、C++ 中不作检查的错误。

#### (4) 内存管理

C 中,程序员使用库函数 `malloc()` 和 `free()` 来分配和释放内存,C++ 中则是运算符 `new` 和 `delete`。再次释放已释放的内存块或释放未被分配的内存块,会造成系统的崩溃;而忘记释放不再使用的内存块也会逐渐耗尽系统资源。在 Java 中,所有的数据结构都是对象,通过运算符 `new` 分配内存并得到对象的使用权,而实际分配给对象的内存可能随程序运行而改变,Java 自动地进行管理并进行自动垃圾收集,有效地防止了程序员误操作引起的错误并更好地利用了系统资源。

#### (5) 数据类型的一致性

在 C 和 C++ 中,不同的平台上,编译器对简单的数据类型如 `int`,`float` 等分别分配不同的字节数。例如:`int` 在 IBM PC 上为 16 位,在 VAX-11 上为 32 位,因此导致了代码的不可移植性。而在 Java 中,对某种数据类型总是分配固定的位数,这就保证了 Java 的平台无关性和可移植性。

#### (6) 类型转换

在 C 和 C++ 中,可以通过指针进行任意的类型转换,导致不安全的可能。在 Java 中,系统要对对象的处理进行相容性检查,防止不安全的转换。

#### (7) 头文件

C 和 C++ 中使用头文件声明类的原型和全局变量及库函数等,大的系统中,维护这些头文件非常的困难。Java 不支持头文件,类成员的类型和访问权限都封装在一个类中,运行时系统以此对访问进行控制,防止对私有成员的操作。另外,Java 中用 `import` 语句与其他类进行通讯,以使用它们的方法。

#### (8) 结构和联合

C 和 C++ 的结构和联合的成员均为公有,带来了安全性的问题。Java 不支持结构和联合,所有的内容封装在类里。

#### (9) 预处理

C 和 C++ 中用宏定义实现的代码影响了程序的可读性。Java 不支持宏,以关键字 `final` 声明常量。

## 1.3 简单的 Java 程序

Java 源代码可被解释为 Java 应用程序(application)或是 Java applet。

Java application 在运行时解释器内可以作为一个主进程运行。Java applet 则是被下载后,在一个 Web 浏览器内运行。

下面我们通过简单的例子来分别说明这两类应用方法。

### 1.3.1 简单的 Java Application

#### 例 1.1

```
class HelloWorldApp {                //an application
    public static void main (String args[]) {
        System.out.println ("Hello World!");
    }
}
```

这个程序只实现了一个功能,即输出一行信息:

Hello World!

通过这个简单的应用程序,我们可以粗略的了解 Java 应用程序的基本结构。

#### • 类定义

所有的 Java 应用程序都是类,本例中即为 HelloWorldApp 类。用保留字 class 来声明新类,public 指明这是一个公共类,Java 程序中可以定义多个类,但最多只能有一个公共类。

#### • main()方法

本例中定义了一个方法,这就是 main()方法。事实上,所有应用程序都必须有且只有一个 main()方法,而且必须用 public、void 和 static 指定,public 指明所有的类都可以使用这一方法,void 指明本方法不返回任何值,static 则指明本方法是一个类方法,可以通过类名直接调用。Java 解释器在没有生成任何实例的情况,以 main()方法作为入口来执行程序。

main()方法定义中,括号中的 String args[]是传递给 main()方法的参数,名为 args,它是类 String 的一个实例,参数可以是一个或多个,每个参数用“类名 参数名”来指定,多个参数间用逗号分隔。

#### • 程序内容

本例的 main()方法中,只有一条语句:

System.out.println("Hello World!");

它用来实现字符串的输出,实现与 C 中的 printf 和 C++中的 cout<<相同的功能。

#### • 注释

与 C++相同,一行中“//”后的内容为注释,“/\*”和“\*/”中的内容也是注释。

现在我们可以运行此程序。首先把它放在名为 HelloWorldApp.java 的文件中,注意:文件名必须与公共类名相同,因为 Java 解释器要求公共类必须放在与之同名的文件中。然后对文件进行编译:

javac HelloWorldApp.java

生成字节码文件 HelloWorldApp.class。最后用 Java 解释器运行它:

java HelloWorldApp

于是在屏幕上显示: HelloWorld!

### 1.3.2 Java Applet

**例 1.2**     import awt.Graphics;  
              import browser.Applet;

```
class SampleApplet extends Applet {  
    /* *  
     * Applet methods  
     */  
    int pos_x,pos_y;  
    String str;  
    public void init() {  
        resize(300, 200);  
    }  
}
```

```

        pos_x=10; pos_y=40; str="SampleApplet: init()";
        repaint();
    }
    public void start() {
        pos_x=10; pos_y=80; str="SampleApplet: start()";
        for(int i=0; i<999999; i++){ } //time delay
        repaint();
    }
    public void stop() {
        pos_x=10; pos_y=120; str="SampleApplet: stop()";
        repaint();
        for(int i=0; i<999999; i++){ }
    }
    /*
    * Paint a rectangle with some message
    */
    public void paint(Graphics g) {
        g.drawRect(0, 0, width - 1, height - 1);
        g.drawString(str,pos_x,pos_y);
    }
    /* *
    * Mouse methods
    */
    public void mouseDown(int x, int y) {
        pos_x=x; pos_y=y; str="SampleApplet: mouseDown";
        repaint();
    }
    public void mouseDrag(int x, int y) {
        pos_x=x; pos_y=y; str="SampleApplet: mouseDrag";
        repaint();
    }
    public void mouseUp(int x, int y) {
        pos_x=x; pos_y=y; str="SampleApplet: mouseUp";
        repaint();
    }
}

```

这个 Applet 在屏幕上显示一个窗口,其中以字符串显示 Applet 的一些状态。  
下面,我们以这个例子来看一看 Java Applet 的结构和运行过程。

## ■ 基本结构

### • 类定义

与 Java 应用程序相同,所有的 Java Applet 也都是类。而且必须是 Applet 类的子类,用

保留字 extend 说明。

### • 方法定义

在 Java Applet 中没有 main() 方法, 这是 Java Applet 与 Java 应用程序之间的主要区别之一。Applet 类实现了许多方法, 构造了一个窗口程序的主要框架, 用户只须重写所需的方法即可, 系统将自动调用。

### • 类库支持

这个程序中用到了类库中的一些方法, 因此必须在程序中进行声明。在 Java 中, 用 import 来输入包, 本例中便输入了 java. awt. Graphic 和 java. Applet 下所有的包。

### ■ 运行 Java Applet

首先要进行编译, 同样也要放在与公共类同名的文件中, 即为 SampleApplet. java, 然后进行编译:

```
javac SampleApplet. java
```

得到字节码文件 SampleApplet. class。

与应用程序不同的是, Java Applet 不能用上面的方法运行。必须编写 HTML 文件, 把 Applet 嵌入其中, 然后用 appletviewer 来运行, 或在支持 Java 的浏览器上运行。HTML 文件类似下面的情况:

### 例 1.3

```
<HTML>
<HEAD>
<TITLE> The First Applet</TITLE>
</HEAD>
<BODY>
<applet code="SampleApplet. class">
</applet>
</HTML>
```

### ■ 运行结果及分析

在装入上述 HTML 文件后, 便会出现一个窗口, 它的左上角有一串字符显示“SampleApplet: init()”。随后它消失, 在它的下面显示“SampleApplet: start()”。现在可以在窗口中点鼠标左键, 在点下处显示“SampleApplet: mouseDown”; 如果拖动鼠标, 则在光标处显示“SampleApplet: mouseDrag”; 松开鼠标则会显示“SampleApplet: mouseUp”。在离开这页时, 还会看到“SampleApplet: stop()”; 而重回时又会看到“SampleApplet: start()”。

下面, 我们来分析为什么发生这些情况。

当一个 Java Applet 被装入时, 首先执行其 init() 函数。因此, 用户的初始化部分可以放入 init() 方法中, 如设置窗口大小, 装入资源等。随后, 便调用 start() 函数, 每次在 Applet 所在页变为可见时, 系统都会调用 start() 方法, 因此用户程序往往由此实际开始。而离开这一页时, stop() 方法将被调用。

与一般的 Windows 程序不同, Java Applet 中没有事件循环, 由 Applet 类实现。因此, 用户程序只需重写自己所需的方法即可, 如例 1.2 中的 mouseDown、mouseDrag 和 mouseUp。系统会自动调用相应的方法。

当窗口需重画时, 系统调用 paint() 方法。如例子中的 paint() 方法便实现了显示不同提



示信息的功能。

## 1.4 Java 的应用远景

由于 Java 语言具有能独立于平台而运行,面向对象,可对动态画面进行设计操作,坚固性好等特点,又具有多线程,内置校验器用来防止病毒入侵等功能,所以用来在 Internet 上研制与开发软件时,特别受到用户的欢迎。

Java 的结构类似于 C 和 C++,所以一般熟悉 C 与 C++ 语言的编程人员稍加学习就不难掌握 Java 的编程技术。

正是 Java 语言具有上述特性,所以有着广泛的应用前景,大体上可以从以下几个方面来考虑其应用:

- 所有面向对象的应用开发,包括面向对象的事件描述,处理,综合等;
- 计算过程的可视化、可操作化的软件的开发;
- 动态过程的可视化,包括图形图象的调用;
- 交互操作的设计(选择交互、定向交互、控制流程等);
- Internet 的系统管理模块的设计,包括 Web 页面的动态设计,管理和交互操作设计等;
- Intranet(企业内部网)上的软件开发(直接面向企业内部用户的软件);
- 与各类数据库连接查询的 SQL 语句实现;
- 其他应用类型的程序。

以上只是对 Java 语言的一个简略而粗糙的概述,目的是使读者在学习和使用 JDBC 时对 Java 本身概念有一定的了解,涉及到具体的编程问题,可以参阅《Java 系列丛书》之一,之三。