计算机技术入门提高精通系列丛书

Visual J++

程序设计入门

东箭工作室　编著

人民邮电出版社

Visual J++ 程序设计入门

东箭工作室 编著

人民邮电出版社

计算机技术入门提高精通系列丛书
Visual J++程序设计入门
东箭工作室　编著
责任编辑　顾翀
*
人民邮电出版社出版发行
北京崇文区夕照寺街 14 号
东箭信息有限公司排版
北京顺义振华印刷厂印刷
新华书店总店北京发行所经销
*
开本：787x1092 1/16　　1997 年 3 月 第一版
印张：11.5　　1997 年 3 月 北京第 1 次印刷
字数：278 千字　　印数：1 - 8 000 册
ISBN 7-115-06441-5/TP·420
定价：18.00 元

内容简介

Java 是由 Sun Microsystems 公司开发的新一代的程序设计语言，它适合于分布式计算，集面向对象、平台无关性、健壮性、安全性和多线程于一身，已经成为 Internet 上的主要开发语言，应用日益广泛。Visual J++是 Microsoft 公司推出的可视化的 Java 语言集成开发环境，可以在同一个环境中完成 Java 程序的建立、调试、修改和运行等全过程，大大提高了 Java 程序的开发效率，缩短了开发周期，而且 Visual J++是目前唯一支持 ActiveX 的 Java 环境。

本书共分为 9 章，系统地介绍了 Java 语言的基本知识，包括面向对象的程序设计基础和 Java 语言的语法。结合 Visual J++的特点，本书还介绍了如何使用 Visual J++的集成开发环境 Microsoft Developer Studio，如何使用 Visual J++提供的 Applet 向导来生成 Applet，如何使用 Resource 向导来生成各种 GUI 资源的 Java 代码，如何利用集成环境的动态调试功能对程序进行动态跟踪调试，如何在 Visual J++提供的庞大的联机帮助中寻找需要的信息。此外针对 Java 的应用需要，本书还介绍了如何进行异常处理，如何编制图形用户界面（GUI），如何利用 Java 语言的特性来实现多线程，以及如何在 Web 上实现声音、动画等多媒体信息内容。

本书不仅讲解了 Java 编程的基本内容，还介绍了许多帮助读者掌握 Visual J++的各种功能，提高开发效率的技巧。读者从中可以学到更深层次的知识。

前 言

Internet 目前在我国已被越来越多的人所认识。在 Internet 上建立自己的主页所需费用低、影响范围广，已经成为许多公司、企事业单位树立自身形象、开展业务宣传的有力手段。同时， Intranet 也正在逐步成为企业内部信息系统的最佳解决方案。正因为如此，作为 Internet 上的编程工具—— Java 也越来越受到人们的重视。

本书不同于一般的 Java 语言参考书，它的重点在于介绍如何利用 Visual J++提供的各种强大功能来提高 Java 程序的开发效率，而不是一板一眼地讲述 Java 编程的每一条语句。有此基础，读者便可轻松地走进 Java 的大门，进一步成为 Java 的编程高手。

本书没有涉及高深晦涩的原理和术语，只有深入浅出的描述和一些简短有趣的例程，帮助读者循序渐进地掌握 Java 编程中最基本也是最精髓的部分，例如面向对象、程序调试、图形用户界面、异常处理和多线程等知识。

我们希望通过这本书给读者提供的，是如何在掌握书中介绍内容的基础上，继续拓展自己知识面的思路和方法，而不仅仅是依照书中的例子编一两个 Java 的应用程序。

Java 连同 Internet 和 WWW，正在改变应用软件的开发和使用方式。或许就在不久的将来， Word 将演化成为 Web 写作工具， Excel 则将演化成 Web 上的电子表格。到那个时候，希望广大读者已经做好万全的准备。

本书的写作过程得到了多方人员的大力支持，在此一并致谢！

由于作者本身的水平有限，再加上本书创作时间紧促，因此本书肯定存在粗疏、忽漏和错误之处，衷心希望各界专家、用户和读者朋友不吝赐教，为本书以及其它东箭图书提供宝贵意见，使得我们能够更好地为广大读者服务。

反馈意见可以发送到以下地址：

电子邮件： support@oa.co.cn

通信地址： 北京崇文区夕照寺街 14 号

人民邮电出版社计算机图书编辑部 收转(100061)

东箭工作室

Oriental Arrow Information Co., Ltd.

1997 年 1 月

目 录

第1章

J++和 Java

随着 Internet 和 WWW （ World Wide Web ）的日益发展， Java 语言也越来越引起人们的重视。 Java 是由 Sun Microsystems 公司开发的新一代的程序设计语言，它适合于分布式计算，集面向对象、平台无关性、健壮性、安全性和多线程于一身，逐步成为 Internet 上的重要开发语言。许多大公司都购买了 Java 的使用许可证，其中 Visual J++就是 Microsoft 公司开发出的支持 Java 语言的集成开发环境。它可以在同一个环境中完成 Java 程序的建立、调试、修改和运行等全过程。

在本章中，将向读者介绍 Java 语言的发展历程并比照 C++介绍 Java 自身的特点，最后还将简要介绍 Visual J++在功能上所做的改进。

1.1 Java 语言概述

Java 语言的白皮书是这样定义 Java 的：

Java 是一个简单的、面向对象的、分布的、健壮的、安全的、与平台无关的、可移植的、高性能的、多线程的以及动态的解释型程序语言。

但是读者可能想不到，现在在 Internet 上大红大紫的 Java 语言开始的时候并不是为了 WWW 而设计的，正所谓“有意栽花花不发，无心插柳柳成荫”。

1.1.1 Java 的发展

Sun 公司的 Java 语言开发小组成立于 1991 年，其目的是开拓消费类电子产品市场，该小组的领导人是 James Gosling 。他是一位非常杰出的程序员。

在研究开发过程中， Gosling 意识到消费类电子产品的用户并不同于工作站的用户，他们并不关心 CPU 的型号，也不欣赏昂贵的专用 RISC 处理器，他们只需要一个建立在标准基础之上的与硬件平台无关的一系列可选方案。

为了使整个系统与平台无关， Gosling 首先着手改写 C++的编译器，但是不久他感到 C++是无法满足需要的，于是在 1991 年 6 月份开始准备开发一个新的语言。 Gosling 从窗外的一颗老橡树得到灵感，于是建立一个目录叫 Oak ，这就是 Java 语言的前身。后来发现 Oak 已是 Sun 公司的另一个语言的注册商标，才改名为 Java （印度尼西亚一个盛产咖啡的岛屿，即我国古典小说中经常提到的爪哇国）。

Java 语言初步设计完成后， Sun 公司使用 Java 语言开发了一些技术上非常成功的实验软件，但是由于激烈的市场竞争和其他一些商业因素，这些产品没有能推向市场， Java 语言几乎中途夭折。

直到 1994 年，图形界面的 WWW 已如火如荼地发展起来。 Sun 公司决定将 Java 定位到 Web 浏览器的应用上，才使 Java 走上了快速发展的轨道。 1995 年 5 月名为 HotJava 的浏览器发表后，引起了巨大的轰动。与此同时， Netscape 公司也宣布在其浏览器中支持 Java 。不久许多著名的大公司，如 IBM 、 Microsoft 、 Novell 、 Oracle 、 SGI 和 Borland 也都相继购买了 Java 的使用许可， Java 的地位终于得到了广泛的肯定。

1.1.2 Java 和 C++的不同

有一个公式可以说明 Java 和 C++之间的关系：

Java="C++"－" Complexity and Ambiguity " + " Security and Portability "

（ Java=“ C++”－“复杂性和奇异性”+“安全性和可移植性”）

从公式可以看出， Java 是以 C++为基础设计的，它在形式上和 C 、 C++极为相似，如果用户已经懂得一些 C 或 C++语言，就可以很快学会 Java 。但是与大名鼎鼎的 C++相比，Java 语言更具有自己的特点。

一、 简单性

Java 是一种简单的语言，它用到的概念不多，而且都为程序员所熟知，所以十分易于学习。

为了保证语言的简单性， Java 删除了 C++中的许多语言功能。这些功能往往很少被使用，或者容易出错。例如， Java 不支持 goto 语句，取而代之的是带有标号的 break 和 continue 语句以及异常（ exception ）处理；不支持头文件、 C 的预处理器、 struct 和 union 语句，还有 C++的运算符重载和多继承也被排除在 Java 之外。 Java 出于语言简单性和安全性的考虑，甚至把 C 程序员推崇备至的指针也取消了，因为指针实际上是 C 和 C++最容易出错的方面，而且往往出了错以后很难定位。

另外，自动无用单元收集也是 Java 的重要特色，它把程序员从复杂的内存管理中解放出来。由于 Java 采取了上述措施，使程序员彻底从指针悬挂、非法指针引用、内存丢失的困扰中解脱出来。

二、 面向对象

Java 是一种比较纯粹的面向对象（ Object-Oriented ）的程序设计语言，这意味着程序员考虑的首先是对象中的数据和定义在数据上的操作，而不仅仅是过程。对于习惯于 DOS 编程的程序员，使用 Java 必须要改变思考问题的方式。

Java 不支持 C++的模板（ template ）机制，但在 Java 中，所有的类都是 Object 类的子类。 Java 还抛弃了 C++中的非面向对象特性，如 struct 、函数调用，还有全局变量。

三、 分布式

Java 支持网络上的应用程序，是一种分布式（ Distributed ）程序设计语言。 Java 提供了一个 java.net 包，通过这个包中的类，可以完成各种层次连接。例如， URL 类支持 Java 应用程序通过 Internet 打开和访问远程对象。使用 Java 打开远程文件和打开本地文件几乎一样简单。 Java 提供一个 Socket 类，这个类可以提供可靠的流式网络连接。这样，我们可以非常方便地创建分布式的 Client/Server 应用程序。

四、 健壮性

Java 是一种比 C++还强的强类型语言。 Java 要求显示的方法声明，这保证了编译器可

以发现方法调用错误，保证了程序更加可靠。

Java 内存模型是提高程序可靠性最重要的手段。Java 不支持指针，这杜绝了内存的非法访问。Java 的自动无用单元收集防止了内存丢失等动态内存分配所导致的问题。Java 解释器运行时也实行检查，可以发现数组和字符串访问的越界。

异常处理是 Java 保证程序健壮的另一重要手段。异常是程序错误或可能导致程序错误的不正常状态。在 Java 语言中，通过使用 try/catch/finally 语句，程序员可以把一组错误处理代码放在一个地方，这样既可以简化错误处理任务和恢复，同时也增强了程序的可读性。

五、 安全性

由于 Java 主要应用于网络应用程序的开发，对安全性有很高的要求。如果没有安全性的保证，用户从网络下载（ download ）程序执行是非常危险的。 Java 自己的安全机制可以有效地防止病毒程序的产生和下载程序对本地文件系统的破坏。

Java 的内存分配模式是防止有害代码的重要手段。Java 取消了指针，程序就无法访问它不该访问的内存。

最重要的是 Java 编译器并不处理内存的布局，这样程序员无法从类的定义中推断出运行时刻的实际内存布局。Java 编译代码的内存引用由 Java 解释器在运行时转化成实际内存的引用。通过放弃 C++的内存布局和指针模型， Java 语言已经去掉了程序员直接进行内存分配及布局的能力。

Java 的另一个有效的安全措施是运行系统对字节代码进行验证，保证了从网络上下载的代码不违反 Java 语言的限制。Java 运行系统不信任任何引入的代码，因此在运行系统中增加了字节代码验证器，在字节代码执行前对其“验明正身”，防止网络病毒和其他形式的入侵者绕过 Java 编译器生成危险的字节代码。形象一点地说，字节代码验证器起到了一个门卫的作用，它将每个输入的代码送给一个规则验证程序，以确保代码遵循如下规则：

- 不存在伪造的指针。
- 没有违反访问权限。
- 严格遵守访问对象的规范。
- 使用恰当的参数调用方法。
- 没有栈溢出。

六、 平台无关性

Java 的字节代码是一种与平台无关（ Platform Independence ）的代码。任何平台只要实现了 Java 虚拟机， Java 程序就可以在上面运行。 Java 的虚拟机是免费的，现在已有 Windows 95 ， Windows NT 以及多种 UNIX 版本，这样开发的程序很容易在多种平台上使用。

七、 可移植

平台无关性本身就提供了一种良好的可移植性（ Portable ）。 Java 还提供了实现无关性（ no implementation dependent ）。例如， Java 中的原始数据类型的长度在任何平台上都一样。一个整型总是 32 位，一个长整型总是 64 位，而 C 和 C++则不能满足这一点。 Java 环境本身对硬件平台和操作系统也是可移植的，因为 Java 的编译器是用 Java 写的， Java 的运行系统是用 ANSI C 写的。

为了使 Java 的应用程序可以不依赖于具体的系统，Java 还提供了一个用于访问底层操作系统功能的可扩展类库，程序使用这些库时，可以确保它能运行在支持 Java 的各种平台上。

八、 高性能

Java 在拥有了安全性、健壮性和可移植性的同时，还保持了较高的性能。而且 Java 还可以即时编译和链入 C 代码以提高性能。所谓即时编译就是由代码生成器先将字节代码转换为本机的机器码。

九、 多线程

Java 是一个支持多线程（ Multithreaded ）的语言，它可以同时运行多个线程处理多个任务。多线程技术可以提高图形用户界面的交互性能。使用 C 或 C++开发多线程应用程序往往是令人头痛的。首要的困难是要保证多个例程可被若干并发线程运行。如果一个例程改变了状态变量的值，那么一次只可能有一个线程执行。用 C 和 C++编写多线程应用程序时，要程序员负责例程的锁定与释放。这种方式既繁复，又极易产生死锁。

Java 提供了语言内置的多线程控制，从而大大简化了多线程应用程序的开发。在 Java.lang 包中提供了一个叫 Thread 的类，由它负责线程的启动、运行、终止并且可以检查线程的状态。

Java 的线程支持还包括一组同步原语，可以确保变量处于一致的状态。

十、 动态性

Java 语言是一种动态（ Dynamic)语言。例如， Java 可以动态地从本地或网上加载类。Java 的类也有运行时的表示，这样即使在运行时刻，程序也能辨别类之间的关系和类型信息。 Java 中的运行类（ runtime class ）定义使 Java 可以动态地把一个类链接到运行系统中去。

Java 的动态特性是其面向对象设计的延伸，这使得 Java 可以在分布环境中动态地维护应用程序及其支持类库之间的一致性，而不用像 C++那样，每当其支持类库升级之后，相应的应用程序都必须重新编译。

十一、 解释型语言

Java 是一种解释型语言。 Java 编译器产生的是字节代码（ byte-code ），而不是特定

的机器码，是一种解释型（ Interpeted ）语言。 Java 程序的字节代码必须运行在一个解释器上， Java 的字节代码是一种与平台无关的对象文件格式，它可以高效地在不同平台之间传输。一个 Java 程序可在任何平台上运行，只要这个平台上有 Java 解释器和运行系统。

1.1.3 Java 的解释器

解释器可以在 Java 程序运行时将字节代码翻译成机器代码。字节代码是由 Java 的编译器生成的，但在不同的系统上解释器是不同的，这样才能使 Java 程序运行在不同的系统之上。

1.1.4 Java 虚拟机

Java 解释器和运行系统就构成了 Java 的虚拟机（ Virtual Machine ）。 Java 在程序编译后生成的不是某种 CPU 的指令码，而是 Java 独有的字节代码（ byte-code ）， Java 字节代码运行在 Java 虚拟机上。 Java 虚拟机可以解释并执行 Java 的字节代码，它的作用类似一个小巧而高效的 CPU 。它负责执行指令，还要管理数据、内存和寄存器。 Java 的虚拟机是 Java 解释器的核心，在那些支持 Java Applet 的 Web 浏览器中，也嵌入了 Java 虚拟机。

每一种操作系统的 Java 解释器是不同的，但实现的 Java 虚拟机是相同的，这是 Java 平台无关的关键所在。 Java 虚拟机由 5 个部分组成：一组指令集，一组寄存器，一个栈（ stack ），一个无用单元收集堆（ garbage-collected heap ），一个方法区域。这五部分是虚拟机逻辑的抽象成分，不依赖于任何实现技术或组织，但它们的功能必须在真实机器上以某种方式实现。例如，采用字节代码方式，编译成特定的机器码，甚至制作一块虚拟机芯片。 Java 虚拟机内存区域不依赖实际内存的位置，也不要求连续，但要求逻辑长度固定。例如， Java 的栈是 32 位的。

Java 的堆（ heap)是运行时刻动态分配的对象存储区域。 Java 可以进行自动的无用单元收集，程序无需显式地释放占有的空间。无用单元自动回收的算法依赖于虚拟机最终实现的硬件环境。

Java 字节代码的运行可以有两种方式：

- **即时编译方式**：有代码生成器先将字节代码转换为本机机器代码，然后可以以较高速度执行。
- **解释执行方式**：由解释器通过每次翻译并执行一小段代码来完成 Java 字节代码的所有操作。

Java 运行系统通常采取第二种方法，而对于运行速度要求较高的程序，可以采取即时编译的方式。

1.2 为什么要用 Visual J++

Microsoft Visual J++ 1.0 是 Microsoft 公司开发出的可视化的 Java 开发环境，它的用户界面采用了 Microsoft 统一的 Developer Studio 。如果用户使用过 Microsoft Visual C++ 4.0 ，