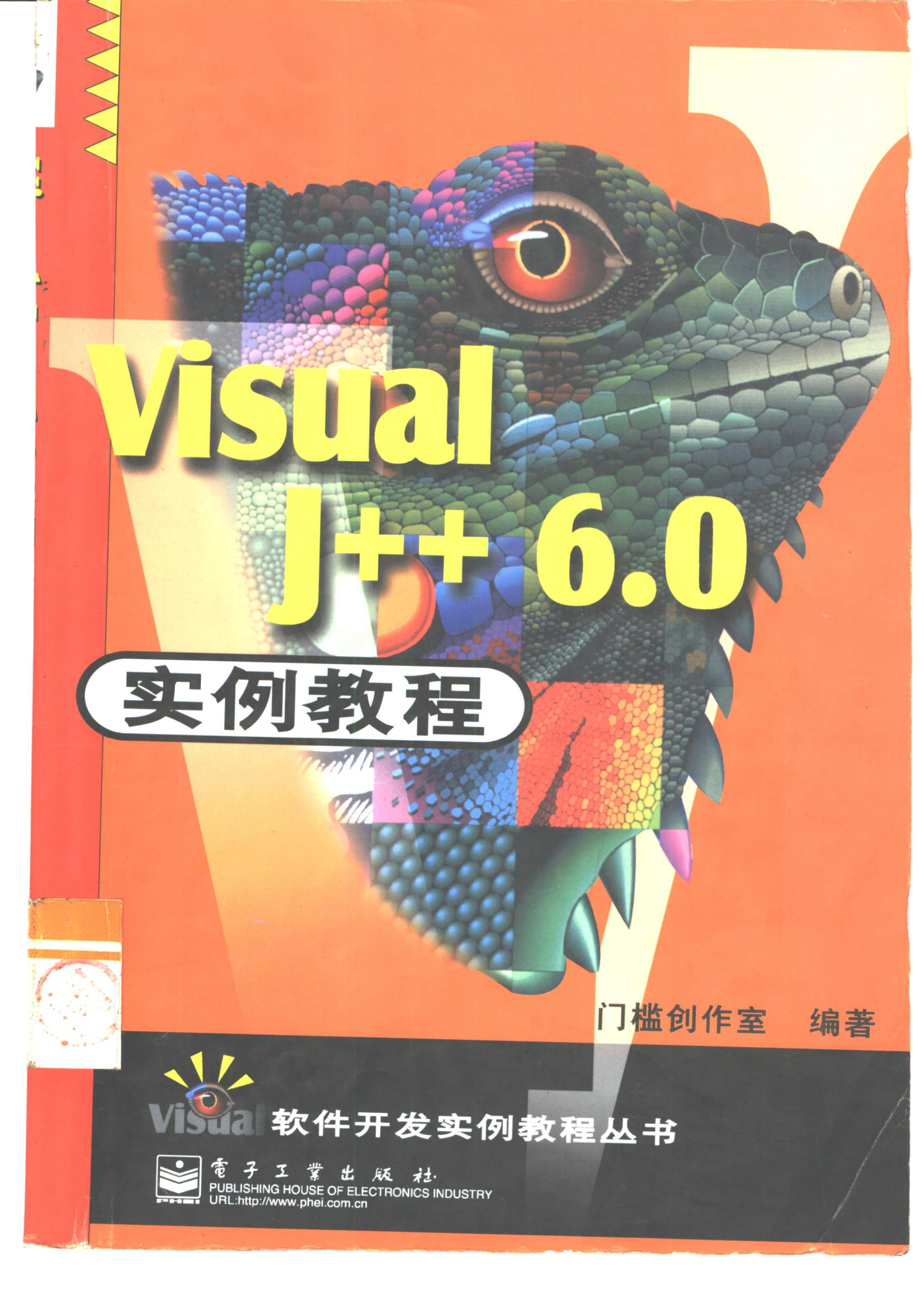

Visual J++ 6.0

实例教程

门槛创作室　编著

Visual 软件开发实例教程丛书

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
URL:http://www.phei.com.cn

软件开发实例教程丛书

Visual J++ 6.0 实例教程

门槛创作室　编著

電子工業出版社
Publishing House of Electronics Industry
北京 • BEIJING

内 容 提 要

Java 语言具有面向对象、高性能、可移植性、动态性和分布式等优良的特性，是 Internet 上的世界语言，Java 的成功及其在全世界范围内的流行，使 Internet 和 WWW 的发展进入了一个新的时代。Visual J++ 6.0 是由 Microsoft 公司提供的高效 Java 应用程序开发环境，掌握了该 Java 开发环境，用户能够更方便、更有效地开发出更合理的 Java 应用程序。

本书共分 12 章，对 Java 语言、Visual J++ 6.0 开发环境、如何利用该开发环境进行 Java 应用的开发进行了深入浅出的、系统的介绍，本书结构严谨、布局合理、重点突出，并有丰富的程序实例，使读者能很快地掌握利用 Visual J++ 6.0 开发环境进行 Java 程序设计的方法和技巧。

本书既可以作为相关专业的本科生和研究生学习 Visual J++ 6.0 的教材和教学参考书，也可以供其他工程技术人员，特别是软件开发人员自学使用。

图书在版编目（CIP）数据

Visual J++6.0 实例教程 / 门槛创作室编著. - 北京：电子工业出版社，1999.3
（软件开发实例教程）
ISBN 7-5053-5236-9
Ⅰ.V… Ⅱ.门… Ⅲ.Java 语言-程序设计 Ⅳ.TP312
中国版本图书馆 CIP 数据核字（1999）第 01172 号

从 书 名：软件开发实例教程丛书
书　　名：**Visual J++ 6.0 实例教程**
编 著 者：门槛创作室
策　　划：赵丽松
责任编辑：吴　源
特约编辑：李银胜
印 刷 者：北京大中印刷厂
出版发行：电子工业出版社　URL：http://www.phei.com.cn
　　北京市海淀区万寿路 173 信箱　邮编 100036
经　　销：各地新华书店
开　　本：787×1092　1/16　印张：20.25 字数：518 千字
版　　次：1999 年 4 月第 1 版　1999 年 4 月第 1 次印刷
书　　号：ISBN 7-5053-5236-9/TP・2611
定　　价：28.00 元

软件开发实例教程丛书

编 委 会

总　序

计算机技术的广泛应用，正在改变着我们的生活。了解计算机、学习计算机、掌握计算机就意味着把握住了新时代的脉搏，把握住了新的机遇，也许意味着新生活的开始。在这一学习计算机技术的大潮中，掌握使用编程技术无疑是新时代弄潮儿所追求的梦想。

然而，编程语言这一统治计算机世界最有力的武器并不是那么容易驾驭的。要学好编程语言不仅要熟悉语言本身，还应能洞察计算机世界内在的运行规律。所以，未来的编程大师们正急待有力的引导和支持。

《软件开发实例教程丛书》正是一套能够帮助大家走进计算机编程奇妙天地并洞察计算机世界内在规律的优秀教程。本套丛书遵循编程语言可视化的最新趋势，向广大读者全面介绍广泛使用的可视化编程工具，如微软公司的最新力作 Visual Studio 6.0 系列语言（包括其中较常用的 Visual Basic 6.0、Visual C++ 6.0、Visual J++ 6.0、Visual FoxPro 6.0 及 Visual InterDev 6.0）等等。本套丛书还重点介绍了流行于 90 年代的面向对象的编程技术和计算机网络编程技术，为读者学习编写软件奠定了良好的理论基础。

《软件开发实例教程丛书》包含大量程序实例，通过这些生动有趣的实例向读者介绍可视化编程的技术和软件开发的思维方式。书中还提供了大量详细注释过的程序代码，对读者具有很高的参考价值。

《软件开发实例教程丛书》经过精心的策划及认真的组织编写，终于及时与广大读者见面。但由于编著者水平有限，时间仓促，书中不当之处，恳请各位同仁及广大读者批评指正。

门槛创作室
http://menkan.yeah.net
Computerman Corporation E-mail:ldandxwh@public.bta.net.cn

前　言

Java 是由 Sun 公司开发而成的新一代编程语言。使用它可在各式各样不同种机器、不同种操作平台的网络环境中开发软件。不论你使用的是哪一种 WWW 浏览器，哪一种操作系统和哪一种计算机，只要在该浏览器上支持 Java，就可以在上面看到各种由 Java 写成的生动主页。

由于 Java 的运行环境与平台无关，所以移植起来十分方便。此外，Java 语言还具有面向对象、高性能、稳定性、安全性和分布式等优良特性。它正在逐步成为 Internet 应用的主流开发语言，被人们誉为“网络上的世界语”。它彻底改变了应用软件的开发模式，带来了自 PC 机以来的又一次技术革命，为迅速发展的信息世界增添了新的活力。

由于 Java 语言具有很大的优势，世界各大计算机公司都竞相购买了 Java 的使用权，如 IBM、HP 等。Microsoft 公司也宣布支持 Java 技术，并推出了自己的 Java 编程开发集成环境 Visual J++。Visual J++ 6.0 是 Microsoft 公司推出的一个全新产品，它具有良好的用户界面，为广大的 Java 开发人员提供了一个优秀的开发环境。

由于网络技术开始在中国流行，很多读者希望能有一本参考书，能为自己学习利用 Visual J++ 6.0 开发环境提供参考。为此，我们编写了本书，深入浅出地介绍了 Java 语言、Visual J++ 6.0 开发环境以及如何利用该环境进行 Java 编程。书中提供了大量的程序实例，为读者的学习提供了方便。

目　录

第一篇　基 础 篇

第二篇 应 用 篇

第三篇 高 级 篇

第一篇

基 础 篇

本篇导读

本篇的内容是本书中的基础部分，分为四个部分：Java 语言基础、Visual J++ 6.0 介绍、Visual J++ 6.0 界面介绍、用 Visual J++ 6.0 建立应用的方法。在 Java 语言基础这一章中，我们向读者介绍了 Java 语言的数据类型、Java 的运算符和表达式、Java 的流程控制、Java 的类、Java 的接口和包等内容，读者可以从中了解 Java 的面向对象编程的特点，同时可以掌握 Java 的语言特点。在 Visual J++ 6.0 介绍这一章中，我们向读者介绍了安装、删除、启动和退出 Visual J++ 6.0 的过程，读者获得 Visual J++ 6.0 的帮助的方法，以及 Visual J++ 6.0 的特点。在 Visual J++ 6.0 界面简介这一章中，我们介绍了 Visual J++ 6.0 的界面情况，包括两方面的内容，即 Visual J++ 6.0 的屏幕构成和 Visual J++ 6.0 的菜单系统。在第 4 章用 Visual J++ 6.0 建立应用的方法中，我们创建了一个简单的 Applet 程序和一个独立的 application 程序，使得读者对 Visual J++ 6.0 的使用环境更加熟悉。通过本篇的学习，读者可以掌握 Visual J++ 6.0 的基本内容。

第1章 Java语言基础

Java是近20年来计算机软件领域最有意义的进步之一。同超文本链接标注语言HTML（Hypertext Markup Language，是一种在World Wide Web上实现静态文本的语言）一样重要，成为现今Internet上的热门话题。Java语言是一种强有力的编程语言，它将面向对象的设计用一种简单和熟悉的语法根植在增强的、易用的环境中。Java有一组丰富的对象类，允许程序员对许多公用的系统功能如窗口操作、网络和输入/输出进行简明的抽象处理。Java具有任何人都可使用的Applet程序。Applet是一种小巧、安全、动态、跨平台、活跃和网络化的应用程序。

本章的主要内容为：

- 面向对象的编程基础
- 一个最简单的Java应用
- Java数据类型
- Java运算符和表达式
- Java流程控制
- Java类
- Java接口和包
- Java API简介

1.1 面向对象的编程基础

面向对象是当前计算机领域最流行的程序设计方法。在现实生活中，人们通过抽象来处理复杂事物。这种将复杂事物进行分层抽象的方法不仅适用于现实世界，也适用于计算机程序。传统的算法程序可以抽象成各种要处理的对象，一系列处理步骤可构成独立对象间的信息集。每个对象都封装了自己的行为，我们将这些对象甚至抽象对象当作现实世界具体的实例，它们能响应外界的刺激并进行相应的动作。这就是面向对象编程的基础。与人类理解复杂事物的方式一样，面向对象的概念构成了Java的核心。面向对象具有封装、继承、多态三个主要特性。

1.1.1 封装

从最基本的角度看,任何程序都包含两个部分：代码和数据。在传统的代码模型中，数据在内存中进行分配并由子程序或函数代码来处理。而面向对象设计的核心一环是将处理数据的代码、数据的声明和存储封装在一起。

可以把封装想象为一个将代码和数据包起来的保护膜。这个保护膜定义了对象的行为，并且保护代码和数据不被任何其它代码任意访问。即一个对象中的数据和代码相对于程序的其余部分是不可见的，它能防止那些非期望的交互和非法的访问。

Java封装的基本单元称为类。用户可以创建自己的类，它是对一组具有行为和结构的

对象的一种抽象。对象是相关类的具体实例，是将类当作模子造出的一个翻版。因此，有时称对象为类的实例。

封装的目的是为了减少复杂性，因此，类具有一套隐藏复杂性的机制。类中的每个方法或变量都可以被定义为公有或私有。类的公有部分可以让外界的用户知道或必须知道，公有的实例变量和方法是类与外部的接口，程序的其它部分通过这个接口使用类的功能。而定义成私有的方法和实例数据则不能被类以外的其它代码所访问。

改变类的其它部分不会对整个程序产生预料之外的影响。只要保持类接口不变，它的内部工作方式可以随便改动。

Java 不仅允许程序设计者自己创建类，还针对各种应用提供了大量的预定义类库，例如屏幕显示、文件访问、数学计算等方面的类库。

1.1.2 继承

大部分人通常都会将世界看成相互关联的可划分层次的各种对象，如动物、哺乳动物和狗。这里哺乳动物是动物的继承，而狗又是哺乳动物的继承。下一层次是上一层次的进一步具体描述，并且下一层次继承了上一层次的所有特性。一个多层次的继承关系构成了一个类树结构。

在面向对象的程序设计中，继承是指在已有类的基础上建立一个新类。新类自动拥有父类的所有元素：方法和实例变量，然后再根据需要添加新任务所需的方法和/或实例变量。合理使用继承可以减少很多的重复劳动。如果类实现了一个特别的功能，那么它的派生类就可以重复使用这些功能，而不再需要重新编程。对 Java 的内置类可以创建派生类，也可以对自己创建的类建立派生类。

一个不由任何类派生来的类称为基类；一个派生类的最近的上层类叫做该类的父类；从某一类派生出来的类叫做该类的子类。子类的层次并不是越多越好，因为如果层次太多的话，还不如重新创建一个新类。

一个类从派生它的基类到它自身可能要经过好几个层次。类不仅能继承其父类的所有方法和实例变量，而且还能继承从它的基类开始到它自身之间经过的所有层次上的类的方法和实例变量。

Java 不支持多重继承，指在 Java 中一个类不能有一个以上的父类。

继承和封装具有很好的合作性。如果一个给定类封装了某些属性，那么它的任何子类将继承这些属性并增加它们特有的属性。

1.1.3 多态

对象中的方法是通过参数来传递信息的。这些参数作为函数的输入需要利用方法加以执行。

为了在大部分函数性编程语言中完成两个不同的任务，需要给两个函数定义不同的名字。多态意味着一个对象具有多个面孔，允许方法根据传递过来的参数类型采用不同的实现。比如，可能有两个方法都叫 print()，一个方法是在屏幕上显示字符串，另一个方法则是在屏幕上显示一个位图。当调用 print()时，到底激活哪一个函数取决于调用的参数是字

符对象还是位图对象。

多态性有时也指方法的重载。方法的重载是指同一个方法名在上下文中有不同的含义，是让类以统一的方式处理不同数据类型的一种手段。它是静态的，这是因为在实现类并编写方法之前要考虑到将要遇到的所有数据类型。在某种程度上，这是非常必要的而且也能得到清晰和可预料的代码，然而它不灵活，在许多代码被冻结或没有源代码的情况下经常需要扩充环境。子类在动态运行时提供了更丰富的多态性。

1.1.4 Java和C++的不同点

C++适应了软件工程界面向对象的新潮流，但 C++并不能真正满足面向对象的编程，而这些在后来创建的 Java 中实现了。

Java 比 C++更优秀，因为它排除了不需要的部分。从 Java 语言的起源与发展来看，一方面，它由 C++发展而来，其语言风格与 C++十分相似；另一方面，Java 又比 C++简单，它抛弃了 C++中一些并不绝对需要的内容，如：

(1) 全局变量

在 C++中全局变量作为程序的状态信号没有很好地进行封装。而在 Java 中，只有类的层次名空间是全局的，不可能创建一个不属于任何类的全局变量。

(2) Goto

在 C++引入异常处理以前，goto 经常被用来在异常处理中跳出循环。Java 没有 goto 语句，Java 中严格定义的异常处理机制使 goto 没有再存在的必要，取消这种随意跳转的语句有利于优化代码以及保持系统的强健性和安全性。

(3) 指针

指针或内存地址是 C++中最有效同时也是最有害的特性。不正确的指针操作会引起许多错误。而在 Java 中是不允许直接使用指针的。

(4) 内存分配

C++中的内存分配与指针操作有同样的危险。其内存分配是通过 malloc()和 free()库函数以及 new 和 delete 两个运算符来实现的。程序员需要自己释放空间。Java 中没有 malloc 和 free 函数。由于每个复杂的数据结构都是对象，它们通过 new 运算符在内存堆上分配空间。一旦不再访问对象，占据的内存空间就会被回收。因此根本不需要 free 和 delete。这种技术被称为“垃圾回收”。

(5) 脆弱的数据类型

不同的 C++编译器根据不同机器的实际配置分配给数据类型以不同的字长，如 16 位、32 位或 64 位。而在 Java 中，所给定的基本数据类型确定一个合理的字长并保持不变。Java 解释器的这种严格与硬件无关的数据类型很难进行优化和实现。但这是唯一能够保证跨平台实现的途径。

(6) 分离的头文件

C++有头文件，而 Java 中没有头文件。

(7) Java 不直接支持多重继承

C++具多重继承的特性。

1.1.5 Java技术的特点

Java 技术能迅速在全世界传播，是因为它拥有一些优秀的技术特性，主要有以下几个方面。

(1) 简单性

由上一节的讨论可知，Java 从 C++而来，原来 C++的程序员很容易转向 Java；另外 Java 比 C++简单，更容易学习，其程序的可读性也增强。

(2) 可移植性

Java 定义出自己的一套虚拟机，以及这套虚拟机上使用的机器码——JavaBytecode。Java 通过预先将源代码编译为接近于机器指令的字节码，有效地克服了传统解释型语言的性能瓶颈。又由于解释执行 JavaBytecode 只需要 Java 运行系统而不需要某个具体平台的支持，因此，这一点保证了 Java 运行环境与平台的无关性，所以移植很方便。

(3) 分布性

Java 支持 WWW 的客户机/服务器计算模式。通过 Java 提供的类库可以处理 TCP/IP 协议，使用户可以通过 URL 地址在网络上访问对象。所以，Java 是一门适合 Internet 和分布式环境的技术。

(4) 纯面向对象

Java 是一种完全面向对象的程序设计语言，Java 程序代码以类的形式组成，抛弃了 C++中非面向对象的特性。

(5) 结构中立性

Java 采用了完全统一的语言文本，Java 的基本数据类型不会随机器的变化而变化，如一个整型数总是 32 位。不像 C++，随着机器的不同，其整型数的长度是不相同的。此外，同时 Java 语言环境还定义了一个用于访问底层 OS 功能的扩展类库，以使 Java 的应用程序能够不依赖于具体系统。

(6) 稳定性和安全性

Java 不支持指针数据类型，也不允许直接对内存进行操作。Java 还提供了内存管理机制，即一个自动的“垃圾回收”功能。

在本节中，我们讨论了面向对象程序设计的一些概念，以及面向对象的三个主要特性：封装、继承和多态。对 Java 和 C++两种编程语言进行了比较，同时指出了 Java 的一些优秀技术特性。Java 的这些技术特性使它能够在全世界得到迅速传播。

1.2 一个最简单的Java应用

Java 源文件是一个或多个类定义组成的文本文件。这些文件以.java 后缀名保存。Java 源代码被编译后，每个类分别保存在按类来定义的带有.class 后缀的输出文件中，由于 Java 没有全局函数或变量，Java 源文件只保存一个或多个类定义，这种限制使小程序更复杂。

现在让我们编写、编译并运行一个标准“Hello World!”应用程序，如程序 1.1 所示。

程序 1.1 标准“Hello World!”应用程序

```
class HelloWorld {
```

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

该文件命名为 HelloWorld.java。要编译这个文件，需要运行 Java 的编译器 javac，并在命令行输入源文件名，如下所示：

C:\>javac HelloWorld.java

Javac 编译器会产生一个名为 HelloWorld.class 的与处理器无关的 bytecode 程序。可以通过将类名 HelloWorld 当作 Java 的命令行参数来执行：

C:\>java HelloWorld

输出结构为：

Hello World!

这样一个最简单的 Java 应用便写成了，就这么简单。

要把文件名与类名取成一样，且两者的大小写要一致。

1.3 Java的数据类型

在 Java 中，数据类型可以分为两大类：基本类型和引用类型。基本类型包括整型、浮点型和布尔型。引用类型包括类、接口和数组。

基本数据类型和引用数据类型的区别在于：基本类型的数据在函数的调用中是以传值方式工作的，引用类型的数据在函数调用中是以传址的方式来工作的。

在这一节中，我们讨论简单类型和数组，类和接口将在后面讨论。在本节的最后，我们将对 Java 的关键字进行一些说明。

1.3.1 简单类型

简单类型表示不可分割的单值的表达式，如整数、浮点数、字符和布尔值。

Java 有 8 个简单类型 byte、short、int、long、char、float、double 和 boolean。它们可以分成 4 组：

整数：byte、short、int 和 long，用来表示正负数。

浮点数：float 和 double，用来表示小数。

字符：char，用来表示字母或数字。

布尔：boolean，用来表示逻辑值。

Java 编译器静态检查它所编译的代码以确保类型的正确。每种类型精确定义了它表示的一组值，以及一组可以在其上进行的操作。

1. 整数

Java 取消了 unsigned 含义，所以 Java 的整数类型都是有符号的。因 Java 缺少无符号