高等职业教育电子信息类贯通制教材（计算机技术专业）

Java 语言编程基础

武马群　赵丽艳　主编

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

内 容 简 介

本书对如何使用 Java 语言进行程序设计做了详细的介绍。全书共分 11 章，包括 Java 语言概述，标识符、关键字和数据类型，表达式和流程控制，数组与字符串，对象、类和方法，Java 语言中的接口、包和异常，Java 语言的输入与输出，Java 语言的图形用户界面，多线程，多媒体编程和网络编程。每章后附有习题，附录中有使用 Java 语言需要注意的问题、JDK 介绍和 Java 语言的内部关键字。

本书内容详尽，实例丰富，列举的实例都在 JDK1.3 的 Windows 环境下编译通过。

本书既可作为高职高专院校教学用书，也可供信息技术领域初、中级读者自学使用。

图书在版编目（CIP）数据

Java 语言编程基础/武马群，赵丽艳主编. —北京：电子工业出版社，2004.1
高等职业教育电子信息类贯通制教材·计算机技术专业
ISBN 7-5053-9378-2

Ⅰ. J… Ⅱ. ①武… ②赵… Ⅲ. Java 语言—程序设计—高等学校：技术学校—教材 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2003）第 125609 号

责任编辑：刘文杰 特约编辑：王银彪
印 刷：北京彩艺印刷有限公司
出版发行：电子工业出版社
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：787×1 092 1/16 印张：13.5 字数：346 千字
印 次：2004 年 1 月第 1 次印刷
印 数：5 000 册 定价：18.00 元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系电话：（010）68279077。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

前　言

近年来，随着 Internet 以及信息技术的发展，人们对网络的需求越来越大，同时对网络应用程序的要求也越来越高，特别是一些能在不同的操作系统平台上运行的应用程序。Java 语言的出现恰恰适应了这种跨平台应用程序的需求，并且得到了广泛的应用。它不仅仅是一种程序设计语言，更是一个网络操作系统。通过对 Java 语言的使用，原本静止的 Web 页面可以变成生动诱人的动画。不仅如此，Java 语言的出现，也使信息技术的应用和影响扩大到空前的范围，它从根本上改变了网络应用程序的开发和使用方式。到目前为止，Java 语言已经在整个 Internet 网络中使用。

本书主要阐述了 Java 语言的基本原理和使用方法。全书共分 11 章。第 1 章 Java 语言概述，全面介绍了 Java 语言的基本情况，引导读者初步认识 Java 语言。第 2~4 章介绍 Java 语言的基础知识，这是必须掌握的基本内容，如果有 C/C++的基础，这部分内容学起来会很轻松。第 5 章对象、类和方法，介绍了 Java 语言面向对象的程序设计。第 6 章 Java 语言中的接口、包和异常，学完本章之后读者会感觉到这些抽象的概念并不难以理解。第 7 章 Java 语言的输入与输出。第 8 章 Java 语言的图形用户界面。第 9 章多线程。第 10 章多媒体编程。第 11 章网络编程是 Java 语言高级编程的内容。

本书的实例都在 JDK1.3 的 Windows 环境下编译通过。每章后附有习题，用于复习和巩固读者所学到的知识，以便于增加读者的实际编程经验。本书既可作为高职高专院校教学用书，也可供信息技术领域初、中级读者自学使用。

本书第 1, 2 章由武马群编写；第 6, 7, 8 章由侯燕萍编写；第 10 章由刘妍东编写；第 3, 4, 5, 9, 11 章和附录由赵丽艳编写。全书由武马群、赵丽艳主编。在这里，我们向所有为编写此书提供帮助的人们表示衷心的感谢。

本书还配有教学指南、电子教案及习题答案（电子版），请有此需要的教师与电子工业出版社联系，我们将免费提供。E-mail：ve@phei.com.cn

由于作者水平有限，书中会有许多缺点和不足，恳请广大读者批评指正。

编　者

2003 年 10 月

目　录

第 1 章　Java 语言概述

最近几年，随着全球 Internet 的迅猛发展及万维网 WWW（World Wide Web）的日益普及和快速增长，整个计算机环境正在经历着深刻的变革。1989 年，超文本标记语言 HTML（Hypertext Markup Language 超文本标记语言）和万维网 WWW 的产生是 Internet 数据描述语言的一次飞跃。Java 语言产生后，由于它独具特点，逐渐成为在 Internet 网络以及操作系统等其他方面的最受欢迎的开发与编程语言。

本章将向读者介绍下列主题：

- Java 语言的起源；
- Java 语言的特点；
- Java 语言诉编程特色；
- Java 虚拟机；
- 面向对象编程技术。

1.1　Java 语言的起源

Java 语言是作为一个研究项目的一部分而产生的，该项目旨在开发用于种类繁多的连网设备和嵌入系统的高级软件。Java 技术是在一个非公开的小型项目中作为编程工具而创建出来的。该项目组是由 Sun 公司的 3 个人于 1991 年发起的，那时他们的目的是创建一种可用于交互性手持式家庭娱乐设备控制器的语言，以实现一些家庭娱乐和家用电器的功能。当时设计者的全部设想是，如何让电冰箱在牛奶变坏时通知使用者、向微波炉发送信息以告诉使用者怎样处理剩菜以及制造出智能灯泡。

小组成员称此新语言为 Oak（橡树），据说这不是因为别的，只是他们的窗外正好有一棵橡树。该小组曾努力为此类装置寻找市场，而当时电视机顶盒及视频点播行业可能最需要这类装置。然而这两个行业都未能让他们乐观，在家用电器上使用这些功能好像不是特别有市场，也没有什么太大的用途。1994 年 Internet 开始盛行，这是计算机世界的一大幸事。该小组意识到，Oak 完全符合在 Internet 上编写、发送和使用应用程序的方式。

Java 技术是独立于平台而设计的。Internet 是世界上最大的客户/服务器系统，它拥有各种不同类型的客户机。Web 设计者肯定无法做到对可能访问其页面的每一台计算机编写不同的程序。Java 技术还可以在 Web 页面内移动对象，而 HTML 技术则不能。

Java 技术于 1995 年公诸于世，而且该语言的第 1 个非试用版本于 1996 年发布。Java 语言伴随着 Internet 迅猛发展而发展，它的发展速度远远超过了其他编程语言，成为当今推广最快的一门计算机程序语言。另外，Java 技术是完全免费的，可以从 Sun 公司的网站上下载。

1.2 Java 语言的特点

Java 语言到底是一种什么样的语言呢？它是一个简单的、面向对象的、网络适用的、解释型的、健壮的、安全的、独立于平台的、可移植的、可扩展的、高性能的、多线程的以及动态的程序设计语言。

1．Java 语言的简单性

Java 语言最初是为对家用电器进行集中控制而设计的一种语言，因此它必须简单明了，易于学习。Java 语言通过提供最基本的方法来完成指定的任务，只需理解一些基本的概念，就可以用它编写出适合于各种情况的应用程序。Java 语言是在 C、C++语言的基础上产生的，它的风格十分接近 C++语言，但要比 C++简单得多。Java 语言略去了 C++语言中容易引发程序错误的地方，例如指针和内存管理。在高级编程语言的所有特性中，不是绝对需要的 Java 语言都已经删去了，例如 Java 语言没有算符重载、预处理、多维数组、多重继承等。增加自动垃圾收集功能，用于回收不再使用的内存区域。这不但使程序易于编写，而且大大减少了由于内存而引发的问题。Java 语言为程序开发者提供了丰富的类库，使程序的编写变得容易、简单。

Java 语言的简单性还体现在小型化上。Java 解释器、系统模块和运行模块都比较小，适合在小型机器上运行，也适合从网上下载。

2．Java 语言的面向对象特性

Java 语言是一种彻底的纯面向对象的程序设计语言，它具有面向对象的四大特点：封装、继承、多态和动态。Java 语言支持单继承类层次结构。这就是说，每个类一次只能继承一个别的类。Java 语言的设计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，实现了代码复用。

3．Java 语言的网络适用性

Java 语言是面向网络的语言。通过它提供的类库可以处理 TCP/IP 协议，用户可以通过 URL 地址很方便地访问网络资源。

Java 语言包括一个支持 HTTP 和 FTP 等基于 TCP/IP 协议的子库，它提供一个 Java.net 包，通过它可以完成各种层次上的网络连接。因此，Java 语言编写的应用程序可凭借 URL 打开并访问网络上的对象，其访问方式与访问本地文件系统几乎完全相同。Java 的语言的另一个 Socket 类提供的可靠流式网络的连接，使程序设计者可以非常方便地创建分布式的客户机（client）和服务器（server）应用程序。

4．Java 语言的类、类装载性

Java 语言提供了大量的类以满足网络化、多线程、面向对象系统的需要。

（1）语言包提供的支持包括字符串处理、多线程处理、例外处理、数学函数处理等，可以用它简单地实现 Java 语言程序的运行平台。

（2）实用程序包提供的支持包括哈希表、堆栈、可变数组、时间和日期等。

（3）输入、输出包用统一的“流”模型来实现所有格式的 I/O 操作，包括文件系统、网络、输入、输出等。

（4）低级网络包用于实现 Socket 编程。

（5）抽象图形用户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、选单、滚动条、对话框等，使得 Java 语言可以移植到不同平台的机器。

（6）网络包支持 Internet 的 TCP/IP 协议，提供了与 Internet 的接口。它支持 URL 连接，WWW 的即时访问，并且简化了客户/服务器模型的程序设计。

5．Java 语言的健壮（鲁棒）性

Java 语言在编译和运行时，要对可能出现的问题进行检查，以防止错误的产生。它提供自动垃圾收集来进行内存管理，防止程序员在管理内存时产生错误。通过集成的面向对象的例外处理机制，在编译时，Java 语言提示出可能出现但未被处理的例外，帮助程序员正确进行选择以防止系统的崩溃。另外，Java 语言在编译时还可以捕获类型声明中的许多常见错误，防止动态运行时不匹配的出现。

6．Java 语言的安全特性

用于网络、分布环境下的 Java 语言必须防止病毒的入侵，Java 语言不支持指针，一切对内存的访问都必须通过对象的实例变量来实现，这样就防止了程序员使用“特洛伊木马”等欺诈手段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。

7．Java 语言的体系结构中立特性

网络一般由各种类型的计算机构成，Internet 也是这样。为了使 Java 程序在任何地方都能运行，Java 解释器生成了与体系结构无关字节码（bytecode）指令，只要安装了 Java 运行时系统，Java 程序就可以在任意的处理器上运行。这些字节码指令对应于 Java 虚拟机中的表示，Java 解释器得到字节码后，对它进行转换，使之能够在不同的平台上运行。

8．Java 语言的可移植特性

与平台无关的特性使 Java 程序可以方便地移植到网络上的不同机器。同时，Java 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 运行时系统有标准 C 语言实现，这使得 Java 系统也具有可移植性。

9．Java 语言的解释特性

Java 语言是解释执行的。程序运行时，Java 解释器直接对字节码进行解释执行。字节码本身携带了许多编译信息，使得连接过程更加简单。

10．Java 语言的高性能

Java 语言的解释器和其他解释执行的语言如 BASIC 不同，Java 语言字节码的设计使之能很容易地直接转换成对应于特定 CPU 的机器码，从而得到较高的性能。

11．Java 语言的多线程

多线程机制使应用程序能够并行执行，并且同步机制保证了对共享的数据的正确操作。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。

12．Java 语言的动态特性

Java 语言的设计使它适合于一个不断发展的环境，在类库中可以自由地加入新的方法和事例变量而不会影响用户程序的执行。另外，Java 语言通过接口来支持多重继承，使之比严格的类继承具有更灵活的方式和扩展性。

1.3 Java 语言与 C、C++语言的区别

对于变量声明、参数传递、操作符、控制语句等方面，Java 语言使用了和 C、C++语言相同的风格，使得熟悉 C、C++语言的程序员很方便地进行编程。同时，Java 语言为了实现其简单性、分布性、安全性等特性，抛弃了 C 和 C++语言中许多不合理的内容。下面介绍 Java 语言和 C、C++语言的区别。

1．全局变量

在 Java 语言程序中，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的、静态的变量来实现一个全局变量。例如：

```
c1ass GlobalVar{
public static global_var；
}
```

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

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

2．goto 语句

Java 语言不支持 C、C++语言中的 goto 语句，而是通过例外处理语句 try, catch, finally 等代替 C++语言中用 goto 语句处理遇到错误时跳转的情况，使程序更可读且更结构化。

3．指针

Java 语言不支持 C、C++中的 goto 语句，因为由指针所进行的内存地址操作会造成不可预知的错误，同时通过指针对某个内存地址进行显式类型的转换后，可以访问一个 C++语言中的私有成员，从而破坏安全性，造成系统崩溃。Java 语言对指针进行完全的控制，程序员不能直接进行任何指针操作，例如，把整数转化为指针，或者通过指针释放某一内存地址等。同时，数组作为类在 Java 语言中实现，很好地解决了数组访问越界这一 C、C++语言中不做检查的错误。

4．内存管理

在 C语言中，程序员通过库函数 malloc()和 free()来分配和释放内存，C++语言中通过运算符 new 和 delet 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 语言中，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的运行而改变，Java 运行系统对此自动地进行管理并且进行垃圾（无用内存）收集，有效地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++语言中，对于不同的平台，编译器对于简单的数据类型如 int 与 float 等分配不同长度的字节数；但在 Java 语言中，对于这些数据类型总是分配固定长度的位数，这就保证了 Java 语言的平台无关性。

6．类型转换

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

7．头文件

在 C、C++语言中，用头文件来声明类的原型以及全局变量、库函数等，在大的系统中维护这些头文件是很困难的。Java 语言不支持头文件，类成员的类型和访问权限都封装在一个类中，运行时系统对访问进行控制，防止对私有成员的操作。同时，Java 语言中用 import 语句来与其他类进行通信，以便使用它们的方法。

8．结构体和联合体

在 C、C++语言中，结构体和联合体中所有成员均为公有，这就带来了安全性问题。Java 语言中不包含结构体和联合体，所有的内容都封装在类中。

9．宏定义

在 C、C++语言中，用宏定义来实现的代码给程序的可读性带来了困难。Java 语言不支持宏，它通过关键字 final 来声明一个常量，以实现宏定义中广泛使用的常量定义。

1.4　安装和设置 JDK

JDK 可以免费从 Internet 上获得，很多网站都提供了 JDK 的下载，如：http://java.sun.com/products/jdk/站点。Sun 公司提供了多种操作系统平台的 JDK，这里以最常见的 Windows 操作系统为例，说明 JDK 的安装和设置。本书使用的是 J2SDK1.3.0 的版本，安装程序名为“j2sdk1_3_0.exe”。

在相应的目录下双击安装文件，根据提示设置要安装的组件（“Program Files”和“Native Interface Header Files”必须选中）和安装目录（本例为“D:\jdk”）后单击“Next”按钮，最

后单击“ Finish ”按钮结束安装。

JDK 安装完毕后，需要进行环境变量的设置。

1.4.1 Windows 95/ Windows 98 操作系统环境变量的设置

环境变量的设置需要修改一下系统盘上的“ Autoexec.bat ”文件。该文件是一个批处理文件，在系统启动的时候自动运行。

用记事本打开“ Autoexec.bat ”文件，在其后追加以下内容：

```
PATH=%PATH%;D:\jdk\bin
Set CLASSPATH=.;D:\jdk\lib\tools.jar;D:\jdk\lib\dt.jar
```

注意：因为在本例中 JDK 安装在 D:\jdk 下面，所以上面用到了 D:\jdk。如果在用户的机器上，需要将其改变为用户的实际安装目录。

重新启动计算机，JDK 安装设置完毕。

1.4.2 Windows 2000 操作系统环境变量的设置

在桌面上右击“ 我的电脑 ”，选择“ 属性 ”，弹出“ 系统属性 ”对话框，单击“ 高级 ”标签，弹出如图 1.1 所示的“ 环境变量 ”对话框。

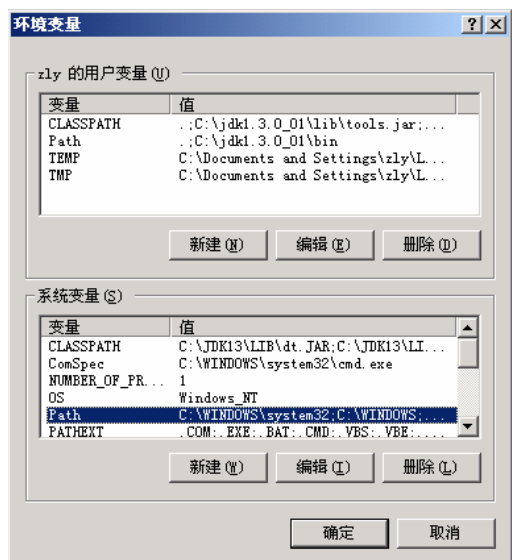

图 1.1 “ 环境变量 ”对话框

在“ 系统变量 ”一栏里选择 Path，单击“ 编辑 ”按钮，弹出“ 编辑系统变量 ”对话框，如图 1.2 所示。

在“ 变量值 ”所对应的编辑框中，输入系统变量 PATH 的值：%PATH%;D:\jdk\bin。也可以在原来的值后面加上以下内容：;D:\jdk\bin。单击“ 确定 ”按钮，然后在“ 环境变量 ”对话框中单击“ 新建 ”按钮，弹出“ 新建系统变量 ”对话框，如图 1.3 所示。

在“ 变量名 ”所对应的编辑框中输入变量 CLASSPATH，然后在“ 变量值 ”所对应的编辑框中输入系统变量 CLASSPATH 的值：.;D:\jdk\lib\tools.jar;D:\jdk\lib\dt.jar。

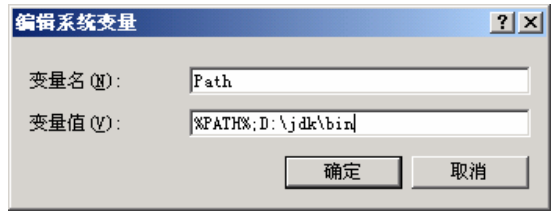

图 1.2　“编辑系统变量”对话框

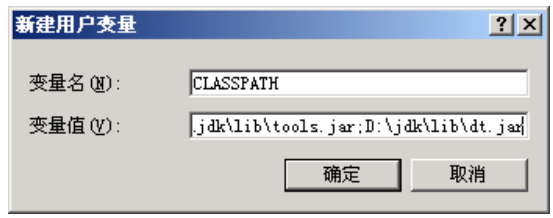

图 1.3　“新建系统变量”对话框

1.5　Java 语言程序举例

使用 Java 语言可以编写两种类型的程序：应用程序（application）和小应用程序（applet）。应用程序是可以在操作系统中独立运行的程序。小应用程序类似于应用程序，但不能独立运行，必须遵循一套约定，运行在浏览器上，执行较小的任务。

1.5.1　Java 语言应用程序（Java Application）

Java 语言应用程序是在单机或在网络上运行的一种程序，它与计算机的操作系统进行交互。

1．创建 Java 源程序

用文本编辑器创建一个名为 WelcomeApplication .java 的文本文件。下面给出该文件的完整内容，请读者务必注意其中的大小写。

例 1.1　一个简单的 Java 源程序，结果如图 1.4 所示。

```
public class WelcomeApplication  {          //创建 WelcomeApplication 类
  public static void main(String argv[]) { //创建 main 方法
    System.out.println("Welcome to Java programming!");
      }                                     //结束 main 方法定义
}                                           //结束类 WelcomeApplication 定义
```

以 WelcomeApplication .java 为文件名，保存该文件并退出编辑器。

2．编译 Java 源文件

从命令行转到 WelcomeApplication .java 文件所在的目录，然后键入以下命令：

javac　WelcomeAppliction .java

此命令将编译源代码，如果编译成功，将在当前目录下产生一个名为 WelcomeApplication.class 的文件。如果编译失败，请仔细检查源程序及编译命令。注意 Java

是区分大小写的。

3．运行应用程序

java WelcomeApplication

此命令为运行所编写的应用程序，在屏幕上会看到如图 1.4 所示的输出结果显示。

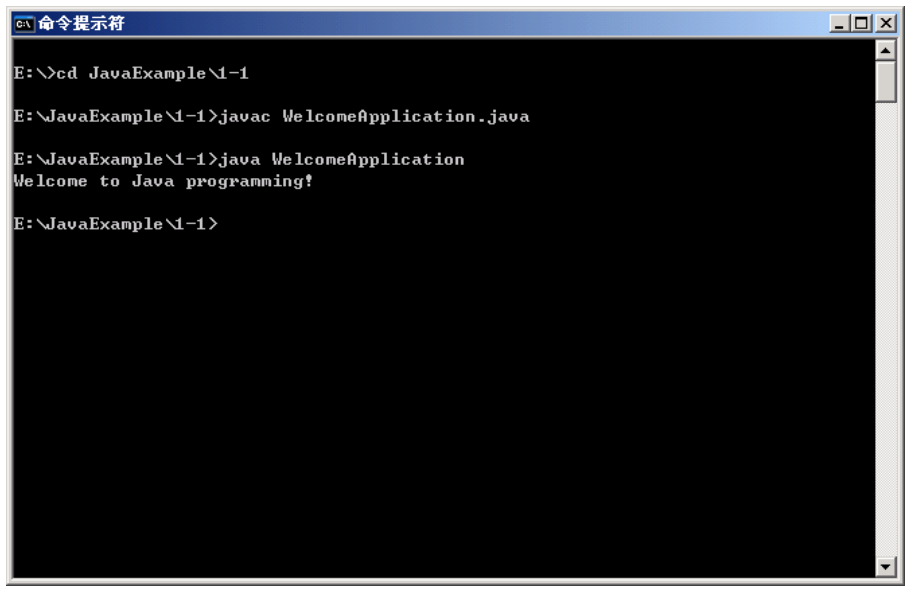

图 1.4 运行应用程序的输出结果显示

1.5.2 Java 语言小应用程序（Java Applet）

Java Applet 与应用程序不同，它们不是完全独立的程序，而更像是应用程序的片段。Java Applet 一般是非常小的程序，运行于浏览器中，具有非常特殊的属性。

编写“WelcomeApplet”小应用程序的过程类似于“WelcomeApplication”应用程序。

1．创建 Java 源程序

用文本编辑器创建一个名为 WelcomeApplet.java 的文本文件。下面给出该文件的完整内容，请读者务必注意其中的大小写。

例 1.2 一个简单的 Java Applet 源程序，输出显示结果如图 1.5 所示。

```
import  java .applet.Applet;                    //引入 Java 系统类 Applet
import  java. awt.Graphics;                     //引入 Java 系统类 Graphics

public  class  WelcomeApplet  extends Applet { //创建 WelcomeApplet 类
  public void paint(Graphics g) {               //创建 paint 方法
     g.drawString("Welcome to Java programming!" ,30,50);

  }                                               //结束 paint 方法定义
}                                               //结束类 WelcomeApplet 定义
```