软件开发伴侣丛书

如何使用 Visual Java 开发程序

荷塘月色创作组

北京理工大学出版社

内 容 简 介

本书通过一些有趣的例子，试图向读者展示 Java 语言的强大功能和高级技巧。其中包括各种图形控件的使用、声音程序的开发、网络程序的开发、三维动画程序的开发及利用 DAO 和 RDO 的 Java 接口调试数据库程序的方法和技巧等。本书中的各段源程序和代码均是作者在日常工作中逐渐积累，并通过测试，希望为读者在工作中起到抛砖引玉的作用。

本书适用于 Java 程序开发人员及大学高年级学生和研究生。

图书在版编目(CIP)数据

如何使用 Visual Java 开发程序/荷塘月色创作组编著．－北京：北京理工大学出版社，1999.1

(软件开发伴侣丛书)

ISBN 7－81045－489－7

Ⅰ．如… Ⅱ．荷… Ⅲ．Java 语言-程序设计 Ⅳ．TP312

中国版本图书馆 CIP 数据核字(98)第 27119 号

责任印制：李绍英　　责任校对：李　军

北京理工大学出版社出版发行

(北京市海淀区白石桥路 7 号)

邮政编码 100081　电话 (010)68912824

各地新华书店经售

北京房山先锋印刷厂印刷

*

787 毫米×1092 毫米　16 开本　18.25 印张　448 千字

1999 年 1 月第 1 版　1999 年 1 月第 1 次印刷

印数：1－4000 册　　定价：28.00 元

前　言

值得高兴的是，历时半年，《软件开发伴侣丛书》终于和读者见面了。本丛书的作者由数名从事软件开发的清华大学的博士生和硕士生组成。在完成教研室繁重的教学科研任务同时，利用工作之余的时间完成的。我们所想的是利用自己的一些专业知识，为中国的计算机软件产业做出更多的贡献。

想写好一套关于软件开发的技术性较强的丛书是一件繁重的工作。这套丛书从选题，到丛书的风格设计，每本书的技术水平的定位，实例的选择等，都浸透了作者们的大量的心血。诚然，由于时间较为仓促，篇幅有限，作者的水平也有待提高，所以可能出现各种各样的缺点，希望读者能够指出，这也是对我们工作的一种支持。

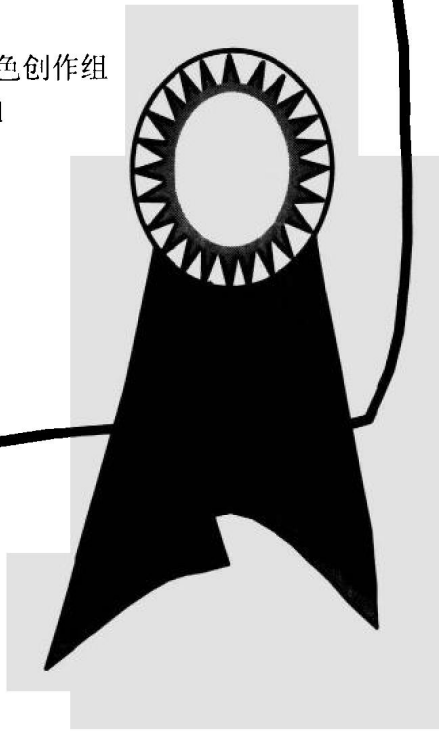
另外，我们采用的制作方式也是独特的，丛书中的每一本书，都由策划，执笔两个层次构成。我们希望，采用这种方式可以提供给读者的是一套风格统一，质量较高的精品丛书。

《如何使用 Visual Java 开发程序》是由杨恒、杨磊、黄海、孙继广、严志、杨卉共同执笔。由杨磊策划和创意的。主要内容包括：

通过程序包建立各种应用程序，其中包括各种图形控件的使用、声音程序的开发、网络程序的开发、三维动画程序的开发及利用 DAO 和 RDO 的 Java 接口开发数据库程序的编写和调试的方法以及技巧等。

荷塘月色创作组

1998.5.1



目　录

第一章 尝试第一个小例子

1.1 Java 语言简介

初学 Java 语言的人会对语法十分注意，Java 的语法与 C++的语法十分相似，本书不对此进行详细的描述。要精通 Java 语言，必须对 Java 语言有一个全面的认识。本节力图在一个较高的层次上对 Java 语言的结构进行一次全面的描述，使读者对 Java 有一个完整的了解。

1.1.1 Java 的历史

在 1990 年，Sun 公司开始了一个 Green 项目，开发用于电器中的软件。一个资深的网络软件设计专家 James Gosling 被分配到这个项目中。Gosling 开始用 C++编写烤面包机，VCR，PDA 等的软件。嵌入软件通常通过加入数字显示或用人工智能方法更好地控制其机制使电器更加智能化。然而，不久他就发现用 C++做这件事是个错误。特别是 C++使用对系统资源的直接引用，要求编程人员跟踪这些资源的管理方法，这对编程人员是一个很大的负担。这种资源管理的负担是编写可靠的、可移植的软件的障碍，也很容易使电器出现严重的问题。

Gosling 用了一个新语言 Oak 来解决这个问题。Oak 保留了熟悉的 C++语法，但省略了明确的资源引用、指针算法与运算符重载等潜在的危险特性。Oak 将内存管理直接加进语言中，编程人员可以专心于程序要完成的任务。为了成为嵌入系统编程语言，Oak 需要能在几微秒内响应实际发生的事件。Oak 还需要具有可移植性，即应当能够运行在不同的微处理器芯片和环境中。这种硬件无关特性使烤面包机制造者能改变用于烤面包机的芯片而不必改变软件，制造者也应能将烤面包机上的程序代码运行在微波炉或其它电器上。这样可以降低开发和硬件成本，同时提高可靠性。

当 Oak 成熟时,全球资讯网(WWW)也正处于高速增长的时期。当时 WWW 浏览器调用 HTML 语言写的文档，可以很容易地显示各种所需要的文本和图片，它还能提供超文本链接，使得用户可以很方便地获得需要的信息。但是，这些 HTML/WWW 浏览器技术只限于文本和图像。如果你想播放一种声音或运行一个演示程序，你不得不下载那个文件并用你本机上的能理解和运行那个文件格式的程序来播放它。Sun 的开发小组意识到 WWW 需要一个中性的浏览器，它不依赖于任何硬件平台和软件平台。它应是一种实时性较高，可靠安全，有交互功能的浏览器。而 Oak 非常适合 Internet 编程，于是开发小组决定用 Oak 开发一个新的 Web 浏览器 WebRunner。这项工作由 Naughton 和 Jonathan Payne 负责，到 1994 年秋天，完成了 WebRunner 的开发工作。在 Sun 的总裁 McNealy 和 Sun 的首席技术

官员 Bert SutherLand 和 Eric Schmidt 的支持下，WebRunner 改名为 HotJava，并于 1995 年 5 月 23 日发表。HotJava 的发表引起计算机界的轰动，确定了 Java 在网络编程中的统治地位。Java 语言最终的结构是由 James Gosling，Bill Joy，Guy Steele，Richard Tuck，Frank Yellin，和 Arthur van Hoff 在许多友人和同事的帮助下决定的。

1.1.2 Java 的特点

Java 是最好的 Internet 开发语言，它一产生就引起了整个 Internet 网络的革命。为什么 Java 会有这么大的魅力呢？这与 Java 的特点是分不开的。在这一节中，我们力图通过对 Java 特点的介绍，使读者对 Java 语言有一个整体的认识。Java 的特点主要有以下几个：

面向对象

Java 是面向对象的语言。对象就是日常生活中我们遇到的实体。在程序设计中，对象就是数据，例如图像、文本、声音等等。面向对象就是以对象为中心进行思考、采取行动。在日常生活中，我们的思维和行动都是面向对象的。面向对象的编程方法就是用日常生活中的思考方式进行程序设计。面向对象的基本机制是封装、继承和多态。

封装是一个将代码和被处理的数据包起来的保护膜。在 Java 中，封装的基本单位是类。你可以创建一个类，它是一组具有相同行为和结构的对象的一种抽象。对象是具有行为和结构的类的具体实例。封装的目的是为了减少复杂性。因此，在类中具有隐藏复杂性的机制。类中的每个方法都可以定义为公有和私有。私有的方法和数据不能被类外的其它代码所访问。而公有部分可以或必须让外界的用户知道。公有接口应认真加以设计以避免过分暴露内部实现细节。例如汽车就是一个封装起来的类，用户不必知道发动机是如何工作的，只要知道如何控制方向盘就可以了。在 Java 中,甚至连所有的变量类型，如 int，float，double，char，string 等等，都是类。

世界是由相互关联的具有层次的对象组成的。如果要有层次的描述对象，就必须要通过继承。继承和封装是紧密联系的。如果一个类封装了某些属性，那么任何子类将继承这些性质并增加了自己特有的属性。在 Java 中，所有的类都是由 Object 类派生出来的。而所有的数据类型，如 double，float，int，long 等类，都是由 Number 类派生出来的。在 Java 中每一个类只允许从一个类中派生出来，不能象 C++有多重继承，这是为了防止多重继承带来的复杂性。但是多重继承有很多好处，它更加接近现实世界的情况。在 Java 中如果要完成多重继承的任务，即一个子类要同时具有两个不相关的父类的属性，就必须通过 Java 接口（interface）。

多态是允许一个方法根据传递过来的参数类型采用不同的实现。这也被称为重载。重载的好处是当传递的参数不同时，不需要使用不同的方法名，这样给编写程序带来很大好处。例如在 String 类中，它的构造函数 String()根据传递的参数不同有 7 种之多。

例：String 类的 7 种构造函数：

```
public String();
public String(byte   ascii[], int   hibyte);
public String(byte   ascii[], int   hibyte, int   offset, int   count);
public String(char   value[]);
```

```
public String(char   value[], int   offset, int   count);
public String(String   value);
public String(StringBuffer   buffer);
```

虚拟机

Java 可以使编程人员只写一遍程序，便可以在 Internet 上任何地方运行它。Java 是通过虚拟机的技术来实现这一强大的功能的。

一般的语言，例如 C，Basic 等，编写的程序，变成计算机可以理解的机器代码的过程有两种。一种最通用的方法叫编译。一般的语言经过编译器编译后生成的是特定硬件的可执行代码。程序在运行时有很高的效率。但是，由于对于不同的硬件需要不同的程序，所以编译生成的代码可移植性较差。例如，在 Intel 处理器上编译的代码不能在 Motorola 的处理器上运行。另一种方法叫解释。在程序运行的同时，一个叫解释器的程序负责把源代码译成机器代码。解释执行程序的最大的缺点是效率低，一般来说，解释执行程序的速度比编译执行的速度慢 10 倍左右。但解释执行的程序的优势在于它支持多平台。只要不同的平台有对应于平台的不同的解释器，对于同一个程序来言，它就可以不需任何更改在所有的平台上运行了。

Java 程序的执行可以说是半编译、半解释过程。一个编写好的 Java 程序源代码，先通过 Java 编译器编译，产生出独立于平台的，Java 虚拟机（VM）上的机器码 Bytecode（如图 1-1）。例如：在不同的计算机系统中，数据类型的长度是不同的，而 Java Bytecode 中的原始数据类型的长度在任何平台上都是一样。

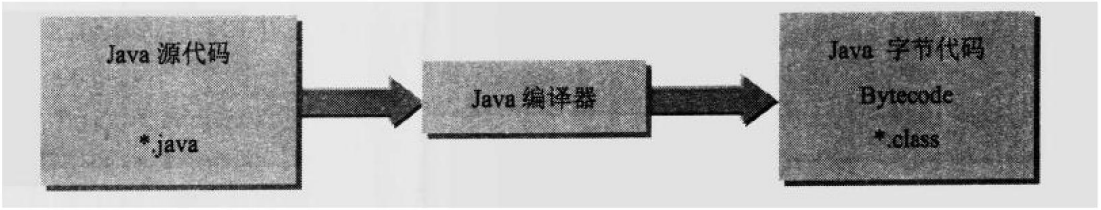


图 1-1　Java 编译器产生字节代码

Java 虚拟机是理想化的 Java 处理器芯片，通常在软件上实现而不是在硬件上实现。Java 虚拟机实际上是 Java 的解释器。在解释 Java 的 Bytecode 时，虚拟机用类装入器（Class Loader）取得 Java 类文件，再把每个类文件送入一个字节代码验证器（verifier），确保该类格式正确。再通过称为 JIT（Just in Time）编译器在执行前把字节代码变成用户机上的实际低级指令，取得较高的效率（如图 1-2）。

安全性

安全是 Internet 网上最重要的问题。网络用户最害怕保密信息被窃取和计算机系统被黑客破坏。Java 的安全机制解决了这个问题。

在 Java 的语言定义中，使用类对数据进行封装。在内存空间的安全维护方面，Java 取消了在 C 和 C++中十分重要的指针，这样可以避免用户访问不属于自己程序的内存空间，防止一些高手通过指针窃取信息和对系统进行恶意破坏。并且 Java 通过称为“垃圾回收”（Garbage Collection）的高级内存管理自动释放分配的内存。防止程序员忘记释放内存而

造成的对系统内存资源的冲击，也防止程序员不小心释放系统正在使用的内存资源，造成系统瘫痪。在 Java 还取消了在 C++中的结构概念和 typedefs、#define 语句。这大大地降低了 Java 语言的复杂性，使 Java 语言更加简单明了。在 Java 中还取消了全局变量，所有的变量只能在所定义的类中有效。Java 中还取消了类的多重继承和运算符重载，为的是减少在编程过程中出现的错误和令 Java 更容易被掌握。

当 Java 的程序在执行时，它还有三道屏障来保证程序的安全性。第一，当类装入器从

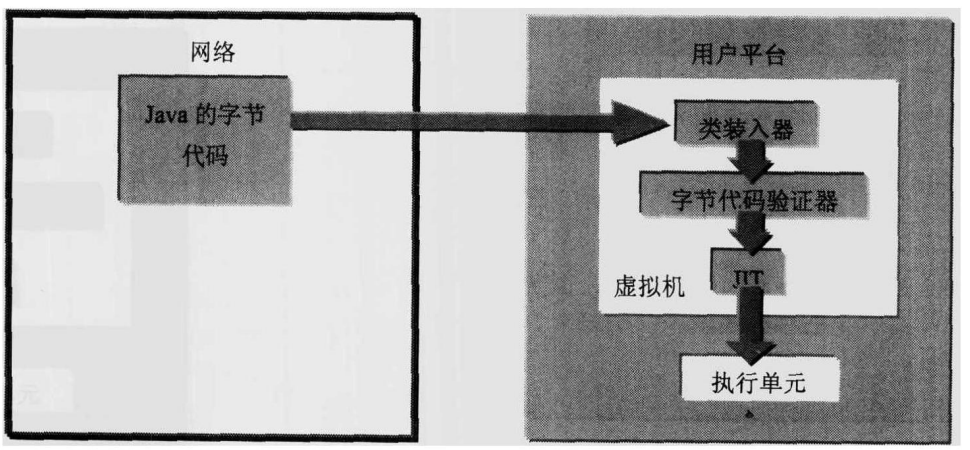


图 1-2 Java 的编译过程

网上获取类时，它把来自不同服务器的类相互分开并与本地类区别开来。通过这样分离，类装入器就可以防止网上装入的类假装成标准的内置类，或干扰从其它服务器装入类的操作。第二，由于 Java 的源代码是编译成 Bytecode 执行的，而一个程序的 Bytecode 可能是由 Java 编译器产生的，也有可能是人为直接编写的。为安全起见，Java 程序执行中由字节代码验证器对 Bytecode 做安全检查，确保 Java 程序的编译正确，遵循虚拟机的访问限制，程序不会访问不可访问的数据。第三就是程序的执行系统，也就是支持 Java 的浏览器。可以执行 Java Applet 的浏览器确定在什么条件下，程序进行什么活动，阻止 Java Applet 对系统资源的不正当使用，以及调整对 Java Applet 的处理方式。有些浏览器甚至不允许 Java Applet 对文件进行读写操作。

多线程

Java 虚拟机可以同时运行多个线程（Thread）。多线程的概念很象多任务。Java 可以把一个程序分成多个任务以便使任务易于完成和最大限度利用 CPU 资源。如果程序要完成一个费时的任务，例如从 Internet 上下载信息，可以通过多线程技术，建立下载的线程，防止用户接口变成不响应状态。在网络上，如果希望程序等待特定时间而不浪费系统资源，也可以利用多线程技术，使程序的线程进入睡眠状态。多线程还可以用于建立服务器应用程序。服务器对多用户请求的响应用多线程技术很容易实现。现在，计算机正在向并行化发展，许多高性能的计算机和网络服务器都是多处理器的。多线程程序可以有效地利用多处理器系统的处理能力。

在 Java 中，线程的产生和运行都是通过 Thread 和 ThreadGruop 类来完成的。产生 Thread

类的对象是产生线程的唯一方法。线程从产生到消失以前，一直处于产生、运行、暂停和消失四种状态之中（如图 1-3）。

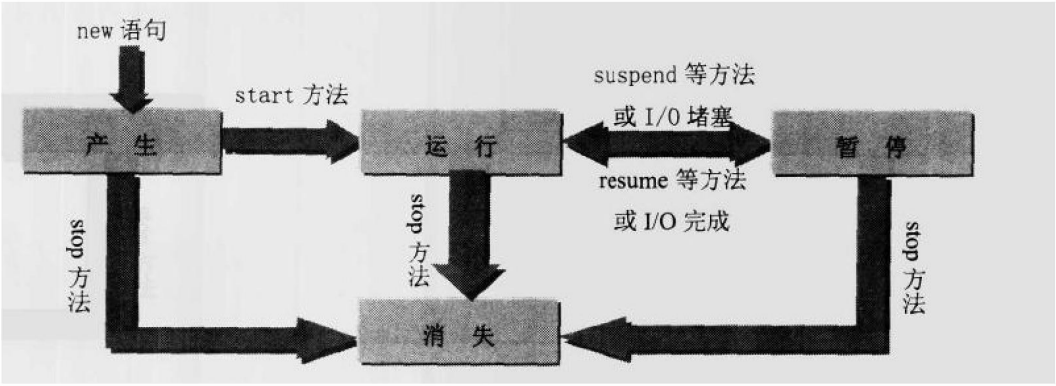


图 1-3　线程的一生只有四种状态产生、运行、暂停和消失

线程的产生是由 new 语句完成的，线程的运行是由线程的 start 方法来实现的。当线程在运行过程中用 suspend，sleep 等方法或遇到了一些 I/O 的堵塞，线程会进入暂停状态。在暂停状态中，线程不会运行，但可以返回可运行状态。如使用 resume 方法，睡眠时间结束或 I/O 操作完成时，线程又返回运行状态。当线程调用 stop 方法后，线程消失。每一个线程都有自己的优先级。当多线程对同样的资源出现竞争时，Java 是通过查看线程优先级的大小来解决的。

为了协调多线程之间的关系，使线程同步运行，Java 使用了叫监视器（monitor）的装置。监视器是一个在同一时间内只允许一个进程处理被监视器保护的代码的高层机构。监视器的主要作用是为了防止死锁。Java 可以通过 synchronized，wait，notify 和 notifyAll 等方法来防止死锁。

1.1.3　Java 包和 API

引进包的概念是为了实现访问控制和解决类名空间的冲突。包是由 Java 接口和类组合成的，通过包可以访问 Java 的 API 即应用程序接口。API 是编写 Java 程序的标准类库，是为了更快地构造软件而使用的功能库。Java 的 API 提供了一组系统平台上常用的功能，熟练的 Java 语言设计者必须充分了解 Java 的 API 的结构和功能。

包

在定义类时，为了和其它类相区别，必须给它们取不同的名字。由于 Java 的分布式的特点，在为一个类取名字时要面对整个 Internet 网，所以难免出现类名冲突。包（package）是类名字空间的一种分隔机制，也是可访问控制机制。可以将类放到包中以免包外知道类的存在。一个类在自己的包中是独一无二的，从而避免了类名的冲突。

Java 程序的第一条语句，是用 package 语句来指定程序文件中定义的类和接口所属的包。然后是用 import 语句说明要被类和接口程序引用的包。当类被程序引用时，编译器检查所有引用包，找出类的定义语句。如果多个引入包中包含同名的类，则必须用包作为前

缀以避免所指定的类产生歧义。类名和包名之间用“.”号作为分隔符。

每个包对应于文件系统中同名的子目录。包可以嵌套形成正确的层次，包的嵌套反映了文件系统中的目录层次。在 Java 编译器和 Java 运行时用 CLASSPATH 变量设置包的位置。所有 Java 内置的基本类都在名为 java 的基本包及其子包中（如图 1-4）。

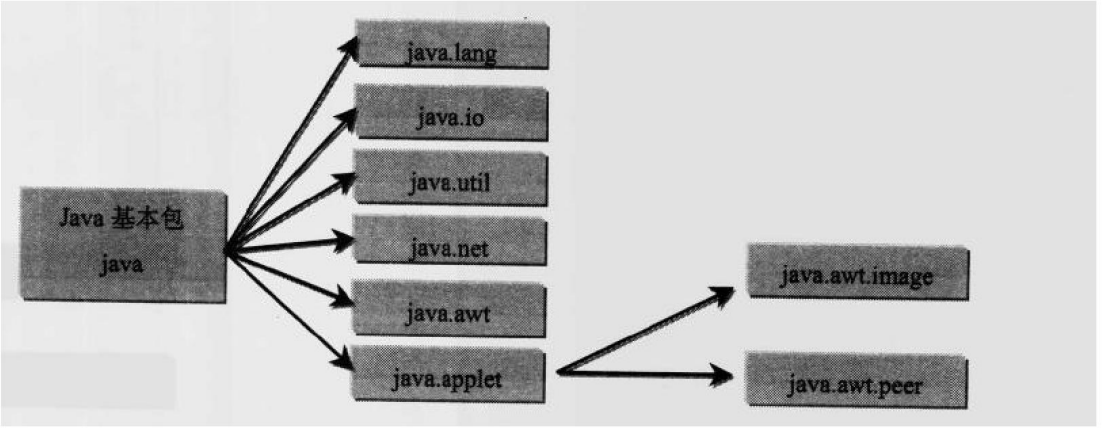


图 1-4 Java 基本内置包的结构

在 Java 编译时，编译器会自动地为程序引入一个基本包：java.lang。在所有的 Java 环境中，还有五个基本包 java.util、java.io、java.applet、java.awt 和 java.net 会在编写 Java 语言中经常用到。这些都是 Java 的 API。其中 java.lang、java.util、java.io 和 java.net 构成 Java 的核心包。而 java.applet 和 java.awt 是 Java 的小应用程序包和 Window 工具包。

核心包

Java 的核心包中有四个子包，以下我们分别介绍它们的作用。

包 java.lang 提供了构成 Java 语言和虚拟机核心的类和接口，其中最重要的类是 Object 类，它是 Java 类继承关系的根，所有 API 中的类都是由它生成的。在 java.lang 中还定义了数值类 Number（包括 int，long，float，和 double 类），字符串 String 类，运行时间类 Runtime，安全管理类 SecurityManager 和线程类 Thread。包 java.lang 是唯一的一个由编译器自动加入每个 Java 程序的包。

Java 是用输入输出流从文件字符串和其它资源中读写数据的。Java 将可读出一个字节序列的对象称为输入流，把可写入某个字节序列的对象称为输出流。输入流和输出流统称流。包 java.io 提供了操作输入输出流的类。在 java.io 包中，所有的输入操作都是抽象类 InputStream 的子类，而所有的输出操作都是抽象类 OutputStream 的子类，唯一特别的类是 RandomAccessFile 类，它可以随意访问文件和能混合读和写文件。

包 java.util 包含了各种各样有效的类和接口，包括产生各种数据结构的类。如 Dictionary、Hashtable、Stack、Vector、操作时间的 Date 类，产生随机数的 Random 类和字符串操作的类 StringTokenizer。包 java.util 还包含了 Observer 接口 Observable 类，通过这个类和接口，可以使对象在变化时通知其它的对象。

包 java.net 提供了支持网络的类，包括对 URL、TCP sockets、UDP sockets 和 IP addresses

操作的类与二进制到文本的转换器。

窗口工具包和小应用程序包

窗口工具包和小应用程序包是编写 Java 时最常用的包，其中窗口工具包是编写用户界面的有效工具，而小应用程序包是编写在 Internet 浏览器上运行的 Applet 的必要工具。下

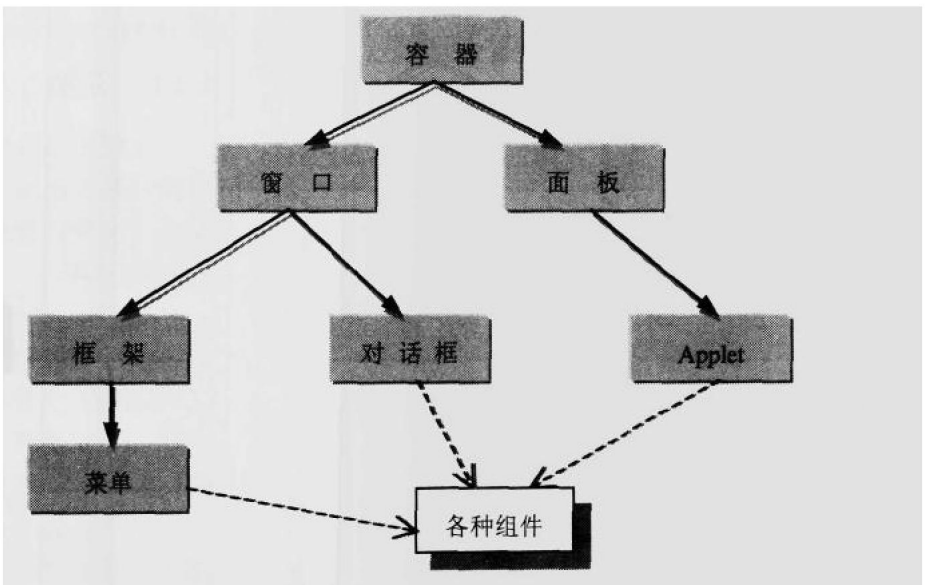


图 1-5　用窗口类构筑用户界面的框架

面对它们分别介绍。

编写窗口的类和接口都在窗口工具包 java.awt 中。AWT 就是抽象窗口工具包。在 java.awt 中集成了编写用户界面，例如窗口、对话框、按键、核对框、列表、菜单、滚动条和文本域等组件的类。在这里还定义了容器的概念，可以把各种元素组合在一起，形成一个整体。用户界面是由容器组成的容器中的组件是按照一定的层次关系组合到一起的。图 1-5 显示了这种组合关系。

在子包 java.awt.image 中提供了处理图像数据的类，包括颜色处理的抽象类 ColorModel、图像处理、设置像素点值等等类和方法。

在子包 java.awt.peer 中提供了把 AWT 成分和特定的平台（如 Micosoft Windows）操作结合起来的接口。

Java 程序有应用程序(application)和小应用程序(applet)两种。它们的区别是：application 可以脱离 Java 环境单独运行，而 applet 只能在 Internet 浏览器上运行。在形式方面，Java application 的一个类中必须有一个 main 方法，而 Java applet 程序不需要有 main 方法，但必须引入 java.applet 包。包 java.applet 中包含的类和接口使得程序可以被嵌入到 HTML 中，从而 applet 能够在例如 IE 的网络浏览器上运行。

1.2 Visual J++简介和使用 Visual J++编译器

Visual J++是 Java 和 Microsoft 的可视化开发技术的精彩结合。Microsoft 的 Developer Studio 集成了可视化界面设计、调试、代码编辑、联机帮助信息和资源编辑等多项功能。Visual J++可能是开发 Java 语言最轻松的途径了。用 Visual J++编写 Java 程序需要充分了解 Developer Studio 的功能，从而开发出更好的产品。本节的目的就在于此。

1.2.1 工程

在使用 Visual J++前，先要介绍工作空间（Workspace）和工程（project）。这两个概念指的是 Visual J++组织和管理 Java 程序的方式。由于为了用 Java 编写一个项目，往往需要多个文件，完全让程序设计者管理这些文件是十分麻烦的，所以 Visual J++提供了自动管理的工具。

在 Visual J++中，一个项目就叫一个工程，指的是一个独立的应用程序（applet 或 application），包括程序所需要的所有资源文件和编译程序中所有的设置和工具。每一个工程的所有文件都在自己的文件夹中，文件夹的名字和工程的名字相同。一个工作空间包含了一个工程所有的文件和配置信息。Visual J++不能同时打开两个工作空间。在大多数情况下，一个工作空间只包含了一个工程，这时工作空间和工程的含义相同。

在 Visual J++中，工作空间是用.dsw 文件存储的，而工程是用.dsp 文件存储的，文件名和工程名相同。.dsw 文件和.dsp 文件都是由 Visual J++自动生成的，它们都是文本文件，可以用 NOTEPAD 打开，但最好不要自己编辑它们。在工程文件夹中，除了.dsw 文件和.dsp 文件外，还有 Java 源文件.java 文件、编译生成的.class 文件和 HTML 文件等等。资源文件如图像、声音等都会在 images，audio 等子目录中找到。

1.2.2 Visual J++的界面

Visual J++的界面主要分三部分（如图 1-6），每个部分为程序设计者提供了不同种类的信息。

Project Workspace 窗口提供了工程的树状层次结构和 Infoviewer 系统（即帮助系统）的树状目录。从这个窗口中，程序设计者可以清楚地了解到工程中有哪些类，每个类中有哪些变量和方法，并且很快找到文件中定义这些类、变量和方法的位置。在编写一个较大的工程时，它可以节省编程人员的大量时间，并且使编程人员对工程结构保持清楚和直观的认识。

Editing/Infoviewer 区域用于显示 Visual J++工程的源代码和 Infoviewer 信息。源程序的编写工作就是在这一区域内完成的。

Output 窗口用于显示编译时错误信息、调试结果和在建立 Visual J++工程时产生的其它信息。如果界面上没有显示这三个窗口，我们可以通过 View 菜单打开这三个窗口。在 View 菜单中选择 Workspace 可以打开 Project Workspace 窗口；选择 Infoviewer Topic 可以打开 Infoviewer 区域；选择 Output 可以打开 Output 窗口。如果在 Editing/Infoviewer 区域中打开了几个文件，则可通过 Window 菜单选择要显示的窗口。

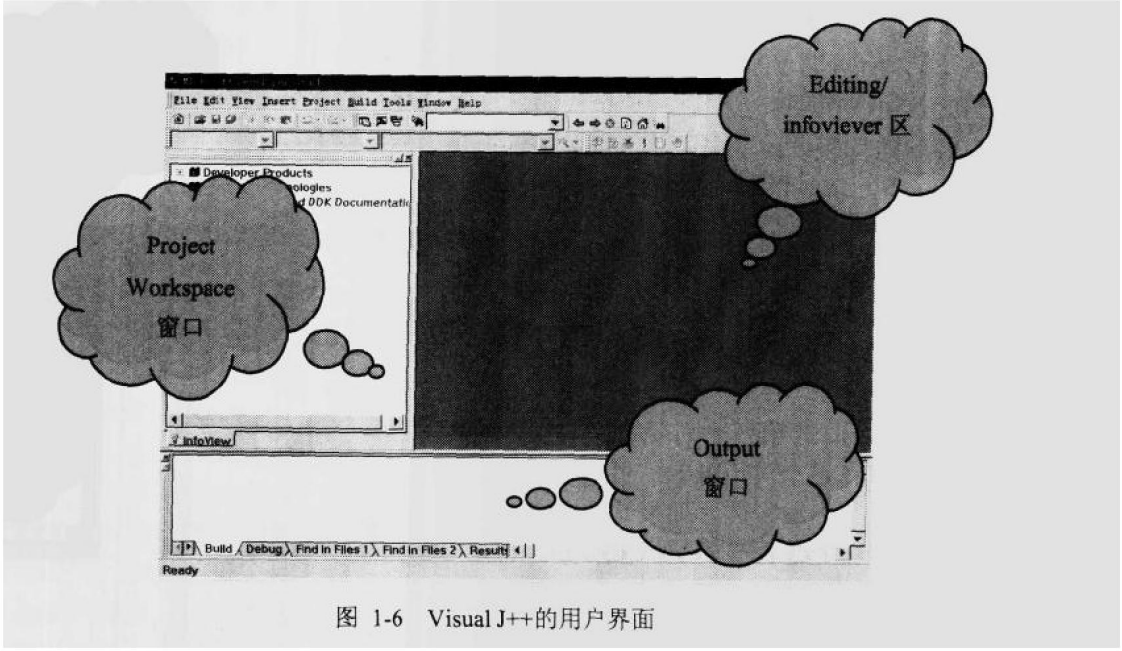


图 1-6　Visual J++的用户界面

1.2.3　使用 Applet wizard

在这一节中，我们将介绍如何使用 Visual J++的 Applet wizard 创建一个 Java Applet。我们将创建一个 Applet 在 Microsoft Internet Explorer 上显示“Visual J++”。

在打开 Visual J++的 Developer Studio 后，从菜单 File 中选取 New，就会出现了一个新的对话框（如图 1-7）。在这个对话框中，我们选择 Projects 表，这说明我们要建立一个新的工程。在工程的种类上，我们选择 Java Applet Wizard，这说明要建立一个在浏览器上运行的 Java Applet。我们在工程名字中添上 VJ，并为工程选择合适的目录，Visual J++会在所在位置自动生成一个名为 VJ 子目录。平台类型（Platforms）是工程的运行环境，在这里我们必须选择 Java 虚拟机（Java Virtual Machine），这样 Applet 才能在虚拟机的环境下解释执行。还应选择产生一个新的工程空间，而不是向现有的工作空间增加工程。

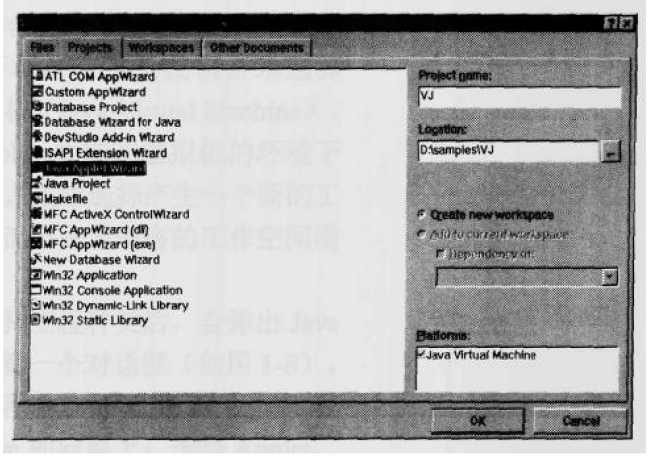


图 1-7　开始一个新的 Applet 工程

在选择工程种类后，会弹出 Java Applet 的第一个对话框（如图 1-8）。在关于工程是 Applet 还是 Applet 和 Application 的问题上，选择 Applet。这是因为我们要完成的工作是在浏览器上显示一个字符串，它只需要小应用程序就行了。

Applet 的名字在默认情况下就是工程的名字，也可以重新定义一个名字。在本例中我

们不改变默认值。

在默认的情况下，Visual J++会在所生成的代码上加注释和 TODO 提示，告诉程序设计者代码的作用或在哪个位置添加什么代码。这些注释和提示对初学者是很有用的，但对于一个有经验的程序设计者来说，可以关闭这一项。然后按 Next 键。在本例中，选择默认情况。

在 Applet Wizard 的第二个对话框中，决定是否创建一个 HTML 样板和指定初始窗口的大小。在本例中，不创建 HTML 文件和使用默认的窗口大小。

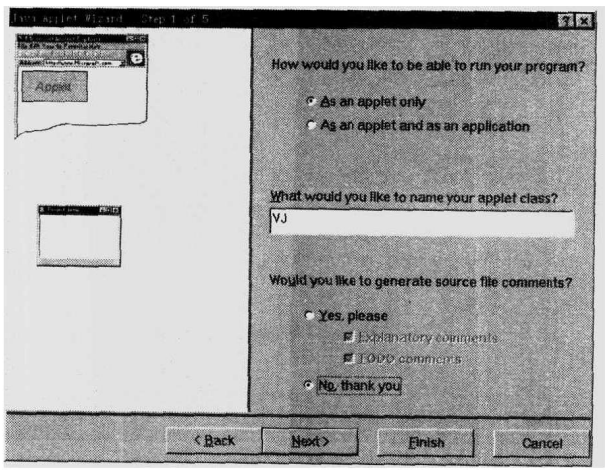


图 1-8 建立一个 Java Applet

Applet Wizard 的第三个对话框如图 1-9 所示。在这个对话框中，可以决定程序是否是多线程的。如果是多线程的，则可选择是否需要 Applet Wizard 提供的旋转地球的动画支持。在本例中，选择不是多线程的程序。在对话框中，还可以决定程序中是否需要鼠标支持。要注意的是，程序只对在程序窗口内的鼠标事件产生响应。在本例中，我们不需要鼠标支持。

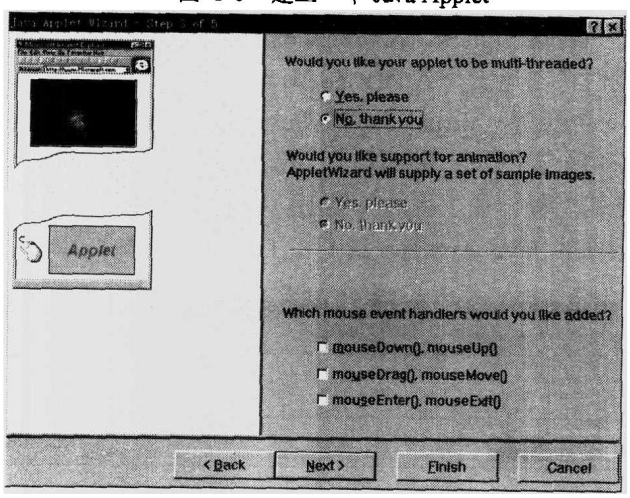


图 1-9 建立 Java Applet 的第三步

第四个对话框是说明一些参数。这些参数用于将从 HTML 文件中获得的输入传递给小应用程序。如果需要这些参数，就必须为每个参数指定名字等条目。这些条目如表 1-1。

如果在对话框中说明参数，Applet Wizard 将在源文件中生成拥有这些参数值的成员变量，还将生成 getParameter()方法用于从 HTML 文件中取得所需的变量值。在本例中，不需要这些参数。

表 1-1 在 HTML 中说明参数时需要指定的条目

Name	参数名
Member	成员变量的名字，创建这些变量是为了接受参数值
Type	成员变量的数据类型
Def-value	参数的默认值。如果 HTML 文件调用 Applet 时没有传递参数值，则使用该默认值

Name	参数名
Description	Applet Wizard 生成的描述参数的注释

最后的工作是让你输入关于小应用程序的信息，可以接收 Applet Wizard 所提供的默认值或对其编辑。在按 Finish 键后，Applet Wizard 会列出我们创建的 Applet 的所有信息。如果发现有错，可以按 Cancel 键重新设定。如果确认我们创建的 Applet 是正确的，则按 OK 键。

在 Applet 建立后，在 Project Workspace 窗口中的 Classview 表中，就会找到 VJ 这个工程。通过单击工程旁边的＋号或－号，可以展开工程的树状结构。在 Classview 表中找到 VJ 类的函数 paint(Graphics)，双击这个函数后，在 Editing/Infoviewer 区域中会出现 paint 函数的代码。把 paint 的代码改为如下：

```
public void paint(Graphics g)
{
    String s = "Visual J++";
    int window_width, window_height;
    int s_width, s_height;
    FontMetrics fontMetrics;
    Font font=new Font("TimesRoman", Font.BOLD, 36);
    window_width = size().width;
    window_height = size().height;
    g.setFont(font);
    fontMetrics = g.getFontMetrics();
    s_width = fontMetrics.stringWidth(s);
    s_height = fontMetrics.getHeight();
    g.drawString(s, (window_width-
s_width)/2, (window_height-s_height)/2);
}
```

经过选择 Build 菜单中的 Build（编译）选项和 Execute（执行）选项，Visual J++生成了名为 VJ.HTML 的超文本文件，并在浏览器 IE 显示区域正中间用 BOLD 字体 36 号大小的字显示“Visual J++”（如图 1-10）。

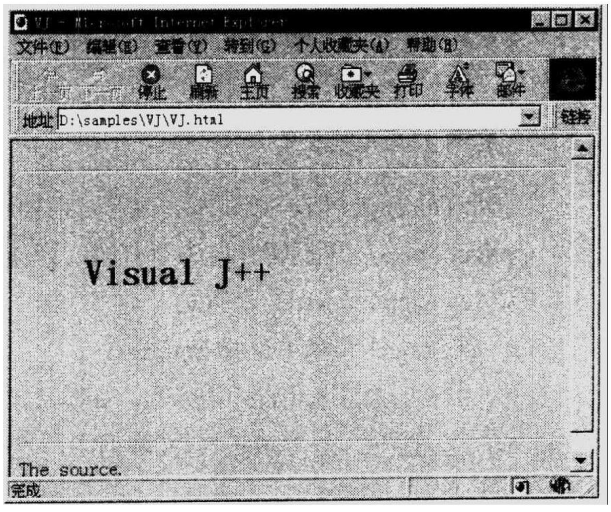


图 1-10　程序 VJ 的运行结果

在该程序中，我们使用了一个 Font 类的对象 font。通过 Graphics 类中 setFont 函数来设置显示区域中的字体。FontMetrics 类的对象 fontMetrics 是用来描述字体大小的。通过 Graphics 类中的 getFontMetrics 方法取得在显示区域中字体的大小，再用 FontMetrics 中的方法来测量要显示的字符串 s 的尺寸。

其中 stringWidth 可以测量指定字符串的宽度，getHeight 方法用来测量字体的高度，用 size()对象测量显示窗口的大小。最后，用 Graphics 类中的 drawString 方法在窗口的正中间显示字符串“Visual J++”。

1.2.4 向工程中添加元素

如果我们还要往工程中增加新的类，则可在 Insert 菜单中选择 New Class...项增加新类。也可在 ClassView 表中用鼠标左键单击工程后，在出现的快捷菜单中选择 New Class...项。出现的对话框如图 1-11。

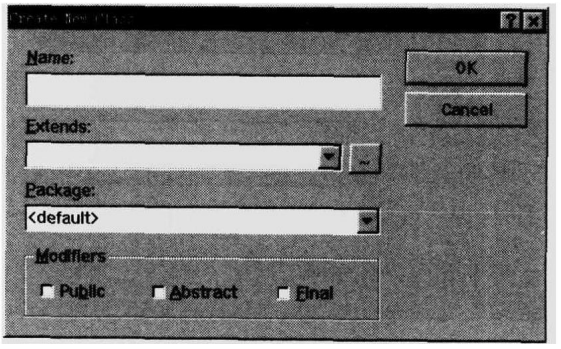


图 1-11 向工程中插入一个新类

在 Name 框中输入新类的名字，在 Extends 中输入新类的父类的名字，在 Package 中选择新类所在包的名字，在 Modifiers 中选择类的修饰符。再按 OK 键，就可以产生新类了。Visual J++会为这个类建立源代码框架，并用一个.java 文件来保存它。文件名和新类的类名是一样的。

如果想向类中增加一个方法，则用鼠标右键单击 ClassView 中的类，在出现的快捷菜单中选择 Add Method...选项，再回答 Add Method 对话框就可以了。

向类中增加一个成员变量与增加方法类似，在 ClassView 窗口中用鼠标右键单击类名，选择 Add Variable...选项。

用上面的方法新增加的元素都会立刻在 ClassView 窗口中找到，新类所在的文件还可用 FileView 访问到。我们也可以不用这种方法向工程中添加元素，而是在源代码中直接添加所需要的元素。但是，只有在经过一次编译以后，它们才能在 ClassView 窗口中找到。

1.2.5 使用 Debugger

在开始 Debugger 前，必须让编译器知道要调试的 Java 程序是 application 还是 applet。因为 application 是在单机解释器中运行的，而 applet 是在浏览器中运行的。

调试 Java application 的步骤是：

(1) 从 Project 菜单中选择 Settings。

(2) 回答出现的 Settings 对话框，如图 1-12 所示。需要调试类的名字是包含应用程序的 main()方法的类。在 Category 文本框中，选择 Stand-alone interpreter，即单机解释器，然后按 OK 键。

(3) 从 Build 菜单中，从 Start Debug 中选择需要的调试方式。

调试 Java applet 的步骤与调试 Java application 的步骤基本相同，不同的是：需要调试的类是包含 init()方法的类。Category 文本框选择的是 Browser。

在调试过程中，我们可以通过 View 菜单中 Debug Windows 中打开的各种调试窗口进行调试。例如，我们在 Watch 窗口中双击 Name 区域，输入想查看的变量名，就可以在调