

目

引言



1

第一部分 理解 Java 和 Web 的能力

1 Java 使可执行内容成为可能	9
Java 能做什么	10
Java 的起源和方向	13
Java 的能力描述	15
Java 可能实现什么	27
小结	28
2 Java 的设计是灵活的和动态的	30
向 Java 问好	30
Java 的技术纵览	35
小结	47
3 Java 改变了 World Wide Web	48
Web 纵览	49
Java 如何改变了 Web	53
Java 加进新的 Web 通信环境和能力	55
Java 和 Web 上的通信环境	56
Java 和 Web 的潜力	57
小结	59

第二部分 开发 Java 的潜力

4 使 Web 页面动起来的 Java	63
移动中的 applet	63
动画 (Animation)	71
使用 Java 的商业节点	75
小结	77
5 给 Web 页面带来交互性的 Java	80
交互的游戏	82

教育应用软件..... 85

小结..... 93

6 发布内容的 Java 95

网络传送 (distribution) 和检索 (retrieval) 的意义 95

处理新型协议和格式..... 97

从网络上检索信息 102

小结 105

第三部分 使用 HotJava 浏览器

7 HotJava 是针对 Java 的浏览器 109

Java/HotJava 的获得方式 109

下载和使用 HotJava 113

和 HotJava 一起的其它软件 118

HotJava for Windows NT 3.5 119

小结 120

8 HotJava 是新型的 Web 浏览器 121

HotJava 标志了浏览器的一个新时代 121

HotJava 特性 123

小结 126

9 用 HotJava 漫游 Web 127

HotJava 用户界面 127

用户使用 HotJava 的 Web 经验 140

小结 142

第四部分 开发 Java 应用程序

10 Java 是一门编程语言 145

Java 开发的基本流程 146

Java 编程要点 151

Java 程序的特点 152

Java 的可执行特性 174

小结 178

11 Java 是面向对象的 179

Java 中类的定义和使用 179

在 Java 中使用子类 186

小结	193
12 applet 开发工作包含设计和实现	194
考虑 applet	194
设计一个 applet	195
实现 applet	197
小结	206
A 进一步的信息资源	207
Java 信息资源	207
关于 Java 的信息资源	210
一般 Web 信息	211
B Java 语言参考	215
Java 语言总结	215
Java 语言信息	221
MIME 类型	225
C HTML 标记参考	228

引 言

滚石乐队和 Sun Microsystems 有什么共同之处？首先，两个组织都为推动他们的工作而在 World Wide Web 设立了节点。其次，两个组织都选择 Java 作为使他们的 World Wide Web 页面活起来的途径。



Java 是什么？ Java 是 Sun Microsystems 开发的一种程序设计语言，它把动画和交互作用带入 World Wide Web。HotJava 是 Sun 开发的一种用来观看 Java 程序并与之相互作用的软件。Java 释放了在 Web 上以前未能实现的一个层次的交互性。本书把 Java 在发布交互性的、可执行内容方面的能力展示和介绍给读者。

个人和组织用 World Wide Web 进行广域的和瞬时的通信。Web 利用 Internet 作为数据传输的网络，使用了一种叫超文本的表示形式，它把相关的信息用网状结构连接起来。与多媒体联合作用的超媒体网络已经为表达和通信提供了新的可能性。

但自从 Web 在 90 年代早期被引入这个世界起，Web 中始终缺少一些东西。尽管 Web 的数据传输量和 Web 上提供信息的计算机数量以惊人的速度上升，但页面的内容却缺少重要的表达性和交互性。Web 尽管经常是有趣的、信息丰富的和有用的，但其内容却达不到很多在非网络计算机上运行的多媒体和超媒体系统所具有的交互性程度。所以，尽管 Web 建立了世界范围内人与信息之间的相互联系，它只能使人们：读文本，看图像，听音乐，寻找信息。

在 Web 上无边无际的可能性可能导致使用户感到超连接只

是引导他们前进，一直走下去，Web 看起来像一条没有终点的路。Java 改变了这种情形。Java 使目的地对 Web 用户成为可能。

Java 使开发者可以编制能传送到用户自己的计算机上并在那儿运行的内容。这个软件支持编程者可以梦想到的任何东西：电子表格、教程、动画、交互式游戏。这个软件使用 Web 页面作为传送平台，可以支持一大类有真正交互性的信息任务；用户可以得到应用程序的连续的、瞬时的反馈，这种反馈可能是视觉效果、动画或计算。

处于 Java 的黎明年代的 Web 用户，确实可以在 Web 上发现一个目的地：一个可以游玩、工作和学习的地方。

Java

Java 是支持 World Wide Web 通信的集成系统的一部分。Java 是与定义超文本的标记格式，即超文本标记语言 (HTML) 完全分开的程序设计语言。Java 不能代替 HTML，它也没有忽视现在 Web 上所有已开发和应用的工作。

Java 通过一个特殊的 HTML 标记 APP 与 HTML 和 Web 相连，这个标记允许开发者把叫做 applet 的特殊的 Java 程序包括在 Web 页面中。这些 applet 是用户的浏览器（作为 Web 页面观察的一部分自动地）下载和执行的重要的软件程序。Java 通过页面上的 applet 可能的图形输入和输出，在 Web 上打开了更深层次的交互性的窗口。

本书的作用

本书的目的是让读者了解什么是 Java，它怎样改变了 Web 通信，以及 Java 的主要技术特征。

本书探讨了 Java 在动画、交互作用和分布式网络通信上提出的可能性，本书旨在帮助读者：

- 欣赏 Java 改变 World Wide Web 的精彩方式。
- 从概念上掌握使用 Java 将涉及到什么，包括 HTML-Java 连接，HotJava 浏览器和 Java 语言的角色。
- 了解 HotJava 浏览器及怎样下载和使用它。
- 理解 Java 语言是如何工作的，包括数据定义和程序控制的基础。
- 透视 Java 如何遵从面向对象的范例并鼓励软件复用。
- 看到支持动画、交互性和网络应用的 Java applet 的例子。

怎样使用本书

本书是对 Java 的精髓的介绍：Java 怎样与 Web 连接，怎样阅读 Java 程序，构造 Java 程序的基础。本书试图组织成使各种水平的用户都可用自己的方式阅读它。

本书的范围

本书讨论 Java 的意义并展示它可以实现的各种应用。本书对 HotJava 浏览器作了粗略的介绍，讲述了怎样得到它，怎样用它漫游 Web。然后，本书总结了 Java 语言的主要特征，描述了它的进程编程特征的基础及它怎样支持面向对象的范例。这个语言的描述包括很多例子，用以展示 applet 和应用的基础。最后，本书评述了很多现存的 Java 程序，并检查了它们使什么样的通信成为可能及它们如何工作。

本书可能对 Java 技术上和表达上迅速而丰富的发展方式有所忽略，所以本书在两个方面有所局限：

1. 本书覆盖从 1995 年夏天可以使用的 Java/HotJava 的 alpha 测试版，Java/HotJava 的 beta 测试版可望于 1995 年晚些时候发布。本书的支持节点 (<http://www.december.com/works/java.html>) 在 beta 版发布后将包含一个对与本书相关的修改的总结。

2. 本书并不是 Java 的一本详细参考或高级编程指导, 相反, 它只是为新用户、规划制订者、开发者提供一个快速起点, 它包括 Java 的简介性的概念和最基本的技术信息, 还对完整的和现行的参考信息的联机地址提供了指示。

本书的组织方式

本书被分成几部分, 每部分有几章。每部分从不同侧面介绍了 Java 的精髓, 每章阐述一个中心内容, 它用一个总结性句子, 作为每章标题。下面是对每部分内容的总结:

第一部分, “理解 Java 和 Web 的能力” 是对 Web 和 Java 的能力的概述。这几章展示了 Java 在传送分布性、可执行内容方面的能力, Java 的灵活性和可扩展性的技术设计, Java 作为传送网络超媒体内容的信息系统与 Web 的关系。

第二部分, “开发 Java 的潜力” 总结了 Java 在动画、交互作用和信息发布方面的能力。这几章讨论了很多展示 Java 行为的例子。读者可以尝试其中一些例子, 了解这些例子中重要的代码编程。

第三部分, “使用 HotJava 浏览器”, 是对 HotJava 浏览器的描述。第 7 章介绍了用户怎样下载和安装 Sun Microsystems 支持 Java 的 Web 浏览器的样板 HotJava。其它章展示了 HotJava 浏览器与其它 Web 浏览器的重要不同及用户怎样使用 HotJava 去漫游 Web。

第四部分, 介绍了开发 Java 程序的基础。这几章解释了 Java 的进程性特征, 以及 Java 支持类 (class) 的面向对象的特征。这个部分以一个研究 applet 设计和实现的简单例子作为总结。

附录描述了进一步的信息源, Java 语言实质的总结, Java 和 Web 发展的其它支持信息。名词表定义了本书中用过的与 Java 和 Web 相关的名词。

谁可以使用本书

- 对 Java 技术可能给自己的组织带来什么感兴趣的经理们和计划人。
- 想知道 Java 如何工作的信息开发者。
- 想尽快知道怎样用 Java 扩展他们的信息的交互性层次的 Web 开发者和编程者。
- 需要掌握 Java 精髓的、联机通信方面的或正在进行超媒体课程学习的学生。

读者怎样使用本书

- 新的 Internet/Web 用户:通过第一部分和第二部分对 Web 和 Java 如何与 Internet 相互联系有个大概了解,跳过第二部分中关于编程特征的描述。如果你的计算机可安装支持 Java 的浏览器,则读一下第三部分或去询问当地联机服务提供商或管理员。开始写 Java applet 时,读第四部分,然后再看一下第二部分中的应用程序。
- 管理者/规划制订者:第 1 章总结了有关 Java 在通信方面能力的重要概念。第 2 章是 Java 的技术总结。第 3 章更详细地探究了 Java 与 Web 的关系。对计划人用户,为访问 Java 内容,必须得到一个第三部分中描述的支持 Java 的浏览器,为制订或管理包含 Java 的编程计划,应读第二和第四部分。
- Web 开发者:直接到第二部分看一看 Java 使 Web 上什么成为可能,看一看你的平台是否可用支持 Java 的浏览器,如有可能,取一个第三部分描述的 HotJava 浏览器;阅读第四部分关于 Java 编程的综述。第一部分是 Java 的综述,特别地读一下第 3 章关于 Java 与 Web 的相互关系和 Java 使什么新的种类的发展成为可能的内容。
- 应用程序编制者:检查一下你的开发平台是否支持第三部

分描述的 HotJava。第 2 章和附录 B 是 Java 的重要技术性总结。如果你的平台可安装支持 Java 的浏览器，试一下第四部分中的例子并浏览一下第二和第一部分的例子，用本书的支持节点获取源代码的最新版本。用附录 A 中列出的联机信息源获取关于 Java 语言特点的更详细的信息。

联机支持

提示

本书支持节点 为了解到关于本书内容的最新信息，打开 URL <http://www.december.com/works/java.html>。这个支持节点提供了关于此书，关于例子的代码，关于资源和相关信息的更新的连接。检查支持节点中的勘误表找出本书的修正，可把其它错误的报告、问题或建议发给本书作者 (john @ december.com)。

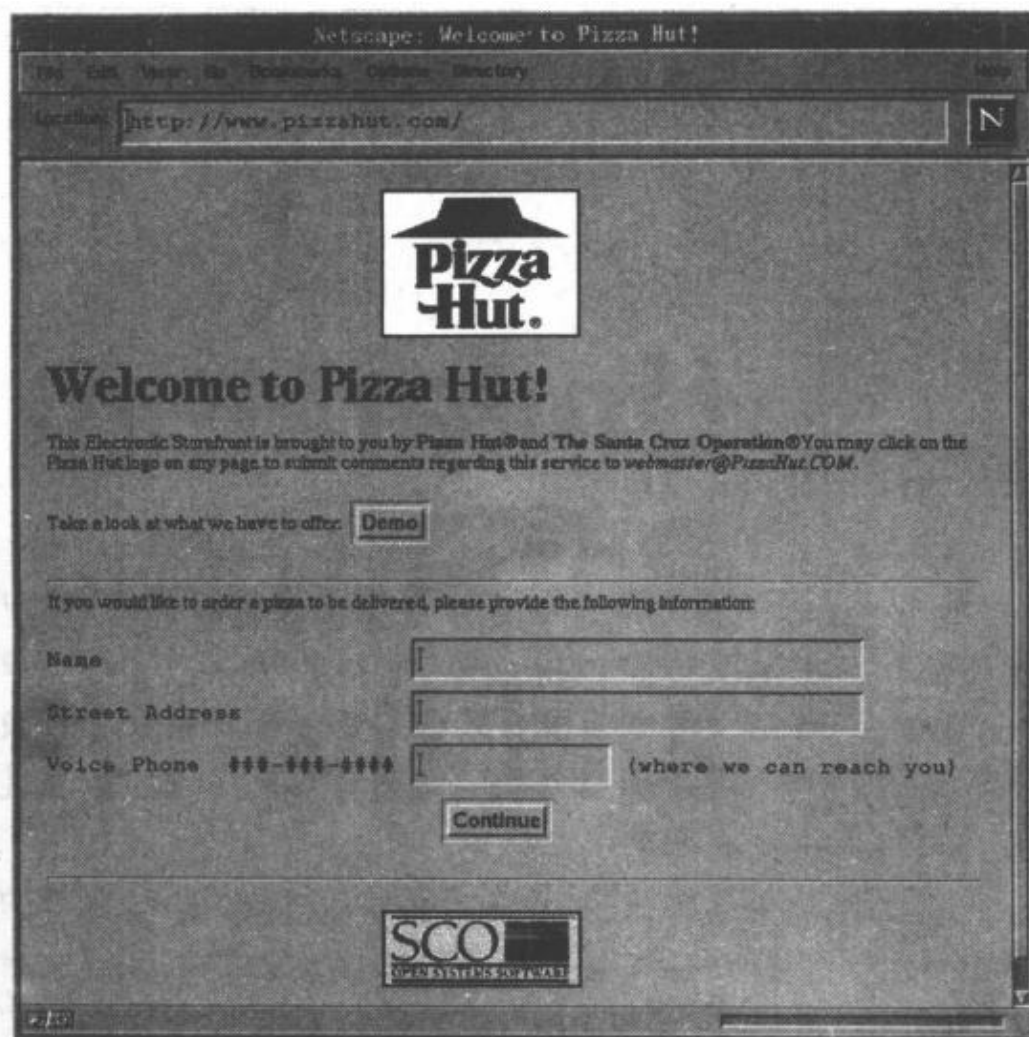


图 P1.1 Web 上交互表单的例子

如何改变了 Web。

这些章节将告诉你：

- Java 如何通过可执行内容丰富了 Web 的交互性。
- Java 如何在技术上适应 Web 通信和它作为一种灵活、动态的语言的粗略概念。
- Java 如何把 Web 的基础改变成一种合作地、非线性地组织信息的系统。

Java 使可执行内容成为可能

World Wide Web 已改变了联机世界。Web 用户再也不必被限制在文本信息系统内，而有更多选择和观看信息的机会。Web 的超文本系统给用户以高度的可选性，超媒体在用户面前拓展了更多的感官输入的选择，用户可以用 Web 接触到信息、图形、文本，甚至图像。但是，Web 缺乏真正的交互性——用户与应用程序之间实时的、动态的、可视的相互作用。

Java 引入交互性 Web，缺补了上述欠缺。用户使用一个支持 Java 的 Web 浏览器可以遇到动画和交互性应用程序，Java 编程者可使用能被任何支持 Java 的浏览器显示的、约定的媒体格式和信息协议。Java 通过让用户可以发布可执行内容——而不仅仅是 HTML 页面和多媒体文件——使 Web 上的通信、信息和交流大为丰富，这种可执行内容的发布就是 Java 的力量。

Java 从最初 Sun Microsystems 设计的一种用户编写可在很多不同设备上运行的软件的程序设计语言已发展成一种可在 Web 上发布可执行内容的语言。现在，Java 书写的 Web 页面中的应用程序可以给用户瞬时反馈，可以连续接受用户用鼠标或键盘发出的输入信息，从而使 Web 更吸引人。

本章中，作者将首先给 Java 一个描述和定义，展示 Java 给 Web 通信带来了什么。然后，作者将通过带读者欣赏一些例子了解 Java 能做些什么。如果读者想直接去了解 Java 编程，请看本书第四部分。如果不是这样，就可顺序读下去，这一章和下一章将对 Java 的能力和技术特征做个概括，为更详细地阅读本书第二部分中提供的 Java 程序的例子作好准备。

Java 能做些什么

Java 使 Web 页面动起来。并使交互性的和特定的应用成为可能。图 1.1 描述了 Web 上使用这种软件如何支持各种通信。用户通过超文本这种信息的基本组织形式可以选择看什么信息，编程者可以编写特定种类的接口程序，它们用 Web 上的这些超文本文件做为界面。当用户使用一个 Web 页面和与之联系的程序时，可以使用数据库或接到一个基于请求的回应。而 Java 的可执行内容使用户有机会与超文本页面进行连续的、实时的、复杂的相互作用，这种可执行内容被全部下载到用户的计算机上，可用来运行动画，进行计算，或指引用户利用远程网络节点上更多的信息。

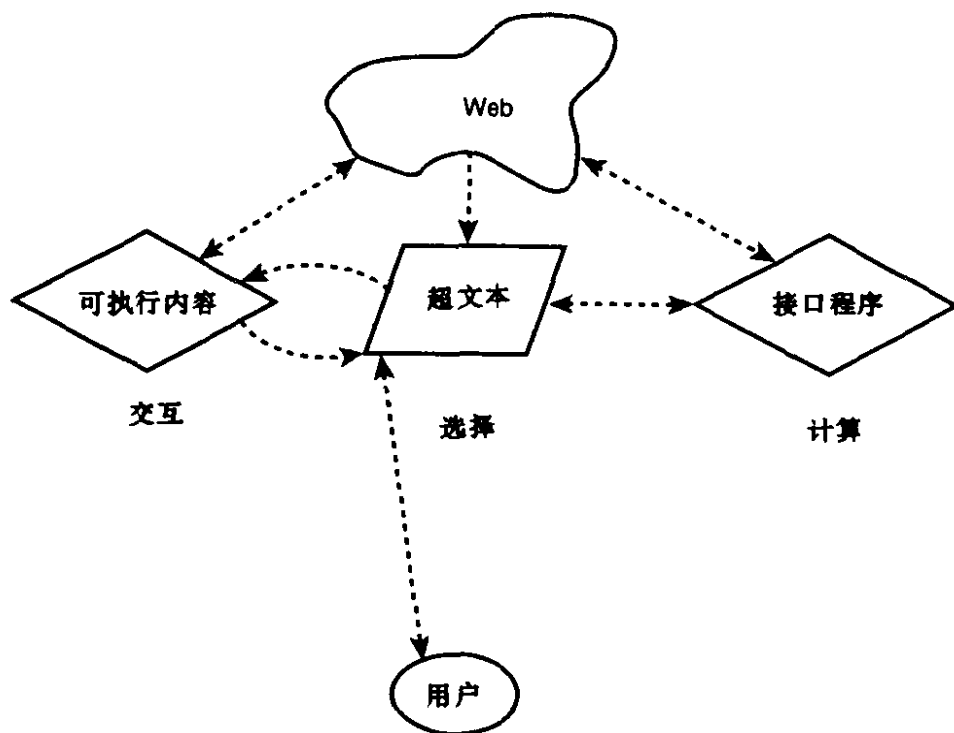


图 1.1 Web 的软件支持选择、显示、计算和交互性

Java 使 Web 页面比仅用接口程序更加活泼和具有更高的交互程度。

Java 的喻义 超文本的一个喻义是它提供了信息的可视性静态页面（包括文本、图形、声音和图像），超文本页面也可以有“深度”，它包括与其它文件和资源的超链接。



Java 把这种静态页面的喻义转换成动态的类型。Java 页面上的信息不仅是静态可视的或局限于某些与用户之间预定义的一组作用方式。使用 Java 程序的用户可以参与更广泛类型的交互行为，只受限于 Java 编程者的想象力和技巧。Java 把超文本页面变成了一个舞台，由演员的操纵来显示和发生事件。而且，用户不再仅是听众，他作为支持 Java 的浏览器的使用者是这一舞台上戏剧的积极的参与者。通过使用 Java，用户可以参与改变所要发生的一切并做出反应，这使用户对发布到 Web 上的信息有更积极的操纵能力。

Java 是什么

Java 这个词表示 Sun Microsystems 发展的一种用来创作可在网络上发布的可执行内容的程序设计语言。通常使用时，Java 表示一组用来创作和执行 Java 程序设计语言书写的可执行内容的工具软件。一种重要的工具软件是可以解释 Java 产生的代码的特殊浏览器，这种叫做 HotJava 的浏览器被用来展示 Java 这种程序设计语言的能力。

HotJava 有与绝大多数 Web 浏览器同样的功能，另外又具有解释和演示 Java 的可执行内容的能力，所以尽管 HotJava 可以显示 Web 的任何页面，不识别 Java 的 Web 浏览器却不能显示 Java 的可执行内容，即 HotJava 浏览器能看到的 Web 更多——包括了 Java 写的应用程序。正如下面描写 Java 的起源与未来的一节中讲述的一样，Java 的特性将可望被加入其它浏览器未来的版本中，

希望得到或下载一个 HotJava 浏览器的读者,参看本书第三部分。



Java 的 Web 主页 Java 的开发者, Sun Microsystems, 在 Web 上提供了一个有关 Java 的信息的节点 <http://java.sun.com/>, 这个节点收录了绝大部分最新的有关 Java 和支持 Java 的浏览器的信息, 这个节点列出的连接包括详细的声明、版本发布信息、文档和 Java 演示。

在你下载 HotJava 浏览器的同时, 你将得到一套 Java 语言发展工具。第二章更详细地描述 Java 的技术设计和工具软件。

什么是可执行内容

可执行内容是一个一般性术语, 标志着支持 Java 的 Web 浏览器所能下载的内容与不支持 Java 的 Web 浏览器所能下载的内容的重要区别。简单地说, 使用一个不支持 Java 的 Web 浏览器, 下载的内容由术语 Multipurpose Internet Mail Extensions (MIME) 定义, 包括一大类多媒体文件格式。这些格式被设置成可以在浏览器或帮助性应用 (用来显示图片、声音、图像等多媒体内容) 中显示。结果是用户选择并观看这些内容。

一个支持 Java 的浏览器同样也可以下载并显示 MIME 所定义的内容。但主要区别在于: 支持 Java 的浏览器可以识别一种特别的超文本项 `<CODE>APP</CODE>`, 当下载一个包含 APP 的 Web 页面时, 支持 Java 的浏览器知道一种特殊的 Java 程序 **applet** 与这个页面相关, 浏览器就会下载一个描述 applet 执行的信息文件, 这个信息文件是用所谓字节码写的。支持 Java 的浏览器解释这些字节码并把它当作用户主机上的一个可执行程序而执行它。这种下载和执行是自动发生的, 不需要用户提出请求。

于是, 当用户用支持 Java 的浏览器在 Web 上“冲浪”时, 不仅能发现 Web 上前 Java 年代提供的超文本内容, 而且可以发现

动态的、可执行的和分布性的内容。更进一步，这种可执行内容包含处理和掌握除信息协议之外新形式的媒体的指导。

Java 如何改变了 Web

Java 给 Web 带来深刻的改变，因为它用以前的 Web 软件系统不可能实现的交互性丰富了 Web。

图 1.2 描述在 Java 的交互性与超文本的选择性与接口程序之间的技术差别。此图表明，接口程序允许计算和响应，但不是实时的。

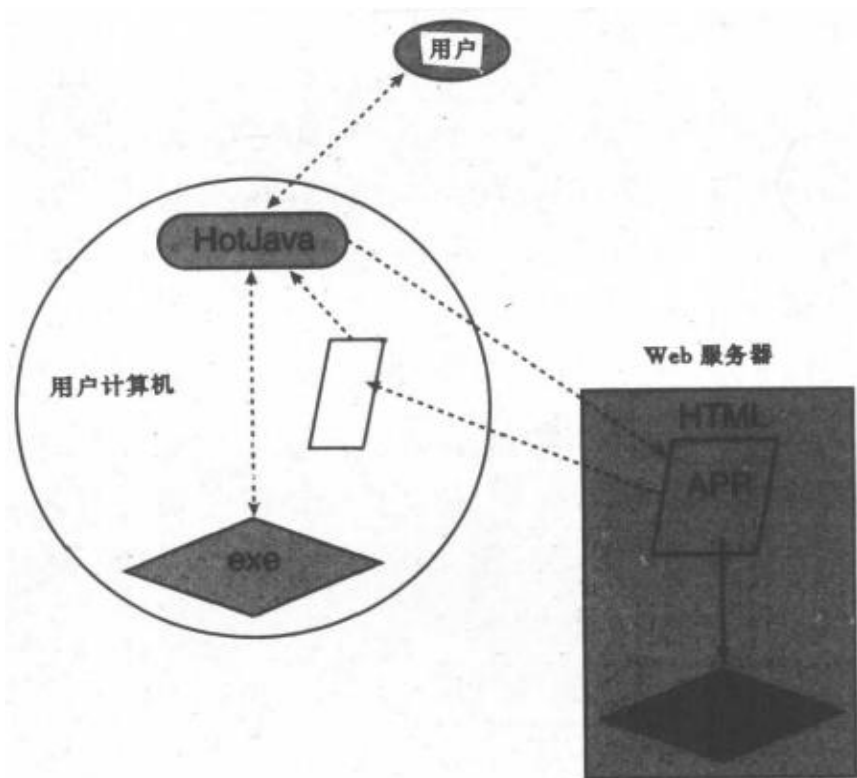


图 1.2 Java 的交互性以下载到用户计算机上的可执行内容为基础

Java 的起源和方向

根据 Michael O'Connell 在 1995 年 7 月 7 日在 Sun World Online (<http://www.sun.com/sunworldonline/swol-07-1995/swol-07-java.html>) 上发表的介绍 Java 起源的总结性文章声称，Java 在加州的 Sun Microsystems 的开发差不多是与 World Wide

Web 1991 年在瑞士的开发同时起步的。用 Java 的创始人 James Gosling 的话来说，这个早期开发小组的目标是开发简单的和无差错的消费者电子产品。要求是产生与平台无关的代码从而允许软件在任何中央处理单元（CPU）上运行。

作为设计一种与平台无关的计算机语言的起点，开发小组首先把目光投向 C++。但是，小组不能使 C++ 完成成为产生一个支持异种机器通信的分布式网络系统所应该完成的所有事情。小组放弃了 C++，发展了一种称为 Oak 的语言（后改名为 Java）。1992 年秋，小组制订了一个规划名为 Star7 (*7)，这是一种个人的、手提式遥控器。

开发小组成立了 FirstPerson 公司，但在一次 Time-Warner 的电视机投标中失败。到 1994 年中，Web 的大众化的迅速发展吸引了小组的注意力。他们决定做一个利用 Java 技术的出色的浏览器。他们做了一个 Web 浏览器，把他们的 CPU 无关的和实时的编程系统带进了 Web。

这个叫做 WebRunner 的浏览器用 Java 写成，完成于 1994 年秋天。Sun Microsystems 的董事们被打动了，看到一个新的浏览器可能带来的技术和商业化可能：工具、服务器、开发环境。

1995 年 5 月 23 日，Sun Microsystems 在旧金山的 Sun World'95 上正式发布了 Java 和 HotJava。

Java 当前状态和时间表

有关 Java 版本发布的最新信息，请看 Sun Microsystems 的 Java 节点 <http://java.sun.com/>。

Java/HotJava 时间表：

1. 1995 年 4 月 11 日，加州，San Jose：Netscape 主席 Jim Clark 和 Sun 首席执行官 Scott McNealy 重申了他们的合作关系并许诺在 Sun World'95 上发布更新的消息。
2. 1995 年 5 月 23 日，加州，旧金山：在 Sun World'95，Netscape 宣布它将获得在 Netscape Navigator 浏览器中

使用 Sun 的 Java 程序设计语言的许可。

3. 1995 年夏: Java 和 HotJava 的 alpha 测试阶段。Sun Solaris 2.3, 2.4 和 2.5 Sparc 上的 alpha 版发布。Microsoft Windows NT 上的 alpha 版发布。正在往 Windows 95 与 Mac OS 7.5 上移植。往其它操作系统, 如 Windows 3.1, Amiga, NeXT, Silicon Graphics, Linux 上的移植在有第三方参与情况下正进行着。
4. 1995 年夏: Sun Microsystems 发起了一个 applet 编程竞赛, 鼓励优秀的 applet 开发, 胜利者于 1995 年 9 月 15 日在 HotJava 主页 (<http://java.sun.com/>) 上宣布。
5. 1995 年秋/冬: Netscape 浏览器引入 Java 技术 (参看 <http://home.netscape.com/>)。

Java 未来的可能发展

Java 技术可以应用于很多系统中——如手持式仪器、电话、VCR 等家用电器。三菱电子已经在他们生产的设备中利用 Java 技术。

Netscape 和 Sun Microsystems 的合作将在 1995 年晚些时候使 Java 技术进入 Netscape 浏览器。鉴于 Netscape 广泛的使用基础, Java 在应用程序中的使用将大大增加。

Sun Microsystems 也计划授权其它 Web 浏览器制作商、联机服务提供者、软件开发商使用 Java 技术。第三方软件市场和 Java 工具库将建立, Java 最上层的软件层将使开发者用更高明的工具产生应用程序。

Java 的能力描述

Java 是新的程序设计语言, Sun Microsystems 以外的编程者只是刚刚发现它的功能。但是, 你现在就可以观看 Java 的行为。本章余下部分将通过例子向读者演示 Java 基本的功能, 强调 Java

是支持可执行内容发布的唯一途径。Java 的所有功能的更详细的应用程序将在本书第二部分中描述。

动 画

Java 的应用程序把动画引入了 Web 页面。图 1.3 显示 Duke——Java 的吉祥物的一幅静止图像，它正在 HotJava 浏览器显示的一个 Web 页面上翻跟头。当用户用 HotJava 浏览器下载这个页面后，由一系列画面组成的翻跟头的 Duke 就在页面上往复运动。

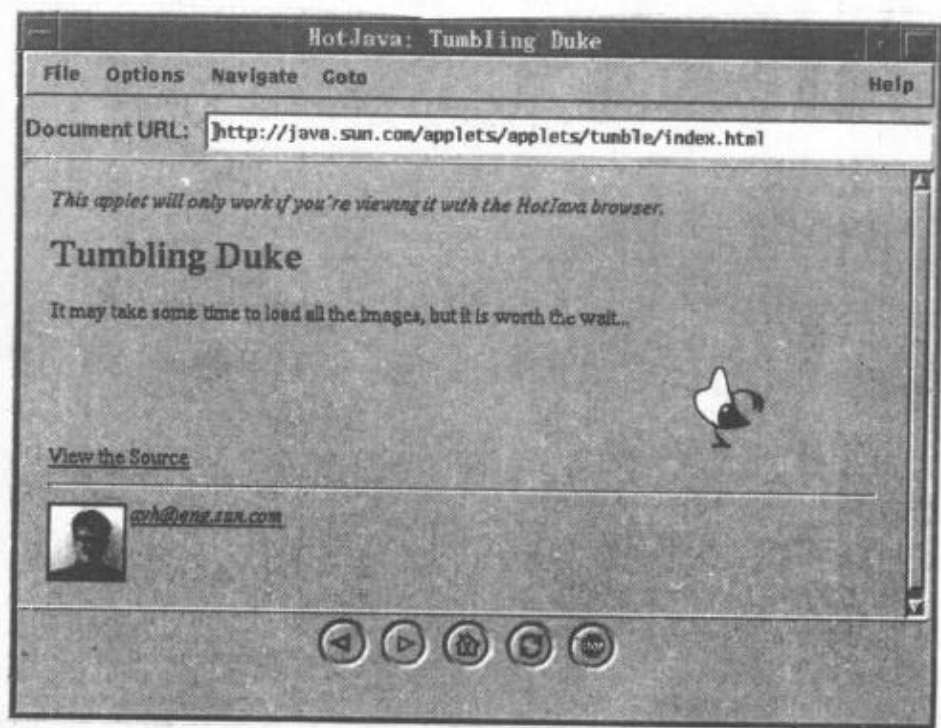


图 1.3 Java 的吉祥物 Duke 在翻跟头

(承蒙 Sun Microsystems 的 Arthur van Hoff 好意提供)

动画并不局限于卡通片。广告标识或文字都可以在页面上运动着或闪烁着穿过屏幕。Java 的动画也不限于事先产生的装饰性图形，也可以是基于计算产生的图形。图 1.4 显示了用一个数学方程式产生的波形。

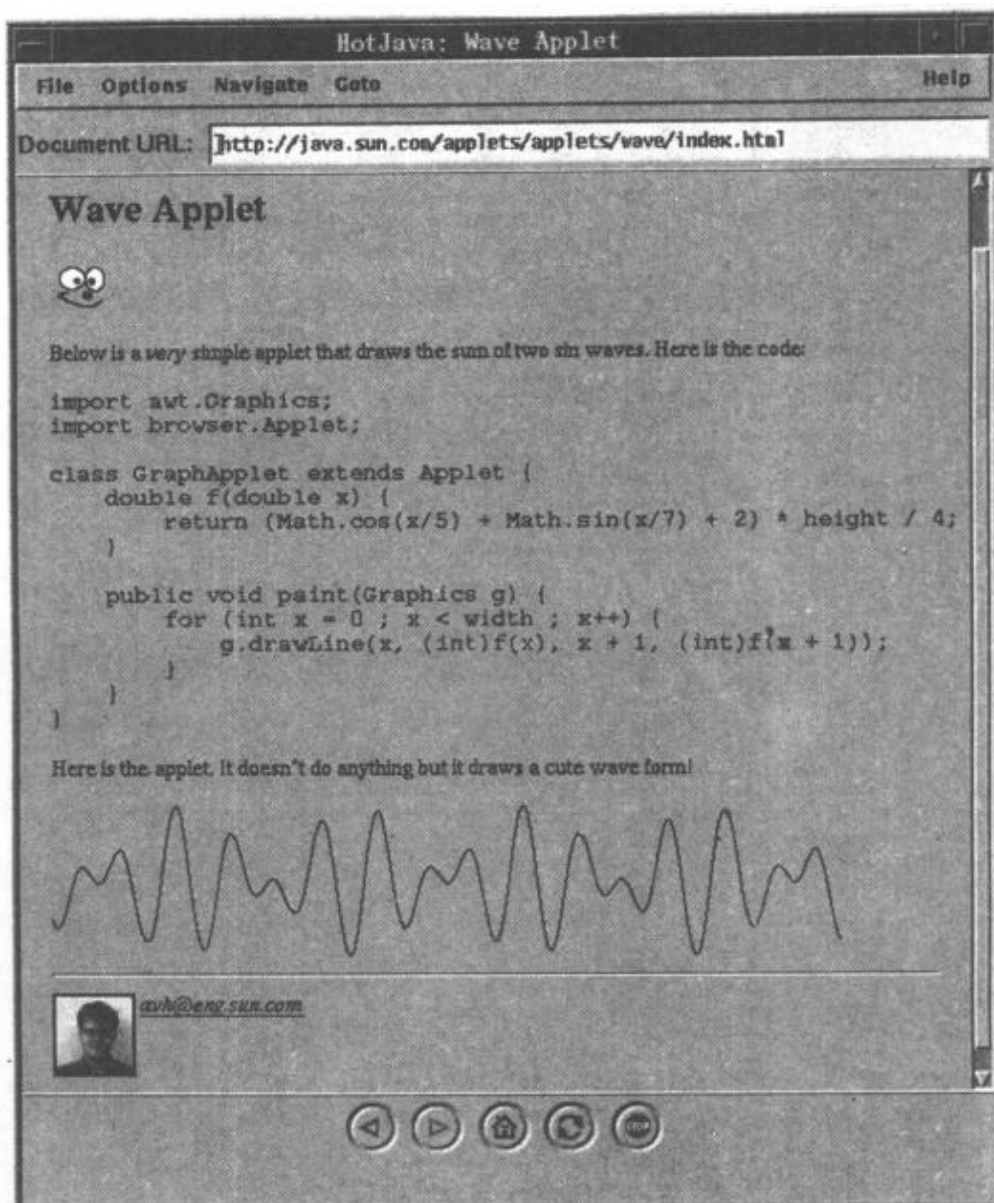


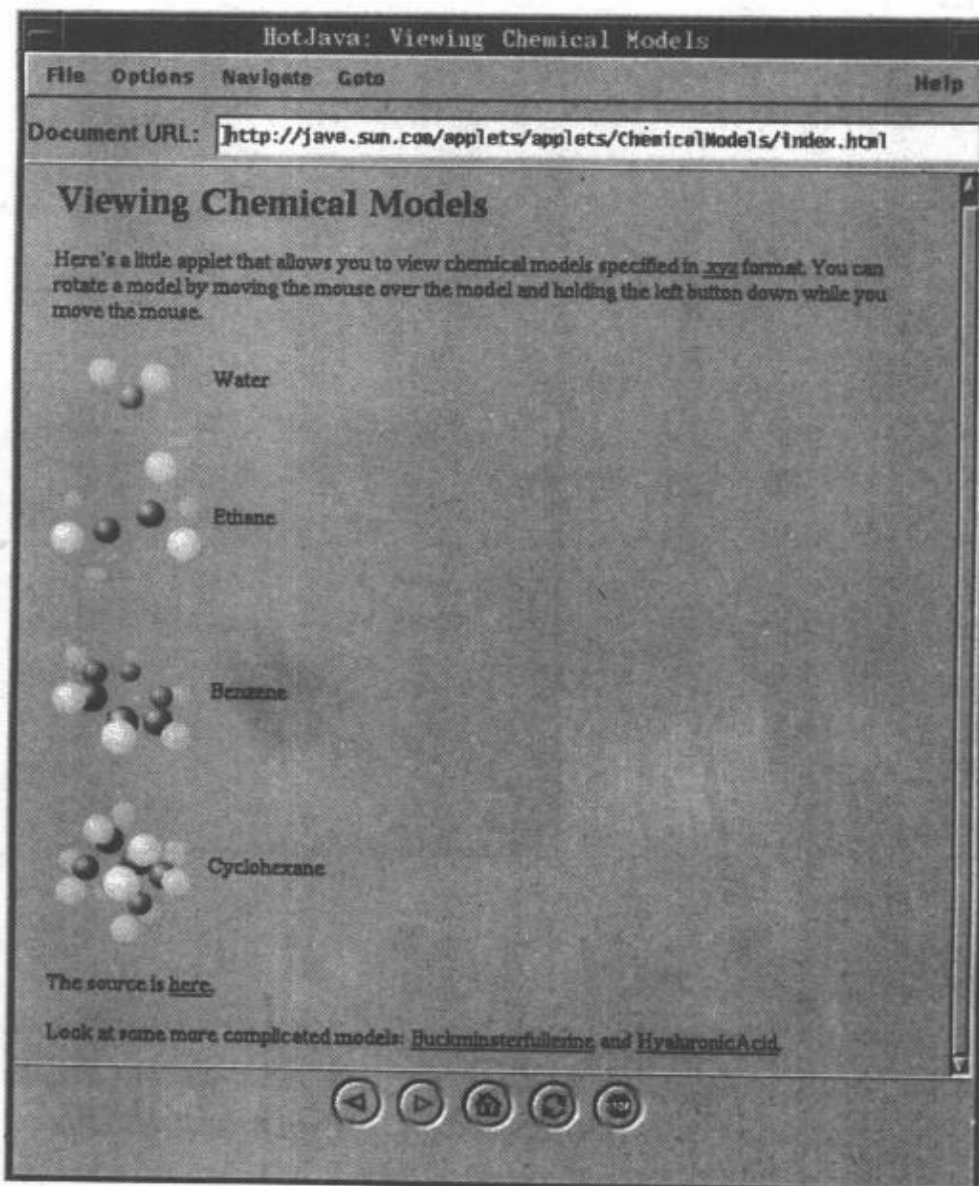
图 1.4 一个数学公式产生的图形

(承蒙 Sun Microsystems 的 Arthur van Hoff 好意提供)

交互作用

尽管显示的动画可以是事先画好或产生的静止图像，或依事先设定的算法运动的画面（如图 1.3 翻跟头的 Duke），动画同样可以做成交互性的，用户 can 对其显示进行操作。

图 1.5 显示了生成的三维化学分子模型。用户可以用鼠标使它以很多角度自旋。与图 1.4 中的图形 applet 的源码不同，化学



误的显示成红色。图 1.6 中，横行 40 被加亮，窗口的顶端显示提示，40 行中的字母“K”是不正确的，因为 26 列被显示成红色（在黑白图像中区分不出来）。这种反馈给用户以纸面填词游戏所没有的优势。这个应用程序的结果是对 Java 如何产生一个与用户相互作用并有即时反馈的界面的极好的演示。

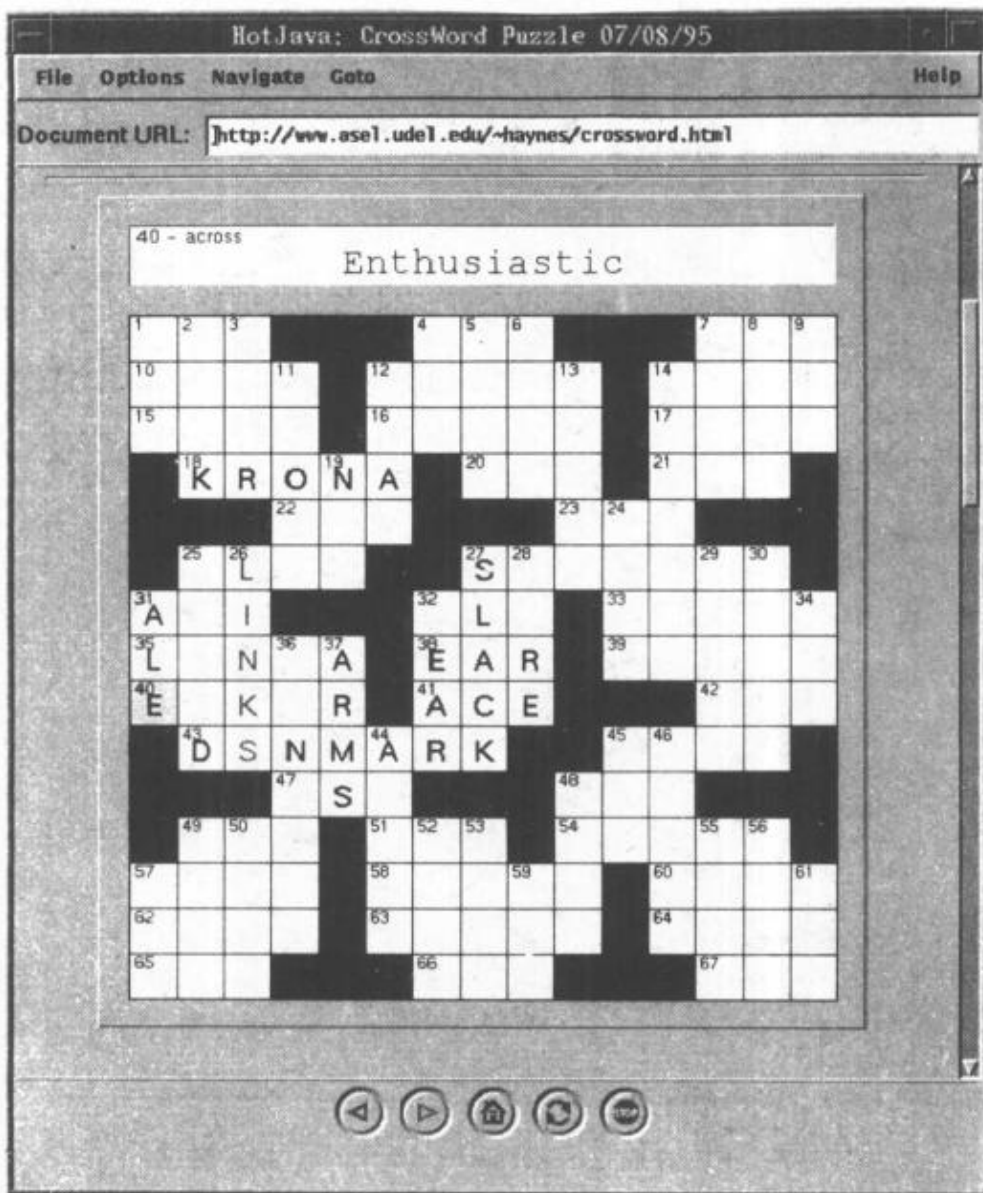


图 1.6 交互式填词游戏
(承蒙 Carl W. Haynes III 好意提供)

交互性的另一个变化是实时交互性。图 1.7 显示了一个交互性应用程序，其中用户可以操纵移动的图形。这是积木游戏，使用者必须尽力使可以旋转的图形填满每一行。利用事先定义的键，

用户与界面交互作用以操纵下落的方块。积木游戏的实现表明用 Java 技术编制类似拱廊的游戏是可能的。

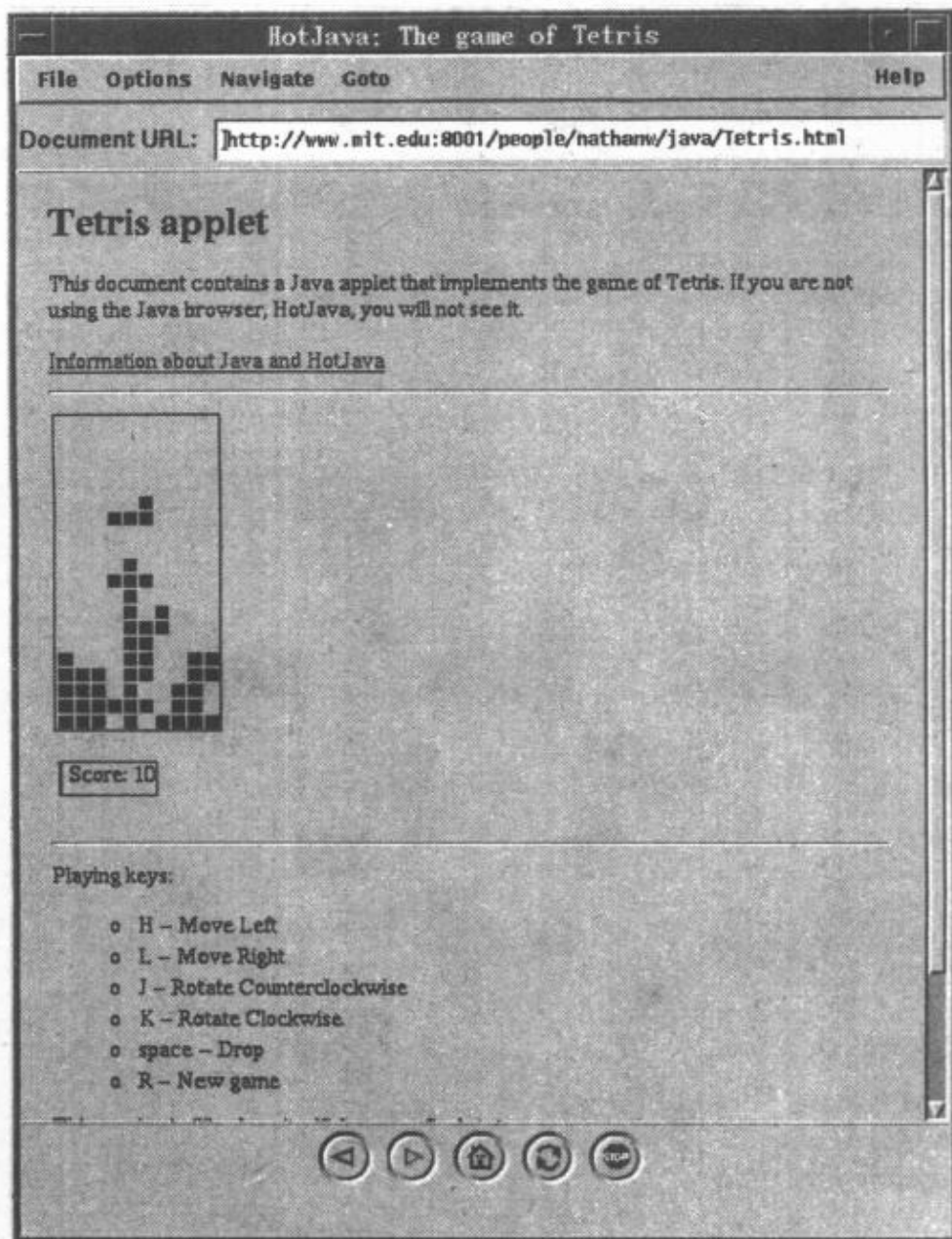


图 1.7 积木游戏 (承蒙 Nathan Williams 好意提供)

交互性和计算

填词游戏和积木游戏表明了交互性和动画是怎样同时实现的。两个 applet 都依用户的输入而改变它们的动态输出，所以两个 applet 实际上都在做计算。不过，图 1.8 显示了一个把这种计

算能力更具体地表示出来的例子，这是一个简单的表格。

这个表格与填字游戏的工作方式很相似，但强调了用户除了解一个字谜外可以利用这种计算能力创造一个工作环境。此表格显示允许用户改变 24 个单元 (A1 到 D6) 中任何一个的内容，包括替换它的标号、数值和表达式。这很像一个真正的表格，比之一个像填字游戏的固定的游戏，它更像一个用户可以工作的环境。实质性的不同是深刻的：用户利用 Java 可以得到一个完全无约束的相互作用的环境，而不仅仅是一些相互作用的选择好的固定集合；用户打开一个 Web 页面就进入了 Web 的舞台。

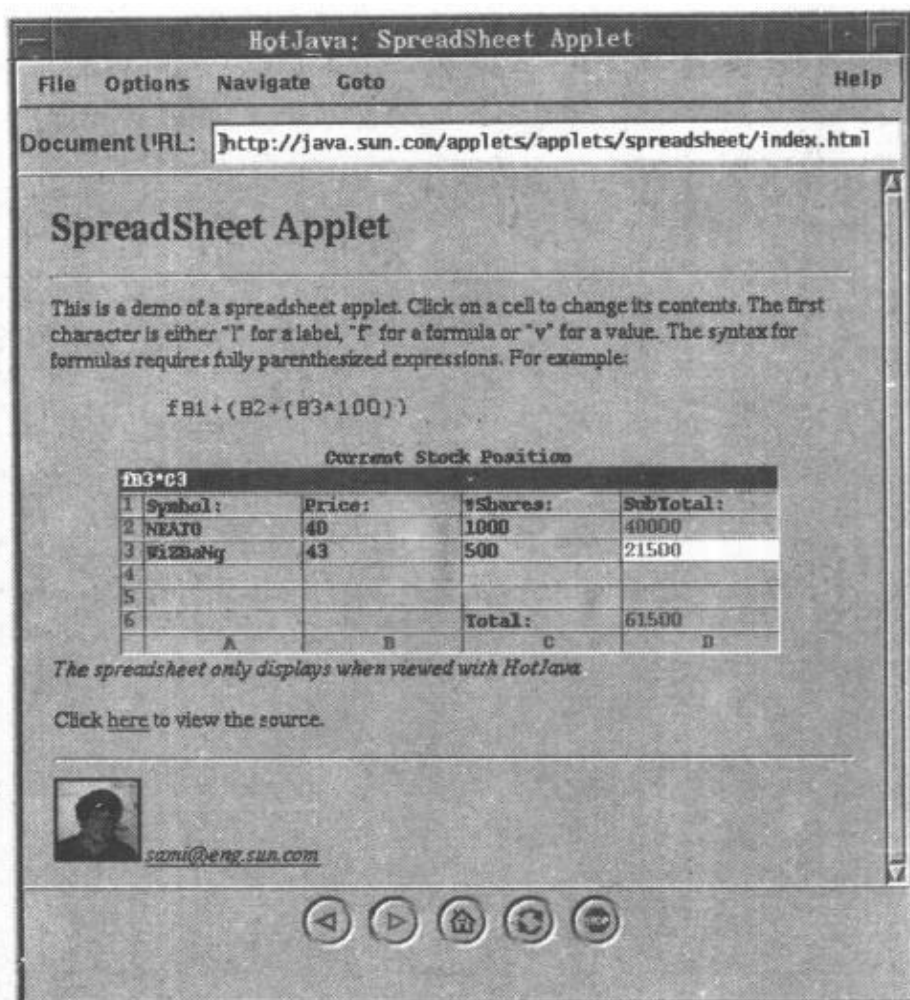


图 1.8 一个简单的表格

(承蒙 Sun Microsystems 的 Sami Shaio 好意提供)

分布式应用程序

图 1.9 显示了在 HotJava 浏览器上操作的示波器，它允许用户操作其设置。这是一个结果有点像化学分子模型的应用程序，但有对用户连续的、独立的反馈。一旦设置好，示波器就将显示电子信号的波形，这个应用程序显示了动画、相互作用、计算和基于用户设置的连续显示。结果就是用户实质上远程下载了一个示波器。

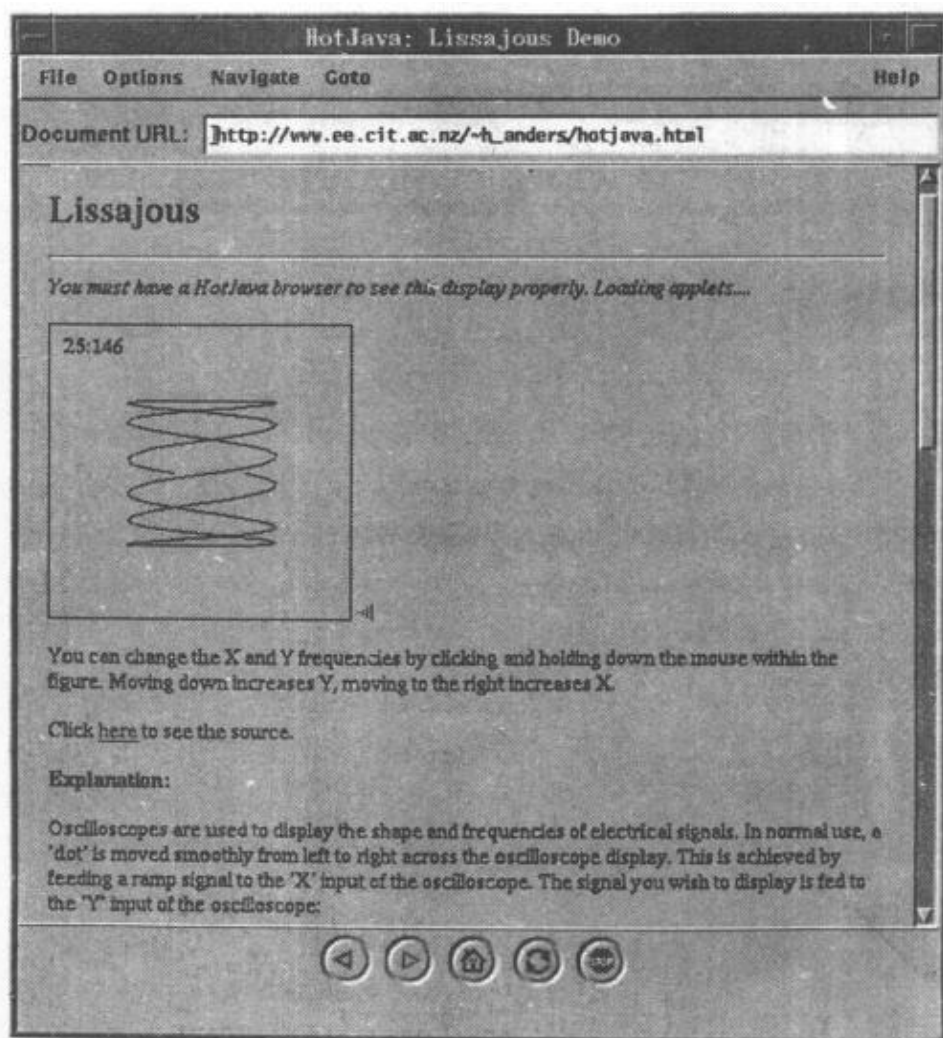


图 1.9 示波器（承蒙 Hugh Anderson 好意提供）

用户不仅可以下载一个示波器，而且可以下载一个几乎可以做任何事情的“工具包”。图 1.10 显示了一个可以展示 Java 的能力的化妆工具包。这个工具包允许用户选择颜色和画笔类型，在

工具包提供的一张脸上绘画。这个非常简单的绘画应用程序向用户显示了用 Java 页面可以建立些什么。



图 1.10 化妆工具包

(承蒙 George Coates Performance Works 好意提供)

通 信

前面的例子演示了 Java 的信息、动画和计算应用程序。另一个应用领域是人与人的通信。Carl W. Haynes 已实现了一个简单的以 Java 为基础的交谈系统,如图 1.11 所示。这个系统允许用户登录到相应的 Web 页面上,书写可为其他正阅读这个页面的人看到的句子。用户打印的句子可以被其他所有用支持 Java 的浏览器阅读此页面的人看见。这个一群人交谈的系统的结果类似于其他用 Telnet, Internet Relay Chat 或 MU* 系统的实现结果。

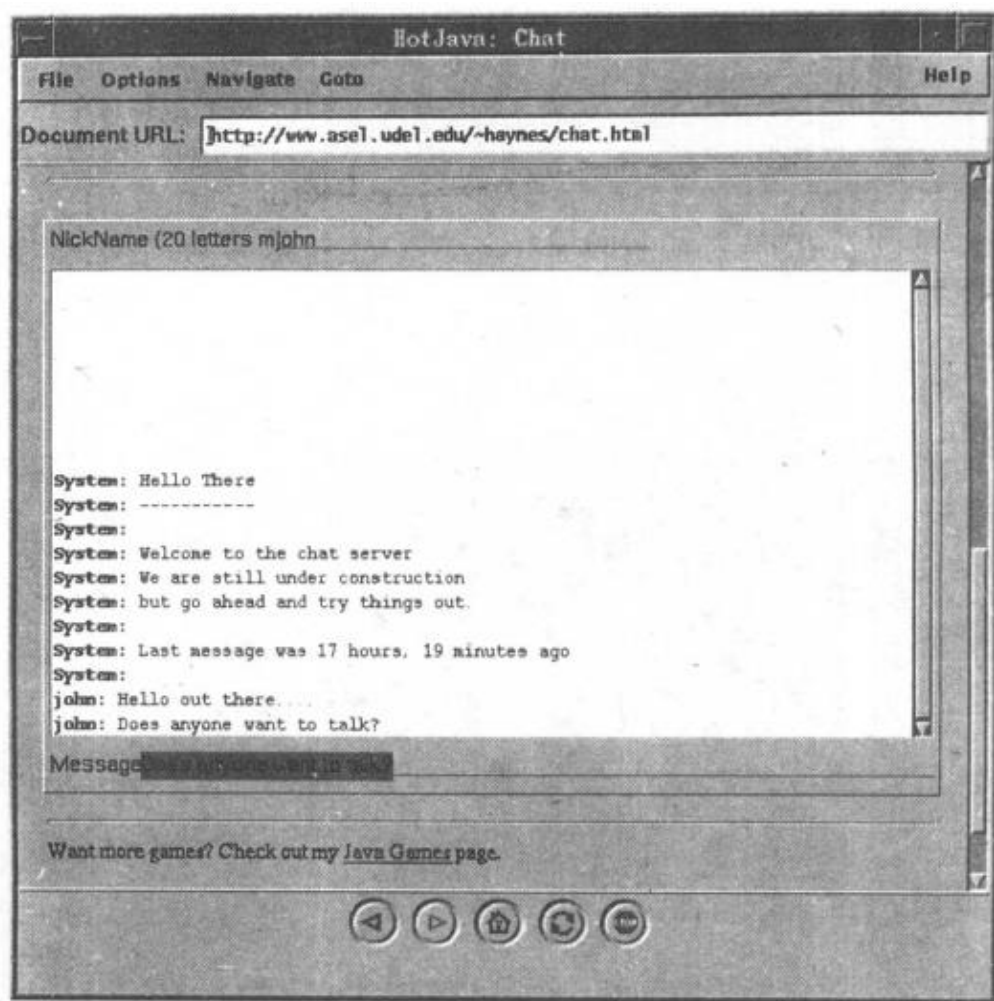


图 1.11 一个基于 Java 的交谈系统

(承蒙 Carl W. Haynes III 好意提供)

当然,通信每时每刻都在所有 Web 页面通过文本或其它媒体实现着。但支持 Java 的浏览器也可以显示多媒体内容。图 1.12 描述了一个简单的报时钟——用户可以用鼠标键单击显示的钟点得到当前时间的有声报时。

Java 也可以用来支持新方式的大量通信。NandO Times 是一种基于 Web 的新闻服务,在 Web 的新闻投递上颇具新意。这个新闻机构使用 Java 在它的封面上提供滚动标题带。在 NandO 标题下的文本不停滚动以显示此时头条国际、国内、运动、政治新闻。目录标题下的图片也在改变,给出不需要读者去选择就可观看的新信息的“滑动显示”。这使 Web 可改变成一些人可以观看并得到新信息的东西。

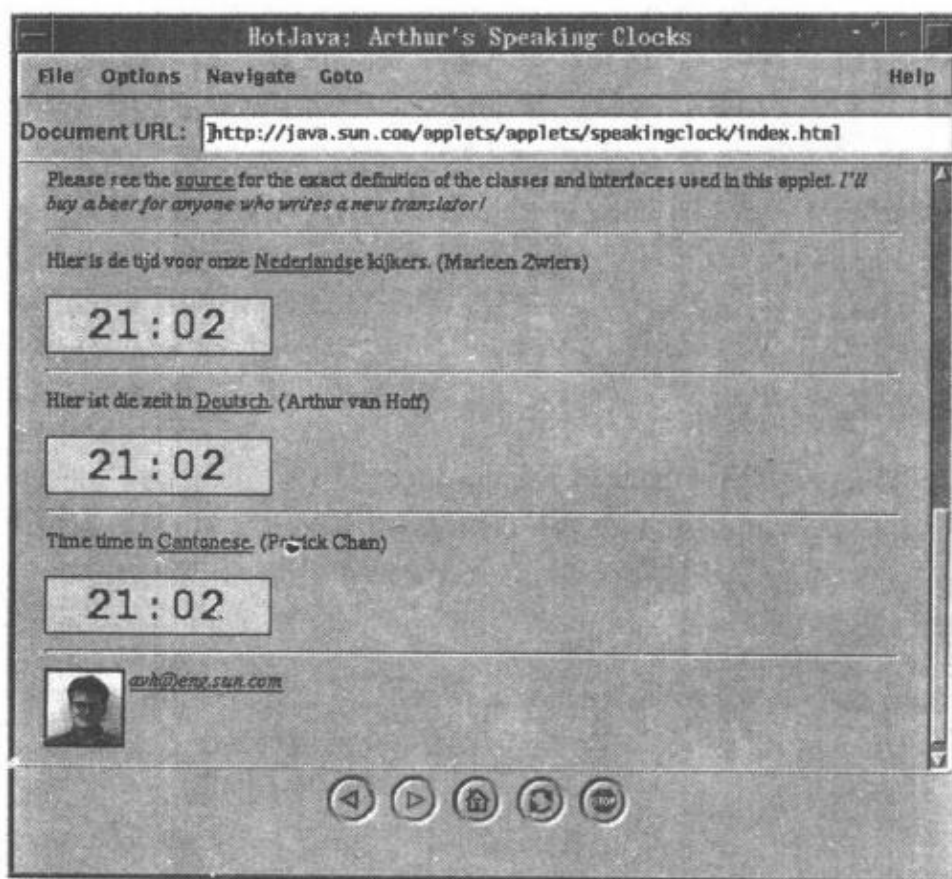


图 1.12 一个报时钟

(承蒙 Sun Microsystems 的 Arthur van Hoff 好意提供)

类似地，图 1.13 显示实时信息传递可以用作特殊行为的监控。这个图显示了 ESPN 的 SportsZone Live Scoreboard 的一个试验，是正在进行的比赛成绩显示。得分改变，显示也改变，体育爱好者可借此跟上现在的比赛和比分。类似 NandO Times 新闻传送，体育传送使 Web 改变成除某种可以与之交互作用的东西以外的另一种可以观看的东西。

应用程序与处理器 (Handler)

除了在这里展示的 applet 外，Java 编程者也可以编制不需要运行支持 Java 的浏览器的应用程序，或独立的程序（HotJava 浏览器本身就是这么一种用 Java 写的应用程序）。应用程序可以想象成与其它网络或本地的资源相互作用的新的浏览器或界面。

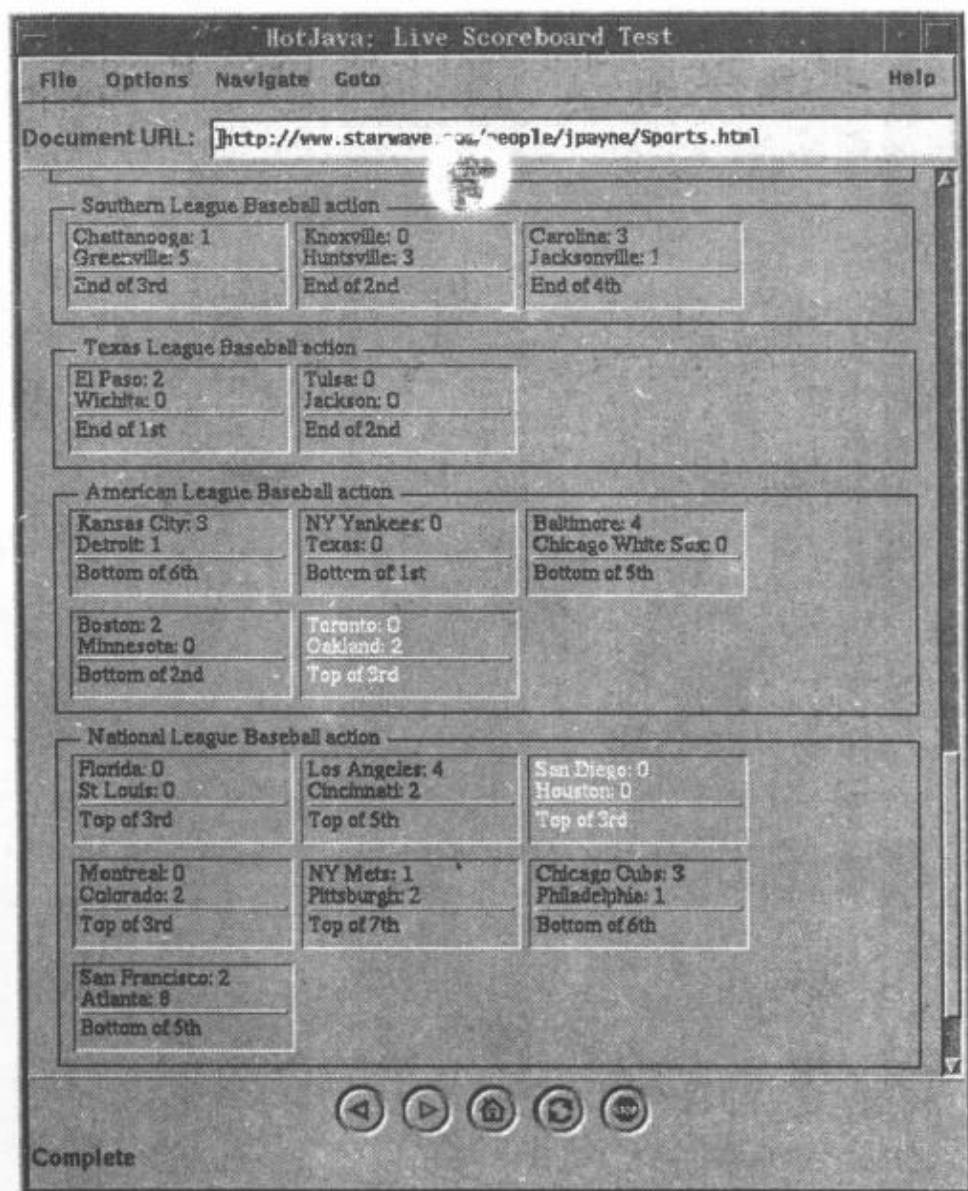


图 1.13 ESPN SportsZone Live Scoreboard

(承蒙 Starwave Corporation 的 Jonathan Payne 提供)

另一种可用 Java 实现的软件程序是处理器。协议处理器使 Java 编程者能指定一个 Java 浏览器怎样去解释一个特定类型的协议。HotJava 浏览器知道怎样基于其分布式编码去解释 Internet 协议如 HTTP, FTP, Gopher 等。但如果新的协议发明了, 一个 Java 编程者可以编制一个协议处理器去处理它。

另一类型的处理器是内容处理器。这种处理器可翻译基于 MIME 的文件类型的特定性质。这种内容处理器会指定 HotJava 浏览器如何去处理一种特定类型的文件。通过在内容处理器中加

进相应项，所有支持 Java 的浏览器都可以阅读这一特定格式。

Java 可能实现的处理器和应用程序有能力极大地扩展 Web 上可浏览的范围。信息开发者很快就会留意到用户是否拥有适当的软件去观看特定类型的文件或处理新的协议。就像 Java 把可执行内容做成 applet 分发一样，协议和内容处理器也可以被分发到发出请求的支持 Java 的浏览器上。

Java 可能实现什么

前面的例子只描述了 Java 的部分能力。这些例子中有些只是像演示 Java 的能力的“玩具”。Java 将促进什么类型的通信？Nando Times 展示了一种创新的提供信息的应用程序，使用户可以安坐观看而不必选择超文本连接。

Java 使 Web 上的交互性和相互作用的可操作性大大加深了一步。在一个 Java 风格的 Web 上，早期的书写页面和连接它们的 Web 发展技术仍是必需的。但是，Java 使可发展的内容有更大的可能性，用户可以用一个支持 Java 的浏览器与 Web 页面相互作用并改变它和数据库的状态。所以，Java 在以下几个方面深刻地改变了 Web 的组织。

- Java 产生了在 Web 路径上停止的地点。在一个超文本页面上好的 Java 应用程序可使用户长时间地停留。页面 Java 不仅可以提供阅读的文本、可供欣赏的声音、图片和图像，而且可以提供一种不必用超连接的方法走到 Web 的其他地方就可以与其他人通信和相互作用的方法。如果说 Web 的超文本像是路，那么 Java 页面就像路上可停留的镇、村、城市，用户可以做些除了观察和“冲浪”以外别的事情。
- Java 增加了 Web 上的动态和竞争程度。就像新的浏览器技术推动 Web 开发人员去创作开发这些具有新特征的更多的应用程序和页面一样，Java 技术同样发起新一轮的 Web 内容发展。

- Java 丰富了 Web 上的交互性。Java 的交互性比接口程序所能实现的更丰富、更迅速、更明显。接口程序在 Web 应用中还将还会有一席之地，因为页面设计和多媒体表现仍将有一席之地。但是，Java 交互性给 Web 上带来了新的机遇。利用 Java 给用户带来迅速的、连续的、进行性的反馈，Web 上的业务将会更顺手。
- Java 使 Web 成为一个软件分发系统。Java 作为一种传递可执行内容的语言的精巧设计使程序设计者可以设计任何种类的软件并用 HotJava 浏览器分发给用户。Java 编程者不必把目光集中于界面，而把目光投向所要求的相互作用和让图形的内置特征能关照其余的实现。结果是，像绘画和表格之类简单应用程序可以很快写成并大范围发布。

Java 改变 Web 的真正能力还处在自己的开始阶段。新的潜在的在商业上、信息传送上和用户相互作用上的应用还有待未来 Java 开发者的想象力和技巧。

小 结

Java 是一种被设计成在网络上传送可执行内容的程序设计语言。用户或程序员应当知道 Java 可能实现什么类型的相互作用和它真正的能力可表现在哪儿：使 Web 活起来，以动画和交互性应用程序丰富信息的显示方式。

- Java 丰富了 Web 上可能的交互性。Java 不仅支持信息内容，而且支持交互性内容，这种内容是可以被下载并运行在任何安装了 Java 解释环境的计算机主机上的软件形式。
- Java 是从平台无关性可执行代码的思想发展出来的。Sun Microsystems 的研究者已把 Java 发展成一个可应用于 Web 上的强有力的程序设计和信息传送系统。
- Java 使动画、交互作用、计算、分布式应用和新形式的通信成为可能。通过协议和内容处理器，Java 有能力使新的

协议和新的格式在 Web 使用。

- Java 把 Web 转换成一个软件传送系统,用户不仅有地方可去,而且有事情可做。Java 有可能使 Web 用户浮光掠影的行为方式变成在新的交互性环境游乐和学习的行为方式。

2 Java 的设计是灵活和动态的

Java 程序设计语言在技术上是唯一的。它拥有一套与它在网络上传送可执行内容的角色相联系的特征，也为编程者提供了一套可扩展的性能部件。本章将概述 Java 的技术设计。我们以一个最小的 Java 程序例子“hello world”开始，以期给读者理解 Java 和 HTML 如何联系起来作个准备。读者可以由此去试试本书第二部分中列出的 Java 程序示例。

Java 同样有特别的性能。本章的第二部分，我们将更多地讨论 Java 如何支持可执行的、分布的应用程序的技术细节。

向 Java 问好

理解 Java 的技术细节的第一步是理解 Java 如何与 Web 的超文本相互作用的。本节的例子将表明 HTML 的一个特别的标记是如何把一个叫 applet 的 Java 程序与一个 Web 页面联系起来的。

一个有 Java 的 applet 的页面用支持 Java 的 Web 浏览器去观看就会因动画与交互作用而活起来。

Java 与 Web 的连接

作为一种在 Web 上传送信息的语言，Java 用一个特别的标记 APP 与 Web 的超文本标记语言 (HTML) 相连接，图 2.1 总结了这种连接。Web 服务器上一个由 HTML 写成的文件被用户下载

到用户的浏览器上，此文件通过 APP 标记识别 Java 的 applet 代码。如果用户的 Web 浏览器是支持 Java 的，applet 就在用户的浏览器上运行。

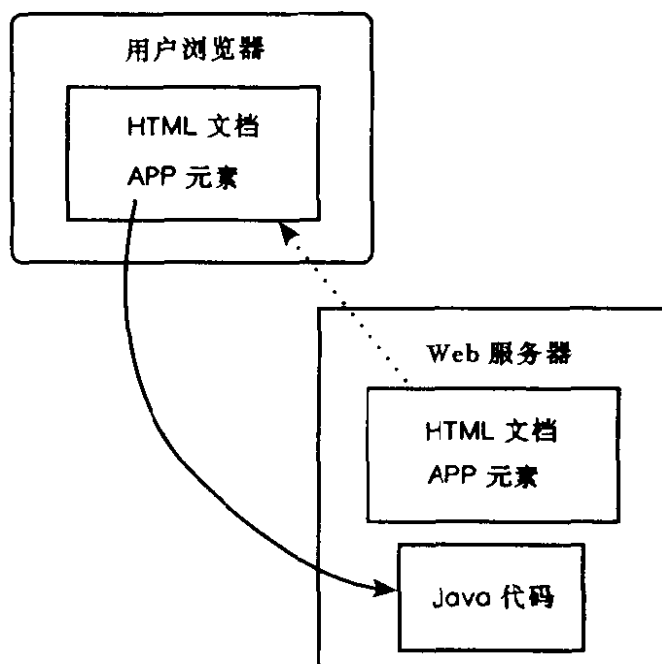


图 2.1 Java 通过 APP 标记与 Web 连接

对 Java 的技术理解要求对 HTML 比较熟悉。HTML 是一种用来书写在 Web 浏览器上显示的文件的标记语言。HTML 并不是一种描写一个超文本页面看起来怎么样的设计语言（尽管 HTML 的很多特征可用来操作页面的外观）。确切地说，HTML 标记了一个文件的结构和文本的意义，从而一个浏览器可以按浏览器的设计和用户对字库的尺寸、风格和其它的特征的设定这样一个方案来显示它。

一个 HTML 文件包含文本和标明文件结构的标记。HTML 文件中的标记用一对括号“<”和“>”括起来。某些标记常常成对出现，即有开始和结束标记。例如，可以在文件的标题两边用<TITLE>和</TITLE>标记表示一个 HTML 文件的标题。其它一些标记并不要求一个对应的结束标记，例如，可用<P>标

记一个段落。

某些标记有属性，用来规定标记的意义。

下面是一个简单的 HTML 文件：

```
<HTML>
<HEAD>
  <TITLE>Example HTML Document </TITLE>
</HEAD>
<BODY>
  <P>
    This is the body of the document.
  <OL>
    <LI>This is the first item in an ordered list.
    <LI>This is the second item.
  </OL>
</BODY>
</HTML>
```

当一个 Web 浏览器解释这些 HTML 标记和文本时，它并不显示出这些“<”和“>”。一个只支持文本的浏览器将显示此 HTML 例子为：

```
Example HTML Document
This the body of the document.
1. This is the first item in an ordered list.
2. This is the second item.
```

附录 C 在参考表里列出了更多的 HTML 标记，显示出 HTML 可实现的更多的特征，注意在写作本书时唯一支持 Java 的浏览器 HotJava 并不能识别全部级别的 HTML 或对 HTML 的扩展。这里的简单 HTML 的例子应当足以向读者介绍 HTML 如何同 Java 连接的初步知识。

一个简单的 Java 程序

HTML 文件中的 APP 标记了一个包含在 Web 页面中的叫做 applet 的 Java 程序，此 applet 的名字被称为它的类名（class

name)。这个名字与运行此 applet 的可执行二进制代码相联系。

下面的 HTML 例子表明怎样把一个 applet 包含进一个 Web 文件。如果读者想试一试，把下列几行写进一个叫 HelloWorld.html 的文件：

```
<HTML>
<HEAD>
  <TITLE>First Hello Applet </TITLE>
</HEAD>
<BODY>
  <P>"This is it!"<APP Class="HelloWorld">
</BODY>
</HTML>
```

当然，读者必须编写名为 HelloWorld 的 Java 源代码。读者可以在本书第四部分中找到用 Java 编程的细节。现在我们只给出一个最小的 Java 的 applet 作为简单的演示：

```
/* This is the first hello. */
import browser.Applet;
import awt.Graphics;
class HelloWorld extends Applet {
    public void init () {
        resize (600, 300);
    }
    public void print (Graphics g) {
        g.drawString ("Hello World!", 50, 100);
    }
}
```

提示

Java 的 alpha 发布版 本书中所有 Java 源代码的例子都是用 Java 的 alpha 发布版编译的。若想把这些源代码例子升级至 Java 的 beta 或更靠后发布版，请参见本书的支持节点 <http://www.december.com/works/java.html>

接下来把 Java 代码放进文件 HelloWorld.java 中。下一步，必须

用 Java 编译器编译 Java 源文件。在操作系统提示符 (\$) 后, 键入:

```
$ javac HelloWorld.java
```

如果没有编译错误, 编译器将产生一个文件 HelloWorld.class, 它包含 HelloWorld applet 的二进制代码。把 HelloWorld.class 文件放入一个子目录 classes。

到这一步, 我们有:

- 一个文件 HelloWorld.html
- 一个文件 HelloWorld.java
- 一个文件 HelloWorld.class 在一个 classes 的子目录中。这是一个 HelloWorld.java 所在目录的子目录。图 2.2 总结了 Java 源代码和编译的关系。

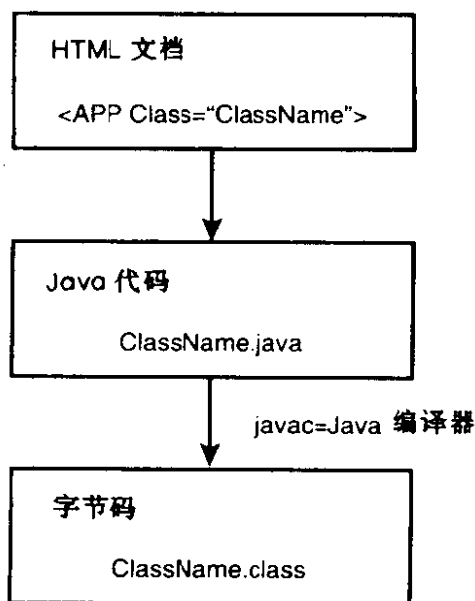


图 2.2 Java 源代码和编译的关系

如果运行一个支持 Java 的浏览器 (参看本书第三部分了解如何得到 HotJava 浏览器), 读者就可以测试这个 applet。用浏览器打开文件 HelloWorld.html, 图 2.3 是本例在 HotJava 浏览器中的显示。

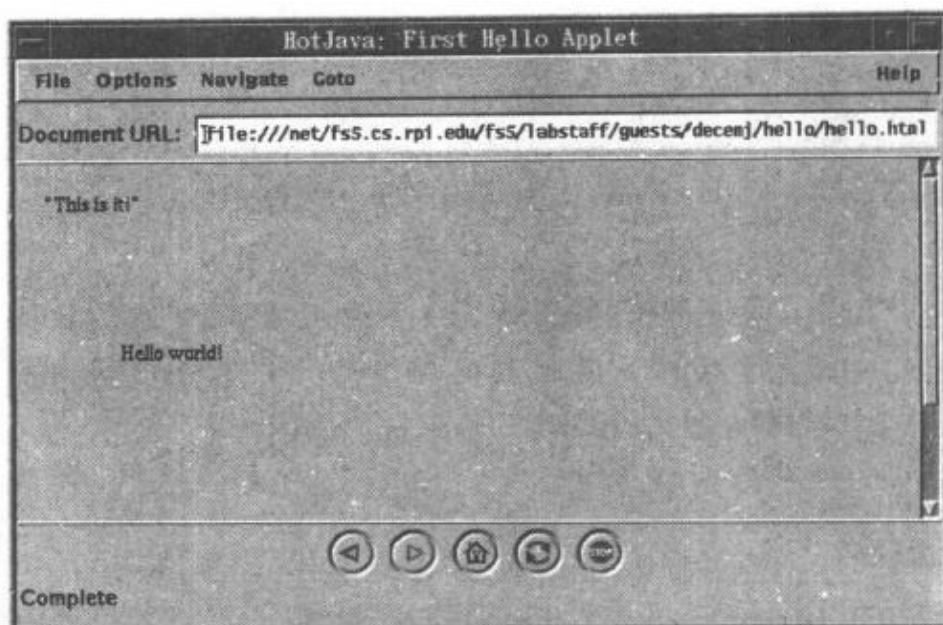


图 2.3 HelloWorld.applet 在 Java 浏览器中的显示

Java 的技术纵览

上一个例子精炼地表明了 Java 的 applet 如何通过 APP 标记与 Web 相连的,但是这只是对 Java 的一个非常粗浅的观察。为帮助读者理解 Java 的设计和能力的,本节将提供一个对这种语言的技术概括。

Java 是一种与支持 Java 的 Web 浏览器联合使用的面向对象的程序设计语言。这种浏览器可以解释由 Java 语言编译器产生的字节码。Java 的技术设计是体系结构无关的。术语体系结构在这里指的是计算机硬件,例如,用户和计算机的体系结构可能是用 Intel 386 芯片的 IBM 个人计算机。程序设计者在编制 Java 程序时不必担心用户计算机的体系结构。相反,HotJava 浏览器却是与用户计算机的体系结构相配的,HotJava 浏览器为用户计算机的特定体系结构解释字节码。这是 Java 的技术设计的关键特征。

Java 周围的网络通信支持环

深入了解 Java 的技术特征前,也许有必要回过头来看一看 Java 如何与 Web 和联机通信相适合的“大图片”。

Java 和支持 Java 的浏览器在 Web 上的全部操作发生在一个较大的联机通信范围内,它要求一大类的网络系统的相互操作。用户使用 HotJava 时不必知道这些系统之间所有的相互操作(读者可以直接到本书的第三部分去寻找如何得到一个 HotJava 的方法。)但是,从 Java 是什么稍稍回过头一点,我们可以看一看它的环境,用一个名词说就是网络 and 应用程序的“支持环”。

Java 的目标是把可执行内容引入 Web。一个支持 Java 的浏览器一经安装就可以提供一个界面支持动画和交互性应用程序。为观看和与这些应用程序相互作用,用户必须有一台安装了支持 Java 的浏览器的计算机,如果用户希望从 Web 的所有地方下载内容,还必须有 Internet 连接。

从 Java 技术的操作要求的最广的环境说起,我们看一看支持 Java 广域地传送信息的系统的必需条件(当然,Java 也可以在没有 Internet 连接的局域网上使用):

1. 电脑空间是人们用来联机或通过计算机进行通信或相互作用的思想模型。电脑空间行为包括提供给用户的各种通信方式、信息和相互作用。电脑空间可被看作由非网络的和网络的区域组成。电脑空间网络区域包括相连的局域、区域和广域的计算机网络活动。

2. Internet 计算机网络是作为利用各种信息传递协议的数据通信的载体。通过网关,电脑空间的许多其它网络可与 Internet 交换数据。因为如此,也因为 Internet 上可获取巨大信息量,它成为电脑空间的网络区域的公共地。

3. Web 是依赖于数据通信的客户机/服务器模式的应用。尽管 Web 可以在不与 Internet 相连的局域网上操作,但它广为人知是由于它对整个 Internet 上可以获取的信息的收集。

4. Web 客户机,即浏览器,是用来解释和显示用各种 Internet 信息协议传输的信息的软件程序。Web 浏览器是用户深入 Web 的界面,与各种帮助应用程序一道工作以演播多媒体内容。

5. HTML 被用来产生 Web 上的超文本文件和标记 Web 文件的语义结构。HTML 由确认文件中的文本的结构和意义的标记和实体组成。

6. APP 标记把 Java 应用程序与 HTML 文件联系起来。这个标记出现在 HTML 文件中确认一个将放在那个文件中的 Java applet。

7. 支持 Java 的浏览器下载超文本和 applet 的可执行字节码。浏览器解释并显示此 applet,并允许用户观看并与 applet 相互作用。

图 2.4 总结了被用来进行广泛信息分发的 Java 支持环。

重复一下,用户不必知道怎样去建立 Java 的“支持环”中整个范围的网络、软件和设备。用户所要做的只是在他的可接入 Internet 的计算机系统上安装一个支持 Java 的浏览器。从用户的观点来说,他的主要焦点是浏览器。图 2.4 画出其它系统是想提醒用户这种通信发生在何种环境中。用户可以利用这种意识设法把自己放入更大的通信环境中去。

Java 作为程序设计语言的特性

尽管用户可能想了解 Java 是怎样适合联机通信的,编程者则想知道 Java 的技术特征。本节的描述将为编程者介绍很多术语。

根据 Sun Microsystems (<http://java.sun.com>) 提供的信息,Java 是简单的、面向对象的、分布式的、可解释的、坚固的、安全的、体系结构无关的、可移植的、高性能的、多线程的和动态的语言。

这个鉴定指出了 Java 的主要技术特点:

简单 Java 的开发者以 C++ 为基础,但去掉了 C++ 中很少用到的或功能不强的面向对象特征。C++ 是一种面

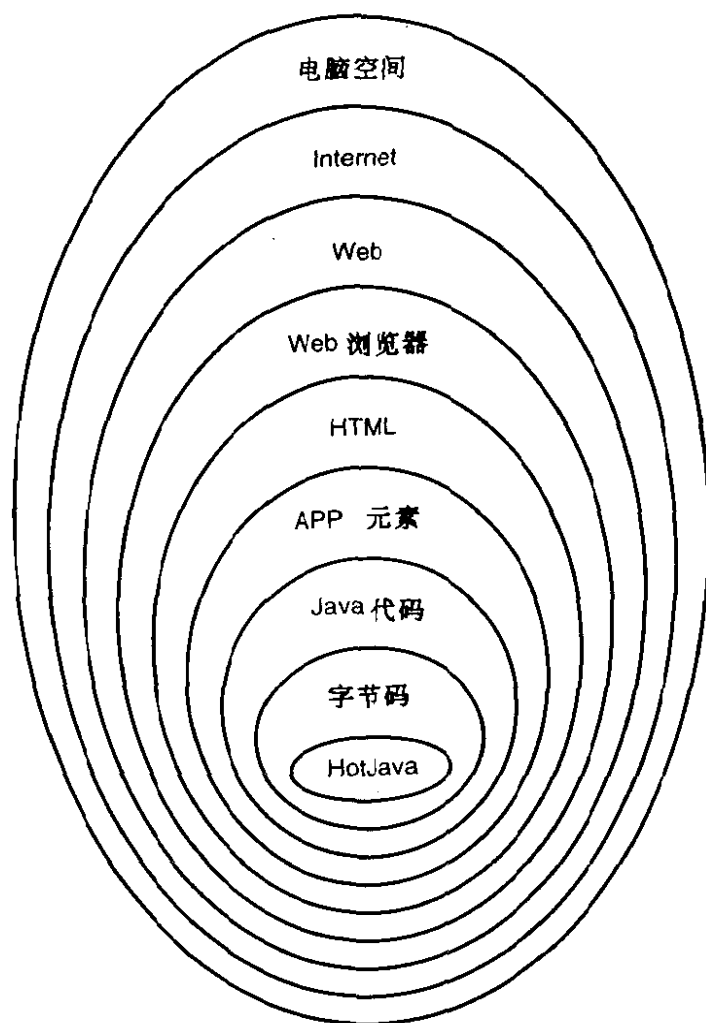


图 2.4 Java 周围的系统支持环

面向对象编程语言，提供了很多强有力的功能。但是，与很多设计成强有力的语言一样，某些功能经常给程序设计者带来麻烦。如经常产生有错误和难懂的代码。因为软件工程的主要费用通常花在代码维护上，而并不是在产生过程上，这种从强有力但难以理解的代码到易懂的代码的转移有利于节省软件开支。详细地说，Java 与 C++（和 C）有以下不同：

1. Java 不支持 struct, union 和 pointer 数据结构。
2. Java 不支持 typedef 或 #define。

3. Java 处理某些特定算符方法不同；它不允许算符重载。
4. Java 不支持多重继承。
5. Java 处理命令行参数与 C 或 C++ 不同。
6. Java 在 `java.lang` 包中有一个 `String` 类。这与 C 或 C++ 中使用的以 `null` 终止的字符串不同。
7. Java 有一个分配和释放内存的自动系统（垃圾收集），所以没必要使用像 C 或 C++ 中那样的分配和释放内存的函数。

面向对象 像 C++ 一样，Java 也支持设计软件的面向对象手段。典型地，面向对象设计允许设计可被重复使用的软件构件。

面向对象程序设计基于把世界用称为对象（object）的软件构件组成。一个对象由数据和可作用于数据上的称为方法（method）的操作组成。这些方法封装或保护了一个对象的数据，因为这些方法是改变数据状态的唯一途径。

面向对象的另一个方面是继承性，对象可以使用其它对象的特征而不必去重复编制那些对象的函数。继承性因此有助于软件复用。继承性的另一个好处体现在软件组织上，使对象按类组织起来意味着同一类中的每个对象都可以继承父亲对象的特征，这使组织文档、理解和从前一代软件受益变得更容易，因为软件的功能当更多的对象被产生时也增多。在一个长的继承链的末端的对象可以是非常专一和有利的。图 2.5 总结了面向对象语言的数据封装、方法和继承性的一般性质。

技术上说，Java 的面向对象特性是 C++ 的特性加上 object C 在动态方法分解方面的扩展。

分布性 Java 与 C++ 或 C 语言不同，它是专门为网络环境的工作而设计的。Java 有一个复杂的类库用来利用 Internet 的 TCP/IP 协议簇进行通信，这些协议包括如 HTTP 和 FTP 等。Java 可以通过 URL 方便地操作资

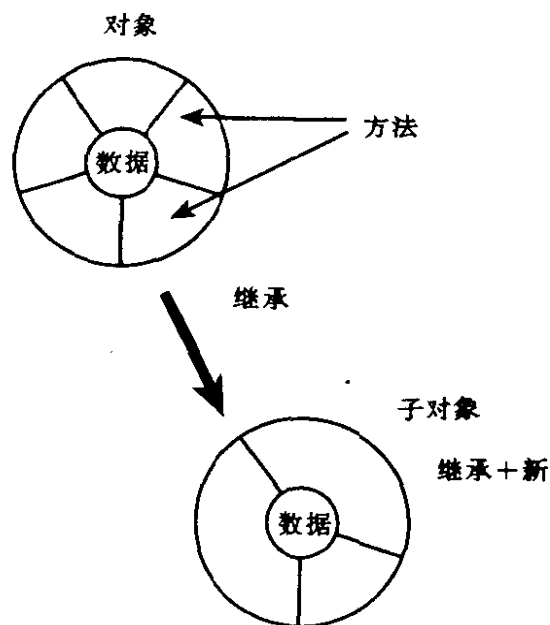


图 2.5 软件中的面向对象

源，就像编程者可以用 C 或 C++ 语言对局部文件系统进行操作一样。

可解释性 当 Java 编译器把 Java class 源文件翻译成字节码后，这种字节码文件可以在任何运行支持 Java 的浏览器的机器上运行，这允许 Java 代码可以写成平台无关的，并消除了客户机的编译和运行循环，因为这种字节码并不是特定为某种机器的，而是具有可解释性的。

坚固性 坚固的软件不容易因其中编程的漏洞而破坏。一个鼓励坚固的软件的程序设计语言经常给正在写源代码的编程者更多的限制。这些限制包含数据类型和指针的使用。C 程序设计语言在检查编译和运行时数据类型的相容性方面由于太宽松而声名狼藉。C++ 被设计得有强壮得多的数据类型；但是，C++ 保留了某些 C 中类型定义的宽松成分。Java 中的类型定义更加严格。例如，编程者不能用 cast 随意地把一个整数转换为指针，另外，Java 不支持指针算术而以数组代之，这个简化消除了像 C 编程者可以用来访问内存任意区域的手法。特别地，

Java 不允许编程者用指针重写内存而破坏其它数据。相反,C 编程者经常可以碰巧(或故意)重写或破坏了数据。

安全性 因为 Java 工作在网络环境,安全问题是开发者应当关心的问题。为 Java 设计的用公开密钥加密技术给数据加密的计划正在实施。Java 对指针的限制将不允许开发者随意访问不允许访问的内存。Java 的这些方面使软件环境更加安全。本章最后一节将粗略地介绍 Java 的安全性的层次。

体系结构无关性 Java 的编译器产生将被发往发出请求的浏览器所在机器并在那里被解释的二进制代码,这些机器上安装了 Java 解释器或支持 Java 的浏览器。

可移植性 体系结构无关性允许很大程度的可移植性,但是,可移植性的另一个方面是硬件如何解释算术操作。在 C 和 C++ 中,源代码运行起来会因不同硬件平台实现的算术操作的差异而稍有不同。在 Java 中,这一点被简化了。Java 中的整数类型 int 是一个有符号的 2 的补码的 32 位整数。实数 float 总是一个在 IEEE 754 标准中定义的 32 位浮点数。

高性能 因为 Java 的字节码是解释性的,其执行有时候不如在一个特殊的硬件平台上直接编译并执行速度快。Java 的编译包括一个选项,可以把二进制代码转换成特定硬件平台的机器码,这可以实现与一般意义上的编译和运行同样的效率。根据 Sun Microsystems 的测试,这种由二进制代码转换为机器码的翻译的性能与从 C 或 C++ 程序直接编译的性能几乎是不可区分的。

多线程 Java 这种语言可以用来产生几件事情同时做的应用程序。Java 基于一种例程系统,这一系统以 C. A. R. Hoare 的 monitor 和条件范例为基础而允许事件的多“线程”。Java 给编程者提供了支持程序的实时的和交互行为的途径。



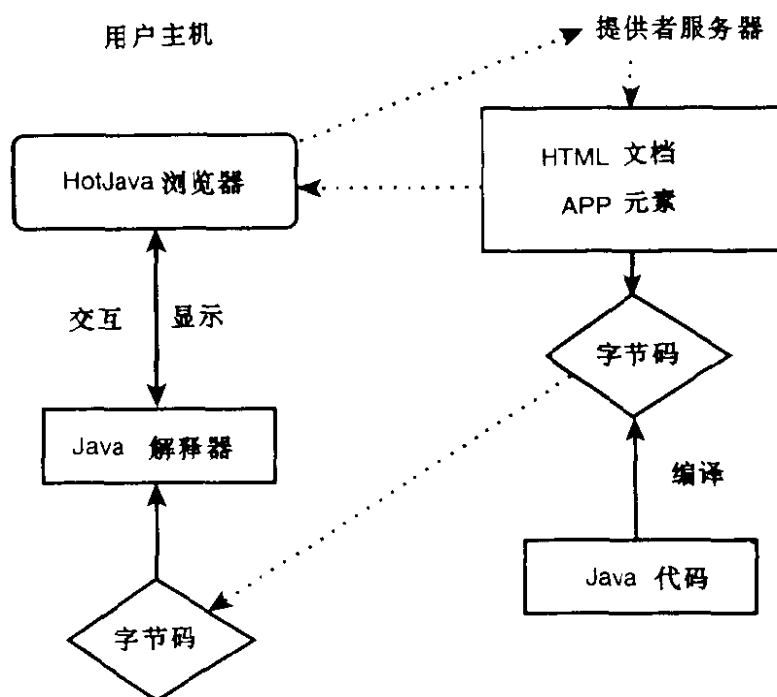


图 2.6 Java 在一个 Web 页面中的操作

world” 程序即是一个 applet。HotJava 浏览器可以用包括在 Java 代码发布包里的 hotjava 命令来运行。

- 应用程序：用 Java 写成的其运行与 HotJava 浏览器无关的独立程序。可用包括在 Java 代码发布包里的 java 命令，即 Java 解释器来实现。
- 协议处理器：被装载进 HotJava 浏览器以解释一种协议的程序。这些协议包括标准协议如 HTTP 等和按一定语法定义的协议。
- 内容处理器：被装载进 HotJava 浏览器以解释 Java 编程者定义的文件类型的程序。Java 编程者提供必要的代码给用户的 HotJava 浏览器以显示/解释文件的特殊格式。
- 内置方法：在一个 Java 类中声明但用 C 语言实现，这些内置方法实质上允许 Java 编程者从 Java 来访问 C 代码。

Java 的发布软件

Java 代码发布包包括以下部分：

- HotJava: World Wide Web 浏览器, 用来观看 Java 程序并与之交互作用, 它由 Java 代码发布包中的 hotjava 命令实现。
- Java 解释器: 这是可移植到各种硬件平台上的 Java 解释器。为观察 Java 内容, 用户必须安装并运行这个解释器或一个支持 Java 的浏览器, 这个基本的解释器和基本的类支持约占 40KB (alpha 版)。标准库和线程支持共占 175KB。相对较小的尺寸使 Java 移植性很好。这个解释器由 java 命令运行。
- Java 编译器: 把人们可读的 Java 源代码翻译成机器可读的字节码的软件。Java 编译器由 javac 命令运行。
- C 头文件和源文件产生子: 产生 C 头文件和源文件用以制作方法, 用 javah 或 javah-g 运行。
- Java 反汇编器: 打印与一个类文件有关的信息 (由 Java 编译器产生的文件)。用 javap 命令运行。
- 文档产生子: 可以产生给定源文件的文档页, 用 javadoc 命令运行, 图 2.7 显示了用 javadoc 命令作用于 HelloWorld 类的结果。
- 定型工具: 把 Java 解释器 java 有 -prof 选项时的输出格式化, 用 javaprof 命令运行。
- 文档: Java 发布包里有大量文档, 可用 HotJava 浏览器中内置的功能来阅读它们 (参见第三部分)。

Java 应用编程界面

Java 应用编程界面 (API) 是随 Java 源代码一起发布的一组类的集合, 编程者可以把它用于 Java 应用程序中。联机提供的 API 文档是 Java 编程者的重要参考资料。API 由 Java 语言写的包组成。API 文档包括以下列表:

- 所有的包
- 一个包中所有的类

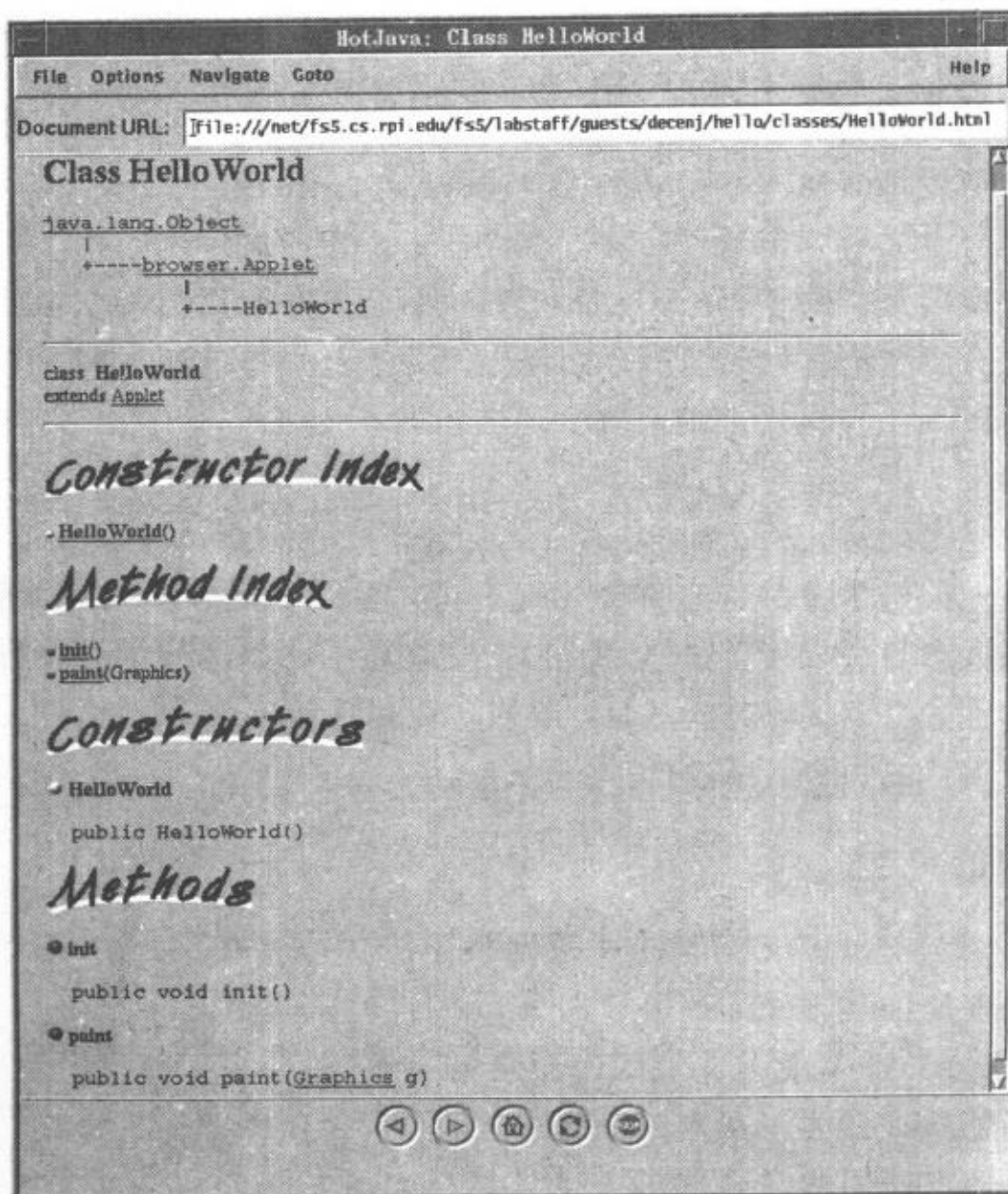


图 2.7 从 HelloWorld 类产生的 Java 文档

● 一个被选择的类

Java 包含有：

- java.lang 包：语言成分，如对象、线程、出错、字符、整数、串。
- java.util 包：实用成分，包括 cache、连接符、栈、矢量。
- java.io 包：输入/输出成分，如输出流、打印流、文本输入流。

Java 编程者可以用这些包和类产生更多功能。本书第四部分

将详细介绍 Java 编程。

Java 虚拟机规范

可从 Sun Microsystems 的 Java 节点 (<http://java.sun.com/>) 得到一个文件叫 Java 虚拟机, 指出 Java 语言怎样设计成在网络上交换可执行内容。这个规范的目的是描述 Java 作为一个非盈利的、开放的语言可以为很多公司使用并作为一个包而销售。

Java 虚拟机规范用抽象名词描述 Java 是如何运作的。它把实现的细节提供给编制 Java 解释器和编译器的编程者。Java 虚拟机规范也详细定义了 Java 代码的交换格式, 称为 Java 交换规范。

虚拟机规范的其它部分定义了可以留给实现者的抽象概念。这些概念与 Java 代码的交换无关。它们包括诸如运行时间数据区的管理、垃圾收集算法、编译器的实现和其他 Java 环境软件, 以及被编译的 Java 代码的优化算法。

Java 的安全性

因为 HotJava 浏览器通过网络下载代码并在用户主机上执行, 安全性与 HotJava 用户及 Java 编程者密切相关。

HotJava 包括几个层次的安全, 具体有:

- Java 语言本身对内存访问有非常严格的限制, 这与 C 语言中使用的内存模型不同。这些限制包括指针算术和非法 cast 算符剔除。
- 在 Java 解释器中的一个字节码验证例程将检验字节码中是否违反任何语言构造 (当使用一个被改过的 Java 编译器时可能会出现这种情况)。这个验证例程检查代码以确证其中没有出现: 伪造指针、访问受限制内存、以对象定义以外的方式访问对象。这个检查也确保调用函数时用了参数的正确类型和数目及没有溢出。
- 装入时类名和访问限制的验证。
- 一个界面安全系统, 在很多层次上实施安全政策。

- 在文件访问层次上，如果一个字节码试图访问一个未经允许的文件，则将弹出一个对话框，允许用户继续或停止执行。
- 在网络层次，将来的版本将包括使用公开密钥加密和其它密码技术的工具去验证已通过网络的代码的来源和它的完整性。这种加密技术将对网络上的资金流动至关重要。
- 在运行期间，与字节码的来源有关的信息将被用来决定代码可做些什么。这种安全机制可以辨别字节码是不是从防火墙内部产生的。用户可以设置自己的安全对策以限制用户不信任的代码。

小 结

Java 程序设计语言是唯一被设计成可在网络上传送可执行内容的语言。作为一种语言，它给编程者编制很多软件提供了灵活的特性。Java 同样保证了在不同平台之间的可操作性和安全性。

- Java 程序设计语言与一种特殊的浏览器和字节码解释器一起工作。Java 能在 World Wide Web 的通信环境内存在，因此，它处于网络上大量支持信息获取的数据通信的应用程序的“顶端”。
- Java 语言是面向对象的，并被特别设计成支持分布式的、可执行的应用程序。
- 在操作上，Java 语言编译器产生将被通过网络下载到用户计算机上的字节码，用户计算机运行这些字节码。
- Java 软件组成包括 HotJava 浏览器、Java 解释器、Java 编译器和开发 Java 应用程序的工具。
- Java 的安全性设计与在网络上发布可执行内容相适应。

3

Java 改变了 World Wide Web

World Wide Web 已经极大地改变了联机世界，并将更加普及。作为一种通信系统，Web 可以给信息提供者广域地、瞬时地发布和收集信息的能力。对用户来说，Web 是对世界范围内百万人的著作和思想的动态考察。Web 从非线性思考的想法起源，是 Internet 上的信息集成器，将在电脑空间中扮演主要角色。

Java 带给 Web 的是一种新的通信形式。Java 的可执行内容使支持 Java 的浏览器“聪明”起来，而不是依赖 Web 服务器提供信息和功能。

本章简单地探究了 Java 怎样改变了 World Wide Web。Web 用超文本组织信息，支持一类通信、信息和交互作用。与超文本一起使用的多媒体叫超媒体，可以丰富 Web 的信息。与 Web 的超文本一起使用的特殊编程技术（如接口编程）和语言（如 Java 或虚拟现实造型语言）可以扩展 Web 的交互性、信息的传送和通信的可能性。

为了解 Java 用在信息的广域发布上的力量，首先必须理解什么是 Web 和 Java 对 Web 的改变的意义。如果读者经常使用 Web，通过前两章的阅读可能已经明白 Java 如何扩展了 Web 的能力；读者也许想跳到第 4 章去阅读细节。本章将仔细探究一下 Web 和其中的 Java 成分。

Web 纵览

World Wide Web 最初是为了满足高能物理界的研究者对信息的需求而发展起来的。今天, World Wide Web 提供了一个局域地或广域地发布超媒体信息的系统。技术上讲, World Wide Web 支持着一个无缝的、广域的多媒体通信系统。信息被有机地组织起来并按用户的要求传送。本节简单地评价 Web 的历史起源和网络技术的思想的汇合如何在今天的广域 Web 上做出成就的。Java 正是在超文本中和 Web 通信方面一系列发明的最新成果。

导致 Web 的思想

Vannevar Bush 在其 1945 年 7 月发表在《大西洋月刊》(The Atlantic Monthly) 上的题为 “As We May Think” (这篇文章可于 Web 上下列地址取到, <http://www.csi.uottawa.ca/~dduchier/misc/vbush/as-we-may-think.html>) 的文章中描述了一个有机地连接信息的系统。

超文本的起源

Bush 叫这个系统为 memex (memory extension), 计划把它作为一个帮助人们的思想与信息接触的工具。Bush 注意到以前的发明已使人类在处理物质世界方面的能力得到扩展, 他希望他的 memex 能使人类的认识在利用人类思想的联系性的本质上有所扩展。

1965 年, Ted Nelson 创造了超文本 (hypertext) 这个词以描述非常接近 Bush 的模型的文本。但 Nelson 的文本不受顺序性约束。如 Nelson 所描述, 超文本连接文件组成一张相互联系的网, 这种关系使文本中 “平凡的” 一行可以连接到其它文本, 文本意义有了扩展和增强的能力。超文本远远不止是像作为关于一个文本的评论或进一步信息的脚注一样; 更进一步的, 超文本扩展了思

想的结构，它使大块的思想或信息可被包括进多种文本的很多部分而可以被利用。Nelson 也发明了超媒体 (hypermedia) 这个词，指不仅限于文本的超文本。超媒体可以包括多媒体的形式——图片、图形、声音和电影。

Web 的起源

Vannevar Bush 和 Ted Nelson 的关于信息系统的思想，在 80 年代晚期的一个计划中又显示出来。1989 年 3 月，瑞士日内瓦的欧洲核子研究中心 (CERN) 的一位学者 Tim Berners-Lee 提出了一个超文本系统以使高能物理界成员可以更有效地分享信息。Berners-Lee 在文本处理、实时软件和通信方面有背景，在 1980 年曾经发展了一个他称之为“Enquire”的超文本系统。Berners-Lee 1989 年的计划，称为“超文本和 CERN”，引起很大的反响。下面是这个计划的重要部分：

- 用户界面，在所有平台上必须保持一致，将允许用户从很多不同的计算机上访问信息。
- 为此界面设计的方案必须能访问各种文件类型和信息协议。
- “普遍访问”的规定，允许网络上任何用户访问任何信息。

到 1990 年底，一个运行在 NeXT 计算机上的 World Wide Web 的操作原型和一个行模式用户界面（称为 WWW）完成了。Web 的关键部分已就位，尽管它还没有在网络上广泛使用。

整个 90 年代早期，对 Web 的兴趣在世界范围内传播增长。1991 年 3 月，WWW 界面被用在一个局域网上，同年 5 月，在 CERN 的中心机器上都可使用它。1992 年 1 月 15 日，WWW 界面可从 CERN 公开得到，CERN 小组在那年余下的日子里在世界范围内对研究者演示 Web。

Mosaic：第一个“杀手”应用程序

1993 年，对 Web 的兴趣迅速增长。美国依利诺依大学 Ur-

bana-Champaign 分校的年轻的大学生 Marc Andreessen 在为美国国家超级计算机应用中心 NCSA 的一个计划工作时，领导一个小组为 Web 开发了一个叫“Mosaic”的浏览器。小组在 1993 年 2 月发布了 X Windows System 上的 Mosaic 的 alpha 版本，这是 Web 的图形界面的第一批产品。Mosaic 由于它新鲜的外观和用鼠标键击设计的 Web 图形界面，引起了人们对 Web 和联机信息的极大兴趣。1993 年底，在纽约的 Internet 世界会议和展示的参加者渴望学习 Web 的图形界面。《纽约时报》欢呼说 Mosaic 是 Internet 的“杀手应用程序”。

1994 年，更多的商业玩家加入 Web 游戏，一些公司，包括 Spry 宣布了 Web 浏览器软件的商业版本。Marc Andreessen 和合作者于同年 3 月离开了 NCSA，与 Jim Clark (Silicon Graphics 前主席) 组建了一个公司，后来称为 Netscape 通信公司 (<http://home.netscape.com/>)。到 1994 年 5 月，对 Web 有兴趣的人是如此热烈以至第一次 World Wide Web 国际会议在日内瓦召开时人满为患。到 1994 年 6 月，有 1500 个已知的公开 Web 服务器。

到 1994 年中，CERN 的 Web 初始开发者们渐渐清楚 Web 稳定的发展应该在一个国际性机构的指导下进行。7 月份，美国麻省理工学院 (MIT) 和 CERN 宣布了 World Wide Web 共同体 (Consortium) 即 W³C 的诞生。

今天的 Web

今天的 W³C (<http://www.w3.org/hypertext/WWW/Consortium>) 指导着 Web 发展的技术开发和标准化。W³C 是一个由大学和私有企业组成的共同体，由 MIT 的计算机科学实验室 (LCS) 与 CERN (<http://www.cern.ch/>) 和法国的一个计算机科学研究所 (INRIA, <http://www.inria.fr/>) 共同合作下运作。

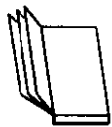
1995 年，Web 发展的标志是快速的商业化和技术改变。Netscape 通信公司的“Mozilla”浏览器继续包括超文本标记语言 (HTML) 的更多扩展，商业现金交易的安全性问题吸引了很多注

意力。1995 年 5 月, 已知公开的 Web 服务器已超过 15 000 个, 是一年前的数目的 10 倍, 很多公司于 1995 年加入了 W³C, 其中包括: AT&T, Digital Equipment Corporation, Enterprise Integration Technologies, FTP Software, Hummingbird Communication, IBM, MCI, NCSA, Netscape Communications, Novell, Open Market, O'Reilly & Associates, Spyglass, 以及 Sun Microsystems。

到 1995 年中期, Java 和虚拟现实造型语言技术的出现使 Web 开始另一轮快速改变。Sun Microsystems 开发了数年的 Java, 承诺将使 Web 呈现比以前多得多的交互性 (参见第 1 章)。虚拟现实造型语言, 允许开发者把三维图像造型化以通过特别的 Web 浏览器传送, 可能极大地改变所能提供的一切。

World Wide Web 的定义

撇开迅速成长和技术发展, 1995 年的 Web 保持着它 1990 年初创时的基本功能成分。但它作为对 Internet 观察方式的广为人知, 妨碍了对它的广泛了解。因为 Web 有时被看作等价于 Internet, 而浏览器有些时候被看作等价于 Web 而不是 Web 的观察工具。Web 实质上与 Internet 和浏览器有明确区别。首先, Web 不是网络, 而是应用程序系统 (软件程序集合); 第二, World Wide Web 可被扩展和使用于很多不同种类的网络 (并不必要是 Internet), 它甚至可以用在完全没有网络的情形或一个不与其它网络相连的局域网上。



Web 的一个比喻 想象一个图书馆, 其中所有书的书脊都被去掉, 建筑物中的引力也被去掉, 允许所有书页自由飘浮。如果人们可以用非常细的线把一页同另外一页连起来, 这将类似 Web 的超文本被安排的方式。页面自由飘浮, 用户可以从一本书的任意一页接触一本书, 并通过从这页连出的线接触到其它著作。

这里是一个 Web 的更技术性的定义：

World Wide Web 是一个依据客户机/服务器模型在 Internet 计算机网络上广泛使用的用来进行数据通信操作的超文本信息和通信系统。

图 3.1 总结了按此定义的 Web 的技术结构。

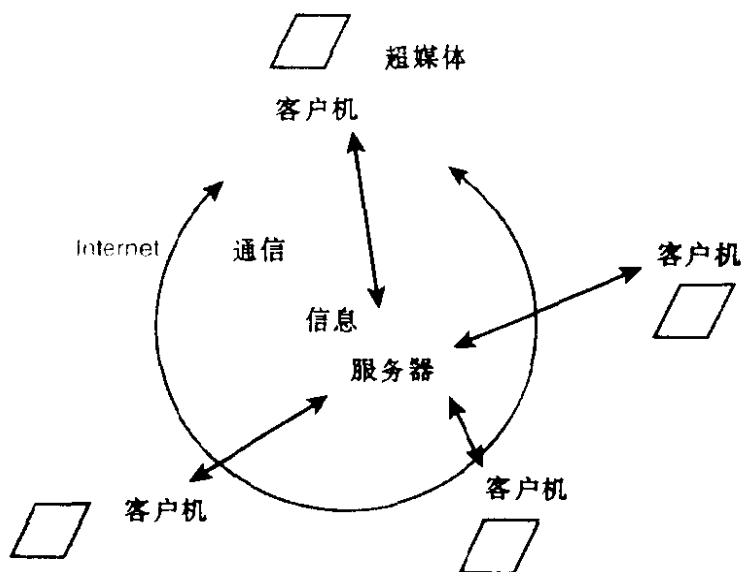


图 3.1 Web 的技术结构

Java 如何改变了 Web

Java 通过给 Web 浏览器带来更多的“智能”改变了 Web。尽管支持 Java 的浏览器有与其它 Web 浏览器很多相同的用户界面，它们技术操作的焦点有明显的改变。Java 的可执行内容要求支持 Java 的浏览器聪明些，即它们必须能解释可执行内容。

Java 支持客户机方交互性

网络计算机系统的客户机/服务器模型包括三个成分：客户机、服务器和网络。客户机是一种最常运行在最终用户的计算机主机上的软件应用程序。服务器是最常运行在信息提供者的计算机主机上的软件应用程序。客户机软件可与用户的硬件系统相适应，它是作为一个从系统到服务器上提供的信息的界面。用户可

统。一个顾客可以购买任何种类的电视机（客户机）去观看任何电视发射塔（服务器）发送的信号，不管用户用便携电视或投影屏电视，电视机以标准格式从发送站接收信息而以适合用户电视机的方式显示。电视节目不必为每种电视机拍摄不同的版本，无论是对彩色或黑白或不同尺寸的电视机。新建电视台将能向现存所有电视机发送信号。

Java 带给客户机/服务器模型另一种尺度。当然，Java 也遵守基本的模型：支持 Java 的浏览器是向 Web 服务器发送请求以获得信息的客户机，支持 Java 的浏览器解释和显示从服务器发来的信息，这种信息包括超文本和字节码。这些字节码是 Java 对这种模型的新贡献。Java 客户机执行从服务器发来的内容。这种字节码如第 2 章所描述的，是体系结构无关的，像其它从 Web 服务器发来的信息一样。

Java 可以消除对辅助应用程序的要求

辅助应用程序包括（不支持 Java 的）Web 浏览器用来把多媒体信息显示给用户的软件，例如，为使用户能看电影，Web 浏览器必须有一个安装好并可使用的电影播放软件，为显示 HTML 文件中的行间图形，Web 浏览器必须是图形的——使用 X Window 系统或 Macintosh 操作系统或 Microsoft Windows 作为图形用户界面。

为支持 Java 的浏览器开发，applet 的编程者可以创作内容处理器去处理媒体格式而不必依赖于辅助应用程序。

Java 加进新的 Web 通信环境和能力

Java 语言和它的浏览器是 Web 上总体通信环境的一部分。不管用户书写和发布 applet 或仅仅是观看 applet，用户已加入了发展了若干年的 Web 的通信活动和交流中去。因为 Java 还是新生的，它并没有在所有 Web 通信环境中出现。在本书以后诸章中，读

者将看到 Java 用在 Web 上更特别的例子。本小节简单地评论一下 Web 的环境和能力以及 Java 在其中能做什么。

Java 和 Web 上的通信环境

Web 上的通信可以以各种形式发生在很多环境中，一些风格，或传统的通信，已被包括在 Web 中，这些风格在很多方面对应于脱机人类通信环境：

- 人际的：Web 给用户提供了建立主页面的方法，这些页面一般将覆盖个人的或专业的信息。创建主页面的实践因定义“缺省”页面的技术必要性而出现，“缺省”页面指当发往 Web 服务器的请求信息只包含主机名或主机名和一个目录名时 Web 浏览器所显示的东西。主页面于是一般是一个服务器、组织或个人的最高层次的页面。由个人书写的主页面经常显示作者详细的个人信息和主页面的目录。同样，个人经常遵从惯例建立到同事和朋友的主页的连接，建立电子部落（数学上，这些电子部落由主页上描述 Web 的有向图上的集团定义）。个人主页面当在人际间使用时，提供一对一通信，尽管 Web 上所有页面的操作都是一对多的。

个人“applet”还没有占主导地位，但 Java 可能使个人可以创建其它的 Web 用户可以与之作用的可执行“persona”。

- 组：如上面人际间定义描述的，个人页面的集团可以定义一个特殊的 Web 部落或组，同样，人们可以在 Web 上建立与地理位置无关而仅仅在某个题目上有共同兴趣的联系。Web 上信息主题树的崩溃经常由描述一个主题的原始材料和资源表的发展和合作性连接产生（参见下节有关在 Web 上放置以主题为基础的信息的讨论）。类似地，一些人的小组在 Web 上因通信上共同的兴趣而联系起来。
- 组织的：很多在 Web 上出现的最初的 Web 服务器不是属

于个人，而是属于某个组织，于是一个服务器的主页面经常表示着这个服务器的拥有者，一个研究所或组织。用这种方式，校园信息系统 (CWIS) 的风格由教育性研究机构的服务器生成。类似地，商业的、政府的和非政府组织在很大程度上遵循 CWIS 建立的形式。

很多组织现在在他们的 Web 页面中利用 Java 增加趣味性并向用户提供服务，在下章中读者将看到这样的例子。

- 大众：正如其它媒体已被用来做信息的一对多传播（报纸、收音机、电视）一样，Web 也被用来做大众通信。很多商业的和非商业的杂志和其它出版物正通过 Web 发布。更进一步，如以前提到的，所有可公开访问的 Web 页面对任何使用 Web 的人是潜在可读的，所以也是潜在的一对多通信。

Java 正被积极用在大众通信，如第 1 章中引自 NandO Times 的例子所显示的那样。

必须理解的一个重要概念是作为通信系统的 Web 可以被用来以多种方式进行通信。通信的分类（按以上列出的种类）依赖于谁加入了通信。Web 上表示的任何精确分类可能因任何 Web 页面潜在的广域扩展而被模糊化。因此，一个个人主页面可以在人际间被使用，但它在 Web 上被访问的次数可能数倍于特意为大众消费而设计的出版物。Java 在传送可执行内容方面的能力给这些种类增加了新的可能性。

Java 和 Web 的潜力

Web 是一个可被用于很多环境的灵活的通信系统，这些环境从在主页上的个人通信到组通信到大众通信，除了这些环境外，Web 还提供以下功能：

- 信息传送：Web 浏览器为用户提供了“透视”FTP 空间、Gopher 空间、或 Web 上的超文本信息的手段。超文本的结构

使用户可以作出选择，因为用户可以在一个超文本中作出很多选择去顺着连接前进，Java 增加了新的协议处理器和内容处理器的功能。

- 通信：人们可以用 Web 超文本举行共享信息、讨论和帮助小组成员相互接触的座谈会。Java 的可执行内容引进了更多的交互性通信的新形式。
- 交互作用：Web 开发者用接口程序可以建立与某应用程序某种程度的交互性，提供给用户一种基于请求的获取约定信息的途径。接口程序也允许用户改变或增加一个信息结构，使用 Java 因其可执行内容而可能获得更高程度的交互性。
- 计算：Web 用接口编程可以提供到其它应用程序和信息处理程序的界面，Web 应用程序根据用户的选择可以通过一个接口程序返回一个计算的或约定的结果。

图 3.3 中的符号可用来描述上述 Web 的某些功能。

图 3.4 用图 3.3 中的符号显示了选择性和接口程序交互性的重要区别。当用户访问左侧的 Web 服务器，内容被用超文本显示。超文本页面中的连接给用户很多选择去访问数据库中的信息，但是，信息并没有按用户的输入定作，或基于用户的要求而计算。尽管这个服务器因页面上超文本的设计而在信息获取方面给用户很大灵活性，但它不是交互性的。

交互性层次的关键，是可执行程序通过 Web 页面接收用户的输入。这种基于用户输入的交互性可以计算出一个结果（可能是利用数据库的信息）并把这个约定的结果返回给用户，更进一步，可执行程序同时允许用户（可能）改变数据库的内容，或对服务器上的文件或数据库进行其它一些修改。这些改变可能包括改变超文本的结构或内容或其它文件的内容，这个可执行程序的构造要求在接口编程方面的技 。

Java 还增加了另一层次的交互性。支持 Java 的浏览器就是一种计算机制，而不必用服务器计算一个结果。



图 3.3 表示 Web 功能的符号

小 结

- Web 从关于信息的联系的、非线性的组织的思想发源。Java 是这个发展过程的另一步。
- Web 是以客户机/服务器模型广泛使用在 Internet 上的超文本信息和通信系统，它通过适当的浏览器有可能提供超媒体显示，某些浏览器需要辅助应用程序。
- 支持 Java 的浏览器把客户机一方的交互性和计算引入 Web，并可以消除对辅助应用程序的要求。
- Web 上的通信可以允许很多形式，可在很多环境下实现，

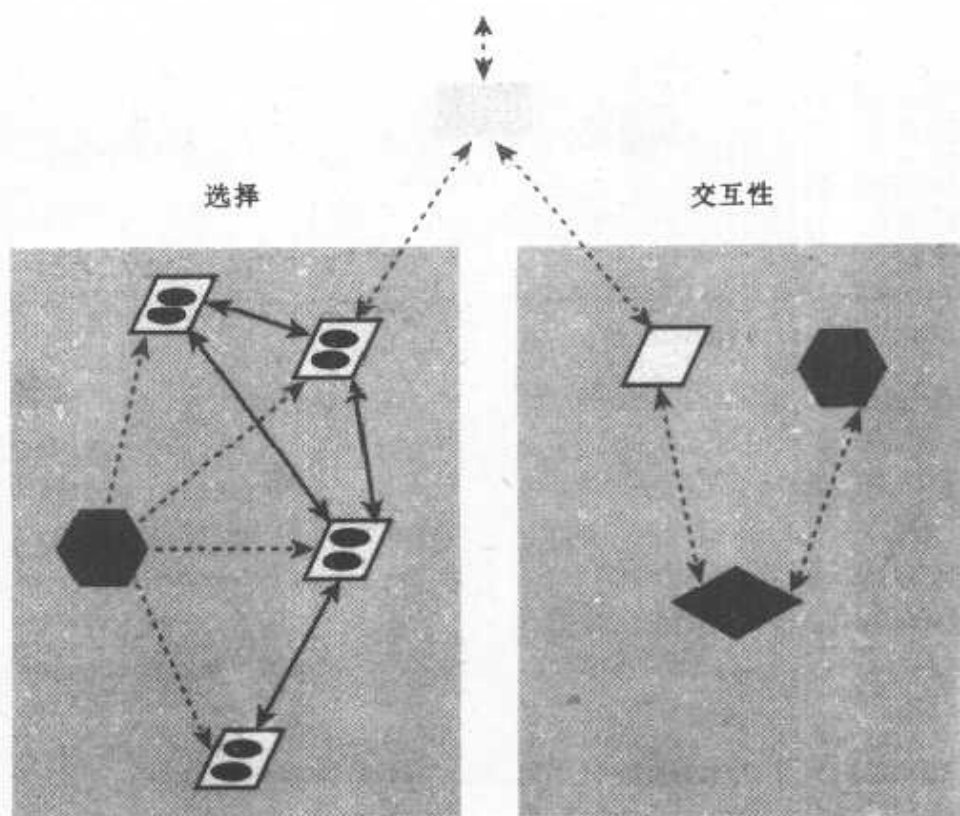


图 3.4 Web 选择性和接口交互性

从个人通信到小组和大众通信, Java 可以潜在地扩展 Web 通信的环境和功能。

第二部分 开发 Java 的潜力

本书着重讨论了 Java 在网络通信方面的变化能力 (transformative power)。通过列举一些从不同的侧面给 Web 页面带来活力的示例,这一部分中的三章着重向我们讲述了 transformative power 的确切含义。

第 4 章探讨如何在 Web 页面中进行动画制作,从简单的字母移动到复杂些的动画片制作,或者一些科教功能的实现。使用 Java 编程,可以使 Web 页面上的文字或图画翻滚、移动或重复一个预先编制好的例程。Java 还可以完成与图像映射 (imagemap) 相同的功能。

第 5 章讨论了 Java 的交互功能,展示了用户的响应如何影响在一个 applet 中执行的信息流,使得某种专门学科的“家庭教师”成为可能,Java 还可以把 applet 制作成虚拟的科学仪器,通过与模拟现实世界现象的 applet 交互作用,用户有了一个研究和学习科学概念的机会。

最后,第 6 章研究了 Java 用于分布式网络工作通信的可能性。Java 的 applet 实现了可执行内容的发送,也使得新型的协议和格式成为可能,Java 还实现了网络信息的动态提取,使得网络浏览器可以自动导航或者导游一次网际页面的旅行。

这几章包括以下内容:

- Java 制作动态网络页面的基本途径的有关知识,包括一些特殊的 applet 示例以及如何用 HTML 的 APP 标记将它们包含到网络页面中去。

- Java 的交互性能如何使得特定的、瞬时的用户响应成为可能。
- Java 如何实现内容的分布；程序员是如何创建新的协议处理器及内容处理器以及用 Java 获取网络信息的基本技巧。

使 Web 页面动起来的 Java

Java 语言和支持 Java 的浏览器给人们带来了一个比以往更好的动态可视的网络，以前，超文本页面只能包含静态的图像，而必须通过辅助应用软件来播放影像。但现在的 Java 网络页面自身就可以容纳动态的图形，文字或任何一种 Java 程序员所能梦想的动态可视元素。

在本章中，我们概观了几个实现动画功能的 Java applet。有些地方还列出了有关源代码中的关键部分来说明这些 applet 是怎样制作的。如果你想更详细的理解这些代码段，请参看本书第五部分的 Java 程序设计基础。否则你也可以略过本章中那些程序代码的部分而在读过第五部分后再回过头来读它们。如果你想试用一下这里描述的 applet，则应对 Java 和 HTML 之间的链接有所了解（参见第 2 章）。

这一章的目的是使你熟悉多种用 applet 实现的动画类型，如果你正准备在自己的 Web 页面上安放一些 applet，那么本章也会对你非常有用，它将指导你如何获取一些公共的示例 applet 并将它们包含在一个超文本页面中使用。

移动中的 applet

如果是初次使用支持 Java 的浏览器，你会立即注意到，一些 Java 页面正在运行动态的文本、图像或在显示动画。这些动态的图像是由使用了可运行 Java 接口功能的 Java applet 实现的。这些 applet 不再仅仅是陈列一幅幅静止的文本或图像，而是可以连贯

的展示一段内容。

错乱的文本 (Nervous Text)

有关动态文本的一个例子是“错乱的文本” applet，它最初是由 Center for Applied Large-Scale Computing 的 Daniel Wyszynski 编制的。Wyszynski 的“Nervous Text” applet 在屏幕上用互相挤撞的字母显示 HotJava! 字样，David Leach 修改了这个 applet 使得它可以显示任意程序员想指定的字符串。图 4.1 展示了实现在 Web 页面上的 Wyszynski 和 David Leach 的“Nervous Text” applet。

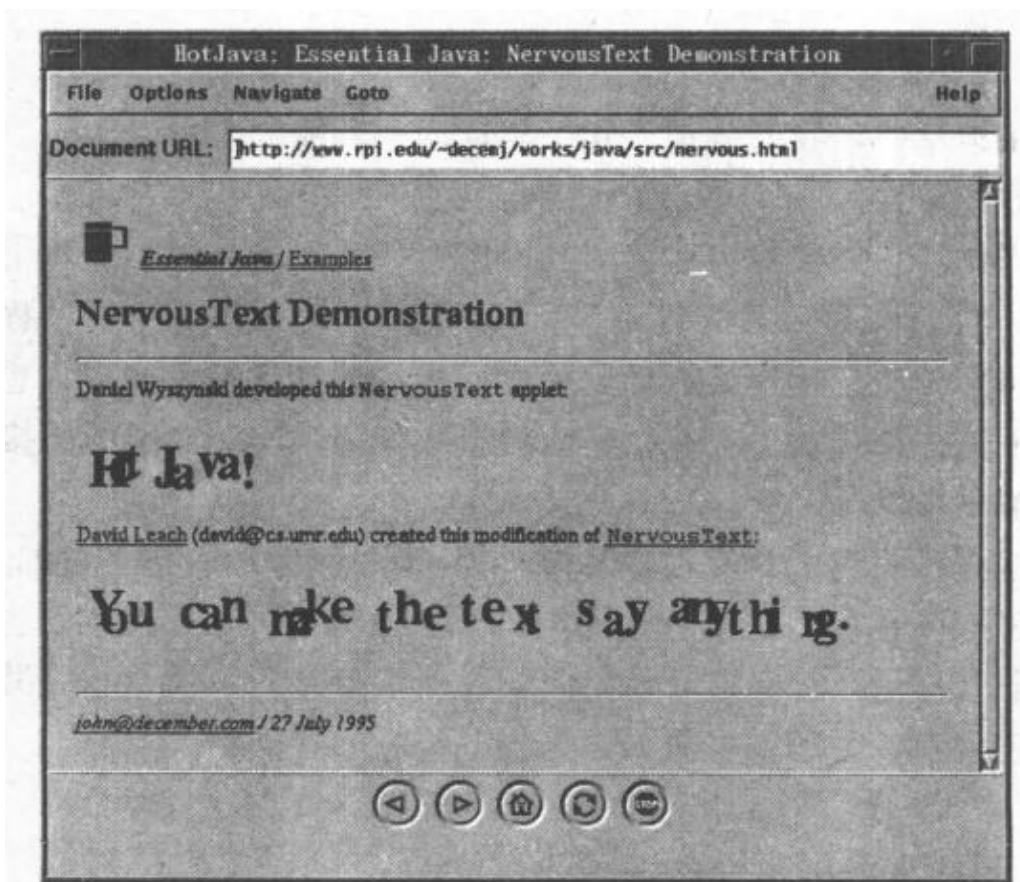
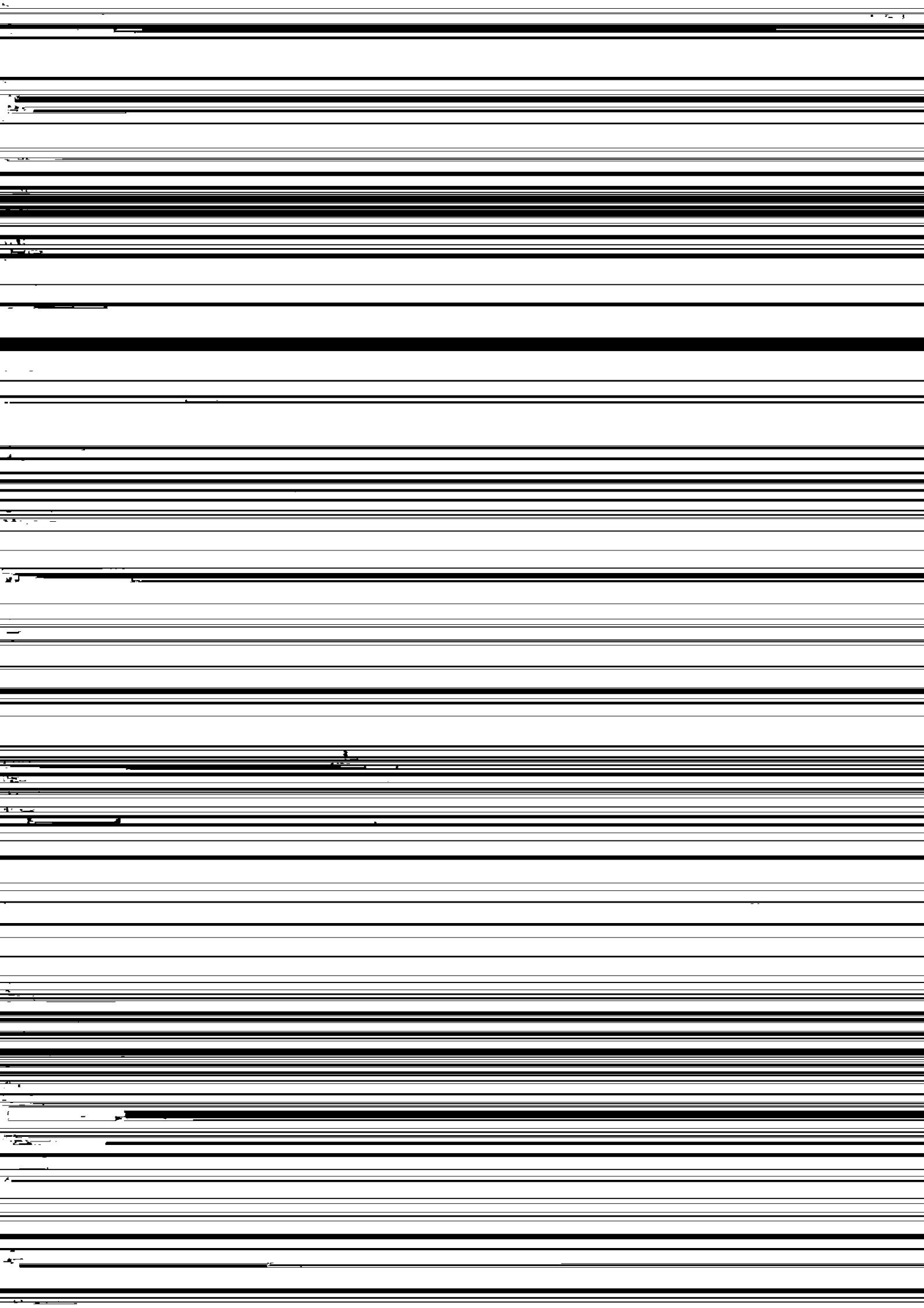


图 4.1 The Nervons Text applet

(由 Daniel Wyszynski 和 David Leach 免费提供)

这个 Nervons Text applet 很好的告诉了大家，一个 applet 可以在任何 Web 页面而不仅仅 applet 的开发者的页面上使用。除了使用自己写的 applet 外，你还可以修改和使用他人开发的 applet，



提示

如何在没有 Java 的情况下使用 Java Applet 你可以在没有 HotJava 浏览器或 Java 编译器的情况下用 APP 语句将 Java Applet 组织到一页 Web 页面上。你只需要有为 applet 服务的编译好的类文件。如果你要用远程结点上的 applet,就需要指明该 applet 的字节码文件在网络上的存放地点。可以用 APP 元素的 src 属性来完成这项工作。当然,没有 HotJava 浏览器的用户是无法观看 applet 的。

如果要重复使用某个远程 applet,建议你下载它并在本地结点上做一个类文件的拷贝。但在此之前,请先与信息的提供者核证一下。当然,还需要确认一下该 applet 是否是可执行内容并可在任何要求观看它的计算机上运行。

在 paint()函数中 David 用到了这个数据串来获取 applet 要绘制的字符。而使字符变得“错乱”的技巧是通过在 paint()中用随机数产生它们的 X、Y 坐标而破坏了平衡来实现的:

```
x_coord= (int) (Math.random ()*10+15*i);
y_coord= (int) (Math.random ()*10+36);
```

自动显示器 (TickerTape)

与 Nervous Text applet 相同, TickerTape 是对 Java 的动态性能的另一个良好的例证。这个 applet 最初是由 Hotwired 的 Sven Heinicke 编制的, 后来由 Missouri-Rolla 大学的 David Leach 和 John stone 改进。你可以通过在 HTML 页面中设置属性来改造 TickerTape。这些属性有以下几种:

属性	注释
MSG=String	要显示的信息串

WIDTH=int	显示字串的宽度
RED=int	文本的红色选值 (0~256)
BLUE=int	文本的蓝色选值 (0~256)
FNT=name	以何种字体显示文本的字体名
GREEN=int	文本的绿色选值 (0~256)
SPEED=int	每个周期中移动信息的速度
REST=int	在移动开始之前等待的时间 (毫秒)

你可以在 HTML 中设置这些属性。下面是一个设置了三种 TickerTape 的例子：

```
<p><APP Src="http://www.hotwired.com/java/"
      Class="TickerTape"
      msgs="This is an example of a ticker tape"
      width="100">
<p><APP Src="http://www.hotwired.com/java/"
      Class="TickerTape"
      msgs="Several tickers can be on a page"
      red="0"    blue="256"    green="100"
      width="500"
      fnt="Helvetica"
      size="30">
<p><APP Src="http://www.hotwired.com/java/"
      Class="TickerTape"
      msgs="... and have different colors, fonts and sizes..."
      red="5"   blue="6"     green="250"
      width="500"
      speed="25"
      rest="2"
      fnt="Courier"
      size="72">
```

图 4.2 展示的是上例的结果 TickerTape 的运行画面 (从动态演示中截取的一幅)。最底一行的自动显示带以非常快的速度移

动，同时上面两行在持续的向左边滚动。

提示

使用 Java 的用户会浪费带宽吗？通过网络将一个 Java applet 的字节码拷贝下来以后，执行它们的就用户的主机了。信息提供结点的工作仅是发送那些 bytecode。这样，applet 用户就可以大大减少原来可能会用在信息提供结点上的大量的带宽和时间，而那是网络所不愿意碰到的。

并且，包含 bytecode 的类文件并不总是很大。如图 4.2 的 TickerTape applet 为 3186 个字节——比许多日常从 Hotwired 服务器上拷贝的图形软件都要小。因此，使用 applet，用户可以看到更好的动态效果而并不一定非要占用更多的网络带宽。当然，如果让浏览器进入自动导航状态（如第 6 章中提到的 Surf-o-Matic applet）而把它放下不管，浏览器一定会占用不少的带宽来下载 Web 页面。

对信息提供者来说，他必须小心掌握他们的 applet 的大小和运行需求。一个大量耗费 CPU 的 applet 可能会导致用户计算机的崩溃。

TickerTape applet 使用了一种使字母滚动的关键的编程技术。它的思想是这样的：在每次重写字符串之前，程序都用在 Speed attribute（速度属性）中定义好的速度来移动字母的 X 位置。下面是作此工作的代码段：

```
xpos -= speed
```

这行代码的意思是从当前字符串横坐标位置的值中减去速度值，`xpos -= speed` 是 `xpos = xpos - speed` 的缩写形式。

焰火表演 (Fireworks)

另一种类型的动画包含了图形——不再仅仅是单词——在页

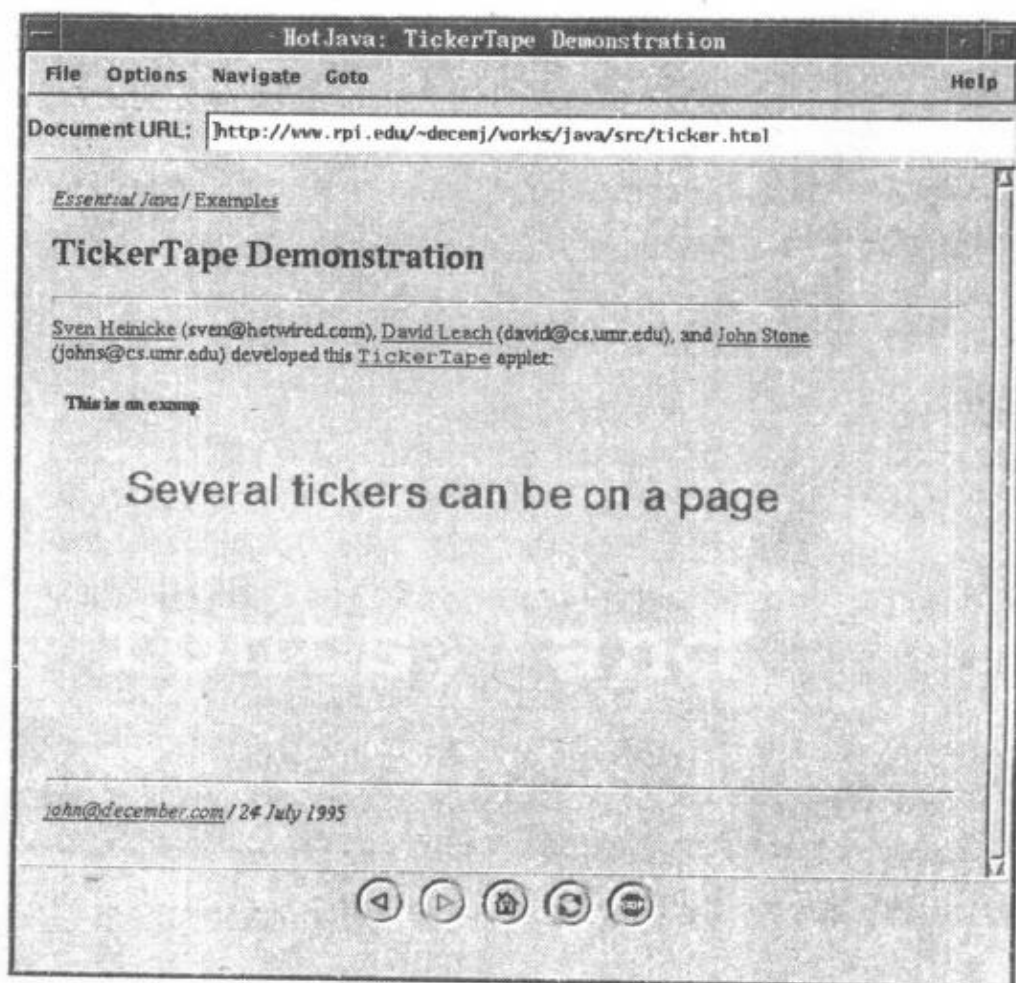


图 4.2 自动显示器 applet (由 Sven Heinicke, David Leach, John Stone 提供)

面上的闪现。Erik Wistrand 制作了一个将一幅页面变成焰火表演的 applet (见图 4.3)。

与 TickerTape applet 相同, 你可以在自己的 Web 页面中加入 Fireworks applet, 也可以控制它的外观。Fireworks 的参数如表 4.1 所示:

表 4.1

属 性	说 明
ROCKETS=int	活动的焰火堆的个数 (缺省为 3)
POINT=int	每个焰火堆点的个数 (缺省为 3)
POINTSIE=int	每个点的大小 (缺省为 2)
LIFELength=int	焰火持续的时间, 以火焰数计 (缺省为 100)

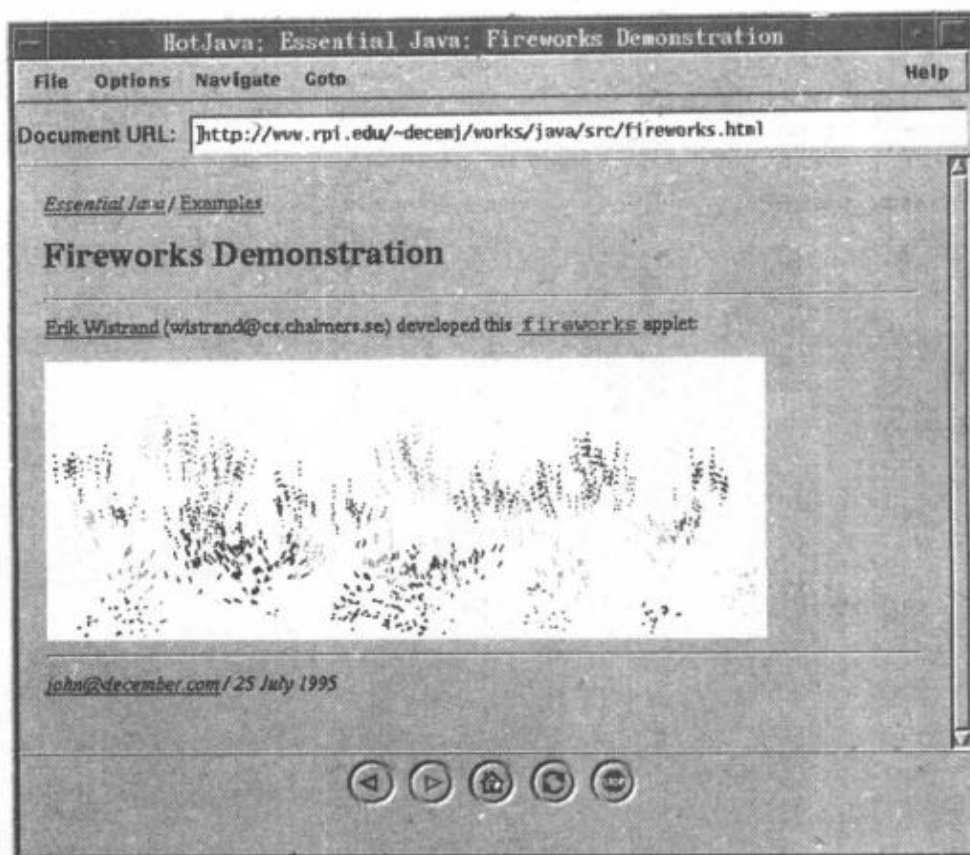


图 4.3 焰火 applet 示例 (由 Erik Wistrand 提供)

续表 4.1

属 性	说 明
GRAV=int	重力; 值越大, 焰火坠落越快 (缺省为 10)
DELAY=int	每两个火焰之间间隔的毫秒数 (缺省为 50)
TRAIL=int	火焰尾迹包含的点数 (缺省为 10)
COLOR=rrggbb	applet 背景色的红、绿、蓝三色的十六进制值 (缺省即为浏览器的原背景)
WIDTH=int	Applet 的宽度 (缺省为 tiny=浏览器缺省值?)
HEIGHT=int	Applet 的高度 (缺省为 tiny=浏览器缺省值?)

下面是在一幅页面中加入 Fireworks applet 所用的 HTML 编码:

```

<p>
<APP Src      = "http://www.cs.chalmers.se/~wistrand/"
Class        = "fireworks"

```

```
ROCKETS      ="50"  
POINTS       ="20"  
LIFELNGTH    ="30"  
GRAV         ="20"  
DELAY        ="3"  
POINTSIZE    ="2"  
TRAIL        ="4"  
COLOR        ="ffebcd"  
WIDTH        ="500"  
HEIGHT       ="200"
```

>

这个例子在页面上设置了一组 50 个焰火(如图 4.3 所示)。在颜色属性中用 16 进制代码定义了背景图像的红、绿、蓝三色值。

动画 (Animation)

并不是所有的动画都只被用于美术目的, 另外一些动画被用在科教性页面中或是成为用户与 Web 页面进行交互作用的一个组成部分。

杂技 applet (Juggling)

Chris Seguin 制作了一个介绍如何玩杂技的页面(<http://www.acm.uiuc.edu/webmonkeys/Juggling/>)。在这里, 他充分利用了动画来演示而不只是用描述来表明如何玩一项杂技。请看图 4.4 的 juggling 页面。从 HotJava 浏览器上看, 这个页面展示了两个模型手怎样交替的抛一个、两个直到三个小球。

Chris 的 applet 源程序中的一个关键技术是他用了阵列来存储球的运动轨迹。他沿着该轨迹不断重新放置同一个球的图形。这与在卡通片中用一张张的画页来显示一个球在其轨迹上的不同位置不同。

Java 程序员在实现动画时用到了图形轨迹和不同的图画帧两种技术。一般来说, 用图形轨迹技术既可以减少对内存的占用

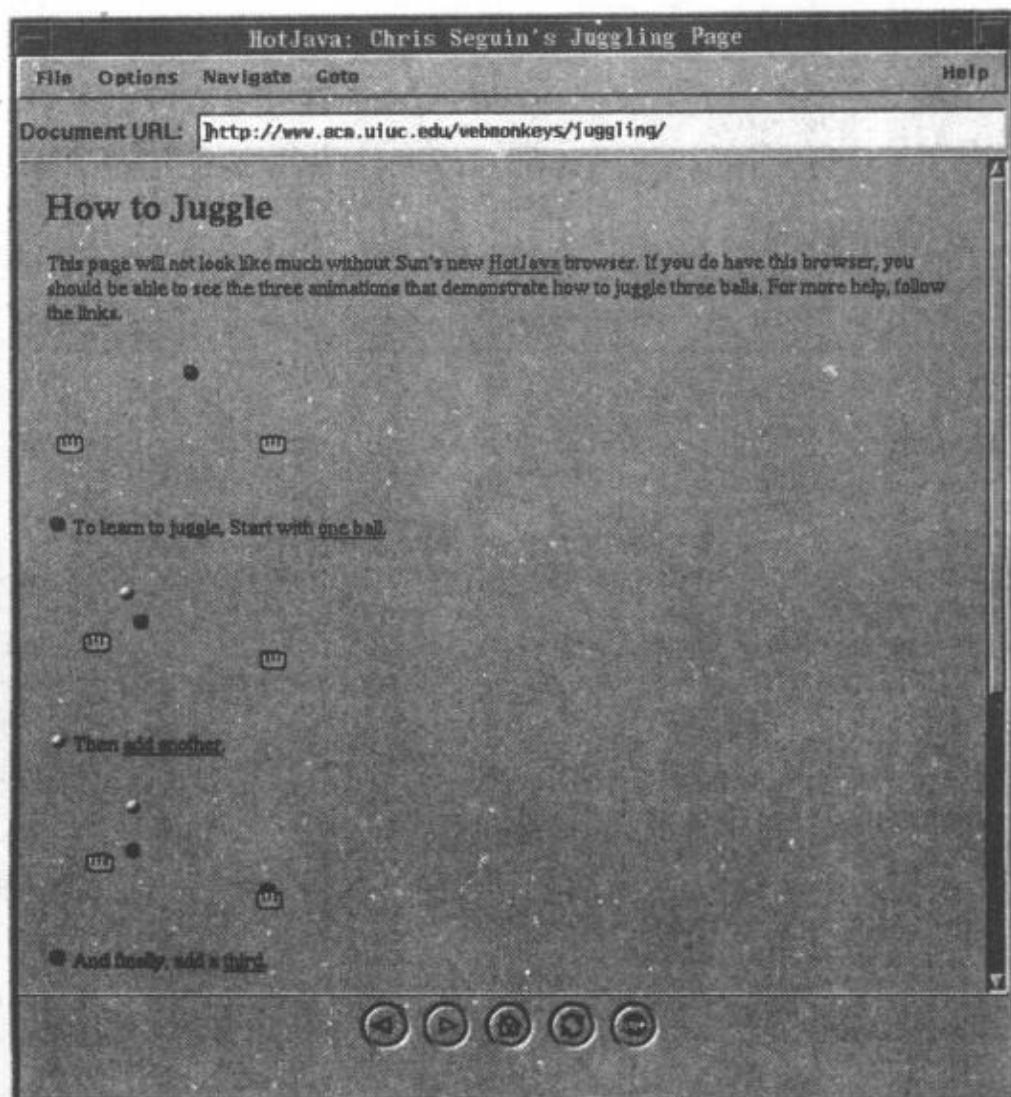


图 4.4 Java Juggling 页面 (由 Chris Seguin 提供)

也更灵活, 但对另一些复杂的图像 (如第 1 章中的 Duke), 则还必须采用多帧图画的技术。

涂抹名画 (Drawing on Famous Pictures)

Web 页面中另外一种类型的图形使你可以在一些画页上随意涂抹。Johan van der Hoeven 制作了一个非常引人的魔术 applet。你可以用它在一些名画上再画些东西, 就好像你在用一种魔术画笔似的。图 4.5 演示了在世界地图和蒙娜丽莎像上所作的一些涂写。

下面是 Johan 的“魔术”applet 的一部分编码。你可能已经注

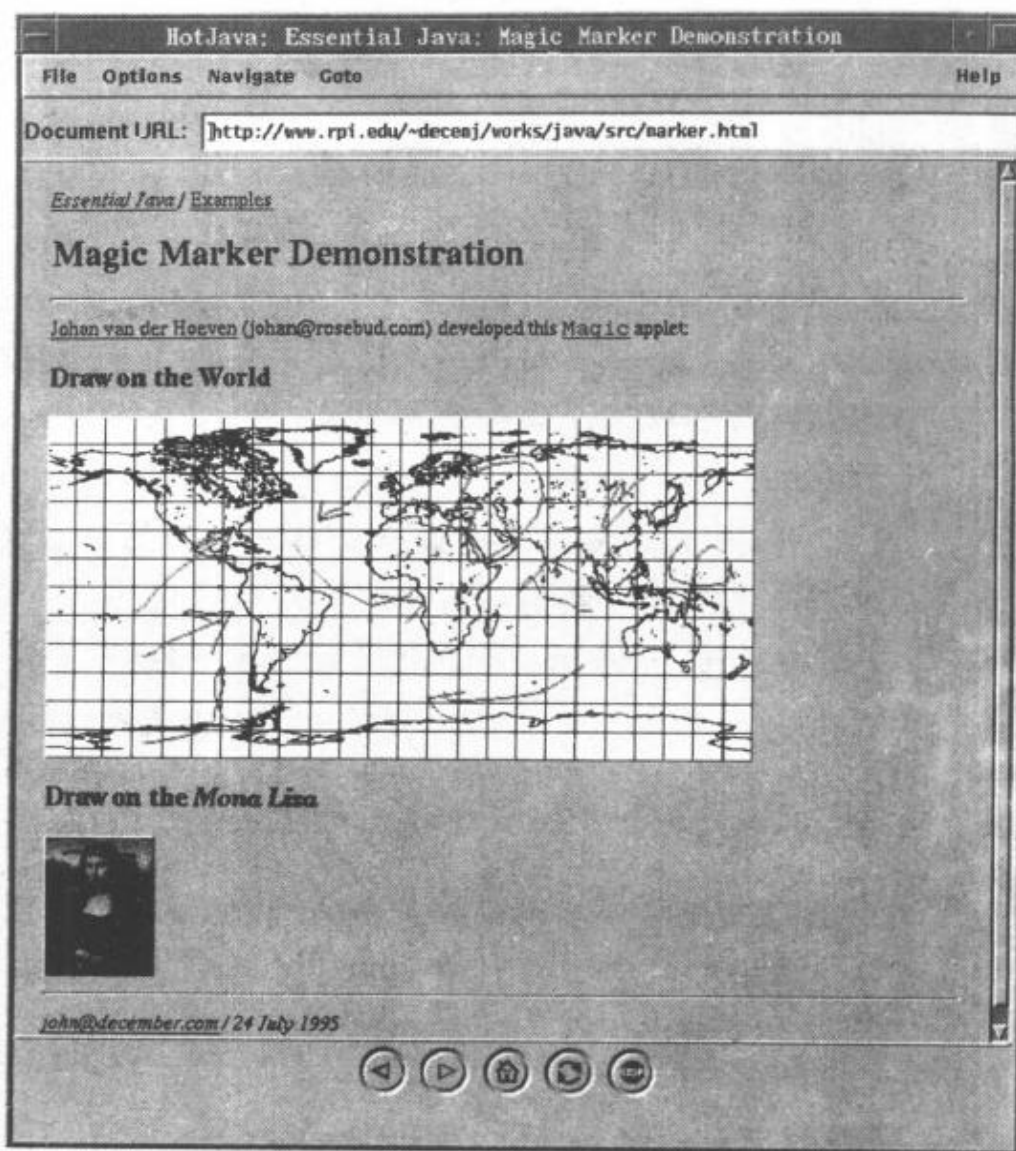


图 4.5 在世界地图和蒙娜丽莎画像上作标记
(由 Johan van der Hoeven 提供)

意到了：原图可以放在网络上的任何位置。

```
<H2>Draw on the World</H2>
```

```
<APP Class = "Magic"
```

```
Src      = "Http://www.rosebud.com/rb/art/jo/"
```

```
imgurl   = "http://www.rpi.edu/~decenj/images/world.gif"
```

```
linecolor = "red">
```

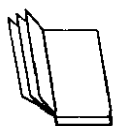
```
<H2>Draw on the <CITE>Mona Lisa</CITE></H2>
```

```
<APP Class = "Magic"
```

```
Src      = "http://www.rosebud.com/rb/art/jo/"
```

```
imgurl   = "http://www.paris.org/Musees/Louvre/Treasures/gifs/Mon-
```

```
aLisaa.gif"  
linecolor    ="blue">
```



谁拥有什么？ 这一节中的绘画 applet 演示了 Java 给网络带来的关于作者权和所有者权的混乱。HTML 页面的作者使用了他人写的 applet 和另一个人制作到网络上的图画来给用户提供了一个修改这幅画的环境，而该画的原画又是在很多年以前由另一个人 Leonardo Da Vinci (达·芬奇) 绘制的！即然这样，那么谁应该算是最后的结果页面的作者呢？谁又是这个混合物的拥有者呢？是制作该 applet 的天才的 Java 程序员？还是把这种种零件组合起来的 Web 页面作者？是浏览器的制造商？还是修改图画的用户？亦或还是画的原作者？达·芬奇绝对不会想到，多年以后，他的画会被人们在世界各地如此广泛的传播并被人们怀着如此兴高采烈的心情涂个面目全非。这个问题引起了一部分关于网络和 Java 使用的法律上以及知识产权上的争议。

动态反馈图像映射 (A Live Feedback ImageMap)

还有另一类的 Java 图形可以实现像 HTML 的图像映射那样的功能：当用户在图像的某个特定位置按下鼠标，将会调用另一个资源。Sun Microsystems 的 Jim Graham 实现了一个演示这类功能的 applet。如图 4.6 所示，这个 applet 演示了与 HTML 图像映射相等的功能，还增加了动态反馈 (live feedback) 的特色。当用户将光标移至画面上面的一个“热点”地区时，这个区域会被高亮度显示以告诉你在此处按下鼠标将调用另一个资源或者一些媒体内容。



图 4.6 Live Feedback ImageMap
(由 Sun Microsystems 的 Jim Graham 提供)

使用 Java 的商业节点

本章中的例子演示了许多种 Java 的动画功能。但是 Java 不仅仅是一个好的演示器,它也已经被用在了许多商业网络结点上。图 1.10 所示的 George Coates Performance Works, 图 1.13 的 ESPNET SportsZone, 还有 Dimension X (<http://www.dimensionx.com/>), HotWired (<http://www.hotwired.com/>), Sun Microsystems (<http://www.sun.com/>) 等等。由于给 Web 页面带

来了如此多的视觉乐趣, Java 在吸引大众注意、传播特殊信息和提供娱乐等方面都有着巨大的潜力。

滚石乐队 (The Rolling Stones)

当滚石乐队在 Internet 的 Multicast Backbone (MBONE—<http://www.eit.com/techinfo/mbone/mbone.html>) 上广播他们 1994 年 11 月 18 日的 Dallas Cotton Bowl 演唱会的部分内容时, 着实大大的风光了一番。现在, 他们用 Java 来展示了。

Stone 的结点上有数个有趣的 Java applet:

- Stone puzzle (<http://www.stones.com/javapuzzle.html>): 用它你可以通过滑动小方块来改变那著名的声音的音调。
- The Stone Java devil (<http://www.stones.com/new.html>): 它播放此节点的 What's new (有什么新内容) 页面。
- 在 Stones Voodoo Lounge (<http://www.stones.com/javaindex.html>) 的开放页面中, 动态的旗子在屏幕上前前后后的穿梭。该页面如图 4.7 所示。

精确信息系统顾问 (Accurate Information System Consultant)

其它的一些商务服务者用 Java 给他们的页面加入动画以增强其吸引力。Accurate Information System (<http://accurate.com.my/>) 就用了 Java 的移动显示屏和跳跃的小球来欢迎用户 (见图 4.8)。

当然, applet 也可能像 BLINK 标志或其它一些图形元素那样在网上被用得太多以至到了滥用的地步。然而, applet 在为网络用户提供动画, 乐趣、信息以及真正的服务的许多方面是十分出色的, 我们还需要充分发掘 Java 的潜能。

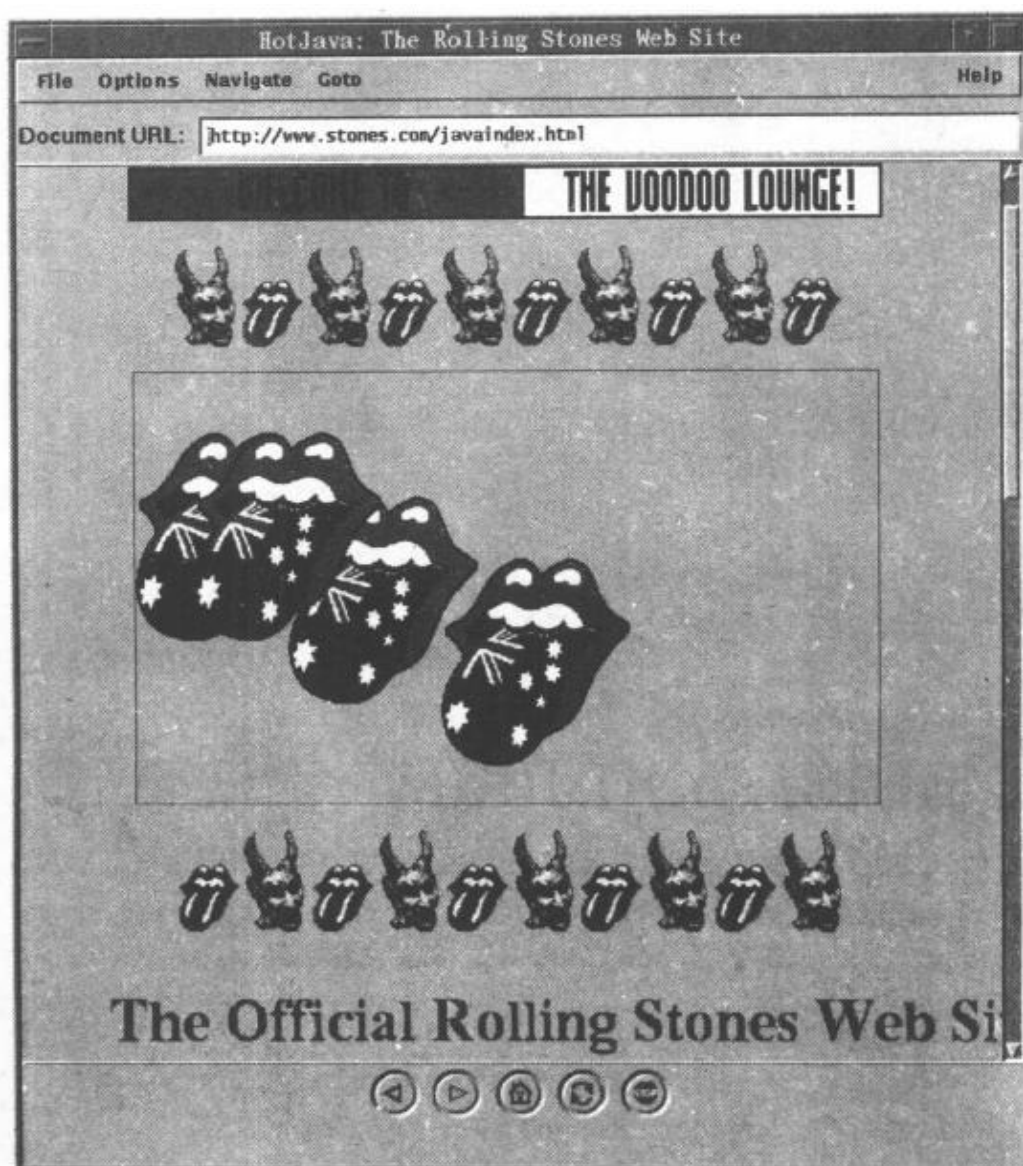


图 4.7 滚石的 Java Voodoo Lounge
(由 Stephan Fitch 和 John Graham 提供)

小 结

你可以用 Java applet 给 Web 页面带来动感。用户可以用 APP 元素给已制作好的 applet 设置参数并把它们加到自己的页面中。开发者也可以创造一个完成新功能的 applet 并把它提供给用户。

●用 NervousText applet 文字会在屏幕上跳跃闪现。用户可

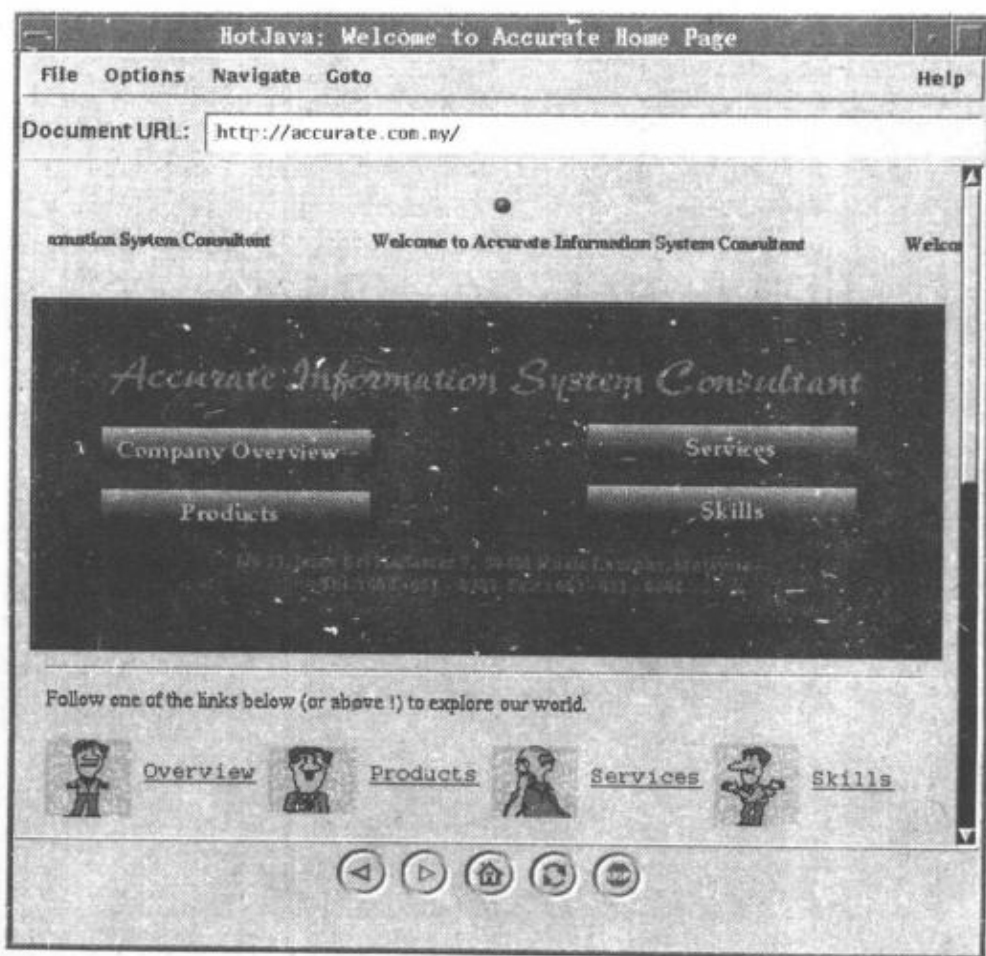


图 4.8 Accurate Information System Consultant
用自动显示屏打出的欢迎画面

(由 Accurate Information System Consultant 提供)

以用 APP 元素把这个 applet 放到他们的页面中。通过对 applet 的属性的设置，用户还可以控制它的特性和表现。

- 如在 TickerTape applet 中所示，文字还可以滚动显示。
- 如在 Fireworks applet 中所示，图形可以重复一可视的模式。
- 也可以用动态的 applet 来教学，像在 Juggling applet 中演示的一样。
- Magic applet 使用户可以对图形进行修改，可以在一幅已有的画面上再画些东西。
- 一个 Java applet 还可以执行与 HTML 的图像相等的功能。与传统的图像映射相比，Java 的长处在于它的图像映

射可以根据用户光标的所在位置不断地给出反馈信息，直到用户点下鼠标之前，这些反馈都不必有延迟。

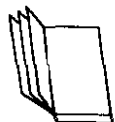
- 已有很多公司为了给 Web 页面增加趣味性而开始使用 Java。这里既有计算机公司，也包括摇滚乐队。

5 给 Web 页面带来交互性的 Java

当通过支持 Java 的浏览器上网时，你首先会注意到的应该是 Java 带来的动态 Web 页面的功能，但这仅仅是些表层的东西。JavaWeb 页面不仅可以容纳各种动态的组件，Java 的 applet 还可以接受由鼠标或键盘传递的用户输入。Java 使得人们可以制作嵌入了纵横字谜的页面，或者是带示波器的、显示化学模型的、包含游戏的，甚至到包含通信系统的页面（参看第 1 章中的图例）。

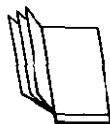
本章中我们概述几个提供交互功能的 Java applet，从简单的游戏 applet 到一些较复杂的教育 applet。因为和本书一样，Java 现在还处于它的发展阶段，Java 技术还未完全成熟，所以本章仅仅是对 Java 可以给网络带来的非常丰富的交互功能的粗略论述。

本章中也指出了在这些 applet 中用到的一些关键编程技术。在本书第五部分中，你可以学到更多的 Java 编程基础以及如何在你的自己的 applets 中使用这些技巧。



何为可交互性 (interactivity)? 在媒体发展的过程中，可交互性已成了个十分热门的话题，很多产品宣布它们能实现交互功能，开发商们也纷纷做出保证，他们要给各种媒体（如电视、CD-ROM、杂志、报纸、以及游戏等等）开发可交互功能。Interactivity 被用得如此广泛以致于它快成了一个像“信息高速公路”这样空洞的提法，没有多少意义，也没有明确的解释。

字典上对于这个词的定义道出了它最主要的思想——交流与相互作用,相互影响:in-ter-ac-tive 形容词 1. 彼此或相互作用的;2. 是或者与一种双向的电子通信系统有关,而这种通信系统(如电话、电缆电视或计算机等)或者可能包括用户的指令(如对信息或商品的要求),或者可能包含用户的响应(如进行一次投票)(摘自联机的 Webster 字典定义)。



从最广泛的意义上说,几乎所有的事都可以被看成是有交互性的。比如烤面包机或者电视机,它们也都是根据用户的指令而工作的。

但是,interactivity 的更高些的层次就涉及到了比仅仅响应用户指令更广泛的内容,这些响应的层次和质量的不同会造成极大的差异。在 Java 出现之前,通过超文本链接,网络可以给用户提供多种选择,然而这种交互功能的层次是相当低的。后来 Gateway (Common Gateway Interface 或称 CGI) 程序设计的出现带来了更好的交互性能,因为它实现了对用户的指示特定的响应。不过,Gateway 的程序不是持续活跃的,因为在执行用户的选择并发送响应以前,它必须等待用户点下某个输入键。

Java 使得对用户连续和即时的反馈成为可能,从而进一步提高了网络交互性能的质量。同时,Java applet 还可以连续的接收和执行用户的输入,Java 还可以对这些直接的输入作出特定的响应。这样,随着 Java 时代的到来,网络变得有非常丰富的交互性能了。

交互的游戏

游戏是程序员人常设计的一种很流行的应用程序。游戏天然地适合于 Java applets 所能实现的读取 (give-and-take) 数据流的形式, 它也很适合于 Java 程序能实现的持续的不知疲倦的循环。本章中叙述的游戏 applet 都是一些最早的 Java applet。即使如此, 它们也展现出了 Java 可以给网络带来的交互功能。

Hang Duke

Sun Microsystems 的 Patrick Chan 开发了一个叫做 Hang-Duke 的 applet; 这个 applet 被做为一个演示程序随 Java 浏览器的发送附带发送。图 5.1 显示一个称为 Duke 的 Java 吉祥物。请注意, 因为用户不可以猜词, 所以 Duke 事先已处于一种被挂起来的阶段 3。

这个 applet 接受用户从键盘上输入的字母, 当输入正确时, 它就把字母显示在单词中相应位置的空格上, 如果错了, 字母就被显示在挂 Duke 的绞架顶部的侧旁, 每错一次, Duke 的一部分就会掉下来。

Hang Duke applet 示范了如何用 Java 轻松地把一个纸和笔的简单游戏转换到网络上去。

Hang Duke 是一个可执行的 applet, 它源代码中一个关键的部分是通过检测键盘按键的方式来接收用户的输入:

```
public void keyDown (int keyPressed)
```

keyPressed 是一个表示字符的整型值参数。可以通过计算把这个代码转换成字符并把它放进一字符串表达式中, 如下所示:

```
(char) keyPressed.
```

Video Poker

Video Poker(可视扑克)是另一个常见的游戏。Java 使你可以

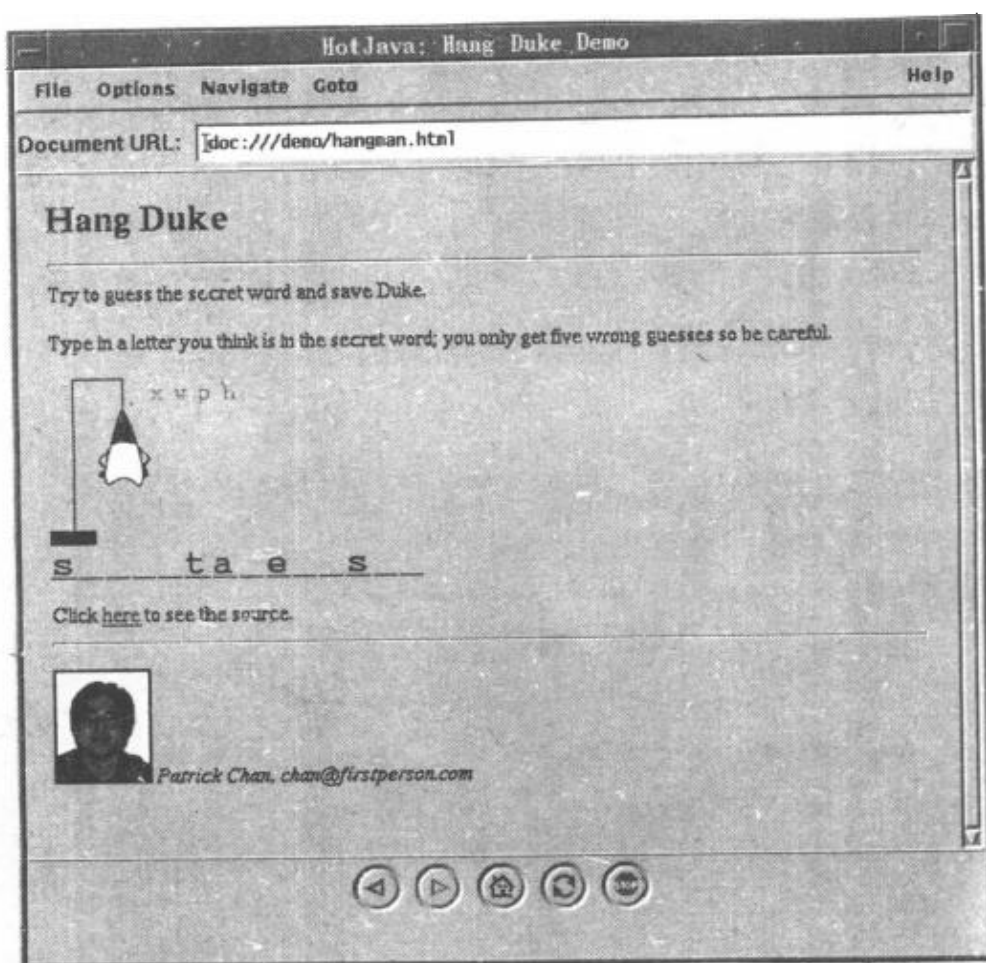
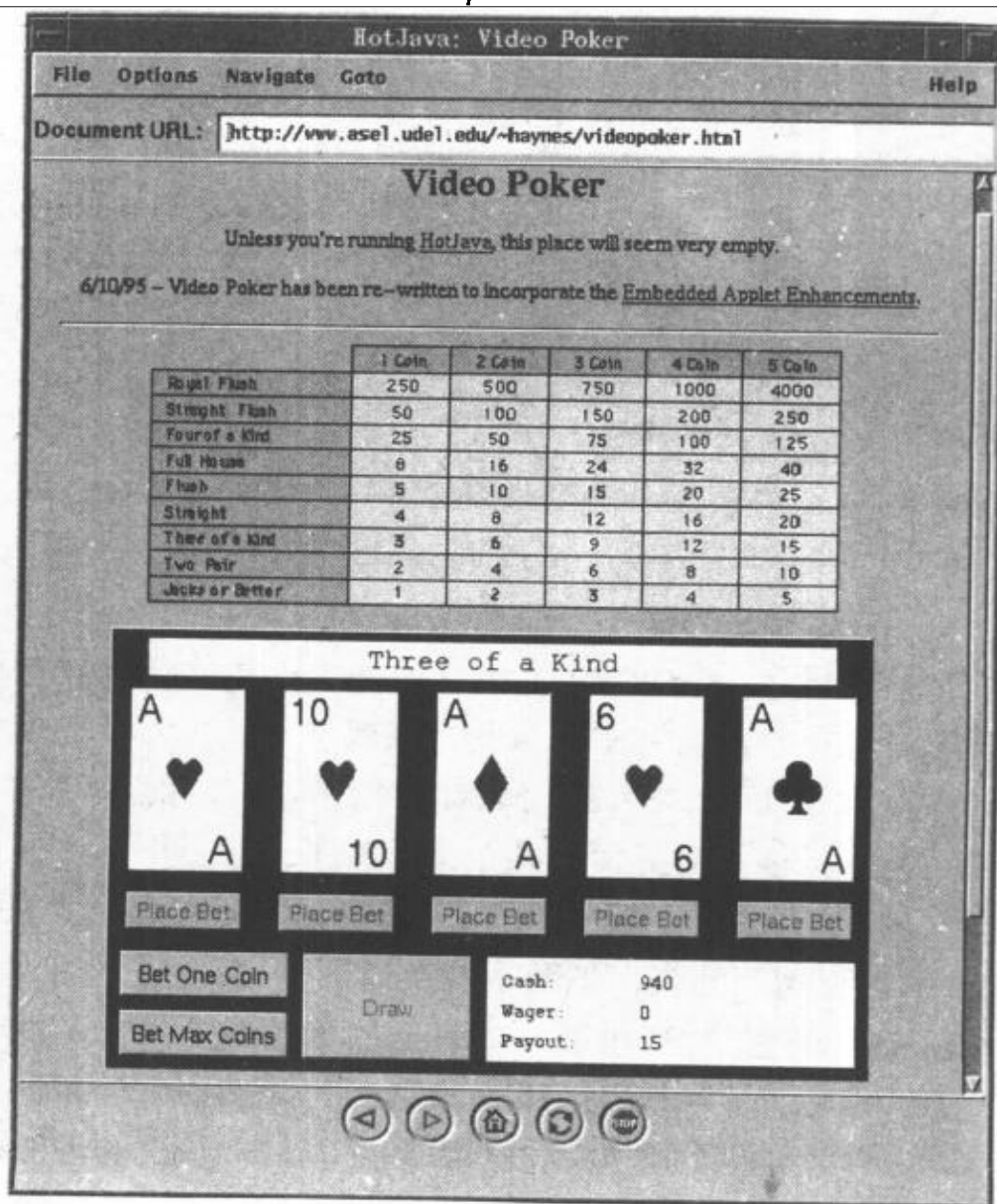


图 5.1 Hang Duke 游戏

(由 Sun Microsystems 的 Patrick Chan 提供)

方便地把它从网上传输过来。Carl Haynes 开发了一个 applet 版本来模拟一种在纸牌赌博中使用的机械或电子的机器。在模拟界面设计中采取了下面的策略：用光标来指示，而鼠标点击来执行。用户通过它们来表示对界面上某个按钮按了一下。图 5.2 显示了这个游戏的界面。用户可以下注，然后抽一手牌，在给出的五张牌中，用户可以保留几张而把剩下的换掉，得到最终的一手牌。在该 applet 页面的顶端，列出了一些可能的牌型组合的种类并给出了它们相应的等级，还在表中规定了与不同牌型等级和不同注码所对应的金额（当然这里是计分而不会是真正的钱）。根据最后所得的这手牌的等级和你所下的赌注，计算机将计算出赌博的结果。

Video Poker applet 用了一组叫做 HotJava Embedded Applet Enhancement 的类，它们是由 Jonathan Payne 开发的，如果



教育应用软件

到目前为止,本章所描述的 Java 游戏 applet 都体现了一种革新,它们采用了 Java 技术使得用户可以与网络进行交互作用。这种可交互性还可以被用于一种更有意义的用途:教育。教育家们数年来一直在用网络进行一些工作,如在网络上提供一些课程,关于他们学校信息的公布,甚至于一些教育模型的实现。但是对于静态的网络文本,由于它相对较差的可交互性,它处理多媒体的能力的限制,使得它虽然在信息传递方面是非常有用的,但却不适合于控制制作一些真正有革新意义的、能引人入胜的应用程序。在这一节里,重点讲述一些早期的 Java 应用程序,这些程序高度显示了 Java 在教育应用方面的潜力。

分形图

分形是一种特殊的几何图形,它的每一个小的组成部分都与整个图形的形状相同(即每个组成部分放大后都是原图)。分形可以这样构造:从一个最基本的形状开始,然后根据全局模型的结构按其基本图案模式一次次细划并改变图形。雪花就是一种分形图像。任何一片雪花它最微小的一个组成部分都会反映了与整个雪花形状完全相同的晶体构造模式。由于分形是如此的难以用语言描述清楚,让我们来看一个更好的选择吧——Java 应用程序。

Sun Microsystems 的 Jim Graham 制作了一个展示如何由简单的基本图素发展到一个分形图形的 applet (地址为 <http://java.sun.com/applets/applets/fractal/index.html>)。图 5.3, 5.4 和 5.5 演示了这个 applets。页面上端的嵌板里展示了一个雪花边缘曲线的呈现过程。从一个简单的峰形折线开始,越来越多的峰形被有规律的加到图中的直线部分直到整个的曲线形成了边缘参差不齐,但却精确的模拟了复杂的雪花图案的构形(经历了从图 5.3 到图 5.4,直到图 5.5 的发展过程)。

同样的，图 5.3 到 5.5 页面底部展示的分形被人们称为 Hilbert 曲线。从一组形成一个 Y 字图案的线条开始，该图形被不断地细化加入细节，直到整个图形看上去就像一幅漂亮的东方地毯——就像一幅准确的迷宫图案。

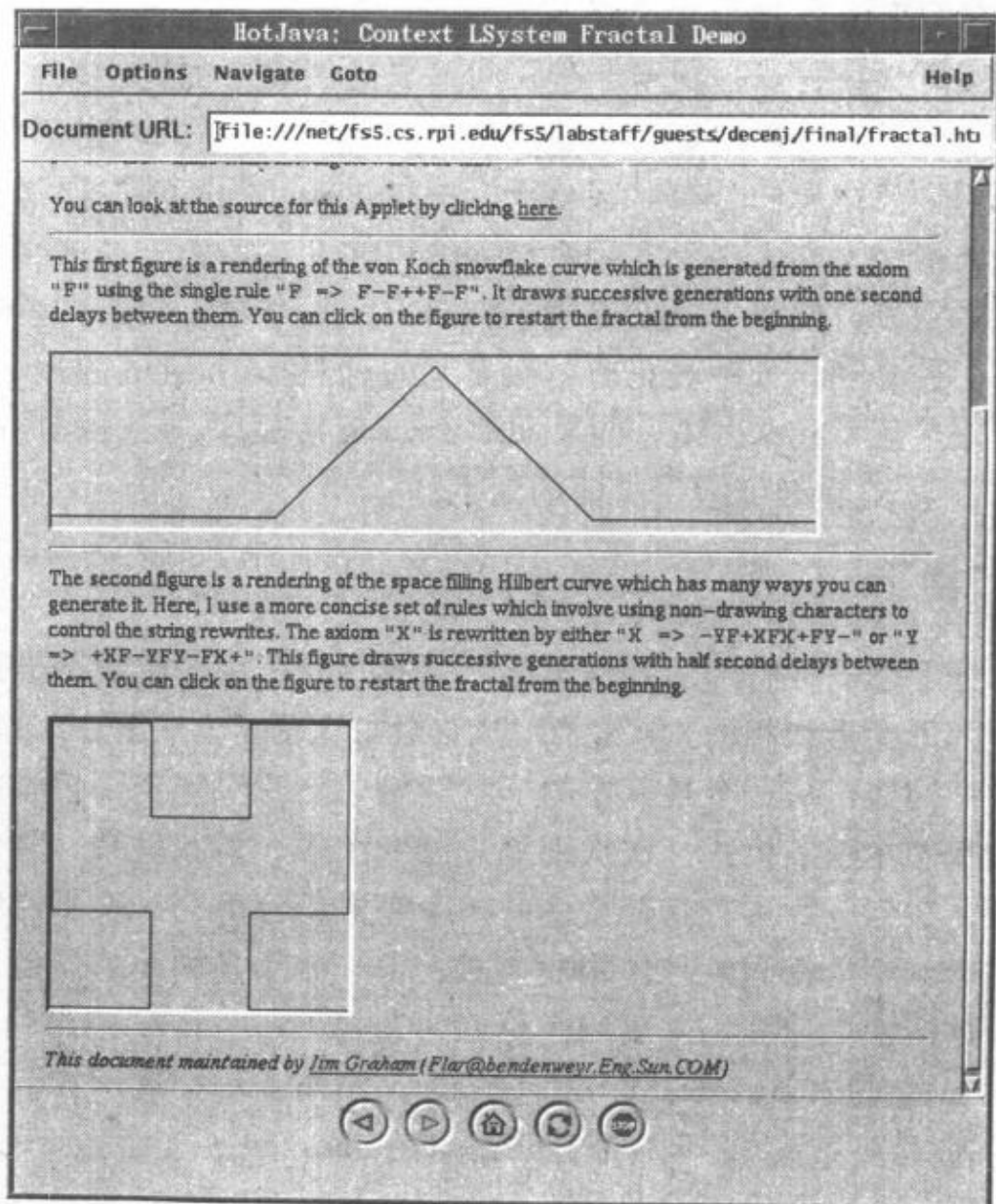


图 5.3 分形课程，初始状态（由 Sun Microsystem 的 Jim Graham 提供）

这些分形从它们的初始状态到结束状态的过程是关键所在：通过将画面嵌入 Web 页面，并从发展过程的图像中截取下画面，制作了一个可展示这些发展过程中画面的科教单元（与本书中这些静止的图例十分相似）。但是，Java 也允许用户回到开始并一遍

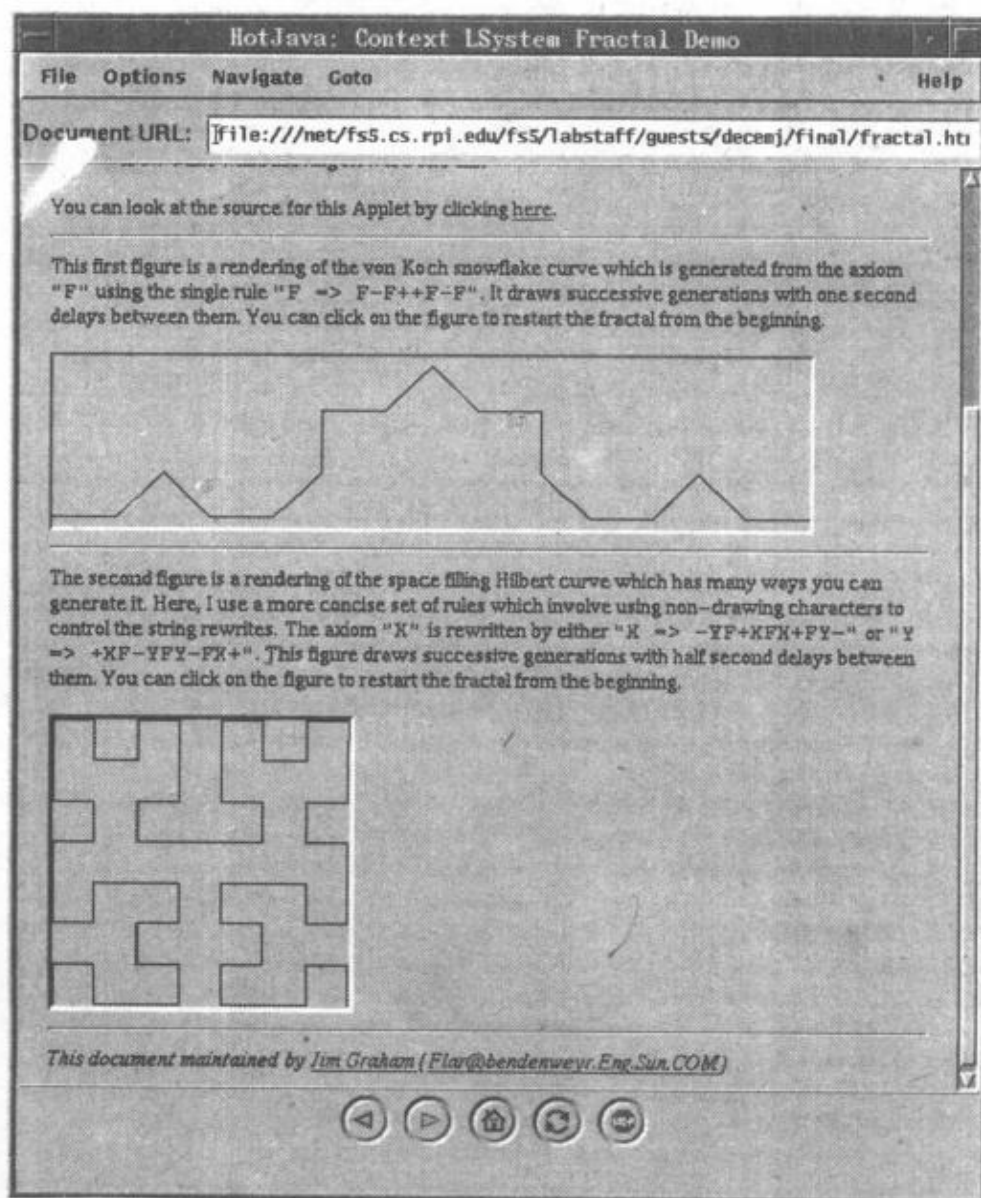


图 5.4 分形课程，中间状态

(由 Sun Microsystems 的 Jim Graham 提供)

遍的观看其过程，以便获得关于算法发展过程的感性认识。用户看到了算法的工作过程而不只是凭空想象它是如何形成的。

单词匹配游戏 (Word Match Game)

在一遍遍地运行以完成对用户的课程教授的过程中, Java applet 所表现出的永不疲倦的功能同样也体现在了单词匹配游戏中。Word Match Game 是由 Sun Microsystems 的 Patrick Chan 开发的(如图 5.6 所示), 这个游戏充分说明 Java 可以完成更为复

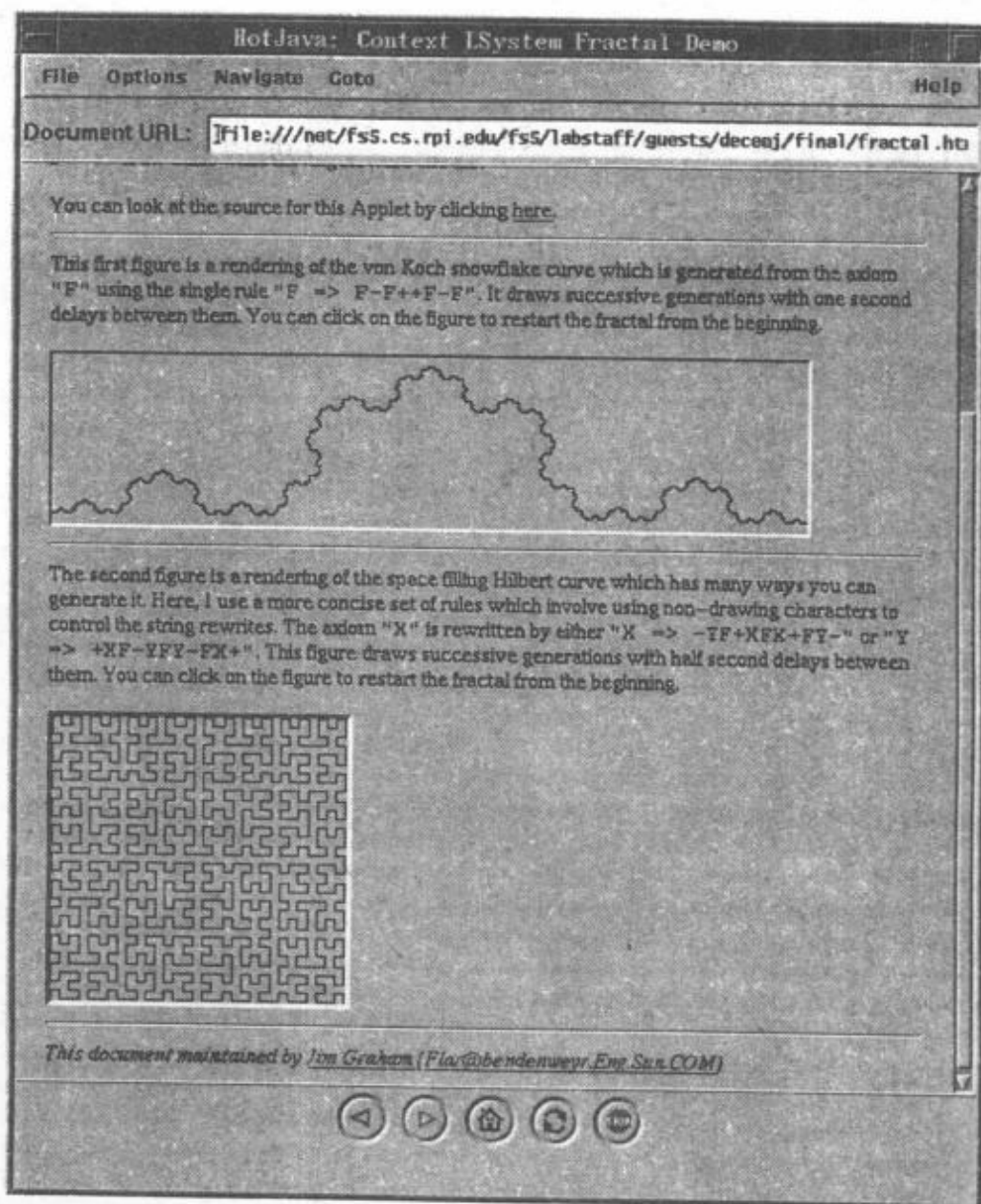


图 5.5 分形课程，结束状态

(由 Sun Microsystems 的 Jim Graham 提供)

杂的工作。在这个游戏中，用户需要对相应的图片给出正确的外语单词匹配。玩游戏时，你先找出你认得的一个物体的图片，在这幅图片上点一下鼠标，然后再在相应的单词上点下鼠标。单词可以用多种不同的语言显示。这样，通过玩游戏就可以扩大你的词汇量。applet 负责从图片到单词画线，当你完成所有的单词匹配以后，请按下 score（计分）按键，applet 将报告出正确匹配的个数，这个游戏还有一个可以使用的扬声器图标，这样你就可以听到在该种语言中单词的发音了。

如在左侧的方框中所示，在这个 applet 中有多种语言可用。在解决了一个问题以后，通过点击语言名，你可以看到该物品在所有语言中的相应的单词。

和分形演示 applet 一样，单词匹配游戏显示出交互的和多媒体的内容给了用户一个在 Web 页面上玩游戏需要的空间，用户不再仅仅是看和选择。当然，这两个课程都不是综合性的家庭教师。相反，它们只是对 Java 可完成功能的一种示例。但就像它们这样的 applet 也已经能实现相当好的教育功能了。

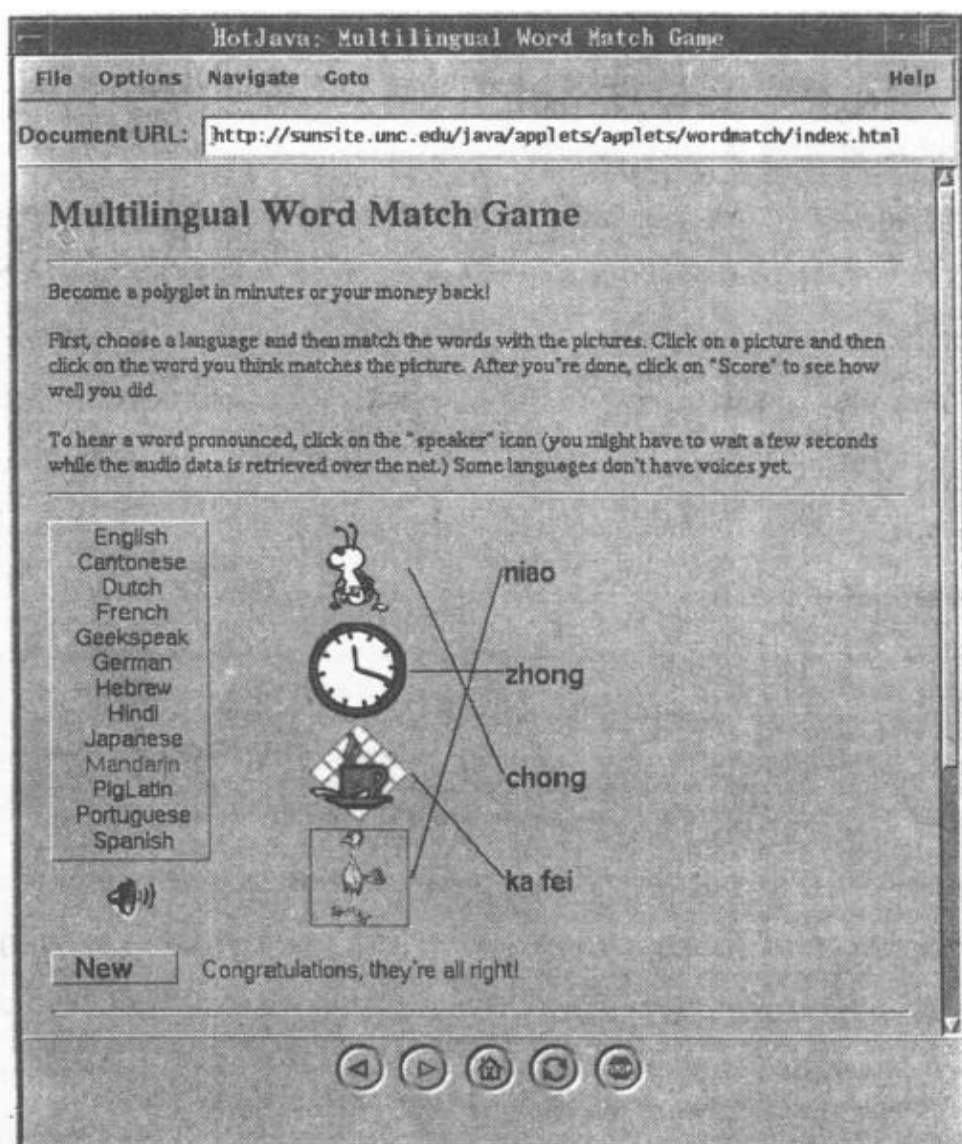


图 5.6 单词游戏课程（由 Sun Microsystems 的 Patrik Chan 提供）

快速傅立叶变换 (Fast Fourier Transform)

在 Java 页面用户的游戏场上,还包括一类执行复杂计算和模拟的 applet。通过这些 applet,用户可以立即得到操作的反馈。关于这种模拟的一个很好的例子是 Fast Fourier Transform (FFT) 课程,由 Gopal Chand David Nicol 和 Calum Sweaton 制作(地址是 <http://www.cee.hw.ac.uk/~calum/java/fft.html>)。傅立叶变换在自然科学和工程中被用于分析数据和信号,FFT 课程(如图 5.7 所示)给用户提供了—条直接观看 Fourier Transform 计算结果的方式。给出一组波形型式的数据点,FFT applet 会计算并显示出结果能量谱。

和本章中列出的其它应用程序一样,FFT 程序也通过提供直接的经验来强化用户的概念。只要进行充分的编程设计(FFT 应用程序使用了 700 多行 Java 代码),一幅 Web 页面可以成为一个为学习提供的高等的科学实验室。如果给予更多的时间和经验,还有更丰富的类库,使用 Java 的教育者可以将他们在网络上的教育工作带进一片新的天地。

Phasor 演示

与苏格兰的 Heriot-Watt 大学的一些研究课题相联系,Yuri Rzhanov(<http://www.cee.hw.ac.uk/~yuri/index.html>)制作了多种用于联机课程信息演示的 applet,他的关于 phasor 的 Java 演示解释说明了电气工程中的一个基本概念。

Rzhanov 的 PhasorApplet 是一个可被用于多种演示用途的通用 applet 的好例子,Yuri 在他的 applet 页面(地址是 <http://www.cee.hw.ac.uk/~yuri:/java/phasor.html>)中解释道,这个 applet 被设计成为“适于讲授者解释一些特殊理论的交互式的图例说明”,而不是为一个学生提供全面研究 phasor 现象的环境。为此目的,这个 PhasorApplet 支持适合讲授者需要的多种属性(见表 5.1)。

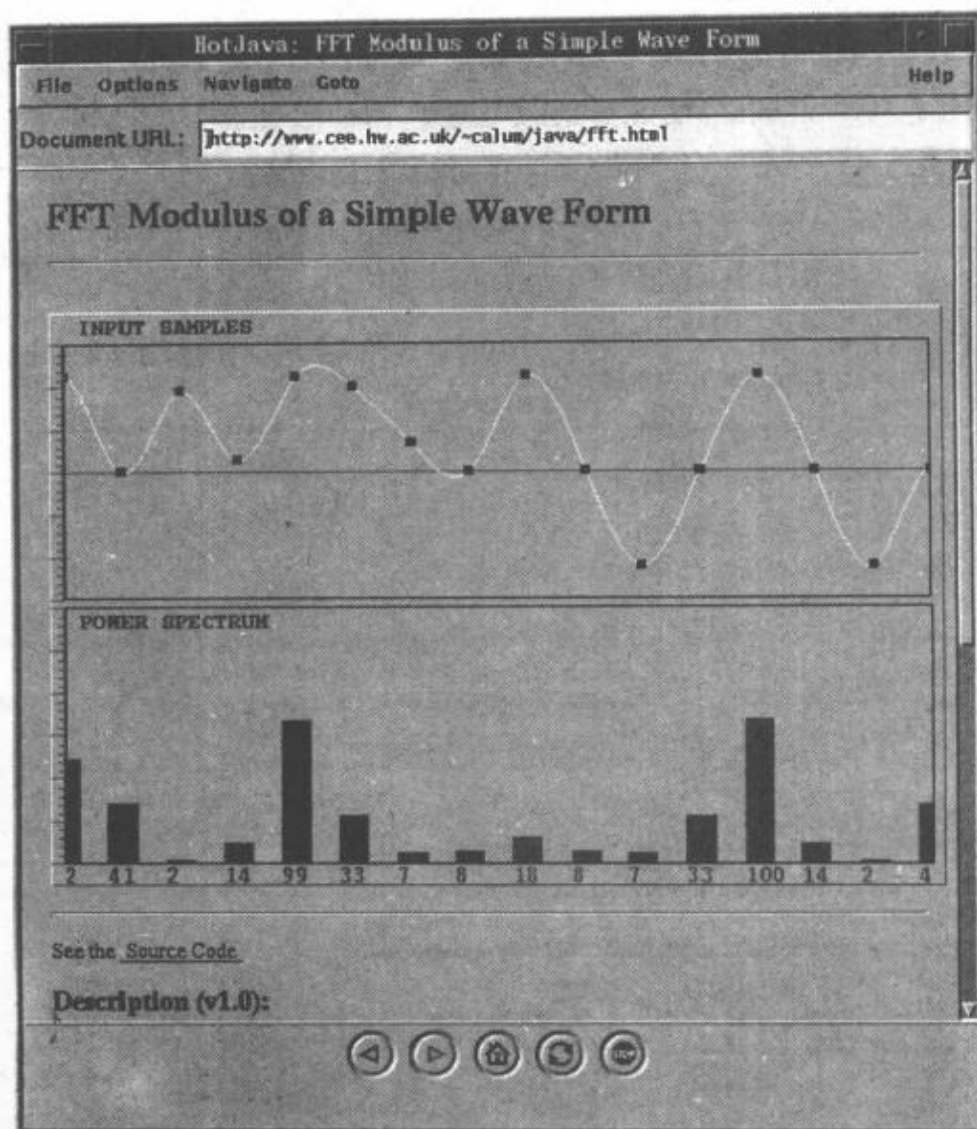


图 5.7 Fast Fourier Transform 课程

(由 Gopal Chand, David Nicol 和 Calum Sweatton 提供)

表 5.1

属 性	解 释
n=int	phasors 的数量
on.top=boolean	Phasor 显示尺寸的总和=显示数字尺寸的布尔值
f0=int	第一个 phasor 的频率
f1=int	第二个 phasor 的频率
p0=int	第一个 phasor 的最初状态
p1=int	第二个 phasor 的初始状态
a0=int	第一个 phasor 的初始质量
a1=int	第二个 phasor 的初始质量

通过包含设置了不同参数组的 applet，一个授课者可以用该 applet 在同一幅页面上创建一组 phasor。

例如，下面是显示 phasor 的 HTML 代码：

```
<P><APP Src="http://www.cee.hw.ac.uk/~yuri/java/"
    Class      ="phasor.PhasorApplet"
    on_top     ="false"
    show_scale ="true"
    n          ="2"
    f0         ="5"
    f1         ="15"
    p0         ="45"
    p1         ="135"
    a0         ="15"
    a1         ="5"
>
<APP Src="http://www.cee.hw.ac.uk/~yuri/java/"
    Class="phasor.StripChartApplet"
    on_top="false"
    show_scale="true"
    n      ="2"
    f0     ="5"
    f1     ="15"
    p0     ="45"
    p1     ="135"
    a0     ="15"
    a1     ="5"
>
```

图 5.8 显示了这些例子中设置的相应显示，

当用户将光标移至 Press to Run (运行) 按钮上并按下鼠标键时，页面左部的 PhasorApplet 中的 phasor 就会以他们被给定的速度进行旋转，而在右边的连环图表 applet (StripChartApplet) 中，就会用图表显示出 phasor 在旋转时，它们的顶点的轨迹。图表演示了一个短的 phasor 是如何以三倍于另一个长些的 phasor 的速度进行旋转的（一个的频率是 15 而另一个的是 5）。

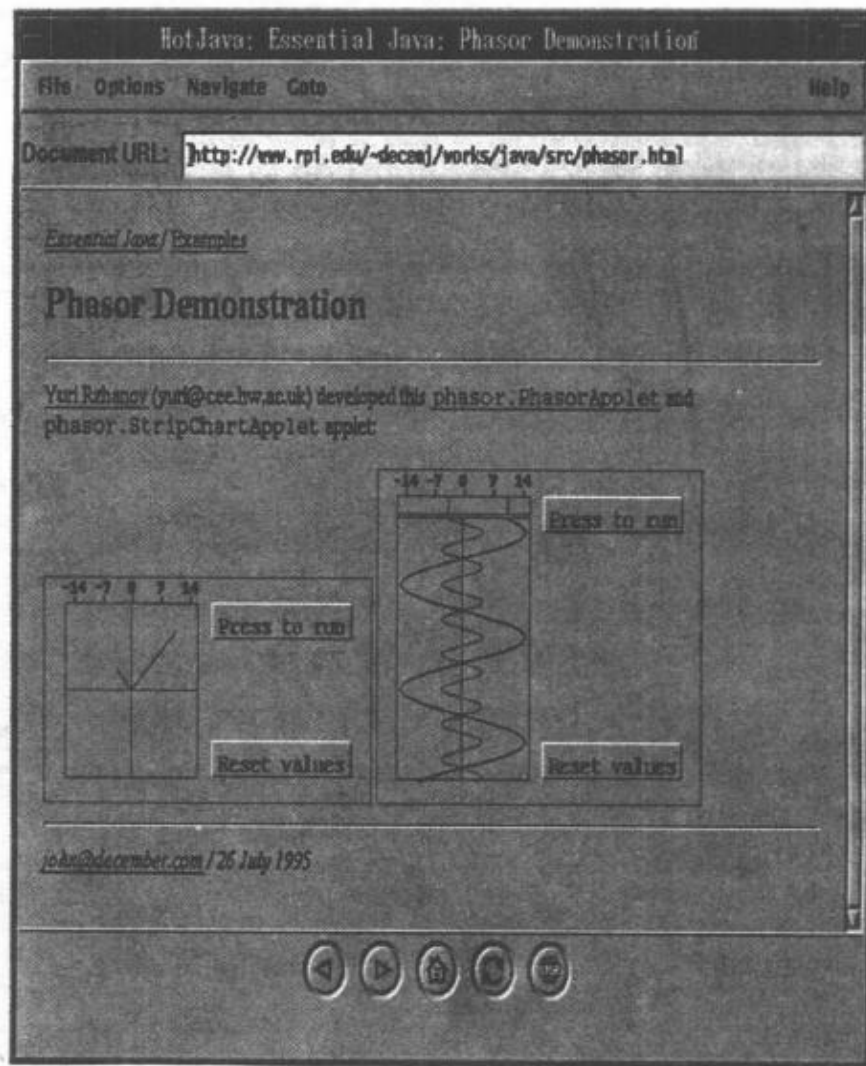


图 5.8 phasor 演示 (由 Yuri Rzhanor 提供)

小 结

Java 不仅给 Web 页面带来了动态画面,它还使得用户可以与这些动画进行交互作用。从实现常见的基于网络的游戏 applet,到新型的可以帮助用户进行学习的 Java 程序,我们可以看出 Java 的可交互性有着相当大的潜力。由于 Java 还处于它的成长阶段,开发更复杂的 Java 程序的关键将会在于一些优质的类库的开发,这些库应该可以帮助 Java 程序员们做到这样一件事:程序员们可以完成一个实现特殊功能的 applet,而不必去考虑对所有的细节进行代码设计。

- 一个简单的 Hang Duke 游戏是为了演示用户怎样可以通过键盘的输入来影响图像显示。
- 可视扑克游戏又给一个 applet 加入了随机的成分和视觉效果模拟的战术。
- 演示分形算法的 applet 可帮助用户获得关于分形算法发展过程的知识。
- 在单词匹配游戏中，通过匹配图片和其相应的单词，用户可以和 applet 进行交互作用，再加上一个发音的扬声器，最终，该 applet 成了一个非常有用的教育工具模型。
- 关于快速傅立叶变换和 phasor applet 的例子说明了 Java 的 Web 页面可以与实验室工作结合起来，这些 applet 可以做为一种科学仪器来使用，或者用于解释并显示输入的数据，帮助学生集中注意力去理解一些科学原理的含义。

发送内容的 Java

由于 Java 是按传递可执行内容的目的而设计的,所以除用户自己本地结点上的那些 applet 以外,其他所有的 applet 都是通过网络传送过来的。然而,做为一种为表达可执行内容而设计的语言,Java 还有比仅仅是发送 applet 的更多的内容。Java 还能实现以下功能:

- 协议处理器:它可以告诉可读 Java 的浏览器怎样管理一种处理信息的新方式。
- 内容处理器:它给可读 Java 的浏览器赋予了解释新型数据格式的能力。
- Java 语言中还有可获取网络资源的语句。这使得 Java 程序可以从用户的可读 Java 浏览器上提取资源。

本章详细讨论了这些功能并给出了一些例子和实现这些功能的关键 Java 语句。本书的第五部分将向你介绍 Java 编程的基本技巧。

网络传送(distribution)和检索(retrieval)的意义

在第 4 章中我们看到了一个允许你在一幅图画上随意涂写的 magic applet (参见图 4.5),在这个简单的 applet 中所涉及到的网络关系有着十分重要的意义。图 6.1 展示了在这个 applet 中用到的发送内容、提取内容和显示内容之间的关系:

这幅图说明网络的提取工作是这样进行的:

- 网络服务器 `www.rpi.edu` 将超文本文件 `marker.html` 送

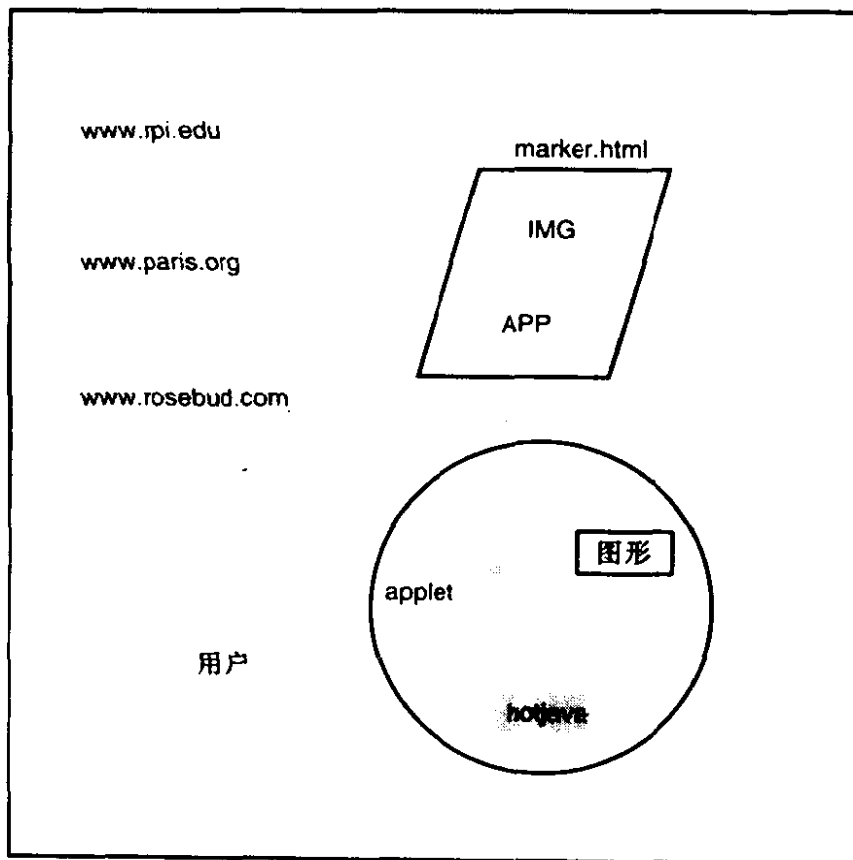


图 6.1 示例 magic applet 中信息传送过程概要图

至用户的 HotJava 浏览器。

- `marker.html` 文件中的 `IMG` 和 `APP` 元素使 HotJava 浏览器到网络服务器 `www.paris.org` 上去提取所要求的图像，而到 `www.rosebud.com` 上去寻找相应的 applet。
- 最终得到的图像、applet 和超文本页面在用户的 HotJava 浏览器上被组装起来。这样，applet 就可在用户的计算机上执行了。

这一网络交互事件涉及了三个网络服务器外加用户自己的计算机，而那些联接 Internet 网络，在几大洲之间转播信息的路由器还不算在其中。这些内容的融合使用展示了 Java 所能帮助实现的一体化功能。

作为网络客户机 (client)，一个可读 Java 的浏览器已经是一个多种协议信息的混合者了。与发送和提取内容的功能相结合，

Java 增加了另一种提取和综合信息的方式。

处理新型协议和格式

可读 Java 的浏览器的一个重要特点就是：它不是固定不可变的（参看第 8 章）。与依靠一套内部设置好的处理信息投递协议的代码和一组固定的辅助应用软件来处理媒体格式不同，可读 Java 的浏览器是以另一种方式建立的。它可以调整自己来学习新型的协议和媒体格式。

这种功能给网络带来了重大的变革：

- 开发者不再被迫依赖于某套固定的协议。
- 可读 Java 的浏览器可以对新的协议作出响应，这些协议或者是可以支持新的应用程序，或者是某种信息传送的新协议。

协议处理器

协议处理器是一种已写好、编译好并被放置在浏览器目录结构中某个特定部分的 Java 类。程序员可以为一种新的内容解释方式而创建一个协议处理器。一个新的协议可以在 URL 的头部加上一个新的关键字。下面的例子向你演示了如何创建一个名为 run 的新协议，它的功能是运行一个 applet。这是根据 Sun Microsystems 在 <http://java.sun.com/progGuide/getStarted/protocol/index.html> 上提供的协议处理器课程编制的。

```
package net. www. protocol. run;
import java. io. * ;
import net. www. html. * ;
/* *
 *
 * Given a URL of the form
 *      run: classname
 * run a compiled java class
```

```
* Currently the class must be in the CLASSPATH.
* /
class Handler extends URLStreamHandler{
    public synchronized InputStream openStream( URL u){
        PipedOutputStream ps=new PipedOutputStream();
        PipedInputStream is=new PipedInputStream (ps);
        PrintStream os=new PrintStream(ps);

        String className=u. file;
        String args;
        className=className. substring(className. lastIndexOf("/") +1);
        if(className. indexOf("?")! =0){
            classname. replace(':', ' ');
        }
        u. setType(URL. content_ html);
        os. println("");
        os. print("");
        os. print("
\n\n");
        os. close();
        return is;
    }
}
```

- 这个例子中的协议处理器是 URLStreamHandler 的一个子类，所以它的说明是：

Class Handler extends URLStreamHandler

而 URLStreamHandler 类是应用程序接口 (API) 的一个组成部分。API 是一组 Java 程序员用来编写应用程序和 applet 的类群。

- 其次，像对任何其它类那样，你需要用 Java 来编辑这段协议处理器代码。然后，把 Handler. class 文件放在你的类目录中的 net/www/protocol 子目录下。如果需要的话，创建 net, www 和 protocol 子目录。
- 你还需要通知浏览器去查看 CLASSPATH 变量来寻找

run 协议类的地址（远程服务器上的类不能运行 run 的这个初步版本）。在 UNIX 系统中，你可以像下面这样设置 CLASSPATH 变量：

```
$ setenv CLASSPATH pathname-of-class-file-directory
```

- 接着，为读入变量的新值，你得重新启动浏览器。
- 现在你的 HotJava 浏览器就可以辨认如下形式的 URL 了：

run: Appletname

这里的 Appletname 是指在 CLASSPATH 环境变量中定义 的类文件目录中的 applet。图 6.2 显示了 run: HelloWorld 的结果。

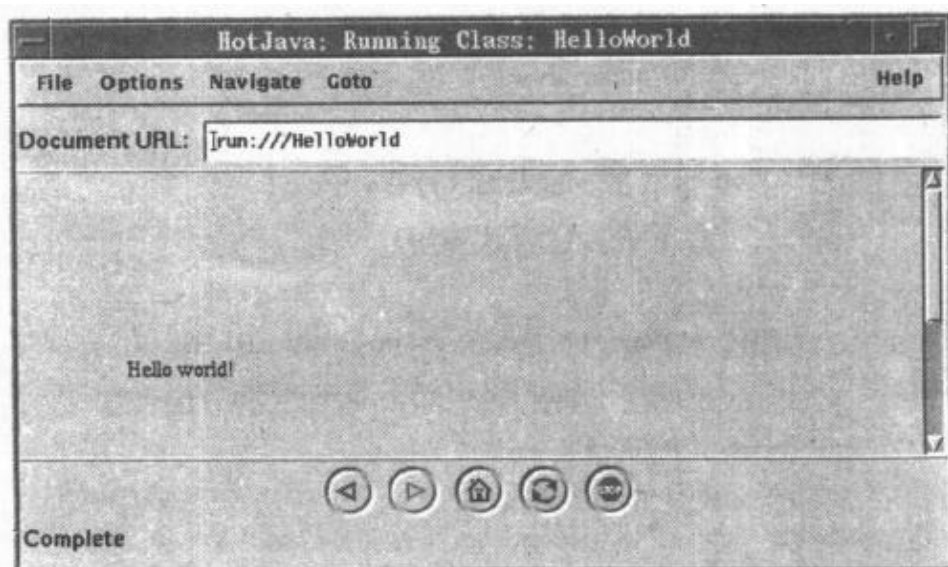


图 6.2 一个协议处理器的简单示例

内容处理器

和协议处理器一样，内容处理器也使得 HotJava 浏览器可以解释新的信息发送的格式。内容处理器把基于 Multipurpose Internet Mail Extensions (MIME) 的文件类型翻译成一种 HotJava 能理解的特殊形式。

Sun 提供了一个很好的关于内容处理目的联机课程，其地址是 <http://java.sun.com/progGuide/getStarted/content/index>。

html。

下面是从示例中摘出的内容处理器类的代码。你可以在一个名为 plain.java 的文件中写入并编译它。

```
/*
 * Plain text file handler
 */
package net.www.content.text;
import net.www.html.ContentHandler;
import net.www.html.URL
import java.io.InputStream;
import awt.GifImage;
awt.GifImage
public class plain extends ContentHandler {
    public Object getContent (InputStream is , URL u) {
        StringBuffer sb=new StringBuffer ();
        int c;
        while ( (c=is.read ()) > =0) {
            sb.appendChar ( (char) c);
        }
        sb.append (" \n\n— and may I add that
                    you look lovely today. \n");
        is.close ();
        return sb.toString ();
    }
}
```

这段代码是十分简单的演示程序，它重复了文件 plain/text 要浏览器显示的内容并在其末尾加了一段话：——and may I add that you look lovely today（我可以加一句吗？你今天看起来很可爱）。

为文件类型 plain/text 创建一个新的内容处理器包含以下几个要点：

●在 plain.java 的源代码中包含把它定义为包的语句：

```
package net.www.content.text;
```

package net.

- 内容处理器解释的文件类型必须是 MIME 的子类型（参看附录 B 中的 MIME 类型表）。
- 你需要在类群目录中与 MIME 类型相对应的一个特殊的子目录 net/www/content/text 中存放和编译 plain.java。
- 为了查找该内容处理器，还应通知浏览器到 CLASSPATH 变量中去找类目录的位置。
- 你可以配置一个名叫 .mailmap 的文件来处理特殊的 MIME 类。如果已在 .mailmap 中加入了一行来处理 plain/text 类型，你需要撤消该行以使 HotJava 可以管理该类（此时，将由 HotJava 浏览器而不是能在 .mailmap 文件中原来定义的那样由辅助应用软件来管理该类）。如果你还没有建立 mailmap 文件，则可以跳过这一步。
- 然后，你就可以用这个内容处理器来解释任一台机器上的任何带 .text 扩展名的文件了。例如在图 6.3 中，展示了 HotJava 对文件 <http://www.rpi.edu/~decemj/words/java/src/test.text> 的解释。

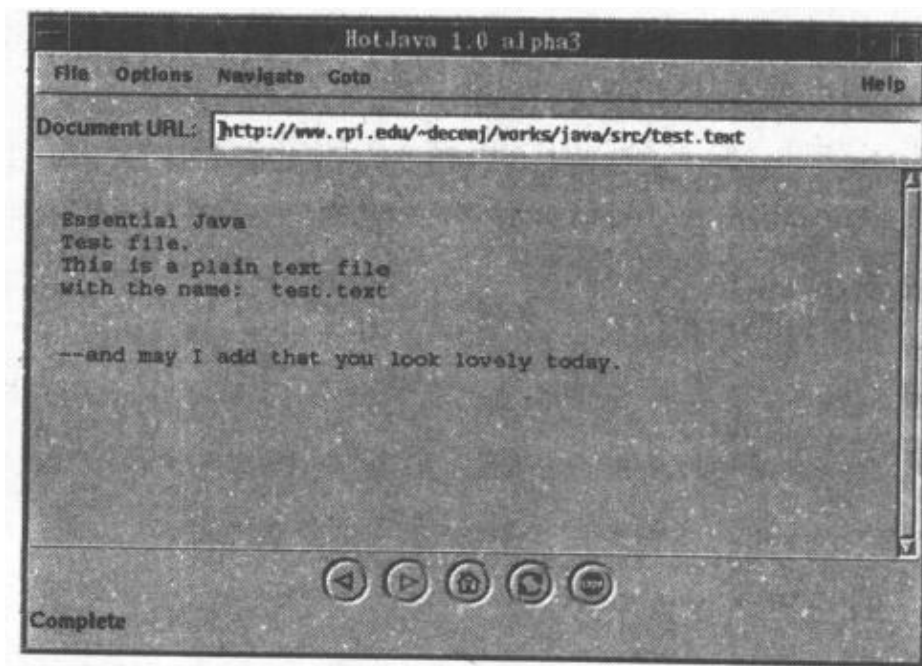


图 6.3 内容处理器的一个简单示例

从网络上检索信息

协议处理器和内容处理器可以扩展支持 Java 的浏览器所能理解和显示的内容,使网络对于各种各样的信息发送方式都是开放的。Java 还有丰富浏览器从网上提取信息途径的功能,给了 Java 程序员一个制作从网上提取信息的 applet 的机会。

网络冲浪 (Surf-o-matic)

Patrick Chan 在提到他的 Surf-o-matic 演示程序(地址: <http://java.sun.com/applets/applets/autopilot/index.html>)时说:“看它就和看电视一样!” Surf-o-matic 是种可以“接管”用户浏览器并自动改变页面的网络旅行导游 (Web tour guide)。这一类的 guide 在广告、教育和其它一些方面的应用上都有很大的潜力。

Surf-o-matic 是一个用了一组类群来完成 Surf-o-matic 控制面板功能的 Java applet。该控制面板允许你根据一些“随机”列出的 URL 列表巡游网络。首先,请选择一个或一些你想要游历的 URL 列表,按下控制面板上的 Go 按钮,然后, Surf-o-matic 就会接管你的浏览器。Surf-o-matic 以每 20 秒或者某个可由你自己设定的时间间隔更换一次页面,从你所选的随机列表中取一幅新的页面放上去。Yahoo (<http://www.yahoo.com>) 的网络页面随机列表是很丰富的。图 6.4 中便是以 Yahoo 做一示例。

图 6.4 演示了正在提取一幅新页面的 Surf-o-matic。即使只是从目前的这种初步的形式,也可以看出 Surf-o-matic 是一个十分有用的独立的应用软件。它的功能之一,就是相当于个展示 Web 内容的报摊。

Surf-o-matic 是根据 Matthew Gray 的 Web Autopilot (网络自动导航器) 发展而来的。该导航器运用了 Netscape 浏览器的用户机推动 (client-pull) 特性。而 HotJava 浏览器可以用 Java 代码

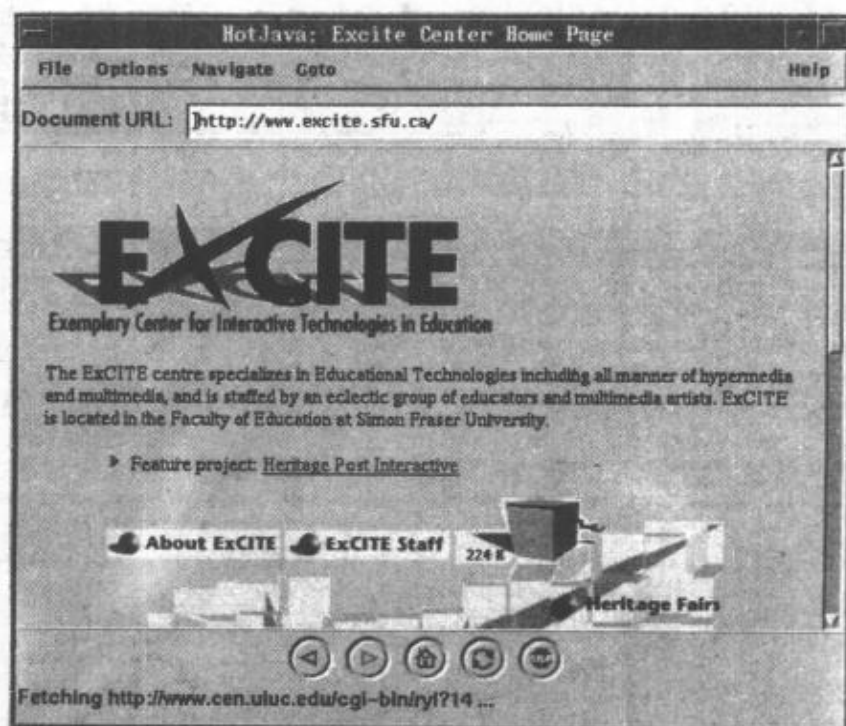
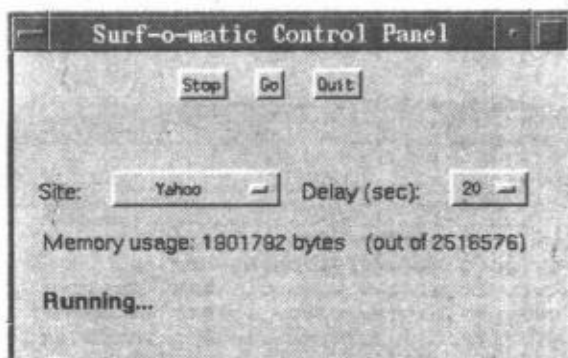


图 6.4 运行中的 Surf-o-matic

(由 Sun Microsystems 的 Patrick Chan 提供)

来编制实现同样功能的程序。那样，它就不必再转换到 client-pull 技术了。Patrick Chan 的 Surf-o-matic 是一个展示 Java 在网络信息提取方面的这种灵活性的小例子。

VPRO 导游 (VPRO Tour Guide)

Netherland (荷兰) 的一个公共广播组织 VPRO (<http://www.vpro.nl>) 制作了另一个本质上与 Surf-o-matic 十分相似的 Tour Guide (<http://www.vpro.nl/htbin/scan/www/interaktief/java/control.html>)。图 6.5 展示了一个正在工作中的 VPRO

导游。

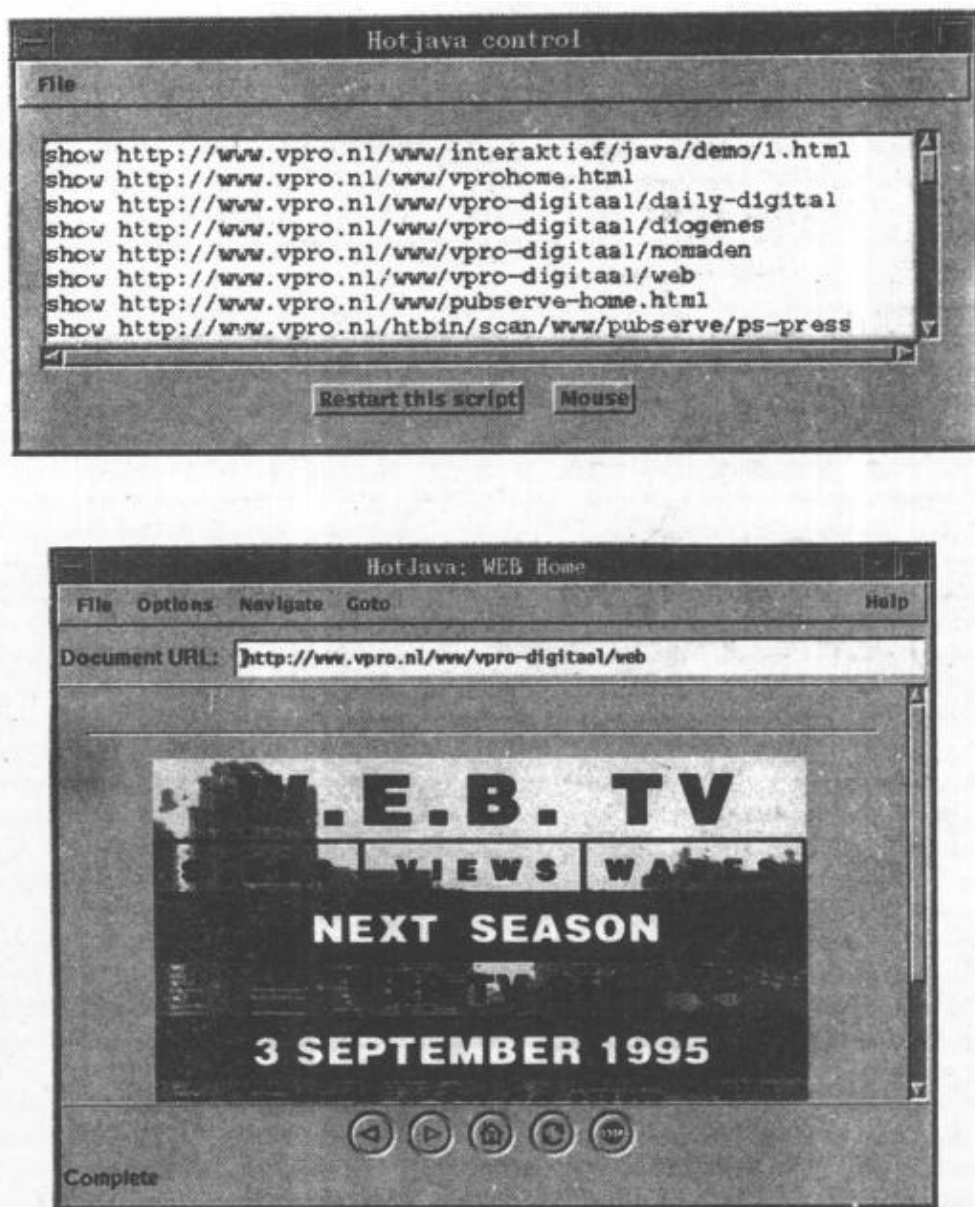


图 6.5 由 VPRO 导游的旅程

(由 Netherlands VPRO 的 Daniel Ockeloen 提供)

这个导游可带领你游历 VPRO 结点上的网络页面。与 Surf-o-matic 不同的是，这种旅行根据一组经过精心选择的 URL 进行，而不是随机的。URL 的列表在图 6.5 中的 HotJava 控制面板上列出。

Surf-o-matic 和 VPRO Tour Guide 都在其结点上公布了他们的程序代码。这样，程序员就可以知道应该如何进行网络提取。

- 协议处理器是以类的型式写的，编译过后放在与其协议类型相应的目录中。
- 内容处理器也是一个类，被特别的写出以指明 HotJava 浏览器怎样处理 MIME 的数据类型。
- Java 实现了在 applet 内提取网络信息，使得一个 applet 可以接管用户的显示并展示一组网络资源。

第三部分 使用 HotJava 浏览器

Java 语言提供了在 Web 上进行广泛交流的许多机会。然而，Java 语言还需要特殊的 Web 浏览器来利用它。

Java 以前的浏览器不能显示 Java 内容，所以必须有一个支持 Java 的浏览器来利用 Java 程序。要是浏览器不是支持 Java 的，它将忽略指明 applet 的 APP 标记，并且不显示 Java 内容。HotJava 被创造来充分利用 Java 的能力，对非商业用途，HotJava 可以免费得到。

开始 HotJava 仅支持几个平台，现在正被移植到更多的平台上。预期更多的商业浏览器会在将来的版本集成 Java 技术。

这个部分的各章详尽地解释了 HotJava 浏览器，展示了如何寻找最新版本、下载、解包和使用。第 7 章描述了写这本书时可以得到的版本，并且给出了寻找其它支持 Java 的浏览器的途径。第 8 章详细解释了在操作和设计方面 HotJava 与过去几代浏览器的不同之处。第 9 章回顾了从用户的观点看，HotJava 如何与以前的 Web 浏览器功能相似。

这几章将给出：

- 对 HotJava 的解释及查找、下载、解包最新的版本
- 理解 HotJava 如何是一个具有内建的学习更多的协议和数据格式的浏览器。它不必依靠固化的内置协议
- 如何使用 HotJava 界面漫游 Web 的知识

HotJava 是针对 Java 的浏览器

正如本书前两部分讲过的, Java 是一种被设计用来通过网络发布可执行内容的程序语言。如果想观看 Java 可执行内容或与之交互, 必须可以解释用 Java 语言的文件建立的字节码的支持 Java 的浏览器 (第 2 章揭示了 Java 语言的技术设计及 Java 源文件与字节码的联系)。

本章描述了获得、下载和安装 Sun Microsystems 的 HotJava 浏览器的必要条件。本章还指出了关于其它支持 Java 的浏览器最新信息的联机资源。

Java/HotJava 的获得方式

HotJava 是来自 Sun Microsystems 的能在 Web 上显示交互内容的浏览器, 因为 HotJava 完全是用 Java 写成的, 它展现了 Java 程序语言的能力。

HotJava 可以直接从 Sun Microsystems 节点 <http://java.sun.com/> 获得, 在这个节点的 HotJava 许可证部分寻找最新版本。

预期许多商业浏览器和联机服务将集成显示 Java 交互内容的能力。如想得到最新的信息, 请阅读 Sun 的 Java 页面或者附录 A 中给出的更多 Web 参考信息的列表, 你还可以利用 Web 检索技术和资源来寻找支持 Java applet 的浏览器的最新信息。

HotJava 的目标

HotJava 是一个 Web 浏览器，但它也能解释可执行内容。通常的过程是：

1. 开发者在自己的计算机上书写 Java 源码。

2. 开发者用 Java 编译器编译源码。

3. 编译的结果是字节码结构无关文件。

4. 如果你的计算机可以运行支持 Java 的浏览器，你可以下载并运行这些字节码。

正如第 2 章描述的那样，Java 的结构无关形态是它的一个主要特点。这个特点意味着支持 Java 的浏览器必须能够根据用户计算机的体系结构解释字节码（它们是体系结构无关的）。

移植的目标

“平台”指硬件、操作系统、用户界面的类型。程序员必须使软件“本地化”使得它能在特定的平台上运行，因为每个平台有不同的特点，而软件不得不适应这些特点。例如，对运行 Windows 的个人计算机而言，Microsoft Word 是可以运行的。发明 Word 的程序员也为 Macintosh 计算机写了一个版本。使软件“本地化”以便能在各种平台运行的过程叫移植（porting）。像许多其它浏览器一样，HotJava 浏览器必须移植到许多平台上。例如 Netscape 浏览器有 X Window 系统的版本、Windows 的版本和 Macintosh 的版本。

图 7.1 显示了一般情况：信息提供者向使用不同类型计算机的用户发布字节码，Java 移植后的代码建立了一个“socket”适应用户的平台，原先的字节码能“适合”移植的 socket，允许传输并非针对任何平台的字节码。另外，信息提供者不必担心用户平台的特点。因此，Java 使之变为可能的可执行内容是体系结构无关的。那些专门为不同平台而写的支持 Java 的浏览器的各种版本提供了这种可能性。

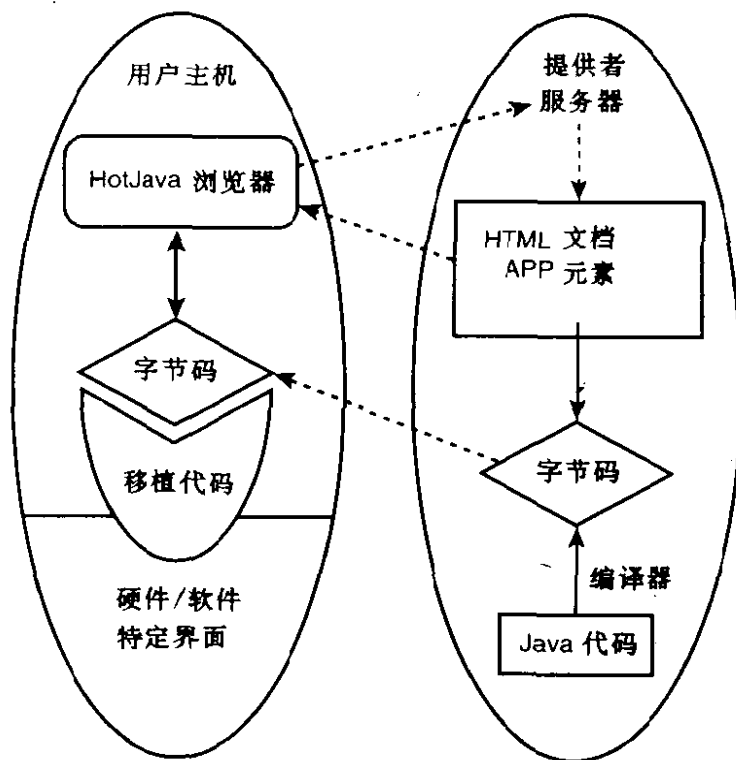


图 7.1 字节码如何“插入”HotJava 浏览器已完成的 Java 移植

已完成的 Java 移植

1995 年夏天，Sun Microsystems 对 HotJava alpha 版的开发完成的有：

●Sun Solaris 2.3, 2.4 和 2.5 基于 SPARC 的机器

Solaris 是 Sun Microsystems 针对网络应用程序的软件平台。Solaris 包括一个操作系统——SunOS。要知道关于 Solaris 的更多信息，请看 <http://www.sun.com/cgi-bin/show?sunsoft/solaris/index.body>。

SPARC 代表 Scalable Process ARChitecture。SPARC 是 Sun Microsystems 和其它公司开发的一种技术。它基于计算机精简指令集 (RISC)，使用 RISC 技术的微处理器被设计成能够有效处理小指令集。要知道更多信息，请看 <http://www.sparc.com/>。

●运行 Windows NT 3.5 的 Intel x86 机器

“Intel x86”意味着使用 Intel “86” 系列微处理器的计算机。

这包括 Intel 286, 386, 486 或 Pentium 处理器。要知道更多的信息, 请看 <http://www.intel.com/>。

Windows NT 是针对 Internet 的 Microsoft 操作系统。要知道更多的信息, 请看 <http://www.microsoft.com/>。

正在进行的移植

Sun Microsystems 为在下列平台移植 HotJava 的开发正在进行 (1995 年夏):

●Microsoft Windows 95

Windows 95 是 Microsoft 下一代个人计算机用户界面软件, 1995 年 8 月发布。更多的信息参看 <http://www.microsoft.com/>。

●Macintosh 操作系统 7.5 (参看 <http://www.apple.com/>)

1995 年夏 Sun Microsystems 之外的开发者正努力在下面这些平台上移植 HotJava:

●Amiga 计算机 (参看 <http://www.yahoo.com/Computers/PCs/Amiga/>)

●NeXT 计算机 (参看 <http://www.next.com/>)

●Linux 操作系统

Linux 是一个类 UNIX 操作系统, 由 Linux Torvald 在网上其它一些人的协助下写成的。它在基于 386/486 的 PC 机上运行。更多的信息, 请看 http://www.yahoo.com/Computers/Operating_Systems/Unix/Linux/。

●Netscape Communications 计划把 Java 集成到自己的浏览器里。最新的信息请看 Netscape 节点 <http://home.netscape.com/>。

联机 Java 安装信息 Sun Microsystems 的 Java 节点是查找有关获取 Java 的最新联机信息的最佳地方。特别地：

<http://java.sun.com/installation.html> 给出下载 HotJava 的现有节点和程序

提示

<http://java.sun.com/source.html> 给出 Java 最新版本信息

<http://java.sun.com/porting.html> 描述最新移植成果

<http://java.sun.com/bugsandfeatures.html> 检查问题和需求特性的最新情况

下载和使用 HotJava

下载和安装过程根据你的平台不同而不同。这一部分以 UNIX 为例说明下载、安装和运行 HotJava 的一般过程。

Java/HotJava 源代码和执行程序都是可以得到的。二进制版包括用来安装 Java/HotJava 的代码，通过向 Sun 特别请求（见对话框内 URL 给出的 Sun Java 节点）。只有那些想知道 Java 到底如何工作或者有兴趣在其它平台上写一个 HotJava 的移植版本的人需要得到源代码。HotJava 的源代码是 Java 语言本身写的，也就是说 HotJava 是 Java 语言的应用程序。

第一步：找出是否有合适于你的平台的移植版本

可执行版本是 Java/HotJava 的典型用户应当获得的。可执行版本包括一个准备在用户平台运行的可执行文件。第一步是找出是否有针对你的平台的 Java 移植版本。在 URL <http://java.sun.com/porting.html> 里确定这些。要是你的平台已经有一个版本，还要知道如何去获得它。通常它们都是通过 FTP 节点或 Web

节点发布。为了下一步下载，你必须知道节点的 URL。

要是你有一个 UNIX 平台，你可以用 `uname` 命令发现它的值。这一命令显示当前系统的名字和其它信息。检查信息的主要部分是这一命令可选项。`-s` 选项显示操作系统的名字，`-r` 选项显示操作系统的版本，`-p` 是显示处理器类型（`-a` 选项显示所有可能的信息）。下面这个例子显示了如何识别用户的平台：

```
$ /usr/bin/uname -srp
```

```
SunOS 5.4 sparc
```

回应不错，因为 Java 已经移植到基于 SPARC 的机器。要是处理器名字不显示，浏览器在这个系统上将不能运行。

第二步：下载

你知道可以得到 Java 的 URL 之后，下一步就是下载。下面这个例子显示了 UNIX 和 PC 平台上下载 HotJava 的典型过程。

非常重要的是要遵守 Java/HotJava 源码的知识产权保护。源码不是公开的，需要再发布的许可证。然而，写 applet 和发布它们不再需要许可证。要得到 HotJava 的源码和执行代码合法过程，可以看 Sun Microsystems（或许可人）提供在源码或执行本身的内容。

Sun Microsystems 为下载 Java/HotJava 提供了一个主节点和一些镜像节点（有发行代码复本的节点）。这些节点提供匿名 FTP 访问，这些节点包括：

- Java.sun.com 的匿名 FTP（Sun Microsystems 管理的主 Java/HotJava FTP 服务器。
- 世界范围的 SunSITE（Sun Information and Technology Exchange）匿名 FTP `sunsite.unc.edu`，路径 `/pub/sun-info/hotjava/`。
- SunSITE 由 Sun Microsystems 在各地的大学建设，想要得到 SunSITE 的列表（那里有 Java/HotJava 的发行版），查看 `http://www.sun.com/sunsite/`。

UNIX 上的 HotJava alpha 版需要图形界面 OpenWindows 版本 3.0 或更高。UNIX 工作站通常安装一个 X Window 系统的变种,要得到更详尽的资料,请看 <http://www.x.org/> 的 X Consortium。

下一个问题是确保能够下载的磁盘空间。未压缩的 Java/HotJava 的 alpha 版需要大约 10MB,所以这是整个需要的磁盘空间的最小值。联机下载介绍将告诉你机器需要多少空间。

要是你没有一个 Web 浏览器,你可以通过匿名 FTP 获得 HotJava。下面的例子显示了一个 UNIX 例程,其中用户输入用黑体表示,这个例子告诉你如何在发行目录中找到最新版本和版本号。也可以用 mget 命令快速给出下载文件名:

```
$ ftp java.sun.com
Connected to java.sun.com.
220 java FTP server (Version wu-2.4 (3) Wed Feb 15 15:57:20 PST 1995)
ready.
Name (java.sun.com : john) : anonymous
331 Guest login ok, send your complete e-mail address as password.
Password : enter your email address here
230-Welcome to the Java/HotJava ftp server.
230-
230-Information about your login and any transfers you do are
230-logged on this host. If you disagree with this policy
230-please disconnect now.
230-
230-Information about this service may be obtained by browsing
230-
230- http://java.sun.com/
230-
230-Please contact csw@sun.com with any technical problems you have
230-with this server.
230-
230-You are user 116 out of a maximum of 150 users.
230-It is Mon Jul 10 17:39:48 1995 in Palo Alto, Ca.
```

```
230-
230-Guest login ok, access restrictions apply.
ftp>cd pub
250-STOP! STOP! STOP! STOP! STOP! STOP! STOP! STOP! STOP! STOP!
250-=====
[informational message about the latest version compatibility]
ftp>binary
200 Type set to I.
ftp>ls
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
java
images
.message
.history.java
starwave
OLD
.orig
hotjava-alpha3-solaris2-sparc.tar.Z
sb.nt.exe
unz512xN.exe
hotjava-alpha2-nt-x86.exe
226 Transfer complete.
150 bytes received in 0.0068 seconds (22 Kbytes/s)
ftp>mget hotjava*
mget hotjava-alpha2-nt-x86.exe? no
mget hotjava-alpha3-solaris2-sparc.tar.Z? yes
200 PORT command successful.
150 Opening BINARY mode data connection for hotjava-alpha3-solaris2-
sparc.tar.Z
(5476531 bytes) .
ftp>quit
```

注意在这个过程中，只需对适合的平台回答“yes”。传输结束后，就可以开始下一步：解包。

第三步：解包

一旦下载后，执行文件用以下的 UNIX 命令解压缩和解包：

```
zcat hotjava-alpha3-solaris2-sparc.tar.Z|tar xf-
```

这样会在当前目录下建立一个叫 hotjava/ 的子目录。这些做完之后，以 .tar.Z 结尾的原始文件不再需要，因此可以删除它们来节省磁盘空间。

第四步：看看你得到了什么

HotJava 解包后，在 hotjava 子目录有一系列文件和目录，它们包括：

README 描述这个版本和它的最新信息的一般信息文件。

bin HotJava 浏览器和相关 Java 语言工具的执行代码目录。

classes Java 发行版中 class 的字节代码（以 .class 结尾的文件）的目录，（对 α 版）包括：

awt 处理窗口和图形特性的类库。

browser 构成 HotJava 的类（记住，HotJava 是用 Java 写成的应用程序）。

java 构成支持 Java 语言的类，如输入输出、实体和工具。

net 支持网络通信的类，包括诸如 news, mail, FTP 等协议和 Web 相关协议 HTTP。

classsrc 在 Java 发行版类的 Java 语言源代码目录（以 .java 结尾的文件），注意这不是以前所说的 Java “源”（它是用来安装 Java 语言的），只是 Java 发行版类的源码，包括 HotJava 浏览器的 Java 源代码。

copyright.html 阅读这个文件可以知道关于 Java 发行版的知识产权保护和合法使用的正式声明。

demo 一系列 Java 演示程序。

doc 用 HTML 写的支持 Java 的完整文档集，包括可以通过 HotJava 阅读的联机文档。

include Java 编程的头文件。

index.html 发行版中提供的用户信息索引；它是 HotJava 浏览器开始的缺省页面。

lib 这些是 Java 编程中使用的库文件。

readme.html 介绍信息的 HTML 版本。

第五步：运行 HotJava

bin 目录包括 HotJava 浏览器和发布的其他工具（包括 Java 语言解释器和编译器），你可以进入这个目录，输入命令 `hotjava&` 来运行 HotJava（程序名之后的 `&` “隐藏”了进程，这样你可以在运行浏览器的同时在 UNIX 提示符下输入更多命令）。下面是进入解包后的目录运行 HotJava 的一个例子：

```
$ cd hotjava
$ ls
README      classsrc    doc         lib
bin          copyright.html include      readme.html
classes      demo        index.html
$ cd bin
$ ls
hotjava      javac       javah       javap-g
java         javac-g     javah-g     javaprof
java-g       javadoc     javap       sun4
$ hotjava &
```

和 HotJava 一起的其它软件

如果你下载了 HotJava 的浏览器，你已经得到开发 Java 应用程序必须的软件工具。这些在第 2 章里作了综述，有些还详尽地作了演示，并且在那些与 Java 发行版一起的帮助页面作了描述

(在 hotjava/doc/man/目录)。本书的第四部分将告诉你如何使用这些工具建立 Java applet 和应用程序，这些工具包括：

- Java 编译器 javac，它把可读 Java 源码翻译成机器可读的字节码
- Java 解释器 java，它运行 Java 程序
- C 头文件和源文件产生器 javah, java_g，它产生用来创建函数的 C 头文件和源文件
- Java 反编译器 javap，它打印出关于类文件的信息
- 文档产生器 javadoc，它可以使给定源文件产生 HTML 文件
- 预定义 tooljavaprof，它使 Java 解释器 java 用-prof 选项建立的性能数据格式化

HotJava for Windows NT 3.5

HotJava 在运行 Windows NT 3.5 的 Intel x86 的 α 版于 1995 年夏季公布。Windows 95 版尚未完成，但 Sun Microsystems 正在开发它。

对 Windows 发行版来说，通过 FTP 下载 Java/HotJava 与前面 UNIX 平台一样，除了用户在问及相应文件时回答“yes”。

```
$ ftp java.sun.com
Connected to java.sun.com.
Name (java.sun.com: john): anonymous
Password: 在此输入 e-mail 地址
ftp>binary
ftp>mget hotjava *
mget hotjava-alpha2-nt-x86.exe? yes
ftp>quit
```

下载结束后，你可以在操作系统提示符下打入执行文件名把浏览器解包：

```
c:\hotjava-xxxx-nt-x86.exe
```

这个命令运行自展开程序把文件和目录解包。

与 UNIX 下的 HotJava 类似，可以在操作系统提示符下打入执行文件名运行 DOS 下的 HotJava：

```
c:\hotjava\bin\hotjava
```

注意，你需要使用 Windows 95 下的程序管理器把 HotJava 当成新程序项安装。

小 结

本章告诉读者获得非商业用途的 Java/HotJava 浏览器的基本步骤。要是读者想得到 HotJava，第一步是了解对你的系统是否有一个已完成的版本；第二步是下载执行文件。要是你已经有了一个 Web 浏览器，你可以通过 HotJava 发行页面的链接来下载；第三步，把下载的文件解包；对 UNIX，你可以使用 `zcat` 命令，对 DOS，下载的二进制文件是一个自展开文档。最后，在操作系统提示符下输入执行文件名（`hotjava`）运行 HotJava。

- 非商业用途的 Java/HotJava 发行版可以在 `ftp://java.sun.com` 或 `http://java.sun.com` 找到并下载。
- 下载、解包、运行 HotJava 需要关于用户平台的操作系统和文件结构的某些知识。

HotJava 是新型的 Web 浏览器

Java 程序语言标志了 Web 上交互性的显著进步，而显示 Java 的 HotJava 则标志了新一代“智能”Web 浏览器的开端。HotJava 不只是包含一些功能的集合，它还能动态地适应和学习新的协议和文件格式。使用 Java 的 Web 信息开发者不再局限于文本、图形和对前 Java 时代的浏览器而言使用很有限的多媒体。所不同的是，HotJava 浏览器为新的协议和新的媒体格式提供了可能性，这些东西以前 Web 上从未有过。

本章给出了一个简要的总结，HotJava 和过去有过的 Web 浏览器显著和不可思议地不同。下一章详尽地描述了 HotJava 的操作，也提及了使用 HotJava 在 Web 上漫游的秘诀。（第 7 章描述了如何获得、下载、安装 HotJava。）

HotJava 标志了浏览器的一个新时代

在 World Wide Web 过去五年的发展历程中，新的 Web 技术常常改变人们关于 Web 和联机通信能力的一般看法。当 Mosaic 浏览器 1993 年发布时，由于它带给 Web 的图形化无缝特性，激发了普通大众对 Web 的关注。Mosaic 不是用不同的工具访问各种信息，而是动态地虚拟地将 Internet 信息联为一体。它的“一点即得”的操作改变了关于 Web 浏览器能力的看法。继之而来的 Netscape，似乎已经成为时尚，并且继续推翻 Web 展示信息的各种障碍。

前 Java 浏览器

最早的 Web 浏览器是 CERN 的行模式浏览器。后来的 Mosaic 类浏览器（1993—1995 年中期的 Mosaic 和 Netscape）很快提供了 Web 的图形界面。然而，Mosaic 类浏览器的工作原理像是基于 Internet 信息的信息过滤器。这些浏览器内置了基本 Internet 协议和媒体格式的知识（如 HTTP，NNTP，Gopher，FTP，HTML 和 GIF）。浏览器把在网络上发现的信息与关于协议和媒体格式的知识匹配，然后显示结果。图 8.1 解释了当浏览器在网络上发现资料并根据协议和常见媒体格式的内部编码解释时的操作。这些浏览器也使用另外的程序帮助显示特殊媒体格式，如动画和语音。

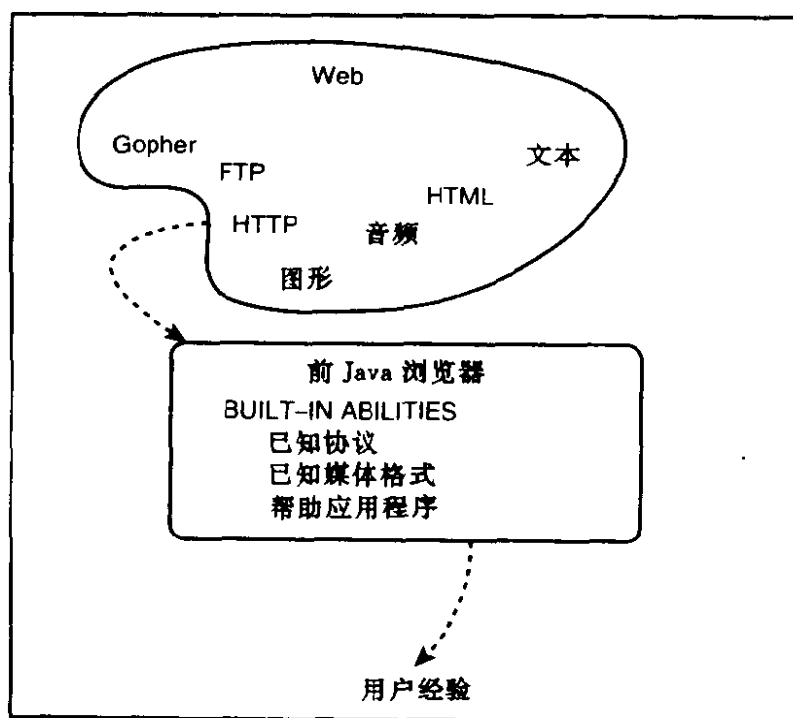


图 8.1 前 Java 浏览器像是过滤器

前 Java 浏览器对网络的常见协议和媒体格式有足够的知识（当然很“笨”）。不幸的是，前 Java 浏览器不能处理未被编程的协议或没有可用的辅助应用程序的媒体格式。这是 Java 时代的浏览器面临的问题。

Java 时代的浏览器

Java 时代的浏览器是“轻型的”，因为它并未内置协议或在它的内部处理媒体格式。不同的是，HotJava 浏览器的主要功能是学习和解释任何协议和媒体格式。当然，作为发行版的一部分，HotJava 浏览器已经告诉如何处理最常见的协议和类型。此外，对任何 Java 程序员可能设计的新类型或协议，HotJava 都能学习它。

如图 8.2 所示，Java 时代的浏览器是“轻型”的，这不是因为预先收集的 Web 知识，而是来自于所有能力中最重要的——学习能力。

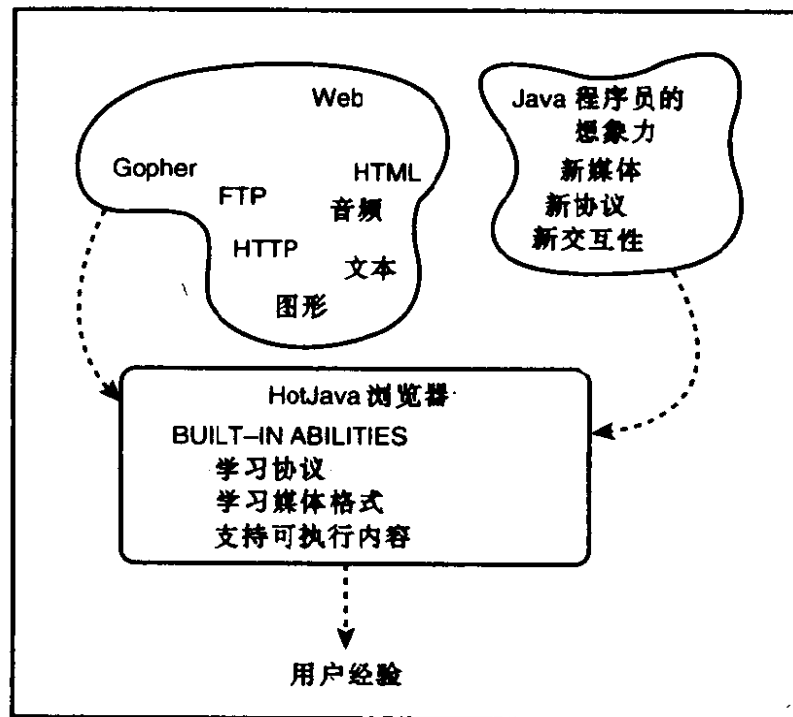


图 8.2 Java 时代的浏览器能够学习

HotJava 特性

除了被设计成与以前的浏览器不同，HotJava 特别地被设计来表现 Java 语言的能力。因为 HotJava 浏览器本身就是用 Java 写成的一个应用程序，毫无疑问它继承了 Java 语言的所有特性：

可执行内容、网络通信和信息检索。HotJava 也包含了图形 Web 浏览器本身应当具有的所有标准特性。

可执行内容

如果你是 HotJava 的一个新用户，你或许会立刻注意到它的一个卓越特性——它能够展示可执行内容。这一点在本书第二部分已经讲过。当 HotJava 浏览器下载一个含有 applet（以 HTML 的 APP 标记来标识）的页面时，那个 applet 的字节码被下载到你的计算机上运行，并在浏览器上显示。在 applet 初次下载之后，applet 可以无限地运行（例如，第 1 章或第 4 章的动画）而不必再通过网络下载。

Java 语言也支持“多线程”applet。这意味着在简单的 Web 页面上同时有几个活动发生。这使得用户有很多事情参与或观看成为可能。

网络内容

HotJava 也被设计用来通过网络发布内容。当然也包括对用户计算机安全的考虑。HotJava 识别称之为 HOTJAVA_WRITE_PATH 的环境变量，这个变量设置了 applet 可写的文件目录。

在 UNIX 下，可以把这个变量设置成文件列表或以冒号分隔的目录。例如：

```
$ setenv HOTJAVA_WRITE_PATH /tmp/:/user/smith/data/:/devices/:  
~/.hotjava
```

这条命令允许 applet 只能在 /tmp, /user/smith/data/, /devices/ 和 ~/.hotjava 目录中写文件。最后这个路径在用户目录下（~/.hotjava）。这种可写设置也可以 HotJava 主菜单的 Options/Properties 选项中进行。

HotJava 支持网络通信的所有主要的 Internet 信息协议：HTTP, news, mailto, FTP 和 Gopher。访问 Usenet 时，

NNTPSERVER 变量可以设置成本地 NNTP 服务器的名字。

HotJava 的浏览器特性

HotJava 拥有许多标准浏览器的接口特性。作为用户，你可以控制一些 HotJava 特性，包括：

- 图形和 applet 装入 你可以关闭图形和 applet 的自动装入。
- URL 请求 用户可以采用不同的方式通过 URL 向信息源发出请求。例如，用主菜单的 File/Open 选项或直接在浏览器顶部 Document URL 对话框中输入 URL。

你还可以在 HotJava 里获得一些有益的信息管理特性，包括：

- 联机文档 HotJava 浏览器在主菜单的 HELP 选项里提供了文档接口。
- URL “热表”管理 这里包括一个历史列表、“热表”、一类特殊的快速访问列表，称之为“goto”列表。

信息开发者可以使用一些 Netscape 扩展。HotJava 版识别 `<CENTER>...</CENTER>` 使文本居中，它也可以识别标尺线 (`<HR>`) 属性：

- Size = “number” 线的粗细（像素数）。图 8.3 中的例 1 是 15 个像素，并占据了整个显示宽度。
- Noshade 取消阴影形成静态线条。图 8.3 的例 2 是 1 个像素静态线条，占据整个显示宽度。
- Width = “number|percent” 线的宽度，或者以像素计算的宽度（数字，例 3 是 100 像素宽），或者是相对显示宽度的百分比（不是页面宽度）。例 4 是显示宽度的 33%。这些线缺省居中（缺省值可以由 Align 属性改变）。
- Align = “left|right|center” 小于整个页面宽度的标尺线的定位。例 5 是 10% 显示宽度的 5 像素线条，居右排列。

HotJava 的 alpha 版不识别颜色背景、字体大小改变、BLINK 元素或表格。下一章将讲述如何使用这些特性。

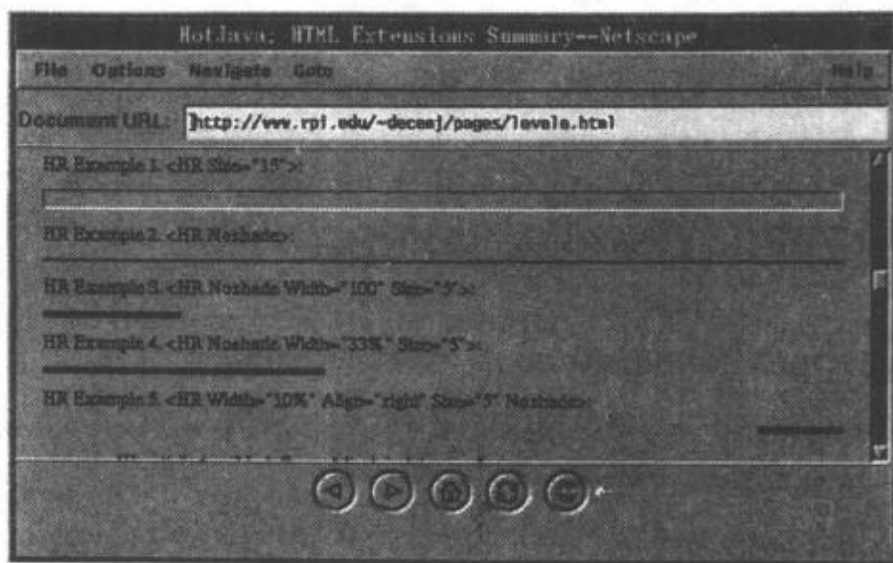


图 8.3 HotJava 对关于 HR 元素的属性的 Netscape 扩展的解释

小 结

本章描述了 HotJava 浏览器与在其之前发明的 Web 浏览器的不同,以及某些情况下的相同之处。HotJava 独一无二的特性是能够学习新的协议和媒体格式以及显示可执行内容。

- HotJava 浏览器给 Web 带来的主要的崭新特性。
- 由于它能够通过网络下载可执行内容, HotJava 能够解释新的协议和媒体格式。
- HotJava 具有网络安全和网络通信特性。
- HotJava 具有典型图形 Web 浏览器的许多标准控制和显示特性。

用 HotJava 漫游 Web

前两章叙述了怎样获得并下载 HotJava 软件以及 HotJava 和其它 Web 浏览器的区别, HotJava 和其它 Web 浏览器在漫游 Web 时的图形界面很相似。本章向用户介绍 HotJava 的界面特别是 HotJava 提供的一些功能性支持界面。

注意

这里叙述的是以 HotJava 浏览器的 alpha 版本为基础, 这个版本将会继续发展, 而且其它采用 Java 技术的浏览器也可能会提供有点不同的界面, 但是, 所有的版本都将提供类似的支持和界面, 与本章描叙的完全一样。

本章首先介绍 HotJava 的用户界面, 以显示它是怎样支持 Web 漫游器的需要, 然后介绍一些在 Web 上查询信息的一般方法, 你可以使用这些方法和技术在 Web 上查询 Java 和其它技术的最新信息。

HotJava 用户界面

图 9.1 是 HotJava 的索引页的图示, HotJava 的用户界面完整地显示了漫游 Web 的全部功能。

图 9.1 显示的 HotJava 的索引页是 HotJava 在 X Window 系统下的 alpha 版本, 该图的组成部分描述如下:

- 浏览器的最顶端显示的是该页面的名字 (此时为 HotJava: welcome to HotJava)。

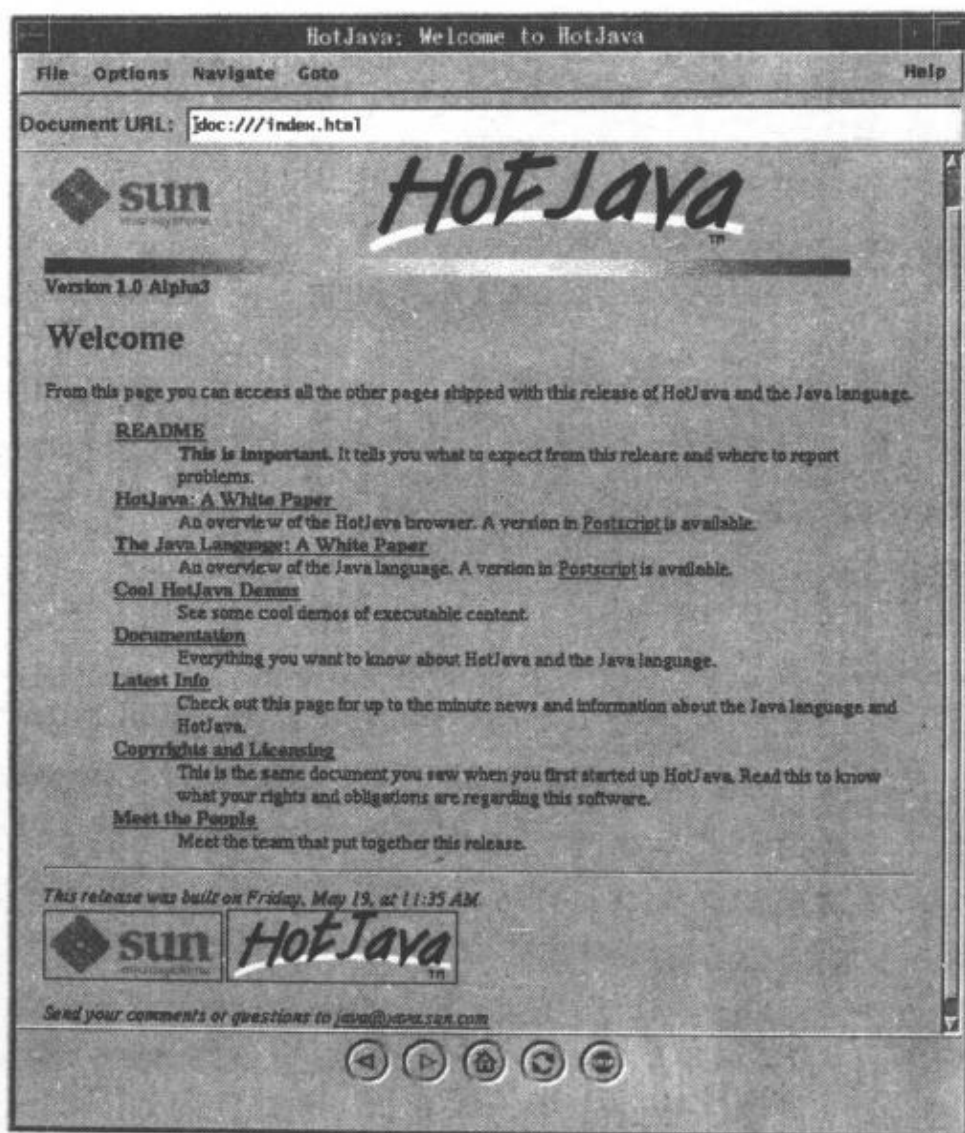


图 9.1 HotJava 的索引页

- 页名字下面的是有菜单选项的标题栏，它包括 File，Option，Navigate 和 Goto 四个菜单命令。
- 文档 URL 窗口，该窗口显示 HotJava 当前显示的信息的 Web 地址。图 9.1 中，这个文档的 URL 地址是：doc:///index.html。URL 中前面的 doc，是显示 Java 文档实际使用的协议。
- 显示区域或浏览器窗口，它位于 URL 窗口下方，该区域显示该页的内容。
- 在浏览器下方是一些命令按钮，它提供一些通用的功能：backward，forward，home，reload 和 stop loading。

HotJava 如何帮助你漫游 Web

当漫游 Web 时,你可能有许多需要或要求,这些需要和要求可以看成一种分级结构。最基本的需求,如信息显示、链接激活和移动等是任一个 Web 浏览器都应提供的核心功能。其它一些高层的需求,则要求您最好有一些现代图形设备。这一节介绍 HotJava 是怎样支持 Web 浏览器的这些分级需求,我们的讨论从最基本的需求开始直至最高层。

1. 信息显示。把一个 Web 文档、网络信息文件和网络信息服务以一种可视、可听或其它可感知的形式展示给你,这是一个 Web 浏览器的最基本功能,因为这种显示是信息传输和查看的基础。

2. 链接激活。激活一个超文本链接以便显示这个链接点的相关信息,这是使得 Web 浏览器能在超文本文档中选取更多资源的关键。

3. 移动。从以前访问的资源中选择一条链接或者建立到某个特定资源的链接,这种能力是能否充分利用 Web 的关键,而且这种能力能使 Web 浏览器在以后的链接路径中更加灵活,更加方便。

4. 信息控制。是指信息获取和显示的能力。这是能否使你有选择地停止或限制资源获取的关键。

5. 交互性。是指你能传输指定的信息(除了刚激活的链接所显示的信息)给 Web 服务器或信息提供者。

6. 选项和反馈。这能使你更有效地使用 Web 浏览器和使它更好地满足你的个人爱好。

7. Web 实用化:它能使一个用户非常熟练地使用 Web 浏览器,而且能很快地看到非常广泛的网络信息。

下面就分节详细介绍 HotJava 是怎样支持这些需求的。

信息显示

HotJava 提供的图形界面,能使你看到文本和图形,标志热线链接,标志任意指定的浏览器符号。

HotJava 中的文本采用了一种已相当于图形 Web 浏览器标准的模式,该模式采用缺省字体来显示 HTML 文档的文本信息,用指定的字体来显示头和 HTML 文档中用格式化标记标志过的字符(可参阅附录 C 中的 HTML 标记)。联机图像和文本放在一起显示。HotJava 还支持超文本页的滚动显示,这就突破了在浏览器中显示一个屏幕内容的限制。

浏览器中显示的文本和图形中的一些地方被链接到其它资源,就像 Web 页和多媒体文件一样。这些链接在 Java 中是采用高亮文本和把文本加上下划线的方式来显示,按照以前的图形 Web 浏览器的传统,HotJava 用蓝色来显示这些链接,当你选中某个链接后,它就会变成另一种蓝色。

HotJava 中的一些特殊符号能为你提供信息显示的条件。这些符号能表示当前正调入的图形和文本的状态。

- 当图形或文本中出现一个灰色的盒子,这意味着正在进行调入的处理,正在下载的图形里面将会有有一个小的肖像图标。
- 如果出现的是黄色的盒子,而且有一个肖像图标在里面,而不是一幅图像,是因为你关掉了“关掉图像自动调入”的选项,你可以更改这个选项来获取图像。
- 如果出现黄色盒子,盒中有一个肖像图标,但在左上角是蓝色的。肖像图标的出现是因为图像被连接到其它资源上而且你关掉了“关掉图像自动调入”的选项。你可以单击蓝色的左上角来选择这条链路,或者单击盒子的黄色部分恢复图像。
- 如果出现的是有齿轮形图标的黄盒子,意味着 applet 还没有调入,因为你关掉“关掉自动调入 applet”选项,你可以

单击黄色的盒子来调入 applet。

- 如果出现带有肖像图标的红盒子,则意味着在调入图形时出现错误。
- 如果出现带有齿轮图标的红盒子,则表示在调入 applet 时出错。
- 如果出现带有肖像图标的红盒子,而且左上角是蓝色的话,则表示在调入一个被链接到其它资源的图像时出错。

链接激活

在 HotJava 中可以通过单击热点来激活一个链接,热点可以是 Web 页中的任意一部分,它被链接到其它 Web 资源,热点可能是用蓝色高亮显示的超文本单词,也可能是图形映像 (image map)。图形映像可以把一幅联机图像的某些像素链接到其它资源。热点也可以是链接到其它资源的一幅图形。总之,当你选取一个热点后,文本颜色将变成激活链接的颜色(缺省情况下,HotJava 为红色),而且由链接标识的资源将被获取并在 HotJava 中显示。

在 HotJava 中图形和 applet 可以单独下载,这意味着,在 HotJava 显示的一页中,可以把图形和 applet 显示成灰盒子(这表示它们正在被下载),这就使得当你下载页中的图形和 applet 时增加了许多看其它部分的机会。

移 动

HotJava 支持“后一页”“前一页”和“Home”功能,在 HotJava 底部的命令按钮使你能使用这些功能,这些功能也可以在菜单栏中的“Navigate”选项中实现。

HotJava 也允许你打开一个任意的 URL 地址。你可以执行菜单中的 File/Open 命令,在出现的对话框中输入 URL 地址即可;或者你可以直接在 URL 窗口中删掉当前显示的 URL 地址,再输入你想要链接的 URL 地址。这种技术对于打开一个与已经显示

过的资源类似的资源非常有用；你可以在 URL 窗口中编辑文本然后按回车。

信息控制

当信息正通过网络被下载时，HotJava 在浏览器的底部会显示状态消息，下面是一种典型的状态消息序列：

```
Fetching... URL of the resource
Formatting...
[1 connection: ××××× bytes remaining (××% completed)]
Complete
```

这种状态消息是非常有用的，因为这可以帮你确定在浏览器操作过程中出现的错误可能在哪儿，状态消息告诉你正有多少字节没有下载完，通过观察这个数字的变化，你可以估计下载这个资源大约需要多长时间。如果你觉得这太长了，HotJava 提供“Stop”按钮（Stop 命令按钮在浏览器底部，参见图 9.1），单击“Stop”命令按钮，你可以放弃下载。如果一页中的文本已经下载，但因图形只下载了一部分，它将继续显示在浏览器中。前面介绍的专门符号可以帮助确认还有哪些图未被下载。

自动下载图形和 applet 的功能可以通过执行菜单命令 Options/Properties 来关掉。用户还可以标记是否延迟图形调入和 applet 调入。

交互性

HotJava 被设计成可以通过 applet 的显示来提供高度的交互性，正如第 2 章和第 5 章描述的那样。

HotJava 还通过 mailto 支持个人通信交互性。HotJava 能辨认 mailto 连接，它产生一项会话为你发送电子邮件。图 9.2 显示了一个电子邮件会话，要发送邮件，可以通过在 HotJava 浏览器底部选取一个链接来实现。该链接是：

```
<A Href="mailto: john@ december. com">john @december. com</A>
```

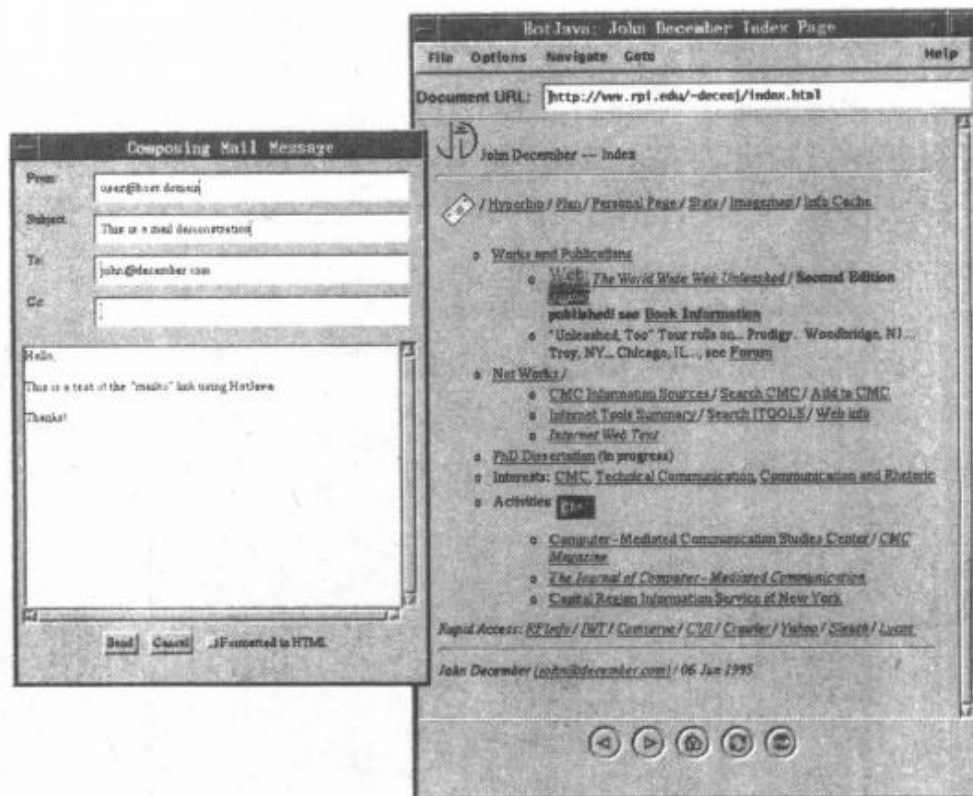


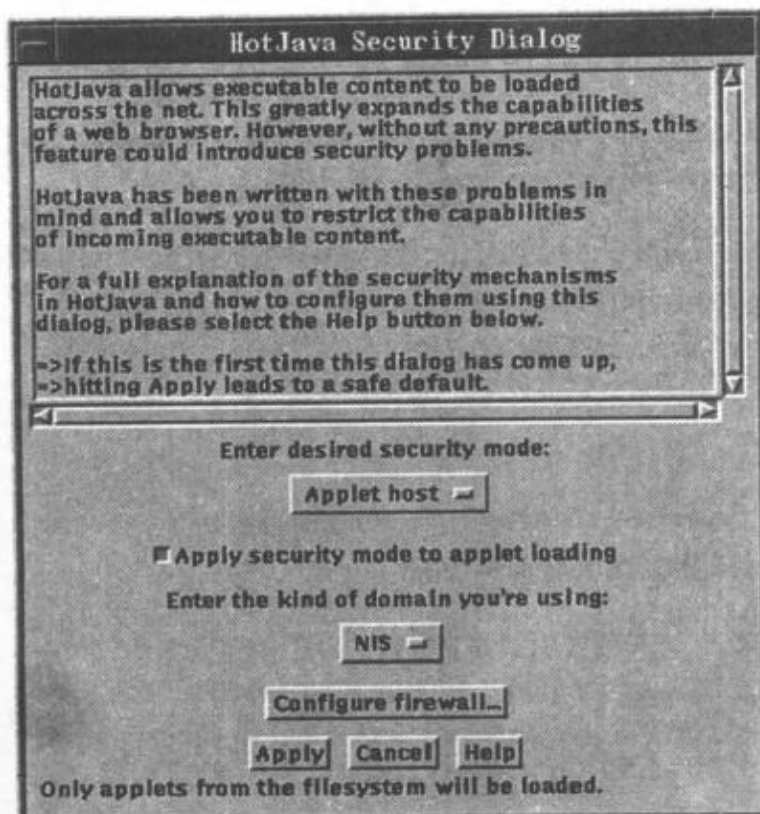
图 9.2 HotJava 邮件会话

安全性是 Java 语言的一个重要部分（参见第 2 章）。HotJava 浏览器为用户提供许多方法来设置文件访问的权限和其它限制。图 9.3 显示了安全对话框的内容，该对话框在执行菜单命令 Options/Security 时显示。

通过安全对话框，你可以标志是否在 applet 调入时使用安全模式，就跟你要确认和配置防火墙等其它安全选项一样。这种安全对话框可以作为另一道防御线，使你能够选择安全属性。譬如，这个对话框允许你只能从自己的主机或在一个防火墙内调入 applet。

选项和反馈

HotJava 提供了许多选项，使你可以定制浏览器，而且你可以得到浏览器操作的反馈信息。



漫游帮助

菜单命令 Navigate 提供下列选择：

- Forward，在浏览器已访问过的资源历史列表中访问下一个地址。如果浏览器正处于历史列表中的最前端，这个命令是灰色的。
- Back，访问历史列表中的前一个资源。如果你正处于历史列表的最开始部分，则在当前资源前没有资源可被访问，此时它变成灰色。
- Home，访问浏览器的主页或起始页。
- Show History，它将弹出一个窗口，它列出你在当前这次 Java 会话中访问过的 Web 页的序列。注意这个序列不必是所有访问过的 Web 页的完整序列，因为 Web 随着页面中包含的多个连接而分支。要记录在 Web 中的漫游路线需要一个算法来填充和刷新一个线性存储栈；这个算法可以截去你所经历过的路径。

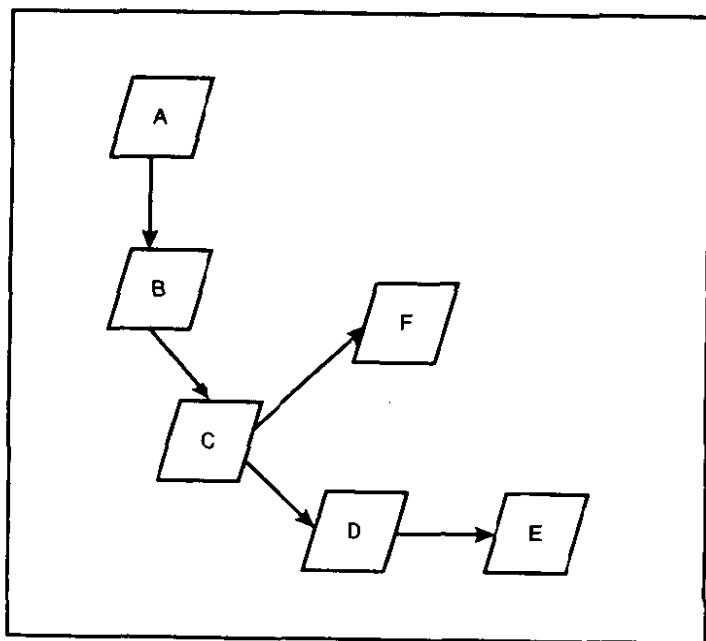


图 9.4 HotJava 移动功能

图 9.4 显示出你在 Web 中的漫游路径，页 A 被连到 B，页 B 被连接到页 C，而页 C 被连接到页 D 和页 F，页 D 被连接到页 E。

如果你从页 A 开始 HotJava 会话,然后顺着超链到达页 E,你的历史列表为:

- page A
- page B
- page C
- page D
- page E

但是如果你此时使用“Back”命令按钮两次返回到页 C,然后从页 C 连接到页 F, 你的历史列表会是:

- page A
- page B
- page C
- page F

页 D 和页 E 路径已经从历史列表中“截去”了,这是因为浏览器不能在其线性栈中存储这种非线性路径。这实际上并不成为问题,只要你意识到历史列表并不是所有访问页的全部历史就够了,它只是最近的线性路径而已。需要申明的是,当你不使用“Back”命令按钮时,这种页 E 和页 D 被截去的情况是不会发生的,还有就是这种历史列表不会传递到下一个 HotJava 会话。

- Add Current to Hotlist, 这个命令把 HotJava 中当前显示的资源加入你的热表中,热表可以存在你的个人磁盘上,而且在两次会话中都可使用。
- Show Hotlist, 这个命令显示当前的热表,它允许你跳转到热表中的任意一项。

图 9.5 显示了一个热表例子。

热表也可被连到菜单栏中的“Goto”命令,通过在热表中选取一个 URL 地址,然后标记“In Goto Menu”命令就可以把这个资源的 URL 地址加到 Goto 菜单中去了。这个命令在热表窗口的底部。

菜单命令 Goto 支持一种热表的快速形式,你可以通过选取

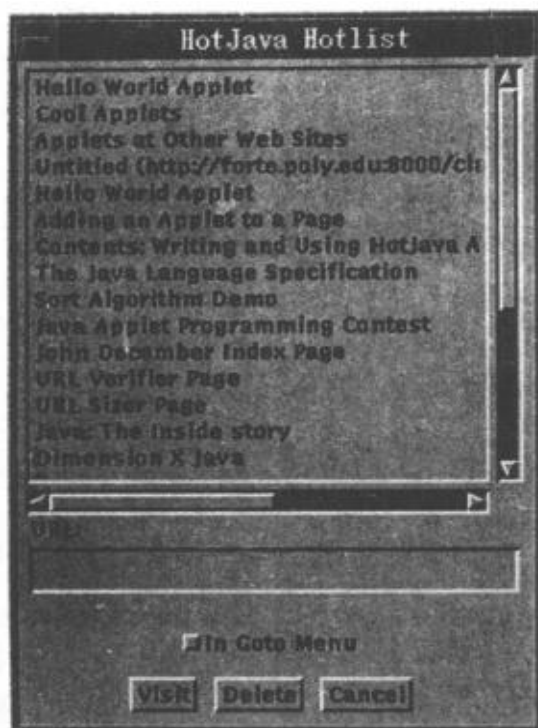


图 9.5 HotJava 热表示例

Goto/Add Current 把一些条目加到 Goto 中去，这就在菜单中为该条目加了一项，从而创建了一个资源快速访问的路径。正如前面所描述的那样，热表中的条目可以快速地转换到 Goto 菜单中去。

内置信息

HotJava 通过 Help 菜单命令提供许多联机帮助信息，这些信息包括：

- 注册信息——注册、版权、授权信息。
- 管理信息——浏览器版本、错误报告和开发队伍。
- 文档——语言详细说明、类库、帮助手册、用户指南和虚拟机详细说明。
- 文档搜索——在一篇或所有文档中的关键字查询。

选 项

HotJava 通过菜单命令 File 提供一系列文件管理，这些命令

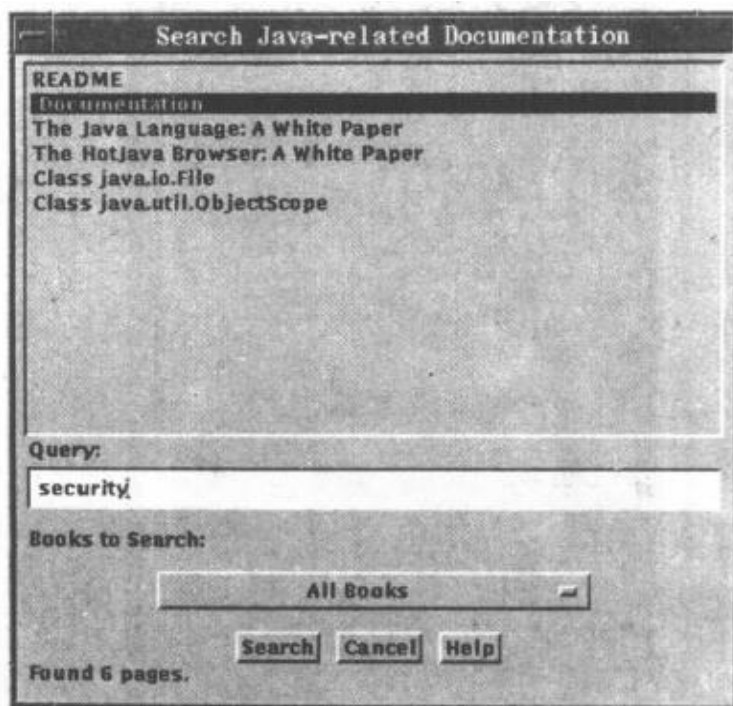


图 9.6 搜索 HotJava 文档

包括：

- Open，这将弹出一个对话框，让你输入一个要获取资源的 URL 地址。
- Reload，这将按照当前文档 URL 窗口内的信息获取资源，它不只是显示的刷新。所有的图形和 applet 跟文本一样，都是从网上获取。
- Print，产生一个弹出式窗口，允许你设置打印选项。
- View Source，产生一个弹出式窗口显示当前资源的 HTML 源文件。
- Quit，结束这次 HotJava 浏览器会话。

菜单栏中的 Options 命令提供：

- Security，设置安全限制的选项和可能的防火墙配置。
- Properties，可以设置许多 proxy，一个 proxy 是指一个服务器，它接收带有指定前缀的 URL 地址，并返回在本地高速缓冲区内查找到的资源。
- Flush Cache，删去存储在浏览器的高速缓冲内存中的图形。如果再次访问含有这些图形的 Web 页时，浏览器将恢

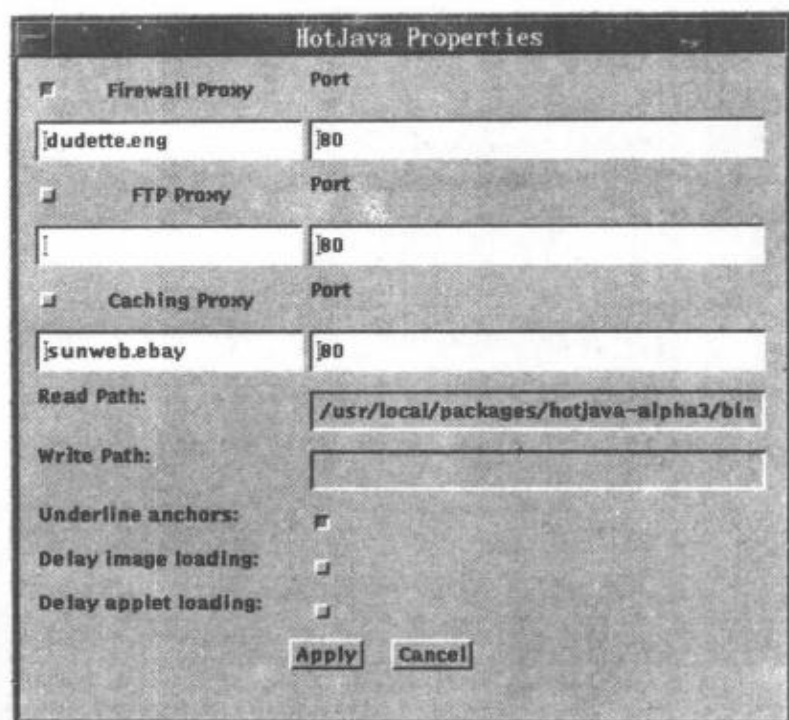


图 9.7 HotJava 浏览器选项

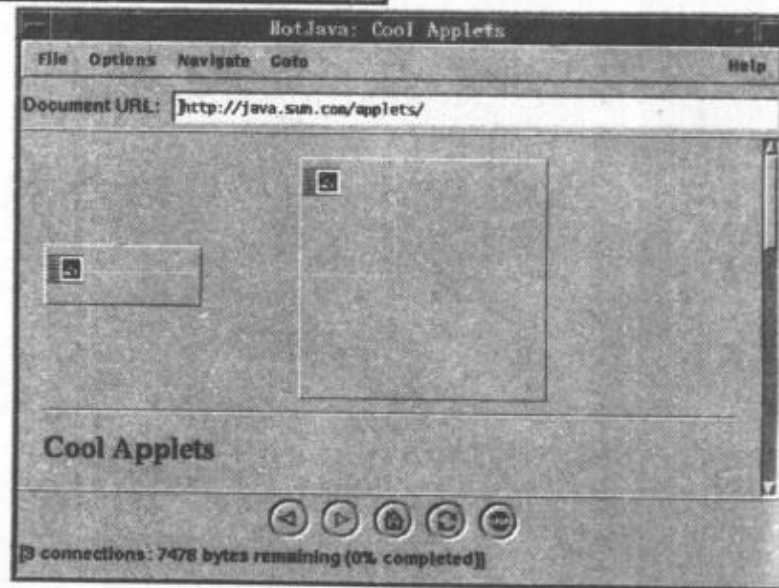
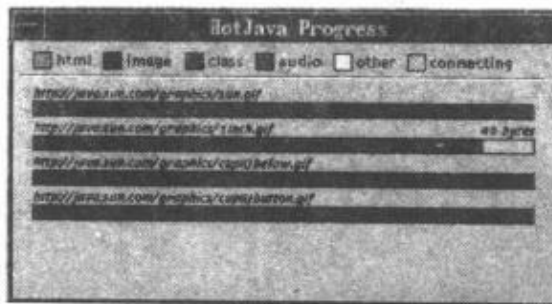
复这些图形。

- Progress Monitor, 产生一个弹出式窗口, 显示正下载资源的进展情况。图 9.8 显示了一个正在下载文件的进展监视器。它显示对一页需要下载什么资源(文本、图形、applet 等等)和哪些还未被下载。这提供了比状态行更多更详细的信息。尤其是当下载多幅图形和 applet 时非常有用。

Web 实用化

HotJava 能为用户提供让用户感到很舒服的界面, 这种舒服的程度依赖于用户的技术培训和 Web 漫游技巧以及经验。总而言之, 你可以通过下面几种方法来提高你的满意程度:

- 意识到有哪些选项可以使用。
- 知道这些选项可满足哪些需要。
- 知道怎样用这些选项来完成有效的工作。



行。因此，无法停止一个 applet 动画，除非 applet 本身提供一个结束的开关。能提供这种开关的 applet 将是一个好的 applet。

弹出窗口

Java applet 可以产生一个弹出式窗口，以提示信息或要求用户输入信息。许多浏览器选项就产生弹出式窗口和用户进行交互（如图 9.8 所示的进展监视器）。applet 同样可产生弹出式窗口，如图 9.9 显示的一个弹出式窗口，它是在 Surf-o-Matic 演示 applet 中出现的。

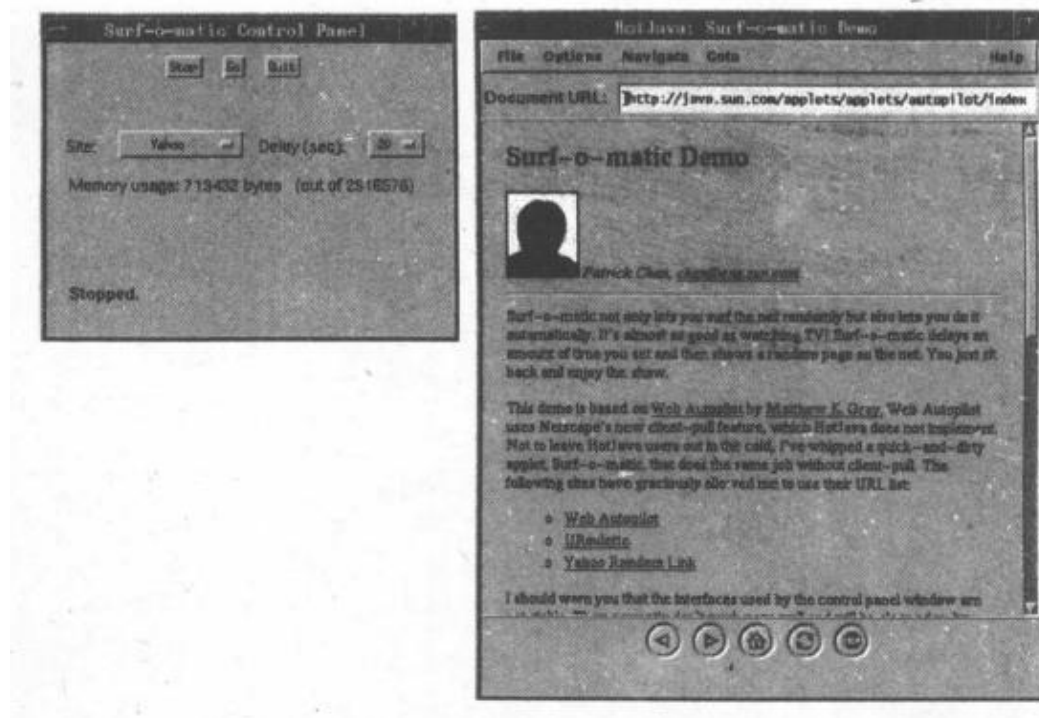


图 9.9 自动巡航弹出式窗口

(Patrick Chan, Sun Microsystems 提供)

浏览器自动巡航

图 9.9 显示的 Surf-o-Matic 就是浏览器自动巡航的例子，通过选取弹出式菜单中的设置，你可以选择自动巡航来漫游 Web。

小 结

本章描述了 HotJava 是怎样支持 Web 漫游器的需求, HotJava 界面给用户自己对漫游功能的控制, HotJava 可用来显示可执行内容。

- HotJava 支持信息显示、链接激活、移动信息控制、交互性、选项和反馈以及 Web 实用化。
- 使用 HotJava 你可以实验动画、弹出式窗口和浏览器自动漫游。

第四部分 开发 Java 应用程序

Java 语言最吸引人的一点在于它并不需要什么高深的技术，它同早先出现的大多数语言一样有着相同的概念、定义与操作。它继承了它的近祖 C++ 的很多特性，但抛弃了 C++ 中不易维护和升级的那部分特征。

这一部分分为三章。概括性地介绍了 Java 语言，并且重点讲解了如何用 Java 语言编写你自己的应用程序和 applets。

alpha Java

提示

这一章中的 applet 是在 alpha (1.3) release of Java 下编译通过的。它们可能在新版本中不能正确工作。详情请洽本书的网络支持节点 <http://www.december.com/works/java.html>。在这一主页中将详细讲述本章代码在新版本中需要作的任何改动。

第 10 章回顾性地介绍了 Java 编程上的一些传统特性：数据定义和程序中的逻辑控制等等。如果你有其它任何一门语言，像 C, C++, PASCAL, FORTRAN 等等的编程经验，你会发现这些特性对你来说不仅从概念上甚至从语法上都很近似。

但是 Java 并不是传统的程序设计语言，它是一门面向对象的编程语言。第 11 章阐述了 Java 的面向对象的特性：Java 对符合类封装标准的软件组织的支持，使用了多层类。你能从中学到如何编写可复用代码和易于二次开发的软件。

最后，Java 是一门唯一将计算和面向对象的特性同不断增加

的对用户界面设计的需求有机结合起来语言,第 12 章通过对最终用户和编程目标的详细分析,讲述了如何编写 applet 和如何使用编程工具,以及如何开发出适合用户需求的 applet 的编程技巧,如何将不同的类、路径和数据结合起来实现 applet 的一整套方法。

这一部分能给予读者以下几点帮助:

- 对 Java 应用程序和 applet 工作原理的理解,以及如何在网络页面上编译、运行和使用 applet。
- 按照 Java 语言语法进行数据定义和逻辑控制编程的基本技巧。
- 对 Java 类和子类的面向对象特性的评价。
- 为 Java 的应用程序和 applet 编写类和子类的方法。
- 如何编写与实现 applet。

alpha HotJava 浏览器中的一个错误

注意

HotJava 浏览器中的一个错误使得如果你在类子目录中重新启动 HotJava,浏览器不能记住你对 applet 的改动。

关于这个错误的最新情况的报告请见 SUN 公司的 WWW 节点:

<http://java.sun.com/>

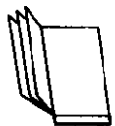
Java 是一门编程语言

拥有 Java，网络将不仅仅是一个传送信息的系统，而是一个交互式系统。网络的这一戏剧性变化依赖于 Java 是一门程序设计语言而不是一门解释语言这一特点。Java 不像网络上通常使用的超文本解释语言——HTML，这是一门文本标记语言，只能做有限的表达式选择。而 Java 是一门表达型的编程语言，提供更多的对信息及其内容的操作。Java 类似于 C++。像 C++ 一样，Java 程序员可以进行逻辑控制、数据操作和执行计算。Java 同 C++ 最大的不同点是：Java 支持网络主页上的交互式操作。

本章将讲述 Java 是如何支持一些正规程序设计语言的特性，像语法结构、数据定义、逻辑控制等等，以及 Java 是如何在它的组织结构和操作中引入面向对象的特性的。

下一章描述了为什么 Java 不仅仅是一门程序设计语言，以及如何利用它的面向对象的概念和特性。

第 12 章介绍了如何编写 Java applet 以及如何利用 Java 的所有特性，它的程序设计能力，它的面向对象的特性，以及它生成可执行文件的能力。



编程语言的变化

●传统的编程语言，把世界看作动作的集合，以便用线性的方法来处理数据，一个传统的程序设计会在问题的处理中产生一个动作，进入一个子动作，这个子动作会用一套方法像过程和子函数来处理信息和数据。

●面向对象把世界看成是一群对象的集合，对象拥有对信息或数据的设置以及定义好的对这些信息（方法）的操作。一个面向对象的程序设计把问题分成是一堆对象的集合，这些对象自身维持着一个现时的状态，它的动作导致用它对应的方法处理这些数据，或向别的对象发出信息。

Java 开发的基本流程

从概念上讲，一个 Java 程序由多个类组成。编程人员用 Java 语言提供的表示符描述这些类，并写入程序代码中。这些代码接下来被编译程序编译一遍，最后产生可被 HotJava 浏览器解释执行的字节码。

一个基本的 Java 应用程序

在第 2 章中作为单个 Java 类演示的“Hello World”程序也是一个最基本的 applet。applet 是可以被支持 Java 功能的浏览器编译并解释执行的 Java 程序。

而另外一种叫作应用程序的则是一种可以脱离 HotJava 浏览器独立执行的 Java 程序，请记住 HotJava 浏览器本身也是一个 Java 应用程序。

下面就是这个非常简单的“Hello World”Java 应用程序的源

代码，文件名为“EssentialApplication.java”。

```
/* *
Program:      EssentialApplication
Purpose:      simple demonstration of an Application
@author       john@december.com
@version      1.00; 17 July 1995
*/
class EssentialApplication {
    public static void main (String args []) {
        System.out.println (" Hello, world!");
    }
}
```

注意

上面的 Java 例程同本书中的其它例程一样，由 Java 编译器的 alpha 测试版本编译。如果你想得到这些例程的面向 Java 编译器的 beta 测试版本或更高版本的 Java 编译器，请询问本书的网络支持页面 <http://www.rpi.edu/~decemj/works/java.html>。

Java 的文件组织遵从常见的格式，用 Java 书写的 EssentialApplication 的源代码存放于扩展名为 .java 的文件中。上面的程序定义了控制流和程序运行时的逻辑结构。这个文件将由 Java 编译器编译并产生机器可读的二进制的类文件，以扩展名 .class 存放。

编程人员可以给源代码文件自由命名，但是编译产生的类文件的缺省的文件名是 Class.class。因此，一个给源代码命名的良好习惯有以下两条：

1. 每个文件只有一个类
2. 文件名同所定义类名相同

你可以编译 EssentialApplication 并且在命令行上运行它（操作系统的提示符假设为 \$ 符号）：

```
$ javac EssentialApplication.java
```

```
$ java EssentialApplication
```

```
Hello world!
```

```
$
```

EssentialApplication.java 的第一部分是注释（由“/*”开始到“*/”结束）。这些注释在编译时被忽略，但是对于以后阅读这些代码的人来说是非常有帮助的。一个良好的习惯是把管理信息和其他信息加入注释中，这些注释使用@author 和@version 作为注释行的抬头，这两行可以被自动全文检索程序检索到。

这个例程的关键部分是对类 EssentialApplication 的定义。Java 关键字 class 表明这儿是一个类的定义，EssentialApplication 是这个类的类名，后面的花括号 { 表明类的主体的开始，类的主体可以由数据和叫做方法的操作构成（传统的程序设计语言有时将这类操作称为函数）。

EssentialApplication 类只包含一个函数 main，但这个函数是每一个应用程序都必须定义的，程序行 public static void main (string args []) { 定义了主函数，主函数 main 前面的修饰符是 Java 的关键字，用来定义函数类型。在阅读本章和下一章中的 Java 语言的细节后，你就会理解它们的含义，但现在你必须记住这些关键字是用来定义 main 函数，以便被任何想用 EssentialApplication 类构造一个例程的人调用。

在 main 后面括号中所跟的参数如 (String args []), String 是字符串对象指针，这是 Java 语言内建的一种数据格式类，指针名是 args，EssentialApplication 应用程序没有使用这些参数，但是每个 main 函数都要定义这些参数，这些参数用来传递命令行参数给应用程序使用。现在你可以在运行程序时跟上两个参数，但输出结果不会改变：

```
$ java EssentialApplication alpha beta
```

```
Hello world!
```

```
$
```

EssentialApplication 从命令行读入参数 alpha 和 beta，并存入数组 args。

接下来的程序段是 EssentialApplication 的 main 函数的主体,这部分由 {} 括号包含。这段程序是包含一个进程调用 println,这个函数用来显示字符串 Hello world!

例程 EssentialApplication 不能做更多的事,它的目的仅仅是向你演示书写 Java 程序的最简单格式,向你示范如何编译并运行它。编写比 EssentialApplication 例程更精细的 Java 应用程序需要了解更多关于 Java 数据表示、计算表达式、流的控制、类和对象组织方面的知识。

基本的 Java applet

当你阅读第 2 章第二小节时,你知道一个 applet 是把 Java 程序连接到 Web 主页上,以便在支持 Java 功能的浏览器上显示的一把钥匙。applet 是一个可被 Web 主页用 HTML 语言中的 APP 标识符链接的一个 Java 程序,它可以由支持 Java 的浏览器解释执行。相反的,像你所看到的 EssentialApplication 例程一样,应用程序是一个不需要浏览器的独立运行的可执行文件,应用程序和 applet 的区别在于 applet 缺少 main 函数,但你可以像编写应用程序一样用 Java 语言书写 applet,下面是一个最简单的 applet 例程:

```
import browser.Applet;
import awt.Graphics;

/* *
Program:      EssentialApplet
Purpose:      simple demonstration of an Applet
@author      john@december.com
@version     1.00; 17 July 1995
--*/

class EssentialApplet extends Applet {
    public void init ( ) {
        resize (600, 300);
    }

    public void paint (Graphics context) {
```

```
        context.drawString (" Hello, world!", 50, 100);  
    }  
}
```

这个例子一样被简化了,以便你更好地理解 applet 是如何工作的。在这一例程中 EssentialApplet 也定义了一个类,像 EssentialApplication 中所做的那样。同样本例程包括了让 applet 可以运行的一切要求。

在注释前定义的部分是用来鉴别其它源代码是否是为本代码编写的子程序,这些 import 定义让其它的源代码可以被本 applet 调用:

```
import browser. Applet;  
import awt. Graphics;
```

browser. Applet 和 awt. Graphics 是 Java Application Programming Interface (API)。这个 API 包含一些常用类,以便可以在编写 Java applet 和应用程序时调用,你可以从以下两种途径得到 API 的全部说明文档:

1. 如果你使用 HotJava 浏览器,选择帮助选项即可。
2. 直接与 Java 主页 (<http://java.sun.com/>) 相连。

接下来的定义 class EssentialApplet extends Applet 将 EssentialApplet 定义为 applet 类的子类,这将使定义为 EssentialApplet 的对象继承内建的 applet 的大多数特性。

applet 的主体必须包含两个函数: init 函数初始化 applet 在网络主页上显示的大小。paint 函数具体产生这个 applet 的显示。

paint 函数包括一个必需的参数 Graphics,它被 applet 中的 context 调用的类型对象,context 在 paint 函数中被调用,用来完成本 applet 的工作——在坐标 (500, 100) 处显示字符串: "Hello world!":

```
context.drawString (" Hello world!", 50, 100);
```

编程人员可以把这个 applet 的源代码保存在文件 EssentialApplet. java 中并编译:

```
$ javac EssentialApplet.java
```

但是你不能直接在命令行上运行这个 applet，如果你试运行一遍，Java 解释器会返回一条出错信息：

```
$ java EssentialApplet
In class EssentialApplet: void main (String args []) is not defined
$
```

当你在命令行上运行这个 applet 时，Java 解释器在代码中搜索 main 函数，以确定你是在运行一个应用程序。

为了运行本 applet，你需要建立一个 HTML 文件，也许叫做 EssentialApplet.HTML：

```
<HTML>
<!--Purpose: HTML file to hold reference to applet. -->
<!--Author: john@december.com-->
<!--Version: 1.10; 21 July 1995-->
<HEAD>
  <TITLE>Essential Applet Web Interface</TITLE>
</HEAD>
<BOOY>
  <P>Essential to an applet: <APP Class=" EssentialApplet" >
</BOOY>
</HTML>
```

上面调用的文件是你用 EssentialApplet.java 编译出来的 EssentialApplet.class。Java 的目录结构是有组织的，它缺省地在目录 classes 下搜寻 .class 文件。

classes 目录必须与 EssentialApplet.HTML 文件在同一目录下。当用户用支持 Java 的浏览器打开 EssentialApplet.HTML 文件时，此 applet 将在浏览器窗口中显示字符串“Hello world!”。

你现在已经阅读了 Java 应用程序和 applet 的两个基本例程，下面让我们来学习关于 Java 语法格式的进一步知识。

Java 编程要点

通过学习上面的应用程序和 applet 的例程，我相信你已经具

有 Java 编程的基本概念。下面将学习更深入的部分,包括 Java 的语法结构,如何使用共享代码类,文件管理和网络资源检索技术。

1. Java 是大小写敏感的,类名 `EssentialApplet` 与 `Essentialapplet` 是两个不同的类。通常,类名是以大写开头而函数名以小写开头。

2. Java 中的注释可以有好几种格式:

```
/* comments can be on
   several lines using the forward slash and star to start and
   the star and forward slash to end
*/
//two forward slashes identify a comment for the rest of the line.
/* * The forward slash and double star is a proposed way to
   identify comments that can be used in automatic
   documentation generation. These comments can go over
   comment keywords such as @author and @version.
*/
```

Java 忽略空白行和注释行,请注意 Java 程序按何种格式书写仅仅是为了人们的阅读方便!对于 Java 编译器来说并无区别,下面就是一段与 `EssentialApplication` 例程完全相同的代码:

```
Class EssentialApplication {public static void main
(String args []) {System.Out.println ("Hello, world!");}}
```

3. Java 的格式大多数来自 C 和 C++ 的编程习惯。Java 编程人员一般用行缩进来区分代码的层次,用花括号 { 来区分嵌套的深度。

4. Java 用 ; 表明一行的结束,用花括号 {} 来表明一段程序的开始与结束。

Java 程序的特点

Java 拥有数据表示、描述计算、控制程序逻辑流程等特性。如果你熟悉 C 和 C++, 你将会发现 Java 语言在格式和数据流程上

都很相似。

基本的 Java 数据

上文的 EssentialApplet 和 EssentialApplication 两个应用程序和 Applet 的例程都过于简单，不能体现 Java 的数据表示特性，下面有一个 Java 应用程序例程演示了如何使用数据表示，使用的数据格式是其他计算机语言的程序员能够认识的。

```
/* *
Program:      EssentialData
Purpose:      simple demonstration of Java data representation.
@author       john@december.com
@version      1.00; 21 July 1995
* /
class EssentialData {
    public static void main (String args []) {
        int      myInteger    =42;
        float    myReal       =3.14159;
        String    someName    =" Mr. Kyle";
        boolean   myTruth     =true;
        int       myArray []  = {1856, 1984};
        System.out.println (myInteger);
        System.out.println (myReal);
        System.out.println (someName);
        System.out.println (myTruth);
        System.out.println (myArray [0]);
        System.out.println (myArray [1]);
        System.out.println (args [0]);
        System.out.println (args [1]);
        System.out.println (args [2]);
    }
}
```

如果你编译并运行此应用程序，在命令行上加上三个参数，结果如下：

```
$ javac EssentialData.java
```

```
$ java EssentialData alpha beta gamma
42
3.14159
Mr. Kyle
true
1856
1984
alpha
beta
gamma
$
```

这个 EssentialData 应用程序演示了 Java 语言内建的基本数据格式。程序行 `int myInteger=42` 定义了一个名为 `myInteger` 的变量，类型为整型，同时将此变量初始化为 42。同样的，后面几行定义了一个实数（浮点数）、一个字符串、一个布尔变量 `myTruth`（可用来判断真值与假值）。

在定义并初始化完这些变量以后，应用 Java 的标准输出函数 `println` 将它们打印出来：

```
System.out.println (myInteger);
```

你可以给一个数组赋值，变量 `myArray` 和后面的 `[]` 表明它是一个数组；

```
int myArray [] = {1856, 1984};
```

`myArray` 是整型数组（由 `int` 定义），此数组由大括号 `{1856, 1984}` 初始化。为了使用数组元素，可以将数组写成后面跟下标的形式，数组元素的下标从 0 开始，`myArray [0]` 是数组的第一个元素，被赋值为 1856。

`myArray` 并不是 EssentialData 中的唯一数组，命令行的参数也被储存在 `main` 函数中所定义的数组 `String args []` 中，此数组直到程序运行时才被赋值。在程序运行时，命令行参数被储存在数组 `args` 中，你可以像使用 Java 中其它字符串一样使用它。例如，这里有一个例程接受命令行参数并显示它们。

为什么从零开始

提示

Java 语言的设计在很大程度上受到 C 和 C++ 的影响。在 C 和 C++ 中数组的下标从零开始, C 编程人员对这个特性是如此的偏爱, 以至于影响到他们以一种奇怪的方式思考。一些程序员会想到一本书有第 0 章, 楼房有第 0 幢, 或者叫他们的第一个孩子是第零个孩子。

```

/* *
Program:      EssentialParams Class
Purpose:      demonstrate writing out parameters;
@author      john@december.com
@version     1.01; 22 July 1995
* /
class EssentialParams {
    public static void main (String args []) {
        int counter;
        for (counter=0; counter<args.length; counter++)
            System.out.println (args [counter]);
    }
}

```

你可以编译并运行这个例程, 当你运行它时, 它会回显所设命令行参数。

```

$ javac EssentialParams.java
$ java EssentialParams regular cafe almond
regular
cafe
almond

```

注意, 函数 `length` 可以用来取得字符串的长度。在 `main` 函数中的 `for` 循环中, 计数器从 0 开始, 到 2 结束, 因为 0 为数组的第一个元素的下标, 2 为数组的最后一个元素的下标。数组同样可以是二维的:

```
/* *
Program:    EssentialArrays Class
Purpose:    demonstrate Java array of arrays;
@author    john@december.com
@version    1.00; 22 July 1995
*/

class EssentialArrays {
    public static void main (String args []) {
        int myTable [][] = new int [2] [3];
                                //this statement sets up a two dimensional array
        System.out.println (" This is an array of arrays...");
        myTable [0] [0] = 1;
        myTable [0] [1] = 2;
        myTable [0] [2] = 3;
        myTable [1] [0] = 4;
        myTable [1] [1] = 5;
        myTable [1] [2] = 6;
        System.out.print (myTable [0] [0]);
        System.out.print (myTable [0] [1]);
        System.out.println (myTable [0] [2]);
        System.out.print (myTable [1] [0]);
        System.out.print (myTable [1] [1]);
        System.out.println (myTable [1] [2]);
    }
}
```

例程产生如下输出:

```
This is an array of arrays...
123
456
```

在附录 B 中列有更多的数据类型,你可以用来进行更加精密的计算。

变量命名的规则

在 Java 中,像其他编程语言一样,命名是受一定限制的,当

你给类、变量和函数命名时，必须遵照以下规则：

1. 命名不能用一些 Java 保留的关键字（Java 利用它们来构造应用程序）。
2. 名字必须以字母（大写或小写）、下划线（_）或者是符号 \$ 开头。
3. 在第一个字母之后可以采用任何字母（包括数字），除了为 Java 保留的关键字（在附录 B 中有这些保留字的列表）。

也就是说，只要遵守上面所述的命名规则，你可以随心所欲地给变量命名，但在实际程序设计过程中，你必须培养一种前后一致的连贯的命名规则，从而使你的程序更易读懂。此命名规则当然是由你决定的，但一些约定俗成的规则是你必须遵守的：

1. 用名词作类名。例如：Car, Bus, Trolley。
2. 用动词作函数名。例如：start, slowdown, stop。
3. 类名以大写开头。
4. 函数名以小写开头。
5. 用大写来区别连词，例如：SubwayCar, IgnitionSwitch, FixEngine。

以上的书写习惯是从面向对象编程中培养出来的，但实际上 Java 程序员也可以发展出一套适合自己编程习惯的命名方法。但最重要的一点，所有命名必须是有意义的。一些早期的语言如 FORTRAN，只允许限定字长的命名，但在 Java 语言中对字长没有限制（除了某些操作系统由于内存不够而产生的限制）。

好的命名习惯使程序的可读性大大增加，并且使程序的二次开发和维护工作较为轻松。有意义的命名也许是提高程序可读性的一条有效途径。看看下面这个函数：

```
public void TUP (T t3) {t3.stg++;}
```

它的作用是什么？对于一个程序员来说，要读懂这行程序是非常困难的，相反，下面这段程序：

```
public void raiseTemperature (Temperature currentTemperature)
{currentTemperature.setting++;}
```

是非常易懂的。输入 `raiseTemperature` 函数需要花更长的时间,但是程序员只需要输入一次,而读者却需要多次阅读这行程序。好的命名习惯可以帮助那些可怜的维护人员,他们在做代码维护工作时,不得不经常破译那些写得艰涩难懂的代码。

基本的 Java 数据类型

数据是给变量所赋的值,例如,一个整形数据是 1856。下面的例程演示了数据是如何被用多种途径表达和显示出来。

```

/* *
Program:    EssentialLiterals
Purpose:    simple demonstration of Java data literals.
@author     john@december.com
@version    1.00;21 July 1995
*/
class EssentialLiterals{
    public static void main(String args[]){
        System.out.println("INTEGER(BASE 10)");
        System.out.println(10);           //This is a base 10 integer
        System.out.println(4294967295);    //large 32 bit integer(int)
        System.out.println(9223372036854775807L);
                                   //large 64 bit integer(L on end=long)
        System.out.println(2E3);           //scientific notation;2 times 1,000
        System.out.println("INTEGERS IN OTHER BASES");
        System.out.println(010);           //A leading 0 means octal(bass 8)
        System.out.println(0x10);
                                   //A leading 0x means hexidicemal(base 16)
        System.out.println("BASE 16");
        System.out.println(0xA);           //A in base 16 is the digit for 10
        System.out.println(0xB);           //B in base 16 is the digit for 11
        System.out.println(0xC);           //C in base 16 is the digit for 12
        System.out.println(0xC);           //D in base 16 is the digit for 13
        System.out.println(0xE);           //E in base 16 is the digit for 14
        System.out.println(0xF);           //F in base 16 is the digit for 15
        System.out.println("BOOLEAN");
    }
}

```

```

        System.out.println(true);
        System.out.println(false);
        System.out.println("FLOATING POINT");
        System.out.println(3.1415);           //real number
        System.out.println(-0.61803);         //negative real number
        System.out.println(2.9E4); //scientific notation; 2.9 times 10,000
        System.out.println("STRINGS");
        System.out.println("");               //empty string
        System.out.println("\"Hi,Lorrie.\""); //quote in string
        System.out.println("one\ntwo");       //linebreak in string
        System.out.println("hot\tjava");       //tab in string
    }
}

$ javac EssentialLiterals.java
$ java EssentialLiterals
INTEGER(BASE 10)
10
4294967295
9223372036854775807
2000
INTEGERS IN OTHER BASES
8
16
BASE 16
10
11
12
13
14
15
BOOLEAN
true
false
FLOATING POINT
3.1415
-0.61803

```

```
29000
STRINGS
"Hi, Lorrie."
one
two
hot      java
```

基本的 Java 操作符

Java 拥有足够的操作符去实现对其内建数据格式的任何计算操作。下面的例程演示了 Java 中一些你熟悉的操作符。

```
/* *
Program:      EssentialOperators
Purpose:      simple demonstration of Java operators.
@author       john@december.com
@version      1.02; 21 July 1995
class EssentialOperators{
public static void main(String args[]){
int      myInteger      =42;
float     myReal         =3.14159;
String    someName      ="Mr. Kyle";
boolean   myTruth        =true;
int       myArray[]     ={1856,1984};
System.out.println("UNARY OPERATORS");
System.out.println(-myInteger);      //negate an integer
System.out.println(-myReal);         //negate a real number
System.out.println(myInteger++);     //display the number then add 1 to it
System.out.println(myInteger--);     //display the number then subtract 1
System.out.println("BINARY OPERATORS");
System.out.println(myInteger+100);    //addition
System.out.println(myInteger-myReal); //subtraction
System.out.println(100 * myReal);     //multiplication
System.out.println(myInteger/3.0);    //division with real
System.out.println(myInteger/3);      //division without remainder
System.out.println(myReal/3);         //real division
System.out.println(myInteger%3);      //modulo arithmetic shows
```

```

//remainder after dividing by 3
System.out.println("RELATIONAL OPERATORS");
System.out.println(myInteger 32)      // greater than
System.out.println(myInteger>=32)    //greater than or equal to
System.out.println(myInteger
$ javac EssentialOperators.java
$ java EssentialOperators

```

UNARY OPERATORS

```

-42
-3.14159

```

```
42
```

```
43
```

BINARY OPERATORS

```

142
38.8584
314.159
14
14
1.0472
0

```

RELATIONAL OPERATORS

```

false
false
true
true
true
false

```

BOOLEAN OPERATORS

```

false
false
true
true
false
true
false
Mr. Kyle is cool.

```


14. = op=

15. ,

在用变量和表达式工作时，另一个需要注意的问题是类型转换。类型转换意味着你将一个表达式从一种数据类型转化为另一种数据类型。例如，将一个实形数据类型转换为整形（小数点后部分被截去），或者把实数、布尔量、浮点数转换为字符串。下面是示范类型转换的一个例程：

```

/* *
Program:    EssentialCasting
Purpose:    simple demonstration of Java operators casting.
@author     john@december.com
@version    1.02; 21 July 1995
class EssentialCasting {
    public static void main (String args []) {
        int      myInteger    =42;
        float    myReal       =3.978;
        boolean  myTruth      =true;
        String   aPhrase;
        float    aReal;
        float    anInteger;
        anInteger= (int) myReal;           //floating to integer
        System.out.println (anInteger);
        aReal=myInteger+0.001;            //floating expression
        System.out.println (aReal);
        aPhrase=" Survey says;" +myInteger; //integer to string
        System.out.println (aPhrase);
        aPhrase=" Survey says;" myTruth;   //boolean to string
        System.out.println (aPhrase);
    }
}

```

运行此例程：

```

$ java EssentialCasting
3
42.001

```

```
Surver says: 42
```

```
Surver says: true
```

此例程显示了如何用类型转换符进行显性的类型转换。例如，你可以像下面这样将一个实数转换为整数：

```
anInteger = (int) myReal;
```

类型转换也可以是隐含的。例如，当你书写一个表达式用来给一个字符型变量赋值时，表达式中的变量会被转换成字符型。例如，EssentialCasting 中的表达式：

```
aPhrase = " Survey says:" + myInteger;
```

使用了隐性的类型转换来将 myInteger 的值转换成一个字符串并附在 Survey says 后面。

基本的表达式

在 Java 的 Math 类中也支持各种数学表达式和特殊函数：

```
/* *
```

```
Program:    EssentialExpress Class
```

```
Purpose:      demonstrate Java math expressions;
```

```
@author    john@december.com
```

```
@version    1.02; 22 July 1995
```

```
class EssentialExpress {
```

```
    public static void main (String args []) {
```

```
        double example; //double precision is available for math
```

```
        //math operations are available;
```

```
        example = (3+2) + 5 * 3/7 + 3.123 + (42 * 7 + 4);
```

```
        System.out.println (" example result=" + example);
```

```
        //common math functions are available;
```

```
        example = Math.sin (0.50) + Math.log (13) + Math.sqrt (2)
                + Math.exp (4);
```

```
        System.out.println (" example result=" + example);
```

```
        //other functions;
```

```
        System.out.println (" absolute value of -3=" + Math.abs (-3));
```

```
        System.out.println (" a random number (0.0 to 1.0) ="
                + Math.random ());
```

```
System.out.println (" round (3.65) =" +Math.round (3.65));  
System.out.println (" 2 to the third power (2 * 2 * 2) ="  
                    +Math.pow (2, 3));  
System.out.println (" maximum of 3.98 and 14.2 ="  
                    +Math.max (3.98, 14.2));  
}  
}
```

下面是输出：

```
$ javac EssentialExpress.java  
$ java EssentialExpress  
example result=308.123  
example result=59.0567  
absolute value of -3=3  
a random number (0.0 to 1.0) =0.396465  
round (3.65) =4  
2 to the third power (2 * 2 * 2) =8  
maximum of 3.98 and 14.2=14.2
```

基本的流程

以上所述的是构筑 Java 程序的“脚手架”和“砖石”，但是如果你想真正用 Java 编制一些实用的东西，你必须学习比简单的从上至下的顺序编程方法更复杂的编程方法。程序流程控制是很多编程语言的关键部分。Java 拥有让你表达任何逻辑流程控制的一切必需品。

程序流程控制分为：

1. 分支：基于某个给定的状态是真或假来控制程序流转向何处执行（见图10.1）。

2. 循环：反复执行某一行（或某一段）程序（见图10.2）。

编程中使用分支是因为需要作决定。使用循环来重复执行某个操作直至某一状态发生改变。Java 支持好几个指令来完成分支或循环。

下面的例程演示了 Java 中基本的程序流程控制：

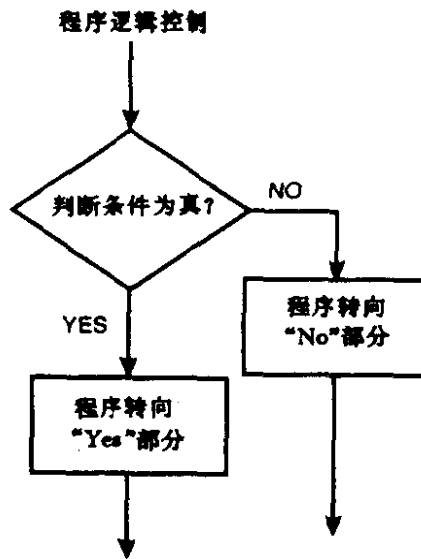


图10.1 分支控制

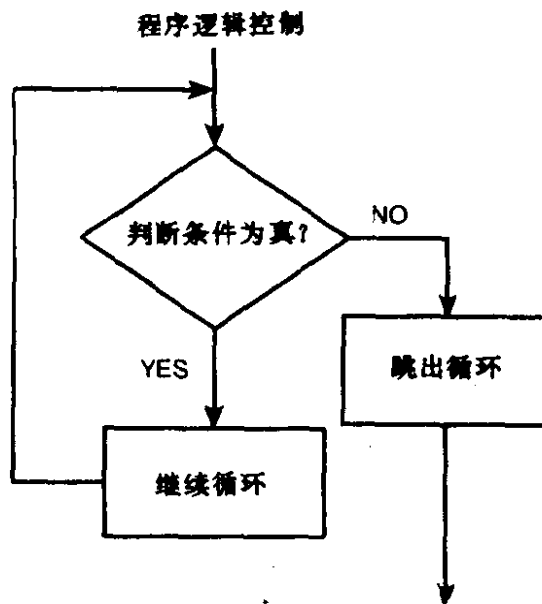


图10.2 循环控制

```

/ * *

```

```

Program: EssentialFlow

```

```

Purpose: simple demonstration of Java control flow

```

```

@author john@december.com

```

```

@version 1.03; 21 July 1995

```

```

* /

```

```

class EssentialFlow {

```

```

    public static void main (String args []) {

```

```
System.out.println (" BRANCHING");
//If-then branching.
boolean OK=true;
//This is a simple if-then statement.
if (OK) {
    System.out.println (" OK. ");
else
    System.out.println (" NOT OK. ");
//Quick version of if-then branching.
OK? System.out.println (" Quick OK. ");
    System.out.println (" Quick NOT OK. ");
//Multiple line if-then branches.
if (OK) {
    System.out.println (" This is");
    System.out.println (" remaining OK. ");
}
else {
    System.out.println (" Or is it");
    System.out.println (" NOT OK?");
}
//SWITCH type branching
int temperature=37;
switch (temperature) {
    case (0);
        System.out.println (" Freezing water!");
        break;
    case (37);
        System.out.println (" Human body. ");
        break;
    case (100);
        System.out.println (" Boiling water!");
        break;
    default;
        System.out.println (" some temperature. ")
}
System.out.println (" LOOPING");
//Simple FOR loop up.
```

```
int counter;
for (counter=1; counter > 0; counter--) {
    System.out.println (" Hi down:" +counter);
}

//FOR loop with non-unit step sizes.
for (counter=-50;
    counter<90;
    counter=counter+50) {
    System.out.println (" Hi again:" +counter);
}

//WHILE loop with condition checked at the start.
counter=0;
OK=true;
while (OK) {
    System.out.println (" Still OK:" +counter);
    counter++;
    OK= (counter<3);
}

//DO-WHILE loop executes at least once; condition check at end.
counter =500;
OK=true;
do {
    System.out.println (" OK=" + OK + " counter=" +counter);
    counter++;
    OK= (counter <3);
} while (OK)
}
```

```
$ javac EssentialFlow.java
```

```
$ java EssentialFlow
```

BRANCHING

OK.

Quick OK.

This is

remaining OK.

Human body.

LOOPING

Hi up: 1

Hi up: 2

Hi down: 3

Hi down: 2

Hi down: 1

Hi again: -50

Hi again: 0

Hi again: 50

Still OK: 0

Still OK: 1

Still OK: 2

OK=true counter=500

Java 语言支持标号跳转和更费解的断点语句。但是好的编程习惯一般都是尽量避免在程序中使用跳转指令。因为断点和 goto 语句使程序员难以跟踪程序指向。精心设计的流程语句,可以在程序中完成任何控制逻辑,而不用一条跳转语句。

基本函数

创建变量,定义控制流程是任何计算机编程的基本步骤。在完成这两步后,理所当然的,一个程序员可以完成任何计算机程序。但是一个大程序在成为一份极长的源代码文件后,很快就变得极其费解。

一个编程人员的准则是坚持编写易于整理的代码块,这对编程和阅读程序都有好处,编制可复用代码是开发易于管理的软件的关键技术。在 Java 中,利用函数来完成这一点。下面是一个函数例程:

```
public static void repeater (String line, int times) {  
    int counter;  
    for (counter=1; counter <=times; counter++) {  
        System.out.println (line);  
    }  
}
```

此函数叫作 repeater。像上面例程中使用的 main 函数一样，此函数可以被任何人调用 (public)，联想到此类在何处定义 (static)，执行完毕不返回值 (void)。不像 main 函数名为事先给定，此函数名是由程序员定义，并且此前没有定义同名函数。repeater 的主体部分定义了一个真正有用的函数，它只是一个小工具，供别的函数调用，也许是一个更强大的程序的一部分。下面是一个演示如何在另一个类中调用 repeater 函数的应用程序实例。

```

/ * *
Program: EssentialMethods
Purpose: simple demonstration of Java methods
@author john@december.com
@version 1.02; 22 July 1995
*/
class EssentialMethods {
    public static void main (String args []) {
        greeter (" Hello, methods.");
        greeter (waver (3));
    }
    public static void greeter (String greeting) {
        repeater (greeting, 1);
    }
    public static String waver (int waves) {
        repeater (" WAVE", waves);
        return (" Done waving.");
    }
    public static void repeater (String ling, int times) {
        int counter;
        for (counter=1; counter<=time; counter++) {
            System.out.println (line);
        }
    }
}

```

此应用程序根据 main 函数定义程序流程中的函数调用，将会显示一些字符串。main 函数是在程序中首先被调用的“发动机”，并

且由它结束程序流程。在 main 函数中调用了函数 greeter，首先用字符串“Hello, methods”，接着用另一个函数调用 waver(3)，函数 waver 被定义为 public static string，这意味着它将返回一个字符串。由 waver 函数的代码可以看出它将返回一个字符串，Done waving 作为其结束标志，此字符串将作为函数 greeter 的参数。

注意，在一个类中，一个函数可以在它被定义和书写之前被调用。greeter, waver 两个函数都调用了 repeater 去完成显示工作，这是一个微妙之处，但也许对后续的代码维护工作是非常有价值的。所有的显示工作都在一个函数中完成。

如果你后来发现显示功能需要改变，你只需要改动函数 repeater 的主体部分。

图10.3显示了调用和参数传递过程，以及程序运行时发生了什么。

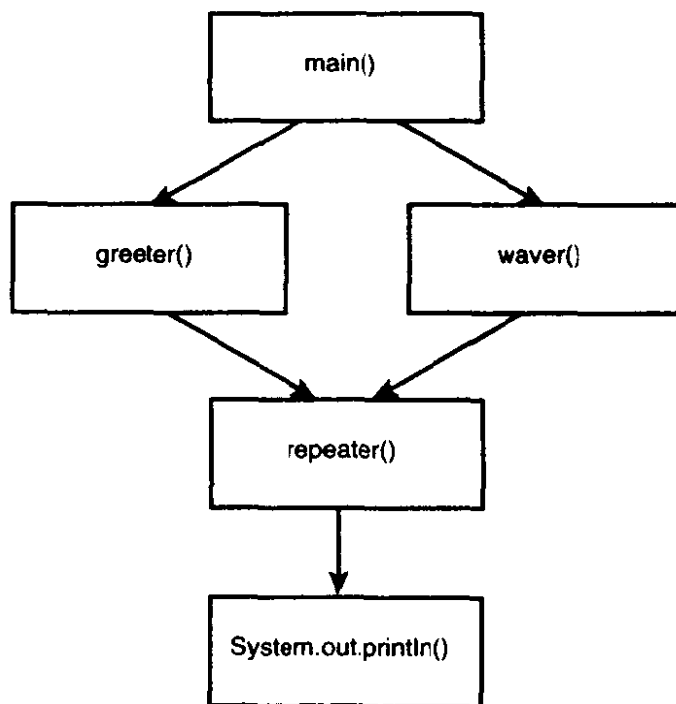


图10.3 EssentialMethods 中的函数调用关系

下面是输出：

```
$ javac EssentialMethods.java
```

```
$ java EssentialMethods
```

```
Hello, methods.
```

```
WAVE
WAVE
WAVE
Done waving.
```

变量作用域

在 Java 程序中，可以在函数内部定义变量。例如：在函数 `repeater` 中定义了整型变量 `counter`，并且在循环体中使用它。此变量被限制在函数 `repeater` 中使用，而不能在 `repeater` 函数以外使用。变量 `counter` 的作用域是在函数 `repeater` 中。作用域指程序代码中某变量的使用范围。

下面的例程演示了 Java 的变量作用域和常量：

```
/* *
Program: EssentialScope
Purpose: simple demonstration of Java scope and constants
@author john@december.com
@version 1.03; 22 July 1995
*/
class EssentialScope {
    static int counter=10;
    static final int MAXIMUM=99;
    public static void main (String args []) {
        //Illegal: MAXIMUM=900; Can't assign values to final variables
        System.out.println (" main: counter=" +counter);
        System.out.println (" main: MAXIMUM=" +MAXIMUM);
        alpha ();
        beta ();
        gamma (counter);
        System.out.println (" main: counter=" +counter);
    }
    public static void alpha () {
        System.out.println (" alpha: counter=" +counter);
        System.out.println (" alpha: MAXIMUM=" +MAXIMUM);
    }
    public static void beta () {
```

```

        int counter=900;
        System.out.println (" beta: counter=" +counter);
        System.out.println (" beta: EssentialScope.counter=" +
                                EssentialScope.counter);
    }

    public static void gamma (int counter) {
        int gammaCounter=500; //undefined variable outside of gamma
        counter=counter+100;
        System.out.println (" gamma: counter=" +counter);
        System.out.println (" gamma: gammaCounter=" +gammaCounter);
    }
}

```

本例程中的类定义了两个变量：counter 和 MAXIMUM，它们被定义为 static，这意味着它们同类 EssentialScope 的联系比此类对象更紧密（关于类和对象的区别，在下一章面向对象特性中详细阐述）。对 MAXIMUM 定义时使用了的关键字 final，将 MAXIMUM 定义为常量（关键字 static 并不能定义为常量）。通常应全部用大写字母书写常量。

下面是例程的输出：

```

$ javac EssentialScope.java
$ java EssentialScope
main: counter=10
main: MAXIMUM=99
alpha: counter=10
alpha: MAXIMUM=99
beta: counter=900
beta: EssentialScope.counter=10
gamma: counter=110
gamma: gammaCounter=500
main: counter=10

```

在本类中的所有函数中都可调用 counter 和 MAXIMUM，但是对 MAXIMUM 不能再赋值，本类中的显示语句调用了并打印出这些变量：

alpha 函数演示了类的数据成员的作用域是如何扩展到函数

体内部。

beta 函数演示了如果函数的一个数据成员同类的一个数据成员有相同的名称, 则类的数据成员被“隐藏”起来。beta 函数的第一条打印语句显示了在函数中定义的数据成员 counter。第二条打印语句演示了可以用加上类名的办法扩大类的数据成员的作用域, 方法是在类名后跟上圆点符和成员名: EssentialScope. counter。所以当类的数据成员被函数的数据成员屏蔽时, 可以通过加上类名的方法在函数中调用类的数据成员。

gamma 函数定义了一个数据成员 gammaCounter。它的值可以在 gamma 函数中被改变。但这种改变也同样被限定在 gamma 函数体内部, 类的数据成员 counter 的值不会被函数 gamma 中的 counter 改变。

Java 的可执行特性

下面的应用程序例程是 Java 语言的基本特点。当然, Java 可以做更多的事情, 你可以用它创作一个“动画”主页。

为了让 Java 程序员产生可执行代码, Java 语言提供了一个叫做 Runnable 的接口, 接口是指可被 Java 类调用的一些函数集。接口 Runnable 使 Java 程序员可以用它产生实用的 applet。当你定义了一个类实现接口 Runnable, 你必须在此接口中定义所有函数。下面是一个应用程序的定义部分, 它做到了这点。

```
import awt. * ;
import browser. Applet;
/* *
Program:      EssentialRunner application
Purpose:      simple demonstration of a runnable applet
@author       john@december.com
@version      1.54; 22 July 1995
* /
public class EssentialRunner extends Applet implements Runnable {
    private Thread flow;
```

```
boolean done, pause;
static int width=500;
static int height=500;
static int interval=5;
int x, y, cycle;
public void init () {
    resize (width, height);
}
public void paint (Graphics g) {    //this method draws the applet
    g.SetColor (Color.green);
    g.drawLine (width/2, width/2, x, y);
    y=y+interval;
    if (y>width) {
        y=0
        cycle++;
        if ( (cycle%2) ==0)
            x=cycle * interval;
        else
            x=width-cycle * interval;
    }
}
public void update (Graphics g) {
    paint (g);
}
public void start () {
    done=false;
    pause=false;
    x=0;
    y=0;
    cycle=0;
    flow=new Thread (this);
    flow.start ();
}
public void run () {
    try {
        while (!done) {
            Thread.sleep (interval);
            if (!pause) repaint ();
        }
    }
}
```

```
    }  
    }  
}  
  
public void stop () {  
    done = true;  
    flow.stop ();  
}  
  
public void mouseUp (int x, int y) {  
    pause = !pause;  
}  
}
```

图 10.4 显示了 EssentialRunner 输出的静止图像。

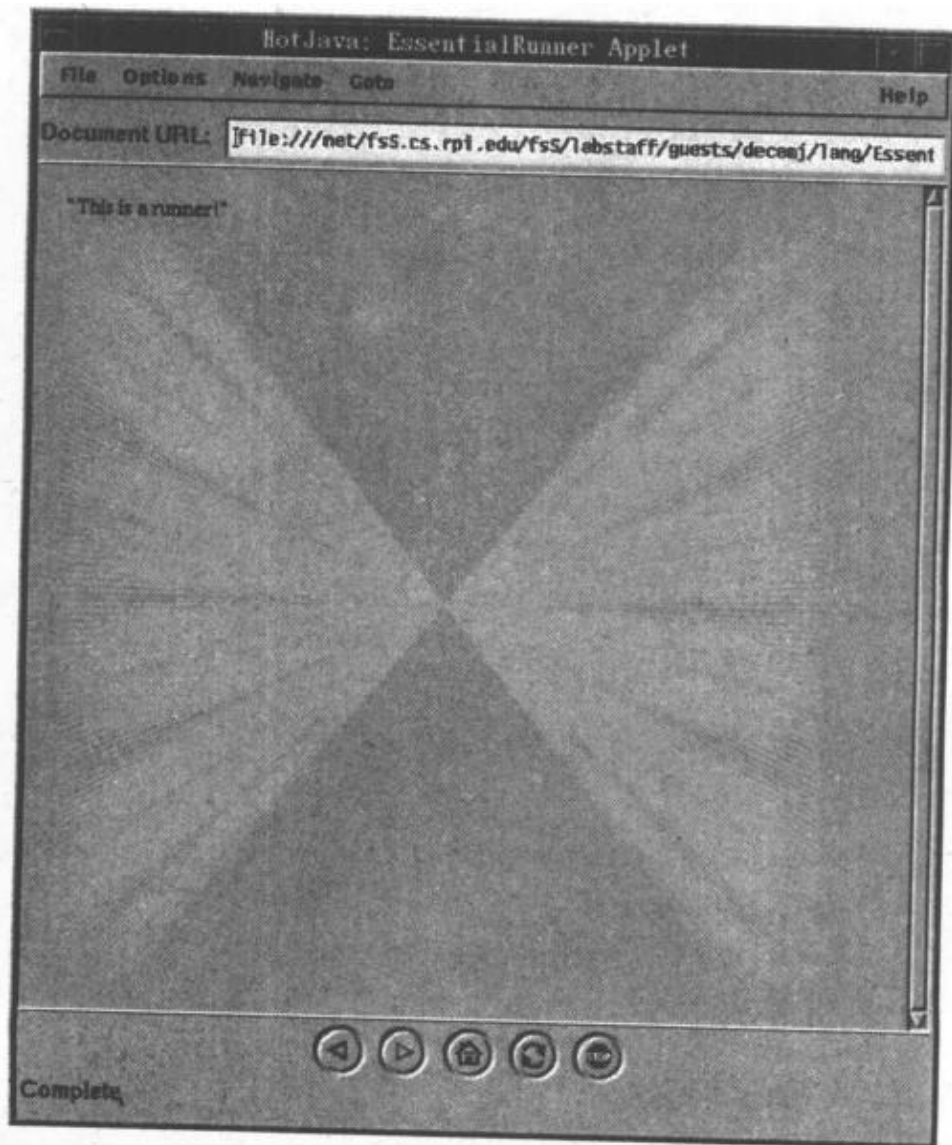


图 10.4 EssentialRunner applet 的输出

这个 applet 连续地在屏幕上画线。如果你将光标移到显示此 applet 的浏览器窗口,你可以通过单击鼠标来停止画线过程,当再单击鼠标时, applet 重新开始运行。

此程序演示了基本的 Runnable applet: run 函数首先被调用并且成为此 applet 的“发动机”。start 函数是首先被调用的,它的关键行是:

```
flow=new Thread (this)
```

这一行创造了一个对象,类型为在 EssentialRunner 类中定义的 Thread, EssentialRunner 类传递一个指针给它本身以使用 this 这个关键词创造一个新的 Thread 对象。

流 thread 控制着 applet 的行为速度,例如:在 run 函数中,下面这一行:

```
Thread.sleep (interval);
```

暂停了程序一小会(间隔设为 5 毫秒)。EssentialRunner 中最复杂的函数是 paint 函数,它利用前一条线的坐标和循环变量计算出下条线的坐标以便绘制。

Java 联机帮助

Sun 公司的节点 <http://java.sun.com> 拥有大量 Java 语言文档,分为以下几部分:

提示

- Java 编程指南 (<http://java.sun.com/prog-Guide/java/TOC.html>)。提供大量 Java 入门课程。
- Java 文档 (<http://java.sun.com/documentation.html>) 提供多个 Java 联机文本链接。
- 一般 Java 文本可由匿名节点 FTP 得到 (<ftp://java.sun.com/docs/>)。javaspec.ps.gz 是 Java 编程语言的特别介绍。

小 结

Java 语言通过生成可执行内容给网络带来了新的潜力。但是,归根结底,Java 是一门编程语言,拥有数据定义、流的控制、计算语句、生成函数和子程序的途径。Java 同时也赋予程序员创建可在实际环境中运行的 applet 的能力。

- Java 的基本构件是类,类是一个用来创建对象的模板,在类的定义中有对函数和数据的定义。
- Java 程序可以是独立运行的应用程序或由 HotJava 浏览器解释执行的 applet。
- Java 数据可以被定义成整型 (int)、实型 (float)、字符型 (String) 和逻辑值 (boolean)。Java 数据可以放在数组中,下标起始值为 0。
- Java 支持操作符的排列组合,以及计算和比较的数学表达式。表达式可以通过类型转换转换为别的数据类型。例如,一个实型表达式可转换为字符型或整型。
- Java 中逻辑控制语句支持分支和循环,分支由 if...else 或 switch...case 语句支持;循环由 while 语句支持,可以循环 0 到任意次,由 do...while 语句支持可以循环 1 到任意次。
- 函数是 Java 中的一个定义,用以产生可复用代码,一个函数像是一个小程序,它自己可以拥有变量和函数。判断何处可以调用一个变量的规则叫变量作用域规则,此规则帮助解决诸如一函数中定义的变量与在类中定义的变量名称相同之类的问题。
- Java Runnable 接口允许所有程序员创建可执行的页面。
- 本书的附录 B 给出了对 Java 语言细节的快速索引。

Java 是面向对象的

上一章讲述了 Java 应用程序和 applet 的基本编程结构和语言特性。因为 Java 由面向对象语法组织起来的，它的结构中把类作为编程的基本构件。

但是，将类定义看成只是编程中的一个框架也许是一种错误。为了真正发挥面向对象方法的威力，用户需要了解类是怎样构成的，及如何用这个类去创造子类和对象。实际上，用户在使用现有的类和利用它们对世界建模方面的技巧对 Java 程序员来说是最重要的。

本章讲述了 Java 中的类及如何用一个类创建子类和对象。

Java 中类的定义和使用

上一章中的例程中的类即是真正的 Java 类。但是由于它们只是演示程序，因而并不能显示由类构成的软件的威力。这一节首先剖析一个类的各部分，包括不同的函数以及它们如何工作的。

一个类例程

类是 Java 程序在概念和语法上的最基本组成部分。Java 类描述了数据和对此数据作用的函数。当用一个类定义的类型去定义一个标志符，此标志符即成为此类的一个对象，因而类像是一个产生由它定义类型的对象的模板。

你可以用类对真实世界建模。例如，保险箱在真实世界里用来储藏并保护物品，只让拥有这些物品的人可以取得它们。一个

简单的保险箱有一个用号码才能打开的门，保险箱既可以锁上也可以打开。你可以用正确的号码去打开保险箱。任何人可以从外面观察此保险箱、箱的标志（例如门的序号）或试试看是否此箱已被锁上，也可以用任意的号码去试验看是否能打开保险箱。但是只有为保险箱设计的号码才能打开此保险箱，只有保险箱的设计者才可以设置号码锁以使得只有唯一的号码才能打开它。

上面的保险箱是可以用 Java 类建模的好对象，下面是定义 safe class 的 Java 代码，你可以在此类中找到比上一章所述更多的一些特性：

```
/* *
Program:      Safe Class
Purpose:      demonstrates a Java Class and protections;
@author      john@december.com
@version     1.04; 23 July 1995
*/
class Safe {
    public      int      doorNumber      =123;
    private     boolean  locked           =true;
    private     int      combination      =456
    public boolean isLocked () {
        return (locked);
    }
    public void unLock (int thisCombination) {
        if (thisCombination==combination) unLock ();
    }
    private void unLock () {
        locked=false;
    }
    private void setCombination (int setting) {
        combination=setting;
    }
    Safe () {}
    Safe (int door) {
        doorNumber=door
```

```
        setCombination (doorNumber); //this is the trick to cracking a safe
    }
}
```

第一行 `class Safe` {通知 Java 编译器, 这是一个类的定义部分, 此类名为 `Safe`。其余的代码定义了类的主体部分。

Safe 类的数据

下面是 `Safe` 类的数据:

```
public int doorNumber =123;
private boolean locked =true;
private int combination=456;
```

并不是一定要把所有的变量放到类的开头, 但是通常良好习惯是将所有变量放在一起以便读者能很快找到哪儿是类的描述部分。`Safe` 类拥有三个变量:

- `doorNumber` 存放着标记在保险箱外的序号, 告诉大家这是哪个保险箱, 它是一个整型变量, 初始值为 123, 这意味着除非改变类的描述, 不然任何一个由 `Safe` 类定义的对象都拥有序号 123。这并不意味着这个序号是不可改变的, 因为定义 `doorNumber` 时并没有使用关键字 `final`, 它不是一个常数。
- `locked` 存放着保险箱的状态: 是打开还是锁上。它可以用一个布尔变量来描述, 初始值为 `true`。
- `combination` 存放着保险箱的号码。拥有这个号码, 保险箱才能被使用者打开。

一个类中的数据可以被定义成对其它类来说有不同级别的权限。在例程 `safe` 中, 显然不能允许任何人都看见号码或任意改变用以存放 `safe` 是锁着还是打开的变量。因此变量 `locked` 和 `combination` 被定义为这个类的私有数据成员, 这意味着不能在这个类的外面检查或改变这两个变量, 被声明为 `Safe` 类型的对象显然不能被骗去号码或被不正确的号码打开。因为序号允许被任何人看见, 所以它是 `public` 的, 此变量在被声明为 `Safe` 类型的对象内部


```
doorNumber=door;  
setCombination (doorNumber); //this is the trick to cracking a safe  
}
```

此构造函数允许你创建一个 Safe 类型的对象，此对象的序号有一个特别的值。这个值被当作一个参数传递。这里有两个构造函数意味着此类中的函数 Safe 是重载函数。用第二个构造函数，你可以制造一个序号为 1927 的保险箱。

```
Safe anotherSafe = new Safe (1927)
```

Java 语言能够分辨出你调用的是哪个构造函数，因为每个构造函数有不同参数：

Safe (1927) 将会调用第二个构造函数，这个构造函数是一个可以打开任意保险箱的“诀窍”，你可用门的序号来代替号码，当然这是一个极差的安全特性，但是它可以很好地解释类 Safe。

如图 11.1 所示，你可以观察类 Safe 的构造。一个保险箱的保护数据和函数包括密码和门的的状态以及拥有操作这些对外界函数来说是被保护的和“隐藏”的数据和函数的能力。

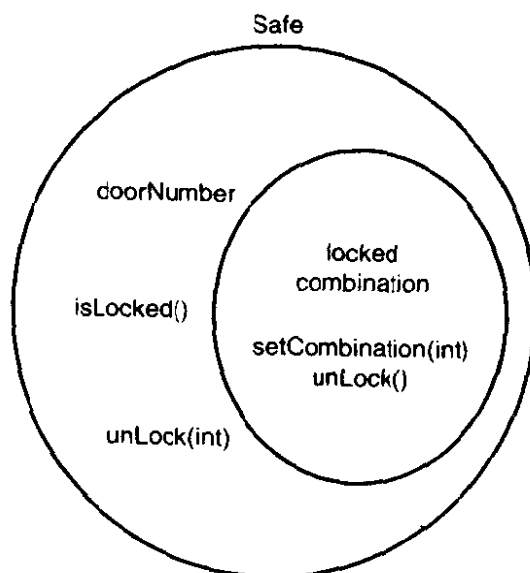


图 11.1 Safe 类示意图

使用 Safe 类

上面的 Safe 类是现实世界中保险箱的一个简单模型。但是这

段代码只是一个类的定义部分，它不能直接运行。你可以用 Java 编译器来编译这个类定义文件：

```
$ javac Safe.java
```

将会产生包含这个类的二进制文件叫 Safe.class，但单独这个文件并无作用，当你用 Java 解释器试图运行它时将返回一个错误信息：

```
In class Safe: void main (String argv []) is not defined.
```

类 Safe 不包含对 extends Applet 的定义，因而它不能在网络主页上作为 applet 来运行，但你可以在其它的类、应用程序或 applet 中使用 Safe 这样的类。

例如，一个程序员可以在一个应用程序中创建和使用 Safe 类型的对象来做一个模拟游戏。在这个游戏中，程序员试图撬开一个保险柜，下面是该应用程序：

```
/* *
Program:      SpyGame Class for application
Purpose:      demonstrates features of class Safe;
@author      john@december.com
@version     1.07; 23 July 1995
*/
class SpyGame {
public static void main (String args []) {
    System.out.println (" SpyGame application...");
    Safe theSafe=new Safe ();
    System.out.println (" Which safe is this? +theSafe.doorNumber);
    System.out.println (" Is the safe looked?" +theSafe.isLocked ());
    /* Try to unlock the safe with a guessed combination of 999; */
        theSafe.unLock (999);
    System.out.println (" Is 999 the combination?" +theSafe.isLocked ());
    /* This won't work; theSafe.unLock ();
        Gives compiler error:" Method void unLock () in class
        Safe is not accessible from class SpyGame." */
    /* How about stealing the combination?
        reference to this won't work; theSafe.combination;
```

```

        Gives compiler error:" Variable combination in class Safe not
        accessible from class SpyGame." */
    /* How about resetting the safe's lock to a know combination?
    This won't work: theSafe.setCombination (999);
    Gives compiler error:" Method void setCombination (int) in class
    Safe is not accessible from class SpyGame." */
    /* How about defacing the door's number? */
    System.out.println (" Changing the door number...");
    theSafe.doorNumber=555;
    System.out.println (" Now the door is number;"
        +theSafe.doorNumber);

    theSafe.unLock (theSafe.doorNumber);
    System.out.println (" Is the safe locked?" +theSafe.isLocked ());
    System.out.println * (" However , you can crack your own safe...");
    Safe mySafe=new Safe (999);
    System.out.println (" Is my safe locked?" +mySafe.isLocked ());
    System.out.println (" My safe has door number" +mySafe.doorNumber)
    System.out.println (" Unlocking safe with combination
        as doorNumber...");
    mySafe.unLock (mySafe.doorNumber);
    System.out.println (" Is my safe locked now?"
        +mySafe.isLocked ( ));
    }
}

$ javac Safe.java
$ javac SpyGame.java
$ java SpyGame
SpyGame application...
Which safe is this? 123
Is the safe locked? true
Is 999 the combination? true
Changing the door number...
Now the door is number: 555
Is the safe lockde? true
However, you can crack your own safe...
Is my safe locked? true

```

```
My safe has door number 999
Unlocking safe with combination as doorNumber...
Is my safe locked now? false
```

在 Java 中使用子类

上面的例子中详细讲述了如何定义一个类以及私有数据成员与公共数据成员的区别。但是只有程序员使用类的继承特性时,面向对象的威力才开始发挥作用。

前面几章中的 applet 例程就是 Java Applet 类的子类,它继承了该类的许多特性。使用子类对父类的继承,程序员可以不用深入了解现有代码而进行强有力的二次开发。多使用继承特性将使得程序员可以用更加灵活协调的方法编程。

类和子类的例程

创建父类需要将现实问题转换为 Java 可以实现的技巧。它需要将程序员所构想的内容转换为相应的数据和函数。上面的保险柜例程是个极好的例子:它几乎包含了所有保险柜的主要特性(从编程的观察角度来说)。

下面这个例程,是模拟戏剧演出的一个片断的类:

```
/* *
Program:      Segment Class
Purpose:      demonstrates a Class;
@author      john@december.com
@version     1.03; 23 July 1995
* /
class Segment {
    private String thePlace=" Unknow";
    private int segmentNumber;
    Segment (int number) {
        segmentNumber=number;
    }
}
```

```

Segment (int number, String aPlace) {
    this (number);
    thePlace=aPlace;
}

public void header () {
    System.out.println (" Segment #" +segmentNumber+" "
                        +thePlace);
}

public void action () {
public void cut () {
    System.out.println (" -----");
}
}
}

```

按这个类的定义，每个片断必须有一个号码，因为它们的构造函数都包含了片断号码。这儿没有一个不包含片断号码的构造函数。如果构造函数判断出某片断没有使用地名则缺省地加上地名“Unknown”。

在第二个构造函数的函数体中第一行 `this (number)` 指向构造函数 `Segment (int number)`。关键词 `this` 指向 `Segment` 类自身，只有一个参数 `int number` 则表明调用满足第一个构造函数。

该类中的其余函数定义了一个片断中可能有的一些动作。

`header()` 宣布片断号码和设置。

`action()` 是一个空函数，不作任何事，但继承 `Segment` 的子类可以实现许多功能。

`cut()` 是用来结束片断的函数。

用 `Segment` 定义的一个对象没有什么大用处，但继承 `Segment` 的子类将会拥有 `Segment` 的全部特性。例如一个动作的片断。

/* *

Program: Action Class

Purpose: demonstrates a child of the Segment Class;

@author john@december.com

```

@version      1.03; 23 July 1995
*/
class Action extends Segment {
    Action (int number) {
        super (number);
    }
    Action (int number, String aPlace) {
        super (number, aPlace);
    }
    public void action () {
        opening ();
        focus ();
    }
    private void opening () {
        System.out.println (" flourish");
    }
    public void focus () {
        System.out.println (" panorama")
    }
}

```

Action 类是 Segment 类的子类，由下面这行定义：

```
Class Action extends Segment
```

Action 的两个构造函数都使用了关键字 super 以使这些构造函数相对于它的父类的构造函数是“落空”的。Action 类增加了一个类的动作函数，调用了两个私有函数 opening 和 focus，你不仅可以用它们创建子类，也可以用它们创建对象，例如下面定义一个 Person 类：

```

/* *
Program:      Person Class
Purpose:      demonstrates a parent Class to create instances;
@author      john@december.com
@version      1.05; 23 July 1995
*/
class Person {

```

```
public static int classLines=0;
public int instanceLines=0;
private String personName=" Unknown";
private String tagLine=" Unknown";
Person () { }
Person (String name) {
    personName=name;
}
Person (String name, String line) {
    this (name);
    tagLine=line
}
public void setTagLine (String line) {
    tagLine=line;
}
public String tagLine ( ) {
    return (tagLine);
}
public void say (String line) {
    System.out.println (personName+": " + " \n" +line+" \n");
    classLines++;
    //this statement increments the class variable classLines by 1
    instanceLines++;
    //this statement increments the instance variable instanceLines by 1
}
}
```

每个 Person 有一个名字和一句特别的“口头禅”(一个人的习惯语言),同时每个 Person 可以说话(使用 say 函数)或改变他的口头禅 (setTagLine); 在这个类的数据部分包含两个变量 classLines 和 instanceLines, 第一个变量被定义成 static 类型, 将它同类本身联系在一起(它可被叫做 class 变量)。这意味着在这个程序中所有的 Person 类的对象都可以“共享”这个变量。另一个变量 instanceLines 没有被定义为 static 类型, 所以它同这个类的每一个“例子”联系起来, 所有 Person 类的对象都拥有一个

独立的变量:instanceLines。可以在 Segment 类的子类中使用 Person 类。

```

/* *
Program:      Dialogue Class
Purpose:      demonstrates an objects of the Person Class;
@author      john@december.com
@version     1.03; 23 July 1995
*/

class Dialogue extends Segment {
    private Person personA;
    private Person personB;
    Dialogue (int number) {
        super (number);
        //this refers to the constructor for the parent class, Segment
    }
    Dialogue (int number, String aPlace) {
        super (number, aPlace);
        //this refers to the constructor for the parent class, Segment
    }
    Dialogue (int number, string aplace,
              String actor1Name; String actor2Name) {
        this (number, aPlace);
        personA=new Person (actor1Name);
        //this creates a person with a given name
        personB=new Person (actor2Name);
    }
    public void action () {
        greetings ();
        talk ();
        wrapup ();
    }
    private void greetings () {
        personA.say (" Hi");
        personB.say (" Hello");
        personA.setTagLine (" Gosh, it is Hot");
    }

```



```
afternoon.header ();
afternoon.action ();
afternoon.cut ();
Action night=new Action (2," city streets");
night.header ();
night.action ();
night.cut ();
}
}
```

这些类可以被编译运行。

```
$ javac Segment.java
$ javac Person.java
$ javac Dialogue.java
$ javac Action.java
$ javac Drama.java
$ java Drama
```

Drama application...

Segment #1 back at the office

Kyle:" Hi"

Georgie:" Hello"

Kyle:" Gosh, it is Hot"

Georgie:" Yup"

Kyle:" Gosh, it is Hot"

Kyle:" Bye"

Georgie:" See ya"

7

4

7

3

Segment #2 city srteets

flourish

panorama

注意, 这个片断的结束部分的数字, 显示了 Kyle 和 Georgie

的两个变量 `classLine` 和 `instanceLine` 的值。

```
System.out.println (personA.classLines);  
System.out.println (personA.instanceLines);  
System.out.println (personB.classLines);  
System.out.println (personB.instanceLines);
```

Kyle 说了四句话, Georgie 说了三句话, Kyle 和 Gerogie 的类变量 `classLine` 被设置成计量该对话场面中一共说了多少句。

Java 语言中, 函数可以被加入一个类中 (像 `Segment`), 而不用重编译它的子类 (除非改变子类以使用新的父函数)。

小 结

本章中的类例程讲述了 Java 程序员构筑强大的 Java 应用程序和 applet 的基本要领, 以及产生并使用类和子类是发挥 Java 面向对象特性的关键之处。

- 类是 Java 程序的基本模块, 类是创建拥有相关的数据和函数的对象的模板。
- 对于其它 Java 类来说, 类中的函数和数据可以在它的定义中用关键字 `private` 隐藏起来, 也可以用关键字 `public` 将其定义为共享。
- 一个类中可以有几个拥有不同参数的同名函数, 这叫重载。
- 子类可以用关键字 `super` 调用它的父类, 也可以用关键字 `this` 调用其自身。
- 可以在构造函数前用关键字 `new` 产生一个类的新对象。
- 一个类中用关键字 `static` 定义的变量可以被程序中该类所有对象共享。
- 在应用程序中使用类和子类可以产生可读性好、可靠性高、可扩展性好的软件。
- 附录 B 包含了更多的 Java 面向对象编程的联机信息的地址。

12 applet 开发工作包含设计和实现

在上面两章中，读者所见的 Java 应用程序演示了 Java 编程的一些基本部分。第10章讲述了 Java 编程在 Java 语法方面需要掌握的基本要领，第11章解释了 Java 的面向对象特性，包括如何使用类创建对象和子类。但作为一个程序员，你要知道得更多：如何使用既考虑到用户界面，又考虑到软件的可靠性与易读性的开发技巧，开发有用的程序。

Java 像其它编程语言甚至人类语言一样，是表述性的，可以用多种组织语言的方式，广泛地选择范围，使得使用 Java 编制 applet 成为一项挑战性的工作。本章阐述 applet 开发，集中讲述设计和实现的考虑过程。在讲述一些通常的 applet 和设计之后，本章剖析了一个完整的 Runnable applet，它使用了多个类，并且举例说明了 Java 设计和实现的要点。

考虑 applet

一个 Java 的 applet 可以是一幅简单的动画，像一个 Duke 从屏幕滚过，或支持一个用户操作的界面，例如电子表格（第1章中提到的）。在设计 applet 过程中，你必需考虑 applet 要做什么，以及它如何在网络上完成工作。一个 Java applet 包含的远不止实现计算和显示的那部分代码，一个 Java applet 同时也是同用户沟通的一种手段，作为 Java 程序员，你必须考虑谁会使用此 applet 以及为什么使用。

在考虑设计一个 applet 时，Java 开发者应注意以下几点：

一、听众

谁是此 applet 的最终听众？

他们为什么使用此 applet？

他们理解此 applet 的知识水平如何？

他们对提供本 applet 的网络使用经验如何？

二、意图

本 applet 支持的动作有哪几种？（这也许包括信息反馈、通信联系和交互性的支持）

使用本 applet 的效果是什么？也许应该包括教育与娱乐。

三、设计要点

本 applet 显示的信息有哪些？本 applet 如何支持或增加更多的文本？

何种风格的用户界面才能更好地适应用户需要？例如，本书早些时候提到的纵横字谜，有种对在纸上玩过类似游戏的用户非常适合的风格，这种风格可以用 applet 对用户理解选择和操作补充。

由 applet 提供的交互操作如何被译成可以理解和相应的小模块？这些模块将影响到类的构成和类的阶层。

设计并不是一个简单的工作，特别是当那儿有太多可能选择时，网络上的超媒体提供极其广泛的可能性，给用户理解与选择。applet 提供了让形式丰富的交互性的构件成为 Web 的一部分的途径。

设计一个 applet

为了讲述如何设计与实现一个 applet，考虑下面的例程要求：

设计一个 applet 模拟一幕在 Web 页面上上演的小话剧，小话剧包括进行对话的演员，以及其于描述的布景。

设计一个符合上述要求的 applet 的步骤如下：

- 定义类：上面描述中的名词一般都用来当做类名。从名词 events, scene, actor, script 即可区分应用程序的主要部分。一个类可以被想象成“可以在其外面画一个盒子的任何东西”，从这个角度说，只要能够将类区分开来，类可以是任何东西。接下来是定义一个类的数据和对数据的操作。通常，描述中的动词用来在类中定义函数名。
- 决定类的关系：图12.1总结了 Scene 包括了一系列 Event，这些 Event 包括 Actor 和 Script。

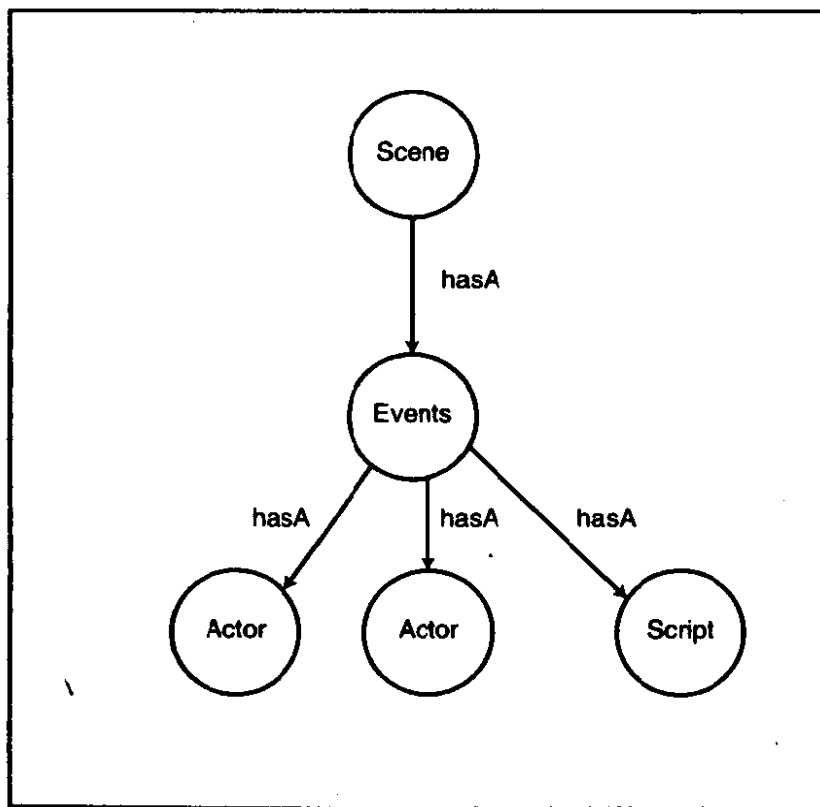


图12.1 类关系图

- 决定类的内容：一个好的类的结构，应当提供良好的概括和特制的能力。概括指将对象中的共同点压缩成一个父类，特制指一个父类加上一些特制的功能产生一个独立的子类。实际上一个类应当同时包括以上两点，这将使编程者不需复制功能即可提供正确的类。
- 决定函数和数据：函数和数据赋予类独一无二的特性，在

先前的例子中，函数判断由一个类产生的一个对象是否可操作，只有 public 函数和数据可由其它类调用。

- Scene: 这个类必须是 Applet 的子类，为了支持一个“动画” Web 页，它必须是 Runnable。它的公共函数包括 Runnable 类的函数，它的私有数据成员包括 Events 类型和 Thread 类型的对象。
- Events: 这个类的对象将被 Scene 类调用，来设计 Scene 中的事件的组成和逻辑关系。公共函数必须包括取得下一事件，并且检查更多的事件，来观察 Scene 是否已结束。私有数据成员包括显示设置，对象表述的一些常量和两个演员对象。
- Actor: Actor 需要产生对话，并且在 Scene 中显示。
- Script: 这个类需要保存这幕戏的标题和对话。

实现 applet

上面设计出的模块将被用来建造一个 applet。Scene 类是一个有趣的“根”类，其它的类支持它。实现 Scene 同第11章中的 Essential Runnable 模式非常相似。

```
import browser. Applet;
import awt. Graphics;
/* *
Program:      Scene class
Purpose:      applet that implements a scene of activity
@author       john@december.com
@version      1. 99; 23 July 1995
* /
class Scene extends Applet implements Runnable {
    static int sceneWidth = 500;
    static int sceneHeight = 500;
    Private Thread flow ;
    Private Events action; //see the class definition for Events
```

```
private boolean done = false;
private boolean pause = false;
public void init () {
    resize (sceneWidth, sceneHeight);
}
public void start () {
    if (flow == null) {
        flow = new Thread (this);
        flow.start ();
    }
    action = new Events ();
}
public void paint (Graphics context) {
    context.clearRect (0, 0, sceneWidth, sceneHeight);
    action.nextEvent ();
    action.paint (context);
    done = (action.moreEvents () == false);
}
public void update () {
    repaint ();
}
public void run () {
    try {
        while (!done) {
            if (!pause) repaint ();
        }
    }
}
public void stop () {
    done = true;
    if (flow.isAlive ()) flow.stop ();
    flow = null;
}
public void mouseDown (int x, int y) {
    pause = !pause;
}
```

Scene 类是整个代码的骨架，它使得主页上的 applet 成为一个 Runnable applet。它的关键函数是 main 函数，它不断循环刷新屏幕显示，直到用户单击鼠标暂停动作。图 12.2 演示了 Scene 类的图形表示，类本身被描述成一个大圆周，它的公共函数和数据在其边缘部分。函数由函数名跟 () 来区分。在大的类圆周中，有另外一个小圆圈，包括类的私有成员。箭头说明类同其它类的关系，Scene 是 Applet 的一个子类，并实现 Runnable 的交互界面。Scene 的私有数据成员包括 flow 对象，它是一个 Thread 对象，以及 action，是一个 Events 对象。

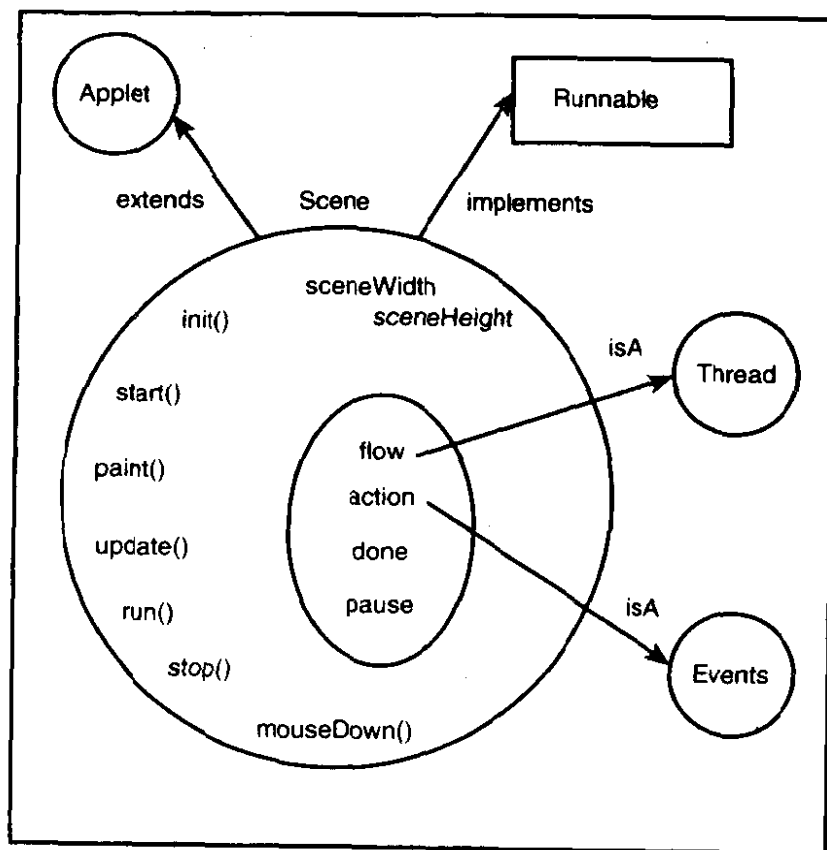


图 12.2 Scene 类

Events 类是一个初始化 scene 并且进一步执行它的主要组织者，它的私有成员包括两个演员和在屏幕安排文本的常量，以及一个 pause 函数。

图 12.3 演示了一个圆形的 Events 以及它与 Script 和 Actor

类的关系。下面是 Events 的实现：

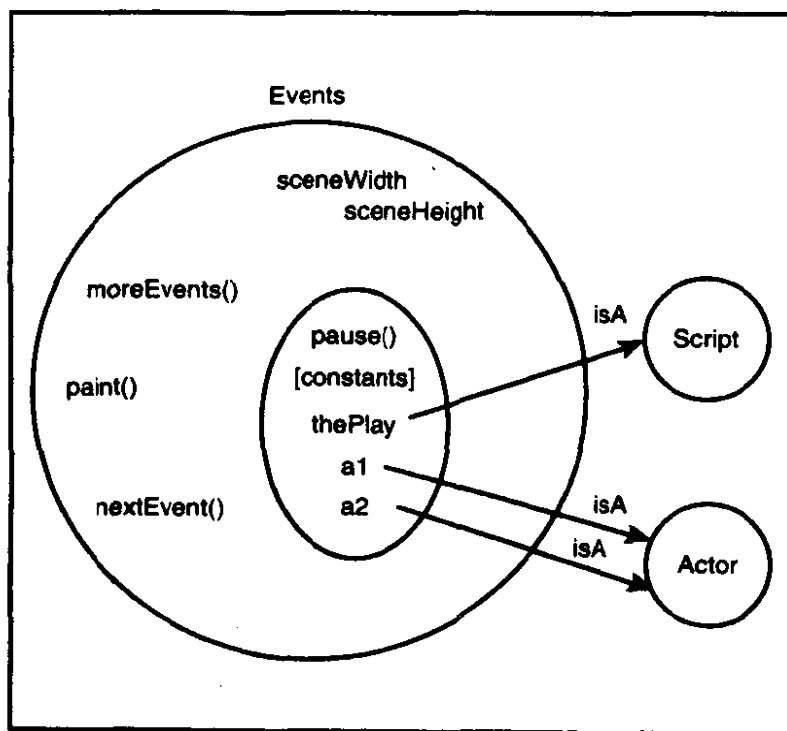


图 12.3 Events 类

```

import awt. Graphics;

/* *
Program:    Events class
Purpose:    a class which executes all the events
@author     john@december.com
@version    1.56;23 July 1995
*/

class Events {
    private final int OVER=999;           //All events over
    private final int MORE=0;             //More events still to come
    private final int HIGHDRAMA=10;       //long,dramatic pauses
    private final int MEDIUMDRAMA=5;     //medium pauses
    private final int LOWDRAMA=1;         //short pauses
    private final int MIDDLE=150;         //stage location
    private final int FRONT=200;          //stage location
    private final int BACK=100;           //stage location
    private final int eventInterval=200;
    private int status=MORE;

```

```

private Script thePlay;
private Actor a1,a2;
Events() {
    // The script and characters are created and placed on the stage.
    thePlay=new Script("The Time Bandits");
    a1=new Actor("Ann",5,100,MIDDLE);
    a2=new Actor("Bob",5,100,FRONT);
}
public boolean moreEvents() {
    return (! (status==OVER));
}
private void pause (int drama) {
    Thread.sleep (eventInterval * drama);
}
public void paint (Graphics context){
    a1.paint(context);
    a2.paint(context);
    pause(MEDIUMDRAMA);
}
public void nextEvent() {
    a1.say(thePlay.getLine (Actor.actorLines));
    a2.say(thePlay.getLine (Actor.actorLines));
    if (thePlay.doneScript (Actor.actorLines))status=OVER;
}
}

```

Actor 类可由一组简单的显示对话及区分状态的函数实现，下面为例程：

```

import awt.Graphics;
/*
Program: Actor class
Purpose: an actor for plays
@author john@december.com
@version 1.51;23 July 1995
*/
class Actor {

```

```
//--Class variable
static int actorLines=0;          //lines all actors have said
//--Private variables
private int lineCount=0;          //number of lines this actor has said
private int stagePosition=0;      //where actor's lines appear
private int leftMargin=0;         //left margin for actor's name
private int lineMargin=0;         //left margin for actor's spoken lines
private String actorName="?"      //actor's stage name
private String nextLine="?"       //the actor's next utterance
//--Constructors
Actor(){ }                        //minimum actor creation
Actor(String name) {
    actorName=name;
}
Actor(String name,int left,int line,int position) { //max actor creation
    actorName=name;
    leftMargin =left;
    lineMargin=line;
    stagePosition=position;
}
//--Methods
public void paint (Graphics context) {
    context.drawString(actorName,leftMargin,stagePosition);
    context.drawString(nextLine,lineMargin,stagePosition);
}
public void setMargins(int left,int line) {
    leftMargin=left;
    lineMargin=line;
}
public void setPosition (int position) {
    stagePosition=position;
}
public int linesSaid() {
    return(lineCount);
}
public void say(String line){
```

```
nextLine=line;  
lineCount++;  
actorLines++;  
}  
}
```

图 12.4 显示了 Actor 类的示意图。

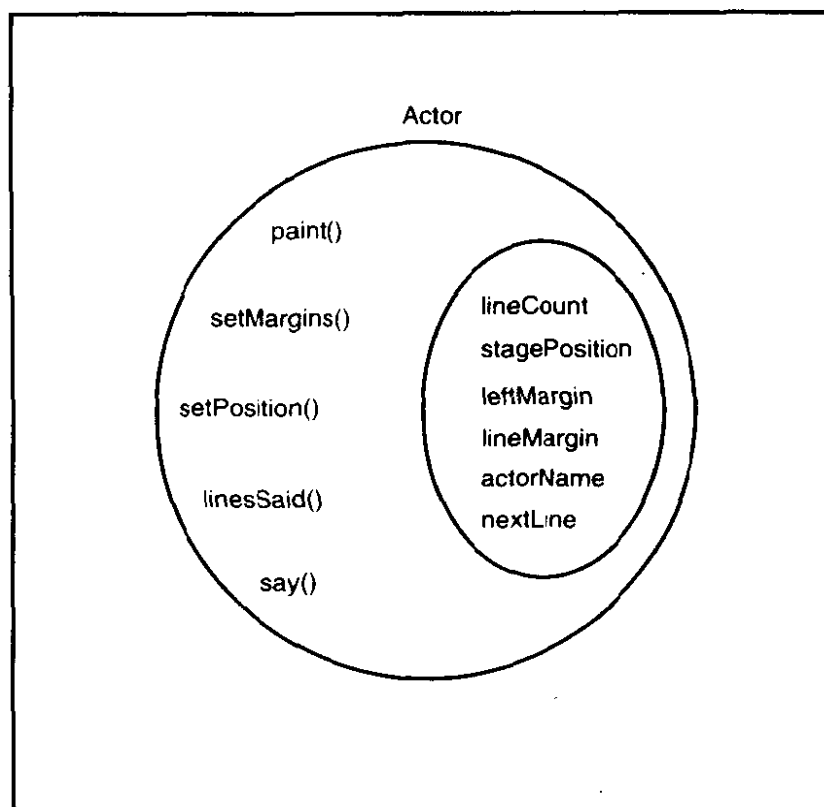


图 12.4 Actor 类

Script 类是非常简单的，它仅仅包含几行对话：

```
/* *  
Program:      Script object  
Purpose:      contains lines for script  
@author      john@december.com  
@version     1.05; 23 July 1995  
* /  
class Script {  
    private String scriptTitle;  
    Script () { }
```

```
Script (String title) {  
    scriptTitle = title;  
}  
  
public boolean doneScript (int line) {  
    return (line == scriptLine.length);  
}  
  
public String getTitle () {  
    return (scriptTitle);  
}  
  
public String getLine (int line) {  
    return (scriptLines [line]);  
}  
  
private String scriptLines [] = {  
    " Hi. ",  
    " Hi there. ",  
    " What time is it?",  
    " Now. ",  
    " This is silly. ",  
    " How much?",  
    " Too much. ",  
    " And so?",  
    " So what. ",  
    " I have to go. "};  
}
```

图 12.5 显示了 Script 的结构。

程序员可以建立下面的 HTML 文件:

```
<HTML>  
<HEAD>  
    <TITLE>Scene Design Applet</TITLE>  
</HEAD>  
<BODY>  
    <P>" This is a scene from a play;" <APP Class=" Scene" >  
</BODY>  
</HTML>
```

程序员接着编译所有的 Java 源文件, 将产生的 .class 文件放

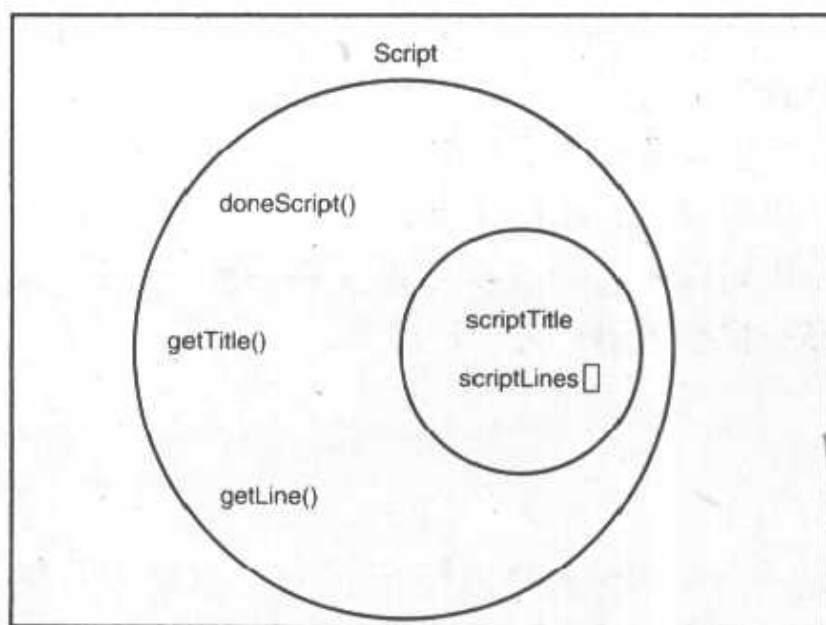


图 12.5 Script 类

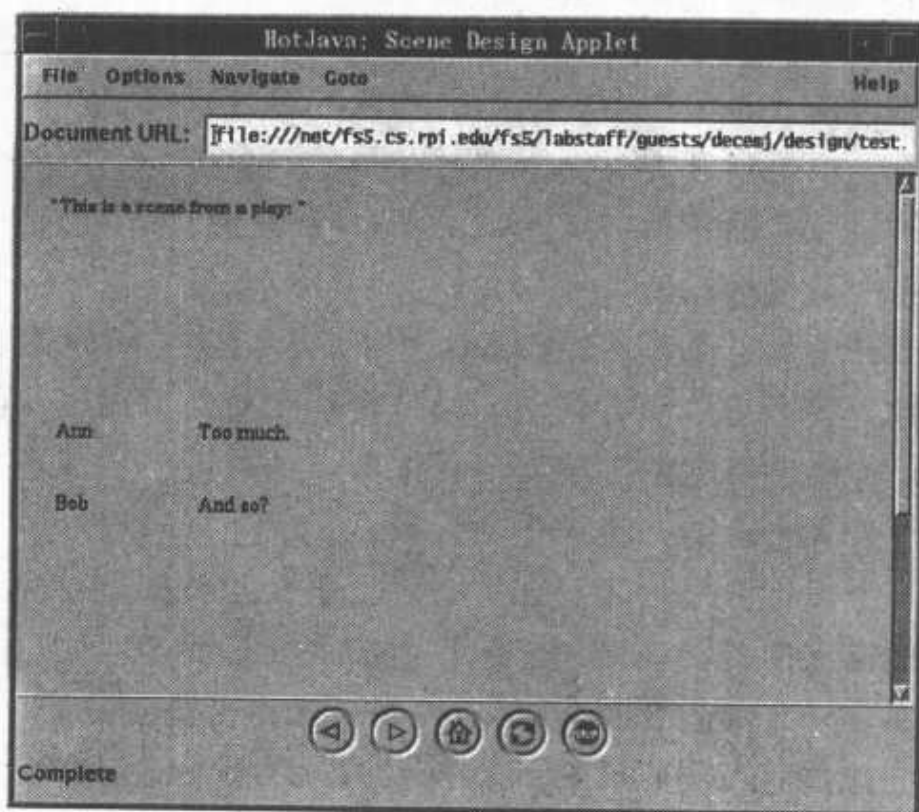


图 12.6 戏剧对话

入一个子目录 classes。这是用来给 HTML 页面定位的。图 12.6

显示了一个运行此 applet 后产生的一个对话框。对话框有一次显示两个演员的各一句对话，直到演员说完所有的对话，最后结果对用户来说，也许是像观看一部字幕电影，但是，这只是用来举例说明设计和实现 applet 的要点。

通过使用超媒体，Actor 类可以变得更精巧，音效文件将被加上，像图形一样给予用户另一种感受。

小 结

Java 是一门面向对象的编程语言，这一点定义了 Java 应用程序的能力，在 applet 使用 Java 类可以像提供计算与信息处理一样提供一个用户交互界面。创建 applet 包括考虑听众要求，设计意图以及由问题核心产生的设计要点。

- 设计一个 applet 的第一步是给出对设计中的应用程序的语言描述，使用名称同问题的现实部分相联系。
- 下一步是定义类，一个类可以是任何东西，但是是一个良好的类应包括概括与特制，对应用程序描述的名词是当作类名的最佳选择。
- 设计中的每个类应能做好某件事，从父类处继承数据和函数，在必须的地方特制，通过使用私有数据成员与函数尽可能地向其它的类隐藏实现细节，这有助于实现更多的不受外界干扰的、模块化的代码。
- 一个设计者最后还要在类中区分函数与数据，对应于程序的描述中的动词是用作函数名的最佳选择。
- 实现一个类包含书写 Java 代码来描述函数与数据。一个类应该是可视的，它的函数和数据可以列出，与其它类定义的关系应该都可以图形化。



进一步的信息资源

本附录提供了很多网上关于 Java Web 和相关课题的信息资源。写这本书时, Java 已结束 alpha 测试, 正准备扩大它的用途。这本附录帮助你查找及更新近来的信息, 以及网上其它参考信息。

Java 信息资源

这一节告诉你专门针对 Java 的信息的链接, 在这里可以找到最新信息以及关于这本书和 Java 的更多信息。

支持本书的 Web

联系关于本书内容的最新消息, 打开 URL <http://www.december.com/works/java.html>, 这支持 Web 提供在网上有关本书、程序举例、源程序修改和相关信息的链接。检查关于修改的反面, 若发现其它错误或有任何问题请通知 John December, Internet 地址: john@december.com。若想和 John December 保持更多联系, 请看 <http://www.december.com/works/>。

Java 主要的书库

这里提供关于 Java 的官方信息, 可得到它的代码发布:

- Sun 的主 Java 节点 (<http://java.sun.com/>): 关于 Java 信息这是第一流的权威资源, 包括 Java/HotJava 代码分布。从地理上讲, 这个服务器位于美国西海岸, 在旧金山湾 (Mountain View, 加州)。这个 Web 站点有一个查寻结构, 这样你能很快找到你所需的 Java 专题的最新信息。你可以

用 click-to-download 结构从页面上选取 HotJava 浏览器最新版本。用于代码分布的匿名 FTP 是 `ftp://java.sun.com/`。

- 在北卡罗莱纳的 SunSITE (`http://sunsite.unc.edu/java/`): 这是 Sun 主要 Java 站点的镜像。这个 SunSITE 位于 Chapel Hill, 北卡罗莱纳。为信息和代码分布的匿名 FTP 是 `ftp://sunsite.unc.edu/pub/language/java`。

在这些 Java sites, 查找:

- The Java FAQ (经常被问的问题) 及答案
- 关于 bugs, ports 和 browsers 的最新信息
- 邮件和讨论名单信息 (Mailing and discussion list information)
- Java 程序员指南 (Programmer guides for Java)
- HotJava 用户指南 (User guides for HotJava)
- 演示程序 (Demonstration applet)

索引

因为 Java 很新, 在 Sun 外很少有 Java 的索引。一旦 Java 变得更流行, Yahoo 索引 (`http://www.yahoo.com/`) 可能成为查找这些消息最好的地方。

- Yahoo 索引为 Computers-languages-Java 的入口 (`http://www.yahoo.com/Computers/Languages/Java/`): 这是关于 Java 信息的列表, 从 Sun 主 Java 站点来的链接, 这个索引最终应包括有关 Java 信息的相关文件官方主页。
- The Wild, Wild World of HotJava (`http://www.science.wayne.edu/~joey/java.html`): 关于 HotJava 浏览器的列表信息, 包括有关移植的最新消息, 这个索引由 Joey Oravec 维护。

Java 联机文献

这是关于 Java 编程语言、HotJava 浏览器和相关技术的主要联机文章出版物:

- Banks, David. "Why Sun thinks HotJava will give you a gift." San Jose Mercury News, March 23, 1995. URL <http://www.sjmercury.com/archives/hotjava.htm>.
- Karpinski, Richard. "HotJava arrives: Sun aims to revolutionize the Web." InteractiveAge, May 22, 1995. URL <http://techweb.cmp.com/ia/15issue/15hotjava.html>.
- Louis, Tristan. "Java on the Menu: Causing Jitters?" WebWeek, June 1995. URL <http://www.mecklerweb.com/ww/news/june-95/paper/df/1-2java.html>.
- Netscape Communications Corporation. "Netscape to License Sun's Java Programming Language." Netscape press release, May 23, 1995. URL <http://home.netscape.com/newsref/pr/newsrelease25.html>.
- O'Connell, Michael. "Java, The Inside Story." Sun World Online, July 1995. URL <http://www.sun.com/sunworldonline/swol-07-1995/swol-07-java.html>.
- Sullivan, Eamonn. "Sun's Java technology perks up WWW." PC Week Online, June 5, 1995. URL http://www.ziff.com:8002/~pcweek/reviews/june_1995/java.html.
- "Sun Brings True Interactivity to The World Wide Web." Sunflash, 77. 25, May 1995. <http://www.flashback.com/may.1995/sunflash/77.25.www.html>.
- "Sun Launches First Interactive WWW Site Powered By Java." Sunflash, 77. 27, May 1995. <http://www.flashback.com/www/may.1995/sunflash/77.27.java.html>.

关于 Java 的信息资源

你可以通过看 Java 使用的例子和面向对象程序设计这一类书来学会 Java 语言,这节告诉你使用 Java(本书曾讨论过)的一些组织,和面向对象的信息。

使用 Java 的组织

- Accurate Information System (<http://accurate.com.my/>):该组织在它的主页中使用 Java。
- Dimension X (<http://www.dimensionx.com/>): 一个 bleeding-edge Internet 技术发展者,看 Iced Java 工作,3D rendering engine。
- George Coates Performance Works(<http://www.george-coates.org/>):使用数字技术、电影和舞台技巧的戏剧表演的制片商。
- HotWired(<http://www.hotwired.com/java/>): Wired 杂志的电子版在它用户界面上使用 Java。
- The Nando Times(<http://www.nando.net/>):在它的界面上使用 Java。
- Netscape Communications (<http://home.netscape.com/>)。这个公司计划在它的 Web 浏览器上许可使用 Java。
- Odegard labs(<http://www.odegard.com/>):这是个技术发展者基于它的 Web 中的使用。
- The Rolling Stones(<http://www.stones.com/>):这个摇滚组合在许多页上用了 Java,包括 Java-ized Voodoo Lounge(见第 4 章)。
- Sun Microsystems(<http://www.sun.com/>):Java 的开发者在 Web 页面上有许多程序例子,特别是你可用可执行

Java 语言的浏览器检查当你将光标越过图标时封面页上的图标如何变化。

- VPRO (<http://www.vpro.nl/>): 来自荷兰 Hilversum 的公众广播业, VPRO 使用 Java 作为 Web 的 experimental 的一部分(见第 6 章)。

面向对象信息来源

- 面向对象信息资源索引(<http://cuiwww.unige.ch/OSG/OOinfo/>): 这是一个查寻多种面向对象信息资源的索引, 包括研究团体、文档、公司、书籍和提要。
- 面向对象设计的联机指导书(<http://www.clark.net/pub/howie/OO/oo-home.html>): 这是面向对象设计的在网上信息的指南, 这是由 Howie Michalski, Lead Database Engineer Infrastructure, CompuServe, Inc 设计的。
- 来自 yahoo 的计算机—软件—面向对象程序设计的章节(<http://www.yahoo.com/Computers/Software/Object-Oriented-Programming/>): 连结各种面向对象信息资源的总索引。

一般 Web 信息

这里有一些 WWW 信息资源, 你可以用它们找到关于 Web 的信息或在 Java 的浏览器上看新程序:

- Web overview/W³C (<http://www.w3.org/>): 对 Web 的浏览来自 WWW Consortium。
- Web Info/EARN (<http://www.earn.net/gnrt/www.html>): 描述性介绍并解释来自 EARN 的 Web。
- Web Guide/Hughes (<http://www.eit.com/web/www.guide/guide.doc.html>) “进入 www Cyberspace 的指导” Kevin Hughes, 还可以看 Hughes 主页 “From Webpace to

Cyberspace”在 <http://www.eit.com/~kevinh/cspace/>。

- Web-Yahoo (<http://www.yahoo.com/Computers/World-Wide-Web>) 来自最流行 Yahoo 索引的计算机—WWW 部分。

Web FAQ

- Web FAQ/Boutell (<http://sunsite.unc.edu/boutell/faq/www-faq.html>) : FAQ 列表和有关 Web 的答案, 这个 FAQ, 由 Thomas Boutell 维护, 包括用户、提供者和一般信息。
- Web FAQs/W³C (<http://www.w3.org/hypertext/www/FAQ/List.html>): 关于 W³ 的 FAQ, 由 Tim Berners-Lee 维护。

访问 Web

- Web via e-mail (<mailto:agora@mail.w3.org> Body: www URL) 你可通过 e-mail 得到 Web 文件, 有个建议, 可向 agora@mail.w3.org 请求帮助。
- Web via telnet (<telnet://telnet.w3.org>): 可通过用 telnet 访问 Web, 这是个 www Consortium 计算机使用 Web 的例子:
[telnet.w3.org](telnet://telnet.w3.org)
- Web ftp info (<ftp://ftp.w3.org/pub/www/>): 来自 CERN 的关于 Web 信息文件, 包括论文、指南和标准草案。
- Web gopher info (<gopher://gopher.w3.org>): 通过 gopher 取得关于 Web 的信息文件。
- Bootstrap info ([http://www.w3.org\(hypertext\)/WWW/FAQ/Bootstrap.html](http://www.w3.org(hypertext)/WWW/FAQ/Bootstrap.html)): 关于如何学习 Web 和进阶的信息。

Web 软件

这有一些查找 Web 浏览器和软件的地址：

- Web clients (<http://www.w3.org/hypertext/WWW/Clients.html>): 程序 (Web 浏览器) 列表, 来自 CERN, 可帮你找到 Web。
- Web Browsers/yahoo (http://www.yahoo.com/Computers/World_Wide/Web/Browsers/): Yahoo 的列表, 计算机—WWW—浏览器。
- Web Browsers source (<ftp://ftp.w3.org/pub/www/bin/>): Web 浏览器不同硬件平台的源代码。
- Web servers (<http://www.w3.org/hypertext/WWW/Daemon/Overview.html>): CERN 的程序 (Web 服务器) 的列表帮你提供 Web 上的信息。
- EIT WSK (<http://wsk.eit.com/wsk/doc/>): Enterprise Integration Technologies Corporation's Webmaster's Starter Kit, 帮你安装 Web 服务器和可选项的资源。
- Web software (<http://www.w3.org/hypertext/WWW/Status.html>): 关于 Web 的软件产品列表。

发展中的 Web 信息

- Web Vlib (<http://www.stars.com/Vlib>): Web virtual library 的发展章节, 为 Web 信息提供者 and 使用者收集的资源, 这个信息包括一般信息和与多种资源的连接。
- HTML Writer's Guild (<http://www.mindspring.com/~guild/>): HTML 作者和 Web 发展者的组织。

一般 Internet 信息

- 信息资源: Internet 和计算机媒质通信 (<http://www.rpi.edu/Internet/Guides/decemj/icmc/top.html>): 关于 Inter-

net 信息资源的列表, 包括技术、应用、文化、论坛和书籍收藏家。

- Internet Web Text (<http://www.rpi.edu/Internet/Guides/decemj/text.html>): Internet 方向、研究、标志和其它资源的列表。
- Yahoo-Internet (<http://www.yahoo.com/Computers/Internet/>): 来自 Yahoo 包括许多题目的广泛范畴的索引。


```
class Safe {  
    public      int      doorNumber = 123;  
    private     boolean   locked     = true;  
    private     int      combination = 456;  
    public boolean isLocked () {  
        return (locked);  
    }  
    public void unLock (int thisCombination) {  
        if (thisCombination == combination) unLock ();  
    }  
    private void unLock () {  
        locked = false;  
    }  
    private void setCombination (int setting) {  
        combination = setting;  
    }  
    Safe () { }  
    Safe (int door) {  
        doorNumber = door;  
        setCombination (doorNumber); // this is the trick to craking a safe  
    }  
}
```

程序设计者可以通过在类描述中使用关键字 `extends` 定义子类。

```
class StrongBox extends Safe {  
    /* 这里是子类定义的其余部分 */  
}
```

子类继承公有函数和父类的数据。和类同名的函数被称为构造函数。类可以有很多同名的函数，只要变量表中变量类型、变量个数或变量类型的顺序有所不同。

关键字 `this` 是类定义中指向自己的指针。子类中关键字 `super` 是父类的指针。

应用程序

Java 程序由 Java 解释器 (java) 执行, 独立于 HotJava 浏览器之外。HotJava 浏览器本身也是一个 Java 应用程序。

这里有一个简单程序, 用来创建和控制类型为 Safe 的对象:

```

/ * *
Program:          SpyGame Class for application
Purpose:          demonstrates features of Safe Class;
@author          John@december.com
@version         1.08a, 23 July 1995
* /
Class SpyGame {
    public static void main (String args []) {
        Safe mySafe = new Safe (999);
        System.out.println (" Is my safe locked? " + mySafe.isLocked ());
        System.out.println (" My safe has door number"
                               + mySafe.doorNumber);
        mySafe.unLock (mySafe.doorNumber);
        System.out.println (" Is my Safe locked Now? "
                               + mySafe.isLocked ());
    }
}

```

applet

Java applet 是专门在 HotJava 浏览器 (java) 上运行的程序。一个 applet 是 Java 类 Applet 的一个子类, 这里是一个很小的源文件名为 HelloWorld.java 的 applet:

```

import browser.Applet;
import awt.Graphics;
Class EssentialApplet extends Applet {
    public void init () {
        resize (600, 300);
    }
}

```

```

    public void paint (Graphics context) {
        context.drawString (" Hello, world!", 50, 100);
    }
}

```

applet 由 Java 编译器 (javac) 编译, 产生一个类文件 (例如 HelloWorld.class)。

applet 包含在一个使用 APP 的 HTML 文件中, 这有一个关于 APP 元素的一般语法

```

<APP Class      = " 类名"
    Src          = " URL 或包含类目录的目录路径"
    Align        = " bottom | middle | top"
    Width        = " 以点阵为单位的宽度"
    Height       = " 以点阵为单位的高度"
AppletSpecificAttribute = " aValue"
>

```

这是一个包括 HelloWorld applet 的 HTML 文件的例子:

```

<HTML>
<HEAD>
    <TITLE>First Hello Applet</TITLE>
</HEAD>
<BODY>
    <P>" This is it!" <APP Class=" EssentialApplet" >
</BODY>
</HTML>

```

这个类文件 HelloWorld.class 在一个称为 classes 的子目录下, 这个子目录在 HTML 文件目录里。

接 □

接口是函数体尚未完成的函数的集合。类可以使用关键字 implements 后面跟上接口名, 这样单个的类就可以完成函数体。

例如:

```

public interface Commerce {
    void public takeOrder (Customer c);
}

```

```

        void public thank (Customer c);
    }

    public class Store implements Commerce {
        void public takeOrder (Customer c);
            fulfill (person. desire);
    }

    void public thank (Customer person) {
        person. accept (say (" thanks"));
    }
}

```

包

包是类和/或接口的集合。它用来在 Java 程序组织名字空间。包在源文件中作为第一个非提示、非空白的关键字由关键字 `package` 识别：

```
package packageName;
```

开头没有 `package` 语句的文件放入缺省包。

通常情况下，包常常由作者机构的前缀集命名。在前缀之后，名字包含由句点分隔的区别字。

在其他文件中，已命名的包可以引入名字空间：

```
import packageName;
```

在一个类中，可以使用 `.` 引用严格指明包中的函数名以避免名字二义性。

考虑一个引入语句：

```
import webex. project. * ;
```

名字以 `webex. project` 开头的包集合或许有同样的函数名，这样可以使用 `.` 引用。例如，如果包 `webex. project. alpha` 和 `webex. project. beta` 都有一个类 `List`，那么可以指出它们：

```
Orders thursday = new webex. project. alpha. List ();
```

```
Orders friday = new webex. project. beta. List ();
```

包编译后的类文件以目录结构组织，这种目录结构同时也使用 `.` 引用指明包。

Java alpha 版的漏洞**提示**

如果在类的子目录重启 HotJava, HotJava alpha 版一个漏洞使得浏览器不能识别对 applet 的修改。在 Sun Microsystems 节点 <http://java.sun.com/> 检查这个漏洞的最新情况。

协议处理器

在一个叫 handler.java 的文件中为处理器建立一个名为 handler 的类代码, 这个文件在名为 net.www.protocol.name 的包中, 这里 name 是新协议的名字。

编译这个处理器, 把它放在类目录的 net/www/protocol/name/子目录, 类目录由环境变量 CLASSPATH 指定。

重启浏览器后, 将识别 run: URLs。

内容处理器

在一个叫 subtype.java 的文件中为处理器建立一个名为 subtype 的类代码, 这个文件在名为 net.www.content.type 包中, 这里 type 是新格式的 MIME 类型 (参见本附录末尾的 MIME 类型表), subtype 是子类型。例如, 类型可能是 text, 子类可能是 plain。

编译源码, 把它放在类目录的 net/www/content/type/子目录, 类目录由环境变量 CLASSPATH 指定。

重启浏览器后, 将能解释格式类型为 subtype 的文件。

出 错

出错是指 Java 程序中可能引起程序中断的错误。程序员可以使用 Java 的出错处理程序使程序正常中止或恢复。

Java 语言有一些自带的出错处理; 查看最新的语言定义可以得到它们的列表。程序员可以定义一个新的出错处理。

```
class ProgrammerDefinedException extends Exception {  
    /* 在这里定义 */  
}
```

Java 程序员使用 try...catch 语句把出错处理语句放入代码:

```
try {  
    result = sum / 0;  
} catch (ArithmeticException exception) {  
    System.out.println (" can't divided by 0!");  
} catch (ProgrammerDefinedException exception {  
    System.out.println (" Some other error");
```

Java 语言信息

当你用 Java 编程时,必须遵守语言规范。这个部分总结了 Java 语法的主要信息。

最新信息

在 Sun 的 Java 节点可以得到 Java 的最新规范信息。可以检查 <http://java.sun.com/documentation.html> 的文档列表。从 Sun 的主 Java 节点可以得到 alpha 3 版本的 Java 规范: http://java.sun.com/1.0alpha3/doc/javadoc/javadoc_1.html。

还可以找到所有 Java 软件的手册页面,例如, alpha 3 手册页面可以在下述地址得到:

<http://java.sun.com/1.0alpha3/doc/man/index.html>

作为合法性检查服务的一部分 (<http://www.halsoft.com/html-val-svc/>), HalSoft 提供了 HotJava Data Type Definition (数据类型定义, DTD), 在 <http://www.halsoft.com/sgml/HotJava/DTD-HOME.html>。这对检查 HTML 文件的适当 applet 语法是有益的。

Java 保留字

abstract	boolean	break	byte
byvalue	case	catch	char
class	const	continue	default
do	double	else	extends
false	final	finally	float
for	goto	if	implements
import	instanceof	int	interface
long	native	new	null
package	private	protected	public
return	short	static	super
switch	synchronized	this	threadsafe
throw	transient	true	try
void	while		

允许字符

Java 程序的字符可以是任何 Unicode 字符，只要不是保留字符。Unicode 是一种字符编码标准，它能支持世界上多种语言。关于 Unicode 的更多信息请看 <http://www.stonehand.com/unicode.html/>。

保留字符

这些字符用来作为操作符或分隔符，因此不能在变量名中使用：

+ - ! % ^ & * | ~ / >

() { } [] ; ? : , . =

下列字符组合用来作为操作符：

++ -- == <= != << >>

>>> += -= *= /= &= |=

●逻辑型

* boolean 取值为 true 或 false

●数组

* 以 [] 为标志; 用 new 创建。例如:

```
int MyNumbers [] = new int [40];
```

Java 操作符

操作符	用途	例子
[]	数组元素值	scores [42]
()	组操作符	counter + (2 * 3)
++	加 1	counter++ (使用, 然后加)
++	加 1	++counter (加, 然后使用)
--	减 1	counter-- (使用, 然后减)
--	减 1	--counter (减, 然后使用)
!	逻辑非	!done (如果 done 为 false, 其值为 true)
~	位求补	~register (把二进制表示中的 0 变成 1)
instanceof	是某类的一个对象	mySafe instanceof Safe
*	乘	4 * 3 是 12
/	整除	7/2 是 3
%	求余 (整除的余数)	35%3 是 2
+	加	42+3 是 45
-	减或改变符号	42-3 是 39
<<<	左移	<<<register
>>>	右移	>>>register
>>>>	填 0 右移	>>>>register
<	小于	3<42 为真
>	大于	3>42 为假
<=	小于或等于	42<=42 为真
>=	大于或等于	45>=45 为真
==	等于	counter==42
!=	不等于	counter!=MAXIMUM
&	逻辑与	a&b (a 和 b 必须都为真)
^	异或	a^b (布尔量 a 和 b 只能一个为真)
	或	a b (a 和 b 必须有一个为真, 可以同时为真)
&&	逻辑与	a&&b (a 和 b 必须同时为真)
	逻辑或	a b (a 和 b 必须有一个为真, 可以同时为真)

续表

操作符	用途	例子
?:	如果, 则	<code>done? say("bye"):say("continue");</code>
=	赋值	<code>counter = 42;</code>
op =	操作符赋值缩写	<code>counter += 42;</code> 把 counter 加 42

MIME 类型

RFC 1521 和 RFC 1522 定义了多功能 Internet 邮件交换格式 (MIME) 的类型和子类型。参见:

<http://ds.internic.net/rfc/rfc1521.txt> 和 <http://ds.internic.net/rfc/rfc1522.txt> 以及 <ftp://ftp.isi.edu/in-notes/iana/assignments/media-types/>。

内容类型的 x-前缀意味着没有被考虑成标准类型, 可以被其他用户另外定义。

类型	子类型	典型文件扩展名
text	enriched	
	html	html htm
	plain	txt text
	tab-separated-values	
	richtext	
multipart	alternative	
	appledouble	
	digest	
	header-set	
	mixed	
message	parallel	
	external-body	
	news	

续表

类型	子类型	典型文件扩展名
application	partial	
	rfc822	
	activemessage	
	andrew-inset	
	applefile	
	atomicmail	
	commonground	
	cybercash	
	dca-rft	
	dec-dx	
	eshop	
	iges	
	mac-binhex40	hqx
	macwriteii	
	mathematica	
	msword	doc
	news-message-id	
	news-transmission	
	octet-stream	tar dump readme bin uu exe
	oda	oda
	pdf	pdf
	postscript	ps eps ai
	remote-printing	
	riscos	
	rtf	rtf
	slate	
	wita	

续表

类型	子类型	典型文件扩展名
	wordperfect5.1	
	x-dvi	dvi
	x-pdf	pdf
	x-tar	tar
	x-tex	tex
	x-www-form-urlencoded	
	x-www-pgp-request	
	x-www-pgp-reply	
	x-www-local-exec	
	zip	zip
image	gif	gif
	ief	ief
	jpeg	jpeg jpg jpe
	rgb	rgb
	tiff	tiff tif
	xbm	xbm
	xpm	xpm
	x-xwindowdump	xwd
	x-pict	pict
audio	basic	au snd
	x-aiff	aif aiff aifc
	x-wav	wav
video	mpeg	mpeg mpg mpe
	quicktime	qt mov
	x-msvideo	avi
	s-sgi-movie	movie

C HTML 标记参考

本附录提供了 HTML 标记的快速参考。注意 HotJava 浏览器不支持表格标记或诸如 Netscape 扩展的背景图形和颜色等 HTML 扩展。检查流行的支持 Java 的浏览器的最新文档, 这样用户就知道它支持哪些 HTML 标记。

文档结构和内容标记

<code><HTML></HTML></code>	HTML 文档的开始和结尾
<code><HEAD></HEAD></code>	文档内容信息开始和结束
<code><TITLE></TITLE></code>	文档标题, 在 HEAD 里
<code>< BASE Href=" URL" ></code>	文档的基本参考 URL, 在 HEAD 里
<code>< BODY></BODY></code>	浏览器显示的文档内容
<code><!--comment--></code>	注释, 浏览器不显示

超 链

<code>Hotspot< /A></code>	使文档 URL 链接 Hotspot 文本的超链
<code>< A Name=" Jump" >Text</code>	与 Text 相连的文本的名字的超链
<code>< A Href=" URL # Jump" > Hotspot</code>	跳到 URL 文档中名字为 Jump 的超链 的超链

文 本 块

<code><BLOCKQUOTE></code>	文本块定义
<code></BLOCKQUOTE></code>	
<code>< UL>< /UL></code>	非排序列表(列表项以为标志)
<code>< OL></code>	有序列表 (列表项以为标志)
<code><MENU></ MENU></code>	菜单列表 (菜单项以为标志)
<code>< DIR></ DIR></code>	目录列表 (目录项以为标志)

<code><DL></DL></code>	定义列表(术语以 <code><DT></code> 为标志,定义以 <code><DD></code> 为标志)
<code><ADDRESS></ADDRESS></code>	地址,常用来作为作者标识

分隔符

<code><H1></H1><H6></H6></code>	从1级标题(最大)到6级标题(最小)
<code><HR></code>	水平线
<code>
</code>	拆行
<code><P></code>	段落开始(结束段落时, <code></P></code> 可选)

实际字符格式

<code></code>	黑体
<code><I></I></code>	斜体
<code><TT></TT></code>	打字机字体
<code><PRE></PRE></code>	预排文本,保留折行

逻辑字符格式

<code></code>	突出
<code></code>	强烈突出
<code><KBD></KBD></code>	键盘输入
<code><CITE></CITE></code>	引用
<code><VAR></VAR></code>	变量名
<code><SAMP></SAMP></code>	文本示例

图 像

<code></code>	在文档中 URL 的图像
<code></code>	对非图形浏览器显示字符代替图像
<code></code>	设置图像之后文本的位置

表 单

<code><FORM Action=" URL" Method =" get post" ></FORM></code>	使用 URL 的一个 gateway 程序和某种方法(得到数据)的表单
---	-------------------------------------

<INPUT Name = " name" Type = " checkbox hidden image password radio reset submit text" >	用一个特定的类型和特征名 name 输入元素
<TEXTAREA Name = " name" Rows=" R", Cols=" C" >	文本区(可编辑文本的各行),其特征名为 name,同时有 R 行和 C 列可见
<SELECT Name = " name" Size = " N" Multiple" >	用特征名 name 选择元素,同时 N 个选择可见,选择用 OPTION 定义
<OPTION Value=" string" >	在 SELECT 中使用的可选元素,返回值为 string

表 格

<TABLE Border></TABLE>	开始、结束有边框表格
<CAPTION></CAPTION>	开始、结束标题
<TR></TR>	开始、结束一行
<TH Colspan=" C" Rowspan=" R" ></TH>	开始、结束一个跨过 R 行 C 列的表头
<TD Colspan=" C" Rowspan=" R" ></TD>	开始、结束一个跨过 R 行 C 列的数据表元

Netscape 颜色扩展

<BODY Background=" URL" >	把文本背景设置为 URL 的图像
<BODY Bgcolor=" #RRGGBB" >	设置背景为给定 16 进制红绿蓝值 (RR, GG, BB)
<BODY Text=" #RRGGBB" >	设置文本颜色
<BODY Link=" #RRGGBB" >	设置未访问链接的颜色
<BODY VLink=" #RRGGBB" >	设置已访问链接的颜色
<BODY ALink=" #RRGGBB" >	设置激活的链接颜色

参见:

- HTML 总结信息 <http://www.rpi.edu/~decemj/pages/intro.html>
- WWW 论坛 <http://www.w3.org/>
- WWW FAQ http://sunsite.unc.edu/boutell/faq/www_faq.html
- Web 开发者图书馆 <http://www.stars.com/Vlib/>

- HTML 作者手册 <http://www.mindspring.com/~guile91>

关于 Web 发展的更多信息，请看“HTML and CGI Unleashed”，这本书由 John December, Mark Ginsburg 和各章的合作者创作。Indianapolis: Sams. net Publishing, 1995. ISBN 0-672-30745-6。

关于 World Wide Web 的更多信息，请看“The World Wide Web Unleashed”，这本书由 John December, Neil Randall 和各章的合作者创作。Indianapolis: Sams. net Publishing, 1995. ISBN 0-672-30737-5。

关于这些书的更多信息，请看书籍支持节点 <http://www.rpi.edu/~decemj/works/wdg.html> 和 <http://www.rpi.edu/~decemj/works/wwwu.html>。

无忧书库书籍版权申明

无忧书库提供的所有书籍、资料版权归原作者和出版商所有。

本站提供下载的书籍仅供个人学习、参考之用，任何集体、个人不得用于商业用途。

请您在下载书籍 24 小时之后删除书籍，购买正版书籍！

本站书籍如有侵犯您的版权，请与我们联系，我们将尽快删除。联系信箱：
pcbook@51soft.com。

无忧书库
<http://pcbook.51soft.com>