



JAVA

语言一月通—— 应用编程及网际通信

主 编 著

科技大学出版社



JAVA 语言一月通

——应用编程及网际通信

陈玉生 编著

电子科技大学出版社

JS/87/24
内 容 简 介

本书在介绍 Java 面向对象编程概念和语法的基础上,详细阐述了用 Java 语言编程的多个方面,包括:Applet 编程、AWT 的使用、例外和线程、动画、Application 编程、流操作及网络联接等,重点介绍了可在 WWW 网中运行的 Applets 编程技术。

本书深入浅出、图文并茂,书中采用了大量详细的编程实例,并配以生动的图片。同时,对用 Java 编程的一些高级概念进行了分析,可以帮助已有一定 Java 编程基础的读者作进一步提高。本书共分十三章,适用于想轻松学习 Java 的读者,用本书在一月内,每天花两小时学通 Java 语言,本书既可作为大专院校的教材,也可供广大自学者使用。

JAVA 语言一月通——应用编程及网际通信

陈玉生 编著

*

电子科技大学出版社出版
(成都建设北路二段四号) 邮编 610054

四川省郫县唐昌印制厂印刷

新华书店经销

*

开本 787×1092 1/16 印张 15.25 字数 372 千字
版次 1997 年 2 月第一版 印次 1997 年 2 月第一次印刷
印数 1—6000 册

ISBN 7-81043-326-1/TP·119

定价: 20.00 元

前 言

作为一种新型的计算机语言，Java 语言在近年来正经历着突飞猛进的发展，它与传统的 C 语言类似，但又有其独特的风格，由于用它编写的程序具有网络操作安全性和跨平台工作等诸多优越性，使之可用于面向 Internet 网络的程序开发，并很有可能成为新一代的网络计算机标准语言。

Java 语言与 C 语言类似，但也有它们不同的地方，如面向对象编程、指针等方面，本书在讲到 Java 的这些方面内容时也提示了它与 C 语言的不同，这样对已有 C 语言编程经验的读者可更容易理解和掌握 Java 语言。

本书深入浅出，为便于初学者学习，通过大量的编程实例向读者展示 Java 语言的各种概念和用途，同时还配以丰富而生动的图片，使得对这一门新型计算机语言的学习不再是件难事，这些实例稍加修改即可再用于读者的程序中去，相信本书会成为读者学习 Java 语言的一本难得入门书、参考书。

全书共分十三章，它们是：Java 入门、基本语法、面向对象的程序设计、方法、Applet 入门、Applet 编程基础、高级抽象窗口工具箱 AWT、例外、线程、动画创作、Java Application 创建及运行、窗口、输入输出流及网络联接。

读者用本书，每天花两小时即可轻轻松松在一个月內掌握用 Java 语言编程，对熟悉 C++ 等面向对象编程语言的读者来说，可在更短的时间内学习完全书的内容。

舒力、彭明、胡渝蓉、周玲、林俭、李京红、蒋东等同志为本书收集资料、审校软件、腾写和复审书稿作了大量工作，在此表示谢意。

编 者

1996 年 10 月

于电子科技大学

目 录

第一章 Java 入门

第一节 什么是 Java	(1)
一、起源	(1)
二、Java 语言风格	(2)
三、Java 语文特点	(3)
第二节 Java 与 Internet	(6)
一、即插即用的 Applets	(6)
二、安全性	(10)
第三节 程序运行及实例	(12)
一、编程工具	(12)
二、创建一 Java Application	(17)
三、创建一 Java Applet	(18)

第二章 基本语法

第一节 语句	(21)
一、注释	(22)
二、块语句	(22)
三、空语句	(23)
四、标号语句	(23)
五、表达式语句	(24)
六、条件语句	(24)
七、循环语句	(25)
八、跳转语句	(25)
第二节 字面	(25)
一、整型字面	(26)
二、浮点字面	(26)
三、布尔字面	(27)
四、字符字面	(27)
五、字符串字面	(28)
六、原型类型数据间的转换	(29)
第三节 变量	(30)
一、数据类型	(30)
二、变量定义	(32)
第四节 表达式和操作符	(33)

一、运算	(33)
二、再论赋值	(34)
三、比较	(36)
四、逻辑运算	(37)
五、位元操作	(37)
六、算子优先级	(38)
七、字串串运算	(39)
第五节 流程控制	(40)
一、? 算子	(40)
二、switch 语句	(40)
三、for 循环	(42)
四、while 循环	(43)
五、do...while 循环	(43)
六、跳出循环	(44)
第六节 数	(46)
一、定义数组变量	(46)
二、创建数组对象	(46)
三、对数组元素访问及改变	(47)
四、多维数组定义及访问	(48)

第三章 面向对象的程序设计

第一节 对象与类	(49)
一、行为与属性	(50)
第二节 继承	(53)
一、等级创建	(56)
第三节 接口 (Interface)	(57)
第四节 包	(62)
第五节 访问控制符	(64)
一、public	(64)
二、package	(65)
三、protected	(66)
四、private	(67)
第六节 类操作	(67)
一、类创建	(67)
二、类中变量	(68)
第七节 对象操作	(70)
一、对象创建	(70)
二、类变量和实体变量的操作	(73)
三、对方法的调用	(75)

四、对象类型的转换	(77)
五、对象的比较	(79)

第四章 方法

第一节 方法的定义	(81)
一、关键词 <i>this</i> 和 <i>super</i>	(82)
二、变量域	(84)
第二节 参数传递	(84)
一、方法修饰	(86)
第三节 重载与覆盖	(87)
一、方法重载	(87)
二、方法覆盖	(88)
第四节 构造与 <i>Finalize</i> ()	(91)
一、构造	(91)
二、重载构造	(92)
三、 <i>finalize</i> ()	(93)

第五章 APPLETs 入门

第一节 Applets	(94)
一、 <i>init</i> ()	(95)
二、 <i>start</i> ()	(95)
三、 <i>stop</i> ()	(95)
四、 <i>destroy</i> ()	(96)
五、 <i>paint</i> ()	(96)
第二节 嵌入 WWW 页面	(97)
一、使用 <i><Applet></i> 标签	(97)
二、 <i><Applet></i> 标签属性	(98)
第三节 参数传递	(101)
一、 <i><PARAM></i> 标签	(101)
二、 <i>getParameter</i> ()	(101)
三、应用举例	(102)

第六章 APPLETs 编程基础

第一节 绘图	(106)
一、直线	(107)
二、矩形	(107)
三、多边形	(109)
四、椭圆	(111)
五、弧线	(112)

六、拷贝与清除	(113)
七、应用举例	(114)
第二节 文本 字体 颜色	(115)
一、文本与字体	(115)
二、颜色	(120)
第三节 图片和声音	(123)
一、图片	(123)
二、声音	(127)
第四节 基本事件处理	(129)
一、鼠标和键盘	(130)
二、键盘	(132)

第七章 高级抽象窗口工具箱 AWT

第一节 AWT 概况	(135)
第二节 UI 部件和 Canvas	(136)
一、Label	(137)
二、Button	(138)
三、Canvas	(139)
四、Checkbox	(141)
五、CheckBoxgroup	(142)
六、Choice	(143)
七、Textfield	(145)
八、Textarea	(146)
九、List	(147)
十、Scrollbar	(149)
第三节 布局	(151)
一、布局管理器	(151)
二、FlowLayout	(154)
三、BorderLayout	(154)
四、CardLayout	(156)
五、GridLayout	(158)
六、GridBagLayout 和 GridBagConstraints	(159)

第八章 例外

第一节 引入例外	(163)
一、问题的提出	(163)
二、Java 例外	(164)
第二节 使用例外	(165)
一、Try/Catch	(165)

二、 <i>Throw</i>	(166)
三、 <i>Finally</i>	(167)

第九章 线程

第一节 Java 多线程	(170)
一、线程创建	(170)
二、一个多线程编程实例	(176)
第二节 多线程运行顺序	(178)
一、安排	(178)
二、一般并行处理中的问题	(180)
三、使用 <i>synchronized</i> 语句	(181)
四、如何使用 <i>synchronized</i> 语句	(182)
五、不能使用 <i>synchronized</i> 语句的地方	(183)

第十章 动画创作

第一节 动画模式	(186)
第二节 调用现成图片创建动画	(188)
第三节 改善动画效果	(193)
一、不清屏	(194)
二、重画改变的部分屏幕	(196)
三、使用双缓冲	(199)

第十一章 JAVA APPLICATIONS 创建及运行

第一节 创建 Java 应用程序	(202)
第二节 参数传递	(203)
第三节 参数处理	(203)

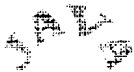
第十二章 窗口

第一节 创建窗口	(206)
第二节 添加菜单	(209)
第三节 对话框	(213)
一、基本对话框	(213)
二、文件对话框	(214)

第十三章 输入输出流及网络联接

第一节 流输入	(216)
一、 <i>InputStream</i> 类	(216)
二、 <i>FileInputStream</i>	(219)
三、 <i>FilterInputStream</i>	(220)

第二节 流输出.....	(222)
一、 <i>OutputStream</i> 类	(222)
二、 <i>FileOutputStream</i>	(223)
三、 <i>FilterOutputStream</i>	(223)
四、 <i>PrintStream</i>	(225)
第二节 网络联接.....	(226)
一、 <i>Applets</i> 间连接	(226)
二、文件流连接.....	(229)
三、客户机/服务器连接	(230)



第一章 Java 入门

本章内容提要:

- 什么是 Java
- Java 与 Internet
- 程序运行及实例

第一节 什么是 Java

一、起源

Java 编程语言最初是由 SUN 公司的一个开发小组设计和实现,当时该开发小组主要设计面向消费类电器的软件,但很快发现了诸如 C 和 C++ 这些现成软件所有的不足之处。

用 C 和 C++ 语言编写的程序必须针对某类特定的计算机芯片进行编译,而一旦出现新的计算机芯片,大多数软件都得重新编译,以充分利用新型芯片的新特性。随着库函数的更新,对一些已编译好的 C 和 C++ 程序,也不能很容易地使用新版软件所提供的库函数,只有从零开始重新编译。

对这个设计消费类电器软件的开发小组,必须面向一些新的芯片编写软件,这样才能提高产品的性能价格比。一旦所采用的处理器或外围器件价格升得太高,就必须采用更经济有效的其它类型芯片,对销售上百万的消费类电器来说,内部器件一个小小的价格波动,都会产生很大的不同。

为消费类电器设计的软件还需要比一般的计算机软件加倍可靠,一旦出现任何故障,往往会导致整个电器瘫痪。

终于在 1990 年,领导这个小组的 James Gosling 开始设计一种更适合于消费类电器的编程语言,而没有 C 和 C++ 这些传统软件所暴露出的问题,其结果是孕育了本书所要介绍的 Java,一种快速、小巧、可靠,并可在任何计算机芯片上运行的新一代编程语言。

James Gosling 起初命名这种语言为 Oak,其思路来自于他办公室外的一棵大橡胶树。

后来发现 Oak 已被早先命名为另一种编程语言的名字,必须重新起名,经过一番苦想,偶然在本地的一间咖啡店中灵感来袭,决定给这种新一代编程语言命名为 Java,一种来自爪哇国的咖啡。



二、Java 语言风格

1. Applications 与 Applets

Java 编程语言从形式上可以分为两类：Applications 与 Applets，Applications 是可由 Java 解释器（Interpreter）单独运行的程序，就象一般的命令行方式；Applets 是一种只在 Internet 万维（World Wide Web — WWW）网提供的浏览器中运行的 Java 程序。

一个 Java Application 可以包含一个或多个类，这些类在规模上可大可小，由 SUN Microsystem 公司开发的支持 Java 的 WEB 浏览器 Hot Java 即为一由 Java 语言编写的 Application，这说明 Java 能够胜任十分复杂的工作。创建一个 Java Application 的一个重要步骤就是在其中某个类中提供一如下标识的程序入口点：

```
public static void main(String arg[])
{...}
```

对只能在 WWW 网浏览器上运行的 Java Applets 来说，就需要在 WEB 页面上用一超级文本设计语言（HTML）标签来实现对其的调用。这种方式可以使 Applets 享用由浏览器提供的多种现成结构：用户接口、事件处理、窗口等等。但系统也常常限制了 Applets 作一些网络或文件操作。



注意

Applications 也可以创建这些结构，但在很多场合并不需要。

2. 面向对象

Java 是一种面向对象编程（OOP）语言，除了数字和布尔数这些简单数据类型外，Java 中所描绘的大部分事物都以对象的方式存在。Java 代码以类的形式组织在一起，每个类定义了一系列可以作用于类对象的行为，同时还定义了类对象可以有的一些状态属性。一个类可以继承另一个类的行为和属性，在类的最顶层是 Object 类。

Java 支持类的单重继承性，也即是一个类不能同时继承多个类，这在很大程度上减少了程序设计的复杂性。但 Java 又提供了对接口（Interface）的支持，多少弥补了一些单重继承性的不足。接口是一种抽象的类，定义在接口中的方法可不用知道具体如何实现，而真正描绘和实现这些方法的代码是出现在可实现这些接口的类中，一个类可以实现多个接口。



3. 编译解释程序

无论对 Applications 或 Applets, 在运行前都将被编译成字节代码 (Byte - Code), 这是一种类似于机器指令的代码, 能使 Java 程序高效率的运行, 并且由于代码的通用性 (不是针对某一具体芯片编写的), 可以在不同的机种上运行, 而勿须重新编译。

Java 源程序通过一 Javac 的编译器被编译成 .class 文件, 文件中就包含了字节代码。当程序运行时, 字节代码解释器会顺序检查 .class 文件方法中的字节代码流, 并执行字节代码流所唯一对应的操作。字节代码解释器的功能如同 CPU, 它检查相应的指令存储单元, 并按指令执行操作, 充当了软 CPU 的角色, 又被称作虚拟引擎 (Virtual Machine)。

4. 多线程支持

现在流行的高级操作系统如用于工作站的 UNIX 和 PC 机上的 Windows95 等, 都提供多任务功能, 即计算机在同一时间可同时执行多个任务。这虽会使单个任务执行的速度下降, 但其优势也是很明显的。

Java 也支持多任务方式, 在程序运行时, 可以允许多个线程 (Thread) 同时并存。例如每个 Applet 可创建一个属于自己的线程, 在该线程内它可不受系统其它部分的干扰而良好的工作。在 Java 中通过使用多线程技术, 可使多个 Applet 在同一页面下同时运行。

由于在同一时间内事件可以不可预见的顺序发生, 这使一般的多任务编程十分困难, 而 Java 中提供了同步 (Synchronization) 特性, 可方便的使关键步骤或易于出错的程序段集中执行。

5. 静态类型定义及垃圾回收

Java 中的任何一种对象 (包括数字、字符、数组等) 的类型在被使用之前都必须定义。由于编译器可检测出类型不匹配的错误, 所以可帮助程序员在早期发现潜在错误。而数组边界和无效指针等错误则可由解释器在运行期间查出。

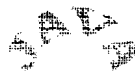
但在另一方面, Java 的所有对象又具有一种动态类型, 在满足一定条件的情况下, 可调用动态类型, 这使程序员在编程时对对象的不同类型进行不同的操作。

在 Java 中程序员可较少关心内存管理问题, 因为系统提供一内置垃圾回收器 (Garbage Collector), 它能不断的扫描内存区并自动释放那些不会再被使用的内存。

三、Java 语言特点

1. 简单

Java 与 C++ 语言十分相似, 但却更为简单, 删去了所有不必要的高级编程技术, 如 Java 中没有运算符重载、头文件、指针运算、结构、多维数组等等。如下为一简单的 Java



Application，它显示字符串“Hello Internet！”。



程序 1-1 HelloInternet 类

```
1. public class HelloInternet{  
2.     public static void main(String args[]){  
3.         System.out.println("Hello Internet!");  
4.     }  
5. }
```

对熟悉 C 和 C++ 语言编程的读者来说，只需花费很少的时间就可掌握 Java 语言，并着手编写自己的 Java 程序。实际上本章中所提到 Java 的许多精彩特性并非 Java 新创，很多都是由诸如 LISP、Smalltalk、PASCAL、C 等语言发明的，Java 只是简单将这些特性良好组合，因此 Java 的关键技术是很容易理解的。

2. 与平台无关性

图 1-1 和图 1-2 表明了传统语言与 Java 语言在开发过程中的不同之处。

当 Java 源程序被编译成字节代码后，就能在所有支持 Java 的平台上运行，即使在一新

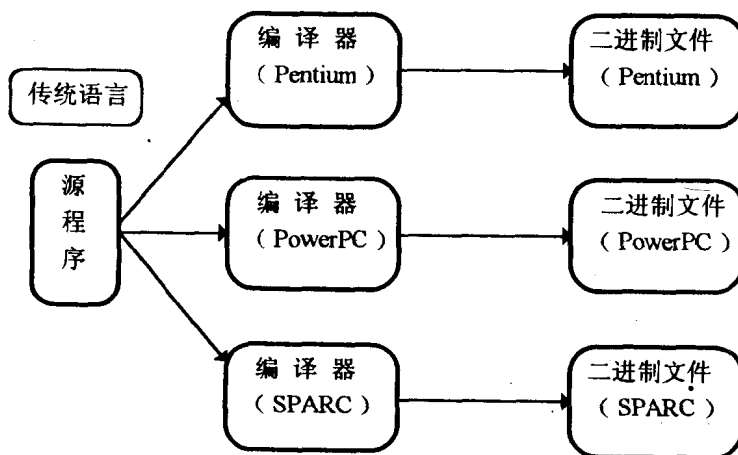


图 1-1

机种上运行也无需重新编译。另外，Java 语言在所有机种上的定义是一致的，如 Int 型就表明其为 32bit，Long 即是 64bit，Char 型就是 16bit 的 Unicode。但现在流行的 C 或 C++ 语言却不是这样，其定义会随编译器和开发环境的不同而有微小差别，这会导致程序移植困难。但 Java 程序的移植却很容易。

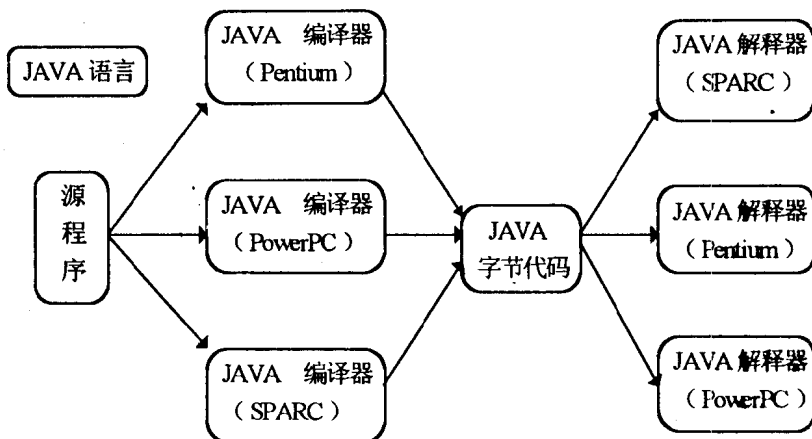
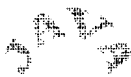


图 1-2

3. 快速

Java 的字节代码效率是很高的，但与 C 语言相比运行速度仍慢大约 20 倍，这主要是因为 Java 程序在执行前不将字节代码转换成本地的机器指令。即使这样 Java 的运行速度在许多场合仍是可接收的。

随着 Java 的发展，已出现多种可提高其运行速度的方法：

- 使用“Just - In - Time”编译器

它不断地记录已实现过的方法的二进制本地指令代码，并不断进行局部优化，若程序运行时再次调用同一方法时，它就直接使用已优化的本地二进制指令代码，这种方法可使 Java 程序运行速度提高十倍以上。

- 对部分方法使用本地（Native）方法

由于本地方法在本地机上可更有效的运行，所以这种办法也可提高 Java 程序的运行速率，但同时也丧失了 Java 的可流通性。

- 使用支持 Java 字节代码的 CPU 或协处理器

随着 Internet 网的飞速发展，Java 语言也日益普及，一些公司（如 SUN Microelectronics 等）相继推出了运行 Java 软件的专用 CPU 芯片。它们的运行速度比传统的通过配置 Java 虚拟引擎的硬件系统（如 Pentium、PowerPC、SPARC）快得多。

为了在传统的计算机系统上也能高效运行 Java 程序，也可应用支持 Java 的协处理器。

所以我们相信随着 Java 的进一步发展，其运行速度将会赶上或超过 C 语言。

4. 安全

我们可从大小两方面来探讨 Java 的安全性：小的方面即指 Java 的稳定性，也就是当程序运行出现故障时，是否会引起诸如死机之类的问题；而大的方面是指程序是否会对系统



和网络产生破坏性。

对许多 PC 机用户来说, 由于程序运行中的错误而导致系统停止工作是一件令人头痛的事, 这会使许多已作过的重要工作报废。而 Java 的程序运行在很大程度上避免了这种情况发生。一方面, 系统会检查每个被访问内存的合法性, Java 程序只能访问那些被允许访问的内存; 另一方面, 即使运行中发生意外的错误也不会导致程序甚至系统的崩溃, 而只是弹出一个例外 (Exception), 最终这一例外会被发现并得到处理。

Java 语言中没有指针设置, 而字节代码又对类型有严格的定义, 这使程序在运行前可得到验证。通过验证的 Java 程序可保证符合其语言规则, 所有可安全运行。验证过程是由 WEB 浏览器完成的, 它通过检查 Java 程序 Applets 的字节代码确保了其中不会携带病毒。

5. 可扩展

有两种原因促使我们有可能在 Java 中定义一些方法为 native, 这些方法是由 Java 以外语言来实现: 一种是为了弥补现有 Java 类库的不足, 如希望在 Java 程序中提供与新外设的接口、利用某类操作系统的一个独特特性等; 另一种则是出于速度上的原因, 目前的 Java 在运行速度上还远远落后于优化后的 C 语言, 而使用 native 方法则在速度上可弥补这个不足。

由于 Java 的数据结构和 C 语言的数据结构及类型极其相似, 所以这种接口的建立并不困难, 只需在 Java 源程序中定义某些方法为 native, 这些本地方法对应到那些定义在软件库中动态联结到虚拟引擎上的函数。

但使用 native 方法也有其不方便之处, 一来现有的大多函数库不支持多线程的运行方式, 二来是出于安全原因, 在网络上传播和使用这种接口方式都受到了限制。

第二节 Java 与 Internet

一、即插即用的 Applets

1. Internet 中的 WWW

WWW 是 World Wide Web 的缩写也可简称为 WEB, 通常我们称为环球网或万维网。1989 年, 欧洲粒子物理实验室 (CERN) 的科学家提出为分布在世界各地的高能物理学家建立一专用信息网络, 并建议计算机间采用简单超文本协议 (HTTP) 传输信息。这是一种基于 Internet 网的、分布式、动态、多平台的交互式超文本信息系统。系统工作方式典型的客户机/服务器(Client/Server)形式。用户处于客户端, 通过 Internet 访问远程站中存储的文件, 作为远程站点的服务器提供用户请求的文件, 客户端使用浏览器浏览接收到的文件。如用户通过 WEB 浏览器取得一 HTML 文件的过程如下:

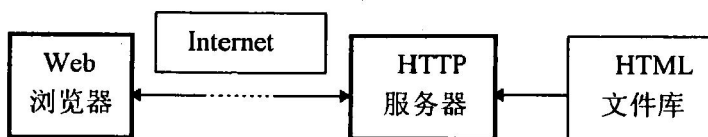


图 1-3

客户机发送 HTML 文件所在的 URL（统一资源定位地址）给 WEB 服务器，WEB 服务器将此 URL 地址理解为自己目录下的文件，就直接将文件发送给客户机供浏览器显示。

2. WWW 浏览器

从上面的传输过程中可看出，浏览器主要起了两方面的作用：

- 接收并解释用户键入的 URL，向相应的服务器发出索取文件的请求并接收这些文件。
- 解释并处理传送过来的 HTML 文件，包括声、像等多媒体信息。

在第一个支持 Java 的浏览器 HotJava（如图 1-4）出现之前，用户只能被动的接收从 WEB 服务器传送过来的超文本信息，最初只有文本信息，以后又增加了对声音、图像信息的欣赏，以至现在可接收新闻和广播节目，但一个共同的特点是这些信息不是可执行文件，只能被静态接收浏览，而不能运行。

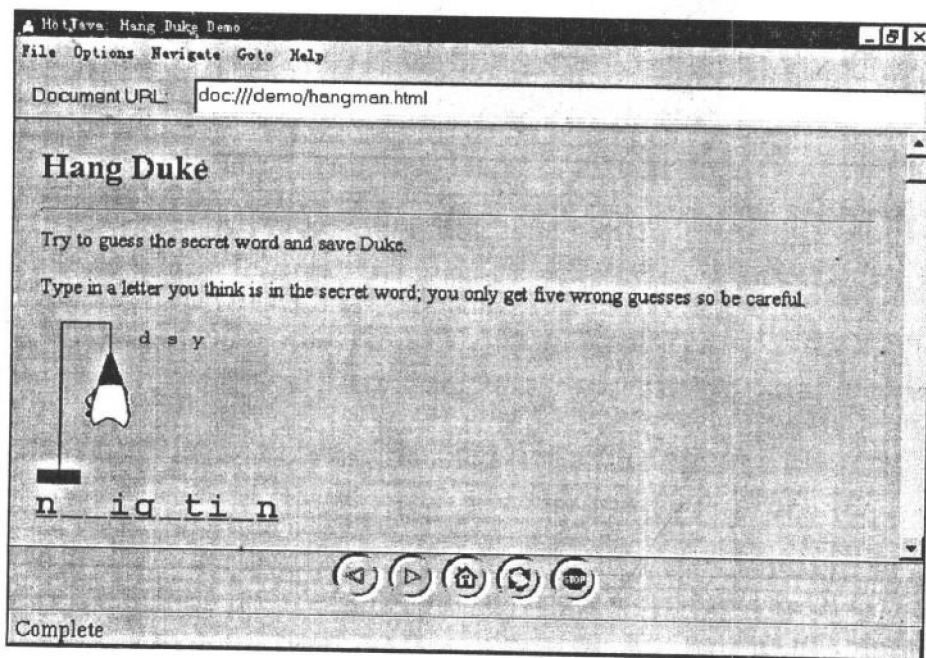


图 1-4