高等学校教材·计算机应用

Web 技术导论

郝兴伟　编著

清 华 大 学 出 版 社

北　京

内 容 简 介

本书是为高等院校计算机应用、信息管理及电子商务等专业的本科学生编写的，集 宰藻遭技术、宰藻遭开发及应用于一体的综合性教材，也可以作为非计算机专业高年级学生学习、了解最新 宰藻遭技术的教材。

全书共 远章，分别介绍了 宰藻遭技术及相关概念、计算模式的演变，宰藻遭服务器的架设和管理，与耘郾酝蕴 载酝蕴标记语言，图像处理、网页动画制作等多媒体技术，允耘藻技术，宰藻遭客户端和服务器端开发等内容。

本书内容丰富、新颖，逻辑严谨，具有很强的专业性、技术性和可操作性，是一本学习 宰藻遭技术、开发及应用的优秀教材，也是 陨栽从业人员和科研人员了解 宰藻遭的综合性参考书。

图书在版编目（悦陨孕）数据

宰藻遭技术导论 辕郝兴伟编著 郾—北京：清华大学出版社，圆园园怨郾园
（高等学校教材·计算机应用）
陨杂月晕 怨苑愿鄄苑鄄猿园圆鄄员远园愿猿鄄猿

陨郾宰…摇陨陨郾郝…摇陨陨陨郾主页制作—程序设计—高等学校—教材摇陨灾郾栽孕猿怨猿郾园怨圆

中国版本图书馆 悦陨孕数据核字（圆园园怨）第 员缘猿缘远苑号

出 版 者：清华大学出版社　　地　　址：北京清华大学学研大厦
　　　　　澡贼贼责:辕辕憎憎憎郾贼怎责郾糟燥皂郾糟灶　　邮　　编：员园园园愿源
　　　　　社总机：园员园鄄远圆苑苑园员苑缘　　客户服务：园员园鄄远圆苑苑远怨远怨
责任编辑：付弘宇
封面设计：王摇永
印 刷 者：
装 订 者：
发 行 者：新华书店总店北京发行所
开摇摇本：员愿缘伊圆远园摇印张：圆猿郾苑缘摇字数：缘苑远千字
版　　次：圆园园怨年 圆月第 员版摇圆园园怨年 圆月第 员次印刷
书　　号：陨杂月晕 怨苑愿鄄苑鄄猿园圆鄄员远园愿猿鄄猿·猿员郾园
印　　数：员～猿园园园
定　　价：猿圆郾园园元

出 版 说 明

改革开放以来，特别是党的十五大以来，我国教育事业取得了举世瞩目的辉煌成就，高等教育实现了历史性的跨越，已由精英教育阶段进入国际公认的大众化教育阶段。在质量不断提高的基础上，高等教育规模取得如此快速的发展，创造了世界教育发展史上的奇迹。当前，教育工作既面临着千载难逢的良好机遇，同时也面临着前所未有的严峻挑战。社会不断增长的高等教育需求同教育供给特别是优质教育供给不足的矛盾，是现阶段教育发展面临的基本矛盾。

教育部一直十分重视高等教育质量工作。[illegible]年 愿月，教育部下发了《关于加强高等学校本科教学工作，提高教学质量的若干意见》，提出了十二条加强本科教学工作提高教学质量的措施和意见。[illegible]年 远月和 [illegible]年 圆月，教育部分别下发了《关于启动高等学校教学质量与教学改革工程精品课程建设工作的通知》和《教育部实施精品课程建设提高高校教学质量和人才培养质量》文件，指出“高等学校教学质量和教学改革工程”，是教育部正在制订的《[illegible]—[illegible]年教育振兴行动计划》的重要组成部分，精品课程建设是“质量工程”的重要内容之一，教育部计划用五年时间（[illegible]—[illegible]年）建设 员[illegible]门国家级精品课程，利用现代化的教育信息技术手段将精品课程的相关内容上网并免费开放，以实现优质教学资源共享，提高高等学校教学质量和人才培养质量。

为了深入贯彻落实教育部《关于加强高等学校本科教学工作，提高教学质量的若干意见》精神，紧密配合教育部已经启动的“高等学校教学质量与教学改革工程精品课程建设工作”，在有关专家、教授的倡议和有关部门的大力支持下，我们组织并成立了“清华大学出版社教材编审委员会”（以下简称“编委会”），旨在配合教育部制定精品课程教材的出版规划，讨论并实施精品课程教材的编写与出版工作。“编委会”成员皆来自全国各类高等学校教学与科研第一线的骨干教师，其中许多教师为各校相关院、系主管教学的院长或系主任。

按照教育部的要求，“编委会”一致认为，精品课程的建设工作从开始就要坚持高标准、严要求，处于一个比较高的起点上；精品课程教材应该能够反映各高校教学改革与课程建设的需要，要有特色风格、有创新性（新体系、新内容、新手段、新思路，教材的内容体系有较高的科学创新、技术创新和理念创新的含量）、先进性（对原有的学科体系有实质性的改革和发展、顺应并符合新世纪教学发展的规律、代表并引领课程发展的趋势和方向）、示范性（教材所体现的课程体系具有较广泛的辐射性和示范性）和一定的前瞻性。教材由个人申报或各校推荐（通过所在高校的“编委会”成员推荐），经“编委会”认真评审，最后由清华大学出版社审定出版。

目前，针对计算机类和电子信息类相关专业成立了两个“编委会”，即“清华大学出版社计算机教材编审委员会”和“清华大学出版社电子信息教材编审委员会”。首批推出的特色精品教材包括：

摇摇 (员 高等学校教材·计算机应用——高等学校各类专业，特别是非计算机专业的计算机应用类教材。

(圆 高等学校教材·计算机科学与技术——高等学校计算机相关专业的教材。

(猿 高等学校教材·电子信息——高等学校电子信息相关专业的教材。

(源 高等学校教材·软件工程——高等学校软件工程相关专业的教材。

(缘 高等学校教材·信息管理与信息系统

清华大学出版社经过近二十年的努力，在教材尤其是计算机和电子信息类专业教材出版方面树立了权威品牌，为我国的高等教育事业做出了重要贡献。清华版教材经过二十多年的精雕细刻，形成了技术准确、内容严谨的独特风格，这种风格将延续并反映在特色精品教材的建设中。

清华大学出版社教材编审委员会

[illegible]

高等学校教材·计算机

编审委员会成员

(按地区排序)

清华大学	周立柱教授
	覃征教授
	王建民教授
	刘强副教授
	冯建华副教授
北京大学	杨冬青教授
	陈钟教授
	陈立军副教授
北京航空航天大学	马殿富教授
	吴超英副教授
	姚淑珍教授
中国人民大学	王珊教授
	孟小峰教授
	陈红教授
北京交通大学	阮秋琦教授
北京信息工程学院	孟庆昌教授
北京科技大学	杨炳儒教授
石油大学	陈明教授
天津大学	艾德才教授
复旦大学	吴立德教授
	吴百锋教授
	杨卫东副教授
华东理工大学	邵志清教授
华东师范大学	应吉康教授
东华大学	乐嘉锦教授
上海第二工业大学	蒋川群教授
浙江大学	吴朝晖教授
	李善平教授
南京大学	骆斌教授

南京航空航天大学摇摇摇摇秦小麟摇教授
南京理工大学 张功萱摇副教授
南京邮电学院 朱秀昌摇教授
苏州大学 龚声蓉摇教授
江苏大学 宋余庆摇教授
武汉大学 何炎祥摇教授
华中科技大学 刘乐善摇教授
中南财经政法大学 刘腾红摇教授
华中师范大学 王林平摇副教授
魏开平摇教授
武汉理工大学 李中年摇教授
国防科技大学 赵克佳摇教授
肖摇侬摇副教授
中南大学 陈松乔摇教授
湖南大学 林亚平摇教授
邹北骥摇教授
西安交通大学 沈钧毅摇教授
齐摇勇摇教授
西北大学 周明全摇教授
长安大学 巨永峰摇教授
西安石油学院 方摇明摇教授
西安邮电学院 陈莉君摇副教授
哈尔滨工业大学 郭茂祖摇教授
吉林大学 徐一平摇教授
毕摇强摇教授
长春工程学院 沙胜贤摇教授
山东大学 孟祥旭摇教授
郝兴伟摇教授
山东科技大学 郑永果摇教授
中山大学 潘小轰摇教授
厦门大学 冯少荣摇教授
福州大学 林世平摇副教授
云南大学 刘惟一摇教授
重庆邮电学院 王国胤摇教授
西南交通大学 杨摇燕摇副教授

前　言

没有哪一项技术像今天的 Internet 一样发展迅速，它对人们的工作、生活的影响范围之广、程度之深，使得人们不能不重视它。无论在书店或在 Internet 上，关于互联网的书籍铺天盖地，令人眼花缭乱。无论是专业的开发人员、普通用户还是网络生活的爱好者，可选择的有关书籍实在太多，以至于人们无所适从。

尽管有如此多的图书供读者选择，可是要找到一本真正适合自己的书却很难。也许你无法说清楚需要什么样的书，因为你对互联网的认识才刚刚开始；也许你要找一本比较全面的书，可是每一项内容都可以写成一本很厚的专业书籍。因此，我编写了这本比较全面的、关于互联网的书籍，取名《Web 技术导论》。这本书将介绍互联网的发展历史，最新的科学进展，Web 的工作原理、实现技术，互联网语言和开发工具，直到 Web 应用的开发、网络安全等内容。这样的内容安排相信对大部分读者都会有所帮助。如果你是一个初学者，这本书会为你答疑解惑；如果你是一个初级的开发人员，这本书可以帮你建立一个基本的开发框架，引领你进入网络开发的广阔天地；如果你是一个高级开发人员，请选择其他更有针对性的书籍。

本书分为 6 章，第 1 章 Web 基础，介绍互联网的发展、Web 的工作机理、相关概念、计算模式、Web 新进展等内容。

第 2 章 Web 服务器的架设和管理，主要介绍 Windows 2000 Server 中的 IIS 和 Apache Tomcat 的架设、管理过程和方法。

第 3 章 HTML 和 XML 基础，介绍这两种标记语言的基本语法，并给出大量的实例，具体解释每种元素的含义和使用。

第 4 章 网页及多媒体制作，在 HTML 和 XML 基础上，介绍可视化的网页制作工具 FrontPage 和 Dreamweaver，同时讲解 Photoshop 图像处理技术和 Flash 动画制作技术。

第 5 章 客户端开发，主要介绍客户端脚本程序 JavaScript、浏览器对象、Web 交互的内容，并详细讲解两个综合性实例。

第 6 章 服务器端开发，介绍 Java 技术、Web 三层体系结构、Servlet/JSP/EJB 服务器端开发，比较了几种主流的开发环境，如 JSP、ASP 和 PHP，同时对 Java 开发工具进行介绍。

虽然我的初衷是要写一本既有理论、又含技术的书籍，但是，要真正地将理论和技术结合起来是很困难的，一方面因为时间仓促，再者需要考虑到读者的实际应用需求。我也很想把更多、更实用的软件代码介绍给读者，并进行讲解，但是，这里受到篇幅的限制，暂时不能如愿。

在本书的写作过程中，我非常感谢我的同事巩裕伟教授，他是一名优秀的教师，将计算机技术深入浅出地传授给学生，受到学生的普遍欢迎。同时，他还是一位很好的程序员，编写了大量的 Java、JSP、VB 代码和数据库应用系统。另外，他还是一位出色的作者，我们合作编写了许多计算机的相关书籍。同时，我要感谢我的同事焦文江老师，他对网络环境有着很深入的研究，对网络设备非常熟悉，对待工作认真负责。此外，还要感谢苏雪女

士，她在一家大型的网络公司任职，主要从事网络开发工作，我们一起合作开发了许多 Web 应用。还要感谢刘丰宁先生，他同样从事网络管理和开发工作。

最后要感谢 Internet 本身，它为我们提供了海量的信息和如此快速、便捷的交流平台。

由于本书涉及的内容非常广泛，在深度和广度上很难完美兼顾，加之作者水平有限，书中肯定存在错误和不足，恳请读者批评指正。

作者 E-mail 为 hxw@sdu.edu.cn。

编者　郝兴伟

2004 年夏

目　　录

第 1 章　Web 基础

今天，互联网已经成为使用最广泛的传播媒体，它正在改变着人们的工作、生活和娱乐方式。利用 Internet，人们可以发布消息、搜索信息、进行商务活动；还可以收发电子邮件、浏览网页、网上交流、视频点播、玩网络游戏……

Internet 就像空气一样，正在渗透到人们生活的每一个角落。本章将简要介绍 Internet 和 Web 的有关概念以及 Web 技术的新进展。

1.1　Internet 与万维网

1946 年，第一台电子计算机“爱尼亚克”(ENIAC)在美国宾夕法尼亚大学莫尔电子工程学院诞生。这是计算技术的革命，它带来了数字信息时代的第一缕曙光。随后，微电子技术和计算机技术经历了日新月异的发展过程。为了进一步提高计算机的使用效率，人们需要将不同的计算机连接起来，传递数据，共享资源，由此计算机网络诞生了。

20 世纪 60 年代出现的各式各样的计算机网络解决了计算机之间的通信和资源共享问题。1969 年，美国国防部高级研究计划署 ARPA 资助了一个有关广域网络的项目，开发一个称作阿帕网(ARPANet)的网络，它的主要思想是构建一个没有中央控制节点的计算机网络，以便使军事计算机系统在受到打击后不会因为部分毁坏而导致整个计算机网络的瘫痪。

1969 年 11 月 21 日中午，6 名科学家聚在美国加利福尼亚大学洛杉矶分校的计算机实验室，观看这里的一台计算机与远在千里之外的斯坦福研究所的另一台计算机连接。这是一个历史性的时刻，正像 20 年后《时代》周刊的评论：这些研究者根本没有想到，他们不只是连接了两台计算机，而是宣告了网络时代的到来。

到 1970 年，ARPANet 已初具雏形，已经将加利福尼亚州(加州)大学洛杉矶分校、加州大学圣巴巴拉分校、斯坦福大学、犹他州大学 4 所大学的 4 台计算机通过分组交换协议连接起来，实现了不同型号、不同操作系统、不同数据格式、不同终端的计算机之间的通信和资源共享。

1972 年，ARPANet 已建成 40 多个网点，开发出了三项主要的功能，即以后被广泛使用的电子邮件、远程登录和文件传输。1974 年，著名的 TCP/IP 协议研究成功，彻底解决了不同的计算机和系统之间的通信问题，扫除了计算机互联的主要障碍。

1975 年，ARPANet 的运行管理被移交给美国国防通信局(DCA)。1982 年，DCA 将 ARPANet 各站点的通信协议全部转为 TCP/IP，同时 ARPAnet 被分成两部分，一部分作为军用，称为 MILnet，另一部分作为民用。这表明 ARPANet 开始从一个实验型网络向实用型网络转变，从而成为全球 Internet 正式诞生的标志。

如果要给 Internet 的发展划分阶段的话，那么 1969－1984 年这个时期可以看作是 Internet 的提出、研究和试验阶段，这时的 Internet 以 ARPAnet 为主干网。由于 ARPAnet

采用离散结构，不设中央网络控制设备，实现了网络渠道的多样性，从而减少了系统彻底崩溃的可能性，网络的生存能力得到了保证，实现了 ARPA 的最初构想。

后来，Internet 的发展超出了任何人的想象。1984—1992 年可以看作是 Internet 的实用发展阶段。为了使全美国的科学家和工程师能够共享那些过去只有军事部门和少数科学家才能够使用的超级计算机设备，美国国家科学基金会(National Science Foundation，NSF)于 1985 年提供巨资在全美建立了 5 个超级计算中心，同时建设了将这些超级计算中心和各科研机构相连的高速信息网络 NSFnet。1986 年，NSFnet 成功地成为 Internet 的第二个骨干网。NSFnet 对 Internet 的推广起到了巨大的推动作用，它使得 Internet 进入了以资源共享为中心的实用服务阶段，而不再是仅有科学家、工程师、政府部门使用的网络。以连接 NSFnet 的局域网数量为例，1988 年 7 月只有 170 个，到 1992 年 1 月这一数量就发展到 4500 个。

1992 年以后，Internet 开始进入商业化发展阶段，Internet 用户开始向全世界扩展，并以每月 15%的速度迅速增长，每 30 分钟就有一个网络连入 Internet。随着网上通信量的急剧增长，Internet 开始不断采用新的技术以适应发展的需求，其主干网由政府部门资助开始向商业计算机公司、通信公司转化。

在 Internet 商业化的过程中，万维网(World Wide Web，WWW)的出现，使 Internet 的使用更简单、方便，开创了 Internet 发展的新时期。1989 年，在瑞士日内瓦欧洲核子物理研究中心(CERN)工作的蒂姆·伯纳斯·李(Tim Berners-Lee)首先提出了万维网的概念，并且成功地开发出世界上第一个万维网服务器和第一个万维网客户机。同年底，蒂姆给他的发明正式定名为 World Wide Web(万维网)；1991 年 5 月，万维网在 Internet 上首次露面，立即引起轰动，被迅速广泛地推广应用。

在 WWW 的发展中，还有一位杰出的人物，他就是马克·安德森，是他改造了 Internet 的使用界面。在早期，万维网只有文字，没有图像、声音，也没有色彩。对普通用户来说，仍缺乏一种简单的使用界面。安德森在就读伊利诺斯大学时，就开始在学校里的国家超级计算中心(NCSA)做兼职工作。由于感觉到 Internet 界面难于使用，他和同事贝纳合作，经过 6 个星期的辛苦工作，终于在 1993 年 1 月取得了初步成果，写出了 Unix 版的马赛克(Mosaic)浏览器。

美国著名的信息专家、《数字化生存》的作者尼葛洛庞帝教授认为：1989 年是互联网历史上划时代的分水岭。这一年出现的万维网技术给 Internet 赋予了强大的生命力，把 Internet 带入了一个崭新的时代。

1.2 Web 概述

万维网是 20 世纪 90 年代 Internet 技术、超文本技术和多媒体技术相结合的产物。

1.2.1 Web 是什么

Web 是什么呢？从 Web 诞生之日起，人们并没有给它一个确切的定义。我们可以从 Internet 的构成和服务来理解 Web。

Internet 是一个网络上的网络，或者说是一个全球范围的网间网。在 Internet 中分布了成千上万的计算机，这些计算机扮演的角色和所起的作用各不相同。有的计算机可以收发电子邮件，有的可以为用户传输文件，有的负责对域名进行解析，更多的机器则用于组织并展示本网络的信息资源，方便用户的获取。所有这些承担服务任务的计算机统称为服务器。根据服务的特点，又可分成邮件服务器、文件传输服务器、DNS 服务器和 Web 服务器等。

Web 服务器就是将本地的信息用超文本组织，方便用户在 Internet 上搜索和浏览信息的计算机。因此 Web 或者说 World Wide Web，是由 Internet 中称为 Web 信息服务器的计算机组成的，由那些希望通过 Internet 发布信息的机构提供并管理。可以这样说，万维网是 Internet 的一个子集，即有：

$$WWW \subset Internet$$

在 Web 世界里，每一个 Web 服务器除了提供自己独特的信息服务外，还可以用超链接指向其他的 Web 服务器。那些 Web 服务器又可以指向更多的 Web 服务器，这样，一个全球范围的、由 Web 服务器组成的 World Wide Web(万维网)就形成了。

1.2.2　超文本、HTML、XML 与 Web 页

Web 是以超文本方式组织信息和提供信息服务的。那么，什么是超文本呢？

超文本(hypertext)是一种人机界面友好的计算机文本显示技术，可以对文本中的有关词汇或句子建立链接，使其指向其他段落、文本或弹出注解。用户在读取超文本时，建立了链接的句子、词语甚至图片将以不同的方式显示，或者带有下划线、或加亮显示、或粗体显示、或以特别的颜色显示，来表明这些文字对应一个超链接。当鼠标移过这些文字时，鼠标会变成手形，单击超链接文字，可以转到相关的文件位置。

Web 信息服务器上的超文本是用超文本标记语言(HyperText Markup Language，HTML)开发编写的，在 HTML 中定义了一系列的特殊标记，来标记一段文本的显示方式。当用户通过 Web 客户端程序(即浏览器)来打开这些文件时，浏览器将遵循 HTML 的规定，对打开的文本以指定的方式显示在屏幕上，这就是通常所说的 Web 页。

现在，除了 HTML 以外，XML 作为一种标记语言正在得到更加广泛的应用，已经成为 Web Service 技术体系的重要成员。

1.2.3　浏览器

浏览器(Browser)就是前面提到的 Web 客户端程序，用户要浏览 Web 页面必须在本地计算机上安装浏览器软件。

浏览器主要分成两类，一类是以 Lynx 为代表的、基于字符的 Web 客户端程序，主要在不具备图形图像功能的计算机上使用。Lynx 是由美国堪萨斯大学的 Lou Montulli 研制的，同类的还包括 CERN 的 LineMode Browser。另一类是以 NCSA(National Center of Supercomputing Application)Mosaic 为代表的、面向多媒体计算机的 Web 客户端程序，它可

以在各种类型的小型机上运行，也可以在 IBM PC 机、Macintosh 机以及 UNIX 操作系统平台上运行。目前，用户使用最多的是 Netscape 和 IE(Internet Explorer)浏览器。

1.2.4 工作原理

Web 是由分布在 Internet 中的 Web 服务器组成的。所谓 Web 服务器，就是那些对信息进行组织、存储并将其发布到 Internet 中去，从而使得 Internet 中的其他计算机可以读取这些信息的计算机。

要使一台计算机成为一台 Web 服务器，必须安装 UNIX、Windows NT Server 或 Windows 2000 Server 等网络操作系统，并且还要安装专门的信息服务器程序，如 Windows 2000 Server 中的 IIS 5.0 或 Apache Tomcat 等。要成为 Web 客户机很简单，只要将计算机连接到 Internet，并安装客户端软件，如 Internet Explorer、Netscape 等浏览器程序即可。

在 Web 中使用的通信协议是 HTTP 协议，通过 HTTP 协议实现客户端(浏览器)和 Web 服务器的信息交换。Web 的基本工作原理如图 1-1 所示。

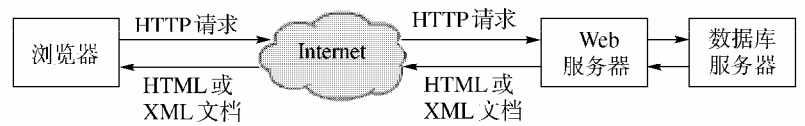


图 1-1 Web 的工作原理

当用户通过 Web 浏览器向 Web 服务器提出 HTTP 请求时，Web 服务器根据请求调出相应的 HTML、XML 文档或 ASP、JSP 文件。如果是 ASP 或 JSP 文档，Web 服务器首先执行文档中的服务器端脚本程序，然后把执行结果返回给客户端浏览器。

现在一般的 Web 应用都是和数据库结合在一起的，服务器端脚本程序主要负责通过 ODBC 与数据库服务器建立连接，完成必要的查询、插入、删除、更新等数据库操作，然后利用获得的数据产生一个新的、包含动态数据的 HTML 或 XML 文档，并将其发送回客户端 Web 浏览器。最后由 Web 浏览器解释该文档，在浏览器窗口中显示给用户。

1.3 相 关 知 识

随着 Internet 的发展，新的技术不断出现，在网络环境下的计算模式不断发展，出现了许多新的概念。下面介绍一些和万维网相关的概念或术语。

1.3.1 常见概念和术语

(1) 网站(Web site)

又称 Web 站点，是 Internet 中提供信息服务的机构，这些机构的计算机连接到 Internet 中，可以提供 WWW、FTP 等服务。

(2) Web 页(Web page)

Web 页是指 Web 服务器上的一个个超文本文件，或者是它们在浏览器上的显示屏幕。Web 页中往往包含指向其他 Web 页面的超级链接。

(3) 主页(homepage)

用户在 Web 服务器上看到的第一个 Web 页，该 Web 页一般的名称为 default.htm 或 index.htm。主页中往往列出了网站的信息目录，或指向其他站点的超链接。

(4) 超级链接(hyperlink)

Web 页中，当用户单击超链接时，可以转到其他 Web 页或当前页面其他地方的文字、图片等对象。超级链接在 Web 页上往往带有下划线或增亮显示，当用户将鼠标指向一个超链接时，鼠标指针会改变成手的形状。

(5) 通用资源定位器(Uniform Resource Locator，URL)

可以惟一标志一个 Web 页或 Internet 上其他资源的地址。它将 Internet 提供的各类服务统一编址，以便用户通过 Web 客户端程序进行信息查询。

URL 的一般形式为

```
信息资源类型://域名/文件路径
```

(6) 端口(ports)

端口是服务器使用的一个通道，可以使具有相同 IP 地址的服务器同时提供多种服务。例如，在 IP 地址为 202.194.7.66 的计算机上同时提供 HTTP 服务和 FTP 服务，HTTP 服务使用端口 80，FTP 服务使用端口 21 等。

另外，在一台计算机上，在同一个 IP 地址下，可以建立多个 Web 站点，站点之间用端口号来区分。

(7) 下载(download)

下载是指通过 Internet 将文件从 FTP 服务器传输到本地计算机的过程。

(8) 上传(upload)

上传是指通过 Internet 将文件从本地计算机传输到 FTP 服务器的过程。

(9) 存储片(cookie)

cookie 是 Web 服务器传送到浏览器端的数据流，用于存储服务器端的数据以及运行的中间结果，以数据文件的形式存储在客户机的硬盘中。

(10) 手机上网

手机上网是指利用手机连接 Internet。手机上网需要两个条件：第一，手机必须支持 WAP 功能；第二，手机服务运营商必须支持 WAP 服务。

WAP 即无线通信协议(Wireless Application Protocol)，它是在数字移动电话、数字个人助理(PDA)、计算机和 Internet 之间进行通信的开放标准。WAP 论坛成立于 1998 年，由 Ericsson、Nokia、Motorola 和 Unwired Palnet 4 家公司发起，其中 Ericsson 是最初的研发者和倡导者。

要使用手机上网，首先要确定手机具有 WAP 功能，其次是到一家电信运营商，例如中国移动或中国联通营业厅办理开通 WAP 服务的手续，然后就可以利用手机上网了。

热门的 WAP 站点有随身网(211.99.40.60)、新通讯(www.5comm.com)、中国移动 WAP

网(wap.chnmobile.com)、搜狐(wap.sohu.com)等。

(11) 蓝牙技术

蓝牙(blue tooth)技术是短距离无线互联技术，它的倡导者是瑞典的 Ericsson 公司，其初衷就是要统一全球的无线通信技术，希望这种技术能够一统天下。蓝牙技术面向的对象是个人和商务的无线连接应用，相对于 WAP，它规范了更为具体的硬件及应用频率等内容。蓝牙技术最初的设计目的是替代电缆，而现在的蓝牙技术可以应用于更多领域，例如手机和计算机的连接、无绳电话、影像传输等。

蓝牙技术在 1998 年实现标准化，手机的发展带动了蓝牙技术的进步。到 2002 年，蓝牙技术开始应用于产量较大的产品中，其中应用最多的就是手机。用户技术解决了手机辐射对身体的伤害，置有蓝牙芯片的手机，一般也带有蓝牙耳机，电话可以把手机放在书桌或工作台上，使用无线耳机在一定距离内接听电话，从而达到降低辐射的目的。

1.3.2 集中式计算模式

在计算机诞生和应用的初期，计算所需要的数据和程序都是集中在一台计算机上进行的，称为集中式计算。

随着网络的发展，这种集中式计算往往发展成一种由大型机和多个与之相连的终端组成的网络结构。当支持大量用户时，大型机自顶向下的维护和管理方式显示出集中式处理的优越性。它具有安全性好、可靠性高、计算能力和数据存储能力强以及系统维护和管理的费用较低等优点。但是它也存在着一些明显的缺点，如大型机的初始投资较大、可移植性差、资源利用率低以及网络负载大等。

1.3.3 客户/服务器(C/S)计算模式

随着微型计算机和网络的发展，数据和应用逐渐转向了分布式，即数据和应用程序跨越多个节点，形成了新的计算模式，这就是 C/S 计算模式(Client/Server，客户/服务器)。这是一种典型的两层计算模式。

C/S 计算模式将应用一分为二：前端是客户机，一般使用微型计算机，几乎所有的应用逻辑都在客户端进行和表达，客户机完成与用户的交互任务，具有强大的数据操纵和事务处理能力。后端是服务器，可以使用各种类型的主机，服务器负责数据管理，提供数据库的查询和管理、大规模的计算等服务。

C/S 计算模式具有以下几个方面的优点：通过异种平台集成，能够协调现有的各种基础结构；分布式管理；能充分发挥客户端 PC 的处理能力，安全、稳定、速度快，且可脱机操作。

但随着应用规模的日益扩大，应用程序的复杂程度不断提高，C/S 结构逐渐暴露出许多缺点和不足，主要包括：它必须在客户端安装大量的应用程序(客户端软件)，开发成本较高，移植困难，用户界面风格不统一，操作复杂，不利于推广使用，维护、升级过程繁琐，信息内容和形式单一，不易应用新技术等。

1.3.4　浏览器/服务器(B/S)计算模式

进入 20 世纪 90 年代以后，随着 Internet 技术的不断发展，尤其是基于 Web 的信息发布和检索技术、Java 技术以及网络分布式对象技术的飞速发展，常出现成千上万台客户机同时向服务器发出请求的情况，这就使很多应用系统的体系结构不得不从 C/S 结构向更加灵活的多级分布式 B/S 结构(Browser/Server，浏览器/服务器模式)演变。

浏览器/服务器计算模式是一种基于 Web 的协同计算，是一种三层架构的瘦客户机/服务器计算模式。

第一层为客户端表示层，与 C/S 结构中的“肥”客户端不同，三层架构中的客户层只保留一个 Web 浏览器，不存放任何应用程序，其运行代码可以从位于第二层的 Web 服务器下载到本地的浏览器中执行，几乎不需要任何管理工作，这就是人们平时所说的“瘦”客户机。第二层是应用服务器层，由一台或多台服务器(Web 服务器也位于这一层)组成，处理应用中的所有业务逻辑，对数据库的访问等工作，该层具有良好的可扩充性，可以随着应用的需要任意增加服务器的数目，由于管理工作主要针对服务器进行，相对于 C/S 而言，无论是工作的复杂性还是工作量都大大降低。第三层是数据中心层，主要由数据库系统组成。

B/S 计算模式与传统的 C/S 结构相比体现了集中式计算的优越性：具有良好的开放性，利用单一的访问点，用户可以在任何地点使用系统；用户可以跨平台以相同的浏览器界面访问系统；因为在客户端只需要安装浏览器，基本上取消了客户端的维护工作，有效地减少了整个系统的运行和维护成本。

1.3.5　网络计算

虽然 Internet 的发展日新月异，但是，人们也正在面临新的挑战。

首先，Internet 上的资源急剧膨胀，其相互关联关系不断发生变化，缺乏有效的管理，呈现出无序状态，人们已经很难有效地控制整个系统。第二，Internet 上的局部自治系统各自为政，相互之间缺乏有效的交互、协作和协同能力，难以联合起来共同完成大型的应用任务，严重影响了全系统综合效用的发挥，也影响了局部系统的利用率。第三，Internet 上的资源存在着多方面的差异，这种千差万别的状态加大了网络计算系统的使用难度，在一定程度上限制了网络计算的发展空间。

Internet 上汇集了成千上万的计算资源、数据资源、软件资源、各种数字化设备和控制系统，它们共同构成了生产、传播和使用知识的重要载体。如何将物理上互连的众多资源汇聚起来，联合提供服务，就需要重新认识网络环境下的计算技术。这样背景下的计算通常称为网络计算。概括地讲，网络计算就是把网络连接起来的各种自治资源和系统组合起来，以实现资源共享、协同工作和联合计算，为各种用户提供基于网络的各类综合性服务。

一般情况下，网络计算分为企业计算、网格计算、对等计算和普及计算四个大类。

- 企业计算是“以实现大型组织内部和组织之间的信息共享和协同工作为主要需求而形成的网络计算技术”，其核心是客户/服务器计算模型和相关的中间件技术。

- 网格计算(Grid Computing)是网络计算的另一个具有重要创新思想和巨大发展潜力的分支。最初，网格计算研究的目标是希望将超级计算机连接成为一个可远程控制的元计算机系统(Meta Computers)；现在，这一目标已经深化为建立大规模计算和数据处理的通用基础支撑结构，将网络上的各种高性能计算机、服务器、PC、信息系统、海量数据存储和处理系统、应用模拟系统、虚拟现实系统、仪器设备和信息获取设备(如传感器)集成在一起，为各种应用开发提供底层技术支撑，将 Internet 变为一个功能强大、无处不在的计算设施。
- 对等计算(Peer-to-Peer，P2P)是在 Internet 上实施网络计算的新模式。在这种模式下，服务器与客户端的界限将消失，网络上的所有节点都可以“平等”地共享其他节点的计算资源。

IBM 公司将 P2P 定义为由若干互联协作的计算机构成，且至少具有如下特征之一：系统依赖于边缘化(非中央式服务器)设备的主动协作，每个成员直接从其他成员而不是从服务器的参与中受益；系统中成员同时扮演服务器与客户机的角色；系统应用的用户能够意识到彼此的存在，构成一个虚拟或实际的群体。不难看出，P2P 把网络计算模式从集中式引向分布式，即网络应用的核心从中央服务器向网络边缘的终端设备扩散：服务器到服务器，服务器到 PC 机，PC 机到 PC 机，PC 机到 WAP 手机，所有网络节点上的设备都可以建立 P2P 对话。

P2P 给 Internet 的分布、共享精神带来了无限的遐想。有观点认为，至少能开发出几百种 P2P 的应用。但从目前的应用看，P2P 的优势主要还体现在大范围的共享和搜索上，诸如对等计算、协同工作、搜索引擎、文件交换等。

- 普及计算(ubiquitous computing or pervasive computing)强调人与计算环境的紧密联系，使计算机和网络更有效地融入人们的生活,让人们在任何时间、任何地点都能方便快捷地获得网络计算提供的各种服务。普及计算研究的内容主要包括自然的人机交互和网络计算两个方面。

1.4 Web 中的服务

基于 Web 的服务越来越多，除了传统的 HTTP、FTP、Email 外，各种各样的服务形式不断出现，成为网络经济的新形式。这些网络服务依托于网络数据中心(IDC)，主要的服务包括如下形式。

1．主机托管服务

主机托管服务是用户租用机房机架和网络带宽，将自己的主机服务器托管在 IDC 机房里。带宽租用包括共享带宽和独享带宽服务。

2．专线接入服务

用通信线路将用户的网络连接到 IDC 的网络，这些服务包括光纤接入、DDN 接入、帧中继等业务，还可以根据用户的特殊要求提供高安、全低成本的虚拟专用网(VPN)和高质

量的 IP 电话解决方案。

3．整机租用服务

用户在租用机房机架和网络带宽的同时，还可以租用 IDC 的主机、标准服务器、操作系统等相关系统平台软件，然后在其上实现自己的应用系统。

4．虚拟主机服务

虚拟主机服务是多个用户共享一台服务器，可各自拥有独立的域名、IP 地址、存储空间、数据库空间等，为中小用户提供应用系统上网的条件。

虚拟主机服务是一种用户投入成本较低的 Internet 接入服务。在 IDC，一台主机服务器的磁盘被分成若干空间，每一个空间就可以称为一台虚拟主机。它的优点包括：一是每台虚拟主机都有自己独立的域名；二是用户无需购买任何设备，但在访问者看来，却好像用户拥有完全独立的服务器。虚拟主机同时还具有网络运行环境好、高速、可任意选择磁盘空间大小、无需自身维护等多项优点。用户可以租用 IDC 的部分主机空间，实现 Web、数据库、邮箱、域名解析以及其他简单的网络应用。花小钱办大事，虚拟主机对中小企业是一种很好的选择。

5．其他增值服务

除了上述的基本服务外，还有一系列的增值服务，例如内容分发服务、防火墙负载均衡服务、虚拟专用网络服务、负载均衡服务、SSL 加速服务、内容高速缓存服务、网络安全服务、存储与备份服务、企业邮箱服务、广告与信息发布服务以及短信服务等。

1.5　Web 的新进展

Internet 的核心是超文本系统，其主要思想是通过统一资源标识符(Uniform Resource Identifier，URI)对互联网上的信息进行标记，使人们可以迅速地对互联网上的信息资源进行定位，从而实现互联网中的信息交流和共享。然而，现有互联网技术并没有对信息的含义进行描述，计算机在处理信息时只是按照 URI 来定位信息，对信息的内容并不关心。而人们真正关心的是信息的内容，也就是互联网上的文本、图片等资源所包含的意义。新的互联网技术的研究目的就是要改变这种状况，包括语义 Web、Web Services 等技术。

1.5.1　语义 Web

语义 Web(Semantic Web)是一种新的互联网技术，它通过扩展现有互联网，在信息中加入表示其含义的内容，从而使计算机可以自动与人进行协同工作。语义 Web 有着重要的研究价值，国外的很多大学、研究机构、大公司都成立了专门的项目组来推动这项技术的发展，W3C(World Wide Web Consortium)组织成立了专门的工作组来推动语义 Web 技术的发展。2001 年 7 月 30 日，在斯坦福大学召开了题为 Infrastructure and Applications for the

Semantic Web 的学术会议。2002 年 7 月 9 日，在意大利召开了 1st International Semantic Web Conference 会议。2002 年，我国的 863 计划将语义 Web 技术列为重点支持项目。

1．语义 Web 分层模型

语义 Web 一般分成 5 层，分别是：

- XML 层，作为语法层。
- RDF(Resource Description Framework，资源描述框架)，数据层。
- 本体层(Ontology Layer)，作为语义层。
- 逻辑层(Logic Layer)，提供了智能推理的规则。
- 证据层(Proof Layer)，支持代理间通信的证据交换。

信息在语法描述上往往存在差异，这种表达上的差异需要通过必要的数据格式转换，将信息转化为目标应用能够处理的语法格式。理想情况是在所有的应用中，信息都采用同样的语法来描述。目前，XML 已经成为 Web 上数据表示和交换的事实标准，是应用之间或者机器之间共享数据的一种有效方式。XML 及其相关技术的发展极大地促进了信息表达和交换过程中语法描述上的统一，越来越多的应用开始选用 XML 作为其数据、配置信息、消息以及服务的语法描述模式。

XML 本身不具备语义描述能力，为此，W3C 推荐以 RDF 标准来解决 XML 的语义局限。RDF 提出了一个简单的模型，用来表示任意类型的数据。这个数据类型由节点和节点之间带有标记的连接弧组成。节点用来表示 Web 上的资源，弧用来表示这些资源的属性。因此，这个数据模型可以方便地描述对象(或者资源)以及它们之间的关系。

本体层将为语义 Web 提供语义级的共享，是语义 Web 实现的关键所在。

2．相关技术

语义 Web 的目标是提高互联网的自动化与智能化，而语义 Web 技术包含着一系列相关技术，其中包括基础研究与应用研究。基础研究方面主要包括本体(ontology)的发展、语义 Web 语言的形式语义和确信(Trust)与证据(Proof)模型的开发。语义 Web 的应用研究主要集中在几个方面：Web Services、基于代理的分布式计算、基于语义的网页搜索引擎和基于语义的数字图书馆。

目前的各种 Web 技术都有可能被应用于语义网，例如 DOM(文档对象模型，一组访问 XML 和 HTML 文档组成部分的标准接口)、XML Schema、RDF、XSL (eXtensible Style sheet Language)、SVG (Scalable Vector Graphic)、SMIL、SOAP、DTD 以及元数据概念等。

1.5.2 Web Services 技术

Web Services(Web 服务)是一系列标准和正在发展中的标准，它们由 W3C 设计和开发，用来促进跨平台的程序之间的通信。Web Services 的核心技术包括 XML、Namespace、XML Schema、SOAP、WSDL、UDDI 等。

HTTP、XML 和 SOAP 可以看作是 Web 服务的核心层。这些层定义了 Web 服务之间交互的方法和途径。这三个协议已经被 W3C 接受并作为标准。使用 HTTP 通信协议，人们可

以从 Internet 上的一个地方向另一个地方发送消息。通过网络发送的消息可以使用 XML 结构化，XML 协议定义这条消息的格式和语义。SOAP(Simple Object Access Protocol，简单对象访问协议)是定义如何从不同环境中的对象调用函数。使用 SOAP，就能够整合不同的操作系统、对象模型和编程语言，使简化整合不同种类的业务处理过程成为可能。

WSDL 协议(Web Service Definition Language，Web 服务描述语言)描述如何与一个 Web 服务通信。在 WSDL 定义中，允许不同类型的通信(绑定)。它可以用来开发 Web 服务，同时也可以用来赚钱。为了实现这个目的，需要一个 Web 服务门户，用户可以在那里发布自己的 Web 服务，其他的人也能在那里找到并使用它，这就需要使用 UDDI(Universal Description, Discovery and Integration，统一描述、发现和整合规范)。

UDDI 建了一个平台独立、开放的框架，通过 Internet 来描述服务，发现业务，并且整合业务服务。它是一套基于 Web 的、分布式的、为 Web 服务提供的信息注册中心的实现标准规范，同时也包含一组使企业能将自身提供的 Web 服务注册以使别的企业能够发现的访问协议的实现标准。

通过使用 UDDI 的发现服务，企业可以单独注册那些希望被别的企业发现的、自身提供的 Web 服务。企业可以通过 UDDI 商业注册中心的 Web 界面，来将信息加入到 UDDI 的商业注册中心。UDDI 商业注册中心在逻辑上是集中的，在物理上是分布式的，由多个根节点组成，相互之间按一定规则进行数据同步。当一个企业在 UDDI 商业注册中心的一个实例中实施注册后，其注册信息会被自动复制到其他 UDDI 根节点，于是就能被任何需要这些 Web 服务的人所发现。

习　题　1

1．什么是万维网？

2．什么是 Web 服务器？简述 Web 的基本工作原理。

3．解释下列概念：

网站、网页、HTTP、URL、端口、cookie、蓝牙技术

4．什么是 B/S 结构？它和 C/S 结构相比，有什么优点？

5．简述语义 Web 模型。

6．什么是 Web Services？它包括哪些主要技术？

第 2 章　Web 服务器的架设和管理

在 Internet 中， Web 服务是最主要的网络服务，几乎一提到 Internet，就会想到万维网(World Wide Web，WWW)。实际上，WWW 只是 Internet 的一个子集，它是由 Internet 中的 Web 服务器和 Web 客户机构成的。Web 服务器就是那些安装了 Web 服务器软件的计算机，而安装了浏览器的计算机就是 Web 客户机，可以通过 Internet 访问 Web 服务器。

要使一台计算机成为 Web 服务器，首先需要安装网络操作系统，同时还需要安装相应的 Web 服务组件。目前，应用最广泛的信息服务有 Windows NT/2000 Server 中的 IIS，以及 Tomcat 和 Apache 的组合应用。本章主要以 Windows 2000 Server 中的 IIS 为例，介绍 Internet 中 Web 服务器的构建、配置和管理，最后对 Tomcat 和 Apache 做简单介绍。

2.1　Windows 2000 和 Internet 信息服务

Windows 2000 是目前应用非常广泛的操作系统，本身具有强大的内嵌网络功能。Windows 2000 有 4 个不同的版本，分别是 Windows 2000 Professional、Windows 2000 Server、Windows 2000 Advanced Server 和 Windows 2000 Datacenter Server，分别针对不同规模的应用。其中，应用最为广泛的是 Windows 2000 Professional 和 Windows 2000 Server，它们被广泛地安装在个人计算机和企业网络服务器上。

无论是 Windows 2000 Professional，还是 Windows 2000 Server，都可以安装 Internet 信息服务组件 IIS，成为 Web 服务器，构建 Web 站点，提供 Web 服务。在 Windows 2000 Server 中，IIS 是默认安装的，在 Windows 2000 Professional 中需要通过“添加/删除程序”功能完成 IIS 的安装。

2.1.1　什么是 IIS 5.0

Internet 信息服务器(Internet Informationn Server，IIS)是一组 Windows 操作系统组件，此组件可以使公司很方便地创建自己的 Web 服务器、FTP 服务器以及简单的 SMTP 和 NNTP 服务器，很方便地将信息和业务应用程序发布到 Web 中。IIS 使用户能容易地为网络应用程序和通信创建功能强大的平台。

2.1.2　IIS 5.0 的组成

IIS 5.0 由若干可选组件构成，用户可以根据需要选择不同的组件进行安装和配置。下面介绍每个组件的功能。

(1) FrontPage 2000 服务器扩展

安装此组件后，用户可以使用 FrontPage 2000 和 Visual InterDev 编写、管理 Web 站点的内容。

(2) Internet 服务管理器

用于配置和管理 IIS 5.0，可以在 MMC 中以管理单元形式显示。

(3) Internet 服务管理器(HTML)

基于 HTML 的 Internet 服务管理器，可以使用浏览器对 IIS 5.0 进行远程管理。

(4) NNTP 服务

NNTP(Network News Transfer Protocol,网络新闻传输协议)是 TCP/IP 协议套件的成员，负责将新闻函件分发到 Internet 上的 NNTP 服务器和 NNTP 客户端。设置了 NNTP 后，就可以将新闻文章存储在服务器上的中央数据库，用户可以选择指定的项目阅读。

(5) SMTP 服务

SMTP(Simple Mail Transfer Protocol，简单邮件传输协议)是 TCP/IP 协议套件的成员，用来管理邮件代理之间的电子邮件交换。

(6) Visual InterDev RD 远程配置支持

启动站点服务器上的远程应用程序。

(7) World Wide Web 服务

即 WWW 服务，用于对 Web 站点的管理与访问。

(8) 文档传输协议 FTP 服务器

用于建立 FTP 站点，支持文件的上传和下载。

2.1.3　安装 IIS 5.0

默认情况下，安装 Windows 2000 Server 时，IIS 5.0 被一并安装。在 Windows 2000 Professional 中，用户也可以通过“控制面板”中的“添加/删除程序”选项来安装 IIS。

1．准备安装

IIS 5.0 是 Windows 2000 Professional/Server 的系统组件，可以在相应的操作系统下安装。安装了 IIS 的 Windows 2000 计算机将成为 Intrenet 信息服务器，可以构建 Web 服务、FTP 服务等，从而成为一台 Web 服务器或者 FTP 服务器。

如果局域网有 Internet 连接，那么，在安装 IIS 5.0 时，需要有 ISP 分配的 IP 地址、子网掩码和默认网关。在网络中安装 IIS，还要完成下列工作。

(1) 安装 DNS 服务

在局域网中安装 DNS 服务器，可以使网络中的计算机通过域名访问 Web 站点，也可以保证计算机之间通过域名通信。如果局域网连接到了 Internet，局域网中的 DNS 服务器的域名和 IP 地址应该在其父域(ISP 处)中注册。

(2) 安装 Microsoft FrontPage

建议安装 Microsoft FrontPage，它是一种“所见即所得”的 HTML 编辑软件，可以为用户创建的 Web 站点创建和编辑 Web 页面文件。

(3) 安装 Microsoft Visual InterDev

Visual InterDev 可以为用户创建和开发交互式的 Web 应用程序。

(4) 采用 NTFS 驱动器

为了安全，安装 IIS 的驱动器最好为 NTFS 文件格式，需要在安装 Windows 2000 Server 操作系统时确定。

2．安装 IIS

IIS 可以在安装操作系统时同时安装，也可以在操作系统安装完成后单独安装。如果要单独安装或需要增加或删除 IIS 中的组件，应该按照下面的步骤操作：

(1) 将 Windows 2000 Professional/Server 系统光盘插入光盘驱动器。

(2) 选择菜单“开始”|“设置”|“控制面板”，在“控制面板”窗口中，双击“添加/删除程序”图标，打开“添加/删除程序”对话框，如图 2-1 所示。

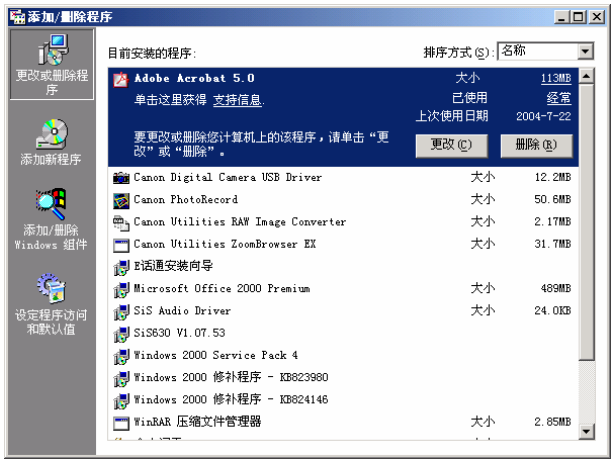


图 2-1 “添加/删除程序”对话框

在“添加/删除程序”对话框中，单击“添加/删除 Windows 组件”图标，打开“Windows 组件向导”对话框，如图 2-2 所示。

图 2-2 安装 Internet 信息服务组件

在组件列表中，选择“Internet 信息服务”，然后单击“详细信息”，选择要安装的 IIS 组件。如果只选择一部分 IIS 组件，则在组件列表中的“Internet 信息服务”前的复选框为灰色。选择结束后，单击“下一步”按钮，如图 2-3 所示。

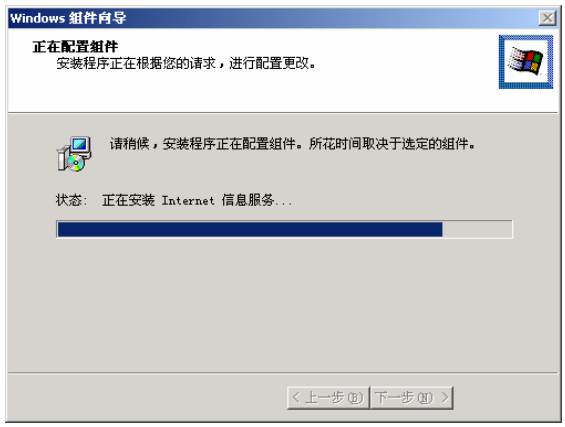


图 2-3　IIS 的安装和配置

向导从光盘中复制文件并进行相关的配置，配置结束后，显示“完成 Windows 组件向导”对话框，此时，单击“完成”按钮，结束 IIS 的安装。

安装结束后，在“控制面板”的“管理工具”中将增加“Internet 服务管理器”程序，Windows 2000 Professional 中安装 IIS 后，“管理工具”窗口如图 2-4 所示。

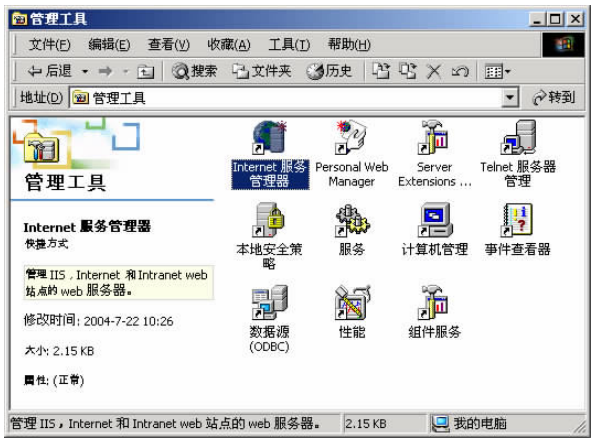


图 2-4　在 Windows 2000 Professional 中安装 IIS 后的“管理工具”窗口

如果要删除 IIS 组件，同样可以通过“添加/删除程序”对话框，在“安装 Internet 信息服务组件”(见图 2-2)的窗口中，将“Internet 信息服务”前面的复选框清除，即可将 IIS 组件从系统中删除。

IIS 安装完成后，服务器的 Inetpub 文件夹下将创建多个文件夹，如图 2-5 所示。

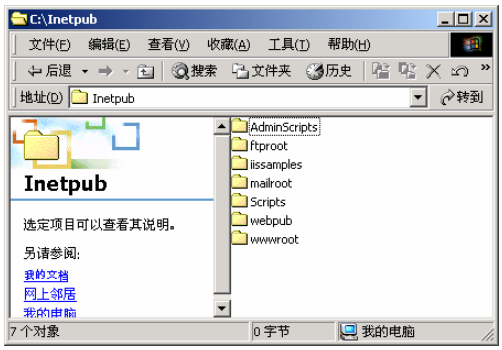


图 2-5 安装 IIS 后相关的文件夹

- ftproot 文件夹 默认为 ftp 站点的主目录。
- iissamples 文件夹 包含许多示例文件。
- mailroot 文件夹 SMTP 服务器的主目录。
- Scripts 文件夹 存储 CGI 脚本的根目录。
- webpub 文件夹。
- wwwroot 文件夹 默认为 Web 站点的主目录。

2.1.4 Internet 信息服务管理器

IIS 5.0 安装完成后，“管理工具”中会增加“Internet 服务管理器”工具。通过 Internet 服务管理器可以监视、配置和控制 Internet 信息服务，创建 Web 站点、FTP 站点，以及对它们进行配置和管理。对 IIS 的管理可以有多种途径，主要包括以下几种。

1．Internet 信息服务管理单元

在 Windows 2000 中，Internet 服务管理器和管理功能集成到一起，作为一个管理单元，称为 Internet 信息服务管理。可以在计算机管理控制台中打开该管理单元，如图 2-6 所示。

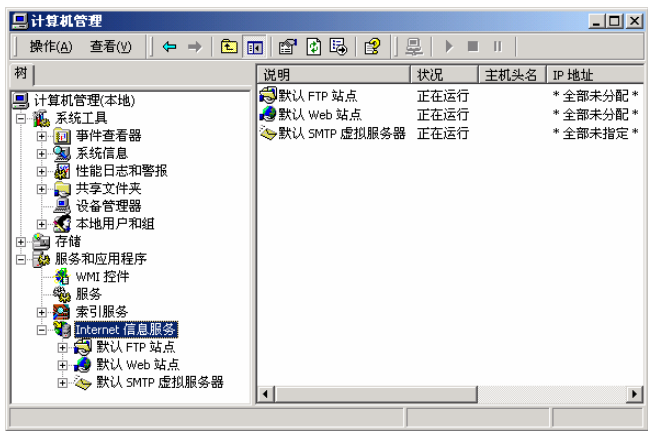


图 2-6 “计算机管理”控制台中的 Internet 信息服务

2．基于浏览器的 Internet 服务管理器(HTML)

在 Windows 2000 中，还提供了基于浏览器的 Internet 服务管理器(HTML)，它允许用户通过 Internet 或 Intranet 远程管理 IIS。

3．Internet 信息服务管理器

一般情况下，用户选择菜单项“开始”|“程序”|“管理工具”，双击“Internet 服务管理器”图标可以直接启动“Internet 信息服务”管理对话框，如图 2-7 所示。

图 2-7　Window 2000 Server 中的“Internet 信息服务”控制台主窗口

在 Windows 2000 Professional 中，安装 IIS 5.0 后，Internet 信息服务中将不包含“管理 Web 站点”和“默认 NNTP 虚拟服务器”。

2.2　Web 站点的构建和配置

利用 IIS 5.0，可以在一台 Windows 2000 Professional/Server 计算机上创建 Web 站点，使得该计算机成为 Internet/Intranet 中的一台 Web 服务器，为用户提供 Web 连接服务。

2.2.1　两个默认的 Web 站点

在安装了 IIS 后，系统会创建两个默认的 Web 站点，即默认 Web 站点和管理 Web 站点，下面介绍这两个默认站点的用途。

1．默认 Web 站点

如果安装了 IIS，就可以在浏览器的地址栏中键入 http://127.0.0.1/，然后按 Enter 键来连接到系统创建的默认 Web 站点。如果系统显示“输入网络密码”对话框，此时可以输入有效的本机账户和密码。也可以在地址栏中输入 http://localhost，而不需要再输入账户和密码，默认的 Web 站点首页如图 2-8 所示。

图 2-8 默认 Web 站点

也可以通过在浏览器的地址栏中键入 http://localhost/iishelp/，然后按 Enter 键来查看 IIS 的文档，如图 2-9 所示。

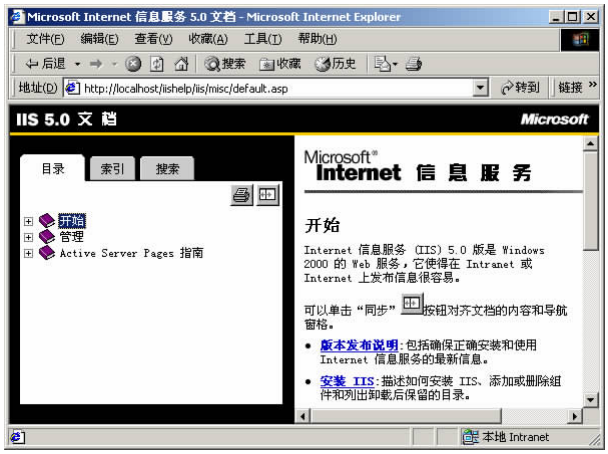


图 2-9 IIS 5.0 文档站点

2．管理 Web 站点

管理 Web 站点中存放了许多 ASP 文件，管理员能够通过浏览器远程管理 IIS 服务器。

2.2.2 连接到 Web 站点

Web 站点是由一系列的文件夹构成的，每个文件夹中包含了一系列的文件和数据。每个 Web 站点都有一个主目录文件夹，该文件夹中包含站点的首页文件。

要连接到一个 Web 站点，应该在浏览器地址栏中输入 Web 站点的 URL。可以用以下 3 种方式之一连接到 Web 站点：

(1) 输入 Web 服务器的 IP 地址，例如 http://211.86.49.1。无论在 Internet 或 Intranet 中，即使在域名解析不正常时，也可以通过 IP 地址连接到一个 Web 站点。

(2) 输入 Web 服务器的 DNS 名称，如果是在局域网中，网络中需要架设 DNS 服务器，来实现 DNS 名称到 IP 地址的解析。

(3) 如果是在 IIS 服务器计算机上，可以输入 http://127.0.0.1 或 http://localhost 访问本机上的 Web 站点。

2.2.3　创建 Web 站点

接下来介绍如何利用 IIS 创建 Web 站点，即构建 Web 服务器，同时还要介绍有关站点的简单操作。

首先需要说明的是，如果 IIS 是安装在 Windows 2000 Professional 上的，则不能创建新的 Web 站点，只能在“默认的 Web 站点”中创建虚拟目录。下面介绍 Windows 2000 Server 操作系统下 IIS 中 Web 站点的创建操作。

在 Windows 2000 Server 上安装了 IIS 后，就可以利用 IIS 创建和管理 Web 站点。选择菜单项“开始”|“程序”|“管理工具”，双击“Internet 服务管理器”，打开“Internet 信息服务”控制台，右击服务器图标，打开快捷菜单，如图 2-10 所示。

图 2-10　新建 Web 站点

在快捷菜单中，选择菜单项“新建”|“Web 站点”，启动“Web 站点创建向导”，然后，单击“下一步”按钮，如图 2-11 所示。

图 2-11　Web 站点创建向导(1)

输入 Web 站点的说明(即新站点的名称)，然后单击“下一步”按钮，如图 2-12 所示。

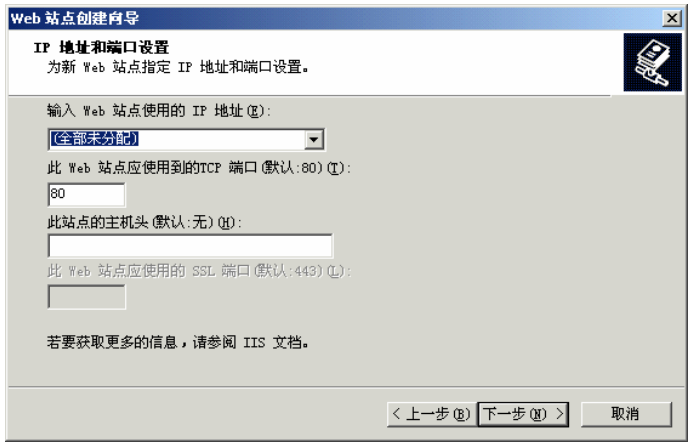


图 2-12 Web 站点创建向导(2)

在 IP 地址后面的下拉列表中，会显示“全部未分配”以及上面设置的多个 IP 地址，从中选择一个 IP 地址。

如果希望对每一个站点采用不同的 IP 地址，用户可以选择默认的端口号 80。如果需要多个 Web 站点使用同一个 IP 地址，可以为不同的 Web 站点指定不同的端口号。指定不同的端口号后，要连接到该站点，在站点的 IP(或域名)后，需要给定对应的端口号，如 http://202.194.7.66:8080。使用非默认的端口号将使客户端连接 Web 站点时，必须知道该站点的端口号，使用不方便。

如果希望使用相同的 IP 地址，又保留 HTTP 默认的端口号 80，还可以使用不同的主机头来区分不同的 Web 站点。这里全部选用默认值，单击“下一步”按钮，如图 2-13 所示。

图 2-13 Web 站点创建向导(3)

在路径下面的文本框中，输入该站点的主目录，该目录保存了该 Web 站点的数据(例如站点的首页 default.html 文件等)。

该路径可以是一个已经建立好的本机路径，也可以是一个合法的域内计算机的共享文

件夹。如果是后者，则需要输入一个网络用户账户(至少具有该共享文件夹的读取权限)和密码，因为当客户浏览网页时(即从远程读取共享文件夹中的文件)，客户需要切换到这个身份来访问数据。

选择“允许匿名访问此 Web 站点”复选框，则用户不需要输入账户和密码就可以浏览该站点 Web 页。然后单击“下一步”按钮，如图 2-14 所示。

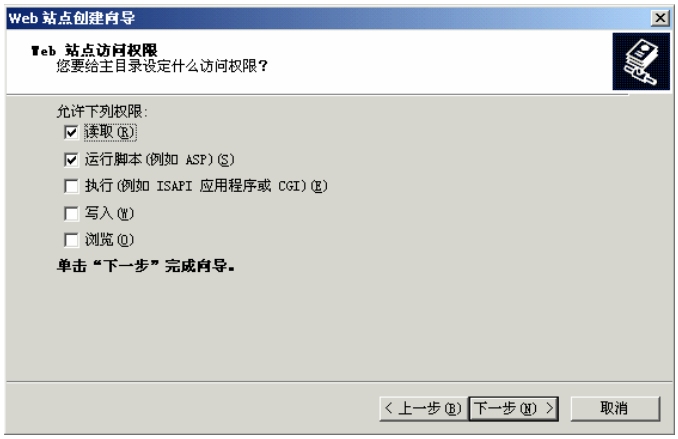


图 2-14　Web 站点创建向导(4)

根据需要对 Web 站点的权限进行设置，从允许的权限中选择相应的权限。一般情况下，需要选择“读取”、“浏览”权限。如果站点中含有脚本程序，还需要选择“运行脚本”权限，否则，脚本将不能运行。有关权限设置的详细介绍，请参见本书 2.3 小节“管理 Web 站点”。

然后单击“下一步”按钮，显示“已经成功完成 Web 站点创建向导”信息窗口。

最后，单击“完成”按钮，返回到“Internet 信息服务”控制台，如图 2-15 所示。

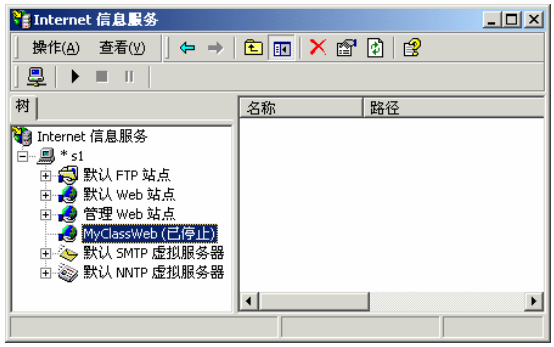


图 2-15　“Internet 信息服务”控制台窗口

新站点刚创建完成后，主目录中没有任何内容。

如果新建的 Web 站点和已经存在的 Web 站点的 IP 地址和端口号完全一样，新站点将被标记为“已停止”。

2.2.4 启动、停止和暂停 Web 站点

上一小节中，由于新建站点和默认 Web 站点 IP 地址和端口号完全一样，使得新建站点被停止。如果要将停止的 Web 站点启动，右击该站点，打开快捷菜单，在快捷菜单中选择“启动”菜单项，该站点将被启动。

如果要停止一个 Web 站点，右击该站点，打开快捷菜单，在快捷菜单中选择“停止”菜单项，该站点将被停止。

当管理人员需要维护系统或网页数据的时候，可以暂停 Web 站点。站点暂停后，它将不接受客户浏览器的连接，等用户工作结束后，再启动该站点。

如果用户试图连接一个暂停的站点，客户端浏览器显示“找不到该页”消息(HTTP 404 – 未找到文件)。如果试图连接一个停止的站点，客户端浏览器显示“该页无法显示”的消息(找不到服务器或 DNS 错误)。

2.2.5 规划 Web 应用

一个 Web 站点建立后，就意味着一个 Web 应用的开始。所谓 Web 应用，是指在 Internet 环境中，应用程序新的开发和使用模式，它是 B/S 结构下程序的实现形式。一个 Web 网站可以简单地看作是一个 Web 应用，它是由主目录下所有的子目录及各种文件构成的。

1．网站首页

一个 Web 应用的开始就是网站的首页。首页(Home Page)是当客户连接到一个站点时首先看到的 Web 页面。可以用 FrontPage 等工具编辑站点的首页文件，首页的默认文件名为 default.htm(特别注意，扩展名不能为 html)，该文件应该保存在 Web 站点的主目录下。用 FrontPage 编辑首页文件 default.htm，并将该文件复制到站点主目录下。

2．规划网站的文件结构

主目录下可以创建子文件夹，用于存放不同类型的文件。例如创建 image 文件夹存放站点中的图像文件，scripts 文件夹存储脚本程序等，如图 2-16 所示。

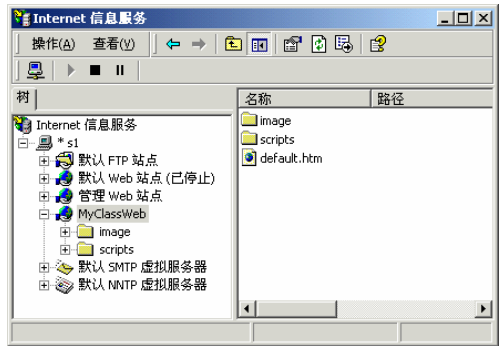


图 2-16 站点主目录内容组织

3．使用虚拟目录

实际存在的主目录及其中的子文件夹，称为物理目录。如果要把本机上其他文件夹，甚至是域中其他计算机上的文件夹加入到主目录下，成为该 Web 站点的内容，则需要虚拟目录。虚拟目录可以看作是 Web 站点主目录下指向其他物理目录的指针。

(1) 使用虚拟目录的优点

使用虚拟目录可以将 Web 站点的数据保存到本机上主目录以外的物理目录，甚至是其他的计算机中。避免 Web 站点数据占用服务器太多的空间。

当数据移动到其他地址时，不会影响 Web 站点结构。此时不需要更改虚拟目录的名称，只需要重设虚拟目录，将虚拟目录指向新的物理目录即可。

(2) 建立虚拟目录

要建立虚拟目录，可按照下面的步骤操作：

在“Internet 信息服务”控制台目录树中，右单击某 Web 站点，打开快捷菜单，选择菜单项“新建”|“虚拟目录”，启动“虚拟目录创建向导”，如图 2-17 所示。

图 2-17　“虚拟目录创建向导”(1)

单击“下一步”，如图 2-18 所示。

图 2-18　“虚拟目录创建向导”(2)

输入虚拟目录名称，该名称将显示在 Internet 信息服务控制台相应的 Web 站点下，单击“下一步”按钮，如图 2-19 所示。

在目录下面的文本框中输入虚拟目录对应的实际物理目录，或者单击文本框后面的“浏览…”按钮选择需要的物理目录。然后单击“下一步”按钮，如图 2-20 所示。

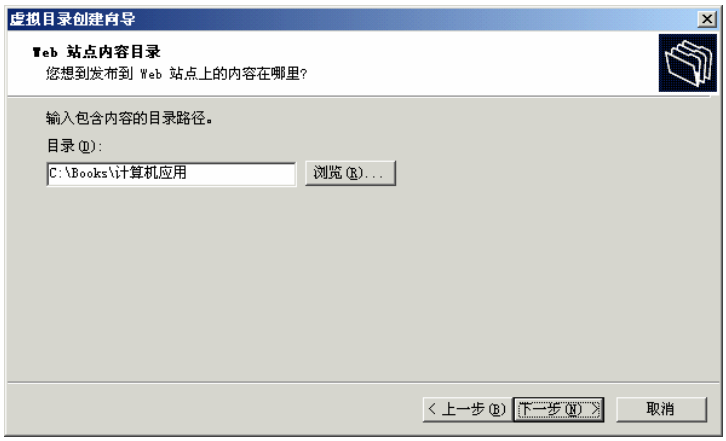


图 2-19 “虚拟目录创建向导”(3)

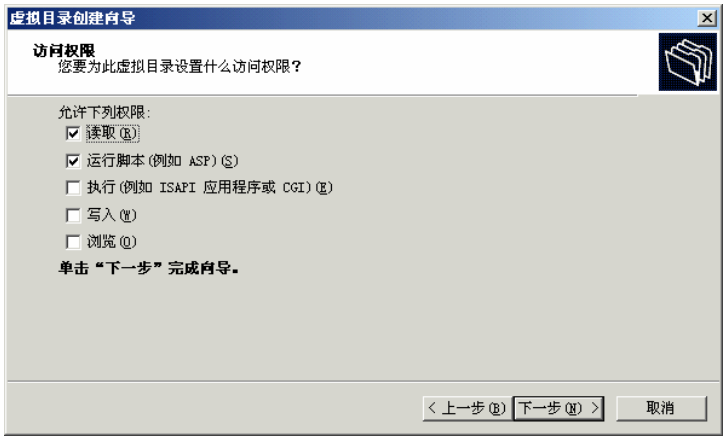


图 2-20 “虚拟目录创建向导”(4)

根据需要设置虚拟目录的访问权限，然后单击“下一步”按钮，显示虚拟目录创建向导完成屏幕。最后单击“完成”按钮，返回“Internet 信息服务”控制台，打开“Internet 信息服务”窗口，将看到新建的虚拟目录，如图 2-21 所示。

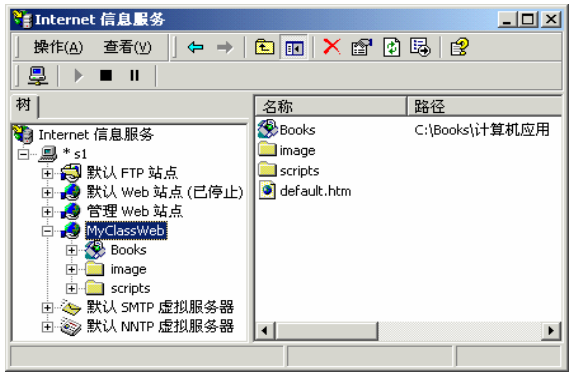


图 2-21 “Internet 信息服务”控制台

使用虚拟目录，可以实现在一个 Web 应用中，简单地增加其他页面，而这些页面和现有的内容可能没有直接关系，它可能是临时性的。此时只需要在浏览器的地址栏输入“ http://域名 ”或“ IP 地址/虚拟目录/文件名(包括括展名) ”，即可访问虚拟目录中的内容，从而可以简化站点的管理。这些内容可以被删除而不会影响原有的站点结构。

例如，在本地计算机的 Web 站点上建立一个名称为 MyJSP 的虚拟目录，对应实际物理目录是 d:\MyJSP，里面包含文件 myJSPHome.htm，要浏览该网页，在地址栏中输入 http://127.0.0.1/MyJSP/myjsphome.htm 即可(Windows 系统内的目录名文件名不区分大、小写)。

2.2.6　运行多个 Web 站点

在一台 IIS 服务器上，可以创建并运行多个 Web 站点。在 Windows 2000 Server 中，IIS 5.0 可以通过三种不同的方式使多个 Web 站点在同一台服务器上同时运行：

第一，不同的 Web 站点使用不同的 IP 地址。

第二，不同的 Web 站点使用相同的 IP 地址、不同的端口。

第三，不同的 Web 站点使用相同的 IP 地址和端口号，但使用不同的主机名。

1．增加 IP 地址

对网卡指定多个 IP 地址，把不同的 IP 地址分给不同的 Web 站点可以在一台服务器上同时运行多个 Web 站点。

在服务器的“ 网络和拨号连接 ”文件夹中，右击“ 本地连接 ”图标，在快捷菜单中选择“ 属性 ”对话框，打开“ 本地连接属性 ”对话框，在“ 常规 ”选项卡中，选择“ Internet 协议(TCP/IP) ”，单击“ 属性 ”按钮，打开“ Internet 协议(TCP/IP)属性 ”对话框，如图 2-22 所示。

单击“ 高级 ”按钮，打开“ 高级 TCP/IP 设置 ”对话框，如图 2-23 所示。

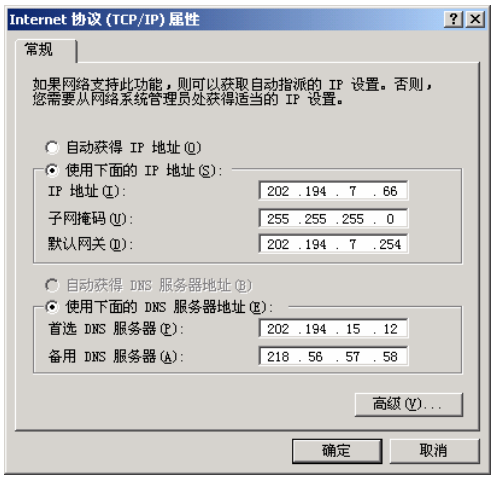


图 2-22　“ TCP/IP 协议属性 ”对话框

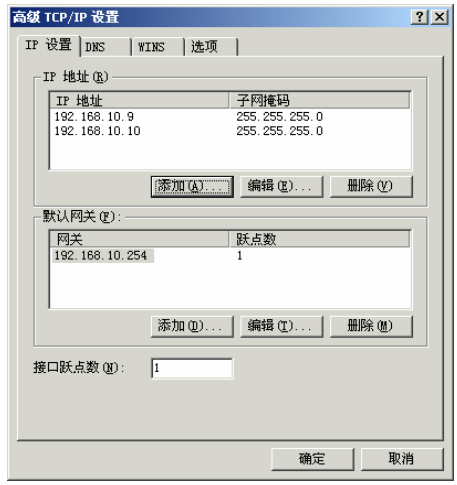


图 2-23　“ 高级 TCP/IP 设置 ”对话框

在 IP 设置选项卡中， 在“ IP 地址 ”区域单击“ 添加 ”按钮，输入新的 IP 地址和子网掩码。这样本地连接就对应了多个 IP 地址。

2．建立并运行多个站点

按照第 2.2.3 小节的步骤建立多个 Web 站点。为保证多个 Web 站点的同时运行，可以为不同的站点选择不同的 IP 地址；或者相同的 IP 地址，而不同的端口号；或者相同的 IP 地址和端口号相同，但主机名不同。

例如可以建立两个 Web 站点 MyPrivateWeb 和 MyStudyWeb，IP 地址相同(均为 202.194.7.66)，端口号不同，前者端口号为 8080，后者默认的端口号为 80。因此客户端要连接到 MyPrivateWeb，在浏览器的地址栏中应该输入 http:// 202.194.7.66:8080。

除了使用 IP 地址和端口号连接到一个 Web 站点外，还可以设置站点的主机头和增加局域网中的 DNS 主机记录，通过域名访问新建的 Web 站点。

在“ Internet 信息服务 ”控制台目录树中，右击 Web 站点目录(如图 2-21 中的 MyClassWeb)，选择“ 属性 ”菜单项，打开站点属性对话框，如图 2-24 所示。

在“ Web 站点 ”的“ Web 站点标识 ”区域中，单击“ 高级… ”按钮，打开“ 高级多 Web 站点配置 ”对话框，如图 2-25 所示。

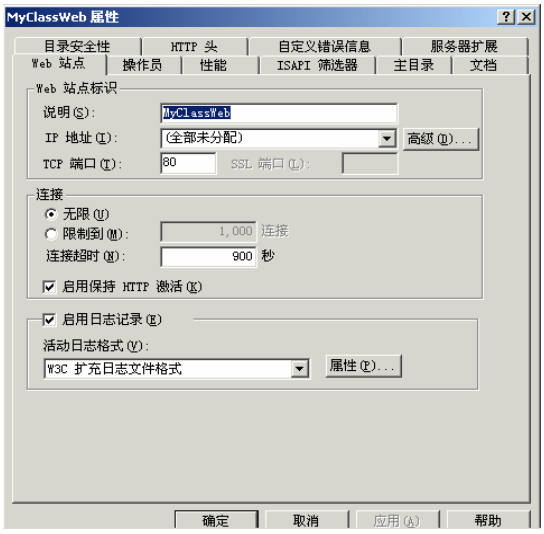


图 2-24 “ Web 站点属性 ”对话框

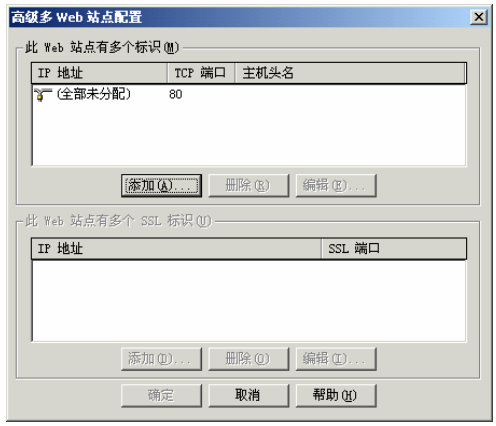


图 2-25 “ 高级多 Web 站点配置 ”对话框

单击“ 添加 ”按钮，打开“ 高级 Web 站点标识 ”对话框，如图 2-26 所示。

输入该站点的 IP 地址、端口号和主机头名，主机头名就是该站点的 DNS 域名。要使客户能够通过域名连接到 Web 站点，需要在局域网中架设 DNS 服务器，增加 Web 站点 IP 地址到域名的主机记录。

关于 DNS 服务器的配置，请参考 Windows 2000

图 2-26 “ 高级 Web 站点标识 ”对话框

Server 中 DNS 的配置和管理，具体介绍略。当为一台 Web 服务器主机增加域名后，要使得客户机能够通过域名访问 Web 服务器，还需要在客户端进行设置。

在客户端，右击“本地连接”，在“本地连接属性”的“TCP/IP 属性”对话框(见图 2-22)中，指定 DNS 服务器的 IP 地址。这样，就可以通过域名连接 Web 站点了。

如果在连接新建的 Web 站点时出现“输入网络密码”对话框，可以在控制台目录树中，在新建 Web 站点上右击，在快捷菜单中选择菜单项“所有任务”|“权限向导”，按照权限向导的提示完成该站点的权限设置，全部采用默认设置即可。

此后在浏览器地址栏内重新输入网站地址将不再出现“输入网络密码”对话框。如果在服务器本机测试，可以输入 http://127.0.0.1/，按回车键。如果服务器上建立了 Internet 连接，而线路不通，此时会出现“脱机连接”对话框，单击“重试”按钮。

2.3　管理 Web 站点

当 Web 站点建立后，还需要对 Web 站点进行管理，管理 Web 站点是通过 Web 站点的属性来完成的。在“Internet 信息服务”控制台目录树中，右击站点目录，执行“属性”命令，打开站点属性对话框，如图 2-27 所示。

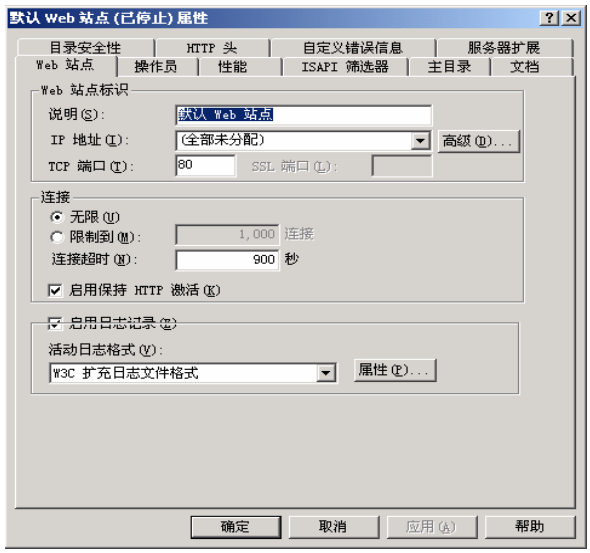


图 2-27　Web 站点属性对话框

2.3.1　“Web 站点”选项卡

在 Web 站点选项卡中，包括以下三个区域的设置。

(1) Web 站点标志

- 说明　输入对该站点的说明性文字，该文字将作为站点名字出现在 Internet 信息服

务管理器控制台目录树中。

- IP 地址　设置此站点要使用的 IP 地址，如果计算机中设置了多个 IP 地址，可以选择其中一个。如果该 IIS 服务器上同时运行多个 Web 站点，单击“高级”按钮，进行进一步的设置。
- TCP 端口　HTTP 服务的默认端口为 80，如果设置其他的端口，客户浏览器在浏览该网站时，在 URL 中需要给出端口号，例如 http://www.teacher.local:8080。

(2) 连接

- 限制到　当 IIS 服务器的内存和网络带宽较小时，可以限制该 Web 站点可以连接的客户数量。选择“限制到”单选钮，可以指定该站点最多的连接数量。
- 连接超时　是指如果客户端建立了连接，在连接超时规定的时间内没有访问操作，系统将该连接强制断开。
- 启用保持 HTTP 激活　是指如果一个网页中插入了其他文件(如图片、动画等)，让网页和其中的文件通过一个连接传送，从而降低 Web 站点的负担。

如果清除“启用保持 HTTP 激活”复选框，当网页中包含多个文件连接时，客户端每下载一个文件就要与 Web 服务器建立一个连接，这样会大大降低 Web 服务器的执行性能。

(3) 启用日志记录

选择该选项将启用 Web 站点的日志记录功能，该功能可记录用户活动的细节并以选择的格式创建日志。启用日志记录后，需要在“活动日志格式”列表中选择格式。

可以选择的活动日志的格式包括以下几种。

- Microsoft IIS 日志格式　固定 ASCII 格式。
- ODBC 日志　仅在 Windows 2000 Server 中提供，记录到数据库的固定格式。
- W3C 扩充日志文件格式　可自定义的 ASCII 格式，默认情况下选择该格式。必须选择该格式才能使用“进程账号”。

单击“属性”按钮可以配置日志文件创建选项(如按周，或按文件大小)，或者配置 W3C 扩充日志记录或 ODBC 日志记录的属性。

2.3.2 “目录安全性”选项卡

Web 站点的安全性设置主要是通过“目录安全性”选项卡完成的。在介绍具体的安全性设置以前，先介绍 Web 站点的访问机制。

当客户端通过浏览器向 Web 站点发出访问某个页面的请求时，Web 站点收到客户的请求后，将启动一个验证过程，来决定是否将相关网页传送给客户端。具体的验证过程如图 2-28 所示。

通过验证过程的验证后，如果网页是 HTML 类型的，则 Web 服务器将把该网页直接传送到客户端浏览器。如果网页为 ASP、JSP 等含有服务器脚本的文件，Web 服务器将先在服务器端执行该文件，然后将执行的结果传送给客户端浏览器。

要进行 Web 站点的安全性设置，在站点属性对话框(图 2-27)中选择“目录安全性”选项卡，如图 2-29 所示。

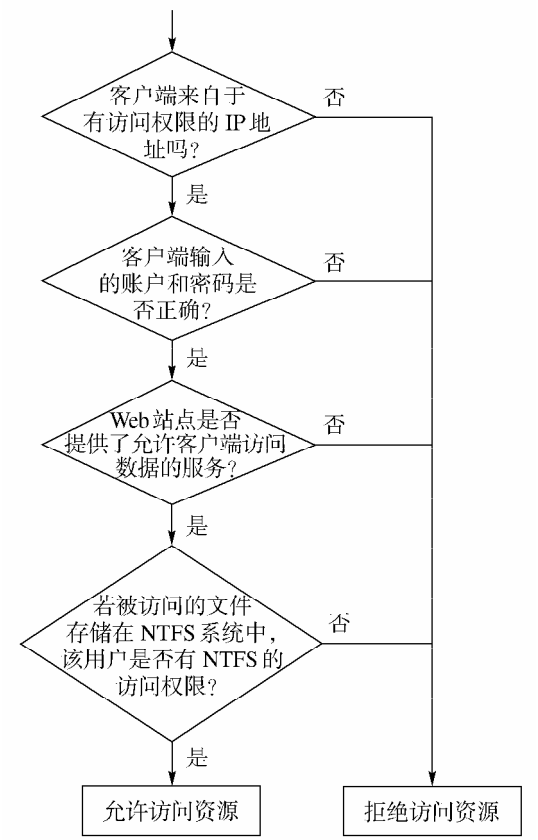


图 2-28　访问请求的验证过程

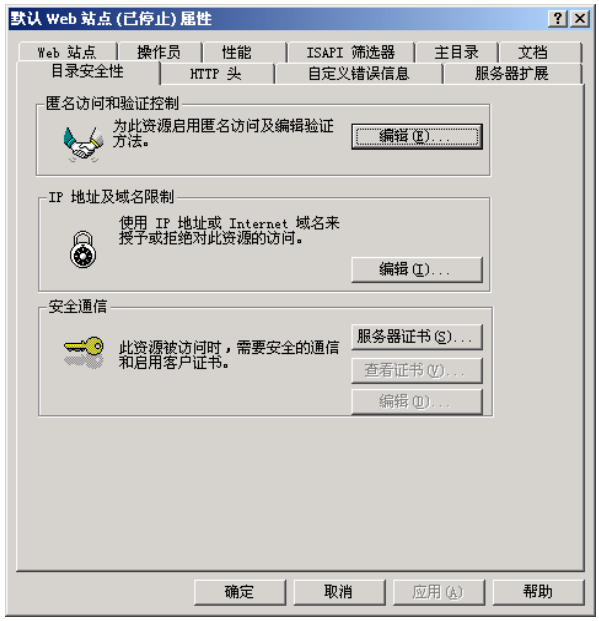


图 2-29　默认 Web 站点属性“对话框”-“目录安全性”选项卡

(1) IP 地址及域名限制

在 IP 地址及域名限制区域中，单击“编辑”按钮，如图 2-30 所示。

在“IP 地址及域名限制”对话框中，选择“授权访问”，然后单击“添加”按钮，可以指定不能访问该站点的 IP 地址。类似地，选择“拒绝访问”单选钮，通过添加，可以指定在拒绝访问中，能够访问该站点的 IP 地址清单。当用户来自拒绝访问的 IP 地址时，客户端浏览器会收到“您没有权限查看网页”的提示信息。

一般情况下，如果网站是公开的，则选择“授权访问”，然后单击“添加”按钮，把不被欢迎的 IP 地址列出。相反，如果网站是一个特殊的站点，只允许部分人访问，则选择“拒绝访问”，然后把可以访问的 IP 列出。

(2) 匿名访问和验证控制

当 Web 站点验证了客户端的 IP 地址后，接下来查看该站点是否允许匿名访问。如果站点不允许匿名访问，或者客户端要访问的文件有特殊的 NTFS 限制，此时客户端需要输入用户账户和密码。

当 Web 站点允许匿名访问时，客户端不需要输入账户和密码就可以访问网站的数据，此时 Web 站点会尝试用 Internet Guest Account 账号“IUSER_计算机名称”这个内部账户让计算机登录。要设置匿名访问，在“匿名访问和验证控制”区域中，单击“编辑”按钮，打开“验证方法”对话框，如图 2-31 所示。

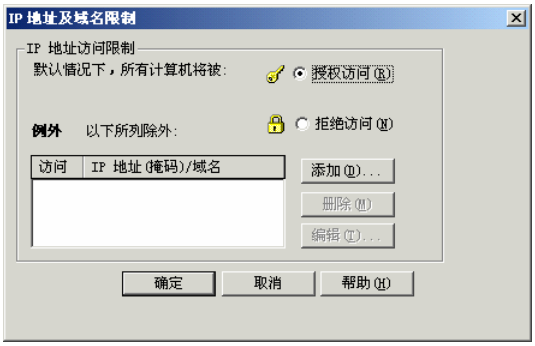


图 2-30 “IP 地址及域名限制”对话框

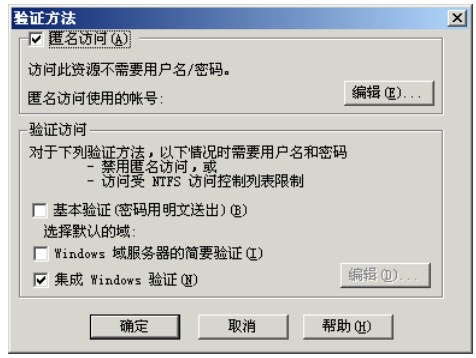


图 2-31 “验证方法”对话框

选择“匿名访问”。单击“编辑”按钮，打开“匿名用户账号”对话框。在该对话框中，可以指定用于匿名访问的匿名用户账号。匿名访问使得每个人都可以使用上述账号访问 Web 网站。如果匿名账户没有足够的 NTFS 权限，系统会根据在“验证访问”区域中选择的验证方式，要求用户输入账号和密码，如果未选择任何验证方法，则系统不提示用户输入账户和密码，而是直接拒绝用户对该页的访问。

一般情况下，如果 Web 站点连接到 Internet，一般选择允许匿名访问。

(3) 使用权限向导

如果 Web 站点设置了匿名访问，当用户访问该站点时，仍然出现“输入网络密码”对话框，这是由于匿名账户未拥有要访问的 NTFS 权限。

为了避免遗漏对 Web 主站点的 NTFS 权限设置，导致客户端不能正常地访问所需要的 Web 页，Internet 信息服务控制台中提供了“权限向导”。右击站点，在快捷菜单中选择菜

单项“所有任务”|“权限向导”，启动“IIS 5.0 权限向导”。接下来，按照向导提示操作，一般取默认值，最后单击“完成”按钮。再次访问该站点，看能否成功。

2.3.3　“主目录”选项卡

根据上面有关用户访问 Web 站点的验证过程，当用户通过身份验证后， Web 站点会根据站点的权限设置，来决定可以提供给用户的服务，例如从网站浏览网页(下载文件)、上传文件等。在网站属性对话框中，单击“主目录”选项卡，如图 2-32 所示。

(1) 访问权限设置

- 读取　默认状态下 Web 站点拥有读取权限，即站点提供用户读取服务器上文件的权限，用户可以从站点中下载文件。
  如果要下载的文件存储在 NTFS 驱动器中，还应该查看 NTFS 对文件的属性设置。
- 写入　允许用户上传文件，或提交表单改变网页内容。它同样受 NTFS 对要上传到的目标文件夹属性的影响。
- 目录浏览　允许用户浏览站点目录，当用户通过浏览器连接到本站点时，如果未指定文件名和目录，站点也没有启用默认文档(参见图 2-34)，或默认文档不存在，将看到此站点的目录列表(不显示虚拟目录)，如图 2-33 所示。

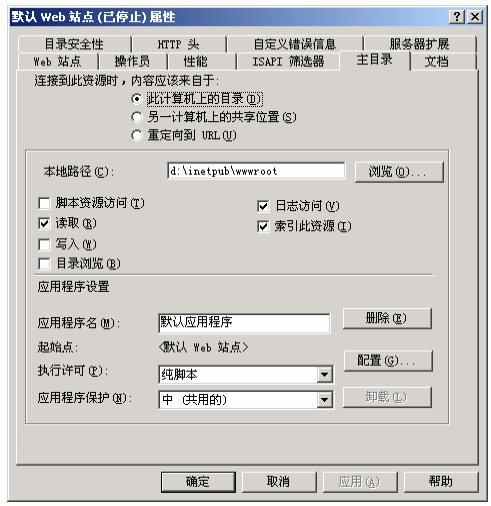


图 2-32　网站属性对话框 -“主目录”选项卡

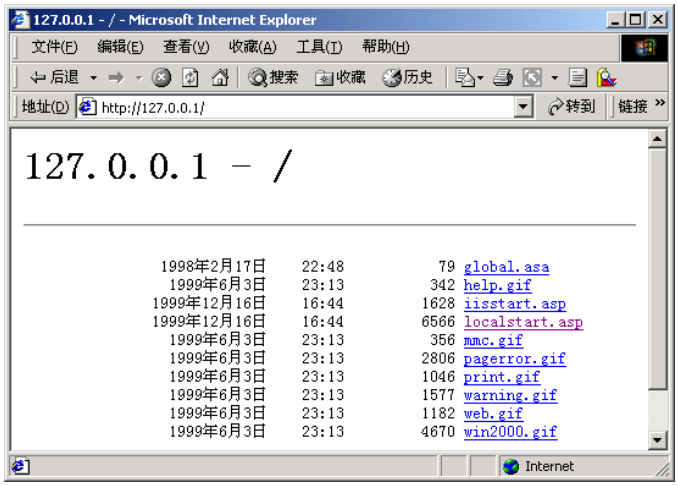


图 2-33　网站的目录浏览界面

(2) 应用程序设置

应用程序设置可以指定何种应用程序可以在 Web 站点执行，在“执行许可”列表中包括“无”、“纯脚本”和“脚本和可执行程序”选项。

如果选择“无”，则不允许在 Web 站点中运行程序(包括服务器端 ASP 脚本)，当浏览一个 ASP 页时，会显示“网页无法显示”，在页面中提示：“您试图从目录中执行 CGI、ISAPI 或其他可执行程序，但该目录不允许执行程序”。因此，如果所建站点中包括服务器端脚本程序，不应该选择“无”。

选择“纯脚本”，则只能执行 ASP 程序等。选择“脚本和可执行程序”，则所有的应用程序(包括 EXE 文件和 DLL 库)都可以在 Web 站点上执行。

2.3.4 “文档”选项卡

下面介绍如何设置站点的默认文档，即相当于站点的首页，默认文档是 HTM 文件或者 ASP 文件。当用户通过浏览器连接到 Web 站点时，如果没有指定要浏览的文档，Web 站点则将默认文档传送给客户端浏览器。

在 Web 站点属性对话框中，选择“文档”选项卡，如图 2-34 所示。

图 2-34 网站属性对话框 -“文档”选项卡

选择“启用默认文档”复选框，用户也可以单击“添加”按钮，增加一个新的默认文档，如 index.htm、startpage.htm 等。

如果有多个默认文档，系统将把排在前面的文档优先传送给客户端浏览器。

如果选择“启用文档页角”复选框，则服务器在传送所请求的网页之前，会在文档的底部插入页角文字，然后再传送。

文档页角也对应一个 HTM 文件，这个 HTM 文件不是一个包含<HTML></HTML>，

<body></body>等标记的完整的 HTM 文件，只能包含文字的大小和颜色设置，如：

```
<h3 align=right>学习站点</h3>
```

文件可以用 Windows 操作系统中的记事本程序编辑，并保存为.htm 类型的文件。

2.3.5　“操作员”选项卡

通过“操作员”选项卡可以指定 Web 站点的操作员，默认状态下 Administrators 组成员是 Web 站点的操作员，并且不能删除。“操作员”选项卡如图 2-35 所示。

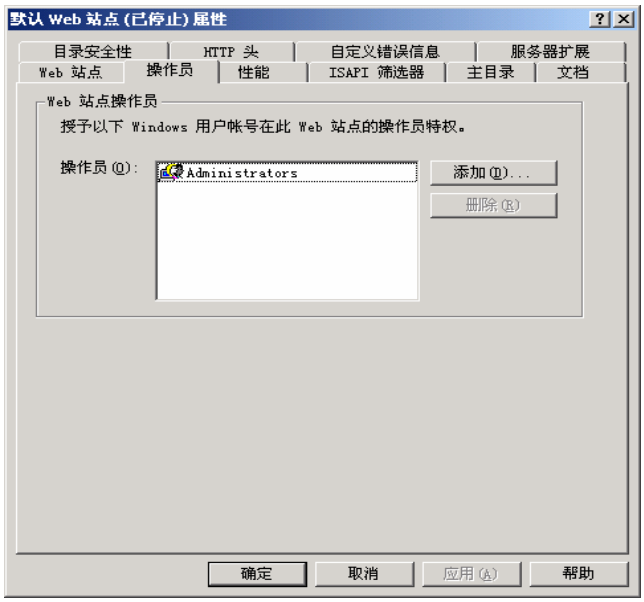


图 2-35　网站属性对话框 -“操作员”选项卡

Administrators 组成员还可以为 Web 站点增加新的操作员。单击“添加”按钮，在“选择用户和组”对话框中选择用户账户，单击“添加”按钮，来增加新的 Web 站点管理员。

与 Administrators 组成员的权限相比，新增加的 Web 站点管理员不具有下列权限：

- 改变站点标志
- 改变匿名访问的用户账户和密码
- 建立虚拟目录
- 启用带宽节流控制

可以使用新增加的操作员，完成下列任务：

- 可以设置 Web 站点的权限
- 启用记录
- 启用默认文档和文档页角
- 启用属性的到期限制，自定义 HTTP 头与设置内容分级

2.3.6 “自定义错误”选项卡

当用户连接到 Web 站点时，可能因为服务器本身的错误或权限不足的原因，导致站点不能回应客户端的请求，此时便返回默认错误信息。

使用 Web 站点的“自定义错误信息”选项卡，可以修改返回到客户端浏览器的错误信息提示。

在站点属性对话框中，选择“自定义错误信息”选项卡，如图 2-36 所示。

图 2-36 网站属性对话框 -“自定义错误信息”选项卡

在 HTTP 错误信息列表中，列出了发生各种错误时返回到客户端的错误提示页面，这些错误提示页面存储在\WINNT\help\iishelp\common 文件夹中。

要自定义错误信息，可以在站点主目录下创建一个保存错误信息的文件夹(例如 help 文件夹)，将每个错误信息编辑成 html 文件。然后，在“HTTP 错误信息”列表中，选中一个错误列表项，再单击“编辑属性”按钮，打开“错误映射属性”对话框，输入该错误对应的错误提示 Web 页文件路径。

管理员通过修改错误提示页,可以把特定的信息传送给客户,因为如果客户在连接 Web 站点时发生问题，这些页面将显示在客户端浏览器中。

2.3.7 “性能”选项卡

一般情况下，性能调整往往针对整个服务器计算机，有时候用户也可以对单个 Web 站点进行性能调整。在 Web 站点属性对话框中，选择“性能”选项卡，如图 2-37 所示。

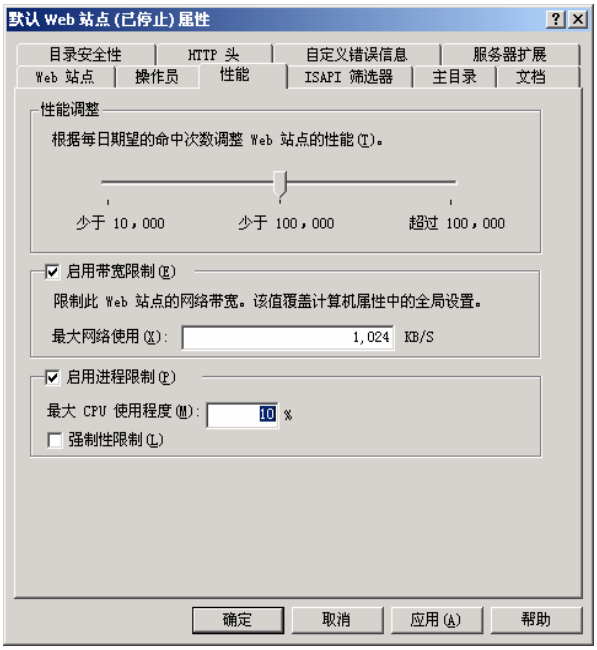


图 2-37　网站属性对话框 -“性能”选项卡

(1) 性能调整

在 Web 站点“性能”选项卡中，微软公司把 Web 站点每天可能的连接数目称为命中次数，Web 站点的连接数目越大，该站点占用的服务器系统资源(内存、CPU 等)就愈多。可以通过调整命中次数，来调整服务器与留给 Web 站点的系统资源。

如果要提高 Web 站点的连接速度,应该使估计的每日连接数目比实际的连接数目要大。如果为一个 Web 站点设置的值过大，将会造成系统资源的浪费，降低服务器的整体性能。

(2) 启用应用程序限制

如果在 Web 站点上执行应用程序，为避免应用程序占用过多的 CPU 时间，影响服务器的性能，可以选择“启用进程限制”复选框，来设置应用程序对 CPU 使用时间的上界。如果不选择“强制性限制”，则当应用程序占用 CPU 时间超过规定的上界时，只是在时间监视器中警告。

2.3.8　“HTTP 头”选项卡

HTTP 头(HTTP Header)是对现有 HTTP 标准的扩充，它有许多复杂的应用，在“HTTP 头”选项卡中可以对站点进行 4 种设置，如图 2-38 所示。

选择“启动内容失效”复选框，可以设置此站点内容到期的时间。当用户浏览一个站点的某个网页时，服务器首先将浏览器要访问的 Web 页的 URL 返回到客户端，客户端在本地硬盘的网页缓存中查找是否存在该页面，如不存在，将要求服务器传送该页面。否则，浏览器将对要下载 Web 页的当前日期和到期日期进行比较，来决定是显示客户端硬盘中网页缓存的页面，还是从 Web 站点下载新的网页。

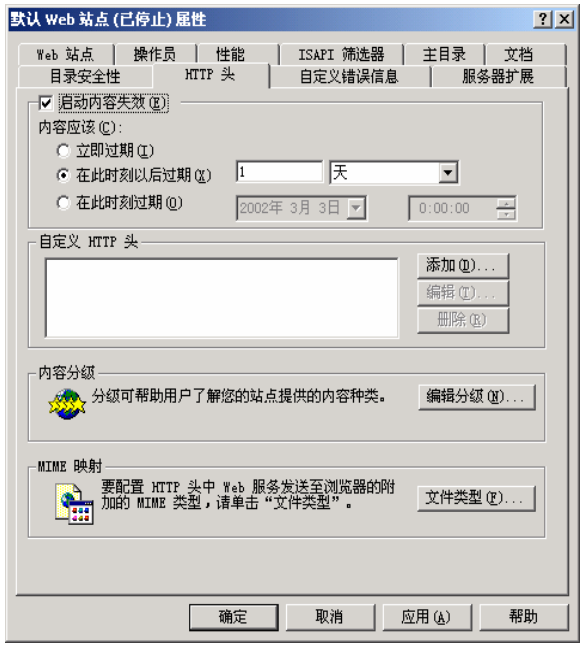


图 2-38　网站属性对话框 -“ HTTP 头 ”选项卡

选择“ 立即过期 ”，则网页内容一下载到浏览器端该页面就过期了。因此，浏览器每次连接到该网站时，无论客户端的本地网页缓存是否存在对应的页面，页面都会被重新下载。它适合于一些显示即时行情的网站，如股市。

选择“ 到此时刻以后 ”，设置网页的有效期，当浏览器连接到该站点浏览网页时，网页被保存在客户端的缓存文件夹中，有效期过后，该网页将自动从客户端缓存中删除。此设置适合于一些需进行固定时间更新的新闻站点和页面。

选择“ 到此时刻过期 ”和“ 到此时刻以后 ”类似，用于设置网页的有效期。

2.4　使用 Apache 和 Tomcat

在 Windows NT/2000 Server 操作系统下，除了使用内置的 IIS 来建立和管理 WWW 站点外，用户还可以使用第三方的软件来建立和管理 Web 应用。常用的软件有 Apache Tomcat，主要用于基于 Java 的 Web 应用，由于它是第三方软件与 IIS 相比，在 Web 应用的管理上要复杂一些。

在 Windows 2000 服务器中，如果已经配置了 Internet 信息服务器 IIS，要使用 Apache 和 Tomcat，需要首先将 Window 的 IIS 服务停止或禁用。对于 Web 服务，IIS 和 Apache Tomcat 都提供了很好的功能，用户可以根据操作系统平台(Windows 平台或者 Unix、Linux 等平台)的需要，以及 Web 开发工具(ASP 或者 JSP)的不同来选择是使用 IIS，还是 Apache Tomcat。

2.4.1　Apache 与 Tomcat

Apache 是使用最广的 Web 服务器之一，它可以运行在几乎所有广泛使用的计算机平台上，以高效、稳定、安全、免费而著称，超过 50%的 Web 服务器采用 Apache。用户可以从 Apache 网站(http://www.apache.org/dist/httpd/binaries/win32)下载 Apache 服务器软件。下载时记住选择 for Win32 的无源代码版本(即*-win32-no_src.msi)。最新版的 Apache for Win32 开始使用.msi 的形式发布[1]，从而使 Windows 环境下安装 Apache 变得非常简单。

Apache 服务器拥有以下特性：

- 支持最新的 HTTP 1.1 通信协议。
- 拥有简单而强有力的基于文件的配置过程。
- 支持通用网关接口 CGI。
- 支持基于 IP 和基于域名的虚拟主机。
- 支持多种方式的 HTTP 认证。
- 集成 Perl 处理模块。
- 集成代理服务器模块。
- 支持实时监视服务器状态和定制服务器日志。
- 支持服务器端包含指令(SSI)。
- 支持安全 Socket 层(SSL)。
- 提供用户会话过程的跟踪。
- 支持 FastCGI。
- 通过第三方模块可以支持 Java Servlets。

Tomcat 是针对于 Apache 服务器开发的 JSP 应用服务器，是 Java Servlet 和 Java Server Pages 技术的标准实现，是基于 Apache 许可证下开发的一种自由软件，可以从网站 http://jakarta.apache.org/site/binindex.cgi 下载不同的 Apache Tomcat 版本。

当在一台机器上配置好 Apache 服务器后，可利用它响应对 HTML 页面的访问请求。Tomcat 部分是 Apache 服务器的扩展，但当运行 Tomcat 时，它实际上是作为一个与 Apache 独立的进程独立运行的。当配置正确时，Apache 为 HIML 页面服务，而 Tomcat 实际上运行 JSP 页面和 Servlet。

要在 Apache 下运行 JSP，最好的方案就是选择 Tomcat，它具有免费、集成度好的优点，缺点是界面不够直观，如果是在 Windows 2000 Professional/Server 平台上，还需要设置环境变量，相对来讲比较麻烦。

2.4.2　Apache 的安装和配置

登录 Apache 网站(http://www.apache.org/dist/httpd/binaries/win32)，选择 for Win32 的无

1 .msi 文件类型是一种可以安装的程序包文件，双击带.msi 扩展名的文件时，操作系统将.msi 文件与 Windows 安装程序关联并运行客户端安装程序服务 msiexec.exe。

源代码版本 apache_1.3.29-win32-x86-no_src.msi 下载(只有 2.23M)。

注意：由于软件版本的不断更新，在读者阅读本书时，上面提到的页面可能不再提供 Apache 1.3.29 版本,而是提供更新的版本下载。同样,对于 2.4.3 小节中的 J2SDK 和 Tomcat，也有类似情况。对于新的软件版本，本书内提到的安装和设置方法依然有效。但需要注意，在安装和设置过程中，相关的目录或参数中包含的软件版本号需要替换为新的版本号。

1．Apache 的安装

当 Apache 服务器下载后，可以按照下列步骤完成 Apache Web 服务器的安装和配置。

(1) 双击 Apache 的安装文件，执行安装向导，如图 2-39 所示。

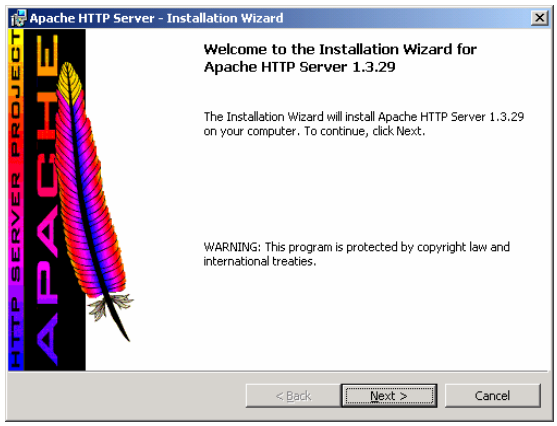


图 2-39 Apache 服务器的安装

(2) 单击 Next 按钮，按照向导提示，分别输入 Network Domain(网络域名，如 xxx.com)，Server Domain(服务器域名，如 www.xxx.com)和网站管理员的 E-mail 地址，按照网站的实际情况填写。如果是个人用户，可能没有上述数据，可按格式填写临时的名字。如图 2-40 所示。

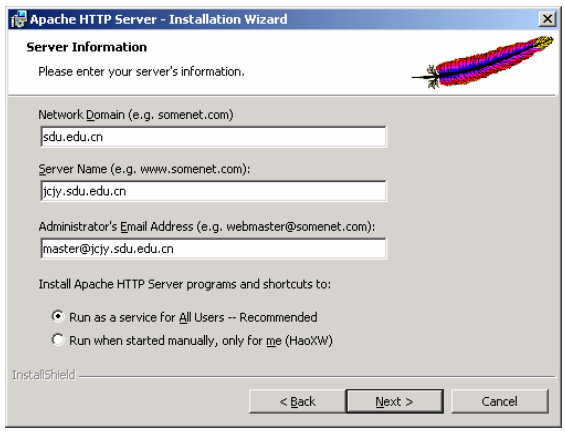


图 2-40 Apache 安装向导输入信息屏幕

(3) 按照向导提示选择安装路径，直至安装完成。

Apache 安装完成后，在“开始”菜单中增加 Apache HTTP Server 程序组。

不再需要重新开机，Apache 会自动启动，此时在 IE 地址栏里输入 http://localhost 或 http://127.0.0.1 可看到默认的 Apache 首页，如图 2-41 所示。

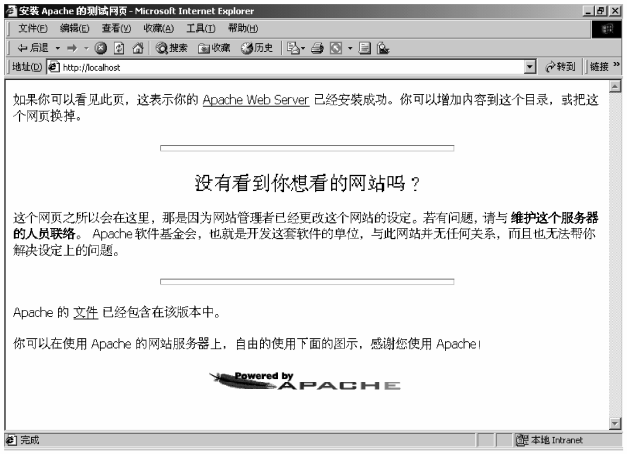


图 2-41　Apache 安装成功后的首页

Apache 服务安装成功后，在控制面板、管理工具文件夹下，双击“服务”图标，显示 Apache 已经启动，以后 Apache 将作为一项服务，随着机器的启动而自动运行。

需要说明的是，如果计算机上已经安装了 IIS，输入 http://localhost 后将首先显示如图 2-8 所示的默认 Web 站点。此时，可以按照第 2.2.4 小节的介绍停止 IIS 中的 Web 服务器，或者给 IIS 中的 Web 服务指定一个不同的端口号(不同于默认的 80 端口)。然后再输入 http://localhost，即可显示 Apache 服务。此时还可以通过 http://localhost: 端口号来继续打开 IIS 中的 Web 服务。

2．Apache 的配置

Apache 的核心配置文件是文件 httpd.conf，默认情况下，它的存储位置为 C:\Program Files\Apache Group\Apache\Conf\。用记事本打开它，可以看到这些配置文件都以文本方式存在，其中“#”为 Apache 的注释符号。可以在记事本菜单中选择“编辑”|“查找”选项，逐一输入下面要配置的关键字，并进行相应配置。

此外，选择菜单项“开始”|“程序”| Apache HTTP Server | Edit the Apache httpd.conf Configuration File 也可以，在最新的 1.3.26 版中，它的作用更加明显。

(1) 配置 DocumentRoot

这个语句指定网站路径，也就是主页放置的目录。可以使用默认的，一般就是 Apache 安装目录下的一个子目录，也可以自己指定，需要注意，这句末尾不要加“\”。此外，路径的分隔符在 Apache Server 里写成“\”。例如，可以在此处将其设置为 D:\GSL1.0，打开主页时，默认打开的文档就直接去该目录下查找。

(2) 配置 DirectoryIndex

这是站点默认显示的主页，一般情况下，在此处还可以加入 Index.htm Index.php

Index.php3 Index.cgi Index.pl Default.htm 等。注意，每种类型之间都要留一空格。

上面两步设置完成后，启动 IE，输入 IP 即可访问自己的 Web 站点。还可以在该文件的 ServerName 处定义域名，在 ServerAdmin 处输入 E-mail 地址。以上两条是在安装时选择配置的，以后可以在此处修改它们的属性。

此外，如果要拒绝一部分人访问该 WWW 站点，可以到 Apache 的安装目录下找到 Access 文件，输入要禁止的 IP 地址即可。

2.4.3 Tomcat 的安装和配置

首先从 Apache 网站 http://jakarta.apache.org/site/binindex.cgi 下载 Apache Tomcat 版本 jakarta-tomcat-5.0.18.exe(集成实现了 Servlet 2.4 和 JSP 2.0 标准)。然后进行安装和配置。

1．安装 Java 环境

在安装 Java 以前，需要介绍几个概念。大家经常看到 JDK、J2SDK 和 JRE，三者是什么关系呢？JDK(Java Develop Kit，JDK)是 Sun 公司早期的 Java 软件开发工具包，包含了所有编写、运行 Java 程序所需要的工具：Java 基本组件、库、Java 编译器、Java 解释器、小应用程序浏览器、以及一些用于开发 Java 应用程序的程序等。现在把 JDK 称为 Java(TM) 2 SDK。Java(TM) 2 SDK 又分成企业版(Enterprise Edition，即 J2EE)和标准版(Standard Edition，即 J2SE)两个版本。

J2SDK 包含了 JDK、JRE 和 Java Plug-in。J2SDK 供开发 Java 程序所用，应用程序用户是不需要开发工具的。而 JRE(Java Runtime Environment)顾名思义是 Java 程序运行所需要的环境。所谓跨平台就是各种平台都有一个中间代理，即 JRE。一般采用 Java 技术开发的软件都装有 JRE，所以 Sun 公司单独提供了 JRE 安装文件，以供 Java 应用程序发布时所用。

以上 Java 软件都可以从 Sun 公司的 Java 网站(http://java.sun.com)上获取，网站上提供了 J2EE SDK、J2SE SDK 和 Java VM(JRE)的各种版本的下载。

Java 2 SDK 的安装界面如图 2-42 所示。

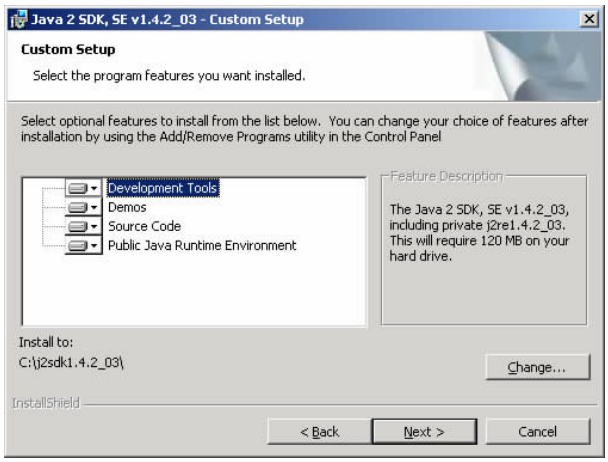


图 2-42 Java2 SDK 标准版的安装向导界面

按照向导提示将 Java 开发环境安装到计算机中，默认的文件夹为 C:\j2sdk1.4.2_03，如图 2-43 所示。

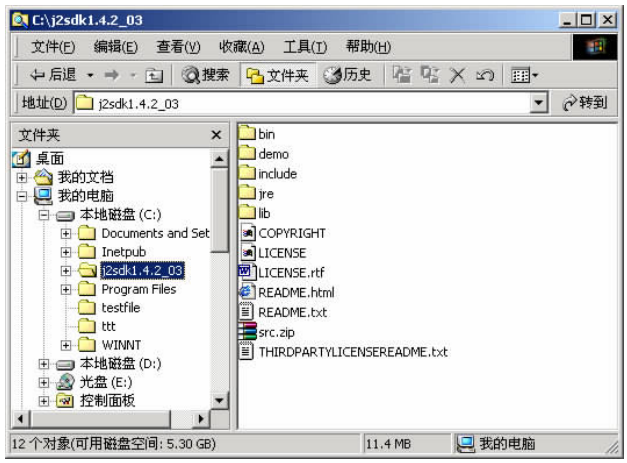


图 2-43　Java 2 SDK 标准版的安装环境

Java 安装完成后，需要进行相应的环境变量设置，包括 Java 主目录、环境变量、路径设置三个部分。一般进行如下设置。

在 DOS 提示符下，运行 sysedit ，在 autoexec.bat 中输入以下内容：

```
set JAVA_HOME =C:\j2sdk1.4.2_03
set CLASSPATH  =.;%JAVA_HOME%\lib(注意,.; 一定不能少,它代表当前路径)
PATH=%PATH%;%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin
```

其中，CLASSPATH 定义 javac 命令搜索类的路径。设置完成后，重新启动计算机，使得上述的设置生效。然后在 DOS 提示符下，可以检验上述设置，如图 2-44 所示。

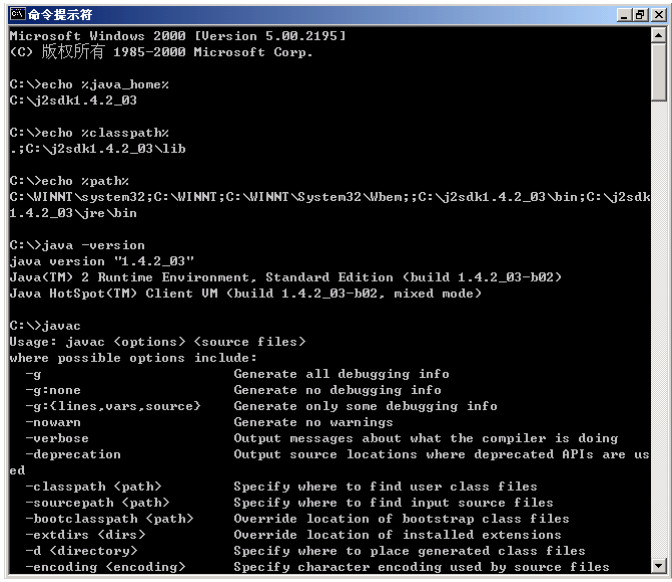


图 2-44　检验 Java 环境变量的设置情况

下面总结一下 Java 环境变量的设置问题。PATH 是操作系统执行可执行程序时用来查找该程序所在的路径。因为，Java 编译器(javac.exe)保存在%JAVA_HOME%\bin 中，Java 解释器(java.exe)保存在%JAVA_HOME%\jre\bin 中，要在任何路径下使用 javac.exe 和 java.exe，就必须将上述路径定义在操作系统的 Path 环境变量中。

CLASSPATH 环境变量记录了 Java 编译器和解释器所需要的类所在的路径。即使是用户自己创建的类，也应该添加到 CLASSPATH，这样做比较麻烦，所以在 CLASSPATH 中添加了一个当前目录(即.;)。这样，当转到用户所在的目录时，由于 javac 编译生成的用户类保存在当前路径，必须把当前路径加到 CLASSPATH 中，这样 Java 解释器才能够找到用户的类。有时候，会看到 CLASSPATH 中包含一个.jar 等压缩的 class 文件[1]，把它加入到 CLASSPATH 中，Java 环境可以读取该文件。

下面用一个简单的 Java 程序来测试 J2SDK 的安装。代码如下：

```
public class Test
{
  public static void main(String args[]) {
       System.out.println("Hello,My Java program ");
  }
}
```

创建文件夹 D:\MyJava，将上述程序代码保存在该文件夹下，文件名为 Test.java。接着打开 DOS 命令提示符窗口，转到 Test.java 所在目录 D:\MyJava，然后键入下面的命令 ：

```
javac Test.java
java Test
```

如果显示“ Hello, My Java program ”，则表明 Java 环境安装成功，用 Dir 命令可看到生成一个 Test.class 文件。否则需要仔细检查环境配置情况。

2．安装 Java VM(JRE)

如果需要运行 Tomcat，还需要在计算机中安装 Java VM(JRE)。因为 Tomcat5 需要 Java VM 的支持(Java VM 安装后的默认路径是 C:\Program Files\Java\j2re1.4.2_03)。Java VM 的安装界面如图 2-45 所示。

图 2-45 JRE 标准版的安装向导界面

按照向导提示可以完成 JRE 的安装，安装完成后，在开始菜单的“ 程序 ”组中，将增加 Java Web Start 程序组，包含Java Web Start 命令。

3．Tomcat 的安装和配置

现在，Tomcat 的最新版本是 5.0.18，它的

1 jar 的全称是 JavaTM Archive（JAR）file，是 Java 存档文件，主要压缩存储 Java 的 class 文件。Jar 是 JavaJDK 中的命令，可以在 DOS 提示符下，使用 jar - help 命令显示 jar 的使用方法。

运行需要 Java Virtual Machine(Java VM)的支持。首先在服务器上安装 Java VM(JRE)，然后执行 Tomcat 安装程序 jakarta-tomcat-5.0.18.exe，启动安装向导，按照向导提示执行下面步骤。

(1) 选择要安装的 Tomcat 组件，如图 2-46 所示。

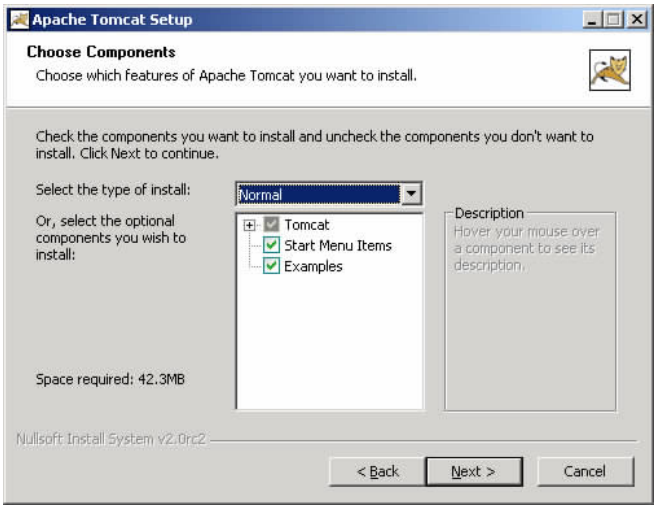


图 2-46　Tomcat 安装向导界面

(2) 选择安装的物理路径，默认路径为 C:\Program Files\Apache Software Foundation\Tomcat 5.0。

(3) 进行 Tomcat 的基本配置，包括 HTTP 端口(Tomcat 的默认值为 8080)、管理员的登录名(默认登录名为 admin)和密码,，密码可以为空。

(4) 选择安装 Java Virtual Machine 的物理路径，默认值为 C:\Program Files\Java\j2re1.4.2_03。

(5) 执行安装，向导将把有关的文件复制到相关的目录下，并自动启动 Tomcat。当 Tomcat 安装完成后，在“开始”|“程序”菜单中，将增加 Apache Tomcat 5.0 子菜单。

Tomcat 为 JSP 的容器，要在 Windows 下运行 JSP，需要安装 Java 开发环境，同时需要一些特殊的环境设置，包括 Tomcat 主目录、环境变量、路径设置。

在 DOS 提示符下，运行 sysedit ，在 autoexec.bat 增加以下内容：

```
set TOMCAT_HOME = C:\Program Files\Apache Software Foundation\Tomcat 5.0
set CATALINA_HOME= C:\Program Files\Apache Software Foundation\Tomcat 5.0
set CLASSPATH =.;%JAVA_HOME%\lib;%TOMCAT_HOME%\common\lib
PATH=%PATH%;%TOMCAT_HOME%;%TOMCAT_HOME%\bin
```

以上具体的路径应该根据安装的实际情况设置，设置完成后，重新启动计算机，使设置生效，然后再启动 Tomcat。

对于环境变量，用户还可以通过“控制面板”|“系统”来设置，具体步骤如下。

(1) 在“控制面板”窗口中，双击“系统”图标，打开“系统特性”对话框，选择“高级”选项卡，如图 2-47 所示。

(2) 在“系统特性”对话框的选择“高级”选项卡中，单击“环境变量”按钮，打开“环境变量”对话框，如图 2-48 所示。

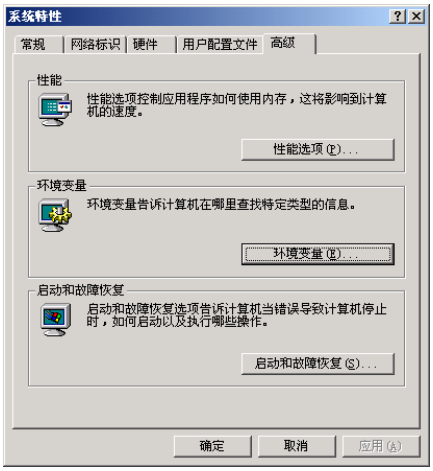


图 2-47 “系统特性”对话框

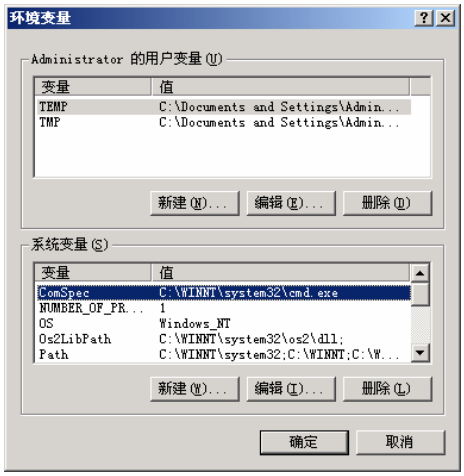


图 2-48 “环境变量”对话框

(3) 在环境变量窗口的系统变量区域，可以新建环境变量，或者对已经存在的环境变量进行修改。例如，要建立一个环境变量 JAVA_HOME，对应 set JAVA_HOME=C:\j2sdk1.4.2_03，具体操作如下。

单击“新建”按钮，打开新建系统变量对话框，输入要新建的系统变量，输入内容如图 2-49 所示。

需要特别注意的是，如果该步骤的环境变量配置不对，Tomcat 将无法启动。另外，如果 Web 应用中只是一般的 htm 文件，不配置环境变量，网站也可以浏览。

设置的环境变量，可以在 DOS 提示符下，通过 set <环境变量>命令显示环境变量的配置情况。例如，输入 set classpath 会显示 classpath 环境变量的配置，输入 set path 会显示 path 环境变量的设置情况等。

4．使用 Tomcat 服务器

当 Tomcat 安装并配置了环境变量后，重新启动计算机使环境变量生效，此时就可以使用 Tomcat 了。在 Windows 2000 的‘开始’菜单中，选择 Start Tomcat 菜单项即可启动 Tomcat，显示 Apache Tomcat 5.0 启动界面，如图 2-50 所示。

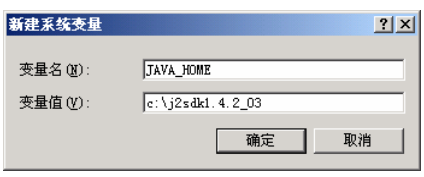


图 2-49 “新建系统变量”对话框

图 2- 50 Apache Tomcat 5.0 启动界面

正确启动后，在任务栏的右边显示 Tomcat 运行标识，如果环境变量或者 server.xml 文件配置不对(见下面的介绍)，可能无法启动 Tomcat。

此时，打开浏览器，在地址栏键入 http://localhost:8080/ 或者 http://127.0.0.1:8080 即可看到 Tomcat 的启动界面，如图 2-51 所示。

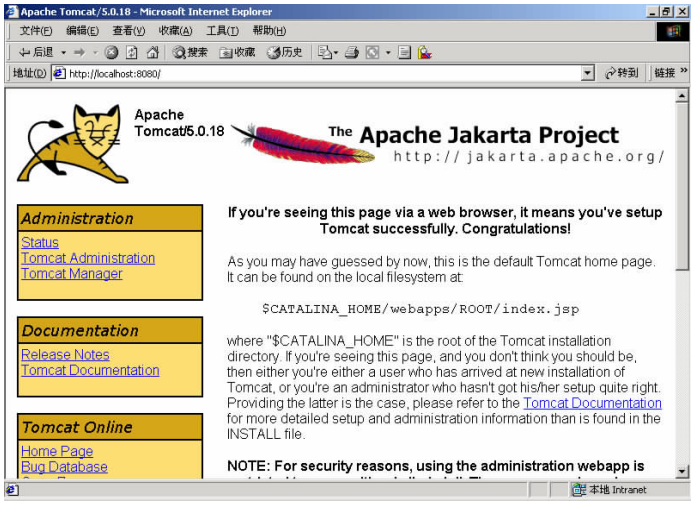


图 2-51　Tomcat 的启动界面

Tomcat 启动后，并不意味着所有的需要运行用户 Web 的设置都完成或正确，在本书 2.4.4 节还会有相应的补充。

2.4.4　建立并部署 Web 应用

Tomcat 安装完成后，建立的文件结构如图 2-52 所示。

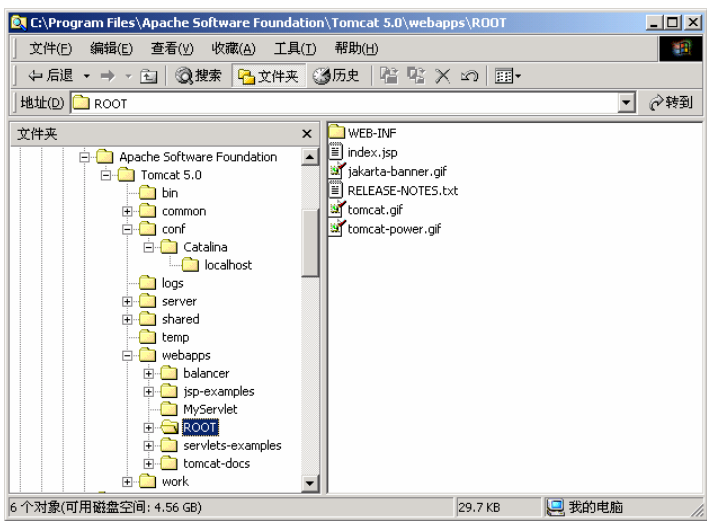


图 2-52　Tomcat 文件结构

在默认情况下，Tomcat 指向一个默认的 Web 应用(C:\Program Files\Apache Software Foundation\Tomcat 5.0\webapps\ROOT)，在 webapps 文件夹下，还包含几个其他 Web 应用，如 jsp-examples、servelets-examples 等。下面介绍在 Tomcat 下新建 Web 应用的不同方法。

1．在 webapps 下建立用户 Web 应用

在 webapps 下，可以创建用户的 Web 应用主目录，例如创建 MyServlet 等项目，在该文件夹下可以存储用户的.jsp 文档等。然后可以通过 http://127.0.0.1:8080/用户项目文件夹/文档名.jsp 可以显示相应的用户文件。

例如，在浏览器地址栏中输入 http://127.0.0.1:8080/MyServelet/1.jsp 即可运行一个用户的 JSP 文档。

2．Web 应用与 server.xml 配置文件

现在用 D:\MyJSP 作为主目录创建用户的第一个 Web 应用，只包含一个首页文件 index.jsp，代码如下：

```
<html>
<head>
   <title>My JSP</title>
</head>
<body>
   <% out.println("Hello,我的 JSP"); %>
</body>
</html>
```

现在，希望通过 Tomcat 来访问这个 Web 应用，那么如何来做呢？上面已经看到，通过 http://127.0.0.1:8080/可以访问 Tomcat 默认的 Web 应用(见图 2-51)。要想通过 http://127.0.0.1 来访问这个新的 Web 应用，需要修改 Tomcat 的一些设置。要使 Tomcat 指向 D:\MyJSP，需要作如下修改。

(1) Tomcat 默认的 Web 服务端口号为 8080(可以在安装过程中修改)，而在实际的应用中 HTTP 默认的端口号是 80，因此需要修改端口号，方法如下：

修改 C:\Program Files\Apache Software Foundation\Tomcat 5.0\conf 下的文件 server.xml。用记事本打开该文件，找到如下段落：

```
<!--Define a non-SSL Coyote HTTP/1.1 Connector on the port specified uring
nstallation  -->
   <Connector            port="8080"maxThreads="150"minSpareThreads="25"
   maxSpareThreads="75"
            enableLookups="false" redirectPort="8443" acceptCount="100"
            debug="0" connectionTimeout="20000"
            disableUploadTimeout="true" />
```

将 Connector ort="8080"改为 Connector ort="80"即可。然后重新启动 Tomcat，重新打开浏览器，输入 http://127.0.0.1/即可，而不需要指定端口 8080。

需要注意的是，如果已经启动了 Apache Server，首先，应该在 Windows 的“开始”|

“程序”菜单中找到 Apache HTTP Server 程序组，执行 Stop 命令，停止 Apache Server。

(2) 增加新 Web 应用的上下文

接下来，需要增加新的 Web 应用上下文。找到并修改<!-- Tomcat Root Context -->段，增加用户应用 D:\MyJSP 的上下文。

```
<!-- Tomcat Root Context -->
      <!--
        <Context path="" docBase="ROOT" debug="0">
      -->
```

在后面增加如下行：

```
<Context path="" docBase=" D:\MyJSP " debug="0">
```

注意：Tomcat 是大小写敏感的，文件夹名字的大小写必须和磁盘上的文件夹名的大小写一致，否则也不运行，如磁盘上为 D:\MyJSP，server.xml 中就不能写成 d:\myjsp。在主目录下可以创建首页文件 index.jsp 和其他的文件夹。

当上述修改完毕后，在任务栏中右击 Tomcat 图标，选择 Shutdown:Tomcat 5 命令，关闭 Tomcat。然后在“开始”菜单中重新启动 Tomcat，尝试运行用户 Web 应用。

在 IE 地址栏中输入 http://127.0.0.1/，显示错误信息页面，显示“Unable to compile class for JSP…No Java compiler was found to compile the generated source for the JSP.”等信息。并且提示将 JAVA_HOME 下的\lib\tools.jar 文件复制到 Tomcat 主目录下的\common\lib 文件夹下，按照提示复制文件。或者在 CLASSPATH 环境变量中添加 %JAVA_HOME%\lib\tools.jar，效果是一样的。

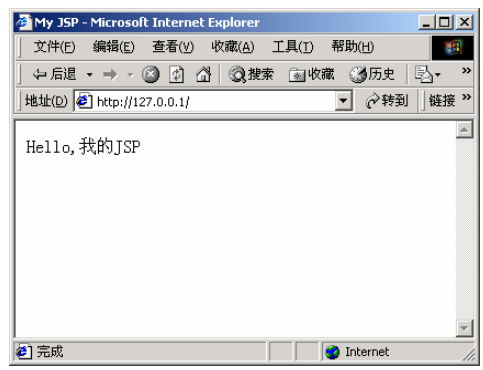


图 2-53　第一个 JSP Web 应用首页

然后重新启动 Tomcat。打开 IE 浏览器，输入 http://127.0.0.1/，显示如图 2-53 所示。

图 2-53 表明 Tomcat 已经运行了用户的 Web 应用 D:\MyJSP 目录下的首页文件 index.jsp。用户可以在主目录下创建其他的 JSP 文件，在 IE 的地址栏内输入 http://127.0.0.1/文件名(包含扩展名)即可执行相应的 JSP 文件了。

如果 JSP 文件中含有 Java 脚本程序，必须要保证 Tomcat 和 J2SDK 的环境变量设置正确，否则 Web 应用将不能运行。如果 Web 页是.htm 文件，运行与环境变量的配置无关。

2.4.5　在 Tomcat 中使用虚拟目录和虚拟主机

在 IIS 中，已经介绍了虚拟目录的概念，这里将介绍 Tomcat 中虚拟目录的使用，以及虚拟主机的概念和配置。

1．使用虚拟目录

在 Tomcat **的** \conf\ **下面的** server.xml **文件中，会看到下面的代码段：**

```
<!-- Tomcat Root Context -->
       <!--
         <Context path="" docBase="ROOT" debug="0">
       -->
```

这是定义虚拟目录用的，其中 path **的值是虚拟目录，**docBase **的值是对应的物理路径。下面是第二个** Web **应用的例子。**

(1**) 在**%TOMCAT%\webapps **目录下新建** Web **应用主目录** myapp

(2**) 在** myapp **下新建一个目录** WEB-INF**，在** WEB-INF **下新建一个文档** web.xml**，内容如下：**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app
   PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
   "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>
<display-name>My Web Application</display-name>
<description>
  A application for test.
</description>
</web-app>
```

(3**) 在用户** Web **应用主目录** myapp **下新建** Web **首页文件** index.jsp**，内容如下：**

```
<%@ page contentType="text/html;charset=gb2312"%>
<html>
<head>
  <title>myapp</title>
</head>
<body>
```

现在的时间是：

```
<%= new java.util.Date() %>
</body>
</html>
```

(4**) 添加虚拟目录**

在 Tomcat **的** \conf\ **下面的** server.xml **文件中，找到如下代码段：**

```
<!-- Tomcat Root Context -->
       <!--
         <Context path="" docBase="ROOT" debug="0">
       -->
```

其中第一个 Web **应用添加了如下一段代码。**

```
<Context path="" docBase="D:\MyJSP" debug="0"/>
```

上述代码没有指定虚拟目录，表明 D:\MyJSP 是连接到 Tomcat 服务器后直接显示的 Web 应用，默认的首页是 index.jsp。

在上述代码的下面增加如下代码：

```
<Context path="/myapp" docBase="myapp" debug="0"/>
```

/myapp 为 myapp 应用的虚拟目录。

修改结束后，保存 server.xml 文件。然后关闭 Tomcat，再重启 Tomcat。打开浏览器，在地址栏内输入 http://localhost/myapp，即可打开 myapp 下面的 index.jsp 页面，显示页面如图 2-54 所示。

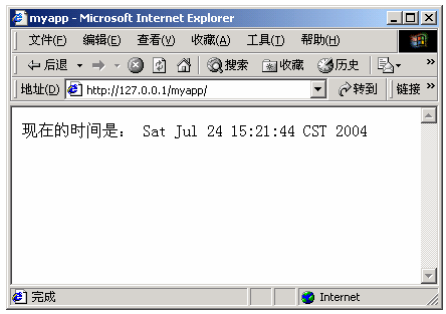


图 2-54　在 Tomcat 中使用虚拟目录

2．虚拟主机的配置

最后简单地介绍一下虚拟主机的概念，在 IIS 中，本书介绍了运行多个 Web 站点的概念。实际上，在一台服务器上运行多个 Web 站点，就是虚拟主机的概念。

在 Tomcat 中，所谓“虚拟主机”是使用特殊的软硬件技术，把一台计算机主机分成一台台“虚拟”的主机，每一台虚拟主机都具有独立的域名和 IP 地址(或共享 IP 地址)，有完整的 Internet 服务器(如 WWW、FTP、E-mail 等)功能。每一台虚拟主机看起来和一台独立的主机完全一样，但它们却是在同一台服务器主机上。

在 Tomcat 上配置虚拟主机，目前主要采用共享 IP 地址、不同域名的办法。可以按照如下步骤进行实验。

(1) 定义两个域名。

假设将要使用的域名是 www.abc.net 和 www.xyz.net。域名需要在 DNS 上做相应的域名解析。

为了测试方便，可以在客户机上进行，即在 Windows 2000 下的\\WINNT\system32\drivers\etc\文件夹下，用记事本打开 hosts 文件，其中记录了 127.0.0.1 的域名 localhost。在该条记录的下面增加下面内容：

```
192.168.0.1 www.abc.net
192.168.0.1 www.xyz.net
```

其中，192.168.0.1 是 Tomcat 服务器主机的 IP 地址，可以根据实际地址更改。

(2) 建立两个 Web 应用

将 Tomcat 目录下的 webapps 子目录在同一目录下复制一份，命名为 webapps 2，然后将 webapps 目录改名为 webapps1 。

写一个简单 HTML 文件用来测试，文件名为 test.html ，文件内容如下：

```
<html>
<head>
```

```
    <title>welcome</title>
</head>
<body>
    欢迎访问 www.abc.net
</body>
</html>
```

将 test.html 文件分别在 tomcat/webapps1/ROOT 和 tomcat/webapps2/ROOT 目录下各放置一个副本，然后将 tomcat/webapps2/ROOT/test.html 文件的输出改为“欢迎访问 www.xyz.net”。

(3) 修改 server.xml 文件

打开 Tomcat 主目录下的/conf/server.xml 文件，找到下面的代码段：

```
<!-- Define the default virtual host
         Note: XML Schema validation will not work with Xerces 2.2.
     -->
     <Host name="localhost" debug="0" appBase="webapps"
      unpackWARs="true" autoDeploy="true"
      xmlValidation="false" xmlNamespaceAware="false">
```

将 Host 元素的原属性全部删掉，在该元素内添加如下属性值：

```
name="www.abc.net" debug="0" appBase="webapps1" unpackWARs="true"
autoDeploy="true"
```

然后配置 www.xyz.net 虚拟主机，定义另一个 Host 元素，其属性如下：

```
name="www.xyz.net" debug="0" appBase="webapps2" unpackWARs="true"
autoDeploy="true"
```

现在可以启动 Tomcat，分别访问 http://www.abc.net:8080/test.html 和 http://www.xyz.net:8080/test.html，此时就可以访问到不同的页面了，表明虚拟主机已经配置成功。

2.4.6 Apache 和 Tomcat 的关系

通过以上的介绍，可以看出只用 Tomcat 也能够建立和运行一个 Web 站点，那么 Apache 和 Tomcat 是什么关系呢？是不是两者必须一起使用呢？

实际上 Apache 和 Tomcat 的作用是不一样的，Apache 主要应用在实现虚拟主机、支持 PHP、站点性能、安全等方面。如果不是要用 Apache 实现以上功能，从开发的角度没必要用 Apache 和 Tomcat 配合，单独应用 Tomcat 即可。即不需要安装 Apache 服务器，单独使用 Tomcat 即可运行 Web 应用。这是因为，Tomcat 有内置的一个 Apache 的 HTTP 服务，但是它仅仅对 JSP 程序体现出比较好的执行效率和性能，对于静态页面的处理速度远不如 Apache。

可见，为了提高 Web 系统的整体性能，需要将 Apache 和 Tomcat 进行整合配置，具体实现请读者参考其他书籍或从网上查找。

2.5　IIS 和 Tomcat 的整合

通过上面的介绍，我们知道 IIS 运行在 Windows 2000 下，主要的开发工具是 ASP。Tomcat 主要是基于 Java 和 JSP 开发的。而通过 IIS 建立的 Web 站点，可以执行 JSP 文件，这就需要 IIS 和 Tomcat 的整合。IIS 和 Tomcat 的整合主要是通过 IIS 中 Web 站点属性对话框的“ISAPI 筛选器”选项卡进行相应的设置。

IIS 和 Tomcat 的系统整合比较麻烦，网上有大量的文章介绍，比较完整的介绍请参考网页 http://www.hclab.com/hclabdata/list.asp?id=177 上的文章“IIS 和 Tomcat 整合解决方法”和网页 http://www.reynir.net/tomcat/tomcat_IIS_service_jk2.html 中的文章“Tomcat and IIS Installation process (jk2)”，并附有所需要的文件代码。具体的整合步骤介绍在此省略。

另外，对于在 IIS、Apache 和 Tomcat 建立的不同的 Web 应用，它们能够同时运行。如果在一台计算机上建立了多个不同的 Web 应用，又分别采用的是 IIS 和 Tomcat，此时只需要给每个不同的 Web 应用不同的端口号即可。

同时，由于 Tomcat 建立的 Web 应用对文件的大小写是敏感的，我们还可以将两种应用融合起来。例如，在基于 Tomacat 的 Web 应用中通过超链接接到 IIS 中的 Web 应用。方法是：在 Tomcat 的 Web 中增加一个指向 IIS 中 Web 应用的链接 http://IP 地址：端口号/，这里用的是 IIS 中建立的 Web 应用的首页。这样多个 Web 应用就可以在一台计算机上同时运行了。

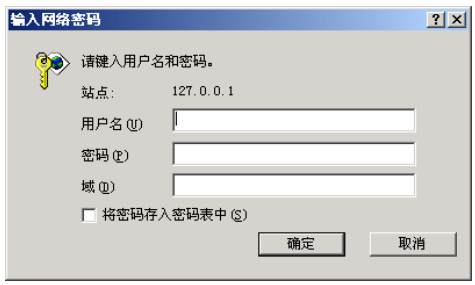
习　题　2

1．什么是 IIS？IIS 5.0 包括哪些可选组件？简述它们的功能。

2．在 Windows 2000 Professional 和 Windows 2000 Server 中都可以安装 IIS 吗？简述　什么不同。

3．简述在 IIS 中 Web 站点的创建过程。

4．在连接新建的 Web 站点时出现下面的“输入网络密码”对话框，为什么？如何解决？



5．什么是虚拟目录？使用虚拟目录有何好处？

6．如何在一台 Windows 2000 Server 计算机上运行多个 Web 站点？

7．当连接到一个 IIS 中的 Web 站点，浏览一个 ASP 页时，显示“网页无法显示”提示页面，并且在页面中提示：“您试图从目录中执行 CGI、ISAPI 或其他可执行程序，但该目录不允许执行程序”，为什么？如何解决？

8．Apache 是一个什么样的软件？简述其主要功能。

9．Tomcat 是什么软件？简述它的功能。自己安装一次 Tomcat，写出在安装情况下创建的 Tomcat 目录，简单说明每个目录的功能。

10．Tomcat 必须和 Apache 集成才能使用吗？为什么？

11．在 Java 开发环境中，简述 JDK、J2SDK 和 JRE 三者之间的关系。

12．如何编译和运行 Java 程序？

13．运行 Tomcat 需要设置哪些环境变量？如何进行设置？

14．如何显示系统配置的 classpath 环境变量？

15．根据本书的介绍或者自己从网上搜索，写出对 IIS 和 Tomca 进行整合的过程。

第 3 章　HTML 和 XML 基础

标记语言是 Web 应用的基础，标记语言是由内容和标签组成的，标签指定了内容在浏览器中的显示形式。在 Web 应用中，所有的 Web 页面都是以标记语言书写的具有特定格式的文档。因此，无论是 Web 应用还是开发，都应该对标记语言有一个基本的认识。

本章将介绍两种主流的标记语言 HTML 和 XML 的基本语法，并通过大量的实例来说明它们的应用。

3.1　万维网联盟(W3C)和 SGML

万维网联盟(World Wide Web Consortium，W3C)是一个国际标准化组织，成立于 1994 年，其主要目标是为万维网的发展开发通用的协议和标准。HTTP 和 HTML 协议就是 W3C 针对 WWW 制定的两个非常重要的协议。

标准通用标记语言 SGML(Standard Generalize Markup Language，SGML)是一个用来定义在电子表格中如何对文件的结构和内容进行描述的国际标准(ISO-8879)。时间可以追溯到 1969 年，当时美国 IBM 公司的研究人员开始设计一种名为 GML(Generalized Markup Language)的语言，在印刷、统计等需要大规模数据处理的行业和部门的支持下，这项研究工作持续了十几年，终于在 1980 年推出了 SGML 语言，并于 1986 年获得国际标准化组织(ISO)的批准。其后，SGML 的发展较为平稳，但不为其领域之外的人们所广泛了解。直至 1991 年，当 HTML(HyperText Markup Language，超文本标记语言)问世之后，人们才开始认识 SGML。

为了满足各种不同的页面表达需要，SGML 设计得非常复杂，SGML 的正式规范多达 500 多页。因此使用起来很不方便，使它未能得到普及和大规模的应用。虽然 SGML 没有 HTML 应用广泛，但是 SGML 定义了标记语言的基本概念，奠定了标记语言的技术基础。

现在，在万维网中普遍应用的 HTML 和 XML 都是在 SGML 基础上开发的，可以说它们都是 SGML 的一个子集。作为互联网信息共享的技术规范，标记语言对互联网的发展起到了巨大的推动作用。如果需要了解 W3C 及有关的标准，可以访问 W3C 的网站 http://www.w3c.org。

3.2　超文本标记语言 HTML

HTML(HyperText Markup Language，超文本标记语言)是一种用来制作超文本文档的简单标记语言包含了文档数据和显示格式两部分，其中文档数据是显示在 WWW 浏览器中的数据内容，显示格式则规定了这些内容在浏览器中以何种格式、样子呈现给用户。

HTML 语言通过利用各种标记(tag)来标识文档的结构以及标识超链接(Hyperlink)的信息。虽然 HTML 语言描述了文档的结构格式，但并不能精确地定义文档信息必须如何显示和排列，而只是建议 Web 浏览器(如 Mosiac、Netscape、IE 等)应该如何显示和排列这些信息，最终在用户面前的显示结果取决于 Web 浏览器本身的显示风格及其对标记的解释能力。这就是为什么同一文档在不同的浏览器中展示的效果会不一样的原因。

HTML 文件是一种纯文本文件，通常它带有.htm 或.html 的文件扩展名(在 UNIX 和 Windows 95 中的扩展名为.html)。可以使用任何文本编辑器，如 Windows 中的“记事本”、“写字板”等来编辑 HTML 文档，然后将文件保存为.htm 格式，或者存为文本格式。

利用“记事本”等文字软件编写 HTML 文件非常麻烦、效率很低，于是出现了很多专门的优秀的 HTML 编辑器。常用的 HTML 编辑软件有 FrontPage、Dreameweaver 等，它们均为所见即所得的编辑器，文档开发人员不必一行一行的书写 HTML 语句。借助 HTML 专用工具，在屏幕上对页面进行交互式设计，将自动生成对应的 HTML 文件，并且都具有预览功能，大大提高了设计 Web 页的效率。

3.2.1 HTML 标记语法和文档结构

HTML 文档是纯 ASCII 码的文本文件，由“显示内容” 和“控制语句”两部分组成。控制语句描述了显示内容以何种形式在浏览器中显示，并负责客户端与服务器端之间的信息交换。控制语句以标记(tag)形式出现，以区分于显示内容。

标记封装在小于号(<)和大于号(>)构成的一对尖括号之中，标记一般分首标记和尾标记，它们成对出现。首标记用于开启某种形式的显示，尾标记含“/”以同首标记区分，用于关闭首标记开启的功能。例如：<u>text with underline</u>，首标记<u>开启下划线功能，尾标记</u>关闭下划线功能。该语句在浏览器中把文本串“text with underline”加上下划线显示。有的控制语句仅需一个首标记，没有尾标记。如：
表示换行。

1．标记类型与标记属性

标记分为“单标记”和“双标记”两种类型。“单标记”是指只需单独使用就能完整地表达意思的一类标记，这类标记的语法是：

<标记>

常用的单标记有
表示换行，<hr>代表一条水平线等。

另一类标记称为“双标记”，它由“首标记”和“尾标记”两部分构成，必须成对使用，其中首标记告诉 Web 浏览器从此处开始执行该标记所表示的功能，而尾标记告诉 Web 浏览器在这里结束该功能。首标记前加一个斜杠(/)即成为尾标记。这类标记的语法是：

<标记>文档内容</标记>

其中“文档内容”部分就是要被这对标记施加作用的部分。例如想突出对某段文字的显示，就将此段文字放在一对...标记中：

```
<em>text to emphasize</em>
```

许多单标记和双标记的首标记内可以包含一些属性，其语法是：

```
<标记 属性 1="属性值" 属性 2="属性值" 属性 3="属性值"...>
```

各属性之间无先后次序，属性之间用空格分开。属性也可省略(即取默认值)，属性值两侧的双引号(")可以省略不写。例如：单标记<hr>表示在文档当前位置画一条水平线(horizonta line)，一般是从窗口中当前行的最左端一直画到最右端。另外，<hr>标记还有 size、align、width 等属性，例如可写作<hr size="3" align="center"　width="50%">。其中 size 属性定义线的粗细，默认为 1；align 属性表示对齐方式，可取 left(左对齐，默认值)、center(居中)和 right(右对齐)；width 属性定义线的长度，可取相对值(整个窗口的百分比)，也可取绝对值(屏幕像素点的个数，如 width="400")，默认值是"100%"。

2．文档结构

一个 HTML 文档以<html>标记开始，以</html>标记结束，表示这对标记间的内容是 HTML 文档。HTML 文档的中间又分成文件头和文件体两个部分，由相应的标记来区分。HTML 文档总体结构如下：

```
<html>
<head>
    头部信息
</head>
<body>
    文档主体
    (语句部分)
</body>
</html>
```

其中，<head>...</head>之间是文件头，如文档总标题等，若不需头部信息则可省略此标记。<body>...</body>之间是文件体，表示正文内容的开始，<body>标记一般不能省略。

在一个 HTML 文档中，<html>标记用于 HTML 文档的开始，用来标识 HTML 文档的开始；</html>标记放在 HTML 文档的最后，用来标识一个 HTML 文档的结束。

因为.htm 或.html 文件被 Web 浏览器默认为是 HTML 文档，有些 HTML 文档可以省略<html>和</html>标记。

3.2.2　文件头及相关标记

在 HTML 文档中，<head>…</head>标记对之间的部分称为文件头。文件头的内容主要用于定义 HTML 文档在 WWW 网中的情况，其内容不在浏览器中显示。用户可以省略<head>的首尾标记，而直接使用有关的标记语句。

文件头中主要的标记对有<title></title>、<script></script>等，下面分别介绍。

1．<title></title>标记

标识网页主题，其中的内容将在浏览器的标题栏中显示，不出现在页面内。

例如：

```
<title>欢迎光临 GSL 网站</title>
```

2．<meta>标记

它是 HTML **文档文件头**<head>…</head>**标记内的一个辅助性标记，往往不能引起用户的注意，但是它对于网页是否能够被搜索引擎检索以及是否能提高网页在搜索列表的排序起着关键的作用，是一个非常有价值的标记。**

<meta>**标记为单标记，没有尾标记。**<meta>**标记共有两个属性，分别是** http-equiv **属性和** name **属性，不同的属性又有不同的参数值，这些不同的参数值实现了不同的网页功能。**

(1**)** name **属性**

name **属性主要用于描述网页，与之对应的属性值为** content**，**content **中的内容主要是便于搜索引擎查找信息和分类信息用的。**meta **标记的** name **属性语法格式是：**

```
<meta name="参数" content="具体的参数值">
```

name **属性主要有以下几种参数值。**

- keywords**(关键字)** keywords **用来告诉搜索引擎该网页的关键字是什么。**

例如：

```
<meta name="keywords" content="science,education,culture,,entertainment ">
```

- description**(网站内容描述)** description **用来告诉搜索引擎网站的主要内容。**

例如：

```
<meta name="description" content="This page is about science, education etc.">
```

- robots**(机器人向导)** robots **用来告诉搜索引擎机器人需要索引的页面有哪些。**
  content **的参数有** all、none、index、noindex、follow、nofollow**。默认是** all**。**

例如：

```
<meta name="robots" content="none">
```

- author**(作者)，标注网页的作者。**

例如：

```
<meta name="author" content="brion@mail.abc.com">
```

(2**)** http-equiv **属性**

http-equiv**，相当于** HTTP **的文件头作用，它可以向浏览器传回一些有用的信息，以帮助正确显示网页内容，与之对应的属性值为** content**，**content **中的内容就是各个参数的变量值。**meta **标记的** http-equiv **属性语法格式如下：**

```
<meta http-equiv="参数" content="参数变量值">
```

http-equiv **属性主要有以下几种参数。**

- content-type**(显示字符集的设置)，设置页面使用的字符集。**

例如：

```
<meta http-equiv="content-type" content="text/html; charset=gb2312">
```

- expires(期限)，用于设置网页的到期时间。一旦网页过期，必须到服务器上重新下载。

例如：

```
<meta http-equiv="expires" content="Fri, 12 Jan 2001 18:18:18 GMT">
```

- pragma(cache 模式)，禁止浏览器从本地计算机的缓存中访问页面内容。

例如：

```
<meta http-equiv="pragma" content="no-cache">
```

该种设置使访问者无法使用脱机浏览功能。

- refresh(刷新)，自动刷新并指向新页面。

例如：

```
<meta http-equiv="refresh" content="60; url=new.htm">
```

则浏览器将在 60 秒后自动转到 new.htm。可以利用这个功能制作一个封面，在若干时间后，自动带读者来到该目录页。

如果 URL 项没有，浏览器则刷新本页，实现了 WWW 聊天室定期刷新的特性。

- set-cookie(cookie 设置)，如果网页过期，那么存盘的 cookie 将被删除。

例如：

```
<meta  http-equiv="set-cookie"  content="cookievalue=xxx;expires=Friday,
12-Jan-2004 20:30:00 GMT;path=/">
```

- window-target(显示窗口的设置)，强制页面在当前窗口以独立页面显示。

例如<meta http-equiv="window-target" content="_top">可以用来防止别人在框架里调用自己的页面。

3．<base>标记

<base>标记定义了文档的基础 URL 地址，在文档中所有相对地址形式的 URL 都是相对于这里定义的 URL 而言的。一篇文档中的<base>标记不能多于一个，必须放于头部，并且应该在任何包含 URL 地址的语句之前。

<base>标记包含如下属性：

(1) href 属性

href 属性指定了文档的基础 URL 地址，该属性在<base>标记中是必须存在的。

例如，如果希望将文档的基础 URL 地址定义为 http://www.abc.com，则可以使用如下语句：

```
<base href = "http://www.abc.com">
```

当定义了基础 URL 地址之后，文档中所有引用的 URL 地址都从该基础 URL 地址开始，

例如，对于上面的语句，如果文档中一个超级链接指向 gsl/welcome.htm，则它实际上指向的是 URL 地址 http://www.abc.com/gsl/welcome.htm。

(2) target

target 属性同框架一起使用，它定义了当文档中的链接被单击后，在哪一个框架中展开页面。如果文档中超级链接没有明确指定展开页面的目标框架集，则使用这里定义的地址代替。常用的 target 的属性值有：

- _blank 表明在新窗口中打开链接指向的页面。
- _self 在当前文档的框架中打开页面。
- _parent 在当前文档的父窗口中打开页面。
- _top 在链接所在的完整窗口中展开页面。
- 例如 <base target="_blank"> 表明页面上所有的链接都在新窗口打开。

4．<link>标记

<link>标记定义了文档之间的包含。在 HTML 的头部可以包含任意数量的<link>标记。有些浏览器并不能很好处理 link 属性，因此不建议使用它。

<link>标记带有很多属性，下面是一些常用的属性：

(1) href href 属性值指向链接资源所在的 URL，其中 URL 是链接的地址。

(2) title title 属性值为一个字符串，用于描述该链接关系。

(3) rel rel 属性定义了文档和所链接资源的链接关系，一般形式是 rel= linktype。linktype 表明链接类型，可能的值包括 alternate，stylesheet，start，next，prev，contents，index，glossary，copyright，chapter，section，subsection，appendix，help，和 bookmark 等。如果希望指定不只一个链接关系，可以在这些值之间用空格隔开。

(4) rev rev 属性定义了文档和所链接资源之间的反向关系。其中 linktype 表明链接类型，其可能的取值同 rel 属性相同。

5．<isindex>标记

可让某些 Web Server 搜索网页内的关键字，若 Web Server 提供这样的搜索功能，客户端浏览器也支持搜索功能，在载入网页时会看到一个简单的搜索。

<isindex>的一般形式是：

```
< isindex  prompt=str>
```

例如，定义< isindex prompt="输入搜索内容">，则在 Web 页面的开始出现“输入搜索内容”，后面有一个文本框，用于输入搜索的内容。

<isindex>标记不仅能用于 LTML 文档头部，还可以用于表单(form)内。

6．注释标记

注释标记一般的形式是<!--注释性文字-->，用于在 HTML 文档中书写说明性文字，内容在浏览器中不显示。

3.2.3　文件体及相关标记属性

在<body>…</body>标记对之间的部分称为 HTML 文档的文件体。文件体中描述的是浏览器中显示的内容。文件体由一系列的控制语句构成，在<body>…</body>标记对之间可包含 … 、
、<hr>等标记，它们所定义的文本、图像等将会在浏览器中显示。

<body>标记是一个非常重要的标记，含有大量的属性，许多重要的网页功能都是通过<body>标记的属性实现的，可以把这些属性分成以下几类。

1．一般属性

<body>标记的一般属性用于页面的一般性设置，见表 3-1。

表 3-1　<body>标记的一般属性

属性	用途	实例
bgcolor="#rrggbb"	设置背景颜色	<body bgcolor="red">红色背景
text="#rrggbb"	设置文本颜色	<body text="#ff0000">红色文本
link="#rrggbb"	设置未阅读过的超文本连接颜色，默认值是蓝色	<body link="blue">链接为蓝色
vlink="#rrggbb"	设置阅读过的超文本连接颜色，默认值是紫色	<body vlink="#ff0000">红色
alink="#rrggbb"	设置动作中的超文本连接颜色，默认值是紫色	<body alink="yellow">设为黄色
leftmargin、topmargin	设置页边距	<body leftmargin="0" topmargin="0">
background	设置页面背景图片的 URL	background="image/10.jpg"
bgproperties	设置成 fixed，则背景图案不滚动。(只适应于 IE 浏览器)	bgproperties=fixed

引号内的 rrggbb 是用 6 个十六进制数表示的 RGB(红、绿、蓝三色的组合)颜色，如 #ff0000 对应红色。此外，还可以使用 HTML 语言所给定的颜色常量名。

例如，<body text="Blue">表示<body></body>标记对中的文本使用蓝色显示在浏览器中。另外，各个属性可以结合使用，如可以写成<body bgcolor="red" text="#0000ff">。

2．事件属性

当一个 Web 文档被加载显示或者退出(关闭)的时候，会发生相应的事件，这些事件在<body>标记中通过属性来表达。常见的事件属性见表 3-2：

表 3-2　<body>标记中的事件属性

事件	触发条件
onLoad	这个页面下载完成的时候
onUnload	退出这个页面的时候
onFocus	包含这个页面的窗口(window)或帧(frame)获得焦点的时候
onBlur	包含这个页面的窗口(window)或帧(frame)失去焦点的时候

除了上述传统的事件外，随着浏览器版本的升级，现在还增加了许多新的事件，以增加人机交互能力，这些事件是：

- onMouseMove　鼠标移动。
- onDblClick　鼠标双击。
- onMouseDown　鼠标被按下。
- onMouseUp　鼠标被释放。
- onKeyDown　键被按下，按键的 ASCII 码值保存在 window.event.keyCode 中。
- onKeyPress　键被按下然后被释放。
- onKeyUp　键被释放。
- onResize　窗口被调整大小。

在上述事件中，有些事件是<body>标记特有的，有些事件可能存在于多个不同的标记中。

事件属性的值往往是一个 JavaScript 函数，来完成 Web 编程任务。具体应用参考后面的章节。

[例 3-1]　一个简单的 HTML 文档示例。

下面是一个 HTML 文档的简单例子。用 Windows 中的记事本程序编写一段简单的 HTML 语句，如图 3-1 所示。

图 3-1　用“记事本”程序编辑 HTML 文档

双击保存后的 HTML 文档，文档将在浏览器中打开，显示结果如图 3-2 所示。

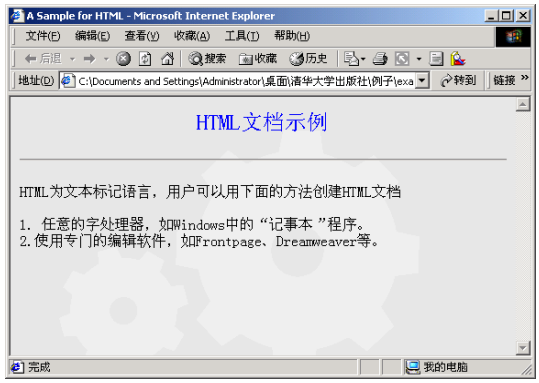


图 3-2　在 IE 浏览器中打开的 HTML 文档

3.2.4　文档内容标记

HTML 文档是由一系列的标记语句组成的，下面介绍每种文档标记的格式和功能。

1．标题、段落标记

在<body>…</body>标记对之间的内容将在浏览器中显示，对显示内容的格式指定是通过各种格式标记来定义的。常用的格式标记包括如下种类。

(1) 标题标记<h1></h1>...<h6></h6>　一级到六级标题标记，对应的字体逐渐缩小。

(2) 段落标记 …　标记一个段落，输出位置转到下一行开始，并增加一个空行。

(3) 回车换行标记
　将输出位置转到下一行的开始。

(4) 水平线标记<hr>　插入一条水平线。

(5) 预排版文本标记<pre>…</pre>　预排格式文本(Preformatted Text)标记可以产生固定宽度的字体。

该标记同时使空格、新行、制表键 tabs 有效（多个空格显示为多个空格，源文件中的换行也在浏览器中产生换行)，这对于显示程序清单和其他一些情况是很有用的。

<pre>标记可以带一个宽度属性 width，指明一行中最多允许的字符数。width 同时通知浏览器选择一个合适的字体以及文本的缩排。

在<pre>作用的部分中也可以加超链接。但是其他的 HTML 标记应该避免在 <pre> 的区间中使用。

注意，由于<, >, 和 & 在 HTML 文件中有特殊含义，在输入这些字符的时候必须使用它们的转义序列（分别为< , > , 和 &） 详细内容参见转义序列。

(6) 块引用标记<blockquote>…</blockquote>，主要用于在文档中标记一段引用文字(如经典论述、名人的演讲等)，它所包容的文字相对于标记外的文字往左缩进两个字符的距离。

2．文本格式标记

用来定义文本输出的字体和格式，如斜体、黑体字、下划线等。常用的标记包括如下。

(1) 字体标记…

属性有：face 属性，指定字体，例如 face=“宋体”；size 属性给出字的大小，HTML 默认的字体 size 是 3，如 size=“5”，size=“+2”表示在默认字体大小基础上加 2；color 属性给出字体的颜色。

(2) …、…、<u>…</u>、<strike>…</strike>标记

…标记为粗体显示。…标记为斜体显示标记。<u>…</u>为下划线标记；<strike>…</strike>为删除线标记。

上述标记可以联合使用，例如：

```
<b><font face="宋体" size="3" color="#0000FF"><i><u>字体标记一</u> </i>路
</font></b>
```

显示效果为：

字体标记一

(3) <big>…</big>、< small>…</small>、_…、[…]标记

< big>…</big>和< small>…</small>标记为指定字体大小的标记，< big>…</big>标记使字体放大，< small>…</small>标记使字体缩小。

_{...}和^{...}分别为上下标显示标记。

例如：

```
<font face="宋体">字体标记二：<big>大字体</big><small>小字体</small></font>
```

显示效果为：

字体标记二：**大字体**小字体

(4) 、、<tt></tt>、<cite></cite>标记

…标记和…标记类似，都是字体加粗标记。…标记和…标记类似，是斜体强调标记。<tt>…</tt>为打字字体 Courier 字体，字母等宽标记。<cite>…</cite>为传记引述斜体效果标记。

3．图像标记<img>

<img>标记用以插入图像及设置图像属性，丰富多彩的图像可以提高网页的吸引力。<img>标记的属性多达 21 个，下面是几个主要的属性。

- align　指定图像的位置是靠左、靠右、居中、靠上或者是靠底。默认情况是靠上，即 align=top。在图文混排时，这个参数很有用。
- id 属性　指定 id 号。
- class 属性　指定图像所属的类型。
- name　用于设置图像的名称。
- src　规定插入图像的 url 地址，也就是含路径的图像文件名。
- title　设置图像的标题。
- alt　表示图像的替代文字，主要用于在浏览器还没有装入图像(或关闭图像显示)时，先显示有关此图像的信息。
- border　设置图片的边框。
- height 和 width　分别用于指定图像的高度和宽度，可以与图片原来的宽度和高度不同。
- hspace 和 vspace　分别用于设置图像的左右边框大小和上下边框大小，在图文混排时会用到。
- ismp 和 usemap　在应用图像地图(map)时使用。ismap 表示图像地图的数据存放在服务器中，当鼠标在图像上的某个区域上时，可以将此区域的坐标传送给服务器处理。usemap 则用于设置图像地图的名称。

- dynsrc　指定要下载的影像片断的 URL 地址。
- controls　设置影像播放的控制按钮。
- start　设置何时开始播放影像片断，有三种选择。start=fileopen 表示页面载入后即开始播放，为默认状态；start=mouseover 表示当鼠标移到影像上即开始播放；start=fileopen,mouseover 表示当上面两种情况之一发生时就开始播放
- loop 指定影像片断的播放次数，当 loop=?1 时，影像片断将循环播放直至页面更新。

除了上述的一般属性外，<img>标记还有以下事件属性。

- onload　图像被载入时触发。
- onabort　图像取消载入时触发。
- onerror　图像载入出错时触发。

<ing>并不是真正地把图像加入到 HTML 文档中，而是将标记的 src 属性赋值为图形文件所在的路径及文件名(图像的格式可以为 GIF 或 JPG)。

[例 3-2]　图像标记的灵活运用示例。

下面代码显示一幅泰山图片，当鼠标移到图片的时候，显示一个简单的介绍，鼠标移开后重新显示图片。代码如下：

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>Taishan Introduction</title>
<style type="text/css">
<!--
.style1 { border-color: #FFFF00 }
-->
</style>
</head>
<body>
<img src="image/t01.jpg" width="200" height="150" border="1" align="left"
name="tai"
    alt="泰山图片" vspace="10" hspace="10" class="style1"
    onmouseover="tai.src='image/t02.gif'"onmouseout="tai.src='image/t01.
jpg'" >
</body>
</html>
```

使用<img>标记还可以启动媒体播放机播放影视片断，代码如下：

```
<img dynsrc="move/taishanfg.avi" start="mouseover" width="300" height=
"200" controls loop="1" alt="泰山风光介绍影像片断">
```

上面这段代码的效果是，当鼠标移到影像片断上时即开始启动媒体播放器播放影像一次，并且还设置了显示播放器控制面板，以便对播放过程进行控制。

在网页中加入影像片断与加普通图像基本相同，如它们都要指定来源、设置大小和替代文字等。但需要说明的是，加入影像片断时要使用“dynsrc”属性，且不能再在同一个<img>标记中使用“src”属性，否则将不能播放影像片断，只能显示替代文字。

4．超链接标记<a>…

超链接标记<a>…可以定义超文本链接、图像链接等。

(1) 超文本链接

超文本链接的一般格式为：

```
<a href="url">字符串</a>
```

href 属性的值为 URL 地址，字符串为显示的文字，一般显示为蓝色。当鼠标指向这个字符串时，鼠标变成手形，表示当前位置是一个超链接。

例如：

```
<a href="http://www.w3c.org">W3C 组织</a>
```

<a>标记有许多属性，主要的属性介绍如下。

- target 属性　定义超链接打开的目标窗口。一般形式是 target="window-name"，还可以取下面一些常量，如_self(相同框架)，_blank(新建窗口)，_top(整页)，_parent(父窗口)。
- title 属性　属性值为一字符串，鼠标指向超链接时，鼠标右下角显示标题文本。
- name 属性　定义一个书签。

对于一个较长的 HTML 文件，能不能在同一文件的不同部分之间建立链接，使用户方便地在上下方之间跳转呢？对于一个完整的文件，可以用它的 URL 来惟一地标识，但对于同一文件的不同部分怎样来标识呢？这就涉及到书签的概念，它是通过链接标记来定义的。

标识一个书签的方法为：

```
<a name="name">书签文本</a>
```

name 属性将放置该标记的地方标记为“ name ”，name 是一个全文惟一的标记串，这样，我们就把放置标记的地方做了一个叫做“ name ”的标记。

有了书签后，href 属性除了指向一个网址外，还可以定位到网页内一个具体的书签，用法是 href="url#name"，如果是同一个网页内，可以写成 href="#name"，其中 name 是书签名，同一网页内不能重名。

(2) 图像链接

<a>…标记可以用于图像链接，一般用法是：

```
<a href="url"><img src="imgeurl"></a>
```

[例 3-3]　超链接标记使用示例。

下述代码演示了超文本链接、在 HTML 文档中插入书签、图像链接的应用。

```
<html>
<head>
<meta http-equiv="Content-Language" content="zh-cn">
<title>Shandong Travel</title>
</head>
<body>
```

```
<p align="center">美丽的山东</p>
<p><a href="#济南"><img border="0" src="image/1.jpg" width="30" height=
"30"></a></p>
<p><a name="济南">美丽的泉城</a></p>
... ...
<p><a name="青岛">美丽的海滨城市??青岛</a></p>
... ...
<a href="http://www.cnta.com/" title="中国旅游网" target="_blank">主要旅游网
</a>
</body>
</html>
```

5．影像地图标记<map> </map>、<area></area>

一个图像映射(map)由一些区域(area)组成，每个区域对应于一个相关联的超级链接，当单击其中某个区域时就转到相应的链接。

(1) <map>…</map>标记

有一个开始标记和结束标记，中间是一系列的 area 标记，具有一个 name 属性，指出图像地图的名称。

(2) <area>…</area>标记

<area>标记主要用于图像地图(map)，通过该标记可以在图像地图中设置作用区域(又称“热点”)，当鼠标移到某作用区域并单击时，执行对应的事件处理函数。

<area>标记属性介绍如下。

- class 属性　指定热点的类型。
- id 属性　指定热点的 id 号。
- href 属性　用于设置该热点所链接的 URL 地址。
- alt 属性　用于设置热点的替代性文字。
- shape 属性和 coords 属性　shape 和 coords 是两个主要的参数，用于设置热点的形状和大小。其基本用法如下：

shape="rect" coords="x1, y1, x2, y2"，表示设置热点的形状为矩形，左上角顶点坐标为(x1, y1)，右下角顶点坐标为(X2, y2)。

shape="circle" coords="x1, y1,r"，表示设置热点的形状为圆形，圆心坐标为(x1, y1)，半径为 r。

shape="poligon" coords="x1, y1, x2, y2 ..."表示设置热点的形状为多边形，各顶点坐标依次为(x1, y1)、(x2, y2)、(x3, y3)……

<area>标记是在图像地图中划分作用区域的，其划分的作用区域必须在图像地图的区域内，所以在用<area>标记划分区域前必须用<map>标记来设置图像地图的作用区域，并为指定的图像地图设置名称。

[例 3-4]　使用图像地图示例。

在一幅中国地图上，当用鼠标单击某个省份时，则打开一个窗口，显示该省(直辖市)的简单介绍。

根据上面的要求，可以通过下面的步骤完成：

(1) 插入中国地图图片(chinamap.jpg)，并设置好图像的有关参数，且在<img>标记中设置参数 usemap="chinamap"　ismap，以表示对图像地图(chinamap)的引用。

(2) 用<map>标记设置图像地图的作用区域，并取名为 chinamap。

(3) 分别用<area>标记针对每个省(直辖市)的位置划分出多边形作用区域，并设置好其链接参数 href。

完成后的 HTML 文档如下：

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>My China Map</title>
</head>
<body>
<img src="image/chinamap.jpg" width="450" height="350" alt="中国地图" hspace
="10" align="center" border="0" usemap="#chinamap" >
<map name="chinamap">
  <area href="MyBeijing.htm"  target=_blank alt="首都北京" title="北京欢迎你"
  shape="circle" coords="340, 139, 13">
  <area href="MyShandong.htm" target=_blank alt="山东省"  title="山东欢迎你"
  shape="polygon" coords="339, 158, 355, 157, 360, 164, 366, 158, 382, 157,
  359, 185, 346, 187, 334, 182, 330, 173, 342, 156">
</map>
</body>
</html>
```

在 IE 浏览器中浏览上述文件，效果如图 3-3 所示。

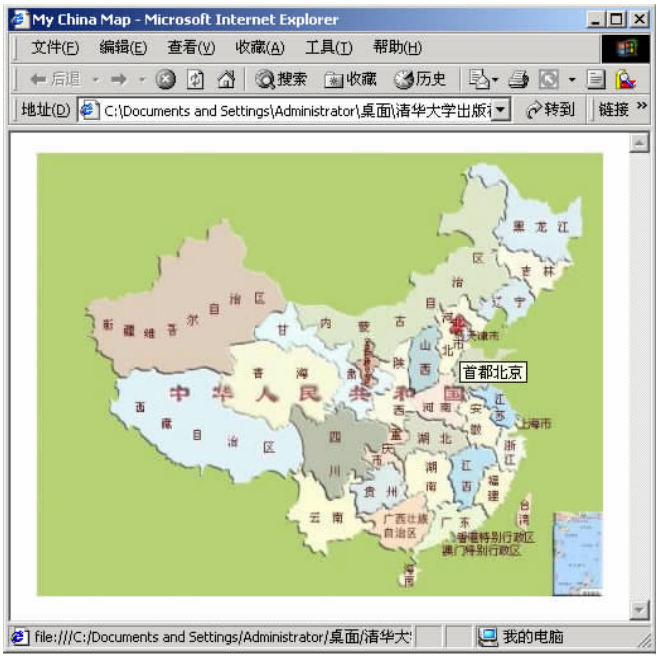


图 3-3　图像地图在浏览器中的显示

对于 Web 编程，经常在地图图像上增加 JavaScript 代码来实现一些用户逻辑。实际上 JavaScript 几乎把<area>标记与超文本链接标记<a>…一样看待。所以也可以把 onClick、onMouseOver 和 onMouseOut 等事件加到<area>标记中。

6．层次块标记 …

为了更好地实现元素的精确定位，动态 HTML（Dynamic HTML）给 HTML 增加了网页中层(layer)和位置(position)的概念。对于两种主流的浏览器，对层的支持是不一样的，IE 支持 标记，Netscape 支持<layer>标记。

 标记用于定义一个块，中间可以包含引起行中断的标记，如<table>标记等。

标记的属性介绍如下。

- id 属性　用于标记一个 块，以便引用该块。
- style 属性

例如，在 IE 浏览器中，可以用下面的方法定义 块：

```
<div  style="background-color:#ffffcc  position:absolute;  left:150px;
top:200px; width: 200px; height:20px;">文本块示例</div>
```

如果是在 Netscape 中，相同的功能应写为：

```
<layer bgcolor="#ffffcc" left=150 top=200 width=200 height=20>文本块示例
</layer>
```

在实际编写 Web 应用时，可以通过下列代码来区分两种不同的浏览器。

```
if (document.all) {
  // 编写 IE 支持的代码
  ...
}
else if (document.layers) {
  // 编写 Netscape 支持的代码
  ...
}
```

[例 3-5]　定义 div 块并将其平滑移动。

在浏览器窗口定义一个 div 块，双击鼠标左键，div 块从左向右平滑移动，单击鼠标左键，则停止移动。实现代码如下：

```
<html>
<head>
<title>Moving Div Sample</title>
<script language="javascript">
var movingID = null;
var scrolling = false;
function startMove()
{
  var left = eval(div1.style.left.replace("px",""));
  if (left <document.body.scrollWidth-150)
      div1.style.left = left+1;
```

```
    else
       div1.style.left =10;
    movingID = setTimeout("startMove()",10);
   }
   function stopMove()
   {
     clearTimeout(movingID);
   }
  </script>
  </head>
  <body onDblClick = "startMove()" onMouseDown="stopMove()">
    <div id="div1"  style="visibility:visible; position:absolute; left:10;
  top:10; z-index:1;">
       <table bgColor ="#ffffcc" border = "1" cellPadding ="0" cellSpacing = "0">
         <tr>
           <td>Div moving...</td>
         </tr>
       </table>
     </div>
     <p>双击鼠标,块开始从左向右移动</p>
     <p>单击鼠标,块开始从左向右移动</p>
  </body>
  </html>
```

上述例子不仅是为了解释 标签，更重要的是要说明<body>标签的事件属性，这不可避免地涉及到尚未介绍的 JavaScript 脚本语言，由于上述代码可读性较好，相信读者能够看懂它的意思。通过这个简单的例子，使读者可以初步体会 Web 应用的编程特点和思路。

7．多媒体标记

(1) <bgsound> 标记，用以插入背景音乐，但只适用于 IE 浏览器。主要属性包括：

- src 属性　给出 midi 音乐文件的。
- autostart 属性　设置音乐文件播送结束后的处理，如果为 true，则自动播放音乐；为 false 则结束不播放；默认值为 false。
- loop 属性　设置是否自动反复播放，loop=2 表示重复两次，infinite 表示重复多次。

(2) <embed>标记

<embed> 用以插入各种多媒体，包括声音、音乐或影像等。格式可以是 MIDI、WAV、AIFF、AU 等，Netscape 及新版的 IE 都支持该标记。

主要属性包括：

- src 属性和 autostart 属性　同<bgsound>标记。
- hidden 属性　是否完全隐藏控制画面，true 为是，no 为否(默认)。
- starttime 属性　设置歌曲开始播放的时间。如 starttime="00:30" 表示从第 30 秒处开始播放。
- volume 属性　设置音量大小，取值范围为 0~100。默认为用户系统自身设置值。
- width 属性　设置控制画面的宽度和高度。

- align 属性　设置控制画面和旁边文字的对齐方式，取值可以是 top、bottom、center、baseline、left、right、texttop、middle、absmiddle、absbottom。
- controls 属性　设置控制画面的外貌。取值可以是 console(一般正常的面板，为默认值)、smallconsole(较小的面板)、playbutton(只显示播放按钮)、pausecutton(只显示暂停按钮)、stopbutton(只显示停止按钮)或 volumelever(只显示音量调整按钮)。

8．<applet> … </applet>标记和<object>…</object>标记

为了满足网页制作更加复杂的功能，HTML 扩展了许多新的标记，用以插入脚本程序、Java 程序、内嵌对象、Active 控件等，从而制作出更加新颖、丰富和交互能力更强的 Web 页面。这些扩展的标记主要是<script>、<bssound>、<embed>、<applet>、<object>等，前面几种标记在上面已经做了介绍，下面介绍 Java 小程序标记和对象标记。

(1) Java 小程序标记<applet> … </applet>用于插入 Java 程序段，一般格式是：

```
<applet
   codebase = "codebaseURL"
   code = "appletFile"
   width = "pixels"
   height = "pixels"
   name = "appletInstanceName"
   align = "alignment"
   vspace = "pixels"
   hspace = "pixels"
   archive = "archiveList"
   alt = "alternateText"
>
   <param  name = "attribute1" value = "value">
   <param  name = "attribute2" value = "value">
    ...
    alternateHTML
</applet>
```

其中，code 属性给出 Java 小程序的来源，codebase 属性给出包含 Java 小程序的 URL，param 给出 Java 小程序的参数及调用时的参数值。alternateHTML 是当浏览器不支持 Java 小程序时显示的消息。

(2) <object>…</object>标记

<object>…</object>标记用于在网页中插入对象，除了可以插入 ActiveX 控件外，还可以插入 OLE 对象，如图像、文档、动画、小程序等。

<object>标记的一般格式是：

```
<object
   classid="clsid:a399591c-0fd0-41f8-9d25-bd76f632415f"
   width= pixels
   height= pixels
   name=browserControlInstanceName
   align= alignment
```

```
    hspace= pixels
    vspace= pixels
    archive=archiveFiles
    ...
>
   <param name=CODE value=browserControlDLL#packageName.mainClass>
   <param name=CODEBASE value=codeBaseURL>
   <param name = attribute1 value = value>
   <param name = attribute2 value = value>
     ...
    alternateHTML
</object>
```

其中，browserControlDLL **文件的名称，此文件中包括要执行的类，指定文件名时不能带有扩展名。**packageName.mainClass **扩展了** java.applet.Applet **类、并包含** init **方法的类的名称，必须指定软件包的名称。**

9．**其他标记**

除了上述的常用 HTML **标记和扩展** HTML **标记外，还有一些其他标记。**

(1**)** <marquee>…</marquee>**标记**

<marquee>**标记是一种活动字幕标记，又称“走马灯”标记。**<marquee>**的属性有：**

- align **属性　设置活动字幕的位置，取值可以是** left、center、right、top **或** bottom**。**
- bgcolor **属性　设置活动字幕的背景颜色，一般是十六进制数。**
- direction **属性　设置活动字幕的滚动方向，取值可以是** left、right、up **或** down**。**
- behavior **属性　设置滚动的方式，主要有三种方式** behavior="scroll"**表示由一端滚动到另一端；**behavior="slide":**表示由一端快速滑动到另一端，且不再重复；**behavior="alternate"**表示在两端之间来回滚动。**
- height **属性和** width **属性　设置滚动字幕的高度和宽度。**
- hspace **和** vspace **属性　设置滚动字幕的左右边框和上下边框的宽度。**
- scrollamount **属性　设置活动字幕的滚动距离。**
- scrolldelay **属性　用于设置滚动两次之间的延迟时间。**
- loop **属性　用于设置滚动的次数，当** loop=?1 **表示一直滚动下去，直到页面更新。**

默认情况下，<marquee>**标记是向左滚动无限次，字幕高度是文本高度，滚动范围为水平滚动的宽度是当前位置的宽度；垂直滚动的高度是当前位置的高度。**

由于<marquee>**标记只能作用于一段(** … **)文本，因此活动字幕为多行时，分行时只能用**
标记，不能用 **标记。例如：**

```
<marquee onmouseover=this.stop() onmouseout=this.start() scrollAmount=1
scrollDelay=60  direction=up width=150 height=200>
```

活动字幕内容第一行
活动字幕内容第二行
活动字幕内容第三行

```
</marquee>
```

上述代码将在一个区域内垂直地滚动多行，鼠标在该区域上时停止滚动，离开时继续滚动。

另外，字幕中也可以加入图像，代码如下：

```
<marquee><img src="image/logo.gif" width="20" height="20">欢迎光临
</marquee>
```

注意：如果是使用 FrontPage **编辑** HTML **文档，对于多行和插入图片的**<marquee>，FrontPage **会自动地处理成一行或者把**<img>**标记放到**<marquee>…</marwuee>**标记的外面，此时需要通过记事本程序手工处理，将多行或图片放回到**<marquee>…</marwuee>**标记内部，实现多行或者带有图片的滚动。**

(2**) 闪烁文字**<blink>…</blink>**标记**

<blink>**标记用于文字闪烁，只适合于** Netscape **浏览器，在** IE **中没有该显示效果。**

用法是：

```
<blink>闪烁的文本</blink>
```

[**例** 3-6]　**超链接标记使用示例。**

下列代码演示了<a>**标记、** **标记以及**<marquee>**标记的组合应用，当鼠标指向一个超文本链接时，在鼠标的右下角显示** marquee **效果。**

```
<html>
<head>
<meta http-equiv="Content-Language" content="zh-cn">
<title>Shandong Travel</title>
<script>
if (!document.all && !document.layers)
event="test"
//若浏览器为 Netscape,则 document.layers 为真。若浏览器为 IE,则 document.all 为真。
function showtip(current,e,text)
{  // IE 浏览器
   if (document.all && document.readyState=="complete"){
      document.all.MyTooltip.innerHTML = "<marquee style=\"border:1px solid
black\" >" + text + "</marquee>";
      document.all.MyTooltip.style.pixelLeft=event.clientX+document.body.
scrollLeft+10;
      document.all.MyTooltip.style.pixelTop=event.clientY+document.body.
scrollTop+10;
      document.all.MyTooltip.style.visibility="visible";
   }
}
function hidetip()
{
  if (document.all)
     document.all.MyTooltip.style.visibility="hidden"
 }
```

```
</script>
</head>
<body>
<a href="http://www.sdta.gov.cn/" onMouseover=showtip(this, event, "大而强富而美的山东") onMouseout="hidetip()">美丽的山东</a>
<div id="MyTooltip" style="position:absolute;visibility:hidden;clip:rect(0 150 50 0);width: 150px;background-color:lightyellow">
</div>
</body>
</html>
```

本小节的最后需要说明的是，在 HTML 中，除了上述标记外，还包括表格(table)标记、表单(form)标记和框架(frame)标记等，这些标记将分别在后续的有关章节详细介绍。

3.2.5 列表

列表(list)是 HTML 文档内容的一种重要表现形式，特别适合罗列有关信息内容，具有清晰明了，易于查阅，操作性强的特点。列表可以嵌套，显示时按层次缩进，清晰明了。HTML 中的列表分为三种。

(1) 不编号列表(unnumbered list)，标记是
…

列表包含若干个列表项(list item)，列表项标记是
…，<li>标记可以有属性 type，取值可以是 disk, circle, square。

(2) 编号式列表(ordered list)，标记是
…

列表包含若干个列表项(list item)。<ol>标记有 start 属性，可以是数字 1、2 等，用于指定开始编号。

(3) 定义式列表(definition list)，标记是<dl>…</dl>

定义列表通常包含有交替出现的定义术语(definition term)(标记为<dt>)和定义描述(definition definition)(标记为 <dd>)。

Web 浏览器通常另起一行显示定义描述。另外，<dl>标记具有 compact 属性，该属性不需要赋值，使得定义术语和定义描述在一行输出。

例如，有如下代码：

```
<UL>
  <LI  type=circle> 艺术的形式
     <OL start=1>
     <LI> 文学
     <LI> 音乐
     <LI> 绘画
     </OL>
  <LI type=disk> 科学与技术的内涵
     <DL compact>
     <dt> 科学
     <dd> 科学是指...
     <dt> 技术
     <dd> 技术是指...
```

```
        </DL>
    </UL>
```

在浏览器中的显示结果为：

- 艺术的形式
  1. 文学
  2. 音乐
  3. 绘画
- 科学与技术的内涵

科学 科学是指...
技术 技术是指...

3.2.6 表格

表格(table)是网页制作中安排布局最好的工具，因为表格不但可以很好地安排文本或图像的显示位置，而且还可以任意进行背景和前景颜色的设置。

(1) <table>…</table>标记

<table>…</table>标记对用来创建一个表格，每个<table>…</table>标记对之间包含若干<tr>…</tr>，一个<tr>对应表格的一行。每一个<tr>…</tr>标记对又包括若干个<td>…</td>标记对，它对应一行中的一个单元格。

<table>标记有许多属性，它定义整个表格的外观特性，通过表格属性可以设置表格的外观。表格的默认属性设置为无边框，并根据内容自动设置表格大小。表格属性见表 3-3。

表 3-3　表格属性列表

属性	用途
bgcolor	设置表格的背景色
background	设置背景图片
border	设置边框的宽度，若不设置此属性，则边框宽度默认值为 0，即无边框、无格线
bordercolor	设置边框的颜色
bordercolorlight	设置边框明亮部分的颜色(当 border 的值大于等于 1 时有效)
bordercolordark	设置边框昏暗部分的颜色(当 border 的值大于等于 1 时有效)
cellspacing	设置表格单元格之间空间的大小，默认值是 2
cellpadding	设置表格单元格边框与其内部内容之间的距离
width	设置表格的宽度，可以是像素值，如 width="200"，或窗口总宽度的百分比，如 width="80%"
height	设置表格中单元格的高度
align	设置表格的浮动对齐方式，只有 left 和 right 两种对齐方式

在表格属性表中，有关宽度、大小的单位用绝对像素值。而有关颜色的属性使用十六进制 RGB 颜色码或 HTML 语言给定的颜色常量名。

因为许多浏览器不支持<table>标记中的 align 属性值 center，因此往往要通过 … 标记来设置整个表格页面布局的居中对齐方式。如果

<table>标记同时含有 align 属性，以<table>标记内的 align 属性设置显示。

(2) <tr>…</tr>、<th>…</th>、<td>…</td>

一个表格由若干行(row)构成，每一行又由若干个单元格(cell)组成，另外一个表格还可能具有一个标题(head)。

根据上述表格的结构，在<table>…</table>标记对之间，用<tr>…</tr>标记对来创建表格中的每一行，表有多少行就有多少个<tr></tr>标记对；<td>则填充由<tr>和<th>组成的表格，<td></td>标记对用来创建表格中一行中的一个单元格，<td></td>标记对之间输入单元格中要显示的内容。

此外，<tr>还有 align 和 valign 属性，分别表示水平对齐和垂直对齐方式。<td>具有 width、colspan、rowspan 和 nowrap 属性。width 是单元格的宽度，单位用绝对像素值或总宽度的百分比；colspan 设置一个单元格跨占的列数(默认值为 1)，rowspan 设置一个单元格跨占的行数(默认值为 1)；nowrap 禁止单元格内的内容自动换行。

(3) <caption>…</caption> 标记

标记表格标题，具有 align 和 valign 参数，设置水平和垂直对齐方式。

[例 3-7] 使用表格示例。

```
<html>
<head>
<title>表格标记综合示例</title>
</head>
<body>
<div align="center">
       <center>
       <table  border="1"  cellpadding="0"  bgcolor="#C0C0C0"  width="400"
       height="75" >
          <caption>
           <p style="margin-right: 16"><font size="5" color="#0000FF">学生
           成绩等记表</font>
          </caption>
          <tr>
             <td align="center" valign="middle" width="40%" height="30">学
               号</td>
             <td align="center" valign="middle" width="20%" height="22">姓
               名</td>
             <td align="center" valign="middle" width="20%" height="22">高
             等数学</td>
             <td align="center" valign="middle" width="20%" height="22">英
               语</td>
          </tr>
          <tr>
             <td align="center" valign="middle" width="40%">2004000001</td>
             <td align="center" valign="middle" width="20%">源  源</td>
             <td align="center" valign="middle" width="20%">95</td>
             <td align="center" valign="middle" width="20%">90</td>
          </tr>
          <tr>
```

```
            <td align="center" valign="middle" width="40%">2004000002</td>
            <td align="center" valign="middle" width="20%">蓉　儿</td>
            <td align="center" valign="middle" width="20%">90</td>
            <td align="center" valign="middle" width="20%">96</td>
          </tr>
          <tr>
            <td align="center" valign="middle" width="40%" rowspan="2">说
            明</td>
            <td valign="middle" width="60%" colspan="3">成绩=平时*20%+期末
            *80%</td>
          </tr>
          <tr>
            <td valign="middle" width="60%" colspan="3">2004 年 6 月</td>
          </tr>
        </center>
</div>
</body>
</html>
```

上述 HTML 文档在 IE 浏览器中的显示效果如图 3-4 所示。

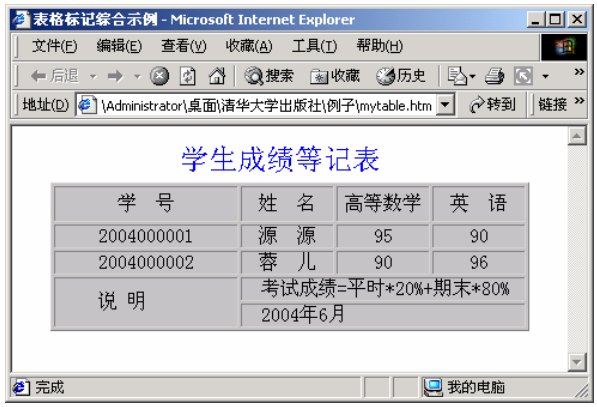


图 3-4　表格在浏览器中的显示

在上述代码中，为了避免由于浏览器窗口大小变化引起单元格和表格大小变化，在<table>标记中，使用了 width 属性，指定绝对像素值，而不是相对比例。这样当窗口的大小变化时表格的大小不变。

在表格中还可以进行表格的嵌套定义，具体情况根据实际决定，同时为了提高浏览器的显示速度，一般不易定义一个大的表格，因为浏览器是等到整个表格下载完后才显示的。

3.2.7　表单

表单(form)在 Web 网页中用来供访问者填写信息，从而能获得用户信息，使网页具有交互能力。表单和 Windows 的对话框类似，是由若干控件组成的，用于实现和用户的交互。当用户填写完信息后做提交(submit)操作，表单的内容将从客户端的浏览器传送到服务器

上，由 Web 服务器上的服务器脚本程序处理。

1．表单及输入类型

表单是 Web 中实现交互的主要界面，表单是由若干个控件构成的，完成不同的用户交互功能，最后用户提交表单，由相应的程序处理用户的输入。

(1) 表单标记<form>…</form>

<form>…</form>标记对用来创建一个表单，即定义表单的开始和结束位置，在标记对之间的一切都属于表单的内容。<form>标记的属性有以下几个。

- name 属性　给出表单的名称，常常用于脚本编程，在一个网页中，可以含有多个表单。
- action 属性　action 的值是表单处理程序的网络路径和程序名，如：<form action="http://www.abc.com/feedback.jsp">，当用户提交表单时，服务器将执行网址 http:// www.abc.com/上的表单处理程序 feedback.jsp。

另外，表单数据还可以通过 email 的方式提交，一般形式是：action=“ mailto:email 地址 ”，这样当单击提交按钮后，弹出一个提示窗口，然后发送到指定的邮箱中。

如果使用 Frontpage 编辑表单，会看到 action 属性被赋值"--WEBBOT-SELF--"。--WEBBOT-SELF--是 FrontPage 扩展所使用的伪代码。如果 Web 服务器上装有 FrontPage 扩展，在用户浏览网页时，服务器会自动把--WEBBOT-SELF--转换成服务器上相应的路径。如果用户在浏览器上得到的就是“ --WEBBOT-SELF-- ”，说明 Web 服务器上没有安装 FrontPage 扩展。

- method 属性　method 属性用来定义服务器表单处理程序从表单中获得信息的方式，有 GET 或 POST 两种传输方式。

GET 方法将数据打包放置在环境变量 QUERY_STRING 中作为 URL 整体的一部分传递给服务器。当使用 GET，服务器就分配变量 QUERY_STRING 给所有的表单数据，这个变量可存储量是有限的，一般限制在 1KB 以下。

与 GET 不同，POST 分别传递数据给服务器表单处理程序，不需要设置 QUERY_STRING 环境变量，因此 POST 有更好的安全性，表单中数据的多少是任意的，因为这些数据从来也不分配到一个变量里。此外用 POS 传递数据还有一个好处，它不会像 GET 那样把传送的数据暴露在浏览器的地址栏中，如 myform.htm?Account =abc&pwd=123456。

- target 属性　target 属性用来指定目标窗口或目标帧，显示表单处理程序的视窗名称。
- enctype 属性　属性值是"MULTIPART/FORM-DATA"。

(2) 输入控件标记<input type="">

<input type="" name = >标记用来定义用户输入区，无尾标注。此标记必须放在<form>…</form>标记对之间。一般形式是：

```
<input type="" name = "" value=""...>
```

- type 属性　给出控件的类型，常用的类型 Text，TextArea、Radio、Checkbox、Button，<SELECT> <OPTION>、Image、Hidden、Password、File Submit/Reset。
- name 属性　name 属性给出输入控件的名字，该名称可以使用户清楚当前的要做

的选择和输入的内容。

- value 属性　保存用户的输入和选择，服务器就是通过调用输入区域的 value 属性值来获得该区域的数据。另外，用户可以通过 value 属性来指定输入区域的默认值。

另外，根据 type 类型的不同，每种输入控件还有不同的其他属性以及事件属性。

2．text 单行文本框

单行文本框，一般形式是：

```
<input type="text"...>
```

主要属性有：

- name 属性　name 为文本框的名称，便于程序获取用户输入。
- value 属性　value 属性存储文本框的初始值。
- size 属性　size 属性表示文本框的显示长度。
- maxlength 属性　maxlength 是文本框中输入的有效数据长度。

例如：有 HTML 代码如下：

```
<form name="myForm" method="POST" action="/progs/feedback.jsp">
用户账户：<input type="text" name="AccountStr" size="10" value="guest"
maxlength= "8">
</form>
```

则显示效果为：

用户账户：guest

3．password(文本框)

输入密码文本框，password 文本框的属性和 text 基本上相同，两者不同是当用户输入密码时，区域内将会显示“*”号。

一般形式是：

```
<input type=" password "...>
```

例如，html 代码如下：

```
<form name="myForm" method="POST" action="/progs/feedback.jsp">
密码：<input type="password" name="myPassword" size="10" maxlength="8">
</form>
```

显示效果为：

密码：*******

4．<textarea>标记

多行文本框，又称滚动文本框。和其他的输入类型不同，它不是通过<input type=" " ...>来指定的，它是一个双标记，写作：<textarea>…</textarea>，此标记对用于<form>…</form>标记对之间。

<textarea>标记的主要属性如下。

- name 属性　name 为多行文本框的名称，便于程序获取用户输入。
- rows 属性和 cols 属性　分别用来设置文本框的列数和行数，列与行以字符数为单位。

例如，有 HTML 代码如下：

```
<table border="1" cellpadding="2" width="350" >
   <tr>
     <th width="30" align="center">简<br><br>介</th>
     <td width="300">
       <form  name="myForm" method="POST" action="--WEBBOT-SELF--">
         <textarea name="brief" rows="5" cols="40">简要说明</textarea>
       </form>
   </tr>
 </table>
```

上述代码将 form 放到了一个 table 中，主要是为了布局方便，显示结果如下所示。

5．button 控件

即按钮控件，一般形式是：

```
<input type="button"...>
```

按钮是最常使用的一种输入控件，除了具有若干属性外，同时还可以接受各种鼠标事件，主要属性包括如下。

- name 属性，name 为文本框的名称，便于程序获取用户输入。
- value 属性，value 为按钮的显示名称。

例如，有以下 THML 代码：

```
<form name="myForm" method="POST" action="/progs/feedback.jsp">
 <input type="Button" value="回前一页" onclick="history.go(-1);return
true;">
 <input type="Button" value="关闭窗口" onclick="window.close();return
true;">
 </form>
```

显示结果如下所示。

当单击“回前一页”按钮时，则回到前面打开页面，单击“关闭窗口”按钮时，当前窗口被关闭。

6．radio 控件

即单选按钮，往往是若干个 radio 为一组。每一个 radio 必须且仅有一个 value，通常有两个或以上 radio 具有相同的 name、不同的 value，从而选择其中之一。

- name 属性　单选按钮的名称，一般是若干个 radio 一组，取相同的 name 值。
- checked 属性　用来设置该单选框默认时是否被选中，相同 name 的多个 Radio 中只能有一个选择，或都不使用该参数。
- value 属性　存储单选按钮的取值，多个具有相同 name 的单选按钮应该具有不同的 value。

例如有 HTML 代码如下：

```
<form name="myForm" method="POST" action="/progs/feedback.jsp">
性别：
 <input type="Radio" name="gender" value="Female">女性
 <input type="Radio" name="gender" value="Male" checked>男性
 <br><br>
学历：
 <input type="Radio" name="degree" value="Bachelor" checked>学士
 <input type="Radio" name="degree" value="Master">硕士
 <input type="Radio" name="degree" value="Doctor">博士
</form>
```

上述代码的显示结果如下所示。

7．checkbox 控件

复选框控件，一般形式是：

```
<input type=" checkbox "...>
```

复选框是对某种输入做出“是”或“否”的选择，除了具有若干属性外，同时还可以接受鼠标事件。与 Radio 不同，每一个 Checkbox 都是独立的。

checkbox 的主要属性如下。

- name 属性　name 为复选框的名称，便于程序获取用户输入。
- value 属性　每一个 checkbox 必须有一个 value，当用户选中该 value 值，页面便会传到表单的 action 属性指定的程序中。
- checked 属性　用来设置该复选框默认时是否被选中。

例如，有 HTML 代码如下：

```
<form name="myForm" method="POST" action="/progs/feedback.jsp">
   兴趣爱好：<br><br>
   <input type="checkbox" name="intrests01" value="Sports" checked>体育
   <input type="checkbox" name="intrests02" value="Music">音乐
```

```
    <input type="checkbox" name="intrests03" value="Arts">文学
    <input type="checkbox" name="intrests03" value="Others" checked>其他
</form>
```

显示结果如下所示。

8．<select> …</select>和<option>标记

<select>…</select>标记对和<option>…</option>标记对用来创建一个下拉列表框或可以复选的列表框。它不需要在<input type=" ">中指定输入类型。

(1) <select>…</select>标记对的一般形式是：

```
<select name=" ">
<option value="  ">...</option>
<option value="  ">...</option>
... ...
</select>
```

<select>标记的属性包括：

- name 属性　name 为下拉式列表控件名称，便于程序获取用户输入。
- size 属性　下拉式列表的高度，默认时值为 1，若没有设置(加入)multiple 属性，显示的将是一个弹出式的列表框。若使用此参数则不会有 PopUp 效果。如果小于可选的项目数量，则出现垂直滚动条。
- multiple 属性　指定是否可以多选。multiple 属性不用赋值，直接加入标记中即可使用，加入此属性后列表框就成为可多选的了。

(2) <option>…</option>标记

<option>标记用来指定列表框中的一个选项，它位于<select>…</select>标记对之间。主要参数包括：

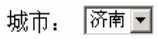
- value 属性　用来给<option>指定的选项赋值，这个值要传送到服务器上，服务器通过调用<select>区域的 name 的 value 属性来获得该区域选中的数据项。
- selected 属性　selected 用来指定默认的选项，一个下拉式复选框可以有一项指定必须选。

例如，有 HTML 代码如下：

```
<form name="myForm" method="POST" action="/progs/feedback.jsp">
城市：
<select name="city" width="100">
  <option value="beijing">北京</option>
  <option value="shanghai">上海</option>
  <option value="guangzhou">广州</option>
  <option value="hk">香港</option>
  <option value="jinan" selected>济南</option>
  <option value="qingdao">青岛</option>
```

```
</select>
</form>
```

显示结果如下所示。



如果指定 <select>的 size 属性，比如 size="5"，则显示高度为 5 的一个列表框，不出现 PopUp 效果，占用较大的屏幕空间。

9．image 控件

image 输入类型，通常用于取代 submit/reset 两个默认的按钮，来制造个性化的按钮，因为这两个按钮并不漂亮 image 输入的一般形式是：

```
<input type="image" ...>
```

主要的属性如下：

- name 属性　所要代表的按键，可以是 submit、reset 或其他。
- src 属性　设置按钮图片，如果图片文件与该 html 文件不在同一目录下，需要设置正确的相对或绝对路径。

例如，有 HTML 代码如下：

```
<form name="myForm" method="POST" action="/progs/feedback.jsp">
  <input type="image" src="myOk.gif" name="submit" width="40" height="40"
  border="0" >
</form>
```

显示结果如下所示：

10．Hidden

Hidden 为定义隐藏表单的元素，在网页上并不显示，只是随表单一起传给表单的 action 属性指定的程序。

一般形式是：

```
<input type="hidden" ...>
```

主要的属性如下：

- name 属性　所要代表的控件名称，便于程序获取用户输入。
- value 属性　默认值。

例如，有如下 HTML 代码：

```
<input type="hidden" name="myID" value="730118">
```

当表单提交后，服务器程序可以获得 myID 的值是 730118，从而实现传送数据的目的。

11．file 控件

file 是 HTML 新增加的表单控件，用于实现文件的上传操作。在介绍具体的使用以前，先介绍有关文件上载的相关知识。

（1）文件上传知识

在 TCP/IP 中最早出现的文件上传机制是 FTP，它是将文件由客户端发送到服务器的标准机制。它很可靠，能够考虑到跨平台的文本和二进制格式文件。

Web 服务器的运行机制是客户浏览器访问 Web 服务器，如果是 HTML 文件，服务器将直接把该文件发送到客户端。如果是 JSP 文件，服务器首先执行其中的服务器端脚本程序，然后将执行结果以 HTML 形式返回客户。这种运行机制决定了客户与服务器的联系采用 HTTP 协议而不是 FTP 协议。

为了实现在 HTTP 协议上上载文件，RFC1867 在表单中增加了新的输入类型，就是 <input type="file"…>。并且在表单本身中加入了不同的编码方案，它不再使用典型的

```
<form action="... .jsp" method="post">,
```

而是使用

```
<form action="... .jsp" method="post" enctype="multipart/form-data">。
```

这种编码方案在传送大量数据时比默认的"application/x-url-encoded"表单编码方案要效率高得多。因为 URL 编码只有很有限的字符集。当使用任何超出字符集的字符时，必须用"%nn"代替(nn 表示两个十进制数)。这样即使是常用的空格字符也要用"%20"代替。那么，通过 URL 编码方式上载的文件将会是原来的 2~3 倍大。而使用 RFC1867 编码方式则只是在传送数据的周围加上很简单的头部来标识文件内容。

（2）<input type="file"…>输入控件

file 类型的输入控件将显示一个文本框，用于文件名的输入，同时在文本框的后面显示一个“浏览…”按钮，允许用户通过浏览的方式选择要上传的文件。

主要的属性包括：

- name 属性　所要代表的控件名称。
- size 属性　所显示文字盒的长度。
- maxlength 属性　可输入字符的个数。
- accept 属性　接受的文件类别，有 26 种选择，但可不设置为"text/html"。

例如，有如下 HTML 代码：

```
<form name="myForm" method="POST" action="/progs/archievefiles.jsp">
  文件：<input type="file" name="upload" size="15" maxlength="50" accept=
  "text/ html">
</form>
```

显示结果如下所示。

12．submit/reset

最后介绍两个很重要的按钮，就是当表单填写完毕后，需要使用的 submit/reset 按钮，它结束表单输入，然后服务器开始处理用户的输入数据。

(1) 表单提交

表单提交按钮就是将表单内容提交给服务器，一般形式是：

```
<input type="submit"...>
```

有如下属性：

- name 属性　和其他控件的属性不同，在提交表单时 name 可以指定为一个函数，需要和 form 标记中 action 属性的程序配合。一般情况下，不需要 name 属性。
- value 属性，提交按钮的显示名字，一般为“确定”、“提交”等易于理解的名字。

(2) 重填按钮

表单清除就是要将表单中已经的输入和选择全部清除，重新填写。

一般形式是：

```
<input type="reset"...>
```

其属性和表单提交按钮相同。

下面是通过 FrontPage 在网页中插入表单的默认代码：

```
<form method="POST" action="--WEBBOT-SELF--">
  <!—webbot bot="SaveResults" U-File="fpweb:///_private/form_ results.txt"
  S-Format="TEXT/CSV" S-Label-Fields="TRUE" -->
   <input type="submit" value="提交" name="B1"><input type="reset" value="
  全部重写" name="B2">
</form>
```

显示结果如下所示：

在上述代码中，<form>标记中的 action 没有指定表单处理程序，则采用 Web 服务器端 FrontPage 扩展中一个默认的宏"--WEBBOT-SELF--"来处理用户的表单数据。这个 action 往往需要用户程序来代替，以便处理用户的输入。

[例 3-8]　利用表单计算阶乘。

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>A simple example for Form</title>
<script language="javascript">
var calcOnly = false;
function fact(n)
{
  if (n==0) return 1;
```

```
    else {
       res = n*fact(n-1);
       return res;
    }
  }
  function calcFact(n)
  {
    var res;
    res=fact(n);
    document.myForm.result.value = res;
    calcOnly=true;
    document.myForm.n.focus();
  }
  </script>
  </head>
  <body>
  <p align="center"><font size="5">使用表单示例</font>
  <hr>
  <form name="myForm">
  <div align="center">
    <center>
    <table border="1" cellpadding="0" width="303" height="62">
      <tr>
        <td colspan="3" height="26" width="292" bgcolor="#C0C0C0">
          <p align="center">计算 n 的阶乘</td>
      </tr>
      <tr>
        <td height="28" width="104">输入 n<input type="text" name="n" size="6"
        onChange="calcFact(this.value)">
        </td>
        <td height="28" width="79"><input type="button" value="阶乘等于"
        name="equ" onClick="calcFact(n.value)"></td>
        <td height="28" width="104"><input type="text" name="result" size="13"
           onChange="if (calcOnly) { alert('This is a calculated field.');}">
        </td>
      </tr>
    </table>
    </center>
  </div>
  </form>
  </body>
  </html>
```

上述网页在 IE 浏览器中的执行结果如图 3-5 所示。

通过上面的例子，可以看出以下几点：①不是所有的 form 表单都需要提交到 Web 服务器处理，即可以没有 action 属性。②如果是用 FrontPage 的 Table 进行页面布局，在单元格中插入 Form 表单控件时，生成的 HTML 代码比较混乱，甚至会出现多个 Form，此时，

要手工调整 html 代码，只要将所有的输入控件包含在<form>…</form>标记对之间即可。

图 3-5　表单在浏览器窗口中的显示

3.2.8　帧

帧(frame)可以用来将浏览器窗口划分为多个区域(子窗口)，每个子窗口中装载一个 HTML 文件。即每个 HTML 文件占据一个帧，而多个帧可以同时显示在同一个浏览器窗口中，这样的 Web 页面称为框架网页。

帧通常的使用方法是在一个帧中放置目录(即对应可供选择的链接)，然后将目录链接的内容在另一个帧中显示。

(1) <frameset>…</frameset>标记

<frameset>…</frameset>标记对放在帧的主文档的<body></body>标记对的外边，也可以嵌在其他帧文档中，并且可以嵌套使用。此标记对用来定义主文档中有几个帧并且各帧是如何排列的。它具有 rows 和 cols 属性，使用<frameset>标记时这两个属性必须至少选择一个，否则浏览器只显示第一个定义的帧。

rows 用来规定主文档中各个帧的行定位，而 cols 用来规定主文档中各个帧的列定位。这两个属性的取值可以是百分数、绝对像素值或星号(“*”)，其中星号代表那些未被说明的空间，如果同一个属性中出现多个星号则将剩下的未被说明的空间平均分配。同时，所有的帧按照 rows 和 cols 的值从左到右，从上到下排列。

例如，通过 FrontPage 2000 新建“ 标题、页脚和目录 ”框架网页，代码如下：

```
<html>
<head>
<meta http-equiv="Content-Type" CONTENT="text/html; charset=gb2312">
<meta name="GENERATOR" content="Microsoft FrontPage 4.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<title>New Page 1</title>
</head>
<frameset rows="64,*,64">
  <frame name="top" scrolling="no" noresize target="contents" src="new_
  page_2.htm">
  <frameset cols="150,*">
```

```
    <frame name="contents" target="main" src="new_page_3.htm">
    <frame name="main" src="new_page_4.htm">
  </frameset>
  <frame name="bottom" scrolling="no" noresize target="contents" src="new_
  page_5.htm">
  <noframes>
  <body>
  <p>此网页使用了框架,但您的浏览器不支持框架。</p>
  </body>
  </noframes>
</frameset>
</html>
```

上述代码将屏幕分成了三行，其中第二行又分成两列，一共四个帧，每个帧对应一个文件，对应<frame>标记中的 src 属性，另外一个文件存储上述的代码，即框架网页，或称为主帧网页。显示结果如图 3-6 所示。

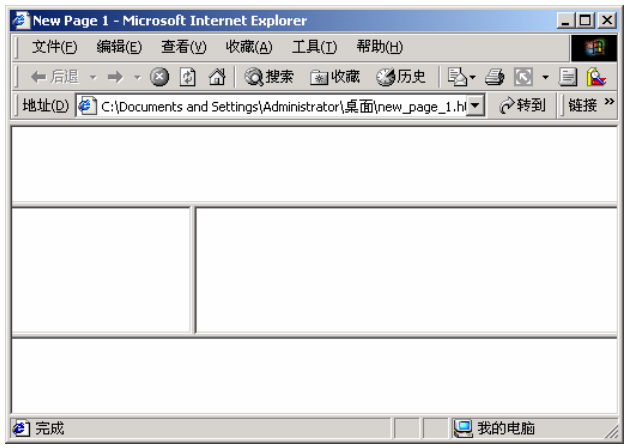


图 3-6 框架网页在浏览器中的显示

(2) <frame>标记

<frame>标记放在<frameset>…</frameset>之间，用来定义一个具体的帧。<frame>标记具有 src 和 name 属性，这两个属性都是必须赋值的。src 是此帧的源 HTML 文件名(包括网络路径，即相对路径或网址)，浏览器将会在此帧中显示 src 指定的 HTML 文件；name 是此帧的名字，这个名字是用来供超文本链接标志<a href="" target="">中的 target 属性指定链接的 HTML 文件将显示在哪一个帧中。

例如，定义了一个帧，名字是 main，在帧中要显示的 HTML 文件为 mint.htm，则代码是<frame src="mint.htm" name="main">。如果有一个链接，在单击了该链接后，需要在 main 的帧中显示文件 newone.htm，则代码为链接中的文本。这样一来，就可以在一个帧中建立网站的目录，加入一系列链接，当单击链接以后在另一个帧中显示被链接的 HTML 文件。

此外，<frame>标记还有 scrolling 和 noresize 属性，scrolling 用来指定是否显示滚动条，取值可以是“yes”(显示)、“no”(不显示)或“auto”(若需要则会自动显示,不需要则不显

示)。noresize 属性直接加入标记中，不需赋值，它用来禁止用户调整一个帧的大小。

(3) <noframes>…</noframes>标记对

有的浏览器不支持框架网页，此时需要使用<noframes>…</noframes>标记对，在不支持帧的浏览器中编辑传统的<body>…</body>部分。

3.2.9　使用层叠样式表 CSS 技术

层叠样式表(Cascading Stylesheet，CSS)技术可以有效地对页面的布局、字体、颜色、背景和其他效果实现更加精确的控制，从而控制整个页面的风格。式样单放在页面中，通过浏览器解释执行。

1．使用内联样式

style 属性可以应用于任意 body 元素(包括 body 本身)，除了 basefont、param 和 script。这个属性将任何数量的 CSS 声明当作自己的值，每个声明用分号隔开。

例如：

```
<p style="color: red; font-family:'New Century Schoolbook', serif">红色 NewCentury Schoolbook 字,如果字体可用的话。</p>
```

注意：在 style 中 New Century Schoolbook 包含在单引号中，因为双引号被用作包含样式声明。

内联样式比其他方法更加灵活。要使用内联样式，必须在文档的<head>…</head>部分包括以下标记：

```
<meta http-equiv="Content-Type" content="text/css">
```

因为与需要展示的内容混合在一起，内联样式会失去一些样式表的优点。例如，当一个样式要应用到所有文档的某种元素(标记)，或者将一个样式用到某个元素的一个场合，而不是全部，此时应使用 id 属性代替 style 属性。

id 属性用于定义一个元素的独特的样式。一个 CSS 规则如#myID1 { font-size: larger }，可以通过 id 属性应用到 HTML 中：

```
<p id=myID1>欢迎使用 ID 属性</p>
```

2．<style>…</style>标记

<style>标记放置在 html 文档的<head>…</head>内，用于定义样式。使用<style>标记可以为网页设置不同的样式属性，一般形式为：

```
<style type="text/css">
  标记 { 属性名: 属性值; 属性名: 属性值; }
</style>
```

在<style>和</style>之间的部分是一系列的样式定义,实际上就是对 HTML 默认的标记重新定义它们的属性值，做到个性化。

例如，要设置整个文档的文字颜色和背景色，可以定义样式为：

```
<style type="text/css">
  body { color: black; background: white; }
</style>
```

例如，要个性化超链接地显示，可以定义下面的样式。

<style type="text/css">

```
 a:hover { color:#FF0000;
           text-decoration:none;
           font-weight:bold}
 a {color:#0000FF;text-decoration:none; font-size:14px}
</style>
```

这样，对于文档中的超链接(<a>…)，文字显示为蓝色，当鼠标指向时则显示红色，同时字体也为 14px，不显示下划线。

3．样式(.css)文件

可以将所有的<style>定义保存为一个独立的文件，扩展名为.css。然后，当某个网页需要使用其中的样式时，在文档的<head>…</head>中增加<link>标记，一般形式如下：

```
<link type="text/css" rel="stylesheet" href="mystyle.css">
```

4．使用样式与 class 属性

对于 HTML 的标准标记，当定义了对应的样式后，文档中的标记将按照新的定义显示。如果要对某个具体的标记使用样式，则应该为这个特定的标记命名，并在标记中使用 class 属性。class 属性用于指定元素使用的样式。

例如，如果想对某个特定的<h1>标题指定上下间距，而不是所有的<h1>标记，则应该为这个特定的标题命名，并在标记中使用 class 属性，示例如下。

样式定义格式为：

```
h1.myH1{ color:#ff0000; margin-top: 1em; margin-bottom: 2em; }
```

则有下面两种<h1>标记的使用：

```
<h1>Old Headline One</h1>
<h1 class="myH1">New Headline One</h1>
```

以上两种标记的显示效果是不同的，其中第一行的显示未变，第二行采用了用户自定义样式，标题 1 的显示为红色，并且增加了上下行间距。

除了 HTML 中的标准标记可以定义样式外，用户还可以定义自己的新样式类。例如，定义一个.css 文件 pub.css，内容如下。

```
.border {
    border: 1px solid #63A6D6;
}
.myword1 {
```

```
    font-family: "宋体";
    font-size: 14px;
    font-weight: normal;
    color: #000000;
    text-decoration: none;
}
```

在其他的 HTML 文档中就可以包含该文件，并引用里面的样式类。例如：
首先，在文档的头部增加<link href="pub.css" rel="stylesheet" type="text/css">。
然后，可以在 HTML 的文档体内使用上面的样式如下。

```
<p class="myword1">
...
</p>
```

3.3　可扩展标记语言 XML

和 HTML 一样，可扩展标记语言 XML(eXtensible Markup Language)也来源于 SGML，是 SGML 的一个子集。XML 1.0 标准于 1998 年 2 月 10 日公布。它的公布引起了人们的广泛关注，XML 被认为是继 HTML 和 Java 编程语言之后的又一个里程碑式的 Internet 技术。

3.3.1　XML 简介

HTML 的出现无疑是 Internet 技术和 Web 技术的一次突破，它第一次使人们能够在 Web 上浏览和显示多种格式的数据，为推动 Internet 和 Web 技术的发展发挥了巨大的作用。可以说，如果没有 HTML，Web 技术就不可能发展到今天。然而，随着 Web 技术的飞速发展，Internet 上的 Web 信息越来越多，内容越来越复杂，数据格式也越来越多，传统的 HTML 的有限标记功能已经无法满足表达日益丰富的数据形式的需要。在这种背景下，XML 技术应运而生。

1．HTML 的问题

由于 Web 技术的飞速发展，传统的 HTML 存在着如下的问题和不足。

(1) HTML 的标记(tag)集合是固定的，大约 70 个。随着 Web 技术的飞速发展，新的数据格式不断产生并需要在网上展示，这就要求有一种比较灵活的标记机制来满足不断发展的 Web 内容的需要。但标准的 HTML 语法格式过于简单，又不允许用户根据自己的需要来创建新的标记。这将无法支持那些专门的页面格式，例如数学公式、化学方程式、音乐乐谱、财务报表以及工程应用等。

(2) DHTML 带来的问题，由于标准 HTML 已经无法满足用户的需求，人们在其基础上增加了动态的成分，如脚本程序等。但是，这些非标准技术制作的网页在不同的浏览器之间互不兼容。

(3) HTML 只是一种表现技术，它并不能揭示 HTML 标签所标记的信息的任何具体含

义。例如，语句<h1>Apple</h1>是表示在 Web 浏览器中用标题 1 显示文本“Apple”，但 HTML 标记却没有表明“Apple”究竟代表什么意思，它可能是指一种水果，也可能是一个公司的名字，或者是一个别的什么东西。HTML 当初在制定规范时并没有考虑到这方面的功能。

另外，随着 Web 文件变得越来越大、越来越复杂，Web 内容提供商已经开始感受到普通的 HTML 已经无法提供用于大规模的商业出版所需要的扩展性、结构化和数据检查功能。

2．XML 的产生

1996 年 8 月，那些关心 SGML 的专家聚集在美国西雅图，成立了一个名为 GCA（Graphic Communications Association，图形通信协会）的组织。在这次会议上，GCA 探讨了 SGML 对 Web 技术的影响。

它们的讨论主要集中在如何开发 SGML 以便它适应和促进 Web 技术的发展。但在 1996 年 8 月，在 Internet 上流行的是 HTML 技术而不是 SGML 技术。虽然 SGML 是一个设计严格的、完整的系统，但是 SGML 标准并没有考虑其在软件应用中是否易于使用。为了使 SGML 被更多的人理解和接受，就必须对 SGML 过于复杂难于被理解和实现的方面进行简化。专家们迅速着手对 SGML 进行精简，去掉其语法定义部分，适当简化 DTD 部分，并增加了部分互联网的特殊成分。为了体现它与 HTML 的不同，工作组将其命名为 XML（eXtensible Markup Language），同时也将自身更名为 XML 工作组。1998 年 2 月 10 日，XML 工作组正式向 W3C 提交了 XML 的最终推荐标准，这就是 XML 1.0 标准。

经过近五年的发展，XML 已经被广大编程人员、数据库开发者和文档编写人员所接受，W3C 所支持的另外两种语言 SMIL（Synchronized Multimedia Integration Language，同步多媒体接口定义语言，用于更好的将音频和图像同步化以便于进行 Web 传输）和 MathXL（数学标记语言，用于显示和处理数学公式），它们都是以 XML 为基础的。

今天，XML 已经逐渐成为整个 Web 的基本结构和未来各种发展的基础，由于 XML 能针对特定的应用定义自己的标记语言，这一特征使得 XML 可以为电子商务、政府文档、报表、司法、出版、联合、CAD/CAM、保险机构、厂商提供各具特色的独立解决方案。

3.3.2 创建 XML 文档

XML 文档是一个纯文本文件，可以用任意的文本编辑器编写，如 Windows 中的“记事本”（Notepad）程序。为了提高编写效率，市场也有一些专门的可视化 XML 创作及编辑工具，例如：Vervet Logic 的 XML Pro 2.0（http://www.wervet.com），微软的 XML Notepad 1.5（http://msdn.microsoft.com/xml/notepad/intro.asp），用户可以从上述站点下载其 Beta 版本。除非另作说明，以下将用 Windows 的 Notepad 作为 XML 文档编辑工具。

1．XML 文档的组成

一个 XML 文档一般包含以下三个部分。

（1）XML 文档声明

位于文档的第 1 行，一般形式为：

```
<?xml version="versionNumber"  [encoding="encodingValue"] [standalone
="yes/no"] ?>
```

其中，VersionNumber 为 XML 文档所遵循的 XML 规范的版本号；可选项 encoding 标志 XML 处理器使用的字符集，默认值为 UTF-8(适合美国英语的 Unicode 压缩版)；可选参数 standalone 取值为 yes 或 no(取决于该文档是否依赖于其他 XML 文档)，默认值为 yes。

例如，<?xml version="1.0" encoding="GB2312" ?>就是一个 XML 文档的声明，并指出使用 GB2312 字符集。

(2) 文档类型定义

在 XML 文档声明和 XML 文档之间的部分是文档类型定义(DTD)部分，一般形式是 <!DOCTYPE …>，如不需要可以省略。

(3) XML 标识的文档内容

XML 文档主要内容如下。

① 声明根元素

每一个有效的 XML 文档有且仅有一个根元素。根元素是在一个 XML 文档中包含所有其他元素的元素，无论是在语法上还是逻辑上，根元素位于所有数据的顶层。根元素的声明和其他元素的声明方法一样，一般形式为：

```
<rootElementName>...</rootElementName>
```

rootElementName 是根元素的名称，必须成对出现，且区分大小写。根元素在逻辑上代表了数据的顶层，它必须位于 XML 声明的下面一行。

② 声明非根元素

在 XML 中，是通过在容器元素中嵌套被包含元素来描述数据对象的，一个被包含元素又可以包含自己的元素。包含其他元素的元素称为容器(container)，所有的非根元素都包含在根元素中，根元素是最上层的容器元素。

```
<containerElement [attributesList]>
   [<containedElement[attributesList]>
    </containedElement>]
   ...
</containerElement>
```

例如用 XML 来描述一部汽车数据。汽车包括发动机、发动机又包括活塞等，用 XML 描述如下：

```
<automobile>
   <engine>
      <piston>... </piston>
      ...
   </engine>
  </automobile>
```

③ 数据元素属性

一个数据元素可以有若干属性，属性必须在一个元素的起始标记中声明，一般形式为：

```
<elementName [属性名 1="属性值"] [属性名 2="属性值"] ...>elementValue
</elementName>
```

其中，元素名和属性名必须以字母或下划线开始，并且只可能包含字母、数字、下划线、连字符和句点。

例如，为汽车定义三个属性为车牌号、车主和制造商，可以将<automobile>标记写为<automobile number="123456" owner="Brion" manufacture="Ford">。

④ 定义名称空间

在 XML 中，用户可以自己定义标记和命名元素。因此，如果把多个 XML 文件合并为一个，就很可能出现冲突。名称空间(namespaces，也称命名空间)就是为此而设计的。

XML namespaces 的严格定义是，“namespaces 是用 URI 加以区别的、在 XML 文件的元素和属性中出现的所有名称的集合”。在没有 namespaces 的 XML 1.0 文件里，元素和属性中出现的名称被称为“本地名称”(local names)。XHL nanrespaces 定义的一般形式为：

```
<namespace: elementName xmlns: namespace="globalUniqueURI">
  [<namespace: containedElement [namespace: attributeName="attribute
  Vaue"]>
   </namespace: containedElement>]
</namespace: elementname>
```

其中，namespace 是名称空间的惟一名称，elementName 是应用名称空间的 XML 文档元素的名称。globalUniqueURI 是统一资源标识符(Uniform Resource Identifier，URI)。attributeName 和 attributeVaue 是和包含元素 containedElement 相关联的一个属性的名称和属性值。

定义名称空间的目的是惟一地标识一个元素或一组元素的属性。例如：

```
<r:customer xmlns:r="http://www.ABCStore.com/CustomerURI">
   <r:name  r:Address="Beijing">Cherry</r:name>
</r:customer>
```

⑤ 包含非标准文本

通过预定义 XML 实体可以在 XML 文档中加入特殊符号，如果需要大量的特殊符号，可以使用 CDATA 段。CDATA 段可以使用户在一个 XML 文档中引用大量的特殊符号文本块，而不需要分别以实体的形式来代表每一个特殊字符。一般形式为：

```
<![CDATA[text]]>
```

text 是包含特殊字符的文本串，该文本不被 XML 分析器检查。XML 处理器负责分析或者以一种有意义的方式使用该文本块。

[例 3-9] 一个简单的 XML 文档。

Brion 给 Jane 的便条使用 XML 格式，如图 3-7 所示。

按照 HTML 的思想，双击该文档将其在浏览器中打开，显示效果如图 3-8 所示。

这个 XML 文档做了什么呢？好像什么都没做。我们需要从观念上转变，不能以 HTML 的思想来理解 XML。XML 不是 HTML 的替代品，HTML 设计的目的是用来显示数据的，重点是显示数据以及如何使数据的显示更美观。XML 是用来存放数据的，XML 设计的目

的是用来描述数据结构以及存储数据。XML 文档只是用 XML 标记来存储信息的文件，目前已编写出专门的软件来发送、接收并显示这种格式的信息。

图 3-7　一个简单的 XML 文档

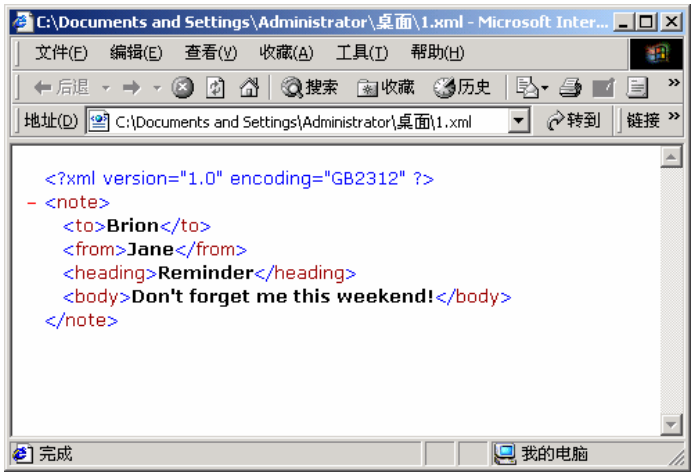


图 3-8　XML 文档在浏览器中的显示效果

2．XML 分析器

XML 文档需要在 XML 分析器中才能运行，XML 分析器有确认型和非确认型两种。确认型 XML 文档分析器检查 XML 文档的语法，将 XML 文档同文档类型定义 DTD 和架构作比较。如果确认规则，XML 分析器还要判断 XML 数据是否和预定义的确认规则相符。非确认型 XML 分析器也进行 XML 文档语法的检查，但不进行 XML 文档和 DTD 及架构的比较。

在微软的 Internet Explorer 5.0 浏览器中内置了 XML 确定性分析器，即 MSXML。在下面的例子中，将以 IE 5.0 显示 XML 文档。

3.3.3 使用文档类型定义 DTD

与 HTML 相比，XML 本身不是一种语言，而是定义语言的一个系统，它不像 HTML 一样拥有一个通用的标记(tag)集合。在 XML 中，标记(在 XML 中又称元素)是通过文档类型定义(Document Type Difinition，DTD)来实现的。DTD 定义了 XML 文档中可以使用的标记符号、标记的属性、标记的排列方式/顺序、标记能够包含的内容等。

在 XML 文档中，DTD 出现在第一行的文档类型声明和文档的根之间。有了文档类型定义，可以根据行业的特点，定义不同行业的 DTD，从而实现行业内部组织之间的数据表示和数据交换。

DTD 可以在 XML 文件中直接设置，也可以保存为一个完全独立的文件(.dtd)。因此，DTD 分为内部 DTD(在 XML 文件中直接设置 DTD)和外部 DTD(在 XML 文件中调用已经编辑好的 DTD 文件)两种。例如，对于一家企业，有自己的供应商、客户、合作伙伴，他们相互之间的交换电子文档都是用 XML 文档。那么可以将这些 XML 文档的 DTD 保存为一个独立的 DTD 文件，存放在某个地方，让所有要交换的 XML 文档都使用此 DTD。

1．声明内部 DTD 和 XML 元素

一个内部 DTD 声明必须写在 XML 声明和 XML 根元素之间，一般形式为：

```
<!DOCTYPE rootElement [dtdRules]>
```

其中，<!DOCTYPE 表示开始设置 DTD；rootElement 指定此 DTD 的根元素的名称，一个 XML 文件只能有一个根元素。注意，如果 XML 文件使用了 DTD，那么文件中的根元素就在这里指定。dtdRules 是在 rootElement 中为一个元素定义 DTD 规则的一条或多条 XML 语句，即定义 XML 文件使用的元素。

DTD 规则的一般格式为：

```
<!ELEMENT element-name (element-definition) >
```

其中，!ELEMENT 是 XML 的保留字，表示开始元素定义。element-name 是为元素起的名称，element-definition 是对元素的定义，即<元素>...</元素>之间能够包含什么内容。元素的内容可以是一般性文字，也可以是其他元素。

其中，元素定义可以是：

EMPTY | ANY | #PCDATA | 元素

- EMPTY　没有内容的元素。在 XML 文件中，空元素不需要结束标记，但必须用</空元素名>这样的写法。
- ANY　表明为所有可能的元素及可解析的数据。
- #PCDATA　只有可解析有字符数据才能作为元素的内容。
- 元素　如果一个元素又包含另外的元素，那么这个元素就是容器元素。

声明容器元素的基本语法为：

```
<!ELEMENT containerElement(containedElement1,containedElement2,...contained
ElementN)>
```

其中，containerElement 为容器元素，containedElement1 至 containedElementN 为被包含元素。被包含元素可以取下列三种格式之一：

- element：要求该元素有且只有一个值。
- element+ 要求该元素有一个或多个值。
- element* 要求该元素有零个或多个值。

例如，<!ELEMENT 书籍（名称,作者,价格)>表示定义了一个元素(即标记)“ 书籍 ”，并且它包含三个子元素元素“ 名称 ”、“ 作者 ”和“ 价格 ”。<!ELEMENT 名称 (#PCDATA)>则定义了一个元素“ 名称 ”，它由#PCDATA 关键字定义，表明此元素仅仅包含一般文字，是基本元素。

在一些容器元素的声明中，有可能它包含的子元素是在多个子元素中的一个，那么在声明此父元素时，就可以把它声明成选择性元素，可供选择的子元素用“ | ”分隔。例如<!ELEMENT 配偶（妻子|丈夫)>。

2．在 DTD 中声明元素属性

与 HTML 标记一样，XML 元素往往也包含不同的属性。在 XML 中，元素属性设置的一般形式是：

```
<!ATTLIST element-name attribute-name Type Default-value>
```

其中，<!ATTLIST 表示开始属性的设置；element-name 表示元素名；attribute-name 是设置的元素的属性名称；Type 是该属性的属性值的类别，属性值的种类见表 3-4。

表 3-4　属性值种类

属性值类别	描述
cdata	属性值为一般文字
enumerated	列出该属性的取值范围，一次只能有一个属性值能够赋予属性
nmtoken	表示属性值只能由字母、数字、下划线、：、- 这些字符组成
nmtokens	表示属性值能够由多个 nmtoken 组成，每个 nmtoken 之间用空格隔开
id	该属性在 XML 文件中是惟一的，常用来表示人的身份证号码
idref	表示该属性值是参考了另一个 id 属性
idrefs	表示该属性值是参考了多个 id 属性，这些 id 属性的值用空格隔开
entity	表示该属性的设置值是一个外部的 entity，如一个图片文件
entities	该属性值包含了多个外部 entity，不同的 entity 之间用空格隔开
notation	属性值是在 DTD 中声明过的 notation(声明用什么应用软件解释某些二进制文件，如图片)

Default-value 是指该属性的默认值种类，有四种不同的属性默认值，分别是：

- #required　表示在标记中必须出现此属性。
- #implied　标记中可以不出现此属性。
- #fix　属性的值是固定的某个值。
- 字符串　标记中如没有指定属性的值，那么此字符串就是此属性的值。

下面是几个属性设置的例子。

- <!ATTLIST 姓名 性别 (男|女) "男"〉>

此元素属性设置语句为“姓名”这个元素设置一个名为“性别”的属性，此属性的属性值类别是 Enumerated，取值范围为“男”或者“女”。如果在 XML 文件中没有为此属性赋值，属性内定值是一个字符串“男”。

- <!ATTLIST 姓名 号码 ID #REQUIRED>

该属性设置语句为“姓名”元素设置一个名为“号码”的属性，属性值类别是 ID，意味着在 XML 文件中为此属性赋值时，值在此 XML 文件中是惟一的，否则将出现解析错误。此属性设置中的属性内定值为#REQUIRED，表示这个属性在 XML 文件的<姓名>标记中必须出现，否则解析会发生错误。

- <!ATTLIST 电话号码 国家代码 CDATA #FIX "86">

该属性设置语句为“电话号码”这个元素设置一个名为“国家代码”的属性，该属性的值是一般的文字。在<电话号码>标记中不能够设置该属性，因为这个属性被设为具有固定值的属性(#FIX 关键字)，解析器会自动地将该属性以及值“86”加到<电话号码>标记中。

3．定义实体与字符数据段

在 XML 的 DTD 中，还可以定义实体(entity)。实体实际上起一种类似“宏”的作用，一些常用的或者不便于直接书写的文字或数据，可以用一个标识定义下来，在数据中可以直接引用。

(1) 预定义 XML 实体

除了五种预定义的实体(字符&、<、>、" 和 ’ 分别表示为&、@lt;、>、"和')外，XML 还允许用户自己定义实体，例如，如果在 XML 文档中需要频繁使用词组“Good Luck”，可以在 DTD 文档中加入<!ENTITY gl "Good Luck">。这样，当使用词组“Good Luck”时，可以敲入≷，从而可以避免拼错和重复敲入相同的信息，这里 gl 就是实体。

在 XML 中，实体可以分成内部实体、外部实体和参数实体。内部实体的一般形式为：

```
<!ENTITY  entityName  "will be replaced string">
```

如果被替换的文本很长，可能要把被替换的信息存储在一个文件中。可以通过外部实体来实现，即在实体名和文件的 URL 中使用关键字 SYSTEM，构成外部实体，一般形式为：

```
<!ENTITY  entityName  SYSTEM "URL">
```

除此之外，在 XML 中还提供了参数实体，它在实体定义中通过在实体名前插入百分号(%)来实现，百分号表示该实体为参数实体。一旦被定义，参数定义可以通过用百分号和分号包围参数名来实现。例如<!ENTITY % role"(boss | manager | employee)">。

也有如下的 XML 实体定义和属性定义：

```
<!ENTITY  p1 SYSTEM  "product.gif">
<!ATTLIST product image ENTITY #IMPLIED>
```

在相关的 XML 中，可以包含语句<product image="p1">。

(2) 字符数据段

通过预定义 XML 实体可以在 XML 文档中加入特殊符号，如果需要大量的特殊符号，可以使用字符数据段，称为 CDATA 段。CDATA 段可以使用户在一个 XML 文档中引用大量的特殊符号文本块，而不需要分别以实体的形式来代表每一个特殊字符。其一般形式为：

```
<![CDATA[text]]>
```

其中，text 是包含特殊字符的文本串，该文本不被 XML 处理器检查。XML 处理器负责分析或者以一种有意义的方式使用该文本块。其中的左右方括号不能省略。

4．声明并保存外部 DTD 文件

如果希望将 DTD 声明应用到其他 XML 文件中，应该将 DTD 文本保存为一个独立的.dtd 文件，然后在 XML 文档中引用。这样一个 DTD 就可以被多个文档调用，从而减少资源浪费。

创建一个外部 DTD 文件的语法为：

```
<!DOCTYPE rootElement SYSTEM "dtdFile ">
```

其中，rootElement 为应用 DTD 文件的根 XML 元素，dtdFile 为外部 DTD 文件名(.dtd)。

对于.dtd 文件，除了没有内部 DTD 中的<!DOCTYPE rootElement[.....]〉语句外，其他都一样。而且有关元素数目、排列顺序、空元素设置、选择性元素、属性设置、Entity 声明等都和内部 DTD 相同。

3.3.4 使用架构 Schema

DTD 的语法相当复杂，并且它不符合 XML 文件的标准，自成一个体系。上面也仅仅是作了一个简介，目的是帮助大家读懂 DTD 文件以及在必要时创建简单的 DTD 文件，因为现在很多的 XML 应用是建立在 DTD 上的。

XML 文档可以表示结构化数据，可作为文本数据库存储数据，也可作为行业中数据交换的标准表示。这些都需要对 XML 文件的数据进行描述，如数据类型、长度等，DTD 就是完成这部分工作的。但是，DTD 采用了与 XML 文档完全不同的语法，非常复杂，需要同时有两套分析器来处理 DTD 和 XML 文档本身。

由于 DTD 的复杂和自成体系，W3C 于 2001 年 5 月正式发布了 XML Schema 的推荐标准，提出了 XML 架构(Schema)的概念。Schema 相对于 DTD 的明显优点是 XML Schema 文档本身也是 XML 文档，而不是像 DTD 一样使用自成一体的语法。这就方便了用户和开发者，因为可以使用相同的工具来处理 XML Schema 和其他 XML 信息，而不必专门为 Schema 使用特殊工具。Schema 简单易懂，懂得 XML 语法规则的人都可以立刻理解它。

其实，XML Schema 本身就是一个 XML 文件。但不同的是，Schema 文件所描述的是引用它的 XML 文件中的元素和属性的具体类型。除此以外，Schema 与 DTD 相比还有以下优点：第一，XML Schema 利用 Namespace 将文档中特殊的结点与 Schema 说明相联系，一个 XML 文件可以有多个对应的 Schema，而一个 XML 文件只能有一个对应的 DTD；第二，XML Schema 内容模型是开放的，可以随意扩充，而 DTD 无法解析扩充的内容；第三，

DTD 只能把内容类型定义为一个字符串，而 XML Schema 允许把内容类型定义为整型、浮点型、布尔型或者许多其他的简单数据类型。

经过数年的大规模讨论和开发，如今 XML Schema 已经成为全球公认的 XML 环境下首选的数据建模工具。

1．声明元素

架构(Schema)作为 DTD 未来的替代品，W3C 推荐的架构目前还不够成熟，支持架构的工具还很少。架构的发展还主要适用于原型设计，来指定 XML 数据的结构和语义规则。在 XML 中，创建架构的语法为：

```
<Schema name="SchemaName"
       xmlns="urn:schema-microsoft-com:xml-data"
       xmlns:dt="urn:schema-microsoft-com:datatypes">
```

其中，name 属性给出架构的名称。xmlns 给出一个微软的架构资源，将其作为一个隐式名称空间，从而可以使用户引用需要创建架构的特殊元素，包括 ElementType、Element、AttributeType 和 Attribute 等。xmlns:dt 属性包括微软的一个数据输入资源，作为一个显示名称空间，从而可以引用微软内置的数据类型，包括 string、integer、boolean 等。

在 XML Schema 中，声明容器元素的一般形式为：

```
<ElementType name="containerElementName"
          content="empty | eltOnly | textOnly | mixed"
          dt:type="dataType"
          model="open | closed"
          order="one | seq | many">
   <attribute type ="AttributeTypeName 1">
     ... ...
   <attribute type ="AttributeTypeName m">
   <element type=" containedElementName 1">
     ... ...
   <element type=" containedElementName n">
</ElementType>
```

其中，containerElementName 为容器的元素名称。<attribute type="AttributeTypeName $_{1...m}$">为容器元素包含的属性，containedElementName$_{1...n}$ 为在容器元素内部定义的被包含元素。其他几项说明如下。

(1) content 用来描述元素的内容模型。其取值包括：

- empty 元素内容为空。
- textOnly 元素只包含文本内容。
- eltOnly 元素只包含子元素内容。
- mixed 元素既可以包含文本，也可以包含子元素。

(2) dt:type 指定元素文本的数据类型，常用的几种数据类型及其含义见表 3-5。

表 3-5　常用的数据类型表

类型名	类型含义	类型名	类型含义
boolean	布尔型	entyties	xml 补充类型
char	字符型	enumeration	xml 补充类型
date	日期型	id	xml 补充类型
date time	日期时间型	idres	xml 补充类型
float	浮点型	idrefs	xml 补充类型
int	整型	nmtoken	xml 补充类型
number	数字型	nmtokens	xml 补充类型
time	时间型	notation	xml 补充类型
uri	统一资源指示器	string	字符串型
entity	xml 补充类型		

(3) model　元素的内容是否遵守 schema 中的定义。其取值包括：

- open　元素内容中可以添加未定义过的元素、文本等。
- closed　元素内容中能添加定义过的元素、文本等。

(4) order　定义子元素的排列顺序。其取值包括：

- one　规定元素内容按多种方式中的某一种排列。
- seq　规定元素内容按指定的顺序排序。
- many　按任意方式排列。

2．声明属性

用户可以用属性来描述 XMLSchema 元素，在声明属性以前，还需要声明一个属性类型。声明属性类型的语法为：

```
<AttributeType name="idref"
           dt:type="primitive-type"
           dt:values="enumerated-values"
           default="default-value"
           required="{yes|no}">
```

有了属性类型，在元素值中声明属性的格式如下：

```
<Attribute type="idref">
```

其中，idref 为属性名，用于内部引用。primitive-type 是指属性的数据类型，包括 entity、entities、enumeration(枚举类型)、id(惟一标识符)、idref(标识符引用)、idrefs(标识符引用集合)、nmtoken(名称记号)、nmtokens(名称记号集合)、notation、string(字符)等。enmuerated-value 是指当属性设为枚举类型时，代表枚举值。default-value 是属性的默认值，required 表示这个属性标志在运行状态下是否必须存在。

下面为一组具体应用：

```
<AttributeType name="location" dt:type="string"/>
<ElementType name="communication" content="textOnly" model="closed">
```

```
    <attribute type="location"/>
</ElementType>
```

上述内容定义了一个 communication 元素，“ communication ”元素没有包含其他的元素，只包含一个 location 属性，该属性的取值为字符串类型。

[例 3-10] 一个 XML Schema 文档(文档名为 myfilm.xml)示例。

```
<?xml version="1.0" encoding="gb2312" ?>
<Schema xmlns="urn:schemas-microsoft-com:xml-data"
xmlns:dt="urn:schemas-microsoft-com:datatypes">
<AttributeType name="上映日期"/>
<ElementType name="电影公司" content="textOnly" dt:type="string" model=
"closed"/>
<ElementType name="电影名称" content="textOnly" dt:type="string" model=
"closed"/>
<ElementType name="导 演 " content="textOnly" dt:type="string" model=
"closed"/>
<ElementType name="男 主 角 " content="textOnly" dt:type="string" model=
"closed"/>
<ElementType name="女 主 角 " content="textOnly" dt:type="string" model=
"closed"/>
<ElementType name="电影" content="eltOnly" model="closed" order="seq">
  <element type="电影公司"/>
  <element type="电影名称"/>
  <element type="导演"/>
  <element type="男主角"/>
  <element type="女主角"/>
  <attribute type="上映日期"/>
</ElementType>
<ElementType name="近期新片" content="eltOnly" model="closed">
  <element type="电影"/>
</ElementType>
</Schema>
```

3．将架构添加到 XML 文档

用户可以在一个 XML 文档的内部应用一个架构，以便在运行状态下，XML 验证分析器将架构规则应用于 XML 数据并验证该文档。语法为：

```
<rootelement xmlns="x-schema:URI">
```

其中，rootelement 为用户希望应用架构的 XML 文档的根元素。前缀 x-schema 对于微软的 MSXML 处理器是必需的。URI 为统一资源标识符，是要附加的架构的名称。

默认的 XML 名称空间(名域)为 xmlns="x-schema:URI"，它告诉解析器应该根据 URI 上的 schema(x-schema)来解析整个文档。

3.3.5 名称空间

在 XML 中，用户可以自己定义标记和命名元素。XML 文档中很可能会定义许多名字

相同而意义不同的元素或属性，尤其在把不同的 XML 文档合而为一时，更容易产生冲突。名称空间(namespaces，也称命名空间)就是为此而设计的。

名称空间也称为“名域”或“命名域”。它不是 XML 1.0 标准的一部分，而是一个被称为“Name Space in XML”的独立标准。对 XML 名称空间的严格定义是，“名称空间是用 URI 加以区别的、在 XML 文件的元素和属性中出现的所有名称的集合”。在没有名称空间的 XML 1.0 文档里，元素和属性中出现的名称被称为“本地名称”(local names)。名称空间定义的一般形式为：

```
<namespace: elementName  xmlns: namespace="globalUniqueURI">
  [<namespace:  containedElement  [namespace:  attributeName="attribute
  Vaue"]>
   </namespace: containedElement>]
</namespace: elementname>
```

其中，namespace 是名称空间的惟一名称，elementName 是应用名称空间的 XML 文档元素的名称。globalUniqueURI 是统一资源标识符(Uniform Resource Identifier，URI)。attributeName 和 attributeVaue 是和包含元素(containedElement)相关联的一个属性的名称和属性值。

定义名称空间的惟一目的是惟一地标识一个元素或一组元素的属性。例如：

```
<pr: payment xmlns:pr="http://www. microsoft.com/payroll">
<pr: employee>Lars Peterson</pr:employee>
<pr: description>Reimburse expenses</pr:description>
<pr: total>199.76</pr:total>
</pr:payment>
```

有了名称空间，用户就可以保证在文件中使用的名称是惟一的。对元素的属性 xmlns 进行定义就表示对该元素指定了一个名称空间。

如果省略 local_prefix(本地前缀)，就构成了默认名称空间。如果对一个元素定义了默认名称空间，那么该元素及其子元素，包括它们的属性都会自动地成为该名称空间的一部分，不用再在每一个元素和属性前面一一标明。例如：

```
<payment xmlns="http://www.microsoft.com/acct">
<customer>1234</customer>
<amount>500.00</amount>
<date_received>12-03-2000</date_received>
```

[例 3-11]　名称空间应用示例。有 XML 文档如下：

```
<?xml version="1.0" encoding="gb2312" ?>
<教师 xmlns="http://www.abc.edu.cn/xml/教师" xmlns:学生="http://www.abc.
edu.cn/xml/学生">
  <姓名>孔丘</姓名>
  <性别>男</性别>
  <年龄>52</年龄>
  <学生>
    <学生:姓名>颜回</学生:姓名>
    <学生:性别>男</学生:性别>
```

```
    <学生:年龄>25</学生:年龄>
  </学生>
  <学生>
    <学生:姓名>子路</学生:姓名>
    <学生:性别>男</学生:性别>
    <学生:年龄>28</学生:年龄>
  </学生>
</教师>
```

在这个例子中，教师的名称定义为本地名称，所以，对于教师所属的元素就不需加名称空间前缀了。而学生的相关元素仍然要加名称前缀。使用中应注意，名称空间的 URI 不一定要求真实存在，只要它是惟一的，不会与别的 URI 重复即可。

3.3.6 使用 CSS 格式化数据

W3C 为了解决 HTML 的结构化问题和实现 Web 中的总体外观控制，于 1996 年底公布了层叠样式表单 CSS(Cascading Style Sheet，CSS)规范。所谓层叠(cascading)是指对于容器元素指定的所有选项，将被自动地应用到其包含的所有元素中。

例如，根元素中指定文本为红色，则整个文档中的文本颜色为红色。如果用户在包含元素中指定了新的显示选项，该选项将覆盖层叠式选项。

CSS 把文档内容和格式化信息分离开来，单独把格式化信息存放在<style>...</style>标记内，或单独保存为一个.css 文件。

[例 3-12] 在 HTML 文档中，通过<style>…</style>标记引入 CSS，指定 H1 和 H2 元素的显示样式。

内容如下：

```
<html>
<head>
<title>The CSS in HTML</title>
<style type="text/css">
    H1 { font-family:楷体_gb2312;font-size: x-large; color: red }
    H2 { font-family:仿宋_gb2312;font-size: large; color: blue }
</style>
</head>
<body>
   <H1>CSS 应用示例</H1>
   <H2>CSS 应用示例</H2>
</body>
</html>
```

上述文档中，默认的<h1>和<h2>标记格式将变成用户自己在<style>中定义的形式。

1．CSS 的结构和使用规则

在 XML 文档所对应的 CSS 文档中，定义了文档中所有元素的显示样式。缺少某种元素的样式在定义时使用默认值。元素样式定义的语法格式为：

```
elementName [,elementName, ...] [.className][#ID 属性名] {
        property : value
        [;property : value...]
}
```

其中，elementName **为** XML **元素的名称，**proverty **为元素的显示特性(如字体、颜色等)，**value **为具体的显示特性值，**ClasssName **为样式类名。**

在 XML **中，一般把** CSS **定义为一个专门的** CSS **文档(**.css**)，它是一个纯文本文件，可通过以下声明导入到** XML **文档中：**

```
<? xml-stylesheet type="text/css" herf="CSS 文件的 URI" ?>
```

从上述格式中可以看到，在 CSS **中定义元素的样式，可以有以下多种方式。**

(1**) 同时为一个或多个元素定义样式。例如：**

```
name,company,price,unit,detials {
   display:block;
   font-weight:bold;
   font-size:0.8em
}
```

(2**) 定义一个样式类，通过对类名的引用，不同的元素可使用同一样式，不同位置的同一元素可使用不同的样式。见下面的例子。**

样式文件 style01.css **内容如下：**

```
student {display:block}
name.c1 {font-size:3em; color:red}
name.c2 {font-size:2em; color:green}
name.c3 {font-size:1em; color:blue}
```

下面是引用该样式的 XML **文档：**

```
<?xml version="1.0" encoding="gb2312" ?>
<?xml-stylesheet type="text/css" href="style01.css"?>
<student>
<name class="c1">张三</name>
<name class="c2">李四</name>
<name class="c3">王五</name>
</student>
```

(3**) 为特定的元素定义样式。可以为该元素指定一个** ID **属性，而在** CSS **中，通过“元素名#** ID **”的方式指定样式。例如，**style02.css **内容如下：**

```
student {display:block}
name {font-size:3em;color:red;
     font-weight:bold}
name#a1 {font-size: 2em; color:green}
```

注意：元素名和 ID **属性前面的#之间不能有空格。**

对于上面的样式，可以引用如下：

```
<?xml version="1.0" encoding="gb2312" ?>
<?xml-stylesheet type="text/css" href="style02.css"?>
<student>
  <name>张三</name>
  <name ID="a1">李四</name>
  <name>王五</name>
</student>
```

2．常用属性

CSS 中常用的格式化属性如表 3-6 所示。

表 3-6　CSS 的常用属性及其取值

属性名	含义	取值	说明
display	显示方式	block	以块显示，前后换行
		inline	和前后的元素在一行中显示(默认值)
		none	不显示
font-size	字体大小	56%, 0.5cm, 0.2in, 10pc, 10pt, 1em, 1ex, 1px	取值也可以是 xx-small, x-small, small, xx-large, x-large, medium, large, smaller, larger 等
font-style	字型	italic	斜体
		normal	正常字体
font-weight	字体粗细	mormal	正常粗细
		bold	粗体
		bolder,lighter	更粗，更细
		100,200, … ,900	九种不同的灰度
color background-color	前景色 背景色	red,green,blue 等	颜色的取值
		rgb(a,b,c)	0<=a,b,c<=255
		#0000FF	#000000--#FFFFFF
		rgb(x,y,z)	0%<=x,y,z<=100%
background-image	背景图	图像的 URL	
background-repeat	背景图重复	repeat	图像在水平和垂直两个方向上重复
		repeat-x	图像在水平方向上重复
		repeat-y	图像在垂直方向上重复
		no-repeat	图像不重复
background-position	背景图位置	top	垂直方向的顶端　写法示例：left top
		center	垂直方向的中央
		bottom	垂直方向的底端
		left	水平方向的左端
		center	水平方向的中央
		right	水平方向的右端

续表

属性名	含义	取值	说明
letter-spacing	文字间距	同 font-size	
text-align	文本对齐	left	左对齐
		center	居中对齐
		right	右对齐
text-decoration	文本划线	underline	下划线
		overline	上划线
		line-through	中划线
		none	无划线
margin	页边距	同 font-size	上、下、左、右页边距
margin-top	上边距	同 font-size	也用作段落间距和左右缩进
margin-bottom; margin-left; margin-right	下边距;左边距;右边距	同 font-size	
border-style	框线样式	dotted, dashed, solid, double, groove, ridge, inset, outset, none	边框线
border-weight	框线粗	同 font-size	
border-color	框线颜色	同 color	
text-indent	首行缩进	同 font-size	
line-height	行距	同 font-size	

下面是一个使用 CSS 建立 XML 文档的综合例子。

[例 3-13]　**一篇科技文献的 XML 文档和相应的 CSS 文档。**

paper.xml **文档内容如下：**

```
<?xml version="1.0" encoding="gb2312"?>
<?xml-stylesheet type="text/css" href="paper.css"?>
<paper>
<title>E-learning 中的个性化技术研究</title>
<author>Hao Xingwei</author>
<address>
(Shandong University,School of Computer Science and Technology ,Jinan,
250100)
</address>
<abstract>Abstract:</abstract>
<abstract ID="a1">
E-learning is now becoming a very important means in modern education, but
the existing E-learning systems are short of personalization, the mode of
teaching is so simple that the procedures of learning are almost the same.
How to build a personalization learning circumstance,...
</abstract>
```

```
<keywords>Kewords:</keywords>
<keywords ID="a1">
E-learning, Knowledge Point, Personalized service, Web discovery
</keywords>
<head>0.前言</head>
<text>
```

E-learning 是指基于网络的、电子化、数字化、多媒体的教学方式。在实际应用中，E-learning 的范畴非常广泛，它不仅包含基于互联网的网络化学习，还包括了基于多媒体资料的数字化学习。随着 Internet 的快速发展和普及，以及教育的全球化和由此带来的教育竞争的加剧，基于 Web 的 E-learning 系统已经成为许多大学和教育机构实施现代远程教育的重要手段[1]。

```
</text>
</paper>
```

paper.css 文档内容如下：

```
paper{
  display:block;
  font-size:20px;
  line-height:160%;
  text-align:center}
author{
  display:block;
  font-size:16px;
  margin-top:5px;
  margin-bottom:5px;
  }
address{
  display:block;
  font-size:16px;
  text-align:center
  }
abstract {
  display:block;
  font-size: 20px;
  font-weight:bold;
  font-style:italic;
  text-align:left
}
abstract#a1{
  display:block;
  font-size:14px;
  font-weight:normal;
  font-style:normal;
  line-height:150%;
}
keywords {
  display:block;
```

```
  font-size:20px;
  font-weight:bold;
  font-style:italic;
  text-align:left
}
keywords#a1{
  display:block;
  font-size:14px;
  font-weight:normal;
  font-style:normal;
  line-height:150%;
}
head{
  display:block;
  font-size:20px;
  text-align:left;
  line-height:150%;
}
text{
  display:block;
  font-size:18px;
  text-align:left;
  text-indent:30px;
  line-height:150%;
}
```

在浏览器中打开文件 paper.xml **时，显示的页面如图** 3-9 **所示。**

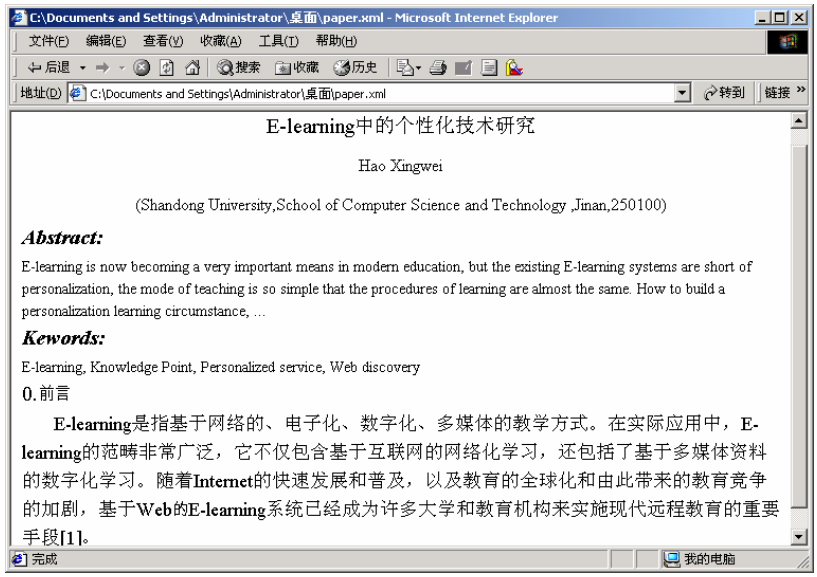


图 3-9　XML **文档显示效果**

由图 3-9 **可以看出，增加了** CSS **后的** XML **文档在浏览器中的显示类似于** HTML **文档。**

3.3.7 可扩展样式语言 XSL

CSS 通过创建样式表单元素来格式化 XML 数据，并且将其显示出来。可扩展样式语言(eXtensible Style Language，XSL)采取的方法将更加引人注目，它将文档转换到一个新的文档树中，该文档树由格式对象组成，然后应用程序就可以将文档显示出来。

XSL 由两个关键部分组成。一个为转换引擎，将原始文档树转换为能够显示的文档树；另一个为格式化符号集，该符号集可以定义应用在 XML 数据上的复杂的格式化规则。目前，IE5 仅支持 XSL 的部分功能，详细信息见 W3C 站点即 http://www.w3c.org/TR/xsl。

1．什么是 XSL

XSL 是为 XML 文件定义的一种标识语言，它提供的强大功能远远超过 CSS，如将元素再排序等。实际上简单的 XML 已可被 CSS 所解释，然而复杂的高度结构化的 XML 数据或 XML 文档则只能依赖于 XSL 极强的格式化能力展现给用户。

XSL 通过包含了一套元素集的 XML 语法规范来定义，该语法规范被用来把 XML 文件转换成 HTML 文件或 XML 文档。XSL 样式表包含了一系列数据格式规则，用以描述 XML 文档的显示方式，并负责将其转换成 HTML 等其他格式。这种转换更加容易被程序员描述。另外，XSL 还将提供多种脚本语言的通道以满足更为复杂的应用需求。因此尽管 XSL 是一项新的标识语言，但程序员完全可以继续充分发挥其所熟练的 HTML 或脚本语言的优势。XSL 凭借其可扩展性能够控制无穷无尽的标签，而控制每个标签的方式也是无穷尽的。这就给 Web 提供了高级的布局特性。

(1) XSL 和 CSS 之间的异同

① XSL 与 CSS 在很多功能上是重复的，但是它比 CSS 功能更强大。不过 XSL 的强大功能与其复杂性是分不开的。

② CSS 只允许格式化元素内容，不允许改变或安排这些内容。但是 XSL 没有这些限制，它可以提取元素、属性值、注释文本等几乎所有的文档内容。在 XML 领域，用 XSL 来格式化文档才是未来发展的方向。

(2) XSL 样式文档的基本结构如下：

① 以下面的指令作为文档开头(其中还可以包含其他属性)。

```
<?xml version="1.0"?>
```

② 通过“xsl:stylesheet”标记导入 XSL 文档的所有内容。它类似于 XML 的根元素。其中的 xmlns:xsl 指明了 XSL 所采用的标准。

③ 通过模板来描述 XML 文档的显示格式。这是 XSL 的主要部分。

④ 通过 XML 数据的引用指明显示的数据。

⑤ 其中包含了大量的 XTML 语句的各种标记。

⑥ 通过 xsl:for-each、xsl:if、xsl:choose 等标记进行数据的循环处理、条件处理、选择处理等工作。

⑦ 可以嵌入 JaveScript 或 VBScript 脚本程序，或者 JaveScript 语句，使 XSLl 具有更

强大的运算功能。

例如，上面的 XSL 文档与相应的 XML 文档结合，可以用表格的形式显示出 XML 中的数据。

2．XSL 的模板

在 XSL 中，数据的显示格式被设计细化成一个个模板，最后再将这些模板组合成一个完整的 XSL。这种方法可以使用户先从整体上考虑整个 XSL 的设计，然后将一些表现形式细化成不同的模块，再具体设计这些模板，最后将它们整合在一起。这样，宏观与微观设计的结合，更符合人们的条理化、规范化要求。由于 XML 的数据保存在具有严格层次结构的各个元素中，这种结构非常适合采用模板化的格式样式。

(1) 模板的语法结构

模板的语法结构为：

```
<xsl:template match="node-context" language="language-name">
</xsl:template>
```

其中的属性意义如下：

- match　确定什么样的情况下执行此模板。作为一种简化的说明，在此处使用标记的名字；其中最上层模板必须将 match 设为“/”。在一个 XSL 文档中根模板是惟一的。
- language　确定在此模板中执行什么脚本语言，其取值与 HTML 中的<script>标记的 language 属性的取值相同，默认值是 JScript。

<xsl:template>用 match 属性从 XML 选取满足条件的节点，针对这些特定的节点形成一个特定输出形式的模板。在一个 XSL 文档中一般要设计多个模板，在各个模板结构中描述了不同层次元素的数据显示格式、数据引用、数据处理等等内容。

(2) 模板的调用

调用模板使用<xsl:apply-templates>标记完成，其语法格式如下：

```
<xsl:apply-templates select="pattern" order-by="sort-criteria-list">
```

其中的属性意义如下：

- select　确定应调用的模板，即选取用<xsl:template>标记建立的模板。
- order-by　排序的依据，当有多个排序依据时以分号(;)分隔。

对于设计好的模板，是通过<xsl:apply-template>调用的，这样，即使日后要对这些模板作相应的修改与扩充也很方便，不致于出现互相干扰、混杂不清的情况。这种从上至下、逐层细化的设计方法，极大地减少了工作复杂程度，也大大减少了差错的产生，可以实现多人协作设计。

注意：若 XML 文档中不同标记有同名的子标记，在为其编写模板时，应把父标记作为其前缀，格式为<parent_mark/child_mark>。

3．选择模式和测试模式

选择模式就是用选择的方式将数据从 XML 中提取出来，这是一种在 XSL 中广泛应用

且操作简单的获得数据的方法。测试模式是先对选择的对象进行测试，然后对符合条件的记录进行预定的处理。这是在 XSL 中经常使用的两种模式。

(1) 选择模式

选择模式有三种不同的元素，各自的语法格式如下所示。

- xsl:Value-of

语法 <xsl:value-of select="模式">

功能 该元素的作用是从 XML 文档中提取指定节点的数据，并将数据结合到样式表中。select 属性用来指定提取的对象。

- xsl:for-each

语法 <xsl:for-each select="模式" order-by="排序列表">

功能 循环处理指定节点的相同的数据。

通过 select 属性逐个选择 XML 中的元素，按当前样式表的规定逐个处理。Order-by 属性的作用是确定排序的控制标记，属性值前加“+”按升序排列，属性值前加“?”按降序排列，不加符号时默认为升序。

- xsl:apply-templates

调用模板的元素，前面已经介绍，不再重复。

(2) 测试模式

测试模式有以下两种元素。

- xsl:if

语法 <xsl:if test="测试条件" script="script 式子" language="script 使用的语言">

功能 测试 test 属性给定的条件，当条件的值为 True 时执行该元素的内容，否则不执行该元素的内容。如果存在 script 属性，则可以以一句脚本语言表达式为测试条件。默认使用的脚本语言为 JScript。

对于测试条件，情况比较复杂。常用的为一个关系表达式或逻辑表达式，其书写规则如下面的例子所示：

(a) 姓名[="张三"] (子元素“姓名”的值为“张三”)。

(b) 成绩[.ge90] (子元素“成绩”的值大于或等于 90)。

(c) 成绩[.gt90 or .lt60] (子元素“成绩”的值大于 90 或小于 60)。

(d) @性别[.="女"] (当前元素的“性别”属性值为“女”时)。

还有一些其他的运算符，包括选择运算符和特殊字符、逻辑运算符、关系运算符以及集合运算符，由这些运算符构成的表达式可以使用在测试条件中，常用的运算符如表 3-7 所示。

表 3-7 XSL 中常用的运算符

运算符	含义	运算符	含义
/	选择子元素，返回左侧元素的直接子元素；位于最左侧的/表示选择根节点的直接子元素	//	引用任意级别的后代元素
		.	当前元素
*	通配符，选择任意元素，不考虑名字	@	属性名的前缀
!*	在相关节点上应用指定方法	@*	通配符，选择任意属性

续表

运算符	含义	运算符	含义
()*	分组，明确指定优先顺序	[ ]	应用过滤样式
:	分隔名字作用范围前缀与元素或属性	[ ] *	下标运算符，在集合中指示元素
And	或写作and或&&，逻辑与	or	或写作or或||，逻辑或
Not()	或写作not，逻辑非		
eq	相等	ieq	相等(不区分大小写)
ne	不等	ine	不等(不区分大小写)
lt	小于	ilt	小于(不区分大小写)
le	小于等于	ile	小于等于(不区分大小写)
gt	大于	igt	大于(不区分大小写)
ge	大于等于	ige	大于等于(不区分大小写)
all	所有项目均满足条件则返回“真”	any	任意项目满足条件则返回“真”
|	集合运算符，返回两个集合的联合		

- xsl:choose

语法　<xsl:choose>
<xsl:when expr="**脚本语言表达式**" language="**脚本语言类型**" test="**模式**">
<xsl:othrwise>

功能　在有多个测试条件的情况下执行满足条件(由 expr **指定测试条件)的元素内容。当所有条件都不满足时执行**<xsl:otherwise>**指定的内容。**

[**例** 3-14]　**下面的** XML **文档是成绩单数据，通过** XSL **设计显示格式，要求在显示学生成绩时，希望成绩优秀(大于或等于** 90 **分)的用粗体显示，成绩不及格的用斜体显示，其他用正常体显示。**

XML **文档** score.xml **内容如下：**

```
<?xml version="1.0" encoding="gb2312" ?>
<?xml-stylesheet type="text/xsl" href="score.xsl"?>
<成绩单>
<学生><编号>990101</编号><班级>一班</班级><姓名>袁珂</姓名><成绩>96</成绩></学生>
<学生><编号>990102</编号><班级>一班</班级><姓名>郝蓉</姓名><成绩>91</成绩></学生>
</成绩单>
```

XSL **文档** score.xsl **内容如下：**

```
<?xml version="1.0" encoding="GB2312"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<xsl:template match="/">
  <html>
  <head><title>成绩单</title></head>
  <body>
  <table border="1" width="500">
    <caption><b><font color="#0000FF" size="5">学生成绩表</font></b>
    </caption>
```

```
    <tr>
      <th width="25%" style="background-color: #ffcc99">编号</th>
      <th width="25%" style="background-color: #ffcc99">姓名</th>
      <th width="25%" style="background-color: #ffcc99">班级</th>
      <th width="25%" style="background-color: #ffcc99">成绩</th>
    </tr>
  <xsl:for-each select="成绩单/学生">
    <tr>
     <xsl:attribute name="STYLE">font:
      <xsl:choose>
        <xsl:when test="成绩[.$ge$90]">bold</xsl:when>
        <xsl:when test="成绩[.$lt$60]">italic</xsl:when>
        <xsl:otherwise>normal</xsl:otherwise>
      </xsl:choose>
    </xsl:attribute>
    <td width="25%" align="center" bgcolor="#ccffcc"><xsl:value-of select=
    "编号"/></td>
    <td width="25%" align="center" bgcolor="#ccffcc"><xsl:value-of select=
    "姓名"/></td>
    <td width="25%" align="center" bgcolor="#ccffcc"><xsl:value-of select=
    "班级"/></td>
    <td width="25%" align="center" bgcolor="#ccffcc"><xsl:value-of select=
    "成绩"/></td>
   </tr>
  </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>
</xsl:stylesheet>
```

在 XSL 文档中，用<xsl:attribute>标记定义了显示内容的属性，通过<xsl:choose>和<xsl:when>指定了字体格式的变化要求。当 XML 文档在浏览器中打开时，所显示的页面如图 3-10 所示。

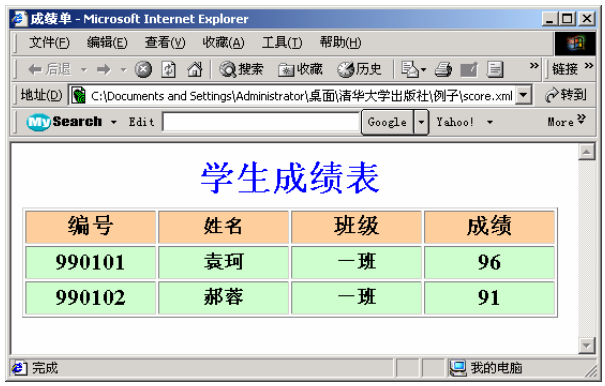


图 3-10 学生成绩单显示页面

3.3.8　创建数据岛

数据岛是指存在于 HTML 页面中的 XML 代码。数据岛允许在 HTML 页面中集成 XML，对 XML 编写脚本，不需要通过脚本或<object>标记读取 XML。几乎所有能够存在于一个结构完整的 XML 文档中的内容都能存在于一个数据岛中，包括处理指示、DOCTYPE 声明和内部子集。

1．创建数据岛

下面是一个带有数据岛的 HTML 页面，数据岛位于<xml>元素中。

```
<html>
<head>
     <title>html with xml data island</title>
</head>
<body>
     <P>Within this document is an XML data island.</P>
     <XML ID="customerXML">
           <Customer>
             <name>Cheery</name>
             <id>20020001</id>
           </Customer>
   </XML>
</body>
</html>
```

上述的 XML 数据岛是内嵌的。另外，数据岛也可以在 XML 标签中通过 src 属性来引用，例如 <XML ID="customerXML" src="customer.xml"></XML>。

2．访问数据岛

用户可以通过 ID 属性访问数据岛，“ customerXML ”称为文档对象的名称。可以利用这个对象的方法和属性访问它的根节点和子节点。在上例中，根节点是<Customer>，子节点是<name>和<custID>。下面列出了一些属性和方法，可用来访问 XML 文档的节点。

- XMLDocument　返回对 XML 文档对象模式的引用。
- documentElement　返回 XML 文档的根节点。
- childNodes　返回节点的子节点目录。
- item　通过索引访问目录中的节点。索引值从 0 开始，item(0)返 回第一 个节点。
- Text　返回节点的内容。

下面的代码访问第一个子节点<name>，并返回它的内容 Cherry。

```
customerXML.XMLDocument.documentElement.childNodes.item(0).text
```

3.3.9 文档对象模型 DOM

文档对象模型(Document Object Model，DOM)是 HTML 文档以及 XML 文档的应用程序接口。W3C 提供了精确的、与语言无关的 DOM 接口规范，可以用任何语言来实现 DOM 接口。作为 W3C 的规范，DOM 提供了一种可以应用于不同环境和应用中的标准的程序接口。它定义了文档的逻辑结构，提供了对文档进行访问和操作的方法。利用 DOM，程序开发人员可以动态地创建文档，遍历文档结构，添加、修改、删除文档内容，改变文档的显示方式等等。可以这样说，文档代表了文档中的数据，而 DOM 则代表了如何去管理这些数据。

1．DOM 的基本结构

文档对象模型 DOM 有五个基本组件，五个组件的关系如图 3-11 所示。

(1) 文档(Document)对象　这是所有数据、所有其他组件，比如注释和处理指令的节点的总的容器。

(2) 节点(Node)对象　这是文档中的每一个单独组件。节点由起始标记和结束标记定义。XML 中的一个元素在 DOM 中为一个节点对象。

(3) 节点列表(NodeList)对象　这是同一级上的所有节点形成的列表，它包含了多个节点对象。列表中的节点都有与 Sibling 相关联的关系。

(4) 解析错误(ParseError)对象　这是为每一个没有包含内容的文档创建的对象。该对象有几个属性，用来存储最后的解析错误的特性。从而可以得到错误代码、出错位置、出错原因等信息。

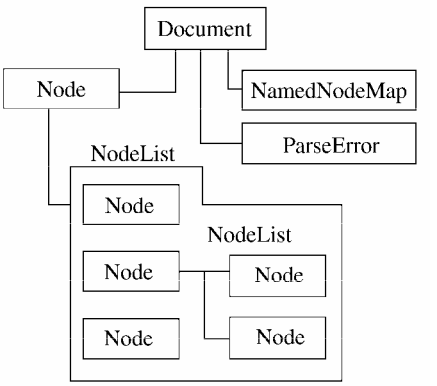


图 3-11　DOM 的五个组件的关系

(5) 名字空间(NamedNodeMap)对象　它提供了对文档中所使用的名字空间的描述。使用它的属性和方法可指定名字空间的有关属性，完成对名字空间的操作。

2．创建文档对象和加载 XML

通过 DOM 访问 XML 数据通常有两种方法：① 使用 C、C++或者 Visual Basic 等语言访问 XML 数据，此时用户必须安装一些特殊的头文件和库。② 使用 HTML 和脚本语言 JavaScript、VBScript 等通过 DOM 实现 XML 数据的访问。这里只看后者的实现方法。

在 HTML 中访问 XML，首先要创建文档对象。在 JavaScript 中，可通过以下语句完成：

```
var Mydocument = new ActiveXObject("Microsoft.XMLDOM")
```

这时创建的 Mydocument 只是一个空对象，还没有加载实际的 XML 文档。可通过 Mydocument 的 load 方法完成 XML 的加载，如下所示。

```
Mydocument.async=false;
```

```
Mydocument.load("XML 文档的 URL"));
```

这里的 async 属性赋值为 false，表明只有当文档下载完毕时，控制权才返回给调用进程。load 方法通知分析器加载指定名字的 XML 文档。XML 文档被加载后，就在内存中形成了一棵 DOM 树。

例如，books.xml 文档内容如下：

```
<?xml version="1.0"  encoding= "gb2312" ?>
  <books>
        <book status="已售完">
     <author>蓝天</author>
     <title>XML 入门</title >
 </book>
        <book status="热卖中">
          <author>白云</author>
          <title>XML 提高</title>
       </book>
</books>
```

形成的 DOM 树如图 3-12 所示。

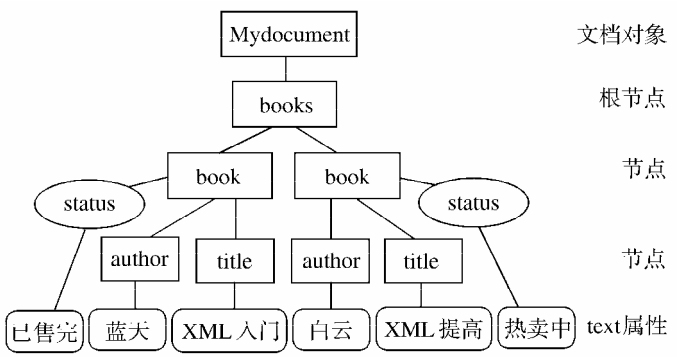


图 3-12　DOM 树示意图

从图 3-12 所示的 DOM 树可以看到，在文档对象中包含了一个与 XML 文档相一致的树。树的根节点对应 XML 的根元素，其他节点对应于 XML 中不同层次上的元素。所有节点都是 Node 类型的对象。

3．DOM 对象的属性和方法

DOM 模型中的对象都有各自的属性集和方法集。通过对象的属性和方法可以随意地处理和操作 XML 数据。下面介绍 Document、Node 和 NodeList 三种对象的属性和方法。

(1) Document 对象的常用属性和方法

Document 对象的常用属性如下所示。

- documentElement　Element 类型的只读属性。返回文档的根节点对象。
- async　逻辑类型的属性。指定是否允许同步下载文件。
- parseError　返回解析错误对象。

Document 对象的常用方法如下所示。

- createAttribute(name)　创建属性方法。
- createNode(Type, Name, namespaceURI)　创建类型为 Type、名称为 Name、并且使用 namespaceURI 作为名称空间的节点。
- load(pathname)：把文件加载到文档对象。
- loadMXL(string)：加载 XML 文档或段。

文档对象还有一些很有用的属性和方法，这里因篇幅所限不再列出，有兴趣的读者可查阅相关的书籍。

(2) Node 对象的常用属性和方法

Node 对象的常用属性如表 3-8 所示。

表 3-8　Node 对象的常用属性

属性名	意义	属性名	意义
atrribute	节点的属性集	nextSibling	当前节点的下一个兄弟节点
childNodes	当前节点所有子节点的 nodeList	nodeName	节点名
dataType	设置或获得节点数据的类型	nodeType	节点类型
firstChild	当前节点的首子节点	parentNode	父节点对象
lastChild	当前节点的末子节点	text	设置或获得节点文本
nameSpace	返回名称空间的 URI	xml	返回指定节点的 xml 表示

Node 对象的常用方法如表 3-9 所示。

表 3-9　Node 对象的常用方法

方法名	功能	方法名	功能
appendChild	添加新节点	removeChild	删除子节点
cloneNode	克隆节点	replaceChild	替换节点
hasChildNodes	当前节点是否有子节点	selectNodes	选择节点列表及其后代
insertBefore	插入节点	selectSingleNode	选择节点及其后代
parsed	节点是否被解析		

(3) NodeList 对象的属性和方法

NodeList 实际上是一个节点对象列表，它只有一个 length 属性，其值为对象中包含节点的个数。它有：item(index)；nextNode()和 reset()三个方法，详细介绍略。

[例 3-15]　对于 XML 文档 book.xml，在 HTML 中通过 DOM 模型将各本书的书名加入到一个列表控件中，单击列表控件中的书名，可以将该书的所有数据显示出来。

book.xml 文档内容如下：

```
<?xml version="1.0" encoding="gb2312" ?>
<中国古典名著>
 <书>
  <书名>三国演义</书名>
```

```
  <作者>罗贯中</作者>
  <故事梗概>略...</故事梗概>
 </书>
<书>
  <书名>西游记</书名>
  <作者>吴承恩</作者>
  <故事梗概>略...</故事梗概>
 </书>
 <书>
  <书名>红楼梦</书名>
  <作者>曹雪芹</作者>
  <故事梗概>略...</故事梗概>
  </书>
 <书>
  <书名>水浒传</书名>
  <作者>施耐庵</作者>
  <故事梗概>略...</故事梗概>
 </书>
 </中国古典名著>
```

book.htm 文档内容如下：

```
<html>
<head><title>the Chinese famous book</title>
<script language="JavaScript">
function selectbook(i)
{
  booknode=root.childNodes.item(i);
  title.innerText =booknode.childNodes.item(0).text;
  author.innerText=booknode.childNodes.item(1).text;
  S1.value=booknode.childNodes.item(2).text;
}
</script>
</head>
<body>
<h1 align="center"><font size="5" color="#0000FF">中国古典名著</font></h1>
<center>
<table border="1" width="400" bgcolor="#CCFFCC">
 <tr>
  <td width="100" height="30" align="center">书 名</td>
  <td width="270" height="30" align="center">简要介绍</td></tr>
 <tr>
   <td width="100" valign="top" align="left" >
    <select size="4" Id="D1" onChange="selectbook(D1.selectedIndex)">
     <script language="JavaScript">
      var dsobook=new ActiveXObject("Microsoft.XMLDOM");
      dsobook.async=false;
      dsobook.load("book.xml");
      root=dsobook.documentElement;
      for(var i=0;i<root.childNodes.length;i++)
      {
        booknode=root.childNodes.item(i);
```

```
        a1=booknode.childNodes.item(0).text;
        document.write("<OPTION>");
        document.write(a1);
        document.writeln("</OPTION>");
       }
      </script>
   </select>
  </td>
  <td width="270" align="left">
   <ul><li>书名:<span id="title"></span></li>
       <li>作者:<span id="author"></span></li>
       <li>故事梗概:<span id="brief"></span></li>
       <textarea rows="4" name="S1" cols="30"></textarea>
   </ul>
  </td>
 </tr>
</table>
</center>
</body>
</html>
```

保存上述两个文件，在浏览器中打开 book.htm 文档，可以看到显示的页面，单击左侧列表控件中的书名时，在右侧的列表中将显示该书的所有数据，如图 3-13 所示。

图 3-13 应用 DOM 对象示例

最后对上例做个总结，通过此例可以看出 XML 文档是对数据的一种描述，是存储数据的一种形式，而 HTML 文档的重点在于如何展示数据，理解这一点对于理解 XML 非常重要。

3.3.10 XLink 和 XPointer 规范

XLink 和 XPointer 规范是为创建 XML 超级链接而制定的。超链接是指从一个文档到

另一个文档的相互关联。无论是 HTML 文档之间，还是 XML 文档之间都可以建立相应的超级链接。但是，XML 超级链接执行要比 HTML 超链接复杂得多。XML 超链接允许开发者动态地指定链接位置，该位置既可以是 XML 文档的，也可以是 HTML 文档的。另外，XML 还允许建立双向链接。

1．建立 XML 超级链接

建立 XML 链接的语法为：

```
<linkname xlink:type="linkType" xlink:href="documentTolinkTo"> text </linkname>
```

其中，linkname 为链接的名字，linkType 取值包括 simple、extended、locator、arc、resource、title 和 none，DocumentTolinkTo 为目标文档的 URI，text 为该链接的显示文本。

对于 XML 链接，除了 type 和 href 属性外，还包括 role、title、actuate、show 等属性。

2．使用 XPointer 定位

定义 XPointer 引用可以创建一个指向目标文档内部的特定元素的链接。

XPointer 引用的基本语法为：

```
<linkname xlink:type="linkType" xlink:href="xmlDdocument#somePointer">
text </linkname>
```

其中，#somePointer 指向被链接的文档中预定义的绝对 XML id。

Xpointer 提供了绝对定位、相对定位、范围定位、属性定位和字符串定位五种不同的在 XML 文件内定位的方法，可将地址定位到相应的地方，功能上比 HTML 中的内链接更为强大。

实际上，在 XML 中定义了两种类型的链接，上述链接称为简单链接。简单链接将单个源文档和多个目标文档相链接。除此之外，XML 还允许在多个源文档和多个目标文档之间建立链接，即所谓的扩展链接。由于扩展链接的规范还在制定中，在此不做介绍。

3.4　XML 开发编辑工具

XML 文档是一种纯文本文档，可以用“记事本”、UltraEdit 等文本编辑程序来编辑，但是非常麻烦。有一些可视化的编辑工具，使用户可以像用 FrontPage 编辑 HTML 文档一样简单地编辑 XML 文档。下面对几种常用的 XML 开发工具做简要介绍。

3.4.1　XMLSpy

XMLSpy 是符合行业标准的 XML 开发环境，专门用于设计、编辑和调试企业级的应用程序，包括 XML、XML Schema、XSL/XSLT、SOAP、WSDL 和互联网服务技术，是 J2EE、NET 和数据库开发人员不可缺少的高性能的开发工具。XMLSpy 目前的版本是 Altova XMLSpy 2004 Enterprise 企业版。

从 XMLSpy 5 开始，Altova 在功能和许可证的授权上都做了一定的调整。把产品线分成了三个独立的产品线，即 XMLSpy、Authentic 和 Stylevision。XMLSpy 是其旗舰产品，包含所有的特性；Authentic 是为企业用户量身订做的；而 Stylevision 则将注意力集中在 Web 网站的开发人员身上。

详细信息请参考其官方网站 http://www.xmlspy.com/。该软件试用版的下载网址为 http://www.altova.com/download。

3.4.2 XML Editor

Oxygen XML Editor 是一款基于 Java 的 XML 编辑器，支持 XML、XSL、TXT、XSD、DTD 文档，能自行校验 XML、XSL、XSD 代码，提示脚本错误。Oxygen 能自动添加结束标签，将代码高亮显示等。该软件的官方网站为：http://www.oxygenxml.com/index.html。

3.4.3 FrontPage 2003

FrontPage 作为传统的网页编辑软件，作为 Office 套件的一部分，一直得到大量用户的支持。随着 FrontPage 2003 的推出，微软公司宣布他们的网页编辑软件 FrontPage 2003 将脱离 Office 办公套件而进行独立销售。在 FrontPage 2003 中增加了许多新的功能，除了开始支持描摹图像、层功能、交互式按钮、行为的应用、规划页面布局等功能外。FrontPage 2003 主要增强了对 XML 的支持，可以建立和编辑 XML 文档。

除了上面介绍的软件外，IBM 的 WebSphere Studio 和 Eclipse 不仅是很好的 Web 开发环境和 Java 开发环境，同时也具有 XML 的编辑功能。

3.5 要使用 XML 吗

人们已经习惯了 HTML 语言，它简单直观，并且有大量的编辑工具。为什么还要使用 XML 呢？对于普通的开发人员，难道就如本章第 3.3.1 小节所说，因为 HTML 存在的一些问题而要放弃 HTML、全部转到 XML 吗？

下面是一段来自于“水木清华站”BBS(http://www.smth.org)的转贴文章：

“这些日子，几乎每个人都在谈论 XML(Extensible Markup Language)，但是很少有人真正理解其含义。XML 的推崇者认为它能够解决所有 HTML 不能解决的问题，让数据在不同的操作系统或应用之间进行灵活交换。确实，所有的观察家们都同意 XML 将引发一场内容发布和知识交换的革命。谁先进入这个领域，谁就能够大获其利。”

(发信人：boris963(txj)，信区：XML，标题：XML 中 20 个热点问题(转载)，发信站: BBS 水木清华站(Tue Jan 13 23:17:25 2004))

下面列举了此文中提到的 20 个热点问题，以便读者了解 XML 的发展方向，从而对 XML 问题有一个自己的判断。

(1) 什么是 XML?

(2) XML 何以重要?

(3) SGML、HTML 和 XML 有什么联系？

(4) 如何实现 XML?

(5) 什么是文件类型定义(DTD)?

(6) 什么是格式完整和有效的文件?

(7) 如何在浏览器中阅读 XML?

(8) RDF 和 XML 有何联系?

(9) Netscape 浏览器中如何实现 XML?

(10) Microsoft 浏览器中如何实现 XML?

(11) OSD 和 CDF 与 XML 的关系如何?

(12) 电子商务(e-commerce)和 XML 的关系如何?

(13) XML 中的层叠样式是怎样的?

(14) XML 如何改进超链接?

(15) 服务器上支持 XML 吗?

(16) 谁应该学习 XML?

(17) 有哪些编写 XML 的工具可供使用?

(18) XML 的国际化指什么?

(19) XML 的未来在哪里?

(20) 哪里能学到更多的 XML 知识?

最后补充一点，XML 和 Web 服务(Web Services)正在成为新的热点，IBM、微软、HP、BEA、Sun、Oracle 等业界巨头都在 Web 服务领域里投入了大量的研发力量，在这其中无处不看到 XML 的身影。甚至微软在 Web 服务之前加上 XML，更为清晰地凸现了 Web 服务的特征——XML Web 服务，非常明确地告诉大家：Web 服务是基于 XML 技术的。

现在看这一节的标题“要使用 XML 吗？”，相信读者已有答案。

习　题　3

1．W3C 是一个什么机构？什么是 HTML？HTML 和 SGML 是什么关系？

2．简述 HTML 文档的基本结构。

3．通过<body>标记可以设置哪些颜色属性？

4．怎样在 HTML 中设置文本的字体、字号、文字颜色、文字加粗、文字倾斜？

5．怎样通过在 HTML 文档中设置文本段落的行距和对齐方式？

6．怎样在 HTML 文档中插入图片？<img>标记的常用属性有哪些？怎样将插入的幅 图片放在居中位置？

7．怎样设置文本或图片的超链接？

8．XML 与 HTML 相比有什么本质不同？

9．请简述 XML 文档的结构。

10．什么是名称空间？在 XML 文档中为什么要使用名称空间？

11．什么是预定义实体、内部通用实体、外部通用实体？
12．什么是 DTD？简述 DTD 和 Schema 的区别。
13．什么是 CSS？请说出 CSS 中五种不同的显示格式属性。
14．什么是 XSL？使用 XSL 和使用 CSS 有什么异同？
15．什么是数据岛？在一个 HTML 文档中如何使用 XML 数据岛？
16．文档对象模型 DOM 有哪些基本组件？
17．Document 对象有哪些常用的属性和方法？
18．Node 对象有哪些常用的属性和方法？
19．有哪些 XML 开发编辑工具？
20．简述你对 XML 热点问题的认识。

第 4 章　网页及多媒体制作

Web 网页就是 HTML 或者 XML 文档，除了满足基本的格式外，还可以在网页中添加客户端和服务器端脚本程序。这些程序和网页的超链接共同来实现 Web 应用的业务逻辑。作为纯文本文件，网页文件可以用任何文本编辑软件来编写。但是，正如本书第 3 章所述，书写 HTML 和 XML 文档是非常麻烦的。为此，一般的网页制作通常通过一些专用的可视化工具来完成。

最常用的网页制作工具有 FrontPage、Dreamweaver 等。在网页制作中，往往还用到一些多媒体素材如图像、动画等，这些多媒体素材也是网页制作的重要内容。

4.1　使用 FrontPage 2000

FrontPage 2000 是 Microsoft Office 2000 办公套件的一部分，是一个可视化的网页制作和站点管理工具。它从窗口的界面到功能的操作等各个方面，都与 Office 2000 中的其他应用程序(如 Word 2000、Excel 2000 以及 PowerPoint 2000)高度统一，从而使得 FrontPage 成为一种使用较广的网页制作工具，用户通过它可以方便地编写 HTML 文档。

4.1.1　FrontPage 2000 的主窗口

FrontPage 2000 的启动与 Windows 下的任何应用程序一样，主界面如图 4-1 所示。

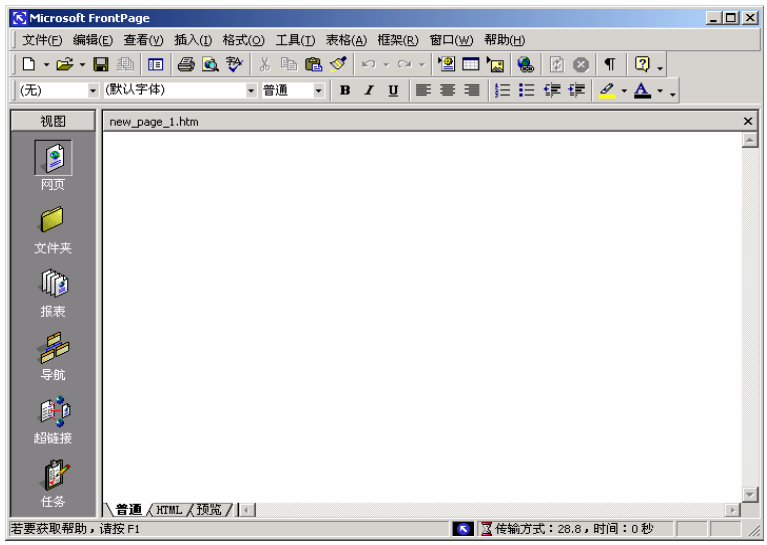


图 4-1　FrontPage 2000 主窗口

从图 4-1 可以看到，在 FrontPage 2000 的主窗口中除具有标准 Windows 窗口的组件以外，用户工作区的左边增加了一个“视图”窗格，其中的按钮可以在当前打开的 Web 站点中实现不同视图的切换，共有网页、文件夹、报表、导航、超链接和任务六种不同的视图。这里使用的是网页视图，即编辑和修改站点中的一个网页。关闭或打开“视图”窗格可以通过“查看”菜单完成。

在网页视图下，用户工作区为网页编辑区，可以在其中同时打开多个网页，这时处于最上层的为当前编辑网页，可以通过“窗口”菜单切换到其他网页。

4.1.2 显示模式

网页编辑区的左下角为一组网页视图模式的选项标签，在无框架的情况下，共有“普通”、HTML 和“预览”三个选项标签。如果是框架网页，还含有其他的几个标签。

(1) 普通模式：这是一种可视化的网页编辑模式。在这里，用户可以采用“所见即所得”的方式设计、编辑和修改网页，系统会自动生成相应的 HTML 代码。

(2) HTML 模式：在该模式下，将显示网页的 HTML 文档内容。用户可以直接在该模式下编辑、修改其中的 HTML 代码，适用于那些对 HTML 比较熟悉的用户。另外，如果网页中包含一些脚本程序，则这些脚本程序也需要通过 HTML 模式编辑。

(3) 预览模式：通过该模式可以预览当前编辑的网页，与通过 IE 浏览器所看到的网页效果一致。另外，如果网页中包含脚本程序，在程序的调试阶段通过 FrontPage 的预览模式可以很容易查出程序出错的位置和错误性质。

4.1.3 Web 站点的创建与管理

FrontPage 不仅是一个页面制作工具，它还可以进行站点管理。包括创建、删除、打开、发布站点，以及设置站点属性等工作，也可以对站点的文件和文件夹进行操作及安全性管理。

1．创建 Web 站点

FrontPage 中提供了创建站点的模板和向导，通过它们可以很容易地建立起一个新站点。操作方法是：选择“文件”|“新建”|“站点”菜单项，打开“新建站点”对话框，选择一个站点模板(这里选择“客户支持站点”)，在“指定新站点位置”下面的文本框中输入站点主目录的路径，这里输入 D:\myCustomer，如图 4-2 所示。

设置好站点的位置，选择一个向导或模板后，单击“确定”按钮。于是，在计算机上创建了一个新的文件夹 D:\myCustomer，该文件夹下又包含子文件夹和若干网页文件。

站点建立后，单击“视图”栏中的“文件夹”图标，即可以在用户工作区显示出站点的文件夹和文件的树状列表，如图 4-3 所示。

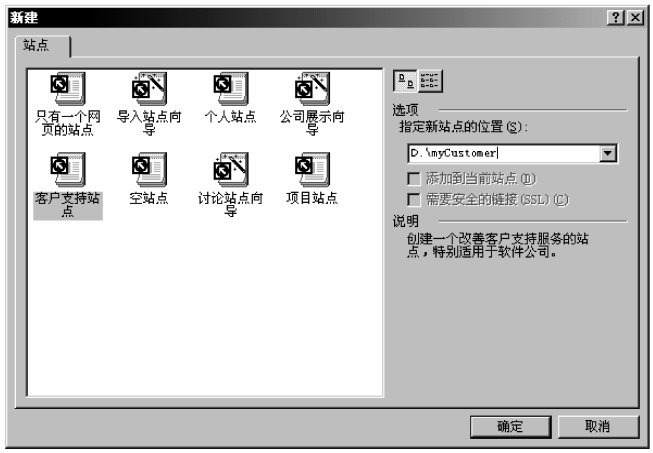


图 4-2　新建用户站点

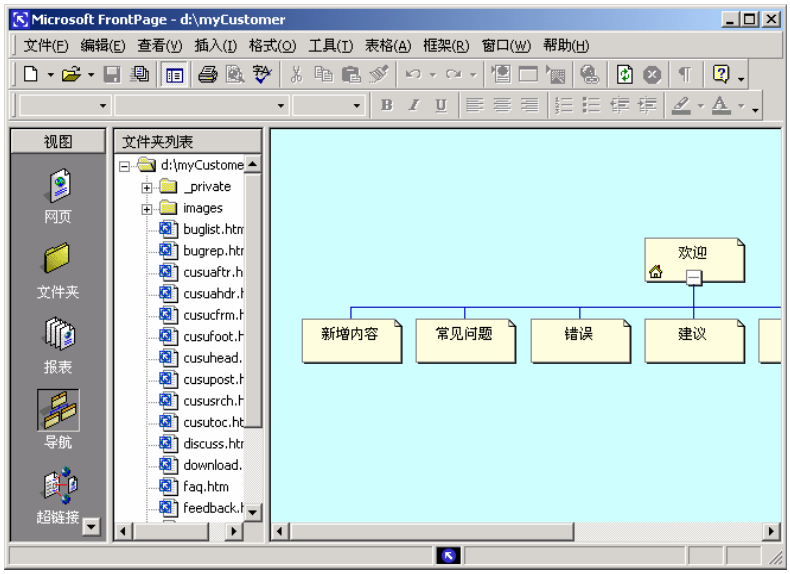


图 4-3　新建站点的文件夹

2．发布站点

所谓“发布站点”，就是将站点所包含的文件复制到指定的目标位置，该位置可以是一台 Web 服务器，或者是一个目标文件夹，甚至是一个可擦写的光盘驱动器。具体步骤如下：

(1) 选择菜单“文件”|“发布站点”，打开“发布站点”对话框，如图 4-4 所示。

(2) 输入要发布的站点的位置，可以是一个文件夹，然后单击“发布”按钮，开始文件复制，最后会显示一个 FrontPage 对话框，显示“成功发布站点！”，则当前 Web 站点所包含的文件已被传送到目标位置。

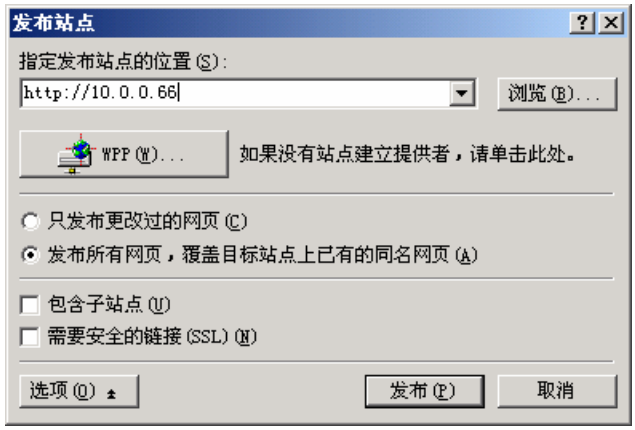


图 4-4 “发布站点”对话框

需要说明的是，当用户运行 FrontPage 时，FrontPage 自动建立一个名为 new_page1.htm 的网页文件，并进入页面工作模式。如果要对站点进行操作，需要选择“文件”|“打开站点…”菜单项，接下来就可以对打开的站点进行操作了，比如站点重命名、导入站点、删除站点等操作。

4.2 新 建 网 页

在 FrontPage 中新建网页，可以单击常用工具栏的“新建”按钮，自动创建一个空白网页，或者选择“文件”|“新建”|“网页”菜单项，打开“新建”网页对话框，如图 4-5 所示。

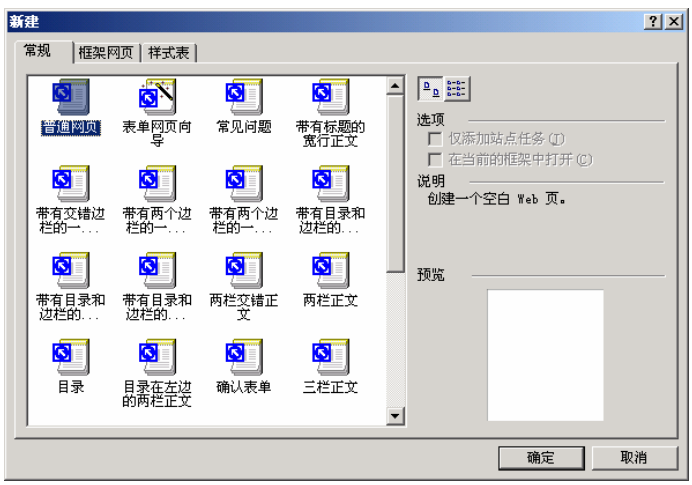


图 4-5 用模板建立网页

FrontPage 提供了许多网页模板，可以用这些模板来建立相应的网页。在右下角的预览区域可以看到选中模板的外观。

另外，FrontPage 打开后将自动新建一个默认网页名为“new_page_1.htm”的新网页文件，用户可以直接编辑网页的内容。新建文件将生成一个 HTML 文档内容的基本框架，在“HTML”显示模式下，可以看到相应的 HTML 内容，如图 4-6 所示。

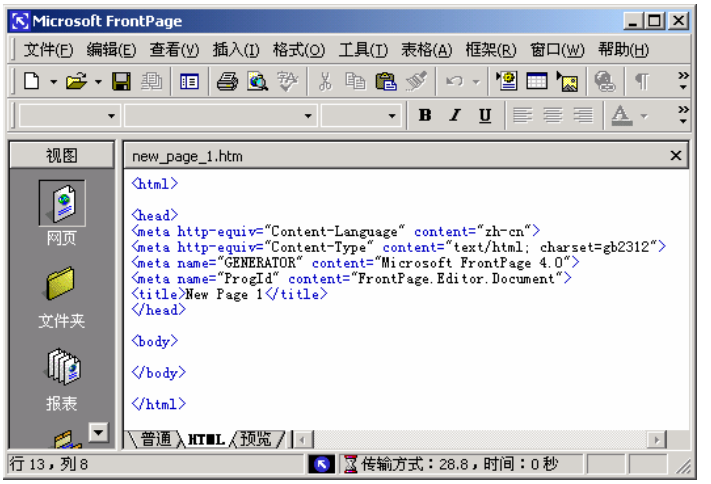


图 4-6　Web 页基本的 HTML 代码框架

用户也可以看到，在上述代码文档的<body>…</body>标记之间还没有内容，具体内容用户可根据需要来设计。

如果要保存网页，选择“文件”|“保存文件”菜单项，或者直接单击“常用”工具栏中的“保存文件”按钮。如果是第一次存盘，则打开“另存为”对话框，根据文档内容选择存储类型，如.htm、.asp、.css 等。

4.3　网页的编辑

网页文件创建后，接下来就是输入内容并进行编辑了。实际上也就是编辑 HTML 文件的内容。HTML 文档是一个纯文本文件，前面曾使用 Windows 的“记事本”程序进行 HTML 文档的编辑，效率很低且容易出错。使用 FrontPage 编辑网页，实际上就是利用 FrontPage 的可视化，在 HTML 文件中输入文本、图片，建立超链接，插入表格、表单等。

4.3.1　输入文本内容

文字是 Web 页的主要内容，在 HTML 文档中有很多标记，如<p>、<font>等用来对文本格式化，即设置文本在浏览器中的显示效果。

在 FrontPage 中，文本的输入和格式化非常简单。在普通模式下，在编辑区域进行输入，再通过“格式”工具栏对文本进行格式化，或者通过“超链接”工具按钮给文本加上超链接(即<a>…标记)。

在普通模式下输入一行文本，并进行简单的格式化，如图 4-7 所示。

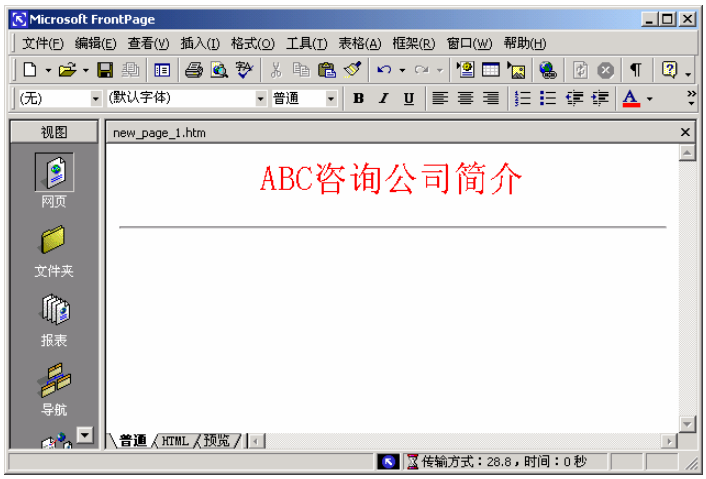


图 4-7 文本的编辑与格式化

转到 HTML 显示模式下，可以看到在<body>…</body>标记内增加了如下代码：

```
<p align="center"><font color="#FF0000" size="6">ABC 咨询公司简介</font></p>
<hr>
```

这段代码是 FrontPage 根据用户的输入自动生成的，可见通过 FrontPage 进行网页设计要比用“记事本”程序方便和快捷得多。这正是 FrontPage、Dreamweaver 等工具的优势所在。

4.3.2 插入图片

1．插入图片

在网页制作中经常需要插入一些图片，以达到丰富信息内容和美化页面的双重功能。在网页内可以插入多种格式的图片，经常采用的是 JPG 格式和 GIF 格式图片，这两种都是压缩的图片格式。特别是 GIF 格式图片，它不但具有很高的压缩比，而且还可以制作成动画图片，以增强网页的动感效果，是目前很流行的图片格式。

在 FrontPage 中，图片插入步骤如下：

(1) 将插入点定位到要插入图片的地方(本例定位到文字的开始)。

(2) 选择菜单“插入”|“图片”|“来自文件”，打开“图片”对话框。在该对话框中将列出当前站点 images 文件夹中所有的图片文件。如果要插入的文件已经在该文件夹中，则只要选中该文件，单击“确定”按钮即可完成插入过程，如图 4-8 所示。

(3) 当要插入的文件不在 images 目录中时，可以单击按钮，打开“选择文件”对话框，设置文件路径、选择文件，最后单击“确定”按钮，完成图片插入过程。

新图片自动以衬于文字下方的形式插入在当前位置，如图 4-9 所示。

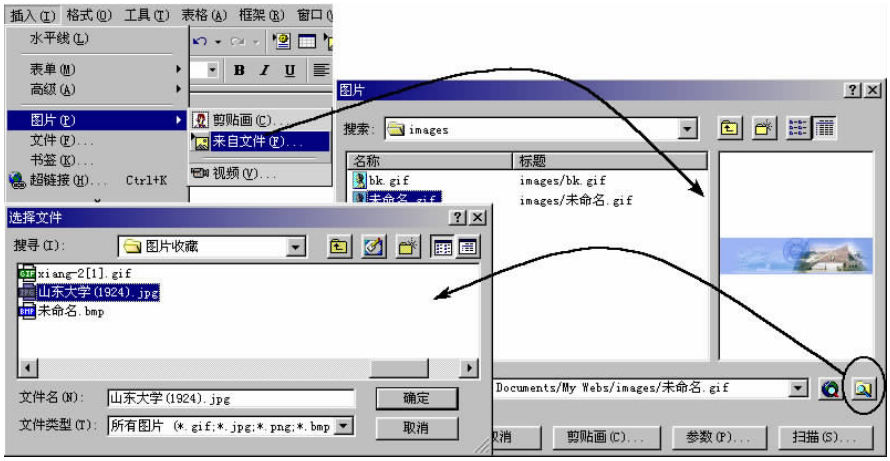


图 4-8　图片插入过程示意图

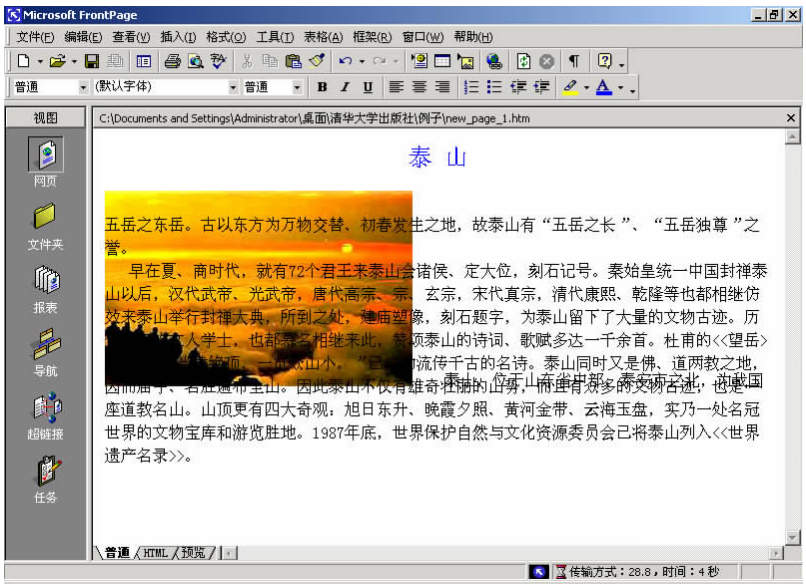


图 4-9　新插入在网页上的图片

由于图片的高度和文字不一致，会看到文字出现重叠，设置了图片的对齐方式后，问题就解决了。

转到 HTML **显示模式，可以看到在**<body>…</body>**标记内增加了一行如下形式的代码：**

```
<img src="file:///c:/.../t01.jpg">
```

这是图片的绝对路径，Web **应用开发中需要避免这种情况。在网页设计时可能不会有问题，但是一旦改变目录，就会出现找不到图片文件的错误，导致图片在浏览器中不能正常显示。**

为什么会产生上述问题呢？因为如果是在一个新建的网页中插入图片，因为当前网页

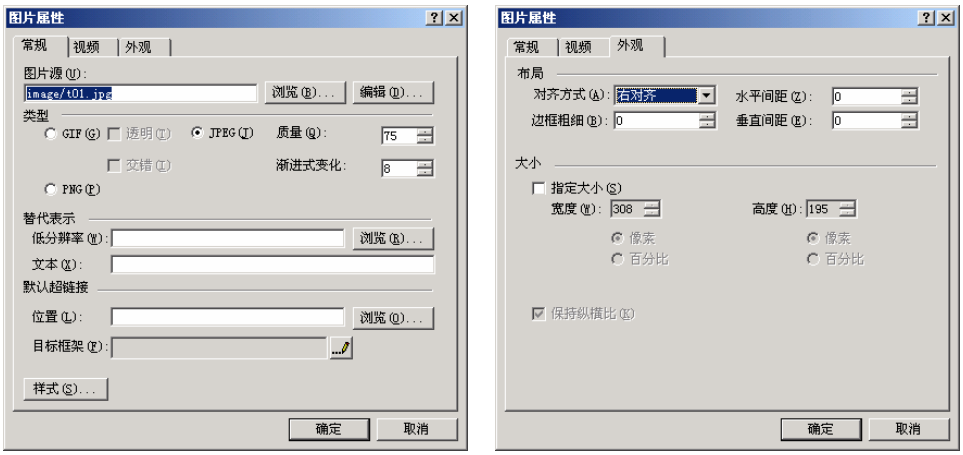
尚未保存，因此无法判断和图片之间的相对路径，只能采用绝对路径的方式插入。如果先将新建的网页保存，再插入图片时，在 HTML 代码中就会使用相对路径。或者，即使不是先插入了图片，只要保存一次 HTML 文件，原先的绝对路径也会变成相对路径。但是，如果 HTML 文件和图片不在一个驱动器上，将不能转换成相对路径。

最后要记住，既然是一个 Web 应用，所有的网页和用到的图片以及其他各种文件都应该保存在 Web 站点的主目录下，或者主目录下面的子目录下。所有这些文件共同构成了一个 Web 应用，所有的引用都应该是相对路径，这样，当应用 Web 即该站点被复制到其他的目标位置时，才不会出现找不到网页或图片的错误。

2．设置图片属性

图片插入以后在页面上的大小、位置等并不一定理想(见图 4-9)。要使得图片符合用户的要求，则必须对它的属性进行设置。

先选定新插入的图片，单击右键，从快捷菜单中选择“ 图片属性 ”(也可以通过“ 格式 ”菜单实现)，打开“ 图片属性 ”对话框，如图 4-10 所示。



(a)“ 常规 ”选项卡　　(b)“ 外观 ”选项卡

图 4-10　“ 图片属性 ”对话框

在“ 图片属性 ”对话框中，可以进行多种属性设置工作。

(1) 图片源：更改图片的路径和文件名。

(2) 类型：确定图片的类型，一般使用默认值。

(3) 替代表示：网页显示时如果图片无法下载，所使用的低分辨率替代图片或替代文本。

(4) 默认超链接：可以设置图片的超级链接。

(5) 图片格式：单击“ 样式 ”按钮，再单击“ 格式 ”按钮，可以设置图片的边框、编号方式、环绕样式、定位样式等内容。

在“ 外观 ”选项卡中，可以设置图片的大小和布局。这里设置对齐方式为“ 右对齐 ”，然后单击“ 确定 ”按钮，则前面编辑的网页“ 泰山 ”显示如图 4-11 所示。

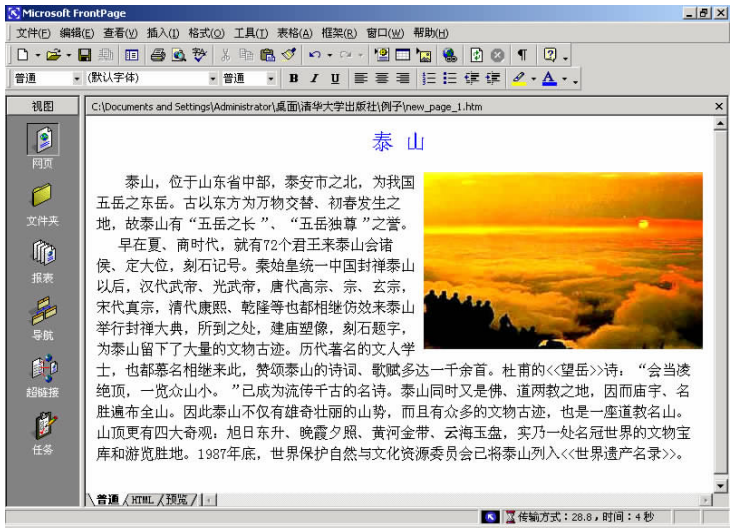


图 4-11　设置属性以后的图片

4.3.3　插入表格

在网页设计中，表格不仅仅用来显示数据，而且还是进行网页布局的重要手段，使用表格，就可对各种网页元素位置进行精确控制，使设计的网页布局整齐、美观。

通过 FrontPage 在网页中加入表格，可以按照下面的步骤操作：

(1) 在网页中加入表格

选择 “ 表格 ” | “ 插入 ” 菜单项，即可插入一个表格。

(2) 表格编辑

插入到网页上的表格，可以进行一系列的编辑操作，如调整行高和列宽，单元格的合并与拆分，删除行、列或单元格，插入行、列或单元格等。

1．表格属性

将插入点放在表格当中，单击右键，从快捷菜单中选择“ 表格属性 ”，打开“ 表格属性 ” 对话框，如图 4-12 所示。

在 “ 表格属性 ” 对话框可进行以下设置。

(1) 布局属性　包括表格布局属性和单元格布局属性。表格布局包括：

① 表格在整个网页中的位置，可以给定具体的位置数值(以像素为单位)，也可以用表格占窗口宽度的百分比确定表格的宽度。

② 对齐方式：表格在页面上的对齐方式。

③ 单元格边距：单元格中文字到表格线的距离。

④ 单元格间距：单元格之间的距离。

(2) 表格边框　可以确定边框的粗细和颜色。

(3) 表格背景　可以确定表格所用的背景色或背景图片。

2．单元格属性

当需要对单元格进行特殊设置时，可以通过“单元格属性”对话框完成。

将插入点放到单元格中，或者选定要设置的单元格，然后单击鼠标右键，从快捷菜单中选择“单元格属性”，打开“单元格属性”对话框，如图 4-13 所示。

图 4-12 “表格属性”对话框

图 4-13 “单元格属性”对话框

可以看到，单元格属性对话框的内容与表格属性对话框很相似，详细介绍略。

4.3.4 超链接

在 FrontPage 中建立超链接非常简单，只需要选定超链接的文本、图像或图像上的一个“热区”，在超链接对话框中输入被链接的 URL 即可完成。

1．文本的超链接

如果为网页上的一段文本建立超链接，可以用以下方法完成：

(1) 选定要建立超链接的文本。

(2) 选择“插入”|“超链接”菜单项，或直接单击常用工具栏上的“超链接”按钮，打开“创建超链接”对话框，如图 4-14 所示。

(3) 用户可以直接在 URL 文本框中输入链接目标的 URL，也可以使用以下四个按钮查找和选择链接目标的 URL。

- 通过 Web 浏览器查找和选择被链接的网页。
- 通过“选择文件”对话框在本地机上查找和选择被链接的文档。同样需要注意，文件和被链接的 HTML 文档之间的路径问题。

如果是同一个 Web 应用中的 Web 页面，必须使用相对路径，方法类似于插入图片。如果是当前 Web 应用以外的 Web 页面，需要给出完整的 URL 地址。

- 为一个 E-mail 地址建立超链接。
- 新建一个空白文档并且与之建立超链接。

(4) 用户还可以设置目标框架和书签，分别对应<a>标记中的 target 属性和 name 属性。

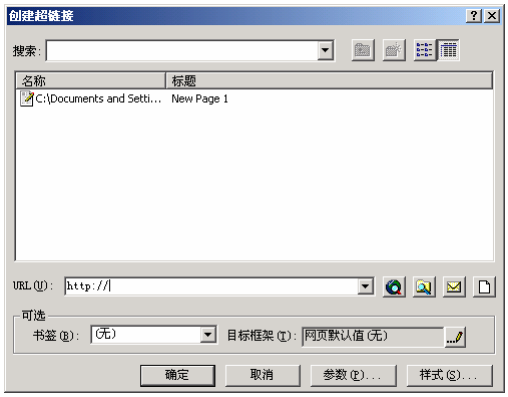


图 4-14　“超链接”对话框

(5) 通过“样式”按钮，可以设置<a>标记的 class 和 ID 属性。

(6) 设置完成后单击“确定”按钮，完成超链接的建立。

当文本包含超链接时，文本将以特殊的颜色显示，超链接文本默认显示为蓝色带下划线的文字。显示网页时，当鼠标指针指向该文本时，鼠标指针将变为手形指针。

2．图像的超链接

为图像建立超链接的方法如下：首先，选中将要建立超链接的图像；然后选择“插入”|“超链接”菜单项，打开“超链接”对话框。接下来的设置与文本的超链接相同。

4.3.5　图像地图

使用 FrontPage 建立图像地图非常容易，具体步骤如下：

(1) 插入图片。

(2) 单击图片，则在 FrontPage 的底部显示图片工具栏，如图 4-15 所示。

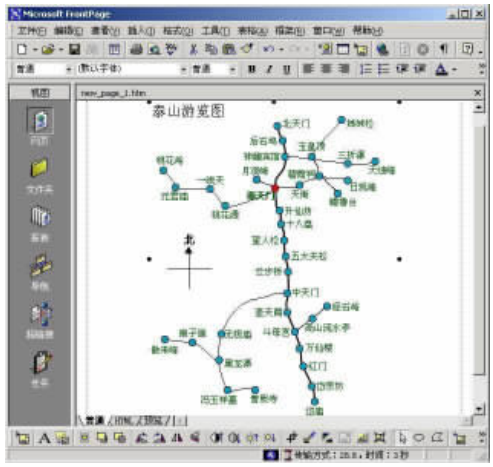


图 4-15　定义图像地图

定义图像的热点可通过“图片”工具栏完成，其中有四个定义热点的工具：长方形热点、圆形热点、多边形热点和突出显示热点。

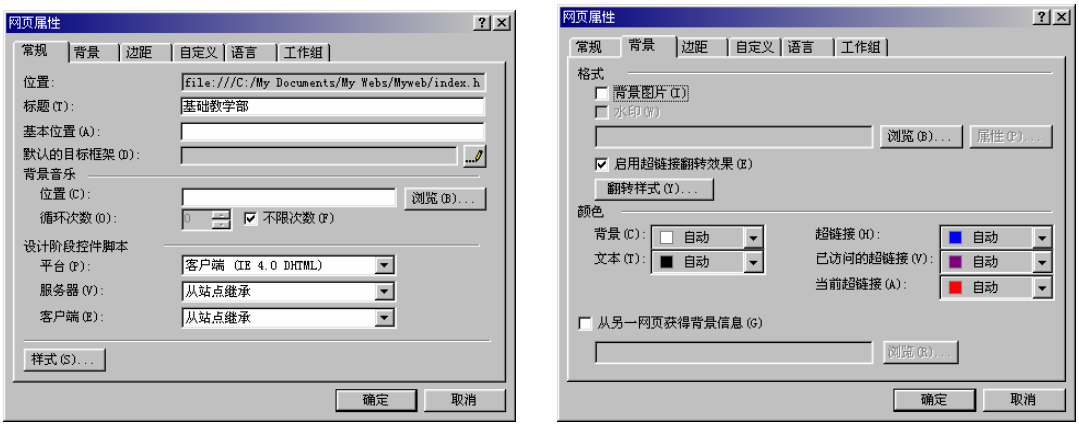
定义热点的方法很简单：选中图像，这时上述的四个工具变为可用；然后按住鼠标左键在图像上的目标区域拖动即可。对于多边形热点，则可以逐个在顶点位置单击，直到形成一个封闭的区域为止。热点区域建立后，将自动打开“超链接”对话框(见图 4-14)，输入要链接的 URL 即可。

网页显示时，图像上的热点区域是不可见的，但是在热点设置时按下“图片”工具栏上的“突出显示热点”按钮，这样在显示网页时，如果单击热点区域，可以看到区域的外框。

4.3.6 网页属性

网页的<body>标记有许多属性设置，这些属性往往影响到整个网页的显示，在 FrontPage 中，大部分的<body>标记属性是通过网页属性设置的。

在网页上单击鼠标右键，从快捷菜单中选择“网页属性...”菜单项，打开“网页属性”对话框，如图 4-16 所示。



(a)“常规”选项卡　　(b)“背景”选项卡

图 4-16　“网页属性”对话框

(1)“常规”选项的设置

- “位置”文本框　可以显示出网页的完整 URL，这一文本框是不能更改的。
- “标题”文本框　可以确定网页的标题。
- “基本位置”　可以输入一个基 URL，基 URL 可用来将网页中的相对 URL 转换为绝对 URL。
- “背景音乐”区域　可以指定网页的背景音乐文件，同时可以设置音乐的循环次数。
- “设计阶段控件脚本”区域　可以设置客户端平台、服务器和客户端的控件脚本。

(2)“背景”选项卡中的设置

- “格式”区域　可以设置背景图片。
- “颜色”区域　可以设置背景颜色、文本颜色以及超链接的颜色等。

如果要从其他网页文档中获得背景信息，可以选中“从另一网页获得背景信息”复选框，然后选择或输入信息来源的网页文件名及路径。

4.4　框 架 网 页

在网页设计中，框架网页是经常使用的一种网页类型，它可以把浏览窗口分成几个区域，每个区域可以显示一个不同的页面。通过框架网页可以在一个窗口中同时显示几个网页。

4.4.1　新建框架网页

在 FrontPage 2000 中创建框架网页，可以通过专门的框架网页模板完成方法如下：

(1) 选择“文件”|“新建”|“网页”菜单项，打开“新建网页”对话框，选择“框架网页”选项卡，如图 4-17 所示。

图 4-17　“新建”对话框中的“框架网页”选项卡

(2) 在左侧的列表框中显示出了多个框架网页模板的图标，单击其中某个图标，可以在右侧的预览区显示出该模板的样式。

(3) 双击选中的模板，或者选中模板后单击“确定”按钮，即可建立起相应的框架网页，图 4-18 就是选中“嵌套式层次结构”后，所建框架网页的初始状态。

在这个框架网页中，浏览窗口被分成了三个框架，各个窗格显示不同的页面。可以看到，初始状态下每个框架中有以下两个按钮：

- 设置初始网页按钮　所谓初始网页，是指浏览器第一次载入框架网页时该框架所示的网页。该按钮就是设置一个已经存在的网页作为该框架的初始网页。单击时可以打开“创建超级链接”对话框，选择一个网页文件或输入一个网页的 URL，然后单击“确定”按钮，即可以在当前框架中打开所超链接的网页文档。

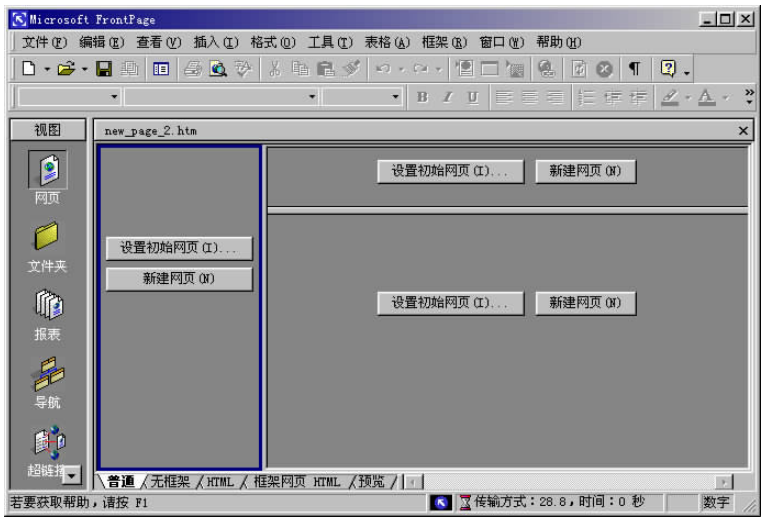


图 4-18 新建的框架网页示例

● 新建网页按钮 单击该按钮时，可以新建一个普通的网页(空白页)作为初始网页。

当新建或打开一个框架网页后，FontPage 窗口下方的显示模式选项标签将发生以下变化：

(1) 增加了“无框架”标签。当浏览器不支持框架网页时所要显示的内容。默认情况下将显示文本“此网页使用了框架，但您的浏览器不支持框架。”。

(2) 在 HTML 标签下，将在各个框架中显示其初始网页的 HTML 代码。

(3) 新增的“框架网页 HTML”标签，用来显示和编辑框架网页的 HTML 代码。

当前网页为框架网页时，“框架”菜单中的菜单项将变为可用，菜单项主要包括“拆分框架”、“删除框架”、“框架属性”等。

4.4.2 拆分与删除框架

为了适应不同的需要，用户可以将一个网页框架随意拆分或合并，形成有自己风格的框架网页。方法如下：

(1) 选中要进行拆分的框架，如在上述的网页示例中选中右下方的框架。选中的框架将用蓝色粗线条包围。

(2) 选择 “框架”|“拆分框架”菜单项，打开“拆分框架”对话框。

(3) 选择“拆分成行”或者“拆分成列”菜单项，即把当前框架窗格拆分上下两个框架，或者拆分成左右两个框架。

如果要删除框架，选中要删除的框架，选择 “框架”|“删除框架”菜单项即可。

4.4.3 改变框架窗格的大小

框架的大小是可以根据需要改变的。方法是：将鼠标指针指向框架，当指针变成↔形或者↕形时，按住左键拖动，到达满意的位置后放开。

4.4.4　设置框架属性

可以用以下方法设置框架属性：鼠标指针指向框架，单击鼠标右键，从快捷菜单中选择“框架属性”，打开“框架属性”对话框，如图 4-19 所示。

图 4-19　“框架属性”对话框

对话框中包含的内容如下。

(1) 名称　框架的名称，对应<frame>标记中 name 属性，主要用于在超链接中指定<a>标记的 target。

(2) 初始网页　第一次显示框架网页时显示的网页。

(3) 框架大小　可以设置列宽和行高。有三种取值：相对(相对于同行或同列的其他框架)、像素和百分比(相对于窗口的高或宽所占的百分比)。

如果选择像素，则在浏览器窗口大小变化的时候，该框架将不会随着变化。

(4) 边距　框架中网页页面的边距，以像素为单位。

(5)“框架网页”按钮　单击此按钮时，可以打开“网页属性”对话框的“框架”选项卡，可以设置框架的宽度，以像素为单位。如果只设置框架的宽度，可以直接打开“网页属性”对话框完成设置。

(6) 可在浏览器中调整大小　选中该复选框时，浏览框架网页时可以改变框架大小(用鼠标拖动框架)。

(7) 显示滚动条　有“在需要时”、“从不”和“始终”三种选择。

(8) 样式　可以修改框架的样式和格式。

4.4.5　设置超链接的目标框架

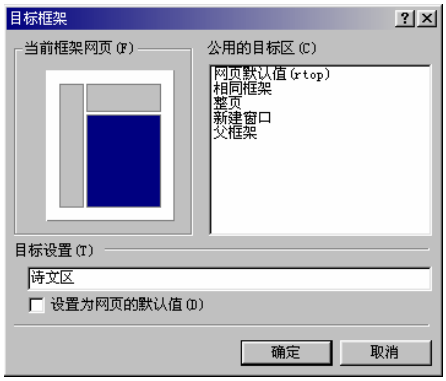


图 4-20　“目标框架”对话框

假设需要设计一个“唐诗欣赏”的框架网页，它包含三个框架：目录区、网页标题区和诗文区。

要求单击目录框架的唐诗标题时，在诗文区可显示出该诗的正文来。下面是具体的实现步骤：

(1) 选中目录区的一首诗的标题“泊秦淮”。

(2) 选择“插入”|“超链接”菜单项，打开“创建超链接”对话框。

(3) 在对话框的下边有一个“目标框架”文本框，单击其后的“…”按钮，打开“目标框架”对话框，如图 4-20 所示。

(4) 在“当前网页框架”图示区域，单击链接网页显示的区域(本例中为右下角区域)，在“目标设置”文本框中自动输入该框架的名称，或从“公用的目标区”选择。

(5) 完成后单击“确定”按钮，回到“创建超链接”对话框中。可以看到，“目标框架”中填入了所选框架的名称。

(6) 选择被链接的 HTML 文档(不再重复)，最后单击“确定”按钮。

按以上方法建立所有诗标题的超链接。完成后的网页如图 4-21 所示。

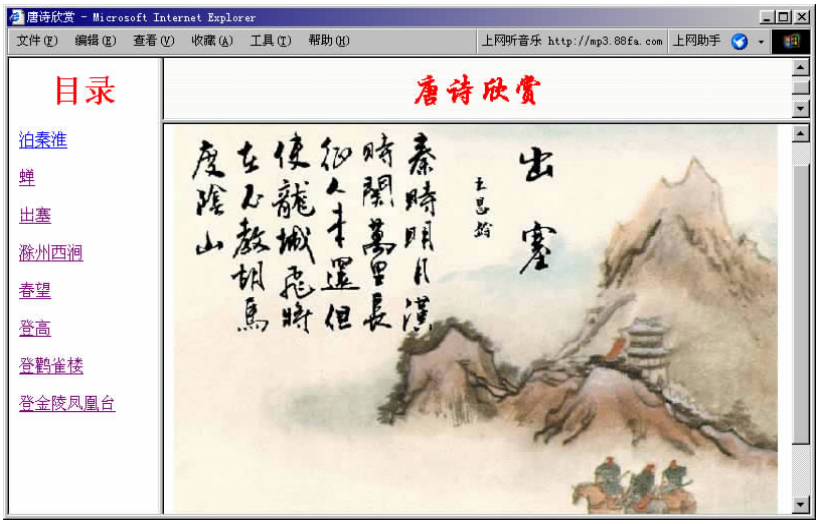


图 4-21 框架网页应用示例

4.5 使用 Dreamweaver

Dreamweaver 是 Macromedia 公司推出的“所见即所得”的网页编辑工具。与 FrontPage 不同，Deamweaver 采用了 Mac 机的浮动面板式的设计风格，对于习惯 FrontPage 的用户一开始可能会感到不适应，相信只需要较短时间习惯其操作方式。

Dreamweaver 最新的版本为 Dreamweaver 4.0 和 Dreamweaver UltraDev 4.0，后者支持 ASP、JSP 等程序的编写与调试。对于一般的网页制作来讲，这两个版本在使用上没有太大的差别。

本节将重点介绍 Dreamweaver 4.0 中一些独特的功能，对于一些 FrontPage 中已经有的普通功能将不做介绍。

4.5.1 认识 Dreamweaver

打开 Dreamweaver，显示主窗口。第一次启动 Dreamweaver 时，同时还显示几个浮动窗口，如图 4-22 所示为关闭浮动窗口后的 Dreamweaver 主窗口界面。

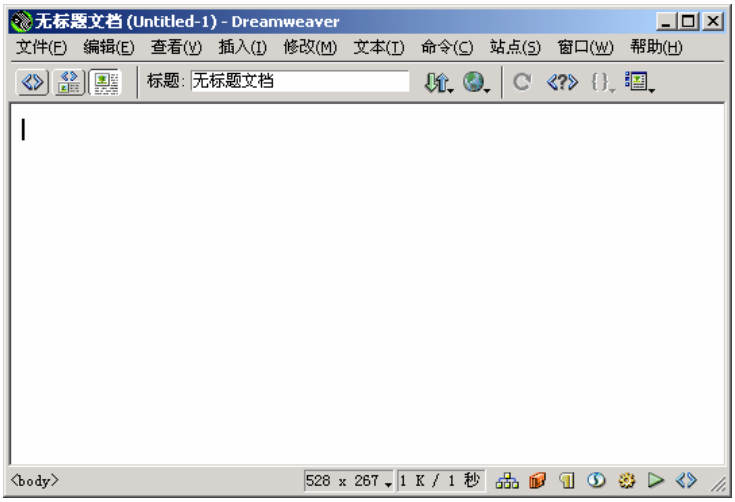


图 4-22　Dreamweaver 主窗口界面

1．浮动窗口

第一次打开 Dreamweaver 时，会自动打开许多浮动窗口，主要包括如下窗口。

(1) 对象面板　在网页编辑的过程中，通过单击对象面板的按钮来为网页添加相应的元素，如图片、表格、层、Flash。这些元素称为对象。单击对象面板上的向下箭头，能插入其他类型的对象，如特殊字符(characters)、表单(froms)等。

(2) 属性面板　用于显示所选中的网页元素的属性，并可在属性面板上修改。选择不同的网页元素，属性面板所显示的内容也有所不同。此外，单击属性面板右下角的小三角可以根据使用的需要，缩小或展开属性面板。

如果希望关闭这些窗口，可以在 Dreamweaver 的主菜单中，打天“窗口”子菜单，可以看到有一些复选菜单项被选中，通过后面的快捷键，可以关闭或打开相应的浮动面板窗口。

如果屏幕上的浮动面板位置过于凌乱，甚至超出了桌面范围而不便操的时，可以进入主菜单中的“窗口”子菜单，选择“排列面板”来自动重排浮动面板。

另外，需要记住那些用来快速打开和关闭经常使用的浮动面板的热键，主要的几个浮动面板热键是属性(Properties)面板(Crtl+F3)、样式(CSS Styles)面板(Shift+F11)、行为(Behaviors)面板(Shift+F3)、对象(Objects)面板(Crtl+F2)。

2．启动器和快速启动栏

在主菜单的“窗口”中，有“启动器”命令，同时在状态栏的右侧有一个“快速启动栏”(又称启动面板)，可用于显示或隐藏相应的浮动面板。启动器与右下角的微型启动栏一一对应。其中，单击按钮可切换到站点管理器，单击按钮可切换到行为面板。

3．视图

Dreamweaver 有三种视图，分别对应于工具条最左边的三个按钮，三种视图分别是：

(1) 按钮　对应显示代码视图，在客户区显示网页对应的 HTML 代码。

(2) 按钮　对应显示代码和设计视图，在客户区水平分割成两个窗格，分别显示网页设计内容和对应的代码。

(3) 按钮　对应显示设计视图，在该视图下，用户可以进行可视化的网页设计，此时往往需要通过“窗口”菜单，分别选择“对象”菜单项和“属性”菜单项，打开“对象”窗口和“属性”窗口。然后在网页中，输入文字、插入图像、表格、表单、框架等内容。

4．站点窗口

在 Dreamweaver 主菜单中有一个“站点”子菜单，用来实现站点的管理。在“站点”菜单下，选择“站点文件”菜单项，打开“站点”窗口，如图 4-23 所示。

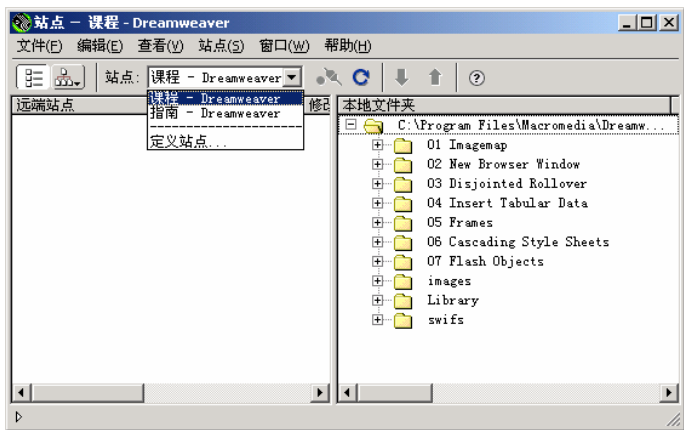


图 4-23　“站点”窗口

站点窗口是 Dreamweaver 另一个重要的窗口，左半部是远程站点的目录，一般显示为空，只有在 FTP 连通状态下才有显示内容；右半部是当前编辑中的本地目录。站点窗口的作用是使用户直观而方便地像管理硬盘里的文件一样管理站点。

4.5.2　定义新网站

在 Dreamweaver 的主菜单中，选择“站点”|“新建站点…”菜单项，或者在站点窗口中(见图 4-23)，在“站点”后面的下拉列表中选择“定义站点…”，打开“站点定义”窗口，如图 4-24 所示。

在“站点定义”窗口中进行如下设置。

(1) 站点名称　为网站起的一个名字，这个名字只起识别作用，与网站发布后真实的域名无关，如本例中给出的站点名称是 E-learning。

(2) 本地根目录　设置网站在本地硬盘的位置，单击后面的文件夹图标可以选择硬盘

的任意目录作为存放网站文件的目录。

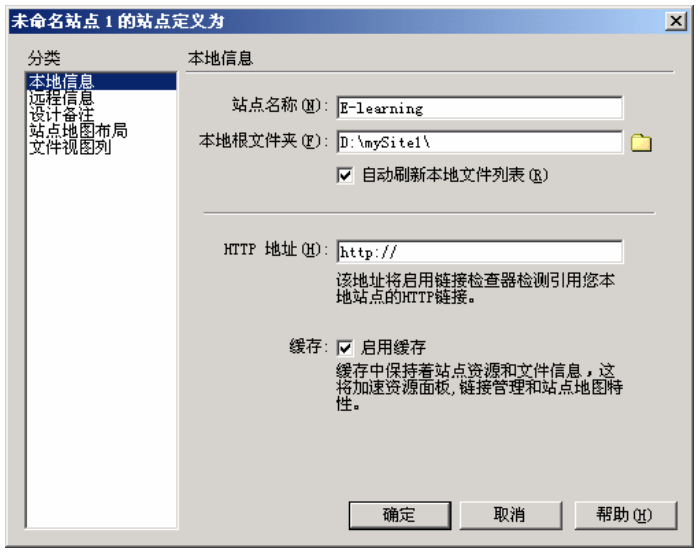


图 4-24　“站点定义”窗口

(3) 缓存　建立缓存可以使文件的移动、改名、查找等站点管理的操作速度大大提高。完成上面的设置后，返回站点窗口，如图 4-25 所示。

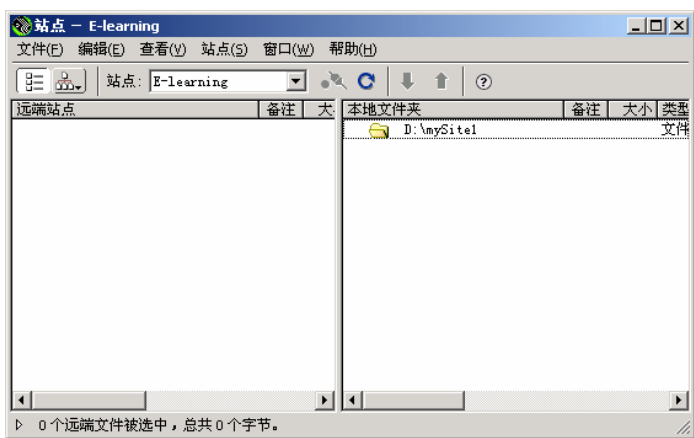


图 4-25　创建新站点后的站点窗口

站点定义完毕之后，站点主目录下只有一个_notes 文件夹，用户可以打开站点窗口的“文件”子菜单，选择“新建文件”或“新建文件夹”菜单项来创建站点的文件夹或网页文件。

4.5.3　制作网页

当站点创建完成后，就可以创建网页了。要在一个站点中创建网页，在站点窗口中，

首先选择“文件”|“新建文件”菜单项。

或者，在站点窗口右侧的“本地文件夹”窗格中，在某个文件夹上右击来创建文件。

例如，在网站根目录下创建一个文件 index.htm。该网页文件创建后，在站点窗口中，双击一个网页文件，将在 Dreamweaver 主窗口中打开该文件。

1．文本及格式化

选择“显示设计视图”，在客户区输入相应的内容，例如输入“欢迎光临 E-learning 网站”，通过“文本”菜单，进行简单的格式化，如图 4-26 所示。

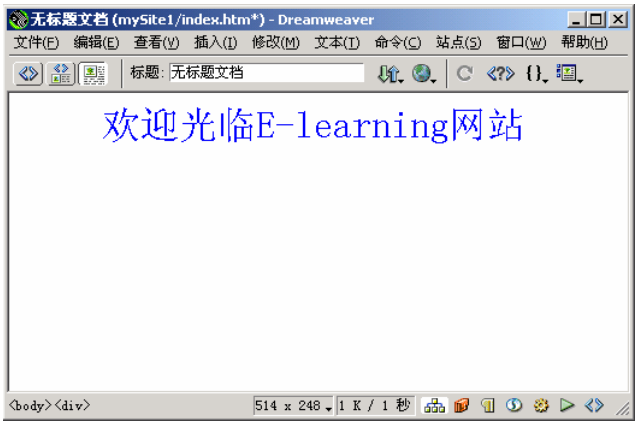


图 4-26 网页设计界面

对文字格式的设置，也可以通过属性面板完成。首先选择要设置的文字，然后打开属性面板，属性面板将显示有关的设置。

2．插入图片及属性设置

如果对象面板没有打开，在 Dreamweaver 主窗口中选择“窗口”|“对象”菜单项，或者按 Ctrl+F2 组合键，打开对象面板。单击对象面板上的 按钮，弹出对话框，选择需要插入的的图片。

图 4-27 选择插入图片

如果被选择的文件是网站主目录以外的文件，系统将提示是否将图片复制到网站内，提示对话框如图 4-28 所示。

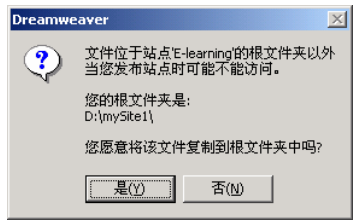


图 4-28　提示窗口

单击“是”，系统打开一个对话框，用户可以选择一个站点内的文件夹来存储该图片。

如果图片不复制到网站内，则网页传上服务器后，图片就不会被显示，因为图片不在该网站内。

接下来，可以对图片进行属性设置。具体步骤如下：

(1) 首先在 Dreamweaver 主菜单中选择“窗口”|“属性”菜单项，打开属性面板。

(2) 在网页中，双击插入的图片，属性面板显示如图 4-29 所示。

图 4-29　属性面板

属性面板显示了当前对象的相关属性，用户可以进行不同的设置，具体设置的含义在此省略。

格式化工作结束后，在“显示代码视图”中，可以看到上述内容的 HTML 代码，如图 4-30 所示。

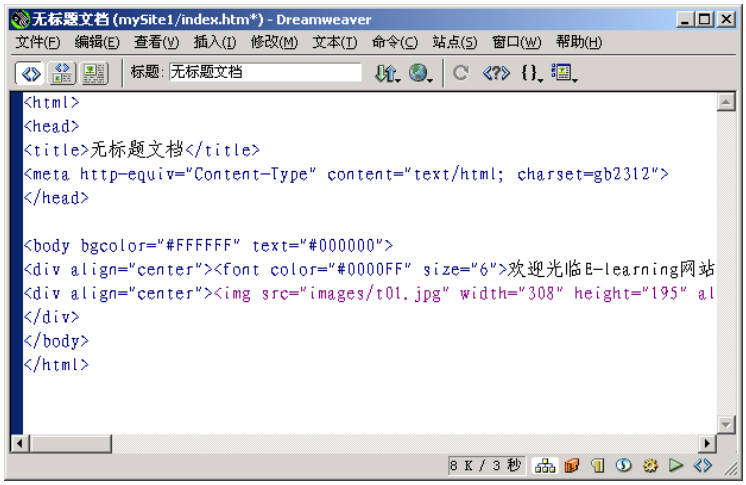


图 4-30　显示代码视图

4.5.4　使用样式表

通过属性面板对文字进行的设置都是最基本的属性设置，如果想使文字的显示更加丰富多彩就需要借助样式表。要使用样式表，可以按照下述步骤操作：

(1) 在编辑状态下按 Shift+F11 组合键，进入样式表对话框(CSS 样式表)，如图 4-31

所示。

(2) 单击右下角的 ，打开“ 新建样式 ”对话框，如图 4-32 所示。

图 4-31 “ CSS 样式 ”表对话框

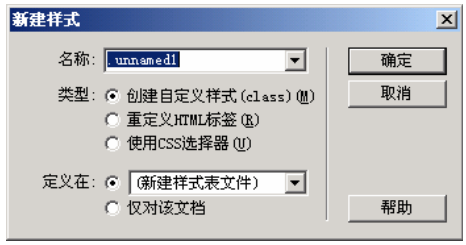


图 4-32 “ 新建样式 ”对话框

在“ 新建样式 ”对话框中，在“ 名称 ”文本框内可为新样式起一个名字，Dreamweaver 默认情况下会在名字前加一个小数点，目的是避免与其他 HTML 标记混淆。“ 定义在 ”后面有两个选择：用于选择用外部样式表还是只在本页建立样式(仅当前文档)，这里选择前者，这样就可以多页共用一个样式表。在本例中输入样式名字为 myStyle1。

设置完成后，单击“ 确定 ”按钮，系统自动打开“ 保存样式表文件为 ”对话框，此时选择站点下的一个文件夹保存样式表，此处选择 pub 文件。接下来进入“ 样式定义 ”对话框，如图 4-33 所示。

图 4-33 “ 样式定义 ”对话框

对样式进行有关的设置，完成后单击“ 确定 ”按钮，返回 CSS 样式对话框(见图 4-31)，则显示一个新的样式，同时在站点窗口中，显示一个 pub.css 文件。

还可以看到，在当前网页的头部增加了使用样式的 HTML 标记：

```
<link rel="stylesheet" href="pub.css" type="text/css">
```

如何将样式应用于文本呢？具体步骤是：

首先选择文本，然后在样式对话框中单击某个样式，该样式则作用于选择的文本；或者选择“ 无 ”，取消使用的样式。包含样式的生成代码如下：

```
<font class="myStyle1">欢迎光临 E-learning 网站</font>
```

4.5.5　使用超链接

超链接包括超文本链接、图像链接和热区链接三种，创建的方法相类似，具体步骤如下：

(1) 首先打开“属性”面板(见图 4-29)。

(2) 选择文字或者图片，在“属性”面板的链接(Link)一项中输入要链接的文档的 URL 地址。

如果要链接到网站内的文件，可以单击图标，打开“选择文件”对话框，如图 4-34 所示。

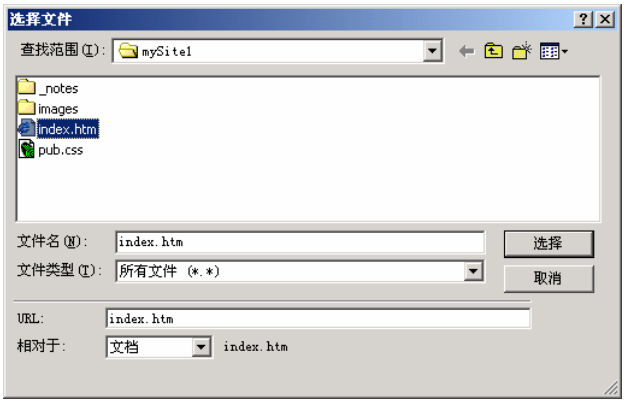


图 4-34　“选择文件”对话框

需要特别注意最下方的“相对于”选项，可以选择相对路径的形式，默认的方式是文档，也可以选择站点根目录。区别在于，文档是指相对路径从本页开始计算，而根目录是指从根目录(即“/”)开始计算。

如果使用站点根目录的模式做成的链接，在本地硬盘直接打开页面浏览可能会出错，但在 Dreamweaver 的预览模式下或上传到 Web 服务器后，则不会有问题。

需要说明的是，如果要链接到一个 E-mail 地址，则在链接后面的文本框中输入“mailto:email 地址”，例如 mailto:webmaster@gsl.sdu.edu.cn。那么，单击该链接的时候，就会打开默认的 E-mail 程序，例如 Outlook，并发送 E-mail 到给定的地址。

另外，如果链接到浏览器无法打开的文件，例如.exe、.zip 等文件，那么浏览者在单击这个链接的时候，就会打开一个对话框，询问文件保存到硬盘的位置，即实现文件下载的功能。

4.5.6　使用表格

网页里通常会包含表格，表格的用途主要是网页排版，可以使网页更美观。在 Dreamweaver 主窗口中，选择“插入”|“表格”菜单项可以插入一个表格。表格的一般性操作和 FrontPage 类似。下面介绍几个技巧。

- 不要试图把整个网页放在一个大的表格里，因为一个大表格里的内容要全部 load 完才会显示，如果整个网页放在一个表格里，那么网页会显示得较慢。
- 动态改变表格的颜色，会使网页更有动感。

例如，在单元格的<td>标记里添加列代码：

```
onMouseOut="this.style.backgroundColor=’’ onMouseOver=this.style.back-
groundColor=’red’
```

把鼠标移到相应的单元格看看效果。

- 用表格代替水平线，插入一个表格，将高度设为 1(按需要设置)，当然也可以将宽设为 1，制作竖线。

需要注意，在 Dreamweaver 里制作时，先将高设为 1 后，切换到代码窗口，将表格里的空格符( )去掉，否则将看不到效果。

- 为了使大量的信息整齐地排列，经常需要使用嵌套的表格。
- 制作有立体感的表格，通过表格属性设置实现一些特殊的显示效果，这些参数包括：

```
border="1"，cellspacing="0"，cellpadding="0"，bgcolor="#9999CC"，border-
color="#FFFFFF"，bordercolorlight="#000000"。
```

通过给它们赋不同的颜色值可达到不同的效果。往往是把 bordercolor 颜色设浅一点，表格就有凸起的效果。

4.5.7 文件预览

在一个网页创作的过程中往往要看一下在浏览器中的显示效果，这就是预览。和 FrontPage 不同，在 Dreameweaver 中，网页的预览是通过 Dreamweaver 主窗口的“文件”|“在浏览器中预览”菜单项来完成的，此时该文档在浏览器中打开。用户可以看是否达到了设计效果，以便下一步修改。

4.5.8 使用层和新的排版功能

层(layer)是 Dreamweaver 中一项非常好用的技术，层往往用于对页面元素的定位。什么是图层呢？通俗地讲，图层就像是含有文字或图形等元素的胶片，一张张按顺序叠放在一起，组合起来形成页面的最终效果。图层可以将页面上的元素精确定位。图层中可以加入文本、图片、表格、插件，也可以在里面再嵌套图层。

图层在平面坐标 x、y 上又添加一个三维空间的坐标 z，使图层有一个叠放顺序。图层的一个重要功能就是可以使页面元素重叠。

单击对象面板的按钮，光标变成一个十字，在编辑窗口上拖动，即可创建出一个层。可以在层上输入绝大部分的网页元素，例如图片、文字和表格等。层可以放置在页面的任何位置，如图 4-35 所示。

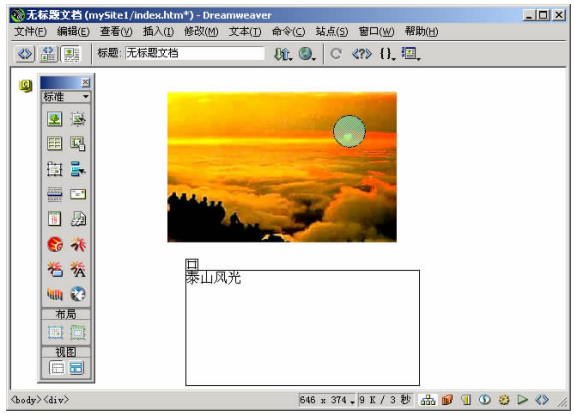


图 4-35　在网页中使用层

插入层后，还需要进行有关的设置。打开属性面板，单击“层”，对应的属性面板如图 4-36 所示。说明如下。

图 4-36　层的属性

- 层编号　层的名称，用于识别不同的层。
- 左(L)，上(T)　层的位置，左是距离页面左边界的距离，上是距离页面上边界的距离。
- 宽(W)，高(H)　层的宽和高。
- Z 轴(Z)　层的 Z 轴顺序。
- 背景图像　层的背景图。
- 显示(V)　层的显示状态，其中的隐藏是将层隐藏，处于不可见状态。
- 背景颜色　设置层的背景颜色。
- 标签　层使用的代码方式，一般使用默认的 DIV 标记即可。
- 溢出　如果层里面的文字太多，或图片太大，层的大小不足以全部显示的时候使用该选项。可以选择 visible(超出的部分正常显示)、hidden(超出的部分隐藏)、scroll(不管是否超出都显示滚动条)或者 auto(有超出时才出现滚动条)。

层的优点很明显，但缺点也同样明显，例如难以制作一个适应不同分辨率的网页；当一个页面使用了多个层后，页面的复杂程度增加而导致编辑起来非常麻烦；编辑状态与浏览状态的实际效果有相当明显的差别等。

通常人们是利用层进行排版，然后将层转换为表格，具体步骤是：

在 Dreamweaver 主菜单中，选择“修改”|“转换”|“层到表格...”菜单项。具体细节请读者自己实验。

下面介绍在 Dream weaver 4 中新推出的 Layout 排版功能，它能更直接方便地对页面布

局进行编排。在对象面板的最下方有四个按钮。

- 标准视图按钮　默认状态下的视图。
- 排版视图按钮　转入排版视图。按下 按钮后，上面布局中的两个按钮即可使用。
- 绘制布局表格按钮　像画图一样在页面里排版表格。
- 布局单元格按钮　在表格中画单元格。

接下来利用以上工具排版一个新页面，具体步骤如下。

新建一个页面，单击按钮进入排版视图，单击按钮 在页面上画出表格，再单击按钮 ，在表格中画出单元格，其中，表格会以绿色表示，单元格会以蓝色表示，未确定的单元格会以灰色表示，如图 4-37 所示。

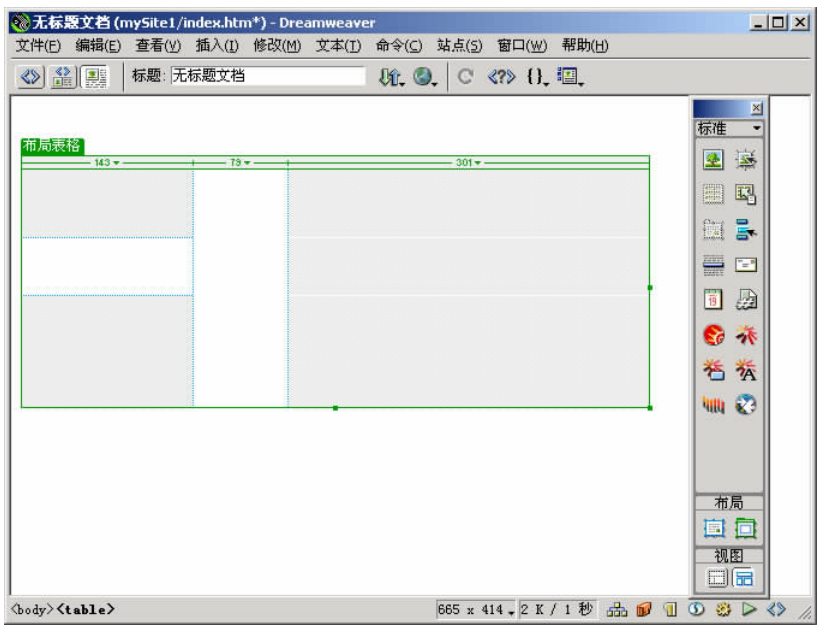


图 4-37　绘制表格

调整完成后，单击按钮，回到标准视图，可以看到刚才画的 Layout Table 已经全部转化为表格了。

4.5.9　制作简单的互动效果

在对象面板中，有一个“插入图标经过图像”按钮，它的作用是当鼠标移到一张图片上时，原来的图片就切换为另一张图片，鼠标移开后，又恢复原来的模样。

具体实现步骤如下：

(1) 准备两张大小相同的图片，一张是鼠标没有移上去时的状态，一张是鼠标移上去后的图片。

(2) 单击对象面板上的 ，打开“插入鼠标经过图像”对话框，如图 4-38 所示。

图 4-38　“插入鼠标经过图像”对话框

对话框内的设置解释如下。

- 图像名称　变换图片效果的标记名称，Dreamweaver 会自动分配。
- 原始图像　指定原始的图片。
- 鼠标经过图像　指定鼠标位于其上方时显示的图片。
- 单击时，转到 URL　图片的链接网址。

设置完毕后，单击“确定”按钮，然后在代码视图中可以看到如下代码：

```
<a href="index.htm" onMouseOut="MM_swapImgRestore()"
   onMouseOver="MM_swapImage('Image1','','images/t02.gif',1)">
   <img name="Image1" border="0" src="images/t01.jpg" width="308"
   height="195">
</a>
```

可以按 F12 键预览设置的效果。

4.5.10　使用行为

在 Dreamweaver 中，更复杂的效果是通过行为(behaviors)功能实现的，“行为”又称为事件的响应。通过网页元素的变化，如鼠标的移动、单击等事件（events)，事件往往对应一个事件的响应(action)。

在窗口菜单中，选择“窗口”|“行为”菜单项，打开“行为”面板，如图 4-39 所示。

在网页加入行为，需要以下三个步骤：

(1) 选取产生行为的元素，例如图片、带链接的文字、层等。如果需要在页面一载入就开始出现效果，可以选择状态栏上的<body>标签。

(2) 单击 + 按钮，打开一个响应列表(不同的元素对应的响应也有所不同)，选择一种响应并在随后的对话框中设置此响应的属性。

图 4-39　“行为”面板

可以选择的响应包括以下内容。

- 播放声音　可以为网页加入声音。
- 打开浏览窗口　可以打开一个小窗口(和网上的弹出窗口一样)。
- 弹出信息　可以弹出一条警告信息。
- 调用 JavaScript　调用网页中包含的 Javascript 程序。

- **对调图像　实际上是交换图像。**
- **更改内容　可以改变已经插入的层的内容。**
- **恢复对调图像　把已经交换的图像恢复过来。**
- **检测插件　可以检测访问者的浏览器是否已安装浏览网页所必需的插件。**
- **检测浏览器　检测访问者使用的是什么类型的浏览器。**
- **检验表单　检验网页中的表单是否合法。**
- **控制** Shockwave **或** Flash　**控制网页中包含的** Shockwave **或** Flash**。**
- **前往** URL　**跳转到其他页面。**
- **设置导航栏的图片　和交换图像差不多。**
- **设置文字：在特定的地方显示文字。**
- **显示或隐藏层　设置图层的显示或隐藏。**
- **时间轴　可以制作更多的动态效果。**
- **跳转菜单　插入跳转导航菜单。**
- **跳转菜单前往　控制导航菜单跳转到哪个页面。**
- **拖动图层　设置图层是否允许拖动。**
- **预载图像　在网页装载前预先载入图像。**
- **显示事件用　设置显示** IE **或** NS **各个版本的事件。**
- **下载更多的行为事件　打开网页，去下载更多的事件。**

在图 4-39 **所示的行为面板中列出了当前元素可以触发的事件，不同的元素可以触发的事件也不相同，通过选择事件，可以决定在什么情况下触发响应。常见的事件有：**

- onMouseOver　**鼠标移到目标上。**
- onMouseUp　**按下鼠标再放开左键时。**
- onMouseOut　**鼠标移开时。**
- onMouseDown　**按下鼠标时(不需要放开左键)。**
- onClick　**单击鼠标时。**
- onDblClick　**双击鼠标时。**
- onLoad　**载入网页时。**
- onUnload　**离开页面时。**
- onResize　**当浏览者改变浏览窗口的大小时。**
- onScroll　**当浏览者拖动滚动条时。**

需要说明的是，当找不到所需要的事件时，单击  **按钮，在列表中选择“显示事件”，选择更高版本浏览器的事件。**

[**例** 4-1]　**浮动提示效果的实现。**

根据上面的介绍，通过层来实现网页的浮动提示效果。具体步骤如下。

(1**) 首先插入一个两行三列的表格，第二行的三个单元格合并起来，边框为** 0**，并填充文字，为文字加上相应的链接，第二行的单元格插上一个图层(命名为** good**)。如图** 4-40 **所示。**

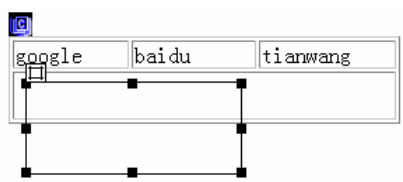


图 4-40　**插入层**

(2) 选取 google，按 Shift+F3 组合键，打开行为面板，单击行为面板上的+，选取“设置文字”|“设置图层文字”菜单项，出现如下对话框(在“新建 HTML”文本框里填上需要的文字)，然后单击“确定”按钮。如图 4-41 所示。

(3) 再单击一次+，重复刚才的步骤，不过这次“新建 HTML”文本框内不填任何文字，然后单击“确定”按钮。行为面板变为如图 4-42 所示。

(4) 单击 onMouseover 旁边的▼按钮，把第二次加入的行为 onMouseover 改为 onMouseout，如果找不到，则显示更多事件，再选其他版本浏览器，这样就有 onMouseout 了。

(5) 对于另外两个搜索引擎名字，操作和上面的步骤一样，只不过加入的文字不同。

把网页保存下来，按 F2 键看看效果。可以看到，当鼠标移到 google 文字上的时候，显示文本“www.google.com 搜索引擎”，即实现了浮动提示效果。

最后，在 Dreamweaver 中，选择“显示代码视图”，可以看到比较复杂的 HTML 代码。

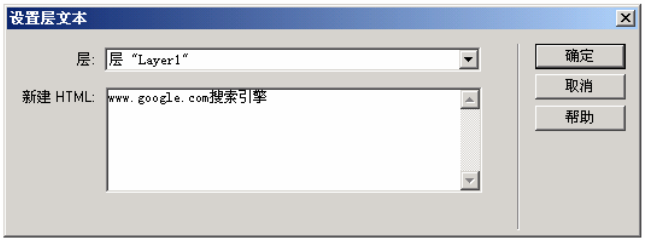


图 4-41　设置层文本

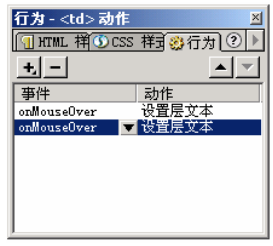


图 4-42　行为面板

4.6　Photoshop 和图像处理

Photoshop 是一个非常优秀的图像处理软件，是 Adobe 公司的产品，具有强大的图像处理功能，在传统的印刷业、平面广告业中占有相当重要的地位。随着近年来网络的发展，网络图像处理激发了一些新的软件，如 Fireworks、Photo Impact 等。为了提高自身的市场竞争力，Photoshop 也不断增强网络图像处理功能，目前使用较多的版本是 Photoshop 6。

Photoshop 的功能很多，在本节将对其做一个概括性的介绍，目的是通过本节的学习，让大家能够使用 Phtotoshop 完成一些基本的图像处理任务。对于一些具体的细节和使用技巧，只能靠在工作中学习和积累，最终达到熟练掌握和使用 Photoshop 的目标。

4.6.1　图像处理基础知识

静止的图片称为图像。图像可以通过扫描设备、数码相机等设备获得。为了适应不同应用的需要，图像往往以不同的格式进行存储。同时为了获得清晰的图像，还需要对原始的图像进行处理，以满足不同的要求。

1．图形与图像

图形是指由外部轮廓线条构成的图，即由计算机绘制的直线、圆、矩形、曲线、图表

等，分为位图和矢量图两种。

图像是由扫描仪、摄像机等输入设备捕捉实际的画面产生的数字图像，是由像素点阵构成的位图。用数字任意描述像素点、强度和颜色。

2．什么是图像处理

图像处理是一种包含图像编码、图像压缩、图像增强、图像复原、图像识别和图像理解等诸多分支的综合技术。广泛地应用于太空图像处理、遥感图像、天气预报、医学图像等各个的领域。美国人利用图像处理技术把从太空探索中拍到的大量不清楚的月球照片进行处理，使得照片中许多重要信息得以清晰再现，引起巨大轰动。

图像处理技术涉及的学科很多，例如图像编码的理论基础是信息论和抽象数学；图像识别需要随机过程信号处理知识。另外，不少课题还需要小波变换、傅立叶变换、神经网络、分形等更加专门的数学知识。

另外，在计算机领域，计算机图形学、模式识别、人工智能、计算机视觉等都与图像处理技术有密切的联系。

3．常用的图像格式

数字图像往往根据应用的不同保存为一些特定的格式。Windows 中的图像以.bmp 或.dib 格式存储。另外还有很多图像文件格式，如.pcx，.pic，.tif，.gif，.tga 和.jpg 等。下面对这些不同的图像文件格式做简要介绍。

(1) PCX 格式

PCX 由 PC Paintbrush 得名，它由 Z-Soft 公司设计，随该公司的图形图像编辑软件 PC Paintbrash 一起发布，故也经常叫做 Z-Soft PCX 图像文件格式，后来微软公司将 PC Paintbrash 移植到 Windows 中，PCX 格式的图像文件得到了广泛的应用。最初的 PCX 格式只支持 16 色图像，现在则能表示 256 色甚至真彩色图像。PCX 是微机上使用最广泛的图像文件格式之一，绝大多数图像编辑软件均能处理这种格式。另外，由各种扫描仪扫描得到的图像几乎都能存成 PCX 格式的文件。

(2) BMP 格式

BMP(Windows Bitmap)是 Windows 图形图像的基本位图格式，是微软公司专为 Windows 环境应用图像而设计的，Windows 软件的图像资源多以 BMP 格式存储。大多数图形图像软件，特别是在 Windows 环境下运行的软件，都支持这种文件格式。BMP 文件有压缩和非压缩之分，一般作为图像资源使用的 BMP 文件都是不压缩的。BMP 支持黑白图像、16 色和 256 色的伪彩色图像以及 RGB 真彩色图像。

(3) GIF 格式

GIF 格式的全称是 Graphics Interchange Format(图形交换文件格式)，是由 CompuServe 公司开发的压缩图像存储格式，支持黑白图像及 16 色和 256 色的彩色图像。GIF 格式的图像文件可以在不同的平台上交流和传输，是世界通用的图像格式。GIF 格式的文件压缩比较高，文件较小。

(4) TIF 格式

TIF(Tagged Image File Format)格式是由 Aldus 和 Microsoft 合作开发的一种通用的位映

射图像文件格式，支持从单色模式到 32 位真彩色的所有图像类型。TIF 格式不针对某一特定的操作平台，可用于多种操作系统及不同机型。TIF 格式文件有多种压缩方式，解压缩比较复杂。

(5) JPG 和 PIC 格式

JPG(Joint Photographic Experts Group)和 PIC 格式原是 AppleMac 机器上使用的一种图像格式，近年来在 PC 机上也十分流行。这两种格式的最大特点是文件非常小，而且可以调整压缩比。JPG 文件的显示比较慢，图像的边缘有不太明显的失真。JPG 的压缩比很高，常用于处理大量图像的场合。它是一种有损压缩的静态图像文件存储格式，压缩比可以选择，支持灰度图像、RGB 真彩色图像和 CMYK 真彩色图像。

(6) PNG 格式

PNG(Portable Network Graphic)格式是 Web 图像中最通用的格式。它是一种无损压缩格式，但是如果没有插件支持，有的浏览器可能不支持这种格式。PNG 格式最多可以支持 32 位颜色，但是不支持动画图。

4．图像的类型、大小和分辨率

(1) 图像的类型

计算机图像分为位图图像和矢量图像两类，它是以数字的方式来记录、处理和保存的，因此又称之为数字图像。

位图图像是以小方型网格(栅格)即像素来代表图像的，每个像素都被分配一个特定位置和颜色值。位图图像与分辨率有关，当以较大的放大倍数显示，或以过低的分辨率打印时，会出现锯齿边缘且会遗漏细节。位图图像在缩放和旋转时会产生失真。但由于它能记录每个点的数据信息，可以精确地记录下色调丰富的图像，尤其在表现阴影和色彩的细微变化方面，位图图像是最佳选择。常用 Photoshop、Corel PhotoPaint 等图像处理软件来处理位图图像。

矢量图像是以数学上的矢量方式来记录图像内容的，它以线条和色块为主，根据图形的几何特性对其进行描述。矢量图像与分辨率无关，缩放到任意大小或在任意分辨率的输出设备上打印出来都不会失真，可以制作 3D 图像，但色调不够丰富。常用 Illustrator、CorelDraw、AutoCAD 等来生成和处理矢量图像。

(2) 图像的大小

图像的大小是指图像的高度和宽度，它有像素(pixel)、英寸(inch)、厘米(cm)等度量单位。屏幕上显示的尺寸是由图像本身的像素个数和显示器的特性决定的。一般常用显示器的像素个数为 640×480、800×600、1024×768 等。在 Photoshop 中，图像像素是直接转换为显示器像素的，当图像分辨率高于显示器的分辨率时，图像将显示得比指定的尺寸大。图像在显示器上的尺寸与打印尺寸无关。

(3) 分辨率

分辨率是指单位长度内所含有的像素个数，分为如下几种。

- 图像分辨率　是指每平方英寸含有多少个像素，单位为点/平方英寸，通常写作 ppi，如 32ppi 的图像每平方英寸含有 32(宽)×32(高)个像素；同样，300ppi 的图像每平方英寸含有 300×300=90000 个像素。分辨率越高，单位尺寸内的像素数越

多，图像就越逼真。

图像的分辨率、图像尺寸的大小和图像文件的大小三者之间有密切的关系。同样尺寸的图像，分辨率越大，单位尺寸内的像素数就越多，文件也越大。

- 显示器分辨率　即显示器上每英寸显示的点数或像素个数(单位是点/英寸，通常写作 dpi)，它取决于显示器的类型、大小及其显示设置。
- 打印机分辨率　即打印机产生的每英寸的油墨点数(单位是点/英寸，通常写作 dpi)，最佳的打印效果应是使用与打印机分辨率成正比的图像分辨率。大多数激光打印机的输出分辨率为 300~1200 dpi 或更高。
- 位分辨率。也叫位深，用来衡量每个像素存储的信息位元数，它决定每个像素存放多少颜色信息。如一个 24 位的 RGB 图像，每个像素都要记录 R、G、B 三原色的值，所存储的位元数为 24 位。

5．色彩模式和模型

色彩模式是指在显示或打印图像时定义颜色的不同方式。常见的模型有 HSB(基于人类对颜色的感觉，表示色相、饱和度、亮度)、RGB(可描述绝大部分的可见光，表示红、绿、蓝)、CMYK(以打印在纸张上油墨的光线吸收特性为基础，部分光谱被吸收，表示青、洋红、黄、黑，使用 K 是为了避免与蓝色混淆)等。色彩模式主要包括 RGB 模式、CMYK 模式、HSB 模式、Lab 模式、灰度模式、位图模式和多通道模式等。

RGB 色彩模式常称为真彩色模式，它给每个像素的 RGB 分量分配一个从 0(黑色)到 255(白色)范围的强度值。当三分量的值相等时显示灰色；都为 0 时为纯黑色；都为 255 时显示纯白色。该模式下每个像素用 3 byte(24 bit)来表示，可表示 1670 余万种颜色，是屏幕显示的最佳模式。

CMYK 模式是四通道图像，包含 32 位/像素，每个像素每种印刷油墨会被分配一个百分比值，最亮颜色分配较低的百分比值，较暗颜色分配较高的百分比值。当四种分量的值都是 0 时，产生纯白色。将 RGB 图像转换成 CMYK 就会产生分色，用印刷色打印制作的图像时，使用 CMYK 模式效果极佳。

6．色调、色相、饱和度和对比度

色调，主要是指彩色画面中色彩的基调，由色彩的明暗和色别所组成。具体讲色调是图形原色的明暗度，如 RGB 图像的原色为 R(Red)、G(Green)、B(Blue)三种。色调的调整也就是明暗度的调整，其范围为 0~255，包括 256 种色调。如灰度模式就是将白色到黑色间连续划分为 256 个色调；RGB 模式中将红色加深就成了深红色。在画面上起着主要作用，或在量上占有相当比重的色彩称为主色调。

色相就是色彩的颜色，如红、橙、黄、绿、青、蓝、紫，每种颜色即代表一种色相，调整色相即是调整图像的颜色。

饱和度是指图像颜色的彩色程度，若饱和度为 0，则彩色图像变成灰色图像。增加饱和度就会增加其彩色程度。

对比度是指不同颜色之间的差异。对比度越大，两种颜色之间的差异越大。如增加一幅灰度图像的对比度后，会使其黑白更加分明，继续增加则变成了一幅黑白图像。

4.6.2　启动 Photoshop 6

将 Photoshop 6 安装到计算机中后，在 Windows 的“开始”菜单中将增加 Adobe 程序组，其中包含 Photoshop 6.0，选择其中的 Adobe Photoshop 6.0 菜单项就可以启动 Photoshop 6。启动 Photoshop 6 后的主界面如图 4-43 所示。

图 4-43　Photoshop 6 主界面

Photoshop 6 的界面和 Dreamweaver 类似，工具栏和活动面板不是依附于主窗口，而是可以单独停靠，从而有效地节省客户工作区空间。对比一下 Windows 的“画图”程序中的“工具栏”和“颜料盒”功能设计，只能显示和隐藏，不能单独地定位和最小化。

1．选项栏

选项栏是在 Photoshop 6 中才出现的，在 Photoshop 5.5 以前，双击工具后会出现其相应的属性活动面板。而在 Photoshop 6 中，Adobe 将所有工具的属性活动面板都统一组织到工具菜单下面的选项栏，选项栏的出现使得修改、设置工具的属性更加方便，并且大大节省了活动面板所占的屏幕空间，从而提高工作效率。

在工具栏中的每一种工具按钮都对应一个相应的选项栏。用户可以通过 Photoshop 6 主菜单的“窗口”|“隐藏选项”菜单项关闭工具选项栏，或者通过“窗口”|“开启选项”菜单项打开工具选项栏。

2．工具栏

对图像的处理是通过各种各样的工具实现的，这些工具被组织在工具栏中。在工具栏

中，可以看到有的工具按钮的右下方带有一个很小的黑色三角箭头，表明该按钮对应了几个不同的按钮，右击该按钮可以打开其他的按钮。

Photoshop 6 工具栏按钮如图 4-44 所示。

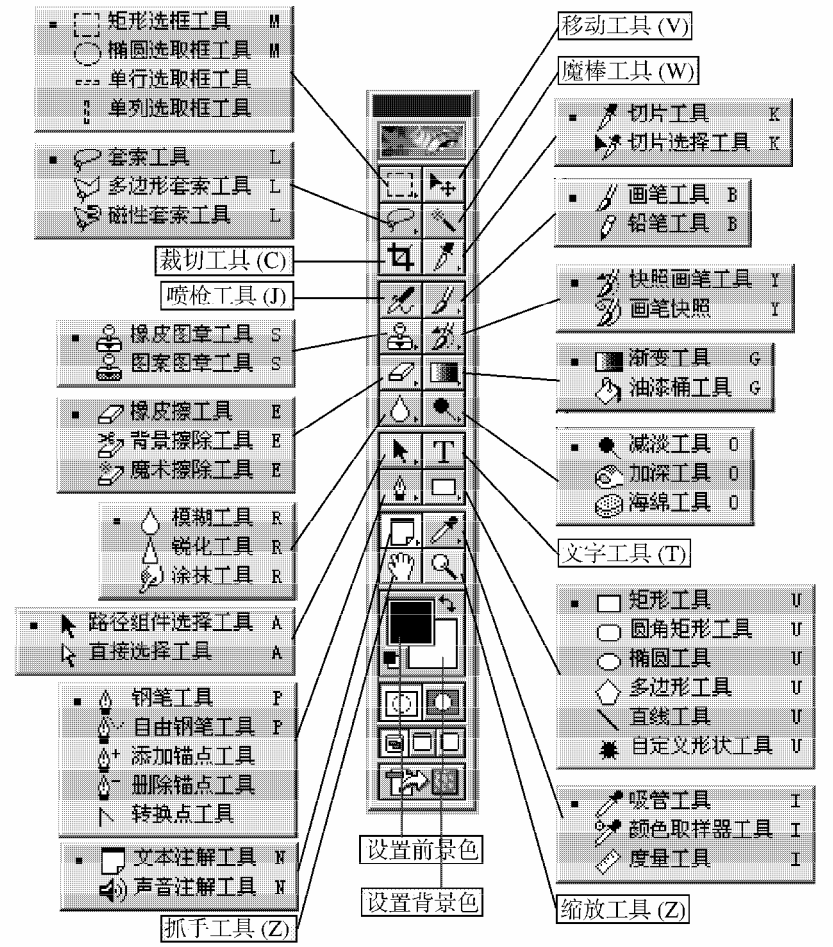


图 4-44 Photoshop 6 工具栏按钮一览

从图 4-44 可以看出，Photoshop 6 有众多的图像处理工具按钮，各按钮的功能和用法将在后面的内容中介绍。

3．活动面板

Photoshop 中的活动面板对应于 Photoshop 主菜单的“窗口”菜单，每个面板的功能将在后面的讲解中介绍。

4.6.3 图像文件

Photoshop 是一个图像处理工具，目的是把一幅图片处理成自己需要的形式。因此图像

处理的第一步就是准备原始图片素材。

1．打开或新建图像文件

Photoshop 不仅可以对现有的图片进行处理，还可以创作图片。要创建一幅新的图片，选择“文件”|“新建”菜单项，打开“新建”对话框，如图 4-45 所示。

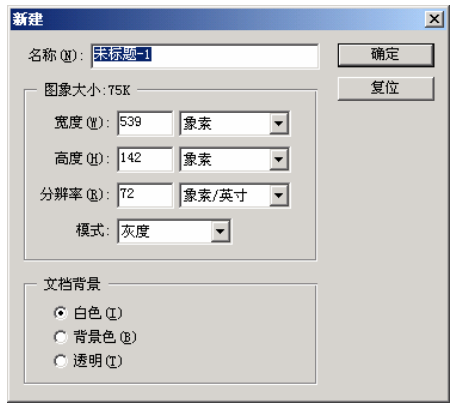


图 4-45　“新建”对话框

按照对话框的提示，分别指定图像文件的名称、图像大小、分辨率及图像模式(灰度、RGB 颜色、CMYK 颜色或 Lab 颜色)以及选定文档背景，最后单击“确定”按钮。将打开一个图像窗口，在新的窗口中就可以编辑图片了。

如果要处理一幅已经存在的图片，在“文件”菜单中，选择“打开…”菜单项，在“打开…”对话框中，选择要打开的文件，该图像文件将被打开。

图 4-46 为创建一个新文件和打开一个已经存在的文件后的结果。

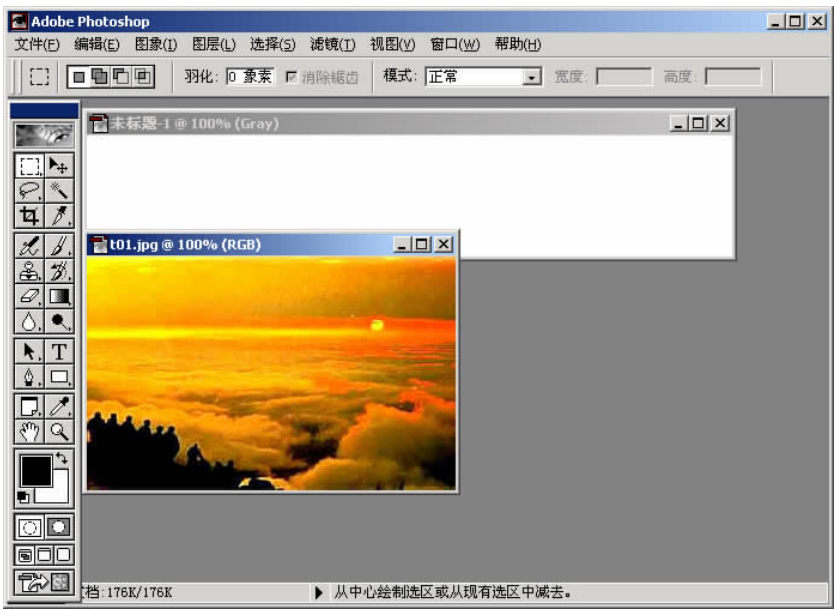


图 4-46　新建或打开图像文件

文件制作或处理完成后，可以选择“文件”|“保存”或“文件”|“另存为”菜单项，选择一个图像文件格式。Photoshop 提供了 BMP、GIF、JPG、PDF、TIF 等多种文件格式。

2．设置图像和画布的大小

打开一幅图像，选择“图像”|“图像大小…”菜单项，打开“图像大小”对话框，如图 4-47 所示。

默认状态下图像的“宽度”和“高度”是锁定在一起的，修改其中一个数值后，另一个也会成比例地随之改变。如果取消“约束比例”，则修改后的图像会变形。若取消“重定图像像素”，则“宽度”、“高度”和“分辨率”就会成比例地锁定在一起，改变尺寸或分辨率也不影响图像自身像素的变化，显示结果相同。单击“自动”按钮，可自动调整图像的分辨率和品质效果。

选择“图像”|“画布大小”菜单项，打开“画布大小”对话框，如图 4-48 所示。

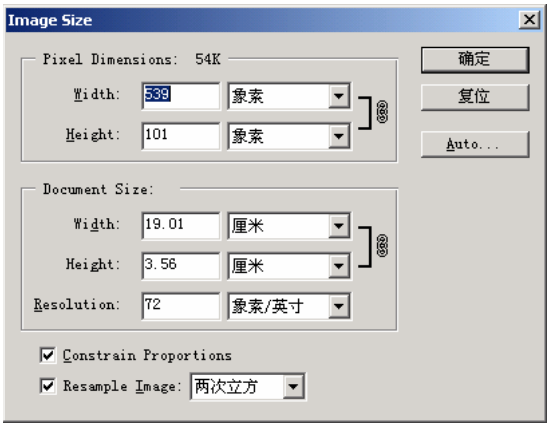


图 4-47 图像大小对话框

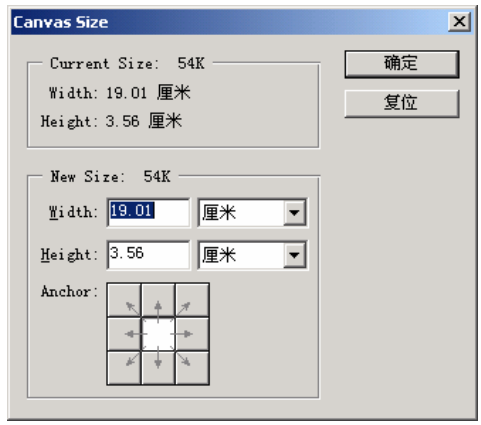


图 4-48 画布大小对话框

进行画布高度宽度的设置不会影响图像本身，“定位”项可以设置图片在画布中的位置，正中心的白色方块表明进行画布的增加或减少时，图像是在中心位置。

3．旋转画布

在“图像”菜单下，进入“旋转画布”子菜单，可以选择“180 度”、“90 度(顺时针)”、“90 度(逆时针)”、“任意角度”、“水平翻转”、“垂直翻转”来旋转画布；若选择“任意角度”，则弹出对话框，可以输入要旋转的角度。

画布旋转可以使其中的图片旋转，从而达到旋转图片的目的。

4.6.4 用 Photoshop 6 绘制图片

在介绍图片处理以前，先介绍一下 Photoshop 6 的绘图功能。

1．图层

在 Photoshop 中，用得最多的恐怕要算是图层、通道和蒙板了，这些概念一开始往往让不少初学者感到迷惑。但是当真正将它们弄清楚后，就可以充分地发挥 Photoshop 强大的图像处理功能了。

什么是图层呢？先看一个例子。

想一下平时作画时的情形：摊开画布，先画远景，可能是一带远山或村落；再画中景，小桥流水；最后才画近景，绿树掩映下的乡居茅舍。

利用计算机软件绘图，道理也是一样的，图像是有层次的，一幅图片由若干的图层构成，有的用于描绘远景，有的用于描绘中景，有的用于描绘近景，越向上的图层表示离我们越近，后面图层位于它的不透明部分以后的内容就被它挡住了。

在 Photoshop 中，图像是画在图层上的，每一幅图像都可以由多个图层拼合而成，各层之间既相互独立又互相联系。在选择某个图层进行编辑时不会影响到其他图层，各层之间的顺序可以自由调整。

要使用图层，选择主菜单的“窗口”|“显示图层”菜单项，弹出图层浮动选项板，如图 4-49 所示。

图 4-49　图层浮动面板

(1) 新建图层

在图层浮动面板的右下角是一组按钮，单击 按钮，可以新建一个图层。新建的图层在图层浮动面板中列出，并给出默认的图层名称“图层 1”…。

(2) 显示图层

在图层浮动面板中，每一个图层的前面都有一个 按钮，该按钮可决定是否显示此图层。

2．在图层上绘画

如何在图层上绘画呢？如要在 t01 图片上添加“泰山日出”四个字，具体步骤如下：

(1) 选中该图片，单击工具栏中的文字工具按钮 T，则自动增加一个文字图层，名称为“图层 1”，输入文字“泰山日出”，此时可以通过菜单栏下面的选项栏对文字格式化。

输入完成后，图层名称自动变成“泰山日出”。

(2) 在图层面板中，单击“泰山日出”图层，在工具栏中，选择“移动工具”按钮 可以将文字移动到合适的位置。如图 4-50 所示。

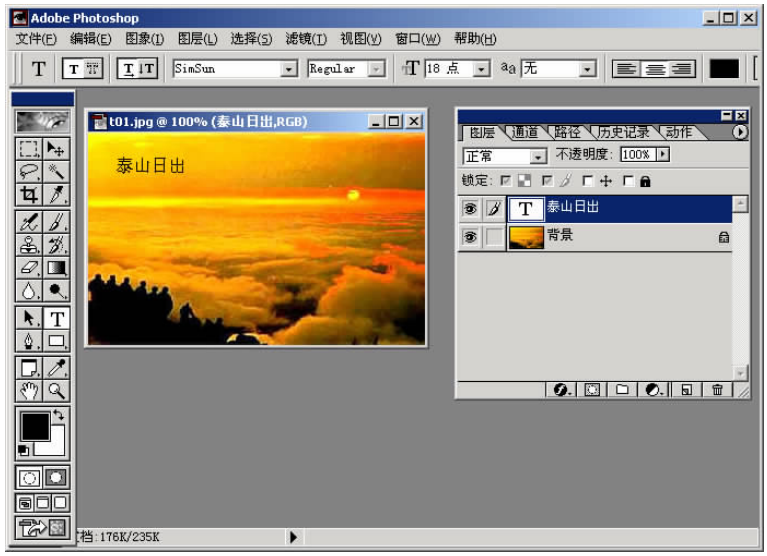


图 4-50　使用图层

3．保存图片文件

当图片编辑完成后，可以选择“文件”|“保存”或者“文件”|“另存为”菜单项，在打开的对话框中的保存选项区域，选择“图层”，可以将图层信息一并保存下来，便于以后的图片处理。

4.6.5 基本绘图操作

在 t01 图片上，新建一个图层，通过该图层来介绍基本的绘图操作。在图层面板中，选择新建的图层，并且取消“背景”和“泰山日出”两个图层的显示，如图 4-51 所示。

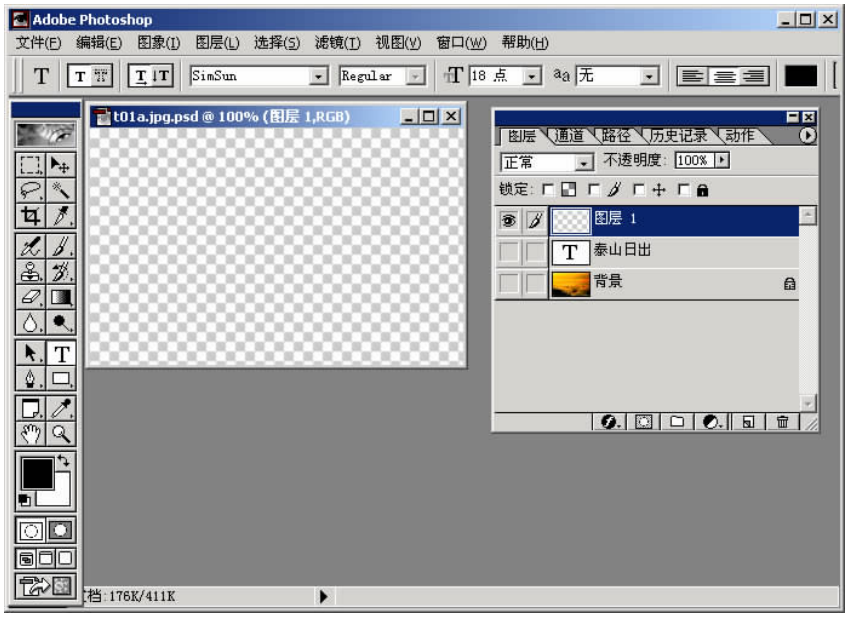


图 4-51 在新建图层上绘图

1．颜色的选取

画图的第一步是选择颜色，颜色又分成前景色和背景色两种。在 Photoshop 6 中有多种选取颜色的途径。

（1）颜色浮动面板。在“窗口”菜单中，选择“显示颜色”或“显示色板”、“显示样式”菜单项，打开“颜色”浮动面板，如图 4-52 所示。

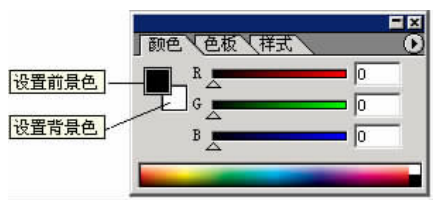


图 4-52 “颜色”浮动面板

如图 4-52 所示，分别单击前景色或背景色按钮，打开“拾取器”对话框，如图 4-53 所示。

在拾取器中，如果选定“只有 Web 颜色”选项，则颜色值都是网页安全颜色，选择 HSB 或 RGB 的某种色彩单旋钮，通过颜色滑杆、颜色域，可选出 1600 多万种颜色。选定的前景或背景颜色，

将影响绘图工具和填充区域。

(a) 选取前景色　　(b) 选取背景色

图 4-53　“ 拾取器 ” 对话框

(2) 用户也可以直接在工具栏中，单击选取前景色或背景色按钮来打开 “ 拾取器 ” 对话框，和上面的方法相同。

2．使用画笔和铅笔工具

(1) 画笔工具用于模拟毛笔的效果在图像或选区内绘图。其使用方法同喷枪工具，单击画笔工具按钮，在主菜单的下面显示画笔选项栏，如图 4-54 所示。

图 4-54　画笔工具选项栏

- 在画笔对应的下拉列表中，可以选择画笔的粗细。
- 在模式对应的下拉列表中，包括五组不同的模式，分别是：正常，溶解，背后；正片叠底，屏幕，叠加，柔光，强光；颜色渐淡，颜色加深；变暗，变亮，差值，排除；色相，饱和度，颜色，亮度。
- 不透明度　用于设置当前图层的不透明程度，设置低的不透明度将使前景色与图像已存在的颜色混合在一起，数值越大越不透明。可直接输入数值或拖移滑块来设置。
- 湿边　在页面上画出的笔触有一个边缘，像水印走过的痕迹。
- 为画笔压力按钮，单击该按钮，打开 “ 画笔压力 ” 对话框，如图 4-55 所示。

图 4-55 “ 画笔压力 ” 对话框

无论是大小、不透明度还是颜色，都有 “ 关闭 ” 和 “ 渐隐 ” 两种选择，它们影响到画笔的绘制功能。

(2) 铅笔工具用于模拟铅笔的效果在图像或选区内绘图，制作出硬边缘线的效果。其使用方法同画笔工具类似，但在属性栏中多了一个 “ 自动抹除 ” 复选框，选择此项可以在包含前景色的区域上绘制背景色。用铅笔工具拖过页面时，绘制的是前景色的颜色；停止拖动，单击并拖动，则绘制

背景色的颜色；停止拖动，单击并拖动，则又绘制前景色的颜色。

3．喷枪工具

喷枪工具模拟传统的喷枪机制，将渐变色调应用到图像，其边缘比较发散(与画笔工具相比)。单击喷枪工具按钮，在主菜单的下面显示喷枪工具选项栏，如图 4-56 所示。

图 4-56　喷枪工具选项栏

各个选项的设置与画笔工具选项栏类似，不做介绍。压力后面的下拉列表用于设置喷色的浓度，值越大，喷色越浓。

4．使用直线、矩形、椭圆等绘图工具

直线、矩形、椭圆等是最基本的图形元素，在 Photoshop 6 中，在工具栏中选择直线、矩形、椭圆等某种绘制工具，即可在某个图层上绘制图形了，操作非常简单。

选择某个图形形状工具后，在主菜单下面显示相应的选项栏，例如，矩形工具按钮的选项栏如图 4-57 所示。

图 4-57　矩形工具选项栏

在选项栏中，包含以下几组不同的按钮。

- 新建形状图层。
- 新建工作路径。
- 填充区域。
- 各个形状的按钮，可在矩形、圆角矩形、椭圆形、多边形、直线、自定义图形等形状工具间切换。

当图形绘制完成后，可以通过“编辑”菜单中的“自由变换”和“变换”菜单项对图形进行缩放操作。

5．文字工具

要在图片上添加文字，可以单击工具栏中的文字工具按钮，在菜单栏的下面显示文字工具选项栏，如图 4-58 所示。

图 4-58　文字工具选项栏

通过文字工具选项栏可以对文字进行格式化。添加文字的时候，会自动地创建一个图层。文字可以任意地调整方向。

6．油漆桶工具和渐变工具

油漆桶工具和渐变工具都是色彩填充工具，只是使用的方式不同。

(1) 油漆桶工具

油漆桶工具为单击色彩相近并连通的区域填充颜色或图案。油漆桶工具选项栏如图 4-59 所示。

图 4-59　油漆桶工具选项栏

“内容”后面的列表指定填充是采用前景色还是图案。

需要说明的是，如果在一个图层上绘制了一个椭圆或者矩形等，在 Photoshop 6 中，如果希望填充，会显示图 4-60 所示的提示对话框。

图 4-60　提示对话框

因为椭圆、矩形等是一种形状图层，它的填充可以通过在图层浮动面板中双击图层，在打开的“效果”对话框中来实现。

或者，打开路径浮动面板，在形状图层上右击，选择“建立选区”菜单项。

在保持选区的情况下，建立新的图层，取消其他图层的显示，可以看到新的图层上有一个选区。然后就可以在新的图层上使用油漆桶填充选区了。

(2) 渐变工具

选择渐变工具，显示渐变工具选项栏，如图 4-61 所示。

图 4-61　渐变工具选项栏

在渐变工具选项栏中，从左到右分别是色彩(对应渐变方式下拉列表)，渐变工具按钮(分别是线性渐变、光线渐变、角度渐变、反射渐变和菱形渐变)。

用鼠标在图像上单击渐变的起点，然后拖动鼠标，在终点再单击鼠标，这样一个渐变就完成了，可以通过线段的拖拉长度和方向改变渐变效果。

用户也可以自己定义渐变。具体步骤是：

① 双击色彩(下拉式列表区域)，打开“渐变编辑”对话框，如图 4-62 所示。

② 渐变类型包括“原色”和“杂色”两种，在滑块区域，有若干游标，可以在某个位置单击鼠标建立新游标，或者选中后，按 Del 键删除游标。

③ 游标分别定义了开始色彩、中间色彩和结束色彩。双击滑块下面的游标，可以打开颜色“拾取器”对话框，如图 4-63 所示，来输入相应的颜色值。

[例 4-2]　绘制一个立体感的球。

步骤如下：

① 新建一个 300×300 像素、RGB 模式、背景色为白色的图像文件。

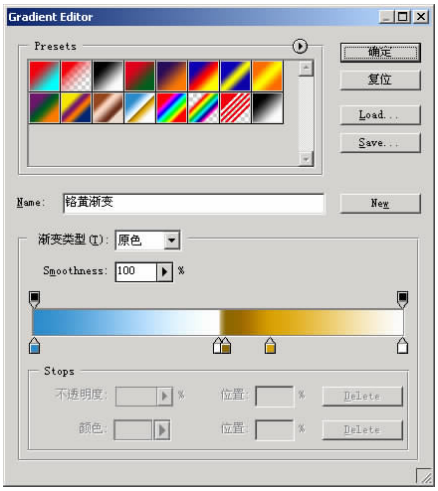


图 4-62 “渐变编辑”对话框

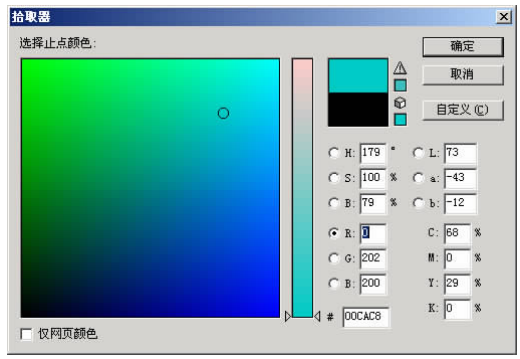


图 4-63 定义颜色的颜色拾取器对话框

② 选择椭圆工具，在对应的工具选项栏中，单击后面的“几何选项”下拉按钮，选择“圆”(Circle)，画一个圆。

③ 在“路径浮动面板”将椭圆建立成一个选区。

④ 一个图层取名为“图层 1”。

⑤ 定义一个渐变，有三个渐变色彩，分别为开始色彩(R:241，G:234，B:180)、中间色彩(R:190，G:199，B:36)和结束色彩(R:232，G:208，B:112)。

⑥ 使用渐变工具填充图层 1。

完成后的球体如图 4-64 所示。

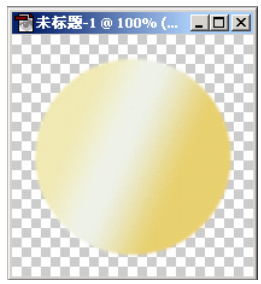


图 4-64 完成后的球体

4.6.6 图像编辑

对完成基本绘制的图像或者现有的一幅图像，还需要做编辑处理。这些操作也是通过工具栏中的工具来实现的。

1．选框、套索和魔棒工具

在 Photophop 6 中，选框、套索和魔棒工具为选择工具，其目的是建立选区，对图像进行编辑

(1) 选框工具

在工具栏中，左上角的第一个按钮即为选框工具。右击该按钮，打开选框工具类型列表。有四种不同的选框，分别是矩形选框、椭圆选框、单行选取框和单列选取框。选择其中一种，在菜单栏的下面显示相应的工具选项栏，如矩形选取框工具箱栏如图 4-65 所示。

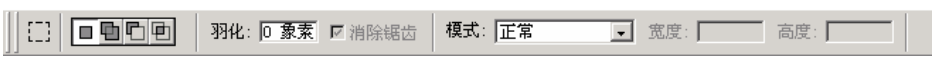


图 4-65 矩形选取框工具选项栏

相应的按钮分别介绍如下。

- 新选择按钮，用于选择一个新选区，原选区(如果存在)被取消。
- 添加到选区按钮，按下该按钮后，所作的新的选区将增加到原有的选区中，成为一个选区，这些选区不一定连通。
- 从选区中减去按钮，即在原有选区的基础上减去新选区。
- 相交选区按钮，取两个选区的重叠部分作为选区。
- “羽化”选项　用于设置选区边界的羽化程度。
- “消除锯齿”　用于清除选区边缘的锯齿。
- 宽度和高度　用于设置选区的宽度和高度。

如图 4-66 所示，在一个矩形选区上添加一个选区，形成一个多边形选区。

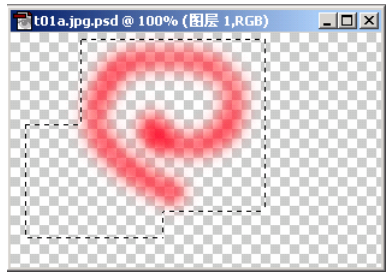


图 4-66　选区示例

对于椭圆、单行、单列选区工具与矩形选区工具功能类似，不再重复。

(2) 套索工具

套索工具主要可以用手控的方式选择一些不规则形状的选区，有三种不同的套索工具，即套索工具、多边形套索工具和磁性套索工具，对应的工具选项栏和选框工具类似，介绍省略。

(3) 魔棒工具

魔棒工具用于选取图像中色彩一致的区域。当在图像中选择某一点时，与之颜色相同或相近的点都自动溶入选区中。魔棒工具选项栏，如图 4-67 所示。

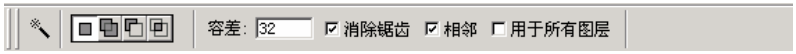


图 4-67　矩形选取框工具选项栏

“容差”选项用于控制色彩的范围，其值为 0~255，默认为 32。数值越大，所选颜色范围就越大；数值越小，所选颜色就越接近。要定义一个平滑边缘，选择“消除锯齿”；“相邻”用于选择单一颜色范围；“用于所有图层”选择所有图层中的颜色数据，否则只对当前图层起作用。

(4) “选择”菜单

图 4-68　“选择”菜单

除了使用工具栏来定义选区外，用户还可以通过 Photoshop 菜单中的“选择”菜单来对选区进行操作。“选择”菜单如图 4-68 所示。

“羽化…”命令，和“消除锯齿”类似，羽化命令用来柔化选区边缘。

- “消除锯齿”选项　是通过软化每个像素与背景像素间的颜色过度，使选区的锯齿状边缘变得比较平滑。应在使用选择工具之前指定此选项，否则不起作用。

- “羽化”选项　可设置选区与其周边像素的过度边界，使边缘模糊。一般在使用选择工具之前，先进行无羽化效果的选择(羽化值为 0)，然后选择“选择”|“羽化”菜单项来进行羽化处理。

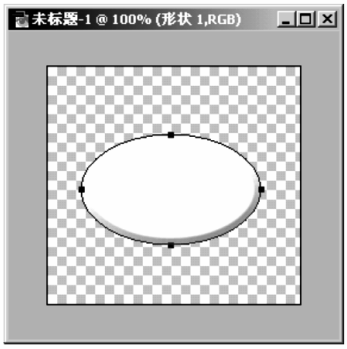
选择“变换选区”菜单项，选区边界闪烁，在选区外面，鼠标的形状变成  等形状，按住鼠标左键可以对选区进行旋转，最后单击选框工具按钮。

2．路径组件选择工具和直接选择工具

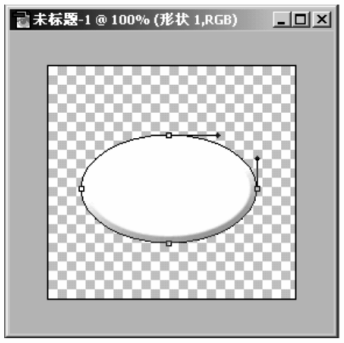
路径组件选择工具  和直接选择工具  分别用于对图形位置的移动和图形形状的调整。

(1) 如果选择路径组件选择工具，单击图形，图形四周显示四个实心的节点，如图 4-69(a)所示。用鼠标可以任意地移动图形。

(2) 选择直接选择工具，然后单击图形，则如图 4-69(b)所示。用户可以拖动节点来任意地改变图形形状。



(a) 路径组件选择工具下的图层



(b) 直接选择工具下的图层

图 4-69　选择工具演示

3．锚点工具

通过图 4-69 可以看到，路径是由一系列的锚点连接而成的，在 Photoshop 的工具栏中，包含了一组有关锚点的工具，分别是钢笔工具  、自由钢笔工具  、添加锚点工具  、删除锚点工具  以及转换点工具  等，可以增加或减少锚点，从而更便于图形形状的变化，制作出丰富的任意的形状，例如火焰等。

4．图像的移动、复制和删除工具

对于选区中的图像，我们可以执行移动、复制和删除操作。

单击移动工具按钮  ，在主菜单栏的下面显示移动工具选项栏，如图 4-70 所示。

图 4-70　移动工具选项栏

(1) 图像的移动

① 选择要移动的图像区域，单击移动工具按钮，将光标置于选区内，光标变成。

② 按下鼠标左键并拖动鼠标至指定位置，原图像位置则被背景色填充。

如果是要移动某个图层，可以将其设为当前图层，然后单击按钮，即可以平滑地移动当前图层中的图像。

如果在一个图层中有几个图形，如一个椭圆、一个矩形，如果只希望移动矩形，如何操作呢？根据上面的介绍，发现选中一个图形并移动时，整个图层都移动了。这与 Photoshop 的设计思想有关，在 Windows 中的“画图”程序中，可以单独的移动画布中的某个区域内的图像。但是，在 Photoshop 中，为了更加精细地处理图像，总是需要把一个复杂的图像分成很多层，然后对每一层进行单独处理。因此，如果一个图像包括一个椭圆和一个矩形，就应该分布在两个不同的图层中。

(2) 图像的复制

① 选中要复制的图像区域，单击移动工具按钮，将光标置于选区内，光标变成。

② 按下 Alt 键，光标变成，按下鼠标左键并拖动鼠标至指定位置。

使用橡皮图章工具和图案图章工具也可完成图像的复制。

(3) 图像的删除

使用橡皮擦工具，可通过鼠标的拖动擦除图像颜色，并用背景色填充。具体介绍参见下面的橡皮擦除工具。

或者选中某区域，按 Del 键即可将其删除。

5．图章工具

复制图章工具分成橡皮图章工具和图案图章工具两种。两种图章的功能都是复制图像，但复制的方式不同。

(1) 使用橡皮图章

橡皮图章工具选项栏如图 4-71 所示。

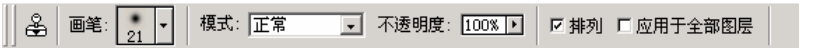


图 4-71　橡皮图章工具选项栏

具体步骤是：

① 选择橡皮图章工具。

② 将鼠标移到想要复制的图像上，按住 Alt 键，鼠标变为。

③ 拖动鼠标，在图像的任意位置开始复制。

(2) 图案图章

图案图章工具选项栏如图 4-72 所示。

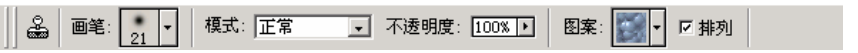


图 4-72　图案图章工具选项栏

使用图案图章首先要定义图案，方法是用矩形选取框选择所需要复制的图像，然后在选项栏的图案下拉列表中选择一个图案，或定义新图案。

6．橡皮擦除工具

橡皮擦除工具分成橡皮擦除工具、背景橡皮擦工具和魔术橡皮擦工具三种。单击橡皮擦除工具按钮，在主菜单的下面显示橡皮擦除工具选项栏，如图 4-73 所示。

图 4-73　橡皮擦除工具选项栏

在其属性栏中选择“ 抹到历史记录 ”，则会擦回到原图像状态。

背景橡皮擦工具可以擦除指定颜色，擦除后，背景图层变为一般图层。

魔术橡皮擦工具可以自动擦除颜色相近的区域。

7．裁切工具

使用裁切工具，可以在图像或图层中裁剪下所选区域的大小。具体步骤如下：

① 打开一幅图像，单击按钮，拖动鼠标则画出矩形裁剪框。

② 松开鼠标按键，图像剩下裁剪框内容，其他内容灰化。

对已经画出的矩形裁剪框，将光标移到边线上可调整裁切框的大小；移到边线外侧，就会出现旋转光标 ，拖动鼠标即可旋转选区。或者按 Esc 键，取消裁剪框。

③ 当裁减框确定后，在框内双击，完成裁剪操作。

另外，也可以使用菜单命令完成图像的裁剪任务。

使用矩形选框工具选择要裁切的选区(设置羽化值为 0)，选择“ 图像 ”菜单中的“ 裁切 ” 命令即可。

8．切片工具

在网页制作中，图片越小下载速度越快。因此，在不影响效果的情况下，切图是目前制作网页的主要方式。在 Photoshop 6 中，切片工具包括切片工具和切片选择工具两种。

(1) 切片工具

单击选择切片工具按钮，显示切片工具选项栏，如图 4-74 所示。

图 4-74　切片工具选项栏

在模式后面的下拉列表中包括如下选项。

- 正常　表示切片的大小由鼠标任意拉出。
- 约束长宽比　输入切片宽和高度的比例。
- 固定大小　输入宽度和高度值。

在正常模式下，单击切片工具，拖动鼠标得到的一个切片如图 4-75 所示。在图 4-75

中，并不是每一个区域都是用户切片，只有用实线圈定的区域才为用户切片，名称为蓝底白字，其他的区域为非用户切片。可以在上面右击，在弹出菜单中选择某个命令。

在所需编辑的切片上右击，打开“切片选项”对话框，如图 4-76 所示。

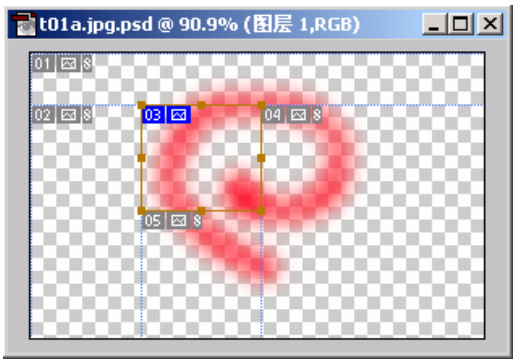


图 4-75　切片工具选项栏

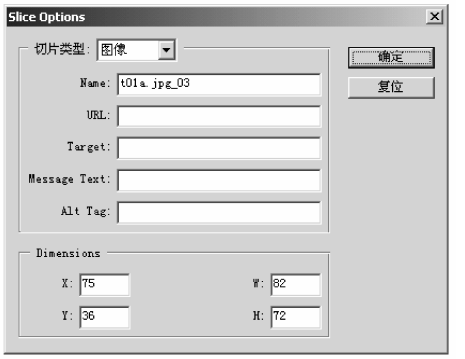


图 4-76　“切片选项”对话框

(2) 编辑切片

在切片选项对话框中，可以为切片命名，增加交互功能，比如输入 URL 等，选择打开浏览器方式等。

(3) 排列切片

切片往往造成图片的重叠，可以使用切片选择工具来排列切片的顺序。

选择切片选择工具，如图 4-77 所示。

图 4-77　“切片选项”对话框

- 置为顶层。
- 前移顶层。
- 后移顶层。
- 置为底层。

9．图像的缩放、旋转等

图像编辑绘制完成后，往往还需要进行大小、位置的调整，可以通过“编辑”菜单中的“自由变换”和“变换”命令来完成。

4.6.7　图像修饰与调整

在图像处理中，除了一些刚性化的编辑外，往往还需要对图像进行修饰，这样得到的图片才更具有艺术美感。

1．模糊、锐化和涂抹

在工具栏中，选择模糊工具，显示模糊工具选项栏，如图 4-78 所示。

图 4-78　模糊工具选项栏

2．减淡、加深和海绵工具

减淡工具和加深工具主要用于改变图像的亮调和黯调，原理来源于胶片曝光显影后，经过部分暗化和亮化来改变曝光效果。

减淡和加深工具选项栏相同，如图 4-79 所示。

图 4-79　减淡/加深工具选项栏

- 画笔。
- 模式　分成暗调、中间调和高光三种。
- 曝光度　调整处理图像时的曝光强度，一般开始值较小，为 15%左右。

海绵工具是一种用于调整图像饱和度的工具，可以提高或降低色彩的饱和度，对应的工具选项栏如图 4-80 所示。

图 4-80　海绵工具选项栏

其中，模式分为去色和加色两种。

3．图像色彩调节

图像色彩的调整是将当前范围的像素值映射到新范围的像素值上。在进行色彩调整之前最好先检查图像质量和色调范围，以便于快速地有针对性地校正和调整色彩。

图像区域中像素数目越多，细节也就越丰富，越易产生高质量的输出。质量差的照片或扫描将很难进行调整。选择“图像”菜单下的“直方图”，弹出对话框。暗色调的图像细节都集中在暗调区，亮色调的图像细节都集中在高光区(必须是可打印的区域)。

如果图片偏暗，可以通过色阶来调整，具体步骤如下：

① 在“图像”菜单中，选择“调整”|“色阶…”菜单项，打开“色阶”对话框，如图 4-81

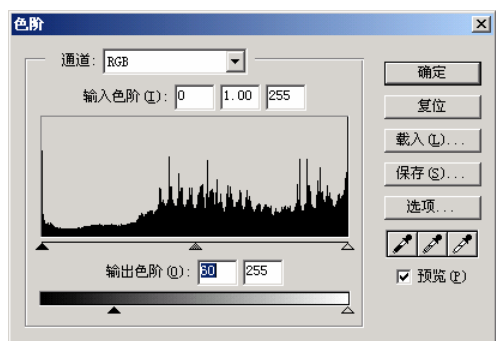


图 4-81　“色阶”对话框

所示。

① 拖动下面的输出色阶滑杆可以调整整个图片的亮度。

图像的色彩调整还包括亮度及对比度、曲线、色调平衡、色相/饱和度、去色、替换颜色、可选颜色、通道混合器、渐变映射、反相、色调均化、阈值、色调分离等。它们都被组织在主菜单的“图像”|“调整”子菜单中。

4.6.8　深入理解图层、通道、路径和图层蒙板

在 Photoshop 6 中，最多的和最重要的就是图层等概念了。

1．图层

前面已经多次讲到图层，下面详细地解释图层的功能，图层示例如图 4-82 所示。

- 图层名称　每一图层都有一个名称以便于区分，若未起名则 Photoshop 6 会自动依次命名为图层 1、图层 2……
- 图层预览缩图　显示图像内容，可以快速识别每一图层。
- 眼睛图标　用于显示或隐藏图层。图层隐藏时，不能对其进行任何编辑操作。
- 当前图层　以蓝色显示，左侧有一历史画笔图标。一个图像只有一个当前图层。

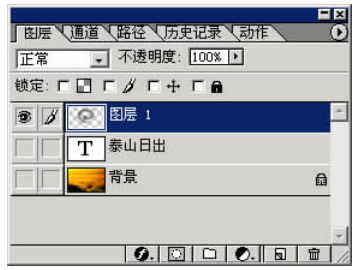


图 4-82　图层

(1) 设置图层的透明度

选择几个图层同时显示，调整不透明度后面的不透明强度值，可以看到几个图层叠加后的总体效果。

(2) 锁定选项

在图层窗口中，包含四个锁定选项，分别是：

- 锁定透明像素(锁定当前图层的透明区域使其不能被编辑)。
- 锁定图像像素(使当前图层和透明区域不能被编辑)。
- 锁定位置工具(使当前图层不能被移动)。
- 全部锁定(所有图层都不能被编辑)。

(3) 其他工具按钮

在图层浮动面板的右下角，有六个工具按钮，分别是：

- 依次是添加图层样式(使当前图层增加图层风格效果)。
- 添加蒙板(建立一个图层蒙板，它将选择区域外的图像保护起来，黑色代表隐藏，编辑完成后去掉蒙板即可)。
- 创建新组(新建一个存放图层的文件夹)。
- 创建新的填充或调整层(可对图层进行颜色填充和效果调整)。
- 创建新的图层(在当前层上面创建一个新层)。
- 删除图层(在选定的图层上单击或直接拖到此处)。

单击右上角的，打开图层浮动选项板菜单，该菜单能完成一些较常用的图层操作，如新建、复制、删除等操作。

使用“图层”菜单也可实现图层功能。选择“图层”菜单，其下拉菜单分 11 组，每一组可实现一组图层功能。

(4) 图层的操作

在图层上右击鼠标，打开快捷菜单，或者在图层浮动选项板中单击，从弹出的菜单中可以完成图层属性、新建图层、复制图层、删除图层、图层透明区域显示状态的设置、图层效果的编辑、图层合并等操作。

2．通道

通道用来存放图像的颜色信息，也可存放选区。当打开一幅图像时，即自动建立了颜色信息通道。不同的颜色模式有不同的颜色通道数目，如 RGB 图像默认有三个颜色通道及一个复合通道；CMYK 图像默认有四个颜色通道及一个复合通道。用户可建立新的 Alpha 通道。通道中可保存蒙板。一个图像最多可有 24 个通道，每个通道都是 8 位灰度图像，显示 256 级灰阶。

选择“窗口”|“显示通道”菜单项，显示浮动选项板。如图 4-83 所示。

图 4-83 通道

(1) 通道浮动选项板

通过它可以完成通道的所有操作。

单击可显示或隐藏当前通道。由于主通道(RGB)是由各原色通道组成，因此当单击某原色通道时主通道会自动隐藏。

通道浮动面板底部有四个工具按钮，分别是：

- 选区载入(用于将通道中的选区调出)。
- 将选区存储为通道(将选区存储为通道，以备用)。
- 创建新通道(创建新的 Alpha 通道)。
- 删除通道。

(2) 通道操作

右击某通道，或者单击按钮弹出通道操作菜单，可以完成新建通道、复制通道、删除通道、通道的分离、合并通道、通道运算等操作。

3．路径

路径是使用钢笔、磁性钢笔等工具勾画出的任何形状的线条，是不包含像素的矢量对象，使用路径可以进行复杂图像的选取，也可将路径转换为选区边框、描边路径等。

(1) 路径编辑工具

路径是由一条或多条直线或曲线组成的封闭或开放的图形，节点表示线段的端点；每个选择的节点显示一条或两条方向线，方向线和方向点的位置确定了曲线段的大小和形状。

平滑曲线由平滑点连接，当移动平滑点一侧的方向线时，该点两侧的曲线会同时调整；有拐角的曲线由角点连接，当移动角点的一条方向线时，只调整与方向线同侧的曲线。

使用钢笔、自由钢笔、添加锚点、删除锚点、转换点、路径组件选择、直接选择等工具，可以绘制和编辑路径。

(2) 路径浮动选项板

选择“窗口”|“显示路径”菜单项，可以显示路径浮动选项板，如图 4-84 所示。

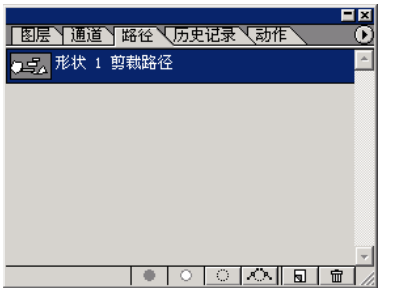


图 4-84　路径

底部的六个按扭依次是用前景色填充路径、用前景色描边路径、将路径作为选区载入、从选区建立工作路径、创建新路径、删除当前路径。若按住 Alt 键再单击它们，则弹出相应对话框。单击右上角的，弹出路径操作菜单。

(3) 路径的操作

路径可以看作图层中的图像，可以对它进行保存、复制、删除、移动等操作。也可完成填充路径、描边路径、剪贴路径和调板选项。

(4) 选区和路径的转换

从“路径菜单”中选择建立工作路径、建立选区命令，可将选区转换成路径，或将路径转换成选区。使用路径选项板中的和也可直接完成选区和路径的转换。

4．图层蒙板

图层蒙板实际上就是对某一图层起遮盖效果的在实际中并不显示的一个遮罩，它在 Photoshop 6 中表示为一个通道，用来控制图层的显示区域与不显示区域及透明区域。蒙板中出现的黑色表示在被操作图层中的这块区域不显示，白色表示在图层中这块区域显示，介于黑白之间的灰色则决定图像中的这一部分以一种半透明的方式显示，透明的程度则由灰度来决定，灰度为百分之多少，这块区域将以百分之多少的透明度来显示。

下面简单介绍图层蒙板的具体操作过程。

(1) 为图层添加蒙板

先选定要添加图层蒙板的图层，然后进入“图层”|“添加图层蒙板”子菜单，在这个菜单下面有两个菜单项，一个是“显示全部”，一个是“隐藏全部”，“显示全部”指的是当使用这个命令时系统会直接把整个图层显示出来，即把图层蒙板全部填上白色，“隐藏全部”则相反，会先把图层蒙板填上黑色，该选择根据实际操作的需要确定，先选择“显示全部”。

(2) 描绘图层蒙板

在图层上添加一个图层蒙板，图层浮动面板眼睛右边的图标变成了一个方块中间有个圆形，如图 4-85 所示。

在图层名称的前面，增加矩形，这就是该层的蒙板。单击该蒙板，可以进行蒙板的绘制，和普通图层的绘制相同。例如使用渐变工具，如图 4-86 所示。

右击蒙板，会自动打开快捷菜单，可以进行删除、停用蒙板等操作。

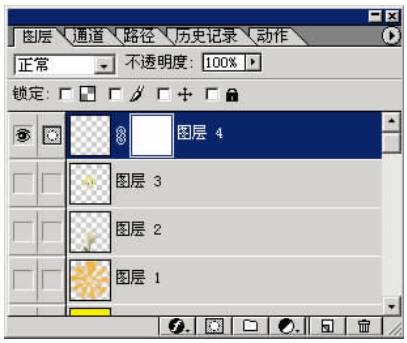


图 4-85 含有蒙板的图层(图层 4)

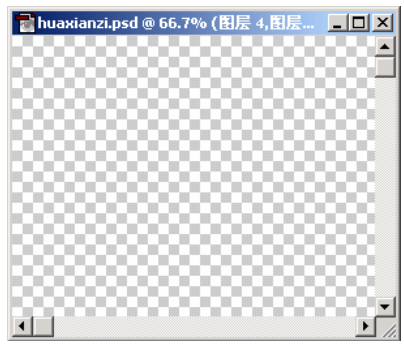


图 4-86 蒙板绘制界面

4.6.9 使用滤镜

滤镜是 Photoshop 中功能最丰富、效果最佳的工具之一。使用滤镜可以生成多种光怪陆离、变化万千的特殊变换效果。Photoshop 同时还支持非 Adobe 公司开发的外挂滤镜，从而进一步扩充 Photoshop 滤镜功能。这些滤镜安装后将出现在“滤镜”菜单的底部，与内置滤镜使用方法相同。

1．滤镜基本知识

使用滤镜时，由于图像处理的过程比较复杂，所以操作的速度会比较慢，可以通过技巧地运用来提高工作效率。

(1) 如果没有必要对整个图像应用滤镜，可以在图像上定义选区，确定要应用滤镜的范围；如果要使处理的部分和图像其他部分自然地过渡，可以定义羽化的选区。

(2) 滤镜可以应用到某一层的单个通道中。

(3) 滤镜的处理以像素为单位，它的效果会因为图像分辨率的不同而不同。

(4) 执行某一个滤镜处理图像之后，“编辑”菜单中的“消退”菜单项变得可用，选择它可以实现效果混合。

(5) 滤镜对话框中按住 Alt 键，在“取消”与“复位”之间切换。

(6) 使用“编辑”菜单中的“清理”、“所有”菜单项，可以释放内存，但如果需要保留以前的操作，不能使用该命令。

2．滤镜的分类

在 Photoshop 中，滤镜的数量众多，在此不一一介绍，图 4-87 所示是 Photoshop 6 中的滤镜菜单。

每个滤镜的功能请自行参考有关的 Photoshop 6 专门书籍。

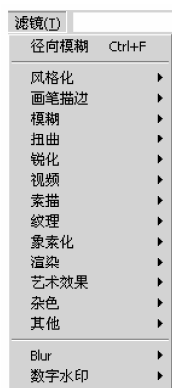


图 4-87 “滤镜”菜单

4.6.10　图片合成举例

Photoshop 具有功能强大的图片合成功能，下面介绍“花仙子”图片的合成过程，来理解 Photoshop 图像处理的综合应用。整个图像处理过程可以分成两大部分，第一部分是制作背景图片。第二部分是将已有的花仙子和荷花图片在背景上图层上合成，构成最后的效果。已有的图片素材如图 4-88 所示。

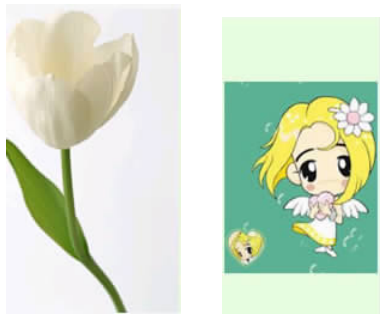


图 4-88　制作素材

详细的制作步骤介绍如下：

(1) 打开 Photoshop，在“文件”菜单中，选择“新建…”菜单项，新建一个 500×500dpi 的图像，自动生成一个白色的“背景”图层。然后先设置前景颜色为黄色，使用油漆桶工具将背景填充为黄色。

(2) 在打开的图层浮动面板中，新建一个图层(图层 1)，选择矩形选框工具，在矩形选框工具选项栏上，单击添加选区按钮，在图层 1 上建立多个矩形选区，并用橙色填充，如图 4-89 所示。

(3) 选择菜单“选择”|“取消选择”(Ctrl+D)来取消选区，在“滤镜”菜单中，指向“扭曲”，执行“极坐标”命令，结果如图 4-90 所示。

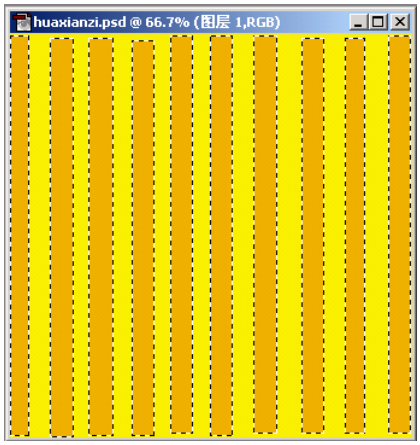


图 4-89　背景图层

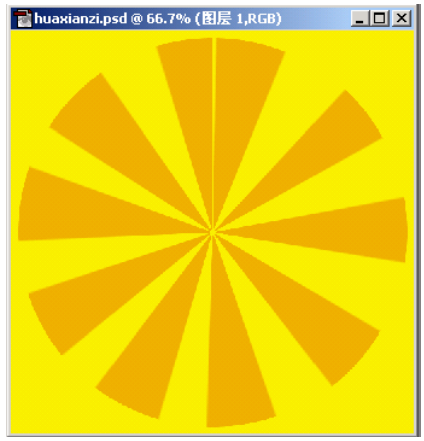


图 4-90　极坐标滤镜效果

(4) 对于图层 1，选择“滤镜”|“模糊”|“特殊模糊”菜单项，打开“模糊效果”对话框，选择相关的设置，如图 4-91 所示。然后单击“确定”按钮，得到的效果如图 4-92 所示。

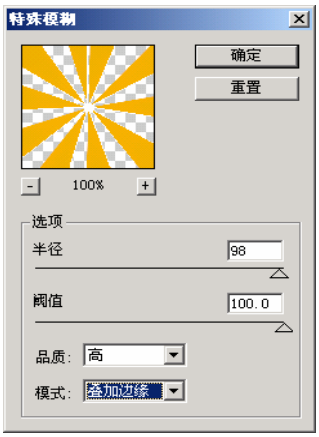


图 4-91 “特殊模糊”对话框

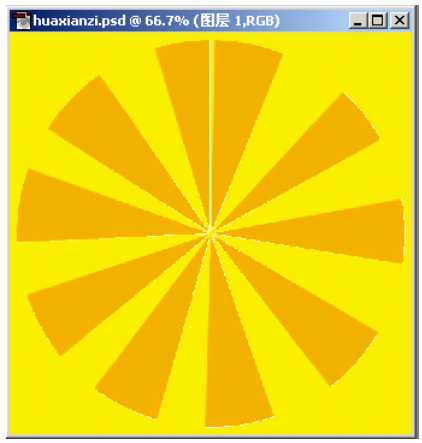


图 4-92 特殊模糊滤镜效果

(5) 选择“滤镜”|“模糊”|“径向模糊”菜单项，打开“径向模糊”对话框，选择相关的设置，如图 4-93 所示。然后单击“确定”，得到的效果如图 4-94 所示。背景制作即完成。

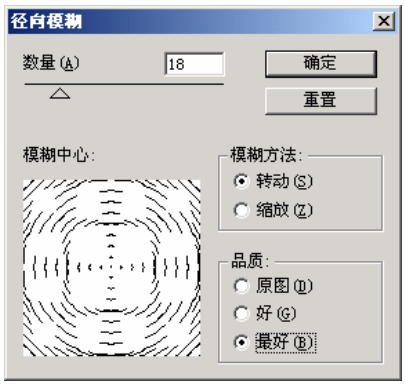


图 4-93 “径向模糊”对话框

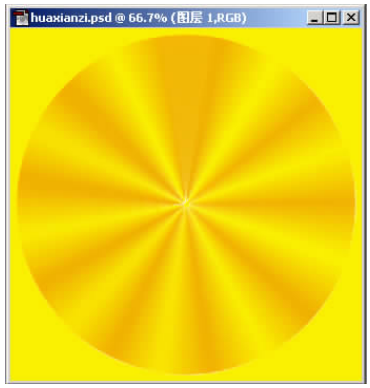


图 4-93 径向模糊滤镜效果

接下来进行图片的合成。

(6) 选择“文件”|“打开…”菜单项，将基本素材的荷花图片在 Photoshop 中打开，并且自动创建一个层，默认名字为“背景”。

由于图片素材大小不同，有时需要通过“图像”|“图像大小…”菜单项，来调整图片的大小。

(7) 将荷花图片的图层加到创作的图像文件中。

具体步骤如下：

① 单击被复制的图片(荷花)，图层面板显示被复制图片的图层信息。

② 在图层面板中通过鼠标，将一个图层拖放到另外一个图像文件(花仙子)窗口中。

这样就将一个图像文件的图层(荷花图片)复制到了另外一个图像文件(花仙子)中，成为一个新的图层(图层 2)。

(8) 将所需要的部分(荷花)从图层中提取出来，具体步骤如下。

① 利用套索工具，或磁性套索工具，定义提取图像的边界；或者用钢笔将荷花外围轮廓勾出，并将所勾路径转换为选区。如图 4-95 所示。

② 在选区内右击鼠标，执行“选择反选”命令，将未被选择的内容选中。

③ 按 Del 键，将选区内的内容删除，即可得到要提取的图像，如图 4-96 所示。

图 4-95　勾出要取得的荷花

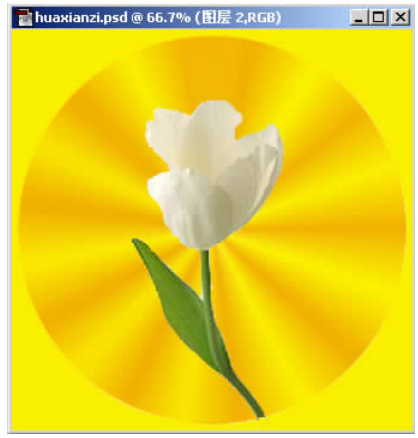


图 4-96　提取的荷花图片

(9) 在 Photoshop 6 中打开基本素材中的卡通图片(仙子)，用类似步骤(6)、(7)的方法，调整图片大小，并将仙子图片拖放到花仙子文件中，创建图层 3，利用移动工具将仙子移动到合适的位置，如图 4-97 所示。

类似步骤(8)，采用磁性套索工具取出仙子图层中的仙子，如图 4-98 所示。

图 4-97　将仙子图片拖放到花仙子文件

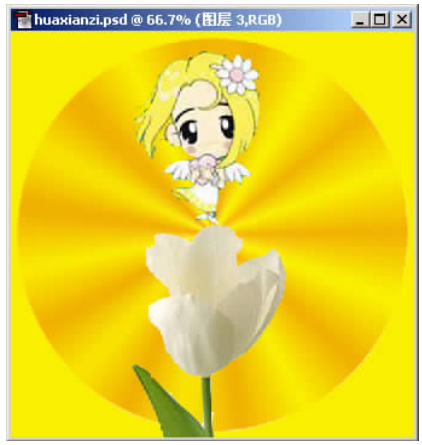


图 4-98　提取的仙子

(10) 调整花仙子图片中荷花和仙子的位置，如图 4-99 所示。

(11) 从图 4-99 可以看到，仙子图层(图层 3)遮挡了荷花图层(图层 2)，在图层浮动面板中，调整仙子图层的不透明度，使得可以看到被遮挡的荷花图层。

然后，回到仙子图层，使用套索工具或者钢笔工具勾画出遮挡荷花图像的部分，并转化为选区。如图 4-100 所示。

图 4-99 调整荷花和仙子的位置

图 4-100 勾出遮挡荷花的部分

(12) 按 Del 键，把仙子图层中遮挡荷花的部分删除，如图 4-101 所示。花仙子对应的图层面板如图 4-102 所示。

图 4-101 最后合成的花仙子图片

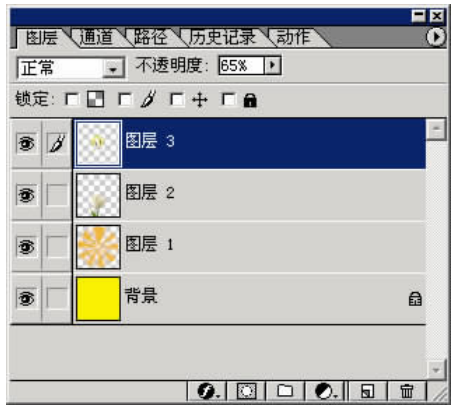


图 4-102 花仙子图像文件对应的图层面板

4.7 Flash 与动画制作

Flash 是一种交互式矢量多媒体技术，其前身是早期网上流行的矢量动画插件 Future Splash， Macromedia 公司收购 Future Splash 以后将其改名为 Flash，并于 1996 年 6 月推出 Flash 4.0，2000 年 7 月推出 Flash 5。

对于网页制作来说，Flash 是一个最精彩的矢量动画工具。但是，2002 年 3 月，随着 Flash MX 的出现，这种情况发生了变化，Flash MX 极大地扩展了 Flash 的功能，它可以创建完整的动态站点。Flash MX 已不再仅仅具有动画设计制作功能，它具有从内容显示到数据库连通，视频调试和整合，以及简单的多媒体编辑功能，是制作各种网络广告、游戏、多媒体应用程序、教程等的理想工具。

Flash 的最新版本是 Flash MX 2004，分为专门面向设计者的 Flash MX 2004 和专门面向开发者的 Flash MX 2004 Professional。但是该版本过于专业，其中的 ActionScript 2.0 语言已经与 Java 等高级编程语言类似。由于使用 Flash MX 的用户非常普遍，本节仍然使用 Flash MX 进行讲解。当然，所有的程序在 Flash MX 2004 下同样适用。

需要说明的是，Flash 的播放需要 Flash Player，一般情况下浏览器都具有此功能，如果不行，需要从网上下载。或者安装 Flash，会自动将 Flash Player 安装到计算机中。

4.7.1　Flash 的功能及特点

为什么要使用 Flash 呢？ HTML 语言是网页制作语言，由于功能有限，很难设计出令人耳目一新的动态效果。为了增加网页的动态功能和交互能力，产生了各种脚本语言，使得网页设计更加多样化。但是，程序设计需要一定的编程能力，很难普及。实际上，人们更需要的不是编程，而是一种既简单直观又功能强大的动画设计工具，Flash 就是要满足这种需求。

1．Flash 的基本功能

Flash 具有如下基本功能：

(1) 绘图功能　Flash 含有一个功能强大的绘图工具箱，可以完成动画所需要的简单的图像绘制。

(2) 矢量动画制作功能　这是 Flash 的基础，是 Flash 最基本的功能，通过 Flash 可以制作出补间动画。

(3) 程序功能　通过脚本编程，可以更好地实现交互动画制作。

另外，新版的 Flash MX 还增加了许多新的功能，例如组件、行为等控制。

2．Flash 的主要特点

(1) 使用矢量图形和流式播放技术与位图图形不同的是，矢量图形可以任意缩放尺寸而不影响图形的质量；流式播放技术使动画可以边播放边下载，网页浏览者不再需要焦急等待。

(2) 通过使用关键帧和元件(symbol)使得所生成的动画(.swf)文件更小，下载迅速，使动画可以在打开网页很短的时间里就得以播放。

(3) 把音乐、动画|、声效、交互方式融合在一起，可以创作出令人叹为观止的动画(电影)效果。

(4) 强大的动画编辑功能使网页设计者可以随心所欲地设计出高品质的动画，通过 action 和 as command 功能可以实现交互性，使 Flash 具有更大的设计自由度。另外，Flash 与当今最流行的网页设计工具 Dreamweaver 配合默契，可以直接嵌入网页的任一位置。

4.7.2 启动 Flash MX

当 Flash MX 在计算机上安装完成后，系统会在“开始”|“程序”菜单中，添加“Adobe”子菜单，选择 Macromedia | Macromedia Flash MX 菜单项，即可启动 Flash MX，打开一个默认文档，文件名为“未命名-1”，如图 4-103 所示。

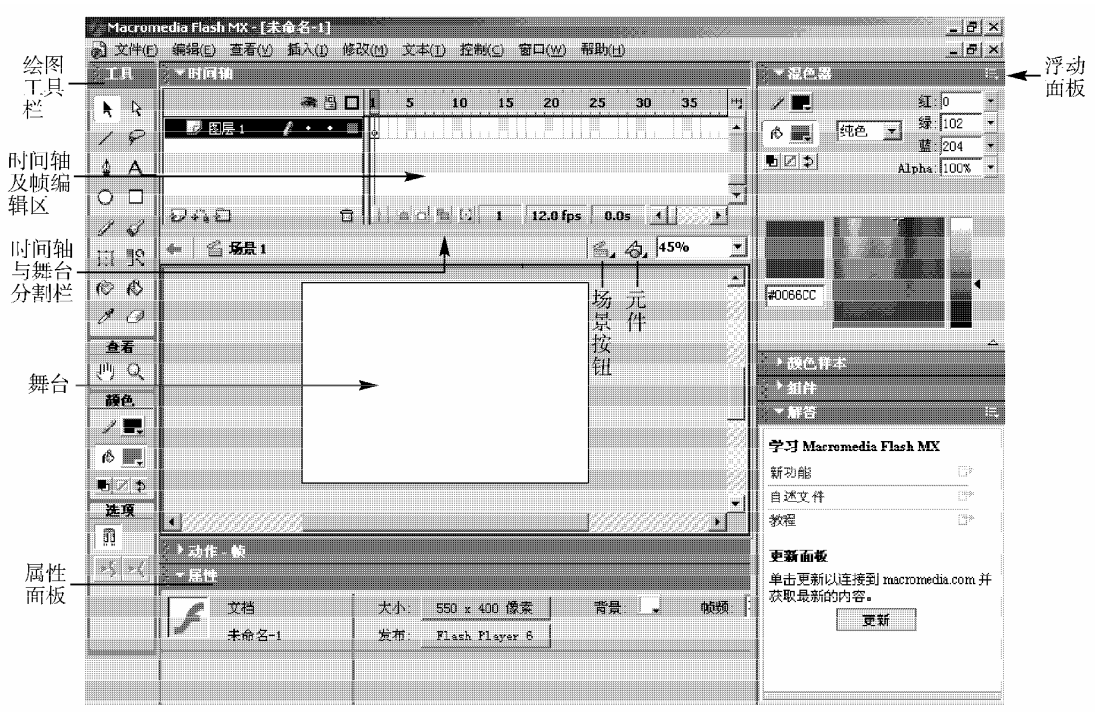


图 4-103 Flash MX 主窗口界面

图 4-103 所示是 Flash MX 的默认布局。用户可以通过“查看”和“窗口”菜单定制自己喜欢的屏幕布局，显示或关闭其中的工具和面板。

1．绘图工具栏

绘图工具栏用于绘制、填充颜色、选择和修改对象。它分为工具区、查看区、颜色区、选项区几大部分，放置了可供图形和文本编辑的各种工具。

2．时间轴及动画帧编辑区

Flash 按时间组织动画帧的演播顺序，在时间轴及动画帧编辑区可以进行插入帧、删除帧、编辑帧等操作。同时，还可以插入、删除图层等。

3．舞台区

动画编辑区，演播时动画可见的部分。包含场景转换按钮、元件按钮和影片显示比例

下拉列表(调节动画显示的大小)。

可以通过折叠动作和属性面板、关闭右侧的浮动面板来扩大舞台区所占的屏幕空间。

4．属性面板区

在 Flash 主窗口的下面，显示文档、帧、或者选择对象的各种属性。其中的动作面板用于编写脚本程序，完成动画的交互性控制。

5．浮动面板

在 Flash 主窗口的右侧， Flash 还为动画的制作提供了一组创建、控制各种动画对象的工作面板。可以通过“ 窗口 ”菜单打开或关闭显示，或者将其拖放到其他位置。

另外，每一个浮动面板标题栏的左边和右边都有一个相应的按钮，单击浮动面板标题栏左侧的按钮，可以打开或折叠面板显示。单击浮动面板标题栏右侧的按钮，打开一个菜单，包含一组当前面板的操作命令，都含有“ 最大化面板 ”、“ 关闭面板 ”等命令。

6．Flash 文档与影片测试

创作中的 Flash 文档扩展名为.fla，发布后的扩展名为.swf。在文档的创作过程中，可以通过“ 控制 ”菜单中的“ 测试影片 ”命令来演示 Flash 导出的.swf 拷贝，或者按 Ctrl+Enter 组合键。

当查看影片之后，选择“ 文件 ”|“ 关闭 ”菜单项可关闭.swf 文档演示。

4.7.3　Flash 动画的基本概念

在正式学习 Flash 动画创作以前，先介绍一些相关的概念，以便于理解学习的内容。

1．矢量图

计算机中显示的图形一般可以分为两大类，即矢量图和位图。矢量图使用直线和曲线来描述图形，这些图形的元素是一些点、线、矩形、多边形、圆和弧线等，它们都是通过数学公式计算获得的。例如一幅花的矢量图形实际上是由线段形成外框轮廓，由外框的颜色以及外框所封闭的颜色决定花显示出的颜色。由于矢量图形可通过公式计算获得，所以矢量图形文件体积一般较小。

矢量图形最大的优点是进行放大、缩小或旋转操作不会失真；最大的缺点是难以表现色彩层次丰富的逼真图像效果。Adobe 公司的 Illustrator、Corel 公司的 CorelDRAW 是众多矢量图形设计软件中的佼佼者。Flash MX 制作的动画也是矢量图形动画。

2．帧、帧频与动画

帧是指电影、电视等动画中的一幅画面。在 Flash 中，帧包含了 Flash 作品的播放内容(图形、音频、素材符号和其他嵌入对象)。一系列的帧以一定的速度播放就形成了动画效果。在一秒钟内显示的帧数称为帧频(frame per second，fps)。高帧频(帧频速度快)会使图像显得有动感；而帧频速度慢会让人产生图像有较大停顿的感觉。

电影进行图像播放时每秒播放 24 个图片，即 24 帧。但为什么人们都感不到图像闪烁呢？原因是电影在放映的时候每个镜头都要重复多放一次，即每秒 48 次。这类似隔行扫描电视。

电视的帧频是 25 帧/秒，因为每秒比电影多一帧，所以每次要在电视上放电影时，都得要进行格式转换(多插一帧，即对某帧进行重播)。电影和电视的帧频为什么不统一为 25 帧/秒或 24 帧/秒呢？电影不愿换成 25 帧/秒的理由是，人们对每秒 24 帧已很满意，换成 25 帧/秒会增加成本；电视不愿换成 24 帧/秒的原因是电网交流电的频率为 50 Hz，如果换成其他帧频，受到如荧光灯之类的灯光调制时会出现差拍。

Flash 作品的播放也是以帧为基本单位进行的，系统默认的帧频率为 12 fps。帧的多少可以衡量播放动画的时间长短。

Flash 中的帧分为三种类型。

- 关键帧　用于表示关键性的图形变化或动作的帧称为关键帧，是 Flash 动画的基础。Flash 的所有展示对象都放置在各个关键帧中，关键帧增多，所形成的动画文件将增大。

在时间轴刻度框中加黑点表示一个关键帧，空心小圆圈表示空白关键帧，即帧内没有内容。

- 静止帧　静止帧是相邻关键帧的延续，对应时间轴刻度框为灰色。静止帧中显示的是与其相邻的前一个静止关键帧中的内容，若前一个关键帧为空白关键帧，则其后的静止帧都为空白帧(时间轴刻度框为白色)。
- 过渡帧　过渡帧出现于补间动画的两个关键帧之间(时间轴刻度框之间有箭头线)。过渡帧中显示了所在 Flash 补间动画的某一个中间效果。对应于动作渐变和形状渐变，有两种过渡帧。一种是两个关键帧之间(补间)用黑色的实线表示，背景为淡紫色(浅蓝色)，表示动作渐变动画。另一种是两个关键帧之间(补间)用黑色的实线表示，背景为淡绿色，表示形状渐变动画。如果两个关键帧之间为虚线，说明中间的过渡存在错误，无法正确实现补间。可以右击鼠标，在快捷菜单中删除。

动画是指物体在一定的时间内发生的变化过程，包括动作、位置、颜色、形状、角度等的变化，在计算机中是用一幅幅的图片来表现这种变化的。当这些图片以一定的速度连续播放时，就会给人一种“动”的感觉。在 Flash 的动画制作过程中只要给出第一幅和最后一幅图片，中间的变化则由计算机自动生成，这大大减轻了动画创作者的负担，使得动画创作由传统的手工制作，转变为电脑的合成，从而为动画制作开创了一个新的天地。

3．时间轴

时间轴为用户提供了一个修改层和帧的界面，如图 4-104 所示。

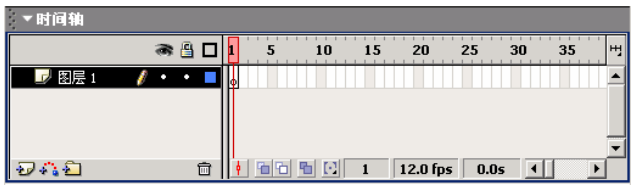


图 4-104　时间轴

时间轴上有许多小格，每个格子代表一帧，5 的整数倍数的帧上有数字序号。一帧可以放一幅图片，许多帧先后排列形成一个 Flash 作品。时间轴上的红色竖线，为时间指针，也称播放头，表示当前的帧位置。时间轴状态栏中包含了多个操作按钮，即显示当前帧、帧频、时间和滚动条。包含的操作按钮分别介绍如下。

- 帧居中。当动画较长且播放头的位置在后半段时，帧居中可以将播放头所在位置居中显示，方便对其前后帧操作。
- 绘图纸外观。可以让工作区中显示多个帧的图像，便于调整帧中对象的位置。这时在帧的上方有一个大括号一样的效果范围，两头可以拖动范围。
- 绘图纸外观轮廓。作用与绘图纸外观类似，可以显示各帧图形的轮廓线，没有填充色，因而显示速度较快。
- 编辑多个帧。
- 修改绘图纸标记。可以设置显示绘图纸的数量和显示帧的标记，默认值为两张绘图纸。

用户可以在帧块上右击，或通过“插入”菜单，对帧(关键帧)进行插入、删除、复制、粘贴等操作。

4．图层

图层显示在时间轴面板的左侧(见图 4-105)。和 Photoshop 类似，Flash 中的图层也可以看成是可以叠放在一起的透明的胶片，在不同层上编辑不同的动画而互不影响。Flash MX 中的图层类型分为：普通图层、引导层、被引导层、遮罩层和被遮罩层。当图层很多时，可将相关的图层进行分类，建立层文件夹，从而简化层的操作和管理。

时间轴面板中的图层编辑区，包含的图层操作按钮有：

- 插入图层按钮。
- 插入图层文件夹按钮。
- 显示/隐藏所有图层。
- 锁定/解除锁定所有图层，被锁住的层可以正常显示，但不能被编辑。
- 显示所有图层的轮廓。

另外，在每一个图层条目中，图层名称的右边也有四个按钮，分别是(表示当前图层)、关闭/打开当前图层显示、图层锁定、显示轮廓。

在图层上右击，弹出快捷菜单，选择“属性”菜单项，将打开“图层属性”对话框，如图 4-105 所示。

在“插入”菜单中，执行“图层”命令，可以插入一个图层，此时时间轴面板左侧的图层列表将添加一个新的图层。舞台上在某一时刻所展示的图像，是由当前帧的所有层中的内容共同组合完成的。

图 4-105　“图层属性”对话框

5．舞台

舞台是对作品中的对象进行编辑、修改的场所，也是欣赏动画作品的场地。选择“修

改”|“文档…”(Ctrl-J)菜单项将打开“文档属性”对话框，可以修改舞台尺寸、背景色、帧频、标尺单位等参数。

6．场景

场景(Scene)是为复杂的动画作品而设计的。一个 Flash 作品可以由若干个场景组成，每一个场景都可以是一个完整的动画序列。在播放动画时，场景与场景之间可以通过交互响应进行切换。如果没有交互切换，则在播放作品时，自动按场景 1、场景 2…的顺序播放。

7．元件、实例和库

元件(Symbol)是指在 Flash MX 中创建的图形、按钮或影片剪辑三种类型的对象，它可以是从其他应用程序中导入的图片，也可是用户自己新创建的。新创建的元件会被 Flash 记录在当前文档的库(F11 键)中。Flash 中的三类元件分别为：

- 图形类(Graphics)，标识符为，用于创建静态的图形。
- 按钮类(Button),标识符为，用于创建影片中对鼠标事件(如单击、滑过等)做出响应的交互式按钮。
- 影片剪辑类(Movie Clip)，标识符为，用于创建可独立于主影片时间轴播放的及可重复使用的动画片断。它是一个独立的小电影，可包含交互式控制，音效及其他的影片剪辑实例，也可做成动态按钮。

实例(Instance)是指位于舞台上或嵌套在另一个元件内的元件副本。实例可以与其元件在颜色、大小和功能上有很大的差别。编辑实例不会影响元件，但编辑元件会更新它的所有实例。库中的元件可以多次被拖放到舞台上而生成多个实例，每个实例可以拥有自己的名字和特征。

库用来存放和组织已创建的元件，也可存放声音文件，位图图形和视频。Flash 中附带了许多已经做好的元件，存放在公用库中，可直接调用。另外，还可以使用“文件”菜单中的“导入到库”命令将任何格式的视频文件导入到 Flash 文档中，包括 MPG、DV(数字视频)、MOV (QuickTime) 和 AVI。

在“窗口”菜单中，选择“库”或“公用库”子菜单中的“按钮”、“声音”、“学习交互”菜单项，将打开库面板。

8．组件

FlashMX 的组件(UI Components)是具有已定义参数的复杂的影片剪辑，这些参数在动画文件的创作期间进行设置，同时组件也带有惟一的动作脚本设置方法。利用组件可减少重复模块的设计，提高程序的标准性，使 Flash 的交互程序设计更加迅捷方便。

在“窗口”菜单中，执行“组件”命令在 Flash 主窗口右侧打开组件面板，包含七个用户界面组件，分别是 CheckBox(复选框)、ComboBox(组合列表框)、ListBox(列表框)、PushButton(按钮)、RadioButton(单选按钮)、ScrollBar(滚动条)和 ScrollPane(滚动板)。

组件可用于各种功能，如界面元素、客户端/服务器交互以及音频/视频对象。Flash MX 包含所有最常用的操作系统界面元素。每个组件都可以单独使用，也可以组合使用，在不

影响易用性的情况下可完全自定义外观和感觉，创建一个完整的人机交互界面。

9．Action Script 脚本语言

Action Script(AS)脚本语言与 JavaScript 脚本语言十分相似。在 Flash 中利用 ActionScript 可以使 Falsh 具有程序功能，可以操作对象、响应外部事件、创建交互动作，从而创造高质量动画影片及动态交互网页，如复杂的游戏、动画聊天室、多媒体教学系统等。

4.7.4　Flash 文档分析

学习 Flash，最好的办法莫过于分析一个简单而又完整的 Flash 作品了，它是创作的基础。下面通过 Flash MX 本身的一个示例文档，来简单地剖析 Flash 动画的构成，为用户开始 Flash 创作建立一个宏观的认识。分析一个 Flash 文件，一般包括查看对象属性、查看时间轴和舞台、查看库，以及使用影片浏览器。

选择“文件”|“打开…”菜单项，定位到 FlashMX 安装文件夹，打开 Tutorials/FlashIntro/stiletto.fla，如图 4-106 所示。

图 4-106　stiletto.fla 文档

现在就可以在创作环境中查看完成的教程影片了。

(1) 要查看主时间轴中的所有图层，可上下拖动舞台和“时间轴”分割栏，或者拖动“时间轴”区域的垂直滚动条。

单击图层区中的显示/隐藏所有图层按钮，关闭所有图层的显示，然后，单击某个帧、单击某个图层上图层名称后面的，来显示一个帧中某一个图层上的内容。

(2) 查看文档属性

如果属性检查器没有打开，选择“窗口”|“属性”菜单项，打开属性面板。属性面板显示当前帧、或当前对象的属性。在舞台中，用箭头工具选择某个对象，在属性面板中显示该对象的属性。

(3) 查看库

“库”面板包含文档中的元件以及导入对象。如果“库”面板没有打开，选择“窗口”|“库”菜单项打开库面板，如图 4-107 所示。

单击 view1.png 菜单项，在“库”面板的顶部预览区域中查看该图像。

依次展开“库”面板中的其他文件夹，查看文档中包括的按钮和影片剪辑等。

查看完之后，关闭“库”面板。

(4) 用影片浏览器分析影片结构

影片浏览器有助于排列、定位和编辑媒体。影片浏览器通过分层树结构提供有关影片的组织和流的信息，这在分析别人创作的影片时特别有用。

如果影片浏览器还没有打开，选择“窗口”|“影片浏览器”菜单项，打开“影片浏览器”窗口，如图 4-108 所示。

图 4-107 stiletto.fla 库面板

图 4-108 “影片浏览器”窗口

检查该列表，可以看到影片中包括的一些资源，并了解它们之间的关系。

选择“显示帧和图层”过滤按钮。向下滚动到“元件定义”类别。展开该类别，双击“Car Animation 影片剪辑”。即可进入影片剪辑的元件编辑模式。在影片浏览器中，选择并展开“Car Animation”类别，展开“View 3 Fade”图标，然后双击“Frame 60”。

在影片剪辑的时间轴中，播放头会移动到 View 3 Fade 图层的第 60 帧处。如图 4-109

所示。

图 4-109　修改后的效果

(5) 文档分析结束后，关闭文档。

4.7.5　Flash 动画创作基础

了解了 Flash 的有关概念后，就可以使用 Flash 进行动画创作了。动画创作不是单纯的软件技巧问题，它和创作人员的艺术素养、艺术感知等许多因素有关。

1．图形绘制

在 Flash 中，图形的绘制和 Photoshop 类似。在主窗口的左边显示工具栏，包含了大量的绘图工具按钮，如图 4-110 所示。

单击某个绘图工具按钮，在主窗口的下面显示相应绘图工具的属性。

有的工具按钮，在工具栏下部还显示相应的选项按钮。即某种工具按钮实际上可以是有几种形式。例如单击套锁工具，在选项中显示三个可选工具按钮：魔术棒、魔术棒属性和多边形模式。

下面介绍几个特别工具的用法：

- 箭头工具，Flash 中，箭头工具是一个特别的工具，它和选取工具不同。后者单击对象时，整个对象被选取。而箭头对象可以选取两个节点之间的一段，对这一段移动(单击)或者变形操作。
- 钢笔工具，使用钢笔工具可以精确地画直线或光滑的曲线，可以给出线段的节点或分段，可以将直线变为曲线，也可以调整节点的斜率来改变其弯曲度。在

舞台上选取画线的位置单击一下，确定了线段的第一个点。然后在不同的位置依次单击，则绘制出一条连续的曲线。

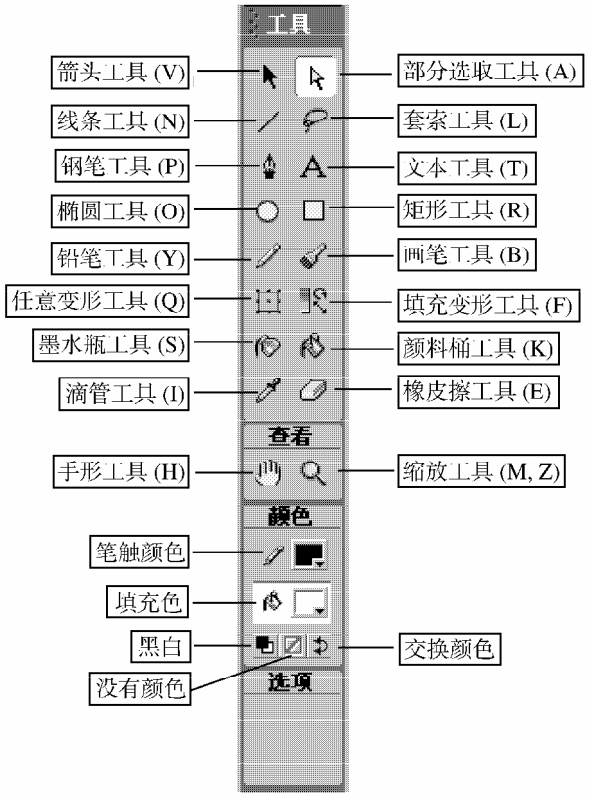


图 4-110 Flash 绘图工具栏

- 颜料桶工具，是用填充色填充一个封闭的图形。如果图形有空隙，颜色将溢出。因此，在 Flash 中选择颜料桶工具，则在工具栏下面的“选项”区域，显示一个空隙大小按钮，对应“不封闭空隙”、“封闭小空隙”、“封闭中等空隙”和“封闭大空隙”几个选项，如果发现颜色溢出，应该撤销上次填充，然后选择“封闭小空隙”等，尝试是否可以使填充不再溢出。

工具栏按钮的使用和其他的绘图软件类似，在此不再重复。

需要说明的是：

(1) 在 Flash 中，当两线条相交、两图形重叠时，线条或图形会被分割开来。例如在舞台上画两条颜色不同的相交的线，鼠标点选不同部分并拖动，会发现线条已被切割，可以移动、修改每个单独的部分。但若绘制的两个图形的颜色相同，则当两图形重叠时会合为一个图形。

(2) Flash 中，图形通常由外部笔触线条(轮廓线)和内部填充色两部分组成，这两部分是相互独立的对象，单击鼠标可以进行单独操作，双击鼠标可选中两者。要去掉图形边线，可用鼠标单击边线并删除，或先选择笔触颜色再选择无色选项(可使用调色板上的按钮)。

2．输入文本

单击文本工具按钮A，在主窗口的下部显示文本工具属性，如图 4-111 所示。

图 4-111　文本属性

在 Flash 中，文本的类型分成三种类型：

- 静态文本，显示静止不变的文本。
- 动态文本，显示动态更新的文本，“动态文本”类型，可以拖出一个文本框，在文本属性面版中，将显示单行、多行文本选择，以及文本变量等。
- 输入文本，接受用户输入的文本，如接受来自键盘的输入。

在文本对象属性面板中，首先设置文本的相关属性，例如选择字体、字号、颜色、对齐方式、段落属性等。然后再开始输入，如果是输入一串字符(汉字)，作为一个对象，可以按 Ctl+B 组合键将文本串对象打散，得到单个的汉字。

3．对象及其操作

在 Flash 中，用户绘制的直线、圆和矩形等，都称为对象。在舞台上绘制的图形往往是由多个对象组合而成。在 Flash 中，可以对对象进行移动、变形、对齐、组合和打散等操作。从而制作出更富有变幻性的图像。

(1) 对象的组合与分离

选择工具后，单击一个对象，则选择该对象，如果要选取分离的多个对象，需要按住 Shift 键。

选择多个对象后，可以通过“修改”菜单中的“组合”菜单项(Ctrl+G)进行组合，成为一个组。要取消组合，使用 Ctrl+Shift+G 组合键。如图 4-112 和图 4-113 所示。

对于多个对象，可以使用“窗口”菜单中的“对齐”命令进行调整。也可以使用光标移动键↑、↓、←和→来移动被选对象的位置。或者直接修改对象属性中的(x,y)坐标。

(2) 对象的缩放、旋转

用箭头工具选中对象，然后选择工具箱中的自由转换工具，此时选中的对象周围会出现八个黑色控制点，如图 4-114 所示。

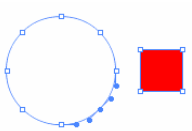


图 4-112　选择多个对象

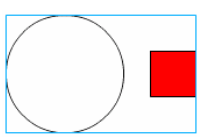


图 4-113　对象的组合

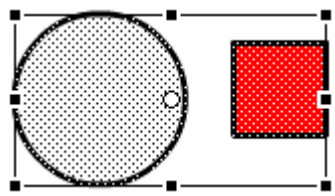


图 4-114　选择对象

沿着控制点的方向拖动小块，即可缩放对象。

旋转操作和缩放对象类似，鼠标移到控制点外面会出现一个弧线箭头，顺着箭头的方向拖动就可以旋转，旋转时要绕着中心拖动。

对象一般是绕着中心旋转的，若想改变对象的旋转中心点，可在选中对象后选择自由转换工具，则对象的中心会出现一个小圆圈，用箭头工具拖动小圆圈就可以改变位置。

4．创建元件与实例

元件(symbol)在 Flash 作品中起着重要的作用，可以将元件拖放到不同的帧中重用。要创建元件，可以通过“插入”菜单中的“新建元件…”菜单项来完成。

另外，也可以将对象转换成元件。从工作区中选取对象，右击，在快捷菜单中选择“转换为元件”，会弹出“创建新元件”对话框，如图 4-115 所示。

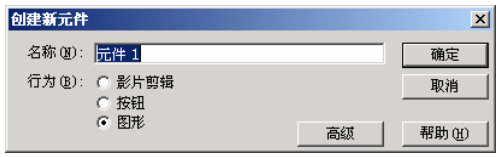


图 4-115 “创建新元件”对话框

定义元件的名称和类型，单击“确定”按钮，进入元件编辑状态。组件做好后，单击时间轴左下角的“场景 1”回到主场景。按 Ctrl+L 组合键打开“库”窗口，可以看到新建好的元件。

元件建好后就可利用它来创建实例。方法是在时间轴上选择一个图层，选取一个关键帧，打开“库”面板，选中元件，把它拖到编辑区。这是一个最简单的实例，实例与元件一样。重新编辑元件，会发现实例也跟着变化；改变实例则不影响元件。

在实例上右击，可选择缩放、旋转等操作。

注意：只有按 Ctrl+B 组合键把实例分离后才能编辑实例。

5．层、引导层和遮罩层

在 Flash 的图层面板中，在“插入”菜单中，选择“图层”菜单项或单击图层区域的插入图层按钮，可以插入一个图层。右击图层，打开“图层属性”对话框，可以将图层重命名，或者改变图层的类型为引导层、遮罩层等。

简单来说，引导图层(guide layer)是用来摆放对象运动路径的图层，它的作用在于确定了指定对象的运动路线。比如用来让一个球按指定的路线移动。

遮罩层(mask layer)用来制作一些特殊的效果。

引导层与遮罩层均可控制它下面的层，除非遇到新的引导层或遮罩层。

4.7.6 动画创作

学习了基本的创作工具的使用后，就可以进行动画作品的创作和发布了。创作环境中的动画作品的文档扩展名为.fla，已发布的 Flash 影片的扩展名为.swf。

1．动画的分类

动画是一帧一帧构成的，每一帧都有一幅图片。在 Flash 中，动画可以分成逐帧动画与关键帧动画两大类。

(1) 逐帧动画(frame by frame)

逐帧动画要求制作者在影片的每一帧中逐帧绘制动作连续的图像以形成动画，我们所看到的动画片大都是以这种形式制作的。制作这种动画的工作量非常大，但能够表达出复杂的动作变形。

(2) 关键帧动画——补间动画

关键帧动画又称补间动画(Tweened Animation)，它是通过内插的方法在两关键帧之间自动生成渐变的中间帧。制作一个物体移动或者形变的动画时，可以定义起始帧和终止帧的图片，中间帧的图片可以由 Flash 系统自动补齐，这样的动画就是补间动画。关键帧动画不适合表现较复杂的变形(比如走路)。

补间动画又分成移动补间动画和形状补间动画两种。补间动画比逐帧动画更小，更便于网上使用。

除此之外，在 Flash 中，根据动画的效果和制作技术，特别是层的使用，还有遮罩动画、引导层动画、交互动画等。

2．补间动画创作举例

补间动画是 Flash 的重点。以制作一个滚动的圆为例，详细步骤如下：

(1) 建立新动画文档

选择“文件”|“新建”菜单项，新建一个影片文件，默认名为“未命名-2”。

(2) 设置舞台属性

选择“修改”|“文档…”菜单项，打开“文档属性”对话框，如图 4-116 所示。

通过“文档属性”对话框，可以修改播放速度(帧频)、舞台尺寸、背景颜色以及选用何种标尺单位等。在这里设置一个尺寸为 300×100 的白色背景舞台，其他保持默认值不变。

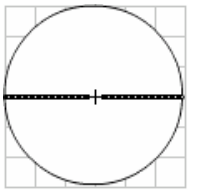
(3) 创建新图形元件

选择“插入”|“新建元件…”菜单项，打开“创建新元件”对话框。

给新建的元件起一个合适的名字，这里命名为 circle，选择行为为图形。然后单击“确定”按钮。进入创建新元件编辑状态，在舞台中央显示一个十字标记 ＋。

为了使图形绘制位置精确，选择“查看”菜单中的“标尺”和“网格”菜单项，在舞台中显示标尺和网格。

图 4-116　“文档属性”对话框



4-117　选中直径

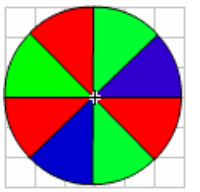


图 4-118　变形并填充

在绘图工具栏中选椭圆工具，在属性面板中设置椭圆工具的属性参数。然后在工作区里按住 Shift 键拖动鼠标绘制一个正圆。用箭头工具选中对象，用键盘的移动键调整圆位置，使得圆心和舞台中心的“+”对准。

利用直线工具画一个圆的直径：使用箭头工具选中此直径(不能使用部分选区工具，因为圆和直径的绘制颜色相同，两者相交后成为一个对象，见图 4-117)。然后，选择“窗口”|“变形”菜单项，打开“变形面板”。

在“变形面板”中输入旋转的角度参数旋转角度设 45 度，连续单击右下角的“旋转复制”按钮三次，就会出现四条直径，把圆分为八叶。在图层 1 上，按显示轮廓按钮。

选择颜料桶工具，选取填充色，用不同的颜色分别填充八个叶，如果出现颜色溢出，在工具栏的下面的“选项”区域，选择“封闭中等空隙”重新填充即可。完成后的元件如图 4-118 所示。

组件做好后，单击时间轴左下角的“场景 1”回到主场景。

(4) 建立运动动画

① 选择“窗口”|“库”菜单项(F11)，打开“库”面板，如图 4-119 所示。可以看到新建的元件。在库窗口中把该元件拖动到工作区的中央，此时时间轴的第 1 帧上的小圆圈由空心变成实心，表明该帧不再为空，成为关键帧。

② 设置动画的目的位置。在时间轴第 15 帧单击鼠标左键，该帧变蓝表示被选，在此处按 F6 键插入一个关键帧，第 15 帧也变为实心小黑点。第 15 帧之前的帧显示为静态帧。

③ 设置帧间的过渡。右击时间轴上的第一帧，选择“创建补间动画”或单击第一帧，并单击图层 1，出现“帧属性”面板，在“补间”后面的下拉列表中，选择“动作”，则下面出现“旋转”下拉列表，选择“顺时针”，第一帧属性设置如图 4-120 所示。

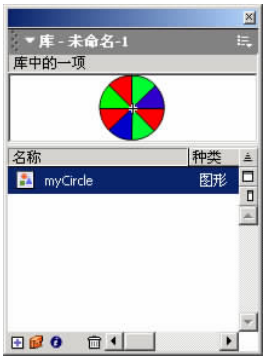


图 4-119 “库”面板

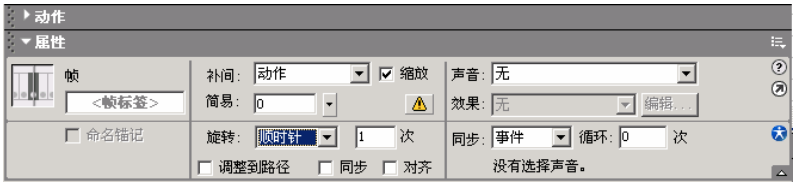


图 4-120 “帧属性”面板

元件运动建好后，在时间轴的第 1 帧和第 15 帧之间出现了一个实线箭头，且背景变成淡紫色，表示这两帧之间有一段运动渐变动画。如果两帧之间出现虚线，表示过渡不成功，右单击虚线帧，在快捷菜单中，执行“删除补间”，即可删除，然后需要重新查看每一个关键帧及其属性。

(5) 测试效果

选择“窗口”|“工具栏”|“控制器”菜单项，打开控制器面板。在控制面板中，按播放钮播放动画，或者按下 Ctrl+Enter 组合键，以全屏方式预览动画。

可看到彩色的圆按照顺时针方向转动，同时时间轴上的红色竖线(时间指针，也称播放头)从第一帧移动到第 15 帧。

(6) 向前滚动

在时间轴上，单击第一帧，用部分选取工具将元件放到舞台左边。再选择第 15 帧，把元件放到舞台右边。

重新测试，则出现滚动效果。

如果希望元件向前移动，可以点一下时间轴的第一帧，在帧属性面板中去掉旋转，就会出现移动而不是滚动的动态效果。

在补间动画创作中，还可更改中间帧对象的大小、颜色等属性以创建更精彩的动画效果。需要注意，这些修改都只能在关键帧上进行，因此，必要时需要插入一个关键帧。另外，还要确保图层不处于锁定状态。

在时间轴上单击某个关键帧，在舞台中显示元件，单击该元件，则显示元件实例属性，如图 4-121 所示。

图 4-121　元件实例属性

- Alpha 变化　在属性面板中，选择“颜色：Alpha”，将百分值调整为 0%，则圆完全透明，产生对象在运动过程中渐渐消失在背景中的效果。
- 大小的变化　在属性面板中，可以修改关键帧中元件实例的大小，产生对象大小变化的效果。
- 速度的变化　单击时间轴第一帧，点图层 1，在属性面板中，将“简易”所对应的指针拖动到最底/顶端，或在输入框中输入?100/100，圆将做变速运动。

(7) 添加引导层

在时间轴面板的图层区域，单击插入图层按钮，插入一个新的图层 2。然后右击该图层，在图层属性窗口中将图层类型修改为“引导层”。

然后用绘图工具栏中的铅笔工具在图层 2(引导层)中绘制一条图层 1 中圆的运动路径(导引层中的路径，在实际播放时不显示)。路径的起点必须与被引导的物件的中心点相重合(单击被导引元会出现一个“+”号，即中心点)。

然后，单击 15 帧，在图层 1 中，将圆拖移到舞台的右边，让圆的中心点与导引线的尾端重合。

在图层列表中，修改图层 1(被引导层)的属性，将类型由“正常”改为“被引导”，被引导层必须列在引导层的下面。完成后的工作区如图 4-122 所示。

重新测试上述的动画，可以看到圆沿着曲线从左向右滚动。

(8) 保存文件

当 Falsh 动画制作完成后，选择“文件”|“保存”菜单项，在弹出的对话框中输入文

件名，单击“确定”按钮。这时系统会把源文件用.fla 格式保存。

也可以在“控制”菜单中，选择“测试影片”菜单项，则系统会自动生成.swf 的电影文件，并存储在.fla 文件相同的目录中。

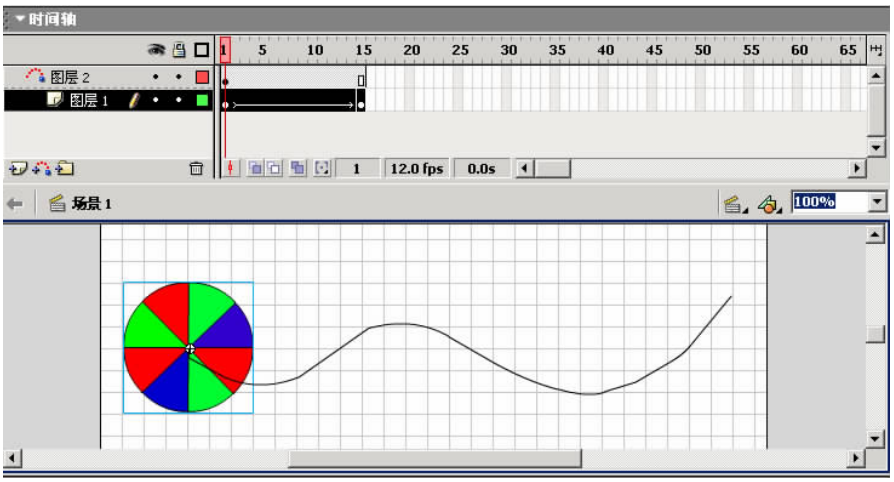


图 4-122 完成后的工作区

3．形变补间动画创作实例

形变补间动画(又称变形动画)可以在两个关键帧之间制作出变形的效果，让一种形状随时间变化成另外一种形状；还可以对形状的位置、大小和颜色进行渐变。

变形动画有两种形式，一种是不可控的(也叫简单变形)，而另外一种则是可控的(通过添加控制点来实现，也叫控制变形)。

下面是一个简单变形动画的制作过程，做一个红色的圆形逐渐变成一个红色的心形。

(1) 新建一个 flash 文档，通过插入菜单，新建一个元件 bluesquare。用绘图工具完成元件 bluesquare 的绘制。同样的步骤再新建一个元件 readcircle。

(2) 元件制作完成后，单击“场景 1”按钮进入场景开始布置。让这个蓝色矩形先停留 10 帧左右，然后开始变形，最后完全变成红色的圆，变形完成后红色的圆延续 10 帧。

在第一帧处，将元件 bluesquare 从库窗口中拖入工作区偏左位置。

既然蓝色矩形停留 10 帧而无变化，那么前 10 帧就不用设置动作了。在第 10 帧处插入关键帧，并将它作为向红色圆变化的起始关键帧。

(3) 蓝色矩形变化为红色圆的过程将延续 20 帧，因此在第 30 帧处插入关键帧，然后将此帧中的蓝色的矩形清除，并加入元件 readcircle。

说明：Flash 中在时间轴的下面有一组专门的多帧编辑与对齐模式按钮，即“洋葱皮”(Onion Skin)效果，可以灵活使用。

(4) 变形完毕后，红色圆会延续 10 帧而无变化，因此在第 40 帧处插入一个过渡帧。

(5) 现在来处理从第 10 帧到第 30 帧的变形动画。先回到第 10 帧，选中组件，按 Ctrl+B 组合键或使用菜单命令“修改”(Modify)、“打散”(Break Apart)，将所选元件打散。

然后再到第 30 帧处，将该处的 readcircle 元件打散。

由于变形操作不支持组件或群组对象，如果不打散而进行变形操作的话，变形将不能成功，而且会弹出警示框。用箭头工具单击元件(内部，不能是边缘)，可以看到打散前与打散后的区别，如图 4-123 所示。

回到第 10 帧，单击图层 1，在帧面板中，在补间后面的下拉列表中，选择“形状”(shape)。最后时间轴状态如图 4-124 所示。

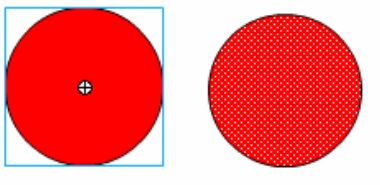


图 4-123　元件打散前后的区别

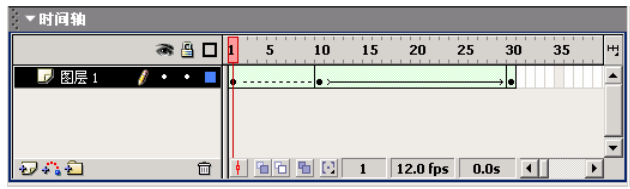


图 4-124　时间轴

进行测试，可以看到变形成功。此时，在时间轴上单击第 10 帧~30 帧之内的帧，可以看到 Flash 生成的补间帧图片。

最后，请记住形变补间动画的关键之处在于打散(Break Apart)。

4．制作按钮

在三种类型的元件中，按钮比较独特，它是制作交互性动画的重要对象。插入新元件，选择“按钮”，进入元件编辑状态下。此时时间轴上会显示四个帧，如图 4-125 所示。

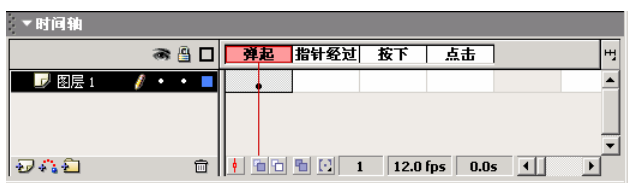


图 4-125　插入按钮元件

- 弹起帧　表示鼠标不在按钮上时的状态，也是一般情况下按钮的状态。
- 指针经过帧　表示鼠标位于按钮上面时的状态。
- 按下帧　表示单击鼠标后尚未释放时的状态。
- 单击帧　定义对鼠标做出反应的区域，这个反应区域在影片中是看不见的。

下面我们介绍按钮的制作步骤：

(1) 插入一个新元件，类型设为“按钮”，名称为 mybtn，时间轴显示如图 4-125 所示。

右击“弹起”帧，在快捷菜单中选择“插入关键帧”菜单项。

利用绘图工具绘制按钮的“弹起”帧图像，此时“弹起”帧下方出现实心的黑点，表示“弹起”帧已有内容。

(2) 右击“指针经过”帧，在快捷菜单中选择“插入关键帧”菜单项。

显示“弹起”帧的内容并修改，得到“指针经过”帧的图像。

(3) 重复步骤(2)，分别为“按下”帧和“单击”帧添加内容。

(4) 调整按钮在四个帧中的大小和位置。

如果按钮在四个帧中的位置不一致，会得不到应有的效果。单击某个帧，然后用箭头工具选择该帧中的对象，显示对象属性面板，如图 4-126 所示。

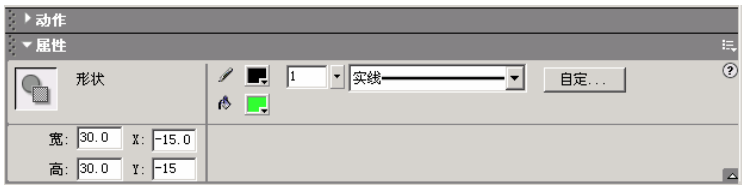


图 4-126 元件属性

输入元件的宽和高，然后输入元件在舞台中的位置坐标，中心点“+”的坐标为(0,0)，元件的中心点是宽和高的一半，将元件移动到指定的位置。如果元件的大小不同，输入元件的高和宽将导致圆渐渐变形，此时需要使用光标移动键↑、↓、←和→，移动元件，使得元件中心和舞台中心重合。

用上面方法将四个帧中对象的大小和位置设置为相同。

在时间轴中，单击“编辑多个帧”按钮，可以看到各个对象的大小和位置一样，和舞台的中心点重合。

这样一个按钮元件就完成了，如图 4-127 所示。

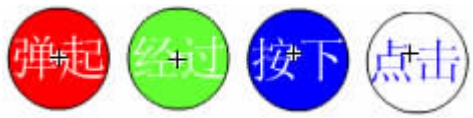


图 4-127 按钮 mybtn 对应的四个元件 btn1~btn4

(5) 使用按钮。单击“场景 1”，回到工作区。单击第一帧，然后打开“库”面板，将按钮元件 mybtn 拖到第一帧编辑区。

(6) 测试。按 Ctrl+Enter 组合键测试按钮。看一下鼠标在、单击和离开时按钮的变化。

(7) 为按钮添加声音。我们可以对按钮增加一些声音效果，例如当鼠标经过或者按下按钮时发出一点响声。方法是：

右击按钮元件 mybtn，在快捷菜单中选择“编辑”菜单项，进入实例编辑。此时，插入一个新的图层 2，选择“窗口”、“公用库”、“声音”菜单项，从“库-声音”面板中间把某个声音拖放到图层 2 中相应的帧。

4.7.7 脚本语言与交互动画

在 Flash 中，动画的交互是通过 ActionScript(AS)脚本语言来完成的。和一般的脚本语言一样，AS 有自己的语法、变量、函数、语句等。

在 F1ash 中，脚本可分为帧和元件两类，其中元件又分为影片剪辑和按钮两种。因此，可以在帧、影片剪辑和按钮元件上添加脚本。

1．增加脚本与动作面板

在 Flash 中，要增加脚本，可以按照下面的步骤完成：

(1) 选择帧、图层、要添加脚本的元件(影片剪辑或按钮)。

(2) 选择“窗口”|“动作”菜单项，打开“动作”面板，图 4-128 所示是选择一个按钮元件后的图动作面板。

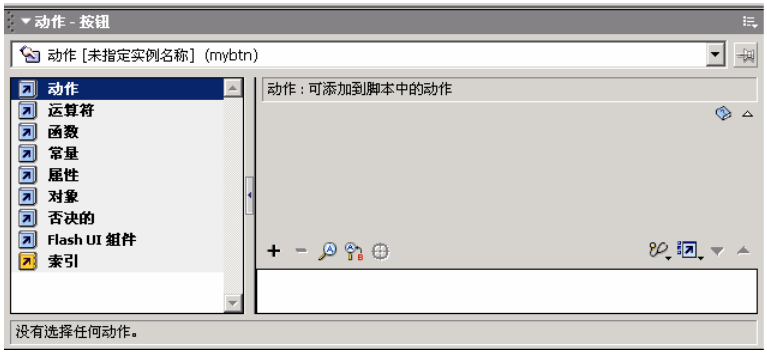


图 4-128　按钮元件对应的动作面板

- 将新项目添加到脚本中。
- 删除所选动作。
- 查找。
- 替换。
- 插入目标路径。
- 调试选项，有设置断点、删除断点等。
- 使图选项，分为标准模式、专家模式、查看行号三种。

“标准模式”是最适合初学者的操作模式，它会随时引导用户输入所需的程序参数，并告诉用户脚本的作用。在“标准模式”，双击左侧所选择的“动作”即可将脚本添加到右边的窗体中，而不必手工输入(仅需在面板中输入少量的参数)。使用“专家模式”，可以直接在右边的脚本窗体中编写程序，该方式适合编写大段的程序。

2．动作

动作是指用 ActionScript 编写的、当特定事件发生时执行的一组命令。可以触发动作的事件是播放到某帧、单击按钮，或用户按键。

在动作面板的左窗格，单击“动作”项，显示动作控制列表，每一类下又有许多控制语句，如图 4-129 所示。

图 4-129　动作面板

主要的语句有：

- gotoAndPlay(["场景号",]"帧号")　跳到一帧或一个场景。
- getURL ("URL")　超链接连到一个 URL。

- play(),stop() 播放或停止动画。
- ifFrameLoaded (帧号) 仅当加载特定帧时才执行动作。
- loadMovieNum ("URL", 图层) 从 URL 加载影片剪辑。
- unloadMovieNum (图层) 卸载用 loadMovie 加载的动画。
- stopAllSounds () 停止声音的播放。
- toggleHighQuality () 调节画面质量。
- tellTarget ("动画实例名") 调用另一个动画。

3．事件与响应函数

当使用鼠标或键盘操作时，将触发特定的事件，对于事件可以编写对应的事件处理函数，完成特定的动作。

添加事件处理函数，方法如下：

(1) 选择一个按钮对象，打开动作控制面板。

(2) 动作控制面板中，打开“动作”|“影片控制”子菜单，双击“on”条目，在动作面板的右侧显示事件列表，如图 4-130 所示。

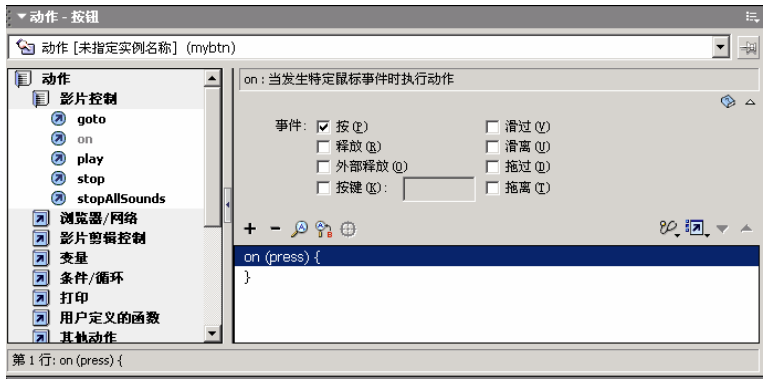


图 4-130 增加按钮对象的事件处理函数列表

例如，选择“按”复选框(Press)，则自动生成 onPress()函数，内容为空。此时，在左侧选择 goto，则在 onPress()中添加代码，最后生成代码如下：

```
on (press){
   gotoAndPlay(1);
}
```

对于“影片剪辑”类的元件对象，其事件、事件响应函数类似，在此省略。

4.7.8 综合举例

在网上经常看到一些幻灯动画(slide show)，例如 Yahoo 中的“焦点新闻”版块中“幻灯集锦”栏目。Flash 幻灯已经成新闻报道的一种重要形式。运用 Flash 技术，可以在推介图片时产生“推”、“拉”、“摇”、“移”的镜头效果，通过将一些静态画面组合起来，

使之具有类似视频的效果。

那么幻灯是如何实现的呢？下面介绍 Flash 幻灯的制作方法。

1．素材准备

几幅要展示的图片，分别命名为 pic01.jpg 和 pic02.jpg。

2．动画制作

(1) 新建一个 flash 文档，设置文件属性参数。

(2) 将图片、声音或动画导入到库中。

选择“文件”|“导入…”或者“导入…”菜单项(前者将导入对象一并添加到当前关键帧)，打开“导入到库”对话框，选择要导入的文件。然后打开“库”面板，可以看到导入文档。如果导入的是 GIF 动画，Flash 会自动生成相应的电影夹子(保存 gif 中的图片)。

3．制作按钮

选择菜单“插入”|“新建元件”，制作四个按钮：“开始”(play)、“停止”(stop)、“前一张”(prev)和“后一张”(next)。通过“窗口”|“库”菜单项打开“库”面板，可以看到制作了四个按钮。

4．制作第 1 帧内容

步骤 3 完成了按钮的制作。在“时间轴”面板中，单击“场景 1”回到文档设计舞台。将 10 幅图像拖放到不同的帧中。

为了管理和说明方便，在图层区双击图层 1，将名字重命名为 pic。同时新建一个图层，命名为 btn，存放按钮。下面以第一帧的制作过程为例进行介绍。

(1) 单击第一帧，打开“库”面板，将图片 pic01.jpg 拖放到图层 pic 中。用箭头工具 单击图片，显示属性面板，修改位图的(x,y)坐标，使得图片在舞台的中央。

(2) 选择图层 btn，在该层上放置按钮。将库中的四个按钮 play、stop、prev 和 next 拖放到第一帧的 btn 图层。并用光标移动键调整各个按钮的位置。

(3) 添加脚本代码

用户希望在观看幻灯时，通过单击按钮可以前后观看不同的图片。即使用按钮从一个场景跳到另一个场景。

为了保证页面在屏幕上不抖动，为每一帧都加 stop 命令，可以加到 btn 图层。方法是：

① 单击第一帧，单击图层 btn，显示帧属性面板，在帧属性面板的右侧，单击“编辑此对象的脚本动作”按钮，打开动作面板。

② 在动作面板中，进入“动作”|“影片控制”菜单，双击其中的 stop 命令。

接下来，定义每一个按钮的函数。现以 next 按钮的定义为例。定义 next 按钮的 onclick 函数。步骤如下：

① 确定已经选择第 1 帧、图层 btn、next 按钮。此时，显示按钮属性面板，在属性面板的右侧单击“编辑此对象的脚本动作”按钮，打开动作面板。

② 在动作面板中进入“动作”、“影片控制”菜单，双击其中的 on 命令，添加“按”

(press)事件处理函数。然后再次双击 goto 命令，选择转到下一帧类型。

最后，第一帧中 next 按钮对应的代码如下：

```
on(press) {
   nextFrame();
}
```

5．其他帧的制作

第一帧制作完成后，接下来就是其他帧的制作了。其他帧的内容和第一帧类似，只是图层 pic 显示的图片不一样，按钮的动作类似。

由于在每一帧上都设置了 stop 指令，因此一帧放映完成后会停止，不会自动放映下面的帧。接着复制剩余的 9 帧。

在第一帧上右击，选择“复制帧”菜单项，然后分别在分别在第 2 帧，第 3 帧…上右击，选择“粘贴帧”菜单项。

复制完成后，分别修改每一帧的图片，以及最后一帧的 next 按钮，使得转到第 1 帧，实现循环。代码如下：

```
on(press) {
   gotoAndStop("1");
}
```

6．第一次测试

完成后的工作区如图 4-131 所示。

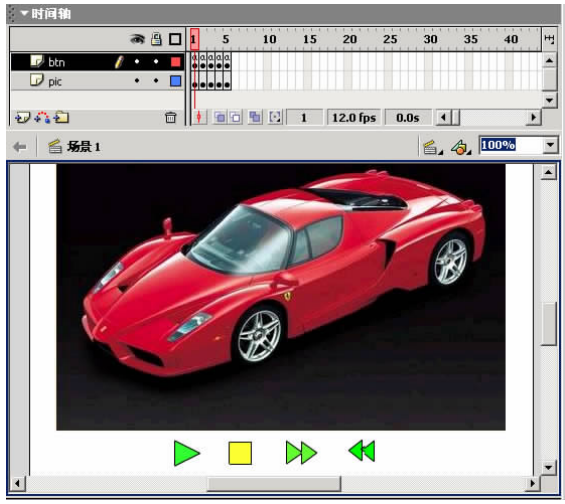


图 4-131　Flash 幻灯的设计

按 Ctrl+Enter 组合键，测试是否制作成功。

7．使用组件

看看上面完成的 Flash 动画，它还比较单调，还需要做以下改进：

(1) 加一个当前图片显示，例如 6/10 表示共 10 张幻灯，目前是第 6 张。

(2) 去掉每一帧的 stop，增加相邻两张幻灯片之间间隔的时间控制。

对于改进(1)，实现非常简单，在每一帧的图层 btn 上。增加一个静态文本，并移动到 play 按钮的前面即可。

对于改进(2)，可以借助于 Flash 本身提供的组件。要使用组件，选择“窗口”|“组件”菜单项(Ctrl+F7)打开 Flash 组件面板。可以将组件拖放的用户的帧、层中。

增加时间间隔控制的具体步骤如下：

(1) 在 next 按钮的后面添加静态文本“速度”。

(2) 然后，选择“窗口”|“组件”菜单项，打开 Flash 组件窗口，将 ListBox 组件拖放到 next 按钮的后面。用箭头工具单击该组件，下面显示组件属性面板，如图 4-132 所示。

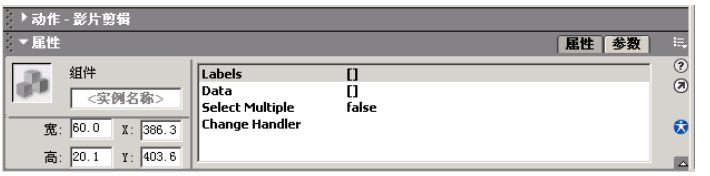


图 4-132　组件属性面板

其中，Labels 是组件的显示名称，可以修改或删除。例如，将一个 button 的 Labels 改成“确定”等。

另外，每个组件还有一个 ChangeHander，单击该项，输入一个用户自定义函数名称，例如输入 onmyselect()。

(3) 编辑用户自定义函数

具体步骤是：

① 确保已经选择组件，单击属性面板右侧的，打开动作面板，选择“专家模式”。

② 进入“动作”|“用户定义函数”子菜单，单击 function，右侧显示函数编辑界面，如图 4-133 所示。

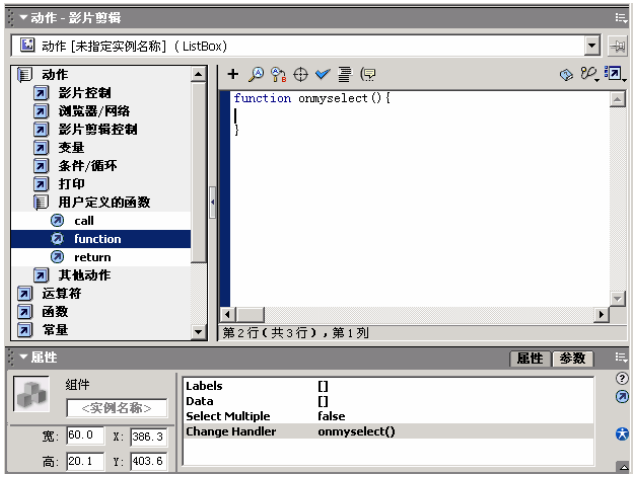


图 4-133　组件属性面板

具体代码在此不再列出，请参考有关 ASP 编程的书籍。

上述修改完成后，继续按 Ctrl+Enter 组合键进行测试。

8．存盘及发布

当 Flash 作品完成后，需要进行存盘和发布。执行“保存”命令，将作品存储为.fla 格式。

在“文件”菜单下，选择“发布设置…”菜单项，打开“发布设置”对话框。进行格式设置，包括选择发布的文件类型(Flash，即.swf，HTML，GIF，JPEG，PNG，EXE，Macintosh，QuickTime 等)，文件名。以及其他 Flash 和 HTML 选项卡的设置。

选择“文件”|“发布”菜单项(Shift+F12)将 Flash 影片发布为.swf 并保存在.fla 相同的目录中。同时生成一个同名的.html 网页文件，可以看到调用该 flash 的 HTML 语句。

习 题 4

1．怎样在 HTML 中设置文本的字体、字号、文字颜色、文字加粗、文字倾斜？

2．怎样在 HTML 文档中设置文本段落的行距和对齐方式？

3．怎样在 HTML 文档中插入图片？<img>标记的常用属性有哪些？怎样将这幅图片 放在居中位置？

4．怎样设置文本或图片的超链接？

5．FrontPage 2000 中不同的显示模式各有什么用途？

6．新建的 FrontPage Web 站点是在本地机上还是在远程的 Web 服务器上？

7．怎样查看已打开站点的站点结构和各种文件的超链接情况？

8．在“图片属性”对话框中可以进行哪些属性的设置？怎样将图片设置成文字环绕的形式？

9．什么是图片的“热点”区域？怎样设置多边形的“热点”？

10．“表格属性”和“单元格属性”有什么不同？怎样将表格中的单元格进行合并或拆分？

11．怎样设置网页的背景图片和背景音乐？

12．使用 FrontPage 2000，规划并建立一个个人网站。

13．简述 Dreamwever 和 FrontPage 的不同。

14．在 Dreamwever 中，层的作用是什么？

15．什么是 Dreamweaver 的行为，如何使用？

16．简述下列概念：

图形、图像、图像处理、图像的分辨率、显示分辨率、打印分辨率

17．在 Photoshop 中，简述图层、通道、路径和图层蒙版的概念。

18．什么是图像的色调、色相、饱和度和对比度？

19．渐变工具包括哪些类型？如何调整图像的尺寸、画布的尺寸？

20．利用 Photoshop 对两幅图片进行合成练习。

21．简述 Flash 功能和特点。

22．解释下列概念：

矢量图、帧、帧频、动画、关键帧、逐帧动画、补间动画

23．简述帧和图层的关系。

24．什么是发布？

25．制作一个测试动画，每一个页面含有一道单项选择题，格式如下。要求用户可以单击一个候选答案，程序将在回答后面显示正确与错误。单击上一题或下一题按钮，可以显示相应的题目。

题干

四个候选答案

回答：

上一题、下一题按钮

第 5 章　客户端开发

网页不仅由 HTML(或 XML)标记构成，往往还包含了大量的脚本程序，它们增加了 HTML 的交互和计算能力,使网页更生动、功能更强大。因此,网页设计除了必须掌握 HTML 和 XML 的各种标记功能外，还应该熟悉脚本语言编程。

脚本语言编程包括两个方面：一方面是客户端编程；另一方面是 Web 服务器端的编程。它们分别在浏览器中和 Web 服务器上执行。本章介绍客户端的编程问题，关于服务器端的编程将在下一章介绍。

5.1　客户端编程与脚本程序语言

客户端程序是利用客户端脚本程序来完成的。脚本语言是介于 HTML 和 Java、Visual Basic 以及 C++等编程语言之间的特殊的语言。HTML 通常用于格式化和链接文本，而编程语言通常用于向机器发出一系列复杂的指令。脚本语言也可用来向计算机发送指令，但它们的语法和规则没有可编译的编程语言那样严格和复杂。脚本语言主要用于格式化文本和使用由编程语言编写的、已编译好的组件。

目前使用较多的脚本程序语言是 JavaScript 和 VBScript，它们是两种对等的脚本程序语言，其中 JavaScript 是客户浏览器端默认的脚本程序语言，而 VBScript 脚本程序语言是 ASP 的默认脚本程序语言。两种脚本程序语言在同一个网页文件内可以混合使用、共用彼此的全局参数。如果需要用非主脚本语言编写程序，应该使用<script>…</script>标记对，并在<script>标记内通过 language 属性指定所用的脚本语言，例如<script language="JavaScript" runat=server >。如果是客户端脚本程序，不需要设置 runat 属性。

5.1.1　脚本引擎

什么是脚本引擎呢？简单地讲，脚本引擎就是指脚本的运行环境，负责脚本程序的解释,具体处理用相应脚本语言编写的脚本命令。例如,ASP 脚本语言必须运行在 IIS(Internet Information Server，Internet 信息服务器)上；Tomcat 是 JSP 和 Servlet 的容器等，也就是运行 JSP 网页必须安装和配置 Tomcat。没有脚本引擎，脚本就不能运行。

在 ASP 结构中，ASP 解释器(ASP.DLL)负责 ASP 页内服务器端脚本程序的解析任务。这需要安装相应脚本程序语言的脚本引擎即脚本程序解释器，来具体处理用相应语言编写的脚本命令，它们以 COM 组件的形式供 ASP 解释器调用。ASP(Active Server Pages)带有两个脚本引擎：Microsoft Visual Basic Scripting Edition(VBScript)和 Microsoft JScript ，其中 ASP 中的默认语言是 VBScript。当安装完 Microsoft NT / 2000 Server 的 Internet 信息服务器 IIS(内含 Active Server Pages)后,VBScript 和 JScript 脚本引擎会自动地安装在服务器上。

如果要使用其他的脚本程序语言，例如 REXX 和 Perl，必须在 Web 服务器上安装相应的脚本引擎。脚本引擎必须遵循 ActiveX 脚本标准并作为一个 COM 对象驻留在 Web 服务器上。

5.1.2　设置主脚本语言

主脚本语言是用于描述在定界符 <% 和 %> 内的命令的语言。Web 服务器管理员可以将任何一种具有脚本引擎的脚本语言作为主脚本语言。主脚本语言可以逐页设置，也可以统一设置所有页的主脚本语言。例如，ASP 默认的主脚本语言是 VBScript。

要设置单个页的主脚本语言，可将 <%@ language %> 指令添加到网页文件的开头，该指令的语法是：<%@ language=ScriptingLanguage %>。如果对某页进行了此种设置，那么该页将忽略在应用程序中对所有页的全局设置。

如果希望将某一种脚本语言设为Web服务器上所有页的主脚本语言，需要使用注册表编辑器regedit.exe程序，修改相应的注册参数。例如，对于ASP，具体的键名为HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Service\W3SVC\ASP\Parameters，在此处填写相应的脚本语言名称即可。

5.2　JavaScript 脚本语言概况

JavaScript 是目前使用最为广泛的脚本语言，它是由 Netscape 公司开发并随 Navigator 浏览器一起发布的，是一种介于 Java 与 HTML 之间、基于对象的事件驱动的编程语言。使用 JavaScript，不需要 Java 编译器，而是直接在 Web 浏览器中解释执行。

JavaScript 语言在早期被 Netscape 的开发者们称为 Mocha 语言，在网上进行 β 测试时，名字改为 LiveScript。Sun 公司推出 Java 之后，Netscape 引进了 Sun 的有关概念，将自己原有的 LiveScript 更名为 JavaScript，它不仅支持 Java 的 Applet 小程序，同时向 Web 开发者提供一种嵌入 HTML 文档进行编程的、基于对象的 Script 程序设计功能。

虽然 JavaScript 采用的许多结构与 Java 相似，但两者有着根本的不同。JavaScript 和 Java 的区别有：

(1) 使用背景不同

Java 是面向对象的程序设计语言(object oriented language)；JavaScript 是一种脚本语言，是一种基于对象的、面向非程序设计人员的编程语言。和 Java 不同，JavaScript 没有提供抽象、继承、多态等有关面向对象程序设计语言的许多功能。

(2) 运行环境不同

JavaScript 源代码无须编译，嵌入 HTML 文档中的 JavaScript 源代码实际上是作为 HTML 页面的一部分存在的。浏览器浏览包含 JavaScript 源代码的 HTML 页面时，由浏览器自带的脚本引擎对该 HTML 文档进行分析、识别、解释，并执行用 JavaScript 编写的源代码。而 Java 则不同，Java 源代码必须进行编译、连接后才能运行。

JavaScript 编程的复杂性是由对应的 HTML 文档所提供的功能决定的，下面是两个简

单的 JavaScript 程序实例。

[例 5-1] 一个简单的 JavaScript 程序。

```
<html>
<title>JavaScript in HTML</title>
<head>
</head>
<body>
   <h1>JavaScript in HTML</h1>
   <script language="Javascript">
     document.write ("Hello!" );
   </script>
</body>
</html>
```

JavaScript 源代码被嵌在一个 HTML 文档中，它可以出现在文档头部(<head>…</head>)和文档体(<body>…</body>)中。Script 标记的一般格式为：

```
<script language="Javascript">
   JavaScript 语句部分
</script>
```

其中，JavaScript 语句可以用分号分开，也可以通过不同行来分开。

[例 5-2] 一个包含 JavaScript 函数的实例。

和所有的程序开发工具一样，在 JavaScript 中不仅包含大量的内部函数，同时也为用户提供了用户自定义函数和调用函数的能力，从而增加 Web 开发的灵活性。

函数的定义与调用同一般的程序设计语言类似，但由于浏览器浏览的 Web 页是顺序地从 Web 服务器调出，并由浏览器解释执行的，函数必须遵循先定义(一般放在<head>…</head>内)、后调用(在<body>…</body>内)的原则。

```
<html>
<title>function in JavaScript </title>
<head>
<script language="Javascript">
function fact(n)
{
   var res=1;
   if (n==0) res = 1;
   else res = n*fact(n-1);
   return res;
}
</script>
</head>
<body>
<p>fact(5) =
<script language="JavaScript">
  //输出 n 的阶乘
  document.write(fact(5));
</script>
```

```
</body>
</html>
```

上例显示结果为: fact(5) = 120。

JavaScript 程序和一般的编译型程序不同，它没有一个由操作系统调用的主程序(如 C 语言中的 main()函数)。一个 JavaScript 函数定义时并不发生作用，只有在引用时(函数定义后的 document.write 语句)才被激活。

5.3　JavaScript 基础

5.3.1　JavaScript 基本符号

任何一种语言都有其自身的字符集和基本符号，它们按照语法构成语言的语句，由语句再构成程序。

1．基本字符

JavaScript 语言的基本字符有字母(a、b、…、z，A、B、…、Z)数字(0、1、…、9)和特殊符号(+、?、*、/、<、=、>等)。

同 C 语言一样，JavaScript 中同样有一些以反斜杠(\)开头、来表示不可显示的特殊字符，通常称为控制字符。

2．关键字

关键字是由字母构成的、具有固定含义的单词，如 var 代表变量说明，if 表示条件语句等。JavaScript 的关键字很多，可以参考 JavaScript 的专门书籍。

需要特别注意的是，关键字要小写。

3．标识符

标识符是表示常量、变量和函数等名称的符号。标识符分为标准标识符和用户自定义标识符。标准标识符是表示标准常量和标准函数等名称的符号。JavaScript 中的标准常量和标准函数见下面的介绍。

用户自定义标识符是指用于说明常量、变量和函数等名称的符号。用户自定义标识符必须以字母开始，由任意的字母、数字和符号_组成。用户在命名标识符时应该有一定的命名规范：

① 用户自定义标识符不应该与标准标识符重名。

② 用户自定义标识符要尽可能反映它所代表的对象的含义。如用 name 代表名称比用 x 代表名称更好。

③ 如果是一个变量名，还应该尽可能反映变量的数据类型和作用域，比如用 nUserID 表示一个用户标识，最前面的小写字母 n 代表变量为整数类型。

④ 大小写混写和较长的标识符可以把含义表达的更清晰。

总之，好的命名规范可以提高程序的可读性，便于程序的维护。

4．注释

为了增加程序的可读性，一般在程序中增加注释语句。注释内容没有语法要求，其内容仅仅是一个提示作用。在 JavaScript 中，注释以字符 // 引导，注释可以单独一行，也可以在语句行的后面。

需要说明的是，与 HTML 的注释(<!---….-->)不同，当在服务器端处理脚本时，JavaScript 中注释将被删除，而不是被送到浏览器。若需要客户端浏览器看到脚本中的注释，应该使用 HTML 注释将注释加进 HTML 页。此时，注释将返回到浏览器。

需要特别注意的是，JavaScript 是识别大小写的，如果编写有误，将显示“ JavaScript 脚本错误 ”的警告。

5.3.2 数据和数据类型

JavaScript 脚本程序语言是一种解释性的程序语言，所写的脚本程序不需要编译和连接，它采用边读边执行的方式运行。在数据类型方面，JavaScript 提供了四种基本的数据类型用来处理数字和文本。

JavaScript 提供的数据类型有数值(整数和实数)，字符串型(用双引号"或单引号'括起来的字符)，布尔型(True 或 False)。

在 JavaScript 基本类型中的数据可以是常量，也可以变量。由于 JavaScript 采用弱类型的形式，因而一个数据的变量或常量不必首先作声明，而是在使用或赋值时确定其数据类型。当然也可以先给变量赋一个初值，来声明该数据的类型。

另外，一个变量的类型在使用时可以改变，例如：

```
var x = "hello";
```

此时，x 为字符串类型，接下来可以给 x 赋一个整数，如 x=100，这样变量 x 就成为整数类型了，而不会和其他语言一样出现赋值不相容的错误。

5.3.3 常量和变量

1．常量和常量定义

常量是指在程序执行过程中，其值不发生变化的量。常量有字面常量和符号常量两种。字面常量就是一些数值或字符串，如 3.14、"hello"等都是字面常量。另外，还可以为常量指定一个名称，常量名是一个用户自定义标识符，例如用 pi 代表圆周率 3.14。

常量命名有两方面的好处，首先恰当的常量名称可以增加程序的可读性；其次，使用常量可以便于程序的维护。例如，可以命名常量 pi=3.14，如果希望提高求解精度，可以修改 pi 的定义为 pi=3.1416，这种修改只在一处进行，不会产生不一致的情况，而且修改简单。

JavaScript 的常量通常又称字面常量，它是不能改变的数据。对于整型常量，其整型常

量可以使用十六进制、八进制和十进制表示。对于实型常量，实型常量是由整数部分加小数部分表示，如 12.32、193.98 。可以使用科学或标准方法表示 5E7、4e5 等。对于布尔值，布尔常量只有两种状态：true 或 false。 它主要用来说明或代表一种状态或标志，以说明操作流程。它与 C++是不一样的，C++可以用 1 或 0 表示其状态，而 JavaScript 只能用 true 或 false 表示其状态。对于字符型常量，是用单引号(')或双引号(")括起来的一个或多个字符，如 "hello"、"5123"、"x+y" 等。另外，JavaScript 还有空值 null，表示什么也没有。如试图引用没有定义的变量，则返回一个 null 值。

2．变量和变量说明

所谓“变量”是指在程序执行过程中，其值发生变化的量。每一个变量都有一个变量名，变量名是一个用户自定义标识符。变量有两个重要的属性，一个是数据类型，一个为操作运算。数据类型决定了变量所占内存空间的大小，也决定了数据的取值范围和操作运算。例如，一个整数类型的变量在内存中占两个字节，能够进行算术运算和关系运算等。字符类型变量占一个字节，能进行关系运算，但不能进行算术运算。

在 JavaScript 中，变量命名的一般形式是：

```
var  <变量名表>;
```

其中，var 是 JavaScript 的保留字，表明接下来是变量说明，变量名表是用户自定义标识符，变量之间用逗号分开。

例如，var x,y；定义了两个变量，名称分别为 x 和 y，没有给出具体的数据类型，也没有赋予初始值。

再如，var myName=”John” 定义了一个变量 myName，同时赋予了它一个字符串值。在 JavaScript 中，变量可以不作声明，而在使用时再根据数据的类型来确定变量的类型。

3．变量的作用域

变量的作用域由声明变量的位置决定，决定哪些脚本命令可访问该变量。在函数外部声明的变量称为全局变量，其值能被所在 HTML 文件中的任何脚本命令访问和修改。在函数内部声明的变量称为局部变量。只有当过程被执行时，变量被分配临时空间，函数结束后，变量所占据的空间被释放。局部变量只能被过程内部的语句访问，只对该函数是可见的，而在函数外部则是不可见的。

5.3.4　表达式和运算符

表达式是指将常量、变量、函数、运算符和括号连接而成的式子。根据运算结果的不同，表达式可分为算术表达式(结果为数值)、字符表达式(结果为字符)和逻辑表达式(结果为 true 或 false)。

JavaScript 提供了丰富的运算功能，包括算术运算、关系运算、逻辑运算和连接运算等。

1．算术运算符

JavaScript 中的算术运算符有单目运算符和双目运算符。双目运算符包括+(加)、?(减)、*(乘)、/(除)、%(取模)、|(按位或)、&(按位与)、<<(左移)、>>(右移)、>>>(右移，零填充)。单目运算符有?(取反)、~(取补)、++(递加 1)、??(递减 1)。

2．关系运算符

关系运算又称比较运算，关系运算符包括<(小于)、<=(小于等于)、>(大于)、>=(大于等于)、=(等于)、!=(不等于)。

关系运算结果为布尔值，如果条件成立，则结果为 true，否则为 false。

3．逻辑运算符

逻辑运算符有&(逻辑与)、|(逻辑或)、!(取反)、^(逻辑异或)。

4．字符串连接运算符

连接运算用于字符串操作，运算符有+(用于强制连接)。

5．?:三目操作符

三目操作符？：格式为：

操作数？表达式 1：表达式 2

其逻辑功能为：若操作数的结果为真，则表述式的结果为表达式 1，否则为表达式 2。

5.3.5 基本语句

任何一种过程式语言编写的程序，都可以分为三种程序结构，即顺序结构、分支结构和重复结构。与此对应的是三种流程控制语句。

1．顺序结构

在过程式的语言中，程序总是从上而下顺序执行的。遇到函数调用，转去执行相应的子程序，执行结束后返回。

对于有若干语句构成逻辑上的复合语句，用{和}括起来，语句之间用分号分开。

2．分支结构

在 JavaScript 中，实现分支结构的语句有三种，它们是条件判断语句 if 语句、if…else 语句以及开关语句 switch 语句。

(1) if 语句

if 语句的一般形式是：

```
if (<布尔表达式>)
```

```
    <语句>;
```

if 语句首先计算布尔表达式的值，若结果为 true，则执行语句部分，否则执行 if 下面的语句。逻辑功能如图 5-1 所示。

(2) if... else 语句

if... else 语句先计算布尔表达式的值，根据计算结果指定要运行的语句。一般形式是：

```
if (<布尔表达式> )
  <语句 1>;
else
  <语句 2>;
```

if...else 语句首先计算条件表达式的值，若为 true，则执行语句 1，否则，执行语句 2。其逻辑功能如图 5-2 所示。

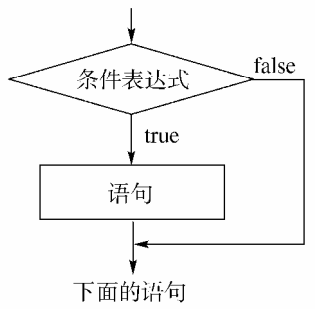


图 5-1　if 语句逻辑功能图

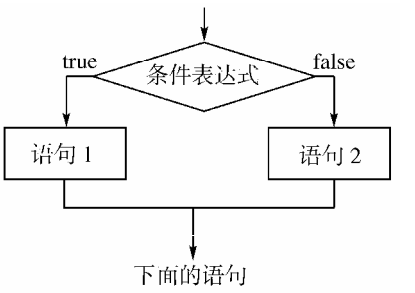


图 5-2　if…else 语句逻辑功能图

通常，条件是使用比较运算符对值或变量进行比较的表达式，if...else 语句可以按照需要进行嵌套。但是嵌套降低了程序的可读性。为此 JavaScript 提供了 switch 语句。

(3) switch 语句

switch 语句提供了 if...else 多层嵌套结构的一个变通形式，可以从多个语句块中选择执行其中的一个。switch 语句提供的功能与 if...else 语句类似，但它可以使代码更加简练易读。一般形式为：

```
switch (<数值或字符串表达式>)
{
   case 表达式 1
        语句 1;
        [break;]
   case 表达式 2
        语句 2;
        [break;]
        ... ...
   case 表达式 n
        语句 n;
        [break;]
}
```

switch 结构在开始处使用一个只计算一次的简单测试表达式。表达式的结果将与结构

中每个 case 的值比较。如果匹配，则执行与该 case 关联的语句块。执行完后，如果希望退出 switch 语句，则需要添加 break 语句，否则，继续下面的 case 匹配。switch 语句的逻辑功能如图 5-3 所示。

3．重复结构

当部分语句需要反复执行时，需要重复语句。重复语句总是由循环体和循环终止条件两部分组成，在 JavaScript 中可使用下列两种形式的重复语句。

(1) while 循环语句

while 循环语句是先判断循环终止条件，然后再执行循环体，因此循环体可能一次也不被执行。while 循环语句的一般形式是：

```
while (<布尔表达式>)
      语句；
```

首先计算条件表达式的值，如为 true，则执行循环体语句，然后无条件地返回 while 语句开始，继续计算条件表达式的值。如果条件表达式结果为 false，则结束循环，执行下面的语句。逻辑功能如图 5-4 所示。

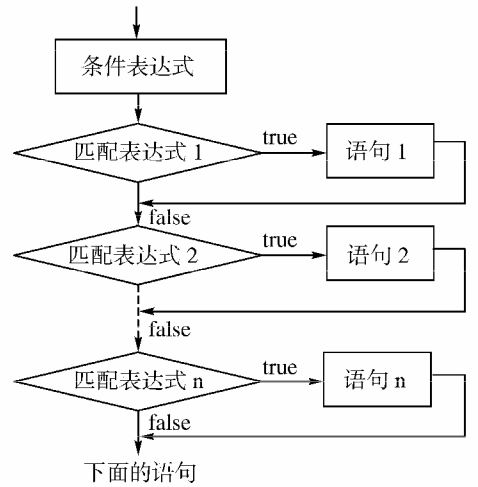


图 5-3 switch 语句逻辑功能图

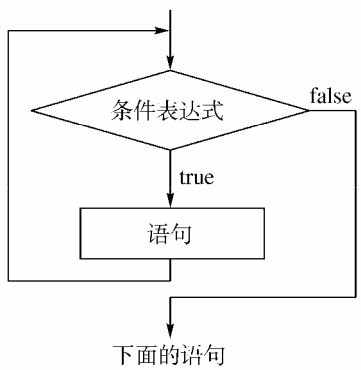


图 5-4 while 语句逻辑功能图

(2) for 循环语句

一般形式为：

```
for (表达式 1；表达式 2；表达式 3)
     语句;
```

执行过程如下：

① 计算表达式 1。

② 计算表达式 2，如果为 true，则执行循环体，然后转步骤③。否则，结束循环。

③ 计算表达式 3。

④ 无条件则转步骤②。

⑤ 循环结束，执行 for 语句下面的语句。

逻辑功能如图 5-5 所示。

另外，循环体的内部也可以包含循环语句，即循环的嵌套，构成多重循环。需要特别注意关键字要小写，例如 for 不应该写成 For。

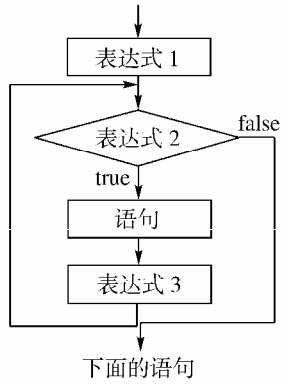


图 5-5　for 语句逻辑功能图

4．break 和 continue 语句

与 C++语言相同，使用 break 语句可使得循环从 for 或 while 中跳出，continue 可使得程序跳过循环内剩余的语句而进入下一次循环。

5.3.6　函数

函数是实现结构化编程的主要手段。它把一个系统中需要反复多次执行的部分定义为一个过程或函数。如判断一个数是否为素数、求两个数的最大公约数等。在 JavaScript 系统中，已经给出了许多标准函数。同时也允许用户自己定义函数。

在 HTML 文档中，函数过程的定义应该在 <script>…</script> 标记内部，<script> 和 </script> 标记对可以出现在 HTML 文档的头部和文档体中。

函数是以 function 开头定义的，由函数头部和函数体构成，一般形式为：

```
function <函数名> (<形式参数表>)
    {
    语句(函数体);
}
```

在函数的定义中，函数名后有形式参数表。这些形式参数变量可能是一个或几个，在 JavaScript 中可通过函数名.arguments.Length 来检查参数的个数。另外，函数体中必须要包含一个返回函数值的语句，即 return 语句。

5.4　事件驱动及事件处理

JavaScript 是基于对象(object-based)的语言。这与 Java 不同，Java 是面向对象(object oriented)的语言。基于对象的基本特征，就是采用事件驱动(event-driven)。鼠标或热键操作称之为事件(event)，事件将激活相应的程序或函数，这些程序或函数称为事件处理程序(event handler)。

JavaScript 中鼠标或热键的动作引发的主要事件如下。

(1) onClick 事件

当用户单击鼠标时，产生 onClick 事件。同时 onClick 对应的事件处理程序或代码将被调用执行。例如可通过下列按钮激活 change()函数：

```
<form>
   <input type="button" value=" " onClick="change()">
</form>
```

在 onClick=后，可以使用自己编写的函数作为事件处理程序，也可以使用 JavaScript 内部的函数。还可以直接使用 JavaScript 的代码等。例：

```
<input type="button" value=" "  onClick="alert('发生 onClick 事件');">
```

(2) onChange 事件

当利用 text 或 textarea 元素进行输入，内容改变时引发该事件。当在 select 列表框中某个选项状态改变时，也会引发该事件。例：

```
<form>
   <input type="text" name="MyAddress" value="RoadNo, City"
          onChange="check(this.value)">
</form>
```

(3) onSelect 事件

onSelect 为选中事件，当 text 或 textarea 对象中的文字被加亮后，引发该事件。

(4) onFocus(获得焦点)事件

单击 text、textarea 或 select 对象时，产生该事件，此时该对象成为当前焦点对象。

(5) onBlur(失去焦点)事件

当 text、textarea 或 select 对象不再拥有焦点时，引发该事件。

(6) onLoad(载入文件)事件

当文档载入时,产生该事件。onLoad 的作用之一就是在首次载入一个文档时检测 cookie 的值，并用一个变量为其赋值，使它可以被源代码使用。

(7) onUnload(卸载文件)事件

当退出 Web 页面时引发 onUnload 事件，并可更新 cookie 的状态。

(8) 鼠标事件和键盘事件

除了上述的一系列针对控件的事件外，还有许多鼠标和键盘事件，常用的有：

- onMouseMove 鼠标移动
- onDblClick 鼠标双击
- onMouseDown 鼠标被按下
- onMouseUp 鼠标被释放
- onKeyDown 键被按下，所按键的 ASCII 码值保存在 window.event.keyCode 中
- onKeyPress 键被按下然后被释放
- onKeyUp 键被释放
- onResize 窗口被调整大小

5.5 对象及其操作

在 20 世纪 90 年代以前，自顶向下、逐步求精的结构化程序设计方法是软件开发的主

要方法，直到现在，这种结构化的程序设计思想仍然被广泛采用。Pascal、C、Basic 和 Fortran 等高级语言很好地实现了结构化编程的思想，通过过程和函数(又称子程序)把一个复杂的问题划分成几个相对独立的、简单的子问题，如果子问题还比较复杂，则继续划分，最后将划分后的每个小问题用过程和函数来实现。

此后，特别是在最近的十几年间，面向对象技术正在对人们近半个世纪来的软件开发思想产生深刻的变革。这一技术强调利用软件对象进行软件开发，它将自然界中的物理对象和软件对象相对应，建立了类和对象的概念。

5.5.1　对象的基本概念

对象(object)是用类来声明的数据结构，如果将类比作数据类型，对象就是相应数据类型的变量。JavaScript 并不是一个完整的面向对象的程序设计语言，它不提供关于对象的抽象、封装及派生等功能。但它可以使用浏览器的内置对象，也可以允许用户自定义对象，从而分享面向对象程序设计带来的好处。

JavaScript 定义“类”和定义“函数”的语法是一样的，而且这样的函数成为该类的构造函数。用户用函数定义来定义类，然后用 new 语句创建该类的一个实例。

在 Javascript 中，类定义的一般形式是：

```
function className(<属性表>)
{
   this.prop1=prop1;           // 属性
   this.prop2=prop2
   ...
   this.meth1=FunctionName1;   // 方法,需要定义对应的函数
   this.meth2=FunctionName2;
   ...
}
```

对象属性和方法的引用主要使用点运算符(.)，一般形式为：

对象名.属性名|方法名

[例 5-3]　定义一个矩形类，它包含三个成员变量，分别是矩形的高、宽和面积，另外包含一个计算面积的成员函数。

```
<html>
<head>
<script language="javascript">
function squArea()
{
 this.result = this.width * this.height;
}
function MyClass(x,y)
{
   this.width = x;
   this.height= y;
   this.result= 0;
   this.area = squArea;
```

```
}
</script>
</head>
<body>
<script language="javascript">
   myObj = new MyClass(5,6);
   myObj.area();
   document.write(myObj.result);
 </script>
</body>
</html>
```

5.5.2 对象的操作

在 JavaScript 中提供了几个用于操作对象的语句和关键字、运算符。

(1) for...in 语句

格式如下：

```
for(对象属性名 in 已知对象名)
```

顺序输出对象各个属性的值，如果是方法，则输出对应的函数代码。例如：

```
function showData(object)
{
    for(var prop in object)
        document.write(object[prop]);
}
```

使用该函数时，在循环体中可以不知道对象属性的个数，for 可以自动将属性取出，直到最后。

(2) with 语句

使用该语句时，在该语句体内，任何对变量的引用被认为是这个对象的属性，从而节省程序代码。一般形式为：

```
with object{
...
}
```

(3) this 关键字

this 是对当前对象的引用，在 JavaScript 中，对象的引用是多层次的，往往一个对象的引用又需要对另一个对象的引用，而另一个对象有可能又要引用另一个对象，这样有可能造成混乱，为此，JavaScript 提供了一个指向当前对象的指针 this。

(4) new 运算符

使用 new 运算符可以创建一个新的对象。创建对象的一般形式是：

```
myobj=new className(参数表);
```

其中，myobj 为要创建的新对象名称，className 是类的名称。

如创建一个日期对象，代码为：

```
myDate=new Date();
myBirthday=new Date("January 18,1973 6:15:00");
```

这样就创建了两个新的日期对象 myDate 和 myBirthday。

5.6　常用内部对象及函数

JavaScript 提供了一些常用的内部对象和函数，用户不需要用脚本来实现这些功能，称它们为内部对象和内置函数。一个程序不是只由简单的语句构成的，往往用到大量的标准函数，还会用到内置对象。要使用一个工具进行软件开发，必须要熟悉该工具所提供的标准函数、内置对象等。

5.6.1　String 对象

在 JavaScript 中，每个字符串都是对象，这类对象不需要像一般自定义对象一样用 new 关键字进行声明。字符串(String)对象封装了 JavaScript 中的字符串以及相关的操作。

字符串对象的创建十分简单，而且是隐式的，不使用 new 关键字，例如：

```
var myStr="Hello";
```

这样，myStr 就是一个字符串对象了，一个变量被声明为字符串对象之后，它就拥有有了这个对象的属性和方法，可以和一般对象一样，使用对象的方法，取得对象的属性。

在一个变量被声明为字符串变量之后，它仍旧可以被赋予数值，这时这个变量就不再是字符串了，而成为一个数值型的变量。例如：

```
var varStr="name and address";
varStr= 1;
```

上述赋值语句使 varStr 成为一个数值型的变量，它不再拥有字符串的属性和方法。

1．字符串对象的属性

字符串对象的属性只有一个，这就是 length(长度)属性，记录字符串的长度。需要说明的是，如果字符串为英文，那么长度属性的值是字母个数加上特殊符号个数，再加上空格数；但是如果字符串是中文，每个中文单字占两个英文字符的长度。

2．查找字符和子串

在字符串的方法函数之中，有一类用于查找在字符串中某一特定字符的所在位置；或者是字符串中特定字符的有无；或者是特定位置的字符值。

(1) charAt()

这个方法返回 String 对象特定位置的字符，一般形式是：

```
StrName.charAt(Position);
```

StrName 是字符串变量名或字符串常量。Position 用来指定希望取得的字符在字符串中的位置，它是大于等于 0 且小于字符串长度（length)的正整数。例如：

```
var myStr="This is a string";
var FiveChar=StrVar.charAt(4);
```

上述语句将取出字符串中的第五个字符，并将它存储于 FiveChar 变量中。需要注意，这个成员方法只适用于英文字符串，对于中文字符串，它将返回一个乱码。因此，要避免在中文字符串中使用这个成员方法。

(2) indexOf()

这个方法返回指定的字符(或字符串)在字符串中的位置，一般形式为：

```
StrName.indexOf(subStr);
```

或者

```
StrName.indexOf(subStr,StartPosition);
```

其中 StrName 是字符串对象名，可以是常量，也可以是字符串变量。subStr 是一个待查找的字符或者字符串，可以是常量，也可以是变量。在省略 StartPosition 的情况下，此函数将从字符串的第一个字符开始查找；当 StartPosition 参数存在的情况下，这个函数将从字符串中的第 StartPosition+1 个字符开始查找；当 StartPosition 超过字符串的长度时，返回?1。

当所希望查找的字符串找不到时，将返回?1。当字符串中有两个以上的待查找字符串，则返回被搜寻字符串中位置在最前面的待查字符串的位置值。

(3) lastIndexOf()

这个方法的使用与 indexOf()方法十分相似，indexOf()是按照从左到右的顺序查找待查找字符串，而 lastIndexOf()是按照从右到左的顺序查找待查找的字符串。

lastIndexOf()也可以指定开始查找的位置，指定的方法和 indexOf()一样。

(4) substring()

返回一个指定的子字符串，它的语法是：

```
StrName.substring(Position1,Position2);
StrName.substring(Position);
```

第一种格式中，返回下标从 Position1 到 Position2?1 的字符串。例如，substring(1，3)返回下标是 1 到 2 的字符，即字符串中的第二和第三个字符。这两个参数的大小关系可以任意，在调用时，取两参数中较小的值作为子串开始的下标值，取较大的参数减 1 作为子串结束的下标值。

第二种格式的 Position 用来指定子串开始的字符下标位置，而子串结束的位置取被搜索字符串的结尾。例如，substring(1)代表取由第二个字符开始到结尾的所有字符。

3．改变字符串大小写

```
StrName.toLowerCase();
StrName.toUpperCase();
```

这两个成员方法分别将字符串中所有的字符变为小写字符和大写字符，将变化后的结果返回。但是，原字符串内容并没有发生变化。

4．形成 HTML 文本格式

网页中常常需要对一个字符串变量进行操作，使它具有一定的 HTML 的特定格式。例如，一个字符串变量定义如下：

```
var aStr = "This is the Discount Table";
```

需要的格式是：

```
<a href="home.html">This is the Discount Table</a>
```

可以利用字符串的 + 操作来将其转化。但是 JavaScript 中提供了以下四种更简单的方法。

(1) anchor() 方法

这个方法用来创建并返回一个字符串，此字符串用以在网页中构造一个锚点。

所谓“锚点”是当网页文档的内容比较多时，一般在这些网页中可以通过单击一些超链接直接跳至此网页的一些特定位置，这些特定位置就是锚点。锚点有自己的名字，这和一个 URL 类似。当希望通过超链接到达锚点时，这个成员方法的语法格式为：

```
StrName.anchor(anchorName);
```

其中，anchorName 是与 HTML 中<a>标识的 name(名字)属性相关的字符串。例如，在 Java Script 中使用如下语句：

```
var aStr=" This is the Discount Table ";
var anchorText = aStr.anchor("Discount");
document.write(anchorText);
```

则 anchorText 的取值是：

```
<a name="Discount"> This is the Discount Table </a>
```

于是，用 document.write()将这个字符串写入到网页中，就有了一个名字是 Discount 的锚点，而这个锚点显示的文字是“This is the Discount Table”。

(2) fontcolor()

创建并返回一个指定颜色的字符串，其语法格式是：

```
StrName.fontcolor(FontColor);
```

其中，FontColor 是一种颜色的名字，或者是一种用 16 进制数表示的颜色。例如：

```
var AStr="Font Color Exmaple";
var FontColorText=AStr.fontcolor("green");
document.write(FontColorText);
```

那么显示的文字应是绿色的，相当于 FontColorText 具有下面的值：

```
<font color=green>Font Color Example</font>
```

(3) fontsize()

创建并返回一个指定大小的字符串，一般形式是：

```
StrName.fontsize(FontSize);
```

其中，FontSize 是数值型的变量或者常数。现在 JavaScript 可识别的字体有 7 种，FontSize 参数由 1 至 7，字体由小到大。当 FontSize 参数大于 7 时，将被当作 7 处理；当其小于 1 时，将被当作 1 处理；这个参数也可以是负数。当参数是?1 时有点特殊，它相当于 FontSize=2。

例如：

```
"FontSize Example".fontsize(4);
```

相当于：

```
<font size=4>"FontSize Example"</font>
```

(4) link()

用来创建并返回一个字符串，此字符串用以在网页中构造一个超链接。它的语法格式是：

```
StrName.link(Href);
```

其中，Href 是超链接的 URL。例如下面的语句：

```
var AStr="A Href Example";
var HrefText=AStr.link("home.html");
```

将使 HrefText 变量具有下面的值：

```
<a href="home.html">A Href Example</a>
```

5.6.2 Math 对象

JavaScript 中的 Math 对象封装了常用的数学常数和一些常用的数学运算，这些运算包括三角函数、对数函数、指数函数和一些舍入函数等。

1．Math 对象的属性

Math 对象的属性与其他对象的属性有一些区别。这些属性是常用的数学常数，它们是定值，因此它们是只读的，不允许对这些对象属性进行写操作。表 5-1 列出了 Math 对象的属性名、描述和近似值，以供参考。

表 5-1 Math 对象的属性

属性名	描述
E	常数 e，或称作欧拉常数，它是自然对数的底，近似值是 2.718
LN2	2 的自然对数，近似值是 0.693
LN10	10 的自然对数，近似值是 2.302

续表

属性名	描述
LOG2E	以 2 为底的、E 的对数，近似值是 1.442
LOG10E	以 10 为底的、E 的对数，近似值是 0.434
PI	π常数，即圆的周长和直径之比，近似值为 3.142
SQRT0.5	0.5 的平方根，近似值为 0.707
SQRT2	2 的平方根，近似值为 1.414

2．Math 对象的成员方法

(1) 三角函数

三角函数包括三角和反三角两类函数，Math 对象的三角函数见表 5-2。

表 5-2　三角函数和反三角函数

Math 对象的成员方法	说明
acos(*Value*)	*Value* 的反余弦值，单位是弧度
asin(*Value*)	*Value* 的反正弦值，单位是弧度
atan(*Value*)	*Value* 的反正切值，单位是弧度
atan2(*Xvalue, Yvalue*)	直角坐标系中(*Xvalue,Yvalue*)点与 *X* 轴所成的角度
cos(*Value*)	*Value* 的余弦值，*Value* 的单位是弧度
sin(*Value*)	*Value* 的正弦值，*Value* 的单位是弧度
tan(*Value*)	*Value* 的正切值，*Value* 的单位是弧度

(2) 对数、指数函数

这一类函数包括几个常用的对数和指数操作，另外，还包括常用的幂函数操作，见表 5-3。

表 5-3　对数和指数函数

Math 对象的成员方法	说明
exp(*Value*)	E 的 *Value* 次方
log(*Value*)	*Value* 的自然对数
pow(*BaseValue,ExpValu*)	*BaseValue* 的 *ExpValue* 次方
sqrt(*Value*)	*Value* 的平方根

(3) 舍入函数

舍入函数的作用是将一些不合乎需要的数值转换为需要的数值，例如将浮点数转换为相应的整数，见表 5-4。

表 5-4　舍入函数

Math 对象的成员方法	说明
abs(*Value*)	*Value* 的绝对值
ceil(*Value*)	大于或者等于 *Value* 的最小整数值
floor(*Value*)	小于或者等于 *Value* 的最大整数值
round(*Value*)	*Value* 四舍五入得到的整数值

(4) 其他

Math 对象的成员函数还有：max(*Value*1,*Value*2)，比较两个数值，得到 *Value*1 和 *Value*2 中较大的值；min(*Value*1,*Value*2)，可以得到 *Value*1 和 *Value*2 中较小的值。

还有一个用来产生随机数的成员方法。它的语法是：

```
Math.random();
```

它没有任何参数。在 JavaScript 中，每次载入相同的网页，产生的随机数(序列)是不相同的。例如：

```
document.write(Math.random());
```

上述语句将显示每次不同的随机数。

5.6.3 Date 对象

Date(日期)对象封装了有关时间和日期的一些变量和函数，利用这些变量，可以掌握当前日期和时间。 Date 对象中没有一个属性是可以直接地设置或者取得的，必须通过成员方法来实现。Date 对象的一般格式是：

```
Date(year, month, day, hours, minutes, seconds)
```

1．创建日期对象

创建日期对象的方法和数组对象有些相似，都需要使用 new 关键字，但它的构造函数比较复杂，它有多个参数格式，可以根据需要选择一种格式来生成一个新的日期对象，下面分别介绍这几种格式的语法和范例。

(1) 格式 1　var DateName = new Date();

这是最简单的一种日期对象的创建格式。没有规定任何的属性值，它是当前时间的一个副本，即它储存了当前时间。这个构造函数是获得当前时间的惟一方法。

(2) 格式 2　var DateName= new Date("月 日，年 小时：分：秒");

在这种格式中，参数是一个字符串，它描述了创建的 Date 对象的各个属性，在这个字符串中有一些需要注意的地方：

① 在月、日之间的空格是可以省略的，但是在年、小时之间的空格是不可以省略的，否则会使日期无效。

② 日、年之间的逗号不能省略。

③ 时间部分的小时、分、秒间的冒号不可省略。

④ 当“日”这一项超过当月的天数时，将自动进位换算到下一个月。“小时”这一项超过 24 时，也将进位换算成日。

例如：

```
var meetDate = new Date("November 24,2003 11:50:00");
```

(3) 格式 3　var Date1 = new Date(年,月,日);

这种格式中也有几点要说明：

① 参数是数值，不是字符串，年取后两位。

② 当“月”参数的值超过 12 或“日”参数的值超过当月天数时，将自动进位换算到下一年和下一月。

③“月”参数取值是从 0 到 11，即实际月份比参数大一。

④ 小时、分、秒将被认为是 0。

(4) 格式 4　var DateName = new Date(年,月,日,小时,分,秒);

这种格式与前一种很相近，例如：

```
var Date1=new Date(96,2,23,20,55,00);
```

2．get 方法组

程序不能直接读取 Date 对象的属性，但是可以利用一系列 get 方法来取得需要的信息。

(1) getYear()　返回整数表示的日期对象的年份值。如果这个年份是 1900 年之后的某年，返回的将是后两位数值，例如 1999 将返回 99；如果是 100~1900 范围内的年份，将返回完全值，例如 1769 将返回 1769。

(2) getMonth()　返回整数表示的月份值。取值为 0~11，0 代表 1 月，1 代表 2 月，以此类推。

(3) getDate()　返回整数表示的、对象日期是在当月的第几天。1 表示是第一天(1 号)。

(4) getDay()　返回整数表示的、对象日期是星期几。它的取值范围为 0~6，0 代表周日，1 代表周一……

(5) getHours()　返回整数表示的、对象时间的小时值，它的取值范围为 0~23。

(6) getMinites()　返回整数表示的、对象时间的分值，它的取值范围为 0~59。

(7) getSeconds()　返回整数表示的、对象时间的秒值，它的取值范围为 0~59。

(8) getTime()　返回一个整数表示的、从 1970 年 1 月 1 日 00:00:00 到对象存储的时间所经过的毫秒数。这个数值可以是正值，也可以是负值。

(9) getTimezoneOffset()　返回当地时区和 GMT 标准时的差别，单位是分钟。

3．set 方法组

利用 set 方法，可以设置 Date 对象的属性，这和 get 方法组使用类似，并且几乎是一一对应的。

(1) setYear(*Year*)　设置对象的年份值为 *Year*，一般用 4 位整数。

(2) setMonth(*Month*)　设置对象的月份值为 *Month*，取值应在 0~11 范围内。

(3) setDate(*Day*)　设置对象在这一月的日期号数为 *Day*，取值应在 0~31 范围内。

(4) setHours(*Hour*)　设置对象的小时值为 *Hour*，取值应在 0~23 范围内。

(5) setMinutes(*Minute*)　设置对象的分值为 *Minute*，取值应在 0~59 范围内。

(6) setSeconds(*Second*)　设置对象的秒值为 *Second*，取值应在 0~59 范围内。

(7) setTime(*Time*)　设置对象的各部分值，*Time* 是从 1970 年 1 月 1 日 00:00:00 算起到当前时间经过的毫秒数(ms)。

[例 5-4] 利用 Date 对象显示一个时钟，以演示 Date 对象的应用方法。

```
<html>
<head>
<script language="javascript">
var timeStr, dateStr;
function clock()
{
  now= new Date();
  //时间
  hours= now.getHours();
  minutes= now.getMinutes();
  seconds= now.getSeconds();
  timeStr= "" + hours;
  timeStr+= ((minutes < 10) ? ":0" : ":") + minutes;
  timeStr+= ((seconds < 10) ? ":0" : ":") + seconds;
  document.clock.time.value = timeStr;
  // 日期
  date= now.getDate();
  month= now.getMonth()+1;
  month= ((month < 10) ? "0" : "")+ month;
  year= now.getYear();
  dateStr= "" + month;
  dateStr+= ((date < 10) ? "/0" : "/") + date;
  dateStr+= "/" + year;
  document.clock.date.value = dateStr;
  Timer= setTimeout("clock()",1000);
}
</script>
</head>
 <body onload="clock()">
  <form name="clock">
     时间:<input type="text" name="time"
     size="10" value=""><br>
     日期:<input type="text" name="date"
     size="10" value="">
  </form>
</body>
</html>
```

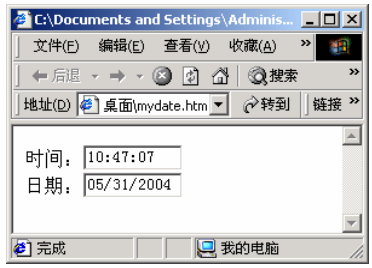


图 5-6 时钟显示示例

显示结果如图 5-6 所示。

4．与字符串有关的成员方法

在 Date 对象中，还有一些成员方法用来将 Date 对象存储的内容转化为字符串。这一类的方法包括 toGMTString()，toLocalString()和 toString()。

这三个成员方法都是将 Date 对象中的所有内容组成一个字符串返回，区别只是这个字符串格式化的方法不同：第一个是按照 GMT 来格式化，第二个是按照本地时间格式化，第三个没有规定明确的格式化方式。

5.6.4　使用数组(Array)对象

与其他的高级语言不同，JavaScript 没有提供明显的数组类型。数组是一种内置对象，它是一组元素对象的集合，元素可以是不同类型的。数组的每一个成员对象都有一个“下标”，用来表示它在数组中的位置(下标从 0 开始)。

1．一维数组

数组定义的一般形式是：

```
var <数组名> = new Array();
```

这样就定义了一个空数组，数组中元素的个数不确定，接下来可以添加数组元素。
添加元素的一般形式是：

```
<数组名>[<下标>] = <表达式>;
```

例如：

```
var colorArray= new Array(); // 定义一个空数组
colorArray[0]= "red";
colorArray[1]= "green";
colorArray[2]= "blue";
colorArray[3]= 255;
```

此外，用户还可以在定义数组的时候直接初始化元素数据，一般形式是：

```
var <数组名> = new Array(<元素 1>, <元素 2>, <元素 3>, ...);
```

例如，下列语句定义了一个数组 myArray，里边的元素分别是：myArray[0] = = 1; myArray[1] = = 4.5; myArray[2] = = 'Hi'。

```
var myArray = new Array(1, 4.5, 'Hi');
```

但是，如果元素列表中只有一个元素，而这个元素又是正整数的话，这将定义一个包含正整数个空元素的数组。
例如，下列语句定义一个长度为 10 的数组，而不是一个数组包含一个正整数元素 10。

```
var myArray = new array(10);
```

2．多维数组

JavaScript 只有一维数组，不能和一般的程序设计语言一样，使用类似 var myArray=new Array(3,4)的方法来定义 4×5 的二维数组，或者用 myArray[1,1]来访问二维数组中的元素。
实际上，所谓多维数组，就是数组的元素本身也是一个数组。因此，要使用多维数组，可用下面的形式来定义：

```
var myArray = new Array(new Array(), new Array(), new Array(), ...);
```

例如，要定义一个 3×4 的二维数组，定义如下：

```
var myArray = new Array(new Array(4), new Array(4), new Array(4));
```

然后，可以用 myArray[i][j]的形式访问二维数组中的元素。

在 JavaScript 中，数组中的元素可以是不同类型的，因此可以定义 oneArray=new Array(new Array(3), new Array(4));,它不是一个 3×4 的二维数组，实际上 oneArray 有两个元素，第一个元素是一个长度为 3 的数组，第二个元素是一个长度为 4 的数组。

3．Array 对象的属性

- length 属性　记录数组的长度，即数组里元素的个数。它等于数组里最后一个元素的下标加 1。因此，想添加一个元素，可写做 myArray[myArray.length] = ...。对于二维数组，例如上面的数组 myArray，document.write(myArray.length)将返回 3，而不是返回 3×4=12。即 myArray 有三个元素，都是长度为 4 的数组。

4．Array 对象的方法

- join(<分隔符>)　把数组中的各个元素串起来，用<分隔符>作为元素之间的分隔符，然后返回这个字符串。该方法不影响数组中的内容。
- reverse()　使数组中的元素顺序反过来。如果对数组[1, 2, 3]使用这个方法，它将使数组变成[3, 2, 1]。
- slice(<s>[, <e>])　返回从数组的第 s 个元素到第 e 个元素的一个子数组，如果未指定 e，则返回的子数组从第 s 个元素到数组的最后一个元素。
- sort([<方法函数>])　使数组中的元素按照一定的顺序排列。如果不指定<方法函数>，则按字母顺序排列。如果指定<方法函数>，则按<方法函数>所指定的排序方法排序。

5.6.5 其他内置对象

上面已经学习了 JavaScript 中的四个内置对象，即字符串对象、数学对象、日期对象和数组对象。在 JavaScript 的较新版本中，还有一些其他的内置对象，如函数对象、数值对象和布尔对象等。它们的使用比较简单，属性和方法比较少，请参阅其他书籍。

5.6.6 预定义函数

除了内置对象外，还有一些功能是经常需要使用的，这些功能被定义成函数的形式。它们不属于任何的对象，因此不用通过引用对象方法的方式来使用它们。

(1) eval()函数

语法形式是：

```
var Value = eval(字符串);
```

参数中的字符串是由一个合法的表达式转化成的字符串，此函数对这个表达式求值，然后返回。此函数的应用如下所示：

```
var tempValue=10;
var str="30+5*tempValue";
var myValue=eval(str);
```

这样，myValue 变量中的值是 80。该函数的作用在于，如果表达式是一个字符串，要想得到表达式的值，就必须利用 eval()函数分析用来存储这个表达式的字符串。

下面将演示一个最简单的计算器。它的缺点是对浏览者输入的错误无法做任何的控制，同时也无法判断用户的输入。

[例 5-5]　一个简单的表达式示例。

```
<html>
<body>
<form name="CalForm">
  输入:<input type="text" name="Input" size="20" value="输入一个表达式">
  <input type="button" value="=" width=8
        onClick="document.CalForm.Result.value=eval(document.CalForm.
        Input.value)">
  结果:<input type="text" name="Result" size="8" value="">
</form>
</body>
</html>
```

(2) parseFloat()函数

将一个合法字符串转换为一个浮点数并返回。例如：

```
var floatStr="1.6e30";
var floatValue=parseFloat(floatStr);
```

parseFloat()函数将从参数字符串的第一个字符开始处理，如果遇到一个合法的表达式中不可能出现的字符，就把这个字符前的所有字符作为字符串来转换，其后的字符将被忽略。

(3) parseInt()函数

它的语法结构是：

```
var value=parseInt(字符串,[数制基数]);
```

在这个函数中，“数制基数”参数是可选的。当函数带有这个参数时，将按照选择的数制格式来转换这个字符串，其他用法与 parseFloat()函数完全相同。

(4) isNaN()函数

当 parseFloat()函数和 parseInt()函数都无法对字符串进行转换时，可以用 isNaN()函数测试结果是否为 NaN。当 Value 是 NaN 时，则 inNaN(value)的值将是 true，否则是 false。

5.7 浏览器内部对象

JavaScript 提供了一些非常有用的常用内部对象和方法，用户不需要用脚本来实现这些功能。这正是基于对象编程的真正目的。

5.7.1 navigator 对象树

在 JavaScript 中，有关浏览器的一些功能和信息被封闭在一系列的对象之中，这些对象被组织成树状结构，称为 navigator 对象树。如图 5-7 所示。

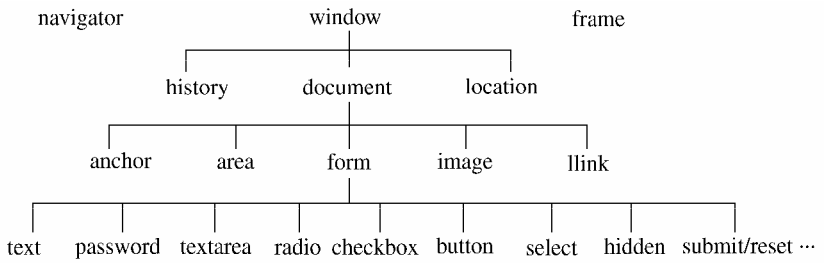


图 5-7 navigator 对象树

navigator 对象树中有多个对象，这些对象各自封装了一些独立的内容，见表 5-5。

表 5-5 navigator 各对象简介

对象名称	描述
navigator 对象	顶层对象，封装了有关操作系统、浏览器版本等环境信息
window 对象	提供有关窗口的一些操作和窗口元素信息
frame 对象	和 windows 对象相似，在浏览器中使用框架网页时使用
history 对象	提供有关浏览历史的相关信息
location 对象	提供有关 URL 的相关信息
document 对象	提供使用文档和文档元素的方法
forms 对象	提供表单的相关信息和方法
anchor 对象	提供使用锚点网页的一些相关信息和方法
link 对象	提供在 JavaScript 中操作链接所需要的方法和信息

需要特别注意的是，上述对象都是小写字符，JavaScript 是大小写敏感的，如果把 navigator 对象写成 Navigator，则程序会提示引用了定义的对象。

5.7.2 navigator 对象

在 navigator 对象树中，navigator 对象处于最顶层，它的主要属性和方法如下。

1．appName 属性

存储表示浏览器名称的字符串。如果客户使用的是 Netscape，则这个字符串是“Netscape”，如果使用的是 Internet Explorer，那么 appName 的值为“Microsoft Internet Explorer”。

2．appVersion 属性

存储客户所用浏览器的版本号，一般形式为：

```
VersionNumber+(PlatformName;+Version;+BitNumber)
```

VersionNumber 为浏览器的版本号；PlatformName 为正在运行的操作系统；Version 是指国内版本还是国际版本；BitNumber 现在只有 16bit 或 32bit 两种，这个属性在一些浏览器中是不提供的。

appName 的另一种用途是判断浏览器所在的平台，其他的属性和方法不再详细说明，将它们列在表 5-6 中。

表 5-6　appVersion 属性列表

属性名	描述
appCodeName	表示当前浏览器的代码名字。对于 Netscape 的所有版本来说，appCodeName 具有相同的值，即 Mozilla。对于大多数的 Internet Explorer 来说也是如此
userAgent	浏览器的完整的用户代理标志，这是 appName，appVersion 和 appCodeName 组合的结果。例如 Mozilla/3.04Gold(Win95;I)
mimeTypes	在浏览器中可以使用的 MIME 类型
plugins	在浏览器中可以使用的 plugins，mimeTypes 和 plugins 存放在两个数组中，这两个数组分别由 mimeTypes 和 plugins 对象组成
javaEnabled()	这个函数返回一个布尔值，表示 Java 是否能在这个客户端浏览器内使用

[例 5-6]　使用 navigator 对象查看客户端浏览器特性。

```
<html>
<head>
<script language="JavaScript">
function TrueFalse (flag)
{
   return(flag? "是":"否");
}
</script>
</head>
<body>
<h2 align=center>您的浏览器特性</h2>
```

浏览器名称：

```
<script language="JavaScript">
   document.write(navigator.appName+"<br>");
```

```
</script>
```

浏览器版本：

```
<script language="JavaScript">
   document.write(navigator.appVersion+"<br>");
</script>
```

支持 javascript**：**

```
<script language="JavaScript">
   document.write(TrueFalse(navigator.javaEnabled())+"<br>");
</script>
</body>
</html>
```

如图 5-8 **所示为上述代码在** IE 4.0 **浏览器中的显示结果。**

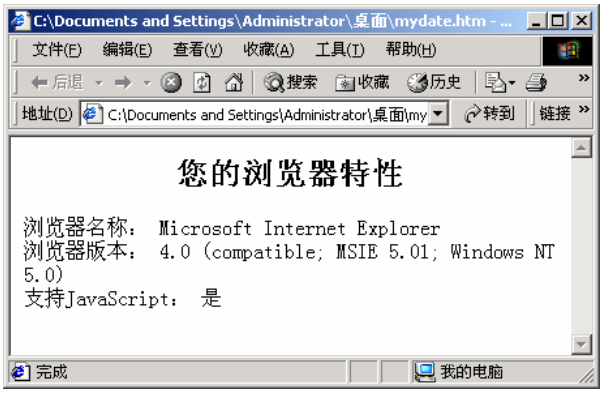


图 5-8　**浏览器特性**

5.7.3　window **对象**

window **即客户的浏览器窗口，**window **对象用于描述窗口的特征，它是** document、location **和** history **对象的父对象。**

window **对象是任何对象、属性和方法的假定父对象。即用户可以在任一对象、属性和方法(不包括** window **对象本身)之前加上一个前缀**“window.”**，作为它的所有者。如果不加，浏览器仍假定其所有者是** window**。例如，语句，**alert("hello,world");**和** window.alert("hello, world");**是等价的。**

1．window，self，parent **和** top

严格地说，window，self，parent **和** top **不能算是** window **对象的属性，更合理的说法是它们是当前环境所涉及的** window **对象实例，或称它们是特殊的关键字。**

self **和** window **所代表的都是当前的窗口，这在功能上发生了重叠，一般也不使用。使用** self **和** window **属性的好处是增加程序可读性，因此在比较复杂的程序中可以考虑使**

用它。

parent 所指的是当前框架或窗口所在的父窗口，这一属性在使用框架的网页中用途最广泛。top 是用以实现所有下级窗口的窗口，即主窗口，它和 parent 有一些相似。

2．defaultStatus 和 status 属性

defaultStatus 是显示在浏览器窗口状态栏上的内容的默认值，status 是在浏览器窗口状态栏上内容的当前值。

刚载入文档时 status 和 defaultStatus 都为空值。在网页中，用户一般期望当鼠标在某个超链接或锚点上经过时，显示一条提示信息，这样的用法可以写成：

```
<a href=URL onMouseOver="status='欢迎光临'; return true">
```

一般情况下，status 属性与 onMouseOver 同时使用。改变 status 属性将直接影响状态栏的显示，如果与 onMouseOver 同时使用，应使此事件处理程序返回 true。当鼠标位于文档窗口内，并且不在任何的超链接和锚点上时，显示 defaultStatus 属性的值。

3．alert()，comfirm()和 prompt()方法

(1) alert(字符串)　显示一个窗口。

(2) confirm(字符串)　是一个确认形式的对话框，但它具有返回值。这个方法调用后显示一个有“确定”和“取消”按钮的对话框，函数返回值是一个布尔值。如果单击“确认”按钮，则返回 true；单击“取消”按钮，返回 false。

(3) prompt()　上面的两个方法都无法从用户那里得到比 yes 和 no 更多的信息。而 prompt()包含两个参数，其语法为：

```
prompt(message,initial input);
```

其中，message 参数的作用类似 alert()的参数；initial input 是输入空白处的默认值；函数调用的返回值是一个字符串。可以这样调用：

```
msg=prompt("请输入您的电话号码？","1234567");
```

对话框内输入电话号码后单击“确定”按钮，那么 msg 这个变量内就保存了该电话号码；相反，如果单击“取消”按钮，msg 是 null。

4．打开和关闭窗口

open()方法打开一个新窗口，它的语法是：

```
open(URL,WindowName,Window Features);
```

其中的第一、第二个参数不用多说，第三个参数是可选的，它表示新窗口的外观，是一个字符串，可以指定多个特性，每一个特性间用逗号分隔。

open()方法是有返回值的，可以认为它的返回值是一个对所创建窗口的引用，或是一个指向这个窗口的指针。

close()用来关闭一个窗口。如果关闭当前窗口，直接使用这个方法即可；如果要关闭

一个由 JavaScript 打开的其他窗口，则需要在打开该窗口时保存好对这个窗口的引用，然后使其调用自身的 close()，如 newWin.close()。

[例 5-7] 定义三个按钮“打开新窗口”、“关闭新窗口”和“查看源码”，单击“打开新窗口”按钮，则打开一个新窗口，单击“查看源码”窗口则显示当前 Web 页的 HTML 源代码。假设窗口的特征是没有工具栏、菜单、状态栏、定位栏，高为 200 像素，宽为 200 像素。HTML 代码如下：

```
<html>
<head>
<script language="javascript">
var temp;  // 全局变量
function newone(url)
{
  newWin=open(url,"subWindow",  "width=200,height=200,toolbar=0,status=0,
  location=0,menubar=0");
  return newWin;
}
function closeone(xWin)
{
    // 若窗口存在,且没有被关闭
    if (xWin != "" && ! xWin.closed)
        xWin.close();
  }
  </script>
  </head>
  <body>
    <input type="button" value="打开新窗口" name="B1"
           onclick="javascript:temp=newone('bookIndex.htm')" > 
    <input type="button" value="关闭新窗口" name="B2"
           onclick="javascript:closeone(temp);" ><br>
    <input type="button" value="显示源文件" name="B3"
           onclick="window.location='view-source:'+window.location.href">
</body>
</html>
```

新窗口对应的 HTML 文件 bookIndex.htm 内容如下：

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>book Index</title>
</head>
<body>
<p align=center>Index for The book</p>
<hr>
<input type="button"  value="关闭窗口"  name="B1"  onclick="window.close()" >
</body>
</html>
```

运行上述代码，单击“打开新窗口”按钮，结果如图 5-9 所示。

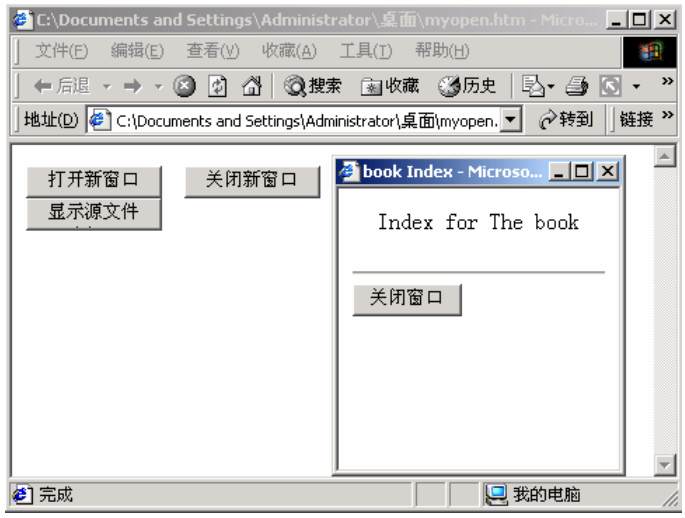


图 5-9　新建窗口示例

5．focus()和 blur()方法

focus()使某个窗口获得焦点，体现为该窗口出现在屏幕屏幕最上方。blur()使某个窗口失去焦点，体现为该窗口被发配到后台。

[例 5-8]　新建一个窗口，不显示任何 HTML 文档，演示如何改变窗口焦点状态。

```
<html>
<title>the operation of window</title>
<body>
<form>
   <input type="button" value="新建窗口"
       onClick='newWin=open("","subWindow","height=200,width=200")'><br>
   <input type="button" value="窗口前台" onClick="newWin.focus()"><br>
   <input type="button" value="隐藏窗口" onClick="newWin.blur()">
</form>
</body>
<html>
```

上述代码可以完成新建窗口，以及实现新窗口前台和后台显示的转换。

6．window 对象的 opener、frames 属性和 scroll 方法

opener 是父窗口的名字，而这个窗口容纳着 open()打开当前窗口时调用的文档。frames 为一个数组，数组的成员是在窗口内的框架(frame)。scroll(x,y)方法有两个参数 x、y，它使窗口卷滚到 x、y 坐标处。

7．setTimeout()和 clearTimeout()方法

Date 对象可以显示当前时间，而无法根据时间的变化动态地进行显示。因此，JavaScript 提供了另外两个方法，即 setTimeout()和 clearTimeout()。

setTimeout 译作“超时”，它的语法是：

```
timeID=setTimeout(expression, time);
```

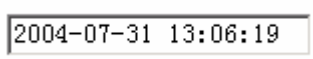
运行 setTimeout()，在 time 时间后，将对 expression 求值一次，但并不是每经过 time 时间就求值一次，因此要想周期地改变，则需要周期地使用这一方法。在实际使用中，expression 参数通常是一个函数，setTimeout()的返回值是一个整数，用以标记一个特定的“超时”。

clearTimeout()的参数只有一个，就是上面的 timeID，通过传递此参数才可以确定要清除哪一个超时设置。

[例 5-9] 演示如何在网页中加入时钟。

```
<html>
<head>
<script language="javascript">
  var timeID;
  var Running=false;
  function StartTime()
  {
    running=true;
    ChangeTime();
  }
  function ChangeTime()
  {
    var currentTime=new Date();
    document.myClock.show.value=currentTime.toLocaleString();
    //递归调用
    timeID=setTimeout("ChangeTime()",1000);
   }
  function StopTime()
  {
     if (Running)
        clearTimeout(timeID);
     Running=false;
  }
</script>
</head>
<body onLoad="StartTime()" onUnload="StopTime()">
<form name="myClock">
   <input type="text" name="show" value=" " size=20>
</form>
</body>
</html>
```

上述代码在浏览器中显示结果如下所示。



5.7.4　document 对象

document 对象是最常用、最重要、最复杂的对象，通过 document 对象的属性和方法，可以实现一切 HTML 文件所完成的功能。前面已经多次用到 document 对象，下面对其做全面的介绍。

1．document 的数值成员属性

所谓数值属性，是指这个属性本身不是任何的对象和数组。document 的属性很多，这些属性的大部分都和 HTML 的标识符相对应。例如，对于<body>标签，可以设置为：

```
<body bgcolor="WHITE" text="BLUE" link="RED" vlink="CYAN" alink="BLACK">
```

则在<body>标识内，设置了一些颜色值。另外，还可以由<title>标识设置网页标题，对于这些设置，可以通过对相应的 document 对象属性赋值来实现。相关的 document 对象属性见表 5-7。

表 5-7　Document 对象属性及描述

属性名	描述
bgColor	网页背景色属性，对应属性 bgcolor="WHITE"。利用 JavaScript 改变这一设置能使网页中背景色立即变化
fgColor	网页前景色，也称为当前文本色，它对应 text="BLUE"。但是当文档已经写入到浏览器窗口内之后，改变这一属性不会立即引起前景色的变化
linkColor	链接文字颜色，它对应 link="RED"，即决定<a href>和之间的文本颜色。与 fgcolor 一样，当文本已经写入浏览器窗口后， 是无法用 JavaScript 的设置来改变该颜色的
vlinkColor	已经被访问过的链接颜色，对应 vlink="CYAN"
alinkColor	激活的链接的颜色，当鼠标在链接上单击时，链接文本显示此颜色，对应 alink="BLANK"
title	这是文档标题属性，对应<title>标题</title>

在这些 document 的颜色属性中，除了 bgColor 属性可以使网页中的相关颜色改变外，其他的属性在文档已经写入到浏览器窗口中后，就无法再改变了。即对这些项的设置应该在<body>中的 JavaScript 执行时来进行，而不应该由事件驱动程序来设置。

[例 5-10]　利用循环来改变背景色，以产生闪烁的效果。

```
<html>
<head>
<script language="javascript">
function ChangeColor()
{
  for (i=0;i<16777216;i++)
     document.bgColor=i;
```

```
}
</script>
</head>
<body>
<form>
  <input type="button" value="改变文档背景颜色" onClick="ChangeColor()">
  <input TYPE="button" value="停止改变背景颜色" onClick="alert('按钮按下')">
</form>
</body>
</html>
```

在上述 html 文档中，当单击第一个按钮“改变文档背景颜色”时，背景色开始改变。此时单击第二个按钮也无法激发对话框。这是因为在 JavaScript 中，事件触发并不会中断正在执行的某个 JavaScript 过程，第一个循环 ChangeColor 非常费时，长时间的循环会干扰事件触发机制。为此，可以使用 JavaScript 的“超时”(SetTimeout)机制来实现。

下面介绍其他几个属性。

lastModified 属性描述 HTML 文件最后被修改的日期，它和 cookie 对象的使用关系紧密。location 属性还有一个名字是 URL，可以用这两个名字来引用同一属性，它反映了现在正在浏览器中显示的 HTML 文件的 URL。referrer 属性是用来描述当前文档的载入文档的 URL，即通过单击文档 A 中的链接到达文档 B，则在 B 中的 referrer 值是 A 的 URL。

例如，document.write(document.lastModified)将显示网页的最后修改日期。

当浏览网页时，有时需要知道网页的发布时间，从而判断网页内容的新旧。此时，可以在浏览器的地址栏中输入：

```
javascript:document.write(document.lastModified)
```

输入结束后，按回车键，则打开一个新的网页，显示当前正在浏览的网页的最后修改日期。然后单击浏览器工具栏中的后退按钮，返回到刚才浏览的网页。

2．anchors 数组

在 HTML 中，锚点(anchor)是用<a name="">标识来定义的。按照它们在 HTML 文档中的定义顺序，anchors 数组中记录了这些锚点。anchors 数组中的每个元素都是一个锚点，这些锚点是只读的。

anchors 数组最简单的用法是采用框架分别设计两个窗口，一个是浏览导航窗口，另一个是主浏览窗口。在浏览导航窗口中根据锚点的数量设置按钮，一个按钮对应一个锚点，按某个按钮则可以在浏览窗口中跳到特定的文字处。

[例 5-11] 锚点应用。

以下是三个 HTML 文档，分别为 main.htm、mainA.htm 和 mainB.htm。

main.htln 文件内容如下：

```
<html>
<frameset cols="*,*">
<frame src="mainA.htm" name="FrameA">
<frame src="mainB.htm" name="FrameB">
```

```
</frameset>
</html>
```

上述 HTML 文档将浏览器窗口分成两部分，左侧窗口用作导航，右侧窗口用作浏览。导航窗口包括几个按钮，它们表示文章中几个可以利用的锚点。

在导航窗口，首先计算浏览窗口中锚点的个数，然后再设置导航按钮的个数。

mainA.htm 文档内容如下：

```
<html>
<script language="javascript">
function SearchAnchor(Index)
{
  parent.FrameB.location.hash="part"+Index;
}
</script>
<body>
<script language="javascript">
var Num=parent.FrameB.document.anchors.length;
for (var i=1;i<=Num;i++)
   {
    document.write("<input type=\'button\' value=\'第",i,"部分")
    document.write("\' onClick=\'SearchAnchor(");
    document.write(i);
    document.write(")\'>");
    document.write("<br>");
  }
</script>
</body>
</html>
```

上述代码中，parent.FrameB.location.hash 是 location 对象的内容，将在后面介绍。

mainB.htm 文件不包含任何 JavaScript 内容，它定义了在 mainA.htm 中引用的必要的锚点，注意锚点的名字是 part1~part3。mainB.htm 的文件内容如下：

```
<html>
<body>
<a name="part1"><b>第一部分 四大名著</b></a>
<ul>
<li>红楼梦
<li>水浒传
<li>西游记
<li>三国演义
</ul>
<a name="part2"><b>第二部分 金庸小说</b></a>
<ul>
<li>笑傲江湖
<li>射雕英雄传
<li>天龙八部
</ul>
<a name="part3"><b>第三部分 中外名著</b></a>
<ul>
```

```
<li>鲁宾逊漂流记
<li>白鹿原
<li>穆斯林的葬礼
</ul>
</body>
</html>
```

3．images 数组成员属性

images 数组的每个元素都是 image 对象。首先看 HTML 中与<img>标签相关的属性，例如：

```
<img name="plane" src="plane.gif" hspace="20" vspace="20" width="30"
height="50" border="5" lowsrc="plane1.gif">
```

上述 HTML 语句中用到了一些不常用的属性，image 对象的成员属性见表 5-8。

表 5-8 image 对象的成员属性

属性名	描述
name	描述图片的名字，此例中是 plane，对应 name="plane"语句
src	描述图片的 URL，在此例中是 plane.gif，对应 src="plane.gif"语句
lowsrc	描述的是在真正的图片还未装载之前，可以先装载的一幅图片的 URL，这幅图片一般是实际图像的低分辨率版本，在本例中是 plane1.gif，对应 lowsrc="plane.gif"语句
border	图像周围边框的宽度，此例中是 5，对应 bofder="5"语句
height	描述图片高度，此例中是 50，对应 height="50"语句
width	描述图片宽度，此例中是 30，对应 width="30"语句
vspace	图片与上面或下面的文字的之间的间隔尺寸，此例中是 20，对应 vspace="20"语句

JavaScript 中使用 image 对象有两个作用，一是可以根据某天的日期和浏览者的特定信息来决定载入哪一张图片；二是可以利用这个对象在网页中实现一些动画功能。这两个作用都是要使图片不再呆滞，而变得动态灵活。

[例 5-12] 动画显示。

```
<html>
<head>
<script language="javascript">
var ImageNum=1;
function Begin()
{
  document.MyImage.src=ImageArray[ImageNum].src;
  ImageNum++;
  if (ImageNum>3)
    ImageNum=1;
}
</script>
</head>
<body>
<img name="MyImage" src="tu.gif" onLoad="setTimeout('Begin()',500)">
<script language="javascript">
var ImageArray=new Array();
```

```
for (i=1;i<=3;i++)
{
   ImageArray[i]=new Image();
   ImageArray[i].src="tu"+i+".gif"
}
</script>
</body>
<html>
```

onLoad 事件设置了 500 毫秒之后运行 Begin()函数，装入下一幅图片后又引起 onLoad 事件，因此 500 毫秒后又运行 Begin()函数，如此循环就实现了一个最简单的动画。

另外，和 image 对象有关的事件包括：onAbort 事件和 onError 事件。当用户放弃载入一个图像时触发一个 abort 事件，于是执行 onAbort 事件处理函数的相应 JavaScript 代码。例如，当在载入的过程中单击一个链接或是单击了浏览器中的 Stop 按钮时，就会出现这种情况。OnError 事件是当浏览器完成一幅图像的载入时触发一个 load 事件，注意，是完成载入后，而不是开始载入时。

4．links 链接数组

链接数组是另一种常用的 document 对象中的数组。链接数组(links)作用就是提供 HTML 中定义的链接给 JavaScript，使它可以得到这些链接的信息并执行操作。

链接数组的每一个元素都是链接对象(link)，或者是 area 对象。link 对象用来存储 URL 的信息。一个完整的 URL 可以分成几个组成部分，例如下面的 URL：

```
http://www.abc.xyz.cn:8080/javascript/index.html#start
```

其中，http 是协议部分，表示使用超级文本传输协议(HTTP)，www 是主机名，abc.xyz 是主机所在的域名，这一部分也可以是数字表示的 IP 地址；8080 是主机用于当前连接的端口号； /javascript/index.html 是路径名；#start 是 hash 数，用以调用文档中的 anchor。

对于 URL 的各个部分，location 对象中有若干个属性与其对应，分别介绍如下。

hash 对应 hash 数，即锚点名，在上例中是#start。host 定义网络主机名、域名或 IP 地址，在例中是 www.abc.xyz.cn。hostname 是主机和端口的组合，在上例中是 www.abc.xyz.cn:8080。 href 代表整个 URL。pathname 是路径，在上例中是/javascript/index.html。port 是服务器端口号，在上例中是 2001。protocol 是协议部分，在上例中是"http"。search 包含任何属性 URL 的查询信息，查询数据前加一个问号。这些数据包含在文档 URL 中的最后一项，字符串中的信息按以下的方式格式化：

```
?ElementName=Element+Value
```

一个链接包括 URL 的所有信息。上面的这些 link 对象成员属性将一个 URL 分解，从而为编程提供了相当大的方便。

[例 5-13]　改变 URL。

```
<html>
<body>
<form>
   <input type="button" value="Google"
```

```
        onClick="document.links[0].href='http://www.google.com' ">
  <input type="button" value="百度"
        onClick="document.links[0].href='http://www.baidu.com' ">
  <input type="button" value="北大天网"
        onClick="document.links[0].href='http://www.pku.edu.cn' ">
</form>
<a href="javascript:alert('选择搜索引擎,然后单击下一步')">下一步</a>
</body>
</html>
```

上述代码中，**下一步****是网页中定义的一个超链接，存储在** document.links[0].href **中。**

一般情况下 href **标记后面的引号中应该是一个** URL**，但是在这个句子中使用的是一种不常见的方式。这里这个写法实际仍旧是一个** URL**，其中**"javascript:"**的字样表示是使用** JavaScript **的协议，而在这个**"JavaScript:"**之后将出现** JavaScript **语句。**

当用户单击其中的一个按钮时，document.links[0].href **的值被改变，因此此时单击“ 下一步 ”超链接就转到相应的** URL**。当鼠标在“ 下一步 ”超链接上经过时，浏览器窗口的状态栏会显示这种** URL **的变化。**

[**例** 5-14] **定义位图映射。**

```
<html>
<body>
<map name="worldMap">
   <area  name="PartOne"  coords="0,0,111,50"    href="javascript:alert
  ('One')">
   <area   name="ParTwo"   coords="0,0,111,111"   href="javascript:alert
  ('Two')">
</map>
<img src="map.gif" usemap="#worldMap">
</body>
</html>
```

在上面的例子中利用<img>**引入一幅图片，并且在该标记中用** USEMAP="#worldMap" **表示使用位图映射，该位图映射由**"worldMap"**这个** MAP **来定义。该** MAP **中定义了位图映射的两个部分,名字分别是** PartOne **和** PartTwo**。**PartOne **这个** Area **的** URL **是一个** JavaScript **语句，这个语句简单地给出一个对话框；**PartTwo **也做相同的工作。**

将鼠标置于图像的上半部分时，鼠标变为手形，说明这是一个链接，单击该处打开一个对话框，显示“ One **”。**

如果希望单击后什么也不做，则可以用函数 void(0)**替代** alert()**，这样单击这个** area **后将不发生任何事情。**

与 link **和** area **对象有关的事件包括：**

- onMouseOver **当鼠标经过一个链接或位图上方时将触发一个** mouseOver **事件。**
- onMouseOut **当鼠标离开链接或位图上方时将触发** mouseOut **事件。当鼠标从位图的一个区域移动到另一个区域时，先发生** mouseOut **事件，然后发生** mouseOver **事件。**
- onClick **这个事件对** area **对象不起作用。当在链接上单击时触发** click **事件。例**

如：

```
<a href=http://www.google.com onClick="alert('使用 Google 搜索引擎')；return
(true)">
```

上面的 return(true)语句表示这个 URL 真正被载入，当返回 false 时是不会真正载入 http://www.google.com 这个地址的。因此，在使用这一事件处理函数时应该注意返回值。

可以利用 onMouseOver 和 onMouseOut 来为网页增添动态效果。例如，可以利用位图映射的链接，并将这个链接的图像做成按钮的开关。当鼠标进入这个区域时，利用 onMouseOver 事件处理函数开始播放一个动画，更换按钮图案，使这个按钮看起来像是在上下跳动，当鼠标离开位图链接，终止这个动画。这样，这个链接看起来会很生动。

document 的属性还包括 applets 数组、cookie 属性、enbeds 数组和 forms 数组等，由于它们使用复杂，在此不做介绍。

上面介绍了 document 对象的大部分属性，通过它们可以完成相当多的功能。下面介绍 document 对象的成员方法。

5．write()和 writeln()方法

write ()方法的一般形式是：

```
document.write(string1,string2,...)
```

这个成员方法的参数可以是任意多个，但是必须至少有一个。当参数不是字符串时，将会被自动转换为字符串。可以如下使用：

```
var Num=4;
document.write("Num=",Num) ;
```

writeln()和 write()类似，只是在输出完括号中的内容后会自动换行。

6．open()，close()和 clear()方法

document 对象的 open()方法允许打开一个已经存在的文件来写入该文件，并且，可以规定这个写入文件的类型，这种文件类型称为 MIME 类型。open()方法的参数支持的 MIME 类型有 test/html、text/plane、image/gif、image/jpeg、image/xbm 和 x-world/plug-in。

使用 open()方法可以创建上面的任一种类型的文档，并向其中写入内容，这样写入窗口的文档称为一个文档流。

如果在使用这个方法时没有参数，那么默认的参数是 text/html，即标准 HTML 文件。text/plain 类型则表示是纯文本格式。image/gif 和 image/jpeg 表示两种图像格式。

用 open()函数打开一个文档流后，就可以用 write()函数向这个窗口内写入文档流。事实上，在浏览器窗口中总有一个打开的文档，前面章节使用的 document.write()就是向这个文档中写入的。当然，也可以打开一个新窗口，并且向这个窗口的新文档中写入。

close()方法的作用是在写完文档流时，关闭这个文档。

clear()则是负责清理文档内容的一个成员方法，但是它必须在 open()方法被调用之后、close()方法被调用之前使用才会有效。

下面的程序先打开一个新窗口，使用新窗口 document 对象的 open()，write()和 close()方法。由于 open()方法的运行是一种创建的过程，原来窗口中的文档将被清除，于是这个例子的效果是在新窗口内刷新窗口中的文档，写入新内容，再刷新，再写入。

[例 5-15] 文档对象的创建。

```
<html>
<body>
<script language="javascript">
AWin=window.open("","","width=200,height=200");
for (var j=0;j<20;j++)
{
  AWin.document.open("text/html");
  AWin.document.write("这是第",j,"个文档！");
  AWin.document.close();
}
</script>
</body>
</html>
```

5.7.5 event 对象

要提高网页的交互性，必须要了解 event 对象。在 IE 和 Netscape 4.0 以上版本的浏览器中都包含 event 对象。它在事件产生时创建，如单击一个可单击的对象，移动鼠标，或或输入焦点等，此时，event 对象被创建然后调用事件处理函数。

1．event 对象的属性

在不同的浏览器中，event 对象的属性不完全相同，其在 IE 和 Netscape 浏览器中的属性见表 5-9 。

表 5-9 event 对象在 IE 和 Netscape 浏览器中的属性

属性描述	IE 中的属性	Netscape 中的属性
事件类型的字符串	type	type
发送给事件对象的字符串	srcElement	target
光标行、列坐标	x、y	x、y
相对于页面的横坐标	clientX	pageX
相对于页面的纵坐标	clientY	pageY
相对于屏幕的横坐标	screenX	screenX
相对于屏幕的纵坐标	screenY	screenY
键代码	keyCode	which

2．可能的事件

在浏览器中可能发生的事件主要有：

- onDblClick　**鼠标双击**
- onKeyDown　**键被按下**
- onKeyPress　**键被按下然后被释放**
- onKeyUp　**键被释放**
- onMouseDown　**鼠标被按下**
- onMouseMove　**鼠标移动**
- onMouseUp　**鼠标被释放**
- onResize　**窗口被调整大小**

3．事件处理方法

事件是由事件处理函数进行处理的。但是，对于两种不同的浏览器，事件的处理途径不完全一样。在 IE **中，采用“事件气泡”的方式处理事件，例如：**

```
<body onclick="f1()">
   <p onclick="f2()">
      <em onclick="f3()">
         <strong onclick="f4()">Click on me</strong>
      </em>
   </p>
</body>
```

对于上述的 HTML **代码，每个标记都接受鼠标单击事件，都创建** event **对象。那么，这个事件如何处理呢？顺序是这样的。单击** strong **标记内的文本，它接收到一个** onClick **事件，执行** f4()**；然后发送** onclick **事件给****标记，执行** f3()**；然后发送到**<p>**标记，执行** f2()**；…；直到主窗口。这样，每个元素分别以自己的方式处理单击事件。**

如果想停止事件上传，可以在事件处理函数中，添加取消事件气泡的代码，方法是：

```
function fx ()
{
... ...;
this.event.cancelBubble = true;
}
```

对于 Netscape**，采用的是事件捕捉(**captureEvents**)和事件释放(**releaseEvents**)的方式，有两个成员函数。例如，用** window **对象描述鼠标移动** MOUSEMOVE **事件，代码为：**

```
window.onMouseMove =yourhandlerFunction;
```

要释放该事件，代码为：

```
window.releaseEvents(Event.MOUSEMOVE);
```

5.7.6　history 对象

一般把 history **对象称做历史清单对象，可以把它看做这样一个列表，它保存窗口或框架在某时间段内的** URL**，并且提供几个方法使得可以在这个列表中前后移动。**

有一点需要注意，出于安全的原因，history 对象并不向脚本提供列表的真正内容，例如一个完整的 URL 名称(如 http://www.microsoft.com)。对这样的 URL 是无法得到这个具体的字符串的，也就是说无法通过它来得到用户曾经访问的 URL 的具体值。这是为了防止恶意的脚本程序获得用户个人的浏览习惯而进行不正当的网络传输或散播，这些信息是大部分用户不愿意泄露的。

history 对象的属性和方法较少，且相对简单，history 对象属性和方法见表 5-10。

表 5-10 history 对象的属性和方法

方法与属性	描述
length	反映历史清单的长度，为只读属性
back()	装载历史清单中的前一个 URL，没有参数
forward()	装载历史清单中的后一个 URL，没有参数
go()	参数既可以是整数，也可以是字符串。当参数是整数 *x* 时，装载历史清单中当前位置偏移 *x* 后的 URL。*x* 既可以是正数，也可以是负数。当参数是字符串时，装载历史清单中含有这个字符串的最新 URL

如果网页中有很多内容，那么靠浏览器提供的 back 按钮返回前一个 URL 较慢，可以在网页中设置“快速返回”按钮，代码如下：

```
<input type='button' value="快速返回" onClick="history.go(-5)">
```

在每页的适当位置放置这个按钮，每单击一次将向后退 5 个 URL，结合浏览器的 back 按钮，可以快速地向后翻页。

5.7.7 location 对象

location(位置)对象用来存储当前的 URL 信息。它储存了一个完整的 URL，并且可能通过对对象的赋值来改变当前的 URL，因此可以改变当前的网页。location 对象的属性和 link 对象是完全相同的，不再做详细地解释。

location 对象的属性有 hash、host、hostname、href、pathname、port 和 protocol 等。

在介绍 document 对象时也曾提到一个 location 属性，要将这两个同名者区分开。只需记住 location 对象是可读可写的，即用户可以对 location 对象的属性赋值；而 document 对象的 location 属性是一个只读属性，它只是提供文档的 URL，改变它是没有意义的。这很好理解，document 的 location 属性描述了文档的 URL，改变它就是表示这个文档被复制到另外的 URL，这是不可能发生的。

5.8 Web 交互

通过 JavaScript 可以完成 Web 的信息交互，在客户端完成动态的 Web 效果。主要的交互包括 window 对象的输入与输出、form 以及 frame。

5.8.1　使用 form 实现 Web 页面的信息交互

表单(form)对象是人机交互的主要手段，是 Web 页面的基本元素，form 对象最主要的功能就是能够直接访问 HTML 文档中的窗体，它封装了相关的 HTML 代码，如下所示：

```
<form name="表的名称" target="指定信息的提交窗口" action="接收窗体程序对应的 url"
method= 信 息 数 据 传 送 方 式 (get | post) enctype ="窗 体 编 码 方 式 " [onsubmit
="JavaScript 代码"]>
```

1．form 对象的方法

form 对象的方法只有一个，即 submit()方法，该方法主要功能是实现 form 信息的提交。例如提交 mytest 表单，则使用下列格式：

```
document.mytest.submit();
```

2．窗体对象的属性

窗体对象中的属性主要包括 elements，name，action，target，encoding，method 等。除 elements 外，其他几个均反映了<form>标记中相应属性的状态，通常是单个 form 标识。而 elements 是多个 form 元素值的数组，例如 myForm.elements[0]、myForm.elements[0]等。

3．访问窗体对象

在 JavaScript 中，访问窗体对象可由两种方法实现。

(1) 通过 form 名称访问

在 form 对象的属性中首先必须指定其名称，而后就可以通过该名称访问 form，如：document.myForm()。

(2) 通过数组来访问 form

除了使用 form 名来访问表单外，还可以使用 form 对象数组来访问 form 对象。但需要注意，因 form 对象是由浏览器环境提供的，而浏览器环境所提供的数组下标是由 0 到 n。所以可通过下列格式实现 form 对象的访问：

```
document.forms[0]、document.forms[1]、document.forms[2]...
```

需要说明的是，在 JavaScript 中对 form 进行引用的条件是：必须先在页面中用<form>标记创建窗体，并将定义窗体部分放在引用之前。

4．form 中的基本元素

form 中的基本元素有文本框(text)按钮、单选按钮(radio)、复选按钮(checkbox)、按钮(button)等组成。在 JavaScript 中要访问这些基本元素，必须通过对应的 form 元素的数组下标或 form 元素名来实现。每一个元素主要是通过该元素的属性或方法来引用。其引用的基本格式为：

```
formName.elements[].propertyName|methodName
```

或者

```
formName.elementName.propertyName|methodName
```

(1) text 单行文本框

属性：name **设置提交信息时的信息名称，对应于** HTML **文档中的** name**；**value **用以设置出现在窗口中对应** HTML **文档中** value **的信息；**defaultvalue **为** text **元素的默认值。**

方法：blur()**将当前焦点移到后台；**select()**加亮文字。**

事件：onFocus**，当** text **获得焦点时，产生该事件；**OnBlur**，从元素失去焦点时，产生该事件；**Onselect**，当文字被加亮显示后，产生该事件；**onchange**，当** text **元素值改变时，产生该事件。例如：**

```
<html>
<body>
<form name="myForm">
<input type="text" name="text1" value="xxx" >
</form>
<script Language = "JavaScript">
   document.myForm.text1.value = "yyy";
   document.myForm.text1.select();
</script>
</body>
</html>
```

(2) textarea 多行多列文本输入

属性：name **设置提交信息时的信息名称，对应** HTML **文档** textarea **的** name**；**value **用以设置出现在窗口中对应** HTML **文档中** value **的信息；**default value **为元素的默认值。**

方法：blur() **将失去输入焦点；**select()**将文字加亮；**

事件：onBlur**，当失去输入焦点后产生该事件；**onFocus**，当输入获得焦点后，产生该事件；**Onchange**，当文字值改变时，产生该事件；**Onselect**，当文字加亮后，产生该事件。**

(3) select 选择元素

功能：实施对滚动选择元素的控制。

属性：name **设置提交信息时的信息名称，对应文档** select **中的** name**；**length **对应文档** select **中的** length**；**options **组成多个选项的数组；**selectIndex **指明一个选项；**selected**，指明当前选项是否被选中；**index **指明当前选项的位置；**defaultselected **默认选项。**

事件：OnBlur**，当** select **选项失去焦点时，产生该事件；**OnFocus**，当** select **选项获得焦点时，产生该事件；**Onchange**，选项状态改变后，产生该事件。**

(4) button 按钮

属性：name **设置提交信息时的信息名称，对应文档中** button **的** name**；**value **用以设置出现在窗口中对应** HTML **文档中** value **的信息。**

方法：click()**方法，类似于一个按下的按钮。**

事件：onclick**，当单击** button **按钮时，产生该事件。**

(5) checkbox 复选框

属性：name **设置提交信息时的信息名称；**value **用以设置出现在窗口中对应** HTML **文**

档中 value 的信息；Checked 该属性指明框的状态 true/false. Defaultchecked 为默认状态。

方法：click()方法，使得复选框的某一项被选中。

事件：onclick，当复选框的选项被选中时，产生该事件。

(6) radio 单选按钮

属性：name 设置提交信息时的信息名称，对应 HTML 文档中 radio 按钮的 name；value 用以设置出现在窗口中对应 HTML 文档中 value 的信息，对应 HTML 文档中的 radio 的 name；Length 为一组单选按钮中的按钮数目；Defaultchecked 为默认按钮；Checked 指明该按钮选中还是没有选中；Index 选中的按钮的位置。

方法：click()，选定一个按钮。

事件：onclick，单击按钮时，产生该事件。

(7) hidden 隐藏

属性：name 设置提交信息时的信息名称，对应 HTML 文档中 hidden 的 name；value 用以设置出现在窗口中对应 HTML 文档中 value 的信息，对应于 HTML 文档 hidden 中的 value；Defaultvalue 为默认值。

(8) password 口令

属性：name 设置提交信息时的信息名称，对应 HTML 文档中 password 中的 name；value 用以设置出现在窗口中对应 HTML 文档中 value 的信息，对应于 HTML 文档中 password 中的 value；defaultvalue 为默认值。

方法：select(),加亮输入口令域；blur(),丢失 password 输入焦点；focus(),获得 password 输入焦点。

(9) submit 提交按钮

功能：实施对一个具有提交功能按钮的控制。

属性：name 设置提交信息时的信息名称，对应 HTML 文档中的 submit；value 用以设置出现在窗口中对应 HTML 文档中 value 的信息，对应 HTML 文档中 submit 按钮的 value。

方法：click()，相当于单击 submit 按钮。

事件：onclick()，单击该按钮时，产生此事件。

[例 5-16]　演示通过单击一个按钮(red)来改变窗口颜色，单击“调用动态按钮文档”按钮来调用一个动态按钮文档。

```
<html>
<head>
<script language="javascript">
// 设置当前文档的颜色
document.bgColor="blue";
document.vlinkColor="white";
document.linkColor="yellow";
document.alinkcolor="red";
// 动态改变颜色
function changecolor()
{
  document.bgColor="red";
  document.vlinkColor="blue";
  document.linkColor="green";
```

```
  document.alinkcolor="blue";
}
</script>
</head>
<body bgColor="White" >
<a href="other.htm">调用其他文档</a>
<form >
  <Input type="button" value="red" onClick="changecolor()">
</form>
</body>
</html>
```

5.8.2 使用 frame 实现复杂的交互

框架(frame)可以将屏幕分割成不同的区域，每个区域有自己的 URL，通过 frames[]数组对象来实现不同框架的访问。实际上框架对象本身也是一种窗口，它具有窗口对象的所有特点，并拥有其所有的属性和方法。

例如，下面是一个包含三个框架的框架网页。

```
<html>
<frameset rows="20%,80%">
   <frame src="frameA.htm">
   <frameset Cols="200,*">
      <frame src="frameB.htm">
      <frame src="frameC.htm">
</frameset>
</frameset>
</html>
```

以上 HTML 标识将屏幕分成三个框架。先将窗口分两行，之后再将第二行分成两列，其中，第一列占用 200 像素的固定宽度。

前面介绍过如何使用 document.forms[]实现一个 form 中不同元素的访问。而要实现对框架中多个 form 的不同元素的访问，则必须使用 window 对象中的 frames 属性。frames 属性同样也是一个数组，它在父框架集中将每一个子框架为一个元素，通过下标可以实现对各框架的访问，如下所示:

```
parent.frames[i].docuement.forms[j]
```

通过 parent.frames.length 确定窗口中窗体的数目。除了使用数组下标来访问 form 外，还可以使用框架名和 form 名来实现各元素的访问，形式如下：

```
parent.frameName.document.formName.elementName;
```

5.9 综 合 举 例

Web 客户端的编程比较简单，它不需要特别的编译和运行环境，只要有一个浏览器即

可。但是，要编写高质量的客户端脚本并不容易，这取决于开发人员的实践经验，也来源于用户的需求。

没有需求，就不会有深入的编程体验，更无法把一个工具学精。为此，在本章的最后，给出了两个相对规模较大的 Web 应用，帮助读者从总体上理解 Web 客户端编程思想。

5.9.1　一个 Web 课件框架

在 E-learning 中，课件是重要的教育资源，一种简单易用的课件界面对于 E-learning 系统有着重要的作用。图 5-10 是 Genaral Self-learning 网络教育平台(GSL)中的用户界面。

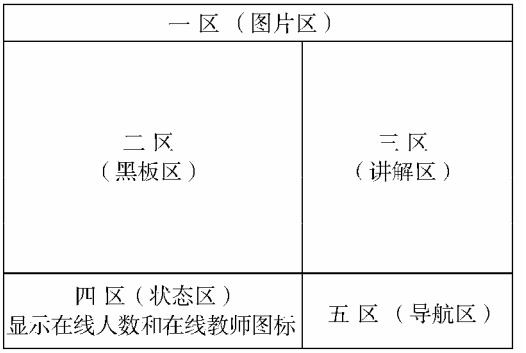


图 5-10　Web 课件界面

图 5-10 中，一区显示课程的标题图片。二区为黑板区，在黑板区中主要展示课程中的插图、动画、视频、程序、公式推导、定理证明等教师课堂教学中在黑板上书写的内容。三区为教案讲解区，主要是文字解说，对应教师在课堂教学中的讲解。黑板区与教案讲解区紧密配合，二区与三区配合完成整门课程的教学内容设计，通过标准的稿本创作规范由教师来完成。四区为状态区，是学生和系统在线交互的主要入口，显示在线学生数量、在线辅导教师以及供学生提问的帮助入口。五区为导航区，完成默认的页面导航，包括目录、知识点、上一页、下一页、实验、练习题等超级链接，实现每一章学习内容的交互。

下面一节将利用 JavaScript 脚本程序来完成上述界面的设计，它包括了 JavaScript 中内嵌对象、框架、表单、参数传递、包含等各种各样的技术。

1．建立 Web 应用中的一个页面——初始框架

首先根据需求设计 Web 应用的总体框架。根据图 5-10，在这个 Web 应用中，第一个网页应该是一个框架网页，这里将主调文档命名为 myframes.htm。

用 FrontPage 工具，完成 myframes.htm 框架网页的设计，代码如下。

主调文档 myframes.htm 的内容：

```
<html>
<head>
<title>GSL 网络课程</title>
</head>
```

```
<frameset rows="60,*,40" framespacing="0" border="0" frameborder="0">
  <frame name="top" scrolling="no" noresize src="mybanner.htm">
  <frameset cols="*,250">
    <frame name="frameA" target="main" scrolling="auto" src="myA.htm" >
    <frame name="frameB" target="_self" scrolling="auto" src="myB.htm" >
  </frameset>
  <frameset cols="*,250">
    <frame name="frameC" scrolling="no"  src="myC.htm">
    <frame name="frameD" scrolling="auto"
           src="myD.htm?chapter=ch01&pagenums=26&page=0" >
  </frameset>
</frameset>
</html>
```

整个窗口分成五个帧，帧的名字分别是 top、frameA、frameB、frameC 和 frameD。其中，top 帧主要负责显示标题图片，frameA 帧用于显示目录超链接，单击某个链接，在 frameB 中显示相应的具体内容。frameC 帧为状态显示，与其他帧没有超链接关系，frameD 为导航帧，它会影响 frameB 的显示内容。

为了使浏览器窗口发生变化时，fameA、frameC 和 frameD 帧不发生变化，定义帧尺寸时使用了帧的绝对像素值，而不是百分比。

2．top 帧的设计

top 帧设计为显示一幅图片或 Flash 动画，其源文件是 mybanner.htm，这是最简单的一个网页，只要设置一幅背景图片即可，或者根据帧的尺寸插入一个 Flash 动画。一个简单的文档 mybanner.htm 内容如下：

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>banner</title>
</head>
<body background="../例子/image/banner.jpg">
</body>
</html>
```

3．frameC 帧的设计

frameC 帧主要用于显示系统的状态，它对应 myC.htm 网页，可以用来显示当前的在线人数，这里暂不讨论。

4．frameD 帧的设计

frameD 帧是页面导航区，包括目录、上一页、下一页、练习题等超级链接，是相对复杂的一个网页。对应的文档 myD.htm 内容如下：

```
<html>
<head>
<meta content="text/html; charset=windows-1252" http-equiv="Content-Type">
```

```
<script src="QueryString.js" ></script>
<script src="indexdatas.js" ></script>
<script language="JavaScript">
  //(1) index
  image1on = new Image();
  image1on.src = "images/indexov.gif";
  image1off = new Image();
  image1off.src = "images/indexup.gif";
  //(2) up button,there status:on,off,and gray
  image2on = new Image();
  image2on.src = "images/backov.gif";
  image2off = new Image();
  image2off.src = "images/backup.gif";
  image2dim = new Image();
  image2dim.src = "images/backdim.gif";
  //(3) next button,there status:on,off,and gray
  image3on = new Image();
  image3on.src = "images/nextov.gif";
  image3off = new Image();
  image3off.src = "images/nextup.gif";
  image3dim = new Image();
  image3dim.src = "images/nextdim.gif";
  //(4) exercise
  image4on = new Image();
  image4on.src  = "images/exerciseov.gif";
  image4off = new Image();
  image4off.src = "images/exerciseup.gif";
  function myButtonInit()
  {
    var mynum = parseInt(mypagenums,10);
    document['image2'].src = image2dim.src;
    if (mynum <= 1)
      document['image3'].src = image3dim.src;
   }
   function showPic()
   {
     if (document.images && PageNums > 1)
       for (var i=0; i < showPic.arguments.length; i+=2)
          {
           if (showPic.arguments[i] == 'image3' && CurrentPage == PageNums)
             return;
          else if (showPic.arguments[i] == 'image2' && CurrentPage == 1)
                   return;
               else
           document[showPic.arguments[i]].src=eval(showPic.arguments
           [i+1]+".src");
          }
    }

  function changePage(cha,thePage)
```

```
  {
    var pagecode;
    switch (cha) {
        case "ch01":
           pagecode = PagesIDs01[thePage];
           break;
         case "ch02":
           pagecode = PagesIDs02[thePage];
           break;
         ... ...
     }
     return pagecode;
}
function NextPage()
{
    if (CurrentPage == PageNums-2)
        return;
    if ((CurrentPage + 1) == PageNums-2)
        document['image3'].src = image3dim.src;
    CurrentPage++;
      pagecode = changePage(mycha,CurrentPage);
        NextPageURLa = mycha + "/" + pagecode +"a"+".htm";
    NextPageURLb = mycha + "/" + pagecode +"b"+".htm";
   eval('parent.frames.frameA.location.href=NextPageURLa' );
    eval('parent.frames.frameB.location.href=NextPageURLb' );
}
function PrevPage()
{
    if (CurrentPage == 0)
        return;
    if (CurrentPage == 1)
        document['image2'].src = image2dim.src;
    CurrentPage--;
    pagecode = changePage(mycha,CurrentPage);
   BackPageURLa = mycha + "/" + pagecode +"a"+".htm";
    BackPageURLb = mycha + "/" + pagecode +"b"+".htm";
    eval('parent.frames.frameA.location.href=BackPageURLa');
    eval('parent.frames.frameB.location.href=BackPageURLb');
}
function openIndex()
{
  if (popup == "" || popup.closed)
  {
    url = "index/indexframes.htm";   // 书本的全部三级目录索引页面
    popup= window.open("","IndexWin","scrollbars=yes,width=550,height=600,
                       top=70,left=120,resizable=yes");
    popup.location = url;
  } else popup.focus();
}
</script>
```

```
<base target="_self">
</head>
<body leftmargin="1" topmargin="1" bgcolor="#000000">
<script language=JavaScript>
  var Request=new QueryString();      // 创建参数对象实例
  mycha     = Request["chapter"];    // 章编号
  mypagenums = Request["pagenums"];  // 当前章页数
  mypage    = Request["page"];     // 当前页
  var PageNums  = parseInt(mypagenums,10);
  var CurrentPage = parseInt(mypage,10);
  var popup="";
</script>
<div align="center">
  <center>
  <table border="0" cellpadding="0" width="100%" height="100%">
    <tr>
      <td align="center" width="100%">
         <table bgcolor="#000000" border="0" cellspacing="0" cellpadding="0">
            <tr>
              <td><a href="javascript:openIndex()"
                   onMouseOver="showPic('image1', 'image1on')"
                   onMouseOut="showPic('image1', 'image1off')">
                   <img name="image1" src="images/indexup.gif" ></a>
              </td>
              <td><a href="PrevPage()"
                   onMouseOver="showPic('image2', 'image2on')"
                   onMouseOut="showPic('image2', 'image2off')">
                   <img name="image2" src="images/backdim.gif" ></a>
              </td>
              <td><a href="NextPage()"
                   onMouseOver="showPic('image3', 'image3on')"
                   onMouseOut="showPic('image3', 'image3off')">
                   <img name="image3" src="images/nextup.gif" ></a>
              </td>
              <td><a href="GoExercise()"
                   onMouseOver="showPic('image4', 'image4on')"
                   onMouseOut="showPic('image4', 'image4off')">
                   <img name="image4" src="images/exerciseup.gif" ></a>
              </td>
            </tr>
          </table>
      </td>
    </tr>
  </table>
  </center>
</div>
<script language=JavaScript>
  myButtonInit();
</script>
</body>
```

```
</html>
```

myD.htm 文档的显示结果如图 5-11 所示。

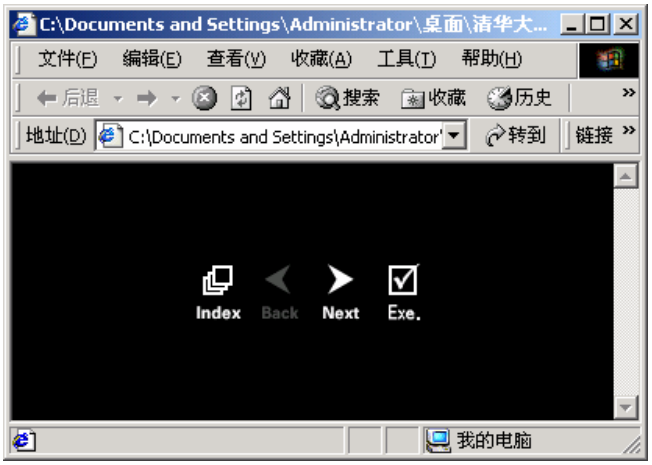


图 5-11 myD.htm 的显示效果

说明：

(1) 在 myD.htm 中，可以看到每一个图片超链接的 href 属性值写成了类似 href="javascript: launchIndex()"的形式，这和传统的 href="launchIndex()"一样吗？读者可以实验一下，如果是第一种写法，launchIndex()将打开一个新窗口。对于第二种写法，launchIndex()并不打开一个新窗口，这与在 HTML 文档的<head>…</head>中写了<base target="_self">有关。

(2) 对于一个图片或一段文本，可以设置为水平居中，如果要使一个图片在网页内水平和垂直居中，其实现方法是将内容插入到一个 1×1 的表格中，并设置表格属性为 width="100%" height="100%"，然后在单元格中插入相应的内容即可。

(3) 包含文件

在 myD.htm 文档中，用到了两个包含文件 QueryString.js 和 indexdatas.js。其中 QueryString.js 定义了一个参数类，用于获得当前页面的参数。在本例子中，传递到 myD.htm 页面的有三个参数，分别是 chapter(当前章编号)、pagenums(当前章的页数)和 page(当前页)。

indexdatas.js 文档内容如下：

```
function QueryString()
{//构造参数对象并初始化
  var name,value,i;
  var str=location.href;          //获得浏览器地址栏 URL 串
  var num=str.indexOf("?")
  str=str.substr(num+1);          //截取?后面的参数串
  var arrtmp=str.split("&");      //将各参数分离,形成参数数组
  for(i=0;i < arrtmp.length;i++)
  {
    num=arrtmp[i].indexOf("=");
    if(num>0)
```

```
    {
      name=arrtmp[i].substring(0,num);  //取得参数名称
      value=arrtmp[i].substr(num+1);    //取得参数值
      this[name]=value;                 //定义对象属性并初始化
    }
  }
}
```

包含文件 indexdatas.js 定义了整个 Web 应用中所有网页用到的全局变量，内容如下：

```
var PageNums01 = 26   // 第 1 章的页数
var PagesIDs01 = new Array(26);  // 存储每一页的 ID,其格式为所在页数+节+小节,节和小节各占两位
PagesIDs01[0] = "p00";
PagesIDs01[1] = "p0100";
...
PagesIDs01[25] = "p0702";
// 以下是第二章、第三章...等其他章的数据，和第一章类似。
```

5．目录索引页设计

在 frameD 帧(见图 5-11)中，单击 index 按钮，将打开一个新的窗口，该窗口保存了一门课程的目录，通过该目录，学生可以进入任何一章、任何一个小节。此时，在 frameA 和 frameB 中将显示该小节的内容。

可以通过 frameD 帧中的 back 和 next 按钮实现前后翻页。

目录索引页 indexframes.htm 设计成一个框架网页，如图 5-12 所示。

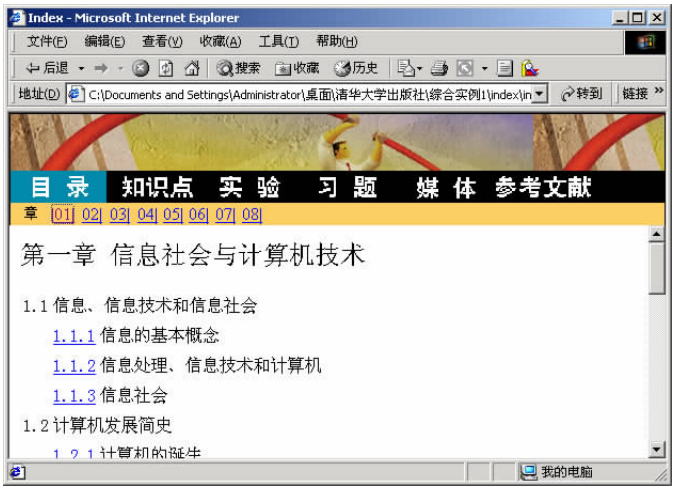


图 5-12　目录索引页

indexframes.htm 框架网页包含了上下两个帧，上面显示菜单(myindexmenu.htm)，下面是每一章的目录(每章对应一个文件/chxx/pindex.htm，其中 xx 为章号)。

indexframes.htm 文档内容如下：

```
<html>
```

```
<head>
<meta content="text/html; charset=windows-1252" http-equiv="Content-Type">
<script>
var activeindex = null;
function showLayer(layer) {
    eval(activeindex+".hide()");
    activeindex = layer;
    eval(layer+".show()");
}
function hideLayer(layer) {
    eval(layer+".hide()");
}
</script>
</head>
<frameset frameborder="0" border="0" framespacing="0" rows="100, *">
   <frame name="indexmenu_frame"  src="myindexmenu.htm" target="_self">
   <frame name="indexcontent_frame" src="pleaseselcha.htm"  target="main">
</frameset>
</html>
```

菜单页面 myindexmenu.htm **文档的内容如下：**

```
<html>
<head>
<meta content="text/html; charset=windows-1252" http-equiv="Content-Type">
<script language="JavaScript">
   var GroupID  = 0;     // 表明所选的是“目录”、“知识点”、“实验”等一级菜单目录
   var SelChapter = 0;  // 所选的章节,菜单的二级目录
   // index button
   image1indexon = new Image();
   image1indexon.src  = "image/1indexdn.gif";
   image1indexoff = new Image();
   image1indexoff.src = "image/1indexup.gif";
   // 其他如知识点、实验、实习、媒体和参考文献按钮对象的操作类似,代码略
  function changeImages()
  {
     document[changeImages.arguments[0]].src = eval(changeImages.arguments
     [1] + ".src");
  }
 function ClearOldSel()
 {
   switch (GroupID)
   {
     case 1:
       changeImages('image1', 'image1indexoff');
         break;
      case 2:
         changeImages('image2', 'image2knowledgeoff');
         break;
       ... ...
    }
```

```
 }
function SelIndex()
 {
    ClearOldSel();
 GroupID = 1;
    changeImages('image1', 'image1indexon');
    eval('parent.frames.indexcontent_frame.location.href = "pleasesel
    cha.htm" ' );
 }
function LeftIndex()
{
    if (GroupID != 1)
    changeImages('image1', 'image1indexoff');
}
// 其他选择一级目录的 SelXXX 代码与上面选择目录的代码类似,略
function GoContents(chapter)
{
   switch (GroupID)
   {
     case 1:
       str = "pindex.htm";  // 每一章的目录页,分别存在 ch01、ch02...文件夹下
       break;
     // 不同的一级目录所对应的帧显示的内容不同,其他几种情况略
    }
   ContentsPageURL = "../" + chapter + "/"+  str ;
   eval('parent.frames.indexcontent_frame.location.href = ContentsPageURL' );
}
</script>
<base target="_self">
</head>
<body leftmargin="1" topmargin="1" bgcolor="#000000">
  <div align="center">
  <table bgcolor="#000000" border="0" cellspacing="0" cellpadding="0"
  width="100%">
     <tr>
     <td width="540" height="53" background="image/indextopbg.gif"></td>
     </tr>
  </table>
  <div align="left">
  <table bgcolor="#000000" border="0" cellspacing="0" cellpadding="0"
  width="100%">
     <tr>
       <td><a href="javascript:SelIndex();"
             onMouseOver="changeImages('image1', 'image1indexon')"
             onMouseOut="LeftIndex()">
             <img border="0" name="image1" src="image/1indexup.gif" ></a>
       </td>
       <td><a href="javascript:SelKnowledge();"
             onMouseOver="changeImages('image2', 'image2knowledgeon')"
             onMouseOut="LeftKnowledge()">
```

```
                <img border="0" name="image2" src="image/2knowledgeup.gif"></a>
            </td>
            <!-其他一级目录代码类似,略->
          </tr>
        </table>
        </div>
        <table border="0" width="100%" cellspacing="0" cellpadding="0" bgcolor=
        "#FFCC66">
            <tr>
              <td width="54" align="center" valign="middle">章:</td>
              <td width="34"><a href="javascript:GoContents('ch01');">|01|</a>
              </td>
              <td  width="30"><a  href="javascript:GoContents('ch02');">02|</a>
              </td>
              <!-其他章类似,代码略->
            </tr>
        </table>
        </div>
    </body>
    </html>
```

6．各章的目录页设计

在目录窗口帧选择了一级项目、再选择章节后，在 indexcontent_frame **帧上将显示相应章的目录，目录文件名为** pindex.htm**。该文件含有大量的超链接，最终，通过这些超链接来刷新** frameA **和** frameB **中的内容。**

例如，某一章的 pindex.htm **页面如图** 5-13 **所示。**

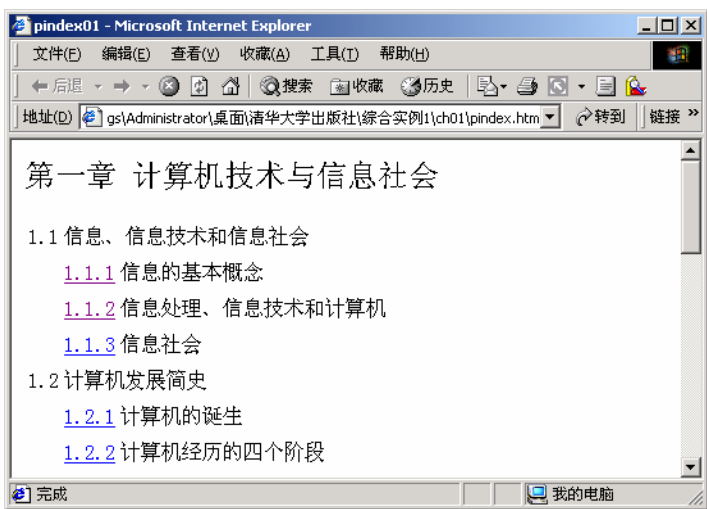


图 5-13　**各章的目录页面**

pindex.htm **文档内容如下：**

```
<html>
```

```
<head>
<meta http-equiv="Content-Language" content="zh-cn">
<title>pindex01</title>

<script src="../indexdatas.js" ></script>
<script language="JavaScript">
    function GoContentPage(selstr)
    {
     Chapter   = selstr.substring(0,2);
     Section   = selstr.substring(2,4);
     Subsection = selstr.substring(4,6);
     str = "p"+selstr.substring(2,6);
     CurrentPage = 0;
     while (CurrentPage<PageNums01)
     {
        if (PagesIDs01[CurrentPage] == str)
           break;
        CurrentPage ++;
     }
     if (CurrentPage == PageNums01)
          CurrentPage = 0;
     NewPageURLa = "../ch"+Chapter+"/"+PagesIDs01[CurrentPage]+"a"+".htm";
     NewPageURLb = "../ch"+Chapter+"/"+PagesIDs01[CurrentPage]+"b"+".htm";
     if (window.parent.opener && !window.parent.opener.closed)
     {
      window.parent.opener.parent.frames[1].location.href = NewPageURLa;
      window.parent.opener.parent.frames[2].location.href = NewPageURLb;
    }
     // 刷新 frameD 帧的内容,使得其中的数据和索引页的选择同步
     newurl = "../myD.htm?chapter=ch01&pagenums=26&page=" + CurrentPage;
     window.parent.opener.parent.frames[4].location = newurl;
     top.blur();
    }
</script>
<!--
<base target="main">
//-->
</head>
<body>
<p><font size="5">第一章 计算机技术与信息社会</font></p>
<div align="center">
  <center>
<table border="0" width="100%">
  <tr>
    <td ><p style="line-height: 150%">1.1</p></td>
    <td width="95%" colspan="2">
      <p style="line-height: 150%">信息、信息技术和信息社会</p>
    </td>
  </tr>
  <tr>
```

```
    <td ><p style="line-height: 150%"></p></td>
    <td ><p><a href = "#" onClick="GoContentPage('010101');">1.1.1</a></p></td>
    <td ><p style="line-height: 150%">信息的基本概念</p></td>
  </tr>
  <!-其他各个小节的代码完全类似,略->
</table>
</center>
</div>
</body>
</html>
```

最终的系统界面如图 5-14 所示。

图 5-14 最终完成的 GSL 课程界面

7．总结

上述的代码很长，从中可以体会到，Web 应用的客户端编程是非常复杂的，但是只要经过不断的积累，一定会编写出精彩的 Web 应用。

5.9.2 一个文本文档批注系统

对于写在纸面上的东西，用户很容易对内容进行修改或添加说明。如果是带有格式的文档，如 Word 文档，可以用不同的颜色、插入批注等方式对文档进行说明。

对于文本文档，由于没有格式，因此无法用 Word 提供的方法来实现对文档的说明。因此，可以利用 HTML 和 JavaScript 技术，开发一个文本文档的批注系统。

1．登录界面设计

首先设计一个登录界面，不同的角色具有不同的批注权限。设计的登录界面如图 5-15

所示。

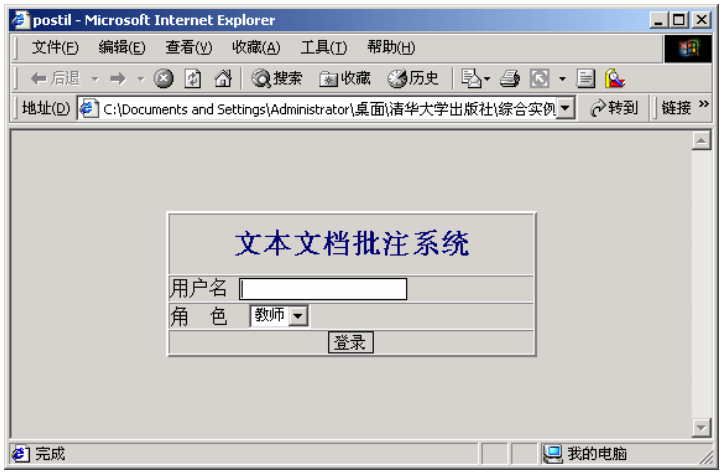


图 5-15　登录界面

对应的 HTML 文档 mypostil.htm 内容如下：

```
<html>
<head>
<title>postil</title>
<link href="css/site.css" rel="stylesheet" type="text/css">
<script language=javascript src="script/myglobal.js"></script>
<script language=javascript>
function login()
{
  var name = document.all.userName.value;
  if( isEmpty( name ) )
  {
    alert( "用户名不能为空！" );
    document.all.userName.focus();
    return;
  }
  var r = document.all.role.value;
  if( r == "teacher" )
     window.open( "fileload.htm" , "_self" );
}
</script>
</head>
<body onload=javascript:document.all.userName.focus(); bgcolor=menu>
<div align="center">
 <center>
 <table border="0" cellpadding="0" width="100%" height="100%">
 <tr>
 <td width="100%">
 <table border=2 cellspacing=0 cellpadding=0 width=300 align=center>
    <tr height=50>
```

```
        <td align=center style="font-size:20;font-weight:bold;color:#000075">
文本文档批注系统</td>
    </tr>
    <tr>
        <td style="font-size:16">用户名 <input type=text name=user-
        Name></td>
    </tr>
    <tr>
        <td style="font-size:16">角  色 
          <select name=role>
             <option value=teacher>教师</option>
             <option value=student>学生</option>
          </select>
        </td>
    </tr>
    <tr>
    <td align=center><input type=button value=登录 onclick=login()></td>
    </tr>
</table>
 </td>
 </tr>
 </table>
 </center>
</div>
</body>
</html>
```

说明：在上述文档中引用了一个.CSS **文件** site.css**，它定义了应用中的样式。同时，还用到了一个包含文件** myglobal.js**，定义各个** HTML **文档用到的公共函数和全局变量。**

(1**)** site.css **样式文件内容**

```
body {
  FONT-FAMILY: "宋体"; FONT-SIZE: 9pt
  SCROLLBAR-HIGHLIGHT-COLOR: rgb(92,149,255);
  SCROLLBAR-3DLIGHT-COLOR: rgb(92,149,255);
  SCROLLBAR-ARROW-COLOR: rgb(70,255,229);
  SCROLLBAR-TRACK-COLOR: rgb(92,149,255);
  SCROLLBAR-DARKSHADOW-COLOR: rgb(92,149,255);
  SCROLLBAR-BASE-COLOR: rgb(88,120,255);
}
table {FONT-FAMILY: "宋体"; FONT-SIZE: 9pt}
td {FONT-FAMILY: "Verdana"; FONT-SIZE: 9pt}
input {
  border:1 solid black;FONT-FAMILY: "Verdana";FONT-SIZE: 9pt; FONT-STYLE:
  normal;FONT-VARIANT:normal;FONT-WEIGHT:normal;HEIGHT:18px; LINE-HEIGHT:
  normal
}
textarea {FONT-FAMILY: "宋体"; FONT-SIZE: 9pt}
select {FONT-FAMILY: "宋体"; FONT-SIZE: 9pt}
a {FONT-FAMILY: "宋体"; FONT-SIZE: 9pt;COLOR: #000075;text-decoration:none}
```

```
a.x {COLOR: #ffffff}
a:hover {COLOR: #cc0000}
pre {FONT-FAMILY: "Courier","mono"; FONT-SIZE: 9pt}
CODE {FONT-FAMILY: "Courier","mono"; FONT-SIZE: 9pt}
.head-x {  font-family:  "Verdana", "宋体"; font-size: 12pt; font-weight:
 bold}
.head-xx {  font-family:  "Verdana", "宋体"; font-size: 10.5pt; font-weight:
 bold}
.head-xxx {  font-family:  "Verdana", "宋体"; font-size: 9pt; font-weight:
 bold}
p {  font-family:  "Courier","Verdana", "宋体"; font-size: 9pt }
p.foot {font-family:"宋体";font-size:9pt}
```

(2) 全局包含文件 myglobal.js **内容**

```
function isEmpty( value ) {
    var temp = value;
    while (temp.indexOf( " " , 0 ) != -1 )
        temp = temp.replace( " " , "" );
     if  (temp == "" ) return true;
    else return false;
}
```

2．选择被批注文件并进行批注的界面

登录后，如果用户角色是教师，则在当前窗口打开一个 fileload.htm **文档，它是选择被批注文件并进行批注的主界面。界面设计如图** 5-16 **所示。**

图 5-16　**设计中的文本批注主界面**

fileload.htm **文档内容如下：**

```
<html>
<head>
<title>file process</title>
<link href="css/site.css" type="text/css" rel="stylesheet">
<script language="javascript" src="script/myglobal.js"></script>
<script language=javascript>
  var fileName;
  var savePath;
```

```
var saveFolder;
var saveName;
var fileObj = new ActiveXObject( "Scripting.FileSystemObject" );
var forReading   = 1;
var forWriting   = 2;
var forAppending = 8;
function readFile() {
  fileName = document.all.homework.value;
  if (fileName=="") {
     alert("请选择一个文本文件");
     return;
  }
  saveName   = createHtmFileName( fileName );
  saveFolder = saveName.substring( 0 , saveName.lastIndexOf( "." ) );
  if( !fileObj.FolderExists( "C:\\" + saveFolder ) )
      var folder = fileObj.CreateFolder( "C:\\" + saveFolder );
  savePath = "C:\\" + saveFolder + "\\" ;
  //显示选择的文本文件
  var textFile = fileObj.OpenTextFile( fileName , forReading );
  if ( textFile.AtEndOfStream ) {
    document.all.t.rows(2).cells(0).innerHTML = "";
    alert( "文件内容为空,请重新读取" );
    return;
   } else {
      var lineNum = 0;
      var str= "";
      //读取第一个字符,如果曾加过编号,该字符一定为<
      var hasNumed = textFile.read(1);
      while( ! textFile.AtEndOfStream ) {
          lineNum ++;
          if (hasNumed != "<")
             str = str + "<b>" + lineNum + ":</b>" ;
          if (lineNum == 1) str += hasNumed;
          while ( ! textFile.AtEndOfLine ) {
              char = textFile.read(1);
              if( char == " " ) {
                //char = " ";
               }
              str += char;
           }
          str += "<br>";
          if( ! textFile.AtEndOfStream ) textFile.skipLine();
      }
      document.all.t.rows(2).cells(0).innerHTML = str;
  }
  textFile.close();
  //显示下面隐藏的表格(带有批注的文本框)
  document.all.t1.style.visibility = "visible";
  document.all.annotation.focus();
}
function createHtmFileName( fileName ) {
```

```
    var str  = fileName;
    var name = fileName.substring( str.lastIndexOf( "\\" ) + 1 );
    var newFileName = name.substring( 0 , name.lastIndexOf( "." ) ) + ".htm";
    return newFileName;
   }
  function updateFile( strSelected ) {
     var textFile = fileObj.OpenTextFile( fileName , forWriting );
     var fileCon = document.all.t.rows(2).cells(0).innerHTML;

     var strWithLink = "<a href=# onclick=\"desOpen('"
                       + strSelected + "','" + getName( fileCon , "A" ) + "')\"
                       name='"
                       + getName( fileCon , "A" ) + "'>"
                       + strSelected + "</a>";
     //add link to string selected
     fileCon = fileCon.replace( strSelected , strWithLink );
     textFile.write( fileCon );
     textFile.close();
    }
   function contProc() {
     var str = document.selection.createRange().text;
     if( isEmpty( str ) ) return false;
     var para = new Array();
     para[0] = str;   //text selected
     para[1] = "";    //link's name
     para[2] = "";    //file contents with descriptions
     var sfe = "dialogWidth:30;dialogHeight:25" ;
     var val = window.showModalDialog( "desCreator.htm" , para , sfe );
     var description = val[0];
     if( description != null && !isEmpty( description ) )
         createHtmlFile( str , description );
     return false;
   }
function createHtmlFile( strSelected , description ) {
   //read contents from new.htm previous so that can be used in description
      writing
  var desFile;
  var fileCon;
  if ( fileObj.FileExists( savePath + saveName ) ) {
     desFile = fileObj.OpenTextFile( savePath + saveName , forReading );
     if ( desFile.AtEndOfStream )
        fileCon = document.all.t.rows(2).cells(0).innerText;
     else  fileCon = desFile.readAll();
     desFile.close();
  } else fileCon = document.all.t.rows(2).cells(0).innerHTML;
  //write contents to new .htm
  var obName = getName( fileCon , "div"  );//id of div with description
  var strWithDescription = "<a href=# name='A_" + obName.substring(2) + "' "
                           + "onmouseover=showDes(1,'" + obName + "') "
                           + "onmouseout=showDes(0,'" + obName + "')>"
                           + strSelected
```

```
                          +  "</a>";
if( !isEmpty( description ) )
       fileCon = fileCon.replace( strSelected , strWithDescription );
   var newFile = fileObj.CreateTextFile( savePath + saveName );
   newFile.write( fileCon );
var stylepro = "'visibility:hidden;position:absolute;
left:'+ window.event.clientX+';top:'+window.event.clientY";
   var desStr = "<table bgcolor=#00ccff>"
                + "<tr><td>" + description + "</td></tr>" + "</table>";
var scriptStr = "<script language=javascript>"
                   +"function showDes(bz,ob){"
                   +"v=\"visible\"\;"
                   +"if(bz==0) v=\"hidden\"\;"
                   +"document.all(ob).style.visibility=v;"
                   +"}"
                   +"</"+"script>"
scriptStr += "<link href=\"css/site.css\" rel=\"stylesheet\" type=\"text/
css\">";
   newFile.write( "<div id=" + obName + " style=" + stylepro + ">" + desStr
   + "</div>" );
   if  ( fileCon.indexOf( scriptStr , 0 ) == -1 )
      newFile.write( scriptStr );
   newFile.close();
   //change current display
   //by add link to text descripted,so that description can be modified
   updateFile( strSelected ); //read content to cells again so that link
   can be displayed
   readFile();
   }
    function getName( fileCon , ob ) {  //create div's id automatically
     var id;
     var temp     = fileCon;
     var obStr    = "<" + ob;
    var firstChar = ob.substring( 0 , 1 );
    var count    = 0;
    while ( temp.indexOf( obStr , 0 ) != -1 ) {
        count++;
       temp = temp.substring( temp.indexOf( obStr , 0 ) + 1 );
     }
     count++;
     id = firstChar + "_" + count;
     return id;
   }
function desOpen( strSelected , linkName ) {
  var para = new Array();
  para[0] = strSelected;   //text selected
  para[1] = linkName;    //link's name
  para[2] = fileObj.OpenTextFile( savePath + saveName , forReading).read-
  All();
  var val = window.showModalDialog("desCreator.htm",para,
"dialogWidth:30;dialogHeight:25");
```

```
  if  ( val[0] == "del" ) deleteDes( val );
     else modifyDes( strSelected , val );
}
function modifyDes( strSelected , val ) {
  var tempFlie;
  tempFile = fileObj.OpenTextFile( savePath + saveName , forWriting );
  tempFile.write( val[2] );
  tempFile.close();
}
function deleteDes( val ) {//delete link from source file
   var linkName = val[1];
   var linkText = document.all(linkName).innerText;
   var linkStr  = document.all(linkName).outerHTML;
   var str1    = fileObj.OpenTextFile( fileName , forReading ).readAll();
   if( str1.indexOf( linkStr , 0 ) == -1 )
        linkStr = "<a" + linkStr.substring( 2 , linkStr.length -2 ) + "a>";
   str1 = str1.replace( linkStr , linkText );
   var  newFile = fileObj.OpenTextFile( fileName , forWriting );
   newFile.write( str1 );
   newFile.close();
   readFile();
   //delete link and div from description file
   var str2 = fileObj.OpenTextFile( savePath + saveName,forReading).read-
   All();
   var linkWithDesStr = "<a href=# name='A_" + linkName.substring(2)
+ "'onmouseover=showDes(1,'d_" + linkName.substring(2)
+ "') onmouseout=showDes(0,'d_" + linkName.substring(2) + "')>";
   var tempStr = str2.substring( str2.indexOf( linkWithDesStr , 0 ) );
var linkText= tempStr.substring( tempStr.indexOf( ">" , 0 ) + 1 ,
tempStr.indexOf( "</a>" , 0 ) );
   linkWithDesStr = linkWithDesStr + linkText + "</a>";
   str2 = str2.replace( linkWithDesStr , linkText );
   var divStr = "<div id=d_" + linkName.substring(2)
              + "style='visibility:hidden;position:absolute;
left:'+window.event.clientX+';top:'+window.event.clientY>"
              + "<table bgcolor=#00ccff><tr><td>" ;
   var tStr   = str2.substring( str2.indexOf( divStr , 0 ) );
   var divText=tStr.substring( tStr.indexOf( "<td>" , 0 )+4,tStr.indexOf
   ("</td>" , 0 ));
   divStr = divStr + divText + "</td></tr></table></div>";
   str2 = str2.replace( divStr , "" );
   //write to source file again
   //write to description file again
   var newFile1 = fileObj.OpenTextFile( savePath + saveName , forWriting );
   newFile1.write( str2 );
   newFile1.close();
}
function saveAnnotation() {
 var desFile;
 var tempFile;
 var annStr  = document.all.annotation.value;
```

```
 if( !fileObj.FileExists( savePath + saveName ) ) {
   if( isEmpty( annStr ) ) {
      alert( "批注内容不能为空" );
      document.all.annotation.focus();
      return;
   }
   tempFile = fileObj.CreateTextFile( savePath + saveName );
   tempFile.write( document.all.t.rows(2).cells(0).innerHTML );
   tempFile.write( "<hr>批注:<br>  " + annStr + "<br>" );
   //alert( "保存成功！" );
   tempFile.close();
   return;
 }
 var str     = fileObj.OpenTextFile( savePath + saveName , forReading ).read-
 All();
 var annStr1 = "";
 // 尚无批注
 if( str.indexOf( "批注" , 0 ) == -1 ) {
   if ( isEmpty( annStr ) ) {
      alert( "批注内容不能为空" );
      document.all.annotation.focus();
      return;
   }
   desFile = fileObj.OpenTextFile( savePath + saveName , forAppending );
   desFile.write( "<hr>批注:<br>  " + annStr + "<br>" );
 } else {
   if ( isEmpty( annStr ) ) {
       var ok = confirm( "删除原有批注吗？" );
       if( ! ok ) {
          document.all.annotation.focus();
          return;
       }
    }
    desFile = fileObj.OpenTextFile( savePath + saveName , forWriting );
    annStr1 = str.substring( str.indexOf( "批注" , 0 ) + 7 , str.indexOf
    ( "<br>" , str.indexOf( "批注" , 0 ) + 8 ) );
    str = str.replace( annStr1 , "  " + annStr );
    desFile.write( str );
 }
desFile.close();
}
-->
</script>
</head>
<body bgcolor=menu>
<table    id=t    border=1       width="600"    bordercolorlight=#333333
bordercolordark=#ffffff>
  <tr height=30>
     <td align=center style="font-size: 18; font-weight: bold" >文本文档批
     注系统</td>
  </tr>
```

```
  <tr height=20>
    <td>
      选择文件<input type=file name=homework onClick="readFile()" title="
      选择文档">
    </td>
  </tr>
  <tr>
    <td oncontextmenu="javascrpt:return contProc();"></td>
  </tr>
  <tr>
    <td>
      <table id=t1 border=0 style="textalign: center; visibility: hidden"
      width="587">
       <tr>
        <td align=center width="15">批<br><br>注</td>
         <td width="558">
           <textarea name=annotation rows=7 cols="90"></textarea><br>
          <input type=button value=保存批注 onclick=saveAnnotation()>
        </td>
       </tr>
      </table>
    </td>
  </tr>
</table>
</body>
```

打开上述 fileload.htm **文件，并选择一个用于批注的文本文档，如图** 5-17 **所示。**

图 5-17　**读取文件后的界面**

说明：

(1**) 在一个** table **中，如果单元格中包含多行文本框(**<textarea >**)，它的属性** cols **的值直接影响表格和单元格的** width **属性，如果** cols **值设置得太大，则即使修改表格和单元格**

的 width 属性，表格似乎也无法缩小。实际上这是由于<textarea >标记把表格撑大了。

（2）oncontextmenu 事件是 IE 新版本中支持的事件，当右击鼠标时发生该事件，可以用于<body>标记，和许多其他的标记，如果要禁止在某个元素上右击，可以赋值为 false，例如<body oncontextmenu="self.event.returnValue=false";>，可禁止用户在网页上右击鼠标。

下面是关于屏蔽鼠标右键和解除屏蔽的操作方法：

```
<script>
fuction stop(){
    alert ('严禁复制')
    return false;
}
document.oncontextmenu=stop
</script>
```

把上面的代码加入到网页中，在该网页中鼠标右键将被屏蔽。无论按 Shift+F10 组合键，还是按键盘上与右侧的 Ctrl 键紧挨着的那个键都不起作用。可以尝试右击鼠标，出现警告窗口时不要松开右键，用左手按键盘上的 Alt+F4 组合键，这时提示窗口关闭，松开鼠标右键，右键仍然不可用。可见常用的解除右键屏蔽的方法均会失效。

对于上述的屏蔽鼠标右键，应该用 JavaScript 来破解。根据上面的代码 document.oncontextmenu=stop，只要能让其中的 stop 失效就可以成功破解。在浏览器地址栏中键入：

```
javascript:alert(document.oncontextmenu='')
```

然后按回车键，此时弹出一个对话框，单击“确定”按钮，然后在页面上右击鼠标就可以看到弹出的菜单了。

（3）对于<input type=file…>，这里用一个 onClick 事件来激活 readFile（读取文件）并显示。实际上，type=file 类型的输入由两个部分组成，前面是一个单行文本输入框，后面是一个按钮。当单击按钮或文本输入框的时候，首先激活 onClick 事件，此时，还没有选择文件。为此可以在 readFile 函数的开始增加以下内容：

```
if (fileName= ="") {
  alert("请选择一个文本文件");
  return;
}
```

上述做法主要是保证第一次单击会打开一个提示窗口“请选择一个文本文件”。由于此时还没有打开系统的“选择文件”对话框，fileName 为空，此时返回，结束 onClick 事件。OnClick 事件处理结束后，才是<input type=file…>标记的打开“选择文件”对话框，这时选择一个文件。要使文件显示出来，仍然要单击一次按钮或者文本框，再次激活 onClick 事件，由于此时 fileName 不为空，将执行 readFile 后面的内容，读出文件并显示。

3．批注、批注修改、删除批注功能设计

用户可以选择一段文本，然后右击鼠标来添加批注。当用户右击时，将打开一个新的窗口文档 desCreator.htm。这是进行批注的添加、修改和删除的页面。例如，在上面打开的

文件中选择第 9 行的 return Str1 **语句，右击鼠标则打开批注页面，如图** 5-18 **所示。**

图 5-18　**批注的添加、修改和删除页面**

然后单击“ 保存 ”按钮返回，可以看到第七行成为一个超链接，如图 5-19 **所示。**

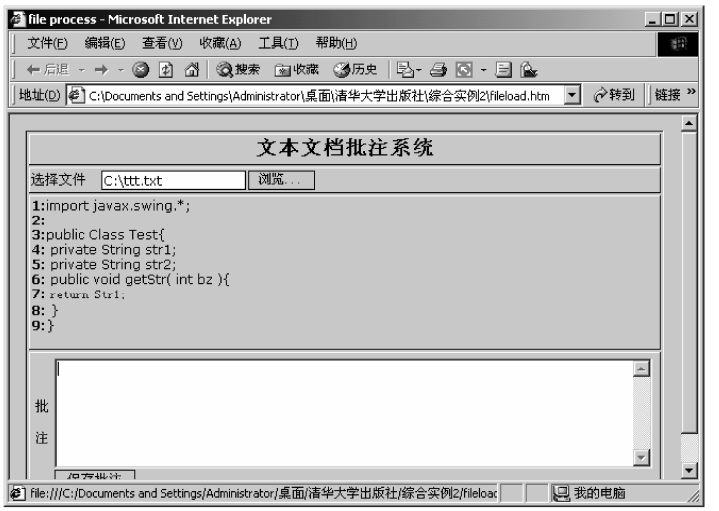


图 5-19　**增加批注后的文档**

desCreator.htm **文档内容如下：**

```
<html>
<head>
<title>批注的添加、修改和删除</title>
<!-- <link href="css/site.css" type="text/css" rel="stylesheet"> -->
<script language="javascript" src="script/myglobal.js"></script>
<script language="javascript">
 var recValue = window.dialogArguments;//received from previous
 function dispStr() {
var strSelected = recValue[0];
```

```
var nameStr   = recValue[1];
document.all.t.rows(1).cells(0).innerText = strSelected;
    // 获取批注的内容
    document.all.des.value = getDescription( nameStr , recValue[2] );
}
 function getDescription( nameStr , desFileCon ) {
    var divName = "d_" + nameStr.substring( 2 );
    var divStr  = desFileCon.substring( desFileCon.indexOf( "id=" + divName ) );
    var description = divStr.substring( divStr.indexOf( "<td>" ) + 4 ,
                                        divStr.indexOf( "</td>" ) );
    return description;
   }
function saveDes() {
    var retValue = new Array();//return to previous
    retValue[0] = document.all.des.value;
    retValue[1] = recValue[1];
    var fileCon = recValue[2];
    retValue[2] = fileCon.replace( getDescription( recValue[1] , recValue[2] ) ,
    retValue[0] );
     if  ( isEmpty( document.all.des.value ) )
        if( recValue[1] == "" ) {
            alert( "批注内容不能为空！" );
           document.all.des.value = "";
           document.all.des.focus();
           return;
        }
     else {
         retValue[0] = "del";
         confirmOp( "确定删除此批注吗？" , retValue );
         return;
       }
    confirmOp( "确定保存吗？" , retValue );
}
function confirmOp( prompt , value ) {
    var ok = confirm( prompt );
    if( ok ) {
       window.returnValue = value;
       window.close();
    }
    else {
       document.all.des.focus();
       return;
    }
 }
 function winClose() {
   var retValue = new Array();
   retValue[0] = "";
   retValue[1] = recValue[1];
   retValue[2] = recValue[2];
   confirmOp( "直接关闭将不会保存填写的批注,确认要关闭吗？" , retValue );
```

```
 }
 function doByKey() {
    if ( window.event.keyCode == 113 ) {   //F2
      saveDes();
    }
    if ( window.event.keyCode == 27 )  {   //Esc
      winClose();
    }
 }
</script>
</head>
<body bgcolor=menu topmargin onload=dispStr() onkeydown=doByKey()>
<div align="center">
  <center>
  <table border="0" cellpadding="0" width="100%" height="100%">
    <tr>
      <td width="100%">
      <table id=t border=1 width=400 bordercolorlight=#333333 bordercol
      ordark=#ffffff align=center >
      <tr>
     <td height=50 align=center style="font-size: 18; font-weight: bold">
     批注的输入、修改与删除</td>
      </tr>
      <tr>
     <td> </td>
      </tr>
      <tr>
      <td>
         请在下面书写批注:<br>
         <textarea id=des cols=61 rows=10></textarea><br><br>
         <input type=button value=保存(F2)  onclick=saveDes()> 
         <input type=button value=关闭(Esc) onclick=winClose()>
     </td>
      </tr>
    </table>
    </td>
    </tr>
  </table>
  </center>
</div>
</body>
```

4．批注生成的 HTML 文档

文本文档批注系统，允许用户添加、修改和删除批注，同时还允许用户添加一个总体的评价意见或说明。系统在原文档的基础上生成一个同名的 HTML 文档，所有的批注都将以 块的形式保存在系统创建的 HTML 文档中，此时，用户可以用浏览器打开这个 HTML 文档。带有批注的地方显示为超链接，鼠标指向该超链接时，页面上将显示所做的批注内容，如图 5-20 所示。

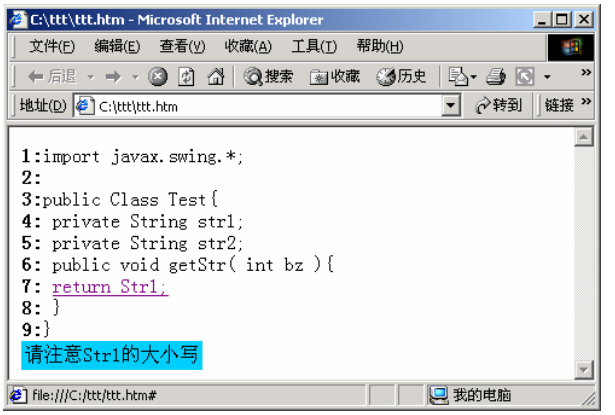


图 5-20 批注后的内容

5．小结

文本文档批注系统比较全面地解释了 JavaScript 中关于文件的处理问题，用到了 window 对象的一个新方法 window.showModalDialog，它和第一个 E-learning 系统相比，对于参数的传递采用了不同的方法，使得在 Web 编程中可以方便地实现传统程序设计中对话框的功能。

习 题 5

1．什么是脚本语言？

2．JavaScript 和 Java 有什么不同？

3．在 JavaScript 中，myArray = new Array(10)是什么意思？如何定义一个 5×6 的二维数 组？

4．什么是网页对象？简述 window 对象和 document 对象常用的属性和方法。

5．画出 navigator 对象树，并简要说明。

6．有那些常用的浏览器事件？它们是如何被触发的？

7．结合图像地图的实例和最后一小节的“文本文档批注”实例，编写一个地图系统，要求当鼠标指向某个省或直辖市时，在鼠标右下角，打开一个多行矩形区域，滚动显示该热点的信息。

第 6 章　服务器端开发

对于 Web 应用，重要的开发任务是服务器端编程，大量的数据管理工作都是通过服务器端程序实现的。服务器端编程与 Web 服务关系密切。针对不同的 Web 服务器环境，所用的开发工具也不相同，这与 Web 服务器中所包含的内置组件有关。

目前，常用的 Web 服务器端开发环境包括 ASP、JSP 和 PHP，它们分别适用于不同的 Web 服务器。例如 Internet 信息服务 IIS，除了提供 Web 服务器、FTP 服务器等服务器的创建和管理工作外，IIS 还包含 Active Server Pages(ASP，活动服务器页)附加组件。通过 ASP 提供的内建对象，组合传统的 HTML 和 XML 页、脚本程序和 ActiveX 组件技术，可以创建和运行动态的、交互功能强大的 Web 服务器应用程序。因此，在 IIS 环境，默认的开发工具是 ASP。同样的，在 Tomcat 服务环境，采用 JSP 等，JSP 作为 Java 技术的一种实现，结合 Servlet 和 EJB，正成为众多 Web 应用首选的工具。

本章将围绕 Java 技术，详细地介绍 Web 应用中服务器端的开发问题，最后对 JSP 技术、ASP 技术和 PHP 技术进行了比较。

6.1　Java 技术及相关概念

JSP 是基于 Java 的技术，和 Java 技术相关的概念很多，发展很快，总有种让人眼花缭乱的感觉。Java 作为最强大的网络语言，随着新技术的不断兴起，新的概念也层出不穷，如 Java Applet、Java Servlet、JavaBeans、EJB、JDBC 等。本节主要介绍与网络开发有关的概念以及它们之间的关系。

6.1.1　Java 概述

Java 是 Sun 公司(Sun Microsystem, Inc.)开发的新一代编程语言。Java 与平台无关，使用它可在各式各样不同的硬件平台、不同操作系统平台的网络环境中进行软件开发，并且具有“一次编写，到处运行”的能力。

1．Java 的出现

1991 年，Sun 公司计划开拓消费类电子产品市场，为电视、烤面包箱等家用消费类电子产品开发一个分布式代码系统，目的是可以通过 Internet 与家电产品进行交互，以便对其进行控制。Sun 公司内部人员把这个项目称为 Green，该小组的领导人是 James Gosling，他是一位非常杰出的程序员。Gosling 于 1984 年加盟 Sun Microsystem 公司，此前在一家 IBM 研究机构工作。他是 SunNeWs 窗口系统的总设计师，也是第一个用 C 实现的 EMACS 文本编辑器 COSMACS 的开发者。

开始，他们准备用 C++语言开发该系统，但是，C++太复杂，且存在安全性问题。于是在 1991 年 6 月份，Gosling 开始准备基于 C++开发一个新的语言，他看着窗外的一棵老橡树，就将这个新的语言命名 Oak，它就是 Java 的前身。Oak 是一种用于网络的精巧而安全的语言，Sun 使用它参加一个交互式电视项目的投标，结果败于 SGI1，为此 Oak 几乎销声匿迹。此时，受到 Mark Ardreesen 开发的 Mosaic 和 Netscape 的启发，他们计划将 Oak 继续完善。但他们发现 Oak 在此之前已是 Sun 公司另一个语言的注册商标，于是他们将这个新的 Oak 改名为 Java，即太平洋上一个盛产咖啡的岛屿(爪哇岛)的名字。

Gosling 在开始写 Java 时，并未局限于扩充语言机制本身，更注重于语言所运行的软硬件环境。他要建立一个系统，这个系统运行于一个巨大的、分布的、异构的网格环境中，完成各种电子设备之间的通信与协同工作。Gosling 在设计中采用了虚机器码(Virtual Machine Code)方式，即 Java 语言编译后产生的是虚拟机，虚拟机运行在一个解释器上，每一个操作系统均有一个解释器。这样一来，Java 就成了平台无关语言。这与 Gosling 设计的 SunNeWs 窗口系统有着相同的技术味道。在 NeWs 中用户界面统一用 Postscript 描述，不同的显示器有不同的 Postscript 解释器，这样便保证了用户界面良好的可移植性。

后来，Patrick Naughton 加入到该项目，Naughton 也是 Sun 公司的技术骨干，曾经是 Open Windows 项目的负责人。整个工作进展神速，经过 17 个月的奋战，整个系统成功完成。它是由一个操作系统、一种语言(Java)、一个用户界面、一个新的硬件平台、三块专用芯片构成的。项目完成后，在 Sun 公司内部做了一次展示和鉴定，观众的反应是：在各方面都采用了崭新的、非常大胆的技术。

2．Java 语言的转折点

1994 年，WWW 已如火如荼地发展起来。Gosling 意识到 WWW 需要一个中性的浏览器，它不依赖于任何硬件平台和软件平台，它应是一种实时性较高、可靠安全、有交互功能的浏览器。于是 Gosling 决定用 Java 开发一个新的 Web 浏览器。这项工作由 Naughton 和 JonathanPayne 负责，到 1994 年秋天，完成了 WebRunner 的开发工作。WebRunner 是 HotJava 的前身，这个原型系统展示了 Java 可能带来的广阔市场前景。WebRunner 改名为 HotJava，并于 1995 年 5 月 23 日发布，在产业界引起了巨大轰动，Java 的地位也随之得到肯定。经过一年的试用和改进，Java1.0 版终于在 1996 年年初正式发布。

随后一些著名的计算机公司纷纷购买了 Java 的使用权，IBM、Apple、DEC、Adobe、SiliconGraphics、HP、Oracle、Toshiba、Netscape 和 Microsoft 等大公司相继购买了 Java 的许可证。另外，众多的软件开发商也开发了许多支持 Java 的软件产品。在以网络为中心的计算时代，不支持 HTML 和 Java，就意味着应用程序的应用范围只能限于同质的环境。

Java 的平台无关性对未来的计算模式产生了革命性的影响，它是继 HTML 后 Internet 发展的又一个里程碑。

1 SGI 公司为美国图形工作站生产厂商。

6.1.2　Java 的技术特征

在 Sun 公司的 Java 语言白皮书中，说明 Java 语言有如下特征：简单、面向对象、分布式、解释执行、健壮、安全、体系结构中立、可移植、高性能、多线程、动态性……

1．简单(simple)

主要体现在以下三个方面：

(1) Java 语言风格来源于 C++，因此 C++程序员可以很快的上手。

(2) Java 摒弃了 C++中容易引发错误的地方，如指针，增加了内存管理等一些新的特色。

(3) Java 提供了丰富的类库，使用户编程更加简单。

2．面向对象(object-oriented)

Java 是面向对象的语言，摒弃了 C++中全局变量等与面向对象思想冲突的内容。

3．体系结构中立(architecture neutral)

一般情况下，网络环境都是异构的，如何使一个应用程序能够在不同硬件、不同操作系统平台的计算机上运行，始终是一个难题。Java 将它的程序编译成一种结构中立的中间文件格式，由 Java 虚拟机来解释执行这种中间代码。这使得 Java 应用程序可以在不同的处理器中执行，现在几乎所有的主流计算机系统都能运行 Java。

4．解释执行(interpreted)

Java 解释器能直接在任何机器上执行 Java 字节码(bytecodes)。

5．可移植(portable)

同体系结构无关的特性使 Java 程序可以在配备了 Java 虚拟机的任何计算机系统上运行。另外，通过定义独立于平台的基本类型及其运算，Java 数据得以在任何硬件平台上保持一致。

6．分布式(distributed)

Java 程序的程序库很容易与 HTTP 和 FTP 等 TCP/IP 协议相配合，从而使 Java 程序可以用 URL 打开并访问网络对象，对程序员来讲，访问方式和访问本地文件系统几乎一样，这就为 Internet 等分布环境提供内容带来了方便。

7．安全性(secure)

Java 是被设计用于网络和分布式环境的，安全性自然是一个重要的考虑因素。Java 的安全性可以从两个方面考虑：

(1) 内存的安全性，如摒弃了 C++中的指针，从而避免了非法内存操作和内存泄漏。

(2) 当用 Java 来创建浏览器内容时，语言功能和浏览器本身的功能结合，使它更安全。

6.1.3 Java 语言的特点

对于习惯了 C++的程序员来说，Java 有许多独到的特点。

1．Java 没有主函数和全局函数

C++并非完全意义上的面向对象语言，最明显的例子是，在 C++中，必须有一个独立的主函数(在 Dos 和 Unix 下是 main()，在 Windows 下是 WinMain())，还要定义可以直接使用的大量的全局函数或者使用 C++中的命名空间“ extern C ”来使用原有的 C 过程调用。

Java 是完全面向对象的语言，它没有主函数等完全孤立的东西，任何函数都必须隶属于一个类。当然，任何程序都有一个入口，Java 程序也有主入口函数，名称同样是 main()，但它必须包含在一个类中，一般形式是：

```
Public class AppName{
   Public static void main(String args[]){
   ...// 代码
   }
}
```

2．Java 没有全局变量

在 Java 程序中，不能在所有类外面定义全局变量，只能通过在一个类中定义公用静态变量来实现全局变量。例如：

```
class GlobalVar {
   public static GlobalVarName;   // 全局变量定义
}
```

因为 public static 成员是一种类属成员变量，只要定义了类，其中的类属成员变量就被分配了空间，而不需要声明类的对象。这样就使得其他类可以访问和操作该变量。

可见，Java 对全局变量进行了更好的封装。而在 C++中，不依赖于任何类的不加封装的全局变量往往会导致系统崩溃。

3．Java 没有结构和联合

在 C++中，为了保持和 C 的兼容，继续支持结构(struct)和联合(union)。但是，在 Java 中，则完全拚弃了这些面向过程时代的概念。

4．字符串不再是字符数组

在 C 和 C++中，字符串操作往往会导致许多内存问题，如内存非法操作、内存泄漏等。因为字符串 char *s 和不定界的字符数组 char s[]是等价的，两者只是为变量 *s* 分配一个指向字符串的指针，存储字符串内容的内存需要申请和释放。

在 Java 中，字符串和字符数组已经被分开了。字符串是一个完全意义上的对象，需要用 String 类来定义。

5．Java 用 Package 来分解命名空间

在大型的软件工程中，怎样保证程序员之间命名的类不重名呢？又如何避免供应商提供的类和程序员自己命名的类不重名呢？虽然有许多方法可以避免重名，但是如果在发现问题以前，工程已经启动了，要修改这些重名问题，就变得非常麻烦。

在 Java 中，引入 Package 概念来解决上述问题。Package 有效的通过集合类来划分命名空间，在不同包内的类可以同名，但不会引起混乱。

Java 并没有彻底解决命名冲突的问题，扩展基类可能引起派生类的冲突。例如用户派生一个类，增加一个方法 foo。如果以后供应商提供新的版本，在新类中也包含了 foo 方法，冲突就出现了。

6．Java 没有独立的头文件

在 C++中，每一个.cpp 实现文件都对应一个.h 头文件，在头文件中，往往包含了.cpp 文件中用到的类的定义。

在 Java 中，关于类的一切东西(属性和方法)都被放到一个单独的文件中，类方法的实现必须在定义的过程中同时进行。

因为类方法的实现代码必须在方法定义时完成，但是一个函数的代码往往是几十行或者几百行的程序代码，这样就使得阅读类很难一下子就看到一个类的全貌。Java 的设计者已经考虑到了这个问题，为此，在 JDK 中提供了两个工具来补偿，Javap 可用来打印类标识，Javadoc 为源代码提供标准的 HTML 文档。

7．数据类型

在 C/C++中，对于不同的平台，编译器对简单数据类型 int、float 等分配不同长度的内存空间，例如 int 在 IBM PC 中为 16 bit，在 VAX-11 中为 32 bit，这导致了代码的不可移植性。但是在 Java 中，对于这些基本数据类型，总是分配固定长度的空间，int 总是 32 bit，这就可以保证 Java 的平台无关性。

在 C/C++中通过指针可以进行强制类型转换，这往往会带来不安全性。在 Java 中，有严格的类型相容性检查。

另外，在 Java 中，没有模板类，而 C++中的模板类(参数化类，即形式参数对应的实际参数是数据类型)可以有效地简化程序代码的编写，但不能减少可执行代码的长度。这意味着在 Java 中，只能靠相似代码的手工复制和修改。

8．常量修饰符 const 的使用限制

在 C++中，const 常量修饰符有着重要的应用，它对于提高代码质量起到了积极的作用。例如，用户可以声明函数参数或者函数的返回值为 const 类型，这样可以有效地防止在函数内部对函数参数的不正当修改或者对返回值的修改。另外，可以将类的一个成员函数声明为 const，表明该方法不能修改该操作的任何对象。

在 Java 中，支持常量操作符、只读变量，通过 final 关键字实现。但是，Java 没有提供一种机制，使得一个变量在参数传递或者返回的过程中只读，或者定义一个不能修改操作

对象的常量方法。上述的省略，为 Java 程序带来了一个可能引起不正当修改错误的隐患。

另外，在 Java 中，宏也不再被支持。

6.2 Java 程序设计基础

Java 程序设计和 C/C++等一般的程序设计语言类似，都包含基本符号、数据、数据类型、表达式、流程控制、类与对象等程序设计的概念，另外，Java 程序设计语言还包含了一些自身的特点，例如接口、包、小程序等。

6.2.1 基本符号

任何一门程序设计语言都有其自身的字符集和基本符号，它们按照语法构成语言的语句，由语句再构成程序。

1．基本字符

Java 语言的基本字符有字母(a、b、…、z，A、B、…、Z)、数字(0、1、…、9)和特殊符号(+、?、*、/、<、=、>等)。

2．保留字(关键字)

由字母构成的具有固定含义的单词，如 if 代表条件语句、while 代表循环语句等。

3．标识符

在 Java 程序中，表示数据类型、类、接口、变量、方法(函数)等名称的符号。标识符是以字母、下划线或 $ 符号开头的字母、数字、下划线组成的字符序列。标识符可为任意长度，区分大小写，用户自定义的标识符不能使用保留字或标准标识符。

4．注释

Java 程序中的注释有以下三种形式：

(1) 以“//”开头的单行注释；

(2) 以“/* … …*/”标记的块注释；

(3) 以“/** … …*/”标记的文档注释，即“/**”和“*/”之间的所有内容(可能占多行)都是注释内容。

6.2.2 数据、数据类型和表达式

程序是对数据的处理，数据分成常量和变量，无论是常量还是变量，每个数据都有一个相应的数据类型。

1．常量和变量

常量是指在程序执行过程中，其值不发生变化的量。根据类型不同，常量可分为整型常量(如 123、?15)、实型(浮点型)常量(如 12.34)、字符常量(如 'x')、布尔常量(如 true)等。

对于字符数据，还有一些特殊字符，在程序中具有特殊含义，如表 6-1 所示。

表 6-1　特殊字符

字符	含义	字符	含义	字符	含义
\n	换行	\f	进纸	\ddd	八进制表示法
\t	水平制表符	\\	表示“\”	\xdd	十六进制表示法
\b	Backspace	\'	单引号	\udddd	十六进制表示 unicode 字符
\r	回车	\"	双引号		

所谓变量，是指在程序执行过程中，其值发生变化的量。每一个变量都有一个变量名，变量名是一个用户自定义标识符，每个变量都有一个数据类型。类型决定变量在内存中所占空间的大小，同时决定变量的取值范围和操作运算。

与 C/C++不同，Java 中没有全局变量，所有的变量都被封装在类中，作为类的成员变量。

2．数据类型

Java 的数据类型可分为基本数据类型和构造数据类型。

(1) 基本数据类型

基本数据类型包括整型(byte、short、int、long)、浮点型(float)、双精度型(double)、布尔型(boolean)和字符型(char)。

(2) 构造类型

在 Java 中，没有 C/C++中面向过程的结构和联合。Java 中的构造类型是用类来描述的，如数组、字符串、对象、类等。

类型用于说明类中的成员变量或成员函数的返回值。一般形式为：

```
<类型名> <变量 1,变量 2,... ...变量 n>;
```

3．表达式

表达式是由常量、变量、函数、运算符以及括号连接而成的式子。常用的表达式运算符如表 6-2 所示。

表 6-2　表达式运算符

运算符	功能	运算符	功能	运算符	功能	运算符	功能
=	赋值运算	&	按位与	!	逻辑非	<	小于
+	加运算	|	按位或	&&	逻辑与	<=	小于等于
?	减运算	~	按位取反	||	逻辑或	==	等于
*	乘运算	^	按位异或	? :	条件运算	!=	不等于

续表

运算符	功能	运算符	功能	运算符	功能	运算符	功能
/	除运算	<<	按位左移	>	大于	++	自加
%	取模运算	>>	按位右移	>=	大于等于	??	自减

在表达式中，运算符的优先级和一般的程序设计语言相似。根据表达式运算结果的不同，表达式又分为算术表达式、字符表达式、逻辑表达式等。

在 Java 表达式中规定，整型、实型、字符型数据可以混合运算。运算过程中，不同类型的数据会自动转换为同一类型，然后再运算。自动转换按低级类型转换成高级类型数据的规则，转换规则为(其中 op 是运算符，如+、?等)：

(1) (byte 或 short)op int → int

(2) (byte 或 short 或 int)op long → long

(3) (byte 或 short 或 int 或 long)op float → float

(4) (byte 或 short 或 int 或 long 或 float)op double → double

(5) char op int → int

如果要把高级类型转换成低级类型，要通过强制类型置换完成，一般形式是：

(类型名)表达式

6.2.3 流程控制

每一种程序设计语言的语句都可以分为顺序、分支和重复语句三种，只是使用的保留字和语法不同而已。

1．赋值语句和输入/输出

无论是哪一种语言(C、Pascal、Fortran 等)，赋值语句、输入和输出语句都是最基本的顺序语句。

(1) 赋值语句

在 Java 中，赋值语句由赋值表达式加一个分号构成，一般形式为：

<变量名> = <表达式>;

赋值语句的逻辑功能是将右边表达式的值赋给左边的变量。

(2) 输入/输出

在 Java 中，可以利用 java.lang.System 类中的两种属性，它提供从键盘输入和从终端输出的方法，即 System.in 和 System.out 方法。

- System.in，具有提供从键盘读取的未经处理的字节的能力。还需要将它转换成 Reader 类并且翻译成字符。利用 BufferedReader 类，这个类允许利用 readline() 方法一次读取一行内容。

例如，如果需要从键盘读取一个串，代码片断如下：

```
BufferedReader lineOfText = new BufferedReader(new InputStreamReader
```

```
(System.in));
String textLine = lineOfText.readLine();
```

其中，Readline()方法将返回整行输入，将它放到 textLine 中并且丢掉异常的数据。

一旦有了输出的队列，就可以简单的分析所需要的数据，例如将输出转换成整数、符点数等，代码如下：

```
int numReadIn = 0;
try {
   numReadIn = Integer.parseInt(textLine);
}
catch (NumberFormatException) {
   System.err.println( “Trouble with the parsing of the number”);
}
```

- System.out，用作发送输出到终端。一般形式为：

```
System.out.println("output string");
```

2．分支语句

分支语句又称选择语句，有如下三种形式。

(1) if 语句

一般形式为：

```
if  (<表达式>)
    <语句>;
```

if 语句首先计算表达式的值，若结果为 true，则执行语句部分，否则执行 if 下面的语句。

(2) if-else 语句

一般形式为：

```
if (<表达式>)
   <语句 1>;
else
   <语句 2>;
```

首先计算表达式的值，若为 true，则执行语句 1，否则，执行语句 2。If-else 语句可以按照需要进行嵌套，但是嵌套降低了程序的可读性。

(3) switch 语句

switch 语句提供了 if-else 语句多层嵌套结构的一个变通形式，可以从多个语句块中选择执行其中的一个。switch 语句提供的功能和 if-else 语句类似，但是可以使代码更加简练易读。一般形式为：

```
switch ( <表达式>) {
    case 常量 1:
        语句 1
      [ break;]
    case 常量 2:
        语句 2
```

```
        [ break;]
    ... ...
    case 常量 n:
        语句 n
        [ break;]
    [default:
        默认处理语句
    ]
}
```

switch 结构在其开始处计算表达式的值。表达式的结果将与结构中每个 case 的常量表达式的值比较。如果匹配，则执行与该 case 关联的语句。Break 语句将使流程退出 switch 语句块，从而执行 switch 语句后面的语句，否则将执行下面的 case 判断。

3．循环控制语句

当部分语句需要反复执行时，需要循环语句。循环语句总是由循环体和循环终止条件两部分组成，在 C 语言中，循环语句有 while,do-while 和 for 三种形式。

(1) while 语句

一般形式为：

```
while (<表达式>)
语句;
```

while 语句首先计算表达式的值，如果结果为 true，则执行循环体。然后无条件的返回 while 语句的开始，继续计算表达式的值；如果条件表达式结果为 false，则结束循环，执行下面的语句。

(2) do-while 语句

一般形式为：

```
do {
   语句(循环体)
} while(<表达式>);
```

直接执行循环体，然后计算表达式的值，如为 true，无条件的返回循环体的第一条语句，继续执行循环体、计算条件表达式的值。如果条件表达式结果为 false，则结束循环，执行 do…while 语句下面的语句。

(3) for 语句

一般形式为：

```
for (表达式 1;表达式 2;表达式 3)
      语句;
```

执行过程如下：

① 计算表达式 1。

② 计算表达式 2，如果为 true，则执行循环体，然后转步骤(3)；否则，结束循环。

③ 计算表达式 3。

④ 无条件转步骤(2)。

⑤ 循环结束，执行 for 语句下面的语句。

在三种循环语句中，任何一种语句都可以写成另外两种语句的形式。也就是说，只要一种形式的循环语句就可以表达程序的重复结构。具体使用哪一种，应该根据三种语句各自的特点和编程人员的编程习惯。

另外，循环体的内部也可以包含循环语句，即循环的嵌套，构成多重循环。

(4) break 语句

break 语句用于跳出最近的循环(语句块{…})，一般形式为：

```
break;
```

例如：

```
for (i=0,p=1;true;i++) {
    p *= 2;
    if (p>1024) break;
}
```

上述代码用于计算 $p=2^i$，当 p 大于 1024 时，退出循环，计算结束。

(5) continue 语句

continue 语句用于结束本次循环，即跳过本次循环中下面尚未执行的语句，接着进入下一次是否执行循环的判断，一般形式为：

```
continue
```

6.2.4 类与对象的概念

在 20 世纪 90 年代以前，自顶向下逐步求精的结构化程序设计是软件开发的主要方法，直到现在，这种结构化的程序设计思想仍然被广泛的采用。Pascal，C，Basic，Fortran 等高级语言很好地实现了结构化编程的思想，通过过程和函数(又称子程序)，把一个复杂的问题划分解成几个相对简单的子问题，如果子问题还比较复杂，再继续划分，最后将划分后的每个问题用过程和函数来实现。

此后，特别是在最近的十年间，面向对象技术正在对人们近半个世纪来的软件开发思想产生深刻的变革。这一技术强调利用软件对象进行软件开发，它将自然界中的物理对象和软件对象相对应，建立了类和对象的概念。由于客观世界的实体和软件结构的对象一一对应，这样就增加了系统的可扩展性和可维护性。

思维方式决定人们解决问题的方式。面向对象技术将自然界的物理对象和软件对象对应起来，在传统的数据结构基础上加入了成员函数(方法)的概念，从而赋予对象以动作。

1．类与对象的概念

类(class)是包含数据和处理这些数据的过程的数据结构。我们可以将类看成是和 int，float 等基本数据类型一样的数据类型，用它来创建数据对象，它指定了相应内存区域的处理和解释规则。

对象(object)是用类来声明的数据结构，如果将类比作数据类型，对象就是相应数据类型的变量。对象是类的实例。

2．类的定义

在 Java 中，类的定义一般形式是：

```
[修饰符] class 类名 [extends 父类名] [implements 接口名列表]
{
    成员变量声明;
    成员函数定义;
}
```

其中修饰符决定类的类型，有以下四种：

- abstract，抽象类，抽象类必须包含至少一个抽象成员函数。抽象类不能够创建对象，需要用其派生类创建。
- final，最终类，说明一个类不能再派生子类。
- public，能被其他类访问。在其他包里要使用该类，需要先用 import 导入，否则只能在自定义的 package 里使用。
- synchronicable，表示所有类的成员函数都是同步的。

上述修饰符可以同时出现两个或以上，但修饰符 abstract 和 final 不能同时使用。

class 为关键字，类名是一个用户自定义的标识符。

如果是派生类,需要通过 extends 给出父类,如果定义接口(interface),需要给出 implents 给出接口名。

花括号内说明类的成员变量和成员函数，又称类的属性(attribute)和方法(method)。

成员变量定义的一般形式是：

```
[修饰符] <类型> <成员变量列表>;
```

其中修饰符为类成员的访问级别声明符号，可以是 private、public、protected。还可以添加的修饰符有 final(最终，初始化后不能再改变其值的变量)、volatile(多线程变量)。这些访问级别可以按任何顺序出现，也可以多次出现。在两个访问级别声明符号之间的成员具有相同的访问控制级别。

如果成员变量包含修饰符 static，此变量称为类变量，不加 static 的变量称为对象变量。

如果变量说明时默认修饰符，则它可被同一包中的所有类访问。

成员函数又称类的方法(method)，是类中定义的可对类进行操作的函数模块。成员函数定义的一般形式为：

```
[修饰符] 返回值类型 <方法名>( [形式参数列表]) [throws 异常列表]
{
// 函数体(Java 程序代码)
}
```

其中修饰符可以是 private、public、protected 和 final(最终，不能由子类改变的方法)、abstract(抽象方法，无方法体)、synchronized(线程同步的方法)、native(本机方法)。修饰

符可以有一个或多个。另外，还可以有 static 来声明静态成员函数，表明此方法为类方法，无 static 修饰的方法为对象方法。

“形式参数列表”给出函数的形式参数及类型，形参之间用逗号分隔。当方法没有形参时圆括号内为空。

3．构造函数

在面向对象技术中，对象是类的实例，每个对象必须按照类的定义来创建，这种机制是通过类的构造函数来实现的。构造函数(constructor)是一种特殊的成员函数，用来在内存中建立具体的对象。构造函数必须申请必要的内存空间，将内存转化为具体的对象，初始化成员变量等。构造函数的名称和类名称相同，一个类可以拥有几个带有不同参数的构造函数。构造函数没有返回类型和返回值。

与一般的函数不同，构造函数不是由用户显式调用(call)的，它是通过编译器来调用的，称为激活(invoke)。

4．构造函数规则和特殊情况

在使用构造函数时候，需要注意以下若干情况：

(1) 构造函数不能描述为 const 和 volatile。

(2) 构造函数不能是 static，因为构造函数需要初始化类的成员变量，但静态构造函数不能访问成员变量。

(3) 构造函数不能被继承。当一个没有构造函数的类从一个含有构造函数的类派生时，将写它自己的构造函数。

(4) 构造函数不能有返回值，也不可以是 void。

(5) 定义一个类时，必须明确定义除默认构造函数和拷贝构造函数以外的所有其他构造函数。

5．创建对象

当定义类后，可以使用 new 来创建对象，一般形式为：

```
对象变量=new 类(参数);
```

如果类的构造函数带有参数，可以通过 new 中类名后面的参数，来激活相应的构造函数。

[例 6-1]　类的定义举例。

在下面的例子中定义了一个类 A，它包含几个属性和几个成员函数。文档名为 exa01.java，内容如下：

```
class CTriangle
{
  private double a,b,c,t;
  public double s=0;
  CTriangle(double x,double y,double z)
  {
    a=x; b=y; c=z;
```

```
  }
  //计算三角形的面积
  public double Area()
  {
    t=(a+b+c)/2;
    s= Math.sqrt(t* (t-a)* (t-b)* (t-c));
    return s;
  }
  public void out1()
  {
    Area();
    System.out.println("Area="+s);
  }
}
class MyTest01
{
  public static void main(String[] args)
  {
    CTriangle myobj = new CTriangle(10,20,15);
    myobj.out1();
  }
}
```

在 Dos **提示符下，执行** javac exa01.java **命令，编译，生成每个类的**.class **文件；然后执行** java MyTest01 **运行上述程序，输出结果如下：**

```
Area=72.61843774138907
```

[**例** 6-2] **类的构造函数激活次序举例。**

在下面的例子中，定义了三个类 A、B、C**，其中** C **中包含两个对象成员，来展示对象构造函数的激活次序。文档名为** exa02.java**，内容如下：**

```
class A {
   int x,y;
   A(int a,int b)
   {
     x=a;  y=b;
   }
   A()
   {
     x=0; y=0;
     System.out.println("A constructor\n");
   }
}
class B {
   B()
   {
     System.out.println("B constructor\n");
   }
}
```

```
class C {
   public A a=new A();
   public B b=new B();
   C()
   {
     System.out.println("C constructor\n");
   }
}

class MyTest02
{
  public static void main(String[] args)
  {
    C myObj=new C();
  }
}
```

执行 java MyTest02**，输出结果为：**

```
A constructor
B constructor
C constructor
```

6.2.5　封装和抽象

在面向对象技术中，一个主要的目标就是对象的封装和抽象。封装(encapsulation**)是指对象可以拥有私有元素，将内部细节隐藏起来的能力。封装将对象封闭起来，管理对象的内部状态。而抽象则和对象的外部状态紧密相关，它通常用来描述对象所表示的具体概念、对象所完成的任务以及处理对象的外部接口。抽象处理的是对象的可见外部特征。**

在 C **中，通过关键词** static **可以实现有限的封装。当一个变量在一个函数内部被说明成** static **形式时，该变量就只在函数中存在，并且只在函数内部有效。另外，一个全程变量被说明成** static **形式时，该变量只在其所在的文件有效，这样可以避免不同的文件中全局变量重名。**

在 C++ **和** Java **等面向对象的程序设计语言中，类的每一个成员都被声明成** public**、**private **和** protected **型，用这些关键词来实现数据的抽象和封装。**

(1**) 关键词** public

类中所有 public **成员构成类的接口，它们是类的抽象性的表现。出于简单、经济和安全的考虑，类的公开元素越少越好。但是，类又必须和外部打交道，**public **成员是不可缺少的。**

(2**) 关键词** private

类中的 private **成员只能被类的成员函数、友元类或外部友元函数访问。从而实现类的封装性。在默认状态下，所有的成员都是私有的。**

(3**) 关键词** protected

在面向对象技术中，派生是类的重要性质。类的 private **成员将不能被派生类中的成员**

访问，这就大大限制了类的灵活性。类的 protected 成员可以被类的派生类成员访问。

类成员访问规则如图 6-1 所示。

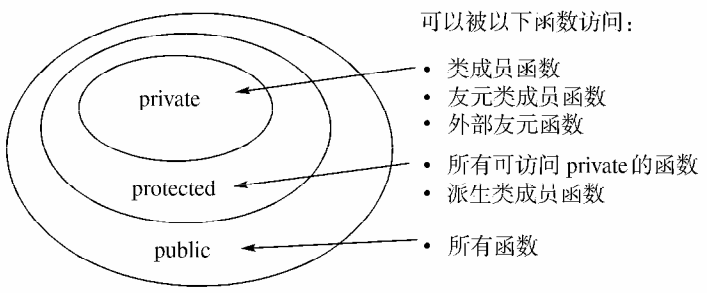


图 6-1 类成员访问规则

6.2.6 静态成员

支持封装和抽象的另外机制还包括关键词 static。当一个成员被声明为 static 时，则该成员在程序中只有一个拷贝存在，而不是在每个对象中都有一个拷贝。所有的对象共享类中的静态成员。在一些面向对象的程序设计语言中，静态成员被称为类变量或类属变量。

例如，定义一个含有静态成员变量的类如下：

```
class CS {
   static int a;
   int b,c,d;
};
CS objs[3];
```

在类 CS 中，包含四个成员变量，其中成员变量 a 为静态成员变量。数组 objs 包含三个 CS 类对象，由于 CS 类包含一个静态成员变量 a，因此三个 CS 对象共享一个静态成员变量 a，每个对象都包含自己的普通成员变量 b,c,d，如图 6-2 所示。

还可以把一个成员函数声明成 static，这意味着在没有任何该类对象的情况下，它仍然可以被执行。正因如此，静态成员函数能够完成不需要任何对象成员变量的操作。

静态成员变量和静态成员函数为类存储和管理属于整个类而不是个别对象的信息提供了一种方法。

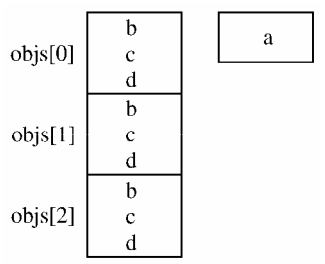


图 6-2 类 CS 的成员

6.2.7 类的继承性与派生类

从现存的对象出发建立一种新的对象类型，使它继承原对象的特点和功能，这就是对象的继承性。继承是许多层次对象的自然描述。

从现存类派生出新类时，现存类称为基类(base class)，派生出的新类称为派生类。派

生类可以对基类作如下变化：

- 增加新的成员变量
- 增加新的成员函数
- 重新定义已有的成员函数

即子类可以对父类的方法覆盖(overriding)或重载(overloading)。所谓覆盖，是指在子类中定义了与父类中同名的函数，这时将根据当前的对象类型执行子类中的代码，即父类中的代码被覆盖。所谓方法重载是子类与父类的方法同名，但是形式参数不同时，将重新实现该方法。

- 改变现有成员函数的属性

派生类不能删除基类的成员变量和成员函数，实际上派生类往往是基类的扩充，是一种具体化和完善的过程。

类的派生创建了一个类族，派生类的对象也是基类的一个对象，它可用在基类对象可被使用的任何地方。可用多态成员函数来调整这种关系，以使派生类在某些地方和它的基类一致，而在另外一些方面表现出其自身的行为特征。

类的派生是一种演化过程，即通过扩展、更改和特殊化从一个已知类出发来建立一个新的类。类的派生建立了一个具有共同关键特性的类族，从而实现代码的重用。假设从一个已知基类 A 建立一个派生类 B，一般形式为：

```
class B  extends  A {
    // 派生类 B 的成员说明
};
```

“class B extends A”称作“类 B 由 A 派生”，它告诉编译器类 B 是一种 A，对基类 A 所作的修改和添加在括号内给出。在派生类对象中，编译器在内存中总是先放入基类的成员，后面是派生类独有的成员，如图 6-3 所示。

图 6-3　派生类对象内存结构

对于派生类的成员的访问原则，见表 6-3。

表 6-3　基类和派生类的访问特性

基类 class B { … }	派生类 class D extends B { … }
不可访问成员	不可访问
private 成员	不可访问
protected 成员	保护
public 成员	公有

所谓不可访问成员，是一个派生类中的概念。根类(不是从其他类派生出来的类)中不存在不可访问的成员。但是在派生类中，不可访问是存在的，如根类中的 private 成员，就构成派生类的不可访问成员。如果再从派生类派生新的类，则派生类的不可访问成员和私有成员就构成了它的派生类的不可访问成员，以此类推。

在一些较复杂的类设计中，常使用关键字 this 和 super 来指明变量或方法是属于子类或父类。

[例 6-3] 对象成员的访问举例。

下述例子演示了类及其派生类中成员的访问规则。文档名 exa03.java，代码如下：

```
class B {
   private int x,y;
   protected int z;
   B()        // 构造函数 1
   {
     x=0;  y=0;  z=0;
     System.out.println("A constructor\n");
   }
   B(int a,int b)  // 构造函数 2
   {
     x=a; y=b;
     z=x+y;
   }
}
class D extends B{
   D(int x,int y)    // 构造函数
   {
     System.out.println("D constructor\n");
   }
   public void out()
   {
     System.out.println("D public out Access class B:z="+z);  // 访问父类的
      protected 成员
   }
}
class MyTest03
{
  public static void main(String[] args)
  {
    D myd= new D(10,20); // 创建派生类对象
    myd.out();          // 外部访问
  }
}
```

执行 java MyTest03，运行结果为：

```
A constructor
D constructor
D public out Access class B:z=0
```

从上面的例子可以看出，创建一个派生类的对象，首先执行父类的构造函数 1，然后再执行本身的构造函数。在派生类中，可以访问父类的 protected 成员。

问题是在一个派生类中，如何执行父类的带有参数的构造函数呢？在 C++中，可以在派生类的构造函数首部写出要调用的父类的构造函数，例如：

```
class Derive::public Base {
    public :
```

```
    Derive(int m=0,int n=0):Base(m) {
    ....
    }
....
}
```

上述代码表示，在创建派生类 Derive 对象时，调用的父类的构造函数是 Base(m)，而不是默认构造函数。

6.2.8　多态性和抽象类

多态性(polymorphism)是指相同的操作作用于不同类的对象时，可以得到不同的结果。多态主要表现为编译时多态和运行时多态两种形式。

所谓编译时多态主要包括操作符重载和函数(方法)重载。在程序编译时，系统根据传递参数的个数和参数类型等信息决定要连接的函数。运行时多态是指直到系统运行时才根据操作对象的类来决定执行何种操作，即执行哪个类中的方法。

运行时多态极大地增强了类的抽象能力，在 C++中，运行时多态是通过在父类中使用虚函数(Virtual)来实现的。在 Java 里没有虚函数的概念，有抽象函数，但抽象函数并不是 C++虚函数在 Java 里的等价物，两者是有区别的。下面是 C++ 和 Java 关于多态性和抽象类概念的对比。

表 6-4　C++与 Java 在多态、抽象类方面的对应概念

C++	Java
虚函数	普通函数
纯虚函数	抽象函数
抽象类	抽象类
虚基类	接口

在 Java 中，抽象函数必须定义在抽象类里面，而且没有方法体。例如：

```
 public abstract TestAbstract {
       public abstract void say();  // 抽象函数
}
```

所谓抽象类，就是用 abstract 修饰符说明的类。当一个类包含一个 abstract 成员函数时，必须将这类定义为 abstract 类。然而，并不是抽象类的所有成员函数都是 abstract 的，抽象类不能有私有成员函数和静态成员函数。

1．重写与重载

方法的重写(overriding)和重载(overloading)是 Java 多态性的不同表现。重写是父类与子类之间多态性的一种表现，重载是一个类中多态性的一种表现。如果在子类中定义某方法与其父类具有相同的名称和参数，则称该方法被重写。子类的对象使用这个方法时，将调用子类中的定义，对它而言，父类中的定义如同被“屏蔽”了。如果在一个类中定义了

多个同名的方法，它们或有不同的参数个数或有不同的参数类型，则称为方法的重载。

[例 6-4] 编译时多态性举例。

定义一个类，包含两个成员函数 Area，分别用于计算矩形和圆的面积，文档 exa04.java 代码如下：

```
class CFigure
{
  public double s=0;
  public double Area(double a,double b)
  {
    s=a*b;
    return s;    //计算矩形的面积
  }
  public double Area(double r)
  {
    s=3.14*r*r;
    return s;    //计算圆的面积
  }
}
class MyTest04
{
  public static void main(String[] args)
  {
    CFigure myobj = new CFigure();
    System.out.println("Rectangle area="+myobj.Area(10,20));
    System.out.println("Circle area="+myobj.Area(10));
  }
}
```

在 Dos 提示符下，执行 javac exa04.java 命令，编译，生成每个类的.class 文件；然后执行 java MyTest04 运行上述程序，输出结果如下：

```
Rectangle Area =200.0
Circle Area =314.0
```

[例 6-5] 运行时多态性举例。

下面是一个演示 Java 中类的多态性的例子，文档名为 CMyFigure.java，内容如下：

```
public abstract class CMyFigure        // 定义抽象类
{
  //public abstract void Draw();
  public abstract void Area();         // 声明一个抽象函数
}
class CRectangle extends CMyFigure     // 派生类
{
  float height,width;
  float s;
  CRectangle(int x,int y)
  {
    height=x;
```

```
    width =y;
  }
  public void Area()                    // Overriding 父类中的抽象函数
  {
    s=height*width;
    System.out.println("The Area of the Rectangle="+s);
  }
}

class CCircle extends CMyFigure         // 派生类
{
  float r,s;
  CCircle(float x)
  {
    r=x;
  }
  public void Area()                    // Overriding 父类中的抽象函数
  {
    s=(float)3.14*r*r;
    System.out.println("The Area of the Circle="+s);
  }
}

class MyTest05
{
  public static void main(String[] args)
  {
    /*
    CMyFigure obj1,obj2;
    obj1 = new CRectangle(10,20);
    obj1.Area();
    obj2 = new CCircle(10);
    obj2.Area();
   */
   CMyFigure objs[];   // = new CMyFigure[2]; 创建数组
   objs[0] = new CRectangle(10,20);
   objs[1] = new CCircle(10);
   for (int i=0;i<2;i++)
       objs[i].Area();
  }
}
```

在上述代码中，将 obj1 **和** obj2 **均声明为** CMyFigure **类，但在程序运行时实际存储的是派生类** CRectangle **和** CCircle **对象。执行** obj1.Area()**和** obj2.Area()**的代码是在程序运行执行了** new **对象后确定的，这是一种运行时多态。**

为了展示在 Java **中数组的应用，用户也可以用数组类型重写上述功能的代码，可以进一步展示运行时多态的强大功能，也可以用它编写更加精致的代码。**

在 Dos **提示符下，执行** javac CMyFigure.java **命令，编译，生成每个类的**.class **文件；然后执行** java MyTest05 **运行上述程序，输出结果如下：**

```
The Area of the Rectangle=200.0
The Area of the Circle=314.0
```

6.2.9 接口

在 Java 中，关于接口的讨论很多，也有众多的不同理解。那么，接口究竟是什么呢？

1．什么是接口

在一个复杂的面向对象的系统中，实现一个有更多方法的新类是经常遇到的。当一个类需要从多个基类派生时，派生类将继承多个基类的特征，在 C++中，这样的机制称为多重继承。在面向对象的程序设计中，继承关系一直存在很多的争议，特别是多重继承。

Java 没有多重继承，可以通过接口来实现多种功能。在 Java 中，所谓接口(Interface)是一组没有给出实现细节的操作(方法)的集合。它需要别的类来实现接口给出的每一个方法，一个类可以实现一个或多个接口。如果一个类实现了某个接口，就相当于声明它能够完成某项工作。

关于接口，不同的 Java 开发人员会有不同的体会。在某次 Java 用户会议上，曾经有人向 Java 之父 Jams Gosling 提过这样的问题：“如果你重新构造 Java，你想改变什么？”“我想抛弃 classes”，他回答。在笑声平息后，他解释说：“真正的问题不是由于 class 本身，而是实现继承(extends 关系)。接口继承(implements 关系)是更好的。你应该尽可能地避免实现继承。”

2．创建接口

接口定义的一般形式为：

```
public interface <Interfacename> [extends Superinterfaces]
{
  成员变量声明(常量) ;
  成员函数(方法)声明;
}
```

在接口定义中，public 指示了接口可以在任何包中的任何类中使用。如果没有指定接口为 public，接口就只能在定义该接口的包中各类中使用了。

一个接口可以扩展另外的接口，这跟类可以扩展一样。但是，类只能扩展一个另外的类，而接口可以扩展任意个接口。Superinterfaces 列出所有的被扩展的接口，以逗号分隔。

接口可以包含常量声明以及方法声明。所有定义在接口中的常量可以是 public、 static 和 final。定义在接口中的成员变量不能使用 transient、volatile 或者 synchronized 修饰符。同样不能在声明接口的成员时使用 private 和 protected 修饰符。

接口中的方法声明后紧跟着一个分号，因为接口中的方法不需要给出具体的实现代码。因此，所有定义在接口中的方法可以隐含地为 public abstact 方法。

3．接口和类

接口的定义和类的定义类似，但是接口不是一个类，而是对符合接口要求的类的一套规范。接口说明了实现接口的类该做什么而不指定如何去做，一个类可以实现一个或多个接口。

实现一个接口需要两个步骤:

(1) 声明类需要实现的接口，声明一个类实现一个接口需要使用 implements 关键字。

(2) 提供接口中的所有方法的定义。为了使用接口，需要编写执行接口的类。一个类实现了某个接口，即这个类实现了在接口中声明的所有方法，相当于声明能够完成某项工作。

实现接口的类继承了定义在接口中的常量，这些类可以使用简单的名字来引用接口中的常量。

4．接口与抽象类

根据接口的定义，可以看到接口和抽象类相似，都是只给出方法，没有给出方法具体的实现细节。那么两者是一种什么样的关系呢？

我们可以把接口理解为各个功能模块之间进行联系的协议，为了保证整个系统中各个功能模块的联系，系统就需要定义一系列的接口，这些接口由系统中的功能模块来实现。接口如同功能模块之间通信的一种规范，一个功能模块可以实现多个接口。

例如，在计算机网络中，可以定义三个网络接口，分别是 RJ45 接口(双绞线)、AUI 接口(粗同轴电缆)和 BNC 接口(细同轴电缆)。可以定义网卡和网络设备(如 hub、switch、router)类，来实现上述的一个或多个接口，这样计算机、hub、switch 和 router 就可以连接到网络了。

上述思想不适于用抽象类来实现。

[例 6-6]　一个抽象类和接口应用实例。

通过下面的例子，演示抽象类、接口、接口的实现的定义和应用，文档名为 HelloWorld.java，内容如下：

```
public class HelloWorld
{
  public static void main(String[] args)
  {
    Dog  animal1 = new Dog();
    Cat  animal2 = new Cat();
    Duck animal3 = new Duck();
    System.out.println("A dog says " +animal1.getHello()
                    +", when scared says: " +animal1.getHello(Animal.SCARED)
                    +", is carnivorous: " +animal1.isCarnivorous()
                    +", is a mammal: " +animal1.isAMammal());
    System.out.println("A cat says " +animal2.getHello()
                    +", when comforted says: " +animal2.getHello(Animal.
                    COMFORTED)
                    +", is carnivorous: " +animal2.isCarnivorous()
                    +", is a mammal: " +animal2.isAMammal());
    System.out.println("A duck says " +animal3.getHello()
                    +", when scared says: " +animal3.getHello(Animal.SCARED)
                    +", is carnivorous: " +animal3.isCarnivorous()
```

```
                    +", is a mammal: " +animal3.isAMammal());
  }
}
//定义抽象类 Animal
abstract class Animal
{
  public static final int SCARED = 1;
  public static final int COMFORTED = 2;
  public boolean isAMammal()   //哺乳动物
  {
    return(true);
  }
  public boolean isCarnivorous()  //食肉类的
  {
    return(true);
  }
  abstract public String getHello();
  abstract public String getHello(int mood);
}
//定义接口 LandAnimal
interface LandAnimal
{
  public int getNumberOfLegs();
  public boolean hasATail();
}
//定义接口 WaterAnimal
interface WaterAnimal
{
  public boolean hasGills();
  public boolean laysEggs();
}
//定义派生类 Dog,实现接口 LandAnimal
class Dog extends Animal implements LandAnimal
{
  // 重载父类的方法
  public String getHello()
  {
    return("Bark");
  }
  public String getHello(int mood)
  {
    switch (mood) {
      case SCARED:
          return("Growl");
      case COMFORTED:
          return("");
    }
    return("Bark");
  }
  // LandAnimal 接口实现
```

```
  public int getNumberOfLegs()
  {
     return(4);
   }
  public boolean hasATail()
  {
    return((true);
  }
}
//定义派生类 Cat,实现接口 LandAninal
class Cat extends Animal implements LandAnimal
{
  // 重载父类的方法
  public String getHello()
  {
    return("Meow");
  }
  public String getHello(int mood)
  {
     switch (mood) {
        case SCARED:
            return("Hiss");
        case COMFORTED:
            return("Purr");
     }
     return("Meow");
   }
   // LandAnimal 接口实现
   public int getNumberOfLegs()
   {
     return(4);
   }
   public boolean hasATail()
   {
     return(true);
   }
}
//定义派生类 Duck,实现接口
class Duck extends Animal implements LandAnimal, WaterAnimal
{
   // 重载父类的方法
   public String getHello()
   {
      return("Quack");
   }
   public String getHello(int mood)
   {
      switch (mood) {
         case SCARED:
             return("Quack, Quack, Quack");
```

```
            case COMFORTED:
                return("");
        }
        return("Quack");
     }
    public boolean isAMammal()
    {
       return(false);
    }
    public boolean isCarnivorous()
    {
        return(false);
    }
    // WaterAnimal 接口实现
    public boolean hasGills()
    {
       return(false);
    }
    public boolean laysEggs()
    {
       return(true);
    }
    // LandAnimal 接口实现
    public int getNumberOfLegs()
    {
       return(2);
    }
    public boolean hasATail()
    {
       return(false);
    }
}
```

执行 javac HelloWorld.java**，编译该程序，生成相应的**.class **文件。然后执行** java HelloWorld**，运行该程序，输出结果如下：**

```
A dog says Bark, when scared says: Growl, is carnivorous: true, is a mammal:
true
A cat says Meow, when comforted says: Purr, is carnivorous: true, is a mammal:
true
A duck says Quack, when scared says: Quack, Quack, Quack, is carnivorous:
false, is a mammal: false
```

6.2.10 包

在 Java **中，为了管理大型名字空间，避免名字冲突，特别引入包(**package**)的概念，包由一组类和接口构成。一般情况下，每一个类或接口都被存储在不同的文件中，为了管理和使用方便，对于那些相关的类和接口可以绑定到一个包中。例如，**Java **的基本类在** java.lang **中，而用于输入和输出的类则在** java.io **中。**

使用包实际上还定义了一个名字空间，一个包中的类和接口的名字与其他包中的名字不会冲突。同时，还可以约束包内的类和外部的类的访问权限。

1．定义包

使用 package 语句可以将一个编译单元定义成包。如果使用 package 语句，编译单元的第一行必须无空格，也无注释，格式如下：

```
package 包名;
```

若编译单元无 package 语句，则该单元被置于一个默认的无名包中。

按照习惯，包名是由“.”号分隔的单词构成，第一个单词通常是开发这个包的组织的名称。

例如有一个 java 文件，文件名为 b.java，内容如下：

```
package yy.hao.ll;
public class B {
   public void add(int i,int j)
   {
     System.out.println(i+j);
   }
}
```

执行 javac -d　D:\　B.java 命令，则在 D:\目录下创建一个全路径为 D:\yy\hao\ll 的文件夹，包含编译后的类文件 B.class。

现在这个包已经创建好了，要使用这个包中的类 B，需要导入，或者把 D:\yy\hao\ll 设置在环境变量 classpath 里。

2．使用包中的类和接口

要使用定义在一个包中的类和接口，可以通过 import 关键词，主要有两种形式。

(1) 在每个引用的类和接口前面给出它们所在的包的名字，一般形式是：

```
acme.project.FooBar  obj = new acme.project.FooBar();
```

(2) 使用 import 语句，导入类或接口，或包含它们的包。导入的类和接口的名字在当前的名字空间可用。导入一个包时，则该包中的所有的公有类和接口均可用，形式如下：

```
import java.util.*;
mydate=new Date();
```

其中，第一个语句表示 java.util 中所有的 public 类被导入当前包，“*”表示导入包中的所有类。在使用一个外部类或接口时，要声明该类或接口所在的包，否则会产生编译错误。

3．Java 标准包

JDK 为用户提供了很多标准的 Java 类和接口，这些包是编写 Java 程序所必需的，知道了每种包所包含的类和接口，并且熟悉这些类和接口是每个 Java 编程人员都应该掌握的基本技能。

(1) java.applet　包含了一些设计小应用程序(Applet)的类和接口，有一个类 Applet 和三个接口：AppletContext、AppletStub 和 AudioClip。

(2) java.awt 是一个窗口工具箱包(Abstract Window Toolkit，awt)，包含一些 GUI 界面相关的类，例如：Button、Checkbox、Choice、Component、Graphics、Menu、Pane1、TextArea 和 TextField。

(3) java.io 包含文件输入/输出类，FileInputStream 和 FileOutputStream。

(4) java.lang 包含 Java 语言类、线程、异常、系统、整数等相关的类，是 Java 程序中默认加载的一个包。

(5) java.net 该包支持 TCP/IP 协议，并包含 Socket 类、URL、与 URL 相关的类。

(6) java.util 这个类包含一些程序的公用类，如 Date、Dictionary 类等。

6.2.11 Java Applet

Java Applet(Java 小程序)是指用 Java 编写的能够在 Web 页中运行的应用程序，它的可执行代码称为 class 文件。它具有安全，功能强和跨平台等特性。IE、Netscape 和 HotJava 等主流浏览器都能够显示包含 Applet 的页面。

1．在 HTML 中使用 Java Applet

首先编写 Apple 源程序，扩展名为.java。编辑完成后使用 javac 命令将它编译转换成字节码文件(.class 文件)，然后再在 HTML 文档中通过加入<applet>标记的方式将 Java 小程序引入到 HTML 文档中。

一般形式是：

```
<applet width=" " height=" " code="myJava.class" codebase=" ">
浏览器不支持 Java 的提示信息。
</applet>
```

通过<applet>标记来定义 Applet 程序的使用，<applet>标记所包含的属性见表 6-5 所示。

表 6-5 Applet 标记常用属性

属性	作用
Code	Applet 的 class 文件名
Codebase	指定 Applet(.class 文件)所在的 URL 地址。它可以是绝对地址，如 www.sun.com。也可以是相对于当前 html 文档所在目录的相对地址，如/AppletPath/Name。如果不指定 codebase 属性，浏览器将使用和 html 文件相同的 URL
Alt	浏览器不支持 Java Applet 时显示的信息
width、height	Applet 窗口的大小
align	Applet 窗口的显示位置，取值可是：top、middle、bottom
vspace 、hspace	在 Applet 窗口周围的水平和垂直空白条的尺寸
name	把指定的名字赋给 Applet 的当前实例
param	指定 Java Applet 参数。格式：PARAM Name="name" VALUE="Liter"

在 FrontPage 中，可以通过“插入”、“高级”、“Java 小程序”菜单项在 HTML 文档中插入一个 Java 小程序。

Applet 是从远程服务器上下载到本地运行的，出于安全的考虑，对它的运行进行了必要的限制。例如，永远无法运行本地机上的程序；只能与它所在的服务器联系；无法对本地机上的文件进行读写操作；除了本地机使用的 Java 版本号、操作系统名称及版本号文件名分隔符、文件路径外，无法获得本地机的其他信息。

2．Applet 类

在包的介绍中，曾谈到过 Applet 包，其中包含了 Applet 类。Applet 类是所有 Applet 应用的基类，所有的 Applet 小应用程序都必须继承该类。例如：

```
import java.applet.*;
public class myApplet extends Applet
{
... ...
}
```

Applet 类的基本方法如表 6-6 所示。

表 6-6　Applet 类的基本方法

方法	功能
Applet()	Applet 类惟一的构造函数
public final void setStub(AppletStub stub)	设置 Applet 的 stub。stub 是 Java 和 C 之间转换参数并返回值的代码位，由系统自动设置
public boolean isActive()	判断一个 Applet 程序是否在活动状态
public URL getDocumentBase()	检索表示该 Applet 运行的文件目录的对象
public URL getCodeBase	获得该 Applet 代码的 URL 地址
public String getParameter(String name)	得到由 name 指定的参数值
public AppletContext getAppletContext()	返回浏览器或小应用程序观察器
public void resize(int width,int height)	调整 Applet 运行的窗口尺寸
public void resize(dimension d)	调整 Applet 运行的窗口尺寸
public void showStatus(String msg)	在浏览器的状态条上显示指定信息
public Image getImage(URL url)	按 url 指定的地址装入图像
public Image getImage(URL url,String name)	按 url 指定的地址和文件名装入图像
public AudioClip getAudioClip(URL url)	按 url 指定的地址获取声音文件
public AudioClip getAudioClip(URL url,String name)	按 url 指定的地址和文件名获取声音
public String getAppletInfo()	返回 Applet 的作者、版本、版权等信息
public Sring[ ] [ ] getParameterInfo()	返回描述 Applet 参数的字符串数组
public void play(URL url)	加载并播放一个 url 指定的音频剪辑
public void destroy()	撤消 Applet 及其所占用的资源。若该 Applet 是活动的，则先终止该 Applet 的运行

Applet 类还有四个基本方法，用来控制其运行状态。

- init()方法　Applet 程序运行前的初始化工作。当一个 Applet 被系统调用时，系统首先调用该方法。通常可以在该方法中完成从网页向 Applet 传递参数，添加用户界面的基本组件等操作。
- start()方法　系统在调用完 init()方法之后，将自动调用 start()方法。而且，每当用户离开包含该 Applet 的 Web 页后再次返回时，系统将再次执行 start()方法。这就意味着 start()方法可以被多次执行，而不像 init()方法。因此，可把只希望执行一遍的代码放在 init()方法中。可以在 start()方法中开始一个线程，如继续一个动画、声音等。
- stop()方法　用户离开 Applet 程序所在的网页时，运行该方法。因此它也是可以运行多次。
- destroy()方法　停止 Applet 程序的运行，撤消 Applet 及其所占用的资源。在关闭浏览器时执行该方法。

[例 6-7]　使用 JavaApplet 举例。

下面是一个创建 Java Applet 并在网页中使用的例子，以此来说明 Java Applet 的使用步骤。

(1) 编辑 Java Applet 源程序，假定文档名为 helloApplet.java，内容如下：

```
import java.awt.*;
import java.applet.*;
public class helloApplet extends Applet
{
  public void paint(Graphics g)
  {
   g.drawString("Hello World!",5,35);
  }
}
```

将该文档保存在 D:\MyJava 文件夹下。

(2) 把 Applet 的.java 源程序转换为字节码.class 文件。

使用 javac helloApplet.java 命令编译该文件，生成.class 文件。

(3) 编制使用 class 的 HTML 文件。在 HTML 文件内放入必要的<applet>语句。

在运行创建的 helloApplet.class 之前，需要创建一个 HTML 文件，appletviewer 或浏览器将通过该文件访问创建的 Applet。

创建的 HTML 文档 myApplet.htm，保存在和.class 相同的文件夹中，内容如下：

```
<html>
<title>helloApplet</title>
<applet code="helloApplet.class" width=200 height=100>
</applet>
</html>
```

双击该 HTML 文档，将在浏览器中看到 Applet 的执行结果。

3．Applet 交互

Applet 支持鼠标、键盘等多种事件，从而使得用户可以对 Applet 进行交互控制。

（1）Applet 的鼠标事件

Java Applet 支持的鼠标操作有 mouseUp、mouseDown、mouseDrag、mouseEnter 和 mouseExit 等。在 Applet 类中包含了这些事件的处理方法，用户只要在多个方法中加入适当的事件处理代码，即可以完成相应事件的处理工作。这些方法的首部如下，其中形参 *x* 和 *y* 表示事件发生时鼠标的位置。

- public boolean mouseDown(Event event，int x，int y)　按下鼠标按键。
- public boolean mouseUp(Event event，int x，int y)　放开鼠标按键。
- public boolean mouseDrag(Event event，int x，int y)　拖动鼠标。
- public boolean mouseEnter(Event e，int x,int y)　鼠标指针指在对象上。
- public boolean mouseExit(Event e，int x,int y)　鼠标指针离开对象。

（2）Java Applet 响应的键盘事件

Java 目前支持两种键盘事件：KEY_PRESS 和 KEY_RELEASE。当按下键盘的某一键时发生 KEY_PRESS 事件；当放开被按下的键时发生 KEY_RELEASE 事件。用 keyDown 和 keyUp 方法处理键盘事件，两种方法的首部如下。

- public boolean keyDown(Event event，int KeyPressed)；
- public boolean keyUp(Event event，int KeyPressed)。

形参 event 表示了事件本身对象，该对象的 id 成员保存了事件发生时对象的当前值，它与 Event 对象的一组成员变量相对应。例如通过 event.id 的值是否为成员变量 Event.KEY_ACTION 可知道是否按了功能键。绝大多数键盘操作都有与之对应的 Event 对象成员变量。Event 的成员变量很多，请参考其他相关书籍。

形参 KeyPressed 对应于所按键的值。对于常规键，其值为 ASCII 码，其他键也都有确定的值，如 ESC 键的键值是 27。

[例 6-8]　设计一个 Java Applet 程序，当在 Applet 窗口内单击鼠标时显示鼠标指针的(*x*，*y*)坐标值。当按键时显示所按键的键值和对应的字符。

Java Applet 代码如下，文档名为 MouseAndKeyApplet.java。

```
import java.awt.Event;
import java.awt.Graphics;
import java.applet.*;
public class MouseAndKeyApplet extends Applet
{
  String mouseDownInf=null;     //保存按下鼠标时的显示信息
  String mouseDragInf=null;     //保存拖动鼠标时的显示信息
  String mouseEnterInf=null;    //保存鼠标指向或离开 Applet 对象时的显示信息
  String keyDownInf=null;       //保存键盘操作信息
  public boolean keyDown(Event evt, int key)
  { //按下某个键时发生的事件
    keyDownInf="按键:"+(char)key+"  键值:"+key;
    repaint();           //重画对象
    return true;     //返回 true,表示事件处理成功
```

```
  }
  public boolean mouseDown(Event evt, int x,int y)
  { //单击鼠标左键或右键时发生的事件
    mouseDownInf="单击位置: ("+x+", "+y+")";
    repaint();
    return true;
  }
  public boolean mouseDrag(Event evt, int x,int y)
  { //按住鼠标按键拖动时发生的事件
    mouseDragInf="拖动位置: ("+x+", "+y+")";
    repaint();
    return true;
  }
  public boolean mouseEnter(Event evt, int x,int y)
  { //鼠标指向 Applet 对象时发生的事件
    mouseEnterInf="Hi,Welcome!";
    repaint();
    return true;
  }
  public boolean mouseExit(Event evt, int x,int y)
  { // 鼠标离开 Applet 对象时发生的事件
    mouseEnterInf="Bye";
    repaint();
    return true;
  }
  public void paint(Graphics g)
  { // 显示 Applet 对象的方法
    if (mouseDownInf!=null)
       g.drawString(mouseDownInf,25,90);
    if (mouseDragInf!=null)
       g.drawString(mouseDragInf,150,90);
    if (keyDownInf!=null)
       g.drawString(keyDownInf,280,90);
    if(mouseEnterInf!=null)
       g.drawString(mouseEnterInf,200,45);
  }
}
```

调用 Java Applet **的** HTML **文档名为** myApplet2.htm**，内容如下：**

```
<html>
<head>
</head>
<body>
<p><font color="#FF0000" size="5">JavaApplet 中的鼠标和键盘事件演示</font>
</p>
<hr>
<applet code = "MouseAndKeyApplet.class" width=500 height=100>
</applet>
</body>
</html>
```

在浏览器中打开 myApplet2.htm 文档，显示结果如图 6-4 所示。

图 6-4　Java Applet 在浏览器中的显示

最后需要强调的是：Java Applet 并不是一个应用程序，也没有一个包含 main()方法的类，它只是一个由已在运行的 Java 应用程序(如 Web 浏览器或 Applet 查看器)装入并运行的 Java 类。

6.2.12　Java 的多线程机制

在操作系统中，操作系统引入多进程/线程的机制，使得操作系统真正具有多任务的功能。“进程(process)”定义为“可并发执行的程序在一个数据集合上的运行过程”。即进程是内存中执行中的程序，但是进程和程序是两个截然不同的概念。进程是操作系统进行资源分配的单位，进程是动态的，程序是静态的。有人把进程解释为程序的一次运行，这种解释是不确切的。实际上，程序运行并不创建一个新的进程，并且程序一次运行可以创建多个进程。

线程(thread)是一种特殊的进程，它是进程下能够独立运行的更小的单位，线程运行在进程空间内。多线程程序设计是指在单个程序中包含并发执行的多个线程。当多线程程序执行时，该程序对应的进程中同时有多个控制流并发运行，不同的线程共享进程的资源。这在网络编程中非常有用，如在网上聊天程序中，可以通过增加线程的方式处理多人同时运行聊天程序的问题。又如在网络数据传输的同时并发地显示页面或运行 Java 小程序等。

1．Java 中多线程的实现

在 Java 中，实现多线程有两种方式：

第一种方法是从 Thread 类派生一个线程类，覆盖它的 run()方法，代码形式如下：

```
public class myThread extends Thread
{
```

```
    public void run() {
    // here is where you do something
    }
}
```

第二种方法是通过实现 Runable **接口，创建一个可执行类，实现它的** run()**方法，代码形式如下：**

```
public class  myRunable  implements Runnable
{
    public void run() {
    // here is where you do something
    }
}
```

两种方法的区别是，如果用户的类已经继承了其他的类，只能选择实现 Runnable **接口，因为** Java **只允许单继承。**

Java **中的线程有运行、就绪、挂起、结束四种状态。一个线程结束了，就说明它已是一个死线程。当调用一个线程实例的** start()**方法时，线程进入就绪状态，注意，并不是运行状态。当虚拟机开始给线程分配** CPU **运行时间片时，线程开始进入运行状态，当线程进入等待状态，例如等待某个事件发生的时候，此时线程处于挂起状态。**

2．线程的创建、启动和终止

启动一个线程只需要调用 start()**方法，针对两种实现线程的方法也有两种启动线程的方法，分别如下：**

(1**) 继承** Thread **类方式**

```
myThread  aThread = new myThread ();    // 创建一个线程对象
aThread.start();                        // 启动线程
```

(2**) 实现** Runnable **接口方式，是通过实现** Runable **接口创建一个可执行类，并利用一个** Thread **对象来启动线程**

```
myRunnable aRunable = new myRunnable ();// 创建一个可执行对象
Thread aThread = new Thread(aRunable);  // 创建一个线程对象,并与可执行对象关联
aThread.start();                        // 启动线程
```

通过线程对象的 start()**方法可以启动一个线程，**start()**方法为这个线程分配资源，并且调用** run()**方法，进入可运行状态。**run()**是线程类一个很重要的方法，需要该线程执行的程序，就放在该方法中。**

当线程的 run()**方法正常退出时，线程将自动撤消。除此之外，通过** stop()**方法可以终止一个线程的运行。**

[**例** 6-9] **编写一个** Java **程序，创建两个线程，分别在屏幕上输出字母“** a **”和“** b **”。**

根据上面的介绍，采用派生 Thread **类的方法创建进程，文档名** HelloThread.java**，代码如下：**

```
import java.io.*;
```

```
import java.lang.*;
public class HelloThread
{
   public static class MyThreadClass1 extends Thread
   {
     long num1=0;
     public void run()
     {
       while(num1<10000)
       {
         System.out.print("a");
         num1++;
       }
     }
   }
   public static class MyThreadClass2 extends Thread
   {
     long num2=0;
     public void run(){
       while(num2<10000)
       {
         System.out.print("b");
         num2++;
       }
     }
   }
   public static void main(String args[])
   {
      Thread MyThread1=new MyThreadClass1();
      Thread MyThread2=new MyThreadClass2();
      MyThread1.start();
      MyThread2.start();
    }
}
```

使用 javac HelloThread.java **命令编译该文件为**.class **文件。然后使用** java HelloThread **命令运行该程序。运行时将看到在输出窗口中不停地显示两个字母。**

6.3　Servlet 与三层体系结构

Java **应用程序可以在服务器上运行，但是不管是在客户机/服务器环境下，还是在基于** Web **的环境下，**JDK **中都没有提供让** Java **应用程序专用于服务器机器的接口或包。认识到** Java **在** Web **环境下作为一种服务器语言的潜力，**Sun Microsystems **编写了** Java Servlet **规范。**Servlet **在许多方面与** Applet **相似，它是为在** Web **服务器上运行而专门设计的** Java **程序。**

Servlet **可以动态地扩展服务器的能力，并采用请求-响应模式提供** Web **服务。**Servlet

可以在支持 Java 的任何 Web 服务器上运行。对于本身并不支持 Java 可执行程序的 Web 服务器，可以安装 Servlet 运行时环境（通常称为 Servlet 引擎)。

6.3.1 Servlet 与 CGI

在一台 Web 服务器控制下，在多台服务器上运行若干小型用户程序，这就是公共网关接口(CGI)程序(又称 CGI 脚本)所起的作用。CGI 应用程序本身往往不是完整的应用程序，在处理接收自 Web 浏览器上用户的信息请求时，CGI 只是整个处理过程中的一个中间步骤。例如，CGI 应用程序的一种常见用途是访问数据库。将它用于这种任务时，CGI 程序提供一种方法，将用户的数据请求连接到能满足这种请求的企业数据库。CGI 程序常常充当一种中间软件，从 Web 浏览器接收请求，决定必须调用哪些计算资源来满足这些请求，并向浏览器发回响应。传统的 CGI 程序一般是由 C 或 C++开发的，这些语言的执行速度较快。随着优化编译器的的引入，用 Java 来实现中间层成为可能。

Java Servlet 与 CGI 程序一样，最适合充当连接前端 Web 请求与后端数据资源的中间层组件。使用 Java Servlet 可以以更高的效率和可移植性来实现 CGI 的目的，并最终会取代传统的 CGI 程序。

6.3.2 三层体系结构

客户机/服务器(C/S)计算模式是一种典型的两层模型，两层模型在当时曾经具有创新意义，因为它将一些计算任务从主处理器上卸载到灵巧的客户机。常规的基于 LAN 的数据库应用程序就是一个例子，其中数据库管理器服务器软件驻留在一个专用的服务器机器上，而用户则通过他们的工作站上的客户机代码(用户程序)来访问数据库。

传统的客户机/服务器两层体系结构设有好的可伸缩性，因为用户连接和数据访问的数量无法预测，而且在一些系统管理上也存在问题。为处理两层体系结构的限制，许多开发集体都在转向三层体系结构(3-tier)。这种体系结构大致可以定义为：客户机层上的表示层、中间的服务器和后端的某种数据库。这种设想的目的是缓和客户机或数据库服务器上的代码膨胀，集中管理业务逻辑，更灵活地使用数据库，而不仅是使用所存储的过程和触发器。

一个三层结构模型通常被想像成有一个 Web 浏览器作为客户层。Web 浏览器由于有可能成为一种真正的通用客户机，使它从观念上取代了两层结构的“胖客户机”。如果浏览器作为 Web 应用程序体系结构的标准瘦客户机获得认可，那么以前驻留在两层模型的胖客户机中的功能会怎么样呢？现在，应用程序专用的功能并不移植回服务器(例如数据库服务器)，而是有意将它驻留在一个新的中间层上。

新的中间层往往是一组 CGI 程序，将来自 Web 浏览器的用户请求传送到不基于 Web 的业务系统(数据库管理系统)，并向浏览器返回响应。由于三层体系结构通常是基于 Web 的，所以中间层应用程序通常工作在 Web 服务器上，被视为 Web 服务器的一种功能扩展。

根据上述介绍，可以把 Web 应用的三层结构分成用户层(浏览器)、Web 存取层和数据层，结构如图 6-5 所示。

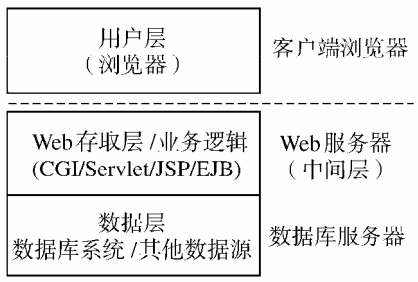


图 6-5　三层结构的分层功能界定

6.3.3　Servlet 编程

中间层的出现，使得 Web 编程向服务器端发展。由传统的 CGI 编程向 Servlet 技术的转移说明三层模型正在增强。

Java Servlet 的出现，为应用程序员使用 Java 创建 Web 应用程序开辟了新途径。但是，Servlet 只是工作在 Web 服务器上的一个连接客户请求和数据库系统的中间层，仅有 Servlet 还不能为真正的企业计算提供完整的模型。

1．编写 Servlet 所需要的开发环境

进行 Servlet 开发所需要的基本环境是 JSDK(Java Servlet Development Kit)以及一个支持 Servlet 的 Web 服务器。

JSDK 包含了编译 Servlet 应用程序所需要的 Java 类库以及相关的文档。对于利用 Java 1.1 进行开发的用户，必须安装 JSDK。JSDK 已经被集成进 Java 1.2 Beta 版中，如果利用 Java 1.2 或以上版本进行开发，则不必安装 JSDK。

除了开发工具之外，还要安装一个支持 Java Servlet 的 Web 服务器，或者在现有的 Web 服务器上安装 Servlet 软件包。如果使用的是最新的 Web 服务器或应用服务器，很可能它已经有了所有必需的软件。通过查看 Web 服务器的文档，或访问 http://java.sun.com/products/servlet/industry.html 查看支持 Servlet 的服务器软件清单。

作为免费软件，Tomcat 是 Servlet 2.2 和 JSP 1.1 规范的官方参考实现。Tomcat 既可以单独作为小型 Servlet、JSP 测试服务器，也可以集成到 Apache Web 服务器中。

2．Servlet 的开发过程

Servlet 不是一个真正意义上的完整的独立的 Java 程序，因为它没有 main()函数。它只是运行在 Web 服务器上的一个负责用户和数据层之间转换的接口，是用 Java 编写的类。并且所有的 Servlet 都必须继承基本的 Servlet 类，定义请求处理的方法。

Servlet 编程要求必须掌握 Java 程序设计语言，还要对面向对象有所了解。Java 语言程序设计主要是对 Java 类库的使用，同样，掌握 Servlet 编程需要熟练使用 Sun 公司的 JSDK。

(1) 编写 Servlet 代码

Java Servlet API 是一个标准的 Java 扩展程序包，包含两个 package：javax.servlet 和

javax.servlet.http。对于想开发基于客户自定义协议的开发者，应该使用 javax.servlet 包中的类与界面；对于仅利用 HTTP 协议与客户端进行交互的开发者，则只需要使用 javax.servlet.http 包中的类与界面进行开发即可。

下面的代码显示了一个简单 Servlet 的基本结构。

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class SomeServlet extends HttpServlet {
        public void doGet(HttpServletRequest request, HttpServletResponse
        response)
        throws ServletException, IOException
        {
           // 使用“request”读取和请求有关的信息(比如 Cookies)和表单数据
           // 使用“response”指定 HTTP 应答状态代码和应答头
           // (比如指定内容类型,设置 Cookie)
           // 使用 "out"把应答内容发送到浏览器
           PrintWriter out = response.getWriter();
     }
}
```

如果某个类要成为 Servlet，则它应该从 HttpServlet 继承，根据数据是通过 GET 还是 POST 发送，覆盖 doGet、doPost 方法之一或全部。doGet 和 doPost 方法都有两个参数，分别为 HttpServletRequest 类型和 HttpServletResponse 类型。HttpServletRequest 提供访问有关请求的信息的方法，例如表单数据、HTTP 请求头等。HttpServletResponse 除了提供用于指定 HTTP 应答状态、应答头(Content-Type，Set-Cookie 等)的方法之外，最重要的是它提供了一个用于向客户端发送数据的 PrintWriter 。对于简单的 Servlet 来说，它的大部分工作是通过 println 语句生成向客户端发送的页面。

注意：doGet 和 doPost 抛出两个异常，必须在声明中包含它们。另外，还必须导入 java.io 包(要用到 PrintWriter 等类)、javax.servlet 包(要用到 HttpServlet 等类)以及 javax.servlet.http 包(要用到 HttpServletRequest 类和 HttpServletResponse 类)。

最后，doGet 和 doPost 这两个方法是由 service 方法调用的，有时可能需要直接覆盖 service 方法，比如 Servlet 要处理 GET 和 POST 两种请求时。

[例 6-10] 一个简单的 Servlet 代码示例(HelloServlet.java)。

```
package test;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class HelloServlet extends HttpServlet
{
  public  void  doGet(HttpServletRequest  request,HttpServletResponse
  response)throws ServletException,IOException
  {
    response.setContentType("text/html");
```

```
        PrintWriter out = response.getWriter();
        out.println("<html><head><title>");
        out.println("This is my first Servlet");
        out.println("</title></head><body>");
        out.println("<center><h1>Hello,Servlet!</h1></center>");
        out.println("</body></html>");
    }
}
```

该 servlet 实现如下功能：当用户通过浏览器访问该 servlet 时，该 servlet 向客户端浏览器返回一个 HTML 页面，显示 Hello Servlet!。

(2) 编译 Servlet 代码

利用 JDK 1.4.2 对例 6-10 中的 Servlet 代码进行编译，其命令行为：

```
D:\MyJSP>javac HelloServlet.java
```

执行上述编译命令，提示 javax.servlet 包不存在。表明在 classpath 路径中缺少一个解析 servlet 类的包。在 Tomcat 5.0 中，需要将 tomcat5.0\common\lib 目录下的 servlet-api.jar 添加到 classpath 中。具体方法是：将 servlet-api.jar 文件复制到 C:\j2sdk1.4.2_03\jre\lib\ext 目录下，再次编译，在 D:\MyJSP 下生成 MyServlet.class 文件，编译成功。

(3) 运行 Servlet

现在可以对 MyServlet 进行测试了，打开浏览器，按照通常的思路，在地址栏内键入：

```
http://127.0.0.1/HelloServlet
```

显示如下错误信息：

```
HTTP Status 404 - /HelloServlet/
type Status report
message /MyServlet/
description The requested resource (/HelloServlet/) is not available.
Apache Tomcat/5.0.18
```

上述信息表明找不到 Servlet，为什么呢？

在 Tomcat 中，运行 Servlet 比较复杂，需要再对各处做手工配置，主要的配置包括以下三个方面。

① 修改%TOMCAT_HOME%\conf\Web.xml 文件。找到如下代码段：

```
<!--
<servlet>
       <servlet-name>invoker</servlet-name>
       <servlet-class>
         org.apache.catalina.servlets.InvokerServlet
       </servlet-class>
       <init-param>
          <param-name>debug</param-name>
          <param-value>0</param-value>
       </init-param>
       <load-on-startup>2</load-on-startup>
```

```
</servlet>
-->
```

然后再找到下述代码段：

```
<!-- The mapping for the invoker servlet -->
<!--
    <servlet-mapping>
        <servlet-name>invoker</servlet-name>
        <url-pattern>/servlet/*</url-pattern>
    </servlet-mapping>
-->
```

上述代码显示 Tomcat **调用器(**InvokerServlet**)被关闭了。在** Tomcat4.1.12 **版本之前，默认情况下调用器是启用的，但由于存在一个安全缺陷，因此在此后的版本中，调用器默认情况下是禁用的。如果要启用的话，只要将**%TOMCAT_HOME%\conf\Web.xml **文件中上述两段代码的注释取消即可。**

② 新建用户 Web **应用项目。为了解释问题的方便，可以新建一个用户** Web **应用项目，主目录为** D:\MyServlet**。并将上述文档** HelloServlet.java **存储(复制)到** D:\MyServlet **中。然后在** D:\MyServlet **中，创建** WEB-INF **文件夹，包含** class **文件夹，存储用户** Web **项目中的** Java **类。**class **文件夹中再创建** test **文件夹，最后将编译后的** HelloServlet.class **复制到** D:\MyServlet\WEB-INF\classes\test **文件夹下。**

在 D:\MyServlet\WEB-INF **目录中，新建一个** web.xml **文档，内容如下：**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application
2.3//EN" "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>
  <display-name>HelloServlet</display-name>
    <servlet-mapping>
        <servlet-name>invoker</servlet-name>
        <url-pattern>/servlet/*</url-pattern>
    </servlet-mapping>
</web-app>
```

上述代码的意思是，碰到没有特别指定的 Servlet**，都用**/servlet/*去映射它。**一般情况下，为了管理上的方便，把所有的** Servlet **放到** Web **应用的主目录下的一个** Servlet **文件夹中。**

用户也可以对每一个 Servlet **单独作映射，形式如下：**

```
<servlet>
   <servlet-name>HelloServlet</servlet-name>
 <servlet-class>test/HelloServlet</servlet-class> <!-- class 文件与项目的相
 对路径-->   </servlet>
 <servlet-mapping>
   <servlet-name>HelloServlet</servlet-name>
   <url-pattern>/servlet/HelloServlet</url-pattern> <!--调用时的方法-->
</servlet-mapping>
```

③ 最后需要配置服务路径。到%Tomcat5%\conf\Catalina\localhost **下，新建一个文件**

myservlet.xml。内容如下：

```
<Context path="/myservlet" docBase="D:\MyServlet\" debug="0" reload
able="true"  privileged="true"/>
```

其中，path 是访问路径，表示在浏览器地址栏里输入 IP 地址后的 Web 应用路径。DocBase 是 Web 应用的物理路径。reloadable 为 true，表明修改 Servlet 文件、JSP 文件后，不用重启 Tomcat 即可生效。

当上述配置完成后，打开浏览器，在地址档中输入 http://127.0.0.1:8080/myservlet/servlet/test.HelloServlet，如图 6-6 所示。

图 6-6　Servlet 运行界面

从浏览器中输入的地址可以看到，在 myservlet 和 test.HelloServlet 之间有一个 servlet，而 Web 应用中的目录结构中并没有这个目录。这是因为，在文档 D:\MyServlet\WEB-INF\web.xml 中包含<url-pattern>/servlet/*</url-pattern>，如果没有 Servlet 文件夹，可以写为<url-pattern>/* </url-pattern>，意即所有的 Servlet 都存放在用户 Web 应用的根目录(/*)下。这样，在地址栏中可以输入 http://127.0.0.1:8080/myservlet/test.HelloServlet 就可以运行指定的 Servlet 了。

或者，写一个调用该 Servlet 的 HTML 文件，通过 HTML 的 form 表单中的 action 调用 Servlet，运行相应的 Servlet。例如：

```
<html>
<head>
  <title>Java Servlets Sample</title>
</head>
<body>
  <form method="get" action="test.HelloServlet">
  <input name="test" type="submit" value="Test HelloServlet">
</body>
</html>
```

6.4 JavaBeans 组件

JavaBeans 组件是一种开发软件组件的技术，软件组件技术可以把软件功能作为一个组件来组装成一个应用程序。在 Java 中，bean 是一些 Java 类，可在一个可视的构建器(builder)工具中操作它们，并且可以将它们一起编写到应用程序中。任何具有某种特性和事件接口约定的 Java 类都可以是一个 bean。通过将一个满足某些规则的 Java 类变为 Java bean，可以有效地实施软件重用。

由于 Servlet 的复杂性，目前 JavaBeans 和 JSP 技术已经成为 Web 应用服务器(三层结构中的中间层业务逻辑)开发的主要工具。

6.4.1 JavaBean 的属性、方法和事件

一个 JavaBean 和一个 Java Applet 类似，是一个遵循某些规则的 Java 类，每个 JavaBean 的功能可能不同，但一般都具有以下特征：

(1) 支持自检，这样构造器(builder)可以分析 bean 是如何动作的。

(2) 支持定制，用户可以通过应用程序构造器工具定制 bean 的外观和行为。

(3) 支持事件，保证 bean 和外部的通信。

(4) 支持属性，使得 bean 真正的具有内部状态，从而根据应用需要定制。

(5) 支持持久性，这样 bean 才能在应用程序构造器中的工具中定制，并将定制的状态存储起来以便以后使用。

1．方法

JavaBean 归根到底是一个 Java 类，所有的 public 方法都可以通过对象外部直接调用。

2．属性

JavaBean 提供了高层的属性(properities)概念。从面向对象的角度看，属性就是传统的对象属性(attribute)。JavaBean 属性描述了 bean 的内部状态，是 JavaBean 中的数据部分，属性的值可以通过适当的 bean 方法进行读写。

JavaBean 有四种类型的属性，分别是单值属性、索引属性、关联属性和限制属性。

(1) 单值属性

单值属性是 JavaBean 中最简单的属性，只需要定义一个包含一个值的数据成员，并为其定义一对设置(set)/获取(get)属性的方法，以便于外部与其联系。如果没有为单值属性提供设置器方法，则该属性为只读型属性；如果没有为单值属性提供获取器方法，则该属性为只写型属性。

单值属性设置器/获取器定义的一般形式为：

```
public void set<PropertyName>(<PropertyType> propertyValue)    // 设置器
public <PropertyType> get<PropertyName>()                      // 拾取器
```

(2) 索引属性

索引属性类似于 Java 中的数组，包括若干个数据类型相同的元素，可以通过整数索引值访问其中的属性，因此称索引属性。

(3) 关联属性

JavaBean API 除了支持单值属性和索引属性外，还提供了一些属性用于增强 JavaBean 的属性管理功能。如关联属性，当修改这类属性时，将发送一个通知给其他元素(如 Applet、application 或其他 JavaBean)，如果与该 JavaBean 中的某个属性相关联，就会注册该属性。因此，只要这个关联属性发生变化，就会有一个通知发送给这些相关部件。这些属性称为关联属性，它们的值发生改变与外部部件有关，外部部件称为监听器。

一个有关联属性的 JavaBean 需要支持如下一对事件监听器的注册方法：

```
public void addPropertyChangeListner(PropertyChangeListner1)
public void removePropertyChangeListner(PropertyChangeListner1)
```

(4) 限制属性

JavaBean API 中的另一种高级属性类型是限制(constrained)属性，它可以使外部部件在接受属性的修改值之前先确认修改值。即需要修改一个限制属性值时，接受属性的外部部件首先要检查这个属性的合理性再决定是否接受修改。

3．事件

JavaBean 和其他软件组件交流信息的主要手段是发送和接受事件。在新的 AWT 事件模型中，一个事件源可以注册事件监听器对象，当事件源检测到某种事件发生时，将激活检测器对象中一个相应的事件处理方法。

JavaBean 必须能够发送事件变化通知和监听到其他 JavaBean 的事件变化，并对监听到的事件变化进行相应的业务处理。事件的监听原理是：首先事件源必须对需要发送的事件进行注册，然后注册事件监听器，并说明该事件源所发生的事件向什么组件发送，即在事件源组件中实现方法并在监听组件中注册该事件源。

[例 6-11]　JavaBean 应用举例。

首先在 D 盘上创建一个目录 D:\MyJavaBean。然后到%Tomcat5%\conf\Catalina\localhost 下，新建一个文件 myjavabeans.xml。输入如下内容：

```
<Context path="/myjavabeans" docBase="D:\MyJavaBean\" debug="0"reloadable=
"true"  privileged="true"/>
```

这样，就可以在浏览器的地址栏中输入 http://127.0.0.1:8080/myjavabeans/文件名.jsp 来访问一个 JSP 文档了。

一个 NameCard 的 JavaBean 代码如下(文件名 NameCard.java)：

```
package cards;
public class NameCard
{
   String Name,Address;
   public NameCard()
   {
```

```
    this.Name="John";
    this.Address="No,Road,City,Country";
  }
  public void setName(String myName)
  {
    this.Name = myName;
  }
  public String getName()
  {
    return (this.Name);
  }
  public void setAddress (String myAddress)
  {
     this.Address = myAddress;
  }
  public String getAddress ()
  {
     return (this.Address);
  }
}
```

将文件保存在 D:\MyJavaBean **中，用** javac NameCard.java **编译生成** NameCard.class**。**

在 bean **中临时增加** main**()方法是为了测试程序用的，写** JavaBean **可以先不必加入到** JSP **程序中调用，而直接用** main**()方法来调试** bean**，调试好以后就可以在** JSP **程序中调用了。**

6.4.2 在 JSP 中使用 JavaBean

如果在 JSP **网页中使用** JavaBean**，可以通过**<jsp:useBean>**标记完成，一般形式是：**<jsp:useBean id="Id " class="yourClass " scope="page | request | session | application" />

其中，Id **对应于一个类的实例，如果这个实例已经存在，将直接引用这个实例。如果这个实例尚未存在，将通过在** class **中的定义从这个** class **创建实例。可以通过** yourId **实例来处理并调用** JavaBeans **中的属性和方法。**Scope **用于定义** Id **这个实例存在的范围，事实上定义了这个实例所绑定的区域及其有效范围。**

[**例** 6-12] **在** JSP **文档中调用** JavaBean**。**

下面编写一个调用上述 bean **的** JSP **文档，文件名为** myCard.jsp**，内容如下：**

```
<html>
<body>
<%@ page language="java" %>
<jsp:useBean id="mycard" scope="application" class="cards.NameCard"/>
<%
  mycard.setName("Hao YY");
  mycard.setAddress("Shanda nan lu 27,Jinan,China");
%>
姓名 : <%=mycard.getName()%><br>
地址 : <%=mycard.getAddress()%><br>
</body>
```

```
</html>
```

将上述 JSP 文档保存到 D:\MyJavaBean 文件夹下。

同时，在 D:\MyJavaBean 文件夹中，创建一个 WEB-INF 文件夹，将%tomcat5.0%\webapps\ROOT\WEB-INF 中的文档 web.xml 复制到 D:\MyJavaBean\WEB-INF 中。然后在 D:\MyJavaBean\WEB-INF 中创建 classes 文件夹，在 classes 文件夹下创建 cards 文件夹(对应包含 JavaBean 的包)。最后将上面的 NameCard.class 文件复制到用户 Web 应用的 D:\MyJavaBean\WEB-INF\classes\cards 文件夹下。

当上述操作结束后，就可以在 JSP 中使用 JavaBean 了。在浏览器地址栏中输入：

```
http://127.0.0.1:8080/myjavabeans/myCard.jsp
```

浏览器将显示如下内容。

```
姓名 : Hao YY
地址 : Shanda nan lu 27,Jinan,China
```

6.4.3　Enterprise JavaBeans

1997 年 4 月 12 日，Sun 公司宣布了一项为企业环境开发 Java 平台的创新成果——Java 2 Platform, Enterprise Edition (J2EE)。使用开放式的 Java Community Process，Sun 公司促进了一组标准的 Java 扩展的开发，称为 Enterprise Java API。这些应用程序编程接口(API) 为各种各样的中间件的实现提供了不依赖供应商的编程接口。Enterprise Java API 的要点是 Enterprise JavaBeans API，后者为 Java 应用程序服务器定义了一个服务器端组件模型，以及一个不依赖供应商的编程接口。

J2EE 为 Enterprise JavaBeans 技术提供了工作环境。事实上，Sun 公司把若干项软件技术都设想为这样的构件块，它们将使大型企业能够把以任务为关键的业务系统移植到 Java 环境中，而 Enterprise JavaBeans 技术不过是这些技术之一。EJB 组件是按它们自己的规范定义的，但 EJB 技术并不是一项独立的技术。它建立在其他 Java 技术之上，这些技术由 Sun 公司和其他 IT 公司联合规定，它们一起提供了这个框架的内容，该框架就称为 Java 2 Platform, Enterprise Edition。

J2EE 中包括以下技术：

- Enterprise JavaBeans (EJB)
- Java Interface Definition Language (IDL)
- Java Message Service (JMS) API
- Java Naming and Directory Interface (JNDI)
- Java Remote Method Invocation (RMI) 和 Object Serialization
- Java Servlet API
- Java Transaction API (JTA)
- Java Transaction Service (JTS)
- Java Server Pages (JSP)
- JDBC 数据库访问 API

Enterprise JavaBeans 技术把 Java 组件的概念从客户机域扩展到了服务器域。这是 Java 技术成长过程中有重大意义的一步，它使 Java 技术发展成为一种强健的、可伸缩的环境，能够支持以任务为关键的企业信息系统。

对 EJB 的介绍已经超出本书的范围，请读者参考其他的相关书籍。

6.5 JSP 技术

使用 Servlet 开发服务器端中间层逻辑相当复杂。为此，Sun 公司于 1999 年推出 JSP 技术。JSP(Java Server Pages)是一种 Java 平台技术，主要用于建立带有动态 Web 内容(如 HTML、DHTML 和 XML)的应用程序。JSP 文档(*.jsp)是在传统的 HTML 文档(*.htm, *.html)中加入 Java 脚本程序(Scriptlet)和 JSP 标记构成的，相对于 Servlet，它很好地实现了业务逻辑和 HTML 内容的分离编写。

JSP 是一种服务器脚本语言，它包装了 Java Servlet 系统界面，简化了 Java 和 Servlet 的使用难度。不仅 JSP 页面上可以直接书写 Java 代码，而且是编译成 Servlet 后才能实际运行。当用户浏览.jsp 网页时，首先在服务器端执行.jsp 文档中服务器端代码，然后把执行结果传送到客户端浏览器，基本上与浏览器无关。

6.5.1 JSP 的运行和开发环境

相对于 ASP，JSP 的运行和开发环境比较复杂。在服务器端，不仅需要启动 Tomcat，还需要安装 Java 环境，另外，根据 Web 服务器管理的需要还要安装和配置 Apache Web 服务器。

1．运行与开发环境

JSP 的运行和开发环境框架模型如图 6-7 所示。

图 6-7 JSP 的运行和开发环境

JSP 的运行开发环境很多，可以采用 Unix、Linux、Windows NT/2000 Server 等操作系统，如果希望在服务器上开发 Servlet/EJB/JSP 应用程序，应该还要安装以下软件：

(1) 安装 Java 环境。

(2) 安装 Java VM(JRE)，因为 Tomcat 5 需要 Java VM 的支持。

(3) 安装 Tomcat。

Tomcat 是针对 Apache 服务器开发的 JSP 应用服务器，是 Java Servlet 和 Java Server Pages 技术的标准实现，是基于 Apache 许可证下开发的自由软件。可以从网站 http://jakarta.apache.org/tomcat/index.html 下载不同的 Apache Tomcat 版本。

可以这样认为，当在一台机器上配置好 Apache 服务器，可利用它响应对 HTML 页面的访问请求。实际上 Tomcat 部分是 Apache 服务器的扩展，但它是独立运行的，所以当运行 Tomcat 时，它实际上作为一个与 Apache 独立的进程单独运行的。当配置正确时，Apache 为.html 页面服务，而 Tomcat 实际上运行.jsp 页面和 servlet。

要在 Apache 下运行 JSP，最好的方案就是选择 Tomcat，具有免费、集成度好的优点，缺点是界面不够直观，如果是在 Windows NT/2000 Server 平台上，需要设置环境变量。

目前，Tomcat 的最新版本是 5.0.18，它的运行需要 Java Virtual Machine(VM)的支持。

(4) Apache 服务器

可以根据需要安装 Apache 服务器。由于本章主要介绍 JSP 的开发，因此，只要安装了 Tomcat 即可。

2．建立一个 JSP 程序的测试站点

为了测试后面要学习的 JSP 程序，首先需要建立一个 JSP 程序测试站点。步骤如下：

(1) 建立测试站点的主目录

假设测试站点的主目录是 D:\MyJSP，然后所有的 JSP 程序就可以存储到该目录下，或者建立虚拟目录。

在主目录下，创建一个 index.jsp 文件。

(2) 修改 Tomcat 的默认设置

在默认情况下，通过 http://127.0.0.1:8080/可以访问 Tomcat 默认的 Web 应用。要想通过 http://127.0.0.1 来访问这个新的 Web 应用，需要修改 Tomcat 的一些设置。要使 Tomcat 指向 D:\MyJSP，需要修改 C:\Program Files\Apache Software Foundation\Tomcat 5.0\conf 下的文件 server.xml。具体修改包括：

① 将 Tomcat 默认的 Web 服务端口号 8080 改为 80。

② 增加新 Web 应用的上下文。

<Context path="" docBase=" D:\MyJSP " debug="0">告诉 Tomcat 如何找到用户的 Web 应用。

除此之外，还可以按照 6.4 节的介绍，和 D:\MyServlet 项目及 D:\MyJavaBean 项目一样，在%Tomcat5%\conf\Catalina\localhost 下，新建一个文件 myjsp.xml。输入如下内容：

```
<Context path="/myjsp" docBase="D:\MyJSP\" debug="0" reloadable="true"
privileged="true"/>
```

这样，就可以在浏览器的地址栏中输入 http://127.0.0.1:8080/myjsp/文件名.jsp 访问一个 JSP 文档了。

(3) 启动 Tomcat

当上述修改完毕后，在任务栏中右击 Tomcat 图标，选择 Shutdown:Tomcat5 菜单项，

关闭 Tomcat。然后在“开始”菜单中重新启动 Tomcat，尝试运行用户 Web 应用。

注意：如果安装了 Apache，在计算机启动时它将被启动，此时需要首先将 Apache Http Server 关闭，然后在运行 Tomcat。

此时，启动浏览器，在地址栏中输入 http://127.0.0.1，就可以看到 D:\MyJSP 中的 index.jsp 文件了。如果显示中文乱码，在浏览器中，选择“查看”|“编码”|“中文简体”菜单项即可。

(4) 运行 JSP 程序

如果要运行 D:\MyJSP 下面的 JSP 文档 1.jsp，在地址栏中输入 http://127.0.0.1/1.jsp。注意，必须要输入扩展名，同时要注意文件名和目录的大小写。

如果建立了虚拟目录，要运行虚拟目录下的文件，输入 http://127.0.0.1/虚拟目录/文件名.jsp。

6.5.2 JSP 的语法结构

JSP 文档是通过在 HTML 文档中加入 Java 脚本程序构成的。Java 脚本程序代码用“<%”和“%>”标记括起来，称为 JSP 元素。JSP 元素可以分为三种类型：脚本元素、指令元素和动作元素。脚本元素规范 JSP 中所使用的 Java 代码；指令元素针对 JSP 引擎控制转译后的 Servlet 的整个结构；动作元素主要连接到用到的组件(如 Javabean 和 Plugin)，另外它还可以控制 JSP 引擎的行为。

1．JSP 指令

JSP 指令是 JSP 的引擎。它们不直接产生任何可视的输出，只是指示引擎如何处理 JSP 页面中的内容。JSP 指令由<%@ …%>标记，一般形式是：

```
<%@ 指令名 属性1="属性值" 属性2="属性值" ... 属性n="属性值" %>
```

主要的两种指令是 page 和 include。

(1) page 指令

page 指令用来定义整个页面的相关属性。该指令的语法和它所包含的属性如下：

```
<%@  page
  [ language="java" ]
  [ extends="package.class" ]
  [ import="{package.class | package.*}, ..." ]
  [ session="true | false" ]
  [ buffer="none | 8kb ]
  [ autoFlush="true | false" ]
  [ isThreadSafe="true | false" ]
  [ info="text" ]
  [ errorPage="relativeURL" ]
  [ contentType="mimeType [ ;charset=characterSet ]" |
                "text/html ; charset=ISO-8859-1" ]
  [ isErrorPage="true | false" ]  %>
```

对于指令的几个常用属性，有以下的简要解释。

- language 属性　所使用的脚本语言。例如<%@ page language="Java" %>。
- import 属性　脚本元素中使用的类，与 Java 中的 import 声明作用相同。应是类的全名，或者类所在的包。例如：

```
<%@ page import="java.util.Date" %>
<%@ page import="java.io.*" %> (java.io 包中的所有类在本页中都可以使用)
```

- session 属性　是否使用 session 对象。
- buffer 属性　对象的输出模式。none 为没有缓冲区；8KB 为缓冲区大小。
- autoFlush 属性　缓冲区已满时是否自动清空。当 buffer 为 none 时该属性不能为 false。
- isThreadSafe 属性　处理对象间的存取是否引入 Thread Safe 机制。如果为 true，则在程序中必须有多线程的程序代码，否则直接实现 SingleThreadModel 机制。
- errorPage 属性　设置异常处理网页。
- isErrorPage 属性　当前网页是否是另一个 JSP 网页的异常处理网页。
- contentType 属性　输出到客户端的 MIME 类型和字符集。默认的字符集是 ISO-8859-1。例如　使用中文字符集：<%@ page contentType="gb2312" %>。

可以在几乎所有的 JSP 页面顶部找到指令 page，使用 page 指令(at page)可以定义网页的处理方式，如到何处寻找 Java 类支持等。常用的有：

```
<%@ page import="java.util.Date" %> (当出现 Java 运行问题时将网络用户指引到何处)
<%@ page errorPage="errorPage.jsp" %>(当出现错误时的错误处理网页)
<%@ page session="true" %>
```

(2) include 指令

这是一个在 JSP 网页中包含其他文件的指令。它是在 JSP 引擎将它转译成 Servlet 时产生作用的指令。其一般形式如下：

```
<%@  include file="插入文件的 url" %>
```

这里要求，被包含进的文件符合 JSP 的语法，应是静态文本、指令元素、脚本元素和动作元素。注意，包含后面不能有分号。

2．声明

JSP 的声明用于定义页面级的变量，如果代码太多，最好写入一个单独的 Java 类中。声明由<%! … %>定义。必须通过分号来结束变量声明，同时任何内容必须是有效的 Java 语句，例如<%! int i=0; %>。注意，声明后面的分号不能省略。

3．表达式

在 JSP 文档中，可以将表达式的结果直接输出到页面中。一般形式是<% =表达式%>。例如：

```
<%= i %> (输出变量 i 的值)
<%= "Hello" %>(输出字符串常量)
```

4．代码段/脚本片段

JSP 代码段或脚本片段是嵌在“<% … %>”标记内。当 Web 服务器响应请求时，这种 Java 代码就会运行。在脚本片段周围可能是纯粹的 HTML 或 XML 代码，在这些地方，代码片段可以使编程人员创建条件执行代码，或只是调用另外一段代码。

例如，以下的代码组合使用表达式和脚本片段，分别按照 H1、H2、H3、H4 和 H5 标题样式显示字符串“ 你好 ”，脚本片段并不局限于一行源代码中。

```
<% for (int i=1; i<=4; i++) { %>
<H<%=i%>>你好</H<%=i%>>
<% } %>
```

5．注释

在文档中加入 HTML 注释，用户可以通过查看页面源代码来看到这些注释的内容。如果不想让用户看到注释内容，应将其嵌入到<%-- … --%>标记对中，一般形式是：

```
<%--注释内容--%>
```

[例 6-13] 编写一个 JSP 文档，显示网页的访问次数。

首先编写一个统计访问次数的文档，文档名为 mycount.htm，内容如下：

```
<%! private int accessCount=0; %>
<table border="0" width="100%" height="60" bgcolor="#FFFF00">
  <tr>
    <td width="20%" height="53">主机名:<%=request.getRemoteHost()%></td>
    <td width="20%" height="53">访问次数:<%=++accessCount %></td>
    <td width="60%" height="53">当前时间:<%=new Date() %></td>
  </tr>
</table>
```

编写一个 JSP 文档 mypage.jsp，包含上述文件，输出一个随机数。内容如下：

```
<html>
<head>
<title>JSP 中的文件包含</title>
</head>
<body>
<%@ page import="java.util.*"%>
<%! Random RdmNumber=new Random();%>
<%@ include file="mycount.htm"%>
<%
   out.println(RdmNumber.nextInt(100));  //输出 100 以内的随机整数
%>
</body>
</html>
```

将上述文件保存在 D:/MyJSP 文件夹中，打开浏览器，输入 http://127.0.0.1/mypage.jsp 可以看到网页的输出结果。

6.5.3　JSP 内置对象

内置对象是由开发环境所定义的具有特定功能的对象，用户可以直接使用。在 JSP 脚本段中，可以访问这些隐含对象来与 JSP 网页中的可执行 servlet 环境交互。JSP 中主要包含的内置对象有：

(1) request 对象　作用范围为整个页面。通过该对象可以获取客户端的输入信息。可以得到请求的参数、请求类型(GET、POST、HEAD 等)以及 HTTP　headers(cookies、referer)等信息。这是一个 javax.servlet.HttpServletRequest 对象。

常用的方法有：getCookie()、getHeaders()、getAttribute()、getMethod()、getParameter()和 getParameterName()等。例如，要获取一个网页的 userName 参数的值，代码如下：

```
<% String name=request.getParameter("userName"); out.println(name); %>
```

(2) response　作用范围为整个页面。它的作用是向客户端返回请求。返回请求信息时，输出流要进行缓存。它是一个 javax.servlet.HttpServletResponse 对象。

常用的方法有 addCookie()、addHeader()、sendError()和 sendRedirect()等。

(3) out　发送响应的输出流，作用是将结果输出到客户端。它最常用的方法有两个：print()和 println()。输出换行符使用 newLine()方法。

例如不用表达式，可以直接访问隐含对象 out 来输出信息。

```
<% out.println("Hello"); %>
```

(4) session　作用范围为会话期间。Session 对象是 JSP 为每一个会话而建立的个人对象，可以存储及提供个别用户独享的永久或半永久信息。它是一个与 request 相关的 javax.servlet.http.HttpServletSession 对象。

所谓会话(session)，是指一个用户和 Web 服务器之间的一次链接。当用户使用浏览器登录到 Web 服务器、并初次浏览一个 JSP 应用的某个网页开始、直到用户超时未继续浏览该 JSP 网页为止，之间的浏览操作算作一次会话(Session)。

(5) application　该对象可存储并提供给一组 JSP 应用所有用户的共享信息，有效范围为构成该 JSP 应用的所有 JSP 页面。一般情况下，可以将 application 对象作为一个存储许多共用对象的容器，这是一个 javax.servlet.ServletContext 对象。可通过 getServletConfig()、getContext()方法获得。

(6) config　用于传递在 Servlet 程序初始化时所需的信息。

(7) pageContext　该对象提供了对页面上的所有对象以及命名空间的访问，用于管理网页属性。

(8) page　当前页面，相当于 Java 中的 this。

(9) exception　错误处理对象，用于处理捕捉到的异常。

[例 6-14]　编写一个 JSP 文档，完成一个登录界面，输入用户名和密码，如果输入 guest 则转移到一个默认的首页，如果不输入用户名，则重新回到该页，直到输入正确的用户名和密码，此时重定向到合法的网页。

文档(文档名为 login.jsp)的具体内容如下：

```
<!-- login.jsp -->
<html>
<head>
<title>login</title>
</head>
<body>
<%
String userName = request.getParameter("userName");
if (userName!=null) {
  if (userName.equals("guest"))
     response.sendRedirect("http://www.sdu.edu.cn/");
  else
     response.sendRedirect("http://gsl.sdu.edu.cn/");
}
%>
<div align="center">
 <center>
 <table border="0" cellpadding="0" width="100%" height="100%">
 <tr>
 <td width="100%">
     <table border=2 cellspacing=0 cellpadding=0 width=300 align=center>
     <form action="login.jsp" method="GET">
        <tr height=50>
              <td align=center style="font-size:20;font-weight:bold;color:
              #000075">用户登录</td>
        </tr>
        <tr><td style="font-size:16">用户名<input type=text name=userName>
        </td></tr>
        <tr><td  style="font-size:16"> 密  码 <input  type=text  name=user
        Password></td></tr>
        <tr><td align=center><input type="submit" value=登录></td></tr>
    </form>
    </table>
 </td>
 </tr>
 </table>
 </center>
</div>
</body>
</html>
```

将文件保存到 D:\MyJSP **文件夹中，在浏览器地址栏中输入** http://127.0.0.1/login.jsp**，执行结果如图** 6-8 **所示。**

执行该程序时，如果是第一次执行，服务器首先对它进行转换、编译形成字节码文件，接着运行该文件。第一次执行该文档时，用户名空，不执行开始处的 Java **代码段，此时输入用户名和密码，单击“ 登录 ”按钮提交。由于表单的** action **属性为** login.jsp**，此时再次执行该文档。这样，直到输入一个合法的用户名和密码，然后将网页重定向到其他网页，否则，总是显示该网页。**

图 6-8　一个 JSP 文档实现的登录界面

[例 6-15]　编写一个 JSP 程序，显示在线人数。

首先编写一个统计会话人数的 Java 类，代码内容如下(文档名为 SessionCounter.java)：

```
package SessionCount;
import javax.servlet.*;
import javax.servlet.http.*;
public class SessionCounter implements HttpSessionListener
{
  private static int activeSessions = 0;
  public void sessionCreated(HttpSessionEvent se)
  {
    activeSessions++;
  }
  public void sessionDestroyed(HttpSessionEvent se)
  {
    if (activeSessions > 0)
       activeSessions--;
  }
  public static int getActiveSessions()
  {
    return activeSessions;
  }
}
```

将文档存储在 D:\MyJSP 文件夹下，编译成 SessionCounter.class，并存储到 D:\MyJSP\WEB-INF\classes\SessionCount 中。

然后编写调用该 SessionCounter.java 的 JSP 文档，文档名为 myonline.jsp，内容如下：

```
<html>
<head>
</head>
<body>
<%@ page import="SessionCount.SessionCounter" %>
在线人数:<%= SessionCounter.getActiveSessions() %>
```

```
</body>
</html>
```

最后，修改 D:\MyJSP\WEB-INF\web.xml。**在**<web-app>…</web-app>**内，添加如下内容：**

```
<!-- Listeners -->
<listener>
        <listener-class>
           SessionCount.SessionCounter
        </listener-class>
</listener>
```

6.5.4 JDBC 与数据库

数据库连接对动态网站来说是最为重要的部分。在 Windows **和** Macintosh[1]**平台上各种数据库的通信中，**ODBC**(**Open Database Connectivity**，开放数据库互连)已经成为一种事实上的标准。**ODBC **已经被广大的开发商和编程人员公认为较好的数据库之间的通信手段。这些数据库包括** Oracle、 Sybase SQL Server、Miscrosoft SQL Server、Miscrosoft Access **以及** My SQL **等。**

ODBC **是图形用户界面和数据库后台之间的通信层，只要使用标准的** SQL **语句便可以访问数据库而不需要使用特定的数据库命令。**ODBC **使用户程序可移植性，不需要或者只需要较少的改动就可以将应用程序中的数据库从一种形式移植到另一种，例如从** Miscrosoft Access **移植到** Miscrosoft SQL Server**。**

1．JDBC 简介

Java **中连接数据库的技术是** JDBC**(**Java Database Connectivity**)，它和** ODBC **类似，通过** java.sql **包中的类、接口和异常事件，可以对数据库进行操作。很多数据库系统带有** JDBC **驱动程序，**Java **程序通过** JDBC **驱动程序实现与数据库的相连，执行查询、提取数据等操作。另外，**Sun **公司还开发了** JDBC-ODBC bridge**，用此技术** Java **程序就可以访问带有** ODBC **驱动程序的数据库，目前大多数数据库系统都带有** ODBC **驱动程序，所以** Java **程序能访问诸如** Oracle、Sybase、MS SQL Server **和** MS Access **等数据库。**

JDBC **是由** java.sql **包实现的，这个包中包含了所有的** JDBC **类和方法，如表** 6-7 **所示。**

表 6-7　JDBC **类**

类型	**类**
驱动程序	Java.sql.Drive
	Java.sql.DriveManger
	Java.sql.DrivePropertyInfo
连接	Java.sql.Connection

1 Apple **公司于** 1984 **年推出的一种系列微机，装有同名的操作系统。**

续表

类型	类
语句	Java.sql.Statement
	Java.sql.PrepareStatement
	Java.sql.CallableStatement
结果集	Java.sql.ResultSet
元数据	Java.sql.DatabaseMetaData
	Java.sql.ResultSetMetaData
日期/时间	Java.sql.Date
	Java.sql.Time
	Java.sql.TimeStamp
异常/警告	Java.sql.SQLException
	Java.sql.SQLWarning
其他	Java.sql.Types
	Java.sql.DataTrancation

每一个 JDBC 类又含有大量的方法，具体介绍略。

2．JSP 访问数据库

JSP 工作在三层结构的中间层，主要的功能是接受客户的请求，从数据库服务器获得信息，然后以 HTML 的形式返给客户浏览器。下面举例说明 JSP 访问数据库的过程。

[例 6-16]　客户信息浏览。

下面通过 Accesss 数据库，建立一个客户信息查询系统。通过 JavaBean+JSP+JDBC 实现数据库的查询以及在客户端浏览器的显示。

(1) 首先建立一个 Access 数据库 Customers.mdb，建立一个数据表 Customers，包含字段 id(自动增量型，并设为主关键字)、name(文本型，长度为 10)、address(文本型，长度为 30)、info(备注型)。

(2) 在 Windows 2000 的“控制面板”中，打开“管理工具”，双击“ODBC 数据源”图标，打开“ODBC 数据源管理器”窗口，添加“系统 DSN”，取名 Customers，并指向 Customers.mdb。

(3) 创建一个 JavaBean，名为 DBconn.java。DBconn.java 主要是封装与数据库的连接操作，内容如下：

```
package myBeans;
import java.sql.*;
public class DBconn
{
  String DBDriver = "sun.jdbc.odbc.JdbcOdbcDriver";
  String ConnStr = "jdbc:odbc:Customers";
  Connection conn = null;
  ResultSet rs = null;
```

```
  public DBconn())
  {
    try {
        Class.forName(DBDriver);
        //加载数据库驱动程序
    }
    catch(java.lang.ClassNotFoundException e)
    {
       System.err.println("DBconn ():" + e.getMessage());
    }
  }
  public ResultSet executeQuery(String sql)
  {
     rs = null;
     try
     {
        conn = DriverManager.getConnection(ConnStr);
        //与 DBMS 建立链接
        Statement stmt = conn.createStatement();
        rs = stmt.executeQuery(sql);
     }
     catch(SQLException ex)
     {
        System.err.println("aq.executeQuery:"+ex.getMessage());
     }
     return rs;
  }
}
```

将 DBconn.java **保存在** D:\MyJSP **文件夹下，编译** DBconn.java**，生成** DBconn.class **文件。**

（3**）建立相应的文件目录结构。即在** D:\MyJSP **中建立** WEB-INF **文件夹，在** WEB-INF **文件夹中建立** classes **文件夹，在** classes **中建立** myBeans **文件夹，保存** DBconn.class **文件。**

然后，在 D:\MyJSP\WEB-INF **下创建** web.xml **文档，从** webapps\ROOT **下复制即可。如果该** web.xml **不存在或者内容错误，则显示** HTTP404 **错误。修改** web.xml **后，需要重新启动** Tomcat**，使修改生效。**

（4**）建立** mycustomers.jsp **文件，在** JSP **中调用以上编译好的** JavaBeans**，其内容如下：**

```
<html>
<head>
  <meta http?equiv="Content?Type" content= "text/html; charset=gb2312">
  <title>客户信息调查</title>
</head>
<body>
<%@ page language="java" import="java.sql.*" %>
<jsp:useBean id="myDBconn" class="myBeans.DBconn" scope="page"/>
<%
  ResultSet RS = myDBconn.executeQuery ("SELECT * FROM Customers");
%>
```

```
<%! private int num=1; %>
<p align=center><b><font color="#0000FF">客户信息调查</font></b></p>
<div align="center">
  <center>
  <table border="1" cellpadding="0" width="500">
  <tr align="center">
     <td>序号</td>
     <td>客户 ID</td>
     <td>姓名</td>
     <td>地址</td>
     <td>基本信息</td>
  </tr>
  <%
  while (RS.next())
  {
    out.print("<tr>");
    out.print("<td align=center>" + num +"</td>");
    out.print("<td>" + "id"  +"</td>");
    out.print("<td>" + RS.getString("name")+"</td>");
    out.print("<td>" + RS.getString("address")+"</td>");
    out.print("<td>" + RS.getString("info")+"</td>");
    out.print("</tr>");
    num++;
  }
  RS.close();
  %>
 </table>
 </center>
</div>
</body>
</html>
```

运行上述代码，显示结果如图 6-9 所示。

图 6-9　客户信息浏览列表

对于同样的一组数据，不同的展示形式对于阅读者会产生不同的感觉效果，可以用数字、图表、折线图、直方图以及饼型图等多种形式来显示。下面通过一个实例来显示数据库中的数据。

6.5.5 使用 JSP 访问 XML 文档数据

XML 文档作为重要的数据存储形式，正在受到越来越多的关注。JSP 支持 XML 技术，JSP 的动作元素本身就采用了 XML 语法格式。在 JSP 中，对 XML 的访问是通过 Java 语言实现的。其访问途径有多种，本节重点介绍通过文档对象模型 DOM 访问 XML 的方法。

1．DOM 文档模型

在第 3 章中介绍了 DOM(文档对象模型)，采用的是 Microsoft 公司的 XML DOM 对象模型(Microsoft.XMLDOM)。它比较适合于在 HTML 中直接使用 VBScript 或 JavaScript 脚本程序来解析 XML 文档。XML DOM 包含在 Internet Explorer 5.x 中，它是一个用来存取 XML 数据的 ActiveX 组件，共有五个组件对象。

在 Java 中，除了可以使用 Microsoft.XMLDOM 模型外，还可以用其他的 DOM 模型对 XML 进行解析。下面简要地介绍 SAX 的 DOM 模型。

SAX(Simple API for XML)的 DOM 解析器是以一组简单的应用程序接口(API)来处理 XML 文档。Apache 的 Xerces 子计划支持一个 DOM 解析器，它位于 org.apache.xerces.parsers.DOMparser。DOMParser 提供了一个生成 W3C DOM 结构树的解析器以及响应方法。Sun 的 JavaXML parser 主要包含以下的包：java.xml.parsers(XML 解析器及相关对象)、org.xml.sax(SAX 模型及相关对象)以及 org.w3c.dom(DOM 模型及相关对象)。这些包被包含在 JAXP(Java API for XML Processing，即用于 XML 文档处理的使用 Java 语言编写的编程接口)中，可到 Sun 公司的网站免费下载。只要在 JSP 程序中引入这些类包，就可以使用解析器来解析 XML 文档。

下面介绍几个主要的 DOM 对象。

- DOMParser 类对象　通过该对象可以创建 DOMParser 的实例。通过该对象的 parse()方法将被解析的 XML 文档引入到解析器中，建立起 DOM 结构树。
- Document 接口对象　这是 DOM 结构树的最顶层对象。它提供了一系列的方法，如通过 getDOcType() 方法可以得到文档的类型，通过 createElement()方法来建立新元素等。一个 Document 对象也可以作为最高层的节点处理，如通过对象的 getChildNodes()方法可以得到它的下一层节点对象。
- NodeList 对象　处于结构树同一层的节点对象列表。其中的元素就是该层的所有节点。它有两个方法：getLength()和 item()。前者可得到列表中的元素个数，后者是元素索引(索引值从 0 开始)。
- Node 对象　这是 DOM 中最重要的接口对象。为 DOM 结构树中的一个节点。节点对象的类型如表 6-7 所示。

表 6-7　Node 对象类型

节点类型	说明	节点类型	说明
ELEMENT_NODE	元素节点	NOTATION_NODE	节点是一符号
ATTRIBUTE_NODE	属性节点	CDATA_SECTION_NODE	节点是 CDATA 节区
TEXT_NODE	文字节点	ENTITY_REFERENCE_NODE	节点是一引用实体
ENTITY_NODE	实体节点	DOCUMENT_TYPE_NODE	节点是一文件类型
COMMENT_NODE	注释节点	PROCESSING_INSTRUCTION_NODE	处理指令节点
DOCUMENT_NODE	文件节点	DOCUMENT_FRAGMENT_NODE	文件碎片

Node 对象的方法很多，如 getAttributes()、getChildNodes()、getFirstChild()、getLastChild()、getNextSibling()、getNodeName()、getNodeType()、getNodeValue()、getParentNode()、hasChildNodes()、setNodeValue()等。

2．应用举例

下面通过一个实例进一步了解 DOM 对 XML 文档的访问方法。

[例 6-17]　对于例 3-6 中的 XML 文档 book.xml，请在 JSP 中通过 DOM 将该文档中的数据以表格的形式显示出来。

JSP 文档(文档名 myxml.jsp)内容如下：

```
<%@ page contentType="text/html;charset=gb2312" %>
<%@ page import="org.apache.xerces.parsers.*" %>
<%@ page import="org.w3c.dom.*" %>
<%  int i;int j;
  String textNodeValue;
  String textNodeName;
  String xmlFile="d:/MyJSP/book.xml";  //被解析的 XML 文件名
  DOMParser MyParser = new DOMParser();//创建一个解析器实例
  MyParser.parse(xmlFile);           //对 XML 文件解析,创建 DOM 结构树
  Document XMLDoc = MyParser.getDocument();     //创建文档节点实例
  NodeList myNodeList = XMLDoc.getChildNodes(); //建立文档的节点列表,其中只有一个根点
  Node rootNode=myNodeList.item(0);            //获得根节点
  NodeList myNodeList1=rootNode.getChildNodes();//获得根节点下的第一层节点列表
  Node thisNode1=myNodeList1.item(1);          //获得根节点的第一层节点的第 1 号节点
  NodeList myNodeList2=thisNode1.getChildNodes();//获得第二层节点列表
  Node thisNode2=myNodeList1.item(0);          //获得第二层下的第 0 号节点
%>
<html>
<head>
<title>使用 JSP 访问 XML 文档数据</title>
</head>
<body background="1.jpg">
<h1 align="center"><font size="3" color="#ff0000">新书预告</font></h1>
<table border="1" width="100%" background="2.gif" height="67">
```

```
<tr>
<%
j=1;     //以下为输出表头。在这里以文档文本节点的节点名作为列标题
while(thisNode2!=null)  //对于第二层的所有节点对象循环处理
{
   if (thisNode2.getNodeType()==Node.ELEMENT_NODE)
   {                                    //当节点对象的类型为节点类型时
     textNodeName=thisNode2.getNodeName();//获得节点名
%>   <!-- 节点名输出到单元格中 -->
<td height="29" align="center"><%=textNodeName%></td>
<% }
   thisNode2=myNodeList2.item(j);j++;  //从节点列表中取下一个节点
}
%>
</tr>
<%
thisNode1=myNodeList1.item(0); i=1;   //获得根节点的第一层节点的第 0 号节点
while(thisNode1!=null)              //对于第二层的所有节点对象循环处理
   {
    if (thisNode1.getNodeType()==Node.ELEMENT_NODE)//当节点对象的类型为节点
    类型时
    {
      myNodeList2=thisNode1.getChildNodes();      //取下层子节点
 %>
 <tr><font size="3">
 <%
 thisNode2=myNodeList2.item(0);j=1;     //从第二层节点的第 0 号节点对象开始
 while(thisNode2!=null)               //对该层的所有节点循环
 {
   if (thisNode2.getNodeType()==Node.ELEMENT_NODE)
   {                                        //只对节点类型的节点对象处理
     NodeList myNodeList3=thisNode2.getChildNodes();  //取下一层的子节点
     Node thisNode3=myNodeList3.item(0);          //该层只有一个文字节点
     textNodeValue=thisNode3.getNodeValue();         //取出文字节点的值输
     出到单元格中
 %>
 <td width="20%" height="29" align="center"><%=textNodeValue%></td>
 <% }
 thisNode2=myNodeList2.item(j);j++; //第二层取下一个节点,继续循环
 }
 %>
</font></tr>
<%  }
  thisNode1=myNodeList1.item(i);i++;//第一层取下一个节点,继续循环
  }
 %>
</table>
</body>
</html>
```

运行上述程序，显示结果如图 6-10 所示。

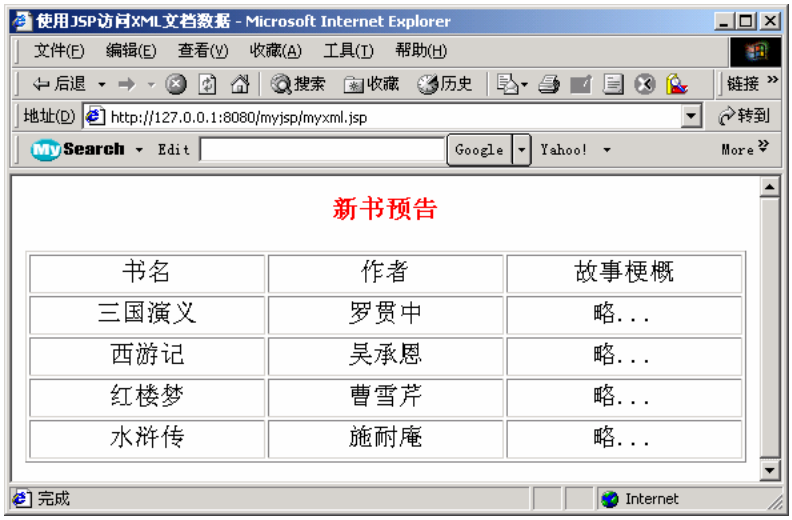


图 6-10　用 JSP 提取 XML 文档数据

6.5.6　JSP 与图形

经常看到网页上绘制各种各样的图形，如股市行情等。可以根据数据的变化，不断地更新图形，那么如何实现图形的绘制呢？

1．java.awt 包和 javax.swing 包

java.awt 包是抽象窗口工具集，其中实现了可以跨平台的图形用户界面组件，包括窗口、菜单、滚动条和对话框等，是 Java 图形用户界面编程的主要工具。

(1) java.awt 包

- java.awt　提供创建用户窗口的接口集以及绘制图形、图像所必需的类集。
- java.awt.color　提供颜色空间的类集合。
- java.awt.datatransfer　提供在应用程序内部和应用程序之间进行数据传输的接口和类的集合。
- java.awt.dnd　拖放“Drag and Drop”是很多图形用户接口系统都提供的机制，这种机制可以用于 GUI 中的两个具有图形表现元素的应用程序之间进行通信。比如 Macintosh 中的拖放执行操作等。
- java.awt.event　提供处理由 AWT 组件触发的不同类型事件的接口和类的集合。
- java.awt.font　提供与字体相联系的类和接口集合。
- java.awt.geom　提供能够在与二维几何相联系的对象上定义和执行操作的 Java 2D 类集合。
- java.awt.im　为输入方法框架提供的接口和类的集合。
- java.awt.im.spi　提供能够为任何 Java 运行时环境所采用的输入方法的开发接口

集合。

- java.awt.image　为创建与修改图形操作提供的基本类集合。
- java.awt.image.renderable　提供产生可以着色的图像的类和接口集合。
- java.awt.print　为通用打印 API 提供的类与接口集合。

(2) javax.swing 包

- javax.swing　提供保证在任何平台上都有相同行为和表现的一套轻量级组件。
- javax.swing.border　提供在 Swing 组件周围绘制特殊边界的类集合和接口。
- javax.swing.colorchooser　包含能够被 JcolorChooser 组件所使用的类和接口集合。
- javax.swing.event　提供能够被 Swing 组件所触发的事件集。
- javax.swing.filechooser　包含能够被 JFileChooser 组件所使用的类和接口集合。
- javax.swing.plaf　提供一个接口和很多抽象类，用于为 Swing 提供可插入的 look-and-feel 能力。
- javax.swing.table　提供处理 javax.swing.JTable 的类和接口集合。
- javax.swing.text　提供处理可编辑的和不可编辑的文本组件的类和接口集合。
- javax.swing.text.html　提供 HTMLEditorKit 类和支持创建 HTML 文本编辑器的支持类集合。
- javax.swing.text.html.parser　提供默认的 HTML 解析器和相应的支持类。
- javax.swing.text.rtf　提供一个用于创建 Rich-Text-Format 文本编辑器的 RTFEditor Kit 类。
- javax.swing.tree　提供处理 javax.swing.JTree 的类和接口集合。
- javax.swing.undo　为诸如文本编辑器一类的应用提供重复和撤消操作能力。
- javax.transaction　包含解分组时可能被 ORB 机制抛出的三个异常。

2．使用 JavaBean 和 JSP 在网页上绘制图形

下面通过例子介绍在 JSP 上绘制图形的方法。

[例 6-18]　利用 JSP+JavaBean 在网页上绘制图形。

利用 JSP+JavaBean 在网页上绘制图形的一般步骤是：

(1) 创建一个用来生成 JPG 文件的 Java Bean，代码如下：

```
package Charts;
import java.io.*;
import java.util.*;
import com.sun.image.codec.jpeg.*;
import java.awt.image.*;
import java.awt.*;

public class ChartGraphics
{
  BufferedImage image;
  public void createImage(String fileLocation)
  {
    try {
```

```
            FileOutputStream    fos = new FileOutputStream(fileLocation);
            BufferedOutputStream bos = new BufferedOutputStream(fos);
            JPEGImageEncoder encoder = JPEGCodec.createJPEGEncoder(bos);
            encoder.encode(image);
            bos.close();
        }
        catch(Exception e) {
            System.out.println(e);
        }
    }
    public void graphicsGeneration(int height[])
    {
        int x=10;
        int imageWidth  = 300; //图片的宽度
        int imageHeight = 310; //图片的高度
        int columnWidth = 30;  //柱的宽度
        int columnHeight= 300; //柱的最大高度
        ChartGraphics g = new ChartGraphics();
        g.image = new BufferedImage(imageWidth,imageHeight,BufferedImage.
        TYPE_INT_RGB);
        Graphics graphics = g.image.getGraphics();
        graphics.setColor(Color.white);
        graphics.fillRect(0,0,imageWidth,imageHeight);
        graphics.setColor(Color.blue);
        graphics.drawRect(0,0,imageWidth-1,imageHeight-1);
        graphics.setColor(Color.red);
        for (int i=0;i<5;i++)
        {
            x += columnWidth;
            graphics.drawRect(x, columnHeight-height[i], columnWidth, height
            [i]);
        }
        g.createImage("C:\\mytemp\\mychart.jpg");
    }
}
```

说明：这里假设创建的图片文件名为 mychart.jpg**，并保存在** C:\mytemp **文件夹中。将文件存储到** D:\MyJSP **文件夹下，编译生成** ChartGraphics.class**。**

(2**) 编写一个读取数据的** JavaBean**，文档名为** GetData.java**，内容如下：**

```
package Charts;
import java.io.*;
public class GetData
{
   int mydatas[] = new int[5];
   public int[] getDatas()
   {
      try {
           RandomAccessFile randomAccessFile = new RandomAccessFile("C:\\
           mytemp\\ mydata.txt","r");
```

```
            for (int i=0;i<5;i++)
            {
              mydatas[i] = Integer.parseInt(randomAccessFile.readLine());
            }
        }
        catch(Exception e)
        {
           System.out.println(e);
        }
        return mydatas;
    }
}
```

说明：文件 C:\mytemp\mydata.txt **为用户绘制图型所需要的数据，是一个文本文件，包含五行内容，每行一个整数。在实际应用中，这些数据可以来自数据库。**

将文件存储到 D:\MyJSP **文件夹下，编译生成** GetData.class**。**

(3**) 建立运行所需要的文件目录结构。在** D:\MyJSP\WEB-INF\classes **中创建一个** Charts **文件夹(对应包含两个** bean **的包)，将** ChartGraphics.class **和** GetData.class **复制到** D:\MyJSP\WEB-INF\classes\Charts **中。**

确保 D:\MyJSP\WEB-INF**下** web.xml **内容正确。**

(4**) 编写一个** JSP **文档，调用上面的** Java Bean**，显示图形。**Mychar.jsp **内容如下：**

```
<%@ page import="Charts.ChartGraphics"%>
<%@ page import="Charts.GetData"%>
<jsp:useBean id="myChart" class="ChartGraphics"/>
<jsp:useBean id="myData" class="GetData"/>
<%! int mydataary[]=new int[5]; %>
<%
mydataary=myData.getDatas();
myChart.graphicsGeneration(mydataary);
%>
<html>
<body>
<div align="center">
  <center>
  <table border="0" cellpadding="0" width="100%">
    <tr align="center">
      <td width="100%"><img src="C:\mytemp\mychart.jpg"></img></td>
    </tr>
  </table>
  </center>
</div>
</body>
</html>
```

(5**) 运行**

运行上述程序 mychart.jsp**，显示结果如图** 6-11 **所示。**

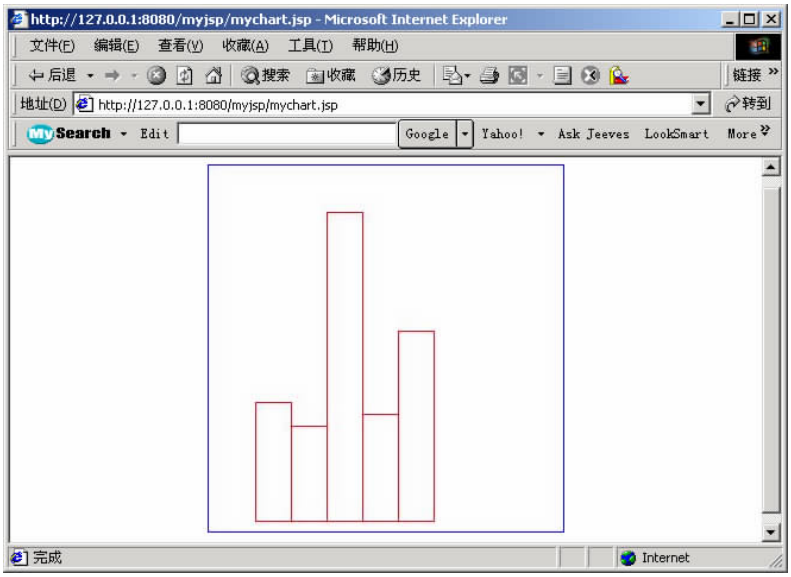


图 6-11　JSP 绘制图形输出页面

除了使用上述的方法外，还可以使用一些图表控件，如 JFreeChart 等，并基于这个控件开发自己的 JavaBean，采用开发的 JavaBean，可以将一些走势图生成 JPG 文件，然后在自己的网页里面调用它，用来显示各种走势图。

6.6　ASP、JSP、PHP 技术比较

当前，在网络开发中，除了 Servlet/EJB/JSP 外，最常用的 Web 服务器编程语言还有 ASP(Active Server Pages)和 PHP (Hypertext Preprocessor)。下面将对这些相关的技术进行对比分析，以便用户对这些技术的特点有一个简单的认识。

6.6.1　IIS 与 ASP

ASP 全称为 Active Server Pages，它是 IIS 信息服务的附加组件，含有若干内建对象，用于 Web 服务器端的开发。利用它可以产生和执行动态的、互动的、高性能的 Web 服务应用程序。ASP 采用脚本语言 VBScript(Java script)作为自己的开发语言。

1．ASP 的技术特点

ASP 具有以下技术特点：

(1) 使用 VBScript、JavaScript 等简单易懂的脚本语言，结合 HTML 代码，即可快速地完成网站的应用程序。

(2) 使用普通的文本编辑器，如 Windows 的记事本，即可进行编辑设计。无须 compile 编译，容易编写，可在服务器端直接执行。

(3) 与浏览器无关，客户端只要使用可执行 HTML 码的浏览器，即可浏览 ASP 所设计的网页内容。

(4) ASP 能与任何 ActiveX scripting 语言兼容。除了可使用 VB Script 或 JavaScript 语言外，还可通过插件(plug-in)的方式，安装第三方的脚本引擎(脚本引擎是处理脚本程序的 COM 对象)，来使用第三方所提供的其他脚本语言，如 REXX 、Perl 等。

(5) ActiveX Server Components(ActiveX 服务器组件)具有无限可扩充性。可以使用 Visual Basic、Java、Visual C++等程序设计语言编写所需要的 ActiveX Server Component。

2．ASP 文档

ASP 文件(即扩展名为.asp)的主体仍然是传统的 HTML 文件，主要由 HTML 标记、文本和脚本程序组成。除了脚本程序外，其他部分的编写和 HTML 文档相同。传统的 HTML 文档已经包含了脚本程序和外插对象，只不过它们主要是在客户端浏览器运行，依赖于客户浏览器对相关技术的支持。然而，ASP 属于服务器端应用，强调服务器端的脚本程序，脚本程序是在服务器端运行的，在性能上类似于传统的 CGI 程序。

在传统的 HTML 页中，客户浏览端运行的脚本程序被写在<script></script>标记对内。ASP 为了插入在服务器端运行的脚本程序，引入了 ASP 脚本程序定界符号(<% 和 %>)，另外在 <script>标记原始定义的基础上扩展了一个 runat 属性，表示其中的脚本在服务器端运行。使用 runat 属性时，必须使用 language 属性指定脚本语言程序的种类，例如<script language=VBScript runat=server>，表示在服务器端运行且脚本语言为 VBScript。

在服务器端编写脚本程序，如果写在 ASP 脚本程序定界符<% 和 %>之间，只能使用 ASP 默认的脚本程序语言编写，ASP 默认的脚本程序语言为 VBScript。若要在一个 ASP 文件中使用多种语言或非默认的脚本程序语言，需使用<script>标记的 language 属性来指明所用的脚本程序语言。

另外，ASP 文件中还允许使用客户端脚本程序，客户端脚本程序写在<script> … </script>标记对内，无 runat 属性，默认的脚本程序语言为 JavaScript。ASP 解释器会把客户端的脚本程序当作普通的 HTML 内容传给客户端浏览器。在客户端脚本程序内，可以插入服务器端脚本程序，借此可以产生动态的客户端脚本程序代码。一般形式为：

```
<script language="VBScript">
<!--
客户端脚本程序代码
<%服务器端脚本程序代码%>
客户端脚本程序代码
... ...
-->
</script>
```

在上面一般形式中含有 HTML 注释标记是因为当一些传统的浏览器不识别<script>标记时，<script>对应的行将作为普通的 HTML 文本在浏览器中显示，而此时，<!--和-->之间的脚本程序内容被看作注释而不被显示在浏览器中。

在客户端脚本程序段内，包含了服务器端脚本程序的代码，这些被包括在<%和%>之间的服务器脚本程序将在服务器端的 ASP 解释器扫描时被执行，产生的脚本程序内容在客

户端浏览器中执行。书写混合的客户端脚本程序可以使应用具有更强的灵活性和可交互性，更好地满足客户端的需求。

[例 6-19]　ASP 脚本文件(文件名为 AspSample1.asp)示例。

```
<html>
<head>
   <title>ASP Sample</title>
</head>
<body >
<p align="center">ASP 服务器端和客户端脚本程序实例-1</p>
<hr>
<% For I=1 to 6 %>
<script language="VBScript">
  document.writeln "<h<%=I%>>标题<%=I%></h<%=I%>>"
</script>
<% Next %>
ASP 网页下载浏览时间:<%=Now %>
</body>
</html>
```

在上面的 ASP 文件中，同时包含了 ASP 服务器端脚本(在 ASP 定界符<%和%>内部)，和客户端 VBScript 脚本。文件 AspSample1.asp 执行后，客户端的显示屏幕如图 6-12 所示。

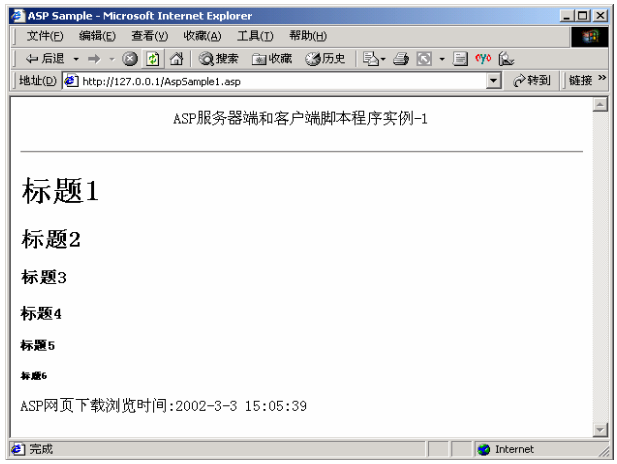


图 6-12　浏览 ASP 页面

在书写脚本程序时，需要特别注意的是，无论是在 ASP 定界符<%和%>内部还是<script></script>标记对之间，应该是完全符合脚本程序语言规范的程序语句，不应该包含独立的 HTML 标记内容。如上面的客户端脚本中，希望输出标题 1、标题 2 到标题 5 五行文本，分别用对应的标题格式。如果写成<h<%=I%>>标题<%=I%></h<%=I%>>，在浏览该网页时将会出现错误，因为它不是合法的 VBScript 语句。

又如，如果希望根据时间在浏览器中显示“上午好”和“下午好”两个不同的问候语，下面的写法是错误的：

```
<% If  Time < #12:00:00 AM#
```

```
Then <h1>上午好！</h1>
Else <h1>下午好！</h1>
EndIf %>
```

因为 HTML 标记<h1>上午好！</h1>和<h1>下午好！</h1>都不是合法的 VBScript 语句或表达式。正确的写法可以是：

```
<% If  Time < #12:00:00 AM#  Then %>
<h1>上午好！</h1>
<% Else %>
<h1>下午好！</h1>
<% EndIf %>
```

或者将第一种错误的写法改写为：

```
<script language="VBScript" >
   If Time < #12:00:00 AM# Then
      hello="上午好"
   Else
      hello ="下午好"
   End If
   document.write hello
</script>
```

3．ASP 的工作机制

和普通的.htm 文档不同，当客户通过浏览器访问某.asp 文档时，.asp 文档将不能直接被下载到客户的浏览器中。在 Web 服务器端，Web 服务器中的 ASP 解释器按照从上到下的顺序扫描 ASP 文件内容，如果遇到任何没有包含在 ASP 定界符或 <script> 标记中的文本或图形都将直接传递到浏览器，如果遇到服务器端脚本程序，将执行脚本程序，并将其可能产生的 HTML 输出依序传送到客户端，直到文件扫描结束。

在图 6-12 所示的浏览器窗口中，选择“查看”|“源文件”菜单项，可以看到文件 AspSample1.asp 执行后生成的 HTML 代码为：

```
<html>
<head>
<title>ASP Sample</title>
</head>
<body>
<p align="center">ASP 服务器端和客户端脚本程序实例-1</p>
<hr>
<script language="VBScript">
    document.writeln "<h1>标题 1</h1>"
</script>
<script language="VBScript">
   document.writeln "<h2>标题 2</h2>"
</script>
<script language="VBScript">
   document.writeln "<h3>标题 3</h3>"
</script>
```

```
<script language="VBScript">
   document.writeln "<h4>标题 4</h4>"
</script>
<script language="VBScript">
   document.writeln "<h5>标题 5</h5>"
</script>
<script language="VBScript">
   document.writeln "<h6>标题 6</h6>"
</script>
ASP 网页下载浏览时间:2002-3-3 15:05:39
</body>
</html>
```

根据 ASP 的工作原理，由于脚本程序驻留在服务器端，用户看到的仅仅是最终产生的 HTML 内容，看不到脚本程序的源代码，这样可以很好地保护网络开发人员的劳动成果。

6.6.2　JSP 技术

JSP(Java Server Page)是由 Sun 公司在 Java 语言上开发出来的一种服务器脚本语言，它包装了 Java Servlet 系统界面，减小了 Java 和 Servlet 的使用难度。在 JSP 页面上可以直接书写 Java 代码，代码需要编译成 Servlet 后由 Java 虚拟机解释执行，这种编译操作仅在对 JSP 页面的第一次请求时发生。

作为 Java 平台的一部分，JSP 拥有 Java 程序设计语言“一次编写，到处运行”的特点，既同平台无关，也同操作系统和 Web 服务器无关。JSP 可以在 Web 服务器 Apache 上运行，由于 Apache 广泛应用在 NT、Unix 和 Linux 上，因此 JSP 有更广泛的执行平台。在 NT/2000 Server 下，IIS 通过一个外加服务器(如 JRUN 或者 ServletExec)就能支持 JSP。

另外，由于 JSP 页面的内置脚本语言是基于 Java 程序设计语言的，而且所有的 JSP 页面都被编译成为 Java Servlet，JSP 页面就具有 Java 技术的所有好处，包括健壮的存储管理和安全性。

6.6.3　PHP 技术

PHP 是一种跨平台的服务器端的嵌入式脚本语言。它大量地借用 C、Java 和 Perl 语言的语法，并耦合 PHP 自己的特性，使 Web 开发者能够快速地写出动态产生页面。

PHP 可以编译成具有与许多数据库相连接的函数。PHP 与 MySQL 是现在最佳的群组合。用户可以自己编写外围的函数去间接存取数据库，通过这样的途径，当更换使用的数据库时，可以轻松地修改编码以适应这样的变化。PHP 的弱点是，提供的数据库接口支持彼此不统一。如对 Oracle, MySQL，Sybase 的接口，彼此都不一样。

PHP 是完全免费的，可以从 PHP 官方站点(http: //www.php.net)自由下载。而且你可以不受限制地获得源码，甚至可以从中加进你自己需要的特色。

6.6.4 ASP、JSP 和 PHP 的比较

ASP、JSP 和 PHP 都是 Web 服务器的开发环境，除了需要的 Web 服务器环境不同外，它们彼此之间有什么样的关系，在性能上有什么不同呢？

1．JSP 与 ASP 的关系

JSP 和 ASP 两者有很多相似之处，甚至在 JSP 中也使用了 ASP 中常用的标记符号，但两者的差异也是明显的，主要表现在以下三个方面。

(1) JSP 与平台无关，采用不同的组件模型标准。JSP 采用的是 JavaBean 标准和 Enterprise JavaBean 标准，而 ASP 采用的是 COM 标准。

(2) JSP 组件对象更多。

(3) 对数据库的访问不同，ASP 采用 ODBC 连接和 ADO，而 JSP 采用 JDBC 连接。另外，由于 JSP 提供了对 XML 的更多的支持，随着 XML 技术的推广，JSP 必将有更好的发展前景。

2．性能比较

下面是三种语言的性能测试结果。

(1) 回圈性能测试。在循环性能测试中，JSP 只用了令人吃惊的 4 秒钟就结束了 20000×20000 的回圈。而 ASP、PHP 只测试 2000×2000 循环，却分别用了 63 秒和 84 秒。

(2) 存取数据库测试。数据库测试中，三者分别对 Oracle 8 数据库进行 1000 次 Insert、Update、Select 和 Delete 操作。JSP 需要 13 秒，PHP 需要 69 秒，ASP 则需要 73 秒。

3．应用比较

目前，PHP 与 ASP 技术应用的非常广泛，而 JSP 由于是一种较新的技术，国内采用的较少。但在国外，JSP 已经是比较流行的一种技术，尤其是电子商务类的网站，多采用 JSP 技术，EJB+Servlet+JSP 几乎成为电子商务的开发标准。

由于 PHP 本身存在的一些缺点，使得它不适合应用于大型电子商务站点，而更适合于一些小型的商业站点。其缺点有以下几点：①PHP 缺乏规模支持。②PHP 缺乏多层结构支持，对于大负荷站点，解决方法只有一个，即分布计算。数据库、应用逻辑层、表示逻辑层彼此分开，而且同层也可以根据流量分开，群组成二维数组。而 PHP 正缺乏这种支持。③PHP 提供的数据库接口支持不统一，这就使得它不适合应用在电子商务中。

ASP 和 JSP 则没有以上缺陷，ASP 可以通过 Microsoft Windows 的 COM/DCOM 获得 ActiveX 规模支持，通过 DCOM 和 Transcation Server 获得结构支持；JSP 可以通过 SUN Java 的 Java Class 和 EJB 获得规模支持，通过 EJB/CORBA 以及众多厂商的 Application Server 获得结构支持。

对于 ASP、JSP 和 PHP 三者而言，JSP 应该是未来发展的趋势。世界上一些大的电子商务解决方案提供商都采用 EJB+Servlet+JSP。如 IBM 公司的 E-business，它的核心采用 JSP/Servlet 的 Web Sphere。

总之，无论是 ASP、JSP 还是 PHP，三者都有相当数量的用户，由此也可以看出三者各有所长。所以用户可以根据 Web 应用的环境以及三种不同开发工具的特点选择一种合适的语言。

6.7　Java 开发工具简介

在计算机开发语言的历史中，从来没有哪种语言像 Java 那样受到如此众多厂商的支持，有如此多的开发工具。每一种工具都各有所长。下面简要介绍一些常见的 Java 开发工具的特点。

6.7.1　JDK（Java Development Kit）

SUN 公司的 Java 不仅提供了一个丰富的语言和运行环境，而且还提供了一个免费的 Java 开发工具集（JDK）。开发人员和最终用户可以利用这个工具来开发 Java 程序。

JDK 简单易学，可以通过任何文本编辑器（如 Windows 记事本、UltraEdit、Editplus、FrontPage 以及 Dreamweaver 等）编写 Java 源文件，然后在 DOS 状态下通过 javac 命令将 Java 源程序编译成字节码（.class）文件，通过 Java 命令来执行编译后的 Java 文件，这能带给 DOS 时代程序员美好的回忆。Java 初学者一般都采用这种开发工具。

从初学者角度来看，采用 JDK 开发 Java 程序能够很快理解程序中各部分代码之间的关系，有利于理解 Java 面向对象的设计思想。JDK 的另一个显著特点是随着 Java(J2EE、J2SE 以及 J2ME)版本的升级而升级。但它的缺点也是非常明显的就是从事大规模企业级的 Java 应用开发非常困难，不能进行复杂的 Java 软件开发，也不利于团体协同开发。

6.7.2　JBuilder

有人说，Borland 公司的开发工具都是里程碑式的产品，从 Turbo C、Turbo Pascal 到 Delphi、C++ Builder 都是经典，JBuilder 是第一个可开发企业级应用的跨平台开发环境，支持最新的 Java 标准，它的可视化工具和向导使应用程序的快速开发得以轻松实现。

JBuilder 进入了 Java 集成开发环境的王国，它满足很多方面的应用，尤其是对于服务器方以及 EJB 开发者们来说。下面简单介绍一下 JBuilder 的特点。

（1）JBuilder 支持最新的 Java 技术，包括 Applet、JSP/Servlet、JavaBean 以及 EJB（Enterprise JavaBeans）的应用。

（2）用户可以自动生成基于后端数据库表的 EJB Java 类，JBuilder 同时还简化了 EJB 的自动部署功能。此外它还支持 CORBA，相应的向导程序有助于用户全面地管理 IDL（Interface Definition Language，分布应用程序所必需的接口定义语言）和控制远程对象。

（3）JBuilder 支持各种应用服务器。JBuilder 与 Inprise Application Server 紧密集成，同时支持 WebLogic Server，支持 EJB 1.1 和 EJB 2.0，可以快速开发 J2EE 的电子商务应用。

（4）JBuilder 能用 Servlet 和 JSP 开发并调试动态 Web 应用。

(5) 利用 JBuilder 可创建(没有专有代码和标记)纯 Java2 应用。由于 JBuilder 是用纯 Java 语言编写的，其代码不含任何专属代码和标记，它支持最新的 Java 标准。

(6) JBuilder 拥有专业化的图形调试界面，支持远程调试和多线程调试，调试器支持各种 JDK 版本,包括 J2ME/J2SE/J2EE。

JBuilder 环境开发程序方便，是纯 Java 开发环境，适合企业的 J2EE 开发；缺点是往往一开始难于把握整个程序各部分之间的关系，对机器硬件要求较高，运行速度较慢。

6.7.3 Eclipse

Eclipse 是一种可扩展的开放源代码 IDE，由 IBM 出资组建。Eclipse 框架灵活、扩展容易，因此很受开发人员的喜爱，目前它的支持者越来越多，大有成为 Java 第一开发工具之势。

2001 年 11 月，IBM 公司捐出价值 4 000 万美元的源代码组建了 Eclipse 联盟，并由该联盟负责这种工具的后续开发。集成开发环境(IDE)经常将其应用范围限定在“开发、构建和调试”的周期之中。为了帮助集成开发环境克服目前的局限性，业界厂商合作创建了 Eclipse 平台。Eclipse 允许在同一 IDE 中集成来自不同供应商的工具，并实现了工具之间的互操作性，从而显著改善了项目工作流程，使开发者可以专注在实际的嵌入式目标上。

Eclipse 框架的这种灵活性来源于其扩展点。它们是在 XML 中定义的已知接口，并充当插件的耦合点。扩展点的范围包括从用在常规表述过滤器中的简单字符串，到一个 Java 类的描述。任何 Eclipse 插件定义的扩展点都能够被其他插件使用，反之，任何 Eclipse 插件也可以遵从其他插件定义的扩展点。除了解由扩展点定义的接口外，插件不知道它们通过扩展点提供的服务将如何被使用。

利用 Eclipse 可以将高级设计(也许是采用 UML)与低级开发工具(如应用调试器等)结合在一起。如果这些互相补充的独立工具采用 Eclipse 扩展点彼此连接，那么当用调试器逐一检查应用时，UML 对话框可以突出显示正在关注的组件。事实上，由于 Eclipse 并不了解开发语言，所以无论 Java 语言调试器、C/C++调试器还是汇编调试器都是有效的，并可以在相同的框架内同时瞄准不同的进程或节点。

Eclipse 的最大特点是它能接受由 Java 开发者自己编写的开放源代码插件，这类似于微软公司的 Visual Studio 和 Sun 公司的 NetBeans 平台。Eclipse 为工具开发商提供了更好的灵活性，使他们能更好地控制自己的软件技术。Eclipse 联盟已经宣布将在 2004 年中期发布其 3.0 版软件。这是一款非常受欢迎的 Java 开发工具，国内的用户越来越多，实际上使用它的 Java 开发人员是最多的。它的缺点是较复杂，初学者理解起来比较困难。

6.7.4 JDeveloper

JDeveloper 的第一个版本是采用 JBuilder 的代码设计的，不过已经完全没有了 JBuilder 的影子。现在 JDeveloper 不仅是很好的 Java 编程工具，而且是 Oracle Web 服务的延伸。

Oracle9i JDeveloper(定为 9.0 版，最新为 10g)为构建具有 J2EE、XML 和 Web Services 功能的复杂、多层的 Java 应用程序提供了一个完全集成化的开发环境。它为运用 Oracle9i

数据库和应用服务器的开发人员提供特殊的功能和增强性能，除此以外，它也有资格成为用于多种用途的 Java 开发的一个强大的工具。

Oracle9i JDeveloper 的主要特点如下：

- 具有 UML（Unified Modeling Language，统一建模语言）建模功能，可以将业务对象及 e-business 应用模型化。
- 配备有高速 Java 调试器（Debugger）、内置 Profiling 工具、提高代码质量的工具 CodeCoach 等。
- 支持 SOAP（Simple Object Access Protocol，简单对象访问协议）、UDDI（Universal Description, Discovery and Integration，统一描述、发现和集成协议）、WSDL（Web Services Description Language，Web 服务描述语言）等 Web 服务标准。

JDeveloper 不仅是很好的 Java 编程工具，而且是 Oracle Web 服务的延伸，支持 Apache SOAP 以及 9iAS，可扩充的环境和 XML、WSDL 语言紧密相关。Oracle9i Jdeveloper 完全利用 Java 编写，能够与以前的 Oracle 服务器软件以及其他厂商支持 J2EE 的应用服务器产品相兼容，而且在设计时着重针对 Oracle9i，能够无缝化跨平台之间的应用开发，提供了业界第一个完整的、集成了 J2EE 和 XML 的开发环境，允许开发者快速开发可以通过 Web、无线设备及语音界面访问的 Web 服务和交易应用，以往只能通过将传统 Java 编程技巧与最新模块化方式结合到一个单一集成的开发环境中之后才能完成 J2EE 应用开发生命周期管理的事实，从根本上得到改变。缺点就是对于初学者来说，较复杂，也比较难。

6.7.5　其他工具和资源

除了上述一些工具外，常见的 Java 开发工具还有：

Visual Café for Java（Symantec 公司）、Visual Age for Java（IBM 公司）、NetBeans 与 Sun Java Studio 5（Sun 公司）、Java Workshop（Sun 公司）、WebLogic Workshop（BEA 公司）、JRUN（Macromedia 公司）、JCreator（Sun 公司）、Microsoft Visual J++(微软公司)、雅加达蚂蚁—ANT（Apache 开放源码组织）、IntelliJ IDEA（IntelliJ 公司）等。

有关 Java 的其他资源还有：

- CSDN 论坛（http://www.csdn.net/）　中国最有名的技术论坛，也是学习和交流 Java 技术非常有名的场所，《程序员》杂志是他们出版的。
- Java 研究组织 http://www.javaresearch.org/　里面有很多原创文章如高水平的文章。
- Java 开发者　http://www.chinajavaworld.com/　有比较全面的 Java 资料。

此外，通过 http://www.google.com 可以找到众多关于 Java 技术和开发的文章，也是大家学习 Java、进行 Web 开发必不可少的手段。

习　题　6

1．简述 Java 的技术特征。

2．为什么说 Java 是一种完全面向对象的程序设计语言？

3．简述对象和类的概念。
4．什么是多态？多态有哪些实现形式？
5．什么是接口？说明接口和抽象类的关系。
6．说明 Web 应用中的三层体系结构，并说明其优势。
7．说明 CGI 的功能，并简述 Servlet 和 CGI 的关系。
8．什么是 JavaBean？编写一个 JavaBean，创建网上日历本。
9．在 JSP 中用 bean 和 servlet 联合实现用户注册、登录功能。
10．使用 JSP 制作留言板。
11．使用 JSP 编写购物程序。
12．使用 JSP，根据一家商场的销售情况生成饼图。
13．利用网上资料，使用 JSP+JavaBean+数据库，尝试编写一个在线聊天程序，可从网上搜寻有关代码。

参 考 文 献

1 吴鹤龄，崔林. ACM 图灵奖——计算机发展史的缩影. 北京：高等教育出版社，2002

2 曾明. World Wide Web 开发使用指南. 北京：人民邮电出版社，1996

3 [美]Douglas E.Comer 著. 马志强，廖卫东译. Internet 导引. 北京：清华大学出版社，1995

4 金勇华，曲俊生. Java 网络高级编程. 北京：人民邮电出版社，2002

5 飞思科技产品研发中心. Java Web 服务应用开发详解. 北京：电子工业出版社，2002

6 郝兴伟. 计算机网络技术及应用. 北京：高等教育出版社，2004

7 史忠植. 知识发现. 北京：清华大学出版社，2002

8 增春，醒春晓，周立柱. 个性化服务技术综述. 软件学报，2002，Vol13(10):1000~9825

9 IBM 网格计算概述. http://www-900.ibm.com/cn/grid/index.shtml

10 网格计算理论及其应用. http://www.chinagrid.com/gridstudy/gridstudyD.htm

11 肖侬，卢锡城，王怀民. 网络计算的四种形式. http://www.yesky.com/20030313/1656808_3. shtml

12 麦中凡，陶伟. 微电脑世界，1997.7

13 http://www.ddvip.net/web/jsp/index1.asp

14 十四种 Java 开发工具点评. http://www.yesky.com/SoftChannel/72348977504190464/20040629/1825534.shtml

15 http://www.codechina.net/

16 陆正中，马进德. JBuilder 9 软件开发项目实践. 北京：清华大学出版社，2004