

Dreamweaver UltraDev 4

Web 应用程序设计

程不功 程 江 编著

人 民 邮 电 出 版 社

图书在版编目(CIP) 数据

Dreamweaver UltraDev 4 Web 应用程序设计/程不功 程江编著.—北京 :人民邮电出版社 ,2001.9
ISBN 7-115-09641-4

I.D... II. 程... 程... III.主页制作—应用软件, Dreamweaver UltraDev 4 IV.TP393.092

中国版本图书馆 CIP 数据核字(2001) 第 059103 号

内 容 提 要

本书重点讲解了利用 Dreamweaver UltraDev 4 进行网站管理、网页设计以及利用 ASP 访问数据库的方法和步骤。对于一些涉及系统内部较深层次的问题,如自定义服务器行为、创建虚拟购货车等,也在有关章节中作了较为详细的论述。

本书的主要宗旨是突出应用。从应用的需要出发提出问题,分析和解决问题。

本书适合于初、中级动态网页设计和网站开发者或业余爱好者自学使用,也可作为大专院校计算机专业学生的教材或参考书。

Dreamweaver UltraDev 4 Web 应用程序设计

◆ 编 著 程不功 程 江

责任编辑 邹文波

执行编辑 苗 颖

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号

邮编 100061 电子函件 315@pptph.com.cn

网址 <http://www.pptph.com.cn>

北京汉魂图文设计有限公司制作

北京 厂印刷

新华书店总店北京发行所经销

◆ 开本: 787×1092 1/

印张: 18.5

字数: 437 千字

2000 年 9 月第 1 版

印数: 1—0 000 册

2000 年 9 月北京第 1 次印刷

ISBN 7-115-09641

定价: 28.00 元

前 言

Web 是 Internet 中应用最广、发展最快、最受欢迎的组成部分，而网页设计和网站建设又是 Web 的核心。

几年来 Macromedia 公司以自己在多媒体领域的独特优势推出的 Dreamweaver 与 Firework、Flash 相结合的开发平台，在开发 Web 网页方面一直处于领先地位，人们用“梦幻组合”来赞颂她的辉煌。然而遗憾的是，这个系统访问数据库的能力不强，作为网站的开发工具，它的功能还显得不够完备。

2000 年初 Macromedia 公司推出了 Dreamweaver UltraDev 1.0 版本，12 月又推出了 4.0 升级版本（2001 年上半年又推出 4.01 版本）。新版本既有 Dreamweaver 的前台优势，又大大加强了后台访问数据库的能力，从而使得情况有了根本的变化。Dreamweaver UltraDev 4.0（4.01）版本与 Firework 4.0、Flash 5 相结合已经成为当今创建网站、开发动态网页的最优秀和最强大的完整工具之一。

学会使用这个系统已经成为广大设计者的共同愿望。如何使用这个系统，这里存在着 3 种不同的层次：正确使用、灵活使用和创造性地使用。一个高明的设计者不仅应该掌握利用系统提供的功能进行网页设计、访问数据库的方法，还要理解源代码的作用并具备有在必要时修改源代码的能力。围绕这种需要，本书的内容作了以下安排。

（1）重点讲解网站管理、网页设计以及利用 ASP 访问数据库的方法和步骤，为正确使用系统提供指导。这也是更高应用层次的基础。当前，网站开发者正面临着越来越大的快速开发网站的压力，在这种形势下，充分利用系统提供的功能具有特别重要的意义。

（2）在讲方法的同时本书也适当地介绍方法的实质以及有关理论，结合代码讲解原理，并且结合实例讲解补充、修改源代码的方法，为灵活使用系统奠定理论基础。

（3）对于一些涉及系统内部较深层次的问题，如自定义服务器行为、创建虚拟购货车等，本书在有关章节中也作了较为详细的论述，读者可以根据自己的需要选读这些内容。暂时跳过去也不会影响对其他章节的学习。

（4）创建网站是一门综合技术，是 JavaScript、VBScript、HTML 4.0、DHTML、CSS、SQL、ADO 数据库访问、浏览器行为、服务器行为等技术的综合。其中每一项技术都可以写成厚厚的一本书。本书则用尽可能短的篇幅将它们的核心部分提取出来介绍给读者。

本书的主要宗旨是突出应用，从应用的需要出发提出问题，结合应用深入浅出地分析问题和解决问题。既讲方法又讲思想，既讲技术又讲概念，使各种不同要求的读者从本书中都能够有所收益。

本书对内容的安排是：从第 1 章到第 13 章重点讲述多媒体网页设计，从第 14 章到 25 章重点讲述 ASP 数据库访问。其中从第 22 到 25 章涉及一些比较深层次的知识，第 25 章“创建虚拟购货车”是一个综合性的应用示例。

本书适合于初、中级网站和动态网页设计者或业余爱好者自学使用，也可作为大专院校计

算机专业学生的教材或参考书。作者曾用本书的初稿对计算机专业学生授课,取得了很好的效果。

编 者

目 录

第 1 章 基础知识	1
1.1 Internet 的发展	1
1.2 WWW 网站的工作特点	2
1.2.1 什么是 WWW	2
1.2.2 WWW 的组成	2
1.2.3 IP 地址与域名管理	4
1.3 URL 资源定位器与服务器目录管理	7
1.3.1 通用资源定位器 URL	7
1.3.2 Web 服务器的目录管理	7
1.4 Dreamweaver UltraDev 4.0 系统简介	9
1.4.1 系统特点	9
1.4.2 系统需求	10
1.4.3 系统的工作环境	10
第 2 章 Web 网站管理	13
2.1 定义本地网站	13
2.1.1 创建网站目录	13
2.1.2 为网站设置参数	14
2.2 网站管理器窗口	16
2.2.1 网站管理器窗口的显示界面	17
2.2.2 快捷菜单	17
2.3 通过站点地图管理网站	18
2.3.1 查看站点地图	18
2.3.2 修改站点地图布局	18
2.3.3 在站点管理器中显示或隐藏站点文件	19
2.3.4 使用站点地图对网页进行操作	20
2.4 浏览器兼容问题	21
2.5 文件的上传与同步	22
2.5.1 向远程网站上传文档	22
2.5.2 建立与远程网站的同步关系	23
第 3 章 HTML 文本编辑	25
3.1 HTML 简介	25
3.1.1 HTML 语言的结构	25
3.1.2 标签(标志)的属性	26
3.1.3 注释语句	26

3.2	HTML 编辑窗口的使用方法	26
3.3	关于 HTML 文档中的头<head>	27
3.3.1	元数据(meta)	27
3.3.2	增添 meta 属性的方法	28
3.4	编辑 HTML 文本	29
3.4.1	属性面板	29
3.4.2	插入文本元素	30
3.5	文本超级链接	31
3.5.1	建立页面之间的超级链接	31
3.5.2	建立与页面内部某段落的超级链接	31
3.5.3	利用链接地图快速建立超级链接	32
3.6	净化 HTML 文本	33
3.7	字符串的移动显示	33
3.8	综合示例	34
第 4 章	表格与网页布局	36
4.1	概述	36
4.2	创建表格	37
4.3	利用模板选择表格样式	37
4.4	行或列的合并与拆分	38
4.5	表格的调整和修改	39
4.5.1	添加或删除表格的行和列	39
4.5.2	表格嵌套	40
4.6	表格内容排序	40
4.7	网页的整体布局	41
4.7.1	布局视图与标准视图	41
4.7.2	使用单元表	42
4.7.3	使用单元格	42
第 5 章	插入和编辑图像	43
5.1	图像的类型	43
5.2	指定图像编辑器	43
5.2.1	Fireworks 4.0 的作用	43
5.2.2	建立 Dreamweaver UltraDev 与 Fireworks 之间的调用关系	44
5.3	插入和编辑图像	45
5.3.1	插入图像	45
5.3.2	为图像设置属性	45
5.3.3	利用 Fireworks 编辑图像	46
5.3.4	利用 Fireworks 优化图像	48

5.4	编辑和插入 GIF 动画	49
5.5	图像的超链接	50
5.6	图像的切换	51
第 6 章	框架与图层	52
6.1	框架	52
6.1.1	概述	52
6.1.2	创建框架集	52
6.1.3	设置框架的属性	55
6.1.4	删除框架	56
6.1.5	对框架内容的控制	56
6.2	图层	56
6.2.1	图层的作用	56
6.2.2	创建图层	57
6.2.3	设置图层的属性	57
6.2.4	图层面板	59
6.2.5	图层间的对齐操作	60
6.3	图层与表格之间的转换	60
6.3.1	图层转换为表格	60
6.3.2	表格转换为图层	61
第 7 章	表单	62
7.1	概述	62
7.1.1	浏览器与服务器的交互过程	62
7.1.2	表单的结构	62
7.2	表单中的两个重要属性	63
7.3	两个专用按钮	63
7.4	表单元素的 HTML 标签	64
7.5	创建表单的可视化操作	65
7.6	设置表单元素的属性	66
7.7	跳转菜单	69
7.8	表单综合示例	70
第 8 章	DHTML 与 CSS 级联样式表	73
8.1	DHTML 技术概述	73
8.1.1	基本概念	73
8.1.2	DHTML 技术	73
8.2	浏览器对象模型 (DOM)	74
8.2.1	window 对象	75

8.2.2	document 对象	76
8.2.3	location 对象	77
8.3	CSS 级联样式表	79
8.3.1	概述	79
8.3.2	调用外部 CSS 文件	80
8.3.3	在网页中自己定义 CSS	82
8.3.4	将 CSS 语句直接插入 HTML 标签的属性中	85
8.3.5	将 CSS 样式转化为 HTML 标签	85
第 9 章	JavaScript 语言基础	87
9.1	JavaScript 语言简介	87
9.1.1	JavaScript 语言的组成	87
9.1.2	JavaScript 语言的特点	88
9.1.3	JavaScript 与 Java 语言	88
9.2	JavaScript 的基本语法	89
9.2.1	JavaScript 的数据类型	89
9.2.2	JavaScript 的运算符	90
9.2.3	运算符的优先顺序	93
9.2.4	JavaScript 的流程控制	93
9.3	JavaScript 的函数	97
9.3.1	JavaScript 的全局函数	97
9.3.2	JavaScript 的函数	98
9.4	JavaScript 核心语言中的对象	100
9.5	在 Dreamweaver Ultradev 中查阅 JavaScript 的方法	103
第 10 章	时间轴	104
10.1	时间轴编辑器	104
10.2	创建一个直线运动的图片	105
10.3	图片的曲线运动	106
10.4	复合运动	106
10.5	调整时间轴	107
10.6	使用时间轴改变图层或图片的属性	107
第 11 章	浏览器端行为	109
11.1	概述	109
11.2	浏览器行为管理窗口	109
11.3	系统提供的行为	110
11.3.1	系统提供的动作	110
11.3.2	系统提供的事件	111

11.4	增加浏览器行为的步骤	112
11.5	浏览器行为示例	113
11.5.1	弹出提示信息	113
11.5.2	切换图片	114
11.5.3	信息的浮动显示及动态菜单	114
11.5.4	给时间轴附加行为	116
11.5.5	在表单中加入行为	117
11.5.6	对图层的操作	119
第 12 章	网页中插入 Flash	123
12.1	概述	123
12.2	Flash 简介	123
12.2.1	矢量图与位图	123
12.2.2	Flash 的文件格式	125
12.3	Dreamweaver 中的 Flash 图标	125
12.4	插入 Flash 电影	126
12.5	插入 Flash 按钮	127
12.5.1	插入 Flash 按钮的步骤	127
12.5.2	Flash 的按钮的属性面板	128
12.6	插入 Flash 文本	128
第 13 章	素材管理器、模板及库	130
13.1	素材管理器(Asset management)	130
13.1.1	素材管理器的作用	130
13.1.2	两种素材的列表方式	130
13.1.3	管理窗口的使用方法	131
13.2	模板	131
13.2.1	模板的作用	131
13.2.2	创建模板	132
13.2.3	调整模板	132
13.2.4	利用模板创建网页	133
13.2.5	利用素材管理器编辑模板	133
13.3	库	133
13.3.1	创建库项目	133
13.3.2	增加库项目到网页	134
13.3.3	修改库项目	134
第 14 章	ASP 技术基础	136
14.1	什么是 ASP	136

14.2	使用 ASP 的优点	137
14.3	ASP 的内建对象	137
14.3.1	Response 与 Request 对象	138
14.3.2	几个共享变量	139
14.3.3	Server 对象	140
14.4	服务器端脚本的编写方法	141
14.5	浏览器端脚本与服务器端脚本的比较	143
14.5.1	两种脚本调用不同的内建对象	143
14.5.2	关于事件驱动	144
14.5.3	脚本选择的一般原则	146
14.6	关于 Global.asa 文件	146
14.7	UltraDev 提供的服务器行为(Server Behavior)	147
第 15 章	建立与数据库的连接	149
15.1	概述	149
15.2	与数据库连接的通用接口	149
15.2.1	ODBC	149
15.2.2	ADO	150
15.3	与数据库的连接方法	151
15.4	创建数据源名(DSN)	152
15.4.1	什么是数据源名	152
15.4.2	创建数据源名的步骤	153
15.5	建立连接的步骤	154
第 16 章	创建记录集	156
16.1	什么是记录集	156
16.2	SQL 语言基础	156
16.3	记录集对象的 ASP 代码	159
16.4	利用记录集操作数据	160
16.5	在 Dreamweaver UltraDev 中创建记录集的步骤	160
16.6	管理记录集	161
16.6.1	复制记录集	161
16.6.2	编辑记录集属性	162
16.6.3	删除记录集	163
第 17 章	动态数据绑定	164
17.1	动态数据源	164
17.1.1	什么是动态数据源	164
17.1.2	数据绑定的基本步骤	164

17.1.3	利用数据绑定窗口设置数据源	165
17.1.4	删除数据源	168
17.2	各类网页元素的数据绑定	168
17.2.1	文本框的动态数据绑定	168
17.2.2	图片的动态绑定	169
17.2.3	单选按钮与复选框的动态数据绑定	170
17.2.4	列表/下拉列表(List/Menu) 的动态数据绑定	174
17.2.5	动态文本	176
17.2.6	动态数据格式	177
17.3	应用举例	179
第 18 章	Web 数据库查询与显示	181
18.1	概述	181
18.2	执行查询的操作	181
18.2.1	准备工作	181
18.2.2	单一条件查询	182
18.2.3	多条件组合查询	183
18.2.4	指定范围浏览	185
18.3	显示查询结果	186
18.3.1	多条记录分页显示	187
18.3.2	为多条记录增添颜色	188
18.3.3	设置导航按钮	190
18.3.4	设置显示/隐藏区	190
18.3.5	利用 Live 面板创建导航按钮	191
18.3.6	显示各页记录数的情况	192
18.3.7	利用 Live Objects 面板显示记录数	192
第 19 章	编辑数据库记录	194
19.1	增添记录	194
19.1.1	利用服务器行为选项 Insert Record 增添记录	194
19.1.2	利用 Live Objects 面板中的对象增添记录	197
19.1.3	利用代码添加记录	198
19.2	修改记录	199
19.2.1	利用服务器行为 Update Record 选项修改记录	199
19.2.2	利用 Live 面板中的图标修改记录	200
19.2.3	编写代码以修改记录	200
19.3	删除记录	201
19.3.1	利用服务器行为 Delete Record 选项删除记录	201
19.3.2	编写代码删除记录	202

19.4	成批处理记录	202
19.4.1	利用 SQL 语句成批处理记录	202
19.4.2	利用记录集成批处理记录	203
19.5	使用服务器行为与直接编写代码编辑记录的比较	204
第 20 章	主/从数据表的同步	205
20.1	基本概念	205
20.2	利用服务器行为实现主/从表同步	205
20.3	利用 Live Data 窗口检查同步效果	208
20.4	利用 Live Objects 面板实现主/从表的同步	209
第 21 章	存储过程	212
21.1	概述	212
21.1.1	什么是存储过程	212
21.1.2	为什么需要存储过程	212
21.2	如何定义存储过程	212
21.3	在 SQL Server 数据库中设置存储过程	213
21.4	存储过程的参数	213
21.5	在 UltraDev 中为网页定义存储过程	214
21.6	在 UltraDev 中将存储过程直接定义为记录集	215
第 22 章	编辑和自定义服务器行为	216
22.1	自定义服务器行为	216
22.1.1	编写代码块的步骤	216
22.1.2	代码的编写方法	218
22.1.3	加入待定参数	218
22.1.4	确定代码块插入的位置	219
22.2	编辑服务器行为	221
22.3	创建服务器行为示例	222
22.3.1	创建过程	222
22.3.2	使用新服务器行为	226
第 23 章	登录检查与网页保护	227
23.1	概述	227
23.2	登录检查网页的设计	227
23.3	对网页的保护	229
23.4	网页保护的原理	231
23.5	退出时的善后处理	232
23.6	重名检查	232

第 24 章 留言板	235
24.1 进入留言板	235
24.2 留言网页的设计	235
24.3 查看留言	237
24.4 留言板的管理	237
24.4.1 删除单条留言	238
24.4.2 批量删除留言	239
24.4.3 权限审查和网页保护	240
第 25 章 创建虚拟购货车	241
25.1 概述	241
25.1.1 系统功能	241
25.1.2 主要的数据表格	242
25.2 主页面(main.asp)	243
25.3 商品分类显示(newspaper.asp 或 magazine.asp)	243
25.3.1 分类的方法	243
25.3.2 提供客户选订的界面	244
25.4 创建购货车(create_car.asp)	245
25.4.1 Session 变量的初始化	245
25.4.2 生成购货车的代码	246
25.5 检查购货车(check_car.asp)	246
25.5.1 显示购货车	246
25.5.2 计算价格	247
25.5.3 购货车中对订单的修改	250
25.6 清除购货车(clear.asp)	251
25.7 将数据存入磁盘	251
25.7.1 关于客户编号	251
25.7.2 客户情况界面(customer.asp)	253
25.7.3 将客户及订单存入数据库中(store.asp)	254
附录 A VBScript 简介	256
A.1 VBScript 语法概要	256
A.1.1 VBScript 中的数据类型	256
A.1.2 变体中的数据类型	257
A.2 变量与数组	258
A.2.1 声明变量	258
A.2.2 变量的作用域	258
A.2.3 数组操作	259

A.3	VBScript 中的函数	260
A.3.1	数学函数	260
A.3.2	格式转换函数	260
A.3.3	日期和时间函数	261
A.3.4	字符串操作函数	261
A.4	流程控制	261
A.4.1	使用条件语句	261
A.4.2	使用循环语句	262
附录 B	创建虚拟购货车(程序清单)	264
B.1	创建购货车网页(create_car.asp)	264
B.2	计算(调整)购货车网页(check_car.asp)	265
B.3	输入客户情况网页(customer.asp)	273
B.4	存入磁盘网页(store.asp)	278

第 1 章 基础知识

1.1 Internet 的发展

为了学习和研究 Internet 在多个领域中的应用,先应该了解 Internet 本身的一些特点。

开始时,网络的应用范围常限于办公楼或一些集中单位的内部,我们称这种网络为“局域网”(LAN, Local Area Network)。后来出现了少量利用远程传输线路和调制解调器(Modem)等组件,将网络的应用范围扩大到较远地区的“广域网”(WAN, Wide Area Network)。但不论“局域网”还是“广域网”,每个网络都采用自己独立的工作平台。常用的有 Unix、Ethernet、Novell、Window NT 等等。由于各种类型的工作平台使用的通信规范和数据格式不同,网络之间传输互不兼容,因此计算机只能和网络内部的计算机通信而不能超越网络的界限。这些网络,如同信息海洋中的一座座“信息孤岛”。

Internet 是在“局域网”和“广域网”的基础上发展起来的。20 世纪 60 年代末,诞生了计算机“网络系统”,这种系统是由多台微型计算机(工作站)、服务器以及通信传输等软、硬件接口组成的系统。系统内的计算机既能独立工作,又能与其他计算机共享系统资源(包括软、硬件资源,数据资源等)。与原来的“多终端分时系统”相比,系统的可伸缩性强、使用方便、成本低、易于管理。它的出现,立即受到用户的热烈欢迎。短短几十年,网络系统得到了蓬勃的发展。

要充分共享信息资源,必须打破网络的界限。为此,一个基于分组交换的计算机通信网络由美国国防部高级研究计划署建立起来,他们开始时的目的只是想将国防部内各个研究机构的网络连到一起,以便交流和共享研究成果。

1980 年,Internet 已经成为一种可以实际运行的系统,1984 年它的连接扩大到了美国政府其他各个部门。1991 年,欧洲国家纷纷与 Internet 连接。1996 年进一步扩大到了全世界。1994 年我国正式与国际 Internet 以 64bit/s 专线连接。几年来,每年与 Internet 连接的计算机数量呈指数增长。

Internet 是继报纸、广播、电视之后的第四大信息传播工具。但和前面 3 种方式有很大的不同,在前面 3 种方式中,广大群众(客户)只是被动地“听”或“看”,而利用 Internet 环境下,更多的人可以通过自己的网页(或网站)发表自己的看法、意见,与其他人进行讨论和交流。

从规模上讲,Internet 是遍布全世界的“巨型网络”,其本质与传统观念中的网络不同,它是由千千万万个“局域网”或“广域网”组成的网,是网上的网,称“因特网”,如图 1.1 所示。

为了实现网络之间的“无缝连接”,Internet 使用了“路由器”(Router),路由器本身也是一台计算机,它的任务就是连接各种网络。在网络上使用统一的通信协议(TCP/IP)和编址方法,以保证与 Internet 连接的计算机都有一个唯一的地址。

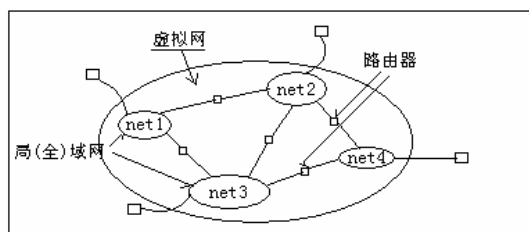


图 1.1 虚拟网络示意图

1.2 WWW 网站的工作特点

1.2.1 什么是 WWW

WWW 是英文 World Wide Web 的缩写，即通常所说的万维网，简称 Web。Web 是一种基于因特网的分布式信息查询系统。它的特点是将 Internet 的各种信息综合到一个图形界面上，将文本与图形相结合，通过超文本和超媒体实现动态传输、与客户交互等功能。由于它形象直观、易于使用且功能完备，因此成为 Internet 中发展最快、使用最广、最受欢迎的领域。

1.2.2 WWW 的组成

WWW 由 Web 浏览器(Web Browser 或称客户机) Web 服务器(Web Server) 组成。两者之间通过 HTTP 通信协议连接，形成一个客户机/服务器系统。其中，客户机主动申请所需数据，服务器产生数据以提供服务。Web 服务器有时又称为 HTTP 服务器。

1. Web 浏览器

Web 浏览器的类型很多。目前流行最广的有：

- Microsoft 公司的 Internet Explorer(简称为 IE)。
- Netscape 公司的 Netscape Navigator(简称为 NS)。

2. Web 服务器

Web 服务器的种类很多，主要有：用于 Windows NT/2000 操作系统的 Internet Information Server(简称为 IIS)，用于 Windows 95/98 操作系统的 Personal Web Server (简称为 PWS)，还有 Netscape Enterprise Servsr、WebSite Server 等。本书将重点介绍 IIS 4/5 与 PWS。

IIS 服务器

IIS 服务器是一个功能强大的 Web 服务器，它可将信息发布到全世界，为全世界的客户服务。IIS 4/5 集成在 Windows NT Server 或 Windows 2000 Server 中，可以充分利用操作系统提供的安全特性和性能方面提供的优势。IIS 4/5 提供了 3 种基本的 Internet 服务功能：文本传输 (FTP) 信息检索 (GOPHER) 和超文本超媒体服务 (WWW)。其中最重要的是 WWW 服务。

IIS 对硬件方面的要求：

- 66 MHz 486 以上 CPU，建议使用 90MHz Pentium 以上 CPU；
- 内存 32MB 以上，最好 64 MB 以上；

- 所需硬盘容量为 50 MB(最小安装), 建议使用 200MB 以上可使用空间 ;
- 显示器为 VGA , 建议使用 Super VGA ;
- CD-ROM 光驱 4 倍速以上 , 建议 6 倍速以上 ;
- 若只用于本机调试 , 则需安装一网络卡 , 指定本机的假定的 IP 地址 ;
- 若建立一个 Internet 网站 , 需安装网络卡、路由器 , 并申请取得 IP 地址。

软件方面的要求 :

- Windows 2000 或 Windows NT 5.0 或 Windows NT 4.0(Windows NT 4.0 需加 Service Pack 补充软件);
- Internet Explorer 5.0(至少 4.01 版本以上)。

PWS 服务器

安装 Windows 95/98 操作系统的 PWS 服务器是一种小型的 Web 服务器,可以在家中通过个人电脑主持 Web 站点或公司内部的 Intranet 服务器。它也能将信息发布到全世界,为全世界的客户服务,但这种服务器只适合于小型站点。对于那些有可能数百人以上同时访问的大型站点最好采用 IIS。

本书使用的是 PWS 4.0 以上版本。PWS 服务器需在 windows 95/98 的环境下安装。通常情况可在 Windows 98 光盘的目录“ \Add-ons\PWS\setup ”下找到安装软件。也可通过 Internet 免费下载。

TCP/IP 协议与 HTTP 通信协议

在 Internet 中,一台服务器常常要同时为众多的客户服务,它不可能保持与某个客户的持续连接,因此采用一种基于分组交换原理的 TCP/IP(Transmission Control Protocol/Internet Protocol) 通信协议。TCP 和 IP 是两套紧密联系而实际上又各自独立的协议,它们先将信息进行分组,将整个信息分割成许多小数据包,然后以数据包为单位进行传输。就像一封长信,先将它分解成多封短信,分别放在信封里发出,信封就相当于文件头,用来说明信的发出地址、目的地址以及其他文件的说明。这些信途经一个个中转站,到达目的地后再把它们重新组装起来。路由器就相当于中转站,它是一种特殊的计算机,它会检测数据包的目的地址,然后确定将数据包送往何处。

TCP 协议(传输控制协议) 规定了分割和重组数据所要遵循的原则以及进行的操作。在分割数据时,TCP 协议将为数据包添加一些附加的信息,指明这些数据来自何方、送往何处,以及这些数据包在文件中的顺序等,以便能在目的地将它们重新组合起来。

IP 协议(网际协议) 则负责在网络上传送由 TCP 装配的数据包。

HTTP(HTTP, HyperText Transfer Protocol) 是在 TCP/IP 基础上,为分布式超媒体信息系统而设计的通信协议。它是一种简单的、无状态的、面向对象的协议,可用于多种任务,如名字服务器和分布式对象管理。在 WWW 中,用于在浏览器与服务器之间进行通信。它不仅可以传输文本文件,还可以传输各种数据文件,如图形、图片、音频、视频以及二进制执行文件。

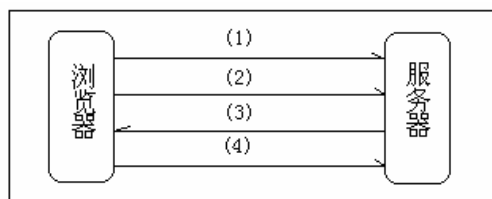
HTTP 被设计成一种简单的协议,却能够高效地处理大量请求。浏览器连接到 HTTP 服务器,只需发送请求方式和 URL 路径等少量信息。HTTP 规范定义了请求模式,最常用的请求模式只有 3 种:GET、HEAD、POST。每一种请求模式都允许客户以不同类型的消息与 HTTP 服务器通信。因为 HTTP 是小巧、简单的程序,所以和 FTP 等其他协议相比,它的传输速度较快且开销较小。

HTTP 是一种无持续连接的协议，这意味着每次连接只限一个请求。在客户连接、发出请求、收回响应后立即断开连接，所以这种协议是十分高效的。因为是无持续连接，所以在没有请求发出时，服务器不会浪费空闲时间，也不会因为保留状态而消耗一些宝贵的资源。

HTTP 由两部分组成：从浏览器发出的请求以及从服务器返回的应答 HTTP 的请求。HTTP 有简单请求和完全请求两种类型。简单请求只用一个简单的 GET 行指明所请求的页面。完全请求则由好几行组成，它的第一行包含一个命令（如 GET 命令），指定页面和协议版本，接下来的行包含 RFC822 的头部，最后是一个空行，表示请求结束。

服务器对 HTTP 请求的应答由状态行和附加信息组成。状态信息可能包含 OK 代码 200 或错误代码（如错误请求代码 400 或页面请求禁止代码 403）。

浏览器与 WWW 服务器的交互过程分为以下 4 个步骤，如图 1.2 所示。



- (1) 浏览器建立与服务器的 TCP 连接
- (2) 浏览器向服务器发出 HTTP 请求，指定要传输的页面
- (3) 服务器作出响应，包括状态码和页面文档
- (4) 浏览器断开与服务器的 TCP 连接

图 1.2 浏览器、服务器的交互过程

1.2.3 IP 地址与域名管理

1. 什么是 IP 地址

为使众多的网络在虚拟网中实现“无缝连接”，必须采用统一的编址方法。使网络内的每台计算机都有一个唯一的地址，这个地址又称 IP 地址。它的功能就相当于电话机的号码一样。

IP 地址在计算机内占用 4 个字节（32bit），可用 4 组十进制数表示，每组数字之间用点隔开。例如：

200.33.67.39

IP 地址实际由网络代码（Network ID）和网络内主机代码（Host ID）两部分组成。为适合各种不同大小网络的需要，IP 地址主要分为 3 类，各类地址的情况如图 1.3 所示。

A 类：用于大规模的网络，IP 地址的前 8 位数用于网络码（其中第 1 位“0”作为标志），后面 24 位用于主机代码。网络码介于 0 到 127 之间，也就是说全世界可以有 126 个 A 类网络地址（0 和 127 保留），而每个 A 类网络中允许有 1600 万个节点。

B 类：用于中等规模的网络。IP 地址的前 16 位数用于网络码（其中“10”作为标志），后面 16 位用于主机代码。也就是说全世界可以有 16000 个 B 类网络地址。

C 类：用于较小规模的网络。IP 地址的前 24 位数用于网络码（其中“110”作为标志），

后面 8 位用于主机代码。也就是说全世界可以有 200 万个 C 类网络地址。

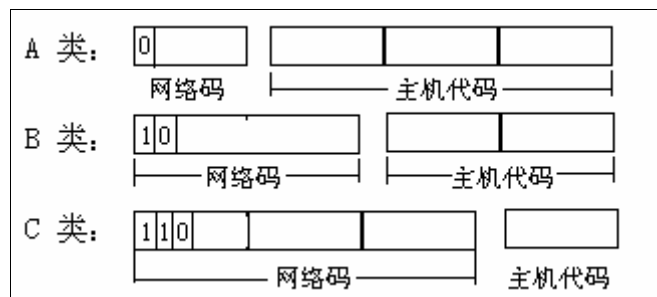


图 1.3 各类 IP 地址结构

为避免地址冲突，一个新加入 Internet 的独立网络，必须先向 InterNIC 组织(Internet Network Information Center) 申请一个 Network ID，然后自己再给网络内各台计算机分配 Host ID，两者组合成计算机的唯一的 IP 地址。

以上的分配方式为静态分配方式，此种方式在连接 Internet 的计算机数量飞速增长的情况下，已产生矛盾，因此一个可改善此现象的新协议 SIP(Simple Internet Protocol) 出现了。SIP 占用 64 个 bit，并且与 IP 地址兼容。

2. 域名管理

数字化的 IP 地址虽然能够满足 Internet 地址的唯一性要求，但也存在着明显的缺点，就是不容易记忆。一种好的编址方法不仅应具备区分功能，还应该直观、容易记忆。因此人们用类似于英文的字符串来代替 IP 地址，这个字符串就是域名。

域名结构

在最初的 Internet 中，域名采用的是一种无层次的命名结构。每一个主机的地址用一串简单的字符来表示。域名的管理也集中于 Internet 网络信息中心(NIC，Network Information Center)。由该组织负责维护整个名字空间，确认名字的合法性。

无层次的域名管理和映射都很简单，但只适合于管理少量的域名，在连接 Internet 的网络大量增加的形势下，逐渐暴露出新的矛盾，突出的问题是容易产生重名冲突，其次是单一集中的方法大大加重了管理机构的负担，执行效率变得越来越低。

于是产生了层次型命名机制，在域名中引入“结构”的概念。域名的结构是层次型的，域名的管理机构也是层次型的。其管理的方式是，先将名字空间划分成若干段，每一段由某个经授权的机构管理，被授权的机构又将其管辖的名字空间作进一步划分，再授权给子机构管理。每层管理机构有权管理本层的名字域，并给主机赋予一定的标志，整个名字空间管理机构组成了一个树型的层次结构。从树叶到树根各标志的有序序列就组成了主机的域名。只要在同一层上的标志不重名，就不会引起重名冲突。

TCP/IP 域名

在 TCP/IP 中所实现的层次型名字管理机制叫做域名系统(DNS，Domin Name System)。域名采用以下格式：

网站名.类型.国家(地区)名

例如：我国清华大学网络的域名是：

tsinghua.edu.cn
其中 cn 代表国家(中国), edu 代表类型(教育组织), tsinghua 代表网站名(清华大学)。
再如：美国 Macromedia 公司的域名是：
Macromedia.com
域名后面没有国名。没有国名时代表美国。
国家(地区) 名的缩写如表 1.1 所示。

表 1.1 几个国家(地区) 名的缩写表

国名	缩写	国名	缩写	国名	缩写
中国	cn	印度	in	巴西	br
韩国	kr	英国	uk	科威特	kw
日本	jp	德国	de		
泰国	th	比利时	be		
新加坡	sg	法国	fr		
智利	cl	瑞典	se		
南非	za	美国	不注明		

网站类型如表 1.2 所示。

表 1.2 网站类型表

网站类型	代码
商业组织	com
教育组织	edu
政府组织	gov
国防组织	mil
网络服务机构	net
国际组织	int
其他非赢利机构	org

域名必须转换成 IP 地址后才能被计算机识别。这个转换工作由 Internet 中的域名服务器 DNS(Domain Name System) 自动完成。域名服务器实际上是一种软件咨询系统。每当它碰到一个域名时，就会通过相应的表格转化成相应的 IP，然后在通信中使用这个 IP 地址。

1.3 URL 资源定位器与服务器目录管理

1.3.1 通用资源定位器 URL

通用资源定位器(URL)是 Uniform Resource Locator 的英文缩写,是用来为 Internet 网上的某个网页或某个文件定位的一串字符。URL 完整的格式为:

Protocol:// Hostname:Post / Path / file /path?perm1=value1&perm2=value2...

其中:

Protocol 为通信协议,如 http、ftp、gopher 等。

Hostname 为服务器域名(或 IP 地址或服务器主机名)。

Post 为主机的端口,通常使用缺省值,Web 服务器的缺省值为 80。

Path 目录别名。

File 代表文件名。

Path 信息行动路径。

Perm1,perm2...为参数名。

Value1,value2...为各参数的值。

例如:一个完整的 URL 如下。

<u>http://www.myset.com/scripts/webtime.dll/lookup?cj=good&sex=female</u>				
通信协议		子目录名	行动路径	
	服务器域名	文件名		参数

如此长的 URL 只能用于程序中。配合服务器的目录管理,用户可以用较短的形式找到服务器及服务器中的文件。

使用的形式可以是 http://域名/文件名。

例如:

http://www.cheng.com/ example.htm

或

http://www.myset.com/ webtime.asp

甚至可用更加短的格式直接找到站点的主页,其形式如下:

http://域名

1.3.2 Web 服务器的目录管理

服务器端的文件必须按照一定的规律存放才有可能迅速被找到,因此服务器上的目录管理比单机上的目录管理显得更加重要。

1. 缺省目录的安装

安装时, IIS(或 PWS) 提供了以下目录作为缺省目录,分类存放以下文件。

■ C:\InetPub\wwwroot 保存 HTML 文件。

■ C:\InetPub\scripts 保存运行文件，如*.dll、*.exe 等。

■ C:\InetPub\FTP 保存 FTP 协议文件。

其中“ C: ”不是固定的，若安装到 D 盘，则改用“ D: ”等，但建议下面的目录不要改动，因为当浏览器访问时，服务器会自动到这些目录下去寻找相关文件。

2. 虚拟目录

虚拟目录有时又称为目录别名。在 URL 中普遍采用虚拟目录代替实际目录，以简化目录的长度，IIS(或 PWS) 提供了几个虚拟目录。

例如：

虚拟目录“ /scripts ”代表的实际路径是“ C:\InetPub\scripts ”。

虚拟目录“ /iisadmin ”代表的实际路径是“ C:\IntSrv\iisadmin ”。

虚拟目录“ /cgi-bin ”代表的实际路径是“ C:\InetPub\wwwroot\cgi-bin ”。

[注意]：实际路径以反斜杠“ \ ”分隔，而虚拟目录用正斜杠“ / ”分隔。

允许在上述目录下增添自己的子目录。例如在实际路径... \scripts 的目录下增添一个子目录“ cheng ”。系统将自动将其转换为虚拟目录“ /scripts/cheng ”。

可以在服务器中设置自己的虚拟目录。以 PWS 服务器为例，创建服务器虚拟目录的步骤如下。

(1) 进入 Personal Web Server 的“ 个人 Web 管理器 ”。

(2) 在 Web 管理器中单击“ 高级 ”图标后再单击“ 添加 ”按钮，进入“ 添加目录 ”对话框，如图 1.4 所示。

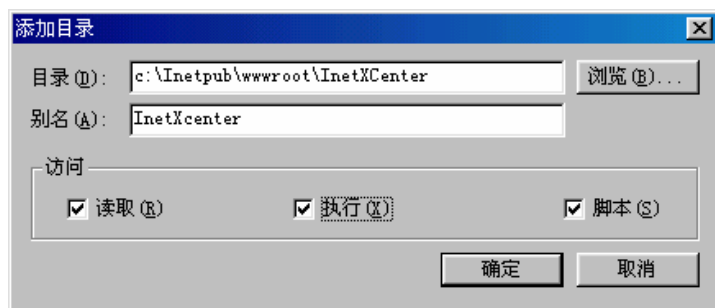


图 1.4 创建虚拟目录

(3) 在添加目录对话框中填入以下参数。

■ 在目录文本框中填入服务器的物理目录。

■ 在别名文本框中填入对应的别名(虚拟目录)。

■ 在访问栏的选项中根据需要选取“ 读取 ”、“ 执行 ”和“ 脚本 ”项。

(4) 单击“ 确定 ”按钮，如果虚拟目录对应的实际目录不存在时，将提示“ 是否创建该目录 ”。此时只需单击“ 是 ”按钮，系统将自动创建该目录。

3. 主页(Home Page) 的缺省目录

为方便查找，通常将文件的索引作为服务器主页(Home Page) 放在缺省目录中，用户访问时只需使用域名或 IP 地址，不必指定特定文件。

例如：

http://服务器域名

或 http://IP 地址

以上两种形式均可自动访问这个站点的主页(默认文档),然后再通过主页的索引进一步找到自己需要的内容。

IIS 或 PWS 提供的主页缺省的实际目录是:

C:\InetPub\WWWroot

主页文件的缺省名是: Default.htm 或者 Default.asp。就是说将主页以 Default.htm 或 Default.asp 命名,并放在 C:\InetPub\WWWroot 目录下时,用户只需通过域名或 IP 地址即能访问到这个文件。

1.4 Dreamweaver UltraDev 4.0 系统简介

1.4.1 系统特点

近几年来 Macromedia 公司推出的 Dreamweaver 与 Firworks、Flash 相结合的软件在创建多媒体网页方面独占鳌头,风靡世界,被人们赞为“梦幻组合”。但遗憾的是它访问数据库的能力并不太强。2000 年初推出的 Dreamweaver UltraDev 1.0 以及 2000 年 12 月推出升级后的 Dreamweaver UltraDev 4.0(2001 年又推出了 4.01 版本)后,情况有了根本性的变化,升级后的 Dreamweaver UltraDev 不仅完全保持了原有的优势还大大加强了后台访问数据库的能力,使得它成为 Web 开发者公认的高效而又完整的首选工具。可应用于快速开发新闻和信息、网上教学、企业内部管理、电子商务以及其他大型综合网站等工作。其系统有以下特点。

- Dreamweaver UltraDev 是由 Dreamweaver 和 UltraDev 两部分组成。Dreamweaver 主要用于开发动态网页,UltraDev 主要用于开发服务器端服务,两者一起组成了一个前台与后台密切结合的完整系统。

- 超强的多媒体功能。除继承了 Dreamweaver 4.0 与 Fireworks 4 和 Flash 5 集成的全部优势外,还以开放的形式将 ActiveX、Java applet 等技术纳入于系统中。

- 系统提供了大量内部函数和脚本,在很多情况下开发者可以不写一行代码,或者只需写上少数几行代码就能完成需要书写几百行代码才能完成的工作。与此同时系统还给设计者提供了编写脚本、创建新服务器行为等方面的工具,为快速而又灵活地使用系统提供了方便。

- 它是第一个允许在同一环境中选用 ASP、JSP 或 ColdFusion 等不同语言的系统,从而能在最大范围满足开发者的不同习惯和爱好。

- 系统提供了非常友好的设计界面。设计者可以在可视界面、代码界面、混合界面之间快速切换。系统还提供了活动数据窗口(Live Data),设计者可以在不脱离设计环境的条件下直接预览到在浏览器中运行的部分情况。

- 系统提供的素材管理、模板、库等工具,为开发组多人共同开发网站提供了在不同的网页设计中重复使用某些网页元素的功能。

1.4.2 系统需求

运行 Dreamweaver UltraDev 4.0 至少应具备以下条件:

- 奔腾 166MHz 及以上的 CPU。
- Windows 95/98/2000/ME 或者 Windows NT 4.0 (Service Pack 5)。
- 64MB 以上内存。
- 170MB 以上可用的硬盘空间。
- 256 色以上 800×600 分辨率的显示器。
- Microsoft IIS 4.0 或 5.0 服务器或者 Microsoft Personal Web Server (Windows 98)。
- Microsoft Data Access 2.5 版本以上。

1.4.3 系统的工作环境

下面对 Dreamweaver UltraDev 系统的工作环境作一个简要的介绍。

1. 进入 Dreamweaver UltraDev 4 环境

为了进入 Dreamweaver UltraDev 4 的工作环境，先单击 Windows 左下角的“开始”按钮，然后选择程序\Macromedia Dreamweaver UltraDev 4\Dreamweaver UltraDev 4 选项即可，这个过程见图 1.5 所示。

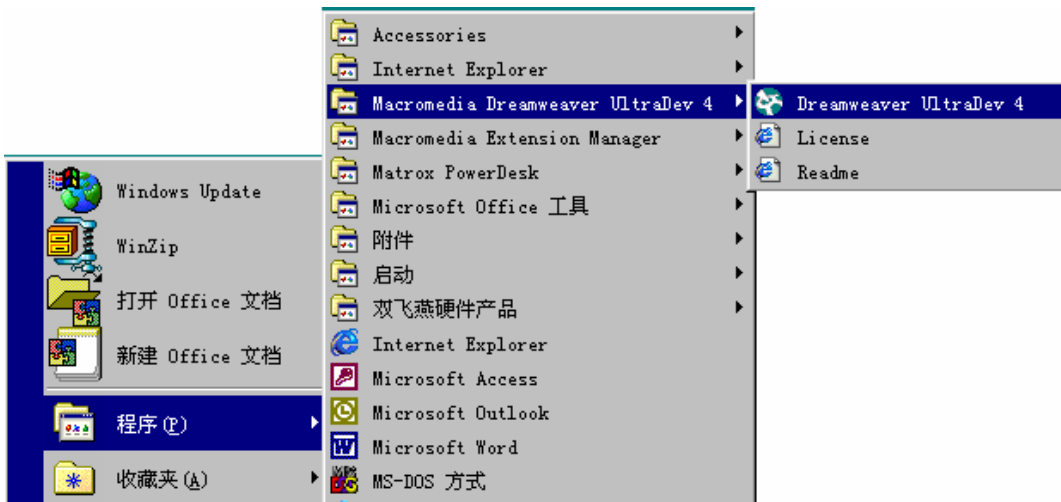


图 1.5 启动 Dreamweaver UltraDev 的界面

2. Dreamweaver UltraDev 4 的设计界面

Dreamweaver UltraDev 4 的设计界面包括十几个大小窗口和面板 (本书中将大的称为窗口，小的称为面板)，这些窗口和面板的作用是帮助管理各类组件，为对象设置属性和行为。窗口和面板中大多采用可视化操作，给网页设计带来极大的方便。但这些窗口和面板平时并不显示在屏幕上，只是在需要使用时才通过 Window 菜单选项把它们显示出来，不需要时再隐藏起来，用这种方法可保持屏幕的清晰度。对于一些常用的窗口和面板也可以利用快捷按钮进行打开和关闭的操作。

图 1.6 列出了几个常用的窗口，图 1.7 列出了主窗口右下方状态栏中的几个快捷按钮，按钮左边还有“屏幕大小(宽*高)”以及“文件容量/下载时间”两项。单击“屏幕大小(宽*高)”右边的“▾”按钮，将弹出几个常用的屏幕尺寸供选择。

3. Dreamweaver UltraDev 4 中几个设计窗口的快速切换

Dreamweaver UltraDev 4 中还提供了几个设计窗口之间快速切换的按钮，如图 1.8 所示。利用它们可以在设计界面、代码界面、代码及设计界面以及活动数据界面之间快速切换。例如你可以快速在代码、设计页面之间转换，甚至可以将设计和代码页面同时显示在同一个界面中。这样，就可以在改写代码的过程中直接观察到设计页面上的效果，也可以在设计页面中增添、删除 HTML 元素时直接观察到源代码的变化情况。

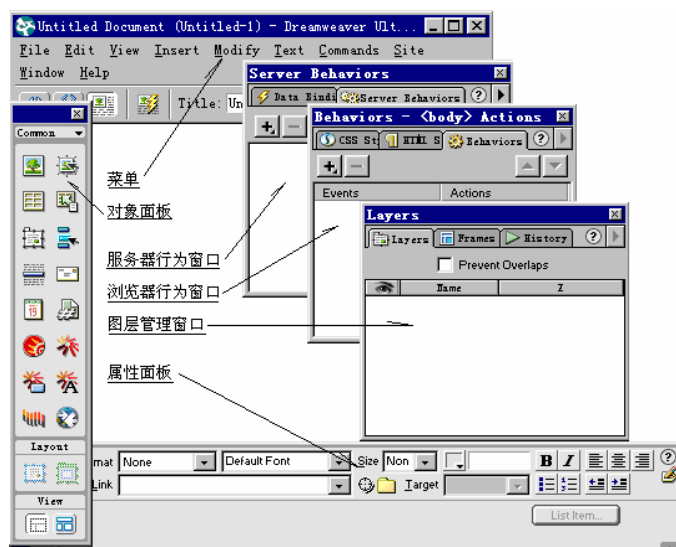


图 1.6 设计界面的几个主要窗口



图 1.7 状态栏



图 1.8 页面间快速转换按钮

图 1.9 显示的是混合界面的情况，窗口的上半部为代码界面，下半部为设计界面。

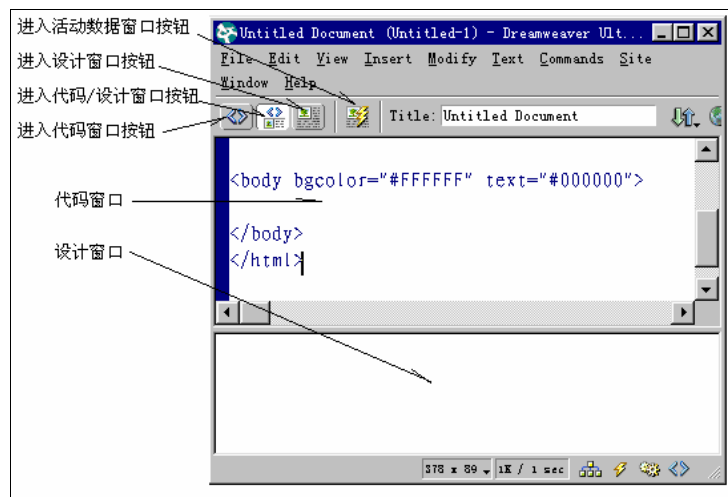


图 1.9 几个设计窗口的快速切换

第 2 章 Web 网站管理

一个网站是 Web 中的一个节点，也是一个信息源，Web 就是由千千万万个这样的信息源组成的信息系统。为了发布信息，网站至少应包括一组为执行某项任务而相互关联的网页，网站可看成是这些网页的集合；为了保存信息，网站还应有自己的数据库等。除此之外，网站还应满足以下条件。

- 一台以上装有 Web 服务器及有关软、硬件的计算机。
- 申请并注册了 Internet 上唯一的域名及 IP 地址。
- 服务器与 Internet 能够保持持续的连接。

由于网站通过 Internet 面向的是全世界，而世界上各个地区的时差相差很大，一个地区的半夜可能正是另一个地区的中午。因此，为了随时接受来自世界各地客户的访问，应保持与 Internet 的不间断连接。

开发网站的过程，通常是先在自己的计算机上创建各类文档，调试成功后再将全部文档上传到服务器去发布。这样，就出现了两个结构相同的网站：一个是本地网站用于创建和调试，另一个是服务器网站用于信息发布。服务器网站又称为远程网站。“远程”在这里只代表分离的意思，不完全是一个距离的概念。

创建一个网站应做好 3 方面的工作。

- 创建并调试好本地网站上的各类文档。
- 将本地网站上的文档上传到远程网站。
- 建立本地网站与远程网站之间的同步。

有些中、小企业需要建立网站，但又一时缺乏足够的人力和财力时，可以从各地的 ISP（Internet Server Provider）公司租借部分磁盘，将自己网站的文档放在租借的磁盘上向全世界发布并接受反馈回来的信息。

2.1 定义本地网站

创建好本地网站是全部工作的基础。创建前要对网站进行全面规划：确定好网站的主题、网站内容针对的主要对象、各网页的内容分工等等。然后再分别对各网页进行精心设计。实践证明，只有那些主题明确、特点鲜明、内容丰富而又形式生动的网页才能吸引广大客户的注意。

下面就创建网站的几个具体技术问题作些介绍。

2.1.1 创建网站目录

Dreamweaver UltraDev 规定，每个网站必须有自己单独的目录，网站中所有网页和其他相关文件（如图片、数据库文件等）都要放在这个目录或子目录下。这样做的好处是显而易见

的，它不仅便于对网页以及其他文件的管理，也有利于与远程网站同步。网站目录示例如图 2.1 所示。

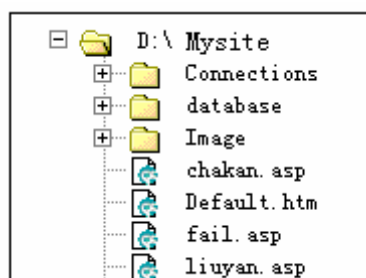


图 2.1 本地网站目录示例

其中 D:\Mysite 是网站的根目录，Connections、database、Image 等是网站的子目录，chakan.asp、Default.htm 等是网站的网页。

本机网站上也应该安装浏览器和服务。不过这不是用来向外发布信息 and 接受信息的，而是用来检查和调试本地网站上的网页的。

2.1.2 为网站设置参数

必须为网站设置必要的参数，网站才能正常工作。Dreamweaver UltraDev 提供了多个对话框，利用这些窗口可以方便地为网站设置参数。

进入 Dreamweaver UltraDev 的设计界面，选用主菜单的 Site | New Site 选项，将弹出图 2.2 所示的对话框。此对话框分左、右两部分。左边的 Category 中的 6 大项，包括了参数的全部内容。右边框体中显示的是各大项中需要设置的具体参数。在本章中将重点说明前 3 项设置的参数。

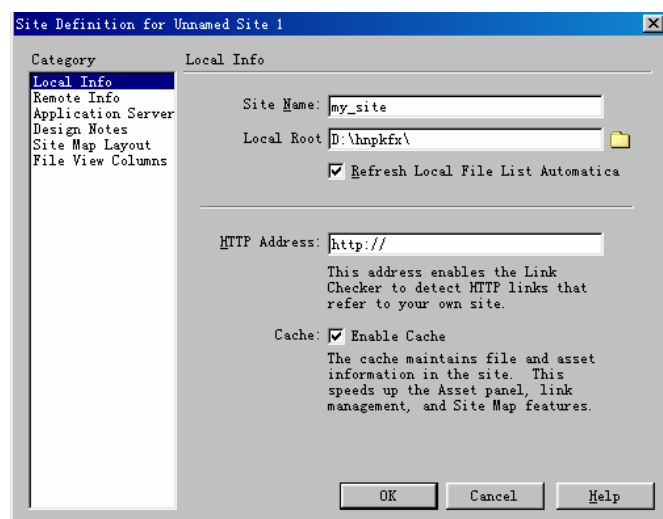


图 2.2 本地网站信息对话框

(1) 选择 Local Info(本地网站信息) 项, 然后在右边 Site Name 对话框中填入网站名, 用手动或者利用右边的文件夹图标在 Local Root 对话框中填入本地网站的根目录。网站名并不会显示在浏览器中, 它的作用只是为了方便网站管理。以后每当用菜单 Site | Open Site 选项选择此网站名时, 将自动打开这个网站并进入该网站的根目录。

Refresh Local File List Automatica 选项如果被选中, 则每当文件拷贝到当前网站, 或者站点内有文件被删除时, 网站管理器窗口将自动进行刷新并显示这些变化。如果不选用此选项, 可以提高文件拷贝的速度, 但站点窗口不会自动反映站点内文件的变化。此时只有单击站点管理器窗口上的 Refresh 按钮, 才会显示拷贝或删除后的文件变化情况。

在 HTTP Address 的文本框中输入站点的 HTTP 地址, 以便 Dreamweaver UltraDev 对文档的绝对地址进行校验, 如果没有申请域名, 可以暂时输入一个简单易记的名字, 申请域名以后再行更换。

选择 Cache 项可以为本地网站创建缓冲区, 以提高站点内文件维护和传送的速度。Dreamweaver 的素材管理器(Assets panel) 也只能在缓冲区中运行, 因此最好选择该项。

(2) 选择 Remote Info 项, 将弹出如图 2.3 所示的对话框。这里的 Remote Info 实际上是本机上安装的服务器的相关信息。

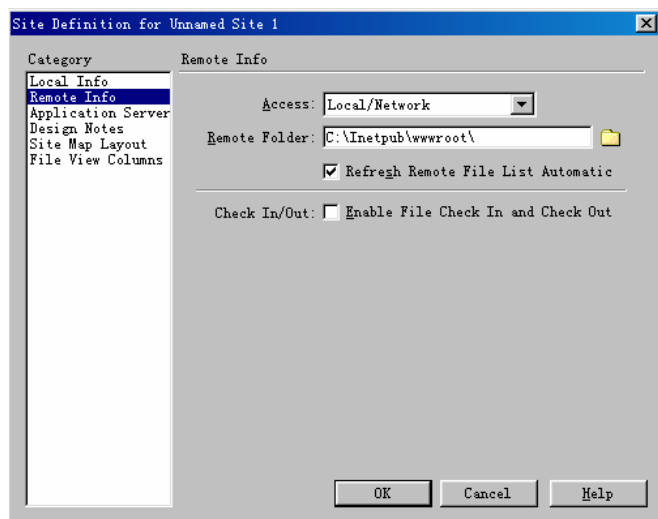


图 2.3 本机服务器信息对话框

在 Access 下拉菜单中有 3 个选项: non、FTP、Local / Network, 它们的作用说明如下。

- non: 当不准备使用服务器时, 可选择本项。选择本项并单击“ OK ”按钮时, 将跳过后面所有的参数设置过程。

- FTP: 当完成本地站点的创建, 需要将文档上传到服务器站点时使用。

- Local/Network: 当需要使用本机的 Web 服务器进行检验和调试时, 应选择此项。选择此项后还应在下面的 Remote Folder 文本框中直接输入或者利用文件夹图标输入本站点文件在本机服务器中存放的目录(包括路径)。在缺省安装条件下, 这个目录是:

C: \ inetpub \ wwwroot \

Refresh Remote File List Automatica 选项与前面 Refresh Local File List Automatica 选项的作

用相同，不过此选项只对站点管理器窗口的 Remote site 窗口起作用。

如果不选用此项，则需用手动方法在站点管理器窗口中，选择菜单 View | Refresh Remote 项时才能更新 Remote site 窗口。

Check In/Out 选项只有在开发组共同开发一个项目时，或者虽然是单独开发，但文件来自不同的机器时才需要设定。

(3) 选择 Application Server technology (应用服务器使用技术) 项后将弹出如图 2.4 所示的对话框。

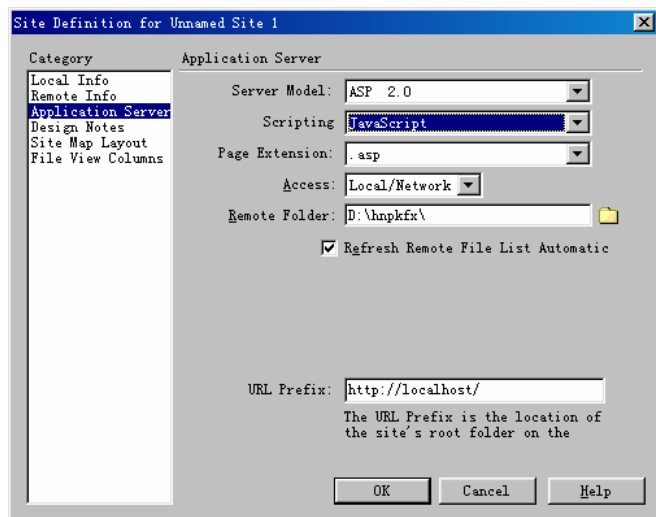


图 2.4 应用服务器参数设置窗口

应用服务器使用的技术包括以下几个方面。

- Server Mode(运行模式): 在 Server Model 的下拉菜单中选择服务器运行模式，其中包括 ASP 2.0、JSP 1.0、ColdFusion 4.0 三种，本书要求选择 ASP 2.0 运行模式。

- Scripting(脚本语言): 如果选择 ASP 2.0 还需在 Scripting 下拉菜单中，选择缺省的脚本语言 VBScript 或 JavaScript。

- Page Extension(网页扩展名): 在 Page Extension 中确定站点中网页使用的缺省扩展名，是 *.htm 还是 *.asp。动态网页通常使用 *.asp 作为扩展名。扩展名 *.htm 或 *.html 只适合于 HTML 文本及图形的网页 (*.asp 网页对两种情况都能适用)，因此通常选择 *.asp 作为网站内网页的缺省扩展名。

2.2 网站管理器窗口

网站管理器窗口是管理网站内各类文档的工具。通过网站管理器窗口可以完成给网站增添新目录、打开网页或增添网页、改变某些网页的属性以及文档的传送等工作。

当在 Dreamweaver UltraDev 的设计界面中选用 Window | Site Files 菜单时，可进入网站管理器窗口界面。网站管理器的窗口界面如图 2.5 所示。

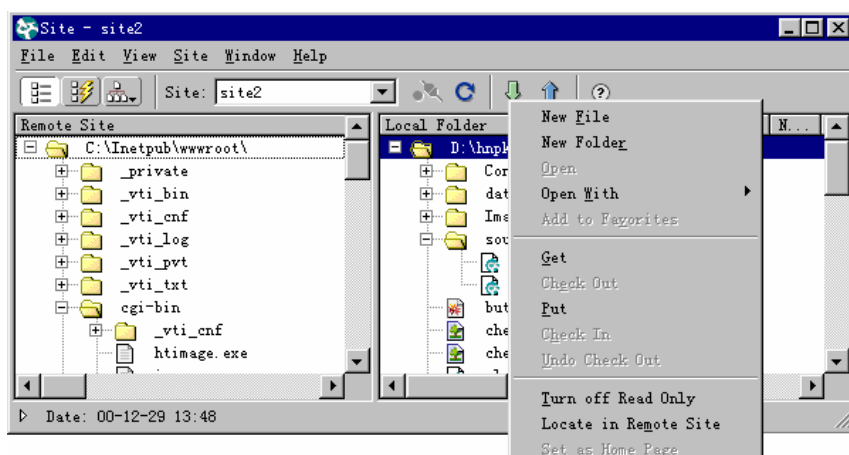


图 2.5 网站管理器窗口

2.2.1 网站管理器窗口的显示界面

整个窗口分为左右两个部分，如果前面对参数的设置正确，左边窗口中将列出服务器站点（Remote Site）下的文档情况，右边窗口中将列出本地网站（Local Folder）的文档情况。

窗口上方有一排按钮，如图 2.6 所示。现将这些按钮的作用说明如下。

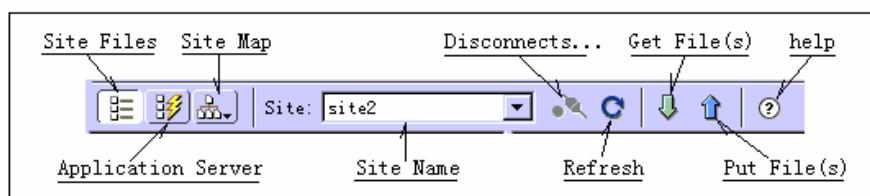


图 2.6 网站管理器控制按钮

- Site Files(View): 在左、右窗口中都显示文件列表。
- Application Server : 显示应用服务器的目录系统。
- Site Map : 用站点地图显示文档之间的链接情况。
- Site Name : 已经创建的站点名列表。
- Connects / Disconnects : 连接/断开连接按钮。当利用 FTP 协议与远程 Web 服务器进行通信时用来控制连接的状态。
- Refresh : 刷新窗口中的文件显示。
- Get File(s) : 从 Remote Site 下载单个或多个文档到 Local Folder 中来。
- Put File(s) : 从 Local Folder 上传单个或多个文档到 Remote Site 中去。

2.2.2 快捷菜单

用鼠标右击窗口，将弹出快捷菜单，如图 2.5 所示。利用其中的选项可对网站中的网页进行多项操作，例如：

- (1) 创建和打开文档，使用 New File、New Folder、Open 等项。
- (2) 传送文档，使用 Get、Put 项。

- (3) 改变文档属性, 使用 Turn off Read Only(解除只读属性) 项。
- (4) 浏览与同步, 使用 Preview in Browser、Synchronize 项。
- (5) 文档编辑, 使用 Cut、Copy、Delete 及 Rename 等项。

2.3 通过站点地图管理网站

2.3.1 查看站点地图

站点地图是一个形象化的工具, 它可用来直观显示网站内网页之间的链接情况。单击网站管理器窗口上方的“”图标, 将弹出 Map Only 与 Map and Files 两个选项, 如图 2.7 所示。



图 2.7 站点地图选项

Map Only 选项只显示站点地图, Map and Files 选项同时在左右两个窗口中分别显示站点地图与文件列表。不管选择哪项, 系统都将弹出提示, 要求先指定主页。

主页就是网站默认的首页。当用户通过网站域名(或 IP 地址) 访问网站而没有指定文件名时将自动打开主页。站点地图也是以主页为“根”显示网页间的链接关系的。

确定主页的方法是, 在文件列表中用鼠标右击某个页名, 在弹出的菜单中选择 Set as Home Page 选项, 该网页就变成了主页。

站点地图的左下方有一百分比数字, 单击后可以拉出几个选项: 100%、75%、50%、25%、12%等, 如图 2.8 所示。这些百分比代表站点地图的显示比例, 在同一个窗口中, 选择的比例愈小, 能够显示的网页数愈多但清晰度也愈下降。

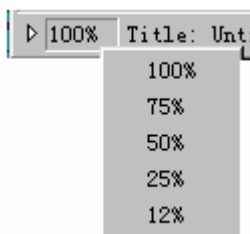


图 2.8 站点地图显示比例

2.3.2 修改站点地图布局

修改站点地图布局的步骤如下。

- (1) 从 Site 菜单中选择 Define Sites 选项, 打开 Define Sites 对话框。

(2) 在对话框中选择站点，然后单击 Edit 按钮，打开 Site Definition 对话框，如图 2.9 所示，在对话框左边的 Category 列表中选择 Site Map Layout 选项。

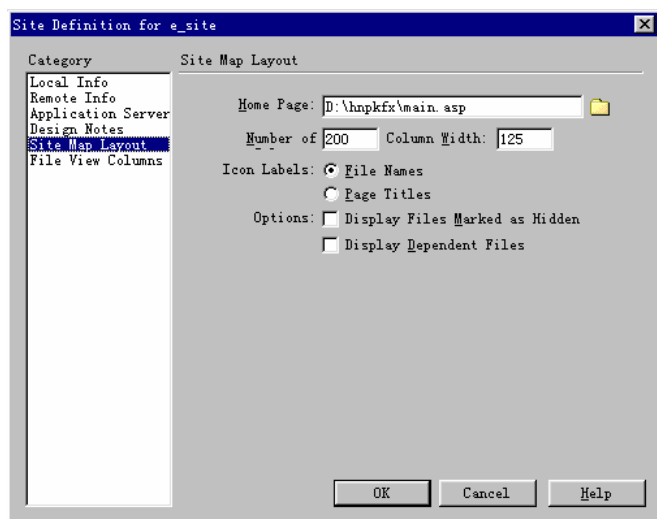


图 2.9 修改站点地图布局

(3) 按如下方法输入参数。

- 在 Home Page(主页) 的文本框中输入主页的名称及路径，也可以利用右边的文件夹图标选择主页文件。如果没有为站点指定主页，并且在站点的根目录中也没有找到名为 Index.htm 或 Index.asp 的文件，则在打开站点地图时，系统会提示必须为站点地图指定一个主页。

- 在 Number of Columns(列数) 文本框中，输入站点地图中每行显示的页面文档的数目。

- 在 Number of Width(列宽) 文本框中，输入站点地图中每列显示的页面文档的数目。

- 在 Icon Labels(图标标记) 中有两项选择。选择 File Names(文件名称) 时，在图标下显示文件名称；选择 Page Titles(页面标题) 时，在图标下显示网页的标题。

- 选中 Display Files Marked as Hidden(显示隐藏文件的标记) 时，将显示隐藏文件，此时的隐藏文件用斜体表示；选中 Display Dependent Files(显示相关文件) 时，将在站点的各个层次中显示图片等非 HTML 内容的相关文件以及它们之间的链接关系。

2.3.3 在站点管理器中显示或隐藏站点文件

直接通过站点地图也可以完成以下操作。

1. 将网页设置为隐藏

在站点地图中选择要隐藏的一个或多个网页。为了选择多个网页，按住 Shift 或者 Ctrl 键不放，然后用鼠标单点各个网页。

然后选用站点管理器窗口的 View 菜单，选用其中的 Show / Hide Link 选项，被选择的文件将立即在站点地图中消失。

2. 显示隐藏文件

为了将隐藏的文件显示出来，以便进行编辑和修改，可选用站点管理器窗口的 View 菜单中的 Show Files Marked as Hidden 选项，隐藏文件将以斜字体显示出来。

3. 取消隐藏属性

为了取消隐藏属性，先选中用斜体字显示的隐藏文件，然后选择 View 菜单的 Show / Hide Link 选项，隐藏属性被取消，斜体字立即转变成正体字。

4. 查看分支文件

如果站点地图中某个文件图标左边有“+”号，说明它的下面还隐藏有文件，用鼠标单击“+”号，隐藏的文件将展开并显示在下面，此时“+”号变成了“-”号，如图 2.10 所示。

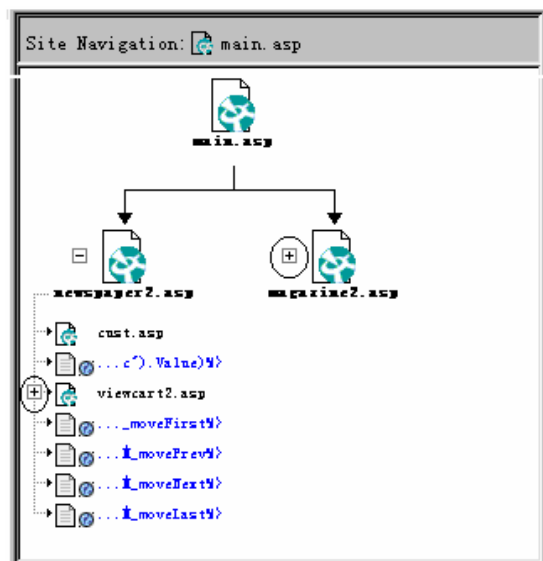


图 2.10 查看分支文件

5. 改变站点地图的根接点

缺省情况下，主页就是站点地图的根节点。但也可以改变根节点的设置，以便更好地观察某个局部。此时用鼠标右击带“+”的文件名，在弹出的菜单中选用 View as Root 选项，窗口内将立即显示以该文件为“根”的文件间的链接结构。

单击上方 Site Navigation 按钮时又可返回以主页为“根”的状态。

6. 保存站点地图

站点地图可以作为一般图形文件保存，然后到图形编辑器中去编辑，此时须选择站点管理器窗口中 File 菜单下的 Save Site Map 选项，在弹出的文件对话框中输入保存的文件名即可。

2.3.4 使用站点地图对网页进行操作

使用站点地图可以直接对网页进行操作，具体内容如下。

1. 增加对网站内其他网页的链接

为了增加对网站内其他网页的链接，先用鼠标单击文件图标，然后选择 Site 菜单下的 Link to Existing File 选项，将弹出文件窗口并在窗口中列出网站内所有文件，选择其中之一后单击

“ OK ” 按钮。源文件就与此文件建立了新的链接关系。

2. 增加对新网页的链接

如果想增加一个新网页并且与网站某个网页链接时,先选择网站内某个网页图标,然后选择 Site 菜单中的 Link to New File 选项,将弹出如图 2.11 所示的对话框。

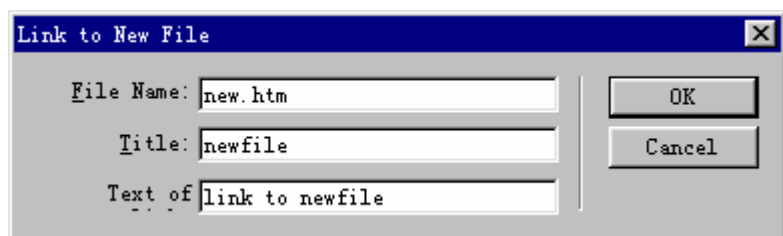


图 2.11 增加对新网页的链接

在对话框中填写有关数据。

- File Name : 为新网页取名。
- Title : 新网页上的标题。
- Text of Link : 链接新网页的字符串。以图 2.11 填写的数据为例,在原来的网页上将出现“ link to newfile ”的链接字符串。如果先选择汉字字符串再打开图 2.11 所示的对话框时,该汉字字符串将成为链接字符串。

3. 改变网页名

为了改变站点中某网页的名字,单击该网页名,该网页名的颜色将呈反衬显示(白底黑字变成黑底白字),此时直接输入新的名字即可。

2.4 浏览器兼容问题

Internet 是一个开放型的系统,它面向的是全世界的浏览器用户,而世界各地的用户使用的浏览器类型和版本都不相同,这里就有一个浏览器兼容问题。网页设计者可以选择一两种浏览器作为目标浏览器,并针对选定的浏览器设计网页,但也要考虑其他类型浏览器访问时的效果,尽量减少不兼容的问题。

在 Dreamweaver UltraDev 中可以设置不同类型的浏览器进行预览,以发现和改进兼容方面的问题。设置的步骤如下。

(1) 单击站点管理器窗口中 Edit 菜单下的 Preferences 选项,打开参数设置对话框,在左边 Category 列表中选择 Preview in Browser 选项,将弹出图 2.12 的对话框。

(2) 单击“ + ”按钮,在弹出的 Add Browse(添加浏览器) 对话框中输入浏览器名和路径,在 Default(默认) 栏中选择 Primary Browse(主浏览器) 或 Secondary Browse(辅助浏览器), 如图 2.12 所示。

(3) 在 Defaults 项中,选择主浏览器或辅助浏览器。

■ 选择 Primary Browse(主浏览器) 项设置主浏览器,当在文档窗口中选择网页后,按 F12 快捷键将进入主浏览器进行浏览。

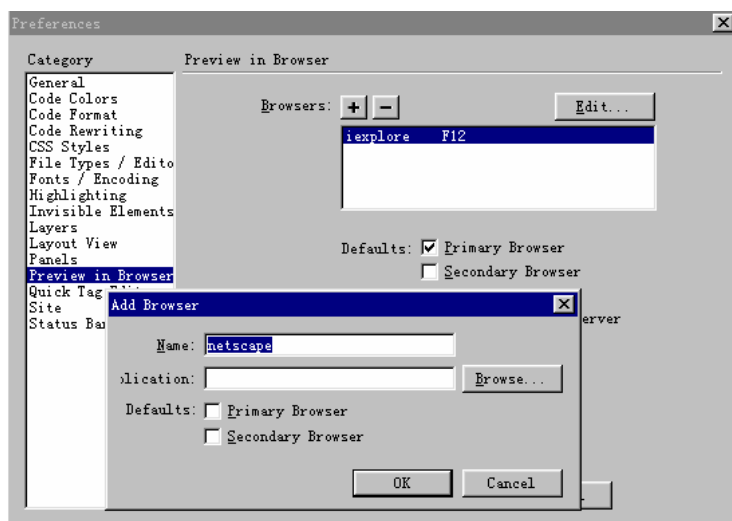


图 2.12 浏览器设置

■ 选择 Secondary Browser(辅助浏览器) 项，将设置辅助浏览器，当在文档窗口预览网页时，按 Ctrl+F12 快捷键进入辅助浏览器进行预览。

(4) 选择 Preview Using Local Server(使用本地服务器预览) 项，允许用户在本地服务器上预览网页。

(5) 单击 OK 按钮，此浏览器设置完成。

(6) 重复以上步骤，添加其他浏览器，最后单击 OK 完成设置。

当完成了以上设置后，在设计网页的过程中，通过 File 菜单选择 Preview in Browser 选项，然后在浏览器列表中选择一种浏览器进行预览，及时发现和解决浏览器之间不兼容的问题。

要想在网页布局、动画、多媒体和交互性等方面实现完全兼容几乎不可能。一般的原则只能是既有重点又照顾一般，重点考虑那些普遍使用的浏览器，在最佳效果与最大兼容之间寻找平衡点。

如有可能，对于一些重要的网页（如主页）最好能够设计多种版本以满足不同浏览器类型的需要。例如设计一个包含框架的网页，就可以在该网页中同时为不支持框架的浏览器设计一个取代的部分；设计一个包含 Flash 动画版本的网页可以同时设计一个不包含 Flash 动画的版本，让用户访问时能够根据自己浏览器的类型自动跳转到适合的版本中去运行。

2.5 文件的上传与同步

2.5.1 向远程网站上传文档

当完成了本地网站的创建工作，准备将文档上传到发布信息的服务器网站时，选择 Site|Defines Sites 菜单项，在弹出的窗口中选择本地网站名并单击 Edit 按钮，将弹出如图 2.13 所示的界面。

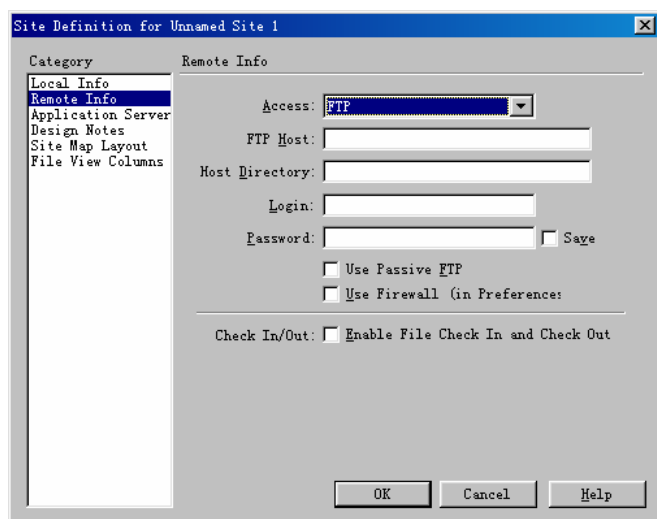


图 2.13 上传文档时定义的窗口

- (1) 在左边的窗口中选择 Remote Info 项。
- (2) 在右边的 Access 下拉菜单中选择 FTP 选项。
- (3) 在 FTP Host 中输入准备上载的 FTP 主机全名(FTP Host)。例如：

ftp.mindspring.com.cn

注意不能采用以下两种格式：

ftp://ftp.mindspring.com.cn

或

mindspring.com.cn

- (4) 在 Host Directory 中填入的目录应该与本地网站的根目录相同，否则将上传到错误的位置，用户将看不到建立的网站，网页之间的链接也会发生问题。

连接之前，远程网站上必须已经存在相应的目录，如果远程网站上还没有对应于本地站点的根目录时，先要求服务器管理员创建一个。

- (5) 在 Login、Password 文本框中输入登录名和口令。选择右边的 Save 项时 Dreamweaver UltraDev 将自动保存口令，否则每次连接时，将要求输入口令。

2.5.2 建立与远程网站的同步关系

一旦你的站点文件上传到远程网站，本地网站就需要经常与远程网站保持同步，以便随时将更新或增添的网页上传到远程网站中去，或者将远程网站中的文件取回到本地站点，及时了解和处理客户反馈回来的信息。

1. 了解文件的变化情况

利用网站管理器的下列菜单项可以查看到最近一次传送后文件的变化情况。其中：

- 菜单 Edit | Select Newer Local 选项可以查看到本地网站中新增添或变化了的文件；
- 菜单 Edit | Select Newer Remote 选项可以查看到远程网站中新增添或变化了的文件。

当选用上述菜单时，新变化的文件名将用深色显示出来。

2. 本地与远程站点之间的文件传送

为使本地站点与远程站点之间的文件取得一致，选择 Site|Synchronize 菜单项，将弹出如图 2.14 所示的对话框。

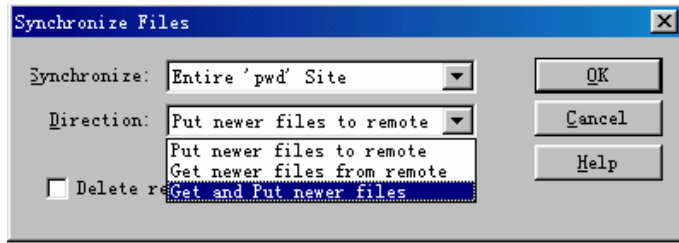


图 2.14 同步窗口

然后按照以下步骤进行。

(1) 如果对整个站点实现同步时，从 Synchronize 小窗口的下拉菜单中选取站点名；如果只同步部分文件时，选用 choose Local Files Only 项。

(2) 在 Direction 小窗口的下拉菜单中，确定文件的传送方向。

- Put newer files to remote：将本地站点中的新文件或更改过的文件上载到远程服务器。
- Get newer files from remote：将远程服务器中的新文件或更改过的文件取回本地站点。
- Get and Put newer files：双向传输。即先比较本地或远程站点中的文件，并将最新版本的文件传向对方。

(3) Delete remote files not on local drive 选项用于选择是否删除那些现在还存在于远程站点但已不存在于本地站点的文件。

(4) 当选择完毕并按“ OK ”按钮时，将弹出如图 2.15 所示的界面，界面中将提示哪些文件将上载（ Put ），哪些文件将取回（ Get ）。如果确定某文件不传送时，取消左边复选框的选择即可。

按“ OK ”按钮，系统将自动完成文件的传送工作。

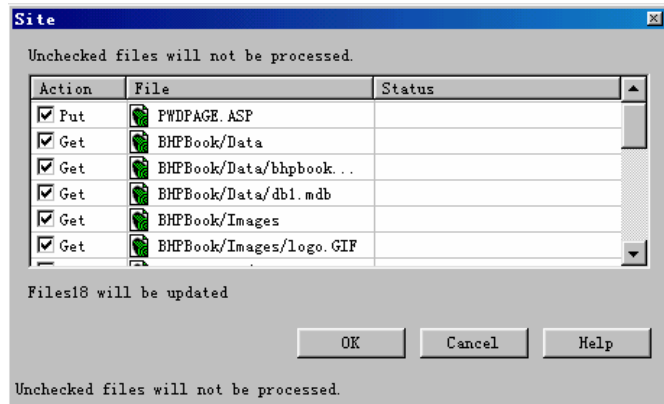


图 2.15 查看需同步的文件窗口

第3章 HTML 文本编辑

3.1 HTML 简介

3.1.1 HTML 语言的结构

HTML(超文本标记语言) 是英文 HyperText Markup Language 的缩写 , 它的作用主要包括 3 个方面 :

- 定义在浏览器上的显示格式 ;
- 建立超级链接 ;
- 集成其他多媒体软件。

HTML 语言目前已经成为各种类型浏览器的通用标准。由于它简单、实用 , 已被广泛应用于 WWW 的各类程序中。

HTML 的语法主要包括元素和标签(标志) 两个方面。

(1) 元素 : 元素有 title(标题) 、IMG(图片) 、table(表格) 等等。元素名不分大小写。

(2) 标签(标志) : 标签(标志) 是用来规定元素的属性和位置的。HTML 中的标签通常成对出现(也有单独出现的标签) 。成对出现时 , 开始的标签是 <元素名> , 结束时的标签是 </元素名> 。单独的标签用 <元素名> 即可。

一个最简单的 HTML 文本格式如下 :

```
<HTML>
  <Head>
    <Title>...</Title>
    ...
  </Head>
  <Body>
    ...
  </Body>
</HTML>
```

HTML 文本是一种树形(层次) 结构。<HTML> 标签是文本的根 , 其他的 HTML 标签全部包括在 <HTML>...</HTML> 以内。<HTML> 下面有两大分支 : <Head>...</Head> 和 <Body> ...</Body> 。其中 <Body>...</Body> 分支为文档的主体 , 主体中的内容将显示在客户端的浏览器中 , <Body> 内又包括若干分支 , 如用 H1、H2 等表示字体 , P、DIV、FORM 等表示块元素等等。在 <Head>...</Head> 段中除 <Title>...</Title> 包括的内容将作为窗口的标题显示在最上方外 , 其余部分主要是关于文档的说明以及某些共用的脚本程序。

在 HTML 中成对出现的标签如下 :

- <HTML>...</HTML> 代表 HTML 文件的开始和结束 ;

- <head>...</head>代表文件头部分的开始和结束；
- <title>...</title>代表网页的标题；
- <body>...</body>代表文件主体部分的开始和结束；
- <h1>...</h1>代表 1 号字体（最大号字体）的开始和结束；
- <h2>...</h2>代表 2 号字体的开始和结束；
- <p>...</p>代表段落的开始和结束；
- ...代表粗体字；
- <i>...</i>代表斜体字。

单独出现的标签如下：

- <p> 段落标签也可作单个标签使用；
- <hr> 代表插入一横线；
-
 中断标记，表示在此处断行。

3.1.2 标签（标志）的属性

在 HTML 中每个标签都有缺省的属性，如果需要改变这些属性时，应将“属性名=值”放在标签名的同一个尖括弧中，例如：

```
<p align = right>本段落将放置在右端</p>
```

其中 p 是段落标签，align 代表“对齐方式”，是属性名。right 代表“右边”，是属性的值。这个标签的含义是将字符串放在浏览器的右端。又如：

```
<hr size=6 width=160 align=center>
```

其中 hr 是水平线标签，size 是水平线的高度，width 是水平线的宽度，align 为对齐属性。这个标签表示增加一条横线，水平线的高度为 6，宽度为 160，显示于浏览器的中间。

3.1.3 注释语句

和其他程序设计语言一样，在 HTML 文本的适当位置上增加注释语句能提高文本的可读性，编译器将不解读注释部分。注释语句的格式如下：

```
<!--注释语句 -->
```

3.2 HTML 编辑窗口的使用方法

Dreamweaver UltraDev 提供了可视化设计网页的工具，设计者可用“所见即所得”的直观方法创建网页，系统在自动将其转化成 HTML 语句后传送到浏览器，从而简化和加快了设计过程。

Dreamweaver UltraDev 4.0 的 HTML 编辑窗口，其中增加了几个页面之间快速转换的按钮。另外，窗口还提供了极强的快速帮助功能，只要将光标放置到某个标签位置上，然后单击窗口上方的  (Reference) 按钮，将立即弹出帮助窗口，对该标签的含义和使用方法加以说明。这个窗口对于设计者来说，既是编写 HTML 语句的窗口，也是学习 HTML 语言的工具。

在站点管理器中创建新文件后，双击文件名即可进入 HTML 编辑窗口，虽然刚进入时，

还只是一个空白网页，但代码窗口却不是空白的，它已经自动将 HTML 语言的框架列出来了，如图 3.1 所示。

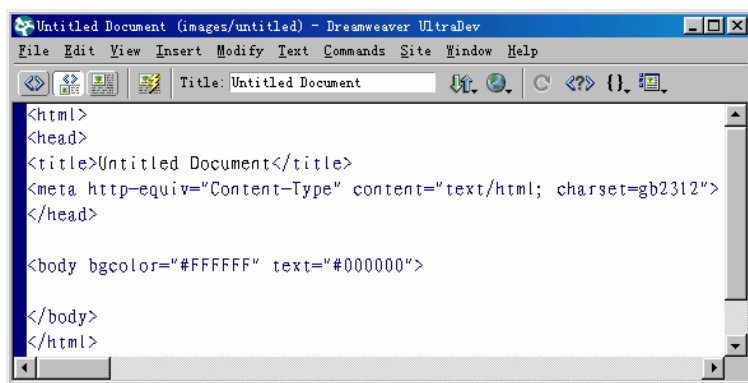


图 3.1 HTML 初始界面

HTML 编辑窗口的右上方有几个按钮，它们的作用说明如下。

-  File Management：文件传递管理（如 Get、Put 等）。
-  Preview/Debug in Browser：在浏览器中预览和调试。
-  Refresh：刷新控制。
-  Reference：快速帮助按钮。
-  Code Navigation；代码浏览。
-  View Options：查看选项。

3.3 关于 HTML 文档中的头<head>

3.3.1 元数据(meta)

元数据(meta)是 HTML 文档头<head>...</head>中的重要组成部分。它不属于文档显示内容而是用来说明文档信息的。在 HTML 4.0 的规范中，允许指定多个能被计算机识别的说明信息。例如：文档的作者、日期和时间，关键字列表，使用的语言格式，刷新间隔等等。

这些信息有些是为了注释文档，也有一些用于通知浏览器执行某些要求（如刷新间隔等）。另外，它还可以为一些搜索引擎提供数据。

有一些大型网站常利用自己的搜索引擎到其他网站中去收集信息，主要就是从各网页的文档头(head)中过滤、归纳相关的<meta>信息，然后在本网站上发布，以丰富本网站的信息量，吸引更多的客户。其结果也帮助被搜索网站做了宣传，给双方都带来好处。为了给这些搜索引擎提供信息，利用网页的 Meta 标签，用鲜明、简练的语句来说明自己网页的特色是十分必要的。如果这方面的说明写得太多，或者太平淡，搜索引擎可能不得不放弃对该网页的信息收集工作。

Dreamweaver UltraDev 4 中为书写这些内容提供了服务。在 Object 组件板的 head 页（如图 3.2 所示）中有以下图标可供利用。



图 3.2 head 页面

 Insert Meta：用于在网页中输入 Meta 标签。

 Insert Keywords：为文档提供关键字列表。

 Insert Description：为文档提供简要说明。

 Insert Refresh：确定浏览器自动刷新的间隔。

 Insert Base：确定本网页的基址(URL)，其他网页都可以它为标准确定相对地址。

 Insert Link：用来定义其他文件与当前文件之间的关系，这种关系将保存在文件头中，它与在<body>...</body>之间定义的链接是不同的。

3.3.2 增添 meta 属性的方法

下面通过两个例子来说明增添 meta 属性的方法。

1. 在文件头中插入 Keywords

(1) 先打开某一网页，然后单击对象窗口 head 页中的 Insert Keywords 图标，将弹出如图 3.3 所示的对话框。

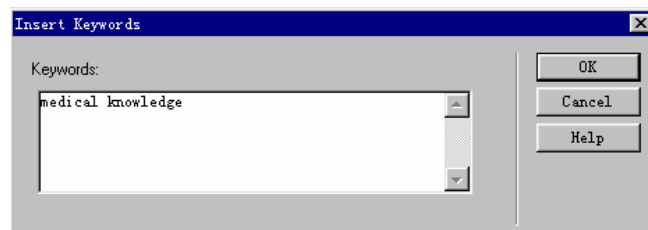


图 3.3 插入 Keywords

(2) 在 Keywords 文本框中填入关键字，如 medical knowledg(医药技术咨询)，然后单击“ OK ”按钮。

(3) 网页的源代码中将出现以下字段：

```
<head>
...
<meta name="keywords" content="medical knowledge">
</head>
```

2. 确定刷新间隔

指定一定时间刷新，可以使网页定时跟踪最新数据，这对于内容变化频繁的网页来说很有必要。方法如下。

(1) 先打开某一网页，然后单击对象窗口 head 页中的 Insert Refresh 图标，将弹出如图 3.4 所示的对话框。

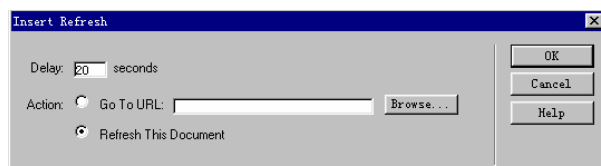


图 3.4 刷新时间间隔的设置

(2) 在 Delay 文本框中指定刷新的时间间隔(本例中设定的间隔是 20 秒)。

(3) 在 Action 中有两个选择。

- 选中 Go To URL 并在其文本框中填写新网页的 URL，到刷新时间间隔时将转向新的网页。

- 选中 Refresh This Document 项(通常选用此选项)，则到刷新时间间隔将只更新内容仍然停留在本页。

若按图 3.4 的设定，网页的源代码中将出现以下代码：

```
<head>
<meta http-equiv="refresh" content="20">
</head>
```

3.4 编辑 HTML 文本

3.4.1 属性面板

应充分利用属性面板对输入的 HTML 文本进行编辑。如果看不到属性面板时，可利用菜单的 Window|Properties 选项打开它。HTML 的属性面板如图 3.5 所示。

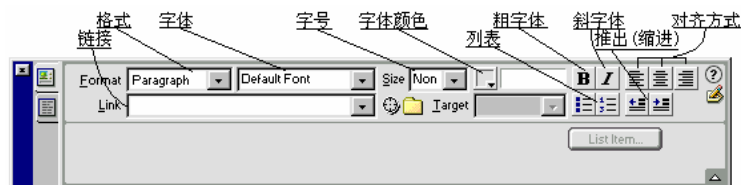


图 3.5 HTML 属性面板

确定文本的格式、字体、字号、字体颜色时，均可在属性面板的各自下拉菜单中选择。先选择好相应的字符串，然后单击粗字体、斜字体、缩进、对齐或列表按钮，就能改变已选字符串的显示属性。

3.4.2 插入文本元素

下面说明几个常用的文本元素的插入方法。

(1) 插入空格：直接按键盘上的“空格键”时，并不能像通常那样移动窗口中的字符。一个简便的方法是先进入汉字的“标准”输入法的“全角”状态，然后再按“空格键”。

(2) 插入横杠：先将光标放置于需要插入横杠的位置，然后单击 Object(对象) 面板 Common 页中的  图标，即可在窗口中增添一横杠。

(3) 插入日期：先将光标放置于需要插入日期的位置，然后单击 Object(对象) 面板 Common 页中的  图标，将弹出一日期格式 (Date Format) 选择窗口，窗口中列出了各种显示格式，如 March 7, 1974、03/7/74、1974-03-07 等，从中选择一种后，即可在该位置上插入当天的日期。

(4) 插入 E-mail 地址：先将光标放置于需要插入 E-mail 的位置，然后单击 Object(对象) 面板 Common 页中的  图标，将弹出如图 3.6 所示的对话框。

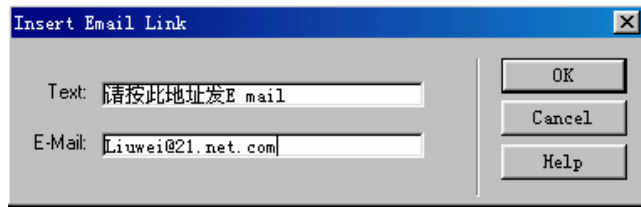


图 3.6 插入 E-mail 地址

在 Text 文本框中填入显示的字符串，在 E-mail 文本框中填入 E-mail 地址。

(5) 插入注释：先指定字符串，然后单击 Object(对象) 面板 Invisibles 页中的  图标，将弹出如图 3.7 所示的 Insert Comment 窗口。

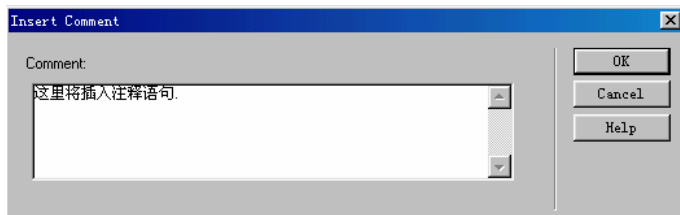


图 3.7 插入注释语句

在窗口中填入注释语句后单击“OK”按钮，字符串后面将出现一标记，同时这些注释语句已经按照 HTML 语法的要求自动放置到源代码中。

(6) 插入特殊字符：当需要插入某些特殊字符时，先将光标放置于插入的位置，然后选择 Object(对象) 面板的 Characters 页，将弹出如图 3.8 所示的特殊字符列表。

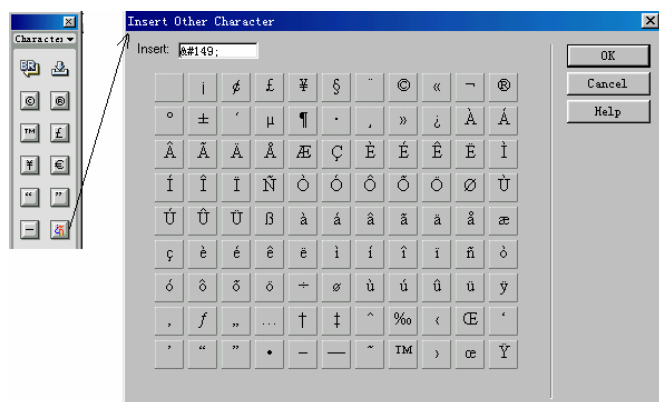


图 3.8 特殊字符列表

按动右下角的图标时将弹出更多的特殊字符，选择其中之一时，上方 Insert 小窗口中将显示出相应的代码。

3.5 文本超级链接

“超级链接”是 HTML 语言的精华。通过超级链接可以随时转向其他页面或者到某个段落去查看想要看的东西，还可以跨越站点，到其他站点上去查阅相关信息。你可能一会儿在亚洲，一会儿到欧洲、美洲等全世界各地浏览最新信息，而这一切只需坐在屏幕前点一点鼠标就能实现。

3.5.1 建立页面之间的超级链接

1. 建立站点内页面之间的超级链接

当被链接的文件与主动链接的文件在同一站点目录下时，HTML 的链接语句格式是：

```
<A HREF="文件名">显示的文本</A>
```

Dreamweaver UltraDev 提供了简化站点内页面之间的链接方法。先选择一段文本，然后从属性面板中，用鼠标从“”符号拖到站点管理器中需链接的页面文件即可。此时 Link 小窗口中将出现被链接的文件名。

2. 建立与站点以外页面的链接

当需要与站点以外的页面建立超级链接时，HTML 的语句格式是：

```
<A HREF="被链接文件的 URL">显示的文本</A>
```

3.5.2 建立与页面内部某段落的超级链接

当需建立与某页面内部的某个段落（又称“锚点”）的超级链接时，HTML 的语句格式是：

```
<A HREF="#锚点名">显示的文本</A>
```

或

```
<A HREF="页名.Asp#锚点名">显示的文本</A>
```

利用页面之间的超级链接时，将直接跳到另一页的起始部分，如果想要跳到页面内部的某

个字段时可预先设置“锚”来实现，具体方法如下。

(1) 将“锚”对象窗口 invisible 页中图标) 放置到某字段的前面，并在弹出的界面中填入锚的名字，如图 3.9 所示的 mark。

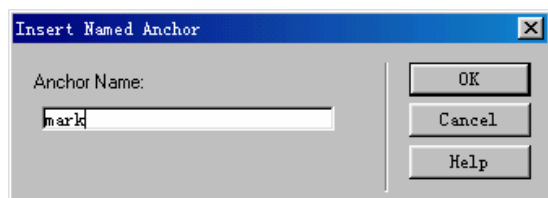


图 3.9 输入锚名

(2) 选择超级链接字段，并在属性窗口的“连接”(Link)属性中填入“#锚点名”(锚就在本页时)、“页名.Asp#锚名”或“页名.htm#锚名”(当锚在其他页面时)。

例如：

#mark

或

store.asp#mark2

store.asp 代表网页名。

3.5.3 利用链接地图快速建立超级链接

利用网站管理器的链接地图 (Map) 可以直观、形象地看到网站内各网页之间的链接情况，还可以通过 Map 快速建立新的链接。

单击按钮时网站管理器窗口中将显示链接地图。第一次单击此按钮时可能会提示要求确定主页。在文件列表中选择一文本后按鼠标右键，在弹出的菜单中选择 Set as Home Page 项，该页就成为链接中的主页了。

显示的方法有两种选择：一种是只显示地图 (Map Only)，另一种是同时显示地图与文本列表 (Map and Files)。图 3.10 是在左右两个框体中分别显示两者的情况。

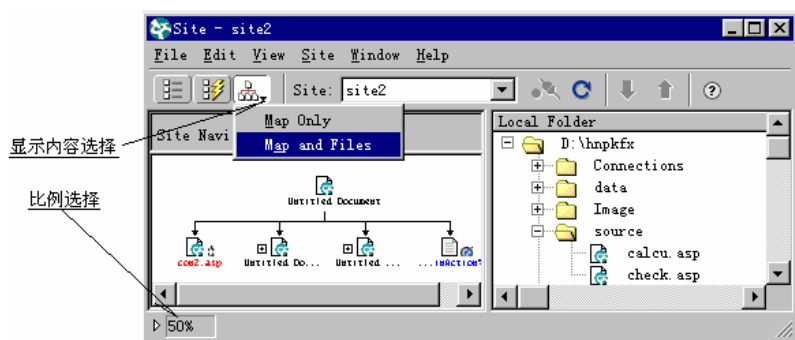


图 3.10 Map 链接地图

Map 显示的比例还可通过左下方的比例按钮选择。

通过 Map 地图快速建立链接的方法是：用鼠标将某个文档名拖向 Map 的某个接点，就建

立了该文档与接点之间的链接。在 Map 地图内部也可以用鼠标拖动的办法建立接点之间的链接。

3.6 净化 HTML 文本

HTML 文本编辑器能自动生成 HTML 文本，这为程序设计者带来极大的方便，但同时也会为文件带来了一些“冗余”(即生成一些当前文档用不上的标记)，这不仅增加了文件的容量、增长了传输和下载时间，也增加了出错的“几率”。为减小文件的容量、减少错误的可能，Dreamweaver 提供了净化 HTML 文本的功能。

(1) 选择主菜单的 Edit(编辑)|Preference(参数选择)|Code Rewriting(代码改写)项，将弹出如图 3.11 所示的界面。

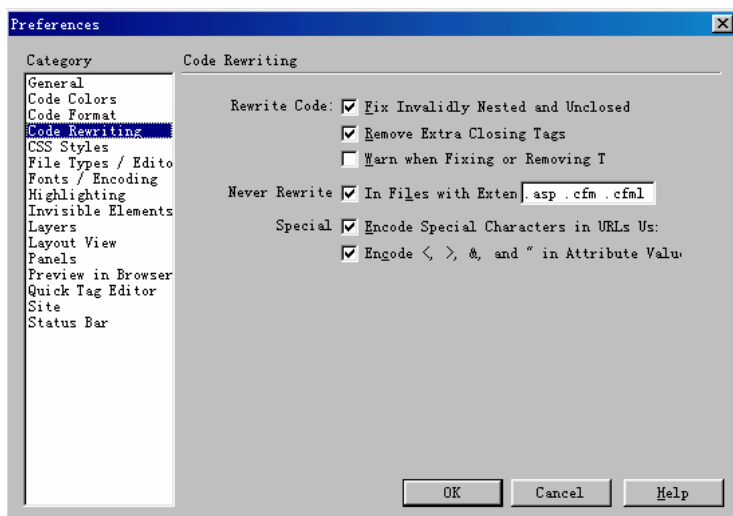


图 3.11 净化 HTML 文本选项

(2) 选择修改的项目后按“OK”按钮。

另外也可选择主菜单中的 Command(命令)|Clean Up HTML(清理 HTML)项，系统将提供选项界面并根据在该界面上的选择对 HTML 文本进行清理和净化。

3.7 字符串的移动显示

下面介绍一种将静态字符串变成为移动的字符串的方法，就是将<marquee>与</marquee>一对标签用手工的方法加在需要移动的字符串的两端。例如：在<h2><i>移动字符串</i></h2>的前后加上<marquee>...</marquee>标签后，该字符串变成了：

<marquee><h2><i> 移动字符串 </i></h2></marquee>

当进入浏览器时，“移动字符串”这 5 个字将从右向左移动，到达左边沿后再快速返回到

右边，继续向左移动，不断循环直到关闭该网页时为止。

3.8 综合示例

下面是一个简单 HTML 文本示例，对照一下它在浏览器上的显示以及代码情况。

(1) 浏览器显示的界面如图 3.12 所示。

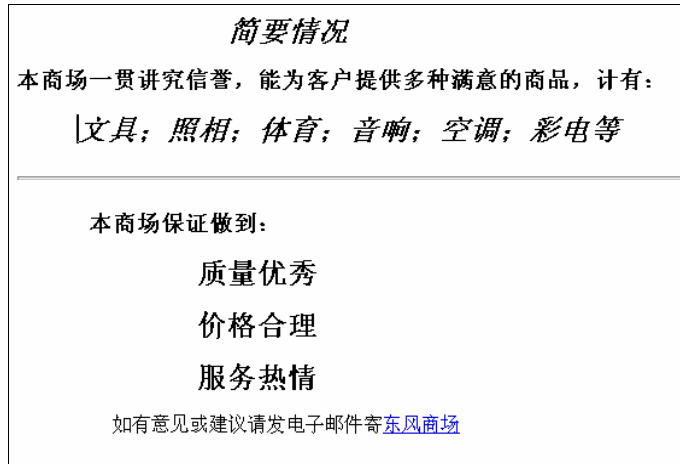


图 3.12 HTML 文本示例

(2) 相应的 HTML 代码如下。

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<body bgcolor="#FFFFFF" text="#000000">
<h2>          <i>简要情况</i></h2>
<h3>本商场一贯讲究信誉，能为客户提供多种满意的商品，计有：</h3>
<marquee> <h2><i>    文具；照相；体育；音响；空调；彩电等</i></h2> </marquee>
<hr>
<h3>    本商场保证做到：</h3>
<h2>        质量优秀</h2>
<h2>        价格合理</h2>
<h2>        服务热情</h2>
<p>如有意见或建议请发电子邮件寄<a href="mailto:dfsc@21.net.com">东风商场</a></p>
</body>
</html>
```

由于有<marquee>与</marquee>一对标签放在“ <h2><i>文具；照相；体育；音响；空调；彩电等</i></h2> ” 的两端，因此字符串“ 文具；照相；体育；音响；空调；彩电等 ” 将是一个从右向左不断循环移动的字符串。

第 4 章 表格与网页布局

4.1 概述

表格是生活中常用于组织数据的形式，如通讯录、课程表、列车时刻表等大都采用表格的形式。将数据或图形放在表格中的好处是可以使数据和图形显示得更有规律、更有利于对照和比较，因此在网页中常用表格组织数据。除此以外，网页中的表格还可用来给元素定位，帮助进行网页的整体布局。

一张表格由行（rows）、列（columns）、单元格（cells）3 部分组成。创建表格实际上就是创建表格的行、列和单元格。在 HTML 语句中：

创建表格的语句是<TABLE>...</TABLE>；

创建行的语句是<TR>...</TR>；

创建栏名的语句是<TH>...</TH>；

创建单元格的语句是<TD>...</TD>。

例如：

```
<TABLE border="3" cellpadding="6" cellspacing="3">
  <caption align="top">
    <font size="5">一次特快列车时刻表</font>
  </caption>
  <tr><th>站名</th>  <th>到站时刻</th>  <th>开车时刻</th> </tr>
  <tr><td>北京西</td>  <td>- </td>          <td>16.0 </td>  </tr>
  <tr><td>石家庄</td>  <td>18.56</td>        <td>9.08 </td>  </tr>
  <tr><td>郑州</td>    <td>0.01</td>          <td>0.14 </td>  </tr>
  <tr><td>武昌</td>    <td>6.42</td>          <td>6.54 </td>  </tr>
  <tr><td>岳阳</td>    <td>9.29 </td>          <td>9.35 </td>  </tr>
  <tr><td>长沙</td>    <td>11.20 </td>         <td>--- </td>  </tr>
</TABLE>
```

上述 HTML 语句对应的表格如图 4.1 所示。

在语句<TABLE border="3" cellpadding="6" cellspacing="3">中：

- border 代表表格边框的宽度；
- cellpadding 代表单元格中内容与边界之间的空间；
- cellspacing 代表单元格之间的间距。

一次特快列车时刻表

站名	到站时刻	开车时刻
北京西	-	16.0
石家庄	18.56	9.08
郑州	0.01	0.14
武昌	6.42	6.54
岳阳	9.29	9.35
长沙	11.20	---

图 4.1 表格实例

4.2 创建表格

Dreamweaver UltraDev 系统用可视化的方法简化了对表格的创建操作。通过单击鼠标能够创建出各类复杂的表格来。具体步骤如下。

(1) 选用下列方法之一开始创建表格。

- 将光标放置在表格将要出现的位置上,选择 Object 窗口的 Common 页,然后单击 Table 按钮。

- 选择 Insert|Table 菜单项。

- 将 Insert Table 图标从 Object 面板的 Common 页中拖到页面相应的位置。

(2) 在弹出的对话框中确定表格的参数。

- Rows(行)代表表格的行数。

- Columns(列)代表表格的列数。

- Cell Padding(单元格填充)代表单元格中内容与边界之间的空间。

- Cell Spacing(单元格间距)代表单元格之间的间距。

- Width(表宽)代表以像素(pixels)或以浏览器窗口的百分比(%)为单位指定表格的宽度。

- Border(边宽)代表表格边框的宽度,当不需显示边框时,输入 0。

(3) 单击“OK”按钮,即完成表格框架的创建工作。

4.3 利用模板选择表格样式

利用模版选择表格的样式,利用系统提供的样式可减少设计时间。方法如下。

(1) 先按照 4.2 节讲述的方法建立表格的框架,然后选用菜单 Commands|Format Table 项,将弹出如图 4.2 所示的对话框。

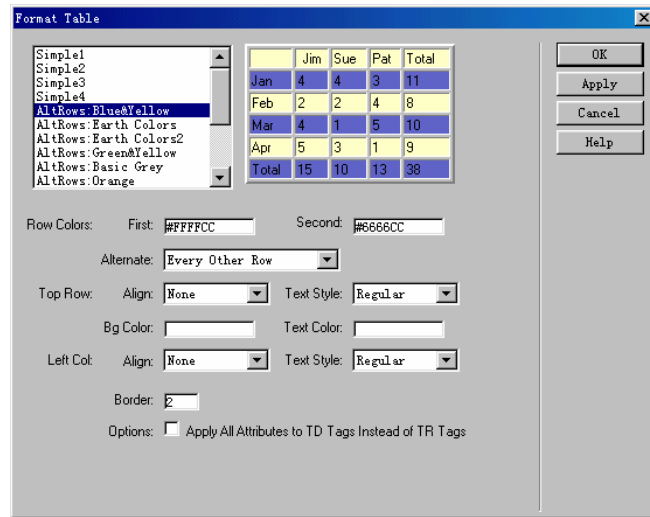


图 4.2 表格模板

(2) 在左边的列表中选择样式的类型，在右边观察显示结果。

(3) 可利用对话框下面的选项对已有的方案作进一步的修改。

在 First、Second 文本框中可以设定第一行和第二行的颜色。

交替的方式在 Alternate 的下拉菜单中可以选择颜色交替的方式，交替的颜色就是第一行和第二行的设置，该下拉菜单提供了以下几种选择。

- <do not alternate>：颜色不交替。
- Every Other Row：隔行颜色交替。
- Every Two Rows：隔两行颜色交替。
- Every Three Rows：隔三行颜色交替。
- Every Four Rows：隔四行颜色交替。

可以在 Align、Text Style 下拉列表中选择第一行的对齐方式和字符的风格，在 Bg Color、Text Color 文本框中设置底色和字符的颜色。

可在 Align、Text Style 下拉列表中选择它的对齐方式和字符风格。

表格边框的宽度在 Border 文本框中设置。

通常情况下，以上的设置主要是针对表格的行。如果要将设计方案优先应用于表格的单元格时，可选用 Apply All Attributes to TD Tags Instead of TR Tags（应用所有的属性于 TD 标记以替代 TR 标记）项。

4.4 行或列的合并与拆分

表格的两行（列）或多行（列）之间既可以合并又可以拆分。方法是：先选择需合并或拆分的行或列，然后按属性窗口中的“合并”或“拆分”按钮。合并与拆分按钮的位置如图 4.3 所示。当按拆分按钮时，将弹出对话框提问分成多少行或列，按照需要填写即可。

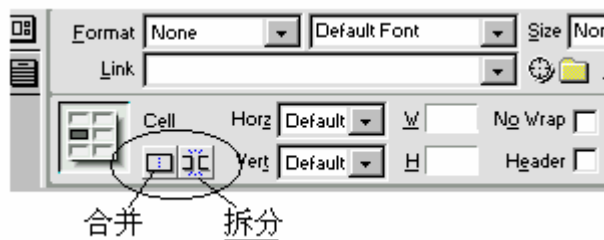


图 4.3 表格行列的合并及拆分按钮

4.5 表格的调整和修改

4.5.1 添加或删除表格的行和列

1. 表格中添加行或列

(1) 在需添加表格的行(列)中,单击其上方插入行的任意单元格,然后选择菜单 Modify|Table, 将弹出如图 4.4 所示的选择菜单。

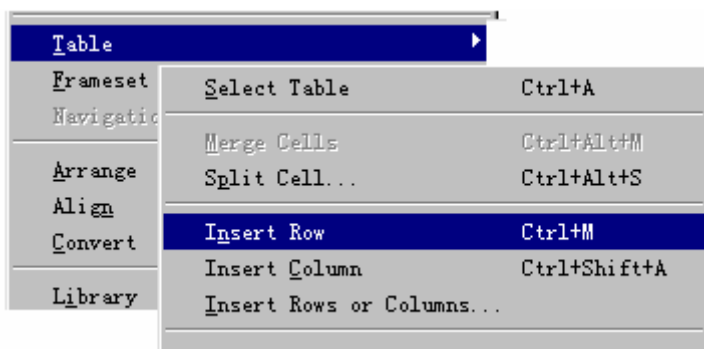


图 4.4 为表格增添行(列)数

(2) 选择 Insert Row, 将给表格增加新行;选择 Insert Column, 将给表格增加新列;如果选择 Insert Rows or Columns, 将弹出如图 4.5 所示的对话框。

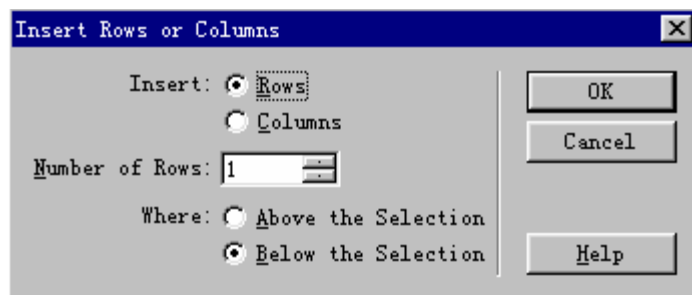


图 4.5 确定增加的行(列)数以及放置的位置

在这个窗口中可以作多项选择。如，是增加行(Rows)还是列(Columns)；一次增加几行(或几列)，把需增加的行或列数填入 Number of Rows 即可；还可以确定增加的行、列是放在已经选好的行(列)的前面还是后面。

2. 表格中删除行或列

选用下列方法之一删除行或列。

- 选中某行或某列，然后按 Delete 键。
- 选中某行(列)，然后选择菜单 Modify|Table|Delete Row 或 Delete Column 项即可。

4.5.2 表格嵌套

可以将一个新表放到原来的表格中，在表格之间形成嵌套。方法如下。

(1) 单击一单元格，然后选择菜单 Insert|Table 项。

(2) 将 Insert Table 图标从 Object 窗口的 Common 页中拖到某一单元格中。

被嵌套的表格就变成了外层表格的一部分，将跟随外表一起放大、缩小和移动。

4.6 表格内容排序

可以根据一列或两列的内容进行排序，步骤如下。

(1) 选中表格，然后选择菜单 Command | Sort Table 项。将弹出如图 4.6 所示的对话框。

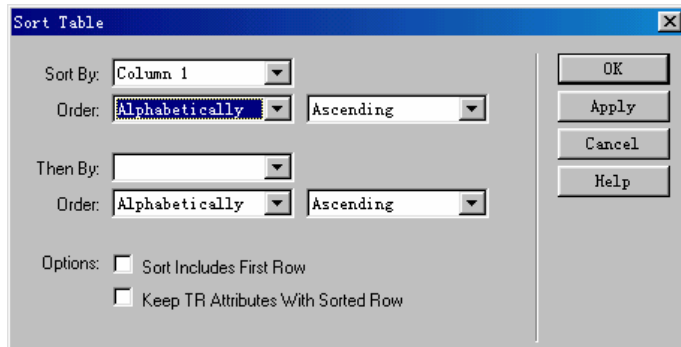


图 4.6 表格排序参数设定

(2) 在对话框中选择参数。

Sort By：在下拉列表中选择按照哪一列的内容进行排序。

Order：在两个下拉列表中选择排序的方式。

- Alphabetically：按字母排序。这种排序方法是按位排序。按升序要求时如 1、10、11、2、20 等。
- Numerically：按数字排序。即按照数字的大小排序。当按升序要求排序时如 1、2、10、11、20 等。
- Ascending：按照升序要求排列。
- Descending：按照降序要求排列。

Then By：选择第二级排序字段。当按照 Sort By 选择的字段排序中有多行相同的排序字段时，这些字段再按照第二级排序字段进行排序。

Sort Includes First Row：排序是否包括第一行。通常第一行是标题，因此此项通常不选。

由于属性 Keep TR Attributes With Sorted Row 涉及的面比较宽，很难解释清楚，一般可以不选择此项，故这里省略不作介绍。

4.7 网页的整体布局

网页的整体布局，特别是主页的整体布局，给人带来的是整体视觉上的感受，因此显得特别重要。

传统的方法是利用表格的定位、对齐等功能来对网页进行布局。首先将网页划分成几个部分，以决定表格的行数和列数，然后创建表格并在表格的各单元格中放置各种网页元素。虽然对表格的大小随时可以进行调整，行列之间也可以拆分和归并，但是在布局设计中仍然常有顾此失彼的感觉。

Dreamweaver UltraDev 4.0 新提供的布局表格和布局单元，可以允许用户以局部到整体的方式进行布局，设计的过程更加灵活、更加符合用户的思路 and 习惯。虽然用这种方式布局的最后仍然是形成表格，但这是由系统自动进行的转换，设计者根本不需要考虑几行或者几列的问题。

4.7.1 布局视图与标准视图

Dreamweaver UltraDev 4.0 的对象面板的下方增加了 4 个图标，可用于对网页的布局工作。4 个图标的情况如图 4.7 所示。

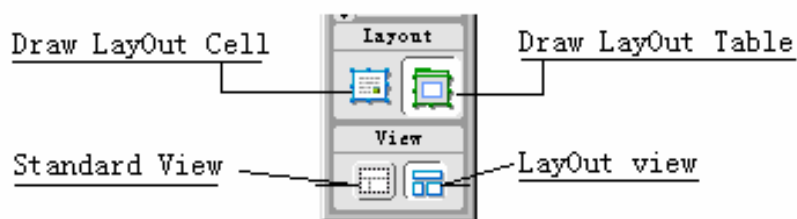


图 4.7 网页布局图标

其中标准视图(Standard View)与布局视图(LayOut View)是一对可以相互切换的互斥的图标，使用标准视图时对象面板上的表格、层等图标都处于有效状态，使用布局视图时上述图标则处于无效状态。

单元表(Draw LayOut Table)与单元格(Draw LayOut Cell)都只有使用布局视图时才有效。

4.7.2 使用单元表

单元表是单元格的容器，只有单元格中才能放置网页元素。

单击 Draw LayOut Table 图标，然后将鼠标移到网页中某个起点上，此时光标变成“+”字形状，拖动鼠标，即可生成一个个 Layout Table，如图 4.8 所示。

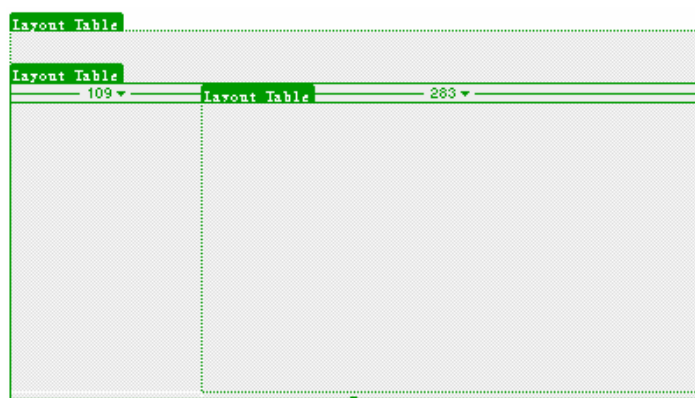


图 4.8 使用单元表布局

可以按照计划用这种办法将网页划分成几个区域。例如可以将上面的区域作为标题，左边区域作为目录列表，右下方区域放入各目录对应的详细内容等等。

4.7.3 使用单元格

单击 Draw LayOut Cell 后可在每个单元表中放置一个或多个单元格，如图 4.9 所示。

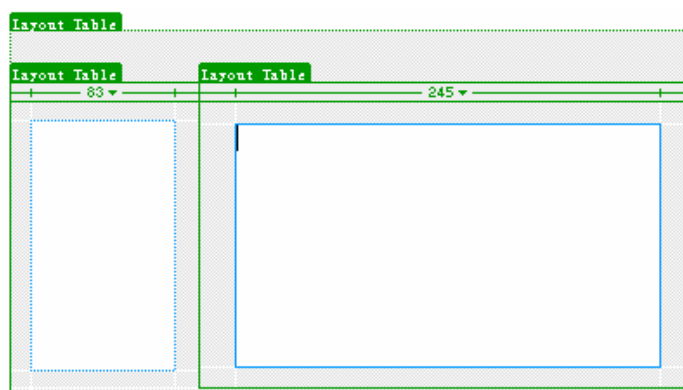


图 4.9 使用单元格布局

然后在单元格中放置网页元素。单元格的边界不能超出单元表的边界，而且最好不要靠得太近，因为单元表的边界有“吸附”作用，当两者靠得太近时会自动地重合到一起。这样做的好处是一个单元格的变化（例如大小的变化）不会波及到单元表以外的其他元素。

单元格也可以单独放置，即不通过单元表而直接放置单元格，此时系统会自动生成相应的单元表。

第5章 插入和编辑图像

Dreamweaver UltraDev 是一个强大的网站管理和网页编辑工具，但它本身不具备图像编辑功能。Fireworks 软件作为“梦幻组合”的成员，与 Dreamweaver UltraDev 紧密结合，提供了这一功能。

本章的前半部分将重点介绍利用 Fireworks 创建和优化图像的方法，后半部分重点介绍通过图像建立链接的方法。

5.1 图像的类型

并不是所有类型的图像都适合于网页的应用，有的图像虽然很漂亮，但由于容量大，网上传输和下载的时间长，因此这样的图像就不适合网页的需要。

另外，网页中增添图像的主要目的是为了使网页变得更加生动、直观和更具有吸引力，对图像本身精细程度的要求并不高。

考虑到以上这些特点，PNG、GIF、JPEG 等类型的图像最适合于网页上使用。这些图像的共同特点是，具有一定的清晰度且压缩比大、容量小，网上传输和下载的时间短。

这些图像的特性如下。

- PNG (Portable Network Graphic) 类型的图像是 Web 图像中的通用格式，也是 Fireworks 软件的基本格式。它用一种无损压缩的方法，最多可支持 32 位颜色，但它不支持动画，如果没有相应的插件，有的浏览器可能不支持这种格式。

- GIF (Graphics Interchange Format) 类型的图像是 Web 中最常用的图像格式，它是一种压缩的 8 位图像文件，最多可支持 256 种颜色，文件很小，适合于存储线条、图标以及卡通和其他大色块图像。GIF 图像还有一个突出的特点，就是它支持动态图、透明图和交织图。

- JPEG (Joint Photographic Experts Group) 是 Web 中仅次于 GIF 的常用图像格式，它是一种压缩得非常紧凑的格式，专用于存储不含大色块的图像。JPEG 图像有一定的失真度，但通常用肉眼不容易分辨出来。JPEG 文件最小只有 GIF 文件的 1/4。JPEG 对图标或含大色块的图像不很有效，不支持透明图和动态图。

5.2 指定图像编辑器

5.2.1 Fireworks 4.0 的作用

Fireworks 是一种与 Dreamweaver UltraDev 紧密结合，用来设计、制作并且优化 Web 图像的编辑器，在这个编辑器中可以用最少的步骤生成容量小、质量高的各种格式的图像，并且能

够给图像赋予动态、交互的特性。

Fireworks 将位图和矢量编辑工具相结合，可以随时对图像进行编辑，还可以用批处理方式工作。

Fireworks 是为适应网页需要而推出来的一套图形解决方案，因此它非常重视图像文件的容量和屏幕输出等因素，力求在它们之间找到最佳效果的平衡。这一点与其他传统传媒的要求不同，对于一些广告印刷品来说，也许 Fireworks 并不是处理图像的最佳选择。

Fireworks 从各种图形编辑软件中吸取优点。例如，它在处理图像的思路、技巧、工具和快捷键等方面吸取了 Photoshop 的特长，还集中了 Flash 中的图符和库面板、Dreamweaver 的快速启动栏等优点。

本书不准备对 Fireworks 作详细的介绍，只是结合 Dreamweaver UltraDev 网页设计的要求，通过几个实际例子来说明它的应用。

5.2.2 建立 Dreamweaver UltraDev 与 Fireworks 之间的调用关系

为了调用 Fireworks 软件编辑图像，必须安装 Fireworks 软件并建立 Dreamweaver UltraDev 与 Fireworks 软件之间的联系。建立联系的方法是，在 Dreamweaver UltraDev 界面中选择菜单 Edit|Preferences 项，将弹出图 5.1 所示的对话框。

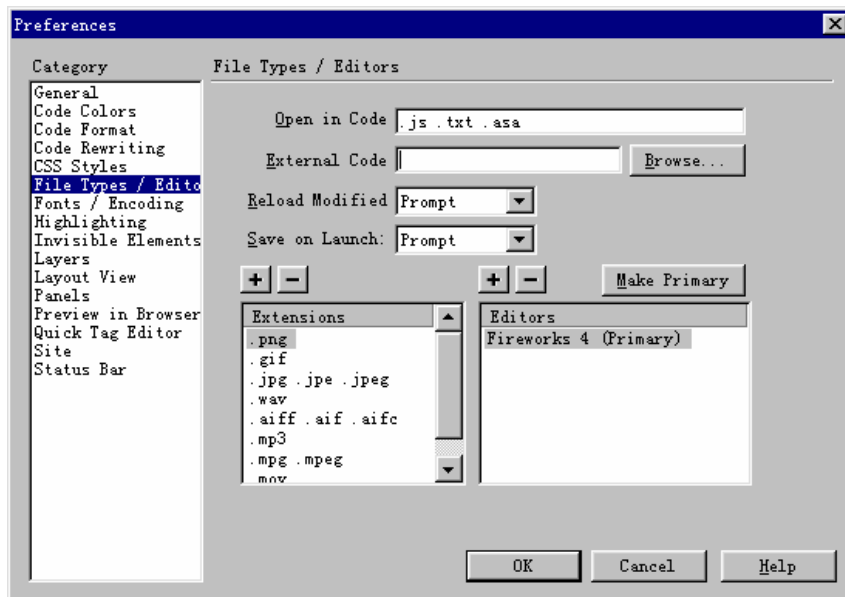


图 5.1 图形编辑器的设定

在左边窗口中选择 File Types / Editor 项，然后在 Extensions 列表框中选择“.png”并单击 Editors 列表框上方的“+”号，在下面的列表框中选择 Fireworks 4”，并单击“Make Primary”按钮。这些操作的目的是指定.png 类型的图形首选的外部编辑器为 Fireworks 4。

用同样的办法将“.gif”、“.jpg .jpe .jpeg”等文件的外部编辑器都指定为 Firworks 4。

5.3 插入和编辑图像

5.3.1 插入图像

先在网页中用鼠标定位图像插入的位置,然后单击 对象窗口 “Common” 页中的 Insert Image 图标,或者选择菜单 Insert|Image 项,都将弹出文件目录窗口,以便从中选择图像文件;若选中的图像不在当前网站的目录下时,系统将询问是否将图像拷贝到网站的目录下。此时单击“OK”按钮,然后选择本站点下的目录,该图像将自动拷贝过来。

5.3.2 为图像设置属性

可在图像属性对话框中设置图像的属性,如图 5.2 所示。

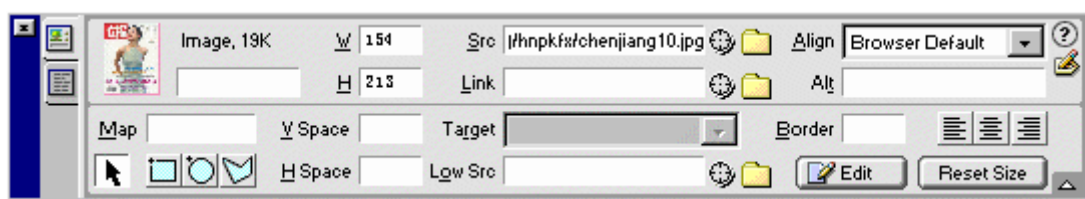


图 5.2 图像属性窗口

图像属性窗口中包括的属性如下。

- 在 W、H 文本框中可分别设置图像的宽和高,也可用鼠标直接拖动图像的边框来改变。
- 在 Src 文本框中可设定图形的名称(含路径)。
- 在 Alt 文本框中可设定当浏览器不支持图像时,代替显示的字符串。这个属性其实对于能够显示图像的浏览器也有用处,当浏览器需要下载比较多的图像,或者图像较大、需要下载的时间较长时,在图像还没有显示之前会暂时用 Alt 属性中的文字代替,这样,客户即使还没有完全看到图像也可了解一些情况,以便及时开始下一步的操作。
- 在 V Space 和 H Space 文本框中可设定图形与边框在垂直方向和水平方向的间距。
- 在 Border 文本框中可设定边框的宽度。
-  代表图像在浏览器中的位置,分别代表左边对齐、居中、右边对齐。
- Edit 按钮的作用是启动图形的外部编辑器。
- Reset Size 按钮的作用是使图像恢复原来的大小。

通过属性窗口还能为图像建立链接,此问题将在后面介绍。

当需要对图像进行编辑时,单击 Edit 按钮,系统将根据预先的设定,自动转入外部图形编辑器(如 Fireworks 4.0 工具软件等)的环境中。

[说明]:有时,单击 Edit 按钮后,系统会提示,图像被“锁”(Lock),不能进行编辑。此时应先在“站点管理器”中为图像文件解锁(Unlock)再转入 Fireworks。方法是先查看该图像文件,若文件名左边有一“锁”图标时,说明该图像被锁(只读),此时可用鼠标右击图像名,在弹出的菜单中选择“除去只读”(Turn off Read Only)项,“锁”图标消失。

5.3.3 利用 Fireworks 编辑图像

Fireworks 是一个功能强大的图形编辑软件，这里只介绍相关部分，主要介绍在图像上增添和美化字符串的方法。以下操作都是在 Fireworks 界面中完成的。

Fireworks 4.0 的设计界面如图 5.3 所示。

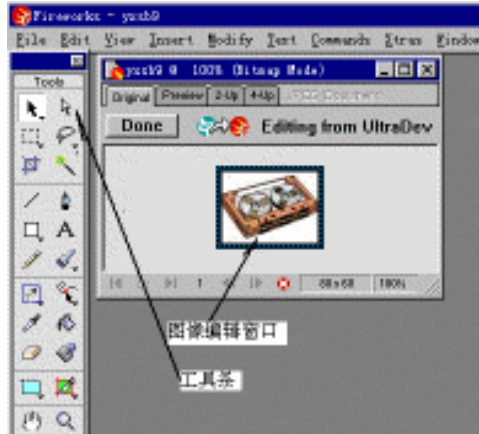


图 5.3 Fireworks 4.0 的界面

在这个界面中利用菜单或工具条上的工具，可以对图像进行编辑和优化工作，完成后单击上方的“Done”按钮，即可返回 Dreamweaver UltraDev 的设计环境。

1. 设置图像或画布

选择菜单 Modify/Image Size 项，在弹出的窗口中设置图像的大小；选择 Canvas Size 项在弹出的窗口中设置画布的大小；选择 Canvas Color 项设置画布的底色；选择 Rotate Canvas 旋转画布的方向。

2. 输入字符串

先单击工具图标“A”，然后到画布上拖动鼠标形成一个矩形，此矩形即字符串放置的位置，此时将弹出文本编辑器（Text Editor），如图 5.4 所示。

在字符串编辑器中先设字号、字符颜色及字体风格等项，再输入字符串。



图 5.4 文本编辑器

3. 编辑和美化字符串

编辑和美化字符串的方法如下。

(1) 颜色的改变。通常对字符串颜色的设置比较简单，只能设成单一颜色。而在这里的设置却是丰富多彩，它可以将字符串设置成混合颜色，如：红—绿—黄、红—绿—蓝等，还可以将颜色的显示由浅逐步加深呈线性 (linear) 变化，也可以呈由深变浅再变深的波浪型 (wave) 变化。利用菜单 window | Fill 项弹出的 Fill 对话框可以设置上述参数，如图 5.5 所示。

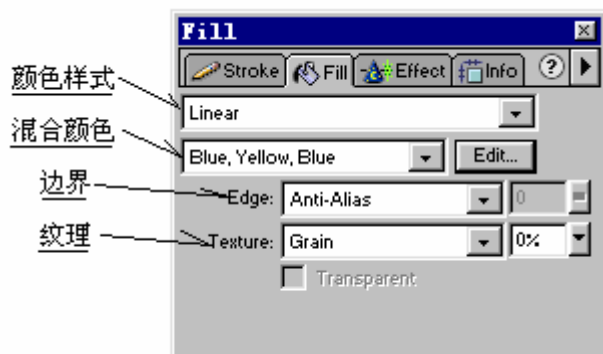


图 5.5 字符串颜色设置

(2) 字号的改变。字符串在这里也不一定设置成同一个字号，可以将字号设成从小到大或由大到小的变化使人们产生自远而近或自近而远的感觉。单击工具条中的  按钮选用  图标，然后用鼠标拖动字符串边框改变字号的变化规律。

(3) 若需改变个别字符时，先用鼠标选择字符串，然后选择菜单 Text | Convert to Paths 项，再单击工具条中的  按钮，将字符串分离成一个个字符。然后执行以下操作之一。

- 单击  按钮后可拖动单个字符以改变其位置。
- 单击  按钮后可改变字符的大小并旋转字符的方向。
- 单击  按钮可设置字符周围的影线。

图 5.6 就是改变了字符串的设置后的情况。



图 5.6 美化字符串

(4) 若需按照指定的曲线位置显示字符串时，先用铅笔在画布上画出曲线，再在按住 Shift 键的同时选择字符串及曲线，选择主菜单的 Text | Attach to Path 项，字符串将附着在曲线上。

在选择 Text|Orientation 后分别选用 Rotate Around Path、Vertical、skew Vertical、Skew Horizontal 等项时，字符串将以不同的方式依附在曲线上；选用 Text|Reverse Direction 时，还可颠倒字符串的前后显示顺序。

若想脱离和曲线的附着关系时，选择 Text|Detach from Path 项即可。

5.3.4 利用 Fireworks 优化图像

一张图像中的颜色种类以及清晰度常常能反映出这张图像的质量。通常情况下，颜色的种类越多、清晰度越好时图像质量也越高，但同时这种图像的容量（占用的字节数）往往也越大，网络上的传输时间以及下载的时间也越长，这些都是网络应用中的不利因素。

如何在图像的“质量”和“容量”之间找到最佳的结合点，是网络应用中必须考虑的问题。Fireworks 软件中的“图像优化”功能在这方面提供了有力的支持。

下面来看看 Fireworks 是如何优化图像的。

（1）如果在 Dreamweaver 界面中插入图像后，直接进行优化，可选择菜单 Commands|Optimize Image in Fireworks 项；如果在 Fireworks 界面中对图像进行了编辑后再进行优化，可在图 5.3 中选择 4-up 页，窗口中将出现 4 张相同的图像，再选择 Fireworks 的菜单 Commands|Optimize 项，将弹出图 5.7 的优化图像窗口。

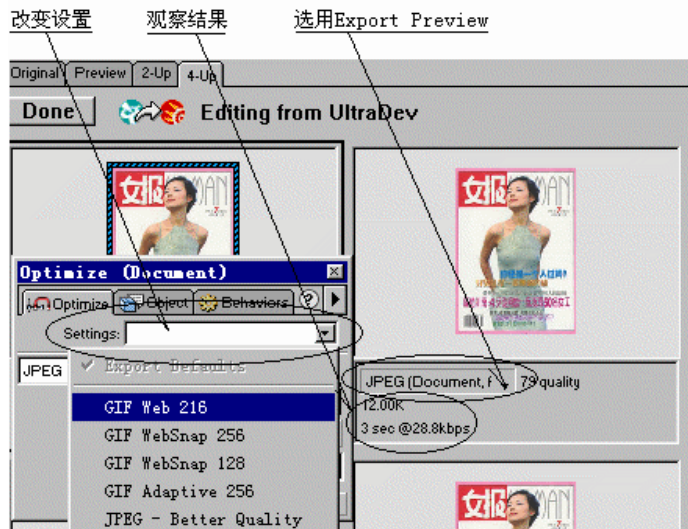


图 5.7 优化图像

（2）单击一张图像，在图像下面的下拉菜单中选择 Export Preview 项。

（3）在优化窗口的 Settings 的下拉菜单中改变选择参数，观察图形的清晰度以及图像下面数据的变化。例如，图中的 12.00k 代表图形的容量，3 sec 代表下载时间（以 28.8kbit/s 的速度下载，时间为 3 秒）。

（4）分别为 4 张图像设置不同的参数，以便进行比较。

（5）设置完图像后单击“Done”按钮，返回 Dreamweaver 环境，优化过程结束。

5.4 编辑和插入 GIF 动画

可以在 Dreamweaver 网页中插入由 Fireworks 创建的 GIF 动画(Animated GIF), 以增强网页的动态效果。

Fireworks 软件是通过改变图像大小、颜色、旋转、变形等方法来产生动态效果的。它先将一系列静止的 GIF 图像放置到不同的帧(Frame)中, 当连续显示这些帧时, 就产生了动画效果。

下面将结合两个简单的示例来说明在 Fireworks 4 环境中创建 GIF 动画的过程。

1. 创建一个旋转体

(1) 选择主菜单的 File|New 项, 在弹出的对话框中确定图像的大小后弹出图 5.8 左图。

(2) 选择菜单 Window|Frames 项, 弹出图 5.8 右图, 图中已出现 Frame1 字样。

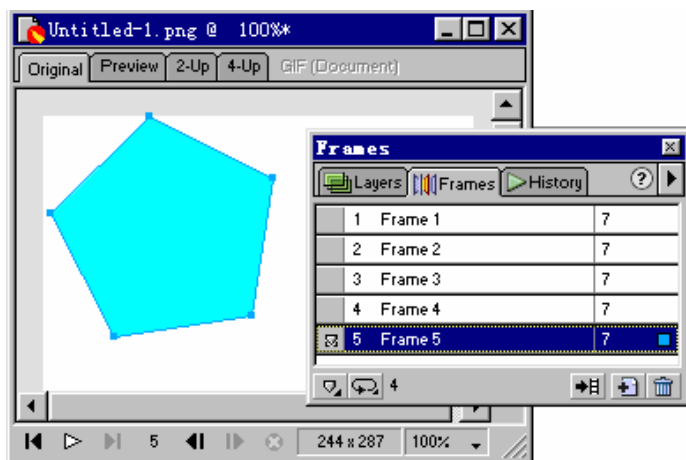


图 5.8 旋转多边形

(3) 在画布上绘图后(本例中绘制一五边形), 然后右击图形, 在弹出的菜单上选择 Edit|Copy, 或者直接选用菜单 Edit|Copy 项。

(4) 单击右图中的“增加新帧”按钮, 增加一帧(Frame2)后选择菜单 Edit|Paste, 将上帧的图形复制过来, 以便在此基础上进行修改。本例中是使五边形旋转一个角度, 然后再按第 3 步的要求复制图形, 并增加新帧, 如此反复设置多帧。

用上述方法绘图的目的在于使图形逐渐变化, 运行起来具有连贯性。

(5) 单击左下角的“播放”按钮观察效果, 如果满意, 按照以下步骤输出文件到站点目录下。

■ 选择 File|Export Wizard 菜单项。

- 在弹出的界面中选 Select an export format 单击“ continue ”按钮。
- 选 Animated GIF 项后按 continue 按钮。
- 优化图形, 完成后单击 Export 按钮, 将文件存放到站点目录下。

2. 创建逐个显示字符的字符串

(1) 确定画布大小并在画布上添加字符串(本例中为“ 欢迎选购 ”), 然后调整各字符的

大小和颜色。

(2) 单击菜单的 Text|Convert to Path 项，将字符串分离成一个个字符。

(3) 按住 Shift 键的同时，用鼠标分别单击各个字符，然后单击“Distribute to Frames”键(), 各字符将自动分配到各帧中，图 5.9 中的 frame1、frame2...就是分配的帧。



图 5.9 动态字符

(4) 参照上例中的(3)(4)步，利用 Copy/Paste 按钮调整或补充各帧的内容。

(5) 单击“播放”按钮将出现从左到右逐个显示字符的效果，如果满意，按上例的第六步输出文件到站点目录下。

5.5 图像的超链接

图像的超链接与文本超链接非常相似，都是在它的属性面板中将它的 Link 属性指向链接的网页。

除此以外，图像的链接还有更强的功能，一张图像可以划分成几个部分，分别链接到不同的网页。比如一张地图，就可以将不同的城市链接到各自的网页上以介绍该城市的情况。

图像属性窗口中的几个不同形状的图标，就是用来在图像上划分链接区域的，各个图标的作用如下。

：选择工具，用来选择图像。

：矩形工具，用来在图像上划分出矩形区域。

：圆形工具，用来在图像上划分出圆形区域。

：多边形工具，用来在图像上划分出多边形区域。

下面通过一个例子来说明利用多边形工具设置链接区域的方法。

假定将图 5.10 的帐篷设置成一个链接区，其步骤如下。



图 5.10 设置链接区示例

- (1) 在图像的属性窗口中选择多边形工具。
- (2) 将鼠标移动到帐篷的左下角，单击鼠标。
- (3) 将鼠标沿着帐篷的外边缘移动（注意此时并没有按下鼠标），到达直线终点时再单击鼠标，这样就画出了多边形的一条边。
- (4) 将鼠标移动到下一个位置后单击，又画出了第二条边。
- (5) 反复多次用多边形围住帐篷，此时整个帐篷变了颜色，表示这是一个链接区，设置完成。在浏览器显示时将看不见改变后的颜色。

链接区设置好以后，再通过它的 Link 属性链接到各自的网页，就完成了图像的链接任务。

5.6 图像的切换

所谓图像切换是指当鼠标移动到图像上时，图像自动切换成另一张图像，当鼠标离开图像时又自动恢复原图。用这种方法产生动态效果。

为了实现这一功能，需要准备两张大小相近的图像，然后单击对象窗口 Common 页中的图标，将弹出图 5.11 所示的对话框。

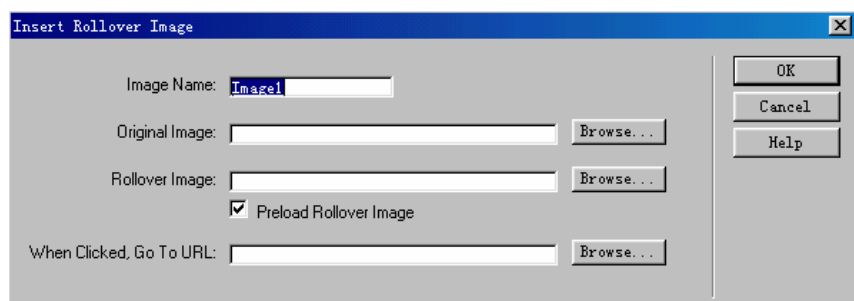


图 5.11 图像切换的设定

在 Original Image 文本框中直接键入或者借助于右边 Browser 图标的帮助输入原图的名字及其路径。

在 Rollover Image 文本框中直接键入或者借助于右边 Browser 图标的帮助输入切换后图形的名字及其路径。

Preload Rollover Image 选项如果被选中，将把切换的图形预先装入内存，以提高切换的速度，当然这样做也占用了较多的内存空间。

在 When Clicked, Go To URL 文本框中填入当鼠标单击图像时转向的网页。

第 6 章 框架与图层

6.1 框架

6.1.1 概述

框架中包括框架集(Frameset)和框架(Frames)两种不同类型的文档。

框架集是一个 HTML(或 ASP) 文档,它只存储框架的结构信息,如框架的数量、大小、边界以及各框架中放置的页面文件等。框架集之间可以嵌套。

框架是 HTML(或 ASP) 文档的容器,当屏幕包括多个框架时,就能将多个文档集中显示在同一个屏幕上。比如可以将目录文档放在一个较小的框架中,这个目录区有时称为“导航区”;将 HTML 文档显示在另一个框架中,这个显示区称为文本区。这些文档本身虽然都是独立的,但可以通过链接的方法把它们联系起来。例如将每一目录项与各自的网页文件链接,当用鼠标点击目录时,相应的文档就显示在目录相邻的框架中。把目录以及目录对应的文档显示在同一张网页的做法,给查阅或对比文档的内容带来很大的方便。

框架集与框架是两个不同层次的文档,框架集可以看成是框架的“父亲”,框架可以看成是框架集的“儿子”。

6.1.2 创建框架集

1. 插入预定义框架集

Dreamweaver UltraDev 定义了 8 种类型的框架集。打开对象面板中的 Frames(框架) 页(如图 6.1 所示), 单击其中一种图标, 网页中即可出现该类型的框架集。



图 6.1 预定义框架

图标中的蓝色区域代表框架集中当前框架的位置。

也可选择菜单 Insert|Frames 项, 然后在子菜单上选择插入一种预定义的框架。

2. 框架集嵌套

一个框架集放在另一个框架集内时,就在框架集之间形成了嵌套关系。创建嵌套框架集时,

先将光标放入需嵌套的框架内，然后采用以下方法之一。

- (1) 选择菜单 Modify|Frameset|Split Fram Up(或 Down、Left、Right) 项。
 - (2) 在对象面版中选择一框架集图标。
 - (3) 在文件窗口中，用鼠标拖曳框架边框，从水平或垂直方向分割框架。
- 当分成上下两个框架时，源代码如下。

```
<html>
<head>
  <title>Untitled Document</title>
  <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<frameset rows="80,*" frameborder="Yes" border="1" framespacing="0">
  <frame name="topFrame" noresize src="file:///D:/fff/UntitledFrame-2" >
  <frame name="mainFrame" src="file:///D:/fff/Frame2.asp">
</frameset>
<noframes>
  <body bgcolor="#FFFFFF" text="#000000">
  </body>
</noframes>
</html>
```

代码中从<frameset...>到</frameset>之间为框架集。框架集中包括两个框架(frame)，其中一个取名为“ topFrame ”，另一个取名为“ mainframe ”。每个框架内都放有网页文件，这些文件都是框架集生成时自动产生的缺省值，都可以在以后的设计中加以修改或重新设定。

当框架集之间嵌套时，其代码如下。

```
<html>
<head>
  <title>Untitled Document</title>
  <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<frameset rows="80,*" frameborder="Yes" border="1" framespacing="0">
  <frame name="topFrame" noresize src="file:///D:/fff/UntitledFrame-2" >
  <frameset cols="*,299" frameborder="Yes" border="1" rows="*">
    <frame name="mainFrame" src="file:///D:/fff/Frame2.asp">
    <frame name="rightFrame" noresize src="file:///D:/fff/UntitledFrame-3">
  </frameset>
</frameset>
<noframes>
  <body bgcolor="#FFFFFF" text="#000000">
  </body>
</noframes>
```

</html>

上述代码代表的框架集的结构如图 6.2 所示。

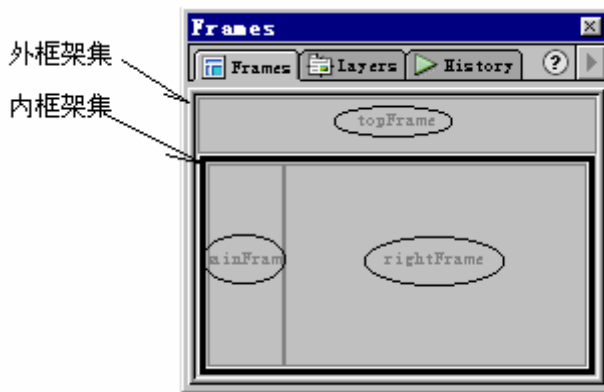


图 6.2 框架集上下嵌套

这是两个框架集嵌套的情况。内框架集包括“mainFrame”和“rightFrame”两个框架，外框架集包括“topFrame”框架和内框架集两部分。

代码的最后还有如下代码。

```
<noframes>
<body bgcolor="#FFFFFF" text="#000000">
</body>
</noframes>
```

这段代码是为那些不支持框架的浏览器准备的。由于早期版本的浏览器不支持框架，访问包括框架的网页时将发出错误信息。为了避免这种情况发生，可以在这段代码的<body...>...</body>标签中间写上相应的字符串。当浏览器不支持框架时将自动跳到这里来显示这段字符串。

3. 框架面板

选择菜单 Window\Frames 项或者利用 Ctrl+F10 组合键即可打开框架面板。框架面板是一个可视化的管理器，通过它可以看到框架集的结构以及各框架名。其中三维边框代表的是框架集的边框，细灰色边框代表的是框架的边框。图 6.3 显示的是两个框架集左右嵌套的情况。

用鼠标单击框架就选中了该框架，单击框架集边界时就选中了该框架集，然后就可以到它们各自的属性面板中去查看或编辑它们的属性了。

从框架面板中可以看到，每个框架都有自己的缺省名字，如 mainFrame、topFrame、leftFrame 等，这些名字可以在属性面板中修改。

4. 为框架集设置属性

框架集与框架分别有自己不同的属性面板，选择框架集后，将弹出如图 6.3 所示的框架集属性面板。框架集的属性包括：框架的行 (Rows)、列数 (Cols)；框架之间是否有边界 (Borders) 以及边界的宽度 (Border Width) 和颜色等。

在框架集边界 (Borders) 的下拉菜单中有 3 个选项，要显示边框时，选用“ Yes ”；不显示边框时选用“ No ”；选择“ Default ”时，由客户的浏览器来决定是否显示边框。如果选择“ Yes ”，

还要在 Border Width 中输入一大于 0 的数。缺省情况下 Borders 的设置是“ No ”，因此常常出现这样的情况，设计时还可以见到的分隔线，到浏览器运行时不见了。这就是由于没有在框架集中设置分隔线的结果。

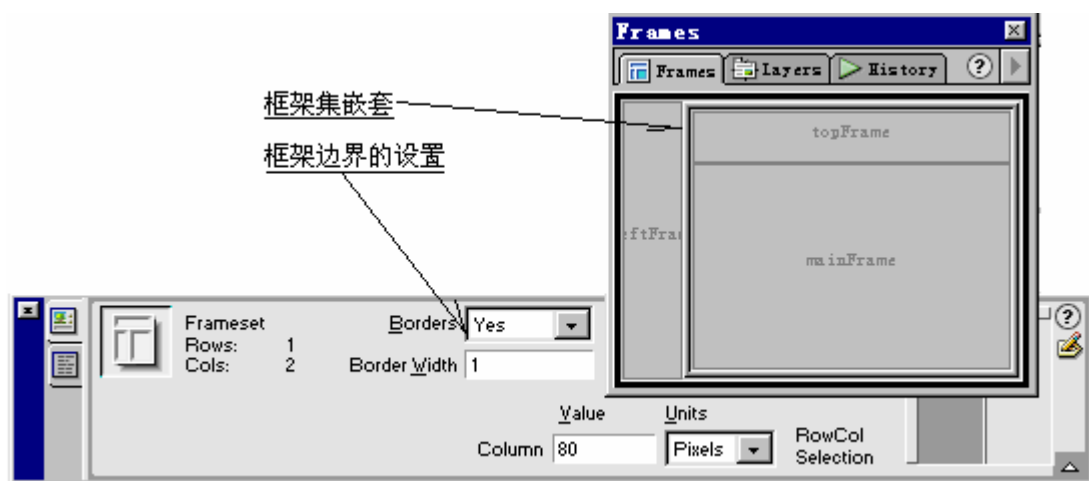


图 6.3 框架面板及框架集属性板

6.1.3 设置框架的属性

框架的最主要属性是框架内放置什么文档。为了确定这个文档，最好先把这些文档 *.htm 或 *.asp 文件) 单独设计好，然后再在框架的属性面板中的 Src 文本框中定义。框架的属性面板如图 6.4 所示。



图 6.4 框架属性面板

除此以外，框架的其他属性还有以下几项。

- Frame Name(框架名): 系统为每个框架都提供了一个缺省名，在这里可以对它们加以修改。
- Scroll(滚动条): 用来确定当内容超出框架时，是否显示滚动条。这里有四种选择：Yes(显示滚动条)、No(不显示滚动条)、Auto(当内容超出框架时自动显示滚动条)、Default (采用浏览器的缺省值，多数浏览器的缺省值是 Auto)。
- No Resize(不能调整大小): 选择此项时，用户不能用鼠标拖动边框来改变框架的大小。
- Border(边框): 确定当前框架是否显示边框，包括 Yes(显示)、No(不显示)、Default (缺省) 3 种选择。大多数浏览器的缺省值是 Yes。

说明：若父框架集的 Border 设为 No，而框架设为 Default，或者所有框架的此项都设为 No 时，框架的边框将被取消。

- Border Color(边框颜色)：本框架的颜色。这里设置的颜色如果与框架集中的设置有矛盾时，以这里的设置为主。
- Margin Width(水平间距)：框架边框与内容在水平方向的间距。
- Margin Height(上下间距)：框架边框与内容在垂直方向的间距。

6.1.4 删除框架

删除框架的方法非常简单，只要将鼠标移动到边框上，当光标形状改变成左右（上下）箭头时，按下鼠标将边框拖出框架集或者整个屏幕，由该边框分隔的框架随即消失。

6.1.5 对框架内容的控制

在一个框架中使用链接，可以改变另一个框架中的内容。具体操作如下。

- (1) 选定框架中的对象或文本。
- (2) 在属性面板中的 Link 文本框中输入链接的文件名，或单击右方的文件夹图标后选择链接的文件。
- (3) 单击属性面板中的扩展按钮，然后在 Target 下拉菜单中选择一个目标框架名。下拉菜单包括如下几项。
 - _blank(空白)：将文件放入一个新的空白窗口中。
 - _Parent(上一级)：将文件放入父窗口中。
 - _self(自身窗口)：在当前框架中打开文件，用新文件取代原有文件。
 - _top(顶部)：在最外面的框架集中打开文件，替换所有的框架内容。

6.2 图层

6.2.1 图层的作用

图层是一个容器，在图层内可以放置各种类型的网页元素。如文本、图像、表格、甚至还可以放置图层（图层嵌套）。每个图层相当于一个小屏幕，一个独立的小屏幕。

图层与框架有何不同？

图层是一个可以任意移动的容器，甚至允许图层之间重叠放置，这是它与框架的不同之处，这也是图层的最大优点。因为放置在图层上的元素，可以随图层被拖放到任意位置，为元素的定位和网页布局带来极大的方便，同时也为控制动态元素奠定了基础。

早期版本的 Internet Explorer 和 Netscape Navigator 浏览器都不支持图层，只有 4.0 及以后的版本才支持图层。但即使是现在，这两种浏览器对图层的定义方法还没有完全统一。例如，

- Internet Explorer 4.0 以及 Netscape Navigator 4.0 以上的浏览器都支持用 <Div>...</Div> 或 ... 来定义层。

- Netscape Navigator 4.0 浏览器还用 <LAYER>、<ILAYER> 来定义层。

在 Dreamweaver UltraDev 中可以使用 DIV、SPAN、LAYER、ILAYER 等标记来定义图层。

6.2.2 创建图层

和表格一样，图层必须在“标准视图”的环境下才能创建。在网页中创建图层可采用下列方法之一。

- 选择菜单 Insert|Layer 项。
- 直接将对象面板 Common 页中的图层图标拖进文档窗口，当松开鼠标时，文档窗口中就出现了一个新的图层。
- 用手工方法直接绘制图层。方法是，先单击图层图标，然后到文档窗口中拖动鼠标，一个所需要的图层将出现在文档窗口中。

前两种方式，图层按缺省设置的大小出现（当然以后还可以改变）；第三种方式则根据当时绘制的情况显示大小。图 6.5 是创建了两个图层后的情况。

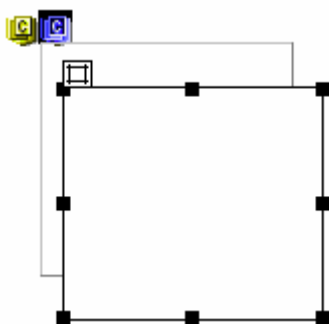


图 6.5 两个图层示例

此时图层相应的源代码如下。

```
<body bgcolor="#FFFFFF" text="#000000">
<div id="Layer1" style="position : absolute; left:68px; top : 34px; width : 160px; height : 134px;
z-index : 1">
</div>
<div id="Layer2" style="position : absolute; left : 121px; top : 60px; width:157px; height : 123px;
z-index : 2">
</div>
</body>
```

此代码说明这里创建了两个图层，每个图层用的标签都是<div...>...</div>。这是 Dreamweaver UltraDev 的缺省设置。第一个图层的名字 id 是 Layer1，第二个图层的名字 id 是 Layer2。其他均为图层的参数，它们的含义将在后面叙述。

6.2.3 设置图层的属性

1. 设置缺省的属性

设置图层的缺省属性，可选择菜单 Edit|Preferences 项，将弹出图 6.6 所示的对话框。然后在左边的 Category 中选择 Layers 项，在右边 Layers 中为图层设置缺省属性。

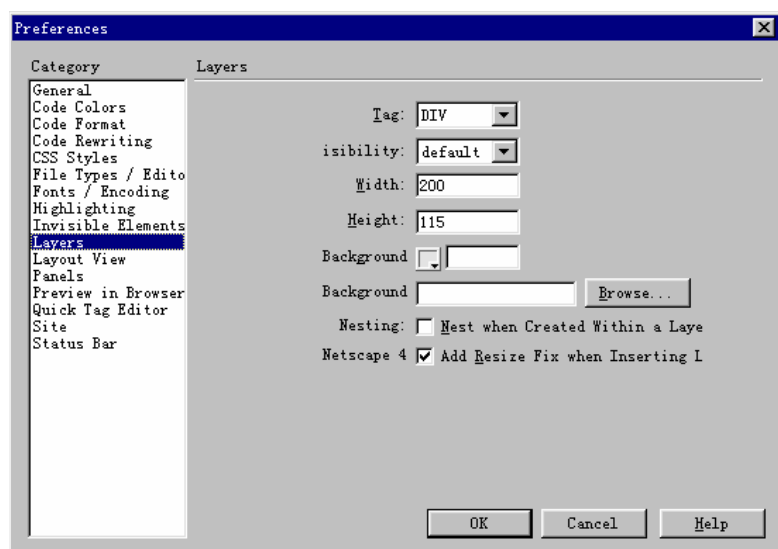


图 6.6 图层缺省值的设定

窗口中的属性包括以下几个方面。

Tag：缺省时的图层定义标签。包括 DIV（缺省）、span、layer、ilayer 几项选择，最好选择 DIV 或 span，因为它适用于两种主要浏览器的 4.0 及以后的版本。layer 或 ilayer 只适用于 Netscape 4 浏览器。

Visibility：决定缺省情况下图层的可视特性。

Width 和 Height：当用菜单 Insert|Layer 创建图层时，图层缺省的宽和高。

Background (Color)：图层缺省的底色。

Background (Image)：图层缺省的底图。

Nesting：当新的图层生成在原有图层中时，是否形成嵌套关系。在按下 Alt 键的同时生成新图层时将改变这里的设定。

2. 修改图层的属性

生成图层时，图层大小、底色等虽然按照缺省的设置生成，但生成以后还可以在图层的属性面板中修改。图层的属性面板如图 6.7 所示。



图 6.7 图层属性面板

图层的主要属性包括以下几方面。

(1) 位置属性 (L 和 T)：其中 L 代表 Left，为图层与页面左边的边距。T 代表 Top，为图层与页面顶端的边距。如果是嵌套的内层图层时，则是与外层的左边距和顶边距。

(2) 大小属性(W 和 H): 图层的宽度(W)和高度(H)。

L、T、W、H 等属性缺省时都是以像素(px)为长度单位, 设计时也可以指定其他类型的单位, 如: p(点)、in(英寸)、mm(毫米)、cm(厘米)、%(百分比)等。单位的缩写必须紧跟在值的后面(中间不能留空格)。

(3) 层次(深度)属性(Z-Index): 深度属性是图层特有的属性。它实际上是图层的堆栈排列的顺序号。当图层堆叠放置时, 底层图层的 Z-Index 最小。

(4) 可视属性(Vis): 图层的可视状态, 包括以下几项选择。

- Default(缺省): 在缺省的设置中选择的可视状态。
- Inheri(继承): 图层嵌套时继承父层的可视状态。
- Visible(可见): 显示图层及其包括的内容。
- Hidden(隐藏): 隐藏图层及其包括的内容。

6.2.4 图层面板

图层面板是用于管理图层的可视化工具。选择菜单 Window|Layers 项或者按 F11 键, 就能打开如图 6.8 所示的图层面板。

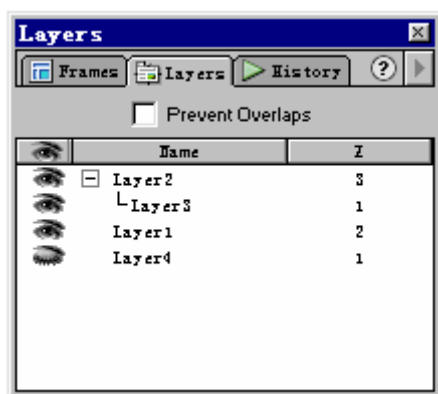


图 6.8 图层面板

每当创建一个图层时, 面板中将出现一个新图层的缺省名字, 图层在深度上的排列与堆栈的排列相似, 层号(Z-Index)是一个整型数, 最先创建的图层的层号最小, 后来创建的层都放在上面, 层号也逐渐增大。

在面板中还可以看出图层之间的嵌套关系, 图中 Layer3 就嵌套在 Layer2 的图层中间。如果图层之间存在嵌套关系, 当拖动外层图层时, 被嵌套的内层将跟随移动; 如果内层的图层的可视属性(Visible)设成继承(Inherit)时, 其可视状态将随外层图层的状态而改变。

当需要改变图层之间的嵌套关系时, 在图层面板中, 按住 Ctrl 键的同时用鼠标将图层名拖出来(或拖进去)即可。

用鼠标单击图层名的左边, 将形象地显示图层的可视状态: 👁 符号代表该图层为可视状态, 👁 符号代表为不可视状态。当用鼠标单击上述图标时, 将在两种符号之间转换。

当需改变图层的深度顺序时可以直接在图层面板的 Z 符号下面填写整型数。

如果选择 Prevent Overlaps 项时, 图层之间将不允许重叠放置。

6.2.5 图层间的对齐操作

如果多个图层需要对齐时，可采用以下方法。

- (1) 选中一个以上的层。
- (2) 选择 Modify|Layers and Hotspots 子菜单中任一对齐项。如果选择 Align Left (Top)，则所有的层都以最后选择的层左边(上边) 对齐。

当需要图层与栅格对齐时，可采用以下方法。

- (1) 先用以下的方法之一显示栅格。
 - 选择菜单 View|Grid|Show 项。
 - 选择菜单 View|Grid|Settings 项后再选 Visible Grid 复选框。
- (2) 选择菜单 View|Grid|Snap To 项打开对齐对话框。
- (3) 选择并拖动一个层，当松开鼠标时，层会自动对齐最靠近的栅格位置。

6.3 图层与表格之间的转换

由于图层定位的灵活性，用它进行网页布局非常方便。但 4.0 版本以前的浏览器不支持图层，因此只有先用图层进行布局设计然后转换成表格，才能为大多数类型的浏览器使用。

6.3.1 图层转换为表格

当多个图层需要转换成表格时，按照以下步骤进行。

- (1) 选择需要转换的图层。
- (2) 选择菜单 Modify|Convert 下面的 Layers to Table 项，将弹出如图 6.9 所示的对话框。

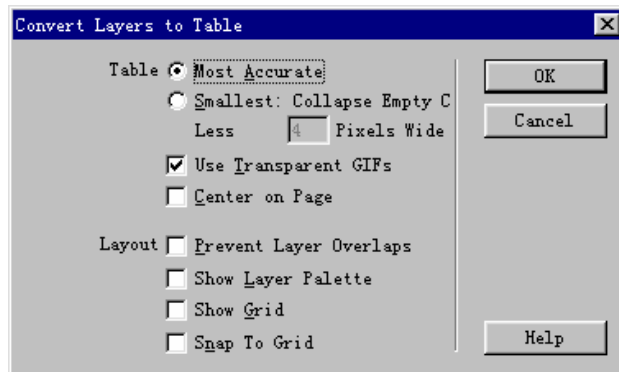


图 6.9 图层转换成表格

- (3) 按照以下要求选择窗口中的选项。

- Most Accurate(最精确): 为每个图层建立一个单元格，如果单元格之间存在着间距，则用插入单元格的办法补充这些间距。

- Smallest(最小): 当选择此项时，将压缩空单元格，使得生成的表格的行列数最小。但是布局的情况可能有些变化。

- Use Transparent GIFs(使用透明的 GIF 图像): 选择此项时，将把透明的 GIF 图像放

在表格的最后一行，以便各种浏览器都显示出相近的布局。

■ Center on Page(放置在网页的中间): 选择此项，则生成的表格位于网页的中间，如果不选择这一项，将对正左边。

(4) 选择希望的布局工具及点阵后击“ OK ”按钮。

6.3.2 表格转换为图层

当表格中的元素想变成可以移动的元素时也可以将表格转换成图层。具体步骤如下。

(1) 选用菜单 Modify|Convert 下面的 Table to Layers 项，将弹出图 6.10 所示的对话框。

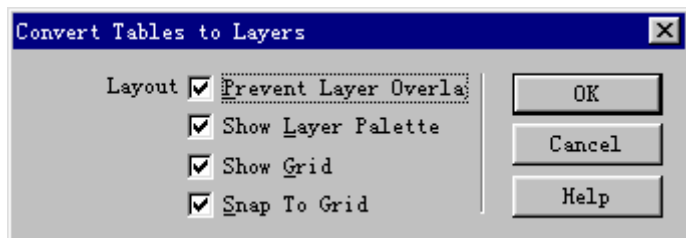


图 6.10 表格转换成图层

(2) 在窗口的 Layout 中选择选项。

■ Prevent Layer Overlap(防止图层重叠): 选择此项时可以在建立、移动或调整图层时防止图层之间重叠。

■ Show Layer Palette(显示图层面板): 选择此项时，在完成转换后显示图层面板。

■ Show Grid(显示栅格): 选择此项时将显示栅格。

■ Snap To Grid(对齐栅格): 选择此项时将自动对齐栅格。

表格转换图层时，空单元不会转换除非它包含有底色。

第 7 章 表单

7.1 概述

7.1.1 浏览器与服务器的交互过程

随着 Web 应用领域的扩大和深入，服务器与客户之间不再是一种简单的信息发布和浏览的关系。以电子商务为例，当客户从事选购或支付等商务活动时，不仅需要查看从服务器发来的商品信息，也需要向服务器发出采购和支付方面的数据。服务器既要发布商品信息也要接收和保存来自客户端的采购数据（例如将客户发来的信息存入数据库中等）。网上的商务活动过程，实际上就是客户端与服务器端之间的交互过程。

客户端为了向服务器发出信息，需要先将信息放在一些专门的网页元素中，这些元素称为“表单元素”，而表单（Form）就是表单元素的容器，它是浏览器向服务器发出信息的关键部分。

从客户端发出信息的过程可用图 7.1 表示。

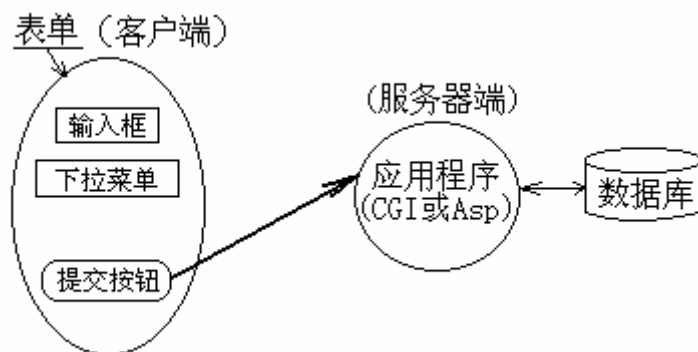


图 7.1 表单的工作示意

客户先在浏览器端的输入框、下拉菜单等表单元素中输入数据，完成输入后单击“提交”按钮，将信息传送到服务器并启动事先指定的页面去处理这些提交的信息。

7.1.2 表单的结构

“表单”（Form）是用 HTML 语句定义的显示界面，每个表单以<Form>标签开始，</Form>标签结束。一张表单是提交数据的一个单元，每张网页中允许创建一个或多个表单。

表单中可以包括任意 ASCII 代码字符串和除 Form 以外的任意 HTML 标签。表单元素包括单行、多行文本输入框，各类按钮，下拉菜单等。表单元素用 HTML 规定的标签定义。表单的大致结构如下。

<Form Action="..."Method="...">	表单开始标签
<Input ...>	单行文本输入框、单选或多选按钮
<Select ...>...</Select>	下拉菜单
<Textarea ...> ...</Textarea>	多行文本框
<Input Type = "Submit" Value = "提交">	提交按钮
<Input Type = "Reset" Value ="复位">	复位按钮
</Form>	表单结束标签

7.2 表单中的两个重要属性

表单中有两个重要属性，Action 属性和 Method 属性。

1 . Action 属性

标签<Form Action="...">中 Action 的等号后面有一串字符，代表 URL 资源定位器，它指向一个网页。当单击“提交”（Submit）按钮时，立即转向该网页，以处理从“表单”中提交的数据。

2 . Method 属性

标签<Form Action="..."Method="...">中 Method 等号后面的字符串，用于定义提交信息的方式，包括 3 种选择：“Get”、“Post”和“Default”。

使用“Get”方式提交信息时，表单中的信息作为字符串自动附加在新的 URL 后面立即送出，因此执行效率高，但由于 URL 的长度最长为 8192 个字符，如果提交的信息加上原 URL 超过上述长度时，超过部分将会被系统切断。

使用“Post”方式提交信息时，需将输出的信息进行包装并存入单独的文件中（不附在 URL 后面），等待服务器取走。利用这种方式提交时效率较低，但信息量没有限制。

选择“Default”方式时，提交信息的方式取决于浏览器的设置。通常情况下为“Get”方式。

7.3 两个专用按钮

表单中有两个专用按钮，它们是“提交”按钮与“复位”按钮。系统已经为它们编写好了脚本，可用来完成以下专门的任务。

- 提交按钮（Submit）用来向服务器提交表单元素中的数据。
- 复位按钮（Reset）用来将表单元素中的数据复位（清空），以便重新输入新数据。

一个网页可以包括多个表单，每个表单最少应有一个提交按钮和一个复位按钮。

以上属性与专用按钮都是用围绕着表单数据的提交而设置的。表单提交数据的工作包括以下几个方面。

- 准备工作：单击“复位”按钮清空表单元素。
- 提交的时机：单击“提交”按钮。

- 提交的方式：用 Method 属性定义。
- 提交的对象：用 Action 属性定义。

7.4 表单元素的 HTML 标签

各类表单元素都是用 HTML 语句定义的。下面通过实例来说明定义的方法。

1. 单行文本框

```
<Input Type="text" Name="College" Value="中南大学" size=30 Maxlength=50>
```

其中<Input Type="text"...>是单行文本框的定义，其他为单行文本框的属性，这里的属性包括：文本框的名字 "College" \ 缺省值 "中南大学" \ 宽度 30 \ 最大允许字符长度 50)。定义不能缺省，而属性的顺序可以变化，属性的项目也可以根据需要取舍。

2. 多行文本框

```
<Textarea Name="suggest" Rows=5 Cols=30> 建议如下：</Textarea>
```

其中<Textarea...>...</Textarea>是多行文本框的定义，其他为多行文本框的属性，这里的属性包括，多行文本框的名字("suggest") \ 行数 5、列数 30。

3. 单选按钮

```
<Input Type="radio" Name="Sex" Value="女" Checked>女<p>
```

```
<Input Type="radio" Name="Sex" Value="男">男<p>
```

其中<Input Type="radio"...>是单选按钮的定义，其他为单选按钮的属性。由于上面两个单选按钮同名，因而是两个(互斥)的按钮，第一个按钮的输出值为 女(因为 Value="女")；第二个按钮的输出值为 男(因为 Value="男")，两个按钮中只能选中一个。缺省时选中的是 女 (因为定义中有 Checked 标签)。

4. 复选框

```
<Input Type="Checkbox" Name="Ball1" Value="篮球">篮球<p>
```

```
<Input Type="Checkbox" Name="Ball2" Value="排球">排球<p>
```

```
<Input Type="Checkbox" Name="Ball3" Value="乒乓球">乒乓球<p>
```

其中<Input Type="Checkbox"...>是复选框的定义，其他为复选框的属性，包括复选框的名字("Ball*")，复选框之间可以同名也可以不同名，根据需要选择。这一点结合后面的应用还将进一步说明。本复选框的输出值包括 篮球 、 排球 、 乒乓球 等。当同时选择前两项时，将输出“ 篮球 ”和“ 排球 ”。

5. “ 提交 ” 和 “ 重置 ” 按钮

```
<Input Type="submit" Value="提交">
```

```
<Input Type="reset" Value="重置">
```

“ 提交 ” 和 “ 重置 ” 按钮是两个专门的按钮，它本身不输出值，Value 是显示在按钮上的符号。

6. 下拉菜单

```
<Select Name="course" >
```

```
<option>语文
```

```
<option>数学
<option>物理
</Select>
```

其中<Select...>...</Select>是下拉菜单的定义，<option>后面是选项。本下拉菜单的名字“course”包括3项内容：语文、数学和物理。

7.5 创建表单的可视化操作

Dreamweaver UltraDev 提供了创建表单的可视化操作。下面介绍利用系统提供的工具创建表单的过程。

(1) 选择下列方法之一开始创建表单的工作。

- 先将光标置于表单将要放置的位置，然后选择菜单 Insert/Form 项。
- 先将光标置于表单将要放置的位置，然后从 Object 面板中选择 Forms 页的 Insert Form 图标。
- 直接将 Form 图标拖动到页面相应的位置。

利用 Forms 页中 Insert Form 图标创建表单的方法如图 7.2 所示。

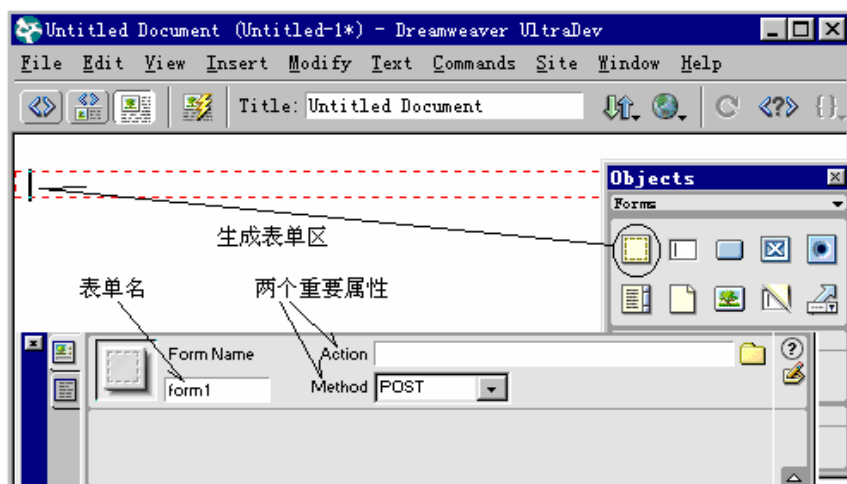


图 7.2 利用对象面板上的图标创建表单

(2) 为表单设置属性。

表单的属性可通过它的属性面板设置。这些属性包括以下几项。

- Form Name：表单名。每个表单赋予一个名称以便在脚本中调用。
- Action：输入用于处理表单数据的网页的地址。可以直接输入 URL，也可以通过右边的文件夹图标来确定。
- Method：提交数据的方法，可从下拉菜单中选择一项，菜单中共提供了 3 项选择。
Get：提交的数据放在 URL 后面一道送出。
Post：提交的数据单独打包提交。

Default：默认。使用浏览器默认的方法。

(3) 在表单域中增添表单元素。可以采用以下方法在表单中增添表单元素。

- 将光标置于表单边界以内，然后从 Insert | Form Object 子菜单中选择表单元素。
- 将光标置于表单边界以内，然后单击 Forms 页中的图标。
- 将 Forms 页中的图标直接拖动到表单内的适当位置。

下面主要介绍利用 Form 页中的图标增添表单元素的方法。

(4) 在属性窗口中分别为各个表单元素设置属性。对表单元素属性的设置，不仅可以改变表单元素的外形、位置和颜色，有的属性甚至还可以改变它们的用途。如同一个文本框，可以把它变成单行、多行文本框或者口令输入框；同一个按钮可以变成提交、复位按钮也可以变成通用按钮。

(5) 如果必要，在表单元素旁放置标签，输入说明性文字。

7.6 设置表单元素的属性

1. 文本框的属性

表单中文本框(Text Field)是输入文本的元素，其属性可在它的属性面板中设置，其属性面板的情况如图 7.3 所示。

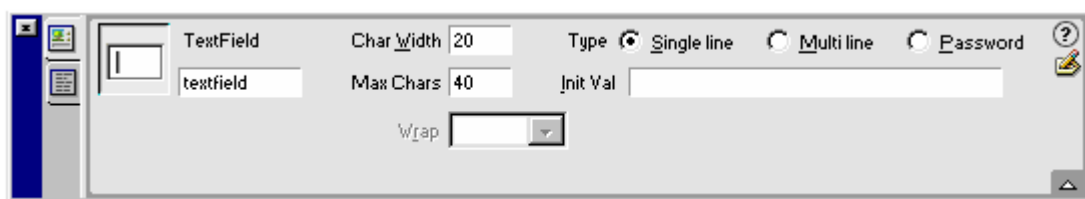


图 7.3 文本输入框属性面板

各属性的作用如下。

- Text Field(文本域)：给文本域指定一个唯一的名字。
- Char Width(可显示的字符长度)：确定文本域中能显示的字符长度。此长度可小于 Max Chars。
- Max Chars(最大字符长度/行数)：设定单行文本域所能显示的最大字符长度或多行文本域所能显示的行数。
- Type(类型)：用来设定是单行文本域、多行文本域或是口令文本域。
 - Single Line (单行文本域)：只能显示一行。
 - Multi Lines(多行文本域)：可以同时显示多行。
 - Password (口令文本域)：输入时只显示“ * ”号，以隐藏真正的数据。
- Init Val(初始值)：设置文本框中的初始值，初始值可以为空。

2. 按钮的属性

表单中的按钮(Button)，可作为触发事件的元素，其属性可在它的属性面板中设置，按钮的属性面板如图 7.4 所示。

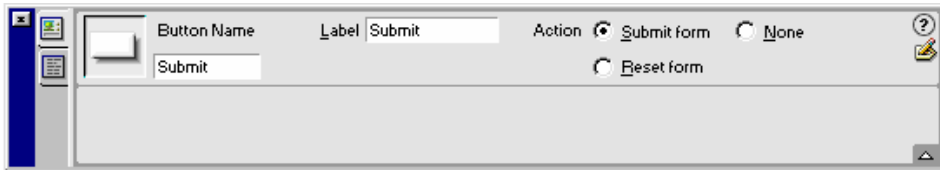


图 7.4 按钮的属性面板

各属性的作用如下。

- Button Name：给按钮分配一按钮名。缺省情况用的是“ Submit ”，代表提交按钮。可根据需要修改。

- Label：在按钮上显示的标签（字符串）。

- Action：代表按钮的作用。包括“ Submit Form ”、“ Reset Form ”和“ None ”3种，前两种的作用是“ 提交 ”和“ 复位 ”，前面已经讲述。第三种代表通用按钮，需要为它编写脚本以实现特定的功能。

3. 图像的属性

表单中的图像（ Image Field ）其属性可设置如下。

- Image Field：为表单中的图像起名字。

- Src：图像源文件所在的位置。可通过右方的文件夹图标选取。

- Alt：确定取代图像的文本。对于一些不支持图像的早期浏览器，将自动在图像位置上显示这些文本。

- Align：图像所处的位置（ 靠左、靠右、居中等 ）。

- Edit Image：编辑图像。单击此按钮时将根据预先的设定，自动进入图像编辑环境（如 Firework 等）。

4. 复选框的属性

表单中的复选框（ CheckBox ），提供了同时多种选择的机会。其属性可在它的属性面板中设置，属性面板的情况如图 7.5 所示。

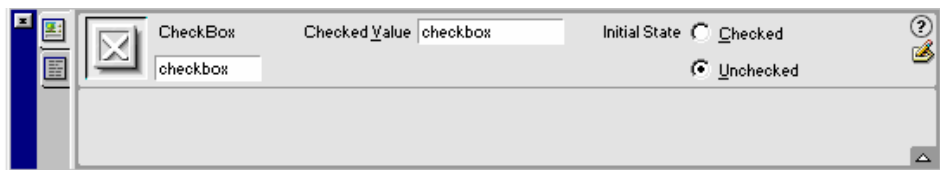


图 7.5 复选框的属性面板

各属性的作用如下。

- CheckBox：给复选框取名。多个 CheckBox 可取同一名称也可以不取同一名称。

- Checked Value：被选中时的输出值。

- Initial State：表单装入时，复选框的初始状态。Checked 代表被选中，Unchecked 代表未被选中。

5. 单选按钮的属性

表单中的单选按钮(RadioButton)如果同名, 按钮之间是一种互斥性的按钮。就是说, 在这组按钮中, 同一时间只能作一种选择。其属性可在属性面板中设置, 属性面板的情况如图 7.6 所示。

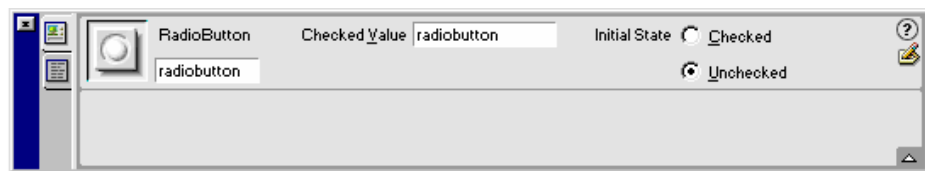


图 7.6 单选按钮属性面板

各属性的作用如下。

- RadioButton: 为单选按钮取名。同一组的单选按钮必须取同一名字。
- Checked Value: 被选中时的输出值。
- Initial State: 确定表单被载入时单选按钮的状态, 选中还是不被选中。Checked 代表被选中, Unchecked 代表未被选中。同一组的多个单选按钮中最多只能有一个被选中。

(6) 列表或下拉菜单的属性

表单中的列表/下拉菜单(List/Menu), 可给客户提供多项 / 单项选择。其属性可在属性面板中设置, 属性面板的情况如图 7.7 所示。

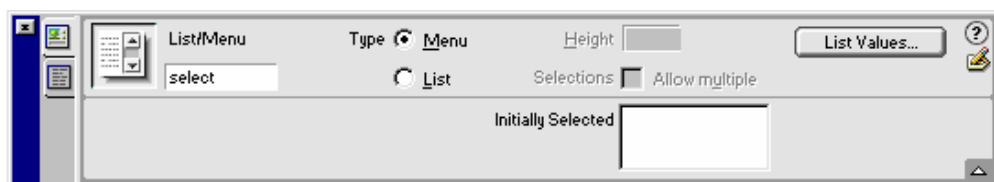


图 7.7 列表或下拉菜单属性面板

各属性的作用如下。

- List/Menu: 给列表/下拉菜单取一个唯一的名字。
- Type: 确定是下拉式菜单(Menu)还是滚动式多行列表(List)。如果是列表, 还可以进一步设置显示框的高度(显示的行数), 以及用户能否同时选择多项等。
- List Values: 单击“ List Value ”按钮, 将弹出一对话框。在这个对话框中可以为下拉菜单或列表增添选项。每一个选项包括标签(label)和值(value)两个方面。

如果 label 与 value 相同, 此时只需设定 label 就可以了, 系统会自动将 value 取成与 label 相同的值。

单击对话框中的“ + ”号, 可增加选项; 单击“ - ”号可删除选项。按向上、向下箭头可调整选项的前后顺序。

7. 文件域的属性

表单中的文件域(FileField), 可在属性面板中设置, 属性面板的情况如图 7.8 所示。

各属性的作用如下。

- FileField Name: 给文件域赋一个唯一的名字。
- Char Width: 确定文件域显示的字符数, 此数字可以小于 Max Chars。

- Max Chars：确定文件域中可以输入的最大字符数。



图 7.8 文件域属性面板

8. 隐藏域的属性

隐藏域 (HiddenField)，是一种运行时在浏览器中不显示出来的元素。这些元素的值通常只表示一种状态，作为内部判断之用。如果想查看隐藏域时，可选择菜单 Edit|Preferences 后再选择 Invisible Elements 项。

HiddenField 的属性可在它的属性面板中设置，属性面板的情况如图 7.9 所示。



图 7.9 隐藏域属性面板

各属性的作用如下。

- HiddenField：给 HiddenField 赋予一个唯一的名字。
- Value：给 HiddenField 赋值。

7.7 跳转菜单

跳转菜单 (Jump Menu) 是由多项链接指针组成的下拉菜单，在跳转菜单中选择不同的选项时可以分别跳转到不同的页面。创建跳转菜单的步骤与创建其他表单元素不同，因此单独列出来讲解。

插入跳转菜单的步骤如下。

(1) 选择 Insert|Form Object|Jump Menu 菜单项或者直接选择 Object 面板 Form 页中的 Jump Menu 图标，将弹出如图 7.10 所示的界面。

(2) Menu Items 列表中是菜单选项。确定菜单选项的方法是先单击“ + ”按钮，然后在 Text 中定义选项的内容，如“ 建校历史 ”、“ 校园情况 ”、“ 师资队伍 ”等。当需要删除选项时，单击“ - ”按钮即可。

(3) When Selected, Go To URL 为每个选项链接的网页，可以直接填写，也可以利用右边的 Browser 文件夹图标选择。

(4) Open URLs In 是指定新网页放置的位置。Main Window 代表放入主窗口中，如果网页中有框架，也可以指定放入框架的某个 Frame 中。

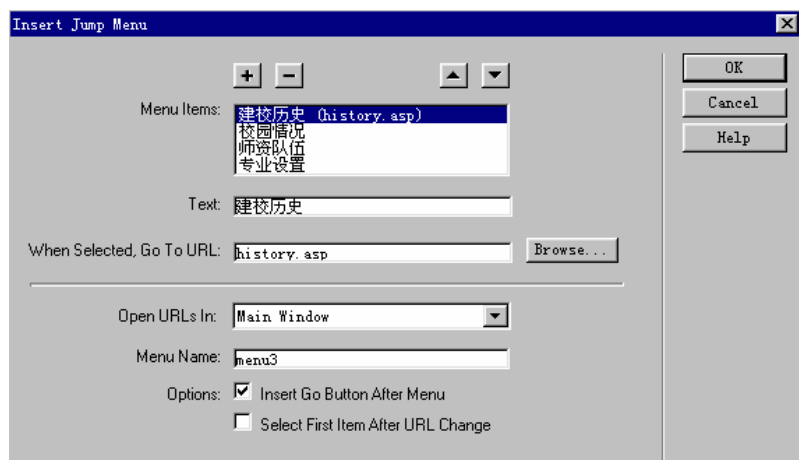


图 7.10 插入跳转菜单

(5) Menu Name 代表菜单名。

(6) Options 的两个选择都是针对菜单的第一选项的。

■ Insert Go Button After Menu 项如果被选中，在菜单后面将出现一个“ Go ”按钮。单击该按钮时才能转向第一项链接的网页。

■ Select First Item After URL Change 项如果被选中，则只有 URL 改变时才能转向第一项链接的网页。

7.8 表单综合示例

下面以一个网上招聘广告为例说明表单的创建。

在浏览器端看到的表单情况如图 7.11 所示。

图 7.11 表单示例

表单的源代码如下。

```

<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<body bgcolor="#FFFFFF" text="#000000">
<h2 align="left"><font color="#CC0033"><i> 招 聘 </i> </font></h2>
<h5 align="left"><i><font color="#000000" size="1">
湖南开拓计算机公司是一个股份制软件开发公司，开办几年来已经为湖南地区的网络发展作出了很大的
贡献，现在由于发展需要，拟招聘十个软件开发人员，特招聘 40 岁以下，大专以上学历，计算机专业毕业并
工作一年以上人员。
凡报名应聘者请于 2001 年 12 月底以前填写以下数据，公司将根据应聘条件进行面试。</font></i></h5>
<form name="form1" method="post" action="handle.asp">
<h5> 姓名：<input type="text" name="textfield" size="10">
      年龄：<input type="text" name="textfield2" size="8">
      性别：男 <input type="radio" name="radiobutton" value="男" checked>
           女 <input type="radio" name="radiobutton" value="女">
      婚姻：已 <input type="radio" name="radiobutton2" value="已婚">
           未 <input type="radio" name="radiobutton2" value="未婚" checked>
</h5>
<h5> 个人爱好：
      体育 <input type="checkbox" name="checkbox" value="体育">
      文艺 <input type="checkbox" name="checkbox2" value="文艺">
      音乐 <input type="checkbox" name="checkbox3" value="音乐">
      绘画 <input type="checkbox" name="checkbox4" value="绘画">
</h5>
<h5> 学历：
      <select name="select">
        <option>大专以上</option>
        <option>本科以上</option>
        <option>研究生以上</option>
        <option>博士生</option>
      </select>
      从事计算机工作年限
      <select name="select2">
        <option>一年以上</option>
        <option>两年以上</option>
        <option>三年以上</option>

```

```
        <option>四年以上</option>
        <option>五年以上</option>
    </select>
    <input type="submit" name="Submit" value="提交">
    <input type="reset" name="Submit2" value="复位">
</h5>
</form>
</body>
</html>
```

第8章 DHTML 与 CSS 级联样式表

8.1 DHTML 技术概述

8.1.1 基本概念

在早期的 Internet 体系中, Web 运作的主要方式只是文本发布。服务器将文本、图片嵌在 HTML 标签中传送给浏览器, 浏览器解读后按 HTML 标签的定义显示数据。在这里, 服务器提供数据, 浏览器显示数据。网页的内容和形式在服务器发出时已经确定, 浏览器下载后不能再行改变, 如果想改变网页上某些元素的表现形式, 例如想改变某个元素的底色, 也只有重载这张网页, 以便启动服务器提供的服务。

这是一种静态关系, 但这种静态关系对于初期的应用来说矛盾并不明显, 因为初期使用 Web 的用户多数是一些科学工作者, 它们中的大多数人都只是浏览网页中的内容, 对网页的表现形式并不十分注重。随着 Web 应用领域的扩大, 对网页的交互和动态方面的要求越来越高, 静态网页已经远远不能满足客观的需要。

动态网页的出现, 适应了形势发展的需要, 是网页设计的重大突破。动态网页技术包括服务器端和浏览器端两个方面。

1. 服务器端的动态技术

服务器为了实时响应客户要求, 及时更换网页内容 (例如更换数据库记录等), 提供了 CGI、ASP、JSP 等服务器端动态技术, 这部分内容将在第 14 章介绍。

2. 浏览器端的动态技术

浏览器端的动态技术又称 DHTML, 它是 Dynamic HTML(动态 HTML) 英文的缩写, DHTML 技术是近年来网络发展进程中最振奋人心, 也最具有实用价值的创新之一。它的设计思想是: 浏览器从服务器端下载文档后, 利用浏览器本身的资源, 在不增加服务器负担和网上传输流量的前提下, 使网页的某些元素“动”起来。

DHTML 目前正在发展, 标准还在进一步完善中。Microsoft 和 Netscape 两大公司对 DHTML 各有自己的看法, 尽管两大公司都使用很多相同的技术术语, 并引用相同的已发表标准的集合, 但具体执行中两者并不完全统一。例如, Netscape 是增加新标记 (主要是 Layer, ilayer 等), 而 Microsoft 的 IE 是对现有标记增加新属性。由此引发的问题是, 有些在 IE 上正常运行的 DHTML, 到 Netscape 浏览器上运行时, 效果可能稍有不同, 反之也是。W3C 目前正在努力开发一种统一的标准使它能够与所有的平台和浏览器相兼容。本章以及随后几章将集中介绍 DHTML 技术。

8.1.2 DHTML 技术

DHTML 不是一种单一的技术, 而是多项技术的综合。在浏览器对象模型(DOM)的基础上, 包括 HTML 4、CSS、事件、脚本等技术。其中 HTML 4、CSS 主要用于控制元素对象

的属性 ; 事件与脚本相结合主要用于控制元素对象的行为。通过这些技术可以改变网页元素的以下几个方面。

- 动态内容(Dynamic content): 动态地增加、删除和修改文本、图形。例如 : 光标通过图片时 , 图片自动进行切换等 ;

- 动态样式(Dynamic styles): 动态改变文本、图片的样式。如改变它们的字体、颜色等。

- 动态定位(Absolute positioning): 将事件、脚本、CSS 等技术相结合可以改变元素的位置。

除此之外 , DHTML 还是一个开放型的系统 , 它将 ActiveX、动态 Firework、Flash、Java Applet 等一些成熟的多媒体技术集成在一起 , 从而把原来静态、呆板的网页变成为一个丰富多彩的艺术品。

值得再一次强调的是 , 通过 DHTML 技术使网页元素由静态转变为动态 , 是依靠浏览器本身的资源来完成的 , 它既没有加重服务器的负担 , 也没有增加网络上的传输容量。

8.2 浏览器对象模型(DOM)

浏览器对象模型 DOM 是英文 Document Object Model 的缩写 , 它是 DHTML 的基础。

什么是 DOM , DOM 是 W3C 制定的标准 , 已被 Internet Explorer 4.0、Netscape 4.0 及以后各种版本的浏览器所接受。按照这个标准 , 浏览器端接受各类网页元素 , 不仅仅只是一种显示格式 , 而是一个个“ 对象 ”。就是说 , 每个网页元素(对象) 都有自己的属性和行为。通过对这些属性和行为的控制可以改变它们的状态和动作。

整个 DOM 是一种由对象组成的层次结构 , 就像一棵倒立的树(树根在上) , 这棵树就称为文档对象模型 DOM。各类浏览器的 DOM 结构稍有不同 , Internet Explorer 4.0 的 DOM 结构如图 8.1 所示。

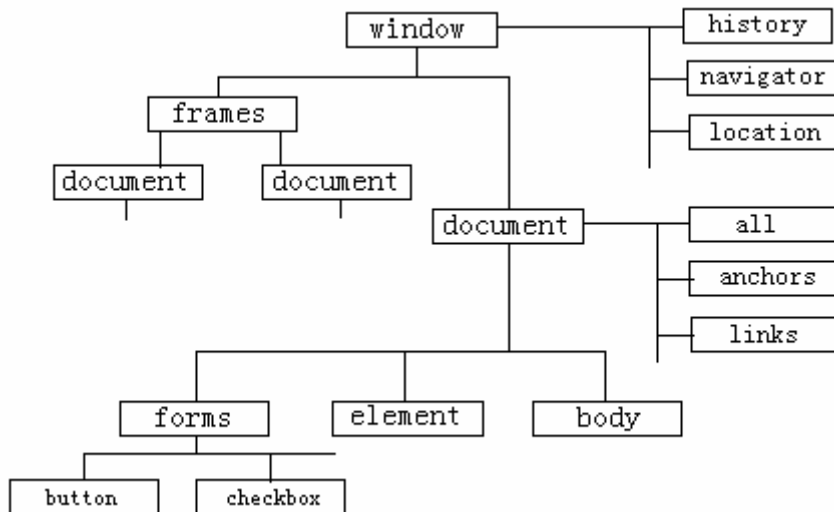


图 8.1 IE 4 的 DOM 结构

在这个层次模型中，每个对象都是它父对象的属性。例如，window 对象是 document 对象的父对象，所以在引用 document 对象时使用 window.document。在这里，document 对象就相当于 window 对象的属性。

对于每一个网页，浏览器都会自动创建 window 对象、document 对象、location 对象、navigator 对象和 history 对象。基于这个层次结构，还可以创建其他对象。对于某个对象的属性，有时需要通过对象的完整路径来引用。例如用 window.document.form1.checkbox1.value 来引用对象 checkbox1 的输出值。其中 form1、checkbox1 都是网页元素名。

window 对象在层次中位于最高一层，具有唯一性，浏览器中的所有内容，包括页面及其他浏览器中设置的信息都存放在 window 对象或者它的子对象中。一般情况下，所有的脚本操作都是假定在当前窗口中进行的，所以调用 window 对象的方法时或者引用 window 的属性时，可以省略 window 对象的引用。例如 window.alert()，可以简写成为 alert()，window.document.write() 可以简写为 document.write()。上面的引用语句也可改写为：

```
document.form1.checkbox1.value
```

在能够明确辨认的前提下，还允许作进一步省略，写成：

```
form1.checkbox1.value
```

下面分别介绍 window 对象、document 对象、location 对象的属性和方法。

8.2.1 window 对象

只要浏览器窗口打开，即使还没有加载任何页面，也会自动建立起 window 对象。

1. window 对象的属性

closed、opener

通过 window 对象的 closed 属性可以判断一个窗口是否已经被关闭。在很多情况下，这种判断是必要的，因为如果一个窗口已被关闭，需要完成一些不同的操作。而如果一个窗口是通过 open() 方法打开的，那么，在 opener 中存放的是打开它的父窗口，通过 opener 属性可以来操纵它的父窗口。当一个窗口打开另一个窗口后，子窗口只能通过 opener 属性和父窗口发生联系；父窗口只能通过 open() 方法的返回值和子窗口发生联系，这两个相关窗口之间就这样实现“互操作”。

例如，通过 opener 属性可以获取父窗口的属性信息，下面的语句可以显示父窗口的名称。

```
alert(opener.name);
```

下面的语句可以判断一个窗口的父窗口是否已经被用户关闭：

```
if (window.opener.closed) {
    父窗口已经被关闭，进行相应的处理
}
else {
    父窗口还没有被关闭，进行相应的处理
}
```

defaultStatus、status

defaultStatus 属性的值是在浏览器窗口下面的状态栏中缺省的显示信息，status 属性是状态栏中当前显示的信息。

例如，将浏览器状态栏的当前信息通过对话框反馈给用户时使用以下语句。

```
alert(window.status)
```

document、history、location

window 的 document 属性、history 属性、location 属性就是浏览器对象的 document 对象、history 对象、location 对象。两者是一致的。

2. window 对象的方法

open(页面地址，窗口名称，窗口风格) 和 close()

使用 open()方法能够打开一个窗口，并且还能指定窗口的显示风格。open()方法将返回对窗口的引用，包含 3 个参数：页面地址、窗口名称、窗口风格。

在窗口的风格中可以用 yes 或 no 来指定浏览器是否拥有工具栏(toolbar)、地址栏(location)、目录图标(directories)、状态栏(status)、菜单条(menubar)、滚动条(scrollbars)等。在窗口的风格中还可以指定窗口的宽(width)和高(height)。

例如，打开一个窗口，网页名是 index.htm，窗口高 300，宽 400，有工具栏、地址栏、目录图标、没有状态栏、菜单条和滚动条。其代码如下：

```
var w_window = window.open ("index.htm", null, " height=300, width=400,toolbar = yes, location = yes, directories = yes, status = no, menubar = no, scrollbars = no")
```

使用 close()方法可以关闭一个窗口，例如对于上一个窗口可以使用以下语句关闭。

```
w_window.close();
```

alert(字符串)、confirm(字符串)、prompt(字符串，缺省值)

在 JavaScript 编写的脚本中，将常使用这 3 种方法和用户进行交互。其中，

- alert()方法用于弹出一个警告框，在警告框中显示字符串。
- confirm()方法用于弹出一确认框，在确认框中显示字符串。例如：

```
if (confirm("Are you sure to submit ?")) {
```

```
    执行表单的提交数据工作
```

```
}
```

■ prompt()方法用来弹出一输入框，框内有缺省值，输入框上面有提示用的字符串。例如：

```
var n_name = prompt("Type in your name please: ", "wang");
```

语句中“Type in your name please”是输入框上面的提示；“wang”是缺省值。当在输入框中输入新数据并且单击“OK”按钮时，将把输入的值赋给变量 n_name。

8.2.2 document 对象

document 对象代表当前整个网页，在 document 对象中存储着当前页面的一些信息，包括页面的前景颜色和背景颜色，也包括页面中的表单、锚标、图片等对象。通过 document 对象，还可以向页面中动态添加文本以及各种标签。

1. document 对象的属性

document 对象的属性非常丰富，下面分类简要介绍如下。

bgColor、fgColor、linkColor、alinkColor、vlinkColor 属性

这些都是对颜色的设置。其中 bgColor 是网页背景的颜色，fgColor 是网页内文本(前景)

的颜色, linkColor 是超链接字符串的颜色, alinkColor 是鼠标单击超链接时的颜色, vlinkColor 是已经访问过的超链接的颜色。

这里的颜色都要以 0xrrggbb 形式表示。其中 0x 代表 16 进制, rr 代表红色的深浅程度, gg 代表绿色的深浅程度, bb 代表蓝色的深浅程度。其范围都从 00 ~ ff。其他颜色都由这 3 种颜色组合而成。例如设置红色时用 0xff0000 表示。

title 属性

title 是页面标题, 也就是在<HTML>中<TITLE>标签中设置的标题, 可以使用以下代码设置它。例如:

```
document.title = "Shopbag";
```

anchors、applets、forms、images、frames 属性

这些属性是用来存放对象的数组。网页中所有超链接都存放在 anchors[] 数组中, 所有 Java Applet 都存放在 applets[] 数组中, 所有表单都存放在 forms[] 数组中, 所有图片都存放在 images[] 数组中, 所有框架都存放在 frames[] 数组中。

利用数组可以引用数组内的对象。例如用 document.forms[0]来引用第一个表单对象。

2. document 对象的方法

write()和 writeln()方法

write()方法用于向网页内写入文本或者标签。writeln()方法的作用与 write()相同, 不同之处是后面包括一个换行符(回车)。不过这个换行符 HTML 并不认识, 要想在 HTML 中换行, 还需要用 write()或 writeln()方法写入
这个 HTML 中的换行标签。

open()、close()方法

open()方法用于打开一个新文档, close()方法用来关闭当前文档。通常情况下要求使用 write()或 writeln()方法前先用 open()方法打开文档, 写完以后用 close()方法关闭, 以保护文档的安全, 例如:

```
document.open();
document.write("Dreamweaver UltraDev"+ "<BR>");
document.close();
```

虽然不写 open()语句, JavaScript 在执行 write()或者 writeln()方法时, 也会自动打开文档。close()语句则是应该写的。

8.2.3 location 对象

在 JavaScript 中, 当前浏览器访问页面的 URL 地址存放在 location 对象中, 使用 location 对象, 可以对这个 URL 进行分析, 并将浏览器引导到指定地址。

1. location 对象的属性

假定有一个这样的 URL 地址:

```
http://www.myset.com.cn:80/welcom/index.htm#section2?id=3
```

其中协议名称为 http, 主机名为 www.myset.com.cn, 端口号为 80, 页面地址为 welcom/index.htm, 在这个网页内有一个锚标, 名称为 section2。使用 location 对象能够分析这个 URL 地址的各个部分。

protocol 属性

protocol 属性指明了通信的协议。在 web 中通常都是采用 http 协议，除此以外，还有 ftp 协议和 gopher 协议。

host 属性

host 属性指明了页面所在 web 服务器的主机名称，可以是域名，也可以是 IP 地址。

port 属性

port 属性指明了服务器通信的端口号，一般在 URL 中不注明，缺省情况下这个端口号是 80。

pathname 属性

pathname 属性指明了页面在服务器上的路径以及页面文件的名称。

hash 属性

如果包括页面内跳转的锚标，在 URL 地址中将包括锚标，以便跳转到指定的部分。此时可以利用 hash 属性获得页面跳转的锚标的信息。

search 属性

在 URL 的后面问号后面常有一些信息（如上例中的“?id=3”）。这个信息是提交到服务器上进行搜索的信息。

例如，现在用 location 对象来分解上述 URL。

```
<html>
<head>
<script LANGUAGE="JavaScript">
<!--
document.write("protocol: "+location.protocol+"<BR>"); // 将得到 http
document.write("host: "+location.host+"<BR>");           // 将得到 www.myset.com.cn
document.write("port: "+location.port+"<BR>");           // 将得到 80
document.write("pathname: "+location.pathname+"<BR>");
                                                    // 将得到 www.myset.com.cn:80
document.write("hash: "+location.hash+"<BR>");           // 将得到 #section2
document.write("search: "+location.search+"<BR>");       // 将得到 id=3
//→
</script>
</head>
<body>
</body>
</html>
```

hostname 属性

hostname 属性将 host 属性与 port 属性结合在一起，既包括主机名，又包括主机端口号，主机名与端口之间用冒号（:）分隔。

href 属性

href 属性提供整个 URL 地址，这个属性将上面几个属性信息结合在一起。

2. location 对象的方法

使用 location 对象的方法能够对页面进行刷新 ,或者将页面引导到另外一个 URL 地址中去。

assign(URL 地址)方法

利用 assign()方法可以将页面引导到另一个 URL 地址上去。例如：

```
location.assign("http://www.mysete2.com.cn/index.htm")
```

将页面引导到 www.myset2.com.cn/index.htm 中去。

reload()方法

reload()方法用于对网页全面刷新。例如定义一个按钮，其功能就是对网页进行刷新。

```
<INPUT TYPE="button" VALUE="RELOAD" OnClick="location.reload()">
```

replace(URL 地址)方法

replace() 方法可以用新的 URL 地址取代当前页面 ,它与 assign()方法不同之处在于 assign()是将页面导航到另外一个页面，使用“后退”按钮时还能返回原来的页面，而使用 replace()方法则是用另一个页面取代当前页面，不能用“后退”按钮返回。

8.3 CSS 级联样式表

8.3.1 概述

级联样式表“CSS”是英文 Cascading Style Sheet 的缩写。它是由 W3C(World Wide Web Consortium) 组织创建，现已为各种类型的浏览器所接受，成为共同使用的标准。

什么是样式表，样式表就是一组用来控制网页元素的外观的属性。使用样式表可以控制某些指定的文本格式，若使用外部样式表时，还可以一次控制一篇或若干篇文档的格式。下面是一个简单的嵌入式样式表。

```
<html>
<head>
<title>
  本页的标题
</title>
<style type="text/css">  // 从这里开始嵌入 CSS 样式表
<!--
H1 {font: 16pt; color: red}
H2 {font: 12pt}
P {font-weight: bold;color:green }
-->
</style>                // 结束样式表
</head>
<body>
  开始本页内容
</body>
</html>
```

在这一段代码中，从<style...>到</style>标签之间就是 CSS，其中“H1”是 HTML 对一号标题的标签，这里 CSS 将它重新定义为字号 16pt，红色；将原来 HTML 的标签“H2”重新定义为字号 12pt；将“P”的属性上“粗字体，绿色”的定义。其中标记“<!-- ... -->”是注释的标记，放在这里的作用是，当有的类型或版本的浏览器不支持<style>...</style>样式表时将忽略样式表的内容，而不会将 CSS 的定义当成字符串显示在浏览器中。

HTML 是一种标记语言，它本身已经规定了浏览器中网页元素的显示格式，为什么还需要 CSS 呢？这是因为随着 Internet 应用范围的扩大，特别是动态网页的出现，已暴露出 HTML 定义属性时的明显不足。比如用 HTML 定义的文本标题，不仅等级少而且跨度大，很难进行微量调整；另外，HTML 给元素定位的能力差，只能利用表格进行粗略定位，而表格通常使文档变得大而复杂。特别是，浏览器在下载表格时，只能在表格代码全部下载完毕后才能将表中的内容一道显示出来。如果表格大而网络的传输速度又不快时，在表格下载的过程中用户可能早已失去了耐心；另外，用 HTML 定义的属性不容易通过脚本进行控制等。

正是在这种背景下，一种新的技术 CSS 出现了，这种技术能够应用于 Netscape 和 Internet Explorer 等浏览器的 4.0 以及以后的版本中（3.0 版本只能部分支持 CSS 技术）。

可以把 CSS 看成是对 HTML 功能的扩展。在 HTML 的基础上，它提供了精确定位或重新定义 HTML 属性的功能。用 CSS 可以控制大多数传统的文本属性，如字体、字型、字号、字距、字重等，还可以控制如页边距、缩排、颜色、底图等排版属性。

使用 CSS 的好处如下。

（1）CSS 提供了丰富的属性可以用来对网页进行精细加工，以进一步美化由 HTML 定义的网页。

（2）一个 CSS 样式表可以用来定义多个 HTML 文本，只要改变 CSS 文件，多个 HTML 文档的显示都将随同改变。反过来，同一个 HTML 文本，用不同的 CSS 样式表定义时，将以不同的风格显示。CSS 使得 HTML 的文本内容与显示风格分离，从而增强了 HTML 文本的可重用性。

（3）CSS 提供的属性能用脚本语言控制，为实现动态网页奠定了基础。

（4）CSS 与 XML 结合能表现出更为复杂的形式。

在 HTML 网页中使用 CSS 的方法有两种：一种是调用外部的 CSS 文件；另一种是在网页中自己定义 CSS。

8.3.2 调用外部 CSS 文件

CSS 的外部文件，是一个以“.css”作为扩展名的文件，该文件中包括一组显示属性的定义。为了调用外部 CSS 文件，应该首先建立与外部 CSS 文件的链接。

链接外部 CSS 文件的方法如下。

（1）选择主菜单的 Window>CSS Styles 项，打开 CSS Styles 管理窗口，如图 8.2 所示。

（2）单击“Attach Style Sheet”（链接外部样式表）按钮，然后在弹出的文件目录中选择外部样式表文件，并且在系统的提示下将该文件拷贝到站点目录下。此时 CSS Styles 窗口中已经列出了该样式表定义的标签。用鼠标选择其中之一，然后单击“Edit Style Sheet”按钮，将弹出 Edit Style Sheet 窗口，如图 8.3 所示。如果再单击窗口中的“Link”按钮时，将提示输入外部文件的 URL。



图 8.2 CSS Styles 管理窗口

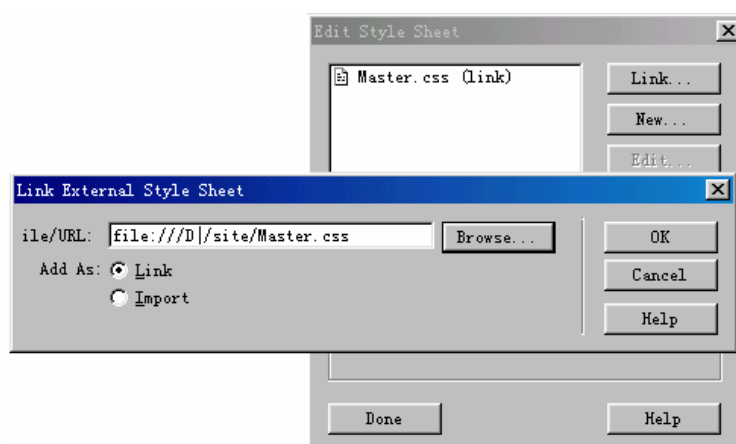


图 8.3 链接外部 CSS 文件

直接键入或通过 Browse 键的帮助选择.css 文件及其路径，并选择下方的 Link 或 Import 选项后单击“ OK ”按钮。

Link 与 Import 选项都能够起链接作用，但使用 Link 选项能适用更多的浏览器。

查看一下源代码就会发现系统已自动加上了链接的语句。

```
<html>
<head>
<title>
  本页的标题
</title>
<link Rel="stylesheet" HREF="URL 地址" >
</head>
```

(3) 如果需要对外部样式表文件进一步编辑和修改时，应该在 Edit Style Sheet 窗口中，先选择文件，然后单击“ Edit ”按钮，将弹出如图 8.4 所示对话框。

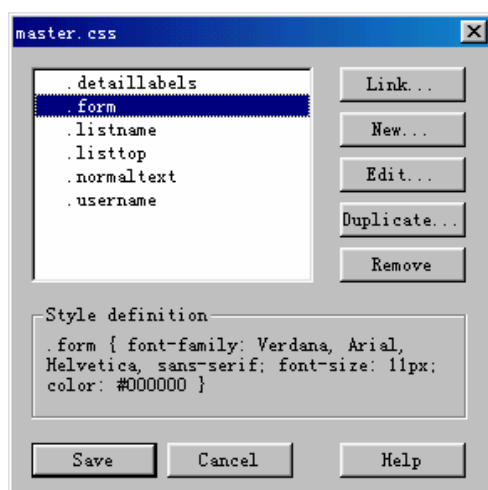


图 8.4 编辑外部 CSS 文件

窗口上方的框体中列出来的是已经定义的标签名,选取其中之一时,下面将列出来为这个标签定义的属性,必要时可按 Edit 按钮对这些定义重新进行编辑和修改,具体编辑方法请参看下一个问题。

使用外部 CSS 样式表定义的属性时,先选择网页中的相关段落,然后单击 CSS Styles 窗口中的标签名,该段字符即按照该标签名定义的属性显示。

8.3.3 在网页中自己定义 CSS

CSS 的属性语句多达数百种,记忆和直接书写起来都十分困难。Dreamweaver UltraDev 为设计者提供的可视化界面,大大简化了 CSS 的定义工作。下面介绍利用 Dreamweaver UltraDev 提供的工具定义 CSS 的方法,先用鼠标单击菜单 Window|CSS Styles 项进入 CSS 管理窗口;单击“New Style”按钮,将弹出 New Style 对话框,如图 8.5 所示。

窗口中提供了 3 种选择:

- 自定义样式表(Make Custom Style)
- 重新定义 HTML 标记(Redefine HTML Tag)
- 使用 CSS 选择器(Use CSS Selector)

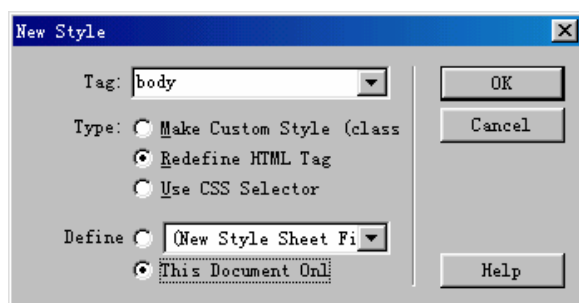


图 8.5 自定义 CSS

下面分别讲述这 3 种方法，先讲第二种方法：

1. 重新定义 HTML 标记

在这里，只能对原有的 HTML 标记重新定义而不能在这以外创建新的标记，定义的内容也不能与原标记完全不同。比如 h1 是对标题的定义，你可以重新规定它的字号、字体、颜色等，但不能让它显示一幅图片。定义的方法如下。

先从 Tag 的下拉菜单中选择需重新定义的标记，如图 8.5 中选择了 body 标签；Define 的选项包括两项：选择定义 CSS 的文件，或者是只在本页中定义，这里选择了后者。单击“OK”按钮，将弹出如图 8.6 所示的对话框。

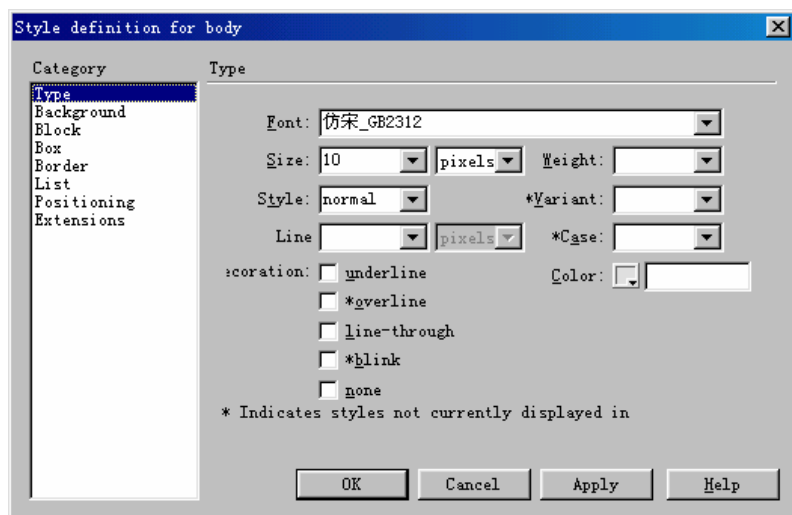


图 8.6 重新定义 Tag 标记

窗口左边的 Category 窗口中包括 8 个选项，右边是各大选项相对应的属性。8 大选项分别如下。

[注意] :样式表的某些属性虽然可以在 Dreamweaver 中设置，但是它们在 Microsoft Internet Explorer 4.0 和 Netscape Navigator 4.0 中显示的外观可能不同。有的属性到目前为止，还不被任何浏览器所支持。

- Type：分类及字体属性。
- Background：背景属性。
- Block：方框属性。
- Box：块属性。
- Border：边框属性。
- List：列表属性。
- Positioning：定位属性。
- Extensions：扩展属性。

选择左边 Category 列表框中的 Type 项后，右边将出现相关的字体、字号、风格、颜色等选项。第一次选择字体时，在 Font 的下拉菜单中可能找不到汉字的选项，此时应先选用 Edit Font List 项，再从弹出的 Edit Font List 窗口中将汉字的字体增加进来。增添汉字类型的列表如图

8.7 所示。

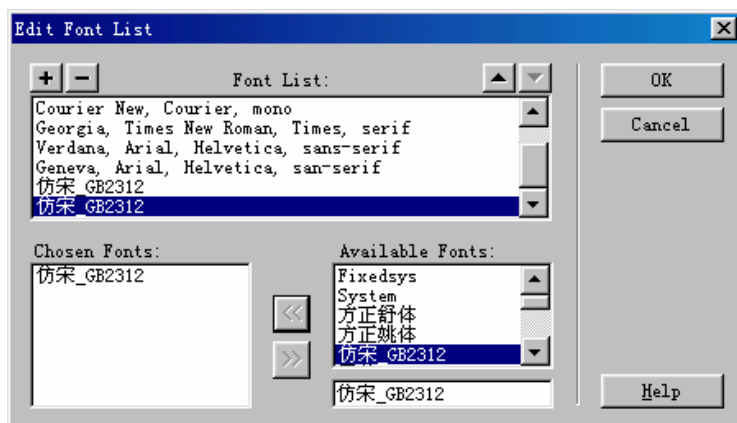


图 8.7 增添汉字类型

设置属性时，需经过反复修改、尝试，直到满意为止。

[注意]：重新定义 HTML 标记改变了网页的显示风格时，可能会影响到网页的布局，因此应注意对布局作适当的调整。

2. 自定义样式表

如果只需要定义个别样式，也许不需要使用自定义样式表的方法，但是当有一批需要定义样式的元素具有很多共性时，使用自定义样式表就比较方便。

定义样式表时，先在 HTML 文档的<head>部分为元素的共性定义一个类，并为这个类起一个名字。然后将这个类名以属性的形式挂接到指定的段落上，这个段落就会按照类定义的属性显示。

具体步骤如下。

(1) 选择菜单 Window|CSS Styles 项进入 CSS 管理窗口，单击“New Style”按钮将弹出如图 8.8 所示的对话框。

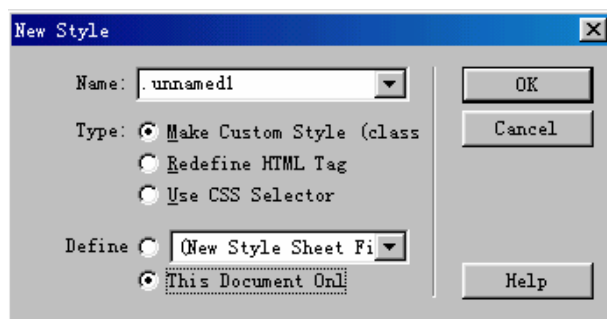


图 8.8 新样式选择

选择“Make Custom Style”(自定义样式表项)，此时窗口中的 Name 下拉表框中已经为新类起了一个名字，将它改名为 newCSS，再单击“OK”按钮将进入设置属性的窗口，当设置完属性后再看看 HTML 的源文件，将发现在<head>...</head>的中间已经增添了以下语句：

```
.newCSS { font-family: "仿宋_GB2312"; font-size: 16px; color: #CC3300}
```

其中“ .newCSS ”是类名,类名前面都有一个句号,表明这是一个 CSS 的类,如果你在填写类名时没有标明这个句号,系统会自动加上;后面大括号中的内容是为该类设置的属性。

(2) 在网页中选择相应的字段后,单击 CSS Style 窗口的类名,该字段即按照上面的定义显示,此时 HTML 源代码是:

```
<span class="newCSS">这是自定义 CSS 的演示</span>
```

系统自动在字段前后加上了...,并在 span 的后面加上了 class="newCSS"属性。

3. 使用 CSS 选择器

使用 CSS 选择器可以定义特殊标签的组合或所有包括指定属性的标签格式,可以定义任何有效的选择器条件,也可以从 Selector 的下拉列表中选择选择器。下拉列表中有文本超连接的样式定义,它的选择器包括:

- a: active: 超链接字符串的显示属性。
- a: hover: 鼠标经过超链接字符串时的显示属性。
- a: link: 单击超链接字符串时的显示属性。
- a: visited: 链接浏览过的网页的字符串的显示属性。

将不同情况下超链接的字符串设置成不同的样式,有利于区别和增加动感。设置这些字符串的样式的方法与前面相同。

缺省情况下,用于链接的字符串是有下划线的,如果想取消下划线时,只需在图 8.6 的对话框下方的 Decoration 中选择 none 项即可。

8.3.4 将 CSS 语句直接插入 HTML 标签的属性中

如果想为某个已定义的标记作些特殊处理,可以用手工方法在该标记中直接插入 CSS 定义。例如:

```
<h2 style="font-size: 9pt; ">...</h2>
```

或

```
<p style="color: red">...</p>
```

前者是将 h2 标记重新定义按 9pt 字号显示,而其他的 h2 仍按原设置显示。后者定义本段的字符采用红色。

前面介绍了多种使用 CSS 样式表的方法,一个 HTML 文本允许同时使用这几种方法。如果多种 CSS 的定义之间出现了矛盾和冲突时,系统将如何处理呢?

应该明确的是,上述定义方法的优先级别是不同的。其规律是:定义的范围越小时,它的优先级越高。直接插入 HTML 标签的 CSS 定义的优先级最高,其次是定义某个段落的 CSS...,外部 CSS 文件的定义优先级最低。这就是“级联样式表”中“级联”的含义。级联在这里就是“优先级”的意思。

8.3.5 将 CSS 样式转化为 HTML 标签

如果网页中用到了图层(Layers)及级联样式表(CSS),又想在不支持这些技术的 3.0 版本的浏览器中使用,可以将上述设定转换成 HTML 标签。方法如下。

(1) 选择菜单 File|Convert|3.0 Browser Compatible 选项，将弹出如图 8.9 所示的界面。

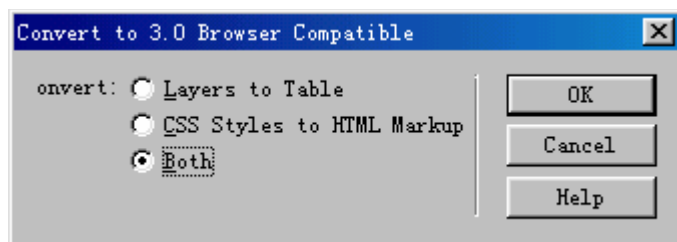


图 8.9 CSS 转换成 HTML

(2) 选择 Layers to Table(将图层转换为表格) 或 CSS Styles to HTML Markup(将 CSS 转换为 HTML 标签) 或者 Both(两种都需转换) 后，单击“ OK ”按钮。

(3) Dreamweaver 将尽可能多地转换成 HTML 标签，并在一个新窗口中打开转换后的文件。

第 9 章 JavaScript 语言基础

JavaScript 是一种功能强大而又使用比较简单的脚本语言,可以直接嵌入到 HTML 页面中和 HTML 紧密地结合在一起。和 VBScript 一样,它也是开发 Web 应用程序理想的语言。JavaScript 既可在浏览器又可在服务器端解释执行,而当前的主要浏览器 Netscape Navigator 完全支持 JavaScript 语言。Internet Explorer 也支持一种与 JavaScript 极为近似的 JScript 语言。因此 JavaScript 的应用范围更为广泛。Dreamweaver UltraDev 提供的内部脚本绝大部分都是用 JavaScript 语言编写的。

9.1 JavaScript 语言简介

JavaScript 是一种基于面向对象(Object Based) 和事件驱动(Event Driver) 的跨平台的脚本语言,目前已经被各大公司所接受,成为一种共同的使用标准。JavaScript 语言提供了非常丰富的内部对象、函数和方法,为创建动态网页提供了强有力的支持。

Dreamweaver UltraDev 系统中提供了大量用 JavaScript 语言编写的内部函数,直接调用这些函数能够快速开发各类 Web 应用程序,这是系统的一大特点和优点。然而,世界上的事情是复杂的,需要也是多方面的,系统提供的函数没有也不可能满足千差万别的不同需要。作为一名 Web 的开发者,一方面应学会调用系统内部函数的方法,充分发挥系统功能的作用;另一方面还要能在必要时修改这些函数或者自己编写新函数来满足实际的需要。这一切都要求设计者掌握 JavaScript 语言。

9.1.1 JavaScript 语言的组成

通常情况下,JavaScript 语言不是用来编写大型、复杂的应用程序,而是用来编写脚本(一种小型程序)的,为客户与页面之间或者客户与服务器之间建立起实时、动态、交互的关系服务。

当前 JavaScript 语言已经发展成 3 个组成部分。

■ JavaScript 的核心部分。JavaScript 的核心部分是语言的核心,在其中定义的语法,是在客户端和服务端都要遵循的基本语法。这些语法包括:操作符、语句、函数以及一些 JavaScript 的内置对象,如 Array 对象、Date 对象和 Math 对象等。JavaScript 的核心部分当前已经相当成熟。在 Netscape 公司和欧洲计算机厂商协会 ECMA(European Computer Manufacturers Association) 及其他公司的共同努力下,在 JavaScript 核心部分的基础上已经制定了一个标准化的、国际化的编程语言,名称为 ECMAScript,该标准已经取得了国际标准化组织 ISO(International Organization for Standards) 的承认,并得到了广泛的应用。

■ JavaScript 在客户端的扩展。JavaScript 在核心语言的基础上扩展了对浏览器对象和文档对象模型 DOM 的控制。客户端下载文档后利用这些控制，使网页中的元素“动”起来，并且能够与客户机的操作（如移动鼠标、按动按钮等）产生交互。

■ JavaScript 在服务器端的扩展：JavaScript 在核心语言的基础上扩展了对服务器端对象的调用，JavaScript 语言与 ASP、JSP、ADO 等对象相结合可以实现与关系数据库互连、对服务器文件进行控制、在应用程序之间交换信息等。在服务器端运行的 JavaScript 应用程序将 JavaScript 核心语言部分和 JavaScript 服务器端的扩展结合起来，可以实现和 CGI（一种外接的服务器端应用程序）同样的功能，不同之处在于 JavaScript 服务器端的扩展仍然和 HTML 页面结合在一起，其运行效率更高、更加易于开发和维护。

9.1.2 JavaScript 语言的特点

概括起来说，JavaScript 语言有以下几个特点。

（1）基于面向对象。将数据和方法组合在一起，封装成对象，程序则由一个个对象组成，这些是面向对象程序设计的基本思想。JavaScript 本身内置了一些基本对象，可以通过它们的属性和方法去调用它们。除此而外客户端运行的 JavaScript 脚本还可以调用浏览器提供的对象，服务器端运行的脚本可以调用服务器提供的对象。这是 JavaScript 功能强大的根本原因。但是 JavaScript 不能利用继承关系创建自己的类，它还不具备完全的面向对象语言的特点，只能是基于面向对象的语言。

（2）跨平台。不论是 Windows、Macintosh 或者 Unix 操作系统，只要用户使用的浏览器支持 JavaScript，JavaScripts 代码就能够在其中正确运行。

（3）易学易用。JavaScript 是一种易学易用的脚本语言，从语言的结构上看，与 C、C++、Pascal、Java 类似，但开发过程比较简单，而且是逐条解释执行的。

JavaScript 语句可以直接嵌入到 HTML 文本中，因此可以使用 Dreamweaver UltraDev 提供的 HTML 编辑器或者其他各类文本编辑器进行编辑。其客户端运行的源代码发送到客户端前不需经过编译，而以字符代码的形式传送到客户端，由浏览器解释执行；服务器端的脚本则先在服务器端运行，将结果转换成 HTML 语言的形式再传向浏览器。

（4）良好的安全性。从服务器端最多可以在浏览器的临时缓冲区中建立和访问 Cookies 临时变量。除此以外，绝对不允许 JavaScript 访问客户的硬盘，以防止对客户端有意、无意的破坏。以保证客户的安全。

9.1.3 JavaScript 与 Java 语言

JavaScript 与 Java 的语法都与 C 和 C++ 相似，且都具有平台无关性，但它们却是两种不同的语言，它们之间的区别是很大的。

Java 是由 Sun 公司开发，是一种编译型语言，必须首先通过 Java 编译器，将源程序编译成 class，从而创建 Java Applet 或 Java Application。Java Applet 是用于 Web 的小型应用程序，在浏览器端下载后运行，Java Applet 在页面中占有自己的一块区域，与 HTML 页面的其他部分分开，Java Applet 在自己这块区域内执行任务；而 JavaScript 是由 Netscape 公司开发的一种解释型语言，它将源代码直接写入 HTML 页面中，并不需要预先编译，运行时也不占用独立的空间，而是对 HTML 页面中的对象进行控制，在页面中完成它的功能。

[注意]：Dreamweaver UltraDev 也能在页面上插入 Java Applet。

从语法风格来看，JavaScript 与 Java 确实有很多相似之处，但 JavaScript 比 Java 要自由得多。Java 是一种强类型语言，语法十分严谨，变量使用前必须声明其类型，在编译中进行强制性的类型检查。而 JavaScript 中，变量不一定要进行声明，在声明中也不一定需要指定数据类型，运行中也不进行强制性的类型检查。例如，允许在脚本中，先给一个变量赋整数类型的值，以后再给它赋字符串类型的值。

Java 是一种基于类（class）的完全的面向对象的语言，在 Java 编程时需要先定义类，定义类的属性和方法，使用时必须创建类的实例（对象）。而 JavaScript 是一种基于面向对象的语言，编程时不区分类和实例，也不需要关心对象的所有属性和方法。

9.2 JavaScript 的基本语法

JavaScript 的语法与 C 或 C++ 语言近似。其特点如下。

- 命令、函数名、变量名都区分大小写。
- 每条语句后用分号结尾。但是 JavaScript 对这一点不像 C 语言那样严格，用 JavaScript 编写脚本时，语句后可以加分号也可以不加分号。建议加上分号，以养成良好的编程习惯。
- 每个过程都是一个函数。
- 语句块（多条相关的语句）用大括号界定。
- 单行注释前用双斜杠（//）表示，多行注释用 /*...*/ 界定。

和 C 语言相比，最大的不同点是，由于 JavaScript 是一个比较松散的语言，变量使用前不一定要进行声明，即使声明时也不一定指定数据类型，运行中也不进行强制性的类型检查。这样的约定既有好处也有缺点。好处是可以节省声明的时间，缺点是一旦发生了错误，不容易找到问题所在。好在 JavaScript 通常只用于编写小型程序（脚本），因此这个缺点带来的后果并不严重。只是在处理数值与字符串表达式时，应记住一条原则：数值加数值结果仍然是数值，但数值与字符串相加时，数值将自动转换为字符串，然后与其他字符串拼接起来。

例如：

```
var x = 10;
var y = 30;
var s = "string";
```

`x + y + s` 结果将是 "40string"。因为两个数值相加在前面，因此先取两值的和再与字符串拼接。

`s + x + y` 的结果将是 "string1030"。因为开始是字符串与数值之和，已转变成字符串，再与最后的数值拼接。

9.2.1 JavaScript 的数据类型

JavaScript 的变量区分大小写。就是说 name、NAME、Name 是 3 个不同的变量。变量名可用字母或下划线（_）开头，但不能用数字 0~9 作为变量名的开头。变量名的第二个以及以后的字符可以用数字、字符或下划线。

JavaScript 的数据类型有以下几种。

(1) 整数型 (Integer) 通过 3 种形式来表示。

■ 十进制 (decimal) : dd...d 其中 d 为 0~9 的整数,但不能用 0 开头。

■ 十六进制 (hexadecimal) : 0xhh...h, 其中 h 为 0-9、A、B、C、D、E、F 的十六进制数。

■ 八进制数 (octal) : 0oo...o, 其中 o 为 0~7 的整数。

(2) 浮点数 (Float) 有两种表示方法。

■ dddd.ddd : 小数点表示法。小数点左侧为整数部分,右边为小数部分。

■ dddd.dddE+ddd 或者 dddd.dddE-ddd : 科学表示法。把一个数用 E (或 e) 分隔开,左侧为真数,右侧为指数 (以 10 为底)。例如 :

566.34E12=566.344*10¹² 最后两个字 12 要向上提半格

345E-10=345*10⁻¹⁰ 与上同-10 要向上提半格

-56E4=-568104 与上同 4 要向上提半格

(3) 布尔值 (Boolean) : 只有两种值, true 和 false。

(4) 字符串 (String) : 用双 (单) 引号括起来的 0~n 个字符。

例如 :

m_message = "Happy New Year! "

就是将 Happy New Year! 字符串赋给了变量 m_message。

■ 特殊字符 : 用一些特殊字符 (Special Characters) 来表示某些控制码, 如表 9.1 所示。

表 9.1 特殊字符及其含义

特殊字符	含 义
"\b"	退格 (backspace)
"\f"	换页 (form feed)
"\n"	换行 (new line character)
"\r"	回车 (carriage return)
"\t"	跳过个 8 空格 (tab character)

■ 空字符 : 空字符 Null 是一个特别的值, 代表空值 (null value)。

9.2.2 JavaScript 的运算符

(1) 基本运算符 : 加 ' + '、减 ' - '、乘 ' * '、除 ' / '。

(2) 余数运算符 : ' % '。例如, 18%4 结果是 2。

(3) 递增(递减)运算符 : ++x、x++、--y、y--等。

++或--代表变量本身 ' 加一 ' 或 ' 减一 ' 的操作。这两个符号可以放在变量名的前面, 也可以放在变量名的后面, 由于两种方法执行的顺序不同, 结果有时是不一样的。例如 :

x = 20;

y = x++;

结果先将 x 赋值给 y, 然后 x 本身再加一。其结果是 y = 20 ; x = 21;

x = 20;
y = ++x;

结果是 x 先加 1，然后再赋给 y，结果 y= 21; x= 21;
(4) 位逻辑运算符：位逻辑运算符(bitwise) 有 3 种，如表 9.2 所示。

表 9.2 位逻辑运算符

运算符	用 法	说 明
&	X & Y	将 X 与 Y 对应的位作 AND 运算
	X Y	将 X 与 Y 对应的位作 OR 运算
^	X ^ Y	将 X 与 Y 对应的位作 XOR 运算

(5) 移位运算符：移位(bitwise shift) 运算符(<<、>>、>>>) 的用法如表 9.3 所示。

表 9.3 移位运算

运算符	用 法	说 明
<< (Left shift)	x<<y	将 x 左移 y 位，右侧补 0
>>(Sign propogating right shift)	x>>y	将 x 右移 y 位，左侧用原来的首位补充
>>> (Zero-fill right shift)	x>>>y	将 x 右移 y 位，左侧补 0

例如：
9<<3 是 1001 左移 3 位变成 1001000，等于 72
9>>>3 是 1001 右移 3 位变成 0001，等于 1
(6) 逻辑运算符：逻辑(logical) 运算符有 3 种：&&、||、!。它们的用法如表 9.4 所示。

表 9.4 逻辑运算符

运算符	用 法	说 明
&&	X && Y	X 与 Y 作 AND 逻辑运算
	X Y	X 与 Y 作 OR 逻辑运算
!	!X	对 X 作 NOT 逻辑运算

逻辑与(AND) 的运算如表 9.5 所示。

表 9.5 逻辑与的运算

AND	TRUE	FALSE
TRUE	TRUE	FALSE
FALSE	FALSE	FALSE

逻辑或(OR) 的运算如表 9.6 所示。

表 9.6 逻辑或的运算

OR	TRUE	FALSE
TRUE	TRUE	TRUE
FALSE	TRUE	FALSE

逻辑否(NOT) 的真值表如表 9.7 所示。

表 9.7 逻辑否的真值表

NOT	TRUE	FALSE
	FALSE	TRUE

(7) 比较运算符包括等于、大于、大于等于、小于、小于等于、不等于 6 种，其用法如表 9.8 所示。

表 9.8 比较运算符

运算符	用 法	说 明
==	x==y	x 等 y 时为 true 否则为 false
>	x>y	x 大于 y 时为 true 否则为 false
>=	x>=y	x 大于或等于 y 时为 true 否则为 false
<	x<y	x 小于 y 时为 true 否则为 false
<=	x<=y	x 小于或等于 y 时为 true 否则为 false
!=	x!=y	x 不等于 y 时为 true 否则为 false

(8) 字符串运算符：字符串运算符中“ + ”和“ += ”可作为字符串的拼接符号。例如：
"I am" + " a student" 结果是 "I am a student"

再如：

```
stt = "I learn JavaScript";
stt += " very well";
结果 stt="I learn JavaScript very well";
```

(9) 其他操作符。

条件运算符

格式如下。

```
结果 = (条件) ? value1 : value2;
// 假定条件为真时结果为 value1，否则结果为 value2。
```

例如：

```
Result = (score>60)? "Pass" : "NoPass";
// 代表 score>60 时，Result = "Pass"，否则 Result = "NoPass"
```

new 操作符

new 操作符可以为用户自定义对象类型或者为 JavaScript 的内置对象创建一个实例。例如，可以用 new 操作符创建一个日期类型的对象。

```
var d_Date = new Date();
```

delete 操作符

```
delete 操作符可以删除对象、对象的属性或者数组中的元素。如：
var c_computer = new Array("mouse","keyboard","CPU","harddisk");
```

```
delete c_computer[3];
```

上述第一句创建了一个数组，包括 4 个元素。第二句删除了其中第四个元素，从此再不能够访问该元素。

this 操作符

JavaScript 利用 this 操作符来引用当前的对象。

9.2.3 运算符的优先顺序

运算符的优先顺序从高到低排列如下。

- 括号()。
- 求补(~) 增量(++) 减量(--)。
- 乘法(*) 除法(/) 余数运算(%)。
- 加法(+) 减法(-)。
- 移位运算<<、>>、>>>。
- 比较运算<、<=、>、>=。
- 相等比较==、!=。
- 位逻辑的与(AND) 运算&。
- 位逻辑的异或(XOR) 运算。
- 位逻辑的或(OR) 运算|。
- 逻辑与(AND) 运算&&。
- 逻辑或(OR) 运算||。
- 条件运算?:。
- 赋值及复合运算+=、-=、*=、/=、%=、<<=、>>=、>>>=、&=、^=、!=。
- 逗号运算 ,。

9.2.4 JavaScript 的流程控制

1. 条件语句

if 语句。

```
if ( 条件表达式 )
```

```
{条件为真时执行的语句}
```

```
[else
```

```
{条件为假时执行的语句}]
```

例如：

```
if (score >= 60)
```

```
status = "pass";
```

```
else
```

```
status = "nopass";
```

表示成绩(score) 大于等于 60 时为合格("pass")，否则不合格("nopass")。

switch 语句

```
switch (表达式){
```

```
case 标签 1:
    代码块 1;
    break;
case 标签 2:
    代码块 2;
    break;
...
[default:
    缺省代码块;]
}
```

上述语句的执行过程是，先计算出表达式的结果，然后将结果与标签 1、标签 2...逐个进行比较，找到第一个能够匹配的标签，就执行该标签下的代码块。如果所有的标签均不能匹配时执行缺省代码块。每个代码块后面都有一个 break 语句，它的作用是执行完本代码块后跳过后面的选择。如果没有这一句，执行完本代码块以后还要和后面的标签一一比较，从而降低了程序的执行效率。另外程序中的 default 以及后面的缺省代码块是可选的部分。如果这一部分没有写，而又找不到匹配的标签时，这段程序将什么事也不做。

例如：

```
if (score >= 90)
    level = "A";
else
    if (score >= 80)
        level = "B";
    else
        if (score >= 70)
            level = "C";
        else
            if (score >= 60)
                level = "D";
            else
                level = "F";
```

改用 switch 语句时语句的格式如下。

```
switch (Math.floor(score/10)) {
case 10:
case 9:
    level = "A";
    break;
case 8:
    level = "B";
    break;
```

```
case 7:
    level = "C";
    break;
case 6:
    level = "D";
    break;
default:
    level = "F"
}
```

本段程序中先将分数除以 10 ,然后取整 ,计算时使用了 Math 对象。Math 对象是 JavaScript 的一个内建对象 ,它的 floor 为取整的方法。另外程序前面连续写了 case 10: case 9:两句代表这两种情况都将执行同一代码块(level="A")。

比较上面两段程序可以看出 ,当情况比较复杂时 ,用 switch 语句显得思路更加清晰一些。

2. 循环语句

JavaScript 支持两种循环结构 :一种是 while ,另一种是 for。另外 break 和 continue 语句也可以用于循环语句中。

```
while 循环语句
初始表达式
while (循环条件){
    代码块
    递增表达式
}
```

例如 :

```
var i = 0;
while (i<=9) {
    window.document.write ("How do you do! ");
    i++;
}
```

程序中循环条件是 $i \leq 9$ 。只要满足这个条件就进入循环 ,window.document.write() 代表在浏览器的网页中写出字符串 ,然后变量 i 加一 ,再进行对循环条件的判断 ,这样一共进行 10 次循环才结束 ,在网页中写出十句“ How do you do! ”。

```
do...while 的语句
do {
    代码块
} while (循环条件)
```

do...while 语句在执行顺序上与 while 语句不同。在 while 语句中是先判断再执行 ,而 do...while 语句却是先执行再判断。因此如果开始循环时条件就不成立 ,do...while 语句中的代码块还得执行一次 ;而 while 语句中就将一次也不执行。

例如 :将前面的 while 语句改写为 do...while 时 ,语句如下。

```

var i = 0;
do
{
    window.document.write ("How do you do! ");
    i++;
} while (i<=9)
    for 循环语句
for ([循环变量赋初值]; [循环条件]; [循环变量增量])
    {代码块}

```

执行 for 循环的步骤如下。

- 对循环控制变量赋初值。
- 判断循环条件是否为真，若为真时进入循环，否则退出循环。
- 进入循环后执行代码块。
- 执行控制变量增量表达式。
- 返回第二步以判断循环条件是否为真。

在 for 循环语句中，[循环变量赋初值]; [循环条件]; [循环变量增量]都是可以选择的项目，因而都可以省略。下面几种写法都是合法的。

缺一项：

```

for ( ; 循环条件; 循环变量增值)...           // 缺第一项
for (循环变量赋初值; ; 循环变量增值) ...       // 缺第二项
for (循环变量赋初值; 循环条件; ) ...           // 缺第三项

```

缺两项：

```

for ( ; ; 循环变量增值)...                     // 缺第一、二项
for (循环变量赋初值; ; )...                     // 缺第二、三项
for ( ; 循环条件; )...                           // 缺第一、三项

```

缺三项：

```

for( ; ; )...                                   // 三项都缺

```

按照语法规则，不论省略哪项，原有的分号都不能省略。被省略语句的功能，应该用其他语句完成。

例如三项都缺的时候，语句可以采取以下的形式。

```

...
给循环变量赋初值;
for ( ; ; )
{
    ...
    给循环变量增值;
    if (循环条件不成立) break;
}

```

如果代码块中不改变循环变量,或者虽然改变循环变量但不循环条件进行判断以跳出循环体,程序将进入“死循环”状态。

在循环代码块中使用 break 和 continue 语句的作用是:break 语句用于跳出循环,continue 语句用于跳过 continue 后面的语句进入新一轮循环。

9.3 JavaScript 的函数

9.3.1 JavaScript 的全局函数

JavaScript 中定义了一些全局函数,这些函数不与任何具体对象发生联系,因此称全局函数。全局函数在脚本的编写中很有用处。

1. Eval(字符串)

eval()函数的功能就是执行括弧内的字符串,如果字符串代表着一个表达式,那么函数将对这个表达式进行运算;如果字符串代表一条或多条语句,那么函数就执行这些语句。

例如:

```
var x1 = 4;
var y = "x1 + 5";
eval(y);
```

函数执行结果: y = 9

再如:

```
var x2 = 5;
var y = "alert(x2)";
eval(y);
```

函数执行结果弹出一对话框,对话框中显示 5,即 x2 的值。

2. parseInt(字符串)和 parseFloat(字符串)

parseInt()和 parseFloat()函数的作用是将括弧内的字符串转变为整数和浮点(小数)。网页中使用表单(Form)时,从文本框中取出来的值,都属于字符串类型,若想运算需先将它们转换成整数或浮点类型,此时可利用本函数完成类型转换的工作。

3. split()分离字符串

```
var listArray = stringList.split(",");
```

函数的作用是将字符串(stringList)以逗号作为分隔符,分解成字符数组(listArray),如 listArray[0]、listArray[1]等等。

4. isNaN(表达式)

isNaN()函数用于判断表达式是否是一个数值型的数据,在使用 parseInt(字符串)和 parseFloat(字符串)函数时,如果字符串不代表一个数值,此时两个函数都会返回 NaN(NaN 的意思是 Not a Number,指出不是一个数字)。通常,在使用上述函数后,还需要用 isNaN 函数进行判断字符串是否为数值,如果不是,需作其他处理。例如:

```
var h = "welcom";
var y = parseInt(h);
```

```

if (isNaN (y) )
{
    alert ("不是数字型。");
}
else {
    alert (" 转换成功。");
}

```

根据上述情况，函数运行的结果将显示“不是数字型”。

5. Number(对象)和 String(对象)

Number()和 String()的参数是对象。Number()将对象转换为数值，String()将对象转换为字符串。和 parseInt、parseFloat 函数一样，如果转换失败，将返回 NaN。

6. escape(字符串)和 unescape(字符串)

escape()和 unescape()函数是一对功能相反的函数，它们执行对字符串进行编码和解码的工作。浏览器对 URL 代码采用 ISO-Latin-1 字符串进行编码，此时的空格都用“% ”、汉字都用一种很难识别的代码取代。可以采用 unescape()函数进行解码，还原成原来的样式。相反，也可以用 escape()函数编码使之变成 URL 需要的格式。

9.3.2 JavaScript 的函数

在 JavaScript 中，函数是一种能够完成一定功能的代码块，函数可以在脚本中被事件或其他语句调用，函数执行完以后可以返回值，也可以不返回值，不返回值的函数就相当于其他语言中的过程。

1. 函数定义

函数定义的格式是：

```

function 函数名(参数列表) {
    代码块
}

```

参数列表是可选部分，如果没有参数时使用一个空括号。如果函数需要返回一个值给调用函数时，代码块中应该包括 return 语句。函数允许嵌套，即在一个函数还允许调用其他函数。

2. 函数的调用

函数只有在被调用时才执行。有两种形式的函数调用语句。

- 函数名(参数列表)。
- 变量名= 函数名(参数列表)。

其中第一种形式适合于不返回参数的调用；第二种形式适合于返回参数的调用。

当脚本的代码长度很长时，应该根据功能将代码块分别写入几个函数中，使得每个函数的功能尽量单一，以增加程序的可读性，同时也便于脚本的编写和调试。如果在多个网页中需要使用同一函数时，可以将该函数单独提取出来写成单独的文件。然后在各个网页前面用 Include 语句，将它和网页组合到一起。

例如：share.asp 中写有几个网页共用的函数。则在这些网页源代码中写上以下语句。

```
<!--#include file="share.asp"-->
```

此语句的位置就是该函数插入源代码的位置。

3. 函数调用的举例

下面举例说明函数的调用过程。

例一：嵌入函数

```
<html>
<head><title>示例二</title>
<script LANGUAGE="JavaScript">
function Add(a, b) {
    c=a+b;
    return c;
}
document.write("a=1, b=2, c=a+b=", Add(1,2) );
</script>
</head>
<body>
<br>
This is the simple "Add"function.
</body>
</html>
```

下面的例子用作事件处理器。

例二：单击按钮的事件处理

```
<html>
<head>
<title>示例之二</title>
<script LANGUAGE="JavaScript">
function demo() {
    alert("You just click the button! ");
}
</script>
</head>
<body>
<form>
<Input type="button" value="try Me" onClick="demo()">
<br>
</form>
</body>
</html>
```

9.4 JavaScript 核心语言中的对象

1. 数组对象

JavaScript 中没有数组这个数据类型,但在编写脚本时,数组的强大功能又是不可替代的。在开发脚本时可以使用数组对象来完成数组的功能。JavaScript 的数组对象除了具备其他语言中数组的功能以外,还有许多其他语言不具备的数组方法。

建立数组

建立数组有两种方法。

第一种方法是,建立数组的同时给数组元素赋值。例如建立一个职业(j_job)数组,其中的值包括 worker、teacher、manager、student 四种,可以通过以下方法建立数组。

```
var j_job = new Array("worker","teacher","manager","student");
```

第二种方法是,建立数组时定义数组的长度,以后再给数组元素赋值:

```
var j_job = new Array(4);
```

JavaScript 对于语法的要求并不严格,允许不事先指定数组长度,而在以后的脚本语句中自动确定数组的长度。

访问数组元素

在 JavaScript 中是通过数组名及其下标来访问数组元素的。值得注意的是,数组下标从零开始,因此 j_job[0]、j_job[1]和 j_job[2]代表访问第一个、第二个和第三个数组元素。

数组对象的属性与方法

数组对象有一个属性,就是数组的长度。通过 j_job.length 就可以获得 j_job 的数组长度。数组的方法有很多,下面介绍其中的两个。

第一种为 sort()方法。利用这个方法能够将数组元素进行排序。例如。

```
j_job.sort();
```

排序后元素的排列是: manager、student、teacher、worker。这是根据各个元素的 ASCII 由小到大的排列结果。

第二种为 reverse()方法。利用这个方法可以将数组内的元素顺序颠倒过来,第一个元素与最后一个元素交换,第二个元素与倒数第二个元素交换等等。如果 j_jop 数组中的元素是 worker、teacher、manager、student。则:

```
j_job.reverse();
```

执行后元素的顺序变成了: student、manager、teacher、worker。

2. 字符串对象

在 JavaScript 中,有字符串类型的数据,同时也有字符串对象,它们的定义方法是不同的。当 st 是一个字符串类型的变量时,定义为 st="hello";当 st 是一个字符串对象时,定义为 var st = new String("hello")。

字符串类型变量和字符串对象在使用时并没有很大区别,字符串变量也可以使用字符串对象的属性和方法。使用时,JavaScript 将自动对字符串变量生成一个字符串临时对象,然后调用这个临时对象的属性和方法,调用完成后,自动将这个临时对象删除。

字符串对象有一个属性 length,代表字符串的长度。例如 st = "hello",那么 st.length 的值就是 5。

字符串还有些常用的方法，现在介绍下面 3 种。

toUpperCase()与 toLowerCase()方法

字符串的 toUpperCase()方法将使字符串的字母全部转换为大写，字符串对象的 toLowerCase()方法将使字符串的字母全部转换为小写。由于 JavaScript 是区分字母大小写的，因此有时需将字符串转变为同一类型（大写或小写）后再进行比较。

indexOf(子字符串)方法

此方法的作用是在母字符串中确定子字符串的位置。如果在母串中包含着子串，将返回子串第一个字符在母串中的下标。如果母串中不包含子串时将返回-1。

例如母串 s = "How are you! "，则 s.indexOf("are")将返回 4。注意字符串的下标从 0 开始。

charAt(位置)方法

charAt()方法中有一个位置参数。它的作用是返回字符串中该位置的字符。例如上面的 s 字符串，用 s.charAt(4)将返回字符 a。

substring(位置 1, 位置 2)方法

substring(位置 1, 位置 2)方法的作用是返回字符串中返回从“位置 1”到“位置 2”之间的字符串。返回的字符串中位置 2 的字符不包括在内。例如 s = "How are you! "。

s.substring(8,10)将只返回 yo 两个字符。

3. 日期对象

JavaScript 中，经常要对日期进行处理，但 JavaScript 中没有日期类型，所以就会常用到日期对象。在 JavaScript 中，日期对象中所存储的不仅仅是日期数据，还包括时间数据。在这个对象中，实际存储的是相对于 1970 年 1 月 1 日 0 时 0 分 0 秒的毫秒数，如果对象代表的时间是在 1970 年以前，存储的是负数，1970 年以后的时间对象存储的是正数。

创建日期对象

创建日期对象要用到 new 操作符，创建的一般格式如下：

var 对象名称= new Date(参数)

■ 创建当前的日期和时间的对象。

当没有参数时，可以使得日期对象获得当前的日期和时间，例如：

var d_Date = new Date();

■ 创建指定日期的对象。

在创建日期对象时，可以指定其中的年、月、日。例如：

var d_Date = new Date(2001,9,1);

值得注意的是这里创建的对象并不是 2001 年 9 月 1 日而是 2001 年 10 月 1 日，因为系统使用 0 到 11 分别代表 1 月到 12 月。

只指定年、月、日而不指定时间时，缺省的时间是 0 时 0 分 0 秒。

■ 创建指定日期和时间的对象。

创建日期对象的同时还可以指定日期和时间。例如：

var d_Date = new Date(2001,9,1,12,20,20);

这里生成的日期和时间是：2001 年 10 月 1 日 12 点 20 分 20 秒。

日期对象的方法

获得日期对象的方法如下。

- `getFullYear()`：获得日期对象的年份。
- `getMonth()`：获得日期对象的月份。
- `getDate()`：得日期对象的日期。
- `getHours()`：获得日期对象的小时数。
- `getMinutes()`：获得日期对象的分钟数。
- `getSeconds()`：获得日期对象的秒数。

另外还有一个方法，`getDay()`可以获得星期几。其返回值从 0 ~ 6，代表星期日到星期六。设置日期对象的方法如下。

- `setYear(年份)`：设置日期对象的年份。
- `setMonth(月份)`：设置日期对象的月份。
- `setDate(日期)`：设置日期对象的日期。
- `setHours(小时数)`：设置日期对象的小时数。
- `setMinutes(分钟数)`：设置日期对象的分钟数。
- `setSeconds(秒数)`：设置日期对象的秒数。

另外也有一个方法 `setDay(星期几)`使日期对象处于星期几。

获得与设置时间

JavaScript 可以用 `getTime()`与 `setTime(毫秒数)`方法来获取和设置从 1970 年 1 月 1 日 0 时 0 分 0 秒开始的毫秒数。如果 `d_Date` 日期对象所代表的时间是 1998 年 2 月 1 日 15 时 30 分 27 秒，那么通过 `d_Date.getTime()`方法获得的值是 886318227000。

4. 数学对象

JavaScript 的数学对象提供了许多强大的数学运算功能，完全能够满足编写脚本中对于数学运算的需要。数学对象与前面几种对象不同，它不需要使用 `new` 操作符来创建对象，而可以直接使用 `Math` 来调用数学对象。

数学对象中有许多属性，其中包括 `PI`，代表圆周率的值，运算时可以直接使用 `Math.PI` 来获得圆周率。除此之外，`Math.SQRT2` 代表 2 的平方根、`Math.LN2` 代表 2 的自然对数等等。

`min(值 1,值 2)`与 `max(值 1,值 2)`

`min()`与 `max()`都有两个参数，分别代表两个值。`min()`的功能是返回两个数值中的小者，`max()`是用来返回两个数值中的大者。例如，`min(12,8)`返回 8；`max(12,8)`返回 12。

`round(数值)`、`ceil(数值)`和 `floor(数值)`方法

`round()`、`ceil()`、`floor()`3 个方法都是将参数取整后返回，不同之处是取整的方法不同。`round()`方法是将参数四舍五入后取整，`ceil()`方法是返回大于或等于参数的最小整数，`floor()`方法是返回小于或等于参数的最大整数。例如 `round(3.5)`返回 4，`ceil(3.5)`返回 4，`floor(3.5)`返回 3。

`random()`方法

使用 `random()`方法能够产生从 0 ~ 1 之间的一个随机数，当我们需要用一定范围的数字测试某个项目时，利用它产生数据特别方便。例如想产生 100 个 10 ~ 100 之间的随机数时，可以利用下面的程序。

```
var i;
var ss = new Array(100);
```

```
for (i=0; i<100; i++) {
    ss[i] = Math.round ((100-10) * Math.random()) + 10;
}
```

sqrt(数值)方法

sqrt()方法的功能是返回数值的平方根。例如，要取 2 的平方根有两种形式，一种是运用 Math.SQRT2 属性，另外一种方法就是用 Math.sqrt(2)方法。

abs(数值)方法

abs()方法的功能是返回数值的绝对值。例如，取-98 的绝对值，可以利用 Math.abs(-98)的方法。

除此之外，JavaScript 中可以用 sin(数值)、cos(数值)、tan(数值)方法获取数值的正弦、余弦、正切函数值。利用 asin(数值)、acos(数值)、atan(数值)等反正弦、反余弦、反正切函数值。这里的数值都是代表角度的弧度值。

9.5 在 Dreamweaver Ultradev 中查阅 JavaScript 的方法

为了快速查阅 JavaScript 语言的变量、函数和语法，可以在设计的环境中直接调用帮助文件。方法如下。

(1) 选择菜单 Window|Reference 项，将弹出如图 9.1 所示的界面。

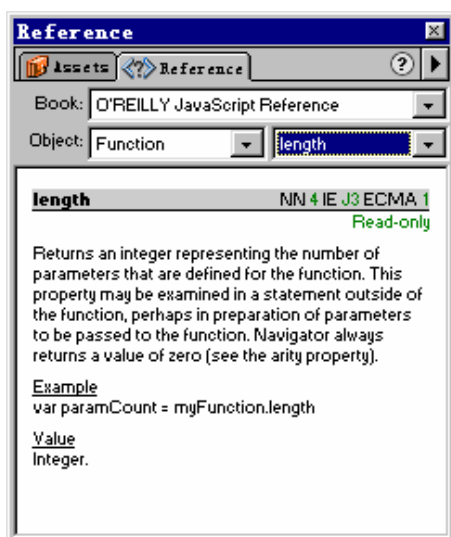


图 9.1 查阅 JavaScript

(2) 在 Book 下拉菜单中选择 O'REILLY JavaScript Reference 项。

(3) 在 Object 中选择查阅的对象，如图中的 Function。

(4) 在右边再进一步选择细项，如图中的 length。

(5) 在列表框中即列出了需要查询函数的作用或示例。

第 10 章 时间轴

动作是时间和空间的函数。大多数页面编辑系统都设有时钟发生器，根据设置的时间间隔周期地发出信号，定时改变图片的位置或状态，以产生动态的效果。Dreamweaver UltraDev 用“时间轴（TimeLine）”代替时钟发生器。在时间轴上可以设置各种不同的时间间隔，既直观又灵活，时间轴还可以直接与图层或图片挂接，定时的改变它们的位置，或者定时触发浏览器端行为以产生动态的效果。下面先介绍时间轴编辑器，然后通过实例介绍图片作直线、曲线或者作复合运动时的设计方法。关于触发浏览器端行为部分将在下一章中介绍。

10.1 时间轴编辑器

选择菜单 Window|Timeline 项，将打开如图 10.1 所示的时间轴编辑器。

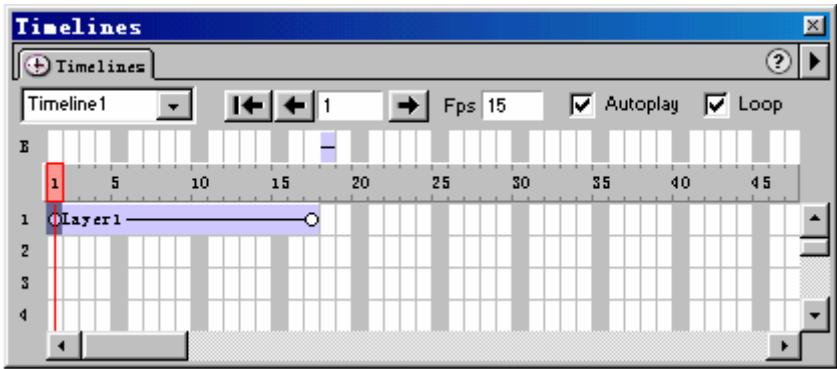


图 10.1 时间轴编辑器

时间轴编辑器中用“帧（Frame）”来形象地表示时间。编辑器中 1、5、10、15、20 等数字就是帧的数字，它代表时间坐标，帧的数字越大代表时间越长。

编辑器上方有一排按钮和数字，可用来编辑动画。这些按钮和数字的含义说明如下。

Timeline1：等时线选择。在 Dreamweaver Ultradev 中可以编辑多条等时线，并在动画过程中切换它们。

1：单击此按钮时返回到第一帧。

←：单击此按钮时退回一帧。

→：单击此按钮时前进一帧。两箭头之间的数字代表当前的帧号。

Fps 15：设定动画的播放速度。这里代表播放的速度是 15 帧/秒。

Autoplay **Loop**：选择 Autoplay 项时，代表打开本页面时，动画自动开始运行；选择 Loop 项时，动画将循环运行。

时间轴中间部分的含义如图 10.2 所示。

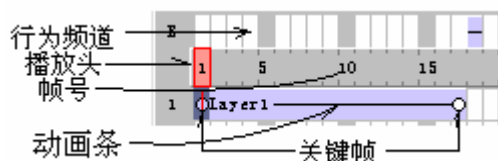


图 10.2 时间轴的部分结构

图中的 Layer1 代表动画条。动画条上的小圆圈为“关键帧”(Key Frame) 标记。播放头指向当前的帧号。

10.2 创建一个直线运动的图片

下面通过创建一个直线运动的图片来了解时间轴的作用。

- (1) 选择主菜单的 Window/Timelines 项，将打开时间轴编辑器。
- (2) 在网页中创建一个图层(Layer1)，图层上放置一图片(这里放置的是一动态 GIF 图片)。
- (3) 用鼠标按住层柄拖入时间轴中，在时间轴编辑器中出现一动画条(Layer1)。
- (4) 将层拖到预计的起点位置，然后选择时间轴的起点，单击鼠标右键选择增添关键帧(Add KeyFrame)项或者按住 Ctrl 键的同时单击动画条的某个位置，把时间的起点与图片的起点位置联系起来。
- (5) 将层拖到预计的终点位置，选择时间轴的终点，再增添一关键帧，如见图 10.3 所示。

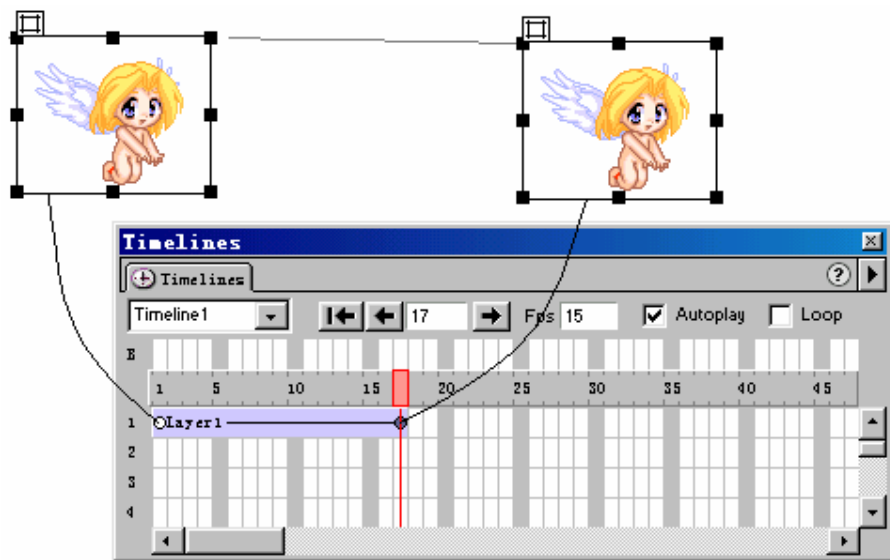


图 10.3 图层的直线运动

- (6) 选择 Autoplay 项。若想循环运动时选择 Loop 项。
- (7) 转入浏览器，即可看见一天使挥动着翅膀飞向右方的运动情况。

10.3 图片的曲线运动

有两种方法设置图片的曲线运动，第一种方法与 10.2 所讲述的方法类似，只不过在两个端点之间多设几个关键帧，每个关键帧对应图层在曲线上的不同位置。

另一种是简便的方法，首先将层移动到起点位置，右击动画条，将弹出一菜单，从中选择“Record Path of Layer”(记忆图层运行的路径)项。然后将鼠标按住层柄按照设想的路线拖动，在拖动过程中，网页上将记忆下层的曲线运动轨迹。此时系统已经根据层的运行情况自动地在动画条的多处位置中设置了相应的关键帧，如图 10.4 所示。

进入浏览器即可看到图层(图片)沿着设想的曲线运行的情况。

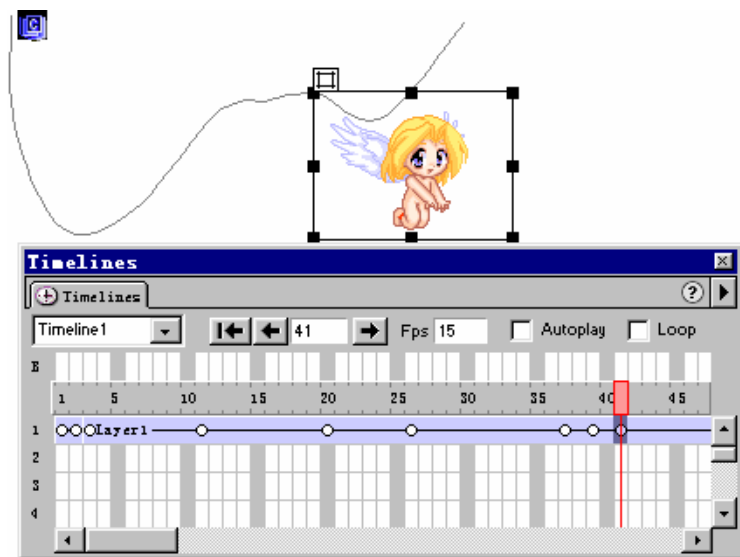


图 10.4 图层的曲线运动

10.4 复合运动

利用时间轴不仅可以组织单项图片的运动，还可以组织多项图片的运动，可以使多个图片同时以不同的方向、不同的方式运动，组成一个非常复杂的运动界面。

下面用一个简单例子来说明组织复合运动的方法。

例如，想在界面上将“迎”、“接”、“挑”、“战”4个字围绕一个中心不断地作圆周运动。为此在界面中创建4个图层，图层上分别写上以上4个字，如图 10.5 所示。

然后每个图层都在时间轴编辑器中建立自己的时间轴 Timeline1、Timeline2、Timeline3、Timeline4。分别按照前面讲述的方法为每一个图层确定运动轨迹。

在 Fps 上设置运行的速度后按住“”按钮作试运行。基本满意时，设置 Autoplay 及 Loop 项以后再到浏览器中观察实际运行的效果。

通过前面的设置可以看出，在复合图片中，各个图片的运动是独立的，依靠时间轴的作用

才将它们同步起来。如果对图片的运动有先后顺序方面的要求时,只需要调整一下时间轴就可以实现。调整时间轴的方法见下一个问题。

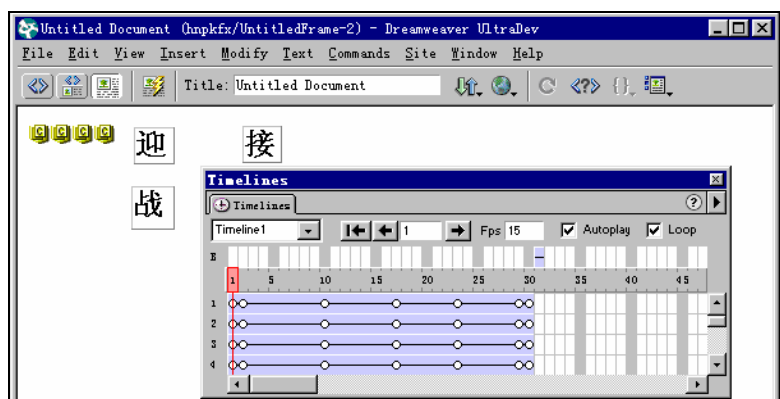


图 10.5 复合图片的设置

10.5 调整时间轴

调整时间轴的方法很多,下面介绍几个常用的方法。

(1) 延长动画的时间:用鼠标拖动动画条结尾帧的标记,所有的关键帧的位置都会向后移动,但是它们之间的相对位置保持不变。如果拖动尾帧时不想让其他帧移动,可在拖动尾帧的同时按下 Ctrl 键。

(2) 改变动画的运行速度:改变 Fps 的值即可改变图层的运行速度。正常的速度为 15 帧/秒。如果设置的数值太大时,由于某些机器的速度跟不上,将可能出现不正常的结果。

(3) 改变动画的开始时间:用鼠标直接左右拖动动画条即可改变动画的开始时间。

(4) 改变整个动画路径的位置:为改变整个动画路径的位置,先选择整个动画条,然后拖动起始帧的位置,此时系统将自动改变其他各关键帧的位置,使得原来的路径作平行移动。

(5) 在时间轴中增添或删除帧:为在时间轴中增添帧时,可以选择菜单 Modify | Timeline | Add Frame; 为在时间轴中删除帧时,先选中需删除的帧,然后选择菜单 Modify | Timeline | Remove Frame 项即可。

10.6 使用时间轴改变图层或图片的属性

使用时间轴不仅可以移动图层(图片),还可以定时改变图片的源文件、可视性、图层的大小以及图层的堆栈顺序等属性。

具体方法如下。

(1) 在要控制改变属性的时间轴中,选择一关键帧。

(2) 选择下列选项之一来改变图层或图片的属性。

- 改变图片的源文件时,在图片的属性面板中利用 Src 右边的文件夹选择新图片。

- 要改变图层的可视性时，在图层的属性面板中 Vis 的下拉菜单中选择新的选项。
 - 要改变图层的大小时，改变图层的 Width 和 Height 属性。
 - 要改变图层的堆栈顺序时在图层的属性面板中改变 Z-Index 的值，或者到图层面板中改变 Z 下面的值。
- (3) 用鼠标单击 Play 按钮预览修改后的情况。

第 11 章 浏览器端行为

11.1 概述

动态网页的重要特点之一就是能够用一定的事件去触发一定的动作，在 Dreamweaver UltraDev 中把这种动作(Action) 和事件(Event) 的结合称之为“ 行为 ”。打开 Dreamweaver UltraDev 的 Window 菜单，可以看到系统中包括有两种类型的行为：一种是“ Behaviors ”，它代表浏览器端行为；另一种是“ Server Behaviors ”，代表服务器端行为。这两种行为在系统中都起着非常重要的作用，但这两种行为的作用不同，调用的方法也各不一样。本章只讨论浏览器端行为，有关服务器端行为部分将在 14 章讨论。

浏览器端行为的作用，主要是使网页中的元素能动态地响应某些事件。例如当用户将光标移动到某张图片上时，onMouseOver 事件引发图片的改变；当用鼠标单击某个按钮时，onClick 事件将打开一个新的窗口；当某个设定的时间到达时，开始或停止播放音乐等等，这些都是浏览器端行为。通过浏览器端行为建立起客户与网页之间的交互关系，使得整个网页变得更有生气。

浏览器端行为是通过脚本实现的。脚本是一种小型程序，需要用 VBScript 或 JavaScript 语言编写程序代码。Dreamweaver UltraDev 系统提供了大量用 JavaScript 语言编写好的脚本。在很多情况下，不必自己去编写代码，而是通过一定的方式调用系统提供的脚本就可以完成需几十甚至上百行代码才能完成的任务，这是系统最有特色的功能之一。缺省安装中系统提供的脚本安装在 Configuration \ Behaviors \ Action 目录下。当然，对于某些特定的行为还是要靠自己编写脚本来实现。

11.2 浏览器行为管理窗口

选择菜单的 Window|Behaviors 选项或者单击窗口右下方的图标都可以打开浏览器行为管理窗口，如图 11.1 所示。

通过这个窗口可以将系统提供的动作与对象的事件连接起来以创建新的行为。

在浏览器行为窗口中：

按钮：用于增添行为；

按钮：用于删除行为；

按钮：用于调整行为的执行顺序。

在 Events 栏下面是事件名。例如图 11.1 中的“ onClick ”就是事件名。

在 Actions 栏下面是动作名。例如图 11.1 中的“ PopUP Message ”就是动作名。

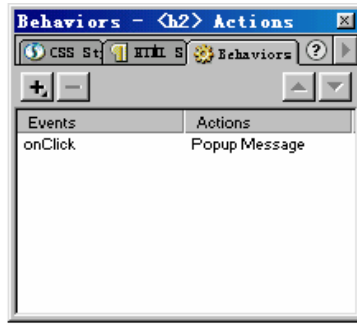


图 11.1 浏览器行为管理窗口

11.3 系统提供的行为

系统提供的行为包括动作(Actions)和事件(Events)两个方面。

11.3.1 系统提供的动作

系统提供的动作包括以下几种。

- Call JavaScript：调用 JavaScript 脚本。这里的脚本需要自己编写。
- Change Property：改变网页元素的属性。
- Check Browser：检查浏览器。
- Check Plugin：检查插件。
- Control Shockwave or Flash：控制 Flash 影像。
- Control Sound：控制声音。
- Display Status Message：在状态栏中显示信息。
- Drag Layer：拖动图层。
- Go To URL：跳至新的 URL 页面。
- Open Browser Window：打开浏览器窗口。
- Popup Message：弹出消息。
- reload Images：预先装入图片。
- Show Hide Layers：显示或隐藏图层。
- Swap Image：切换图片。
- Swap Image Restore：恢复被切换的图片。
- Validate Form：检查表单中的数据。
- Play Timeline：启动时间轴。
- Stop Timeline：停止时间轴运行。
- Go to Timeline Frame：跳转至时间轴的某个指定的“帧”。

一个具体对象并不能支持上面的全部动作，而只能支持其中的一部分。有的对象支持这几项动作，有的对象支持另外几项动作。同一种对象在不同类型的浏览器中，甚至相同类型但版本不同的浏览器中能够支持的动作也各不相同。系统将根据选择的对象及浏览器的类型和版本

显示所支持的动作部分。

11.3.2 系统提供的事件

在 DHTML 中行为是靠事件触发的。什么是事件，事件代表某项事情发生。例如用鼠标在某个图片上单击、双击；某个时间间隔到达；某个元素被装入、卸载；某个元素的状态有了变化等等，这些都是事件，在 DHTML 中往往就是利用事件来调用某个脚本(函数)，触发某个动作去完成一定的任务。

不同类型的对象支持的事件各不相同。同一种对象，浏览器类型不同，甚至同一类型浏览器版本不同时，支持的事件也不完全相同。通常情况下新版本的浏览器支持的事件多于老版本。浏览器类型和版本可在动作或事件的下拉列表中选择，具体方法在下面叙述。

表 11.1 中列出了各类事件，并说明事件适合的浏览器类型以及发生的时机。其中用 NS3 代表 Netscape Navigator 3.0，用 NS4 代表 Netscape Navigator 4.0 及以后的版本，用 IE3 代表 Internet Explorer 3.0，用 IE4 代表 Internet Explorer 4.0 及以后的版本。

表 11.1 事件发生的时机及适用的浏览器类型

事件名	浏览器类型	事件发生时机
onAbort	NS3, NS4, IE4	中断浏览器的运行时发生。例如当装载图片的过程中按动浏览器停止按钮
onAfterUpdate	IE4	网页中与数据源连接的元素已完成数据修改时发生
onBeforeUpdate	IE4	网页中与数据源连接的元素已完成数据修改而且即将失去焦点时发生
onBlur	NS3, NS4, IE3, IE4	元素失去光标焦点时发生
onBounce	IE4	用 Marquee 定义移动的元素到达移动边界时发生
onChange	NS3, NS4, IE3, IE4	网页内元素改变值时发生。例如客户在菜单中选择某项；或者客户改变某个文本域后，将光标移到另一元素时发生
onClick	NS3, HS4, IE3, IE4	客户用鼠标单击某个特定的元素（链接字符串、图形、按钮等）时发生
onDbClick	NS4, IE4	客户用鼠标双击某个特定的元素（链接字符串、图形、按钮等）时发生
onError	NS3, NS4, IE4	浏览器出现错误时发生
onFinish	IE4	用 Marquee 定义的元素完成了循环时发生
onFocus	NS3, NS4, IE3, IE4	与 onBlur 事件相反，当光标聚焦到某个元素时发生
onHelp	IE4	客户单击浏览器的帮助按钮或者选择浏览器的帮助菜单时发生
onKeyDown	NS4, IE4	客户按下键盘上的键的瞬间发生

续表

事件名	浏览器类型	事件发生时机
onKeyPress	NS4, IE4	按下并松开键盘上的键时发生
onKeyUp	NS4, IE4	松开键盘上的键的瞬间发生
onLoad	NS3, NS4, IE3, IE4	图片装载完成时发生
onMouseDown	NS4, IE4	按下鼠标时发生
onMouseMove	IE3, IE4	移动鼠标位置时发生
onMouseOut	NS3, NS4, IE4	鼠标离开某个特定元素时发生。这个特定元素通常是一个超链接
onMouseOver	NS3, NS4, IE3, IE4	鼠标移动到某个特定元素上时发生。这个特定元素通常是一个超链接
onMouseUp	NS4, IE4	压下的鼠标松开时发生
onMove	NS4	窗体 (Window) 或者框架 (Frame) 移动时发生
onReadyStateChange	IE4	某些特定元素的状态有了变化时发生
onReset	NS3, NS4, IE3, IE4	表单 (Form) 中的数据复位(清除)时发生
onResize	NS4, IE4	表单 (Form) 的大小有变化时发生
onRowEnter	IE4	记录指针改变后发生
onRowExit	IE4	记录指针将要改变时发生
onScroll	IE4	网页上下滚动时发生
onSelect	NS3, NS4, IE3, IE4	在字符域中选择字符时发生
onStart	IE4	用 Marquee 定义的元素开始循环时发生
onSubmit	NS3, NS4, IE3, IE4	表单中的数据提交时发生
onUnload	NS3, NS4, IE3, IE4	客户离开网页时发生

11.4 增加浏览器行为的步骤

在浏览器行为窗口中增加行为的步骤可分为： 选择对象； 为对象选择动作； 确定相应的事件 3 大步。具体做法如下。

(1) 先用鼠标单击选择的对象。可以将整个页面作为被选对象，此时单击<body>标签，也可以选择其他某个对象。

(2) 用鼠标单击行为管理窗口中的“ +”按钮，在弹出的动作列表中选择动作。弹出的动作列表如图 11.2 所示。

(3) 先通过 Show Events For 选项选择浏览器类型及版本，再选择需要的动作。

(4) 单击事件(Events)右边的按钮，在下拉列表选取相对应的事件。下拉列表列出的事件如图 11.3 所示。

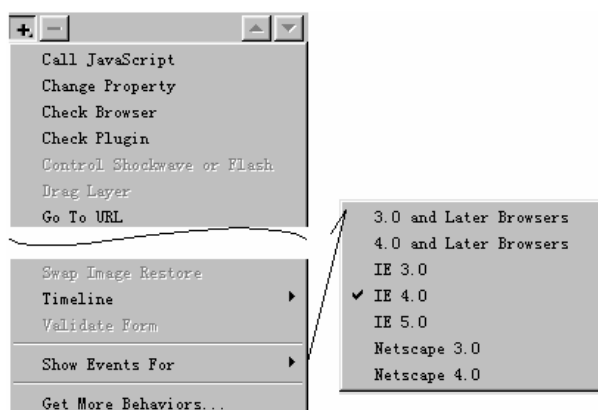


图 11.2 选择浏览器

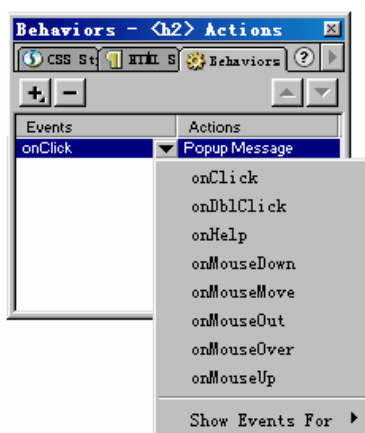


图 11.3 选择相应的事件

若某一事件需要触发多个动作时,可重复前面的设置为该事件设置多个动作项,这时将有多个行为项列在行为管理窗口中。事件触发时,系统将按照它们的排列顺序逐个执行。必要时,可利用窗口右上方的两个图标按钮(“▲”和“▼”)来调整它们的排列顺序。

当需要撤消某项行为时,先选中该行为项,然后单击“-”按钮。

11.5 浏览器行为示例

下面将通过示例说明创建不同类型行为的方法,这些类型中有完全利用系统提供的功能,也有自己编写部分脚本,还有完全由自己编写脚本这几种不同的情况。

11.5.1 弹出提示信息

利用系统提供的 Pop-up Menu 项弹出提示窗口,显示有关信息。具体步骤如下。

(1) 选择网页中的一对象(字符串、图片等),然后打开行为管理窗口。

[注意]：如果选择的是字符串对象，而字符串还没有与其他网页链接时，要在它的 Link 属性中键入“ # ”号。如果已经与其他网页链接时，则不需补充这个设置。

- (2) 单击行为窗口中的“ + ”号按钮，从弹出的动作菜单中选择 Pop-up Menu 项。
- (3) 将提示信息键入弹出的 Message field 小窗口中，如图 11.4 所示。

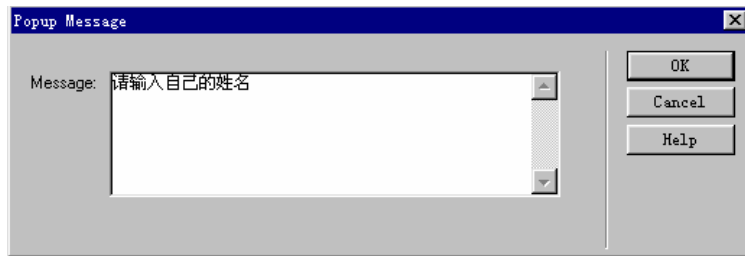


图 11.4 输入提示信息

在 Message 框体中输入需要提示的信息。

- (4) 单击“ OK ”按钮。
- (5) 从 Event 的下拉列表中选择对应的事件。例如选择了 onMouseOver 事件。
- (6) 到浏览器中运行，当鼠标到达原来选择的对象上时，将弹出一信息窗口，提示：“请输入自己的姓名”。

11.5.2 切换图片

利用系统提供的 Swap Image 动作可以在网页中自动切换图片。方法是先在网页中放入一张图片。选择该图片后，再选择 Swap Image 动作项，将弹出如图 11.5 所示的对话框，在 Set Source to 文本框中输入另一张图片的位置（也可以利用 Browser 文件夹图标选择该图片）。

选择 Preload Images 选项时，将预先装入被切换的图片以提高转换速度；选择 Restore Images onMouseOut 项时，鼠标离开图片时将自动恢复原来的图片。

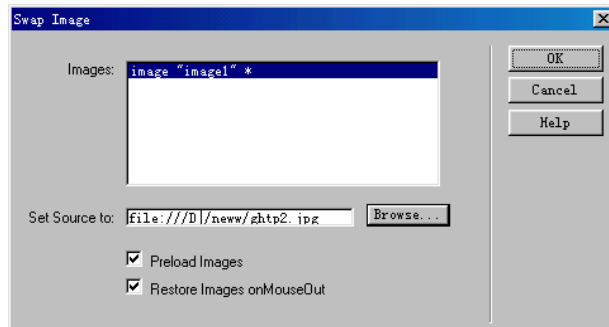


图 11.5 选择切换的图片

11.5.3 信息的浮动显示及动态菜单

1. 信息的浮动显示

为了动态地显示信息，可以利用图层的隐藏/显示特性。先设置多个图层，将不同的信息

分别放在各个图层上,并且将图层拖放到适当的位置,然后隐藏起来。再用事件触发图层在显示与隐藏之间转换。

下面结合一简单示例来说明设计的过程:假设有 3 种音响设备出售,用动态方式显示它们的价格。

(1) 3 种音响设备的图片(如图 11.6 所示)分别放入表格中。



图 11.6 动态显示价格

(2) 将 3 个图层拖到适当位置,分别写上各设备的价格后隐藏起来。

(3) 选中一张图片后,在行为管理窗口中选择动作 Show-Hide Layers 项,将弹出如图 11.7 所示的界面。

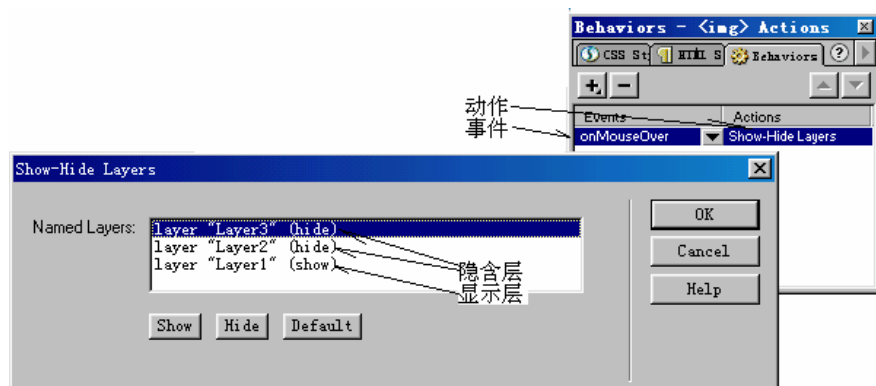


图 11.7 确定图层的显示状态

(4) 确定隐含和显示的图层。

(5) 在行为管理窗口中的 Events 中选择 onMouseOver 事件。

上面设置的结果是,当鼠标移动到该图片上时 onMouseOver 事件发生,某个图层连同图层上的信息显示出来。另外一些图层则隐含起来;当鼠标离开时 onMouseOut 事件发生,又可为此事件设计新的图层显示和隐含的动作。

2. 动态菜单

动态菜单是网页中常用的技术。所谓动态菜单是指使用时才显示,不使用时自动消失的菜单。动态菜单技术使得众多的菜单项有条不紊的出现在屏幕上。

可以使用类似前面信息浮动的方法来实现动态菜单的设计,即将“图层”作为多个菜单项的容器,当鼠标进入图层区时显示图层及菜单;当鼠标离开图层区时,菜单与图层一起自动消失。

下面通过简单的示例来说明创建的要点。

(1) 设置 3 个图层。用其中之一显示标题(不消失)；另一图层做菜单项的容器，鼠标进入标题时显示该图层；第三层为子菜单的容器，当鼠标进入最后一项菜单的选项时自动显示第三层，鼠标退出菜单区时自动消失。

(2) 为了使菜单项中的最后一项打开子菜单，需将最后一项字符串的链接属性(Link)设为“#”。

(3) 分别为下列各对象(图层或菜单项)指定以下行为。

- 鼠标进入标题所在的图层时，onMouseOver 事件发生，显示菜单所在的“图层”。
- 鼠标退出菜单所在的图层时 onMouseOut 事件发生，隐含本菜单所在的图层。
- 菜单中最后一选项的 onMouseOver 事件发生，显示子菜单所在的“图层”。
- 子菜单所在图层的 onMouseOver 事件发生，显示菜单所在的“图层”及子菜单所在的图层。
- 子菜单所在图层的 onMouseOut 事件发生，隐含菜单及子菜单所在的图层。

11.5.4 给时间轴附加行为

给时间轴的某个“帧”附加行为，以便定时触发相应的动作。操作方法如下。

(1) 在时间轴窗口中某个帧的上方行为通道中双击，将弹出如图 11.8 所示的提示框，窗口提示在行为窗口中弹击“+”按钮。



图 11.8 提示增加行为

[注意]：如果选用“Don't show me this message again”以后就不再出现这个提示了。

(2) 到行为窗口中去设置行为和对应的事件。

例如：利用时间轴定时、循环地显示某个广告(标语)。

- 先调入几个“层”，分别在各层中放置不同的图片或标语。
- 选择菜单“Modify/Layers and Hotspots/Align Left(Right,Top,Bottom)”项，使多个层对齐。
- 确定时间间隔，给相应的帧附加行为。方法是用鼠标单击相应帧的上方(B区)，该区将出现一小横杠“-”，表示此处将触发某些行为。

例如选择第 1、15、30、45 帧作为时间间隔，在选定的帧上方点击后，再到 Behavior 管理界面中单击“+”后，选择“Show-Hide Layers”行为项，在弹出的界面上确定哪些层显示，哪些层隐含。

- 在最后一帧中选择 Timeline | Go to Timeline Frame 行为项，在弹出的界面中选择 Go

to Frame N ”作为新循环的起点。再次选择 Timeline|Play Timeline 行为项，重新开始 Timeline 的运行。

- 选择 Timeline 管理器右上角的 Autoplay”及 Loop”项，以保证进入浏览器即启动时间轴并且开始循环。

- 进入浏览器查看运行效果。

11.5.5 在表单中加入行为

下面介绍在表单(Form)中加入 3 种行为的方法，这 3 种行为是：检查文本域输入的合法性、自动导出数据并提交和复位前的最后确认。

1. 检查文本域输入的合法性

动作“ Validate Form ”项可用来检查表单中文本域的内容，以减少输入数据的错误。可以将这个动作与表单区相连，用 onBlur(光标退出输入框) 的事件触发；或者将这个动作与 onSubmit 事件相连，当客户单击 Submit 按钮时被触发。如果被检查出表单区内有非法数据时，将会提示错误并禁止将表单的信息提交给服务器。

例如，表单内放置有 3 个文本域，在行为(Behavior)窗口中单击 + ”按钮后选择 Validate Form 项，将弹出如图 11.9 所示的对话框。

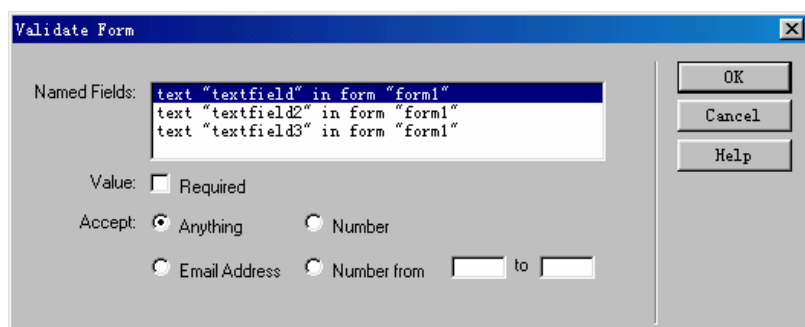


图 11.9 限定文本框的输入特性

分别选取各个文本域并为其输入约束条件。这些条件包括：

- Required：表明此项数据不能空缺。
- 在 Accept 中如果选择 Anything 项，表明输入的数据类型不限；如果选用 Number 项时表明只限于输入数字型数据。

- Email Address：输入时检查是否符合 E-Mail 地址的规定(必须包括“ @ ”符号)。

- Number from..to..：输入数据的允许的范围。

事件被触发时自动按照这里的设置的条件进行检查，如果有不符合要求时提示错误信息。

2. 自动导出数据

表单中的原始数据必须输入，这是没有问题的。但是根据原始数据引导(计算)出来的数据虽然也可用手工输入，但是从可靠性和效率等方面考虑，应该由系统自动导出。

假设表单中放置有 4 个文本域(textfield1...textfield4)，前 3 个文本域用于输入数据，第 4 个文本域用于计算前 3 个输入值之和。为了自动导出这一数据需进行以下操作。

(1) 创建一表单(Form1)，表单内放 4 个文本域(textfield1...textfield4)。

(2) 选中 textfield4 后设置动作项 Call JavaScript，对应的事件为 onFocus (光标聚焦)。动作项 Call JavaScript 实际上只提供了一个框架，具体的脚本仍需要自己编写代码，这样做虽然麻烦一些，但也提供了极大的灵活性，为熟悉 JavaScript 语言的设计者提供了极大的方便。

(3) 在弹出的对话框中可以填入两种形式的内容：程序语句或者函数名。

输入以下程序语句：

```
if (form1.textfield3.value > '3000') {alert ('Number is too big');}
```

此时如果 textfield3 中输入的数据大于 3000 时，将立即进入 textfield4 并弹出提示框，提示：

" Number is too big"

也可以输入函数名。例如像图 11.10 那样输入 calcu ()。

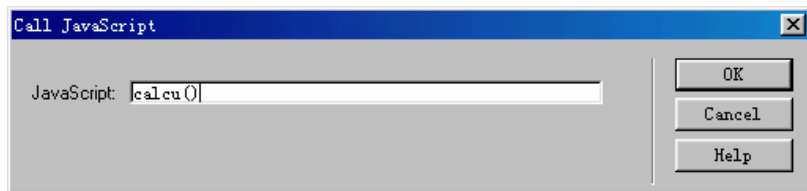


图 11.10 输入函数名

当输入的是函数名时，必须在源代码窗口的<head>...</head>区内写出函数的代码，写入以下代码以满足本例的需要。

```
function calcu () {
var s1,s2,s3,s4;
s1 = eval (form1.textfield1.value);
s2 = eval (form1.textfield2.value);
s3 = eval (form1.textfield3.value);
s4 = s1+s2+s3;
form1.textfield4.value = s4;
}
```

其中，函数 eval () 的作用是返回与字符串算术表达式相应的值。例如：

```
result = eval ("3*5-2");
```

result 为 13

为什么输入程序语句或者输入函数名两种方法都能获得相应的结果呢，这是因为源代码中调用函数的代码是：

```
<Input type="text" name="textfield4" onFocus="MM_callJS('参数') ">
```

被调用的函数代码是：

```
function MM_callJS (jsSTR) {
return eval (jsSTR) }
```

图 11.9 中输入的只是调用语句中的“ 参数 ”部分。如果它是一个语句，将返回该语句的执行结果；如果它是一函数名，则此函数将作为参数传递，先执行该函数，再返回该函数的执行结果。

3. 提交或清除操作的最后确认

为了慎重，有时需要在表单数据提交(或清除)之前，作最后的确认。比如单击“提交”按钮后，提示“真要传送吗？”；单击“复位”按钮后提示“真要复位吗？”。只有在最后确认后下才继续执行上述操作，否则上述操作失效。由于系统没有提供这方面的功能，因此，需直接在网页中书写程序代码，程序代码如下。

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<SCRIPT LANGUAGE="JAVASCRIPT">
<!--
function BeforeReset() {
    confirm("真要清除吗？");
}
function BeforeSubmit() {
    confirm("真要传送吗？");
}
//-->
</SCRIPT>
</head>
<form name="form1" method="post" action="" OnReset="return BeforeReset()" OnSubmit="return
BeforeSubmit()">
```

上面的斜字体部分为直接写入的程序的代码。在网页上运行这段程序时，当填完数据单击 Submit(提交)按钮或 Reset(复位)按钮时，将弹出如图 11.11 所提示的提示信息以最后确定提交或复位的操作。

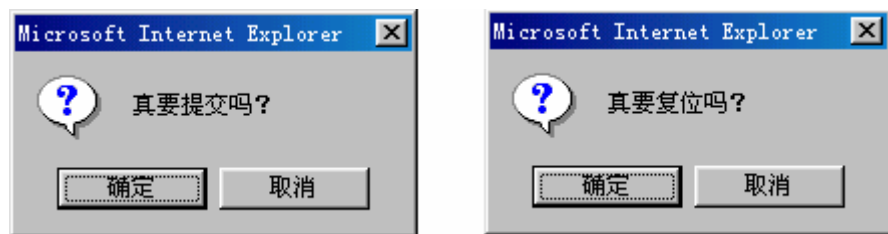


图 11.11 提交/复位的最后确认

11.5.6 对图层的操作

1. 图层的拖动

图层是一种容器，在图层上面可以放置各种文本和图形，对图层的操作实际上是对这些文本或图形的操作。对图层的操作有很多方面，最基本的是使图层能被拖动。在设计阶段，图层是能被任意拖动的。但要使图层在程序运行中也能被拖动，就需要事先加以设定了。设置包括

以下几个方面。

- 拖动的方向：水平方向、垂直方向、任意方向。
- 拖动的目的地以及接近目的地时启动的事件、触发的动作。

下面讲述设置的方法。

(1) 将图层拖进网页中，在图层上放入图片或字符串。

(2) 单击图层，然后选择 Drag Layer 动作项，将弹出如图 11.12 所示的对话框，该对话框包括 Basic/Advanced 两页。在 Basic 中作基本设置，在 Advanced 中作进一步设置。

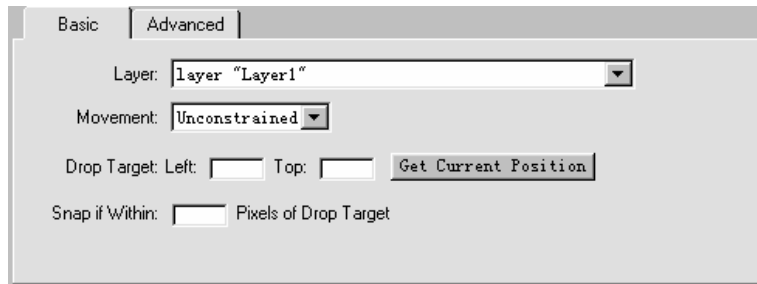


图 11.12 拖动图层的基本设置

图 11.12 是基本设置的界面。

- Layer：代表图层名。
- Movement：代表拖动的方向，包括不限制方向(Unconstrained)和限制方向(Constrained)两种。这里是不限制方向时的界面，限制拖动方向时的界面如图 11.13 所示。
- Drop Target：拖动的目的地。其中 Left 代表左边距，Top 代表上边距。单击 Get Current Position 按钮时可以显示图层的当前位置。
- 在 Snap if Within ...Pixels of Drop Target 中填写数字，当被拖动的图层与目的地的距离等于或小于这个数字时启动一事件触发新的行为。

图 11.13 是限制拖动方向时的界面。主要增加了 Up、Down、Left、Right 等限制数据。

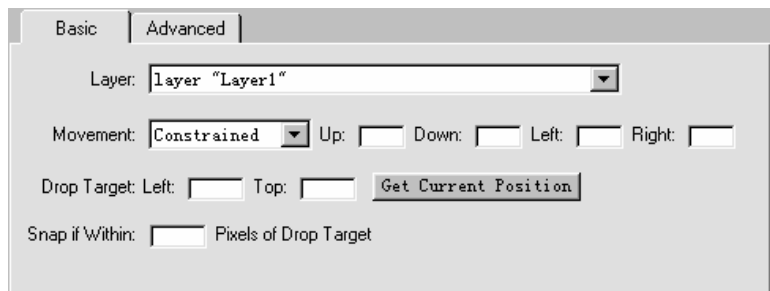


图 11.13 图层拖动的限制范围

- 由于拖动的行为应该事先设置，因此应选择<body>对象的 onLoad 事件。注意不能与 onMouseDown 或 onClick 等事件挂接。

(3) 选择 Advanced 页时将弹出如图 11.14 所示的对话框。在这个对话框中主要由以下参数。

- Drag Handle：设定拖动的手柄是整个图层还是图层中的一部分。

■ 拖动时,被拖动的层出现在最上层,停止拖动时是否恢复原来的层;进入拖动的目的区时调用的 JavaScript 脚本或语句;停止拖动时调用的 JavaScript 脚本或语句。

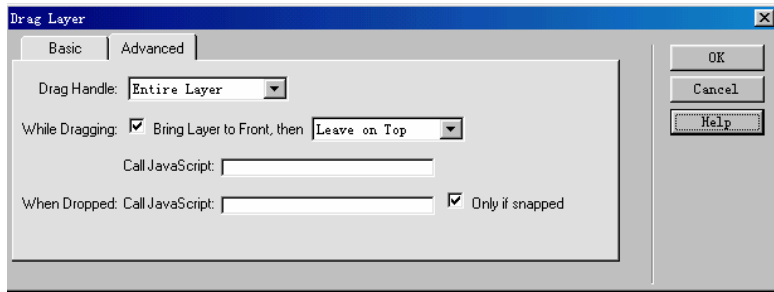


图 11.14 拖动图层的进一步设置

2. 图层的重叠与展开

为了形象地介绍各类商品,可先将各商品的外形、名称以及详细说说的链接点综合到图片中,然后将图片放置到“图层”上,并将这些图层分类重叠放置在网页中(像人们收藏“扑克牌”时一样),需要时利用改变属性(Change Property)的动作,改变各个图层的位置属性将它们展开。实现这一功能的具体方法如下。

(1) 将图片放置到各个层中。

例如将两张图片分别放入两个层(Layer1、Layer2)后的情况如图 11.15 所示。

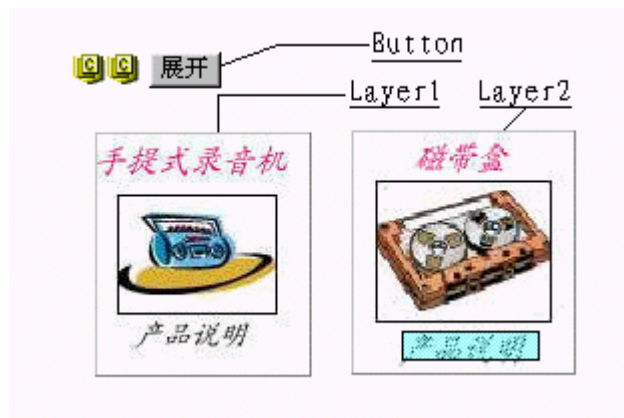


图 11.15 两张图片放入图层中

(2) 将两个图层重叠对齐。

选中两个层 先用鼠标选中一个,按住 Shift 键再用鼠标选择另一个),再通过菜单 Modify | Layers and HotSpots | Align Left (Align Top)项使两个层左边界(上边界)重叠对齐。

(3) 利用行为(Behaviors)展开层。

放入一按钮(Button)并选中该按钮后,选用改变属性(Change Property)动作项,此时将弹出 Change Property 对话框,如图 11.16 所示。

(4) 在对话框中设置以下参数。

- 在 Type of Object 下拉列表中选择 LAYER。
- 在 Named Object 下拉列表中选择 layer “Layer2”。

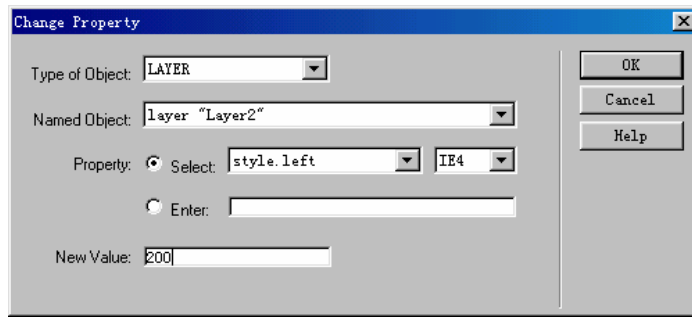


图 11.16 改变图层的位置属性

■ 在属性 Property 中可以利用 Select 的下拉列表选择属性，也可以直接在 Enter 文本框中填入属性名。本例中是从下拉列表中选择的 style.left 属性。在右边下拉列表中选择浏览器类型，它的下拉列表中只包括 NS3、IE3、NS4、IE4 四种类型（本例中指定的是 IE4）。这是因为这里使用的方法是改变 CSS 图层的位置属性。而 Netscape Navigator 和 Internet Explorer 3.0 以前的版本都不支持 CSS。

- 在 New Value 文本框中填入 200。即本行为是将 Layer2 移动到距左边距 200px 的位置。
- 在浏览器运行时按动按钮，两个图层将按照上面设定的参数展开。

第 12 章 网页中插入 Flash

12.1 概述

在 Dreamweaver UltraDev 环境中可以使用多种多媒体技术。例如可以插入 Firework 动画、Flash 电影、Flash 按钮和文本、Java applet、ActiveX 控件等等。综合利用这些多媒体技术，可以将网页设计得更丰富多彩。本章只介绍在网页中插入 Flash 电影、Flash 按钮和文本等方面的问题。

12.2 Flash 简介

Flash 是 Macromedia 公司创建的基于向量的二维图片和动画技术。由于它的文件容量小，因此非常适合于制作 Web 网页或者嵌入网页中。目前 Netscape Navigator 和 Microsoft Internet Explorer 以及大多数浏览器的近期版本都支持 Flash 技术。

Web 设计者使用 Flash 可以创建导航条、动画小图标、较长的动画和同步声音。因为 Flash 电影（又称为 Shockwave Flash 电影）是压缩的矢量图形，所以，它可以在网上迅速传输，并且可以按照浏览器的视窗大小来决定显示的比例。使用 Flash 技术，可以将手工绘图和已存在的画面排列在同一个场景内，然后建立起动画的时序并加入交互性。Flash 既可以制作电影，也可以制作响应一定的鼠标事件的动态按钮和文本。

电影制作完毕以后，可以将其导出为 Flash 播放器电影文件（*.swf 文件），也可以把它嵌入 HTML 页，连同 HTML 页一同送往浏览器去浏览。

Flash 在 Netscape Navigator 浏览器中作为一种插件（Plugin），在 Internet Explorer 中作为 ActiveX 控件使用。当将 Flash 电影（按钮或文本）插入到网页中时，Dreamweaver 使用 Object（适用于 Microsoft 的 ActiveX control）和 EMBED（适合于 Netscape）两种标签来表示，这样做可以适应大多数浏览器的需要。

12.2.1 矢量图与位图

矢量图和位图是计算机显示图形的两种主要方法。这里先简要介绍一下矢量图和位图的基本概念以及它们之间的区别。

1. 矢量图形

矢量图形用包括颜色和位置属性的直线、曲线（称为矢量）和面来描述图片。比如一片树叶包括两个部分。

- 一些形成树叶边缘轮廓的线段。
- 由轮廓线封闭的平面。

树叶的颜色包括轮廓线的颜色及由轮廓线封闭区域的颜色，与轮廓内单独的点无关。可以通过修改描述叶的轮廓线来改变图形的性质，也可以移动、放大、缩小、变形，或者在不改变图形显示质量的前提下，改变矢量的颜色。

矢量图具有独立的分辨率，即它可以在不损失质量的前提下，以不同的分辨率显示在输出设备中。

图 12.1 就是树叶的矢量图。其中左边是图片的全貌，右边是部分放大后的视图。

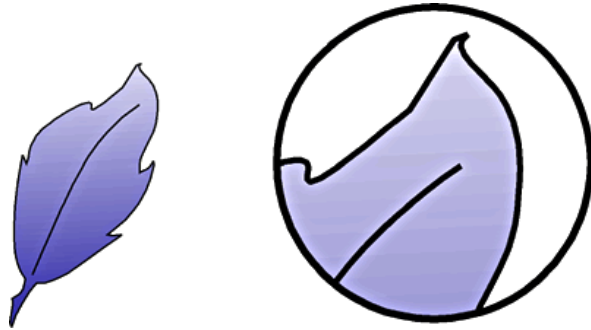


图 12.1 树叶矢量图

2. 位图图像

位图图像是通过对每一个栅格点的不同颜色来描述的，这些栅格点称为像素。例如，上面所说的树叶的图片，可以用所有组成该树叶的像素点的位置和颜色来描述。因为编辑位图图形时，修改的是像素，而不是直线和曲线。因而，不可能通过修改轮廓线来更改树叶的图形性质。

位图图像的分辨率不是独立的，因为描述图片的数据是对特定大小的栅格中的图片而言的，因而，编辑位图图形会改变它的显示质量。尤其是当放大和缩小一个位图图形时，常常会因为像素在栅格中的重新分配而导致图像边缘变得粗糙。当在比位图图形本身的分辨率低的输出设备上显示图片时，图像的显示质量也会降低。图 12.2 就是树叶的位图。其中左边是图片的全貌，右边是部分放大后的视图。

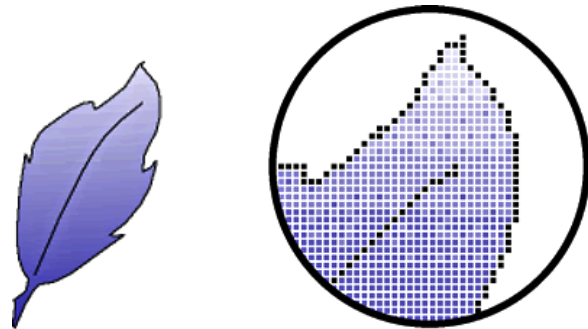


图 12.2 树叶的位图图片

比较两种图片的特点可以看出：位图是使用栅格的点阵来定义图片的，它表现力丰富，可以非常柔和地表现颜色的渐变情况，适合于立体图形的应用。但是这种类型的图片的文档容量通常比较大，而且当图片放大、缩小时，文档质量及文件容量的改变也大。

矢量图用线和面来定义图片，适合于对平面图形的设计。它的表现力虽不及位图那么丰富，

但文档容量却大大小于位图，而且放大缩小图片时，文档的质量和容量变化也不大。因此非常适合于网络上的应用。

12.2.2 Flash 的文件格式

Flash 有源文件、动画文件、模板文件 3 种不同的格式，它们各自的特点分述如下。

(1) Flash 源文件(文件后缀为.flas)。这种格式的文件用于在 Flash 环境中进行编辑和调试，而不能在 Dreamweaver 或者浏览器中打开运行。当编辑完成以后，可以将它输出成 SWF 格式的文件，这时就可以在 Dreamweaver 中预览，并在浏览器中运行了。

(2) Flash 动画文件(文件后缀为.swf)。这是一种压缩后的 Flash 文件格式，它被优化以便在网络上传输和应用。这种格式的文件可以在浏览器中打开、运行，也可以在 Dreamweaver 中预览，但不能在 Flash 环境中编辑，它相当于其他语言的.exe 文件。

(3) Flash 模板文件(文件后缀为.swt)。它可以用于修改和取代 Flash 动画文件中的部分信息。下面将要用到的 Flash 按钮、文本对象使用的就是这种格式文件。在这种文件中可以使用自己的文本或者链接方法来修改模板，从而进一步创建一个自定义的 SWF 文件并插入到 Dreamweaver 的文档中。

在 Dreamweaver/Configuration/Flash Objects/Flash Button(或 Flash Text) 文件夹中可以找到这些模板文件。

12.3 Dreamweaver 中的 Flash 图标

在 Dreamweaver UltraDev 对象模板的 Common 页中有 3 个有关 Flash 的图标，它们可分别用于插入电影、按钮和文本，如图 12.3 所示。

- Insert Flash 图标，用于插入 Flash 电影。
- Insert Flash Button 图标，用于插入 Flash 按钮。
- Insert Flash Text 图标，用于插入 Flash 文本。

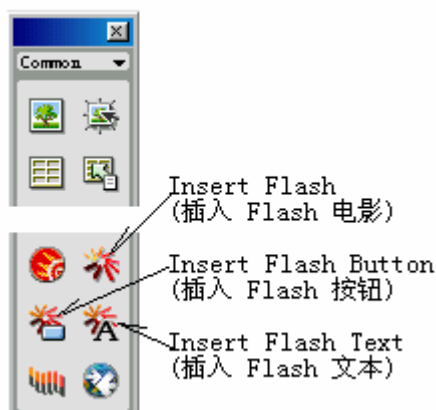


图 12.3 有关 Flash 的图标

12.4 插入 Flash 电影

在 Dreamweaver UltraDev 网页中插入 Flash 电影非常简单,只需选择下面 3 种方法之一即可。

(1) 先将光标放在预计的放置点,然后用鼠标单击对象面板(Object palette)上的 Flash 电影图标。

(2) 用鼠标将对象面板(Object palette)上的 Flash 电影图标拖到文件窗口的适当位置。

(3) 先将光标放在预计的放置点,然后选择菜单的 Insert|Media|Flash 选项。

当完成前面的操作后,通常要求你在弹出的文件窗口中选择相关的源文件。然后到 Flash 的属性面板中去确定参数。Flash 电影的属性面板如图 12.4 所示。

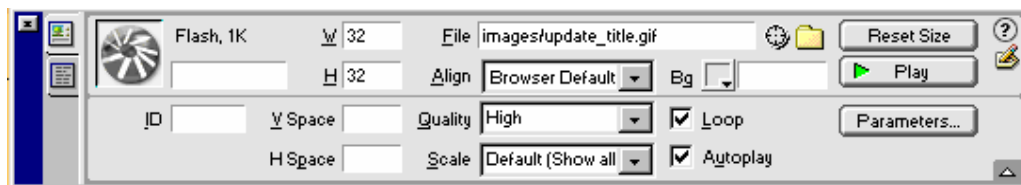


图 12.4 Flash 电影的属性面板

属性面板中的属性说明如下。

- W、H：显示板的宽(W)和高(H)。W 与 H 可以用 pixels 作为单位,也可以用以下单位定义: p(points)、in(inches)、mm(millimeters)、cm(centimeters)、or%(父对象大小的百分比)。注意缩写符号与数值之间不能留空格,例如 3mm 等。

- File：文件名(包括文件的路径)。可以直接输入也可以利用 Browser 文件夹图标选择。

- Align：决定图片在显示板中的对齐方式。

- Bg：Flash 图片的背景颜色。

- ID：标签。只限于作为 ActiveX 控制组件时相互传递信息之用。

- V Space 和 H Space：确定图片与边框之间上下、左右的空间(pixels)。

- Quality：质量参数。可选的参数从低到高包括 Low、Auto Low、Auto High 和 High 几种。质量越高图象越清晰,但容量也越大,网络上传输、下载的时间也越长。

- Scale：缩小或放大的比例。

- Parameters：打开一对话框,以传递参数。

- Reset Size：恢复原来图形的大小。

- Play：单击此按钮时开始在文档窗口显示编辑的效果,此按钮也同时改变为 Stop(停止)按钮。

- Autoplay：进入浏览器后自动运行。

- Loop：循环运行。

12.5 插入 Flash 按钮

Flash 按钮、文本与一般按钮、文本的主要区别是，它能产生一定的动态效果，主要表现在颜色的变化上，当鼠标从上面经过时，按钮或文本的颜色将自动变化。网页中它们可以作为链接组件使用。由于 Flash 按钮、文本、电影都是矢量图形，因此很容易对它们进行放大和缩小的操作，并且可以在显示质量与文件容量之间找到最佳的结合点。

12.5.1 插入 Flash 按钮的步骤

系统提供了几十种按钮的样式以供选择，还允许设计者对这些样式进行一定范围的编辑和修改，例如改变它们的颜色、大小以及按钮上的字符串等。为了插入 Flash 按钮，按照以下步骤操作。

(1) 单击 Flash 按钮的图标，或者用鼠标将图标拖进文档窗口，或者选用菜单 Insert|Interactive Image|Flash Button 项，将弹出图 12.5 的对话框。

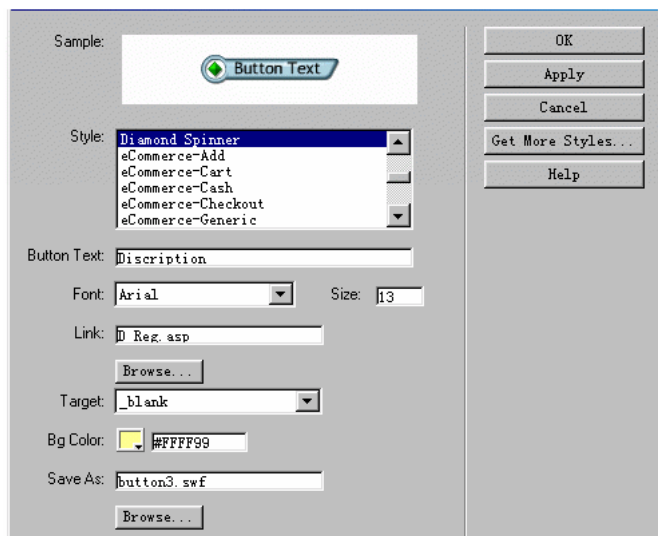


图 12.5 Flash 按钮选择

(2) 从样式 (Style) 的列表中选择按钮的样式。每选中一种样式时，上面就会出现该样式的外形。

(3) 编辑按钮的外形参数。

■ 在 Button Text 文本框中输入按钮上的字符串。当按钮样式上没有字符串时不能设置此项。

[注意]：目前版本中的 Flash 按钮以及文本对汉字的支持还不理想，相信不久后将会改进。

- 在 Font 的下拉列表中选择字体。
 - 在 Size 文本框中输入字体的大小。
 - 单击“BgColor”按钮选择背景颜色或直接在其文本框内输入颜色的代码。
 - 在 Link 文本框中输入链接的文件名，或者借助 Browser 文件夹图标选择链接的文件。
- Flash Button 只能链接网站内部的文件，而不能链接网站以外的文件。

- 在 Target 的下拉列表中确定被链接的文件将要显示的窗口。
- 单击“Apply”按钮后，可以在文档窗口中看到按钮的动态效果（鼠标移到按钮的上方时颜色的变化）；单击“Get More Styles”按钮时将自动转至 Macromedia Exchange Web 网站，在那里可以下载更多的按钮样式。

12.5.2 Flash 的按钮的属性面板

在改变 Flash 按钮参数的同时，将改变 Flash 显示面板以及图形的外貌。Flash Button 的属性面板如图 12.6 所示。

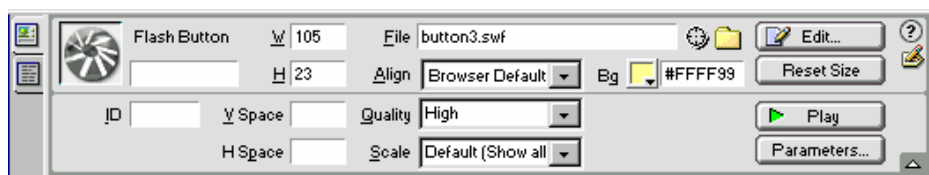


图 12.6 Flash 按钮的属性面板

此属性面板与 Flash 电影的属性面板相似，其中只有微小的差别。这些差别如下。

- 少了 Loop、Autoplay 两项选择。这种差别是显而易见的。因为 Flash 电影要运行，而 Flash 按钮只是属性上作动态地变化。
- 多了 Edit 按钮。单击此按钮时将进入图 12.5 所示的对话框，以便重新进行编辑。而 Flash 电影只能单独到 Flash 环境中去编辑。

当在属性面板中‘改变了按钮的属性时，系统将自动将这些属性的变化转变为对应的参数放置到 Object 和 EMBED 标签中。

12.6 插入 Flash 文本

当需要插入 Flash 文本时，需按照以下的步骤进行。

(1) 单击 Flash 文本图标，或者用鼠标将图标拖进文档窗口，或者选择菜单 Insert|Interactive Image|FlashText 项，将弹出如图 12.7 所示的对话框。

(2) 在对话框中设置 Flash Text 的参数，除了大家已经熟悉的设置以外，这里只介绍几点比较特殊的地方。

- Text：文本。这是本组件的主体部分。[注 1]

[注意]：对外形参数的改变不能在样式窗口中直接显示，只有到文档窗口或浏览器中才可看到。

- Color：文本的颜色。
- BgColor：文本的背景颜色。
- Rollover Color：鼠标经过时切换的颜色。
- Link：在 Link 文本框中输入链接的文件名，或者借助 Browser 文件夹图标选择链接的文件。Flash Text 只能链接网站内部的文件，而不能链接网站以外的文件。

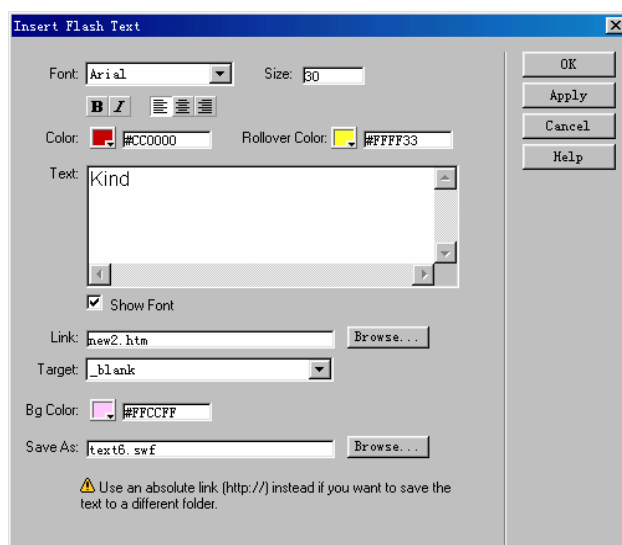


图 12.7 Flash 文本对话框

- Target：在 Target 窗口的下拉列表中确定被链接的文件将要显示的窗口。

[注意]：必须在插入 Flash 按钮和文本之前先保存现有的文档。

第 13 章 素材管理器、模板及库

13.1 素材管理器(Asset management)

13.1.1 素材管理器的作用

素材管理器的作用是使设计者更加容易管理和组织站点内的各类素材。在站点管理器中素材是分散在各个目录下的,而素材管理器却将它们分类集中放置在窗口中,因而查看、对照或者编辑、修改起来都非常方便。在素材管理器中,被管理的素材划分成以下几种类型。

-  Images : 图形文件。包括 GIF、JPEG 或 PNG 格式图形。
-  Colors : 颜色。文档和 CSS 中使用的颜色包括文本颜色、背景颜色和链接颜色。
-  URLs : 站点链接外部的 URL ,包括对 FTP、gopher、HTTP、HTTPS、JavaScript、E-mail (mailto) 以及 local 文件(file://) 的链接。
-  Flash movies : 压缩后的 Flash 可运行电影文件(*.swf)。
-  Shockwave movies : Macromedia 公司的 Shockwave 电影文件。
-  MPEG and QuickTime movies : MPEG 或者 Quick Time 格式的电影文件。
-  Scripts : 单独的 JavaScript 或 VBScript 脚本。不包括嵌入 HTML 文档中的脚本。
-  Templates : 模板文件。
-  Library items : 库文件。

站点中还有一些其他文件虽然有时也称为素材文件,但如果不属于以上范围时,将不会进入素材管理器中。

13.1.2 两种素材的列表方式

选择菜单 Window | Assets 项, 打开如图 13.1 所示的素材管理窗口。

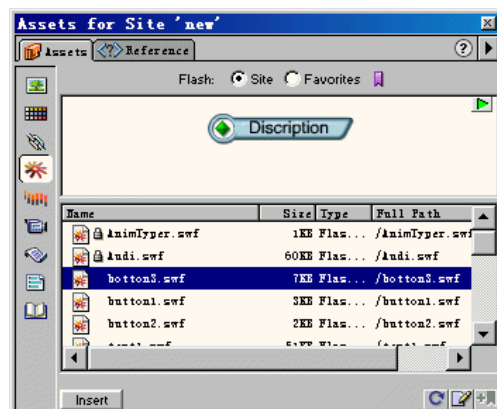


图 13.1 素材管理窗口

[注意]: 打开素材管理器之前应该预先设置好网站的 Cache 项。从菜单 Site | New 的 Local Info 页中可以看到该选项。

主窗口包括上下两部分: 上面为显示部分, 下面为列表部分。表的最上方有 Site 和 Favorites 两种选择, 它代表窗口中有两种列表的方式: 站点素材列表和常用素材列表。

1. 站点素材列表(Site list)

将站点内所有素材分类并按照英文字母顺序排列在窗口中, 称这种排列方式为站点素材列表。从这个列表中或者直接从素材文件的组件中将常用部分抽出来, 可组成常用素材列表。

2. 常用素材列表(Favorites list)

常用素材列表开始时是空的, 选择下面几种方法之一可增加列表的内容。

- 利用 Ctrl 键与光标, 在站点素材列表中选择一项或多项, 然后右击窗口, 在弹出的菜单中选择 Add to Favorites 项。

- 用鼠标右击文档窗口中的素材组件, 在弹出的菜单中选择 Add to Favorites 项。

- 在站点管理器的窗口中用鼠标右击某个素材文件名, 在弹出的菜单中选择 Add to Favorites 项。

13.1.3 管理窗口的使用方法

素材管理窗口的使用方法包括以下几个方面。

1. 查看素材列表

在窗口左方选择素材类型, 上方选择列表类型, 即可查看素材列表。

2. 查看或编辑素材

(1) 单击某项素材, 上面的窗口中将显示该素材的外形。如果属于动态类型, 再单击右上方的  图标, 即可演示动态的状态。

(2) 改变窗口的大小。用鼠标放在两个窗口的交界处, 当鼠标形状改变后再上下拖动。

(3) 编辑素材。单击右下方的  (Edit) 按钮, 即可进入编辑环境。

(4) 将素材插入到文档窗口中。先将光标放置于插入点后选择某项素材再按  (Insert) 按钮。

[注意]: 在站点中增加或删除素材并不能自动反映到素材管理窗口中来。只有按动刷新 “  ” 按钮后才能看到这些变化。

13.2 模板

13.2.1 模板的作用

模板是一种可重用的网页样板, 在模板中既定义了网页共同的表现形式, 又留出了各网页自己的编辑空间。当大量网页使用同一个模板时, 就意味着这些网页在布局上, 在使用的网页元素方面, 使用了相同的显示风格。

模板技术适用于大型网站, 因为一个大型网站不仅需要有个性的主页来吸引客户的注意, 同时也需要有风格一致的子页, 才不至于给客户造成一种 “ 杂乱 ” 的感觉。特别是当由多人组成的小组共同开发子页时, 模板的作用更为重要。此时如果大家使用共同的模板, 就很容

易使得设计出来的网页在显示风格上取得一致。

13.2.2 创建模板

可以从一个已存在的网页中创建模板。其步骤如下。

- (1) 在站点管理器窗口中打开一张准备作为模板样式的网页。
- (2) 选择菜单 File | Save as Template 项，将弹出如图 13.2 所示的对话框。



图 13.2 为模板取名

- (3) 为模板取名。即在 Save As 文本框中填入模板的名字。

- (4) 进入站点管理器，将看见在站点的根目录下出现了一个新的目录“ Templates ”以及该目录下的模板名(本例中将是 published_1.dwt)

[注意]：请不要改变 Templates 目录的位置，不要将模板文件移出这个目录，也不要将非模板文件放到这个目录中来。

到此为止，已经创立了一个模板。但是到目前为止，这个模板还没有任何用处。因为这个模板和原来的网页完全相同，而且不能改变。

真正的模板应该既包括不能变动的部分，也要包括可以改变的部分，以便让各个网页设计出不同的内容。因此还必须对前面设置的模板进行调整。调整的核心是在模板中划分出可编辑部分。

13.2.3 调整模板

调整模板的目的是在模板中划分出可编辑区。方法如下。

- (1) 用鼠标双击网站管理器的模板文件名，以便打开模板。
- (2) 在模板中选取某些文本(图形、表格、图层等)，一次只能选取一个连续的空间。
- (3) 选择主菜单的 Modify|Templates|New Editable Region 项，将弹出对话框要求为可编辑区取名，取名后单击“ OK ”按钮。被选中的区域上将出现浅绿色的编辑区名。现在如果再查看源代码时将可以看见在可编辑区域的前后出现了以下标签。

```
<!--#BeginEditable "编辑区名">
```

```
...
```

```
<!--#EndEditable ->
```

- (4) 如果想删除某个编辑区时，选择菜单 Modify | Templates | Remove Editable Region 项，

然后在弹出的窗口中选择需要删除的编辑区即可。

(5) 如果想设置多个可编辑区时, 重复前面的(2)(3) 步。

(6) 存储修改后的模板, 此时系统将提示是否按照新设置的模板改变原来使用此模板的网页, 如果单击“ Yes ” 按钮, 则所有使用此模板的网页都将按照新的设置改变过来。

13.2.4 利用模板创建网页

利用模板创建网页时, 按以下方法进行。

(1) 选择菜单 File|New from Template 选项, 将弹出对话框, 选择模板名。

(2) 在弹出的模板界面上对可编辑区的内容(或形式) 进行编辑, 其他地方则不能改变。

13.2.5 利用素材管理器编辑模板

为了利用素材管理器编辑模板, 应按照以下步骤进行。

(1) 选择菜单 Window|Templates 项打开素材管理窗口, 窗口上方将显示模板的外观, 下面是模板的列表。

(2) 在素材管理窗口中, 选用以下两种方法之一进行操作。

- 双击模板文件名。

- 先选择模板文件再单击“ Edit ” 按钮。

(3) 在模板中增加、删除、改变可编辑区。

(4) 完成编辑后利用 File|Save 菜单存储。

13.3 库

为了在多张网页之间共享某些网页元素, Dreamweaver UltraDev 允许将重复使用和改变得比较频繁的图像、文本或其他网页元素存入库中成为库项目(Library Item)。需要使用库元素时, 又可将库项目拖放到文档中。此时 Dreamweaver UltraDev 会将网页文档中插入该库项目的 HTML 源代码复制, 并仍然与它保持一种引用关系。一旦修改了库项目后, 再使用菜单 Modify | Library 选项时, 站点内各网页中所有与该库项目相关连的网页元素都将自动进行更新。如果没有用到库, 遇到上述情况时, 将不得不到各个网页中去一一进行修改。

Dreamweaver UltraDev 的库目录(Library) 放在站点的根目录下, 各个库文件都放在库文件目录中。每个站点可以设置不同的库和库文件。

13.3.1 创建库项目

按以下步骤创建库项目。

(1) 打开一选定的网页, 但先不要选择文档窗口中的元素。如果有元素被选中时, 打开库面板时, 该项就会变成库项目。

(2) 选择菜单 Window|Library 选项打开库面板。

(3) 选择文档中需要存为库项目的网页元素(图像、文本等)。

(4) 用鼠标右击素材管理器窗口, 在弹出的菜单中选择 New Library Item , 或者单击素材

面板中的 New Library Item 图标，然后在素材管理器窗口中输入新的库项目名。

13.3.2 增加库项目到网页

为了将库元素增加到网页中，可按照以下方法操作。

- (1) 将光标放置于插入库元素的位置。
- (2) 选择菜单 Window|Library 选项或者在素材面板中单击 Library 图标打开库面板。
- (3) 将库面板中某个库元素拖到网页中，或者先选择库元素，然后单击库面板左下方的“Insert”按钮。

此时网页中出现了新的网页元素，代码中不仅出现了元素代码，而且在代码的前后还加上了以下语句。

```
<!-- #BeginLibraryItem"/Library/Untitled1.lbi" -->
```

元素代码

...

```
<!--EndLibraryItem -->
```

其中/Library/Untitled1.lbi 为库项目的文件名。

代码说明网页中不仅增加了库元素，而且还与库元素文件保持着联系。

13.3.3 修改库项目

可按照以下方式修改库项目。

- (1) 选择菜单 Window|Library 选项或者在素材面板中单击 Library 图标打开库面板，如图 13.3 所示。

- (2) 在库面板中双击需编辑的库项目，或者先选择库项目再单击面板下面的“Edit”按钮。Dreamweaver UltraDev 将打开一个新的编辑窗口，库元素将显示在窗口中，窗口的背景呈灰色。

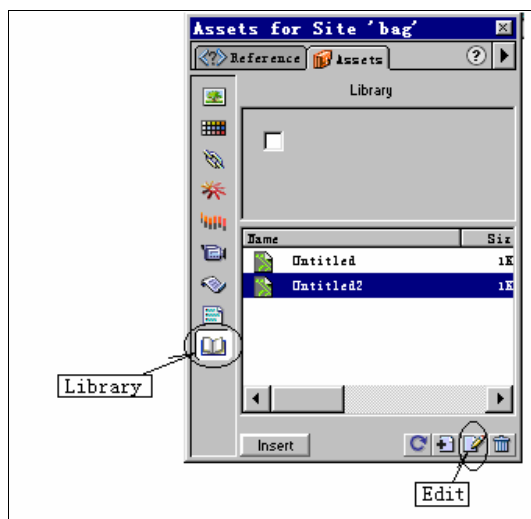


图 13.3 库面板

- (3) 删除原有的库元素，换上新库元素。
- (4) 当关闭编辑窗口时，将弹出如图 13.4 所示的提示框，此时选择“ 是 ”就可以了。



图 13.4 提示是否保存新库元素

(5) 系统将弹出一对话框将所有包括该库元素的文档列出。如果在对话框中单击“ Update ”按钮，则所有文档的库元素都全部更新。如果单击“ Don't Update ”按钮时，网页中的库元素将不更新。以后需要更新时还可以利用菜单 Modify|Library|Update Current Page 选项更新当前网页。

第 14 章 ASP 技术基础

从这一章开始，将讨论网站技术的另一个重要组成部分：Web 数据库访问。讨论的问题包括，ASP、SQL、ADO 以及服务器脚本等技术。

14.1 什么是 ASP

为了理解 ASP 的作用和特点，先回忆一下 ASP 产生的历史过程。

在客户端——服务器——数据库的三层结构中，服务器是关键的中环节，它要能及时接受和响应客户的要求，并能随时根据需要访问数据库以更新网页的内容。为了作到这一点，初期的 Web 服务器普遍采用“ CGI ”技术。CGI 是英文 Common Gateway Interface 的缩写，代表服务器端的一种通用（标准）接口。每当服务器接到客户更新数据的要求以后，利用这个接口去启动外部应用程序来完成各类计算、逻辑处理或访问数据库的工作，处理完后将结果返回 Web 服务器，再返回浏览器。外部应用程序是用 C、C++、Perl、Pascal、Java 或其他语言编写的程序，程序运行在独立的地址空间中。

后来出现了“ ISAPI ”或“ NSAPI ”等技术，其功能与 CGI 相同，但技术方面有些改进。外部应用程序改用动态连接库（DLL），被载入 Web 服务器的地址空间运行，并且用“线程”代替“进程”，因而显著地提高了运行效率。但不论 CGI 还是 ISAPI / NSAPI，都需要编写外部应用程序，而编写外部应用程序并不是一件容易的事情。

1996 年 11 月 Microsoft 公司在此基础上推出的 ASP 技术。ASP 是英文 Active Server Page 的缩写，是用来开发 Internet/Intranet 应用程序的技术。按照微软公司自己的定义：“ASP 是一种服务器端的脚本技术，用来创建动态的、交互的 Web 应用程序”。

ASP 的主要特征是：

- 服务器脚本与 HTML 合并到一个文本中以生成动态网页；
- ASP 提供了一些内建对象，使得脚本变得更加强大；
- 根据需要可以与其他 ActiveX 组件集成；
- 利用一些特定的对象（如 Active Data Object（ADO）组件）访问数据库。这是 ASP 的一个非常强大的功能。

ASP 技术中需要用到以“.ASP”作为后缀的网页（简称为 ASP 网页），它就是服务器端脚本与 HTML 相结合的文档。当浏览器中的客户访问 ASP 网页时，服务器先解读并运行其中的服务器端脚本，并将结果转换成 HTML 形式，然后再与其他内容一道送往浏览器。

对于浏览器来说，访问“.HTM”网页与访问“.ASP”网页的过程几乎没有区别；而对于服务器来说，访问两种网页的本质差别在于，后缀为 HTM（或 HTML）的网页是服务器不经过任何处理就送往浏览器的网页，而后缀为 ASP 的网页则需先运行其中的服务器脚

本或 ASP 命令，并将结果转换成 HTML 语句后再送往浏览器。当然 ASP 网页也不是每次都要在服务器端重新解读和运行，对于那些曾经请求过而又没有变化 ASP 网页，服务器会直接从缓冲区中取出结果而不再次重复运行。

即使不包含服务器端脚本或 ASP 命令的 HTML 网页，也允许使用“ ASP ”作为文件的后缀。因为当服务器解读这种网页时，当发现不包括服务器端脚本或 ASP 命令时，就会直接将文本送往浏览器，其他什么事情也不做，其结果只是稍微降低了程序的运行效率。因此尽管允许将纯 HTML 的网页也使用“ .ASP ”后缀，但并不提倡这样做。反过来，如果网页中包括有服务器端脚本或 ASP 命令而仍然采用后缀“.HTM ”或“.HTML ”时，将出现错误。

14.2 使用 ASP 的优点

ASP 是一种基于面向对象的结构。使用 ASP 技术的优点概括起来有以下几点。

(1) ASP 脚本采用 VBScript 或 JavaScript 等比较简单的语言编写脚本，并能结合系统提供的内建对象快速开发网站的应用程序。

(2) 脚本和命令与 HTML 放在同一个文档中，和 HTML 语句结合得非常紧凑，不需要专门设置对外接口。

(3) 无须编译就可在服务器端直接运行，并与 Web 服务器运行在同一个进程中，运行的效率高。ASP 还可以与扩展标记语言(XML) 组件对象模型(COM) 相结合，组成功能强大的交互式网站。

(4) 与浏览器无关。由于脚本是在服务器端运行，将结果转换成 HTML 的格式后再送往浏览器的。而 HTML 是各类浏览器普遍使用的语言，因此都能正常显示结果。

(5) ASP 的源程序不传到浏览器，因而数据的保密程度高，所写的源程序也可防止被他人剽窃，从而提高了程序的安全性并保护开发者的版权。

14.3 ASP 的内建对象

ASP 提供了大量的内建对象和可安装的 ActiveX 组件，利用它们可以使脚本的功能变得更加强大。这些对象主要有以下几种。

- Response 对象：用来向浏览器发送信息。
- Request 对象：用来获取浏览器端提交的信息。
- Application 对象：用来存储应用中所有用户的共享信息。
- Session 对象：用来存储一个访问者在多张网页中的共享信息。
- Cookies 对象：用来在浏览器端暂存单客户的共享变量。
- Server 对象：用来创建其他对象的实例。
- ObjectContext 对象：在执行 Microsoft Transaction Server (MTS) 过程中进行管理。
- ActiveX Data Object 对象：ActiveX Data Object(ADO) 是用于对数据库进行访问和管理的对象，在后面几章中将详细论述。

下面先重点介绍几个常用的内建对象。

14.3.1 Response 与 Request 对象

Response 与 Request 是服务器与浏览器之间交换信息的两个最常用和最重要的对象。

1. Response 对象

Response 对象的作用是向客户端传送信息。可以用以下几种形式来使用 Response。

Response.Write (数据或变量) ;

或

<%=数据或变量 %>

例如：

Response.Write ("您好");

或

<%= "您好！" %>

两种格式的效果完全相同，都将在浏览器上显示“ 您好！”的字符串。符号“ <%=...%> ”就相当于“ Response.Write ”。

Response 还可用于转向其他网页。使用的语句是 Response.Redirect()。

例如：

```
if (条件) {
    Response.Redirect("新网页的 URL");
}
```

语句的含义是，如果条件成立，立即转向新的网页。

2. Request 对象

Request 是 ASP 的内置对象，它的主要作用是获取从客户端提交的数据。可以选择 5 种方法之一来完成数据传递的任务。

- Request.QueryString：用来获取浏览器表单(Form) 用 Get 方法提交的数据。
- Request.Form：用来获取浏览器表单(Form) 用 Post 方法提交的数据。
- Request.ServerVariables：用来获取服务器工作的环境变量。
- Request.Cookies：用来访问或存储浏览器端的 Cookie 变量。
- Request.ClientCertificates：用来获取有关客户身份的数据。

当客户的表单(Form) 用 Get 方法提交数据时，提交的数据将附在 URL 字符串的"?"号后面，例如：

http://www.Csshop.com/result.asp?Select=40086

此时服务器应使用下列语句来获取提交的信息：

接收变量=Request.QueryString ('表单元素名') // 上例中的表单元素名为 Select

当客户在表单(Form) 中用 Post 方法提交数据时，提交的数据将单独保存，不放在 URL 后面。此时服务器应使用下列语句获取提交的信息：

接收变量=Request.Form ('表单元素名')

ASP 还可以使用一种简略的形式来接受表单中提交的数据：

接收变量=Request ('表单元素名')

此时系统将自动逐个试用，寻找相适用的情况。

关于 Request.ServerVariables 的使用方法，现举例说明如下。例如我们想知道正在访问服务器的用户浏览器是什么类型时，就可以在文档的<body>与</body>之间添加以下代码，

```
<%= Request.ServerVariables("HTTP_USER_AGENT") %>
```

如果客户是在 Windows NT 下使用 Internet Explorer 5.0 访问该页面时，就可得到如下的显示结果：

```
Mozilla/4.0 (compatible; MSIE 5.0; Windows NT; DigExt)
```

14.3.2 几个共享变量

由于 HTTP 是一种不保持连接的协议，网页内的变量不能在网页之间共享。要想共享必须使用系统提供的 Session、Cookies、Application 等对象。

1. Session 对象

Session 对象的作用是保存当前对话客户的信息，为单个用户在多张网页中共享数据提供服务。

(1) Session 的属性有以下两种。

- SessionID：用于区别每个 Session 的标签，它是一个长整型数，由服务器自动生成。

- Timeout：用来设置 Session 的有效期限，以分钟为单位。缺省情况下的有效时间为 20 分钟。即设置的 Session 变量 20 分钟以后将自动失效。可以在网页中使用以下语句延长 Session 的有效时间，例如：

```
<% Session.Timeout=60 %>
```

将 Session 的有效时间延长至 60 分钟。

(2) Session 的方法是 Abandon。用于结束当前的会话。

(3) 使用示例。

```
Session("变量名称") = 变量
```

```
Set Session("对象名称") = 对象
```

例如：

```
<% Session("hello")="欢迎访问本网页！"
```

```
Response.Write (Session("hello")) %>
```

将在浏览器上显示“ 欢迎访问本网页！”字符串。

在 Web 应用程序中，每一个新访问（会话）者都将产生一个新的 Session 对象。用这个 Session 保存的变量可以供这个用户在该 Web 应用的所有页面中共享。如果另一位客户也同时访问该 Web 应用，它也将拥有自己的 Session 变量，但两个用户之间不能通过 Session 变量共享信息。

当一个新用户进入应用程序，即一个新的会话开始时，Session_onStart 方法被调用，当该用户离开 Web 应用程序时，Session_onEnd 方法被调用。

2. Application 对象

Application 对象的作用是用于存放应用程序中多个用户的共享信息。

(1) Application 对象的属性有 Lock 和 Unlock。

- Lock：锁定对象，不允许其他进程访问。

■ Unlock：解锁，允许其他进程访问。

(2) 方法：无。

(3) 应用示例：

例如，统计访问人数的部分程序段：

```
Application.Lock           // 锁定 Application 对象，避免多用户竞争访问。
Application("counter")=Application("counter") + 1
Application.Unlock         // 解除对 Application 对象的锁定
```

Application 对象将 Web 应用程序的所有用户(会话) 作为单实例的对象，用于保存 Web 应用程序中所有 ASP 页面的公用信息。例如，可以保存当前会话者的数目，或者所有会话共享的数据库连接信息等。打开应用程序时，Application_onStart 方法立即被调用。当最后一个客户离开应用程序(从 Web 服务器上卸载) 时，Application_onEnd 方法被调用。

3．关于 Cookies 变量

Cookies 变量的作用与 Session 相同，可作为单个用户的共享变量。只是它将这些共享信息保存在浏览器的缓冲区中。由于网页中有关 Cookies 的代码必须在 HTML 标记之前送往浏览器，因此它的内容一般不会太大。简化后的 Cookies 的语法如下：

向浏览器写 Cookies 变量的语句是：

```
Response.Cookies("Cookies 变量名") = 值
```

如果该 Cookies 变量在浏览器缓冲区中已经存在时，则用新变量覆盖原有变量。

读取 Cookies 变量的语句是：

```
Request.Cookies("Cookies 变量名")
```

Cookies 有以下几个属性。

(1) ExpiresAbsolute 属性。本属性可以赋一个日期，过了这个日期 Cookie 就不能再被使用了。通过给 Expires 属性赋一个过期的日期，就可以删除 Cookie。例如，下面的语句就能够删除“ Customer ”的 Cookie 变量。

```
<%
Response.Cookies("Custtomer").ExpiresAbsolute="1/1/1995"
%>
```

(2) Domain 属性。本属性定义 Cookie 要传送的唯一域。例如 Cookie 只传送给 Microsoft 网站的人，可以采用以下语句。

```
<%
Response.Cookies("Custtomer").Domain="www.microsoft.com"
%>
```

(3) Path 属性。本属性定义了 Cookie 只发给指定路径的请求。如果 Path 没有设置，则使用应用程序的缺省路径。

另外 Cookie 还有 Secure、HasKeys 等属性，这里就不介绍了。

14.3.3 Server 对象

Server 对象的主要作用是创建和控制以 COM 为基础的类的实例以及 Scripting 组件等。

(1) Server 对象只有一个属性 :ScriptTimeout 允许设置一个 ASP 程序的最长的运行时间。

语法结构如下：

```
Server.ScriptTimeout = n; //n 代表秒数
```

默认情况下，系统的 ASP 运行的时限是 90 秒。就是说，如果脚本超过该时间仍没有执行完时，它将被终止。使用以下代码就可以在浏览器看到这个时间：

```
<%= "ASP 运行的时限是" + Server.ScriptTimeout + "秒" %>
```

可以利用以下语句来延长限制的时间，例如使用以下语句，

```
<%
```

```
Server.ScriptTimeOut = 200
```

```
%>
```

将 ASP 程序运行的时间延长到 200 秒。

(2) Server 的方法有：CreateObject、MapPath、URLEncode、HTMLEncode。

- CreateObject：用于创建 ActiveX Server 组件的实例。
- MapPath：用于将文档当前的虚拟路径转换为实际的物理路径。
- URLEncode：将 URL 代码转换为字符串。
- HTMLEncode：将指定的字符串进行 HTML 编码。

14.4 服务器端脚本的编写方法

ASP 的脚本可以使用 JavaScript 也可以使用 VBScript 语言编写，在同一张网页中甚至可以交叉使用这两种语言。ASP 缺省的语种是 VBScript，但也可以在网页中重新定义本网页的缺省语言，此时须在网页的第一行用下列格式定义：

```
<%@ LANGUAGE="JAVASCRIPT"%>
```

以指定本网页的缺省语言为 JAVASCRIPT。

ASP 脚本可以放在网页代码的任何位置，可以放在<HTML>标签的前面、</HTML>的后面或者两者之间。如果不是函数，放置的位置就代表执行的顺序。如果是函数则只在被调用时才执行。

ASP 脚本可以使用两种格式。

(1) 使用<Script...>...</Script>格式。

例如：

```
<Script Language="JavaScript" Runat="Server">
```

```
脚本语句
```

```
</Script>
```

(2) 使用<%...%>格式。

例如：

```
<% 脚本语句 %>
```

第二种格式比较简单，但只能使用缺省语言。第一种格式中 Runat="Server"是关键。如果不写这一句，代表它是浏览器端的脚本，只能运行在客户端。第一种格式的最大优点是可以根据需要选择语种。下面通过几个例子来说明 ASP 脚本的编写方法。

例 1：分段编写 ASP 脚本代码

ASP 脚本允许分段编写但不允许嵌套。下面是一段用 VBScript 语句书写的服务器端脚本，前面不用分段，后面一个使用分段。

```
<% Dim I
  For I=0 To 4
    Response.Write "欢迎使用 ASP 技术 ！<Br>"
  Next
%>
```

下面的脚本使用分段方法编写：

```
<% Dim I                // 第一段
  For I=0 To 4
%>

Response.Write "欢迎使用 ASP 技术 ！<Br>"
<% Next %>             // 第二段
```

分段书写 ASP 代码虽然阅读起来不太容易，但由于在段落之间可以插放其他字符串或 HTML 标记，因而给设计工作带来了更多的灵活性。

例 2：用循环语句执行 Response 对象的过程。

```
<% @ Language="JAVASCRIPT" %>
<!-- 说明缺省语言使用 JAVASCRIPT →
<% for (var i=1; i<6; i++ ) {%>
<FONT SIZE=<%=i %> COLOR=#0000FF>
ASP 代码示例</FONT><BR>
<!-- 将 i 的值赋给 SIZE →
<% } %>
```

脚本采用了分段的编写方法，如果不分段<%= i %>将被包含在<%...%>之中变成了嵌套结构，而嵌套是不允许的。

这个网页被访问时，脚本将首先在服务器端运行，将循环语句变成多条 HTML 语句，然后再送至浏览器。实际上送往浏览器的是循环中产生的 HTML 语句。这些语句是：

```
<FONT SIZE=1 COLOR=#0000FF> ASP 代码示例
</FONT><BR>
<FONT SIZE=2 COLOR=#0000FF> ASP 代码示例
</FONT><BR>
<FONT SIZE=3 COLOR=#0000FF> ASP 代码示例
</FONT><BR>
<FONT SIZE=4 COLOR=#0000FF> ASP 代码示例
</FONT><BR>
<FONT SIZE=5 COLOR=#0000FF> ASP 代码示例
</FONT><BR>
```

在浏览器端执行的结果将是 5 条由小逐步变大的字符串，具体情况如下：

ASP 代码示例

ASP 代码示例

ASP 代码示例

ASP 代码示例

ASP 代码示例

例 3：Request 与 Response 对象相结合处理表单数据。本例的情况如图 14.1 所示。

客户端从表单 (Form) 中输入并提交数据。Form 中包括两个文本输入框、一个提交按钮。文本输入框一个名为“ Name ”，另一个名为“ Job ”。

接收数据的网页 <%=Request("Name")%>和<%=Request("Job") %>语句处理数据。语句中用 Request()获取客户端数据，并用<%=...%>部分(相当于 Response.Write) 将获取的数据送回浏览器显示。这是将 Request()与 Response()结合使用的典型方法，后面将要经常使用。

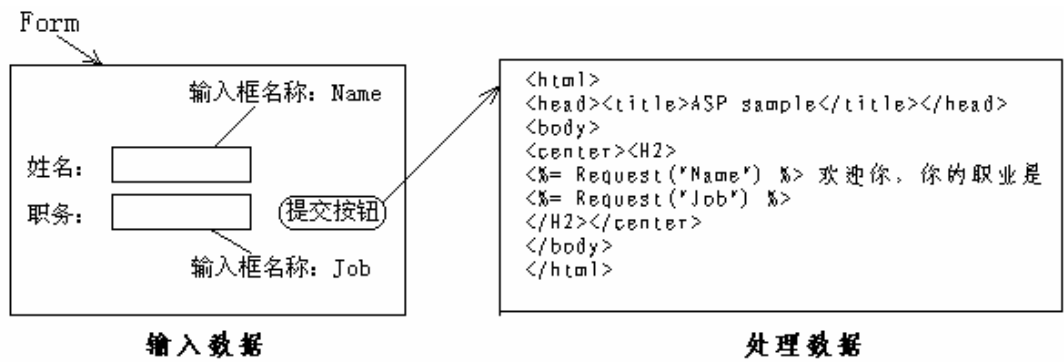


图 14.1 表单数据的提交与处理

图中左边为表单网页，右边为处理表单数据的网页。如果在表单的姓名中填写“ 刘香 ”；职务中填写“ 教师 ”时，右边网页将显示“ 刘香 欢迎你，你的职业是教师 ”字符串。

14.5 浏览器端脚本与服务器端脚本的比较

同一张网页中可能既包括浏览器端脚本又包括服务器端脚本。这两种脚本有哪些不同点呢？让我们用对比的方法加以说明。

14.5.1 两种脚本调用不同的内建对象

浏览器端脚本调用的是浏览器提供的内建对象(函数)，而服务器端脚本调用的是服务器提供的内建对象，而浏览器与服务器两者提供的对象是不同的。下面用一个简单的示例说明。

例如，用 JavaScript 语言编写一段服务器脚本，其功能是上午(6 ~ 12 点)显示“ Good morning! ”，下午(12 ~ 18 点)显示“ Good afternoon! ”。

用服务器端脚本表示如下。

```
<script Language="JAVASCRIPT" Runat="SERVER">
```

```

{
var now = new Date();
var m_hour;
m_hour = now.getHours();
<!-- 获取时间 (小时) -->
if ((m_hour>6) && (m_hour<12))
{Response.Write("Good morning!");}
else if((m_hour>=12) && (m_hour<18))
{Response.Write("Good afternoon!");}
}
</script>

```

服务器在解读这段代码时，发现文本中包含服务器端脚本代码，因此先在服务器端运行，然后再将运行结果和文本的其他语句一道送往浏览器。虽然这段脚本能够正常运行，但脚本反映的时间是服务器所在位置的时间，不符合客户端的情况。

为了能正确反映客户所在时区的情况，应将这段代码放在客户端运行。因此将脚本 `<script...>` 中的 `RunAt="server"` 部分去掉，这段脚本也就变成了浏览器端脚本。但改变后的脚本却不能正常运行，系统将提示“缺少对象"Response"”的错误信息。

为什么会产生这样的错误呢？原来 `Response` 是服务器提供的内建对象，它不能在浏览器端运行。浏览器端脚本只能调用浏览器端提供的内建对象，因此只有用 `document.write` 来替换 `Response.Write` 时，脚本才能在浏览器端运行。修改后的脚本代码如下。

```

<script Language="JAVASCRIPT" >
{
var now = new Date();
var m_hour;
m_hour=now.getHours();
<!-- 获取时间 (小时) -->
if ((m_hour>0) && (m_hour<12))
{document.write("Good morning!");}
else if((m_hour>=12) && (m_hour<23))
{document.write("Good afternoon!");}
<!-- 用 document 取代 Response -->
}
</script>

```

14.5.2 关于事件驱动

浏览器端脚本与服务器端脚本的运行机制不同。浏览器端脚本是对下载后的网页进行控制，而服务器端脚本只在网页接受访问并打开时才执行。浏览器为脚本提供了很多事件（如 `onClick`、`onMouseOver` 等等），用来控制网页元素属性改变的时机。而服务器却没有给用户提供什么事件。如果说有事件的话，那就是访问（或打开）网页。为了充分利用这一事件，常常

在网页中设一些带标签的元素，让服务器根据标签的状态去完成不同的工作。

下面用一个例子来说明。

假定在表单(Form)上放有 3 个文本输入框(textfield、textfield2、textfield3) 及一个提交按钮以完成简单的加法运算，如图 14.2 所示。本功能可以利用浏览器端脚本来实现，其脚本请参考第 11.5.5 节。

为了进行对比，这里再利用服务器端脚本来完成同样的工作，看看两者之间有什么不同。

将 Form 的 Action 属性设置成自己网页，为了区别刚进入网页以及后续计算两种不同的情况，设置一标签框，其值为“ yes ”。并用下列语句将它隐含起来。

```
<input type="hidden" name="bz" value="yes">
```

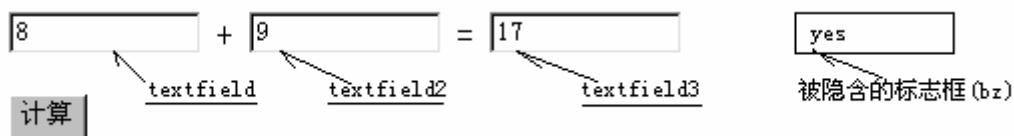


图 14.2 简单的加法运算

脚本代码如下。

```
<%@LANGUAGE="JAVASCRIPT"%>
<!-- 缺省语言为 JavaScript -->
<% if (Request("bz")=="yes"){           // 服务器端脚本
    ss1=parseFloat(Request("textfield")); // 从 textfield 文本框中取值并转换为浮点数
    ss2=parseFloat(Request("textfield2")); // 从 textfield2 文本框中取值并转换为浮点数
    ss3=ss1+ss2;}                       // 完成加法运算
else
    { ss1= 1; ss2=1; ss3=2;}           // 初次进入网页时赋值
%>
<form name="form1" method="post" action="calcu.asp">
    <!-- 表单提交时转向自己网页 -->
    <p> <input type="text" name="textfield" size="12" value=<%=ss1%>>
        <!-- 输入框动态赋值 -->
        <input type="text" name="textfield2" size="12" value=<%=ss2%>>
        <input type="text" name="textfield3" size="12" value=<%=ss3%>>
    </p>
    <p>
    <input type="submit" name="Submit" value="计算">
        <!-- 提交按钮 -->
    <input type="hidden" name="bz" value="yes">           // 访问标签
    </p>
</form>
```

当初次进入网页时，bz 输入框的标签还未定义(是 undefined)，故 ss1、ss2 都是 1，ss3

为 2；当填入数据单击“ 计算 ”按钮时，ss1、ss2 的值来自两个输入框提交的数据，并将其转换成浮点数，ss3 是两者的和并赋值给 textfield3。这段计算是在重新打开网页时在服务器端完成的。

14.5.3 脚本选择的一般原则

从前面的例子可以看出，有些任务既可以放在浏览器端执行，也可以放在服务器端执行，这种情况下，究竟应放在那端执行才合理呢？

对于这个问题要作具体分析。在 Web 系统中一台服务器往往要同时为众多的客户服务，服务器的负担通常是比较重的。因此只要有可能，任务都应该放在客户端完成。

大体上看，凡是只涉及网页元素表现形式上的变化，通常都利用浏览器脚本完成；凡涉及内容方面的变化，如对数据库查询、记录的增添、修改等数据库的操作等只能在服务器端完成。

有些任务，比如对用户登录口令的检查，放在客户端或者服务器端都能完成，而且放在客户端执行时，还能够减少网络上的传输，提高执行效率。但是这样的任务放在客户端执行是不合理的。因为这里存在着一个数据安全问题。由于从浏览器可以查看到网页的源代码，因此将正确的口令传送到客户端去检查时，很容易泄密，造成严重的后果。类似的问题还有客户帐号、信用卡号、支票密码等商业秘密，从数据安全的角度出发，这些任务都只能放在服务器端处理而不能放在客户端执行。

另外，还有一个任务难度方面的考虑。通常，客户端计算机的软、硬件配置都比较简单，而服务器端的配置通常比较高。因此，一些较为复杂的处理工作放在服务器端处理比较恰当。

14.6 关于 Global.asa 文件

Global.asa 文件是大多数应用程序中一个重要的文件，打开应用程序时系统首先执行的就是这个文件。虽然这个文件对于每一个应用程序来说，并不是必需的，但如果需对 Application、Session 等变量进行初始化，或者有些预处理或善后处理工作要做时，使用 Global.asa 文件将非常方便。应用程序中如果设置该文件时，必须将它放在站点的根目录下。

文件中通常包括：Application_OnStart、Application_OnEnd 以及 Session_OnStart、Session_OnEnd 等事件。

其中，Application_OnStart 是第一个客户进入应用程序时引发的事件，Application_OnEnd 是最后一个客户离开应用程序时引发的事件；Session_OnStart 是每一个新客户进入应用程序，开始会话时引发的事件，Session_OnEnd 是每一个会话结束时引发的事件。

例如想为一 Session 变量（如 M_dd）设初值。可在 Global.asa 文件中的 Session_OnStart 段中写出以下语句。

```
<script Language="JavaScript" Runat="Server">
function Session_OnStart()
{
    Session("M_dd") = "";
}
```

</script>

14.7 UltraDev 提供的服务器行为(Server Behavior)

一个 Web 应用程序常包括各种类型的动态网页。这些网页主要有：数据库查询网页、主/从表同步网页以及记录编辑（修改、插入、删除记录等）网页等等。网页类型不同时相应的 HTML 代码，动态内容以及服务器行为都不同。UltraDev 提供了一组预先定义好的服务器行为来简化动态网页的管理和设计工作。

服务器行为(Server Behavior)，是用 VBScript、JavaScript、Java 或者 ColdFusion 编写的服务器端运行的脚本。这些脚本涵盖了动态网页设计中的常用部分。

选择菜单 Window|Server Behaviors，可进入服务器行为窗口，单击“ + ”按钮，将看到 UltraDev 预先定义好的 15 组行为，这些行为的列表如图 14.3 所示。

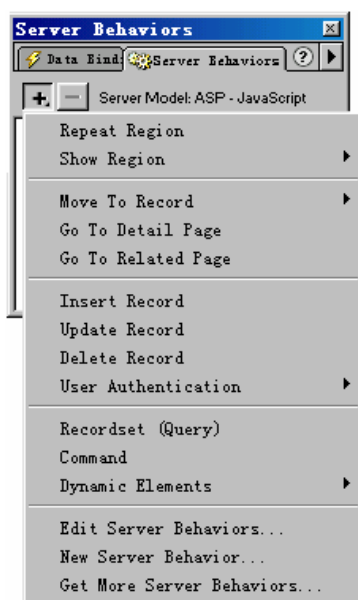


图 14.3 服务器行为

其中每项行为的含义如下。

Repeat Region：显示多行记录。

Hide Region：隐含区。

Move To Record：创建浏览按钮。

Go To Detail Page：转向从网页。

Go To Related Page：转向相关网页。

Insert Record：插入新记录。

Update Record：修改记录。

Delete Record：删除记录。

User Authentication：用户访问权限设定。

Recordset(Query)：创建记录集。

Command：创建命令集。

Dynamic Elements：设置动态元素。

Edit Server Behaviors：编辑已有的服务器行为。

New Server Behavior：定义新服务器行为。

Get More Server Behaviors：获取更多服务器行为。

当选择其中一行为项时，这项行为的脚本代码将自动插入到网页中。若需删除某个已经设定的行为项时，先在此窗口中选中该行为，然后单击窗口中的“ - ”按钮。

这些行为项将在后续各章中结合具体应用加以介绍。

第 15 章 建立与数据库的连接

15.1 概述

建立 Web 应用程序与数据库的连接，以便随时从数据库中提取数据或者向数据库存入数据，对于需要使用大量数据的网站来说，是一项经常而又十分重要的工作。

然而，实现这种连接并不容易，因为市场上有着各种类型的数据库，每种数据库采用的文件格式都不相同。如果要求程序设计者为不同类型的数据库编写不同的应用程序，或者为各类数据库编写不同的接口，都将是一件既烦琐又极为复杂的任务。

如何解决这个矛盾呢？解决的方法就是使用由系统提供通用的接口。

下面介绍微软（Microsoft）公司提供的几个通用接口。

15.2 与数据库连接的通用接口

15.2.1 ODBC

ODBC 是英文 Open Database Connectivity 的缩写，中文意思是“开放数据库互联”。这是由微软公司用 C 语言开发的连接数据库的解决方案，实际上是由很多函数组成的应用程序接口（Application Program Interface, API），它将所有对数据库的底层操作隐藏在 ODBC 的驱动程序之中。应用程序只需要用统一的接口指向 ODBC，然后 ODBC 再用不同的驱动程序去驱动不同类型的数据库。

有了这个统一的接口，Web 设计者设计应用程序时不用再需要考虑数据库是什么类型，来自什么公司以及使用什么样的格式等具体情况。ODBC 接口的情况如图 15.1 所示。

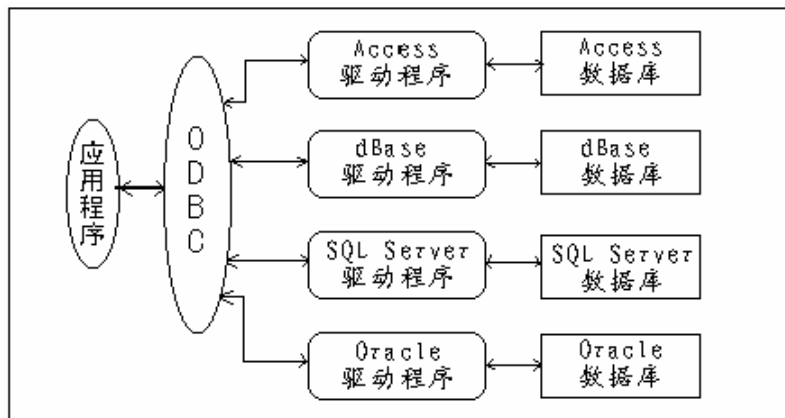


图 15.1 ODBC 接口示意

尽管通过 ODBC 已经能够驱动大多数常用的数据库，但因为这种编程接口还过于复杂，而且还没有进行优化。所以后来微软公司又在 ODBC 的基础上专门针对 Access 的.mdb 数据库创建了优化编程接口 DAO (Data Access Object)；针对 SQL Server 数据库，创建了优化编程接口 RDO(Remote Data Access Objects)，这种接口还能应用于 Oracle 数据库。

15.2.2 ADO

1. ADO 的特点

ADO 是微软公司进军 Internet 后的最新产品，它是 Active Data Object 的英文缩写，是建立在 OLE DB 技术基础上的接口技术。OLE DB 在 ODBC 的基础上，用面向对象的思想对 ODBC 的函数重新进行了分类和包装，形成了一种新的标准。可以说 ODBC 是 OLE DB 的子类，而 OLE DB 是 ODBC 的超类。利用 OLE DB 不仅能访问关系型数据库，还能访问非关系型数据（如文件等）。

ADO 又对 OLE DB 的接口进行了优化。ADO 是 ODBC 和 OLE DB 的上层接口技术。它比 RDO、DAO 等接口具有更高的性能、更小的容量及更简便的操作，特别适合于 Web 应用程序的需要。当前，ADO 已经成为取代前面几种技术的最新技术。

使用 ADO 接口几乎能够访问全部常用的数据库，如 Access、Paradox、DBase、Foxpro、SQL Server、Oracle 等，还能访问非关系型文件。

ADO、OLE DB、ODBC 三者之间的关系如图 15.2 所示。

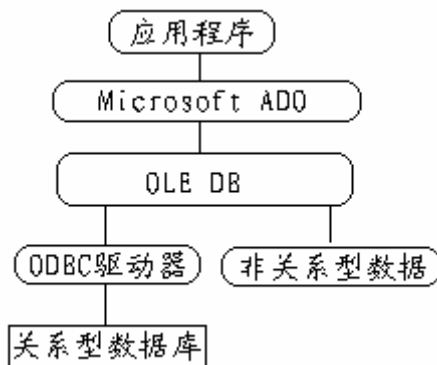


图 15.2 ADO 与 OLE DB, ODBC 关系示意

ADO 的特点归纳起来有以下几点：

(1) 易于使用。相对于 OLE DB 和 ODBC 来说，ADO 属于高层应用，它具有面向对象的特征。在 ADO 的对象结构中，对象之间的层次关系并不明显。相对于其他数据库访问技术（如 DAO）来说，又不必关心对象的构造顺序和构造层次。

(2) 高速访问数据库。由于 ADO 基于 OLE DB，所以，它也继承了 OLE DB 访问数据库的高速性。

(3) 可以访问不同的数据源。ADO 技术不仅可以访问关系数据库，还能访问非关系型数据库的文件系统，这一点也是从 OLE DB 中继承下来的。这个特点使得应用程序具有很好的灵活性和通用性。

(4) 可以用于 Microsoft Active 页。ADO 技术可以以 ActiveX 控件的形式出现, 可以用于 Microsoft Active 页。这个特征可以简化 Web 页的编程技术。

(5) 程序占用的内存少。

2. ADO 的内置对象

ADO 有几个内置对象, 通过这些内置对象, 不仅能够访问数据库, 而且还能操作数据库中的数据, 这些内置对象有以下几方面。

(1) Connection 连接对象。一个 ADO Connection 对象代表了针对一个数据库引擎的物理上的连接。这个连接既可以通过远程网络(如 internet) 连接到远程数据库, 也可以连接到本地文件系统中的某个数据库文件。因此, 数据库在何处是没有什么关系的, 不管数据库在哪里, 都可以使用基本相同的 Connection 对象, 并使用 Connection 对象的方法来操纵数据。

(2) Recordset 记录集对象。一个 Recordset 对象是一个数据库查询的结果。可以从一个打开的 Connection 对象中创建 Recordset 对象, 也可以在 Connection 之外建立 Recordset 对象。当使用 Recordset 的 open 方法时, 一个新的连接对象将会自动被创建。

一个 Recordset 是通过查询语句产生的临时数据表, 是原数据表的子表, 这个子表还具有一个游标(cursor), 它指示着 Recordset 的当前记录。操纵这个游标就可以在子表的记录之间移动。

(3) Command 命令对象。一个 Command 对象代表一组(多条) SQL 语句。将一些需多次重复使用的查询或者交互过程保存在 Command 对象中, 使用时只需调用命令名即可, 既方便、可靠又提高了执行效率。

(4) Error 错误对象。数据库连接或查询中所引发的问题可用 Error 对象来表示。在 Connection 对象中有一个 Error 集, 其中包括了上一次与数据库连接时所产生的所有错误。例如, 试图打开一个不存在的数据库, 或者打开一个通过网络不能到达的数据库, 此时在 Connection 对象的 Errors 集中, 可能会得到几个不同的 Error 对象。其中一个 Error 对象代表从底层的 OLE DB 引擎返回的错误; 另一个 Error 对象可能代表了由于不能建立物理上的连接, 从 OLE DB 引擎返回的错误等。这些问题都可用 Error 对象来表示。

实际上 ADO 中前 3 个对象: 连接对象、记录集对象和命令对象都能完成对数据库查询或其他方面的操作, 而且都是通过 SQL 语句来实现这些操作的。一个相同的任务可以使用 3 个对象中的任一个来完成。但为了使设计思想更加清晰, 使程序的易读性更强, 通常习惯于作以下分工。

- 连接对象主要用于建立与数据库的连接。
- 记录集对象主要用于完成数据库的查询操作。
- 命令对象主要用于执行插入、删除记录的操作, 或者执行事先在数据库中写好的 SQL 命令(执行存储过程), 或者进行参数查询。

15.3 与数据库的连接方法

ASP 的连接对象是一种可以在网站中各网页共享的对象。连接对象中应该包括与数据库连接的各种信息。

利用 Connection 对象的 Open 方法来初始化一个连接。只有使用了 Connection 对象的 Open 方法以后，Connection 对象才会真正存在，然后才能够对数据表进行操作。

使用 Open 方法有两种途径。

(1) 直接将连接字符串传给 Open 方法。

例如：

```
<%
set conn = Server.CreateObject( "ADODB.Connection" );
conn.Open "DSN=s_database", "sysmage", "111111";
%>
```

其中第一句是创建一个连接对象的实例 conn，第二句是利用 Open 打开一个与数据源名 (DSN) 的连接。其中数据源名设为 s_database，数据源名代表着连接的各种参数。用户名为 sysmage，口令为 111111。其中用户名和口令为可选项。

(2) 利用ConnectionString 属性。

例如：

```
<%
set conn = Server.CreateObject( "ADODB.Connection" );
conn.ConnectionString= "DSN=s_database", "sysmage", "111111";
conn.Open
%>
```

在这里先建立 ConnectionString，然后再使用 Open 方法。

下面我们介绍利用系统提供的可视化工具建立数据源名的方法。

15.4 创建数据源名(DSN)

15.4.1 什么是数据源名

数据源名代表数据库的接口，包括数据库的类型、位置、驱动程序和缓冲区大小等参数。数据源名相当于接口的逻辑名。用逻辑名代替物理名，不仅简化了程序，而且增加了灵活性。当数据库的类型或位置有了改变时，只需改变数据源名的设置而不用改变源程序。

有两种不同类型的 ODBC 数据源：机器数据源和文件数据源。

1. 机器数据源

机器数据源把信息保存在系统的注册表中，因而只能被该计算机使用。当程序转移到新的计算机时，要在新计算机上重新设置。机器数据源又分为系统数据源和用户数据源两种。系统数据源可被 Web 服务器上所有用户访问，而用户数据源只能被 Web 服务器中特定的用户访问。通常情况下选用系统数据源。

2. 文件数据源

文件数据源把信息存放在后缀为.dsn 的文件中，如果把文件放在网络共享的驱动器中，那么它可以为局域网中的所有计算机使用。

15.4.2 创建数据源名的步骤

1. 准备工作

创建数据源名之前,必须确认机器上是否已安装有该数据库的驱动程序。为了查看驱动程序,在 Windows 95/98/NT 中,单击开始界面进入控制面板(Control Panel),双击 ODBC DataSource 图标后,再进入 Drivers 页时,就可看见系统已经安装的数据库驱动程序列表。如果你使用的是 Windows 2000 ,则选择 Start|Programs|Administrative Tools|Data Sources 来查看驱动程序列表。列表的情况如图 15.3 所示。

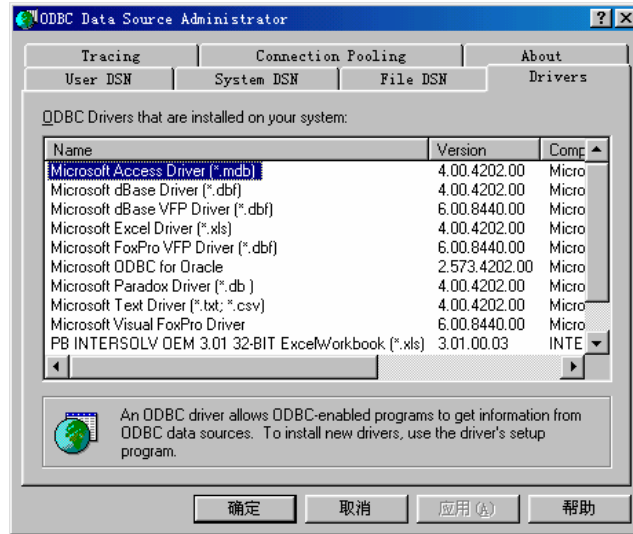


图 15.3 数据库驱动程序列表

还可以为 Excel spreadsheets(电子表格) 和 Text files(文本文件) 创建 DSN , 供 Web 应用程序使用。

下面将结合 Access 数据库“ dbdemos.mdb ”来说明设置过程。

2. 创建数据源名的步骤

创建数据源名的步骤如下。

- (1) 从 Windows 的桌面开始,通过“ 设置 ”|“ 控制面板 ”进入控制面板。
- (2) 用鼠标双击“ ODBC Data Source ”图标。
- (3) 选择 System DSN 页后单击“ Add ”按钮。
- (4) 在弹出的窗口中选择 Microsoft Access Driver [*.mdb]项后单击“ 完成 ”按钮。
- (5) 在数据源名(Data Source Name)文本框中输入数据源的名字(DSN),单击“ 选取 ”(Select)按钮选择连接的数据库(本例中选择 dbdemos.mdb 库)。图 15.4 是选择数据库的界面。

(6)必要时,还可单击“ 高级”(Advanced)按钮进一步设置数据库的登录名(Login Name)和口令(Password);也可单击“ 选项”(Options)按钮设置以下属性。

页超时(Page Timeout): 从开始连接到连接成功,可持续的时间。

缓冲区大小(Buffersize): 缓冲区大小。

独占(Exclusive): 是否排它性连接(通常不选此项)。

只读 (ReadOnly): 是否只读性连接 (通常不选此项)。



图 15.4 创建数据源名

(7) 单击“OK”按钮后再单击“确定”按钮完成对数据源名的创建工作。

15.5 建立连接的步骤

在 UltraDev 中可以利用系统提供的可视化界面创建数据库连接，具体步骤如下。

(1) 进入 Dreamweaver UltraDev 的界面后，选择菜单 Modify|Connection 项将弹出连接窗口，在窗口中单击“New”按钮。在弹出的菜单中选择“Data Source Name (DSN)”项，将弹出如图 15.5 所示的对话框。

(2) 在对话框中填入以下数据。

- Connection Name: 连接对象名。图中填入的是 conn。
- Data Source Name(DSN): 数据源名。在下拉列表中选择。

必要时还可以规定使用本连接的用户名 (User Name) 和口令 (Password)。

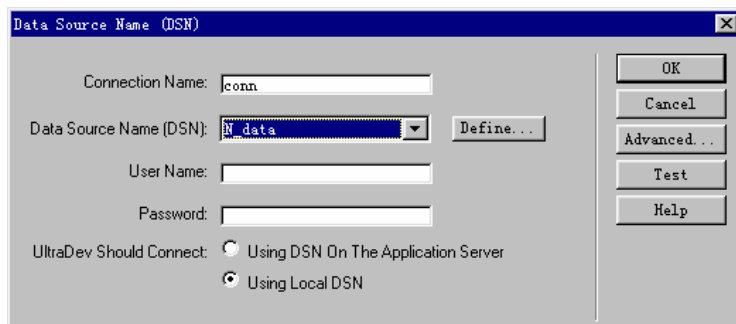


图 15.5 利用 DSN 进行连接

(3) 单击 “ Test ” 按钮检查连接情况。如果连接成功，单击 “ OK ” 按钮后再单击 “ Done ” 按钮。

当完成了上述工作后查看网页的源代码时，并不能看见代码中的连接信息，这是为什么呢？

原来这里创建的连接只是为以后开发 Web 应用程序时的“ 备用 ”，其代码保存在 Dreamweaver UltraDev 安装目录的 Dreamweaver UltraDev | Configuration | Connections 的文件夹的 connections.xml 文件中。当网页创建记录集时才会使用到这些代码。关于创建记录集的问题将在第 16 章中叙述。

第 16 章 创建记录集

16.1 什么是记录集

记录集(Recordset) 是用 SQL 语句从数据表中检索出来的记录的集合, 是数据表记录的子集, 每张网页都可以为本网页设置一个或多个记录集, 每个记录集相当于一张放在服务器内存中的临时数据表。在 ASP 中对数据库的访问都是对记录集的访问, 当然对记录集中数据所作的改变也会使硬盘数据库中的数据作相应的改变。

因此, 记录集是处理数据的最重要的对象。尽管 Connection 对象和 Command 对象也能用于处理数据, 但它们最终的目标常常也是用来创建一个记录集(Recordset) 的。

为什么不直接对数据表进行操作而要通过记录集这个中介呢?

这是因为数据表通常包括比较完整的数据, 只有这样, 数据表才具有通用性。而对于某个具体应用来说, 要用到数据表的全部数据的机会是很少的。

通过 SQL 语句组成记录集, 就是为了从一张或多张表格中提取本次应用中需要用到的数据。尽管通过 SQL 语句将数据表的全部记录都拿到记录集中来也很容易, 但并不提倡这样做。正确的做法应该是将尽量少的记录放到记录集中来, 以减少内存消耗和通讯流量, 这些都是网络系统需要考虑的重要因素。

更为重要的是, 记录集是一种动态数据, 它是用 SQL 语句对数据表进行操作的结果; 而数据表记录却是一种相对静止的数据, 不对记录进行增加、删除、修改, 它是不会变化的。

16.2 SQL 语言基础

SQL 是英文 Structure Query Language 的缩写。它是一种非过程的、结构化数据库查询语言; 它是一种标准, 绝大多数关系数据库都使用 SQL 语言。

实际上, SQL 语言的功能不限于做查询工作, 它还能对数据库进行其他操作。SQL 本身可分为数据定义语言 DDL(Data Definition Language) 和数据处理语言 DML(Data Manipulation Language) 两部分。

DDL 用于创建、更改和删除数据库以及它们的索引; DML 用于选择、插入、更新和删除数据库中的数据。

1. 操作符及标准函数

操作符

- 算术操作符有: +、-、*、/
- 比较运算符有: <、>、=、<>、>=、=<、IS NULL、IS NOTNULL
- 逻辑运算符有: NOT、AND、OR

■ 字符串连接符： ||

集聚函数

查询中可以使用的集聚函数有以下几种。

- SUM()：对数字型字段求总和。
- AVG()：对数字型字段求平均值。
- MIN()：确定字段中的最小值。
- MAX()：确定字段中的最大值。
- COUNT()：统计符合条件的记录数。

字符串处理函数

字符串处理函数有以下几种。

- UPPER()：将字符串改变成大写。
- LOWER()：将字符串改变成小写。
- TRIM()：在字符串左端，右端或两端删除规定字符的重复部分。
- SUBSTRING()：从字符串中分离出一个子串。

2. 操作语句

DML 的操作语句包括以下几种。

- SELECT 语句：用于查询已经存在的数据。
- INSERT 语句：用于在数据库中插入新记录。
- UPDATE 语句：用于修改已经存在的记录。
- DELETE 语句：用于删除数据库中的记录。

语句中的英文字母不分大小写，如果一条 SQL 语句很长，可分成几行书写，在需要分行处按回车(Enter)键即可，但绝不能将 SQL 中的一个字(Word)分写在两行上。

SQL 语句中包括一些常用的标点符号。其中逗号用来分隔语句中的并列项(如表中的字段名用逗号隔开)，句号用来标识某一项的附加信息，如两个表都含 Number 项时，可用 Table1.Number 和 Table2.Number 来区分。

3. 关于 Select 查询语句

Select 是使用得最为频繁的选择命令。一条 Select 命令可以用于查询一张表，也可以同时查询多张表。

命令的格式：

SELECT 字段名列表

FROM 表名

[WHERE 查询条件]

[ORDER BY 排序的字段]

[GROUP BY 分组字段]

命令中的英语部分为保留字。其含义通过下面的示例说明。

例 1：显示全部字段

```
Select * From TSGL
```

命令的作用是显示 TSGL 表的全部字段。命令中的“*”是通配符，代表全部字段。

例 2：显示部分字段

```
Select NUM, SM From TSGL
```

只显示 TSGL 表中的几个字段。其中 NUM、SM 为字段名。

例 3：有条件地显示全部字段

```
Select * From TSGL
```

```
Where NUM >= '0055'
```

显示 TSGL 表中符合 NUM >= '0055' 条件的全部字段，其中 NUM 字段属于字符串型。

例 4：使用条件(Where) 语句将两表拼接(join) 显示

```
Select Tsk1.bh1,Tsk1.sm,Tsk2.bh2,Tsk2.zsh From Tsk1,Tsk2
```

```
Where (Tsk1 =Tsk2)
```

Tsk1 和 Tsk2 是两张表，拼接的条件是：Tsk1.bh1=Tsk2.bh2；显示的内容是每张表中的两个字段。

例 5：使用分组语句(GROUP BY) 进行分组统计

```
select part_no, sum(quantity)
```

```
from parts
```

```
group by part_no
```

group by 是对记录进行分组的语句，分组字段名必须同时出现在 select 子句中，分组语句必须与积聚函数同时使用。

例 5 的含义是：在 parts 表中，按照 part_no 字段进行分组，显示各组的 part_no 字段以及各组 quantity 字段的总和。这里的 quantity 必须是数字型字段。

例 6：使用排序(ORDER BY) 子句

```
select customer_no
```

```
from "c:\data\customer"
```

```
order by customer_no desc
```

例 6 的含义是：在 c:\data\customer 数据表中，以 customer_no 的降序显示该字段。若需以升序显示时，用 Asc 取代 desc 或者不加说明。

4. 其他命令示例

例 1：修改数据 UPDATE 语句

```
update goods
```

```
set city = '长沙'
```

```
where goods.city = '南京'
```

含义：选择 goods 表中 city='南京'的记录，将其中的 city 字段改成“长沙”。其中 update、set 均为保留字。

例 2：插入记录 INSERT 语句

```
insert into goods (part_no, city)
```

```
values ( 'Te0094', '长沙' )
```

语句的含义是，将一条新记录插入 goods 表中，字段 part_no、city 的值分别是“Te0094”、“长沙”。其中 insert、values 为保留字。

例 3：删除记录 DELETE 语句

```
delete
```

```
from goods
```

```
where part_no = 'Te0095'
```

语句的含义是，删除 goods 表中 part_no='Te0095'的记录，其中 delete 为保留字。

16.3 记录集对象的 ASP 代码

1. 在连接对象的基础上创建记录集对象

在连接对象的基础上创建记录集时的 ASP 代码如下。

```
<%
var cn = Server.CreateObject("ADODB.Connection");
// 利用 Server 对象创建连接对象的实例
cn.Open("DSN=MisDatabase","admin","1111"); // 通过数据源名连接数据库
var strSQL = "SELECT * FROM Customer";      // 定义 SQL 语句
var rs = cn.Execute(strSQL);                 // 执行 SQL 语句创建记录集
%>
```

前两句是创建连接对象，其中假定数据源名是 MisDatabase；admin 是用户名；1111 是口令。后两句是创建记录集。

```
var rs=cn.Execute (strSQL);
```

其中：rs 为记录集名，cn 为连接对象，strSQL 是 SQL 语句。这条语句表明记录集是在连接对象的基础上，通过执行 SQL 语句来创立的。

2. 单独创建记录集对象

记录集对象也可以独立于数据库连接对象来创建。此时，必须提供一个连接字符串和一个命令字符串。

连接字符串定义了如何连接到一个数据库，命令字符串定义了一个查询操作或者存储过程，并指示如何将记录提交到 Recordset 对象中。此时的 ASP 代码如下。

```
<%
var rs = Server.CreateObject("ADODB.Recordset"); // 创建记录集对象
rs.ActiveConnection = "dsn=MisDatabase;uid=admin;pwd=1111;"; // 设置连接字符串
rs.Source = "SELECT * FROM Customer"; // 设置 SQL 命令字符串
rs.Open(); // 打开记录集
%>
```

这里是先创建记录集对象，再通过连接字符串和 SQL 给记录集赋予具体内容，直到用 rs.Open()打开记录集时，创建工作才算完成。在 Dreamweaver UltraDev 系统的源代码中都是采用这种方式创建记录集的。

16.4 利用记录集操作数据

前面对数据库的操作中都是一次只操作一条记录，被操作的记录称为“当前记录”，而记录指针就指向“当前记录”，为了能够操作多条记录，必须使记录指针在多条记录之间移动。

设 rs1 为记录集名，记录指针移动的命令如下。

rs1.MoveFirst：指针移至第一条记录。

rs1.MoveLast：指针移至最后一条记录。

rs1.MoveNext：指针移至下一条记录。

rs1.MovePrevious：指针移至前一条记录。

rs1.move n, start：

其中 n 为整数，表示移动的记录数，这个值可正也可负，正时表示向后移动，负时表示向前移动。

start 为一选择变量。代表指针移动的起点。0 是默认值，用来表明当前记录是移动的起点；1 表明第一条记录是移动的起点，2 表明最后一条记录是移动的起点。

16.5 在 Dreamweaver UltraDev 中创建记录集的步骤

下面介绍利用系统提供的工具创建记录集的步骤。由于记录集与网页捆绑在一起，因此创建前必须先打开网页。

(1) 打开网页后选择菜单 window | Data Bindings 项，打开数据绑定窗口。

(2) 在 Data Bindings 窗口中单击“ + ”按钮后，选择 Recordset(Query) 项，将弹出如图 16.1 所示的界面。

(3) 单击“ Simple ”按钮，将弹出一个如图 16.1 所示的界面。在这个界面中，借助一些可视化操作可以很方便地建立起记录集来。

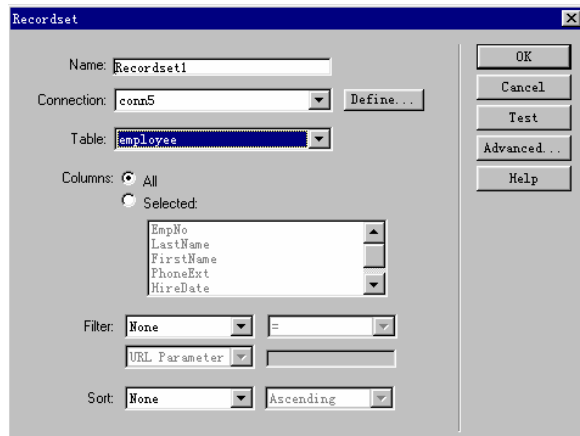


图 16.1 设置记录集的界面

(4) 在窗口界面中设置以下参数。

- Name：记录集名。图中填入的是 Recordset1。
- Connection：连接对象名，在下拉列表中选择。如果还没有设置连接对象时，可单击右边的“ Define ”按钮进行设置。
- Table：数据表名。本例中选用的是“ employee ”数据表。
- Column：字段栏。可选用全部(All)也可选用一部分(Selected)。如果选择 Selected 选项时，还需在下面的窗口中选择使用的字段。
- Filter：显示记录的条件(过滤)。在同步等的应用中将用到这一参数。
- Sort：排序的要求。右边有两下拉列表中，其中靠左边的用于选择排序字段，右边的用于确定排序的要求：是升序(Ascending)还是降序(Descending)。
- (5) 单击“ Test ”按钮以检查记录集设置的结果。如果合乎要求时，单击“ OK ”按钮。
- (6) 在 Data Binding 窗口中，将出现新 Recordset 名以及它所包括的字段。
- 当需要给记录集设置更多条件时单击“ Advanced ”按钮，打开如图 16.2 所示的界面。

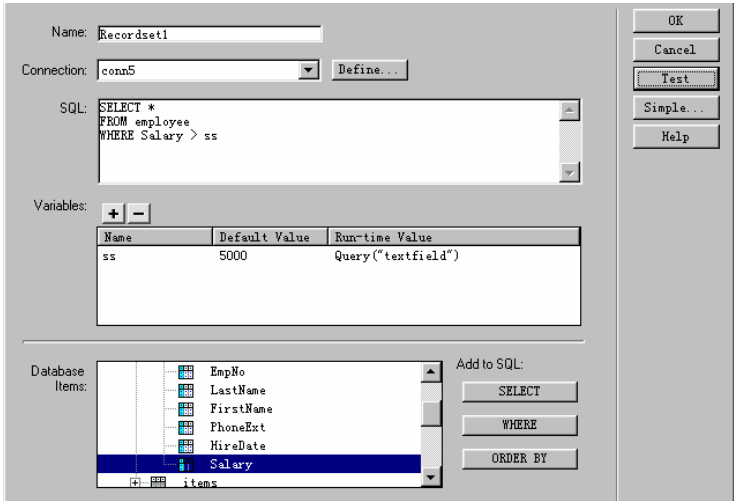


图 16.2 进一步设置记录集界面

这是一个编写 SQL 语句的界面。在上面填写记录集名、连接对象名。SQL 窗口用于编写创建记录集的 SQL 语句，Variables 窗口用来确定 SQL 中的待定参数，右下方 3 个按钮，SELECT、WHERE、ORDER BY 是用来帮助书写 SQL 语句的。在后面的章节中将结合应用讲述对它们的使用方法。

16.6 管理记录集

16.6.1 复制记录集

记录集属于某个特定的网页。若多个网页的记录集设置相同时，可以将一个网页的设置拷贝到其他网页中，以减少重复劳动。步骤如下。

- (1) 打开已经设置好记录集的网页。
- (2) 进入数据绑定(Data Bindings)窗口，如图 16.3 所示。

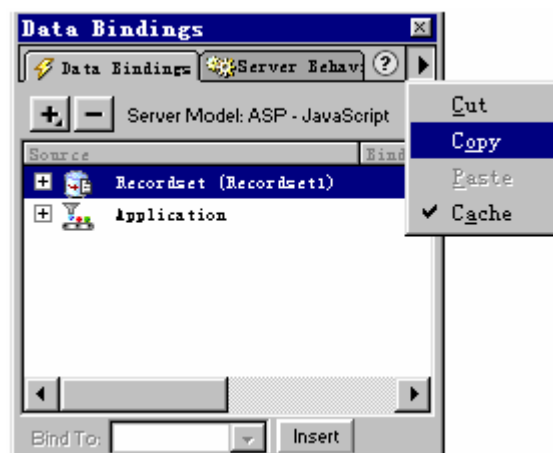


图 16.3 记录集拷贝

(3) 选择要拷贝的记录集。

(4) 单击右上角的向右箭头，在弹出的菜单中选择 Copy 项；或者用鼠标右击数据绑定窗口，在弹出的快捷菜单中选择 Copy 项。

(5) 打开要粘贴的网页。

(6) 右击它的数据绑定窗口或者单击右上方的向右箭头，在弹出的菜单中选用 Paste 选项。

16.6.2 编辑记录集属性

可按以下步骤对记录集的属性进行编辑。

(1) 在数据绑定窗口中选择记录集。

(2) 打开记录集的属性面板，如图 16.4 所示。如果看不见属性面板时，选择菜单 Window|Property 项。

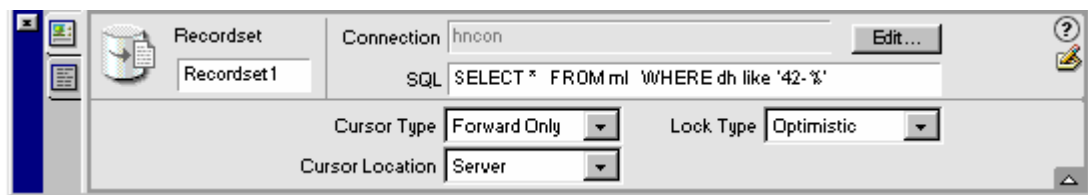


图 16.4 记录集的属性面板

(3) 在 Recordset 文本框中可改写记录集名称。

(4) 单击“ Edit ”按钮或者直接在数据绑定窗口中双击记录集名都可以打开如图 16.2 所示的对话框，在窗口中可以重新对记录集进行设定。

(5) 也可以直接在属性面板中改变 SQL 语句。

(6) 在 Cursor Type(游标类型) 的下拉列表中包括以下几种选择。

■ Forward Only (只向前) : 此项为缺省选择。选择此项鼠标指针只能在记录集中向前移动。此设定是将记录集的 Cursor Type 的属性值设为 0。

■ **Keyset (键集)**: 选择此项鼠标指针可以在记录集中向前或向后移动, 如果另一个用户删除或改变了一条记录, 记录集中将反映这个变化。但是, 如果另一个用户增添了一条新记录, 新记录不会出现在记录集中。设定“ 键集游标 ” 实际上是将记录集的 Cursor Type 属性值设为 1。

■ **Dynamic (动态)**: 选择此项鼠标指针可以在记录集中前后移动。其他用户对记录的任何改变都会反映到记录集中来, 设置“ 动态游标 ” 时是将记录集的 Cursor Type 的属性值设为 2。

■ **Static (静态)**: 选择此项鼠标指针可以在记录集中前后移动, 但是它不能反映其他用户对记录集的改变。设置“ 静态游标 ” 时是将记录集的 Cursor Type 的属性值设为 3。

(7) 数据锁定(Lock Type) 的选择。

在网络系统中, 几个程序同时更新同一条记录的情况是很容易发生的, 数据锁定就是对这种情况的处理原则。选择的项目有以下几种。

■ **Read Only (只读)**: 此项为缺省选择。选择此项时将记录集设成只读状态, 选择此项是将 LockType 的属性值设为 1。

■ **Pessimistic (悲观)**: 此项选择对应的 LockType 属性值为 2。选择此项是采用严格的锁定方式, 当程序通过数据集改变某条记录时(例如对该记录赋值), 即进入锁定状态, 直到执行完 Update 语句或者离开该记录时才开锁。

■ **Optimistic (乐观)**: 此项选择对应的 LockType 属性值为 3。选择此项, 只有执行 Update 语句修改记录时, 记录才进入锁定状态。

■ **BatchOptimistic (批量乐观)**: 此项选择对应的 LockType 属性值为 4。选择此项后, 更改记录时, 暂不把数据写入数据库, 直到调用 UpdateBatch 函数时才将整批更新数据写入数据库, 也只有在这个时候才锁定被更新的记录。使用这一设定时更新和取消更新的语句不再是 Update 和 CancelUpdate 而是 UpdateBatch 和 CancelBatch。

锁定模式与执行效率是成反比的, 锁定模式越严格, 安全程度越高, 但执行效率相对来说也越低。应根据对数据库安全程度的要求来选择锁定模式。有些信息发布的网页, 不含更新的记录时使用乐观锁定的方式较好。

(8) 在 Cursor Location (游标提供) 的下拉列表中有以下几种选择。

■ **Serve (服务器)**: 表明由服务器提供游标, 实际上是将记录集的 CursorLocation 属性设为 2。

■ **Client (客户端)**: 表明由客户端游标库提供游标, 实际上是将记录集的 CursorLocation 属性设为 3。

(9) 设置完毕后, 单击属性板以外的位置, 即完成了对记录集属性的编辑工作。

16.6.3 删除记录集

按以下步骤删除记录集。

(1) 进入数据绑定窗口。

(2) 单击要删除的记录集。

(3) 单击窗口上方的“ - ” 按钮, 记录集的名字将在数据绑定窗口中消失。

第 17 章 动态数据绑定

一张单纯用于发布信息的网页，它的内容往往是固定的，虽然使用 DHTML 技术可以使网页中一些元素“动”起来，但那只是表现形式上的变化。如果想改变网页的内容，只有修改或重新对网页进行设计，这种网页称之为静态网页。

另一种网页称之为动态网页，它除有自己固定的内容以外，还能将服务器端送来的信息（如数据库信息）显示在网页上。由于服务器输出的变化，网页的内容不再是固定的。这样的网页可以适用多种情况的需要，做到“一网多用”。

实现动态网页的关键是将网页元素与服务器输出捆绑在一起。数据捆绑包括被绑定对象与动态数据源两个方面。在 Dreamweaver UltraDev 中，几乎所有网页元素的属性，包括文本、图片、各类输入框、按钮以及下拉列表的属性都可以成为数据捆绑的对象。而动态数据源主要是服务器输出的各类对象。

Dreamweaver UltraDev 提供了多种数据绑定的方法，例如利用数据绑定窗口，利用属性面板，利用服务器行为，还可以直接书写代码等方式来实现动态数据绑定。

17.1 动态数据源

17.1.1 什么是动态数据源

动态数据源就是由服务器输出的各种类型的数据，包括以下几个方面。

- 记录集对象(Recordset)。这是使用得最多的一种动态数据源。
- 命令对象(Command)。并不是所有的命令对象都能够成为数据源。只有命令对象能够返回记录集时才可以成为数据源。
- ASP 的 Request 对象获取的数据。
- ASP 的 Session 对象定义的会话作用域变量。
- ASP 的 Application 对象定义的应用程序域变量。

这些数据源实际上分成两大类，一类是记录集 Recordset 对象。命令对象实质上属于这一类；另一类是服务器对象中的共享变量，如 Request、Session、Application 等。

不同数据源的作用域也各不相同，Recordset 与 Request 对象的作用域是当前网页，只在设置的网页中才起作用；Session 变量的作用域是单用户多张网页组成的会话域；Application 的作用域是整个应用程序。

17.1.2 数据绑定的基本步骤

在网页中实现动态数据绑定，通常包括以下 4 个步骤。

- (1) 确定动态数据源。

- (2) 指定数据绑定对象或者动态数据插入的位置。
- (3) 将数据源从数据绑定窗口(或者其他方法) 放置到预定位置。此时 Dreamweaver UltraDev 会自动添加相应的数据绑定脚本。
- (4) 必要时还可以添加其他服务器行为。

17.1.3 利用数据绑定窗口设置数据源

Dreamweaver UltraDev 的数据绑定窗口(Data Bindings), 就是用来管理数据源的窗口。下面先集中介绍利用数据绑定窗口设置动态数据源的方法。

选择菜单 Window|Data Bindings 选项, 就可以打开数据绑定窗口, 单击窗口中的“ + ”按钮, 在弹出的下拉列表列出了各种类型的数据源, 其情况如图 17.1 所示。

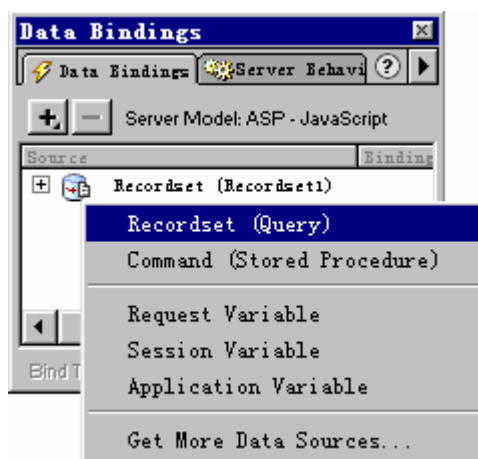


图 17.1 动态数据源的类型

窗口中列出的动态数据源包括 6 种。下面对其中几种数据源的设置方法加以说明。

1. 设置 Request 类型的数据源

在 ASP 中, 服务器通过 Request 对象可以获取从浏览器端提交的数据, 这些数据又可以反馈到浏览器端去作为动态数据源。

当选择 Request Variable 选项时将弹出如图 17.2 所示的对话框。

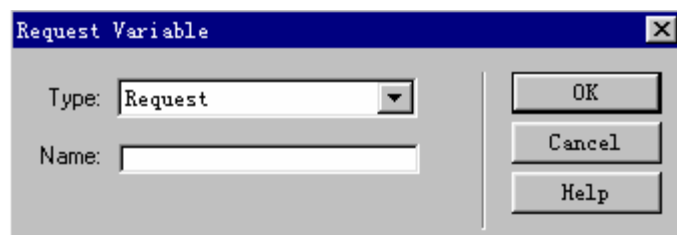


图 17.2 Request 对象数据源

在 Type 的下拉列表中包括 6 种类型, 如图 17.3 所示。

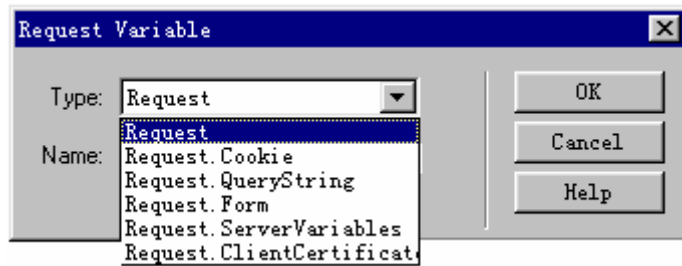


图 17.3 Request 数据源的种类

各种类型的 Request 的作用如下。

- Request：服务器自动匹配各种提交方法以获取浏览器端的数据。
- Request.QueryString：用于获取浏览器以 GET 方法提交的数据。
- Request.Form：用于获取浏览器以 POST 方法提交的数据。
- Request.ServerVariables：用于获取服务器端的环境变量。
- Request.Cookie：用于获取或保存浏览器端的 Cookie 变量。
- Request.ClientCertificate：用于获取有关客户的身份权限方面的数据。

其中 Request.QueryString、Request.Form、Request.Cookie 是用得比较多的数据源。采用 Request 方法时，系统能够自动匹配浏览器端提交数据的方法，适用面最广但效率稍低。

例如：浏览器端的表单中有 3 个文本输入框，分别命名为 Name、Sex、Job，表单用 Post 方式提交文本框的数据。在接收信息的网页中可按以下方法设置动态数据源。

选择图 17.3 中的 Request.Form 项，并在 Name 中分别填入 Name、Sex、Job，设置 3 个动态数据源。在数据绑定窗口中将出现如图 17.4 所示的结果。

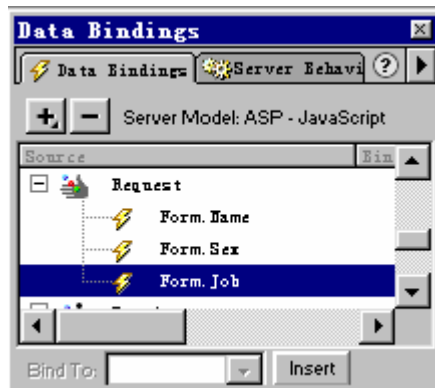


图 17.4 Request.Form 数据源

2. 定义 Session 类型的数据源

(1) 打开数据绑定窗口，单击“+”按钮，选择 Session Variable 选项，将弹出如图 17.5 所示的对话框。

(2) 填入 Session 的变量名(Name)。如本例中的 Customer。



图 17.5 Session 变量数据源

(3) 在数据绑定窗口中将出现设定的 Session 变量，如图 17.6 所示。

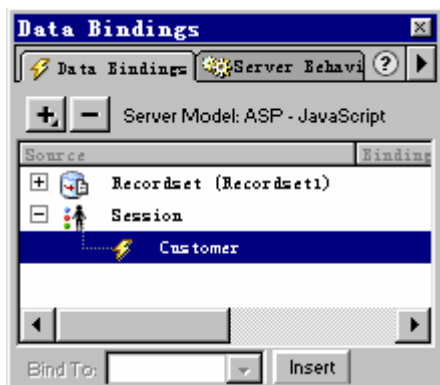


图 17.6 数据绑定窗口中的 Session 数据源

与此同时，同一会话的其他网页的数据绑定窗口中也会出现同样的 Session 变量 Customer。

3. 定义 Application 类型的数据源

(1) 打开数据绑定窗口，单击窗口中的“+”按钮，选择 Application Variable 选项，将弹出如图 17.7 所示的对话框。

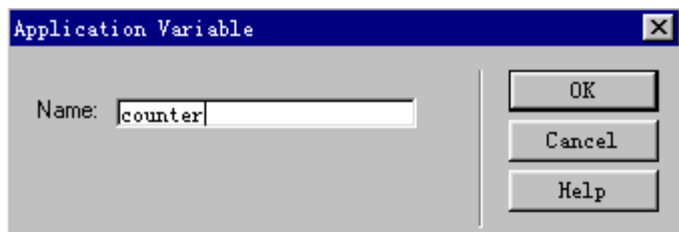


图 17.7 为 Application 数据源取名

(2) 填入 Application 的变量名(Name)。如本例中的 counter。

(3) 在数据绑定窗口中将出现定义的 Application 变量，如图 17.8 所示。

与此同时，同一应用程序的其他网页的数据绑定窗口中也将出现同样的 Application 变量 counter。

4. 获取更多的数据源

(1) 打开数据绑定窗口，单击“+”按钮，选择 Get More Data Sources 选项。

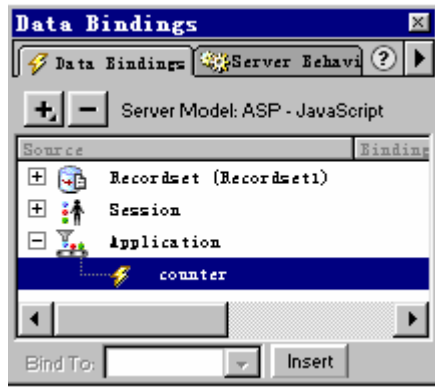


图 17.8 数据绑定窗口的 Application 数据源

(2) 系统将自动转向 Internet 中的下列网站：

WWW.macromedia.com / exchange / ultraDev/

在该网站中可以查看到更多的相关信息，或者从该网站下载一些有关的补充软件。

17.1.4 删除数据源

删除数据源的步骤如下。

(1) 在数据绑定窗口中选中要删除的数据源。

(2) 用鼠标单击窗口上方的“ - ”按钮。

(3) 有的数据源如 Session、Application 等在数据绑定窗口中，是一种树型结构。一个 Session 或 Application 下面可能包括多个变量，删除这样的数据源要先删除“ 叶 ”节点，然后才能删除“ 根 ”节点。

17.2 各类网页元素的数据绑定

17.2.1 文本框的动态数据绑定

1. 数据绑定的步骤

下面通过具体示例来说明文本框数据绑定的步骤。假定记录集中包括 编号 ”、“ 姓名 ” 等字段，现在将这些字段作为数据源与文本框进行动态绑定。

(1) 在网页中创建记录集。

(2) 在网页中放入一表单(Form)，表单内放进两个文本框，标上“ 编号 ”、“ 姓名 ” 等标签。选择其中一个文本框。

(3) 拉开记录集中各个字段并选择其中之一。

(4) 展开数据绑定窗口下面的 Bind To 下拉列表，从中选择准备将字段绑定到文本框的哪个属性上，如图 17.9 所示。

值得说明的是，不同网页元素的属性列表是不同的。

(5) 这里选择 input.value 属性，这是一个常用的属性，其作用是将记录集字段的值显示

在文本框中。

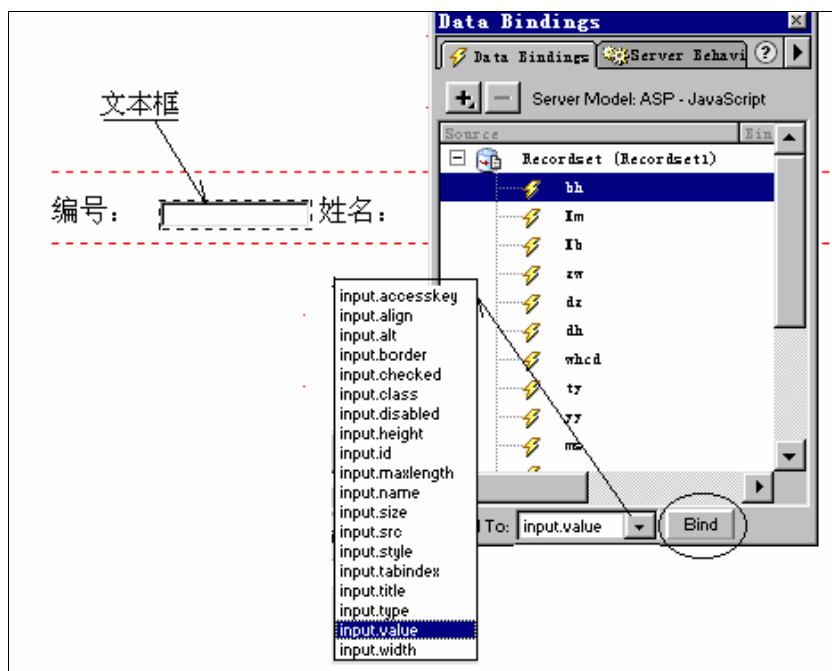


图 17.9 选择绑定的字段

(6) 单击“ Bind ”按钮实现数据绑定。

2. 理解代码

完成了前面的设定以后，进入网页代码窗口，将看到以下代码：

```
<input type="text" name="textfield"
value="<%= (Recordset1.Fields.Item("bh").Value) %>">
```

这段代码说明，文本框的 value 已经和记录集中的 bh 字段的值捆绑在一起。

17.2.2 图片的动态绑定

为了使网页中的图片能够跟随记录的变化而变化，可以进行图像的动态绑定，具体步骤如下。

- 先将数据表增加一字段（可设成文本型），分别写上各图片的名称（如：01.gif，02.jpg 等）。这些图片必须已经放在站点的目录中。

- 用记录集指向该数据表。

- 将 Objects 面板 Forms 页中的 Insert Image Field 图标拖放到网页的适当位置。

[注意]：在 Objects 面板的 Common 页中也有一个 Insert Image 图标，它与在 Form 页中的 Insert Image Field 图标的作用相似，都是用来显示图片。只不过在 Form 页中的图标可以使得图片与表单（Form）中的其他数据一道提交给服务器。

- 在数据绑定窗口中单击图片（zp）字段，并在 Bind To 的下拉列表中选择 img.src 属性，如图 17.10 所示。

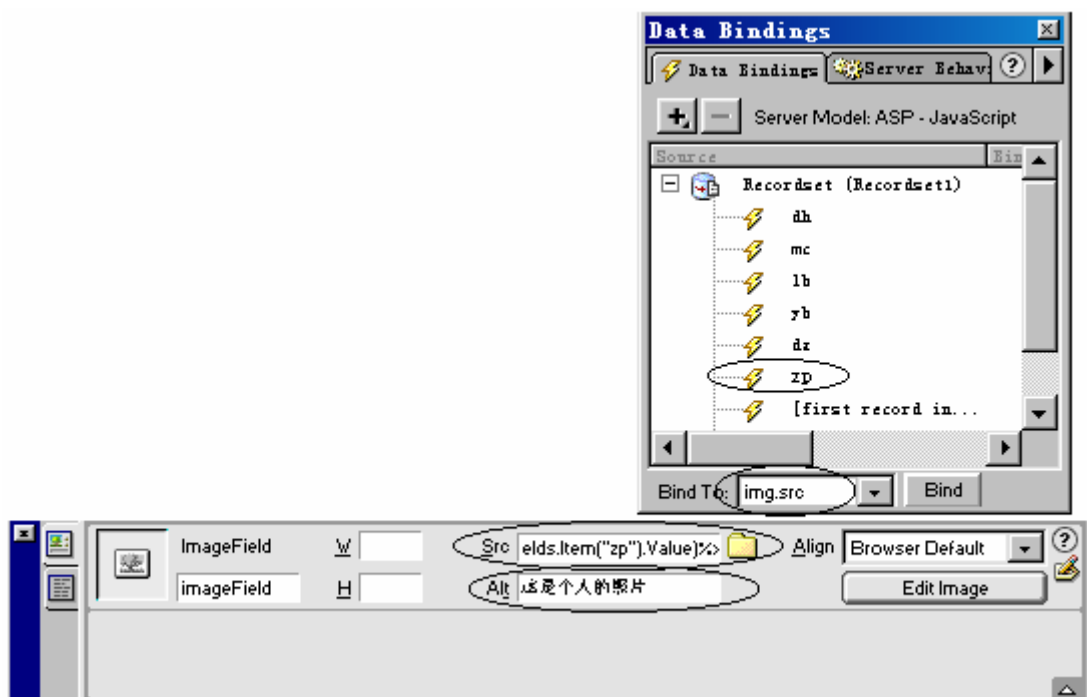


图 17.10 图片的动态数据绑定

- 单击“Bind”按钮以实现动态数据绑定。此时在图片的 Src 属性中将出现以下字符串：

`<%=Rs1.Fields.Item("zp").Value %>`

代码中 Rs1 代表记录集名。此段代码表明，图片的源文件名是记录集 zp 字段中的值。与此同时“Bind”按钮上的名字也变成了 unBind，这说明动态数据绑定成功。

当需要解脱两者之间的绑定关系时，先选择图片然后单击“unBind”按钮即可。

- 在图片属性窗口的 Alt 框体中填入替代的字符串。当浏览器类型或版本不支持图片时，将自动用这段字符串替代。

17.2.3 单选按钮与复选框的动态数据绑定

1. 单选按钮与复选框的比较

单选按钮和复选框通常都是放入表单中的元素，但它们的作用和使用方法却不相同。

如果多个单选按钮取相同的名字时，它们是同一组的单选按钮，在一组单选按钮中，一次最多只能作出一种选择。就是说同一组单选按钮之间存在着互斥的关系，选择其中之一时，其他被选中的选项将被自动取消。

对于复选框来说，通常情况下各个复选框都起有不同的名字，某些特定情况下也允许多个复选框同名（如 25 章的示例）。和单选按钮的最大不同点是，复选框允许同时作出一个或多个选择。如果各复选框不同名时，系统将分别对这些选择进行处理。如果多个被选中的复选框同名时，则将输出由各个被选中复选框的输出值组成的字符串，各复选框输出值之间用逗号隔离。

单选按钮与复选框的比较如图 17.11 所示。其中“个人爱好”用的是复选框；“个人最高学历”用的是单选按钮。

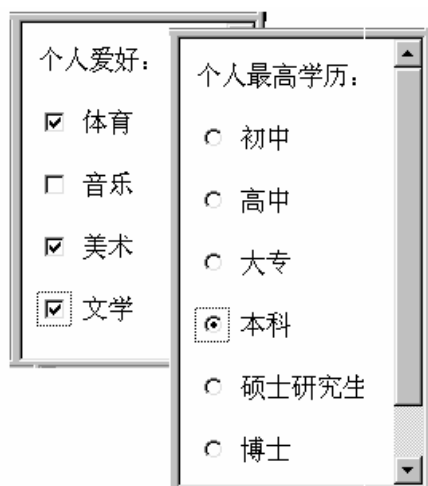


图 17.11 复选框与单选按钮

虽然一组单选按钮中能够提供多个选项，但实际应用中通常只使用两个互斥的选项。如，男、女；已婚、未婚；通过、未通过等。

2. 单选按钮的动态数据绑定

数据绑定的步骤

下面通过具体示例来说明单选按钮数据绑定的步骤。假定记录集中包括“性别”字段，现在将该字段作为数据源与一组单选按钮动态绑定。

(1) 为网页创建记录集，并放入一表单 (Form)，表单内放进两个同名的单选按钮，标上“男”、“女”标签。

(2) 选择主菜单 Window | Server Behaviors 打开服务器行为窗口，单击窗口中的“+”按钮，在弹出的菜单中选择 Dynamic Elements 项以弹出动态元素的选项，如图 17.12 所示。

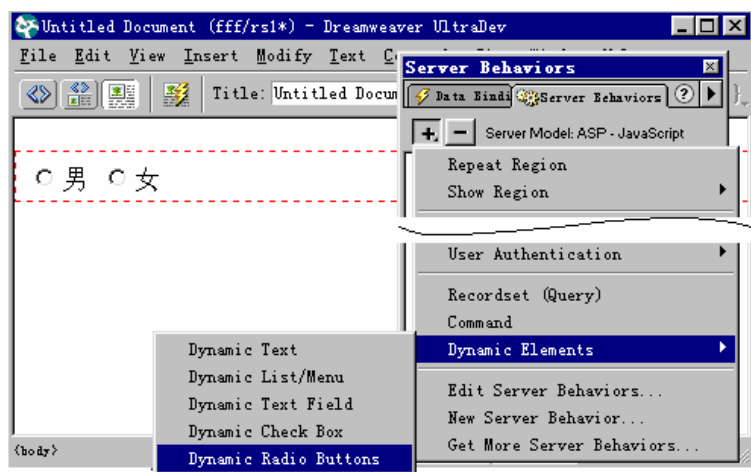


图 17.12 动态元素选项

(3) 选择 Dynamic Radio Buttons 项将弹出新的对话框，如图 17.13。在对话框的 Radio

Button Group 下拉列表中选择一组单选按钮，并在 Value 文本框中分别为每个按钮确定相应的值，如男、女等。

窗口中 Value 的作用是，当记录集字段的值与 Value 相同时，该按钮显示被选中状态，否则为未被选中状态。

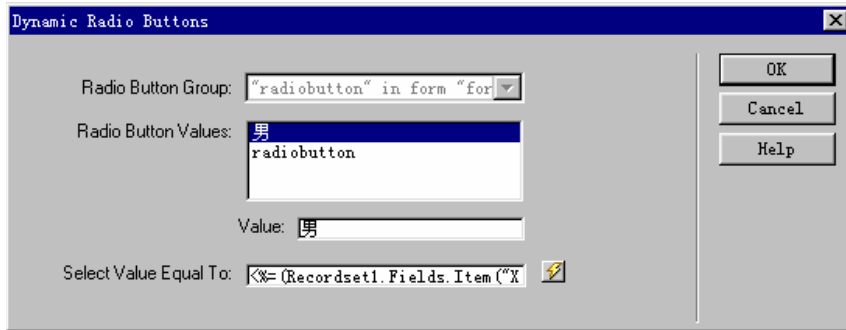


图 17.13 单选按钮的设置

(4) 单击 Select value Equal To 右边的“闪电”图标以建立与数据集字段的捆绑关系，此时将弹出 Dynamic Data 对话框，如图 17.14 所示。

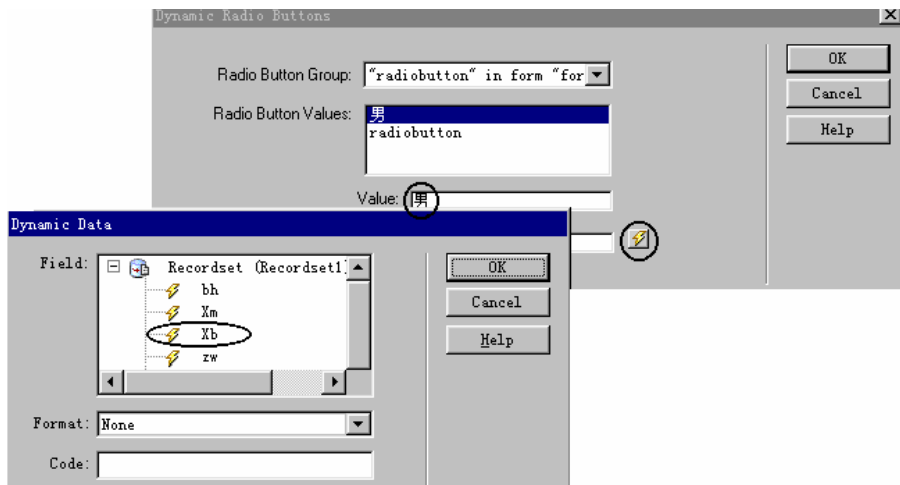


图 17.14 与 Xb 字段动态绑定

选择相应的字段(如，性别字段 Xb)。

(5) 单击“OK”按钮完成数据绑定工作。此时图 17.13 的 Select Value Equal To 的文本框中将出现以下代码：

```
<%=((Recordset1.Fields.Item("Xb").Value %>
```

理解代码

经过前面的设置后，打开网页的源代码，可以看见以下代码：

```
<input <%=((Recordset1.Fields.Item("Xb").Value == "男")?"CHECKED":"" )%>
type="radio" name="radiobutton" value="男"> 男
<input <%=((Recordset1.Fields.Item("Xb").Value == "女")?"CHECKED":"" )%>
```

```
type="radio" name="radiobutton" value="女"> 女
```

代码中使用的是“？”加“:”号的条件语句,第一个<input...>语句中的条件是记录集中性别(“Xb”)字段的值是否等于“男”,如果相等,选择“CHECKED”,代表该按钮被选中。如果不等于“男”,选用空“”,代表按钮没有被选中;第二个<input...>语句的含义与上一句相同,只不过用是否等于“女”作为判断的条件。

有的读者可能对这种条件语句不大熟悉,可将代码改用 if()语句表达如下:

```
<input<%= if (Recordset1.Fields.Item("Xb").Value == "男")
    Response.Write("CHECKED");
Else
    Response.Write(""); %>
Type="radio" name="radiobutton" value="男"> 男
```

3. 复选框的动态数据绑定

对复选框的数据绑定方法与单选按钮的绑定方法基本相同,只不过由于复选框可以进行多项选择,因此记录集中应包括多个字段,每个复选框对应其中一个字段。

动态数据绑定的步骤

下面通过选择“个人爱好”的具体示例来说明对复选框数据绑定的步骤。假定记录集中包括“体育(ty)”、“音乐(yy)”、“美术(ms)”、“文学(wx)”几个字段,当对某项有爱好时,在该字段中填入“true”。

数据绑定的步骤如下。

(1) 为网页创建记录集并放入一表单(Form),表单内放 4 个复选框,标上“体育”、“音乐”、“美术”、“文学”等标签。

(2) 先选择一复选框,然后用主菜单 Window | Server Behaviors 选项打开服务器行为窗口,单击窗口中的“+”按钮,在弹出的菜单中选用 Dynamic Elements 后再选用 Dynamic Check Box 选项,将弹出如图 17.15 所示的对话框。

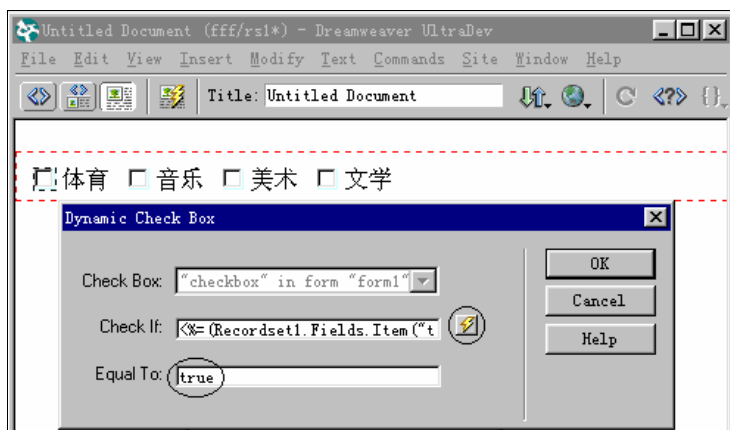


图 17.15 复选框的数据绑定

- 单击 Check If 右边的“闪电”图标,建立与记录集某个字段的捆绑联系。
- 在 Equal To 文本框中填入 true。代表如果该字段的值是“true”时,复选框以被选中

的状态显示。

■ 单击“ OK ”按钮完成对该复选框的设置。按照同样办法为其他复选框逐个进行数据绑定。

理解代码

打开网页的源代码窗口，将看到以下代码。

```
<input <%=((Recordset1.Fields.Item("ty").Value == "true")?"CHECKED":"")%>
    type="checkbox" name="checkbox" value="checkbox"> 体育
<input <%=((Recordset1.Fields.Item("yy").Value == "true")?"CHECKED":"")%>
    type="checkbox" name="checkbox2" value="checkbox"> 音乐
<input <%=((Recordset1.Fields.Item("ms").Value == "true")?"CHECKED":"")%>
    type="checkbox" name="checkbox3" value="checkbox"> 美术
<input <%=((Recordset1.Fields.Item("wx").Value == "文学")?"CHECKED":"")%>
    type="checkbox" name="checkbox4" value="checkbox"> 文学
```

代码使用的是条件语句，第一个<input...>语句中的条件是记录集中体育“ ty ”字段的值是否等于“ true ”，如果相等，选择“ CHECKED ”，代表该复选框被选中。如果不等于“ true ”，选择空“ ”，代表复选框没有被选中。其他几个<input...>语句的含义与此相同。只不过对应的字段不同而已。

17.2.4 列表/下拉列表(List/Menu) 的动态数据绑定

1. 数据绑定的步骤

为了给列表/下拉列表进行动态数据绑定，按照以下步骤进行。

(1) 在网页的表单中放置一列表/下拉列表，然后到它的属性面板中去设置属性。列表/下拉列表的属性面板情况如图 17.16 所示。

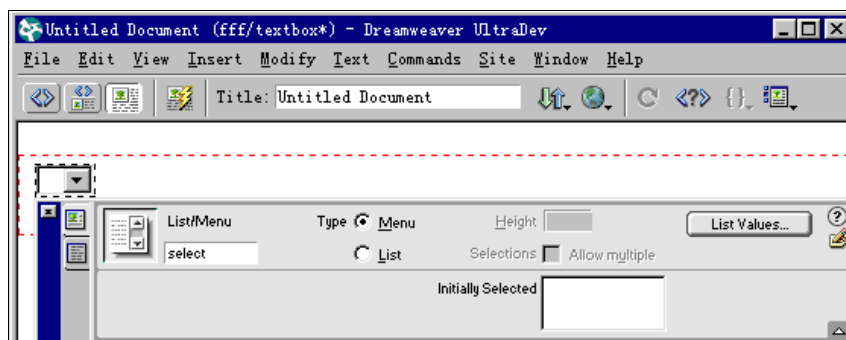


图 17.16 下拉列表的属性面板

(2) Type 属性。类型 Type 属性包括两种：Menu 和 List。

- Menu：设置成下拉列表。
- List：设置成列表框。

这里选用下拉列表(Menu)。

(3) 为下拉列表设置选项。

为了给下拉列表设置选项，单击“List Values”按钮以打开 List Values 对话框，如图 17.17 所示。

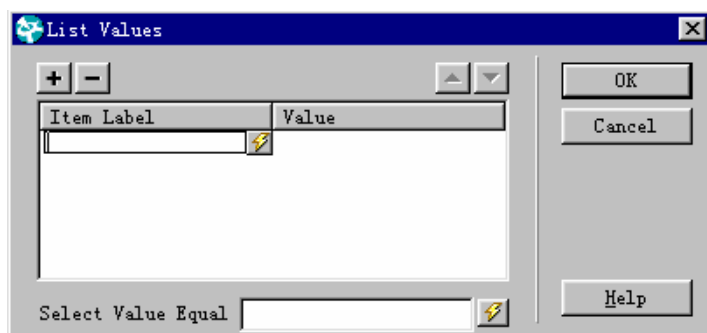


图 17.17 菜单选项的确定

窗口包括 Item Label 与 Value 两栏，用来为下拉列表设定显示值（Label）和使用值（Value）。

为什么需要两种值呢？这是为了使用方便，以商品分类为例，对系统内部来说，使用字符或数字代码表示比较方便，它能使程序代码变得更加紧凑。可是在客户面前商品的名称只用代码显示将会给客户带来极大的不便。这里就出现了两种不同的要求，需要采用两种不同的显示方式来解决。这种情况下，列表项的内部使用值（Value）可以采用代码，而外部的显示值（Label）可以采用习惯的商品名称。

当然很多情况下这两种值是相同的，此时只需设 Item Label 就可以了，不必设 Value 的值，系统将自动将 Label 值赋给 Value，两者取值相同。

（4）单击“+”或“-”按钮可以增加或删除列表框或下拉列表的项目。各项目的内容可以单独编写也可以利用数据表中的数据。如果想利用数据表的数据时，需将各个选项与数据源捆绑起来。捆绑的方法是：单击 Item Label 右边的“闪电”（) 图标，将弹出 Dynamic Data 窗口，如图 17.18 所示。

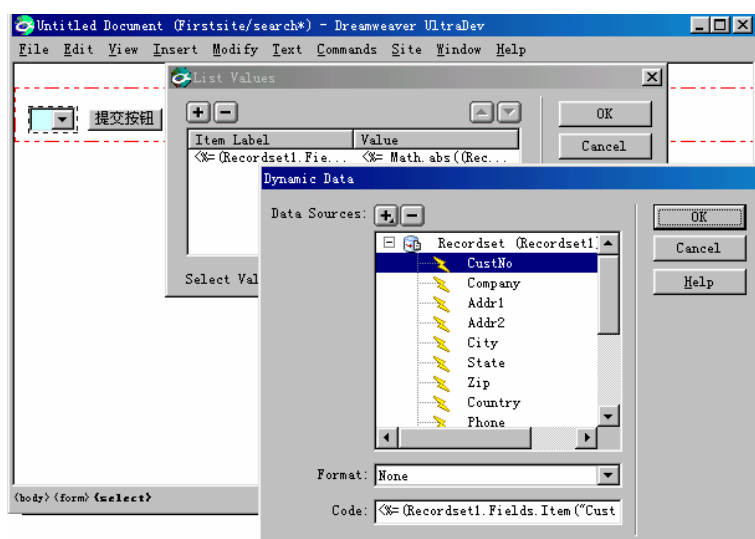


图 17.18 下拉列表项与记录集字段绑定

(5) 选择菜单项对应的字段后单击“ OK ”按钮。此时窗口下面的 Code 窗口中将出现了以下代码：

```
<%= (Recordset1.Fields.Item("CustNo").Value) %>
表明该菜单项已经与记录集的 CustNo 字段动态捆绑。
2. 理解源代码
进入网页的代码窗口，将看见以下代码。
<select name="select">
<%
while (!Recordset1.EOF)
{
%>
<option value="<%= (Recordset1.Fields.Item("bh").Value) %>" >
<%= (Recordset1.Fields.Item("bh").Value) %>
</option>
<%
Recordset1.MoveNext();
}
%>
</select>
```

这是一段循环语句，通过这段程序可以看出，系统是通过循环程序将数据表中某字段的值取出来逐个赋给下拉列表的选项的。

正由于它使用了循环语句，改变了记录指针的位置，因此如果将它的动态绑定放在其他网页元素动态绑定的前面时，会影响它后面元素动态绑定的结果，带来新的错误。这一点使用中要特别注意。

17.2.5 动态文本

网页中的标题以及其他说明性的字符串通常是一种静态文本，现在也可以用动态数据（下称动态文本）去替换其中的某些部分，或者直接插入到原有文本的中间。例如商品名称、特性以及价格等都是随商品品种不同而不同的数据。将这部分文本与数据表相对应的字段捆绑在一起，就可以在同一张网页中显示不同商品的数据了。

对待动态文本可以像对待普通文本一样使用各种文本格式或 CSS 样式。如果在原来的静态文本中已经定义好了格式，则替换后的动态文本将自动执行原有静态文本的格式。当然也可以在数据绑定窗口中为动态文本另外确定显示格式，设置的方法将在后面叙述。

设置动态文本的步骤是：先选择静态文本或者指定插入的位置，然后单击数据绑定窗口中的动态数据源，再单击窗口下方的“ Insert ”按钮或者将动态数据直接拖放到替换或插入的位置。

例如：用一张网页摘录各种报纸的发行情况。

(1) 打开网页，并在网页中建立静态文本。

(2) 创建记录集指向报纸发行的数据表。

(3) 选择数据绑定窗口中的各个字段,如:名称 mc)、类别 lb)、代号 dh)、地址 dz) 等取代网页中相对应的静态文本。

此时,网页中的静态字符串都变成了动态文本。当记录改变时,各文本字段都将相应地改变,如图 17.19 所示。

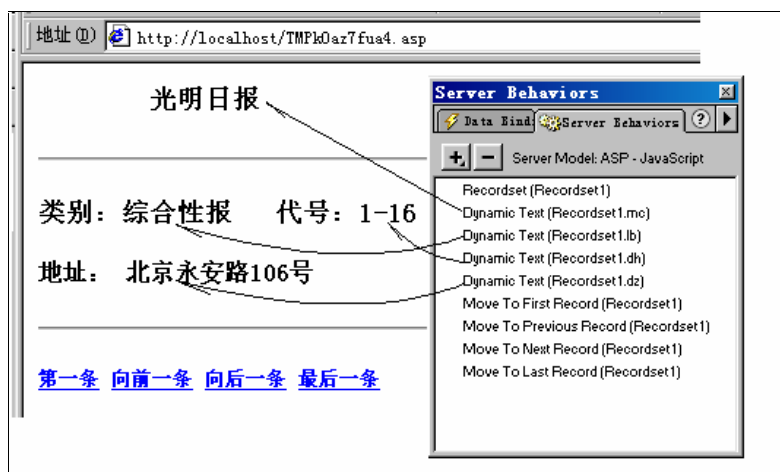


图 17.19 动态文本

图中箭头代表用动态文本取代静态文本的情况。

17.2.6 动态数据格式

改变动态数据的显示格式只对当前网页起作用,并不影响数据表的内容。例如,某条记录中的商品价格是 17.989,可以通过选择小数点后只取两位的格式,让它只显示 17.99,第三位自动四舍五入。其他对日期/时间格式、货币格式、百分比格式等都可根据需求和人们使用的习惯来设定。设定的一般过程如下。

(1) 选择动态数据源。

(2) 在数据绑定窗口中单击右边的 Format 栏的向下箭头。将出现一新的窗口,如图 17.20 所示。如果看不见 Format 栏时,将窗口拉大。

(3) 在窗口的下拉列表中选择 Edit Format List 以打开格式编辑表。

(4) 双击其中一选项,然后进行编辑。

(5) 确认后单击“OK”按钮。

允许给动态数据创建新格式,方法是在前面第三步的基础上单击“+”按钮,选择一种 format type(例如选择 Currency),然后定义新格式并单击“OK”按钮;为新格式命名后单击“OK”按钮关闭 Edit Format 对话框。

下面通过几个示例说明设置动态数据格式的过程。

例 1: 改变日期/时间格式

选择 Date/Time 项,将弹出如图 17.21 所示的界面。界面中列举了各类日期/时间的格式,选择其中之一,该数据源的日期/时间就采用该形式来显示。

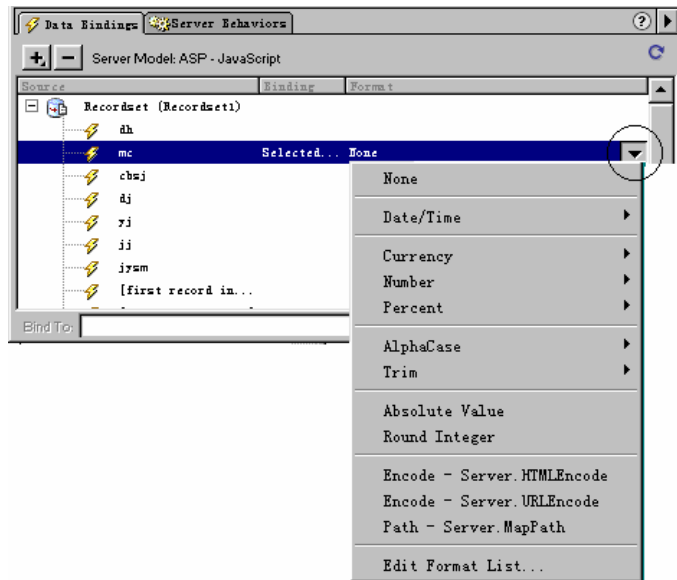


图 17.20 动态数据格式

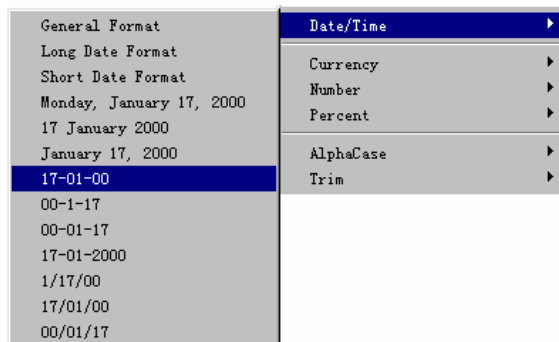


图 17.21 日期 / 时间格式选择

例 2：改变数字格式

选择 Number 项，将弹出如图 17.22 所示的菜单，菜单中列举了各类数字的格式。如，只取两位小数(2 Decimal Places) 四舍五入后取整(Rounded To Integer) 小数前加 0/不加 0 (Leading 0 If Fractional / No Leading 0 If Fractional) 等项。选择其中之一，该数据源就按照该格式显示数字。

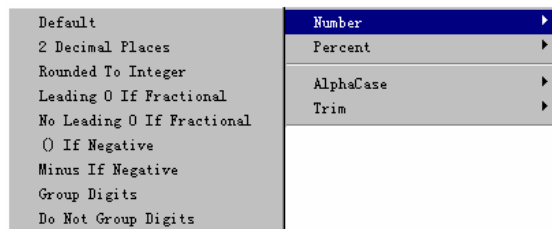


图 17.22 数字格式选择

例 3：剔除字符串外的空格

选择 Trim 项时将弹出如图 17.23 所示的菜单，可以从中选择：是去掉字符串左边的空格（ Left ） 右边的空格（ Right ） 还是去掉前后的空格（ Both ）。

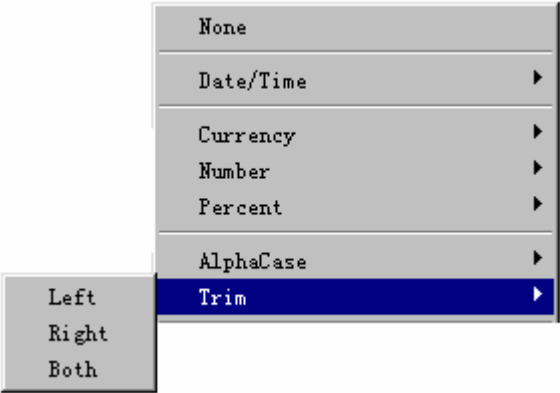


图 17.23 剔除字符串外空格的选择

17.3 应用举例

下面以一个简单的人事档案为例说明动态数据绑定的方法。

（ 1 ）数据表。数据表的字段及内容如表 17.1 所示。

表 17.1 人事档案简表

编号 (bh)	姓名 (xm)	性别 (xb)	电话 (dh)	地址(dz)	体育 (ty)	音乐 (yy)	美术 (ms)	文学 (wx)	照片 (zp)
0010	刘伟大	男	07314485644	湖南宏大厂 技术科	true		true		001.jpg
0011	李华丽	女	07315541243	湖南宏大厂 财务科			true	true	002.jpg
0012	陈功	女	07315564586	湖南宏大厂 一车间	true	true			003.jpg

（ 2 ）显示界面。

显示界面如图 17.24 所示。在这个界面中使用了文本框、单选按钮、复选框以及图片等几种网页元素。把这些元素放置在表格中可以使得它们的排列比较整齐美观。

（ 3 ）为了进行数据绑定，需在数据绑定窗口中创建记录集指向人事档案简表。然后将记录集的各字段与网页元素动态绑定。其中。

- 编号、姓名、电话、地址几个文本框通过 input.value 属性分别与 bh、xm、dh、dz 几个字段动态绑定。
- 两个单选按钮取相同的名字与 xb 字段动态绑定。



图 17.24 应用示例

- 4 个复选框取不同的名字分别与 ty、yy、ms、wx 字段动态绑定。
- 将 Objects 面板的 Forms 页中的 Insert Image Field 图标拖放到照片的框体中。并将图片的 img.src 属性与 zp 字段动态捆绑。
- 用鼠标选择标题“个人的简要档案”中的“个人”两个字，然后将 xm 字段拖到“个人”上面以实现文本的动态数据绑定。这样做的结果是，例如当指向“成红”的记录时表格上面的标题将变成：

成红的简要档案

第 18 章 Web 数据库查询与显示

18.1 概述

数据库查询的目的,是从数据库中查出符合条件的记录。例如网上选购商品时查看某类商品的情况;从网络图书馆中查阅某类图书、资料;从人事档案中查阅某人的历史数据等。数据库查询是 Web 应用中使用得最为频繁、最重要的操作。网上的下拉列表数据库操作,如修改记录、增加记录、删除记录等,都常常是在查询的基础上进行的。

整个查询工作包括两个方面,一方面要提供查询条件,另一方面是执行查询操作并显示查询结果。查询的过程如图 18.1 所示。

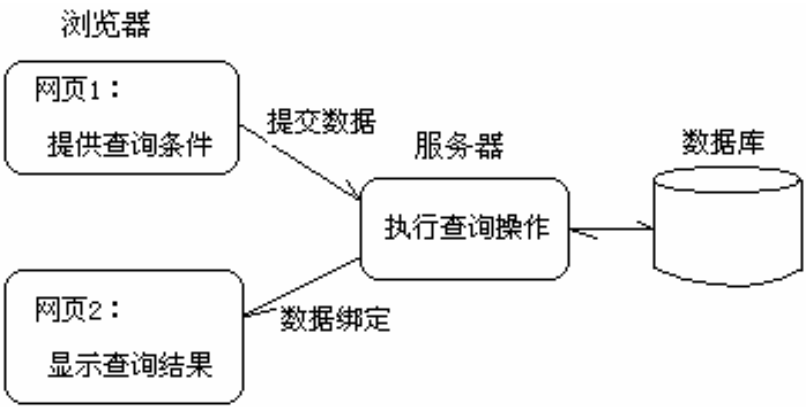


图 18.1 查询过程示意图

为了进行查询,通常要设置两张网页,首先在网页 1 中提供查询条件,然后将数据提交给服务器;服务器执行查询操作,然后将查询结果显示在网页 2 中。服务器执行查询操作的方法就是根据查询条件生成新的记录集,显示查询结果的方法就是将记录集上的字段与网页 2 的网页元素捆绑在一起。执行查询操作与显示查询结果的代码都写在第二张网页上。

18.2 执行查询的操作

18.2.1 准备工作

下面结合几个示例来说明查询的方法。

(1) 设数据表名为 Employee, 它的字段如表 18.1 所示。

表 18.1

数据表字段。

字段名	EmpNo	LastName	Sex	Branch	PhoneExt	HireDate	Salary
含义	职工编号	姓名	性别	部门	电话	聘用日期	薪金
字段类型	数字	文本	文本	文本	文本	日期/时间	数字

(2) 创建两张网页，一张用于输入查询条件，命名为 search.asp；另一张执行查询操作并显示结果，命名为 result.asp。

18.2.2 单一条件查询

1. 在 search.asp 中完成以下设置。

(1) 打开 search.asp 网页，放入一表单(Form)，为表单设置以下参数。

- Action：指向结果网页 result.asp。
- Method：提交方法选用 Post。

(2) 为网页创建一记录集指向 Employee 数据表。本记录集主要用于与下拉菜单进行数据捆绑。

(3) 表单中放一下拉菜单，取名 Select，菜单的选项与记录集的职工编号(EmpNo) 动态捆绑。

2. 打开 result.asp，完成以下设置

(1) 打开数据绑定窗口(Data Bindings)，单击窗口中的 + 按钮，选择 Recordset (Query) 以打开记录集设置窗口，单击“ Advanced ”按钮进入 Advanced 窗口，如图 18.2 所示。

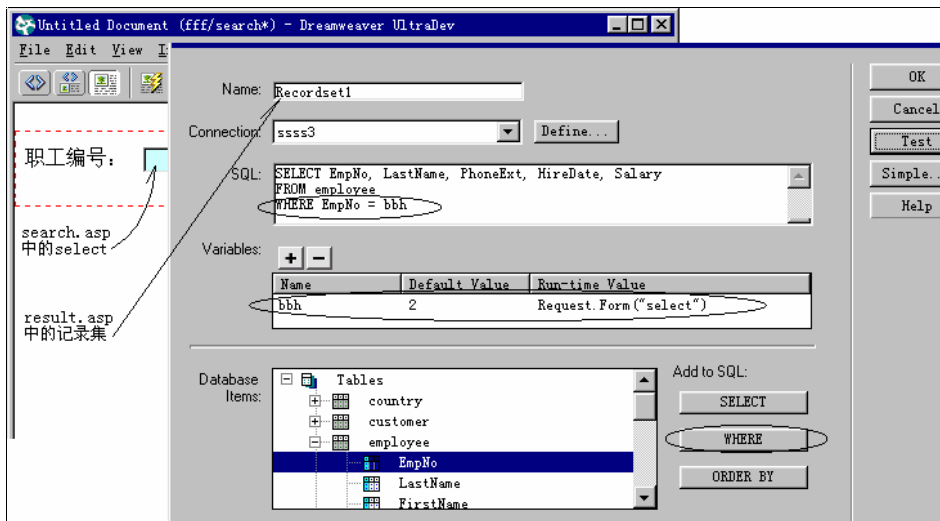


图 18.2 创建记录集

(2) 为记录集设置参数。

记录集中的参数包括以下几个方面：

- 填写 Name、Connection 等项。
- 在 SQL 框体中直接写入或者借助于 WHERE 按钮的帮助写入 SQL 语句。本例的语句

如下。

```
select *
from employee
where EmpNo = bbh
```

语句中的 bbh 为待定变量，它的值在程序运行中才能确定，确定的方法在 Variables 框体中确定。

■ 单击 Variables 框体上方的 “+” 按钮，然后分别在 Name 栏下面填写待定变量名 bbh，在 Default Value 栏下面填写缺省值，Run-time Value 栏下面填写语句 Request.Form("select")。

请注意 Request.Form("select") 语句是区分大小写的。这条语句是记录集设置中的关键语句，它代表程序运行中，待定变量 bbh 是从下拉菜单 select 网页元素中取得的。因此记录集中只包括职工编号(EmpNo) 与 select 中选择的值相等的记录。

需要指出的是，这里的语句只适合于数字类型的字段，如果字段属于文本类型时还应该适当地加上引号，详细情况请参看后面的例子。

■ 单击 Test 测试按钮检查参数的设置情况。如果设置正常，将显示缺省条件的记录。

18.2.3 多条件组合查询

多条件组合查询是经常用到的一种查询方式，它的特点是，界面上列有多个查询条件，允许输入其中的部分条件，也可以输入全部条件。查询结果必须符合所有输入的条件。也就是说在这些输入条件之间存在着 “与” 关系。

为了进行多条件组合查询须完成以下几项工作。

(1) 在查询网页(search.asp) 中放入一表单(取名为 Form1)，表单内放一表格(用于定位)，表格内分别放入 3 个文本输入框及两个按钮。3 个输入框的名字分别是：textfield、textfield2、textfield3 分别用于输入姓名、性别与部门等查询条件，两个按钮分别是提交按钮及复位按钮。其界面如图 18.3 所示。

请输入查询条件	
姓名:	textfield
性别:	textfield2
部门:	textfield3
提交 复位	

图 18.3 组合查询界面

(2) 将 Form1 的属性 Method 设为 Post，将 action 属性指向 Result.asp。

(3) 打开 result.asp 网页，创建记录集如图 18.4 所示。

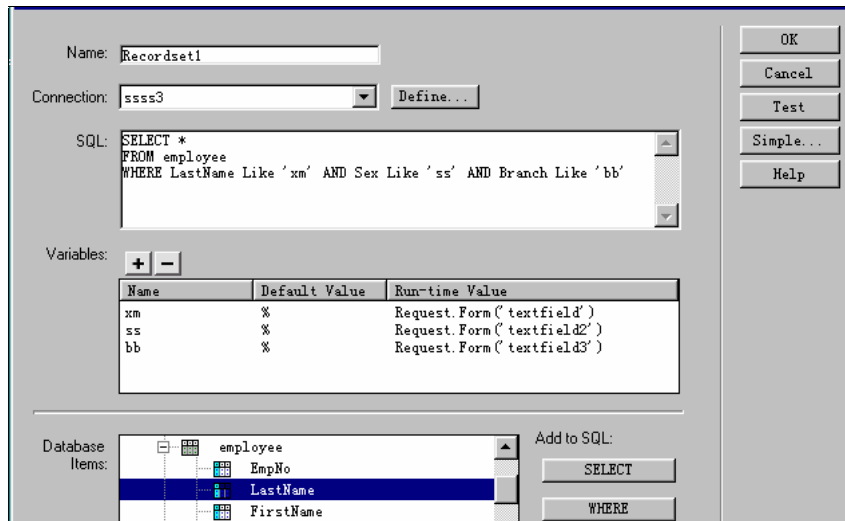


图 18.4 组合查询 SQL 语句

(4) 在 SQL 框体中输入的 SQL 语句是：

```
select *
```

```
From Employee
```

```
Where LastName Like 'xm' And Sex Like 'ss' AND Branch Like 'bb'
```

其中“ xm ”、“ ss ”、“ bb ”都是待定变量，由于这 3 个字段都是文本类型，因此它们的待定变量都加上单引号。另外对于字符类型可以用“ Like ”代替“ = ”号。使用 Like 的好处是可以使用通配符作为查询条件。

值得注意的是，在 ADO 中，查询条件的通配符与 SQL 中的通配符有点差别。在 ADO 中用“%”(百分号) 代表任意字符串，用“ _ ”(下划线) 代表任意的单个字符。

(5) 确定待定变量。

在 Variables 窗体中分别写出以下字段：

Name	Default value	Run-time Value
xm	%	Request.Form("textfield")
ss	%	Request.Form("textfield2")
bb	%	Request.Form("textfield3")

它代表 3 个待定变量分别来自 Search.asp 网页的 3 个文本输入框，它们的缺省值是任意字符串("%")。因此如果设置正确，按 Test 按钮时应显示全部记录。

(6) 重要的补充设置。

表面上看，到此为止似乎已完成了多项条件组合查询的网页设计，然而还有一项重要的工作需要补充。即在多条件查询中应该允许缺项。就是说，不一定要输入全部查询条件，应该允许有的条件为空白。如果按照前面的设置，当某个文本框为空白时，系统将把空白字符也当成查询条件，结果将什么也查不到。

如何解决上述问题呢？解决的方法是用手工办法修改源代码。打开源代码窗口将看到下面一段源代码。通过这段代码可以了解系统是怎样读取 textfield 中的数据。

```
<%@LANGUAGE="JAVASCRIPT"%><%
```

```
var Recordset1__bb = "%";
```

```
if(String(Request.Form('textfield')) != "undefined")
```

```
{ Recordset1__bb = String(Request.Form('textfield'));
```

这段源代码告诉我们，系统在读取 textfield 中的数据以前先作了个判断，只有 textfield 中的值不等于“ undefined ”时，才将 textfield 中的值读过来。

什么情况 textfield 的值等于“ undefined ”呢？

网页刚被打开时，textfield 的值是“ undefined ”，此时不允许读入 textfield 的值，可以避免发生一些意外的错误。可是当提交数据时，textfield 的空格已不是“ undefined ”了，因此它也会读过来作为查询条件。

现在将上述代码按照以下方法加以修改，就能解决由于空格带来的问题了。修改后的语句如下。

```
if(String(Request.Form('textfield')) != "undefined" && String(Request.Form('textfield'))!="")
```

```
{ Recordset1__bb = String(Request.Form('textfield'));
```

语句中带下杠的部分就是增加的语句，其实只是在 if 语句中增加一个判断的条件：使得只有当 textfield 的值既不是“undefined”又不为空(“”)时才读取其值，否则继续使用缺省值(“ % ”)。

这样，3 条判断语句都改变成为以下形式。

```
<%@LANGUAGE="JAVASCRIPT"%>
```

```
<%
```

```
var Recordset1__bb = "%";
```

```
if(String(Request.Form('textfield')) != "undefined" && String(Request.Form('textfield'))!="")
```

```
{ Recordset1__bb = String(Request.Form('textfield'));
```

```
var Recordset1__kk = "%";
```

```
if(String(Request.Form('textfield2')) != "undefined" && String(Request.Form('textfield2'))!="")
```

```
{ Recordset1__kk = String(Request.Form('textfield2'));
```

```
var Recordset1__xx = "%";
```

```
if(String(Request.Form('textfield3')) != "undefined" && String(Request.Form('textfield3'))!="")
```

```
{ Recordset1__xx = String(Request.Form('textfield3'));
```

```
%>
```

18.2.4 指定范围浏览

1. 利用 Where 和 Between 条件语句

利用 SQL 的 Where 加 Between 语句可以浏览某个范围的数据，条件语句的格式是：

Where 字段名 Between 参数 1 AND 参数 2

例如：

```
SELECT *
```

```
From employee
```

```
Where Salary Between low AND hig
```

其中 Salary 是“薪金”字段，low 与 hig 是两个待定变量，其值在程序运行中从 search.asp 的文本框中读入，设置方法与前同。在结果界面上将显示出薪金在 low 与 hig 范围（包括 low 与 hig 两个边界上的数值）内的记录。

如果访问 Access 数据库，而参数属于日期/时间类型时，应在待定参数的前后加上#号。例如：

```
Select *
```

```
From employee
```

```
Where HireDate Between #StartDate# AND #EndDate#
```

在记录集窗口中设置的情况如图 18.5 所示。

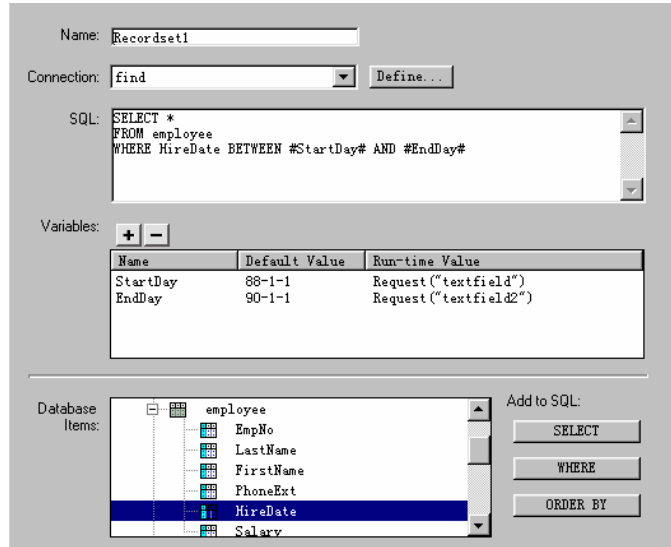


图 18.5 指定范围浏览

在结果页面上将显示聘用时间从 StartDay 到 EndDay 期间的职工记录。

2. 利用 Where 和 in 语句显示符合某个集合条件的记录

利用 SQL 条件语句 Where 和 in (...) 语句可以将属于某个集合条件的记录显示出来，例如：

```
Where M_code in ( "0010","0012","0013","0014" )
```

其中 M_code 代表某个字段名（例如代码字段）。此条件是将代码等于“0010”或“0012”或“0013”或“0014”的记录显示出来。

18.3 显示查询结果

为了在网页上显示查询结果，将记录集作为数据源，将网页中的表格单元或者文本输入框与记录集的有关字段动态绑定。

18.3.1 多条记录分页显示

为了将查询结果分页显示,先按单条记录显示的方法设置,然后再利用服务器行为设置成多条显示。

(1)先在 result.asp 网页上放置一个表格,设定成两行、多列。第一行用于写字段标题(如职工编号、职工姓名等),第二行用于显示记录内容。用鼠标将记录集中的相关字段从 Data Bindings 窗口拖到表格第二行的单元格中。也可以先选中单元格及记录集的相关字段,然后单击“Insert”按钮。数据绑定的情况如图 18.6 所示。

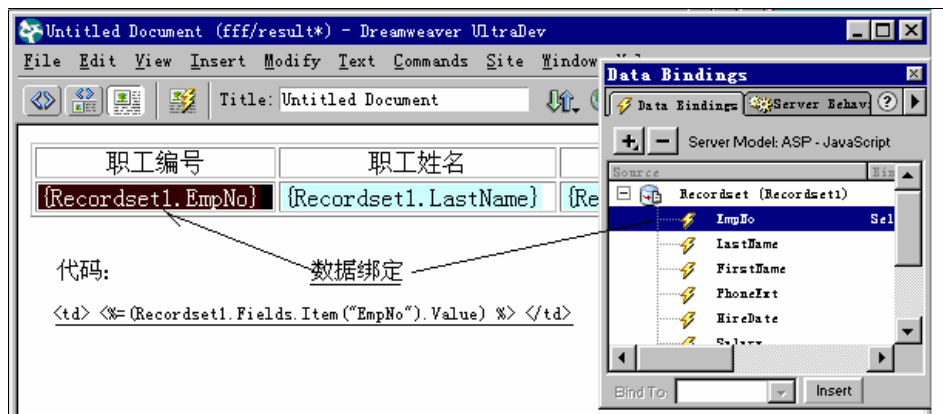


图 18.6 记录集字段与单元格数据绑定

此时系统将自动生成数据绑定代码。

(2)在表格中选择显示记录内容的行(不包括显示标题的行),或者先按住 Ctrl 键,再用鼠标分别单击各个单元格,各单元格的外框将变粗。

(3)打开服务器行为窗口,单击“+”按钮,在弹出的菜单中选择 Repeat Region 项,将弹出如图 18.7 的对话框。



图 18.7 确定每页记录数

(4)在窗口中先在下拉菜单中选择记录集名(Recordset),再确定每页包括的记录数。可以一次显示全部记录(All Records)也可以填入每页包括的记录数。

(5)选择好后单击“OK”按钮。此时表格的记录上方将出现 Repeat 图标,Server Behaviors 窗口中将出现 Repeat Region(Recordset1)项。

(6)选择 View | Live Data 菜单选项,可以立即显示出数据表每页中的多条记录情况。

如果想撤消多条记录的设置时，只需在 Server Behaviors 窗口中先选择 Repeat Region 项，再单击上方的“ - ”按钮。

经过前面的设置后，打开网页的源代码窗口时，将看到以下的代码。

```
<%
    while ((Repeat1__numRows-- != 0) && (!rs1.EOF))
    { %>
        // Repeat1__numRows 代表每页的记录数
    <tr>
    <td><%=rs1.Fields.Item("EmpNo").Value) %></td>
    <td><%=rs1.Fields.Item("LastName").Value) %></td>
    <td><%=rs1.Fields.Item("Sex").Value) %></td>
    ...
    <td><%=rs1.Fields.Item("Salary").Value) %></td>
    </tr>
    <%
    rs1.MoveNext();
        // 指针下移一条记录
    } %>
```

这是一段用分段方式写出的循环语句，其中 rs1 代表记录集名，“EmpNo”、“LastName”等代表字段名。每一次循环显示一条记录，然后指针下移一条记录。直到到达每页的记录数或表格结束时为止。循环条件中的 Repeat1__numRows 代表每页中显示的记录数，根据前面的设置确定。如果指定显示全部记录时，该变量被赋值为-1，由于减 1 后永远不会等于 0，因此对记录的条数限制不再起作用。

这段程序的运行过程是这样的。首先将 Repeat1__numRows 变量减 1，然后检查结果是否不等于 0；再看另一个条件，数据集是否没有结束，如果两个条件均成立，则继续循环。否则，只要有一个条件不成立时立即退出循环。

18.3.2 为多条记录增添颜色

为了使表格变得更加美观，可将每行记录设置成不同的底色。表格中记录的底色可通过 HTML 的属性来设置。如：

```
<tr bgcolor="silver">
```

代表将此条记录的底色设为银灰色。为了使表格中隔行显示银灰色时，可利用以下 JavaScript 代码。

```
<tr
    <%
        RecordCounter++;
        If (RecordCounter % 2 ==1 )
            Response.Write("bgcolor = silver");
    %>
>
```

这段代码的执行过程是：先将 RecordCounter 变量加 1，然后将此变量用 2 整除，如果余

数等于 1 时，表明这是单行，将底色设为银灰色。将这段程序插入循环体中。

```
<% while ((Repeat1__numRows-- != 0) && (!Recordset1.EOF)) { %>
<%
    var RecordCounter = 0
    %>
<tr
<%
    RecordCounter++;
    if (RecordCounter % 2 == 1)
        Response.Write("bgcolor = silver");
    %>
    >
    ...

```

将显示如图 18.8 所示的界面。在这个界面中，单行的记录底色都变成了银灰色，而双行的记录仍然保持原色不变。



图 18.8 给记录增加不同的底色

如果进一步想将单行底色设成银灰色，而将双行设成红色时，可以将下面这段程序插入到循环体中。

```
<tr
<%
    RecordCounter++;
    if (RecordCounter % 2 == 1)
        Response.Write("bgcolor=silver");
    else
        Response.Write("bgcolor=#FF0000");

```

```
%>
```

```
>
```

18.3.3 设置导航按钮

当数据表的记录有多页(或多条)时,要设置导航按钮,以便在分页(多条)之间转换。设置的步骤如下。

- (1) 先在网页的适当位置上写出“ 第一页 ”、“ 向前一页 ”、“ 向后一页 ”、“ 最后一页 ”等。
- (2) 选中其中之一,如选“ 第一页 ”。
- (3) 单击服务器行为窗口中的“ + ”按钮,在弹出的菜单中选择 Move To Record 项将弹出如图 18.9 所示的菜单。
- (4) 选择 Move To First Record 项,在弹出的小窗口中确认后单击“ OK ”按钮。
- (5) 用同样的方法设置“ 向前一页 ”...“ 最后一页 ”等项。

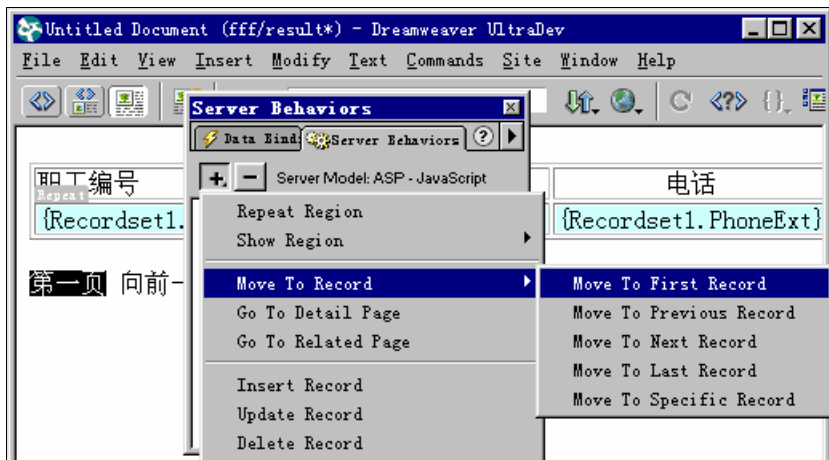


图 18.9 导航按钮设置

18.3.4 设置显示/隐藏区

利用服务器行为可以使某些区域在“显示”和“隐藏”状态之间自动切换。以导航按钮为例,当已经进入第一页时“ 第一页 ”和“ 向前一页 ”的导航标志都可以隐藏;进入最后一页时,“ 向后一页 ”和“ 最后一页 ”的导航标志也应该隐藏,下拉列表情况下这几个标志又应该恢复显示。

为了设置这一功能,按照以下步骤进行。

- (1) 选择“ 第一页 ”和“ 向前一页 ”的导航标志。
- (2) 打开服务器行为窗口,单击“ + ”按钮后选择 Show Region | Show Region If Not First Record 项,如图 18.10 所示。
- (3) 在弹出的对话框中选择记录集,然后单击“ OK ”按钮。

上述设置的含义是,只要不是第一条(页)时“ 第一页 ”和“ 向前一页 ”的标志都要显示。反过来说,进入第一条(页)记录时,该标志就隐藏起来。用类似的方法逐个设置下拉列表需要隐藏的标志。

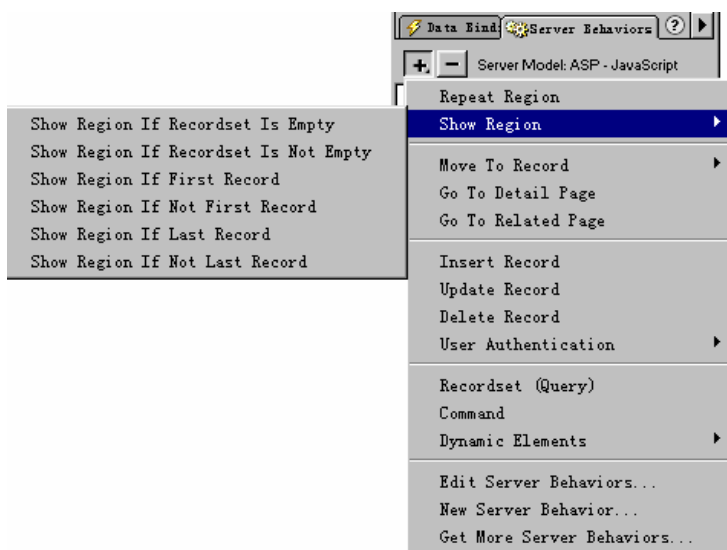


图 18.10 对显示/隐藏的设置

显示/隐藏区的行为不仅能应用于导航标志，还能应用于表格、图层甚至整个 HTML 的 <Body>...</Body>区。例如先选择整个表格，然后选择 Show Region | Show Region If Recordset Is Not Empty 项时，此表格只有当记录集不空时才会显示。

18.3.5 利用 Live 面板创建导航按钮

Dreamweaver UltraDev 4.0 新增加的 Live 面板提供的工具进一步简化了创建导航按钮的过程。具体步骤如下。

(1) 先确定导航按钮的位置，然后选择菜单 Insert | Live Objects | Recordset Navigation Bar 或者单击 Live 面板上的 Insert Recordset Navigation Bar 图标，将弹出如图 18.11 所示的界面。



图 18.11 字符/图像选择

(2) 选择记录集以及显示方式：图形方式还是字符方式。两种显示方式的区别如图 18.12 所示。

(3) 单击“OK”按钮后，系统已经设置好了以下代码。

- 一组.gif 图标或字符串。
- 一组移动记录指针的代码。
- 一组显示/隐藏的规则。

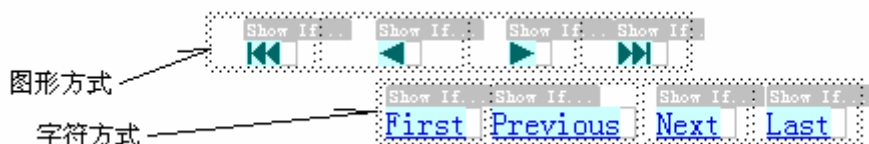


图 18.12 导航按钮的两种显示方式

(4) 根据设计者的需要可修改上述设定,方法是在服务器行为窗口中,双击有关项后在弹出的窗口或属性面板中改变相关的参数。

18.3.6 显示各页记录数的情况

在 Data Bindings 窗口的数据集的最后都有三项 {first record index}、{last record index}、{total records} 分别代表在本页中记录开始的条数,结束的条数以及整个记录集的记录条数。用鼠标将它们按照图 18.13 所示的方法拖动到网页中来,就可以显示出各页记录数的情况。



图 18.13 记录数的显示

18.3.7 利用 Live Objects 面板显示记录数

Dreamweaver UltraDev 4.0 版本新增加的 Live Objects 面板提供的工具进一步简化了设置显示记录数的工作。具体设置步骤如下。

(1) 确定显示位置后,选择 Insert | Live Objects | Recordset Navigation Status 菜单项,或者单击 Live Objects 面板上的 Insert Recordset Navigation Status 图标,将弹出如图 18.14 所示的窗口。

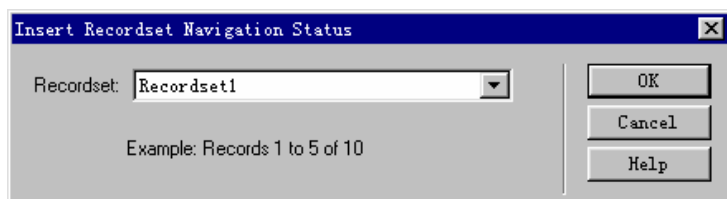


图 18.14 显示记录数的另一种设置方法

- (2) 在窗口中选择记录集后单击“ OK ” 按钮，设置完成。
- (3) 需要时可双击服务器行为窗口的有关项目并对显示方式作进一步的修改。

第 19 章 编辑数据库记录

对数据库记录的编辑工作包括对记录的增添、删除和修改。在 Dreamweaver UltraDev 4 中，可以使用多种方式来完成这些编辑工作。这些方式包括：

- 利用服务器行为中的选项；
- 利用 Live 面板中提供的对象；
- 直接利用记录集的相关命令。

下面将分别介绍这 3 种方式。

19.1 增添记录

19.1.1 利用服务器行为选项 Insert Record 增添记录

利用服务器端行为选项为数据库增添记录的工作包括两部分：输入新记录和增添记录并显示结果。这两部分内容可分别放置到两张网页上也可合并成一页。

1. 输入新记录

各种网页元素如文本框、单选按钮、复选框、列表/下拉列表等都可以用于向数据库输入新记录，但是它们都必须放在网页的表单中，用提交数据的方法向服务器提供新记录的数据。下面通过实例说明输入新记录的过程。

(1) 建立输入界面。

为了输入新记录需在网站中增添一新网页，并在其中放入一表单，表单内放入一表格，表格分两行，第一行是数据表中的字段名，第二行放入各种类型的网页元素。另外在表单中设“提交”和“重置”按钮。

假定“职工档案”表格中包括“编号”、“姓名”、“性别”、“部门”、“职务”、“联系电话”等字段。现在设置以下输入界面。其中性别、部门用下拉列表输入，其他均用文本框输入，如图 19.1 所示。

编号	姓名	性别	部门	职务	联系电话
0045	吴从德	男	市场部	业务员	5543425

未定

经理室

市场部

财务部

保卫部

图 19.1 输入记录界面

由于“性别”字段的选项固定，可采用固定式下拉列表。为此，先选择该元素后，单击它的属性面板中的“List Values”按钮，将弹出如图 19.2 所示的界面，直接在 Item Label 下面填写“男”、“女”选项即可。

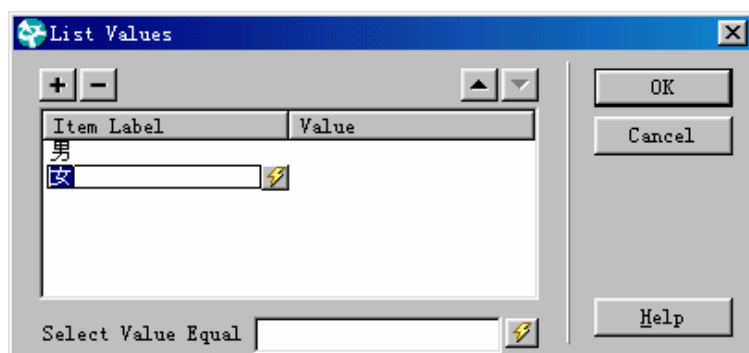


图 19.2 设置固定式下拉列表

而部门字段的选项只具有相对的稳定性。一个单位设立的部门在一段时期内可能是固定的，但从长期看也可能改变。为此，最好为部门单独创建一数据表（设取名为“部门表”），表中的部门（“BM”）字段用作菜单的选项。这样做的好处是，如果部门的设置有变化时，改变部门表中的记录就相应地改变了下拉列表的选项。因此这种菜单又称为“可变式”下拉列表。

为了创建可变式下拉列表，需在网页中创建记录集以指向部门表，然后单击图 19.2 中 Item Label 右边的闪电（）图标，打开如图 19.3 所示的窗口，在窗口中选择“部门（BM）”字段。使得该字段与下拉列表的选项动态捆绑。

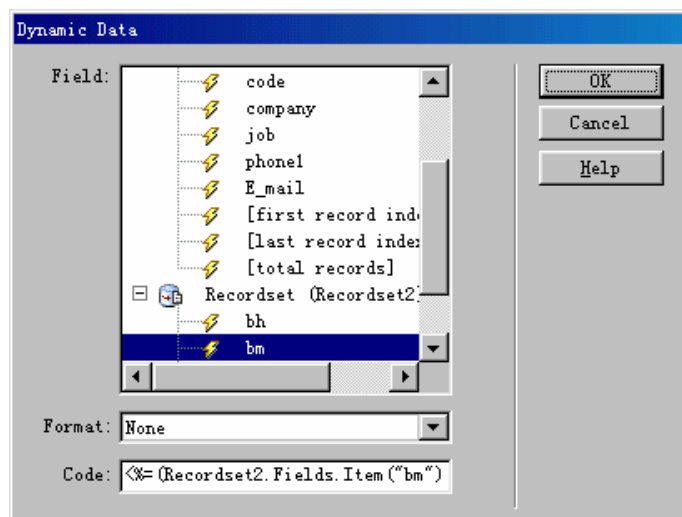


图 19.3 设置可变式下拉列表

（2）启用 Insert Record 服务器行为。

用鼠标先选取表单（Form），然后到服务器行为窗口中单击“+”按钮，在弹出的菜单中选择 Insert Record 选项，将弹出如图 19.4 所示的对话框。

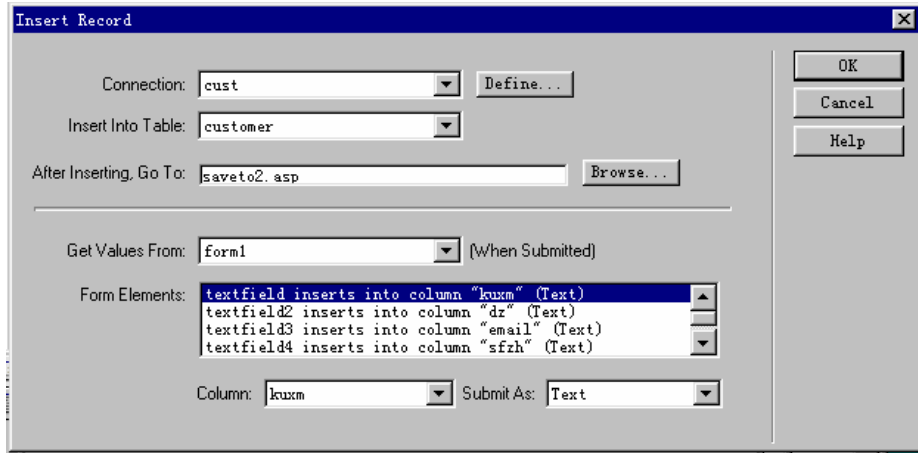


图 19.4 插入新记录

按以下要求填入有关数据。

Connection：在下拉列表中选择连接对象名。

Insert Into Table：在下拉列表中选择插入新记录的数据表名。

After Inserting, Go To：增添记录后跳转到的下一个页面，可利用 Browser 文件夹图标来选择。

[注意]：在这里设置以后不要再到 Form 的 Action 中去作新的设置。这一点与前面讲的方法不同，请加以注意。

Get Values From：在下拉列表中选择数据来自的表单名。

Form Elements：将自动列出表单上的网页元素名。

在 Form Elements 中逐个选择网页元素，并在 Column 的下拉列表中选择对应的数据表的字段，使得各网页元素与数据表字段之间联系起来。

(3) 设置输入时的校验功能。

为减少输入时的错误，应利用浏览器端行为设置表单的校验功能。方法是，选择菜单 Window|Behaviors|Validate Form 项，在弹出的 Validate Form 窗口中输入限制条件，如图 19.5 所示。

例如，记录中的姓名字段必须填写不能空缺时，选用该项 Value 的 Required 项；联系电话必须是数字时，选用该项的 Accept 中的 Number 项；电子邮件地址的输入字段，选用 Email Address 选项 此时规定输入必须包括@符号) 如果某项数字输入有范围限制时，选用 Number，并在 from...to...右边的小框体中分别填入该数字的下限及上限。

2. 显示结果部分

显示结果的界面可以放在输入新记录的同一张网页中，也可以单独为它设置一网页。

(1) 结果显示与输入记录在同一张网页。

在表单外另外放入一表格，建立一记录集对象指向增添记录的表格，并将记录集中的字段拖放到表格的单元格中，再利用服务器端行为“ Repeat Region ”将表格中的记录呈多行显示，并在表格下面设导航按钮。

(2) 结果显示放在单独的页面。

在输入数据的网页中，选择 Form 然后在 Server Behavior 窗口中，选择 Go to Related Page 行为项，在弹出的菜单中选择结果网页的 URL，此时在 Form 的下面将出现 Related 标记。

程序运行时，填入新记录后按动 Related 标记时，将转入显示网页。

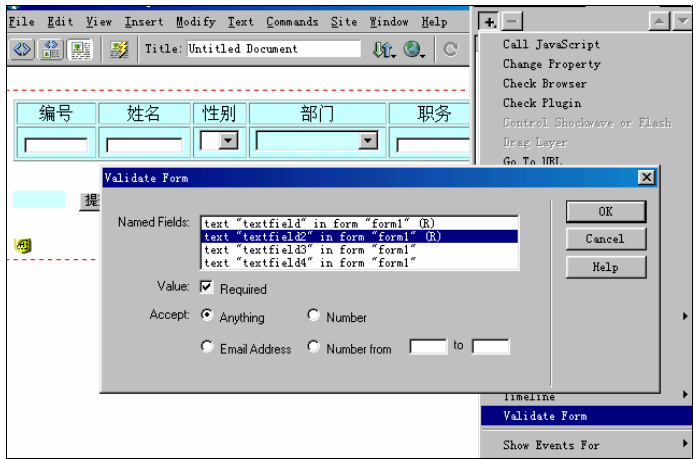


图 19.5 输入校验条件

19.1.2 利用 Live Objects 面板中的对象增添记录

Dreamweaver UltraDev 4.0 在 Object 面板中新增加了 Live 页，页中提供了 5 个图标，与此同时在 Insert 主菜单中增添了 Live Objects 项以及它的 5 个子项，它们的作用相同，都包括插入主/从页、设置浏览按钮、设置浏览状态、插入记录、修改记录等工作，如如图 19.6 所示。



图 19.6 Live 面板

利用 Live Objects 对象增添记录与前面讲的方法基本相同，它们之间的最大区别是，在已经建立连接对象的基础上能自动生成网页的输入界面，从而进一步简化了设计过程。其具体步骤如下。

- (1) 打开一新网页。
- (2) 用鼠标单击 Live 页中的“ 插入记录 ”图标，将弹出如图 19.7 所示。在窗口上方填入连接对象名及表格名后，在 Form Fields 列表框中将显示该表格中各字段的情况。



图 19.7 利用 Live Objects 面板插入新记录

(3) 在对话框中可以进一步编辑各显示字段。

- 利用“ - ”按钮删除某些不需显示的字段，用“ + ”按钮恢复已被删除的字段。注意凡是自动生成的字段（包括自动递增的字段）不应作为输入的内容。
- 在 Label 文本框中编辑显示的字段标题，可改用汉字显示。
- 在 Display As 窗口的下拉列表中选择输入的网页元素（如选择使用文本输入框、下拉列表、复选框、单选按钮等）。
- 在 Submit As 中确定各网页元素提交时的数据类型。
- 在 Default Value 中输入缺省值。

(4) 此时的输入界面已经如图 19.7 所示在网页中自动生成, 单击“OK”按钮。

若界面中包括下拉列表 (Menu) 时, 还应按照前面讲述的方法设置固定或可变式菜单项。

以上两种增添记录的方法，系统内部都是使用命令对象(Command)而没有直接使用记录集对象。因此在设置方法上与使用记录集时的方法稍有不同。

19.1.3 利用代码添加记录

下面介绍另一种添加记录的方法，它是将记录集(Recordset)的 AddNew()与 Update()方法相结合来实现的，使用这种方法时需要设计人员自己编写代码，虽然麻烦一些，但也增加了灵活性。这段代码应该放在<html>标记之前，记录集创建以后。代码的格式如下。

```
<%  
rs1.AddNew();  
rs1.Fields.Item("字段名").Value = Request.Form("表单元素名");  
...  
rs1.Update();  
%>
```

其中 rs1 代表记录集名，rs1.Update()是将新记录保存到数据库中的语句，这一语句如果省

略，则只有当重新调用 AddNew() 或者执行移动指针命令（如 MoveFirst 等）时，上一条记录才会存入数据库中。

有的情况，需要在增添新记录前作最后的确认，此时语句的后面可以增加判断语句。如：

```
rs1.AddNew();
rs1.Fields.item("字段名").value = 值;    // 为记录的各字段赋值
...
if (条件)                                // 增加判断语句
    rs1.Update ();
else
    rs1.CancelUpdate ();
```

其中 rs1.Update ()是执行增添记录命令，而 rs1.CancelUpdate ()是放弃增添记录的命令。

19.2 修改记录

19.2.1 利用服务器行为 Update Record 选项修改记录

记录修改通常是在查询的基础上进行的，即先查找记录，再对找到的记录进行修改。因此应建立两张网页，一张用于输入查询条件，另一张进行查询并实施对记录的修改。设计步骤如下。

（1）打开一查询网页，放置一表单（Form），表单上放置输入元素和提交（查询）按钮，本例中利用 List/menu 设置的可变式下拉列表与 BH（编号）字段相连，以输入查询条件。



图 19.8 查询界面

（2）打开结果网页，放入表单，表单内放入输入框、按钮等，通过记录集与数据表连接。在记录集中设置 SQL 的查询语句，如图 19.9 所示。

（3）打开服务器端行为（Server Behaviors）窗口，单击窗口中的“+”按钮，在弹出的菜单中选择 Update Record 项，将弹出如图 19.10 所示的对话框。

对话框的填写方法与上一节基本相同，主要的不同点是：要确定关键字段（Unique Key Column），因为关键字是查询的根据，本身是不能被修改的。再逐个选择 Form Elements 中的项目，并到 Column 的下拉列表中找到对应的字段。数据表中凡自动进行编码的字段都不能作

为修改的字段。

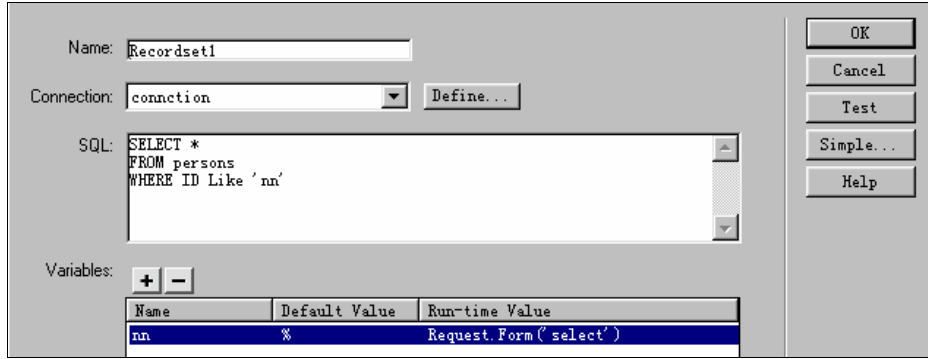


图 19.9 修改前的查询

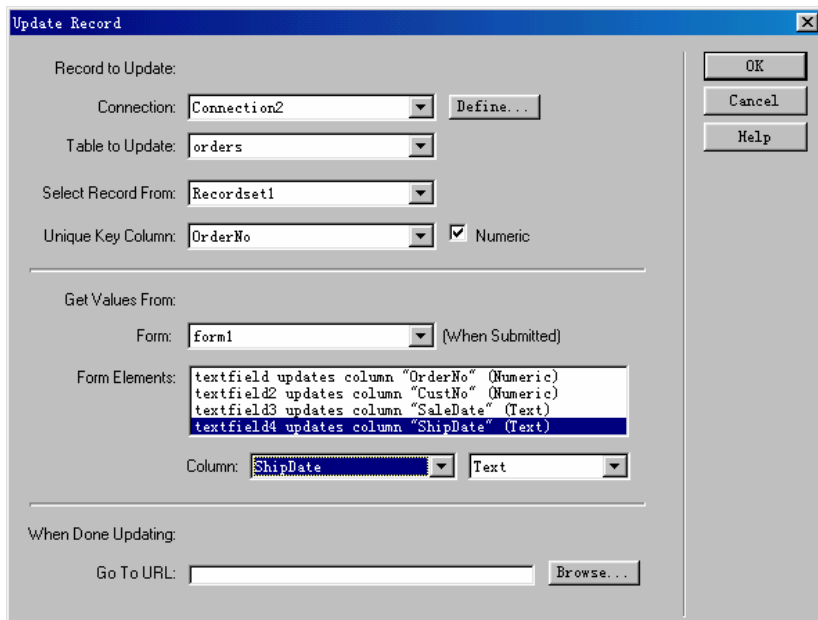


图 19.10 修改记录的设置

(4) 设导航按钮以便查看修改记录后的实际效果。

19.2.2 利用 Live 面板中的图标修改记录

利用 Live 面板中的图标修改记录与上面讲的修改记录的方法相似但更加简单。创建查询条件网页的方法与前面相同，但在创建结果网页时，先要创建记录集对象，然后从 Live 页面中点入记录修改图标，将弹出类似于图 19.10 的对话框，填写完相应的数据后，你将看见系统已经创建好了一个修改记录的界面，必要时还可以对这一界面作些补充和修改。

19.2.3 编写代码以修改记录

利用记录集(Recordset)的 Update 命令可以修改记录。为此须在<html>标记之前，记录

集创建以后编写相应的 ASP 代码，格式如下。

```
<%
rs1.Fields.Item ("字段名").Value = Request.Form ("表单元素名");
...
// 其他字段的赋值语句
rs1.Update ();
%>
```

读者可能会发现，这段代码与增添记录的代码非常相似，只是在前面少了一句 "rs1.AddNew();" 其他完全相同。

与增添记录一样，可以在执行修改前增加判断语句，以便作最后的确认。语句中同样可以利用 CancelUpdate () 语句来放弃对记录的修改。

19.3 删除记录

19.3.1 利用服务器行为 Delete Record 选项删除记录

删除记录往往是在查询基础上进行的，这一点和修改记录相同，查询的过程这里就不重复了。在删除的网页上需创建记录集和表单 (Form)，用来指定要删除的记录，可以创建另一个表单来显示删除后的数据表。具体步骤如下。

(1) 新增加一网页，放入一表单 (Form)，表单内放入一文本输入框 (Textfield) 及提交按钮。

(2) 创建一记录集 (Recordset)。

(3) 在服务器行为 (Server behavior) 中选择 Delete Record 项，将弹出如图 19.11 所示的对话框。

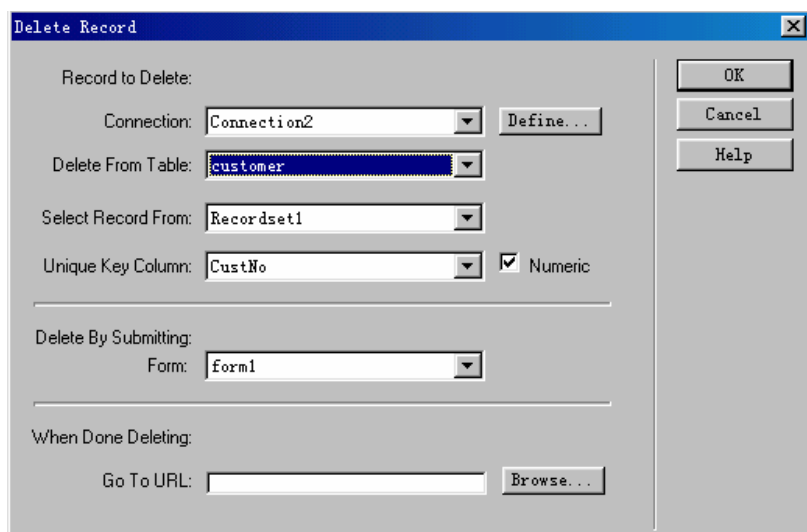


图 19.11 删除记录的设置

(4) 在窗口中需要填入的有关数据如下。

Connection : 连接对象。

Delete From Table : 被删除记录所在的数据表。如果已设置主/从关系的表格, 在从表的有关记录被删除之前, 不允许直接删除主表的有关记录。

Select Record From : 记录集对象。

Unique Key Column : 关键字栏。删除工作必须根据关键字进行, 否则可能出现混乱。

Delete By Submitting : 从哪里发出删除信息, 下面的 Form 窗口即填写发出信息的地方。

When Done Deleting : 当记录被删除时, 下面的 Go To URL 指出转移到那个页面。如果仍然转向本页时, 这个小窗口可以不填写。

(5) 如果就在今页上查看结果时, 还需另外创建一个表单, 表单上放一表格, 利用前面设置的数据集 Recordset1 将有关字段拖到表格中的相应位置中来。

(6) 通过服务器行为设置显示多条记录及浏览按钮。

文件存盘后到浏览器运行以查看实际效果。

19.3.2 编写代码删除记录

为了使用记录集 (Recordset) 删除记录, 须在<HTML>标记之前、记录集创建以后编写如下代码。

```
<%
Recordset1.Delete();
%>
```

此命令将删除记录指针所指的当前记录。

19.4 成批处理记录

前面讲的对数据库记录的编辑方法, 都是针对单条记录进行的。实际上需要成批处理记录的情况也是经常遇见的。比如一次需增添多条记录, 删除多条指定的记录等。虽然对多条记录的处理也可以一条一条地进行, 但这样做加大了网上负载, 降低了运行效率, 并不是理想的选择。

19.4.1 利用 SQL 语句成批处理记录

利用 SQL 语句可以成批地处理记录。如 :

DELETE FROM 数据表名

WHERE xb = '女'

一次将数据表中性别 (xb) 为 '女' 的记录全部删除掉。

再如 :

DELETE FROM 数据表名

WHERE s_bh in (数据集合)

将同时删除 s_bh 符合 " 数据集合 " 内的所有记录。例如数据集合为 : (" 0033 " ; " 0034 " ,

“ 0035 ”,“ 0041 ”,“ 0098 ”,“ 1000 ”,“ 2243 ”) 时,将同时删除 s_bh 等于这 7 条记录之一的记录。

19.4.2 利用记录集成批处理记录

通过记录集成批处理记录时,还可以增加对“ 事务 ”的控制。

1. 事务(Transaction)

(1) “ 事务 ” 是数据库技术中的一个重要概念,它代表一组彼此相关、连续执行的语句。这些语句被看成是一个整体(一个原子),要么全部执行(不允许只执行其中一部分),要么全部不执行(恢复执行前的状态)。“ 事务 ” 是用于保证数据完整性的重要手段。

成批处理数据的语句,被看成是“ 事务 ”。执行的结果只有两种可能,要么完全成功,要么全部失败。例如,执行批量增添记录的过程中发生了某种“ 异常 ”,此时只能将前面增加的记录全部作废,恢复增加前的状态。

(2) 通过连接对象(connection)处理事务。

连接对象中用于处理事务的内部函数有:

- BeginTrans(): 开始一个事务;
- CommitTrans(): 承认并存储事务处理的结果;
- RollbackTrans(): 取消当前的事务操作并返回初始状态。

2. 成批增添记录时的代码

现将成批增添记录时的代码举例说明如下。

```
cnn.BeginTrans (); // 开始事务
// 创建记录集 rs1
// 请注意记录集的创建必须放在开始事务之后。否则事务对记录集不起作用
for (i=0; i < 20; i++)
{
    rs1.AddNew ();
    rs1.Fields.Item ("字段名").value = 值 ; // 为新记录字段赋值
    ...
    rs1.Update (); // 在事务中更新,但在事务结束前这个更新只是暂时的
}
if (条件) { // 如果条件中出现了异常
{
    cnn.RollbackTrans (); // 滚回,恢复初始状态
}
else
{
    cnn.CommitTrans (); // 结束事务,保存结果
}
}
```

这段程序是通过循环语句连续增添 20 条记录。Cnn 代表连接对象,rs1 代表记录集。当遇

见某种“异常”时将取消全部增加的记录，恢复初始状态，否则完成记录增添工作并保存增加记录后的结果。

19.5 使用服务器行为与直接编写代码编辑记录的比较

前面介绍了 3 种编辑记录的方法，实际上这些方法可以分成两大类。利用服务器行为或利用 Live Data 图标时，系统内部都是采用命令对象（Command）来编辑记录；而用编写代码的方法都是通过记录集来编辑记录。两者之间有什么区别呢？

让我们先来看看命令集对象是如何编辑数据库记录的。

下面是一段简化了的命令集给数据表增添记录的源代码。

```
<%
MM_editQuery = "insert into " + 数据表名 + "(" + 字段名 + ") values (" + 字段的新值 + ")";
// 上面是将一个 SQL 语句，赋值给变量 MM_editQuery
// 这是一条增添记录的 SQL 语句，其中的数据表名、字段名、字段新值等
// 都是从输入新记录的网页中获取的
if (条件) {
    // 执行 insert 语句
    var MM_editCmd = Server.CreateObject('ADODB.Command');
    // 创建一个命令对象的实例 MM_editCmd
    MM_editCmd.ActiveConnection = MM_editConnection;
    MM_editCmd.CommandText = MM_editQuery;
    // 将前面的 SQL 语句赋值给命令对象的 CommandText
    MM_editCmd.Execute();
    // 命令集执行 SQL 语句
    MM_editCmd.ActiveConnection.Close();
    ...
    // 跳转到新网页
}
%>
```

为了修改和删除记录，命令集对象的源代码中只要改变一句，即将修改或删除的 SQL 语句赋值给 MM_editQuery 变量，后面的代码都不需改变。

从这些代码可以看出，不论是命令对象还是记录集对象，处理记录的核心都是 SQL 语句。因此两者之间并没有本质上的区别。然而两种形式毕竟在功能和效率方面还是有些不同。

概括地讲，使用系统提供的服务器行为（或 Live Data 图标）比较方便，不需要编写成段代码，而且运行效率高；而直接编写代码通过记录集处理记录时，虽然运行效率稍低，难度也稍大一些，但处理的灵活性也高一些，还可以在执行前用判断语句作最后的确认。特别是能够用处理“事务”的方法控制批量处理记录的过程，因而可以提高数据的完整性和可靠性。

第 20 章 主/从数据表的同步

20.1 基本概念

有两种情况需要采用主/从表的结构形式：一种是数据表内容之间存在着“一对多”的关系，另一种是数据表的栏目(字段)过多。

(1) 数据表的内容之间存在着“一对多”的关系。例如：某公司有多个客户，每个客户有多份订单就属于这种情况。如果把这种关系综合在一张表中，则每条记录都必须包括客户的有关数据(如姓名、单位、地址、E-mail 地址、邮编、职务以及身份证号等等)，以及他们的订单情况。如果每个客户平均有 10 项订单，则客户的情况也得重复 10 遍。这样的表格显然存在着大量的数据冗余。

如果把这种“一对多”的关系用两张表来处理。将其中一张表作为“主表”，记录共同的数据；另一张作为“从表”，主要记录不同的数据。以客户订单为例，用主表记载客户的情况，每个客户的情况用一条记录描述，而“从表”用来记载订单的情况(另外还包括同步字段)，两张表同步运行，这样就改善了数据表的结构，不仅减少了数据冗余，还能清晰地显示出它们之间的层次关系。

(2) 表格的栏目(字段)过多，超过了网页的宽度时，不便一道在浏览器中显示，此时可以将它分解成多张表，以主/从表的形式放在不同的网页中，并且让它们同步运行。

20.2 利用服务器行为实现主/从表同步

下面通过具体示例来说明实现主/从表的同步过程。设 Customer(客户)表为主表，orders(订单)表为从表，两表之间通过 CustNo(客户号)字段同步。

建立主/从表同步的具体步骤如下。

(1) 在网站中增添两张网页，一张网页中放置主表(设网页名为 Master.asp)，另一张网页中放置从表(网页名为 Detail.asp)。

(2) 打开主表所在的网页，为网页创建连接对象(Connection1)与记录集对象(Recordset1)，使网页与主表 Customer 连接。

在网页中放入一表单(Form)，表单中放置表格(Table)，并在表格中放入若干单行文本框(Textfield)。从 Data Bindings 窗口中将 Recordset1 下面的字段拖到 Textfield 文本框中，以实现动态数据绑定，网页的界面如图 20.1 所示。

创建浏览按钮。方法是选择服务器行为窗口 Server Behavior 中的“+”按钮|Move To Record|选项创建“第一条记录(First)”“向前一条记录(Previous)”“向后一条记录(Next)”“最后一条记录>Last)”等浏览按钮。

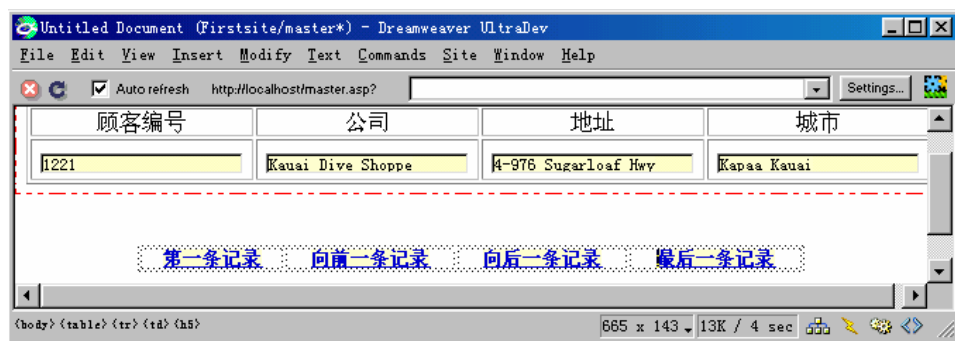


图 20.1 主表网页的界面

检查连接效果。方法是选用菜单 View|Live Data 激活 Live Data Window 窗口，以检查与主表的连接情况，正常情况下第一条记录的内容将显示在表格中。

(3) 在从表网页中放一表单 (Form)，表单中显示从表的多条记录。与从表 orders 连接时使用与主表相同的连接对象 (Connection1)，并创建自己的记录集。此时的记录集名虽然可能和主表的相同，但它们是不同的记录集，分别捆绑于不同的网页上。

在设置记录集 (Recordset) 时选择从表 (orders) 并在 Filter 中建立与主表的同步关系。即在 Filter 的下拉列表框中选择与主表的同步字段 CustNo，如图 20.2 所示。

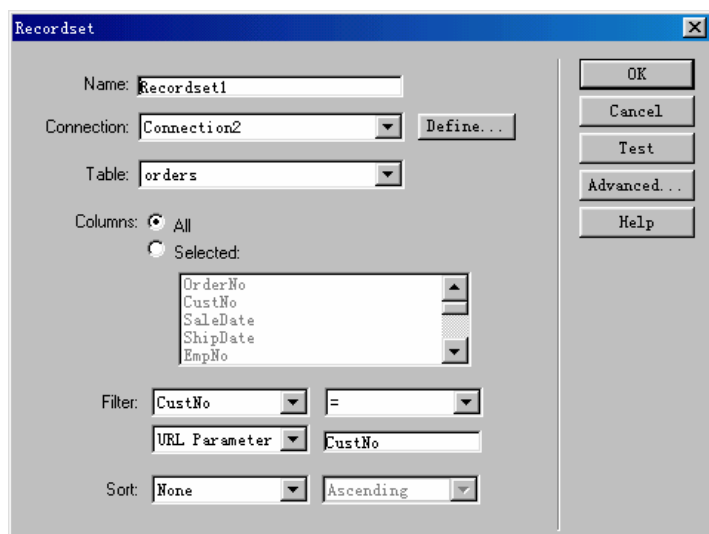


图 20.2 主从表同步设置

也可以单击 “Advanced” 按钮进入 Advanced 页面中直接输入 SQL 语句以及同步条件，如图 20.3 所示。

其中 SQL 的语句是：

```
select *
from orders
where CustNo = MMColParam
```

语句中 MMColParam 是一个待定变量，运行时再赋值。请注意它的表示方法：当待定变量属于数值型时，不用外加引号，缺省值(Default Value)直接采用数值；当待定变量属于字符类型时，要外加单引号，即变量名'，可以用"Like"取代前面的"="。例如：

Where Cust Like 'bh'

当用"Like"取代"="时，缺省值可以使用通配符"%"；当待定变量属于日期类型时，变量名的前后都要加上"#"号，即 #日期型变量名#。这些问题前面说过，这里再强调一次。而在下面 Variables 窗口的 Run-time Value 中使用双引号。如：

Request.QueryString("CustNo"); // CustNo 为主表网页同步字段的网页元素名

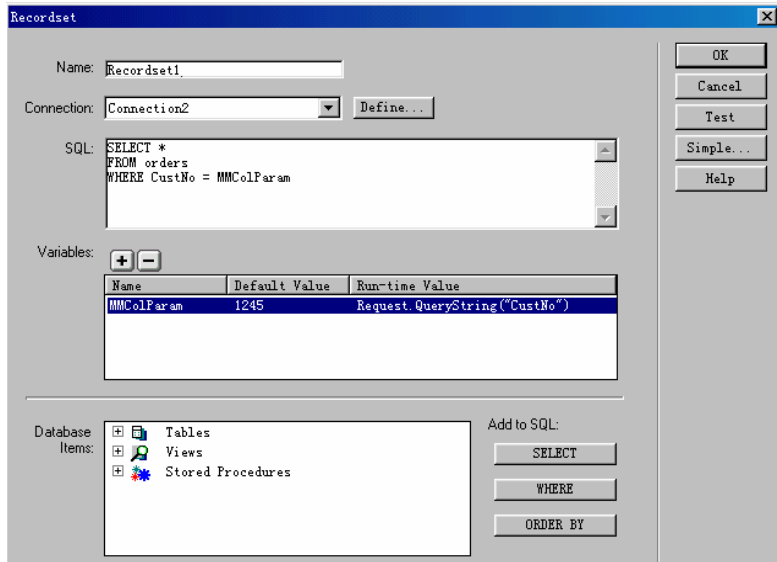


图 20.3 利用 SQL 语句同步

(4) 保存从表网页。

(5) 打开主页 master.asp 并在适当位置写上“ 同步 ”标签，再选择 Server Behavior| “ + ” |GoTo Detail Page 项，将弹出如图 20.4 所示的对话框。

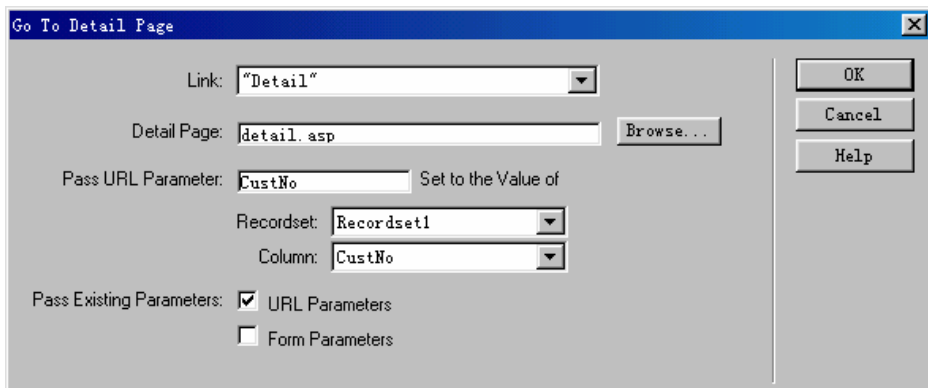


图 20.4 转向从表的服务器行为

在窗口中填写以下参数。

- Link：在下拉菜单中选择 Selection : "同步" 项。
- Detail Page：利用 Browser 文件夹图标选取 Detail.asp 子表网页。
- Recordset：从下拉列表中选取传送给从表网页的 Recordset。
- Column：从下拉列表中选取传送给从表网页的字段，让从表接受后与主表取得同步。
- 选取 Pass Existing Parameters 中的 URL Parameters 项。选择这项的作用是使得传送的参数将放在 URL 后面（类似于 GET 方法的传送）。

设置完成后 Master.asp 页面上将出现 Detail 同步标签，它说明主/从表的同步关系已经建立。

（6）将主表网页存盘，然后到浏览器中检查同步效果。当在主表网页中单击 Detail 同步标签时，将转向从表网页并显示从表中的同步记录。

20.3 利用 Live Data 窗口检查同步效果

Live Data 是 Dreamweaver UltraDev 中用来检查动态数据的窗口。它允许在设计阶段直接显示动态内容，还能对显示的动态内容进行某些编辑，系统提供这个窗口的目的是为了对动态内容的检查。不然的话，你只能到浏览器中才能看到动态内容，为了编辑动态内容，可能不得不在浏览器和设计界面之间反复切换，这将是一件比较麻烦的事情。当然并不是所有的动态内容都可以在 Live Data 中看到，比如网页之间的链接情况，就看不到，还只能到浏览器中去浏览。

利用 Live Data 窗口检查同步效果时，先打开从表网页并通过菜单 View | Live Data 打开 Live Data 窗口，此时原来已经打开的文档会临时拷贝到一个特定的服务器中去处理，当处理完毕再次显示网页时，这个临时拷贝文件又将自动被取消。

Live Data 窗口的界面如图 20.5 所示。



图 20.5 利用 Live Data 窗口检查同步

单击“ Settings ”按钮将弹出如图 20.6 所示的对话框。

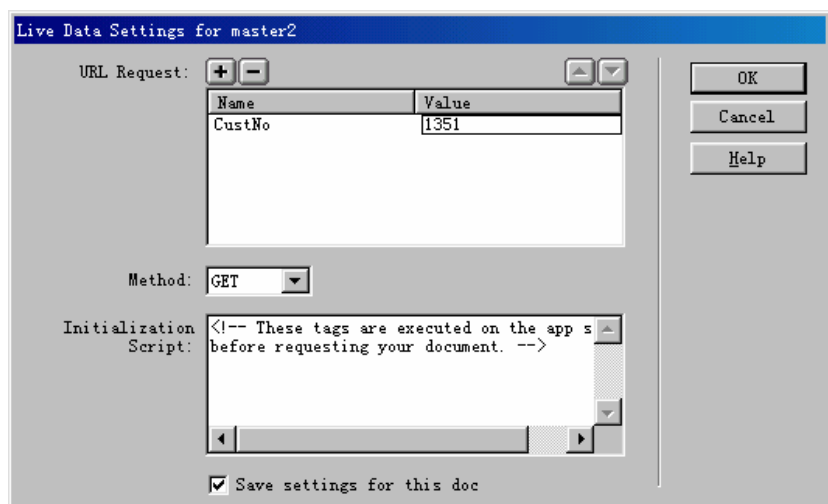


图 20.6 为 Live Data 窗口设置参数

单击“+”按钮，在 Name 下面输入传入的同步参数（本例为 CustNo），在 Value 下面输入设定的值，Method 必须设置为 GET，单击“OK”按钮返回从表界面，然后单击“Refresh”按钮（或选择 Auto Refresh 项），即可观察同步的效果。

20.4 利用 Live Objects 面板实现主/从表的同步

下面介绍利用 Live Objects 面板上的 Insert Master-Detail Page Set 图标实现主/从表同步的设计方法。这种方法适合于将过多字段的表格分解成主/从表的情况。使用这种方法时非常迅速、简单，可以做到“一步到位”。整个设计过程只需要在一两个对话框中填写数据，就可以完成全部设计工作。

设计的步骤如下。

（1）在网站上增加两个空的网页，一个为主页（设取名为 Master.asp），另一个为从页（取名为 Detail.asp）。

（2）打开主页并建立记录集，该记录集中既包括主表中显示的字段又包括从表中显示的字段。

（3）选择菜单 Insert|Live Objects|Master Detail Page Set 项，或者直接单击 Live Objects 面板中的 Insert master-Detail Page Set 图标，将弹出如图 20.7 所示的主/从页的对话框。

（4）在窗口中设置有关数据。窗口的上半部为主页的有关数据，下半部为从页的有关数据。

主页的有关数据

■ Recordset：选择记录集。

■ Master Page Fields：主表中将放入的字段。打开上述对话框时该框体内列出了记录集中的全部字段。利用“-”按钮减少字段，只保留主表中显示的部分。还可以利用▲▼按钮改变字段显示的顺序。由于主表在网页中是横向显示的，因此这里向上移动时对应于字段左移，

向下移动对应于字段右移。

- Link To Detail Form：主表中与从表的链接字段，从下拉列表中选择。
- Pass Unique Key：标签记录的字段，在下拉列表中选择（通常选用记录编号）。
- Show：每页显示的记录数。

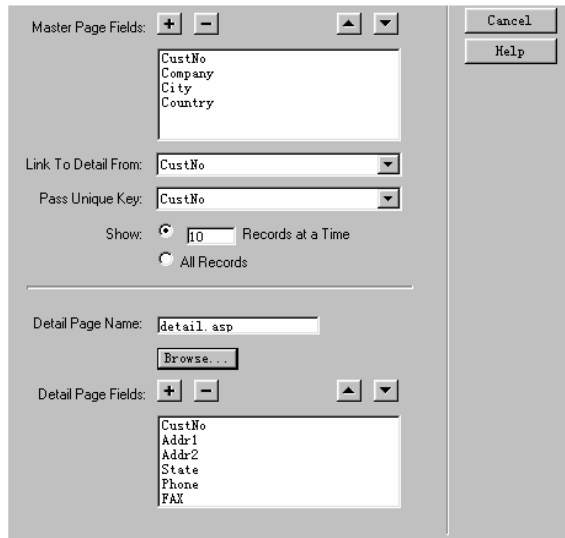


图 20.7 利用 Live Objects 实现同步

从页的有关数据

- Detail Page Name：从表的网页名。可以利用 Browser 按钮选择另一张空的网页。
- Detail Page Fields：从表中包括的字段。打开网页时，这个框体中包括记录集的全部字段。利用“-”号按钮减去不在从表中显示的字段，利用▲▼按钮调整字段的显示顺序。

（5）设置完成后单击“OK”按钮，此时系统已经为两表同步设置好了初步条件。两张网页的情况如图 20.8 所示。

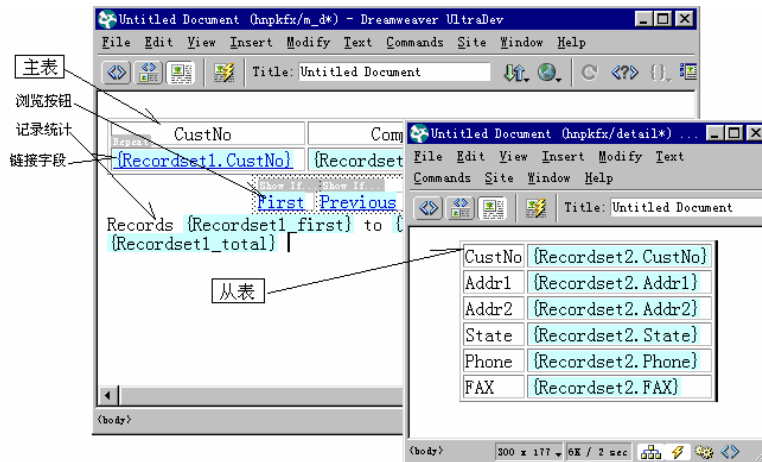


图 20.8 同步网页的界面

从图中可以看出，系统已在主网页中设置好了以下部分。

- 显示记录的表格，以及动态数据绑定。
- 多条记录显示的设置。
- 记录的导航按钮。
- 记录的计数显示。
- 链接从表的指针。

在从表网页上已经设置了以下几个方面。

- 显示从表的表格以及动态数据绑定；
- 按照对话框的设置显示的相应字段。

(6) 根据需要可以对前面的设置进行修改和补充。

第 21 章 存储过程

21.1 概述

21.1.1 什么是存储过程

虽然标准的 SQL 是一种非过程语言,但大部分数据库都增加了流程控制语句,如 MS SQL Server 7.0 将增强后的语言称为 T-SQL 语言。Oracle 数据库将增强后的语言称 PL/SQL 语言。利用增强后的语言将一条或多条 SQL 语句组织成为一个过程,存储起来以便以后的某个时候调用,这个过程就是存储过程。

21.1.2 为什么需要存储过程

存储过程可以用来对数据表进行查询、插入、删除等操作,甚至可以用来产生新的数据表。但它不像通常的 SQL 语句那样和网页的源代码放在一起,而是存放在数据库的系统表中。当第一次被编译时,除检查正确性以外还会优化访问表的信息路径。每次执行时只要调用存储过程的名即可在数据库端直接运行,从而带来以下好处。

(1) 将经常进行的操作定义成存储过程,可以简化操作,提高可靠性。

(2) 存储过程放在数据库端,运行时只需调用过程名,由数据库执行。这种方式减少了通信流量,提高了运行效率。

值得强调的是,存储过程是数据库提供了一种功能,一些大型数据库,如 SQL Server、Oracle 等都提供了这一功能。而一些小型数据库如 Access、Dbase、Paradox 等都没有提供这一功能。因此存储过程不能在上述这些小型数据库中运行。本章也只对存储过程作一些简单的介绍。

21.2 如何定义存储过程

不同类型数据库提供的存储过程语句并不完全相同。下面只介绍 SQL Server 数据库的存储过程语句。

存储过程的语句实际上包括两个组成部分:前一部分是定义过程名和有关参数,从 As 以后为第二部分,主要是增强后的 SQL 语句,这是整个定义的主体部分。

例如:

```
CREATE PROCEDURE all_employees  
As  
Select * From Employees
```

其中 CREATE PROCEDURE...As 等是定义存储过程的保留字。这里定义的存储过程名是

all_employees 相应的 SQL 语句是 `Select * From Employees`。

21.3 在 SQL Server 数据库中设置存储过程

为了在 SQL Server 数据库中设置存储过程，需按照以下步骤进行。

- (1) 进入 SQL Sever 的 Enterprise Manager 工作界面，展开服务器组，再展开服务器。
- (2) 展开数据库组再展开建立存储过程的数据库。
- (3) 用鼠标右击 Stored Procedures 项，在弹出的菜单中选择 New Stored Procedure 项，打开如图 21.1 所示的编辑窗口。

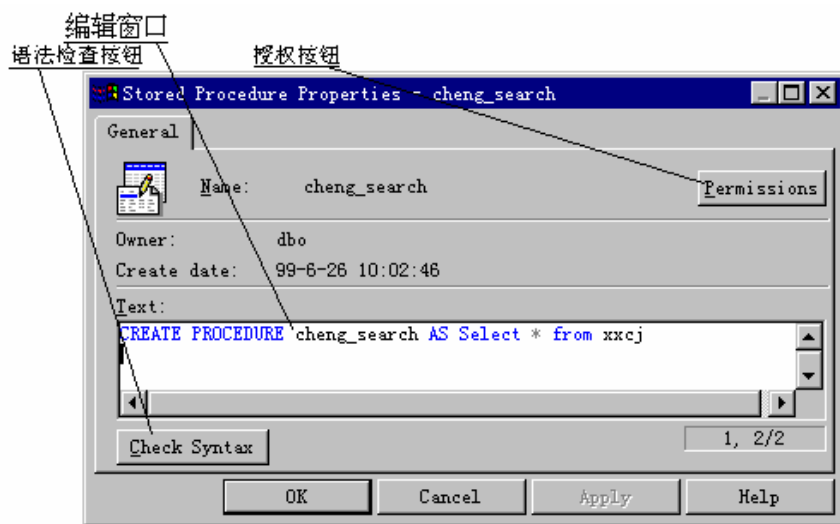


图 21.1 存储过程编辑窗口

- (4) 在编辑窗口中编写 T-SQL 语句，写完后可单击“ Check Syntax ”按钮检查语法的正确性。
- (5) 语法通过后，单击“ OK ”按钮返回原来界面。
- (6) 左击 New Stored Procedure 项，再右击设置的存储过程，选择 Properties 项，再一次进入第 4 步的编辑窗口，单击“ Permissions ”按钮，给用户授予使用权限。

21.4 存储过程的参数

调用服务器的存储过程，有时需传递一些参数。这些参数分为 4 种类型：

- 输入参数：由客户向存储过程传递的值。
- 输出参数：由存储过程向客户返回的结果。
- 输入/输出参数：既可向存储过程传递，又可返回结果。
- 状态参数：由存储过程向客户返回的错误信息。

并不是所有服务器都支持以上几种类型。

例如，在 SQL Server 中建立了一张学生成绩表（取名为 xxcj），并且建立了一个存储过程取名为 E_level，使用以下语句。

```
CREATE PROCEDURE E_level @aaa int=60
As
Select * From xxcj
Where Eng < @aaa
```

21.5 在 UltraDev 中为网页定义存储过程

先要在数据库中创建存储过程，然后在网页中将它作为数据源使用。在 ASP 服务器中将存储过程对象称为 Command。

可以利用存储过程来定义记录集。给存储过程定义记录集时，按照以下步骤进行：

- （1）在 UltraDev 中打开一动态网页。
- （2）在数据绑定面板中单击“+”按钮，在下拉列表中选择 Command (Stored Procedure) 选项，将弹出一对话框，如图 21.2 所示。
- （3）在对话框中为存储过程输入名称，在 Connection 的下拉列表中选择连接包括有存储过程的数据库。
- （4）在 Type 的下拉列表中选择 Stored Procedure。
- （5）选择 Return Recordset 项，并在 Name 文本框中输入记录集的名字。

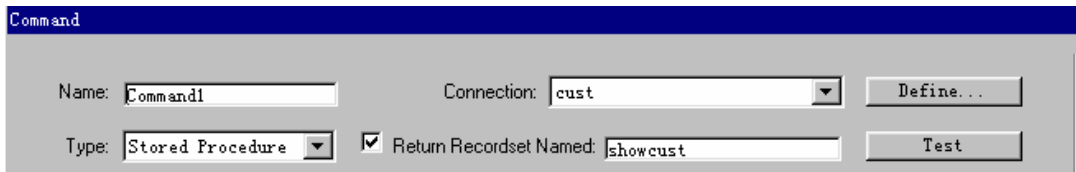


图 21.2 定义存储过程之一

（6）展开窗口下方的存储过程（Store Procedures）列表，选择返回记录集的存储过程，如图 21.3 所示。

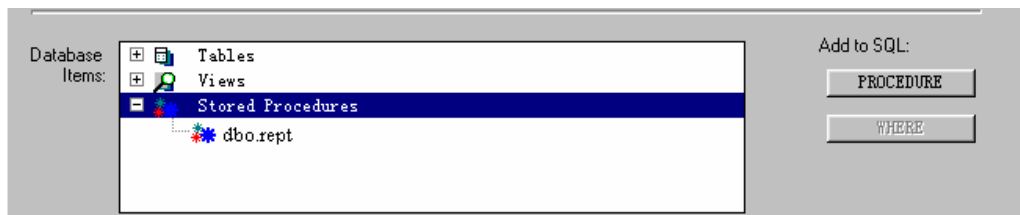


图 21.3 定义存储过程之二

- （7）在 Variables 文本框中输入需要的参数。不要在返回值（return_Value）中输入任何参

数。

(8) 单击“ PROCEDURE ”按钮完成设置工作。

经过前面的设置以后,存储过程对象返回的记录集将出现在数据绑定窗口中。展开记录集各个字段,将每个字段作为数据源可以和网页元素动态绑定。

21.6 在 UltraDev 中将存储过程直接定义为记录集

在 UltraDev 中也可以将存储过程直接定义为记录集来引用存储过程,方法如下。

(1) 打开一需要使用记录集的网页。

(2) 进入数据绑定窗口,单击“ + ”按钮,选择 Recordse(Query) 选项,打开记录集对话框。如果进入的是 Simple 窗口时,单击“ Advanced ”按钮进入 Advanced 对话框。

(3) 为记录集取名,并选择连接对象连接包含存储过程的数据库。设置情况如图 21.4 所示。

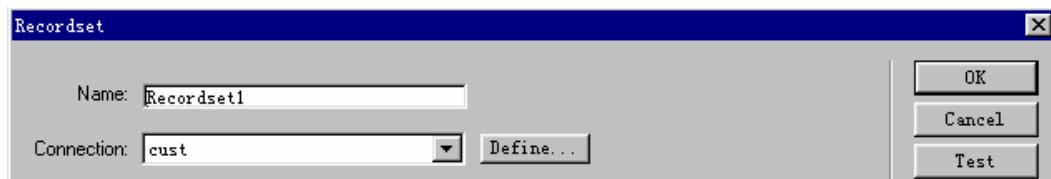


图 21.4 直接将存储过程定义为记录集之一

(4) 展开窗口下方存储过程(Store Procedures) 的列表。选择返回记录集的存储过程,如图 21.5 所示。

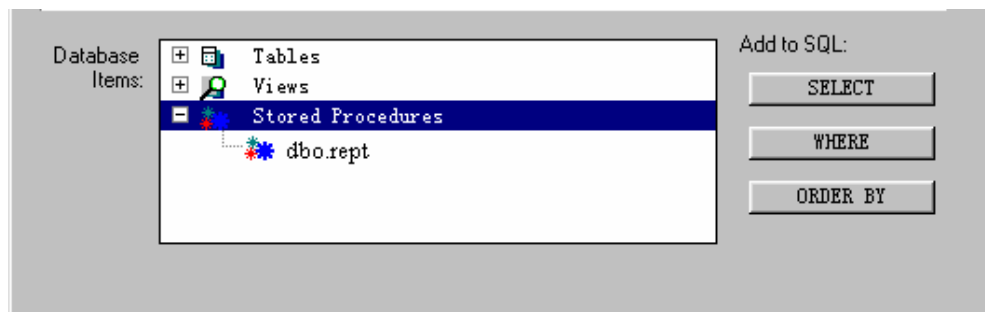


图 21.5 直接将存储过程定义为记录集之二

(5) 如果存储过程需要提供参数时,在 Variables 文本框中输入参数。

(6) 单击“ OK ”按钮完成设置工作。

第 22 章 编辑和自定义服务器行为

Dreamweaver UltraDev 提供了大量服务器行为,同时又为修改这些行为和创建新行为提供了手段。UltraDev 4.0 在服务器行为窗口中新增加的“Edit Server Behaviors”和“New Server Behavior”选项,就是为编辑原有服务器行为和创建新行为提供的服务,这说明系统具备有利用本身提供的工具进行自我完善的能力。

学习编辑和自定义服务器行为可以进一步了解系统编写的源代码的一些特点,为进一步发掘系统潜力准备条件。但这毕竟不是经常要进行的工作。当系统提供的内部脚本(函数)不能完全满足需要时,可以直接在网页的代码窗口中去修改或编写脚本代码,在代码窗口中对代码的改动或者增加的新代码只对当前的网页起作用。只有对于那些普遍性的问题,才有必要编写新的服务器行为代码,以供大家重复使用。

编辑和定义服务器行为的设计者应对系统编码的特点有比较清楚的了解,特别是修改系统内部的脚本时更要慎重,搞不好可能会影响后续系统的运行。

本章将先介绍一般原则,然后通过一个综合示例来说明创建新服务器行为的具体方法。由于内容涉及一些系统内部的编码情况,对于初学者来说可能有些困难,可以暂时跳过这一章,它不会影响对后面章节的学习。

22.1 自定义服务器行为

首先要指出的是,为了实现一项服务器行为可能需要用到多个代码块,其中每段代码又可能需要插入到源代码的不同位置。

22.1.1 编写代码块的步骤

这里先介绍编写代码块的步骤,使读者有一个全局性的了解,然后再比较详细地介绍其中某些步骤的具体方法。

(1) 在服务器行为窗口中用鼠标单击“+”按钮,在下拉列表中选择 New Server Behavior 选项,弹出如图 22.1 所示的对话框。

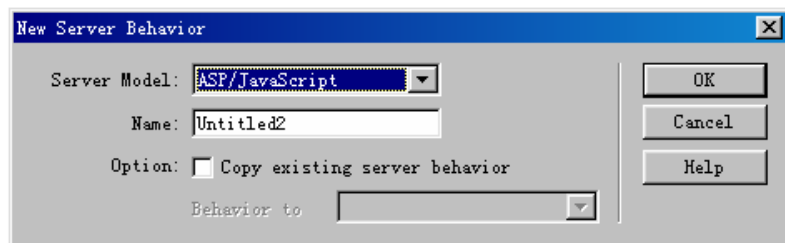


图 22.1 为新服务器行为取名及选择模式

(2) 在 Server Model 下拉列表中输入新服务器行为的模式。这些模式可以在下拉列表中选择，它包括以下几种。

- ASP/JavaScript：使用 ASP/JavaScript 脚本语言。
- ASP/VBScript：使用 ASP/VBScript 脚本语言。
- Cold Fusion：使用 Cold Fusion 脚本语言。
- JSP：使用 JSP 脚本语言。

在此选择 ASP/JavaScript 选项。

在 Name 文本框中输入新服务器行为的名称，然后单击“ OK ”按钮，将弹出如图 22.2 所示的服务器行为编辑器(Server Behavior Builder)对话框。

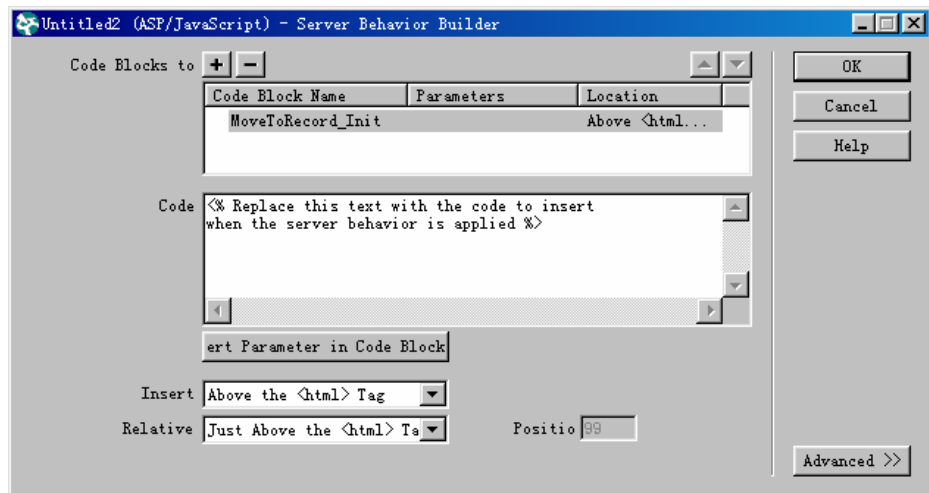


图 22.2 服务器行为编辑器

服务器行为编辑对话框是编辑服务器行为的主要窗口。

(3) 单击对话框中的“ + ”按钮，在弹出的对话框中输入代码块的名字。习惯上代码块的名字采用以下形式：

服务器行为名_代码块名

例如：

MoveToRecord_Init

(4) 在代码(Code)窗口中编写脚本代码。

先将 Code 列表框中“ <% Replace...is applied %> ”中的英文字符去掉(这段英文就是提示你服务器行为代码放置的位置)，然后将新的脚本代码写入<%...%>之间。这些代码连同前后的符号一起将插入到网页的代码中。

(5) 如果要求使用本行为时输入待定参数，代码中要加入参数标志。加入待定参数标志的方法在下面说明。

(6) 利用 Insert 和 Relative 的下拉列表确定代码块在网页代码中插入的位置。具体方法将在下面说明。

(7) 若需进一步设置时，单击“ Advanced ”按钮。

(8) 重复(3) ~ (7) 步以建立多个代码块。

(9) 利用右上方的向上/向下箭头来调整代码块的相对顺序。

(10) 如果不需要网页设计者提供待定参数时，单击“ OK ”按钮。

此后，当单击服务器行为窗口的“ + ”按钮时，将看到一个新的服务器行为名已经出现在行为列表中。这种方式创建的服务器行为将不包括输入待定参数的对话框。

(11) 如果你创建的行为需要网页设计者提供参数时，先输入参数标志，再单击“ Next ”按钮，此时 UltraDev 将帮助你创建输入待定参数时的对话框。

22.1.2 代码的编写方法

上面步骤中的第四步，编写脚本代码是整个过程中的关键步骤，这里对这一步骤的要求作进一步的说明。

代码必须按照 ASP 以及服务器端脚本的语法要求编写。代码应该是紧凑和健壮的，要尽可能地防止可能出现的错误，如：输入参数的错误、记录集找不到记录时的错误等。除此而外还应注意以下几点。

(1) 运行代码必须是单层 (single tag) 结构或者是单一的 ASP/JSP 脚本块。如果需要多层标记或者在脚本之间嵌套时，设法将它们分离成多个平行的单元，分别放到不同的代码块中。

(2) 重名问题必须防止。不论是变量名或者是函数名不能与系统的保留字重复，也不能与 Dreamweaver UltraDev 系统定义的脚本代码重名。

在 Dreamweaver UltraDev 定义的脚本中，不论是变量或者内部函数名都以“ MM_ ”作为前缀。例如：

变量名：var MM_ERROR_STRING = "...";

函数名：function MM_hideLayer() {...}。

定义新行为时，为新行为的函数和变量命名时，最好确定一个有别于其他名字的前缀。例如“ S_ ”、“ P_ ”等。请记住，绝不要用“ MM_ ”作为前缀，以避免重名冲突。

22.1.3 加入待定参数

在新服务器行为中增加待定参数，可以让以后的使用者根据使用的需要给参数命名，从而增加了网页设计者设计的灵活性。如果需要网页设计者提供参数时，必须为新行为增加对话框以提示网页设计者使用时输入待定参数。

首先你必须在代码中增添待定参数，然后再为该参数创建对话框。

(1) 代码中增添待定参数的方法如下。

待定参数的格式是前后加两个“ @ ”符号，如：

@@待定参数名@@

例如一个 Session 变量的变量名是 abcd，其值是 5 时，原来的代码是：

```
<% Session("abcd") = 5; %>
```

现在改用待定参数来定义 Session，应该用下列符号来代替 abcd 和 5。

```
<% Session("@@Session@@")= @@Value@@; %>
```

可以直接将上述代码写入代码块中，也可以先用鼠标选择 abcd(或 5)，然后单击下方的 Insert Parameter In Code Block 按钮，然后输入参数名 Session(或 Value)，再单击“ OK ”按钮。

UltraDev 将自动增加前后的符号 @@...@@。

(2) 创建对话框。

当你增添完待定参数单击“Next”按钮时，系统将显示所有需要增加的待定参数。可通过向上、向下箭头来改变待定参数的顺序，然后单击“OK”按钮。系统将自动为待定参数创建对话框。按照本例，系统创建的对话框包括两个输入项，Session 与 Value。对话框如图 22.3 所示。



图 22.3 为参数创建对话框

当网页设计者使用本行为时，单击服务器行为窗口中的行为名，将弹出此对话框以提示输入待定参数。

22.1.4 确定代码块插入的位置

服务器行为代码编写完以后，还必须确定各代码块插入网页代码中的位置。允许将代码块插在<HTML>标志的前面，也允许插在</HTML>的后面。或者插放在两者之间的某个 HTML 标志的相对位置上。

1. 将代码块插入<HTML>的前面

当需要将代码块插入<HTML>的前面时的做法如下：

- (1) 先在 Insert 代码的下拉列表中选择 Above the <html>Tag 项。
- (2) 再从 Relative 的下拉列表中作进一步选择。

可以选择放在文件的最前面 (beginning of the file)、放在记录集打开之前 (just before open Recordsets)、放在记录集刚刚打开之后 (just after open Recordsets)、正好放在<HTML>标志之前 (just above the <html>) 或者自定义位置 (custom position)。

(3) 如果是自定义位置时，还需在 Position 小窗口中分配权值 (weights)。UltraDev 中的权值是一整型数，权值越大时放置的位置越靠后。权值对于确定多个代码块的分配位置时很有用处。概括地讲，系统将前 50 的数分配给了记录集的代码块。现在如果你有 3 个代码块都想放在记录集的后面，你可以将其中之一的权值分配为 60，另一个分配为 65，第三个分配为 70，用这种方法来确定它们之间的插入顺序。如果你将几个代码块都分配成相同的权值时，系统将

任意安排它们的顺序。UltraDev 4.0 内部脚本的几个主要的服务器行为的权值(weights) 分配如表 22.1 所示。

表 22.1 UltraDev 系统脚本的权值分配表

服务器行为 (Server Behavior)	代码块 (Code Block)	位置 (Location)
Recordset	Connection Ref	Above 10
	Variables	Above 50
	Main	Above 50
	Close	Above 50
Recordset Statistics	declare stats	above 70
	count stats	above 75
	stats in document	replace selection
Repeat Region	RR init	above 70
	begin loop	before selection
	end loop	after selection
Move To	declare stats	above 70
	declare vars	above 81
	move to specific-param	above 82
	move to extra nav	above 84
	move to update stats	above 85
go to detail	move to keep params	above 90
	goto attribute in document	attribute a.href
update record	Connection Ref	above 10
	editops variables	above 15
	update record set vars	above 20
	update record query	above 30
	edittops from attribute	attribute of from action
	edittops method attribute	attribute of form method
	update hidden attribute	attribute of hidden
	editop hidden id	attribute of hidden

[注意]：凡与记录集(Recordset) 有关的行为都要放在记录集创建之后，否则会出现错误。

2. 将代码块插入</html>的后面

(1) 在 Insert 代码的下拉列表中选择 Below the </html> Tag 项。

(2) 从 Relative 的下拉列表中作进一步选择。你可以选择正好放在</HTML>标志之后(just after the </html>) 放在记录集关闭之前(just before close Recordsets) 放在记录集刚好关闭之后(just after close Recordsets) 或者整个文件关闭之前(just before the end of file) 或者自定义位置(custom position)。

(3) 如果是自定义位置时, 还需在 Position 小窗口中分配权值(weight)。分配方法同前。

3. 插入到某个 html 标志的相对位置

(1) 在 Insert 代码的下拉列表中选择 Relative To a Specific Tag 项。

(2) 在标志窗口中键入标志名或从下拉列表中选择。注意键入标志时前后不要加尖括号。

(3) 在 Relative 的下拉列表中选择与标志的相对位置。可以选择该标志刚好打开之前或关闭之后、用代码取代标志、将代码作为标志的属性值(将弹出窗口让你选择属性) 将代码放在打开标志的内部。

4. 由网页设计者决定插入位置

(1) 在 Insert 代码的下拉列表中选择 Relative To the Selection 项。

(2) 在 Relative 的下拉列表中确定与选择项的相对位置。

(3) 可以选择代码放在该选择之前或之后; 替换该选择或者将代码环绕该选择的周围。

(4) 如果你要将代码环绕在某个选择的周围时, 该选项必须包括打开和关闭标志。而且两标志之间不能包括其他标志。例如用<CFIF>作为打开标志, 用</CFIF>作为关闭标志时:

```
<CFIF Day="Monday"></CFIF>
```

代码中的打开部分(opening tag piece) 将放在打开标志的前面; 代码中的关闭部分将放在关闭标志的后面。

22.2 编辑服务器行为

可以在服务器行为编辑器中编辑任何服务器行为, 包括自己定义的或者系统提供的服务器行为。编辑的步骤如下。

(1) 在服务器行为窗口中单击“ + ”按钮, 从下拉列表中选择 Edit Server Behavior 项, 弹出如图 22.4 所示的对话框, 在此对话框中将列出全部服务器行为。

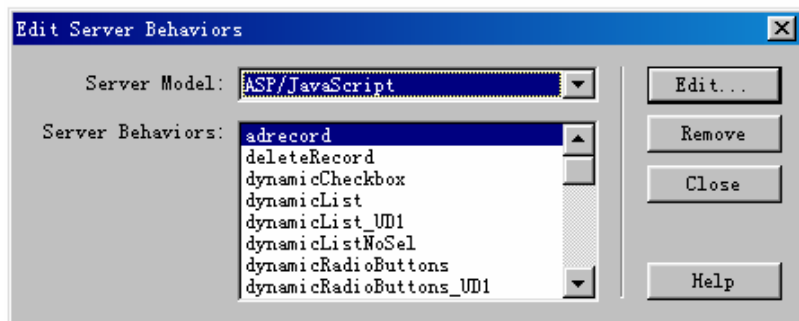


图 22.4 服务器行为列表

(2) 选择其中一项服务器行为后单击“ Edit ”按钮，将弹出如图 22.5 所示的服务器行为编辑窗口。

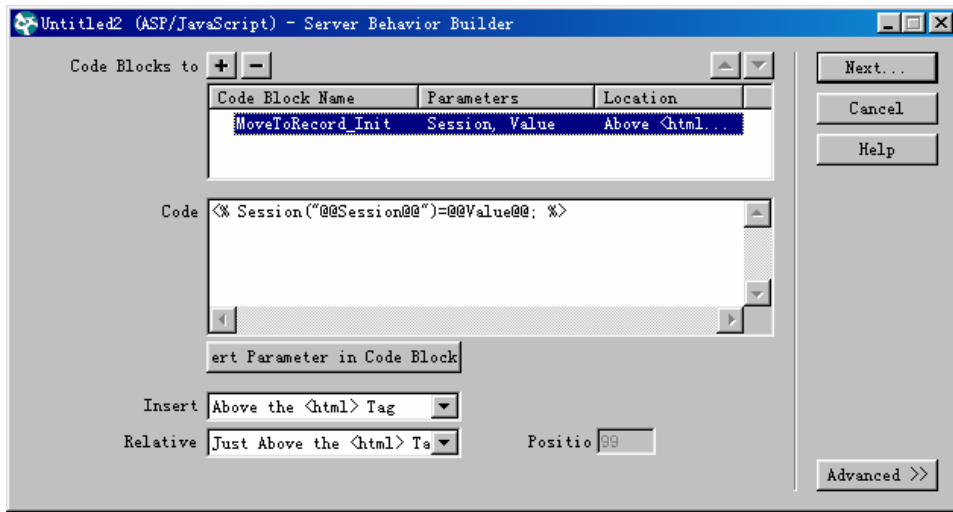


图 22.5 显示源代码窗口

- (3) 单击某行为块后修改其代码。
- (4) 如果需要，在代码中改变或增添参数标志。
- (5) 通过下拉列表改变或设置各代码块在网页代码中插放的位置。
- (6) 如果代码中不包括网页设计者提供的参数时，单击“ OK ”按钮。一个新代码块将出现在服务器行为的下拉列表中。此代码不包括对话框。

(7) 如果修改的代码需要网页设计者提供参数时，单击“ Next ”按钮。UltraDev 将提问怎样创建对话框，设置完后单击“ OK ”按钮。UltraDev 将把改变后的行为存放在行为的 XML 文件中。

22.3 创建服务器行为示例

22.3.1 创建过程

下面通过一简单示例来说明服务器行为的创建过程。假定有一显示人事档案 (摘要) 的网页，从数据库中提取数据显示以下 6 项，如图 22.6 所示。

现在将表格中的数据与数据表对应的字段动态绑定。比如备注 ("BZ") 字段的数据绑定代码是：

```
<td width="69%">
    <div align="center"><%= (Recordset1.Fields.Item("bz").Value)%></div>
</td>
```

人事档案摘要:

编号:	{Recordset1.bh}
姓名:	{Recordset1.xm}
性别:	{Recordset1.xb}
部门:	{Recordset1.bm}
职称:	{Recordset1.zc}
备注:	{Recordset1.bz}
提交 复位	

图 22.6 人事档案界面

由于“备注”栏只用于对一些特殊情况的说明，不少人的备注栏是空的，其他如职称栏也有类似的情况（不少人没有职称）。现在要求创建一通用的服务器行为，其作用是当某项数据空缺时，该项被压缩，不显示在浏览器中。一旦以后又增加了相应的数据时，又自动显示出来。为此需按照以下的方法设计服务器行为。

（1）在服务器行为窗口中选择“New Server Behaviors”项，如图 22.7 所示。

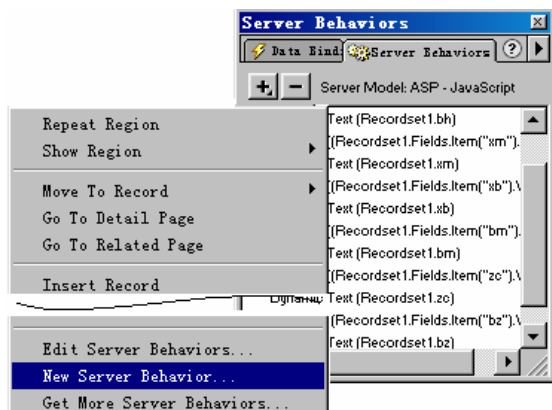


图 22.7 开始创建新服务器行为

（2）在弹出的窗口中选择模式并为新行为命名。

本例中选择“ASP/JavaScript”模式，新行为取名为“Hide when Column is null”，如图 22.8 所示。

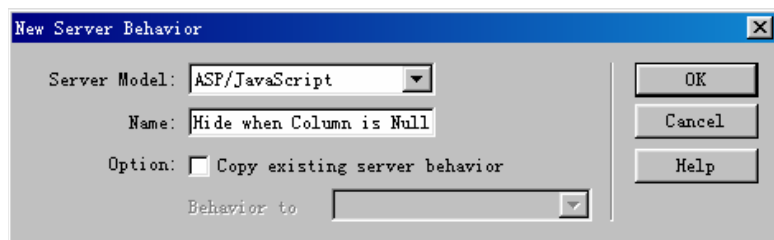


图 22.8 选择 Server Model 及 Name

(3) 单击“OK”按钮，在弹出的对话框中单击“+”按钮，为代码块取名(这里就利用缺省名)，然后将备注栏的动态绑定的语句暂时拷贝到代码(Code)窗口中来，如图 22.9 所示。

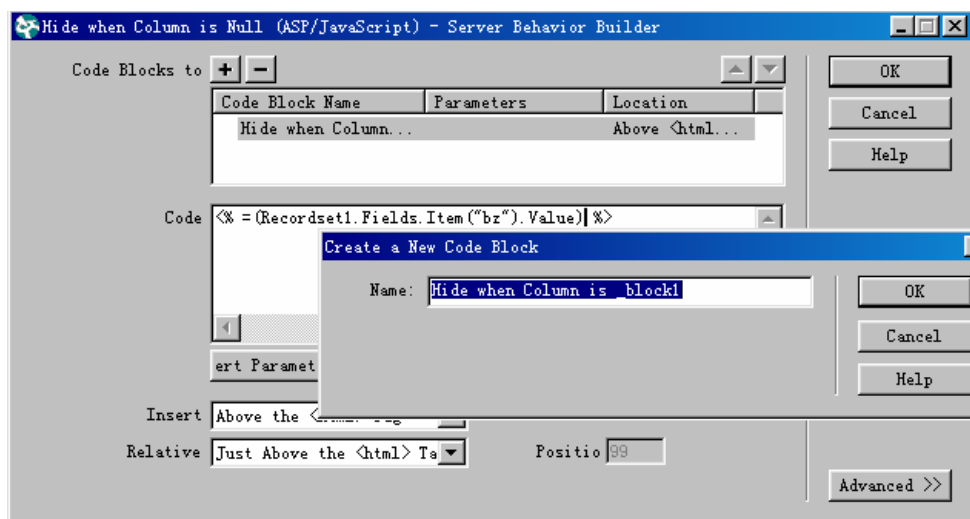


图 22.9 为代码块取名

(4) 按照以下内容改写代码：

```
<% if (Recordset1.Fields.Item("bz") != null) { %>
```

为了将 Recordset1 变成待定参数以增加脚本的通用性，先用光标选择代码中的 Recordset1，然后单击“Insert Parameter in Code Block”按钮。

在弹出的窗口中将参数名(Parameter Name) 改为 RecordsetName，如图 22.10 所示。

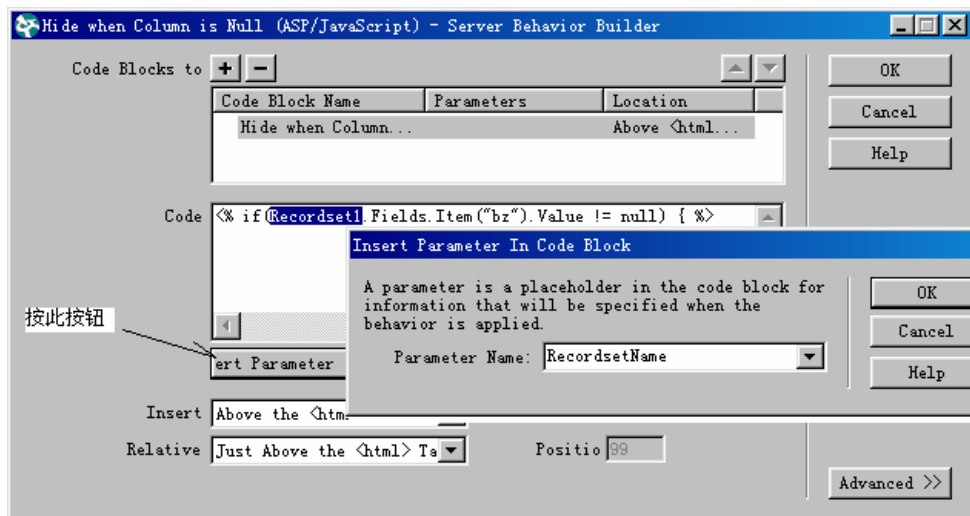


图 22.10 修改原代码

(5) 单击“OK”按钮。

(6) 为了将 bz 字段变成待定参数以增加脚本的通用性，用同样的办法选择字段名“bz”

后为第二个参数取名为“ ColumnName ”。此时系统将为这两个参数的前后都加上了“ @@...@@ ”符号。

(7) 按照如图 22.11 所示的方法为第一个代码块确定插入网页代码的位置。

- 在 Insert 下拉列表中选择“ Relative to the Selection ”。
- 在 Relative 下拉列表中选择“ Before the Selection ”。

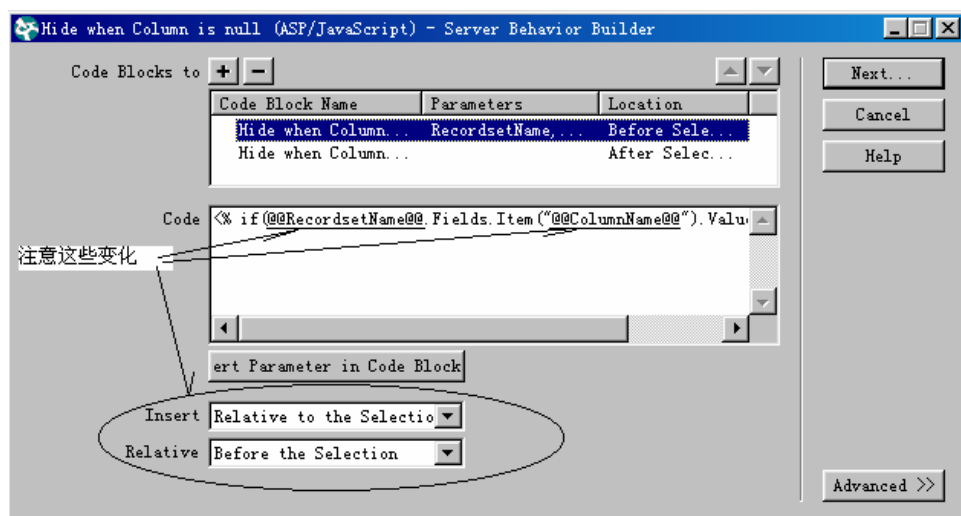


图 22.11 确定代码插入位置

(8) 按照图 22.12 所示的方法设置第二个代码块，方法与第一代代码块设置相同。

其代码是： <% } %>

放置的位置是：“After the Selection”

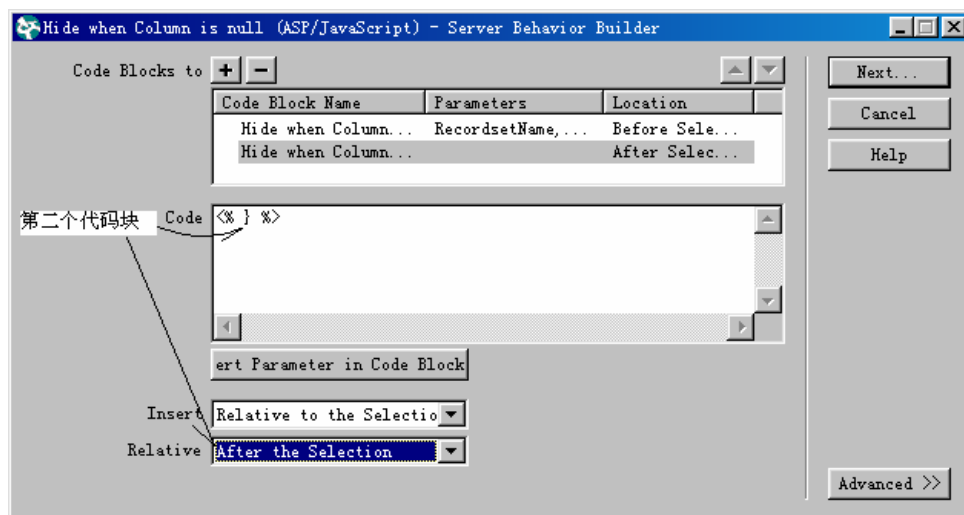


图 22.12 设置第二个代码块

(9) 单击“ Next ”键后，为待定参数创建对话框。

- 先将 RecordsetName 升至第一行。
- 在 Display 下拉列表中选“ Recordset Menu ”项。

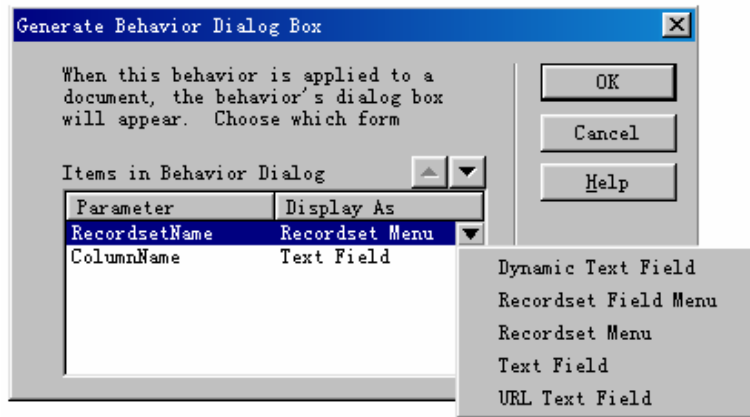


图 22.13 选用 Recordset Menu

(10) 单击“ OK ”按钮，整个创建工作完成。

22.3.2 使用新服务器行为

为了使用这一新服务器行为，首先在网页的设计界面上，选择“备注”栏（包括标题及数据两个单元），然后在服务器行为窗口中选择新行为“Hide when Column is null”。将弹出待定参数对话框，填入字段名。此时网页中的源代码已经有了变化。变化的情况如下。

```
<% if(Recordset1.Fields.Item("bz").Value !=null) { %>
    <td width="31">
        <div align="center">备注：</div>
    </td>
    <td width="69%">
        <div align="center"> <%= (Recordset1.Fields.Item("bz").Value) %> </div>
    </td>
</tr>
<% } %>
```

注意在备注标题的前面增加了以下语句。

```
<% if(Recordset1.Fields.Item("bz").Value !=null) { %>
```

在单元格的后面</tr>增加了：

```
<% } %>
```

它说明只有在“ bz ”栏不空的情况下，才显示这个单元格。如果对“ 职称 ”栏也有同样的要求时，先选择“职称”栏（包括标题及数据两个单元），再直接使用已经定义的方法即可。

第 23 章 登录检查与网页保护

23.1 概述

Internet 虽然是一种开放型的网络系统，然而其中有些网页并不是对所有的人都开放的，例如：

- 一些用于公司内部管理的网页只对公司内部人员开放；
- 有些网站设立的收费项目，只对那些进行了登录并交纳了一定费用的客户开放；
- 有些商业网站实行“会员制”，只有事先注册的会员，才有权参加某些商业交易的活动等等。

以上情况都可以用设置口令，登录检查的方法，防止未经授权的人进入特定的网页参与一些特定的网上活动，以保证网站的合法权益，保护商业秘密和电子商务的安全。

另外还要看到，网页的保护是有层次的。以内部管理的网页为例，公司内部的职员通过登录检查可以进入网站管理部分，但也不是说能够访问管理部分的所有网页。由于职务、分工不同，能够访问的网页也有所不同。为了实现这种区别，可以先给公司人员划分成不同的“角色”，并为各类“角色”设置访问网页的第二道“防线”。

Dreamweaver UltraDev 4.0 版本增加的登录检查、网页保护等服务器行为，为设计这些网页提供的帮助，大大地简化了设计过程。新增加的服务器行为放在菜单 Window|Server Behaviors|Authentication 的选项中，该选项包括以下 4 部分。

- Log In User：客户的登录检查。
- Restrict Access To Page：对网页访问的限制。
- Log Out User：客户退出登录时的善后处理。
- Check New Username：在注册表中增添新客户时的重名检查。

下面逐个介绍实现上述 4 种功能的网页的设计方法。

23.2 登录检查网页的设计

为了进行登录检查，需事先将有关人员（或会员）的姓名、口令登记到一张注册表中并为各个客户分配一定的角色（或称访问等级）。表 23.1 就是一张注册表的示例。

表 23.1 注册表示例

M_bh	M_name	M_pass	M_job
001	刘伟大	XT2001	M_system
002	陈有清	RS2001	M_person

续表

M_bh	M_name	M_pass	M_job
003	王强	YW2001	M_business
004	张友谊	YW2002	M_business
005	李大有	YW2003	M_business
006	成规立	CW2001	M_finance

表中：

M_bh： 代表客户的编号。

M_name： 代表客户的姓名。

M_pass： 代表客户的口令。

M_job： 代表由系统管理人员为客户分配的角色。

当客户访问某个特定的网页时,要求先输入姓名和口令,然后系统到注册表中去查找和匹配,如果找到该姓名和口令时,说明此用户是合法的客户(或会员),自动取出该客户对应的角色(或访问等级),然后引导该客户转向相关的主页,否则转向登录失败的提示网页。

为此,创建登录网页的同时至少还应设置两张网页:一张是登录成功时转入的网页,通常就是某项工作的主页;另一张是登录失败后用于提示的网页。

创建登录网页的步骤如下。

(1) 在网页中设置表单(Form), 表单中放置表格和姓名、口令的文本输入框以及提交按钮等网页元素,其中用于口令的文本输入框,应在它的属性面板中设置成 Password 类型,登录的界面如图 23.1 所示。

图 23.1 登录的界面

(2) 选择菜单 Window|Server Behaviors 项,进入服务器行为窗口,单击窗口中的“+”按钮,在弹出的菜单中选择 User Authentication|Log In User from the pop-up menu 项将弹出如图 23.2 所示的 Log In User 对话框。

(3) 在 Log In User 窗口中设置有关数据。

对话框的数据分为四层。

第一层

- Get Input From Form 为输入姓名和口令的表单名。
- Username Field 为输入姓名的文本输入框名。
- Password Field 为输入口令的文本输入框名。

第二层为注册表的有关数据

- Validate Using Connection 为与注册表链接的连接对象名。
- Table 为注册表名。

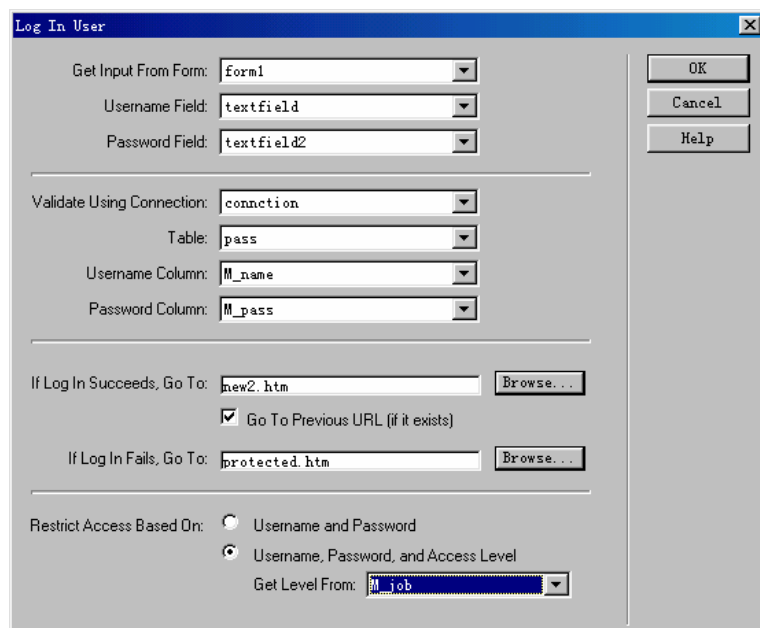


图 23.2 Log in User 对话框

- Username Column 为注册表的姓名字段名。
- Password Column 为注册表的口令字段名。

第三层为完成输入后转向的网页

■ If Log In Succeeds, Go To 为登录成功时转向的网页,通过 Browser 文件夹图标选择或直接输入网页名。

■ If Log In Fails, Go To 为登录失败时转向的网页,通过 Browser 文件夹图标选择或直接输入网页名。

第四层为限制访问的依据

■ Username and Password: 姓名和口令是访问限制的依据。

■ Username, Password, and Access level: 姓名、口令和“角色”(访问权限)是访问限制的依据。如果选择本项时还需要在 Get Level From 的下拉列表中确定注册表中“角色”(访问权限)相对应的字段。

(4) 单击“OK”按钮完成登录检查网页的设计。

23.3 对网页的保护

登录检查网页能够通过姓名、口令检验访问者的身份,但是如果客户直接利用主页的 URL 进入,绕过登录检查网页时,也就避开对身份的检查,前面设置的保护也就失去了作用。因此,在设计登录检查网页的同时还必须采用其他网页的保护措施,才能真正达到保护网页的目的。设置网页保护措施的步骤如下。

(1) 打开被保护的网页。

[注意]: 本方法只能保护*.asp 网页而不能保护*.htm 网页。

(2) 选择主菜单中的 Server Behavior|User Authentication|Restrict Access To Page 选项，将弹出如图 23.3 所示的对话框。

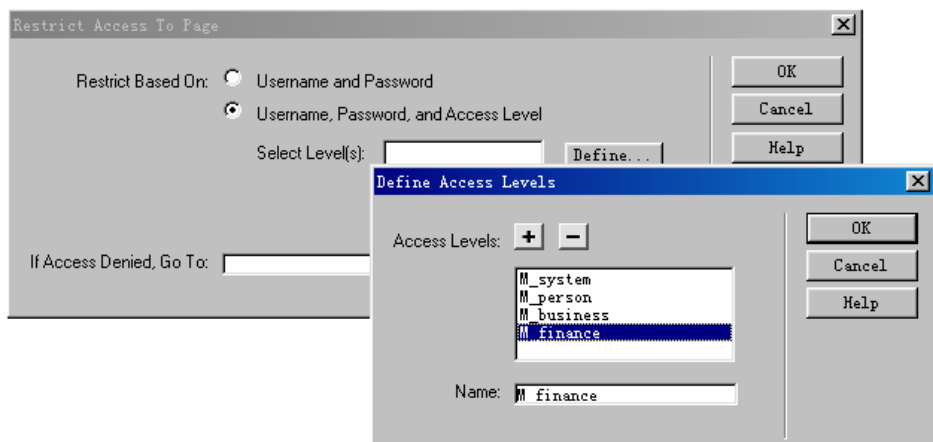


图 23.3 对网页保护的设置

(3) 选择 Username, Password, and Access Level 选项后单击“ Define ”按钮。

(4) 在新弹出的窗口中，先按 “ + ” 号，然后再通过 Name 文本框将各类角色的名字填入 Access Levels 中。输完后单击“ OK ”按钮返回前一个界面。

(5) 在返回的窗口中确定允许访问的“ 角色 ”以及拒绝访问时转向的网页。设置情况如图 23.4 所示。

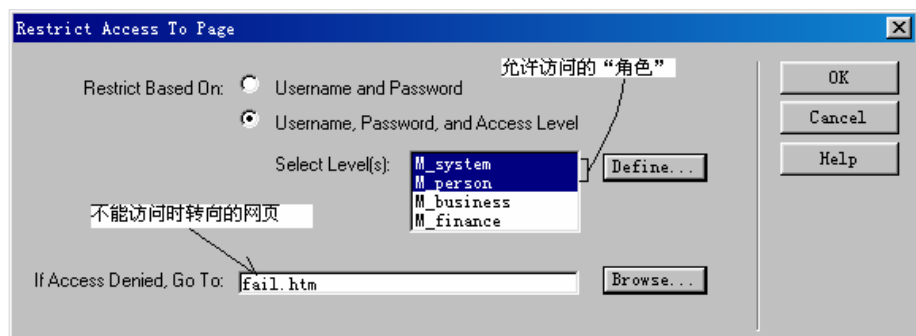


图 23.4 确定允许访问本网页的“ 角色 ”

图 23.4 中的设置代表有两种角色允许访问本网页，它们是 M_system 和 M_person。其他角色想要访问本页时，将遭到拒绝并转向 fail.htm 网页。

设置完成后，服务器行为窗口中将出现“ Restrict Access To Page ”标记。

若多个网页的保护条件相同时，可直接利用 Copy / Paste 方法将上述标记粘贴到其他网页的服务器行为窗口中去，以简化重复设置的过程。

具体方法是，在已经设置好保护的网页的服务器行为窗口中，按右上角的  按钮，在弹出的菜单中选择 Copy 项；然后到需要设置网页保护的网页的服务器行为窗口中按  按钮后在弹出的菜单中选择 Paste 项，就可以将原来的设置拷贝过来，如图 23.5 所示。

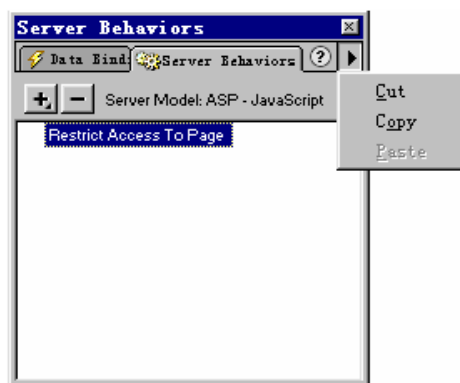


图 23.5 拷贝网页保护的设置

23.4 网页保护的原理

为什么上面的设置能保护网页呢？下面通过对代码的分析来了解一下网页保护的原理。

系统中是通过设置会话级共享变量 Session 来实现网页保护的。Session 变量在这里好比一把“锁”，放在被保护网页的前面。当客户要访问本网页时，先检查 Session 变量是否合乎要求，合乎要求时才允许进入，否则转向其他用于提示的网页。

系统使用的 Session 变量是 Session("MM_Username")，它是在登录检查网页中赋值的，如果客户绕过了登录检查网页直接进入本网页时，该变量仍处于"undefined"状态，因而可以识别出这是一位不走“正门”的客户，应该拒之门外。

对于那些通过登录网页的客户，如何检查他们的访问权限呢？作为第一步，根据客户的姓名和口令在注册表中进行匹配来检查和筛选。注册表中未登记的客户被拒之门外。

对于通过了第一道检查的客户，如何进一步判断他的访问权限呢？前面的设置已经作好了两项准备工作。

(1) 在登录网页匹配客户的姓名和口令时，已经从注册表中提取出该客户所对应的角色(等级)，并放入 Session("MM_UserAuthorization")变量中。使用的语句是：

```
Session("MM_UserAuthorization")=String(MM_User.Fields.Item("MM_fld").Value);
// MM_User 是注册表；MM_fld 代表角色(等级)字段。
```

(2)而在被保护的网页中设置网页保护的过程中已经将允许访问的“角色”组成一字符串，并放入变量 MM_authorizedUsers 中。以图 23.4 的设置为例，该变量的设置是：

```
var MM_authorizedUsers="M_system, M_person";
// 代表允许 M_syste 及 M_person 角色(等级)访问。
```

在上述两项准备工作的基础上，当客户从登录检查网页转向本网页时作最后的判断，判断的语句是。

```
If ( MM_authorizedUsers.indexOf (String(Session("MM_UserAuthorization"))>=0)) )
{允许进入本网页}
else
```

{转向无效网页}

语句中的 `indexOf()` 是 JavaScript 的内部函数，它的作用是在母串中匹配子串，其格式是：
`Nn=String1.indexOf( SubStr[,Start] )`；

其中：`String1` 代表母串，`SubStr` 代表子串，`Start` 代表匹配的起始位置，如果不指定起始位置时，从母串的第一个字符开始匹配，`Nn` 为返回值，当找不到匹配字段时，返回-1，其他情况返回匹配字符串的第一个字符在母串中的下标。

现在是在母串 `MM_authorizedUsers` 中匹配子串 `MM_UserAuthorization`。如果母串中包含子串，说明新访问的客户包含在允许访问的“角色”之中，因此允许进入本网页，否则转向无效网页。

23.5 退出时的善后处理

客户登录成功时系统已经创建了包括客户名 `Session` 在内的各种变量。当客户离开网站时，可以利用 `Log Out User` 服务器行为来清除所有 `Session` 变量并且转向另一网页作些善后处理，例如表示“再见”或者“欢迎再来”等等。

建立 `Log Out User` 行为的方法如下。

- (1) 先确定一链接字符串(本例中使用“退出网站”)。
- (2) 选用服务器行为中的 `Log Out User` 选项，将弹出如图 23.6 所示的对话框。

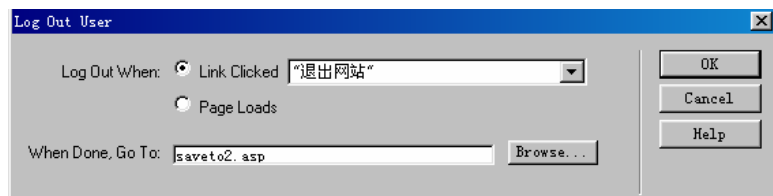


图 23.6 `Log Out User` 网页设置

(3) 可以利用 `Link Clicked` 转入一链接的网页，在该网页中处理退出时的善后事宜。也可以选择 `Page Loads` 项，此时一进入本网页立即自动处理善后事宜。

- (4) 利用 `Browser` 文件夹图标或者直接填入完成处理后转向的网页(`When Done, Go To`)。
- (5) 单击“ `OK` ”按钮结束设置。

系统处理善后事宜的方法就是调用 `Session.Abandon()` 函数，以终止各个 `Session` 变量的作用。

23.6 重名检查

在注册表中增添新客户时，为了保证注册表中的用户名(或口令) 的唯一性，可利用系统提供的 `Check New UserName` 服务器行为。

1. 网页设计的步骤

(1) 为了在注册表中增添新成员，需首先创建一个向注册表中增添记录的网页。网页中放入表单(Form) 表单中放入文本输入框(textfield) 及提交按钮(submit)。选择服务器行为的 Insert Record 选项以便向注册表中增添记录。此时将弹出如图 23.7 所示的对话框，在对话框口中按照增添记录的要求填写参数。

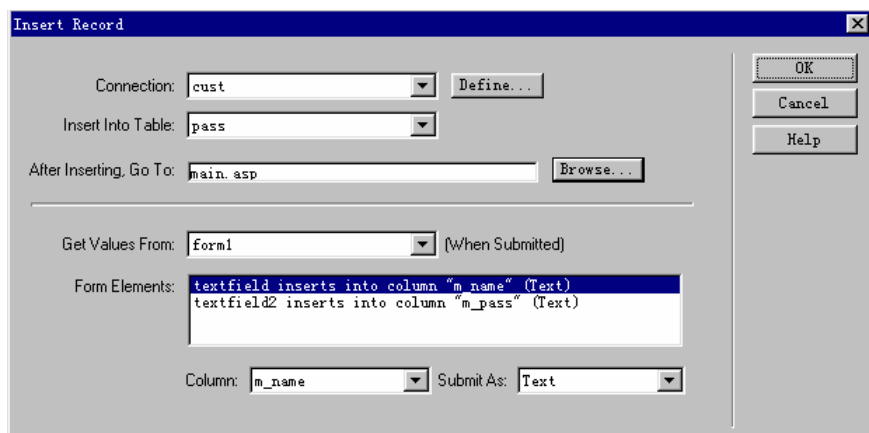


图 23.7 Insert Record 对话框

(2) 选择主菜单中的 Server Behavior|User Authentication|Check new UserName 选项。将弹出如图 23.8 所示的对话框。

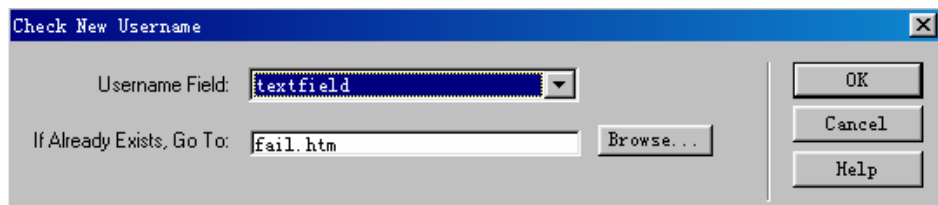


图 23.8 重名检查的设置

(3) 在 Username Field 的下拉列表中选择不可重名的字段。可以选择姓名对应的字段也可以选择口令对应的字段。

(4) 在 If Already Exists, Go To 的文本框中确定一旦不允许重名的字段出现了重名时转向的网页。

(5) 单击“ OK ”按钮完成设置。

2. 代码分析

程序设计的思路是首先取出不允许重复的文本框的数据，然后到注册表中去查找对应的字段，如果找不到，允许加入新记录，否则说明新输入的数据与数据表中的数据重复，不允许增加该条记录，转向提示的网页。

简化后的代码如下。

```
<%  
var MM_dupKeyUsernameValue = String(Request.Form("textfield2"));  
// 提取文本框的数据放入 MM_dupKeyUsernameValue 变量中
```

```
var MM_dupKeySQL = "SELECT m_pass FROM pass
WHERE m_pass=" + MM_dupKeyUsernameValue + ""
// SQL 的查询语句
var MM_adodbRecordset = "ADODB.Recordset";
var MM_rsKey = Server.CreateObject(MM_adodbRecordset); // 创建记录集
MM_rsKey.ActiveConnection = MM_rsKeyConnection;
MM_rsKey.Source = MM_dupKeySQL; // 执行前面的 SQL 查询语句
MM_rsKey.CursorType=0;
MM_rsKey.CursorLocation=2;
MM_rsKey.LockType=3;
MM_rsKey.Open();
if (!MM_rsKey.EOF || !MM_rsKey.BOF) { // 既不是文件头又不是文件尾，说明记录找到
    // the username was found - can not add the requested username
    ...
    Response.Redirect(MM_dupKeyRedirect); // 跳转到添加失败的网页
}
MM_rsKey.Close();
}
```

%>

第 24 章 留言板

留言板可以为客户提供网上发言的机会，它不受时间、地点的限制，是及时获取客户反馈信息的一种好方式。

网站的留言板应能给客户提供一个简单、友好的输入界面；还要能自动保存这些信息并允许客户随时查看其他客户的留言。

留言是一种面向社会开放的机制，因此也难免不会出现一些不健康的留言，应该允许网站管理人员随时或定期地清理那些过时的信息或者信息“垃圾”。

设计留言板网页要综合应用增添记录、校验输入、登录检验、等技术，在删除留言中还要用到批量删除记录的命令对象(Command) 等。利用 Dreamweaver UltraDev 提供的服务，可以很方便地实现这些综合功能。

24.1 进入留言板

图 24.1 是一个简单的留言板界面。界面中只包括 3 个链接指针：开始留言、查看留言与留言管理。通过它们打开留言区网页、查看网页以及管理留言网页。



图 24.1 简单的开始界面

24.2 留言网页的设计

留言网页的设计需按照以下步骤进行。

(1) 为了保存客户的留言，须先建立一数据表，包括留言编号、姓名、主题、电子邮件、留言时间以及留言内容等几个字段，其中留言编号是关键字（不能重复），在 Access 数据表中

用自动增量方式生成，但不一定显示在界面上。

为了保留客户当时的留言时间，数据表中应利用 Access 数据表的内部函数 Now()。具体方法是在 Access 数据表的设计界面中将“留言时间”字段设置成日期/时间类型，然后用鼠标单击默认值右方的  按钮，在弹出的表达式生成器中选择函数|内置函数|日期/时间|Now。上方的框体将出现 Now() 符号。单击“确定”按钮完成设置工作。此时数据表的留言时间字段（m_Date）的默认值就是留言的当时时间。

设置的过程如图 24.2 所示。

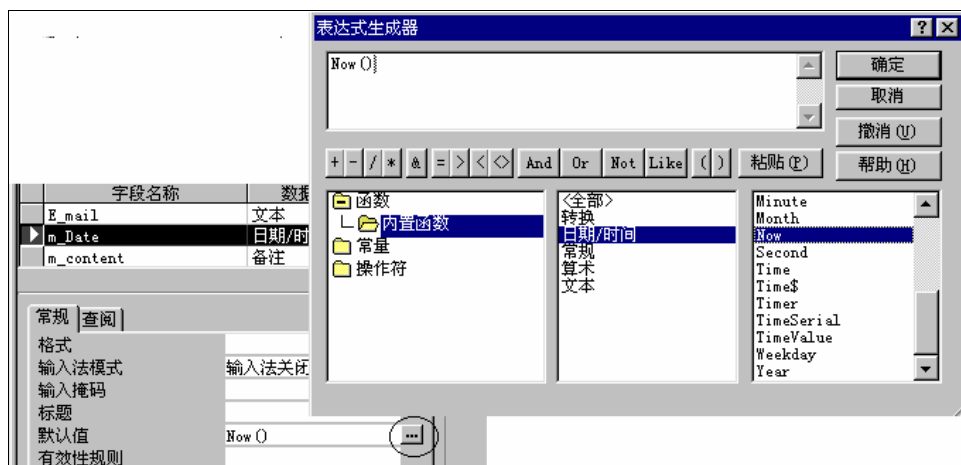


图 24.2 在 Access 表中为日期/时间设默认值

(2) 在网页的表单中放入表格、文本输入框、提交、复位按钮等。其中留言内容的文本框 Type 设成多行 (multiline) 属性。留言板的界面如图 24.3 所示。



图 24.3 留言界面

(3) 利用服务器行为中的 Insert Record 项使各个输入框与留言表相对应的字段挂接，如图 24.4 所示。

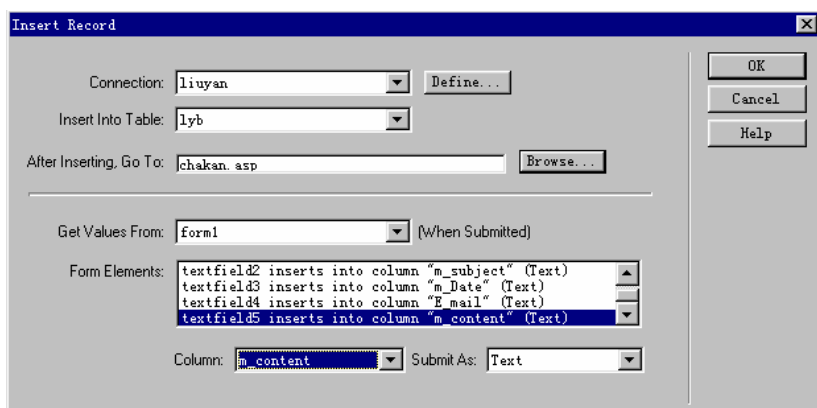


图 24.4 插入记录设置

(4) 为了减少留言过程中填写错误, 利用浏览器端行为的 Validate Form 选项。确定那些项不能缺省, 比如内容不能空缺、E-mail 必须按照一定格式填写等, 如图 24.5 所示。

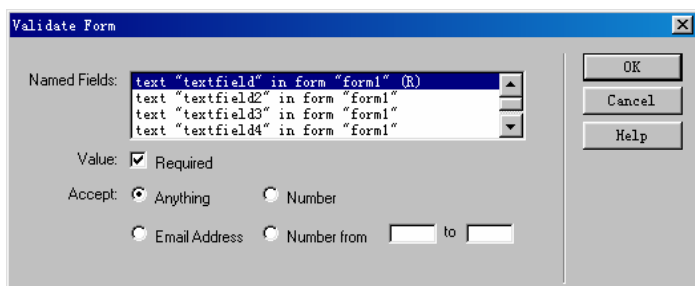


图 24.5 输入校验

24.3 查看留言

查看留言的网页除按照通常使用的方法将表格或文本框与数据表的相关字段动态捆绑在一起, 还应当设置成多条显示, 而且显示的内容应该按照留言时间的先后, 由近而远地显示出来。为此记录集的 SQL 语句应按照成以下形式编写:

```
Select * from lyb
```

```
Order by m_Date DESC
```

其中 lyb 代表留言的数据表, m_Date 代表数据表中的时间字段, DESC 代表按降序排列。

24.4 留言板的管理

网站管理人员应该定期或不定期地对留言板进行清理, 及时删除一些过时的信息或者一些“信息垃圾”。为此必须为管理留言板的工作设计一些网页, 要求进入这些网页的人应该具有一定的权限。

主要的网页有两个：一个用于选择删除的留言项，另一个用于执行批量删除工作。执行的过程是：管理人员先通过复选框选择一条或多条需删除的留言，然后转移到执行网页，在执行网页中按照预先的选择进行批量删除。下面分别讲述这两种网页的设计方法。

为了由浅入深地说明问题，先假定只删除单条记录，然后再说明一次删除多条记录的情况。

24.4.1 删除单条留言

1. 选择删除留言网页的设计

网页上放置一表单 (Form)，表单内放一表格，表格内放入文本输入框以及提交、复位按钮以及浏览按钮等。并将 Form 的 Action 指向执行删除的网页，将 Method 设为 Post。

为了提供选择功能，需为每项留言前增加一个复选框。方法是：用鼠标右击表格的第一个单元格（如“姓名”单元格），在弹出的菜单中选择 Table|Split Cell 项，在弹出的小窗口中选择 Column；在 Number of 中选择 2，就可将该单元格划分成左右两部分，在左边的空格中放入一 checkbox 复选框。并将复选框的输出值与留言编号动态绑定，如图 24.6 所示。

绑定的语句是：

```
<input type="checkbox" name="checkbox" value = "<%= (Recordset1.Fields.Item ("m_bh"). Value)%>" >
```

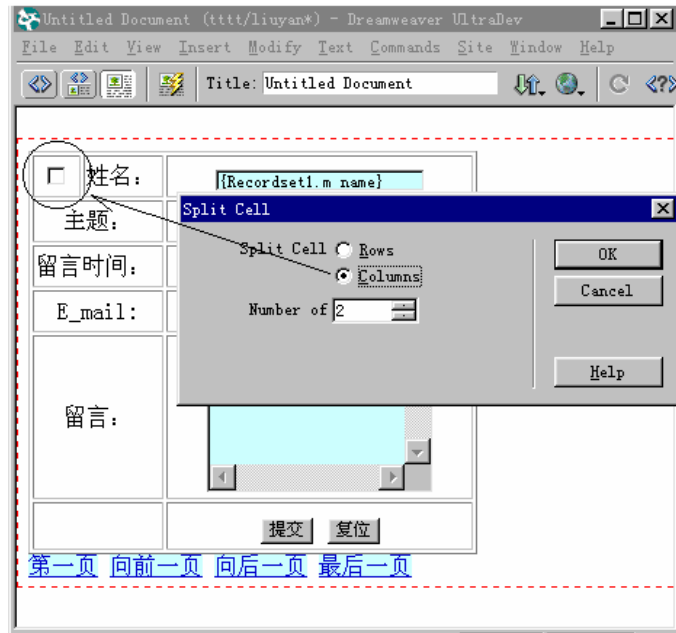


图 24.6 增加复选框

2. 执行删除留言网页的设计

在执行删除留言的网页中，将利用命令对象 (Command) 来完成删除记录的操作。

(1) 打开网页的数据绑定 (Data Binding) 窗口，单击“+”按钮，选择 Command (Stored Procedure) 选项，将弹出命令 (Command) 对话框，如图 24.7 所示。

(2) 在 Name 文本框中填入命令对象名，选择 Connection 对象指向留言表的 DSN，在 Type 的下拉列表中选择 Delete 项。

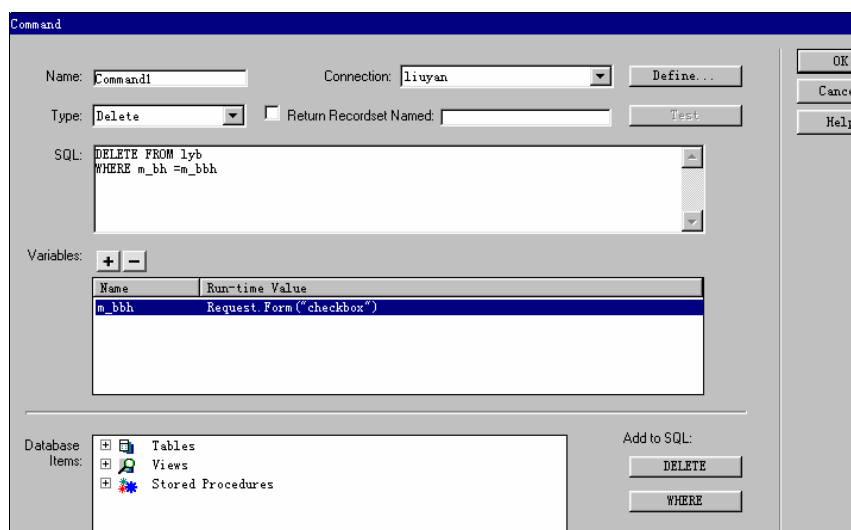


图 24.7 删除单条留言

(3) 在 SQL 窗口中填入以下语句。

```
DELETE FROM lyb
```

```
WHERE m_bh = bbh
```

在 Variables 列表框中先单击“ + ”按钮，在 name 和 Run-time Value 的下面分别填上：

bbh 和 Request.Form ("checkbox")；

其中 lyb 代表留言表；m_bh 代表留言编号，checkbox 代表复选框的组件名。

此语句表明删除前一个网页中被选中的复选框 (checkbox) 的留言。

(4) 在网页的显示界面上写出“ 该记录已被删除 ”等字符串。

24.4.2 批量删除留言

1. 选择删除多条留言的网页

为了选择多条留言，需利用服务器行为项 Repeat Region 设置成多条显示。这样做的结果在网页中展开的多条留言中出现了多个 checkbox 的重名现象。如何处理多个重名复选框的输出是这里要解决的主要问题。

多个重名的复选框被选中时，复选框将要输出的是所有被选中复选框输出值连接成的字符串。由于各复选框的输出已经和留言编号捆绑在一起，因此它的输出就是各个被选择删除的留言编号。例如选择了第 2、4、7 三条留言时。Checkbox 的 value 输出将是 2、4、7。

2. 执行批量删除记录的网页设计

(1) 和删除单条记录一样打开命令对象 (Command) 窗口，如图 24.8 所示。

(2) 将 SQL 列表框中的语句改写成：

```
DELETE FROM lyb
```

```
WHERE m_bh in ( s_bh )
```

(3) 在 Variables 列表框的 Name 和 Run-time Value 下面分别写上：

s_bh 及 Request.Form ("checkbox")

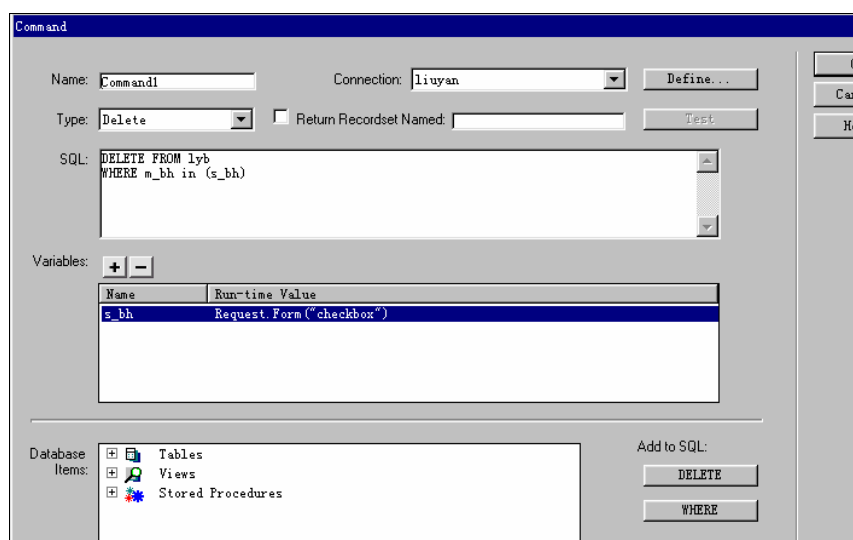


图 24.8 批量删除记录的设置

上述语句的含义是凡是留言编号(m_bh) 属于集合(s_bh) 的一律删除。而 s_bh 是从 checkbox 获得的所有被选中的留言编号。

24.4.3 权限审查和网页保护

对留言的清理属于管理员的职责范围，因此对这上述两页都应该设置登录检查和网页保护，具体做法见第 23 章，这里就不再重复了。

第 25 章 创建虚拟购货车

25.1 概述

传统观念中的商店除必须有商品以外，还需要有店房、门面和货架等。而网上商店主要依靠网络技术来完成商品展示、订购以及支付等工作。网上商店既没有店房、门面也没有货架。因此网上商店常被称之为“虚拟”商店。

人们到超级商场去购买东西时，总得先将想买的商品从货架上取出来，放到购货车中，然后集中一道付款。网上商店模拟这个购物过程，先让客户从不同的网页中选取商品，将它们的订货单集中到“购货车”后一道算帐。网上购货车不同于实际的购货车，它是一种虚拟结构，可称为“虚拟购货车”。

“虚拟购货车”是商务网站的重要组成部分，创建“虚拟购货车”时，要用到动态数据绑定，Global.asa 文件、Application 变量、Session 变量，还需要用手工对 Dreamweaver UltraDev 系统的源代码作一些修改或补充，它是一个很好的有代表性的综合性的系统。初学的读者开始阅读本章内容时可能会遇到一些困难，但如果真正搞清楚了这个过程，对理解源代码的工作原理以及掌握修改源代码的方法将有很大帮助。

25.1.1 系统功能

下面以一个简单的“报刊、杂志预订”为例，说明创建虚拟购货车的方法。本系统将完成以下功能。

- 主页面设计。
- 报刊、杂志分类显示。
- 预订报刊、杂志并将订单放到购货车中。
- 集中算帐。
- 订单修改。
- 批量存储。
- 清除购货车等。

系统使用了 10 张网页，它们的作用和相互关系如图 25.1 所示。

系统的执行过程是：进入主页，从报纸或杂志的目录界面中进行订购；创建购货车并将订单放入其中；检查并调整购货车内的订单并计算价格；确定订单无误后输入客户的姓名、地址等情况；将客户情况与订单一并送至服务器并存入服务器磁盘。

每份订单可能包括多项定货，订单中有一客户编号，为了保证唯一性，客户编号必须自动编号并单独存入磁盘。读入或向磁盘保存客户编号均在 Global.asa 文件中完成。另外 empty.asp 网页用来显示购货车空的状态；clear.asp 网页用于清除购货车内的订单。

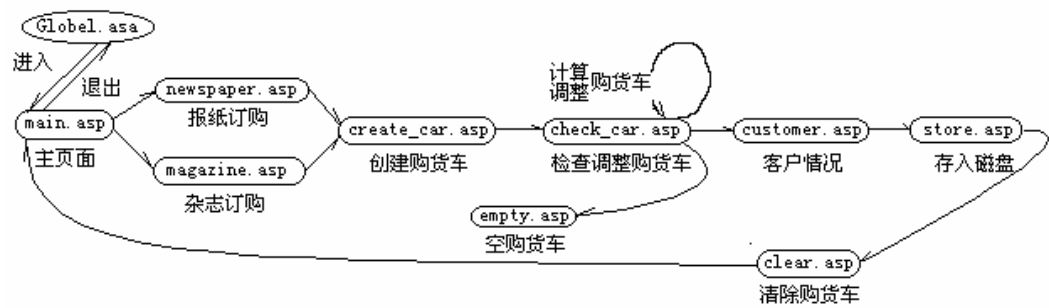


图 25.1 网页间的联系

25.1.2 主要的数据表格

1. 报刊、杂志目录表

假定某省的报刊、杂志发行的目录摘要如表 25.1 所示。

表 25.1 报刊杂志目录表

代号 dh	名称 mc	出版时间 cbsj	单价 dj	月价 yj	季价 jj	简要说明 jysm
41-1	湖南日报	日	0.54	16.5		
41-7	今日女报	周二、四	1.00	8.3		
41-13	文萃报	周二、五	0.50	4.34		
41-25	体坛周报	周二、四	1.5	12.5		
42-3	文艺生活	每月 1、15 日	2.5		15.0	
42-9	湖南教育	每月 1、15 日	3.0		9.0	
42-24	创作	双月	8.0		24.0	
42-31	理论与创作	双月	5.0		15.0	

[说明]：代号中的前三位“41-”代表报纸；“42-”代表杂志。为了简化设计过程，假定报纸只有单价和月价，杂志只有单价和季价。表中的简要说明字段是一指针，用它链接到其他网页以便对报刊作更详细的介绍，其内容这里省略。

2. 客户情况表

客户情况表的字段(m_customer)如表 25.2 所示。

表 25.2 客户情况表的字段

Khbh (客户编号)	Khmc (客户名称)	Dz (地址)	E_mail (E_mail)	Sfzh (身份证号)	Kssj (开始时间)

3. 订单表

订单表(m_order) 的字段如表 25.3 所示。

表 25.3 订单表的字段

Khbh 客户 编号	Dh 商品 代号	Mc 商品 名称	Kssj 开始时间	Dj 单价	Yj 月价	Jj 季价	Fs 份数	Sj 时间	Hj 合计

两表之间是一对多的关系，即一个客户可能有多份订单。客户编号(khbh) 是两张表的同步字段。

25.2 主页面(main.asp)

下面是一个简化的主页面，页面中主要包括“报刊”与“杂志”两个链接指针，分别链接到报刊和杂志的目录网页上，如图 25.2 所示。



图 25.2 简化的主页面

25.3 商品分类显示(newspaper.asp 或 magazine.asp)

25.3.1 分类的方法

将各类商品的主要数据集中放到一张数据表中，有利于管理和保持数据一致。但是如果将各类商品的数据都显示在同一张网页上，将会显得繁杂，不便于客户选择。为了便于顾客选订，应将商品分类显示到不同的网页中。

最好的方法是根据商品的代码来分类，例如湖南省报刊发行代码的前三位是“ 41- ”时代表报纸，前三位是“ 42- ”时代表杂志。

因此用于显示报纸的网页，在创建记录集时，可使用如图 25.3 所示的 SQL 语句。

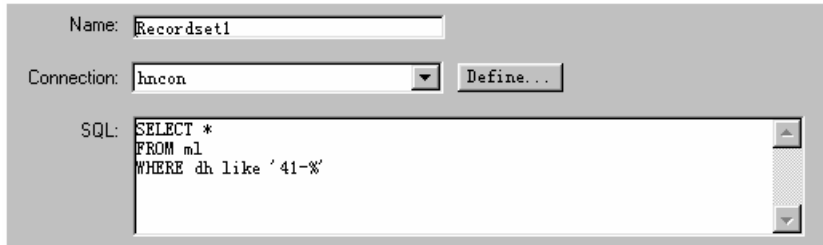


图 25.3 显示报纸的记录集的设置

其中 ml 为报刊目录的数据表名，dh 为商品代号的字段名，%为通配符。执行这段 SQL 语句将只显示以“ 41- ”开头的商品（即在湖南发行的报纸）。

同样，用于显示湖南杂志的网页，在建立数据集时，应使用以下语句：

```
SELECT *
FROM ml
WHERE dh Like '42-%'
```

25.3.2 提供客户选订的界面

为便于客户选订，在表单（Form）内的记录前增加一复选框（CheckBox），用来传递选订商品的信息。

复选框是怎样传递订购商品信息的呢？

复选框有 3 个主要属性：类型（type），名字（name）和输出值（value）。还有一个“选中/未被选中”的标签（Checked/Unchecked）。只有当复选框被选中时才提交输出值。

如果几个同名的复选框都被选中时，将提交什么呢？

多个同名复选框被选中时，将会将各个被选中的复选框的输出值组合成一字符串输出，并在各个复选框的输出值之间用逗号分割。利用这一特点，可以将复选框的输出值（value）与商品代码动态绑定，使得它们的输出值就直接代表被选中的商品代码，从而组成需要的订单或者购货车。这就是创建购货车的基本思想。数据绑定的过程是这样的。

（1）利用服务器行为 Repeat Region 将记录（包括 checkbox）设成多条显示。此时，展开后的复选框将同名。

（2）单击 checkbox 元素属性面板中的  按钮。

（3）单击 value 右边的“闪电”图标 ，在弹出的窗口中将输出值与数据表中的商品代码字段（dh）绑定在一起。

这样，复选框的源代码将出现如图 25.4 所示的情况。

代码 value="<%= (Recordset1.Fields.Item("dh").Value) %>" 表明记录集中的商品代码字段（dh）已经与复选框的输出绑定在一起。

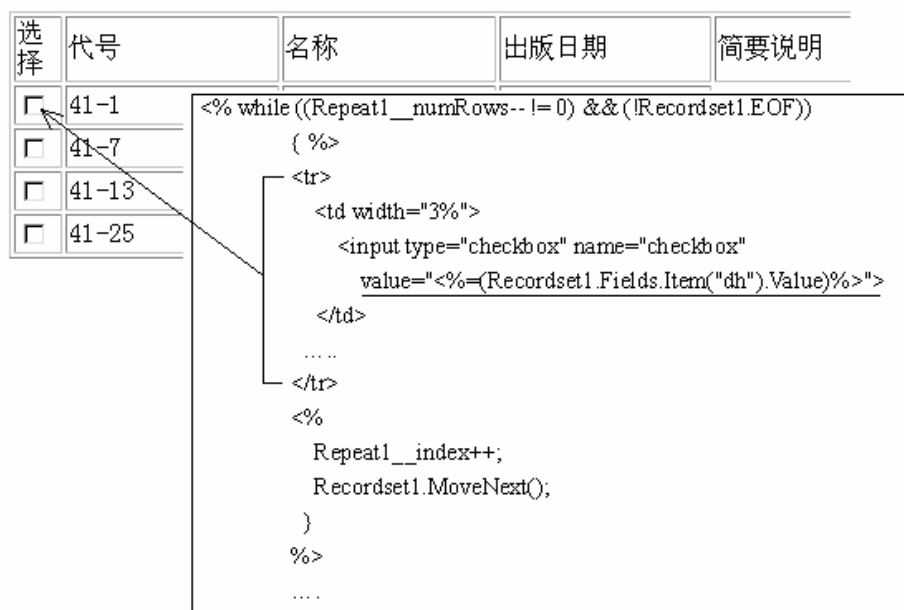


图 25.4 复选框输出值的数据绑定

25.4 创建购货车(create_car.asp)

由于“购货车”是从不同网页上选取的订单的集合，因此必须采用一个能在各网页之间共享的变量来代表。在 Dreamweaver UltraDev 环境中，网页之间能共享的变量有 Application、Cookies 和 Session3 种。

Application 是全局性的变量，能在不同客户之间共享，不适合于构建购货车。Cookies 与 Session 变量都是会话级变量，可在单用户多页面之间共享，因此都可用于创建购货车。下面以 Session 变量为例，说明创建购货车的过程。

25.4.1 Session 变量的初始化

现在用 Session("M_dm")变量代表购货车，在 Global.asa 文件中给它赋初值。在该文件中书写以下代码。

```
<script Language="JavaScript" Runat="Server">
function Session_OnStart()
{
    Session("M_dm") = "";
}
</script>
```

即当客户开始“会话”时，将 Session("M_dm")字符串变量清空。

25.4.2 生成购货车的代码

下面是用 VBScript 编写的一段生成购货车的代码。程序包括 3 个组成部分：

(1) 从复选框的输出中获取商品代码。

(2) 将获取的代码分解为字符串数组。

(3) 将字符串数组逐个与 Session 变量比较，看是否已经包括在 Session 变量中，只有不包括时，才将数组并入 Session 变量中。

这样做的目的是为了保证各个商品代码在 Session 字符串中不能重复。

程序代码如下。

```
<script language = "VBScript" runat = "server">
    pp = Session("M_dm")
    bbh = Request.Form("checkbox") ' 从 checkbox 的输出值中获取商品代码
    m_ddm = split(bbh, ",") ' 分成 m_ddm() 字符串数组
    for i = 0 to UBound(m_ddm)
        if Len(pp) = 0 then ' Session("M_dm")开始为空时，直接放入
            pp = "" & m_ddm(i) & ""
        ElseIf Instr(pp,m_ddm(i))<= 0 then ' 逐个比较代码是否已经包含在 Session()中
            pp = pp & "," & "" & trim(m_ddm(i)) & "" ' 当不包括时才并入。
        ' 其中 trim()的作用是消除字符串前后的空格
    End if
    Next
    Session("M_dm")=pp ' 将并入后的字符串送回 Session()变量
</script>
```

代码中用到了几个内部函数，它们的作用说明如下。

■ `m_ddm = split (bbh, ",")`

`split()`函数的作用是将字符串 `bbh` 以 `,` 为界分解成字符串数组，如：`m_ddm(0)`，`m_ddm(1)` 等。

■ `Instr(pp, m_ddm(i)) <= 0`

语句中 `Instr()`函数的作用是判断字符串数组 `m_ddm(i)`是否已经包含在 `pp` 之中，只有未包括(`<=0`)时，才将字符串并入 `pp` 中。

25.5 检查购货车(check_car.asp)

25.5.1 显示购货车

为了显示购货车，需在网页中完成以下工作。

(1) 在网页的代码前面将 Session 代码引入：

```
<% var ddm=Session("M_dm") %>
```

(2) 在网页的记录集中使用下列 SQL 语句：

```
SELECT *
```

```
FROM ml
WHERE dh in ("+ddm+")
```

这是一个选择显示的语句，最后一句是关键，即：

Where dh in ("+ddm+")代表只显示包含在 Session("M_dm")内的商品代码记录。

25.5.2 计算价格

在购货车中需填写订购的数量（份数）和时间（月或季数），然后由系统自动完成价格的计算工作。为此，对购货车的显示界面须作一些调整。用手工的方法在表单（Form）中增加 3 个字段，前两个（textfield1, textfield2）用于填写预订的份数和时间，第三个（textfield3）用于显示每条记录合计的计算结果。

让 Form 提交数据时仍然转向本页面（即 action 指向本网页），以便在本页面中显示计算结果。这样，进入本页面的情况有两种：

一种情况是第一次进入，此时还没有填写预定的份数和时间，此时新增加的文本框应填入初始值，这里假定份数和时间都是 1，合计中填写该报刊（杂志）的月（季）的单价。

另一种情况是填完预订数后提交时再次进入，此时应该从相应的字段中提取数据进行计算。

为了区别上述两种情况，设置一文本框，这里给它命名为“jsbj”（计算标记的意思），并给它赋值“yes”，作为内部判断之用。当第一次进入页面时，此标签不起作用（此时文本框的值是 undefined），只有填完数据再次进入本网页时，才发挥作用。jsbj 文本框的设置语句是：

```
<Input Type = "hidden" name = "jsbj" value = "yes">
```

这是一个隐藏的文本输入框，并不显示在界面中。另外，为了使得问题简化，下面的程序还作了以下假定：报纸只能分月订阅，杂志只能分季订阅（杂志的月单价为 0）。

1. 购货车中只有单条记录时

为了使问题易于理解，先假定一个简单的情况，如果购货车中只有单条记录。

由于购货车中只有单条记录，因此可以直接从文本框中提取数据进行计算。程序段如下。

```
<%
if (Request("jsbj") == "yes")           // 再次进入本网页时
{
m_fs = parseInt(Request("textfield1")); // 提取预订份数
if (m_fs < 1) { m_fs = 1;}              // parseInt()是转换成整型数的函数
m_sj = parseInt(Request("textfield2")); // 提取预订时间
if (m_sj < 1) { m_sj = 1;}
}
else                                     // 第一次进入网页
{ m_fs = parseInt("1");                 // 置入初始值
  m_sj = parseInt("1");
}
m_jg = parseInt( Recordset1("yj")*100) / 100;
if (m_jg == 0)                          // 如果月价为 0 时，取季价
```

```
{m_jg = parseInt(Recordset1("jj")*100) / 100;}
hj = parseInt(m_fs * m_sj * m_jg * 100) / 100;
// 计算合计：份数*时间*单价
%>
```

程序中语句：parseInt(记录集字段*100) /100 是为了取小数点后两位。这段程序因为要使用记录集，因此必须放在记录集(Recordset1)创建的后面。

2. 购货车中有多条记录时

购货车中的订单记录数取决于客户的订单项，可能是一条，也可能多条。因此，应利用服务器行为的 Repeat Region 选项设置成多条显示。

大家知道，系统是通过循环语句，将记录展开成多条显示的。这样，展开后的记录中将出现了多个 textfield1、textfield2 和 textfield3。文本输入框之间出现了重名现象。

这种情况不是我们希望得到的，它与对前面复选框的要求不同，这里是要求分别计算出各条订单的不同结果，我们并不希望文本框重名。如何解决这一矛盾呢？

(1) 将文本框的名字与商品代码动态绑定

解决的方法就是将各文本框的名字(name)与商品代号动态绑定,并加上不同的前缀。绑定方法与对复选框输出值(value)的绑定相似，如图 25.5 所示。

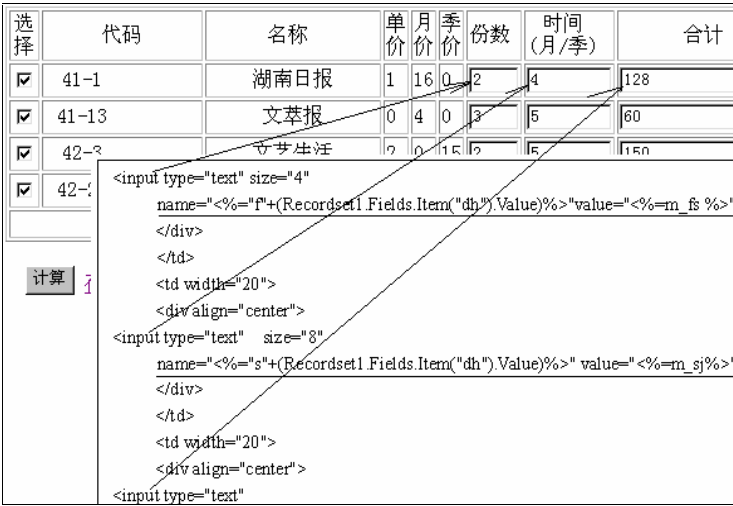


图 25.5 文本输入框名字的动态绑定

为此，先将 3 个文本输入框的名字(name)与商品代码字段动态绑定，然后手工办法将各个文本输入框的名字改写成以下形式。

原 textfield1 的名字(name)改成为：

```
name = "<%=\"f\" + (Recordset1.Fields.Item(\"dh\").Value)%> "
```

原 textfield2 的名字(name)改成为：

```
name = "<%=\"s\" + (Recordset1.Fields.Item(\"dh\").Value)%> "
```

原 textfield3 的名字(name)改成为：

```
name = "<%=\"h\" + (Recordset1.Fields.Item(\"dh\").Value)%> "
```

(2) 文本框输入值的确定。

文本框的值 (value) 则按以下方法取出。

```
<%
sum=0;                                     // 为总和设初值
while ((Repeat1__numRows-- != 0) && (!Recordset1.EOF)) {
    if (Request.Form("jsbj") == "yes")      // 再次进入本网页时
    {
        m_fs = parseInt(Request.Form ("f" + Recordset1("dh"))); // 提取预订份数
        m_sj = parseInt(Request.Form ("s" + Recordset1("dh"))); // 提取预订时间
    }
    else                                     // 第一次进入网页
    { m_fs = parseInt ("1");                // 置入初始值
      m_sj = parseInt ("1"); }
      m_jg = parseInt (Recordset1("yj")*100) / 100;
      if (m_jg == 0)                        // 如果月价为 0 时，取季价
          { m_jg = parseInt (Recordset1("jj")*100) / 100; }
      hj = parseInt (m_fs*m_sj*m_jg*100) / 100;          // 计算合计
      sum = parseInt ((sum+hj)*100) / 100;                // 计算总和
      // 下面增加三个 Session () 变量，以便在 store.asp 网页中使用
      Session("f"+Recordset1("dh")) = m_fs;
      Session("s"+Recordset1("dh")) = m_sj;
      Session("h"+Recordset1("dh")) = hj;
}%>
```

然后将计算结果赋给各个文本输入框。

预定份数的输入框的 value = "<%=m_fs %>";

预定时间的输入框的 value = "<%=m_sj %>";

预定合计的输入框的 value = "<%=hj %>";

(3) 文本输入框的综合代码。

将文本输入框的名字 (name) 和输出值 (value) 合并起来的代码段如下。

```
...
<td width = "34"><div align = "center">
    <input type = "text" size = "4" name = "<%=f"+(Recordset1.Fields.Item("dh").Value)%>" value = "<%=
m_fs %>" > </div>
</td>
<td width="20"><div align="center">
    <input type="text" size="8" name="<%=s"+(Recordset1.Fields.Item("dh").Value)%>"
value="<%=m_sj%>" > </div>
</td>
<td width="20"><div align="center">
```

```



```

此段代码一定要放在循环体中间,只有这样,才能利用每次循环记录集中的商品代码将各文本输入框获得不同的名字。

(4) 汇总的代码。

为了将订单价格的汇总放在购货车的右下角,用手工的方法在表格结束的语句</table>前面加上一段代码。

```

<tr>
<td align = "right" ColSpan = 7> <Font Color = Red> 总价格 = <%= sum %>
</tr>

```

25.5.3 购货车中对订单的修改

在购货车中还应该允许客户作最后的修改,这种修改包括两个方面:一是数字方面的修改(如改变订单的份数或时间);另一种是撤消某些订单。如果想增添订单项时应回到前面的界面去选择。

对数字的修改利用前面的代码已经能够实现。如果想撤消某项订单,只需用鼠标撤消复选框(checkbox)的选择即可。而在代码上应增加重新构建订单(购货车)的部分。在本网页代码前面增添以下代码。

```

<%@LANGUAGE = "JAVASCRIPT"%>
<!--#include file = "Connections/hncon.asp" -->
<script language = "VBScript" runat="server">
if (Request.Form ("jsbj") = "yes") then
    pp = ""
    bbh = Request.Form ("checkbox")
    m_ddm = split (bbh, ",")
    for I = 0 to UBound (m_ddm)
        pp = pp & "" & trim (m_ddm(i)) & "" & ", "
    Next
    Session ("M_dm") = pp
end if
</script>
<% ddm = Session ("M_dm"); %>
<% if (ddm.length == 0)
    {Response.Redirect ("empty.asp");}
%>

```

这段代码与前面构建购货车的代码相似,只有两项微小的区别。

(1)前面构建购货车时,是从多个页面汇集起来的,因此 Session("M_dm")应保持连续性;而这里只需按照本页的 checkbox 的选项即可。

(2) 在这里不会出现选择重复的问题,使用 split()函数将字符串转变成字符串数组的目的只是为了在每个商品代码的前后加上引号。

25.6 清除购货车(clear.asp)

为了清除购货车中的订单,只需在清除网页的前面写上一句 :<%Session("m_dm") = "" %> 即可。当然在网页的界面上写一些提示是值得提倡的。该网页的代码如下。

```
<%@LANGUAGE="JAVASCRIPT"%>
<% Session("M_dm") = "" %>
<html>
<head>
...
</head>
<body bgcolor = "#FFFFFF" text = "#000000">
<h2><i>你已经清除购货车!</i></h2>
<p><a href = "main.asp">返回主页面</a></p>
</body>
</html>
```

25.7 将数据存入磁盘

本系统中需要存入磁盘的数据包括两方面:客户数据与订单数据。

25.7.1 关于客户编号

客户编号是一项重要数据,每份订单的客户必须有一个唯一的客户编号。为此,我们将利用 Application("u_dm")变量作为客户的编号,每当订单存盘前将该变量自动加一后赋给用户及订单。为了适时将 Application("u_dm")从磁盘读出或保存到磁盘中,我们要利用 Global.asa 文件。

在第 14 章中曾经介绍过 ASP 的 Global.asa 文件。该通常包括:Application_OnStart、和 Application_OnEnd 以及 Session_OnStart、Session_OnEnd 等事件。现在我们就利用 Application_OnStart 事件从文件中读取数值并赋给 Application("u_dm")变量;利用 Application_OnEnd 事件将变化后的 Application("u_dm")变量存入磁盘中。

1. Application_OnStart 事件函数

在这个函数中,首先创建一个应用程序作用域变量 Application("u_dm"),然后从站点根目录下的 m_usr.txt 文件中读取上次存入的“客户编号”。

下面要用到 JavaScript 的文件系统的一些内部函数。现在将程序代码以及说明列出如下。

```
function Application_OnStart()
{ Application("u_dm");
```

```

var fs = Server.CreateObject ("Scripting.FileSystemObject");
// 先创建一个 FileSystemObject 系统文件组件的实例，并将它赋值给 fs 变量
var m_usrFilePath = Server.MapPath ("/m_usr.txt");
// 获得 m_usr.txt 文件的物理路径，并赋值给 m_usrFilePath 变量
var ts = fs.OpenTextFile (m_usrFilePath, 1, true);
// 以只读方式打开 m_usr.txt 文件
// 若 m_usr.txt 不存在，则创建一个 m_usr.txt 文件
if (ts.AtEndOfStream != true)
// 如果文件不空，则读出数值赋给 Application ("u_dm")
Application("u_dm") = ts.ReadLine ();
Else
// 否则 Application ("u_dm")赋初值 1000
Application ("u_dm") =1000;
ts.Close();    }    // 关闭文件

```

2 . Application_OnEnd 事件函数

在这里 Application_OnEnd 事件函数的作用是将 Application ("u_dm")存入磁盘的 m_usr.txt 的文件中。然而下面的程序代码是有错误的。先让我们说明代码的含义然后分析错误的所在以及解决的办法。

```

function Application_OnEnd ()
{ var Server.CreateObject ("Scripting.FileSystemObject");
var m_usrFilePath = Server.MapPath ("/m_usr.txt");
var ts = fs.CreateTextFile (m_usrFilePath,true);
// 创建一个 m_usr.txt 文件，如果该文件已经存在，则覆盖它
ts.WriteLine (Application ("u_dm"));
// 将 Application ("u_dm")写入文件
ts.Close ();
}

```

这段代码错在那里呢？

因为 Application 对象结束之前(即 Application_OnEnd 事件被启动时)，除 Application 对象以外，其他 ASP 对象包括 Response、Request、Server、Session 等都已失效。而上面这段程序中仍然使用了 Server 对象，因此是错误的。

如何解决这一问题呢？

解决的方法是在对象失效之前将它们保存在 Application 变量中。以本例的情况看，应该将 FileSystemObject 对象的实例 fs 以及 m_usr.txt 文件的路径 m_usrFilePath 在 Application_OnStart 事件中就保存到 Application 变量中，以便在 Application_OnEnd 事件中使用。相应的语句是：

```

Application ("FileSys") = fs;
Application ("UserFile") = m_usrFilePath;

```

3 . Global.asa 的文件代码

修改后正确的 Global.asa 文件代码如下。

```
<script Language = "JavaScript" runat = "server">
function Application_OnStart ()
{ Application("u_dm");
  var fs = Server.CreateObject ("Scripting.FileSystemObject");
  var m_usrFilePath = Server.MapPath ("/m_usr.txt");
  var ts = fs.OpenTextFile (m_usrFilePath,1,true);
  Application ("FileSys") = fs;
  Application ("UserFile") = m_usrFilePath;
  if (ts.AtEndOfStream != true)
    Application("u_dm") = ts.ReadLine ();
  else
    Application ("u_dm") = 1000;
  ts.Close ();    }

function Application_OnEnd ()
{ var fs = Application ("FileSys");
  var m_usrFilePath = Application ("UserFile");
  var ts = fs.CreateTextFile (m_usrFilePath,true);
  ts.WriteLine(Application ("u_dm"));
  ts.Close ();
  }
}</script>
```

25.7.2 客户情况界面(customer.asp)

当客户最后确定了订单以后，转向客户情况界面，要求输入客户的有关数据。

客户情况网页(customer.asp)中放置一表单(Form)及若干文本输入框以及提交(submit)复位按钮(reset)。界面情况如图 25.6 所示。

按照第 19 章 19.1 增添记录(Insert Record)的要求将客户情况增添到 m_customer 表中。本网页的特殊之处只在于增添了一个客户编号字段(khbh)。该字段是一个只读字段，它的值与 Application("u_dm")动态绑定。而在网页的前面先为 Application("u_dm")作加一处理。其代码如下。

```
<% var numString = Application ("u_dm");
   var num = parseInt (numString,10);
   // 将 numString 转换为整数型
   num = num + 1;
   // 代码加一
   Application.Lock ();
   // 赋值以前先进行锁定
```

```

Application("u_dm") = num;
Application.Unlock ();
// 释放锁定
%>

```

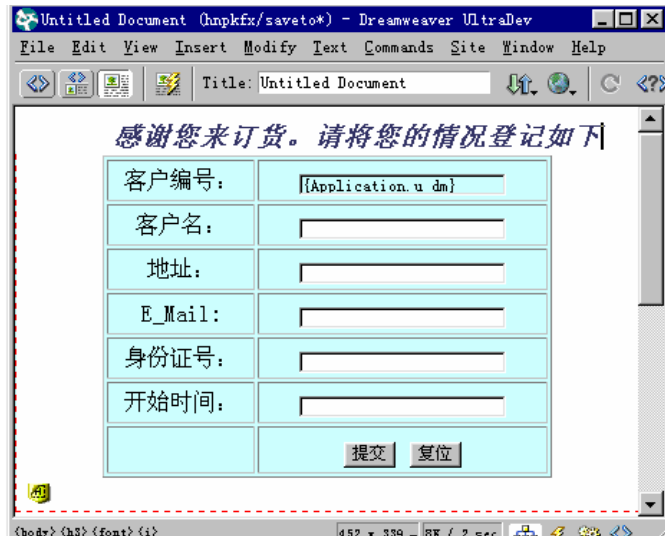


图 25.6 客户情况界面

客户编号(khbh)输入框的代码设置如下：

```
<input type="text" name="khbh" value="<%=Application("u_dm")%>" READONLY>
```

将该字段设成“只读”(READONLY)属性。

25.7.3 将客户及订单存入数据库中(store.asp)

store.asp 网页完成成批存入数据库的工作。在本网页中应完成以下三项设置

1. 设置两个记录集

(1) Recordset1 与 check_car.asp 网页中的记录集相同，可以利用拷贝的方法直接拷贝过来。

(2) Recordset2 指向 m_order 订单表。

2. 将购货车的 Session 变量，引入网页中

引入购货车的代码如下：

```

<script Language="VBScript" RunAt="Server">
ddm=trim(Session("M_dm"))
</script>

```

3. 用 ASP 循环语句将多条订单项存盘

ASP 循环语句如下。

```

<%
Recordset1.MoveFirst();
while (!Recordset1.EOF){

```

```
Recordset2.AddNew();
Recordset2.Fields.Item("khbh").value=Application("u_dm");
// 将 Application("u_dm")变量赋值给表格的第一个字段
Recordset2.Fields.Item("dh").value=Recordset1.Fields.Item("dh").value;
Recordset2.Fields.Item("mc").value=Recordset1.Fields.Item("mc").value;
Recordset2.Fields.Item("dj").value=Recordset1.Fields.Item("dj").value;
Recordset2.Fields.Item("yj").value=Recordset1.Fields.Item("yj").value;
Recordset2.Fields.Item("jj").value=Recordset1.Fields.Item("jj").value;
Recordset2.Fields.Item("fs").value=Session("f"+Recordset1("dh"));
Recordset2.Fields.Item("sj").value=Session("s"+Recordset1("dh"));
Recordset2.Fields.Item("hj").value=Session("h"+Recordset1("dh"));
Recordset2.Update();
Recordset1.MoveNext();
}
%>
```

利用循环中的 AddNew() / Update()将 Recordset1 中各个字段的值赋值给 Recordset2 直到 Recordset1 记录集结束，以达到将全部订单项存盘的目的。

在附录 B 中列出了 create_car.asp、check_car.asp、customer.asp 以及 store.asp4 个主要网页的全部源代码，并扼要地作了一些中文说明。有兴趣的读者可以进一步参考。

附录 A VBScript 简介

A.1 VBScript 语法概要

A.1.1 VBScript 中的数据类型

在 VBScript 中只有一种数据类型，被称为变体(Variant)。变体是一种特殊的数据类型，它能够包含不同种类的信息，在 VBScript 中所有函数的返回值也都是这种数据类型。

变体能够包括所有的基本类型的数据。这些基本类型在 VBScript 中称为子类型(Subtypes)。在大多数情况下，只需将各种子类型的数据直接放到变体中就可以了，不需要事先声明子数据的类型。变体能够自动根据不同的子类型数据分配不同内存空间，并且用最适合的方式来体现特定数据类型的特征和行为。子类型包括以下几种。

- (1) Empty(空类型)：未初始化的 Variant。对于数值变量，值为 0；对于字符串变量，值为零长度的字符串。
- (2) Null(空)：不包括任何有效数据的 Variant。
- (3) Boolean(布尔类型)：这种变量的值只有两种：True(-1) 或者 False(0)。
- (4) Byte(字节)：代表长度只有一个字节的整型数，表示的范围是从 0 到 255。
- (5) Integer(整型数)：代表长度为两个字节的整型数，表示的范围从 -32768 到 32767。
- (6) Currency(货币)：可以表示从 -922337203685477.5808 到 922337203685477.5807 之间的数字。
- (7) Long(长整型)：代表长度为四个字节的整型数，表示的范围从 -2147483648 到 2147483647。
- (8) Single(单精度数)：用来表示浮点运算中的小数，占用四个字节的长度。表示负数时，范围从 -3.402823E38 到 -1.401298E-45。表示正数时，范围在 1.401298E-45 到 3.402823E38 之间。
- (9) Double(双精度数)：用来表示浮点运算中的小数，占用 8 个字节的长度。表示负数是范围从 -1.79769313486232E308 到 -4.94065645841247E-324 之间。表示正数时，范围从 4.94065645841247E-324 到 1.79769313486232E308 之间。
- (10) Date(日期类型)：用来表示日期，其范围从公元 100 年 1 月 1 日到公元 9999 年 12 月 31 日之间。
- (11) String(字符串)：包括长度可变的字符串，最大的长度可达 20 亿个字符。
- (12) Object(对象)：对象是一种特殊的数据类型，可以用来表示对各类对象的引用。
- (13) Error(错误)：保存由 VBScript 生成的错误代码。

例如：

```
<script LANGUAGE = "VBScript">  
<!--
```

```
xx=33          ‘ 将一个数字赋给变量 xx
xx="VBScript"  ‘ 将一个字符串副给变量 xx
--!>
</script>
```

这段程序中使用了一个 xx 的变体类型的变量，使用以前并没有对这个变量进行说明，也没有指明它是那一类数据子类型。程序中先给它赋一整型数，然后又给它赋一字符串，这样做都是允许的。

A.1.2 变体中的数据类型

1. 使用 VarType 函数

尽管在 VBScript 中不需要指定数据类型，但在许多情况下，需要了解变体中究竟包括什么子类型。为了获得这个信息，可以使用 VBScript 的内建函数 VarType()。其语法如下。

VarType(变量名)

它的返回值是一个整数，表示变量的子类型。返回值与子类型的关系如表 A.1 所示。

表 A.1 VarType 返回值与子类型的关系

返回值	常数	变体中的数据类型
0	vbEmpty	未初始化(默认)
1	vbNull	未包含任何有效数据
2	vbInteger	整数子类型
3	vbLong	长整数子类型
4	vbSingle	单精度子类型
5	vbDouble	双精度子类型
6	vbCurrency	货币子类型
7	vbDate	日期子类型
8	vbString	字符串子类型
9	vbObject	对象
10	vbError	错误(Error)子类型
11	vbBoolean	布尔子类型
12	vbVariant	变体(仅仅用于变体数组)
13	vbDataObject	数据进入对象
14	vbDecimal	十进制子类型
17	vbByte	字节子类型
8192	vbArray	数组

2. 使用 TypeName 函数

另一个判断变体中数据类型的有关函数是 TypeName 函数。与 VarType 函数不同的是，TypeName 函数返回值是代表变量数据类型的字符串。如果一个变体类型的变量中包括的数据是整型，则返回值是“ Integer ”。

A.2 变量与数组

数组与变量既有联系又有区别，数组可以被理解为具有特定数目的一组变量，它可以用来连续地地保存多个数据。下面简要介绍 VBScript 中的变量和数组。

A.2.1 声明变量

在 VBScript 中可以用 Dim 语句来声明变量并且分配存储空间。其使用方法如下。

Dim 变量名[[([subscripts])][,变量名[[([subscripts])]]]...

其中，subscripts 仅当要声明一个数组时才需要被指定，它指定了一个数组变量的维数。

在给变量命名的时候，通常在变量名前面加上 3 个字母的前缀来标志在变量中准备使用的数据类型，例如：

```
intMyNumber
strMystring
lngCount
```

A.2.2 变量的作用域

所谓变量作用域是指变量能够在什么样的范围内起作用。变量的作用范围分为脚本级和过程级两种，Dim 语句在脚本级声明的变量对该脚本中所有的过程都可使用；而在过程级中声明的变量仅在该过程内可用。例如：

```
<script LANGUAGE = "VBScript">
<!--
    Sub M_Routine()
        Dim intNumber
        IntNumber = 10
        ...
    End Sub
→
</script>
```

由于 intNumber 变量实在过程中声明的，因此它的作用域只在过程内部。当过程结束之时也就是它生命结束之时。

再例如：

```
<script LANGUAGE = "VBScript">
<!--
```

```

Dim intM_number1
Sub M_Rountin1()
    int_number1 = 20
End Sub
Sub M_number2()
    Dim int_number2
    int_number2 = int_number1 + 10
    ...
End Sub
→
</script>

```

在这个例子中 int_number1 是脚本级作用范围的变量，在脚本内的各个过程中共享。因此在第二个过程中 intM_number2 的值是 30；而 int_number2 变量是在过程内部定义的，因此它的作用域也只在过程内部。

A.2.3 数组操作

在 VBScript 中有 3 种类型的数组。

- 静态数组
- 动态数组
- 多维数组

1. 使用静态数组

定义数组与定义变量差不多。用 Dim 关键字来定义数组时语法为：

```
Dim 数组名(subscripts)[,数组名(subscripts)]...
```

其中，subscripts 是指数组变量的维数，最多可以声明 60 维的数组，subscripts 参数的使用语法是：

```
上界[,上界]...
```

上界指定了一个数组中这一维可用的最大的数组元素编号，而在 VBScript 中，一个数组某一维的下界始终是 0。例如：

```
Dim strWeekday(6)
```

定义了一个名为 strWeekday 的数组，其中包括 7 个元素。最小的下界是 0，最大的下界是 6。

2. 使用动态数组

动态数组就是不能确定数组中包括多少个元素的数组，声明一个动态数组时只需在数组名后面使用一对空括号。如：

```
Dim strArray()
```

在声明了一个动态数组以后，可以在后面的过程中使用 ReDim 语句来定义数组的维数和元素的个数。使用 ReDim 的语法是：

```
ReDim[Preserve]动态数组名(subscripts)[,动态数组名(subscripts)]...
```

语句中的 Preserve 是可选的关键字。如果不使用这一关键字，用 ReDim 定义动态数组时，

旧的数组中的所有数据都会丢失。如果使用这个关键字，旧数组中的数据会自动复制到新数组中。但是如果新数组的元素少于旧数组时，超出的数组元素将会丢失。

由于动态数组中元素的数目是变化的，有时需要知道数组中究竟有多少元素时，可以调用 VBScript 的内建函数 UBound()。这个函数可以返回数组维数的上界。

3. 多维数组简介

多维数组可以存放大量的数据，同时也占用了大量的内存空间。例如：

```
Dim strMArray(9,9,9)
```

定义了一个三维数组，包括 $10 \times 10 \times 10 = 1000$ 个元素。若每个元素占用 15 个字节时，这个数组将占用 15KB 的内存空间。

A.3 VBScript 中的函数

A.3.1 数学函数

Sin(number)：返回 number 的正弦。

Cos(number)：返回 number 的余弦。

Tan(number)：返回 number 的正切。

Atn(number)：返回 number 的反正切。

Abs(number)：返回 number 的绝对值。

Exp(number)：返回 number 的以 e 为底的指数。

Log(number)：返回 number 的自然对数。

Rnd(number)：返回一个随机数。

Sgn(number)：返回 number 的符号。

Sqr(number)：返回 number 的平方根。

A.3.2 格式转换函数

Cbool(expression)：将 expression 转换为布尔子类型。

Cbyte(expression)：将 expression 转换为单字节子类型。

Cdate(expression)：将 expression 转换为日期子类型。

CDbl(expression)：将 expression 转换为双精度子类型。

Chr(expression)：将 expression 转换为一个字符。

CInt(expression)：将 expression 转换为整数子类型。

CLng(expression)：将 expression 转换为长整数子类型。

CSng(expression)：将 expression 转换为单精度子类型。

CStr(expression)：将 expression 转换为字符串子类型。

Hex(expression)：将 expression 转换为 16 进制字符。

Int(expression)：将 expression 取整。

Fix(expression)：将 expression 取整。

Oct(expression)：将 expression 转换为八进制字符。

Val(expression)：将 expression 转换成一个数。

A.3.3 日期和时间函数

Date()：返回系统的当前日期。
DateSerial(year,month,day)：返回日期子类型。
DateValue(String)：将字符型转换为日期型。
Day(Date)：返回给定日期中的天数。
Hour(Time)：返回给定时间的小时。
Minute(Time)：返回给定时间的分钟。
Month(Date)：返回日期的月份。
Now()：返回当前系统的日期和时间。
Second(Time)：返回给定时间的秒数。
Time()：返回当前系统的时间。
TimeSerial(hour,minute,sec)：返回时间子类型。
TimeValue(String)：将字符串转换为时间型。
WeekDay(Date)：返回给定日期的星期几。
Year(Date)：返回给定日期的年份。

A.3.4 字符串操作函数

Asc(string)：返回第一个字符的 ANSI 符号。
Instr(string1,string2)：返回 string2 在 string1 的位置。
Lcase(string)：将 string 转换为小写。
Left(String,num)：返回字符串 string 左边的 num 个字符。
Len(String)：返回字符串的长度。
Ltrim(String)：去掉字符串 String 左边的空格。
Mid(string,start,len)：返回 string 的第 start 个字符开始的 Len 个字符。
Right(String,num)：返回字符串 string 右边的 num 个字符。
Str(num)：将 num 转换为字符串。
Rtrim(string)：去掉字符串 String 右边的空格。
StrComp(string1,string2)：比较两个字符串。
String(Len,char)：创建含有 Len 个字符的字符串。
Trim(string)：去掉字符串 string 前后的空格。
Ucase(String)：将 String 转换为大写。

A.4 流程控制

A.4.1 使用条件语句

VBScript 中可以使用以下两种条件语句：

- If...Then...Else 语句；

- Select Case 语句。

现分别说明如下。

1. 使用 if...Then...Else 条件语句

语句格式如下：

- 第一种格式。

If (条件) Then

...

End If

- 第二种格式。

If (条件) Then

...

Else

...

End If

2. 使用 Select Case 多条件判断语句

语句格式如下：

Select Case 变量表达式

Case 常量 1

...

Case 常量 2

...

Case 常量 3

...

Case 常量 N

...

End Select

执行 Select 语句的过程是，先执行变量表达式，然后将运算结果与 Case 后面的常量逐个进行比较，当结果能够与某个常量相等时，执行该 Case 下面的语句。

A.4.2 使用循环语句

1. 使用 Do...Loop 语句

语句格式如下：

Do

...

Loop While(条件)

语句的执行过程是，先执行 Do 后面的语句，再判断条件，当条件为 True 时退出循环。由于本语句是先执行再判断，因此即使开始时条件已经是 True，也得至少执行一次。

2. 使用 While...Wend 语句

语句的格式如下：

While (条件)

...

Wend

语句的执行过程是，先判断条件，只有条件为 True 时才进入循环，一直到条件变成 False 时才退出循环。

3．使用 For...Next 语句

语句格式如下：

For 循环控制变量 = 初值 To 终值 Step 增量

...

Next

语句的执行过程是，先将循环控制变量赋初值，然后判断是否超过了终值，如果没有超过则进入循环，每当循环一次完成，循环控制变量增加一次增量，然后再与终值进行比较。增量的值可以是正也可以是负。当语句中没有写“ Step 增量 ”时，代表增量的值是 1。

附录 B 创建虚拟购货车(程序清单)

本附录将第 25 章 “创建虚拟购货车” 中的创建购货车网页 (create_car.asp)、计算 (调整) 购货车网页 (check_car.asp)、输入客户情况网页 (customer.asp) 以及存入磁盘网页 (store.asp) 的源代码列出, 供读者进一步参考。其中对源代码所作的补充、修改部分都加了中文注释。

B.1 创建购货车网页(create_car.asp)

```
<%@LANGUAGE="JAVASCRIPT"%>
<!--#include file="Connections/cust.asp" -->
<script language="VBScript" runat="server">
    pp=Session("M_dm")
    bbh=Request.Form("checkbox")      // 从 checkbox 中获取订购的商品代号
    m_ddm=split(bbh,",")
    for i=0 to UBound(m_ddm)        // 创建购货车
        if Len(pp)=0 then
            pp="" & m_ddm(i) & ""
            ElseIf InStr(pp,m_ddm(i))<=0 then
                pp=pp & "," & "" & trim(m_ddm(i)) & ""
        End if
    Next
    Session("M_dm")=pp              // Session("M_dm")代表购货车
</script>
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<body bgcolor="#FFFFFF" text="#000000">
<i><font color="#0066FF">您已经将订单放入购货车中! </font></i>
<a href="check_car.asp">查看购货车</a>      // 三个链接指针
<a href="newspaper.asp">预定报纸</a>
<a href="magazine.asp">预定杂志</a>
<p>&nbsp;</p>
</body>
```

```
</html>
```

B.2 计算(调整) 购货车网页(check_car.asp)

```
<%@LANGUAGE="JAVASCRIPT"%>
<!--#include file="Connections/cust.asp" -->
<% ddm=Session("M_dm"); %>                                // 将购货车 Session("M_dm") 引入 ddm 变量
<% if (ddm.length==0)                                       // 若购货车中没有订单时转向 empty.asp 网页
    {Response.Redirect("empty.asp");}
%>
<script language="VBScript" runat="server">
if (Request.Form("jsbj")="yes") then
    pp=""
    bbh=Request.Form("checkbox")
    m_ddm=split(bbh,",")
    for i=0 to UBound(m_ddm)
        pp=pp&""&trim(m_ddm(i))&""&","
    Next
    Session("M_dm")=pp
end if
</script>
<%
var Recordset1 = Server.CreateObject("ADODB.Recordset");
Recordset1.ActiveConnection = MM_cust_STRING;
Recordset1.Source = "SELECT * FROM ml WHERE dh in ("&ddm&")"; // 显示购货车中的订单
Recordset1.CursorType = 0;
Recordset1.CursorLocation = 2;
Recordset1.LockType = 3;
Recordset1.Open();
var Recordset1_numRows = 0;
%>
<%
var Repeat1__numRows = 10;                                // 为显示多条记录作准备
var Repeat1__index = 0;
Recordset1_numRows += Repeat1__numRows;
%>
<%
// *** Recordset Stats, Move To Record, and Go To Record: declare stats variables
```

```

// set the record count
var Recordset1_total = Recordset1.RecordCount;
// set the number of rows displayed on this page
if (Recordset1_numRows < 0) {           // if repeat region set to all records
    Recordset1_numRows = Recordset1_total;
} else if (Recordset1_numRows == 0) {    // if no repeat regions
    Recordset1_numRows = 1;
}
// set the first and last displayed record
var Recordset1_first = 1;
var Recordset1_last  = Recordset1_first + Recordset1_numRows - 1;
// if we have the correct record count, check the other stats
if (Recordset1_total != -1) {
    Recordset1_numRows = Math.min(Recordset1_numRows, Recordset1_total);
    Recordset1_first   = Math.min(Recordset1_first, Recordset1_total);
    Recordset1_last    = Math.min(Recordset1_last, Recordset1_total);
}
%>
<% var MM_paramName = ""; %>
<%
// *** Move To Record and Go To Record: declare variables
var MM_rs      = Recordset1;
var MM_rsCount = Recordset1_total;
var MM_size    = Recordset1_numRows;
var MM_uniqueCol = "";
    MM_paramName = "";
var MM_offset = 0;
var MM_atTotal = false;
var MM_paramIsDefined = (MM_paramName != "" && String(Request(MM_paramName)) != "undefined");
%>
<%
// *** Move To Record: handle 'index' or 'offset' parameter
if (!MM_paramIsDefined && MM_rsCount != 0) {

    // use index parameter if defined, otherwise use offset parameter
    r = String(Request("index"));
    if (r == "undefined") r = String(Request("offset"));
    if (r && r != "undefined") MM_offset = parseInt(r);
    // if we have a record count, check if we are past the end of the recordset

```

```

if (MM_rsCount != -1) {
    if (MM_offset >= MM_rsCount || MM_offset == -1) { // past end or move last
        if ((MM_rsCount % MM_size) != 0) { // last page not a full repeat region
            MM_offset = MM_rsCount - (MM_rsCount % MM_size);
        } else {
            MM_offset = MM_rsCount - MM_size;
        }
    }
}
}

// move the cursor to the selected record
for (var i=0; !MM_rs.EOF && (i < MM_offset || MM_offset == -1); i++) {
    MM_rs.MoveNext();
}

if (MM_rs.EOF) MM_offset = i; // set MM_offset to the last possible record
}

%>
<%

// *** Move To Record: if we dont know the record count, check the display range
if (MM_rsCount == -1) {
    // walk to the end of the display range for this page
    for (var i=MM_offset; !MM_rs.EOF && (MM_size < 0 || i < MM_offset + MM_size); i++) {
        MM_rs.MoveNext();
    }

    // if we walked off the end of the recordset, set MM_rsCount and MM_size
    if (MM_rs.EOF) {
        MM_rsCount = i;
        if (MM_size < 0 || MM_size > MM_rsCount) MM_size = MM_rsCount;
    }

    // if we walked off the end, set the offset based on page size
    if (MM_rs.EOF && !MM_paramIsDefined) {
        if ((MM_rsCount % MM_size) != 0) { // last page not a full repeat region
            MM_offset = MM_rsCount - (MM_rsCount % MM_size);
        } else {
            MM_offset = MM_rsCount - MM_size;
        }
    }
}

// reset the cursor to the beginning
if (MM_rs.CursorType > 0) {
    if (!MM_rs.BOF) MM_rs.MoveFirst();
}

```

```
    } else {
        MM_rs.Requery();
    }
    // move the cursor to the selected record
    for (var i=0; !MM_rs.EOF && i < MM_offset; i++) {
        MM_rs.MoveNext();
    }
}
%>
<%
// *** Move To Record: update recordset stats
// set the first and last displayed record
Recordset1_first = MM_offset + 1;
Recordset1_last  = MM_offset + MM_size;
if (MM_rsCount != -1) {
    Recordset1_first = Math.min(Recordset1_first, MM_rsCount);
    Recordset1_last  = Math.min(Recordset1_last, MM_rsCount);
}
// set the boolean used by hide region to check if we are on the last record
MM_atTotal = (MM_rsCount != -1 && MM_offset + MM_size >= MM_rsCount);
%>
<%
// *** Go To Record and Move To Record: create strings for maintaining URL and Form parameters

// create the list of parameters which should not be maintained
var MM_removeList = "&index=";
if (MM_paramName != "") MM_removeList += "&" + MM_paramName.toLowerCase() + "=";
var MM_keepURL="",MM_keepForm="",MM_keepBoth="",MM_keepNone="";

// add the URL parameters to the MM_keepURL string
for (var items=new Enumerator(Request.QueryString); !items.atEnd(); items.moveNext()) {
    var nextItem = "&" + items.item().toLowerCase() + "=";
    if (MM_removeList.indexOf(nextItem) == -1) {
        MM_keepURL += "&" + items.item() + "=" + Server.URLEncode(Request.QueryString(items.item()));
    }
}
// add the Form variables to the MM_keepForm string
for (var items=new Enumerator(Request.Form); !items.atEnd(); items.moveNext()) {
    var nextItem = "&" + items.item().toLowerCase() + "=";
```

```

    if (MM_removeList.indexOf(nextItem) == -1) {
        MM_keepForm += "&" + items.item() + "=" + Server.URLEncode(Request.Form(items.item()));
    }
}

// create the Form + URL string and remove the initial '&' from each of the strings
MM_keepBoth = MM_keepURL + MM_keepForm;
if (MM_keepBoth.length > 0) MM_keepBoth = MM_keepBoth.substring(1);
if (MM_keepURL.length > 0) MM_keepURL = MM_keepURL.substring(1);
if (MM_keepForm.length > 0) MM_keepForm = MM_keepForm.substring(1);
%>
<%

// *** Move To Record: set the strings for the first, last, next, and previous links

var MM_moveFirst="",MM_moveLast="",MM_moveNext="",MM_movePrev="";
var MM_keepMove = MM_keepBoth; // keep both Form and URL parameters for moves
var MM_moveParam = "index";

// if the page has a repeated region, remove 'offset' from the maintained parameters
if (MM_size > 1) {
    MM_moveParam = "offset";
    if (MM_keepMove.length > 0) {
        params = MM_keepMove.split("&");
        MM_keepMove = "";
        for (var i=0; i < params.length; i++) {
            var nextItem = params[i].substring(0,params[i].indexOf("="));
            if (nextItem.toLowerCase() != MM_moveParam) {
                MM_keepMove += "&" + params[i];
            }
        }
        if (MM_keepMove.length > 0) MM_keepMove = MM_keepMove.substring(1);
    }
}

// set the strings for the move to links
if (MM_keepMove.length > 0) MM_keepMove += "&";
var urlStr = Request.ServerVariables("URL") + "?" + MM_keepMove + MM_moveParam + "=";
MM_moveFirst = urlStr + "0";
MM_moveLast = urlStr + "-1";
MM_moveNext = urlStr + (MM_offset + MM_size);

```

```

MM_movePrev = urlStr + Math.max(MM_offset - MM_size,0);
%>
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<body bgcolor="#FFFFFF" text="#000000">
<table width="600" border="0" cellpadding="0" cellspacing="0" mm:layoutgroup="true">
  <tr>
    <td width="87" height="37"></td>
    <td width="305" valign="top" height="37">
      <font color="#FF33CC"> </font>
      <div align="center"></div>
      <h2 align="center"><font color="#FF33CC"><b><i>采购清单</i></b></font></h2>
    </td>
    <td width="394" height="37"></td>
  </tr>
  <tr>
    <td height="2" width="87"></td>
    <td width="305" height="2"></td>
    <td width="394" height="2"></td>
  </tr>
  <tr>
    <td height="189" colspan="3" valign="top">
      <form name="form1" method="post" action="shopcart2.asp">
        <table border="1" align="center" width="600">
          <tr>
            <td width="30">
              <div align="center">选择</div>
            </td>
            <td width="128" nowrap>
              <div align="center">代码</div>
            </td>
            <td width="128" nowrap>
              <div align="center">名称</div>
            </td>
            <td width="128">
              <div align="center">单价</div>
            </td>
          </tr>
        </table>
      </form>
    </td>
  </tr>

```

```

<td width="128">
    <div align="center">月价</div>
</td>
<td width="128">
    <div align="center">季价</div>
</td>
<td width="34">
    <div align="center">份数</div>
</td>
<td width="58">
    <div align="center">时间(月/季)</div>
</td>
<td width="134">
    <div align="center">合计</div>
</td>
</tr>
<%
sum=0;                                // 汇总 sum 取初值
while ((Repeat1__numRows-- != 0) && (!Recordset1.EOF)) { // 循环显示多条记录
    if (Request.Form("jsbj")=="yes") // 判断是否第一次进入本网页
    {
        m_fs=parseInt(Request.Form("f"+Recordset1("dh"))); // 提取预定的份数
        m_sj=parseInt(Request.Form("s"+Recordset1("dh"))); // 提取预定的时间
    }
    else
    { m_fs=parseInt("1"); // 是第一次进入本网页, 取初值
      m_sj=parseInt("1"); }
    m_jg=parseInt(Recordset1("yj")*100)/100;
    if (m_jg==0)
    { m_jg=parseInt(Recordset1("jj")*100)/100; }
    hj=parseInt(m_fs*m_sj*m_jg*100)/100; // 计算每条记录的合计
    sum=parseInt((sum+hj)*100)/100; // 计算总和
    Session("f"+Recordset1("dh"))=m_fs; // 送入 Session 变量以便下一个网页使用
    Session("s"+Recordset1("dh"))=m_sj;
    Session("h"+Recordset1("dh"))=hj;
    %>
</tr>
<td width="24">
    <input type="checkbox" name="checkbox"
        value="<%= (Recordset1.Fields.Item("dh").Value) %>" checked>

```

```

// 复选框的输出与商品代码捆绑, 以便重构购货车
</td>
<td width="60" nowrap>
    <div align="center"><%=Recordset1.Fields.Item("dh").Value)%></div>
</td>
<td width="140" nowrap>
    <div align="center"><%=Recordset1.Fields.Item("mc").Value)%></div>
</td>
<td width="8">
    <div align="center"><%=Recordset1.Fields.Item("dj").Value)%></div>
</td>
<td width="8">
    <div align="center"><%=Recordset1.Fields.Item("yj").Value)%></div>
</td>
<td width="8">
    <div align="center"><%=Recordset1.Fields.Item("jj").Value)%></div>
</td>
<td width="34">
    <div align="center">
        <input type="text" size="4"
            name="<%=f"+(Recordset1.Fields.Item("dh").Value)%>" value="<%=m_fs %>" >
        // 文本框的 name 与商品代号捆绑并加上"f"前缀以示区别。
    </div>
</td>
<td width="20">
    <div align="center">
        <input type="text" size="8"
            name="<%=s"+(Recordset1.Fields.Item("dh").Value)%>" value="<%=m_sj %>" >
        // 文本框的 name 与商品代号捆绑并加上"s"前缀以以示区别。
    </div>
</td>
<td width="20">
    <div align="center">
        <input type="text" name="<%=h"+(Recordset1.Fields.Item("dh").Value)%>" value="<%=hj
%>">

        // 文本框的 name 与商品代号捆绑并加上"h"前缀以便分别计算各条记录的合计。
    </div>
</td>
<%

```

```

Repeat1__index++;
Recordset1.MoveNext();
}
%>

        <tr>
            <td align="right" ColSpan=7><Font Color=Red>总价格=<%=sum%>          // 显示总价格
        </td>
        </tr>
    </table>
    <p>&nbsp;</p>
    <input type="submit" name="Submit" value="计算" >
    <font color="#FF0099"><a href="customer.asp">存入订单</a></font> <font color="#FF66CC">
        <a href="clear.asp">清除购货车</a></font>
    <input name="jsbj" value="yes" type="hidden">
    </p>
</form>
</table>
<p>&nbsp;</p>
<p>&nbsp;</p>
</body>
</html>
<%
Recordset1.Close();
%>

```

B.3 输入客户情况网页(customer.asp)

```

<%@LANGUAGE="JAVASCRIPT"%>
<!--#include file="Connections/cust.asp" -->
<%
// *** Edit Operations: declare variables
// set the form action variable
var MM_editAction = Request.ServerVariables("URL");
if (Request.QueryString) {
    MM_editAction += "?" + Request.QueryString;
}
// boolean to abort record edit
var MM_abortEdit = false;
// query string to execute
var MM_editQuery = "";

```

```

%>
<%
// *** Insert Record: set variables
if (String(Request("MM_insert")) != "undefined") {
    var MM_editConnection = MM_cust_STRING;
    var MM_editTable    = "m_customer";
    var MM_editRedirectUrl = "saveto2.asp";
    var
                                MM_fieldsStr
                                =
"khbhlvalue1textfield1value1textfield2value1textfield3value1textfield4value1textfield5value";
    var
                                MM_columnsStr
                                =
"khbhlnone,none,NULLlkuxml',none,'ldzl',none,'lemaill',none,'lsfzhl',none,'lkssjl',none,'";
    // create the MM_fields and MM_columns arrays
    var MM_fields = MM_fieldsStr.split("|");
    var MM_columns = MM_columnsStr.split("|");

    // set the form values
    for (var i=0; i+1 < MM_fields.length; i+=2) {
        MM_fields[i+1] = String(Request.Form(MM_fields[i]));
    }
    // append the query string to the redirect URL
    if (MM_editRedirectUrl && Request.QueryString && Request.QueryString.length > 0) {
        MM_editRedirectUrl += ((MM_editRedirectUrl.indexOf("?") == -1)? "?" : "&") + Request.QueryString;
    }
}
%>
<%
// *** Insert Record: construct a sql insert statement and execute it

if (String(Request("MM_insert")) != "undefined") {

    // create the sql insert statement
    var MM_tableValues = "", MM_dbValues = "";
    for (var i=0; i+1 < MM_fields.length; i+=2) {
        var formVal = MM_fields[i+1];
        var MM_typesArray = MM_columns[i+1].split(",");
        var delim =    (MM_typesArray[0] != "none") ? MM_typesArray[0] : "";
        var altVal =    (MM_typesArray[1] != "none") ? MM_typesArray[1] : "";
        var emptyVal = (MM_typesArray[2] != "none") ? MM_typesArray[2] : "";
        if (formVal == "" || formVal == "undefined") {

```

```

        formVal = emptyVal;
    } else {
        if (altVal != "") {
            formVal = altVal;
        } else if (delim == "") { // escape quotes
            formVal = "" + formVal.replace(/"/g, "") + "";
        } else {
            formVal = delim + formVal + delim;
        }
    }
    MM_tableValues += ((i != 0) ? "," : "") + MM_columns[i];
    MM_dbValues += ((i != 0) ? "," : "") + formVal;
}
MM_editQuery = "insert into " + MM_editTable + " (" + MM_tableValues + ") values (" + MM_dbValues +
    ")";

    // 为客户情况数据表增添记录的 SQL 语句
if (!MM_abortEdit) {
    // execute the insert
    var MM_editCmd = Server.CreateObject('ADODB.Command'); // 创建命令对象
    MM_editCmd.ActiveConnection = MM_editConnection;
    MM_editCmd.CommandText = MM_editQuery; // 执行给客户情况数据表增添记录的
SQL 语句
    MM_editCmd.Execute();
    MM_editCmd.ActiveConnection.Close();

    if (MM_editRedirectUrl) {
        Response.Redirect(MM_editRedirectUrl);
    }
}
%>
<%
var Recordset1 = Server.CreateObject("ADODB.Recordset");
Recordset1.ActiveConnection = MM_cust_STRING;
Recordset1.Source = "SELECT * FROM m_customer";
Recordset1.CursorType = 0;
Recordset1.CursorLocation = 2;
Recordset1.LockType = 3;
Recordset1.Open();

```

```

var Recordset1_numRows = 0;

%>
<% var numString = Application("u_dm");    // 客户编号加 1
    var num = parseInt(numString,10);
    num = num + 1;
    Application.Lock();
    Application("u_dm") = num;
    Application.Unlock();
%>
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<body bgcolor="#FFFFFF" text="#000000">
<h3><font color="#333366"><i>        感谢您来订货。请将您的情况登记如下：</i></font></h3>
<form ACTION="<%=MM_editAction%>" METHOD="POST" name="form1">
    <table width="75%" border="1" align="center">
        <tr>
            <td width="34%">
                <div align="center">客户编号：</div>
            </td>
            <td width="66%">
                <div align="center">
                    <input type="text" name="khbh" value="<%=Application("u_dm")%>"READONLY>
                        // 在输入界面中显示客户编号，此字段为只读型不能修改。
                </div>
            </td>
        </tr>
        <tr>
            <td width="34%">
                <div align="center">客户名：</div>
            </td>
            <td width="66%">
                <div align="center">
                    <input type="text" name="textfield">
                </div>
            </td>
        </tr>
    </table>

```

```
<tr>
  <td width="34%">
    <div align="center">地址 :</div>
  </td>
  <td width="66%">
    <div align="center">
      <input type="text" name="textfield2">
    </div>
  </td>
</tr>
<tr>
  <td width="34%">
    <div align="center">E_Mail:</div>
  </td>
  <td width="66%">
    <div align="center">
      <input type="text" name="textfield3">
    </div>
  </td>
</tr>
<tr>
  <td width="34%">
    <div align="center">身份证号 :</div>
  </td>
  <td width="66%">
    <div align="center">
      <input type="text" name="textfield4">
    </div>
  </td>
</tr>
<tr>
  <td width="34%">
    <div align="center">开始时间 :</div>
  </td>
  <td width="66%">
    <div align="center">
      <input type="text" name="textfield5">
    </div>
  </td>
</tr>
```

```

</tr>
<tr>
  <td width="34%">&nbsp;</td>
  <td width="66%">
    <div align="center">
      <input type="submit" name="Submit" value="提交">
      <input type="reset" name="Submit2" value="复位">
    </div>
  </td>
</tr>
</table>
<input type="hidden" name="MM_insert" value="true"> // 用于判断的文本框，隐藏型
</form>
<h3>&nbsp;</h3>
<h3><font color="#333366"><i>
  <p>&nbsp;</p>
</i></font></h3>
<h3>&nbsp;</h3>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>
</body>
</html>
<%
Recordset1.Close();
%>

```

B.4 存入磁盘网页(store.asp)

```

<%@LANGUAGE="JAVASCRIPT"%>
<!--#include file="Connections/cust.asp" -->
<script Language="VBScript" RunAt="Server">
ddm=trim(Session("M_dm"))
</script>
<%=ddm %>
<%
var Recordset2 = Server.CreateObject("ADODB.Recordset"); // Recordset2 记录集指向订单表

```

```

Recordset2.ActiveConnection = MM_cust_STRING;
Recordset2.Source = "SELECT * FROM m_order";
Recordset2.CursorType = 0;
Recordset2.CursorLocation = 2;
Recordset2.LockType = 3;
Recordset2.Open();
var Recordset2_numRows = 0;
%>
<%
var Recordset1 = Server.CreateObject("ADODB.Recordset");    // Recordset1 记录集指向购货车中的订单
Recordset1.ActiveConnection = MM_cust_STRING;
Recordset1.Source = "SELECT * FROM ml WHERE dh in (" + ddm + ")";    // 只显示已预定的记录
Recordset1.CursorType = 0;
Recordset1.CursorLocation = 2;
Recordset1.LockType = 3;
Recordset1.Open();
var Recordset1_numRows = 0;
%>
<%
Recordset1.MoveFirst();
while (!Recordset1.EOF){                                // 将购货车中的订单逐条成批送往订单表保存
Recordset2.AddNew();
Recordset2.Fields.Item("khbh").value=Application("u_dm"); // 先将客户编号送至“ 客户编号 ” 字段
Recordset2.Fields.Item("dh").value=Recordset1.Fields.Item("dh").value;
Recordset2.Fields.Item("mc").value=Recordset1.Fields.Item("mc").value;
Recordset2.Fields.Item("dj").value=Recordset1.Fields.Item("dj").value;
Recordset2.Fields.Item("yj").value=Recordset1.Fields.Item("yj").value;
Recordset2.Fields.Item("jj").value=Recordset1.Fields.Item("jj").value;
Recordset2.Fields.Item("fs").value=Session("f"+Recordset1("dh"));
Recordset2.Fields.Item("sj").value=Session("s"+Recordset1("dh"));
Recordset2.Fields.Item("hj").value=Session("h"+Recordset1("dh"));
Recordset2.Update();
Recordset1.MoveNext();
}
%>
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">

```

```
</head>
<body bgcolor="#FFFFFF" text="#000000">
<h3>&nbsp;</h3>
<h3>&nbsp;</h3>
<h3><font color="#333366"><i>感谢您来订货。你的订单已经记下。</i></font></h3>
<h3>&nbsp;</h3>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>                                <a href="clear.asp">清除购物车</a></p>
</body>
</html>

<%
Recordset2.Close();
%>

<%
Recordset1.Close();
%>
```