第3章 ASP 组件

前面的章节介绍了怎样使用 ASP的内嵌对象，如Request和 Response对象。ASP组件和这些对象十分近似。但组件和ASP的结合并不十分紧密，只是用来扩展内嵌对象的核心函数。通常用来去实现那些特定的任务。下面列出了集成在 ASP中的一些组件：

- Ad Rotator组件：用来在主页上显示广告图标。可以利用这个组件来限定不同广告图标显示变换的频率。
- Browser Capabilities组件：可以根据不同浏览器的性能来显示不同的主页。例如，可以利用显示单屏或者根据浏览器适应的分屏方式。
- Content Linking组件：可以连接相当数量的主页，这样就可以更容易进行网络导航。例如，可以利用这个组件来显示一部在线教程。
- Counters组件：用来追踪访问该站点访问者的数量，可以利用这个组件在主页中添加点击计数器。
- Permission Checker组件：可以用来检验只有授权用户才能访问的某些连接。可以利用这个组件来创建管理员的维护主页。
- ActiveX Data对象：ActiveX Data对象(ADO)提供从诸如SQL Server这样的数据库系统中存储数据的手段。这种对象相当重要，所以，将会在所有对象全部讨论完毕后单独分出一部分对它进行介绍。

在使用一个组件之前，需要首先创建一个它的实例，可以在任何一个 ASP文件中随意地运用内嵌对象的属性、方法、集合，但是对于组件，必须在特定范围内创建实例来运用它。

3.1 广告轮显组件

现在，Web广告充满了整个网络，如何在网页上建立强大的广告系统以及在有限的空间显示更多的广告呢？Ad Rotator（广告轮显）组件能帮助你建立可以循环显示不同广告的 ASP页面，并容易添加新广告。Ad Rotator组件能帮助你轻松地添加或修改广告地超级连接，这样用户就可以通过单击广告来访问广告客户的 Web 节点。Ad Rotator组件还可以旋转显示广告的图像，并能通过重定向文件跟踪广告的点击次数。Ad Rotator组件根据在 Rotator Schedule 文件中指定的信息显示一个新广告。

3.1.1 所须文件

1. Rotator Schedule 文件

Rotator Schedule 文件是一个文本文件，包含 Ad Rotator 组件用于管理和显示各种广告图像的信息。在该文件中，用户可以指定广告的细节，例如广告的空间大小、使用的图像文件以及

每个文件的显示时间所占百分比。下面是一个 Ad Rotator 组件的Rotator Schedule 文件：

```
------ adrot.txt ------
redirect /asp/redirect.asp
width 180
height 180
border 1
*
asp/image/pic1.jpg
http://www.microsoft.com
广告画片一
2

asp/image/pic1.jpg
http://www.microsoft.com
广告画片二
2

asp/image/pic1.jpg
http://www.microsoft.com
广告画片三
3

asp/image/pic1.jpg
http://www.microsoft.com
广告画片四
3
```

这个文件的前4行为广告的全局设置，第1行指定redirect.asp（重定向文件）的路径，该路径必须是完整的路径（http://www.myserver.com/asp/redirect.asp），或是相对的虚拟路径（/asp/redirect.asp）。第2、3行以像素为单位指定了网页上广告的高度和宽度，默认值是 440、60。第4行以像素为单位指定了网页上广告边框的宽度，默认值是 1，如设为0则没有边框。

星号下面以每4行为一组描述具体的广告细节，第 1行指出广告图像的位置，第 2行指向广告主页的位置。如果广告客户没有主页，请在该行写上一个连字符 (-)，指出该广告没有链接，第3行是在浏览器不支持图形或关闭图像功能的情况下显示的替代文字。第 4行指出广告显示所占的百分比，及显示频率。可将各个广告的显示频率相加，如上面的例子总数是 10，那么第一个广告的频率是2，就是说每调用 Ad Rotator 组件10次，这个广告就显示 2次。这个数字可以是从 0 到 4 294 967 295 的数。

2. 重定向文件

重定向文件是用户创建的文件。它通常包含用来解析由 Ad Rotator 组件发送的查询字符串的脚本并将用户重定向到与用户所单击的广告所相关的 URL。用户也可以将脚本包含进重定向文件中，以便统计单击某一特定广告的用户的数目并将这一信息保存到服务器上的某一文件中。增加计数器和重定向用户是通过下面 ASP 脚本来实现的：

```
< %
```

```
response.redirect(request.querystring("url"))
%>
```

3.1.2 属性

1. Border

该属性可以使用户设置显示广告时，广告的四周是否有边框及边框的大小，初始值在 Rotator Schedule 文件中设定。

2. Clickable

Clickable 属性允许用户设置是否将广告作为超链接显示。

3. TargetFrame

TargetFrame 属性指定链接将被装入的目标框架。该属性完成的功能等价于 HTML 语句中的 TARGET 参数。

3.1.3 方法

Ad Rotator 组件支持的唯一方法是 GetAdvertisement，它的参数就是Rotator Schedule 文件。

Ad Rotator 组件的使用方法是用Server.CreateObject 方法创建 Ad Rotator 组件的对象。 Ad Rotator 组件的 PROGID 属性是 MSWC.AdRotator。如:

```
<%@ LANGUAGE = "VBScript" %>
<% Set Ad = Server.CreateObject("MSWC.Adrotator") %>
<%= Ad.GetAdvertisement("adrot.txt")%>
```

3.2 浏览器兼容组件

在Inernet技术日新月异的今天，并不是所有的浏览器的功能都是一样的。有一些特性，某些浏览器支持而另一些浏览器却不支持，如 : ActiveX 控件、影像流、动态 HTML、 Flash 以及脚本程序等。使用 ASP 的 Browser Capabilities（浏览器兼容） 组件，就能够设计“智能”的 Web 页，以适合浏览器性能的格式呈现内容。

Browser Capabilities 组件创建一个 BrowserType 对象，该对象提供带有客户端网络浏览器的功能说明的用户脚本。

当浏览器连接到网络服务器上时，它自动发送一个 User Agent HTTP 头文件。该头文件是一个声明浏览器及其版本的 ASCII 字符串。此 BrowserType 对象将该头文件和在 Browscap.ini 文件中的项进行比较。

如果找到匹配的项，则该 BrowserType 对象将认为浏览器列表属性与 User Agent 头文件匹配。

若该对象在 browscap.ini 文件中找不到与该头文件匹配的项，那么将使用默认的浏览器属性。若该对象既未找到匹配项且 browscap.ini 文件中也未指定默认的浏览器设置，则它将每个属性都设为字符串 “UNKNOWN”。

可以通过更新 browscap.ini 文件将属性或新的浏览器定义添加到该组件中。

browscap.ini 文件必须与浏览器功能组件 browscap.dll 放在同一目录下，在默认情况下，browscap.ini 文件被存放在 WINDOWS\SYSTEM\INERSRV(如果是 95/98+PWS4) 或 NT\SYSTEM32\INERSRV(如果是 NT) 目录中，你可以自己编辑这个文本文件，以添加自己的属性或者根据最新发布的浏览器版本的更新文件来修改该文件。

用Server.CreateObject 方法可以为Browser Capabilities 组件创建一个BrowserType 对象，我们可以用创建的对象访问与 User Ageng HTTP头文件匹配的浏览器列表属性。如下面的脚本返回当前浏览器是否支持VB Script脚本语言：

```
< %
Set BrowserCap=Server.CreateObject("MSWC.BrowserType")
IF BrowserCap. VBScript =True THEN
Response.Write你的浏览器支持VBScript!"
ELSE
Response.Write对不起，你所使用的浏览器不支持VBScript!"
END IF
%>
```

下面列出了 browscap.ini 中的一部分内容 (不包括注释部分)：

```
[IE 4.0] ;;HTTPUserAgentHeader
browser=IE ;指定该浏览器的名称
Version=4.0 ;指定该浏览的版本号
majorver=4 ;指定主版本号
minorver=0 ;指定副版本号
frames=TRUE ;指定该浏览器是否支持框架
tables=TRUE ;指定该浏览器是否支持表格
cookies=TRUE 指定该浏览器是否支持cookies
backgroundsounds=TRUE指定该浏览器是否支持背景音乐
vbscript=TRUE 指定该浏览器是否支持VBScript
javascript=TRUE 指定该浏览器是否支持JScript
javaapplets=TRUE 指定该浏览器是否支持 Java程序
ActiveXControls=TRUE 指定该浏览器是否支持ActiveX控件
Win16=False ;指定该浏览器是否支持Win16
beta=False ;指定该浏览器是否测试版
cdf=True ;指定该浏览器是否支持用于 Web 预测的 Channel Definition Format

;;ie 4.01
[Mozilla/4.0 (compatible; MSIE 4.01*; Windows 95)]
parent=IE 4.0 父标签允许第二个浏览器继承第一个浏览器的定义
version=4.01
minorver=01
platform=Win98

;;Default Browser ;;

指定默认的浏览器的设置
[Default Browser Capability Settings]
browser=Default
```

```
frames=FALSE
tables=TRUE
cookies=FALSE
backgroundsounds=FALSE
vbscript=FALSE
javascript=FALSE
```

在上面的例子中，父标签允许浏览器继承第一个浏览器的定义，以便 Microsoft Internet Explorer 4.01 定义得以继承 Microsoft Internet Explorer 4.0 定义的全部属性 (例如，frames=TRUE 、 tables=TRUE 以及 cookies=TRUE) 。并通过添加 platform=Win98 行来指定平台，用 version=4.01 重写版本信息。

下面的例子显示用户浏览器的一些功能，在记事本中输入以下代码，保存为 Browser.asp，用HTTP方式进行浏览。

```
< % IF (bc.frames = TRUE) THEN %>
你的浏览器支持框架!< br>
< % ELSE %>
难道现在你还在使用不支持框架的浏览器???< br>
< % END IF %>
< % IF (bc.tables = TRUE) THEN %>
你的浏览器支持表格。< br>
< % ELSE %>
难道现在你还在使用不支持框架的浏览器???< br>
< % END IF %>
< % IF (bc.BackgroundSounds = TRUE) THEN %>
有没有听到美妙的音乐???< br>
< % ELSE %>
可惜，您的浏览器不支持背景音乐。< br>
< % END IF %>
< % IF (bc.vbscript = TRUE) THEN %>
您的浏览器支持Vbscript。< br>
< % ELSE %>
您的浏览器不支持Vbscript。< br>
< % END IF %>
< % if (bc.javascript = TRUE) THEN %>
您的浏览器支持Javascript。< br>
< % ELSE %>
您的浏览器不支持 Javascript。< br>
< % END IF %>
```

3.3 文件操作组件

File Access（文件操作） 组件可用来访问计算机文件系统。用户可以使用 File Access 组件创建 FileSystemObject 对象，该对象提供用于访问文件系统的方法、属性和集合。

用File Access 组件创建 FileSystemObject 对象的方法是Server.CreateObject。

用FileSystemObject 对象可以方便地创建一个文件或打开一个已经有的文件进行修改，包含

以下常用方法。

1. 创建文件

语法：object.CreateTextFile(filename[, overwrite[, unicode]])

参数：

object 必选。应为 FileSystemObject对象的名称。

filename 必选。字符串表达式，指明要创建的文件及其路径。

overwrite 可选。Boolean 值指明是否可以覆盖现有文件。如果可覆盖文件，该值为 True；如果不能覆盖文件，则该值为 False 。如果省略该值，则不能覆盖现有文件。

unicode 可选。Boolean 值指明是否以 Unicode 或 ASCII 文件格式创建文件。如果以 Unicode 文件格式创建文件，则该值为 True；如果以 ASCII 文件格式创建文件，则该值 为 False。如果省略此部分，则假定创建 ASCII 文件。

下面脚本创建ASCII文本文件：

```
<%
Set fso = CreateObject("Scripting.FileSystemObject")
Set fs = fso.CreateTextFile("c:\ InetPub\wwwroot\guest.log")
FOR i=1 to 32
   fs.WriteLine("Hello World!")
NEXT
fs.Close
%>
```

上面脚本用CreateTextFile方法建立文本文件 c:\ InetPub\wwwroot\guest.log，用WriteLine方法输出32行同样的字符串。

2. 打开文件

语法：

```
object.OpenTextFile(filename[, iomode[, create[, format]]])
```

参数：

object 必选。应为 FileSystemObject 对象的名称。

filename 必选。字符串表达式，指明要打开的文件名称及其路径。

iomode 可选。输入/输出模式，是下列两个常数之一：ForReading（以只读的模式打开文件） 或 ForAppending（打开文件并在文件末尾进行写操作）。

create 可选。Boolean 值，指出当指定的 filename 不存在时是否能够创建新文件。允许创建新文件时为 True，否则为 False。默认值为 False。

format 可选。3个 Tristate 值之一，指出以何种格式打开文件。若忽略此参数，则文件以 ASCII 格式打开。

TristateUseDefault 以系统默认格式打开文件。

TristateTrue 以 Unicode 格式打开文件。

TristateFalse 以 ASCII 格式打开文件。

下面脚本打开ASCII文本文件：

```
<%
Set fso = CreateObject("Scripting.FileSystemObject")
Set fs = fso.OpenTextFile("c:\ InetPub\wwwroot\guest.log")
WHILE not fs.AtEndOfStream
   Response.Write(fs.ReadLine)
WEND
fs.Close
%>
```

这个脚本将文本文件guest.txt文件中所有的内容读出来并且显示在浏览器上。如果文件不存在，会显示相应错误信息。

其中WHILE...WEND循环是将文件内容一行一行地循环读取，其中如果没有到达文件末尾，那么AtEndOfStream属性就会是False，直到末尾时变为True。

下面是读取文件时可能会用到的属性：

- AtEndOfLine 判断是否到了文件中一个特定行的末尾，如果不是，则为False，反之为True。
- AtEndOfStream 判断是否到了该文件的结尾并依此返回True和False。上面例子中就用到这个属性。
- Column 判断当前字符在该行的位置，返回一个整数值。
- Line 判断当前行在文件中的行数，返回一个整数值。

一个文件被打开或建立后，就得到一个 TextStream 对象，该对象有一个光标，就好象是在字处理程序中的光标一样，指出接下来要敲入的字符将出现的位置，它同时也指出你要读取的字符的位置。不能通过 CreatObject 来创建一个 TextStream 对象，得到 TextStream 对象的唯一方法是如前所述的用 FileSystemObject 对象打开一个存在的文本文件或者创建一个新的文件。

TextStream 对象用于对已打开或建立的文件进行具体的操作。如下脚本第一句建立一个FileSystemObject对象fso后用fso对象打开guest.log文件，得到一个TextStream对象fs：

```
<%
Set fso = CreateObject("Scripting.FileSystemObject")
Set fs = fso.OpenTextFile("c:\ InetPub\wwwroot\guest.log")
%>
```

下面列出了 TextStream 对象的属性和方法：

TextStream.AtEndOfLine 只读布尔量，当光标在当前行的末尾时，其值为true，反之则为 false。

TextStream.AtEndOfStream 只读布尔量，如果光标在流的末尾时，其值为 true，否则为 false。

TextStream.Column 只读的整数，统计从行首到当前光标位置之间的字符数。

TextStream.Line 只读的整数，指明光标所在行在整个文件中的行号。

TextStream.close() 关闭流以及对应的文本文件。

TextStream.read(Num) 指定从光标的当前位置开始从文本文件中读取一定数目的字符。

TextStream.readall() 将整个流读入一个字符串中。

TextStream.readline() 将一整行的字符读入一个字符串中。

TextStream.write(text) 将一个字符串写入流中。

TextStream.writeline() 将一个文本串写入流中。

TextStream.skip(Num) 在流中，将光标的位置移动一定数目的字符串长度。

TextStream.skiplines() 在流中，将光标移动一定数目的行数。

TextStream.writeblank 将一定数目的空行写入流中。

TextStream.CopyFile source, destination,[OverWrite] 这个方法将文件进行复制，可以使用source参数通配符来在一个时刻进行多个文件的复制。 OverWrite参数将在目的文件已经存在的情况下进行覆盖操作。

TextStream.MoveFile source, destination 这个方法对文件进行移动操作，同样可以使用通配符来移动多个文件，不过如果目的文件已经存在，则会报错而不允许覆盖。

TextStream.DeleteFile FileSpecifier 这个方法删除指定文件，同样还是可以利用通配符来进行多文件的删除。如果没有符合通配符的文件，将会报错。

在使用这些方法之前，首先要创建一个 FileSystemObject对象的实例。下面就是一个完整的使用示例：

```
<%
'创建一个FileSystemObject的事例
Set MyFileObject=Server.CreateObject("Scripting.FileSystemObject")
'创建一个要进行操作的文件
Set MyFile=MyFileObject.CreateTextFile("c:\test.txt")
MyFile.WriteLine("Hello")
MyFile.Close
'复制文件操作
MyFileObject.CopyFile "c:\test.txt" "c:\test2.txt"
'移动文件操作
MyFileObject.MoveFile "c:\test.txt" "c:\test3.txt"
'删除这些文件
MyFileObject.DeleteFile "c:\test.txt"
MyFileObject.DeleteFile "c:\test3.txt"
%>
```

以上，就是ASP的File Access 组件的基本使用方法，可以用File Access 组件打开修改文件，建立各种应用。下面例子就是用 File Access组件实现的简单计数器的例子：

```
<% dim
   visitors
Set fs=CreatObject("Scripting.File SystemObject")
Set Countfile=fs.openTextFile("Number.txt")
Visitors=Countfile.readline
Countfile.Close
Visitors=Visitors+1
Response.Write"<FONTface='Arial' color=#FF0000, size=5>"
Response.Write Visitors&"</FONT>"
Set Countfile=fs.CreatTextfile("Nomber.txt")
Countfile.Writeline(Visitors)
Countfile.close
Set fs=Nothing
%>
```