

第1章 ASP 基础

Microsoft的动态服务器网页(Active Server Pages, ASP)技术目前已发展到了3.0版。对很多人来说,用ASP来创建Windows服务器平台上的动态Web网页、整个站点和基于Web的应用程序已经成为极其自然而然的方法。在浏览器地址栏中,文件扩展名.asp同表示动态创建的网页的文件扩展名.pl或.cgi一样,已被浏览者所接受。事实上,对于在微软的操作系统上工作的Web程序员来说,ASP正在变成一种不再令人激动的技术,而仅是一种工作方式。

这种看法的改变体现了一种技术的逐步成熟。由于ASP的应用程序的数量日益增多,ASP的工具也在日益增加,同时有越来越多的第三方开发商开发出一系列基于ASP或使用ASP的附加功能模块、ActiveX服务器组件,甚至成套的“自己做(do-it-yourself)”Web站点工具。人们几乎不再关注微软公司最初使用的奇特的名字。现在ASP在行业内已经成为一个公认的名词。

这是一本关于ASP成熟技术最新版本的书,其目的是期望读者在一定程度上熟悉ASP的组成,了解ASP能够做什么及如何使用它。本书的目的不仅是使读者开始使用ASP,并且还要使读者了解ASP新版本的变化和它能够做到而早期版本不能做到的事情。

假如你了解以前版本的ASP,可以阅读《Beginning Active Server Pages 2.0》,该书由Brian Francis、John Kauffman、Juan T Llibre、David Sussman和Chris Ullman编写,Wrox出版社出版,书号为ISBN 1861001347。

这并不意味着读者必须成为专家才能阅读本书,但希望读者已了解Web工作的基本方式和脚本语言的相关知识,如VBScript或JavaScript。

本书将更多地从研究和技术的角度讨论ASP,使读者对ASP如何工作有个更好的理解,有助于读者编写的ASP代码达到更高的水平。ASP 3.0核心可能没有很多本质上的改变,但有很多令人激动的使用方法。

本章将主要探讨:

- ASP的简介。
- ASP与IIS如何相互联系。
- 设置问题和管理。
- ASP 3.0 对象模型的概述。
- 对象环境的概念。
- ASP 3.0中的新内容。

本章所采用方法的重要特点是:着重于用ASP页或应用程序的前后关系来研究ASP结构,而不是孤立地研究ASP对象。但是,最初将快速浏览ASP本身使读者能够理解基本概念。

1.1 ASP的起源

本节先简单回顾一下ASP是如何产生的,以及ASP为什么能够在Web程序开发人员中流行。

首先从了解Web的基础和动态Web页面的发展开始。把ASP与其他许多能够提供这种动态化的技术相比较,由此深入了解ASP的发展情况,同时了解Web应用程序的发展,而不仅限于了解单纯的动态Web站点。

1.1.1 HTML的起源

万维网(World Wide Web, WWW)起源于设在瑞士的CERN实验室。Tim Berners-Lee及其开发小组,花费大量的时间,研究建立了一种以一定格式传输信息的方法,这就是众所周知的超文本传输协议(简称为HTTP)。该协议使用了超文本标记语言(HTML)。HTML设计简单,结构灵活,允许在Web浏览器及其他兼容的应用程序中显示文本及图像。文档的某些部分可以成为超链接,即当其被选择时,可以显示不同的页面或同一页面的不同部分。

标记语言是由特定字符分隔的基本元素,指定基本元素中所包含的文本或其他条目如何显示。例如This is some Emphasized text。HTML是一种广泛基于标准通用标记语言(Standard Generalized Markup Language, SGML)的标记语言。SGML是描述语言的一种方法,其本身并不是用来创建网页的语言。

HTML语言从简单开始,不断增加一些附加的功能,直至形成了今天所用的HTML 4.0版本。这些新增的特性提供了更灵活的文本字体风格(例如元素),以及对输出页面布局的更多控制(如窗体和帧的使用)。

早期的网页所缺乏的是动态的内容。刚开始时,这还不是一个问题,因为HTML的开发是为了在计算机、网络及操作系统之间显示和传送信息(特别是技术和科学信息)。这种标准化的文档仅是一种简单的文本和标记,其中的图像和其他非文本的内容以单独文件形式存放,它们可以在各种网络上自由传送。另外,由于信息的格式是固定的,其组成元素的含义也是在HTML中定义好的。对于一个“阅读器”或浏览器应用程序,用任何程序语言在任何平台或操作系统中相对来说是比较容易实现的。

只要建立这些信息网页,文本文件和图像可通过Web服务器应用程序传送给用户,Web服务器应用程序简单地从磁盘中读取它们并且把输出转换成能够在网络中传送的正确HTTP协议。在客户或用户端,浏览器接收传送到信息流,并转换成能够显示的页面。

HTML和HTTP除了具有跨平台特性之外,其最主要的长处就是其文档可以包含有关内容的相关信息、作者意欲表达的方式等。它可以被应用程序阅读而不一定显示出来:例如盲人可以使用特殊的程序把它转换成为语音。同样,其内容可以特殊方式显示出来,以便那些视力差或有其他缺陷的人们更容易进行访问。因此,技术界经常用“用户代理”(user agent)这一通用术语代替“浏览器”(browser)。

1.1.2 动态页面的起源

对于新类型的文档内容,特别是那些设计成为允许在页面上输入信息的文档(例如HTML中<INPUT>等的窗体控件),开发一种能够读取这种信息并加以利用的应用程序成为必然。很明显,传递从其他应用程序中得到的内容,特别是特定类型的数据库的内容,需要一种新的方法。每次都需要重写一个基于文本的页面,对于提供实时性很强的信息当然不是一个理想的方法。

对于Web服务器来说,提供一个接口使其他应用程序能够与之相连成为一种常用的方法。通过这个接口,定制的可执行程序能够接收来自客户端的信息,包括通过点击超链接或在浏览器中键入统一资源定位符(URL)所提出的页面请求的细节。应用程序对客户端的请求能够生成相应的响应,而不是从服务器磁盘上读取文本或标记文件。从这些早期方法开始,逐渐形成一套完整的系列方法动态创建网页,以响应用户的请求或信息的变化。

用于这些应用程序的接口仍然在使用,并被称为公共网关接口(Common Gateway Interface, CGI),这是一种可用任何语言(如C语言)实现的标准。它产生于那些使用cgi-bin目录的应用程序(这里“bin”代表二进制代码,而不是文本)。早期的应用程序都是编译后的程序,通常用C或C++编写。然而这自然要求懂得C语言的编程方法,并且每当对程序做很小的文字上或标记上的变动,都必须重新编译、重新生成可执行文件,这就限制了CGI和动态页面的使用。

取而代之的是开发了一种用脚本语言创建网页的方法,这种语言就是实用摘要和报告语言(Practical Extraction and Reporting Language),或简称为Perl,它允许信息的创建者以一种非常类似于简化版C或C++的语言编写代码。在Perl脚本中,可以“写”文本和标记,用标准输入(stdin)和标准输出(stdout)函数通过CGI与Web服务器通信,输出到浏览器。

Perl在Web上仍然是一种流行的语言,特别是在基于UNIX或Linux的系统上。然而这种语言掌握起来不是很容易,特别是对于那些没有C或C++语言基础的Web开发人员更是如此。现在,一些新的脚本语言出现了,使创建网页更加容易。我们主要看一下它们如何使开发人员的工作变得更加容易。

1. 服务器端脚本技术

对于服务器端的脚本需要用某种类型的中间应用程序,或插件程序来连接。它必须能够接受用户请求,读取并解释合适的基于服务器的脚本文件,接着创建输出页,并传送给Web服务器,在那里作为响应发送给客户端。

在某些情况下,这个任务划分为两个部分:

- 一个应用程序或插件程序处理与Web服务器的往来通信(一般通过CGI)。
- 另一个处理解释和执行脚本。

这就是ASP中的情况,脚本引擎的使用与在其他环境下相同。

Perl是第一个流行的服务器端脚本语言,但是目前已经出现很多其他的语言。在UNIX及基于Linux的系统上,一种新的称之为PHP(Personal Home Page)的语言正变得越来越流行。还有一些语言的目标是一些特定类型的用户,例如TCL就是一种在科学环境下使复杂的数学运算简单化的语言。

2. 微软的动态页面创建技术

微软随Windows NT 3.51推出了他们的Web服务器软件——Internet Information Server(IIS) 1.0。这是一个提供了很多功能的相当标准的软件,它支持CGI。然而微软也提供了另外一个接口,允许使用C和C++等编译语言生成可执行文件,使操作更加高效。这就是Internet服务器应用编程接口(Internet Server Application Programming Interface, ISAPI)。它能够比传统Perl引擎和其他技术所依赖的stdin和stdout更为广泛的对Web服务器的访问。

此后微软和其他第三方开发商推出了许多能通过ISAPI连接到IIS的应用软件,这也就是为什么ASP能够连接到IIS以及其他微软服务器端动态技术。在ASP以前,用得最广泛的是

Internet数据库连接器(Internet Database Connector, IPC)。ASP为Web开发者使用微软的平台开辟了一个新的天地,这使采用数据库中的数据创建动态网页更加容易。特别是它引入了模板(template),即包含了带有能够插入数据库查询结果的文本和标记的模板文件。

现有的(或将过时的)其他微软服务器端页面创建技术有 dbWeb和OLEISAPI。事实上,dbWeb就是OLEISAPI的实现,而对于多数人来说,迄今只是证实了这一技术在实际应用中的不适应性。OLEISAPI是通过特殊版本的ISAPI接口与IIS通信的一些COM对象。Web服务器软件调用COM对象中单个指定函数并以参数为用户的请求提供细节。COM对象返回的是作为字符串的页面的文本和标记,然后作为响应发送给客户端。

OLEISAPI首创了COM对象的动态 Web页面创建,为编程人员提供通过编译的 Active X DLL创建动态页面的能力。然而它所使用的特定的实现和数据通信技术对于较小的任务和内部网(intranet)工作缺乏有效性和可扩展性。同时,每当改变页面的文本和标记时,也需要重新编译动态链接库。图 1-1 给出了到目前为止所讨论的技术的相互关系。

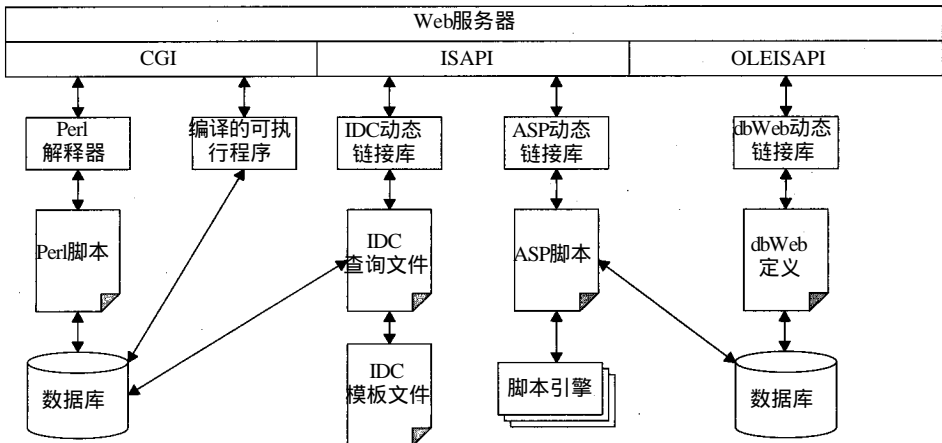


图 1-1 主要页面创建技术的相互关系

3. 动态 Web 页面创建方法的比较

比较动态页面创建技术是一个既困难又敏感的任务,然而理解各种接口和与之相关联的应用程序的差异性又是很重要的。当创建与 Web 服务器交互的应用程序时,涉及到的两个主要的问题是它们对 Web 服务器本身稳定性的影响,以及处理多发或并发页面请求的效率。这两个问题是相互联系的,又是相互排斥的。

应用 CGI 和 ISAPI 的编译的可执行应用程序(不是 DLL 文件),通常在服务器计算机上以进程外(out-of-process)方式运行,这就意味它们要作为单独应用程序运行,占有单独的与 Web 服务器应用程序不同的内存空间。操作系统将其作为一个单独的进程进行管理,禁止它们访问本身所占内存以外的内存。

因此,如果一个进程外应用程序失败,不会影响到 Web 服务器。同样,假如应用程序包含错误,企图直接写入 Web 服务器的内存,会因一般性保护错误停止运行。进程外应用程序也会因用户或操作系统的命令而中止,这时代码会从内存中自动卸载。

因为运行进程外程序意味着访问 Web 服务器内存的请求被禁止,所要求的或产生的输出结果值不能直接传送给 Web 服务器。所以必须执行一种跨进程调用,但这比在同一进程内访问内存要多花费几倍的时间。对于可执行文件的装载和卸载时间还有一定的影响。

相反，应用进程内 (in-process) 程序——通常是能够使用 ISAPI 或 OLEAPI 接口的 DLL (不是单独的可执行文件)，因为其运行在 Web 服务器的内存空间中，可以直接访问 Web 服务器内存中的值，这可提供更快的访问和响应。然而其代码的错误或失败会影响到 Web 服务器。例如 DLL 文件中的代码直接写入包含 Web 服务器操作代码的内存空间，可能引起 Web 服务器的失败。其关系如图 1-2 所示。

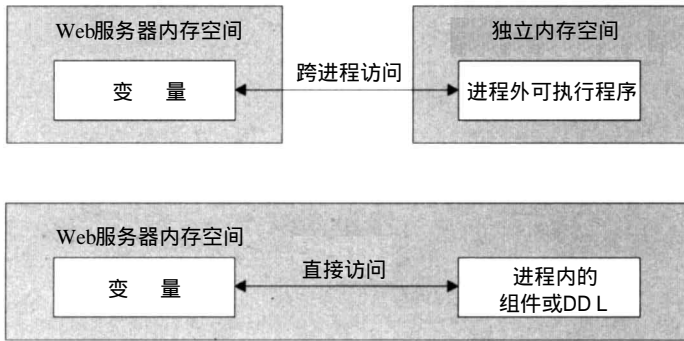


图1-2 不同类型程序占用内存的情况

进程外应用程序实例是 Perl 脚本解释器和使用 ISAPI 接口的 .exe 文件。进程内组件的实例有 dbWeb、IDC 以及 ASP 技术。然而，ASP 和 IIS 相互连接的方式还有许多种。因此，先把其他相关技术放在一边，进一步研究 ASP。

1.2 ASP 如何与 IIS 连接

ASP 本身包含了一个 DLL 文件，名字为 asp.dll，缺省安装在 Winnt\System32\inet_srv 目录下。这个 DLL 文件负责得到一个 ASP 页面 (由文件扩展名 .asp 标识)，然后对它进行分析，寻找服务器端脚本内容。这个脚本传送给相应的脚本引擎，脚本的执行结果与 ASP 页中的 HTML 和模板文本结合在一起。完整的页面会送到 Web 服务器，从那里送往原先提出请求的客户端。

1.2.1 关于应用程序的映射

为了更好地理解这个过程，需要研究一下 Windows 2000 中的应用程序映射的工作方式。对于每一个在 IIS 下设置好的 Web 站点，服务器上都有一个根目录。安装 IIS 时，缺省的 Web 站点通常是 C:\InetPub\WWWRoot，除非在安装过程中改变了路径。对于这个目录以及其中的子目录 (这个我们稍后再讨论)，有一组定义目录如何配合 IIS 的属性。

从 Start 菜单中的 Administrative Tools 打开 Internet Services Manager，将运行微软管理控制台 (Microsoft Management Console, MMC)，显示 IIS 的情况。

用鼠标右击 Default Web Site，选择 Properties (属性)，然后出现 Home Directory 选项卡，如图 1-4 所示。

可以看到缺省的站点被设置成为一个虚拟的应用程序。在选项卡的下半部有 Application name、Execute Permissions 和 Application Protection 选项。IIS 使用虚拟应用程序的方式来隔离页面集和所使用的组件的实例，以失败影响到防止 Web 服务器。正如早先所看到的，这是通过在单独的内存空间中执行页面和进程外组件实现的。我们将在本章后面讨论这个问题。

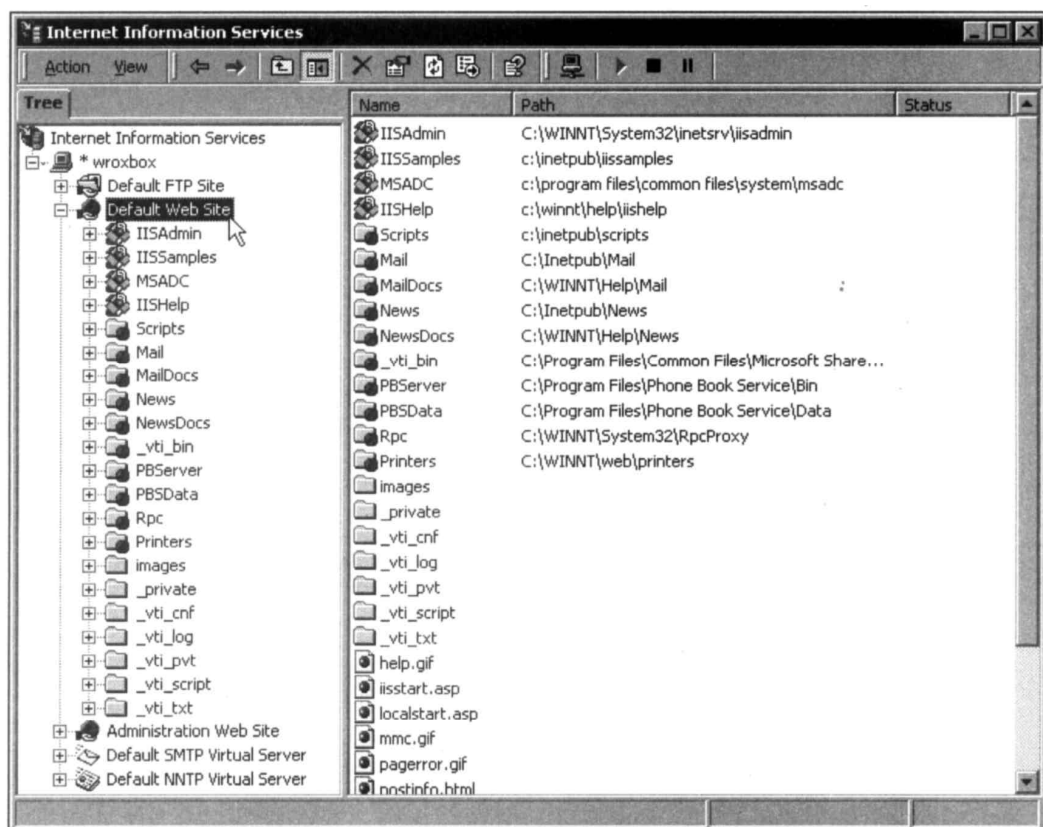


图1-3 显示IIS的情况

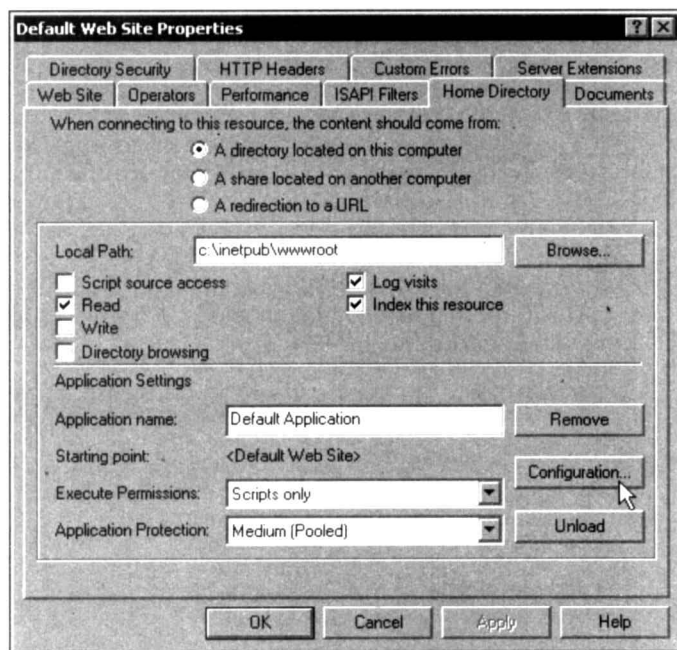


图1-4 Home Directory选项卡

单击Configuration按钮打开Application Configuration对话框,见图1-5。在App Mappings选项卡里,可以看到IIS与各类文件采用特定的DLL相链接。任何含有扩展文件名.asp的网页都送给asp.dll进行处理;有未映射的扩展文件名的页面,如HTML页面的.html和.htm及XML文件的.xml,只需从磁盘上载入并直接发送给客户端。

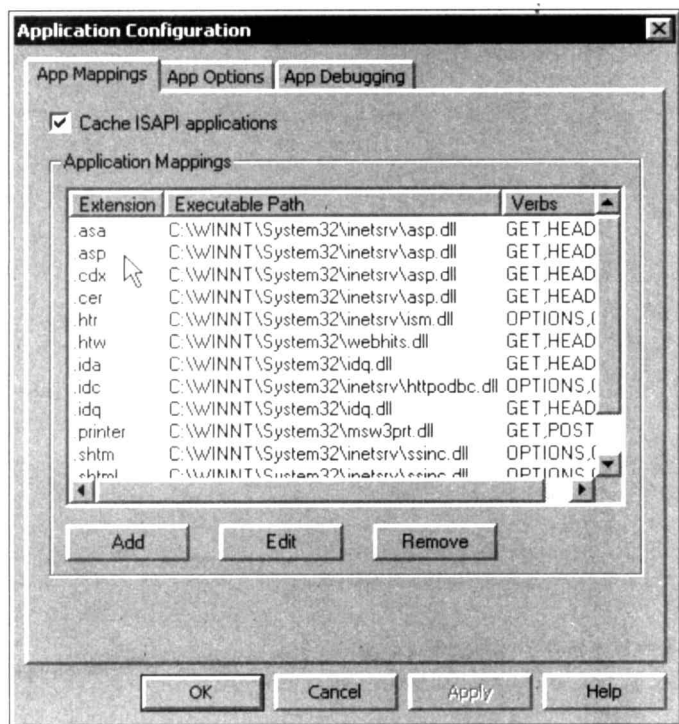


图1-5 应用程序映射情况

读者可能想了解这一选项卡上的其他文件类型。.ida、.idc和.idq文件扩展名是IDC模板文件和查询文件所使用的,因此,一个IDC查询页面(.idc)将直接送到动态链接库httpodbc.dll进行处理。从文件名也可猜出,它使用ODBC执行SQL语句,返回包含在页面中的一组数据记录。同样,.shtm、.shtml和.stm文件扩展名与文件名为ssinc.dll的动态链接库相对应,这些文件类型一般用于请求服务器端包含(server-side include,SSI)处理的文件。我们将在本书的后续部分给出一些相关的实例。

打开Application Configuration和Properties对话框时,读者可能很想了解各个设置的功能。本书的很多地方都要用到这些对话框和设置,查看这些设置有助于增加感性认识,但不要改变这些设置,除非知道为什么这样做。

1.2.2 处理一个ASP文件

我们知道一个ASP页面提供给ASP动态链接库等待解释和执行,那么接着ASP会做些什么呢?

第一步判断是否有ASP服务器端的代码需要执行。假如没有,它仅是简单地通知IIS这种情况,并让IIS向客户端发送页面。事实上,Windows 2000的一个新特性允许对所有页面使用

扩展名.asp，包括对那些非服务器端的脚本代码，而不牺牲任何性能。

在早先版本的IIS和ASP中，所有有.asp扩展名的页面，都会被逐行解释，即使它们含有非ASP服务器端代码。这当然要比IIS直接把它们从磁盘发送给客户端要慢很多。

当ASP从IIS接收到包含有服务器端脚本代码的页面时，它会逐行进行解释。那些非服务器端的脚本，或不需要ASP进行服务器处理的，将被返回给IIS，进而发送给客户端。送来的脚本都会送给相应的脚本引擎，脚本引擎处理后的结果被发送回IIS时，都会插入页面上相应位置上。

为提高操作的效率，ASP也常把脚本引擎创建的编译代码，放在高速缓存中以备再次调用。这个代码与发往客户端的输出结果是不相同的。客户看到的是脚本代码在经过解释、语法检查和编译后的执行结果。在服务器上高速缓存的只是编译后的代码，在原来的源文件变化后，这些代码会被放弃。

1. 辨别服务器端脚本段

ASP的解释器能够根据两种分隔方式之一，分辨出代码是否为服务器端脚本。

(1) 用<% %>脚本分隔符

<%和%>用以分隔脚本段的最常用字符是：

```
<HTML>
<BODY>
This is text and HTML that <B>will not</B> be executed, and is passed to the
client.
<%
REM This is server-side script code that will be interpreted and executed by ASP
%>
</BODY>
</HTML>
```

这个方法也用于在返回页面的其余文本和HTML中需要插入变量值或小段脚本语句的地方：

```
<HTML>
<%
intResult = 7 + 6 - 1
strTheSum = "seven plus six minus one"
%>
<BODY>
The result of calculating <% = strTheSum %> is <% = CStr(intResult) %>
</BODY>
</HTML>
```

这段代码产生如下结果：

The result of calculating seven plus six minus one is 12

(2) 使用<SCRIPT>元素

当编写在浏览器内执行的脚本时，使用<SCRIPT>元素。当在此元素中包含了RUNAT属性并设定其值为“SERVER”时，该元素也可用在服务器端：

```
<HTML>
<BODY>
This is text and HTML that <B>will not</B> be executed, and is passed to the
client.
<SCRIPT RUNAT="SERVER">
```



```
REM This is server-side script code that will be interpreted and executed by ASP
</SCRIPT>
</BODY>
</HTML>
```

认识到 ASP 页面可以包含服务器端脚本和客户端脚本是很重要的。客户端的脚本（包含 `RUNAT="CLIENT"`，或省略 `RUNAT` 属性）不被 ASP 解释器处理，像带有 `.htm` 或 `.html` 文件扩展名的普通 HTML 页一样直接送给客户端：

```
<HTML>
<BODY>
This is text and HTML that <B>will not</B> be executed, and is passed to the
client.
<SCRIPT RUNAT="SERVER">
    REM This is server-side script code that will be interpreted and executed by ASP
</SCRIPT>
<SCRIPT>
<!--
    REM This is client-side script code that will be executed within the client
    REM browser.
-->
</SCRIPT>
</BODY>
</HTML>
```

(3) 包含单独的脚本文件

ASP 页面中能够包含单独的文件，而文件中包含脚本代码，这对编写可用于其他网页的通用函数是非常方便的。用这种方法，改变这个文件中的脚本代码，则包含这个文件的所有脚本在执行时都自动做相应的改变。为了包含单独的脚本文件，可使用 `<SCRIPT>` 元素的 `SRC` 属性，以指定相对的、物理的或虚拟的路径和文件名。

```
<SCRIPT RUNAT="SERVER" SRC="/myscripts/script106.inc"></SCRIPT>
```

这个单独的文件必须仅包含有效的脚本代码，不能包含原有页面内容如文本或 HTML。假如使用这个技术，其他的代码不能放在 `<SCRIPT>` 元素内，它必须是空的。如果要为页面增加其他脚本，需使用另外一个 `<SCRIPT>` 元素或者由 `<%...%>` 分隔的脚本。

也可以包含来自包含脚本、文本或 HTML 的文件的文本，通过使用服务器端包含 (Server-Side Include, SSI) 指令可以实现这一点。在第 4 章中将研究这个问题。

2. 定义脚本语言

ASP 有两个脚本引擎：VBScript 和 JScript。安装 ASP 时这两个脚本引擎已缺省安装。也有由其他应用程序使用的脚本引擎，如微软的 Internet Explorer Web 浏览器和 Windows Scripting Host。在 Windows 2000 中该浏览器的目前版本是 5.0，可能还有新的升级版本。还有其他的脚本引擎，如 TCL 和 PerlScript（一种 ActiveX 脚本解释器而非传统的基于 CGI 的 Perl）。

因此必须告诉 ASP，ASP 页用什么引擎。通常使用的方法是用特定的环境声明元素中定义引擎，这必须放在文件的第一行，并只能定义一次。这个元素一般用在 ASP 代码分隔符后面跟着字符 `@` 来表示：

```
<%@LANGUAGE="language_name"%>
```

这个定义行还可包含其他的定义内容，在本章后面能看到相关内容。定义一个用 VBScript 编码的页面，如下所示：

```
<%@LANGUAGE="VBScript"%>
```

对于JScript引擎,使用:

```
<%@LANGUAGE="JScript"%>
```

经过上述定义,在<%...%>段内的页面的所有代码将被送至定义元素所定义的脚本引擎。对于这种代码分隔方式,这是指定脚本语言的唯一方法。

然而,使用<SCRIPT>定义元素,可以单独定义每一段的脚本语言,如果需要的话在同一页面上可使用不止一种脚本语言:

```
<%@LANGUAGE="VBScript"%>
<HTML>
<BODY>
This is text and HTML that <B>will not</B> be executed, and is passed to the
client.

<SCRIPT RUNAT="SERVER" LANGUAGE="VBScript">
  REM This is server-side VBScript code
</SCRIPT>

<SCRIPT RUNAT="SERVER" LANGUAGE="JScript">
  // This is server-side JScript code
</SCRIPT>

<!-- the next two script sections define script code stored in separate -->
<!-- files, which is included into the script element by ASP at runtime -->
<SCRIPT RUNAT="SERVER" LANGUAGE="VBScript" SRC="/myscripts/scr106.inc"></SCRIPT>
<SCRIPT RUNAT="SERVER" LANGUAGE="JScript" SRC="/myscripts/scr106.inc"></SCRIPT>

<%
  REM This is server-side VBScript code because the default language for the page
  REM is set as VBScript in the declaration element at the start of the page.
%>
</BODY>
</HTML>
```

不像在客户端,在注释元素内不需要隐藏脚本代码,因为当它执行时,代码将从页面移走,取而代之的仅是输出结果。在客户端查看ASP页(在浏览器中选择View Source)时,只能看到文本、HTML和其他客户端脚本代码。所有的服务器脚本都被执行,只有结果被送到客户端。

缺省的脚本语言

当没有指定ASP页的脚本语言或没有单独的<SCRIPT>元素时,ASP将使用缺省的脚本引擎。首次安装IIS时,缺省的脚本语言是VBScript。但对于整个Web站点或者一个站点内的独立的虚拟应用程序,根据需要可相应改变设置。

在本章前面提到的Application Configuration对话框中,有一个App Options选项卡包含了设置缺省语言的文本框,如图1-6所示。

缺省语言也可在IIS里通过编辑Active Directory段的值来改变设置。在Internet Services Manager MMC插件中所能见到的设置,都存放在Windows 2000中的Active Directory中,只要有相应的权限就可以进行读取和编辑。本书将在后面章节详细介绍Active Directory。

除此以外,可根据个人喜好选择脚本语言。假如读者认为一种语言很好,可一直坚持使用它。因为所有的ASP代码都在服务器端执行而不用担心浏览器的兼容性问题。假如读者精通VBScript和JScript或者其他语言,可根据需要选择最合适的一种。然而因为VBScript各个版本的功能不断扩充(例如VBScript 5.0现在支持正规表达式),通常会选择VBScript。其对类型和语法的要求不是很高,并且有更简单的多语句结构,因此是一种容易使用的工具。

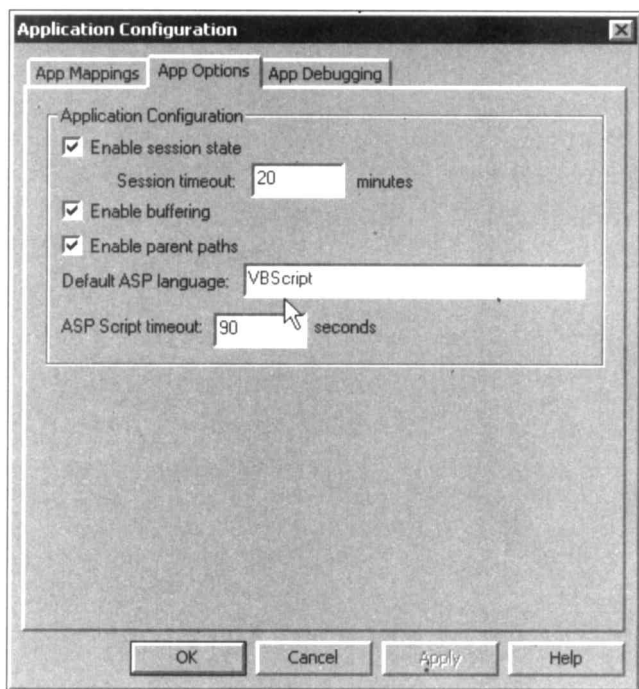


图1-6 设置缺省语言

3. 脚本性能问题

一般来说，Web服务器处理器的速度是足够满足使用的（除非特别繁忙的站点），因为它们的主要任务是从磁盘中载入页面并发往客户端。因此，每个页面的请求结果都使处理器等待磁盘。这意味着执行ASP脚本通常对性能的影响非常小。而且如果在一个页面上某段脚本代码多次执行，而这段代码的已编译版本已被高速缓存，那么只须执行它，而不必多次编译，这样对性能的影响就更小了。

当然，随着请求数量的增加，服务器负载也不断增加，解析和执行每个ASP页面就有相应的代价。应尽可能压缩ASP解释器的工作量。下面是一些有用的提示。

(1) 避免在同一页面上混用脚本语言

如果同一页面上有几种脚本语言，ASP不得不一个接一个地加载多种脚本引擎，并把相应的代码送给相应的引擎。这将降低处理速度，增加内存使用量。另外一个副作用是，假如编写的是一个顺序执行的代码（而不是一系列从其他代码段调用的函数或子程序），可能会以与它们在页面中出现的顺序不同的顺序执行。

例如，下面的代码可能不会产生所希望的结果，因为无法确保JScript代码的结果在网页中是首先出现，或是在第三位出现。

```
<%@LANGUAGE="JScript"%>
<HTML>
<BODY>
<SCRIPT RUNAT="SERVER" LANGUAGE="JScript">
    Response.Write('First<BR>')
</SCRIPT>
<SCRIPT RUNAT="SERVER" LANGUAGE="VBScript">
    Response.Write "Second<BR>"
```

```
</SCRIPT>
<%
    Response.Write('Third<BR>') // JScript is the default in this page
%>
</BODY>
</HTML>
```

(2) 在脚本和其他内容中避免过多的环境切换

每当ASP遇到一个脚本段，必须执行并把结果发到 IIS，然后再次返回去解释页面。因此，使用Response.Write语句(只创建发往客户端的文本，类似于Print命令行)能使页面的效率更高。例如下面这段VBScript：

```
<BODY>
<%
    intResult = 7 + 6 - 1
    strTheSum = "seven plus six minus one"
    Response.Write "The result of calculating " & strTheSum & " is " & _
        CStr(intResult)
%>
</BODY>
```

与下段相比，效率更高：

```
<%
    intResult = 7 + 6 - 1
    strTheSum = "seven plus six minus one"
%>
<BODY>
The result of calculating <% = strTheSum %> is <% = CStr(intResult) %>
</BODY>
```

(3) 构建单独的组件

假如在一个页面不得不做大量的运算，或者运行一个过于复杂的脚本，通常的好办法是构建一个组件，并安装在Web服务器上。组件通常是编译过的可执行代码，相对于解释ASP脚本代码，使用的效率更高。本书后面将探讨构建组件的问题。

1.3 相关设置问题和管理

ASP是随着IIS 5.0自动安装的，设置程序为用户提供了大多数配置决定。根据计算机的主IP地址，自动设置一个缺省的Web站点并绑定在该地址上。这意味着可使用机器名(在局域网上)或者使用计算机的URL访问Web站点：

http://sunspot	<-通过局域网访问缺省站点
http://sunspot.stonebroom.com	<-全局访问缺省站点

请记住，IIS 5.0不仅仅是一个通过HTTP提供WWW服务的Web服务器，还能够提供服务以支持FTP(File Transfer Protocol, 文件传输协议)和SMTP(Simple Mail Transfer Protocol, 简单邮件传输协议)，并提供一个RADIUS服务以允许远程用户验证，加上内建的安全性及用户权限管理等特性。

1.3.1 IIS的安装

安装Windows 2000服务器时，缺省状态是不安装IIS的，因为不是所有的服务器都用作Web服务器。然而，在Windows 2000安装过程完成后，Windows 2000 Configure Server页将在

IE中打开，这是安装IIS及相关的软件和服务的地方。

假如已经安装了不带 IIS 的 Windows 2000，可以在 Start 菜单处，选择 Programs|Administrative Tools|Configure Server，打开这个页面。

在图 1-7 中，选择 Advance 选项，点击 Optional Components。在右侧的页面，点击 Start the Windows Components wizard，将打开显示一系列可供安装的组件的对话框（类似 NT 4 中的 Option Pack）。

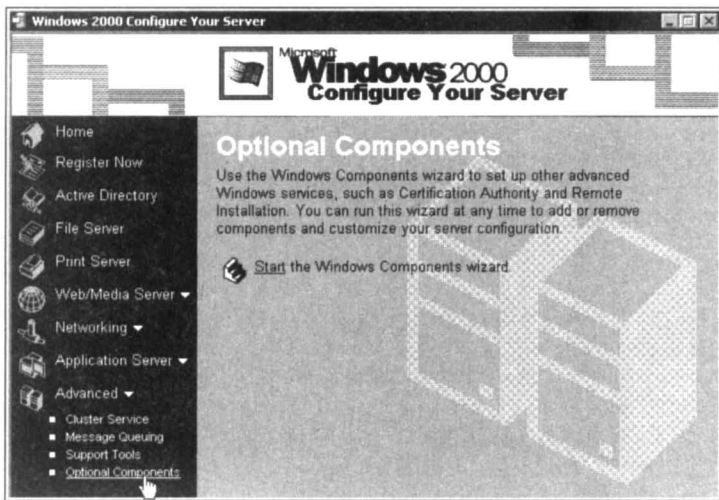


图 1-7 在 Windows 2000 安装 IIS 的屏幕 1

也可以在控制面板中使用 Add/Remove Programs 打开这个对话框。

在图 1-8 中，选择 Internet Information Services (IIS) 选项，点击 Details 按钮。

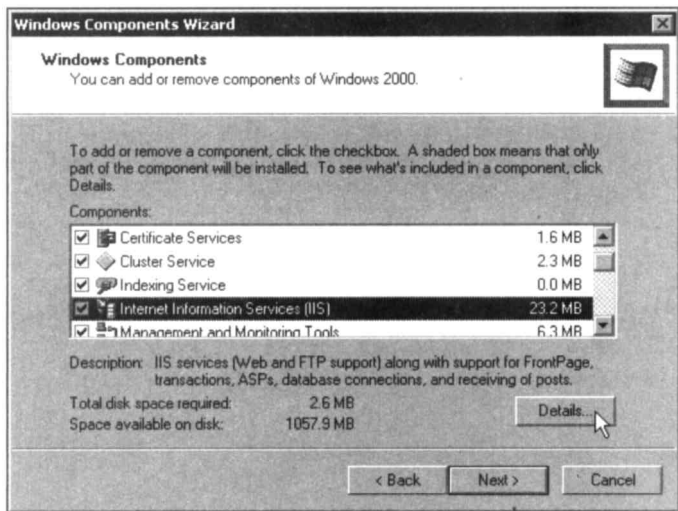


图 1-8 在 Windows 2000 安装 IIS 的屏幕 2

出现的窗口中列出了 IIS 5.0 的一些子组件，这些子组件多数已被缺省选中，包括 FTP 和

WWW(World Wide Web Server)服务, 如图 1-9所示。

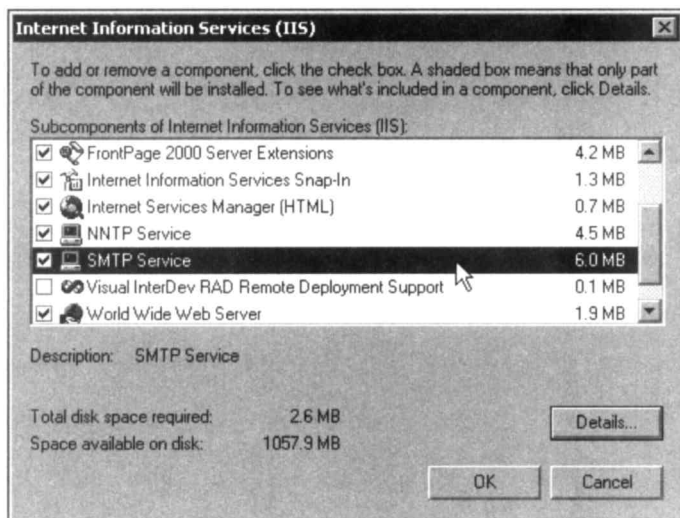


图1-9 在Windows 2000安装IIS的屏幕3

在学习 ASP时, 值得把 IIS全部(或大多数)子组件安装在计算机上, 这样当研究 ASP和使用Windows其他服务的应用程序时, 能够获得这些子组件的所有文档。

完成设置后, 可通过 Services对话框 (Start|Programs| Administrative Tools|Services)关闭不需要的子组件服务, 以减少服务器的负载。

假如想通过 IIS提供邮件服务, 必须选择 SMTP Service选项。在本书后面将看到, 这将在创建 ASP邮件应用程序时所需要的各种文件。根据需要也可安装网络新闻传输协议 (Network News Transfer Protocol, NNTP)服务, 提供“新闻组”功能。

确保选择 Internet Information Service Manager Snap-in选项, 这可通过 Start菜单中的 Internet Services Manager来从 MMC管理 Web服务器。如果想使用 Visual InterDev或FrontPage 访问在服务器上的网页, 可以安装 FrontPage 2000 Server Extensions。

在设置过程中需要提供给 IIS的唯一信息是缺省的 Web和FTP站点路径。设置程序建议用户采用 \InetPub\WWW Root和\InetPub\FTPRoot。如果你有多个驱动器, 你可能只想改变驱动器。也可以把它们放在与包含 Windows系统文件的驱动器不同的另外一个物理驱动器上, 以提高对文件的访问速度。

其他有用的 Windows组件

回到主 Windows Component Wizard对话框, 如图 1-10所示, 可选择其他想安装的 Windows服务。在本书中, 我们将要用到 Message Queuing Services(MSMQ)和Microsoft Indexing Service(NT 4 Option Pack中的Index Server)。可以马上安装它们, 对这两个均选择缺省选项安装。可以在这个向导中安装的一个有用工具是 Microsoft Script Debugger。我们将在第7章中介绍这个工具, 你可以看到该工具使调试页面非常方便。然而确保不要在一个公用服务器或“生产”服务器上安装 Script Debugger, 而仅安装在试验或开发用的服务器上。

安装 IIS以后, 可以直接使用它们, 安装的缺省页面指出了此站点正在建设中。还有一个页面描述了 IIS的功能和用途, 并且有与各个管理程序的链接。这个页面在 <http://Server>

_name_or_url/localstart.asp中，只有在站点根目录下没有 Default.asp或Default.htm页时才加载。



图1-10 在Windows 2000安装IIS的主屏幕

需要记住，访问ASP页使用的是HTTP协议。假如想在Explorer中查看Web目录的内容，即使是在作为Web服务器的同一个机器上或通过一个局域网，也不能通过双击它们来加载，必须在浏览器的地址栏中键入机器的URL(以http://开始)。

1.3.2 IIS管理工具

安装的各种服务后，Windows Components Wizard允许安装用来管理IIS的工具。其中一个已经提到过的Internet Services Manager(ISM)，这是Microsoft Management Console(MMC)的一个插件。还有一组HTML页面能够用来管理IIS，它们是Windows 2000的可选子组件。

这些页面和ISM都能提供远程管理功能，差别在于ISM必须安装在远程计算机上，而HTML管理页面只要求远程计算机安装浏览器(最好安装IE 4.0或更高版本)。

ISM的HTML版本

HTML管理网页为远程管理IIS提供真正简便的方式，而且更快、更有效。在服务器上，可以通过选择Start菜单中的Internet Services Manager(HTML)来打开它。图1-11为缺省Web站点的内容。

注意，这个页面的URL包含了一个端口号，这个端口号是6369。安装程序产生一个介于1000~9999之间的随机端口号，并将之分配给安装HTML管理页面时所创建的管理Web站点(Administration Web Site)。必须在URL中指定这个端口号，它被自动加到Start菜单项中。这个初步的安全措施防止不知道端口号的人员的访问。

从远程的计算机上访问HTML管理页面，必须知道端口号。这可以从Administration Web Site的Properties对话框中得到。这个TCP的端口号显示在这个对话框的Web Site选项卡上，如图1-12所示。

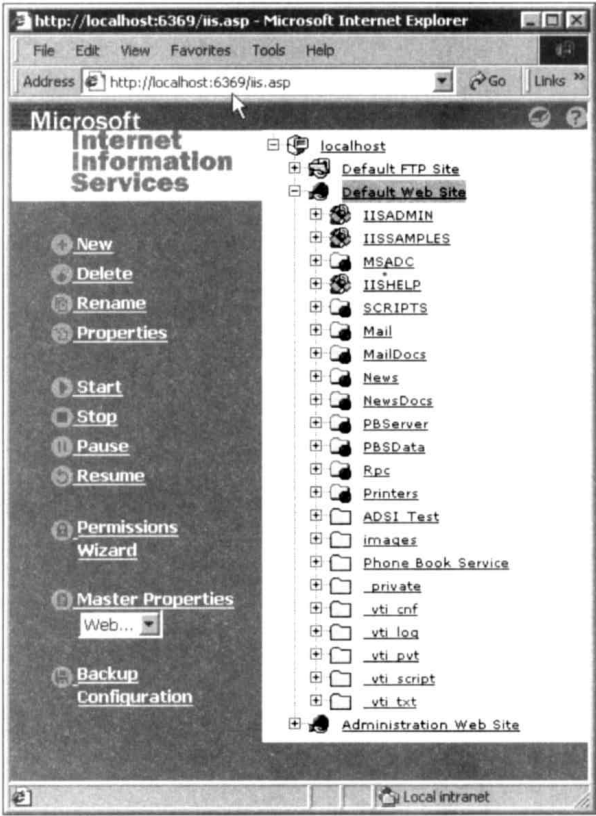


图1-11 缺省Web站点的内容

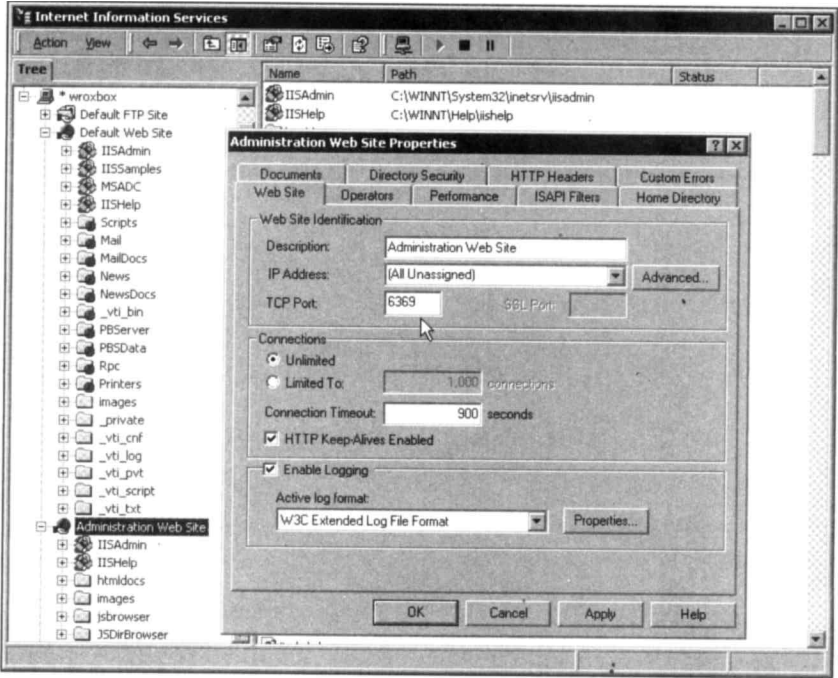


图1-12 Administration Web Site Properties对话框

然而，那还不是全部。缺省时，只有安装在 Web 服务器上的浏览器能够调出这些页面，这是因为也设置了 IP 限制。在 Administration Web Site，选择 IISADMIN 虚拟应用程序，打开这个程序的 Properties 对话框，然后在 Directory Security(目录安全)选项卡上，在“IP address and domain name restrictions”(IP 地址和域名限制)框中选择 Edit 按钮，如图 1-13 所示。

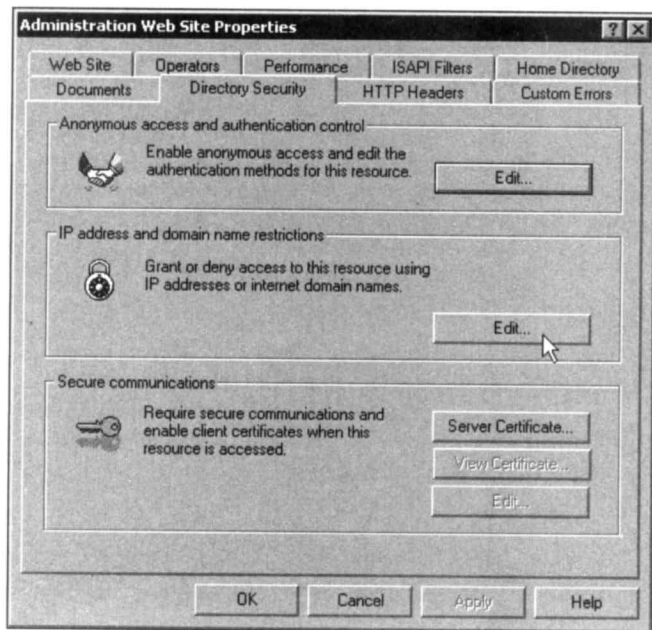


图1-13 Directory Security选项卡

这时打开一个对话框，可以看到这个限制，见图 1-14。只有 IP 地址为 127.0.0.1(本地服务器)才能访问这个页面，即使从其他 IP 地址传来的请求包含正确的端口号，也不能访问。

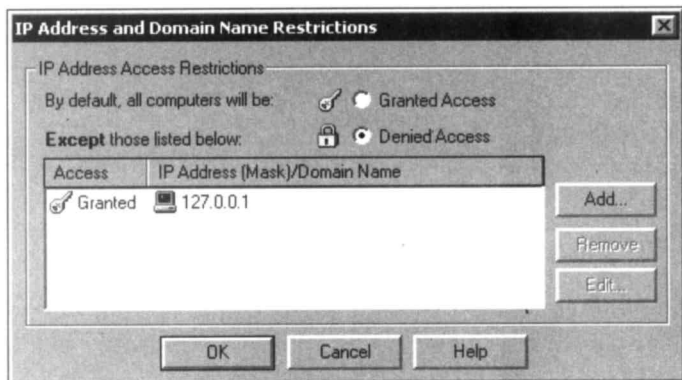


图1-14 显示IP限制的屏幕

可根据需要改变这个设置，既可删除这个限制(如果与 Internet 相连，则这是一个危险的方法)，也可在列表中增加自己的 IP 地址。该地址可以是局域网上远程计算机的 IP 地址，或者是代理服务器或 ISP 的 IP 地址(如果通过网络连接)。然而，为了安全，应该通过证书为此应用程序建立一个安全的目录，并通过 https(http secure)访问而不是通过 http 访问，即使用 SSL 或类似

的方法。我们在后面讨论这些主题内容。

1.3.3 常见的管理任务

IIS的缺省配置对于大多数的应用是合适的。在开始建立站点时，当然可以改变这种配置，使ASP与IIS以不同方式工作。我们已经看到过一些缺省 Web站点的Properties对话框。假如你习惯于在IIS 4.0上使用ASP 2.0，就会发现多数的设置是相当熟悉的。许多新设置选项的意义可以通过控件标签理解。

在Properties对话框中最可能需要改变的设置，显示在图 1-15中。多数情况下，对于一个完整的Web站点，可通过这个站点的 Properties对话框进行这些设置。而对站点内的各个目录进行设置时，要打开目录的 Properties对话框。

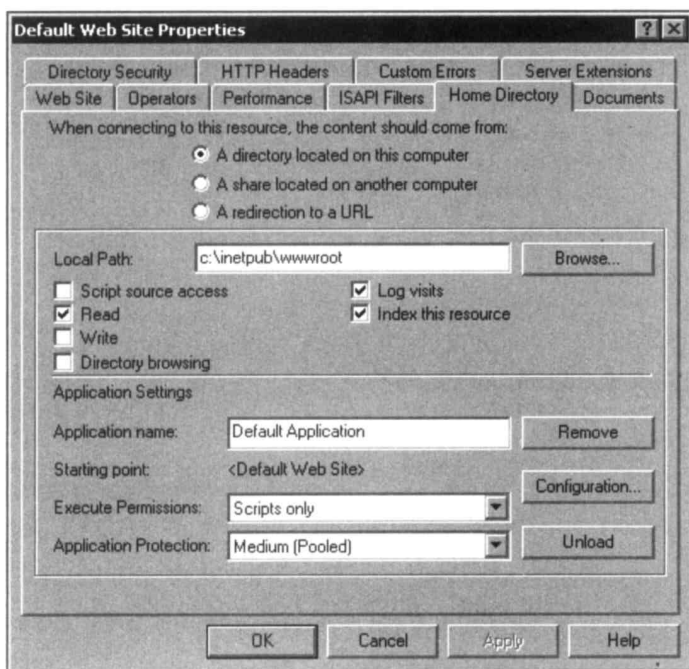


图1-15 Properties对话框

1. Home Directory 选项卡

站点的文件或目录方面的设置要在 Home Directory选项卡中设置。注意，最上面的选项允许指定用户请求应被定向到哪儿，这可以是在本地计算机的一个目录，远程计算机上的一个共享目录，甚至可以是一个URL。最后一种选择允许将浏览者重定向到另外一台计算机上。

中间部分的选项，包含控制这个站点或目录上什么功能被启用的设置。可以打开或关闭“写”和“读”权限；可浏览这个目录（没有缺省的Web页时）；可通过 Microsoft Indexing Service(MIS)建立页面索引，记录访问和用户。也可以允许访问源文件，有些 Windows 2000 新增选项用于使用 Distributed Authoring and Versioning (DAV)等远程编辑技术。

此选项卡的下部是设置的虚拟应用程序的地方。缺省 Web站点(图1-15中所示)被自动安装为一个虚拟的应用程序，与由安装程序创建的用于管理的目的或由其他服务使用的其他目录一样。在页面中开始使用 Active Server Component时，将详细讨论虚拟应用程序。

2. Web Site选项卡

Web Site选项卡用来向外界标识此Web站点，管理并发连接的数量及控制访问日志。

上部的选项设置站点的IP地址和ICP端口。对于缺省站点，Web服务将响应对于服务器来说可用的所有IP地址(假如有多个网络适配器或一个网络适配器中多个地址)，除非它们被分配给另一个站点。缺省的WWW访问端口是端口80，如图1-16所示。使用这个端口号意味着浏览者在他们请求中不需指定端口号。

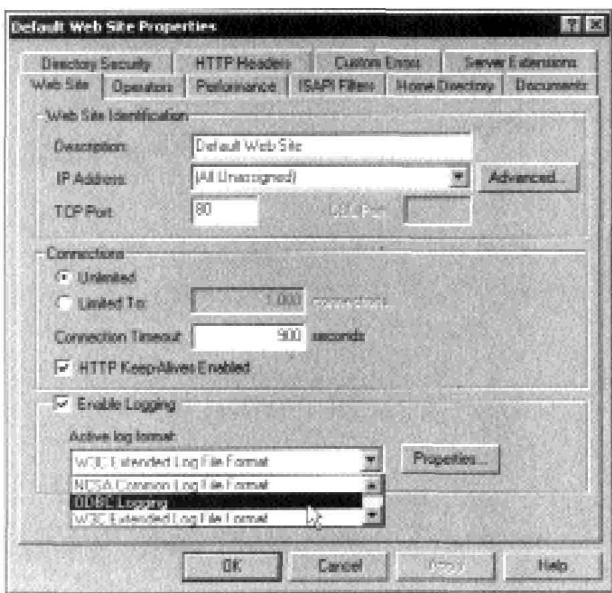


图1-16 Web Site选项卡

中间的选项控制可接受的并发连接数目和长时间运行 ASP脚本而不能完成执行的中止时限。也允许指定是否使用 HTTP Keep-Alives，这可为浏览器提供更好的性能，支持它们为多个请求保持连接打开状态。

在下部可设置想采用的访问日志的格式。缺省的是 W3C扩展日志文件格式(W3C Extended Log File Format)。也可以用 Properties按钮，打开一个对话框，进一步指定记录信息的细节。如果要记录到一个数据库，需要选择 ODBC Logging选项，并且为所使用的数据库表提供的 ODBC 系统数据源名称(一个系统 DSN，见第8章)。

3. Documents选项卡

此选项卡比上面提到的两个选项卡要简单得多。如果访问一个目录而没有指定文件名，将显示它指定的缺省页。例如，按图1-17中所示的设置，对于对http://stone broom.com的请求，将返回给用户的网页是http://stonebroom.com/Default.asp。

在Windows 2000中，不像早期版本，设置程序把 Default.asp放在列表的第一位，后面是 Default.htm。这是因为不包含ASP脚本的 .asp页面的处理速度几乎与纯 HTML一样快，因此微软推荐所有的页面均采用 .asp文件扩展名。

这个列表下面的复选框允许我们指定，将加到每个响应末尾的文本或 ASP文件的名称，这样可以为从这个站点或目录发送的所有页面加上一个标准的页脚。

4. HTTP Headers选项卡

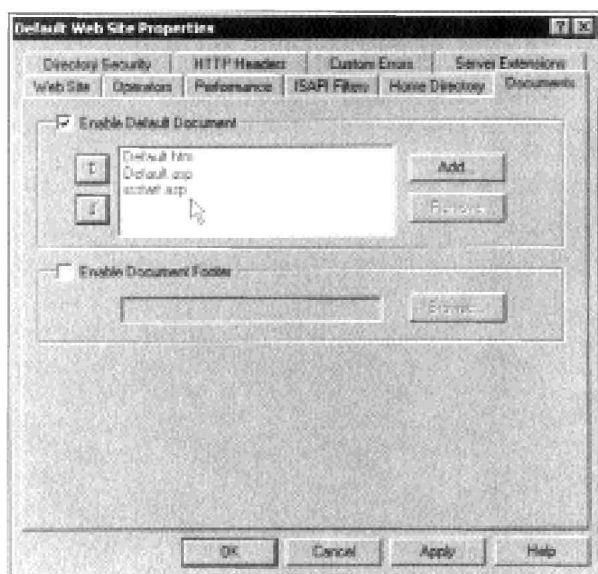


图1-17 Documents选项卡

先讨论一下什么是HTTP报头，如何和为什么使用它们，细节在下一章中讨论。先看一下HTTP Headers选项卡，对其所能完成的工作有一个感性认识，并说明如何找到相应的控件以改变相应的设置。

HTTP Headers选项卡如图1-18所示，上部选项是设置这个站点或目录中每一个文档的有

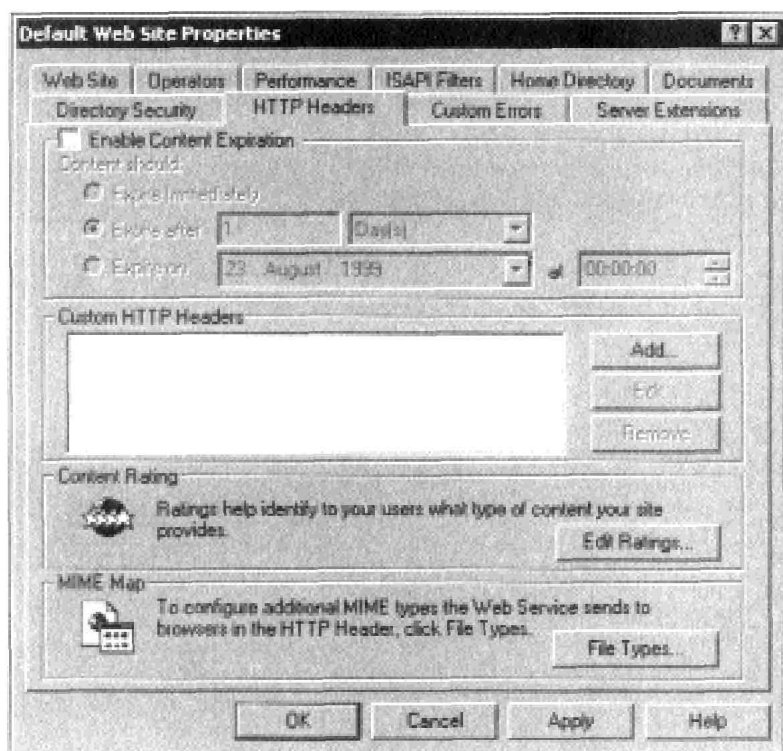


图1-18 HTTP Headers选项卡

效期的日期和时间。在这个时间后，在浏览器或代理服务器的缓存中的任何页面拷贝将变成无效的，并且不能显示。这个选项允许用户控制在必须从站点载入新页面之前缓存中的页面“保存”多长时间。

中间的选项允许在从这个站点或目录返回的所有页面响应中增加定制的 HTTP 报头。这主要应用于定制的客户应用程序，或特定的定制的数据管理。

下部的两个选项允许设置这个站点或目录页内容的等级划分，以及服务器发回客户端的 MIME 类型报头。内容等级划分用来描述页面上的内容的等级，如“性”、“暴力”、“恶性语言”等内容级别。MIME 类型报头用来指明客户期望从服务器上得到的数据类型。

5. Server Extensions 选项卡

这里要讨论的最后一个选项卡是 Server Extensions 选项卡。IIS 的一个令人激动的新功能是 Distributed Authoring and Versioning (DAV)，尽管在 Windows 2000 没有完全实现，但 DAV 最终将提供一个良好环境允许用户在他们各自的浏览器上编辑存放在 Web 服务器上的文档。DAV 扩展了目前由微软的 Visual InterDev 和 FrontPage 提供的功能，它允许编辑非 Web 文档如 Word 文档或 Excel 电子表格，对于一个 Intranet 环境，这些特征为用户提供了更加灵活和易于交互的系统。

DAV 可使用一系列驻留 Web 服务器上的软件扩展，类似于 FrontPage 扩展 (FrontPage 扩展用来使 InterDev 和 FrontPage 能够访问和下载服务器上的文件)。我们将在后续章节详细研究 DAV，先看一下 Server Extensions 选项卡，如图 1-19 所示。

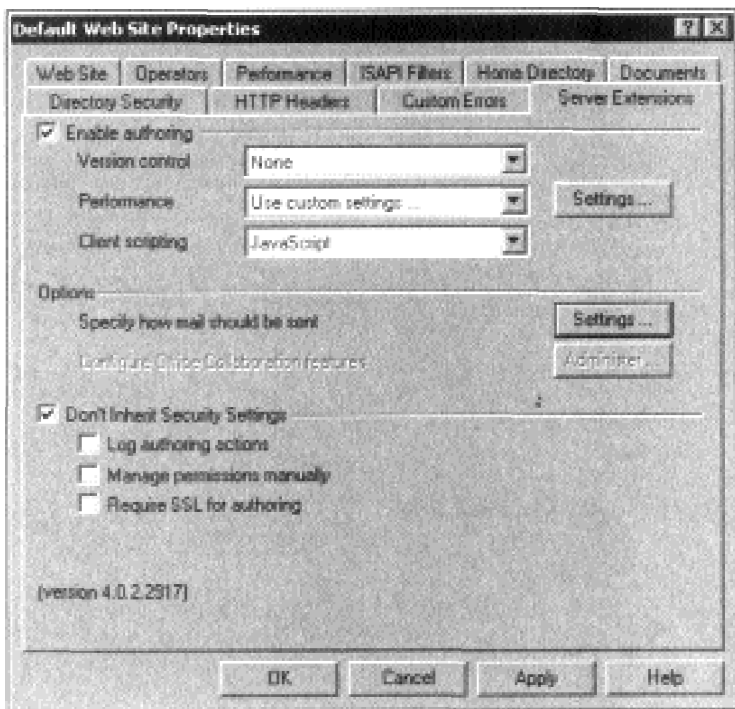


图1-19 Server Extensions 选项卡

这个选项卡允许控制这个虚拟应用程序所用到的著作和版本控制特性也包括指定用于网页编辑命中次数这种扩展功能的方式 (协调服务器和控制页面的缓存从而得到最佳的性能)。当使用 DAV 时，还可提供从页面上发送电子邮件的缺省设置，并设定安全性限制来保护内容。

1.4 ASP 3.0对象模型概要

在从编程的角度研究ASP的细节之前，必须看一下ASP对象模型。这是非常重要的。因为如果你没有在ASP 2.0上做过同样的工作，必须考虑ASP产生页面的方式。

1.4.1 对象环境概念

在版本1.0中，把能够为Web页面增加一些动态内容当作是一个令人激动的新方法。可以使用它从数据库读取数据，或操作从浏览器发来的数值。在ASP 2.0中，发生了很多非常引人注目的变化，增加了Microsoft Transaction Server(MTS)，它能够处理多个并发组件实例并提供进一步的扩展能力。这意味着动态Web页工作的整个概念发生了变化。

在此基础上，MTS允许使用分布在多个组件、应用程序和服务中的事务。例如，当通过Message Queue Service(以前是MSMQ)向远程计算机发送消息时，一个ASP页能够更新本地数据库。假如整个事务中的一部分失败，整个处理过程将被恢复到系统原来的状态。

MTS，以及组件的使用日益广泛，导致了使用ASP的Web应用程序的发展，而不再是单独的简单动态页。在由ASP脚本实例化的组件内，这个ASP页的环境是可用的。环境包含了所有内部的ASP对象(我们很快要遇到的)。因此，可以使用它获得用户请求的信息并创建相应的响应。

由于这个环境包含了整个ASP的对象模型，因此，它允许程序员对ASP以及所使用的各组件所执行的复杂处理进行更多的控制。通常认为对象模型的“根”是请求、响应和其他内部对象；而真实情况是(自从ASP 2.0以来就已经是)，这个根是一个称为ObjectContext的对象。

1. 引用ASP页面的环境

在ASP 1.0中，引用环境的唯一方式是通过每次开始执行ASP页时由ASP引发的事件：OnStartPage。这个事件以一个参数为ASP页提供ObjectContext对象。在一个组件内，能够在一个全局变量中捕获对ObjectContext的一个引用，以供代码使用。例如，下面这段VB代码把环境存储在一个称为objContext的局部变量中。

```
Dim objContext

Public Sub OnStartPage(theContext As ScriptingContext)
    Set objContext = theContext
End Sub

Public Sub DoSomething()
    ...
    objContext.Response.Write "This is some text written into the page"
    ...
End Sub
```

从上面的代码中看出，在ASP 1.0中，对象的环境是ScriptingContext类型，这是一种在ASP DLL中定义的对象类型并为创建引用文件asp.dll的代码所用。然而很明显，控制事务并提供高效的进程外的组件执行(ASP 2.0中MTS管理的一个任务)，必须采取不同的做法。页环境必须是显式可用的，无须在每个页面的开始处保留对它的引用。

因此，在ASP 2.0中，微软引入了ObjectContext对象。然而，由于ScriptingContext仍然通过OnStartPage事件起作用，许多组件的创作者考虑到程序的向后兼容性，避免使用ObjectContext对象，甚至宁愿以降低性能为代价。现在，由于有了ASP 3.0，情况发生了变化。

在Windows 2000中，MTS做为COM+技术的一部分被融入操作系统中，除非明确决定避免它，否则将在缺省状态时被用于在ASP中实例化的任何一个组件。

2. 引用ObjectContext 对象

从ASP 2.0开始，已经能够通过 ASP提供的GetObjectContext 方法获得一个对当前页面环境的引用。这就意味着不必再通过一个页面存储对环境的引用，可以在任何需要的时候得到它。

```
Set objLocalContext = GetObjectContext
```

```
objLocalContext.Response.Write "This is some text written into the page"
```

这允许我们的对象变成无状态的 (Stateless)，换句话说，当完成执行一个特定的方法后，不需要保留对任何数值或对象的引用。假如以前你没有使用过 ASP 2.0和MTS，这看起来也许是一个有些深奥的概念。然而它是极其重要的，我们将在后续章节讨论关于这个问题的大量细节。

1.4.2 ASP内置的对象

看到了ASP如何以ObjectContext对象的形式提供一个“根”对象，就可以来了解其他的内置对象如何构建在它上面，以提供对客户端请求、我们所创建的响应和其他使编写脚本更容易的对象的访问。

原来的ScriptingContext对象仍然在使用，并且 OnStartPage 事件也是可用的，但现在已经陈旧了，应该只用于在有 ASP 1.0的IIS 3.0上执行的页面，或在需要绝对的向后兼容性时使用。

由ASP提供的两个主要内置对象，直接映射访问 Web服务器时客户端的两个行为。另外 4个提供了附加功能，对编写脚本是非常有用的。每个对象都提供了一系列的集合、属性和方法，这在后续章节将加以介绍。

Request对象为脚本提供客户端在请求一个页面或传送一个窗体时提供的所有信息，这包括能够标识浏览器和用户的 HTTP变量，存储他们的浏览器对应于这个域的 cookie，以及附在 URL后面的值(查询字符串或页面中 <Form>段中的 HTML控件内的值)。它也给我们提供了通过Secure Socket Layer(SSL)，或其他的加密通信协议，访问证书的能力并提供有助于管理连接的属性。

Response对象用来访问所创建的并返回客户端的响应。它为脚本提供了标识服务器和性能的HTTP变量，发送给浏览器的信息内容和任何将在 cookie中存储的信息。它也提供了一系列用于创建输出页的方法，如无所不在的 Response.Write方法。

Application对象是在为响应一个 ASP页的首次请求而载入 ASP DLL时创建的，它提供了存储空间用来存放变量和对象的引用，可用于所有的页面，任何访问者都可以打开它们。

独特的Session对象是在每一位访问者从 Web站点或Web应用程序中首次请求一个 ASP页时创建的，它将保留到默认的期限结束(或者由脚本决定中止的期限)。它与Application对象一样提供一个空间用来存放变量和对象的引用，但只能供目前的访问者在会话的生命期中打开的页面使用。

Server对象提供了一系列的方法和属性，在使用 ASP编写脚本时是非常有用的。最常用的

是Server.CreateObject 方法，它允许我们在当前页的环境或会话中在服务器上实例化其他COM对象。还有一些方法能够把字符串翻译成在 URL和HTML中使用的正确格式，这通过把非法字符转换成正确、合法的等价字符来实现。

ASPErrror对象是ASP 3.0中的一个新的对象，通过 Server对象的GetLastError方法使用。它提供了发生在ASP中的上一次错误的详细信息。

可以把这些对象看作是基于ObjectContext对象的一个层次关系的成员，这有助于理解它们与接受和响应客户请求的过程之间的关系，如图 1-20所示。图中表现了ASP和创建及服务于ASP页的过程之间的关系。

我们将在后续章节详细研究每一个对象，以及使用它们的方式。

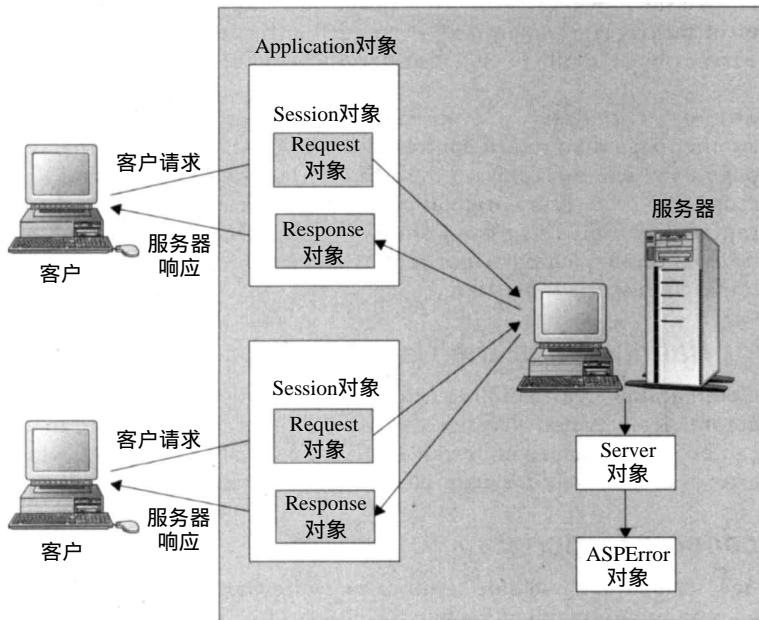


图1-20 各种对象之间的关系

1.5 ASP 3.0中的新特性

假如读者已经熟悉了ASP 2.0，并正在寻找3.0版本中的实际改变的列表，那么将在下面发现这些信息。假如读者是一个 ASP的初学者，可以越过本章到下一章，那里循序渐进地介绍了ASP对象和它们的用法。

1.5.1 ASP 3.0新特性概要

在ASP 3.0中，有一些新的特性或经历较大的变化或改进的特性。

1. 无脚本的ASP

如早先提到的，ASP处理不包括任何脚本的.asp页的速度是很快的，假如你正在创建的站点或Web应用程序文件最终可能使用ASP，最好让这些文件使用.asp文件扩展名，而不用考虑它们是包含服务器端脚本还是仅仅包含静态(HTML和文本)内容。

2. 新的流向控制能力

到目前为止，假如想把执行转向另外的一个 ASP 页，不得不使用 Response.Redirect 语句，这个工作通过向客户端发送一个响应来指示其载入新的页面来实现。然而这对客户端来讲是费事的。而且当代理服务器用于客户端时，会引起错误的消息。ASP 3.0 为 Server 对象提供了两个新的方法，允许在服务器上转换页面而不需要新的客户端的请求。

Server.Transfer 是转换执行到另一个页面；而 Server.Execute 是执行另一个页面，然后将控制返回原来的页面。在新的页面里可访问原来页面的环境，包括 Response 和 Request 等所有 ASP 对象，但是不能访问页面范围的变量。假如原始的页面使用了一个事务标志（在开放的 <% @...%> 元素中），事务的环境被传递到新的页面。假如第二个 ASP 文件的事务标志表明事务是受到支持的或需要的，则现有的事务将被使用，而不会开始一个新的事务。

3. 错误处理和新的 ASPError 对象

通过提供一个用 Server.Transfer 方法自动调用的定制的 ASP 页面，提供了可配置的错误处理。在这个 ASP 页面中，Server.GetLastError 可被用来返回一个 ASPError 对象的实例，其中包含了错误的细节，例如错误的描述和相关的行号。

4. 编码后的 ASP 脚本

ASP 脚本和客户端脚本现在可以使用 BASE64 加密法进行编码。更高水平的加密计划将出现在 ASP 的未来新版本里（注意，这个特征是由 VBScript 5.0 和 JScript 5.0 脚本引擎实现的，因此在脚本被执行时要求这些引擎存在）。编码后的脚本将在运行时由脚本引擎解码。因此不必使用别的工具，尽管这不是很安全的加密方法，但能够保护脚本不被一般的用户浏览和拷贝。

5. 包含脚本文件的一种新方式

除了使用 <!--#Include....--> 元素使服务器端的 IIS 包含脚本代码文件，ASP 3.0 也能够“包含”其自己。<SCRIPT> 元素与 RUNAT=“SERVER”和 SRC=“path_and_filename”属性共同使用，来包含基于服务器的脚本代码文件。相对物理路径或虚拟路径也可以用在 SRC 属性中：

```
<SCRIPT LANGUAGE="language" RUNAT="SERVER" SRC="path_and_filename"></SCRIPT>
```

6. Server Scriptlets

ASP 3.0 支持一种强有力的新的脚本技术，称之为 Server Scriptlets。这些是驻留在服务器上的 XML 格式的文本文件，可以像一般的 COM 对象（即 Active 服务器组件）为 ASP 所用。这样可以把 Web 应用程序的业务逻辑脚本过程更容易地实现为一个可重用的组件。

7. 增强性能的 Active 服务器组件

ASP 中的许多 Active 服务器组件得到了改进，能够提供更好的性能和附加的功能，一个例子就是新的 Browser Capabilities 组件。除此之外，还有一些新的组件。例如，XML 分析器使应用程序可以处理服务器上的 XML 格式的数据。同时，提供了 ADO 与 XML 更加紧密的集成（通过 Windows 2000 所提供的新的 ADO 2.5 版），这为以 XML 格式存贮和获取数据，提供了新的机会。

8. 性能

为改善 ASP 和 IIS 的性能和可扩展性，新版本做了大量的工作。这包含 ASP 中的自我调整特征，它可以检测阻塞情况并自动增加可用线程的数量。当请求在执行中受到外部资源的阻塞时，ASP 能够检测出来，并为同时执行附加请求和继续正常处理提供更多的线程。但是，假如 CPU 变得超负荷，ASP 会减小可用线程的数量，以便当过多的非阻塞请求同时执行时，

将线程切换次数最小化。

1.5.2 对ASP 2.0的改进

下面的一些特征是从2.0版本中改进或升级来的。

1. 缓冲缺省为打开状态

ASP提供可选的输出缓冲。从IIS 4.0开始,这使得脚本执行得更快,并提供对流向浏览器的输出的控制能力。在ASP 3.0这个改进的性能通过改变Response.Buffer属性的缺省设置为True而反映出来。缺省状态下缓冲是打开的,这意味着最终的输出只有在进程完成时,或脚本调用Response.Flush或Response.End方法时,才送至客户端。

注意,可以通过设置Response.Buffer属性为False,关闭缓冲。只有这样,才能发送XML格式化输出给客户端,让XML分析器在收到输出后开始工作。也可以使用Response.Flush发送大页面的一部分,这样使用户可以很快看到部分输出。

2. Response.IsClientConnected的变化

Response.IsClientConnected属性可以在没有任何内容发送给客户端的情况下被读取到。在ASP 2.0中,这只在至少有一部分内容被发送后才能返回准确的信息。这一改进解决了IIS必须响应每个客户的请求(即使客户可能已经转移到另一个页面或站点)的问题。同时如客户在3秒内没有再连接,服务器上创建的完整的输出信息将被丢弃。

3. 带有默认文档的查询字符串

假如一个用户访问一个站点而不提供所请求页面的名字,默认的文档(如存在的话)将被送往客户端。然而假如他们提供了附在URL后面的查询字符串,这在早先的ASP版本中是被忽略的,而在IIS 5.0和ASP 3.0中这个查询字符串将被送到缺省页面。例如,在一个URL为: <http://www.wrox.com/store/>的目录中缺省页面为default.asp,则下面这两种情况都将名称/值对Code=1274送往default.asp页面:

```
http://www.wrox.com/store/?code=1274  
http://www.wrox.com/store/default.asp?code=1274
```

4. 服务器端包含文件的安全性

服务器端的包含文件常用于一些敏感的信息,如数据库连接字符串或其他访问细节。一个虚拟的路径(即URL而不是完整的物理磁盘文件路径)可以用来指定这些文件。在这种情况下,早先的ASP版本不核对文件的安全设置和用户的证书;换句话说,授权(验证后)的用户和匿名的Web服务器帐号都没有与文件的访问控制列表相比较。在IIS 5.0和ASP 3.0,这些证书将被检查以防止非授权访问。

5. 可配置项移到元数据库中

在IIS 5.0中ProcessorThreadMax和ErrorsToNTLog的注册项被移到元数据库中,所有的ASP可配置参数能够通过Active Directory和Active Directory服务接口(ADSI)在元数据库中修改。

6. 应用程序中的多线程对象的性能

为了在常有多个并发请求的ASP中获得最佳性能,组件应是多线程的(Both-Threaded)——即单线程单元(Single Threaded Apartment, STA)和多线程单元(Multi-Threaded Apartment, MTA),并且支持COM Free-Thread Marshaller(FTM)。不支持FTM的双线程的COM对象假如

被存储在 ASP Application 状态对象中，将导致运行失败。

7. 更早释放 COM 对象

在 IIS 5.0 中，实例化的对象或组件可更早释放。在 IIS 4.0 中，COM 对象只有在 ASP 处理完一个页面时才能释放。在 IIS 5.0 中，假如一个 COM 对象不使用 OnEndPage 方法，且对象的引用计数达到零，则这个对象在处理完成之前就被释放了。

8. 缺省时 ASP 允许进程外组件

定制的本地服务器组件现在可以从 IIS 中被实例化，而不需要改变元数据库的设置。控制本地服务器实例化的元数据库属性 `AspAllowOutOfProcComponents` 缺省值为 1，在 IIS 早期版本为 0。

9. COM 对象的安全性

IIS 使用了新的由 COM+ 提供的 cloaking 特性，因此，从 ASP 实例化的本地服务器应用程序可以运行在原始的客户端的安全环境中。在早期版本中，安全环境被指派到本地服务器 COM 对象，依赖于调用进程的身份。

10. 缺省时组件运行在进程外

在 ASP 早期版本中，所有在 ASP 页面环境中创建的组件缺省时运行在进程内。

为在组件的性能和 Web 服务器安全性之间折衷，对于一个虚拟的应用程序，可以从 Properties 对话框中 Application Protection 的三个选项中选择：

- Low (IIS Process)

这种设置的 ASP 虚拟应用程序的所有应用程序可执行文件和组件都运行在 Web 服务器可执行文件 (Inetinfo.exe) 的进程 (即内存空间) 中。因此，Web 服务器就有受到可执行文件或组件失败影响的风险，然而这提供了最快的和最少资源浪费的应用程序执行选项。

- Medium (Pooled)——这是缺省设置

这种设置的 ASP 虚拟应用程序的所有应用程序可执行文件和组件都运行在单个共享的 DLLHost.exe 实例的进程 (即内存空间) 中。这保护了 Web 服务器可执行文件 (Inetinfo.exe) 免受任何一个可执行文件或组件失败带来的风险。然而，可执行文件或组件的失败会引起 DLLHost.exe 进程失败，进而所有其他驻留其中的可执行文件和组件也会失败。

- High (Isolated)

这种设置的 ASP 虚拟应用程序的所有应用程序可执行文件和组件都运行在单个 DLLHost.exe 实例的进程 (即内存空间) 中，但是每个 ASP 应用程序都有自己的 DLLHost.exe 实例。DLLHost.exe 对应用程序而言是独有的，这保护 Web 服务器可执行文件免受任何一个可执行文件或组件失败带来的危险，同时也保护了虚拟应用程序，免受因其他虚拟应用程序的可执行文件或组件失败带来的问题。微软建议在任何一个 Web 服务器上最多驻留 10 个隔离的虚拟应用程序。

推荐的配置是：在它们自己的进程中运行对于任务关键的应用程序，即 High (Isolated)；余下的所有应用程序在一个共享的进程中运行，即 Medium (Pooled)。也可设置组成每个虚拟应用程序的脚本和组件的执行权限 (Execute Permission)，三个选项是：

1) None：在这个虚拟的应用程序中不能运行脚本或可执行文件。在实际效果上，这提供了一个在必要时快速和简便地禁止一个应用程序的方式。

2) Scripts only：仅允许脚本文件，诸如 ASP、IDC 或其他，能够在这个虚拟应用程序中运

行，可执行文件不能运行。

3) Scripts and Executables：允许任何脚本和可执行文件在这个虚拟应用程序中运行。

1.5.3 VBScript 5.0中的新特性

能够在ASP中应用的特性包括了那些由脚本引擎所提供的特性，这意味着 VBScript的改进也可在ASP中应用。VBScript的改进如下所述。

1. 在脚本中使用类

在VBScript中实现完整的VB类(class)模型，但明显的例外是在 ASP服务器端的脚本事件。可以在脚本中创建类，使它们的属性和方法能够用于页面的其余代码，例如：

```
Class MyClass

    Private m_HalfValue                'local variable to hold value of HalfValue

    Public Property Let HalfValue(vData) 'executed to set the HalfValue property
        If vData > 0 Then m_HalfValue = vData
    End Property

    Public Property Get HalfValue()      'executed to return the HalfValue property
        HalfValue = m_HalfValue
    End Property

    Public Function GetResult()          'implements the GetResult method
        GetResult = m_HalfValue * 2
    End Function

End Class

Set objThis = New MyClass

objThis.HalfValue = 21

Response.Write "Value of HalfValue property is " & objThis.HalfValue & "<BR>"
Response.Write "Result of GetResult method is " & objThis.GetResult & "<BR>"
...
```

这段代码产生如下结果：

```
Value of HalfValue property is 21
Result of GetResult method is 42
```

2. With结构

VBScript 5.0支持With结构，使访问一个对象的几个属性或方法的代码更加紧凑：

```
...
Set objThis = Server.CreateObject("This.Object")

With objThis
    .Property1 = "This value"
    .Property2 = "Another value"
    TheResult = .SomeMethod
End With
...
```

3. 字符串求值

Eval函数(过去只在JavaScript和JScript中可用)目前在VBScript 5.0中已经得到了支持。它允许创建包含脚本代码的字符串，值可为 True或False，并在执行后可得到一个结果：


```

...
datYourBirthday = Request.Form("Birthday")
strScript = "datYourBirthday = Date()"

If Eval(strScript) Then
    Response.Write "Happy Birthday!"
Else
    Response.Write "Have a nice day!"
End If
...

```

4. 语句执行

新的Execute函数允许执行一个字符串中的脚本代码，执行方式与 Eval函数相同，但是不返回结果。它可以用来动态创建代码中稍后执行的过程，例如：

```

...
strCheckBirthday = "Sub CheckBirthday(datYourBirthday)" & vbCrLf _
    & "    If Eval(datYourBirthday = Date()) Then" & vbCrLf _
    & "        Response.Write "Happy Birthday!" & vbCrLf _
    & "    Else" & vbCrLf _
    & "        Response.Write "Have a nice day!" & vbCrLf _
    & "    End If" & vbCrLf _
    & "End Sub" & vbCrLf
Execute strCheckBirthday
CheckBirthday(Date())
...

```

一个回车返回(如程序中示)或冒号字符“:”可用来分隔一个字符串中的各条语句。

5. 设置地区

新的SetLocale方法可以用来改变脚本引擎的当前地区，可正确显示特殊的地区特定字符，如带重音符的字符或来自不同字符集的字符。

```

strCurrentLocale = GetLocale
SetLocale("en-gb")

```

6. 正则表达式

VBScript 5.0 现在支持正则表达式 (过去只在 JavaScript、JScript 和其他语言中可用)。RegExp 对象常用来创建和执行正则表达式，例如：

```

strTarget = "test testing tested attest late start"
Set objRegExp = New RegExp                                'create a regular expression

objRegExp.Pattern = "test*"                               'set the search pattern
objRegExp.IgnoreCase = False                             'set the case sensitivity
objRegExp.Global = True                                   'set the scope

Set colMatches = objRegExp.Execute(strTarget)             'execute the search

For Each Match in colMatches                             'iterate the colMatches collection
    Response.Write "Match found at position " & Match.FirstIndex & ". "
    Response.Write "Matched value is '" & Match.Value & "'.<BR>"
Next

```

执行结果如下：

```

Match found at position 0. Matched value is 'test'.
Match found at position 5. Matched value is 'test'.
Match found at position 13. Matched value is 'test'.
Match found at position 22. Matched value is 'test'.

```

7. 在客户端VBScript中设置事件处理程序

这不是直接应用于ASP的脚本技术,这个新的特性在编写客户端的VBScript时是很有用的。现在可以动态指定一个函数或子程序与一个事件相关联。例如,假设一个函数的名称为MyFunction(),可把它指定给按钮的OnClick事件:

```
Function MyFunction()  
...  
'Function implementation code here  
...  
End Function  
...  
Set objCmdButton = document.all("cmdButton")  
Set objCmdButton.OnClick = GetRef("MyFunction")
```

这提供了JavaScript和JScript中的类似功能,函数可以被动态地指定为一个对象的属性。

8. VBScript中的On Error Goto 0

尽管这个技术早先没有被文档记载,但在现有的VBScript版本中能够使用(有着VB背景并且有好奇心的人可能早已发现这个秘密)。它现在已被记录在文档中,并且在执行On Error Resume Next后能够用来“关闭”页面中的定制错误处理。结果是任何后来的错误将引发一个浏览器级或服务级的错误及相应的对话框/响应。

1.5.4 JScript 5.0中的新特性

JScript 5.0唯一的改变是引入了错误处理。

Java风格的try和catch结构在JScript 5.0中得到了支持。例如:

```
function GetSomeKindOfIndexThingy() {  
  
    try {  
        // If an exception occurs during the execution of this  
        // block of code, processing of this entire block will  
        // be aborted and will resume with the first statement in its  
        // associated catch block.  
        var objSomething = Server.CreateObject("SomeComponent");  
        var intIndex = objSomething.getSomeIndex();  
        return intIndex;  
    }  
  
    catch (exception) {  
        // This code will execute when *any* exception occurs during the  
        // execution of this function  
        alert('Oh dear, the object didn't expect you to do that');  
    }  
}
```

内建的JScript Error对象有3个属性,它们定义了上次的运行期错误。可在catch块中使用它们获得有关错误的更多信息。

```
alert(Error.number); // Gives the numeric value of the error number  
// AND the result with 0xFFFF to get a 'normal' error number in ASP
```

```
alert(Error.description); // Gives an error description as a string
```

假如你想抛出自己的错误,可用一个定制的异常对象引发一个错误(或异常)。然而,由于

没有内建的异常对象，必须自己定义一个结构：

```
// Define our own Exception object
function MyException(intNumber, strDescription, strInfo) {
    this.Number = intNumber;           // Set the Number property
    this.Description = strDescription;  // Set the Description property
    this.CustomInfo = strInfo;         // Set some 'information' property
}
```

这样的对象可用在页面中引发定制的异常。这通过使用 throw 关键字，然后检查 catch 块中的异常类型来实现：

```
function GetSomeKindOfIndexThingy() {
    try {
        var objSomething = Server.CreateObject("SomeComponent");
        var intIndex = objSomething.getSomeIndex();
        if (intIndex == 0) {
            // Create a new MyException object
            theException = new MyException(0x6F1, "Zero index not permitted",
                                           "Index_Err");

            throw theException;
        }
        return intIndex;
    }

    catch (objException) {
        if (objException instanceof MyException) {
            // This is one of our custom exception objects
            if (objException.Category == "Index_Err") {
                alert('Index Error: ' + objException.Description);
            }
            else {
                alert('Undefined custom error: ' + objException.Description);
            }
        }
        else {
            // Not "our" exception, so display it and raise to next higher routine
            alert(Error.Description + ' (' + Error.Number + ')');
            throw exception;
        }
    }
}
```

1.5.5 其他的新特性

还有几个新特性已经能够在 IIS 5.0 中使用了。

1. DAV(Distributed Authoring and Versioning)

这个标准由 Internet Engineering Task Force (IETF) 创建，目前为 1.0 版本。它允许作者在几个不同的位置共同创建和维护 Web 页和其他的文档。它用于提供上传 (upload) 和下载访问，并控制版本号使工作过程能够得到相应的管理，IE 包含与 IIS 5.0 中的 DAV 的集成的特性。但是，在 IETE 标准和当前的 IIS 5.0 版本中，尚未实现版本控制能力。

2. 引用类型库

在过去，常常使用服务器端的包含文件，把常数从一个类型库（例如脚本对象、ADO 或 MSMQ）增加到 ASP 页面。这是必须的，因为 ASP 不能像 VB 那样创建对类型库或组件 DLL 的引用。在 IIS 5.0 中，不必再为常数使用包含文件。可以在 <HEAD> 部分放一个 HTML 注释风格的元素，来直接访问一个组件的类型库。

```
<!-- METADATA TYPE="typelib" FILE="c:\WinNT\System32\scrrun.dll" -->
```

这将使指定文件中的常量在当前 ASP 页面中都可用 (尽管这是 IIS 5.0 中的一个新特性,但在 IIS 4.0 中虽没有记入文档,但已经可以使用了)。

3. FTP 下载续传

FTP 服务现在终于提供了下载的续传功能。假如一个文件部分下载后停止,它能够从断点处继续下载。这意味着没有完成下载的文件不需要再次下载整个文件。

4. HTTP 压缩

IIS 现在能够自动实现对静态或动态产生的文件的 HTTP 数据流压缩并高速缓存压缩的静态文件。在与适当地准备好的客户端通信时,这会提供更快响应并减少网络的负载。

1.6 小结

本章简单地讨论在使用 ASP 3.0 时需要注意的主要问题。主要从有经验的 Web 开发人员的角度来讨论,并假设读者通过使用早先版本的 ASP,有了 ASP 方面的经验,或者至少了解当客户机与服务器交互时 Web 是如何工作的。

至此,读者应该对 ASP 能够提供的功能有了一个大致的了解,即早先版本已有的特性,以及 ASP 3.0 的新特性。假如感觉到自己还不能完全理解 ASP 对象模型的概念,或从其他来源访问这个对象模型的方式,不用担心这个问题。现在只要对这里所讨论的问题有一个大致全面的了解,在后续章节中详细研究相关问题时,再进一步加深理解。

事实上,下一章将研究两个主要对象 Request 和 Response。在后面章节中,将陆续研究其他对象,然后将研究在页面中如何使用 Active 服务器组件,包括通过 VBScript 和 JScript 脚本引擎可用的脚本对象。在第 7 章,将研究错误的处理 (这在 ASP 3.0 中改进很大),以及如何调试 ASP 页。

本章主要讨论了下列重要题目:

- 什么是 ASP。
- ASP 如何与 IIS 连接。
- 安装问题与管理。
- 对象环境的概念。
- ASP 3.0 对象模型。
- ASP 3.0 版本中的新特性。

不要犹豫,马上进入第 2 章吧,开始学习关于客户端的请求和服务器的响应的知识。