# ASP 教程

ASP（Active Server Pages 动态服务器页面）是一种生成动态交互性网页的强有力工具。

在我们的 ASP 教程中，您将学到 ASP 的相关知识，以及如何在服务器上执行脚本。

ASP 脚本只能在服务器端执行，所以无法在浏览器中查看 ASP 代码，只能看到由 ASP 输出的纯粹 HTML 代码。 在菜鸟教程，每个实例均可显示出以往被隐藏的 ASP 代码。这样，您就可以更容易地理解它们的工作原理。

# ASP 简介

ASP 文件可包含文本、HTML 标签和脚本。ASP 文件中的脚本可在服务器上执行。

## ASP 是什么？

* ASP 代表 **A**ctive **S**erver **P**ages （动态服务器页面）
* ASP 是一项微软公司的技术
* ASP 是在 **IIS** 中运行的程序
* IIS 代表 **I**nternet **I**nformation **S**ervices （Internet 信息服务）
* IIS 是 **Windows 2000** 的免费组件
* IIS 同时也是 **Windows NT 4.0 Option Pack** 的组成部分
* 此可选包可从微软站点 **下载**
* **PWS** 的体积更小 - 不过拥有 IIS 的完整功能
* PWS 可在您的 **Windows 95/98 CD** 中找到

## ASP 兼容性

* 运行 IIS，需要 Windows NT 4.0 或更高的版本
* 运行 PWS，需要 Windows 95 或更高的版本
* ChiliASP 是一种在非 Windows 操作系统上运行 ASP 的技术
* InstantASP 是另一种在非 Windows 操作系统上运行 ASP 的技术

## ASP 文件是什么？

* ASP 文件与 HTML 文件类似
* ASP 文件可包含文本、HTML、XML 和脚本
* ASP 文件中的脚本可在服务器上执行
* ASP 文件的文件扩展名是 ".asp"

## ASP 和 HTML 有何不同？

* 当浏览器请求某个 HTML 文件时，服务器会返回这个文件
* 当浏览器请求某个 ASP 文件时，IIS 将这个请求传递给 ASP 引擎。ASP 引擎会逐行地读取这个 ASP 文件，并执行文件中的脚本。最后，ASP 文件将以纯 HTML 的形式返回到浏览器。

## ASP 能为您做什么？

* 动态地编辑、改变或者添加网页的任何内容
* 对由用户从 HTML 表单提交的查询或者数据作出响应
* 访问数据或者数据库，并向浏览器返回结果
* 为不同的用户定制网页，提高这些页面的可用性
* 用 ASP 替代 CGI 和 Perl 的优势在于它的简易性和速度
* 提供安全性保障 - •由于 ASP 代码不能从浏览器查看
* 优秀的 ASP 编程可最小化网络流量

IMG_256**注释：**由于 ASP 脚本在服务器上运行，浏览器无需支持脚本就可以显示 ASP 文件！

# ASP 基本语法规则

在我们的 ASP 教程中，每个实例都提供隐藏的 ASP 源代码。这样会使您更容易理解它们的工作原理。

## 向浏览器写输出

ASP 文件通常包含 HTML 标签，就像 HTML 文件。然而，ASP 文件也能包含服务器脚本，这些脚本被分隔符 <% 和 %> 包围起来。

服务器脚本**在服务器上执行**，可包含你所选用的脚本语言的合法的表达式、语句、程序或者运算符。

### **response.write 命令**

response.write 命令用来向浏览器写输出。下面的实例向浏览器传送了一段文本："Hello World"：

## 在 ASP 中使用 VBScript

您可以在 ASP 中使用若干种脚本语言。然而，默认的脚本语言是 VBScript：

<!DOCTYPE html>  
<html>  
<body>  
<%  
response.write("Hello World!")  
%>  
</body>  
</html>

上面的实例向文档的 body 部分写入了文本 "Hello World!"。

## 在 ASP 中使用 JavaScript

如果需要设置 JavaScript 为某个特定页面的默认脚本语言，您必须在页面的顶部插入一行语言说明：

<%@ language="javascript"%>  
<!DOCTYPE html>  
<html>  
<body>  
<%  
Response.Write("Hello World!")  
%>  
</body>  
</html>

**注释：**与 VBScript 不同，JavaScript 对大小写敏感！你必须根据 JavaScript 的需要使用不同的大小写字母编写您的 ASP 代码。

## VBScript 与 JavaScript 的不同

当从一个用 VBScript 编写的 ASP 文件中调用 VBScript 或者 JavaScript 子程序时，可以使用 "call" 关键词，后面跟着子程序名称。假如子程序需要参数，当使用 "call" 关键词时，参数必须包含在括号内。假如您省略了 "call" 关键词，则参数不必包含在括号内。如果子程序没有参数，那么括号则是可选的。

当从一个用 JavaScript 编写的 ASP 文件中调用 VBScript 或者 JavaScript 子程序时，必须在子程序名后使用括号。

# ASP 表单和用户输入

Request.QueryString 和 Request.Form 命令用于从表单取回信息，比如用户的输入。

# ASP Cookies

cookie 常用于识别用户。

## Cookie 是什么？

cookie 常用用于识别用户。cookie 是一种服务器留在用户计算机上的小文件。每当同一台计算机通过浏览器请求页面时，这台计算机将会发送 cookie。通过 ASP，您能够创建并取回 cookie 的值。

## 如果浏览器不支持 Cookie 该怎么办？

如果您的应用程序需要与不支持 cookie 的浏览器打交道，那么您不得不使用其他的办法在您的应用程序中的页面之间传递信息。这里有两种办法：

### **1. 向 URL 添加参数**

您可以向 URL 添加参数：

<a href="welcome.asp?fname=John&lname=Smith">Go to Welcome Page</a>

然后在 "welcome.asp" 文件中取回这些值，如下所示：

<%  
fname=Request.querystring("fname")  
lname=Request.querystring("lname")  
response.write("<p>Hello " & fname & " " & lname & "!</p>")  
response.write("<p>Welcome to my Web site!</p>")  
%>

### **2. 使用表单**

您可以使用表单。当用户点击 Submit 按钮时，表单会把用户输入传给 "welcome.asp" ：

<form method="post" action="welcome.asp">  
First Name: <input type="text" name="fname" value="">  
Last Name: <input type="text" name="lname" value="">  
<input type="submit" value="Submit">  
</form>

然后在 "welcome.asp" 文件中取回这些值，如下所示：

<%  
fname=Request.form("fname")  
lname=Request.form("lname")  
response.write("<p>Hello " & fname & " " & lname & "!</p>")  
response.write("<p>Welcome to my Web site!</p>")  
%>

# ASP Session 对象

Session 对象用于存储关于用户会话（session）的信息，或者更改用户会话（session）的设置。

## Session 对象

当您在计算机上操作某个应用程序时，您打开它，做些更改，然后关闭它。这很像一次对话（Session）。计算机知道您是谁。它清楚您在何时打开和关闭应用程序。然而，在因特网上问题出现了：由于 HTTP 地址无法保持状态，Web 服务器并不知道您是谁以及您做了什么。

ASP 通过为每个用户创建一个唯一的 cookie 来解决这个问题。cookie 被传送至用户的计算机上，它含有可识别用户的信息。这种接口被称作 Session 对象。

Session 对象用于存储关于用户会话（session）的信息，或者更改用户会话（session）的设置。

存储于 Session 对象中的变量存储单一用户的信息，并且对于应用程序中的所有页面都是可用的。存储于 session 变量中的公共信息通常是 name、id 和参数。服务器会为每个新的用户创建一个新的 Session，并在 session 失效时撤销掉这个 Session 对象。

## Session 何时开始？

Session 开始于：

* 某个新用户请求了一个 ASP 文件，并且 Global.asa 文件引用了 Session\_OnStart 子程序
* 某个值存储在 Session 变量中
* 某个用户请求了一个 ASP 文件，并且 Global.asa 使用 <object> 标签通过 session 的 scope 来实例化某个对象

## Session 何时结束？

如果用户没有在规定的时间内在应用程序中请求或者刷新页面，session 就会结束。默认值为 20 分钟。

如果您想要将超时的时间间隔设置为比默认值更短或更长，可以使用 **Timeout** 属性。

下面的实例设置了一个 5 分钟的超时时间间隔：

<%  
Session.Timeout=5  
%>

要立即结束 session，请使用 **Abandon** 方法：

<%  
Session.Abandon  
%>

**注释：**使用 session 时主要的问题是它们该在何时结束。我们不会知道用户最近的请求是否是最后的请求。因此我们不清楚该让 session "存活"多久。为某个空闲的 session 等待太久会耗尽服务器的资源。然而如果 session 被过早地删除，用户就不得不一遍又一遍地重新开始，这是因为服务器已经删除了所有的信息。寻找合适的超时间隔时间是很困难的！

IMG_256**提示：**在 session 变量中仅存储少量的数据！

## 存储和取回 Session 变量

Session 对象最大的优点是可在其中存储变量，以供后续的网页读取，其应用范围是很广的。

下面的实例把 "Donald Duck" 赋值给名为 *username* 的 Session 变量，并把 "50" 赋值给名为 *age* 的 Session 变量：

<%  
Session("username")="Donald Duck"  
Session("age")=50  
%>

当值被存储在 session 变量中，它就能被 ASP 应用程序中的任何页面使用：

Welcome <%Response.Write(Session("username"))%>

上面这行代码返回的结果是: "Welcome Donald Duck"。

您也可以在 Session 对象中存储用户参数，然后通过访问这些参数来决定向用户返回什么页面。

下面的实例规定，假如用户使用低显示器分辨率，则返回纯文本版本的页面：

<%If Session("screenres")="low" Then%>  
This is the text version of the page  
<%Else%>  
This is the multimedia version of the page  
<%End If%>

## 移除 Session 变量

Contents 集合包含所有的 session 变量。

可通过 Remove 方法来移除 session 变量。

在下面的实例中，如果 session 变量 "age" 的值小于 18，则移除 session 变量 "sale"：

<%  
If Session.Contents("age")<18 then  
Session.Contents.Remove("sale")  
End If  
%>

如需移除 session 中的所有变量，请使用 RemoveAll 方法：

<%  
Session.Contents.RemoveAll()  
%>

## 遍历 Contents 集合

Contents 集合包含所有的 session 变量。您可以通过遍历 Contents 集合，来查看其中存储的变量：

<%  
Session("username")="Donald Duck"  
Session("age")=50  
  
dim i  
For Each i in Session.Contents  
Response.Write(i & "<br>")  
Next  
%>

结果：

username  
age

如果您不知道 Contents 集合中的项目数量，您可以使用 Count 属性：

<%  
dim i  
dim j  
j=Session.Contents.Count  
Response.Write("Session variables: " & j)  
For i=1 to j  
Response.Write(Session.Contents(i) & "<br>")  
Next  
%>

结果：

Session variables: 2  
Donald Duck  
50

## 遍历 StaticObjects 集合

您可以通过遍历 StaticObjects 集合，来查看存储在 Session 对象中的所有对象的值：

<%  
dim i  
For Each i in Session.StaticObjects  
Response.Write(i & "<br>")  
Next  
%>

# ASP Application 对象

在一起协同工作以完成某项任务的一组 ASP 文件称为一个应用程序。

## Application 对象

Web 上的一个应用程序可以是一组 ASP 文件。这些 ASP 文件一起协同工作来完成某项任务。ASP 中的 Application 对象用于把这些文件捆绑在一起。

Application 对象用于存储和访问来自任何页面的变量，类似于 Session 对象。不同之处在于，所有的用户分享一个 Application 对象，而 Session 对象和用户的关系是一一对应的。

Application 对象存有会被应用程序中的许多页面使用的信息（比如数据库连接信息）。可以从任何的页面访问这些信息。同时您也可以在一个地方改变这些信息，随后这些改变会自动反映在所有的页面上。

## 存储和取回 Application 变量

Application 变量可被应用程序中的任何页面访问和改变。

您可以在 "Global.asa" 中创建 Application 变量，如下所示：

<script language="vbscript" runat="server">  
  
Sub Application\_OnStart  
application("vartime")=""  
application("users")=1  
End Sub  
  
</script>

在上面的实例中，我们创建了两个 Application 变量："vartime" 和 "users"。

您可以访问 Application 变量的值，如下所示：

There are  
<%  
Response.Write(Application("users"))  
%>  
active connections.

## 遍历 Contents 集合

Contents 集合包含着所有的 application 变量。您可以通过遍历 Contents 集合，来查看其中存储的变量：

<%  
dim i  
For Each i in Application.Contents  
Response.Write(i & "<br>")  
Next  
%>

如果您不知道 Contents 集合中的项目数量，您可以使用 Count 属性：

<%  
dim i  
dim j  
j=Application.Contents.Count  
For i=1 to j  
Response.Write(Application.Contents(i) & "<br>")  
Next  
%>

## 遍历 StaticObjects 集合

您可以通过遍历 StaticObjects 集合，来查看存储在 Application 对象中的所有对象的值：

<%  
dim i  
For Each i in Application.StaticObjects  
Response.Write(i & "<br>")  
Next  
%>

## 锁定和解锁

您可以使用 "Lock" 方法来锁定应用程序。当应用程序锁定后，用户们就无法改变 Application 变量了（除了正在访问 Application 变量的用户）。您还可以使用 "Unlock" 方法来解锁应用程序。这个方法会移除对 Application 变量的锁定：

<%  
Application.Lock  
'do some application object operations  
Application.Unlock  
%>

# ASP 引用文件

## #include 指令

通过使用 #include 指令，您可以在服务器执行 ASP 文件之前，把另一个 ASP 文件的内容插入到这个 ASP 文件中。

#include 指令用于创建函数、页眉、页脚或者其他多个页面上需要重复使用的元素等。

## 如何使用 #include 指令

这里有一个名为 "mypage.asp" 的文件：

<!DOCTYPE html>  
<html>  
<body>  
<h3>Words of Wisdom:</h3>  
<p><!--#include file="wisdom.inc"--></p>  
<h3>The time is:</h3>  
<p><!--#include file="time.inc"--></p>  
</body>  
</html>

这是 "wisdom.inc" 文件：

"One should never increase, beyond what is necessary,  
the number of entities required to explain anything."

这是 "time.inc" 文件：

<%  
Response.Write(Time)  
%>

如果您在浏览器中查看源代码，它将如下所示：

<!DOCTYPE html>  
<html>  
<body>  
<h3>Words of Wisdom:</h3>  
<p>"One should never increase, beyond what is necessary,  
the number of entities required to explain anything."</p>  
<h3>The time is:</h3>  
<p>11:33:42 AM</p>  
</body>  
</html>

## 引用文件的语法

如需在 ASP 页面中引用文件，请把 #include 指令放在注释标签中：

<!--#include virtual="somefilename"-->  
  
or  
  
<!--#include file ="somefilename"-->

### **Virtual 关键词**

请使用关键词 virtual 来指示以虚拟目录开始的路径。

如果一个名为 "header.inc" 的文件位于虚拟目录 /html 中，下面这行代码会插入 "header.inc" 文件中的内容：

<!-- #include virtual ="/html/header.inc" -->

### **File 关键词**

请使用关键词 file 来指示一个相对路径。相对路径是以含有引用文件的目录开始的。

如果您在 html 目录中有一个文件，且 "header.inc" 文件位于 html 头部，下面这行代码将在您的文件中插入 "header.inc" 文件中的内容：

<!-- #include file ="headersheader.inc" -->

请注意被引用文件 (headersheader.inc) 的路径是相对于引用文件的。如果包含 #include 声明的文件不在 html 目录中，这个声明就不会生效。

## 提示和注释

在上面的一部分中，我们已经使用 ".inc" 来作为被被引用文件的文件扩展名。请注意：如果用户尝试直接浏览 INC 文件，这个文件中内容将会被显示出来。如果您的被引用文件中的内容包含机密的信息或者是您不想让任何用户看到的信息，那么最好还是使用 ".asp" 作为扩展名。ASP 文件中的源代码被编译后是不可见的。被引用的文件也可引用其他文件，同时一个 ASP 文件可以对同一个文件引用多次。

**重要事项：**在脚本执行前，被引用的文件就会被处理和插入。下面的脚本无法执行，这是由于 ASP 会在为变量赋值之前执行 #include 指令：

<%  
fname="header.inc"  
%>  
<!--#include file="<%fname%>"-->

您不能在脚本分隔符之间包含文件引用。下面的脚本无法执行：

<%  
For i = 1 To n  
<!--#include file="count.inc"-->  
Next  
%>

但是这段脚本可以执行：

<% For i = 1 to n %>  
<!--#include file="count.inc" -->  
<% Next %>

# ASP Global.asa 文件

## Global.asa 文件

Global.asa 文件是一个可选的文件，它可包含被 ASP 应用程序中每个页面访问的对象、变量和方法的声明。

所有合法的浏览器脚本（JavaScript、VBScript、JScript、PerlScript 等等）都能在 Global.asa 中使用。

Global.asa 文件只能包含下列内容：

* Application 事件
* Session 事件
* <object> 声明
* TypeLibrary 声明
* #include 指令

**注释：**Global.asa 文件必须存放在 ASP 应用程序的根目录中，而且每个应用程序只能有一个 Global.asa 文件。

## Global.asa 中的事件

在 Global.asa 中，您可以告诉 application 和 session 对象当 application/session 开始时做什么，当 application/session 结束时做什么。完成这项任务的代码被放置在事件句柄中。Global.asa 文件能包含四种类型的事件：

**Application\_OnStart** - 此事件会在第一个用户调用 ASP 应用程序的第一个页面时发生。此事件会在 Web 服务器重启或者 Global.asa 文件被编辑之后发生。"Session\_OnStart" 事件会在此事件发生之后立即发生。

**Session\_OnStart** - 此事件会在每当新用户请求他（她）在 ASP 应用程序中的第一个页面时发生。

**Session\_OnEnd** - 此事件会在每当用户结束 session 时发生。在规定的时间（默认的时间为 20 分钟）内如果用户没有请求任何页面，用户 session 就会结束。

**Application\_OnEnd** - 此事件会在最后一个用户结束其 session 之后发生。典型的情况是，此事件会在 Web 服务器停止时发生。这个子程序用于在应用程序停止后清除设置，比如删除记录或者向文本文件中写入信息。

**注释：**由于我们无法在 Global.asa 文件中使用 ASP 的脚本分隔符 (<% 和 %>) 插入脚本，我们需要把子例程放置在 HTML 的 <script> 元素内部。

## <object> 声明

可通过使用 <object> 标签在 Global.asa 文件中创建带有 session 或者 application 作用域的对象。

**注释：**<object> 标签应位于 <script> 标签外部！

### **语法**

<object runat="server" scope="*scope*" id="*id*" {progid="*progID*"|classid="*classID*"}>  
....  
</object>

|  |  |
| --- | --- |
| **参数** | **描述** |
| scope | 设置对象（Session 或 Application）的作用域。 |
| id | 为对象指定一个唯一的 id。 |
| ProgID | 与 ClassID 关联的 id。ProgID 的格式是：[Vendor.]Component[.Version]。  ProgID 或 ClassID 必需被指定。 |
| ClassID | 为 COM 类对象指定一个唯一的 id。  ProgID 或 ClassID 必需被指定。 |

### **实例**

第一个实例通过使用 ProgID 参数创建了一个名为 "MyAd" 的 session 作用域对象：

<object runat="server" scope="session" id="MyAd" progid="MSWC.AdRotator">  
</object>

第二个实例通过使用 ClassID 参数创建了一个名为 "MyConnection" 的 application 作用域对象：

<object runat="server" scope="application" id="MyConnection"  
classid="Clsid:8AD3067A-B3FC-11CF-A560-00A0C9081C21">  
</object>

在 Global.asa 文件中声明的对象可被应用程序中的任何脚本使用：

GLOBAL.ASA:  
  
<object runat="server" scope="session" id="MyAd" progid="MSWC.AdRotator">  
</object>  
  
您可以从 ASP 应用程序中的任意页面引用 "MyAd" 对象：  
  
某个 .ASP 文件：  
  
<%=MyAd.GetAdvertisement("/banners/adrot.txt")%>

## TypeLibrary 声明

TypeLibrary（类型库）是一个容器，其中装有对应于 COM 对象的 DLL 文件。通过在 Global.asa 文件中包含对 TypeLibrary 的调用，可以访问 COM 对象的常量，同时 ASP 代码也能更好地报告错误。如果您的 Web 应用程序依赖于已在类型库中声明的数据类型的 COM 对象，您可以在 Global.asa 中对类型库进行声明。

### **语法**

<!--METADATA TYPE="TypeLib"  
file="*filename*" uuid="*id*" version="*number*" lcid="*localeid*"  
-->

|  |  |
| --- | --- |
| **参数** | **描述** |
| file | 规定指向类型库的绝对路径。  file 参数或者 uuid 参数，两者缺一不可。 |
| uuid | 规定了类型库的唯一的标识符。  file 参数或者 uuid 参数，两者缺一不可。 |
| version | 可选。用于选择版本。如果没有找到需要的版本，将使用最接近的版本。 |
| lcid | 可选。用于类型库的地区标识符。 |

### **错误值**

服务器会返回以下的错误消息之一：

|  |  |
| --- | --- |
| **错误代码** | **描述** |
| ASP 0222 | 无效的类型库规范 |
| ASP 0223 | 没有找到类型库 |
| ASP 0224 | 无法加载类型库 |
| ASP 0225 | 无法包装类型库 |

**注释：**METADATA 标签可出现在 Global.asa 文件中的任何位置（在 <script> 标签的内外皆可）。然而，我们还是推荐将 METADATA 标签放置于 Global.asa 文件的顶部。

## 限定

关于可以在 Global.asa 文件中引用的内容的限定：

* 您无法显示 Global.asa 文件中的文本。此文件无法显示信息。
* 您只能在 Application\_OnStart 和 Application\_OnEnd 子例程中使用 Server 和 Application 对象。在 Session\_OnEnd 子例程中，您可以使用 Server、Application 和 Session 对象。在 Session\_OnStart 子例程中，您可以使用任何内建的对象。

## 如何使用子例程

Global.asa 常用于初始化变量。

下面的实例演示了如何检测访客首次到达 Web 站点的确切时间。时间存储在名为 "started" 的 Session 对象中，并且 "started" 变量的值可被应用程序中的任何 ASP 页面访问：

<script language="vbscript" runat="server">  
sub Session\_OnStart  
Session("started")=now()  
end sub  
</script>

Global.asa 也可用于控制页面访问。

下面的实例演示了如何把每个新的访客重定向到另一个页面，在这个例子中会定向到一个名为 "newpage.asp" 的页面：

<script language="vbscript" runat="server">  
sub Session\_OnStart  
Response.Redirect("newpage.asp")  
end sub  
</script>

您可以在 Global.asa 文件中包含函数。

在下面的实例中，当 Web 服务器启动时，Application\_OnStart 子例程也会启动。然后，Application\_OnStart 子例程会调用另一个名为 "getcustomers" 的子例程。"getcustomers" 子例程会打开一个数据库，然后从 "customers" 表中取回一个记录集。此记录集会赋值给一个数组，在不查询数据库的情况下，任何 ASP 页面都能够访问这个数组：

<script language="vbscript" runat="server">  
  
sub Application\_OnStart  
getcustomers  
end sub  
  
sub getcustomers  
set conn=Server.CreateObject("ADODB.Connection")  
conn.Provider="Microsoft.Jet.OLEDB.4.0"  
conn.Open "c:/webdata/northwind.mdb"  
set rs=conn.execute("select name from customers")  
Application("customers")=rs.GetRows  
rs.Close  
conn.Close  
end sub  
  
</script>

## Global.asa 实例

在这个实例中，我们将创建一个计算当前访客数量的 Global.asa 文件。

* 当服务器启动时，Application\_OnStart 设置 Application 变量 "visitors" 的值为 0。
* 每当有新的访客来访时，Session\_OnStart 子例程就会给变量 "visitors" 加 1。
* 每当 Session\_OnEnd 子例程被触发时，该子例程就会从变量 "visitors" 减 1。

Global.asa 文件：

<script language="vbscript" runat="server">  
  
Sub Application\_OnStart  
Application("visitors")=0  
End Sub  
  
Sub Session\_OnStart  
Application.Lock  
Application("visitors")=Application("visitors")+1  
Application.UnLock  
End Sub  
  
Sub Session\_OnEnd  
Application.Lock  
Application("visitors")=Application("visitors")-1  
Application.UnLock  
End Sub  
  
</script>

在 ASP 文件中，显示当前访客的数量：

<!DOCTYPE html>  
<html>  
<head>  
</head>  
<body>  
<p>There are <%response.write(Application("visitors"))%> online now!</p>  
</body>  
</html>

# ASP 使用 CDOSYS 发送电子邮件

CDOSYS 是 ASP 中的内建组件。此组件用于通过 ASP 发送电子邮件。

## 使用 CDOSYS 发送电子邮件

CDO (Collaboration Data Objects) 是一项微软的技术，设计目的是用来简化通讯应用程序的创建。

CDOSYS 是 ASP 中的内建组件。我们将向您演示如何通过 ASP 使用该组件来发送电子邮件。

## CDONTs 怎么样？

微软已经在 Windows 2000、Windows XP 和 Windows 2003 中淘汰了 CDONTs。如果您已经在您的 ASP 应用程序中使用 CDONTs，那么您需要更新代码，并使用新的 CDO 技术。