

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。

[现在开始学习 ASP！](#)

通过实例学习 ASP

在本站的 ASP 教程中包含了 100 多个实例。

我们的 ASP 在线实例让您能够更简单的学习 ASP，实例中包含了 ASP 的源码及运行结果。

实例

```
<!DOCTYPE html>
<html>
<body>
<%
response.write("My first ASP script!")
%>
</body>
</html>
```

[演示实例 »](#)

点击"演示实例"按钮查看在线实例运行结果。

ASP 参考手册

在菜鸟教程，我们为您提供了完整的 ASP 参考手册，其中包括内建对象和组件，以及它们的属性和方法。

[ASP 参考手册](#)

ASP 实例

通过实例来学习！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 实例！](#)

□ [点我分享笔记](#)

[反馈/建议](#)

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。

您应当具备的基础知识

在继续学习之前，您需要对以下知识有基本的了解：

HTML / XHTML

脚本语言，比如 JavaScript 或者 VBScript

如果您希望首先学习这些项目，请在我们的 [首页](#) 访问这些教程。

ASP 是什么？

ASP 代表 **Active Server Pages**（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 **Internet Information Services**（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

☐ 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 教程

在自己的 PC 上运行 ASP ☐

☐ 点我分享笔记

反馈/建议

Copyright © 2013-2018 菜鸟教程 runoob.com All Rights Reserved. 备案号：闽ICP备15012807号-1



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

☐ ASP 简介

ASP 语法 ☐

在自己的 PC 上运行 ASP

您可以在自己的 PC 上运行 ASP 。

把自己的 Windows PC 作为 Web 服务器

如果您安装了 IIS 或 PWS，就可以把自己的 PC 配置为一台 Web 服务器。

IIS 或 PWS 可以把您的计算机转变为 Web 服务器。

微软的 IIS 和 PWS 是免费的 Web 服务器组件。

IIS - Internet Information Server（Internet 信息服务）

IIS 是一个基于因特网的服务的集合，由微软开发，在微软 Windows 平台上使用。

Windows 2000、XP、Vista 以及 Windows 7 均提供 IIS。Windows NT 也可用 IIS。

IIS 很容易安装，是开发和测试 web 应用程序的理想工具。

PWS - Personal Web Server

PWS 用于更老的 Windows 系统，比如 Windows 95、98 以及 NT。

PWS 很容易安装，可用于开发和测试包含 ASP 的 Web 应用程序。

我们不推荐使用 PWS，除非是用于培训。它已经过时，并存在安全问题。

Windows Web 服务器版本

Windows 7（所有版本）自带 IIS 7.5

Windows Vista 商业，企业和旗舰版自带 IIS 7

Windows Vista 高级家庭版自带 IIS 7

Windows Vista 家庭版不支持 PWS 或 IIS

Windows XP 专业版自带 IIS 5.1

Windows XP 家庭版不支持 IIS 或 PWS

Windows 2000 专业版自带 IIS 5.0

Windows NT 专业版自带 IIS 3，同时还支持 IIS 4

Windows NT 工作站支持 PWS 和 IIS 3

Windows ME 不支持 PWS 或 IIS

Windows 98 自带 PWS

Windows 95 支持 PWS

如何在 Windows 7 和 Windows Vista 上安装 IIS

请根据以下几个步骤来安装 IIS：

1. 从开始菜单打开控制面板
2. 双击"程序和功能"
3. 点击"打开或关闭 Windows 功能"
4. 选择"Internet 信息服务(IIS)"的复选框，然后点击确定

在您安装完成 IIS 之后，请确保安装所有补丁包（运行 Windows 更新）。

如何在 Windows XP 和 Windows 2000 上安装 IIS

请根据以下几个步骤来安装 IIS：

1. 在开始菜单上，点击设置，并选择控制面板
2. 双击"添加/删除程序"
3. 点击"添加/删除 Windows 组件"
4. 点击"Internet 信息服务(IIS)"
5. 点击"详细信息"
6. 选择"万维网服务"复选框，点击确定
7. 在 Windows 组件中，单击"下一步"安装 IIS

在您安装完成 IIS 之后，请确保安装所有补丁包（运行 Windows 更新）。

测试您的安装

在您安装完 IIS 或 PWS 之后，按照下面的步骤测试是否安装成功：

1. 在您的硬盘中查找名为 **Inetpub** 的文件夹
2. 打开 **Inetpub** 文件夹，找到名为 **wwwroot** 的文件夹
3. 在 **wwwroot** 下创建一个新文件夹，比如 "MyWeb"
4. 使用文本编辑器编写几行 ASP 代码，将这个文件取名为 "test1.asp" 保存在 "MyWeb" 文件夹中
5. 确保您的 Web 服务器正在运行，使用下面的方法确认它的运行状态：进入控制面板，然后是管理工具，然后双击"IIS 管理器"图标。
6. 打开您的浏览器，在地址栏键入 "http://localhost/MyWeb/test1.asp"，就可以看到您的第一个 ASP 页面了。

注释：在您的开始菜单或者任务栏中查找 IIS (或 PWS) 符号。程序提供了开启和停止 Web 服务器，启用和禁用 ASP，以及其他更多的功能。

如何在 Windows 95、Windows 98 和 Windows NT 上安装 PWS

Windows 98: 打开您的 Windows CD 上的 **Add-ons** 文件夹，找到 **PWS** 文件夹并运行其中的 **setup.exe** 文件来安装 PWS。

Windows 95 或 Windows NT: 从微软的站点下载 "Windows NT 4.0 Option Pack" 来安装 PWS。

根据前面的描述测试您的安装。

如何在 Windows Server 2003 上安装 IIS

1. 当您启动 Windows Server 2003 后，您会看到服务器管理向导
2. 如果向导没有显示，请打开管理工具，然后选择配置您的服务器向导
3. 在向导中，点击 **Add or Remove a Role**，点击下一步
4. 选择自定义配置，点击下一步
5. 选择应用程序服务器角色，点击下一步
6. 选择启用 **ASP.NET**，点击下一步
7. 现在，向导会请求 **Server 2003 CD**。插入 CD 后继续运行向导直到完成，然后点击完成按钮
8. 向导现在应该显示"应用程序服务器角色已安装"
9. 点击管理此应用程序服务器打开应用程序服务器管理控制台 (MMC)
10. 展开 **Internet 信息服务 (IIS)管理器**，然后展开您的服务器，然后是站点文件夹
11. 您会看到默认的网站，并且它的状态应该是运行中
12. IIS 正在运行中！
13. 在 **Internet 信息服务 (IIS)管理器** 中点击 **Web 服务扩展** 文件夹
14. 这里，您将看到 **Active Server Pages** 是被禁止的（这是 IIS 6 的默认配置）
15. 选中 **Active Server Pages**，并且点击允许按钮
16. 这样 ASP 就被激活了！

[☐ ASP 简介](#)

ASP 语法 [☐](#)

[☐ 点我分享笔记](#)

反馈/建议

Copyright © 2013-2018 菜鸟教程 [runoob.com](#) All Rights Reserved. 备案号：闽ICP备15012807号-1



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

[☐ 在自己的 PC 上运行 ASP](#)

ASP 变量 [☐](#)

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":

实例

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

[演示实例 »](#)

还有一种 `response.write` 命令的简写方法。下面的实例也是向浏览器传送了一段文本：**"Hello World"**：

实例

```
<!DOCTYPE html>
<html>
<body>
<%
="Hello World!"
%>
</body>
</html>
```

[演示实例 »](#)

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。

其他的脚本语言

ASP 与 **VBScript** 和 **JScript**（**JScript** 是微软的 **JavaScript** 实现）的配合是原生性的。如果您想要使用其他语言编写脚本，比如 **PERL**、**REXX** 或者 **Python**，您必须安装相应的脚本引擎。

更多实例

[给文本添加一些 HTML 标签](#)

ASP 变量

变量用于存储信息。

☐

尝试一下 - 实例

[声明变量](#)

变量用于存储信息。本例演示如何声明变量，为变量赋值，并在程序中使用这个变量。

[声明数组](#)

数组用于存储一系列相关的数据项目。本例演示如何声明一个存储名字的数组。

[循环生成 HTML 标题](#)

本例演示如何循环生成 6 个不同的 HTML 标题。

[使用 Vbscript 制作基于时间的问候语](#)

本例演示如何根据服务器时间向用户显示不同的消息。

[使用 JavaScript 制作基于时间的问候语](#)

本例同上，演示如何根据服务器时间向用户显示不同的消息，只是语法不同而已。

变量的生存期

在子程序外声明的变量可被 ASP 文件中的任何脚本访问和修改。

在子程序中声明的变量在每次子程序执行时被创建和撤销。子程序外的脚本无法访问和修改该变量。

如需声明供多个 ASP 文件使用的变量，请将变量声明为 session 变量或者 application 变量。

Session 变量

Session 变量用于存储单一用户的信息，并且对一个应用程序中的所有页面均有效。存储于 Session 中的典型信息有姓名、id 和参数。

Application 变量

Application 变量同样对一个应用程序中的所有页面均有效。Application 变量用于存储一个特定的应用程序中所有用户的信息。

[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)[ASP 变量](#)[ASP 表单](#)

ASP 子程序

在 ASP 中, 您可通过 VBScript 调用 JavaScript 子程序, 反之亦然。

子程序

ASP 源代码可包含子程序和函数:

实例

```
<!DOCTYPE html>
<html>
<head>
<%
sub vbproc(num1,num2)
response.write(num1*num2)
end sub
%>
</head>
<body>

<p>Result: <%call vbproc(3,4)%></p>

</body>
</html>
```

[演示实例 »](#)

将 `<%@ language="language" %>` 这一行写在 `<html>` 标签的上面, 就可以使用另一种脚本语言来编写子程序或者函数:

实例

```
<%@ language="javascript" %>
<!DOCTYPE html>
<html>
<head>
<%
function jsproc(num1,num2)
{
Response.Write(num1*num2)
}
%>
</head>
<body>

<p>Result: <%jsproc(3,4)%></p>


</body>
</html>
```

[演示实例 »](#)

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。



更多实例

[使用 VBScript 调用子程序](#)

本例演示如何在一个 **ASP** 文件中调用 **VBScript** 子程序和 **JavaScript** 子程序。

 ASP 变量

ASP 表单 

 [点我分享笔记](#)

反馈/建议




 ASP 程序

ASP Cookies 

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。



尝试一下 - 实例

[使用 method="get" 的表单](#)

本例演示如何使用 **Request.QueryString** 命令与用户进行交互。

[使用 method="post" 的表单](#)

本例演示如何使用 **Request.Form** 命令与用户进行交互。

[使用单选按钮的表单](#)

本例演示如何使用 **Request.Form** 命令通过单选按钮与用户进行交互。

用户输入

Request 对象可用于从表单取回用户信息。

HTML 表单实例

```
<form method="get" action="simpleform.asp">
First Name: <input type="text" name="fname"><br>
Last Name: <input type="text" name="lname"><br><br>
<input type="submit" value="Submit">
</form>
```

用户输入可通过 **Request.QueryString** 或 **Request.Form** 命令取回。

Request.QueryString

Request.QueryString 命令用于收集使用 method="get" 的表单中的值。

使用 GET 方法从表单传送的信息对所有的用户都是可见的（出现在浏览器的地址栏），并且对所发送信息的量也有限制。

如果用户在上面的 HTML 表单中输入 "Bill" 和 "Gates"，发送至服务器的 URL 会类似这样：

```
http://www.w3cschool.cc/simpleform.asp?fname=Bill&lname=Gates
```

假设 "simpleform.asp" 文件包含下面的 ASP 脚本：

```
<body>
Welcome
<%
response.write(request.querystring("fname"))
response.write(" " & request.querystring("lname"))
%>
</body>
```

浏览器将把文档的 body 部分显示如下：

```
Welcome Bill Gates
```

Request.Form

Request.Form 命令用于收集使用 method="post" 的表单中的值。

使用 POST 方法从表单传送的信息对用户是不可见的，并且对所发送信息的量没有限制。

如果用户在上面的 HTML 表单中输入 "Bill" 和 "Gates"，发送至服务器的 URL 会类似这样：

```
http://www.w3cschool.cc/simpleform.asp
```

假设 "simpleform.asp" 文件包含下面的 ASP 脚本：

```
<body>
Welcome
<%
response.write(request.form("fname"))
response.write(" " & request.form("lname"))
%>
</body>
```

浏览器将把文档的 body 部分显示如下：

```
Welcome Bill Gates
```

表单验证

只要有可能，就尽量在浏览器上对用户的输入进行验证（通过客户端脚本）。浏览器的验证速度更快，并可以减少服务器的负载。

如果用户输入会保存到数据库中，那么您应该考虑使用服务器端验证。有一种在服务器端验证表单的好方法，就是将（验证过的）表单传回表单页面，而不是转至不同的页面。用户随后就可以在同一个页面中得到错误的信息。这样做更易于用户发现错误。



ASP Cookies

cookie 常用于识别用户。

尝试一下 - 实例

Welcome cookie

本例演示如何创建 Welcome cookie。

Cookie 是什么？

cookie 常用于识别用户。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。

如何创建 Cookie？

"Response.Cookies" 命令用于创建 cookie。

注释：Response.Cookies 命令必须出现在 <html> 标签之前。

在下面的实例中，我们将创建一个名为 "firstname" 的 cookie，并将其赋值为 "Alex"：

```
<%
Response.Cookies("firstname")="Alex"
%>
```

向 cookie 分配属性也是可以的，比如设置 cookie 的失效时间：

```
<%
Response.Cookies("firstname")="Alex"
Response.Cookies("firstname").Expires=#May 10,2012#
%>
```

如何取回 Cookie 的值？

"Request.Cookies" 命令用于取回 cookie 的值。

在下面的实例中，我们取回了名为 "firstname" 的 cookie 的值，并把值显示到了页面上：

```
<%
fname=Request.Cookies("firstname")
response.write("Firstname=" & fname)
%>
```

输出： Firstname=Alex

带有键的 Cookie

如果一个 cookie 包含多个值的集合，我们就可以说 cookie 带有键（Keys）。

在下面的实例中，我们将创建一个名为 "user" 的 cookie 集合。"user" cookie 带有包含用户信息的键：

```
<%
Response.Cookies("user")("firstname")="John"
Response.Cookies("user")("lastname")="Smith"
Response.Cookies("user")("country")="Norway"
Response.Cookies("user")("age")="25"
%>
```

读取所有的 Cookie

请阅读下面的代码：

```
<%
Response.Cookies("firstname")="Alex"
Response.Cookies("user")("firstname")="John"
Response.Cookies("user")("lastname")="Smith"
Response.Cookies("user")("country")="Norway"
Response.Cookies("user")("age")="25"
%>
```

假设您的服务器将上面所有的 cookie 传给了某个用户。

现在，我们需要读取这些传给某个用户的所有的 cookie。下面的实例向您演示了如何做到这一点（请注意，下面的代码通过 HasKeys 属性检查 cooki

e 是否带有键)：

```
<!DOCTYPE html>
<html>
<body>

<%
dim x,y
for each x in Request.Cookies
response.write("<p>")
if Request.Cookies(x).HasKeys then
for each y in Request.Cookies(x)
response.write(x & ":" & y & "=" & Request.Cookies(x)(y))
response.write("<br>")
next
else
Response.Write(x & "=" & Request.Cookies(x) & "<br>")
end if
response.write "</p>"
next
%>

</body>
</html>
```

输出：

firstname=Alex
user:firstname=John
user:lastname=Smith
user:country=Norway
user:age=25

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>")
%>
```

[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)☐ ASP CookiesASP Application 对象 ☐

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** 被传送到用户的计算机上，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。寻找合适的超时时间间隔是很困难的！

☐ **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 Cookies](#)

[ASP Application 对象](#) ☐

[☐ 点我分享笔记](#)

[反馈/建议](#)

Copyright © 2013-2018 菜鸟教程 [runoob.com](#) All Rights Reserved. 备案号：闽ICP备15012807号-1



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

[☐ ASP Session 对象](#)

[ASP 引用文件](#) ☐

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 Application 对象](#)

ASP Global.asa [☐](#)

[☐ 点我分享笔记](#)

反馈/建议

Copyright © 2013-2018 菜鸟教程 [runoob.com](#) All Rights Reserved. 备案号：闽ICP备15012807号-1



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

[☐ ASP 引用文件](#)

ASP 使用 CDOSYS 发送电子邮件 [☐](#)

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。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。

一个 Global.asa 文件可能如下所示：

```
<script language="vbscript" runat="server">

sub Application_OnStart
'some code
end sub

sub Application_OnEnd
'some code
end sub

sub Session_OnStart
'some code
end sub

sub Session_OnEnd
'some code
end sub

</script>
```

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>
```

[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)[ASP Global.asa](#)ASP Response 对象 [□](#)

ASP 使用 CDO.SYS 发送电子邮件

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

使用 CDO.SYS 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。

CDO.SYS 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。

使用 CDO.SYS 的实例

发送文本电子邮件:

```
<%
Set myMail=CreateObject("CDO.Message")
myMail.Subject="Sending email with CDO"
myMail.From="mymail@mydomain.com"
myMail.To="someone@somedomain.com"
myMail.TextBody="This is a message."
myMail.Send
set myMail=nothing
%>
```

发送带有 Bcc 和 CC 字段的文本电子邮件:

```
<%
Set myMail=CreateObject("CDO.Message")
myMail.Subject="Sending email with CDO"
myMail.From="mymail@mydomain.com"
myMail.To="someone@somedomain.com"
myMail.Bcc="someoneelse@somedomain.com"
myMail.Cc="someoneelse2@somedomain.com"
myMail.TextBody="This is a message."
myMail.Send
set myMail=nothing
%>
```

发送 HTML 电子邮件:

```
<%
Set myMail=CreateObject("CDO.Message")
myMail.Subject="Sending email with CDO"
myMail.From="mymail@mydomain.com"
myMail.To="someone@somedomain.com"
myMail.HTMLBody = "<h1>This is a message.</h1>"
myMail.Send
set myMail=nothing
%>
```

发送一封内容为某个网站的某个网页的 HTML 电子邮件:

```
<%
Set myMail=CreateObject("CDO.Message")
myMail.Subject="Sending email with CDO"
myMail.From="mymail@mydomain.com"
myMail.To="someone@somedomain.com"
myMail.CreateMHTMLBody "http://www.w3cschool.cc/asp/"
myMail.Send
set myMail=nothing
%>
```

发送一封内容为您的计算机中某个文件的某个网页的 **HTML** 电子邮件：

```
<%
Set myMail=CreateObject("CDO.Message")
myMail.Subject="Sending email with CDO"
myMail.From="mymail@mydomain.com"
myMail.To="someone@somedomain.com"
myMail.CreateMHTMLBody "file://c:/mydocuments/test.htm"
myMail.Send
set myMail=nothing
%>
```

发送一封带有附件的文本电子邮件：

```
<%
Set myMail=CreateObject("CDO.Message")
myMail.Subject="Sending email with CDO"
myMail.From="mymail@mydomain.com"
myMail.To="someone@somedomain.com"
myMail.TextBody="This is a message."
myMail.AddAttachment "c:mydocumentstest.txt"
myMail.Send
set myMail=nothing
%>
```

使用远程服务器发送一封文本电子邮件：

```
<%
Set myMail=CreateObject("CDO.Message")
myMail.Subject="Sending email with CDO"
myMail.From="mymail@mydomain.com"
myMail.To="someone@somedomain.com"
myMail.TextBody="This is a message."
myMail.Configuration.Fields.Item _
("http://schemas.microsoft.com/cdo/configuration/sendusing")=2
'Name or IP of remote SMTP server
myMail.Configuration.Fields.Item _
("http://schemas.microsoft.com/cdo/configuration/smtpserver")="smtp.server.com"
'Server port
myMail.Configuration.Fields.Item _
("http://schemas.microsoft.com/cdo/configuration/smtpserverport")=25
myMail.Configuration.Fields.Update
myMail.Send
set myMail=nothing
%>
```

尝试一下 - 实例

[使用 ASP 写文本](#)

本例演示如何使用 **ASP** 来写文本。

[在 ASP 中使用 HTML 标签格式化文本](#)

本例演示如何使用 **ASP** 将文本和 **HTML** 标签结合起来。

[将用户重定向至一个不同的 URL](#)

本例演示如何将用户重定向至一个不同的 **URL**。

[显示随机的链接](#)

本例演示如何创建一个随机的链接。

[控制缓冲区](#)

本例演示如何控制缓冲区。

[清空缓冲区](#)

本例演示如何清空缓冲区。

[在处理过程中终止脚本并返回结果](#)

本例演示如何在处理过程中终止脚本。

[设置页面在失效前在浏览器中缓存时间](#)

本例演示如何规定页面在失效前在浏览器中的缓存时间。

[设置页面缓存在浏览器中的失效日期或时间](#)

本例演示如何规定页面缓存在浏览器中的失效时间日期或时间。

[检查用户是否仍然与服务器连接](#)

本例演示如何检查用户是否已与服务器断开。

[设置内容类型](#)

本例演示如何规定内容的类型。

[设置字符集名称](#)

本例演示如何规定字符集的名称。

Response 对象

ASP Response 对象用于从服务器向用户发送输出的结果。它的集合、属性和方法描述如下：

集合

集合	描述
Cookies	设置 cookie 的值。如果 cookie 不存在，则创建 cookie ，并设置指定的值。

属性

属性	描述
Buffer	规定是否缓冲页面的输出。
CacheControl	设置代理服务器是否可以缓存由 ASP 产生的输出。
Charset	将字符集的名称追加到 Response 对象中的 content-type 报头。
ContentType	设置 Response 对象的 HTTP 内容类型。
Expires	设置页面在失效前的浏览器缓存时间（分钟）。
ExpiresAbsolute	设置浏览器上页面缓存失效的日期和时间。
IsClientConnected	指示客户端是否已从服务器断开。
Pics	向 response 报头的 PICS 标签追加值。
Status	规定由服务器返回的状态行的值。

方法

方法	描述
AddHeader	向 HTTP 响应添加新的 HTTP 报头和值。
AppendToLog	向服务器日志条目的末端添加字符串。
BinaryWrite	在没有任何字符转换的情况下直接向输出写数据。
Clear	清除已缓冲的 HTML 输出。
End	停止处理脚本，并返回当前的结果。
Flush	立即发送已缓冲的 HTML 输出。
Redirect	把用户重定向到一个不同的 URL。
Write	向输出写指定的字符串。

☐ 点我分享笔记

反馈/建议



ASP Request 对象

Request 对象用于从访客那里获取信息。

QueryString 集合实例

[当用户点击链接时发送查询信息](#)

本例演示如何在链接中向页面发送查询信息，并在目标页面中取回这些信息（在本例中是同一页面）。

[QueryString 集合的简单应用](#)

本例演示如何使用 QueryString 集合从表单取回值。（此表单使用 GET 方法，这意味着所发送的信息对用户来说是可见的。）

[如何使用来自表单的信息](#)

本例演示如何使用从表单取回的值。此表单使用 GET 方法。

[来自表单的更多信息](#)

本例演示如果输入字段包含若干相同的名称，QueryString 集合会包含什么内容。它将展示如何使用 Count 关键词来对 "name" 属性进行计数。此表单使用 GET 方法。

Form 集合实例

Form 集合的简单应用

本例演示如何使用 **Form** 集合从表单取回值。（此表单使用 **POST** 方法，这意味着所发送的信息对用户来说是不可见的。）

如何使用来自表单的信息

本例演示如何使用从表单取回的值。此表单使用了 **POST** 方法。

来自表单的更多信息

本例演示如果输入字段包含若干相同的名称，**Form** 集合会包含什么内容。它将展示如何使用 **Count** 关键词来对 "name" 属性进行计数。此表单使用了 **POST** 方法。

带有单选按钮的表单

本例演示如何使用 **Form** 集合通过单选按钮与用户进行交互。此表单使用 **POST** 方法。

带有复选框的表单

本例演示如何使用 **Form** 集合通过复选框与用户进行交互。此表单使用 **POST** 方法。

实例

其他实例

获取服务器变量

本例演示如何取得访客的浏览器类型、IP 地址等信息。

创建 welcome cookie

本例演示如何创建一个 Welcome Cookie。

探测用户发送的字节总数

本例演示如何探测用户在 **Request** 对象中发送的字节总数。

Request 对象

当浏览器向服务器请求页面时，这个行为就被称为一个 **request**（请求）。**Request** 对象用于从用户那里获取信息。它的集合、属性和方法描述如下：

集合

集合	描述
ClientCertificate	包含了存储在客户证书中的所有的字段值。
Cookies	包含了 HTTP 请求中发送的所有的 cookie 值。
Form	包含了使用 post 方法由表单发送的所有的表单（输入）值。
QueryString	包含了 HTTP 查询字符串中所有的变量值。
ServerVariables	包含了所有的服务器变量值。

属性

属性	描述
TotalBytes	返回在请求正文中客户端发送的字节总数。

方法

方法	描述
BinaryRead	取回作为 post 请求的一部分而从客户端发送至服务器的数据，并把它存储在一个安全的数组中。



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

[☐ ASP Request 对象](#)

[ASP Session 对象](#) ☐

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** 对象用于把这些文件捆绑在一起。

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。**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。

Application 对象的集合、方法和事件的描述如下：

集合

集合	描述
Contents	包含所有通过脚本命令追加到应用程序中的项目。
StaticObjects	包含所有使用 HTML 的 <object> 标签追加到应用程序中的对象。

方法

方法	描述
Contents.Remove	从 Contents 集合中删除一个项目。
Contents.RemoveAll()	从 Contents 集合中删除所有的项目。
Lock	防止其他的用户修改 Application 对象中的变量。
Unlock	使其他的用户可以修改 Application 对象中的变量（在被 Lock 方法锁定之后）。

事件

事件	描述
Application_OnEnd	当所有用户的 session 都结束，并且应用程序结束时，此事件发生。
Application_OnStart	在第一个新的 session 被创建之前（即 Application 对象第一次被引用时），此事件会发生。

[☐ ASP Request 对象](#)

[ASP Session 对象](#) ☐

[☐ 点我分享笔记](#)

反馈/建议



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。



尝试一下 - 实例

[设置并返回 LCID](#)

本例演示 "LCID" 属性。该属性设置并返回一个指示位置或者地区的整数。类似于日期、时间和货币等内容都要根据位置或者地区来显示。

[返回 SessionID](#)

本例演示 "SessionID" 属性。该属性为每位用户返回一个唯一的 id。这个 id 由服务器生成。

[session 的超时](#)

本例演示 "Timeout" 属性。该属性设置并返回 session 的超时时间（分钟）。

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 对象的集合、属性、方法和事件的描述如下：

集合

集合	描述
Contents	包含所有通过脚本命令追加到 session 的条目。
StaticObjects	包含了所有使用 HTML 的 <object> 标签追加到 session 的对象。

属性

属性	描述
CodePage	规定显示动态内容时使用的字符集。
LCID	设置或返回指定位置或者地区的一个整数。诸如日期、时间好以及货币的内容会根据位置或者地区来显示。
SessionID	为每个用户返回一个唯一的 id。此 id 由服务器生成。
Timeout	设置或返回应用程序中的 Session 对象的超时时间（分钟）。

方法

方法	描述
----	----

Abandon	撤销一个用户的 session。
Contents.Remove	从 Contents 集合删除一个项目。
Contents.RemoveAll()	从 Contents 集合删除所有项目。

事件

事件	描述
Session_OnEnd	当一个会话结束时此事件发生。
Session_OnStart	当一个会话开始时此事件发生。

☐ ASP Application 对象

ASP Server 对象 ☐

☐ 点我分享笔记

反馈/建议



☐ ASP Session 对象

ASP ASPError 对象 ☐

ASP Server 对象

Server 对象用于访问有关服务器的属性和方法。

尝试一下 - 实例

[此文件最后被修改的时间是？](#)

探测文件的最后修改时间。

[打开并读取某个文本文件](#)

打开文件 "Textfile.txt" 以供读取。

[自制的点击计数器](#)

Server 对象

ASP **Server** 对象用于访问有关服务器的属性和方法。其属性和方法描述如下：

属性

属性	描述
ScriptTimeout	设置或返回在一段脚本终止前它所能运行时间（秒）的最大值。

方法

方法	描述
CreateObject	创建对象的实例。

Execute	从另一个 ASP 文件中执行一个 ASP 文件。
GetLastError()	返回可描述已发生错误状态的 ASPError 对象。
HTMLEncode	把 HTML 编码应用到某个指定的字符串。
MapPath	把一个指定的路径映射到一个物理路径。
Transfer	把一个 ASP 文件中创建的所有信息发送（传输）到另一个 ASP 文件。
URLEncode	把 URL 编码规则应用到指定的字符串。

[❏ ASP Session 对象](#)

[ASP ASPError 对象](#) ❏

[❏ 点我分享笔记](#)

[反馈/建议](#)

Copyright © 2013-2018 菜鸟教程 [runoob.com](#) All Rights Reserved. 备案号：闽ICP备15012807号-1



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

[❏ ASP Server 对象](#)

[ASP FileSystem 对象](#) ❏

ASP ASPError 对象

ASPError 对象用于显示在 ASP 文件的脚本中发生的错误信息。

ASPError 对象

ASPError 对象在 ASP 3.0 中生效，且在 IIS5 及更高版本中可用。

ASPError 对象用于显示在 ASP 文件的脚本中发生的任何错误的详细信息。

注释：当 `Server.GetLastError` 被调用时，ASPError 对象就会被创建，因此只能通过使用 `Server.GetLastError` 方法来访问错误信息。

ASPError 对象的属性描述如下（所有属性都是可读的）：

属性

属性	描述
ASPCode	返回由 IIS 生成的错误代码。
ASPDescription	返回错误的详细信息（如果错误和 ASP 相关）。
Category	返回错误来源。（错误是由 ASP、脚本语言还是对象引起的？）
Column	返回在出错文件中的列位置。
Description	返回关于错误的简短描述。

File	返回出错 ASP 文件的名称。
Line	返回错误所在的行数。
Number	返回关于错误的标准 COM 错误代码。
Source	返回错误所在行的实际的源代码。

[❏ ASP Server 对象](#)

ASP FileSystem 对象 [❏](#)

[❏ 点我分享笔记](#)

[反馈/建议](#)

Copyright © 2013-2018 菜鸟教程 [runoob.com](#) All Rights Reserved. 备案号：闽ICP备15012807号-1



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

[❏ ASP ASPError 对象](#)

ASP TextStream 对象 [❏](#)

ASP FileSystemObject 对象

FileSystemObject 对象用于访问服务器上的文件系统。



尝试一下 - 实例

[指定的文件存在吗？](#)

本例演示如何检查某个文件是否存在。

[指定的文件夹存在吗？](#)

本例演示如何检查某个文件夹是否存在。

[指定的驱动器存在吗？](#)

本例演示如何检查某个驱动器是否存在。

[取得某个指定驱动器的名称](#)

本例演示如何获取某个指定的驱动器的名称。

[取得某个指定路径的父文件夹的名称](#)

本例演示如何获取某个指定的路径的父文件夹的名称。

[取得文件名](#)

本例演示如何获取指定的路径中的最后一个成分的文件名。

[取得文件扩展名](#)

本例演示如何获取指定的路径中的最后一个成分的文件扩展名。

[取得文件或文件夹的基名称](#)

本例演示如何获取指定的路径中文件或者文件夹的基名称。

FileSystemObject 对象

FileSystemObject 对象用于访问服务器上的文件系统。

此对象可对文件、文件夹和目录路径进行操作。也可通过此对象获取文件系统的信息。

下面的代码会创建一个文本文件 (c:\test.txt)，然后向这个文件写一些文本：

```
<%
dim fs,fname
set fs=Server.CreateObject("Scripting.FileSystemObject")
set fname=fs.CreateTextFile("c:\test.txt",true)
fname.WriteLine("Hello World!")
fname.Close
set fname=nothing
set fs=nothing
%>
```

FileSystemObject 对象的属性和方法描述如下：

属性

属性	描述
Drives	返回本地计算机上所有驱动器对象的集合。

方法

方法	描述
BuildPath	将一个名称追加到已有的路径后。
CopyFile	从一个位置向另一个位置拷贝一个或多个文件。
CopyFolder	从一个位置向另一个位置拷贝一个或多个文件夹。
CreateFolder	创建新文件夹。
CreateTextFile	创建文本文件，并返回一个可以读取或者写入文件的 TextStream 对象。
DeleteFile	删除一个或者多个指定的文件。
DeleteFolder	删除一个或者多个指定的文件夹。
DriveExists	检查指定的驱动器是否存在。
FileExists	检查指定的文件是否存在。
FolderExists	检查指定的文件夹是否存在。
GetAbsolutePathName	针对指定的路径返回从驱动器根部起始的完整路径。
GetBaseName	返回指定文件或者文件夹的基名称。
GetDrive	返回指定路径中所对应的驱动器的 Drive 对象。
GetDriveName	返回指定的路径的驱动器名称。
GetExtensionName	返回在指定的路径中最后一个成分的文件扩展名。
GetFile	返回一个针对指定路径的 File 对象。
GetFileName	返回在指定的路径中最后一个成分的文件名或者文件夹名。
GetFolder	返回一个针对指定路径的 Folder 对象。
GetParentFolderName	返回在指定的路径中最后一个成分的父文件夹名称。
GetSpecialFolder	返回某些 Windows 的特殊文件夹的路径。
GetTempName	返回一个随机生成的文件或文件夹。
MoveFile	从一个位置向另一个位置移动一个或多个文件。

MoveFolder	从一个位置向另一个位置移动一个或多个文件夹。
OpenTextFile	打开文件，并返回一个用于访问此文件的 TextStream 对象。

[❏ ASP ASPError 对象](#)

[ASP TextStream 对象](#) ❏

[❏ 点我分享笔记](#)

反馈/建议

Copyright © 2013-2018 菜鸟教程 [runoob.com](#) All Rights Reserved. 备案号：闽ICP备15012807号-1



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

[❏ ASP FileSystem 对象](#)

[ASP Drive 对象](#) ❏

ASP **TextStream** 对象

TextStream 对象用于访问文本文件的内容。



尝试一下 - 实例

[读取文本文件](#)

本例演示如何从文本文件中读取内容。

[读取文本文件中的一个部分](#)

本例演示如何仅仅读取一个文本流文件的部分内容。

[读取文本文件中的一行](#)

本例演示如何从一个文本流文件中读取一行内容。

[读取文本文件的所有行](#)

本例演示如何从文本流文件中读取所有的行。

[略过文本文件中的一部分](#)

本例演示如何在读取文本流文件时跳过指定的字符数。

[略过文本文件中的一行](#)

本例演示如何在读取文本流文件时跳过一行。

[返回行数](#)

本例演示如何返回在文本流文件中的当前行号。

[取得列数](#)

本例演示如何取得在文件中当前字符的列号。

TextStream 对象

TextStream 对象用于访问文本文件的内容。

下面的代码会创建一个文本文件 (c:\test.txt)，然后向此文件写一些文本（变量 **f** 是 **TextStream** 对象的一个实例）：

```
<%  
dim fs,f  
set fs=Server.CreateObject("Scripting.FileSystemObject")  
set f=fs.CreateTextFile("c:\test.txt",true)
```

```
f.WriteLine("Hello World!")
f.Close
set f=nothing
set fs=nothing
%>
```

如需创建 **TextStream** 对象的一个实例，您可以使用 **FileSystemObject** 对象的 **CreateTextFile** 方法或者 **OpenTextFile** 方法，也可以使用 **File** 对象的 **OpenAsTextStream** 方法。

TextStream 对象的属性和方法描述如下：

属性

属性	描述
AtEndOfLine	如果文件指针正好位于 TextStream 文件中行尾标记的前面，则该属性值返回 True ；否则返回 False 。
AtEndOfStream	如果文件指针在 TextStream 文件末尾，则该属性值返回 True ；否则返回 False 。
Column	返回 TextStream 文件输入流中的当前字符位置的列号。
Line	返回 TextStream 文件中的当前行号。

方法

方法	描述
Close	关闭一个打开的 TextStream 文件。
Read	从一个 TextStream 文件中读取指定数量的字符并返回结果。
ReadAll	读取整个 TextStream 文件并返回结果。
ReadLine	从一个 TextStream 文件读取一整行（到换行符但不包括换行符）并返回结果。
Skip	当读取一个 TextStream 文件时跳过指定数量的字符。
SkipLine	当读取一个 TextStream 文件时跳过下一行。
Write	写入指定的文本到一个 TextStream 文件中。
WriteLine	写入指定的文本和换行符到一个 TextStream 文件中。
WriteBlankLines	写入指定数量的换行符到一个 TextStream 文件中。

[☐ ASP FileSystem 对象](#)

[ASP Drive 对象 ☐](#)

[☐ 点我分享笔记](#)

反馈/建议



[☐ ASP TextStream 对象](#)

[ASP File 对象 ☐](#)

ASP Drive 对象

Drive 对象用于返回关于本地磁盘驱动器或者网络共享驱动器的信息。



尝试一下 - 实例

[取得指定驱动器的总容量](#)

本例演示如何使用 **TotalSize** 属性来获得指定驱动器的总容量。

[取得指定驱动器的可用空间数](#)

本例演示如何首先创建一个 **FileSystemObject** 对象，然后使用 **AvailableSpace** 属性来获得指定驱动器的可用空间。

[取得指定驱动器的剩余空间容量](#)

本例演示如何使用 **FreeSpace** 空间属性来取得指定驱动器的剩余空间。

[取得指定驱动器的驱动器字母](#)

本例演示如何使用 **DriveLetter** 属性来获得指定驱动器的驱动器字母。

[取得指定驱动器的驱动器类型](#)

本例演示如何使用 **DriveType** 属性来获得指定驱动器的驱动器类型。

[取得指定驱动器的文件系统信息](#)

本例演示如何使用 **FileSystem** 来取得指定驱动器的文件系统信息。

[驱动器是否已就绪？](#)

本例演示如何使用 **IsReady** 属性来检查指定的驱动器是否已就绪。

[取得指定驱动器的路径](#)

本例演示如何使用 **Path** 属性来取得指定驱动器的路径。

[取得指定驱动器的根文件夹](#)

本例演示如何使用 **RootFolder** 属性来取得指定驱动器的根文件夹。

[取得指定驱动器的序列号](#)

本例演示如何使用 **Serialnumber** 属性来取得指定驱动器的序列号。

Drive 对象

Drive 对象用于返回关于本地磁盘驱动器或者网络共享驱动器的信息。**Drive** 对象可以返回有关驱动器的文件系统类型、剩余容量、序列号、卷标名等信息。

注释：您无法通过 **Drive** 对象返回有关驱动器内容的信息。要达到这个目的，请使用 **Folder** 对象。

如需操作 **Drive** 对象的相关属性，您需要通过 **FileSystemObject** 对象来创建 **Drive** 对象的实例。首先，创建一个 **FileSystemObject** 对象，然后通过 **FileSystemObject** 对象的 **GetDrive** 方法或者 **Drives** 属性来实例化 **Drive** 对象。

Drive 对象的属性描述如下：

属性

属性	描述
AvailableSpace	向用户返回在指定的驱动器或网络共享驱动器上的可用空间容量。
DriveLetter	返回识别本地驱动器或网络共享驱动器的大写字母。
DriveType	返回指定驱动器的类型。
FileSystem	返回指定驱动器所使用的文件系统。
FreeSpace	向用户返回在指定的驱动器或网络共享驱动器上的剩余空间容量。
IsReady	如果指定驱动器已就绪，则返回 true 。否则返回 false 。
Path	返回其后有一个冒号的大写字母，用来指示指定驱动器的路径名。
RootFolder	返回一个文件夹对象，该文件夹代表指定驱动器的根文件夹。
SerialNumber	返回指定驱动器的序列号。

ShareName	返回指定驱动器的网络共享名。
TotalSize	返回指定的驱动器或网络共享驱动器的总容量。
VolumeName	设置或者返回指定驱动器的卷标名。

[❏ ASP TextStream 对象](#)

[ASP File 对象](#) ❏

[❏ 点我分享笔记](#)

[反馈/建议](#)

Copyright © 2013-2018 菜鸟教程 [runoob.com](#) All Rights Reserved. 备案号: 闽ICP备15012807号-1



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

[❏ ASP Drive 对象](#)

[ASP Folder 对象](#) ❏

ASP File 对象

File 对象用于返回关于指定文件的信息。



尝试一下 - 实例

[文件最后被修改的时间?](#)

本例演示如何使用 **DateLastModified** 属性来取得指定文件最后被修改的日期和时间。

[文件最后被访问的时间?](#)

此例演示如何使用 **DateLastAccessed** 属性来取得指定文件最后被访问的日期和时间。

[返回指定文件的属性](#)

本例演示如何使用 **Attributes** 来返回指定文件的属性。

File 对象

File 对象用于返回关于指定文件的信息。

如需操作 **File** 对象的相关属性和方法，您需要通过 **FileSystemObject** 对象来创建 **File** 对象的实例。首先，创建一个 **FileSystemObject** 对象，然后通过 **FileSystemObject** 对象的 **GetFile** 方法，或者通过 **Folder** 对象的 **Files** 属性来实例化 **File** 对象。

下面的代码使用 **FileSystemObject** 对象的 **GetFile** 方法来实例化 **File** 对象，并使用 **DateCreated** 属性来返回指定文件的创建日期：

实例

```
<%
Dim fs,f
Set fs=Server.CreateObject("Scripting.FileSystemObject")
Set f=fs.GetFile("c:\test.txt")
Response.Write("File created: " & f.DateCreated)
set f=nothing
set fs=nothing
%>
```

[演示实例 »](#)

File 对象的属性和方法描述如下：

属性

属性	描述
Attributes	设置或返回指定文件的属性。
DateCreated	返回指定文件被创建的日期和时间。
DateLastAccessed	返回指定文件最后被访问的日期和时间。
DateLastModified	返回指定文件最后被修改的日期和时间。
Drive	返回指定文件或文件夹所在的驱动器的驱动器字母。
Name	设置或返回指定文件的名称。
ParentFolder	返回指定文件的父文件夹对象。
Path	返回指定文件的路径。
ShortName	返回指定文件的短名称（8.3 命名约定）。
ShortPath	返回指定文件的短路径（8.3 命名约定）。
Size	返回指定文件的尺寸（字节）。
Type	返回指定文件的类型。

方法

方法	描述
Copy	把指定文件从一个位置拷贝到另一个位置。
Delete	删除指定文件。
Move	把指定文件从一个位置移动到另一个位置。
OpenAsTextStream	打开指定文件，并返回一个 TextStream 对象来访问此文件。



ASP Folder 对象

Folder 对象用于返回关于指定文件夹的信息。

Folder 对象

Folder 对象用于返回关于指定文件夹的信息。

如需操作 Folder 对象的相关属性和方法，您需要通过 FileSystemObject 对象来创建 Folder 对象的实例。首先，创建一个 FileSystemObject 对象，然后通过 FileSystemObject 对象的 GetFolder 方法来实例化这个 Folder 对象。

下面的代码使用 FileSystemObject 对象的 GetFolder 方法来实例化 Folder 对象，并使用 DateCreated 属性来返回指定文件夹的创建日期：

```
<%
Dim fs,fo
Set fs=Server.CreateObject("Scripting.FileSystemObject")
Set fo=fs.GetFolder("c:\test")
Response.Write("Folder created: " & fo.DateCreated)
set fo=nothing
set fs=nothing
%>
```

输出：

Folder created: 10/22/2008 10:01:19 AM

Folder 对象的集合、属性和方法描述如下：

集合

集合	描述
Files	返回指定文件夹中所有文件的集合。
SubFolders	返回指定文件夹中所有子文件夹的集合。

属性

属性	描述
Attributes	设置或返回指定文件夹的属性。
DateCreated	返回指定文件夹被创建的日期和时间。
DateLastAccessed	返回指定文件夹最后被访问的日期和时间。
DateLastModified	返回指定文件夹最后被修改的日期和时间。
Drive	返回指定文件夹所在的驱动器的驱动器字母。
IsRootFolder	如果文件夹是根文件夹，则返回 true ，否则返回 false 。
Name	设置或返回指定文件夹的名称。
ParentFolder	返回指定文件夹的父文件夹。
Path	返回指定文件夹的路径。
ShortName	返回指定文件夹的短名称（8.3 命名约定）。
ShortPath	返回指定文件夹的短路径（8.3 命名约定）。
Size	返回指定文件夹的大小。
Type	返回指定文件夹的类型。

方法

方法	描述
Copy	把指定文件夹从一个位置拷贝到另一个位置。

Delete	删除指定文件夹。
Move	把指定文件夹从一个位置移动到另一个位置。
CreateTextFile	在指定文件夹创建一个新的文本文件，并返回一个 TextStream 对象来访问此文件。

❏ ASP File 对象

ASP Dictionary 对象 ❏

❏ 点我分享笔记

反馈/建议

❏

首页

HTML

CSS

JS

本地书签

❏ ASP Folder 对象

ASP ADO ❏

ASP Dictionary 对象

Dictionary 对象用于在名称/值对中存储信息。

尝试一下 - 实例

[指定的键存在吗？](#)

本例演示如何创建一个 **Dictionary** 对象，然后使用 **Exists** 方法来检查指定的键是否存在。

[返回一个所有项目的数组](#)

本例演示如何使用 **Items** 方法来返回一个所有项目的数组。

[返回一个所有键的数组](#)

本例演示如何使用 **Keys** 方法来返回一个所有键的数组。

[返回一个项目的值](#)

本例演示如何使用 **Item** 属性来返回一个项目的值。

[设置一个键](#)

本例演示如何使用 **Key** 属性来在 **Dictionary** 对象中设置一个键。

[返回键/项目对的数量](#)

本例演示如何使用 **Count** 属性来返回键/项目对的数量。

Dictionary 对象

Dictionary 对象用于在名称/值对（等同于键和项目）中存储信息。**Dictionary** 对象看似比数组更为简单，然而，**Dictionary** 对象却是更令人满意的处理关联数据的解决方案。

比较 **Dictionaries** 和数组：

键用于识别 **Dictionary** 对象中的项目

您无需调用 **ReDim** 来改变 **Dictionary** 对象的尺寸

当从 **Dictionary** 中删除一个项目时，其余的项目会自动上移

Dictionary 不是多维，而数组是多维

Dictionary 比数组带有更多的内建函数

Dictionary 在频繁地访问随机元素时，比数组工作得更好

Dictionary 在根据它们的内容定位项目时，比数组工作得更好

下面的实例创建了一个 Dictionary 对象，并向对象添加了一些键/项目对，然后取回了键 gr 的项目值：

```
<%
Dim d
Set d=Server.CreateObject("Scripting.Dictionary")
d.Add "re","Red"
d.Add "gr","Green"
d.Add "bl","Blue"
d.Add "pi","Pink"
Response.Write("The value of key gr is: " & d.Item("gr"))
%>
```

输出：

The value of key gr is: Green

Dictionary 对象的属性和方法描述如下：

属性

属性	描述
CompareMode	设置或返回用于在 Dictionary 对象中比较键的比较模式。
Count	返回 Dictionary 对象中键/项目对的数目。
Item	设置或返回 Dictionary 对象中一个项目的值。
Key	为 Dictionary 对象中已有的键值设置新的键值。

方法

方法	描述
Add	向 Dictionary 对象添加新的键/项目对。
Exists	返回一个布尔值，这个值指示指定的键是否存在于 Dictionary 对象中。
Items	返回 Dictionary 对象中所有项目的一个数组。
Keys	返回 Dictionary 对象中所有键的一个数组。
Remove	从 Dictionary 对象中删除指定的键/项目对。
RemoveAll	删除 Dictionary 对象中所有的键/项目对。



ASP ADO

ADO 用于从网页访问数据库。

从 ASP 页面访问数据库

从 ASP 页面内部访问数据库的通常方式是：

1. 创建至数据库的 ADO 连接
2. 打开数据库连接
3. 创建 ADO 记录集
4. 打开记录集
5. 从数据集中提取您需要的数据
6. 关闭数据集
7. 关闭连接

ADO 是什么？

- ADO 是一项微软公司的技术
- ADO 代表 **ActiveX Data Objects**
- ADO 是一个微软的 **Active-X** 组件
- ADO 会随着微软 IIS 自动安装
- ADO 是一个访问数据库中数据的编程接口

如果您想学习更多关于 ADO 的知识，请阅读我们的 [ADO 教程](#)。

[点我分享笔记](#)

[反馈/建议](#)

ASP AdRotator 组件

ASP AdRotator 组件

每当用户进入网站或刷新页面时，ASP AdRotator 组件就会创建一个 AdRotator 对象来显示一幅不同的图像。有关图像的信息是包含在一个文本文件中。

注释：AdRotator 在 Internet Information Server 7 (IIS7) 中不起作用。

语法

```
<%
set adrotator=server.createobject ("MSWC.AdRotator")
adrotator.GetAdvertisement("textfile.txt")
%>
```

ASP AdRotator 实例

假设我们有一个名为 "ads.txt" 的文件，如下所示：

```
REDIRECT banners.asp
*
runoob.gif
http://www.runoob.com
Free Tutorials from RUNOOB
50
xmlspy.gif
http://www.altova.com
XML Editor from Altova
50
```

在上面的文本文件中型号下面的行规定了要显示的图像的名称、超链接地址、图像的替换文本和每百次点击中的显示几率。
上述文本文件的第一行指定当访客点击图像时进行的动作。重定向页（banners.asp）将收到一个带有重定向 URL 的查询字符串。

提示：如需规定图像的高度、宽度和边框，您可以在 REDIRECT 下面插入如下代码：

```
REDIRECT banners.asp
WIDTH 468
HEIGHT 60
BORDER 0
*
runoob.gif
...
```

"banners.asp" 文件如下所示：

实例

```
<%
url=Request.QueryString("url")
If url<>"" then Response.Redirect(url)
%>

<!DOCTYPE html>
<html>
<body>
<%
set adrotator=Server.CreateObject("MSWC.AdRotator")
response.write(adrotator.GetAdvertisement("textfile.txt"))
%>
</body>
</html>
```

演示实例 »

好了，以上就是全部的内容！

ASP AdRotator 属性

属性	描述	实例
Border	规定围绕广告的边框的尺寸。	<pre><% set adrot=Server.CreateObject("MSWC.AdRotator") adrot.Border="2" Response.Write(adrot.GetAdvertisement("ads.txt")) %></pre>

Clickable	规定广告本身是否是超链接。	<% set adrot=Server.CreateObject("MSWC.AdRotator") adrot.Clickable=false Response.Write(adrot.GetAdvertisement("ads.txt")) %>
TargetFrame	显示广告的框架名称。	<% set adrot=Server.CreateObject("MSWC.AdRotator") adrot.TargetFrame="target='_blank'" Response.Write(adrot.GetAdvertisement("ads.txt")) %>

ASP AdRotator 方法

方法	描述	实例
GetAdvertisement	返回在页面中显示广告的 HTML。	<% set adrot=Server.CreateObject("MSWC.AdRotator") Response.Write(adrot.GetAdvertisement("ads.txt")) %>



ASP Browser Capabilities 组件

ASP Browser Capabilities 组件

ASP Browser Capabilities 组件会创建一个 **BrowserType** 对象，这个对象可测定访客浏览器的类型、性能和版本号。

当浏览器连接到服务器时，就会向服务器发送一个 HTTP User Agent 报头。这个报头包含关于浏览器的信息。

BrowserType 对象会把报头中的信息与服务器上名为 "Browsercap.ini" 的文件中的信息作比较。

如果报头中的浏览器类型和版本号与 "Browsercap.ini" 文件中信息匹配，那么我们就可以使用 **BrowserType** 对象列出这个匹配的浏览器的相关属性。

如果上述情况不匹配，这个对象会把每个属性设置为 "UNKNOWN"。

语法

```
<%  
Set MyBrow=Server.CreateObject("MSWC.BrowserType")  
%>
```

ASP Browser Capabilities 实例

下面的实例会在 ASP 文件中创建一个 **BrowserType** 对象，并显示当前浏览器性能的一些信息：

实例

```
<!DOCTYPE html>
<html>
<body>
<%
Set MyBrow=Server.CreateObject("MSWC.BrowserType")
%>

<table border="0" width="100%">
<tr>
<th>Client OS</th><th><%=MyBrow.platform%></th>
</tr><tr>
<td>Web Browser</td><td><%=MyBrow.browser%></td>
</tr><tr>
<td>Browser version</td><td><%=MyBrow.version%></td>
</tr><tr>
<td>Frame support?</td><td><%=MyBrow.frames%></td>
</tr><tr>
<td>Table support?</td><td><%=MyBrow.tables%></td>
</tr><tr>
<td>Sound support?</td><td><%=MyBrow.backgroundsounds%></td>
</tr><tr>
<td>Cookies support?</td><td><%=MyBrow.cookies%></td>
</tr><tr>
<td>VBScript support?</td><td><%=MyBrow.vbscript%></td>
</tr><tr>
<td>JavaScript support?</td><td><%=MyBrow.javascript%></td>
</tr>
</table>

</body>
</html>
```

输出：

Client OS	WinNT
Web Browser	IE
Browser version	5.0
Frame support?	True
Table support?	True
Sound support?	True
Cookies support?	True
VBScript support?	True
JavaScript support?	True

[演示实例 »](#)

Browscap.ini 文件

"Browscap.ini" 文件用于声明属性，并设置各浏览器的默认值。

这部分内容不是关于如何维护 **Browscap.ini** 文件的教程，我们只提供一些关于 "Browsercap.ini" 的基础知识和概念，让您对该文件有个大概的了解。

"Browscap.ini" 文件可包含下面的信息：

```
[;comments]
[HTTPUserAgentHeader]
[parent=browserDefinition]
[property1=value1]
[propertyN=valueN]
[Default Browser Capability Settings]
[defaultProperty1=defaultValue1]
[defaultPropertyN=defaultValueN]
```

参数	描述
comments	可选项。任何起始于分号的代码行都被 BrowserType 对象忽略。

HTTPUserAgentHeader	可选项。规定与在 propertyN 中设定的 browser-property 值声明相关的 HTTP User Agent 报头。允许使用通配符。
browserDefinition	可选项。规定作为父浏览器使用的某个浏览器的 HTTP User Agent header-string。当前浏览器的定义会继承在父浏览器的定义中所有声明过的属性值。
propertyN	可选项。规定浏览器的属性。下面的表格列出了某些可能的属性： ActiveXControls - 支持 ActiveX® 控件？ Backgroundsounds - 支持背景声音？ Cdf - 支持针对网络广播的频道定义格式？ Tables - 支持表格？ Cookies - 支持 cookies？ Frames - 支持框架？ Javaapplets - 支持 Java applets？ Javascript - 支持 JScript？ Vbscript - 支持 VBScript？ Browser - 规定浏览器的名称 Beta - 浏览器是否为 beta 软件？ Platform - 规定浏览器运行的平台 Version - 规定浏览器的版本号
valueN	可选项。规定 propertyN 的值。可为字符串、整数（前缀带 #）或者布尔值。
defaultPropertyN	可选项。规定浏览器属性的名称，假如已定义的 HTTPUserAgentHeader 值中没有值能与浏览器发送的 HTTP User Agent 报头相匹配，则为这个属性分配一个默认的值。
defaultValueN	可选项。规定 defaultPropertyN 的值。可为字符串、整数（前缀带 #）或者布尔值。

"Browscap.ini" 文件可能如下所示：

```
;IE 5.0
[IE 5.0]
browser=IE
Version=5.0
majorver=#5
minorver=#0
frames=TRUE
tables=TRUE
cookies=TRUE
backgroundsounds=TRUE
vbscript=TRUE
javascript=TRUE
javaapplets=TRUE
ActiveXControls=TRUE
beta=False

;DEFAULT BROWSER
[*]
browser=Default
frames=FALSE
tables=TRUE
cookies=FALSE
backgroundsounds=FALSE
vbscript=FALSE
javascript=FALSE
```

ASP Content Linking 组件



尝试一下 - 实例

[Content Linking 组件](#)

本例构建一个内容列表。

[Content Linking 组件 2](#)

本例使用 **Content Linking** 组件在一个文本文件所列的页面间进行导航。

ASP Content Linking 组件

ASP Content Linking 组件用于创建快捷便利的导航系统！

Content Linking 组件会返回一个 **Nextlink** 对象，这个对象用于容纳需要导航网页的一个列表。

语法

```
<%  
Set nl=Server.CreateObject("MSWC.NextLink")  
%>
```

ASP Content Linking 实例

首先，我们会创建一个文本文件 - "links.txt"：

```
asp_intro.asp ASP 简介  
asp_syntax.asp ASP 语法  
asp_variables.asp ASP 变量  
asp_procedures.asp ASP 程序
```

上面的文本文件包含需要导航的页面。页面的排列顺序应该与它们的显示顺序相同，并包含对每个文件名的描述（使用制表符来分隔文件名和描述信息）。

注释：如果您希望向列表添加页面，或者改变在列表中的页面顺序，那么您需要做的仅仅是修改这个文本文件而已！导航会自动更新！

然后我们创建一个引用文件，"nlcode.inc"。.inc 文件创建一个 **NextLink** 对象来在 "links.txt" 中列出的页面间进行导航。

"nlcode.inc"：

```
<%  
dim nl  
Set nl=Server.CreateObject("MSWC.NextLink")  
if (nl.GetListIndex("links.txt")>1) then  
Response.Write("<a href='" & nl.GetPreviousURL("links.txt")  
Response.Write("'>Previous Page</a>")  
end if  
Response.Write("<a href='" & nl.GetNextURL("links.txt")  
Response.Write("'>Next Page</a>")  
%>
```

请在文本文件 "links.txt" 列出的每个 .asp 页面中放置一行代码：<!-- #include file="nlcode.inc"-->。这行代码会在 "links.txt" 中列出每个页面上引用 "nlcode.inc" 中的代码，这样导航就可以工作了。

ASP Content Linking 组件的方法

方法	描述	实例
GetListCount	返回内容链接列表文件中所列项目的数量。	<% dim nl,c Set nl=Server.CreateObject("MSWC.NextLink") c=nl.GetListCount("links.txt") Response.Write("There are ") Response.Write(c) Response.Write(" items in the list") %> 输出： There are 4 items in the list
GetListIndex	返回在内容链接列表文件中当前条目的索引号。第一个条目的索引号是 1。如果当前页面不在内容链接列表文件中，则返回 0。	<% dim nl,c Set nl=Server.CreateObject("MSWC.NextLink") c=nl.GetListIndex("links.txt") Response.Write("Item number ") Response.Write(c) %> 输出： Item number 3
GetNextDescription	返回在内容链接列表文件中所列的下一个条目的文本描述。如果在列表文件中没有找到当前文件，则返回列表中最后一个页面的文本描述。	<% dim nl,c Set nl=Server.CreateObject("MSWC.NextLink") c=nl.GetNextDescription("links.txt") Response.Write("Next ") Response.Write("description is: ") Response.Write(c) %> 输出： Next description is: ASP Variables
GetNextURL	返回在内容链接列表文件中所列的下一个条目的 URL。如果在列表文件中没有找到当前文件，则返回列表中最后一个页面的 URL。	<% dim nl,c Set nl=Server.CreateObject("MSWC.NextLink") c=nl.GetNextURL("links.txt") Response.Write("Next ") Response.Write("URL is: ") Response.Write(c) %> 输出： Next URL is: asp_variables.asp

GetNthDescription	返回在内容链接列表文件中所列的第 N 个页面的描述信息。	<% dim nl,c Set nl=Server.CreateObject("MSWC.NextLink") c=nl.GetNthDescription("links.txt",3) Response.Write("Third ") Response.Write("description is: ") Response.Write(c) %> 输出: Third description is: ASP Variables
GetNthURL	返回在内容链接列表文件中所列的第 N 个页面的 URL。	<% dim nl,c Set nl=Server.CreateObject("MSWC.NextLink") c=nl.GetNthURL("links.txt",3) Response.Write("Third ") Response.Write("URL is: ") Response.Write(c) %> 输出: Third URL is: asp_variables.asp
GetPreviousDescription	返回在内容链接列表文件中所列的前一个条目的文本描述。如果在列表文件中没有找到当前文件,则返回列表中第一个页面的文本描述。	<% dim nl,c Set nl=Server.CreateObject("MSWC.NextLink") c=nl.GetPreviousDescription("links.txt") Response.Write("Previous ") Response.Write("description is: ") Response.Write(c) %> 输出: Previous description is: ASP Variables
GetPreviousURL	返回在内容链接列表文件中所列的前一个条目的 URL。如果在列表文件中没有找到当前文件,则返回列表中第一个页面的 URL。	<% dim nl,c Set nl=Server.CreateObject("MSWC.NextLink") c=nl.GetPreviousURL("links.txt") Response.Write("Previous ") Response.Write("URL is: ") Response.Write(c) %> 输出: Previous URL is: asp_variables.asp

[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)[ASP Content Linking](#)[AJAX 简介](#)

ASP Content Rotator 组件 (ASP 3.0)

ASP Content Rotator 组件

ASP Content Rotator 组件创建一个 ContentRotator 对象，每当访客进入网站或刷新页面时，该对象就会显示一段不同的内容字符串。

有关内容字符串的信息是包含在一个名为内容目录文件（Content Schedule File）的文本文件中。

内容字符串可包含 HTML 标签，这样您就可以显示 HTML 可呈现的任何类型的内容：文本、图像、颜色或者超链接。

语法

```
<%  
Set cr=Server.CreateObject("MSWC.ContentRotator")  
%>
```

ASP Content Rotator 实例

每当访客浏览网页时，下面的实例就会显示不同的内容。

首先，创建一个名为 "textads.txt" 的文本文件，并把它放置在名为 "text" 的子文件夹中。

"textads.txt":

```
%% #3  
<h2>This is a great day!!</h2>  
  
%% #3  
  
  
%% #4  
<a href="http://www.runoob.com">Visit RUNOOB</a>
```

请注意在每个内容字符串起始位置的 # 号码。这个号码是一个可选的参数，用来指示 HTML 内容字符串的相对权重。在上面的文本文件中，Content Rotator 有十分之三的几率显示第一个内容字符串，有十分之三的几率显示第二个内容字符串，有十分之四的几率显示第三个字符串。

然后，创建一个 ASP 文件，并插入下面的代码：

实例

```
<html>  
<body>  
<%  
set cr=server.createObject("MSWC.ContentRotator")  
response.write(cr.ChooseContent("text/textads.txt"))  
%>  
</body>  
</html>
```

[演示实例 »](#)

ASP Content Rotator 组件的方法

方法	描述	实例
----	----	----

ChooseContent	获取并显示某个内容字符串。	<% dim cr Set cr=Server.CreateObject("MSWC.ContentRotator") response.write(cr.ChooseContent("text/textads.txt")) %> 输出： <input type="text"/>
GetAllContent	取回并显示文本文件中所有的内容字符串。	<% dim cr Set cr=Server.CreateObject("MSWC.ContentRotator") response.write(cr.GetAllContent("text/textads.txt")) %> 输出： This is a great day!! <input type="text"/> Visit RUNOOB

[ASP Content Linking](#)

[AJAX 简介](#)

[点我分享笔记](#)

[反馈/建议](#)

Copyright © 2013-2018 菜鸟教程 [runoob.com](#) All Rights Reserved. 备案号：闽ICP备15012807号-1



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

[ASP Content Rotator](#)

[ASP – AJAX 与 ASP](#)

AJAX 简介

AJAX 是一种在不重载整个页面的情况下对网页的某些部分进行更新的技术。

AJAX 是什么？

AJAX= Asynchronous JavaScript and XML.

AJAX 是一种用于创建快速动态网页的技术。

AJAX 通过在后台与服务器进行少量数据交换，使网页实现异步更新。这意味着可以在不重载整个页面的情况下，对网页的某些部分进行更新。

传统的网页（不使用 AJAX）如果需要更新内容，必须重载整个页面。

有很多使用 AJAX 的应用程序案例：Google Maps、Gmail、Youtube 和 Facebook。

AJAX 如何工作

AJAX

AJAX 基于因特网标准

AJAX 基于因特网标准，并使用以下技术组合：

- XMLHttpRequest 对象（与服务器异步交互数据）
- JavaScript/DOM（显示/取回信息）
- CSS（设置数据的样式）
- XML（常用作数据传输的格式）

☐ AJAX 应用程序与浏览器和平台无关的！

谷歌搜索建议（Google Suggest）

随着谷歌搜索建议功能在 2005 的发布，AJAX 开始流行起来。

[谷歌搜索建议（Google Suggest）](#) 使用 AJAX 创造出动态性极强的 web 界面：当您在谷歌的搜索框中键入内容时，JavaScript 会把字符发送到服务器，服务器则会返回建议列表。

今天就开始使用 AJAX

在我们的 ASP 教程中，我们将演示 AJAX 如何在不重载整个页面的情况下对网页的某些部分进行更新。服务器脚本我们将采用 ASP 来编写。如果您想要学习更多关于 AJAX 的知识，请访问我们的 [AJAX 教程](#)。

☐ ASP Content Rotator ASP – AJAX 与 ASP ☐

☐ [点我分享笔记](#)

反馈/建议



☐ AJAX 简介 AJAX 数据库 ☐

ASP - AJAX 与 ASP

AJAX 被用于创建交互性更强的应用程序。

AJAX ASP 实例

下面的实例将演示当用户在输入框中键入字符时，网页如何与 Web 服务器进行通信：

实例

Start typing a name in the input field below:

First name:

Suggestions:

实例解释 - HTML 页面

当用户在上方的输入框中键入字符时，会执行 `showHint()` 函数。该函数由 `onkeyup` 事件触发：

```
<!DOCTYPE html>
<html>
<head>
<script>
function showHint(str)
{
  if (str.length==0)
  {
    document.getElementById("txtHint").innerHTML="";
    return;
  }
  if (window.XMLHttpRequest)
  { // code for IE7+, Firefox, Chrome, Opera, Safari
    xmlhttp=new XMLHttpRequest();
  }
  else
  { // code for IE6, IE5
    xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
  xmlhttp.onreadystatechange=function()
  {
    if (xmlhttp.readyState==4 && xmlhttp.status==200)
    {
      document.getElementById("txtHint").innerHTML=xmlhttp.responseText;
    }
  }
  xmlhttp.open("GET","gethint.asp?q="+str,true);
  xmlhttp.send();
}
</script>
</head>
<body>

<p><b>Start typing a name in the input field below:</b></p>
<form>
First name: <input type="text" onkeyup="showHint(this.value)" size="20">
</form>
<p>Suggestions: <span id="txtHint"></span></p>

</body>
</html>
```

源代码解释：

如果输入框是空的（`str.length==0`），该函数会清空 `txtHint` 占位符的内容，并退出该函数。

如果输入框不是空的，那么 `showHint()` 会执行以下步骤：

创建 `XMLHttpRequest` 对象

创建在服务器响应就绪时执行的函数

向服务器上的文件发送请求

请注意添加到 `URL` 末端的参数（`q`）（包含输入框的内容）

ASP 文件

上面这段通过 `JavaScript` 调用的服务器页面是名为 `"gethint.asp"` 的 `ASP` 文件。

`"gethint.asp"` 中的源代码会检查姓名数组，然后向浏览器返回对应的姓名：

```
<%
response.expires=-1
dim a(30)
'Fill up array with names
a(1)="Anna"
a(2)="Brittany"
a(3)="Cinderella"
a(4)="Diana"
a(5)="Eva"
a(6)="Fiona"
a(7)="Gunda"
a(8)="Hege"
a(9)="Inga"
a(10)="Johanna"
a(11)="Kitty"
a(12)="Linda"
a(13)="Nina"
a(14)="Ophelia"
```

```

a(15)="Petunia"
a(16)="Amanda"
a(17)="Raquel"
a(18)="Cindy"
a(19)="Doris"
a(20)="Eve"
a(21)="Evita"
a(22)="Sunniva"
a(23)="Tove"
a(24)="Unni"
a(25)="Violet"
a(26)="Liza"
a(27)="Elizabeth"
a(28)="Ellen"
a(29)="Wenche"
a(30)="Vicky"

'get the q parameter from URL
q=ucase(request.querystring("q"))

'lookup all hints from array if length of q>0
if len(q)>0 then
hint=""
for i=1 to 30
if q=ucase(mid(a(i),1,len(q))) then
if hint="" then
hint=a(i)
else
hint=hint & " , " & a(i)
end if
end if
next
end if

'Output "no suggestion" if no hint were found
'or output the correct values
if hint="" then
response.write("no suggestion")
else
response.write(hint)
end if
%>

```

解释：如果 JavaScript 发送了任何文本（即 `strlen($q) > 0`），则会发生：

1. 查找匹配 JavaScript 发送的字符的姓名
2. 如果未找到匹配，则将响应字符串设置为 "no suggestion"
3. 如果找到一个或多个匹配姓名，则用所有姓名设置响应字符串
4. 把响应发送到 "txtHint" 占位符

[AJAX 简介](#)

[AJAX 数据库](#)

[点我分享笔记](#)

[反馈/建议](#)

Copyright © 2013-2018 菜鸟教程 [runoob.com](#) All Rights Reserved. 备案号：闽ICP备15012807号-1



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

[ASP – AJAX 与 ASP](#)

[ASP 快速参考](#)

AJAX 数据库实例

AJAX 可用来与数据库进行交互式通信。

AJAX 数据库实例

下面的实例将演示网页如何通过 **AJAX** 从数据库读取信息：

实例

Select a customer: ▼

Customer info will be listed here...

实例解释 - HTML 页面

当用户在上面的下拉列表中选择某位客户时，会执行名为 **"showCustomer()"** 的函数。该函数由 **"onchange"** 事件触发：

```
<!DOCTYPE html>
<html>
<head>
<script>
function showCustomer(str)
{
  if (str=="")
  {
    document.getElementById("txtHint").innerHTML="";
    return;
  }
  if (window.XMLHttpRequest)
  { // code for IE7+, Firefox, Chrome, Opera, Safari
    xmlhttp=new XMLHttpRequest();
  }
  else
  { // code for IE6, IE5
    xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
  xmlhttp.onreadystatechange=function()
  {
    if (xmlhttp.readyState==4 && xmlhttp.status==200)
    {
      document.getElementById("txtHint").innerHTML=xmlhttp.responseText;
    }
  }
  xmlhttp.open("GET","getcustomer.asp?q="+str,true);
  xmlhttp.send();
}
</script>
</head>
<body>

<form>
<select name="customers" onchange="showCustomer(this.value)">
<option value="">Select a customer:</option>
<option value="ALFKI">Alfreds Futterkiste</option>
<option value="NORTS " ">North/South</option>
<option value="WOLZA">Wolski Zajazd</option>
</select>
</form>
<br>
<div id="txtHint">Customer info will be listed here...</div>

</body>
</html>
```

源代码解释：

如果没有选择客户（**str.length==0**），那么该函数会清空 **txtHint** 占位符，然后退出该函数。

如果已选择一位客户，则 **showCustomer()** 函数会执行以下步骤：

- 创建 **XMLHttpRequest** 对象
- 创建在服务器响应就绪时执行的函数
- 向服务器上的文件发送请求

ASP 文件

上面这段通过 **JavaScript** 调用的服务器页面是名为 **"getcustomer.asp"** 的 **ASP** 文件。

"getcustomer.asp" 中的源代码会运行一次针对数据库的查询，然后在 **HTML** 表格中返回结果：

```
<%
response.expires=-1
sql="SELECT * FROM CUSTOMERS WHERE CUSTOMERID="
sql=sql & "'" & request.querystring("q") & "'"

set conn=Server.CreateObject("ADODB.Connection")
conn.Provider="Microsoft.Jet.OLEDB.4.0"
conn.Open(Server.MapPath("/db/northwind.mdb"))
set rs=Server.CreateObject("ADODB.recordset")
rs.Open sql,conn

response.write("<table>")
do until rs.EOF
for each x in rs.Fields
response.write("<tr><td><b>" & x.name & "</b></td>")
response.write("<td>" & x.value & "</td></tr>")
next
rs.MoveNext
loop
response.write("</table>")
%>
```

[☐ ASP – AJAX 与 ASP](#)

[ASP 快速参考](#) ☐

[☐ 点我分享笔记](#)

反馈/建议



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

[☐ AJAX 数据库](#)

[ASP 总结](#) ☐

ASP 快速参考

来自菜鸟教程的 **ASP 快速参考**。打印出来，放入口袋，以备随时使用。

基础语法

ASP 脚本由 **<%** 和 **%>** 包围。向浏览器写输出：

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

ASP 中的默认语言是 **VBScript**。如需使用其他脚本语言，请在 **ASP** 页面顶部插入一段语言说明：

```
<%@ language="javascript" %>
```

```
<html>

<body>

<%

....

%>
```

表单和用户输入

`Request.QueryString` 用于收集 `method="get"` 的表单中的值。使用 `GET` 方法从表单传送的信息对所有的用户都是可见的（出现在浏览器的地址栏），并且对所发送信息的量也有限制。

`Request.Form` 用于收集使用 `method="post"` 的表单中的值。使用 `POST` 方法从表单传送的信息对用户是不可见的，并且对所发送信息的量没有限制。

ASP Cookies

`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`。

`Response.Cookies` 命令用于创建 `cookie`：

```
<%

Response.Cookies("firstname")="Alex"

Response.Cookies("firstname").Expires="May 10,2002"

%>
```

注释：`Response.Cookies` 命令必须出现在 `<html>` 标签之前！

`"Request.Cookies"` 命令用于取回 `cookie` 值：

```
<%

fname=Request.Cookies("firstname")

response.write("Firstname=" & fname)

%>
```

引用文件

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。

语法：

```
<!--#include virtual="somefile.inc"-->
```

或者

```
<!--#include file ="somefile.inc"-->
```

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" -->
```

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" -->
```

请使用带有语法 `(..)` 的关键词 `file` 来引用更高层级目录中的文件。

Global.asa

`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。

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` 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

Application 和 Session 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。**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> 标签内部：

```
<script language="vbscript" runat="server">
sub Application_OnStart
' some code
end sub
sub Application_OnEnd
' some code
end sub
sub Session_OnStart
' some code
end sub
sub Session_OnEnd
' some code
end sub
</script>
```

<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>
```

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="typelibraryuuid"
version="versionnumber"
lcid="localeid"
-->
```

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）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。

集合

- Contents - 包含所有通过脚本命令追加到 session 的条目。
- StaticObjects - 包含了所有使用 HTML 的 <object> 标签追加到 session 的对象。
- Contents.Remove(*item/index*) - 从 Contents 集合删除一个项目。
- Contents.RemoveAll() - 从 Contents 集合删除全部项目。

属性

- CodePage - 规定显示动态内容时使用的字符集。
- LCID - 设置用于显示动态内容的区域标识符。

SessionID - 返回 **session id**

Timeout - 设置或返回 **session** 的超时时间。

方法

Abandon - 撤销 **session** 对象中的所有对象。

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** 对象用于把这些文件捆绑在一起。所有的用户分享一个 **Application** 对象。**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）。

集合

Contents - 包含所有通过脚本命令追加到应用程序中的项目。

StaticObjects - 包含所有使用 **HTML** 的 **<object>** 标签追加到应用程序中的对象。

Contents.Remove - 从 **Contents** 集合中删除一个项目。

Contents.RemoveAll - 从 **Contents** 集合中删除所有的项目。

方法

Lock - 防止用户修改 **Application** 对象中的属性。

Unlock - 允许用户修改 **Application** 对象中的属性。

Response 对象

Response 对象用于从服务器向用户发送输出的结果。

集合

Cookies(name) - 设置 **cookie** 的值。如果 **cookie** 不存在，则创建 **cookie**，并设置指定的值。

属性

Buffer - 规定是否缓冲输出。当输出设置缓冲时，服务器会阻止向浏览器的响应，直到所有的服务器脚本均被处理，或者直到脚本调用了 **Flush** 或 **End** 方法。如果要设置此属性，它应当位于 **ASP** 文件中的 **<html>** 标签之前。

CacheControl - 设置代理服务器是否可以缓存由 **ASP** 产生的输出。如果设置为 **Public**，则代理服务器会缓存页面。

Charset(charset_name) - 将字符集的名称（比如 **"ISO8859-1"**）追加到 **Response** 对象中的内容类型报头。

ContentType - 设置 **Response** 对象的 **HTTP** 内容类型（比如 **"text/html"**, **"image/gif"**, **"image/jpeg"**, **"text/plain"**）。默认是 **"text/html"**。

Expires - 设置页面在失效前的浏览器缓存时间（分钟）。

ExpiresAbsolute - 设置浏览器上页面缓存失效的日期和时间。

IsClientConnected - 指示客户端是否已从服务器断开。

Pics(pics_label) - 向 **response** 报头的 **PICS** 标签追加值。

Status - 规定由服务器返回的状态行的值。

方法

AddHeader(name, value) - 向 **HTTP** 响应添加新的 **HTTP** 报头和值。

AppendToLog string - 向服务器记录项目（**server log entry**）的末端添加字符串。

BinaryWrite(data_to_write) - 在没有任何字符转换的情况下直接向输出写数据。

Clear - 清除已缓冲的输出。使用该方法来处理错误。如果 **Response.Buffer** 未设置为 **true**，该方法将产生 **run-time** 错误。

End - 停止处理脚本，并返回当前的结果。

Flush - 立即发送已缓冲的输出。如果 **Response.Buffer** 未设置为 **true**，该方法将产生 **run-time** 错误。

Redirect(url) - 把用户重定向到另一个 **URL**。

Write(data_to_write) - 向用户写文本。

Request 对象

当浏览器向服务器请求页面时，这个行为就被称为一个 **request**（请求）。**Request** 对象用于从用户那里获取信息。

集合

- ClientCertificate** - 包含了存储在客户证书中的所有的字段值。
- Cookies(*name*)** - 包含了 **HTTP** 请求中发送的所有的 **cookie** 值。
- Form(*element_name*)** - 包含了使用 **post** 方法由表单发送的所有的表单（输入）值。
- QueryString(*variable_name*)** - 包含了 **HTTP** 查询字符串中所有的变量值。
- ServerVariables(*server_variable*)** - 包含了所有的服务器变量值。

属性

- TotalBytes** - 返回在请求正文中客户端发送的字节总数。

方法

- BinaryRead** - 取回作为 **post** 请求的一部分而从客户端发送至服务器的数据。

Server 对象

Server 对象用于访问服务器上的属性和方法。

属性

- ScriptTimeout** - 设置或返回在一段脚本终止前它所能运行时间（秒）的最大值。

方法

- CreateObject(*type_of_object*)** - 创建对象的实例。
- Execute(*path*)** - 从 **ASP** 文件内部执行另一个 **ASP** 文件。在被调用的 **ASP** 文件执行完毕后，控制权返回原先的 **ASP** 文件。
- GetLastError()** - 返回可描述已发生错误状态的 **ASPErrors** 对象。
- HTMLEncode(*string*)** - 对字符串应用 **HTML** 编码。
- MapPath(*path*)** - 把相对或虚拟路径映射为物理路径。
- Transfer(*path*)** - 把所有状态信息发送到另一个文件以备处理。在传送之后，程序的控制权不会返回原先的 **ASP** 文件。
- URLEncode(*string*)** - 对字符串应用 **URL** 编码规则。

您已经学习了 **ASP**，下一步呢？

ASP 总结

本教程已经为您讲解了如何为您的网站添加服务器端脚本，使得您网站的动态性和交互性更强。

您已经学习了如何动态地编辑、改变或者添加网页内容，对来自 **HTML** 表单提交的数据进行响应，访问数据或数据库并向浏览器返回结果，为不同的用户定制页面从而使这些页面的可用性更强。

如需了解更多关于 **ASP** 的信息，请查阅我们的 [ASP 实例](#)。

现在您已经学习了 **ASP**，下一步学习什么呢？

下一步应该学习的内容是 **SQL** 和 **ADO**。

SQL

SQL 是用于访问和处理数据库系统的标准的计算机语言。

SQL 语句用于取回和更新数据库中的数据。**SQL** 与数据库程序协同工作，比如 **MS Access**、**DB2**、**Informix**、**MS SQL Server**、**Oracle**、**Sybase** 以及其他数据库系统。

如果您想要学习更多关于 **SQL** 的知识，请访问我们的 [SQL 教程](#)。

ADO

ADO 是从网站访问数据库中数据的编程接口。

ADO 使用 **SQL** 来查询数据库中的数据。

如果您想要学习更多关于 **ADO** 的知识，请访问我们的 [ADO 教程](#)。

[☐ ASP 快速参考](#)

[ASP 实例 ☐](#)

[☐ 点我分享笔记](#)



反馈/建议

Copyright © 2013-2018 菜鸟教程 [runoob.com](#) All Rights Reserved. 备案号：闽ICP备15012807号-1



[首页](#) [HTML](#) [CSS](#) [JS](#) [本地书签](#)

[☐ ASP 总结](#)

[ASP Cookies 集合 ☐](#)

ASP 实例

基础

[用ASP写文本](#)

[向文本添加HTML](#)

变量

[声明变量](#)

[声明数组](#)

[循环生成HTML标题](#)

[使用Vbscript制作基于时间的问候语](#)

[使用JavaScript制作基于时间的问候语](#)

Date/Time 函数(VBScript)

[日期和时间](#)

[获得天的名字](#)

[获得月的名字](#)

[获得今天的月 and 天](#)

[3000年倒计时](#)

[从今天起计算N天后](#)

[格式化日期和时间](#)

[这是一个日期吗？](#)

其他函数(VBScript)

[大写或小写的字符串](#)

[字符串去掉前后空格](#)

[如何转化一个字符串？](#)

[四舍五入](#)

[一个随机数](#)

[从一个字符串返回特定的字符](#)

[替换字符串的特定字符](#)

[从字符串中返回指定数目的字符](#)

程序

[调用使用VBScript的子程序](#)

[调用使用JavaScript的子程序](#)

[调用使用VBScript和JavaScript的子程序](#)

表单

[使用method="get"的表单](#)

[使用method="post"的表单](#)

[使用单选按钮的表单](#)

Cookie

[Welcome cookie](#)

Response 对象

[使用ASP写文本](#)

[在ASP中使用HTML标签格式化文本](#)

[将用户重定向至不同的URL](#)

[显示随机的链接](#)

[控制缓存](#)

[清空缓存](#)

[在处理过程中终止脚本并返回结果](#)

[设置在页面失效前把页面在浏览器中缓存多少分钟](#)

[设置页面缓存在浏览器中的失效日期或时间](#)

[检查用户是否仍然与服务器相连](#)

[设置内容类型](#)

[设置字符集](#)

Request 对象

[链接url传参数](#)

[一个QueryString集合的简单应用](#)

[从表单获取信息](#)

[从表单获取更多信息](#)

[表单集合的简单应用](#)

[如何使用来自表单的信息](#)

[从表单获取更多信息](#)

[带单选按钮的表单](#)

[带复选框的表单](#)

[如何获得访问者的浏览器类型，IP地址或更多](#)

[获取所有要求的servervariables](#)

[Welcome cookie](#)

[用户发送的字节总数](#)

Session 对象

[返回SessionID](#)

[session 的超时](#)

Server 对象

[此文件最后被修改的时间是？](#)

[打开并读取某个文本文件](#)

[自制的点击计数器](#)

FileSystem 对象

[指定的文件存在吗？](#)

[指定的文件夹存在吗？](#)

[指定的驱动器存在吗？](#)

[取得某个指定驱动器的名称](#)

[取得某个指定路径的父文件夹的名称](#)

[取得文件扩展名](#)

[取得文件或文件夹的基名称](#)

TextStream 对象

[读文本文件](#)

[读文本文件中的一个部分](#)

[读文本文件中的一行](#)

[读取文本文件的所有行](#)

[略过文本文件的一部分](#)

[略过文本文件的一行](#)

[返回行数](#)

[取得列数](#)

Drive 对象

[取得指定驱动器的可用空间数](#)

[取得指定驱动器的剩余空间容量](#)

[取得指定驱动器的总容量](#)

[取得指定驱动器的驱动器字母](#)

[取得指定驱动器的驱动器类型](#)

[取得指定驱动器的文件系统信息](#)

[驱动器是否已就绪？](#)

[取得指定驱动器的路径](#)

[取得指定驱动器的根文件夹](#)

[取得指定驱动器的序列号](#)

File 对象

[文件何时被创建](#)

[此文件何时被修改](#)

[此文件何时被访问过](#)

[返回指定文件的属性](#)

Dictionary 对象

[指定的键存在吗？](#)

[返回一个所有项目的数组](#)

[返回一个所有键的数组](#)

[返回某个项目的值](#)

[设置一个键](#)

[返回键/项目对的数目](#)

AdRotator 组件

[简单的AdRotator实例](#)

Browser Capabilities 组件

[每一个访问网站的浏览器的类型、性能以及版本号](#)

ContentRotator 组件

[Content Rotator 组件](#)

Content Linking 组件

[Content Linking 组件](#)

[Content Linking 组件2](#)

[☐ ASP 总结](#)

ASP Cookies 集合 [☐](#)

[☐ 点我分享笔记](#)

[反馈/建议](#)