

Manual de Introducción al ASP (Active Server Pages)

Manual de introducción al ASP

- 1 - Introducción
- 2 - Principios básicos
- 3 - Objetos
- 4 - Funciones básicas
- 5 - Introducción a las bases de datos
- 6 - Alojamiento ASP
- 7 - Acerca de este manual

1.- Introducción

¿Qué es ASP?

Las siglas ASP corresponden a las palabras **Active Server Pages** (Páginas Activas en el Servidor). Es una tecnología que impulsó Microsoft hace ya varios años, y que en la actualidad es uno de los lenguajes de programación web más utilizados. Su funcionamiento se basa, principalmente, sobre servidores Microsoft con Internet Information Server para Windows NT o 2000, y en caso de contar con un sistema operativo Windows 95 o 98 se utiliza un servidor web personal, especialmente el Personal Web Server.

Requisitos esenciales

Hay que destacar que las páginas ASP se ejecutan del lado del servidor, devolviendo al cliente los resultados, esto quiere decir que no importa el navegador o sistema operativo del usuario, ya que el mismo verá una simple página HTML. Para correr aplicaciones ASP bajo sistemas con Windows 95 o 98 es necesario contar con el **Personal Web Server**, el cual está incluido en el CD de Instalación de Windows 98. Si el sistema es Windows NT o 2000 hay que contar con el IIS (Internet Information Server). Hoy en día, es posible correr páginas ASP bajo servidores Unix/Linux, utilizando algún software como puede ser Instant ASP o Chillisoft.

Aplicaciones

Una de las características más importantes de las páginas ASP es la posibilidad de conectar con diferentes tipos de bases de datos, para extraer-agregar-eliminar datos de ellas, y generar páginas con esos datos. Estas páginas se generan en forma dinámica, dependen de las sentencias que se establezcan, para si obtener los resultados del proceso realizado. Pueden conectarse a motores de bases de datos SQL, Access, Oracle, y a cualquier otro con soporte de conexión ODBC.

¿Con que hacer aplicaciones ASP?

No hay un software específico para realizar páginas ASP, se puede utilizar hasta el Bloc de Notas de Windows, Frontpage, DreamWeaver, entre otros. Pero siempre es conveniente utilizar algún editor de texto como el Edit Plus o Gasp, los cuales hacen mucho más fácil la programación.

2.- Principios básicos

Bloques de código ASP

Al igual que el lenguaje HTML, los códigos ASP tienen un tag de inicio y fin de una página.

En HTML se utiliza <tag> para abrir, y </tag> para cerrar, mientras que en ASP se utiliza <% y %>:

```
<%  
...  
%>
```

Declaración del lenguaje

Como ocurre en otros lenguajes de programación, se suele declarar el lenguaje a utilizar. Aunque esta sentencia es opcional en ASP muchos programadores suelen utilizarla:

Manual de Introducción al ASP (Active Server Pages)

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

Esta declaración se escribe al principio del archivo a utilizar, antes de cualquier otra expresión.

Comentarios

Al igual que otros lenguajes, es posible comentar el código.

```
<%
' Esto es un comentario
rem Esto es un comentario
% >
```

Un comentario dentro del código ASP se señala con una ' (comilla simple) antes del comentario, también se suele cambiar la ' por la palabra rem. De esta forma, toda la línea que contenga un comentario el interprete de asp del servidor no lo leerá ni lo ejecutará.

Los comentarios son muy útiles cuando tenemos cientos o miles de líneas de código y luego queremos hacer alguna modificación, poniendo diferentes comentarios pueden ayudarnos mucho más adelante.

Una simple página .ASP por dentro

El aspecto "interno" de una página ASP es el siguiente:

```
<%
' Referencia del lenguaje
' Declaración de variables
' etc.
% >
<html >
<title>WebExperto - Manual de ASP</title>
<body>
<%
' Conexión bases de datos
' Cálculos, etc.
% >
</body>
</html >
```

Como pueden observar, se trata de lenguaje HTML común y corriente con agregados de scripts de código ASP.

Declaración de variables

En una página ASP no hay que señalar de que tipo de variable se trata, todas son del tipo Variant.

La declaración de las variables es opcional, pero es recomendable hacerlo ya que evita posibles errores y facilita la lectura del código.

Las variables se declaran con el comando **Dim**. Para forzar a que se declaren todas las variables de una página se utiliza la función "**Option Explicit**". Ejemplo:

```
<%
Option Explicit
' Declaramos las variables y las cargamos
Dim nombre, apellido, email
nombre = "Fabian"
apellido = "Muller"
email = "fabian@webexperto.com"
% >
<html >
<head></head>
<body>
Los datos son:<br>
Nombre: <%=nombre%><br>
Apellido: <%=apellido%> <br>
Email: <%=email%><br>
</body>
</html >
```

En este ejemplo se declaran tres variables y se cargan con los datos correspondientes, y luego se imprimen en medio del

Realizado por Fabian Müller (fabian@webexperto.com) - www.webexperto.com

Manual de Introducción al ASP (Active Server Pages)

código html normal.

3.- Objetos

Objeto Response

Este objeto es uno de los mas utilizados en las paginas ASP, ya que dispone de la comunicación entre el cliente y el servidor.

Response.Write

Sirve para escribir en la pagina, ya sea texto común o variables. Siguiendo el ejemplo anterior:

```
<%
Option Explicit
' Declaramos las variables y las cargamos
Dim nombre, apellido, email
nombre = "Fabian"
apellido = "Muller"
email = "fabian@webexperto.com"
%>
<html>
<head></head>
<body>
<%
Response Write "Los datos son:<br>"
Response Write "Nombre: " & nombre & "<br>"
Response Write "Apellido: " & apellido & "<br>"
Response Write "Email: " & email & "<br>"
%>
</body>
</html>
```

Response.Redirect

Sirve para redireccionar una pagina hacia otra:

```
<%
Response Redirect "pagina2.asp"
%>
```

Siempre debe utilizarse antes de los tags Html, ya que de otra forma dará error.

Response.Cookie

Sirve para "plantar" una cookie en la PC del cliente:

```
<%
Response.Cookies("nombre") = "Fabian"
Response.Cookies("edad") = "18"
%>
```

En la cookie **nombre** se guardará la palabra **Fabian**, y en **edad 18**.

Objeto Request

Es el encargado de tomar los datos, ya sea de un formulario o de otro tipo de variables.

Request.Form

Toma los datos ingresados en un formulario. En el siguiente ejemplo el formulario incluido dentro del archivo formulario.htm, envía a procesar los datos al archivo procesa.asp:

```
formulario.htm
<html>
<head></head>
```

Realizado por Fabian Müller (fabian@webexperto.com) - www.webexperto.com

Manual de Introducción al ASP (Active Server Pages)

```
<body>
<form method="post" action="procesa.asp">
<input type="text" name="camponombre">
<input type="text" name="campoapellido">
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

procesa.asp

```
<%
' Como vimos anteriormente, declaramos variables
Option Explicit
Dim nombre, apellido
' Recogemos los datos del formulario y los guardamos
nombre=Request.Form("camponombre")
apellido=Request.Form("campoapellido")
%>
<html>
<head></head>
<body>
Datos que ingresaste:<br>
Nombre: <%=nombre%><br>
Apellido: <%=apellido%><br>
</body>
</html>
```

Y luego se imprimen los resultados.

Request.QueryString

Este método se utiliza para pasar valores de una página hacia otra a través de un link:
tomardatos.asp?id=121&plataforma=windows

En lo remarcado en negrita, pueden notar las variables:

id=121

plataforma=windows

Para recoger estos datos, se utiliza (dentro del archivo tomardatos.asp):

```
<%
Option Explicit
Dim id, plataforma
' Recogemos los datos del querystring
id=Request.QueryString("id")
plataforma=Request.QueryString("plataforma")
%>
```

Request.Cookies

Con este método podemos recoger los datos almacenados en una cookie, a la cual cargamos con algunos datos con el objeto Reponse.

```
Request.Cookies("nombredelacookie")
```

Así obtendríamos la información de la PC del usuario. Un ejemplo para utilizarlo, es crear un formulario que pida un nombre, y luego guardar ese valor en una cookie, para cuando nos vuelva a visitar lo reconozcamos y lo saludemos:

Manual de Introducción al ASP (Active Server Pages)

```
<%
nombre=Request.Cookies("nombre")
Response.Write "Hola, " & nombre & ". "
%>
```

Objeto Session

Permite almacenar información necesaria para una sesión del usuario. Las variables almacenadas con el objeto Session no se pierden al cambiar de página, sino que perduran hasta que se eliminan por el servidor, o bien cuando se llama al método Abandon que cierra la sesión.

```
<% session("nombre")="Fabian"%>
```

De esta forma, almacenaremos dentro de la variable de sesión **nombre** la palabra Fabian. También podemos, a partir de una variable de session, guardarla información en otra variable:

```
<% nombre= session("nombre")%>
```

Para destruir una variable session puede ocurrir dos cosas: una es que el usuario esté 20 minutos sin actividad dentro del sitio, y la otra es llamar al método Abandon:

```
<% Session.Abandon%>
```

Objeto Application

Este objeto se utiliza para compartir información entre todos los usuarios de una aplicación. Como varios usuarios pueden compartir este objeto, se utilizan los métodos Lock y Unlock para no permitir que dos o más usuarios puedan al mismo tiempo modificar la propiedad.

- **Lock**

Asegura que solo un usuario pueda modificar el objeto Application a la vez.

```
<% Application.Lock%>
```

- **Unlock**

Desbloquea al objeto previamente bloqueado, para poder ser modificado por otro usuario después de haberlo bloqueado mediante el método Lock. Si no se desbloquea el objeto Application, el servidor lo hace automáticamente cuando la página.asp termina o transcurre el tiempo de espera.

```
<% Application.Unlock%>
```

Ejemplo:

```
<%
Application.Lock
Application("visitas") = Application("visitas")+1
Application.Unlock
%>
Sos el visitante N° <%=Application("visitas")%>
```

4.- Funciones básicas

Fecha y hora

La fecha y hora son muy utilizadas en las páginas ASP.

- **Fecha**

FUNCION	DESCRIPCION
---------	-------------

Manual de Introducción al ASP (Active Server Pages)

Date	devuelve la fecha actual
Day(fecha)	devuelve el número de día
Month(fecha)	devuelve el número de mes
Year(fecha)	devuelve el año

· Hora

FUNCION	DESCRIPCION
Now	devuelve la fecha y hora actual
Time	devuelve la hora actual
Hour	devuelve la hora
Minute	devuelve los minutos
Second	devuelve los segundos

Ejemplos de fecha y hora

· Imprimir fecha y hora actual

Código

```
La fecha actual es <% =date()% > <br>
La hora actual es <% =time()% > <br>
```

Resultado

```
La fecha actual es 25/3/2001
La hora actual es 21:25:5
```

· Imprimir Día, Mes y Año por separado

Código

```
<%
' Declaramos variables
Option Explicit
Dim fecha, año, mes, día
' Guardamos la fecha actual dentro de fecha
fecha=Date()
' A partir de fecha, sacamos el día, mes y año
año=year(fecha)
mes=month(fecha)
día=day(fecha)
% >
Día: <% =día% ><br>
Mes: <% =mes% ><br>
Año: <% =año% ><br>
```

Resultado

```
Día: 25
Mes: 3
Año: 2001
```

If...Then...Else

Esta instrucción es conocida en todos los lenguajes de programación. Su tarea es ejecutar una instrucción a partir del resultado de una condición.

If condicion then instruccion

*if condicion then
instruccionA*

Manual de Introducción al ASP (Active Server Pages)

else
instruccionB
end if

Ejemplos:

```
<%  
Option Explicit  
Dim numero, dia  
numero=1  
dia="viernes"  
  
if numero = 1 AND dia="viernes" then  
color="verde"  
else  
color="negro"  
end if  
% >
```

```
<%  
Option Explicit  
Dim numero, dia  
numero=1  
dia="viernes"  
  
if numero = 1 OR dia="viernes" then  
Response Write "Color Verde"  
else  
Response Write "Color Negro"  
end if  
% >
```

Noten que cuando se trabaja con números se omiten las comillas (").

For...Next

Esta instrucción sirve para repetir un grupo de instrucciones una determinada cantidad de veces.

for contador = principio TO fin [Step incremento]
[instrucciones]
exit for
[instrucciones]
next

Ejemplos:

```
<%  
for cont = 1 TO 10  
Response Write "Esta es la línea número: " & cont  
next  
% >
```

```
<%  
for cont = 1 TO 10  
if cont=2 then  
Response Write "La línea N° 2"  
end if  
next  
% >
```

Manual de Introducción al ASP (Active Server Pages)

5.- Introducción a las bases de datos

Tipo de conexiones

Existen varios tipos de conexión a bases de datos. Algunos, como el ODBC requieren la configuración de un DSN en el servidor, otros en cambio, trabajan directamente si tener que realizar alguna configuración. Algunas de las conexiones que no requieren de DSN son OLEDB o DBQ.

· ODBC

```
<%
Set con = Server.CreateObject("ADODB.Connection")
' creamos el objeto de conexión
con.Open "NombreConección"
' abrimos la conexión por ODBC al DSN NombreConección
con.Close
' cierra la conexión
% >
```

· Por OLEDB

```
<%
set con=Server.CreateObject("ADODB.Connection")
' Conexión por OLEDB
con.Open "Provider=MicrosoftJetOLEDB.4.0;           Data
Source='&Server.MapPath("basededatos m db")& ';"
% >
```

· Por DBQ

```
<%
set con=Server.CreateObject("ADODB.Connection")
' Conexión por DBQ
con.Open "Driver={Microsoft Access Driver (*.mdb)}; DBQ=" &
Server.MapPath("basededatos m db")
% >
```

Recordset

Se utiliza para realizar operaciones sobre las tablas de la base de datos. Para crear el objeto RecordSet se utiliza la línea `set rs=Server.CreateObject("ADODB.Recordset")` seguida de una instrucción SQL para realizar alguna operación.

Instrucciones SQL

A través de estas instrucciones, podemos determinar que tipo de operación vamos a realizar sobre la tabla de la base de datos, puede ser seleccionar (select), borrar (delete), agregar (insert), entre otros. Para ejecutarla hay que crear primero el objeto del RecordSet, y luego de la instrucción SQL escribir:

```
set rs=conn.Execute(SQL)
```

Aquí explicamos brevemente algunas de ellas:

· Select

Recupera registros de una tabla.

```
SELECT [ALL | DISTINCT] lista_selección [INTO [nueva_tabla]] [FROM {tabla | consulta} [, {tabla2 | consulta2}
[... , {tabla16 | consulta16}]] [WHERE criterio] [GROUP BY criterio] [HAVING criterio] [ORDER BY criterio]
[COMPUTE criterio] [FOR BROWSE]
```

Algunos ejemplos simples:

```
SELECT * FROM TABLA
```

Selecciona todos los registros de Tabla

```
SELECT nombre, apellido FROM TABLA
```

Selecciona los registros nombre y apellido de Tabla

Realizado por Fabian Müller (fabian@webexperto.com) - www.webexperto.com

Manual de Introducción al ASP (Active Server Pages)

```
SELECT * FROM TABLA WHERE nombre="Fabian"
```

Selecciona los registros donde nombre sea igual a Fabian

```
SELECT * FROM TABLA ORDER BY apellido
```

Selecciona todos los registros y los ordena por el apellido

Delete

Elimina un registro de una tabla.

```
DELETE [FROM] {tabla|consulta} [WHERE criterio]
```

Algunos ejemplos simples:

```
DELETE FROM TABLA WHERE id=55
```

Elimina el registro con el ID 55

```
DELETE FROM TABLA WHERE nombre="Fabian"
```

Elimina todos los registros en donde el nombre sea igual a Fabian

Insert

Agrega registros a una tabla

```
INSERT [INTO] {tabla|consulta} [(columnas)] {DEFAULTVALUES | valores| instruccion_select}
```

Ejemplo:

```
INSERT TABLA (nombre, apellido, email) VALUES ('Fabian','Muller','fabian@webexperto.com')
```

Inserta los datos en los campos correspondientes

Update

Actualiza los registros de una tabla

```
UPDATE {tabla | consulta} SET [{tabla | consulta}] { columnas | variables| variables_y_columnas}[, {columnas2| variables2| variables_y_columnas2}.. [, {columnasN| variablesN| variables_y_columnasN}]] [WHERE criterio]
```

Ejemplo:

```
UPDATE TABLA SET nombre='Fabian' WHERE nombre='Marcelo'
```

Modifica los registros que contengan como nombre Marcelo por Fabian.

Ejemplo sencillo de consulta:

```
<%
' Creamos los objetos de conexión y recordset
set con=Server.CreateObject("ADODB.Connection")
set rs=Server.CreateObject("ADODB.Recordset")
' Conexión por OLEDB
con.Open "Provider=Microsoft.Jet.OLEDB.4.0;           Data
Source='&Server.MapPath("basededatos.mdb")&';"
sql="select * from Alumnos order by nombre"
rs=con.Execute(SQL)

' Hacemos un listado simple de los registros
Response.Write "<table border='1'>" & rs.Fields("apellido") & " "& rs.Fields("nombre") & "
"& rs.Fields("edad") & " años.<br>"

' Cerramos la conexión y el recordset
rs.Close
con.Close
set rs=nothing
set con=nothing
%>
```

Manual de Introducción al ASP (Active Server Pages)

» >

6 - Alojamiento ASP

En la actualidad están apareciendo muchos servidores que dan servicio gratuito de hosting con soporte de ASP y bases de datos Access, con la única condición de que aparezcan sus banners o ventanas con publicidad.

- **Websamba (<http://www.websamba.com>)**
Ofrece 30mb, soporta ASP y conexiones a bases de datos Access sin limitaciones, y la posibilidad de subir nuestras páginas por FTP. El único inconveniente que encontramos en este servidor son sus continuas caídas, quedando a veces varias horas sin funcionar. No pone publicidad.
- **Brinkster (<http://www.brinkster.com>)**
Al igual que Websamba, ofrece 30mb de espacio, soporte asp y bases de datos access. No incluye publicidad en las páginas, pero nos vemos obligados a subir los archivos a través de un File Manager desde su sitio.
- <http://www.webexperto.com/directorio/default.asp?catid=43&cattitle=Web+Hosting>
Aquí encontrarán un listado de servicios gratuitos de hosting con y sin soporte de ASP.

Si tenemos dinero para invertir, es recomendable contratar un servicio de hosting, ya que son muchísimos más rápidos y además, brindan soporte técnico especializado.

7 - Acerca de este manual

El presente manual de ASP fue realizado por Fabian Müller, de <http://www.webexperto.com>

Para mayor información acerca de este u otros manuales visite <http://www.webexperto.com/manuales>

En <http://www.webexperto.com/secciones/articulos/ver.asp?temav=asp> podrán encontrar numerosos artículos referente a este lenguaje, con ejemplos y explicaciones muy completas.