Taller de ASP

Manual por: DesarrolloWeb.com [http://www.desarrolloweb.com/] **Versión on-line:**
"Tu mejor ayuda para aprender a hacer webs" http://www.desarrolloweb.com/manuales/11

Mejorar el registro en buscadores

El principio de una página con éxito es sin duda la calidad e interés de sus contenidos. No obstante, a este éxito contribuyen muchos otros factores como el diseño o la promoción.

Este ultimo aspecto, la promoción, tiene como uno de sus ejes principales de desarrollo el correcto registro en motores de búsqueda. Los motores se encargan de, llegados a la página a registrar, recorrerla de arriba abajo a partir de los enlaces que en ella figuran y de esta forma almacenar los datos que a él le interesen: etiquetas meta, texto...

El problema con las páginas dinámicas es que, en muchas ocasiones, los enlaces presentan URLs complejas por las que se están pasando variables y que resultan difícilmente digeribles por los motores. Resultado : la cantidad de páginas registradas por el buscador se ve sensiblemente disminuida.

Para determinados sitios, esto no resulta un verdadero problema sin embargo, para otros en los que una única visita interesada puede suponer un beneficio importante o bien sitios de tiendas virtuales en los que se venden artículos un tanto exóticos, el no estar eficientemente registrado puede suponer una seria limitación. En cualquier caso, a todos nos gusta que nuestra página aparezca bien situada en una búsqueda.

Este supuesto problema de registro debido a la utilización de páginas dinámicas puede resultar, contrariamente, una ventaja valiosisima. Lo único que hay que hace es evitar pasar parámetros por medio de las URLs. Pero... ¿Cómo?

Sencillamente, en lugar de proponer enlaces del tipo:

http://www.misitio.com/script.asp?variable=valor

Lo que haremos es mostrar un enlace como este:

http://www.misitio.com/valor/

Además, crearemos un directorio llamado "valor" dentro del cual introduciremos un archivo que contendrá asignación del valor de la variable e incluiremos el script principal como un archivo anexo. El archivo el interior del directorio seria el siguiente:

```
<%
variable=valor
%>
<!-- #include file="../script.asp" -->
```

Como ejemplo podemos pensar en una tienda virtual, sin ir mas lejos nuestra conocida librería virtual. En este caso, podríamos crear un directorio por titulo existente que podría llevar como nombre la referencia del libro. Dentro de cada directorio introduciríamos un archivo del tipo anteriormente visto:

```
<%
id=referencia_del_libro
%>
<!-- #include file="../busqueda.asp" -->
```

En el script de busqueda.asp podríamos además introducir unas etiquetas meta que hiciesen llamadas a la base de datos de la cual podríamos sacar informaciones especificas del libro en cuestión, por ejemplo:

```
<META name="description" content="<%=RecordSet("descripcion")%>">
```

Con esto no solamente permitimos que el buscador pueda tener acceso a las páginas de todos los artículos sino que, también personalizamos las etiquetas meta de una manera automática y permitimos así posicionar nuestra página de una manera eficiente ante determinadas búsquedas de internautas interesados por una obra en particular.

Por otro lado, la creación de los directorios y del archivo que contiene la variable pueden ser fácilmente realizadas y coordinadas al sistema intranet a partir del cual realizamos las actualizaciones de nuestra base de datos. Esta automatización podría ser culminada con un procedimiento que permitiese, en caso de eliminación de un libro de nuestra base datos, modificar el archivo que contiene la variable por otro que redirecciona a la página de entrada de nuestro sitio. De esta forma no solo ganamos visitantes por nuestro catálogo actual sino que conseguimos atraerlos por existencias anteriores!!

Como puede verse, trabajar con un lenguaje dinámico para la creación de un sitio basado en la explotación de una base de datos puede resultar asombrosamente útil si se realiza de una forma inteligente. Con una sola página y un script de automatización estamos dispuestos a registrar en motores de búsquedas tantas URLs como registros tengamos en nuestra base todo esto de una forma sencilla y eficaz. Además, este tipo de estrategia puede ser fácilmente adoptada por sitios ya existentes sin para ello trastocar para nada el código.

Include. Incluir archivos ASP

Incluir archivos en una página consiste en insertar un trozo de código que tenemos alojado en un archivo aparte dentro del código de la página ASP que estamos programando. De este modo, podemos guardar en un archivo códigos de las páginas que utilizamos habitualmente, como podría ser la cabecera de nuestras páginas o el marco con el que se presetan todas las páginas del sitio, e incluso scripts en ASP que se dediquen a realizar algunas acciones muy repetidas, como conexiones con bases de datos. Trabajar realizando includes es muy práctico, pues no escribimos dos veces el mismo código y además, si tenemos que cambiarlo en algún momento sólo tendríamos que cambiar el código de ese archivo aparte y todas las páginas se modificarán automáticamente.

Para incluir archivos en ASP se han de utilizar los típicos include que ya hemos visto en capítulos anteriores pero que vamos a revisar en este capítulo. También vamos a mostrar limitaciones de esta técnica y a mostrar alguna posible solución.

Incluir archivos

Se utiliza con el enunciado de include, que puede tener dos formas:

```
<!--#include virtual="/plantilla.asp"-->
<!--#include file="plantilla.asp"-->
```

Las diferencias entre realizar el include "virtual" en lugar de "file" consisten en que la primera tiene en cuenta la raiz de tu dominio y con la segunda se tiene en cuenta el lugar donde está el documento desde el que estamos incluyendo el archivo. En un principio puede ser más útil trabajar con el modo "virtual" pues si movemos de directorio las páginas de nuestro sitio los includes seguirían funcionando. Si vemos varios ejemplos podremos entenderlo mejor.

```
<!--#include virtual="/plantillas/cabecera.asp"-->
```

Incluimos un archivo llamado cabecera que está en el directorio plantillas, que está a su vez en la raiz del dominio. Con este método el include funcionará independientemente de donde esté el archivo que lo incluye.

<!--#include file="plantillas/cabecera.asp"-->
Esto solo funcionará si el archivo que realiza el include está en la raiz del dominio. Pero fallaría si estamos en el cualquier subdirectorio del dominio.

<!--#include file="../plantillas/cabecera.asp"-->
Así conseguiríamos que funcionase si el archivo desde el que incluimos la cabecera está en un subdirectorio inmediatamente después de la raiz del dominio.

Include y la seguridad

En principio podemos utilizar cualquier extensión en el archivo vamos a incluir. Extensiones corrientes son .txt, .inc, .html o .asp. Sin embargo, es recomendable por motivos de seguridad que los nombremos como .asp, sobretodo si hemos insertado código ASP dentro del archivo a incluir.

El motivo por el que es más seguro nombrar los includes con extensión .asp es que si un usuario intenta acceder al archivo que se incluye directamente tecleando su URL en la barra de direcciones del navegador, algo así como http://www.dominio.com/archivos_a_incluir/funciones_comunes.inc, recibiría como respuesta el código completo y sin ejecutar del include y, probablemente, no nos interese que vean ese código de nuestro sitio web. Es especialmente delicado este problema cuando dentro del código colocamos información crítica como puede ser la cadena de conexión a la base de datos.

Con la extensión .asp nos aseguramos que, en caso de que un usuario consiga acceder a los includes, estos se ejecutarán en el servidor y se eliminarán todos los códigos ASP que no deseamos distribuir.

Modo de funcionamiento y problemática

El modo de funcionar de este tipo de includes consiste en que primero se insertan los archivos en la página, creando un conjunto que sería formado por el archivo fuente y todos los includes insertados dentro de él. Una vez creado ese gran fichero se ejecutan los scripts ASP.

Podría haber un problema si queremos incluir un archivo cuyo nombre debemos extraer de una variable ASP. Imaginaros que tenemos que incluir un archivo que tenemos su nombre en una variable. Así:

```
<%
archivoInclude = "miarchivo.html"
%>
<!--#include virtual=archivoInclude-->
```

Como el código ASP no se ejecuta hasta después de haber incluido el archivo, a la hora de incluirlo no se ha resuelto el nombre que se desea incluir. Dicho de otro modo, ASP no sabe qué es lo que vale la variable archivoInclude cuando va a incluirlo, por lo que no realiza el include tal como desearíamos.

Para parametrizar bien una página web y que su actualización sea fácil más tarde o más temprano lo anterior será un inconveniente. Así que podemos arreglar el problema del siguiente modo, creando una función que realice el include del archivo. Veamos cómo sería esa función:

```
<%
function incluye(archivo)
  archivo= request.serverVariables("APPL_PHYSICAL_PATH") & archivo
  set confile = createObject("scripting.filesystemobject")
  set fich = confile.openTextFile(archivo)
```

```
  while not fich.atEndOfStream
    lineactual = fich.readLine
    response.write(lineactual)
  wend
end function
%>
```

Varias cosillas habría que comentar sobre esta función. Recibe el nombre del archivo que deseamos abrir. Para construir la ruta completa de este archivo tendremos que utilizar el objeto request para extraer la variable del servidor que contiene la ruta de disco duro donde se encuentra el domino. Una vez que hemos construido el nombre de archivo con su ruta creamos el archivo en las 2 siguientes líneas. Finalmente hacemos un bucle que va extrayendo del archivo el texto correspondiente e imprimiendolo en la página.

Solo destacamos un punto en contra de esta forma de incluir los archivos. Se trata de que no podremos incluir más que código HTML, nunca código ASP pues no se ejecutaría y quedaría impreso en la página junto con el código HTML. Es una limitación significativa pero en los casos en los que solo necesitemos incluir texto puede servir muy bien.

Crea tu propio buscador

Cuando trabajamos con sitios basados en bases de datos y nuestros contenidos empiezan a crecer, puede resultar muy práctico para el navegante poder recurrir a un formulario en el que introducir palabras claves y operadores que le ayuden a matizar el elemento que está buscando. Este tipo de campos pueden verse en multitud de páginas y, aunque distintos en su funcionamiento en muchos casos, todos tienen algo en común: Se trata de programas que permiten procesar una variable de tipo cadena y transformarla en una orden de búsqueda para la base de datos.

En este reportaje os proponemos un par de funciones que, usadas conjuntamente, permiten la creación de una instrucción SQL. El script permite especificar cuáles serán los campos de búsqueda y dentro de qué tabla la realizaremos.

Este programa debería ir combinado con otro pequeño script de toma de datos por formulario como los vistos en nuestro **manual de ASP** [http://www.desarrolloweb.com/articulos/253.php] a partir del cual obtendríamos la variable *cadena* introducida por el internauta. En este sentido, en DesarrolloWeb.com hemos publicado un manual que explica la creación de un buscador simple en ASP [http://www.desarrolloweb.com/manuales/25], empezando desde la definición de la base de datos, continuando por la inserción de los datos en la base y la búsqueda por palabras clave.

Estas dos funciones pueden ser utilizadas directamente para cada caso particular a condición de especificar los campos de búsqueda en el array *campos*, especificar la *tabla* y modificar la función *generasql* para que realice la selección de los campos que deseemos.

En este caso, con el objeto de facilitar la comprensión, hemos simplificado al máximo las funciones que el buscador puede realizar. En realidad, el buscador sólo tratará campos usando el operador *like*. Por otra parte, únicamente empleará como operadores "+" y "-".

Todo tipo de modificaciónes más o menos complejas pueden ser introducidas de manera a mejorar su versatilidad:

-Empleo de paréntesis para mayor eficacia de los operadores

-Eliminación de secuencias repetidas como "++" o "--"

-Eliminación de operadores al principio y final de la cadena

-Utilización de comodines...

Pasemos ahora a mostrar el listado para comentarlo más detalladamente a continuación:

```
<%
function Sacar(cadena,campos)
dim i
dim SacarAux
while InStr(cadena,"  ")
   Cadena=Replace(Cadena,"  "," ")
wend
if len(cadena)>0 then
   if InStr(cadena," ")>0 then
      Sacar= Sacar(left(cadena,InStr(cadena," ")-1),campos) & " OR " & Sacar(right(cadena,len(cadena)-InStr(cadena," 
")),campos)
   elseif InStr(cadena,"+")>0 then
      Sacar=Sacar(left(cadena,InStr(cadena,"+")-1),campos) & " AND "& Sacar(right(cadena,len(cadena)-InStr
(cadena,"+")),campos)
   elseif InStr(cadena,"-")>0 then
      Sacar=Sacar(left(cadena,InStr(cadena,"-")-1),campos) & " AND NOT " & Sacar(right(cadena,len(cadena)-InStr(cadena,"-
")),campos)
   else
   'fijamos la sentencia
   SacarAux=""
   i=1
   SacarAux= "( " & campos(i) & " Like '%" & cadena & "%'"
   i=i+1
   while len(campos(i))>0
      SacarAux= SacarAux & " OR " & campos(i) & " Like '%" & cadena & "%'"
      i=i+1
   wend
   SacarAux=SacarAux & " )"
   Sacar=SacarAux
   end if
else
sacar=""
end if
end function

function GeneraSql(cadena,tabla,campos)
if len(cadena)>0 then
   generaSql="Select * from " & tabla & " Where " & Sacar(cadena,campos)
else
   Response.Write "No hay criterios"
end if
end function

dim campos(3) 'el tamano del array debe superar en uno al numero de campos
campos(1)="nombre_campo1"
campos(2)="nombre_campo2"

'para mostrar cual seria el resultado...
cadena="hola cariola+cocacola-colacao"
tabla="cualquiera"
resultado=GeneraSql(cadena,tabla,campos)

Response.Write resultado
%>
```

Aquí puedes **descargar** [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/buscador/buscador.zip] y **ejecutar** [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/buscador/buscador.asp] este script.

Como hemos dicho, el script consta principalmente de dos funciónes. La primera de ellas, *sacar*, se encarga de tratar la cadena para separar las palabras y, teniendo en cuenta los operadores, construir el fragmento final de la sentencia SQL. La búsqueda se realiza dentro de una serie de campos que son definidos al exterior de la función en forma de array.

En esta función se emplean diferentes funciones de tratamiento de variables de tipo cadena las cuales explicamos a continuación:

InStr(cadena,subcadena)	Devuelve el número de las posiciones en las que una determinada subcadena aparece en la cadena

	principal
Replace (cadena,subcadena1,subcadena2)	Sustituye un fragmento (subcadena1) de una cadena por un nuevo fragmento (subcadena2)
Len(cadena)	Nos devuelve la longitud de la cadena
Left(cadena,numero_caracteres)	Selecciona una determinada cantidad de carácteres de nuestra cadena comenzando por la izquierda
Right(cadena,numero_caracteres)	Selecciona una determinada cantidad de carácteres de nuestra cadena comenzando por la derecha

Es interesante observar la técnica de recursividad que se utiliza para desglosar la cadena principal en palabras independientes. Esta técnica se basa en que función *sacar* se llama a si misma para cada una de las subcadenas de la cadena principal. Aparte de esa pequeña astucia el resto de la función es de lectura fácil.

La segunda función, *generasql*, se encarga de añadir al fragmento final creado por nuestra función *sacar* un fragmento inicial en el que especificamos la tabla y el tipo de selección que queremos realizar. En este caso se han selecciónado como resultado todos los campos de la tabla pero, evidentemente, esto no tiene por qué ser así.

La última parte del script consiste en la especificación de los campos dentro de los cuales deseamos llevar a cabo nuestra búsqueda. A notar que el array ha de tener una longitud mayor al número de campos para el correcto funcionamiento del programa.

> **Referencia:** Hemos publicado un artículo en DesarrolloWeb.com donde explicamos como integrar el presente script en un buscador en ASP. El artículo en concreto es Buscador simple en ASP mejorado [http://www.desarrolloweb.com/articulos/930.php?manual=25], y el buscador que mejora está comentado en el manual Buscador simple en ASP [http://www.desarrolloweb.com/manuales/25].

Esperamos que este buscador os sirva de base para realizar el propio vuestro con todo tipo de personalizaciones que consideréis oportunas.

El Objeto RecordSet

En nuestro **manual de ASP I** [http://www.desarrolloweb.com/manuales/8/] hemos introducido las nociones imprescindibles para la interacción con bases de datos. Hemos explicado cómo conectar con la base de datos y de qué forma podemos llevar a cabo las consultas mediante sentencias SQL.

Sin embargo, temiendo cargar en exceso un manual orientado a aprendices, no hemos presentado de una manera oficial los ADO (ActiveX Data Objects).

Cuando interaccionamos con una base de datos hay tres acciones principales que son llevadas a cabo por tres objetos diferentes:

-Primeramente se realiza una **conexión** a la base de datos. Esta tarea es realizada por el objeto **Connection** al cual tendremos que especificar la base de datos a la que nos queremos conectar mediante el interface ODBC.

-El objeto **Command** se encargará a continuación de **ejecutar** la sentencia SQL pertinente.

-Los **resultados** de la selección son almacenados en el objeto **RecordSet** en forma de una tabla que puede ser consultada y explotada de muchas maneras.

La tabla RecordSet se sirve fundamentalmente de un cursor que se sitúa inicialmente en el primer registro y que puede ser desplazado de múltiples formas a lo largo de la tabla para, de

este modo, extraer las informaciones que puedan interesarnos.

Para mover este cursor he aquí los métodos que pueden ser empleados:

Método	Descripción
MoveFirst	Posiciona el cursor en el primer registro
MoveLast	Posiciona el cursor en el último registro
MoveNext	Avanza el cursor en un registro
MovePrevious	Retrasa el cursor en un registro

Los métodos y propiedades que ofrece este objeto son asombrosamente numerosos. Aquí sólo comentaremos los más frecuentemente utilizados dejando al lector **documentarse** [http://msdn.microsoft.com/library/default.asp?url=/library/en-us/ado270/htm/mdobjodbrec.asp] sobre el resto.

Elemento	Descripción
Open	Abre el objeto RecordSet
Close	Cierra el objeto RecordSet
Eof	End Of File. Devuelve el valor True cuando el cursor ha llegado más allá del último registro.
Bof	Begining Of File. Devuelve el valor True cuando el cursor está posicionado antes del primer registro. Muy útil cuando se leen tablas al revés.
AbsolutePage	En caso de estar paginando los resultados, nos indica la página actual en la que nos encontramos.
PageCount	En caso de estar paginando los resultados, nos indica el numero de página s total.
RecordCount	Nos devuelve la cantidad de registros contenidos en el objeto RecordSet.
Fields.Count	Nos devuelve la cantidad de campos contenidos en el objeto RecordSet

Puede que despues de leer este artículo, todavía resulten vagos algunos conceptos relativos a la aplicación de estos métodos y propiedades. Esperamos que nuestro siguiente capítulo, en donde pondremos en práctica muchos de estos elementos, permita aclararos todo tipo de dudas.

Paginar resultados en ASP

En **artículos precedentes** [http://www.desarrolloweb.com/articulos/293.php] hemos visto que cuáles son las propiedades y métodos principales del objeto RecordSet, ahora sólo nos queda ponerlos en práctica para afianzar nuestros conocimientos.

En este capítulo os proponemos y describimos un script que divide en distintas páginas los registros obtenidos a partir de una consulta a la base de datos mediante la explotación del objeto RecordSet.

Se trata de un derivado del clásico script que podemos encontrar en cualquier tipo de buscador. De hecho, este script puede ser muy fácilmente fusionado con el prototipo de **buscador** [http://www.desarrolloweb.com/articulos/292.php] propuesto en este mismo taller de manera a gestionar la construcción de una sentencia SQL que, después de su ejecución, va a poder ser paginada por medio de este script.

A continuación os presentamos el listado que, para más claridad, utiliza una única sentencia SQL. Dejamos a vuestra cuenta la fusión de este script con el del buscador.

```
<html>
<head>
  <title>Repaginador de resultados</title>
</head>
<%
sSQL="select lo que tu quieras"
'esta sentencia SQL puede ser creada a partir de un buscador como el que hemos visto en otro reportaje
'y almacenada en una session para emplearla sucesivas veces en el script: session("ssql")=ssql

'actualizamos numero de pagina
If Request.QueryString("pag")<>"" Then
  Session("pagina")=Request.QueryString("pag")
Else
  Session("pagina")=1
End If

'constantes ADO VBScript
Const adCmdText = &H0001
Const adOpenStatic = 3

Set Conn = Server.CreateObject("ADODB.Connection")
Set Command = Server.CreateObject("ADODB.Command")
Set RS =Server.CreateObject("ADODB.RecordSet")
Conn.Open "nombre de tu base de datos"
RS.Open sSQL,Conn,adopenstatic,adcmdtext

'resultados por pagina a elegir arbitrariamente
num_registros = 5

'Dimensionamos las paginas y determinamos la pagina actual
RS.PageSize=num_registros
RS.AbsolutePage=Session("pagina")
%>

<body>
<div align="center">
Número de página actual: <b><%=Session("pagina")%></b>
<br>
Número de páginas total: <b><%=RS.PageCount%></b>
<br>
Número de registros por página: <b><%=RS.PageSize%></b>
<br>
Número de registros seleccionados: <b><%=RS.RecordCount%></b>
</div>
<br><br>

<table cellspacing="2" cellpadding="2" border="1" align="center">
<%
'Contamos el numero de campos
num_campos=RS.Fields.Count
For campo=0 to num_campos-1%>
  <td align="center"><%=RS(campo).Name%></td>
<%
Next
registros_mostrados = 0
While (Not RS.eof And registros_mostrados < num_registros)
  registros_mostrados = registros_mostrados +1
%>
  <tr>
  <%For campo=0 to num_campos-1 %>
    <td align="center"><%=RS(campo)%></td>
  <%Next%>
</tr>
<%
  RS.MoveNext
Wend
%>
<tr>
<td colspan="<%=num_campos%>" align="center">
<%
  i=0
While i<RS.PageCount
  i=i+1
```

```
%>
  <b><a href="paginar.asp?pag=<%=i%>"><%=i%></a></b>
<%
Wend
%>
</td>
</tr>
</table>
<%
RS.Close
Conn.Close
%>
</body>
</html>
```

Puedes **descargar** [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/paginar/paginar.zip] y **ejecutar** [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/paginar/paginar.asp] el script conectado una tabla ejemplo creada por nosotros.

El primer paso ha sido definir le sentencia SQL cuyos resultados vamos a páginar. Si queremos fusionar este script al del buscador deberemos guardar la orden generada por nuestras funciones en una variable session para no perderla cada vez que pinchamos sobre uno de los enlaces para movernos a otra página con resultados.

A continuación definimos la página en la que nos encontramos. Esta será la primera por defecto o tendrá un valor definido (pasado por URL) si el internauta se encuentra navegando por los resultados.

La apertura del objeto RecordSet ha de hacerse en este caso de una forma distinta a como hemos visto en casos anteriores. Una serie de parámetros en forma de constantes son requeridos para su correcta apertura. No vamos a entrar en detalles sobre lo que ello significa únicamente diremos que estos parametros forman parte de una colección de constantes ADO que podéis descargar **aquí** [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/paginar/adovbs.txt] en formato texto. Diremos también que ésta es la forma clásica de apertura que nos permite seleccionar los registros mediante SQL y movernos en todos los sentidos con el puntero del objeto RecordSet.

Ejecutada la sentencia SQL nos quedan por dimensionar las páginas. Deberemos especificar el número de registros a mostrar en cada una (*PageSize*) y en cuál de ellas nos encontramos (*AbsolutePage*). El objeto se encargará de calcular otra serie de propiedades como son el número de páginas total (*PageCount*) y el número de registros presentes en nuestro objeto (*RecordCount*).

Hemos querido aprovechar este artículo para presentar dos propiedades útiles para la gestión de campos: La coleccion *Fields*, que nos da acceso al conjunto de campos cuyo número puede ser fácilmente computado (*RS.Fields.Count*) y, por otro lado, la propiedad *Name* que nos permite extraer el nombre de dichos campos tal y como podemos ver en el primer bucle *for*.

La forma de mostrar los resultados en pantalla no difiere de lo visto en otros scripts. Tan sólo hay que acordarse de introducir una condición suplementaria en el bucle *While* y es que la cantidad de registros mostrados no sobrepase el número de registros por página previamente definido.

Finalmente, antes de cerrar los objetos, generamos dinámicamente los enlaces que apuntan hacia esta misma página por los que pasamos el número de página que queremos visualizar. Este bucle, simplificado al máximo, puede ser complementado con un enlace a la página anterior, otro a la siguiente, uno a la primera y otro a la última. Podemos también generar el numero de página actual no en forma de enlace sino en forma de texto simple puesto que esto le permite al visitante ver rápidamente en qué página se encuentra y evita el mostrar un enlace que apunta al mismo contenido que se está viendo.

Otras mejoras posibles son: Calcular el intervalo de resultados que se está mostrando en la página (La frase típica: "He aquí los resultados de x a y"), un mensaje de "Ningún resultado obtenido" para casos en los que el *Recordset* esté vacio (*RS.Eof*), presentar un formulario para efectuar una nueva búsqueda sin salir de la página...

Esperamos que saquéis provecho de este script para vuestro sitio web.

Global.ASA

Para hablar del Global.ASA es necesario que recordemos o aclaremos previamente un par de conceptos como son las aplicaciones y las sesiones.

Aplicaciones y sesiones

Una aplicación se puede entender como más o menos lo que es un sitio web. Los dominios de la aplicación son el directorio raíz y los subdirectorios de esta. En un servidor web como PWS tenemos una aplicación en la raíz del servidor, y por cada directorio virtual que creemos tendremos otra aplicación.

El funcionamiento de las aplicaciones está ligado al de las sesiones. Las aplicaciones son globales a todo el sitio y las sesiones son particulares de cada usuario. Concretamente funcionan con este proceso:

- La aplicación está parada, así como las sesiones, porque no hay ningún usuario dentro.
- La aplicación se pone en marcha cuando entra el primer usuario.
- El primer usuario pone en marcha una sesión
- Los siguientes usuarios ya tienen en marcha la aplicación, con lo que solo desatan la creación de una sesión cada uno.
- Con cada usuario que abandona la página (para ello se pasa 20 minutos o más sin consultar ninguna otra página), se cierra su sesión.
- Con el último usuario que sale se cierra la aplicación.

Se pueden crear variables de aplicación y de sesión en cualquier momento. Remarcamos, las variables aplication van a ser comunes para todos los recursos de la aplicación, todos los usuarios, etc. todos acceden a la misma variable. Las variables de sesión son locales a cada usuario, por lo tanto cada usuario tiene una copia de la variable sesión distinta de la de otro usuario, que puede tener valores distintos.

Para crear variables de aplicación

```
Application("nombreDeVariable") = valor
```

Para asegurarse exclusión mutua al acceder a una variable de aplication se utilizan los métodos lock y unlock, de esta manera:

```
Application.lock
Application("nombreDeVariable") = valor
Application.unlock
```

Crear variables de sesión

```
Session("nombreDeVariable") = valor
```

Qué es el global.asa

Global.ASA nos va a servir para controlar los eventos principales asociados con el inicio y fin de la aplicación, así como con el inicio y fin de sesión. Dicho de otro modo, con global.asa podemos hacer cosas cuando se inicien y acaben las aplicaciones y las sesiones.

El global.asa se coloca en el directorio raíz de la aplicación y tiene la siguiente sintaxis

```
<SCRIPT LANGUAGE=VBScript RUNAT=server>

sub application_onStart()
'sentencias que se ejecutan al entrar el primer usuario
end sub

sub application_onEnd()
'sentencias a ejecutar al irse el último usuario
end sub

sub session_onStart()
'sentencias que se ejecutan cada vez que entra un usuario
end sub

sub session_onEnd()
'sentencias a ejecutar al irse el cada usuario
end sub

</SCRIPT>
```

Un detalle a destacar por su utilidad práctica es que si queremos sustituir nuestro global.asa por otro se tendrá que apagar el servicio y volver a encender, o incluso volver a arrancar el ordenador si lo anterior no funciona. Si no hacemos esto siempre tomará la versión antigua del global.asa, aunque se haya cambiado. Por esta misma razón, hay que tener cuidado que no tenga errores el archivo global.asa, pues nuestra página dará errores hasta que reiniciemos el servicio, con el consiguiente espacio de tiempo en el que estará el dominio caído.

Contar usuarios activos

Podemos ver un ejemplo de global.asa utilizado para llevar el control de los usuarios que acceden a la página web. La cuenta nos informa del número de usuarios que están activos en el sitio, es decir del número de sesiones abiertas.

> **Referencia:** El archivo global.asa, su utilidad y funcionamiento, ha quedado resumido en otro taller de ASP. http://www.desarrolloweb.com/articulos/295.php?manual=11

```
<SCRIPT LANGUAGE=VBScript RUNAT=server>

sub application_onStart()
'sentencias que se ejecutan al entrar el primer usuario
application("num_usuarios")=0
end sub

sub session_onStart()
'sentencias a ejecutar al irse el último usuario
application.lock
application("num_usuarios") = application("num_usuarios") + 1
application.unlock
end sub

sub session_onEnd()
'sentencias que se ejecutan cada vez que entra un usuario
application.lock
application("num_usuarios") = application("num_usuarios") - 1
application.unlock
end sub

</SCRIPT>
```

Es muy fácil de entender, cuando de inicia la aplicación se inicializa a cero la variable que cuenta el número de usuarios activos. No es necesario bloquear el objeto application porque en este momento no se tiene procesos concurrentes que pudiesen estar modificando la aplicación, por que aun no ha llegado nadie. En el inicio de una sesión se incrementa en uno el número de

usuarios y en el fina se decrementa en uno. No hace falta definir acciones al acabar la aplicación porque la variable de aplicación que guarda el numero de usuarios desaparece sola al acabar la aplicación.

Visualizar el número de usuarios

Ahora, si deseamos ver el número de usuarios en cualquier momento lo único que tendremos que hacer es sacar en la página el contenido de la variable de aplicación que cuenta el número de usuarios, en un script parecido al siguiente.

```
<html>
<head>
<title>Muestro usuarios activos</title>
</head>

<body>
<h1>Usuarios activos
<%
application.lock
response.write application("num_usuarios")
application.unlock
%>
</h1>

</body>
</html>
```

No tiene tampoco ninguna dificultad. Si queremos lo podemos probar en nuestro ordenador. Ya sabemos que el global.asa se coloca en el directorio raíz de una aplicación o en un directorio virtual que hayamos creado con un servidor como Personal Web Server. Si queremos ver como se incrementa el número de sesiones sin necesidad de que otros usuarios se conecten podemos entrar en la página con navegadores distintos y se tomarán como sesiones distintas aunque el ordenador desde donde se accede es el mismo.

Recorrido arrays y colecciones

En nuestro manual de ASP hemos visto varias estructuras de control, una de ellas todavía la tenemos que ver en algunos ejemplos más. Se trata de la estructura FOR...EACH que sirve para recorrer todos los elementos de una colección, por ejemplo un array o una colección como las variables del servidor. Para acabar veremos un ejemplo práctico adicional que consiste en el recorrido genérico de las variables que recibimos por POST o por GET.

FOR...EACH

El bucle FOR...EACH se utiliza con la siguiente sintaxis:

```
For Each elemento in colección
   'Hacemos cosas para cada elemento de la colección
Next
```

Recorrido genérico de Arrays

Un ejemplo para ver el funcionamiento de este bucle podría ser el siguiente recorrido genérico de un array

```
<%
dim nombres(3)
nombres(1) = "pepe"
nombres(2) = "luis"
nombres(3) = "margarita"
for each valor in nombres
  response.write valor & "<br>"
next
```

```
%>
```

Se recorre el array de nombres recogiendo cada valor e imprimiendolo. Se puede ver el ejemplo en funcionamiento [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/arrays/ejforeach.asp].

Recorrido genérico de colecciones

Las colecciones son como arrays con la diferencia que se puede acceder al valor que guarda el array en cada posición con una cadena de caracteres en vez de un número.

Un ejemplo muy interesante de recorrido de colecciones lo vimos en su momento para mostrar todas las variables del servidor, las ServerVariables.

```
<%
For Each elemento in Request.ServerVariables
  Response.Write elemento&" : "&Request.ServerVariables(elemento)& "<br>"
Next
%>
```

Este ejemplo es más interesante, por que se trata de una colección y no un array típico. La colección de ServerVaribles se accede de la siguiente manera.

request.servervariables("nombre de la variable")

donde "nombre de la variable" podría ser HTTP_ACCEPT_LANGUAGE, APPL_PHYSICAL_PATH o cualquier otra variable de la lista de serverVariables, que se puede ver en el ejemplo de este script [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/arrays/servervariables.asp].

Recorrido genérico de las variable enviadas por POST o GET

Otro ejemplo interesante y práctico es el que viene a continuación. Se trata de recoger las variables que le llegan a la página de un formulario o por la URL de manera genérica, es decir, sin importarnos cuántos elementos llegan y cuáles son. Este ejemplo lo puedes llegar a necesitar cuando no sepas que tipo de formulario te están mandando ni sepas cuáles ni cuántas variables te envian por la URL.

En nuestro ejemplo vamos a centrarnos en un formulario con method POST, con lo que recogeremos las variables con el método request.Form. Los ejemplos para una lista de variables llegada por la URL, con el método GET, se pueden recoger de un modo similar, usando request.QueryString.

```
<%
For Each elemento in Request.form
  Response.Write elemento & ": " & Request.form(elemento) & "<br>"
Next
%>
```

Es exactamente el mismo ejemplo que para recorrer una colección como la ServerVaribles, pero mostrando ahora la colección request.form.

Podremos utilizar este script cuando queramos realizar recorridos genéricos. En las siguientes líneas se puede ver un ejemplo de formulario que utilizamos para ilustrar este ejemplo por la práctica.

```
<form action="recorridoform.asp" method="POST">
Nombre: <input type="Text" name="nombre">
<br>
<br>
Edad: <input type="Text" name="edad" size=3>
<br>
<br>
Profesion: <input type="Text" name="profesion">
<br>
```

```
<br>
<input type="Submit" value="Enviar">
</form>
```

Se puede ver el ejemplo de las dos páginas funcionando [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/arrays/form.html].

Más sobre cookies

Ya hemos visto en el manual básico de ASP un capítulo dedicado a las cookies [http://www.desarrolloweb.com/articulos/254.php], que sería adecuado repasar para tener la base necesaria para poder entender este nuevo capítulo.

Las cookies están en una colección

Las cookies se guardan en una colección, al igual que muchas otras cosas en ASP. Podríamos acceder a todas las cookies de una página a través del bucle For Each.

```
<%
response.cookies ("id") = "Pepe"
response.cookies ("idioma") = "ing"
response.cookies ("color") = "verde"

cadena = ""
for each galleta in Request.cookies
  cadena = cadena & "Cookie " & galleta & ": " & Request.cookies(galleta) & "<br>"
next
response.write cadena
%>
```

En este ejemplo primero creamos unas cookies y luego hacemos un recorrido en la colección de cookies para mostrar su valor. Los valores de intermedios se guardan en una variable cadena por que es aconsejable no escribir nada en la página hasta no haber realizado todo el trabajo con las cookies. Una vez recorrido toda la colección se muestra la variable cadena en la página, que contiene todas las cookies de la página.

Probablemente te muestren una cookie más aparte de las que se acaban de crear, que es una cookie que se guarda en desarrolloweb para conservar la personalización del color que introdujo el usuario. Puedes ver este ejemplo [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/cookies/cookies1.asp].

Subclasificación de cookies

Para continuar, vamos a ver cómo se puede meter en una cookie un mayor número de datos a través de la subclasificación. Donde antes creabamos una cookie ahora aprenderemos a crear una subcolección de cookies.

```
<%
Response.cookies ("usuario")("id") = "eugim"
Response.cookies ("usuario")("idioma") = "esp"
Response.cookies ("usuario")("color") = "rojo"
%>
```

Se puede ver este ejemplo en funcionamiento [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/cookies/cookies2.asp]

Si ahora volvemos a ver el listado de las cookies de este web [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/cookies/cookies1.asp] podremos comprobar cómo se almacenan las cookies en una subcolección, con pares nombre=valor.

Bucle anidado para mostrar subcolección

Si deseamos mostrar las subcolecciones de una manera más bonita pondemos hacer un bucle anidado. El bucle más general sería el que va recorriendo las cookies de primer nivel. En cada iteración del bucle se preguntará si la cookie actual tiene subcookies. Si no tiene, escribe la cookie actual y continúa con la siguiente iteración. En caso de que hubiera subcookies, se mete en un bucle que muestra cada una de ellas. El código para hacer esto sería el siguiente.

```
<%
cadena = ""
for each galleta in request.cookies
  if request.cookies(galleta).HasKeys then
    for each subgalleta in request.cookies(galleta)
      cadena = cadena &"    " & subgalleta & ": " & request.cookies(galleta)(subgalleta) & "<br>"
    next
  else
    cadena = cadena & galleta & ": " & request.cookies(galleta) & "<br>"
  end if
next
response.write cadena
%>
```

La manera de trabajar con subclasificaciones es muy parecida a la de trabajar con matrices o arrays de 2 dimensiones. Podemos ver el resultado de este script [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/cookies/cookies3.asp].

Fecha de caducidad programada

Como se dijo en el capítulo de cookies básico, para que las galletitas se guarden en el ordenado una vez acabada la sesión durante un tiempo, hay que ponerles fecha de caducidad. Es muy útil ponerle una fecha de caducidad relativa a la fecha actual, por ejemplo 1 año más tarde, así cuando pase un tiempo, la página guardará las cookies más adelante en el tiempo también. Para ello vamos a utilizar una función de fecha que se llama DateAdd() y funciona conforme a este esquema.

DateAdd(intervalo, numero, fecha)

Intervalo es un string que reprenta las unidades sobre las que vas a añadir a la fecha. Por ejemplo meses se indica con "m", años con "yyyy", días con "d".
Numero se utiliza para indicar cuantos intervalos se añaden, 1 año, o 8 meses por ejemplo.
Fecha es la fecha a partir de la cual se añade el número de intervalos. Se suele utilizar en este parámetro la palabra Now que indica que se tome la fecha actual.

```
<%
response.cookies("id").expires = DateAdd("d",1,Now)
%>
```

En el ejemplo, la cookie caducaría en un día despué.

Detección de navegador

En algunas ocasiones es necesario detectar el navegador del usuario y su versión para mostrar un contenido distinto en cada caso. Por ejemplo podremos mostrar un script en Javascript distinto dependiendo del navegador o un banner distinto. Un detalle importante es que no debemos construir páginas distintas para cada navegador, debemos intentar que las páginas sean compatibles y en algunas ocasiones este método nos puede ser útil.

Para detectarlo disponemos de la **colección ServerVariables** y en concreto la variable **HTTP_USER_AGENT** nos dará la información que queremos. Para empezar podemos empezar por visualizar el contenido de esa variable

```
<% Response.write Request.ServerVariables("HTTP_USER_AGENT") %>
```

Puedes ver el contenido de tu tag HTTP_USER_AGENT aquí [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/detecta_nav/http_user_agent.asp].

Este tag tiene toda la información que puedas necesitar, pero probablemente sea un problema rescatar los datos de ese texto para enterarse de todos los detalles que necesitas. Un ejemplo para tratar la cadena puede ser el siguiente.

Vamos a hacer uso de alguna función de cadena. En concreto vamos a utilizar la función inStr, que sirve para encontrar un substring dentro de otra cadena de caracteres.

InStr([start,]string1, string2[, comparación])

Inicio, es opcional e indica la posición inicial a partir de la cuál buscar
String1, es la cadena donde realizamos la búisqueda
String2, es la cadena que estamos buscando
Comparación, opcional, indica si se realiza una comparación binaria (valor 0, por defecto) o textual (valor 1)

InStr devuelve 0 si no encuentra el string y si lo encuentra devuelve la posición del primer caracter.

```
<%
nav = Request.ServerVariables("HTTP_USER_AGENT")
if instr(nav,"MSIE") then
   response.write "Tu navegador es Internet Explorer"
elseif instr(nav,"Mozilla") then
   response.write "Tu navegador probablemente sea de Netscape"
else
   response.write "Tu navegador no es Netscape ni Explorer"
end if
%>
```

Puedes ver el resultado de este script en tu navegador [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/detecta_nav/http_user_agent2.asp]. Utilizamos la salida de la función InStr como expresión a evaluar en el IF. Si InStr no encontró nada devolverá un cero, que será interpretado como un false. Si encontró la cadena devolverá la posición del primer carácter, que será distinta de cero y será interpretada como true.

No podemos estar seguros de que el navegador sea Netscape porque no indica claramente en su tag HTTP_USER_AGENT la marca del navegador.

Ponemos un caso en el que no sabemos qué navegador es, o más bien, en el que estamos convencidos que no es ni Explorer ni Netscape. Serán pocos los casos en los que no se pueda detectar. En estos caso estamos casi seguros de que el navegador será antiguo y debemos hacer acciones básicas que sean compatibles con los navegadores menos avanzados.

Browser Capabilites

Pero esta no es la única forma de detectar el navegador. Existe un interesante objeto que nos puede devolver los datos que andamos buscando de una manera más ordenada. Se trata del componente Browser Capabilites. Que se usa de la siguiente manera.

```
<html>
<head>
   <title>Browser Capabilities</title>
</head>
<body>

<%
Set nav = Server.CreateObject("MSWC.BrowserType")
%>
Navegador: <%=nav.browser %>
<br><br>
Versión: <%=nav.version%>
```

```
<br><br>

<%
if (nav.vbscript = TRUE) then
%>
  Puedes ejecutar VBScript
<% else %>
  NO puedes ejecutar VBScript
<% end if %>

<br><br>

<% if (nav.javascript = TRUE) then %>
  Puedes ejecutar Javascript
<% else %>
  NO puedes ejecutar Javascript
<%
end if
set bc=nothing
%>

</body>
</html>
```

Podemos ver en marcha este ejemplo en la siguiente página [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/detecta_nav/browsercapabilites.asp], pero probablemente nos demos una desilusión porque en determinados navegadores no se detecta correctamente los valores. Esto es debido a que nuestro servidor web **tiene poco actualizado el archivo browscap.ini**.

La responsabilidad de que el archivo no esté bien actualizado es de nuestro proveedor de hosting, pero si en algún caso queremos actualizar el archivo lo podemos hacer. Para conseguir una versión actualizada podemos visitar la página http://www.cyscape.com/browscap/ donde se puede descargar el archivo.

Si tenemos un proveedor de hosting debería brindarse a actualizar el archivo, si tenemos nuestro propio servidor podremos actualizarlo nosotros mismos sobreescribiendo nuestro browscap.ini actual, que en nuestro sistema está en el directorio c:\windows\system\inetsrv.

Este método es interesante, pero según nuestra experiencia no es suficiente para asegurar que todos los navegadores se detectan correctamente. Una última posibilidad es comprar un componente comercial, como el que venden en la misma página donde podemos descargar browscap.ini, que informan que tiene mayores posibilidades para detectar navegadores, plug-ins, definición de pantalla... El problema es que cuesta bastante dinero. http://www.cyscape.com/.

Trabajo con el buffer de la página

Las páginas ASP se construyen primero en un buffer del servidor y más tarde se mandan al cliente. Esta es la configuración predeterminada de ASP 3.0, aunque no era así en las versiones anteriores. Esto quiere decir que el código HTML que va generando el servidor al procesar la página ASP, se va guardando en una memoria intermedia hasta que se termina de procesar, momento en el que el servidor se encarga de mandar el contenido del buffer.

Nota: parece que será importante saber en que versión de ASP estamos trabajando. Los IIS 4.0 tienen instalado ASP 2.0 por defecto y los IIS 5.0 tienen ASP 3.0.

Nosotros podemos en cierta medida controlar el envío de la página al cliente para que este se realice en los momentos que nosotros deseamos. Para ello tenemos unos cuantos métodos y propiedades del objeto response a nuestra disposición.

Response.Buffer es una propiedad que si está a true (la opción por defecto en ASP 3.0) indica al servidor que debe utilizar el buffer. Si está a false (opción por defecto para versiones

anteriores de ASP) no lo utiliza.
Response.Flush método para ordenar al servidor que mande lo que tenga en el buffer al cliente.
Response.Clear método que limpia el buffer, lo vacía y se pierde su contenido. Solo es útil si se está utilizando el buffer
Response.End método que detiene la ejecución de la página, la termina.

Algunas utilidades

Con la propiedad buffer y los métodos anteriores podremos implementar funcionalidades varias. Por ejemplo, si una página está compuesta por grandes bloques de procesamiento en los que utiliza bastante tiempo podemos, entre bloque y bloque mandar al cliente los datos que se hayan escrito ya.

```
<%
'Código de la página lento de procesar
...
Response.Flush 'envío lo escrito hasta ahora

'Otro trozo de código de la página lento de procesar
...
Response.Flush 'envío lo escrito hasta ahora
...
%>
```

Trabajo con encabezados y texto

También podemos con estas herramientas evitar algún error derivado de escribir en los encabezados del HTTP después de haber escrito en la página texto o HTML. Recordemos las cookies, se dijo cuando se comentaban que viajan en los encabezados del HTTP y que éstos se mandan al cliente antes que el texto con el código de la página. Por esta razón, todo el trabajo con las cookies se ha de realizar antes de escribir ningún texto en la página. Es parecido a lo que pasa con el objeto método redirect del objeto response. En el ejemplo siguiente vamos a ver un ejemplo para ilustrar este asunto.

Se trata de escribir mucho texto en la página y luego tratar de leer una cookie del cliente. Debería dar un error, pero depende de que la página esté o no en el buffer, pues si está en el buffer no se ha mandado y por lo tanto no hemos escrito ningún texto en el cliente. Como hemos dicho que se mande o no se mande depende de la propiedad response.buffer.

Este ejemplo, con response.buffer en true, funciona perfectamente porque, aunque ya hemos escrito cosas en la página, no las hemos mandado a el cliente, porque están en el buffer. Esta es la opción predeterminada en ASP 3.0. Se puede ver el ejemplo en ejecución en esta página [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/buffer/ejemplobuffer1.asp].

```
<%
response.buffer = true
for i=0 to 1000
   response.write "Taller de ASP de desarrolloweb.com "
next
response.clear
response.cookies ("ppepe") = "pqpqpqpqpqpqpqpqpq"
response.write request.cookies ("ppepe")
%>
```

Con response.buffer a false el mismo script va a dar un error porque no se pueden leer o escribir cookies si ya hemos escrito algo en la página y se ha mandado al cliente. False es la opción predeterminada en ASP 2.0 y 1.0. Se puede ver el ejemplo en ejecución en esta página [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/buffer/ejemplobuffer2.asp].

```
<%
response.buffer = false
for i=0 to 1000
   response.write "Taller de ASP de desarrolloweb.com "
```

```
next
response.cookies ("ppepe") = "pqpqpqpqpqpqpqpqpq"
response.write request.cookies ("ppepe")
%>
```

Salirse si el usuario se ha ido

Otro ejemplo interesante para ilustrar lo que se puede hacer controlando los flujos es detenerlos en el caso de que el cliente abandone la página. Imaginar que tenemos un proceso que requiere mucho procesamiento y el cliente se va de la página antes de acabarlo. Para evitar todo ese tiempo perdido en el procesamiento de su solicitud se puede ir comprobando a medida que va pasando el tiempo si el cliente continúa a la espera, pues si no continúa será inútil terminar el procesamiento que estábamos realizando.

Para conseguir saber si el cliente continúa a la espera se puede utilizar una **propiedad del objeto response llamada IsClientConnected** que devuelve true en caso de que el cliente siga a la espera y false en caso contrario. El ejemplo siguiente acabará de aclararlo todo.

```
<%
' Procesamiento muy largo en el tiempo
...
if not Response.IsClientConnected then
   Response.End
end if
...
' Procesamiento muy largo
%>
```

Es interesante destacar que la propiedad Response.IsClientConnected no se puede utilizar con seguridad hasta ASP 3.0. En ASP 2.0 fue implementada, aunque no es aconsejable usarla porque tenía algún problema para dar resultados correctos.

Utilización o no de la Caché

Es habitual que los programadores de páginas dinámicas se las tengan que ver con la caché en algunas ocasiones y tengan que evitar que los navegadores la utilicen para que todo marche correctamente. Existen varias opciones con las que configurar el comportamiento de la caché a través de las cabeceras del HTTP, que son las que vamos a introducir en este capítulo.

Por lo general los navegadores guardan una copia de cada archivo al que acceden en la caché del navegador y cuando consultan otra vez un archivo que ya tienen en la caché local comprueban la fecha de la copia que tienen con la que está en el servidor. Si las dos tienen la misma fecha de modificación el servidor le envía un mensaje informándolo. Si por el contrario, la página que está en el servidor es más nueva, el servidor la envía al cliente. Esta comprobación puede ser alterada gracias a las cabeceras del HTTP, a través de las propiedades Response.Expires y Response.ExpiresAbsolute.

Response.Expires

Sirve para indicar, en minutos, el tiempo que ha de pasar hasta que la página caduque. Durante este tiempo no se solicitará la página al servidor y se utilizará la página que está en la caché.

```
<%
response.expires=2 'caducara a los 2 minutos
%>
```

Si igualamos la propiedad a 0, la página caduca instantáneamente, con lo que se evita la caché

```
<%
response.expires=0 'caduca instantáneamente
%>
```

Response.ExpiresAbsolute

Con esta propiedad se puede expresar el momento exacto en el que caducará la página y a partir del cuál el navegador no debe tomarla de la caché.

```
<%
response.expiresAbsolute=dateAdd("yyyy",1,now) 'caducara dentro de un anyo
%>
```

En este ejemplo se define que la página caduque al año de haberse recibido, puesto que se utiliza la función dateAdd() para añadir un año a la fecha actual. Podríamos haber puesto cualquier otra fecha compuesta por el día y la hora con el formato correspondiente en nuestro sistema.

Ejemplos:

Podemos ver a continuación unos ejemplos online con cada una de estas tres posibilidades [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/cache/ejemplocache.asp]. Los resultados que se obtienen en estos ejemplos pueden variar puntualmente en algún navegador, pero las pruebas realizadas con varios navegadores han dado resultados positivos.

Páginas públicas o privadas

En ocasiones es importante controlar la privacidad de la información que se manda. Supongamos que en nuestro acceso a Internet tenemos un proxy cerca. Los servidores proxy tienen su propia memoria caché y si un usuario accede a una página a la que ya había accedido otro usuario posiblemente el proxy le envíe la copia de la página que tenía en su memoria caché. En algunas ocasiones nos interesará que la página la sirva el proxy y en otras nos interesará que la página la sirva directamente el servidor y esto lo podremos dominar fácilmente con Response.Expires. Pero hay un caso especial en el que puede ser especialmente problemático que un usuario acceda a una página que estaba construida para otro, por ejemplo en el caso de que la información fuese confidencial o extraída para el usuario primero.

Para evitar que páginas con contenidos personales se puedan mandar a otras personas distintas de su dueño por culpa de los servidores proxy está la propiedad Response.CacheControl. Si le asignamos el valor "Private" la página no será guardada por los servidores proxy. Si le asignamos el valor "Public" las páginas si que se almacenarán en los servidores proxy y podrán ser enviadas a otras personas.

```
<%
Response.CacheControl = "Public"
%>

<%
Response.CacheControl = "Private"
%>
```

Otro truquillo

Otra manera de evitar la caché se puede realizar de una manera más artesanal. Se trata de conseguir que la URL a la que accedemos siempre varíe. Como siempre será distinta nunca se buscarán los archivos de la caché.

Conseguir que el URL de la página sea siempre distinto se puede hacer gracias al paso de parámetros por la URL, aunque luego en la página no utilicemos los parámetros para nada, conseguiremos que la URL varíe. Para mandar en un parámetro siempre un número distinto podemos adjuntar una variable que construimos con la hora del sistema.

```
<%
Tiempo_url = year(now) & month(now) & day(now) & hour(now) & minute(now) & second(now)
%>
```

```
<a href="ejemplocachev21.asp?parametro=<%=Tiempo_url%>">Ver</a>
```

Vemos cómo se construye un enlace que tiene un parámetro que cambia cada segundo. Podemos verlo funcionando para que quede más ilustrado [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/cache/ejemplocachev2.asp].

No llegar a guardar la página en la caché

Puede ser útil decirle al navegador que no llegue a guardar la página en la caché mandándole una cabecera especial en el HTTP, sobretodo en casos como el anterior, que estamos guardando muchas páginas en la caché, una por cada vez que se accede con un parámetro distinto en la URL.

```
<%
Response.AddHeader "PRAGMA", "NO-CACHE"
%>
```

Con Response.AddHeader se pueden mandar más tipos de encabezamientos del http como la fecha de modificación del documento (LAST-MODIFIED) o el tiempo en el que se tiene que realizar un refresco de la página [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/cache/refresh.asp] (REFRESH).

Objeto Server

El objeto server sirve para realizar una amplia gama de cosas, ya lo habíamos utilizado en muchas ocasiones, como para conectar con una base de datos o para conectar con el sistema de archivos, pero sus usos son mucho más amplios. En este capítulo vamos a conocerlo con mayor profundidad.

Como su propio nombre indica, el objeto Server se utiliza para realizar acciones relacionadas con el servidor, como instanciar objetos, ejecutar páginas o transferir la ejecución a otras páginas. Vamos a ver las propiedades y métodos del objeto.

Propiedades

ScriptTimeout Sirve para definir el tiempo máximo de ejecución de un script ASP. Limitar el tiempo sirve para evitar que una página que tenga algún problema en su ejecución llegue a bloquear el servidor o algún recurso del sistema, de modo que cuando pasa un tiempo la página para su ejecución y libera los recursos.

Esta variable solo la tendremos que utilizar si el procesamiento de la página tarda mucho tiempo, para que el script no se quede a mitad de su procesamiento por exceder el tiempo límite. El tiempo por defecto es 90.

```
<%
Server.ScriptTimeout = 200
%>
```

Métodos

CreateObject ("Id") Sirve para instanciar objetos en el servidor, como objetos que conectan con bases de datos, con el sistema de archivos, objetos para mandar emails, etc. Este método devuelve siempre una instancia del objeto con ese identificador. Ya hemos visto ejemplos de él en otros capítulos y también los veremos en lo sucesivo.

Execute("URL") Desde ASP 3.0 (IIS 5.0) se puede ejecutar una página web desde otra

página. El control y el estado de la ejecución se transfiere de la primera página a la que se indica como parámetro y cuando se acaba la ejecución de la página se devuelve a la página original.

Transfer("URL") Es lo mismo que la anterior solo que el control de la página no se devuelve a la página original. Solo para ASP 3.0.

GetLastError() Devuelve el último error que ocurrió en la ejecución de la página ASP como una instancia de un ASPError que tiene información sobre el nombre del archivo, la página donde sucedió el error, etc.

HTMLEncode("str") Recibe una cadena a la que le sustituye los caracteres especiales del HTML, por ejemplo < se convierte en < o ¿ se sustituye por ¿

```
<%
cadena = Server.HTMLEncode("Esto es una <b>Tontería</b>, este texto ""es falso"" ¿Vale?")

response.write cadena
%>
```

Esto nos devolvería
Esto es una Tontería, este texto "es falso" ¿Vale?

URLEncode("str") Cuando mandamos información como parámetro por la URL debemos convertir cierto tipo de caracteres que no son válidos, como los espacios u otros símbolos, por sus correspondientes códigos, por ejemplo + se utiliza para el espacio o %23 para el carácter #.

```
<%
cadena = Server.URLEncode("carácter no valido!")
response.write cadena
%>
```

Que nos devolvería
car%E1cter+no+valido%21 y lo podríamos utilizar para construir un enlace de este modo.

```
<a href="http://www.desarrolloweb.com?param= <%=Server.URLEncode("carácter no valido!")%>">
carácter no valido!
</a>
```

MapPath("url") Devuelve la ruta completa en el disco del archivo indicado con el parámetro URL.

```
<%
cadena = server.MapPath("index.asp")
response.write cadena
%>
```

Devolvería algo parecido a esto
C:\inetpub\wwwroot\index.asp

Intranets con ASP

Este artículo se sale un poco de la temática del Taller de ASP [http://www.desarrolloweb.com/manuales/11/], pero creo que también puede dar ideas a los lectores sobre qué se puede hacer con ASP en la práctica en el mundo de la empresa.

Hace unos días, un medico de aquí de la zona donde vivo, me dijo:

***Necesito que me hagas un programa para el hospital**, te animas?*

El reto me sonó interesante y al enterarme de los detalles, vi que era una encuesta que querían cargar a la PC, para luego realizar algunas estadísticas con la información recabada.

Yo le dije que me animaba, y analizando el proyecto me encontré con una encuesta de 170 opciones y campos.

Decidí realizarla e ASP utilizando una base de datos de Access para guardar la información.

La programación se logró en un par de días y todo anduvo muy bien.

Ellos mediante un Servidor NT que tenían podrían cargar los datos a través de formularios y luego utilizar la información de la base de datos para realizar diferentes consultas.

A lo que quiero ir en definitiva a que **ASP también lo podemos ocupar para intranets**, desde un Server NT podemos hacer maravillas, en desarrollo de Programas empresariales, ya que podemos hacer consultas, que nos arroje gráficos, búsquedas, etc.

A modo de ejemplo, demos que en nuestra empresa tenemos una red, y nuestro negocio es la venta de libros.

Entonces nosotros con un poco de conocimiento administramos una Base de Datos creando distintos campos, le incluimos un buscar, algunas consultas que nos listen las últimas entrada, los gastos, entradas y de esta forma organizar la información de nuestro inventario!!!

Realmente ASP aparte de ser un lenguaje para Web se le puede dar utilidades muy grandes para intranets, para el manejo que nosotros queramos darles.

Quizás algunos digan que cuales serán las ventajas de ASP, yo les puedo comentar algunas:

1. **Es liviano** y puede correr en PCs normales que tengan Windows y un servidor web.
2. **Se puede utilizar desde cualquier computadora que esté conectada a la red** que tenga instalado un navegador.
3. **Es muy fácil de programar**, y tiene muchas utilidades que con una breve línea de aprendizaje pueden ser modificadas a su gusto.

Anímense como yo me anime, y diseñen sus "Programas" ASP para intranets, que por supuesto también pueden ser utilizados en Internet.

El punto está en que las nuevas tecnologías para Internet pueden tener muchos usos y ser el reemplazo de programas comunes en una empresa.

En la práctica tu mismo puedes construir una intranet con la ayuda de desarrolloweb.com, simplemente se trata de aprender ASP con los manuales [http://www.desarrolloweb.com/asp] de la página. Además, tal como está relatado en los manuales, necesitarás instalar un servidor web en tu equipo para probar las páginas que vas creando. El servidor web podrá ser Internet Information Server [http://www.microsoft.com/technet/iis/] si trabajas con NT o Personal Web Server [http://www.desarrolloweb.com/directorio/sistemas/personal_web_server/] si tu sistema es Windows 95 o superior.

Cuando estás probando los scripts en local ya tienes una Intranet montada. Si pruebas a llamar a ese servidor desde el navegador de cualquier computador de la red podrás acceder a las páginas que habías programado. Al servidor lo podemos llamar con una URL como http://nombre_de_la_maquina/tu_pagina.html o http://168.255.1.1/tu_pagina.html, siendo el número IP el del servidor al que deseas acceder.

Lectura y escritura de archivos en ASP

En algunas ocasiones es necesario que nuestras aplicaciones realicen acciones de lectura o escritura de ficheros de texto en el servidor.

Por poner un ejemplo, podríamos guardar todos los documentos de los reportajes de nuestro sitio en archivos de texto y desde nuestras páginas ASP podríamos abrir esos archivos de texto y mostrarlos dentro del diseño de nuestro sitio. Esto es una técnica habitual, que utilizamos también en desarrolloweb. Cualquiera que haya programado un poquito conocerá la importancia que tiene el manejo de archivos de texto en programas, de modo que no serán necesarios más ejemplos de usos posibles.

Como estamos en ASP, tenemos que tener en cuenta que **los archivos que podemos manipular se encuentran en el servidor**, ya que ASP puede tener acceso a los recursos del servidor y no a los del cliente.

Para leer o escribir archivos de texto en el servidor utilizando ASP se ha de crear un objeto **File System Object** (FSO), que sirve para tener acceso al sistema de archivos del servidor donde están nuestras páginas. Por ahora no hemos hablado mucho del FSO en desarrolloweb.com, pero tenemos pensado hacer una serie de reportajes para tratarlo a fondo. De momento será suficiente que sepamos que para leer o escribir un fichero debemos apoyarnos en el FSO necesariamente. Para crear una conexión con el FSO en nuestras páginas ASP hacemos lo siguiente.

```
set con_FSO = createObject("scripting.filesystemobject")
```

En este momento ya tenemos acceso al sistema de archivos a través del File System Object, nuestra variable *con_FSO* guarda el objeto que realiza la conexión con el sistema de archivos. Ahora debemos crear el objeto **TextStream**, que será el **objeto final que necesitaremos tener para leer o escribir el fichero**.

Existen en ASP tres métodos para crear el TexStream que se utilizan en casos distintos, según las acciones que pretendamos realizar con el fichero. Los tres métodos devuelven un objeto TextStream.

CreateTextFile (*archivo*,*sobreescribe*,*unicode*)
Este método creará un fichero en nuestro sistema de archivos y devolverá un objeto TextStream, que es el que utilizaremos para movernos por el fichero, leer o escribir. El parámetro *archivo* es para indicar la ruta del sistema de archivos del servidor donde se creará el fichero. Los otros dos parámetros son opcionales. Cuando está a true *sobrescribe*, indica que si había un archivo con ese nombre se sobrescribe. Cuando *unicode* está a true se indica que el archivo se debe crear con juego de caracteres unicode.

OpenTextFile (*archivo*,*tipo_acceso*,*crear*,*formato*)
Con este método abrimos un archivo de texto para leer o escribir y nos devuelve el objeto TextStream necesario para realizar las acciones sobre el fichero. El parámetro *archivo* indica la ruta del fichero a crear. El parámetro *tipo_acceso* es opcional, indica el tipo de acceso que vamos a realizar, por defecto se abre para lectura, ForReading (1), también podemos abrirlo para escritura ForWriting (2) y para añadir ForAppending (8). Con el parámetro *crear*, también opcional, se indica si se debe crear el fichero en caso de que no exista, por defecto no se crea. Por último, *formato* nos sirve para indicar el formato del fichero, ASCII es el predeterminado.

OpenAsTextStream (*tipo_acceso*,*formato*)
Es el tercer método para obtener un TextStream, la diferencia es que se aplica a el objeto File (fichero) que no hemos visto, en lugar del objeto FileSystemObject.

Si queremos crear un objeto TextStream a partir de un FSO necesitamos utilizar uno de los dos primeros métodos, siendo el tercero útil para crear un TexStream a partir de un objeto File.

El objeto TextStream

Como hemos señalado, es el que se utilizamos para movernos por el fichero y realizar las acciones sobre el texto, enfocadas a la lectura o escritura. Tiene las siguientes propiedades.

AtEndOfLine vale true si nos encontramos al final de la línea en el fichero.
AtEndOfStream vale true si estamos al final del fichero.
Colum guarda el valor de la columna del carácter actual en el que estamos situados dentro del fichero.
Line guarda el valor de la línea actual.

Los métodos del objeto son los siguientes.

Close() cierra el fichero. Necesario una vez acabado el trabajo.
Read(*numero*) devuelve un *numero* de caracteres del fichero.
ReadAll() lee todo el fichero y lo devuelve.
ReadLine() lee una línea entera.
Skip(*numero*) pasa un *numero* de caracteres dado.
SkipLine() salta una línea.
Write(*texto*) escribe un *texto* dado dentro del fichero
WriteLine(*texto*) escribe un *texto* y coloca al fina un salto de línea.
WriteBlankLines(*numero*) coloca un *numero* dado de saltos de línea.

Estas notas servirán seguro como buena referencia para realizar cualquier tipo de acción sobre ficheros, pero antes de acabar vamos a ver un ejemplo completo de trabajo con ficheros. Vamos a hacer un pequeño script que crea un fichero y escribe los números del 0 al 9. Luego cerraremos el fichero y lo volveremos a abrir para lectura para sacar su contenido en la página web.

```
<%
'creamos el nombre del archivo
archivo= request.serverVariables("APPL_PHYSICAL_PATH") & "pruebas.txt"

'conectamos con el FSO
set confile = createObject("scripting.filesystemobject")

'creamos el objeto TextStream
set fich = confile.CreateTextFile(archivo)

'escribimos los números del 0 al 9
for i=0 to 9
   fich.write(i)
next

'cerramos el fichero
fich.close()

'volvemos a abrir el fichero para lectura
set fich = confile.OpenTextFile(archivo)

'leemos el contenido del fichero
texto_fichero = fich.readAll()

'imprimimos en la página el contenido del fichero
response.write(texto_fichero)

'cerramos el fichero
fich.close()
%>
```

En la primera línea del código creamos el nombre del archivo. El nombre del archivo es una variable cadena que contiene la ruta absoluta del archivo en el sistema del servidor. Como probablemente ocurra en vuestro alojamiento ASP, vosotros no sabéis en qué directorio y disco del servidor están alojadas vuestras páginas (a no ser que seáis vosotros los administradores o estéis trabajando en local). Si es así, necesitaréis algún mecanismo para obtener la ruta donde están vuestros archivos y donde tenéis permisos de lectura y escritura. Con el método **request.serverVariables("APPL_PHYSICAL_PATH")** obtenemos la ruta física donde están nuestras páginas. A este camino le concatenamos el nombre del archivo que deseamos crear y ya tenemos el la ruta completa del archivo.

El resto de líneas se pueden entender bien con los comentarios y las notas de arriba en este artículo.

Es importante señalar que para leer y escribir en vuestros directorios necesitaréis los correspondientes permisos de lectura y escritura. Si estáis haciendo pruebas en un sistema Windows95 no necesitaréis permisos, ya que ese sistema no los implementa. En servidores un poco más serios como los que tendrá vuestro proveedor de alojamiento necesitaréis que os otorguen permisos a los directorios donde vayáis a escribir ficheros. Los permisos de lectura suelen estar otorgados desde el principio.

Eso es todo por ahora, esperemos que este artículo sea bien útil para resolver vuestras dudas con el tratamiento de ficheros de texto en ASP.

Cálculo de días que faltan para una fecha

Vamos a construir un script que calcule los días que faltan para llegar a una fecha cualquiera. Para que la página web pueda informar al usuario diciendo algo como "Faltan 28 días para mi cumpleaños".

Este script lo podríamos utilizar en una página web de un evento, feria o cualquier otro suceso. Resulta anticuado anticuado dar este ejemplo, pero podríamos haber hecho una página que dijese "Faltan 280 días para el año 2000"

Como estamos trabajando con scripts del servidor, la fecha sobre la cual se calculará los días que falta será la fecha del sistema servidor. Esto tiene como inconveniente que la hora del servidor puede ser ligeramente distinta que la del ordenador del cliente, si los dos sistemas se encuentran en países con franjas horarias distintas. Podríamos hacer un script parecido en Javascript de cliente y tomaría la fecha del cliente, que puede ser útil en algunos casos. Simplemente lo señalamos aquí porque es interesante que lo sepa el lector.

Averiguamos las fechas

Para realizar nuestro cálculo empezamos por averiguar la fecha actual del servidor y la fecha del evento futuro.

```
fecha_actual = Now
fecha_futura = CDate("1/1/2025")
```

En estas dos líneas de código obtenemos primero la hora actual asignando a la variable fecha_actual el valor de la variable de sistema Now, que contiene la fecha del servidor.

Posteriormente obtenemos la fecha futura a partir de una cadena de caracteres, convirtiéndola a un objeto Date (fecha) con la función CDate(). En el ejemplo obtenemos la fecha correspondiente a el 1 de enero de 2025, podría ser cualquier otra.

Función DateDiff

Existe una función de Visual Basic Script muy útil para el ejercicio. Es la función DateDiff, que calcula la diferencia entre dos fechas y puede hacer el cálculo en días, horas, minutos, segundos, etc.

La sintaxis es la siguiente:

DateDiff(intervalo, fecha1, fecha2)

Donde intervalo es la unidad en la que deseamos hacer el cálculo "s" para segundos, "d" para días, "h" para las horas, "m" para meses, "yyyy" para años...

Los parámetros fecha1 y fecha2 son las dos fechas involucradas en la resta.

Podemos acceder a una descripción más detallada de esta función en la Librería MSDN

[http://msdn.microsoft.com/library/default.asp?url=/library/en-us/script56/html/vsfctdatediff.asp] de Microsoft.

Calculamos los días

La utilización de la función DateDiff con las fechas que habíamos obtenido previamente es muy simple.

dias_restantes = DateDiff ("d", fecha_actual, fecha_futura)

Con esto ya hemos obtenido el valor buscado y podríamos imprimirlo en la página directamente.

Todo Junto

Finalmente vamos a ver todo el ejercicio de una vez en una página ASP.

```
<html>
<head>
<title>Calculo de los días que faltan para una fecha</title>
</head>
<body>

<%
fecha_actual = Now
fecha_futura = CDate("1/1/2025")

dias_restantes = DateDiff ("d", fecha_actual, fecha_futura)
%>

Faltan <%=dias_restantes%> días para el año 2025

</body>
</html>
```

Podemos ver este script en funcionamiento [http://www.guiarte.com/desarrolloweb/tallerasp/calculodias.asp].

Hasta aquí llega este taller de ASP que también nos puede servir para aprender a calcular otro tipo de diferencias, por ejemplo los segundos que faltan para una fecha, los meses, o cualquier otra cosa. Sólo habría que utilizar la función DateDiff pasándole otro intervalo por parámetro.

Componentes de servidor ASP

Vamos a intentar dar unas nociones muy básicas sobre lo que son los componentes de servidor de ASP, en el marco de el Taller de ASP y pensando en futuros artículos en los que explicaremos el funcionamiento de algún que otro componente interesante para los servidores IIS (en los que se ejecutan las páginas ASP de Microsoft)

Qué son los componentes de servidor

Para hablar de manera sencilla, los componentes del servidor son programas que sirven para realizar acciones más o menos complejas en el nuestras páginas ASP. Éstas acciones pueden ser por ejemplo, el envío de correo electrónico, realizar upload de ficheros al servidor, conectar con una base de datos, etc. Los que conozcan ASP sabrán que el lenguaje con el que se escribe (VBScript o Jscript) nos permite unas funcionalidades que no van más allá de las básicas de cualquier lenguaje: trabajo con variables, tipos, estructuras de control y un juego de funciones (que en el caso de VBScript es bastante limitado).

Así que, si en una página ASP estamos pensando en hacer algo un poco complejo, lo más seguro es que lo tengamos que realizar a través de algún componente del servidor. Como decíamos, incluso las conexiones y accesos a bases de datos que muchos de vosotros realizaréis habitualmente se hacen a través de un componente del servidor.

Active X

Para hablar sobre los componentes de servidor es necesario hablar también de la tecnología ActiveX de Microsoft. Ésta se trata de un conjunto de tecnologías independientes del lenguaje de programación orientadas a posibilitar que trabajen entre si los distintos componentes en entornos de red.

Los componentes ActiveX no son otra cosa que los componentes de servidor que estamos comentando. Por otra parte, están los controles ActiveX (controles esta vez, no componentes) que son pequeños programas que se insertan en las páginas web a través de las etiquetas <OBJECT> y <PARAM>. Los controles se ponen en marcha en el cliente, cuando se ejecutan lo hacen dentro de la página web. Un ejemplo típico es la invocación de una animación de Flash o Shockwave. El motor de Flash o Shockwave, es un control ActiveX. Por otra parte y como decíamos, los componentes ActiveX se ponen en marcha en el servidor.

Los componentes ActiveX, por tanto, son los que nos interesan a nosotros en este artículo, pues son los que se invocan desde ASP y se ejecutan en el servidor al mismo tiempo que la página, antes de enviarla al cliente. Además, nos vamos a centrar en el uso de componentes y no en su programación, que es un tema demasiado amplio.

A título de comentario, se puede indicar que para crear componentes de servidor se puede utilizar cualquier lenguaje de programación. Aunque muy habitualmente se hacen en Visual Basic, se pueden hacer también en Delphi, Visual C++ o el propio C++ por ejemplo. Para su programación es necesario que se sigan unas normas y estructuras.

Cómo se trabaja con los componentes

Los componentes son objetos que, como objetos que son, tienen propiedades y métodos. Las propiedades son las características del objeto y los métodos son sus funcionalidades. Para trabajar con un componente primero debemos instanciarlo (crearlo e inicializarlo). Una vez creado, habitualmente, lo configuraremos accediendo a sus propiedades y actualizando sus valores. Finalmente llamaremos a sus métodos para poner en marcha sus funcionalidades.

La instanciación de un componente de servidor se hace a través del objeto server de ASP. Es el server.createobject que muchos de vosotros habréis visto en más de una ocasión en códigos ASP.

Set mi_componente = Server.CreateObject(IDENTIFICADOR)

El identificador que le pasamos es una cadena de caracteres que contiene el código del componente que se quiere crear. En el caso de una conexión con una base de datos, el identificador es "ADODB.Connection". Cada componente de servidor tiene su propio identificador, definido por el programador del componente.

En el manual de ASP I se encuentra una explicación detallada del componente de acceso a datos (ADO), utilizado para acceder a las bases de datos.

En cualquier caso, para aprender a manejar un componente tendréis unas instrucciones precisas en la documentación que acompaña a cada componente. Es necesario leer la documentación porque cada componente tiene sus propias propiedades y métodos.

Componentes de interés

Ya hemos señalado algunos ejemplos de componentes útiles, como el envío de correo electrónico por el servidor o subir archivos al servidor, pero podemos ver muchos más:

- Acceso al sistema de archivos del servidor

- Creación de imágenes en el servidor
- DNS lookup
- Ejecución de programas o comandos en el servidor

Muchos de ellos son comerciales y habrá que adquirirlos a cambio de una cantidad de dinero. Es uno de los problemas de ASP, que todo cuesta bastante dinero, cuando en otros lenguajes como PHP lo puedes encontrar "de casa" y/o gratis.

Ejemplos típicos de lugar donde se pueden adquirir componentes variados es Serverobjects.com [http://www.serverobjects.com/] o Persits.com [http://persits.com/], páginas de empresas que se dedica a programarlos y venderlos. También podemos encontrar un directorio de componentes y controles ActiveX en ActiveX.com [http://www.activex.com/].

Instalar componentes en nuestro servidor

Algunos de los componentes que necesitamos en la programación de páginas ASP están ya instalados por defecto en los servidores web, es el caso de el componente de conexión con la base de datos o el de conexión con el sistema de archivos del servidor (File System Object). Sin embargo, otros componentes si que necesitaremos instalarlos en la máquina que vaya a utilizarlos.

Un componente suele ser un archivo .dll, -librería de Windows- y para instalarla en nuestro sistema deberemos seguir sus instrucciones de instalación. Tanto las instrucciones de instalación como las de manejo del componente deberían acompañar a la dll entre los archivos de descarga del componente.

Es habitual que la instalación de esa dll se realice manualmente. Para ello copiaremos el archivo .dll en nuestro directorio system (\winnt\system32 en NT o \windows\system en Win95) y luego registraremos la dll en nuestro sistema con el comando **regsvr32 mi_componente.dll**, que debemos ejecutar desde línea de comandos (C:\>).

En algunos casos, el componente se instala en Windows igual que cualquier otra aplicación. Como decíamos, cada componente puede instalarse de manera diferente.

Si tenemos alojada la página en un servidor que no es nuestro, en un servidor de hosting, es importante que preguntemos al soporte técnico de ese proveedor la manera de instalar los componentes, pues generalmente habrá un procedimiento definido para instalarlos o bien ni siquiera estará permitido. Es importante que preguntéis antes de contratar un servidor de alojamiento sobre este punto, no sea que una vez pagado os digan que no permiten componentes propios y vosotros necesitéis uno para construir la página.

AspUpload

Muchos son los componentes ASP disponibles en el mercado, gratuitos o no, encargados de gestionar la transferencia de archivos por medio de un navegador hacia el servidor.

> **Referencia: Componentes de servidor en ASP [http://www.desarrolloweb.com/articulos/657.php].**
> Podéis encontrar una descripción de los componentes de ASP [http://www.desarrolloweb.com/articulos/657.php] muy instructiva en DesarrolloWeb.com. Interesante de leer para todo aquel al que sea una novedad hablar de los componentes ASP en el servidor.

Este tipo de componentes añaden otro elemento de interacción con el usuario, quien sería capaz de enviar cualquier tipo de archivo a nuestro servidor. Este archivo podría a su vez ser abierto y mostrado al resto de los usuarios en tiempo real.

Una aplicación ejemplo donde podríamos usar este tipo de objetos sería un sitio web de chat

donde, cada persona que entra, aporta si lo desea una foto suya que será visualizada por el resto de los interlocutores.

Gestionar esta o cualquier otra aplicación del estilo implica que haremos frente a toda una serie de situaciones que hay que solventar de la manera más sencilla:

- No permitir la transferencia de archivos por encima de un determinado tamaño.
- Controlar el tipo de extensión de los archivos que hay que colgar
- Asegurarnos que los archivos puedan o no ser reescritos
- Guardar parámetros como la anchura y altura de una imagen
- ...

Como hemos dicho, existen multitud de componentes que pueden ayudarnos en esta tarea [http://www.aspin.com/home/components/file/upload]. Nosotros comentaremos aquí el que hemos utilizado en alguna ocasión: AspUpload de Persits Software [http://www.persits.com/].

AspUpload permite al servidor aceptar, guardar y manipular archivos que hayan sido enviados por un usuario a partir de un clásico formulario en HTML. El contenido de este formulario es enviado a un script que invoca a un objeto que es quien realmente se encarga de realizar todas las funciones necesarias para la toma y almacenamiento de los archivos.

Entre otras cosas, este objeto nos permite:

- Limitar el tamaño del archivo a colgar.
- Permitir o no el sobrescribir un archivo.
- Controlar los atributos del archivo.
- Mover, copiar, renombrar y borrar el archivo recibido.
- Almacenar los archivos en bases de datos.
- Guardar los archivos en memoria, no en el disco duro.
- Nos informa sobre las dimensiones y tipo de archivo grafico colgado.

A diferencia de otros componentes análogos, AspUpload no es freeware. La razón de que hayamos expuesto éste y no algún otro es simplemente, como ya hemos dicho, porque se trata del que ya hemos utilizado en alguna ocasión. Puede ser no obstante, que el servidor que utilices ya lo haya adquirido y que su utilización no te cueste nada. Una pequeña consulta a tu proveedor puede resolver esta duda. Si éste no es tu caso y deseas hacerte con un componente parecido pero gratuito no tienes mas que buscar en ASP In [http://www.aspin.com/home/components/file/upload].

En cualquier caso, este componente nos ha parecido bastante profesional y de un manejo fácil y completo. Altamente recomendable para este tipo de tareas.

Además, el componente viene documentado de manera excelente en la página de la empresa que lo construye, con ejemplos rápidos y prácticos para ponerlo en marcha en un momento.

Ejemplo rápido

Para hacernos una idea básica sobre cómo se utiliza este componente vamos a realizar un ejemplo muy rápido y sencillo.

A continuación podemos ver el código que podríamos utilizar para colocar un formulario en una página web con el que podríamos seleccionar algún archivo para subir al servidor.

```
<HTML>
<BODY BGCOLOR="#FFFFFF">

<FORM METHOD="POST" ENCTYPE="multipart/form-data" ACTION="cargar.asp">
<INPUT TYPE=FILE SIZE=60 NAME="FILE1"><BR>
<INPUT TYPE=FILE SIZE=60 NAME="FILE2"><BR>
<INPUT TYPE=FILE SIZE=60 NAME="FILE3"><BR>
<INPUT TYPE=SUBMIT VALUE="Upload!">
</FORM>
```

```
</BODY>
</HTML>
```

Es importante colocar el atributo ENCTYPE="multipart/form-data" , que indica que se debe enviar el contenido del archivo y no exclusivamente el nombre.

También podemos ver cómo sería el script que toma el archivo y lo guarda en el servidor. El archivo cargar.asp:

```
<HTML>
<BODY>

<%
Set Upload = Server.CreateObject("Persits.Upload.1")
Count= Upload.Save("d:\xvrt\midominio.com\html\upload")
%>
<% = Count %> ficheros subidos.

</BODY>
</HTML>
```

Lo único raro que hacemos es crear el componente en el servidor y asciarlo a la referencia "Upload", esto lo hacemos con *Server.CreateObject*. Posteriormente utilizamos el método *save* del objeto para guardar el archivo en el servidor.

Recordamos nuevamente que en la página del componente [http://www.aspupload.com/] podemos obtener la documentación entera, que está muy clara e interesante.

> **Nota: PHP ya tiene la opción de upload de salida.**
> Una de las ventajas de PHP es que este tipo de acciones se pueden realizar sin tener que instalar ningún componente especial, ya que el sistema de upload ya viene implementado en el propio lenguaje.

Taller de paso de variables por URL

Este taller resultará un poco básico para algunas personas, ya que no vamos a explicar nada exclusivamente nuevo y que no hayamos comentado ya en otros manuales. Sin embargo, seguro que muchas otras personas agradecerán esta explicación para contestarse algunas dudas a la hora de hacer páginas avanzadas con ASP o PHP.

Resulta que me han llegado algunas duda a mi correo de personas que no comprendían muy bien la manera de realizar páginas que, dependiendo de la variable que reciban por la URL presenten una información u otra. A continuación podemos ver estas dudas para aclararnos.

Duda 1: (Sobre el lenguaje ASP, aunque da igual, ya que la metodología de uno y otro lenguaje son las mismas. Sólo difiere la sintaxis.)

Hola me llamo Orel y necesito si me pueden dar una ayudita con mi pagina web.
Bueno primero que nada estoy creando mi paina en asp y no entiendo cómo poner un enlace por ejemplo:
Cuando colocan varios enlaces y todos llevan a la misma pagina ver.asp lo que cambia es ? Id=emuladores o ?Id=Roms etc.
No se cómo hacer eso.
Por que si yo pongo la pagina ver.asp?Id=emuladores Me aparece la pagina de emuladores mientras que si pongo ver.asp solo me muestra la pagina de inicio ¿Como hacen eso de que con la misma pagina puedan crear varias?

Duda 2: (Esta duda es de PHP, pero da igual el lenguaje para el que estemos trabajando, en el

fondo es la misma duda)

Como se que vosotros sabéis mucho de php, os quería comentar una duda que tengo. ¿Que necesito para crear una web de carátulas en la cual las carátulas salgan en una página que sea una especie de "plantilla", en la cual solo cambie la carátula? Éste tipo de sistemas los he visto en webs de carátulas y son de la forma:
http://nombredelaweb/caratula.php?id=587
http://nombredelaweb/caratula.php?id=351
También cambia el texto y otras cosas.

Respuestas

Pues bien, planteadas las dudas, primero decir que en DesarrolloWeb tenemos todos los contenidos necesarios para aprender técnicamente a utilizar estas variables que se pasan por la URL. Es importante que conozcamos la sintaxis para poder hacer los ejercicios donde dudamos.

Aquí tenemos los enlaces:

- Para aprender a pasar y recoger variables por la URL en ASP [http://www.desarrolloweb.com/articulos/252.php?manual=8]
- Lo mismo pero en PHP [http://www.desarrolloweb.com/articulos/317.php?manual=12].

Una vez aprendida la parte técnica podemos pensar en cómo hacer estos sistemas en ambos lenguajes. Lo primero que debemos saber es que esas variables que recibimos por parámetro las podemos utilizar para cualquier cosa. En la duda 1 las vamos a utilizar para mostrar una información de una sección y en la duda 2 las vamos a utilizar para extraer la información de la carátula que se desea ver. Creo que lo mejor sería ver cada uno de los dos casos, en el lenguaje que se preguntan.

Respuesta de la Duda 1, para ASP

Necesitamos extraer el valor de la variable que nos pasan por parámetro y hacer unas cosas u otras en función de dicho valor. Veamos a grandes rasgos el código que utilizaría.

```
<%
'extraigo el valor de la sección que se desea ver
seccion_ver = request.QueryString("id")

'Dependiendo de la sección muestro unos u otros contenidos
if seccion_ver = "emuladores" then
'entonces coloco todo lo que queramos mostrar en la sección emuladores
%>
<h1>Sección de emuladores</h1>
<p>Bienvenidos a mi sección de emuladores, donde puedes conocer todos los principales emuladores del mercado y descargarte los programas.
<p>... (En fin, todo lo que queramos escribir en esta sección)
<%
elseif seccion_ver = "Roms" then
'entonces coloco todo lo que queramos en la sección de Roms
%>
<h1>Sección de Roms</h1>
<p>Bienvenidos a mi sección de Roms, tenemos más de 500 roms de juegos de todos los tiempos. Cada uno con enlace para descarga e instrucciones de uso.
<p>... (En fin, todo lo que queramos escribir en esta sección)
<%
else
'paso por aquí si no era la sección emuladores ni Roms
'entonces hago lo que desee, que en la duda era simplemente mostrar la página principal.
```

```
Response.redirect("index.asp")
'esta instrucción te redirecciona a la página principal de tu sitio web, si es que se llamaba
index.asp y está en el mismo directorio que la página actual.
end if
%>
```

Como se puede comprobar, dependiendo de lo que se reciba por parámetro se mostrará una sección u otra. Si no se recibe nada por parámetro o el valor no corresponde a ninguna de las secciones que hemos definido, se redirecciona a la página índex.asp, que suponemos que es la página principal del sitio web.

Respuesta de la Duda 2, para PHP

Este ejemplo es, si cabe, más simple que el primero. La forma de trabajar típica que se realiza con un ejemplo como este es que todas las carátulas estén en una tabla de una base de datos. Entonces, cada vez que se accede a la página caratula.php se recibe el identificador de la carátula que se quiere visualizar en una variable en la URL, de esta manera: caratula.php?id=12. Así estamos accediendo la página que muestra carátulas y le decimos que la carátula que deseamos ver es la que tiene el identificador (id) igual a 12.

Supongamos que metemos la información de la carátula en una tabla que tiene esta forma:

Tabla caratula

id	Identificador de la carátula
titulo	Título del disco
imagen	Nombre del archivo imagen de la carátula
info	Descripción complementaria de la carátula

El código en PHP para hacer algo así sería el siguiente. En este código vamos a obviar algunas instrucciones de conexión con la base de datos que realmente no vienen al caso. Recordamos que en el Manual de PHP I [http://www.desarrolloweb.com/manuales/12/] tenemos todas las explicaciones para aprender a manejar bases de datos.

```
<?
//Extraigo el identificador de la carátula a mostrar
$id_caratula = $HTTP_GET_VARS["id"];

//creo la sentencia SQL que extrae datos de esa carátula
$ssql = "select * from caratula where id=" . $id_caratula;

//obtengo los datos de la carátula
$rs = mysql_query($ssql);
$fila = mysql_fetch_object($rs);

//muestro los datos de esa carátula
?>
Disco: <?echo $fila->titulo;?>
<br>
Imagen: <img src="<?echo $fila->imagen;?>">
<br>
Descripción: <?echo $fila->info;?>

<?
//cierro la base de datos
mysql_free_result($rs);
//… todo lo que haga falta para terminar la página...
```

```
?>
```

Conclusión

Con esto acabamos. Esperamos que os sirva de ayuda este artículo o, por lo menos, haya resuelto vuestras dudas sobre la utilización de variables en la URL tanto en ASP como en PHP. Sólo pedir disculpas porque los ejemplos que he utilizado no son páginas completas al 100%, ya que faltan las cabeceras de las páginas y otras instrucciones que no venían al caso. Además, como son ejemplos parciales, se me habría podido escapar alguna falta de sintaxis, al no poderse poner en marcha los scripts así, de manera parcial.

Lo importante es que hayáis cogido la idea y entendido mejor para qué puede servir pasar variables en la URL y cómo hacen los desarrolladores para que la página sea distinta dependiendo de los valores de las variables. Espero que haya sido así.

Recoger campos de formulario complejos en ASP

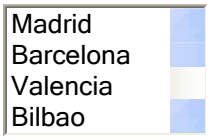
No debería significar un problema recoger los campos de formulario corrientes, como campos de texto, botones de radio, textareas, etc, donde sólo se envía un dato vinculado a dicho campo campo. Se hace con la método form del objeto request, indicando entre paréntesis el nombre del campo que queremos recoger.

```
mi_variable = request.from("nombre_campo")
```

Para el que esto le resulte nuevo, podemos encontrar en el Manual de ASP [http://www.desarrolloweb.com/manuales/8] un capítulo donde se detalla el proceso de recoger datos del formulario [http://www.desarrolloweb.com/articulos/253.php?manual=8]. Nosotros en este artículo vamos a tratar de explicar un método para recoger datos un poco más complejos, donde nos pueden enviar varios valores dentro de un mismo campo de formulario. En concreto vamos a tratar de extraer los datos de un campo de formulario "Select múltiple" que se consigue con las siguientes etiquetas HTML:

```
<select name="equipo" multiple>
<option value="Madrid">Madrid
<option value="Barcelona">Barcelona
<option value="Valencia">Valencia
<option value="Bilbao">Bilbao
<option value="Sevilla">Sevilla
<option value="ATMadrid">At. Madrid
<option value="Cadiz">Cadiz
<option value="Dep. Coruña">Dep. Coruña
<option value="Santander">Santander
</select>
```

Hemos colocado un select normal que tiene el atributo múltiple, que vemos que no se iguala a nada, simplemente se coloca si queremos que el usuario pueda seleccionar varias opciones. Para ello debería seleccionar una de las opciones y con el la tecla de Control (Ctrl) o mayúsculas (La flechita hacia arriba) seleccionar otra opción. Si se utiliza la tecla de Control se seleccionan las dos opciones, la que había y la nueva. I se utiliza mayúsculas se seleccionarían todas las opciones entre la primera y la última. Podemos practicar a seleccionar varias opciones, para el que lo desee, con el campo de abajo.



Cómo llegan los datos

Cuando recibimos este campo de formulario nos llegan todos los valores que un usuario haya seleccionado, separados por comas. Así, un valor posible que se puede recibir por el formulario sería el siguiente:

Madrid, Barcelona, Valencia, ATMadrid, Dep. Coruña

Este valor lo obtenemos utilizando el objeto request de ASP, tal como comentábamos antes. Para acceder entonces a ese dato escibiríamos.

request.form("equipo")

El método reques.form en este caso devuelve, como indicábamos, todos los equipos seleccionados separados por comas. De manera adicional, podemos tratar el valor devuelto como una colección, que es una estructura de datos especial, parecida a los arrays, que se recorren fácilmente con un bucle FOR EACH.

> **Referencia:** Tenemos un taller de ASP en el que hacemos un par de ejemplos del bucle FOR EACH para recorrer arrays y colecciones [http://www.desarrolloweb.com/articulos/297.php?manual=11].

En algún caso puede interesarnos volcar la información de esa colección a un array para tratar luego los equipos en otros procesos. Como práctica también puede ser útil ver como se haría y de paso, conocemos un poco mejor el bucle FOR EACH.

Construir el array

Nuestro ejercicio requiere un array que tenga un número indeterminado de casillas, que depende del número de valores que recibamos desde el campo <select> del formulario. Si queremos hacer esto bien vamos a necesitar un array dinámico, al que iremos asignando posiciones según lo necesitemos. En el Manual de Visual Basic Script [http://www.desarrolloweb.com/manuales/1] podemos encontrar la explicación sobre cómo tratar arrays dinámicos [ttp://www.desarrolloweb.com/articulos/176.php?manual=1].

```
dim equipos()
redim equipos(0)
```

Así creamos el array y le indicamos que vamos a utilizar como índice máximo el cero, que corresponde con una casilla, equipos(0).

Script para meter los datos en un array

Vamos a hacer un script que se encargue de rellenar el array con todos los valores del campo. Dicho script recorre la lista de valores que llegan del formulario y los introduce en el array, a la vez que va aumentando en una casilla el espacio del array antes de añadir un nuevo elemento.

```
Dim equipos(), I
I = 0
For Each Valor In Request.Form("equipo")
   Redim Preserve equipos(I)
   equipos(I) = Valor
   I = I + 1
Next
```

Empezamos el script declarando el array (sin definir sus casillas, para poder redimensionarlo dinámicamente) y una varible para llevar la cuenta de los valores introducidos, que inicializamos a cero en la siguiente línea.

El bucle FOR EACH recorre cada una de las posiciones de la colección Request.Form("equipo") y en cada iteración introduce en una variable, en este caso llamada Valor, el contenido de la posición actual. Una vez dentro del bucle se redimensiona el array preservando su contenido previo (redim preserve) para que contenga las posiciones necesarias para almacenar los valores

que vamos extrayendo de la colección. Dicho de otro modo, en cada iteración creamos una nueva posición (guardando todas las posiciones anteriores) e introducimos en la posición creada el contenido de la variable Valor, que guardaba la posición actual de la colección. Por último incrementamos en uno el número de posiciones que debe contener el array, para utilizarlo si volvemos a pasar por el bucle.

> **Nota:** Podíamos haber creado un array utilizando la función split, incorporada en VBScript. Dicha función recibe una cadena y un separador y devuelve un array donde en cada casilla se ha introducido una sub-cadena creada al romper por cada separador.
>
> De modo que, a partir del request.form("equipo") (que es también una cadena donde cada valor aparece separado por una coma), aplicando la función Split e indicando que el separador es el caracter coma (,), obtendremos el array deseado.
>
> equipos = Split(request.form("equipo"),",")
>
> Este método es muy rápido y simple, pero podría fallar si uno de los valres del select contiene una coma, tal como apuntan algunos colaboradores con sus comentarios al artículo.

Espero que con lo dicho hasta ahora hayamos podido encontrar sentido al ejercicio y resulte válido como práctica de VBScript y la tecnología ASP. Podemos ver el ejemplo en marcha para examinar su funcionamiento [http://www.guiartemultimedia.com/desarrolloweb/tallerasp/formulario.htm].

Además, podemos descargar en un zip los fuentes que hemos utilizado con ASP para este ejercicio [http://www.desarrolloweb.com/descargas/descargar.php?descarga=6364], por si los queremos tener y probar en nuestro computador.

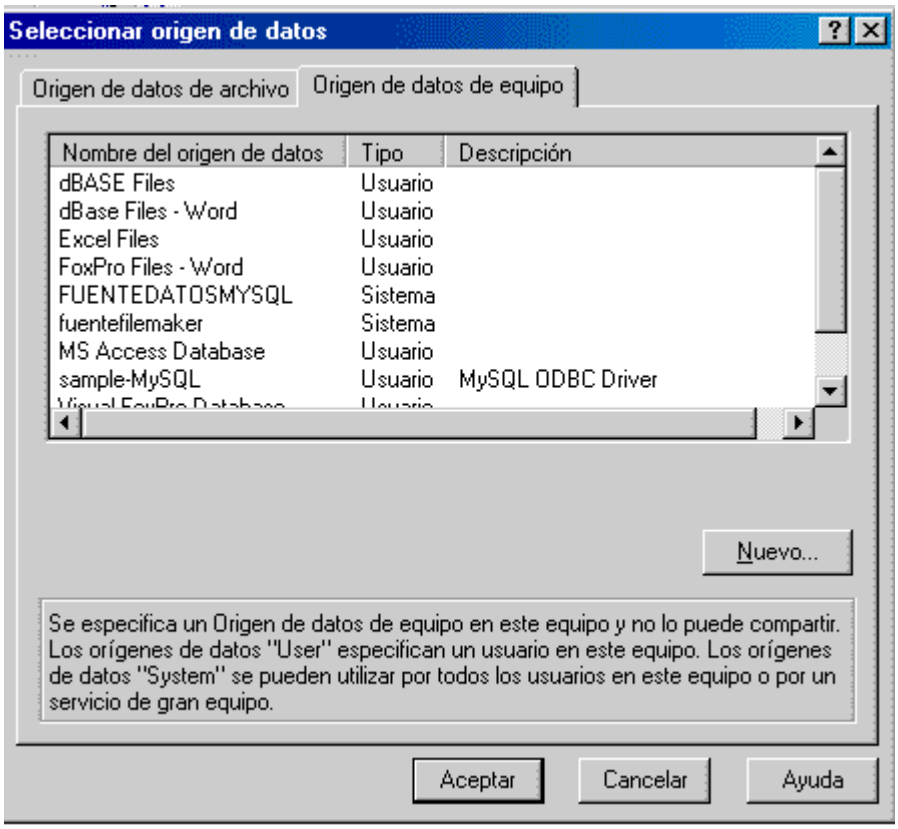
> **Créditos y agradecimientos**
>
> Este reportaje ha sido mejorado gracias a los comentarios, investigaciones y correcciones de las siguientes personas:
>
> **Sergio Flores: flaco@lamatufia.com.ar [mailto:flaco@lamatufia.com.ar]**. Agradecimientos especiales a este compañero, que indico el script para recorrer la colección
>
> **Antonio Guerrero agnotario@eresmas.com [mailto:agnotario@eresmas.com]**. Nos apuntó la solución con la función split.
>
> **César Nieto http://www.desarrolloweb.com/articulos/cesarnieto@terra.es**. También nos recomendó utilizar split.
>
> **Aston aston@maestrosdelweb.com [mailto:aston@maestrosdelweb.com]**. Que nos propuso la solución con Split y también un recorrido de colecciones.
>
> Al pie del artículo se conservan todos los comentarios enviados por los anteriores compañeros, y otros que podrán ir incorporándose. Muchas gracias a todos.

Exportar datos de MySQL a Microsoft Access 2000

Migrar datos de una base de datos a otra es algo a lo que muchos de nosotros hemos tenido que confrontarnos en algún momento. A continuación os explicamos cómo recuperar información almacenada en un servidor de datos Mysql hacia una base Access 2000.

> **Referencia:** Para realizar esta tarea es necesario que hayamos descargado el driver ODBC y lo hayamos instalado en nuestro sistema Windows. Esta labor se puede conocer en un artículo de DesarrolloWeb.com: Instalar el driver ODBC para MySQL [http://www.desarrolloweb.com/articulos/897.php].

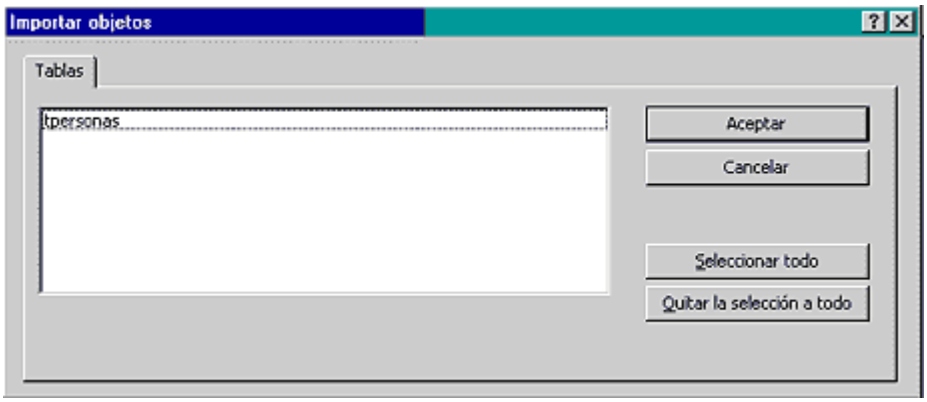
Para importar una tabla de Mysql a Microsoft Access, desde Access, y con la base de datos en la que se quieren importar los datos abierta, seleccionar el menu Archivo->Obtener datos Externos->Importar. En la pantalla de Importar datos, en la opcion Tipo de archivo seleccionar ODBC databases().



Seleccionar origen de datos de equipo, y dentro de esta, el nombre de la fuente de datos que hemos creado anteriormente. Una vez la has seleccionado, y has hecho clic sobre "Aceptar", aparecerá la pantalla de configuración del driver por si deseas marcar para esta acción en concreto, algunas de las opciones de configuración que aparecen en el driver ODBC, si no deseas marcar ninguna, clic sobre "OK".

> **Nota:** pudiera ser en algún caso que los tipos de los datos de la base en los sistemas MySQL y Access no sean totalmente compatibles y se produzca alguna anomalia al exportarlos. Realmente es una posibilidad que pensamos, aunque en las pruebas que hemos realizado no hemos visto ningún tipo de problema, bien es cierto que los campos que hemos trabajado no eran muy raros.

Aparecerá una ventana donde pregunta qué tabla de Mysql se desea exportar a Access:



Selecciona la tabla , y haz clic sobre "Aceptar"

> **Nota:** si estamos exportando los datos hacia o desde un servidor de bases de datos alojado en algún proveedor de Hosting, tenemos que tener en cuenta que estos no siempre incluyen en su paquete básico el acceso remoto al servidor de base de datos, o requiere de un aviso explicito por parte del cliente para su configuración.

Referencia: si deseamos realizar una migración de datos en el otro sentido, es decir, desde Access hacia MySQL, será muy indicado leer otro artículo en DesarrolloWeb que explica el proceso detalladamente. Exportar datos de Access 2000 a MySQL [http://www.desarrolloweb.com/articulos/867.php].

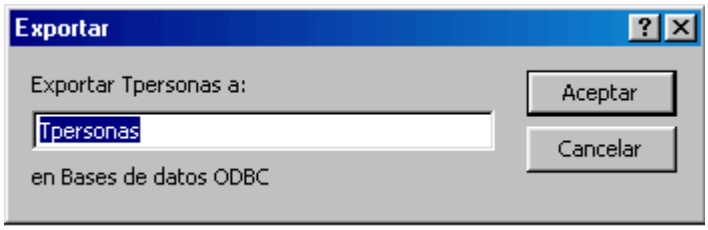
Exportar datos de Access 2000 a MySQL

No es de extrañar que hayamos comenzado a hacer nuestros pinitos en la web sirviéndonos de una base de datos sencilla como Access. Tampoco es de extrañar que, llegado el momento, pasemos a cosas más serias y nos pasemos a un servidor de datos como MySQL. Aquí os mostramos una manera bastante práctica de migrar los datos de la una a la otra.

Referencia: Para realizar esta tarea es necesario que hayamos descargado el driver ODBC y lo hayamos instalado en nuestro sistema Windows. Esta labor se puede conocer en un artículo de DesarrolloWeb.com: Instalar el driver ODBC para MySQL [http://www.desarrolloweb.com/articulos/897.php].

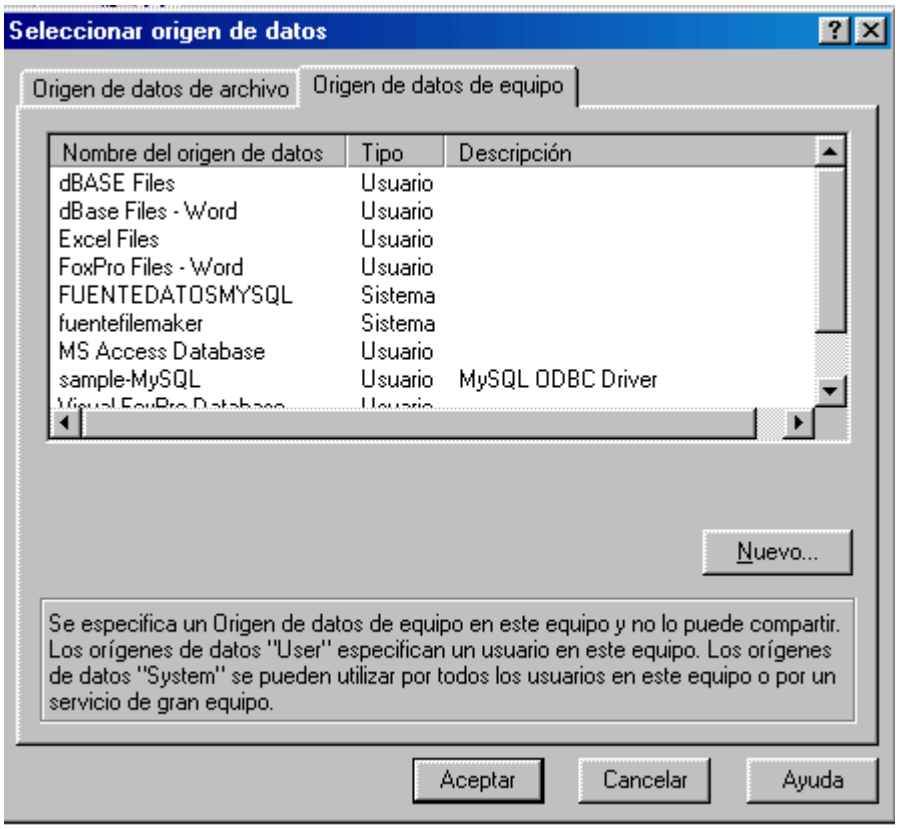
Para exportar una tabla a Mysql, hay que abrir la base de datos y seleccionar la tabla. Después, hacer clic sobre Archivo->Exportar. En la pantalla de exportar, en la opción Guardar como tipo, seleccionar ODBC databases().

Una vez se ha hecho esto, aparece una ventana que nos pregunta el nombre que le queremos dar a la tabla en Mysql, por defecto aparece el mismo.



Haz clic sobre "Aceptar", y aparecerá la pantalla en la que se pide que selecciones el origen de datos ODBC:

Nota: pudiera ser en algún caso que los tipos de los datos de la base en los sistemas MySQL y Access no sean totalmente compatibles y se produzca alguna anomalia al exportarlos. Realmente es una posibilidad que pensamos, aunque en las pruebas que hemos realizado no hemos visto ningún tipo de problema, bien es cierto que los campos que hemos trabajado no eran muy raros.



Seleccionar origen de datos de equipo, y dentro de esta el nombre de la fuente de datos que hemos creado anteriormente. Una vez la has seleccionado y has hecho clic sobre "Aceptar", aparecerá la pantalla de configuración del driver por si deseas marcar para esta acción en concreto algunas de las opciones de configuración que aparecen en el driver ODBC. Si no deseas marcar ninguna, haz clic sobre "OK" y los datos comenzarán a exportarse.

> **Nota:** si estamos exportando los datos hacia o desde un servidor de bases de datos alojado en algún proveedor de Hosting, tenemos que tener en cuenta que estos no siempre incluyen en su paquete básico el acceso remoto al servidor de base de datos, o requiere de un aviso explicito por parte del cliente para su configuración.

> **Referencia:** si deseamos realizar una migración de datos en el otro sentido, es decir, desde MySQL hacia Access, será muy indicado leer otro artículo en DesarrolloWeb que explica el proceso detalladamente. Exportar datos de MySQL a Microsoft Access 2000 [http://www.desarrolloweb.com/articulos/865.php].

Buscador simple en ASP mejorado

El buscador que se explica en el manual Buscador Simple para ASP [http://www.desarrolloweb.com/manuales/25] se puede mejorar, como creo que ya comentamos, utilizando el script del artículo Crea tu propio buscador [http://www.desarrolloweb.com/articulos/292.php], que implementaba un sistema para que el sistema de búsqueda utilizase varias palabras clave y algunos operadores para relacionarlas, como el operador + o el operador espacio, que quieren decir que han de relacionarse las distintas palabras clave con la función lógica AND y OR respectivamente.

El objetivo de este artículo es realizar esa integración de un sistema con el otro y la publicación del script resultante, comentado en la medida de lo posible. Lo primero sería tener claros las dos fuentes de información con las que estamos trabajando, para lo que aconsejamos necesariamente su lectura comprensiva.

- Buscador simple en ASP [http://www.desarrolloweb.com/manuales/25]

- Crea tu propio buscador [http://www.desarrolloweb.com/articulos/292.php]

Unimos los dos scripts

La página del buscador simple que tendremos que editar se llama buscar.asp. Para empezar, vamos a colocar las funciones que hay en el ejercicio Crea tu propio buscador [http://www.desarrolloweb.com/articulos/292.php] en cualquier parte de buscar.asp, aunque preferiblemente en la parte de arriba. Estas funciones, por si alguien no las identifica, son:

function Sacar(cadena,campos)
Extrae cada uno de los campos del criterio de búsqueda introducido y los va relacionando según los operadores que se han utilizado. Todo esto lo va volcando en una sentencia en lenguaje SQL.

function GeneraSql(cadena,tabla,campos)
Crea la sentencia SQL definitiva.

Las descripciones completas de estas funciones y la explicación de sus acciones está, más o menos realizada en el artículo Crea tu propio buscador [http://www.desarrolloweb.com/articulos/292.php].

Editamos el script de buscar.asp para construir una sentencia SQL mejorada

La línea donde antes se construía la sentencia SQL (variable Temp.) la comentamos y la volvemos a hacer de otra manera.

Primero debemos crear un array con los campos de la tabla donde vamos a realizar las búsquedas. El array debe tener una posición más de las necesarias, porque si no nos fallará el ejemplo. Vamos a buscar en los campos "Des" y "pal" de la tabla, que contienen las descripciones y palabras clave de cada uno de los registros.

```
dim campos(3) 'el tamano del array debe superar en uno al numero de campos
campos(1)="Des"
campos(2)="pal"
```

Luego extraemos la cadena de búsqueda del formulario que estamos recibiendo
`cadena=Request.form("palabra")`

Creamos una variable con el nombre de la tabla de la base de datos donde hay que buscar.

`tabla_bbdd="Tabla1"`

Finalmente creamos la sentencia SQL realizando una llamada a la función GeneraSql comentada antes. Guardamos la sentencia SQL en la variable Temp, que era la variable donde antes se había guardado la sentencia del script buscar.asp original.

`Temp=GeneraSql(cadena,tabla_bbdd,campos)`

> **Nota:** He decidido no cambiar los nombres de las variables para que el ejemplo de buscar.asp siga funcionando sin problemas.

Ya está todo

Con los cambios indicados hasta ahora ya tenemos el ejercicio terminado. Ahora las búsquedas serán mucho más complejas porque la sentencia SQL es mucho más avanzada. Las posibilidades de este script se han multiplicado.

El resto del código se deja como estaba. Al ejecutar la sentencia SQL guardada en la misma variable Temp, pero que ahora es más avanzada se produce un recordset cuyo recorrido se realiza de la misma manera que antes.

He dejado la sentencia SQL a la vista, imprimiéndola en la página, para que cualquiera que ejecute el script pueda ver qué sentencia se está generando.

Para terminar, os ofrecemos la posibilidad de que descarguéis el archivo buscar.asp modificado [http://www.desarrolloweb.com/descargas/descargar.php?descarga=1119].

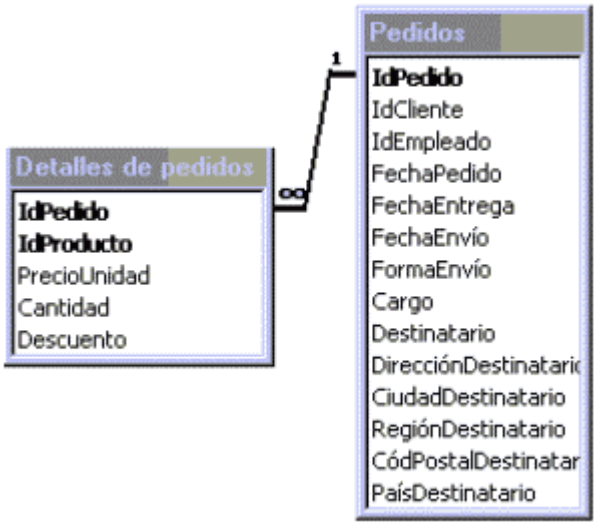
Recoger valor del campo autonumérico después de insertar en ASP

Cuando en una base de datos tenemos dos tablas relacionadas con una relación uno a varios, la clave primaria de la tabla con cardinalidad 1 se encuentra en la tabla con cardinalidad n como clave foránea. Normalmente las claves primarias suelen utilizar campos de tipo autoincremento.

El presente artículo detalla cómo averiguar el valor que se ha asignado a la clave primaria en una operación de inserción, para de esta manera poder introducir registros en la tabla relacionada que cuenta con la clave foránea.

> **Nota:** Leer los comentarios, añadidos por los visitantes acerca de este artículo, para obtener algunas guías con las que resolver de otra manera el problema planteado. Hay una vía más sencilla y un ejemplo de cómo utilizarla, que puede ayudar si da problemas la descripción planteada en este documento. Los archivos que se ofrecen para descarga de manuales no incluyen los comentarios de los visitantes, por lo que hay que verlos en la versión online de la página.

Para el ejemplo, utilizaremos las siguientes tablas:



En un principio vamos a partir que tenemos un nuevo pedido para el cual tenemos que añadir varios detalles. En este supuesto, necesitamos insertar primero el pedido, recoger el valor que la base de datos le ha asignado al campo IdPedido, y a continuación insertar los distintos detalles con ese valor de IdPedido.

Para hacer esto, vamos a utilizar un objeto recordset:

```
Dim rspedidos
Set rspedidos=Server.CreateObject("ADODB.Recordset")
```

A continuación abrimos el Recordset:

```
With rspedidos
```

```
.open "Pedidos" ,adOpenDynamic, AdLockOptimistic,adCmdTableDirect
```

adOpenDynamic: Cursor que no tiene un conjunto único de registros, y en el que los cambios serán visibles en el recordset.
AdLockOptimistic: El registro no se bloquea hasta que no se insertan los datos
adCmdTableDirect: sirve para indicar el nombre de la tabla

La combinación de cursor, tipo de bloqueo y si el campo esta indexado o no hace que podamos recoger el campo ID, otras combinaciones pueden no funcionar.

A continuación, creamos un nuevo registro, e insertamos en el los valores de los campos:

```
.AddNew
  .Fields("idcliente")=16
  .Fields("cargo")=500
  .Fields("destinatario")="pepe"
```

Una vez se han introducido todos los valores de los campos, se actualiza el recordset

```
.update
```

Y a continuación, sacamos el ID que le ha asignado la base de datos:

```
  MiID= .Fields("IdPedido")
End With
```

Ahora contamos con una variable miID que nos permite hacer los inserts correspondientes en la base de datos, bien con recordsets o con sentencias sql del tipo Insert Into.

> **Nota:** Hemos visto como insertar registros en ASP, pero hasta ahora siempre habíamos explicado otro método, consistente en construir la sentencia SQL para la inserción y ejecutándola por medio del método execute() del objeto connection. Podemos ver ese tipo de inserciones en el artículo Creación de un nuevo registro [http://www.desarrolloweb.com/articulos/258.php?manual=8]

Código completo:

```
Dim rspedidos
Set rspedidos=Server.CreateObject("ADODB.Recordset")
With rspedidos
  .open "Pedidos" ,adOpenDynamic, AdLockOptimistic,adCmdTableDirect
  .AddNew
    .Fields("idcliente")=16
    .Fields("cargo")=500
    .Fields("destinatario")="pepe"
 .update
  MiID= .Fields("IdPedido")
End With
```

> **Nota:** Puede ser que nuestro sistema no tenga declaradas las variables del sistema del tipo adOpenDynamic, AdLockOptimistic o adCmdTableDirect. Para que queden definidas debemos incluir el archivo "adovbs.inc".

Mandar mails desde ASP

Una de las tareas más habituales y que resultan más útiles en el trabajo con ASP y en general en cualquier lenguaje de programación de servidor, es el envío de mails desde el propio servidor de páginas web.

> **Referencia:** En caso de que necesitemos programar el envío de correo electrónico en nuestra página utilizando PHP, también hemos publicado un artículo en DesarrolloWeb.com llamado Mandar mails desde PHP [http://www.desarrolloweb.com/articulos/969.php].

El caso más directo en el que utilizar un envío de mails a través del servidor consiste en mandar datos de un formulario rellenado por el visitante a los administradores de la página. Éstos datos se pueden enviar colcando en la etiqueta <form> el atributo action="mailto:correo@tudominio.com", tal como pudimos explicar en el los capítulos de creación de formularios en nuestro manual de HTML [http://www.desarrolloweb.com/articulos/647.php?manual=21]. El problema de este tipo de envío consiste en que se tiene que realizar a través del correo electrónico que un usuario tenga configurado en su ordenador y, en caso de que no tuviera ningún email configurado, no se podría realizar el envío. Si enviamos el mail con el servidor, siempre se podrá realizar el envío sin problemas.

Otro caso en el que podríamos utilizar el envío de mensajes con el servidor es la confirmación de una compra en una tienda virtual, o el envío de un boletín de novedades mensual a los correos de los usuarios que tenemos en la base de datos.

Cómo enviar correos con ASP

Para el envío de correos electrónicos desde ASP debemos utilizar un componente especial del servidor.

> **Referencia:** Hablamos de componentes de servidor en un artículo de DesarrolloWeb.com [http://www.desarrolloweb.com/articulos/657.php?manual=11].

Algún componente que podemos señalar:

CDONTS: El más popular componente porque lo entregan en las distribuciones básicas del motor ASP, aunque no está incluido de casa en XP profesional.

> **Nota:** Si queremos instalar la librería CDONTS en WinXP podemos conseguir la el archivo dll de otro servidor y moverlo al directorio system del XP. Luego la registramos como se indica en el artículo Componentes de servidor [http://www.desarrolloweb.com/articulos/657.php?manual=11].

AspEmail [http://www.aspemail.com/] Componente comercial de la empresa Persits [http://www.persits.com/], que incluye más facilidades que CDONTS.

AspMail [http://www.serverobjects.com/products.htm#aspmail]. Es otro componente comercial para el envío de correos. De serverobjects.com [http://www.serverobjects.com/].

Estas son algunas de las opciones, por lo menos las más comunes. Para aprender a manejar estos componentes lo mejor es acceder a las páginas de las empresas que los distribuyen, ya que guardan documentación muy buena.

> **Referencias:** Si queremos aprender cómo utilizar CDONTS podemos leer el un artículo de DesarrolloWeb.com que trata sobre el manejo de CDONTS para hacer un sistema de recomendación de una página a través de correo electrónico [http://www.desarrolloweb.com/articulos/999.php]. Además, recomiendo la lectura de un artículo de ASP Fácil que explica los distintos usos y configuraciones de CDONTS [http://www.aspfacil.com/articulos/cdomail.asp].
>
> También tenemos un artículo que muestra como enviar un formulario por mail desde ASP [http://www.desarrolloweb.com/articulos/1037.php] utilizando en este caso el componente ASPEmail.

La elección de uno u otro componente depende un poco del servidor donde vamos a trabajar. Muchos proveedores de hosting disponen de componentes instalados en sus servidores para realizar estas tareas, en este caso la elección estaría clara, pues deberíamos utilizar ese componente. Deberíamos, pues, preguntar a nuestro proveedor de hosting cuál es la opción que debemos utilizar. En caso de no tener ningún componente instalado, podemos probar el CDONTS, que podríamos utilizarlo, en principio, en cualquier caso.

Crear base de datos Access desde ASP

Estas son unas pautas de trabajo y unas referencias que he encontrado útiles a la hora de solucionar un problema en ASP, consistente en crear una base de datos en Access (fichero .mdb) desde cero, es decir, crear el archivo .mdb vacío para, una vez creadas también las tablas, trabajar con la base de datos como si hubiese sido creada con Access.

Lógicamente, con Access es mucho más sencillo crear las bases de datos, pero este mecanismo tiene dos ventajas. Una simple: que no necesitas tener Access para crear las bases de datos, y otra más interesante: que puedes crear bases de datos en línea y dinámicamente, de modo que queden guardadas en tu servidor para realizar los trabajos que estimes oportuno.

> **Nota:** Se puede crear una base de datos de Access a través de Visual Basic, ASP, u otros medios porque en realidad lo que se está utilizando es el motor Jet, que son unas DLL que tienen todos los sistemas Windows para manejar los archivos .mdb. Más información en la FAQ Qué es el motor de base de datos Microsoft Jet [http://www.desarrolloweb.com/faq/115.php].

Métodos para crear una BD Access 2000

Existen dos métodos para crear una base de datos Access: ADO y DAO. Vamos a ver un pequeño ejemplo de cada uno.

Atención: Para los dos casos, los directorios tienen que tener permisos de escritura para que funcione correctamente.

Ejemplo en ADO, Requiere MDAC 2.0

Para el ejemplo se necesita tener instalado el Microsoft Data Access Component (MDAC 2.0), que se incluye con la instalación de Microsoft Access 2k, o, sino, bajar la ultima versión del (MDAC) desde http://www.microsoft.com/data (donde también se encuentra el Component Checker para saber que versión tienes instalada).

Según información de Microsoft la versión 1.5 del MDAC contiene algunos bugs, por lo que recomiendan actualizarse.

```
Dim basedatos
Set basedatos = CreateObject("ADOX.Catalog")
basedatos.Create "Provider=Microsoft.Jet.OLEDB.4.0;Jet OLEDB:Engine Type=5;Data Source=" & Server.MapPath
("prueba.mdb")
Set basedatos = Nothing
```

En la cadena utilizada para crear la base de datos, el valor Engine Type indicado (5) sirve para que la base de datos tenga formato Access 2000. Podíamos haber puesto otros valores para otras versiones del motor, por ejemplo 3 para Access 95, o el valor 4 para Access 97.

Ejemplo en DAO, requiere DAO

Se necesita DAO 3.6 ó DAO 3.5. En caso de utilizar DAO 3.5, se ha de modificar "DAO.DBEngine.36" por "DAO.DBEngine.35"

```
Dim motor
Set motor = CreateObject("DAO.DBEngine.36")
motor.CreateDatabase Server.MapPath(NomBD&".mdb"), ";LANGID=0x0409;CP=1252;COUNTRY=0", 64
Set motor = Nothing
```

El tercer parámetro del método para crear la base de datos (64) es para indicar la versión de la base de datos, en este caso Access 2000. En el caso de desear otro formato se puede modificar por ejemplo a 32 en caso de Access 97 ó 16 para Access 95.

Conclusión

Las pruebas las hice en 2 PCs con Windows Professional 2000 (SP3) y Office 2000 (SP3) instalado y funcionaron perfectamente, lo mismo que en mi servidor (hosting contratado) y en el de Brinkster (cuenta gratuita) sin ningun drama. También se han probado con éxito, en la redacción de DesarrolloWeb.com, utilizando el sistema Windows 98 y el servidor Personal Web Server.

Por último, aquí se puede ver un ejemplo que hice para generar las db desde ADO ó DAO (3.6). El código esta bastante comentado con respecto al funcionamiento y los requerimientos del servidor.

```
<%@LANGUAGE="VBSCRIPT"%>
<%
Generar = Request.Form("action")
if Generar <> "" then

On Error Resume Next

Metodo = Request.Form("metodo")
Formato = Request.Form ("formato")
NomBD = Request.Form ("nomBD")

' ************* Comprobamos segun que método ************

if Metodo = "ADO" then

   if Formato = "1" then
      FormatoBD = 5
      FormatoN = "Ms Access 2000"
   elseif Formato = "2" then
      FormatoBD = 4
      FormatoN = "Ms Access 97"
   elseif Formato = "3" then
      FormatoBD = 3
      FormatoN = "Ms Access 95"
   end if
' **********************
'    Para los dos casos los directorios tienen que
'    tener permisos de escritura para que funcione
'    correctamente.
' **********************
'    Ejemplo en ADO, requiere MDAC 2.0
'
'    para saber que version esta instalada en el sistema
'    se puede usar el Component Checker Tool de
'    Microsoft
'    http://www.microsoft.com/data/download.htm#CCinfo
'    ó bajar la ultima version en
'     http://www.microsoft.com/data/download.htm
'
'    Jet10 = 1
'    Jet11 = 2
'    Jet20 = 3 <---------- para Access 95
'    Jet3x = 4 <---------- para Access 97
'    Jet4x = 5 <---------- para Access 2000
' **********************
' ***** Comienzo ADO ************
Dim Catalog
Set Catalog = CreateObject("ADOX.Catalog")
Catalog.Create "Provider=Microsoft.Jet.OLEDB.4.0;Jet OLEDB:Engine Type="&FormatoBD&";Data
Source="&Server.MapPath(NomBD&".mdb")
Set Catalog = Nothing
' ************ Fin ADO *****************

else
   if Formato = "1" then
      FormatoBD = 64
      FormatoN = "Ms Access 2000"
   elseif Formato = "2" then
      FormatoBD = 32
      Formato = "Ms Access 97"
   elseif Formato = "3" then
      FormatoBD = 16
```

```
      FormatoN = "Ms Access 95"
   end if

****************
'    Ejemplo en DAO, requiere DAO 3.6 ó DAO 3.5 para
'    DAO 3.5 modificar "DAO.DBEngine.36" por
'    "DAO.DBEngine.35"
'
'    dbVersion10 = 1
'    dbVersion11 = 8
'    dbVersion20 = 16 <------------ para Access 95
'    dbVersion30 = 32 <------------ para Access 97
'    dbVersion40 = 64 <------------ para Access 2000
****************
' ********** Comienzo DAO **********
Dim Engine
Set Engine = CreateObject("DAO.DBEngine.36")
Engine.CreateDatabase Server.MapPath(NomBD&".mdb"), ";LANGID=0x0409;CP=1252;COUNTRY=0", FormatoBD
Set Engine = Nothing
' ********** Fin DAO ***********
end if
end if
%>

<html>
<head>
<title>Crear *.mdb</title>
<style type="text/css">
<!--
body {
   font-family: Arial, Helvetica, sans-serif; font-size: x-small
}
-->
</style>
</head>
<body bgcolor="#FFFFFF" text="#000000">

<%
if Generar <> "" then
   if Err then
   Response.Write "Hubo un error.<br><br>"
   Response.Write "<u>Error:</u> " & Err.Description & "<br>"
   Response.Write "<u>Error N°:</u> " & Err.Number &"."
   else
   Response.Write "Base de datos <u>"& NomBD &"</u> fué creada exitosamente!!<br>"
   Response.Write "Formato: "&FormatoN & "<br>"
   Response.Write "Metodo usado: "&Metodo
   end if
else
%>

<!-- Formulario de creacion //-->
<form name="creador" action="crear_db.asp" method="POST">
Nombre de la BD:
<input type="text" name="nomBD"><br>
Formato:
<select name="formato">
   <option value="1" selected>Access 2000</option>
   <option value="2">Access 97</option>
   <option value="3">Access 95</option>
</select><br>

Método:<br>
<input type="radio" name="metodo" value="ADO" checked><small>ADO (requiere MDAC 2.0)</small><br>
<input type="radio" name="metodo" value="DAO"><small>DAO (requiere DAO 3.6)</small><br><br>

<input type="submit" value="Crear">
<input type="hidden" name="action" value="si">
</form>
<!-- Fin formulario //-->

<%
end if
%>
```

```
</body>
</html>
```

El código de este ejemplo se puede descargar en un archivo comprimido [http://www.desarrolloweb.com/descargas/descargar.php?descarga=2561].

Una referencia útil que se consultó para documentar este artículo es Crear una base de datos MDB con Visual Basic [http://www.pstruh.cz/tips/detpg_createmdb.htm] (En inglés).

Uso de CDONTS para script de recomendar a un amigo en ASP

Este es un ejemplo completo y sencillísimo de cómo enviar un correo usando el componente CDONTS de IIS.

Dicho componente está presente en muchas de las versiones de Windows en su modo servidor, como Windows NT 4 o Windows 2000 , pero no así en Windows 98, por no disponer de un servidor de envío de correo (SMTP). Atención, que en Windows NT y 2000 habría que tener el servidor SMTP correctamente configurado y los usuarios de Windows XP, que no disponen de esa herramienta desde el principio, aunque podrían obtenerla desde otro sistema Windows, tal como se cuenta en la FAQ: CDONTS en Windows XP [http://www.desarrolloweb.com/faq/99.php]

> **Referencia:** Podemos conocer otras opciones para el envío de correos desde el servidor con ASP [http://www.desarrolloweb.com/articulos/937.php] en un artículo de DesarrolloWeb.com

Script para recomendar a un amigo

El ejercicio siguiente es un script que permite mandar un email desde nuestra página a un amigo del visitante que lo desee. Es uno de los típicos sistemas de "recomienda a un amigo".

Los datos para rellenar el mail, los ponemos a mano. Si se quiere enviar los datos de un formulario sólo tenemos que recogerlos con instrucciones tan sencillas como estas:

```
cBody = Request.Form("Body")
cPara = = Request.Form("Amiguete")
```

Y ahora el ejemplo, en el que se muestra como enviar a un amigo, una copia y una copia oculta, lo común.

```
<%@ Language=VBScript%>
<html>

<head>
  <title>Enviar a un amigo</title>

</head>
<body><%
Dim cBody, n

For Each n In Request.Form
   cBody = cBody & n & ": " & Request.Form(n) & chr(13)
Next

Set oCDO = Server.CreateObject("CDONTS.NewMail")

'Asignamos las propiedades al objeto
oCDO.From = "fernan@tudominio.com"
oCDO.To = "foc@tudominio.com"
oCDO.Subject = "Asunto del mensaje"
oCDO.Body = cBody
'oCDO.Cc = "resal@tudominio.com;webmaster@tudominio.com"
oCDO.Bcc = "quinqui@tudominio.com"
'oCDO.MailFormat = 0

oCDO.Send
```

```
Set oCDO = Nothing 'Liberar...
'Mostramos mensaje de que seenvió con éxito.
Response.Write "¡Se envió Ok, qué fácil!!"

%>
</body>
</html>
```

Bueno, amigos, espero que les pueda servir, como ven, hay poco que programar.

> **Referencia:** Se puede consultar un artículo de ASP Fácil para conocer más funcionalidades del envío de correos con CDONTS. http://www.aspfacil.com/articulos/cdomail.asp [http://www.aspfacil.com/articulos/cdomail.asp]
>
> Disponemos de otro artículo en DesarrolloWeb.com que realiza un script con básicamente la misma funcionalidad [http://www.desarrolloweb.com/articulos/1372.php], por si interesa ver otro punto de vista.

Enviar un formulario por email con ASP

Si deseamos que, al pulsar el botón de envío de un formulario, se manden los datos por email utilizando el servidor y sin estar supeditados a la configuración del cliente para saber si ciertamente ese mensaje se pudo enviar, debemos utilizar alguna tecnología de programación de páginas del lado del servidor. En este artículo vamos a presentar un ejemplo sobre cómo hacer esta tarea con ASP.

ASP (Active Server Pages) [http://www.desarrolloweb.com/articulos/393.php] es la tecnología de scripting del lado del servidor [http://www.desarrolloweb.com/articulos/715.php?manual=27] desarrollada por Microsoft. Con este tipo de programación podemos construir páginas que muestren un formulario y manden un correo electrónico automáticamente con los datos recibidos, una vez enviado.

Podríamos utilizar un esquema de código como el siguiente:

```
<%
if request.form="" then
'no recibo formulario, entonces lo muestro
%>
<form action="formulario_mail_asp.asp" method="POST">
Nombre: <input type="Text" name="nombre" size="12" maxlength="200">
<br>
Email: <input type="Text" name="email" size="12" maxlength="200">
<br>
<input type="submit" value="Enviar">
</form> <%
else
'si que recibo un formulario, entonces lo trato
'recojo los datos
nombre = request.form("nombre")
email = request.form("email")
'compongo el cuerpo del mensaje
cuerpo = "Formulario recibido" & VBNEWLINE & VBNEWLINE
cuerpo = cuerpo & "Nombre: " & nombre & VBNEWLINE
cuerpo = cuerpo & "Email: " & email
'mando el correo...
'.....................
response.write "Gracias por rellenar el formulario. Se ha enviado correctamente." end if %>
```

En el anterior script utilizamos if (request.form="") para saber si estamos recibiendo o no información de un formulario.

Si no hemos recibido nada (porque en request.form tenemos una cadena vacía) este if saldría por su caso verdadero, en el que tendremos que presentar el formulario en la página.

El caso else, cuando sí que recibíamos un formulario, recogemos sus datos y creamos el cuerpo

del mail que enviaremos a la dirección del administrador.

El resto del código, utilizado para definir las propiedades del email y enviarlo, todavía no lo hemos indicado. Antes una aclaración.

ASP no tiene entre las funciones del lenguaje una que sirva para enviar correos electrónicos. Sin embargo, podemos utilizar un componente ActiveX del servidor [http://www.desarrolloweb.com/articulos/657.php] para realizar esas acciones.

El componente CDONTS [http://www.desarrolloweb.com/articulos/999.php] (presente en muchas de las instalaciones de IIS [http://www.desarrolloweb.com/manuales/36/]) serviría para realizar el envío del mail, pero también existen en el mercado otros componentes de servidor comerciales para realizar esas acciones con mayores funcionalidades. Uno de ellos es AspEmail [http://www.aspemail.com/], que es el que vamos a utilizar en este script, pero no es el único. Hablamos de las distintas opciones en el artículo Enviar mails desde ASP [http://www.desarrolloweb.com/articulos/937.php].

> **Nota:** Insistimos en que el siguiente código, que hace la acción de enviar un email, no funcionará si no tenemos el componente ASP AspEmail [http://www.aspemail.com/] instalado correctamente en nuestro servidor. Además, deberíamos consultar la documentación del componente, para comprobar que en la versión que descarguemos sigue funcionando todo como aparece a continuación.

El código para enviar un mail podría ser algo como esto:

```
'creo el objeto correo
set mail = server.createObject("Persits.MailSender")
'configuro el mensaje
'señalo el servidor de salida para enviar el correo
mail.host = "mail.tudominio.com"
'indico la dirección de correo del remitente
mail.from = "loquesea@tudominio.com"
'indico la dirección del destinatario del mensaje
mail.addAddress "loquesea@tudominio.com"
'indico el cuerpo del mensaje
mail.body = cuerpo
'lo envio
'aseguro que no se presenten errores en la página si se producen
On Error Resume Next
mail.send
if Err ><0 then
response.write "Error, no se ha podido completar la operación"
else
response.write "Gracias por rellenar el formulario. Se ha enviado correctamente." end if
```

Nos podemos guiar por los comentarios para hacernos una idea del funcionamiento de este trozo de código. Básicamente se crea el objeto mail, que contendrá el correo que se va a generar, y se definen sus propiedades como el servidor de correo a utilizar, el destinatario o el asunto. Luego se envía y se realiza una sencilla tarea de detección de errores para saber si se pudo realizar la tarea con éxito o no y mostrar un mensaje adecuado para cada caso.

> **Nota:** no se pretenden explicar los pormenores de este Control Activex del servidor, ya que hay una excelente documentación y ejemplos en la página web de la empresa que lo ha desarrollado. En caso de que se desee aprender algo más, por favor, dirigiros a la página http://www.aspemail.com/ o la del componente del que dispongáis.que dispongáis.

Multicultura en ASP.NET

Créditos

Este artículo nos lo ofrece **Clikear.com** [http://www.clikear.com/], una web dedicada por completo a .NET, que ofrece alojar tus páginas ASP.NET gratuitamente.

En este articulo presentamos un ejemplo de una pagina ASP .NET para conocer las clases que sirven para hacer una web multicultura, una de las nuevas e interesantes características de la plataforma .NET.

Presentamos una pagina en la que se nos muestra los diferentes formatos de hora, fecha, y numeros en distintos tipos de configuraciones para distintas culturas.

Estudiando este ejemplo el lector podrá conocer algunos de los metodos mas usados de la clase System.Globalization, que se utiliza para crear webs multiculturales.

Se puede descargar el archivo comprimido con el ejemplo [http://www.desarrolloweb.com/descargas/descargar.php?descarga=6991].

Servidores en EEUU y el asp

Cuando usamos asp en servidores americanos hay que tener en cuenta unos pequeños detalles a la hora de programar. Principalmente estos problemas se derivan de los formatos de fecha y los formatos de decimales.

Problemas con las Fechas:

Al ser el asp un lenguaje de servidor, es decir que el servidor es el que lo interpreta y nos devuelve código html. El problema viene cuando el servidor de USA, nos interpreta las fechas, y no las devuelve con el formato americano, es decir:

formato americano: mes/dia/año
formato español: día/mes/año
Esto puede dar lugar a confusiones entre los usuarios de la web, que pueden ver la fecha de un artículo y confundirla.

Problemas con los decimales:

El otro problema viene dado por el formato de los decimales de los números, ya que en el formato americano los decimales se representan por coma y en español se determinan por punto.

formato americano: 2,78
formato español: 2.78

Solución a los problemas:

Todos estos problemas tienen una solución más fácil de lo esperada ya que hay que poner una sola línea de código en cada página para arreglar este problema:

<%@LCID = 1034%>

Este código debe sustituir al habitual que ponemos en toda página asp:

<%@LANGUAGE="VBSCRIPT" CODEPAGE="1252"%>

El número de LCID encima, 1034, pondrá los ajustes de lugar sobre el servidor al español, cambiar el lugar a su propio país el que usted tendrá que usar la Carta LCID en el fondo de esta página para encontrar que su propio lugar ID numera y luego sustituye 1034 por su propio número de LCID.

Para que esto funcione es necesario que en su servidor estén instalados los archivos necesarios para el cambio de lengua, normalmente tienen múltiples lenguas instaladas,pero en el caso de

poner una no instalada en su servidor recibirá el siguiente error ' LCID especificado no es disponible ' en ese caso solo podrá solucionarse poniéndose en contacto el administrador de su servidor para que la instale.

Esto sólo cambiará el formato muestran las fechas, veces, y divisas,pero no moverá fechas a otra zona horaria.

Locale Description	LCID	Locale Description	LCID
Afrikaans	1078	Icelandic	1039
Albanian	1052	Indonesian	1057
Arabic - United Arab Emirates	14337	Italian - Italy	1040
Arabic - Bahrain	15361	Italian - Switzerland	2064
Arabic - Algeria	5121	Japanese	1041
Arabic - Egypt	3073	Korean	1042
Arabic - Iraq	2049	Latvian	1062
Arabic - Jordan	11265	Lithuanian	1063
Arabic - Kuwait	13313	FYRO Macedonian	1071
Arabic - Lebanon	12289	Malay - Malaysia	1086
Arabic - Libya	4097	Malay – Brunei	2110
Arabic - Morocco	6145	Maltese	1082
Arabic - Oman	8193	Marathi	1102
Arabic - Qatar	16385	Norwegian - Bokmål	1044
Arabic - Saudi Arabia	1025	Norwegian - Nynorsk	2068
Arabic - Syria	10241	Polish	1045
Arabic - Tunisia	7169	Portuguese - Portugal	2070
Arabic - Yemen	9217	Portuguese - Brazil	1046
Armenian	1067	Raeto-Romance	1047
Azeri - Latin	1068	Romanian - Romania	1048
Azeri - Cyrillic	2092	Romanian - Moldova	2072
Basque	1069	Russian	1049
Belarusian	1059	Russian - Moldova	2073
Bulgarian	1026	Sanskrit	1103
Catalan	1027	Serbian - Cyrillic	3098
Chinese - China	2052	Serbian - Latin	2074
Chinese - Hong Kong SAR	3076	Setsuana	1074
Chinese - Macau SAR	5124	Slovenian	1060
Chinese - Singapore	4100	Slovak	1051
Chinese - Taiwan	1028	Sorbian	1070
Croatian	1050	Spanish - Spain	1034
Czech	1029	Spanish - Argentina	11274
Danish	1030	Spanish - Bolivia	16394
Dutch - the Netherlands	1043	Spanish - Chile	13322
Dutch - Belgium	2067	Spanish - Colombia	9226
English - Australia	3081	Spanish - Costa Rica	5130
English - Belize	10249	Spanish - Dominican Republic	7178
English - Canada	4105	Spanish - Ecuador	12298
English - Caribbean	9225	Spanish - Guatemala	4106

English - Ireland	6153	Spanish - Honduras	18442
English - Jamaica	8201	Spanish - Mexico	2058
English - New Zealand	5129	Spanish - Nicaragua	19466
English - Phillippines	13321	Spanish - Panama	6154
English - South Africa	7177	Spanish - Peru	10250
English - trinidad	11273	Spanish - Puerto Rico	20490
English - United Kingdom	2057	Spanish - Paraguay	15370
English - United States	1033	Spanish - El Salvador	17418
Estonian	1061	Spanish - Uruguay	14346
Farsi	1065	Spanish - Venezuela	8202
Finnish	1035	Sutu	1072
Faroese	1080	Swahili	1089
French - France	1036	Swedish - Sweden	1053
French - Belgium	2060	Swedish - Finland	2077
French - Canada	3084	Tamil	1097
French - Luxembourg	5132	Tatar	1092
French - Switzerland	4108	thai	1054
Gaelic - Ireland	2108	Turkish	1055
Gaelic - Scotland	1084	Tsonga	1073
German - Germany	1031	Ukrainian	1058
German - Austria	3079	Urdu	1056
German - Liechtenstein	5127	Uzbek - Cyrillic	2115
German - Luxembourg	4103	Uzbek – Latin	1091
German - Switzerland	2055	Vietnamese	1066
Greek	1032	Xhosa	1076
Hebrew	1037	Yiddish	1085
Hindi	1081	Zulu	1077
Hungarian	1038		

Redirigir al navegador a una URL con ASP al detalle

Nos ha llegado una consulta interesante que creo que no habíamos tratado con profundidad anteriormente. Se trata del método redirect del objeto response, que en determinadas situaciones da un error, que nos relataba un visitane de la siguiente manera:

Siempre trabaje en win2k e IIS para paginas ASP personales, pero ahora estoy en Win98 con PWS y las paginas que tenia funcionando en IIS con PWS me genera un error el Response.Redirect con el siguiente mensaje de error:

"Los enbezados HTTP ya están escritas en el explorador cliente. Cualquier cambio en el encabezado HTTP se debe hacer antes de escribir el contenido de la página."

Método redirect

Es un método del objeto response y sirve para mandar al navegador a una página que debe indicarse como parámetro. Un ejemplo de utilización sería:

```
response.redirect "tu_pagina.html"
```

Esto haría que nuestro navegador se situara en la página tu_pagina.html.

El redirect se realiza enviando en las cabeceras del HTTP la orden de redirección. El caso es que no se puede enviar nada en las cabeceras del HTTP si ya se ha empezado a escribir texto de la página web, aunque sea un simple espacio.

Por tanto, este código es incorrecto:

```
<head>
<title>Hola</title>
</head>
<%
   response.redirect "xxx.asp"
%>
```

Es incorrecto porque antes de hacer el redirect se ha escrito dentro de la página web, en este caso es información de cabecera, pero esta cabecera del documento HTML no es la mencionada cabecera del HTTP, que la mandan los servidores internamente sin que el programador participe activamente, salvo para introducir información como un redirect o un refresh a partir de funciones ASP.

Almacenamiento en buffer de la página

En instalaciones de ASP a partir de la 3.0 se utiliza un buffer de almacenamiento de la página que tiene como misión ir recogiendo el código HTML resultante de la ejecución de la página ASP, antes de mandarlo al cliente. El buffer se va llenando y cuando termina de ejecutarse la página se manda definitivamente por Internet hasta el ordenador del usuario.

La utilización del buffer en ASP 3.0 (IIS 5.0) se encuentra predeterminada, mientras que en otras versiones de ASP, como la del Personal Web Server (PWS), está predeterminado para no utilizarse.

Según se apunta entonces, en ASP 2.0 y anteriores se va enviando la página al cliente a medida que se va ejecutando y en ASP 3.0 hasta que no se termina de ejecutar entera, la página no se envía al cliente.

Así pues, el código anterior, donde se escribía en la página antes de ejecutar un redirect, aunque está mal diseñado en cualquier caso, puede dar error o puede no dar ningún error. Todo depende de si se había o no enviado texto al cliente previamente a la realización del redirect.

Todo esto tiene mucho que ver con el buffer, pues, si se está utilizando, no se envía nada al cliente y el redirect se realiza sin haber enviado texto, aunque sí se haya escrito en la página. Si no se utiliza el buffer el texto colocado antes del redirect sí se hubiese enviado al cliente y al ejecutar el redirect aparecería el error que nos comentaba nuestro lector.

Utilización o no del buffer

Para indicar que se desea utilizar el buffer hay que escribir esta sentencia al principio del código ASP.

```
response.buffer = true
```

Si se coloca en ASP 3.0 no sirve para nada, porque esa opción ya estaba predeterminada. Pero, en cualquier caso, no molesta y nos aseguramos que se utilizará el buffer si las páginas se trasladan a un servidor con ASP 2.0.

Si deseamos hacer un redirect después de haber escrito texto en el buffer, lo correcto sería ejecutar estas dos sentencias juntas:

```
response.clear response.redirect "http://www.loquesea.com"
```

La primera línea indica que se debe vaciar el buffer, con lo que se elimina lo que se haya escrito en la página, y la segunda indica que se haga la redirección. Insisto, esto sólo es posible si se está utilizando el buffer.

> **Referencias:** La sentencia redirect se utiliza en muchos ejemplos dentro de DesarrolloWeb, pero se explica en el artículo Objeto response [http://www.desarrolloweb.com/articulos/260.php?manual=8]. Este artículo se engloba, junto con otros artículos de ASP básicos, en el manual Programación en ASP [http://www.desarrolloweb.com/articulos/manuales/8/].

Cómo hacer zonas aleatorias en asp

Una idea interesante para dar un poco de dinamismo adicional a un diseño web es crear áreas donde se muestre un contenido aleatorio, que cambie cada vez que se carga la página.

Un ejemplo de esta idea puede ser mostrar una imagen en la cabecera de la página distinta, pero no sólo nos quedamos ahí, pues podemos incluir mensajes promocionales o enlaces distintos que acompañen a la imagen.

El ejercicio es muy simple. Se debe generar un número aleatorio, para lo que utilizaremos algunas funciones matemáticas de ASP, y una estructura condicional para mostrar un contenido distinto dependiendo de el número aleatorio.

El código

Las zonas aleatorias en ASP se conseguirían con un código similar al siguiente:

```
<%
'Definimos una constante para representar el número total de zonas aleatorias.

Const totalzonas = 4

' Inicialializamos el generador del número al azar.
' Éste es el comando nos da números al azar cuando utilizamos Rnd más adelante.

Randomize


' El fórmula para generar un número al azar en una gama es:
'
' Int((upperbound - lowerbound + 1) * Rnd + lowerbound)
'
' donde upperbound sería el número máximo a alcanzar
' y lowerbound sería el número mínimo a alcanzar
' en nuestro caso el lowerbound será de 1
' y la resta del upperbound - el lowerbound será representada por
' la constante totalzonas, q determina el número total de zonas

Dim zona ' variable que determinará la zona aleatoria

' la forma descrita determinará un número a lazar entre 1 y el número de zonas(4)

zona = Int((totalzonas * Rnd) + 1)

' A continuatión através de un condicional if determinaremos que zona se mostrará
%>

<% if zona=1 Then ' si el número aleatorio es un 1 muestrame la siguiente zona%>

<b>zona1</b>

<%End if%>

<% if zona=2 Then ' si el número aleatorio es un 2 muestrame la siguiente zona%>
```

```
<b>zona2</b>

<%End if%>

<% if zona=3 Then ' si el número aleatorio es un 3 muestrame la siguiente zona%>

<b>zona3</b>

<%End if%>

<% if zona=4 Then ' si el número aleatorio es un 4 muestrame la siguiente zona%>

<b>zona4</b>

<%End if

' El beneficio de usar este método aleatorio es que no solo se pueden hacer
' imágenes aleatorias, sino que puedes hacer que cualquier cosa sea aleatoria
' un flash, una tabla, un formulario.... simplemente hay que meterlo
' entre el condicional

%>
```

Selects dependientes con ASP y base de datos

Este es un sistema de "selects dependientes" (es decir, que interactúan el uno con el otro), tomando a la vez los datos desde una base de datos Access.

Sólo será necesario **1 módulo ASP** (selecs_dependientes.asp) y la **base de datos** correspondiente (ssdd.mdb).

> **Referencia:** Tenemos otro ejemplo de selects dependientes en el que se trabaja únicamente del lado del cliente [http://www.desarrolloweb.com/articulos/1281.php] con Javascript, para el que le interese más una opción como esa.

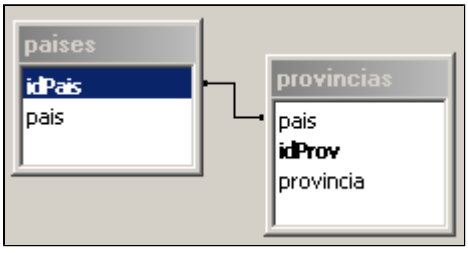
En este ejemplo, se trabajará con dos selects:

- Uno de países
- Otro de provincias o comunas

Base de datos ssdd.mdb
La base de datos deberá contener dos tablas relacionadas entre sí.

TABLA	CAMPOS
paises	idPais (autonumérico) [clave principal] pais (texto)
provincias	pais (numero) idProvincia (autonumérico) [clave principal] provincia (texto)

Con una relación entre tablas como esta:



selects_dependientes.asp

La página ASP es bastante sencilla de entender. Se trata de un script para construir los dos campos select a partir de los valores de la base de datos.

La primera vez que carga la página se muestra únicamente el primer campo select con los valores que extrae de la tabla de países. Al campo select se le ha incluído el evento Javascript **onchange** para que, en el momento que cambie la opción seleccionada, se cargue la página pasándo por parámetro el identificador del país seleccionado.

La segunda vez que carga la página -porque se cambió el valor del primer select- recibe por la URL el identificador del país seleccionado. Entonces muestra el segundo select con las opciones relacionadas con el país que recibe por parámetro, es decir, si seleccionó el país Argentina, se cargarán únicamente las provincias de ese país.

El código del ejemplo quedaría de esta manera.

```
<form name=formulario>
PAÍS: <select name=idPais onChange="location.href('selects_dependientes.asp?idPais=' + formulario.idPais.options
[formulario.idPais.selectedIndex].value)">
<%
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" & Server.MapPath("ssdd.mdb"))

SQL_pais="select * from paises order by pais asc"
set RS_pais=createobject("ADODB.Recordset")
RS_pais.open SQL_pais,conn
do while not RS_pais.eof
pais = RS_pais("pais")
idPais = RS_pais("idPais")
p = request.querystring("pais")
   if p <> "" then

      if p = pais then
      response.write "<option value="&idPais&"&pais="&pais&" selected>"&pais&"</option>"
      elseif p <> pais then
      response.write "<option value="&idPais&"&pais="&pais&">"&pais&"</option>"
      end if
else
response.write "<option value="&idPais&"&pais="&pais&">"&pais&"</option>"
end if

RS_pais.movenext
loop
RS_pais.close
%>
</select>
<% if request.querystring("idPais") <> "" then %>
PROVINCIA: <select name=ubicacion>
<%
SQL_prov="select * from provincias where pais="&request.querystring("idPais")&" order by provincia asc"
set RS_prov=createobject("ADODB.Recordset")
RS_prov.open SQL_prov,conn
do while not RS_prov.eof

pais = request.querystring("pais")
provincia = RS_prov("provincia")
%>
<option value="<%=provincia%>, <%=pais%>"><%=provincia%></option>
<%
RS_prov.movenext
loop
RS_prov.close
%>
</select><% end if %></form>
```

Se pueden descargar los archivos relacionados con este taller [http://www.desarrolloweb.com/descargas/descargar.php?descarga=1927]. (Base de datos, script y comentario del autor)

Sistema de Recomendación en ASP con CDONTS

Este sistema permite al usuario enviar desde la web, un e-mail a otra persona, invitándolo a visitar el sitio.

Solo requiere **un módulo ASP** que se encargará de mostrar el formulario de recomendación, enviar el e-mail y devolver un acuse de envío.

> **Nota:** Requiere que tu servidor soporte CDONTS Mail.

recomendar.asp

```
<!-- formulario de recomendación -->
<% if request.querystring("accion") = "" then
%>
<form method="post" action="recomendar.asp?accion=enviar" name="recomienda">
<b>Recomienda este sitio</b><br><br>
Tu Nombre: <input type="text" name="n_remitente" size="10"><br>
Tu E-mail: <input type="text" name="e_remitente" size="20"><br>
Nombre de tu amigo: <input type="text" name="n_destinatario" size="10"><br>
E-mail de tu amigo: <input type="text" name="e_destinatario" size="20"><br><br>
<input type="submit" value="Recomendar">
</form>

<!-- envío del formulario y acuse de envío o información de errores -->
<%
elseif request.querystring("accion") = "enviar" then

' recojo las variables que vienen desde el formulario
n_destinatario = request.form("n_destinatario")
e_destinatario = request.form("e_destinatario")
n_remitente = request.form("n_remitente")
e_remitente = request.form("e_remitente")

    ' si los campos no están vacíos
    if n_destinatario <> "" and e_destinatario <> "" and n_remitente <> "" and e_remitente <> "" then

    ' indica la url de tu sitio
    url = "http://www.tusitio.com"
    ' indica el nombre de tu sitio
    nombre_del_sitio = "Tu Sitio"
    ' indica el asunto del mensaje
```

```
    asunto = n_remitente & " te recomienda un sitio"

    ' redacta el mensaje
    mensaje = "Hola " & n_destinatario & ":<br>"
    mensaje = mensaje & n_remitente & " te recomienda que visites <b>" & nombre_del_sitio & "</b>.<br>"
    mensaje = mensaje & "Puedes verlo en <a href='" & url & "'>" & url & "</a><br><br>Saludos!"

    ' comienza envío
    Set envio = Server.CreateObject ("CDONTS.NewMail")

    ' indica que el e-mail es en formato HTML
    envio.BodyFormat = 0
    envio.MailFormat = 0

    ' envía el mensaje
    envio.Send e_remitente, e_destinatario, asunto, mensaje
    Set envio = Nothing

    ' Informa al usuario que se ha enviado el mensaje
    reponse.write "<b>El mensaje ha sido enviado</b>.<br>Gracias por recomendarnos!"

    ' si existen campos vacíos, envía un mensaje de error
    else
    response.write "Por favor, es necesario que completes todos los campos.<br>"
    response.write "<a href='recomendar.asp'>Pincha aquí</a> para corregir los campos."
    end if

end if
%>
```

Referencia: Disponemos de otro artículo en DesarrolloWeb.com que realiza un script con básicamente la misma funcionalidad [http://www.desarrolloweb.com/articulos/999.php], por si interesa ver otro punto de vista.

Gestión de descarga de archivos

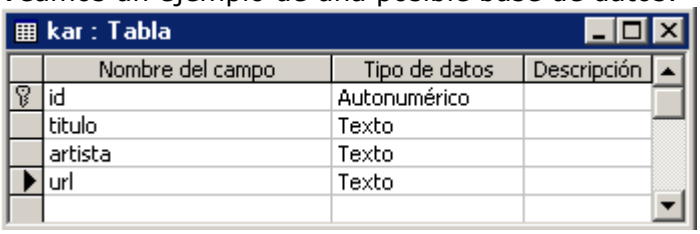
Existen servidores de hosting como *Geocities, Fortunecity*, entre otros, que no permiten la descarga externa de archivos. Es decir que si un usuario intenta descargar un archivo hospedado en *Geocities* cuyo enlace se encuentra en *i(España)*, por ejemplo, no podrá hacerlo. Pero ¿Qué sucede si nuestro servidor no restringe esta posibilidad? Y sí. Obvio. Podríamos sugerirle que lo hiciera. Pero mientras tanto ... ¿Qué podemos hacer?

Existe una solución aproximada si trabajamos con ASP, cookies e incluso puede mejorarse si a esto le agregamos el uso de JavaScript.
Hay que tener en cuenta, que esta "solución aproximada" no es del todo segura. Si nos encontramos con un WebRobaMaster medianamente astuto (y sino lo es, con ganas de trabajar demasiado), podría llegar a "burlar" dicha seguridad (aunque no es del todo frecuente).

El secreto está en ***crear una cookie en el ordenador del usuario*** *y* ***leerla para la descarga del archivo****.*

Vamos a explicar esto un poco mejor y con ejemplos.
Lo primero que debemos hacer, es crear una base de datos con nuestros archivos. Además que esto nos ayuda a tener un sitio más profesional y de fácil actualización, nos ayudará para proteger nuestros archivos.
Esta base de datos, deberá contener la url exacta de cada archivo.

Veamos un ejemplo de una posible base de datos:


kar : Tabla

	Nombre del campo	Tipo de datos	Descripción
🔑	id	Autonumérico	
	titulo	Texto	
	artista	Texto	
▶	url	Texto	

En este ejemplo, tenemos una tabla a la que llamamos ***kar*** que incluirá la base de datos de todos los archivos *.kar (karaoke) para descargar en nuestro sitio.
Los campos:
id > *campo autonumérico: genera u número único para cada archivo*
titulo > *será el título de la canción*
artista *> el artista o compositor de la obra*
url *> la ruta completa del archivo*

Imaginamos entonces que esta tabla tiene X cantidad de registros. Supongamos que dos de ellos contengan los siguientes datos:

id 12
titulo 50 martillazos
artista Claudia Puyó
url KAR74/c_puyo/50_martillazos.kar

id 13
titulo Desert Song
artista Def Leppard
url KAR74/def_leppard/desert_song.kar

Pues bien. Lo primero que debemos hacer es crear una página ASP en la que se muestre el listado de los archivos. Hasta aquí, todo debe hacerse normalmente. Lo que nos interesa a nosotros es *la URL que indicaremos para la descarga*.
Supongamos que la página donde se muestra el listado de archivos se llama ***muestra_archivos.asp***
En ella, debemos hacer que como url de descarga, figure otra página ASP, que será la que verifique las cookies que crearemos en ***muestra_archivos.asp*** y redireccione automáticamente al archivo. Esta página que comprobará dichos valores, la llamaremos ***comprueba.asp****.*
Entonces, volviendo a ***muestra_archivos.asp*** colocaremos el enlace de descarga de "50 martillazos de Claudia Puyó", de esta forma:

<a href="comprueba.asp?id=**12**">
donde **12** será el ID correspondiente al archivo. Por ejemplo **<%=RS("id")%>**

Nos quedará algo como esto:
50 Martillazos (Claudia Puyó) Bajar [http://www.desarrolloweb.com/articulos/comprueba.asp?id=12]
Desert Song (Def Leppard) Bajar [http://www.desarrolloweb.com/articulos/comprueba.asp?id=13]

Bien. El segundo paso entonces, será crear una cookie en la página ***muestra_archivos.asp***.
Una forma sencilla de hacerlo (aunque como dije al principio, puede descubrirse con un poco de astucia), puede ser esta:
Creamos una cookie con un valor absoluto:

```
<% response.cookies("Nombre_de_la_cookie")="valor de la cookie" %>
<% response.cookies("Nombre_de_la_cookie").Expires=Date+1 %>
```

Donde Nombre_de_la_cookie será **ArchKar** y valor **autorizado**. El código entonces quedará así:

```
<% response.cookies("ArchKar")="autorizado" %>
<% response.cookies("ArchKar").Expires=Date+1 %>
```

Nótese que este código debe ir ***al comienzo de la página muestra_archivos.asp***.
Esto, provocará que si el usuario ingresa a nuestra página ***muestra_archivos.asp*** se le creará esta cookie. Pero, si jamás ingresa, obviamente, la cookie no se creará.

Pasemos entonces a la página de descarga del archivo, ***comprueba.asp***.
Esta página tendrá como primer función, comprobar si la cookie **ArchKar** existe y si su valor es **autorizado**.

```
<% ArchKar = request.cookies("ArchKar")
if ArchKar = "autorizado" then
ArchKar = true
end if
%>
```

Este código nos está diciendo que si **ArchKar** tiene como valor **autorizado**, **ArchKar** será igual a **true** (verdadero).

El siguiente paso, será establecer que sucede si **ArchKar** es distinto a **true**. Es decir, si **ArchKar** no tiene valor **"autorizado"** o su valor es *nulo* o directamente *no existe*.

```
<%
if ArchKar <> true then
response.redirect "error.htm"
%>
```

Si **ArchKar** es distinto que **true**, enviar al usuario a la página **error.htm** (esta página puede contener un mensaje del tipo "El archivo no puede ser descargado desde un servidor externo").

Ahora solo bastará indicar que hacer si el valor de **ArchKar** es igual a **true**. Lo que haremos, será descargar el archivo automáticamente.

```
<%
else
idkar = request.querystring("id")

'abrimos la base de datos y creamos el recordset y hacemos la consulta
sqltxt="select * from kar where id="&idkar
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" & Server.MapPath("archivos.mdb"))
set rs=createobject("ADODB.Recordset")
rs.open sqltxt,conn
url = rs("url")
response.redirect url
rs.close
end if
%>
```

Entonces, si **ArchKar** ***no es distinto a true*** (es decir, que es igual a true), redireccionamos al usuario hacia el archivo en cuestión.

Síntesis del proyecto:

1. **Base de datos**
   1. Tabla: **kar**
   2. Campos requeridos
      - **id** autonumérico (clave única)
      - **url** texto > ruta completa de la descarga del archivo
   3. Campos adicionales: serán los referentes a las distintas características del archivo y variarán según las necesidades de cada caso.
2. **mostrar_archivos.asp**
   Página en la cual se muestra el listado de archivos y se crean las cookies

1. *Código Creación de cookies*
```
<% response.cookies("ArchKar")="autorizado" %>
<% response.cookies("ArchKar").Expires=Date+1 %>
```
2. *Enlace de descarga*
```
<a href="comprueba.asp?id=<%=rs("id")%>">
```

3. **comprueba.asp**
```
<%
ArchKar = request.cookies("ArchKar")
if ArchKar = "autorizado" then
ArchKar = true
end if
if ArchKar <> true then
response.redirect "error.htm"
else
idkar = request.querystring("id")
sqltxt="select * from kar where id="&idkar
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" & Server.MapPath("archivos.mdb"))
set rs=createobject("ADODB.Recordset")
rs.open sqltxt,conn
url = rs("url")
response.redirect url
rs.close
end if
%>
```

Consejos finales:

Mucho puede mejorarse esta rutina. Desde buscar asignar a la cookie valores difíciles de entender ("autorizado" es facil de entender para quien examine la cookie en su ordenador, pero si tal le asignamos un valor tipo "gafdsrs2454dfhgyr", no le dirá mucho a quien lo lea y seguramente no sepa de que se trate). Aún mas puede mejorarse, si el nombre de la cookie, estuviese destinado a distraer la atención, por ejemplo, si la cookie se llamara en ves de ArchKar, *publicidad* ¿Quien podría imaginarse el verdadero objetivo para el cual fue creada? Y si aún, queremos evitar que un WebRobaMaster, se tome el trabajo de "hormiga" de parar el explorador cada vez que desde nuestro sitio, sea redireccionando al archivo, para poder copiar (en caso de descargas lentas) la dirección url verdadera del archivo, lo que podemos hacer es agregar un script al primer enlace de descarga, indicando que éste se abra en una ventana sin la barra de direcciones y sin la barra de estado.
Son muchas las posibilidades de mejorar la seguridad de esta rutina y sobre todo la de nuestro sitio.

Por experiencia propia, puedo asegurar que más de 10.000 archivos de mi sitio, quedaron protegidos de descargas externas, gracias a este método.

Uso de cookies

¿Qué son y para qué sirven las *cookies*?
Las *cookies* son pequeños archivos de texto que se guardan en el ordenador del cliente y almacenan información referente a éste. Esta información puede ser utilizada para generar distintos tipos de configuraciones y opciones que el usuario elija. Así como también, con un poco de imaginación, pueden utilizarse para infinidad de funciones que veremos más adelante.

¿Qué podemos hacer con las *cookies*?
Básicamente las *cookies* pueden **escribirse en el ordenador del cliente** con **response.cookies** y **leerlas** con **request.cookies**.

Sintaxis de escritura de una *cookies*

- **Escritura de una *cookie***

<% response.cookies("nombre_de_la_cookie")="valor_de_la_cookie" %>

- **Lectura de una *cookie***

<% = request.cookies("nombre_de_la_cookie") %>

Las *cookies* deben tener una fecha de caducidad. De lo contrario se borrarían automáticamente luego de ser creadas. La sentencia que indica la **caducidad de una *cookie*** es la siguiente:
Fecha exacta de caducidad
<% response.cookies("nombre_de_la_cookie").Expires=#August 25, 2003# %>
Esta *cookie* indica que finalizará el 25 de agosto de 2003.

Cantidad de días
<% response.cookies("nombre_de_la_cookie").Expires=Date+365 %>
Esta *cookie* indica que finalizará el dentro de 365 días.
Ejemplos de aplicación de *cookies*
Con este ejemplo vamos a crear una *cookie* en la cual recogeremos los datos de personalización que el usuario elija para ver una página. En este caso, el usuario deberá elegir el *color de fondo* y el *color de fuente* que desea ver cuando ingresa a nuestro sitio sitio.

Para ello crearemos un pequeño formulario en el cual daremos al usuario la opción de elegir dichos colores:
Este formulario lo guardaremos como ***form.htm***

```
<form method="post" action="cookies.asp">
<p align="center"><b>
Selecciona los colores que deseas ver:</b></p>
<p style="margin-top: 0; margin-bottom: 0" align="center">
Fondo de la página: <select size="1" name="fondo">
<option value="white">Blanco</option>
<option value="black">Negro</option>
<option value="blue">Azul</option>
<option value="red">Rojo</option>
<option value="green">Verde</option>
</select></p>
<p style="margin-top: 0; margin-bottom: 0" align="center">Color de
fuente: 
<select size="1" name="fuente">
<option value="white">Blanco</option>
<option value="black" selected>Negro</option>
<option value="blue">Azul</option>
<option value="red">Rojo</option>
<option value="green">Verde</option>
</select> </p>
<p style="margin-top: 0; margin-bottom: 0" align="center"><input type="submit"
value="Enviar"></p></form>
```

Luego crearemos la página de proceso donde se crearán las cookies y se mostrarán los resultados. La llamaremos ***cookies.asp***.

```
<%
'recogemos los datos del formulario
fondo = request.form("fondo")
fuente = request.form("fuente")

'creamos las cookies
response.cookies("ColorFondo")=fondo
response.cookies("ColorFuente")=fuente

'creamos la caducidad de las cookies
response.cookies("ColorFondo").Expires=Date+7
```

```
response.cookies("ColorFuente").Expires=Date+7

'ahora, mostramos el resultado obtenido escribiendo las cookies
%>
<body
text="<%=request.cookies("ColorFuente")%>" bgcolor="<%=request.cookies
("ColorFondo")%>">
<center><font size="6">Así se verán los colores</font></center>
```

Ver el resultado de la ejecución del código.
[http://www.guiartemultimedia.com/desarrolloweb/tallerasp/cookies/cookies_color.asp]

Galería de imágenes en ASP

Vamos a ver cómo realizar rápidamente un sistema de almacenamiento y visualización de imágenes dinámico en ASP, en el que los visitantes puedan ver fotografías clasificadas en distintos apartados. Al acceder a una de las clasificaciones de imágenes se mostrarán una serie de miniaturas de las fotos que hay disponibles y, pulsando cada miniatura, se verá la fotografía a tamaño completo.

Obviamente, este no es un sistema que se pueda desarrollar desde cero en unas pocas horas, más bien necesitaríamos un esfuerzo de varios días. Sin embargo, vamos a atajar el problema de una manera mucho más rápida, haciendo uso de un software gratuito que permite instalar fácilmente una galería fotográfica en nuestro servidor ASP.

Podemos ver un ejemplo de este programa en funcionamiento en
http://www.iloire.com/demo/online_photo_catalog_vbscript/
online_photo_catalog_vbscript.asp

Casi para cualquier tecnología de desarrollo, existen a disposición de los desarrolladores ciertos programas que podemos instalar en nuestro servidor. Estos programas nos pueden ayudar a simplificar algunas cosas o dotar a nuestras páginas de ciertas funcionalidades. ASP no es un lenguaje donde se ofrezcan muchos sistemas de manera gratuita -son más habituales los de pago-, pero en este caso vamos a ver un sencillo script que sí resulta gratuito y será muy útil

para clasificar nuestras fotos y publicarlas en la web.

Online Photo Catalogue

Este programa se puede conseguir de manera gratuita desde la página de inicio del producto.
http://www.iloire.com/vbscript/online_photo_catalog_vbscript.asp
Ha sido construido para simplificar al máximo la tarea de publicación de las imágenes, evitando el uso de componentes externos a ASP, bases de datos, sistemas de upload de ficheros, etc.

Para utilizarlo necesitamos un servidor que soporte la programación en ASP. Además, si el servidor también permite utilizar .NET, las funcionalidades disponibles mejorarán a la hora de mostrar las imágenes en miniatura.

Online Photo Catalogue utiliza el objeto del sistema de ficheros (FSO: File System Object) para conocer qué imágenes hay en un directorio y qué subdirectorios puede también contener. Con los datos que obtiene del sistema de ficheros, se encarga de construir un árbol de directorios, a los que se puede acceder para visualizar las fotos que hay en cada carpeta.

Por tanto, no es necesario dar de alta las imágenes en ninguna base de datos, simplemente se tienen que almacenar en una carpeta y a partir de ese momento estarán disponibles para el visitante.

Instalación

El proceso para poner en marcha este programa es extremadamente simple. Primero tendremos que descargar los archivos con el software, que tendremos que colocar en nuestro servidor.

Luego, tendremos que colocar las imágenes en la carpeta "IMAGES". Dentro de esta carpeta podemos crear otras subcarpetas para hacer un árbol de directorios a gusto del administrador. Podemos crear una carpeta para nuestros amigos y otra para nuestros viajes. Por ejemplo, dentro de la carpeta de viajes, también podremos crear otros subdirectorios con cada uno de los lugares a los que hemos ido. Si utilizamos un servidor remoto las imágenes las tendremos que subir por FTP.

A partir de ahora, podemos acceder a la foto-galería y ver en la parte superior izquierda el árbol de directorios creado para clasificar las imágenes. Pulsando en cada carpeta podremos ver las fotos, en miniatura, que hay en esa carpeta. Pulsando finalmente en una de las miniaturas, podremos ver la imagen a tamaño completo.

Para la generación dinámica de las miniaturas se utiliza un poco de programación en ASP.NET. Como no todos los alojamientos que permiten ASP también permiten .NET, se puede deshabilitar la generación de miniaturas a través de la edición de las constantes de configuración del programa. En concreto deberemos editar la variable cUseThumbnailFile y asignarle el valor false (true es el que viene por defecto para esta variable)

```
Const cUseThumbnailFile=false
```

También hay otras variables de configuración, pero se explican mejor en el documento de ayuda del software.

Como detalles adicionales, el administrador puede comentar las fotografías. Esto se hace mediante la creación de un fichero de texto llamado con el mismo nombre que la fotografía, pero con extensión txt. Por ejemplo, si tenemos una imagen llamada foto1.jpg y deseamos incluir un comentario en ella, tendremos que crear un archivo llamado foto1.txt y escribiremos dentro el texto que queramos. El archivo de texto tiene que guardarse en el mismo directorio que la fotografía.

También permite la inclusión de comentarios a las fotografías por parte de los visitantes. Para

guardar los comentarios se utiliza un archivo de texto en formato XML, por lo que no es necesario configurar nada. Tan sólo hay que estar seguros que los permisos de escritura estén habilitados para ese fichero.

El ejemplo de este programa en funcionamiento se puede ver en:
http://www.iloire.com/demo/online_photo_catalog_vbscript/online_photo_catalog_vbscript.asp

La página de inicio del producto se puede ver en:
http://www.iloire.com/vbscript/online_photo_catalog_vbscript.asp

Dot Net Nuke, sistema portal en ASP.NET

Mucha gente conocerá PHP Nuke, un sistema para crear un portal fácilmente en PHP y con base de datos MySQL. Vamos a presentar ahora un sistema para hacer un portal en ASP.NET un poco más desconocido, aunque la potencia y desarrollo del programa indican que es totalmente válido y muy interesante para lanzarse rápidamente al mundo editorial en la web.

DotNetNuke es un sistema para hacer páginas web de tipo portal. Incluye un parte pública, a la que acceden los visitantes -el propio portal- y una privada, a la que acceden los administradores de la página, para editar los contenidos de la parte pública.

Características del programa

Ante todo es importante hablar de la gratuidad del software. En contra de lo que muchas veces ocurre con los sistemas Microsoft, podemos utilizar libremente este programa para cualquier propósito. Podemos modificarlo, redistribuirlo o, incluso, venderlo o prestar soporte técnico. La única condición es que debemos mencionar la procedencia del software, en la página o el código HTML.

DotNetNuke tiene muchas posibilidades, algunas de ellas auténticamente interesantes. Entre ellas podemos encontrar:

- Creación de múltiples portales utilizando un mismo código y base de datos, lo que significa que se pueden crear varios sitios utilizando un mismo alojamiento que soporte ASP.NET.
- Registro de usuarios y completa personalización del portal dependiendo del tipo de usuario.
- Posibilidad de crear servicios o contenidos de acceso restringido, con enlace directo a métodos de pago para los usuarios que deseen acceder a esos servicios.
- Acceso de administrador para gestionar los contenidos y servicios. Acceso a estadísticas completas del uso del sitio.
- Sistema de gestión de banners integrado.
- Acceso de superusuario para crear nuevos portales.
- Posibilidad de crear skins (apariencias) por el administrador para dotar de un aspecto al portal totalmente personalizado.
- Permite trabajar con varias bases de datos distintas, aunque para algunas hace falta algún módulo adicional de soporte.
- Múltiples tipos de contenidos que se pueden administrar. Entre ellos Noticias, contacto, foros de discusión, documentos, eventos, etc. Una larga lista que se puede ampliar con una serie de módulos disponibles.

Módulos de DotNetNuke

Los módulos son piezas del programa que ofrecen soporte a funcionalidades específicas para los usuarios del portal. La versión actual de DotNetNuke dispone de 24 módulos para realizar tareas diversas como gestión de usuarios, calendario de eventos, banners, noticias, etc. Además, existen otros módulos gratuitos, ofrecidos por portales que tratan sobre DotNetNuke y

también algunos de pago que ofrecen ciertas empresas y desarrolladores.

Los módulos pueden instalarse, desinstalarse y configurarse siempre que se quiera, para dotar al portal de las funcionalidades que necesitamos en cada momento. Entre los módulos disponibles en la versión inicial se encuentran los siguientes:

- Gestión de cuentas
- Noticias o novedades
- Gestión de banners
- Contacto, que ofrece la posibilidad de enviar correo a personas o grupos de personas
- Foros de discusión, con temas configurables
- Gestión de documentos, con opción a descarga o visualización
- Calendario de eventos
- FAQs
- Links
- Búsquedas
- Encuestas

La lista todavía continúa con otros módulos para temas más específicos o tan especiales como servicio de información del tiempo o un servicio Whois, para hacer búsquedas de registro de nombres de dominios.

Algunos ejemplos de páginas realizadas con DotNetNuke

Existen ya bastantes ejemplos de buenos trabajos realizados con DotNetNuke. Entre los mejores hay algunos contribuidores de desarrollo de la plataforma.

4Birckhead http://www.4birckhead.com/ En esta página, aparte de un agradable diseño, podremos encontrar nuevos módulos gratuitos para extender DotNetNuke, como álbum de fotos o Tikers de índices bursátiles.

ByDesignWebsights http://www.bydesignwebsights.com/
Ofrecen soporte y recursos sobre DotNetNuke. En el momento en el que se escribió este artículo ofrecían alojamiento gratuito sobre la plataforma.

Ciber Huis http://www.cyberhuis.com/
Otro sitio de recursos con posibilidad de descargar nuevos módulos gratuitos.

DotNetNuke Skins http://www.dnnskins.com/ Un sitio para encontrar nuevas apariencias para DotNetNuke. Hay que registrarse para acceder a los skins, pero puede merecer la pena porque tienen disponibles decenas de diseños distintos.

DotNetNuked http://www.dotnetnuked.com/ Otro ejemplo de sitio realizado con DotNetNuke que ofrece también recursos de utilidad.

Snowcovered http://www.snowcovered.com/ Una página más comercial sobre DotNetNuke, en la que se ofrecen recursos útiles, aunque la mayoría de las veces de pago.

Cómo obtener DotNetNuke

En la página de inicio de DotNetNuke (http://dotnetnuke.com/) podemos encontrar mucha más información e instrucciones para instalar DotNetNuke.

Para empezar hay disponible una guía de instalación del software que explica paso a paso el proceso de instalación. También disponen de guías de usuario y de administrador del portal, así como diversa documentación y FAQs.

Para descargar DotNetNuke hay que registrarse en la página y hacer el inicio de sesión.

Podremos encontrar varios ficheros de descarga con distintas versiones del programa.

Conclusión

Dot Net Nuke es una buena muestra de lo que se puede hacer con .NET y del empeño de ciertas comunidades, junto con Microsoft, de popularizar el uso de software libre en esta nueva plataforma de desarrollo. Efectivamente, se trata de un software gratuito desarrollado por una comunidad de personas a que se sienten a gusto trabajando con sistemas Microsoft y que desean popularizar sus herramientas. El producto del trabajo lo podemos ver en dotnetnuke.com (http://dotnetnuke.com/) y en un primer vistazo parece que han conseguido una buena herramienta.

Utilizar funciones de VBScript con Jscript

Para la programación de páginas web del lado del servidor con ASP, los programadores disponen de dos lenguajes distintos de programación, que pueden elegir según las preferencias de cada uno.

Estos dos lenguajes para programar en ASP son VBScript (Visual Basic Script) y JScript (una versión de Microsoft del lenguaje Javascript). En principio, con los dos lenguajes se pueden hacer las mismas cosas y el programador puede utilizarlos indistintamente.

En este artículo vamos a ver cómo invocar funciones de VBScript desde el lenguaje Javascript.

Este artículo tiene sentido porque algunas funciones de VBscript son muy útiles, sobre todo para dar formato a fechas y valores, pero en JScript no existe su equivalente.

Una opción muy llamativa es utilizar las funciones de VBScript de la siguiente manera:

1) Crear script de VBScript con una función, que luego llamaremos de Javascript.

```
<SCRIPT LANGUAGE="VBSCRIPT" RUNAT="SERVER">
Function FormatValue(Value)
   FormatValue = FormatCurrency(Value)
End Function
</SCRIPT>
```

2)Invocar la función desde JScript

```
<%=FormatValue( (rTarifas.Fields.Item("valor1").Value) )%>
```

Espero que este ejemplo, a pesar de ser extremadamente sencillo, sirva de utilidad para mucha gente.

Escribir un fichero Excel desde ASP

Desde una página programada en ASP podemos obtener múltiples salidas, aparte de la tradicional en una página web. En este caso vamos a ver cómo realizar una salida en un formato de tipo CSV, que es un archivo compatible con Microsoft Excel u otros sistemas de hoja de cálculo comunes. El archivo CSV, para el que no lo sepa, es en realidad un fichero de texto en el que se ponen los valores de cada una de las columnas separados por comas y cada fila de valores en una línea independiente.

Veremos cómo conseguir un archivo CSV en el que figuran unos valores obtenidos desde una base de datos. Para ello vamos a apoyarnos en un script que se distribuye como código abierto en licencia GNU [http://www.fsf.org/copyleft/gpl.html]. Este tipo de licencia implica que el Script es gratuito y se puede distribuir, utilizar o modificar de cualquier manera, siempre que se cite la procedencia original.

En concreto, el script al que nos referimos se llama GenerateXSL, desarrollado por Brain Scan Studios, que se puede obtener en la dirección http://www.brainscanstudios.com/dev_projects/GenerateXLS/.

El script se puede instalar en cualquier servidor ASP, en un sitio web o en un servidor local o de Intranet, y se puede modificar fácilmente. Para utilizarlo en nuestros trabajos simplemente tendríamos que modificar unas variables. Para empezar, la variable "sDSN" que contiene el DSN origen de nuestra base de datos. También debemos modificar la variable "sFields", para asignarle los nombres de los campos de la base de datos, y "sTable", para indicar el nombre de la tabla.

El script, que podemos modificar para ajustarlo a nuestras necesidades, es el siguiente:

```
<%@ Language=VBScript %>
<%
' GenerateXLS Version 1.0 by Brian Kirsten (bkirsten@brainscanstudios.com)
' 1st modified 11/29/00
' 2nd modification 10/25/02
' copyright Ó 2000 Brain Scan Studios, Inc. (http://www.brainscanstudios.com)
' source distributed under the gnu general public license.
' let me know if your site is using the code i will put a link up to your page!

Dim sTable
Dim sDSN
Dim sFields

sDSN = "<DSN>" 'Name of your DSN
sFields = "<FIELDS>" 'List of fields comma delimited
sTable = "<TABLE_NAME>" 'Name of your table or View

Set DB = Server.CreateObject("ADODB.Connection")
Set RS = Server.CreateObject("ADODB.Recordset")

DB.Open sDSN

RS.Open "select "& sFields &" from "& sTable,DB

Response.ContentType = "application/csv"
Response.AddHeader "Content-Disposition", "filename=mydata.csv;"
' lets print the fields on top

for i = 0 to RS.Fields.Count-1
if i = (RS.Fields.Count - 1) then
Response.Write lcase(RS.Fields(i).Name)
else
Response.Write lcase(RS.Fields(i).Name) & ","
end if
next

Response.write vbNewLine
Response.write vbNewLine

while not RS.EOF

for u=0 to RS.Fields.Count - 1
if u = (RS.Fields.Count - 1) then
Response.Write RS.Fields(u).Value
else
Response.Write RS.Fields(u).Value & ","
end if
next

response.write vbNewLine

rs.MoveNext
wend

Response.write vbNewLine
Response.write vbNewLine
```

```
Set RS = Nothing
Set DB = Nothing

%>
```

Podemos acceder a la página de los autores del script [http://www.brainscanstudios.com/dev_projects/GenerateXLS/], donde podemos encontrar también un enlace para ver el ejemplo en funcionamiento.

DLL para mostrar fechas completas en ASP

En este artículo vamos a presentar una DLL que te permite mostrar la fecha en formato completo en una pagina ASP. Para ello se toma como entrada la fecha en formato corto, algo como "29/01/2005". Incluimos un ejemplo del funcionamiento de la librería.

Esta DLL no es gran cosa pero sucedió a raiz de una página que estaba haciendo, en la que necesitaba escribir la fecha completa, ej: Lunes 28 de febrero de 2005. Estuve buscando como hacerlo un largo rato en Internet y no encontré nada, ni siquiera alguna forma utilizando Javascript. Entonces, rápidamente me puse a diseñar esta DLL, que hace justo lo que necesitaba. Quizás a alguien en mi situación pueda servirle.

Podemos encontrar en este enlace el archivo DLL para descarga [http://www.desarrolloweb.com/articulos/ejemplos/dll-fecha/LongDateMV.zip]. No es más que una clase, que sirve para poder mostrar la fecha completa en una pagina ASP.

Se le proporciona la fecha deseada en el parámetro:

```
ldate.fecha "17/11/2005" 'La fecha que quieres
```

Como resultado, se almacenará el valor (la fecha con formato completo) en una variable de session llamada LaFecha, que nos servirá para poder recuperar el resultado en el momento que sea necesario.

Es importante señalar que, antes de utilizar la DLL es necesario registrarla en nuestro sistema. Para ello, desde el símbolo del sistema, ejecuta el comando

```
regsvr32 C:\carpeta\LongDateMV.dll
```

Ejemplo de funcionamiento

Veamos un pequeño código de página ASP donde se hace uso de la DLL.

```
<body>
<%
Dim ldate

Set ldate = Server.CreateObject("LongDateMV.ldate")

ldate.fecha "17/11/2005"

response.write session("LaFecha") & "<BR>"
ldate.about

Set ldate = Nothing

%>
</body>
</html>
```

Para comentarios o consultas cimersa@hotmail.com [mailto:cimersa@hotmail.com]

> **Nota:** Cualquier persona que lo desee, puede distribuir esta DLL libremente a traves de la Red.

Tratar errores en sentencias SQL ejecutadas en ASP

Vamos a ver un pequeño taller que puede servir para hacer nuestras aplicaciones ASP más completas y compactas. Se trata de evitar un error bastante frecuente en la programación de páginas, que ocurre cuando se intenta ejecutar una sentencia SQL mal formada. En esos casos, el servidor nos informa del error, aunque siempre ofrece información que no sirve de mucho al visitante, incluso despista. Sería preferible que nosotros tratásemos el error e informáramos por nuestra cuenta al usuario, de una manera agradable y clara.

Para ello vamos a realizar un tratamiento de errores en ASP, que evitará que la página se detenga ante un error y nos permita hacer cosas si ocurre.

```
On error Resume Next
conn.execute(ssql)
if Err<>0 then
' realizo acciones para tratar el error
end if
```

Con la primera línea estamos diciendo a ASP que si ocurre un error no informe de ello, sino que continúe. En la segunda línea ejecutamos una sentencia SQL.

En el siguiente bloque se evalúa la variable de sistema "Err", que almacena un posible error. Si la variable vale cualquier cosa distinto de 0, entonces es que ocurrió un error. Por tanto, en el caso positivo del if podremos realizar cualquier tipo de acción para tratar el error.

Tratar el error de una manera sencilla

Para tratar el error de una manera fácil de implementar, dentro del if podríamos llamar a una función que se encargase de hacer todas las acciones frente a un error.

Esa función podría recibir el error e introducirlo en un log de errores. Podría también informar por correo al administrador y por supuesto, mostrar un mensaje de error claro al visitante.

```
sub tratar_error_ssql(ssql,mierror)

'trato el posible error
'meto la sentencia erronea en un archivo de texto

'creamos el textstream del archivo
archivo= request.serverVariables("APPL_PHYSICAL_PATH") & "errores\log.txt"
set confile = createObject("scripting.filesystemobject")
set fich = confile.OpenTextFile (archivo,8)
'escribo en el archivo
fich.WriteLine(ssql)
fich.WriteLine(Err.Description)
fich.WriteLine("----------------------------")

'cerramos el fichero
fich.close()

'voy a informar al administrador
set obj_mail = server.createObject("Persits.MailSender")
asunto_mensaje = "Error SQL en la página"
email_origen_mensaje = "correo@dominio.com"
texto_origen_mensaje = "Empresa"
txt_mail = "Hemos detectado un error. Consulte el log de errores para encontrar descripción."
txt_mail = txt_mail & VBNEWLINE & VBNEWLINE & ssql
txt_mail = txt_mail & VBNEWLINE & VBNEWLINE & mierror

obj_mail.host = "smtp.dominio.com"
obj_mail.from = email_origen_mensaje
obj_mail.FromName = texto_origen_mensaje
obj_mail.Subject = asunto_mensaje
```

```
obj_mail.AddAddress email_alertas
obj_mail.body = txt_mail
'lo envio
obj_mail.send

'voy a informar al usuario
response.write "Lo sentimos, pero tu acción no ha podido ser realizada. Ponte en contacto con los administradores para obtener ayuda."

end sub
```

Esta función hace todo lo comentado para tratar el error. Primero escribe el log de errores en un fichero de texto, luego envía un correo electrónico al administrador y finaliza mostrando un error al usuario. Está comentada para entenderla mejor y todo lo que hemos realizado (como enviar el email desde ASP o abrir el fichero de texto) lo hemos visto ya en otros talleres de ASP.

Formularios reentrantes en ASP

Uno de los trabajos más habituales en la programación de páginas dinámicas del servidor consiste en tratar formularios, para insertar o actualizar informaciones en una base de datos. En este artículo vamos a ver una manera sencilla de realizar, en ASP, las tareas de inserción de los datos de un formulario en una base de datos, concentrando todas las tareas en una misma página, por medio de formularios reentrantes.

Con el concepto de formulario reentrante, me refiero a una página que contiene un formulario, cuyo submit está dirigido hacia la misma página. De modo que esa página sería encargada de mostrar el formulario, recibir los datos, comprobar su validez y, si todo fue correcto, insertar la información en la base de datos. Él resumen de acciones que debe realizar una página es el siguiente:

```
Compruebo si se reciben datos del formulario
1) Si no recibo datos del formulario, muestro el formulario vacío
2) En caso contrario, compruebo que los datos sean correctos
   2.1) Si no eran correctos, muestro el formulario con los datos que se han recibido.
   2.2) Si eran correctos, inserto los datos en la base de datos
```

Como se puede comprobar en el esquema anterior, el mismo formulario se puede mostrar en dos ocasiones: Si no se reciben datos por POST, o bien, si los datos recibidos por POST no son válidos. Para facilitar las tareas de mantenimiento de la página y crear un código más claro, vamos a hacer un único formulario, que incluiremos en cualquier punto que se necesite.

El esquema anterior, codificado en ASP, sería el siguiente. Nótese que se incluye el formulario en dos puntos distintos del código. Luego veremos el código del formulario.

```
<%
'comprobamos si se recibe del formulario
if request.form="" then
    'si no se recibe algo muestro el formulario
    %><!--#include file="formulario.asp"--><%
else
    'es que estamos recibiendo datos del formulario
    'creo una variable para concretar el error
    errores=""
    'valido todo para obtener lista de errores
    if len(request.form("nombre"))<2 then
       errores = errores & "<li>El nombre no es correcto"
    end if
    if len(request.form("telefono"))<9 then
       errores = errores & "<li>El teléfono no es correcto"
    end if

    'compruebo si hubieron errores
    if (errores<>"") then
       'hubo errores en la recogida de datos
```

```
      response.write "Errores en el formulario: <ul style='color:red'>" & errores & "</ul>"
      %><!--#include file="formulario.asp"--><%
   else
      'es que todos los datos eran correctos. inserto en la base de datos
      ssql = "insert into cliente (nombre, telefono) values ('" & request.form("nombre") & "', '" & request.form
("telefono") & "')"
      'ahora habría que conectar con la base de datos y ejecutar la sentencia
      'Estos temas se explican en otros artículos
   end if
end if
%>
```

El código está comentado para que se pueda entender más fácilmente. Como vemos se ha seguido el esquema comentado al principio del artículo. Las validaciones se realizan utilizando la función len() de VBScript, que devuelve el número de caracteres que contiene un string.

Código del formulario

Ahora nos centraremos en el formulario. Se trata de un código HTML simple, en el que se ha incluido una manera de memorizar el valor que pudieran tener los campos en anteriores envíos. Esto es porque, cuando se envía el formulario, si hubo errores en los datos, se debe mostrar el error y el formulario con los valores que tuviera anteriormente.

```
<FORM METHOD="post" ACTION="index.asp" NAME="form03">
<TABLE BORDER="0" CELLSPACING="2" CELLPADDING="2">
<TR>
   <TD>
   Nombre:
   </TD>
   <TD>
   <INPUT TYPE="text" NAME="nombre" VALUE="<%=request.form("nombre")%>">
   </DIV>
   </TD>
</TR>
<TR>
   <TD>
   Teléfono:
   </TD>
   <TD>
   <INPUT TYPE="text" NAME="telefono" VALUE="<%=request.form("telefono")%>">
   </DIV>
   </TD>
</TR>
<TR>
   <TD COLSPAN="2" align=right>
   <INPUT TYPE="submit" VALUE="Insertar">
   </TD>
</TR>
</TABLE>
</FORM>
```

Para recuperar esos valores, simplemente se define un value para la etiqueta <input>, en el que se coloca lo que haya en el array request.form("nombre_del_campo_a_recuperar").

Espero que estas indicaciones sean de interés, por lo menos para los usuarios menos experimentados.

Validar una fecha con ASP

Veamos una función creada con Active Server Pages para hacer una comprobación de fechas. La función recibe un día, un mes y un año y comprueba que estos sean numéricos y que la fecha creada entre todos es correcta y posible en el calendario.

La función recibe los datos de día mes y año como string, tal como podrían llegarnos a través de un formulario o como parámetro en una URL. No obstante, dado que ASP es un lenguaje

poco tipado, aunque pasemos valores numéricos a la función seguirá dando resultados correctos.

Como resultado la función devolverá un valor verdadero o falso, dependiendo si la fecha es correcta o incorrecta.

El código de la función es el siguiente:

```
'valido una fecha
'espero recibir tres strings con el dia, el mes y el año

function validarFecha(dia,mes,ano)

  'elimino posibles espacios a los lados de los números que recibo por parámetro

  dia = trim(dia)
  mes = trim(mes)
  ano = trim(ano)

  'compruebo nº de caracteres que recibo en cada parámetro son los permitidos
  'El año puede tener hasta 4 caracteres

  if len(dia)=0 or len(dia)>2 or len(mes)=0 or len(mes)>2 or len(ano)=0 or len(ano)>4 then
    validarFecha = false
    exit function
  end if

  'compruebo que los caracteres de los parámetros son números

  if (not isNumeric(dia)) or (not isNumeric(mes)) or (not isNumeric(ano)) then
    validarFecha = false
    exit function
  end if

  'El mes no puede ser mayor que 12 ni menor que 1

  mes = cint(mes)
  if mes > 12 or mes < 1 then
    validarFecha = false
    exit function
  end if
  dia = cint(dia)

  'El día no puede ser menor que 1

  if dia < 1 then
    validarFecha = false
    exit function
  end if

  'El dia, dependiendo del mes que sea, puede tener unos u otros valores

  if mes=1 or mes=3 or mes=5 or mes=7 or mes=8 or mes=10 or mes=12 then

    'en estos meses puede haber 31 dias

    if dia > 31 then
      validarFecha = false
      exit function
    end if
  elseif mes=2 then

    'en febrero tenemos que ver si es año bisiesto
    'consigo el numero de año de 4 cifras.
    'si nos dan un valor de 2 cifras < 31 se refiere a 2000 más ese valor

    if ano < 31 then
      ano = ano + 2000

    'si nos dan un valor de 2 cifras > 31 se refiere a 1900 más ese valor

    elseif ano < 100 then
```

```
    ano = ano + 1900
  end if

  'calculo si el año es bisiesto
  'si es divisible por cuatro y (no divisible por 100 o divisible por 400)

  if ((ano mod 4)=0) and ((ano mod 100)<>0 or (ano mod 400)=0) then

    'es bisiesto

    if dia > 29 then
      validarFecha = false
      exit function
    end if
  else

    'NO es bisiesto

    if dia > 28 then
      validarFecha = false
      exit function
    end if
  end if
 else

  'en todos los demás meses llegan a tener 30 dias

  if dia > 30 then
    validarFecha = false
    exit function
  end if
 end if

 'si estoy aquí es que todas las comprobaciones han sido positivas

 validarFecha = true
end function
```

El código de la función está comentado para que se pueda entender más fácilmente. Básicamente lo que hace es realizar una serie de comprobaciones, una tras otra. Si alguna de las comprobaciones hace que la fecha sea incorrecta, se devuelve el valor false y se sale de la función. Si se llega al final de la función y ninguna de las comprobaciones falló, se devuelve true, pues la fecha es correcta.

A la hora de calcular si un día es válido, tenemos que saber los días que tiene un mes. Hay que prestar especial atención al mes de febrero, para saber si es bisiesto. Para saber si un año es bisiesto existe la regla siguiente:

- Son bisiestos todos los años divisibles por 4, excluyendo los que sean divisibles por 100, pero no los que sean divisibles por 400.

Un Chat en ASP

El código que os muestro es el utilizado en el chat de esta web, el introducir mejoras ya es cosa vuestra, es fácil añadir funcionalidades al estilo de las que veréis en otros chat en Java, como colores, lista de usuarios, etc ...

Bueno, empezamos por el principio.

1.-Lo primero que necesitamos es un lugar para almacenar las frases que los usuarios introduzcan para luego mostrarlas. Para ello usaremos un array de nueve elementos, que seran la cantidad de frases que mostraremos en el chat (podéis cambiar el numero a lo que os parezca oportuno). Este array lo guardaremos dentro del objeto Application, a fin de que pueda

ser accedido por todo el mundo. A fin de que este disponible siempre, la crearemos dentro del archivo global.asa.

Global.asa

```
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">
Sub Application_OnStart
   Dim Auxiliar()
   Redim Auxiliar(9)
   Application("Opiniones")=Auxiliar
End Sub
</SCRIPT>
```

A partir de este momento tenemos una variable de aplicación denominada "opiniones" que contiene nuestro chat

2.-Lo siguiente que necesitamos es una página que nos muestre el contenido de nuestro chat (es decir, lo que hay en application("opiniones"). A este modulo lo denominaremos visualización.asp. Consiste en una pagina que se llama a si misma cada x segundos (META HTTP-EQUIV="REFRESH" CONTENT="5), con lo que nos refresca la información, y que muestra los datos contenidos en Application("opiniones") mediante un sencillo bucle FOR

Visualizacion.asp

```
<% PaginaActual="http://"&_
Request.ServerVariables("SERVER_NAME")&_
Request.ServerVariables("SCRIPT_NAME") %>
<html>

<head>
<META HTTP-EQUIV="REFRESH" CONTENT="5;<%=PaginaActual%>">
<title>MiniChat (visualización)</title>
</head>

<body>
<FONT FACE="Comic Sans MS" COLOR="Blue" size="1">
<%
IF NOT isArray( Application("Opiniones")) THEN
Application.Lock
Dim Auxiliar()
Redim Auxiliar(9)
Application("Opiniones")=Auxiliar
Application.UnLock
END IF

Temporal=Application("Opiniones")
FOR Opinion=8 to 0 step -1%>
<%= Temporal(Opinion) %> <BR>
<% NEXT %>
<FONT>
</body>

</html>
```

3.-Nuestro tercer modulo sera "incluir.asp" el cual nos va a permitir que los usuarios escriban opiniones en el chat. Para que un usuario escriba una aportación, este usuario debe estar identificado por un apodo o "nick", este apodo lo guardaremos en una cookie.

incluir.asp

```
<%
IF Request.Cookies("Apodo")="" and request.form("opinion")<>"" THEN
if request.form("apodo")<>"" then
Response.Cookies("Apodo")=Request.Form("Apodo")
else
Response.Cookies("Apodo")="Anonimo"
end if
Application.Lock
```

```
Temporal=Application("Opiniones")
FOR i=7 TO 0 STEP -1
Temporal(i+1)=Temporal(i)
NEXT
if request.form("apodo")<>"" then
Temporal(0)="<FONT COLOR=""#000000"">** " & Request.Form("Apodo") & " ** Entra en el minichat</FONT>"
else
Temporal(0)="<FONT COLOR=""#000000"">** Anonimo ** Entra en el minichat</FONT>"
end if
Application("Opiniones")=Temporal
Application.Unlock

END IF
IF Request.Form("Opinion")<>"" THEN
Apodo=Request.Cookies("Apodo")
Application.Lock
Temporal=Application("Opiniones")
FOR i=7 TO 0 STEP -1
Temporal(i+1)=Temporal(i)
NEXT
Temporal(0)=Apodo&": "&Request.Form("Opinion")
Application("Opiniones")=Temporal
Application.Unlock
END IF%>

<html>

<head>
<title>incluir opinion</title>
<base target="_self">
</head>

<body bgcolor="#6699FF">
<FORM METHOD="POST" ACTION="incluir.asp">
<% IF Request.Cookies("Apodo")="" THEN %>
<font color="#FFFFFF">
Apodo:</font> <INPUT TYPE="TEXT" SIZE=10 NAME="Apodo">
<input type="hidden" name="go" size="20" value="si"><BR>
<% END IF %>
<INPUT TYPE="TEXT" SIZE=30 NAME="Opinion">
<INPUT TYPE="SUBMIT" VALUE="Enviar">
<a href="cerrar.asp" target="_top">Salir</a>
</FORM>
</body>

</html>
```

4.- Cuando el usuario termine su sesión de chat debemos eliminar la cookie que lo identifica, para ello usaremos el modulo cerrar.asp

cerrar.asp

```
<% if Request.cookies("Apodo")<>"" then
Application.Lock
Temporal=Application("Opiniones")
FOR i=7 TO 0 STEP -1
Temporal(i+1)=Temporal(i)
NEXT
Temporal(0)="<FONT COLOR=""#FF0000"">** " &Request.cookies("Apodo")&" ** Se va del minichat</FONT>"
Application("Opiniones")=Temporal
Application.Unlock
response.cookies("apodo")=""

END IF%>
<HTML>
<HEAD>
<script language="JavaScript">
{close();}
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>
```

5.- Ahora lo montamos todo en una pagina de frames

default.asp

```
<html>

<head>
<title>Salon de conversación</title>
</head>

<FRAMESET rows="*,70">
<FRAME SRC="visualizacion.asp">
<FRAME SRC="incluir.asp" target="_self">
</FRAMESET>

</html>
```

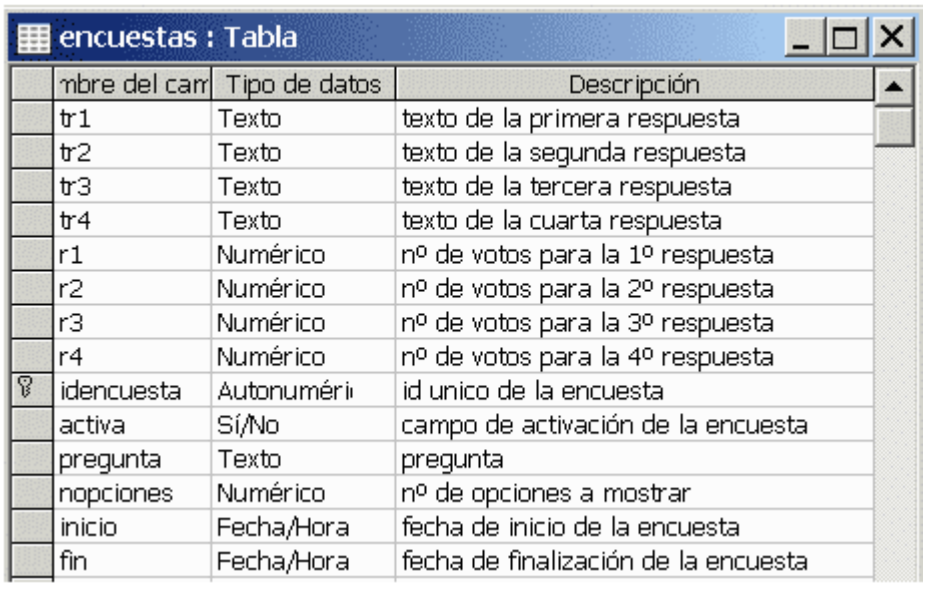
6.- Para que quede "bonito" mostraremos el chat en una nueva ventana sin barras y mas pequeña, eso lo haremos con javaScript

```
<HTML>
<HEAD>
<script language="JavaScript">
<!--
function openWindow(url, name) {
popupWin = window.open(url, name, 'scrollbars,resizable,width=400,height=350')
}
// -->
</script>
</HEAD>
<BODY BGCOLOR="#FFFFFF" TEXT="#000000" LINK="#000000" VLINK="#000000">
<a HREF="javascript:openWindow('default.asp', 'minichat');">Enter chatroom</a>
</BODY>
</HTML>
```

Sistema de encuestas con ASP

Lo basamos en una bd que contendrá la pregunta y las posibles respuestas a estas, esto nos evitar tocar el código html cada vez que variemos la encuesta. Mantendremos un registro por cada encuesta y un campo en dicho registro indicara cual es la encuesta activa en ese momento.

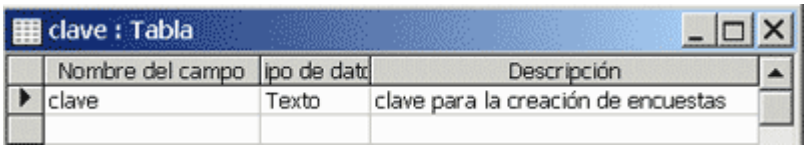
La estructura de la tabla encuestas en la BD Tencuestas es :



Nuestra aplicación consta de 2 módulos principales(encuesta.asp y verencuesta.asp) mas un modulo de visualización de todas las encuestas (historico.asp).

Además he incluido el código de 2 módulos de mantenimiento, que nos servirán para incluir nuevas encuestas desde la web (crearencuesta.htm y crearencuesta.asp). El uso de estos módulos implica la creación en la bd de una nueva tabla para el almacenamiento de la clave de creación de encuestas.

Estructura de la tabla clave en la BD Tencuestas



Para mostrar la encuesta en una página lo haremos con un include

```
#include file="encuesta.asp"
```

Lo que nos permite mostrar la encuesta en cualquier página con solo una línea de código. el aspecto seria el que podeis ver en la página principal de www.asptutor.com

Módulos básicos del sistema

encuesta.asp

```
<%Set Connae = Server.CreateObject("ADODB.Connection")
Connae.Open("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & Server.MapPath("/xxxxx/Tencuestas.mdb"))

set rse=createobject("ADODB.Recordset")
sqltxt="SELECT * from encuestas where activa=true"
rse.open sqltxt,connae
if not rse.EOF then
%>

<div align="center">
<center>
<table border="1" width="150" bgcolor="#C0C0C0" bordercolor="#0000FF">
<tr>
<td width="100%">
<form method="POST" action="encuestas/verencuesta.asp" target="_blank">
<table border="0" width="100%">
<tr>
<td width="100%" bgcolor="#0000FF">
<p align="center"><font face="Verdana" size="1" color="#00FFFF"><b><i>Encuesta
del mes</i></b></font></td>
</tr>
<tr>
<td width="100%">
<p align="center"><font face="Verdana" size="1"><b><%=rse("pregunta")%></b></font></td>
;</tr>
<%for i = 0 to rse("nopciones")-1 step 1 %>
<tr>
td width="100%"><input type="radio" value="<%=i%>" name="opcion" checked>
<font face="Verdana" size="1"><%=rse(i)%></font></td>
</tr>
<%next
rse.close%>
</table>
<p align="center">
<input border="0" src="../images/opinar.gif" name="I1" type="image"></p>
  </form>
</center>
<p align="center"><a href="encuestas/historico.asp" target="_blank"><img border="0" src="images/ir.gif"
alt="Historico de encuestas"></a></p>
</td>
```

```
</tr>
</table>
</div>
<%else
rse.close
end if
connae.close
set connae=nothing
%>
```

verencuesta.asp

```
<% @LCID = 1034 %>
<%response.expires=-1000%>

<%
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & Server.MapPath("/xxxxx/Tencuestas.mdb"))

if request.form("opcion")<>"" then

select case true

case request.form("opcion")=0
    campo="r1"
case request.form("opcion")=1
  campo="r2"
case request.form("opcion")=2
  campo="r3"
case request.form("opcion")=3
  campo="r4"
end select

sqltxt="update encuestas set "&campo&" = "&campo&"+1 where activa=True"

conn.execute sqltxt
end if
%>

<html>

<head>

<title>Las encuestas de AspTutor</title>
</head>

<body>
<%if request.form("opcion")<>"" then%>
<p align="center"><font color="#FF0000"><b><font face="Verdana" size="1" color="#FF0000">Muchas gracias por
participar en la
encuesta</font><br>
</b></font><br>

<%end if
dim colores(3)
colores(0)="red"
colores(1)="blue"
colores(2)="green"
colores(3)="black"
tamatabla=400 'tamaño de la tabla

set rs=createobject("ADODB.Recordset")
sqltxt="SELECT * from encuestas where activa=true"
rs.open sqltxt,conn
if not rs.EOF then
%>

<div align="center">
<center>
<table border="1" width="<%=tamatabla%>" bgcolor="#C0C0C0" bordercolor="#0000FF">
<tr>
<td width="<%=tamatabla%>" bgcolor="#0000FF">
<p align="center"><font color="#FFFFFF" face="Verdana" size="1"><b><%=rs("pregunta")%></b></font></td>
</tr>
```

```
<tr>
<td width="<%=tamatabla%>">

<table border="0" width="<%=tamatabla%>">
<%totvotos=rs("r1")+rs("r2")+rs("r3")+rs("r4")
if totvotos< 1 then totvotos=1
for i = 0 to rs("nopciones")-1 step 1 %>
<%porcentaje=formatnumber(rs(i+4)*100/totvotos,2)%>
<tr>
<td width="<%=tamatabla*30/100%>" valign="top"><font face="Verdana" size="1"><b><%=rs(i)%
></b></font><br>
<font face="Verdana" color="#FFFFFF" size="1"><b><%=porcentaje%>%</b></font></td>

<td width="<%=tamatabla*70/100%>">

<table border="0" width="<%=round(porcentaje)%>%" bgcolor="<%=colores(i)%>">
<tr>
<td width="100%">
</td>
</tr>
</table>
</td>
</tr>
<%next

%>
</table>

</td>
</tr>
<tr>
<td width="300">
<p align="center"><font face="Verdana" size="1"><b>Total de votos: <font color="#FF0000"><%=totvotos%
></font> </b></font></p>
</td>
</tr>
<%if rs("inicio")<>"" then%>

<tr>
<td width="300">
<font face="Verdana" size="1">Fecha de inicio: <%=rs("inicio")%></font>
</td>
</tr><%end if%>

<%if rs("fin")<>"" then%>
<tr>
<td width="300">
<font face="Verdana" size="1">Fecha de cierre: <%=formatdatetime(rs("fin"),2)%></font>
</td>
</tr>
<%end if%>
</table>
</center>
</div>
<%rs.close
set conn = nothing
end if%>
<p align="center"><font face="Verdana" size="1"><a href="javascript:close();">Cerrar
ventana</a></font></p>
<p align="center"><font face="Verdana" size="1"><a href="historico.asp">Ver todas las encuestas</a></font></p>


<p align="right"><font face="Verdana" size="1">Sistema de encuestas de <a href="http://www.asptutor.com"
target="_blank"><img border="0" src="http://www.asptutor.com/images/asptutor88x31.gif"
align="center"></a></font></p>

</body>

</html>
```

historico.asp

```
<html>
```

```
<head>

<title>Historico de encuestas</title>
</head>

<body>
<font face="Verdana, Arial, Helvetica, sans-serif" size="2"><h3

style="background-color: rgb(0,255,255)"><b>Histórico de encuestas</b> </h3>
</font>
<br>

<%
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & Server.MapPath("/xxxxx/Tencuestas.mdb"))
dim colores(3)
colores(0)="red"
colores(1)="blue"
colores(2)="green"
colores(3)="black"
tamatabla=400 'tamaño de la tabla

set rs=createobject("ADODB.Recordset")
sqltxt="SELECT * from encuestas order by idencuesta desc"
rs.open sqltxt,conn
do while not rs.EOF%>

<div align="center">
<center>
<table border="1" width="<%=tamatabla%>" bgcolor="#C0C0C0" bordercolor="#0000FF">
<tr>
<td width="<%=tamatabla%>" bgcolor="#0000FF">
<p align="center"><font color="#FFFFFF" face="Verdana" size="1"><b><%=rs("pregunta")%></b></font></td>
</tr>
<tr>
<td width="<%=tamatabla%>">

<table border="0" width="<%=tamatabla%>">
<%totvotos=rs("r1")+rs("r2")+rs("r3")+rs("r4")
if totvotos< 1 then totvotos=1
for i = 0 to rs("nopciones")-1 step 1 %>
<%porcentaje=formatnumber(rs(i+4)*100/totvotos,2)%>
<tr>
<td width="<%=tamatabla*30/100%>" valign="top"><font face="Verdana" size="1"><b><%=rs(i)%
></b></font><br>
<font face="Verdana" color="#FFFFFF" size="1"><b><%=porcentaje%>%</b></font></td>

<td width="<%=tamatabla*70/100%>">

<table border="0" width="<%=round(porcentaje)%>%" bgcolor="<%=colores(i)%>">
<tr>
<td width="100%">
</td>
</tr>
</table>
</td>
</tr>
<%next

%>
</table>

</td>
</tr>
<tr>
<td width="300">
<p align="center"><font face="Verdana" size="1"><b>Total de votos: <font color="#FF0000"><%=totvotos%
></font> </b></font></p>
</td>
</tr><%if rs("inicio")<>"" then%>

<tr>
<td width="300">
<font face="Verdana" size="1">Fecha de inicio: <%=rs("inicio")%></font>
```

```
</td>
</tr><%end if%>

<%if rs("fin")<>"" then%>
<tr>
<td width="300">
<font face="Verdana" size="1">Fecha de cierre: <%=formatdatetime(rs("fin"),2)%></font>
</td>
</tr>
<%end if%>
</table>
</center>
</div>
<br>
<%rs.movenext
loop
rs.close
set rs=nothing
conn.close
set conn=nothing

%>

<p align="right"><font face="Verdana" size="1">Sistema de encuestas de <a href="http://www.asptutor.com"
target="_blank"><img border="0" src="http://www.asptutor.com/images/asptutor88x31.gif"
align="center"></a></font></p>

</body>

</html>
```

Módulos de mantenimiento

crearencuesta.htm

```
<html>

<head>

<title>Crear una nueva encuesta</title>
</head>

<body>

<div align="center">
<center>
<table border="0" width="80%">
<tr>
<td width="100%" bgcolor="#0000FF">

<p align="center"><font color="#FFFFFF">Crear una nueva encuesta</font></p>
</td>
</tr>
<tr>
<td width="100%" bgcolor="#C0C0C0">
<p align="center"><font face="Verdana" size="1">Sistema de encuestas de <a href="http://www.asptutor.com"
target="_blank"><img border="0" src="http://www.asptutor.com/images/asptutor88x31.gif"
align="center"></a></font></p>

<form method="POST" action="crearencuesta.asp">
<div align="center">
<center>
<table border="0" width="90%">
<tr>
<td width="34%" bgcolor="#0000FF"><font color="#FFFFFF">Pregunta</font></td>
<td width="66%" align="center"><input type="text" name="pregunta" size="40"></td>
</tr>
<tr>
<td width="34%" bgcolor="#0000FF"><font color="#FFFFFF">Respuesta 1</font></td>
<td width="66%" align="center"><input type="text" name="tr1" size="40"></td>
</tr>
<tr>
<td width="34%" bgcolor="#0000FF"><font color="#FFFFFF">Respuesta 2</font></td>
```

```
<td width="66%" align="center"><input type="text" name="tr2" size="40"></td>
</tr>
<tr>
<td width="34%" bgcolor="#0000FF"><font color="#FFFFFF">Respuesta 3</font></td>
<td width="66%" align="center"><input type="text" name="tr3" size="40"></td>
</tr>
<tr>
<td width="34%" bgcolor="#0000FF"><font color="#FFFFFF">Respuesta 4</font></td>
<td width="66%" align="center"><input type="text" name="tr4" size="40"></td>
</tr>
<tr>
<td width="34%"> </td>
<td width="66%" align="center"> </td>
</tr>
<tr>
<td width="34%" bgcolor="#0000FF"><font color="#FFFFFF">Clave de creación</font></td>
<td width="66%" align="center"><input type="password" name="clave" size="40"></td>
</tr>
</table>
</center>
</div>
<p align="center"><input type="checkbox" name="activar" value="ON">
Activar la encuesta una vez insertada</p>
<p align="center"><input type="submit" value="Enviar" name="B1"><input type="reset" value="Restablecer"
name="B2"><br>
</p>
</form>
</td>
</tr>
</table>
</center>
</div>

</body>

</html>
```

crearencuesta.asp

```
<html>

<head>

<title>Creacion de encuestas</title>
</head>

<body>
<%Const adOpenForwardOnly = 0
Const adOpenKeyset = 1
Const adOpenDynamic = 2
Const adOpenStatic = 3
Const adLockReadOnly = 1
Const adLockPessimistic = 2
Const adLockOptimistic = 3
Const adLockBatchOptimistic = 4

Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & Server.MapPath("/xxxxx/Tencuestas.mdb"))

set rs=createobject("ADODB.Recordset")
if request.form("activar")="ON" then
fin=month(date)&"/"&day(date)&"/"&year(date)

sqltxt="update encuestas set fin=#"&fin&"# where activa=True"
'response.write sqltxt
conn.execute sqltxt

sqltxt="update encuestas set activa=False where activa=True"
conn.execute sqltxt

end if

sqltxt="select * from clave"
```

```
rs.open sqltxt,conn
if rs("clave")<> request.form("clave") then
rs.close
%>
<div align="center">
<center>
<table border="0" width="50%">
<tr>
<td width="100%">
<table border="10" width="100%" bgcolor="#FF0000" bordercolor="#000000">
<tr>
<td width="100%">
<p align="center"><font color="#FFFFFF">Clave ERRONEA</font></td>
</tr>
</table>
</td>
</tr>
<tr>
<td width="100%">
<p align="center"><a href="javascript:window.history.back()">Intentalo de nuevo</a>
</td>
</tr>
</table>
</center>
</div>
<%else
rs.close
pregunta=trim(request.form("pregunta"))
tr1=trim(request.form("tr1"))
tr2=trim(request.form("tr2"))
tr3=trim(request.form("tr3"))
tr4=trim(request.form("tr4"))
if request.form("activar")="ON" then
activar=True
else
activar=false
end if

if tr1="" or tr2="" or pregunta="" then%>
<div align="center">
<center>
<table border="0" width="50%">
<tr>
<td width="100%">
<table border="10" width="100%" bgcolor="#FF0000" bordercolor="#000000">
<tr>
<td width="100%">
<p align="center"><font color="#FFFFFF">Al menos se deben ofrecer una pregunta y las alternativas 1 y 2
</font></td>
</tr>
</table>
</td>
</tr>
<tr>
<td width="100%">
<p align="center"><a href="javascript:window.history.back()">Intentalo de nuevo</a>
</td>
</tr>
</table>
</center>
</div>

<%
else
nopciones=2
rs.CursorType = adOpenKeyset
rs.LockType = adLockOptimistic
rs.open "encuestas",conn
rs.addnew
rs("pregunta")=pregunta
rs("tr1")=tr1
rs("r1")=0
rs("tr2")=tr2
rs("r2")=0
```

```
if tr3 <> "" then
nopciones=3
rs("tr3")=tr3
rs("r3")=0
if tr4 <> "" then
nopciones=4
rs("tr4")=tr4
rs("r4")=0
end if
end if
rs("nopciones")=nopciones
rs("activa")=activar
if request.form("activar")="ON" then
rs("inicio")=date
end if
rs.update%>
<div align="center">
<center>
<table border="0" width="90%" cellpadding="2" bgcolor="#C0C0C0">
<tr>
<td width="100%" bgcolor="#0000FF">
<p align="center"><font color="#FFFFFF">Nueva Encuesta creada</font>
</td>
</tr>
<tr>
<td width="100%"><br><div align="center">
<center>
<table border="0" width="80%">
<tr>
<td width="21%" bgcolor="#0000FF"><font color="#FFFFFF">identificador </font></td>
<td width="79%" bgcolor="#FFFFFF"><%=rs("idencuesta")%></td>
</tr>
<tr>
<td width="21%" bgcolor="#0000FF"><font color="#FFFFFF">Pregunta</font></td>
<td width="79%" bgcolor="#FFFFFF"><%=rs("pregunta")%></td>
</tr>
<tr>
<td width="21%" bgcolor="#0000FF"><font color="#FFFFFF">Opción 1</font></td>
<td width="79%" bgcolor="#FFFFFF"><%=rs("tr1")%></td>
</tr>
<tr>
<td width="21%" bgcolor="#0000FF"><font color="#FFFFFF">Opción 2</font></td>
<td width="79%" bgcolor="#FFFFFF"><%=rs("tr2")%></td>
</tr>
<tr>
<td width="21%" bgcolor="#0000FF"><font color="#FFFFFF">Opción 3</font></td>
<td width="79%" bgcolor="#FFFFFF"><%=rs("tr3")%></td>
</tr>
<tr>
<td width="21%" bgcolor="#0000FF"><font color="#FFFFFF">Opción 4</font></td>
<td width="79%" bgcolor="#FFFFFF"><%=rs("tr4")%></td>
</tr>
<tr>
<td width="21%" bgcolor="#0000FF"><font color="#FFFFFF">Activacion</font></td>
<td width="79%" bgcolor="#FFFFFF"><%=activar%></td>
</tr>
</table>

</center>
</div>

<br>
</td>
</tr>
</table>
</center>
</div>
<p> </p>

<%rs.close
conn.close
end if
end if%>
```

```
<p align="right"><font face="Verdana" size="1">Sistema de encuestas de <a href="http://www.asptutor.com"
target="_blank"><img border="0" src="http://www.asptutor.com/images/asptutor88x31.gif"
align="center"></a></font></p>

</body>

</html>
```

> **Nota:** podéis usar y modificar este código a vuestro antojo, pero en el caso de su uso en Internet, deberá mantener una referencia a www.asptutor.com

Aplicación de foros en ASP

Antes de liarte con el ejemplo, puedes darte una vuelta por Los foros de la cueva [http://www.asptutor.com/asp/ejemplos/foros.asp] para que te hagas una idea del funcionamiento de la aplicación.

Este ejercicio, sin ser complicado, si requiere de una base minima de formación en HTML y ASP-ADO, asi que no te lies con él antes de haber echado un vistazo a TODOS los ejemplos anteriores.

La base de datos sobre la que esta elaborado es ACCESS y utiliza la alocación directa del Driver sin usar DSN (esta es la forma de declaración a la que te obligan la mayoria de los servidores gratuitos al no dejarte definir DSN's de sistema)

Definición de la base de datos

Nombre de la Base de Datos: PyR.mdb

Consta de 3 tablas:

Tabla FOROS

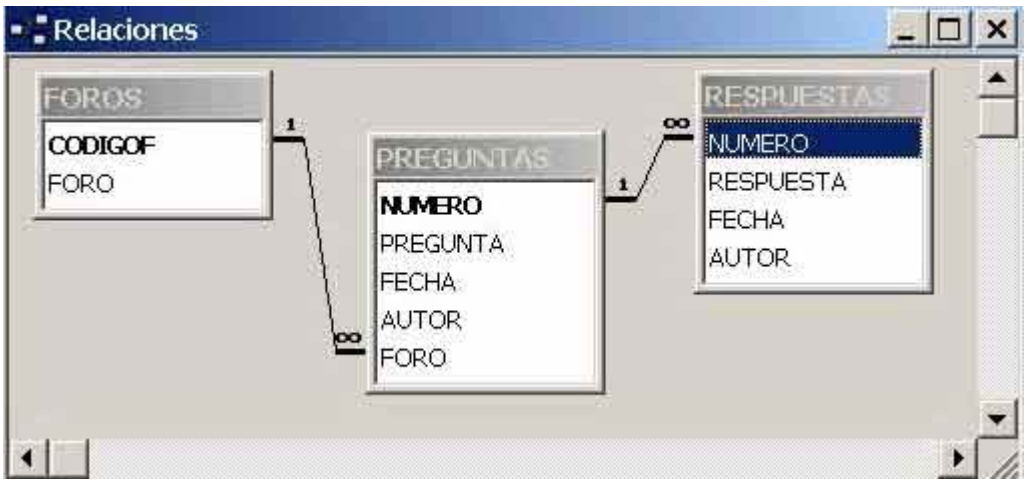
Tabla PREGUNTAS

Tabla RESPUESTAS

RESPUESTAS : Tabla

Nombre del campo	Tipo de datos
NUMERO	Numérico
RESPUESTA	Memo
FECHA	Fecha/Hora
AUTOR	Texto

Relaciones entre las tablas:



Modulos y paginas estaticas de la aplicación

La aplicación consta de 8 paginas asp y 1 html estatico. Se podria reducir el numero de modulos reutilizando parte del codigo de estos, pero por claridad he preferido plantearlo de esta forma.

addforo.asp	Modulo de inclusión de un nuevo foro en la tabla Foros
addpregunta.asp	Modulo de inclusión de una nueva pregunta en la tabla Preguntas
addrespuesta.asp	Modulo de inclusión de una nueva respuesta en la tabla Respuestas
foros.asp	Modulo de presentación, listado de todos los foros
insertarforo.htm	Formulario de inserción de un nuevo foro
insertarpregunta.asp	Formulario para insertar una nueva pregunta
pyr.asp	Modulo de visualización de todas las preguntas de un determinado foro
responder.asp	Formulario para insertar una respuesta a una pregunta
vr.asp	Modulo de visualización de todas las respuestas a una determinada pregunta.

addforo.asp

```
<%
Const adOpenForwardOnly = 0
Const adOpenKeyset = 1
Const adOpenDynamic = 2
Const adOpenStatic = 3
Const adLockReadOnly = 1
Const adLockPessimistic = 2
Const adLockOptimistic = 3
Const adLockBatchOptimistic = 4%>
<%foro=request.form("foro")
if foro="" then %>
<html>

<head>
```

```
<title>Error al insertar pregunta</title>
</head>

<body bgcolor="#CCFFCC">

<p align="center"><img border="0" src="images/titulo.jpg"></p>

<p align="center">Ningun campo puede estar en blanco,
<a href="insertarforo.htm">
inténtalo otra vez</a></p>
</body>

</html>

<%else
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" & Server.MapPath("\ewebcity\db\pyr.mdb"))

set rs=createobject("ADODB.Recordset")
rs.CursorType = adOpenKeyset
rs.LockType = adLockOptimistic

rs.open "foros",conn
rs.addnew
rs("foro")=foro
rs.update
rs.close
response.redirect "foros.asp"
end if
%>
```

addpregunta.asp

```
<%
Const adOpenForwardOnly = 0
Const adOpenKeyset = 1
Const adOpenDynamic = 2
Const adOpenStatic = 3
Const adLockReadOnly = 1
Const adLockPessimistic = 2
Const adLockOptimistic = 3
Const adLockBatchOptimistic = 4%>
<%autor=request.form("autor")
nforo=request.form("nforo")
pregunta=request.form("pregunta")
if autor="" or pregunta="" or nforo="" then %>
<html>

<head>

<title>Error al insertar pregunta</title>
</head>

<body bgcolor="#CCFFCC">

<p align="center"><img border="0" src="images/titulo.jpg"></p>

<p align="center">Ningun campo puede estar en blanco,
<a href="insertarpregunta.asp?nforo=<%=nforo%>">
inténtalo otra vez</a></p>
</body>

</html>

<%else
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" & Server.MapPath("\ewebcity\db\pyr.mdb"))

set rs=createobject("ADODB.Recordset")
rs.CursorType = adOpenKeyset
rs.LockType = adLockOptimistic

rs.open "preguntas",conn
```

```
rs.addnew
rs("foro")=nforo
rs("autor")=autor
rs("fecha")=now()
rs("pregunta")=pregunta
rs.update
rs.close
donde="pyr.asp?nforo="&nforo
response.redirect donde
end if%>
```

addrespuesta.asp

```
<%

Const adOpenForwardOnly = 0
Const adOpenKeyset = 1

Const adOpenDynamic = 2
Const adOpenStatic = 3
Const adLockReadOnly = 1
Const adLockPessimistic = 2
Const adLockOptimistic = 3
Const adLockBatchOptimistic = 4%>
<%nombre=request.form("nombre")
nforo=request.form("nforo")
respuesta=request.form("respuesta")
numero=request.form("numero")
if nombre="" or respuesta="" or numero="" or nforo="" then %>
<html>

<head>

<title>Error al insertar pregunta</title>
</head>

<body bgcolor="#CCFFCC">

<p align="center"><img border="0" src="images/titulo.jpg"></p>

<p align="center">Ningun campo puede estar en blanco,
<a href="responder.asp?numero=<%=numero%>&nforo=<%=nforo%>">
inténtalo otra vez</a></p>
</body>

</html>

<%else
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" & Server.MapPath("\ewebcity\db\pyr.mdb"))

set rs=createobject("ADODB.Recordset")
rs.CursorType = adOpenKeyset
rs.LockType = adLockOptimistic

rs.open "respuestas",conn
rs.addnew
rs("numero")=numero
rs("autor")=nombre
rs("fecha")=now()
rs("respuesta")=respuesta
rs.update
rs.close
donde="vr.asp?numero="&numero&"&nforo="&nforo
response.redirect donde
end if%>
```

foros.asp

```
<html>
<%sqltxt="select * from foros order by foro"
Set Conn = Server.CreateObject("ADODB.Connection")
```

```
Conn.Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" & Server.MapPath("\ewebcity\db\pyr.mdb"))

set rs=createobject("ADODB.Recordset")
set rsr=createobject("ADODB.Recordset")
rs.open sqltxt,conn%>

<head>
<title>La Cueva de Smaug Foros</title>
</head>

<body bgcolor="#CCFFCC">

<p align="center"><img border="0" src="images/titulo.jpg"></p>

<p align="center"> </p>
<div align="center">
<center>
<table border="0" width="575">
<tr>
<td colspan="2" align="center" bordercolor="#008000" bgcolor="#008000" width="565"><font face="Verdana"
color="#FFFFFF" size="2">Los

Foros de la Cueva TEMAS LIBRES</font></td>

</tr>
<%if rs.eof then%>
<tr><td colspan="2" width="565" bgcolor="#FFCCCC">No hay foros disponibles</td></tr>
<%else
do while not rs.eof%>
<tr>
<td width="470" bgcolor="#FFCCCC"><font face="Verdana" size="1"><%=rs("foro")%></font></td>
<td width="89" bgcolor="#FFCCCC">
<p align="center"><font face="Verdana" size="1"><a href="pyr.asp?nforo=<%=rs("codigof")%>"><img border="0"
src="images/ir.gif" WIDTH="13" HEIGHT="13"></a></font></p>
</td>

<%rs.movenext
   loop
rs.close
end if%>

</table>
</center>
</div>
<br>
<div align="center">
<center>
<table border="0" width="50%" bgcolor="#FFFFFF">
<tr>
<td width="30%" align="center"><font face="Verdana" size="1"><a href="insertarforo.htm">Crear
un nuevo Foro</a></font></td>
</tr>
</table>
</center>
</div>

</body>

</html>
```

Insertarforo.htm

```
<html>

<head>

<title>Crear un Nuevo Foro</title>
</head>

<body bgcolor="#CCFFCC">

<p align="center"><img border="0" src="images/titulo.jpg" width="450" height="75"></p>
```

```
<p align="center"><font face="Verdana">Crear un nuevo Foro</font></p>

<form method="POST" action="addforo.asp">
<table border="1" width="100%">
<tr>
<td width="100%" bgcolor="#008000"><font color="#FFFFFF">Nombre del Foro</font></td>
</tr>
<tr>
<td width="100%">
<p align="center"><input type="text" name="foro" size="50">
</p>
</td>
</tr>

</table>
<p align="center"><input type="hidden" name="nforo" value="<%=nforo%>">
<input type="submit" value="Enviar" name="B1"><input type="reset" value="Restablecer" name="B2"></p>
</form>

</body>

</html>
```

Insertarpregunta.asp

```
<%nforo=request.querystring("nforo")%>
<html>

<head>

<title>Insertar una pregunta</title>
</head>

<body bgcolor="#CCFFCC">

<p align="center"><img border="0" src="images/titulo.jpg"></p>

<p align="center"><font face="Verdana">Insertar una pregunta</font></p>

<form method="POST" action="addpregunta.asp">
<table border="1" width="100%">
<tr>
<td width="100%" bgcolor="#008000"><font face="Verdana" size="2" color="#FFFFFF">Pregunta</font></td>
</tr>
<tr>
<td width="100%">Pregunta insertada por: <input type="text" name="autor" size="20">
</td>
</tr>
<tr>
<td width="100%">
<p align="center"><textarea rows="5" name="pregunta" cols="50"></textarea></td>
</tr>
</table>
<p align="center"><input type="hidden" name="nforo" value="<%=nforo%>">
<input type="submit" value="Enviar" name="B1"><input type="reset" value="Restablecer" name="B2"></p>
</form>

</body>

</html>
```

pyr.asp

```
<%nforo=request.querystring("nforo")
if nforo="" then
response.redirect "foros.asp"
end if
sqltxt="select * from foros where codigof="&nforo
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" & Server.MapPath("\ewebcity\db\pyr.mdb"))
set rs=createobject("ADODB.Recordset")
set rsr=createobject("ADODB.Recordset")
rs.open sqltxt,conn
```

```
if rs.eof then
response.redirect "foros.asp" 'evitamos el error de foro borrado
end if
nombredelforo=rs("foro")
rs.close

sqltxt="select * from preguntas where foro="&nforo
rs.open sqltxt,conn%>
<html>

<head>
<title>La Cueva de Smaug  (Foros)</title>
</head>

<body bgcolor="#CCFFCC">

<p align="center"><img border="0" src="images/titulo.jpg"></p>

<p align="center"><font face="Verdana"><font size="4">Foro: </font> <%=nombredelforo%>
<div align="center">
<center>
<table border="0" width="575">
<tr>
<td width="62" align="center" bordercolor="#008000" bgcolor="#008000"><font face="Verdana" size="1"
color="#FFFFFF">Fecha</font></td>
<td width="254" align="center" bordercolor="#008000" bgcolor="#008000"><font face="Verdana" size="1"
color="#FFFFFF">Pregunta</font></td>
<td align="center" bordercolor="#008000" bgcolor="#008000" width="82"><font face="Verdana" size="1"
color="#FFFFFF">Numero
de Respuestas</font></td>
<td width="94" align="center" bordercolor="#008000" bgcolor="#008000">
<p align="center"><font face="Verdana" size="1" color="#FFFFFF">Ver
Respuestas</font></td>
<td width="49" align="center" bordercolor="#008000" bgcolor="#008000">
<p align="center"><font face="Verdana" size="1" color="#FFFFFF">Responder</font></p>
</td>
</tr>
<%if rs.eof then%>
<tr><td colspan=5 width="565" bgcolor="#FFCCCC">No hay preguntas</td></tr>
<%else
do while not rs.eof%>
<tr>
<td width="62" bgcolor="#FFCCCC"><font face="Verdana" size="1"><%=formatdatetime(rs("fecha"),2)%
></font></td>
<td width="254" bgcolor="#FFCCCC" valign="middle">
<font face="Verdana" size="1"><%=rs("pregunta")%></font></td>
<%sqltxt2="select count(*) from respuestas where numero ="&rs("numero")
rsr.open sqltxt2,conn %>

<td width="82" bgcolor="#FFCCCC">
<p align="center"><font face="Verdana" size="1"><%=rsr(0)%></font></td>

<td width="94" valign="middle" align="center" bgcolor="#FFCCCC">
<%if rsr(0)<> 0 then%>
<a href="vr.asp?numero=<%=rs("numero")%>&nforo=<%=nforo%>"><font face="Verdana" size="1"><img
border="0" src="images/ir.gif" width="13" height="13"></font></a>
<font face="Verdana" size="1">
<%else%>
  <%end if
rsr.close%>
</font>
</td>
<td width="49" align="center" bgcolor="#FFCCCC">
<p align="center"><font face="Verdana" size="1"><a href="responder.asp?numero=<%=rs("numero")%
>&nforo=<%=nforo%>"><img border="0" src="images/ir.gif" width="13" height="13"></a></font></p>
</td>
</tr>
<%rs.movenext
loop
rs.close
end if%>
</table>
</center>
</div>
```

```
<p align="center"> </p>
<div align="center">
<center>
<table border="0" width="50%" bgcolor="#FFFFFF">
<tr>
<td width="30%" align="center"><font face="Verdana" size="1"><a href="insertarpregunta.asp?nforo=<%=nforo%
>">Insertar
una pregunta</a></font></td>
</tr>
</table>
</center>
</div>
<br>
<div align="center">
<center>
<table border="0" width="50%" bgcolor="#FFFFFF">
<tr>
<td width="30%" align="center"><font face="Verdana" size="1"><a href="foros.asp">Volver a
Foros</a></font></td>
</tr>
</table>
</center>
</div>
</body></html>
```

responder.asp

```
<%numero=request.querystring("numero")
nforo=request.querystring("nforo")
if numero="" or nforo="" then
response.redirect "pyr.asp"
end if
sqltxt="select * from preguntas where numero="&numero
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" & Server.MapPath("\ewebcity\db\pyr.mdb"))

set rs=createobject("ADODB.Recordset")
rs.open sqltxt,conn%>

<html>

<head>
<title>Responder a una pregunta</title>
</head>

<body bgcolor="#CCFFCC">

<p align="center"><img border="0" src="images/titulo.jpg"></p>

<p align="center"><font face="Verdana">Responder
a una pregunta</font></p>
<table border="0" width="100%">
<tr>
<td width="100%" bgcolor="#008000"><font face="Verdana" color="#FFFFFF" size="2">Pregunta
realizada por <%=rs("autor")%> el dia <%=formatdatetime(rs("fecha"),2)%></font></td>
</tr>
<tr>
<td width="100%" bgcolor="#FFFFFF"><%=rs("pregunta")%></td>
<%rs.close%>
</tr>
</table>

<p> </p>
<form method="POST" action="addrespuesta.asp?nforo=<%=nforo%>">
<table border="1" width="100%">
<tr>
<td width="100%" bgcolor="#008000"><font face="Verdana" size="2" color="#FFFFFF">Respuesta</font></td>
</tr>
<tr>
<td width="100%">Respuesta elaborada por: <input type="text" name="nombre" size="20">
</td>
</tr>
<tr>
```

```
<td width="100%">
<p align="center"><textarea rows="5" name="respuesta" cols="50"></textarea></td>
</tr>
</table>
<p align="center"><input type="hidden" name="numero" value="<%=numero%>">
<input type="hidden" name="nforo" value="<%=nforo%>">
<input type="submit" value="Enviar" name="B1"><input type="reset" value="Restablecer" name="B2"></p>
</form>

</body>

</html>
```

vr.asp

```
<%numero=request.querystring("numero")
nforo=request.querystring("nforo")
if numero="" then
response.redirect "pyr.asp"
end if
sqltxt="select * from preguntas where numero="&numero
sqltxt2="select * from respuestas where numero="&numero
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" & Server.MapPath("\ewebcity\db\pyr.mdb"))

set rs=createobject("ADODB.Recordset")
rs.open sqltxt,conn%>

<html>

<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<meta name="GENERATOR" content="Microsoft FrontPage 4.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<title>Respuestas a una pregunta</title>
</head>

<body bgcolor="#CCFFCC">

<p align="center"><img border="0" src="images/titulo.jpg"></p>
<%if rs.eof then
response.write "no hay respuestas para esta pregunta"
else%>

<table border="1" width="100%">
<tr>
<td width="100%" bgcolor="#008000"><font face="Verdana" color="#FFFFFF" size="2">Pregunta
realizada por</font><font face="Verdana" color="#00FFFF" size="2"> <%=rs("autor")%>
</font><font face="Verdana" color="#FFFFFF" size="2">el dia</font><font face="Verdana" color="#00FFFF"
size="2"> <%=formatdatetime(rs("fecha"),2)%></font></td>
</tr>
<tr>
<td width="100%" bgcolor="#FFFFFF"><%=rs("pregunta")%></td>
<%rs.close%>
</tr>
</table>
<%rs.open sqltxt2,conn%>
<%if not rs.EOF then
nrespu=0%>
<p> </p>
<table border="0" width="100%" cellspacing="0" cellpadding="0">
<%do while not rs.EOF
nrespu=nrespu+1%>
<tr>
<td width="25%" bgcolor="#008000"><font face="Verdana" size="2"><font color="#FFFFFF">Respuesta nº
</font><font color="#FFFF00"><%=nrespu%></font></font></td>
<td width="42%" bgcolor="#008000"><font face="Verdana" size="2"><font color="#FFFFFF">autor 
</font><font color="#FFFF00"><%=rs("autor")%></font></font></td>
<td width="33%" bgcolor="#008000"><font face="Verdana" size="2"><font color="#FFFFFF">el dia </font><font
color="#FFFF00"><%=formatdatetime(rs("fecha"),2)%></font></font></td>
</tr>
<tr>
<td width="100%" bgcolor="#FFFFFF" colspan="3"><%=rs("respuesta")%>
```

```
</td>
</tr>
<%rs.movenext
loop
end if%>
</table>

<p> </p>

<div align="center">
<center>
<table border="0" width="61%">
<tr>
<td width="50%" bgcolor="#FFFFFF">
<p align="center"><a href="responder.asp?numero=<%=numero%>&nforo=<%=nforo%>"><font face="Verdana"
size="1">Insertar
Respuesta</font></a></td>
<td width="50%" bgcolor="#FFFFFF">
<p align="center"><font face="Verdana" size="1"><a href="pyr.asp?nforo=<%=nforo%>">Volver
al Foro</a></font></td>
</tr>
</table>
</center>
</div>
<%end if%>

</body></html>
```

Generando eventos en la Master Page en ASP

Una de las funcionalidades añadidas al ASP.NET 2.0 es el manejo de plantillas como parte de la creación de nuestras páginas. Estas plantillas, conocidas como Páginas Maestras o Master Pages, nos permiten crear un diseño que será compartido por otras muchas páginas de nuestro sitio a las que llamaremos Páginas de Contenido o Content Pages. Lo interesante es que las Master Pages no solo contendrán la apariencia visual que compartirán otras muchas páginas, sino que también contendrán el comportamiento. Es decir, en nuestras Master Pages podemos colocar controles como botones, combos, etc. Y todos los eventos que implementemos para esos controles, quedarán implementados para todas las Content Pages que utilicen la Master Page.

No es interés de este artículo mostrar cómo crear una Master Page. Existen infinidad de artículos en Internet que explican cómo hacerlo. El objetivo de este trabajo es mostrar cómo podemos relacionar nuestra Master Page con las diferentes Content Pages y cómo pasar eventos entre ellas.

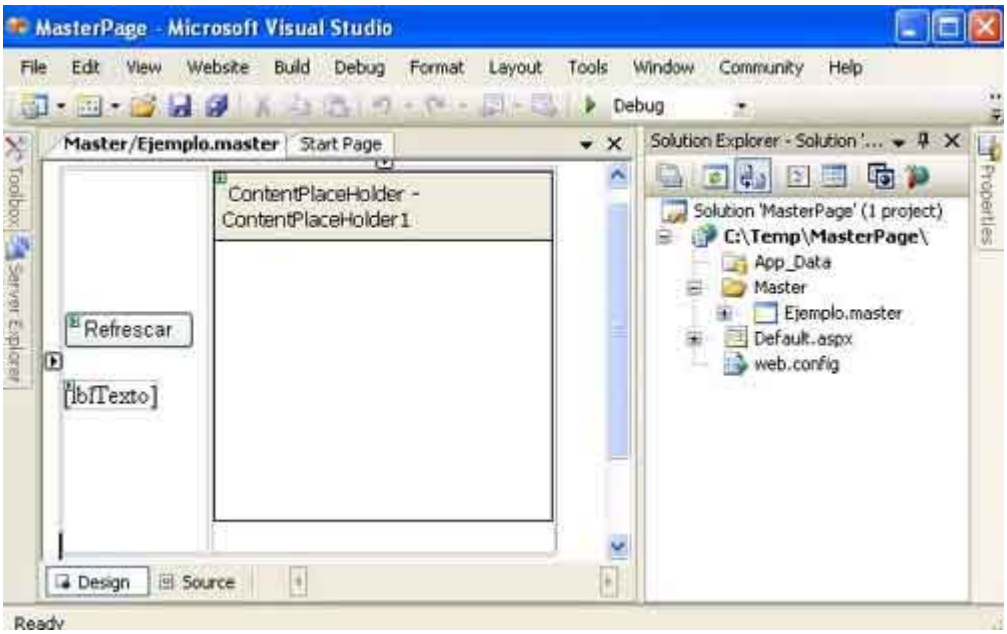
Comencemos creando una aplicación que nos permita ejemplificar todo esto.

Supongamos que deseamos que un grupo de nuestras páginas tenga un botón y una etiqueta. Para lograrlo, colocaremos ese botón y esa etiqueta en una Master Page. Supongamos que deseamos que cada página que use la Master Page, pueda colocar en la etiqueta el texto que desee y que ese texto se coloque cuando se presione el botón Refrescar colocado en la Master Page. Hagámoslo.

Vamos a partir con que tenemos un proyecto web lo mismo en el Visual Studio 2005 que en el Visual Web Developer. En ese proyecto creamos una carpeta llamada Master y dentro de ella colocamos una Master Page a la que llamamos Ejemplo.master.

En nuestra plantilla vamos a colocar una tabla que contenga un botón Refrescar cuyo ID es btnRefrescar, un label con ID lblTexto y el ContentPlaceHolder donde las Content Pages que utilicen esta Master Page colocarán sus contenidos.

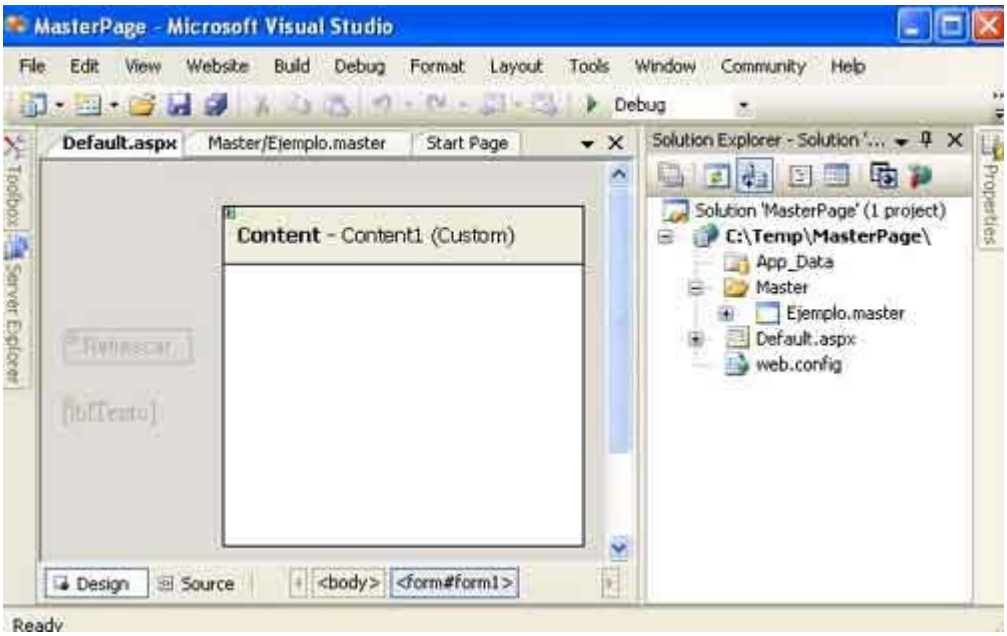
La Figura 1 muestra cómo se vería nuestro proyecto hasta aquí.



Ahora necesitamos que una Content Page pueda acceder al label lblTexto para que pueda modificar su información.

Para eso, borremos la página Default.aspx que se nos crea por defecto cuando construimos el proyecto, y agreguemos una nueva WebForm, a la que llamemos Default.aspx. y en su creación digamos que utilice la Master Page que hemos creado.

Una vez hecho esto, el Visual Studio se vería tal y como muestra la Figura 2.



Ahora desearíamos poder acceder, desde la página Default.aspx al label que colocamos en la Master Page, para poder modificarle el texto. Tratemos de hacer ese acceso, desde el método Page_Load de la página.

Nuestra página por defecto va a tener una propiedad llamada Master, que es una referencia a la Master Page que utiliza. A través de esa propiedad podemos acceder a cualquier control colocado en la página maestra, utilizando el método FindControl que nos ofrece la MasterPage. El método Page_Load de nuestra página Default.aspx podría quedar de la siguiente manera:

```
Protected Sub Page_Load(...) Handles Me.Load
```

```
    CType(Master.FindControl("lblTexto"), Label).Text = "Ejemplo"
End Sub
```

No obstante, podemos usar otra vía para acceder al label directamente, sin tener que buscarlo. Para lograrlo, tenemos el inconveniente de que el label es un campo protegido de la Master Page, por lo que para accederlo tenemos que añadirle a nuestra página maestra, una propiedad pública a través de la cual accedamos al label. El código de nuestra Master Page nos podría quedar de la siguiente manera:

```
Partial Class Master_Ejemplo
  Inherits System.Web.UI.MasterPage

    Public ReadOnly Property LabelTexto() As Label
      Get
        Return Me.lblTexto
      End Get
    End Property
End Class
```

Una vez hecho eso, es muy fácil acceder desde nuestra página Default.aspx, al label de la página maestra, utilizando la propiedad Master que nuestra página tiene. Para eso, solo tenemos que hacerle un casting a la propiedad Master, de tal forma que la veamos como una Master_Ejemplo, que es el tipo real de nuestra Master Page y de esa manera poder acceder a la nueva propiedad creada por nosotros.

El método Page_Load de nuestra página Default.aspx podría quedar como sigue.

```
Protected Sub Page_Load(...) Handles Me.Load
  CType(Me.Master, Master_Ejemplo).LabelTexto.Text = "Ejemplo"
End Sub
```

No obstante, podemos evitarnos el casting, si indicamos el tipo a través de la siguiente directiva <%@ MasterType VirtualPath="~/Master/Ejemplo.master" %>

Es decir, debemos colocar esa directiva al inicio de nuestra página Default.aspx, para poder acceder a la nueva propiedad LabelTexto directamente desde la variable Master.

El documento Default.aspx quedaría con el siguiente contenido:

```
<%@ Page Language="VB" MasterPageFile="~/Master/Ejemplo.master" ... %>
<%@ MasterType VirtualPath="~/Master/Ejemplo.master" %>
<asp:Content ID="Content1" ...>
</asp:Content>
```

Y el código de su método Page_Load quedaría de la siguiente manera:

```
Protected Sub Page_Load(...) Handles Me.Load
   Master.LabelTexto.Text = "Ejemplo"
End Sub
```

Sin embargo, lo ideal sería que valorizáramos el label no en el Page_Load de la página, sino como resultado del OnClick del botón que colocamos en la MasterPage. La pregunta entonces es: ¿Cómo lograr que nuestra página se entere de un evento que se está generando en la Master Page? La respuesta a esta pregunta la encontraremos en el uso de los Eventos.

Añadamos a nuestra clase Master_Ejemplo, es decir, a nuestra Master Page, un evento público al que llamemos Refrescar_Click y que sea del tipo EventHandler. Este tipo EventHandler no es más que un delegado de la biblioteca de clases de .NET que se encuentra en el espacio de nombre System.

Una vez que hemos creado el evento, añadamos a nuestra Master Page un método btnRefrescar_Click que manipule el evento Click del botón Refrescar y dentro de ese método, simplemente lancemos el evento Refrescar_Click que hemos creado.

Resumiendo. Nuestra intensión es colocar un evento público en nuestra página maestra, al que se puedan registrar todas las páginas que usen esa Master Page. Esas páginas entonces, registrarán métodos a ese evento, de tal forma que cuando la Master Page genere el evento, como resultado del OnClick del botón, las otras páginas se enteren y puedan realizar alguna acción al respecto, como puede ser, cambiar el texto del label.

Finalmente el código de nuestra Master Page quedaría de la siguiente manera.

```
Partial Class Master_Ejemplo
  Inherits System.Web.UI.MasterPage

  Public Event Refrescar_Click As EventHandler

  Public ReadOnly Property LabelTexto() As Label
    Get
      Return Me.lblTexto
    End Get
  End Property

  Protected Sub btnRefrescar_Click(...) Handles btnRefrescar.Click
    RaiseEvent Refrescar_Click(sender, e)
  End Sub
End Class
```

Solo nos queda por hacer un pequeño ajuste para que desde nuestra página Default.aspx podamos escuchar el evento Refrescar_Click de la página maestra.

Sucede que la propiedad Master que tenemos en nuestra página Default.aspx, no fue declarada con la palabra reservada WithEvents. Debemos entonces crear un nuevo campo en nuestra clase a través del cual manipulemos los eventos generados en la Master Page. Ese campo debe tener el modificador WithEvents y lo podemos crear del tipo Master_Ejemplo. En el Page_Load de la página le asignaríamos a este nuevo campo, el valor del campo Master que ya teníamos, y con eso ya podemos manipular los eventos.

El código de nuestra página nos quedaría de la siguiente manera:

```
Partial Class _Default
  Inherits System.Web.UI.Page

  Protected WithEvents MyMaster As Master_Ejemplo

  Protected Sub Page_Load(...) Handles Me.Load
    MyMaster = Master
  End Sub

  Protected Sub MyMaster_Refrescar_Click(...) Handles MyMaster.Refrescar_Click
    Master.LabelTexto.Text = "Ejemplo"
  End Sub
End Class
```

Si echamos a correr la aplicación y le damos click al botón Refrescar, veremos como se muestra el texto “Ejemplo”

Conclusión

En este artículo hemos visto cómo acceder desde cualquier página a los diferentes controles contenidos en la Master Page y sobre todo, como implementar los diferentes eventos que en ella se puedan generar.

Redondear decimales en ASP

Vamos a realizar un ejercicio muy simple que se trata de redondear una cifra, de modo que

tenga el número de decimales que deseamos. El redondeo de decimales en ASP es muy sencillo, gracias a una función de Visual Basic Script llamada Round().

Función Round()

La función Round() recibe dos parámetros, el número que se desea redondear y el número de decimales que se desea que tenga:

Round (número, num_decimales)

Devuelve el número, redondeado de manera que tenga tantas posiciones decimales como se envíe en el segundo parámetro. El segundo parámetro es opcional y si no se indica, se entiende que se desea redondear sin decimales.

La función Round() siempre redondea al valor más próximo, con los decimales indicados.

Posibles ejemplos, que nos pueden dar una idea exacta de la manera de funcionar de Round(), son:

Round (2.2) devuelve 2
Round (2.6) devuelve 3
Round (41.2855,3) devuelve 41.286
Round (41.2009,2) devuelve 41.2
Round (41.2009,3) devuelve 41.201

Funciones Int() y Fix()

Estas dos funciones están relacionadas con Round(). Sirven para obtener la parte entera de un número con decimales. No existen diferencias entre Int() y Fix(), las dos hacen exactamente lo mismo.

Veamos un par de ejemplos:

Int (54.23) devuelve 54
Fix (0.2) devuelve 0

Como vemos, no es un redondeo, sino la obtención del número entero una vez hemos quitado los decimales.

> **Nota:** Tenemos otro artículo para aprender a redondear decimales en PHP [http://www.desarrolloweb.com/articulos/2442.php]

GeoLite Country para ASP. Conocer el país de una IP

Vamos a hablar de GeoLite Country, un producto gratuito que nos sirve para conocer el país de procedencia de nuestros visitantes, que se deduce a partir de su dirección IP. Cada dirección IP corresponde a un proveedor de acceso a Internet, localizado en un país. Cada proveedor tiene asignado un rango de direcciones IP para realizar sus actividades. Estos productos tienen una base de datos con esos rangos de direcciones IP que se corresponden a cada proveedor y país.

Este software se puede descargar gratis, pero no es tan preciso como GeoIP Country, la versión avanzada de pago, también desarrollada por MaxMind. Como GeoLite Country es gratuito, se ofrece sin soporte técnico y las actualizaciones de la base de datos son menos frecuentes que la versión de pago. Trataremos en este artículo concretamente la versión gratuita para instalarla en un servidor ASP.

En la página de MaxMind [http://www.maxmind.com/] se pueden encontrar tanto las versiones gratuitas como las de pago de GeoIP. Tienen además varios formatos o módulos, para averiguar países o ciudades y un API que se puede utilizar en páginas web para consultar la base de datos de IPs.

La versión gratuita de GeoIP está en http://www.maxmind.com/app/geoip_country

Lo más sencillo es utilizar el API que ofrecen gratuitamente para hacer localizaciones de países. El API está en la dirección http://www.maxmind.com/app/api

Como estamos interesados en la instalación del API en un servidor ASP, de todos los lenguajes en los que se ofrece el API, nos interesa el "MS COM Object" (Objeto COM de Microsoft), que está en la dirección http://www.maxmind.com/app/com

Allí veremos un enlace para descarga gratis, que nos llevará a un directorio donde encontraremos varios archivos. Nosotros hemos seleccionado el archivo GeoIP-COM-1.0.zip. Por supuesto, habría que obtener la versión más nueva disponible. Una vez descargado, podemos descomprimirlo y veremos que incluye varios ficheros, con todo lo que nos hace falta para comenzar nuestra localización de países por IP. Entre lo que encontraremos en este fichero está un archivo .dll que tenemos que instalar en el servidor y una serie de páginas con ejemplos de programación para localizar el país del visitante por su IP. Hay ejemplos de programación en ASP, PHP, Cold Fusion, Perl, Python… También hay un txt con instrucciones para instalar el componente y usarlo correctamente.

Instalar el componente GeoIP

Para instalar un componente COM en el servidor tenemos que realizar una serie de pasos relatados en el artículo Componentes de servidor ASP [http://www.desarrolloweb.com/articulos/657.php]. Aunque en el txt nos explican también la tarea:

1. Copiar GeoIPDemo.dat en C:\Archivos de programa\geoip\GeoIPDemo.dat
   (Este directorio podemos cambiarlo a nuestro gusto. Luego, en la programación ASP, indicaremos el directorio concreto donde hayamos colocado el archivo)
2. Copiar GeoIPCOM.dll en el directorio "system32". Dependiendo de nuestra instalación de Windows este directorio puede estar en uno u otro sitio. Por ejemplo, en Win XP Pro está en C:\windows\system32
3. Registrar el componente COM, con las instrucciones:
   C:\> cd windows\system32
   C:\windows\system32> regsvr32 GeoIPCOM.dll

De todos modos, muchas veces querremos instalar el componente en un servidor de alojamiento compartido, donde tendremos que realizar acciones distintas. Lo normal es que en el servidor nos indiquen como registrar componentes COM. Seguramente algunos proveedores no lo permitan, pero lo normal es que sí nos dejen instalarlos. En caso que no sea así, tendríamos que pensar en mudarnos de servidor de hosting.

En cuanto a copiar .dat en C:\Archivos de programa\geoip\GeoIPDemo.dat, lógicamente, en un servidor de hosting normal no vamos a poder, porque no vamos a tener acceso a ese directorio. Por lo que tendremos que subir por FTP el archivo y alojarlo en un directorio donde sí podamos subir archivos. Como hemos dicho, posteriormente, en el script ASP para localizar la IP, podremos indicar dónde se ha colocado ese fichero.

Script ASP de prueba del componente

Como decíamos, en el propio paquete de descarga tenemos una serie de ejemplos en varios lenguajes de programación para probar el sistema de localización de IP de GeoLite. Nosotros vamos a utilizar el archivo .asp, que hemos modificado un poco para añadirle unos comentarios y simplificarlo a nuestro gusto.

El código del ejemplo que ha funcionado perfectamente en nuestro servidor local es el siguiente:

```
<% Response.Buffer = TRUE %>
<html>
<title>GeoIP Lite Test</title>
<body bgcolor=#CCCCDD>
<h1><center>GeoIP Lite Test</center></h1>
<br><center>
<br>
<form action="GeoIPTest.asp" method="POST">
  <table border=0>
    <tr><td>hostname:</td><td><input type=text name=hostname></td></tr>
  </table>
  <input type=submit value="Submit" name=submit>
</form>
<%
  if Request.Form("submit") = "Submit" then
    'si se reciben datos de formulario

    'recibo el nombre del host que queremos ver la ip
  hostname = Request.Form("hostname")

 'creo el objeto geoip
 set geoip = Server.CreateObject("GeoIPCOM.GeoIP")
  'cargo el fichero con los datos de ips y países
  geoip.loadDataFile("C:\Archivos de programa\geoip\GeoIPDemo.dat")
  'obtengo datos sobre el hostname recibido por formulario
  country_code = geoip.country_code_by_name(hostname)
  country_name = geoip.country_name_by_name(hostname)

  'mostramos los datos de este host
  Response.Write("<table cellpadding=2 border=1><tr><th colspan=2>Results</th></tr>")
  Response.Write("<tr><td>Hostname</td><td>" + hostname + "</td></tr>")
  Response.Write("<tr><td>ISO 3166 Country Code</td><td>" + country_code + "</td></tr>")
  Response.Write("<tr><td>Full Country Name</td><td>" + country_name + "</td></tr>")
  Response.Write("</table>")

 end if
%>
</center>
</body>
</html>
```

Devolver un archivo de Word desde una página ASP

Veremos aquí un típico ejemplo de creación de un archivo de Word desde una página web ASP. El archivo de Word se generará automáticamente y la página ASP, en vez de mostrar el contenido como una página web, mostrará un archivo de Word, que se podrá descargar utilizando cualquier navegador. Si es que se accede con Internet Explorer, también se podrá visualizar el archivo de Word en el propio navegador.

El ejemplo es sencillo. Simplemente hay que tratar con el ContentType, para indicar que el tipo de contenido que se envía es un archivo de Word. El propio navegador, al ver que se le envía un archivo de Word en lugar de una página web, mostrará la opción de descargarlo o abrirlo. Para especificar el ContetType de Word utilizamos esta línea de código al principio del archivo ASP:

```
<% Response.ContentType = "application/msword"%>
```

Con ello ya hemos realizado la mayor parte del trabajo. Ahora, cualquier contenido que coloquemos después de esa etiqueta, será parte del contenido del archivo de Word que se enviará al visitante.

Podemos colocar contenido HTML y se verá perfectamente en el archivo de Word:

```
<% Response.ContentType = "application/msword" %><html>
<html>
<head>
<title>Creacion de word</title>
</head>
<body bgcolor="orange" text="blue">
<h1>hola</h1>
<a href="http://www.guiarte.com">Esto es un enlace a guiarte.com</a>
<p>
<i>Gracias por tu visita</i>
</p>
</body>
</html>
```

Otra cosa que podemos hacer es tener el contenido en un archivo aparte e incluirlo por medio de un include de ASP.

```
<% Response.ContentType = "application/msword"%>
<!--#include file="archivo-incluir.html"-->
```

Así el navegador recibirá un archivo de Word generado con el contenido que hay en el fichero archivo-incluir.html.

Estadísticas en ASP

La lógica del módulo puede ser portada a cualquier otro lenguaje.

El artículo está dividido en tres secciones principales:

- Modelo de datos: en donde se explica las tablas necesarias para el módulo.
- Desarrollo: en donde se explica la programación.
- Generar reportes: en donde se explica como utilizar el modulo.

Modelo de datos

Primero debemos crear dos tablas: la tabla de conceptos, en donde se guardarán las páginas que van a ser monitoreadas, y la tabla de estadísticas en donde se guardarán todos los registros de movimientos.

Statistics (Tabla Estadisticas)

	Nombre de columna	Tipo de datos	Longitud	Permitir valores nulos
🔑	Id	int	4	
	IP	nvarchar	50	
	DateLog	datetime	8	
	Concept	int	4	
	BrowserOS	nvarchar	200	✓
	Querystring	nvarchar	200	✓
	Form	nvarchar	200	✓

StatisticsConcepts (Tabla Conceptos)

	Nombre de columna	Tipo de datos	Longitud	Permitir valores nulos
🔑	Id	int	4	
	Name	nvarchar	50	✓
	Page	nvarchar	50	✓

Cargar los datos

Antes de comenzar a utilizar nuestro módulo de estadísticas debemos configurarlo, esto significa cargar los datos correspondientes para que funcione. La tabla de Estadísticas no requiere carga de datos, ya que en ésta tabla se guardarán las acciones de nuestros visitantes.

En la tabla de Conceptos debemos cargar las páginas que deseamos nuestro módulo de estadísticas monitoree.

Desarrollo

A continuación se dividirá el desarrollo de las funciones para el módulo de estadísticas en dos apartados:

Frontend: en donde explicaré las funciones de registro
Backend: en donde explicaré como armar los reportes

Front-end

Acceso a datos

Primero vamos a establecer la conexión que utilizaremos tanto para registrar los acciones de los visitantes como así también para mostrar los reportes:

```
Set cn = Server.CreateObject("ADODB.Connection")

cn.Open "driver={SQL Server};server=localhost;uid=user;pwd=pass;database=ohstudio"
```

Por supuesto la cadena de conexión corresponde a donde tengan alojada la base de datos. En éste caso se trata de una base de datos de mssql server. Esta cadena de conexión puede ser puesta al comienzo de la página o en un include.

Función para registrar movimientos

La functión doStatistics grabará todos los movimientos que realizará el visitantes en nuestro sitio web.

La información entre las páginas de cualquier sitio web fluyen por variables POST o variables GET. Esta información es la que capturará la función.

Además la función capturará información del visitante de nuestro sitio web: ip, navegador y módulo operativo (éstos dos últimos valores son traídos desde la misma variable de servidor).

```
<%
' Función: doStatistics
' Descripción:
'     Registra los movimientos de un usuario
Function doStatistics ()
' Declara las variables
   dim lsUserInfo, lsActionsQS, lsActionsPost
   dim lsVisitedPage, liSearchConcept
   dim lrsStatisticConcept, lrsInsertStatics, lsSQL
'    Instancia los objetos
    Set lrsInsertStatics = Server.CreateObject("ADODB.Recordset")
   Set lrsStatisticConcept = Server.CreateObject("ADODB.Recordset")
   ' Recupera la pagina visitada para luego buscar su concepto
   lsVisitedPage = trim(Request.ServerVariables("SCRIPT_NAME"))
    ' Recupera las acciones
   lsActionsQS = Request.QueryString
    lsActionsPost = Request.Form
   ' Consulta los conceptos
```

```
    lsSQL = "SELECT * FROM StatisticsConcepts"
   Set lrsStatisticConcept = cn.Execute(lsSQL)
    if not lrsStatisticConcept.EOF then
       do while not lrsStatisticConcept.EOF
      ' Busca el concepto para la página visitada
      liSearchConcept = InStr(1, lsVisitedPage, trim(lrsStatisticConcept.Fields("Pagina").Value))
      ' Si encuentra el concepto registra los datos
       if liSearchConcept <> 0 then
       lsSQL = "INSERT INTO Statistics (IP, DateLog, Concept, BrowserOS, Querystring, Form)
lsSQL = lsSQL & "VALUES ("
       lsSQL = lsSQL & "'" & Request.ServerVariables("Remote_Addr") & "', "
       lsSQL = lsSQL & "'" & Now & "', "
       lsSQL = lsSQL & lrsStatisticConcept.Fields("Id").Value & ", "
      lsSQL = lsSQL & "'" & Request.ServerVariables("HTTP_USER_AGENT") & "', "
       lsSQL = lsSQL & "'" & lsActionsQS & "', "
      lsSQL = lsSQL & "'" & lsActionsPost & "') "
      cn.Execute(lsSQL)
   end if
     lrsStatisticConcept.MoveNext
   loop
    end if
   Set lrsInsertStatics = nothing
   Set lrsStatisticConcept = nothing
End Function
```

%> Lo más recomendable es que ésta función se guarde en un include y luego este include se incorpore a las páginas donde deseamos registrar movimientos.

Una vez disponible ésta función, hay que llamarla. La llamada puede ser dentro de una condición, donde comienza la página, donde termina la página, depende de en qué momento deseamos que comience a registrar los movimientos.

Es importante cargar en la tabla de Conceptos (StatisticsConcepts) todas las páginas asp que deseamos monitorear y luego agregar la siguiente llamada en las mismas:

<%Call doStatistics()%>

Listo!!! Nuestro sitio ya comenzó a registrar cada movimiento que realizan los visitantes. Ahora pasemos a analizar dichos movimientos…

Back-end

Reportes

Nuestro sitio ya ha comenzado a acumular información sobre los navegantes y sus movimientos. Es tiempo de idear una manera de mostrar dicha información. Los reportes nos ayudarán a enfocarnos en los puntos en donde nuestro sitio es más fuerte, para luego darles prioridad a la hora de actualizar el sitio web.

Funciones para generar reportes

Para generar reportes utilizaremos dos funciones: buildReport y getValue4QS.

La función buildReport es la que nos permitirá construir las consultas a la tabla de estadísticas. La función getValue4QS es la que nos permitirá filtrar por un valor que deseemos buscar en un conjunto de variables de post o get, que como ya sabemos se guardarán gracias a la función doStatistics en nuestra tabla de estadísticas. <%

```
' Función: buildReport

' Descripción:
'    Genera reportes
' Parámetros:
'    concept: id de concepto de estadísticas
'    variables: array de variables que se quiere discriminar para una estadística en particular, ej. moneda
'    values: array de valores para las variables declaradas arriba
'     form: true si se tiene que buscar en las acciones del request.form - false en la de request.querystring
```

```
'    start_date: fecha de comienzo para el filtro
'    end_date: fecha de fin para el filtro
'    count: true devuelve la cantidad de registros - false devuelve recordset con los registros
function buildReport(concept, variables(), values(), form, start_date, end_date, count)
    dim lsSQL, lrsRecordset, lsActionSel
   dim i, liCounter
    Set lrsRecordset = Server.CreateObject("ADODB.Recordset")
' Comienza a armar la consulta:
' Devuelve la cantidad o todos los registros
   if count then
      lsSQL = "SELECT COUNT(*) as total FROM Statistics WHERE Id <> 0 "
   else
      lsSQL = "SELECT * FROM Statistics WHERE Id <> 0 "
    end if
    ' Si no viene el concepto filtrar por todos
    if concept <> "" then
      lsSQL = lsSQL & "AND concept = " & concept
    end if
   ' Discrimina acciones de querystring o post
    if form then
      lsActionSel = "Form"
    else
      lsActionSel = "Querystring"
    end if
    ' Agrega todos los criterios de búsqueda
    for i = 0 to ubound(variables)
      lsSQL = lsSQL & "AND " & lsActionSel & " LIKE '%" & variables(i) & "=" & values(i) & "%' "
    next
    ' Filtra por rango de fechas
    if start_date <> "" and end_date <> "" then
      lsSQL = lsSQL & "AND cast(floor(cast(Fecha as float)) as datetime) between "
lsSQL = lsSQL & " CONVERT(datetime, '" & start_date & "', 101) and CONVERT(datetime, '" & end_date & "', 101) "
    else
      if start_date <> "" then
      lsSQL = lsSQL & "AND cast(floor(cast(Fecha as float)) as datetime) between "
lsSQL = lsSQL & "CONVERT(datetime, '" & start_date & "', 101) and CONVERT(datetime, GETDATE(), 101) "
   end if
    if end_date <> "" then
      lsSQL = lsSQL & "AND cast(floor(cast(Fecha as float)) as datetime) between "
lsSQL = lsSQL & "CONVERT(datetime, '" & end_date & "', 101) and CONVERT(datetime, GETDATE(), 101) "
    end if
    end if
   Set lrsRecordset = cn.Execute(lsSQL)
    ' Devuelve la cantidad de registros o los registros mismos
    if count then
      liCounter = cint(lrsRecordset.Fields("total").value)
      buildArrowReport = liCounter
    else
      set buildArrowReport = lrsRecordset
    end if
    set lrsRecordset = nothing
end function
' Función: getValue4QS
' Descripción:
'     Recupera el valor de una variable en una cadena de formato querystring
' Parámetros:
'     value: cadena de formato querystring
'    variable: variable a buscar dentro de la cadena de formato querystring
function getValue4QS(value, variable)
    Dim arrActions, arrTemp, i, j
    if Trim(value) <> "" then
      arrActions = split(value, "&")
      for i = 0 to ubound(arrActions)
         arrTemp = split(arrActions(i), "=")
        for j = 0 to ubound(arrTemp)
           if arrTemp (0) = variable then
              returnValue4QS = arrTemp (1)
         end if
      next
       next
   else
      getValue4QS = ""
    end if
end function
```

```
%>
```

Generar reportes

Esponja

Revisemos el siguiente escenario:

- Tenemos un sitio web de catálogo de productos de limpieza.
- Dentro de nuestro sitio web tenemos un buscador de productos.
- El buscador consta de un campo de texto llamado "SearchString".
- Cuando un visitante realiza una búsqueda se envia por post los datos del campo "SearchString" a la páginas "search.asp".

Primero debemos cargar la página search.asp en nuestra tabla de conceptos de estadísticas StatisticsConcepts. Supongamos que el código de este nuevo registro es 2.

Deseamos saber cuantas búsquedas se realizaron con la palabra "Esponja" y quienes la realizaron. Para ello debemos hacer simplemente una llamada a la función buildReport y mostrar los resultados…

Llamamos a la función:

Como la función nos devolverá un conjunto de registros hacemos un set a una variable que previamente seteamos como tipo recordset. También es posible que la función sólo devuelve la cantidad, para ello en el último parámetro debemos indicarle que sea false y sacar la instrucción set, ya que no se seteará ningún objeto, solamente devolverá un entero.

```
<%
Dim rs
Set rs = Server.CreateObject("ADODB.Recordset")
Set rs = buildReport(2, array("buscar"), array(""), true, f_i, f_f, false)
%>
```

Para mostrar los registros simplemente nos queda recorrer el recordset.

```
<table>
<%
if not rs.EOF then
%>
<tr>
   <td>Fecha</td>
<td>Búsqueda realizada</td>
   </tr>
<%
do while not rstotal.EOF
%>
   <tr>
      <td><%=(rs.Fields("DateLog").value)%></td>
      <td><%=getValue4QS(rs.Fields("Form").value, "buscar")%></td>
   </tr>
<%
   rstotal.MoveNext()
   loop
end if
set rs = nothing
%>
</table>
```

Google

Supongamos ahora que deseamos saber la cantidad de veces que accedió el robot de Google a nuestro sitio. En este caso no necesitamos utilizar la función buildReport(), solamente debemos

consultar a la tabla de estadísticas. La consulta tendría que ser de la siguiente manera:

```
<%
Dim lsSQL
lsSQL = "SELECT COUNT(*) as Total FROM Statistics WHERE BrowserOS like '%google%'"
Set lrsRecordset = cn.Execute(lsSQL)
%>
```

Sólo nos resta mostrar el valor del campo Total y tendremos la cantidad de veces que accedió el robot de Google a nuestro sitio web.

Conclusión

En este artículo hemos visto una manera de monitorear los accesos y movimientos de sus visitantes. Además como tratar dicha información recolectada. A fin de conocer el comportamiento de sus visitantes es importante realizar un análisis de los lugares que acceden y que acciones realizan. De ésta manera usted podrá mejorar constantemente su sitio web a fin de que sea mucho más útil podrá presentar el crecimiento que ha tenido para capturar más audiencia.

Por hacer:

Estas son algunas de las tareas que se podrían realizar para expandir el funcionamiento del módulo…

- ABM de Conceptos
- Tabla de Excepciones
- Clases de php, o componente del COM+

Implementado en:

Estos son los sitios en los que fue implementado el servicio de estadística.

- 3Gcommerce: http://3gbyte.sytes.net:1034/
- Cordón Turístico: http://www.cordonturistico.com.ar%20(implementado%20en%20php)/

Averiguar la dirección IP de un visitante y bloquear IPs, en ASP

Veremos como averiguar la dirección IP de un visitante en una página ASP. Es un proceso muy sencillo, ya que existen dos variables de servidor que nos ofrecerán directamente esta información.

Pueden ocurrir dos casos que tenemos que comprobar: 1) que el visitante navegue él mismo sobre la página y 2) que navegue a través de un proxy. Esto lo podemos comprobar de la siguiente manera:

```
' Guardar la IP del visitante
'El visitante puede acceder por proxy, entonces tomo la IP que lo está utilizando
ip = request.servervariables("HTTP_X_FORWARDED_FOR")
'Si no venía de un proxy, tomo la ip del visitante
if ip ="" then
   ip = Request.servervariables("REMOTE_ADDR")
end if
```

Si queremos evitar que nuestras páginas las puedan navegar un visitante con una IP dada, simplemente tenemos que comprobar con un IF que esa IP que queremos bloquear no es la del visitante.

```
if ip = "127.0.0.1" then
```

```
  response.write ("bloqueo")
end if
```

Imaginemos que tenemos una lista de IPs que queremos bloquear. Entonces sería útil colocar un sencillo array de IPs a bloquear y un bucle para recorrerlo, de modo que podamos comprobar si la IP del visitante está en el array de bloqueadas.

```
'creo un array de ips bloqueadas
dim ips_bloqueadas(5)
ips_bloqueadas(0) = "10.10.1.1"
ips_bloqueadas(1) = "103.10.1.21"
ips_bloqueadas(2) = "1.130.41.1"
ips_bloqueadas(3) = "30.105.61.13"
ips_bloqueadas(4) = "102.210.161.1"

'para cada ip bloqueada
for each ip_actual in ips_bloqueadas
  'si la ip del visitante es igual a una de las que hay que bloquear
  if ip = ip_actual then
    response.redirect "explica_bloqueo.html"
end if
next
```

Listado de los archivos de un directorio con ASP

En este taller de ASP, que está englobado dentro del manual de File System Object ASP [http://www.desarrolloweb.com/manuales/45/], vamos a mostrar como obtener un listado de los archivos que hay en una carpeta, dentro del sistema de archivos del servidor.

ASP dispone de una serie de objetos para acceder a cualquier recurso dentro de los discos duros del servidor web. Ahora bien, será necesario que en esos discos duros tengamos permisos para poder trabajar. Posiblemente, si probamos los scripts dentro de nuestro ordenador local, no habrá ningún problema para listar los contenidos de cualquier directorio, pero si lo probamos en un servidor de alojamiento compartido, posiblemente tengamos problemas con los permisos, a no ser que estemos intentando acceder a una carpeta que esté dentro de nuestro directorio de publicación.

Ahora veamos un código para mostrar los archivos contenidos en una carpeta. Primero que todo tenemos que decidir el nombre de la carpeta que deseamos explorar, es decir el directorio físico que deseamos listar sus archivos. Algo como C:\inetpub\wwwroot\

Si estamos haciendo pruebas en nuestro ordenador local, podemos conocer perfectamente el nombre de cualquier carpeta, porque sabemos la estructura de discos y directorios de nuestro ordenador. Pero si estamos en un alojamiento compartido, tendremos que hacer una pequeña acción para saber el nombre del directorio físico donde podemos listar sus archivos.

En local, podemos decidir directamente el nombre de la carpeta:

```
'declaro el nombre de una carpeta
nombre_carpeta = "C:\"
```

En un alojamiento compartido, podríamos obtener así el nombre de la carpeta física donde está el archivo ASP que estamos ejecutando:

```
nombre_carpeta = Server.MapPath(".") & "\"
```

Como se ve, se utiliza el método de Server.mappath() para obtener el nombre de un directorio. A este método se le pasa la ruta relativa del directorio que se desea obtener el nombre físico. En este caso, pasándole ".", nos devolverá el directorio físico donde está el script ASP que estamos ejecutando. Algo como X:\webs\midominio.com\Html\

Una vez tenemos el nombre del directorio que deseamos listar archivos, tenemos que obtener el objeto carpeta de este directorio. Para ello debemos conectar con el sistema de archivos del ordenador:

```
'Conecto con el sistema de archivos
set FSO = server.createObject("Scripting.FileSystemObject")
```

Y luego tenemos que extraer el objeto carpeta del directorio deseado:

```
'creo el objeto carpeta
Set carpeta = FSO.GetFolder(nombre_carpeta)
```

El siguiente paso es sacar todos los archivos de esa carpeta:

```
'traigo los archivos de la carpeta
Set archivos = carpeta.Files
```

Para finalizar, listamos el nombre de cada archivo contenido en la carpeta, mediante un bucle for… each:

```
'para cada archivo, muestro su nombre.
for each nombre_archivo in archivos
   response.Write "<br>" & nombre_archivo
next
```

El ejemplo completo sería algo como lo que sigue:

```
<%
'obtengo el directorio físico de la carpeta donde está este script
nombre_carpeta = Server.MapPath(".") & "\"
response.write "<h1>-" & nombre_carpeta & "</h1>"

'Conecto con el sistema de archivos
set FSO = server.createObject("Scripting.FileSystemObject")

'creo el objeto carpeta
Set carpeta = FSO.GetFolder(nombre_carpeta)

'traigo los archivos de la carpeta
Set archivos = carpeta.Files

'para cada archivo, muestro su nombre.
for each nombre_archivo in archivos
   response.Write "<br>" & nombre_archivo
next
%>
```

Recorrido genérico por un recordset con ASP

En este taller de ASP vamos a realizar una consulta a una tabla de una base de datos para sacar un conjunto de registros. Luego vamos a hacer un recorrido por ese conjunto de registros genérico, sin importar los nombres de los campos y el número de registros o de campos, mostrando todos los datos extraídos en el recordset. La consulta que realicemos sobre la tabla es indiferente, es decir, el script está preparado para realizar el recorrido sea cual sea el conjunto de registros resultante y los nombres de los campos.

El objetivo es mostrar todos los datos del recordset en una tabla. En la primera fila de la tabla colocaremos los nombres de los campos que tiene cada registro y en las siguientes filas, los valores de los campos de cada registro obtenido en la consulta.

Recordemos que las explicaciones sobre conectar y trabajar con una base de datos están en nuestro manual de ASP. [http://www.desarrolloweb.com/manuales/8/]

El primer trozo de código es una conexión con la base de datos por medio de DSN y la ejecución de una sentencia SQL.

```
'conecto BBDD y genero RS
set conn = server.createobject("adodb.connection")
conn.open "miDSN"
sSQL="select * from tabla"
set rs=conn.execute(sSQL)
```

Una vez creado el recordset en la variable "rs", tenemos que escribir la cabecera de la tabla con los nombres de los campos que tiene cada registro recibido.

```
'nombres de los campos como cabecera de la tabla
response.write "<table border='1'><tr>"
for i = 0 to rs.fields.Count - 1
   response.write "<th align='center' bgcolor='#eeeeee'>"
   response.write rs.fields(i).name
    response.write "</th>"
next
response.write "</tr>"
```

En el código anterior se ha realizado un recorrido a cada uno de los campos del primer registro del recordset y para cada uno de ellos se ha escrito su nombre, almacenado en rs.fields (i).name, la propiedad name del campo. Así mostraremos una celda por cada nombre de los campos del registro.

El paso siguiente será mostrar el contenido de cada uno de los registros, realizando un recorrido genérico por todo el recordset.

```
'ahora, para cada registro
while not rs.EOF
    response.write "<tr>"
   'muestro todos los campos que tiene
    for i = 0 to rs.fields.Count - 1
       response.write "<td>" & rs.fields(i) & "</td>"
    next
    response.write "</tr>"
    rs.movenext
wend
response.write "</table>"
```

El bucle "while not rs.EOF" es para recorrer cada registro. Para cada registro se crea una fila. Luego se anida otro bucle "for i = 0 to rs.fields.Count" para recorrer cada campo del registro. Para cada campo se escribe el valor en una columna. Para terminar, se cierra la tabla.

El script ya ha terminado. Sólo nos queda cerrar las conexiones con la base de datos.

```
'cierro la bbdd y el recordset
rs.close
conn.Close
```

Podemos ver el código fuente completo a continuación:

```
<%
'conecto BBDD y genero RS
set conn = server.createobject("adodb.connection")
conn.open "miDSN"
sSQL="select * from tabla"
set rs=conn.execute(sSQL)

'nombres de los campos como cabecera de la tabla
response.write "<table border='1'><tr>"
for i = 0 to rs.fields.Count - 1
   response.write "<th align='center' bgcolor='#eeeeee'>"
    response.write rs.fields(i).name
    response.write "</th>"
next
```

```
response.write "</tr>"
'ahora, para cada registro
while not rs.EOF
    response.write "<tr>"
   'muestro todos los campos que tiene
    for i = 0 to rs.fields.Count - 1
    response.write "<td>" & rs.fields(i) & "</td>"
    next
    response.write "</tr>"
    rs.movenext
wend
response.write "</table>"

'cierro la bbdd y el recordset
rs.close
conn.Close
%>
```

Autores del manual:

Hay que agradecer a diversas personas la dedicación prestada para la creación de este manual. Sus nombres junto con el número de artículos redactados por cada uno son los siguientes:

- **Rubén Alvarez**
  (5 capítulos)
- **Miguel Angel Alvarez**
  Director de DesarrolloWeb.com
  (29 capítulos)
- **Mario Matías Sebely**
  WebMaster ConozcaMisiones.com
  http://www.conozcamisiones.com/
  (1 capítulo)
- **Carlos Luis Cuenca**
  http://www.helloworldsolutions.com/
  (4 capítulos)
- **Luis Marcelo Sosa**
  http://www.i-arg.com.ar/
  (1 capítulo)
- **Fernando Ortiz**
  http://www.laventanita.net/
  (1 capítulo)
- **Clikear.com**
  http://www.clikear.com/
  (1 capítulo)
- **Fabio Núñez Iturriaga**
  Diseñador web
  http://www.nedial.net/
  (2 capítulos)
- **Eugenia Bahit**
  Desarrolladora ASP y PHP
  http://www.cmzk.com.ar/
  (4 capítulos)
- **Mauricio Lemos**
  (1 capítulo)

- **David Barquero**
  (1 capítulo)

- **Pedro Rufo Martín**
  Webmaster de www.asptutor.com
  http://www.asptutor.com/
  (3 capítulos)

- **Eduardo Noriega de Armas**
  http://www.aprendamas.net/
  (1 capítulo)

- **Leonardo Alberto Celis**
  http://www.ohstudio.com.ar/
  (1 capítulo)

Volver [http://www.desarrolloweb.com/manuales/11]