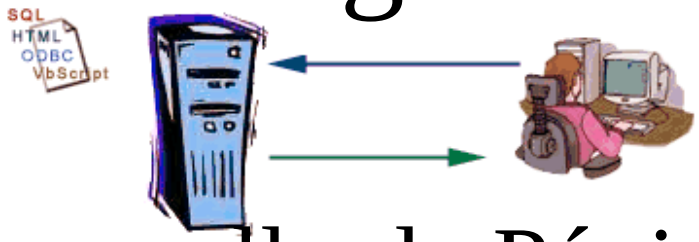
Universidad Técnica Federico Santa Maria
Departamento de Electrónica
Valparaíso – Chile

Trabajo de Investigación



“Desarrollo de Páginas Mediante la Tecnología ASP”

Nombre : Fabrizio Parraguirre Cid
Rol : 9821014-9
Profesor : Agustín González
Fecha Entrega : Miércoles 30 de Octubre de 2002

Introducción

ASP (**Active Server Pages**) es una tecnología desarrollada por Microsoft en diciembre de 1996, para crear páginas Web de contenido dinámico apoyándose en scripts ejecutados en el servidor. Básicamente una página ASP es una mezcla entre una página HTML y un programa que da como resultado una página HTML que es enviada al cliente (navegador).
El principio de la tecnología ASP es el **VBScript**, pero existe otra diversidad de lenguajes de programación que pueden ser utilizados como lo es Perl, JScript, etc.

Es parte del Internet Information Server (IIS) desde la versión 3.0 y es una tecnología de páginas activas que permite el uso de diferentes scripts y componentes en conjunto con el tradicional HTML para mostrar páginas generadas dinámicamente.

Traduciendo la definición de Microsoft: "ASP es un ambiente de aplicación abierto y gratuito en el que se puede combinar código HTML, scripts y componentes ActiveX del servidor para crear soluciones dinámicas y poderosas para el Web".

Resumen

El ASP es una tecnología dinámica funcionando del lado del servidor, lo que significa que cuando el usuario solicita un documento ASP, las instrucciones de programación dentro del script son ejecutadas para enviar al navegador únicamente el código HTML resultante. La ventaja principal de las tecnologías dependientes del servidor radica en la seguridad que tiene el programador sobre su código, ya que éste se encuentra únicamente en los archivos del servidor que al ser solicitado a través del Web, es ejecutado, por lo que los usuario no tienen acceso más que a la página resultante en su navegador. En el presente informe mostraremos los comandos básicos para la creación de páginas utilizando la tecnología ASP.

Creación de Páginas con ASP

Implementación Necesaria

Los archivos .asp son archivos de texto normales, no es necesario ningún editor especial para crearlos, puede usarse cualquier editor que genere código ASCII. Un archivo .asp puede contener texto, código HTML, código ASP o cualquier combinación de estos. Si no contiene código ASP se comporta como un archivo .html normal. Lo importante no es la creación, sino la ejecución dentro del servidor de los scripts, los cuales entregan sus resultados al cliente en forma de html. Es por esto que dentro de la maquina que se va a usar para albergar las páginas se hace necesario cierto software para la ejecución de ASP, como son:

Para la implantación de un servidor Web

· Windows NT 4.0, 2000 o XP

· IIS 4.0 o 5.0 (Internet Information Server 4.0 - 5.0) Ó IIS3.0 + ASP.EXE

O para desarrollo Intranet o local

· Windows 95 + Personal Web Server 1.0 + ASP.EXE

· Windows 98 o Millenium + Personal Web Server 4.0

En caso del uso de un servidor Linux, Chilisoft ha desarrollado el Chilisoft ASP que también permite el uso de esta tecnología.

Programación ASP

ASP es principalmente utilizado sirviéndose del lenguaje Visual Basic Script que no es más que una versión light del Visual Basic. Sin embargo, es posible programar páginas ASP en Java Script. Lo único que hay que hacer es especificar en la propia página qué tipo de lenguaje estamos utilizando. Dado que el lenguaje ASP está muy frecuentemente embebido dentro del código HTML, es importante poder marcar al servidor qué partes están escritas

en un lenguaje y cuáles en otro. Es por ello que todas las partes del archivo que están escritas en ASP se distinguen del resto del texto del archivo mediante delimitadores (*un delimitador es un carácter o secuencia de caracteres que marca el principio o final de una unidad*), en este caso por los símbolos: <% y %>. De este modo, cuando realicemos nuestros scripts, lo primero que debemos definir es el tipo de lenguaje utilizado, lo cual se hace del siguiente modo:

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

Para el caso en el que programemos en Visual Basic Script, que será el que se usará en este trabajo, ya que presenta toda una serie de prestaciones que lo hacen sin duda más accesible y apto para ASP.

Un ejemplo sencillo de un código sería:

```
<HTML>
<BODY>
Hola, bienvenido a mi página, estamos a: <%=Now( )%>
</BODY>
</HTML>
```

La función Now() de VBScript devuelve la fecha y hora actuales. Cuando el servidor Web procese la página nos devolverá el siguiente resultado al explorador:

Hola, bienvenido a mi página, estamos a: 10/30/2002 15:40:00 PM

Sintaxis

Dentro de las características más destacables están:

- No hay distinción entre mayúsculas y minúsculas.
- Las instrucciones terminan con un retorno de carro.
- No es necesario definir las variables antes de usarlas, pero por claridad se hará.
- Las cadenas de texto se delimitan entre comillas dobles. “”
- Los comentarios empiezan con una comilla simple ‘ y terminan al final de línea.

Manejo de Variables y Tipos de Datos

En ASP no es necesario definir las variables antes de usarlas, pero es conveniente su declaración para evitar errores. Esto se realiza con la sentencia Dim. Tampoco tienen tipos, es decir, que una misma variable puede contener un número y luego puede contener caracteres. Aun así, podemos hacer distinciones más precisas acerca de la naturaleza de la información a través de los Subtipos de las variables. Además VBScript pone a nuestra disposición funciones para convertir los datos de un subtipo a otro.

Tipos de Subtipos:

Subtipo	Descripción	Valor de Vartype
Empty	Variable sin inicializar	0
Null	Variable intencionadamente vacia	1
Boolean	Dos valores posibles **True** o **False**	11
Byte	Entero entre 0 y 255	17
Integer	Entero entre -32.768 y 32.768	2
Currency	Numero entre -922.337.203.685.477,5808 y 922.337.203.685.477,5807	6
Long	Numero entre -2.147.483.648 y 2.147.483.647	3
Single	Numero de precisión simple	4
Double	Numero de doble precisión	5
Date	Fecha entre 1-1-100 y 31-12-9999	7
String	Cadena de longitud variable hasta 2.000.000.000 de caracteres.	8
Object	Contiene un Objeto	9
Error	Contiene un numero de error	10

Tabla Nº 1

Todas las funciones de conversión de la Tabla Nº 2 tienen la misma sintaxis:
Función(expresión), siendo *expresión* el dato que se desea convertir.

Cbool	Convierte una expresión a tipo Boolean
Cbyte	Convierte una expresión a tipo Byte
Cint	Convierte una expresión a tipo Integer
Clng	Convierte una expresión a tipo Long
Csng	Convierte una expresión a tipo Single
Cdbl	Convierte una expresión a tipo Double
Ccur	Convierte una expresión a tipo Curency
Cdate	Convierte una expresión a tipo Date
Cstr	Convierte una expresión a tipo String

Tabla Nº 2

Como restricciones de los nombres de variables se tienen: que deben comenzar con un carácter alfabético, no puede contener un punto y no debe superar los 255 caracteres.

La asignación de valores, ya sea numero, texto u otro, se realiza al igual que la mayoría de los lenguajes de programación, con el símbolo igual (=). Los valores de fecha se asignan entre almohadillas (Ejm: fecha_cumple = #12-1-1980#), y los strings con comillas dobles (Ejm: texto = “Hola Mundo”). Las constantes se definen con la sentencia CONST (Ejm: Const cte = 5)

Declaración de Arreglos y Matrices

Se declaran del mismo modo que C/C++, con la diferencia de que los arreglos utilizan paréntesis ( ) en vez de los paréntesis [ ] (Ejm: Dim Arreglo(10)). A diferencia de C/C++, VBScript numera los elementos a partir del 0, lo que implica que un arreglo definido como Arreglo(10) (Caso Ejm.) tendría auténticamente 11 elemento. Las matrices pueden tener hasta 60 dimensiones. Cada una de estas se definen separadas por una coma (Ejm: Dim Matriz(5,10)). También podemos definir arreglos que cambien de tamaño durante la ejecución de la secuencia de comandos (matrices dinámicas o seudo vectores), para ello las declararemos sin poner el número de dimensiones *Dim Arreglo()* y determinaremos las dimensiones con la sentencia Redim: *Redim Arreglo(22).* Si queremos conservar los valores

almacenados en la matriz cuando variamos su tamaño debemos añadir la sentencia Preserve:

```
Redim MiVector(12)
.....
Redim Preserve MiVector(20)
```

Operadores Básicos

Entre los operadores más usuales se encuentran los aritméticos, de comparación y los lógicos. Estos presentan la misma simbología y sintaxis que para C/C++, salvo los del recuadro a la derecha

El operador igual se representa con solo un =, en vez de dos
El operador distinto se representa con <>
El operador lógico y (en C/C++ &&) se representa por and
El operador lógico o (en C/C++ ||) se representa por or
El operador de negación se representa por not

Sentencia Condicionales y Lazos

La estructura base de las instrucciones if then else es la siguiente:

```
IF condición THEN
  Instrucciones
ELSE
  Instrucciones
END IF
```

```
SELECT CASE variable
  CASE "valor 1"
    Instrucciones
  CASE "valor 2"
    Instrucciones
END SELECT
```

Para las sentencias Select Case se tiene la sintaxis expuesta en el recuadro de la Izquierda

La sintaxis, en el recuadro de la derecha, para las sentencias for

```
FOR contador = número inicial TO número final STEP incremento
  Instrucciones
NEXT
```

```
DO WHILE condición
  Instrucciones
LOOP
```

Las sentencias Do While se representan según el recuadro de la Izquierda

Procedimientos y Funciones

Para los procedimientos la sintaxis es la que se observa en el recuadro de la derecha

```
SUB Nombre(parametro1, parametro2,...)
    Instrucciones;
END SUB
```

Para llamar a un procedimiento tenemos dos sintaxis distintas:
Sin paréntesis: Nombre parámetro 1, parámetro 2, …
Con paréntesis: CALL Nombre(parámetro 1, parámetro 2, …)

```
FUNCTION Nombre(parametro1, parametro2,...)
    Instrucciones;
    Nombre = Valor de retorno
END FUNCTION
```

Las funciones son iguales que los procedimientos pero estas nos permiten devolver un valor. Su sintaxis es el del recuadro de la Izquierda

Creación y Uso de Librerías

Para crear una librería con diversos procedimientos, funciones o variables, solo es necesario crear un archivo con extensión asp que contenga todas estas antes mencionadas. La instrucción para incluir una librería en alguna página es:

```
<!--#include file="nombre_de_archivo" -->
```

Objetos integrados de ASP

El ASP es un lenguaje diseñado para la creación de aplicaciones en Internet. Esto quiere decir que existen toda una serie de tareas bastante corrientes a las cuales debe dar un tratamiento fácil y eficaz. Nos referimos por ejemplo al envío de e-mails, acceso a archivos, gestión de variables del cliente o servidor como pueden ser su IP o el lenguaje aceptado...

El lenguaje VB propiamente dicho no da una solución fácil y directa a estas tareas sino que invoca a los denominados objetos que no son más que unos módulos incorporados al lenguaje que permiten el desarrollo de tareas específicas. Estos objetos realizan de una manera sencilla toda una serie de acciones de una complejidad relevante. A partir de una llamada al objeto este realizará la tarea requerida. En cierta forma, estos objetos nos ahorran el tener que hacer largos programas para operaciones sencillas y habituales.

Algunos de estos objetos están incorporados en el propio ASP, otros deben de ser incorporados como si se tratase de componentes accesorios. Por supuesto, no podríamos ejecutar correctamente un script en el cual tuviésemos que llamar a un objeto que no estuviese integrado en el servidor. Este tipo de "plug-in" son generalmente comprados por el servidor a empresas que los desarrollan.

Como todo objeto del mundo real, los objetos del mundo informático tienen sus propiedades que los definen, realizan un cierto número de funciones o métodos y son capaces de responder de una forma definible ante ciertos eventos.

La descripción de la totalidad de objetos con sus métodos y propiedades resulta demasiado extenso. Por esto, se describirán los más frecuentemente utilizados y ejemplificados de la manera más práctica dejando la enumeración exhaustiva para un manual más extenso.

· **Objeto Application:** El objeto Application se utiliza para compartir información entre todos los usuarios de una aplicación. Como varios usuarios pueden compartir un objeto

Application, existen los métodos Lock y Unlock para asegurar la integridad del mismo (varios usuarios no puedan modificar una misma propiedad al mismo tiempo).

Lock

El método Lock asegura que sólo un cliente puede modificar o tener acceso a las variables de Application al mismo tiempo.

Sintaxis: ApplicationLock

Unlock

El método Unlock desbloquea el objeto Application para que pueda ser modificado por otro cliente después de haberse bloqueado mediante el método Lock. Si no se llama a este método de forma explícita, el servidor Web desbloquea el objeto Application cuando el archivo .asp termina o transcurre su tiempo de espera.

Sintaxis: Application.Unlock

Ejemplo:

```
<% Application.Lock
Application("visitas") = Application("visitas")+1
Application.Unlock %>
Eres el visitante numero <%= Application("visitas") %>
```

En el ejemplo anterior el método Lock impide que más de un cliente tenga acceso a la variable visitas al mismo tiempo, mientras que el método Unlock libera el objeto bloqueado de forma que el próximo cliente puede incrementar la variable.

En el objeto Application pueden almacenarse matrices, pero estas son almacenadas como un objeto, es decir, no podemos almacenar o recuperar un solo elemento de la matriz, si no que cargaremos o recuperaremos la variable con la matriz completa.

Ejemplo:

```
<%Dim parametros(2)
parametros(0) = "verde"
parametros(1) = 640
parametros(2) = 480
Application.Lock
Application("Param") =parametros%>
Application.UnLock
```

Con estas instrucciones almacenaríamos TODA la matriz en la variable de aplicación "Param". Para recuperar los valores de la matriz primero recuperamos esta en una variable normal **<%Apliparam=Application("Param")%>.** Ahora podremos operar con los valores de la tabla en las variables Apliparam(0), Apliparam(1) y Apliparam(2).

· **Objeto Request:** El objeto Request se utiliza para tener acceso a la información que se pasa en las peticiones HTTP. Entre dicha información se incluyen los parámetros que se pasan desde los formularios HTML mediante el método POST o el método GET, cookies y certificados de cliente. Dependiendo de la forma en que enviemos los datos al servidor tendremos que utilizar una u otra de las diversas colecciones del objeto Request. Las más típicas son:

Form: recupera datos enviados desde un formulario mediante el método POST.
Querystring: recupera datos enviados como cadena de consulta HTTP.
Cookies: recupera los valores de las Cookies.
Sintaxis General: Request.coleccion(elemento)

Ejemplos:

Form

Supongamos que enviamos la información desde el siguiente formulario:

```
<form method="POST" action="recibir.asp" >
<p>Nombre: <input type="text" name="Nombre" size="20"></p>
<p>Nacionalidad: <input type="text" name="Nacionalidad" size="20"></p>
<p><input type="submit" value="Enviar" name="Enviar"></p>
</form>
```

```
Hola Sr/a <%=request.form("nombre")%> <br>
Asi que usted es de nacionalidad <%=request.form("nacionalidad")%>
```

En nuestra página "recibir.asp" podríamos usar la secuencia Izquierda

Con lo que el resultado seria ✍

```
Hola Sr/a Julian
Asi que usted es de nacionalidad francesa
```

Querystring

Supongamos que enviamos la información en forma de cadena de consulta (Notar que una cadena de consulta HTTP esta especificada por las parejas de valores que siguen al signo "?"): **<a href="recibir.asp?nombre=Julian&nacionalidad=francesa">**.

En nuestra página recibir.asp" podríamos utilizar lo siguiente:

```
Hola Sr/a <%=request.querystring("nombre")%> <br>
Asi que usted es de nacionalidad <%=request.querystring("nacionalidad")%>
```

Con lo que el resultado sería igual al anterior

· **Objeto Response:** El objeto Response se utiliza para controlar la información que se envía al usuario. Esto incluye el envío de información directamente al explorador, la redirección del explorador a otra dirección URL o el establecimiento de valores de las cookies.

Sintaxis general: Response.metodo [valor]

Entre los métodos mas interesantes del objeto Response están los siguientes:

Write

El método Write escribe una cadena de resultado en el navegador cliente (Nota: cuando se usa la sintaxis <%=variable%> estamos usando implícitamente el método Response.Write).

Ejemplo:

```
<%response.write "<center>Hola mundo</center>" %>
```

Obtenemos: Hola mundo

Redirect

El método Redirect hace que el explorador se conecte con una dirección URL diferente. *Nota: debemos usar este método antes de enviar cualquier resultado al navegador cliente, en caso contrario produce un erro*r.

Ejemplo:

```
<%response.redirect "www.renfe.es"%>
```

El navegador se dirigirá a la URL especificada

· **Objeto Server:** El objeto Server proporciona acceso a los métodos y las propiedades del servidor. El método utilizado más frecuentemente es el que crea una instancia de un componente ActiveX (Server.CreateObject).

ScriptTimeout

Especifica la cantidad máxima de tiempo que puede tardar la ejecución de una secuencia de comandos (Tiempo máximo que puede tardar en ejecutarse una página dada).

Sintaxis: Server.ScriptTimeout= nº de segundos

Ejemplo:

```
<% Server.ScriptTimeout=120 %>
```

La página puede ejecutarse durante 120 segundos antes de que el servidor la termine.

CreateObject

Crea una instancia de un componente ActiveX en el servidor.

Sintaxis: Server.CreateObject (IdProg)

IdProg es el identificativo del tipo de componente que queremos crear, nos viene suministrado por el fabricante del componente.

Ejemplo:

```
<% set Mitabla = CreateObject("ADODB.Recordset") %>
```

Instancia un objeto de tipo recordset y lo asigna a la variable "Mitabla".

· **Objeto Session:** El objeto Session permite almacenar la información necesaria para una determinada sesión de usuario. Las variables almacenadas en el objeto Session no se descartan cuando el usuario pasa de una página a otra dentro de la aplicación, si no que dichas variables persisten durante todo el tiempo que el usuario tiene acceso a las páginas de la aplicación. También puede utilizar los métodos de Session para terminar explícitamente una sesión y establecer el periodo de tiempo de espera de inactividad de las sesiones.

Creación de una variable en Session
Sintaxis: Sesion("Nomvariable")= valor

Ejemplo: <% Session("Color")="Rojo" %>
Para recuperar ese valor: <% ColorFavorito=Session("Color") %>
Esto nos almacenaría el valor "rojo" en la variable "ColorFavorito"

En el objeto Session pueden almacenarse matrices, pero estas son almacenadas como un objeto, es decir, no podemos almacenar o recuperar un solo elemento de la matriz, si no que cargaremos o recuperaremos la variable con la matriz completa

Ejemplo:

```
<%Dim cestacompra(2)
cestacompra(0) = 1
cestacompra(1) = 8
cestacompra(2) = 22
Session("Cesta") =cestacompra%>
```

Con estas instrucciones almacenaríamos TODA la matriz en la variable de sesión "Cesta".Para recuperar los valores de la matriz primero recuperamos esta en una variable normal: <%Micesta=Session("Cesta")%>. Ahora podremos operar con los valores de la tabla en las variables Micesta(0), Micesta(1) y Micesta(2).

Abandon
Destruye todos los objetos y variables almacenados en el objeto Session.
Ejemplo:<% Session.Abandon %>

Conclusiones

Como conclusión en primera instancia debo decir que esta tecnología desarrollada por Microsoft es una herramienta poderosa y extensa para el desarrollo dinámico de páginas, y su interacción con el cliente. Lamentablemente, es tanto lo que se puede mencionar, que este trabajo es más bien un pequeño resumen que un manual, que menciona de manera introductoria y básica el desarrollo de paginas mediante esta técnica. No obstante, establece los cimientos necesarios para que cualquier persona sin grandes conocimientos de ASP, pero con el esquema de pensamiento de algún lenguaje de programación como lo es C/C++, pueda aventurarse y dar los primeros pasos, para posteriormente profundizarse en el tema.

Bibliografía y referencias

Desde Internet:

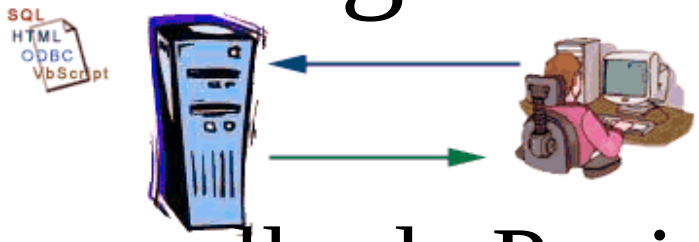
· ASPTutor.com
http://www.asptutor.com/asp/default.asp
· DesarrolloWeb.com
http://www.desarrolloweb.com/articulos/244.php?manual=8
· MaestrosDelWeb.com
http://www.maestrosdelweb.com/editorial/tecnologias/asp.asp
· WebEstilo.com
http://www.webestilo.com/asp/

Textos y Otros:

· Active Server Pages 2.0 - Editorial WROX

Universidad Técnica Federico Santa Maria
Departamento de Electrónica
Valparaíso - Chile

Trabajo de Investigación



“Desarrollo de Paginas mediante la tecnología ASP”

Anexos

Instalación del Personal Web Server

El PWS 4.0 es un servidor Web para Windows 95/98 o NT Workstation con capacidad para el uso de ASP. Es una versión reducida del servidor IIS (Internet Information Server) para máquinas corriendo Windows NT Server.

Hay que tomar muy en cuenta que existen diferencias entre el PWS para el NT Workstation y para el Win95/98 por lo que hay que estar muy atento a estos detalles. La versión para WinNT es más completa e incluye acceso FTP mientras que la versión para Win95/98 es para uso personal y por eso no incluye esta característica entre otras.

Pasando a la práctica, para correr el PWS es necesario el Internet Explorer 4.01 o mayor.

Para instalarlo, procedemos a descargar el NT Option Pack desde el sitio de Microsoft. Hay que tomar muy en cuenta que el paquete se llama NT Option Pack pero no es precisamente para NT pues también existe la opción para Win95/98.

Descargar este programa puede ser una tarea pesada pues el download completo es de 31Mb. Los usuarios de Windows 98 pueden evitar la descarga de este programa, tomándolo del CD de Windows98. Para llegar hasta el hay que dirigirse a Inicio > Configuración > Control Panel > Agregar/Quitar Programas > Internet > Personal Web Server. Esto instalará un icono en nuestro sistema para proceder a instalar el programa. También podemos acceder directamente al instalador desde el CD en D:\add-ons\pws\setup.exe (Donde D:\ es la asignación de nuestro CD-ROM)

Al ejecutar el instalador primero detectará si tenemos instalado Winsock 2.0. En caso de que no lo tengamos, este será instalado automáticamente. A continuación, el asistente nos preguntará sobre las herramientas a instalar. Podemos seleccionar la opción típica o escoger bien las opciones en la opción personalizada. Para finalizar el asistente nos pedirá rebotar el sistema.

Luego, al iniciar nos debería de aparecer el icono del PWS en nuestra barra de programas.

Para revisar que la instalación sea correcta podemos abrir nuestro navegador e ingresar el URL http://nombre_maquina (Donde nombre_maquina es el nombre de la máquina en la red.). Si no estamos seguros del nombre podemos usar el IP http://127.0.0.1/ para ingresar. Esto nos llevará a la página de bienvenida del PWS.

El directorio predeterminado del PWS para guardar nuestros documentos es C:\Inetpub \wwwroot (Donde C:\ es la asignación de nuestro disco duro principal)

En este directorio podemos incluir un archivo .asp para evaluar si el soporte de ASP esta funcional.

En versiones anteriores del PWS se incluía un modulo instalador llamado asp.exe que permitía el uso de ASP en el PWS. En la última versión no es necesario utilizarlo y deberá evitarse el uso de este archivo pues puede dañar el PWS.

Una vez completada nuestra instalación y confirmado su funcionamiento, ya tenemos la plataforma completa para desarrollar nuestros ASP's de forma local y sin recurrir a servidores de terceros.

Hint: Muchas personas resultan con un error al instalar el PWS que hace mención de fallo en la instalación del MTS. La solución es que hay que bajarse la nueva versión de mtssetup.dll. Luego hay que copiar los instaladores del PWS al disco duro y reemplazar esta dll, para luego instalar el PWS. Si lo habían instalado antes, deben desinstalarlo antes de realizar este proceso.

Páginas Dinámicas vs. HTML

A pesar de que las páginas dinámicas nos puedan en un principio limitar a causa de su mayor complejidad con respecto al HTML, todas las ventajas que nos ofrecen compensan con creces este esfuerzo inicial.

No obstante, hay que ser consciente del posible interés que pueda tener para uno el lanzarse en esta aventura de aprender un nuevo lenguaje y volver a rediseñar su propio sitio.

Si la página que queremos desarrollar o que queremos reestructurar es relativamente pequeña, no necesita estar al día continuamente sino que sus contenidos son perennes y no tenemos previsto el pagar por mantenerla, el empleo de páginas dinámicas puede estar de más y resultar improductivo.

Por el contrario, si el sitio es extenso y sus contenidos cambian rápidamente, nos interesa el automatizar en la medida de lo posible todas las tareas de tal forma que podamos gestionar su explotación de la manera más óptima.

Para dejar más claro hasta que punto resulta útil utilizar páginas dinámicas lo mejor será ejemplificarlo a partir de un sitio Web modelo.

Supongamos que hemos decidido realizar un portal de televisión donde una de las informaciones principales a proveer podría ser la programación semanal. Efectivamente, esta información suele ser dada por las televisiones con meses de antelación y podría ser muy fácilmente almacenada en una base de datos. Si trabajásemos con páginas HTML, tendríamos que construir una página independiente para cada semana en la cual introduciríamos "a mano" cada uno de los programas de cada una de las cadenas. Asimismo, cada semana nos tendríamos que acordar de descolgar la página de la semana pasada y colgar la de la anterior. Todo esto podría ser fácilmente resuelto mediante páginas dinámicas. En este caso, lo que haríamos sería crear un programa (solo uno) que se encargaría de recoger de la base de datos de la programación aquellos programas que son retransmitidos en las fechas que nos interesan y de confeccionar una página donde aparecerían ordenados por cadena y por hora de retransmisión. De este modo, podemos

automatizar un proceso y desentendernos de un aspecto de la página por unos meses.

Este hecho lo podríamos aplicar a otras situaciones: podemos preparar el horóscopo de todos los días, las promociones de un sitio de e-comercio...

Además, tampoco resultaría complicado el introducir una pequeña caja de búsqueda que nos permitiera dar rápidamente con el programa que queremos ver, saber a qué hora y en qué cadena se emite.

Volviendo a nuestro portal de televisión, en él hay una sección en la cual presentamos todas las series actualmente emitidas con comentarios sobre ella, fotos, etc. Podríamos, en lugar de hacer una página HTML por serie, hacer una única página dinámica en contacto con una base de datos en la cual visualizamos las fotos y comentarios relativos a la serie que nos interesa. Asimismo, si lo que buscamos es modificar el formato del texto de dicha sección, podemos automatizar este proceso sin necesidad de cambiar a mano cada una de las etiquetas font y sin hacer uso de la hojas de estilo las cuales no son reconocidas por la totalidad de los navegadores.

Otra serie de aspectos tales como la gestión de las lenguas, podrían ser fácilmente resueltos sin para ello duplicar el número de páginas y buscar los textos a traducir penosamente entre el código HTML.

En realidad, a partir de estas herramientas, podemos plantearnos cuantas cosas queramos. Los únicos limitantes son la imaginación y deseo de aprender.