<Introducción a HTML y Javascript>

</Jorge Juan Gómez Basanta>

El lenguaje HTML (Hyper Text Markup Language) es un formato muy simple para crear documentos de hipertexto que pueden ser visualizados en múltiples plataformas (Macintosh, PC, Unix). Así al dar cierto formato a la información, nos aseguramos de que cualquier usuario de una computadora personal pueda verla, incluyendo elementos como imágenes, audio, video, e incluso programas completos.

HTML fue creado en 1990 y es la base del World Wide Web (WWW), la parte gráfica de Internet. Antes de ese año, en Internet sólo se podían transmitir textos y programas a través de servicios como FTP y Gopher, lo que limitaba su uso a círculos científicos de diversas universidades a nivel mundial.

A medida que Internet se populariza, más personas se encuentran con la necesidad de introducir páginas en la supercarretera de la información, lo que crea nuevas posibilidades para los diseñadores. Internet no es algo estático, y a medida que crecen las necesidades de los usuarios están apareciendo nuevas tecnologías para añadir elementos multimedia a las páginas como Shockwave y Quicktime.

La creación de páginas

La mayoría de las personas que han navegado en Internet sienten cierta curiosidad por poner a disposición de todo el mundo algún tipo de información. Aunque actualmente existen sofisticados programas que aseguran hacernos prescindir del conocimiento del lenguaje HTML, la experiencia demuestra que estos programas cometen errores de formato o agregan instrucciones innecesarias en una página, por lo que el conocimiento del lenguaje es aun altamente recomendable.

Una de las principales características de una página y todos sus elementos es el espacio que ocupa en disco, que debe ser mínimo, no tanto por las limitantes del servidor, sino por el tiempo que que tarda en transmitirse esta información a través de una conexión por modem. Este es uno de los principales factores que se deberán tomar en cuenta al diseñar una página, con el fin de evitar que se desesperen los lectores.

Lo básico

Una página de Internet es simplemente un archivo de texto que puede ser escrito en cualquier procesador de palabras. Existen algunos programas que facilitan la creación de páginas.

La gran mayoría de las instrucciones que controlan la apariencia y los elementos de una páginas encierran entre corchetes angulados <>. A estas instrucciones se les denomina tags, y son invisibles para el lector. Los tags afectan una determinada porción del texto o a algún otro elemento de una página. También pueden definir un nuevo elemento de la página, como en el caso de una imágen o una línea horizontal. Por este motivo, cuando la etiqueta o tag sirve para modificar a algún elemento de la página, es necesario indicar donde empieza y donde termina su acción. Por ejemplo, para poner un texto en bold, escribiríamos: <B>Bold</B>. El tag que cierra la acción es siempre idéntico a la primera

palabra del tag que abre la acción, salvo que es precedido por una diagonal /. Otro aspecto importante de los tags es que pueden contar con varios atributos, que se escriben después del tag, modificando su forma de actuar. El orden en que aparezcan dichos atributos no tiene relevancia, solo deben estar escritos con una sintaxis perfecta y separados por espacios. No es neceario incluir todos los atributos de cada instrucción, sino sólo los que queremos modificar, los demás adquieren valores por omisión que generalmente funcionan. He aquí un ejemplo:

```
<IMG SRC="navegacion.gif" WIDTH=185 HEIGHT=20 ISMAP BORDER=3
ALIGN=RIGHT> similar a: <IMG SRC="navegacion.gif" HEIGHT=20
ISMAP ALIGN=RIGHT WIDTH=185 BORDER=3 >
```

Tags fundamentales de una página:

<HTML></HTML>

Especifica el lenguaje en el que está escrita la página, en este caso HTML. Algunas personas incluyen la versión de HTML en la que trabajan. La última que fue reconocida como oficial fue la versión 3.2, pero algunos fabricantes de programas para navegación han inventado instrucciones nuevas, que permiten un mejor diseño de la página, pero que no son soportadas por todos los browsers o navegadores.

<HEAD></HEAD>

Controla los elementos de cabeza de la página, como son el título, la base y las especificaciones META, útiles para que algunos robots pertenecientes a sistemas de búsqueda obtengan más información sobre alguna página. Es aquí donde también se definen las funciones que se utilizarán cuando se incluye programación en Javascript, con el fin de que se carguen antes que cualquier otro elemento de la página, eliminando así la posibilidad de que marque un error de función no definida. (Este tema se tratará con más detalle en la Parte 2: Javascript)

<BASE></BASE>

Este elemento opcional provee de una dirección base para interpretar URLs relativos cuando el documento se lee fuera de su contexto habitual.
Base Address: BASE. (Ver la sección de links)

Un elemento META es un contenedor que incluye información especializada sobre eldoumento. Tiene dos funciones básicas: proveer información sobre el acceso a información especial, o incluir información sobre el contenido, la calidad y las características de la página que estamos creando.
Atributos:

•HTTP EQUIV crea el nexo entre un elemento y la respuesta que se debe dar sobre este elemento cuando sea solicitada por un sercidor HTTP
•NAME especifica el nombre del elemento META
•CONTENT indica el valor de dicho elemento
Ejemplos:
<META NAME="Author" CONTENT="Jorge Juan Gómez Basanta">

<META NAME="Generator" CONTENT="User-Agent: 3.0Gold (Macintosh; I; 68K)">

`<TITLE></TITLE>`

Aquí se escribe el texto que servirá como título de la ventana del browsero o navegador.

`<BODY></BODY>`

Sirve para especificar el cuerpo de la página, que incluye todos los elementos que el usuario final encontrará.

Atributos:

•BGCOLOR indica el color de fondo. Si se quiere utilizar una textura o una imagen como fondo, esta instrucción se reemplaza por BACKGROUND.

Ejemplo 1: BGCOLOR=darkblue. La lista de los colores que se pueden usar aparece en el Apéndice A.

Ejemplo 2: BACKGROUND="pics/fondo.jpg". Es importante especificar bien la ruta de acceso al archivo, es decir, los directorios o carpetas en que está contenido este, de manera jerárquica, con respecto a la localización del archivo HTML que contiene dicha referencia a la textura de fondo.

•TEXT, LINK, VLINK, ALINK especifican los colores del texto (TEXT), de los vínculos o links (LINK), de los links que han sido visitados (VLINK) o de los links activos (ALINK) (cuando se mantiene presionado el botón del mouse sobre un link).

Diferentes opciones para acomodar el texto

A continuación se describen los diferentes estilos que se pueden utilizar para cualquier texto.

Parpadeo (Blink)	<BLINK></BLINK>
Negritas (Bold)	<B></B>
Mucho Enfasis (Strong Emphasis)	<STRONG></STRONG>
Cita	<CITE></CITE>
Itálicas	<I></I>
Variable	<VAR></VAR>
Código	<CODE></CODE>
Definición	<DFN></DFN>
Monoespaciadas	<TT></TT>
Teclado	<KBD></KBD>
Ejemplo	<SAMP></SAMP>
Subrayado	<U></U>
Tachado	<STRIKE></STRIKE>
Subíndice	
Superíndice	

Opciones para el párrafo:

Cabeceras o Headers. Existen 6 tamaños de letra básicos, que van de Header 1 <H1><H1> a Header 6 <H6></H6>, siendo <H1> el más grande y <H6> el más pequeño.

Otras instrucciones que dan formato al texto son:

Dirección (Address)	<ADDRESS></ADDRESS>
Texto formateado	<PRE></PRE>
Elemento de una lista	<LI></LI>
Título de descripción	<DT></DT>
Texto de descripción	<DD></DD>
Block Quote	<BLOCKQUOTE></BLOCKQUOTE>

Alineación

•Center Line <CENTER></CENTER> Sirve para centrar imágenes o texto.
•Center Paragraph <P ALIGN=CENTER,LEFT,RIGHT></P> Esta es otra manera de centrar un párrafo determinado.
La instrucción ALIGN es uno de los atributos más frecuentes en instrucciones complejas, tales como las que definen imágenes o tablas.

Espacio entre líneas

•Retorno (Line Break)
 Es equivalente a escribir un retorno en una procesador de palabras
•Line Break <BR CLEAR=LEFT,RIGHT> sirve el texto al lado de las imágenes. Al poner <BR CLEAR=LEFT> indicamos que todo el espacio a la izquierda de la imagen lo queremos libre, y el texto se puede colocar ahí.
•Paragraph Break<P> es equivalente a dejar un doble espacio entre dos renglones o dos imágenes.
•No Line Break <NOBR></NOBR> Las palabras encerradas entre esta instrucción quedarán escritas en un solo renglón.

La instrucción FONT **<FONT></FONT>**

Esta instrucción es relativamente nueva para HTML. Permite determinar el tamaño, color y tipo de letra para una porción del texto, anulando el color definido en el cuerpo (BODY) de la página. El atributo de esta instrucción que cambia el tipo de letra (FACE) solo funciona hasta el momento en el Netscape Navigator 3.0.

Atributos:
•Tamaño SIZE=-2, -1, 0, +1, +2, +3, +4. El tamaño de la letra también puede ser especificado con <FONT SIZE=+4>, ofreciendo gran variedad de tamaños adicionales a los presentes en las instrucciones <H1></H1> a <H6></H6>.
•Color COLOR=forestgreen Se puede utilizar cualquier color de la lista antes mencionada o su correspondiente código hexadecimal.

•Tipo de letra FONT="Helvetica,Arial" Esta instrucción tiene dos parámetros separados por una coma. El primero indica el nombre del tipo de letra para la plataforma Macintosh. El segundo es para las Pcs con Windows.

Ejemplo:

```
<FONT SIZE=+3 COLOR=crimson FACE="Apple Garamond
bk,Bookman"></FONT>
```

Comentarios:

Si se desea hacer un comentario sobre alguna parte de la estructura de la página, para referencia en el futuro, se puede poner del siguiente modo, sin afectar a la página.

```
<!–comentarios–>
```

Sangrías

Para dejar un espacio a modo de sangria antes de la primera oración de un párrafo, se agrega .

Caracteres especiales

Los caracteres especiales, tales como las vocales acentuadas, los símbolos de admiración e interrogación y las comillas elegantes, se escriben con los llamados códigos de escape, debido a que no todas las computadoras manejan de igual manera los acentos y otros símbolos. Así, el código de escape para la letra ú es ú. Para el símbolo ¡ es &#161, y para el símbolo ¿ es ¿. La mayoría de las programas para crear páginas han previsto esta inconveniencia, por lo que sustituyen los caracteres especiales por códigos, por lo cual no es necesario memorizarlos ni incluirlos en este manual.

LOS LINKS O VINCULOS: La instrucción ANCHOR <A></A>

Los links son aquellas palabras o imágenes que hacen posible la existencia de páginas de hipertexto, navegando así de una página a otra con solo presionar el botón del mouse. Constituyen la manera más común de relacionar dos o más páginas, o la relación entre diferentes partes de una misma página. He aquí la sintáxis típica de la instrucción:

```
<A HREF="ruta de acceso/archivo.html">Texto</A>
```

El color del texto encerrado en esta instrucción está definido por LINK=color, en el cuerpo o body de la página. La ruta de acceso a un archivo puede ser de dos tipos:

•Absolutas: Se refieren a direcciones de internet completas (URLs).Ejemplo:

```
<A HREF="http://home.netscape.com">Netscape</A>
```

•Relativas: Se refieren a un documento que se encuentra en algún directorio del servidor. En este tipo de rutas en necesario empezar a considerar la estructura de directorios a partir del lugar en el disco duro en el que se encuentra la página que estamos creando. Es decir, si nuestra página se encuentra en una carpeta o directorio llamado empresas, y a

este mismo nivel existe una carpeta llamada empleados donde existe un documento al que nos interesa referirnos mediante un vínculo, la ruta de la página será: empleados/juanperez.html.

Ejemplo:

```
<A HREF="empleados/juanperez.html">Juan P&eacute;rez</A>
```

Si la página a la que queremos referirnos se encuentra en un directorio superior, es necesario subir a la raiz del servidor, y a partir de ahí entrar a los directorios necesarios hasta alcanzar el archivo deseado. Esto se logra anteponiendo una diagonal a la ruta de acceso, que indica que debe subir a la raiz del servidor. Así:

```
<A HREF="/negocios/empresas/empleados/pedro.html">Pedro</A>
```

Referencias dentro de un mismo documento

Otro uso para la instrucción ANCHOR <A></A> son las referencias en un mismo documento. Son útiles por ejemplo para colocar un índice en la parte superior de la página, creado por hipertexto. Al momento de presionar cualquiera de las opciones del índice podremos acceder a la información sobre dicha opción, que se encuentra en el mismo documento.

Primero es necesario definir el target en la página, es decir la información sobre la opción antes mencionada. Esta información deberá encerrarse en una estructura como sigue:

```
<A NAME="osos">Informaci&oacute;n sobre osos polares. Caracte-
rísticas de esta especie...</A>
```

A esta parte se le denomina target, objetivo.

Posteriormente se hace la referencia en el índice (de acuerdo con el ejemplo) haciendo una referencia al target, de la manera siguiente:

```
<A HREF="#osos">1. Los osos polares</A>
```

Sobre los nombres de archivos

En todas las instrucciones que involucran la inclusión de los nombres de archivos (de cualquier tipo: imágenes, archivos HTML, archivos multimedia...) o directorios es muy importante mantener los nombres idénticos, con una total correlación entre el nombre del archivo y el nombre que aparece en la referencia que hagamos. Es importante considerar que algunos sistemas operativos son sensibles al uso de mayúsculas y minúsculas, es decir una "a" minúscula es diferente a una mayúscula. Es importante también no incluir espacios en los nombres de archivo. En particular, los servidores con sistema operativo UNIX (que son los más usados como servidores de Internet por su eficiencia) poseen esta característica.

Otra recomendación importante es agregarles una extensión a todos los archivos que planeamos colocar en un servidor. Las extensiones que se utilizan como standard para los documentos comunes en Internet son:

.html	para archivos en Hypertext Markup Language
.gif	para imágenes comprimidas como GIF o GIF89

.jpg o .jpeg	para imágenes con compresión JPG
.wav	para archivos de sonido en formato Windows WAVE
.aif o .aiff	para sonidos en formato para Macintosh
.au	para archivos de sonido en formato de SUN
.pdf	para archivos de Adobe Acrobat
.txt	para archivos de texto
.mov	para películas en formato MOV
.qt	para películas Quicktime
.hqx, .sit, .sea	para archivos comprimidos para Macintosh
.zip	para archivos comprimidos para PC
.gzip, .tar	para archivos comprimidos para UNIX
.ram, .ra	para archivos de Real Audio
.mpeg	para archivos comprimidos con este formato

Tipos de archivos

Las imágenes que se colocan en Internet sólo pueden ser de tres formatos básicos, gif, jpg o png. Este último no será tratado en este manual.

Formato GIF: El Graphics Interchange Format (GIF) es comúnmente usado para la transferencia de domentos entre diferentes plataformas por su versatilidad, tanto en tamaño como en aplicaciones que lo reconocen. Es un formato de alta compresión por lo que minimiza los tiempos de transferencia a través de líneas telefónicas. Este formato soporta resoluciones de imágen de 1 bit a 8 bits, es decir hasta 256 colores. Se recomienda comprimir arte de línea y otras imágenes que utilicen pocos colores con este formato, debido a su modo de actuar. Mientrás más zonas de color similar continuas (horizontalmente) existan en un documento, la compresión será mejor, y el archivo ocupará muy poco espacio.

Formato JPEG: Las siglas de este formato significan Joint Photographic Experts Group. Este formato economiza la manera en que la información de una imagen es guardada, descartando información basura. Cuando se comprime una imágen con este formato, la imagen ya no será la misma, habrá perdido valiosa información por lo que no es recomendable editar una imagen después de haberla comprimido con este formato. Generalmente la diferencia entre la imagen original y la imagen comprimida no es perceptible para el ojo humano, pero si existe. Este método de compresión ofrece diferentes opciones en cuanto a la relación calidad y tamaño que ocupa en disco, comprimiendo de radios desde 5:1 hasta 15:1. Este formato es capaz de guardar información para imágenes de 24 bits, es decir para 16 millones de colores, por lo que produce una calidad más alta con respecto al formato GIF, aunque no siempre un menor tamaño. Se recomienda comprimir fotografías y dibujos complejos con este formato.

Inclusión de imágenes en una página: La instrucción <IMG>

Lo primero que destaca en esta instrucción es que NO necesita </IMG>, ya que no es un tag modificador, ya que define un elemento nuevo de la página.

Atributos:

Fuente (Source) SRC: este atributo es el único esencial para IMG, ya que define la ruta de acceso y el nombre del archivo que será empleado en la página. Se siguen los mismos lineamientos empleados en la instrucción ANCHOR <A></A>, para definir la ubicación de la imágen en el disco.

Representaciones alternativas:

Algunas personas que consideran que su navegación por internet es demasiado lenta, ya sea debido al modem que usan, a su proveedor de acceso o a su línea telefónica, prefieren navegar sin imágenes, por lo que es importante incluir una representación alternativa para las imágenes que incluyamos en una página. Esta representación alternativa es opcional, pero muy útil, y puede ser un texto, una imágen o ambas.

Texto Alternativo ALT="texto". Se incluye dentro de la instrucción <IMG> y generalmente se refiere a la imágen a la que está sustituyendo.

Representación gráfica alternativa LOWSRC="ruta de acceso y nombre del archivo". Sustituye a la imágen original con una versión más pequeña de la imagen, o de menor calidad. En concreto, debe ocupar poco espacio en disco, esto es alrededor de 5K.

Espacio vertical (Vertical Space) VSPACE=#. Asigna un valor en pixeles para el espacio alrededor de la imagen, a la izquierda y a la derecha.

Espacio horizontal (Horizontal Space) HSPACE=#. Asigna un valor en pixeles para el espacio alrededor de la imagen, en la parte superior e inferior.

Altura HEIGHT=#. Indica la altura real de la imagen. Se puede poner cualquier valor, pero se distorsiona la imagen.
Ancho WIDTH=#. Indica el ancho real de la imagen. Se puede poner cualquier valor, pero no se recomendable, ya que la imagen se distorsiona.
No se indispensable incluir las dimensiones de la imagen, pero si es muy recomendable. Al incluir las dimensiones, el texto de la página puede aparecer aunque las imágenes no se hayan terminado de cargar. Así, los elementos de la página se acomodan y el usuario no se desespera.

Alineación ALIGN=TEXTTOP, CENTER, ABSCENTER, BOTTOM, LEFT, RIGHT. Sirve para alinear el texto alrededor de la imagen. Las últimas dos opciones, LEFT y RIGHT, alinean la imagen a la izquierda o a la derecha de la pantalla, y el texto aparece a la derecha o izquierda, respectivamente.
Si lo que se desea es que el texto fluya perfectamente alrededor de la imagen, como en los periódicos y revistas, se debe determinar la alineación con LEFT o RIGHT. Después es necesario incluir la instrucción <BR CLEAR=LEFT> o <BR

CLEAR=RIGHT>, con la que se especifica que el espacio a la izquierda o derecha de la imagen debe permanecer vacio, permitiendo que el texto fluya a la perfección. Concretamente, las combinaciones adecuadas para conseguir esto son:

```
<IMG SRC="xxxx.gif" ALIGN=LEFT><BR CLEAR=RIGHT>
<IMG SRC="xxxx.gif" ALIGN=RIGHT><BR CLEAR=LEFT>
```

Se aplican las combinaciones contrarias si se busca que el texto NO fluya alrededor de la imagen.

```
<IMG SRC="xxxx.gif" ALIGN=LEFT><BR CLEAR=LEFT>
<IMG SRC="xxxx.gif" ALIGN=RIGHT><BR CLEAR=RIGHT>
```

ISMAP determina si la imagen es un mapa. Los mapas son aquellas imagenes en las que cada porción de la imagen puede servir como hipertexto, teniendo una sola imagen en la que al presionar en determinada porción nos conduce a diferente página. ISMAP se incluye en las imágenes que sirven como mapas del lado del servidor, aquellos en que la descripción del mapa es un archivo externo, .map. Este tema se tratará con más calma más adelante.

USEMAP=#nombre. Se utiliza en los mapas del lado del cliente, en los que la descripción del mapa está presente en la misma página HTML que la imagen que lo incluye.

BORDER=# Indica el grueso del borde alrededor de la imagen en pixeles. Por defecto el borde es cero, siempre y cuando la imagen no sea un LINK.

Cabe aclarar que una imagen puede servir como botón al encerrarla en una instrucción ANCHOR <A></A>. Ejemplo:

```
<A HREF="resultados.html"><IMG SRC="result.gif" ALIGN=LEFT
BORDER=0></A>
```

Ejemplo de una instrucción <IMG> con la mayoría de sus atributos.

```
<IMG SRC="Logo.gif" LOWSRC="LogoLow.gif" ALT="logotipo"
HSPACE=5 VSPACE=10 BORDER=5 HEIGHT=50 WIDTH=60 ALIGN=RIGHT
ISMAP>
```

Las listas

Las listas permiten acomodar los elementos de una página para darle una apariencia más profesional. Existen varios tipos de listas.

Listas sin orden:

<UL></UL>. Unordered Lists. Indica que lo que sigue son elementos de una lista sin orden, cada uno de ellos precedido por <LI>. Por defecto, las cada elemento de la lista aparece antecedido por un círculo relleno. Este círculo se puede cambiar, por un cuadro o por un disco. Esto se logra definiendo la lista con su atributo TYPE=square, TYPE=circle, TYPE=disc.

Ejemplo:

```
<H1>Lista de animales en extinci&oacute;n</H1>
<UL TYPE=square>
<LI>Oso polar
<LI>Tigre blanco
<LI>Guacamaya
</UL>
```

Orden jerárquico en listas

Las listas pueden tener varios niveles. Esto se logra al incluir una instrucción <UL> dentro de otra instrucción <UL>. La lista que se encuentra adentro de la otra aparece indentada, con una sangía, lo que permite lograr una apariencia mejor.

Listas ordenadas

Las listas ordenadas son aquellas que tienen un orden fijo, es decir, cada elemento es antecedido por una letra o número que le da cierta posición. Las listas ordenadas se definen con la etiqueta Ordered Lists <OL></OL>. De manera similar a las listas sin orden, cada elemento se define con <LI>, sin necesidad de cerrar la instrucción. Por defecto las listas ordenadas aparecen con números arábigos, pero pueden ser modificadas, al utilizar el atributo TYPE, de la manera siguiente:

TYPE=I	Para números romanos en mayúsculas
TYPE=i	Para números romanos en minúsculas
TYPE=a	Para letras del abecedario en minúsculas
TYPE=A	Para letras del abecedario en mayúsculas

También se puede modificar el elemento de inicio de la lista, para indicar por ejemplo que empiece a numerar en el número 8. Se utiliza el atributo opcional START=# para lograr esto.

Ejemplo:

```
<H1>Ingredientes del pastel</H1>
<OL TYPE=i START=8>
<LI>Harina
<LI>Huevos
<LI>Royal
</OL>
```

Listas de directorios

Se indican con <DIR></DIR>. Cada elemento se establece con <LI>

Menú de opciones

Se indican con <MENU></MENU>. Cada elemento se establece con <LI>

Listas de descripciones

Se indican con <DL></DL>. Cada elemento se establece con <LI>

TABLAS

Las tablas son uno de los elementos más útiles del lenguaje HTML, ya que permite organizar la información tabular en columnas y renglones, además de su importancia para organizar los elementos de una página HTML. Además permite agrupar elementos en entidades uniformes dándole cierto orden a la página. Una tabla típica se escribe como a continuación aparece

Una tabla se define con la instrucción <TABLE></TABLE> En este particular caso es muy importante cerrar la instrucción <TABLE>, ya que de lo contrario ninguna parte del texto que añadimos aparece. ‘

Las tablas inician con un texto opcional encerrado entre <CAPTION></CAPTION>, que sirve como título para la tabla. Tiene dos posibilidades, ALIGN=TOP o ALIGN=BOTTOM

Para definir un renglón se emplea la instrucción <TR>, que puede permanecer sin cerrar, ya que una anula a la otra.

De manera análog a a una hoja de cálculo, cada renglón puede estar formado por una o más celdas, que pueden ser de dos tipos. El primero se define con <TD> representando a una celda normal, que mantiene los estilos que defina el usuario. Por default, las celdas <TH> centran el texto contenido en ellas y las celdas <TD> alinean el texto a la izquierda. El segundo tipo son las celdas definidas con <TH>, que fueron originalmente diseñadas para que sirvieran de cabecera para una tabla llena de valores numéricas. Hoy en día su connotación ha cambiado y ya casí no se utilizan.

Atributos de las tablas <TABLE></TABLE>

ALIGN=LEFT, RIGHT Indica la posición en que estará colocada la tabla en una página

BORDER=# Especifica el grueso del borde de la tabla

COLS=#, ROWS=#. Es útil indicar el número de columnas y el número de renglones en una tabla con el fin de que se empiece a cargar más rápido. De lo contrario, algunos browsers se esperan a tener toda la información de la tabla antesd de desplegarla.

CELLSPACING=# Indica el número de pixeles que separa a una celda de otra

CELLPADDING=# Especifica el número de pixeles que dejará como margen dentro de cada celda, para la escritura de los datos.

WIDTH=# Indica el ancho de la tabla, los elementos se acomodan respetando el ancho. Su valor se puede definir con un porcentaje poniendo un signo de por ciento % después de la cantidad.

HEIGHT=# Define el mínimo alto de la tabla. Esta atributo solo se respeta si la información de la tabla lo permite. Su valor se puede definir con un porcentaje poniendo un signo de por ciento % después de la cantidad.

BGCOLOR=red. De reciente introducción, esta instrucción permite controlar el color de fondo de la tabla.

Atributos de las celdas (<TD></TD>)

Con la instrucción ALIGN, se puede definir como se va a acomodar el texto dentro de un celda, con tres posiblidades: LEFT, CENTER, RIGHT.

COLSPAN=# Especifica cuantas columnas de la tabla deben ser abarcadas por una sola celda.

ROWSPAN=# especifica cuantos renglones debe abarcar una sola celda. Un valor que sea mayor a los renglones especificados será truncado.

VALIGN toma los valores TOP, MIDDLE, BOTTOM y BASELINE, para centrar verticalmente una celda.

NOWRAP Evita que se rompa la integridad en una línea.

WIDTH, HEIGHT, BGCOLOR: Estas instrucciones se pueden incluir en una etiqueta <TD>, controlando la apariencia individual de una celda en el caso de BGCOLOR o de una columna o una fila, al especificar WIDTH y HEIGHT.

Así, una tabla de 3 renglones y cinco columnas se definiría así:

```
<TABLE ALIGN=RIGHT BORDER=3 CELLSPACING=3 CELLPADDING=3>
<CAPTION>Estructura básica de la tabla</CAPTION>
<TR><TH>Celda 1<TH>Celda 2<TH>Celda 3<TD>Celda 4<TD>Celda 5
<TR><TD>Celda 6<TD>Celda 7<TD>Celda 8<TD>Celda 9<TD>Celda 10
<TR><TD>Celda 11<TD>Celda 12<TD>Celda 13<TD>Celda 14<TD>Celda
15
</TABLE>
```

Una celda puede contener gran cantidad de elementos diferentes, tales como imágenes, formas, párrafos, texto preformateado, listas, películas, imágenes,... Las celdas pueden permanecer vacías. Es muy importante aclarar que cada celda tiene su propio estilo, por lo que no se puede definir un estilo, tamaño o color de letra para toda la tabla.

FRAMES

Un documento de frames tien básicamente la misma estructura que otro documento HTML normal, con la excepción de que el <BODY></BODY> es reemplazado por la instrucción <FRAMESET></FRAMESET>, que define los documentos sub-HTML o frames que conforman una página.

```
<HTML>
<HEAD>
</HEAD>
<FRAMESET>
</FRAMESET>
</HTML>
```

SINTAXIS DE LOS FRAMES

Ls sintaxis es similar en profunidad y complejidad a la utilizada en tablas, y ha sido diseñanda para ser procesada rápidamente en los browsers.

<FRAMESET>

Este es el principal contenedor del frame. Contiene dos atributos, ROWS y COLS. Un documento de Frames no tiene BODY, y las tags que aparecerían normalmente en esta zona aparecen despues de FRAMESET, aunque sólo son reconocidas los tags FRAMES, NOFRAME y otro FRAMESET anidado.

ROWS="lista de valores de altura para los renglones"

Este atributo toma una lista delimitada por coma como sus valores. El número de valores india el número de renglones a considerar. El alto total de las filas debe igualar al alto de la ventana, por lo que se realiza un reajuste para lograr esto.

Sintaxis de la lista de valores.

VALOR: un simple valor numérico es interpretado como un tamaño fijo en pixeles. Es peligroso incluir este tipo de valores ya que no podemos determinar el tamaño de la ventana del usuario, y esta es muy variable. Lo recomendable, es combinarlo con otros tipos de valores.

VALOR %: es un porcentaje. Si el total es mayor a 100, todos los porcentajes son reajustados. Si es menor que 100, el espacio restante se distribuye entre los otros frames definidos de manera relativa, en caso de no existir, son reordenados para llenar la ventana.

VALOR *: Un valor * representa un valor relativo para el tamaño del frame. Si todos los valores se definen así, el espacio de la ventana se distribuye proporcionalmente entre ellos. Si se pone un número antes de *, se dice que tendrá tantas veces ese tamaño.

Ejemplos:

```
<FRAMESET ROWS="20%,60%,20%">
<FRAMESET ROWS="100,*,100">
```

COLS="lista de valores de anchos de columna"
Este atributo tiene la misma sintaxis que ROWS.

Es posible anidar varios tags FRAMESET. En este caso el subframe ocupa el espacio correspondiente al frame.

<FRAME>

Esta instrucción define un frame dentro de un frameset. Tiene 6 posibles atributos. Como no es un contenedor, no es necesario cerrarlo.

SRC="url"
Este atributo toma su valor del URL del documento que aparecerá en un frame particular. FRAMEs sin SRC se representan como un espacio en blanco en el espacio asignado a dicho FRAME.

NAME="nombre de de la ventana"
Es un nombre que se usa para que se puedan hacer referencias a este frame, generalmente por otros frames del mismo documento. Por defecto, todos los frames carecen de nombre. Los nombres tienen que empezar con un caracter alfanumérico.

MARGINWIDTH="valor"
Se utiliza cuando el autor del documento controla los márgenes del frame. Se especifica en pixeles. No pueden ser menores que 1. Por defecto, el browser decide cuanto margen dejar.

MARGINHEIGHT="value"
Es similar al MARGINWIDTH, pero controla los márgenes superior e inferior.

SCROLLING="yes|no|auto"
Este atributo se usa para describir si el frame tendrá o no una SCROLLBAR. Automático coloca las barras de navegación sólo cuando son necesarias. El valor por defecto es AUTO.

NORESIZE
No tiene valor. Indica si el usuario puede o no modificar las dimensiones del frame. Los frames adyacentes a los bordes no se pueden modificar.Por defecto se puede modificar.

<NOFRAMES>

Esta instrucción se utiliza para incluir instrucciones para aquellos programas que no soportan frames.

CON TABLAS

```
<TABLE WIDTH="100%" HEIGHT="100%" BORDER>
    <TR><TD ROWSPAN=2>CELL1</TD><TD>CELL2</TD></TR>
    <TR><TD ROWSPAN=2>CELL3</TD></TR>
    <TR><TD ROWSPAN=2>CELL4</TD></TR>
    <TR><TD>CELL5</TD></TR>
</TABLE>
```

CON FRAMES

```
<FRAMESET COLS="50%,50%">
  <FRAMESET ROWS="50%,50%">
    <FRAME SRC="cell.html">
    <FRAME SRC="cell.html">
  </FRAMESET>
  <FRAMESET ROWS="33%,33%,33%">
    <FRAME SRC="cell.html">
    <FRAME SRC="cell.html">
    <FRAME SRC="cell.html">
 </FRAMESET>
</FRAMESET>
```

Información NOFRAMES

```
<NOFRAMES>
<h1 align=center><blink>Frame ALERT!</blink></h1>
<p>
Este documento fue dise&ntilde;ado para ser visto
con<b>Netscape 2.0</b>. Si est&aacute;s viendo este mensaje,
tu browser no soporta frames.
</p>
<p>
Puedes conseguir un browser <b>capaz de desplegar frames</b>
en<a href=home.netscape.com>Netscape Communications</a>.
</p>
</NOFRAMES>

<FRAMESET ROWS="50%,50%">
  <FRAME SRC="cell.html">
  <FRAME SRC="cell.html">
</FRAMESET>
<FRAMESET ROWS="33%,33%,33%">
  <FRAME SRC="cell.html">
  <FRAME SRC="cell.html">
  <FRAME SRC="cell.html">
</FRAMESET>
```

TARGET

Es importante tener control sobre donde aparecerá la información cuando el usuario presione un link. Antes, cuando un usuario presionaba un link, el nuevo documento aparecía en la ventana en la que estaba trabajando dicho usuario o si el usuario deseaba, en una nueva ventana. El hacer un target a una ventana permite al creador de la página HTML la asignación de nombres a ventanas específicas, y permite que ciertos documentos solo aparezcan en la ventana que tiene un nombre idéntico al que especificamos.

Un nombre puede ser asignado a una ventana de tres maneras distintas:
1. Enviando el documento con http. Esto hace que el documento se cargue en determinada ventana, y si esta no existe, se crea.
2. Un documento puede ser accesado por un link con target.
3. Una ventana creada a partir de un FRAMESET puede ser nombrada a partir del atributo NAME del tag FRAME.

HTML

Target en la instrucción ANCHOR <A></A>

Esta instrucción normalmente carga una página en una ventana. Aquí solo se especifica a que ventana o frame se quiere cargar la página. <A HREF="url" TARGET="nombre">texto que el usuario presionará</A>

TARGETs Mágicos.

Todos los nombres van precedidos de un caracter de subrayado.

TARGET="_blank" : Causa que un link sea cargado en una ventana nueva, vacía. Esta ventana no tiene nombre.

TARGET="_self": El link se carga en la misma ventana en la que el link está ubicado, sirve para desactivar BASE

TARGET="_parent": Hace que el link se cargue en el FRAMESET parte del documento, en caso de no existir, funciona como self.

TARGET="_top": Causa que la instrucción se cargue en toda la ventana, saliendo de la estructura de frames.Actúa como self si ya está en el nivel superior.

FORMAS

Una forma es una plantilla para sets de información, con un método asociado y una acción URI. El set de información es una secuencia de nombre/valor pares de campos. Los nombres son especificados en el atributo NAME y los valores pueden ser inicializados o editados por el usuario. El resultado de llenar la forma se utiliza para accesar a un servicio por medio de una acción y un método.

Los elementos de la forma pueden estar mezclados con las instrucciones de la página., esto proporciona gran flexibilidad en el diseño de las páginas con formas.

<FORM> El elemento FORM contiene una secuencia de input elements, combinado con las instrucciones para dar estructura a la página. Atributos:

ACTION

Especifica la acción para la forma, es decir la ruta de acceso y el nombre del programa encargado de manejar la forma. Generalmente es un programa llamado CGI, que significa Common Gateway Interface.

METHOD

Especifica el método para mandar la forma, para accesar al programa CGI que la maneja. En otras palabras, selecciona un método HTTP para accesar al CGI. Existen dos métodos:

METHOD=GET: La información que se manda por este método es anexada al final de la acción que fue solicitada. El programa CGI recibirá la información en un ambiente de variables denominado QUERY_STRING

METHOD=POST: Este método transmite la información antes del URI solicitado.Los resultados se reciben en forma codificada en un archivo STDIN. Es necesario saber cuanta información se debe leer de este archivo utilizando un entorno de variables llamado CONTENT_LENGTH.

ENCTYPE

Especifica el tipo de encriptación con el que se codifica el archivo para transportarlo. En el método POST, el atributo es application/x-www-form-urlencoded por defecto.

<INPUT>

Esta instrucción permite introducir una sola palabra o línea de texto, con un valor por defecto de 20 caracteres. Generalmente es precedido por un texto descriptivo. Atributos

NAME="nombre"

Especifica el nombre del campo

VALUE="texto"

Asigna un valor por defecto a un campo, es decir el valor que aparecerá cuando la forma sea cargada por primera vez.

SIZE=#
Especifica el tamaño del campo en caracteres. El usuario puede escribir más caracteres de los que aparecen, y podrá ver el texto a medida que lo escribe.

MAXLENGTH=#
Indica el máximo número de caracteres que el usuario puede escribir en un campo. La forma hará un sonido de error si el usuario intenta escribir más caracteres de los previstos.

TYPE="text, number, password, checkbox, radio, submit, reset"
Si este atributo no es especifica, el valor por defecto es texto. Este atributo sirve para especificar que tipo de campo se va a utilizar.Text, number y password, tienen la apariencia de campos de texto. Checkbox, submit, radio y reset son botones o checkboxes.

Campo de texto INPUT TYPE=TEXT
Sirve para la entrada de una sola línea de texto.
Atributos Requeridos:
NAME, un nombre para este elemento
Atributos opcionales
MAXLENGTH
SIZE
VALUE

Reset, Submit INPUT TYPE=RESET, INPUT TYPE=SUBMIT
Estos botones son creados con INPUT tags.

El botón RESET se utiliza para poner los valores por defecto de cada campo, si es que fueron definidos, como si el usuario no hubiera editado ninguno de los campos.

El botón SUBMIT transfiere toda la información escrita por el usuario al URL especificado en el elemento ACTION.
Si se define el atributo de NAME en un botón tipo SUBMIT, la forma transmitirá todos los contenidos del elemento VALUE, permitiendo así la existencia de varios botones SUBMIT. Esto puede ser utilizado para crear botones de BACK y FORWARD, eliminando las desventajas de utilizar gráficos.

Se puede enviar la forma de manera interna:
```
<input type=submit name=recipient value=internal><br>
```
o de manera externa
```
<input type=submit name=recipient value=world>
```

Ejemplo:

```
<INPUT TYPE=SUBMIT NAME=BUTTON VALUE=HOME>
<INPUT TYPE=RESET NAME=BUTTON VALUE=HOME>
```

<TEXTAREA></TEXTAREA>
TEXTAREA crea un área para texto con múltiples renglones. Es un tag contenedor, es decir, se tiene que cerrar. Ejemplo:

```
<FORM METHOD=POST ACTION="MAILTO:JJGB@mexred.net.mx">
<TEXTAREA ROWS=5 COLS=64 NAME="mensaje">Esta es una prueba</
TEXTAREA>
<INPUT TYPE=SUBMIT VALUE="Enviar un correo">
</FORM>
```

o

```
<TEXTAREA NAME="address" ROWS=6 COLS=64>
HaL Computer Systems
1315 Dell Avenue
Campbell, California 95008
</TEXTAREA>
```

Atributos:
NAME, similar a los vistos anteriormente
ROWS, COLS, indican el número de renglones y columnas para el texto.
Texto por defecto. Si se desea que aparezca algún texto por defecto, es necesario incluirlo entre la apertura y el cierre de la instrucción.

Campo escondido INPUT TYPE=HIDDEN
Es posible esconder un campo de la vista del usuario, pero que si podrá mandar información al momento de apretar el botón de SUBMIT.
Ejemplo: <INPUT NAME="prueba" VALUE="experimento" TYPE=HIDDEN>

Contraseñas, passwords INPUT TYPE=PASSWORD
Son idénticos a los campos de texto, pero al momento de escribir ponen balas (•) o asteriscos (*) en lugar del texto real.
Ejemplo:

```
 <INPUT NAME="contraseña" VALUE="experimento" TYPE=PASSWORD
SIZE=8 MAXLENGTH=8>
```

En el VALUE, se puede especificar un valor para una password en general.

NOTA: es importante especificar el SIZE de un campo cuando se pone un MAXLENGTH arbitrario, para estar seguros de que el usuario podrá ver todo el texto que escriba.

Checkbox INPUT TYPE=CHECKBOX
Un INPUT de este tipo es una opciòn booleana. Un set de n elementos con el mismo nombre represent un campo de selección de n elementos. Atributos obligatorios:
NAME Nombre simbólico para el campo correspondiente e este elemento.

VALUE Porción del valor del campo que este elemento proporciona.
Atributos opcionales:
CHECKED indica que el estado inicial del checkbox es activo.
Ejemplo:
<p>Que sabores te gustan?

```
<input type=checkbox name=sabor value=vainilla>Vainilla<br>
<input type=checkbox name=sabor value=fresa>Fresa<br>
<input type=checkbox name=sabor value=chocolate
checked>Chocolate<br>
```

Radio Button INPUT TYPE=RADIO

Es una opción booleana. Un set de este tipo de elementos con el mismo nombre permite seleccionar sólo uno de los elementos del set. Los atributos NAME y VALUE son requeridos, como en el caso de las CHECKBOXES. Los atributos opcionales son:

CHECKED indica que el estado inicial del checkbox es activo.

A cada momento, sólo uno de los botones del set está marcado. Si ningún elemento del set es marcado inicialmente, se marca el primero.

Ejemplo:

```
<input type=radio name=sabor value=vainilla>Vainilla<br>
<input type=radio name=sabor value=fresa>Fresa<br>
<input type=radio name=sabor value=chocolate
checked>Chocolate<br>
```

Este tipo de elementos no son recomendables para listas con muchos elementos, ya que los browsers tienen problemas en desplegar muchos botones. Para esos casos es mejor utilizar SELECT. Si se tienen muchos elementos de este tipo con el mismo nombre pero sin valores específicos, el servidor tendrá problemas en definir cual deberá ser seleccionado, ya que alguno de ellos debe tener un valor de encendido. Una vez que se ha seleccionado un botón de este tipo, no puede ser deseleccionado sin escoger otro botón con el mismo nombre. El valor por default solo puede regresar apretando el botón RESET, si es que existe.

Image Pixel: INPUT TYPE=IMAGE

Un elemento de este tipo especifica la entrada de un recurso gráfico, y permite la entrada de dos campos: las coordenadas x y y. TYPE=IMAGE implica un proceso TYPE=SUBMIT, esto es, cuando un pixel es seleccionado, la forma es enviada.

El atributo NAME es requerido como en otros casos. También es necesario incluir el atributo SRC. ALIGN es opcional, de manera similar a la instrucción IMG.

Ejemplo:

```
<p>Selecciona un punto en el mapa:
<input type=image name=point src="map.gif">
```

Selección: SELECT

El elemento SELECT limita a un campo a una lista de valores enumerados. Estos valores se dan en elementos OPTION. Sus atributos son:

MULTIPLE. Indica que más de una opción puede estar incluida en el valor
NAME. Especifica el nombre del campo
SIZE. Especifica el nombre de los items visibles. Los del tipo 1 son tipicamente pop-down menus, mientrás que otros números representan listas típicas, con una barra de scroll.

EJEMPLO:

```
<SELECT NAME="sabor">
<OPTION>Vainilla
<OPTION>Fresa
<OPTION value="ciruelapasa">Ciruelas pasas
<OPTION selected>Durazno y Naranja
</SELECT>
```

En el estado inicial, la primera opción es seleccionada, a menos que alguna de las opciones tenga el atributo SELECTED.

Opción: OPTION
El elemento opción solo puede ocurrir dentro de un elemento SELECT. Representa una de las opciones, y tiene los siguientes atributos:

SELECTED
Indica si la opción está inicialmente marcada
VALUE
indica el valor que va a regresar si esta opción es seleccionada. El valor por default es igual al nombre del elemento.

Enviado de formas

Un browser empieza por procesar la forma al presentar los campos en su estado inicial. El usuario puede modificar los campos, con las limitaciones de cada tipo de campo.Cuando el usuario indica que desea mandar la forma, esta es enviada, ya sea mediante un botón submit o mediante un image input. La información contenida en dicha forma es procesada de acuerdo con su METHOD, ACTION URI, y ENCTYPE. Cuando sólo hay una línea de texto para ser llenada, el presionar retorno o la tecla enter funcionará para mandar la forma, en lugar de presionar el botón.

form-urlencoded Media Type

El tipo de codificación por defecto para las formas es 'application/x-www-form-urlencoded'. El set de información aparece como a continuación se explica.

1. Los nombres y valores de los campos son cambiados por sus códigos de escape. Los espacios son reemplazados por signos +. Los caracteres no alfa-numéricos son reemplazados por un símbolo de por ciento seguido de un código hexadecimal que representa el

código ASCII de dicho caracter.

Ejemplo: %HH.

Los saltos de línea en campos de múltiples líneas se representan como pares CR LF.

Ejemplo: %0D%0A.

2. Los campos son listados en su orden de aparición en el documento, con el nombre separado del valor por un signo =, y los pares separados uno de otro por un &. Los campos con valores nulos pueden ser omitidos. Los radio buttons y checkboxes no seleccionados son ignorados. Los campos invisibles con un valor escondido están presentes.

Query Forms: METHOD=GET

Para procesar una forma cuyo ACTION URL es una dirección HTTP y cuyo método es GET, se empieza poniendo el ACTION URL seguido de ? y de la información del set, en un formato 'application/x-www-form-urlencoded'. Se transmite este valor como si fuera un ANCHOR, como un link.

Otras formas: METHOD=POST

Si el servicio asociado al procesamiento de la forma tiene repercusión (como la modificación de una base de datos o la suscripción a un servicio), el método debe ser POST.

Para procesar una forma que tiene como ACTION URL una dirección HTTP y cuyo método es POST, se hace una transacción utilizando el ACTION URL y el mensaje con formato 'application/x-www-form-urlencoded'. La respuesta es similar al método GET.

Ejemplo de una forma de cuestionario.

Considera el siguiente documento

```
<!DOCTYPE HTML PUBLIC «-//IETF//DTD HTML 2.0//EN">
<title>Sample of HTML Form Submission</title>
<H1>Sample Questionnaire</H1>
<P>Please fill out this questionnaire:
<FORM METHOD="POST" ACTION="http://www.w3.org/sample">
<P>Your name: <INPUT NAME="name" size="48">
<P>Male <INPUT NAME="gender" TYPE=RADIO VALUE="male">
<P>Female <INPUT NAME="gender" TYPE=RADIO VALUE="female">
<P>Number in family: <INPUT NAME="family" TYPE=text>
<P>Cities in which you maintain a residence:
<UL>
<LI>Kent <INPUT NAME="city" TYPE=checkbox VALUE="kent">
<LI>Miami <INPUT NAME="city" TYPE=checkbox VALUE="miami">
<LI>Other <TEXTAREA NAME="other" cols=48 rows=4></textarea>
</UL>
Nickname: <INPUT NAME="nickname" SIZE="42">
<P>Thank you for responding to this questionnaire.
<P><INPUT TYPE=SUBMIT> <INPUT TYPE=RESET>
</FORM>
```

El estado inicial de la información es:

name	«"
gender	«male"
family	«"
other	«"
nickname	«"

Es importante notar que el INPUT del Radio Button tiene un valor inicial, mientras que el CHECKBOX no tiene.

El usuario puede editar los campos y enviar la forma. En este punto, supongamos que los valores son:

name	«John Doe"
gender	«male"
family	«5"
city	«kent"
city	«miami"
other	«abc\ndef"
nickname	«J&D"

El browser conduce un método POST en la dirección especificada 'http://www.w3.org/sample'. El cuerpo del mensaje sería (ignorando el cambio de línea)

```
name=John+Doe&gender=male&family=5&city=kent&city=miami&
other=abc%0D%0Adef&nickname=J%26D.
```

IMAGE MAPS

Un image map es una imagen a la que está asociado un archivo que incluye coordenadas con respecto a la imagen. Cuando el usuario presiona algún punto de la imágen, se consultan las coordenadas de dicho punto y se le asocia una ruta de acceso, que puede ser absoluta o relativa. Una imagen puede tener tantos links asociados como pixeles contenga, pero incluir un link para cada pixel sería imposible.

Los image maps pueden ser de dos tipos: del lado del cliente y del lado del servidor.

Los mapas del lado del servidor están constituidos por un archivo externo con terminación .map, que incluyen una serie de coordenadas asociadas a referencias absolutas o relativas.

Funcionan anclando determinada imagen al archivo del mapa. Cada vez que el usuario presiona en una parte de la imagen, se debe consultar el archivo .map en el servidor, para ver cual es la dirección asociada este proceso puede ser en ocasiones lento. La barra de status en la parte inferior del browser muestra una serie de coordenadas cuando el puntero pasa por encima, siendo impráctico porque no se sabe el nombre de la página de destino para cierto punto de un mapa.

Se utilizan así:

```
<A HREF="navegacion.map"><IMG SRC=navegacion.gif ISMAP></A>
```

siendo navegacion.map un archivo de texto externo.

Ejemplo:

```
circle /poniente.html 34,184 39,184
rect /centro.html 23,22 82,33
poly /sur.html 93,234 84,222 78,206 72,192 70,189 70,172
74,167 73,159 76,153 80,150 82,146 88,143 87,140 92,137 91,129
98,124 103,122 104,118 124,114 127,115 130,116 132,115 139,114
139,132 144,133 148,134 149,142 153,149 159,152 168,166
```

Como se puede notar, existen tres figuras que se pueden definir como links dentro de una imagen: poly, circle y rect.

RECT representa un cuadrado y está definido por cuatro valores, las coordenadas del extremo superior izquierdo y las coordenadas del extremo inferior derecho, es decir x,y,x2,y2.

CIRCLE está definido por cuatro coordenadas que representan un cuadrado en el que al intersectar sus diagonales se obtiene el centro.

POLY representa un polígono irregular, que tiene tantos puntos como puntos tenga el polígono, cada uno representado por dos coordenadas x y y. Cada figura lleva una ruta de acceso absoluta o relativa asociada. Es además necesario incluir la instrucción ISMAP en la declaración de la imagen. Como en el ejemplo anterior.

Los mapas del lado del cliente se escriben directamente en la página html. Tienen la ventaja de que no es necesario accesar a un archivo externo para poder utilizarlos, ahorrando así tiempo de servidor. Por otro lado, permiten conocer la dirección de la página a la que se dirige con cada porción a través de la barra de status, en la parte inferior de la ventana del browser. Un mapa del lado del cliente se define con el tag <MAP></MAP> Se pueden definir áreas dentro de la imagen que sean links a diferentes URLs. Las áreas pueden ser rectángulos, círculos o polígonos.

```
<MAP NAME="nav">
   <AREA SHAPE="RECT" COORDS="0,0,16,14"
HREF="navegacion.html">
   <AREA SHAPE="CIRCLE" COORDS="20,20,5"
HREF="navegacion2.html">
   <AREA SHAPE="POLY" COORDS="42,67 42,43 75,45 82,31 95,27
66,0 107,0 114,13 129,11 141,9 154,9 171,-1 182,4 186,3 197,16
193,21 193,31 186,54 186,68 162,74 170,87 175,98 164,97 160,95
159,96 152,95 144,91 137,88 135,88"   HREF="navegacion3.html">
</MAP>
```

En la imagen se pone así:

```
<IMG SRC=imagen.jpg USEMAP="#nav">
```

Aunque ambos son muy parecidos, es importante decidir cual se aplicará, ya que algunos browsers no soportan los del lado del cliente, pero sus beneficios son muy marcados. Lo ideal es incluir ambos tipos.

```
<MULTICOL>
<SPACER>
<FONT FACE>
<FRAME BORDERCOLOR BORDER>
```

Introducción a Javascript

Javascript es un lenguaje compacto basado en objetos para el desarrollo de aplicaciones cliente servidor.Netscape Navigator reconoce las instrucciones Javascript intercaladas entre otros elemento HTML. Por ejemplo, se puede crear una instrucción Javascript para verificar que los usuarios escriban información válida en un campo de una forma que pida un número telefónico o un código postal. Javascript también se puede utilizar para realizar una acción, como reproducir un archivo de música, ejecutar un applet, o comunicarse con un plug-in como respuesta a una acción del usuario.

Aunque el lenguaje Javascript es muy sencillo es a la vez muy poderoso, sin requerir de compiladores sofisticados ni una sintaxis rigurosa. El lenguaje javascript es interpretado por un browser, pero no compilado.

Usando Javascript en HTML

Javascipt puede ser introducido (embedded) en HTML de dos maneras diferentes:
•Como declaraciones y funciones en el tag SCRIPT.
•Como event handlers dentro de una instrucción HTML.

SCRIPT
Se utiliza la intrucción SCRIPT de la siguiente manera:

```
<SCRIPT LANGUAGE="javascript">
Declaraciones o instrucciones de javascript
</SCRIPT>
```

Javascript es sensible a las minúsculas y mayúsculas.
Los scripts pueden ser escondidos entre comentarios para garantizar que los navegadores viejos no lo desplegarán como texto.

Ejemplo simple:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE=JAVASCRIPT>
<!– para esconder el contenido de javascript de navegadores
viejos
   document.write("Hello World Wide Web.")
// fin del camuflaje para navegadores viejos–>
</SCRIPT>
<BODY>
Fin del mensaje
</BODY>
</HTML>
```

Declaración de funciones

```
<HTML>
<HEAD>
```

```
<SCRIPT LANGUAGE=JAVASCRIPT>
<!– para esconder el contenido de javascript de navegadores
viejos
   function square (i)
         {
         return i * i
         }
document.write(«La funci&oacute;n regres&oacute; «,
square(5),".")
// fin del camuflaje para navegadores viejos–>
</SCRIPT>
<BODY>
<p>Fin del mensaje
</BODY>
</HTML>
```

Uso de la instrucción Write.

Una de los métodos más usados en javascript es write. Es muy útil, por ejemplo para desplegar texto frente a ciertas circunstancias. Se pueden concatenar las variables string con un signo +, en este método.

Ejemplo:

```
<HTML>
<HEAD>
<SCRIPT>
<!– para esconder el contenido de javascript de navegadores
viejos
   function barra (ancho)
         {
         document.write(«<HR ALIGN='left' WIDTH="+ancho+"%>")
         }

   function escribe (nivel,texto)
         {
         document.write(«<H"+nivel+">", texto, «</H"+nivel+">")
         }
// fin del camuflaje para navegadores viejos–>
</SCRIPT>
<BODY>
<SCRIPT>
<!– para esconder el contenido de javascript de navegadores
viejos
barra(25)

escribe(2, «Este es un ejemplo de Javascript")
// fin del camuflaje para navegadores viejos–>
</SCRIPT>
<p>Este texto fue creado tradicionalmente, el que aparece
arriba fue generado.
</BODY>
</HTML>
```

En el código que aparece arriba, se definieron dos funciones, barra y escribe.

Event Handlers (manejadores o controladores de eventos)

Las aplicaciones en Javascript dependen mucho de los eventos que sucedan. Un evento puede ser casí cualquier acción del usuario, como mandar una forma, apretar un botón, seguir un link, o incluso mover el puntero del mouse sobre un link.

Cada evento es reconocido por ciertas instrucciones HTML, que se resumen en la siguiente tabla:

Evento	Aplica a	Ocurre cuando	comando
blur	text, textareas, select	El usuario sale de la edición del elemento.	onBlur
click	buttons, radio, submit, checkboxes, reset, links	El usuario hace click en un elemento o link	onClick
change	text, textareas, select	El usuario cambia el valor de un elemento	onChange
focus	text, textareas, select	El usuario entra a la edición de un elemento	onFocus
load	body	El usuario carga la página en Navigator	onLoad
mouseover	links	El usuario mueve el puntero sobre un link	onMouseOv
select	text, textareas	El usuario selecciona el campo	onSelect
submit	submit	El usuario manda una forma	onSubmit
unload	body	El usuario sale de una	onUnload

Ejemplo de uso de event handler.

```
<HTML>
<HEAD>
<SCRIPT>
<!– para esconder el contenido de javascript de navegadores
viejos
   function calcular (forma)
         {
         if (confirm(«Presiona Yes para evaluar esta
expresion."))
                                  forma.resultado.value =
eval(forma.expresion.value)
         else
                                  alert(«Regresa pronto")
         }
// fin del camuflaje para navegadores viejos–>
</SCRIPT></HEAD>
<BODY>
<FORM>
Escribe una expresi&oacute;n matem&aacute;tica:
<INPUT TYPE=text NAME=expresion SIZE=15>
<INPUT TYPE=button  VALUE=Calcular
onClick="calcular(this.form)">
<p>
<H4>Resultado</H4>
<INPUT TYPE=text NAME=resultado SIZE=15>
</FORM>
</BODY>
</HTML>
```

En esta página se utilizan algunas instrucciones útiles. A continuación se proporciona una breve explicación.

confirm(“Mensaje”)

Método que despliega en pantalla una ventana de diálogo, con el mensaje especificado y botones para cancelar o aceptar. Es posible utilizarla dentro como event handler.

```
<INPUT TYPE=BUTTON VALUE=QUIT onClick=“confirmCleanup()”>
```

alert(“Mensaje”)

Despliega una ventana de alerta, con un mensaje y un botón OK.

Ejemplo:

```
<HTML>
<HEAD>
<SCRIPT>
<!– para esconder el contenido de javascript de navegadores
viejos
   function pruebaValor (elemento)
         {
```

```
        if ((elemento.length > 8)||(elemento.length < 1)){
                                alert(«Por favor escribe un
nombre que tenga entre 1 y 8 caracteres.")
        }
   }
// fin del camuflaje para navegadores viejos—>
</SCRIPT></HEAD>
<BODY>
<FORM>
Nombre:
<UL><INPUT TYPE=text NAME=nombreUsuario
onBlur="pruebaValor(nombreUsuario.value)"></UL><br>
Direcci&oacute;n:
<UL><INPUT TYPE=text NAME=direccion></UL><br>
Colonia:
<UL><INPUT TYPE=text NAME=colonia></UL><br>
</FORM>
</BODY>
</HTML>
```

Creación de pistas con onMouseOver

Por default, cuando mueves el puntero sobre un hipertexto, la barra de status pone el URL de destino del link. Se puede definir el status en el event handler on MouseOver, para poner pistas en el link. El event handler debe regresar true para definir el status.

```
<A HREF="indice.html" onMouseOver="window.status='Presiona
aquí para ver el indice."; return true">Indice</A>
```

El objeto MATH

Este objeto tiene propiedades matemáticasy metodos para funciones y constantes matemáticas.

Ejemplo.

La constante π PI, regresa el valor 3.142592. Se debe usar como
Math.PI

Por ejemplo, para usar la función tangente de este objeto, se escribe
Math.sin(3.3)

Lista de métodos

abs	valor absoluto
sin, cos, tan	funciones trigonométricas en radianes
acos, asin, atan	funciones trigonométricas inversas, en radianes
exp, log	exponente y logaritmo natural con base e
ceil	regresa el entero mayor que o igual al número dado
floor	regresa el entero menor que o igual al número dado
min, max	regresa el mayor o el menor de dos números
pow	(base, exponente) eleva un número a determinado exponente

round	redondea hacia el entero más cercano
sqrt	saca la raiz cuadrada

Para evitar tener que escribir Math cada vez que se utiliza, se puede poner de este modo.

```
with(Math) {
   a = PI * r * r
   y = r * sin(theta)
   x = r * cos(theta)
}
```

```
<HTML>
<HEAD>
<SCRIPT>
<!—
   function areacirc (forma)
         {
         if (confirm(«Presiona Yes para evaluar esta
expresion."))
                                    forma.resultado.value =
Math.PI * forma.expresion.value * forma.expresion.value
         else
                                    alert(«Regresa pronto")
         }
// —>
</SCRIPT></HEAD>
<BODY>
<FORM>
Escribe el radio de un c&iacute;rculo:
<INPUT TYPE=text NAME=expresion SIZE=15>
<INPUT TYPE=button  VALUE="Obtener el area"
onClick="areacirc(this.form)">
<p>
<H4>Resultado</H4>
<INPUT TYPE=text NAME=resultado SIZE=15>
</FORM>
</BODY>
</HTML>
```

Consejos prácticos para el diseño de páginas HTML

Planeación del Site

1. Antes de empezar a diseñar páginas es necesario definir hacia que tipo de público van dirigidas, que tipo de información sería de interés para las personas.

2. Es importante fijar un objetivo antes de desarrollar las páginas. El propósito puede ser informar, educar, entretener o involucrar al lector.

3. Desarrollar contenido antes de diseñar. Es importante definir cual es la información que se desea comunicar a las personas. El diseño debe funcionar de acuerdo con el propósito del site.

4. Agregar información valiosa. Es importante agregar información que sea de utilidad a las personas, que sea útil.

Proporcionar lo básico.

1. Es importante proporcionar información sobre el o los autores de las páginas, incluyendo una manera de contactarlo.

2. Debe haber una identificación sobre la compañía que desarrolla el site.

3. Es conveniente incluir la dirección del URL en la página, por si es impresa.

4. Es importante brindar a los usuarios información relativa a la última actualización de la página, para que sepan que tan vigente es la información

Uso de ventanas y frames

Javascript permite la creación y manipulación de ventanas y frames para presentar contenido html. El objeto window se encuentra en el primer nivel de la jerarquía de objetos. El objeto frame tiene la misma jerarquía pero son considerados sub ventanas.

Cuando se abre el Navigator, este crea una ventana de manera automática. Se pueden crear más ventanas con New Web Browser.

Abrir una ventana
Se puede crear una ventana con el método open.
Por ejemplo:

```
msgWindow = window.open(“sesame.html”)
```

El siguiente ejemplo crea una ventana llamada Ejemplo que despliega la página principal de México en Red.

```
Ejemplo=window.open(“http://www.mexred.net.mx”)
```

Una ventana puede tener dos nombres. El ejemplo siguiente muestra a una ventana con dos nombres. El primero, msgWindow, se refiere a sus propiedades, a sus métodos y a su jerarquia. El segundo, displayWindow, se refiere a la ventana como un target para ser llamado por una forma o un link.

```
msgWindow= window.open(“sesame.html”,”displayWindow”)
```

El segundo nombre no es necesario a menos de que se quiera accesar a la ventana a través de un link.
Cuando se abre una ventana, se pueden especificar atributos tales como su altura y su ancho, si tiene barra de herramientas y si tiene scrollbar.

```
MsgWindow = window.open(“sesame.html”, “displayWindow”,
“toolbar=no”, “scrollbars=no”)
```

Ejemplo:

```
<html>
<body>
<script>
msgWindow=window.open(«Ejemplo2.html","nombre","toolbar=no,width=300,height=300")
</script>
<H4>Esta es una prueba</H4>
</body>
</html>
```

Para cerrar una ventana se utiliza

```
window.close() ó
self.close
```

Para referirse a una ventana:

self o window:

son sinónimos de la ventana activa. Se pueden utilizar indistintamente para referirse a la ventana activa.

Top o parent:

top se refiere a la ventana en el nivel más arriba. Parent se refiere a una ventana que contiene un frameset.

Por ejemplo:

parent.frame2.document.bgColor="khaki" cambia el color de fondo de un frame llamado frame2 al color khaki.frame2 es un frame en el frameset activo.

Debido a que la existencia de la ventana es asumida, no es necesario hacer referencia a ella. Por ejemplo, con sólo poner close() se puede cerrar una ventana. Cuando se incluye una instrucción close en un event handler si es necesario especificar window.close(), ya que de otro modo, el no especificar un objeto en este punto nos llevaría a document.close().

Para referirse a objetos en otras ventanas. Se pueden manipular objetos de otra ventana sin darles focus, esto es, sin que la ventana activa deje de serlo. Ejemplos de esto pueden ser:

```
//Determinar si una checkbox está marcada
if (ventana.document.formamusica.checkbox2.checked) {
   alert("La checkbox ha sido marcada") }

//marcar la checkbox
ventana.document.formamusica.checkbox2.checked = true

//Determinar si una opción en un campo select ha sido selec-
cionada
if
(ventana.document.formamusica.tiposmusica.options[1].selected)
{
   alert("La opcion 1 ha sido seleccionada.") }
```

Para referirse a un frame de otra ventana

```
ventana2.frame2.document.bgColor="violet"
```

Utilizar un botón para crear una ventana

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

```
<FORM>
<p>
<INPUT TYPE=button VALUE="Abrir ventana 2"
onClick="msgWindow=window.open(‘ ‘,
‘window2’,’resizable=no,width=200,height=200')">
<p>
<A HREF="Ejemplo2.html" TARGET="window2">Cargar un archivo en
la ventana 2</A>
<p>
<INPUT TYPE=button VALUE="Cerrar ventana2"
onClick="msgWindow.close()">
</FORM>
</body>
</html>
```

Ejemplos de validación de funciones

El primer ejemplo sólo permite enviar la forma en caso de que el texto que se introduzca ocupe dos caracteres. Cualquier otro número de caracteres mandará un mensaje de error.

```
<HTML>
<HEAD>
<SCRIPT>
function checkData2(item)
   {
   num=0
   if (item.value.length == 2)
         {num++
         alert(«La forma ha sido enviada.")
         window.document.forms[0].submit()}
   else
         {returnVal=false
         alert(«Lo siento, la forma no pudo ser enviada.")}
   }
</SCRIPT>
<BODY>
<H3>Validaci&oacute;n</H3><P>
<FORM NAME=PREGUNTAS METHOD="POST"
ACTION="mailto:JJGB@mail.proesa.com.mx" >
Texto 1: <INPUT TYPE="text" NAME="txt1" SIZE=10><BR>
<INPUT TYPE="button" VALUE="Enviar"
onClick="checkData2(this.form.txt1)">
</FORM>
</BODY>
</HTML>
```

El siguiente ejemplo checa las cantidades para que estén en un determinado rango de valores. En caso contrario, no permite mandar la forma.

```
<HTML>
<HEAD>
```

```
<SCRIPT>
function checa(item,min,max)
{
   var returnVal=false
   if (parseInt(item.value) < min)
         alert(«Por favor escribe un numero mayor.")
   else if (parseInt(item.value) > max)
         alert(«Por favor escribe un numero menor.")
   else
         returnVal = true
   return returnVal
}

function validamanda(laforma)
   {
   if (checa(laforma.cantidad, 0, 999)) {
         alert(«Tus respuestas han sido enviadas.")
         laforma.submit()  }
   else
         alert(«Lo siento, tus respuestas no pudieron enviar-
se.")
   }
</SCRIPT>
</HEAD>
<BODY>
<H3>Mandar una forma</H3><P>

<FORM method="post" action"mailto:JJGB@mail.proesa.com.mx">
Numero 1: <INPUT TYPE="text" NAME="cantidad" SIZE=10
onchange=checa(this,0,999)><BR>
Numero 2: <INPUT TYPE="text" NAME="cantidad" SIZE=10
onchange=checa(this,0,999)><BR>
Numero 3: <INPUT TYPE="text" NAME="cantidad" SIZE=10
onchange=checa(this,0,999)><BR>
<INPUT TYPE="button" VALUE="Envia la forma"
onClick="validamanda(this.form)">
</FORM>
</BODY>
</HTML>
```

Se puede introducir un llamado directo a una función Javascript desde un link. Al utilizar un URL del tipo javascript:, se pueden ejecutar instrucciones de javascript en vez de cargar un documento. Un ejemplo puede ser:

```
<A HREF=“javascript:history.go(0)”>Vuelve a cargar ahora
(Reload)</A>

Otro ejemplo puede ser este contador para saber cuantas veces
se presiona un link.

Function cuentabrincos() {
   parent.p1++
```

```
window.location=pagina1
}
```

Para llamar a esta función utilizamos un Javascript URL

```
<A HREF="javascript:cuentabrincos()">Pagina 1</A>
```

En este ejemplo se asume que pagina1 es un string que representa a un URL

Navegación a través de el elemento <SELECT></SELECT> de una forma

```
<HTML>
<HEAD><TITLE>Seleccionador</TITLE>
<SCRIPT>
function seleccionador()
{
        var seleccion = document.menuform.menu.selectedIndex;
        var nuevaloc = null;
              if (seleccion == 1) nuevaloc = 'Ca.Ci.Sr.html';
              else if (seleccion == 2) nuevaloc =
'Ca.Ci.Nt.html';
              else if (seleccion == 3) nuevaloc =
'Ca.Ci.Ot.html';
              else if (seleccion == 4) nuevaloc =
'Ca.Ci.Pt.html';
              else if (seleccion == 5) nuevaloc =
'Ca.Ci.Ct.html';
              if (nuevaloc) setTimeout(«location = \'" +
nuevaloc + «\'", 0);
}
</SCRIPT>
</HEAD>
<BODY BGCOLOR=white TEXT=crimson>
<CENTER><FONT SIZE=+2 FACE="Helvetica,Arial,Avant
Garde">Aqu&iacute; se muestra como ir a una p&aacute;gina a
trav&eacute;s de la etiqueta<BR>
 &lt;Select&gt; &lt;/Select&gt;.</FONT>
<FORM NAME=menuform>
<SELECT NAME="menu">
<OPTION>Selecciona una zona
<OPTION>Zona Sur
<OPTION>Zona Norte
<OPTION>Zona Oriente
<OPTION>Zona Poniente
<OPTION>Zona Centro
</SELECT>
<INPUT TYPE=button VALUE="Ir a..." onClick="seleccionador();">
</FORM>
</CENTER>
</BODY>
</HTML>
```

El objeto DATE

```
<HTML>
```

```
<HEAD>
<SCRIPT>
<!—
var timerID = null
var timerRunning = false

 function parareloj()
{
        if (timerRunning)
              clearTimeout(timerID)
        timerRunning = false
}

  function empiezareloj()
{
        parareloj()
        reloj()
}

   function reloj ()
        {
        var now = new Date()
        var hours = now.getHours()
        var minutes = now.getMinutes()
        var seconds = now.getSeconds()
        var timeValue = «" + ((hours > 12) ? hours - 12 :
hours)
        timeValue += ((minutes < 10) ? «:0" : «:") + minutes
        timeValue += ((seconds < 10) ? «:0" : «:") + seconds
        timeValue += (hours >= 12) ? « P.M." : « A.M."
        document.clock.face.value = timeValue
        timerID = setTimeout(«reloj(),1000")
        timerRunning=true
        }
// —>
</SCRIPT>
</HEAD>
<BODY ONLOAD="empiezareloj()">
<FORM  NAME="clock" onSubmit="0">
La hora es: <INPUT TYPE=text NAME="face" SIZE=12 VALUE="">
</FORM>
</BODY>
</HTML>
```

Uso de instrucciones exclusivas de Netscape Navigator 3.0

```
<multicol>
<spacer>
<embed>
<font face>
```