Modulo VI.
Web Development

1. HTML.
Elementos de una Página Web

Una página Web está formada por tres componentes básicos: texto, imágenes y enlaces (link), ((Ayuda en línea: link: palabras o imágenes que me permiten por medio de un clic del mouse ir a otra página. Destaquemos que los textos están subrayados y el cursor del mouse cambia de formato cuando lo posicionamos sobre un enlace, convirtiéndose en una manito)).
Al texto se le pueden aplicar los formatos más comunes, como negrita, itálica, subrayado, tipografía, tamaño, color, justificación, etc..
Los textos que hacen de enlace se diferencian del resto porque aparecen subrayados y por lo general en azul.
Las imágenes pueden ser, sólo decorativas o como enlace. Recordemos que nuestra ayuda para identificar texto o imagen que son links, es el mouse y la manito.

Una página Web puede contener, además otros elementos como patterns (patrón) de fondo, GIF animados, frames, tablas de contenido y otros elementos que dependen de los navegadores y sus avances.
((Ayuda en línea. GIF animados: imagen gravada con el formato GIF89a, la cual contiene varias imágenes, que el navegador reproduce en forma secuencial. De esta manera se ven como animaciones)).
((Ayuda en línea. FRAMES: conocido como Marco, nos permite tener una parte

estática, donde por lo general encontramos enlaces a otras páginas, las cuales se presentanen un sector de ella que cambia según la página que se llamó)). Tema que desarrollaremos más adelante.

Organizando el Contenido

Existen modelos para la construcción de los sistemas hipermediales, que fueron apareciendo y agregando características con el correr de los años. En este momento vemos que la organización podrá estar planteada por el programador, pero la navegación es decisión de cada usuario, dependiendo de sus intereses, asociaciones y limitaciones en cuanto al planteo del programador.

• **Árbol**: de la "Home Page" o página principal "cuelgan" las diferentes ramas, las páginas secundarias, que por lo general son opciones del menú. A su vez, cada página secundaria puede funcionar como nodo del árbol del cual se desprenderán otras ramas.

• **Grafo o red**: si bien contamos con una Home Page, en las páginas secundarias de cualquier nivel tenemos otros enlaces a páginas que contienen otro tema, o sea, de otra rama, rompiendo así las jerarquías inferiores.

Es lógico pensar que depende de la complejidad del sitio, de los contenidos y de los elementos que estemos dispuestos a utilizar en las páginas. Si hablamos de Frames, lo relacionamos con una estructura tipo grafo.

Una vez decidida la organización y antes de pasar al "armado" debemos pensar en el aspecto gráfico que tendrán las páginas.

¿Qué imágenes, videos, planillas, logo, etc. tenemos para acompañar los textos?. Aunque nos parezca bonito que una página contenga gran cantidad de imágenes, fondos y videos, no olvidemos que debe viajar por la red y verse en una computadora remota, por lo tanto, trate de evitar incluir archivos de gran tamaño, harán lenta la bajada de su página y es posible que el visitante se canse de esperar, cancelándola.

Entonces, es conveniente que un sitio de Internet sea "liviano y atractivo a la vez" sin descuidar el contenido y su objetivo.

Editores de Páginas Web

Muchas de las aplicaciones habituales están capacitadas para crear este tipo de archivos (HTML), por ejemplo: Word en su versión Office '97, pero también existen programas "especializados", como From Page Express que se incluye en el paquete de Internet Explorer, Netscape Composer, Dreamweave de Macromedia, o Allaire HomeSite.

En realidad hacen la tarea más fácil, ya que encubren el código, que el usuario no necesita conocer. Pero no es tan así, cuando uno quiere darle un toque personal o corregir algún error esto se convierte en una limitación.

Programando en HTML

Ya sabemos que las páginas Web se encuentran "escritas" en un lenguaje hipertextual llamado HTML, generando una serie de líneas de código que, al ser

interpretadas arman la página como usted la ve. Sin embargo no es un lenguaje de programación como el C, C++, Cobol o Pascal, sino un lenguaje descriptivo con el cual es posible codificar un hipertexto tal que los navegadores lo entiendan.
Esta codificación se lleva a cabo a través de marcas, tags, que dan forma al documento. Los archivos html son documentos de texto ASCII (como los txt), con determinadas marcas que funcionan como comandos.
Para generarlo podemos utilizar el block de notas y grabarlo como documento de texto, con extensión HTM o HTML.
Para visualizarla cómo va quedando se carga en cualquier navegador. Al hacer algún cambio o agregarle algún elemento, habrá que "actualizar" su contenido.

Los tags

Son expresiones que se codifican incluyendo la "instrucción" dentro de los símbolos "<" y ">", quedando: <tags>.
No todas las etiquetas actúan de igual manera, algunas sólo se abren y otras se abren y se cierran.
Es como por ejemplo en el Word si se quiere subrayar un texto:
• Presionar el icono de subrayado para activarlo.
• Escribir el texto.
• Presionar de nuevo el icono de subrayado para desactivarlo.
Esto, en Html sería:
• Abrir el tag subrayado.
• Escribir el texto.
• Cerrar el tag subrayado.
Su codificación: <tag> Texto </tag>
De este modo decimos que un tag se abre con <tag> y se cierra con </tag>

Estructura general de una página html

Un programa en HTML posee una estructura que está dada por etiquetas o tags.
- Debe comenzar siempre con la instrucción <HTML> y finalizar con </HTML>.
- Dentro de la página podemos encontrar dos elementos básicos, la cabecera marcada
con <HEAD> y </HEAD>, y el cuerpo de página, delimitado por <BODY> y </BODY>.
La cabecera se utiliza para definir información general sobre la página, pudiendo utilizar las siguientes etiquetas o "meta tags":
<TITLE> Para definir el título de una página.
<ISINDEX> Informa al navegador que el documento está indexado y puede ser examinado buscando a través de palabras clave.
<BASE> Permite definir una dirección URL absoluta que podrá ser utilizada después por enlaces que empleen direcciones URL relativas.
<META> Para incluir información adicional, por ejemplo, el nombre del programador.
<LINK> Para definir una relación entre la página actual y otro documento o

página.
Siendo <TITLE> y <META> LAS MÁS USADAS.
Dentro del cuerpo de la página situaremos todos los elementos, tales como texto con su formato, imágenes, enlaces, etc...
Para comenzar a escribir nuestro código, abrimos un editor, el Edit para DOS, un block de notas para Windows o el que posea en su PC, según la plataforma donde esté trabajando, y tipeamos nuestro código, por ejemplo:

```
<HTML>
<!—Nuestro primer ejemplo de página Web-->
<HEAD>
<TITLE> Esta es nuestra primer página </TITLE>
</HEAD>
<BODY>
<H1> Esta es nuestra primer página </H1>
<BR>
<P> Escribimos nuestro texto </p>
</BODY>
</HTML>
```

Analicemos algunas etiquetas:
<!-- -- > Se utiliza par incluir comentarios dentro de un documento.
<H1> </H1> Para mostrar un texto, que es el título del Texto.
<P> </P> Es para escribir un párrafo.

 Deja una línea en blanco.
Al terminar nuestro código, lo grabamos como un archivo con extensión htm o html.
Para ejecutarlo, realizamos un doble clic sobre él y abrirá el navegador que Ud. Posea en su máquina, visualizando su página.

Formato de página y texto.

Dentro del <BODY> se pueden agregar argumentos que mejorarán la visualización
de su página, por ejemplo el color del fondo y / o de la letra de su texto.
- Colores de página: la manera de codificar los colores es bastante complicada. Cada color está formado por distintas combinaciones de rojo, verde y azúl. Según la cantidad de gotas que mezclemos de cada uno, es el color que obtendremos. El máximo es 255 gotas, el mínimo es 0.
Esta cantidad se expresa en valores hexadecimales, el máximo es FF (255) y el mínimo es 00, (siempre van 2 caracteres por color). Precedido por el carácter “#”, a continuación el valor que corresponda al rojo, verde y azul en ese orden, donde:
000000 es negro.
#FFFFFF es blanco.
Un buen ejercicio puede ser abrir el programa MS-WORD, colocar de fondo un

color que Ud. elija, guardar su documento como HTML y visualizarlo.
En la opción de menú Ver, encontrará Código html, esto le mostrará el código html que Word en forma automática escribió en función de lo que Ud. hizo en el documento.
Fíjese en la combinación escrita y comience a jugar.
Para colocar el color de fondo y texto necesitamos los atributos:
BGCOLOR color de fondo de la página.
TEXT color del texto de la página.
Esto quiere decir que no sólo utilizaremos tags o etiquetas, sino, también atributos o propiedades y argumentos:
<BODY BGCOLOR=”#000000” TEXT=”FFFFFF”>
En esta página aparecerá el fondo negro,
y el texto en blanco
</BODY>
Si queremos separar nuestros textos, podemos colocar una línea horizontal con la etiqueta <HR> de apertura solamente. Por ejemplo:
<BODY>
<HR>
<HR WIDTH = 50%>
</BODY>
<HR> sin parámetros dibuja una línea completa, del ancho de la página.
<HR WIDTH = 50%> dibujará una línea centrada, de tamaño equivalente a la mitad del ancho de la página.
Para cambiar el tamaño del texto.

<BODY>
<H1>Encabezado tamaño 1<H1>
<H2>Encabezado tamaño 2<H2>
<H3>Encabezado tamaño 3<H3>
<H4>Encabezado tamaño 4<H4>
<H5>Encabezado tamaño 5<H5>
</BODY>

Donde el valor 1 nos muestra el máximo tamaño de letra y el valor 5 el mínimo tamaño de letra.
También podemos utilizar:
<B> o <STRONG>, Negrita
<I> o <CITE>, Itálica
<U>, Subrayado
<BLINK>, Parpadeante
<TT>, Letra Teletipo
<BIG>, Agranda el texto muy poco
<SMALL>, Achica el texto muy poco
<KBD>, Teletipo más negrita

<S> o <STRIKE>, Texto tachado
<SUB>, Texto en formato subíndice
<SUP>, Texto en formato superíndice
<PRE>, Deja un renglón en blanco y usa letra teletipo (además incluye salto de línea automático, es decir que no hay que incluir BR sino agregar un [ENTER] para bajar de renglón).
Cabe aclarar que estos tags pueden abarcar una sola palabra, todo un renglón, todo el documento o la parte del texto que ustedes deseen.

Justificación

Puede tomar los valores Left (izquierda), Center (centrado), o Right (derecha).
<CENTER>
Zapatos rotos
Zapatos rotos
Con esa facha, ¿adónde vas?
</CENTER>
Con la etiqueta <HN> también se puede utilizar la justificación. Por ejemplo:
<H3 ALIGN=CENTER> texto </H3>

Colocamos imágenes en la página

Para colocar una imagen en la página utilizamos el tag <IMG> de apertura solamente, acompañado del atributo SRC (source “fuente” donde se encuentra el archivo, ya sea en nuestro disco o en el servidor) y el parámetro es el nombre y extensión del archivo que contiene la imagen:
<IMG SRC=“Imagen.gif”>
Si la imagen está en un subdirectorio se colocará todo el camino, separando con “/”.
<IMG SRC=“Gráficos/Imagen.gif”>
Los parámetros que la acompañan son:
Ubicación:
HSPACE: Mínima distancia que separa la imagen de otro elemento ubicado al costado, ya sea imagen o texto se expresa con pixeles.
VSPACE: Al igual que HSPACE, pero marcando distancias hacia arriba y abajo.
Tamaño:
HEIGHT: altura de la imagen.
WIDTH: ancho de la imagen.
Por ejemplo:
• Colocamos una imagen con borde pero sin separación.
<IMG SRC=”imagen.gif” HEIGHT=40 WIDTH=70 BORDER=3>
• Colocamos una imagen sin borde separada horizontalmente y verticalmente.
<IMG SRC=”imagen.gif” HEIGHT=40 WIDTH=70 HSPACE=10 VSPACE=20>
Ubicación:
ALIGN: con los argumentos LEFT (izquierda), RIGHT (derecha) o CENTER

(centro), justifica la imagen con respecto del texto que la rodea. Esta secuencia necesita el tag <P> </P>.
<P ALIGN=CENTER><IMG SRC=”imagen.gif”></P>
Tenga en cuenta que las imágenes pueden ser GIF o JPG dependiendo de lo que almacene.((A.L- una misma imagen ocupa más espacio si es JPG, por eso se justifica usarlos únicamente para las fotos, donde necesitamos la nitidez).
Una imagen también se puede colocar de fondo de la página, reemplazando al color como vimos anteriormente:
<BODY BACKGROUND=“fondo.gif”>
Si queremos lograr el efecto de despegar el texto del fondo, o sea, cuando desplazamos nuestro texto por la pantalla quede fijo nuestro fondo, agregamos BGPROPERTIES=FIXED, por ejemplo:
<BODY BACKGROUND=“fondo.gif” text=“FFFFFF”
<BGPROPERTIES=FIXED>
Si queremos un texto que se desplace recorriendo el ancho de la pantalla, “marquesina” podemos utilizar el tag <MARQUEE> con los siguientes atributos:
• BEHAVIOR, define el movimiento. Los posible valores son:
scroll: el texto aparece por un lado de la ventana y desaparece completamente por el lado contrario (entra y sale).
Slide: aparece por un lado de la pantalla y se desplaza hasta ocupar su posición definitiva (entra y se queda).
Alternate: el texto se desplaza de lado a lado sin desaparecer (rebota entre los bordes).
Otras cláusulas son:
• BGCOLOR, color de fondo (por donde pasará el texto) #RRVVAA
• DIRECTION: para determinar la direción de desplazamiento (left o right).
• ALIGN, define la alineación: top, middle, o botton
• HEIGHT, define el tamaño de la marquesina y se mide en pixeles o porcentaje de pantalla
• HSPACE, espacio horizontal hasta el próximo objeto
• VSPACE, espacio vertical hasta el próximo objeto
• LOOP, cantidad de veces que pasa por la pantalla; puede ser un número o INFINITE (indica que será una retroalimentación permanente)
• SCROLLDELAY, velocidad de la marquesina (en milisegundos).
• SCROLLAMOUNT: cantidad de pixeles
Entonces un ejemplo puede ser:
<MARQUEE ALIGN = top\ middle\ botton BEHAVIOR = sroll\ slide\ alternative BGCOLOR = “#rrvvaa” DIRECTION = left\ right HEIGHT = “...” HSPACE = num LOOP = “...” SCROLLAMOUNT = num SCROOLDELAY = num VSPACE = num WIDT = “...”> texto </MARQUEE>.
Donde: “...” es un valor numérico entre comillas y num es un valor numérico que no necesita las comillas.
Si su navegador no soporta el tag MARQUEE, el texto se visualiza fijo como si hubiese escrito con el tag <P>.

Los Enlaces

Ya sabemos que uno de los elementos de una página WEB son los “límites” o “enlaces”, lo que lo convierte en Hipertexto. Esto también ofrece interactividad con el usuario, ya que no hay un recorrido predeterminado, esto lo establece cada Usuario.
Según su aspecto podemos distinguir tres tipos de enlaces:
• De texto, el enlace está dado por una palabra o una frace
• Imagen, el enlace es un gráfico, dibujo o imagen
• Mixto, el enlace está representado por una imagen y un texto.
Recuerde que cuando sitúe el cursor sobre un enlace el puntero del ratón cambiará, mostrando una mano.
Según el recurso al cual apuntan distinguimos dos clases:
• Locales, cuando el enlace se hace dentro de la misma página
• Enlaces URL, cuando se llama a otro recurso de la Red, ya sea una página, imagen o archivo especial, (audio, zip, ejecutable para bajar, video, etc.)
Para definir un enlace o “ancla” utilizamos el comando <A>, con la cláusula HREF, la dirección a la cual hace referencia.
Cuando es local:
<A HREF=“# catálogo”><PALIGN=CENTER> consulte nuestro catálogo de productos </P> </A>
La dirección que acompaña a HREF es una palabra, “#” indica que se encuentra en nuestra página actual. Lo que aparece subrayado, enlace, en la frase “consulte nuestro catálogo de productos”.
Hemos marcado el inicio, o sea la partida, nos falta indicar la llegada, o sea, nos falta especificar la ubicación de la referencia # catálogo.
Para ello donde tenemos dicha palabra colocamos:
<A NAME=“catálogo”> <P> catálogo de productos </P> </A>
Cuando es un URL necesita la dirección completa con el nombre del archivo y la palabra o frase de enlace. Por ejemplo:
<A HREF=http://www.latinchat.com> chat </A>
Si el enlace es una imagen:
<A HREF=http://www.yahoo.com> <IMG SRC=“buscar.gif”></A>
La imagen aparecerá en un borde de color. Para evitarlo coloque el atributo BORDER con argumento 0.
<A HREF=http://www.yahoo.com> <IMG SRC=“buscar.gif”BORDER=0> </A>
Otra posibilidad es realizar un enlace a una dirección de correo electrónico.
Cuando vimos los tipos de URL, encontramos mailto para correo, pues éste es el que debemos utilizar:
<A HREF= mailto://noesuar@hotmail.com> noesuar@hotmail.com </A>

Organizando el contenido-Tablas

Hasta ahora, vimos la justificación como una forma posible de organizar texto e imágenes, pero no es suficiente.

Las tablas son matrices formadas por filas, columnas y celdas (al igual que en una hoja de cálculo).
Una tabla no tiene por qué ser regular, o sea, todas las celdas del mismo tamaño, para lograr esto necesitaremos varias propiedades para definirla en HTML, los cuales son optativos.
El tag para crear una tabla es <TABLE>, los parámetros son:
WIDTH= ancho total de la tabla en pixeles o porcentaje de la página.
BORDER= espesor de la línea que se dibuja como borde.
BORDERCOLOR= color del borde de la tabla (formato # RRVVAA).
CELLSPACING= espacio entre celdas de la tabla.
CELLSPADDING= espacio entre el texto y los bordes de las celdas.
BGCOLOR= color de fondo de toda la tabla (formato # RRVVAA)
BACKGROUND= imagen de fondo (solo para Internet Explorer).
Luego de definida la tabla podemos colocarle un título (opcional):
<CAPTION> Título de la tabla </CAPTION>
Las filas y las celdas se definen una vez definida la tabla con las siguientes etiquetas:
<TR>- define cada fila, o sea, cada fila comienza con dicha instrucción. (“table row” o “fila de entrada” TR)
<TD>- Una vez definida la fila se definen las columnas o celdas. Dentro de cada <TD> se incluye el texto, imágenes, enlaces que usted desee y cierra con </TD>.
Tanto <TR> como <TD> permiten utilizar propiedades como la definición de la tabla, lo que permitirá que cada fila o cada celda tenga su propio aspecto:
BGCOLOR es el color de fondo de la fila, formato # RRVVAA
BORDERCOLOR es el color del borde de la celda que estén dentro de esa fila
ALIGN es la justificación del texto, puede llevar LEFT, RIGHT o CENTER
VALIGN es la justificación vertical del contenido que puede ser TOP arriba), CENTER (centrado verticalmente) o BOTTOM (al pie de la tabla).
Por ejemplo:

```
<HTML>
<HEAD>
<TITLE> ejemplo de tablas </TITLE>
</HEAD>
<BODY>
<H1> ejemplo de una tabla de tres columnas y dos filas </H1>
<BR>
<TABLE BORDER= 3>
<TR>
<TD> columna 1 fila 1 </TD>
<TD> columna 2 fila 1 </TD>
<TD> columna 3 fila 1 </TD>
</TR>
<TR>
```

```
<TD> columna 1 fila 2 </TD>
<TD> columna 2 fila 2 </TD>
<TD> columna 3 fila 2 </TD>
</TR>
</TABLE>
</BODY>
</HTML>
```

Las celdas se pueden acotar, o sea, se les puede dar una medida fija, cuidando que la medida total, dada por la suma de las medidas de las celdas coincidan con el ancho de la tabla.
Para esto debemos incluir el parámetro WIDT# dentro del tag <TD>.
Supongamos que definimos una tabla de dos filas, la primera tiene dos columnas, la celda de la izquierda mayor que la de la derecha, en las dos hay texto.
Encontramos 3 celdas cada una de un tamaño, dentro de ellas encontramos imágenes, texto y enlaces.

```
<TABLE BORDER CELLSPACING=1 BORDERCOLOR="#000000"
CELLPADDING=4 WIDTH= 1006>
<TR>
<TD WIDTH= "40%" VALIGN= "TOP" HEIGHT=47>
<P> <FONT SIZE=2> primer celda, fila 1 columna 1 </FONT>
<A HREF= mailto://hgomez@hotmail.com> envía tu mail </A>
</TD>
<TD> WIDTH= "60%" VALIGN= "TOP" COLSPAN=2 HEIGHT=47>
<P> <IMG SRC= "image5.gif" WIDTH=48 HEIGHT=45>
</TD>
</TR>
<TR>
<TD WIDTH= "400" VALIGN= "TOP" HEIGHT=18>
<P> <A HREF= http://www.cordoba.com.ar> córdoba </A>
</TD>
<TD> WIDTH= "516" VALIGN= "TOP" HEIGHT=18>
<P> <A HREF= "# catálogo"> visitá nuestros productos </A>
</TD>
<TD WIDTH= "90" VALIGN= "TOP" HEIGHT=18> <P></P>
</TD>
</TR>
</TABLE>
```

Repasemos lo que vimos y tenemos en el código anterior:
En la primera línea comienza la definición de la tabla con <TABLE> con su configuración a continuación.
Por cada <TR> que define una fila existen tantas <TD> como columnas define.

En la 1º fila definimos el tamaño con porcentajes en WIDTH y en la 2º fila con cantidad de pixeles.
En la 1º celda de la 1º fila hay texto y un enlace a un e-mail. En la 2º celda encontramos una imagen.
En la 2º fila definimos un enlace a otro punto de la misma página en la 2º celda, mientras que en la última celda definimos solo configuración.
Por último tenemos dos atributos más en un tag <TD>
ROWSPAN: indica cuantas filas de la tabla ocupará una celda.
COLSPAN: indica cuantas columnas de la tabla ocupará una celda.

Listas

Es muy común tener que realizar una lista de elementos, conceptos, preguntas, etc..
En una página también podremos, como en un editor de textos.
Existen dos tipos de listas en una página WEB: numeradas y sin numerar.
<OL> </OL> Para marcar distintos niveles. Con la cláusula TYPE colocamos el tipo de viñeta, para las numeradas el argumento a elegir puede ser: A, a, I, i, o 1.
Para las no numeradas puede ser: CIRCLE, SQUARE o DISK.
<UL> </UL> tabula hacia la derecha todo lo que se encierre entre ellas.
<LI> </LI> Para el texto de cada viñeta.
Por ejemplo:

```
<OL TYPE=I>
<LI> clasificación de los productos </LI>
<OL TYPE=1>
<LI> por su uso </LI>
<LI> por su costo </LI>
</OL>
</OL>
```

Se verá:
I-) Clasificación de los productos
1- por su uso
2- por su costo
Si no es numerada:

```
<OL TYPE=SQUARE>
<LI> clasificación de los productos </LI>
<OL TYPE=CIRCLE
<LI> por su uso </LI>
<LI> por su costo </LI>
</OL>
</OL>
```

Se verá:
Clasificación de los productos
• Por su uso
• Por su costo

Por ejemplo, en la siguiente página encontramos una lista dentro de un texto con título:

```
<A NAME="1"></A>
<P><B>¿Qué es Internet?</B>
<BR>Internet es el nombre que se le da a un conjunto de servicios de cobertura mundial que se accesa mediante el empleo de una computadora. También se puede definir como un conjunto de computadoras privadas y públicas entrelazadas, que contienen información variada y que pueden ser consultadas por cualquier persona que cuente con una PC y una clave de acceso a Internet.
</P>
<A NAME="2"></A><P><B>¿Qué servicios existen en Internet?</B><BR>
<P>Los servicios disponibles en Internet son muy variados y diferentes entre sí. Los más comunes son los siguentes:</P>
<UL>
<LI>Correo Electrónico o en sus siglas en inglés E-mail
<LI>WWW o Web
<LI>IRC o Internet Relay Chat
<LI>Foros de discusión o servicios - Forum
<LI>Servidores de Noticias - News
<LI>Carga y descarga de Ficheros - FTP
</UL>
Estos son algunos de los servicios existentes. Existen decenas de los mismos cada uno con
sus peculiaridades y servicios especializados.
</P>
<A NAME="3"></A><P><B>Glosario de los términos más usados en internet.</B><BR>
<P>Para poder consultar el Glosario de Términos, acceda a la siguiente dirección (URL):</P>
<A HREF="http://www.intercom.es/util/ayudas/glosari2.htm">
http://www.intercom.es/util/ayudas/glosari2.htm</A>
</UL>
<P><HR></P>
```

Formularios en la web

Hasta el momento hemos visto cómo mostrar información al usuario y cómo organizarla, pero cómo nos comunicamos con ellos? Sólo por e-mail? No, claramente existen otras posibilidades, como por ejemplo a través de un formulario en la página.

Un formulario es una página Web en la cual el internauta introduce los datos que se solicitan, la cual será recibida en el servidor.

El concepto de formulario es el que ud. conoce de programación visual, en él encuentra objetos, tales como cajas de texto, listas desplegables, de opción, etc...

Cuando se envían estos datos, se transmite utilizando el protocolo HTTP. Los datos se reciben en ASC II y con el formato “nombre = valor”, donde nombre es el nombre del campo y valor es el dato introducido.
Para definir un formulario utilizamos el tag <FORM>, </FORM>. Las dos cláusulas más importantes de esta etiqueta son:
ACTION: determina la acción que se realizará, en una cadena de caracteres con la dirección URL donde se enviará la información. Por ejemplo, si es un e-mail deberá especificar correctamente los datos siendo su formato mailto: contexto@arnet.com
METHOD: determina cómo se transmitirán los datos. Los valores posibles son:
-GET: concatena los campos y valores con el símbolo &.
-POST: los datos aparecen como una “entrada estándar” en el programa o aplicación del servidor.
Tenemos creado el formulario, ahora comenzaremos a definir los objetos que nos permitan ingresar los datos.
Con el tag <INPUT> creamos los campos, el tipo de dato está definido con el valor TYPE. Por defecto es text.
El nombre del objeto se define con la cláusula NAME. El valor del campo se asigna a la cláusula VALUE.
Por ejemplo:
<FORM ACTION=mailto:casatía@arnet.com.ar METHOD= “POST”>
<INPUT TYPE = “TEXT” NAME = “txt Nombre”> Nombre <P>
<INPUT TYPE = “TEXT” NAME = “txt Teléfono”> Teléfono <P>
<INPUT TYPE = “TEXT” NAME = “txt Correo”> Correo electrónico <P>
</FORM>
Colocamos 3 cuadros de texto para ingresar el “Nombre”, “Teléfono” y “e-mail” del usuario, pero no validamos los datos, ni le damos el tamaño.
Por ejemplo:
<INPUT NAME = “ciudad”VALUE = “Córdoba” SIZE = 12 MAXLENGTH = 15>
• Botones: TYPE = submit (para enviar la información).
TYPE = reset (para eliminar el contenido del formulario).
En algunas versiones, TYPE = “BUTTON” (define un botón sin acción asociada).
<INPUT TYPE = reset NAME = “botón” VALUE = “cancelar”>
• Contraseñas: TYPE = password (apareciendo * en pantallas cuando introduzca su clave).
• Casillas de selección: TYPE = checkbox (sin la cláusula VALUE). Cuando sea seleccionada el valor devuelto será “on”. Para que el valor devuelto por defecto sea “on” se coloca CHECKED.
<IMPUT TYPE = check box NAME = “contado” CHECKED>
• Botones de selección: TYPE = radio (para definir el conjunto de botones, se asigna el mismo nombre en NAME).

<INPUT TYPE = radio NAME = “Tipo Factura” VALUE = “A”>
<INPUT TYPE = radio NAME = “Tipo Factura” VALUE = “B” CHECKED>
Por defecto el seleccionado será el que tenga la cláusula CHECKED.
• Imágenes: TYPE = image (imágenes en un formulario que realizan alguna tarea) por ejemplo para enviar los datos como si fuese un botón submit.)
NAME SRC = ruta y/o nombre del archivo de la imagen.
ALIGN para especificar su ubicación.
Para crear cajas de texto de más de una línea debemos utilizar <TEXTAREA> en vez de <INPUT>. Sin TYPE, ROWS y COLS para determinar el tamaño en filas y columnas, siendo todas obligatorias.
<TEXTAREA NAME = “...” ROWS = “...” COLS = “...” ALIGN = top\ middle\ botton\ left\ right DISABLED ERROR = “...”>
Texto <TEXTAREA>.
DISABLAED para desactivar la caja de texto, impidiéndole al usuario agregar o modificar su contenido.
ERROR permite visualizar un mensaje de error si los datos no son correctos.
• Listas: necesita los tags <SELECT> y <OPTION>, una para definir la lista y la otra sus componentes. No cabe duda que es el objeto más complejo, por lo tanto veremos una por vez.
- Con SELECT:
NAME = “...”, SIZE para determinar la cantidad de filas con 1 tenemos una lista desplegable, con un valor >1, tendremos una lista de N filas. Si es MÚLTIPLE nos permitirá seleccionar más de un elemento (como en Windows).
<SELECT NAME = “...” SIZE = 5 MULTIPLE> Opciones </SELECT>
En algunos navegadores podrá utilizar ALIGN, DISABLED, ERROR, SRC, WIDTH, HEIGHT.
- Con OPTION determinamos el texto que aparecerá en la lista, ubicándose entre <OPTION > <|OPTION>
Si queremos codificar las operaciones colocamos el valor a VALUE.
<OPTION VALUE = “1”> es el primero <\OPTION>
Si queremos que aparezca uno seleccionado por defecto utilizamos la cláusula SELECTED.
Como conclusión tenemos por ejemplo:
<SELECT NAME = “carrera” SIZE = 1>
<OPTION VALUE = “1”> Diseño Gráfico </OPTION>
<OPTION SELECTED VALUE = “2”> Informática </OPTION>
</SELECT>

2. Programando en la red del lado del cliente

Todo lo que aprendimos de HTML nos permite crear páginas que permiten cierta interactividad, pero en el uso de formularios, por ejemplo, los datos se validan recién cuando llegan al servidor.

En un esfuerzo por aumentar la interactividad se incorporó el diseño en HTML funciones basadas en código de lenguajes de programación.
Para programar se necesita de la etiqueta <SCRIPT> </SCRIPT>
Los SCRIPT más utilizados son:
• VBScript: lenguaje de programación basado en Visual Basic, orientado a sucesos, es decir, el código se ejecuta en respuesta a una interacción del usuario con el interfaz gráfico de usuario o GUI. Pero no todos los navegadores y versiones son compatibles con este lenguaje.
• JavaScript: es un lenguaje muy similar VBScript en sus funciones pero distinto en su sintaxis.
De hecho, la programación para navegadores, es de utilidad para realizar validaciones de datos del lado del cliente antes de enviar el formulario al servidor.
En este libro desarrollaremos parte del lenguaje JavaSscript. Tenga en cuenta que habrá conceptos de programación que ud. deberá manejar para hacer aplicaciones más complejas.
JavaScript es un lenguaje de guiones basado en JAVA. ((A.L.. Lenguaje de programación que desarrollaremos más adelante)) ideal para desarrollar pequeñas funciones de control dentro de las páginas Web.
Algunas de sus características son:
• Es un lenguaje interpretado por el cliente.
• Está orientado a objetos, utilizando objetos ya construidos.
• El código se encuentra integrado dentro de la página Web.
• No es necesario especificar los tipos de datos de las variables, ni su declaración previa.
• Se comprueba su validez cuando se ejecuta la función.

Funciones en JavaScript

Una función es un procedimiento (programa) escrito en JavaScript, es decir, un conjunto de sentencias que realizan una determinada tarea.
Una función consta de las siguientes partes básicas:
• Un nombre de función.
• Los parámetros pasados a la función separados por comas y entre paréntesis.
• Marcar el inicio y el final de la función entre llaves ({})
((A.L. Parámetros: valores que recibe la función, como constantes o variables.))
No es lo mismo definir una función que llamarla.
Por ejemplo:
• Definimos una mascota que tenemos con:
Nombre
Todas sus características.
Función que cumple.
• Para llamarla, sólo la nombramos.
Definir una función es simplemente especificar su nombre y definir qué acciones realizará

en el momento en que sea invocada. Para ello se emplea la palabra reservada function.

```
{
.....
return <valor_retorno>
}
```

Cuando llamamos a esta función realizará las acciones especificadas con los parámetros indicados.

Nombre_de_la_función(a,b);

Las funciones pueden definirse en cualquier parte del documento pero es aconsejable declararlas en la cabecera; de esta forma se garantiza que todas las funciones se carguen antes se que el usuario tenga la oportunidad de realizar ninguna acción en la que llame a una de ellas.

El código del SCRIPT se escribe en la cadena. Por ejemplo, si queremos presentar un texto:

```
<html>
<head>
  Mi primer JavaScript!
</head>
<body>
<br>
<br>
<script language="JavaScript">
document.write("Esto es JavaScript!")
</script>
<br>
</body>
</html>
```

La instrucción document.write() agrega lo que se incluye entre comillas al documento HTML, en el lugar donde fue llamado. Puede ser tanto texto como una imagen, si se incluye dentro del paréntesis '<IMG SRC="imagen.gif">'.

Si queremos ejercer control sobre un botón, necesitamos eventos, que irán en la definición del objeto dentro del BODY. Por ejemplo, escribimos el clásico "Hola Mundo!".

```
<HTML>
<HEAD>
<TITLE> Ejemplo en JavaScript del programa Hola Mundo </TITLE>
<SCRIPT LANGUAJE = "JavaScript">
<!--
function pulsame ( ) // function, palabra reservada obligatoria,
pulsame es el nombre de la función.
```

```
{ // Representa el inicio de la función.
alert( “Hola Mundo “);
return true; //Retorna verdadero.
}
-->
</SCRIPT>
</HEAD>
<BODY BGCOLOR = “WHITE”>
<FORM>
// Definimos un botón para un evento.
```

Usa este método para consultar al usuario una decisión que requiere una respuesta del tipo
SI/NO. El mensaje que se pasa como argumento determina la pregunta a realizar al usuario.
La función devuelve un valor de retorno en función de la respuesta del usuario: TRUE si el usuario confirma y FALSE en caso contrario.
Recordemos que un “evento” es una reacción que se produce, generalmente, como respuesta a una acción del usuario sobre alguno de los objetos de la página. Algunos eventos son por ejemplo, un click de ratón, la focalización de un campo en un formulario, modificar un campo de texto o mover el cursor.
Cada evento es reconocido por ciertos objetos, etiquetas HTML, como son:
• Focalización, desfocalización y edición: campos de texto, textareas y selections.
• Clicks: buttons,radio buttons, checkboxes, submit buttons, reset buttons, y enlaces.
• Selección: text fields, textareas.
• Señalización: enlaces.
A partir de la versión 3.0 de Netscape, onBlur y onFocus se aplican tambien a ventanas y framesets.

Eventos que se pueden controlar en los objetos de una página Web:

Evento	Manejador	Descripción
Click	*onClick*	El usuario pulsa sobre un elemento de form o un enlace
Señalización	*onMouseOver*	El usuario coloca el ratón sobre una determinada area
Carga	*onLoad*	El usuario carga la página
Descarga	*onUnload*	Es usuario sale de la página
Focalización	*onFocus*	El usuario selecciona un elemento de un form como entrada
Desfocalización	*onBlur*	El usuario quita la selección de un elemento de un formulario como entrada
Edición	*onChange*	El usuario cambia el valor de un elemento text, textarea o select de un formulario
Selección	*onSelect*	El usuario selecciona el campo de entrada de un formulario
Interrupción	*onAbort*	El usuario interrumpe la carga de una imagen
Error	*onError*	La carga de un documento o imagen produce un error.
Des-señalización	*onMouseOut*	El usuario saca el ratón de un área (imagemap) o enlace.
Inicialización	*onReset*	El usuario pulsa el botón de reset en un form.

Algunos ejemplos sencillos son:
• Obtener el nombre del navegador que está utilizando.

```
<SCRIPT LANGUAJE = “JavaScript”>
function que_browser( ) // La función se llama que_browser
{
var browser= navigator.appName;
document.write (“Su navegador es: “ + browser)
}
</SCRIPT>
```

* La 1er. Línea de la función define la variable browser y almacena el nombre del navegador.
* La 2da. Línea de la función escribe en la pantalla la cadena resultante de la concatenación
del literar con el contenido de la variable.
Recuerde llamar la función desde el BODY, con las etiquetas <SCRIPT>, </SCRIPT>
y el nombre de la función.
•
Cambiar el color de fondo por medio de botones.

```
<SCRIPT LANGUAJE = “JavaScript”>
function fondo(color ) { // La función fondo recibe el parámetro color.
document.bgColor = color
}
</SCRIPT>
<BODY TEXT = “blanck”>
<CENTER>
Elegí el color de fondo
<FORM NAME = “Colores”>
// Llama la función fondo y envía el color.
<INPUT TYPE=”button” VALUE=”Amarillo” onClick=”fondo( ‘yellow’)”>
<INPUT TYPE=”button” VALUE=”Rojo” onClick=”fondo( ‘read’)”>
</FORM>
</CENTER>
</BODY>
</HTML>
```

• *Ingresar en una caja de texto y mostrarla en una ventana.*

```
<html>
<head>
<script language="JavaScript">
<!-- esconde el script de viejos browsers
function getname(str) {
alert("Hola "+ str+"!");
}
// end hiding contents -->
</script>
</head>
<body>
```

Por favor, escriba su nombre:

```
<form>
<input type="text" name="name" onBlur="getname(this.value)"
value="">
</form>
</body>
</html>
```

Otra vez tenemos nuevos elementos implementados en este script:
El 'onBlur' en el tag <input> le dice al programa cliente que función es la que tiene que invocar cuando algo se introduce en esta forma o casilla.
La función 'getname(str)' será invocada cuando usted deja en blanco esta forma ('leave') o presiona 'enter' sin haber escrito nada.

'This.value' significa el valor que usted escribió en la forma.
• Mostrar una ventana cuando se pasa por un enlace.

```
<html>
<head>
<script language="JavaScript">
<!-- Escondemos la funcion
function hello() {
alert("Hola!");
}
// -->
</script>
</head>
<body>
<a href="" onMouseOver="hello()">link</a>
</body>
</html>
```

Esto es completamente fácil. Usa el método 'onMouseOver' y la función hello() es invocada cuando este evento ocurre.
• Mostrar mensajes en la barra de estado.
Con botones:

```
<html>
<head>
<script language="JavaScript">
<!--
function statbar(txt) {
window.status = txt;
}
// -->
</script>
</head>
<body>
<form>
<input type="button" name="look" value="Escribir!"
onclick="statbar('Hola! esta es la barra de estado!');">
<input type="button" name="erase" value="Borrar!"
onclick="statbar('');">
</form>
</body>
</html>
```

Creamos dos botones, ambos invocan la función statbar(txt). El 'txt' en

los paréntesis muestra que la función toma una variable. Esta función es invocada desde el tag <form>, escribimos el texto que queremos que nos muestre la barra de estado.
La función es invocada y define la variable txt esta toma el valor que se colocó entre comillas simples ‘ en los paréntesis cuando llamo a la función. De esta forma cuando se oprime el boton 'Escribir!', la función statbar(txt) es invocada y txt conserva el valor ('string') 'Hola! esta es la barra de estado!'.
Observe el segundo botón. Este invoca la misma función, si no tuvieramos la variable, necesitariamos dos funciones diferentes.
Ahora qué hace la función statbar(txt)? Bueno, esto es simple. Usted solo escribe los contenidos de txt a la variable window.status. Esto es hecho por window.status = txt;. Escrbiendo un valor nulo ('') hace que la barra se borre.
Note que tenemos que usar las comillas simples ' porque usamos comillas dobles " para definir onClick. El browser necesita saber a quien le pertenecen las comillas, por eso se tiene que alternan con comillas sencillas y dobles.
<a href="tpage.html" onMouseOver="window.status='Otro link tonto...';
return true">
Con enlace:
Ensaye el siguiente script : una página con un link aquí. Sólo mueva el puntero sobre él no haga click!

```
<html>
<head>
<script language="JavaScript">
<!--
function moveover(txt) {
window.status = txt;
setTimeout("erase()",1000);
}
function erase() {
window.status="";
}
// -->
</script>
</head>
<body>
<a href="página.htm"
onMouseOver="moveover('Desapareciendo!');return true;">
aquí</a>
</body>
</html>
```

Usted reconocerá muchas partes de este script. La función

moveover(txt) es solo la función statbar(txt) después de algún trabajo de copiar y pegar sobre ella! Algo similar sucede con la función erase().
En el <body> de la página de HTML creamos un link con la ya conocida propiedad onMouseOver. Nuestra función moveover(txt) es invocada con la string 'Desapareciendo!' pasándola por encima de la variable txt.
El window.status toma los contenidos de txt. Lo mismo que la función statbar(txt).
Aunque la invocación de la función setTimeout(...) es nueva. Esta función configura un tiempo de finalización. La función setTimeout(...) define una hora de arranque y una hora de finalización.
En nuestro ejemplo la función erase() es invocada después de 1000 milisegundos (un segundo).
Su función moveover(txt) termina luego de que el tiempo es configurado.
El browser invoca la función erase() automáticamente luego de un segundo.
Luego de que el tiempo de finalización a concluido, el tiempo de inicio no arranca de nuevo, claro que se puede invocar otra vez con la función erase().
Para aplicaciones más complejas dijimos que necesitamos saber programar, ya que utilizaremos las estructuras lógicas de programación, operadores, creación de objetos, etc...

Algo más sobre la programación en JavaScript

Valores y variables

Reconoce cuatro tipos de valores: numéricos, lógicos, cadena y null. Cuando necesitamos una variable la definimos en el lugar con:
Var edad=12
Podemos cambiar su tipo asignándole un valor de distinto tipo:
Edad = “doce”
Para concatenar, utilizamos “+”, permitiendo datos de distintos tipos.
Si necesitamos convertir una cadena en números:
T=parseInt(“100”) + 25
O usamos parseFloat(decimal).

Los operadores

Reconoce todos los del lenguaje C.
• Aritméticos: +, -, *, /, %
• Asignación: =, +=, -+, *=, /=, %=
• Relacionales: ==, !=, <, >, <=, >=
• Lógicos: &&, !!, !

Sentencias de JavaScript

Se dividen en cuatro grupos: condicionales, repetitivas, manipulación de objetos y comentarios.

• **Comentarios**: Se realizan de la misma forma que en C o Java, // para una línea, /* */
para un párrafo de más de una línea.

• **Condicionales**: Tenemos sólo el IF
Es la sentencia de toma de decisiones.
Sintaxis:

```
if (condicion){
sentencias1
}
// no se da la condición
else {
sentencias2
}
```

Argumentos:

• condición: Cualquier expresión válida que tras ser evaluada devuelve el valor true o false.
• sentencias1: Acciones a ejecutar si se cumple la condición.
• sentencias2: acciones a ejecutar si el resultado de evaluar condición es falso.
•

Repetitivas: Dispone de dos, For y While, y las instrucciones breck y continue dentro
de ellas.

Sentencia while

Esta sentencia es la más simple para realizar bucles sujetos a una condición. Evalúa una espresión y si es cierta ejecuta el bloque de sentencias.
Sintaxis:

```
while (condición) {
sentencias
}
```

Argumentos:

• condición: Sentencia evaluada en cada iteración.
• sentencias: Conjunto de sentencias que se ejecutarán mientras el valor de condición sea cierto.

Funcionamiento:

• Se evalua la condición y si es cierta se ejecuta las sentencias incluidas entre llaves.
• Si la condición inicial no es cierta nunca, no se producirá ninguna iteración.
• Si la condición inicial es cierta siempre, se ejecutará un bucle infinito. Por ejemplo:

```
while (true)
{
```

...
}

Sentencia for

Esta sentencia crea un bucle que consta de tres expresiones op ionales, encerradas entre paréntesis y separadas por ";" (sin espacios en blanco), seguidos por las sentencias a ejecutar en el bucle.

Sintaxis:

for ([expresión de arranque;][condición de mantenimiento;][expresión de incremento]) {
sentencias
}

Argumentos:

• expresión inicial: es una sentencia o una declaración de variables. Su uso más común es el de inicializar una variable de tipo contador.
• Condición de mantenimiento: es la condición a evaluar en cada iteración. Es opcional y si se omite el resultado de la evaluación es siempre cierto.
• Expresión de incremento: Generalmente se utiliza para actualizar o incrementar el valor del contador.
• Sentencias: es un bloque de sentencias que son ejecutadas mientras se cumple la condición de mantenimiento.

Funcionamiento:

1. Ejecución de la expresión de arranque. Usualmente se inicializan uno o más contadores
2. Se evalua la condición expresada en condición de mantenimiento.
• Si el valor de retorno de la condición es true (la condición se cumple) el bucle se ejecuta, es decir, se ejecutan las sentencias incluidas en el bucle (rodeadas por llaves).
• Si el valor es falso (la condición de mantenimiento no se cumple) y el bucle acaba
• .
3. Tras cada iteración se ejecuta la expresión de incremento.
4. Tras ejecutar las sentencias se devuelve el control al paso 2.

Sentencia break

Esta sentencia interrumpe la ejecución de un bucle while o for y continúa la ejecución del programa en la sentencia siguiente al bucle.

Sintaxis:

break

Sentencia continue

La sentencia continue termina la presente iteración de un bucle for o while y pasa a la siguiente

Continúa hasta que no acaba completamente la ejecución del bucle.

Sintaxis:

continue

Funcionamiento

• En un while salta a la evaluación de la condición.
• En un for, salta a la expresión de actualización.

Manipulación de objetos:

Los objetos pueden ser construidos por el programador, formar parte del navegador o tratarse de objetos de JavaScript.

• NEW (Operador que se utiliza para crear nuevos objetos)

NuevoObjeto = new Objeto(parámetros)

Por ejemplo, un nuevo objeto fecha.

NuevaFecha = new Date()

• THIS (Para hacer referencia al objeto actual)

THIS.nombre // donde nombre es una propiedad de un objeto creado.

• WIDTH (Se utiliza para establecer un objeto por defecto con el cual se operará)

```
WIDTH (objeto) {
Instrucciones
}
```

Por ejemplo: Utilizamos el objeto Math de JavaScript para calcular el resultado de multiplicar R por el cosena de PI.

```
WIDTH (Math) {
X = R * COS(PI)
}
```