Guía de HTML

Contenidos:

1.- ¿Qué es el HTML?

El **HTML** (HiperText Markup Language) es el lenguaje utilizado para representar documentos en la WWW (World Wide Web). Además de texto normal incluye también, elementos multimedia (gráficos, vídeo, audio) y existen enlaces (links) que permiten saltar a otras partes del documento o a otro sitio cualquiera de Internet. Otra característica muy importante de este lenguaje es que es **portable**, es decir, se pueden visualizar las páginas con cualquier sistema operativo y, por supuesto también crearlas.

Las **etiquetas** constituyen la filosofía de este lenguaje. Por medio de ellas se pueden controlar los elementos tipográficos del texto: tipo, color y tamaño de las fuentes, el estilo (negrita, cursiva, etc), así como también la inclusión de tablas, listas, formularios, la inserción de fotos, sonidos, fondos.

Las etiquetas se pueden modificar por medio de sus **atributos**, éstos son del tipo atributo="valor" y se colocan detrás del nombre de la etiqueta. El nombre de la etiqueta y sus atributos se colocan entre los símbolos **<** y **>** y normalmente se usan dos, una de inicio y otra final, para conseguir el efecto deseado.

Por ejemplo si escribimos
`<FONT COLOR="#ff0000" size="2">`El texto se verá rojo y en tamaño un poco menor de lo normal `</font>` Se verá como El texto se verá rojo y en tamaño un poco menor de lo normal. En la actualidad el uso de la etiqueta `<font>` está descontinuado. Para aplicarle estilos al texto se utilizan herramientas como el lenguaje CSS.

El uso de estas etiquetas, y por ende el aprendizaje del HTML, no es difícil. Precisamente el objetivo de esta pequeña guía es servir de introducción y referencia de las características más usadas del **HTML**

Estructura Interna de una Página HTML

```
<html>
  <head>
  <title> titulo de la página</title>
  </head>
  <body>
Cuerpo de la página.
  </body>
</html>
```

Si observamos toda página comienza con la marca:`<html>` y finaliza con la marca:`</html>`. Todo el texto que dispongamos dentro del `<body>` aparece dentro del navegador tal cual lo hayamos escrito.

Otra cosa importante es que el lenguaje HTML no es sensible a mayúsculas y minúsculas, es decir podemos escribirlo como más nos guste, además no requiere que dispongamos cada marca en una línea (podríamos inclusive escribir toda la página en una sola línea! cosa que no conviene ya que somos nosotros quienes tendremos que modificarla en algún momento).

**2.- Salto de línea
**

Para indicarle al navegador que queremos que continúe en la próxima línea debemos hacerlo con el elemento HTML `<br>`. Cuando aparece la marca `<br>` el navegador continúa con el texto en la línea siguiente. Es uno de los pocos elementos HTML que no tiene marca de cerrado como habíamos visto hasta ahora. `<br>` viene de break.

3.- Párrafo <p>

Un párrafo es una oración o conjunto de oraciones referentes a un mismo tema. Todo lo que encerremos entre las marcas `<p>` y `</p>` aparecerá separado por un espacio con respecto al próximo párrafo. Dentro de un párrafo puede haber saltos de línea `<br>`.

Veamos con un ejemplo como disponer dos párrafos:

```
<html>
<head>
</head>
<body>
<p>
SQL, Structure Query Language (Lenguaje de Consulta Estructurado) es un lenguaje de
programacion para trabajar con base de datos relacionales como MySQL, Oracle,
etc.<br>MySQL es un interpretador de SQL, es un servidor de base de datos.<br>MySQL
permite crear base de datos y tablas, insertar datos, modificarlos, eliminarlos,
ordenarlos, hacer consultas y realizar muchas operaciones, etc., resumiendo: administrar
bases de datos.
</p>
<p>
Esta guia tiene por objetivo acercar los conceptos iniciales para introducirse en el
mundo de las bases de datos.
</p>
</body>
</html>
```

Tenemos en esta página HTML dos párrafos, cuando el navegador interpreta esta página, separa los contenidos de los dos párrafos con un espacio horizontal. Además el primer párrafo contiene varios saltos de línea. Normalmente uno agrupa en párrafos para dar más sentido a nuestro escrito.
Cuando modificamos la ventana del navegador los párrafos se acomodan automáticamente de acuerdo al ancho de la ventana. `<p>` viene de paragraph.

4.- Títulos <h1><h2><h3><h4><h5><h6>

Otros elementos HTML muy utilizados son los que nos sirven para indicar los títulos, para esto contamos con los elementos:
`<h1><h2><h3><h4><h5><h6>`

El título de mayor nivel es `<h1>`, es decir el que tienen una fuente mayor Confeccionaremos una página que contenga un título de primer nivel <h1> y luego dos títulos de nivel `<h2>`. Definiremos un párrafo para cada título de segundo nivel.

```
pagina1.html
<html>
<head>
</head>
<body>
<h1>Tipos de datos en MySQL</h1>
<h2>varchar</h2>
<p>
Se usa para almacenar cadenas de caracteres. Una cadena es una secuencia de caracteres.
Se coloca entre comillas (simples): 'Hola'.<br> El tipo "varchar" define una cadena de
longitud variable en la cual determinamos el máximo de caracteres. Puede guardar hasta
255 caracteres. Para almacenar cadenas de hasta 30 caracteres, definimos un campo de tipo
varchar(30).
</p>
<h2>int</h2>
<p>
```

```
Se usa para guardar valores numéricos enteros, de -2000000000 a 2000000000
aproximadamente.<br> Definimos campos de este tipo cuando queremos representar, por
ejemplo, cantidades.
</p>
</body>
</html>
```

Cada título aparece siempre en una línea distinta, no importa si lo tipeamos seguido en el archivo, es decir el resultado será igual si hacemos:

```
<h1>Tipos de datos en MySQL</h1>
<h2>varchar</h2>
```

o esto:

```
<h1>Tipos de datos en MySQL</h1><h2>varchar</h2>
```

El navegador dispone cada título en una línea nueva. `<h1>` viene de heading que significa título.

5.- Énfasis ( )

Enfatizar algo significa realzar la importancia de una cosa, por ejemplo una palabra o conjunto de palabras. Así como tenemos seis niveles de títulos para enfatizar un bloque contamos con dos elementos que son (`<em>` `<strong>`) El elemento de mayor fuerza de énfasis es `strong` y le sigue `em`. La mayoría de los navegadores muestran el texto enfatizado con `strong` con un texto en negrita y para el elemento `em` utilizan letra itálica. Otra cosa importante que podemos hacer notar es que estos elementos HTML no producen un salto de línea como los de título (h1,h2 etc.)

Ejemplos:
`<em>` viene de empathize que significa énfasis.
`<strong>` viene de strong que significa fuerte.

6.- Hipervínculo a otra página del mismo sitio <a>

`<a>` viene de anchor que significa ancla.

El elemento más importante que tiene una página de internet es el hipervínculo, estos nos permiten cargar otra página en el navegador. Normalmente un navegador al encontrar esta marca muestra un texto subrayado, y al hacer clic con el mouse sobre éste el navegador carga la página indicada por dicho hipervínculo.

La marca de hipervínculo a otra página del mismo sitio tiene la siguiente sintaxis:

```
<a href="pagina2.html">Noticias</a>
```

Implementemos dos páginas que contengan hipervínculos entre si, los nombres de las páginas HTML serán: pagina1.html y pagina2.html

pagina1.html	pagina2.html
<html>	<html>
<head>	<head>
</head>	</head>
<body>	<body>

```
<h1>Página principal.</h1>
<a href="pagina2.html">Noticias</a>
</body>
</html>
```

```
<h1>Noticias.</h1>
<a href="pagina1.html">Salir.</a>
</body>
</html>
```

El valor de la propiedad href en este caso es pagina2.html (es otro archivo HTML que debe encontrarse en nuestro sitio y en el mismo directorio).

El segundo archivo pagina2.html tiene un hipervínculo a la primera página:

```
<a href="pagina1.html">Salir.</a>
```

7.- Hipervínculo a otro sitio de internet <a>

La sintaxis para disponer un hipervínculo a otro sitio de internet es:

```
<a href="http://www.google.com">Buscador Google</a>
```

Ahora la propiedad `href` la inicializamos con el nombre del dominio del otro sitio. Algo importante que hay que anteceder al nombre del dominio es el tipo de protocolo a utilizar. Cuando se trata de una página de internet, el protocolo es el http.

8.- Imágenes dentro de una página <img>

Para insertar una imagen dentro de una página debemos utilizar el elemento HTML `<img>`, la misma no tiene una marca de finalización (similar a la marca). Generalmente, la imagen se encuentra en el mismo servidor donde se almacenan nuestras páginas HTML. Los formatos clásicos son los archivos con extensiones gif, jpg y png.

La sintaxis de esta marca es:

```
<img src="foto1.jpg" alt="Pintura geométrica">
```

Como mínimo, debemos inicializar las propiedades `src` y `alt` de la marca HTML `img`.

En la propiedad `src` indicamos el nombre del archivo que contiene la imagen (en un servidor Linux es sensible a mayúsculas y minúsculas por lo que recomiendo que siempre utilicen minúsculas para los nombres de archivos).

Como la imagen se encuentra en el mismo directorio donde se almacena la página HTML, con indicar el nombre de archivo basta (no es necesario indicar ninguna ruta de carpetas).

Otra propiedad obligatoria es `alt`, donde disponemos un texto que verán los usuarios que visiten el sitio con un navegador que sólo permite texto (o con un navegador que tenga desactivada la opción de descarga de imágenes). El texto debe describir el contenido de la imagen.

Si la imagen se encuentra en una subcarpeta llamada imágenes, luego la sintaxis para recuperarla será:

```
<img src="imagenes/foto1.jpg" alt="Pintura geométrica">
```

Es decir, antecedemos al nombre de la imagen el nombre de la carpeta y la barra /

Si la imagen se encuentra en una carpeta padre de donde se encuentra la página HTML luego la sintaxis será:

```
<img src="../foto1.jpg" alt="Pintura geométrica">
```

Es decir, le antecedemos y la barra / al nombre de la imagen

Si queremos establecer ancho y alto utilizaremos los atributos `width` y `height`,
Ejemplo:

```
<img src="../foto1.jpg" alt="Pintura geométrica" width=”100” y height=”100”>
```

`<img>` viene de image
`src` viene de source
`alt` viene de alternative

9.- Hipervínculo mediante una imagen <a> y <img>

Como ya conocemos los hipervínculos y como insertar imágenes en nuestra página, ahora podemos implementar un hipervínculo pero en vez de mostrar un texto mostraremos una imagen.

La solución es simple y consiste en disponer la marca `<img>` encerrada entre la marca de comienzo y fin del enlace `(<a>)`

Las imágenes se encuentran en una carpeta llamada imágenes que depende directamente de la raíz del sitio:

```
<html>
<head>
</head>
<body>
<h2>Presione alguna de las imagenes para conocer más sobre esa obra.</h2>
<a href="pagina2.html"><img src="/imagenes/foto1.jpg" alt="Pintura Geométrica"></a>
<br>
<a href="pagina2.html"><img src="/imagenes/foto2.jpg" alt="Pintura Geométrica"></a>
</body>
</html>
```

Lo que debe quedar bien en claro es que las imágenes se encuentran en un directorio llamado imágenes en la raíz del sitio. Es bueno practicar con esto ya que cuando implementemos sitios muy grandes seguramente agruparemos cada módulo en distintas carpetas.

10.- Hipervínculo a un cliente de correo <a>

El elemento "a" permite direccionar un hipervínculo a un programa de envío de correos que tengamos configurado en nuestra computadora.

```
<html>
<head>
</head>
```

```
<body>
<h1>Reclamos</h1>
<a href="mailto:guardialara@gmail.com">Enviar mail.</a>
</body>
</html>
```

Cuando se presiona el enlace se abre el programa de envío de correos que tiene configurado el equipo y dispone como receptor del mensaje la dirección que configuramos en el propio enlace seguido de la palabra mailto:

La sintaxis para disponer un título por defecto y un cuerpo de mensaje es:

```
<a href="mailto:diegoestevanes@gmail.com?subject=título del mensaje&body=cuerpo del
mensaje">Enviar mail.</a>
```

Es decir luego de especificar el destinatario del mail disponemos un caracter de interrogación '?' seguido la palabra subject, un igual y el título por defecto que debe aparecer en la ventana de envío de mail. Por último separamos con un ampersand '&' la inicialización de `subject` y el `body` (es decir el cuerpo del mensaje)

Podemos inclusive añadir el envío de mail con copia y con copia oculta a otras direcciones:

```
<html>
<head>
</head>
<body>
<h1>Reclamos</h1>
<a href="mailto:diego1@gmail.com?subject=aquí el
título&cc=diego2@gmail.com&bcc=diego3@gmail.com&body=Este
es el cuerpo">Enviar mail.</a>
</body>
</html>
```

En este ejemplo enviamos un mail a diego1@gmail.com, con copia a diego2@gmail.com y con copia oculta a diego3@gmail.com

11.- Anclas llamadas desde la misma página.

Otra posibilidad que nos brinda el HTML es disponer una referencia dentro de la página para poder posteriormente disponer un hipervínculo a dicha marca. Es una práctica común cuando queremos desplazarnos dentro de una página de gran tamaño. Se disponen hipervínculos a diferentes anclas.

La sintaxis para definir un ancla es:

```
 <a name="nombreancla"></a>
```

No debemos confundir un ancla con un hipervínculo, más allá que se utiliza el mismo elemento a. Para un ancla inicializamos la propiedad `name` con el nombre del ancla.

Ahora la sintaxis para ir a un ancla desde un hipervínculo es la siguiente:

```
<a href="#nombreancla">Introducción</a>
```

Vemos que en la propiedad `href` indicamos el nombre del ancla.

Haremos un ejemplo, donde dispondremos una lista de hipervínculos que llaman a una serie de anclas dispuestas en la misma página:

```
<html>
<head>
</head>
<body>
<h1>Tutorial de MySQL</h1>
<a href="#introduccion">Introducción</a><br>
<a href="#mostrarbasedatos">show databases</a><br>
<a href="#creaciontabla">Creación de una tabla y mostrar sus campos</a><br>
<a href="#cargarregistros">Carga de registros a una tabla y su recuperación</a><br>
<a name="introduccion"></a>
<h2>Introducción</h2>
<p>
SQL, Structure Query Language (Lenguaje de Consulta Estructurado) es un lenguaje
de programacion para trabajar con base de datos relacionales como MySQL, Oracle,
etc.<br>
</p>
<a name="mostrarbasedatos"></a>
<h2>show databases</h2>
<p>
Una base de datos es un conjunto de tablas.<br>
</p>
<a name="creaciontabla"></a>
<h2>Creación de una tabla y mostrar sus campos</h2>
<p>
Una base de datos almacena sus datos en tablas.<br>
</p>
<a name="cargarregistros"></a>
<h2>Carga de registros a una tabla y su recuperación</h2>
<p>
Usamos "insert into". Especificamos los nombres de los campos entre
paréntesis y separados por comas y luego los valores para cada campo, también
entre paréntesis y separados por comas.<br>
</p>
</body>
</html>
```

Cada hipervínculo hace referencia a un ancla que se encuentra en la misma página:

```
<a href="#introduccion">Introducción</a><br>
<a href="#mostrarbasedatos">show databases</a><br>
<a href="#creaciontabla">Creación de una tabla y mostrar sus campos</a><br>
<a href="#cargarregistros">Carga de registros a una tabla y su recuperación</a><br>
```

Luego la definición de las anclas son:

```
<a name="introduccion"></a>
<h2>Introducción</h2>
```

Como podemos observar la definición del ancla se hace inmediatamente anterior al título donde queremos que el navegador se sitúe.

12.- Anclas llamadas desde otra página.

También es perfectamente válido la llamada a anclas desde otra página (no importa si se encuentra en el mismo sitio o en otro). Debemos conocer el nombre de la página a llamar y el nombre del ancla, luego la sintaxis para la llamada al ancla es:

```
<a href="pagina2.html#introduccion">Introducción</a>
```

Es decir luego del nombre de la página que llamamos disponemos el caracter # y seguidamente el nombre del ancla.

pagina1.html

```
<html>
<head>
</head>
<body>
<h1>Tutorial de MySQL</h1>
<a
href="pagina2.html#introduccion">Introducción
</a><br>
<a href="pagina2.html#mostrarbasedatos">show
databases</a><br>
<a href="pagina2.html#creaciontabla">Creación
de una tabla
y mostrar sus campos</a><br>
<a href="pagina2.html#cargarregistros">Carga
de registros a una
tabla y su recuperación</a><br>
</body>
</html>
```

pagina2.html

```
<html>
<head>
</head>
<body>
<h1>Tutorial de MySQL</h1>
<a
href="#introduccion">Introducción</a><br>
<a href="#mostrarbasedatos">show
databases</a><br>
<a href="#creaciontabla">Creación de una
tabla y mostrar sus campos</a><br>
<a href="#cargarregistros">Carga de
registros a una tabla y su
recuperación</a><br>
<a name="introduccion"></a>
<h2>Introducción</h2>
<p>
SQL, Structure Query Language (Lenguaje de
Consulta Estructurado) es un lenguaje de
programacion para trabajar con base de
datos relacionales como MySQL, Oracle,
etc.<br>
</p>
<a name="mostrarbasedatos"></a>
<h2>show databases</h2>
<p>
Una base de datos es un conjunto de
tablas.<br>
</p>
<a name="creaciontabla"></a>
<h2>Creación de una tabla y mostrar sus
campos</h2>
<p>
Una base de datos almacena sus datos en
tablas.<br>
</p>
<a name="cargarregistros"></a>
<h2>Carga de registros a una tabla y su
recuperación</h2>
<p>
Usamos "insert into". Especificamos los
nombres de los campos entre paréntesis y
separados por comas y luego los valores
para cada campo, también
entre paréntesis y separados por
comas.<br>
</p>
</body>
</html>
```

13.- Listas ordenadas (<ol>)

Este elemento HTML es útil cuando debemos numerar una serie de objetos.

```
Listas ordenadas
<ol>
<li>Rodriguez Pablo</li>
<li>Gonzalez Raul</li>
<li>Lopez Hector</li>
</ol>
```

La marca `<ol>` y su correspondiente marca de cerrado es

En su interior cada uno de los ítems se los dispone con el elemento li, que también tiene la marca de comienzo `<li>` y la marca de fin de ítem `</li>`

Luego se encarga el navegador de numerar cada uno de los ítems contenidos en la lista, tengamos en cuenta que se numeran porque se trata de una lista ordenada.

`<ol>` viene de ordened list

`<li>` viene de list ítem

14.- Listas no ordenadas (<ul>)

Una lista no ordenada como su nombre lo indica no utiliza un número delante de cada ítems sino un pequeño símbolo gráfico. La forma de implementar este tipo de listas es idéntica a las listas ordenadas.

Listas No ordenadas

```
<ul>
<li>C</li>
<li>C++</li>
<li>Java</li>
<li>C#</li>
</ul>
```

`<ul>` viene de unordered list
`<li>` viene de list ítem

15.- Listas anidadas

El lenguaje HTML nos permite insertar una lista dentro de otra. Se pueden anidar listas de distinto tipo, por ejemplo podemos tener una lista no ordenada y uno de los ítem puede ser una lista ordenada.

Para el anidamiento de listas solo debemos tener cuidado en la correcta apertura y cerrado de las marcas

Listas Anidadas

```
<ol>
<li>Argentina
<ul>
<li>La Nación</li>
<li>Clarín</li>
```

```
</ul>
</li>
<li>España
<ul>
<li>El País Digital</li>
<li>ABC</li>
</ul>
</li>
</ol>
```

16.- Tabla (<table><tr><td>)

El objetivo fundamental de las tablas es mostrar una serie de datos en forma ordenada, organizado en filas y columnas.

Para la creación de una tabla intervienen una serie de elementos: `<table>` Es la marca de comienzo de la tabla. `<tr>` Es la marca de comienzo de una fila. Esta marca debe estar dentro del elemento `table`. `<td>` Es la marca de comienzo de una celda. Esta marca debe estar dentro del elemento `tr`. Todos los elementos requiere la marca de cierre.

Para recordar el nombre de estos elementos HTML:
`<tr>` viene de table row que significa fila de la tabla.
`<td>` viene de table data que significa dato de la tabla.

```
<html>
<head>
</head>
<body>
<table border="1">
<tr>
<td>China</td><td>1300 millones</td>
</tr>
<tr>
<td>India</td><td>1080 millones</td>
</tr>
<tr>
<td>Estados Unidos</td><td>295 millones</td>
</tr>
</table>
</body>
</html>
```

Lo primero que aparece es la apertura del elemento `table`, donde inicializamos la propiedad `border` con el valor 1, con esto el contorno de cada celda será visible (pruebe de asignarle el valor cero y otros valores superiores a 1)

17.- Tabla con encabezado (<th>)

La primera característica que le vamos a agregar a una tabla son las celdas de encabezado. Normalmente la primera fila de una tabla puede representar los títulos para cada columna. Para indicar que se trata de una celda de encabezado utilizamos el elemento `<th>` en lugar de `<td>`

Confeccionemos el mismo problema del concepto anterior pero agregando un título a cada columna de la tabla mediante celdas de encabezamiento:

```
<html>
<head>
</head>
<body>
<table border="1">
<tr>
<th>Paises</th><th>Cantidad de habitantes</th>
</tr>
<tr>
<td>China</td><td>1300 millones</td>
</tr>
<tr>
<td>India</td><td>1080 millones</td>
</tr>
<tr>
<td>Estados Unidos</td><td>295 millones</td>
</tr>
</table>
</body>
</html>
```

`<th>` viene de table header cell que significa celda de encabezado de tabla.

18.- Tabla y combinación de celdas

En algunas situaciones se necesita que una celda ocupe el lugar de dos o más celdas en forma horizonzal o vertical, para estos casos el elemento `td` o `th` dispone de dos propiedades llamadas `rowspan` y `colspan`. A estas propiedades se les asigna un valor entero a partir de 2. Si queremos que una celda ocupe tres columnas luego inicializamos la propiedad `rowspan` con el valor 3:

```
<td colspan="3">Facturación de los últimos tres meses</td>
```

Si por el contrario queremos que una celda se extienda a nivel de filas luego hacemos:

```
<td rowspan="3">Secciones</td>
```

Veamos un ejemplo empleando el concepto de combinación de celdas:

```
<html>
<head>
</head>
<body>
<table border="1">
<tr>
<th rowspan="4">Recursos</th><th colspan="4">Facturación de los últimos tres meses</th>
</tr>
<tr>
<td>Discos Duros</td><td>23000</td><td>27200</td><td>26000</td>
</tr>
<tr>
<td>CPU</td><td>73000</td><td>67300</td><td>51000</td>
```

```
</tr>
<tr>
<td>Monitores</td><td>53000</td><td>72000</td><td>88000</td>
</tr>
</table>
</body>
</html>
```

Como podemos observar la primer celda la expandimos hacia abajo 4 casilla y la segunda celda la expandimos hacia la derecha en 4 celdas. Cuando tenemos que disponer las celdas de la segunda fila debemos tener en cuenta que la primera está ocupada. Luego el código es:

```
<tr>
<td>Discos Duros</td><td>23000</td><td>27200</td><td>26000</td>
</tr>
```

19.- Contenido de la cabecera de la página (<title>)

Hasta ahora habíamos dispuesto la cabecera vacía, ya que casi toda la información que disponemos en ella no se visualiza en el navegador. La única excepción corresponde al elemento `title`.

El elemento `title` nos permite definir el título que aparecerá en la barra del navegador (es decir en la parte más alta de la ventana).Veamos una simple página que muestre un mensaje y contenga un hipervínculo a una segunda página que muestre otro título:

pagina1.html

```
<html>
<head>
<title>Título de la primer página</title>
</head>
<body>
<h1>Prueba del elemento title</h1>
<a href="pagina2.html">Ir a la segunda página</a>
</body>
</html>
```

20.- Contenido de la cabecera de la página (<meta>)

Un elemento que no se visualiza es el meta, que tiene por objetivo especificar información sobre el propio documento. Veamos las dos propiedades fundamentales de la marca meta y los valores más comunes con lo que podemos inicializarlos:

```
<meta name="nombre de la propiedad" content="valor de la propiedad">
```

`Name` almacena el nombre de la propiedad y `content` el valor de la propiedad.

No existe ninguna especificación de la W3C que defina los valores posibles para el atributo `name`, si bien existen algunos que son estándares de facto (`description`, `keywords`, `author` etc.)

Veamos las propiedades y valores más comunes

```
<meta name="keywords" content="html, programación, webmaster">
```

Los buscadores tienen en cuenta los metadatos. Si en la página inicializamos la propiedad `name` del elemento meta con el valor `keywords` luego buscará en la propiedad `content` las distintas palabras claves más representativas para dicha página. Esto es muy útil para posicionar nuestra página según el contenido que provee.

Veamos otras inicializaciones del elemento meta:

```
<meta name="description" content="El objetivo es presentar los conceptos básicos de HTML.
Es objetivo prioritario respetar los estándares del W3C">
```

En este caso especificamos una descripción de la página, pudiendo ser del sitio si se trata de la página principal.

```
<meta name="author" content="Diego Rodriguez">
<meta name="copyright" content="Interpolacion inc.">
```

21.- Comentarios dentro de una página <!-- -->

Un comentario es un texto que solo le interesa a la persona que desarrolló la página, el navegador ignora todo el contenido que se encuentra dentro de esta marca. Los comentarios son muy útiles para el desarrollador de la página. Uno deja anotaciones para facilitar el mantenimiento del sitio.

Además hay que tener en cuenta que puede ser otra persona la que desarrolle en otro momento el mantenimiento de las páginas que desarrollamos nosotros. Lo que para uno puede ser muy obvio a otro desarrollador no necesariamente

Otro uso muy habitual cuando estamos desarrollando la página si queremos deshabilitar una parte del código podemos encerrarla entre los caracteres de comentarios.

La sintaxis para definir un comentario es:

```
<!-- Aquí va el comentario -->
```

Es obligatorio luego del caracter de menor < disponer el signo de admiración y los dos guiones seguidos. Cerramos el comentario con dos guiones y el signo de mayor >

Un comentario puede abarcar varias líneas:

```
<!--
  comentarios - comentarios - comentarios
  comentarios - comentarios - comentarios
 -->
```

22.- Sintaxis para caracteres especiales

Posiblemente hasta ahora no se ha preguntado cómo disponer dentro de una página los caracteres: < y >. Veremos que no los podemos disponer directamente ya que el navegador no los entendería. La solución es utilizar otra sintaxis para dichos caracteres, veamos los más utilizados:

<	<
>	>
&	&
"	"
	//Espacio en blanco.
©	©
€	€

23.- Formulario - <form>

Un formulario permite que el visitante al sitio cargue datos y sean enviados al servidor. Es el medio ideal para registrar comentarios del visitante sobre el sitio, solicitar productos, sacar turnos etc.

De todos modos veremos que el lenguaje HTML solo tiene el objetivo de crear el formulario. El HTML no tiene la responsabilidad de registrar los datos en el servidor, esta actividad está delegada a un lenguaje que se ejecute en el servidor (PHP, ASP, ASP.Net, JSP etc.)

Veamos la sintaxis básica para crear un formulario donde ingresemos nuestro nombre. Para crear un formulario debemos utilizar el elemento `form`, que tiene marca de comienzo y fin. Dentro de la marca `form` veremos otros elementos para crear botones, editores de línea, cuadros de chequeo, radios de selección etc.

Confeccionaremos un formulario para el ingreso de nuestro nombre y un botón para el envío del dato ingresado al servidor:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
Ingrese su nombre:
<input type="text" name="nombre" size="20">
<br>
<input type="submit" value="enviar">
</form>
</body>
</html>
```

Veamos detenidamente la estructura de un formulario elemental, lo primero la apertura de la marca `form` donde debemos definir dos propiedades (`action` y `method`):

```
<form action="/registrardatos.php" method="post">
```

La propiedad `action` se inicializa con el nombre de la página que procesará los datos en el servidor. Todo los formularios y los que usted implementará como ejercicios deben llamar a esta página: "/registrardatos.php" más adelante cuando conozca un lenguaje de servidor podrá almacenarlos en una base de datos, consultar otros datos, modificar datos existentes etc.

La segunda propiedad que debemos inicializar es method. Esta propiedad puede almacenar únicamente dos valores (`post` o `get`). Normalmente un formulario se envía mediante post (los datos se envían con el cuerpo del formulario) En caso de utilizar get los datos se envían en la cabecera de la petición de la página, utilizando el método `get` estamos limitados en la cantidad de datos a enviar, no asi con el método `post`.

Ahora veamos el cuadro de texto donde se ingresa el nombre:

```
Ingrese su nombre: <input type="text" name="nombre" size="20">
```

El mensaje "Ingrese su nombre:" es un texto fijo.

El elemento input permite definir un cuadro de texto (editor de línea) si asignamos a la propiedad `type` el valor `"text"`.

Todo cuadro de texto debe inicializar la propiedad `name` con un nombre para el cuadro de texto. Este es un dato fundamental para poder recuperar el dato ingresado en el servidor (por ejemplo mediante PHP)

Por último inicializamos la propiedad `size` con el valor 20, esto significa que el cuadro de texto se dimensiona para permitir mostrar 20 caracteres (no se limita la cantidad de caracteres a ingresar por parte del visitante sino la cantidad de caracteres que se pueden visualizar)

Seguidamente: `<input type="submit" value="enviar">`

También mediante el elemento input definimos un botón para el envío de datos al servidor. Debemos inicializar la propiedad t`ype` con el valor `submit`, con esto ya tenemos un botón para el envío de datos. La propiedad `value` almacena la etiqueta que debe mostrar el botón. Finalmente cerramos el formulario: `</form>`

24.- Formulario - input type="text"/ input type="password"

En el concepto anterior vimos cómo implementar un formulario básico.

Veamos ahora con más detenimiento el elemento input. Este elemento hemos visto que nos permite definir cuadros de texto y botón para subir los datos al servidor. Ahora veremos que también podemos definir cuadros para el ingreso de una clave y botones para borrar el contenido de todos los controles del formulario.

Confeccionaremos un formulario que solicite el ingreso del nombre de un usuario y su clave:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
Ingrese su nombre:
<input type="text" name="nombre" size="20">
<br>
Ingrese su clave:
<input type="password" name="clave" size="12">
<br>
<input type="submit" value="enviar">
<input type="reset" value="borrar">
</form>
</body>
</html>
```

Veamos la sintaxis nueva para definir un cuadro de texto para el ingreso de una clave:

```
<input type="password" name="clave" size="12">
```

Utilizamos el mismo elemento input pero inicializamos la propiedad `type` con el valor `"password"`, con esto logramos que cuando el visitante ingrese la clave se visualicen asteriscos en lugar de los caracteres que tipeamos.

Luego para definir un botón que permita borrar todos los datos ingresados hasta el momento lo hacemos mediante:

```
<input type="reset" value="borrar">
```

Es decir inicializamos la propiedad `type` con el valor `"reset"`, con esto sabe el navegador que cuando dicho botón sea presionado debe borrar todos los controles de ingreso de datos de dicho formulario.

25.- Formulario – textarea

El elemento de tipo textarea nos permite el ingreso de varias líneas a diferencia del cuadro de texto (input/text). Es muy utilizado cuando queremos ingresar un comentario de una longitud de caracteres grande.

Confeccionemos un formulario para que un visitante pueda ingresar su nombre, su mail y un comentario del sitio, empleando para este último dato a ingresar un elemento de tipo textarea:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
Ingrese su nombre:
<input type="text" name="nombre" size="30"><br>
Ingrese su mail:
<input type="text" name="mail" size="50"><br>
Comentarios:<br>
<textarea name="comentarios" rows="5" cols="60"></textarea>
<br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

La sintaxis para definir un área de texto para el ingreso de múltiples líneas es:

```
<textarea name="comentarios" rows="5" cols="60"></textarea>
```

Además de tener la propiedad `name` similar a los otros elementos relacionados a formularios tiene dos propiedades llamadas `rows` y `cols`. Estas dos propiedades indican la cantidad de filas y columnas que visualiza el área de texto.

26.- Formulario - input type="checkbox"

El elemento `checkbox` es otro control que se puede insertar en un formulario. Un `checkbox` es una casilla de selección que puede tomar dos valores (seleccionado/no seleccionado)

Para ver su funcionamiento implementaremos un formulario que solicite el ingreso del nombre de una persona y 4 elementos de tipo `checkbox` para que seleccione los lenguajes de programación que conoce:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
Ingrese su nombre:
<input type="text" name="nombre" size="30"><br>
Seleccione los lenguajes que conoce:
<br>
<input type="checkbox" name="java">Java<br>
<input type="checkbox" name="cmasmas">C++<br>
<input type="checkbox" name="c">C<br>
<input type="checkbox" name="csharp">C#<br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

Veamos la sintaxis para definir controles de formulario de tipo `checkbox`:

```
<input type="checkbox" name="java">Java<br>
```

Como vemos también utilizamos el elemento HTML input, donde inicializamos la propiedad `type` con el valor `checkbox`. Un control `checkbox` no muestra texto, solo una casilla que el operador puede tildar o destildar.

27.- Formulario - input type="radio"

Cuando tenemos un conjunto de opciones pero solo una puede ser seleccionada debemos emplear controles visuales de tipo radio. Para definir dichos controles también utilizamos el elemento `input` inicializando la propiedad `type` con el valor "radio"

Veamos un ejemplo del empleo de este control gráfico, supongamos que necesitamos indicar el tipo de estudios que tenemos utilizando controles de tipo radio:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
Ingrese su nombre:
<input type="text" name="nombre" size="30"><br>
Seleccione el máximo nivel de estudios que tiene:
<br>
```

```
<input type="radio" name="estudios" value="1">Sin
estudios<br>
<input type="radio" name="estudios" value="2">Primario<br>
<input type="radio" name="estudios" value="3">Secundario<br>
<input type="radio" name="estudios" value="4">Universitario<br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

Veamos cómo se definen los controles de tipo radio:

```
<input type="radio" name="estudios" value="1">Sin estudios<br>
<input type="radio" name="estudios" value="2">Primario<br>
<input type="radio" name="estudios" value="3">Secundario<br>
<input type="radio" name="estudios" value="4">Universitario<br>
```

Como podemos observar todos tienen el mismo valor en la propiedad `name`, con esto se logra que cuando seleccionamos uno se deseleccione el actual.

28.- Formulario - select (cuadro de selección individual)

El elemento `select` es un cuadro de selección. Este elemento HTML nos permite seleccionar una opción entre un conjunto. Confeccionemos un formulario que solicite cargar el nombre de una persona y el país donde vive, este último elemento mediante un control de tipo `select` permitir seleccionar el país.

El archivo pagina1.html es:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
Ingrese su nombre:
<input type="text" name="nombre" size="30"><br>
Seleccione su pais:
<select name="pais">
<option value="1">Argentina</option>
<option value="2">España</option>
<option value="3">México</option>
</select>
<br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

Veamos la sintaxis para crear un cuadro de selección, primero abrimos la marca `select` la cual tiene definido la propiedad `name`: `<select name="pais">`

Luego sin cerrar el `select` definimos tantos elementos de tipo `option` como opciones tendrá el cuadro de selección:

```
<option value="1">Argentina</option>
<option value="2">España</option>
<option value="3">México</option>
```

El elemento `option` define el texto a mostrar y en la propiedad `value` indica el valor a enviar al servidor en caso de estar seleccionada dicha opción.

29.- Formulario - select (cuadro de selección múltiple)

Una variante del cuadro de selección que vimos en el concepto anterior es permitir que el visitante del sitio pueda seleccionar varias opciones. Supongamos que tenemos un cuadro de selección con una lista de colores y queremos que el visitante pueda elegir varios y no uno solo.

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
Seleccione uno o varios colores (Presione Ctrl para seleccionar varios colores)<br>
<select name="colores[]" size="4" multiple>
<option value="1">Rojo</option>
<option value="2">Verde</option>
<option value="3">Azul</option>
</select>
<br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

Podemos observar la sintaxis para la definición de un cuadro de selección múltiple:

```
<select name="colores[]" size="4" multiple>
```

Definimos una propiedad llamada `multiple` y no le asignamos valor, por otro lado al nombre definido en la propiedad `name` le agregamos al final los caracteres [] para que desde el servidor podamos identificar que el control retorna posiblemente muchos valores.

30.- Formulario - select (agrupamiento de opciones)

Hemos visto que podemos crear cuadros de selección individual o de selección múltiple. Ahora veamos que podemos agrupar las opciones que tiene el cuadro de selección, esto tiene sentido si el cuadro de selección tiene muchos ítems.

Se cuenta con un nuevo elemento llamado `optgroup` que agrupa un conjunto de elementos `option`. Veamos un ejemplo de agrupar una serie de opciones, agruparemos una serie de frutas y verduras:

```
Agrupar Opciones
Seleccione una fruta o verdura:
<select name="articulo">
```

```
<optgroup label="Frutas">
<option value="1">Naranjas</option>
<option value="2">Manzanas</option>
</optgroup>
<optgroup label="Verduras">
<option value="3">Lechuga</option>
<option value="4">Acelga</option>
</optgroup>
</select>
```

Como podemos observar para agrupar una serie de opciones dentro de un `select` debemos encerrarlas con el elemento `optgroup`:

```
<optgroup label="Frutas"> <option value="1">Naranjas</option> <option
value="2">Manzanas</option> <option value="3>Sandia</option> <option
value="4">Frutilla</option> <option value="5">Durazno</option> </optgroup>
```

La propiedad `label` del elemento `optgroup` aparece dentro del control `select` pero no se puede seleccionar, es un título.

31.- Formulario – button

El elemento `button` es un control visual que se puede utilizar para sustituir los controles:

```
<input type="submit" value="Enviar">
<input type="reset" value="Borrar">
```

La ventanja de este elemento es que podemos agregar imágenes dentro del botón. La sintaxis de este elemento es la siguiente:

```
<button type="submit"> Texto a mostrar dentro del botón. </button>
```

Todo lo que está contenido entre las marcas de comienzo y fin del elemento `button` aparece dentro del botón, como por ejemplo una imagen, un párrafo, enfatizado de una palabra etc.

La propiedad `type` se puede inicializar con alguno de estos tres valores: `"submit"`, `"reset"` y `"button"`. Los dos primeros cumplen las funciones que ya conocemos es decir envío de los datos al servidor y borrado del contenido de los controles. En cuanto al tercer valor posible de la propiedad `type` significará que deberemos codificar una función en JavaScript para procesar el evento.

Para ver el funcionamiento confeccionaremos un formulario que solicite el ingreso del nombre de una persona y dos elementos `button` para subir el dato al servidor o borrar el dato cargado:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/htmlya/registrardatos.php" method="post">
Ingrese su nombre:
```

```
<input type="text" name="nombre" size="20">
<br>
<button type="submit">Enviar<img src="/imagenes/enviar.gif"
alt="envio"></button>
<button type="reset">Borrar<img src="/imagenes/borrar.gif"
alt="borrado"></button>
</form>
</body>
</html>
```

Perfectamente podemos definir un texto y cargar una imagen dentro del botón:

```
<button type="submit">Enviar<img src="/imagenes/enviar.gif" alt="envio"></button>
```

32.- Formulario - input type="file"

El control de tipo file nos permite enviar un archivo al servidor. Nuevamente el HTML solo indica al navegador que debe enviar el archivo al servidor pero debe haber en el servidor un programa que lo almacene en una carpeta del servidor.

Veamos la sintaxis para disponer un control de tipo file:

```
<input type="file" name="archi">
```

Otra cosa muy importante a tener en cuenta cuando hacemos `upload` de archivos al servidor es inicializar la propiedad `enctype` del elemento `form`:

```
<form method="post" action="/registrardatos.php" enctype="multipart/form-data">
```

Con esto indicamos al navegador que el formulario almacena uno o más archivos que deben ser enviados al servidor.

Confeccionemos una página que solicite el ingreso de un nombre y la foto de la persona:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post" enctype="multipart/form-data">
Ingrese su nombre:
<input type="text" name="nombre" size="30"><br>
Seleccione la foto:
<input type="file" name="foto">
<br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

33.- Formulario - input type="hidden"

Un campo `hidden` se lo denomina campo oculto. Este tipo de control no visualiza nada dentro del formulario. Su utilidad se presenta cuando desde el servidor se genera una página dinámica por ejemplo mediante PHP y se almacena en un campo oculto un valor que se rescatará al subir el formulario al servidor.

Ejemplo:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
<input type="hidden" value="10:20" name="hora">
Ingrese su nombre:
<input type="text" name="nombre" size="30">
<br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

Como vemos la sintaxis para definir un campo oculto es: `<input type="hidden" value="10:20" name="hora">`

34.- Formulario - agrupamiento de controles

El HTML dispone de un elemento llamado `fieldset` que solo tiene el objetivo de recuadrar y agrupar un conjunto de controles de un formulario. Además para agregar un título a este recuadro debemos agregar otro elemento HTML llamado `legend`.

Confeccionemos un formulario que solicite los datos personales de una persona y los datos del lugar donde trabaja, cada grupo de datos los dispondremos en un `fieldset`:

```
Formulario: Agrupamiento de controles
<fieldset>
<legend>Datos personales</legend>
Apellido y nombre:
<input type="text" name="nombre" size="30"><br>
</fieldset>
<fieldset>
<legend>Datos Laborales</legend>
Nombre de la empresa:
<input type="text" name="nombreempresa" size="30"><br>
</fieldset>
```

35.- Formulario - controles con valores iniciales

Un control puede aparecer cargado con un valor por defecto. Veamos como inicializar con valores por defecto para cada uno de los controles que hemos visto.

Para inicializar un control de tipo `text` debemos dar un valor a la propiedad `value`:

```
<input type="text" value="aqui su nombre" name="nombre" size="20">
```

El control aparece cargado con la cadena "aqui su nombre".

Para inicializar un control de tipo textarea debemos indicar el dato entre el comienzo y el fin de la marca:

```
<textarea rows="10" cols="40" name="curriculum">Ingrese aqui su curriculum</textarea>
```

El control `textarea` se inicializa con la cadena "Ingrese aqui su curriculum"

Para inicializar un control de tipo `checkbox` debemos disponer la propiedad `checked` sin asignar valor:

```
<input type="checkbox" name="java" checked>Opcion 1<br>
```

Con esto logramos que el `checkbox` aparezca tildado apenas aparece el formulario.

Para inicializar un control de tipo radio debemos definir la propiedad `checked` sin valor, igual que un `checkbox`, con la salvedad que solo un control de tipo radio puede tener definida esta propiedad:

```
<input type="radio" name="estudios" value="1" checked>Opción 1<br>
```

Para inicializar un control de tipo `select` con selección individual debemos definir la propiedad `selected` de los elementos `option`:

```
<select name="pais">
<option value="1">Argentina</option>
<option value="2" selected>España</option>
<option value="3">México</option>
</select>
```

En este caso aparece seleccionado España, más allá que sea el segundo `option` en la lista.

Para inicializar un control de tipo `select` con selección múltiple debemos definir la propiedad `selected` de varios elementos `option`:

```
<select name="colores[]" size="4" multiple="multiple">
<option value="1" selected>Rojo</option>
<option value="2">Verde</option>
<option value="3" selected>Azul</option>
<option value="4">Amarillo</option>
</select>
```

En este ejemplo los ítems Rojo, Azul aparecen seleccionados desde un comienzo.

Confeccionaremos como ejemplo un formulario que solicite el ingreso del nombre de una persona. Luego que seleccione si es mayor de edad o no (por defecto inicializar en si), seguidamente el teléfono (cargar por defecto 453-) y por último en un `textarea` solicitar que ingrese comentarios.

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
Apellido y nombre:
<input type="text" name="nombre" size="30"><br>Es mayor de edad?:<br>
<input type="radio" name="radio1" checked value="si">Si<br>
<input type="radio" name="radio1" value="no">No<br>
Telefono:
<input type="text" value="453-" name="telefono" size="15"><br>
<textarea name="comentarios" rows="5" cols="40">Ingrese aqui sus
comentarios</textarea><br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

36.- Formulario - Inhabilitar controles

Todos los controles que hemos visto podemos hacer que aparezcan inhabilitados. Supongamos que disponemos 3 controles de tipo radio para indicar que sección del sitio deseamos ingresar. Nosotros queremos mostrar que tiene 3 secciones pero una no está disponible. Esto lo resolvemos deshabilitando un radio:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
Seleccione la sección donde desea ingresar:
<br>
<input type="radio" name="seccion" value="1" disabled>Niños<br>
<input type="radio" name="seccion" value="2">Adolescentes<br>
<input type="radio" name="seccion" value="3">Mayores<br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

Para deshabilitar el primer radio añadimos la propiedad `disabled` sin asignarle valor:

```
<input type="radio" name="seccion" value="1" disabled>Niños
```

Los siguientes elementos pueden inhabilitarse: `button`, `input`, `optgroup`, `option`, `select` y `textarea`.

Esta propiedad tiene mucha aplicación si se aplica JavaScript en la página. Mediante JavaScript podemos luego de haber sido cargado el documento modificar el estado de los controles, habilitando y deshabilitando de acuerdo a los datos que carga el visitante al sitio.

37.- Formulario - text/password/textarea y readonly

Si definimos la propiedad `readonly` a un control este será de solo lectura y no podremos modificar su contenido. Esta propiedad tiene uso cuando mediante un lenguaje de script (generalmente JavaScript) modificamos el control cambiándolo de estado ante ciertos eventos. La diferencia con la propiedad `disabled` es que con esta no toma foco el control y generalmente aparece con un color que indica que el control está deshabilitado.

Confeccionemos un formulario que aparezca el nombre de una empresa en un `text` y el texto de un contrato en un `textarea`, ambos de solo lectura.

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/htmlya/registrardatos.php" method="post">
Ingrese su nombre:
<input type="text" name="nombre" size="30" value="Interpolacion" readonly><br>
Contrato:<br>
<textarea name="comentarios" rows="5" cols="60" readonly>
Por la presente ..........................
</textarea>
<br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

38.- Formulario - Envío de datos mediante mail.

La solución para enviar datos mediante el programa cliente de mail que esté configurado en la computadora debemos inicializar la propiedad `action` de la siguiente forma:

```
<form action="mailto:pizzasya@htmlya.com" method="post" enctype="text/plain">
```

Además inicializamos la propiedad `enctype` con el valor `"text/plain"` con lo que le indicamos que se trata de un archivo de texto plano. Tengamos en cuenta que no podemos enviar archivos adjuntos.

Para probar esta funcionalidad confeccionaremos una página que permita hacer un reclamo de reparaciones y se envíen los datos a una dirección de correo. Se debe poder ingresar el nombre, dirección y un comentario del problema.

La página HTML es:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
```

```
<body>
<h2>Reclamos</h2>
<form action="mailto:reclamos@gmail.com" method="post"
enctype="text/plain">
Ingrese su nombre:
<input type="text" name="nombre" size="20">
<br>
Ingrese su dirección:
<input type="text" name="dirección" size="30">
<br>
Informe del problema:
<br>
<textarea rows="5" cols="40" name="problema"></textarea>
<br>
<input type="submit" value="enviar">
</form>
</body>
</html>
```

Debe llegar a la casilla de correos reclamos@gmail.com un mail con el contenido de los datos cargados en el formulario. El mail contiene el nombre del control y el contenido ingresado por el operador.

Si queremos que el correo llegue con un título debemos inicializar `subject`:

```
<form action="mailto:reclamos@gmail.com?subject=pedido de reparación" method="post"
enctype="text/plain">
```

Con esto logramos ubicar perfectamente todos los mail que llegan a nuestra casilla de correos reclamos@gmail.com

39.- Frames

Con los `frames` se pueden mostrar más de un archivo HTML en la misma ventana del navegador. Podemos hacer que los `frames` interactúen, por ejemplo al presionar un enlace en un `frame` podemos cargar una página en otro `frame`.

Solo se aconseja emplear `frames` cuando la situación lo amerita, hay que tener en cuenta que el uso de `frame` hace menos accesible el sitio y es mucho más difícil imprimir su contenido.

Veamos un ejemplo de implementar dos `frames`:

```
<html>
<head>
<title>prueba de frames</title>
</head>
<frameset cols="20%,80%">
<frame src="pagina2.html">
<frame src="pagina3.html">
<noframes>
<p>El navegador no soporta frames</p>
</noframes>
</frameset>
</html>
```

Esta página es la que define la ubicación de los `frames` dentro del navegador. La cabecera tiene una sintaxis similar a todas las páginas que hemos visto, pero no existe el `body`, en su lugar encontramos el elemento `frameset`.

En este ejemplo dividimos la ventana del navegador en dos secciones que mostrarán una página HTML cada una, mediante la propiedad `cols` indicamos cuanto ocupará cada ventana en porcentaje:

```
<frameset cols="20%,80%">
```

En el interior del elemento `frameset` definimos las dos páginas HTML que deben mostrarse mediante el elemento `frame`. Este tiene una propiedad llamada `src` (source que significa fuente) que la inicializamos con el nombre de la página a mostrar. Así definimos las dos páginas:

```
<frame src="pagina2.html">
<frame src="pagina3.html">
```

Otra elemento importante es el `noframes` donde indicamos un mensaje en el caso que el navegador no cuente con la capacidad de mostrar `frames` (podemos disponer enlaces a las páginas en forma individual)

```
<noframes>
<p>El navegador no soporta frames</p>
</noframes>
```

Las otras dos páginas son iguales a las que hemos venido haciendo.

40.- Frames - Actualización de un frame a partir del enlace de otro frame

Una actividad habitual con frames es disponer hipervínculos en uno de los frame y actualizar el contenido de otro frame.

Veamos con un ejemplo la sintaxis:

pagina1.html

```
<html>
<head>
<title>prueba de frames</title>
</head>
<frameset cols="20%,80%">
<frame src="pagina2.html">
<frame src="pagina3.html" name="ventanadinamica">
<noframes>
<p>El navegador no soporta frames</p>
</noframes>
</frameset>
</html>
```

Podemos observar que para el `frame` que queremos acceder posteriormente para modificar su contenido debemos inicializar la propiedad `name`:

```
<frame src="pagina3.html" name="ventanadinamica">
```

pagina2.html

```
<html>
<head>
<title>prueba de frames</title>
</head>
<body>
<h2>Enlaces.</h2>
<ul>
<li><a href="pagina3.html" target="ventanadinamica">Enlace
1</a></li>
<li><a href="pagina4.html" target="ventanadinamica">Enlace
2</a></li>
</ul>
</body>
</html>
```

Este archivo es el `frame` de la izquierda, que contiene los hipervínculos a dos páginas. Para indicar que `frame` debe mostrar las páginas de estos hipervínculos agregamos la propiedad target inicializándola con el valor del `name` definido para el `frame` (en nuestro caso es "ventanadinamica")

Tengamos en cuenta que el frame de la derecha comienza mostrando el archivo pagina3.html y luego según que hipervínculo se seleccione mostrará el archivo: pagina3.html o pagina4.html

Los contenidos de los dos archivos pagina3.html y pagina4.html no tienen nada nuevo:

pagina3.html
```
<html>
<head>
<title>prueba de frames</title>
</head>
<body>
<h1>Página A</h1>
<h2>Este es el contenido de página del
archivo:pagina3.html</h2>
</body>
</html>
```

pagina4.html
```
<html>
<head>
<title>prueba de frames</title>
</head>
<body>
<h1>Página B</h1>
<h2>Este es el contenido de página del
archivo:pagina4.html</h2>
</body>
</html>
```

41.- Frames - Asignación de medidas en píxeles

En los ejemplos anteriores definimos las medidas de los `frames` en porcentajes:

```
<frameset cols="20%,80%">
```

Cuando lo indicamos en porcentajes al redimensionar la ventana del navegador el tamaño de los frame se redimensiona en forma proporcional.

Hay situaciones donde queremos que un `frame` no se redimensione, por ejemplo que el `frame` de la izquierda donde ubicaríamos un menú de opciones siempre permanezca inalterable. Esto lo logramos indicando un valor absoluto para dicho `frame`.

Veamos un ejemplo donde definimos 3 `frames` dividiendo la ventana en tres columnas. Luego queremos que el `frame` de la izquierda y la derecha tengan medidas inalterables, para esto lo definimos de la siguiente forma:

```
<frameset cols="200,*,200">
```

Veamos que significa el asterisco para la segunda columna. Como sabemos una ventana puede redimensionarse y las tarjetas gráficas tienen distintas resoluciones en píxeles (640x480, 800x600, 1024x768 etc.), con esta sintaxis indicamos que la primer columna ocupe siempre 200 píxeles, lo mismo la tercer columna, pero la segunda tendrá un ancho de los píxeles que restan (es decir el ancho de ventana menos 400).

42.- Frames - Propiedades del elemento frame

Hasta ahora hemos utilizado y definido las propiedades para la marca de inicio del elemento `frame: src` y `name`.

Otras propiedades que pasaremos a ver, comprender y probar su funcionamiento son:

`noresize` Esta propiedad no requiere que se le asigne un valor. Si se encuentra presente el `frame` no podrá ser redimensionado con el mouse por el visitante del sitio.
Por ejemplo si disponemos un menú de enlaces en un `frame` ubicado a la izquierda es muy probable que definamos la propiedad `noresize` ya que poca utilidad tiene agrandar o contraer esta región de pantalla.

`frameborder` Esta propiedad puede tomar los valores 1 o 0. Por defecto un `frame` aparece con borde es decir esta propiedad por defecto tiene el valor 1. Si queremos que el borde no aparezca debemos inicializarla con 0.
Hay que tener en cuenta que por más que los bordes no existan si se puede redimensionar el `frame` con el mouse.

`scrolling` Los valores posibles de esta propiedad son: `"auto","yes","no"`. Por defecto está inicializada con el valor `"auto"`. El valor auto significa que el navegador decide si se debe mostrar la barra de `scroll`. La mostrará solo si algún contenido del `frame` no se ve. Si definimos el valor "yes" estamos indicando que siempre debe estar visible la barra de navegación y por último si asignamos el valor "no" estaremos indicando que nunca debe aparecer la barra de navegación para dicho `frame`.

Resolvamos el siguiente problema:

Confeccionar una ventana con dos `frame` verticales. No permitir redimensionarlos y no mostrar el borde de los `frames`. Hacer que el `frame` de la derecha siempre muestre la barra de desplazamiento.

pagina1.html

```
<html>
<head>
<title>prueba de frames</title>
</head>
<frameset cols="200,*">
<frame src="pagina2.html" frameborder="0" noresize>
<frame src="pagina3.html" frameborder="0" scrolling="yes"
noresize>
<noframes>
<p>El navegador no soporta frames</p>
</noframes>
</frameset>
</html>
```

43.- Frames - Anidamiento de frameset

El lenguaje HTML nos permite definir un `frameset` en la ubicación de un `frame`, esto se logra anidando `frameset`.

Vamos a crear una página que contenga dos columnas. La segunda columna la dividimos en dos filas:

Para resolver este formato de página tenemos que plantear los `frameset` de la siguiente manera:

pagina1.html

```
<html>
<head>
<title>prueba de frames</title>
</head>
<frameset cols="200,*">
  <frame src="pagina2.html" noresize>
  <frameset rows="70,*">
    <frame src="pagina3.html" noresize>
    <frame src="pagina4.html" noresize>
  </frameset>
<noframes>
<p>El navegador no soporta frames</p>
</noframes>
</frameset>
</html>
```

44.- Iframes

El HTML dispone de un elemento llamado `iframe` que permite disponer un `frame` con el flujo de la página, similar a disponer una imagen en la página.

Veamos un ejemplo como disponer este tipo de `frame` tan particular:

pagina1.html

```
<html>
<head>
<title>prueba de iframes</title>
</head>
<body>
<h1>Esto es una prueba de un iframe</h1>
<iframe src="pagina2.html" width="400" height="200">
No tiene disponible el navegador la capacidad de iframe
</iframe>
<h2>Esto ya está fuera del iframe</h2>
</body>
</html>
```

Como podemos ver cuando necesitamos agregar el `iframe` dentro de la página disponemos:

```
<iframe src="pagina2.html" width="400" height="200">
No tiene disponible el navegador la capacidad de iframe
</iframe>
```

Le indicamos el ancho y alto que debe tomar el `iframe`, la ubicación continúa el flujo de la página.

Algunas propiedades útiles aplicables a un `iframe`:

`src`: Archivo a mostrar dentro del `iframe`.
`width`: Ancho en píxeles.
`height`: Alto en píxeles.
`frameborder`: Podemos asignarle los valores 1 o 0. Si vale 0 el borde no se muestra.
`scrolling`: Los valores posibles de esta propiedad son: `"auto","yes","no"`. Por defecto está inicializada con el valor "auto". El valor auto significa que el navegador decide si se debe mostrar la barra de `scroll`. La mostrará solo si algún contenido del `iframe` no se ve. Si definimos el valor "yes" estamos indicando que siempre debe estar visible la barra de navegación y por último si asignamos el valor "no" estaremos indicando que nunca debe aparecer la barra de navegación para dicho `iframe`.
`name`: Nombre del `iframe` si queremos acceder desde otra página. Por ejemplo si queremos actualizar su contenido desde un enlace ubicado en otra página.

**45.- Agrupación de elementos: y **

El elemento `<div>` permite agrupar conjuntos de otros elementos en forma de bloques para separar secciones lógicas de la página web y en compañía del atributo `id` podemos identificar las secciones. Por ejemplo:

```
<!-- Aquí comienza el div -->


<div id="cabecera" style="border: 1px solid blue;">
cabecera
</div>

<div id="menu" style="float: left; border: 1px solid blue; width: 200px;">
<div>menu 1</div>
<div>menu 2</div>
<div>menu 3</div>
<div>menu 4</div>
</div>

<div id="contenido" style="border: 1px solid blue; margin-left: 225px;">
contenido1
contenido2<br />
<div>contenido3</div>
<div>contenido4</div>
</div>

<div id="menu-derecha" style="float: right;">
derecha
</div>

<div id="pie" style="border: 1px solid blue; clear: both">
pie
</div>
```

Como se puede observar en el ejemplo anterior, cada sección lógica de la página está identificada con un <`div`>. No exceda el uso del `<div>` para que pueda crear estilos más claros usando CSS.

El atributo `style` permite especificar el estilo que se le aplicará al . Sin embargo existe una mejor forma de incluir estilos y es empleando una hoja de estilos, un archivo con extensión .css. En este último caso utilizaríamos el atributo

`class` para crearle una clase específica que se aplicará a todos aquellos elementos que la incluyan. Repitamos el ejemplo anterior utilizando el atributo class:

```
<!-- Aquí comienza el div -->


<div class=”cabecera”>
cabecera
</div>

<div class="menu">
<div id=”menu1” class=”menu”>menu 1</div>
<div> id=”menu2” class=”menu”>menu 2</div>
<div id=”menu3” class=”menu”>menu 3</div>
<div id=”menu4” class=”menu”>menu 4</div>
</div>

<div class="contenido">
contenido1
contenido2<br />
<div id=”contenido3” class=”contenido”>contenido3</div>
<div id=”contenido4” class=”contenido”>contenido4</div>
</div>

<div class="derecha">
derecha
</div>

<div class="pie">
pie
</div>
```

Podemos notar ahora que hemos utilizado el atributo `id` para aplicar estilos locales y el atributo `class` para aplicar estilos globales. Recordamos que en este último ejemplo estamos utilizando un archivo .css que puede ser incluido de la siguiente manera:

```
<link rel="stylesheet" type="text/css" href="estilos.css">
```

Con esta línea estamos incluyendo el archivo “estilos.css” que contiene cada uno de los estilos aplicados al documento HTML. ¿Dónde se debe colocar esta línea? Esta línea debe incluirse dentro de los tags `<head>` y `</head>` de la siguiente manera:

```
<head>
<link rel="stylesheet" type="text/css" href="estilos.css">
</head>
```

En la línea anterior estamos indicando el tipo de archivo con el que estamos trabajando y luego la ruta del documento.

Por otra parte el elemento `<span>` trabaja de manera similar al elemento div, la diferencia entre ellos es que el `div` incluye un salto de párrafo mientras que `span` aplicará los atributos que estén definidos dentro de él. Por ejemplo:

```
<!-- Aquí comienza el div -->


<div class=”cabecera”>
cabecera
</div>

<div class="menu">
<span id=”contenido1” class=”contenido”>contenido1</span>
<span id=”contenido2” class=”contenido”>contenido2</span><br />
<div id=”contenido3” class=”contenido”>contenido3</div>
<div id=”contenido4” class=”contenido”>contenido4</div>

</div>

<div class="contenido">
contenido1
contenido2<br />
<div id=”contenido3” class=”contenido”>contenido3</div>
<div id=”contenido4” class=”contenido”>contenido4</div>
</div>

<div class="derecha">
derecha
</div>

<div class="pie">
pie
</div>
```

En el ejemplo anterior podemos observar que para el `<div>` de menú agregamos dos `<span>` que sólo definen el estilo de los contenidos 1 y 2 respectivamente. No utilizamos `<div>` porque queríamos descartar cualquier salto de párrafo entre dichos contenidos.

46.- Utilizar otros lenguajes dentro de HTML:

En el punto anterior ya vimos que se puede incluir un archivo .css dentro del documento HTML. Si quisiéramos agregar ya sean códigos JavaScript o extensión de JavaScript: JQuery, a nuestro documento debemos hacerlo de la siguiente forma dentro de los tags <head></head>:

```
<head>
<script type="text/javascript" src="widget/lib/jquery-1.7.1.min.js"></script>
<script type="text/javascript" src="widget/lib/jquery.ui.rcarousel.min.js"></script>
</head>
```

Al igual que con el archivo .css visto anteriormente, estas líneas señalan que estamos trabajando con un script en este caso de JQuery. La primera línea indica el core o núcleo de la extensión y la segunda es el plugin que estamos utilizando.

Ahora bien un ejemplo para sólo JavaScript:

```
<head>
<script type="text/javascript">
  alert("Un mensaje de prueba");
</script>
</head>
```

Esta es una manera de incluir códigos cortos y sencillos. La otra manera es tal y como incluimos el core para JQuery:

```
<script type="text/javascript" src="/js/codigo.js"></script>
```

Es indiferente qué forma se utilice para incluir código JavaScript pero por buena práctica de programación es recomendable separar los códigos extensos de otros lenguajes que serán incluidos en el documento HTML para un mejor manejo de los archivos.

Incluir código PHP:

El código PHP a diferencia de los anteriores lo podemos incluir en cualquier sección del documento HTML. Sólo necesitamos los tags de apertura y cierre de PHP para embeberlo en donde sea necesario. Veamos el siguiente ejemplo:

```
<html>
  <head>
    <title>titulo de la pagina</title>
  </head>
  <body>
    <h1> ejemplo Php</h1>
    <?php
      echo "hola PHP";
    ?>
  </body>
</html>
```

Una vez embebido el código PHP dentro del documento HTML debemos guardarlo con la extensión .php y ejecutarlo a través de un servidor web.

47.- HTML 5

HTML 5 es la próxima versión de HTML (Lenguaje de Marcado de Hipertexto) que es el lenguaje universal de la web. Este incluye una serie de nuevos elementos y atributos que son típicos en los sitios Web modernos. Algunos de ellos reemplazan semánticamente a los elementos `<div>` y `<span>`, por ejemplo `<nav>` (bloque de navegación web) y `<footer>`. Otros elementos proporcionan una funcionalidad nueva a través de una interfaz estandarizada, como los elementos o etiquetas `<audio>` y `<video>`. El objetivo de HTML es guardar contenido y es por eso que algunos elementos obsoletos de HTML 4.01 se han eliminado, incluyendo elementos para la presentación de contenido como `<font>` y `<center>`, cuyos efectos se consiguen mediante las Hojas de Estilo en Cascada o CSS.

Nuevos Elementos

- `section`*: Puede ser un capítulo, una sección de un capítulo o básicamente cualquier cosa que incluya su propio encabezamiento.*
- `header`*: La cabecera de una página. No confundir con el elemento head*

- `footer`*: El final de la página.*
- `nav`*: Una colección de links a otras páginas*
- `article`*: Una entrada independiente en un blog, revista, etc.*
- `aside`*: Es un bloque semántico que representa una nota, un consejo una explicación….*
- `figure`*: Se utilizará para representar una imagen*
- `dialog`*: Se utilizará para representar una conversación entre varias personas*
- `time`*: Se utilizará para marcar un momento temporal en una historia*
- `meter`*: Se utilizará para indicar ciertas medidas dependiendo de los atributos*
- `progress`*: Representará el estado de cierto proceso*
- `video`*: Archivo de video*
- `audio`*: Archivo de audio*
- `details`*: Más detalles*
- `datagrid`*: Una tabla, una recopilación de datos formateados*
- `menu`*: Menu de navegación*

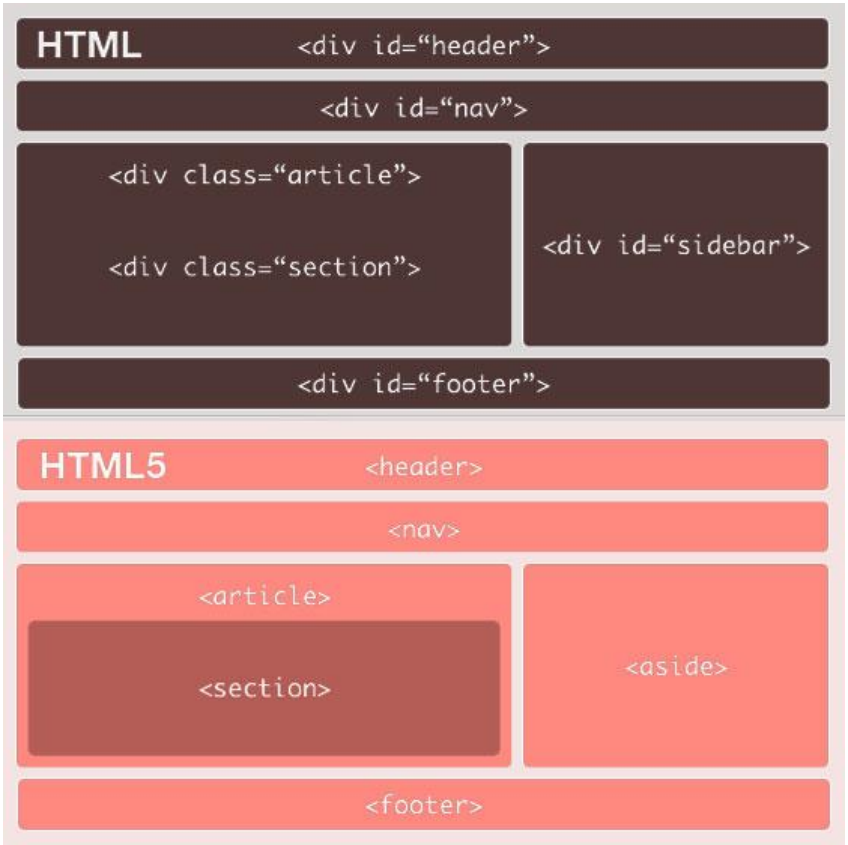
Nuevos Atributos

- `media`: Para conseguir una mayor consistencia con el link en los elementos `<a />`
- `ping`: Especificaremos una lista separada por espacios donde produciremos un ping cuando se siga el enlace, para los elementos `<area />` `y <a />`
- `target`: Disponible para mejorar la consistencia con el elemento `<a />`.
- `autofocus`: Destinado para indicar el elemento `<input />` (no `hidden`), `<select />`, `<textarea />` o `<button />` que ha de coger el foco al cargar la página.
- `form`: Atributo para `<input />`, `<ouput />`, `<select />` `<textarea />`, `<button />` y `<fieldset />` que permite que se asocien con un simple formulario.
- `replace`: atributo para <`input />; <button /> y <form />` que le afectará cuando el contenido del elemento sufra algún cambio.
- `data`: Para `<form />`, <select /> y `<datalist />`.
- `required`: Para elementos <input /> (Excepto hidden e image) y `<textarea />`, indica que el campo es obligatorio.
- `inputmode`: Atributo para `<input />` y `<textarea />`.
- `disabled`: Para `<fieldset />`, permite desactivar el `fieldset` completo.
- autocomplete, min, max, pattern, step: Para elementos <input /> permite delimitar las posibilidades de nuestros elementos de entrada.
- `list`: Para elementos `<datalist />` y `<select />`.
- `template`: Para `<input />` y `<button />` podrá usarse para repetir templates.

- `scoped`: Para elemento `<style />`, permitirá usar hojas de estilo “scoped” ??
- `async`: Para el elemento `<script />` el ajax hecho atributo.

Atributos Eliminados

- `rev` y `charset` en `<link />` y `<a />`
- `target` en `<link />`
- `nohref` en `<area />`
- `profile` en `<head />`
- `version` en `<html />`
- `name` en `<map />`
- `scheme` en `<meta />`
- `archive, classid, codetype, declare` y `standby` en `<object />`
- `valuetype` en `<param />`
- `charset` en `<script />`
- `summary` en `<table />`
- `header`, `axis` y `abbr` en `<td />` y `<th />`



Mejoras en los formularios

El elemento input adquiere gran relevancia al ampliarse los elementos que se permitirán en el "type".

- `<input type="search">` para cajas de búsqueda.
- `<input type="number">` para adicionar o restar números mediante botones.
- `<input type="range">` para seleccionar un valor entre dos valores predeterminados.
- `<input type="color">` seleccionar un color.
- `<input type="tel">` números telefónicos.
- `<input type="url">` direcciones web.
- `<input type="email">` direcciones de email.
- `<input type="date">` para seleccionar un día en un calendario.
- `<input type="month">` para meses.
- `<input type="week">` para semanas.
- `<input type="time">` para fechas.
- `<input type="datetime">` para una fecha exacta, absoluta y tiempo.
- `<input type="datetime-local">` para fechas locales y frecuencia.

BIBLIOGRAFIA

Autores del manual:

Hay que agradecer a diversas personas la dedicacion prestada para la creacion de este manual. Sus nombres junto con el numero de articulos redactados por cada uno son los siguientes:

Rubén Alvarez
Redactor de DesarrolloWeb.com
http://www.desarrolloweb.com
(23 capitulos)

Miguel Angel Alvarez
Director de DesarrolloWeb.com
http://www.desarrolloweb.com
(20 capitulos)

Christian Santalucía
http://www.criarweb.com
(2 capitulos)

Luciano Moreno
Consultor, disenador y desarrollador web en ParaRedeBJS. Especialista en usabillidad y diseno centrado en el usuario.
(6 capitulos)

Volver