Tutorial HTML Ya

1 - ¿Qué es HTML?

HTML es el lenguaje que se emplea para el desarrollo de páginas de internet.

Este lenguaje está constituido de elementos que el navegador interpreta y los despliega en la pantalla de acuerdo a su objetivo. Veremos que hay elementos para disponer imágenes sobre una página, hipervínculos que nos permiten dirigirnos a otra página, listas, tablas para tabular datos, etc.

Para poder crear una página HTML se requiere un simple editor de texto y un navegador de internet (IExplorer, FireFox, etc.).

Lo más importante es que en cada concepto desarrolle los ejercicios propuestos y modifique los que se presentan ya resueltos.

Este curso lo que busca es acercar el lenguaje HTML a una persona que nunca antes trabajó con el mismo. No pretende mostrar todos los elementos HTML en forma alfabética.

Como veremos, de cada concepto se presenta una parte teórica, en la que se da una explicación completa, luego se pasa a la sección del ejercicio resuelto donde podremos ver el contenido de la página HTML y cómo la visualiza el navegador. Por último y tal vez la sección más importante de este tutorial es donde se propone que usted haga páginas en forma autónoma (donde realmente podrá darse cuenta si el concepto quedó firme).

2 – Estructura interna de una página HTML.

Las instrucciones HTML están encerradas entre los caracteres: < y >.

Muchos elementos HTML requieren una marca de comienzo y otra de finalización. Todo aquello que está fuera de las marcas del lenguaje se imprime en la pantalla (dentro del navegador).

La estructura básica de una página HTML es:

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

Una página HTML es un archivo que generalmente tiene como extensión los caracteres html. Por ejemplo, llamaremos a nuestra primera página con el nombre: pagina1.html

Estos son los elementos básicos que toda página HTML debe llevar.

Si observamos toda página comienza con la marca: <html> y finaliza con la marca: </html>

Los fines de marcas tienen el mismo nombre que el comienzo de marca, más el carácter /

Una página HTML tiene dos secciones muy bien definidas que son la cabecera:

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

Y el cuerpo de la página:

```
<body>
  Cuerpo de la página.
</body>
```

Todo el texto que dispongamos dentro del <body> aparece dentro del navegador tal cual lo hayamos escrito.

Todas las páginas tienen como mínimo esta estructura: cabecera y cuerpo.

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).

Problema: Confeccione una página con las marcas mínimas que debe tener y en el cuerpo de la misma disponga su nombre y apellido.

3 – Salto de línea

Todo el texto que disponemos en el cuerpo de la página aparece en la misma línea, no importa si cuando tipeamos la página disponemos cada palabra en una línea distinta (es decir un navegador no tiene en cuenta la tecla ENTER).

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.

Implementemos una página que muestre los nombres de distintos lenguajes de programación uno por línea:

```
<html>
<head>
</head>
<body>
PHP<br>
JavaScript<br>
Java<br>
C<br>
C++
</body>
</html>
```

Como vemos solo hemos agregado la marca <br> cada vez que queremos comenzar una línea. Tengamos en cuenta que es indistinto si disponemos la marca en la misma línea o en la siguiente:

PHP

Es lo mismo:

PHP

Para recordar los nombres de los elementos HTML es bueno ver cual es la palabra completa de la misma:

**
** viene de **break**

Problema: Confeccionar una página HTML que muestre su nombre y apellido y en la siguiente línea los nombres de sus padres separados por un guión.

4 – Párrafo

Un párrafo es una oración o conjunto de oraciones referentes a un mismo tema. Todo lo que encerramos entre las marcas y 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 programación para trabajar con base de datos relaciones 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>
Este tutorial 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.

Para recordar el nombre de esta marca HTML:

<**p**> viene de **paragraph**

Problema: Confeccione una página que muestre en un párrafo datos referentes a sus estudios y en otro párrafo su nombre y mail.

5 – Títulos <h1><h2><h3><h4><h5><h6>

Otros elementos HTML muy utilizados son 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 tiene una fuente mayor (veremos que es el navegador el responsable de definir el tamaño de la fuente, más adelante podrá ver que uno puede modificar la fuente, tamaño, color, etc.)

Según la importancia del título utilizaremos alguno de estos elementos HTML. Requiere la marca de cerrado del título con la barra invertida como hemos visto.

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.

Recordemos que el HTML no tiene la responsabilidad de indicar el tamaño de las fuentes. El navegador definirá el tamaño de fuente según el nivel de título que indiquemos. La de tamaño más grande es la de nivel 1 <h1>.

<h1> viene de **heading**

heading significa título.

Problema: Confeccionar el titular de un periódico con un título de nivel 1. Luego definir dos títulos de segundo nivel con los textos (Noticias políticas y Noticias deportivas), en cada una de estas secciones definir dos titulares de tercer nivel con un párrafo cada una. Al final de la página mostrar un título de cuarto nivel con el nombre de la empresa propietaria del periódico.

6 – 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 ()

El elemento de mayor fuerza de énfasis es strong y le sigue em

Veamos un ejemplo del empleo de estos dos elementos HTML:

```
<html>
<head>
</head>
<body>
<p><strong>Tipos de datos</strong> en MySQL</p>
<p><em>TEXTO</em>: Para almacenar texto usamos cadenas de caracteres. Las cadenas se colocan entre comillas simples. Podemos almacenar dígitos con los que no se realizan operaciones matemáticas, por ejemplo, códigos de identificación, números de documentos, números telefónicos.
<p><em>NUMEROS</em>: Existe variedad de tipos numéricos para representar enteros, negativos, decimales. Para almacenar valores enteros, por ejemplo, en campos que hacen referencia a cantidades, precios, etc., usamos el tipo integer. Para almacenar valores con decimales utilizamos: float o decimal.</p>
<p><em>FECHAS Y HORAS</em>: para guardar fechas y horas dispone de varios tipos: date (fecha), datetime (fecha y hora), time (hora), year (año) y timestamp.</p>
</body>
</html>
```

Podemos ver que la sintaxis para el elemento strong es:

Tipos de datos

La mayoría de los navegadores muestran el texto enfatizado con strong con un texto en negrita y para em utilizan letra itálica (de todos modos esto no es obligatorio, pero seguramente mostrarán los textos enfatizados).

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.)

Para recordar el nombre de estos elementos HTML:

**** viene de **empathize** que significa énfasis.
**** viene de **strong** que significa fuerte.

7 – Hipervínculo a otra página del mismo sitio (<a>)

El elemento más importante que tiene una página de internet es el hipervínculo, estos nos permiten cargar otra página en el navegador. Esto es lo que hace diferente la página de un libro con la página de un sitio en internet. Normalmente un libro lo recorremos en forma secuencial, pero en un sitio de internet podemos disponer estos enlaces entre un conjunto de páginas y luego tener distintas alternativas de recorrido.

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.

Primero veremos cuál es la sintaxis para disponer un hipervínculo a una página que se encuentra en el mismo sitio (es decir otra página que hemos desarrollado nosotros).

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

Noticias

Como vemos, se trata de otro elemento HTML que tiene comienzo de marca y fin de marca. Lo que se encuentra entre el comienzo de marca y el fin de la marca es el texto que aparece en la página (normalmente subrayado).

Lo nuevo que aparece en este elemento es el concepto de una propiedad. Una propiedad se incorpora en el comienzo de una marca y tiene un nombre y un valor.

El valor de la propiedad debe ir entre comillas dobles.

La propiedad *href* del elemento "a" hace referencia a la página que debe mostrar el navegador si el visitante hace clic sobre el hipervínculo.

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

pagina1.html

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

pagina2.html

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

Como podemos observar lo nuevo en la pagina1.html es el hipervínculo a la pagina2.html:

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

Toda propiedad toma el valor que se encuentra seguidamente del carácter =

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>
```

Para recordar el nombre de esta marca HTML:

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

Problema: Confeccionar una página principal con dos hipervínculos a las páginas *pagina2.html* y *pagina1.html*. Luego en las dos páginas secundarias disponer hipervínculos a la página principal.

8 – 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.*

Resumiendo, a la propiedad *href* la inicializamos con el nombre del protocolo (http) seguida de dos puntos (:) y dos barras (//) luego la cadena (www.) y finalmente el nombre del dominio del sitio a enlazar.

La siguiente página muestra un hipervínculo al sitio principal del buscador Google:

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

Si analizamos un poco y pensamos que esta marca nos permite pedir una página a un servidor para que la cargue en el navegador: Qué página nos retorna del dominio www.google.com? La respuesta es que todo servidor cuando recibe una petición de una página sin indicar su nombre (es decir sólo está el nombre de dominio) selecciona y envía una página que tiene configurada el servidor como página por defecto (generalmente esa página es la principal del sitio y a partir de la cual podemos navegar mediante hipervínculos a otras páginas que se encuentran en dicho dominio).

Podemos enlazar a una página determinada de otro sitio. Veamos un ejemplo, si queremos disponer un enlace (hipervínculo) a la página about.html de google, la sintaxis será la siguiente:

```
<a href="http://www.google.com/intl/en/about.html">Acerca de Google</a>
```

Debemos conocer exactamente el nombre de la página (en este caso about.html) y también si la página se encuentra en el directorio raíz, debemos saber exactamente el camino de directorios (en este caso /int/en/).

Problema: Confeccionar una página que contenga un hipervínculo a un periódico (indicar sólo el nombre de dominio del periódico). Disponer además un segundo hipervínculo a una página determinada de ese periódico.

9 – 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 <br>).

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).

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.

Confeccionemos una página que muestre una imagen llamada foto1.jpg (La imagen se encuentra almacenada en el servidor en la misma carpeta donde se localiza esta página)

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

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

```
<img src="imágenes/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 subir dos carpetas, escribimos:

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

Por último, si queremos acceder a una imagen que se encuentra en una carpeta llamada imágenes pero que está al mismo nivel:

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

Primero le indicamos que subimos al directorio padre mediante los dos puntos .. y seguidamente indicamos el nombre de la carpeta y la imagen a mostrar.

<img> viene de **image**

src viende de **source**

alt viene de **alternative**

Problema: Desarrollar una página que muestre dos imágenes llamadas foto2.jpg y foto3.jpg, las mismas se encuentran en el servidor en la misma carpeta donde se almacenará la página que usted desarrollará. Disponer un título a cada imagen.

10 – 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>)

Confeccionemos una página que muestre dos imágenes (foto1.jpg y foto2.jpg) como hipervínculos. Al ser presionados llamar a otra página.

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="/imágenes/foto1.jpg" alt="Pintura
Geométrica"></a>
<br>
<a href="pagina2.html"><img src="/imágenes/foto2.jpg" alt="Pintura
Geométrica"></a>
</body>
</html>
```

Como podemos observar insertamos la marca HTML img en el lugar donde disponíamos el texto del hipervínculo. Eso es todo.

Lo que debe quedar bien en claro es que las imágenes se encuentran en un directorio llamado imágenes en la raíz del sitio (luego para indicar la referencia al archivo lo hacemos antecediendo la barra invertida / con lo que hacemos referencia a que partimos desde la raíz del sitio) en una carpeta llamada imágenes (/imágenes/foto1.jpg)

Es bueno practicar con esto ya que cuando implementemos sitios muy grandes seguramente agruparemos cada módulo en distintas carpetas.

Problema: Crear tres páginas con una foto cada una (foto1.jpg, foto2.jpg y foto3.jpg) luego al ser presionada avanzar a la siguiente página, es decir de la pagina1.html llamar a la pagina2.html, de la pagina2.html pasar a la pagina3.html y de ésta a la primera. Las imágenes se encuentran en una carpeta llamada imágenes que depende directamente de la raíz del sitio.

11 – Apertura de un hipervínculo en otra instancia del navegador.

El elemento “a” tiene una propiedad target que nos permite indicar que la referencia del recurso sea abierta en otra página.

Esta propiedad se llama target y debemos asignarle el valor “_blank” para indicar que el recurso sea abierto en otra ventana.

Confeccionemos una página que contenga dos hipervínculos, el primero abra el sitio en el mismo navegador y el segundo en otra instancia del navegador:

```
<html>
<head>
</head>
<body>
<h1>Apertura de enlaces en el mismo navegador y en otra instancia del navegador</h1>
<p>
<a href="http://www.lanacion.com.ar">Periódico La Nación</a>
<br>
<a href="http://www.clarin.com.ar" target="_blank">Periódico Clarín</a>
</p>
</body>
</html>
```

Podemos ver la diferencia entre el primer hipervínculo:

```
<a href="http://www.lanacion.com.ar">Periódico La Nación</a>
```

y el segundo hipervínculo que indica que el sitio sea abierto en otra ventana del navegador:

```
<a href="http://www.clarin.com.ar" target="_blank">Periódico Clarín</a>
```

12 – 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 nuestro ordenador.

Confeccionaremos una página que disponga un hipervínculo a un cliente de correo de mail:

```
<html>
<head>
</head>
<body>
<h1>Reclamos</h1>
<a href="mailto:gonzalop67@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:gonzalop67@gmail.com?subject=título del mensaje&body=cuerpo del mensaje">Enviar mail.</a>
```

Es decir luego de especificar el destinatario del mail disponemos un carácter 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:gonzalop67@gmail.com?subject=aquí el
título&cc=gonzalop68@gmail.com&bcc=gonzalop69@gmail.com&body=Este es el
cuerpo">Enviar mail.</a>
</body>
</html>
```

En este ejemplo enviamos un mail a gonzalop67@gmail.com, con copia a gonzalop68@gmail.com y con copia oculta a gonzalop69@gmail.com

13 – 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.

Un ancla se la define en una parte de la página que queremos que el operador llegue a partir de un hipervínculo.

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

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

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, Structured Query Language (Lenguaje de Consulta Estructurado) es un
lenguaje de programación 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.<br>
Ingresando instrucciones en la línea de comandos o embebidas en un lenguaje
como PHP nos comunicamos con el servidor. Cada sentencia debe acabar con
punto y coma (;).<br>
La sensibilidad a mayúsculas y minúsculas, es decir, si hace diferencia entre
ellas, depende del sistema operativo, Windows no es sensible, pero Linux si.
Por ejemplo Windows interpreta igualmente las siguientes sentencias:<br>
create database administración;<br>
Create Database administración;<br>
Pero Linux interpretará como un error la segunda.<br>
Se recomienda usar siempre minúsculas. Es más el sitio mysqlya.com.ar está
instalado sobre un servidor Linux por lo que en todos los ejercicios deberán
respetarse mayúsculas y minúsculas.
</p>
<a name="mostrarbasedatos"></a>
<h2>show databases</h2>
<p>
Una base de datos es un conjunto de tablas.<br>
Una base de datos tiene un nombre con el cual accederemos a ella.<br>
Vamos a trabajar en una base de datos ya creada en el sitio, llamada
"administración".<br>
Para que el servidor nos muestre las bases de datos existentes, se lo
solicitamos enviando la instrucción:<br>
show databases;<br>
Nos mostrará los nombres de las bases de datos, debe aparecer en este sitio
"administracion".<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>
Una tabla es una estructura de datos que organiza los datos en columnas y
filas; cada columna es un campo (o atributo) y cada fila, un registro. La
intersección de una columna con una fila, contiene un dato específico, un
solo valor.<br>
Cada campo (columna) debe tener un nombre. El nombre del campo hace referencia
a la información que almacenará.<br>
Cada campo (columna) también debe definir el tipo de dato que almacenará.<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>
Es importante ingresar los valores en el mismo orden en que se nombran los
campos, si ingresamos los datos en otro orden, no aparece un mensaje de error
y los datos se guardan de modo incorrecto.<br>
Note que los datos ingresados, como corresponden a campos de cadenas de
caracteres se colocan entre comillas simples. Las comillas simples son
OBLIGATORIAS.
</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>
<p>
```

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

14 – 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 carácter # y seguidamente el nombre del ancla.

Confeccionemos dos páginas y que la primera llame a diferentes anclas definidas en la segunda:

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>
<a name="introduccion"></a>
<h2>Introducción</h2>
<p>
SQL, Structured Query Language (Lenguaje de Consulta Estructurado) es un lenguaje de programación 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.<br>
Ingresando instrucciones en la línea de comandos o embebidas en un lenguaje
como PHP nos comunicamos con el servidor. Cada sentencia debe acabar con
punto y coma (;).<br>
La sensibilidad a mayúsculas y minúsculas, es decir, si hace diferencia entre
ellas, depende del sistema operativo, Windows no es sensible, pero Linux si.
Por ejemplo, Windows interpreta igualmente las siguientes sentencias:<br>
create database administración;<br>
Create Database administración;<br>
Pero Linux interpretará como un error la segunda.<br>
Se recomienda usar siempre minúsculas. Es más el sitio mysqlya.com.ar está
instalado sobre un servidor Linux por lo que en todos los ejercicios deberán
respetarse mayúsculas y minúsculas.
</p>
<a name="mostrarbasedatos"></a>
<h2>show databases</h2>
<p>
Una base de datos es un conjunto de tablas.<br>
Una base de datos tiene un nombre con el cual accederemos a ella.<br>
Vamos a trabajar en una base de datos ya creada en el sitio, llamada
"administración".<br>
Para que el servidor nos muestre las bases de datos existentes, se lo
solicitamos enviando la instrucción:<br>
show databases;<br>
Nos mostrará los nombres de las bases de datos, debe aparecer en este sitio
"administracion".<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>
Una tabla es una estructura de datos que organiza los datos en columnas y
filas; cada columna es un campo (o atributo) y cada fila, un registro. La
intersección de una columna con una fila, contiene un dato específico, un
solo valor.<br>
Cada campo (columna) debe tener un nombre. El nombre del campo hace referencia
a la información que almacenará.<br>
Cada campo (columna) también debe definir el tipo de dato que almacenará.<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>
Es importante ingresar los valores en el mismo orden en que se nombran los
campos, si ingresamos los datos en otro orden, no aparece un mensaje de error
y los datos se guardan de modo incorrecto.<br>
Note que los datos ingresados, como corresponden a campos de cadenas de
caracteres se colocan entre comillas simples. Las comillas simples son
OBLIGATORIAS.
</p>
</body>
</html>
```

15 – Lista ordenada (<ol>)

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

Veamos con un ejemplo una lista ordenada para conocer su sintaxis. Mostraremos el orden de llegada de tres corredores:

```
<html>
<head>
</head>
<body>
<ol>
<li>Rodriguez Pablo</li>
<li>Gonzalez Raul</li>
<li>Lopez Hector</li>
</ol>
</body>
</html>
```

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

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.

Para recordar el nombre de estos elementos HTML:

<ol> viene de **ordered list**
<li> viene de **list item**

Problema: Confeccione una lista ordenada con los tres países con mayor población del planeta. Disponer un título de segundo nivel y debajo de la lista la suma de habitantes de esos tres países enfatizado.

16 – Lista no ordenada (<ul>)

Una lista no ordenada como su nombre lo indica no utiliza un número delante de cada ítem sino un pequeño símbolo gráfico.

La forma de implementar este tipo de listas es idéntica a las listas ordenadas.

Veamos un ejemplo donde implementamos una lista no ordenada:

```
<html>
<head>
</head>
<body>
<h2>Lenguajes de programación.</h2>
<ul>
<li>C</li>
<li>C++</li>
<li>Java</li>
<li>C#</li>
</ul>
</body>
</html>
```

Para recordar los nombres de estas marcas HTML:

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

Problema: Confeccionar una lista no ordenada que contenga hipervínculos a distintos periódicos que usted conoce. Agregar tantos ítems como periódicos conoce.

17 – Lista de definiciones (<dl>)

Como su nombre lo indica se utiliza para asociar un término y la definición del mismo. El navegador se encarga de destacar y separa el término y su definición.

Crearemos una lista con la definición de varios lenguajes de programación:

```
<html>
<head>
</head>
<body>
<dl>
<dt>C++</dt>
<dd>Es un lenguaje de programación, diseñado a mediados de los años 1980, por Bjarne Stroustrup, como extensión del lenguaje de programación C.</dd>
<dt>Java</dt>
<dd>Es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los 90.</dd>
<dt>JavaScript</dt>
<dd>Es un lenguaje interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje C.</dd>
</dl>
</body>
</html>
```

Como podemos observar intervienen más marcas que en los otros dos tipos de listas. Las marcas que encierran a la lista son <dl> (Definition List) </dl>

Ahora debemos poner a pares estos dos elementos <dt> (Definition Term) y <dd> (Definition Description)

El navegador se encarga de hacer el sangrado del contenido del elemento dt

Para recordar los nombres de estas marcas HTML:

<dl> viene de **definition list**
<dt> viene de **definition term**
<dd> viene de **definition description**

18 – 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 ítems puede ser una lista ordenada.

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

Implementaremos una página que enumere una serie de países en una lista ordenada y luego en cada país dispondremos una lista de hipervínculos de periódicos de dicho país:

```
<html>
<head>
</head>
<body>
<ol>
<li>Argentina
<ul>
<li><a href="http://www.lanacion.com.ar">La Nación</a></li>
<li><a href="http://www.clarin.com.ar">Clarín</a></li>
<li><a href="http://www.pagina12.com.ar">Página 12</a></li>
</ul>
</li>
<li>España
<ul>
<li><a href="http://www.elpais.es">El País Digital</a></li>
<li><a href="http://www.abc.es">ABC</a></li>
<li><a href="http://www.elmundo.es">El Mundo</a></li>
</ul>
</li>
<li>México
<ul>
<li><a href="http://www.jornada.unam.mx">La Jornada</a></li>
<li><a href="http://www.el-universal.com.mx">El Universal</a></li>
</ul>
</li>
</ol>
</body>
</html>
```

Se puede insertar en un elemento li una lista como podemos ver:

```
<li>Argentina
<ul>
<li><a href="http://www.lanacion.com.ar">La Nación</a></li>
<li><a href="http://www.clarin.com.ar">Clarín</a></li>
<li><a href="http://www.pagina12.com.ar">Página 12</a></li>
</ul>
</li>
```

19 – Tablas (<table><tr><td>)

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

Algo importante es que no utilizaremos las tablas para organizar la página completa (esto se hace en la actualidad mediante CSS, es decir hojas de estilo).

Para la creación de una tabla intervienen una serie de elementos:

<table> Es la marca de comienzo de la tabla. Este elemento requiere la marca de cierre.

<tr> Es la marca de comienzo de una fila. Esta marca debe estar dentro del elemento table. Este elemento requiere la marca de cierre.

<td> Es la marca de comienzo de una celda. Esta marca debe estar dentro del elemento tr. Este elemento requiere la marca de cierre.

Para recordar el nombre de estos elementos HTML:

<table>

<tr> viene de **table row** que significa fila de la tabla.

<td> viene de **table data** que significa dato de la tabla.

La mejor forma de entender y dominar este concepto es implementar tablas y ver su resultado dentro del navegador. Confeccionemos una tabla que muestre los nombres de países en una columna y su cantidad de habitantes en otra:

```
<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)

```
<table border="1">
```

Seguidamente viene la apertura de la primera fila de la tabla con el elemento tr:

```
<tr>
```

Continuamos con la apertura de la primera celda de la primera fila de la tabla:

```
<td>
```

Luego el dato propiamente dicho de la celda:

```
India
```

Cerramos la celda:

```
</td>
```

Abrimos la próxima celda:

```
<td>
```

Disponemos el valor de la celda:

```
1300 millones
```

Cerramos la celda:

```
</tr>
```

El mecanismo de la siguiente fila es similar a la primera.

20 – Tablas 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>

El navegador representa distinto las celdas de datos y las celdas de encabezamiento.

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>
```

Si observamos el código de la página, para indicar que se trata de una celda de tipo encabezado utilizamos el elemento th:

```
<tr>
<th>Paises</th><th>Cantidad de habitantes</th>
</tr>
```

El elemento th debe estar contenido entre las marcas <tr> y </tr>, es decir pertenecen a una fila.

Para recordar el nombre de este nuevo elemento HTML:

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

21 – Tablas con título (<caption>)

Para disponer un título a una tabla debemos incorporar el elemento caption inmediatamente después que abrimos la marca table. El elemento caption requiere la marca de apertura y cierre.

Dispongamos un título a nuestra tabla con la población de distintos países:

```
<html>
<head>
</head>
<body>
<table border="1">
<caption>Población de los paises con mayor cantidad de habitants.</caption>
<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>
```

Como podemos observar la etiqueta caption está inmediatamente después que se abre la etiqueta de comienzo de la tabla:

```
<table border="1">
<caption>Población de los paises con mayor cantidad de habitants.</caption>
<tr>
```

Para recordar el nombre de este nuevo elemento HTML:

<caption> significa título.

22 – Tablas y combinación de celdas.

En algunas situaciones se necesita que una celda ocupe el lugar de dos o más celdas en forma horizontal 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 inicializamos la propiedad colspan 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, hacemos:

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

Veamos un ejemplo 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>
```

Veamos como definimos la primera fila de la tabla:

```
<tr>
<th rowspan="4">Recursos</th><th colspan="4">Facturación de los últimos tres
meses</th>
</tr>
```

Como podemos observar la primera celda la expandimos hacia abajo 4 casillas 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>
```

23 – 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).

Siempre debemos buscar un título lo más significativo en cuanto a lo que muestra la página.

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 primera página</title>
```

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

pagina2.html

```
<html>
<head>
<title>Título de la segunda página</title>
</head>
<body>
<h1>Prueba del elemento title (segunda página)</h1>
<a href="pagina1.html">Ir a la primera página</a>
</body>
</html>
```

24 – 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 los 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 de este tutorial 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="Gonzalo Penaherrera">

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

25 – 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 puede no ser tan obvio.

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 carácter de menor < disponer el signo de admiración y los dos guiones seguidos. Cerramos el comentario con dos guiones y el signo de mayor >

Confeccionaremos una página donde emplearemos comentarios:

```
<html>
<head>
</head>
<body>
<!—Corresponden a datos del año 2006. Modificar a principios de 2007 -->
<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>
```

Un comentario puede abarcar varias líneas:

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

De todos modos, hay que tener en cuenta que cuando un navegador pide la página a un sitio el servidor envía el archivo HTML completo, es decir con los comentarios. Los comentarios consumen ancho de banda del servidor.

26 – 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 los confundiría con los caracteres que se utilizan para las marcas HTML.

La solución es utilizar otra sintaxis para dichos caracteres, veamos los más utilizados:

```
<     &lt;
>     &gt;
&     &amp;
"     &quot;
             //Espacio en blanco.
©     &copy;
€     &euro;
```

Es decir, la sintaxis es disponer un ampersand seguido de un nombre significativo (por lo menos para los que entienden inglés) y finalmente un punto y coma.

Para ver su funcionamiento mostraremos la siguiente ecuación:

```
10+x*y < 12*z
```

pagina1.html

```
<html>
<head>
<title>Título de la primera página</title>
</head>
<body>
<h1>
10+x*y &lt; 12*z
</h1>
</body>
</html>
```

27 – Formulario - <form>

Un formulario permite que el visitante cargue datos al sitio 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 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.)

Como en este curso nos concentramos solamente en el lenguaje HTML no veremos como registrar los datos en el servidor. Si está impaciente puede visitar y ver como capturar los datos en el servidor mediante PHP (http://www.phpya.com.ar/)

Si recién comienza en el mundo del desarrollo de páginas web es recomendable primero aprender y conocer todos los elementos para la creación de formularios en HTML y en un paso posterior estudiar el registro en el servidor.

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 etiqueta 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 aperture de la marca forma 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. Como no conocemos la sintaxis de un lenguaje de servidor almacené en la raíz de este sitio una página codificada en PHP que recibe los datos del formulario y los imprime en una nueva página (recordar que en este curso no se ve como programar en el servidor)

Todos los formularios que implemento 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 así 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">
<br>
```

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 type 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>
```

28 – 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 definer un cuadro de texto para el ingreso de una clave:

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

Utilizamos el mismo element inputo 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.

Otra cosa que hay que tener en cuenta que la propiedad name de cada elemento input debe tener un nombre distinto (esto debido a que en el servidor se lo rescata a partir de este nombre)

29 – Formulario – textarea

El element 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>
```

Es un elemento que requiere una marca de comienzo y una de finalización.

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.

30 – 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="sumbit" value="Enviar">
</form>
</body>
</form>
```

Veamos la sintaxis para definer 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.

Si queremeos que aparezca un mensaje junto al checkbox solo lo agregamos seguido al elemento input.

Es importante hacer notar que los caracteres permitidos de la propiedad name son los caracteres alfabéticos y los números siempre y cuando no sea el primero.

31 – 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 controles de tipo radio 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>
Selecciones 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.

El valor que se rescata en el servidor es el dato almacenado en la propiedad value.

Si queremos disponer varios grupos de controles de tipo radio debemos definirles a cada grupo la propiedad name nombres distintos.

32 – 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. Veremos en el concepto próximo que según como la configuramos podemos seleccionar varias opciones.

Veamos con un ejemplo como crear un control de tipo select. 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>
<option value="4">Guatemala</option>
<option value="5">Honduras</option>
<option value="6">El Salvador</option>
<option value="7">Venezuela</option>
<option value="8">Colombia</option>
<option value="9">Cuba</option>
<option value="10">Bolivia</option>
<option value="11">Perú</option>
<option value="12">Ecuador</option>
<option value="13">Paraguay</option>
<option value="14">Uruguay</option>
<option value="15">Chile</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.

Luego de definir todas las opciones de nuestro cuadro de selección cerramos la marca select:

```
</select>
```

Una variante gráfica de este control es inicializar la propiedad size del elemento select con un valor distinto a uno, con esto creamos un cuadro de selección que muestra simultáneamente varios elementos (de todos modos, solo uno se puede elegir).

Es decir que con la propiedad size logramos un cambio estético del control.

33 – 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.

La página que resuelve este problema es:

```
<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>
<option value="4">Amarillo</option>
<option value="5">Blanco</option>
<option value="6">Negro</option>
<option value="7">Naranja</option>
<option value="8">Violeta</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.

Es también común inicializar la propiedad size con un valor mayor a 1 para que sea más fácil la selección.

La mayoría de los navegadores permiten seleccionar opciones que no se encuentran juntas mediante el mouse y presionando simultáneamente la tecla Ctrl.

34 – 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:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
Seleccione una fruta o verdura:
```

```
<select name="articulo">
<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>
<option value="6">Ciruela</option>
</optgroup>
<optgroup label="Verduras">
<option value="7">Lechuga</option>
<option value="8">Acelga</option>
<option value="9">Zapallo</option>
<option value="10">Papas</option>
<option value="11">Batatas</option>
<option value="13">Zanahorias</option>
<option value="14">Rabanitos</option>
<option value="15">Calabaza</option>
</optgroup>
</select>
<br>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

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>
<option value="6">Ciruela</option>
</optgroup>
```

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

La propiedad label del elemento optgroup es el texto que se debe mostrar dentro del select.

Se puede hacer agrupamiento de opciones y permitir selecciones múltiples.

35 – 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">
```

Entre otras las ventajas 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>
```

36 – Formulario – input type="button"

Otro tipo de botón que podemos crear es mediante el element input y fijando en la propiedad type el valor “button”.

Este tipo de botón no se lo puede hacer que actúe como los botones de tipo submit o reset, su actividad dependerá de un programa desarrollado generalmente en JavaScript. Si quiere puede introducirse luego en este lenguaje visitando el sitio http://www.javascriptya.com.ar/

Si bien no podemos ver su funcionamiento ya que no conocemos JavaScript si podemos implementar una página que muestre este control. Confeccionaremos una página que muestre el teclado de una calculadora:

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
<h1>Resultado:<h1>
<input type="button" name="boton7" value=" 7 ">
<input type="button" name="boton8" value=" 8 ">
<input type="button" name="boton9" value=" 9 ">
```

```
<input type="button" name="botondiv" value=" / ">
<br>
<input type="button" name="boton4" value=" 4 ">
<input type="button" name="boton5" value=" 5 ">
<input type="button" name="boton6" value=" 6 ">
<input type="button" name="botondiv" value=" * ">
<br>
<input type="button" name="boton1" value=" 1 ">
<input type="button" name="boton2" value=" 2 ">
<input type="button" name="boton3" value=" 3 ">
<input type="button" name="botonmas" value=" + ">
<br>
<input type="button" name="boton0" value=" 0 ">
<input type="button" name="botonigual" value=" = ">
<input type="button" name="botonmenos" value=" - ">
</form>
</body>
</html>
```

Como verá cuando se presiona el botón no sucede nada. Esto es así porque no hemos asignado ninguna actividad cuando sea presionado. Recordemos que HTML solo tiene Contenido, si queremos funcionalidad deberemos definir los eventos para dichos botones.

37 – 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">
```

Nuevamente utilizamos el element HTML input para definer este tipo de control. En la propiedad type inicializamos con el valor file. Inicializar la propiedad name también es importante ya que mediante este nombre se lo recupera en el servidor.

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.

Confeccionaremos 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>
```

Recordemos siempre inicializar la propiedad enctype del elemento form:

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

Tenga en cuenta que si no se hace ningún programa en el servidor el archivo no se almacena. Si quiere conocer como se hace esto en PHP puede visitar: http://www.phpya.com.ar/

38 – 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.

Su utilidad real solo podrá ser comprendida cuando estudie un lenguaje de servidor, pero veamos y conozcamos su sintaxis con un ejemplo.

Confeccionar un formulario que solicite ingresar el nombre de una persona y en un campo oculto almacene una hora cualquiera:

```
<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">
```

En el atributo value almacenamos el valor (este valor no se puede modificar desde el formulario)

Podemos imaginar una utilidad, supongamos que queremos que un visitante reenvíe un formulario cargado con todos los datos en un tiempo limitado, podríamos almacenar en el campo oculto la primera vez que solicita la página que contiene el formulario donde se registrará dicha hora. Luego al subir el formulario ya cargado al servidor controlaremos mediante un lenguaje de servidor si la hora actual y la hora de pedido del formulario no hacen invalidad los datos.

39 – 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.

Debemos encerrar todos los controles a agrupar entre las marcas <fieldset> y </fieldset>. Además para agregar un título a este recuadro debemos agregar otro elemento HTML llamado legend.

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

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
<fieldset>
<legend>Datos personales</legend>
Apellido y nombre:
<input type="text" name="nombre" size="30"><br>
Documento de identidad:
<input type="text" name="dni" size="8"><br>
Fecha de nacimiento:
<input type="text" name="fechanacimiento" size="12"><br>
Dirección:
<input type="text" name="direccionpersona" size="30"><br>
</fieldset>
<fieldset>
<legend>Datos Laborales</legend>
Nombre de la empresa:
<input type="text" name="nombreempresa" size="30"><br>
Actividad:
<input type="text" name="actividad" size="50"><br>
Dirección:
<input type="text" name="direccionempresa" size="30"><br>
</fieldset>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

Podemos ver que cada grupo de controles está encerrado con el elemento fieldset:

```
<fieldset>
<legend>Datos personales</legend>
Apellido y nombre:
<input type="text" name="nombre" size="30"><br>
Documento de identidad:
<input type="text" name="dni" size="8"><br>
Fecha de nacimiento:
<input type="text" name="fechanacimiento" size="12"><br>
Dirección:
<input type="text" name="direccionpersona" size="30"><br>
</fieldset>
```

Luego el título de cada fieldset lo disponemos con:

```
<legend>Datos personales</legend>
```

40 – 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 texto debemos dar un valor a la propiedad value:

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

El control aparece cargado con la cadena "aquí 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 aquí su curriculum</textarea>
```

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

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

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

Para inicializar un control de tipo select con selección individual debemos definer 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>
<option value="4">Guatemala</option>
<option value="5">Honduras</option>
<option value="6">El Salvador</option>
<option value="7">Venezuela</option>
<option value="8">Colombia</option>
<option value="9">Cuba</option>
<option value="10">Bolivia</option>
<option value="11">Perú</option>
<option value="12">Ecuador</option>
<option value="13">Paraguay</option>
<option value="14">Uruguay</option>
<option value="1">Chile</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>
<option value="5" selected>Blanco</option>
<option value="6">Negro</option>
<option value="7">Naranja</option>
<option value="1">Violeta</option>
```

```
 </select>
```

En este ejemplo los items Rojo, Azul y Blanco 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 sí), 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>
```

41 – Formulario – orden de foco de controles.

Todos los controles de formulario pueden definir una propiedad llamada tabindex que es un valor entero entre 0 y 32767. Este número indica el orden en que los controles toman foco. Cuando se presiona la tecla tab el navegador pasa el foco al siguiente control.

Para probar el funcionamiento implementaremos un formulario que contenga una matriz de tres filas y tres columnas de elementos de tipo text. Haremos que el foco sea por columna, es decir primero tomará foco el text de la primera fila y primera columna, luego del text de la segunda fila y primera columna, etc. (si no definimos la propiedad tabindex la carga de datos se efectúa por fila):

```
<html>
<head>
<title>Prueba de formulario</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
<input type="text" name="text1" size="5" tabindex="1">
<input type="text" name="text2" size="5" tabindex="4">
<input type="text" name="text3" size="5" tabindex="7">
<br>
<input type="text" name="text4" size="5" tabindex="2">
<input type="text" name="text5" size="5" tabindex="5">
<input type="text" name="text6" size="5" tabindex="8">
<br>
<input type="text" name="text7" size="5" tabindex="3">
<input type="text" name="text8" size="5" tabindex="6">
<input type="text" name="text9" size="5" tabindex="9">
```

```
<br>
<input type="submit" value="enviar" tabindex="10">
</form>
</body>
</html>
```

Como podemos observer los valores definidos para la propiedad tabindex para la primera fila de controles text es:

```
<input type="text" name="text1" size="5" tabindex="1">
<input type="text" name="text2" size="5" tabindex="4">
<input type="text" name="text3" size="5" tabindex="7">
```

Podemos observer que la propiedad tabindex no tiene valores consecutivos. Pero si vemos los text por columna podremos observar que si van en forma secuencial.

Todos los controles de formularios pueden definir la propiedad tabindex para indicar el orden de activación o foco del control.

42 – 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.

43 – Formulario – text/password y maxlength

Los controles de tipo text y password pueden limitar la cantidad de caracteres que puede ingresar el usuario a partir de la propiedad maxlength.

Debemos asignarle un valor entero que indica hasta cuantos caracteres está permitido ingresar.

No hay que confundir el objetivo de la propiedad size con maxlength. Con la propiedad size solo indicamos la cantidad máxima de caracteres a mostrar dentro del control antes de hacer scroll de los datos.

Confeccionaremos un formulario que solicite el nombre de usuario y su clave y solo permitiremos ingresar nombres de usuarios de hasta 20 caracteres y claves de hasta 12.

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

Cuando ejecute está página no podremos cargar un nombre de usuario de más de 20 caracteres, el teclado se inhabilita cuando se han ingresado 20 caracteres en el primer control.

44 – Formulario – text/password/textarea y readonly

Otra propiedad que podemos asignarle a los controles que creamos con el elemento input y también el elemento textarea es readonly.

Si definimos la propiedad readonly a un control, el mismo es de solo lectura y no podemos modificar su contenido. Esta propiedad tiene uso cuando mediante un lenguaje de script (generalmente javascript) modificamos el control cambiándolo de estado ante ciertos eventos.

Cuando un control tiene la propiedad readonly el control toma foco pero no se puede cambiar su contenido. 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 texto 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>
```

45 – Formulario – Envío de datos mediante mail.

Hasta ahora siempre configuramos la propiedad action de la marca <form> con el nombre de un archivo que procesa los datos en el servidor:

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

Para poder resolver esto debemos conocer el lenguaje PHP o algún otro lenguaje de servidor.

Conociendo solo HTML la solución es enviar los datos mediante el programa cliente de mail que esté configurado en la computadora. Para esto inicializamos la propiedad action de la siguiente forma:

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

Es decir, inicializamos la propiedad action con el texto mailto seguido de dos puntos y la dirección de mail a la que queremos enviar los datos del formulario, recordemos siempre que utilizamos mailto, el emisor del mail depende como esté configurado nuestro software de mail en nuestra computadora.

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 llegar a nuestra casilla de correos reclamos@gmail.com

La desventaja del envío de datos mediante mail es que la persona no puede hacer el envío del formulario desde una máquina ubicada en un ciberbar donde muy posiblemente no nos dejen configurar un cliente de mail.

46 – Formulario – label

Una última etiqueta relacionada con los formularios es la label.

Hasta este momento siempre que queríamos disponer un mensaje antes o después de un control de formulario lo escribíamos sin más.

Existe en HTML un elemento que permite asociar un texto con un control de formulario. Esto será muy útil si se accede desde un navegador no gráfico o una persona ciega que utiliza un programa que lee en voz alta el contenido de la página.

Veamos cómo lo hacíamos hasta ahora:

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

Utilizando el elemento label podemos hacer una referencia entre el texto y el control de entrada de datos:

```
<label for="nombre">Ingrese su nombre:</label>
<input type="text" name="nombre" size="20" id="nombre">
```

Veamos que hemos agregado:

- Hemos definido la propiedad id a la marca input.
- El elemento label tiene su marca de comienzo y fin, entre medio se dispone el texto a mostrar.
- Para vincular esta label con el elemento input debemos inicializar la propiedad for con el nombre asignado a la propiedad id del elemento input. Más adelante veremos que la propiedad

id la pueden tener todos los elementos HTML y es de vital importancia para CSS (Hojas de Estilo) y JavaScript.

Confeccionaremos un ejemplo completo:

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

Como podemos ver asociamos cada etiqueta con el correspondiente control de entrada de datos:

```
<label for="nombre">Ingrese su nombre:</label>
<input type="text" name="nombre" size="30" id="nombre"><br>

<label for="mail">Ingrese su mail:</label>
<input type="text" name="mail" size="50" id="mail"><br>

<label for="comentarios">Comentarios:</label><br>
<textarea name="comentarios" rows="5" cols="60"
id="comentarios"></textarea>
```

Normalmente las propiedades id y name de los controles de entrada de datos (input, textarea, etc.) se les asigna el mismo nombre, de todos modos no es obligatorio.

La propiedad for de la label hace referencia al id del control y no al name, esto es importante si inicializamos con valores distintos el id y name de los controles.

47 – 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 página HTML que deben mostrarse mediante el elemento frame.

El elemento frame 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">
```

Otro 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>
```

Finalmente cerramos el elemento frame y la página:

```
</frameset>
</html>
```

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

```
<html>
<head>
<title>prueba de frames</title>
</head>
<body>
<h2>Frame 1</h2>
</body>
</html>
```

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

```
<title>prueba de frames</title>
</head>
<body>
<h2>Frame 2</h2>
</body>
</html>
```

48 – 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 para actualizar un frame a partir del enlace de otro:

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>
```

49 – 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 (640x800, 800x600, 1024x768, etc.), con esta sintaxis indicamos que la primera columna ocupe siempre 200 píxeles, lo mismo la tercera columna, pero la segunda tendrá un ancho de los píxeles que restan (es decir el ancho de ventana menos 400).

Veamos la solución:

pagina1.html

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

pagina2.html

```
<html>
<head>
<title>prueba de frames</title>
</head>
<body>
<h1>Página A</h1>
</body>
</html>
```

pagina3.html

```
<html>
<head>
<title>prueba de frames</title>
</head>
<body>
<h1>Página B</h1>
</body>
</html>
```

pagina4.html

```
<html>
<head>
<title>prueba de frames</title>
</head>
<body>
<h1>Página C</h1>
</body>
</html>
```

Veamos algunas variantes para utilizar el asterisco:

- La primera columna es de 200 píxeles y los píxeles restantes se distribuyen entre el segundo y tercer frame.

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

- Con esta sintaxis los píxeles que restan luego de aplicar los 200 píxeles del primer frame se asignan 2/3 partes al frame central y 1/3 al frame de la derecha.

  <frameset cols="200,2*,*">

- Podemos mezclar las unidades de medida. Con esta sintaxis el 50% corresponde al frame central, 200 píxeles al frame de la izquierda y los píxeles restantes se asignan al frame de la derecha.

  <frameset cols="200,50%,*;>

50 – Frames – Propiedades del elemento frame

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

src
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 presenta 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 ó 0. Por defecto un frame aparece con bordes es decir esta propiedad por defecto tiene el valor 1. Si queremos que el borde no aparezca debemos inicializarla con 0.

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 "no" estaremos indicando que nunca debe aparecer la barra de navegación de dicho frame.

Resolvamos el siguiente problema:

Confeccionaremos 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>
```

pagina2.html

```
<html>
<head>
<title>prueba de frames</title>
</head>
<body>
<h1>Página A</h1>
<h1>Página A</h1>
<h1>Página A</h1>
<h1>Página A</h1>
<h1>Página A</h1>
<h1>Página A</h1>
</body>
</html>
```

pagina3.html

```
<html>
<head>
<title>prueba de frames</title>
</head>
<body>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
```

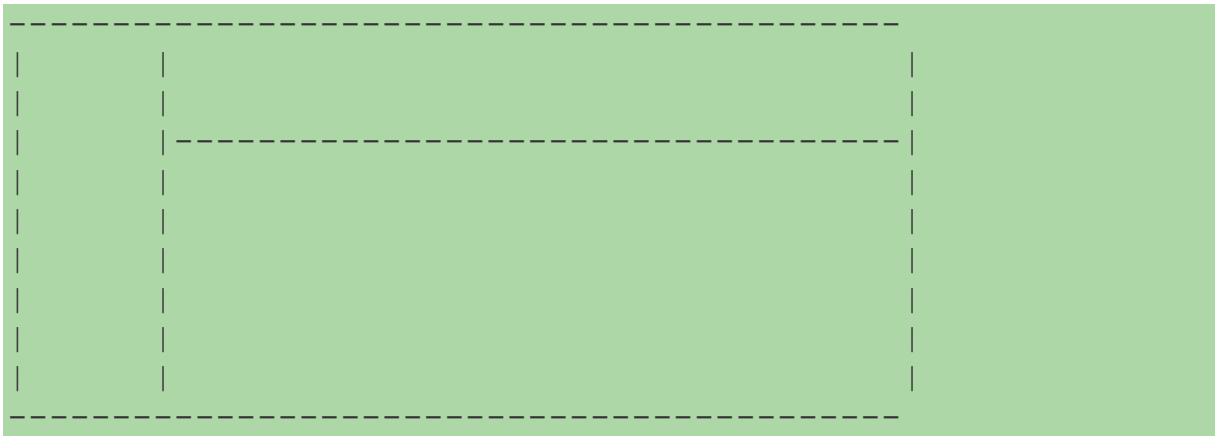
```
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
```

```
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
<h1>Página B</h1>
</body>
</html>
```

51 – 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>
```

Veamos más detenidamente como hemos creado los frameset, primero el frameset externo define en la propiedad cols dos valores

<frameset cols="200,*">

Con esto sabemos que estamos indicando que tendrá dos columnas, la primera de 200 píxeles y la segunda del resto de píxeles.

Luego debemos definir los frames de las dos columnas. Acá está la diferencia con los otros frames vistos:

```
<frame src="pagina2.html" noresize>
<frameset rows="70,*">
```

El primer frame lo ocupa el archive pagina2.html, pero el Segundo frame no existe, en su lugar se define otro frameset que inicializa la propiedad rows con los valores 70 y *. Con esto estamos indicando que la segunda columna se divide en dos filas, una de 70 píxeles y la otra del resto de píxeles de la ventana.

Las otras tres páginas no tienen nada nuevo:

pagina2.html

```
<html>
<head>
<title>prueba de frames</title>
</head>
<body>
<h1>Página A</h1>
</body>
</html>
```

pagina3.html

```
<html>
<head>
<title>prueba de frames</title>
</head>
<body>
<h1>Página B</h1>
</body>
</html>
```

pagina4.html

```
<html>
<head>
<title>prueba de frames</title>
</head>
<body>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
```

```
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
<h1>Página C</h1>
</body>
</html>
```

52 – 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.

La página que muestra el iframe no introduce ningún concepto nuevo:

```
<html>
<head>
<title>prueba de iframes</title>
</head>
<body>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
<h2>Este texto está dentro del iframe.</h2>
</body>
</html>
```

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 ubica en otra página.

53 – Declaración DOCTYPE.

Hasta ahora no hemos hablado de un concepto de vital importancia que es la utilización de elementos HTML estándares. De todos modos, no hemos introducido elementos HTML propietarios o desaprobados (ej. font, center, etc.).

La organización que define los estándares para la web es W3C.

Utilizar en lo posible las directivas de este comité de estándares nos traerá como ventaja que nuestras páginas en un futuro sigan viéndose correctamente en las nuevas versiones de navegadores.

La versión más actual de HTML es la 4.01.

Ahora introduciremos una nueva sección de nuestra página que es la declaración del tipo de documento (DTD Document Type Declaration), esta sección se ubica en la primera línea del archivo HTML, es decir antes de la marca html.

Según el rigor de HTML 4.01 utilizado podemos declararla como:

Declaración transitoria:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

Veamos la sintaxis para definir la sección del DOCTYPE, la primera diferencia con cualquier otro elemento HTML es que el carácter siguiente del < es el signo de admiración (!), luego debe ir la palabra DOCTYPE indicando que se trata de un archivo HTML. Más adelante debemos indicar la versión de HTML y si se trata de HTML transitorio o estricto:

Si es HTML transitorio disponemos:

"-//W3C//DTD HTML 4.01 Transitional//EN"

Si es HTML estricto:

"-//W3C//DTD HTML 4.01//EN"

Es decir que cuando es HTML estricto no disponemos la palabra Transitional

Por último, se define la dirección de internet donde se encuentra un archivo que enuncia los elementos y propiedades permitidos en el HTML 4.01, discriminando entre HTML estritco y transitorio.

¿Por qué utilizar HTML transitional, si lo más correcto sería utilizar HTML estricto?

Puede ser que tengamos muchas páginas desarrolladas en el pasado y nos lleve un tiempo hacerlas compatibles con el HTML estricto.

Para ver si una página cumple el estándar específico podemos acceder a un programa validador que se encuentra en validator.w3.org

Pruebe de copiar la siguiente página y verifique si se valida correctamente en la dirección indicada en la línea anterior.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Enlaces a periódicos</title>
</head>
<body>
```

```
<ol>
<li>Argentina
<ul>
<li><a href="http://www.lanacion.com.ar">La Nación</a></li>
<li><a href="http://www.clarin.com.ar">Clarín</a></li>
<li><a href="http://www.pagina12.com.ar">Página 12</a></li>
</ul>
</li>
<li>España
<ul>
<li><a href="http://www.elpais.es">El País Digital</a></li>
<li><a href="http://www.abc.es">ABC</a></li>
<li><a href="http://www.elmundo.es">El Mundo</a></li>
</ul>
</li>
<li>México
<ul>
<li><a href="http://www.jornada.unam.mx">La Jornada</a></li>
<li><a href="http://www.el-universal.com.mx">El Universal</a></li>
</ul>
</li>
</ol>
</body>
</html>
```

Pruebe de borrar el elemento title, tanto su contenido como las marcas de comienzo y final. Valide nuevamente.

Pruebe de borrar la marca <ol> y valide.

54 – Declaración DOCTYPE. HTML Transitional

Cuando indicamos que una página utiliza HTML Transitional podemos hacer uso de algunos elementos HTML de presentación (fuentes, alineamiento, colores), además la ubicación y anidamiento de elementos es más elástico.

Veamos si nuestra primera página que desarrollamos pasa la validación HTML Transitional (Agregándole la sección del DOCTYPE):

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
</head>
<body>
PHP - Java - JavaScript - C - C++
</body>
</html>
```

Como verás después que lo valides dará un error indicando que faltan datos en la marca head. Como mínimo deberemos agregar el elemento title.

Luego la página queda:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
```

```
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Mi primer pagina</title>
</head>
<body>
PHP - Java - JavaScript - C - C++
</body>
</html>
```

Ahora prácticamente valida correctamente, solo le falta un meta con la descripción de codificación:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Mi primer pagina</title>
</head>
<body>
PHP - Java - JavaScript - C - C++
</body>
</html>
```

Ahora si está totalmente correcta.

55 – Declaración DOCTYPE. HTML Estricto

En el DTD Estricto de HTML 4.01 se excluye los elementos y atributos de presentación que el W3C espera que dejen de utilizarse a medida que se extienda por parte de todos los navegadores el soporte de las hojas de estilo.

Los webmaster deben usar el DTD Estricto siempre que sea posible, pero pueden usar el DTD Transitional si es necesario el soporte de elementos y atributos de presentación.

Prácticamente todas las páginas que hemos desarrollado debemos hacerle algunos retoques para que cumplan con la validación de HTML estricta.

Veremos varias de las páginas desarrolladas y los cambios que debemos hacer para que pasen la validación:

Problema 1: Confeccionar una página que muestre los nombres de 5 lenguajes de programación separados por un guión:

```
<html>
<head>
</head>
<body>
PHP - Java - JavaScript - C - C++
</body>
</html>
```

Con los cambios para que valide:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Página de prueba del DTD</title>
</head>
<body>
<p>PHP - Java - JavaScript - C - C++</p>
</body>
</html>
```

¿Qué modificaciones hemos hecho?

1. Definimos el DOCTYPE
   <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "httm://www.w3.org/TR/html4/strict.dtd">
2. Agregamos el meta:
   <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
3. Definimos el element title:
   <title>Página de prueba del DTD</title>
4. Definimos el texto de la página dentro de un párrafo (no puede un texto depender directamente del body):
   PHP – Java – JavaScript – C – C++

Todos estos cambios son obligatorios para que valide como HTML 4.01 estricto.

Problema 2: Confeccionar una página principal que tenga un hipervínculo a otra página secundaria.

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

Con los cambios para que valide:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Página de prueba del DTD</title>
</head>
<body>
<h1>Página principal.</h1>
<p><a href="pagina2.html">Noticias</a></p>
</body>
</html>
```

¿Qué modificaciones hemos hecho?

1. Definimos el DOCTYPE
   <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/stric.dtd">
2. Agregamos el meta:
   <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
3. Definimos el elemento title:
   <title>Página de prueba del DTD</title>
4. Definimos el hipervínculo de la página dentro de un párrafo (no puede un hipervínculo depender directamente del body):
   Noticias

Problema 3: Confeccionar un formulario que solicite el ingreso del nombre de un usuario y su clave. Mostrar asteriscos donde se ingresa la clave. Disponer dos botones, uno para el envío de datos al servidor y otro para borrar el contenido de todos los controles (editores de texto) que contiene el formulario.

```
<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>
```

Con los cambios para que valide:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Página de prueba del DTD</title>
</head>
<body>
<form action="/registrardatos.php" method="post">
<p>
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">
</p>
```

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

1. Definimos el DOCTYPE
   <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
2. Agregamos el meta:
   <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
3. Definimos el elemento title:
   <title>Página de prueba del DTD</title>
4. Definimos todo el contenido del form dentro de un párrafo. No se puede disponer texto o controles de formulario que dependan directamente del elemento form:
    Ingrese su nombre: <input type="text" name="nombre" size="20">
 Ingrese su clave: <input type="password" name="clave" size="12">
 <input type="submit" value="enviar"> <input type="reset" value="borrar">

Todos estos cambios son obligatorios para que valide como HTML 4.01 estricto.

56 – Declaración DOCTYPE para Frames

Cuando empleamos frames debemos utilizar una declaración distinta para el elemento DOCTYPE.

Luego la sección del DOCTYPE para una página que implementa frames debe ser:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">

Es decir, se incorpora la palabra Frameset donde disponíamos la palabra Transitional o estaba vacía si validamos HTML estricto.

FUENTES BIBLIOGRAFICAS

https://www.tutorialesprogramacionya.com/htmlya/