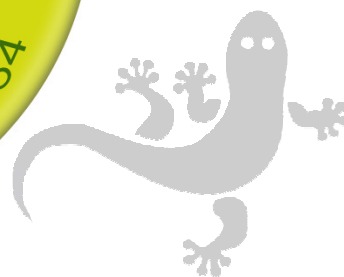


# I.E.S. DOMINGO PÉREZ MINIK



## **CICLO FORMATIVO DE GRADO SUPERIOR:** ***“ADMINISTRACIÓN DE SISTEMAS INFORMÁTICOS EN RED (L.O.E.)”***



### **MÓDULO:**

**Lenguajes de marcas y sistemas de  
gestión de información**

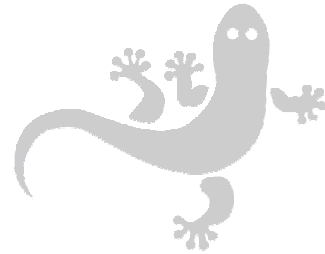
**(4 horas semanales)**





# Índice

<b>TEMA 2.- HTML.</b>	<b>3</b>
1.- HISTORIA DEL HTML.	3
2.- ESTRUCTURA BÁSICA DE LAS ETIQUETAS.	4
3.- ESTRUCTURA BÁSICA DEL DOCUMENTO.	5
4.- DOCTYPE: TIPO DEL DOCUMENTO.	6
5.- CONTENIDOS DE HEAD.	6
6.- ATRIBUTOS DE BODY.	7
7.- FORMATEO BÁSICO DEL TEXTO.	8
7.1.- Los encabezados.	8
7.2.- Modificadores básicos de letra.	8
7.3.- Control de párrafos, espacios y saltos.	9
7.4.- Control de la letra.	11
8.- LISTAS.	12
8.1.- Listas ordenadas.	12
8.2.- Listas no ordenadas.	12
8.3.- Listas de definición.	13
9.- ENLACES.	14
9.1.- Enlaces dentro de la misma página.	15
9.2.- Enlaces con otra página nuestra.	15
9.3.- Enlaces externos.	17
9.4.- Enlaces a otros protocolos.	17
10.- IMÁGENES.	18
11.- MAPAS DE IMÁGENES.	19
12.- TABLAS.	20
12.1.- Definiendo la tabla.	20
12.2.- Definiendo las filas.	21
12.3.- Definiendo las celdas.	21
12.4.- Combinación de celdas.	22
13.- MARCOS.	24
13.1.- Navegación entre marcos.	27
13.2.- Marcos complejos.	27
14.- FORMULARIOS.	28
14.1.- El control INPUT.	29
14.2.- INPUT de texto.	29
14.3.- INPUT de SUBMIT y RESET.	30
14.4.- INPUT de contraseña.	30
14.5.- INPUT ocultos.	30
14.6.- INPUT para activación de varias opciones simultáneas.	30
14.7.- INPUT para activación de opciones excluyentes.	31
14.8.- Áreas de texto.	32
14.9.- Listas de opciones.	32
14.10.- Botones genéricos y programación.	34
15.- SONIDOS Y OTROS OBJETOS.	36



## TEMA 2.- HTML.

---

### 1.- Historia del HTML.

HTML significa **HyperText Markup Language**, o lenguaje de marcas de hipertexto. Originalmente la palabra hipertexto fue pensada como un enlace dentro de un texto que te permite ir a otro documento.

El origen de HTML se remonta a 1980, cuando el físico Tim Berners-Lee, trabajador de CERN (Organización Europea para la Investigación Nuclear) propuso un nuevo sistema de "hipertexto" para compartir documentos.

La primera descripción de HTML disponible públicamente fue un documento llamado *HTML Tags* (Etiquetas HTML), publicado por primera vez en Internet por Tim Berners-Lee en 1991. Describe 22 elementos comprendiendo el diseño inicial y relativamente simple de HTML. Trece de estos elementos todavía existen en HTML 4.

La primera propuesta oficial para convertir HTML en un estándar se realizó en 1993 por parte del organismo IETF (Internet Engineering Task Force). Aunque se consiguieron avances significativos (en esta época se definieron las etiquetas para imágenes, tablas y formularios) ninguna de las dos propuestas de estándar, llamadas HTML y HTML+ consiguieron convertirse en estándar oficial.



El organismo IETF organiza un grupo de trabajo de HTML en 1995 y consigue publicar, el 22 de septiembre de ese mismo año, el estándar **HTML 2.0**. A pesar de su nombre, HTML 2.0 es el primer estándar oficial de HTML. A partir de 1996, los estándares de HTML los publica otro organismo de estandarización llamado W3C (World Wide Web Consortium). El 14 de enero de 1997 se publicó la versión **HTML 3.2** y es la primera recomendación de HTML publicada por el W3C. Esta revisión incorpora los últimos avances de las páginas web desarrolladas hasta 1996, como applets de Java y texto que fluye alrededor de las imágenes.

El **HTML 4.0** fue publicado el 24 de Abril de 1998 (siendo una versión corregida de la publicación original del 18 de Diciembre de 1997) y supuso un gran salto desde las versiones anteriores. Entre sus novedades más destacadas se encuentran la posibilidad de incluir pequeños programas o scripts en las páginas web, las hojas de estilos **CSS**, mejora de la accesibilidad de las páginas diseñadas, tablas complejas y mejoras en los formularios.

El **HTML 4.01** (la última especificación oficial de HTML) se publicó el 24 de diciembre de 1999. Se trata de una revisión y actualización de la versión HTML 4.0, por lo que no incluye novedades significativas. Desde la publicación de HTML 4.01, se detuvo la actividad de estandarización de HTML y el W3C se centró en el desarrollo del estándar XHTML. Por este motivo, las empresas Apple, Mozilla y Opera mostraron en el año 2004 su preocupación por la falta de interés del W3C en HTML y decidieron organizarse en una nueva asociación llamada WHATWG (Web Hypertext Application Technology Working Group). En la actualidad la actividad del WHATWG se centra en el **futuro estándar HTML 5**, cuyo primer borrador oficial se publicó el 22 de enero de 2008. Debido a la fuerza de las empresas que forman el grupo WHATWG y a la publicación de los borradores de HTML 5.0, en marzo de 2007 el W3C decidió retomar la actividad estandarizadora de HTML.

Paralelamente a su actividad con HTML, W3C ha continuado con la estandarización de **XHTML**, que es una versión avanzada de HTML y basada en XML. La primera versión de XHTML se denomina **XHTML 1.0** y fue publicada el 26 de Enero de 2000 (y posteriormente se revisó el 1 de Agosto de 2002). XHTML 1.0 es una adaptación de HTML 4.01 al lenguaje **XML**, así que mantiene casi todas sus etiquetas y características, pero añade algunas restricciones y elementos propios de XML. La versión **XHTML 1.1** ya ha sido publicada en forma de borrador y pretende modularizar XHTML. También ha sido publicado el borrador de XHTML 2.0, que supondrá un cambio muy importante respecto de las anteriores versiones de XHTML.

## **2.- Estructura básica de las etiquetas.**

HTML heredó la sintaxis de SGML. Las marcas o etiquetas siguen la siguiente estructura:

**<ETIQUETA atributos-opcionales> contenido </ETIQUETA>**



EJEMPLO: **<b> este texto está en negrita </b>**

Como ves las etiquetas se encierran en `<..>`. Cuando existe la etiqueta de cierre ésta se escribe con `</..>`. No todas las etiquetas llevan cierre.

Por otra parte muchas etiquetas pueden llevar atributos cuyo uso es opcional. Pongamos un ejemplo:

```
<p> esto es un párrafo </p>
```

```
<p align="right"> esto es un párrafo alineado a la izquierda </p>
```

Como ves los atributos suelen respetar la siguiente sintaxis:

```
<ETIQUETA atributo1="valor1" atributo2="valor2" ...>contenido </ETIQUETA>
```



### 3.- Estructura básica del documento.

Un documento HTML es un archivo de texto plano que viene definido por el siguiente conjunto mínimo de etiquetas obligatorias:

```
<HTML>
<HEAD>
  <TITLE>Este es el título de la página web </TITLE>
</HEAD>

<BODY>
  Éste texto aparece en el cuerpo de la página web
</BODY>
</HTML>
```



Como aspectos básicos debes conocer los siguientes:

- El lenguaje ignora los espacios en blanco y los saltos de línea. Podrías escribir todo el código anterior en una única línea. No obstante es importante acostumbrarnos a escribir de esta manera, respetando estas tabulaciones. Esto se conoce como **indentación** y es fundamental para asegurarnos la comprensión rápida y la detección de errores.
- El lenguaje NO es **CASE-SENSITIVE**. Esto significa que no distingue entre mayúsculas y minúsculas. Da igual pongas la etiqueta `<HEAD>` que `<head>` que `<Head>`. Más adelante veremos que XHTML suprime esto.

Bien, como ves la etiqueta `<HTML>` debe estar colocada al comienzo de la página. Es lo que avisa al navegador del lenguaje que estamos utilizando. Lo último del fichero será el cierre `</HTML>`.



**RECUERDA:** un documento HTML se divide en dos secciones:

Una única cabecera, marcada por la etiqueta `<HEAD>`.

Un único cuerpo, marcado por la etiqueta `<BODY>`.



El cuerpo es la parte visible de la página web, en él escribiremos nuestro texto marcado. La cabecera contiene el título y se usa también para cuestiones más avanzadas como los script (código en lenguajes de programación, del que el más popular es JavaScript) y definiciones de estilo (que suelen escribirse en CSS).

#### 4.- DOCTYPE: tipo del documento.

La etiqueta DOCTYPE debe colocarse como primera línea del documento, incluso antes de <HTML>. Es una etiqueta especial (y opcional) que sirve para definir el tipo de página web que estamos creando, indicando, por ejemplo la versión de HTML en que está escrito. Veamos la sintaxis:

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



En este caso es un documento HTML que usa una gramática “**strict**” (estricta). Esto significa que sólo contiene etiquetas aprobadas de forma definitiva por el W3C. Es el estándar que usaremos habitualmente y suele usarse cuando se emplea CSS para la presentación.

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



Éste será un documento HTML que usa una gramática **transicional**. Esto significa que puede contener etiquetas que están en proceso de ser aprobadas por el W3C o bien han sido desaprobadadas por el mismo organismo.

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



Ésta será la forma de indicar que este documento HTML usa **marcos** (frames).

Como ves DOCTYPE indica la versión de HTML y el idioma. EN = english.

#### 5.- Contenidos de HEAD.

Como ya hemos mencionado la etiqueta HEAD no representa la parte visible de un documento. Esta etiqueta debe obligatoriamente contener la etiqueta TITLE, pero además puede contener otras como META, SCRIPT, LINK. Por el momento sólo vamos analizar la etiqueta **META**.

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<meta name="generator" content="Lenguajes de marcas" />
<meta name="description" content="HTML, Pérez Minik, aprender " />
<meta name="keywords" content=" HTML, Pérez Minik, aprender " />
```



```
<meta http-equiv="author" content="IES D.Pérez Minik" />
<meta name="Content-language" content="es"/>
<meta name="city" content="La Laguna" />
<meta name="country" content="Spain" />
```

La etiqueta META sirve para definir metadatos, esto es, información acerca de esta página que estamos escribiendo, como el autor, el contenido, etc. Podemos incluir tantas etiquetas META como queramos y cada una de ellas definirá una propiedad de la página, por ejemplo:

```
<meta name="city" content="La Laguna" />
```

Define la propiedad “city” cuyo valor es “La Laguna”.

Ahora bien, ¿para qué sirve esta información?. De forma básica sólo para eso, para informar. No obstante existen una serie de propiedades como “keywords”, “robots”, “content”, que son utilizadas por los motores de búsqueda (Google, Terra, Lycos, ...) para mostrar las páginas cuando los usuarios de Internet hacen búsquedas por palabras.

## 6.- Atributos de BODY.

La etiqueta BODY puede contener atributos opcionales. Veamos algunos:

**<body bgcolor=”codigoRGB”>**

Configura el color de fondo de la página. Ahora es el momento de hablar de los colores en HTML: se especifican siguiendo un código RGB (Red-Green-Blue) escrito en Hexadecimal.

El color FF0000 tiene valor 255 para el rojo (FF=255), valor 00 para el verde y valor 00 para el azul → es el color rojo.

Cada color primario puede oscilar entre los valores 0 y 255 (un byte por color).

**<body text=”codigoRGB”>** Define el color del texto.

**<body link=”codigoRGB”>** Define el color por defecto de los enlaces (links).

**<body vlink=”codigoRGB”>** Define el color de los enlaces visitados (Visited links)

**<body alink=”codigoRGB”>** Define el color de los enlaces activos, esto es, en el momento de hacer click sobre ellos.

**<body vlink=”codigoRGB”>** Define el color de los enlaces visitados (Visited links)

**<body background=”url”>** Rellena el fondo con una imagen. Para decirle dónde está tendremos que especificar su ruta en el URL (Universal Resource Locator). Un URL es una dirección válida para encontrar un recurso, por ejemplo: <http://www.google.es>

NOTA 1: estos atributos pueden combinarse. Recuerda que sólo existe una etiqueta BODY en cada página.



```
<BODY text="00FF00" bgcolor="FFFFFF">
```

Además si solo vas a usar colores de la paleta básica, esto es, tonos puros como el rojo, marrón, etc puedes color directamente su nombre en inglés:

```
<BODY text="green" bgcolor="white">
```

NOTA: estos atributos fueron desaprobados en el estándar HTML 4.01. En su lugar se usa las hojas de estilo CSS para cuestiones relacionadas con los colores.

## 7.- Formateo básico del texto.

En este apartado empezamos a ver marcas o etiquetas para formatear el texto. Todas ellas deben colocarse dentro de BODY porque afectan a la parte visible del documento.

### 7.1.- Los encabezados.

Los encabezados (Header) son etiquetas predefinidas que presentan el texto en diferentes tamaños. Existen seis etiquetas de encabezados y cada una de ellas debe llevar su correspondiente cierre. Su nombres son <H1>, <H2>, <H3>, <H4>, <H5> y <H6>. El texto dentro de <H1> será más grande que el texto encerrado en un <H2> y así sucesivamente.




**PRACTICA 01.-** Realiza una página web llamada practica01.html que contenga una línea de texto definida en cada uno de los seis encabezados. Para hacer un salto de línea tendrás que utilizar la etiqueta <br>.



**PRACTICA 02.-** Copia el fichero anterior y llámalo practica02.html. Debe tener color de letra azul y fondo naranja.

### 7.2.- Modificadores básicos de letra.

El siguiente conjunto de etiquetas se utiliza para el formato de letra. Todas tienen que llevar su etiqueta de cierre correspondiente.



<EM>	Énfasis
<STRONG>	Énfasis mayor (parecido a negrita)
<CITE>	Texto que es una cita o referencia
<ABR>	Texto que es una abreviatura
<ACRONYM>	Texto que es un acrónimo
<BLOCKQUOTE>	Usado para citas largas (varias líneas). Se representan con una sangría.
<Q>	Igual que blockquote para citas cortas. Se representan encerrados entre alguna marca como comillas u otra.
<SUB>	Subíndice
<SUP>	Superíndice



<I>	Letra cursiva (I = italic)
<B>	Letra negrita (B= bold)
<U>	Subrayado (desaprobado en 4.01)



**PRACTICA 03.-** Haz una página con los primeros párrafos del Quijote. Coloca palabras en negrita, cursiva, subrayado. Coloca también el título de la obra como encabezado de nivel 1. Por último pon la primera palabra en negrita, cursiva y letra grande (para esto último usa la etiqueta BIG, y también existe la etiqueta SMALL para letra pequeña, pero casi no se utilizan).



**PRACTICA 04.-** Con la página de la práctica anterior agrega al final del título un 1 en superíndice. Al final de la página y en formato cita agrega el nombre del autor de la obra.

### 7.3.- Control de párrafos, espacios y saltos.

Ya habrás visto en las prácticas anteriores que HTML ignora los espacios en blanco y los saltos de línea. Para tratar de solucionar esto (que se soluciona mejor con CSS) se incluyeron un conjunto de etiquetas para un control básico del espacio:

Comencemos hablando de los **caracteres especiales** en HTML. Imagina que estás haciendo una página donde debe aparecer la siguiente frase:

El HTML encierra sus etiquetas entre los signos < y >.

Si colocas este texto directamente observarás que no aparece < y >. Esto es debido a que HTML al ver < cree que está empezando una etiqueta y por lo tanto interpreta que se ha escrito la etiqueta <y> que no existe. Cuando HTML encuentra una etiqueta mal formada o inexistente simplemente la ignora. ¿Cómo podemos escribir entonces un signo de menor que <? Pues usando los **caracteres especiales**.

Los **caracteres especiales** siempre empiezan por & y terminan con punto y coma. Veamos algunos:

&aacute;	á (forma de poner la a con tilde). Sirve para todas la vocales, &eacute; etc.
&lt;	Menor que, < (less than).
&gt;	Mayor que, > (greater than)
&ntilde;	La ñ.
&amp;	El ampersand &.
&copy;	El símbolo de copyright.
&#x0020;	Un espacio en blanco.
&#x0009;	Un tabulador





Utilizando los dos últimos podríamos incluir espacios en blanco y tabuladores en nuestro texto. No obstante esto sería muy tedioso. Veamos entonces la definición de párrafos, que usa la etiqueta `<P>`.

`<P align="alineación">` Esto es un párrafo `</P>`



La alineación puede ser “left” (izquierda), “right” (derecha) o “center” (centrado). Recuerda que el modificador Align es opcional.

En realidad la etiqueta P, muy utilizada, solamente incluye un salto de línea al final. Lo que nos sirve para separar el texto. No obstante existe una etiqueta también para hacer los saltos de línea:

`<BR>`



`<CENTER>`texto centrado `</CENTER>`



BR = break row (romper línea). Equivale a pulsar un Enter en el teclado. Es una etiqueta que NO lleva cierre. CENTER por su parte sirve para centrar un texto.

`<HR>`



HR = horizontal rule. Coloca una línea horizontal como separador. Opcionalmente podrás incluirle el modificador noshade.

`&nbsp;`



Es un carácter especial (No blank space), que indica que entre dos palabras no se debería incluir un salto de línea.

`<PRE>`

Esto es un párrafo



Preformateado.

`</PRE>`

Si con todas las etiquetas anteriores nuestro texto no respeta los espacios en blanco tal y como queremos podemos usar PRE (preformateo). Etiqueta para respetar exactamente el espaciado. Aunque pertenece al estándar 4.01 hay que ver el resultado en los navegadores.



**PRACTICA 05.-** Utiliza P para separar los párrafos del Quijote de la práctica anterior. Además antes de la cita con el autor inserta una línea horizontal.



**PRACTICA 06.-** ¿Has leído a Ángel González? Es un poeta sorprendente. Haz una página con su poema “Eso era amor”. Respetar el formato, agrega el título al comienzo. Al final de la página y justificado a la derecha coloca el nombre del autor.

Le comenté:

- Me entusiasman tus ojos.


Y ella dijo:  
 - ¿Te gustan solos o con rímel?  
 - Grandes,  
     respondí sin dudar.  
 Y también sin dudar  
 me los dejó en un plato y se fue a tientas.

## 7.4.- Control de la letra.

La tipografía utilizada se llama **fuente**. Existen multitud de fuentes como Arial, Times New Roman, Verdana, Courier, etc. Para controlar la fuente en HTML se utiliza la etiqueta FONT que permite con sus atributos controlar desde la tipografía hasta el color o tamaño.

Una cosa que debemos tener en cuenta siempre con las páginas web es que los usuarios las visualizarán en sus equipos y no podemos controlar las configuraciones particulares de esos equipos. ¿Qué pasaría si hago una página en la fuente “Bauhaus 93” y la ve un usuario que no tiene esa fuente instalada?. Pues bien, la recomendación inicial hasta que estudiemos CSS es utilizar sólo fuentes estándar.

También debo advertirte que la etiqueta FONT está desaprobadada en el estándar HTML 4.01, ya que la apariencia de la letra se control con CSS, pero por el momento vamos a utilizarla.



```
<FONT face="lista de fuentes separadas por coma"
      Size="de 1 a 7" color="código RGB">
  Texto afectado por la etiqueta </FONT>
```

NOTA: la lista de fuentes separadas por coma es para evitar el problema comentado anteriormente. Ejemplo:

```
<FONT face="Bauhaus 93, Arial, Verdana"> HOLA </FONT>
```

Se presentará la palabra HOLA en Bauhaus 93, pero si no se encuentra esa fuente instalada en el equipo se intentará mostrarla en Arial, y así sucesivamente. Siempre coloca Verdana al final porque es una fuente que conocen todos los navegadores.



**PRÁCTICA 07.-** Ahora un extracto de una obra de teatro. El autor es Santiago Serrano y la obra se llama Dinosaurios. Puedes encontrar el texto completo en [www.santiagoserrano.com](http://www.santiagoserrano.com). Veamos una pequeña parte del texto:

Nicolás: Basta por favor, sólo quería ayudarla.

Silvina: Yo me callo si no se acerca más.

Nicolás: Está bien, no voy a moverme. (CORTANTE) Pero no grite más.



Silvina: (LUEGO DE UN TENSO SILENCIO EN QUE HACE QUE LEE PERO LO MIRA DE REOJO) ¿Por qué está tan callado? Diga algo. ¿En qué piensa? Me asusta pensar en qué piensa.

Nicolás: Qué complicada es usted. No hay nada que le venga bien.

Silvina: Podría estar planeando algo.

Nicolás: Si la tranquiliza no planeo nada. No tengo ganas ni interés de planear nada. Si me callo es porque no tengo nada que decir, así que terminemos.

(DESPLIEGA EL DIARIO, LO ABRE, Y SE PONE A LEER CASI CUBIERTO).

Bien, crea una página indicando claro el autor y título de la obra. Los nombres de Nicolás y Silvina aparecerán en negrita. Los diálogos de Nicolás en azul y los de Silvina en rojo. Por otra parte las anotación a los actores aparecerán en letra más pequeña y gris.

## 8.- Listas.

En HTML tenemos tres tipos de listas. Las ordenadas (Ordered List), las no ordenadas (Unordered List) y las listas de definición (Definition List). Al igual que el resto de las etiquetas las listas pueden anidarse para formar listas más complejas.

### 8.1.- Listas ordenadas.

Utilizan OL (Ordered List) y LI (list item).

```
<OL type="tipo" start=numero>
  <LI> Elemento 1 </LI>
  <LI> Elemento 2 </LI>
  ...
  <LI> Elemento n </LI>
</OL>
```



Opciones para el atributo type:

"1" → ordena con números decimales.

"a" → ordena con letras minúsculas.

"A" → ordena con letras mayúsculas.

"i" → ordena con n° romanos en minúsculas.

"I" → ordena con n° romanos en mayúsculas.

Start indica el número o letra de comienzo. Si hacemos start="3" con una lista ordenada por letras comenzaremos en la C.

### 8.2.- Listas no ordenadas.

Utilizan UL (Unordered List) y LI (list item).

```
<UL type="tipo">
  <LI> Elemento 1 </LI>
  <LI> Elemento 2 </LI>
  ...
  <LI> Elemento n </LI>
</UL>
```



Opciones para el atributo type:


"circle" → utiliza círculos en los elementos.

"square" → utiliza cuadrados.

"disc" → utiliza círculos rellenos.

### 8.3.- Listas de definición.

Utilizan DL (Definition List), DT (Definition Term) y DD (Definition Data). Se usan para listas de glosario en la que se definen términos (como un diccionario). Veamos un ejemplo:



```
<DL>
  <DT>http
  <DD> Hipertext Transfer Protocol

  <DT>ftp
  <DD> FileTransfer Protocol
  ...
</DL>
```



**PRÁCTICA 08.-** Realiza una página web que muestre la siguiente lista:

#### LENGUAJES A ESTUDIAR

- i. HTML
- ii. CSS
- iii. XHTML
- iv. XML
- v. XSL
- vi. RSS



**PRÁCTICA 09.-** Realiza una página web que muestre la siguiente lista:

#### Protocolos de red:

- HTTP
- FTP
- TCP
- UDP
- IP
- ARP



**PRÁCTICA 10.-** Realiza una página web que muestre la siguiente lista (ojo con el color):

#### Acrónimos:

HTML  
HyperText Markup Language  
XML  
Extended Markup Language  
CSS  
Cascade Style Sheet



**PRÁCTICA 11.-** Realiza una página web que muestre la siguiente lista:

**Administración de Sistemas  
Informáticos en Red:**

- Primer curso
  1. Formación y orientación laboral
  2. Fundamentos de hardware
  3. Gestión de bases de datos
  4. Implantación de sistemas operativos
  5. Lenguajes de marcas
  6. Inglés
  7. Planificación y administración de redes
- Segundo curso
  1. Administración de SGBD
  2. Administración de SO
  3. Implantación de aplicaciones web
  4. Empresa e iniciativa emprendedora
  5. Formación en centros de trabajo
  6. Seguridad y alta disponibilidad
  7. Servicios de red e internet
  8. Inglés
  9. Proyecto

## 9.- Enlaces.

La base de Internet fue el llamado **Hipertexto**. Un hipertexto es un texto que nos permite alcanzar información de otros textos mediante **enlaces (LINKS)**. Los enlaces son por tanto la base de las páginas web, ya que son los que nos permiten navegar entre páginas. Básicamente un enlace es aquel punto donde podemos hacer click con el ratón para ir a ver otro documento.

Los enlaces también son llamados links, vínculos o anclas. Todos los enlaces utilizan la etiqueta <A>. Esa A viene del inglés “anchor” que significa “ancla”. Los enlaces **no se pueden anidar**.

Estructura general de la etiqueta <A>:

<A href=”URI”> Texto visible en la página </A>



URI: es la ruta o localizador del recurso web. Ejemplo:

<A href=<http://www.google.es>> IR A GOOGLE </A>

Con esto convertimos el texto IR A GOOGLE en un enlace.

Dependiendo del URI existen cuatro tipos de enlaces:

- 1.- Enlaces dentro de la misma página.
- 2.- Enlaces con otra página nuestra.
- 3.- Enlaces con una página externa.
- 4.- Enlaces a otros protocolos.



### 9.1.- Enlaces dentro de la misma página.

Si nuestra página tiene mucho contenido y es muy extensa nos puede interesar colocar saltos abreviados para, por ejemplo, ir rápidamente a determinadas secciones. Estos enlaces que nos llevan a un punto de la misma página se llaman también enlaces internos y tenemos que definir el enlace por una parte y el destino por otra. La manera de hacerlo es la siguiente:

Definiendo el enlace:

```
<A HREF="#destino"> Texto visible </A>
```

Definiendo el destino:

```
<A NAME="destino"></A>
```

NOTA: destino es un nombre cualquiera que nosotros colocaremos a nuestra elección, pero deben coincidir en las dos partes. Observa que definiendo el enlace antes de destino insertamos el carácter almohadilla #, que es el que indica que se trata de un enlace interno.



**PRÁCTICA 12.-** Para probar los enlaces internos debemos hacer una página con bastante contenido. Crea una página con la siguiente estructura:

Don Quijote de La Mancha:

[Ir al Capítulo 1.](#)

[Ir al Capítulo 2.](#)

#### **CAPÍTULO 1.**

Copia aquí los primeros párrafos del capítulo 1 del Quijote. Coloca bastante texto.

[Volver al índice.](#)

#### **CAPÍTULO 2.**

Copia aquí los primeros párrafos del capítulo 2 del Quijote. Coloca bastante texto.

[Volver al índice.](#)

Continuación del enunciado: observa que hemos definido cuatro enlaces internos, son los que he colocado subrayados y en color azul. Implementa los enlaces internos, recuerda que para cada uno definimos el enlace y el destino.

### 9.2.- Enlaces con otra página nuestra.

Habitualmente nosotros crearemos un **site**, es decir, un sitio web. Esto consisten en tener un conjunto de páginas entre las que navegaremos de una a otra mediante los enlaces que definimos. Imagina que estás haciendo tu site personal, en una página puedes escribir tu currículum, en otra tus hobbies, en otra tu galería de fotos, etc. Tendrás que permitir que los usuarios naveguen de una página a otra.



Para hacer un enlace a otra página tuya tendrás que colocar en href la URI o ruta local a la página referenciada.

Imagina la siguiente carpeta con tus páginas:

Carpeta MISITIO:

Carpeta: MISPAGINAS:

Index.html

Curriculo.html

Hobbies.html

Fotos.html

Carpeta: MIS FOTOS

Yo.gif

Mis\_amigos.gif

Mi\_perro.gif

Queremos hacer en index.html un enlace para visitar el curriculo y otro enlace para mostrar la foto de mi\_perro.gif. Podemos hacerlo de dos formas:

Con **ruta relativa**: significa especificar la ruta desde la carpeta actual donde está index.html.

<A href="Curriculo.html"> Ver mi currículo </A>

<A href=" ../MIS FOTOS/Mi\_perro.gif"> Ver mi perro </A>

Recuerda que .. significa “carpeta padre” y . significa “carpeta actual”.

Con **ruta absoluta**: significa especificar toda la ruta desde la unidad de disco actual (supongamos que es C:\):

<A href="C:\MISITIO\MISPAGINAS\Curriculo.html"> Ver mi curriculo </A>

<A href="C:\MISITIO\MISFOTOS\Mi\_perro.gif"> Ver mi perro </A>



**PRÁCTICA 13.-** Haz una nueva página web con enlaces a otras páginas de las prácticas anteriores. Mira una propuesta:

#### ESTUDIANDO HTML:

[Ver ejemplo de lista ordenada.](#)

[Ver ejemplo de lista no ordenada.](#)

[Ver ejemplo de lista de definición.](#)

Implementa los enlaces teniendo en cuenta la estructura de carpetas donde has colocado tus prácticas. Llama a esta página index.html y complétala con más enlaces para ayudarte a estudiar el temario.

NOTA: se pueden combinar los enlaces a otra página nuestra con los enlaces internos. Observa un ejemplo:

<A href="mipagina2.html#capitulo2"> Ver capítulo 2 </A>

Esto saltaría a mipagina2.html y dentro de ella buscaría el destino <A name="capitulo2"> y dejaría el cursor en ese punto.

### 9.3.- Enlaces externos.

Se usan cuando queremos poner un enlace a una página que no es nuestra. La única cosa que debemos hacer es colocar en href el URL (Universal Resource Locator) de la página externa, esto es, su dirección web. Mira el ejemplo:

```
<A href=http://www.hotmail.com> Ir a Hotmail </A>
```

La única cuestión a tener en cuenta es que estas páginas pueden estar alojadas en servidores Windows, Linux, Unix, etc. Con lo que es necesario, sobre todo en el caso de servidores Linux y Unix, que copiemos la dirección respetando mucho las mayúsculas y minúsculas.



**PRÁCTICA 14.-** Agrega a la página index.html de la práctica anterior unos enlaces interesantes donde hay tutoriales, códigos, etc que te ayudarán a resolver dudas de esta materia. Las páginas serán:

**WEBS INTERESANTES:**

[La web del programador.](#)

[Web Estilo.](#)

[Desarrollo Web.](#)

Búscalas en Google para agregar sus direcciones. Por cierto, son webs muy recomendadas.

### 9.4.- Enlaces a otros protocolos.

Las páginas web escritas en HTML se envían por la red desde el servidor hasta el ordenador del usuario usando el protocolo de red http (HiperTexto Transfer Protocol). Sin embargo podemos hacer enlaces a otros protocolos para descargar ficheros (FTP) o enviar emails (mailto). Veamos un ejemplo:

```
<A href=mailto:antoniolopez@hotmail.com>Contactar</A>
```

El funcionamiento de esto es muy relativo. Lo único que hace es abrir un cliente correo electrónico que debe estar disponible en el ordenador del usuario y colocar el destinatario, de forma que el usuario pueda rellenar el resto de campos. ¿Pero qué pasa si el usuario no tiene ningún cliente de correo como Outlook sino que usa su correo desde una web como Hotmail o Yahoo? Pues que no funcionaría.



**PRÁCTICA 15.1.-** Agrega a la práctica anterior un enlace para contactar contigo. Pruébalo. Comprueba que tienes un cliente de correo instalado.



**PRÁCTICA 15.2.-** Agrega a la práctica anterior un enlace para descargar el dossier del tema 1. Es un fichero .pdf y pretendes que al pulsar el enlace el cliente se descargue el fichero.



## 10.- Imágenes.

Para insertar imágenes usamos la etiqueta IMG, cuya sintaxis básica es la siguiente:

```
<IMG src="URI_fichero" alt="texto alternativo"> </IMG>
```



El atributo src debe contener la ruta relativa o absoluta hacia el fichero que contiene la imagen. Src es la abreviatura de source (fuente, origen). El fichero debe estar creado en uno de los formatos de imagen compatible con Internet: .png, .gif, .jpg, etc. La etiqueta de cierre </IMG> es opcional.

El atributo alt indica un texto descriptivo del contenido de la imagen. Este texto suele aparecer unos breves segundos al pasar el ratón sobre la imagen y se usaba cuando había navegadores que no soportaban multimedia como mensaje al usuario (como el antiguo navegador lynx que sólo soportaba texto).

Ejemplo: `<IMG src="mi_perro.gif" alt="foto de Toby"></IMG>`

Con las imágenes observarás rápidamente que el HTML es algo deficiente a la hora de colocar las imágenes en el sitio exacto que deseas. También puede ocurrir que el fichero contenga una imagen demasiado grande, con lo que la carga de la página se ralentiza y además puede verse deformada. Para esto, aparte de src y alt, existen una serie de atributos básicos:

```
<IMG src="..." alt="..." align="alineación horizontal" valign="alineación vertical">
```

**align** puede contener los valores: left (izquierda), center (centrado), right (derecha).

**valign** puede contener los valores: top (arriba), middle (centrado), bottom (abajo).

Es importante saber que los valores de align y valign no tienen porqué estar referido a la posición de la imagen dentro del documento. En realidad se refieren a la posición del documento padre, ya que IMG se interpreta como un “contenedor” que contiene un fichero y lo que se alinea es el fichero. Por eso suelen colocarse las imágenes dentro de tablas, que veremos en el siguiente apartado, para mostrarlas en el sitio exacto en que queramos. No obstante el uso de align puede observarse con la posición del texto. Observa el ejemplo:

```
<IMG SRC="imagen.gif" ALIGN=LEFT> Este texto está a la izquierda de la imagen.
```

```
<BR> Este también está a la izquierda, en la línea siguiente.
```

```
<BR CLEAR=LEFT> Este otro ya está debajo.
```

Es decir, que para romper la alineación del texto usamos `<br clear="left,center o right">`

Otro problema con las imágenes son sus dimensiones. Para resolver el problema del tamaño del fichero usaremos los atributos **width** (ancho) y **height** (alto). Tenemos varias maneras de indicar las dimensiones del contenedor, pero yo recomiendo dos:

1.- Con un número de píxeles concreto: `width="150px"`. La desventaja es que tenemos que contar con el hecho de que cada usuario visualiza la página en ventanas de tamaño diferente, lo que puede afectar al diseño.

2.- Con un porcentaje: width="20%". Tiene la ventaja de que la imagen cambia su tamaño si modificamos el tamaño de la ventana. Por supuesto el 20% está referido al elemento padre. Lo probaremos en prácticas para que observes la diferencia.

Por último podemos usar imágenes como enlaces. Basta encerrar la etiqueta <IMG> dentro de un vínculo. Observa el ejemplo:

```
<A href=www.todocoches.com> <IMG src="deportivo.gif"></A>.
```

Con esto hemos convertido toda la imagen en un vínculo. Para esto utiliza imágenes pequeñas y que el usuario sepa que son vínculos. Al hacer la imagen un enlace nos aparece muchas veces un molesto rectángulo azul (los enlaces suelen aparecer subrayados en azul). Para quitarlo agrega a la imagen el atributo border:

```
<A href=www.todocoches.com> <IMG src="deportivo.gif" border=0></A>.
```



**PRÁCTICA 16.-** Crea una página con fotos de animales que te gusten. Todas las fotos deben tener un ancho de 150 píxeles y junto a ellas habrá una descripción de la raza mostrada. Alinea las imágenes de forma diferente con respecto al texto.



**PRÁCTICA 17.-** Convierte alguna imagen de la práctica anterior en un enlace.

## 11.- Mapas de imágenes.

Un mapa de imágenes es una imagen donde hemos definido regiones y cada una de estas regiones es a su vez un hipervínculo. No es una característica muy utilizada de HTML por lo tedioso que sería definir las regiones (que pueden ser círculos, rectángulos o polígonos complejos). Observa el siguiente código de uso de la etiqueta <MAP>.

```
<IMG src="barranavl.gif" usemap="#mimapa">
<MAP name="mimapa">
<AREA href="laguna.html" shape="rect" coords="0,0,118,28">La laguna</a>
<AREA href="sc.html" shape="rect" coords="118,0,184,28">Santa Cruz</A>
<AREA href="teguste.html" shape="circle" coords="184,200,60">Teguste</A>
<AREA href="otros.html" shape="poly"
coords="276,0,276,28,100,200,50,50,276,0">Otros municipios</A>
</MAP>
```



Observa que primero colocamos una imagen y decimos que usa el mapa llamado "mimapa". Tras esto definimos la etiqueta <MAP> que encierra todas las áreas. Y por cada <AREA> definimos una región cuya forma (shape) puede ser un rectángulo (rect), un círculo (circle) o un poliedro irregular (Poly).

NOTA: a cada área podemos ponerle también el atributo ALT con un texto alternativo, de forma que cuando el usuario pase el ratón sobre la imagen le vaya apareciendo el texto de la zona seleccionada.



Para crear mapas de imágenes lo mejor es contar con un **editor de mapas gratuito**. Uno muy sencillo de usar es **MapEdit**. También puedes utilizar Image mapper, bajado de softonic.



**PRÁCTICA 18.-** Creemos un mapa de imágenes sencillo con una imagen de los municipios de Tenerife. Usando MapEdit genera áreas sensibles en los municipios de La Laguna, Santa Cruz y Tegueste que nos lleven a páginas con información de la población de éstos. Los datos de población los puedes conseguir en el Instituto de Estadística de Canarias: [www.gobiernodecanarias.org/istac/](http://www.gobiernodecanarias.org/istac/)  
**Atiende a la realización del primer municipio por el profesor.**

## 12.- Tablas.

En las prácticas anteriores habrás observado lo complicado que puede resultar colocar los elementos de texto o imagen en el lugar deseado. Durante muchos años los creadores de páginas webs han utilizado tablas sin border para posicionar elementos en pantalla. Es por eso que es fundamental saber crear tablas complejas en HTML. Y también es muy habitual encontrarnos con páginas web formadas por multitud de tablas anidadas. Es decir, las tablas, además de presentar información, nos sirven para realizar **maquetación del texto**. Expliquemos, pues, las etiquetas básicas usadas en las tablas.

### 12.1.- Definiendo la tabla.

Toda tabla comienza y termina con la etiqueta <TABLE>. Esta etiqueta tiene muchos atributos opcionales, que definiremos sólo cuando los necesitemos. Veamos los más utilizados:

```
<TABLE cols="n" rows="n" border="n" align="alineación"
        cellpadding="n" cellspacing="n" width="n">
```

.... Código de la tabla ...

```
</TABLE>
```



Los atributos opcionales *cols* y *rows* definen respectivamente el número máximo de columnas y filas. Son opcionales, no es necesario ponerlos puesto que se autocalculan con el contenido de la tabla. No obstante veremos que ayudan al navegador en tablas complejas que tengan celdas combinadas. Su valor será un número entero.

El atributo *border* especifica el ancho del borde. Para eliminar el borde coloca este atributo a cero. En HTML 4.00 también se definió *bordercolor* como atributo, pero estas cuestiones son todas suprimidas con el uso de CSS.

*Align* y *width* ya han sido explicados con anterioridad. Recuerda que *align* puede ser "center", "left" o "right" y que *width* puede llevar un número de píxeles o un porcentaje (lo que variará el tamaño de la tabla para ajustarlo a la ventana actual). La alineación en este punto afectará a todas las celdas de la tabla.



El atributo **cellpadding** es exclusivo de <TABLE>. Lleva asociado un número que indica el espacio interior de cada celda, es decir, el margen que se deja en cada celda entre el borde de la misma y el texto contenido.

**Cellspacing** también es exclusivo de <TABLE>. También es un número que indica el espacio entre dos celdas adyacentes.

## 12.2.- Definiendo las filas.

Toda tabla HTML se escribe fila a fila. La fila está constituida por un conjunto de celdas que se escriben en la misma línea. Y toda fila comienza y termina con la etiqueta <TR>.

```
<TR border="n" align="alineación" width="n">
    .... Contenido de esta fila ...
</TR>
```



La etiqueta TR significa en inglés Table Row (fila de tabla). Los atributos opcionales funcionan como los ya explicados pero en este caso sólo afectarán a las celdas de esta fila. Recuerda que *valign* puede contener los valores “top”, “bottom”, “middle”.

## 12.3.- Definiendo las celdas.

Dentro de las filas habrá tantas celdas como columnas tenga la tabla. Cada celda vendrá definida por la etiqueta <TD>, que significa Table Data (datos de tabla).

```
<TD border="n" align="alineación" valign="alineación-vertical" width="n"
    colspan="n" rowspan="n" nowrap>
    .... Contenido de esta celda (código HTML – incluso otra tabla) ...
</TD>
```



Dentro de cada celda podrá anidarse todo el código HTML que sea necesario, textos, imágenes, enlaces, listas, otras tablas, etc.

El atributo opcional *nowrap* avisa al navegador de que no intente romper una fila de texto en un espacio. A veces se usa para intentar justificar el texto. Pero esto se logra de forma eficiente con CSS. *Colspan* y *rowspan* son para combinar celdas, ahora lo vemos, pero antes presentemos un ejemplo de tabla sencilla:

```
<table width="50%" border="2px">
  <tr>
    <td>Celda 1</td>
    <td>Celda 2</td>
    <td>Celda 3</td>
  </tr>
  <tr>
    <td>Celda 4</td>
```



```
<td>Celda 5</td>
<td><a href="www.google.es">Ir a google</a></td>
</tr>
</table>
```


Y el resultado sería:

Tabla de ejemplo:

Celda 1	Celda 2	Celda 3
Celda 4	Celda 5	<a href="#">Ir a google</a>



**PRÁCTICA 19.-** Crea una tabla como la de la figura. Atención a la letra negra.

<b>Isla de Tenerife:</b>	
	<b>Población en 2009:</b> Hombres: 446.171 Mujeres: 453.662
<b>Capital:</b> Santa Cruz de Tenerife	<b>Fuente:</b> <a href="#">ISTAC</a>

## 12.4.- Combinación de celdas.

Para la combinación de celdas recuerda siempre que diseñamos las tablas pensando en la presentación fila a fila. Una celda puede expandirse varias columnas (colspan) o expandirse varias filas (rowspan). Observa la siguiente tabla:

Celda 1	Celda 2	Celda 3
Celda 4	Celda 5	Celda 6
	Celda 7	

El código HTML sería:

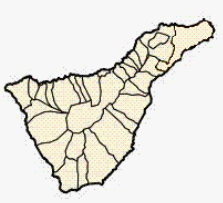
```
<table cols=3 rows=3>
  <tr> <td>celda1</td> <td>celda2</td> <td>celda3</td> </tr>
  <tr> <td rowspan="2"> celda 4 </td> <td>celda5</td> <td>celda6</td> </tr>
  <tr> <td colspan="2" align="center"> celda 7 </td> </tr>
</table>
```

Ten en cuenta que una misma celda puede expandirse tanto en las filas como en las columnas. Esto se logra combinando colspan y rowspan dentro del mismo <TD>.



**PRÁCTICA 20.-** Crea una tabla como la de la figura. Atención a la distribución de celdas y a la alineación en la última celda:

Isla de Tenerife:

	Población en 2009:
	Hombres: 446.171
	Mujeres: 453.662
Capital: Santa Cruz de Tenerife	
Fuente: <a href="#">ISTAC</a>	



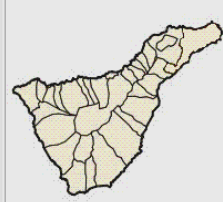
**PRÁCTICA 21.-** Crea una tabla como la de la figura, con el texto descrito en cada celda, si lo tiene.

A	Celda 1	Celda 2	
B		Celda 3	Celda 4
C		Celda 5	Celda 6



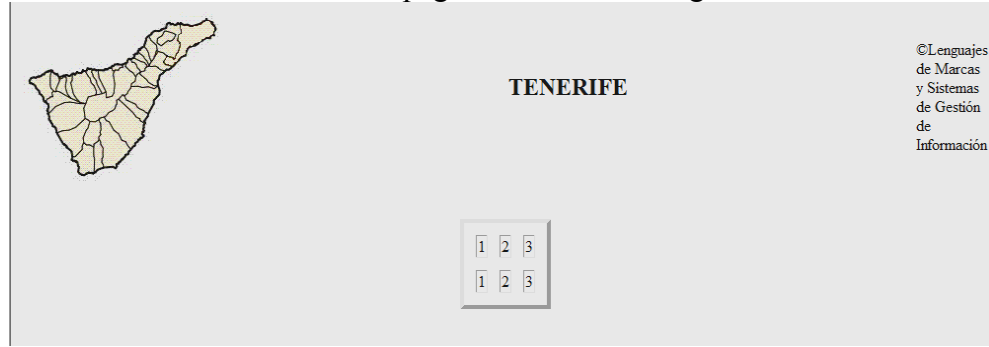
**PRÁCTICA 22.-** Creemos una tabla anidada dentro de otra. Haz una página web con una tabla como la de la figura. La tabla anidada tiene border=4. La externa tiene border=2.

Isla de Tenerife:

	Municipios que imparten ciclos de informática:
	La Laguna
	Santa Cruz
	Puerto de La Cruz
	San Juan de La Rambla
	Candelaria
	Granadilla
	Adeje
Fuente: <a href="#">ISTAC</a>	



### PRÁCTICA 23.- Realiza una página web como la siguiente:



### 13.- Marcos.

Lo primero que tenemos que saber sobre los marcos (en inglés, frames) es que atentan contra todas las normas de accesibilidad web. Fueron un recurso muy utilizado y aún se encuentran en muchos sites pero hoy en día está desaconsejado su uso.

Un marco consiste en partir la zona visible de un navegador, de forma que en cada parte pueda cargarse una página web diferente. Cada zona (llamada marco) podrá actuar independientemente, tener sus propias barras deslizadores (scrolling), tener navegación separada, etc.

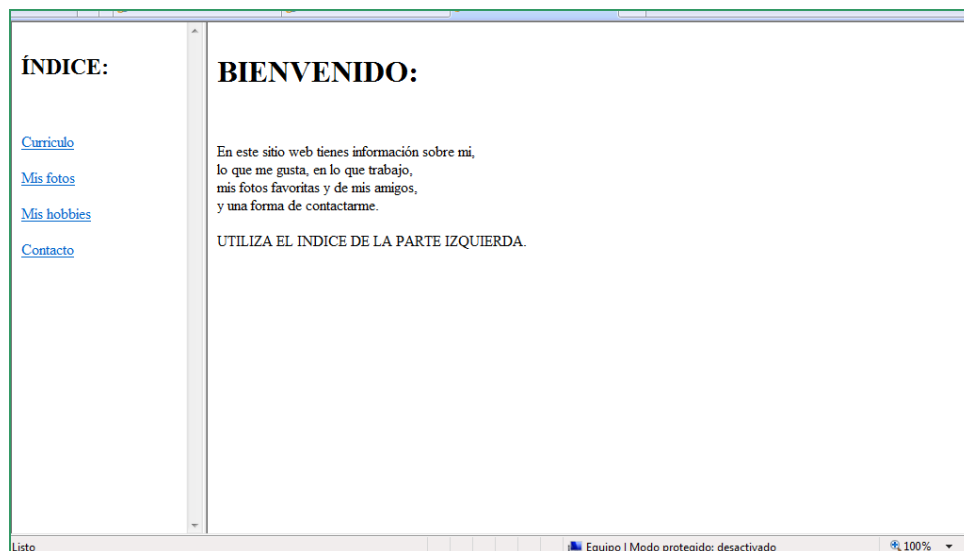
Vamos a realizar un primer ejemplo para que vayas comprendiendo cómo se usan los marcos. La idea es tener un marco en el lateral izquierdo que tenga un índice y el resto de la pantalla para mostrar contenidos. Haremos tres páginas web:

Inicio.html → contendrá la definición de los marcos.

Marco01.html → será el marco del índice.

Marco02.html → será el marco con el contenido.

El resultado final será:



Y pulsando en los enlaces del marco del índice tendremos que cargar diferentes páginas en el marco del contenido. Esa es la idea.

Lo primero que haremos será crear una página donde definimos los marcos. Llama al fichero inicio.html. Su contenido solamente el siguiente código:

```
<FRAMESET COLS="20%, 80%">
  <FRAME SRC= "indice.html" NAME="indice" scrolling="yes">
  <FRAME SRC= "bienvenida.html" NAME="contenido">
</NOFRAMES>
  <P>Lo siento, tu navegador no soporta marcos.</P>
</NOFRAMES>
</FRAMESET>
```

Observando el código anterior verás que lo primero que hacemos es definir un conjunto de marcos con la etiqueta <FRAMESET>.

<FRAMESET cols="x,y,..." rows="x,y,..." >  
 .... Definición de marcos con <FRAME> ...  
 .... O definición de conjuntos de marcos anidados con <FRAMESET>  
 </FRAMESET>



Dentro de una etiqueta <FRAMESET> sólo podrá existir un atributo cols o bien un atributo rows pero NO AMBOS. Lo que hacen es dividir el espacio en columnas o en filas. Observa los ejemplos:

Cols="10%, 30%, \*" → creará espacio para tres marcos en columnas. El primero ocupará un 10% de la pantalla, el segundo un 30% y el tercero (\*) el espacio restante.

1	2	3
---	---	---

Cols="150, \*" → creará espacio para dos marcos en columnas. El primero tiene ancho fijo de 150 píxeles, el segundo (\*) ocupa el espacio restante.

150px	resto
-------	-------

Rows="10%,80%,10%" → creará espacio para tres marcos en filas. El primero y el último ocupan el 10%, el del centro el 80%.

10% de alto
80% de alto
10% de alto



Además de col ó row, dentro de frameset se podrán utilizar los atributos frameborder o framespacing para definir bordes o espacio libre entre los marcos.

Volvamos a nuestro ejemplo. La página inicio.html:

```
<FRAMESET COLS="20%, 80%">
  <FRAME SRC= "indice.html" NAME="indice" scrolling="yes">
  <FRAME SRC= "bienvenida.html" NAME="contenido">
</NOFRAMES>
  <P>Lo siento, tu navegador no soporta marcos.</P>
</NOFRAMES>
</FRAMESET>
```

Se define espacio para dos marcos en columnas, el primero ocupa el 20% y el segundo el 80%. Ahora habrá que definir cada uno de estos dos marcos. Para eso utilizamos la etiqueta <FRAME>, una por cada marco.

```
<FRAME src="archivo" name="nombre"
      scrolling="yes o no" noresize frameborder="n">
```



Atributos opcionales:

NAME	Asigna un nombre a un marco para que después podamos referirnos a él (es para la navegación entre marcos). Es un nombre simbólico.
SRC	Indica la URL de la página que vamos a cargar en este marco.
SCROLLING	Decide si se colocan o no barras de desplazamiento al marco para que podamos movernos por su contenido. Su valor es por defecto AUTO, que deja al navegador la decisión. Las otras opciones que tenemos son YES y NO.
NORESIZE	Si lo especificamos el usuario no podrá cambiar de tamaño el marco.
FRAMEBORDER	Al igual que su homónimo en la etiqueta <FRAMESET>, si lo igualamos a cero se eliminará el borde con todos los marcos contiguos que tengan también este valor a cero.

La etiqueta <NOFRAME> se utiliza sólo por cuestiones de compatibilidad. Si el usuario utiliza un navegador que no soporta marcos obtendrá un mensaje de aviso.

Continuando con nuestro ejemplo:

```
<FRAME SRC= "indice.html" NAME="indice" scrolling="yes">
<FRAME SRC= "bienvenida.html" NAME="contenido">
```

El primer marco mostrará la página indice.html y además a este marco le hemos dado el nombre de “indice”. El segundo marco, que hemos llamado “contenido” mostrará inicialmente la página bienvenida.html.



**PRÁCTICA 24.-** En esta primera práctica completa el ejemplo desarrollado en este apartado. Tendrás que implementar las páginas a las que hacen referencia los marcos como se muestra en la figura de la página 44.



### 13.1.- Navegación entre marcos.

Si queremos continuar desarrollando el ejemplo tendremos que hacer el resto de páginas de cada enlace del índice: `curriculo.html`, `mis_fotos.html`, `mis_hobbies.html`, `contacto.html`.

Prueba a realizar la primera de las páginas. ¿Qué ocurre al pulsar en el enlace? No se carga en el marco adecuado. Por defecto los enlaces cargan las páginas en el marco actual, tendremos que indicar que queremos cargar la página en otro marco. Para eso usamos en el vínculo el atributo **target** (destino). Y aquí es donde participa el atributo **name** de `frame`.

```
<a href="curriculo.html" target="contenido">Currículo</a>
```

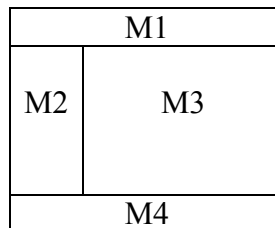


**PRÁCTICA 25.-** Finaliza la práctica haciendo que los enlaces funcionen correctamente.

Nota: el atributo `target` se usa en los vínculos para más cosas. Otros valores que puede contener son `_blank`, `_top`, `_parent`, `_self`, etc. Sirven para cargar páginas en ventanas nuevas o en otro orden de jerarquía anulando incluso marcos. En nuestro ejemplo la página `inicio.html` “es padre” de las páginas `indice.html` y todas las demás.

### 13.2.- Marcos complejos.

Anidando la etiqueta `<FRAMESET>` se puede realizar construcciones complejas de marcos. Veamos una sencilla:



```
<frameset rows="10%, 80%, 10%">
  <frame src="..." name="M1">
  <frameset cols="20%, 80%">
    <frame src="..." name="M2">
    <frame src="..." name="M3">
  </frameset>
  <frame src="..." name="M4">
</frameset>
```



## PRÁCTICA 26.- Crea la siguiente distribución de marcos:

M1		
M2	M3	M6
	M4	
M5		

### 14.- Formularios.

Los formularios en HTML son agrupamientos de unos elementos especiales que se llaman **controles**, cuyo objetivo general es permitir al usuario de la página web introducir información, o seleccionar aspectos que le interesan, es decir, darle cierta interactividad.

Iremos entendiendo los formularios mejor conforme veamos los tipo de controles existentes: botones, listas de selección, áreas de texto, etc. Pero comencemos diciendo que todo formulario viene definido por la etiqueta `<FORM>` cuya sintaxis básica es la siguiente:

```
<FORM action="url" method="método">
    ... texto, etiquetas y controles del formulario...
</FORM>
```



- El atributo `action` contiene una URL o localización web. Típicamente es la página o programa que recibirá los datos del formulario.
- El atributo `method` puede contener uno de los siguientes valores: GET o POST. La diferencia entre ellos es la manera de enviar la información de los controles mediante el protocolo http, utilizado para transferir las páginas web a través de la red. Por el momento sólo debes saber que mientras GET envía los datos en la cabecera de los paquetes http, POST lo hace en el cuerpo de los mismos.

#### Curiosidad:

Los formularios abren una puerta a posibles ataques de Hawking en nuestro servidor. El método GET además hace que los parámetros de programación internos resulten visibles para los usuarios, siendo más inseguro que el método POST.



Podemos enviar los datos del formulario a un correo electrónico. Para eso podemos modificar la etiqueta FORM de la siguiente manera:

```
<FORM action=mailto:micuenta@miservidor
        enctype="text/plain" method="POST">
    .... Contenido del formulario ...
</FORM>
```

Pero un formulario no tiene mucho sentido hasta que incorpora algún **control**, luego comencemos definiendo los controles existentes en HTML.

### 14.1.- El control INPUT.

Es el control que permite recibir entradas del usuario. El valor por defecto define una entrada de texto simple, pero existen multitud de controles INPUT distintos, definidos por el atributo TYPE. Veamos la sintaxis de esta etiqueta:

```
<INPUT TYPE="tipo" NAME="nombre" atributos opcionales>
```



Es necesario saber que input no tiene etiqueta de cierre. Veamos, uno por uno, los distintos tipos de INPUT.

### 14.2.- INPUT de texto.

Define una entrada de texto sencillo. Si en la etiqueta input no se define el atributo type estaremos en un INPUT de texto por defecto. Sintaxis:

```
<INPUT TYPE="TEXT" NAME="nombre" value="texto inicial"
      Size="ancho" maxlength="ancho de escritura">
```



Prueba este ejemplo:

```
<HTML>
<HEAD><TITLE>Ejemplo de formulario:</TITLE></HEAD>
<BODY>

  <h3>DATOS PERSONALES:</H3><br>
  <FORM name="miformulario" action="mimail" method="post">
    NOMBRE:
    <INPUT type="text" name="nombre" value="su nombre aquí" size="20"
maxlength="20">
    <br>
    APELLIDO:
    <INPUT type="text" name="ape" value="su apellido aquí" size="40"
maxlength="50">
    <br>
    EDAD:
    <INPUT type="text" name="edad" size="3" maxlength="2">
  </FORM>
</BODY>
</HTML>
```



### 14.3.- INPUT de SUBMIT y RESET.

El formulario no se enviará hasta que se acepte. Para aceptar un formulario será necesario pulsar en un botón de tipo SUBMIT. La sintaxis es la siguiente:

```
<INPUT TYPE="submit" value="texto del botón">
```



Este botón provoca que el formulario se envíe a la URL definida en el atributo action.

```
<INPUT TYPE="reset" value="texto del botón">
```



Los botones de tipo reset, por su parte, limpian el formulario restableciendo todos los controles a sus valores por defecto, si los tienen.



**PRÁCTICA 27.-** Con el código de la práctica anterior modifica el formulario agregando un botón con el texto ACEPTAR y otro con el texto CANCELAR. Al pulsar el primero se enviará un correo a tu cuenta. Al pulsar el segundo se enviarán los campos.

Nota: tienes que tener un cliente de correo, como OUTLOOK, instalado, y en tu navegador tiene que estar definido como el programa de correo predeterminado (mira en opciones de Internet → programas). Prueba con varios navegadores como Explorer, FireFox o Safari para que observes las diferencias.

### 14.4.- INPUT de contraseña.

Es un botón exactamente igual que TEXT pero lo que escribe el usuario aparece oculto sólo para él mismo, de forma análoga a cuando introduces una contraseña. Tiene todos los atributos que tienen los botones de texto, salvo que TYPE="password".

### 14.5.- INPUT ocultos.

Se utilizan sobre todo en programación. Es un botón cuyo TYPE="hidden", esto significa que no aparecen en la página (no se visualizan). Se utilizan para transferir entre formularios datos de páginas anteriores o datos que el usuario no controla.

### 14.6.- INPUT para activación de varias opciones simultáneas.

Se trata de botones con TYPE="checkbox", o traducido "caja de verificación". La idea es que el usuario pueda seleccionar varias opciones que son de su interés a la vez. La sintaxis es la siguiente:

```
<INPUT TYPE="checkbox" name="nombre" checked>
```



El atributo name se recibirá con los valores on/off, dependiendo de si el usuario los seleccionó o no. Puedes hacer varios checkbox dando a cada uno un atributo name

diferente. En caso de que quieras que uno de ellos aparezca seleccionado de forma predeterminada utiliza el atributo checked.



**PRÁCTICA 28.-** Realiza el código HTML para obtener un formulario como el de la figura. Además define un campo oculto con nombre IES cuyo valor sea “Pérez Minik”.

#### 14.7.- INPUT para activación de opciones excluyentes.

Si queremos utilizar opciones excluyentes, entre las que sólo puede estar activa una de ellas, utilizaremos botones con TYPE=”radio”. Para que sean excluyentes entre sí debemos agruparlos dándoles el mismo atributo name.

`<INPUT TYPE=”radio” name=”nombre” value=”valor” checked>`



En este caso al activar el formulario se enviará el atributo value de la opción que esté seleccionada. Podemos hacer los conjuntos de botones radio que queramos, pero recuerda que el atributo name es el que los agrupa en un conjunto excluyente. Por su parte checked funciona igual en los ckeckbox.



**PRÁCTICA 29.-** Completa el formulario anterior para que aparezca así:



### 14.8.- Áreas de texto.

Este control permite introducir un texto de varias líneas. Se usa para textos relativamente grandes, como por ejemplo, recoger los comentarios de un usuario.

```
<TEXTAREA cols="columnas" rows="filas" name="nombre" >  
    Texto inicial que queramos mostrar  
</TEXTAREA>
```



Especificamos las columnas y las filas en la cabecera. Observa que la etiqueta de cierre es obligatoria. Si quieres ajustar mejor el texto en el interior puedes definir en la cabecera el atributo wrap. En principio se soportan comentarios de 32700 caracteres. No existe el atributo value.

### 14.9.- Listas de opciones.

Este control permite seleccionar una o varias opciones de una lista.

```
<SELECT name="nombre" size="n" multiple>  
    <OPTION VALUE="valor1" selected> Texto1  
    <OPTION VALUE="valor2"> Texto 2  
  
    ....  
    </OPTION VALUE="valorN"> Texto N  
  
</SELECT>
```



Comenzaremos con la etiqueta select, en la que especificamos el nombre y opcionalmente la altura visible en líneas (size) y si permite selección múltiple (multiple). En ausencia de atributos la altura visible es uno y no permiten selección múltiple.

Después por cada opción que queramos colocar a la lista pondremos una etiqueta OPTION. El atributo value es interno y no visible por el usuario, y corresponde con el valor que el formulario enviará. El texto de cada opción es lo que el usuario observará en la página web. Si quieres que una opción esté seleccionada por defecto debes colocarle el atributo selected.

Lo que se envía al aceptar el formulario será el nombre del atributo name de SELECT junto al value de la opción seleccionada.

Si estás trabajando con selección múltiple recuerda que el usuario podrá seleccionar varias opciones manteniendo la tecla de mayúsculas pulsada.





### PRÁCTICA 30.- Crea el siguiente formulario:

La lista de color de fondo favorito debe enviar el código RGB del tono escogido por el usuario. No permite selección múltiple y por defecto estará seleccionado el rojo.

La lista de intereses sí es de selección múltiple y permite visualizar 4 filas de altura. Además de las visibles existe una quinta opción que es “Cotilleos” y una sexta que será “Noticias”.

A menudo nos puede interesar agrupar opciones dentro de una lista. Algo parecido a hacer un submenú. Para eso puedes encerrar las opciones que quieras entre la etiqueta <OPTGROUP>, que tiene su propio cierre obligatorio. Observa:

```
<SELECT name="Aeropuertos" size="6" >
  <OPTGROUP LABEL="Europa">
    <OPTION VALUE="01"> Franckfurt
    </OPTION VALUE="02"> London Gateway
  </OPTGROUP>
  <OPTGROUP LABEL="Asia">
    <OPTION VALUE="03"> Bankok
    </OPTION VALUE="04"> Shangai
  </OPTGROUP>
</SELECT>
```



Prueba la visualización de esta lista.



### PRÁCTICA 31.- Agrega la tercera lista al formulario anterior con información de las sucursales, así como un textarea para recoger los comentarios:



**INTERESES:**

COLOR DE FONDO FAVORITO:

INTERESES:

SUCURSALES:

Introduzca sus comentarios:

Además de las visibles las sucursales en La Gomera están en Vallehermoso y San Sebastián. En la Palma las tenemos en Mazo, El Paso y en Garafía.

#### 14.10.- Botones genéricos y programación.

Existen también los `<INPUT TYPE="image">` y los `<INPUT TYPE="button">`. Los primero sería botones que contendrían una imagen. Y los segundos son botones que permiten hacer click con el ratón sobre ellos. Pero estos dos se utilizan menos que la etiqueta `<BUTTON>` que define un botón genérico que puede contener texto o una imagen.

```
<BUTTON name="nombre" value="Rótulo" disabled eventos>
... contenido, ya sea texto, imagen u ambos...
</BUTTON>
```



Observa que lo primero que es diferente es que button tiene etiqueta de cierre obligatoria. Podemos poner el texto del botón en el atributo value o ignorar este y colocarlo entre las etiquetas. Podemos poner entre las etiquetas una imagen.

Disabled es un atributo que tienen todos los botones (todos los INPUT), los textarea y los select. Permite desactivar un control, esto es, no se envía su valor por el formulario y el usuario no puede alterarlo.

Hay otro atributo similar es readonly. En este caso se permite el envío del valor por el formulario pero el usuario es el que se encuentra el control bloqueado.

¿Para qué se usan los botones BUTTON? Pues son fundamentales en la programación Web porque permiten enlazar con scripts o trozos de código escritos en lenguajes de programación. Típicamente en las páginas web se usa JavaScript para ejecutar código dinámico en el equipo del cliente. La conexión entre el código HTML y

el lenguaje de programación pertinente se hace por medio de los **EVENTOS**, que empiezan con la palabra **on** (= **cuando...**). Observa algunos eventos:

- Onclick → se lanza cuando se hace click de ratón sobre el control.
- Onmouseover → se activa cuando pasa el ratón por encima.
- Onmouseout → se activa cuando el ratón sale del área del control
- Onfocus → se activa cuando el control logra tener el cursor.
- Onblur → se activa cuando el control pierde el cursor.
- Onkeypressed → se activa al pulsar una tecla en el teclado.

Hay multitud de eventos, para todos los gustos. Y cada control tiene los suyos propios definidos. Sin embargo en esta asignatura no podemos ver el lenguaje de programación JavaScript. Hay multitud de cursos en la web, te recomiendo que si te gusta esto del diseño y programación web aprendas este lenguaje. Nosotros veremos solamente unos ejemplos sencillos:



**EJEMPLO 01.**- Observa la siguiente página. Hemos definido un SELECT con un evento **onchange**. También hemos definido un BUTTON con un evento **onclick**.

**EVENTO ONCHANGE sobre una etiqueta SELECT:**

COLOR DE FONDO FAVORITO: cambia el color

EVENTO ONCLICK sobre un BUTTON:

Ahora observa el código fuente HTML. A un evento siempre le acompaña un código JavaScript que ejecuta alguna función de programación. En este caso acceder a document y a su propiedad bgcolor (color de fondo) para cambiarlo.

```
<HTML>
<HEAD><TITLE>Ejemplo de formulario:</TITLE></HEAD>
<BODY>

<h3>EVENTO ONCHANGE sobre una etiqueta SELECT:</H3>
<FORM name="miformulario">
  COLOR DE FONDO FAVORITO: cambia el color
  <SELECT name="color" onchange="document.bgColor=this.value">
    <option value="00FF00"> Verde
    <option value="FF0000"> Rojo
    <option value="0000FF"> Azul
    <option value="FFFFFF" selected> Blanco
  </SELECT>
  <hr noshade>
  EVENTO ONCLICK sobre un BUTTON:
  <BUTTON name="btn1" onclick="document.bgColor='#C7C7C7'">
    Poner el fondo gris
  </BUTTON>
</FORM>
</BODY>
</HTML>
```



Ahora haremos un efecto llamado **rollover**. Consiste en tener una imagen y cuando el ratón pasa por encima se muestra otra diferente. Tenemos dos imágenes llamadas `triste.jpg` y `alegre.jpg`. Haremos un Button que contiene a la imagen triste. Y la idea es que cuando el ratón pase por encima se vuelva alegre y cuando el ratón deje de estar encima vuelva a estar triste. Observa el código:



#### **EJEMPLO 02.-** Efecto ROLLOVER:

Ratón por encima (onmouseover):



Ratón no encima (onmouseout):



Aquí está el código:

```
<HTML>
<HEAD><TITLE>Ejemplo de formulario:</TITLE></HEAD>
<BODY>

    EFECTO ROLLOVER:
    <BUTTON name="btn1" onmouseover="document.icono.src='alegre.jpg'"
              onmouseout="document.icono.src='triste.jpg'">
      <IMG name="icono" src="triste.jpg" width="50px" height="50px">
    </BUTTON>

</FORM>
</BODY>
</HTML>
```

Realmente considero que es una pena que no podamos dar JavaScript en esta materia. Es muy interesante lo que puede llegar a hacerse con las páginas web.

## **15.- Sonidos y otros objetos.**

Una página del WEB puede tener sonidos incorporados, bien sea como un fondo sonoro que se ejecuta automáticamente al cargar la página, o como una opción para que la active el propio usuario.

Para poder escuchar los sonidos es necesario disponer, como es lógico, de una tarjeta de sonido con sus correspondientes altavoces. Pero esto no es suficiente, pues no todos los navegadores están capacitados en la misma medida.

Explorer de Microsoft incorporó opciones de sonido a partir de su versión 2.0. Es capaz de reproducir fondos sonoros sin necesidad de añadir nada, y no hay ninguna complicación con los servidores, como ocurre con el Netscape. Además, a partir de la versión 3.0 del Explorer, es incluso compatible con los plug-ins del Netscape.

En Netscape no está soportado antes de las versiones 2.0. Requiere activar complementos para ejecutar formatos como el .wav o el .mid. En la versión 2.0 se desarrolló un plug-in que había que instalar. Y la versión 3.0 ya traía instalado un complemento llamado Live-Audio.

\* Sonido de fondo en Internet Explorer:

```
<BGSOUND src="URL del archive de sonido LOOP="n">
```



Así podemos ejecutar un archivo de sonido de fondo. Loop es el número de veces que lo oiremos. Si queremos oírlo sin parar haremos LOOP=infinite. Si no existe LOOP se ejecuta el archivo una única vez.

\* Sonido de fondo en Netscape:

```
<EMBED src="URL del archive de sonido width="x" height="y">
```



Puedes agregar los modificadores Autostart="true" y Loop="True". En principio aparece una consola que puedes ocultar con el atributo Hidden="true".

Para oír el sonido en ambos exploradores podemos colocar las dos etiquetas. Cada navegador ignorará la que no sea capaz de reconocer. Recuerda que estas etiquetas se definieron para formatos .wav y .mid.

```
<BGSOUND SRC="sonido.mid" LOOP=INFINITE>
<EMBED SRC="sonido.mid" WIDTH=200 HEIGHT=55 AUTOSTART="TRUE"
LOOP="TRUE" HIDDEN="TRUE">
```

**Nota:** estas etiquetas pueden cambiar entre las distintas versiones de los navegadores. No son estándar con lo que pueden ocasionar problemas. Además recuerda lo molesto que es una página web de reproduce sonido de fondo que no podemos parar.

Para evitar ese sonido de fondo puede ser necesario que el sonido lo pueda activar el propio usuario. Lo que podemos hacer es definir un vínculo a un archivo de sonido que se abra en otra página distinta a la nuestra:

```
<A HREF="file:misonido.mid" target="_BLANK">Escuchar sonido</A>
```

Para que funcione el navegador tiene que tener activado los plug-in de reproductores de sonido en el formato adecuado. Si no es así podrá preguntarnos en qué programa ejecutamos después de descargar el fichero. Podrías hacer el enlace con un icono colocando la imagen dentro del vínculo.



Los sonidos no son lo único que se puede incrustar en las páginas web. Existe una etiqueta genérica llamada `<OBJECT>` que permite vincular en una página cualquier objeto que un navegador sea capaz de reconocer. Cada fabricante debe indicar los parámetros de object necesarios para visualizar correctamente el objeto. Esto es lo que se usa, por ejemplo, para adjuntar los applets, pequeños programas capaces de descargarse automáticamente al ordenador del usuario y escritos en el lenguaje JAVA.

Para funcionar la etiqueta `<OBJECT>` necesita los atributos `CLASSID` con una especificación del objeto a incrustar. Y para que ese objeto reciba información de configuración se pueden utilizar los atributos `PARAM`.

Esta etiqueta depende y varía con cada objeto, con lo que no la veremos aquí. En Internet tienes multitud de ejemplo de objetos que puedes incrustar en tus páginas.



### **PRÁCTICA FINAL.**

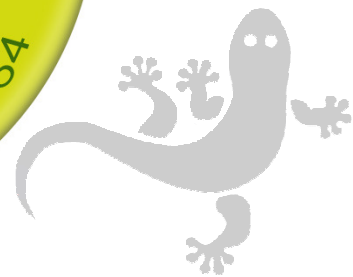
Crea tu propia página personal.

Dentro de tu página crea un pequeño site con información del lenguaje HTML.

Utiliza un diseño sencillo y efectivo. Evita colores demasiado intensos o muchas imágenes animadas saltando, etc. Intenta que tenga un aspecto correcto.



# I.E.S. DOMINGO PÉREZ MINIK



## **CICLO FORMATIVO DE GRADO SUPERIOR:** ***“ADMINISTRACIÓN DE SISTEMAS INFORMÁTICOS EN RED (L.O.E.)”***



**MÓDULO:**

**Lenguajes de marcas y sistemas de  
gestión de información**

**(4 horas semanales)**