

Tema

DISEÑO WEB: HTML 5 y CSS 3



El **diseño de páginas Web** es una amplia área de aplicación del diseño gráfico en la cual se integran conocimientos propios del diseño como son la composición, el uso de color y la tipografía con conocimientos técnicos del medio como son los lenguajes HTML y CSS , así como conocimientos sobre usabilidad, accesibilidad y organización de un sitio web.

1

Antes de empezar a diseñar

■ Consejos generales para diseñar una página

La realización y creación de páginas WEB es una labor que requiere de una cierta estructura y planificación al comienzo de su elaboración. El resultado final de nuestro trabajo dependerá de la organización, la originalidad y los elementos utilizados en las páginas que se creen. Debe tenerse siempre presente, a quien va dirigida nuestra WEB, para adecuar los contenidos y el estilo en función del colectivo que va a leer nuestras páginas. Cuide que el lector no se pierda en la navegación, por lo que se debe estructurar de una forma clara la relación de unas páginas con otras.

1. Definición de los objetivos.

El primer paso, para la creación de páginas WEB bien estructuradas, es saber de antemano el mensaje que se desea transmitir. Las imágenes y los elementos a utilizar serán distintos si se plantea una WEB para vender un producto, o para dar información o para publicar un trabajo científico. En este último caso, la página debe dar una imagen más formal.

2. Perfilar una línea de diseño.

Hay que tener en cuenta que los WEBs profesionales tienen una apariencia más formal, y que su diseño es tan importante como el de la imagen de una institución.

En un plano más profesional, se debe cuidar la presentación, manteniendo siempre una imagen corporativa a lo largo de todas las páginas y un estilo adecuado al perfil de la institución. Esto te permitirá recopilar los materiales que incluirás en ella: imágenes, gráficas, impresos, formularios, logotipos, etc.

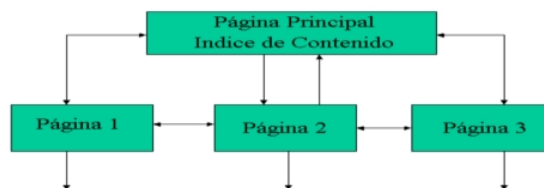
La línea de diseño debe ser coherente y homogénea, que mantenga una apariencia visual común: los mismos colores o motivos de fondo, el mismo estilo de líneas divisorias horizontales y verticales, los mismos iconos o viñetas o los mismos logotipos gráficos siempre en la misma posición de la página.

Su aspecto profesional con elementos comunes hará que los usuarios visiten las páginas con orden y no se pierdan pensando que están en otra Web.

3. Crear el organigrama de navegación.

A continuación debes planificar cómo van a desplazarse los usuarios por ellas. Se debe realizar un diagrama de flujo claro para definir los índices principales, las páginas secundarias y las ramificaciones de todos los documentos.

Si utilizas una sola página WEB y esta es muy extensa deberás poner enlaces a las diferentes partes del documento al principio y al final de tu página y de esta forma facilitar la lectura de la misma, sin tener que depender tanto de las barras de desplazamiento.



4. Definición de estilo

Una vez creada la estructura básica del WEB, se debe definir el estilo de las páginas. Resulta práctico crearse una página de estilo (CSS) para aplicar a todas las páginas que vayamos generando, incluyendo los logotipos, encabezados y pies.

■ Reglas para estructurar y diseñar una página Web

A continuación vamos a ver unas cuantas normas que deberemos tener en cuentas antes de empezar a programar una Web:

1. Definir el contenido de las páginas:

- ✓ Use gráficos y colores y varíe el tamaño de la letra para realzar la página.
- ✓ Incluya siempre una alternativa de texto a los elementos gráficos.
- ✓ Reduzca el texto al mínimo, es decir no es conveniente presentar textos muy largos, ya que el usuario no los lee.
- ✓ Reduzca al mínimo los estilos de encabezamientos y subtítulos cuando organice el contenido y utilice los estilos de forma coherente.
- ✓ Puede insertar líneas horizontales para separar visualmente las secciones del documento.

2. Combinación de fondos con el texto:

El fondo puede hacer que la lectura de los documentos en pantalla sea más atractiva. Sin embargo, los colores e imágenes oscuras pueden interferir con los colores del texto, dificultando su lectura. Es conveniente aplicar al texto un color de contraste, por ejemplo: un color claro para texto sobre fondo oscuro y viceversa. Pero, trate de evitar la utilización de textos de color blanco, pues algunas impresoras no escriben en dicho color y también evitar las texturas complejas en los fondos de las páginas.

3. Establecer la longitud de la página (Resolución de pantalla):

Si diseñamos una página para una resolución dada, ocupando toda la ventana del navegador, aquellos usuarios que la visualicen a resoluciones menores no tendrán espacio en pantalla para contener toda la página, por lo que se verán obligados a usar las barras de desplazamiento del navegador. Por el contrario, aquellos usuarios que la visualicen a resoluciones mayores tendrán demasiado espacio en pantalla para tan poca página, por lo que les quedará bastante espacio vacío, sin contenidos.

Para solucionar estas diferencias, lo normal es que se diseñen las páginas web para una resolución base, generalmente la más usada en la actualidad (800x600), y se construyan internamente mediante tablas o capas de tamaños relativos, con anchuras definidas en %, con lo que se consigue que al ser visualizadas en monitores de más resolución se "abran", ocupando todo el espacio de pantalla disponible.

Otra posibilidad es maquetar toda la página dentro de un contenedor padre (una tabla o capa) y asignar a éste una alineación centrada, con lo que la página quedará en el centro de la pantalla si se usa una resolución mayor que la de diseño.

No es recomendable hacer una página web muy grande verticalmente mejor dividirla en varias páginas que nunca supere el tamaño de página y media.

4. Reducir el tamaño de los archivos de imágenes:

Las imágenes son un elemento importante en un sitio Web. Estas vienen en dos formatos básicos: GIF y JPEG. Cada formato tiene sus ventajas e inconvenientes.

Las imágenes GIF son preferibles cuando la imagen va a contener menos de 256 colores y cuando se utilizan transparencias. Los GIF son también apropiados para animaciones sencillas. Las imágenes JPEG se utilizan en imágenes con más de 256 colores, como son las fotografías escaneadas.

Para reducir el tamaño de un archivo gráfico, utilice menos colores, reduzca el alto y el ancho del gráfico o recórtalo.

5. Herramientas de diseño y organización:

Las tablas han sido un elemento fundamental en HTML porque con ellas organizábamos todos los elementos de una página Web. **Las capas** están sustituyendo a las tablas como herramienta de diseño y de organización de la información. Las columnas, los gráficos y el texto pueden organizarse de forma que aparezcan alineados en los navegadores Web. Si no se utilizan estos elementos, es complicado mantener los gráficos y el texto alineados en HTML (formato de las páginas Web).

6. Uso de distintos navegadores y dispositivo (Responsive):

Cada navegador tiene pequeñas diferencias al interpretar el código HTML.

Es recomendable probar nuestras páginas en navegadores que sigan fielmente los estándares **de HTML** (creados por W3C) como por ejemplo **Amaya o Mozilla Firefox**.

Nunca hagas páginas específicas para un navegador, tu página tiene que óptima para cualquier resolución de pantalla y cualquier navegador.

Recomiendo probar las páginas que vais haciendo en los navegadores más utilizados que son Chrome, Firefox y edge. También es recomendable probarlo en Tablet y móvil.

■ ¿Qué necesitas para crear una Web?

Para empezar a crear tu página necesitas un editor de textos. Por ejemplo el **Bloc de notas o Notepad** (Windows) o **Gedit** (Linux). Existen también otros editores especializados en este lenguaje que nos pueden hacer la vida más fácil.

Existen programas que nos permiten diseñar la página como si estuviéramos escribiendo un documento con un editor del tipo de Word. El editor de HTML es el encargado de vérselas con el lenguaje y programar internamente la página con el código HTML según lo que nosotros estamos diseñando. No hace falta conocer el lenguaje HTML para programar una página Web. Estos programas se denominan habitualmente **WYSIWYG (What You See Is What You Get)** porque cuando trabajas con ellos lo que ves que estás creando con el editor es lo que obtienes luego cuando grabas la página. En el mercado existen multitud de editores de HTML WYSIWYG, es importante elegir un editor bueno porque nuestros trabajos van a depender de sus resultados. Actualmente el rey de los editores y el que os aconsejaríamos sin duda es el **Dreamweaver**. Otros programas son **Mozilla Composer** y **Frontpage (Microsoft)** que no aconsejo su uso porque no genera HTML correcto.

Ventajas e inconvenientes HTML / WYSIWYG	
Escribiendo el HTML	Con un editor WYSIWYG
<p>Dominas con mayor precisión el código de la página, queda más limpio. Si dominas bien el HTML nunca tendrás ningún problema para hacer lo que desees.</p> <p>Es más complicado el aprendizaje, más lento y cuando se llega a un nivel avanzado también se hace considerablemente más difícil.</p> <p>Hacer una página cuesta más trabajo y tiempo.</p>	<p>El código de la página tiene peor calidad, incluso puede llegar a tener errores, más o menos visibles, que cuestan de arreglar. Es la máquina la que domina el trabajo.</p> <p>El aprendizaje es muy sencillo, tal como puede ser trabajar con Word. Solo se trata de manejar un programa más.</p> <p>Es muy rápido.</p>

Recomiendo saber HTML y utilizar un programa WYSIWYG como Dreamweaver para crear las páginas pero modificando el código HTML a tu gusto cuando te haga falta.

Otra cosa que necesitas es un navegador de Internet para poder ver como está quedando tu página. Para diseñar tu página no es necesario por el momento que dispongas de una conexión a Internet. Puedes estar modificando el código de tu archivo en el editor de textos y estarlo visualizando en el navegador sin tener que conectarte.

2

Introducción HTML

HTML es un lenguaje que para describir la estructura de una web. El autor le cuenta al navegador cómo es la web; y es trabajo del navegador decidir cómo va a mostrar cada sección (colores, tamaños, tipos de letra, ...). Si el autor quiere además especificar el diseño, puede usar CSS (hojas de estilo), que se explica más adelante. **No utilices HTML para establecer el diseño de tus páginas.**

Existen varias versiones diferentes de HTML, Se hizo estándar en 1995, y desde entonces las cosas han cambiado (aunque el lenguaje no se ha modificado mucho). En cada página que hagas, tendrás que decirle al navegador qué versión de HTML estás usando (no puede detectarlo). Eso se hace con la primera línea, la del **!DOCTYPE**.

Cada versión de HTML tiene variantes, (Va colocada al principio del documento HTML):

La línea utilizada para el HTML 5 (Utilizar en clase)

```
<!DOCTYPE html>
```

La línea utilizada para XHTML (Más estricto)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Una vez conocemos el concepto de HTML os vamos a adelantar algunas cosas más. Así pues, el archivo donde está contenido el código HTML es un archivo de texto, con una peculiaridad, que tiene **extensión .html** o **.htm** (es indiferente cuál utilizar).

El lenguaje consta de etiquetas que tienen esta forma **Texto ** (Convierte en negrita el Texto) o **<P> Texto </P>** (Establece un párrafo con los espacios correspondientes). Cada etiqueta tiene una función predeterminada y puede llevar **atributos** para definir la etiqueta . Ejemplo:

```

```

Nota: Cuando sólo se escribe una etiqueta en lugar de dos se cierra al final como en el ejemplo

Algunos consejos para escribir HTML son:

- Escribir las etiquetas en minúsculas
- Los atributos siempre entrecomillados.
- Cerrar todas las etiquetas y **en orden**. Ejemplo:

No es válido la sentencia: **<P> Texto </P>**
La sentencia válida sería: **<P> Texto </P>**

En los próximos capítulos se explicará cuáles son las etiquetas que son recomendables utilizar y las que se están quedando obsoletas (aunque todavía su uso está muy extendido) porque se puede hacer lo mismo utilizando CSS con las ventajas que conlleva esto.

■ Validar una página de HTML

Para comprobar que nuestras páginas utilizan un HTML correcto que funcione en todos los navegadores se suele utilizar una herramienta como el **validador de HTML** (<http://validator.w3.org/>) del W3C (es quien creó el estándar HTML). Le pones la dirección de la página o un fichero de tu disco, y te dice qué errores le encuentra.

No hagas eso de probar en varios navegadores a ver si se ve bien; la prueba definitiva es ver si el código valida en el test o no. Si valida, todos los navegadores lo mostrarán bien, porque todos entienden HTML.

3

Primeros pasos

En este apartado vamos a explicar como se organiza un sitio web y crearemos nuestra primera página.

Mi sitio Web

Crear un sitio Web consiste en elaborar la estructura de directorios que van a contener nuestras paginas, imágenes, documentos, hojas de estilos,...

Crearemos un **directorio** donde guardaremos todos los ficheros estructurados en diferentes carpetas dependiendo de nuestro sitio web. El fichero principal suele llamarse **index.html**. Como mínimo tiene la siguiente estructura:

- /Mi pagina Web (carpeta)
- /Imagenes (carpeta)
- /Documentos (carpeta)
- /CSS/estilos.css (carpeta)
- index.html (fichero principal)
-

Nota: Se recomendable no incluir acentos ni espacios en los nombres de los ficheros y carpetas de nuestro sitio web.

Dreamweaver

Para establecer un sitio web (**Sitio/Crear sitio**) te pedirá la siguiente información:

- Carpeta de tu sitio Web y dirección Web (si la tienes)
- Tecnología de servidor (ASP,JSP,...)
- Configurar nuestro servidor para trabajar localmente y en el servidor conjuntamente.

Estructura de una página Web

Todos los documentos en XHTML 1.0 (Estricto) deben tener la siguiente estructura:

Nota: Si utilizamos en lugar de xhtml una versión de html colocamos su doctype y la etiqueta <html> no lleva atributo xmlns

```
<!DOCTYPE html>
<html >
  <head>
    <title>Título principal</title>
    <meta name="description" content="Esto es una descripción" />
    <meta name="keywords" content="palabra1, palabra2, palabra3" />
    <meta name="author" content="Nombre" />
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  </head>
  <body>
    <p>Aquí va el contenido.</p>
  </body>
</html>
```

La etiqueta **<title>** es muy importante porque lo usan los buscadores para indexar la página,y las personas para saber en qué web entrar cuando un buscador muestra cientos de ellas. Así que elige un buen título, y ni se te ocurra dejarlo en blanco.

4

Introducción CSS-Hojas de estilo

Las hojas de estilo son útiles para definir atributos visuales en documentos HTML . Esto proporciona métodos poderosos para definir el estilo visual del documento mientras separa la parte semántica (HTML) de la de presentación (hojas de estilo).

Las Hojas de estilo(CSS) sirven para aplicar Colores, tamaños, márgenes, interlineados, efectos, fondos, bordes,... a nuestra página escrita en HTML. Al tener la página y los css por separado es muy fácil modificar el estilo de la página sin tocar el código HTML.

■ Propiedades del estilo

Las propiedades y sus valores son definidos utilizando una sintaxis simple de CSS (hojas de estilo en cascada). Estas definiciones no pertenecen al estándar HTML, pero están pensados para reemplazar los atributos de presentación que HTML ha adquirido en sus primeros tiempos.

nombre-propiedad: valor-propiedad;

De esta forma, puedes definir múltiples valores para múltiples propiedades en un mismo bloque dado que una definición básica es separada de la siguiente por el punto y coma. Un grupo de propiedades y valores puede ser aplicado a un elemento, el cual establecerá el estilo visual de dicho elemento en todo el documento.

Aunque el código CSS puede ir dentro del código HTML como mostraremos recordamos que es recomendable utilizar siempre clases por medio de un fichero externo. Vamos ahora a describir los diferentes usos de las CSS introducidos en el anterior capítulo. Vamos por orden, describiendo los puntos según su dificultad e importancia.

■ Definir estilos a una página o sitio Web

Lo más normal es escribir todo el código CSS en un fichero externo, de extensión css, y luego incluir en cada página entre **<head></head>**:

```
<!DOCTYPE html>
<html " ">
<head>
  <title>Título principal</title>
  <meta name="description" content="Esto es una descripción" />
  <link rel="stylesheet" href="archivo.css" type="text/css">
</head>
```

Esto permite usar el mismo estilo para varias páginas HTML distintas.

Ejemplo de un fichero externo de CSS (Estilos.css)

Estas propiedades generales afectaran a todas las paginas web que implementen el fichero estilos.css. Las propiedades van entre llaves y delante de la llave está la etiqueta HTML que afectará.

```

/*Etiquetas generales*/
body {
    background-color : #006600; /* Color del fondo*/
    font-family : arial; /*Tipo de letra: Arial*/
    color : White; /*Color para el texto blanco*/
    margin:10px 20px 10px 20px; /*márgenes para la web*/
}

p {
    font-size : 12px; /*Tamaño de letra: 12 pixeles*/
    font-family : arial,helvetica; /*Tipo de letra: Arial, helvetica */
    font-weight : normal; /* Estilo de letra: normal*/
}

h1 {
    font-size : 36pt; /*Tamaño de letra: 36 puntos*/
    font-weight : bold; /* Estilo de letra: negrita*/
    font-family : verdana,arial; /*Tipo de letra: Verdana, Arial */
    text-decoration : underline; /* Decoración del texto: subrayado/
    text-align : center; /* alineación del texto: centro/
    background-color : white; /* Color de fondo*/
    border-style: solid; /* Estilo del borde: solido*/
    border-color: blue; /* Color del borde: azul*/
    border-width: 2px; /* Ancho del borde: 2 pixeles*/
    padding: 10px; /* Espacios entre el borde y el elemento -> 10
pixeles*/
}

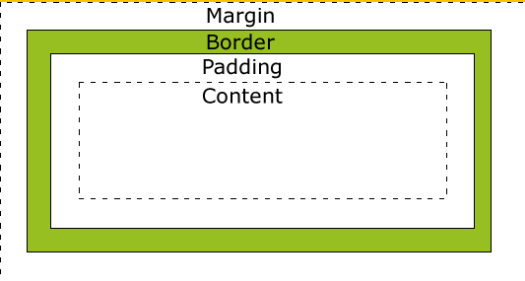
/*Clases-se utilizaría dentro de la etiqueta HTML con
class="nombredelaclase" */

.nombredelaclase{color:#345334}

/*Selectores-se utilizaría dentro de la etiqueta HTML con
id="nombredelselector" */

#nombredelselector {color:red}

```

Diferencias entre Margin y padding	Unidades absolutas	Unidades Relativas
	<p>El milímetro, mm El centímetro, cm El pixel, px El punto, pt La pica, pc</p>	<p>El porcentaje, %. (Se define respecto del ancho disponible. Por ejemplo el ancho del contenedor que atrapa una capa. Es común que a esta capa de "orden superior" se le aplique un ancho fijo.)</p> <p>em (la unidad em es equivalente al tamaño de fuente del elemento base, o del padre. Por ejemplo, si el tamaño de la tipografía del elemento padre es de 25px, entonces el valor 2em corresponderá a 50px.</p> <p>rem (es similar a em, pero en este caso el valor siempre es el mismo, ya que parte del tamaño de la tipografía del elemento root)</p>

■ Definir secciones reducidas o

Para definir estilos en secciones reducidas de una página se utiliza la etiqueta ****. Con su atributo **class o style** indicamos en sintaxis CSS las características de estilos. Lo vemos con un ejemplo, pondremos un párrafo en el que determinadas palabras las vamos a visualizar en color verde.

Podemos utilizar el atributo style e indicar directamente las propiedades **CSS (NO RECOMENDADO)**:

```
<p>
  Esto es un párrafo en varias palabras <SPAN style="color:green">en color
  verde</SPAN>. Resulta muy fácil.
</p>
```

Podemos utilizar el atributo **class** y utilizar una clase declarada previamente. Los programas como **Dreamweaver** utiliza esta opción. La ventaja que tiene es que la una vez declarada la podemos reutilizar en otros elementos de nuestra página Web

```
<p>
  Esto es un párrafo en varias palabras <SPAN class="NombredeClase">en color
  verde</SPAN>. Resulta muy fácil.
</p>
```

■ Estilos definidos en una parte de la página <DIV>

Con la etiqueta **<DIV>** podemos definir secciones de una página y aplicarle estilos con el atributo style, es decir, podemos definir estilos de una vez a todo un bloque de la página. También podemos usar una clase junto a la etiqueta **div**.

```
<div style="color:#000099; font-weight:bold">
  <p> Estas etiquetas van en <em> azul y negrita</em> </p>
  <p> Seguimos dentro del DIV, luego permanecen los estilos </p>
</div>
```

Recomendada

```
<div id="nombreclase">
  <p> Estas etiquetas van en <em> azul y negrita</em> </p>
  <p> Seguimos dentro del DIV, luego permanecen los estilos </p>
</div>
```

■ Clases/Selectores y Subclases en CSS

Una clase en CSS es una forma de agrupar propiedades que pueden posteriormente ser aplicadas a un elemento específico usando su atributo **"class"**. Estos grupos reciben nombres y pueden ser definidos para elementos específicos o para todos ellos. Para definir una clase los autores deben escribir el elemento para el cual es declarada, seguido de un punto y del nombre de la clase. El bloque de propiedades es encerrado por corchetes como en los ejemplos anteriores. Las clases se suelen utilizar mucho en programas como **dreamweaver**.

La principal diferencia entre este tipo de selector y el selector de clase tiene que ver con HTML y no con CSS. En una misma página, el valor del atributo **id** debe ser único, de forma que dos elementos diferentes no pueden tener el mismo valor de **id**. Sin embargo, el atributo **class** no es obligatorio que sea único, de forma que muchos elementos HTML diferentes pueden compartir el mismo valor para su atributo **class**. De esta forma, la recomendación general es la de utilizar el selector de ID cuando se quiere aplicar un estilo a un solo elemento específico de la página y utilizar el selector de clase cuando se quiere aplicar un estilo a varios elementos diferentes de la página HTML.

Estas clases están declaradas en nuestro fichero externo (estilos.css)

```
p.importante { color: red; } /* notación más correcta de la clase si siempre se utiliza en párrafo*/
.grande { font-size: 12px; }
#destacado { color: red; }
div#aviso { color: blue; } /* notación más correcta del identificador si siempre se utiliza en párrafo*/
```

La **clase grande** se puede utilizar con cualquier etiqueta y la **clase importante** es una subclase de la etiquetas **p** y solo se podrá utilizar en esa etiqueta.

El selector **aviso** y **destacado** solamente se pueden utilizar una vez por página web html y **aviso** solo podrá ir en un párrafo

En nuestro fichero HTML se utilizarían de

```
<p class="importante">Importante</p>
<p class="grande">Grande</p>
<div id="destacado">Destacado</div>
<div id="aviso">Aviso</div>
```

Un uso muy común de los id son con los div (solamente se utiliza esa vez)

En nuestro fichero CSS se utilizarían de

```
div#azulnegrita {.....} /*aplicar estilo a toda la capa*/
div#azulnegrita p {.....} /* es una manera formal de declarar estilos a los párrafos que
están dentro de la capa azulnegrita sin utilizar clase para los párrafos*/
```

En nuestro fichero HTML se utilizarían de

```
<div id="azulnegrita">
  <p> Estas etiquetas van en <em> azul y negrita</em> </p>
  <p> Seguimos dentro del DIV, luego permanecen los estilos </p>
</div>
```

■ Validar una página de CSS

CSS es otro lenguaje, y si no lo conoces puedes equivocarte al escribirlo. El validador de CSS (<http://jigsaw.w3.org/css-validator/>) te dirá qué errores le ve a tu código, y si no le ve errores es que es correcto, porque también son ellos quienes se encargan del CSS.

5

Cabeceras y Texto

El HTML nos ofrece seis etiquetas distintas para mostrar títulos y subtítulo (Cabeceras de los documentos).

Etiqueta	Definición
<code><h1>.....</h1></code> <code><h2>.....</h2></code> <code><h3>.....</h3></code> <code><h4>.....</h4></code> <code><h5>.....</h5></code> <code><h6>.....</h6></code>	Sirven para definir la estructura del texto, y poder poner títulos de sección, subsecciones, etc. Los buscadores las usan para saber de qué se habla en una página, así que es importante usarlas sólo para lo que se han diseñado. No utilizarlas para aumentar o disminuir el tamaño de las letras para eso tenemos los estilos (CSS)

Existen muchas etiquetas en HTML para **formatear un texto**. A continuación mostraremos solo las imprescindibles para formatear el texto con ayuda de las hojas de estilo:

Etiqueta	Definición
<code><p>.....</p></code>	Establece un párrafo con sus correspondientes saltos de líneas.
<code>.....</code>	Resalta una palabra o frase (cursiva)
<code>.....</code>	Resalta más una palabra (Negrita)

Otros elementos de utilidad que no son considerados texto:

Etiqueta	Definición	Atributo
<code><hr/></code>	Barra horizontal utilizada para separar secciones.	* width="tamaño porcentajes o pixeles" (Utiliza css en lugar de este atributo)
<code>
</code>	Salto de línea	
<code><!-- ... --></code>	Introduce comentarios en nuestro código. Todo texto introducido en las etiquetas no sera interpretado por el navegador	

6

Listas

Las listas son los elementos con los más se puede jugar al aprender CSS. Sólo con HTML son muy aburridas, pero añadiendo pocos estilos se pueden hacer maravillas como menús de navegación, botones, pestañas, tablas, y muchas más cosas.

En las páginas web abundan muchas listas, aunque no te hayas dado cuenta. Son una forma muy buena de estructurar la información, aparte de ser sencillas y de verse bien en todos los navegadores.

Etiqueta	Definición
<pre> Primer elemento Segundo elemento </pre>	Lista desordenada: <ul style="list-style-type: none"> Primer elemento Segundo elemento
<pre> Primer elemento Segundo elemento </pre>	Lista ordenada: <ol style="list-style-type: none"> Primer elemento Segundo elemento
<pre><dl> <dt>Primer elemento</dt> <dd>Definición primer elemento</dd> <dt>Segundo elemento</dt> <dd>Definición segundo elemento</dd> <dt>Tercer elemento</dt> <dd>Definición tercer elemento</dd> </dl></pre>	Lista definición: <pre>Primer elemento Definición primer elemento Segundo elemento Definición segundo elemento Tercer elemento Definición tercer elemento</pre>

7

Enlaces

El poder verdadero de html radica en la capacidad de manejar hipertexto como algunos le llaman, y se logra por medio de enlaces o links. Esto es, a través de un click en un texto o una imagen, es posible encontrar más información relacionada con la que originó ese click. Dicha información puede encontrarse en otras páginas dentro y fuera de nuestro servidor, o en algún punto concreto de páginas dentro y fuera de nuestro servidor.

Sintaxis general

La sintaxis queda entonces de la siguiente forma:

contenido

Siendo el contenido un **texto o una imagen**. Por su parte, destino será una página, un correo electrónico o un archivo. En función del destino los enlaces son agrupados del siguiente modo:

Enlaces internos: los que se dirigen a otras partes dentro de la misma página.

Enlaces locales: los que se dirigen a otras páginas del mismo sitio web.

Enlaces remotos: los dirigidos hacia páginas de otros sitios web.

Enlaces con direcciones de correo: para crear un mensaje de correo dirigido a una dirección.

Enlaces con archivos: para que los usuarios puedan hacer download de ficheros.

Atributos

✓ **target="_blank"**

Este atributo sirve para que la página se abra en otra hoja del navegador. Si no lo pones se abrirá en la misma página siempre y cuando el usuario no haya elegido otro sitio con el ratón.

CSS en los enlaces

Los enlaces tienen algunas diferencias en las hojas de estilo con respecto a los otros elementos que se han estudiado. Las posibles opciones que dispone un enlace son:

Estilos.css (el orden de colocación es importante)

```
a:link {
    /* Aplicar estilo al enlace*/
}
a:visited {
    /* Aplicar estilo al enlace cuando ya ha sido visitado o pulsado por el usuario*/
}
a:hover {
    /* Aplicar estilo al enlace cuando el cursor pasa por encima*/
}
```

Ejemplos

→ Enlace a una página local:

`.....`

→ Enlace a una web remota (y se abrirá en otra página obligatoriamente):

`.....`

→ Enlace a una dirección de correo:

`.....`

→ Enlace a un fichero pdf (que está en la carpeta documentos):

`.....`

→ Enlace interno a otra parte de la misma página:

`Ir abajo`

Una vez pulsado el enlace saltará al trozo de página que
esté escrita la siguiente etiqueta: `.....`

8

Imágenes

En Internet se usan básicamente dos: **GIF** y **JPG**. Ambos formatos comprimen las imágenes para reducir su tamaño, de este modo se asegura una transferencia más rápida por la red. Esto es importante, ya que si la imagen se tarda en cargar, es posible que el visitante de nuestra página la abandone por ello.

X El **formato GIF** usa 256 colores y se emplea sobretodo con imágenes pequeñas como iconos. No se suele usar con fotos porque da más calidad el formato **JPG** a pesar de que no tenga ningún tipo de compresión. Posee varias características importantes y es que **permiten pequeñas animaciones y transparencias**.

X El **formato JPG** usa 16.7 millones de colores, por lo que se emplea con imágenes de alta resolución. Con este formato se obtiene un grado de compresión más alto que con el GIF.

X El **formato PNG**: El PNG es un formato gráfico cada vez más usado en lugar de GIF. Se muestra correctamente en los navegadores, su uso está libre de derechos y permite una alta compresión así como una reproducción progresiva de imágenes con hasta 16,7 millones de colores. También **permite transparencias**.

Sintaxis general

La etiqueta que utilizaremos para insertar una imagen es `` (image). Esta etiqueta no posee su cierre correspondiente y en ella hemos de especificar obligatoriamente el paradero de nuestro archivo gráfico mediante el atributo `src` (source).

La sintaxis queda entonces de la siguiente forma:

```

```

Atributos (Los atributos con asterisco (*) se usan cada vez menos siendo sustituidos por CSS)

✓ **alt="descripción del fichero"**

Sirve para describir a la imagen brevemente. Es muy recomendable ponerlo en todas las imágenes por si el navegador no puede cargar la imagen que aparezca su descripción

✓ ***height="tamaño en píxeles" y *width="tamaño en píxeles"**

Definen la altura y anchura de nuestra foto. Es muy recomendable indicar las dimensiones de nuestra imagen al navegador para ayudarle a confeccionar la página.

Ejemplos

Vamos a mostrar una imagen que tenemos en la carpeta imágenes con tamaño (120 x100), sin borde y con una breve descripción

Recomendada: Con CSS

```

```

En este ejemplo vamos a convertir una imagen en un enlace a la página google.

```
<a href="http://www.google.es">  </a>
```

CSS en las imágenes

Las imágenes puede alinearse junto al texto utilizando una clase generada en el fichero de estilos, a continuación tienes algunos ejemplos

Estilos.css

En este ejemplo alineamos una imagen a la derecha y el texto se coloca a la izquierda e introducimos una separación entre la imagen y el texto

```
.alinear {  
    float: left;  
    padding: 4px; /*Relleno para situar varios espacios entre la imagen y el texto*/  
    margin: 0 7px 2px 0; /* Margen para las fotos*/  
    display: inline; /*El texto se sitúa en línea con la imagen*/  
}
```

Como centrar una imagen

El más correcto sería hacer que la etiqueta img sea tratada como elemento de bloque y darle a la propiedad margin un valor de auto, de modo que el navegador asignará márgenes proporcionales en referencia a la caja padre que contenga a la imagen:

Primer método(no funciona en Internet Explorer):

```
img {  
    display: block;  
    margin: auto;  
}
```

Sin embargo, esto no funciona en un navegador (a ver si lo adivinan...), en IE5.x, que a día de hoy sigue siendo muy usado, por lo que, aunque menos correcta, y siguiendo la argumentación del funcionamiento del atributo text-align, si se le asigna el valor center a una etiqueta que sea contenedora de la imagen, esta se centrará, ya que, como decíamos, este valor afecta al contenido, no a la etiqueta a la que se le asigne, así que, ya sea con etiquetas div o p (siempre elementos de bloque), creando una clase se conseguiría este efecto:

Segundo método (Funciona en todos los navegadores)

```
.centrar-imagen {  
text-align: center; }
```

Y luego lo aplicamos así en el HTML:

```
<div class="centrar-imagen"></div>
```


9

Tablas

Una tabla en un conjunto de celdas organizadas dentro de las cuales podemos alojar distintos contenidos. Las tablas deben utilizarse solamente para mostrar datos de forma organizada y tabulada nunca para organizar y maquetar los elementos de una página web (como antiguamente se hacía). A través de las hojas de estilo podemos conseguir tablas muy atractivas sin utilizar los engorrosos atributos (bgcolor, align, border,)

Sintaxis general

Para introducir una tabla en Dreamweaver (**Insertar/Tabla**), elije el número de filas y columnas que tendrá la tabla y el tipo de encabezado.

Las tablas son definidas por las etiquetas **<table>** y **</table>**. Dentro de estas dos etiquetas colocaremos las otras etiquetas:

- > La etiqueta **<caption>.... </caption>** sirve para introducir un título en la tabla que describe el contenido de la misma. Es conveniente que siempre utilices esta etiqueta.
- > Las tablas son descritas por líneas de izquierda a derecha. Cada una de estas filas es definida por otra etiqueta y su cierre: **<tr>** y **</tr>**
- > Asimismo, dentro de cada línea, habrá diferentes celdas. Cada una de estas celdas será definida por otro par de etiquetas: **<td>** y **</td>**. Dentro de estas etiquetas será donde coloquemos nuestro contenido.
- > Las etiquetas **<th>** **</th>** tienen la misma función de definir la celda dentro de una fila (igual que **<td>**) pero con la diferencia que se utilizan para los encabezados de la tabla que suelen estar destacados con otros colores. Por medio de los estilos podemos definir a nuestro gusto.

Aquí tenéis un ejemplo de estructura de tabla:

```
<table class="ensayo" summary="Tabla ejemplo" >
<caption>Título de tabla</caption>
<tbody>
  <tr class="principal">
    <th></th>
    <th>Encabezado </th>
    <th>Encabezado</th>
    <th>Encabezado</th>
  </tr>
  <tr class="etapas">
    <th colspan="4"> Encabezado 1</th>
  </tr>
  <tr >
    <th> Primera</th>
    <td>Celda 1, línea 1</td>
    <td colspan="2"> Celda 2, línea 1</td>
  </tr>
  <tr>
    <th>Segunda</th>
    <td> Celda 1, línea 2</td>
    <td> Celda 2, línea 2</td>
    <td> Celda 3, línea 2</td>
  </tr>
  <tr>
    <th> Tercera </th>
    <td> Celda 1, línea 3</td>
    <td> Celda 2, línea 3</td>
    <td> Celda 3, línea 3</td>
  </tr>
</tbody>
</table>
```

Título de tabla

Encabezado		Encabezado	Encabezado
Encabezado 1			
Primera	Celda 1, linea 1	Celda 2, linea 1	
Segunda	Celda 1, linea 2	Celda 2, linea 2	Celda 3, linea 2
Tercera	Celda 1, linea 3	Celda 2, linea 3	Celda 3, linea 3

Atributos (Los atributos con asterisco (*) son obsoletos y están siendo sustituidos por CSS)

✓ ***cellspacing="número"**

Sirve para describir el espacio que hay entre el borde de las celdas y el de la tabla. Suele poner siempre valor 0.

✓ ***cellpadding="número"**

Sirve para describir el espacio que hay entre el borde de las celdas y el contenido de la celda. Se suele poner siempre valor 0 o no ponerlo para poder modificarlo después a través de CSS.

✓ **colspan="número celdas"**

Sirve para combinar celdas horizontalmente.

✓ **rowspan="número celdas"**

Sirve para combinar celdas verticalmente.

CSS en los enlaces

Las tablas son elementos muy atractivos para las hojas de estilos. Aquí teneis un ejemplo para aplicar colores de celdas, bordes, espaciados, etc.....

Estilos.css

```
/*Utilizamos estilo para el título de la tabla*/

table {
    border: 1px solid #cccccc;
    width: 500px;
    margin: 20px auto;
    border-collapse: collapse; /*celdas no separadas*/
    /*Si queremos separación entre celdas horizontal y vertical
    border-collapse: separate
    border-spacing: 5px 10px;*/
}
caption {
    font: bold 25pt Tahoma, Arial, sans-serif;
    color: #545498;
    text-align: center;
    margin: 10px auto;
}
tr.principal th {
    font: bold 15px Tahoma, Arial, sans-serif;
```

```
        color:white;
        background: #5FACF3;
        text-align: left;
        padding: 5px;
        border: 0;
    }
tr.etapas th{
    font: bold 15px Tahoma, Arial, sans-serif;
    color: #ffffff;
    background: #DC4503;
    text-align: left;
    padding: 5px;
    border-top: 1px solid #eee;
}
th {
    font: normal 15pt Tahoma, Arial, sans-serif;
    color:#ffffff;
    background-color: #F5CA64;
    text-align: left;
    padding: 5px;
    border-top: 1px solid #eee;
}

td {
    font: 15px Tahoma, Arial, sans-serif;
    color: #666666;
    background: #ffffff;
    text-align: left;
    padding: 5px;
    border-top: 1px dotted #eeeeee;
}
```

10

Maquetar una página con capas

Antes de comenzar con el CSS quiero exponer aquí la importancia de usar los divs, un div es básicamente un contenedor, en el podemos meter cualquier clase de contenido, con la gran ventaja de que los podemos manipular a nuestro antojo con CSS, colocarlo en cualquier parte de la página, de cualquier tamaño, de cualquier color, con bordes o sin ellos, con imágenes de fondo o sin ellas, a diferencia de las tablas, los divs no se dividen por dentro, pero podemos anidarlos y organizarlas casi como las tablas (en caso de ser necesario). En HTML 4 la página se maquetaban a través de div en HTML5 se utilizan nuevas etiquetas que realizan la misma función que se utilizan para la indexación por los buscadores

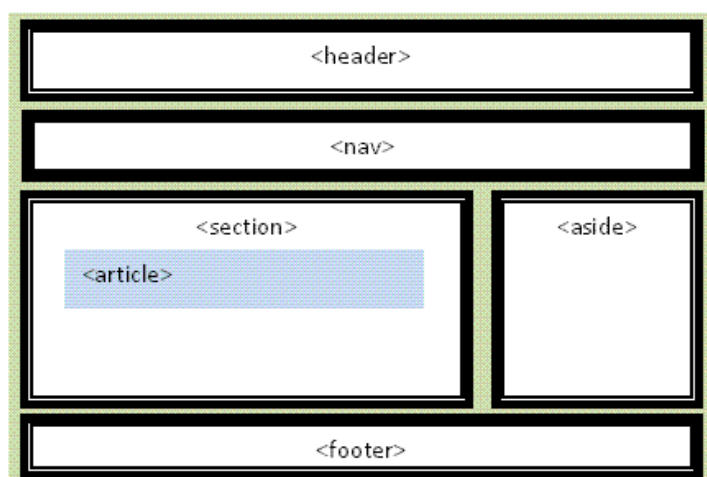
Estas son las nuevas etiquetas de HTML5:

- ✚ **<header></header>**: Para la presentación o cabecera de la página.
- ✚ **<nav></nav>**: Para menús de navegación con enlaces a otras páginas del sitio o a otras webs.
- ✚ **<footer></footer>**: Para el pie de página o de una sección. Normalmente se incorpora aquí información como el autor, el copyright, la fecha, etc.
- ✚ **<section></section>**: Para la presentación de una sección general dentro de un documento. Normalmente incluimos en esta etiqueta el cuerpo principal de la página.
- ✚ **<article></article>**: Para incluir artículos o subsecciones en otras partes de la página. Representa un componente autónomo dentro de la página, que puede repetirse. Por ejemplo en una web de venta de libros, cada uno de los libros que se muestran.
- ✚ **<aside></aside>**: Representa un contenido de la página distinto del que lo rodea. Es por lo tanto independiente del resto. Puede ponerse aquí contenido publicitario, otros elementos de navegación o contenido que se considera separado del contenido principal de la página.

Nota: Si alguna etiqueta no encaja con su función puedes utilizar <div id=...>. Por ejemplo section y article se usaría en un blog o un periódico.

Cuando vamos a plantearnos la estructura de una página Web nos tenemos que plantear muchas cosas como por ejemplo:

- ¿Cuántas partes vamos a dividir nuestra página? Encabezado, pie de página, menú navegación.....
- Diseño que vamos a emplear: colores, tipos de letras, imágenes,



Vamos a utilizar la estructura de la foto como ejemplo para ver como la diseñamos a través de capas

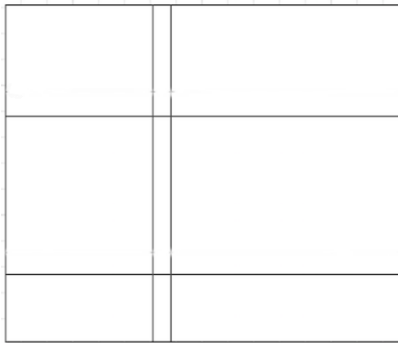
Ejemplo



Estructura de HTML 5:

Todas las capas tienen que estar dentro de una capa que las contenga a todas que le vamos a llamar **contenedor**. Dentro de una capa puede contener más capas.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <title>Grid Layout</title>
  <link rel="stylesheet" href="cssgrid.css">
</head>
<body>
  <div id="container">
    <header>
      <h2>Cabecera</h2>
    </header>
    <nav>
      <h2>Menu</h2>
      <ul>
        <li>Opcion 1</li>
        <li>Opcion 2</li>
        <li>Opcion 3</li>
        <li>Opcion 4</li>
      </ul>
    </nav>
    <section>
      <h2>Main</h2>
      <p>Este es el contenido principal. Lorem ipsum</p>
    </section>
    <footer>
      <h2>Footer</h2>
    </footer>
  </div>
</body>
</html>
```



Imaginemos un Grid compuesto por 3 columnas y 3 filas. Usaremos la columna como espaciador.

```
body {
    margin: 0px auto;
}
div#container {
    display: grid;
    /* Grid de 3x3 */
    /* 3 columnas: 1a de 200px, 2a como margen (1%), 3a ocupa el espacio restante */
    grid-template-columns: 200px 1% 1fr;
    /* grid-rows: 3 filas ancho automático */
    grid-template-rows: auto auto auto;
    grid-template-areas: "header header header"
                        "menu margen cuerpo"
                        "pie pie pie";
}
header {
    grid-area: header;
    color: white;
    background-color: #0D47A1;
    padding: 1.2em;
}
Header h2 {color:red}
footer {
    grid-area: pie;
    background-color: #2196F3;
    padding: 1.2em;
}
section {
    grid-area: cuerpo;
    background-color: #BBDEFB;
    padding: 1.2em;
}
nav{
    grid-area: menu;
    background-color: #EEEEEE;
}
Nav h2{color:red}

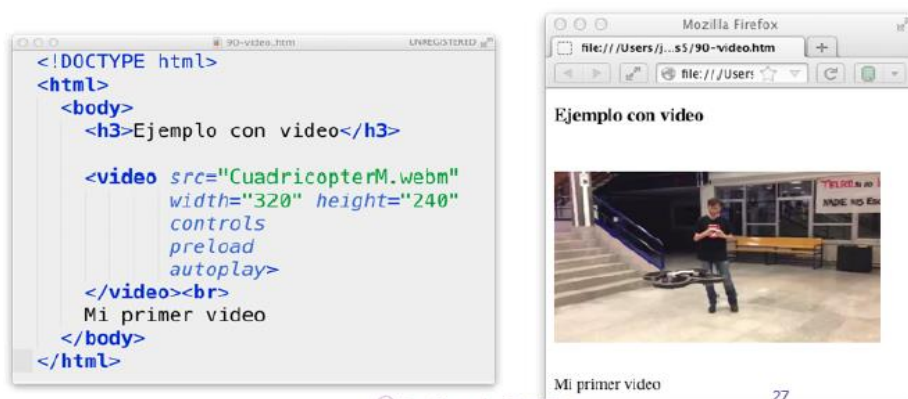
/* Web responsive */
/*Cuando el tamaño de pantalla sea menor de 400px (o como máximo 400px) le quitamos
la propiedad display: grid a la clase container que lo engloba todo, por display: block.*/
@media (max-width: 400px) {
    .container {
        display: block;
    }
}
```

11

Video, audio y canvas

Video con HTML5

La etiqueta **video** de **HTML5** está en principio destinada a reemplazar Flash como la aplicación que permitía la presentación de elementos animados en páginas web.



Sin embargo, esta nueva característica HTML5 tiene alcances más amplios. Además, su integración con los demás elementos de HTML5 así como con CSS y Java Script, y su creciente integración con los navegadores, presenta un panorama futuro muy positivo en cuanto a la sencillez y seguridad de la codificación.

Atributos principales de la etiqueta video de HTML5

Además de source, la etiqueta video puede condicionar su funcionamiento a través de otros atributos. Algunos de ellos son:

- *Width* y *height*, que establecen el tamaño del rectángulo donde se reproducirá el video. Si se omiten estos dos parámetros, el navegador asumirá las dimensiones, lo cual podría afectar a la distribución de los elementos dentro de nuestra página web.
- *Autoplay* es el atributo que le indica al navegador que inicie la reproducción una vez que el video haya sido cargado en la página.
- *Loop* es un atributo que informa al navegador utilizado que debe repetir la reproducción desde el principio cada vez que ésta concluya.
- *Controls* avisa al navegador que debe mostrar los controles necesarios para que el usuario interactúe con el video reproducido. Es la típica imagen gráfica con los controles *play*, *pause*, *stop*.

Sonido con HTML5

Tal como sucede con la nueva etiqueta *video*, HTML5 incluye una etiqueta *audio*, que maneja la posibilidad de reproducción de sonido en forma natural, sin necesidad de recurrir a un plug-in tal como Flash. Su codificación es, asimismo, muy sencilla y seguramente estamos frente a la presencia de un nuevo estándar.

```
1<audio controls>
2<source src="micancion.ogg" type="audio/ogg" />
3<source src="micancion.mp3" type="audio/mpeg" />
4</audio>
```

En este caso, el atributo *controls* se incluye para ordenar al navegador que presente un elemento

gráfico mediante el cual el usuario pueda manejar, como mínimo, las características de *play* y *pause* de su reproductor.

Los atributos *autoplay* y *loop* son similares a los utilizados para la etiqueta video.

```
<audio src="audio.ogg" controls autoplay loop>
<p>Tu navegador no implementa el elemento audio</p>
</audio>
```

Este código de ejemplo usa los atributos del elemento `<audio>`:

- `controls`: Muestra los controles estándar de HTML5 para audio en una página Web.
- `autoplay`: Hace que el audio se reproduzca automáticamente.
- `loop`: Hace que el audio se repita automáticamente.

Dibujo y gráficos: canvas

Un nuevo elemento HTML5, *canvas*, ofrece la posibilidad de producir diferentes variedades de gráficos, composiciones fotográficas, animaciones, a través de scripts que pueden ser programados en JavaScript o algún otro medio similar.

La utilización de *canvas* en todas sus posibilidades es sencilla, aunque requiere conocimiento de HTML y Java Script para lograr excelentes resultados. En principio, hay que tomar en cuenta que, como sucede con varias aplicaciones de HTML5, esta característica no es compatible aún con varios navegadores (puedes consultar caniuse para más detalles).

A través de *canvas* se pueden dibujar todo tipo de figuras geométricas, líneas, figuras e incluso complejos gráficos derivados de fórmulas matemáticas.

Como sencillos ejemplos, podemos citar los siguientes:

- `fillRect(x, y, ancho, alto)` representa los parámetros para el dibujo de un rectángulo.
- `strokeRect(x, y, ancho, alto)` sirve para dibujar un rectángulo sólido (relleno).

Un ejemplo algo más complejo:

- `Arc(x, y, radio, anguloInicial, anguloFinal, clockwise/anticlockwise)`: es la redacción que permite dibujar un arco, donde *x* e *y* son las coordenadas del centro del círculo; *radio* define la longitud de este elemento de la curva; los siguientes dos parámetros definen los ángulos inicial y final de la curva en radianes, y el último determina la dirección en el sentido de las agujas del reloj o en la orientación inversa.

Así, debemos internarnos en todas las posibilidades de *canvas* para lograr gráficos con la complejidad deseada y su animación, lo cual nos da muchas alternativas.

Ejemplo en HTML de mover una bandera con el ratón



```
<!DOCTYPE HTML>
<html>
<head>
<title>Introduccion a HTML5 Ejercicio 3 </title>
<style type="text/css">
.uno,.dos{
    float: left;
    width: 200px;
    height: 400px;
    border-bottom-style:solid;
    border-bottom:2px dashed #3ADF00;
    margin-top: 20px;
    margin-left: 100px}
</style>

<script type="text/javascript">
function allowDrop(ev)
{
    ev.preventDefault();
}

function drag(ev)
{
    ev.dataTransfer.setData("Text",ev.target.id);
}

function drop(ev)
{
    {
    var data=ev.dataTransfer.getData("Text");
    ev.target.appendChild(document.getElementById(data));
    ev.preventDefault();
    }
}

function ejerc3(){

    var bandera=document.getElementById('bandera');

    if (bandera && bandera.getContext) {
        var ctx=bandera.getContext ('2d');
        if (ctx){
            ctx.fillStyle = 'red';
            ctx.fillRect (4 ,0 ,150 ,25);
            ctx.fillStyle = 'yellow';
            ctx.fillRect (4 ,25 ,150 ,50);
            ctx.fillStyle = 'red';
            ctx.fillRect (4 ,75 ,150 ,25);
            ctx.fillStyle = 'grey';
```

```
        ctx.fillRect (0 ,0 ,3 ,400);

    }

}

</script>
</head>
<body onload="ejerc3()">

<div class="uno" ondrop="drop(event)" ondragover="allowDrop(event)">
    <canvas id="bandera" width="180" height="420" draggable="true" ondragstart="drag(event)" >
        <p> Este texto se muestra para los navegadores no compatibles con canvas.</p>
    </canvas>
</div>

<div class="dos" ondrop="drop(event)" ondragover="allowDrop(event)">
</div>
</body>
</html>
```

12

Formularios

El elemento HTML para definir un formulario Web se especifica con la etiqueta **<FORM>** y tiene este aspecto:

```
<FORM ACTION="url" METHOD="método de envío">  
  <INPUT> | <SELECT> | <TEXTAREA> | <BUTTON> | <DATALIST> | <OUTPUT>  
</FORM>
```

Estos son los **atributos** que encontramos en la etiqueta FORM:

ACTION: la URL de un programa que procesa la petición en el lado servidor.

METHOD: el método HTTP que el navegador utiliza para mandar los datos del formulario del cliente al servidor. Hay dos opciones:

- **post:** corresponde al método POST HTTP, por el cual los datos del formulario son incluidos en el cuerpo del formulario y son enviados al servidor. Los más habitual es elegir este método.
- **get:** corresponde al método GET HTTP, por el cual los datos del formulario son adjuntados a la URI del atributo *action* con un '?' como separador, y la URI resultante es enviada al servidor.

Los elementos del formulario

Los **elementos** que se pueden tener en un formulario son:

- **<INPUT>:** entrada o campo del formulario. Los tipos de input más utilizados son:
 - **text:** cuadro de texto.
 - **password:** cuadro de texto para introducir contraseña (no se ve contenido).
 - **hidden:** texto oculto, el navegador no lo visualiza en el documento Web, sin embargo si vemos su código podemos ver su contenido.
 - **checkbox:** casilla de verificación.
 - **radio:** casilla de selección.
 - **submit:** botón de envío del contenido del formulario al servidor.
 - **reset:** botón que facilita el borrado de los datos completados en el formulario.
 - Otros tipos que aparecieron con la versión de HTML5 son: *color, date, datetime-local, email, month, number, range, search, tel, time, url* y *week*.
- **<SELECT>:** lista de selección con múltiples valores.
- **<TEXTAREA>:** campo de texto multilínea.
- **<BUTTON>:** botón de acción.
- **<DATALIST>:** facilita un listado de opciones predeterminadas para un cuadro de texto.
- **<OUTPUT>:** facilita mostrar la salida a un cálculo realizado en el formulario.

Aprovechando las bondades del tag **LABEL** podemos mejorar tanto la accesibilidad como la usabilidad de los formularios en las interfaces web.

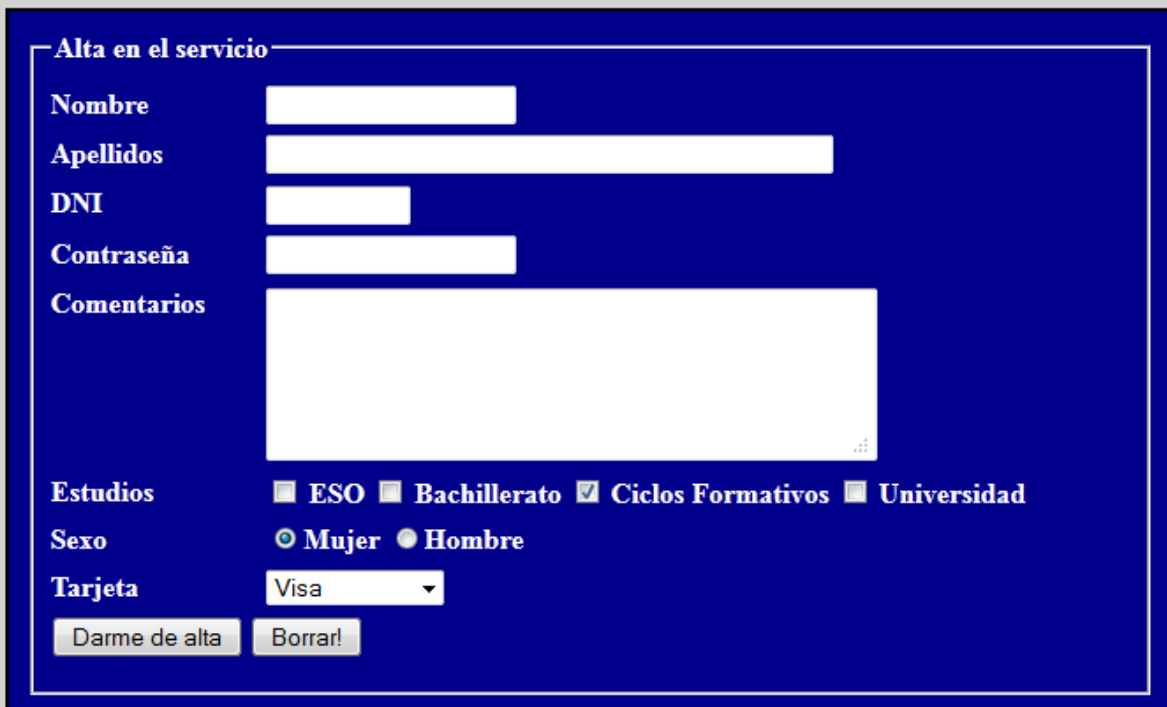
El uso es muy simple, basta con indicar el atributo `for` con el identificador al que se quiera asociar la etiqueta. Por lo tanto los atributos `for` (del LABEL) e `id` (de INPUT) han de ser iguales.

Al igual que la etiqueta FORM, cada uno de los elementos del formulario tiene sus propios atributos. A modo de ejemplo, estos son los más habituales del elemento `<INPUT>` cuadro de texto (*text*):

- **id**: identificador del elemento.
- **name**: nombre del elemento. Lo habitual es que tenga el mismo valor que el atributo "id".
- **type**: para indicar el tipo (text, password, hidden, etc).
- **required**: indica si es imprescindible poner información en el campo, se pone en el campo sólo la palabra "required" (también es posible escribiendo `required="true"`).
- **placeholder**: da información del dato que se espera introduzca el usuario (aparece en el cuadro de texto hasta que el usuario hace click en él).
- **size**: tamaño del cuadro de texto.
- **maxlength**: longitud máxima del texto.
- **value**: para inicializar con dicho valor el cuadro de texto.
- **Min/max**: Valor mínimo y máximo

Primer ejemplo

Estructura de HTML:



Alta en el servicio

Nombre

Apellidos

DNI

Contraseña

Comentarios

Estudios ☐ ESO ☐ Bachillerato ☒ Ciclos Formativos ☐ Universidad

Sexo ☒ Mujer ☐ Hombre

Tarjeta

```

<!DOCTYPE html>
<html>
<head>
    <title>Mi primer formulario</title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <link href="estilos.css" rel="stylesheet" type="text/css" />
</head>
<body>
<form id="formul1" name="formul1" method="post" action="pagina-destino.php">
<fieldset>
    <legend>Alta en el servicio</legend>
    <div>
        <label for="nombre">Nombre</label>
        <input type="text" id="nombre" name="nombre" />
    </div>
    <div>
        <label for="apellidos">Apellidos</label>
        <input type="text" id="apeliidos" name="apellidos" size="50" />
    </div>
    <div>
        <label for="dni">DNI</label>
        <input type="text" id="dni" name="dni" size="10" maxlength="9" />
    </div>
    <div>
        <label for="contrasena">Contraseña</label>
        <input type="password" id="contrasena" name="contrasena" />
    </div>
    <div>
        <label for="comentarios">Comentarios</label>
        <textarea cols="40" rows="5" id="comentarios" name="comentarios"></textarea>
    </div>
    <div >
        <label for="estudios">Estudios</label>
        <input id="estudios" name="estudios" type="checkbox"> ESO</input>
        <input id="estudios" name="estudios2" type="checkbox"> Bachillerato</input>
        <input id="estudios" name="estudios3" type="checkbox" checked=""> Ciclos Formativos</input>
        <input id="estudios" name="estudios4" type="checkbox"> Universidad</input>
    </div>
    <div >
        <label for="sexo">Sexo</label>
        <input id="sexo" name="sexo" value="mujer" type="radio" checked="">Mujer</input>
        <input id="sexo" name="sexo" value="hombre" type="radio">Hombre</input>
    </div>
    <div>
        <label for="tarjeta">Tarjeta</label>
        <select id="tarjeta" name="tarjeta">
            <option value="visa" selected>Visa</option>
            <option value="master">Master-Card</option>
        </select>
    </div>
    <div>
        <input type="submit" value="Darme de alta" />
        <input value="Borrar!" type="reset">
    </div>
</fieldset>
</form>
</body>
</html>

```

Estructura del fichero css

PROPIEDAD DISPLAY

`display` es la propiedad más importante para controlar estructuras. Cada elemento tiene un valor de `display` por defecto dependiendo de qué tipo de elemento sea. El valor por defecto para la mayoría de los elementos es usualmente `block` (de bloque) o `inline` (en línea). Un elemento que es `block` es comúnmente llamado elemento block-level. Un elemento `inline` siempre es llamado elemento `inline`.

✓ **Block**

`<div>` El `div` es el elemento block-level estándar. Un elemento block-level comienza en una nueva línea y se estira hasta la derecha e izquierda tan lejos como pueda. Otros elementos block-level muy comunes son `p` y `form`, y algunos nuevos en HTML5 son `header`, `footer` y `section`.

✓ **inline**

El `span` es el elemento `inline` estándar. Un elemento `inline` puede contener algo de texto dentro de un párrafo `` como esto `` sin interrumpir el flujo del párrafo. El elemento `a` es el elemento `inline` más común, ya que se usa para links.

✓ **none**

Otro valor común de `display` es `none`. Algunos elementos especializados como `script` usan este por defecto. Es comúnmente usado en JavaScript para ocultar o mostrar elementos sin eliminarlos ni recrearlos.

Esto es diferente de `visibility`. Usar `display: none` no dejará espacio donde el elemento se encontraba, pero `visibility: hidden`; dejará un espacio vacío.

```
body{
    margin: 20;
    font: Arial 14px;
    color: white;
    font-weight: bold;
    background-color: lightgrey;}

form{
    margin: auto;
    background: darkblue;
    border:solid black 2px;
    width: 600px; /*Tamaño fijo del formulario*/
    padding: 10px;}

div {
    display: block;
    clear: both; /* Todos los div son tipo bloques para que estén en una linea diferente*/
    margin: 10px 0 0 0;}

div label {
    display: inline;
    width: 20%; /*Tamaño de la etiqueta y posición izquierda el resto se ocupará por el INPUT*/
    float: left;
}
```

Ejemplo formulario (CSS Avanzado con validación HTML)

Formulario

Nombre

Email

Contraseña

Estudios

☐ ESO
☐ Bachillerato
☒ Ciclos
☐ Universidad

Sexo

☐ Hombre
☒ Mujer

Tarjeta

Visa ▼

Darme de alta

Borrar!

Fichero HTML

```
!DOCTYPE html>
<html>
<head>
    <title>Mi primer formulario</title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <link href="estilos.css" rel="stylesheet" type="text/css" />
</head>
<body>
<form name="formul1" id="formul1" method="post" action="pagina-destino.php"
<div class="encabezado">
<h1>Formulario</h1>
</div>
<div class="bloque">
    <label for="nombre">Nombre </label>
    <input type="text" name="nombre" id="name" maxlength="20" size="30" required="true" placeholder="Introducir
nombre"/>
</div>
<div class="bloque">
    <label for="email">Email</label>
    <input type="text" id="email" name="email" size="40" required="true" placeholder="Introducir Email"/>
</div>
<div class="bloque">
    <label for="contrasena">Contraseña </label>
    <input type="password" id="contrasena" name="contrasena" size="30" required="true" placeholder="Introducir contraseña"
/>
</div>
<div class="bloquecontenedor">
<div class="linea">
        <label for="estudios">Estudios</label>
        <input id="estudios" name="estudios" type="checkbox"> ESO</input>
        <input id="estudios" name="estudios2" type="checkbox"> Bachillerato</input>
        <input id="estudios" name="estudios3" type="checkbox" checked=""> Ciclos</input>
        <input id="estudios" name="estudios4" type="checkbox"> Universidad</input>
</div>
<div class="linea">
        <label for="sexo">Sexo</label>
```

```
<input id="sexo" name="sexo" name="sexo" value="hombre" type="radio">Hombre</input>
<input id="sexo" name="sexo" name="sexo" value="mujer" type="radio" checked="">Mujer</input>
</div>
<div class="linea">
    <label for="tarjeta">Tarjeta</label>
    <select id="tarjeta" name="tarjeta" >
        <option value="visa" selected>Visa</option>
        <option value="master">Master-Card</option>
    </select>
</div>
</div>
<div class="bloquebotones">
    <input type="submit" value="Dar de alta" />
    <input value="Borrar!" type="reset"/>
</div>
</form>
</body>
</html>
```

Fichero CSS

```
body{
    margin: 20px 0 0 0;
    font: Arial 14px;
    color: black;
    font-weight: bold;
    background-color: white;
}
form{
    border:solid 2px #708090;
    background:#dcdcdc;
    margin:0 auto;
    width:500px;
    padding:10px;
}

div.encabezado {
    text-align: center;
}

h1{margin-top:5px}

div.bloque{ /*Cada elemento del formulario va en una linea diferente*/
    display: block;
    clear: both;
    margin: 10px 0 0 0;
}

div.bloque label { /* Etiqueta tres primeras filas*/
    display:inline;
    font-weight:bold;
    text-align:left;
    width:20%;
    float:left;
}

div.bloque input{
    float:left;
    font-size:17px;
    border:solid 1px #708090;
    margin:2px 0 20px 10px;
```



```
}

div.bloquecontenedor{ /*Contenedor de las tres capas que están en linea*/
    display: block;
    clear: both;
    margin: 15px 0 0 0;
    height: 120px;
    width: 100%;
}
div.linea{
    display: inline;
    clear: none;
    width: 32%;
    float: left;
    margin: .4em 4px 0 0;
    height: 100%;
    border: dotted 1px darkgrey;
}
div.linea label{
    display: block;
    clear: both;
    border:dotted 1px darkgrey;
    margin-left:1px;
    margin-bottom: 10px;
    text-align: center;
    color: black;
    font-weight: bold;
}
div.linea input{
    display: inline;
    width: 35%;
    clear: both;
}
div.bloquebotones{ /*Botones de envio*/
    display: block;
    clear: both;
    text-align: center;
    margin: 15px 0 0 0;
}
div.bloquebotones input{
    background:black;
    color:white;
    font-weight: bold;
    width:145px;
    height:35px;
}
/* Opcional - Bordes Redondeados
Solo funciona con Firefox / Safari / Google Chrome */

    -moz-border-radius:10px;
    -webkit-border-radius:10px;
}
/* Hover a nuestro Botón
No funciona en IE6 (Uno de los peores navegadores) */
div.bloquebotones input:hover{
    background:#418eb6; color:#d0e8f7;
}
}
```