

LMSGI > UT02 > Apuntes

Tema 2. “Utilización de lenguajes de marcas en entornos web”

Actualizado al 11/10/2022

2.1.	Del HTML al XHTML.....	2
2.1.1.	¿Qué es HTML?	2
2.1.2.	Evolución	2
2.1.3.	Aparición de XHTML	2
2.2.	Estructura de un documento HTML.....	3
2.2.1.	Prólogo	3
2.2.2.	Ejemplar	4
2.3.	Identificación de etiquetas y atributos de HTML.....	5
2.3.1.	Clasificación de los atributos comunes según funcionalidad.....	6
2.3.2.	Elementos HTML	7
2.4.	XHTML frente a HTML.....	19
2.4.1.	XHTML: diferencias sintácticas y estructurales con HTML.....	19
2.4.2.	Ventajas e inconvenientes de XHTML sobre HTML	20
2.5.	Herramientas de diseño web	21
2.6.	Hojas de estilo o CSS	21
2.6.1.	Soporte de CSS en los navegadores	22
2.6.2.	Cómo incluir CSS en un documento HTML o XHTML	22
2.6.3.	Sintaxis de las reglas de estilo	26
2.6.4.	Atributos principales	27
2.6.5.	CSS de posicionamiento (CSS-P).....	37
2.6.6.	Unidades de tamaño	39
2.6.7.	Definición y uso de clases.....	40
2.6.8.	Definición y uso de identificadores	41

2.1. Del HTML al XHTML

2.1.1. ¿Qué es HTML?

HTML es el lenguaje utilizado para crear la mayor parte de las páginas web. Es un estándar reconocido en todos los navegadores, por lo tanto, todos ellos visualizan una página HTML de forma muy similar, independientemente del sistema operativo sobre el que se ejecutan.

2.1.2. Evolución

El origen de HTML fue un sistema de hipertexto para compartir documentos electrónicos en 1980. La primera propuesta oficial para convertir HTML en un estándar se realizó en 1993. Aunque ninguna de las dos propuestas de estándar que se hicieron (HTML y HTML+) consiguieron convertirse en estándar oficial.

- | | |
|-----------|---|
| HTML 2.0 | Primera versión oficial de HTML. El IETF lo publicó en septiembre de 1995. |
| HTML 3.2 | Publicado por el W3C el 14 de enero de 1997. Incorpora los applets de Java y texto alrededor de las imágenes. |
| HTML 4.0 | Se publicó el 24 de abril de 1998. Entre las novedades que presenta se encuentran las hojas de estilos CSS y la posibilidad de incluir pequeños programas en las páginas web. |
| HTML 4.01 | Se publicó el 24 de diciembre de 1999. Es una actualización de la versión anterior. En ese momento el W3C detuvo la actividad de estandarización de HTML hasta marzo de 2007, momento en que se retoma debido a la fuerza de las empresas del grupo WHATWG y a la publicación de borradores de HTML 5. |
| HTML 5.0 | Se publicó el 28 de octubre de 2014 por el W3C. En esta versión, se presentan nuevas características para ayudar a los creadores de aplicaciones web, mediante la introducción de nuevos elementos que respondan a las necesidades existentes y palien las carencias del estándar anterior en este aspecto. Ha sufrido dos revisiones posteriores (5.1 y 5.2), ahondando en la necesidad de facilitar webs responsivas, interactivas y, en general, más como aplicaciones que como meras contenedoras de información. |

2.1.3. Aparición de XHTML

Tras la publicación del estándar HTML 4.01 se detecta su incompatibilidad con herramientas basadas en XML. Para evitar estos problemas, se crea lenguaje XHTML que combina la sintaxis de HTML 4.0 con la de XML.

- XHTML 1.0 Fue la primera versión, se publicó el 26 de Enero de 2000. Es una adaptación de HTML 4.01 al lenguaje XML, por lo que mantiene sus características, y añade algunas restricciones y elementos de XML, para componer documentos HTML, pero “bien formados” desde el punto de vista de XML.
- XHTML 1.1 Esta nueva revisión pretende hacer modular la versión inicial de XHTML.
- XHTML 2.0 Rompe con la compatibilidad hacia atrás, generando una gran controversia.
- XHTML 5.0 Incorpora las novedades de HTML5. Pretende establecerse como estándar “en paralelo” con HTML5, pero lo cierto es que termina abandonado a favor de éste (HTML5) quien se erige, de facto, en el estándar que marca el presente y futuro de HTML.

2.2. Estructura de un documento HTML

2.2.1. Prólogo

La estructura de una página HTML ha de ser coherente con la que hemos visto en el tema anterior para cualquier documento XML. Por ello tendrá un prólogo y un ejemplar.

Todo documento HTML ha de tener una declaración del tipo de documento donde se le indica al navegador el tipo de documento que se va a iniciar y la versión de HTML utilizada para la codificación de este y, además, le permite interpretarlo correctamente.

Para la versión HTML 4.0, hay tres prólogos distintos que definen tres tipos de documentos:

HTML 4.0 Strict Es la DTD utilizada por defecto con HTML 4.0. En estos documentos no se permite el uso de los elementos declarados *deprecated* (desaprobados), de otras versiones o recomendaciones anteriores HTML. La declaración del tipo de documento correspondiente es:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 //EN"  
"http://www.w3.org/TR/REC-html40/strict.dtd">
```

HTML 4.0 Transitional Permite el uso de todos los elementos que permite el HTML 4.0 Strict, además de los elementos *deprecated*. Se entiende que era un estándar más permisivo, de cara a la migración a estándares más estrictos. La declaración del tipo de documento correspondiente es:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 transitional//EN"  
"http://www.w3.org/TR/REC-html40/loose.dtd">
```

HTML 4.0 Frameset Es una variante de HTML 4.0 Transitional para documentos que usan frames. En estos documentos el elemento body hay que reemplazarlo por un elemento frameset. La declaración del tipo de documento es:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN"  
"http://www.w3.org/TR/REC-html40/frameset.dtd">
```

Por su parte, HTML 5 simplifica mucho la declaración, quedando simplemente en:

```
<!DOCTYPE html>
```

2.2.2. Ejemplar

En un documento HTML está delimitado por las etiquetas <html> y </html>. El ejemplar puede, a su vez dividirse en dos partes:

2.2.2.1. La cabecera

Delimitada por las etiquetas <head> y </head>. Contiene la información sobre el título de la página, el autor, palabras clave, etc. Dentro de esta sección se define, por ejemplo, el título del documento, mediante las etiquetas <title> </title>.

Su información no se presentará en la ventana del navegador, salvo el título que aparecerá en la barra de título de la parte superior.

2.2.2.2. El cuerpo

Contiene la información que se va a presentar en pantalla. Está limitado por las etiquetas <body> y </body>, salvo en los documentos de tipo HTML 4.0 Frameset, que es <frameset />.

2.3. Identificación de etiquetas y atributos de HTML

Un documento HTML está conformado por etiquetas y atributos, así como por el contenido de los elementos delimitados entre su etiqueta de apertura hasta la de clausura. Por ejemplo:

```
<body>
  
  <p class="pie_imagen">
    Algo muy b&aacute;sico
  </p>
</body>
```

Aquí encontramos las etiquetas “body” (cuerpo de la página), “img” (imagen) y “p” (párrafo) y los atributos “src” (archivo imagen), “alt” (texto alternativo) y “class” (clase). El texto “Algo muy básico” (nótese que á indica la “á” con tilde) es el contenido del elemento “p”.

Y es que, al igual que en XML, algunos elementos pueden poseer etiquetas de apertura <etiqueta> y de cierre </etiqueta>. Pero hay dos grandes diferencias con XML:

- La primera es que algunas etiquetas no requieren cierre. Basta su presencia en su “versión de apertura” para indicar algo. Como por ejemplo
, que si aparece en medio de un texto debe interpretarse como un salto de nueva línea, así, sin más.
- La otra es que la cantidad de etiquetas de HTML está limitada a aquellas que están definidas por el lenguaje, no pudiendo utilizarse arbitrariamente las que se quieran.

Aunque HTML define una gran cantidad de etiquetas, estas no son suficientes para crear por completo las páginas, ya que es necesario aportar mucha información que no puede estar únicamente en la etiqueta (textos, imágenes, tamaños, colores, vínculos, etc.). Como no es posible crear una etiqueta por cada posibilidad diferente, se añade la información adicional a las etiquetas mediante atributos y también mediante el propio contenido entre la apertura y la clausura de la etiqueta. Al igual que en XML, llamamos “elemento” a la composición de etiqueta de apertura + atributos + contenido + etiqueta de clausura.

Para cada uno de los atributos hay definido un conjunto de valores que se le puede asignar. Si el valor de un atributo no es válido, el navegador simplemente lo ignora.

En las especificaciones del lenguaje HTML se definen, para cada etiqueta, qué atributos se puede utilizar y con qué valores, si bien algunos de ellos son comunes a muchas etiquetas.

2.3.1. Clasificación de los atributos comunes según funcionalidad

2.3.1.1. Atributos básicos

Se pueden usar en casi todas las etiquetas HTML.

Atributo	Descripción
<code>name = "texto"</code>	Permite asignar el nombre "texto" a un elemento HTML.
<code>title = "texto"</code>	Asigna un título a un elemento HTML, mejorando así la accesibilidad. Dicho título es mostrado por los navegadores cuando el usuario pasa el ratón por encima del elemento. Es especialmente útil con los elementos: a, link (enlace), img (imagen), object (objeto incrustado), abbr (abreviatura) y acronym (acrónimo).
<code>id = "identificador"</code>	Permite vincular al elemento HTML un identificador que debería ser único en la página. Es útil cuando se trabaja con CSS o Javascript. No pueden empezar por números y sólo puede contener letras, números y guiones medios o bajos.
<code>style = "estilo"</code>	Permite aplicar al elemento HTML el estilo "estilo" directamente. Este estilo describe su apariencia. La forma de usar estilos se verá más adelante.
<code>class = "clase"</code>	Permite agrupar al elemento dentro de una clase con nombre "clase". Las clases facilitan, entre otras cosas, asignar el mismo estilo a todos sus miembros a la vez. El nombre de la clase debe ser un identificador válido: no pueden empezar por números y sólo puede contener letras, números y guiones medios o bajos.

2.3.1.2. Atributos para internacionalización

Los utilizan las páginas que muestran sus contenidos en varios idiomas o aquellas que quieren indicar de forma explícita el idioma de sus contenidos

Atributo	Descripción																
dir = "ltr/rtl"	Indica la dirección del texto por lo que sólo puede tomar dos valores: <ul style="list-style-type: none">ltr (left to right) de izquierda a derecha. Es el valor por defecto.rtl (right to left) de derecha a izquierda.																
lang = "codigo"	<div>Especifica el idioma del elemento mediante un código predefinido. Los posibles valores se encuentran en el documento RFC 1766, algunos de ellos son:</div> <table><tr><th>Código</th><th>Idioma</th><th>Código</th><th>Idioma</th></tr><tr><td>en</td><td>Inglés (Gran Bretaña)</td><td>es</td><td>Español</td></tr><tr><td>en-US</td><td>Inglés americano</td><td>fr</td><td>Francés</td></tr><tr><td>ja</td><td>Japonés</td><td>fr-CA</td><td>Francés de Canadá</td></tr></table>	Código	Idioma	Código	Idioma	en	Inglés (Gran Bretaña)	es	Español	en-US	Inglés americano	fr	Francés	ja	Japonés	fr-CA	Francés de Canadá
Código	Idioma	Código	Idioma														
en	Inglés (Gran Bretaña)	es	Español														
en-US	Inglés americano	fr	Francés														
ja	Japonés	fr-CA	Francés de Canadá														
xml:lang = "codigo"	Al igual que antes, especifica el idioma del elemento mediante un código definido según la recomendación RFC 1766. En XHTML, el atributo xml:lang tiene más prioridad que lang y es obligatorio incluirlo siempre que se incluye el atributo lang.																

2.3.1.3. Atributos de eventos y atributos para elementos que pueden recibir foco

Se utilizan en las páginas web dinámicas que usan JavaScript. No los estudiaremos por ahora.

2.3.2. Elementos HTML

Un elemento HTML está formado por:

- Una etiqueta de apertura (obligatorio).
- Cero, uno o más atributos.
- Texto encerrado entre las etiquetas de apertura y cierre (opcional).
- Una etiqueta de cierre (no siempre se requiere).

Como ya se ha comentado, en HTML no todas las etiquetas pueden contener texto y/o etiqueta de cierre, esto depende de su cometido.

Según el modo en que ocupan el espacio disponible en la página los elementos pueden ser:

- **Elementos en línea**
Sólo ocupan el espacio necesario para mostrar su contenido. Pueden combinarse unos al lado de otros, dispuestos (por defecto) como texto de izquierda a derecha.

- **Elementos de bloque**

Los elementos de bloque siempre empiezan en una nueva línea y ocupan todo el ancho disponible hasta el margen derecho, aunque su contenido no llegue hasta allí. Dicho contenido puede ser texto, elementos en línea u otros elementos de bloque.

Existen elementos cuyo comportamiento puede ser en línea o de bloque, según las circunstancias. También es posible modificar el comportamiento predeterminado que tienen (lo más normal: hacer que un elemento de bloque se muestre en línea con otro).

El siguiente ejemplo muestra la diferencia entre ambos comportamientos:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>Ejemplo de elementos en línea y elementos de bloque</title>
  </head>
  <body>
    <h1>Los encabezados son elementos de bloque</h1>
    <p>Los párrafos también lo son.</p>
    <p>
      Sin embargo, los <a href="https://www.cifpcarlos3.es">enlaces</a>, o los
      <span style="font-weight: bold;">span</span> son elementos en línea, por eso
      todo esto se verá en una misma línea, sin saltos.
    </p>
  </body>
</html>
```

Puede verse, en la cabecera <head>, el empleo de un elemento <meta> (metadato) cuyo atributo “charset” permite definir la codificación del documento de texto. Por su parte, el atributo “lang” de la raíz define el lenguaje de la página. Esto forma parte de HTML 5.

Al publicarlo en un navegador como Firefox tendríamos:



2.3.2.1. Elementos de la estructura básica del documento

Recordamos aquí que la estructura básica de un documento viene determinada por las siguientes etiquetas:

Elemento	Descripción
<html>	Documento HTML. Es el elemento raíz.
<head>	Cabecera del documento. Se ubica en primer lugar, dentro de <html>.
<body>	Cuerpo del documento. Ubicado en el mismo nivel de <head>, detrás de ella.

En el punto anterior ya vimos un documento HTML básico que utiliza estos elementos.

2.3.2.2. Elementos de la sección de cabecera

Elementos contenedores

Son contenedores los elementos que poseen contenido dentro de sí.

Elemento	Descripción
<title>	Título del documento. Generalmente, su contenido se muestra en la barra de título del navegador, aunque esto depende del uso que haga cada navegador de este dato.
<script>	Permite incrustar un script (programa del lado del cliente) o bien hacer referencia a un fichero que lo contiene. En http://www.w3schools.com/tags/tag_script.asp vemos este ejemplo: <pre><script> document.getElementById("demo").innerHTML = "Hello JavaScript!"; </script></pre>
<style>	Permite definir el estilo aplicado al documento, utilizando hojas de estilo CSS. La hoja de estilo irá "incrustada" entre las etiquetas <style> y </style>. Si se desea que vaya en un fichero aparte, debe usarse la etiqueta <link> en su lugar. En http://www.w3schools.com/tags/tag_style.asp puede verse un ejemplo y, con el botón "Try it yourself", se puede probar a cambiar la definición de los estilos y ver el resultado final.

Elementos no contenedores

Se limitan a complementar la información sobre la página o su comportamiento por defecto.

Elemento	Descripción
<base>	<p>Define la URI base del documento. Posee dos atributos principales:</p> <ul style="list-style-type: none"> • <code>href="url"</code>, con dicha URL, la cual puede aplicarse por defecto a todos los enlaces. • <code>target="target"</code>, con la forma por defecto en la que se abren los enlaces. Sus posibles valores son (pueden consultarse en http://www.w3schools.com/tags/att_a_target.asp): <ul style="list-style-type: none"> • <code>_blank</code> : en una página en blanco. • <code>_self</code> : en el mismo marco/ventana. • <code>_parent</code> : en el marco/ventana padre, que contiene al actual. • <code>_top</code> : en el cuerpo completo de la ventana. • <code>nombre</code> : al indicar un nombre, se indica el del marco "destino" en el que abrir.
<link>	<p>Permite enlazar a ficheros externos en los que apoyarse: hojas de estilo, librerías... Por ejemplo, aquí la vinculación a una hoja de estilos externa, en un fichero "theme.css", que se alojaría en la misma URL en la que estuviera la página html:</p> <pre><head> <link rel="stylesheet" type="text/css" href="theme.css"> </head></pre>
<meta>	<p>Aporta información sobre la página (metadatos). Uno de sus principales usos es facilitar a los buscadores que el documento sea indexado a través de términos de búsqueda clave.</p> <p>La etiqueta tiene dos atributos principales:</p> <ul style="list-style-type: none"> • <code>name</code> : que indica el nombre del metadato • <code>content</code> : que da el valor al metadato <p>En este ejemplo (http://www.w3schools.com/tags/tag_meta.asp) puede verse como indicar la codificación de la página, la descripción, las palabras clave relacionadas con ella y el autor:</p> <pre><head> <meta charset="UTF-8"> <meta name="description" content="Free Web tutorials"> <meta name="keywords" content="HTML,CSS,XML,JavaScript"> <meta name="author" content="Hege Refsnes"> </head></pre>

2.3.2.3. Elementos que dan estructura a un documento

Sirven para delimitar diferentes partes del documento. En lugar de ver un documento HTML como un bloque monolítico de contenido, gracias a estas etiquetas podemos crear partes que se vuelven a descomponer en otras partes, de una forma jerárquica, para poder abordar cada división de una forma más simple y recomponer después el documento completo como una composición de dichas partes.

IMPORTANTE: Casi todos los elementos que vamos a ver a continuación son elementos que, de forma predeterminada, se comportan como elementos de disposición EN BLOQUE.

Elemento	Descripción
<div>	Delimita una parte del documento (capa), que a su vez se puede descomponer en otras partes (divs). Se puede decir que el cuerpo (body) de un documento HTML se descompone jerárquicamente en capas cada vez más pequeñas, hasta llegar a un elemento visual (imagen, párrafo, vídeo, etc.)
	Delimita una parte dentro de un párrafo, sin comenzar en un párrafo nuevo, para destacar visualmente o bien simplemente identificar dicha parte del resto del párrafo. Este es el único elemento que se comporta EN LÍNEA de forma predeterminada.
<p>	Delimita párrafos, generalmente de texto, si bien pueden contener otro tipo de elementos visuales como parte de dicho párrafo. Vienen a ser una especie de <div> pero que, en lugar de representar contenedores, sirven para separar párrafos pertenecientes a un mismo texto dentro de un contenedor común.
<h>	Encabezado (header) de nivel <i>i</i> , donde <i>i</i> es un número entero entre 1 y 6, ambos inclusive. El nivel es más general cuanto menor sea el número <i>i</i> . Así, los títulos de nivel principal son los h1. Los subtítulos de siguiente nivel serían h2 y así sucesivamente.

2.3.2.4. Elementos que dan formato al texto de un párrafo

IMPORTANTE: Antes que nada, hay que comentar que la tendencia actual es dejar el formato en manos de las hojas de estilo (por ejemplo, CSS). En el documento HTML deberíamos incorporar únicamente etiquetas “semánticas”, esto es, que delimitan las partes del documento indicando a qué se refiere cada una, pero que no definen su aspecto final.

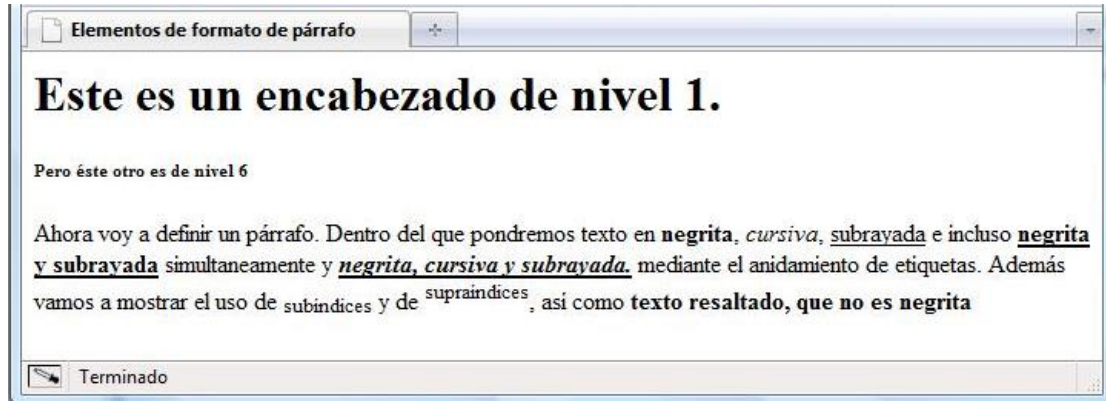
En todo caso, por operatividad y por compatibilidad con la ingente cantidad de archivos HTML que ya hay hechos y que las usan, veremos algunas de ellas:

Elemento	Descripción
	Negrita (bold).
<i>	Cursiva (<i>itálica</i>).
<u>	Subrayado (<u>underlined</u>).
<sup>	Superíndice ^(sup) .
<sub>	Subíndice _(sub) .
	Resaltado (fuerte). Habitualmente los navegadores resaltan el texto poniéndolo en negrita , si bien podría existir algún estilo que lo hiciera de otra forma.

Veamos un ejemplo de un documento HTML que utiliza estos elementos:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 transitional//EN" "http://www.w3.org/TR/REC-html40/loose.dtd">
<html>
  <head>
    <title>Elementos de formato de párrafo</title>
  </head>
  <body>
    <h1>Este es un encabezado de nivel 1.</h1>
    <h2>Pero este otro es de nivel 2.</h2>
    <p>Ahora voy a definir un párrafo. Dentro del que pondremos texto en <b>negrita</b>,
    <i>cursiva</i>, <u>subrayada</u> e incluso <b><u>negrita y subrayada</u></b> simultáneamente y
    <b><i><u>negrita, cursiva y subrayada.</u></i></b> mediante el anidamiento de etiquetas.
    Además vamos a mostrar el uso de <sub>subíndices</sub> y de <sup>supraíndices</sup>,
    así como <strong>texto resaltado, que no es negrita</strong>
    </p>
  </body>
</html>
```

Visto en Firefox quedaría así:



2.3.2.5. Elementos para listas

Hay tres tipos de listas:

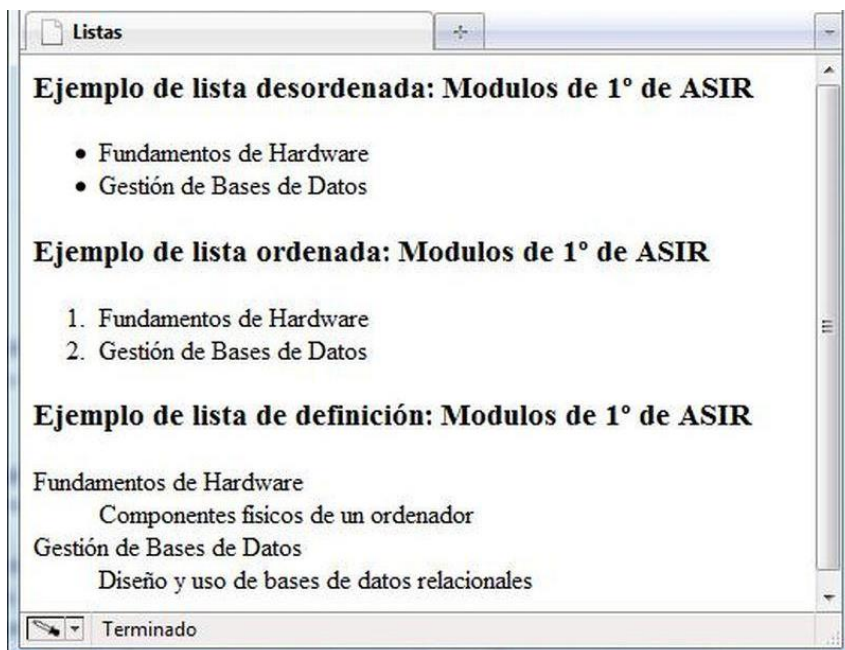
- Ordenadas: los ítems se numeran con ordinales 1, 2, 3...
- Desordenadas: los ítems aparecen con un símbolo (como esta lista).
- Listas de definición: los ítems son palabras acompañadas de texto.

A continuación, puede verse un ejemplo HTML y su apariencia en Firefox:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 transitional//EN" "http://www.w3.org/TR/REC-html40/loose.dtd">
<html>
  <head>
    <title>Listas</title>
  </head>
  <body>
    <h3>Ejemplo de lista desordenada: Modulos de 1º de ASIR</h3>
    <ul>
      <li>Fundamentos de Hardware</li>
      <li>Gesti&oacute;n de Bases de Datos</li>
    </ul>

    <h3>Ejemplo de lista ordenada: Modulos de 1º de ASIR</h3>
    <ol>
      <li>Fundamentos de Hardware</li>
      <li>Gesti&oacute;n de Bases de Datos</li>
    </ol>

    <h3>Ejemplo de lista de definici&oacute;n: Modulos de 1º de ASIR</h3>
    <dl>
      <dt>Fundamentos de Hardware</dt>
      <dd>Componentes f&iacute;sicos de un ordenador</dd>
      <dt>Gesti&oacute;n de Bases de Datos</dt>
      <dd>Diseño y uso de bases de datos relacionales</dd>
    </dl>
  </body>
</html>
```



Las etiquetas involucradas en los ejemplos son éstas:

Elemento	Descripción
	Delimita los elementos que forman una lista desordenada
	Delimita los elementos que forman una lista ordenada
	Indica cada uno de los elementos de una lista
<dl>	Delimita los elementos que forman una lista de definición
<dt>	Cada uno de los términos que se definen de una lista de definición.
<dd>	Cada una de las definiciones de una lista de definición.

2.3.2.6. Elementos para tablas

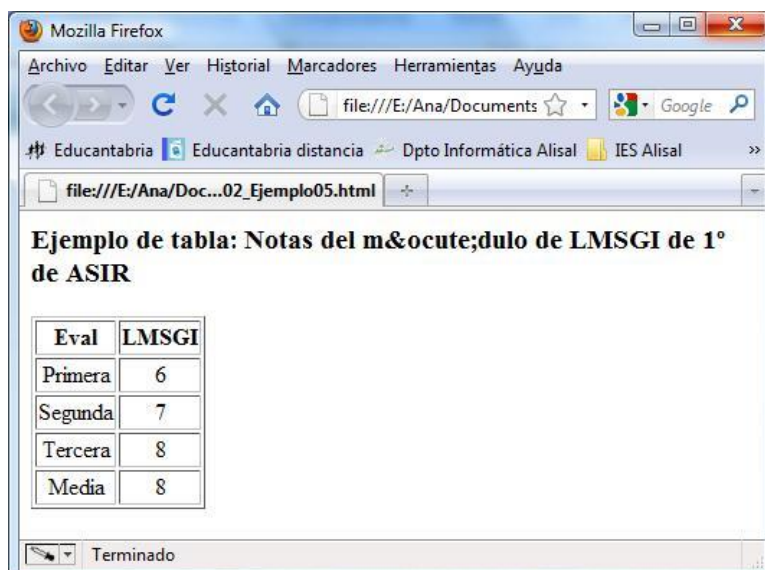
Las tablas también son un recurso que se usa cada vez menos, pero existe una gran cantidad de código HTML que las usa y por ello debemos estudiarlas.

Elemento	Descripción		
<table>	Delimita la tabla.	<tbody>	Permite agrupar líneas de la tabla.
<tr>	Delimita cada fila.	<thead>	Define la línea “cabecera” de la tabla.
<td>	Delimita cada celda.	<th>	Delimita cada celda de la cabecera.
<colgroup>	Permite agrupar columnas.	<tfoot>	Define la fila “pie” de la tabla.

Como siempre, mejor con un ejemplo:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 transitional//EN" "http://www.w3.org/T
<html>
<head>
<title>Tablas</title>
</head>
<body>
<h3>Ejemplo de tabla: Notas del m&ocute;dulo de LMSGI de 1º de ASIR</h3>
<table border = "1">
<thead> <tr> <th>Eval</th> <th>LMSGI</th> </tr> </thead>
<tfoot align="center"> <tr> <td>Media</td> <td>8</td> </tr> </tfoot>
<tbody align="center">
<tr> <td>Primera</td> <td>6</td> </tr>
<tr> <td>Segunda</td> <td>7</td> </tr>
<tr> <td>Tercera</td> <td>8</td> </tr>
</tbody>
</table>
</body>
</html>
```

Que interpretado por el navegador quedaría:



2.3.2.7. Elementos para formularios

Algunos de los elementos que puede contener un formulario son los siguientes:

Elemento	Descripción
<form>	Delimita el contenido del formulario.
<input>	<p>Caja para editar texto corto. Dependiendo del valor del atributo "type", podemos estar ante:</p> <ul style="list-style-type: none"> "text" : Un texto de carácter general. "password" : Contraseña, donde al escribir no se visualice el contenido. "radio" : Botón de radio que el usuario podrá elegir de entre otros que aparezcan. "checkbox" : Casilla de verificación que el usuario podrá activar o desactivar. "submit" : Botón de confirmación, para procesar el contenido del formulario. "button" : Botón de uso genérico (su pulsación debe procesarse con scripts). <p>HTML5 soporta nuevos tipos, con usos muy específicos pero que ayudan a los navegadores a tareas tales como cuidar que los datos de entrada sean válidos o mostrar el teclado virtual adecuado. En el siguiente enlace pueden verse, aquí simplemente los enumero (aunque, sabiendo inglés, podrás saber qué significa cada tipo): color, date, datetime, datetime-local, email, month, number, range, search, tel, time, url, week. El enlace es este:</p> <p>http://www.w3schools.com/html/html_form_input_types.asp</p>
<textarea>	<p>Caja de texto para texto largo. Dos atributos permiten definir las filas (rows) y las columnas que debe poseer (cols). Por ejemplo, así se mostraría una caja de 50 columnas y 4 filas:</p> <pre><textarea rows="4" cols="50"> At w3schools.com you will learn how to make a website. We offer free tutorials in all web development technologies. </textarea></pre>
<select>	Crea un menú desplegable que permite elegir de una lista de opciones que contiene el elemento.
<option>	<p>Delimita cada una de las opciones de un menú desplegable (<select>) que le contiene. Veamos un ejemplo de <select> y sus <option>s:</p> <pre><select> <option value="volvo">Volvo</option> <option value="saab">Saab</option> <option value="mercedes">Mercedes</option> <option value="audi">Audi</option> </select></pre>
<button>	Permite definir un botón. Su principal ventaja frente a los botones hechos con input es que este elemento permite introducir en el botón cualquier otro elemento de HTML, como por ejemplo, imágenes.

<fieldset>	Permite agrupar elementos de un formulario. El caso más típico es agrupar una caja de texto con el rótulo que le da título. Aquí puedes ver un ejemplo y, nuevamente, “jugar” con él a través del botón “Try it yourself”: http://www.w3schools.com/tags/tag_fieldset.asp
<legend>	Permite poner un título al fieldset. Ha quedado visto en <fieldset> y sus ejemplos.
<label>	Etiqueta de un campo del formulario. Es decir, un texto que sirve de “título” para un campo.

En los contenidos de EaD podéis ver una página HTML con un formulario de ejemplo. Si buscáis la opción del navegador de “ver código fuente”, veréis sus elementos y sus atributos.

2.3.2.8. Imágenes

Incrustar una imagen en un documento HTML es sencillo, basta con usar la etiqueta , la cual no requiere clausura. Eso sí, debemos profundizar algo en sus atributos más importantes:

Atributo	Descripción
src = "<ruta>"	Indica el nombre y la ruta, absoluta o relativa a la página, donde se encuentra la imagen que debe mostrarse en la página. Su valor puede ser: <ul style="list-style-type: none"> Una URI completa (ej: “https://www.cifpcarlos3.es/img/logo.png”) Simplemente el nombre del archivo, si está alojado junto a la página. Una ruta relativa a la página (por ejemplo “/img/corporate/logo.png”)
alt = “<texto>”	Establece una descripción referida a la imagen, la cual será mostrada por varios motivos, entre los que destacan: <ul style="list-style-type: none"> La imagen no se encuentre en la ruta indicada en “src”. El formato es incorrecto o está corrompida. Las imágenes están deshabilitadas por accesibilidad (invidentes).

Existen otros atributos tradicionalmente vinculados a las imágenes, pero tienen mucho que ver con su apariencia (dimensiones, posición...) así que mejor obviarlos y esperar a la parte de CSS, que es como realmente debemos modificar la apariencia de las cosas. Pueden consultarse en https://www.w3schools.com/tags/tag_img.asp.

2.3.2.9. Hipervínculos

Seguramente la etiqueta más emblemática de HTML, ya que la mayor aportación de este protocolo (desde sus inicios) fue dar la posibilidad de vincular unos documentos con otros (o partes de un mismo documento).

Hablamos de la etiqueta `<a>`, que requiere clausura, convirtiendo en hipervínculo todo aquello que esté contenido en su interior. Aunque lo normal es que dicho contenido se limite a un texto, en realidad es posible encerrar otros elementos (también contenedores con contenido propio). Un ejemplo de cada opción mencionada:

```
<a href="acercade.html">Acerca de</a>
<a href="acercade.html"></a>
```

Repasemos sus atributos principales:

Atributo	Descripción
<code>href = "<URI>"</code>	Referencia al recurso apuntado por el hipervínculo.
<code>target = "<destino>"</code>	Modifica el comportamiento al ejecutar la acción de ir al recurso apuntado. Los valores más comunes son: <ul style="list-style-type: none"> <code>_self</code> (por defecto), para utilizar el marco/ventana actual <code>_blank</code>, para utilizar una nueva ventana o pestaña <code>_parent</code>, para utilizar la ventana padre (que dio lugar a la actual) <code>_top</code>, para utilizar el cuerpo de la página actual <code>frame</code>: si se indica un nombre de <i>frame</i>, se utiliza éste como contenedor
<code>download</code>	La sola presencia de este atributo (sin darle valor) fuerza a que el recurso apuntado sea descargado cuando se ejecute la vinculación hacia él. Tiene sentido incluirlo en vínculos que apuntan a archivos, sea del tipo que sea (pero claro, si es un archivo <code>".html"</code> lo normal sería mostrar la página, no descargarla). Si además queremos proponer un valor por defecto para el archivo, entonces se puede acompañar al atributo del nombre propuesto como valor (por defecto, se propone el nombre original del archivo).

Merece la pena echar un vistazo a https://www.w3schools.com/tags/tag_a.asp para ver más atributos y posibilidades de esta importantísima etiqueta.

2.4. XHTML frente a HTML

El lenguaje XHTML es muy similar al lenguaje HTML. De hecho, no es más que una adaptación de HTML al lenguaje XML, el estándar XHTML 1.0 sólo añade pequeñas mejoras y modificaciones menores al estándar HTML 4.01, por lo que este último está prácticamente incluido en el primero, lo que hace que pasar del HTML 4.01 Strict a XHTML no requiere casi ningún cambio.

El lenguaje HTML tiene una sintaxis muy permisiva, por lo que es posible escribir sus etiquetas y atributos de muchas formas diferentes. Las etiquetas, por ejemplo, podían escribirse en mayúsculas, en minúsculas e incluso combinando mayúsculas y minúsculas. El valor de los atributos de las etiquetas se puede indicar con o sin comillas. Además, el orden en el que se abrían y cerraban las etiquetas no era importante.

La flexibilidad de HTML da lugar a páginas con un código desordenado, difícil de mantener y muy poco profesional.

XHTML soluciona estos problemas añadiendo ciertas normas en la forma de escribir las etiquetas y atributos.

2.4.1. XHTML: diferencias sintácticas y estructurales con HTML

El esquema básico del documento, para considerarse conforme a la especificación deberá cumplir las siguientes condiciones:

- El elemento raíz del documento debe ser <html>.
- El elemento raíz del documento debe indicar el espacio nominal XHTML usando el atributo *xmlns*. El espacio nominal para XHTML es <http://www.w3.org/1999/xhtml>.
- Debe haber una declaración DOCTYPE en el prólogo del documento. El identificador público incluido en la declaración DOCTYPE debe hacer referencia a alguna de las tres DTD definidas por el W3C, usando el identificador formal público correspondiente:

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

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

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

Restricciones básicas que introduce XHTML respecto a HTML en la sintaxis de sus etiquetas. Casi todas ellas son heredadas de las reglas de XML bien formado:

- Las etiquetas se tienen que cerrar en orden inverso al que se abren, nunca pueden solaparse.
- Los nombres de las etiquetas y atributos siempre se escriben en minúsculas.
- El valor de los atributos, incluso los numéricos, siempre se encierra entre comillas.
- Los atributos en los que el nombre coincide con su valor, no puede darse el valor por entendido, es decir, no se pueden comprimir (restricción heredada de XML). Este tipo de atributos no son muy habituales. Ocurre por ejemplo con “enabled”.
- Todas las etiquetas deben cerrarse siempre. XHTML permite que en lugar de abrir y cerrar de forma consecutiva la etiqueta (

) se puede utilizar la sintaxis
 para indicar que es una etiqueta vacía que se abre y se cierra en ese mismo punto.

Otras restricciones:

Además de las cinco restricciones básicas, XHTML incluye otros cambios más avanzados respecto a HTML, entre ellas:

- Antes de acceder al valor de un atributo, se eliminan todos los espacios en blanco que se encuentran antes y después del valor. Además, se eliminan todos los espacios en blanco sobrantes dentro del valor de un atributo. Por ejemplo, el texto

```
nombre = " Paco "
```

Se interpreta como:

```
nombre="Paco"
```

- El código JavaScript debe encerrarse entre unas etiquetas especiales (<![CDATA[y]]>) para evitar que el navegador interprete de forma errónea caracteres como & y <.
- Las páginas XHTML deben prescindir del atributo *name*. En su lugar, siempre debe utilizarse el atributo *id*.
- En XHTML es necesario separar el formato del contenido. Los párrafos deben separarse consistentemente y las cabeceras h1-h6 sólo deben usarse para destacar los diferentes apartados. Es recomendable dar el formato mediante CSS.

2.4.2. Ventajas e inconvenientes de XHTML sobre HTML

2.4.2.1. Ventajas

- Compatibilidad parcial con navegadores antiguos: la información se visualiza, aunque sin formato.
- Un mismo documento puede adoptar diseños radicalmente distintos en diferentes apartados.
- Sencillez a la hora de editar y mantener el código.

- Es compatible con los estándares que está desarrollando el W3C como recomendación para futuros agentes de usuario o navegadores.
- Los documentos escritos conforme a XHTML 1.0 presentan mejor rendimiento en las actuales herramientas web que aquellos escritos conforme a HTML.
- La separación de los contenidos y su presentación hace que los documentos XHTML se adapten mejor a las diferentes plataformas: pantallas de ordenador, pantallas de dispositivos móviles...
- Como es XML, se pueden utilizar fácilmente herramientas creadas para procesar documentos XML genéricos (editores, validadores, convertidores de formato, etc.).

2.4.2.2. Inconvenientes

- Algunos navegadores antiguos no son totalmente compatibles con los estándares, lo que hace que las páginas no siempre se muestren correctamente. Esto cada vez es menos problemático ya que estos navegadores van cayendo en desuso.
- Muchas herramientas de diseño web aún no generan código XHTML correcto.

2.5. Herramientas de diseño web

Este breve punto queda tratado en los apuntes de la plataforma. Su estudio sólo es necesario para abordar los exámenes on-line.

2.6. Hojas de estilo o CSS

CSS (Cascading Style Sheets) permite a los desarrolladores Web controlar el estilo y el formato de múltiples páginas Web al mismo tiempo.

Antes del uso de CSS, los diseñadores de páginas web debían definir el aspecto de cada elemento dentro de las etiquetas HTML de la página. El principal problema de esta forma de definir el aspecto de los elementos es que habría que definir el formato de cada uno de los elementos que formen la página, lo cual hace que sea muy difícil de actualizar.

CSS permite separar los contenidos de la página y su aspecto. Para ello se define en una zona reservada el formato de cada uno de los elementos de la web. Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a ella en las que aparezca ese elemento. Las hojas de estilo están compuestas por una o más reglas de estilo aplicadas a un documento HTML o XML.

Al crear una página web, se utiliza en primer lugar el lenguaje HTML/XHTML para marcar los contenidos, es decir, para designar la función de cada elemento dentro de la página: párrafo, cabecera, texto destacado, etc. Una vez creados los contenidos, se utiliza el lenguaje CSS para definir el formato de cada elemento.

CSS obliga a crear documentos semánticos HTML/XHTML, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.

Las hojas de estilos aparecieron poco después que el lenguaje de etiquetas SGML, alrededor del año 1970. Desde la creación de SGML, se observó la necesidad de definir un mecanismo que permitiera aplicar estilos a los documentos electrónicos. La guerra de navegadores y la falta de un estándar para la definición de los estilos dificultaban la creación de documentos que tuvieran igual apariencia en distintos navegadores.






El organismo W3C propuso la creación de un lenguaje de hojas de estilos específico para el lenguaje HTML. En 1995, el W3C añadió a su grupo de trabajo de HTML el desarrollo y estandarización de CSS.

- CSS 1, se publicó en 1996, es la primera recomendación oficial.
- CSS 2, publicada en 1998, es la segunda recomendación oficial.
- CSS 3, continúa en desarrollo desde 1998, dividiéndose en módulos cuyas especificaciones aparecen separadamente.

2.6.1. Soporte de CSS en los navegadores

El diseño web siempre está limitado por las posibilidades de los navegadores que utilizan los usuarios para acceder a sus páginas. Por este motivo, es imprescindible conocer el soporte de CSS en cada uno de los navegadores más utilizados del mercado.

Por ejemplo, en https://www.w3schools.com/cssref/css3_pr_box-shadow.asp es posible consultar la documentación sobre la propiedad “box-shadow” de CSS3. Dentro de la sección “Browser support”, podemos ver los navegadores que soportan esta propiedad y la versión desde la cual lo hacen:

Browser Support					
The numbers in the table specify the first browser version that fully supports the property.					
Numbers followed by -webkit- or -moz- specify the first version that worked with a prefix.					
Property					
box-shadow	10.0 4.0 -webkit-	9.0	4.0 3.5 -moz-	5.1 3.1 -webkit-	10.5

2.6.2. Cómo incluir CSS en un documento HTML o XHTML

Pueden verse ejemplos de cómo hacerlo en http://www.w3schools.com/css/css_howto.asp

Como paso previo hay que tener en cuenta que, en todo caso, las hojas de estilo CSS se definen mediante texto plano, usando una sintaxis concreta. Veamos un ejemplo preliminar:

```
h3 { color: green; }  
p { color: orange; font-family: Verdana; }
```

Esta podría ser una hoja CSS muy sencilla, compuesta por dos estilos:

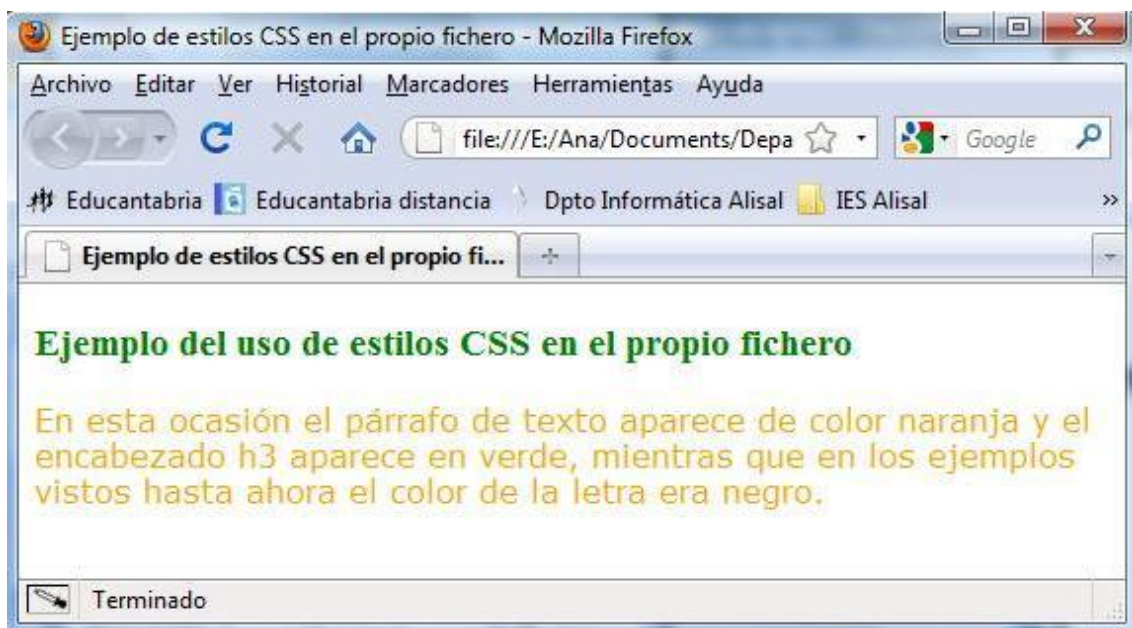
- Un estilo que se aplicará a los encabezados de nivel 3, estableciendo el color de fuente en verde.
- Otro estilo que se aplicaría a los párrafos, estableciendo el color de fuente en naranja y la tipografía de la fuente en “Verdana”.

En el supuesto de que la hoja de estilos resida en un archivo, éste debe tener la extensión **.css**.

Imaginando que los estilos anteriores se aplicaran sobre este <body> de una página html:

```
<body>  
<h3>Ejemplo del uso de estilos CSS en el propio fichero</h3>  
<p>En esta ocasión el párrafo de texto aparece de color naranja y el encabezado h3  
aparece en verde, mientras que en los ejemplos vistos hasta ahora el color de la letra era negro.</p>  
</body>
```

El resultado en Firefox sería:



2.6.2.1. Definir CSS en un archivo externo (External Style Sheet)

En este caso, todos los estilos CSS se incluyen en uno o varios archivos de texto plano (con extensión **.css**), que en HTML se enlazan mediante el elemento **<link>** de la cabecera (**<head>**).

Puesto que una página web puede tener asociados varios ficheros CSS, es recomendable agrupar estos últimos en un directorio.

El navegador descarga los archivos CSS externos, además de la página web asociada a ellos, y aplica los estilos a los contenidos de la página antes de mostrar sus contenidos.

Esta es la forma más utilizada de incluir CSS en las páginas HTML. La principal ventaja es que se puede incluir un mismo archivo CSS en multitud de páginas HTML, por lo que se garantiza la aplicación homogénea de los mismos estilos a todas las páginas que forman un sitio web.

Además, el mantenimiento del sitio web se simplifica al máximo, ya que el cambio en un solo archivo CSS permite variar de forma instantánea los estilos de todas las páginas vinculadas a él.

En el siguiente ejemplo:

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

Se muestra la porción <head> de un documento HTML válido. Por supuesto, faltan otras etiquetas, como <title>. En todo caso, puede verse la etiqueta <link> realizando la labor de enlazar la hoja de estilos. Tres atributos han contribuido a completar la vinculación:

- rel: para indicar que el vínculo (*link*) se refiere a una hoja de estilos.
- type: que indica que el archivo externo es de texto y con sintaxis *css* (*text/css*).
- href: que indica el nombre y, opcionalmente, la ruta absoluta o relativa hasta él.

A propósito del atributo *href*. Es importante comentar que hay tres formas de gestionar la ubicación del fichero *css*:

- **En la misma url** que la página. El ejemplo muestra esta forma, por eso *href* no incluye información de la ruta, sólo el nombre “mystyle.css”. Esta forma puede ser indicada para proyectos pequeños.
- **En una carpeta reservada** para los estilos, pero alojada en una ruta cercana a la página que hace referencia a ella. En este caso debe incluirse la ruta relativa. En el primer ejemplo, la hoja de estilos está situada en la carpeta “css” que hay dentro de la ubicación donde está la página HTML que hace referencia a ella:

```
<link ... href="css/mystyle.css" ...>
```

En éste, en cambio, la página html está dentro de una carpeta paralela a “css”:

```
<link ... href="../../css/mystyle.css" ...>
```

El nombre “css” no es obligatorio, podría ser cualquier otro o cualquier otra ruta.

- **En una URL diferente**, ya que la hoja de estilos puede pertenecer a un dominio diferente o servir a páginas de diferentes dominios. En este caso, se puede incluir la URL completa como referencia apuntada por href. Ejemplo:

```
<link ... href="http://www.cifpcarlos3.es/css/mystyle.css" ...>
```


2.6.2.2. Incluir CSS en el documento HTML (Internal Style Sheet).

Se consigue haciendo uso de la etiqueta `<style>`, que ya se comentó. El contenido de la etiqueta es idéntico al que aparecería en el fichero externo, no requiere nada especial.

Este método se emplea cuando se definen pocos estilos o cuando se quieren incluir estilos específicos en una determinada página HTML que completen los estilos globales de todas las páginas del sitio web.

Tiene el inconveniente de que, para modificar los estilos definidos, es necesario modificar todas las páginas que incluyen el estilo que se va a cambiar. Por eso, sólo se aconseja en proyectos muy pequeños y en situaciones que obliguen a mantener el menor número de archivos posible. En general, siempre será mejor el planteamiento anterior. Veamos un ejemplo:

```
<head>
  <style>
    body { background-color: linen; }
    h1 { color: maroon; margin-left: 40px; }
  </style>
</head>
```

En la página http://www.w3schools.com/css/tryit.asp?filename=trycss_howto_internal se muestra el ejemplo completo (incluyendo `<body>` con un par de elementos que usan los estilos), también con la posibilidad de alterar los valores y probar el resultado del cambio.

Una posibilidad a caballo entre las dos anteriores sería la de “importar” el archivo para que pase a formar parte del archivo HTML. Se comporta como un reemplazamiento de texto. Esta forma no aporta ninguna ventaja sobre la primera, por lo que no se suele usar. Al contrario, presenta el inconveniente de que siempre que se recargue la página, se tendrá que transmitir también la información de los estilos mientras que, si permanece como vínculo, el servidor web puede que no lo retransmita si entiende que, por no haber cambiado, no es necesario. En todo caso, esta sería la forma de “incrustar” una hoja de estilos “formatos.css” en la página:

```
<head>
  <style>
    @import 'formatos.css';
  </style>
</head>
```

2.6.2.3. Incluir CSS en los elementos HTML (Inline styles).

El último método para incluir estilos CSS en documentos HTML es el peor y el menos utilizado, ya que para modificar un formato hay que cambiar todos los elementos que estén asociados a

él. Consiste en indicar el estilo en la propia etiqueta que hace referencia al elemento al que se lo queremos aplicar. Esto se lleva a cabo mediante el atributo “*style*”, ideado para tal fin.

Solamente se utiliza, de forma muy excepcional, en determinadas situaciones en las que se debe incluir un estilo muy específico para un solo elemento concreto.

```
<html>
  <body>
    <h1 style="color:blue; margin-left:30px;">
      This is a heading.
    </h1>
  </body>
</html>
```

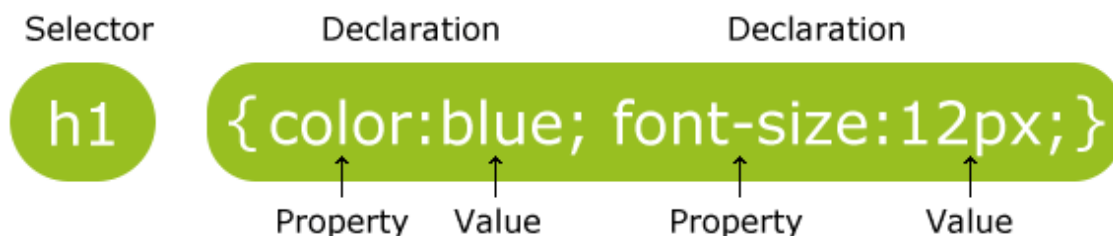
Solamente se utiliza, de forma muy excepcional, en determinadas situaciones en las que se debe incluir un estilo muy específico para un solo elemento concreto.

2.6.3. Sintaxis de las reglas de estilo

Cada uno de los estilos que componen una hoja de estilos CSS se denomina regla. Las reglas se escriben como texto plano, una a continuación de la otra.

Cada regla se forma por:

- **Selector/es:** Determina los elementos HTML a los que se aplica la regla CSS
- **Llave de apertura:** {
- **Declaración:** Especifica en qué consiste el cambio. Se compone de:
 - **Propiedad:** Nombre del aspecto o atributo del elemento que va a cambiar.
 - **Separador “:”:** Separa la propiedad de su valor.
 - **Valor:** Indica el nuevo valor del atributo modificado en el elemento.
 - **Separador “;”:** Permite establecer más de una declaración en una regla.
- **Llave de cierre:** }



En la página http://www.w3schools.com/css/css_syntax.asp puede ampliarse información sobre la sintaxis básica de CSS.

Veamos un ejemplo de regla:

```
p {color : blue}
```

En este caso el selector es "p", la declaración es: "color : blue" y, dentro de ésta, podemos diferenciar la propiedad "color" y el valor "blue". Fácilmente, puede deducirse que la regla lo que indica es que el color de fuente debe ser azul para los párrafos etiquetados como <p>.

Un archivo CSS puede contener infinitas reglas CSS, cada regla puede contener uno o varios selectores y, como ya se ha dicho, contener una o varias declaraciones.

En caso de presentar diferentes selectores, hay que separarlos por comas (,). Como ya se ha visto, las diferentes declaraciones se separan con punto y coma (;). Si la regla anterior afectara también a los encabezados de nivel 3 y pretendiera usar "Verdana" como tipografía, sería:

```
p,h3 {color : blue; font-family : Verdana}
```

Como puede verse, no es obligatorio terminar la última declaración con punto y coma. Tampoco es necesario encerrar los valores entre comillas, al contrario XML u otros lenguajes.

2.6.4. Atributos principales

2.6.4.1. Atributos de color y fondo

Los principales atributos de color y fondo son los que enumeramos a continuación:

Propiedad	Descripción
color	Indica el color del texto. Lo admiten casi todas las etiquetas de HTML. El valor de este atributo es un color que puede expresarse de muchas formas, que pueden estudiarse en la información dada tras esta tabla.
background-color	Color de fondo del elemento. Se aplica lo dicho anteriormente.
background-image	Permite colocar una imagen de fondo del elemento. El valor que toma es el nombre de la imagen con su ruta relativa o absoluta. También se recomienda ver la información tras la tabla, sobre cómo expresar rutas.
background-repeat	Indica si ha de repetirse la imagen de fondo y, en ese caso, si debe ser horizontal o verticalmente. Los valores que puede tomar son: repeat-x, repeat-y o no-repeat.
background-attachment	Especifica qué hace la imagen del fondo si se produce un scroll sobre el contenido. Los valores que pueden tomar son: <ul style="list-style-type: none"> • scroll La imagen acompaña al contenido • fixed La imagen permanece fija

background-position	Es una medida, porcentaje o el posicionamiento vertical u horizontal con los valores establecidos, que sirve para posicionar la imagen de fondo con respecto al contenido. Se recomienda visitar el link para consultar las posibilidades respecto a los valores que puede tomar.
background	Establece en un solo paso cualquiera de las propiedades de background anteriores. Para ello, admite que se le den múltiples valores, separados por espacios. Visitar link para ver ejemplos.

Formas de indicar el color

Por su importancia, vamos a concretar aún más cómo puede indicarse el valor de color. Básicamente éstas son:

- Por su nombre (blue, red, ...)
- Por su proporción de colores primarios RGB (rojo, verde y azul)
 - Usando el carácter # seguido de los valores hexadecimales de R, G y B, que siempre ocuparán dos dígitos. (Ej. #FF0000 para el rojo, #0000FF para azul...).
 - Usando la función rgb(), a la que se le pasan 3 valores decimales, entre 0 y 255, para las proporciones de RGB. (Ej. rgb(255,255,255) para blanco).
 - También existe la función rgba(), semejante a rgb() pero que habilita un cuarto parámetro: el nivel de opacidad o “canal alfa”, cuyo valor va de 0 a 1 (admite decimales), donde 0 es totalmente transparente y 1 totalmente opaco.

Más información en: http://www.w3schools.com/cssref/css_colors_legal.asp

Lista de los nombres de colores en: http://www.w3schools.com/cssref/css_colornames.asp

Formas de expresar rutas

Según se ha visto, algunas propiedades incluyen en sus valores referencias a archivos o recursos. Aquí no se descubre nada nuevo en cuanto a que dichas referencias pueden ser:

- Hechas únicamente a un archivo (si este está en la misma ubicación que la hoja).
- Relativas a una carpeta diferente pero cercana (usando “.” y “..” como siempre).
- Absolutas, indicando la URL completa que nos lleva al recurso a referenciar.

Ya se ha comentado que los valores de las propiedades CSS se indican sin comillas. Sin embargo, es importante tener en cuenta que, tanto el nombre del recurso como su ruta (relativa o absoluta) pueden contener espacios y estos NO están permitidos en los valores.

Por ello, CSS habilita una función “url()”, que sirve para indicar justamente eso: urls válidas y que sean interpretadas como tales. La url se indicará entre comillas (simples o dobles). Por supuesto, puede contener una referencia relativa o absoluta.

Si por ejemplo queremos que el fondo de nuestra página sea una imagen llamada “captura 2.png” y que se encuentra en una subcarpeta “img” que cuelga dentro de nuestra hoja, entonces la declaración de la regla sería algo así:

```
body { background-image : url('img/captura 2.png'); }
```

Nótese que hay un espacio entre “captura” y “2”. Algunos servidores podrían plantear un problema a esto, pero se resolvería usando la notación manejada por los navegadores:

```
body { background-image : url('img/captura%202.png'); }
```

2.6.4.2. Atributos de fuente

En este apartado vamos a ver los distintos atributos que podemos utilizar referentes a las fuentes de nuestro documento y que son:

Propiedad	Descripción
font-size	Indica el tamaño de la fuente. Puede expresarse de varias formas que, por su importancia, se describen separadamente, a continuación de esta tabla.
font-family	Establece la familia a la que pertenece la fuente. Si el nombre de una fuente tiene espacios se utilizan comillas para que se entienda bien. Pueden establecerse varios valores separados por comas, a modo de alternativas, de forma que el navegador las recorrerá y se quedará con la primera cuyas fuentes tenga instaladas en el sistema.
font-weight	Define el grosor de los caracteres. Los valores que puede tomar son: normal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800 o 900
font-style	Determina si la fuente es normal o cursiva. El estilo oblique es similar al de cursiva. Los valores posibles son: normal, italic, oblique.
font-variant	Determina si la fuente es normal o mayúsculas pequeñas (efecto conocido como “Versales”). Los valores que puede tomar son: normal o small-caps
line-height	El alto de una línea y por tanto, el espaciado entre líneas. Es una de esas características que no se podían modificar utilizando HTML. Usa las unidades de medidas que se verán en el apartado 2.6.6 de estos apuntes.
font	Al igual que ocurría con “background”, permite establecer todas las propiedades anteriores en el orden que se indica a continuación: font-style, font-variant, font-weight, font-size[line-height], font family. Los valores han de estar separados por espacios. No es obligatorio el uso de todos los valores.

Formas de expresar el tamaño de fuente

Antes que nada, comentar que en la página:

http://www.w3schools.com/cssref/playit.asp?filename=playcss_font-size&preval=medium

pueden seleccionarse las diferentes formas que se van a enumerar aquí, viendo en pantalla un ejemplo del código CSS requerido para usar el tamaño y una vista previa del resultado.

También diremos que en este vínculo: <https://css-tricks.com/almanac/properties/f/font-size/> se explica la diferencia entre las formas absolutas, relativas y como porcentaje. También hay ejemplos e información adicional sobre las unidades de tamaños en CSS.

Absoluta

Se fundamenta en que el tamaño expresado no depende del que tenga ningún otro elemento, sino que queda plenamente establecido con el valor que se esté dando. El resultado final sólo se verá afectado por el zoom (u otro mecanismo de magnificación) que el usuario tenga establecido en su navegador.

A su vez, se puede indicar el tamaño absoluto de varias formas:

- Mediante una unidad estándar de CSS (hay varias), que suelen referirse a píxeles de la pantalla, centímetros, el tamaño de un carácter, etc. Puede verse una descripción ampliada de estas unidades en el apartado 2.6.6. Se recomienda encarecidamente al alumno que consulte estas unidades. En el link de w3schools anterior, los ejemplos “10px” y “20px” (píxeles) son de este tipo.
- Mediante un identificador que se refiere a si es “normal”, “grande”, “pequeño”, etc. pero que, al final, resulta ser un “alias” de una medida y valor concretos. Concretamente estos identificadores son “xx-small”, “x-small”, “small”, “medium”, “large”, “x-large” y “xx-large”. Pueden verse también en el link de w3schools anterior.

Relativa

La diferencia sustancial es que aquí el tamaño depende del elemento “padre” que contenga al que queremos afectar. Tiene la ventaja de que se “adapta” al contexto en el que estemos. Es decir, el tamaño indicado será “más grande” o “más pequeño” que el que se estaba usando. Así, si el tamaño del elemento contenedor cambia, también lo hace el del elemento afectado por la regla relativa, manteniéndose constante la proporción entre ambos tamaños.

Traduciendo al inglés los términos entrecomillados anteriores, tenemos justamente los valores que expresan el tamaño de fuente de forma relativa. Estos son: “larger” y “smaller”.

Por ejemplo, al indicar un tamaño “larger” para un elemento contenido en otro con tamaño de letra “small”, éste quedará como “medium”.

Porcentaje

Es una forma relativa particular, donde el tamaño concreto se define mediante un valor numérico seguido del carácter %. Por ejemplo:

- 100% no provocaría ningún cambio de tamaño respecto del elemento padre
- 150% incrementaría en un 50% el tamaño del elemento hijo respecto a su contenedor.
- 50% reduciría a la mitad el tamaño de la letra, con respecto a su contenedor.

2.6.4.3. Atributos de texto

En el apartado anterior vimos los atributos relacionados con las fuentes y en este vamos a ver los relacionados con el texto en sí. Son los siguientes:

Atributo	Descripción
text-decoration	<p>Establece la decoración del texto en cuanto a líneas. Los posibles valores son:</p> <ul style="list-style-type: none"> • none: Por defecto, sin decoración alguna. • underline: Subrayado • overline: Sobrerayado • line-through: Tachado.
text-align	<p>Indica la alineación del texto. Aunque las hojas de estilo permiten el justificado de texto, no funciona en todos los sistemas. Los valores que puede tomar son los tradicionales: left, right, center o justify (justificación a ambos lados).</p>
text-indent	<p>Determina la tabulación del texto. Los valores que toma son una longitud, en unidades CSS (según 2.6.6.), o un porcentaje de la establecida.</p>
text-transform	<p>Nos permite transformar las mayúsculas/minúsculas del texto. Posibilidades:</p> <ul style="list-style-type: none"> • none: Por defecto, sin conversión alguna. • capitalize: Pone en mayúscula la primera letra de cada palabra. • uppercase: Todas las letras a mayúscula. • lowercase: Todas las letras a minúscula.
word-spacing	<p>Determina el espaciado entre las palabras, según unidades vistas en 2.6.6.</p>
letter-spacing	<p>Determina el espaciado entre letras, según unidades vistas en 2.6.6.</p>
vertical-align	<p>Establece la alineación vertical del texto. Sus valores posibles son:</p> <ul style="list-style-type: none"> • baseline: Por defecto, la <i>línea base</i>* alineada con el resto. • sub: Alinea el texto como un subíndice. • super: Alinea el texto como un superíndice. • top: La parte superior alineada con el elemento más alto. • text-top: Igual que la anterior, pero atendiendo al texto más alto. • middle: Se alinea a la mitad de la altura del elemento padre. • bottom: La parte inferior alineada al fondo del elemento padre. • text-bottom: Igual que la anterior, pero con el texto más bajo. • % Porcentaje de la altura del elemento contenedor padre.

	<p>* La <i>línea base</i> de un texto es esa línea imaginaria sobre la que se entiende que están “apoyadas las letras”, sin contar apéndices como el rabo de la “p” o la “q”.</p> <p>También se puede indicar una unidad de las vistas en (2.6.6), como longitud de desplazamiento vertical sobre la línea base.</p>
<u>line-height</u>	Altura de la línea. Puede establecerse mediante tamaño (ver 2.6.6) o porcentaje.

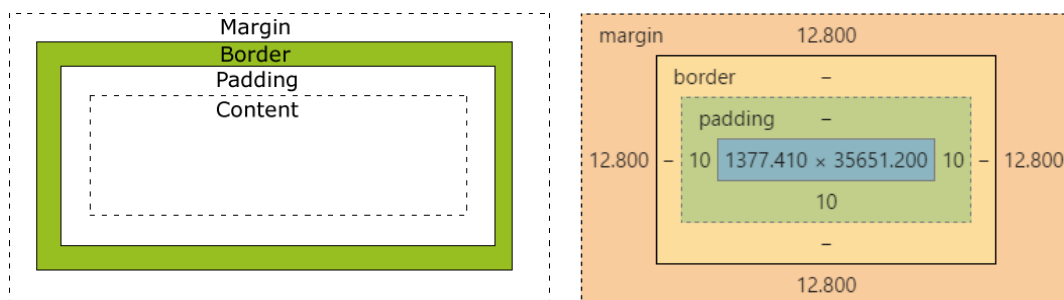
En la URL http://www.w3schools.com/cssref/playit.asp?filename=playcss_vertical-align&preval=baseline es posible probar varias de estas posibilidades y contemplar su efecto.

Para practicar todo lo aprendido, es recomendable hacer el ejercicio que se propone en los apuntes de la plataforma, al final de este mismo apartado (6.4.3)

2.6.4.4. El modelo de caja

Este modelo describe cómo se representa espacialmente cada uno de los elementos que participan en una web. Esto es. Se puede decir que cada elemento constituyente de la web (textos, imágenes, controles...) es una “caja”, puesto que ocupa un área rectangular. Así, los textos quedan encerrados en una “caja” o área, los botones también, etc.

En la web http://www.w3schools.com/css/css_boxmodel.asp se describe oficialmente el “*box model*” (modelo de caja). La información complementaria que aparece aquí está extraída de dicha url. Si representamos la “caja” a la que se refiere el modelo vemos que, en realidad, no se considera una única área, sino más de una. Representadas gráficamente éstas son:



La ilustración de la izquierda es la representación hecha en w3schools.com, mientras que la de la derecha es la que hace el navegador Google Chrome al inspeccionar los elementos de la página (cosa que puede hacerse con menú derecho sobre un elemento o con la tecla F12).

En ambos casos se pueden ver las áreas a las que nos referimos: margin, border, padding y content (en azul en la de la derecha).

- **Margin:** Área completamente libre (no puede ocuparse con nada) que queda desde el área ocupada por la caja hasta el siguiente elemento que esté junto a ella.
- **Border:** Área ocupada por un borde visible, a modo de marco, que delimita el perímetro del elemento. Es posible definir el color de este borde.

- **Padding:** Área libre que debe dejarse entre el borde del elemento y cualquier otro elemento contenido dentro de él.
- **Content:** Representa el contenido de un elemento, que puede ser contenido propio o bien otros sub-elementos incorporados dentro de él (si el elemento al que nos referimos es un contenedor).

Los atributos que veremos a continuación se refieren a los tres primeros anillos, que son los que realmente son “adicionales” al propio contenido (content).

Es importante recordar que las unidades en las que se pueden expresar los grosores admiten las mismas posibilidades que las enumeradas en el punto 2.6.6. Por su parte, los colores admiten las posibilidades estudiadas en el apartado 2.6.4.1.

También hay que destacar el valor “auto” que admiten los atributos de la familia “margin” (los relacionados con el margen. Este valor hace que, en realidad, se aplique el valor más alto posible o bien que se reparta (a partes iguales) si es que hay más de un margen con el mismo valor.

El ejemplo más claro puede apreciarse cuando se indica que tanto el margen izquierdo como el derecho son “auto”. Un elemento así aparecería centrado dentro de su contenedor, ya que el margen se ha hecho todo lo grande que podía, pero repartiéndose entre los dos laterales.

Propiedad	Descripción
margin-left	Tamaño del margen izquierdo.
margin-right	Tamaño del margen derecho.
margin-top	Tamaño del margen superior.
margin-bottom	Tamaño del margen inferior.
margin	Permite establecer los cuatro márgenes de una vez. Hay que seguir el orden: superior, derecho, inferior e izquierdo.
padding-left	Espacio izquierdo entre el borde y el contenido.
padding-right	Espacio derecho entre el borde y el contenido.
padding-top	Espacio superior entre el borde y el contenido.
padding-bottom	Espacio inferior entre el borde y el contenido.
padding	Establece el espacio entre los bordes y el contenido de una sola vez. Hay que respetar el orden superior, derecho, inferior e izquierdo.
border-left-color	Color del borde izquierdo del elemento.

border-rigth-color	Color del borde derecho del elemento.
border-top-color	Color del borde superior del elemento.
border-bottom-color	Color del borde inferior del elemento.
border-color	Establece el color de los bordes del elemento de una sola vez. Hay que seguir el orden superior, derecho, inferior e izquierdo.
border-style	<p>Establece el estilo del borde, los valores significan:</p> <ul style="list-style-type: none"> • none: ningún borde. • dotted: punteado (no funciona siempre). • dashed: a base de guiones más anchos que los puntos. • solid: sólido, un color uniforme en todo el trazo. • double: borde doble. <p>Además de éstos, están los valores groove, ridge, inset y outset, que son bordes con varios efectos 3D.</p>
border-left-width	Grosor del borde izquierdo. Sus valores posibles son: thin, medium, thick o un tamaño según 2.6.6.
border-rigth-width	Grosor del borde derecho. Sus valores posibles son: thin, medium, thick o un tamaño según 2.6.6.
border-top-width	Grosor del borde superior. Sus valores posibles son: thin, medium, thick o un tamaño según 2.6.6.
border-bottom-width	Grosor del borde inferior. Sus valores posibles son: thin, medium, thick o un tamaño según 2.6.6.
border-width	Establece el tamaño de los bordes del elemento al que lo aplicamos. Hay que seguir el orden superior, derecho, inferior, izquierdo.
width	Establece el ancho del contenido del elemento. El valor es automático (auto), un porcentaje de la anchura de su contenedor o un tamaño según 2.6.6.
height	Establece la altura del contenido del elemento. El valor es automático (auto), un porcentaje o un tamaño según 2.6.6.
float	<p>Sirve para indicar que un elemento puede “flotar” (recombinarse con otros elementos a la misma altura) y, en caso de hacerlo, si para ello debe alinearse a la izquierda o la derecha. Admite estos valores:</p> <ul style="list-style-type: none"> • none: Valor por defecto, no flota, no admite esa recombinación. • left: El elemento se alinea a izda. y admite otros a su dcha. • right: El elemento se alinea a dcha. y admite otros a su izda.
clear	Al contrario que el anterior, evita que un elemento pueda recombinarse a

izquierda o derecha con otro elemento, a pesar de que este tenga el atributo "float" establecido para tal fin Los valores posibles son:

- none: Valor por defecto, no existe esa prohibición.
- left: Se prohíbe recombinación por la derecha
- right: Se prohíbe recombinación por la izquierda.
- both. Se prohíbe recombinación por cualquier lado.

Para que practiques todo lo aprendido te recomiendo que intentes hacer el ejercicio que se propone en los apuntes de la plataforma, al final de este mismo apartado (6.4.3)

2.6.4.5. Atributos de clasificación

En este apartado vamos a ver otros atributos que hemos etiquetado como atributos de clasificación y que son los siguientes:

Atributo	Descripción
display	Determina el comportamiento del elemento, respecto a qué modo de diseño ha de seguir para ubicarse en la escena, junto a sus compañeros. Por su importancia, veremos los posibles valores tras la tabla.
white-space	Indica el modo en que se ha de gestionar los espacios en blanco que hay en el elemento, es decir, si se mantienen todos los existentes tal y como estén en el documento o si se anulan a uno las secuencias de blancos, es el valor por defecto y el de la opción normal. Valores que puede tomar son: pre, nowrap, normal.
list-style-type	Indica cual es el símbolo que se utiliza como marcador en las listas. Siguiendo el vínculo de la izquierda es posible consultar los diferentes valores y probarlos.
list-style-image	Permite utilizar una imagen como marcador en una lista. El valor que toma es la ruta del fichero imagen, usando la función <i>url</i> que ya vimos para <i>background</i> .
list-style-position	Determina la posición del marcador en una lista. Puede tomar los valores: outside o inside, según se desee fuera o dentro del área contenedora del item.
list-style	Permite establecer de una única vez todas las características de una lista. Hay que seguir el orden siguiente: list-style-type, list-style-position y list-style-image.

Valores para el atributo display

- **inline:** Es el valor por defecto, se indica que el elemento se ubicará siguiendo el flujo normal (en orden de escritura), a continuación del elemento anterior.
- **block:** Indica que se comportará como "elemento de bloque". Esto es, como elemento que no se recombina por la izquierda y derecha con ninguno, sino que ocupa la anchura completa del área disponible.

- **flex:** nuevo en CSS3 y tremendamente útil. Se dedica el anexo 2 de este tema a él. Básicamente permite definir el comportamiento del elemento en cuanto a cómo ha de alinearse, si puede crecer automáticamente, que proporción ha de guardar con otros elementos que también crecen automáticamente, etc.
- **table:** viene a sustituir al deprecado elemento html <table>, es decir, al indicar “display:table” el elemento se comporta como contenedor de una tabla. Por su parte, hay otro conjunto de posibles valores para referirse a los elementos de la tabla:
 - **table-header-group:** equivale al elemento html <thead>
 - **table-footer-group:** equivale al elemento html <tfoot>
 - **table-row-group:** equivale al elemento html <tbody>
 - **table-row:** equivale al elemento html <tr>
 - **table-cell:** equivale al elemento html <td>

Ejemplo de documento

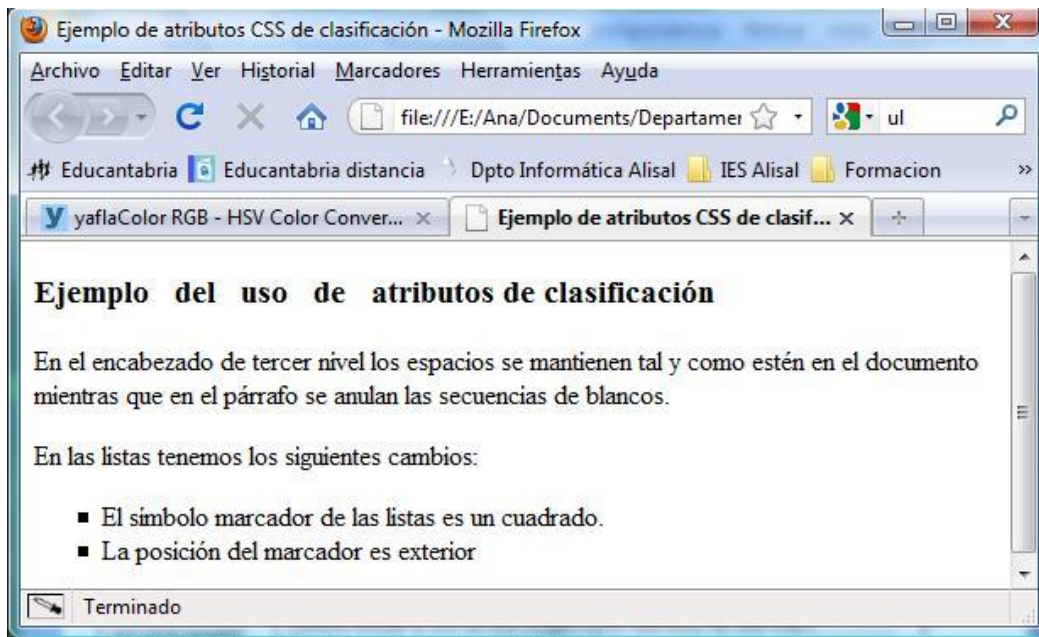
Un ejemplo de un documento XHTML en el que se utiliza este método para incluir formatos es:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

  <head>
    <title>Ejemplo de atributos CSS de clasificaci&acute;n</title>
    <style type="text/css">
      h3 { white-space:pre; }
      p { white-space:normal; }
      ul { list-style:square; list-style-position: outside;}
    </style>
  </head>

  <body>
    <h3>Ejemplo del uso de atributos de clasificaci&acute;n</h3>
    <p>En el encabezado de tercer nivel los espacios se mantienen tal y como est&eacute;n en el documento mientras que en el p&arrafo se anulan las secuencias de blancos.</p>
    <p> En las listas tenemos los siguientes cambios:</p>
    <ul>
      <li>El s&mbolo marcador de las listas es un cuadrado.</li>
      <li>La posici&n del marcador es exterior</li>
    </ul>
  </body>
</html>
```

Al publicarlo en un navegador (Firefox, por ejemplo), tendríamos:



2.6.5. CSS de posicionamiento (CSS-P)

A veces podemos encontrar el término CSS-P para referirnos a una parte de CSS que se refiere al posicionamiento, esto es, atributos que cambian el comportamiento de cómo ubicar en la pantalla a los elementos afectados. Estos son los principales atributos de este grupo:

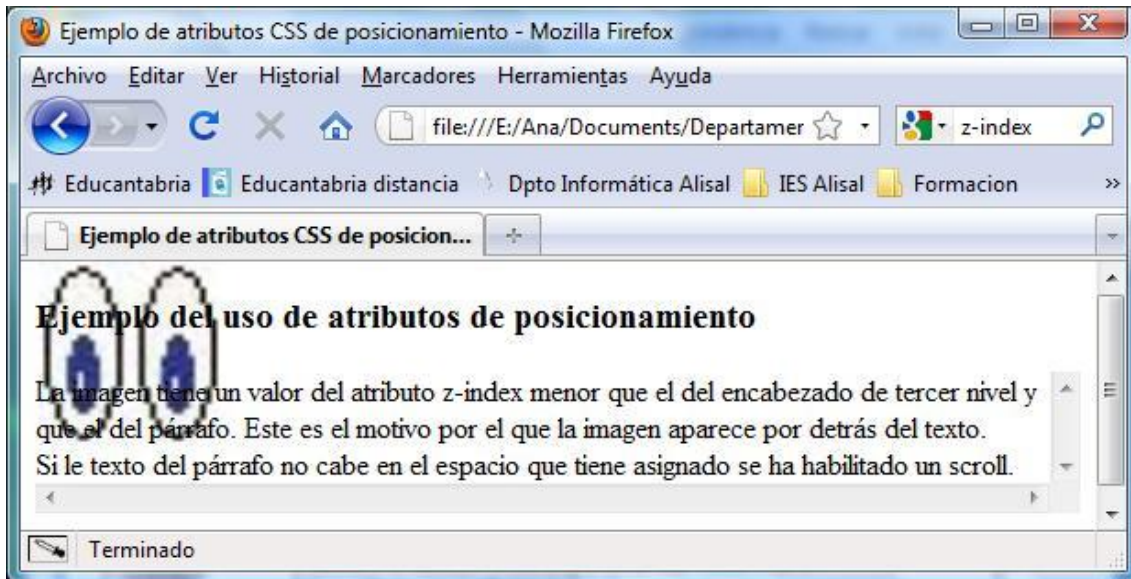
Elemento	Descripción
clip	Permite recortar una zona. Los valores que puede tomar son: <ul style="list-style-type: none"> rect (<i>top, right, bottom, left</i>): con esas cuatro unidades definimos el recorte. auto: no se aplica recorte, lo cual es el valor por defecto.
height	Permite establecer la altura de un elemento. Los valores que puede tomar son: auto o un tamaño, según las unidades válidas de 2.6.6.
width	Permite establecer la anchura de un elemento. Los valores que puede tomar son: auto, un tamaño o un porcentaje.
visibility	Indica si el elemento sobre el que actúa será visible o no. Los valores que puede tomar son: visible, hidden (oculto) o collapse (sólo para tablas, para ocultar filas).
left	Indica la posición del lado izquierdo del elemento. Los valores que puede tomar son: auto, un tamaño o un porcentaje.
top	Indica la posición del lado superior del elemento. Los valores que puede tomar son: auto, un tamaño o un porcentaje.

overflow	Indica si el elemento será visible o no en caso de superar los límites del contenedor. Los valores que pueden tomar son: visible, hidden, scroll o auto.
position	<p>Determinan si el posicionamiento de un elemento es:</p> <ul style="list-style-type: none"> • static: valor por defecto, el elemento se dibuja por orden, conforme aparece en el documento. • absolute: la posición dada para el elemento será relativa al contenedor en el que esté incluido. • fixed: la posición dada para el elemento será relativa a la ventana del navegador en el que la página aparezca. • relative: la posición dada para el elemento será relativa a la que debería ocupar en condiciones normales.
z-index	Define la posición del elemento en el tercer eje de coordenadas, permitiendo superponer unos elementos sobre otros como si fueran capas. El z-index tiene un impacto directo sobre qué elementos tapan a otros, en caso de superposición.

Un ejemplo de documento html que hace uso de posicionamiento absoluto para ubicar una imagen (la coloca a 10 pixels a la izquierda de la esquina superior izquierda de su contenedor) sería el siguiente:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Ejemplo de atributos CSS de posicionamiento</title>
    <style type="text/css">
      img{ position:absolute; left: 10px; top: 0px; z-index:-1;}
      p{overflow:scroll;}
    </style>
  </head>
  <body>
    <h3>Ejemplo del uso de atributos de posicionamiento</h3>
    </img>
    <p> La imagen tiene un valor del atributo z-index menor que el del encabezado de tercer nivel y que el del p<aaacute>rrafo. Este es el motivo por el que la imagen aparece por detr<aaacute>s del texto. <br><br> Si le texto del p<aaacute>rrafo no cabe en el espacio que tiene asignado se ha habilitado un scroll.</p>
  </body>
</html>
```

Al visualizar la página en Firefox se obtiene este aspecto:



2.6.6. Unidades de tamaño

Las principales unidades que podemos utilizar para indicar tamaños son las siguientes:

- Relativas
 - Element (em): Expresa el tamaño relativo al tamaño de la fuente utilizada.
 - X-height (ex): Expresa el tamaño relativo al de la letra "X".
 - Viewport width (vw): Porcentaje sobre el ancho de la ventana del navegador.
 - Viewport height (vh): Porcentaje sobre el alto de la ventana del navegador.
- Absolutas
 - Pixel (px): En monitores o dispositivos de alta definición (hdpi), 1px podría corresponderse con más de un pixel físico de dicho dispositivo, por lo que esta medida puede considerarse también como relativa.
 - Milímetros (mm).
 - Centímetros (cm).
 - Pulgadas (in): Recordemos que cada pulgada equivale a 2,54 cm.
 - Puntos (pt): Cada punto son 1/72 pulgadas (in).
 - Picas (pc): Cada pica son 12 puntos (pt).

En la URL http://www.w3schools.com/cssref/css_units.asp podemos consultar éstas y otras unidades de medida. Se recomienda encarecidamente al alumno visitar este vínculo y probar las opciones aquí descritas a través de los botones "Try it" que se habilitan para cada una.

2.6.7. Definición y uso de clases

Hemos visto como confeccionar las reglas para aplicar estilos referidos a muchos aspectos sobre la apariencia de la web. Comenzábamos el tema hablando de que, la sintaxis de estas reglas obliga a indicar en primer lugar el selector o selectores (separados por comas).

En el siguiente punto se explicará mejor qué puede ser un selector. De momento, recordemos que únicamente hemos usado un identificador de etiqueta como selector. Esto es, hemos establecido reglas donde el selector era “p”, en otros casos “h1”, etc. De esta forma, aplicábamos el estilo a todo el contenido que hubiera encerrado en el elemento o elementos flanqueados con dicha etiqueta.

Sólo con este tipo de selector estamos algo limitados. Por ejemplo, no podemos hacer que el formato de un párrafo <p> sea diferente de otro párrafo <p>, ya que tienen la misma etiqueta.

Planteado ya el problema, planteamos ahora una solución: permitir definir cualquier elemento html como de una “clase”, de forma que sea posible aplicar estilo a los elementos de dicha clase. Dicho y hecho, cualquier elemento html puede vincularse a una clase mediante el atributo *class*. Por ejemplo:

```
<h3 class="txt_azul">Queremos encabezados de nivel 3 azules</h3>
```

En realidad, aún no hemos conseguido el efecto, pero al menos ya hemos indicado que este encabezado será de la clase “txt_azul”. Ahora falta completar el CSS, lo cual se haría así:

```
.txt_azul { color:blue }
```

Indicamos que estamos estilizando una clase porque indicamos su nombre precedida del punto. Está será la forma de hacerlo, evitando confundir clases con elementos html.

Esta clase creada “txt_azul” puede usarse con elementos de otro tipo, por ejemplo, con párrafos:

```
<p class="txt_azul">Esto también saldrá azul</p>
```

Para restringir ese formato a un único elemento, podríamos haber puesto su identificador antes del punto (en el CSS, se entiende). Por ejemplo, si hacemos este cambio en el CSS:

```
.txt_azul { color:blue }  
h3.txt_azul { color:blue }
```

El encabezado sigue viéndose azul pero el párrafo ya no. Esto es así porque, con la nueva definición, el estilo se aplica a la clase “txt_azul” sólo cuando el elemento definido es <h3>.

Con la introducción del punto (.) hemos visto como acceder a un nuevo tipo de selectores, ya que se dice que el punto es el selector a nivel de clase.

2.6.8. Definición y uso de identificadores

Las clases nos resultan muy útiles porque permiten aplicar el mismo formato a elementos diferentes o bien usar múltiples formatos con elementos del mismo tipo. En todo caso, las usamos siempre que haya que aplicar el mismo formato a varias “partes” del documento.

En ocasiones, queremos aplicar formato a un único elemento de nuestra página. Necesitamos entonces un selector que aisle a dicha parte.

Los identificadores sirven para, como su nombre indica, identificar unívocamente a un elemento aislado de la página. Si *class* era nombre del atributo que definía la clase, será el atributo *id* el que de valor al identificador de un elemento. Es importante que el valor dado a un elemento no se repita en otro, de lo contrario se pierde la capacidad de identificarlo plenamente. Imaginemos que nuestra página posee un párrafo especial que queremos poner en negrita. El código html podría ser algo así:

```
<p id="nota_importante">MUY IMPORTANTE: Nótese que...</p>
```

Gracias al identificador podemos dar formato sólo a ese párrafo. El código CSS sería:

```
#nota_importante { font-weight:bolder }
```

En el que puede verse que el selector que actúa a nivel de elemento identificado es el que se consigue anteponiendo al nombre identificador el carácter almohadilla (#).

Estos identificadores cobran especial importancia cuando se aborda el estudio de Javascript, ya que buena parte del código que ha de relacionarse con elementos de la página, lo hace precisamente a través del identificador del elemento (aunque no siempre tiene que ser así).