

TECNOLOGÍAS WEB DEL LADO CLIENTE: HTML, CSS, JAVASCRIPT Y OTRAS

A la hora de desarrollar una aplicación basada en web, su propio nombre o requisito no funcional indica que se ha de ejecutar sobre un navegador web. Su arquitectura deberá ser cliente/servidor, pero tanto en el lado cliente como en el servidor, las posibilidades son inmensas en cuanto a los lenguajes y paradigmas a utilizar.

Un navegador web es capaz de interpretar y visualizar información codificada en HTML (HyperText Markup Language), organizarla según indique el CSS (Cascade Style Sheet), interpretar y ejecutar scripts (Javascript, VBScript[®]) e incluso mediante el uso de plugin de terceros, visualizar películas Flash[®] que ejecuten un programa hecho en lenguajes como Flex. También, gracias a un complemento, se pueden ejecutar *applet* Java[®], o ejecutar un videojuego online hecho con Unity[®].

Son tales las posibilidades que a la hora de planificar esta asignatura hemos tenido que seleccionar uno de los caminos posibles, y hemos optado por centrarnos en lo más común y a su vez más propio de la web: HTML + CSS + JS. ¿Por qué tres cosas?, pues porque cada una se encarga de una parte de lo que el navegador va a realizar:

- HTML codifica la estructura y el contenido de lo que queremos mostrar
- CSS define la estética y formato, independientemente del contenido
- JS contiene el código dinámico de la aplicación en el lado cliente

Esta enumeración anterior es muy importante, pues su cumplimiento o no denota la profesionalidad y conocimiento del que lo ha hecho y la calidad del producto desarrollado. No es lo mismo estructurar un documento (dividir su contenido en componentes distintos como encabezados, párrafos, listas, etc.) y darle formato (poner cursivas, cambiar colores, márgenes, etc.).

HTML5



HTML es un lenguaje de marcas. Una página HTML consiste en una mezcla de texto e imágenes enmarcados en etiquetas que indican al navegador su función lógica o semántica.

Por ejemplo, las dos siguientes líneas:

```
<h1>Sistemas de Información Basados en Web</h1>  
<p> Sistemas de Información Basados en Web</p>
```

representan el mismo texto ("Sistemas de Información Basados en Web") pero su función semántica en el documento HTML son diferentes. En la primera línea es un encabezado de primer nivel, y en el segundo caso es un párrafo normal. ¿Cómo se mostrarán? No nos interesa saberlo ahora, HTML tan sólo se debe ocupar del contenido, de la estructura semántica de la información.

¿Por qué tanto empeño en separar contenido de formato? Veamos varios motivos:

- Incrustar elementos de formato en el contenido conlleva un más costoso mantenimiento de lo desarrollado, pues los mismos valores pueden estar repetidos en varias etiquetas (márgenes, colores, fuentes, alineación, etc...)

- Un documento que contenga sólo contenido y organización lógica puede ser mostrado de diferentes maneras sin tener que tocarlo, tan sólo cambiando la parte estética.
- Los documentos lógicos transmiten significado. Por ejemplo, sería muy fácil realizar índices tan sólo leyendo los encabezados... y pensemos la utilidad de esto para un lector de pantalla para invidentes.

Por tanto, vamos a centrarnos en este apartado sólo en HTML, y en concreto en la versión HTML5 surgida en 2012.

El desarrollo de HTML5 ha sido coordinado por el W3C, y han intervenido los principales actores del negocio de la Web: AOL, Apple, Google, IBM, Microsoft, Mozilla, Nokia, Opera, etc.

Ha sido desarrollado pensando en que cualquier contenido necesario para la Web se pueda implementar sin necesidad de plugins adicionales, por lo que soporta nativamente reproducción de audio y video, visión 3D, animaciones, etc.

Además, como las versiones anteriores de HTML, es multiplataforma

Pasemos por tanto a desgranar los elementos más importantes de HTML5 que nos ayudarán a desarrollar un Sistema de Información Basado en Web

En el código 1 podemos ver un documento HTML con el conjunto mínimo de etiquetas exigido para que sea válido.

Se puede apreciar que las etiquetas son elementos enmarcados entre los símbolos < y >. En ese ejemplo podemos encontrarnos con las etiquetas <html>, <head>, <meta>, <title> y <body>.

La mayoría de éstas se encuentran por pares, pues su función es delimitar partes del contenido. Por tanto, podemos concretar que un elemento HTML se compone de:

```
<!DOCTYPE html>
<html>

  <head>
    <meta charset="UTF-8">
    <title>Título del documento</title>
  </head>

  <body>
    <!-- Aquí va el contenido..... -->
  </body>

</html>
```

Código 1: Documento HTML mínimo

- Una etiqueta de *inicio*
- Una serie de *atributos*, que son pares elemento-valor que añaden información importante a la etiqueta (p.ej. charset="UTF-8")
- Contenido textual
- Una etiqueta de *fin*, que tiene el mismo nombre que la de inicio pero es precedida por el símbolo '/'. Algunos elementos carecen de etiqueta de fin, como es el caso de *meta*, pero son los menos.

```
<!DOCTYPE html>
```

es la declaración del tipo de documento, y en este caso es específica para HTML5. La declaración <!DOCTYPE> es necesaria pues es la que le indica al navegador con qué tipo de documento se va a enfrentar y cómo tiene que interpretarlo. La del ejemplo es la estándar para HTML5, aunque hay muchas otras. Por ejemplo en la web de la UGR (www.ugr.es) encontramos la de XHTML1 estricto:

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

La estructura global de un documento HTML es la delimitada por la etiqueta <html> y se compone de dos secciones principales:

- La cabecera, delimitada por la etiqueta <head>, que incluye información general sobre el documento, como su título (delimitado por <title>), información para buscadores (mediante etiquetas <meta>), información sobre el formateado de la página, con CSS incrustado o vinculado, scripts, etc.
- El cuerpo, delimitado por la etiqueta <body> que incluye el contenido real de la página.

En nuestro ejemplo del código 1 hay poco más que enseñar, de hecho no se vería más que una página en blanco. La línea <!-- Aquí va el contenido..... --> es ignorada, pues **los símbolos <!-- y --> delimitan los comentarios.**

Pasamos por tanto a comentar algunos elementos fundamentales a la hora de desarrollar una página web en HTML5.

Hiperenlaces <a>

La etiqueta <a> define un hiperenlace, esto es, un contenido en el que al pulsar sobre él nos dirigimos a otra página o a otra parte dentro de la misma página.

El atributo más importante de la etiqueta <a> es el href que nos indica el destino del enlace. Si se define href, entonces se pueden incluir otros atributos como hreflang, download, target, etc.

Ejemplos:

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

Nos muestra el texto [Google](http://www.google.com/) y abrirá una nueva pestaña del navegador con la url www.google.com si hacemos click en dicho texto. ¿Cómo sabemos que es una nueva pestaña? Por el valor del atributo target, que es "_blank". Otros valores del atributo target son _parent, _self o _top.

```
<a href="notas.html">Ver Mis notas</a>
```

Nos muestra el texto [Ver Mis notas](#) y al hacer clic sobre él nos llevará a la página notas.html del mismo directorio en el que nos encontramos en la misma ventana y pestaña del navegador.

Un caso especial son los elementos <a> sin atributo href, pero con id. En este caso se refieren a marcas internas dentro la página, a las que se puede llegar por enlaces. Por ejemplo:

```
<a href="#frase1">Todo lo que des, lo recibas duplicado.</a>
```

nos llevaría al elemento HTML

```
<a id="frase1">Frase importante a no olvidar</a>
```

que puede estar antes o después del enlace, o incluso ser referido como "pagina.html#frase1" desde otro documento html.

Como se puede comprobar, es la etiqueta <a> la que permite a la Web ser lo que es hoy en día.

Imágenes

El elemento imagen es de los denominados *independientes*, esto es, emplea una única etiqueta, . Su sintaxis es:

```

```

Cuando el navegador se encuentra con una etiqueta de imagen, realiza una nueva petición HTTP al servidor para descargar el archivo especificado en el atributo `src` (de source).

El texto alternativo, indicado bajo el atributo `alt` es útil para mostrar información cuando el navegador no puede enseñar la imagen (porque ha desaparecido del servidor o porque se ha cortado la conexión), o cuando Google visita la página y queremos indicarle qué hay en ella.

Si además queremos especificar el ancho y alto con el que queremos mostrar la imagen, pues tan sólo hay que decirlo:

```

```

El uso de imágenes en una página web ha de tratarse con cierto cuidado, pues hay que distinguir entre dimensiones y tamaño. Podemos caer en la tentación de usar una imagen original de 1024x768 píxeles y 2MB de tamaño para una miniatura de 50x30 píxeles, lo que a todas luces es excesivo. Lo que hay que hacer en ese caso es una versión reducida de la imagen, que quizá nos ocupe 15-20KB y utilizarla para mostrar la miniatura.

Una ventaja de introducir el ancho y el alto de la imagen en el código es que el navegador, a pesar de no haber recibido el archivo con la imagen aún, conoce sus dimensiones y puede ir organizando el contenido de la página.

Elementos de Texto

Todo texto se agrupa en párrafos y tiene una serie de encabezados que sirven para organizar las secciones. Pues el texto HTML se organiza también igual, mediante sus etiquetas:

- Las cabeceras se definen mediante etiquetas que van de <h1> a <h6>
- Los párrafos se definen mediante la etiqueta <p>

Cuestiones importantes:

- ✓ Las cabeceras **no se usan** para poner el texto más grande o en negrita. Se usan para indicar que el texto enmarcado por <hX> </hX> es una cabecera.
- ✓ Las cabeceras son usadas por los motores de búsqueda para indexar la web
- ✓ Los saltos de párrafo se indican acabando un párrafo (</p>) y comenzando otro, no insertando un salto de línea

Otros elementos de descripción de texto interesantes son las listas, numeradas y no numeradas. Las listas numeradas se indican con la etiqueta `` (de *ordered list*) mientras que las no numeradas se indican con `` (*unordered list*). Ambas están formadas por elementos o ítems indicados con la etiqueta ``. En el código 2 se puede ver cómo cambia la visualización usando `` o bien ``. También se ve cómo es posible anidar las listas.

```
<ul>
<li>Primer ítem</li>
<li>Segundo ítem</li>
<li>Tercer ítem</li>
</ul>
<ol>
<li>Primer ítem</li>
<li>Segundo ítem</li>
<li>Tercer ítem</li>
<ul>
<li>Tres uno</li>
<li>Tres dos</li>
</ul>
</ol>
```

- Primer ítem
- Segundo ítem
- Tercer ítem

1. Primer ítem
2. Segundo ítem
3. Tercer ítem
 - Tres uno
 - Tres dos

Código 3: Listas en HTML

```
<table border="1">
  <thead>
    <tr>
      <th>Mes</th>
      <th>Ingresos</th>
    </tr>
  </thead>
  <tfoot>
    <tr>
      <td>Total</td>
      <td>1800,00€</td>
    </tr>
  </tfoot>
  <tbody>
    <tr>
      <td>Enero</td>
      <td>1000,00€</td>
    </tr>
    <tr>
      <td>Febrero</td>
      <td>800,00€</td>
    </tr>
  </tbody>
</table>
```

Mes	Ingresos
Enero	1000,00€
Febrero	800,00€
Total	1800,00€

Código 2: Tablas en HTML. A la derecha, salida obtenida.

Las tablas. `<table>`

Vamos a intentar detenernos brevemente en las tablas, una de las etiquetas peor tratadas históricamente en la Web y que aún es usada por algunos desarrolladores no muy avezados o anclados en el pleistoceno informático para maquetar ¡¡cuando no es su propósito!!

Como su propio nombre indica, las tablas se deben utilizar única y exclusivamente para mostrar información de forma tabulada, es decir, organizada por filas y columnas.

Una table se divide en filas (englobadas por la etiqueta `<tr>`), y cada fila se divide en celdas (delimitadas con la etiqueta `<td>`). `td` viene de "table data," y es el lugar donde se muestra el contenido de la celda, que puede ser texto, enlaces, imágenes, formularios e incluso otra tabla.

En el código 3 podemos ver una tabla completa, con la mayoría de las etiquetas posibles:

- `<table>` delimita la tabla
- `<thead>` delimita el encabezado de la tabla. Tiene la particularidad que cada celda no viene delimitada por `<td>` sino por `<th>`
- `<tfooter>` delimita el pie de la tabla.
- `<tbody>` delimita el cuerpo principal de la tabla, donde se encuentran los datos.

Si queremos que una celda ocupe más de dos columnas habrá que utilizar el atributo `colspan="n"` siendo `n` el número de columnas que queremos que ocupe.

```
<td colspan="2">Mira, ocupo dos columnas</td>
```

Agrupando elementos `<div>` y ``

La etiqueta `<div>` permite definir bloques de información, a modo de contenedor que agrupe otros elementos HTML. Su utilidad principal vendrá dada por la información semántica y organizativa que aporta, pues el navegador forzará a un salto de línea tras el `</div>`

Sin embargo, cuando se usa conjuntamente con CSS permite aplicar estilos a grandes bloques de contenido, definir las dimensiones del bloque, sus propiedades de fuente, color, e incluso su posición en la página.

La etiqueta `` es un elemento que se denomina *inline*, es decir, que no produce salto de línea cuando se utiliza, y sirve como contenedor de texto dentro de un párrafo o un ítem de lista. Se suele utilizar combinado con CSS para aplicar estilos especiales a sólo ciertas palabras del texto.

Formularios

Los formularios son una parte del estándar HTML imprescindible para el desarrollo de Sistemas de Información Basados en Web. Los usuarios interactúan con estos componentes, introducen información y pulsan un botón de envío. El navegador recopila los datos y los envía al servidor para su procesamiento. Algunas veces, las modificaciones o los datos introducidos en el formulario desencadenan acciones Javascript que modifican la apariencia de la página.

Todo formulario HTML está enmarcado en un elemento `<form>`, y todo lo que hay dentro es interpretado como parte del formulario

```
<form>  
...  
</form>
```

Dentro de un formulario se puede incluir cualquier etiqueta HTML, pero lo más importante son aquellas relativas a su función inherente: recopilar información, a saber:

<code><input type="checkbox"/></code>	Casilla de verificación
<code><input type="text"/></code>	Cuadro de texto de una línea
<code><input type="password"/></code>	Cuadro de texto de una línea que oculta lo escrito
<code><textarea>...</textarea></code>	Cuadro de texto de varias líneas
<code><input type="radio"/></code>	Selección única entre varias opciones
<code><input type="submit"/></code>	Botón de envío del formulario
<code><input type="image"/></code>	Imagen que hace las veces de botón de envío
<code><input type="reset"/></code>	Botón que reinicia el formulario
<code><input type="button"/></code>	Botón que al pulsar permite desencadenar un script
<code><select></code> <code><option>...</option></code> ... <code><select></code>	Lista en la que se pueden elegir una o más opciones
<code><label></code>	Es la etiqueta de un <code><input></code>
<code><fieldset></code>	Permite agrupar elementos
<code><legend></code>	Etiqueta o nombre para un <code>fieldset</code> enmarcado

Todos los elementos tienen varios atributos:

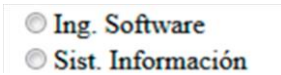
- Name. Identifica a la variable que almacenará el valor introducido
- Value. Permite dejar un valor por defecto o el texto del "submit"

Ejemplos:

```

<form>
<input type="radio" name="especialidad" value="IS">Ing. Software<br>
<input type="radio" name="especialidad" value="SI">Sist.
Información<br>
</form>

```




Código 5 Ejemplo Input Radio

```

<form>
<h3>Aficiones</h3>
<input type="checkbox" name="aficion" value="Fútbol">Fútbol<br>
<input type="checkbox" name="aficion" value="Ajedrez">Ajedrez<br>
</form>

```



Código 4 Ejemplo Input CheckBox

```

<form>
<form action="">
<select name="coche">
<option value="volvo">Volvo</option>
<option value="saab">Saab</option>
<option value="fiat">Fiat</option>
<option value="audi">Audi</option>
</select>
</form></form>

```



Código 7 Ejemplo Lista

```

<form action="">
<fieldset>
<legend>Datos personales:</legend>
<label>Nombre</label> <input type="text" size="30"><br>
<label>Apellidos</label> <input type="text" size="30"><br>
</fieldset>
</form>

```

Código 6 Ejemplo Fieldset y Label

Novedades en HTML5

HTML5 es la más reciente versión del estándar HTML. En el consorcio que lo ha desarrollado se encuentran AOL, Apple, Google, IBM, Microsoft, Mozilla, Nokia, Opera, y cientos de empresas

Las novedades más importantes que trajo bajo el brazo HTML5 fueron:

- El elemento `<canvas>` para dibujo 2D y 3D
- Reproducción nativa de `<video>` y `<audio>`
- Soporte para almacenamiento local
- Elementos específicos con semántica:
`<article>`, `<footer>`, `<header>`, `<nav>`, `<section>`
- Nuevos controles: calendario, fecha, hora, e-mail, URL, etc.

Otro detalle interesante es que se puede insertar en una página HTML directamente código MathML o SVG.

Algo también muy interesante es la desaparición de ciertas etiquetas que “invitaban” a generar un código despreciable, a saber: ``, ``, `<frame>`, `<center>` o `<big>`.

Uno de los detalles más interesantes y menos conocidos es la capacidad de usar almacenamiento local. Básicamente es como tener una base de datos en el lado cliente, para uso que anteriormente se dejaban en manos de cookies:

- Recuperar el estado de la aplicación al reiniciar el navegador
- Mantener datos en caché
- Guardar preferencias de usuario

Esta gestión de datos en el lado cliente se realiza mediante Javascript, accediendo al DOM de HTML5.

Obviamente, no todo es tan maravilloso, hay algunas pequeñas pegas, como puede ser la falta de un mecanismo de caducidad de la información (como tienen las cookies) o la imposibilidad de encriptar la información.

Finalmente, a falta de que usted complete sus apuntes con los elementos introducidos por el estándar HTML5 en sus apuntes, mostramos en la figura __ un gráfico tomado de www.html5doctor.com/semantics que le guiará a la hora de construir un sitio web:

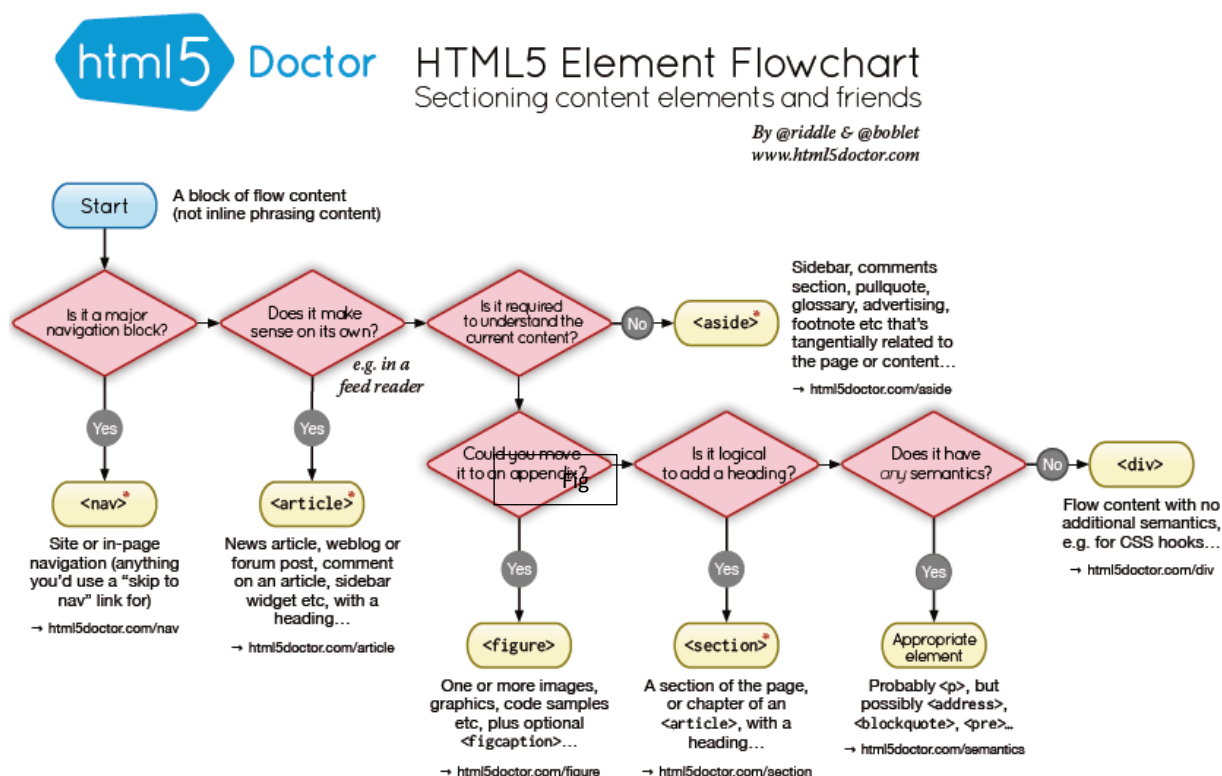


Ilustración 1: Diagrama de decisión sobre estructuras semánticas de

Las hojas de estilo se conocen por el nombre de su estándar, CSS (*Cascading Style Sheets*) y permiten definir la apariencia de los elementos de una página Web.

En el código HTML de una página hay normalmente una serie de directivas que indican al navegador que existe un estilo para mostrar dicho código HTML. Estas hojas de estilo se pueden aplicar de tres formas distintas:

- Mediante una hoja de estilo externa almacenada en un archivo independiente. Se indica mediante la etiqueta <link> dentro de la sección <head>:

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

- Mediante una hoja de estilo interna, incrustada en el propio documento HTML dentro de la sección <head>, como lo que nos encontramos en la página de inicio de la UGR:

```
<style type="text/css" media="all" >
  a.banner:hover{
    background-image:url(/img/banners/4Lcongresosugr_generico.png)
  }

  li.model-modern-normal_amarillo {
    background-image:url(/img/banners/modern-normal_amarillo.png)
  }
</style>
```

- Directamente dentro de la etiqueta de inicio de un elemento HTML, como también vemos en la página de inicio de UGR

```
<div id="fondo-dialog" style="display: none;"></div>
```

Anatomía de las reglas CSS

Las hojas de estilo contienen sólo una cosa: reglas, y todas tienen la misma estructura:

```
selector { propiedad:valor }
```

Por ejemplo, si queremos que los encabezados de primer nivel tengan un tamaño de fuente de 12px y sean de color azul, pondríamos:

```
h1 { color:blue;font-size:12px;}
```

Aunque lo normal es escribirlo de una forma más legible:

```
h1 {
  color:blue;
  font-size:12px;
}
```

El selector identifica el tipo de contenido al que se desea dar formato. La propiedad identifica el tipo de formato que quiere aplicarse, y el valor establece un rango para dicha propiedad. Las propiedades se separan por ;.

También es posible crear una única instrucción de formato que afecte a varios elementos distintos. Por ejemplo, si queremos que todos los encabezados hasta nivel sean de color #24fe21 haríamos lo siguiente:

```
h1,h2,h3 {
    color: #24fe21;
}
```

Los comentarios se incluyen entre los símbolos `/* y */`.

Los selectores de clase y de ID

¿Qué ocurre si sólo queremos aplicar un estilo a un encabezado de una sección muy concreta? Con las reglas anteriores se aplica a todos los h1 por igual, sin discriminar el contexto.

HTML y CSS proporcionan los selectores de clase y de ID, que permiten identificar elementos concretos de la página Web. Su sintaxis es como sigue:

```
<div class="menu">
  <ul class="listamenu" >
    <li id="seleccionado"> primero </li>
    <li> segundo </li>
  </ul>
</div>
```

De esta forma , podemos tener un CSS que especifique:

```
/* Selector de clase: se aplica a todos los elementos class="menu" */
.menu {
    background: #111111; /* color de fondo gris oscuro */
    font-size: 12px;
}

/* selector de clase: se aplica sólo a los <ul> de clase "listamenu"*/
ul.listamenu {
    list-style-type: none; /* sin marcador de lista */
}

/* CSS anidado:
se aplica sólo a los <li> contenidos en <ul> de clase "listamenu"*/
ul.listamenu li {
    font-weight: bold; /* negrita */
}

/* Selector de ID: se aplica al element que tenga ID="seleccionado"*/
#seleccionado {
    font-size: 14px; /* letra 14px (en lugar de 12px del resto) */
}
```

La diferencia entre `class` e `id` es que sólo puede haber un elemento en todo el documento HTML con un ID concreto, sin embargo, sí puede haber varios elementos de una misma clase.

En el caso de este ejemplo, si tenemos otras reglas en el CSS para los elementos `` en general, y hay conflicto con lo especificado para la clase `listamenu`, prevalece lo indicado para la clase.

Buenas reglas a la hora de escribir código CSS

- Los selectores se nombran en minúsculas
- El nombre de los selectores debe ser específico y claro
- El nombre de las clases no debe describir una característica visual

- Los nombres de clases e identificadores deben seguir una visión semántica
- Separar palabras mediante guiones o mayúsculas
- Agrupar las reglas según su selector
- Estructurar visualmente los atributos

El modelo de cajas

Para entender los estilos CSS, lo más adecuado es pensar en cualquier elemento como una caja, cuyas dimensiones y atributos pueden ser controlados para producir gran cantidad de efectos. Las propiedades de la caja se muestran en la figura 19.

Todas las propiedades mostradas se pueden adaptar y modificar para prácticamente cualquier etiqueta HTML, si bien son aquellas que delimitan bloques en las que más se usan (div, span, p, h, img, etc.).

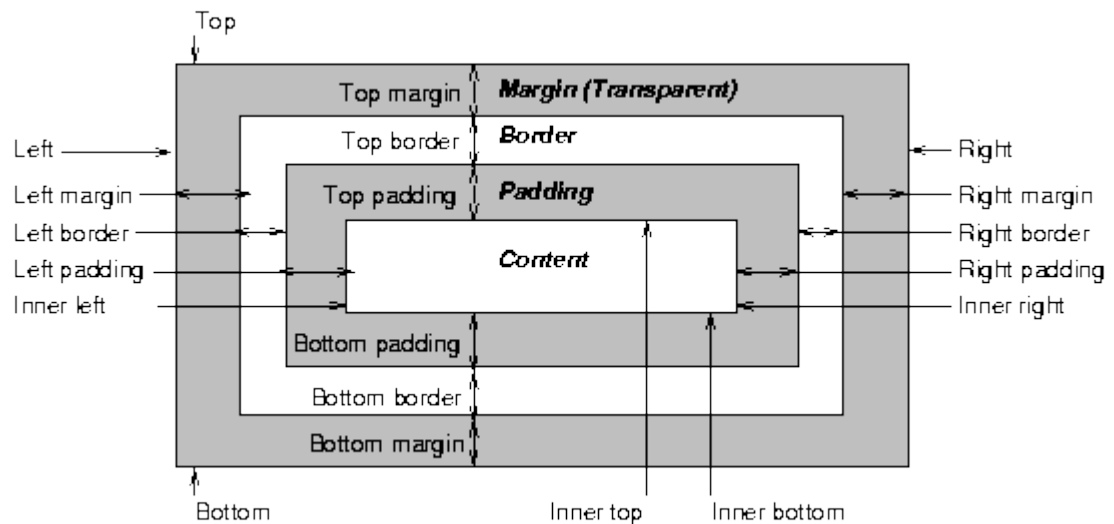


Ilustración 2: Modelo de caja CSS

Generalmente, las páginas HTML usan un modelo de diseño flotante: el contenido fluye desde la parte superior de la ventana del navegador a la parte inferior. El navegador coloca los elementos correlativamente para mostrarlos consecutivamente. Al usar propiedades CSS, los elementos “salen” de ese flujo y se organizan según las distintas reglas definidas.

La propiedad float permite extraer el elemento al que se aplica del flujo normal de HTML, y el resto de contenidos que se muestran posteriormente en el código “flotan” o se distribuyen alrededor de ese elemento flotante.

```

<!DOCTYPE html>
<html>
<head>
<style>
.cajaIzda
{
float:left;
width:110px;
height:90px;
margin:5px;
border: 1px;
background: #000;
color: #fff;
}
</style>
</head>

<body>
<h3>Ejemplo float</h3>
<p> Lo que va antes del div que flota a la izquierda
no se ve afectado por las propiedades de este div.
<div class="cajaIzda">
<p> Caja flotante </p>
</div>
<p>El texto que va después del div fluye a la derecha
de la caja que está float:left.</p>

</body>
</html>

```

Ejemplo float

Lo que va antes del div que flota a la izquierda no se ve afectado por las propiedades de este div.



El texto que va después del div fluye a la derecha de la caja que está float:left.

Ilustración 3: Ejemplo de elemento flotante.

Otra posibilidad para posicionar las cajas es colocarlas en una posición fija. Para ello, se indica la posición `absolute` y se marcan las propiedades de posición `top`, `left`, `bottom` y `right`.

Pasamos a mostrar algunos atributos de los más usados en CSS y que pueden servir para la elaboración de las prácticas

Atributo	Descripción	Valores
Atributos de fuentes		
color	Indica el color del texto	#RRGGBB
font-size	Tamaño de la fuente	Unidades (px, em) xx-small, x-small, large, etc.
font-family	Familia de la tipografía	serif sans-serif monospace cursive fantasy cualquier otra
font-weight	Anchura de los caracteres	Normal bold bolder lighter 400 700 ...
font-style	Estilo de la fuente	normal italic oblique
Atributos de párrafos		
line-height	Espaciado entre líneas	normal unidades (px, em)
text-decoration	Decoración del texto	none underline overline line-through
text-align	Alineación del texto	left right center justify
text-indent	Sangrado de párrafo	unidades (px, em)
text-transform	Transformar el texto	capitalize uppercase lowercase none
Atributos de fondo		
background-color	Color de fondo	#RRGGBB

Background-image	Imagen de fondo	URL de la imagen
Atributos de tablas		
caption-side	Posición del título	Top bottom
border-spacing	Espaciado entre los bordes	unidades (px, em)
Atributos de visibilidad		
overflow	Comportamiento del contenido si se desborda en la caja	visible hidden scroll auto
visibility	Visibilidad de las cajas	visible hidden collapse
display	Muestra una caja con diferentes estilos	inline block none list-item table inherit inline-table etc...
Atributos de listas		
list-style-type	Estilo del marcador	Disc circle square decimal lower-roman upper-roman lower-alpha none...
list-style-image	Imagen aplicable a los elementos de la lista	url("http://...")

Precedencia de estilos

La precedencia de estilos es una manera de indicar que un estilo definido prevalece por encima de otro definido en el mismo o en otro archivo CSS. El criterio general es que prevalece el definido en la regla más específica.

Además de este criterio, se puede obligar que un atributo se aplique por encima del resto de reglas, utilizando la declaración `!important` justo antes del punto y coma que cierra la regla:

```
p {
  background-color: red!important;
  background-color: yellow;
}
```