# CSS (Cascading Style Sheets)

## Índice de contenido

# 1 Estructura y sintaxis

Una hoja de estilo (**Cascading Style Sheet** - Hoja de estilo en Cascada) define la presentación de un documento, con independencia de su estructura.

Una hoja de estilo estará siempre ligada a un documento principal (HTML,XML, etc) al que llamaremos documento fuente.

**Una hoja de estilo CSS:**

- comienza con una cabecera compuesta por las llamadas reglas @, que aportan información sobre el contenido de la propia hoja.

```
@charset "UTF-8";
```

- Sigue con una secuencia de sistemas de reglas :

```
selector {
        propiedad1: valor;
        propiedad2: valor;
        /* ..... */
        propiedadN: valor;
}
```

**Selector:** indica el elemento (o elementos) del documento fuente a los que queremos aplicar estilo. Por ejemplo: un párrafo HTML o una imagen.
**Propiedad:** característica que queremos modificar. Por ejemplo: tamaño de letra o color.
**Valor:** que queremos que tome la característica. Por ejemplo: 1 cm o rojo.
**Declaración**: pareja propiedad : valor

- CSS no distingue entre mayúsculas y minúsculas .
- Un error en el selector invalida todo el sistema de reglas.
- Un error en una propiedad o en un valor anula sólo la declaración implicada, aplicándose el resto de reglas del mismo sistema.
- Si aparecen dos o más declaraciones sobre la misma propiedad se quedará con la última que pueda aplicar.
- Las palabras reservadas (propiedades o valores) que comienzan por guión (-) o guión bajo (_) son añadidos de los navegadores que no funcionarán en otras aplicaciones. La W3C recomienda no usarlas. (los añadidos de Iexplore comienzan por mso-, los de Mozilla por -moz-, los de Opera por -o-)
- El separador decimal será el punto (.)
- Al igual que en HTML podemos encerrar las cadenas entre comillas dobles o simples, pero sin mezclar ambas.
- El carácter de escape (\) se antepone a los caracteres especiales o saltos de líneas para forzar a que se ignoren.
- El carácter de escape (\) se antepone a números hexadecimales para incluir la correspondiente entidad de caracteres según la tabla ISO 10646, equivalente a Unicode. Si tras el número viene otro carácter hexadecimal tendremos que marcar su final, bien con un espacio en blanco, bien forzando su tamaño a 6 dígitos con ceros a la izquierda.

- Los comentarios CSS se encierran entre /* y */ No son anidables.
- Si nuestro código CSS está incrustado en un elemento <style> de HTML entonces los comentarios se encierran entre <!-- y -->

# 1.2 CSS incrustado o CSS externo

- **CSS externo** en un fichero separado .css y que tendremos que relacionar con LINK en el documento fuente .

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

- **CSS incrustado** aplicar estilo CSS dentro del propio documento HTML, incrustado.

```
<head>
        <meta http-equiv="Content-Style-Type" content="text/css">
</head>
<body>
        <p style="color:blue">Mi párrafo</p>
</body>
```

y también

```
<head>
 <style type="text/css">
        p {
                color: blue;
                background-color: red;
          }
 </style>
</head>
```

# 1.3 Formato de valores

CSS es tan flexible que permite indicar las medidas y colores de muchas formas diferentes.

## ➢ Unidades de Medidas:

Las medidas en CSS se emplean para definir la altura, anchura y márgenes de los elementos y para establecer el tamaño de letra del texto. Todas las medidas se indican como un valor numérico entero o decimal seguido de una unidad de medida (sin ningún espacio en blanco entre el número y la unidad de medida).

CSS divide todas las unidades de medida en dos grupos: *absolutas y relativas.*

1. Las medidas relativas definen su valor en relación con otra medida, por lo que para obtener su valor real, se debe realizar alguna operación con el

valor indicado.

2. Las unidades absolutas establecen de forma completa el valor de una medida, por lo que su valor real es directamente el valor indicado.

Si el valor es 0, la unidad de medida es opcional. Si el valor es distinto a 0 y no se indica ninguna unidad, la medida se ignora completamente, lo que suele ser una fuente habitual de errores para los diseñadores que empiezan con CSS. Algunas propiedades permiten indicar medidas negativas, aunque habitualmente sus valores son positivos.

### Unidades relativas :

- em, (no confundir con la etiqueta  de HTML) relativa respecto del tamaño (en puntos) de letra empleado. Aunque no es una definición exacta, el valor de 1em se puede aproximar por la anchura de la letra M del tipo de letra que se esté utilizando .
- ex, relativa respecto de la altura de la letra x ("minúscula") del tipo de letra que se esté utilizando .
- px, (píxel) relativa respecto de la pantalla del usuario.

Si se utiliza una tipografía de 12 puntos, 1em equivale a 12 puntos. El valor de 1ex se puede aproximar por 0.5 em.

Ej: Indica que el tamaño de letra del texto de la página debe ser el 90% del tamaño por defecto

```
body { font-size: 0.9em; }
```

/* Por lo tanto, 0.9em significa que se debe multiplicar 0.9 por el tamaño de letra por defecto.Si este tamaño por defecto es 12, el valor 0.9em sería igual a 0.9 x 12 = 10.8. */

La propiedad *font-size* permite establecer el tamaño de letra del texto de un elemento. En este caso, la referencia para el valor de font-size de un elemento siempre es el tamaño de letra de su elemento padre (es decir, del elemento en el que se encuentra). Si el `elemento` no se encuentra dentro de ningún otro elemento, la referencia es el tamaño de letra del elemento <body>. Si no se indica de forma explícita un valor para el tamaño de letra del elemento <body>, la referencia es el tamaño de letra por defecto del navegador.

Las medidas relativas no se heredan directamente, sino que se heredan sus valores reales una vez calculados.

```
body {
        font-size: 12px;
        text-indent: 3em;
    }
h1 { font-size: 15px }
```

Se calcula el valor real de 3em para el elemento <body>: 3em x 12px = 36px. Una

vez calculado el valor real, este es el valor que se hereda para el resto de elementos.

### Unidades absolutas:

- in, del inglés "inches", pulgadas (1 pulgada son 2.54 centímetros) cm, centímetros
- mm, milímetros
- pt, puntos (1 punto equivale a 1 pulgada/72, es decir, unos 0.35 milímetros) es la que más se utiliza.
- pc, picas (1 pica equivale a 12 puntos, es decir, unos 4.23 milímetros)

### Porcentajes:

CSS define otra unidad de medida relativa basada en los porcentajes. Un porcentaje está formado por un valor numérico seguido del símbolo % y siempre está referenciado a otra medida.

Los porcentajes se pueden utilizar por ejemplo para establecer el valor del tamaño de letra de los elementos:

```
body { font-size: 1em; }
h1 { font-size: 200%; }
h2 { font-size: 150%; }
```

Los tamaños establecidos para los elementos <h1> y <h2> mediante las reglas anteriores, son equivalentes a 2em y 1.5em respectivamente, por lo que es más habitual definirlos mediante em.

Los porcentajes también se utilizan para establecer la anchura de los elementos:

```
div#contenido { width: 600px; }
div.principal { width: 80%; }

<div id="contenido">
    <div class="principal">
        ...
    </div>
</div>
```

En el ejemplo anterior, la referencia del valor 80% es la anchura de su elemento padre. Por tanto, el elemento  cuyo atributo class vale principal tiene una anchura de 80% x 600px = 480px.

Normalmente se utilizan píxel y porcentajes para definir el layout del documento (básicamente, la anchura de las columnas y elementos de las páginas) y em y porcentajes para el tamaño de letra de los textos.

## ➢ Colores:

Los colores en CSS se pueden indicar de cinco formas diferentes: palabras clave, colores del sistema, RGB hexadecimal, RGB numérico y RGB porcentual. El método más habitual es el del RGB hexadecimal .

### Palabras clave:

CSS define 17 palabras clave para referirse a los colores básicos. Las palabras se corresponden con el nombre en inglés de cada color:

aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, orange, purple, red, silver, teal, white, yellow Este método prácticamente no se utiliza .
Además de la lista básica, los navegadores modernos soportan muchos otros nombres de colores. La lista completa se puede ver en http://en.wikipedia.org/wiki/Websafe.

### RGB decimal

El modelo RGB consiste en definir un color indicando la cantidad de color rojo, verde y azul que se debe mezclar para obtener ese color.

En CSS, las componentes de los colores definidos mediante RGB decimal pueden tomar valores entre 0 y 255.

El siguiente ejemplo establece el color del texto de un párrafo:

```
p { color: rgb(71, 98, 176); }
```

### RGB porcentual

El valor de las componentes RGB puede tomar valores entre 0% y 100%. El mismo color del ejemplo anterior se puede representar de forma porcentual:

```
p { color: rgb(27%, 38%, 69%); }
```

### RGB hexadecimal

Es el método más complicado y utilizado con mucha diferencia.

El sistema hexadecimal, utiliza 16 símbolos para representar sus números. Como sólo conocemos 10 símbolos numéricos, el sistema hexadecimal utiliza también seis letras (de la A a la F) para representar los números.

Para obtener el color completo en formato RGB hexadecimal, se concatenan los valores de las componentes RGB en ese orden y se les añade el prefijo #.

Una de las ventajas del formato RGB hexadecimal es que se pueden comprimir sus valores cuando todas sus componentes son iguales dos a dos .

#AAA = #AAAAAA
#A0F = #AA00FF

### Colores del sistema

Hacen referencia al color que muestran algunos elementos del sistema operativo. La lista completa de colores definidos se puede ver en:

http:// www.w3.org/TR/CSS21/ui.html#system-colors.

Este método no se utiliza.

### Colores web safe

Esta paleta de colores podía ser utilizada por los diseñadores con la seguridad de que se verían correctamente en cualquier navegador de cualquier sistema operativo de cualquier usuario.

La lista completa de colores web safe y sus valores hexadecimales se pueden consultar en http://en.wikipedia.org/wiki/Web_colors#Web-safe_colors.

# 2 Propiedades para color y fondo

| ***Propiedad*** | ***función*** | ***Valor*** | ***observaciones*** |
|---|---|---|---|
| *color* | Color del texto | color | |
| *background-color* | Color de fondo | color | |
| *background-image* | Imagen de fondo | url | Conviene indicar también un color de fondo para las zonas transparentes de la imagen y por si no se encuentra la imagen |
| *background-repeat* | Mosaico con repeticiones de la imagen | **repeat** | La repite horizontal y verticalmente |
| | | **repeat-x** | La repite horizontalmente |
| | | **repeat-y** | La repite verticalmente |
| | | **no-repeat** | Sólo aparece una vez |
| *background-attachment* | | **scroll** | Se desplaza con el bloque |
| | | **fixed** | Fija respecto del acceso visual |
| *background-position* | Alineación de la imagen dentro de la caja(relleno) | porcentaje | 0% indica limite izquierda o arriba<br>100% indica limite derecha o abajo |
| | | medida | desde la esquina superior izquierda |
| | | **top** | uno o dos valores, se pueden mezclar con medidas y porcentajes<br>cuando sólo hay un valor, el valor que falta se supone centrado |
| | | **bottom** | |
| | | **left** | |
| | | **right** | |
| | | **center** | |

# 3 Propiedades para la fuente

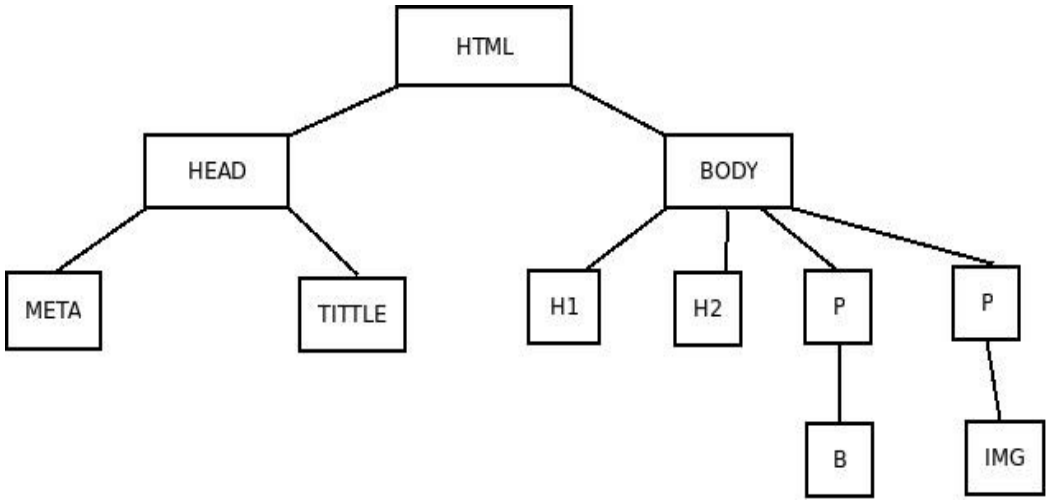
| ***Propiedad*** | ***función*** | ***Valor*** | ***observaciones*** |
|---|---|---|---|
| *font-family* | | lista entre comas de familias | Toma la primera disponible, se recomienda que la última sea genérica. Las genéricas de CSS son serif (como times), sans-serif (como helvética), cursivo, fantasia y monospace . |
| *font-style* | | **normal** | estos nombres cambian según la BD que acompaña a la fuente; si no se dispone de italic, intentará oblique (simulación por software de itálica) |
| | | **italic** | |
| | | **oblique** | |
| *font-variant* | | **normal** | |
| | | **small-caps** | Versalitas: minúsculas con aspecto de mayúsculas pero más pequeñas . |
| *font-weight* | grosor | **100\|<br>200\|....\|<br>900** | la única garantía es que cada uno es igual o<br>más oscuro que el anterior. |
| | | **normal** | como 400 |
| | | **bold** | como 700 |
| | | **bolder** | +100 sobre el valor heredado |
| | | **lighter** | -100 sobre el valor heredado |
| *font-size* | tamaño de fuente | medida | absoluta o relativa |
| | | **xx-small** | de menor a mayor tamaño |
| | | **x-small** | |
| | | **small** | |
| | | **medium** | |
| | | **large** | |
| | | **x-large** | |
| | | **xx-large** | |
| | | **larger** | un escalón más sobre el heredado |
| | | **smaller** | un escalón menos sobre el heredado |

# 4 El árbol del documento fuente

*El DOCUMENTO FUENTE ES EL QUE DEFINE EL CONTENIDO* , aquel contenido al que pretendemos aplicar estilo con CSS. En los ejemplos de este tema el documento fuente estará escrito en HTML, pero debe quedar claro que se puede aplicar estilo CSS a documentos fuente escritos en otros lenguajes de marcas.

En cualquier caso, el documento fuente tendrá siempre una estructura arbórea con una sola raíz, en la que cualquier nodo puede tener un sólo padre pero indefinidos hijos.

Dado un documento fuente, CSS lo primero que hará será construir el árbol asociado. El árbol correspondiente a un documento HTML podría ser el siguiente:




```
<html>
 <head>
   <meta http-equiv="content-type" content="text/html;charset=UTF-8">
   <title>documento html básico</title>
 </head>
 <body>
    <h1>IES Julio Verne</h1>
    <h2>introducción</h2>
    <p>el <b>instituto julio verne</b> se encuentra en el barrio pino montano de sevilla.</p>
    <p><img alt="imagen de la fachada" src="fachada.png"></p>
 </body>
</html>
```

NOTA: Si hemos hecho una indentación correcta, veremos que todos los elementos HTML que son hermanos en el árbol, comienzan en la misma columna del documento.

NOTA : Obsérvese que la declaración del tipo del documento (DOCTYPE) no tiene un nodo en el árbol. Sólo los elementos HTML forman parte del árbol.

La estructura en árbol da lugar a una serie de conceptos que, aunque son muy intuitivos, señalamos a continuación para ratificarlos:

➢ raíz: nodo primero del que cuelga todo el árbol. Es el único nodo que no tiene padre.

➢ padre: (de un nodo) aquel inmediatamente superior al que está ligado. Todos los nodos del árbol tienen uno y sólo un padre (salvo el nodo raíz que no tiene padre)
➢ hijo: (de un nodo) aquellos inmediatamente inferiores a los que está ligado. Cada nodo puede tener un número ilimitado de hijos o puede no tener ninguno. (los nodos sin hijos se llaman hojas)
➢ descendiente: (de un nodo) el hijo, el hijo de sus hijos, etc.
➢ antepasado: (de un nodo) el padre, el padre de su padre, etc.
➢ hermanos: los que comparten un mismo padre. Puesto que el orden de los hermanos importa, se distingue entre hermanos precedentes (los que están a su izquierda), hermanos siguientes (los que están a su derecha) y hermanos adyacentes (los que están seguidos, sin otros hermanos entre ellos)
➢ nodo precedente: (de un nodo) el que es su antepasado o su hermano precedente.
➢ nodo siguiente: (de un nodo) el que es su descendiente o su hermano siguiente.

# 5 Selectores (I)

Selector: indica el elemento del documento fuente al que se aplicará la regla.

## 5.1 Selectores Simples

➢ Selector universal: * las reglas se aplicarán a todos los elementos del documento.
➢ Cualquier elemento HTML: Las reglas se aplicarán a todas las ocurrencias de ese elemento HTML en el documento.

Ej:

```
* {
        font-family: serif;
  } /* indica la fuente para todos los elementos del documento */

li {
        font-family: serif;
    } /* indica la fuente para todos los elementos de listas */
```

## 5.2 Selectores Específicos

Un selector simple seguido del símbolo almohadilla (#) y seguido de un identificador: La regla se aplica al elemento indicado siempre que tenga un atributo **id** cuyo valor corresponda al identificador.

```
img#logo {
                margin: 2cm;
            }
```

Se lee la imagen de id logo.
NOTA: Se puede omitir el elemento quedando #identificador

## 5.3 Selectores de Clase

Un selector simple seguido de un punto (.) y seguido de un nombre de clase: La regla se aplicará a todos los elementos indicados que tengan un atributo class cuyo valor incluya al identificador. (Un atributo class puede asociarse a varios elementos dentro de la página).

```
img.playa {
            margin: 2cm;
            }
```

se lee "los IMG de la clase playa"

## 5.4 Selectores Compuestos

- Varios selectores separados por comas: Las reglas se aplicarán a cada uno de los selectores incluidos.

```
li, p {
        font-family: serif;
    }
```

Afecta a todos los párrafos y a todos los elementos de lista.

- Varios selectores separados por espacios: descendientes

Selecciona los elementos que se encuentran dentro de otros elementos.
El motivo es que en el selector descendente, un elemento no tiene que ser "hijo directo" de otro. La única condición es que un elemento debe estar dentro de otro elemento, sin importar lo profundo que se encuentre.

La sintaxis formal del selector descendente se muestra a continuación:

elemento1 elemento2 elemento3 ... elementoN

```
li p {
        font-family: serif;
    }
```

se lee "los P que son descendientes de LI". Obsérvese que se lee hacia atrás o, dicho de otro modo, los elementos finales a los que afectan las reglas son los que están inmediatamente antes de la llave de apertura ({).

- Varios selectores separados por el signo "mayor que" (>): hijos

```
ol > li {
font-family: serif;
}
```

se lee “los LI que son hijos de OL". La regla se aplicará a los elementos de listas ordenadas”.

- Varios selectores separados por el signo "más" (+): hermanos adyacentes

```
h1 + p {
font-family: serif;
}
```

se lee "los P que siguen inmediatamente a H1". La regla se aplicará al primer párrafo justo después de un encabezado de primer nivel.

## 6. Colisión de estilos en CSS.

Al escribir código CSS para una página, sobre todo en páginas complejas, puede ser que se apliquen diferentes valores en una propiedad, para una misma etiqueta. El navegador debe decidir qué estilo aplicar en función de unas prioridades:

Ejemplo: Se aplican estas dos etiquetas a la misma página:

p { color: red }

p { color: blue }

El navegador debe decidir si el color de los párrafos será rojo (red) o azul (blue), en este caso, y si no influye ninguna otra circunstancia, tendrá prioridad el valor designado en segundo lugar. Pueden darse otros casos. Para ello se establece una lista de prioridades.

Lo más específico tiene mayor prioridad:

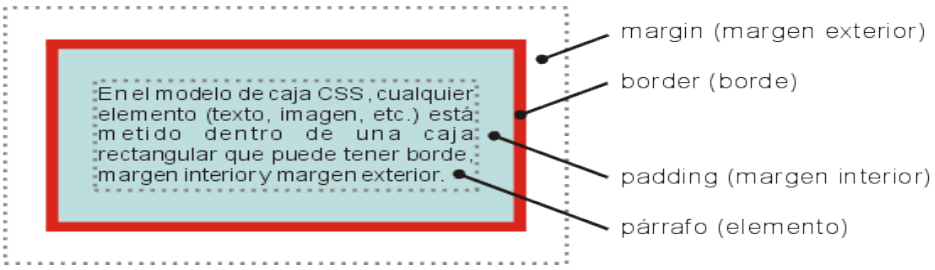
- Especifidad del selector: A mayor especifidad del selector, mayor prioridad en su aplicación y quedaría como:
  - selector de #id.
  - selector de clase .class
  - selectores de etiqueta.
  - selector universal (*) es el menos prioritario.
- Especifidad tipo de hoja de estilo: Hay tres formas de enlazar el código css, desde un archivo exterior, que sería el menos específico, desde una etiqueta **style** en la cabecera, más específico que el anterior, y desde el atributo **style=".."** en la propia etiqueta, que sería el que tiene mayor prioridad.
- Si después de lo anterior, varias reglas tienen la misma especifidad, tendrá prioridad la última regla indicada en el código.
- Podemos hacer una excepción en las normas de prioridad mediante la palabra clave ! `important`. Al escribirla detrás de cualquier valor damos prioridad a esa propiedad sobre cualquier otra con la que colisione.

## 7 Propiedades del texto

| ***propiedad*** | ***Función*** | ***Valor*** | ***observaciones*** |
|---|---|---|---|
| **text-indent** | sangría de primera línea | medida | puede ser negativa |
| | | porcentaje | respeto del ancho de la caja de contención |
| **text-align** | alineación horizontal del texto | **left** | izquierda |
| | | **right** | derecha |
| | | **center** | centrado |
| | | **justify** | justificado: izquierda y derecha a la vez |
| **text-decoration** | | **none** | Sin rayar |
| | | **underline** | subrayado |
| | | **overline** | sobrerayado |
| | | **line-through** | tachado |
| | | **blink** | parpadeo intermitente |
| **letter-spacing** | espaciado de letras | **normal** | |
| | | medida | puede ser negativa |
| **word-spacing** | espaciado de palabras | **normal** | |
| | | medida | puede ser negativa |
| **text-transform** | | **capitalize** | 1a letra de cada palabra en mayúscula |
| | | **uppercase** | todo mayúsculas |
| | | **lowercase** | todo minúsculas |
| | | **none** | respeta el original |

# 8 Modelo de Cajas

El "box model" es el comportamiento de CSS que hace que todos los elementos incluidos en una página HTML se representen mediante cajas rectangulares. CSS permite controlar el aspecto de todas las cajas. Las cajas de una página se crean automáticamente. Cada vez que se inserta una etiqueta o elemento en la página, se crea una nueva caja rectangular que encierra los contenidos del elemento.
CSS presenta todos los elementos del documento fuente en una caja rectangular de 4 zonas concéntricas que, de fuera a dentro, se denominan Margen, Borde, Relleno y Contenido.




El diseño de cualquier página XHTML está compuesto por cajas rectangulares. Las partes que componen cada caja y su orden de visualización desde el punto de vista del usuario son las siguientes:
▪ Contenido (content): se trata del contenido HTML del elemento (las palabras de un párrafo, una imagen, el texto de una lista de elementos, etc.)
▪ Relleno (padding): espacio libre opcional entre el contenido y el borde que lo encierra.
▪ Borde (border): línea que encierra completamente el contenido y su relleno.
▪ Imagen de fondo (background image): imagen que se muestra por detrás del contenido y el espacio de relleno.
▪ Color de fondo (background color): color que se muestra por detrás del contenido y el espacio de relleno.
▪ Margen (margin): espacio libre entre la caja y las posibles cajas adyacentes.

El color del contenido, el relleno y el borde será el mismo, el que hayamos establecido como color de fondo del elemento mediante la propiedad background-color. El color del margen será siempre transparente y no lo podemos modificar.

## 8.1 Anchura y altura

### Anchura

La propiedad CSS que controla la anchura de los elementos se denomina width.
Los valores de width pueden ser: <medida> | <porcentaje> | auto | inherit

Se aplica a Todos los elementos, salvo los elementos en línea que no sean imágenes, las filas de tabla y los grupos de filas de tabla . Valor inicial auto .

La propiedad width no admite valores negativos y los valores en porcentaje se calculan a partir de la anchura de su elemento padre.

- El valor inherit indica que la anchura del elemento se hereda de su

elemento padre.

- El valor auto, que es el que se utiliza si no se establece de forma explícita un valor a esta propiedad, indica que el navegador debe calcular automáticamente la anchura del elemento, teniendo en cuenta sus contenidos y el sitio disponible en la página.

### Altura

Controla la altura de los elementos se denomina height. Tiene los mismos valores que width.

## 8.2 Margen

| *PROPIEDAD* | *FUNCIÓN* | *VALOR* | *OBSERVACIONES* |
|---|---|---|---|
| **margin-top** | tamaño del margen superior | medida | Sin efecto en elementos de línea |
| **margin-bottom** | tamaño del margen inferior | medida | |
| **margin-right** | tamaño del margen derecho | medida | |
| **margin-left** | tamaño del margen izqdo | medida | |
| **margin** | tamaño del margen | de 1 a 4 medidas entre espacios | |

CSS define cuatro propiedades para controlar cada uno de los márgenes horizontales y verticales de un elemento: margin-top ,margin-right ,margin-bottom ,margin-left .

Sus valores son : <medida> | <porcentaje> | auto | inherit

Se aplica a todos los elementos, salvo margin-top y margin-bottom que sólo se aplican a los elementos de bloque y a las imágenes .

Valor inicial 0.

Las unidades más utilizadas para indicar los márgenes de un elemento son los píxeles (cuando se requiere una precisión total), los em (para hacer diseños que mantengan las proporciones) y los porcentajes (para hacer diseños líquidos o fluidos).

La propiedad que permite definir de forma simultanea los cuatro márgenes se denomina margin. (Este tipo de propiedades resumidas se denominan propiedades de tipo "shorthand").

| **Margin** | margen |
|---|---|
| valores | ( <medida> \| <porcentaje> \| auto ) {1, 4} \| inherit |
| Se aplica a | Todos los elementos salvo algunos casos especiales de elementos mostrados como tablas |
| Valor inicial | - |

La notación {1, 4} de la definición anterior significa que la propiedad margin admite entre uno y cuatro valores, con el siguiente significado:

▪ Si solo se indica un valor, todos los márgenes tienen ese valor.

▪ Si se indican dos valores, el primero se asigna al margen superior e inferior y el segundo se asigna a los márgenes izquierdo y derecho.

▪ Si se indican tres valores, el primero se asigna al margen superior, el tercero se asigna al margen inferior y el segundo valor se asigna los márgenes izquierdo y derecho.
▪ Si se indican los cuatro valores, el orden de asignación es: margen superior, margen derecho, margen inferior y margen izquierdo.

**Código CSS original:**

```
div img {
margin-top: .5em;
margin-bottom: .5em;
margin-left: 1em;
margin-right: .5em;
}
```

**Alternativa directa:**

```
div img {
margin: .5em .5em .5m 1em;
}
```

**Otra alternativa:**

```
div img {
margin: .5em;
margin-left: 1em;
}
```

# 8.3 Relleno

CSS define cuatro propiedades para controlar cada uno de los espacios de relleno horizontales y verticales de un elemento.

| | |
|---|---|
| padding-top<br>padding-right<br>padding-bottom<br>padding-left | Relleno superior<br>Relleno derecho<br>Relleno inferior<br>Relleno izquierdo |
| valores | <medida> \| <porcentaje> \| inherit |
| Se aplica a | Todos los elementos excepto algunos elementos de tablas como grupos de cabeceras y grupos de pies de tabla |
| Valor inicial | 0 |

CSS también define una propiedad de tipo "shorthand" para establecer los cuatro rellenos de un elemento de forma directa. La propiedad que permite definir de forma simultánea los cuatro márgenes se denomina padding.

| | |
|---|---|
| padding | Relleno |
| valores | <medida> \| <porcentaje> \| inherit |
| Se aplica a | Todos los elementos excepto algunos elementos de tablas como grupos de cabeceras y grupos de pies de tabla |
| Valor inicial | - |

## 8.4 Borde

Para cada borde se puede establecer su anchura, su color y su estilo.

1. Anchura:

| | |
|---|---|
| border-top-width<br>border-right-width<br>border-bottom-width<br>border-left-width | Anchura del borde superior<br>Anchura del borde derecho<br>Anchura del borde inferior<br>Anchura del borde izquierdo |
| valores | ( <medida> \| thin \| medium \| thick ) \| inherit |
| se aplica a | Todos los elementos |
| valor inicial | medium |

La medida más habitual para indicar la anchura de los bordes es el píxel .
ej:
div {
border-top-width: 10px;
border-right-width: 1em;
border-bottom-width: thick;
border-left-width: thin;
}

Si se quiere establecer la misma anchura a todos los bordes :

| | |
|---|---|
| border-width | Anchura del borde |
| valores | ( <medida> \| thin \| medium \| thick ) {1,4}\| inherit |
| se aplica a | Todos los elementos |
| valor inicial | medium |

Si se indica un solo valor, se aplica a los cuatro bordes. Si se indican dos valores, el primero se aplica al borde superior e inferior y el segundo valor se aplica al borde izquierdo y derecho.
Si se indican tres valores, el primero se aplica al borde superior, el segundo se aplica al borde izquierdo y derecho y el tercer valor se aplica al borde inferior. Si se indican los cuatro valores, el orden de aplicación es superior, derecho, inferior e izquierdo.

2. Color:

| | |
|---|---|
| border-top-color<br>border-right-color<br>border-bottom-color<br>border-left-color | color del borde superior<br>color del borde derecho<br>color del borde inferior<br>color del borde izquierdo |
| valores | <color> \| transparent \| inherit |
| se aplica a | Todos los elementos |

| | |
|---|---|
| valor inicial | - |

Ej: div {
border-top-color: #CC0000;
border-right-color: blue;
border-bottom-color: #00FF00;
border-left-color: #CCC;
}

Si se quiere establecer el mismo color para todos los bordes :
**border-color** cuyos valores son:
( <color> | transparent ) {1, 4} | inherit

3. Estilo:

| | |
|---|---|
| border-top-style<br>border-right-style<br>border-bottom-style<br>border-left-style | estilo del borde superior<br>estilo del borde derecho<br>estilo del borde inferior<br>estilo del borde izquierdo |
| valores | none \| hidden \| dotted \| dashed \| solid \| double \| groove \| ridge \| inset \| outset \| inherit |
| se aplica a | Todos los elementos |
| valor inicial | none |

ej:
div {
border-top-style: dashed;
border-right-style: double;
border-bottom-style: dotted;
border-left-style: solid;
}

Si se quiere establecer el mismo estilo para todos los bordes : **border-style** cuyos valores pueden ser: (none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset ) {1, 4} | inherit

- CSS define una serie de propiedades de tipo "shorthand" que permiten establecer todos los atributos de los bordes de forma directa.

Antes de presentar las propiedades, es conveniente definir los tres siguientes tipos de valores:
<medida_borde> = <medida> | thin | medium | thick
<color_borde> = <color> | transparent
<estilo_borde> = none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset

| | |
|---|---|
| border-top<br>border-right | estilo completo del borde superior<br>estilo completo del borde derecho |

| | |
|---|---|
| border-bottom<br>border-left | estilo completo del borde inferior<br>estilo completo del borde izquierdo |
| valores | ( <medida_borde> \|\| <color_borde> \|\| <estilo_borde> ) \| inherit |
| se aplica a | Todos los elementos |
| valor inicial | - |

ejemplo: el valor del ancho se queda al por defecto:medium

```
h1 {
border-bottom: solid red;
}
```

otro ejemplo:

```
div {
border-top: 1px solid #369;
border-bottom: 3px double #369;
}
```

CSS define una propiedad de tipo "shorthand" global para establecer el valor de todos los atributos de todos los bordes de forma directa : border

| | |
|---|---|
| border | estilo completo del borde |
| valores | ( <medida_borde> \|\| <color_borde> \|\| <estilo_borde> ) \| inherit |
| se aplica a | Todos los elementos |
| valor inicial | - |

Cuando los cuatro bordes no son idénticos pero sí muy parecidos, se puede utilizar la propiedad border para establecer de forma directa los atributos comunes de todos los bordes y posteriormente especificar para cada uno de los cuatro bordes sus propiedades particulares:

```
h1 {
border: solid #000;
border-top-width: 6px;
border-left-width: 8px;
}
```

# 9 Selectores (II)

## 9.1 Selectores de atributo

Los selectores de atributo permiten seleccionar elementos del documento fuente en función de sus atributos o los valores de estos.

SS1[atrib] Los SS1 que tengan el atributo atrib
SS1[atrib=valor] Los SS1 que tengan el atributo atrib con el valor indicado
SS1[atr~=val] Los SS1 con el atributo atr y valor una lista entre espacios que incluye val
SS1 [atr|=val] Los SS1 con atributo atr y valor que empieza por val- ó valor 'val'

## 9.2 Pseudo-elementos

Permiten apuntar a partes no estructuradas del documento fuente

| | |
|---|---|
| SS1:first-line | La primera línea de los SS1 |
| SS1:first-letter | La primera letra de los SS1 |

SS1:before añade contenido al principio de los SS1. El contenido a añadir será el valor de la propiedad content
SS1:after añade contenido al final de los SS1. El contenido a añadir será el valor de la propiedad content

## 9.3 Pseudo-clases

Permiten elegir elementos según características que no se deducen de la estructura del documento fuente, sino de las acciones del usuario.
A:link Afecta a los enlaces que aún no han sido visitados por el usuario
A:visited Afecta a los enlaces que ya han sido visitados por el usuario
SS1:hover Afecta a los SS1 mientras se están señalando con un dispositivo apuntador
SS1:active Afecta a los SS1 mientras están activos (entre click y soltar)
SS1:focus Afecta a los SS1 mientras tienen el foco

# 10. Control de flujo de cajas

Los elementos de una página web están contenidos en una caja rectangular, de acuerdo con el modelo de caja. La manera en que los diferentes elementos de una página web se distribuyen en la pantalla dependen del esquema de posicionamiento elegido.

Existen tres esquemas de posicionamiento:

- normal,
- flotante y
- absoluto

1. El ***esquema normal o estático*** es el que se aplica por omisión. Los elementos se muestran en la pantalla en el mismo orden que se encuentran en el código fuente de la página. En este modelo, ninguna caja se desplaza respecto de su posición original, por lo que sólo se tiene en cuenta si el elemento es de bloque o en línea.

   Los elementos pueden ser elementos de bloque o elementos en línea:

   Los elementos en línea:

   - no empiezan necesariamente en nueva línea y sólo ocupan el espacio necesario para mostrar sus contenidos.
   - están contenidos dentro de elementos de bloque y
   - un elemento en línea puede aparecer tanto dentro de un elemento de bloque como dentro de otro elemento en línea.
   - Los elementos en línea definidos por HTML son: a, abbr, acronym, b, basefont, bdo, big, br, cite, code, dfn, em, font, i, img, input, kbd, label, q, s, samp, select, small, span, strike, strong, sub, sup, textarea, tt, u, var.

   Los elementos de bloque:

   - siempre empiezan en una nueva línea y ocupan todo el espacio disponible hasta el final de la línea.
   - ocupan todo el ancho de la página y la altura necesaria para mostrar todo su contenido.
   - No pueden insertarse dentro de elementos en línea y tan sólo pueden aparecer dentro de otros elementos de bloque.
   - Los elementos de bloque definidos por HTML son: address, blockquote, center, dir, div, dl, fieldset, form, h1, h2, h3, h4, h5, h6, hr, isindex, menu, noframes, noscript, ol, p, pre, table, ul.
   - Los siguientes elementos también se considera que son de bloque: dd, dt, frameset, li, tbody, td, tfoot, th, thead, tr.

   Los siguientes elementos pueden ser en línea y de bloque según las circunstancias: button, del, iframe, ins, map, object, script.

Los elementos de bloque forman lo que CSS denomina "contextos de formato de bloque". En este tipo de contextos, las cajas se muestran una debajo de otra comenzando desde el principio del elemento contenedor. La distancia entre las cajas se controla mediante los márgenes verticales.

Si un elemento se encuentra dentro de otro, el elemento padre se llama "elemento contenedor" y determina tanto la posición como el tamaño de todas sus cajas interiores.

Si un elemento no se encuentra dentro de un elemento contenedor, entonces su elemento contenedor es el elemento <body> de la página. Normalmente, la anchura de los elementos de bloque está limitada a la anchura de su elemento contenedor, aunque en algunos casos sus contenidos pueden desbordar el espacio disponible.

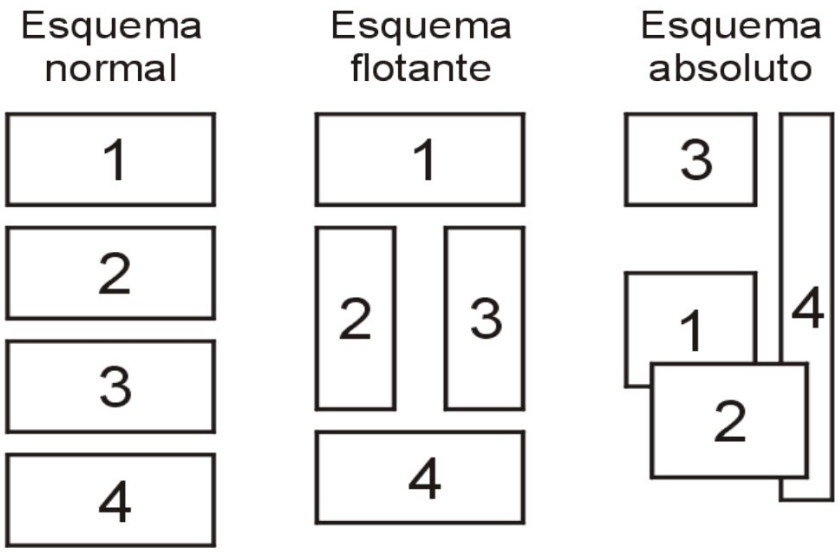
Los elementos en línea forman los "contextos de formato en línea". En este tipo de contextos, las cajas se muestran una detrás de otra de forma horizontal comenzando desde la posición más a la izquierda de su elemento contenedor. La distancia entre las cajas se controla mediante los márgenes laterales. Si las cajas en línea ocupan más espacio del disponible en su propia línea, el resto de cajas se muestran en las líneas inferiores. Si las cajas en línea ocupan un espacio menor que su propia línea, se puede controlar la distribución de las cajas mediante la propiedad *text-align* para centrarlas, alinearlas a la derecha o justificarlas.

2. El ***esquema absoluto*** permite asignar cualquier posición a un elemento en la página . En el esquema de posicionamiento absoluto, los elementos pueden mostrarse en la pantalla en un orden diferente al que tienen en el código fuente, e incluso superponerse. Para asignar el esquema de posicionamiento absoluto a un elemento se utiliza la propiedad ***position***.

3. El ***esquema flotante*** es el que permite que un elemento no ocupe todo el ancho de la pantalla y se muestre a la izquierda o a la derecha de la página con otros elementos a su lado. En el esquema flotante, los elementos se muestran en la pantalla en el mismo orden en que se encuentran en el código fuente, pero pueden estar desplazados a derecha a izquierda. Para asignar el esquema de posicionamiento flotante a un elemento se utiliza la propiedad ***float***.

**Ejemplo:** La imagen siguiente muestra cómo podrían mostrarse cuatro bloques de una página que en el código fuente estuvieran seguidos.

- En el esquema normal los bloques se muestran en el mismo orden que en el código fuente.
- en el esquema flotante se puede colocar un bloque a la izquierda (el bloque 2) y el bloque siguiente se puede colocar a la derecha del flotante
- en el esquema absoluto los bloques se pueden colocar en cualquier orden y posición.




# 10.1 Esquema absoluto

La propiedad **position** establece el esquema de posicionamiento absoluto de un elemento. Existen cuatro tipos en el esquema de posicionamiento absoluto:

- static: posicionamiento absoluto estático
- relative: posicionamiento absoluto relativo
- fixed: posicionamiento absoluto fijo
- absolute: posicionamiento absoluto absoluto

| **position** | posicionamiento |
|---|---|
| valores | Static,relative, fixed, absolute, inherit |
| Valor inicial | Static |

| | |
|---|---|
| descripción | Selecciona el posicionamiento con el que se mostrará el elemento |

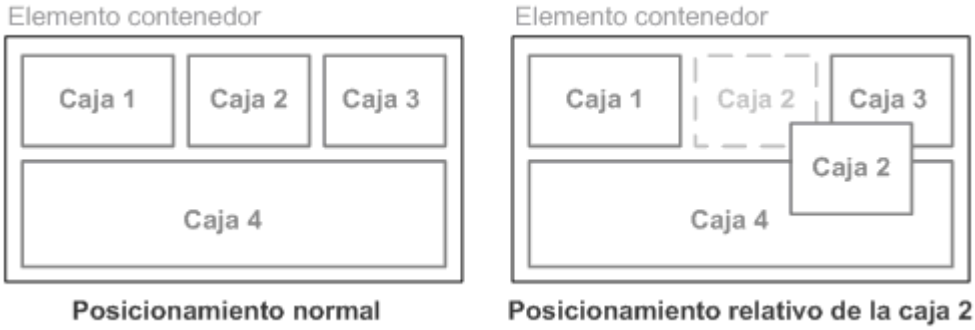
### 10.1.1 Posicionamiento absoluto estático: static

Cuando se establece la propiedad ***position*** como **static**, el elemento se sitúa en la posición que le corresponde de acuerdo con el flujo normal del documento. Es decir, el posicionamiento absoluto estático es equivalente al esquema normal. Si se utiliza este valor, se ignoran los valores de las propiedades top, right, bottom y left que se verán más adelante. ( es el valor por defecto de position luego no es necesario indicarla).

### 10.1.2 Posicionamiento absoluto relativo: relative

Cuando se establece la propiedad position como relative, el navegador reserva el espacio en la página que tendría el elemento de acuerdo con el flujo normal del documento, pero el elemento se puede desplazar con respecto a esa posición. Es decir, el "hueco" de la posición original se mantiene, aunque el elemento se haya desplazado.

Es decir: El posicionamiento relativo permite desplazar una caja respecto de su posición original establecida mediante el posicionamiento normal. El desplazamiento de la caja se controla con las propiedades top, right, bottom y left. El desplazamiento de una caja no afecta al resto de cajas adyacentes, que se muestran en la misma posición que si la caja desplazada no se hubiera movido de su posición original.




El desplazamiento relativo se establece mediante las propiedades **top, bottom, left y right**.

El **desplazamiento vertical** se puede establecer tanto con la propiedad top como con la propiedad bottom:

- La propiedad **top** desplaza el elemento contando desde su borde superior. Si se da un valor positivo, el elemento se desplaza hacia abajo. Si se da un valor negativo, el elemento se desplaza hacia arriba.

- La propiedad **bottom** desplaza el elemento contando desde su borde inferior. Si se da un valor positivo, el elemento se desplaza hacia arriba. Si se da un valor negativo, el elemento se desplaza hacia abajo.

El **desplazamiento horizontal** se puede establecer tanto con la propiedad left como con la propiedad right:

- La propiedad **left** desplaza el elemento contando desde su borde izquierdo. Si se da un valor positivo, el elemento se desplaza hacia la derecha. Si se da un valor negativo, el elemento se desplaza hacia la izquierda.
- La propiedad **right** desplaza el elemento contando desde su borde derecho. Si se da un valor positivo, el elemento se desplaza hacia la izquierda. Si se da un valor negativo, el elemento se desplaza hacia la derecha.

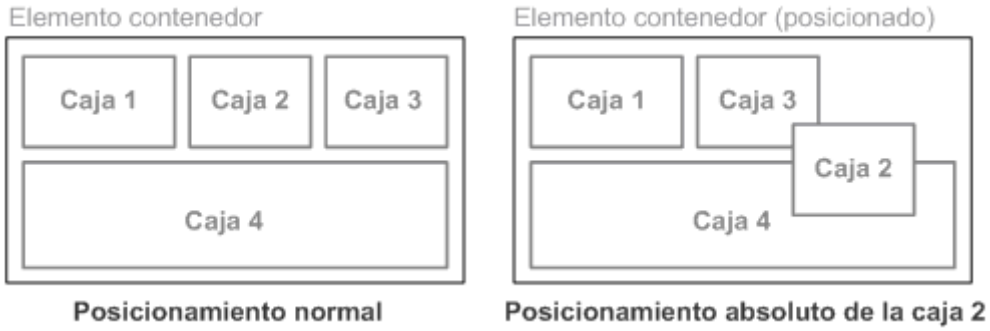
*(hacer ejercicio 4)*

### 10.1.3 Posicionamiento absoluto absoluto: absolute

El desplazamiento de la caja también se controla con las propiedades top, right, bottom y left, pero su interpretación es mucho más compleja, ya que el origen de coordenadas del desplazamiento depende del posicionamiento de su elemento contenedor.

Cuando una caja se posiciona de forma absoluta, el resto de elementos de la página la ignoran y ocupan el lugar original ocupado por la caja posicionada. Al igual que en el posicionamiento relativo, cuando se posiciona de forma absoluta una caja es probable que se produzcan solapamientos con otras cajas.

En el siguiente ejemplo, se posiciona de forma absoluta la caja 2:




La caja 2 está posicionada de forma absoluta, lo que implica que el resto de elementos ignoran que esa caja exista. Por este motivo, la caja 3 deja su lugar original y pasa a ocupar el hueco dejado por la caja 2.

Por otra parte, el desplazamiento de una caja posicionada de forma absoluta se indica mediante las propiedades top, right, bottom y left. A diferencia de posicionamiento relativo, en este caso la referencia de los valores de esas propiedades es el origen de coordenadas de su primer elemento contenedor posicionado.

Determinar el origen de coordenadas a partir del cual se desplaza una caja posicionada de forma absoluta es un proceso complejo que se compone de los siguientes pasos:

- Se buscan todos los elementos contenedores de la caja hasta llegar al elemento <body> de la página.
- Se recorren todos los elementos contenedores empezando por el más cercano a la caja y llegando hasta el <body>
- De todos ellos, el navegador se queda con el primer elemento contenedor que esté posicionado de cualquier forma diferente a position: static
- La esquina superior izquierda de ese elemento contenedor posicionado es el origen de coordenadas.

Una vez obtenido el origen de coordenadas, se interpretan los valores de las propiedades top, right,

bottom y left respecto a ese origen y se desplaza la caja hasta su nueva posición Cuando se establece la propiedad **position** como **absolute**, el elemento se sitúa en la posición exacta que se le indique. Pero si la página es larga y se puede desplazar verticalmente, el elemento absoluto se desplazará con el resto de elementos.

El navegador no tiene en cuenta la posición de los elementos con posicionamiento absoluto absoluto al mostrar el resto de elementos de la página, es decir, que los elementos con posicionamiento absoluto absoluto no reservan espacio.

*Si se quiere posicionar un elemento de forma absoluta respecto de su elemento contenedor, es imprescindible posicionar el elemento contenedor.* Para ello, sólo es necesario añadir la propiedad position: relative, por lo que no es obligatorio desplazar el elemento contenedor respecto de su posición original.

## 10.1.4 Posicionamiento absoluto fijo: fixed

El estándar CSS considera que el posicionamiento fijo es un caso particular del posicionamiento absoluto, ya que sólo se diferencian en el comportamiento de las cajas posicionadas.

Cuando una caja se posiciona de forma fija, la forma de obtener el origen de coordenadas para interpretar su desplazamiento es idéntica al posicionamiento absoluto. De hecho, si el usuario no mueve la página HTML en la ventana del navegador, no existe ninguna diferencia entre estos dos modelos de posicionamiento.

La principal característica de una caja posicionada de forma fija es que su posición es inamovible dentro de la ventana del navegador. El posicionamiento fijo hace que las cajas no modifiquen su posición ni aunque el usuario suba o baje la página en la ventana de su navegador.

Si la página se visualiza en un medio paginado (por ejemplo en una impresora) las cajas posicionadas de forma fija se repiten en todas las páginas. Esta característica puede ser útil para crear encabezados o pies de página en páginas HTML preparadas para imprimir.

El posicionamiento fijo apenas se ha utilizado en el diseño de páginas web hasta hace poco tiempo porque el navegador Internet Explorer 6 y las versiones anteriores no lo soportan.

Cuando se establece la propiedad **position** como **fixed**, el elemento se sitúa en la posición indicada y no se puede desplazar. Aunque la página sea larga y se pueda desplazar verticalmente, el elemento fijo estará siempre en la misma posición de la pantalla.

El navegador no tiene en cuenta la posición de los elementos con posicionamiento absoluto fijo al mostrar el resto de elementos de la página, es decir, que los elementos con posicionamiento absoluto fijo no reservan espacio.

**Nota**: Internet Explorer 7 no aplicaba correctamente el posicionamiento fijo (position: fixed) a una tabla..

**Nota**: Internet Explorer 6 no era capaz de representar este tipo de posicionamiento.

## 10.1.5 Profundidad: z-index

Al utilizar posicionamiento absoluto (relativo, fijo o absoluto) dos o más elementos se pueden solapar. La propiedad z-index, que debe ser un número entero (positivo, cero o negativo), permite establecer qué elemento se muestra encima, pero también influye si los elementos tienen o no posicionamiento y el orden en que aparecen los elementos en el código fuente.

Las reglas a aplicar son las siguientes:

- Los elementos sin propiedad z-index se tratan como si tuviera un z-index igual a 0.
- Los elementos sin posicionamiento se tratan como si tuvieran un z-index igual a 0 (aunque tengan propiedad z-index).
- Los elementos con mayor z-index se colocan sobre los elementos con menor z-index,
- Si los z-index son iguales, los elementos con posicionamiento se colocan sobre los elementos sin posicionamiento
- Si los z-index son iguales y los elementos tienen posicionamiento, los elementos que aparecen después en el código fuente se colocan encima de los elementos que aparecen antes en el código fuente

# 10.2 Esquema flotante

## 10.2.1. Posicionamiento flotante: float y clear

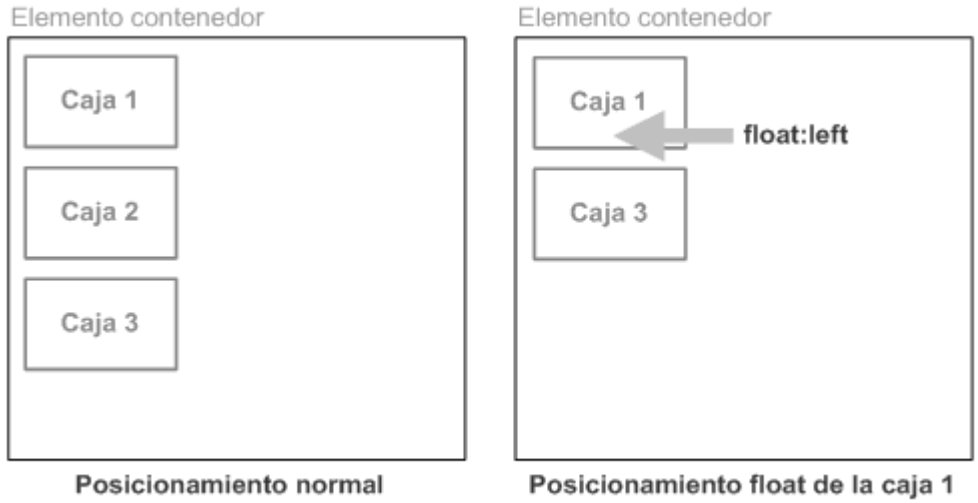
Cuando una caja se posiciona con el modelo de posicionamiento flotante, automáticamente se convierte en una caja flotante, lo que significa que se desplaza hasta la zona más a la izquierda o más a la derecha de la posición en la que originalmente se encontraba.

Cuando se posiciona una caja de forma flotante:

▪ La caja deja de pertenecer al flujo normal de la página, lo que significa que el resto de cajas ocupan el lugar dejado por la caja flotante.

▪ La caja flotante se posiciona lo más a la izquierda o lo más a la derecha posible de la posición en la que se encontraba originalmente.

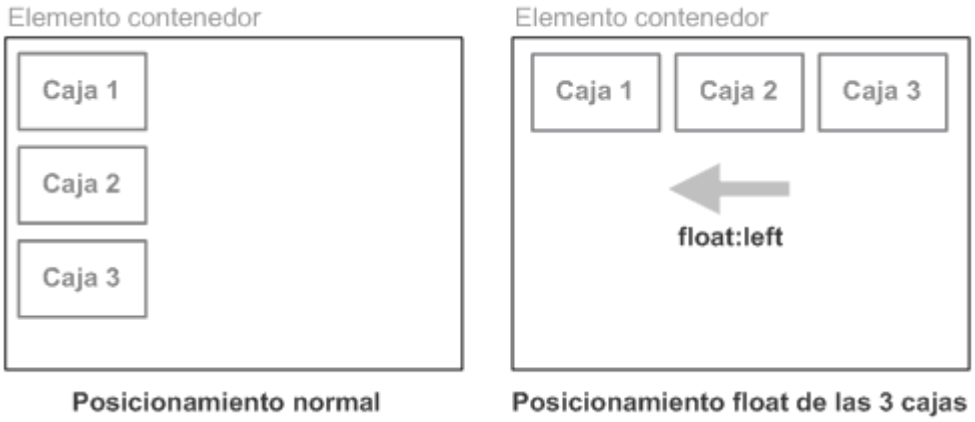
Si en el anterior ejemplo la caja 1 se posiciona de forma flotante hacia la izquierda, el resultado es el que muestra la siguiente imagen:




Si existen otras cajas flotantes, al posicionar de forma flotante otra caja, se tiene en cuenta el sitio disponible.

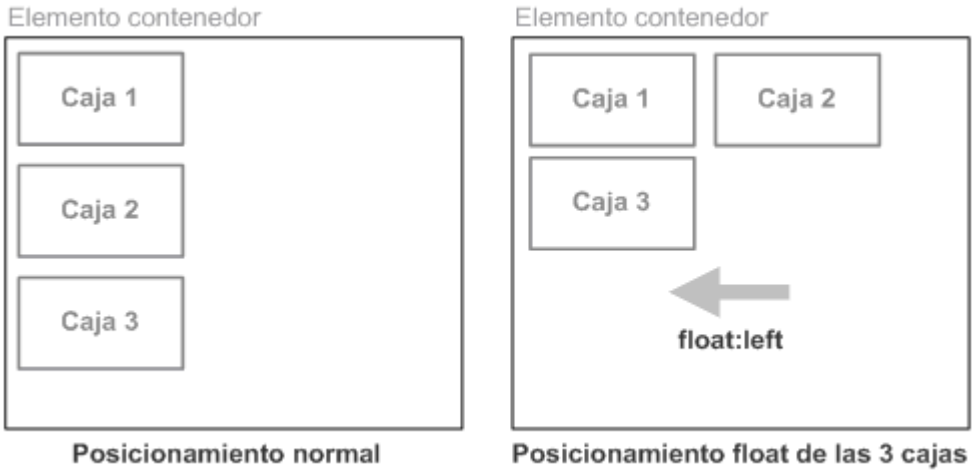
La propiedad **float** establece el esquema de posicionamiento flotante para un elemento. Cuando existe un elemento flotante, los elementos que se encuentran a continuación del elemento flotante fluyen a lo largo de él, salvo que haya un elemento que tenga establecido la propiedad clear.

Las propiedades float y clear se pueden aplicar a cualquier elemento de una página web.




En el ejemplo anterior, las cajas no se superponen entre sí porque las cajas flotantes tienen en cuenta las otras cajas flotantes existentes. Como la caja 1 ya estaba posicionada lo más a la izquierda posible, la caja 2 sólo puede colocarse al lado del borde derecho de la caja 1, que es el sitio más a la izquierda posible respecto de la zona en la que se encontraba.

Si no existe sitio en la línea actual, la caja flotante baja a la línea inferior hasta que encuentra el sitio necesario para mostrarse lo más a la izquierda o lo más a la derecha posible en esa nueva línea:




Las cajas flotantes influyen en la disposición de todas las demás cajas. Los elementos en línea hacen sitio a las cajas flotantes adaptando su anchura al espacio libre dejado por la caja desplazada. Los elementos de bloque no les hacen sitio, pero sí que adaptan sus contenidos para que no se solapen con las cajas flotantes.

Si se indica un valor **left**, la caja se desplaza hasta el punto más a la izquierda posible en esa misma línea (si no existe sitio en esa línea, la caja baja una línea y se muestra lo más a la izquierda posible en esa nueva línea). El resto de elementos adyacentes se adaptan y fluyen alrededor de la caja flotante.

El valor **right** tiene un funcionamiento idéntico, salvo que en este caso, la caja se desplaza hacia la derecha.

El valor **none** permite anular el posicionamiento flotante de forma que el elemento se muestre en su

posición original.

Los elementos que se encuentran alrededor de una caja flotante adaptan sus contenidos para que fluyan alrededor del elemento posicionado .

La propiedad **clear** permite modificar el comportamiento por defecto del posicionamiento flotante para forzar a un elemento a mostrarse debajo de cualquier caja flotante.

## 10.2.2 Posicionamiento flotante de caracteres: letra capital

Se puede crear una letra capital aplicando la propiedad float a la primera letra de un párrafo. En el ejemplo siguiente se utiliza la pseudo-clase :first-letter, por lo que no hace falta aplicar la propiedad clear.

**Nota**: Firefox no muestra la página igual que Internet Explorer y Chrome. La primera letra se ve más pequeña.

## 10.2.3 Posicionamiento flotante de imágenes

### La propiedad float

Las imágenes son elementos en línea, es decir, que se insertan como si fueran caracteres, formando parte del párrafo o del elemento de bloque en el que se insertan. La altura de la línea en la que está insertado el elemento aumenta lo necesario para poder alojar la imagen.

Si se quiere que una imagen aparezca a la izquierda (o a la derecha) de un texto, es decir, que el texto fluya a lo largo de la imagen, hay que utilizar la propiedad **float**. Esta propiedad sólo admite dos valores, **left y right**, que sitúan la imagen a la izquierda o a la derecha.

Para que las imágenes salgan correctamente alineadas con el texto, la imagen debe insertarse al principio del texto, independientemente de la posición final que vaya a tener la imagen.

Ejemplo: Las dos imágenes están insertadas antes del texto:

```
<p><img izquierda /><img derecha />Texto</p>
```

# 11. Visualización en el modelo de cajas

CSS define otras cuatro propiedades para controlar su visualización: display, visibility, overflow y z-index.

# 11.1 Visibility y display

- La propiedad **display** permite ocultar completamente un elemento haciendo que desaparezca de la página. Como el elemento oculto no se muestra, el resto de elementos de la página se mueven para ocupar su lugar.

| **Display** | Visualización de un elemento |
|---|---|
| valores | inline \| block \| none \| list-item \| run-in \| inline-block \| table \| inline-table \| table-row-group \| table-header-group \| table-footer-group \| table-row \| table-column-group \| table-column \| table-cell \| table-caption \| inherit |
| Se aplica a | Todos los elementos |
| Valor inicial | Inline |
| descripción | Permite controlar la forma de visualizar un elemento e incluso ocultarlo |

La propiedad display modifica la forma en la que se visualiza un elemento.

Los valores más utilizados son inline, block y none.

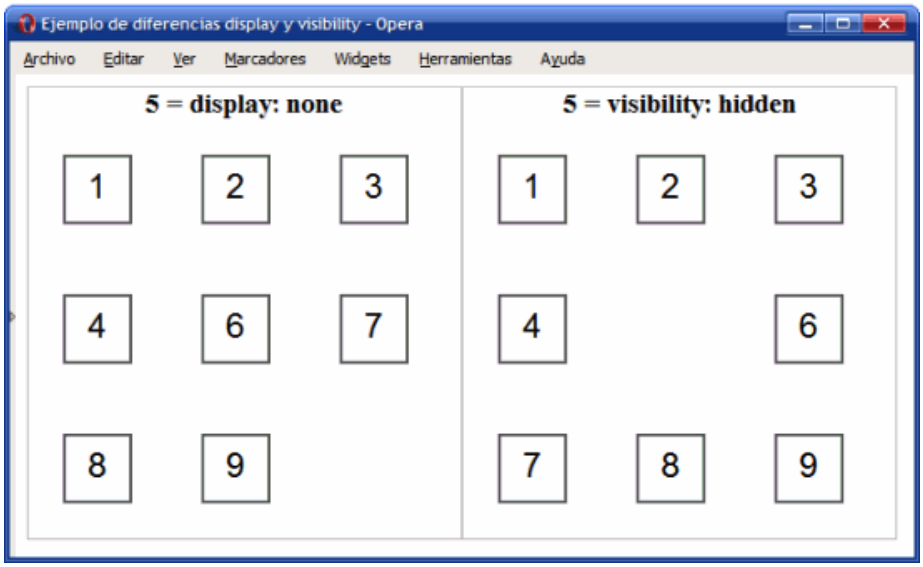
- ✔ El valor block muestra un elemento como si fuera un elemento de bloque, independientemente del tipo de elemento que se trate.
- ✔ El valor inline visualiza un elemento en forma de elemento en línea, independientemente del tipo de elemento que se trate.
- ✔ El valor none oculta un elemento y hace que desaparezca de la página.

- Por otra parte, la propiedad **visibility** permite hacer invisible un elemento, lo que significa que el navegador crea la caja del elemento pero no la muestra. En este caso, el resto de elementos de la página no modifican su posición, ya que aunque la caja no se ve, sigue ocupando sitio.

| **visibility** | Visibilidad de un elemento |
|---|---|
| valores | visible \| hidden \| collapse \| inherit |
| Se aplica a | Todos los elementos |
| Valor inicial | visible |
| descripción | Permite hacer visibles e invisibles a los elementos |

Inicialmente todas las cajas que componen la página son visibles.

- ✔ Empleando el valor **hidden** es posible convertir una caja en invisible para que no muestre sus contenidos. El resto de elementos de la página se muestran como si la caja todavía fuera visible, por lo que en el lugar donde originalmente se mostraba la caja invisible, ahora se muestra un hueco vacío.
- ✔ El valor **collapse** sólo se puede utilizar en las filas, grupos de filas, columnas y grupos de columnas de una tabla. Su efecto es similar al de la propiedad display, ya que oculta completamente la fila y/o columna y se pueden mostrar otros contenidos en ese lugar. Si se utiliza el valor collapse sobre cualquier otro tipo de elemento, su efecto es idéntico al valor hidden.




# 11.2. Relación entre display, float y position

Cuando se establecen las propiedades display, float y position sobre una misma caja, su interpretación es la siguiente:

1. Si display vale none, se ignoran las propiedades float y position y la caja no se muestra en la página.

2. Si position vale absolute o fixed, la caja se posiciona de forma absoluta, se considera que float vale none y la propiedad display vale block tanto para los elementos en línea como para los elementos de bloque. La posición de la caja se determina mediante el valor de las propiedades top, right, bottom y left.

3. En cualquier otro caso, si float tiene un valor distinto de none, la caja se posiciona de forma flotante y la propiedad display vale block tanto para los elementos en línea como para los elementos de bloque.

# 11.3. Propiedad overflow

En algunas ocasiones el contenido de un elemento no cabe en el espacio reservado para ese elemento y se desborda. (esto ocurre por ejemplo cuando se establece la anchura y/o altura de un elemento mediante la propiedad width y/o height. Otra situación habitual es la de las líneas muy largas contenidas dentro de un elemento <pre>, que hacen que la página entera sea demasiado ancha.

CSS define la propiedad overflow para controlar la forma en la que se visualizan los contenidos que sobresalen de sus elementos.

| **overflow** | Parte sobrante de un elemento |
|---|---|
| valores | visible \| hidden \| scroll \| auto \| inherit |
| Se aplica | Elementos de bloque y celdas de tablas |
| Valor inicial | Visible |
| Descripción | Permite controlar los contenidos sobrantes de un elemento |

Los valores de la propiedad overflow tienen el siguiente significado:

- visible: el contenido no se corta y se muestra sobresaliendo la zona reservada para visualizar el elemento. Este es el comportamiento por defecto.
- hidden: el contenido sobrante se oculta y sólo se visualiza la parte del contenido que cabe dentro de la zona reservada para el elemento.
- scroll: solamente se visualiza el contenido que cabe dentro de la zona reservada para el elemento, pero también se muestran barras de scroll que permiten visualizar el resto del contenido.
- auto: el comportamiento depende del navegador, aunque normalmente es el mismo que la propiedad scroll.

# 11.4. Propiedad z-index

Además de posicionar una caja de forma horizontal y vertical, CSS permite controlar la posición tridimensional de las cajas posicionadas. De esta forma, es posible indicar las cajas que se muestran delante o detrás de otras cajas cuando se producen solapamientos.

La posición tridimensional de un elemento se establece sobre un tercer eje llamado Z y se controla mediante la propiedad z-index. Utilizando esta propiedad es posible crear páginas complejas con varios niveles o capas.

| z-index | Orden tridimensional |
|---|---|
| valores | auto \| <numero> \| inherit |
| Se aplica | Elementos que han sido posicionados explícitamente |
| Valor inicial | Auto |
| Descripción | Establece el nivel tridimensional en el que se muestra el elemento |

El valor más común de la propiedad z-index es un número entero. Aunque la especificación oficial permite los números negativos, en general se considera el número 0 como el nivel más bajo.

Cuanto más alto sea el valor numérico, más cerca del usuario se muestra la caja. Un elemento con z-index: 10 se muestra por encima de los elementos con z-index: 8 o z-index: 9, pero por debajo de elementos con z-index: 20 o z-index: 50.

La propiedad z-index sólo tiene efecto en los elementos posicionados, por lo que es obligatorio que la propiedad z-index vaya acompañada de la propiedad position. Si debes posicionar un elemento pero no quieres moverlo de su posición original ni afectar al resto de elementos de la página, puedes utilizar el posicionamiento relativo (position: relative).

## 12.Layouts

Este apartado refleja cómo crear algunas de las estructuras o layouts más habituales de los diseños web utilizando exclusivamente CSS.

# 12.1 Centrar una página horizontalmente

A medida que aumenta el tamaño y la resolución de la pantalla, se hace más difícil diseñar páginas que se adapten al tamaño de la ventana del navegador. Por otra parte, los navegadores alinean por defecto las páginas web a la izquierda de la ventana. Cuando la resolución de la pantalla es muy grande, la mayoría de páginas de anchura fija alineadas a la izquierda parecen muy estrechas y provocan una sensación de vacío.

La solución más sencilla para evitar los grandes espacios en blanco consiste en crear páginas con una anchura fija adecuada y centrar la página horizontalmente respecto de la ventana del navegador.

En CSS sería: agrupar todos los contenidos de la página en un elemento  y asignarle a ese  unos márgenes laterales automáticos. El  que encierra los contenidos se suele llamar contenedor (wrapper o container):

```
#contenedor {
        width: 300px;
        margin: 0 auto;
        }
```

```
<body>
<div id="contenedor">
<h1>Centrar la página</h1>
...
</div>
</body>
```

Los márgenes superior e inferior son iguales a 0 y los márgenes laterales toman un valor de auto. Cuando se asignan márgenes laterales automáticos a un elemento, los navegadores centran ese elemento respecto de su elemento padre. En este ejemplo, el elemento padre del  es la propia página (el elemento <body>), por lo que se consigue centrar el elemento  respecto de la ventana del navegador.

Modificando ligeramente el código CSS anterior se puede conseguir un diseño fluido que se adapta a la anchura de la ventana del navegador y permanece siempre centrado:

```
#contenedor {
width: 70%;
margin: 0 auto;
}
```

Estableciendo la anchura del elemento contenedor mediante un porcentaje, su anchura se adapta de forma continua a la anchura de la ventana del navegador. De esta forma, si se reduce la anchura de la ventana del navegador, la página se verá más estrecha y si se maximiza la ventana del navegador, la página se verá más ancha.

## 12.2. Centrar una página verticalmente

Cuando se centra una página web verticalmente, el objetivo es que sus contenidos aparezcan en el centro de la ventana del navegador y por tanto, que sus márgenes verticales se adapten de forma dinámica en función del tamaño de la ventana del navegador.

A continuación se muestra la forma de centrar una página web respecto de la ventana del navegador, es decir, centrarla tanto horizontalmente como verticalmente.

La solución correcta para centrar verticalmente una página web se basa en el posicionamiento absoluto e implica realizar un cálculo matemático. A continuación se muestran los cuatro pasos necesarios para centrar una página web en la ventana del navegador:

1. En primer lugar, se asigna una altura y una anchura al elemento que encierra todos los contenidos de la página.

   ```
   #contenedor {
       width: 500px;
       height: 500px;
   }
   ```

2. A continuación, se posiciona de forma absoluta el elemento contenedor y se asigna un valor de 50% tanto a la propiedad top como a la propiedad left. El resultado es que la esquina superior izquierda del elemento contenedor se posiciona en el centro de la ventana del navegador:

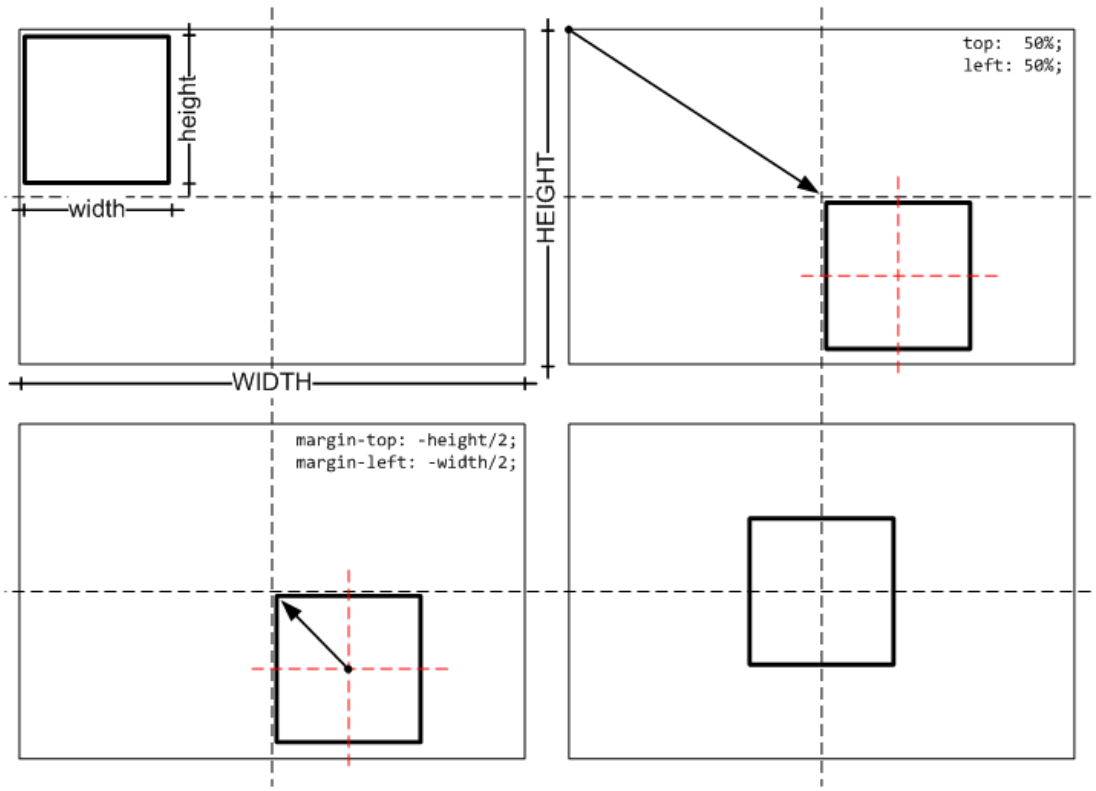
```
#contenedor {
        width: 500px;
        height: 500px;
        position: absolute;
        top: 50%;
        left: 50%;
        }
```

3. Como se desprende de la imagen, la página web debe moverse hacia arriba una cantidad igual a la mitad de su altura y debe desplazarse hacia la izquierda una cantidad equivalente a la mitad de su anchura. Utilizando las propiedades margin-top y margin-left con valores negativos, la página se desplaza hasta el centro de la ventana del navegador.

```
#contenedor {
        width: 500px;
        height: 500px;
        position: absolute;
        top: 50%;
        left: 50%;
        margin-top: -250px; /* height/2 = 500px / 2 */
        margin-left: -250px; /* width/2 = 500px / 2 */
        }
```

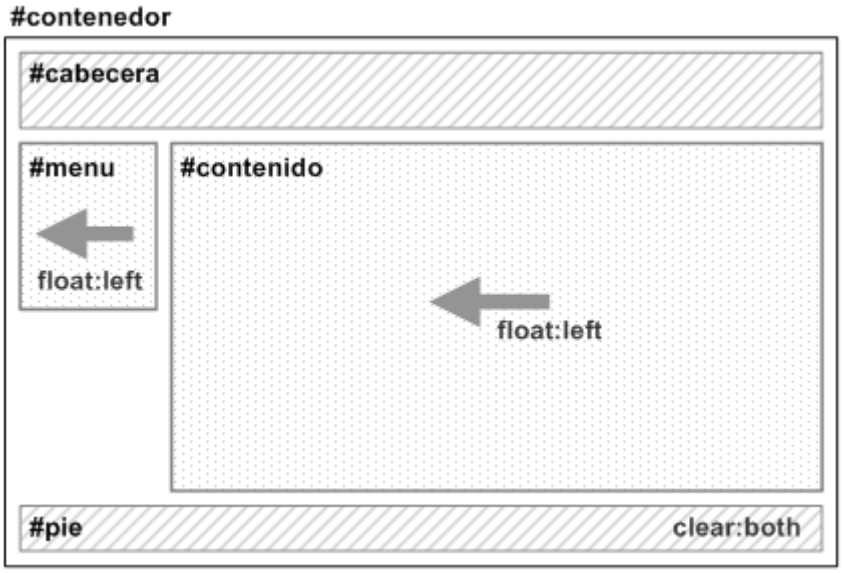
4. Para centrar una página sólo verticalmente, se debe prescindir tanto del posicionamiento horizontal como del desplazamiento horizontal:

```
#contenedor {
        width: 500px;
        height: 500px;
        position: absolute;
        top: 50%;
        margin-top: -250px;  /* height/2 = 500px / 2 */
}
```




## 12.3 Diseño a 2 columnas con cabecera y pie de página

La solución CSS se basa en el uso de la propiedad float para los elementos posicionados como el menú y los contenidos y el uso de la propiedad clear en el pie de página para evitar los solapamientos ocasionados por los elementos posicionados con float.



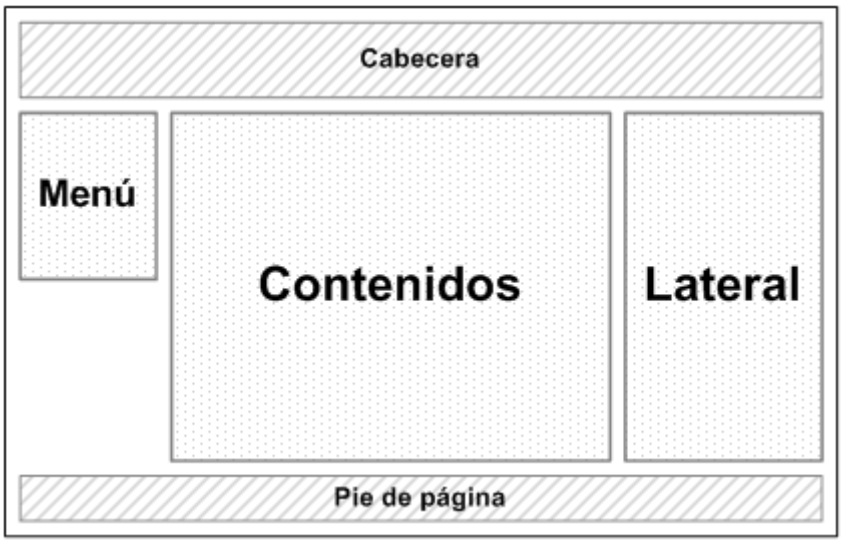
```
#contenedor {
		width: 700px;
		}
#cabecera {
	}
#menu {
	float: left;
	width: 150px;
	}
#contenido {
	float: left;
	width: 550px;
	}
#pie {
	clear: both;
	}


	<body>
		<div id="contenedor">
			<div id="cabecera">
			</div>
			<div id="menu">
			</div>
			<div id="contenido">
			</div>
			<div id="pie">
			</div>
		</div>
	</body>
```

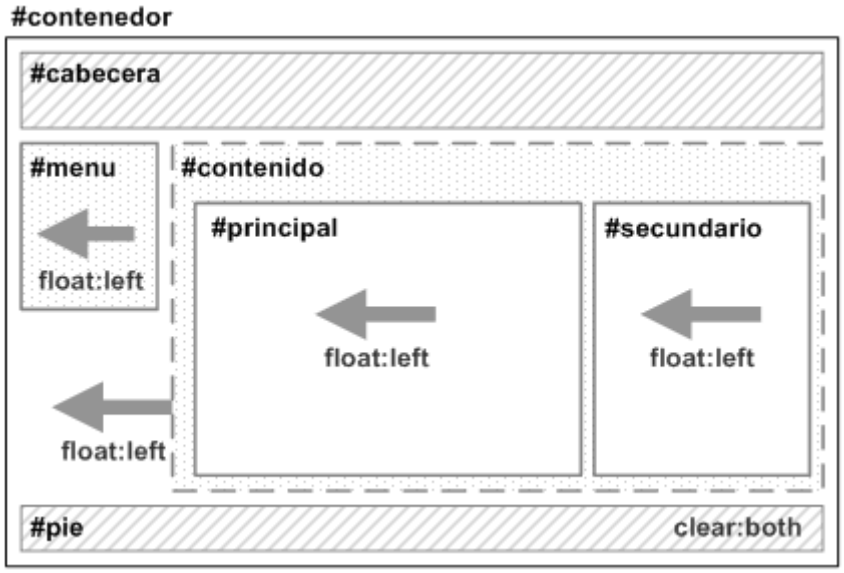
El diseño anterior es de anchura fija, lo que significa que no se adapta a la anchura de la ventana del navegador. Para conseguir una página de anchura variable y que se adapte de forma dinámica a la ventana del navegador, se deben aplicar las siguientes reglas CSS:

```
#contenedor {
}
#cabecera {
}
#menu {
        float: left;
        width: 15%;
}
#contenido {
        float: left;
        width: 85%;
}
#pie {
        clear: both;
}
```

Si se indican la anchuras de los bloques que forman la página en porcentajes, el diseño final es dinámico. Para crear diseños de anchura fija, basta con establecer las anchuras de los bloques en píxel.

## 12.4 Diseño a 3 columnas con cabecera y pie de página






```
#contenedor {
}
#cabecera {
}
#menu {
	float: left;
	width: 15%;
}
#contenido {
	float: left;
	width: 85%;
}
#contenido #principal {
	float: left;
	width: 80%;
}
#contenido #secundario {
	float: left;
	width: 20%;
}
#pie {
	clear: both;
}
```

```
<body>
	<div id="contenedor">
		<div id="cabecera">
		</div>
		<div id="menu">
		</div>
		<div id="contenido">
			<div id="principal">
			</div>
			<div id="secundario">
			</div>
		</div>
		<div id="pie">
		</div>
	</div>
</body>
```

# 12.5. Alturas/anchuras máximas y mínimas

Cuando se diseña la estructura de una página web, se debe tomar la decisión de optar por un diseño de anchura fija o un diseño cuya anchura se adapta a la anchura de la ventana del navegador. Sin embargo, la mayoría de las veces sería conveniente una solución intermedia: que la anchura de la página sea variable y se adapte a la anchura de la ventana del navegador, pero respetando ciertos límites. En otras palabras, que la anchura de la página no sea tan pequeña como para que no se puedan mostrar correctamente los contenidos y tampoco sea tan ancha como para que las líneas de texto no puedan leerse cómodamente.

CSS define cuatro propiedades que permiten limitar la anchura y altura mínima y máxima de cualquier elemento de la página. Las propiedades son max-width, min-width, max-height y min-height.

| | |
|---|---|
| Max-width<br>Max-height | Anchura máxima<br>Altura máxima |
| valores | <medida> \| <porcentaje> \| none \| inherit |
| Max-width se aplica a<br>Max-height se aplica a | Todos los elementos salvo filas y grupos de filas de tablas<br>Todos los elementos salvo columnas y grupos de columnas de tablas |
| Valor inicial | None |
| Descripción max-width<br>Descripción max-heght | Permite definir la anchura máxima de un elemento<br>Permite definir la altura máxima de un elemento |

| | |
|---|---|
| min-width<br>min-height | Anchura mínima<br>Altura mínima |
| valores | <medida> \| <porcentaje> \| inherit |
| min-width se aplica a<br>min-height se aplica a | Todos los elementos salvo filas y grupos de filas de tablas<br>Todos los elementos salvo columnas y grupos de columnas de tablas |
| Valor inicial | 0 |
| Descripción min-width<br>Descripción min-heght | Permite definir la anchura mínima de un elemento<br>Permite definir la altura mínima de un elemento |

De esta forma, para conseguir un diseño de anchura variable pero controlada, se podrían utilizar reglas CSS como la siguiente:

```
#contenedor {
    min-width: 500px;
    max-width: 900px;
}
```

Desafortunadamente, Internet Explorer 6 y las versiones anteriores no soportan ninguna de las cuatro propiedades sobre ningún elemento.

# 13 Tipos de medios

HTML es independiente de los medios, es decir, representa el contenido sin tener en cuenta el medio (pantalla de PC, PDA, TV, impresora, voz, braille, ...) en el que será representado. CSS permite diferenciar la presentación según el medio al que va destinado. Usando CSS tal y como hemos visto hasta ahora las reglas se aplican a todos los medios (las que puedan aplicarse, porque hay propiedades que son específicas para algunos medios -el salto de página sólo tiene sentido en medios paginados y los colores no tienen sentido en medios auditivos- otras propiedades son de aplicación en varios medios pero interesa darles valores distintos a cada uno -por ejemplo, el tamaño de letra para pantallas o para proyectores-)

Los tipos de medios definidos en CSS son:

| MEDIO | DESCRIPCIÓN |
|---|---|
| all | afecta a todos los medios |
| braille | táctiles de Braille |
| embossed | impresoras braille |
| handheld | dispositivos de mano, pantalla pequeña, poco ancho de banda |
| print | medios paginados |
| projection | presentaciones proyectadas |
| screen | pantallas de ordenador |

| | |
|---|---|
| speech | sintetizadores de voz |
| tty | terminales de texto, rejillas de caracteres de tamaño fijo |
| tv | pantallas a color, baja resolución, sonido disponible |

La sintaxis para actuar sobre los medios es:

```
@media nombreMedio {
        selector {
                propiedad: valor;
                  }
}
```

donde nombreMedio puede ser un medio o una lista de medios separados por comas.
EJEMPLO:

```
@media print {
                P { text-indent: 2cm; }
}
```

donde establecemos una sangría para la primera línea de todos los párrafos cuando se visualice en un medio paginado como la impresora.

La regla @import nos permite incluir las reglas CSS que tenemos en un fichero externo:
@import url(“fichero.css”);

Podemos restringir el efecto de una regla @import a un único tipo de medios (o a varios separados por comas):

@import url(“fichero.css”) nombreMedio1, nombreMedio2;

NOTA: Recuerda que las reglas @ tienen que ir colocadas al principio del documento CSS, antes de los sistemas de reglas y que la primera regla @ habrá de ser obligatoriamente @charset sin poder estar precedida ni siquiera de un espacio.

## Reglas específicas para medios paginados

| PROPIEDAD | VALORES | DESCRIPCIÓN |
|---|---|---|
| orphans | número por defecto: 2 | Establece el número mínimo de líneas huérfanas o viudas que se permiten. Si el flujo de documento provoca un número menor, entonces todo el bloque se traslada. |
| widows | | |

NOTA: Cuando un párrafo queda dividido en dos partes por un salto de línea, las líneas de la parte que queda al final de la página se llaman huérfanas y las líneas que quedan al principio de la página siguiente se llaman viudas.

| Propiedad | Función | Valor | Acción |
|---|---|---|---|
| page-break-before<br>page-break-after | Salto de página previo<br>Salto de página posterior | always | salto de página siempre |
| | | avoid | lo evita |
| | | left | uno o dos saltos de página hasta que la siguiente página sea izquierda o derecha respectivamente |
| | | right | |
| | | auto | ni fuerza ni prohibe |
| page-break-inside | dentro del bloque | auto<br>avoid | |

Estas propiedades se aplican exclusivamente a elementos de bloque en el flujo normal. En caso de contradicción (que puede suceder porque un mismo punto del documento estará afectado por la regulación de los saltos de línea de la caja que lo contiene, de la caja precedente y de la caja siguiente) always, left y right mandan más que avoid.

## Las cajas de página y sus márgenes

Cada página se considera una caja especial formada por dos cajas concéntricas: el margen y el área. Para actuar sobre los márgenes de una caja de página usamos las propiedades margin ya conocidas pero el selector habrá de ser uno de los siguientes:

| SELECTOR | PÁGINAS SOBRE LAS QUE ACTÚA |
|---|---|
| @page | Actúa sobre todas las páginas |
| @page :first | Sólo afecta a la primera página. Útil para crear una portada |
| @page :left | Afecta sólo a las páginas de la izquierda o de la derecha. |
| @page :right | Impresión a doble cara y necesitamos márgenes distintos en cada lado. |

EJEMPLO:

```
@page :first {
        margin: 4cm;
}
```

Establece un margen de 4 centímetros por los cuatro lados, pero sólo para la primera página del documento.

# 14 Listas

Las siguientes propiedades sólo se aplican a los elementos de tipo lista, esto es, todos aquellos a los que asignemos display: list-item; y a los que ya lo tienen por defecto como el elemento LI de HTML.

| PROPIEDAD | VALORES | DESCRIPCIÓN |
|---|---|---|
| **list-style-type** | **decimal** | Desde 1 |
| | **decimal-leading-zero** | decimal desde 01 siempre con dos cifras |
| | **lower-roman** | números romanos minúsculas |
| | **upper-roman** | números romanos mayúsculas |
| | **georgian** | numeración georgiana |
| | **armenian** | numeración armenia |
| | **lower-latin o lower-alpha** | alfabeto latino minúscula |
| | **upper-latin o upper-alpha** | alfabeto latino mayúscula |
| | **lower-greek** | alfabeto griego |
| | **disc** | círculo |
| | **circle** | circunferencia |
| | **square** | cuadrado |
| **list-style-image** | url | imagen que queremos usar. Es conveniente poner también un valor de list-style-type para el que recurrirá si hay problemas con la imagen |
| **list-style-position** | **inside** | la segunda línea comienza debajo de la viñeta |
| | **outside** | la viñeta se queda a la izquierda de todas las líneas |
| **list-style** | | propiedad resumida: tipo, imagen y estilo |

# 15 Tablas

Las propiedades para tablas se aplican a los elementos con los siguientes valores para la propiedad display: table, inline-table, table-row-group, table-header-group, table-footer-group, table-row, table-column-group, table-column, table-cell, table-caption.

En HTML esos elementos son: TABLE, TR, THEAD,.. etc. Recuérdese que CSS no sólo se aplica a HTML, también se aplica a otros lenguajes como XML y esos otros lenguajes pueden no tener ningún elemento así definido por defecto. En esos casos tendremos que hacerlo explícitamente con la propiedad display.

Las celdas heredan propiedades de columnas y grupos de columnas aunque no sean descendientes, que no lo son.

| PROPIEDAD | NOTAS | VALORES | DESCRIPCIÓN |
|---|---|---|---|
| **caption-side** | se aplica a la tabla | **top** | arriba de la tabla (por defecto) |
| | | **bottom** | abajo de la tabla |
| **border-collapse** | se aplica a la tabla | **collapse** | líneas de extremo a extremo |
| | | **separate** | líneas alrededor de las celdas |

Después de establecer el border-collapse siguen sin verse los bordes, porque el estilo

por defecto de bordes es "none". Tendremos que aplicar border-style al elemento de la tabla que queramos que tenga bordes (celda, fila, columna, grupo de filas o columnas) mientras que el border-collapse se aplica a la tabla entera.

| PROPIEDAD | NOTAS | VALORES | DESCRIPCIÓN |
|---|---|---|---|
| **border-spacing** | se aplica a la tabla.<br>sólo tiene sentido si border-collapse: separate.<br>indica la separación entre bordes de celdas adyacentes. | 1 medida | para todos |
| | | 2 medidas | 1º horizontal<br>2º vertical |
| Los elementos internos de tablas no tienen márgenes, sólo border-spacing. | | | |
| **empty-cells** | aplicable a celdas TD, TH<br>para mostrar celdas vacías es imprescindible que esté el TD o el TH, o sea, que si hemos obviado las etiquetas no se mostrarán | **show** | las muestra |
| | | **hide** | no las muestra |

# 16 Recomendaciones generales sobre CSS

El W3C (World Wide Web Consortium) dispone de un validador online que permite validar reglas CSS sueltas, páginas XHTML con CSS incluído y archivos CSS independientes. El validador se puede acceder en http://jigsaw.w3.org/css-validator/

**Uso de atributos id y class**

Técnicamente, los valores de los atributos id y class deben cumplir las siguientes restricciones:

• Sólo pueden empezar por un guión medio (-), un guión bajo (_) o una letra.

• El resto de caracteres, pueden ser números, guiones medios (-), guiones bajos (_) y letras.

• Los navegadores distinguen entre mayúsculas y minúsculas.

• Aunque es posible utilizar letras como ñ y acentos, no se recomienda hacerlo porque no es seguro que funcione correctamente en todas las versiones de todos los navegadores.

Un error común al comenzar a desarrollar páginas con estilos CSS es la utilización excesiva de etiquetas  y atributos class. Diseñar páginas con exceso de etiquetas  no mejora la semántica del documento y sólo consigue complicar el código HTML.

**Estructuración del código CSS**

Las reglas CSS de las hojas de estilos complejas se suelen agrupar según su funcionalidad y se suelen incluir en el siguiente orden:

▪ Estilos básicos (<body>, tipo de letra por defecto, márgenes de <ul>, <ol> y <li>, etc.)

▪ Estilos de la estructura o layout (anchura, altura y posición de la cabecera, pie de página,
zonas de contenidos, menús de navegación, etc.)

▪ Enlaces (estilos normales, estilos :hover, etc.)

▪ Estilos de cada una de las zonas (elementos de la cabecera, titulares y texto de la

zona de contenidos, enlaces, listas e imágenes de las zonas laterales, etc.)

Otra característica común de los mejores sitios web es el uso de comentarios CSS para mejorar la estructura de las hojas de estilos muy largas.

**División de los estilos en varios archivos CSS**

Normalmente, los estilos de una página compleja se dividen en varios archivos CSS diferentes para hacerlos más manejables.
Una vez creados los archivos CSS, existen dos estrategias para enlazar varios archivos CSS en las páginas HTML:
Si se puede modificar fácilmente la cabecera del documento (por ejemplo porque las páginas se generan dinámicamente) lo habitual es incluir tantos elementos <link> como archivos CSS se enlazan:

```
<head>
...
<link rel="stylesheet" type="text/css" href="/css/basico.css" media="screen" />
...
</head>
```

Si no se puede modificar de forma sencilla la cabecera de los documentos para añadir, eliminar y modificar los archivos CSS que se enlazan, lo habitual es enlazar un único archivo CSS que se encarga de importar todos los demás:

```
<head>
...
<link rel="stylesheet" type="text/css" href="/css/estilos.css" media="all" />
...
</head>
```

El contenido del archivo estilos.css debería ser el siguiente para ser equivalente al ejemplo
anterior:

```
@import url("basico.css") screen;
@import url("seccion.css") screen;
```