

ÍNDICE

1. CSS (CASCADE STYLE SHEET)	3
2. FUNDAMENTOS	4
2.1. NOMENCLATURA Y ESTRUCTURA	4
2.2. COLORES	5
2.3. UNIDADES DE MEDIDA	6
3. SELECTORES.....	8
3.1. SELECTORES DE ETIQUETA (O DE ELEMENTO)	8
3.2. SELECTOR DE IDENTIFICACIÓN	8
3.3. SELECTOR DE CLASE	9
3.4. SELECTOR DE ATRIBUTO	10
3.5. SELECTOR DE PSEUDOCCLASE	11
3.6. SELECTORES MÚLTIPLES	12
4. PROPIEDADES.....	14
4.1. FORMATEAR TEXTO	14
4.2. FUENTES	17
4.3. CAJAS.....	18

1. CSS (CASCADE STYLE SHEET)

CSS (Cascading Style Sheets u Hojas de Estilo en Cascada) es el lenguaje utilizado para aplicar formato mayormente pero no exclusivamente visual (colores, formas, tamaños...) en la web.

- **Cascading:** El formato aplicado a un elemento se propaga, de manera descendente o en cascada, a los elementos contenidos en él.
- **Style:** Aplicamos estilos, visuales en la mayoría, a los diferentes elementos de las páginas web.
- **Sheets:** El conjunto de estilos aplicados se denomina "*hoja de estilos*".

Es un lenguaje estandarizado, bajo la tutela de la W3C, que, actualmente, se encuentra en su versión 3 (CSS3).

Nota: **NO** es un lenguaje de programación (como tampoco lo es HTML).

+Info:

<https://www.w3schools.com/css>

<https://developer.mozilla.org/es/docs/Web/CSS/Reference>

2. FUNDAMENTOS

```
selector
p {
  propiedad      valor
  color: #ff00ff;
               declaración
}
```

REGLA PREDETERMINADA (O REGLA)

2.1. NOMENCLATURA Y ESTRUCTURA

Llamamos regla predeterminada (o simplemente regla) al conjunto de declaraciones aplicadas a un elemento. Cada regla consta de:

- Selector: indica a qué elemento o elementos se aplicará la regla.
- Declaración: cada uno de los estilos que se aplicarán al selector
 - Propiedad: el estilo a aplicar
 - Valor: el valor aplicado al estilo

En el ejemplo de la imagen 1, el selector es p (etiqueta HTML de párrafo), la propiedad es color (que modifica el color del texto) y el valor es #ff00ff (el color en concreto que queremos aplicar).

Un selector puede incluir más de un elemento. En este caso, se separarán utilizando comas (el último elemento no lleva coma):

```
p, h1, h3, li {
  color: #ff00ff;
}
```

En el ejemplo anterior, aplicaremos el estilo color a las etiquetas HTML p, h1, h3 y li.

Cada selector puede ir en una nueva línea, siempre que no olvidemos la coma separadora.

```
p,  
h1,  
h3,  
li {  
    color: #ff00ff;  
}
```

Una regla puede incluir varias declaraciones:

```
p {  
    color: #ff00ff;  
    font-size: 32px;  
    text-decoration: underline;  
}
```

RESUMEN:

- La propiedad y el valor de cada declaración se separan siempre con dos puntos.
- Cada declaración termina en un punto y coma.
- Una regla puede constar de varias declaraciones.
- Cada regla va dentro de un par de llaves.
- El selector (o selectores) se sitúan fuera de las llaves y delante de la de apertura.
- Podemos tener varios selectores por regla (separados por comas en caso de ser más de uno).

2.2. COLORES

Los colores en CSS se pueden indicar de distintas maneras:

- Por su nombre en inglés (no sirven todos, existe un listado limitado de dichos nombres):

```
h3 { color: orange; }
```

- Por su código rgb en hexadecimal. Son seis caracteres en total. Los dos primeros suministran la cantidad de rojo, los dos siguientes la de verde y los dos siguientes la de azul. Esos dos caracteres por color son un número en hexadecimal comprendido entre 00 y ff (0 y 255 en decimal). Al principio de los seis caracteres se antepone el símbolo de almohadilla #.

```
h3 { color: #0189af; }
```

- Por sus valores rgb Tres números, separados por comas, comprendidos entre 0 y 255, ambos inclusive.

```
h3 { color: rgb(128, 34, 128); }
```

- Por sus valores rgba. Cuatro números, separados por comas. Los tres primeros están comprendidos entre 0 y 255, ambos inclusive. El último será un número entre 0 y 1 (la coma decimal se representa con un punto, no con una coma). Este último número indica el grado de transparencia del color: 0 opaco, 1 transparente.

```
h3 { color: rgba(128, 34, 128, 0.37); }
```

+Info: <https://htmlcolorcodes.com/>

2.3. UNIDADES DE MEDIDA

En CSS podemos usar distintas unidades de medida, organizadas en dos grandes grupos: absolutas y relativas.

2.3.1. ABSOLUTAS

Son unidades que proporcionan un tamaño per se, sin depender del valor de otro elemento. Excepto los píxeles, son habitualmente utilizadas en hojas de estilo para impresión física en papel:

- px: píxeles
- cm: centímetros
- mm: milímetros

- in: pulgadas
- pt: puntos
- pc: picas

2.3.2. RELATIVAS

Son unidades que toman su valor de otras unidades definidas anteriormente.

- em: proporcional al tamaño del elemento contenedor (2em sería dos veces el tamaño de la fuente actual).
- rem: proporcional al tamaño del elemento raíz (2rem sería dos veces el tamaño de la fuente definido en el body).
- vw: relativo al 1% del ancho del display o monitor (5vw será igual al 5% del ancho del monitor)
- vh: relativo al 1% del alto del display o monitor (5vh será igual al 5% del alto del monitor)
- %: relativo al tamaño del elemento contenedor

3. SELECTORES

- Selectores de etiqueta (o de elemento):
Permite seleccionar cualquier elemento HTML.
- Selector de identificación:
Permite seleccionar elementos por su ID.
- Selector de clase:
Permite seleccionar elementos por su clase.
- Selector de atributo:
Permite seleccionar elementos por su atributo
- Selector de pseudoclase:
Permite seleccionar elementos cuando tengan un estado especificado

3.1. SELECTORES DE ETIQUETA (O DE ELEMENTO)

Nos permite utilizar **cualquier etiqueta HTML** como elemento selector. Todas las etiquetas del documento se verán afectadas. En CSS, NO utilizaremos los símbolos <> para escribir el nombre de la etiqueta.

```
h3 {  
    color: #0000ff;  
    font-size: 48px;  
    text-decoration: underline;  
}
```

3.2. SELECTOR DE IDENTIFICACIÓN

Nos permite utilizar el atributo **id** para identificar el elemento al que queremos aplicar el estilo. Situremos el signo # delante de la id correspondiente:

```
#cabeceraprincipal {  
    color: #0000ff;  
    font-size: 48px;  
    text-decoration: underline;  
}
```


En el ejemplo anterior el id utilizado debe tener como valor “cabeceraprincipal” (sin el #), independientemente de la etiqueta HTML donde se encuentre.

```
<section id="cabeceraprincipal">
```

NOTA: Recordemos que, dentro de un único documento html, **NO SE PUEDE** repetir el valor de un atributo id.

```
<ul id="listado1">
    <li id="elemento1">Texto</li>
    <li id="elemento2">Texto</li>
</ul>
<ul id="listado2">
    <li id="elemento1">Texto</li>
    <li id="elemento2">Texto</li>
</ul>
```

En el ejemplo anterior se produciría un error, debido a que hemos usado los mismos valores (elemento1 y elemento2) para los id de las etiquetas li.

3.3. SELECTOR DE CLASE

Nos permite utilizar el atributo **class** para identificar el elemento al que queremos aplicar el estilo. Sitaremos un punto (.) delante de la clase correspondiente:

```
.colortextoverde {
    color: #00ff00;
}
```

En el ejemplo anterior el class utilizado debe tener como valor “colortextoverde” (sin el .), independientemente de la etiqueta HTML donde se encuentre.

```
<section class="colortextoverde">
```

A diferencia de id, class **SÍ** se puede reutilizar dentro de un mismo documento HTML y en cualquier etiqueta:

```
<div id="elemento1">
  <p class="colortextoverde">Texto</p>
  <p>Texto</p>
  <ul id="listado1">
    <li class="colortextoverde">Texto</li>
    <li>Texto</li>
  </ul>
</div>
```

3.4. SELECTOR DE ATRIBUTO

Nos permite identificar una etiqueta HTML a través de uno de sus atributos y/o del valor que dicho atributo tenga.

```
<p><a href="imagen.jpg" title="tengo algo">Algo</a></p>
<p><a href="noseque.html" title="tengo título">Algo</a></p>
<p><a href="documento.pdf" title="un documento">Algo</a></p>
```

Se escribe poniendo, todo junto, el nombre de la etiqueta y el nombre del atributo (este entre corchetes).

Este ejemplo eliminará el subrayado de todas las etiquetas a que tengan atributo title.

```
a[title] { text-decoration: none; }
```

El siguiente ejemplo, cambiará el color del texto de las etiquetas a que tengan atributo title y que el contenido de dicho atributo sea "tengo título"

```
a[title="tengo título"] { color: #00ff00; }
```

El siguiente ejemplo, cambiará a mayúsculas el texto de las etiquetas a que tengan atributo href y que el contenido de dicho atributo termine en ".pdf". En este caso, ponemos un símbolo de dólar antes de símbolo de igual, para indicar que "finaliza en".

```
a[href$=".pdf"] { text-transform: uppercase; }
```

Aquí, usamos el símbolo de asterisco * para indicar que href “contiene” el texto “palabra”.

```
a[href*="seque"] { text-transform: uppercase; }
```

Ahora, usamos el símbolo de acento circunflejo, para indicar que “empieza con” la palabra “inicio”.

```
a[href^="inicio"] { text-transform: uppercase; }
```

3.5. SELECTOR DE PSEUDOCCLASE

Nos permite seleccionar una etiqueta que tenga un estado especificado. El ejemplo más conocido, posiblemente, es :hover (cuando pasamos el ratón por encima del elemento). Pero hay muchos más.

Se escribe poniendo, todo junto, el nombre de la etiqueta y el nombre de la pseudo clase, sin olvidar el símbolo de dos puntos entre ambos:

```
p:hover { text-transform: uppercase; }
```

El ejemplo anterior hará que cualquier párrafo cambie todo su texto a mayúsculas si pasamos el ratón por encima.

p:first-child

Selecciona el primer párrafo de cada etiqueta padre que contenga párrafos.

p:last-child

Selecciona el último párrafo de cada etiqueta padre que contenga párrafos.

p:nth-child(odd)

Selecciona los párrafos impares de cada etiqueta padre que contenga párrafos.

p:nth-child(even)

Selecciona los párrafos pares de cada etiqueta padre que contenga párrafos.

p:nth-child(x)

Selecciona el párrafo número x de cada etiqueta padre que contenga párrafos.

p:nth-child(3n)

Selecciona los párrafos múltiplo de 3 de cada etiqueta padre que contenga párrafos, empezando desde el primer múltiplo (el tercer párrafo).

p:nth-child(3n-1)

Selecciona los párrafos múltiplo de 3 de cada etiqueta padre que contenga párrafos, empezando desde el primer múltiplo menos 1 (el segundo párrafo).

3.5.1. ESPECIALES

p::after { content: "algo"; }

Añade "algo" después del contenido de la etiqueta p.

Se pueden añadir más declaraciones tras "content", declaraciones que solo afectaran al contenido añadido.

p::before { content: "algo"; }

Añade "algo" antes del contenido de la etiqueta p.

Se pueden añadir más declaraciones tras "content", declaraciones que solo afectaran al contenido añadido.

3.6. SELECTORES MÚLTIPLES

div, .soyverde { ... }

Todos los divs y todas las etiquetas con clase soyverde

div.soyverde { ... }

Todas las etiquetas div que sean de la clase soyverde.

div .soyverde { ... }

Todas las etiquetas con clase soyverde que estén dentro de un div, sin importar el nivel de "profundidad".

div > .soyverde { ... }

Todas las etiquetas con clase soyverde que estén dentro de un div, pero únicamente en el primer nivel de este.

div ~ .soyverde { ... }

Todas las etiquetas con clase soyverde que estén precedidas, en su mismo nivel, de un div, sin importar que haya otras etiquetas en medio.

4. PROPIEDADES

Las propiedades nos indican que tipo de estilo vamos a aplicar. Si vamos a cambiar el color de un texto, del fondo de una etiqueta, el tamaño de una imagen... No indicaremos todas las posibilidades, este texto es a modo de introducción de las más básicas.

4.1. FORMATEAR TEXTO

4.1.1. COLOR

Cambia el color del texto de la etiqueta referida.

```
h3 { color: #0000ff; }
```

4.1.2. TEXT-ALIGN

Alinea el texto horizontalmente.

```
p { text-align: center; }
```

Sus principales valores:

- left
- right
- center
- justify

4.1.3. TEXT-DECORATION

Añade o elimina una línea decorativa al texto.

```
p { text-decoration: overline red dotted 3px; }
```

Se le pueden proporcionar hasta cuatro valores:

- Posición de la línea (es posible utilizar, simultáneamente, más de un valor):

- overline
 - line-through
 - underline
 - none
- Color
- Aspecto:
 - solid
 - double
 - dotted
 - dashed
 - wavy
- Grosor

El único valor imprescindible es el primero (posición de la línea). Se puede prescindir de cualquiera de los otros tres (o de los tres).

```
p { text-decoration: underline; }  
p { text-decoration: overline underline 3px; }
```

Como otras muchas propiedades de CSS, existe la posibilidad de desglosarla en cada uno de sus valores:

- text-decoration-line
- text-decoration-color
- text-decoration-style
- text-decoration-thickness

```
p {  
    text-decoration-line: overline underline;  
    text-decoration-color: red;  
    text-decoration-style: solid;  
    text-decoration-thickness: 3px;  
}
```

4.1.4. TEXT-TRANSFORM

Convierte un texto todo en mayúsculas, minúsculas o lo capitaliza. Sus valores:

- uppercase
- lowercase
- capitalize;

```
p { text-transform: capitalize; }
```

4.1.5. ESPACIADO

Distintas propiedades relativas al espaciado entre letras, palabras... Pueden usarse valores negativos.

- text-indent: separación de la primera línea de un texto en relación a las demás.

```
p { text-indent: 50px; }
```

- letter-spacing: separación entre las letras de cada palabra.

```
p { letter-spacing: 0.1rem; }
```

- line-height: espacio vertical entre las líneas del texto.

```
p { line-height: 3rem; }
```

- word-spacing: espacio entre las palabras de un texto.

```
p { word-spacing: 0.1em; }
```

4.1.6. TEXT-SHADOW

Nos permite añadir un efecto de sombra a un texto.

```
p { text-shadow: 2px 2px 10px red; }
```

Se le pueden proporcionar hasta cuatro valores (solo son necesarios los dos primeros):

- Desplazamiento horizontal
- Desplazamiento vertical
- Color de la sombra
- Desenfoque de la sombra

Se pueden añadir varias sombras, separando con comas los elementos que formen parte de cada una de ellas:

```
p { text-shadow: 0px 0px 4px yellow, 0px 0px 8px orange, 0px 0px 12px red; }
```

4.2. FUENTES

4.2.1. FONT-FAMILY

Indica la fuente tipográfica que se va a utilizar. Se pueden indicar varias opciones, en orden de preferencia, por si el usuario no dispone de alguna de ellas. Se escribirán los nombres de las fuentes, separándolos por comas. Si el nombre consta de más de una palabra, se escribirá entre comillas.

```
p { font-family: Verdana, Arial, "Times New Roman", serif; }
```

En el ejemplo anterior, la fuente preferida será Verdana. Si el usuario no la tiene instalada en su dispositivo, buscará la fuente Arial. Si tampoco está, pasará a buscar la "Time New Roman". Si tampoco existe, utilizará la fuente serif genérica que el dispositivo tenga instalada.

4.2.2. ESTILOS PARA FUENTES

Distintas propiedades relativas al aspecto visual de la fuente utilizada.

- font-style: texto normal, en itálica o en cursiva.
 - normal
 - italic
 - oblique
- font-weight: texto normal o en negrita.
 - normal
 - bold
- font-variant: texto normal o en versalita (las minúsculas se convierten en mayúsculas, pero sin aumentar su tamaño).
 - normal
 - small-caps
- font-size: establece el tamaño de la fuente.

```
p {
    font-style: italic;
    font-weight: bold;
    font-variant: small-caps;
    font-size: 2rem;
}
```

La propiedad font unifica varias de las propiedades anteriores, utilizándose en el siguiente orden:

font-style > font-variant > font-weight > font-size/line-height > font-family

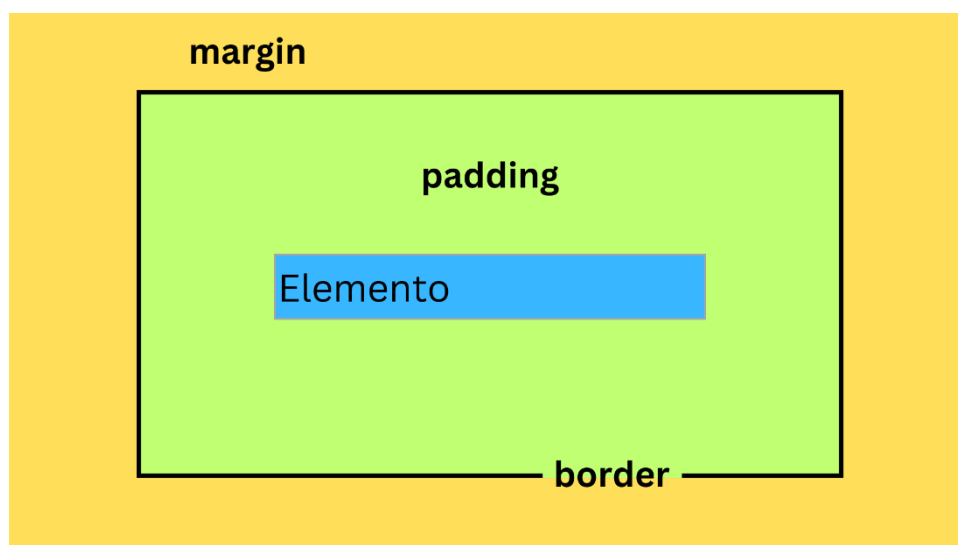
El tamaño de la fuente y el del alto de línea van juntos, separados únicamente por una barra invertida /

```
p { font: italic small-caps bold 12px/30px Georgia, serif; }
```

Si usamos font, solo son requeridos los valores de font-size y font-family. El resto de valores cogerán los que existan por defecto.

```
p { font: 12px Georgia, serif; }
```

4.3. CAJAS



Todas las etiquetas HTML ocupan un espacio visual en el documento, espacio al que llamaremos caja. Esta caja divide a las etiquetas en dos grandes grupos: etiquetas de bloque y etiquetas en línea.

- Las etiquetas de bloque hacen que su caja ocupe todo el espacio, a lo ancho, de que disponen, “empujando” a la siguiente etiqueta a continuación. Un párrafo, un título o un div son claros ejemplos de una etiqueta de bloque.
- Las etiquetas en línea no interrumpen el flujo del documento, ya que su caja solo ocupa el espacio que necesita y se mantiene “en línea” con el resto del contenido. No “empujan” a otras etiquetas. La etiqueta `a` es uno de sus ejemplos más evidentes.

Esta diferencia de comportamiento afecta al funcionamiento de algunos estilos, que solo son aplicables si la etiqueta se ajusta al tipo requerido. Por ejemplo, la propiedad `width` determina el ancho de un elemento, pero solo funciona si la etiqueta a la que se lo aplicamos es una etiqueta de bloque: una etiqueta `a` no puede tener `width`, pero una `p`, sí (por ahora, al menos...).

4.3.1. DISPLAY

Para imitar el comportamiento de ambos tipos de elementos, de bloque y de línea, tenemos una propiedad:

- **display**
 - `none`: el elemento simplemente desaparece visualmente, dejando libre su espacio.
 - `inline`: el elemento se comportará como un elemento en línea.
 - `block`: el elemento se comportará como un elemento de bloque.
 - `inline-block`: el elemento seguirá comportándose como un elemento en línea, pero podrá adquirir propiedades de un elemento de bloque (por ejemplo, se le podrá dar un `width`).

`a { display: inline-block; }`

4.3.2. TAMAÑOS, MÁRGENES, BORDES...

4.3.2.1. MODELO DE CAJA

Cuando le demos el tamaño a una caja, debemos considerar dos posibles opciones:

- Incluir las medidas de padding y border en el tamaño.
- No incluirlas.

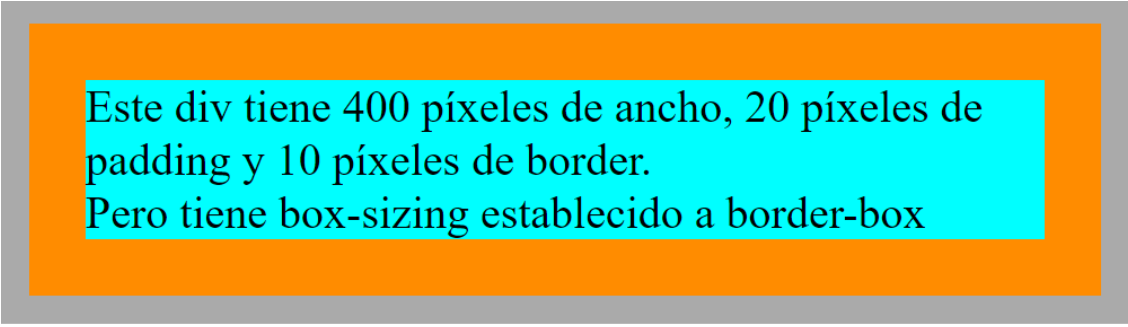
Para esto, usaremos la propiedad **box-sizing**, que puede tomar, entre otros, dos valores:

- **border-box**

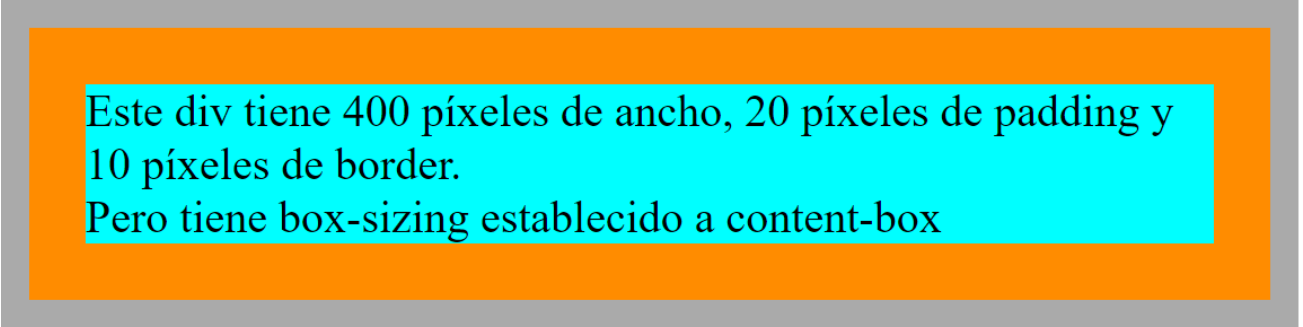
El tamaño de la caja INCLUYE las medidas de *padding* y de *border*.

- **content-box**

Las medidas de *padding* y de *border* SE AÑADEN a las medidas de la caja.



Este div tiene 400 píxeles de ancho, 20 píxeles de padding y 10 píxeles de border.
Pero tiene box-sizing establecido a border-box



Este div tiene 400 píxeles de ancho, 20 píxeles de padding y 10 píxeles de border.
Pero tiene box-sizing establecido a content-box

4.3.2.2. TAMAÑO

Podemos definir el ancho y el alto de un elemento usando **width** y **height**, respectivamente:

```
p { width: 50%; }
```

```
h3 { height: 100px; }  
h6 { width: 50%; height: 100px; }
```

Estas propiedades solo funcionan con elementos de bloque.

El elemento continuará ocupando el 100% del espacio disponible, el efecto es visual.

4.3.2.3. PADDING

Alrededor de cada elemento existen cuatro rellenos que separan este elemento de su borde.

```
p { padding: 10px; }  
p { padding: 10px 20px; }  
p { padding: 10px 20px 10px; }  
p { padding: 10px 20px 10px 20px; }
```

- Si solo ponemos un valor, se aplicará este a los cuatro rellenos (arriba, derecha, abajo e izquierda).
- Si ponemos dos valores, se aplicará el primero al relleno superior y al inferior, y el segundo, a derecha e izquierda.
- Si ponemos tres valores, se aplicará el primero al relleno superior; el segundo, a derecha e izquierda, y el tercero, al relleno inferior.
- Si ponemos cuatro valores, se aplicarán en este orden: arriba, derecha, abajo e izquierda.

4.3.3. BORDER

Rodeando los rellenos (*padding*), cada elemento tiene un borde.

Los *borders* tienen tres valores:

- width
- style
- color

```
p { border: 10px solid red; }
```

Border admite **none** como valor (en este caso, se elimina por completo el borde). Como en otros casos, se puede desglosar los valores de border en sus propiedades individuales:

- border-color
- border-style
- border-width

```
p { border-color: red; }
```