

# Lenguajes de marcas y sistemas de gestión de la información

## TEMA 3: CSS3, maquetación y diseño Web.

### Introducción

#### **Limitaciones del HTML**

No es fácil armonizar semántica y formato en un solo lenguaje

- HTML, hoy en día, es puramente semántico
- Es más fácil de mantener un sitio web si el contenido y el formato se independizan
- No es un lenguaje pensado para dar formato, está pensado para ser simple

#### **Funcionamiento más apropiado**

- HTML->Descripción de elementos
- CSS->Cómo se deben presentar esos

#### **Versiones de Css y compatibilidad**

- CSS1 (1996)
- Formatos de texto, párrafo, caja, lista, tamaños de imágenes
- CSS2 (1998) y CSS 2.1
- Posicionamiento, tipos de medio

- CSS3 (actual)
- <http://www.w3.org/Style/CSS/current-work> 30 módulos en trabajo, estándar vivo (parte de HTML5)

Tenemos dos páginas que nos ayudan a ver la compatibilidad

- <http://caniuse.com> (uso de CSS3 en los navegadores)
- <http://css3test.com> (compatibilidad de mi navegador)

Existen una serie de prefijos que se utilizan para cada navegador y que no son compatibles

Las nuevas propiedades (sin compatibilidad) utilizan unos prefijos determinados:

- -moz- ->Mozilla
- -webkit- ->Motor de Safari, Chrome y (ahora) Opera
- -ms- ->Microsoft
- -o- ->Versiones anteriores de Opera

## Vincular los estilos CSS a páginas HTML


¿Cómo se aplican a una página HTML?

Para ello tienes básicamente 3 opciones:

### 1. Hojas de estilo CSS inline

La primera opción consiste en usar el atributo **“style”** en un elemento HTML tal como se puede ver aquí:

```
<h1 style="font-family:Verdana; color:red">Titulo de nivel 1</h1>
```



En este caso, no hay selector puesto que no hay elemento que seleccionar, es el que es. Por tanto, se especifican simplemente las propiedades a aplicar al elemento en cuestión. Este tipo de reglas tienen mayor prioridad de todas las reglas.

**Se suele recomendar encarecidamente evitar esta manera de vincular estilos CSS puesto que al tener que aplicarse a cada elemento individualmente, implican un esfuerzo y mantenimiento disparatado.**

No obstante, hay casos de uso muy concretos en los que son prácticos. Por ejemplo, en un widget sencillo de WordPress que permita configurar directamente ciertas propiedades CSS desde su interfaz de configuración puede ser una buena opción implementar estas propiedades con esta técnica.

## 2. Hojas de estilo CSS internas

La segunda opción es escribir las reglas dentro de un elemento `<style>` dentro de esta etiqueta que, a su vez, se encontraría dentro del elemento `<head>` de la página HTML.

**Obviamente la desventaja es que en este caso las reglas sólo son visibles para esa página concreta en las que se han declarado y no se pueden usar para otras páginas.**

## 3. Hojas de estilo CSS externas

La última opción es la más usada y consiste en crear uno o varios ficheros con extensión “.css” en los cuales se declaran las reglas CSS.

La manera de vincular estas reglas a una página HTML consiste en referencia el fichero .css desde esa página. De este modo se puede crear un fichero global para una web entera y usarlo en todas las páginas que se quiera.

En este caso, la declaración también se incluye dentro del elemento `<head>` y tiene el siguiente aspecto:

```
<link rel="stylesheet" type="text/css" href="/css/estilos-1.css" />  
<link rel="stylesheet" type="text/css" href="/css/estilos-2.css" />
```

Aquí hay que tener también en cuenta que en el caso de usar varios ficheros el orden de inclusión marca precedencia en el orden inverso. En el caso de existir un conflicto por haber

diferentes reglas con igual selector en varios ficheros, las reglas de los ficheros siguientes sobrescriben los anteriores.

Es decir, si en estilos-1.css tenemos una regla como:

```
p a {  
    text-decoration: underline;  
}
```

Y luego en estilos-2.css otra que dice:

```
p a {  
    text-decoration: none;  
    color: blue;  
}
```

La propiedad “text-decoration” genera un conflicto que se resuelve dando prioridad a la propiedad de estilos-2.css.

Para realizar una página web usando un archivo CSS externo, se deben seguir los tres pasos siguientes:

- Se crea un archivo de texto plano con las definiciones de los formatos.
- Dicho archivo de texto se guarda con extensión .css
- Se enlaza el archivo CSS externo mediante la etiqueta <link> en la cabecera de la página web.

**El elemento <link>** puede tener definidos cuatro atributos cuando se enlaza un archivo CSS:

- **rel**, indica el tipo de relación que tiene el archivo enlazado y la página HTML. Para los archivos CSS, siempre se utiliza el valor stylesheet
- **type**, indica el tipo de recurso enlazado. Para los archivos CSS su valor siempre es text/css
- **href**, indica la URL del archivo CSS que contiene los estilos. Puede ser relativa o absoluta y puede referenciar a un recurso interno o externo al sitio web.
- **media**, indica el medio en el que se van a aplicar los estilos del archivo CSS.

Se puede obtener el mismo resultado anterior utilizando el elemento **<style>** en lugar de <link>.

En este caso, se usa una regla de tipo @import seguida de una cadena de texto encerrada con comillas simples o dobles que se corresponde con la URL del archivo CSS, o de url() conteniendo

dicha cadena entre los paréntesis. Las siguientes reglas `@import` son equivalentes para un fichero `formatos.css` que está en el directorio `css`:

```
@import '/css/formatos.css';  
@import "/css/ formatos.css";  
@import url('/css/ formatos.css');  
@import url("/css/ formatos.css");
```

## ¿Cómo funcionan las hojas de estilo en cascada?

Las hojas de estilo CSS son un conjunto reglas que enumeran en un fichero `.css` y que describen el **aspecto** que deben tener los diferentes **elementos HTML** de una página.

Lo interesante de esto es que funcionan con una **filosofía de patrones o plantillas**, es decir, no es necesario especificar cada uno de los elementos, sino que se pueden definir reglas como estas dos:

- “Los títulos de nivel 1 y 2 han de ser de color negro y un tamaño de fuente de 16 y 14 pixeles respetivamente.”
- “El texto de los párrafos están alineados a la izquierda, tienen un tamaño de fuente de 12 pixeles y un color gris oscuro.”

A modo de comparación, si dominas el uso de **estilos en Microsoft Word**, verás que esto se parece mucho al concepto de estilo en Word, aunque las CSS son infinitamente más potentes que Word en todas sus posibilidades.

Veamos un ejemplo concreto de cómo se expresarían las reglas anteriores en el lenguaje de las CSS:

```
h1, h2 {  
    color: black;  
    font-size: 16px;  
}  
  
h2 {  
    font-size: 14px;  
}  
  
p {  
    color: rgb(32,32,32);  
    text-align: left;  
}
```

## Sintaxis de las reglas de estilo

Cada uno de los estilos que componen una hoja de estilos CSS se denomina regla. Cada regla se forma por:

- Selector: indica el elemento o elementos HTML a los que se aplica la regla CSS
- Llave de apertura, {
- Declaración: especifica los estilos que se aplican a los elementos.  
  
Propiedad: permite modificar el aspecto de un atributo del elemento.  
  
Valor: indica el nuevo valor del atributo modificado en el elemento.
- Llave de cierre, }.

```
Selector {  
  propiedad1:valor1;  
  propiedad2:valor2;  
  ...  
}
```

## Reglas, selectores y propiedades

En el ejemplo anterior tenemos un total de **3 reglas**, cada una se compone de un selector que indican a qué elementos HTML aplica la regla que es el texto que precede a las llaves de apertura y cierre de la regla. En la primera regla, el selector es “h1, h2”, en la segunda “h2” y en la tercera “p”.

Cada regla se compone a su vez de **propiedades** que especifican qué exactamente se hará con esos los elementos HTML a los que aplica.

El selector de la primera regla, por ejemplo, indica que esta regla se refiere a los elementos HTML `<h1>` y `<h2>` y que a ellos se aplicarán el color negro (propiedad “color”) y un tamaño de letra de 16 pixeles al texto (propiedad “font-size”).

Hay **cientos de propiedades** de las cuales destacaré luego las que, en mi experiencia, me han parecido las más frecuentemente usadas y más útiles. En cuanto a los selectores, los ejemplos han sido triviales puesto que se refieren a elementos HTML sin más, pero hay formas mucho más sofisticadas de diseñar selectores.

Un ejemplo rápido de esto:

```
p a {  
    text-decoration: underline;  
}  
  
p.text-izq {  
    text-align: left;  
}
```

En la primera regla se han anidado dos elementos para indicar que esta regla sólo se refiere a enlaces (etiqueta `<a>` en HTML) dentro de párrafos de texto (etiqueta `<p>` en HTML) y que a estos enlaces se le aplicará un subrayado. Es decir, esta regla **no aplica** a elementos `<a>` que no se encuentren dentro de un párrafo (`<p>`).

En la segunda regla se ha especificado algo parecido, pero jugando con un **elemento** y un **atributo de clase**. Es decir, esta regla aplicará a párrafos que contienen un atributo “**class**” con el valor “**text-izq**” tal como éste:

```
<p class="text-izq">Texto del párrafo</p>
```

Fíjate que en la regla “text-izq” empieza con un punto. Es la manera de especificar en el lenguaje CSS que “text-izq” se refiere a un atributo de clase.

Ya sólo con esto ya tienes cierto juego y creo que puedes ver la filosofía de cómo funcionan los selectores y qué pretenden.

## Tipos de selectores básicos

**\*** es el selector universal. Selecciona todo los elementos.

**#id** selecciona el elemento que tenga ese valor (id en este ejemplo) en el atributo id.

**.class** selecciona los elementos que tengan ese valor (class en este ejemplo) en el atributo class.

**etiqueta** selecciona esas etiquetas concretas.

**selector1,selector2** sirve para cambiar las propiedades de los elementos seleccionados por ambos selectores

**selector1 selector2** sirve para cambiar las propiedades de los elementos seleccionados por selector2 que se encuentran dentro de aquellos seleccionados por selector1.

**selector1>selector2** sirve para cambiar las propiedades de los elementos seleccionados por selector2 que son hijos directos (en el árbol DOM) de aquellos que se seleccionen con selector1.

**selector1+selector2** sirve para cambiar las propiedades de los elementos seleccionados por selector2 que están justo después de aquellos que se seleccionan mediante selector1.

**[atribut expre valor]** siendo expre (=,~=,|=,\$=,\*..) para seleccionar elementos atendiendo al valor de sus atributos.

## Comentarios

`/* comentario */`

## La cascada: herencia, sobreescritura y conflictos de estilos

El nombre de hojas de estilo en “cascada” no es casual, expresa que los estilos que especifican con reglas se pueden heredar de una manera jerárquica.

Es decir, veamos esta regla:



```
body {  
  font-family: Arial;  
}
```

Aquí estamos diciendo que la etiqueta <body> que es la que envuelve el contenido de cualquier página web tenga un tipo de letra “Arial”. Esto no tendría mucho sentido si no fuera porque gracias a la capacidad de herencia de las reglas de este modo por defecto cualquier elemento hijo como un título o un párrafo va a “heredar” ese estilo, salvo que especifique lo contrario como, por ejemplo, en esta regla:

```
p {  
  font-family: Verdana;  
}
```

En esta última regla aplica el principio de “especificidad”. En principio, se plantearía un conflicto entre la regla general de <body> con lo que dice la regla de <p>, pero se resuelve fácilmente puesto que se aplica la regla más específica y referirse a un párrafo es más específico que referirse a “los elementos hijos que pueden dentro de <body>”.

Igualmente, una regla con un selector “p a” (enlace dentro de un párrafo) tendría precedencia sobre una regla con un selector “a” que se refiere a un enlace a secas.

## Colisiones de estilo

En las hojas de estilos complejas, es habitual que varias reglas CSS se apliquen a un mismo elemento HTML. El problema de estas reglas múltiples es que se pueden dar colisiones como la del siguiente ejemplo:

```
p { color: red; }  
p { color: blue; }
```

<p>...</p>

¿De qué color se muestra el párrafo anterior? CSS tiene un mecanismo de resolución de colisiones muy complejo y que tiene en cuenta el tipo de hoja de estilo que se trate (de

navegador, de usuario o de diseñador), la importancia de cada regla y lo específico que sea el selector.

El método seguido por CSS para resolver las colisiones de estilos se muestra a continuación:

- Determinar todas las declaraciones que se aplican al elemento para el medio CSS seleccionado.
- Ordenar las declaraciones según su origen (CSS de navegador, de usuario o de diseñador) y su prioridad (palabra clave !important).
- Ordenar las declaraciones según lo específico que sea el selector. Cuanto más genérico es un selector, menos importancia tienen sus declaraciones.
- Si después de aplicar las normas anteriores existen dos o más reglas con la misma prioridad, se aplica la que se indicó en último lugar.

**Hasta que no se expliquen más adelante los conceptos de tipo de hoja de estilo y la prioridad, el mecanismo simplificado que se puede aplicar es el siguiente:**

- 1. Cuanto más específico sea un selector, más importancia tiene su regla asociada.**
- 2. A igual especificidad, se considera la última regla indicada.**

Como en el ejemplo anterior los dos selectores son idénticos, las dos reglas tienen la misma prioridad y prevalece la que se indicó en último lugar, por lo que el párrafo se muestra de color azul.

En el siguiente ejemplo, la regla CSS que prevalece se decide por lo específico que es cada selector:

```
p { color: red; }  
p#especial { color: green; }  
* { color: blue; }
```



```
<p id="especial">...</p>
```

Al elemento `<p>` se le aplican las tres declaraciones. Como su origen y su importancia es la misma, decide la especificidad del selector. El selector `*` es el menos específico, ya que se refiere a "todos los elementos de la página". El selector `p` es poco específico porque se refiere a "todos los párrafos de la página". Por último, el selector `p#especial` sólo hace referencia a "el párrafo de la página cuyo atributo `id` sea igual a `especial`". Como el selector `p#especial` es el más específico, su declaración es la que se tiene en cuenta y por tanto el párrafo se muestra de color verde.


## Unidades y colores en CSS

### Tipos de unidades medida

- **Absolutas.** Su medida no depende de ningún aspecto.
  - Su tamaño es predecible y exacto
  - Su tamaño real depende del dispositivo (en la mayoría de ocasiones)
- **Relativas.** Dependen de un tamaño base. Su tamaño es proporcional a este
  - El tamaño se adapta al dispositivo
  - Es menos predecible su tamaño exacto
  - Facilita la creación de páginas responsive

### Unidades absolutas

- **in.** Pulgada
- **cm.** Centímetro
- **mm.** Milímetro

- 
- **pt.** Punto
  - **pc.** Pica
  - **px.** Píxel

## Unidades relativas

- **em.** Usa la letra M mayúscula como referencia: 2em ->el doble del tamaño de la M mayúscula
- **ex.** Usa la equis minúscula
- **ch.** Usa el cero
- **%.** Porcentaje
- **rem.** Relativa a la letra base absoluta (el resto lo hace respecto a su contenedor)
- **vw.** 1% de la anchura del viewport
- **vh.** 1% de la altura del dispositivo
- **vmin.** La menor de los dos anteriores
- **vmax.** La mayor de las dos anterior

## Formas de indicar colores

Hexadecimal: #RRGGBB ->Rojo, Verde, Azul.

Ejemplo: #FF0000

Función RGB: rgb(255,0,0) ó rgb(100%,0,0)

Con transparencia: rgba(100%,0,0,0.5)->50% de transparencia

hsl(189,50%,60%) ->Tono 189, Saturación 50%, Luz 60%



hsla(189,50%,60%,.5)->Con transparencia

Por el nombre: black

## Asistencia para elegir colores

- Adobe Color. <https://color.adobe.com/>
- Color Lovers. <http://www.colourlovers.com/>
- Color Combos. <http://www.colorcombos.com/>
- Paletton. <http://paletton.com/>

## Propiedad opacity

Definir colores o capas transparentes: opacity

El valor de la propiedad opacity se establece mediante un número decimal comprendido entre 0.0 y 1.0. La interpretación del valor numérico es tal que el valor 0.0 es la máxima transparencia (el elemento es invisible) y el valor 1.0 se corresponde con la máxima opacidad (el elemento es completamente visible). De esta forma, el valor 0.5 corresponde a un elemento semitransparente y así sucesivamente.

## Fuentes de letras

La regla @font-face de CSS3 permite:

Utilizar fuentes tipográficas sin necesidad de tenerlas instaladas, utilizando repositorios gratuitos de fuentes como Google Fonts.

Crear uno (o varios) bloques donde definir las tipografías a cargar en el documento.

La directiva @font-face permite describir las fuentes que utiliza una página web. Como parte de la descripción se puede indicar la dirección o URL desde la que el navegador se puede descargar la fuente utilizada si el usuario no dispone de ella

Ejemplo: <https://fonts.googleapis.com/css?family=Open+Sans:300,300italic,600,600italic,700>

Sintaxis:

```
@font-face {  
  
    font-family: <un-nombre-de-fuente-remota>;  
  
    src: <origen> [,<origen>]*;  
  
    [font-weight: <peso>;  
  
    [font-style: <estilo>;  
  
}
```

Las fuentes de google disponibles se encuentran en la página <https://fonts.google.com/>

## Propiedades básicas que deberías conocer

### 1. Maquetación básica

- **[width](#)**: Ancho de un elemento.
- **[height](#)**: Alto de un elemento.
- **[vertical-align](#)**: Alineamiento vertical dentro de un elemento.
- **[margin](#)**: Espacio que se añade entre el elemento y sus vecinos. Se puede diferencia por lado (arriba, abajo, izquierda, derecha).
- **[padding](#)**: Relleno interior para generar espacio alrededor del contenido de un elemento, dentro de los bordes definidos. A diferencia de margin, cuenta para el tamaño del elemento.
- **[float](#)**: Mueve el elemento todo lo posible hacia el lado indicado. Esta propiedad se usa en el posicionamiento flotante de CSS. El tema del posicionamiento en CSS no es trivial y conviene estudiar cómo funciona antes de usar esta propiedad.

- **Border:** La propiedad **border** se utiliza para establecer el mismo grosor, estilo y/o anchura de todos los bordes de un elemento. Al contrario que las propiedades **margin** y **padding**, con la propiedad **border** no es posible indicar diferentes valores para cada uno de los cuatro bordes.



## 2. Fuentes y texto

- **font-family:** Tipo de letra
- **font-size:** Tamaño de letra
- **font-weight:** Peso (normal, negrita, ...)
- **font-style:** Estilo (normal, cursiva, ...)
- **text-decoration:** “Decoraciones” como subrayado, tachado, etc.
- **text-align:** Alineación del texto (izquierda, derecha, etc.)
- **text-transform:** Mostrar un texto en mayúsculas, minúsculas o la primera letra de cada palabra en mayúsculas.

## 3. Color y fondos

- **color:** Color del elemento. Se puede especificar en diferentes formatos como palabras predefinidas (red, green, etc.) RGB o como valor hexadecimal.
- **background-color:** Color del fondo del elemento.
- **background-image:** Permite especificar una imagen de fondo.
- **background-repeat:** Permite usar una imagen a modo de mosaico en diferentes modalidades.
- **box-shadow:** Crear un efecto de sombra para un elemento.

## 4. Listas

- **list-style-image:** Usar la imagen especificada como viñeta para la lista.
- **list-style-type:** Diferentes estilos de viñetas y estilos de numeración para elementos de lista.

- [list-style](#): Establece algunas o todas las propiedades del marcador de los elementos de la lista.

## 5. Bordes

- [border](#): Añade un borde a un elemento y establece algunas propiedades (grosor, estilo de línea, etc.)
- [border-color](#): Color del borde.
- [border-style](#): Diferentes estilos para el borde (sólido, puntos, etc.)
- [border-radius](#): Permite crear esquinas redondeadas para un elemento.

## Propiedad display

- Todos los elementos tienen una forma de comportarse visualmente
- La forma de comportarse lo controla la propiedad CSS display
- Esta propiedad se puede modificar

Es capaz de *modificar* cualquier elemento de tu página de un tipo a otro. Con esta propiedad mágica puedo, por ejemplo, hacer que mis enlaces (que al principio eran elementos en línea) aparezcan en forma de bloque:

Valor	Ejemplos	Descripción
<b>inline</b>	<code>&lt;a&gt;</code> , <code>&lt;em&gt;</code> , <code>&lt;span&gt;</code> ...	Elementos en una línea. Se colocan uno al lado del otro.
<b>block</b>	<code>&lt;p&gt;</code> , <code>&lt;div&gt;</code> , <code>&lt;section&gt;</code> ...	Elementos en forma de bloque. Se encuentran uno debajo del otro y se pueden redimensionar.
<b>inline-block</b>	<code>&lt;select&gt;</code> , <code>&lt;input&gt;</code>	Elemento posicionados uno al lado del otro (como elementos en línea), pero que se pueden redimensionar (como bloques).



none	<head>	Elementos no mostrados.
------	--------	-------------------------

## Valores para la propiedad display

**display:inline** Elemento cuyo contenido se muestra en la misma línea

- Admite márgenes, padding, bordes,...
- **No** admite propiedades width, height ni de posicionamiento
- Ejemplos (por defecto):
  - strong
  - em
  - mark
  - q
  - span

**display:block** Elemento que genera un rectángulo

- Aparece separado
- Por defecto se expande en todo el ancho de su contenedor
- Permite modificar todas las propiedades
- Ejemplos (por defecto):
  - div
  - p
  - h1
  - section
  - article
  - ul

**display:inline-block** Genera un rectángulo con el contenido

- El tamaño se adapta al contenido

- No aparece separado
- Permite modificar width y height
- Admite modificar float
- No permite modificar las propiedades de posicionamiento
- Ejemplos (por defecto):
  - img
  - input
  - textarea
  - button
  - video

Aquí van algunas cosas que recordar sobre los elementos *inline-block*:

- Están colocados uno al lado del otro .
- Se les puede asignar dimensiones específicas.
- **display:none** No se muestra el elemento
- **display:list-item** Hace que el elemento sea un ítem de lista
  - Ejemplo:

Li
- **Otros**
  - table. Comportamiento de tabla (como <table>)
  - table-row. Comportamiento de fila de tabla (como <tr>)
  - table-cell. Comportamiento de celda (como <td>)
  - table-column. Comportamiento de columna (como <col>)

- table-column-group- Como <colgroup>
- table-header-group- Como <thead>
- table-footer-group- Como <tfoot>
- table-row-group- Como <tbody>

## Posicionamiento Absoluto, Fijo y Relativo

Existen algunas técnicas bastante especiales que se emplean para posicionar elementos con precisión en la página:

- Posicionamiento absoluto: nos permite posicionar un elemento en cualquier parte de la página (superior izquierda, inferior derecha, centro, etc.).
- Posicionamiento fijo: es lo mismo que el posicionamiento absoluto, pero aquí el elemento permanece visible incluso si te desplazas hacia la parte inferior de la página.
- Posicionamiento relativo: permite al elemento estar fuera de su posición normal.

Los posicionamientos absolutos, fijos y relativos también funcionan en etiquetas en línea.

Sin embargo, verás que es más frecuente usarlos en etiquetas de bloque que en etiquetas en línea.

En primer lugar tendrás que elegir una de las tres opciones de posicionamiento. Para ello, se utiliza la propiedad CSS **position** y se le da uno de estos valores:

- **absolute**: posicionamiento absoluto;
- **fixed**: posicionamiento fijo;
- **relative**: posicionamiento relativo.

Pasaremos a examinar cada una de estas opciones de posicionamiento.

## Posicionamiento absoluto

El posicionamiento absoluto permite situar un elemento en cualquier lugar de la página. Para situar un elemento de manera absoluta has de introducir:

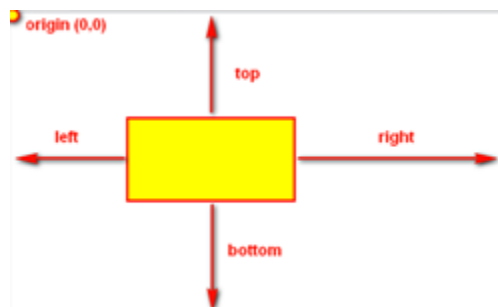
```
element
{
    position: absolute;
}
```

¡Pero no basta con esto! Aunque hemos especificado que el posicionamiento ha de ser absoluto, hemos de indicar *en qué parte de la página queremos que se sitúe el bloque*.

Para ello, vamos a utilizar cuatro propiedades CSS:

- **left:** posición relativa en la izquierda de la página;
- **right:** posición relativa en la derecha de la página;
- **top:** posición relativa en la parte superior de la página;
- **bottom:** posición relativa en el pie de la página;

Puedes darles un valor en píxeles, como 14px, o un porcentaje, como 50%.



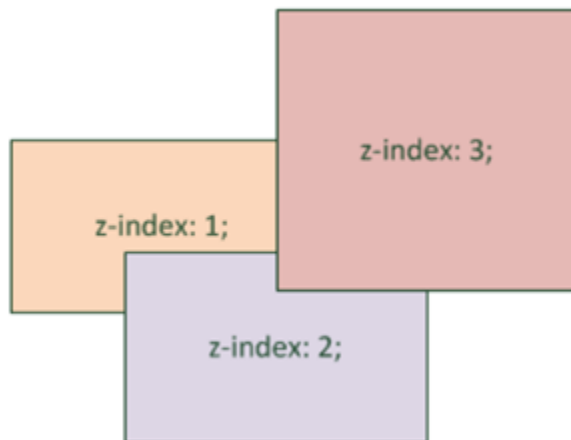
Si por ejemplo introduces:

```
element
{
    position: absolute;
    right: 0px;
    bottom: 0px;
}
```

Significa que el bloque se situará a la derecha, en la parte inferior derecha de la página (0 píxeles tomando como referencia la derecha de la página, 0 tomando la parte inferior de la página), es decir, el elemento se sitúa en la esquina inferior derecha de la pantalla .

Los elementos posicionados de manera absoluta están sobre el resto de elementos de la página. Además, si situas dos elementos de manera absoluta en el mismo lugar es posible que se solapen. En estos casos, utiliza la propiedad **z-index** para especificar qué elemento debería aparecer encima de los otros elementos.

El elemento con el valor **z-index** más alto se situará por encima de los otros, tal y como se muestra en la siguiente ilustración.



**Una pequeña aunque importante aclaración técnica:** el posicionamiento absoluto no siempre funciona tomando como referencia la esquina superior izquierda de la ventana. Si utilizas el posicionamiento absoluto para situar un bloque A dentro de otro bloque B, que se ha posicionado de manera absoluta, relativa o fija, tu bloque A se posicionará tomando como referencia la esquina superior izquierda del bloque B.

## Posicionamiento fijo

El principio es *exactamente el mismo* que el del posicionamiento absoluto, salvo que esta vez el bloque queda fijo en su posición incluso si te desplazas hacia el pie de la página.

## Posicionamiento relativo

El posicionamiento relativo es algo más complicado y puede dar con facilidad algunas complicaciones. Este tipo de posicionamiento te permite realizar «ajustes»: **el elemento está fuera de su posición normal**.

Tomemos como ejemplo un texto importante situado entre dos etiquetas **<strong>**. Para empezar, lo pondré sobre un fondo rojo para poder identificarlo mejor.

```
strong
{
  background-color: red; /* Fondo rojo */
  color: yellow; /* Texto amarillo */
}
```

El punto de coordenadas (0, 0) ya no se encuentra en la parte superior izquierda de la venta como en el caso anterior. No, esta vez el origen está situado en la parte superior izquierda... de la posición actual de tu elemento

El diagrama muestra el texto "No hay duda este texto es importante si quieres entender" donde "este texto es importante" está resaltado en rojo. Una etiqueta "Point of origin (0,0)" con un punto rojo indica que el origen de las coordenadas está ahora sobre el texto resaltado, desplazado de su posición original superior izquierda.

No hay duda **este texto es importante** si quieres entender

Así pues, si introduces **position: relative;** y aplicas cualquiera de las propiedades **top**, **left**, **right** o **bottom**, el texto sobre fondo rojo se moverá tomando como referencia su posición actual.

Utilicemos un ejemplo: quiero desplazar mi texto 55 píxeles a la derecha y 10 píxeles hacia abajo; así, lo que le voy a indicar es que quiero que se sitúe a 55 píxeles del borde izquierdo y a 10 píxeles del borde superior.

```

strong
{
  background-color: red; /* Fondo rojo */
  color: yellow; /* Texto amarillo */

  position: relative;
  left: 55px;
  top: 10px;
}

```

El texto estará fuera de su posición inicial, tal como se indica en la siguiente imagen



## Flotación: Propiedad float

- Normalmente los elementos HTML aparecen en la posición indicada en el código
- Dependiendo de la propiedad display, puede haber otros elementos en la misma línea (inline o inline-box) o aparte (box)
- Sin embargo, **podemos indicar de qué forma deben aparecer respecto a su alrededor. Un elemento “flota” si admite que el texto fluya a su alrededor, la propiedad que lo consigue es float.**

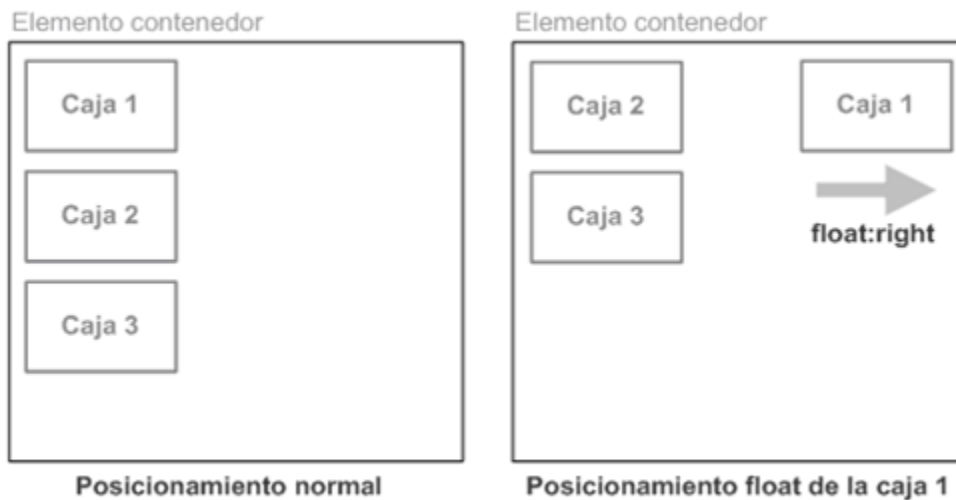
Valores

- float:none
- float:left
- float:right

El posicionamiento flotante es el más difícil de comprender pero al mismo tiempo es el más utilizado. La mayoría de estructuras de las páginas web complejas están diseñadas con el posicionamiento flotante.

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.

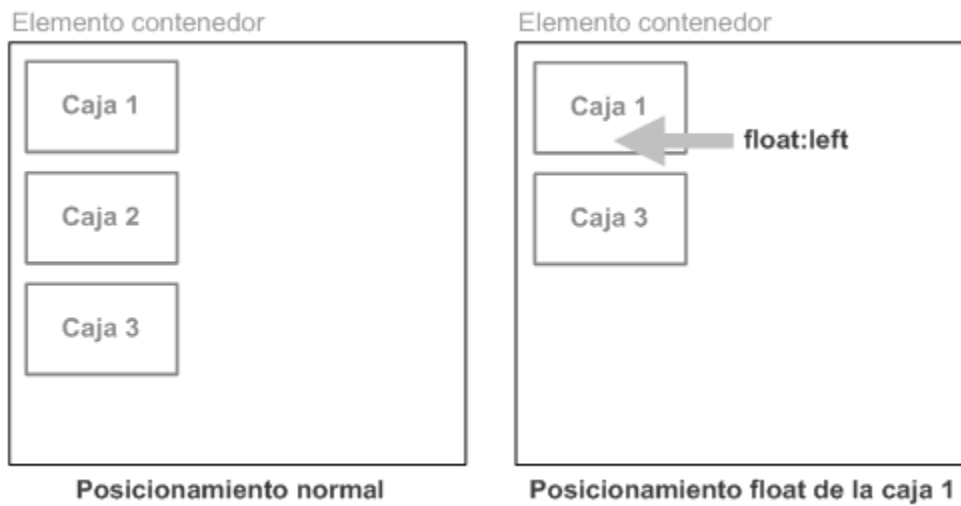
La siguiente imagen muestra el resultado de posicionar de forma flotante hacia la derecha la caja



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:

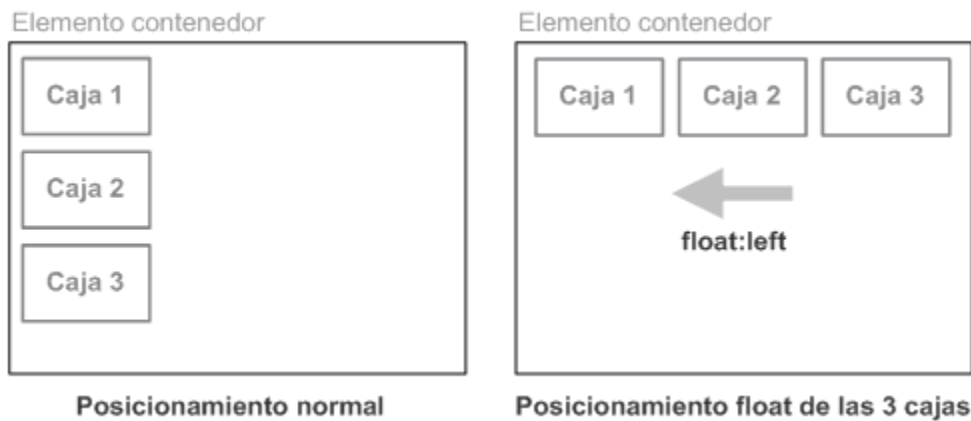




La caja 1 es de tipo flotante, por lo que desaparece del flujo normal de la página y el resto de cajas ocupan su lugar. El resultado es que la caja 2 ahora se muestra donde estaba la caja 1 y la caja 3 se muestra donde estaba la caja 2.

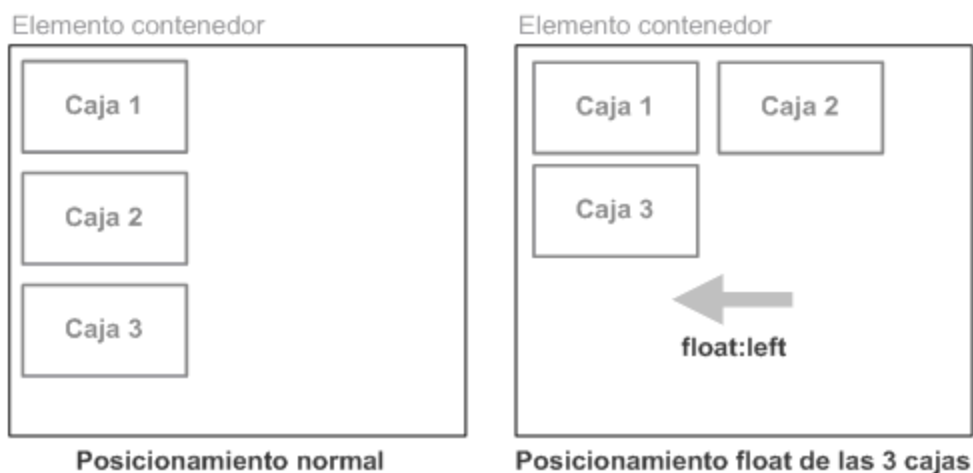
Al mismo tiempo, la caja 1 se desplaza todo lo posible hacia la izquierda de la posición en la que se encontraba. El resultado es que la caja 1 se muestra encima de la nueva posición de la caja 2 y tapa todos sus contenidos.

Si existen otras cajas flotantes, al posicionar de forma flotante otra caja, se tiene en cuenta el sitio disponible. En el siguiente ejemplo se posicionan de forma flotante hacia la izquierda las tres cajas:



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:

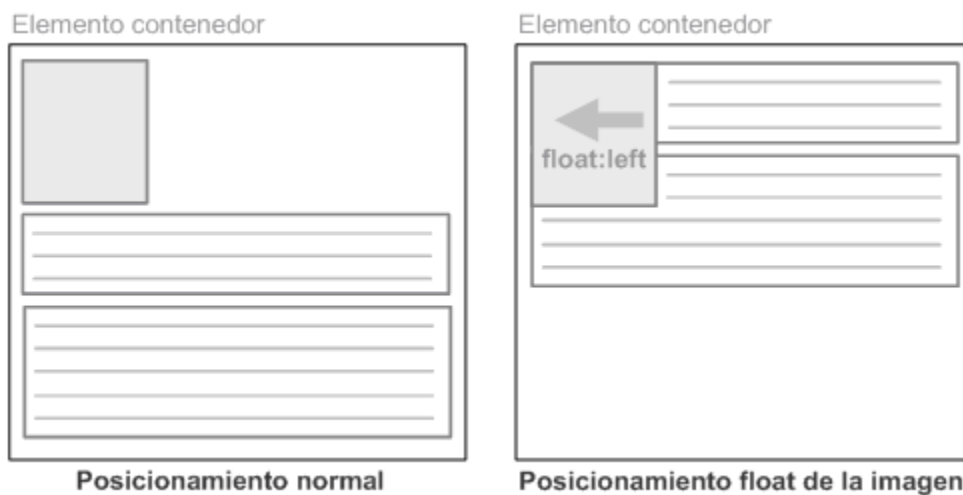


Las cajas flotantes influyen en la disposición de todas las demás cajas. Los elementos en línea hacen sitio a las cajas flotantes adaptando su anchura al espacio libre dejado por la caja desplazada. Los elementos de bloque no les hacen sitio, pero sí que adaptan sus contenidos para que no se solapen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



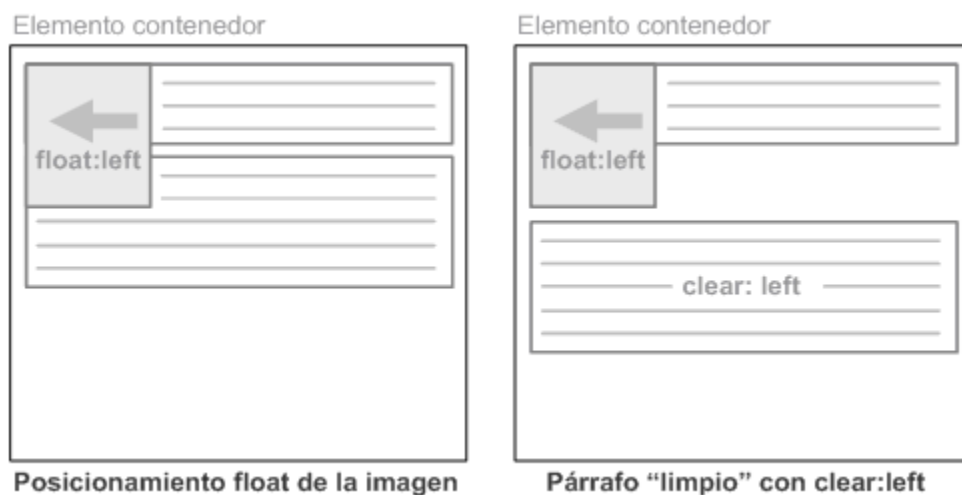
La regla CSS que se aplica en la imagen del ejemplo anterior es:

```
img {  
  float: left;  
}
```

Uno de los principales motivos para la creación del posicionamiento float fue precisamente la posibilidad de colocar imágenes alrededor de las cuales fluye el texto.

CSS permite controlar la forma en la que los contenidos fluyen alrededor de los contenidos posicionados mediante float. De hecho, en muchas ocasiones es admisible que algunos

contenidos fluyan alrededor de una imagen, pero el resto de contenidos deben mostrarse en su totalidad sin fluir alrededor de la imagen:



La propiedad **clear** permite modificar el comportamiento por defecto del posicionamiento flotante para forzar a un elemento a mostrarse debajo de cualquier caja flotante. Indica el lado del elemento que no debe ser adyacente a ninguna caja flotante.


La regla CSS que se aplica al segundo párrafo del ejemplo anterior es la siguiente:

```
<p style="clear: left;">...</p>
```

La **propiedad clear** indica el lado del elemento HTML que no debe ser adyacente a ninguna caja posicionada de forma flotante. Si se indica *el valor left*, el elemento se desplaza de forma descendente hasta que pueda colocarse en una línea en la que no haya ninguna caja flotante en el lado izquierdo.

La especificación oficial de CSS explica este comportamiento como "un desplazamiento descendente hasta que el borde superior del elemento esté por debajo del borde inferior de cualquier elemento flotante hacia la izquierda".

Si se indica **el valor right**, el comportamiento es análogo, salvo que en este caso se tienen en cuenta los elementos desplazados hacia la derecha.



El valor **both** despeja los lados izquierdo y derecho del elemento, ya que desplaza el elemento de forma descendente hasta que el borde superior se encuentre por debajo del borde inferior de cualquier elemento flotante hacia la izquierda o hacia la derecha.

## En resumen

- CSS se utiliza para el diseño de página de un sitio web. Tenemos a nuestra disposición diferentes técnicas.
- El posicionamiento flotante (utilizando la propiedad **float**) es una de las técnicas más utilizadas actualmente. Te permite, por ejemplo, situar un menú a la izquierda o a la derecha de la página. Esta propiedad, sin embargo, no se diseñó en un principio con este fin, y, de ser posible, será mejor evitar esta técnica.
- El posicionamiento **inline-block** consiste en asignar el tipo **inline-block** a nuestros elementos mediante la propiedad **display**. Se comportarán como elementos en línea (posicionamiento izquierda a derecha), pero podrán ser redimensionados como bloques (mediante **width** y **height**). Esta técnica es preferible a la de posicionamiento flotante.
- El posicionamiento absoluto te permite situar un elemento en cualquier lugar de la página, indicando la distancia en píxeles.
- El posicionamiento fijo es parecido al posicionamiento absoluto, pero el elemento permanece visible incluso si te desplazas hacia el pie de la página.
- El posicionamiento relativo permite que un bloque esté fuera de su posición normal.
- Un elemento A, posicionado de manera absoluta dentro de otro elemento B (a su vez posicionado de modo absoluto, fijo o relativo), se posicionará tomando como referencia este elemento B y no la esquina superior izquierda de la página.