INTRODUCCIÓN A CSS

4. Layout

© Francisco Pérez
ies_paco@yahoo.es

CONTENIDO

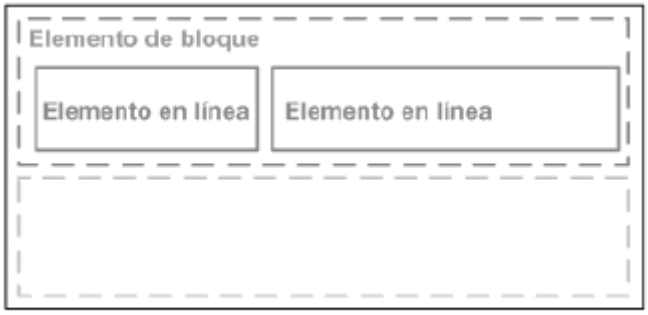
1. Introducción

Los navegadores crean y posicionan automáticamente todas las cajas que forman cada página (X)HTML. Sin embargo, CSS permite al diseñador modificar la posición en la que se muestra cada caja. Esto posibilita la creación de estructuras complejas y diseños avanzados, que de otra forma no serían posibles. La operación de diseño de una estructura de este tipo recibe el nombre de layout o posicionamiento de cajas.

2. Tipos de posicionamiento

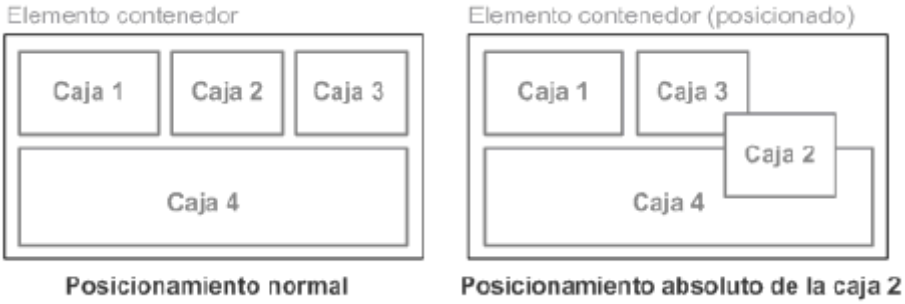
CSS define cinco modos diferentes para posicionar una caja. La diferencia entre los distintos modos de posicionamiento reside en el punto que se toma como origen de coordenadas para posicionar la caja:

- ***Posicionamiento normal o estático:*** se trata del posicionamiento que utilizan los navegadores por defecto. Las etiquetas se muestran según el orden en el que aparecen en el código fuente. A este orden se le denomina el ***flujo normal de la página.*** Normalmente la anchura de los elementos de bloque está limitada a la anchura de su elemento contenedor, aunque en algunos casos el contenido los contenidos pueden desbordar el espacio disponible. Si las cajas en línea ocupan más espacio del disponible en su propia línea, el resto de cajas se muestran en las líneas inferiores.



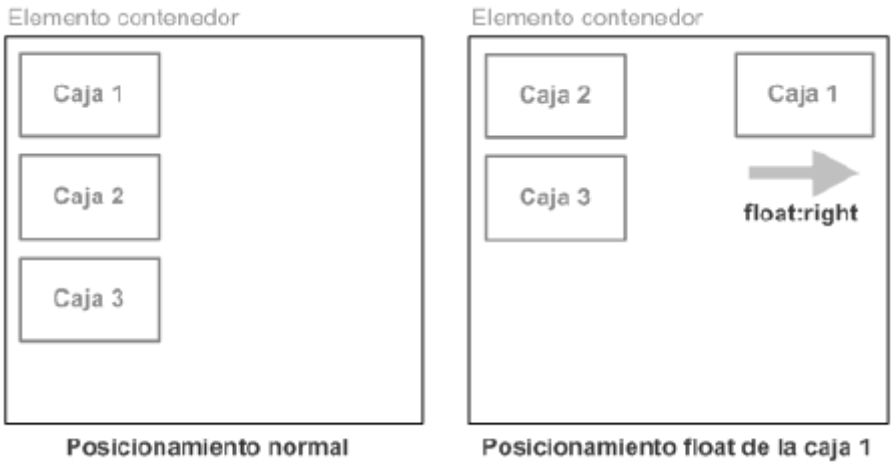
- ***Posicionamiento relativo***: variante del posicionamiento normal que consiste en posicionar una caja según el posicionamiento normal, tomando este punto como origen de coordenadas, y después se desplaza la caja con respecto a dicha posición original. Después de desplazar una caja con posicionamiento relativo, las restantes cajas de la página se mantienen en sus posiciones. La caja con posicionamiento relativo puede solapar a las restantes cajas de la página.

- ***Posicionamiento absoluto:*** en este modo de posicionamiento se toma como origen de coordenadas la esquina superior izquierda del **elemento contenedor más inmediato que tenga posicionamiento distinto del posicionamiento normal** y a partir de este punto se desplaza la caja hasta las coordenadas deseadas. Después de desplazar una caja con posicionamiento absoluto, las restantes cajas de la página ocupan (si es posible) el lugar dejado libre por la caja con posicionamiento absoluto. Se dice que la caja con posicionamiento absoluto abandona el ***flujo normal de la página***. El comportamiento es el mismo que si la caja estuviera en un plano por encima del resto de elementos de la página. Por tanto, es posible que se produzcan solapamientos.

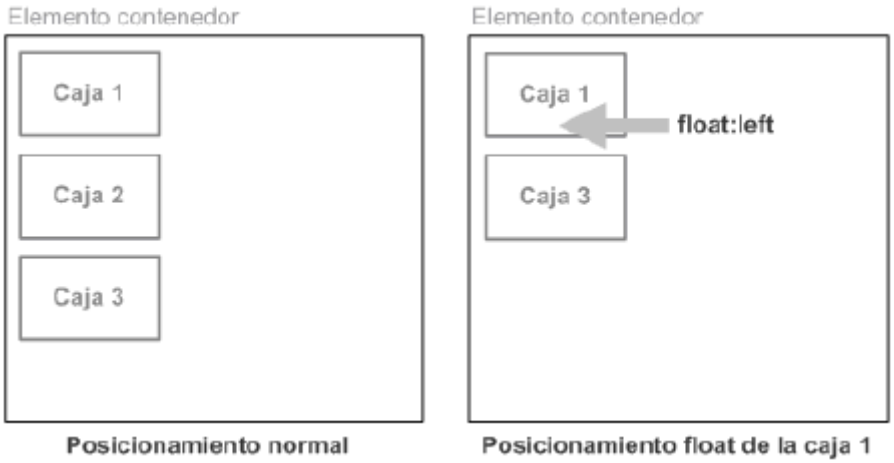


- ***Posicionamiento fijo:*** variante del posicionamiento absoluto que convierte una caja en un elemento inamovible, de forma que su posición en la pantalla siempre es la misma independientemente del resto de elementos e independientemente de si el usuario sube o baja la página en la ventana del navegador. Resulta útil para mostrar elementos como una cabecera o un pie de página siempre visible. El origen de coordenadas se establece de la misma forma que en el posicionamiento absoluto y, al igual que en dicho posicionamiento, la caja con posicionamiento fijo sale del flujo normal de la página.

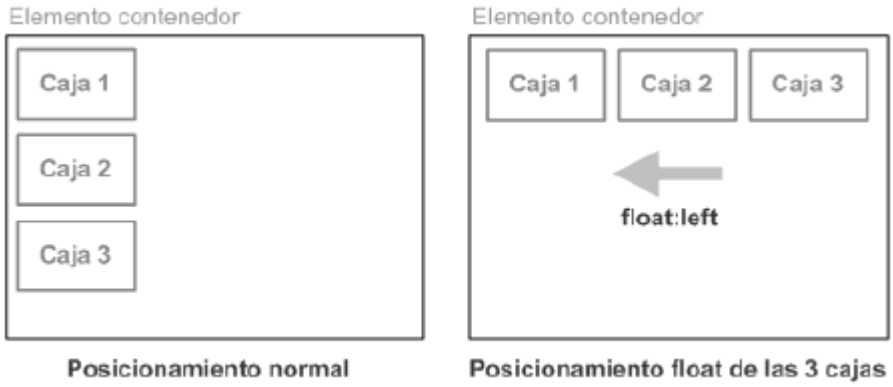
- ***Posicionamiento flotante:*** se trata del modelo más especial de posicionamiento y posiblemente el más utilizado, ya que desplaza las cajas todo lo posible hacia la izquierda o hacia la derecha de la línea en la que se encuentran. Al aplicar a una caja el posicionamiento flotante, dicha caja sale del flujo normal de la página y las restantes cajas no flotantes fluyen alrededor de la caja flotante. En siguiente ejemplo, se ha aplicado un posicionamiento flotante (hacia la derecha) a la caja 1:



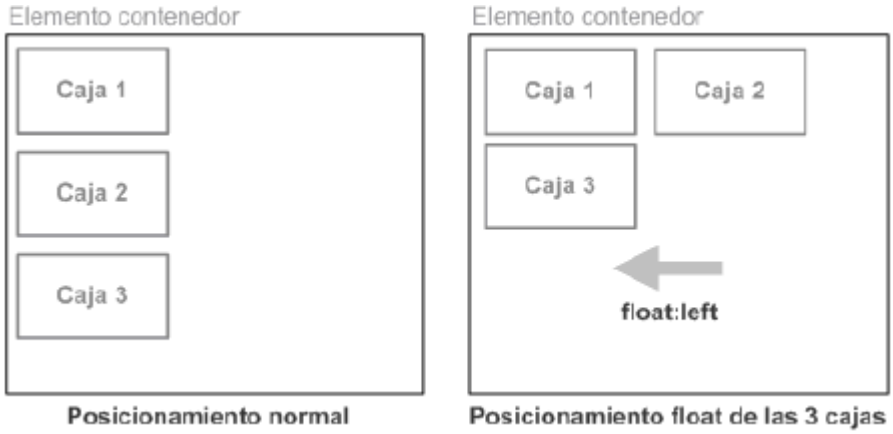
En el siguiente ejemplo, se aplica posicionamiento flotante (hacia la izquierda) a la caja 1 por lo que dicha caja sale fuera del flujo normal de la página y se alinea a la izquierda. Como la caja 1 sale fuera del flujo normal de la página, las restantes cajas ocupan el hueco que deja libre y la caja 1 solapa a la caja 2.



Por último, una característica del posicionamiento flotante es que todas las cajas con posicionamiento flotante al salir del flujo normal de la página, se sitúan en un plano superior superpuesto y en dicho plano se vuelve a establecer un flujo normal de página entre las cajas que se encuentran en dicho plano. En el siguiente ejemplo se ha aplicado posicionamiento flotante a las tres cajas (hacia la izquierda) por lo que las tres cajas salen fuera del flujo normal de la página, y forman un nuevo flujo normal en un plano superior, de manera que no hay solapamiento entre ellas:



En el supuesto anterior, si no existiera sitio en la línea para albergar las cajas con posicionamiento float, la caja más a la derecha bajaría a la línea siguiente hasta encontrar el espacio necesario para mostrarse lo más ala izquierda posible:



3. Establecimiento del modo de posicionamiento

El modo de posicionamiento de una caja se establece mediante la propiedad ***position:***

position	Establece el modo de posicionamiento de la caja
Se aplica a:	Todas las etiquetas
Valores admitidos:	**static** | relative | absolute | fixed | inherit

La propiedad position sólo indica el modo de posicionamiento de una caja pero no la desplaza. El valor ***static*** es el valor por defecto y por tanto se corresponde con el posicionamiento normal.

La propiedad position no permite controlar el modo de posicionamiento float, por lo que para establecer dicho modo se utiliza la propiedad ***float:***

float	Establece el modo posicionamiento float
Se aplica a:	Todas las etiquetas
Valores admitidos:	left| right | **none** |inherit

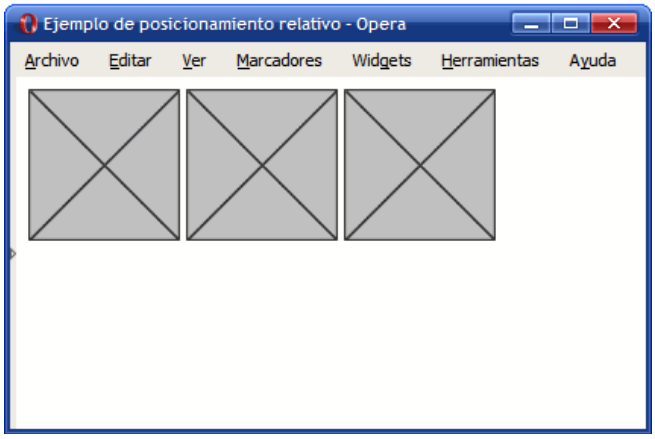
4. Establecimiento del desplazamiento

El modo de posicionamiento únicamente indica al navegador si la caja sale del flujo normal y el origen de coordenadas respecto al cual posicionarla. Sin embargo, el modo de posicionamiento por sí sólo no indica el desplazamiento respecto a dicho origen de coordenadas. Para hacer que una caja se desplace respecto al origen de coordenadas se utiliza una combinación de las propiedades ***top, bottom, right*** y ***left:***

top,bottom,right,left	Establecen el desplazamiento del borde superior, inferior, derecho o izquierdo (respectivamente) respecto al origen de coordenadas.
Se aplica a:	Todas los elementos **posicionados.**
Valores admitidos:	<unidades relativas o absolutas>| **auto** |inherit

Las propiedades anteriores sólo se aplican a ***elementos posicionados***, es decir, aquéllos elementos para los que se ha establecido la propiedad ***position***. La propiedad ***left*** desplaza la caja hacia su derecha, la propiedad ***right*** la desplaza hacia su izquierda, la posición ***top*** desplaza la caja de forma descendente y la propiedad ***bottom*** desplaza la caja de forma ascendente. Si se utilizan valores negativos en estas propiedades, su efecto es justamente el inverso. Siempre se cumple left = -right y top = -bottom por lo que para desplazar una caja normalmente sólo se usan las propiedades ***top*** y ***bottom***.

Por ejemplo, aplicando posicionamiento relativo, se desplaza la primera imagen de forma descendente:




```
img.desplazada
       {
       position: relative;
       top: 8em;
       }

<img class="desplazada" src="img/image.png" alt="dummy image" />
<img src="img/image.png" alt="dummy image" />
<img src="img/image.png" alt="dummy image" />
```

5. La propiedad ***clear***

Normlamente, las cajas no flotantes fluyen alrededor de una caja con posicionamiento float. La propiedad clear permite modificar este comportamiento para forzar a un elemento a mostrarse debajo de cualquier caja flotante:

clear	Indica el lado de la caja que no debe ser adyacente a una caja flotante.
Se aplica a:	Todas las etiquetas de bloque
Valores admitidos:	both | left| right | **none** |inherit

Como puede verse, 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.

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.

A continuación se muestra el funcionamiento de esta propiedad. Supongamos que queremos diseñar la siguiente estructura:

Para generar este diseño creamos un div que representará el recuadro gris del fondo. Por otro lado, los elementos “<< Anterior” y “Siguiente >>” los encerramos en una etiqueta span para poder aplicarles estilos de manera independiente. Comenzamos con el siguiente código fuente:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
        <head>
                <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
                <title>Ejercicio posicionamiento float</title>
                <style type="text/css">

                        div#paginacion
                                {
                                border: 1px solid #CCC;
                                background-color: #E0E0E0;
                                padding: .5em;
                                }

                        .derecha
                                {
                                float: right;
                                }

                        .izquierda
                                {
                                float: left;
                                }

                </style>
        </head>
```

```
        <body>
                <div id="paginacion">
                        <span class="izquierda">
                                &laquo; Anterior
                        </span>

                        <span class="derecha">
                                Siguiente &raquo;
                        </span>
                </div>
        </body>
</html>
```

Sin embargo, si probamos el código anterior en el navegador obtenemos lo siguiente:

Lo que ocurre es que los dos elementos creados dentro del se han posicionado mediante float y por tanto han salido del flujo normal del documento. De este modo, el elemento no tiene contenidos y su altura no cubre a los elementos (que, a efectos prácticos, al estar fuera del flujo normal, no se consideran contenidos dentro del).

Para solucionar esta situación podemos incluir dentro del otro nuevo elemento invisible para que el div contenedor aumente su altura. A este nuevo podemos establecerle la propiedad ***clear*** para garantizar que se va a mostrar por debajo de las etiquetas . A esta operación de añadir un elemento invisible para que el fondo se extienda más allá de los elementos flotantes se la denomina “limpiar el float”. Además de un elmeento invisible, también se puede utilizar un <p> invisible o un <hr>.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
	<head>
		<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
		<title>Ejercicio posicionamiento float</title>
		<style type="text/css">

			div#paginacion
				{
				border: 1px solid #CCC;
				background-color: #E0E0E0;
				padding: .5em;
				}

			.derecha
				{
				float: right;
				}

			.izquierda
				{
				float: left;
				}

			div.clear
				{
				clear: both;
				}
		</style>
	</head>

	<body>
		<div id="paginacion">
			<span class="izquierda">
				&laquo; Anterior
			</span>

			<span class="derecha">
				Siguiente &raquo;
			</span>

		<div class="clear"></div>
		</div>
	</body>
</html>
```

6. Centrar una página horizontalmente

A medida que aumenta el tamaño y la resolución de las pantallas de ordenador, se hace más difícil diseñar páginas que se adapten al tamaño de la ventana del navegador. El principal reto que se presenta con resoluciones superiores a 1024 x 768 píxel, es que las líneas de texto son demasiado largas como para leerlas con comodidad. Por ese motivo, normalmente se opta por diseños con una anchura fija limitada a un valor aceptable para mantener la legibilidad del texto.

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. Las siguientes imágenes muestran el aspecto de una página centrada a medida que aumenta la anchura de la ventana del navegador.

Utilizando la propiedad margin de CSS, es muy sencillo centrar una página web horizontalmente. La solución consiste en 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 (en inglés se denomina wrapper o container). 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.

6.1. Diseño líquido vs diseño fijo

Cuando se define la anchura del div contenedor de la página en porcentaje se consigue un efecto denominado diseño líquido o diseño fluido. Este efecto consiste en que el ancho del div contenedor se ajusta a la resolución de cada pantalla para mantener siempre las mismas proporciones.

El inconveniente del diseño líquido radica en que si la página llega a visualizarse en un dispositivo con resolución muy baja puede dar lugar a que el texto y los demás contenidos no quepan dentro del div contenedor y se desborden, causando una visualización desesctructurada. Del mismo modo, si se visualiza en una pantalla con una resolución demasiado alta, las líneas pueden hacerse demasiado largas para su lectura y el aspecto puede verse deformado.

Si, por el contrario, el tamaño del div contenedor se especifica en unidades relativas que no sean porcentaje (por ejemplo en píxeles), se obtiene un mayor control sobre la visualización en cualquier circunstancia, ya que el tamaño de las cajas será siempre el mismo. El inconveniente de este diseño radica en que al pasar a resoluciones muy altas el tamaño del div contenedor puede verse demasiado pequeño. El caso inverso también es posible: con una resolución muy baja el div contenedor puede ser de mayor tamaño que el ancho de pantalla y obligar a utilizar el scroll horizontal.

Muchos sitios utilizan una combinación de diseño fluido para el div contenedor y diseño fijo para los distintos div que conforman la estructura interna de la página. A

esto hay que sumar la disposición de una hoja de estilos alternativa para ajustar la visualización en dispositivos móviles.

Por todo ello, para el diseñador web resulta imprescindible manejar estadísticas acerca de las principales resoluciones de pantalla del público objetivo del sitio web.

7. Modos de visualización

CSS incluye las propiedades ***display*** y ***visibility*** que permiten mostrar u ocultar las cajas. Estas propiedades suelen utilizarse en scripts para crear efectos dinámicos en diseños web complejos donde se requieren efectos avanzados y animaciones:

`display`	Controla el modo de visualización de una caja.
Se aplica a:	Todas las etiquetas.
Valores admitidos:	**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

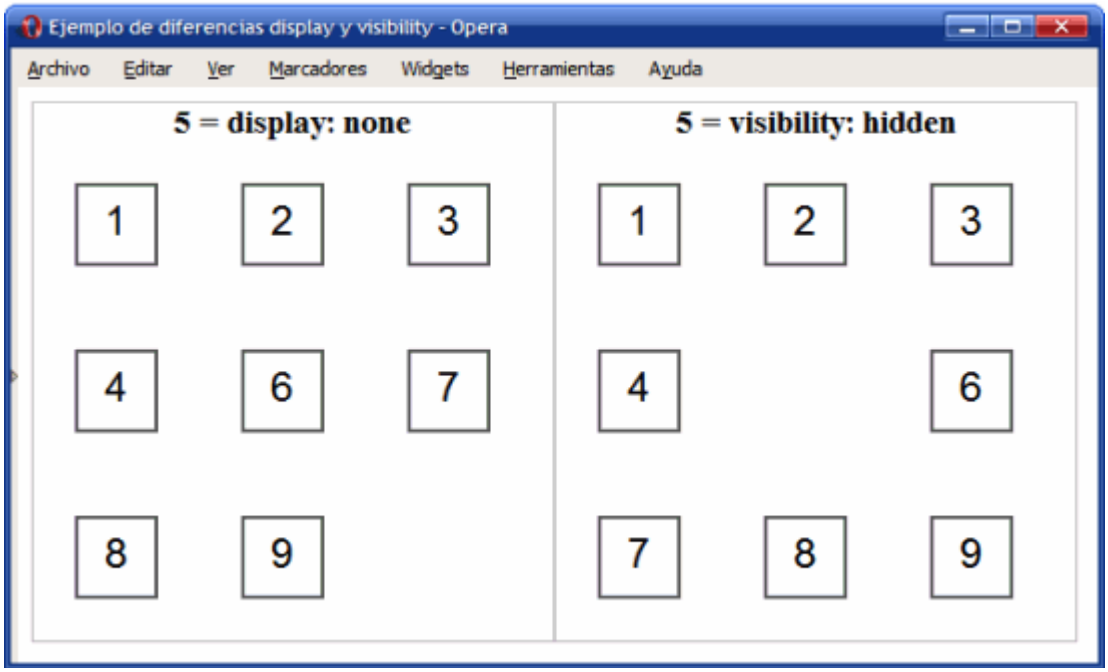
`visibility`	Controla la visibilidad de una caja.
Se aplica a:	Todas las etiquetas.
Valores admitidos:	**visible** | hidden | collapse | inherit

Ambas propiedades, ***display*** y ***visibility***, controlan la visualización de los elementos:

- La propiedad ***display*** permite ocultar completamente un elemento haciendo que **salga del flujo normal del documento** y desaparezca de la página. Como el elemento que se ha ocultado sale fuera del flujo normal, los restantes elementos ocupan el lugar dejado libre.

- La propiedad ***visibility*** también permite hacer invisible un elemento, pero, a diferencia de la propiedad anterior, dicho elemento no sale fuera del flujo normal de la página por lo que las restantes etiquetas mantienen su posición en la página.

La siguiente imagen muestra la diferencia entre ocultar la caja número 5 mediante la propiedad display u ocultarla mediante la propiedad visibility:



Además de permitir mostrar u ocultar elementos, la propiedad ***display*** modifica la forma en la que se visualiza un elemento. Por ejemplo, el valor **block** muestra un elemento como si fuera un elemento de bloque, independientemente del tipo de elemento que se trate. Del mismo modo, el valor **inline** visualiza un elemento en forma de elemento en línea. Por otro lado, el valor **none** oculta un elemento tal y como hemos visto anteriormente.

La regla **display: inline** resulta útil al ser aplicada a listas (etiquetas <ul> y <ol>) que quieran mostrar horizontalmente y la propiedad **display: block** se emplea frecuentemente para los enlaces que forman el menú de navegación.

8. Desbordamiento

En ocasiones el contenido de un elemento no cabe en el espacio reservado para ese elemento y dicho contenido se desborda. La situación más habitual en la que el contenido sobresale de su espacio reservado es 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`	Controla el modo de desbordamiento de un elemento.
Se aplica a:	Elementos de bloque y celdas de tablas.
Valores admitidos:	**visible** | hidden | scroll | auto | inherit

Los valores de la propiedad overflow tienen el siguiente significado:

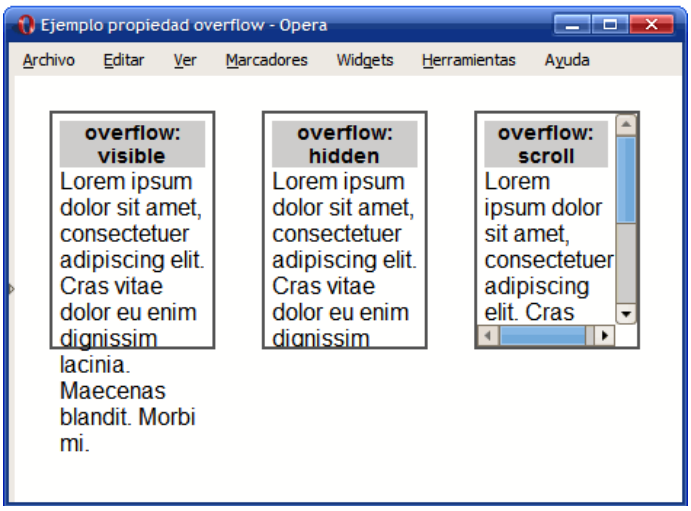
- ***visible***: el contenido no se corta y se muestra sobresaliendo la zona reservada para visualizar el elemento. Este es el comportamiento por defecto.

- ***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.

La siguiente imagen muestra un ejemplo de los tres valores típicos de la propiedad overflow:



9. La propiedad *z-index*

Al utilizar posicionamiento relativo o absoluto es posible solapar múltiples cajas. La propiedad ***z-index*** controla dicho solapamiento, permitiendo indicar qué capa se muestra delante o detrás respecto a otra.

z-index	Establece la posición de las cajas en el eje Z.
Se aplica a:	Elementos posicionados.
Valores admitidos:	**auto** | <número> |inherit

Aunque la especificación oficial admite números negativos, generalmente 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. Por ejemplo, un elemento con un **z-index: 10** se muestra por encima de los elementos con **z-index: 8**, pro por debajo de otro elemento con **z-index: 20.**