ESEN

30 AÑOS Tu punto de partida

Clase 5
Posicionamiento y visualización

Desarrollo Web I

Introducción (1)

Para crear una buena composición de los elementos en pantalla, es necesario conocer todas las opciones posibles al posicionamiento de elementos y, sobre todo, tener algunas nociones de los tipos de elementos con los que trabaja CSS a la hora de posicionarlos en pantalla.

Para cumplir con el *box model* los navegadores crean una caja para representar cada elemento de una página HTML. Para hacerlo, los navegadores llevan a cabo una compleja operación en la que se tienen en cuenta los siguientes parámetros:

1. Las propiedades `width` (ancho) y `height` (alto) de las cajas, en caso de haber especificado unas dimensiones especificas.

Introducción (2)

2. El tipo de cada elemento HTML, bien si es un elemento de bloque o un elemento de línea.
3. Posicionamiento de las cajas (estático, relativo, absoluto, fijo o flotante).
4. El parentesco y descendencia de cada elemento.
5. Las dimensiones de la ventana del navegador en función de la resolución de pantalla o las dimensiones de las imágenes, entre otros.

A este procedimiento se le conoce comúnmente como "*modelo de formato visual*", que definido para entenderlo rápidamente puede ser algo así como el modelo que siguen los navegadores para procesar el contenido de un documento para los medios visuales.

Tipos de elementos

Existen dos grandes clasificaciones de los elementos HTML que distingue un navegador:

- **Elementos de bloque**

Los elementos de bloque ("*block elements*" en inglés) siempre empiezan en una nueva línea, ocupan todo el espacio disponible hasta el final de la línea y pueden contener otros elementos de bloque o de línea.

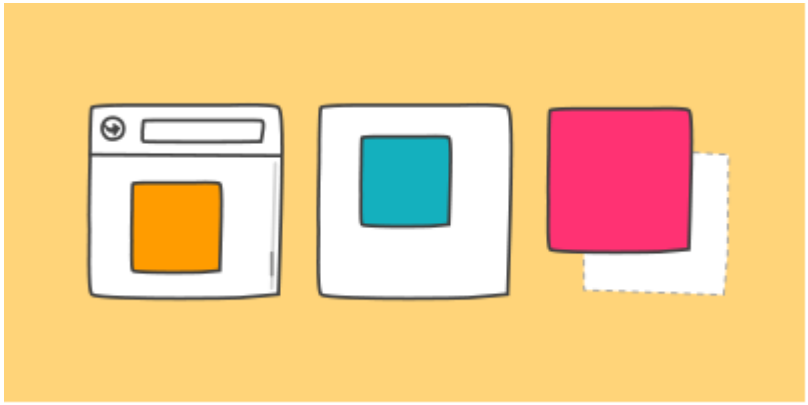
- **Elementos de línea**

Los elementos de línea ("*inline elements*" en inglés) no empiezan necesariamente en nueva línea, sólo ocupan el espacio necesario para mostrar sus contenidos, y solo pueden contener otros elementos de línea o contenido.

Ejemplos de elementos de bloque y de línea

- Algunos ejemplos de elementos de bloque son: *header*, *nav*, *aside*, *section*, *article*, *div*, *p*, *footer*, *h1*, *h2*, *h3*, *h4*, *h5*, *h6*, etc.

- Por su parte, algunos ejemplos de elementos de línea son: *a*, *b*, *i*, *br*, *cite*, *img*, *span*, *sup*, *sub*, etc.



POSICIONAMIENTO

Posicionamiento(1)

Los navegadores crean y posicionan de forma automática todas las cajas que forman cada página HTML. No obstante, CSS permite al diseñador modificar la posición en la que se muestra cada caja.

Utilizando las propiedades que proporciona CSS para alterar la posición de las cajas es posible realizar efectos muy avanzados y diseñar estructuras de páginas que de otra forma no serían posibles.

Posicionamiento(2)

El estándar de CSS define cinco modelos diferentes para posicionar una caja:

- **Posicionamiento normal o estático.** Se trata del posicionamiento que utilizan los navegadores si no se indica lo contrario.
- **Posicionamiento relativo:** variante del posicionamiento normal que consiste en posicionar una caja según el posicionamiento normal y después desplazarla respecto de su posición original.

Posicionamiento(3)

- **Posicionamiento absoluto:** la posición de una caja se establece de forma absoluta respecto de su elemento contenedor y el resto de elementos de la página ignoran la nueva posición del elemento.
- **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.
- **Posicionamiento flotante:** se trata del modelo más especial de posicionamiento, ya que desplaza las cajas todo lo posible hacia la izquierda o hacia la derecha de la línea en la que se encuentran.

Propiedad position (1)

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

position	Posicionamiento
Valores	`static` | `relative` | `absolute` | `fixed` | `inherit`
Se aplica a	Todos los elementos
Valor inicial	static
Descripción	Selecciona el posicionamiento con el que se mostrará el elemento

Propiedad position (2)

El significado de cada uno de los posibles valores de la propiedad position es el siguiente:

- **static:** corresponde al posicionamiento normal o estático. Si se utiliza este valor, se ignoran los valores de las propiedades *top, right, bottom y left* que se verán a continuación.
- **relative:** corresponde al posicionamiento relativo. El desplazamiento de la caja se controla con las propiedades *top, right, bottom y left*.

Propiedad position (3)

- **absolute:** corresponde al posicionamiento absoluto. 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.
- **fixed:** corresponde al posicionamiento fijo. El desplazamiento se establece de la misma forma que en el posicionamiento absoluto, pero en este caso el elemento permanece inamovible en la pantalla.

La propiedad position no permite controlar el posicionamiento flotante, que se establece con otra propiedad llamada float y que se explica más adelante. Además, la propiedad position sólo indica cómo se posiciona una caja, pero no la desplaza.

Propiedades de desplazamiento (1)

Normalmente, cuando se posiciona una caja también es necesario desplazarla respecto de su posición original o respecto de otro origen de coordenadas. CSS define cuatro propiedades llamadas *top, right, bottom y left* para controlar el desplazamiento de las cajas posicionadas:

top **right** **bottom** **left**	Desplazamiento superior Desplazamiento lateral derecho Desplazamiento inferior Desplazamiento lateral izquierdo
Valores	`<medida> | <porcentaje> | auto | inherit`
Se aplica a	Todos los elementos posicionados
Valor inicial	auto
Descripción	Indican el desplazamiento horizontal y vertical del elemento respecto de su posición original

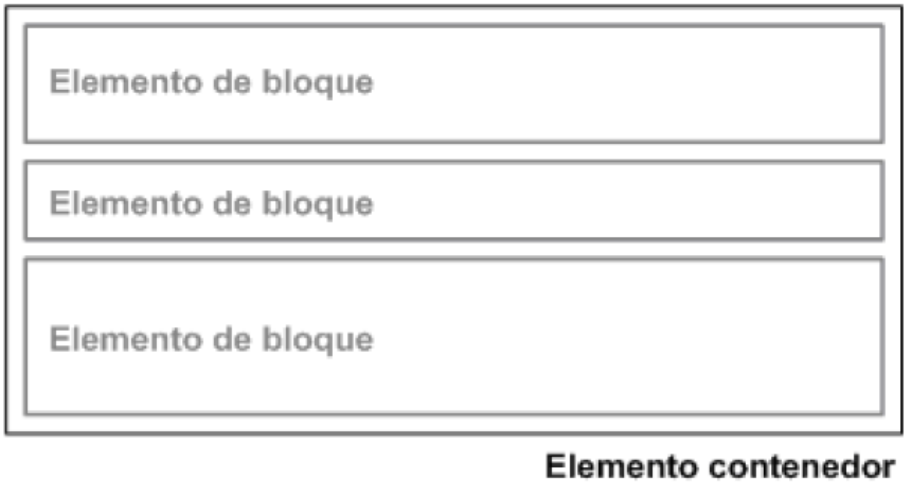
Propiedades de desplazamiento (2)

- En el caso del posicionamiento relativo, cada una de estas propiedades indica el desplazamiento del elemento desde la posición original de su borde superior/derecho/inferior/izquierdo. Si el posicionamiento es absoluto, las propiedades indican el desplazamiento del elemento respecto del borde superior/derecho/inferior/izquierdo de su primer elemento padre posicionado.
- En cualquiera de los dos casos, si el desplazamiento se indica en forma de porcentaje, se refiere al porcentaje sobre la anchura (propiedades right y left) o altura (propiedades top y bottom) del elemento.

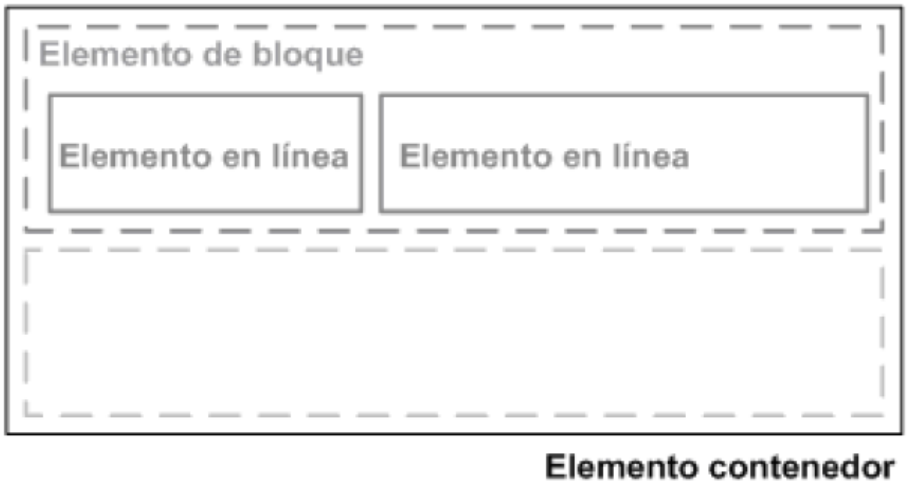
Posicionamiento normal (1)

- El posicionamiento normal o estático es la fórmula que, por defecto, aplican los navegadores para mostrar los elementos de una página.
- Un elemento posicionado de manera estática se posiciona de acuerdo al flujo normal de una página. En ese flujo normal, los elementos de bloque se muestran verticalmente unos debajo de los otros, y los elementos de línea horizontalmente unos detrás de otros.

Posicionamiento normal (2)



Posicionamiento normal de los elementos de bloque

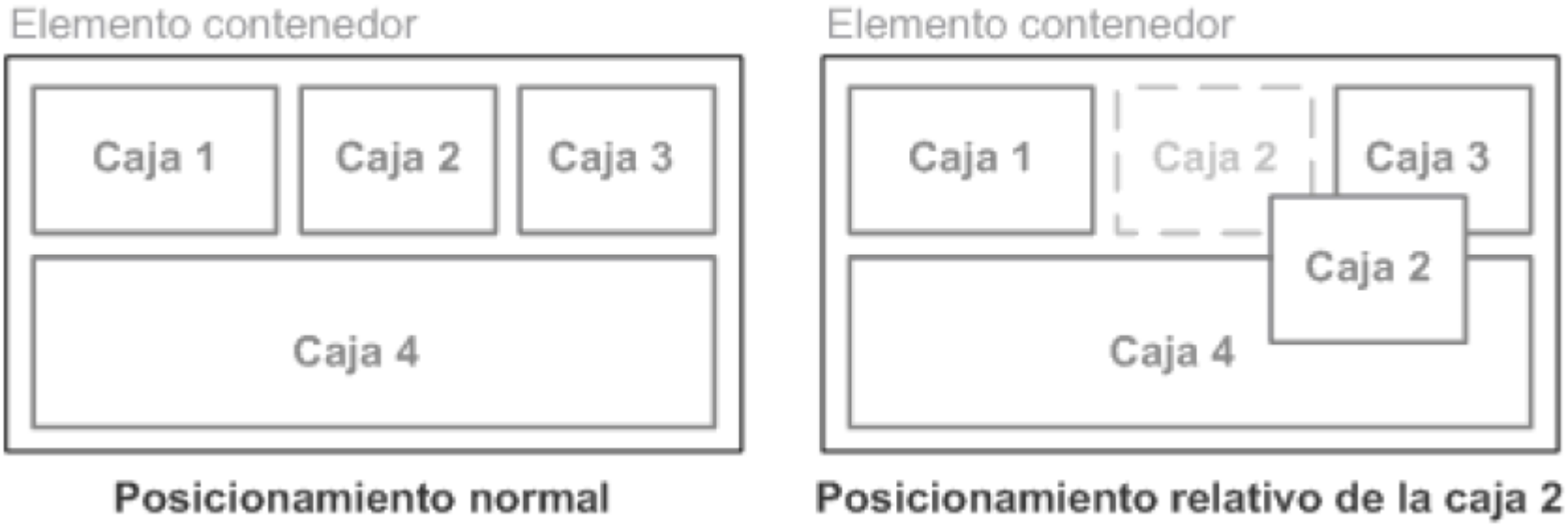


Posicionamiento normal de los elementos de línea

Posicionamiento relativo (1)

- El posicionamiento relativo parte de las mismas premisas que el posicionamiento estático, introduciendo la posibilidad de 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.

Posicionamiento relativo (2)



- En la imagen anterior, la caja 2 se ha desplazado lateralmente hacia la derecha y verticalmente de forma descendente. Como el resto de cajas de la página no modifican su posición, se producen solapamientos entre los contenidos de las cajas.

Posicionamiento relativo (3)

- 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.

```
img.desplazada {
        position: relative;
        top: 40px;
}
```

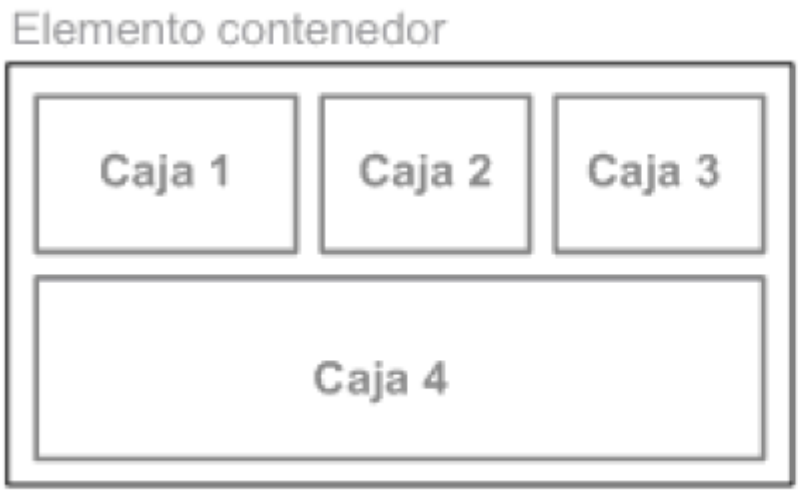
En este ejemplo se desplaza la imagen con clase "desplazada" 40 pixeles hacia abajo

Posicionamiento absoluto (1)

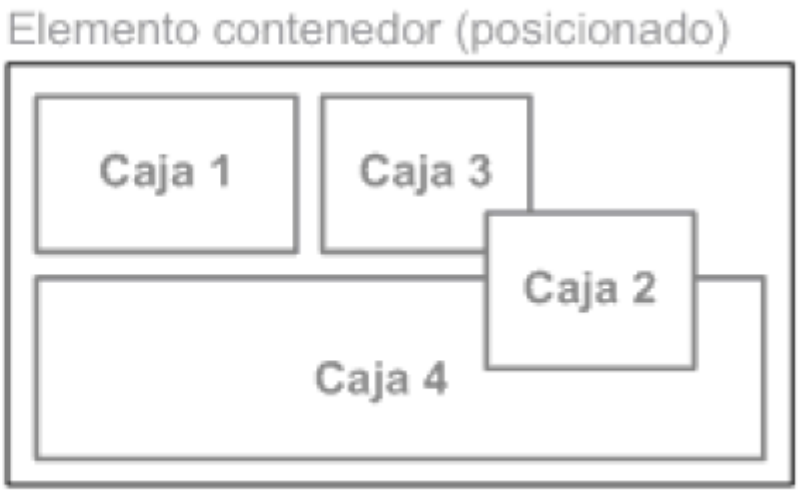
- La posición absoluta se utiliza para ubicar de forma precisa la ubicación de la caja o elemento. A diferencia de la posición relativa, donde la caja sale de su posición original sin afectar a las cajas adyacentes, el posicionamiento absoluta provoca el desplazamiento de las cajas adyacentes, que terminan ignorando a la caja absoluta y ocupando su lugar en la pantalla.

Posicionamiento absoluto (2)

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



Posicionamiento normal



Posicionamiento absoluto de 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.

Posicionamiento absoluto (3)

- En el estándar de CSS, esta característica de las cajas posicionadas de forma absoluta se explica como que la caja *sale por completo del flujo normal del documento*. De hecho, las cajas posicionadas de forma absoluta parece que están en un nivel diferente al resto de elementos de la página.
- Por otra parte, el desplazamiento de una caja posicionada de forma absoluta se indica mediante las propiedades `top`, `right`, `bottom` y `left`.

Posicionamiento absoluto (4)

- La clave del posicionamiento absoluto reside en el origen de sus coordenadas (superior-izquierda), basadas en las coordenadas de origen de su elemento contenedor. **Y para seleccionar este elemento contenedor, el navegador escoge únicamente aquél elemento contenedor que esté posicionado de cualquier forma diferente a `position: static`**. La esquina superior-izquierda de ese elemento será el origen de coordenadas.

Posicionamiento fijo

- 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.

```
div{
position: fixed;
top: 200px;
right: 50px;
weight: 400px;
}
```

Posicionamiento flotante (1)

- 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.

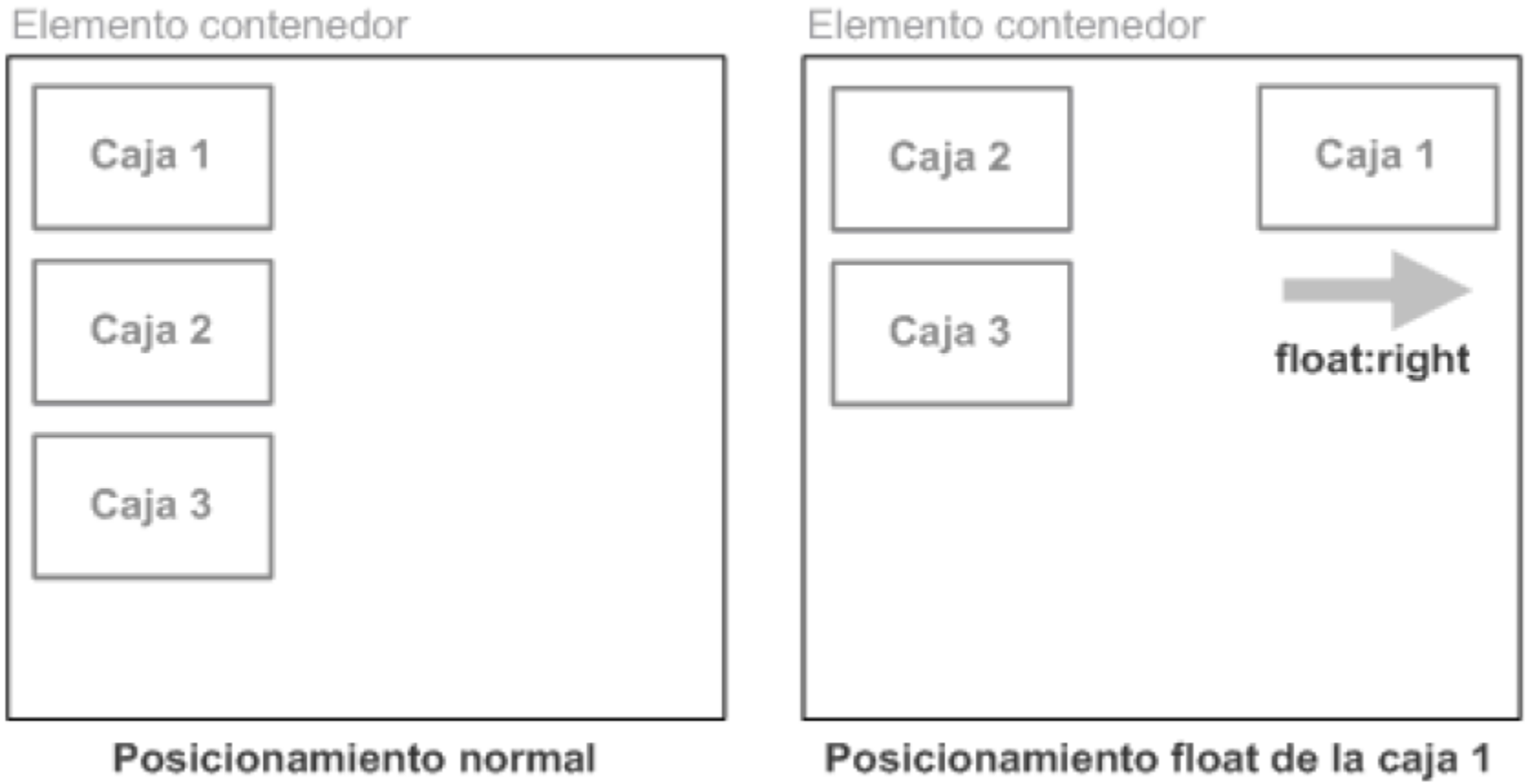
Posicionamiento flotante (2)

La propiedad CSS que permite posicionar de forma flotante una caja se denomina `float`:

`float`	Posicionamiento float
Valores	`left` | `right` | `none` | `inherit`
Se aplica a	Todos los elementos
Valor inicial	none
Descripción	Establece el tipo de posicionamiento flotante del elemento

Posicionamiento flotante (3)

- La siguiente imagen muestra el resultado de posicionar de forma flotante hacia la derecha la caja 1:



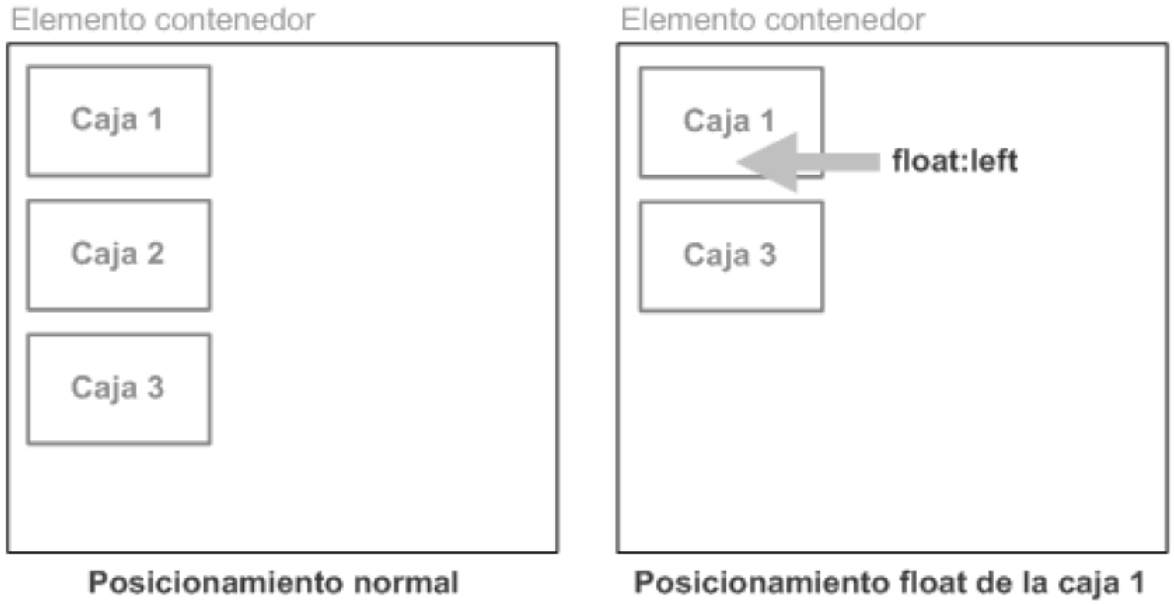
Posicionamiento flotante (4)

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.

Posicionamiento flotante (5)

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:



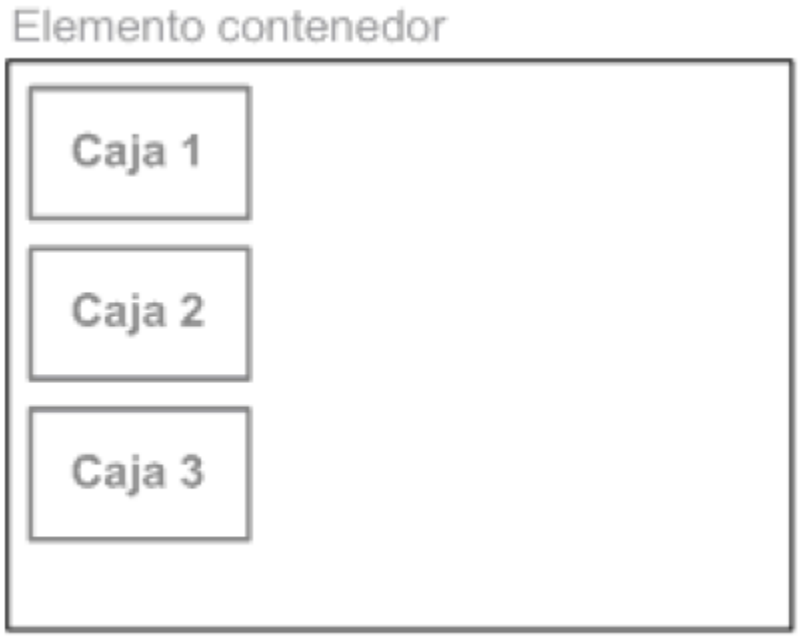
Posicionamiento normal

Posicionamiento float de la caja 1

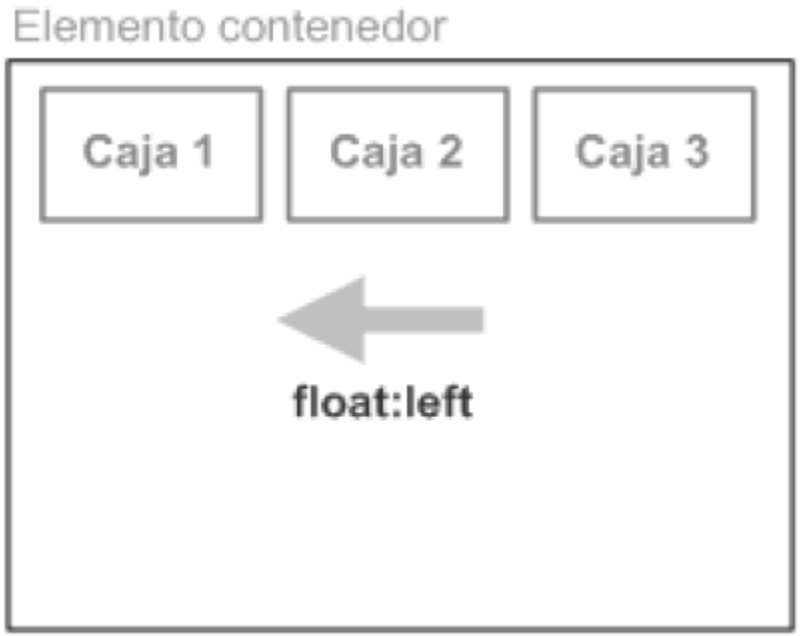
La caja 1 es de tipo flotante, por lo que *desaparece del flujo normal* de la página y el resto de cajas ocupan su lugar. El resultado es que la caja 2 ahora se muestra donde estaba la caja 1 y la caja 3 se muestra donde estaba la caja 2.

Posicionamiento flotante (6)

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:



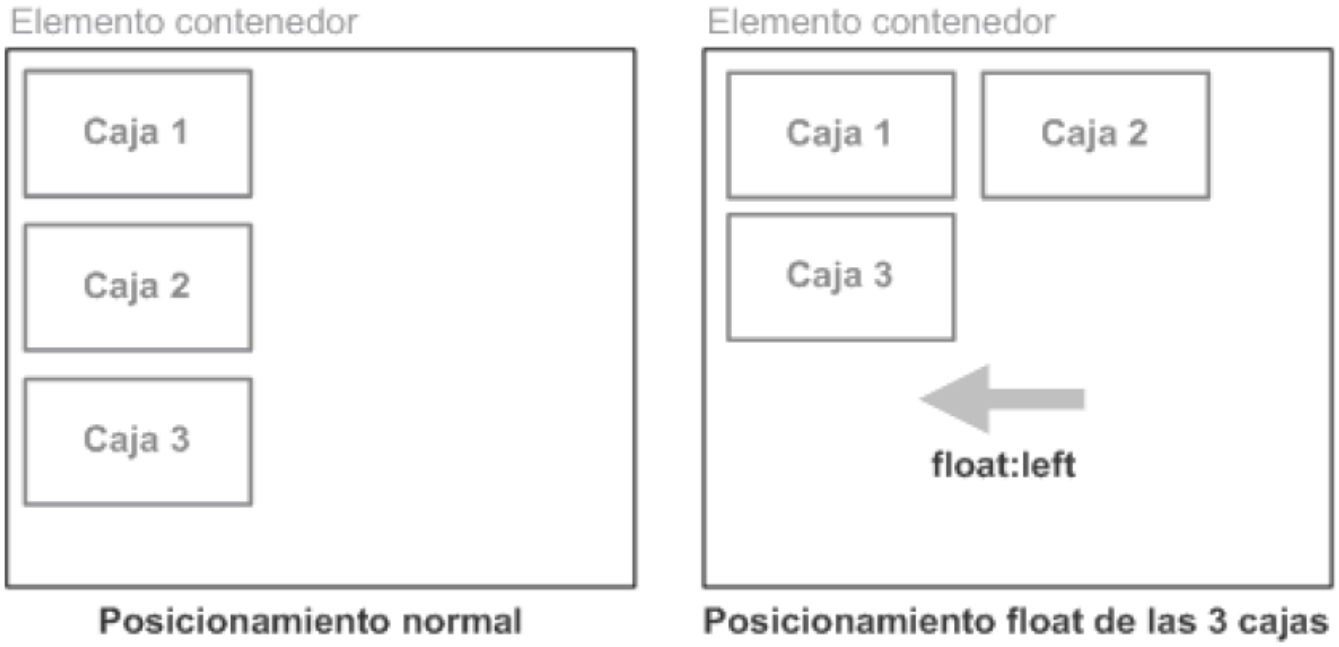
Posicionamiento normal



Posicionamiento float de las 3 cajas

Posicionamiento flotante (7)

- 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:



Posicionamiento flotante (8)

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

Posicionamiento normal

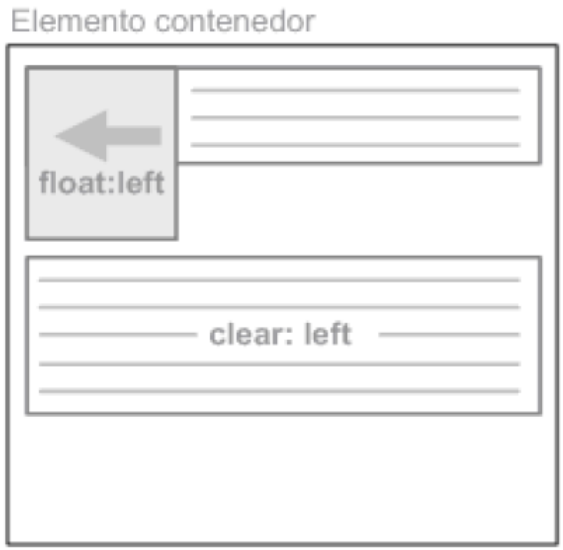
Posicionamiento float de la imagen

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

Posicionamiento flotante (9)

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:

Posicionamiento float de la imagen



Párrafo “limpio” con clear:left

Propiedad clear (1)

- La propiedad **clear** permite modificar el comportamiento por defecto del posicionamiento flotante para forzar a un elemento a mostrarse debajo de cualquier caja flotante. La regla CSS que se aplica al segundo párrafo del ejemplo anterior es la siguiente:

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

Propiedad clear (2)

La definición formal de la propiedad `clear` se muestra a continuación:

`clear`	Despejar los elementos flotantes adyacentes
Valores	`none` | `left` | `right` | `both` | `inherit`
Se aplica a	Todos los elementos de bloque
Valor inicial	none
Descripción	Indica el lado del elemento que no debe ser adyacente a ninguna caja flotante

Propiedad clear (3)

- 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.

Propiedad clear (4)

- 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.
- La propiedad clear es imprescindible cuando se crean las estructuras de las páginas web complejas.

Propiedad clear (5)

Considere el siguiente código CSS y HTML:

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

derecha {
float: right;
}
.izquierda {
float: left;
 }
```

Propiedad clear (6)

- Como los dos elementos creados dentro del elemento se han posicionado mediante float, los dos han salido del flujo normal del documento. Así, el elemento no tiene contenidos y por eso no llega a cubrir el texto de los dos elementos :

**Los elementos posicionados vacían de contenidos al **

Propiedad clear (7)

Existen dos forma de solucionar este problema, una más elegante que la otra.

- **Solución 1**

Esta solución consiste en añadir un elemento adicional invisible que *limpie* el float forzando a que el original cubra completamente los dos elementos .

```
<div id="paginacion">
    <span class="izquierda">&laquo; Anterior</span>
    <span class="derecha">Siguiente &raquo;</span>
    <div  style="clear:both;">
</div>
```

Propiedad clear (8)

El añadido "limpia" el float y fuerza la altura del original

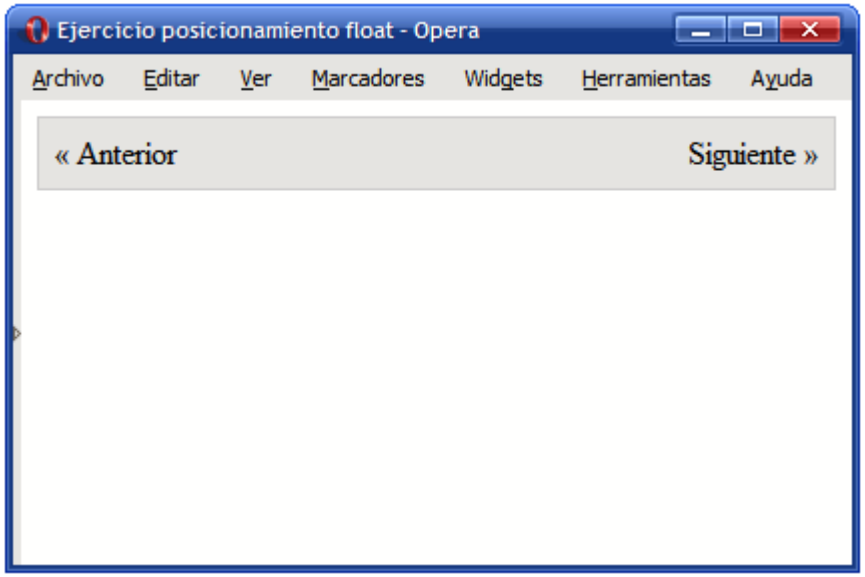
- Al añadir un con la propiedad clear: both, se tiene la seguridad de que el añadido se va a mostrar debajo de cualquier elemento posicionado con float y por tanto, se asegura que el original tenga la altura necesaria como para encerrar a todos sus contenidos posicionados con float.

Propiedad clear (9)

- **Solución 2**

Esta es la solución más elegante y consiste en utilizar la propiedad `overflow` sobre el elemento contenedor:

```
#paginacion {
	border: 1px solid #CCC;
	background-color: #E0E0E0;
	padding: .5em;
	overflow: hidden;
}
.derecha { float: right; }
.izquierda { float: left; }
```



VISUALIZACIÓN

Visualización

- Además de las propiedades que controlan el posicionamiento de los elementos, CSS define otras cuatro propiedades para controlar su visualización: `display`, `visibility`, `overflow` y `z-index`.
- Utilizando algunas de estas propiedades es posible ocultar y/o hacer invisibles las cajas de los elementos, por lo que son imprescindibles para realizar efectos avanzados.

Propiedad display (1)

- Es la primera de las propiedades de visualización en CSS y es una de las más útiles para diseñadores.

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

Propiedad display (2)

Los valores más utilizados son `inline`, `block` , none e `inline-block`.

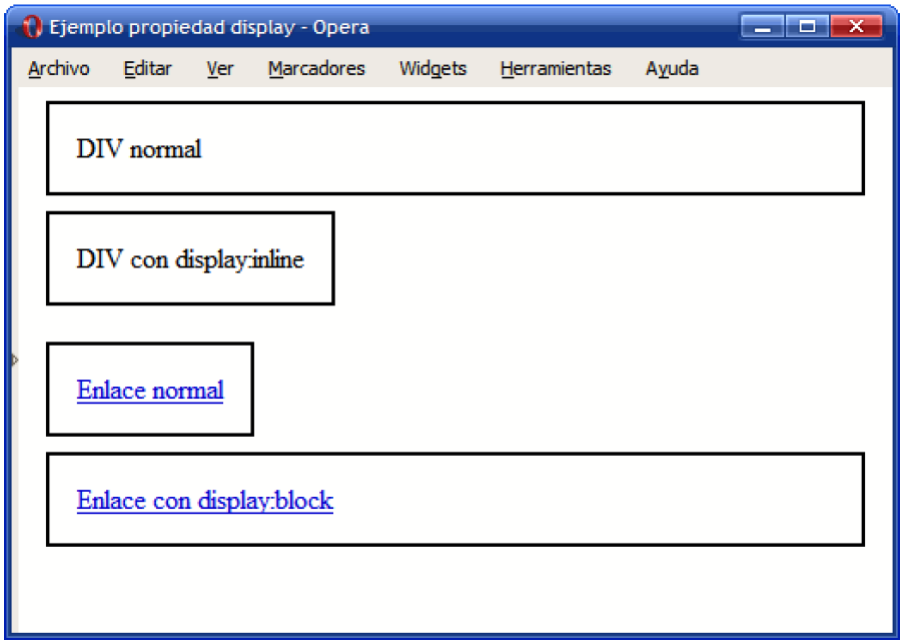
- El valor `block` permite mostrar un elemento como si fuera un elemento de bloque, independientemente del tipo de elemento que se trate.
- El valor `inline` permite visualizar un elemento en forma de elemento en línea, independientemente del tipo de elemento que se trate.
- El valor none permite ocultar un elemento y hacer que desaparezca de la página. El resto de elementos se visualizan como si no existiera el elemento oculto, es decir, pueden ocupar el espacio en el que se debería visualizar el elemento.

Propiedad display (3)

- El valor `inline-block` es muy interesante ya que permite obtener un comportamiento intermedio entre los elementos de línea y los de bloque. **Los elementos inline-block fluyen con el texto y demás elementos** como si fueran elementos en-línea **y además respetan el ancho, el alto y los márgenes verticales**. Son lo mejor de los dos mundos. Con este valor se pueden conseguir comportamientos similares que con el uso de la propiedad float pero sin el dolor de cabeza de tener que usar propiedades como clear u overflow.

Propiedad display (4)

El siguiente ejemplo muestra el uso de la propiedad display para mostrar un elemento de bloque como si fuera un elemento en línea y para mostrar un elemento en línea como si fuera un elemento de bloque:




```
<div>DIV normal</div>
<div style="display:inline">
DIV con display:inline</div>

<a href="#">Enlace normal</a>
<a href="#" style="display:block">
Enlace con display:block</a>
```

Propiedad display (5)

- Ahora se presenta un ejemplo de uso del valor inline-block:

```
.box2 {
    display: inline-block;
    width: 200px;
    height: 100px;
    margin: 1em;
}
```

Propiedad display (6)

- El ejemplo anterior produciría un resultado similar al obtenido con float: left, pero sin la necesidad de hacer arreglos usando propiedades como `clear` u `overflow`.

Propiedad visibility

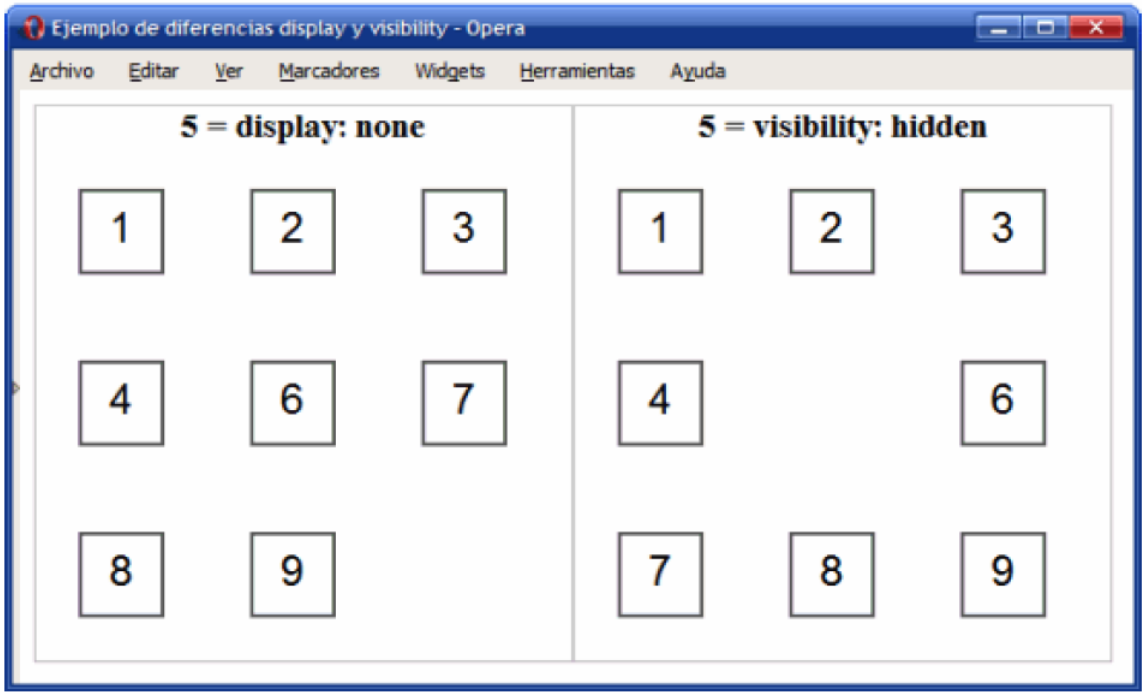
Mediante el uso de la propiedades `visibility` es posible ocultar un elemento manteniendo intacto su espacio (al contrario de lo que ocurriría con `display: none`).

Con esta propiedad es posible ocultar elementos del diseño sin afectar al resto de elementos.

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

Ocultando elementos: Display vrs Visibility (1)

- Las propiedades display y visibility controlan la visualización de los elementos. Las dos propiedades permiten ocultar un elemento de la página.
- La siguiente imagen muestra las diferencias entre la propiedad display y la propiedad visibility al ocultar una caja (la número 5):



Ocultando elementos: Display vrs Visibility (2)

- La propiedad `display` permite ocultar un elemento haciendo que desaparezca de la página. Como el elemento oculto no existe, el resto de elementos de la página se muestran como si no existiera y ocupan el hueco dejado por la caja oculta.
- Por otra parte, la propiedad `visibility` permite hacer invisible un elemento creando su caja pero no mostrando sus contenidos. En este caso, la posición del resto de elementos sí que se mantiene igual que si el elemento no fuera invisible.
- En general, cuando se oculta un elemento no es deseable que siga ocupando sitio en la página, por lo que la propiedad `display` se utiliza mucho más que la propiedad `visibility`.

Propiedad overflow (1)

- Mediante el uso de esta propiedad es posible controlar el comportamiento de los contenidos cuando éstos se salen de su elemento contenedor.
- En el diseño mediante el modelo de caja es habitual que el contenido se desborde fuera de su elemento contenedor. Esto ocurre cuando al elemento contenedor se le aplican unas dimensiones concretas mediante el uso de las propiedades `width` y `height`.

`overflow`	Parte sobrante de un elemento
Valores	`visible` | `hidden` | `scroll` | `auto` | `inherit`
Se aplica a	Elementos de bloque y celdas de tablas
Valor inicial	visible
Descripción	Permite controlar los contenidos sobrantes de un elemento

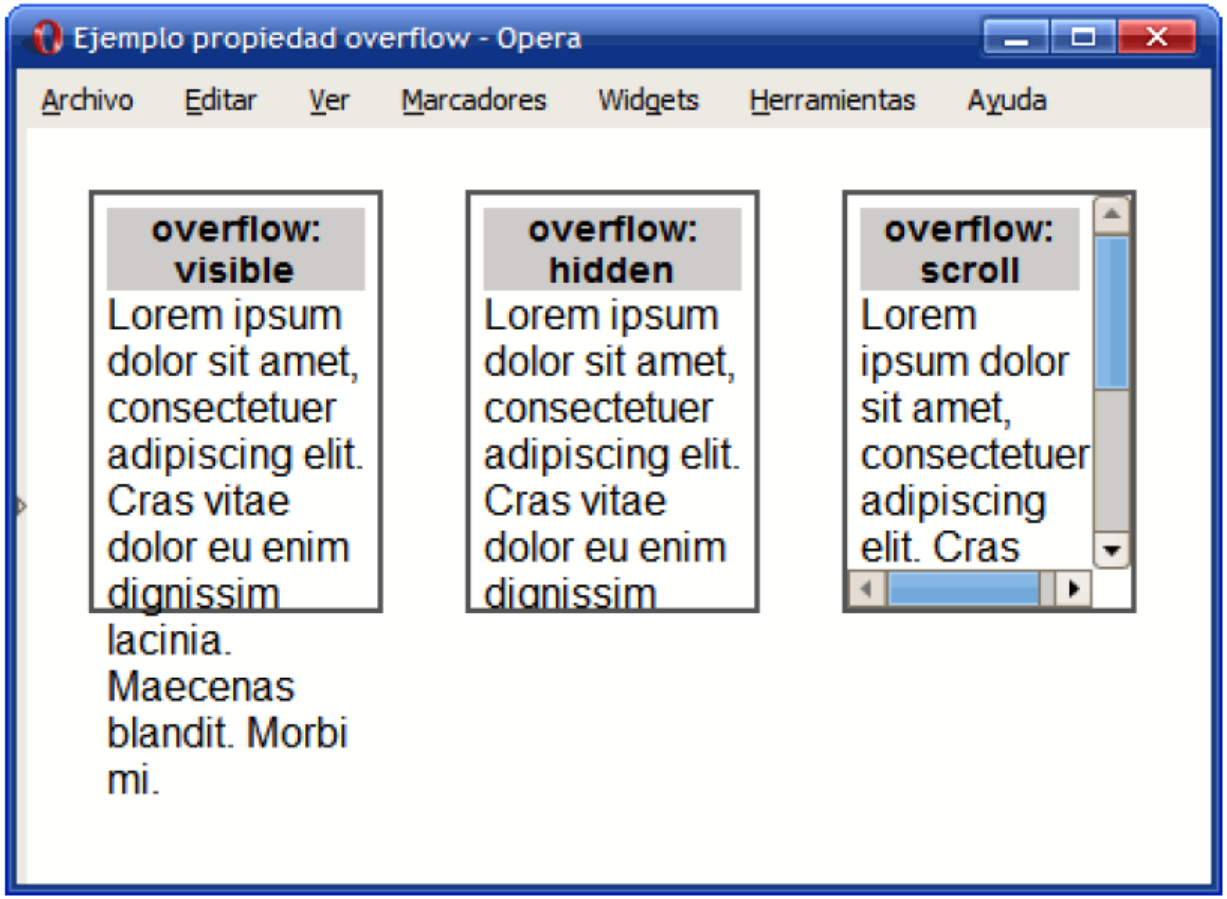
Propiedad overflow (2)

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.

Propiedad overflow (3)

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



Propiedad z-index (1)

- 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 controlar las cajas que se muestran delante o detrás de otras cajas cuando se producen solapamientos.
- La posición tridimensional 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.

Propiedad z-index (2)

- A continuación se muestra la definición formal de la propiedad z-index:

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

- 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.

Propiedad z-index (3)

- Considérese el siguiente ejemplo:

```html
<div>
  <span class="red">Rojo</span>
</div>
<div>
  <span class="green">Verde</span>
</div>
<div>
  <span class="blue">Azul</span>
</div>
```

```css
.red, .green, .blue {
  position: absolute;
  width: 100px;
  color: white;
  line-height: 100px;
  text-align: center;
}

.red {
  z-index: 3;
  top: 20px;
  left: 20px;
  background: red;
}
```

```css
.green {
  z-index: 2;
  top: 80px;
  left: 80px;
  background: green;
}

.blue {
  z-index: 1;
  top: 140px;
  left: 140px;
  background: blue;
}
```

Propiedad z-index (4)

- El resultado generado sería: