Capítulo 5. Posicionamiento y visualización

Cuando los navegadores descargan el contenido HTML y CSS de las páginas web, aplican un procesamiento muy complejo antes de mostrar las páginas en la pantalla del usuario.

Para cumplir con el modelo de cajas presentado en el capítulo anterior, los navegadores crean una caja para representar a cada elemento de la página HTML. Los ***factores que se tienen en cuenta para generar cada caja son:***

- Las propiedades **width** y **height** de la caja (si están establecidas).
- El *tipo de cada elemento HTML* (elemento de ***bloque*** o elemento ***en línea***).
- ***Posicionamiento de la caja*** (**normal, relativo, absoluto, fijo o flotante**).
- Las *relaciones entre elementos* (dónde se encuentra cada elemento, elementos descendientes, etc.)
- Otro tipo de información, como por ejemplo el tamaño de las imágenes y el tamaño de la ventana del navegador.

En este capítulo se muestran los cinco tipos de posicionamientos definidos para las cajas y se presentan otras propiedades que afectan a la forma en la que se visualizan las cajas.

5.1. Tipos de elementos

El estándar HTML clasifica a todos sus elementos en dos grandes grupos: elementos en línea y elementos de bloque:

- Los ***elementos de bloque*** (*"block elements"* en inglés) siempre empiezan en una nueva línea y ocupan todo el espacio disponible hasta el final de la línea.
- Los ***elementos en línea*** (*"inline elements"* en inglés) no empiezan necesariamente en nueva línea y sólo ocupan el espacio necesario para mostrar sus contenidos.

Ejemplo de las cajas que crea el navegador para representar diferentes elementos HTML:

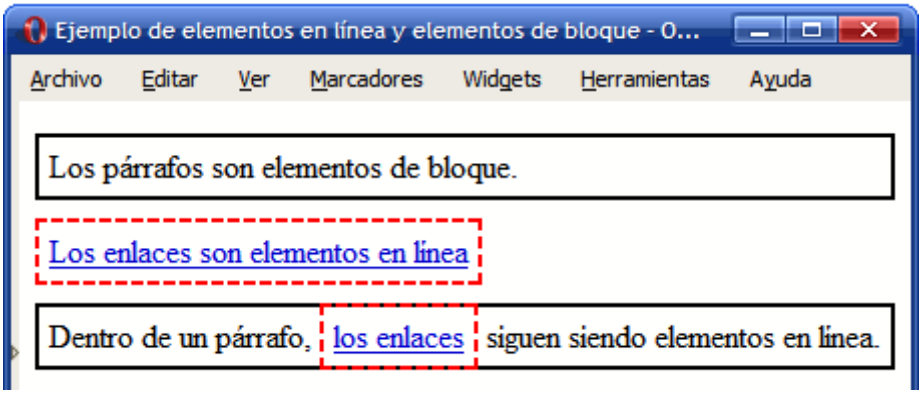

Figura 5.1. Cajas creadas por los elementos de línea y los elementos de bloque

Por sus características, los elementos de bloque no pueden insertarse dentro de elementos en línea y tan sólo pueden aparecer dentro de otros elementos de bloque. En cambio, un elemento en línea puede aparecer tanto dentro de un elemento de bloque como dentro de otro elemento en línea.

5.2. Posicionamiento

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***.

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 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.

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	***Especifica el posicionamiento del elemento***

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.
- **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.

Desplazamiento

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***

Para el **posicionamiento relativo**: estas propiedades indican el *desplazamiento del elemento desde la posición original* de su borde superior/derecho/inferior/izquierdo.

Para el **posicionamiento 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.

5.3. Posicionamiento normal

El posicionamiento **normal o estático** es el modelo que ***utilizan por defecto los navegadores*** para mostrar los elementos de las páginas. En este modelo, ninguna caja se desplaza respecto de su posición original, por lo que *sólo se tiene en cuenta si el elemento es de bloque o en línea*.

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

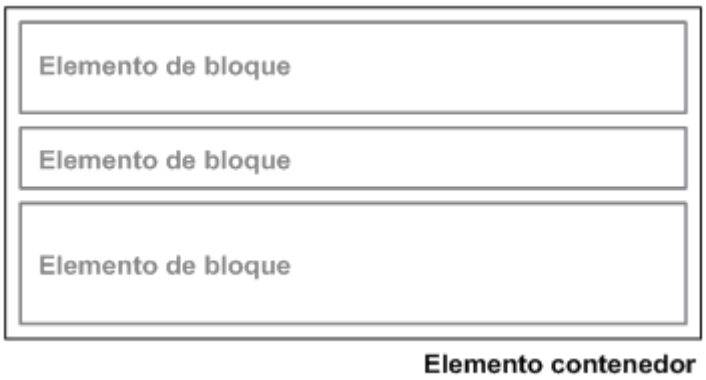

Figura 5.2. Posicionamiento normal de los elementos de bloque

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

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

Los elementos en línea forman los *"**contextos de formato en línea**"*. En este tipo de contextos, las cajas se muestran una detrás de otra de forma horizontal comenzando desde la posición más a la izquierda de su elemento contenedor. La distancia entre las cajas se controla mediante los márgenes laterales.

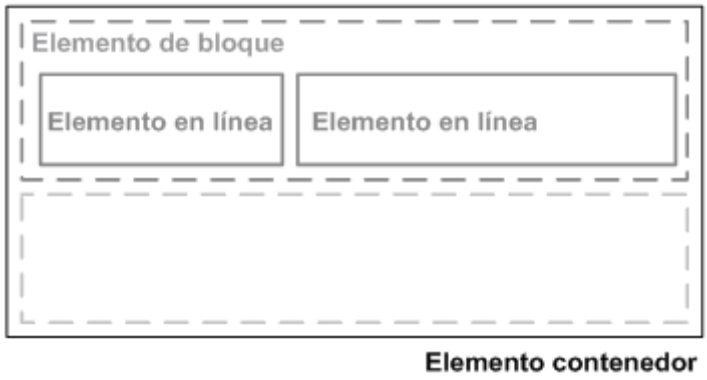

Figura 5.3. Posicionamiento normal de los elementos en línea

Si las cajas en línea ocupan más espacio del disponible en su propia línea, el resto de cajas se muestran en las líneas inferiores. Si las cajas en línea ocupan un espacio menor que su propia línea, ***se puede controlar la distribución de las cajas mediante la propiedad text-align para centrarlas, alinearlas a la derecha o justificarlas***.

5.4. Posicionamiento relativo

El estándar CSS considera que el posicionamiento relativo es un caso particular del posicionamiento normal, aunque en realidad presenta muchas diferencias.

El **posicionamiento relativo** permite ***desplazar*** *una caja respecto de su posición original establecida mediante el posicionamiento normal.* El **desplazamiento** de la caja se controla con las ***propiedades top, right, bottom y left***.

El desplazamiento de una caja **no afecta** al **resto de cajas adyacentes**, que se muestran en la misma posición que si la caja desplazada no se hubiera movido de su posición original.

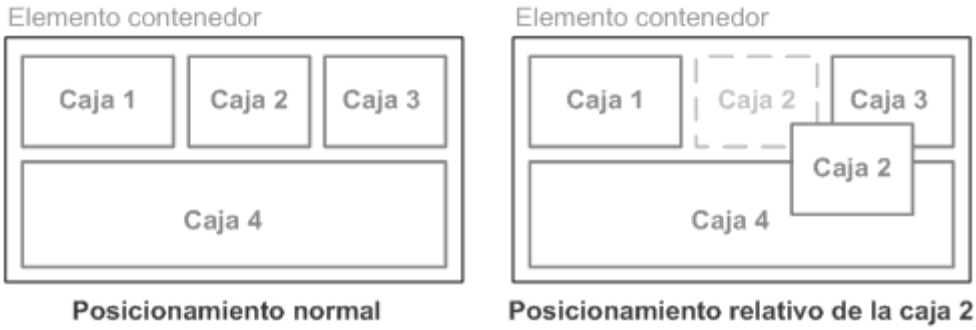

Figura 5.4. Ejemplo de posicionamiento relativo de un elemento

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.

Propiedad	Desplazamiento "positivo"	Desplazamiento "negativo"
left	Desplaza hacia la derecha	Desplaza hacia la izquierda
right	Desplaza hacia la izquierda	Desplaza hacia la derecha
top	Desplazamiento descendente	Desplazamiento ascendente
bottom	Desplazamiento ascendente	Desplazamiento descendente

Nota: si se han especificado valores tanto a **left** como a **right**, uno de los dos valores se tiene que ignorar porque son mutuamente excluyentes. En este caso CSS tiene en cuenta la propiedad **direction** para determinar el valor de la propiedad que tendrá en cuenta. La propiedad **direction** permite establecer la dirección del texto de un contenido:

- Si el valor de direction es **ltr**, el *texto se muestra de izquierda a derecha*, que es el método de escritura habitual en la mayoría de países → en este caso sólo se tendría en cuenta el valor de la propiedad **left.**
- Si el valor de direction es **rtl**, el método de escritura es de derecha a izquierda → en este caso sólo se tendría en cuenta el valor de la propiedad **right.**

El siguiente ejemplo muestra tres imágenes posicionadas de forma normal:

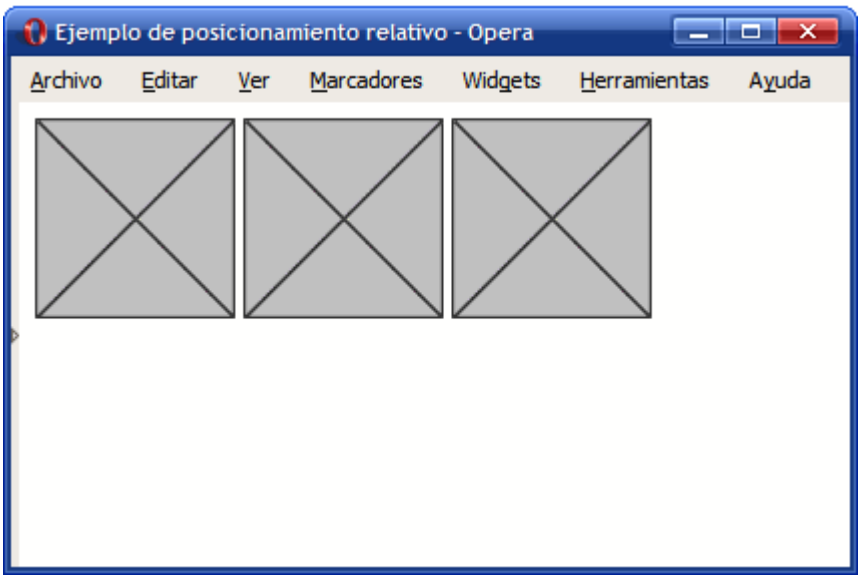

Figura 5.5. Elementos posicionados de forma normal

Aplicando el posicionamiento relativo, se podría desplazar la primera imagen de forma descendente:

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

<img class="desplazada" src="imagenes/imagen.png" alt="Imagen genérica" />
<img src="imagenes/imagen.png" alt="Imagen genérica" />
<img src="imagenes/imagen.png" alt="Imagen genérica" />
```

El aspecto que muestran ahora las imágenes es el siguiente:

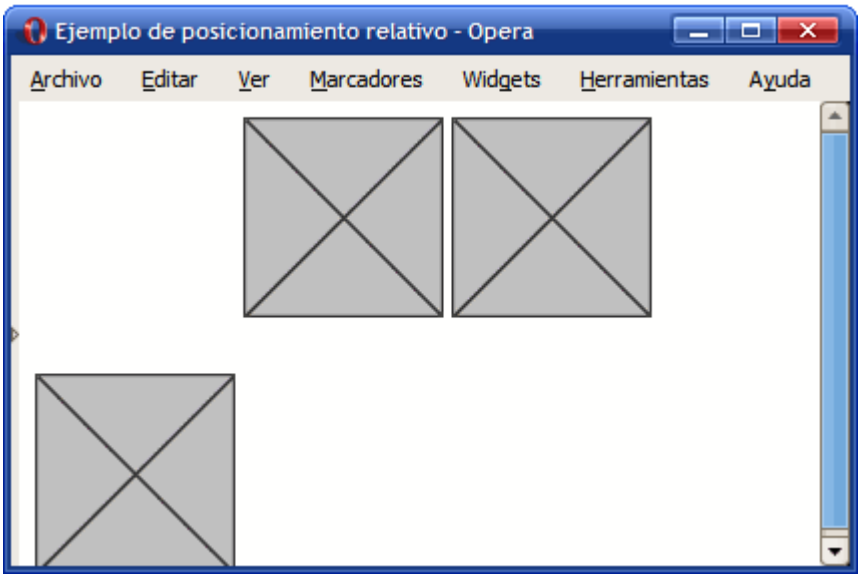

Figura 5.6. Elemento posicionado de forma relativa

El resto de imágenes no varían su posición y por tanto no ocupan el hueco dejado por la primera imagen, ya que el posicionamiento relativo no influye en el resto de elementos de la página. El único problema de posicionar elementos de forma relativa es que ***se pueden producir solapamientos*** con otros elementos de la página.

5.5. Posicionamiento absoluto

El posicionamiento absoluto se emplea ***para establecer de forma precisa la posición en la que se muestra la caja de un elemento***. La nueva posición de la caja se indica mediante las **propiedades top, right, bottom y left**. La interpretación de los valores de estas propiedades es mucho más compleja que en el posicionamiento relativo, ya que en este caso dependen del posicionamiento del elemento contenedor.

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

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

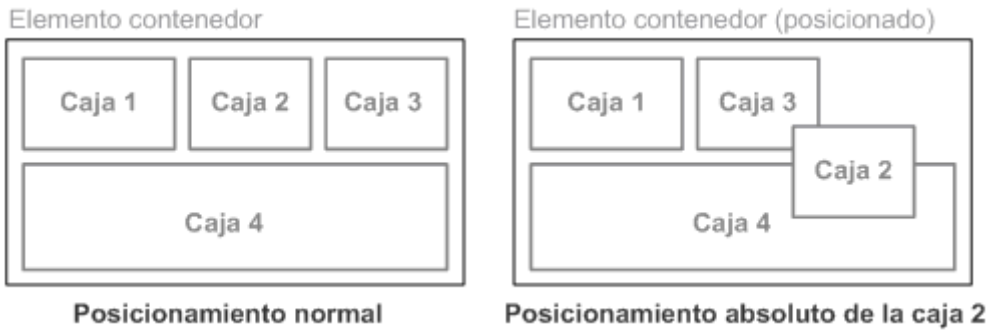

Figura 5.7. Ejemplo de posicionamiento absoluto de un elemento

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.

A diferencia de posicionamiento relativo, ***en este caso la referencia de los valores de las propiedades top, right, bottom y left es el origen de coordenadas de su primer elemento contenedor posicionado***.

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

1. Se buscan todos los elementos contenedores de la caja hasta llegar al elemento <body> de la página.
2. Se recorren todos los elementos contenedores empezando por el más cercano a la caja y llegando hasta el <body>
3. *De todos ellos, el navegador se queda con el primer elemento contenedor que esté posicionado de cualquier forma diferente a position: static*
4. La esquina superior izquierda de ese elemento contenedor posicionado es el origen de coordenadas.
5. Una vez obtenido el origen de coordenadas, se interpretan los valores de las propiedades top, right, bottom y left respecto a ese origen y se desplaza la caja hasta su nueva posición.

En los siguientes ejemplos, se va a utilizar la página HTML que muestra la siguiente imagen:

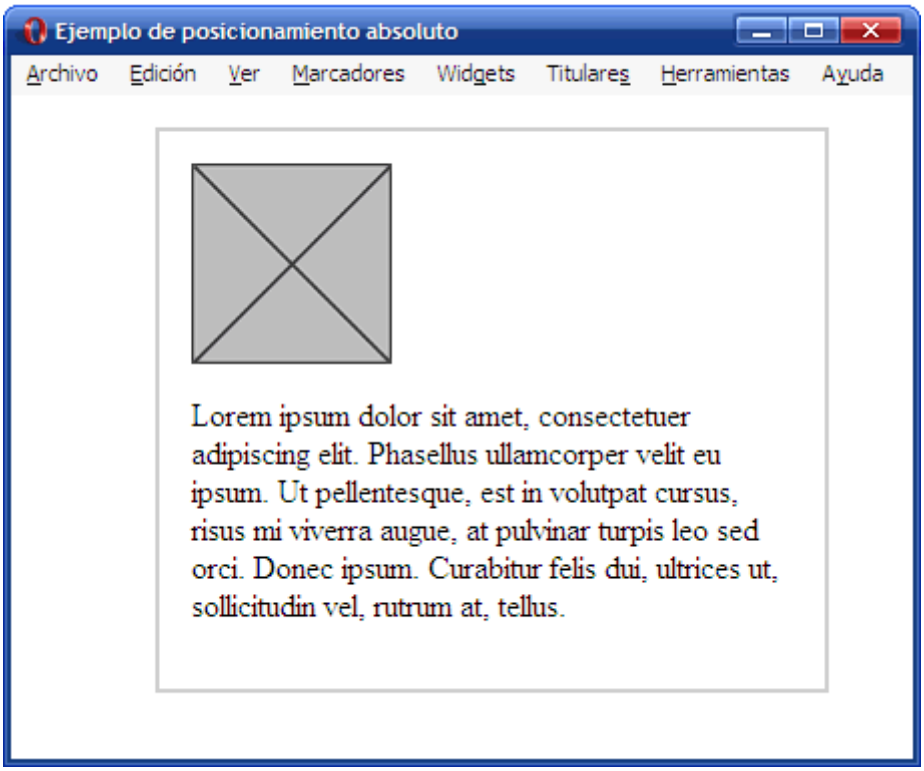

Figura 5.8. Situación original antes de modificar el posicionamiento

A continuación, se muestra el **código HTML y CSS de la página original**:

```
div {
 border: 2px solid #CCC;
 padding: 1em;
 margin: 1em 0 1em 4em;
 width: 300px;
}

<div>
        <img src="imagenes/imagen.png" alt="Imagen genérica" />
        <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Phasellus
        ullamcorper velit eu ipsum. Ut pellentesque, est in volutpat cursus, risus
        mi viverra augue, at pulvinar turpis leo sed orci. Donec ipsum. Curabitur
        felis dui, ultrices ut, sollicitudin vel, rutrum at, tellus.</p>
</div>
```

Posicionar de forma absoluta la imagen:

Mediante la propiedad ***position*** y se indica su nueva posición con ***top*** y ***left***:

```
div img {
 position: absolute;
 top:  50px;
 left: 50px;
}
```

El resultado visual se muestra en la siguiente imagen:

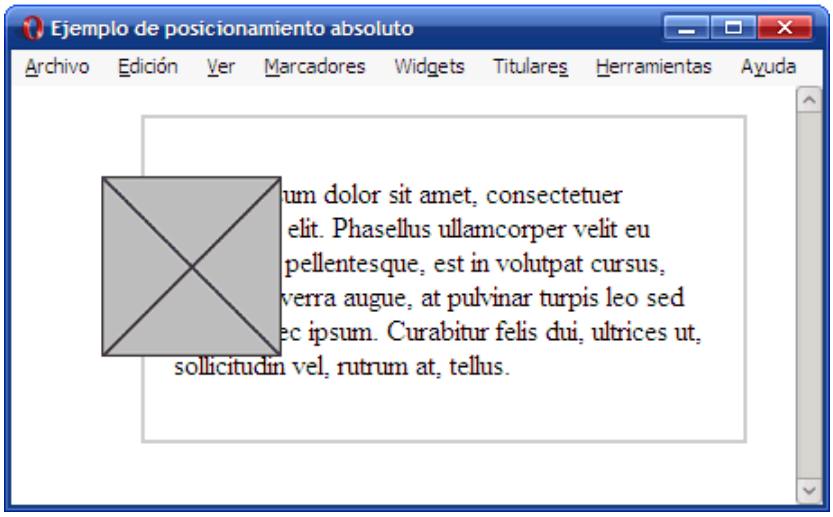

Figura 5.9. Imagen posicionada de forma absoluta

La **imagen posicionada de forma absoluta** no toma como origen de coordenadas la esquina superior izquierda de su elemento contenedor , sino que su ***referencia es la esquina superior izquierda de la página:***

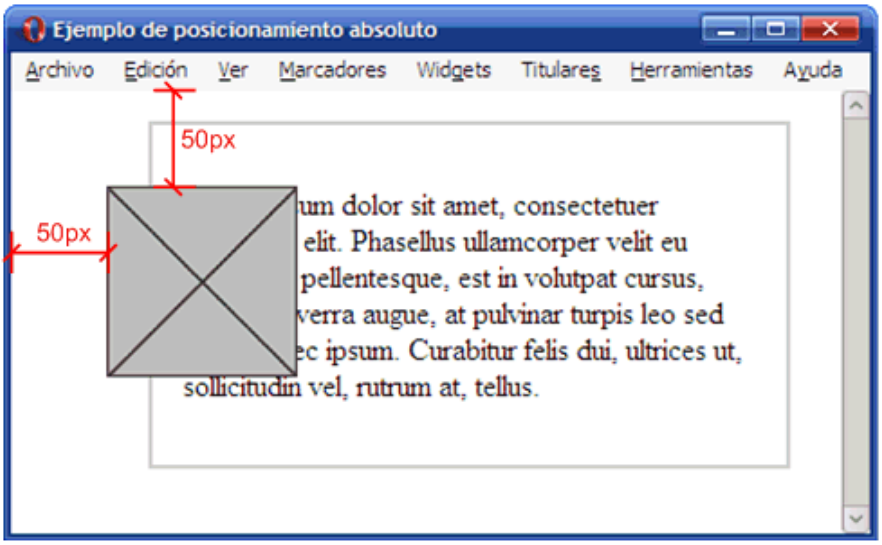

Figura 5.10. La referencia del posicionamiento absoluto es la página entera

Para **posicionar** la imagen **de forma absoluta**, el navegador realiza los siguientes pasos:

1. Obtiene la lista de elementos contenedores de la imagen: y <body>.
2. Recorre la lista de elementos desde el más cercano a la imagen (el) hasta terminar en el <body> buscando el primer elemento contenedor que esté posicionado.
3. El primer elemento contenedor es el , pero su posicionamiento es el normal o estático, ya que ni siquiera tiene establecida la propiedad position.
4. Como el siguiente elemento contenedor es el <body>, el navegador establece directamente el origen de coordenadas en la esquina superior izquierda de la página.
5. A partir de ese origen de coordenadas, la caja se desplaza 50px hacia la derecha (left: 50px) y otros 50px de forma descendente (top: 50px).

Además, el párrafo que se mostraba debajo de la imagen sube y ocupa el lugar dejado por la imagen. El resultado es que el elemento ahora sólo contiene el párrafo y la imagen se muestra en un nivel superior y cubre parcialmente los contenidos del párrafo.

**Si se posiciona de Forma Relativa el elemento **

La única propiedad añadida al es **position: relative** por lo que el *elemento contenedor se posiciona pero no se desplaza respecto de su posición original*:

```
div {
 border: 2px solid #CCC;
 padding: 1em;
 margin: 1em 0 1em 4em;
 width: 300px;
 position: relative;
}

div img {
 position: absolute;
 top: 50px;
 left: 50px;
}
```

La siguiente imagen muestra el resultado obtenido:

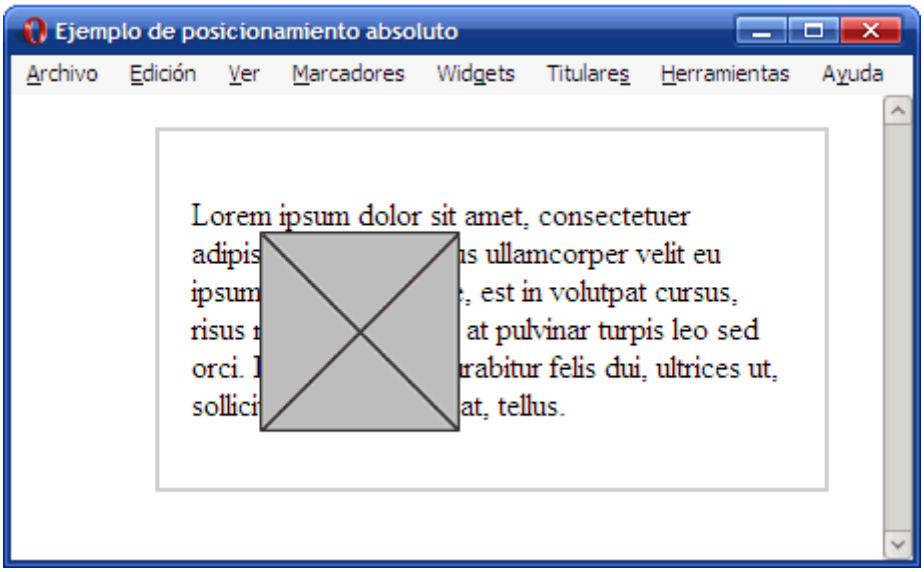

Figura 5.11. Imagen posicionada de forma absoluta

En este caso, *el origen de coordenadas* para determinar la nueva posición de la imagen *corresponde a la esquina superior izquierda del elemento* *:*

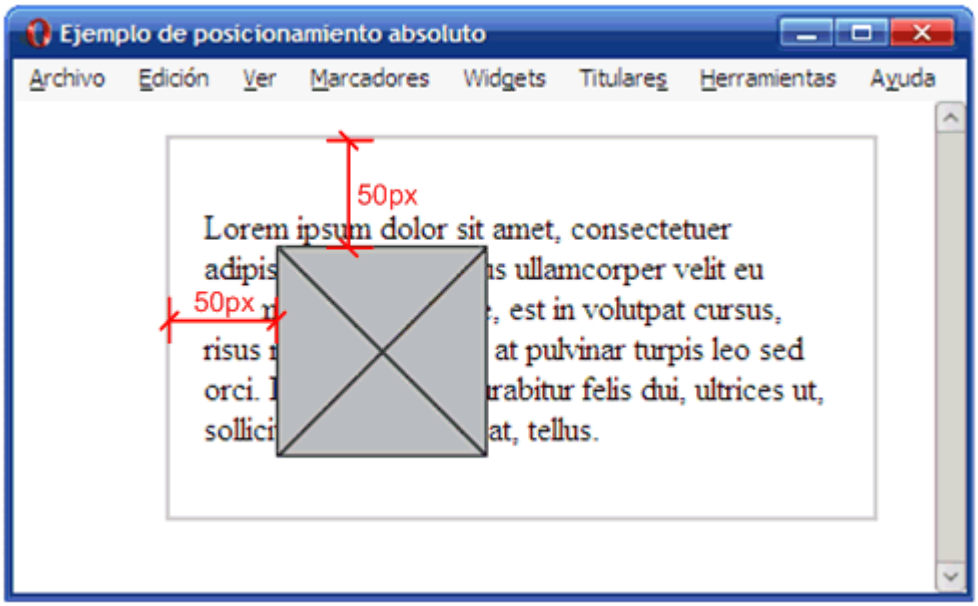

Figura 5.12. La referencia del posicionamiento absoluto es el elemento contenedor de la imagen

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

5.6. Posicionamiento fijo

Cuando **una caja se posiciona de forma fija**, ***el origen de coordenadas*** para interpretar su desplazamiento ***es el origen de coordenadas del navegador (esquina superior izquierda).***

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

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

5.7. Posicionamiento flotante

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, como se verá más adelante.

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 1:

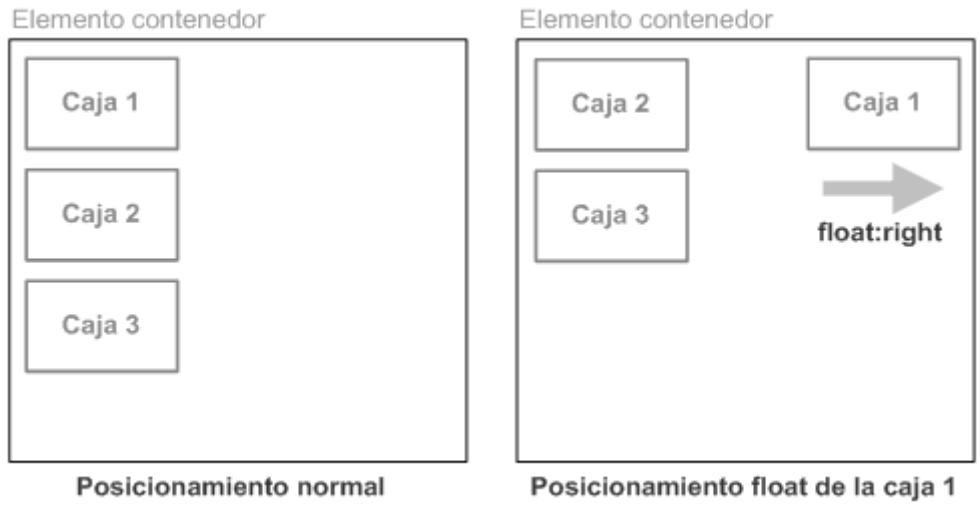

Figura 5.13. Ejemplo de posicionamiento float de una caja

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:

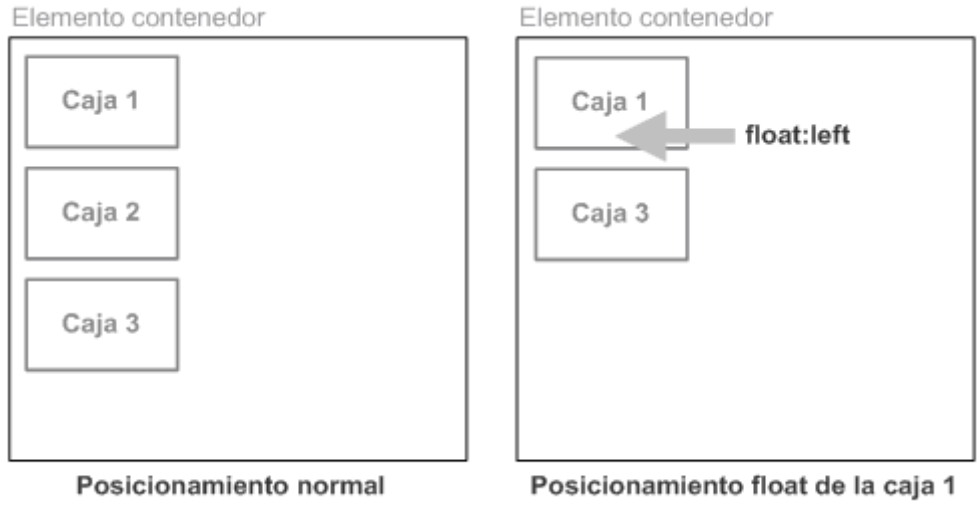

Figura 5.14. Ejemplo de posicionamiento float de una caja

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.

Posicionar de forma flotante las tres cajas hacia la izquierda:

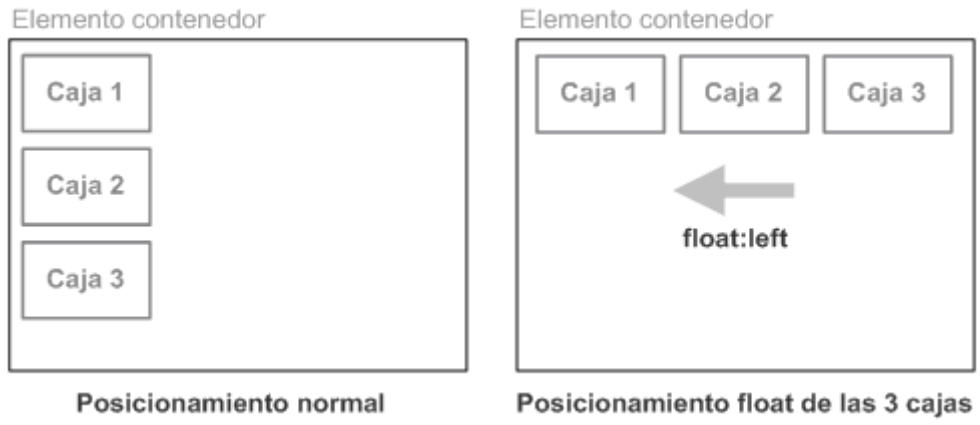

Figura 5.15. Ejemplo de posicionamiento float de varias cajas

En el ejemplo anterior, l***as 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:

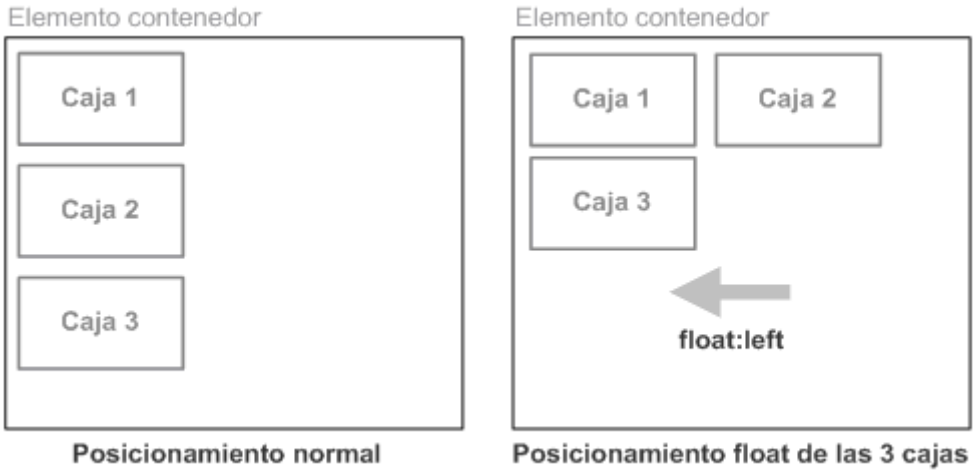

Figura 5.16. Ejemplo de posicionamiento float cuando no existe sitio suficiente

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.

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***

- 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:

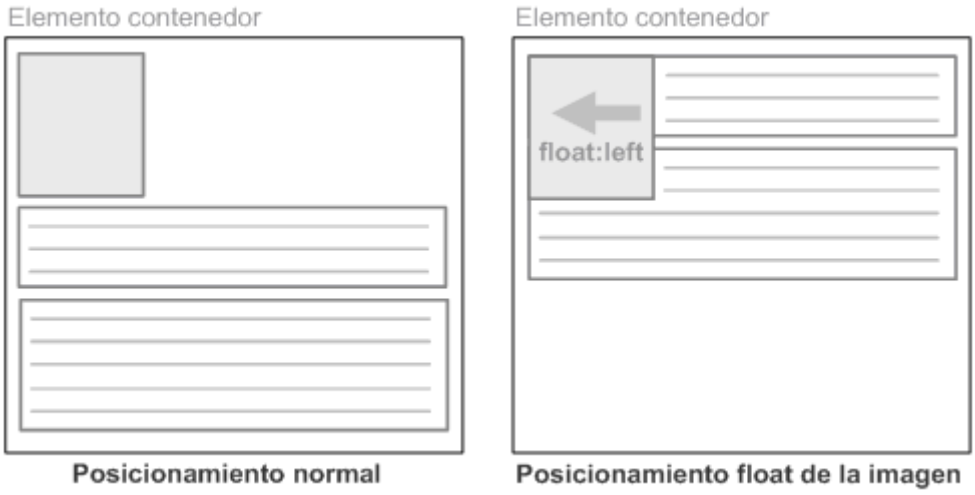

Figura 5.17. Elementos que fluyen alrededor de un elemento posicionado mediante float

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:

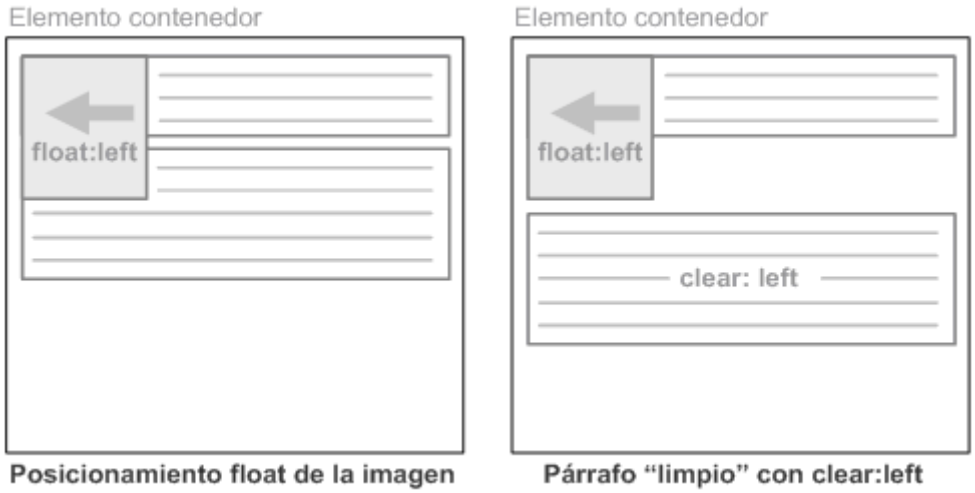

Figura 5.18. Forzando a que un elemento no fluya alrededor de otro elemento posicionado mediante float

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>
```

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***

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.

Como se verá más adelante, ***la propiedad clear es imprescindible cuando se crean las estructuras de las páginas web complejas***.

Ejemplo: Determinar las reglas CSS necesarias para obtener el siguiente resultado.

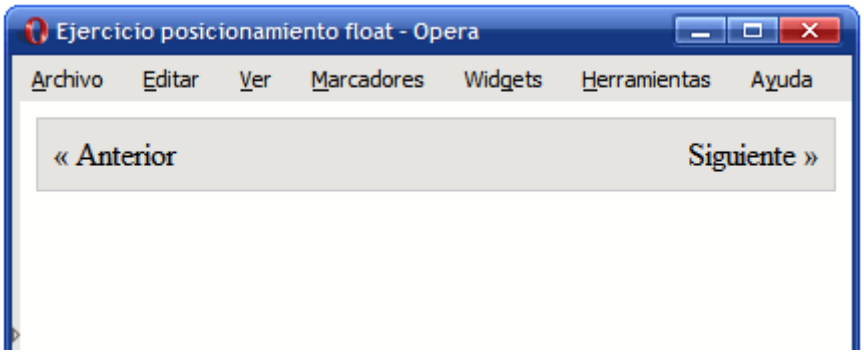

Figura 5.19. Ejemplo elementos posicionados mediante float

Se podría definir un elemento contenedor con borde y fondo, que contenga dos elementos . Un elemento span ("Anterior") posicionado mediante float a la izquierda y el otro ("Siguiente") posicionado a la izquierda (también mediante float).

Si se realiza de esta forma se obtendrá como resultado:

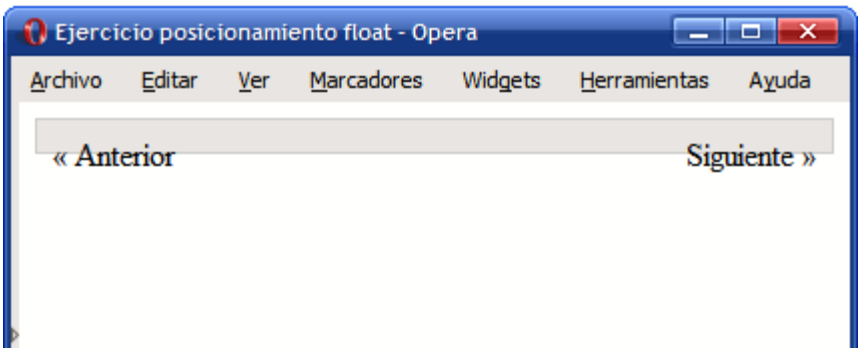

Figura 5.20. Visualización incorrecta de dos elementos posicionados mediante float

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 :

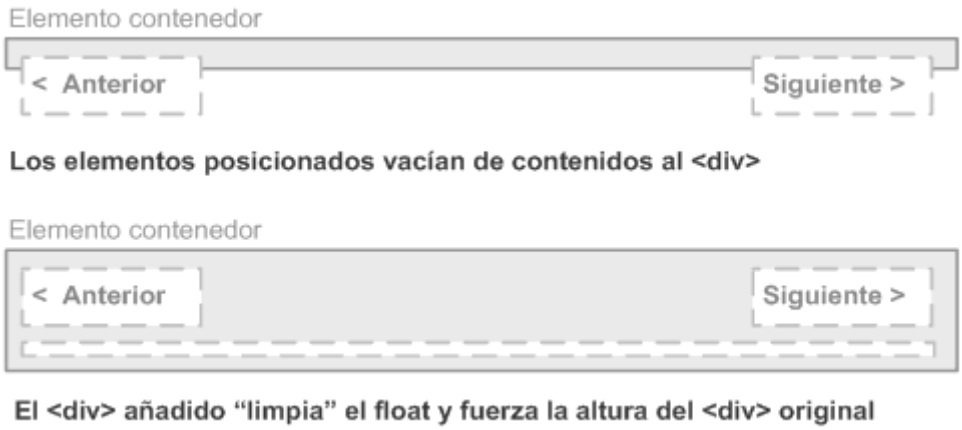

Figura 5.21. Esquema del problema y solución de la visualización incorrecta de dos elementos posicionados mediante float

La **solución** consiste en añadir un elemento adicional invisible que *limpie* el float forzando a que el original cubra completamente los dos elementos . El código HTML y CSS final se muestra a continuación:

```
<!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=iso-8859-1" />
<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>
```

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.

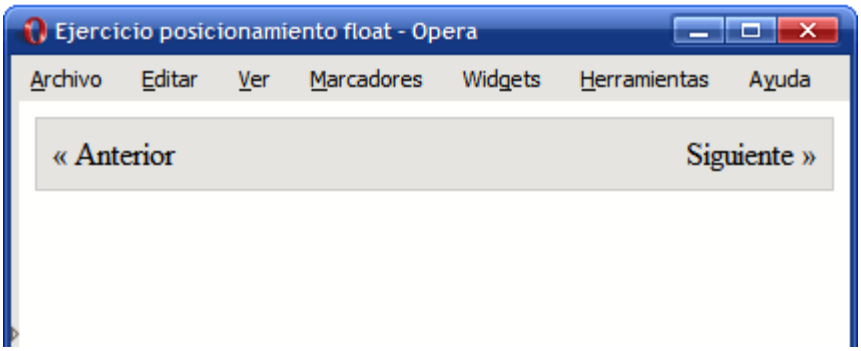

Figura 5.22. Visualización correcta de dos elementos posicionados mediante float

Además de un elemento invisible, también se puede utilizar un <p> invisible o un <hr/> invisible.

5.8. 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 y animaciones.

5.8.1. Propiedades display y visibility

Las propiedades **display** y **visibility** controlan la ***visualización de los elementos***. Las dos propiedades permiten **ocultar cualquier elemento de la página**. Habitualmente se utilizan junto con **JavaScript** para crear efectos dinámicos como mostrar y ocultar determinados textos o imágenes cuando el usuario pincha sobre ellos.

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

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

La siguiente imagen muestra la diferencia entre *ocultar* la caja número 5 mediante la **propiedad display** o hacerla *invisible* mediante la **propiedad visibility**:

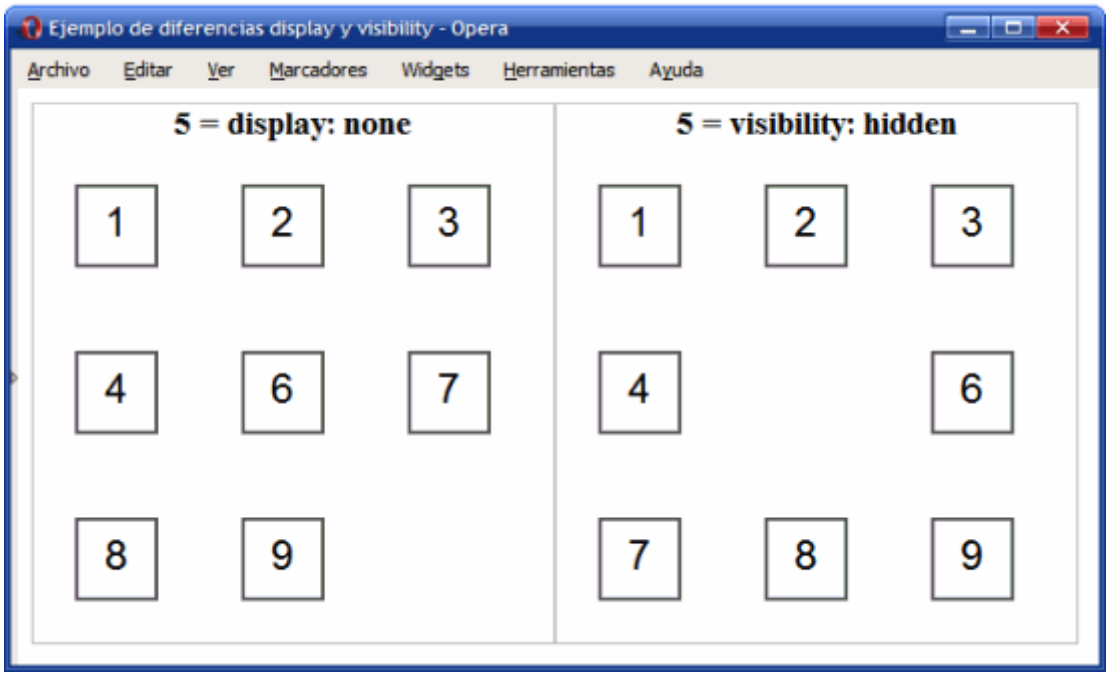

Figura 5.22. Diferencias visuales entre las propiedades display y visibility

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.

A continuación se muestra la definición completa de la **propiedad display**:

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***

Las posibilidades de la propiedad display son mucho más avanzadas que simplemente ocultar elementos. ***En realidad, la propiedad display modifica la forma en la que se visualiza un elemento***.

Los valores más utilizados son inline, block y none:

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

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:

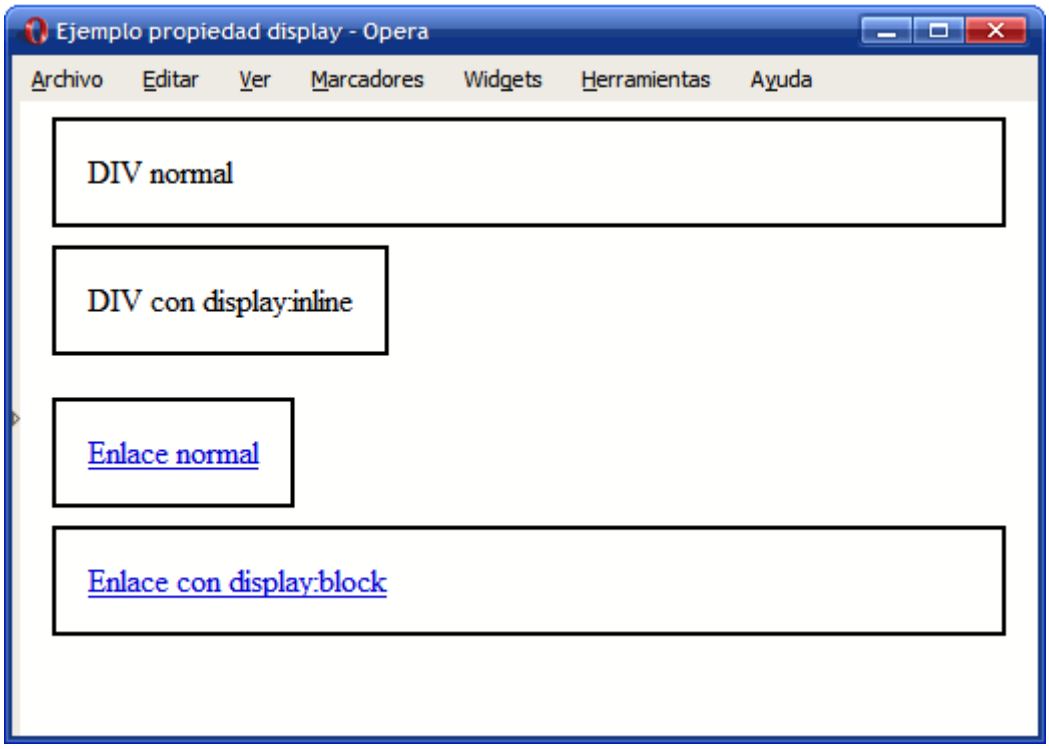

Figura 5.23. Ejemplo de propiedad display

Las reglas CSS del ejemplo anterior son las siguientes:

```
<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>
```

Por su parte, la definición completa de la **propiedad visibility** es mucho más sencilla:

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***

Las posibilidades de la propiedad visibility son mucho más limitadas que las de la **propiedad display**, ya que sólo ***permite hacer visibles o invisibles a los elementos de la página***.

Inicialmente todas las cajas que componen la página son visibles. Empleando el valor **hidden** es posible convertir una caja en invisible para que no muestre sus contenidos. El resto de elementos de la página se muestran como si la caja todavía fuera visible, por lo que en el lugar donde originalmente se mostraba la caja invisible, ahora se muestra un hueco vacío.

Por último, el **valor collapse** de la propiedad visibility sólo *se puede utilizar en las filas*, grupos de filas, columnas y grupos de columnas de una tabla. Su efecto es similar al de la propiedad display, ya que *oculta completamente la fila y/o columna y se pueden mostrar otros contenidos en ese lugar*. Si se utiliza el valor collapse sobre cualquier otro tipo de elemento, su efecto es idéntico al valor hidden.

5.8.2. Relación entre display, float y position

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

1) **Si display vale none**, se ignoran las propiedades float y position y *la caja no se muestra en la página*.
2) **Si position vale absolute o fixed**, la caja se posiciona de forma absoluta, se considera que *float vale none* y la propiedad *display vale block* tanto para los elementos en línea como para los elementos de bloque. La posición de la caja se determina mediante el valor de las propiedades top, right, bottom y left.
3) **En cualquier otro caso, si float tiene un valor distinto de none**, la caja se posiciona de forma flotante y la *propiedad display vale block* tanto para los elementos en línea como para los elementos de bloque.

5.8.3. Propiedad overflow

Normalmente, los contenidos de un elemento se pueden mostrar en el espacio reservado para ese elemento. Sin embargo, *en algunas ocasiones el contenido de un elemento no cabe en el espacio reservado para ese elemento y 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.

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

overflow	**Contenido sobrante ante un "desbordamiento"**
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***

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 valores típicos de la **propiedad overflow**:

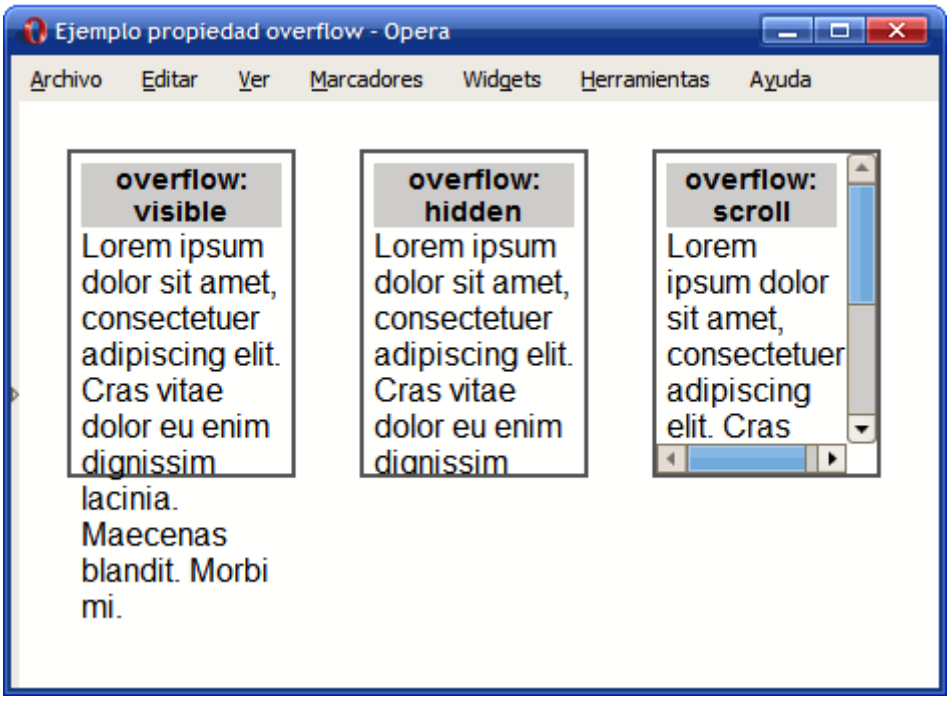

Figura 5.24. Ejemplo de propiedad overflow

El código HTML y CSS del ejemplo anterior se muestran a continuación:

```
div {
  float: left;
  margin: 1em;
  padding: .3em;
  border: 2px solid #555;
  width: 100px;
  height: 150px;
  font: 1em Arial, Helvetica, sans-serif;
}

        <div>
                <h1>overflow: visible</h1>
                <p>Lorem ipsum dolor sit amet, consectetuer  adipiscing elit. Cras vitae
                dolor eu enim dignissim lacinia. Maecenas blandit. Morbi mi. </p>
        </div>

        <div style="overflow:hidden">
                <h1>overflow: hidden</h1>
                <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Cras vitae
                dolor eu enim dignissim lacinia. Maecenas blandit. Morbi mi. </p>
        </div>

        <div style="overflow:scroll">
                <h1>overflow: scroll</h1>
                <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Cras vitae
                dolor eu enim dignissim lacinia. Maecenas blandit. Morbi mi. </p>
        </div>
```

5.8.4. Propiedad z-index

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

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

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

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

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

La siguiente imagen muestra un **ejemplo** de uso de la propiedad z-index:

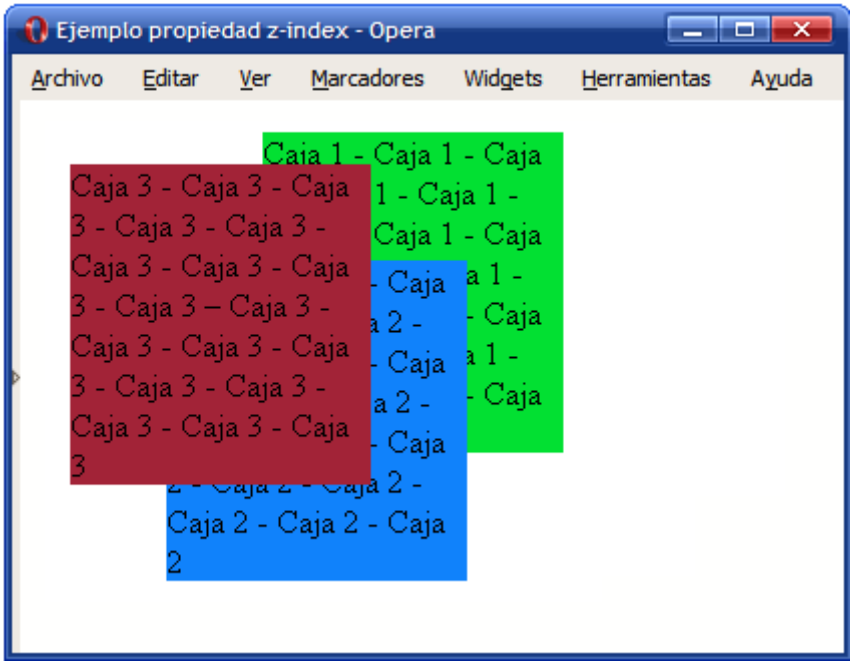

Figura 5.25. Ejemplo de propiedad z-index

El código HTML y CSS del ejemplo anterior es el siguiente:

```
div { position: absolute; width: 150px;}
#caja1 { z-index: 5;  top: 1em; left: 8em; background-color: #02e032; }
#caja2 { z-index: 15;  top: 5em; left: 5em; background-color: #1082fb; }
#caja3 { z-index: 25;  top: 2em; left: 2em; background-color: #a22337; }

<div id="caja1">
        Caja 1 - Caja 1 - Caja 1 - Caja 1 - Caja 1 - Caja 1 - Caja 1 - Caja 1 –
        Caja 1 - Caja 1 - Caja 1 - Caja 1 - Caja 1 - Caja 1 - Caja 1 - Caja 1
</div>

<div id="caja2">
        Caja 2 - Caja 2 - Caja 2 - Caja 2 - Caja 2 - Caja 2 - Caja 2 – Caja 2 –
        Caja 2 - Caja 2 - Caja 2 - Caja 2 - Caja 2 - Caja 2 - Caja 2 - Caja 2
</div>

<div id="caja3">
        Caja 3 - Caja 3 - Caja 3 - Caja 3 - Caja 3 - Caja 3 - Caja 3 - Caja 3 –
        Caja 3 - Caja 3 - Caja 3 - Caja 3 - Caja 3 - Caja 3 - Caja 3 - Caja 3
</div>
```

La propiedad z-index sólo tiene efecto en los elementos posicionados, por lo que es obligatorio que la propiedad z-index vaya acompañada de la propiedad position.

Si debes posicionar un elemento pero no quieres moverlo de su posición original ni afectar al resto de elementos de la página, puedes utilizar el posicionamiento relativo (position: relative).