Capítulo 5. Posicionamiento y visualización

5.1. Tipos de elementos

Los elementos HTML se clasifican en dos grandes grupos: elementos en línea y elementos de bloque.

Los elementos de bloque siempre empiezan en una nueva línea y ocupan todo el espacio disponible hasta el final de la línea. Los elementos en línea sin embargo no empiezan en nueva línea y sólo ocupan el espacio ocupado por sus contenidos.

La siguiente imagen muestra el espacio ocupado por los elementos en línea y los elementos de bloque en un documento HTML:

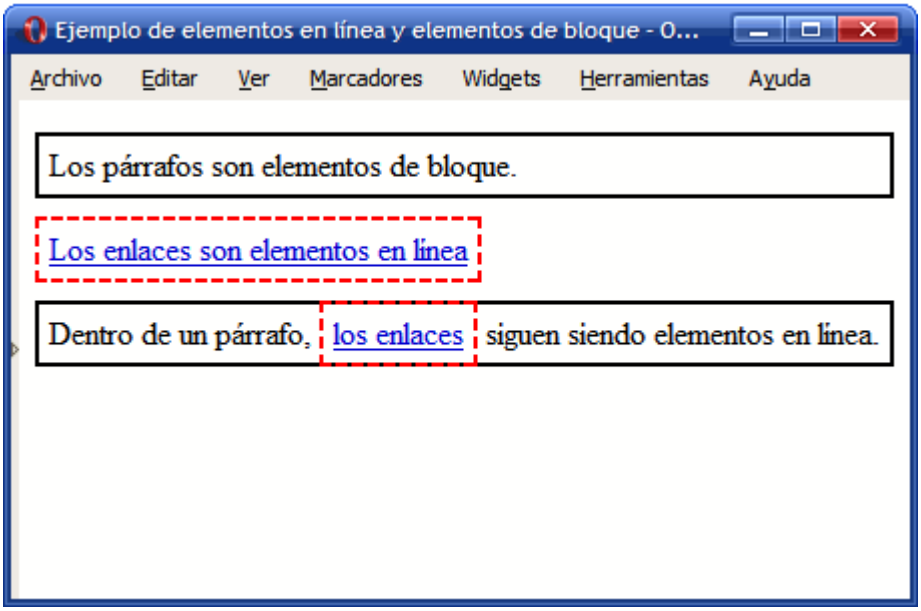

Figura 5.1. Cajas creadas por los elementos de línea y los elementos de bloque

Un elemento de bloque no puede aparecer dentro de un elemento en línea. En cambio, un elemento en línea puede aparecer dentro de un elemento de bloque y dentro de otro elemento en línea.

Los elementos en línea definidos por HTML son: `a`, `abbr`, `acronym`, `b`, `basefont`, `bdo`, `big`, `br`, `cite`, `code`, `dfn`, `em`, `font`, `i`, `img`, `input`, `kbd`, `label`, `q`, `s`, `samp`, `select`, `small`, `span`, `strike`, `strong`, `sub`, `sup`, `textarea`, `tt`, `u`, `var`.

Los elementos de bloque definidos por HTML son: `address`, `blockquote`, `center`, `dir`, `div`, `dl`, `fieldset`, `form`, `h1`, `h2`, `h3`, `h4`, `h5`, `h6`, `hr`, `isindex`, `menu`, `noframes`, `noscript`, `ol`, `p`, `pre`, `table`, `ul`.

Los siguientes elementos también se considera que son de bloque: `dd`, `dt`, `frameset`, `li`, `tbody`, `td`, `tfoot`, `th`, `thead`, `tr`.

Los siguientes elementos pueden ser en línea y de bloque según las circunstancias: `button`, `del`, `iframe`, `ins`, `map`, `object`, `script`.

5.2. Posicionamiento

Cuando los navegadores cargan una página web, muestran sus elementos de la mejor manera posible teniendo en cuenta las limitaciones de espacio y el tipo de cada elemento (de bloque o en línea). Además de esta forma automática, CSS permite controlar de forma precisa el lugar en el que se coloca cada elemento, es decir, el posicionamiento de cada elemento.

CSS define tres mecanismos para posicionar las cajas de cada elemento HTML:

- Normal: posicionamiento por defecto de los elementos en línea y los de bloque. Además, incluye el posicionamiento relativo.
- Float: posicionamiento que consiste en posicionar el elemento según el esquema normal y una vez colocado, desplazarlo todo lo posible hacia la izquierda o hacia la derecha.
- Absoluto: posicionamiento que consiste en extraer por completo el elemento de su posicionamiento normal y colocarlo en la posición indicada respecto de su elemento padre.

La propiedad de CSS que define el posicionamiento del elemento se denomina `position`.

Tabla 5.1. 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

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

- `static`: es el posicionamiento que se utiliza por defecto y todos los elementos inicialmente se muestran de esta forma. No se tienen en cuenta los valores de las propiedades `top`, `right`, `bottom` y `left`.
- `relative`: la nueva posición del elemento se calcula a partir de la posición que tendría si no se utilizara la propiedad `position`. Las posiciones de los elementos contiguos no se ven afectadas por el desplazamiento de este elemento.
- `absolute`: la nueva posición del elemento se determina mediante las propiedades `top`, `right`, `bottom` y `left`. Los valores de esas propiedades indican la posición del elemento respecto de la posición de su elemento padre. La posición del siguiente elemento se calcula como si no existiera el elemento que se desplaza, ya que este último se desentiende por completo del posicionamiento normal.

- `fixed`: la nueva posición del elemento se calcula de forma idéntica al posicionamiento absoluto. La diferencia reside en que en el caso de `fixed`, el elemento no se mueve cuando se desplaza la ventana del navegador. En los medios visuales (como la pantalla) el elemento se muestra en una posición fija independiente del movimiento de la ventana del navegador. En los medios impresos, el elemento se muestra en todas las páginas.

Las cuatro propiedades relacionadas con la propiedad `position` son las que determinan el desplazamiento de los elementos respecto de sus posiciones originales. CSS define para ello las propiedades `top`, `right`, `bottom` y `left`.

Tabla 5.2. Propiedad top, right, bottom, left

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

En el caso del posicionamiento relativo, cada una de estas propiedades indica el desplazamiento del elemento desde 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 elemento padre.

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** es la estrategia que utilizan por defecto los navegadores para mostrar los elementos de las páginas.

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 padre. La distancia de las cajas se controla mediante los márgenes verticales.

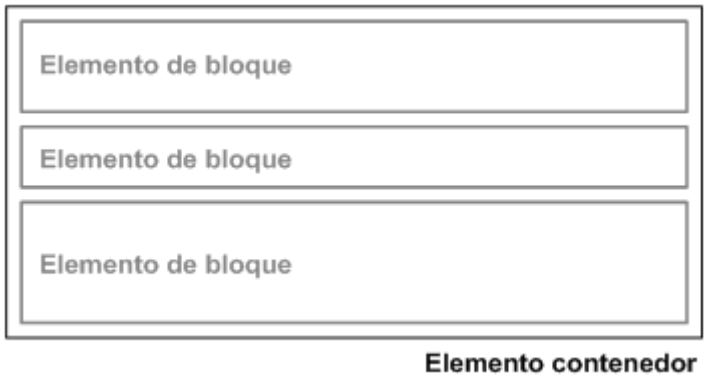

Figura 5.2. Posicionamiento normal de los elementos de bloque

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 del elemento padre. La distancia entre las cajas se controla mediante los márgenes laterales.

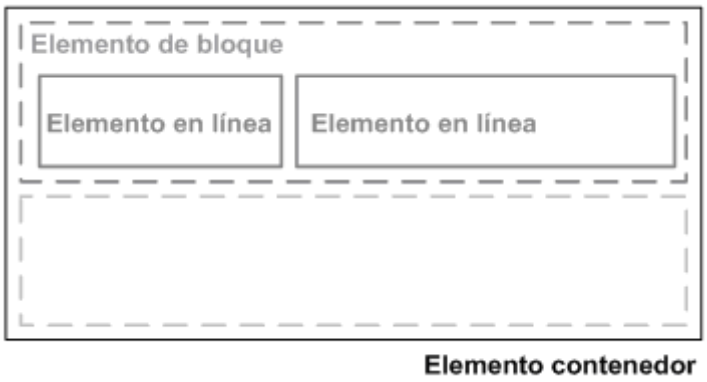

Figura 5.3. Posicionamiento normal de los elementos en línea

Si las cajas en línea ocupan más espacio del disponible en su propia línea, el resto de cajas se muestran en las líneas siguientes. 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.

Por otra parte, el **posicionamiento relativo** también se considera parte del posicionamiento normal. El posicionamiento relativo consiste en desplazar un elemento respecto de su posición original. En este caso, el desplazamiento de una caja no afecta al resto, que se muestran en la misma posición que si la caja desplazada no se hubiera movido de su sitio original.

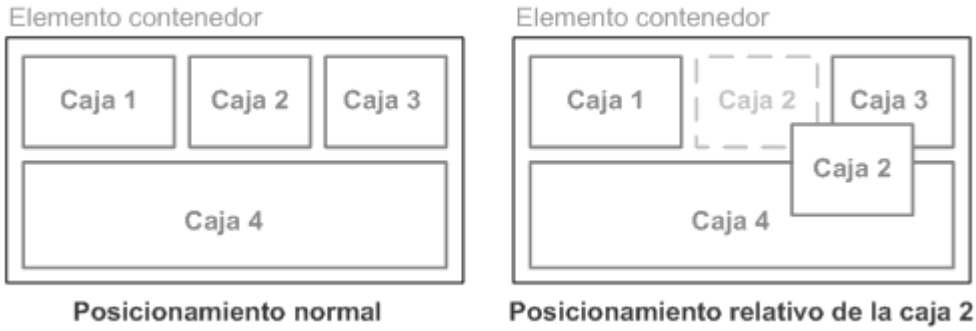

Figura 5.4. Diferencias visuales entre el posicionamiento normal y el posicionamiento relativo

El siguiente ejemplo muestra tres imágenes con el posicionamiento normal definido por CSS:

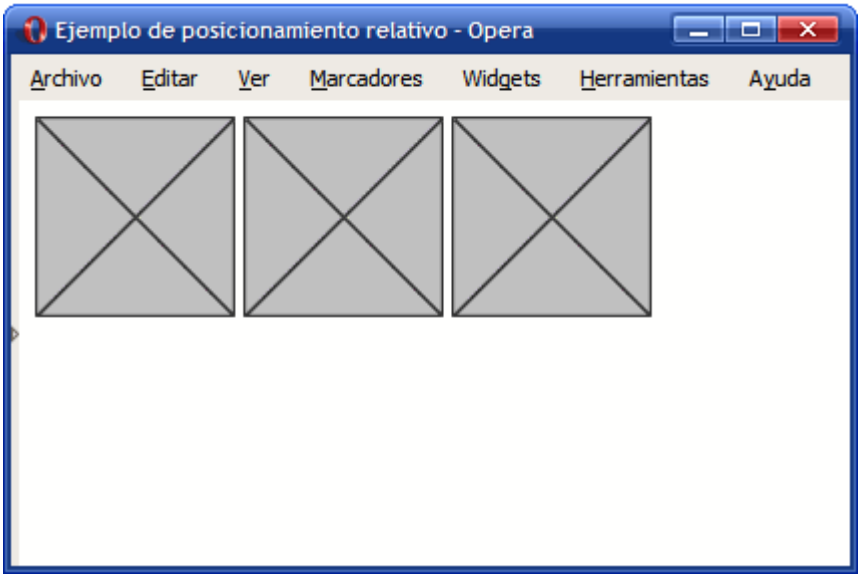

Figura 5.5. Elementos posicionados de forma normal

Aplicando el posicionamiento relativo, se desplaza 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:

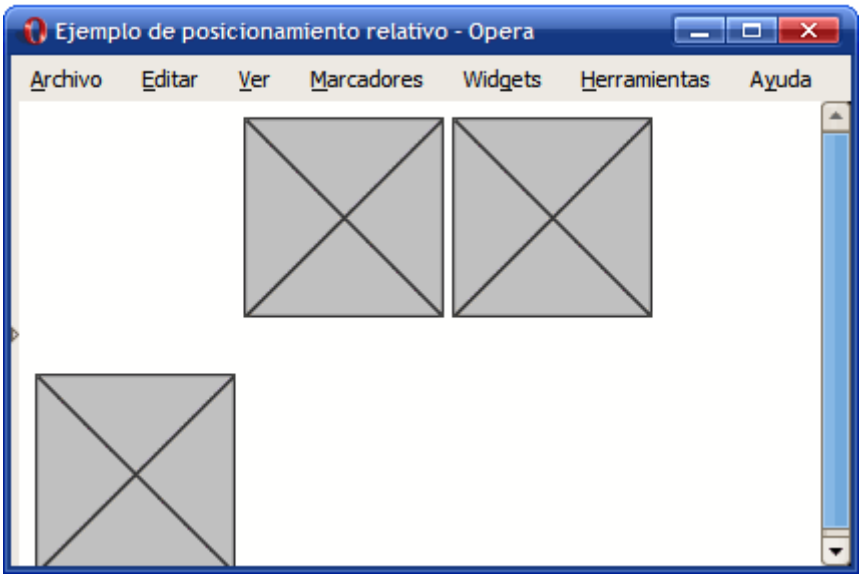

Figura 5.6. Elemento posicionado de forma relativa

El resto de imágenes no varían su posición y no ocupan el hueco dejado por la primera imagen, porque el posicionamiento relativo no influye en el resto de elementos de la página. La única incidencia que tiene el posicionamiento relativo sobre el resto de elementos, es que el elemento desplazado puede solaparse con otros elementos de la página.

5.4. Posicionamiento float

El **posicionamiento float** es uno de los más utilizados, sobre todo al definir la estructura de las páginas, como se verá más adelante. Una caja posicionada mediante la propiedad `float`, se desplaza hasta la zona más a la izquierda o más a la derecha de la línea en la que se debería mostrar si no se desplazara.

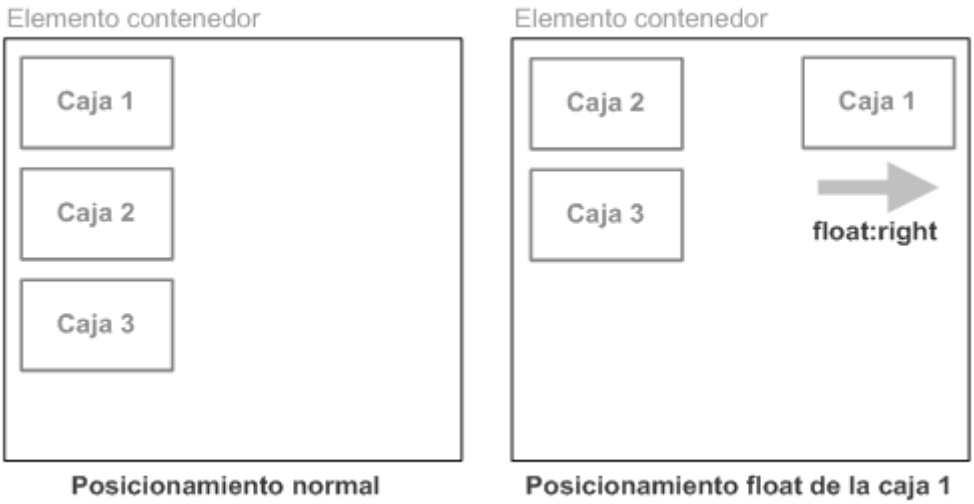

Figura 5.7. Ejemplo de posicionamiento float de una caja

Una caja desplazada mediante `float` no pertenece al posicionamiento normal de un documento, por lo que los elementos de bloque anteriores y posteriores se visualizan como si la caja desplazada no existiera.

Si en el anterior ejemplo la "Caja 1" se posiciona mediante un `float: left` el resultado sería el que se muestra en la imagen:

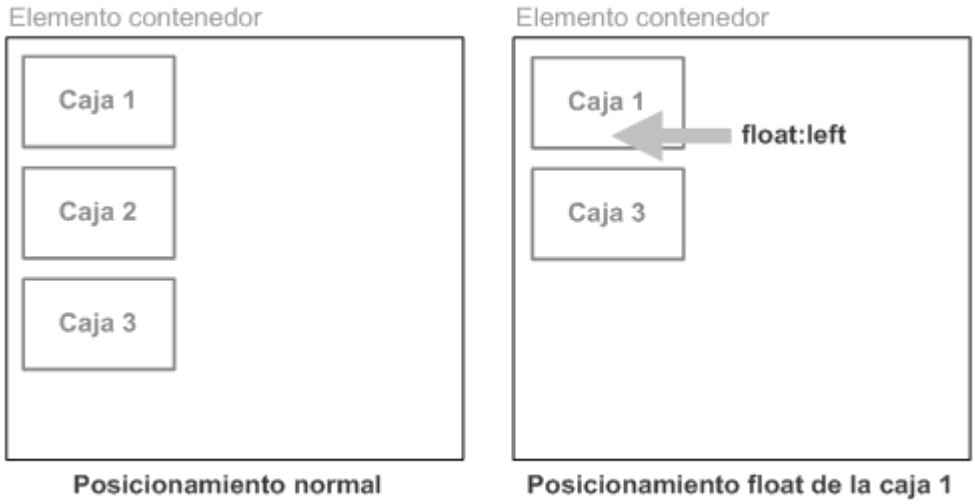

Figura 5.8. Ejemplo de posicionamiento float de una caja

En las imágenes anteriores, el resto de cajas ocupan la posición libre dejada por la "Caja 1" original. Como la "Caja 1" está desplazada a la izquierda, se coloca por encima de la nueva posición que ocupa la "Caja 2", a la que cubre por completo.

Si existen otras cajas desplazadas hacia la izquierda o derecha, la nueva caja desplazada se coloca al lado de las demás cajas. El siguiente ejemplo desplaza las tres cajas:

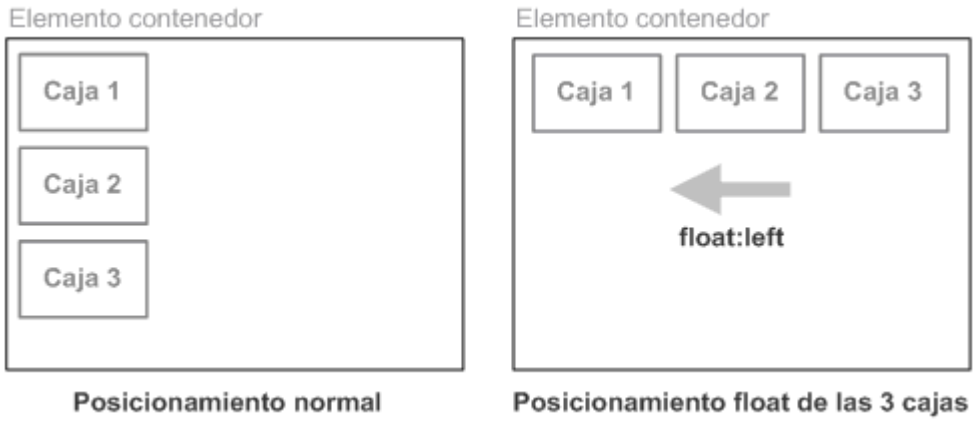

Figura 5.9. Ejemplo de posicionamiento float de varias cajas

Si no existe sitio en la línea actual, la caja pasa a la siguiente línea hasta que encuentra el sitio necesario para mostrarse a la izquierda o derecha de la línea.

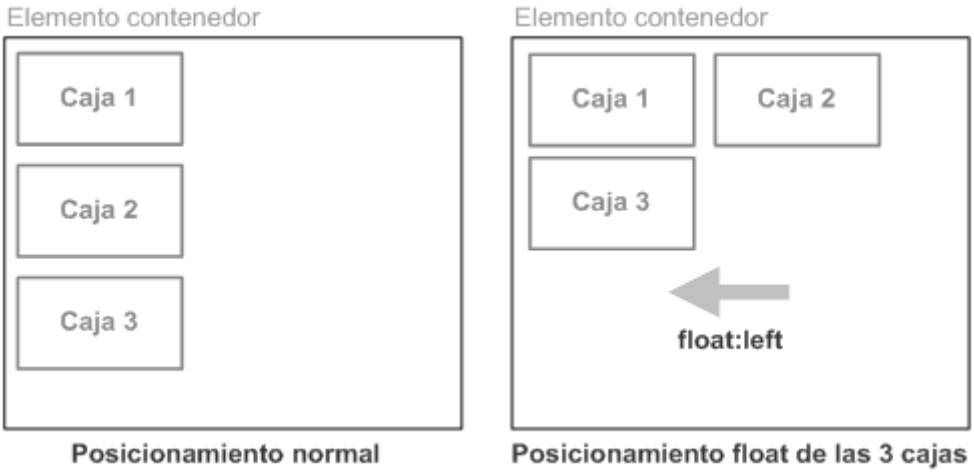

Figura 5.10. Ejemplo de posicionamiento float cuando no existe sitio suficiente

Una caja posicionada mediante `float` influye en la disposición de todas las demás cajas. CSS permite definir si el resto de los elementos fluyen alrededor de la caja desplazada o no lo hacen.

Los elementos en línea que se encuentran al lado de las cajas desplazadas mediante `float` adaptan su anchura al espacio libre dejado por la caja desplazada. Si en la línea donde se encuentra la caja desplazada no existe sitio necesario para los contenidos de los elementos en línea, estos se visualizan en la línea inmediatamente inferior.

Tabla 5.3. Propiedad float

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

Los posibles valores de la propiedad determinan el posicionamiento del elemento y el comportamiento de los elementos adyacentes. Si se indica un valor `left`, el elemento se desplaza hasta el punto más a la izquierda posible en esa misma línea (si no existe sitio en esa línea, el elemento 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 del elemento desplazado.

El valor `right` tiene un funcionamiento idéntico, salvo que en este caso el elemento se desplaza hacia la derecha. El valor `none` permite eliminar el posicionamiento y que el elemento se muestre en su posición original.

Ejercicio 6 Ver enunciado en la página 186

Los elementos que se encuentran alrededor de un elemento que ha sido posicionado mediante `float`, adaptan sus contenidos para que fluyan alrededor del elemento posicionado:

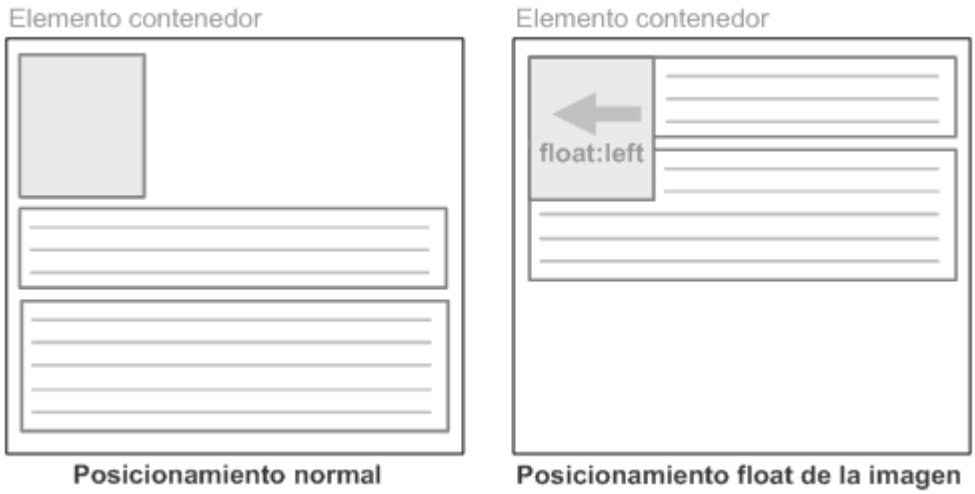

Figura 5.11. Elementos que fluyen alrededor de un elemento posicionado mediante float

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.

La flexibilidad de 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 el texto del primer párrafo fluya alrededor de una imagen, pero el resto de los párrafos deberían mostrarse en su totalidad y no fluyendo alrededor de la imagen:

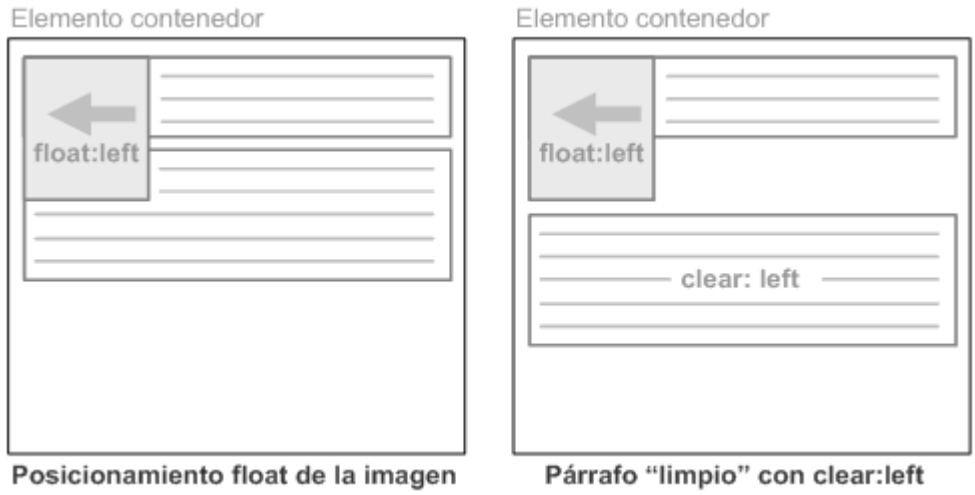

Figura 5.12. Forzando a que un elemento no fluya alrededor de otro elemento posicionado mediante float

La propiedad `clear` permite contrarrestar el comportamiento por defecto de los `float` y permite forzar a un elemento a mostrarse debajo de cualquier elemento posicionado con `float`.

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

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

Tabla 5.4. Propiedad clear

`clear`	**Despejar los elementos adyacentes**
Valores	`none` | `left` | `right` | `both` | `inherit`
Se aplica a	Todos los elementos de bloque
Valor inicial	none
Descripción	Permite limpiar el lado derecho, izquierdo o ambos de un elemento, para que no se muestren elementos adyacentes

La propiedad `clear` indica el lado del elemento HTML que no debe ser adyacente a otros elementos desplazados mediante `float`.

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 ningún elemento 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 desplazado hacia la izquierda"*.

Si se indica el valor `right`, el comportamiento es el mismo, 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 de cualquier borde inferior de los elementos desplazados hacia la izquierda y hacia la derecha.

Como se verá más adelante, la propiedad `clear` es imprescindible para la creación de estructuras y layouts de páginas complejas. En el ejercicio anterior, se utilizaba `float` para desplazar los dos elementos:

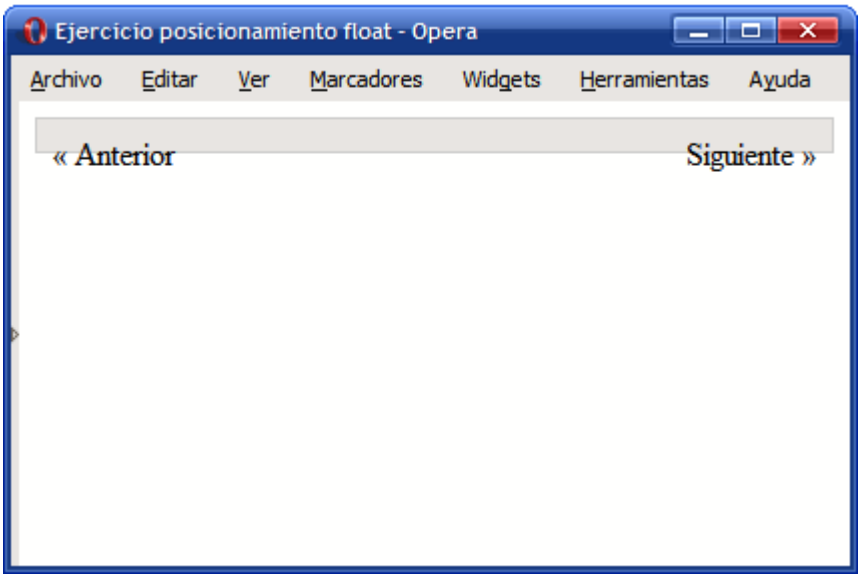

Figura 5.13. Visualización incorrecta de dos elementos posicionados mediante float

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

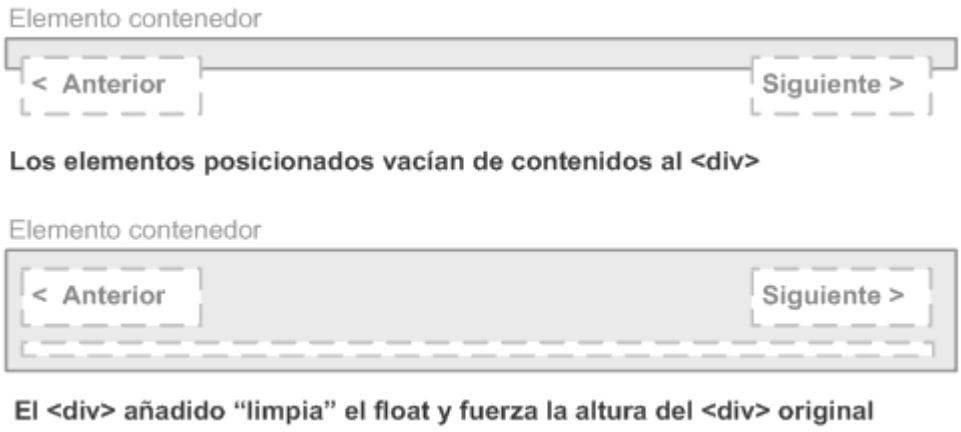

Figura 5.14. Esquema del problema y solución de la visualización incorrecta de 2 elementos posicionados mediante float

La solución consiste en añadir un elemento adicional invisible que *limpie* el `float` forzando a que el `<div>` original cubra completamente los dos elementos `<span>`. 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 `<div>` con la propiedad `clear: both`, se tiene la seguridad de que el `<div>` añadido se va a mostrar debajo de cualquier elemento posicionado con `float` y por tanto, se asegura que el `<div>` original tenga la altura necesaria como para encerrar a todos sus contenidos posicionados con `float`.

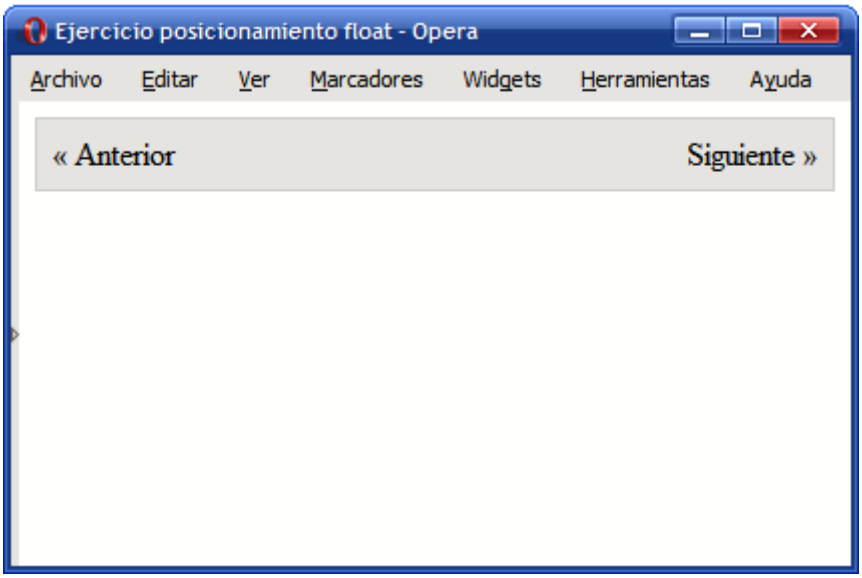

Figura 5.15. Visualización correcta de 2 elementos posicionados mediante float

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

5.5. Posicionamiento absoluto

El **posicionamiento absoluto** implica que el elemento desplazado sale por completo del flujo normal del documento y por tanto, la posición del resto de elementos se determina como si no existiera el elemento desplazado.

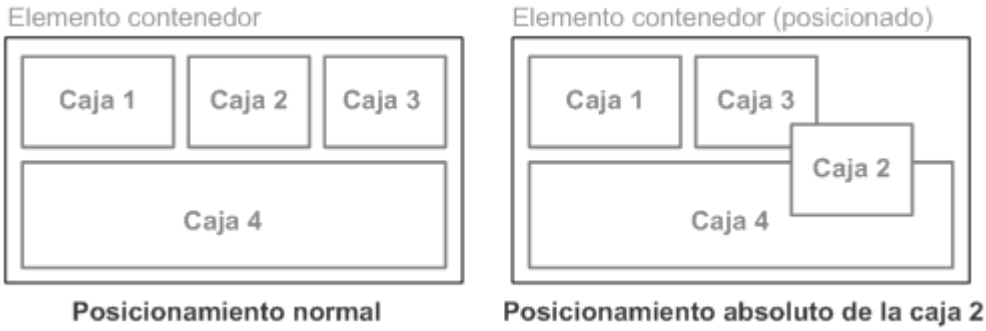

Figura 5.16. Diferencias visuales entre el posicionamiento normal y el posicionamiento absoluto

La referencia que toma el posicionamiento absoluto para determinar la nueva posición de un elemento es el primer elemento padre que esté posicionado. Si ningún elemento padre del elemento posicionado tiene establecido un posicionamiento, la referencia se toma respecto del documento HTML completo.

El siguiente ejemplo muestra las diferencias entre un elemento padre posicionado y otro que no lo está:

Situación original:

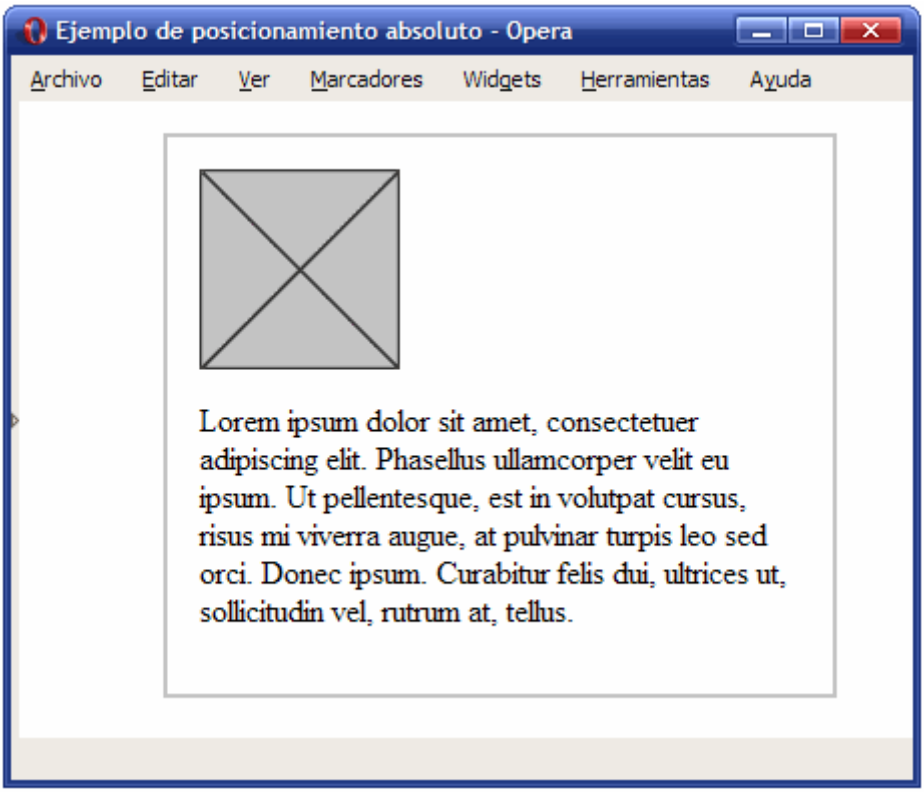

Figura 5.17. Situación original antes de modificar el posicionamiento

Código HTML 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>
```

A continuación, se posiciona la imagen de forma absoluta y con unos valores `top` y `left` iguales a `3em`:

```
div img {
  position: absolute;
  top: 3em;
  left: 3em;
}
```

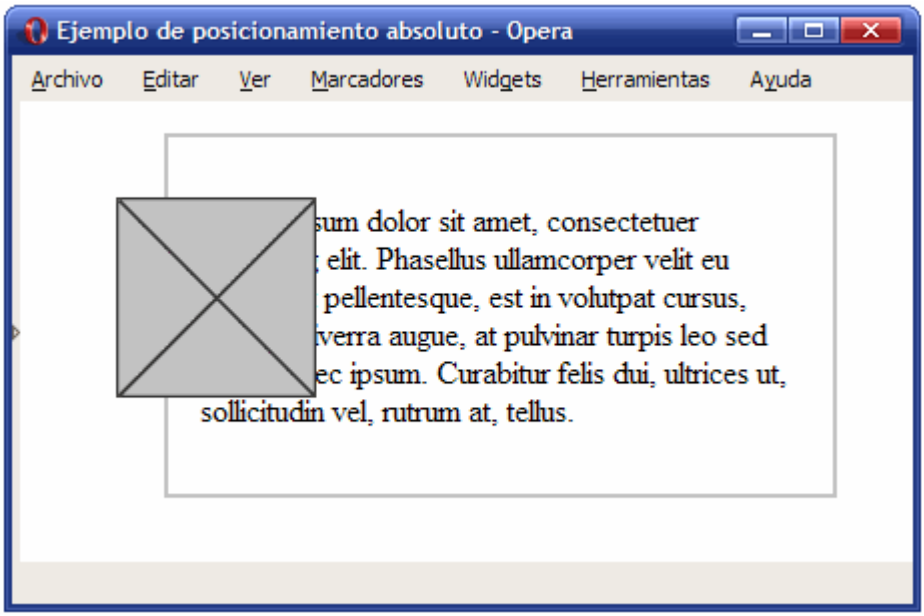

Figura 5.18. Imagen posicionada de forma absoluta

La imagen posicionada absolutamente no toma como origen la esquina superior izquierda del `<div>` en el que se encuentra, sino que su referencia es la esquina superior izquierda de la página:

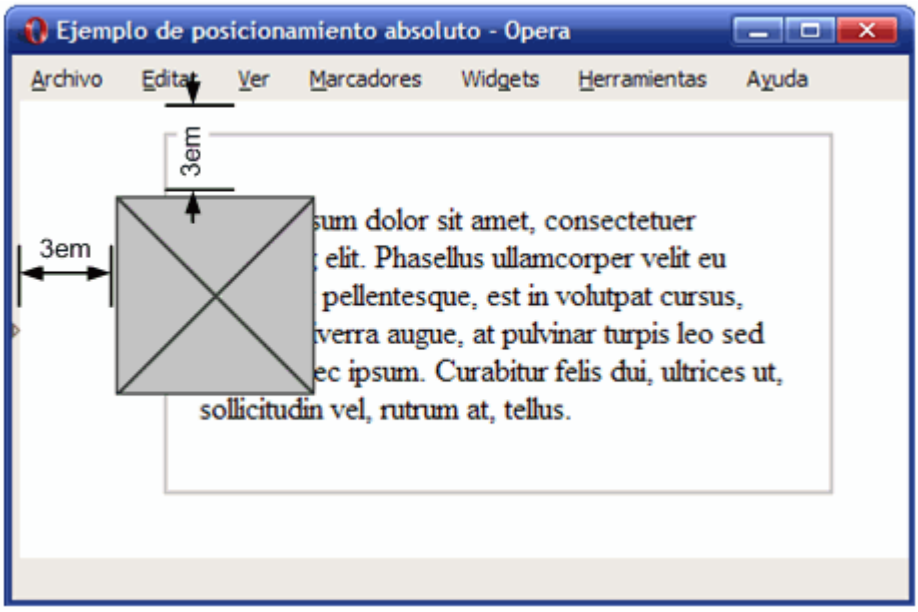

Figura 5.19. La referencia del posicionamiento absoluto es la página entera

Sin embargo, si el elemento contenedor de la imagen (es decir, el elemento `<div>`) se posiciona de forma relativa, la imagen posicionada de forma absoluta varía su posición:

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

```
  position: relative;
}

div img {
  position: absolute;
  top: 3em;
  left: 3em;
}
```

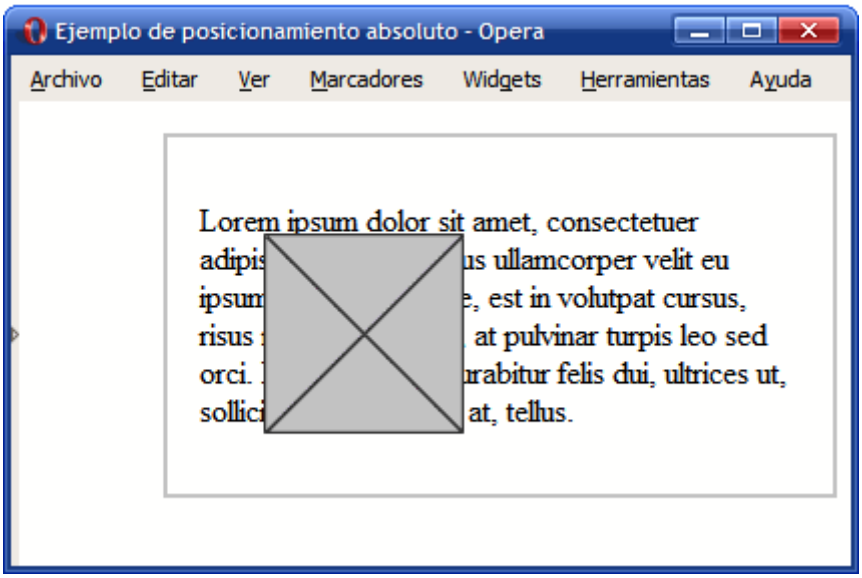

Figura 5.20. Imagen posicionada de forma absoluta

En este caso, la imagen sí que toma correctamente la referencia de su posición desde la esquina superior izquierda el elemento `<div>`:

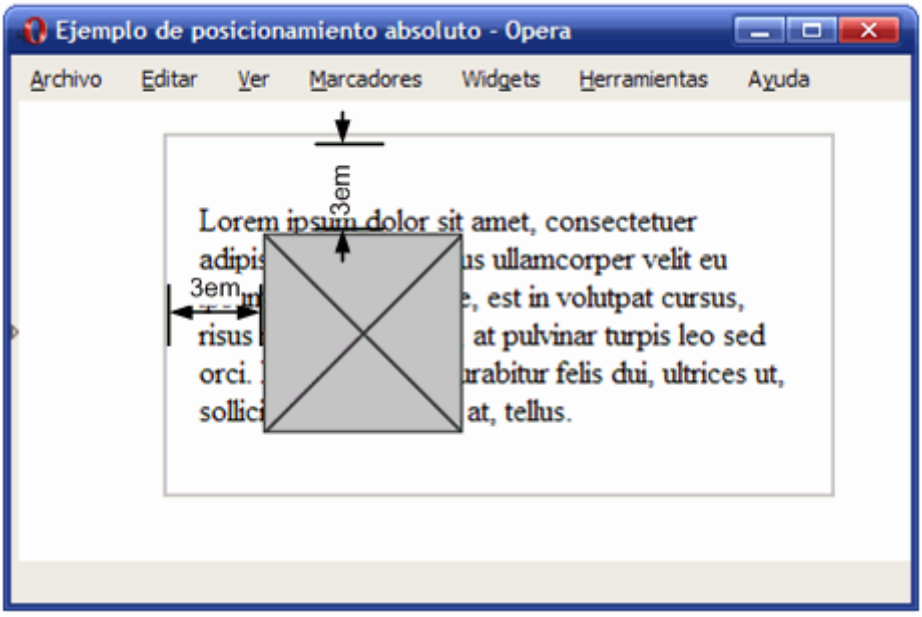

Figura 5.21. La referencia del posicionamiento absoluto es el elemento contenedor de la imagen

Por otra parte, el **posicionamiento fijo** es un caso especial de posicionamiento absoluto. Los elementos que utilizan un posicionamiento fijo se muestran de forma estática en la ventana del navegador y no varían su posición ni aunque se desplace la ventana del navegador o se haga *scroll* sobre ella.

Desafortunadamente, Internet Explorer 6 y las versiones anteriores no soportan el valor `fixed` para la propiedad `position` y se debe simular su comportamiento mediante JavaScript o mediante *hacks* específicos de ese navegador, como se verá más adelante.

5.6. Visualización

Además de las propiedades que controlan el posicionamiento de los elementos, CSS define otras cuatro propiedades que controlan la visualización de todos los elementos: `display`, `visibility`, `overflow` y `z-index`.

5.6.1. Display y visibility

Las propiedades `display` y `visibility` controlan la visualización de los elementos. Las dos propiedades permiten hacer invisible cualquier elemento de la página. Habitualmente se utilizan junto con JavaScript para crear efectos y aplicaciones dinámicas (mostrar y ocultar determinados textos, hacer aparecer imágenes, etc.)

La siguiente imagen muestra las diferencias entre ocultar una caja (la número 5) con la propiedad `display` y con la propiedad `visibility`.

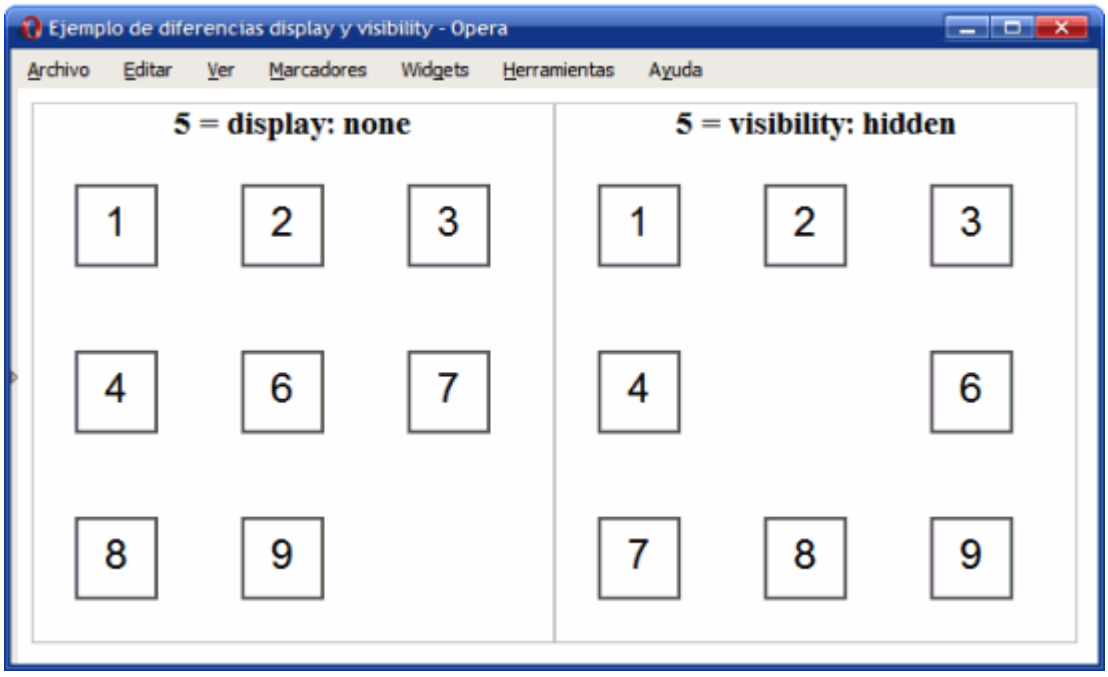

Figura 5.22. Diferencias visuales entre las propiedades display y visibility

La propiedad `display` se utiliza para no visualizar el elemento y para que el resto de elementos se muestren como si el elemento oculto no existiera. La propiedad `visibility` permite mantener la estructura de la página pero sin mostrar el elemento. En este caso, la posición del resto de elementos sí que se mantiene igual que si el elemento no

estuviera oculto. Por este motivo, la propiedad `display` se utiliza mucho más que la propiedad `visibility`.

Tabla 5.5. 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

Los valores más utilizados son `inline`, `block` y `none`. 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 no mostrarlo en el documento. 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.

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:

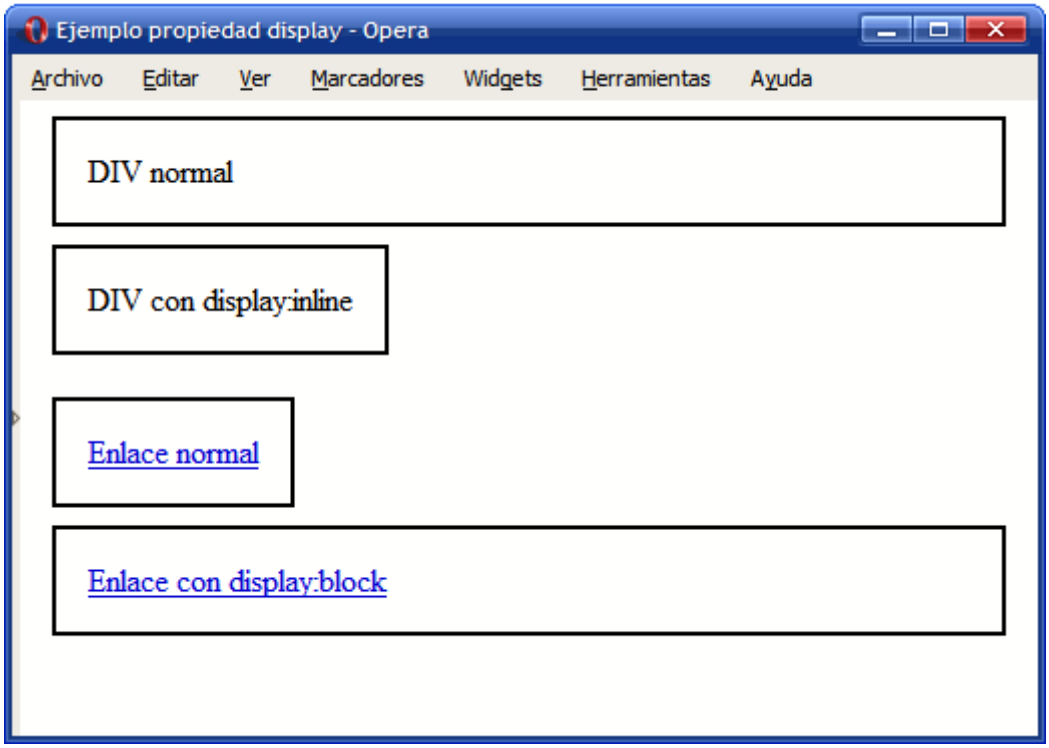

Figura 5.23. Ejemplo de propiedad display

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>
```

La propiedad `display:inline` se utiliza habitualmente con los elementos `<div>` que se quieren posicionar mediante `float`. También se utiliza para las listas (`<ul>`, `<ol>`) que se quieren mostrar horizontalmente. La propiedad `display:block` se emplea frecuentemente para los enlaces que forman el menú de navegación.

Tabla 5.6. Propiedad visibility

`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

Por defecto, todos los elementos se consideran como visibles (`visibility: visible`). Sin embargo, es posible hacer invisible un elemento asignándole el valor `hidden`. Aunque el elemento no sea visible en pantalla, el resto de elementos se muestran como si el elemento sí que fuera visible, es decir, que en el lugar donde originalmente se mostraba el elemento, ahora se muestra un hueco vacío.

5.6.2. Overflow

Normalmente, los contenidos de los elementos 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.

La situación más habitual en la que el contenido sobresale su espacio reservado es cuando se establece la anchura de un elemento mediante la propiedad `width`. 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.

Tabla 5.7. Propiedad overflow

`overflow`	**Parte sobrante de un elemento**
Valores	`visible` | `hidden` | `scroll` | `auto` | `inherit`

`overflow`	**Parte sobrante de un elemento**
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.
- `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`:

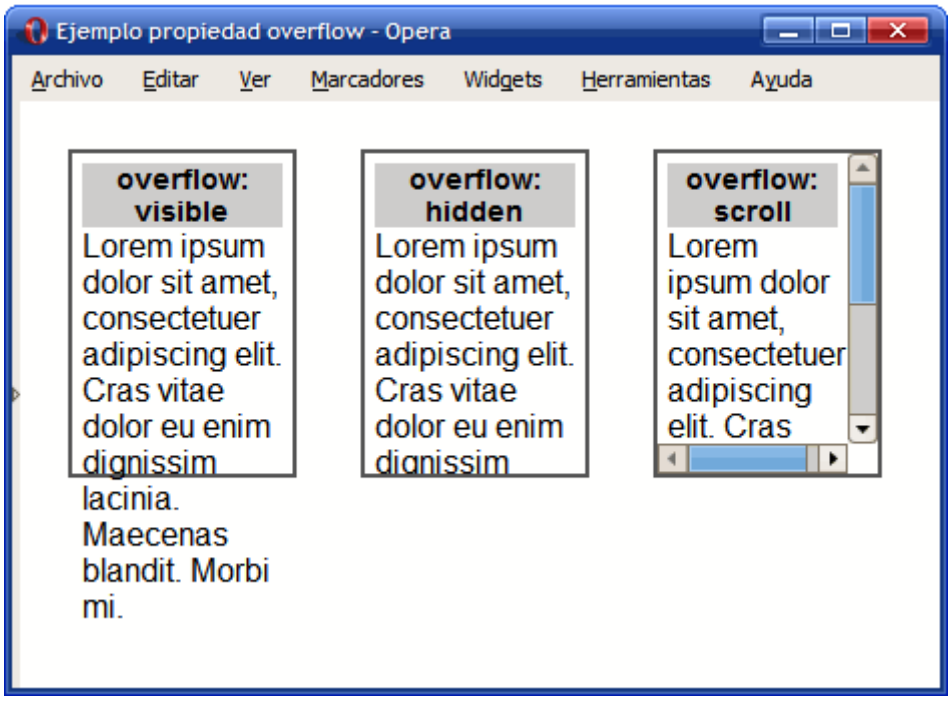

Figura 5.24. Ejemplo de propiedad overflow

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

```
div {
  display: inline;
  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> Lorem ipsum dolor sit amet, consectetuer
adipiscing elit. Cras vitae dolor eu enim dignissim lacinia. Maecenas
blandit. Morbi mi.</div>

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

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

El valor `visible` es el valor por defecto que se aplica a los elementos HTML. El valor `hidden` oculta todos los elementos que no caben en la zona reservada para el elemento. Por último, el valor `scroll` solamente muestra los contenidos que caben en la zona reservada pero también muestra barras de *scroll* que permiten visualizar el resto del contenido.

5.6.3. Z-index

Además de la posición horizontal y vertical, CSS permite asignar una posición tridimensional a cada caja del documento. La posición de las cajas se controla mediante un eje adicional llamado Z, por lo que la propiedad se denomina `z-index`.

La propiedad `z-index` permite crear páginas web complejas con varios niveles o capas. El orden de las capas determina los elementos que se mostrarán encima o debajo de otros elementos. Por este motivo, si se utiliza la propiedad `z-index` se pueden producir solapamientos entre los diferentes elementos de la página.

Tabla 5.8. Propiedad z-index

`z-index`	**Orden tridimensional**
Valores	`auto` | `<numero>` | `inherit`
Se aplica a	Elementos que han sido posicionado 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, más cerca del usuario se mostrará ese nivel. Un elemento con `z-index: 10` se mostrará por delante de otros elementos con `z-index: 8` o `z-index: 9`, pero por detrás de elementos con `z-index: 20` o `z-index: 50`.

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

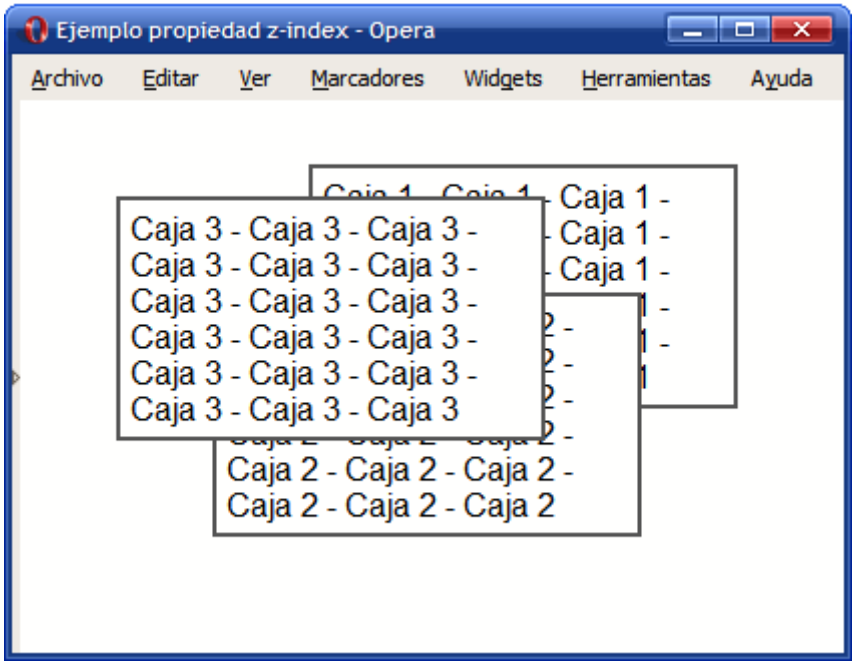

Figura 5.25. Ejemplo de propiedad z-index

El código CSS utilizado se muestra a continuación:

```
div { position: absolute; }
#caja1 { z-index: 5;  top: 1em; left: 8em;}
#caja2 { z-index: 15;  top: 5em; left: 5em;}
#caja3 { z-index: 25;  top: 2em; left: 2em;}

<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 - 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 - 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 - Caja 3</div>
```

Capítulo 6. Texto

6.1. Tipografía

CSS define numerosas propiedades para establecer la apariencia del texto y de la letra utilizada en el texto. A pesar de que no dispone de tantas posibilidades como los lenguajes y programas específicos para crear documentos impresos, CSS permite aplicar estilos complejos y muy variados al texto de las páginas.

La propiedad básica que define CSS relacionada con la tipografía se denomina `font-family` y se utiliza para indicar el tipo de letra con el que se muestra el texto.

Tabla 6.1. Propiedad font-family

`font-family`	**Tipo de letra**
Valores	`(( <nombre_familia> | <familia_generica> ) (,<nombre_familia> | <familia_generica>)* ) | inherit`
Se aplica a	Todos los elementos
Valor inicial	Depende del navegador
Descripción	Establece el tipo de letra utilizado para el texto

El tipo de letra del texto se puede indicar de dos formas diferentes:

- Mediante el nombre de una *familia* tipográfica: en otras palabras, mediante el nombre del tipo de letra, como por ejemplo "Arial", "Verdana", "Garamond", etc.
- Mediante el nombre genérico de una *familia* tipográfica: los nombres genéricos no se refieren a ninguna fuente en concreto, sino que hacen referencia al estilo del tipo de letra. Las familias genéricas definidas son `serif` (tipo de letra similar a *Times New Roman*), `sans-serif` (tipo *Arial*), `cursive` (tipo *Comic Sans*), `fantasy` (tipo *Impact*) y `monospace` (tipo *Courier New*).

Los navegadores muestran el texto de las páginas web utilizando los tipos de letra instalados en el ordenador del propio usuario. De esta forma, si el diseñador indica en la propiedad `font-family` que el texto debe mostrarse con un tipo de letra especialmente raro o rebuscado, casi ningún usuario dispondrá de ese tipo de letra.

Para evitar el problema común de que el usuario no tenga instalada la fuente que quiere utilizar el diseñador, CSS permite indicar en la propiedad `font-family` más de un tipo de letra. El navegador probará en primer lugar con el primer tipo de letra indicado. Si el usuario la tiene instalada, el texto se muestra con ese tipo de letra.

Si el usuario no dispone del primer tipo de letra indicado, el navegador irá probando con el resto de tipos de letra hasta que encuentre alguna fuente que esté instalada en el ordenador del usuario. Evidentemente, el diseñador no puede indicar para cada propiedad `font-family` tantos tipos de letra como posibles fuentes parecidas existan.

Para solucionar este problema se utilizan las familias tipográficas genéricas. Cuando la propiedad `font-family` toma un valor igual a `sans-serif`, el diseñador no indica al navegador que debe utilizar la fuente Arial, sino que debe utilizar *"la fuente que más se parezca a Arial de todas las que tiene instaladas el usuario"*.

Por todo ello, el valor de `font-family` suele definirse como una lista de tipos de letra alternativos separados por comas. El último valor de la lista es el nombre de la familia tipográfica genérica que más se parece al tipo de letra que se quiere utilizar.

Las listas de tipos de letra más utilizadas son las siguientes:

```
font-family: Arial, Helvetica, sans-serif;
font-family: "Times New Roman", Times, serif;
font-family: "Courier New", Courier, monospace;
font-family: Georgia, "Times New Roman", Times, serif;
font-family: Verdana, Arial, Helvetica, sans-serif;
```

Ya que las fuentes que se utilizan en la página deben estar instaladas en el ordenador del usuario, cuando se quiere disponer de un diseño complejo con fuentes muy especiales, se debe recurrir a soluciones alternativas.

La solución más sencilla consiste en crear imágenes en las que se muestra el texto con la fuente deseada. Esta técnica solamente es viable para textos cortos (por ejemplo los titulares de una página) y puede ser manual (creando las imágenes una por una) o automática, utilizando JavaScript, PHP y/o CSS.

Otra alternativa es la de la sustitución automática de texto basada en Flash. La técnica más conocida es la de sIFR, de la que se puede encontrar más información en `http://wiki.novemberborn.net/sifr`

Una vez seleccionado el tipo de letra, se puede modificar su tamaño mediante la propiedad `font-size`.

Tabla 6.2. Propiedad font-size

`font-size`	**Tamaño de letra**
Valores	`<tamaño_absoluto>` | `<tamaño_relativo>` | `<medida>` | `<porcentaje>` | `inherit`
Se aplica a	Todos los elementos
Valor inicial	medium
Descripción	Establece el tamaño de letra utilizado para el texto

Además de todas las unidades de medida relativas y absolutas y el uso de porcentajes, CSS permite utilizar una serie de palabras clave para indicar el tamaño de letra del texto:

- `tamaño_absoluto`: indica el tamaño de letra de forma absoluta mediante alguna de las siguientes palabras clave: `xx-small`, `x-small`, `small`, `medium`, `large`, `x-large`, `xx-large`.

- `tamaño_relativo`: indica de forma relativa el tamaño de letra del texto mediante dos palabras clave (`larger`, `smaller`) que toman como referencia el tamaño de letra del elemento padre.

La siguiente imagen muestra una comparación entre los tamaños típicos del texto y las unidades que más se utilizan:

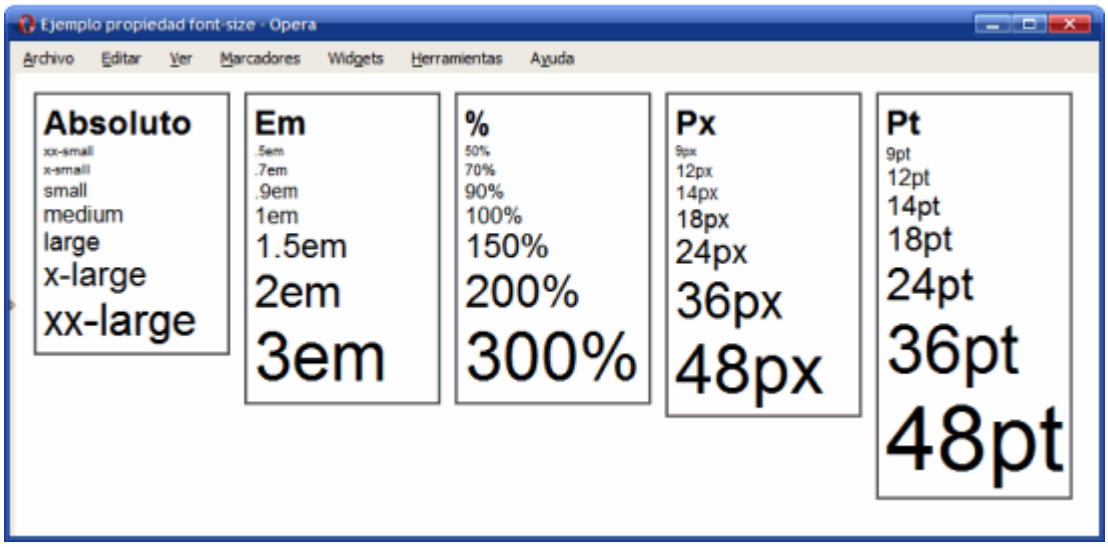

Figura 6.1. Comparación visual de las distintas unidades para indicar el tamaño del texto

CSS recomienda indicar el tamaño del texto en la unidad `em` o en porcentaje (`%`). Además, es habitual indicar el tamaño del texto en puntos (`pt`) cuando el documento está específicamente diseñado para imprimirlo.

Por defecto los navegadores asignan los siguientes tamaños a los títulos de sección: `<h1>` = `xx-large`, `<h2>` = `x-large`, `<h3>` = `large`, `<h4>` = `medium`, `<h5>` = `small`, `<h6>` = `xx-small`.

Una vez indicado el tipo y el tamaño de letra, es habitual modificar otras características como su anchura (texto en negrita) y su estilo (texto en cursiva). La propiedad que controla la anchura de la letra es `font-weight`.

Tabla 6.3. Propiedad font-weight

font-weight	**Anchura de la letra**
Valores	`normal` | `bold` | `bolder` | `lighter` | `100` | `200` | `300` | `400` | `500` | `600` | `700` | `800` | `900` | `inherit`
Se aplica a	Todos los elementos
Valor inicial	normal
Descripción	Establece la anchura de la letra utilizada para el texto

Los valores que normalmente se utilizan son `normal` (el valor por defecto) y `bold` para los textos en negrita. El valor `normal` equivale al valor numérico `400` y el valor `bold` al valor numérico `700`.

El siguiente ejemplo muestra una aplicación práctica de la propiedad `font-weight`:

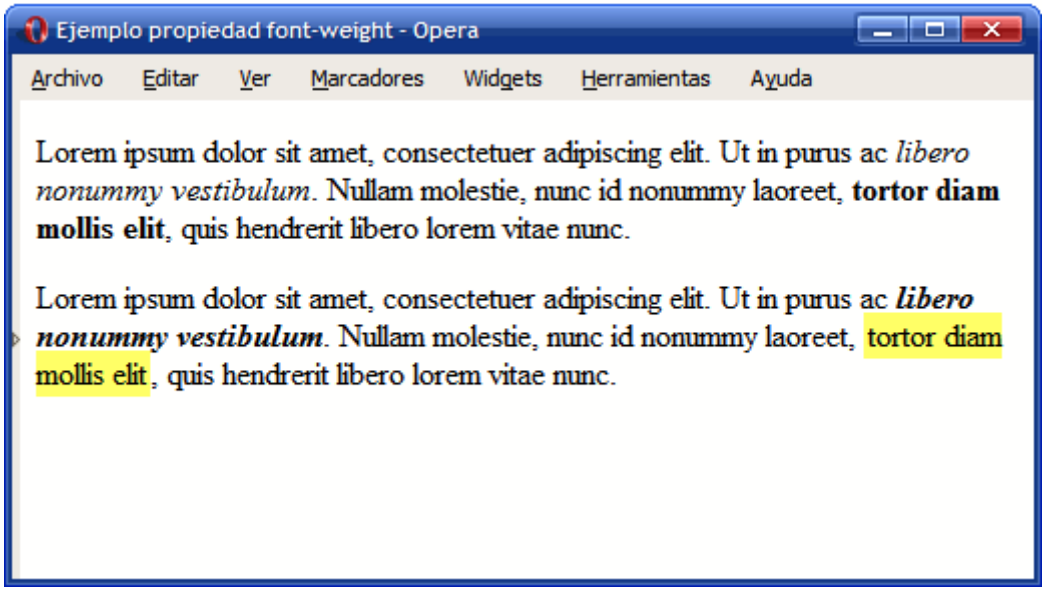

Figura 6.2. Ejemplo de propiedad font-weight

Por defecto, los navegadores muestran el texto de los elementos `<em>` en cursiva y el texto de los elementos `<strong>` en negrita. La propiedad `font-weight` permite alterar ese aspecto por defecto y mostrar por ejemplo los elementos `<em>` como cursiva y negrita y los elementos `<strong>` destacados mediante un color de fondo y sin negrita.

Las reglas CSS del ejemplo anterior se muestran a continuación:

```
#especial em {
  font-weight: bold;
}
#especial strong {
  font-weight: normal;
  background-color: #FFFF66;
  padding: 2px;
}

<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut in
purus ac <em>libero nonummy vestibulum</em>. Nullam molestie, nunc id
nonummy laoreet, <strong>tortor diam mollis elit</strong>, quis hendrerit
libero lorem vitae nunc.</p>

<p id="especial">Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
Ut in purus ac <em>libero nonummy vestibulum</em>. Nullam molestie, nunc id
nonummy laoreet, <strong>tortor diam mollis elit</strong>, quis hendrerit
libero lorem vitae nunc.</p>
```

Además de la anchura de la letra, CSS permite variar su estilo mediante la propiedad `font-style`.

Tabla 6.4. Propiedad font-style

font-style	Estilo de la letra
Valores	normal | italic | oblique | inherit
Se aplica a	Todos los elementos
Valor inicial	normal
Descripción	Establece el estilo de la letra utilizada para el texto

Normalmente la propiedad `font-style` se emplea para mostrar un texto en cursiva mediante el valor `italic`.

El ejemplo anterior se puede modificar para personalizar aun más el aspecto por defecto de los elementos `<em>` y `<strong>`:

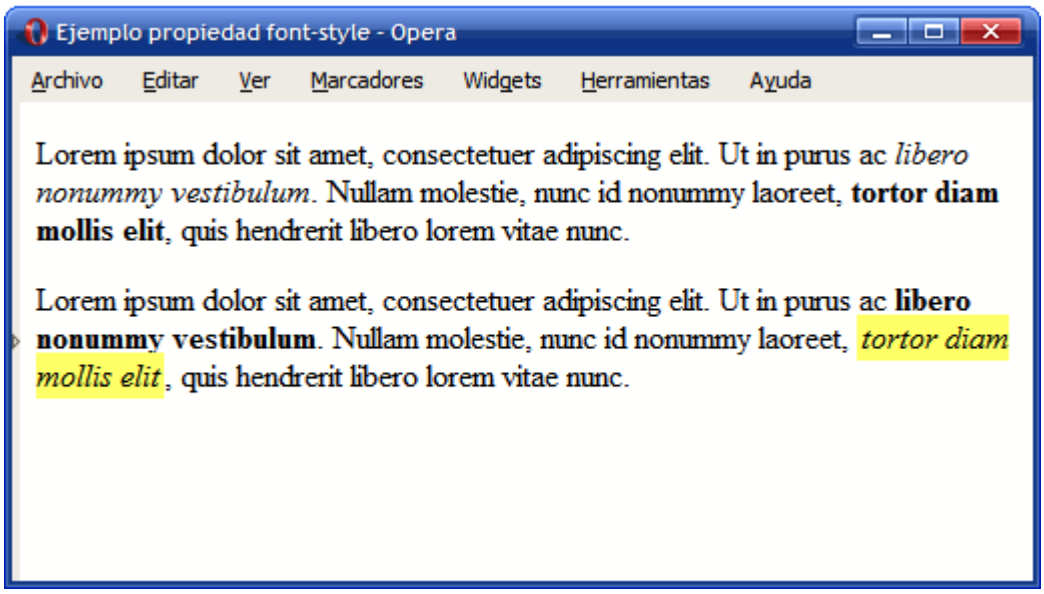

Figura 6.3. Ejemplo de propiedad font-style

Ahora, el texto del elemento `<em>` se muestra como un texto en negrita y el texto del elemento `<strong>` se muestra como un texto en cursiva con un color de fondo muy destacado.

El único cambio necesario en las reglas CSS anteriores es el de añadir la propiedad `font-style`:

```
#especial em {
  font-weight: bold;
  font-style: normal;
}
#especial strong {
  font-weight: normal;
  font-style: italic;
  background-color:#FFFF66;
  padding: 2px;
}
```

Por último, CSS permite otra variación en el estilo del tipo de letra, controlado mediante la propiedad `font-variant`.

Tabla 6.5. Propiedad font-variant

`font-variant`	**Estilo alternativo de la letra**
Valores	`normal | small-caps | inherit`
Se aplica a	Todos los elementos
Valor inicial	normal
Descripción	Establece el estilo alternativo de la letra utilizada para el texto

La propiedad `font-variant` no se suele emplear habitualmente, ya que sólo permite mostrar el texto con *letra versal* (mayúsculas pequeñas).

Siguiendo con el ejemplo anterior, se ha aplicado la propiedad `font-variant: small-caps` al segundo párrafo de texto:

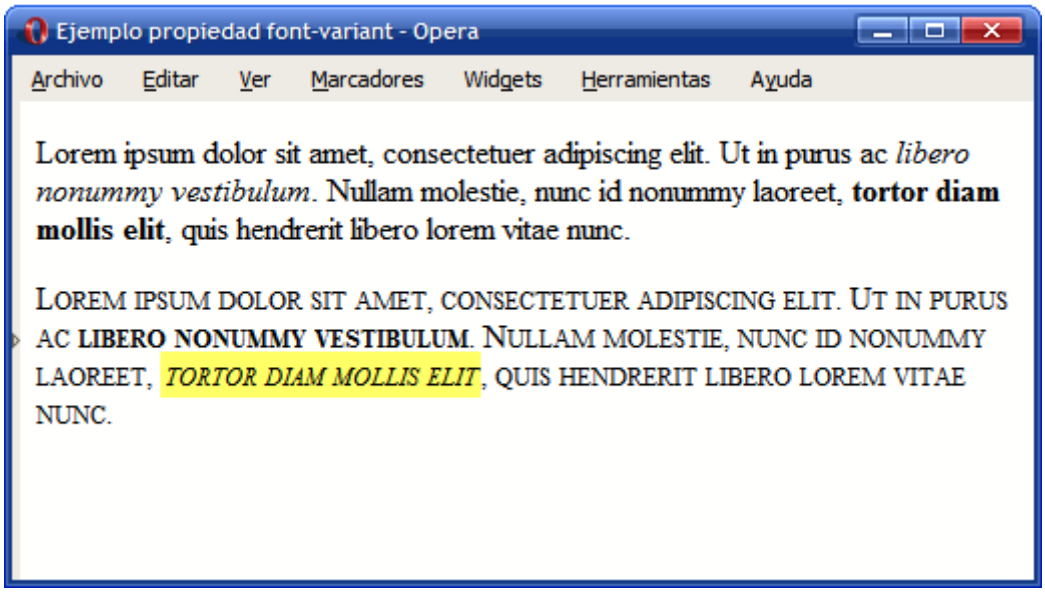

Figura 6.4. Ejemplo de propiedad font-variant

Para este último ejemplo, solamente es necesario añadir una regla a los estilos CSS:

```
#especial {
  font-variant: small-caps;
}
```

Por otra parte, CSS proporciona una propiedad tipo *"shorthand"* denominada `font` y que permite indicar de forma directa todas las propiedades de la tipografía de un texto.

Tabla 6.6. Propiedad font

font	Tipografía
Valores	((<font-style> || <font-variant> || <font-weight>)? <font-size> (/ <line-height>)? <font-family>) | caption | icon | menu | message-box | small-caption | status-bar | inherit
Se aplica a	Todos los elementos
Valor inicial	-
Descripción	Permite indicar de forma directa todas las propiedades de la tipografía de un texto

El orden en el que se deben indicar las propiedades del texto es el siguiente:

- En primer lugar y de forma opcional se indican el `font-style`, `font-variant` y `font-weight` en cualquier orden.
- A continuación, se indica el valor de `font-size` seguido opcionalmente por el valor de `line-height`.
- Por último, se indica el tipo de letra a utilizar.

Ejemplos de uso de la propiedad `font`:

```
font: 76%/140% Verdana,Arial,Helvetica,sans-serif;
font: normal 24px/26px "Century Gothic","Trebuchet MS",Arial,Helvetica,sans-serif;
font: normal .94em "Trebuchet MS",Arial,Helvetica,sans-serif;
font: bold 1em "Trebuchet MS",Arial,Sans-Serif;
font: normal 0.9em "Lucida Grande", Verdana, Arial, Helvetica, sans-serif;
font: normal 1.2em/1em helvetica, arial, sans-serif;
font: 11px verdana, sans-serif;
font: normal 1.4em/1.6em "helvetica", arial, sans-serif;
font: bold 14px georgia, times, serif;
```

Aunque su uso no es muy común, la propiedad `font` permite indicar el tipo de letra a utilizar mediante una serie de palabras clave: `caption`, `icon`, `menu`, `message-box`, `small-caption`, `status-bar`.

Si por ejemplo se utiliza la palabra `status-bar`, el navegador mostrará el texto con el mismo tipo de letra que la que utiliza el sistema operativo para mostrar los textos de la barra de estado de las ventanas. La palabra `icon` se utilizaría para mostrar el texto con el mismo tipo de letra que utiliza el sistema operativo para mostrar el nombre de los iconos y así sucesivamente.

Ejercicio 7 Ver enunciado en la página 187

6.2. Texto

Además de las propiedades relativas a la tipografía del texto, CSS define numerosas propiedades que determinan la apariencia del texto en su conjunto. Estas propiedades adicionales permiten controlar al alineación del texto, el interlineado, la separación entre palabras, etc.

La propiedad que define la alineación del texto se denomina `text-align`.

Tabla 6.7. Propiedad text-align

`text-align`	**Alineación del texto**
Valores	left | right | center | justify | inherit
Se aplica a	Elementos de bloque y celdas de tabla
Valor inicial	left
Descripción	Establece la alineación del contenido del elemento

Los valores definidos por CSS permiten alinear el texto según los valores tradicionales: a la izquierda (`left`), a la derecha (`right`), centrado (`center`) y justificado (`justify`).

La siguiente imagen muestra el efecto de establecer el valor `left`, `right`, `center` y `justify` respectivamente a cada uno de los párrafos de la página.

Figura 6.5. Ejemplo de propiedad text-align

La propiedad `text-align` no sólo alinea el texto que contiene un elemento, sino que también alinea todos sus contenidos, como por ejemplo las imágenes.

El interlineado de un texto se controla mediante la propiedad `line-height`, que permite controlar la altura ocupada por cada línea de texto:

Tabla 6.8. Propiedad line-height

`line-height`	**Interlineado**
Valores	normal | <numero> | <medida> | <porcentaje> | inherit
Se aplica a	Todos los elementos
Valor inicial	normal
Descripción	Permite establecer la altura de línea de los elementos

Además de todas las unidades de medida y el uso de porcentajes, la propiedad `line-height` permite indicar un número sin unidades que se interpreta como el múltiplo del tamaño normal de la letra. Por tanto, estas tres reglas CSS son equivalentes:

```
p { line-height: 1.2; font-size: 1em }
p { line-height: 1.2em; font-size: 1em }
p { line-height: 120%; font-size: 1em }
```

Siempre que se utilice de forma moderada, el interlineado mejora notablemente la legibilidad de un texto, como se puede observar en la siguiente imagen:

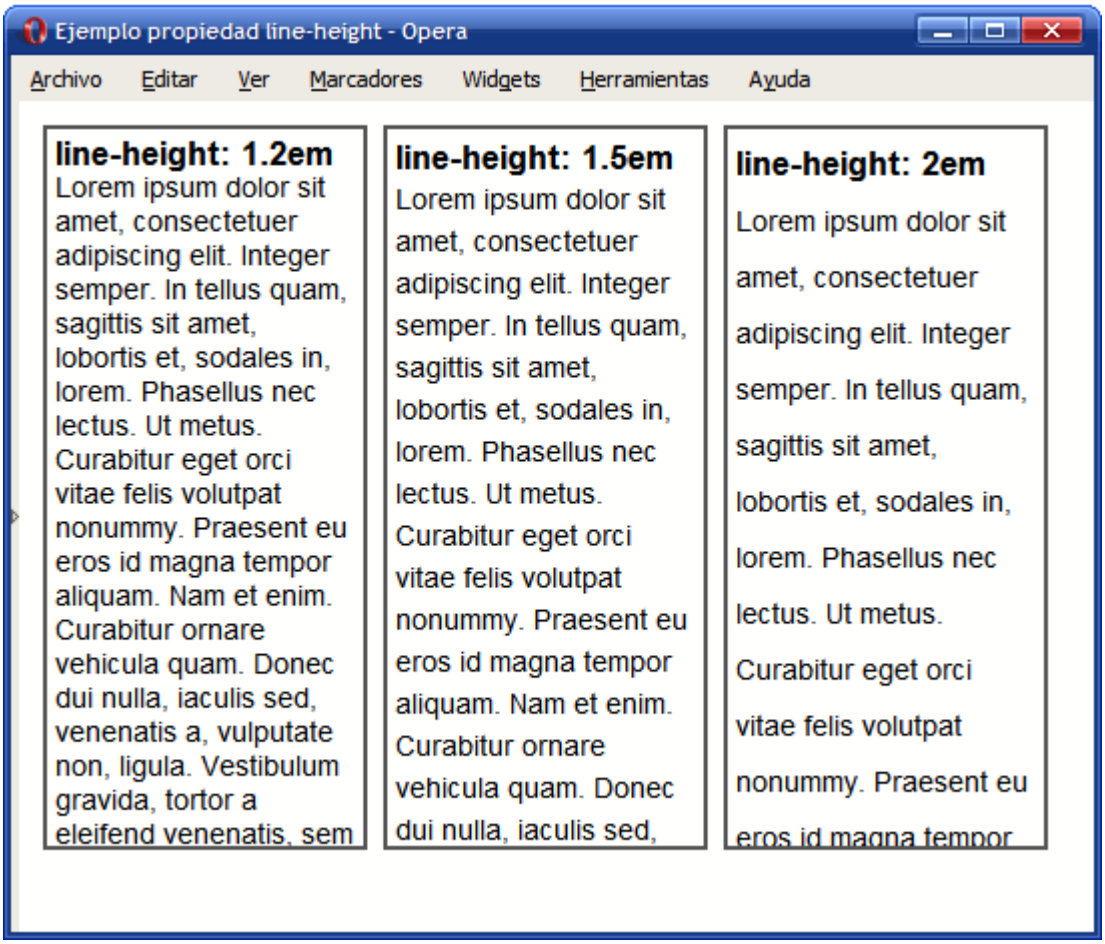

Figura 6.6. Ejemplo de propiedad line-height

Además de la decoración que se puede aplicar a la tipografía que utilizan los textos, CSS define otros estilos y decoraciones para el texto en su conjunto. La propiedad que decora el texto se denomina `text-decoration`.

Tabla 6.9. Propiedad text-decoration

`text-decoration`	**Decoración del texto**
Valores	`none | ( underline || overline || line-through || blink ) | inherit`
Se aplica a	Todos los elementos

text-decoration	Decoración del texto
Valor inicial	none
Descripción	Establece la decoración del texto (subrayado, tachado, parpadeante, etc.)

El valor `underline` subraya el texto, por lo que puede confundir a los usuarios haciéndoles creer que se trata de un enlace. El valor `overline` añade una línea en la parte superior del texto, un aspecto que raramente es deseable. El valor `line-through` muestra el texto tachado con una línea continua, por lo que su uso tampoco es muy habitual. Por último, el valor `blink` muestra el texto parpadeante y se recomienda evitar su uso por las molestias que genera a la mayoría de usuarios.

Una de las propiedades de CSS más desconocidas y que puede ser de gran utilidad en algunas circunstancias es la propiedad `text-transform`, que puede variar de forma sustancial el aspecto del texto.

Tabla 6.10. Propiedad text-transform

text-transform	Transformación del texto
Valores	`capitalize` | `uppercase` | `lowercase` | `none` | `inherit`
Se aplica a	Todos los elementos
Valor inicial	none
Descripción	Transforma el texto original (lo transforma a mayúsculas, a minúsculas, etc.)

La propiedad `text-transform` permite mostrar el texto original transformado en un texto completamente en mayúsculas (`uppercase`), en minúsculas (`lowercase`) o con la primera letra de cada palabra en mayúscula (`capitalize`).

La siguiente imagen muestra cada uno de los posibles valores:

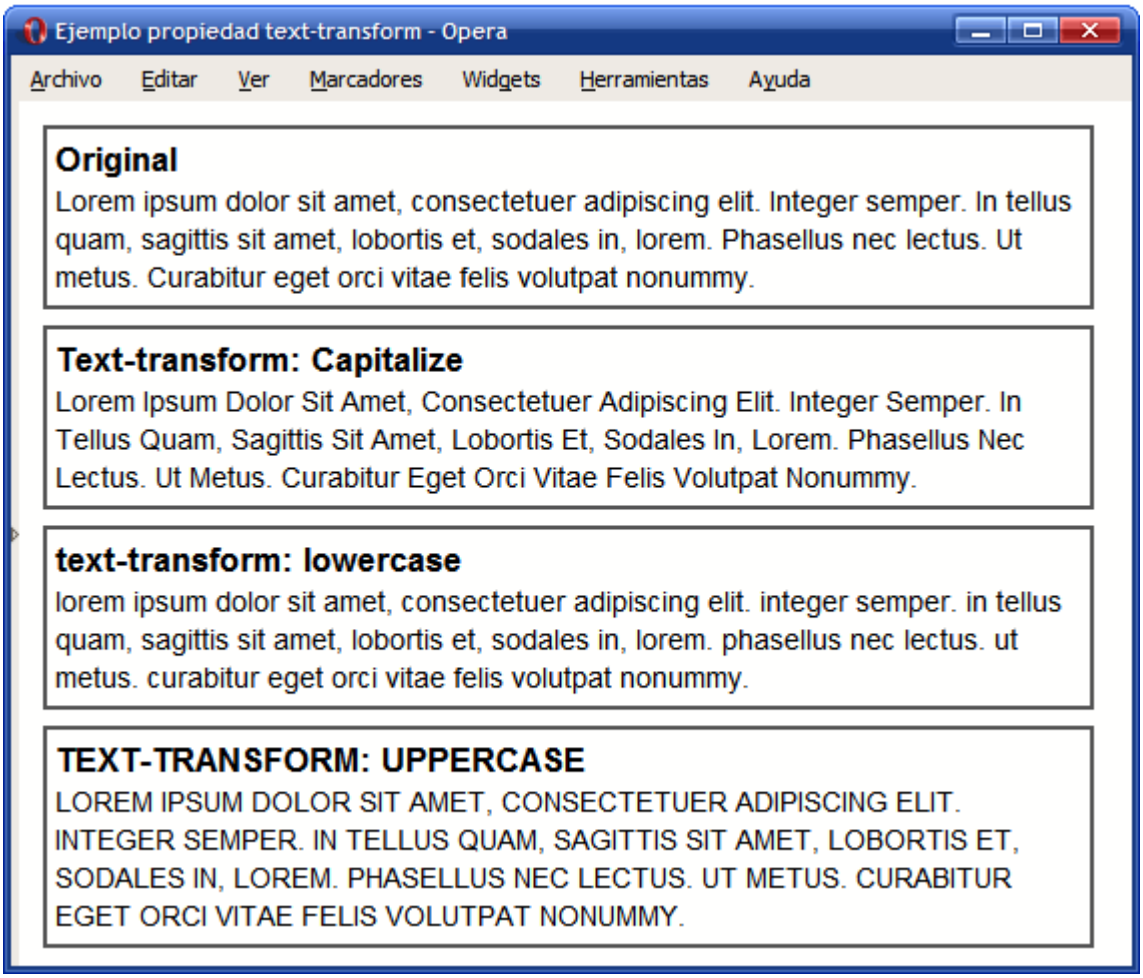

Figura 6.7. Ejemplo de propiedad text-transform

Las reglas CSS del ejemplo anterior se muestran a continuación:

```
<div style="text-transform: none"><h1>Original</h1>Lorem ipsum dolor
sit amet, consectetuer adipiscing elit. Integer semper. In tellus quam,
sagittis sit amet, lobortis et, sodales in, lorem. Phasellus nec lectus.
Ut metus. Curabitur eget orci vitae felis volutpat nonummy.</div>

<div style="text-transform: capitalize"><h1>text-transform: capitalize</h1>
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Integer semper.
In tellus quam, sagittis sit amet, lobortis et, sodales in, lorem. Phasellus
nec lectus. Ut metus. Curabitur eget orci vitae felis volutpat nonummy.</div>

<div style="text-transform: lowercase"><h1>text-transform: lowercase</h1>
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Integer semper.
In tellus quam, sagittis sit amet, lobortis et, sodales in, lorem. Phasellus
nec lectus. Ut metus. Curabitur eget orci vitae felis volutpat nonummy.</div>

<div style="text-transform: uppercase"><h1>text-transform: uppercase</h1>
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Integer semper.
In tellus quam, sagittis sit amet, lobortis et, sodales in, lorem. Phasellus
nec lectus. Ut metus. Curabitur eget orci vitae felis volutpat nonummy.</div>
```

Uno de los principales problemas del diseño de documentos y páginas mediante CSS consiste en la alineación vertical en una misma línea de varios elementos diferentes como imágenes y texto. Para controlar esta alineación, CSS define la propiedad `vertical-align`.

Tabla 6.11. Propiedad vertical-align

`vertical-align`	**Alineación vertical**
Valores	`baseline` | `sub` | `super` | `top` | `text-top` | `middle` | `bottom` | `text-bottom` | `<porcentaje>` | `<medida>` | `inherit`
Se aplica a	Elementos en línea y celdas de tabla
Valor inicial	baseline
Descripción	Determina la alineación vertical de los contenidos de un elemento

A continuación se muestra una imagen con el aspecto que muestran los navegadores para cada uno de los posibles valores de la propiedad `vertical-align`:

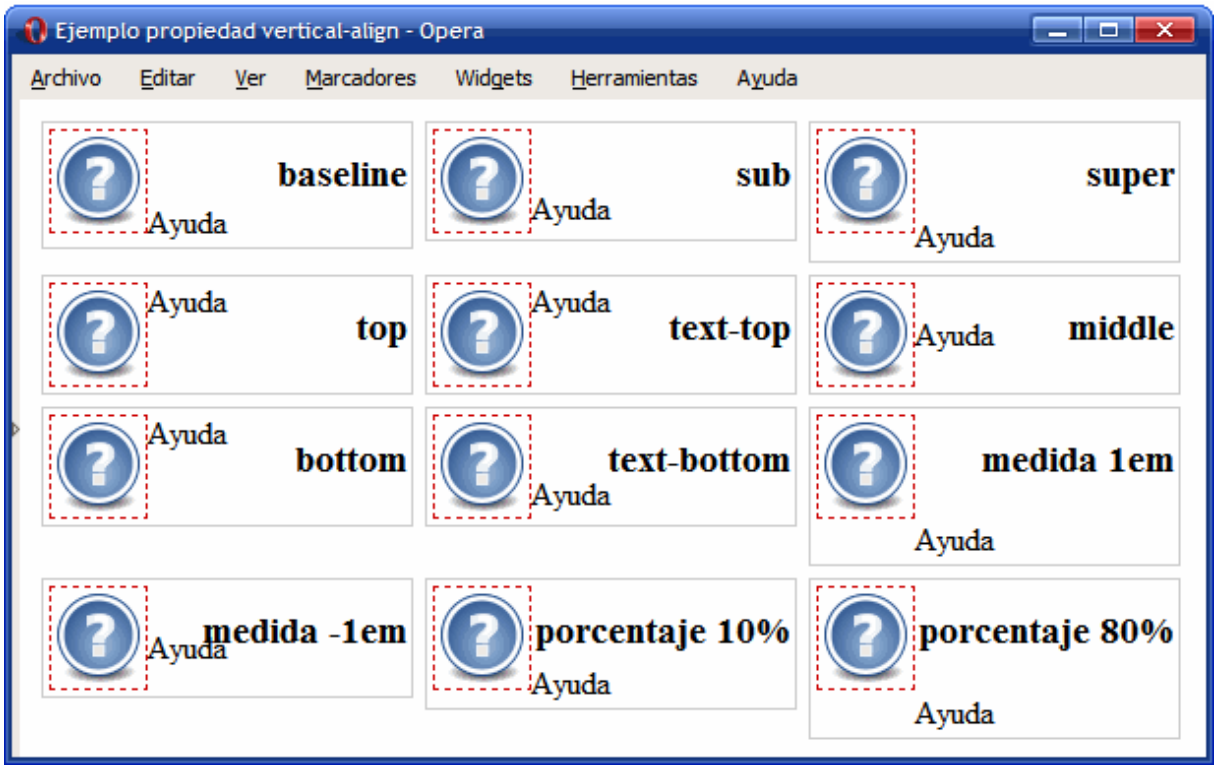

Figura 6.8. Ejemplo de propiedad vertical-align

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

```
img {border:1px dashed #CC0000; padding: 1px;}
div#baseline img { vertical-align:baseline; }
div#sub img { vertical-align:sub; }
div#super img { vertical-align:super; }
div#top img { vertical-align:top; }
div#text-top img { vertical-align:text-top; }
```

```
div#middle img { vertical-align:middle; }
div#bottom img { vertical-align:bottom; }
div#text-bottom img { vertical-align:text-bottom; }
div#medida_1em img { vertical-align:1em; }
div#medida_-1em img { vertical-align:-1em; }
div#porcentaje_10 img { vertical-align:10%; }
div#porcentaje_80 img { vertical-align:80%; }

<div id="baseline">
  <h1>baseline</h1>
  <img src="images/help.png" alt="Ayuda" />
  <span>Ayuda</span>
</div>

<div id="sub">
  <h1>sub</h1>
  <img src="images/help.png" alt="Ayuda" />
  <span>Ayuda</span>
</div>

<div id="super">
  <h1>super</h1>
  <img src="images/help.png" alt="Ayuda" />
  <span>Ayuda</span>
</div>

<div id="top">
  <h1>top</h1>
  <img src="images/help.png" alt="Ayuda" />
  <span>Ayuda</span>
</div>

<div id="text-top">
  <h1>text-top</h1>
  <img src="images/help.png" alt="Ayuda" />
  <span>Ayuda</span>
</div>

<div id="middle">
  <h1>middle</h1>
  <img src="images/help.png" alt="Ayuda" />
  <span>Ayuda</span>
</div>

<div id="bottom">
  <h1>bottom</h1>
  <img src="images/help.png" alt="Ayuda" />
  <span>Ayuda</span>
</div>

<div id="text-bottom">
  <h1>text-bottom</h1>
  <img src="images/help.png" alt="Ayuda" />
  <span>Ayuda</span>
</div>
```

```
<div id="medida_1em">
  <h1>medida 1em</h1>
  <img src="images/help.png" alt="Ayuda" />
  <span>Ayuda</span>
</div>

<div id="medida_-1em">
  <h1>medida -1em</h1>
  <img src="images/help.png" alt="Ayuda" />
  <span>Ayuda</span>
</div>

<div id="porcentaje_10">
  <h1>porcentaje 10%</h1>
  <img src="images/help.png" alt="Ayuda" />
  <span>Ayuda</span>
</div>

<div id="porcentaje_80">
  <h1>porcentaje 80%</h1>
  <img src="images/help.png" alt="Ayuda" />
  <span>Ayuda</span>
</div>
```

El valor por defecto es `baseline` y el valor más utilizado cuando se establece la propiedad `vertical-align` es `middle`.

En muchas publicaciones impresas suele ser habitual tabular la primera línea de cada párrafo para facilitar su lectura. CSS permite controlar esta tabulación mediante la propiedad `text-indent`.

Tabla 6.12. Propiedad text-indent

text-indent	**Tabulación del texto**
Valores	`<medida> | <porcentaje> | inherit`
Se aplica a	Todos los elementos
Valor inicial	0
Descripción	Tabula desde la izquierda la primera línea del texto original

La siguiente imagen muestra la comparación entre un texto largo formado por varios párrafos sin tabular y el mismo texto con la primera línea de cada párrafo tabulada:

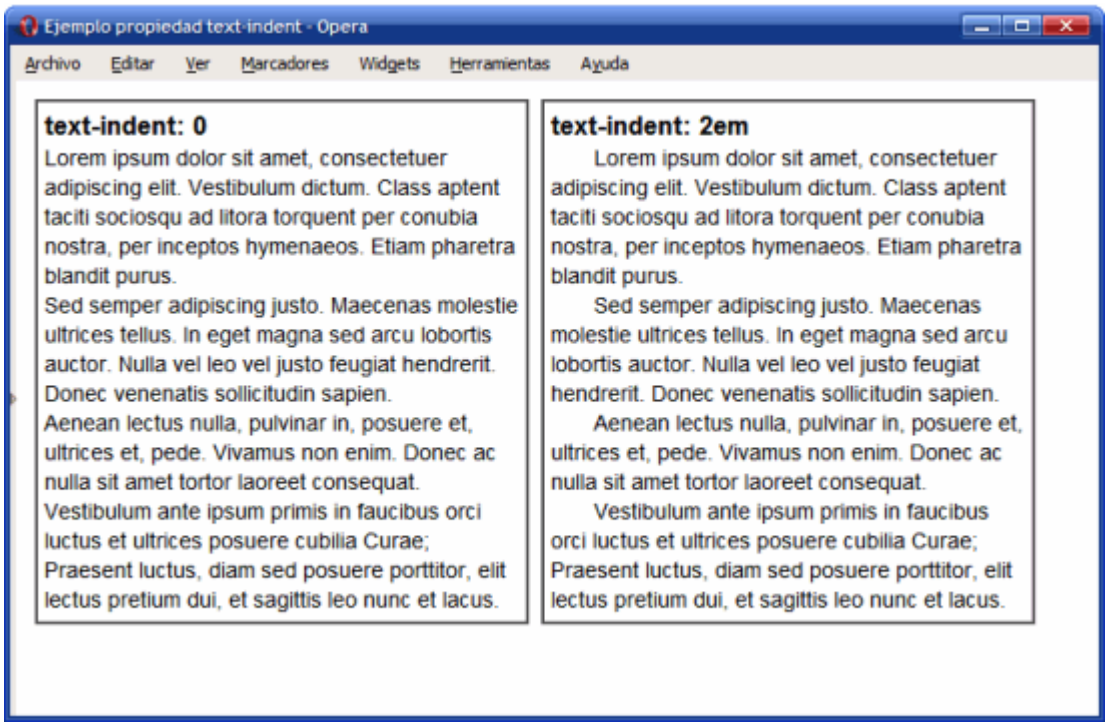

Figura 6.9. Ejemplo de propiedad text-indent

CSS también permite controlar la separación entre las letras que forman las palabras y la separación entre las palabras que forman los textos. La propiedad que controla la separación entre letras se llama `letter-spacing` y la separación entre palabras se controla mediante `word-spacing`.

Tabla 6.13. Propiedad letter-spacing

`letter-spacing`	**Espaciado entre letras**
Valores	`normal | <medida> | inherit`
Se aplica a	Todos los elementos
Valor inicial	normal
Descripción	Permite establecer el espacio entre las letras que forman las palabras del texto

Tabla 6.14. Propiedad word-spacing

`word-spacing`	**Espaciado entre palabras**
Valores	`normal | <medida> | inherit`
Se aplica a	Todos los elementos
Valor inicial	normal
Descripción	Permite establecer el espacio entre las palabras que forman el texto

La siguiente imagen muestra la comparación entre un texto normal y otro con las propiedades `letter-spacing` y `word-spacing` aplicadas:

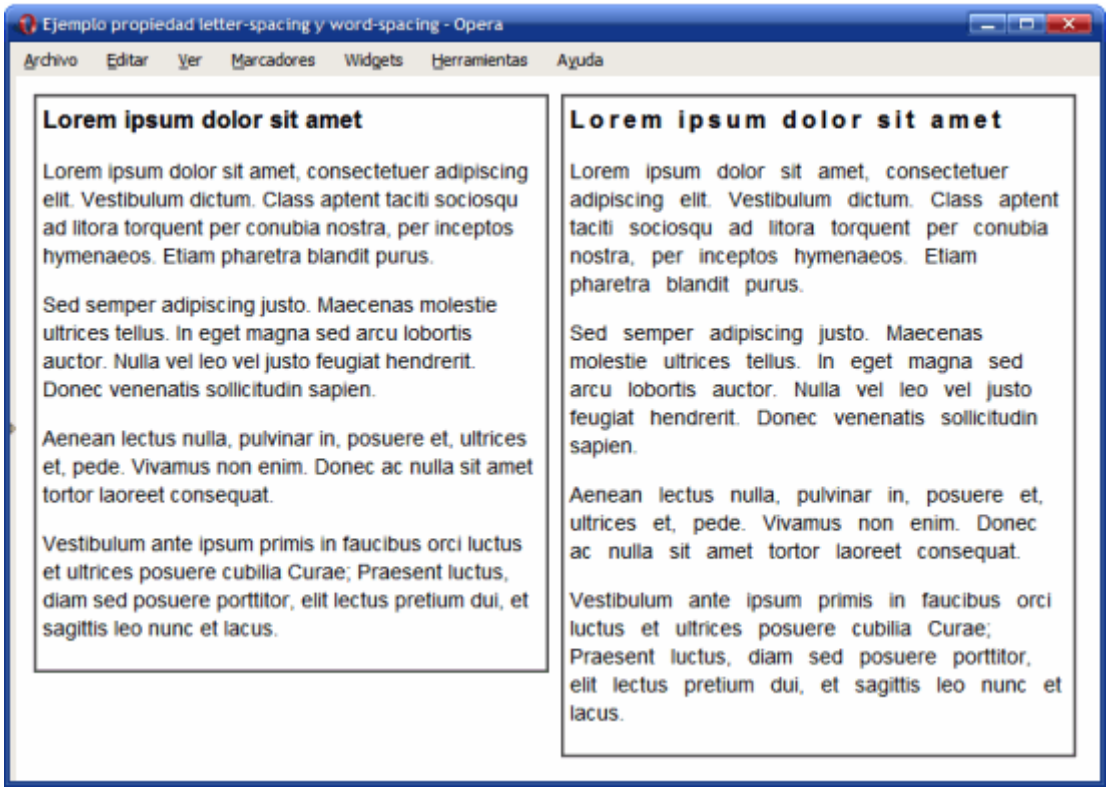

Figura 6.10. Ejemplo de propiedades letter-spacing y word-spacing

Las reglas CSS del ejemplo anterior se muestran a continuación:

```
.especial h1 { letter-spacing: .2em; }
.especial p { word-spacing: .5em; }

<div><h1>Lorem ipsum dolor sit amet</h1>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Vestibulum
dictum. Class aptent taciti sociosqu ad litora torquent per conubia nostra,
per inceptos hymenaeos. Etiam pharetra blandit purus.</p>
...
</div>

<div class="especial"><h1>Lorem ipsum dolor sit amet</h1>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Vestibulum
dictum. Class aptent taciti sociosqu ad litora torquent per conubia nostra,
per inceptos hymenaeos. Etiam pharetra blandit purus.</p>
...
</div>
```

Como ya se sabe. el tratamiento que hace HTML de los espacios en blanco es bastante problemático cuando se empiezan a crear las primeras páginas. Básicamente, HTML elimina todos los espacios en blanco sobrantes (es decir, todos salvo el espacio en blanco entre cada palabra) y todas las nuevas líneas.

Para forzar los espacios en blanco adicionales se debe utilizar la entidad HTML ` ` y para forzar nuevas líneas, se utiliza el elemento `<br/>`. Además, HTML proporciona el

elemento `<pre>` que muestra el contenido tal y como se escribe, respetando todos los espacios en blanco y todas las nuevas líneas.

CSS también permite controlar el tratamiento de los espacios en blanco de los textos mediante la propiedad `white-space`.

Tabla 6.15. Propiedad white-space

`white-space`	**Tratamiento de los espacios en blanco**
Valores	`normal | pre | nowrap | pre-wrap | pre-line | inherit`
Se aplica a	Todos los elementos
Valor inicial	normal
Descripción	Establece el tratamiento de los espacios en blanco del texto

El significado de cada uno de los valores es el siguiente:

- `normal`: comportamiento por defecto de HTML.
- `pre`: se respetan los espacios en blanco y las nuevas líneas (exactamente igual que el elemento `<pre>`). Si la línea es muy larga, *se sale* del espacio asignado para ese contenido.
- `nowrap`: elimina los espacios en blanco y las nuevas líneas. Si la línea es muy larga, *se sale* del espacio asignado para ese contenido.
- `pre-wrap`: se respetan los espacios en blanco y las nuevas líneas, pero ajustando cada línea al espacio asignado para ese contenido.
- `pre-line`: elimina los espacios en blanco y respeta las nuevas líneas, pero ajustando cada línea al espacio asignado para ese contenido.

La siguiente imagen muestra las diferencias entre los valores permitidos para `white-space`. El párrafo original contiene espacios en blanco y nuevas líneas y se ha limitado el espacio en el que se debería mostrar los contenidos:

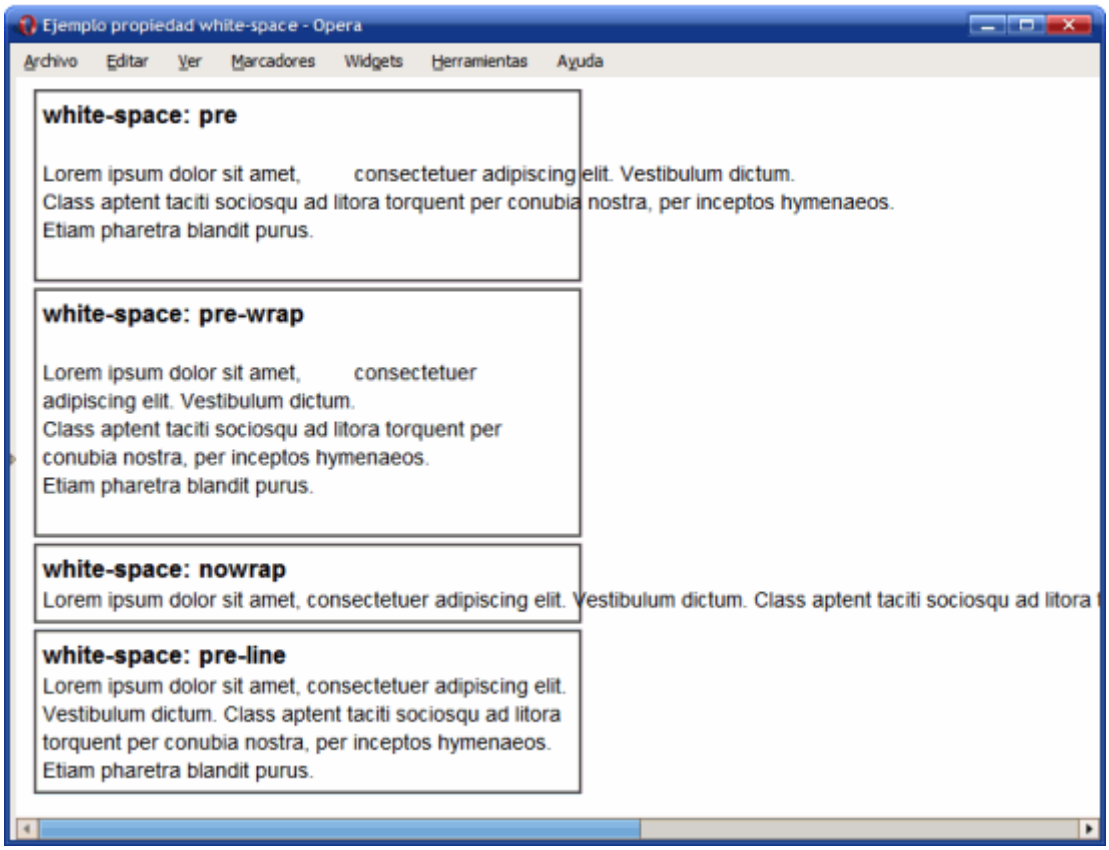

Figura 6.11. Ejemplo de propiedad white-space

Por último, CSS define unos elementos especiales llamados *"pseudo-elementos"* que permiten aplicar estilos a ciertas partes de un texto. En concreto, CSS permite definir estilos especiales a la primera frase de un texto y a la primera letra de un texto.

El pseudo-elemento `:first-line` permite aplicar estilos a la primera línea de un texto. Las palabras que forman la primera línea de un texto dependen del espacio reservado para mostrar el texto o del tamaño de la ventana del navegador, por lo que CSS calcula de forma automática las palabras que forman la primera línea de texto en cada momento.

La siguiente imagen muestra cómo aplica CSS los estilos indicados a la primera línea calculando para cada anchura las palabras que forman la primera línea:

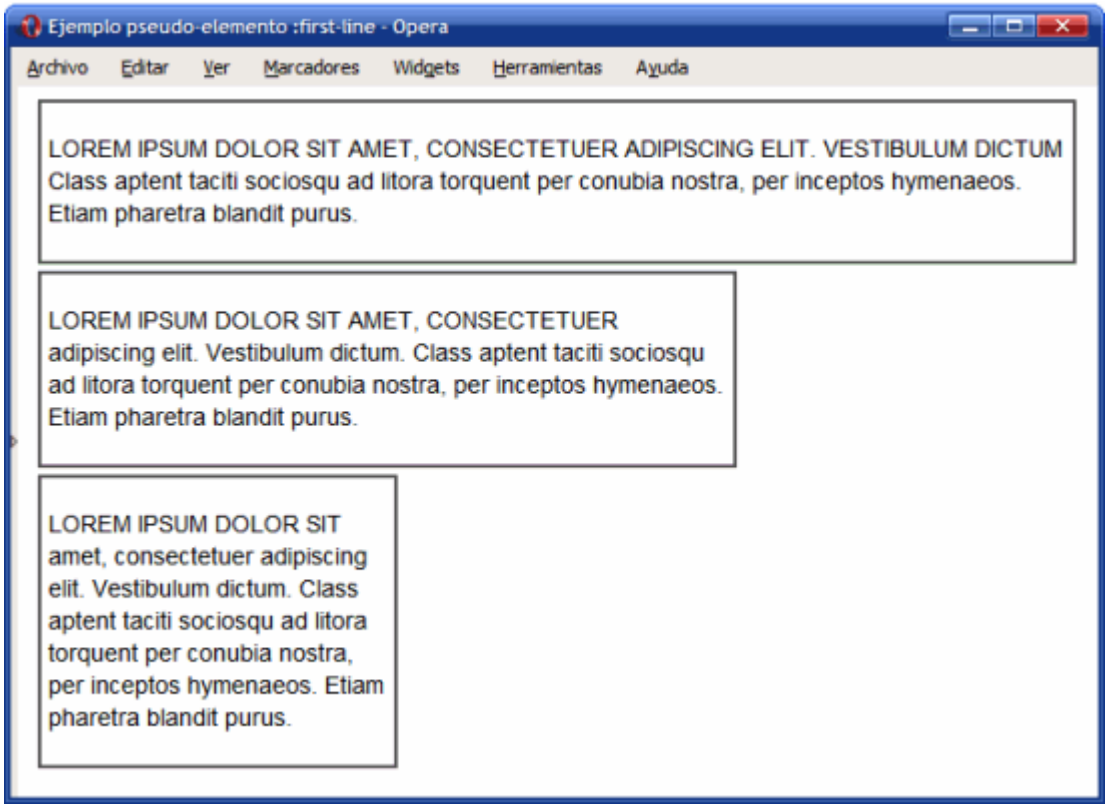

Figura 6.12. Ejemplo de pseudo-elemento first-line

La regla CSS utilizada para los párrafos del ejemplo se muestra a continuación:

```
p:first-line {
  text-transform: uppercase;
}
```

De la misma forma, CSS permite aplicar estilos a la primera letra del texto mediante el pseudo-elemento `:first-letter`. La siguiente imagen muestra el uso del pseudo-elemento `:first-letter` para crear una letra capital:

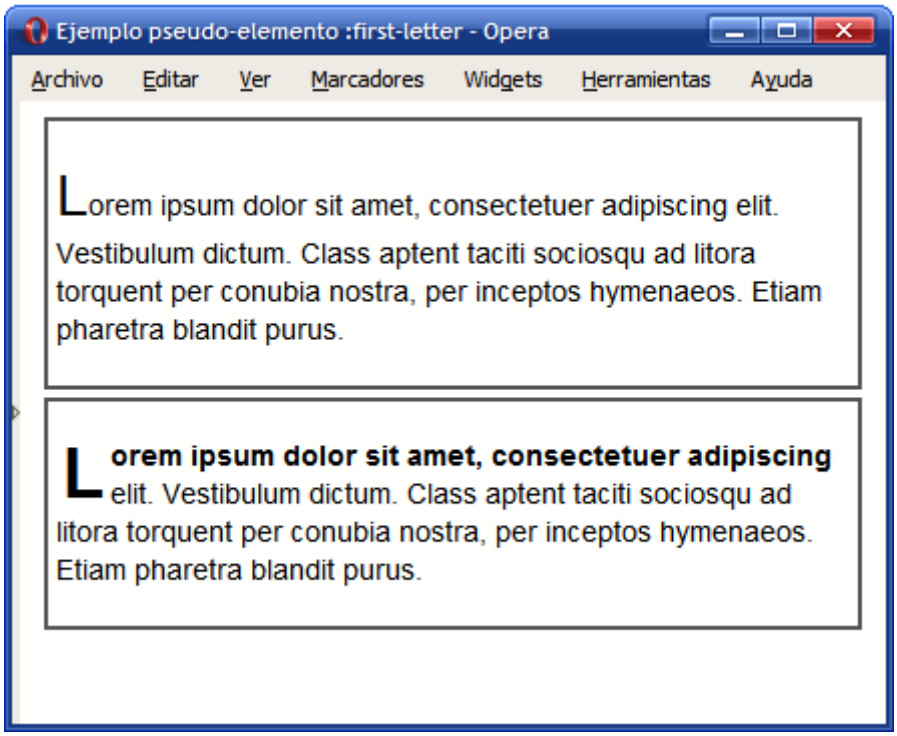

Figura 6.13. Ejemplo de pseudo-elemento first-letter

El código HTML y CSS se muestra a continuación:

```
#normal p:first-letter {
  font-size: 2em;
}
#avanzado p:first-letter {
  font-size: 2.5em;
  font-weight: bold;
  line-height: .9em;
  float: left;
  margin: .1em;
}
#avanzado p:first-line {
  font-weight: bold;
}

<div id="normal">
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Vestibulum
dictum. Class aptent taciti sociosqu ad litora torquent per conubia nostra,
per inceptos hymenaeos. Etiam pharetra blandit purus.</p>
</div>
<div id="avanzado">
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Vestibulum
dictum. Class aptent taciti sociosqu ad litora torquent per conubia nostra,
per inceptos hymenaeos. Etiam pharetra blandit purus.</p>
</div>
```

Capítulo 7. Enlaces

7.1. Estilos básicos

7.1.1. Tamaño, color y decoración

Los estilos más sencillos que se pueden aplicar a los enlaces son los que modifican su tamaño de letra, su color y la decoración del texto del enlace. Utilizando las propiedades `text-decoration` y `font-weight` se pueden conseguir estilos como los que se muestran en la siguiente imagen:

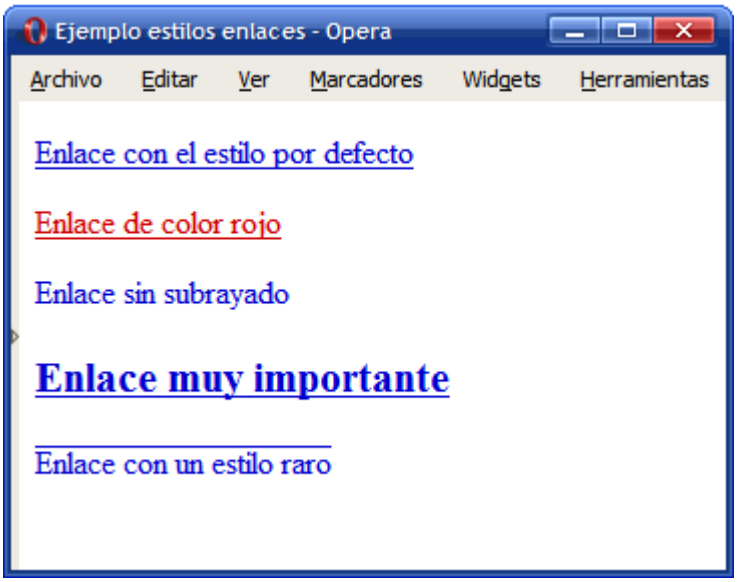

Figura 7.1. Ejemplo de enlaces con estilos diferentes

A continuación se muestran las reglas CSS del ejemplo anterior:

```
a { margin: 1em 0; display: block;}

.alternativo {color: #CC0000;}
.simple {text-decoration: none;}
.importante {font-weight: bold; font-size: 1.3em;}
.raro {text-decoration:overline;}

<a href="#">Enlace con el estilo por defecto</a>
<a class="alternativo" href="#">Enlace de color rojo</a>
<a class="simple" href="#">Enlace sin subrayado</a>
<a class="importante" href="#">Enlace muy importante</a>
<a class="raro" href="#">Enlace con un estilo raro</a>
```

7.1.2. Pseudo-clases

CSS define cuatro *pseudo-clases* que permiten aplicar estilos avanzados para los enlaces de los documentos. Las *pseudo-clases* permiten aplicar diferentes estilos a un mismo

enlace en función de su estado: enlace no visitado, enlace visitado, enlace en el que se pasa el puntero del ratón por encima y enlace activo en ese momento.

Cada una de las pseudo-clases definidas se muestra a continuación:

- `:link`, permite aplicar estilos para los enlaces que aún no han sido pinchados.
- `:visited`, aplica estilos a los enlaces que han sido pinchados anteriormente (el navegador del usuario elimina automáticamente el historial de enlaces visitados cada cierto tiempo).
- `:hover`, estilos que muestra el enlace cuando el usuario posiciona el puntero del ratón sobre el enlace.
- `:active`, estilos que se aplican al enlace cuando el usuario está pinchando sobre el enlace (el tiempo durante el que se aplica este estilo es muy breve).

El orden recomendado para la definición de las pseudo-clases de los enlaces es `:link`, `:visited`, `:hover`, `:active`.

Las pseudo-clases `:link` y `:visited` solamente están definidas para los enlaces, pero las pseudo-clases `:hover` y `:active` se definen para todos los elementos HTML. Desafortunadamente, Internet Explorer 6 y sus versiones anteriores solamente las soportan para los enlaces.

Las pseudo-clases de los enlaces permiten variar el comportamiento por defecto que los navegadores aplican a los enlaces. El siguiente ejemplo oculta el subrayado a los enlaces cuando el usuario pasa el ratón por encima de un enlace:

Figura 7.2. Ocultando el subrayado de los enlaces al pasar el ratón por encima

El código CSS necesario se muestra a continuación:

```
a { }

a:hover { text-decoration: none; }
```

Ejercicio 8 Ver enunciado en la página 188

7.2. Estilos avanzados

7.2.1. Decoración personalizada

La propiedad `text-decoration` permite añadir o eliminar el subrayado de los enlaces. Sin embargo, el aspecto del subrayado lo controla automáticamente el navegador, por lo que su color siempre es el mismo que el del texto del enlace y su anchura es proporcional al tamaño de letra.

No obstante, utilizando la propiedad `border-bottom` es posible añadir cualquier tipo de subrayado para los enlaces de las páginas. El siguiente ejemplo muestra algunos enlaces con el subrayado personalizado:

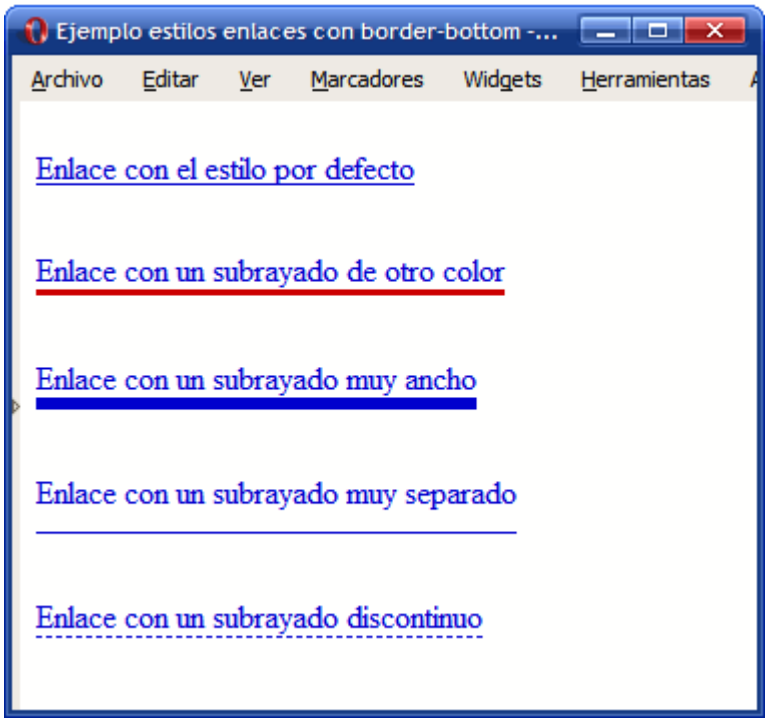

Figura 7.3. Enlaces con subrayado personalizado mediante la propiedad border

Las reglas CSS definidas en el ejemplo anterior se muestran a continuación:

```
a { margin: 1em 0; float: left; clear: left; text-decoration: none;}
.simple {text-decoration: underline;}
.color { border-bottom: medium solid #CC0000;}
.ancho {border-bottom: thick solid;}
.separado {border-bottom: 1px solid; padding-bottom: .6em;}
.discontinuo {border-bottom: thin dashed;}

<a class="simple" href="#">Enlace con el estilo por defecto</a>

<a class="color" href="#">Enlace un subrayado de otro color</a>

<a class="ancho" href="#">Enlace con un subrayado muy ancho</a>
```

```
<a class="separado" href="#">Enlace con un subrayado muy separado</a>

<a class="discontinuo" href="#">Enlace con un subrayado discontinuo</a>
```

7.2.2. Imágenes según el tipo de enlace

En ocasiones, puede resultar útil incluir un pequeño icono al lado de un enlace para indicar el tipo de contenido que enlaza, como por ejemplo un archivo comprimido o un documento con formato PDF.

Este tipo de imágenes son puramente decorativas en vez de imágenes de contenido, por lo que se deberían añadir con CSS y no con elementos de tipo `<img>`. Utilizando la propiedad `background` (y `background-image`) se puede personalizar el aspecto de los enlaces para que incluyan un pequeño icono a su lado.

La técnica consiste en mostrar una imagen de fondo sin repetición en el enlace y añadir el `padding` necesario al texto del enlace para que no se solape con la imagen de fondo.

El siguiente ejemplo personaliza el aspecto de dos enlaces añadiéndoles una imagen de fondo:

Figura 7.4. Personalizando el aspecto de los enlaces en función de su tipo

Las reglas CSS necesarias se muestran a continuación:

```
a { margin: 1em 0; float: left; clear: left; }

.rss {
  color: #E37529;
  padding: 0 0 0 18px;
  background: #FFF url(imagenes/rss.gif) no-repeat left center;
}

.pdf {
  padding: 0 0 0 22px;
  background: #FFF url(imagenes/pdf.png) no-repeat left center;
}

<a href="#">Enlace con el estilo por defecto</a>
```

```
<a class="rss" href="#">Enlace a un archivo RSS</a>

<a class="pdf" href="#">Enlace a un documento PDF</a>
```

7.2.3. Mostrar los enlaces como si fueran botones

Las propiedades definidas por CSS permiten mostrar los enlaces con el aspecto de un botón, lo que puede ser útil en formularios en los que no se utilizan elementos específicos para crear botones.

El siguiente ejemplo muestra un enlace simple formateado como un botón:

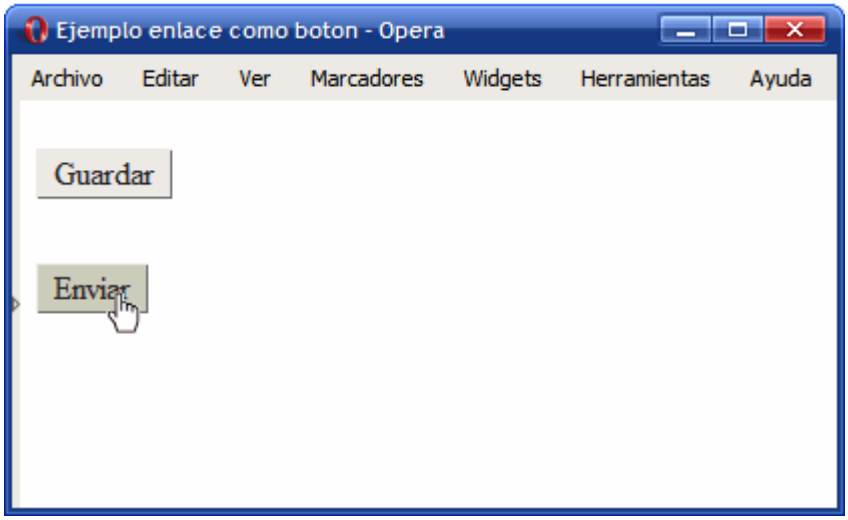

Figura 7.5. Mostrando un enlace como si fuera un botón

Las reglas CSS utilizadas en el ejemplo anterior son las siguientes:

```
a { margin: 1em 0; float: left; clear: left; }
a.boton {
  text-decoration: none;
  background: #EEE;
  color: #222;
  border: 1px outset #CCC;
  padding: .1em .5em;
}
a.boton:hover {
  background: #CCB;
}
a.boton:active {
  border: 1px solid #000;
}

<a class="boton" href="#">Guardar</a>
<a class="boton" href="#">Enviar</a>
```

Capítulo 8. Imágenes

8.1. Estilos básicos

8.1.1. Eliminar el borde de las imágenes con enlaces

Cuando una imagen forma parte de un enlace, los navegadores por defecto muestran las imágenes con un borde azul ancho. Por tanto, una de las reglas más utilizadas en los archivos CSS es la que elimina los bordes de las imágenes con enlaces:

```
img {
  border: none;
}
```

Ejercicio 9 Ver enunciado en la página 189

8.2. Estilos avanzados

8.2.1. Sombra (drop shadow)

La mayoría de aplicaciones de diseño gráfico permiten añadir una sombra (llamada *drop shadow* en inglés) a las imágenes. CSS no incluye propiedades que permitan mostrar de forma sencilla sombras en los elementos.

No obstante, existen varias técnicas sencillas y otras más avanzadas que permiten crear sombras que se adapten a cualquier imagen o elemento. A continuación se muestra una técnica sencilla para añadir sombra a las imágenes.

El estilo final de las sombras creadas con esta técnica se muestra a continuación:

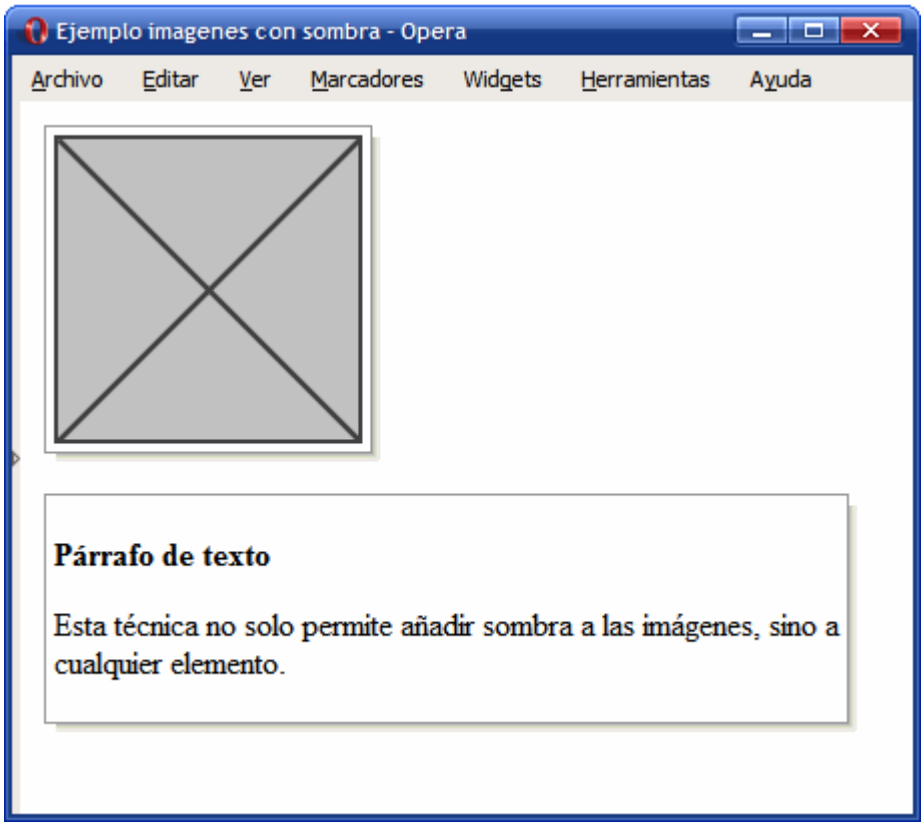

Figura 8.1. Sombra aplicada a las imágenes y otros elementos mediante CSS

La técnica mostrada se presentó por primera vez en la página web `http://wubbleyew.com/tests/dropshadows.htm` y consiste en la utilización de un par de elementos `<div>` alrededor del elemento que va a mostrar una sombra. La sombra se consigue mediante una imagen muy grande que contiene una sombra orientada hacia el lado derecho e inferior.

La ventaja de este método es que es sencillo y solamente requiere añadir un par de `<div>` por cada elemento. Además, como la sombra es una imagen muy grande, el efecto funciona con elementos de cualquier tamaño.

El mayor inconveniente de este método es que la sombra siempre se muestra en la misma dirección (derecha inferior) y que en Internet Explorer 6 y versiones anteriores sólo muestran la sombra correctamente cuando el color de fondo de la página es blanco (ya que Internet Explorer 6 y versiones anteriores no soportan imágenes en formato PNG con transparencias).

El código HTML y CSS del ejemplo anterior es bastante sencillo:

```
.dropshadow {
  float:left;
  clear:left;
  background: url(imagenes/shadowAlpha.png) no-repeat bottom right !important;
  background: url(imagenes/shadow.gif) no-repeat bottom right;
  margin: 10px 0 10px 10px !important;
  margin: 10px 0 10px 5px;
  padding: 0px;
}
.innerbox {
```

```
  position:relative;
  bottom:6px;
  right: 6px;
  border: 1px solid #999999;
  padding:4px;
  margin: 0px 0px 0px 0px;
}

<div class="dropshadow">
<div class="innerbox">
  <img src="imagenes/imagen.png" alt="Imagen genérica" />
</div>
</div>

<div class="dropshadow">
<div class="innerbox">
  <h4>Párrafo de texto</h4>
  <p>Esta técnica no sólo permite añadir sombra a las imágenes, sino a cualquier
elemento.</p>
</div>
</div>
```

Capítulo 9. Listas

9.1. Estilos básicos

9.1.1. Viñetas personalizadas

Por defecto, los navegadores muestran los elementos de las listas no ordenadas con una viñeta formada por un pequeño círculo de color negro. Los elementos de las listas ordenadas se muestran por defecto con la numeración decimal utilizada en la mayoría de países.

No obstante, CSS define varias propiedades para controlar el tipo de viñeta que muestran las listas, además de poder controlar la posición de la propia viñeta. La propiedad básica es la que controla el tipo de viñeta que se muestra y que se denomina `list-style-type`.

Tabla 9.1. Propiedad list-style-type

`list-style-type`	**Tipo de viñeta**
Valores	`disc` | `circle` | `square` | `decimal` | `decimal-leading-zero` | `lower-roman` | `upper-roman` | `lower-greek` | `lower-latin` | `upper-latin` | `armenian` | `georgian` | `lower-alpha` | `upper-alpha` | `none` | `inherit`
Se aplica a	Elementos de una lista
Valor inicial	disc
Descripción	Permite establecer el tipo de viñeta mostrada para una lista

En primer lugar, el valor `none` permite mostrar una lista en la que sus elementos no contienen viñetas, números o letras. Se trata de un valor muy utilizado, ya que es imprescindible para los menús de navegación creados con listas, como se verá más adelante.

El resto de valores de la propiedad `list-style-type` se dividen en tres tipos: gráficos, numéricos y alfabéticos.

- Los valores gráficos son `disc`, `circle` y `square` y muestran como viñeta un círculo relleno, un círculo vacío y un cuadrado relleno respectivamente.
- Los valores numéricos están formados por `decimal`, `decimal-leading-zero`, `lower-roman`, `upper-roman`, `armenian` y `georgian`.
- Por último, los valores alfanuméricos se controlan mediante `lower-latin`, `lower-alpha`, `upper-latin`, `upper-alpha` y `lower-greek`.

La siguiente imagen muestra algunos de los valores definidos por la propiedad `list-style-type`:

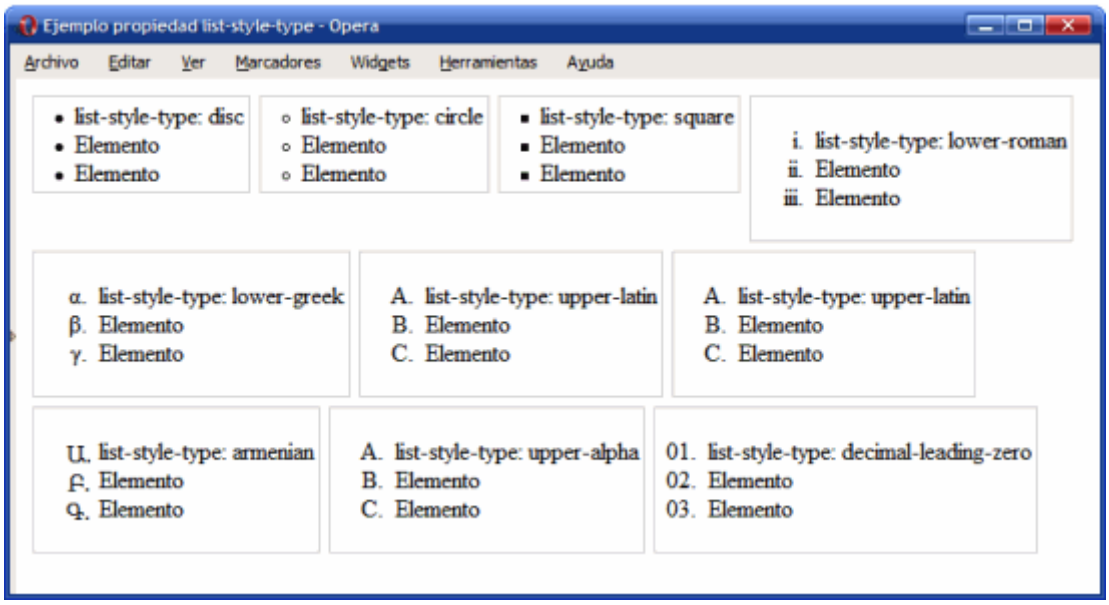

Figura 9.1. Ejemplo de propiedad list-style-type

El código CSS de algunas de las listas del ejemplo anterior se muestra a continuación:

```
<ul style="list-style-type: square">
  <li>list-style-type: square</li>
  <li>Elemento</li>
  <li>Elemento</li>
</ul>
<ol style="list-style-type: lower-roman">
  <li>list-style-type: lower-roman</li>
  <li>Elemento</li>
  <li>Elemento</li>
</ol>
<ol style="list-style-type: decimal-leading-zero; padding-left: 2em;">
  <li>list-style-type: decimal-leading-zero</li>
  <li>Elemento</li>
  <li>Elemento</li>
</ol>
```

La propiedad `list-style-position` permite controlar la colocación de las viñetas.

Tabla 9.2. Propiedad list-style-position

`list-style-position`	**Posición de la viñeta**
Valores	`inside` | `outside` | `inherit`
Se aplica a	Elementos de una lista
Valor inicial	outside
Descripción	Permite establecer la posición de la viñeta de cada elemento de una lista

La diferencia entre los valores `outside` y `inside` se hace evidente cuando los elementos contienen mucho texto, como en la siguiente imagen:

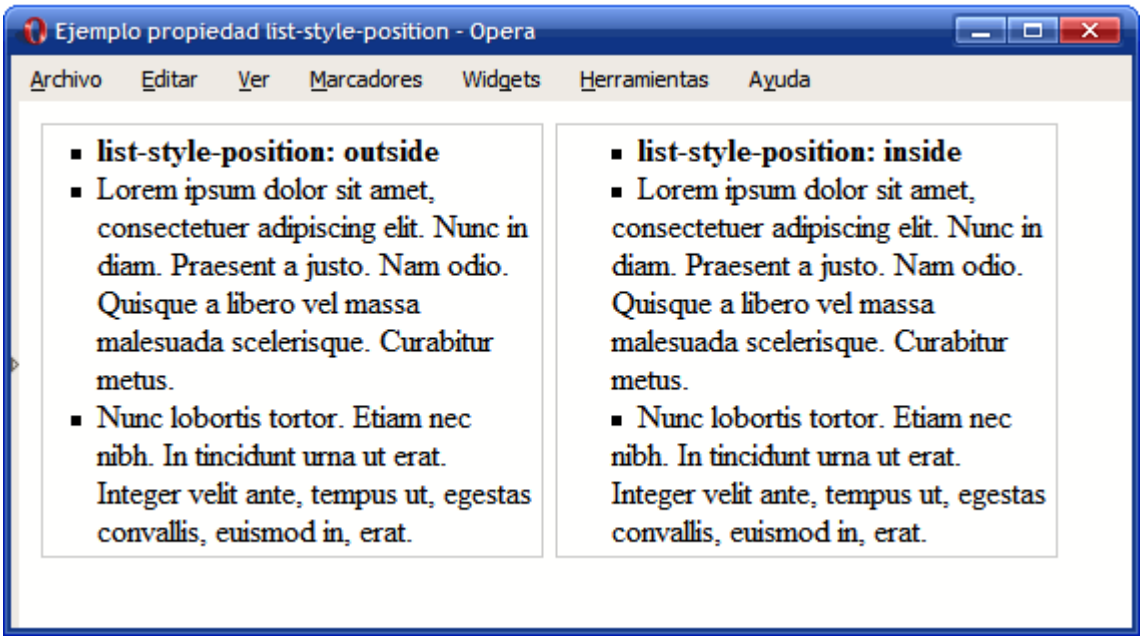

Figura 9.2. Ejemplo de propiedad list-style-position

Utilizando las propiedades anteriores (`list-style-type` y `list-style-position`), se puede seleccionar el tipo de viñeta y su posición, pero no es posible personalizar algunas de sus características básicas como su color y tamaño.

Cuando se requiere personalizar el aspecto de las viñetas, se debe emplear la propiedad `list-style-image`, que permite mostrar una imagen propia en vez de una viñeta automática.

Tabla 9.3. Propiedad list-style-image

`list-style-image`	**Imagen de la viñeta**
Valores	`<url>` | `none` | `inherit`
Se aplica a	Elementos de una lista
Valor inicial	none
Descripción	Permite reemplazar las viñetas automáticas por una imagen personalizada

Las imágenes personalizadas se indican mediante la URL de la imagen. Si no se encuentra la imagen o no se puede cargar, se muestra la viñeta automática correspondiente (salvo que explícitamente se haya eliminado mediante la propiedad `list-style-type`).

La siguiente imagen muestra el uso de la propiedad `list-style-image` mediante tres ejemplos sencillos de listas con viñetas personalizadas:

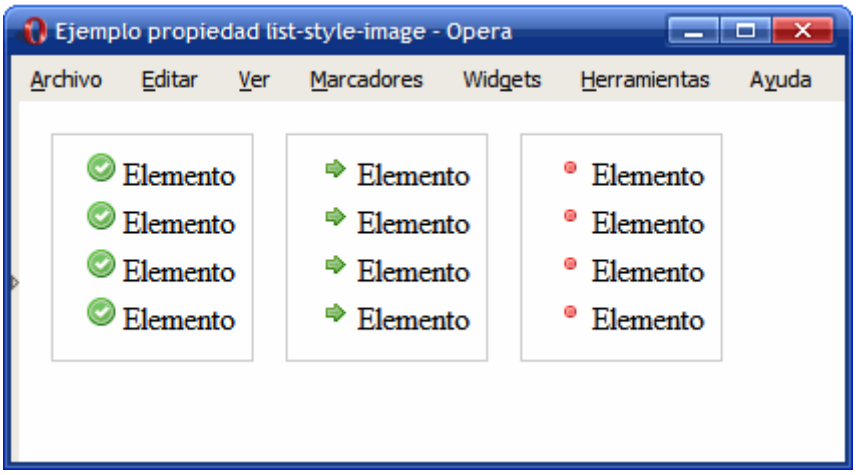

Figura 9.3. Ejemplo de propiedad list-style-image

Las reglas CSS correspondientes al ejemplo anterior se muestran a continuación:

```
ul {
  margin:0;
  padding-left: 1.5em;
  line-height: 1.5em;
}
ul li { padding-left: .2em; }
ul.ok { list-style-image: url(imagenes/ok.png); }
ul.go { list-style-image: url(imagenes/bullet_go.png); }
ul.redondo { list-style-image: url(imagenes/bullet_red.png); }

<div>
<ul class="ok">
  <li>Elemento</li>
  <li>Elemento</li>
  <li>Elemento</li>
  <li>Elemento</li>
</ul>
</div>
<div>
<ul class="go">
  <li>Elemento</li>
  <li>Elemento</li>
  <li>Elemento</li>
  <li>Elemento</li>
</ul>
</div>
<div>
<ul class="redondo">
  <li>Elemento</li>
  <li>Elemento</li>
  <li>Elemento</li>
  <li>Elemento</li>
</ul>
</div>
```

Como es habitual, CSS define una propiedad de tipo *"shorthand"* que permite establecer todas las propiedades de una lista de forma directa. La propiedad se denomina `list-style`.

Tabla 9.4. Propiedad list-style

`list-style`	**Estilo de una lista**
Valores	`( <list-style-type> || <list-style-position> || <list-style-image> ) | inherit`
Se aplica a	Elementos de una lista
Valor inicial	-
Descripción	Propiedad que permite establecer de forma simultánea todas las opciones de una lista

En la definición anterior, la notación || significa que el orden en el que se indican los valores de la propiedad es indiferente. El siguiente ejemplo indica que no se debe mostrar ni viñetas automáticas ni viñetas personalizadas:

```
ul { list-style: none }
```

Cuando se utiliza una viñeta personalizada, es conveniente indicar la viñeta automática que se mostrará cuando no se pueda cargar la imagen:

```
ul { list-style: url(imagenes/cuadrado_rojo.gif) square; }
```

9.1.2. Menú vertical sencillo

Las listas HTML se suelen emplear, además de para su función natural, para la creación de menús de navegación verticales y horizontales.

A continuación se muestra la transformación de una lista sencilla de enlaces en un menú vertical de navegación.

Lista de enlaces original:

```
<ul>
  <li><a href="#" title="Enlace genérico">Elemento 1</a></li>
  <li><a href="#" title="Enlace genérico">Elemento 2</a></li>
  <li><a href="#" title="Enlace genérico">Elemento 3</a></li>
  <li><a href="#" title="Enlace genérico">Elemento 4</a></li>
  <li><a href="#" title="Enlace genérico">Elemento 5</a></li>
  <li><a href="#" title="Enlace genérico">Elemento 6</a></li>
</ul>
```

Aspecto final del menú vertical:

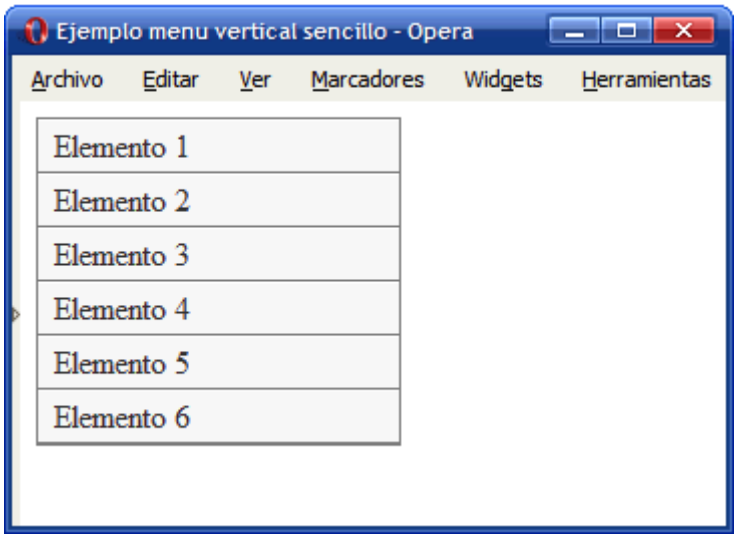

Figura 9.4. Menú vertical sencillo creado con CSS

El proceso de transformación de la lista en un menú requiere de los siguientes pasos:

1) Definir la anchura del menú:

```
ul.menu { width:180px; }
```

Figura 9.5. Menú vertical: definiendo su anchura

2) Eliminar las viñetas automáticas y todos los márgenes y espaciados aplicados por defecto:

```
ul.menu {
  width: 180px;
  list-style: none;
  margin: 0;
  padding: 0;
}
```

Figura 9.6. Menú vertical: eliminar viñetas por defecto

3) Añadir un borde al menú de navegación y establecer el color de fondo y los bordes de cada elemento del menú:

```
ul.menu {
  width: 180px;
  list-style: none;
  margin: 0;
  padding: 0;
  border:1px solid #7C7C7C;
}
ul.menu li {
  border-bottom:1px solid #7C7C7C;
  border-top: 1px solid #FFF;
  background: #F4F4F4;
}
```

Figura 9.7. Menú vertical: añadiendo bordes

4) Aplicar estilos a los enlaces: mostrarlos como un elemento de bloque para que ocupen todo el espacio de cada `<li>` del menú, añadir un espacio de relleno y modificar los colores y la decoración por defecto:

```
ul.menu li a:link, ul.menu li a:visited {
  padding: .2em 0 .2em .5em;
  display: block;
  text-decoration: none;
  color: #333;
}
```

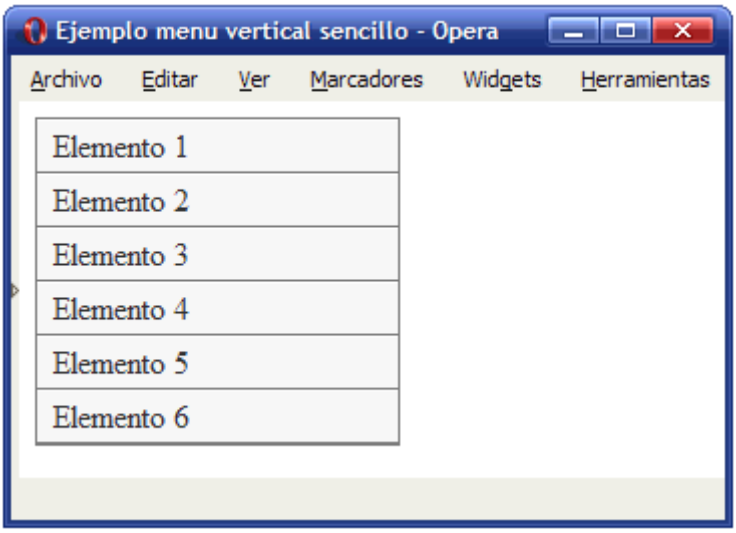

Figura 9.8. Aspecto final del menú vertical sencillo creado con CSS

Los tipos de menús verticales que se pueden definir mediante las propiedades CSS son innumerables, como se puede observar en la página `http://www.exploding-boy.com/images/EBmenus/menus.html`

9.1.3. Menú vertical avanzado

Ejercicio 10 Ver enunciado en la página 190

9.2. Estilos avanzados

9.2.1. Menú horizontal básico

A partir de un menú vertical sencillo, es posible crear un menú horizontal sencillo aplicando las propiedades CSS conocidas hasta el momento.

A continuación se muestra la transformación del anterior menú vertical sencillo en un menú horizontal sencillo. En este ejemplo, las propiedades para establecer el aspecto de los elementos del menú se definen en los elementos `<a>` en lugar de definirlas para los elementos `<li>` como en el ejemplo anterior. En cualquier caso, es indiferente el elemento en el que se aplican los estilos que definen el aspecto de cada opción del menú.

Código HTML del menú horizontal:

```
<ul>
  <li><a href="#" title="Enlace genérico">Elemento 1</a></li>
  <li><a href="#" title="Enlace genérico">Elemento 2</a></li>
  <li><a href="#" title="Enlace genérico">Elemento 3</a></li>
  <li><a href="#" title="Enlace genérico">Elemento 4</a></li>
  <li><a href="#" title="Enlace genérico">Elemento 5</a></li>
  <li><a href="#" title="Enlace genérico">Elemento 6</a></li>
</ul>
```

Código CSS del menú vertical anterior:

```
ul.menu {
  width: 180px;
  list-style: none;
  margin: 0;
  padding: 0;
  border: 1px solid #7C7C7C;
}
ul.menu li {
  border-bottom:1px solid #7C7C7C;
  border-top: 1px solid #FFF;
  background: #F4F4F4;
}
ul.menu li a:link, ul.menu li a:visited {
  padding: .2em 0 .2em .5em;
  display:block;
  text-decoration: none;
  color: #333;
}
```

Aspecto final del menú horizontal:

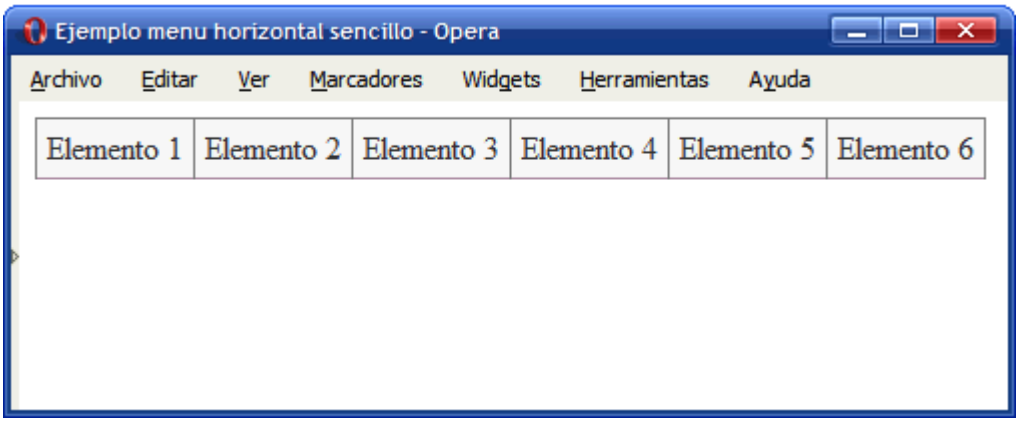

Figura 9.9. Menú horizontal sencillo creado con CSS

El proceso de transformación del menú vertical en un menú horizontal consta de los siguientes pasos:

1) Eliminar la anchura y el borde del elemento `<ul>` y aplicar las propiedades `float` y `clear`:

```
ul.menu {
  clear: both;
  float: left;
```

```
  width: 100%;
  list-style: none;
  margin: 0;
  padding: 0;
}
```

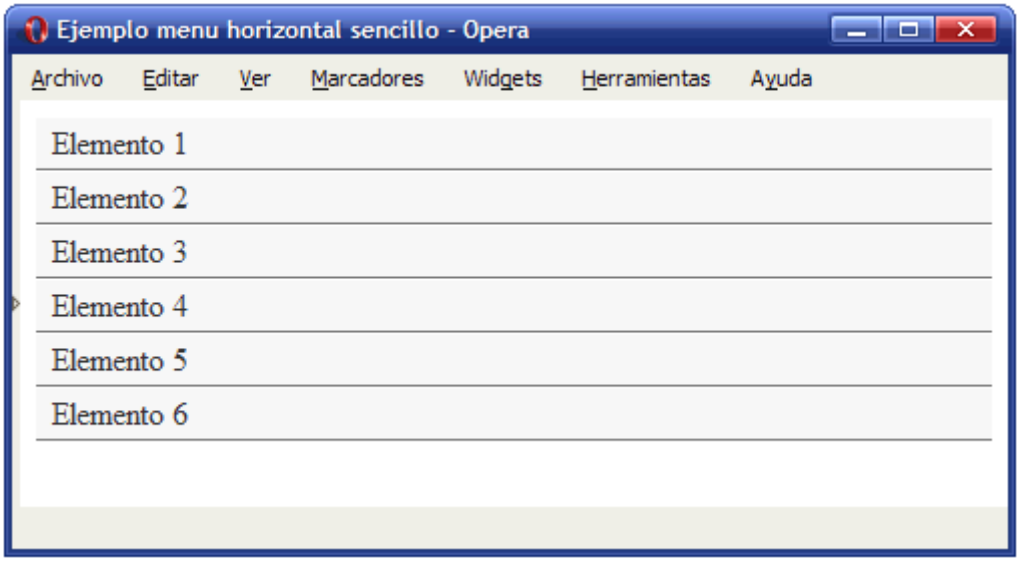

Figura 9.10. Menú horizontal: definiendo anchura y bordes

2) La clave de la transformación reside en modificar las propiedades `float` y `display` de los elementos `<li>` del menú:

```
ul.menu li {
  display: inline;
  float: left;
}
```

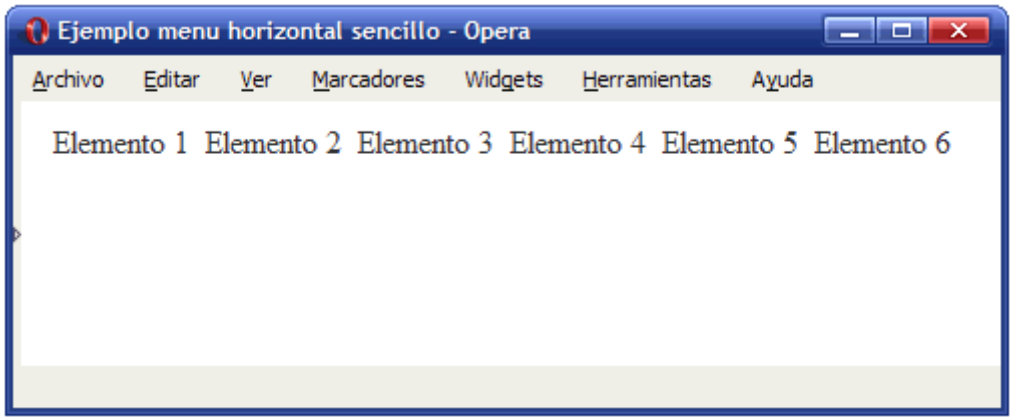

Figura 9.11. Menú horizontal: propiedades float y display

3) Modificar el `padding` y los bordes de los enlaces que forman el menú:

```
ul.menu li a:link, ul.menu li a:visited {
  padding: .3em;
  display: block;
  text-decoration: none;
  color: #333;
```

```
  background: #F4F4F4;
  border-top: 1px solid #7C7C7C;
  border-right: 1px solid #7C7C7C;
  border-bottom: 1px solid #9C9C9C;
}
```

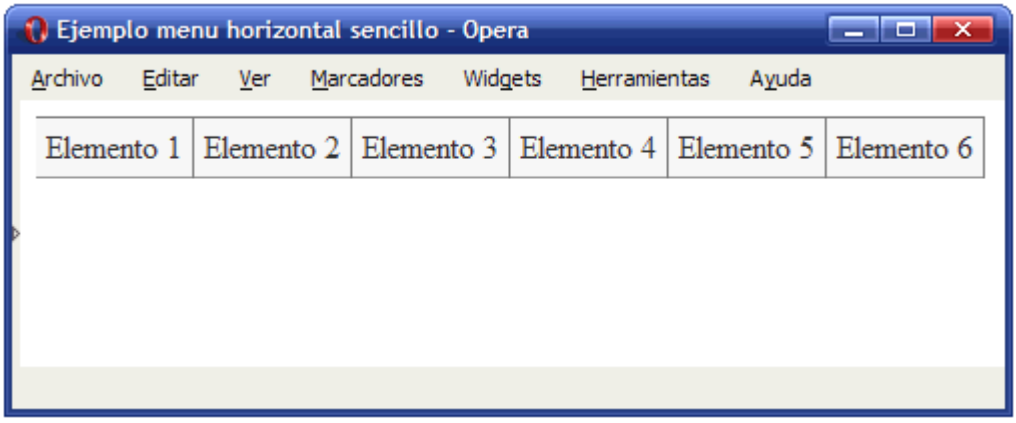

Figura 9.12. Menú horizontal: relleno y borde de los elementos

4) Por último, se añade un borde izquierdo en el elemento `<ul>` para homogeneizar el aspecto de los elementos del menú:

```
ul.menu {
  clear: both;
  float: left;
  width: 100%;
  list-style: none;
  margin: 0;
  padding: 0;
  border-left: 1px solid #7C7C7C;
}
```

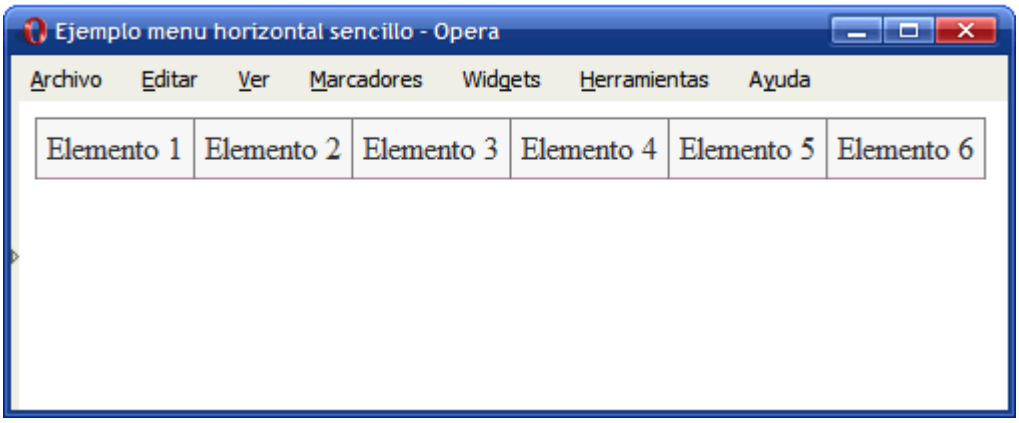

Figura 9.13. Aspecto final del menú horizontal sencillo creado con CSS

El código CSS final se muestra a continuación:

```
ul.menu {
  clear: both;
  float: left;
  width: 100%;
```

```
  list-style: none;
  margin: 0;
  padding: 0;
  border-left:1px solid #7C7C7C;
}
ul.menu li {
  display: inline;
  float: left;
}
ul.menu li a:link, ul.menu li a:visited {
  padding: .3em;
  display: block;
  text-decoration: none;
  color: #333;
  background: #F4F4F4;
  border-top:1px solid #7C7C7C;
  border-right:1px solid #7C7C7C;
  border-bottom:1px solid #9C9C9C;
}
```

9.2.2. Menú horizontal con solapas

Modificando los estilos de cada elemento del menú y utilizando imágenes de fondo y las pseudo-clases `:hover` y `:active`, se pueden crear menús horizontales complejos, incluso con el aspecto de un menú de solapas o pestañas:

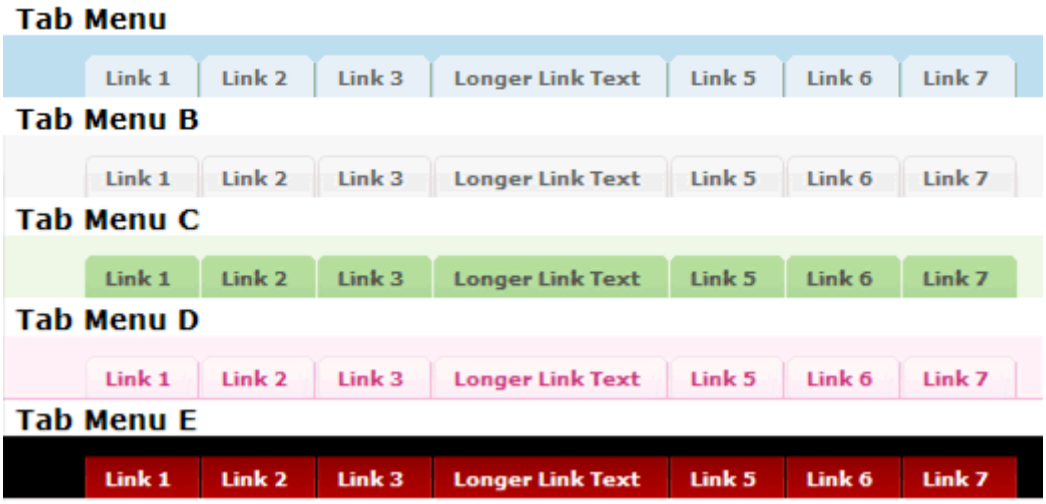

Figura 9.14. Ejemplos de menús horizontales con pestañas creados con CSS

El código fuente de los menús de la imagen anterior y muchos otros se puede encontrar en `http://exploding-boy.com/images/cssmenus/menus.html`

9.2.3. Menú horizontal avanzado

Además de los menús horizontales de solapas, existen muchos otros tipos diferentes de menús horizontales (y verticales) que se pueden realizar empleando exclusivamente CSS:

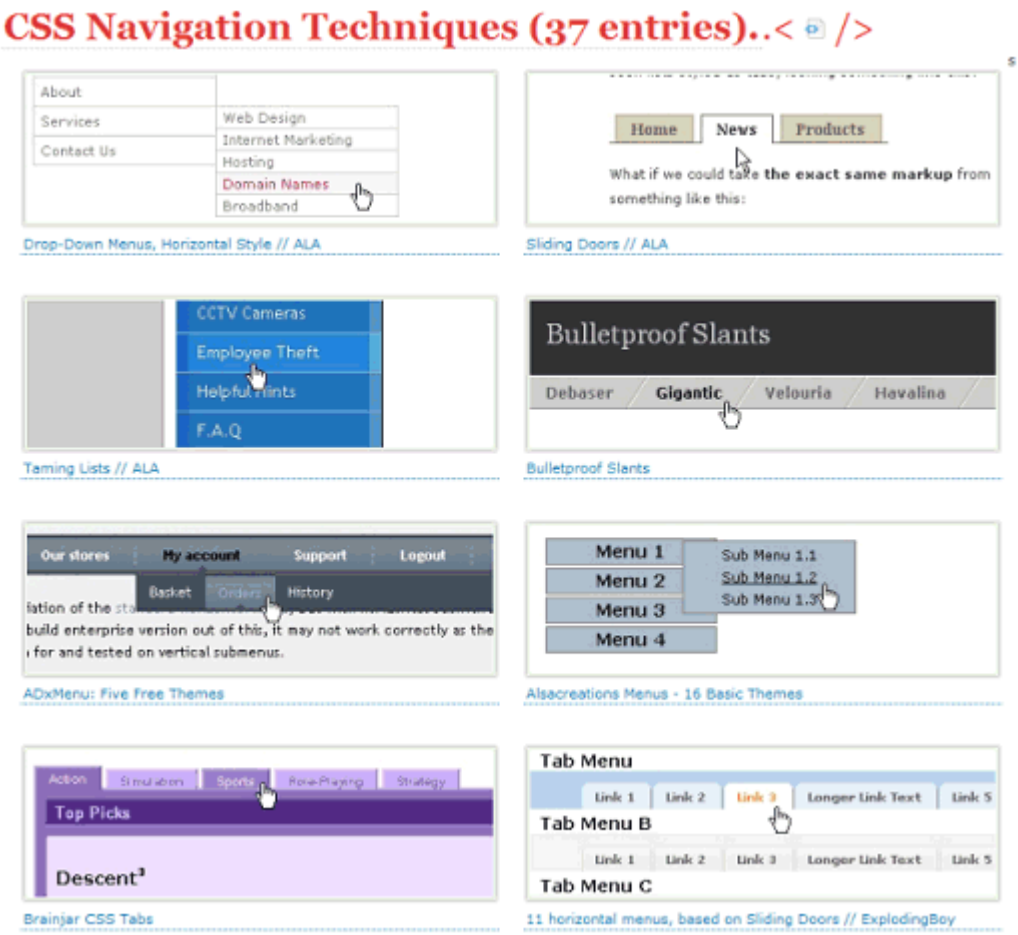

Figura 9.15. Ejemplos de menús horizontales y verticales complejos creados con CSS

El código CSS de todos los ejemplos de la imagen anterior y muchos otros se pueden encontrar en: `http://alvit.de/css-showcase/css-navigation-techniques-showcase.php`

Capítulo 10. Tablas

10.1. Estilos básicos

Cuando se aplican bordes a las celdas de una tabla, el aspecto por defecto con el que se muestra en un navegador es el siguiente:

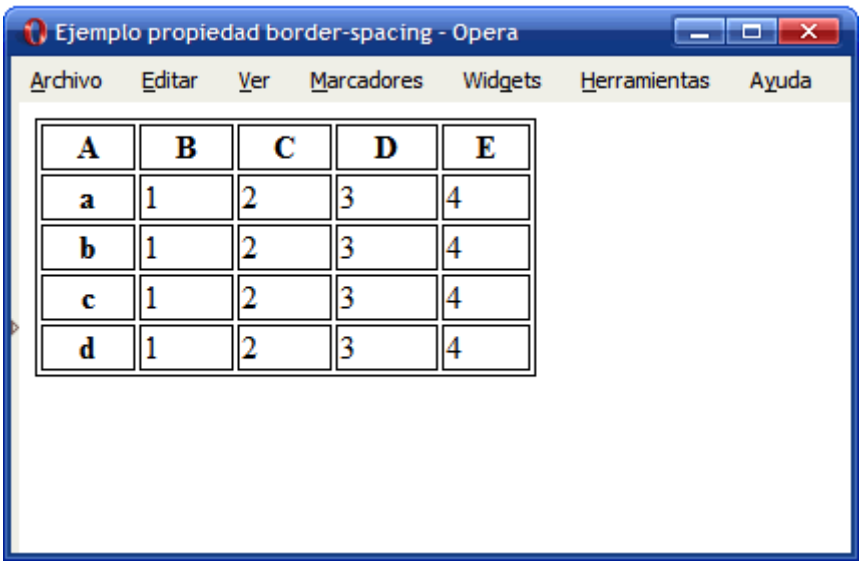

Figura 10.1. Aspecto por defecto de los bordes de una tabla

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

```
.normal {
  width: 250px;
  border: 1px solid #000;
}
.normal th, .normal td {
  border: 1px solid #000;
}

<table class="normal" summary="Tabla genérica">
  <tr>
    <th scope="col">A</th>
    <th scope="col">B</th>
    <th scope="col">C</th>
    <th scope="col">D</th>
    <th scope="col">E</th>
  </tr>
  ...
</table>
```

Normalmente, la separación entre los bordes de las diferentes celdas no es deseable, ya que es preferible mostrar las celdas sin separación. CSS define la propiedad `border-spacing` para controlar la separación entre las celdas de una tabla.

Tabla 10.1. Propiedad border-spacing

border-spacing	Espaciado entre bordes
Valores	`<medida> <medida>? | inherit`
Se aplica a	Todas las tablas
Valor inicial	0
Descripción	Establece la separación entre los bordes de las celdas adyacentes de una tabla

Si solamente se indica una medida como valor, se asigna ese valor como distancia horizontal y vertical. Si se indican dos medidas, la primera es la separación horizontal y la segunda es la separación vertical entre celdas.

Por tanto, el ejemplo anterior se puede modificar para no mostrar ningún espaciado entre las celdas adyacentes:

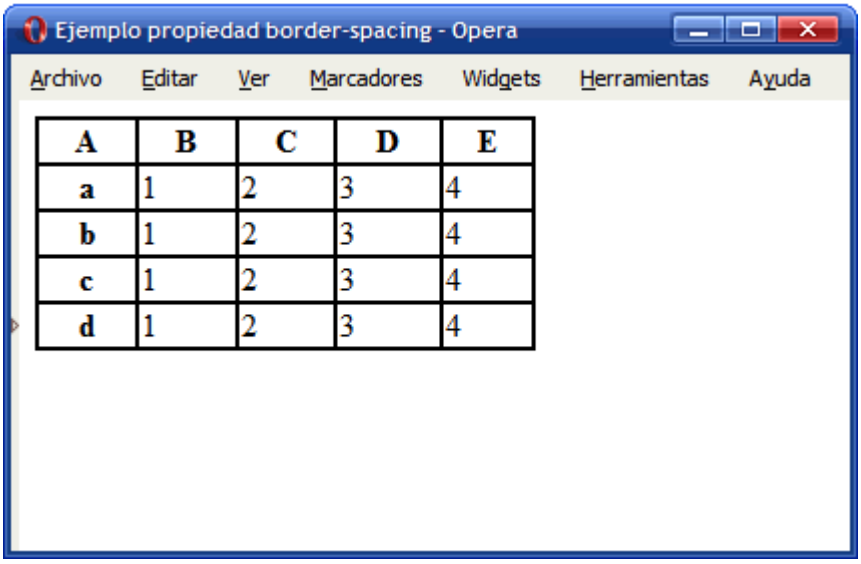

Figura 10.2. Ejemplo de propiedad border-spacing

La única regla añadida al código CSS es la siguiente:

```
.normal {
  width: 250px;
  border: 1px solid #000;
  border-spacing: 0;
}
```

Aunque la propiedad `border-spacing` soluciona el problema de la separación de las celdas adyacentes, origina otro problema: los bordes de las celdas adyacentes se juntan y por tanto, se muestran como si fueran un borde de anchura doble.

La propiedad `border-collapse` permite definir el mecanismo a seguir cuando se juntan dos bordes de celdas adyacentes.

Tabla 10.2. Propiedad border-collapse

border-collapse	**Fusión de bordes**
Valores	`collapse | separate | inherit`
Se aplica a	Todas las tablas
Valor inicial	separate
Descripción	Define el mecanismo de fusión de los bordes de las celdas adyacentes de una tabla

El valor `collapse` indica al navegador que los bordes de las celdas adyacentes se deben fusionar para que no se muestren bordes de anchura doble. Por lo tanto, el ejemplo anterior se puede rehacer para mostrar la tabla con bordes sencillos y sin separación entre celdas:

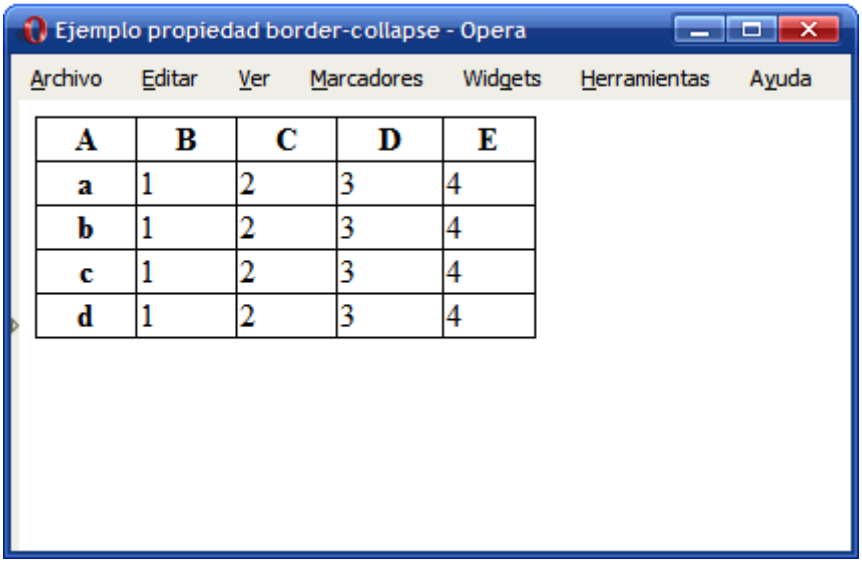

Figura 10.3. Ejemplo de propiedad border-collapse

El código CSS completo del ejemplo final se muestra a continuación:

```
.normal {
  width: 250px;
  border: 1px solid #000;
  border-spacing: 0;
  border-collapse: collapse;
}
.normal th, .normal td {
  border: 1px solid #000;
}

<table class="normal" summary="Tabla genérica">
  <tr>
    <th scope="col">A</th>
    <th scope="col">B</th>
    <th scope="col">C</th>
    <th scope="col">D</th>
    <th scope="col">E</th>
```

```
  </tr>
  ...
</table>
```

El mecanismo que sigue el valor `collapse` es muy complejo, ya que para fusionar diferentes bordes tiene en cuenta la anchura de cada borde, su estilo, el tipo de celda que contiene el borde (columna, fila, grupo de filas, grupo de columnas), etc.

10.2. Estilos avanzados

CSS define otras propiedades específicas para el control del aspecto de las tablas. Una de ellas es el tratamiento que reciben las celdas vacías de una tabla, que se controla mediante la propiedad `empty-cells`.

Tabla 10.3. Propiedad empty-cells

`empty-cells`	**Tratamiento de las celdas vacías**
Valores	`show | hide | inherit`
Se aplica a	Celdas de una tabla
Valor inicial	show
Descripción	Define el mecanismo utilizado para el tratamiento de las celdas vacías de una tabla

El valor `hide` indica que las celdas vacías no se deben mostrar. Una celda vacía es aquella que no tiene ningún contenido, ni siquiera un espacio en blanco o un ` `.

La siguiente imagen muestra las diferencias entre una tabla normal y una tabla con la propiedad `empty-cells: hide`:

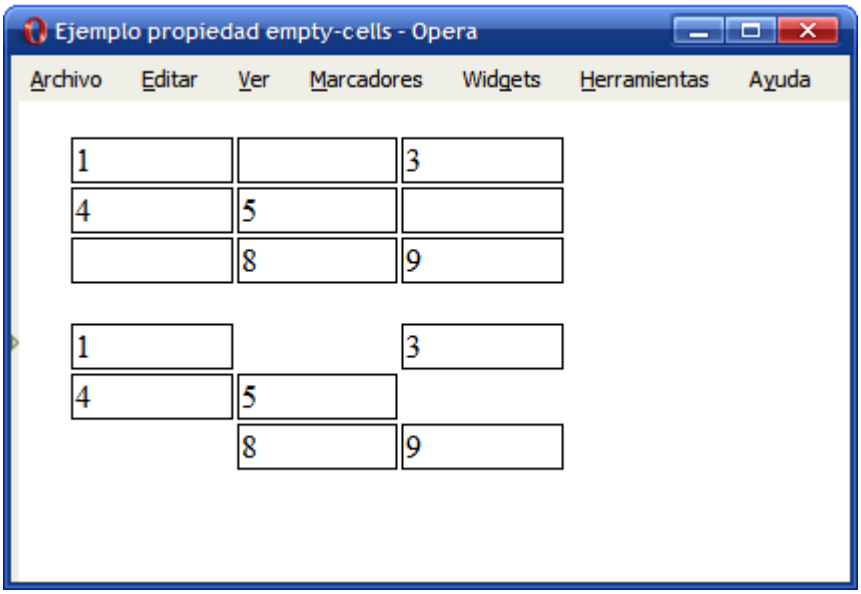

Figura 10.4. Ejemplo de propiedad empty-cells

Desafortunadamente, Internet Explorer 6 y las versiones anteriores no interpretan correctamente esta propiedad y muestran el ejemplo anterior de la siguiente manera:

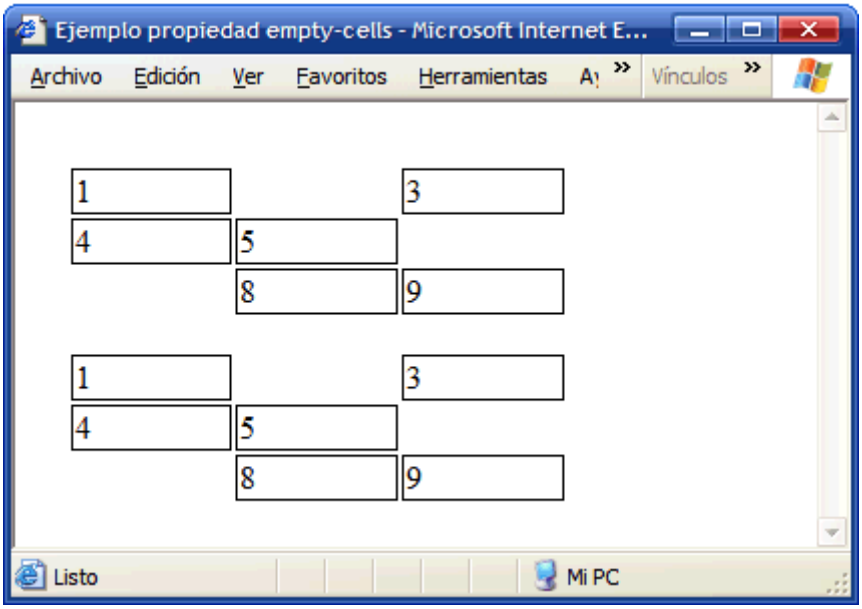

Figura 10.5. Internet Explorer no soporta la propiedad empty-cells

Por otra parte, el título de las tablas se establece mediante el elemento `<caption>`, que por defecto se muestra encima de los contenidos de la tabla. La propiedad `caption-side` permite controlar la posición del título de la tabla.

Tabla 10.4. Propiedad caption-side

`caption-side`	**Posición del título de la tabla**
Valores	`top` | `bottom` | `inherit`
Se aplica a	Los elementos caption
Valor inicial	top
Descripción	Establece la posición del título de la tabla

El valor `bottom` indica que el título de la tabla se debe mostrar después de los contenidos de la tabla. La alineación horizontal se controla mediante la propiedad `text-align`.

A continuación se muestra el código HTML y CSS de un ejemplo sencillo de uso de la propiedad `caption-side`:

```
.especial {
  caption-side: bottom;
}

<table class="especial" summary="Tabla genérica">
  <caption>Tabla 2.- Título especial</caption>
  <tr>
    <td>1</td>
    <td>2</td>
```

```
    <td>3</td>
  </tr>
  ...
</table>
```

Desafortunadamente, el navegador Opera 9 y versiones anteriores y el navegador Internet Explorer 6 y versiones anteriores no soportan esta propiedad y muestran el título de la tabla siempre encima de sus contenidos:

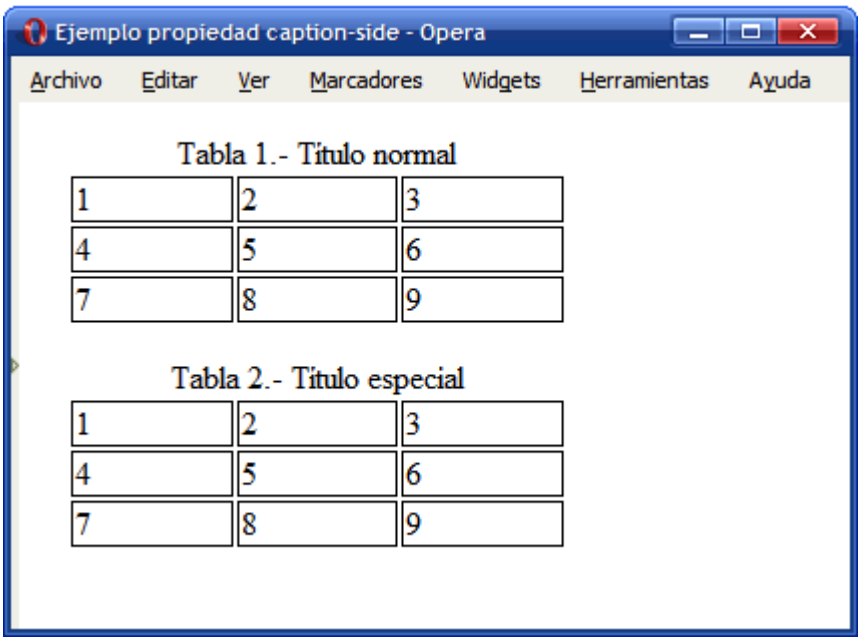

Figura 10.6. Opera e Internet Explorer no soportan la propiedad caption-side

El navegador Firefox sí que soporta esta propiedad y muestra el título de la segunda tabla debajo de sus contenidos, tal y como se ha indicado en el ejemplo:

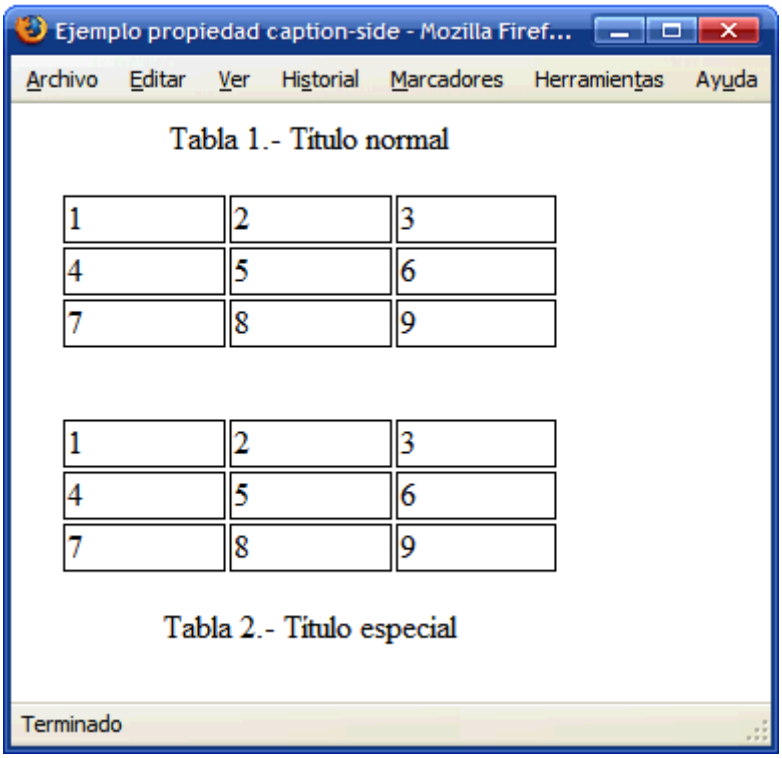

Figura 10.7. Ejemplo de propiedad caption-side

Ejercicio 11 Ver enunciado en la página 192

El resultado final del ejercicio anterior se podría completar añadiendo una pequeña mejora: que el color de las filas varíe cuando el usuario pasa el ratón por encima de cada fila. La *pseudo-clase* `:hover` permite añadir fácilmente esta característica:

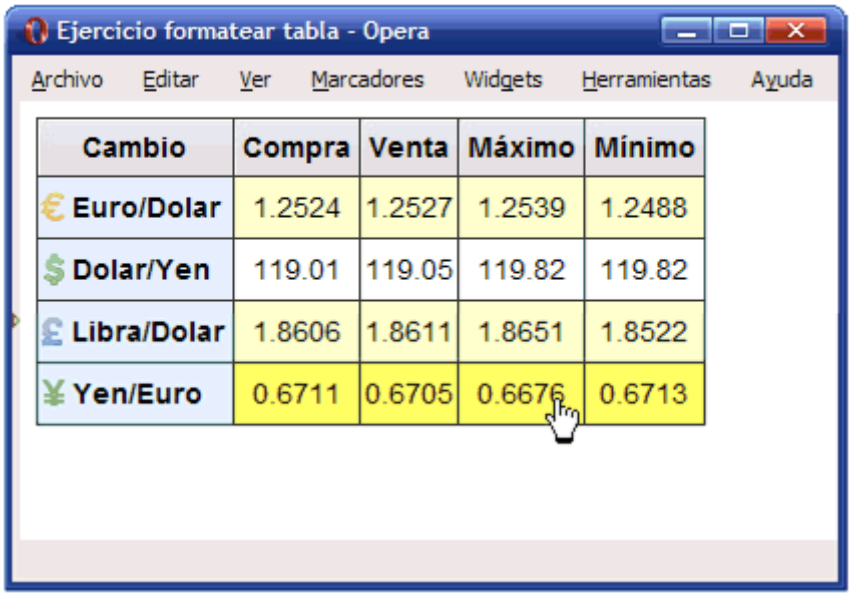

Figura 10.8. Pseudo-clase :hover en las filas de una tabla

La regla CSS necesaria se muestra a continuación:

```
table tr:hover {
  background: #FFFF66;
}
```

Desafortunadamente, Internet Explorer 6 y las versiones anteriores no soportan la *pseudo-clase* `:hover` en elementos diferentes a los enlaces, por lo que se debe recurrir a soluciones con JavaScript para mostrar de otro color la fila activa.