2.4 Aplicando CSS a nuestra plantilla

Como aprendimos más temprano en este mismo capítulo, todo elemento estructural es considerado una caja y la estructura completa es presentada como un grupo de cajas. Las cajas agrupadas constituyen lo que es llamado un Modelo de Caja.

Siguiendo con los conceptos básicos de CSS, vamos a estudiar lo que es llamado el Modelo de Caja Tradicional. Este modelo has sido implementado desde la primera versión de CSS y es actualmente soportado por cada navegador en el mercado, lo que lo ha convertido en un estándar para el diseño web.

Todo modelo, incluso aquellos aún en fase experimental, pueden ser aplicados a la misma estructura HTML, pero esta estructura debe ser preparada para ser afectada por estos estilos de forma adecuada. Nuestros documentos HTML deberán ser adaptados al modelo de caja seleccionado.

> **IMPORTANTE:** El Modelo de Caja Tradicional presentado posteriormente no es una incorporación de HTML5, pero es introducido en este libro por ser el único disponible en estos momentos y posiblemente el que continuará siendo utilizado en sitios webs desarrollados en HTML5 durante los próximos años. Si usted ya conoce cómo implementarlo, siéntase en libertad de obviar esta parte del capítulo.

2.5 Modelo de caja tradicional

Todo comenzó con tablas. Las tablas fueron los elementos que sin intención se volvieron la herramienta ideal utilizada por desarrolladores para crear y organizar cajas de contenido en la pantalla. Este puede ser considerado el primer modelo de caja de la web. Las cajas eran creadas expandiendo celdas y combinando filas de celdas, columnas de celdas y tablas enteras, unas sobre otras o incluso anidadas. Cuando los sitios webs crecieron y se volvieron más y más complejos esta práctica comenzó a presentar serios problemas relacionados con el tamaño y el mantenimiento del código necesario para crearlos.

Estos problemas iniciales hicieron necesario lo que ahora vemos como una práctica natural: la división entre estructura y presentación. Usando etiquetas **`<div>`** y estilos CSS fue posible reemplazar la función de tablas y efectivamente separar la estructura HTML de la presentación. Con elementos **`<div>`** y CSS podemos crear cajas en la pantalla, posicionar estas cajas a un lado o a otro y darles un tamaño, color o borde específico entre otras características. CSS provee propiedades específicas que nos permiten organizar las cajas acorde a nuestros deseos. Estas propiedades son lo suficientemente poderosas como para crear un modelo de caja que se transformó en lo que hoy conocemos como Modelo de Caja Tradicional.

Algunas deficiencias en este modelo mantuvieron a las tablas vivas por algún tiempo, pero los principales desarrolladores, influenciados por el suceso de las implementaciones Ajax y una cantidad enorme de nuevas aplicaciones interactivas, gradualmente volvieron a las etiquetas **`<div>`** y estilos CSS en un estándar. Finalmente el Modelo de Caja Tradicional fue adoptado a gran escala.

Plantilla

En el Capítulo 1 construimos una plantilla HTML5. Esta plantilla tiene todos los elementos necesarios para proveer estructura a nuestro documento, pero algunos detalles deben ser agregados para aplicar los estilos CSS y el Modelo de Caja Tradicional.

Este modelo necesita agrupar cajas juntas para ordenarlas horizontalmente. Debido a que el contenido completo del cuerpo es creado a partir de cajas, debemos agregar un elemento **`<div>`** para agruparlas, centrarlas y darles un tamaño específico.

La nueva plantilla lucirá de este modo:

```
<!DOCTYPE  html>
<html  lang="es">
<head>
  <meta charset="iso-8859-1">
  <meta name="description" content="Ejemplo de HTML5"> <meta
  name="keywords" content="HTML5, CSS3, JavaScript">
  <title>Este texto es el título del documento</title> <link
  rel="stylesheet" href="misestilos.css">
</head>
<body>
<div id="agrupar">
  <header id="cabecera">
    <h1>Este es el título principal del sitio
  web</h1> </header>
  <nav id="menu">
    <ul>
      <li>principal</li>
      <li>fotos</li>
      <li>videos</li>
      <li>contacto</li>
    </ul>
  </nav>
  <section id="seccion">
    <article>
      <header>
        <hgroup>
          <h1>Título del mensaje uno</h1>
          <h2>Subtítulo del mensaje uno</h2>
        </hgroup>
        <time datetime="2011-12-10" pubdate>publicado 10-12-2011 </time>
      </header>
      Este es el texto de mi primer
      mensaje <figure>
        <img src="http://minkbooks.com/content/myimage.jpg">
        <figcaption>
          Esta es la imagen del primer
        mensaje </figcaption>
      </figure>
      <footer>
        <p>comentarios
      (0)</p> </footer>
    </article>
    <article>
      <header>
        <hgroup>
          <h1>Título del mensaje dos</h1>
          <h2>Subtítulo del mensaje dos</h2>
        </hgroup>
        <time datetime="2011-12-15" pubdate>publicado 15-12-
        2011 </time>
      </header>
      Este es el texto de mi segundo
      mensaje <footer>
        <p>comentarios
      (0)</p> </footer>
    </article>
  </section>
  <aside id="columna">
    <blockquote>Mensaje número uno</blockquote>
    <blockquote>Mensaje número dos</blockquote>
  </aside>
  <footer id="pie">
```

```
    Derechos Reservados &copy; 2010-
  2011 </footer>
</div>
</body>
</html>
```

Listado 2-25. *Nueva plantilla HTML5 lista para estilos CSS.*

El Listado 2-25 provee una nueva plantilla lista para recibir los estilos CSS. Dos cambios importantes pueden distinguirse al comparar este código con el del Listado 1-18 del Capítulo 1. El primero es que ahora varias etiquetas fueron identificadas con los atributos **id** y **class**. Esto significa que podemos referenciar un elemento específico desde las reglas CSS con el valor de su atributo **id** o podemos modificar varios elementos al mismo tiempo usando el valor de su atributo **class**.

El segundo cambio realizado a la vieja plantilla es la adición del elemento ** ** mencionado anteriormente. Este ** ** fue identificado con el atributo y el valor **id="agrupar"**, y es cerrado al final del cuerpo con la etiqueta de cierre ** **. Este elemento se encarga de agrupar todos los demás elementos permitiéndonos aplicar el modelo de caja al cuerpo y designar su posición horizontal, como veremos más adelante.

Hágalo usted mismo: Compare el código del Listado 1-18 del Capítulo 1 con el código en el Listado 2-25 de este capítulo y ubique las etiquetas de apertura y cierre del elemento ** ** utilizado para agrupar al resto. También compruebe cuáles elementos se encuentran ahora identificados con el atributo **id** y cuáles con el atributo **class**. Confirme que los valores de los atributos **id** son únicos para cada etiqueta. También necesitará reemplazar el código en el archivo HTML creado anteriormente por el del Listado 2-25 para aplicar los siguientes estilos CSS.

Con el documento HTML finalizado es tiempo de trabajar en nuestro archivo de estilos.

Selector universal *

Comencemos con algunas reglas básicas que nos ayudarán a proveer consistencia al diseño:

```
{
margin:   0px;
padding: 0px;
}
```

Listado 2-26. *Regla CSS general.*

Normalmente, para la mayoría de los elementos, necesitamos personalizar los márgenes o simplemente mantenerlos al mínimo. Algunos elementos por defecto tienen márgenes que son diferentes de cero y en la mayoría de los casos demasiado amplios. A medida que avanzamos en la creación de nuestro diseño encontraremos que la mayoría de los elementos utilizados deben tener un margen de 0 pixeles. Para evitar el tener que repetir estilos constantemente, podemos utilizar el selector universal.

La primera regla en nuestro archivo CSS, presentada en el Listado 2-26, nos asegura que todo elemento tendrá un margen interno y externo de 0 pixeles. De ahora en más solo necesitaremos modificar los márgenes de los elementos que queremos que sean mayores que cero.

Conceptos básicos: Recuerde que en HTML cada elemento es considerado como una caja. El margen (**margin**) es en realidad el espacio alrededor del elemento, el que se encuentra por fuera del borde de esa caja (el estilo **padding**, por otro lado, es el espacio alrededor del contenido del elemento pero dentro de sus bordes, como el espacio entre el título y el borde de la caja virtual formada por el elemento **<h1>** que contiene ese título). El tamaño del margen puede ser definido por lados específicos del elemento o todos sus lados a la vez. El estilo **margin: 0px** en nuestro ejemplo establece un margen 0 o nulo para cada elemento de la caja. Si el tamaño hubiese sido especificado en 5 pixeles, por ejemplo, la caja tendría un espacio de 5 pixeles de ancho en todo su contorno. Esto significa que la caja estaría separada de sus vecinas por 5 pixeles. Volveremos sobre este tema más adelante en este capítulo.

Hágalo usted mismo: Debemos escribir todas las reglas necesarias para otorgar estilo a nuestra plantilla en un archivo CSS. El archivo ya fue incluido dentro del código HTML por medio de la etiqueta **<link>**, por lo que lo único que tenemos que hacer es crear un archivo de texto vacío con nuestro editor de textos preferido, grabarlo con el nombre **misestilos.css** y luego copiar en su interior la regla del Listado 2-26 y todas las presentadas a continuación.

Nueva jerarquía para cabeceras

En nuestra plantilla usamos elementos **<h1>** y **<h2>** para declarar títulos y subtítulos de diferentes secciones del documento. Los estilos por defecto de estos elementos se encuentran siempre muy lejos de lo que queremos y además en HTML5 podemos reconstruir la jerarquía H varias veces en cada sección (como aprendimos en el capítulo anterior). El elemento **<h1>**, por ejemplo, será usado varias veces en el documento, no solo para el título principal de la página web como pasaba anteriormente sino también para secciones internas, por lo que tenemos que otorgarle los estilos apropiados:

```
h1 {
  font: bold 20px verdana, sans-serif;
}
h2 {
  font: bold 14px verdana, sans-serif;
}
```

Listado 2-27. *Agregando estilos para los elementos <h1> y <h2>.*

La propiedad **font**, asignada a los elementos **<h1>** y **<h2>** en el Listado 2-27, nos permite declarar todos los estilos para el texto en una sola línea. Las propiedades que pueden ser declaradas usando **font** son: **font-style**, **font-variant**, **font-weight**, **font-size/line-height**, y **font-family** en este orden. Con estas reglas estamos cambiando el grosor, tamaño y tipo de letra del texto dentro de los elementos **<h1>** y **<h2>** a los valores que deseamos.

Declarando nuevos elementos HTML5

Otra regla básica que debemos declarar desde el comienzo es la definición por defecto de elementos estructurales de HTML5. Algunos navegadores aún no reconocen estos elementos o los tratan como elementos *inline* (en línea). Necesitamos declarar los nuevos elementos HTML5 como elementos *block* para asegurarnos de que serán tratados como regularmente se hace con elementos ** ** y de este modo construir nuestro modelo de caja:

```
header, section, footer, aside, nav, article, figure,
figcaption, hgroup{
  display: block;
}
```

Listado 2-28. *Regla por defecto para elementos estructurales de HTML5.*

A partir de ahora, los elementos afectados por la regla del Listado 2-28 serán posicionados uno sobre otro a menos que especifiquemos algo diferente más adelante.

Centrando el cuerpo

El primer elemento que es parte del modelo de caja es siempre **<body>**. Normalmente, por diferentes razones de diseño, el contenido de este elemento debe ser posicionado horizontalmente. Siempre deberemos especificar el tamaño de este contenido, o un tamaño máximo, para obtener un diseño consistente a través de diferentes configuraciones de pantalla.

```
body {
  text-align: center;
}
```

Listado 2-29. *Centrando el cuerpo.*

Por defecto, la etiqueta **<body>** (como cualquier otro elemento *block*) tiene un valor de ancho establecido en 100%. Esto significa que el cuerpo ocupará el ancho completo de la ventana del navegador. Por lo tanto, para centrar la página en la pantalla necesitamos centrar el contenido dentro del cuerpo. Con la regla agregada en el Listado 2-29, todo lo que se encuentra dentro de **<body>** será centrado en la ventana, centrando de este modo toda la página web.

Creando la caja principal

Siguiendo con el diseño de nuestra plantilla, debemos especificar una tamaño o tamaño máximo para el contenido del cuerpo. Como seguramente recuerda, en el Listado 2-25 en este mismo capítulo agregamos un elemento ** ** a la plantilla para agrupar todas las cajas dentro del cuerpo. Este ** ** será considerado la caja principal para la construcción de nuestro modelo de caja (este es el propósito por el que lo agregamos). De este modo, modificando el tamaño de este elemento lo hacemos al mismo tiempo para todos los demás:

```
#agrupar {
  width: 960px;
  margin: 15px auto;
  text-align: left;
}
```

Listado 2-30. *Definiendo las propiedades de la caja principal.*

La regla en el Listado 2-30 está referenciando por primera vez un elemento a través del valor de su atributo **id**. El carácter **#** le está diciendo al navegador que el elemento afectado por este conjunto de estilos tiene el atributo **id** con el valor **agrupar**.

Esta regla provee tres estilos para la caja principal. El primer estilo establece un valor fijo de 960 pixeles. Esta caja tendrá siempre un ancho de 960 pixeles, lo que representa un valor común para un sitio web estos días (los valores se encuentran entre 960 y 980 pixeles de ancho, sin embargo estos parámetros cambian constantemente a través del tiempo, por supuesto).

El segundo estilo es parte de lo que llamamos el Modelo de Caja Tradicional. En la regla previa (Listado 2-29), especificamos que el contenido del cuerpo sería centrado horizontalmente con el estilo **text-align: center**. Pero esto solo afecta contenido *inline*, como textos o imágenes. Para elementos *block*, como un ** **, necesitamos establecer un valor específico para sus márgenes que los adapta automáticamente al tamaño de su elemento padre. La propiedad **margin** usada para este propósito puede tener cuatro valores: superior, derecho, inferior, izquierdo, en este orden. Esto significa que el primer valor declarado en el estilo representa el margen de la parte superior del elemento, el segundo es el margen de la derecha, y así sucesivamente. Sin embargo, si solo escribimos los primeros dos parámetros, el resto tomará los mismos valores. En nuestro ejemplo estamos usando esta técnica.

En el Listado 2-30, el estilo **margin: 15px auto** asigna 15 pixeles al margen superior e inferior del elemento ** ** que está afectando y declara como automático el tamaño de los márgenes de izquierda y derecha (los dos valores declarados son usados para definir los cuatro márgenes). De esta manera, habremos generado un espacio de 15 pixeles en la parte superior e inferior del cuerpo y los espacios a los laterales (margen izquierdo y derecho) serán calculados automáticamente de acuerdo al tamaño del cuerpo del documento y el elemento ** **, efectivamente centrando el contenido en pantalla.

La página web ya está centrada y tiene un tamaño fijo de 960 pixeles. Lo próximo que necesitamos hacer es prevenir un problema que ocurre en algunos navegadores. La propiedad **text-align** es hereditaria. Esto significa que todos los elementos dentro del cuerpo y su contenido serán centrados, no solo la caja principal. El estilo asignado a **<body>** en el Listado 2-29 será asignado a cada uno de sus hijos. Debemos retornar este estilo a su valor por defecto para el resto del documento. El tercer y último estilo incorporado en la regla del Listado 2-30 (**text-align: left**) logra este propósito. El resultado final es que el contenido del cuerpo es centrado pero el contenido de la caja principal (el ** ** identificado como **agrupar**) es alineado nuevamente hacia la izquierda, por lo tanto todo el resto del código HTML dentro de esta caja hereda este estilo.

> **Hágalo usted mismo:** Si aún no lo ha hecho, copie cada una de las reglas listadas hasta este punto dentro de un archivo de texto vacío llamado **misestilos.css**. Este archivo debe estar ubicado en el mismo directorio (carpeta) que el archivo HTML con el código del Listado 2-25. Al terminar, deberá contar con dos archivos, uno con el código HTML y otro llamado **misestilos.css** con todos los estilos CSS estudiados desde el Listado 2-26. Abra el archivo HTML en su navegador y en la pantalla podrá notar la caja creada.

La cabecera

Continuemos con el resto de los elementos estructurales. Siguiendo la etiqueta de apertura del ** ** principal se encuentra el primer elemento estructural de HTML5: **<header>**. Este elemento contiene el título principal de nuestra página web y estará ubicado en la parte superior de la pantalla. En nuestra plantilla, **<header>** fue identificado con el atributo **id** y el valor **cabecera**.

Como ya mencionamos, cada elemento *block* , así como el cuerpo, por defecto tiene un valor de ancho del 100%. Esto significa que el elemento ocupará todo el espacio horizontal disponible. En el caso del cuerpo, ese espacio es el ancho total de la pantalla visible (la ventana del navegador), pero en el resto de los elementos el espacio máximo disponible estará determinado por el ancho de su elemento padre. En nuestro ejemplo, el espacio máximo disponible para los elementos dentro de la caja principal será de 960 pixeles, porque su padre es la caja principal la cual fue previamente configurada con este tamaño.

```
#cabecera { background:
  #FFFBB9;
  border: 1px solid
  #999999; padding: 20px;
}
```

Listado 2-31. *Agregando estilos para <header>.*

Debido a que **<header>** ocupará todo el espacio horizontal disponible en la caja principal y será tratado como un elemento *block* (y por esto posicionada en la parte superior de la página), lo único que resta por hacer es asignar estilos que nos permitirán reconocer el elemento cuando es presentado en pantalla. En la regla mostrada en el Listado 2-31 le otorgamos a **<header>** un fondo amarillo, un borde sólido de 1 pixel y un margen interior de 20 pixeles usando la propiedad **padding**.

Barra de navegación

Siguiendo al elemento **<header>** se encuentra el elemento **<nav>**, el cual tiene el propósito de proporcionar ayuda para la navegación. Los enlaces agrupados dentro de este elemento representarán el menú de nuestro sitio web. Este menú será una simple barra ubicada debajo de la cabecera. Por este motivo, del mismo modo que el elemento **<header>**, la mayoría de los estilos que necesitamos para posicionar el elemento **<nav>** ya fueron asignados: **<nav>** es un elemento *block* por lo que será ubicado debajo del elemento previo, su ancho por defecto será 100% por lo que será tan ancho como su padre (el ** ** principal), y (también por defecto) será tan alto como su contenido y los márgenes predeterminados. Por lo tanto, lo único que nos queda por hacer es mejorar su aspecto en pantalla. Esto último lo logramos agregando un fondo gris y un pequeño margen interno para separar las opciones del menú del borde del elemento:

```
#menu {
  background: #CCCCCC;
  padding: 5px 15px;
}
#menu li {
  display: inline-
  block; list-style:
  none; padding: 5px;
  font: bold 14px verdana, sans-serif;
}
```

Listado 2-32. *Agregando estilos para <nav>.*

En el Listado 2-32, la primera regla referencia al elemento **<nav>** por su atributo **id**, cambia su color de fondo y agrega márgenes internos de **5px** y **15px** con la propiedad **padding**.

> **Conceptos básicos:** La propiedad **padding** trabaja exactamente como **margin**. Cuatro valores pueden ser especificados: superior, derecho, inferior, izquierdo, en este orden. Si solo declaramos un valor, el mismo será asignado para todos los espacios alrededor del contenido del elemento. Si en cambio especificamos dos valores, entonces el primero será asignado como margen interno de la parte superior e inferior del contenido y el segundo valor será asignado al margen interno de los lados, izquierdo y derecho.

Dentro de la barra de navegación hay una lista creada con las etiquetas **<ul>** y **<li>**. Por defecto, los ítems de una lista son posicionados unos sobre otros. Para cambiar este comportamiento y colocar cada opción del menú una al lado de la otra, referenciamos los elementos **<li>** dentro de este elemento **<nav>** en particular usando el selector **#menu li**, y luego asignamos a todos ellos el estilo **display: inline-block** para convertirlos en lo que se llama cajas *inline*. A diferencia de los elementos *block*, los elementos afectados por el parámetro **inline-block** estandarizado en CSS3 no generan ningún salto de línea pero nos permiten tratarlos como elementos block y así declarar un valor de ancho determinado. Este parámetro también ajusta el tamaño del elemento de acuerdo con su contenido cuando el valor del ancho no fue especificado.

En esta última regla también eliminamos el pequeño gráfico generado por defecto por los navegadores delante de cada opción del listado utilizando la propiedad **list-style**.

Section y aside

Los siguientes elementos estructurales en nuestro código son dos cajas ordenadas horizontalmente. El Modelo de Caja Tradicional es construido sobre estilos CSS que nos permiten especificar la posición de cada caja. Usando la propiedad **float** podemos posicionar estas cajas del lado izquierdo o derecho de acuerdo a nuestras necesidades. Los elementos que utilizamos en nuestra plantilla HTML para crear estas cajas son **<section>** y **<aside>**, cada uno identificado con el atributo **id** y los valores **seccion** y **columna** respectivamente.

```
#seccion {
  float: left;
  width: 660px;
  margin: 20px;
}
#columna {
  float: left;
  width: 220px;
  margin: 20px 0px;
  padding: 20px;
  background: #CCCCCC;
}
```

Listado 2-33. *Creando dos columnas con la propiedad float.*

La propiedad de CSS **float** es una de las propiedades más ampliamente utilizadas para aplicar el Modelo de Caja Tradicional. Hace que el elemento flote hacia un lado o al otro en el espacio disponible. Los elementos afectados por **float** actúan como elementos *block* (con la diferencia de que son ubicados de acuerdo al valor de esta propiedad y no el flujo normal del documento). Los elementos son movidos a izquierda o derecha en el área disponible, tanto como sea posible, respondiendo al valor de **float**.

Con las reglas del Listado 2- 33 declaramos la posición de ambas cajas y sus respectivos tamaños, generando así las columnas visibles en la pantalla. La propiedad **float** mueve la caja al espacio disponible del lado especificado por su valor, **width** asigna un tamaño horizontal y **margin**, por supuesto, declara el margen del elemento.

Afectado por estos valores, el contenido del elemento **<section>** estará situado a la izquierda de la pantalla con un tamaño de 660 pixeles, más 40 pixeles de margen, ocupando un espacio total de 700 pixeles de ancho.

La propiedad **float** del elemento **<aside>** también tiene el valor **left** (izquierda). Esto significa que la caja generada será movida al espacio disponible a su izquierda. Debido a que la caja previa creada por el elemento **<section>** fue también movida a la izquierda de la pantalla, ahora el espacio disponible será solo el que esta caja dejó libre. La nueva caja quedará ubicada en la misma línea que la primera pero a su derecha, ocupando el espacio restante en la línea, creando la segunda columna de nuestro diseño.

El tamaño declarado para esta segunda caja fue de 220 pixeles . También agregamos un fondo gris y configuramos un margen interno de 20 pixeles. Como resultado final, el ancho de esta caja será de 220 pixeles más 40 pixeles agregados por la propiedad **padding** (los márgenes de los lados fueron declarados a **0px**).

> **Conceptos básicos:** El tamaño de un elemento y sus márgenes son agregados para obtener el valor real ocupado en pantalla. Si tenemos un elemento de 200 pixeles de ancho y un margen de 10 pixeles a cada lado, el área real ocupada por el elemento será de 220 pixeles. El total de 20 pixeles del margen es agregado a los 200 pixeles del elemento y el valor final es representado en la pantalla. Lo mismo pasa con las propiedades **padding** y **border**. Cada vez que agregamos un borde a un elemento o creamos un espacio entre el contenido y el borde usando **padding** esos valores serán agregados al ancho del elemento para obtener el valor real cuando el elemento es mostrado en pantalla. Este valor real es calculado con la fórmula: **tamaño + márgenes + márgenes internos + bordes**.
>
> **Hágalo usted mismo:** Lea el código del Listado 2- 25. Controle cada regla CSS creada hasta el momento y busque en la plantilla los elementos HTML correspondientes a cada una de ellas. Siga las referencias, por ejemplo las claves de los elementos (como **h1)** y los atributos **id** (como **cabecera**), para entender cómo trabajan las referencias y cómo los estilos son asignados a cada elemento.

Footer

Para finalizar la aplicación del Modelo de Caja Tradicional, otra propiedad CSS tiene que ser aplicada al elemento **<footer>**. Esta propiedad devuelve al documento su flujo normal y nos permite posicionar **<footer>** debajo del último elemento en lugar de a su lado:

```
#pie {
  clear: both; text-
  align: center;
  padding: 20px;
  border-top: 2px solid #999999;
}
```

Listado 2-34. *Otorgando estilos a <footer> y recuperando el normal flujo del documento.*

La regla del Listado 2-34 declara un borde de 2 pixeles en la parte superior de **<footer>**, un margen interno (**padding**) de 20 pixeles, y centra el texto dentro del elemento. A sí mismo, restaura el normal flujo del documento con la propiedad **clear**. Esta propiedad simplemente restaura las condiciones normales del área ocupada por el elemento, no permitiéndole posicionarse adyacente a una caja flotante. El valor usualmente utilizado es **both**, el cual significa que ambos lados del elemento serán restaurados y el elemento seguirá el flujo normal (este elemento ya no es flotante como los anteriores). Esto, para un elemento *block*, quiere decir que será posicionado debajo del último elemento, en una nueva línea.

La propiedad **clear** también empuja los elementos verticalmente, haciendo que las cajas flotantes ocupen un área real en la pantalla. Sin esta propiedad, el navegador presenta el documento en pantalla como si los elementos flotantes no existieran y las cajas se superponen.

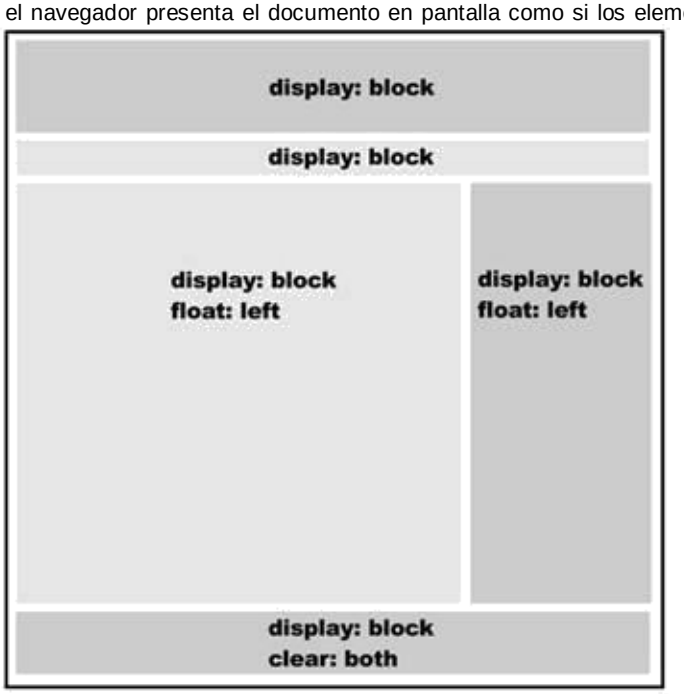

Figura 2-2. Representación visual del modelo de caja tradicional.

Cuando tenemos cajas posicionadas una al lado de la otra en el Modelo de Caja Tradicional siempre necesitamos crear un elemento con el estilo **`clear: both`** para poder seguir agregando otras cajas debajo de un modo natural. La Figura 2-2 muestra una representación visual de este modelo con los estilos básicos para lograr la correcta disposición en pantalla.

Los valores **`left`** (izquierda) y **`right`** (derecha) de la propiedad **`float`** no significan que las cajas deben estar necesariamente posicionadas del lado izquierdo o derecho de la ventana. Lo que los valores hacen es volver flotante ese lado del elemento, rompiendo el flujo normal del documento. Si el valor es **`left`**, por ejemplo, el navegador tratará de posicionar el elemento del lado izquierdo en el espacio disponible. Si hay espacio disponible luego de otro elemento, este nuevo elemento será situado a su derecha, porque su lado izquierdo fue configurado como flotante. El elemento flota hacia la izquierda hasta que encuentra algo que lo bloquea, como otro elemento o el borde de su elemento padre. Esto es importante cuando queremos crear varias columnas en la pantalla. En este caso cada columna tendrá el valor **`left`** en la propiedad **`float`** para asegurar que cada columna estará continua a la otra en el orden correcto. De este modo, cada columna flotará hacia la izquierda hasta que es bloqueada por otra columna o el borde del elemento padre.

Últimos toques

Lo único que nos queda por hacer es trabajar en el diseño del contenido. Para esto, solo necesitamos configurar los pocos elementos HTML5 restantes:

```
article {
  background: #FFFBCC;
  border: 1px solid
  #999999; padding: 20px;
  margin-bottom: 15px;
}
article footer {
  text-align: right;
}
time {
  color: #999999;
}
figcaption {
  font: italic 14px verdana, sans-serif;
}
```

Listado 2-35. *Agregando los últimos toques a nuestro diseño básico.*

La primera regla del Listado 2-35 referencia todos los elementos **`<article>`** y les otorga algunos estilos básicos (color de fondo, un borde sólido de 1 pixel, margen interno y margen inferior). El margen inferior de 15 pixeles tiene el propósito de separar un elemento **`<article>`** del siguiente verticalmente.

Cada elemento **`<article>`** cuenta también con un elemento **`<footer>`** que muestra el número de comentarios recibidos. Para referenciar un elemento **`<footer>`** dentro de un elemento **`<article>`**, usamos el selector **`article footer`** que significa “cada **`<footer>`** dentro de un **`<article>`** será afectado por los siguientes estilos”. Esta técnica de referencia fue aplicada aquí para alinear a la derecha el texto dentro de los elementos **`<footer>`** de cada **`<article>`**.

Al final del código en el Listado 2-35 cambiamos el color de cada elemento **`<time>`** y diferenciamos la descripción de la imagen (insertada con el elemento **`<figcaption>`**) del resto del texto usando una tipo de letra diferente.

> **EJERCICIO 1 Hágalo usted mismo:** copie cada regla CSS listada en este capítulo desde el Listado 2-26, una debajo de otra, dentro del archivo **`misestilos.css`**, y luego abra el archivo HTML con la plantilla creada en el Listado 2-25 en su navegador. Esto le mostrará cómo funciona el Modelo de Caja Tradicional y cómo los elementos estructurales son organizados en pantalla.

DESARROLLE LOS SIEGUIENTES EJEMPLOS EN UNA PAGINA WEB Y APLIQUE DIFERENTES FORMATOS DE ESTILO EN CSS3 Y CREE UNA PLANTILLA DE MODELO DE CAJA TRADICIONAL COMO EN EL EJERCICIO ANTERIOR (TEMA LIBRE).

AGREGUE:

- 4 PARRAFOS CON SUS ENCABEZADOS CADA UNO
- 4 IMÁGENES
- 1 FONDO
- 1 FORMULARIO RELACIONADO AL TEMA QUE UD ELIJIO.

Ejemplo de colocar una imagen de fondo en una página web con CSS.

Para colocar una imagen de fondo en una página web con CSS utilizamos el siguiente código:

```
<html>
<head>

<style type="text/css">
body
{
background-image:
url('IMAGEN.jpg')
}
</style>

</head>

<body>
</body>

</html>
```

Ejemplo de manejo del formato del texto en CSS

Mediante hojas de estilo, CSS, podemos darle al texto diferentes formatos de manera muy simple:
Como subrayar el texto, atravezarlo con una línea, poner una línea superior o hacerlo parpadear (no funciona en IE), así como agregar diferentes colores a las ligas de una página.

Aquí tenemos un ejemplo sencillo sobre el manejo del formato del texto en CSS:

```
<html>
<head>
<style type="text/css">
h1 {text-decoration: overline}
h2 {text-decoration: line-through}
h3 {text-decoration: underline}
h4 {text-decoration: blink}
a {text-decoration: none}
```

```
</style>
</head>

<body>
<h1>Encabezado 1</h1>
<h2>Encabezado 2</h2>
<h3>Encabezado 3</h3>
<h4>Encabezado 4</h4>
<p><a href="http://www.ejemplode.com"> Liga </a></p>
</body>

</html>
```

Ejemplo de Fondo fijo en CSS

Alguna vez te has preguntado cómo se hacen los fondos de las páginas que no caminan, es decir, que cuando recorres para leer el texto el fondo se queda fijo. Pues aquí te tengo la solución para poner un fondo fijo con CSS.

```
html {

    background: url(img/bg.jpg) no-repeat center center fixed;

    -webkit-background-size: cover;

    -moz-background-size: cover;

    -o-background-size: cover;

    background-size: cover;

}
```