
HTML
CSS

Ya hemos visto cómo organizar la estructura del documento mediante html. Ahora es momento de analizar CSS, su relevancia dentro de esta unión estratégica y su influencia sobre la presentación de documentos HTML.

Oficialmente CSS nada tiene que ver con HTML5. CSS no es parte de la especificación y nunca lo fue. Este lenguaje es, de hecho, un complemento desarrollado para superar las limitaciones y reducir la complejidad de HTML. Al comienzo, atributos dentro de las etiquetas HTML proveían estilos esenciales para cada elemento, pero a medida que el lenguaje evolucionó, las páginas se volvieron más complejas y HTML por sí mismo, no pudo satisfacer las demandas de los diseñadores. En consecuencia, CSS pronto fue adoptado como la forma de separar la estructura de la presentación. Desde entonces, CSS ha crecido y ganado importancia, pero siempre desarrollado en paralelo, enfocado en las necesidades de los diseñadores y apartado del proceso de evolución de HTML.

La integración entre HTML y CSS es vital para el desarrollo web y esta es la razón por la que cada vez que mencionamos HTML5, también estamos haciendo referencia a CSS3, aunque oficialmente se trate de dos tecnologías completamente separadas.

En este momento, las nuevas características incorporadas en CSS3 están siendo implementadas e incluidas, junto al resto de la especificación, en navegadores compatibles con HTML5.

Incorporar estilos al documento

Aplicar estilos a los elementos HTML cambia la forma en que estos son presentados en pantalla. Los navegadores proveen estilos por defecto que, en la mayoría de los casos, no son suficientes para satisfacer las necesidades de los diseñadores. Para cambiar esto, podemos sobrescribir estos estilos con los nuestros usando diferentes técnicas:

Estilos en línea: Una de las técnicas más simples para incorporar estilos CSS a un documento HTML es la de asignar los estilos dentro de las etiquetas por medio del atributo style.

```
<p style="font-size: 20px">Mi texto</p>
```

Este método es una buena manera de probar estilos y obtener una vista rápida de sus efectos, pero no es recomendable para aplicar estilos a todo el documento.

Estilos embebidos: Una mejor alternativa es insertar los estilos en la cabecera del documento y luego usar referencias para afectar los elementos
HTML correspondientes.

```
 <style>
         p { font-size: 20px }
</style>
```

El elemento <style> permite a los desarrolladores agrupar estilos CSS dentro del documento.

Este método sería bueno si sólo tuviéramos un documento en nuestra página, pero como habitualmente tendremos páginas formadas por varios documentos.

Archivos externos: Es el método más recomendable. La solución es mover todos los estilos a un archivo externo, y luego utilizar el elemento <link> para insertar este archivo dentro de cada documento que los necesite. Este método nos permite cambiar los estilos por completo, simplemente, incluyendo un archivo diferente. También nos permite modificar o adaptar nuestros documentos a cada circunstancia o dispositivo.

```
<link rel="stylesheet" href="./misestilos.css">
```

Con la línea anterior le decimos al navegador que cargue el archivo misestilos.css, que contendrá todos los estilos necesarios para presentar el documento en pantalla.

Funcionamiento básico de las reglas CSS

CSS define una serie de términos que permiten describir cada una de las partes que componen los estilos CSS. El siguiente esquema muestra las partes que forman un estilo CSS muy básico:

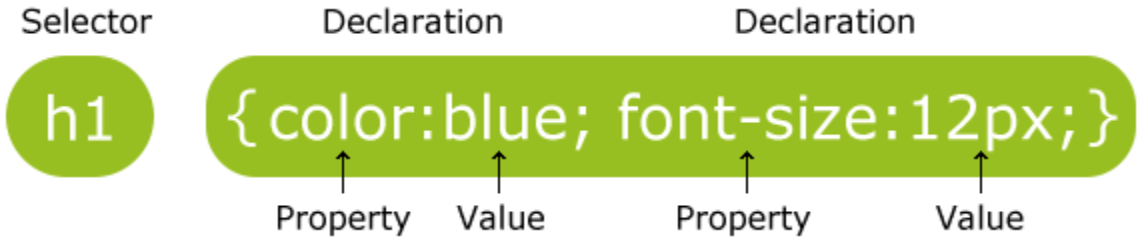

Imagen: Regla CSS

Los diferentes términos se definen a continuación:

Regla

Cada uno de los estilos que componen una hoja de estilos CSS. Cada regla está compuesta por una parte denominada "selectores", un símbolo de "llave de apertura" ({), otra parte denominada "declaraciones" y, por último, un símbolo de "llave de cierre" (}).

- ✓ Selector: indica el elemento o elementos HTML a los que se aplica la regla CSS.
- ✓ Declaración: especifica los estilos que se aplican a los elementos. Está compuesta por una o más propiedades CSS.
- ✓ Propiedad: permite modificar el aspecto de una característica del elemento. Valor: indica el nuevo valor de la característica modificada en el elemento.

Un archivo CSS puede contener infinitas reglas CSS, cada regla puede contener infinitos selectores y cada declaración puede estar formada por un número infinito de pares propiedad/valor.

Selectores

Para crear diseños web profesionales, es imprescindible conocer y dominar los selectores de CSS. Como ya hemos comentado, una regla de CSS está formada por una parte llamada "selector" y otra parte llamada "declaración". La declaración indica "qué hay que hacer" y el selector indica "a quién hay que hacérselo". Por lo tanto, los selectores son imprescindibles para aplicar de forma correcta los estilos CSS en una página.

A un mismo elemento HTML se le pueden asignar infinitas reglas CSS y cada regla CSS puede aplicarse a un número infinito de elementos. En otras palabras, una misma regla puede aplicarse sobre varios selectores y un mismo selector se puede utilizar en varias reglas.

A continuación, veremos algunos de los selectores más importantes:

Selector universal *

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

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

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 píxeles. Para evitar tener que repetir estilos constantemente, podemos utilizar el selector universal.

Con la regla indicada anteriormente, nos aseguramos de que todo elemento tendrá un margen externo e interno de 0 píxeles. De ahora en adelante, sólo necesitaremos modificar los márgenes de los elementos que queremos que sean mayores que cero.

Selector de tipo o etiqueta

Selecciona todos los elementos de la página cuya etiqueta HTML coincide con el valor del selector. El siguiente ejemplo selecciona todos los párrafos de la página:

```
p { ... }
```

Si se quiere aplicar los mismos estilos a dos etiquetas diferentes, se pueden encadenar los selectores. Para ello, se incluyen todos los selectores separados por una coma (,).

```
h1, h2, h3 { color: #8A8E27; font-weight: normal; }
```

Selector descendente

Selecciona los elementos que se encuentran dentro de otros elementos. Un elemento es descendiente de otro cuando se encuentra entre las etiquetas de apertura y de cierre del otro elemento.

El selector del siguiente ejemplo selecciona todos los elementos de la página que se encuentren dentro de un elemento <p>:

```
p span { color: red; }
```

Si el código HTML de la página es el siguiente:

```
<p>
<span>texto1</span>
<a href=""><span>texto2</span></a>
</p>
```

El selector p span selecciona tanto texto1 como texto2. El motivo es que en el selector descendente, un elemento no tiene que ser "hijo directo" de otro. La única condición es que un elemento debe estar dentro de otro elemento. A los elementos de la página que no están dentro de un elemento <p>, no se les aplica la regla CSS anterior.

Selector de clase

Una de las soluciones más sencillas para aplicar estilos a un solo elemento de la página, consiste en utilizar el atributo class de HTML sobre ese elemento para indicar directamente la regla CSS que se le debe aplicar.

```
<p class="destacado">Lorem ipsum dolor sit amet...</p>
```

A continuación, se crea en el archivo CSS una nueva regla llamada destacado con todos los estilos que se van a aplicar al elemento. Para que el navegador no confunda este selector con los otros tipos de selectores, se prefija el valor del atributo class con un punto (.), tal y como muestra el siguiente ejemplo:

```
.destacado { color: red; }
```

En ocasiones, es necesario restringir el alcance del selector de clase.

```
<p class="destacado">Lorem ipsum dolor sit amet...</p>
<p>sed lacus et <a href="#" class="destacado">est adipiscing</a>
accumsan</p>
```

¿Cómo es posible aplicar estilos solamente al párrafo cuyo atributo class sea igual a destacado? Combinando el selector de tipo y el selector de clase, se obtiene un selector mucho más específico:

```
p.destacado { color: red }
```

El selector **p.destacado** se interpreta como "aquellos elementos de tipo <p> que dispongan de un atributo class con valor destacado".Es posible aplicar los estilos de varias clases CSS sobre un mismo elemento. La sintaxis es similar, pero los diferentes valores del atributo class

se separan con espacios en blanco.

En el siguiente ejemplo:

```
<p class="especial destacado error">Párrafo de texto...</p>
```

Al párrafo anterior se le aplican los estilos definidos en las reglas .especial, .destacado y .error.

Selector de ID #

En ocasiones, es necesario aplicar estilos CSS a un único elemento de la página. Aunque puede utilizarse un selector de clase para aplicar estilos a un único elemento, existe otro selector más eficiente en este caso.El selector de ID permite seleccionar un elemento de la página a través del valor de su atributo id.

Este tipo de selectores sólo seleccionan un elemento de la página, ya que el valor del atributo id no se puede repetir en dos elementos diferentes de una misma página.

La sintaxis de los selectores de ID es muy parecida a la de los selectores de clase, salvo que se utiliza el símbolo de la almohadilla (#), en lugar del punto (.), como prefijo del nombre de la regla CSS:

```
#destacado { color: red; }
<p>Primer párrafo</p>
<p id="destacado">Segundo párrafo</p>
<p>Tercer párrafo</p>
```

En el ejemplo anterior, el selector `#destacado`, solamente selecciona el segundo párrafo (cuyo atributo id es igual a destacado). La recomendación general es la de utilizar el selector de ID cuando se quiere aplicar un estilo a un solo elemento específico de la página, y utilizar el selector de clase cuando se quiere aplicar un estilo a varios elementos diferentes de la página HTML.

Selector de hijos >

Se trata de un selector similar al selector descendente, pero muy diferente en su funcionamiento. Se utiliza para seleccionar un elemento que es hijo directo de otro elemento y se indica mediante el "signo de mayor que" (>):

```
p > span { color: blue; }
<p><span>Texto1</span></p>
<p><a href="#"><span>Texto2</span></a></p>
```

En el ejemplo anterior, el selector **p > span** se interpreta como "cualquier elemento que sea hijo directo de un elemento <p>", por lo que el primer elemento cumple la condición del selector. Sin embargo, el segundo elemento no la cumple porque es descendiente, pero no es hijo directo de un elemento <p>.

Selector adyacente +

El selector adyacente utiliza el signo + y su sintaxis es:

```
elemento1 + elemento2 { ... }
```

La explicación del comportamiento de este selector no es sencilla, ya que selecciona todos los elementos de tipo elemento2 que cumplan las dos siguientes condiciones:

- ✓ elemento1 y elemento2 deben ser hermanos, por lo que su elemento padre debe ser el mismo.
- ✓ elemento2 debe aparecer inmediatamente después de elemento1 en el código HTML de la página.

En el siguiente ejemplo:

```
h1 + h2 { color: red }
<body>
      <h1>Titulo1</h1>
      <h2>Subtítulo</h2> ...
      <h2>Otro subtítulo</h2>
      ...
</body>
```

Los estilos del selector h1 + h2 se aplican al primer elemento <h2> de la página, pero no al segundo <h2>, ya que:

El elemento padre de <h1> es <body>, el mismo padre que el de los dos elementos <h2>.Así, los dos elementos <h2> cumplen la primera condición del selector adyacente.

El primer elemento <h2> aparece en el código HTML justo después del elemento <h1>, por lo que, este elemento <h2> también cumple la segunda condición del selector adyacente.Por el contrario, el segundo elemento <h2> no aparece justo después del elemento <h1>, por lo que no cumple la segunda condición del selector adyacente y, por tanto, no se le aplican los estilos de h1 + h2.

Selector de atributos

Los selectores de atributos permiten seleccionar elementos HTML en función de sus atributos y/o valores de esos atributos.

Los cuatro tipos de selectores de atributos son:

- ✓ [nombre_atributo], selecciona los elementos que tienen establecido el atributo llamado nombre_atributo independientemente de su valor.
- ✓ [nombre_atributo=valor], selecciona los elementos que tienen establecido un atributo llamado nombre_atributo con un valor igual a valor.
- ✓ [nombre_atributo~=valor], selecciona los elementos que tienen establecido un atributo llamado nombre_atributo y al menos uno de los valores del atributo es valor.
- ✓ [nombre_atributo|=valor], selecciona los elementos que tienen establecido un atributo llamado nombre_atributo, cuyo valor es una serie de palabras separadas con guiones, pero que comienza con valor. Este tipo de selector sólo es útil para los atributos de tipo ***lang*** que indican el idioma del contenido del elemento.

A continuación, se muestran algunos ejemplos de estos tipos de selectores:

/* Se muestran de color azul todos los enlaces que tengan un atributo "class", independientemente de su valor */

```
a[class] { color: blue; }
```

/* Se muestran de color azul todos los enlaces que tengan un atributo "class" con el valor "externo" */

```
a[class="externo"] { color: blue; }
```

/* Se muestran de color azul todos los enlaces que apunten al sitio "http://www.ejemplo.com" */

```
a[href="http://www.ejemplo.com"] { color: blue; }
```

/* Se muestran de color azul todos los enlaces que tengan un atributo "class" en el que al menos uno de sus valores sea "externo" */

```
a[class~="externo"] { color: blue; }
```

/* Selecciona todos los elementos de la página cuyo atributo "lang" sea igual a "en", es decir, todos los elementos en inglés */

```
*[lang=en] { ... }
```

/* Selecciona todos los elementos de la página cuyo atributo "lang" empiece por "es", es decir, "es", "es-ES", "es-AR", etc. */

```
*[lang|="es"] { color : red }
```

Pseudo-clases

El concepto pseudo-clase se introdujo para permitir la selección de elementos sobre la base de la información que se encuentra fuera de la estructura del documento, o que no se puede expresar con los otros selectores simples.Una pseudo-clase se compone siempre de "dos puntos" (:) seguido del nombre de la pseudo-clase y, opcionalmente, por un valor entre paréntesis.

Las pseudo-clases pueden ser dinámicas, es decir, un elemento puede adquirir o perder una pseudo-clase, mientras que un usuario interactúa con el documento.Ya en CSS2 se definieron una serie de pseudo-clases dinámicas que nos permitían cambiar los estilos de los enlaces en función de su estado o de cómo interactuara el usuario con ellos:

- ✓ :**link** permite aplicar estilos para los enlaces que aún no han sido visitados.
- ✓ :**visited** aplica estilos a los enlaces que han sido visitados anteriormente.
- ✓ :**focus** estilos que se aplican al enlace cuando este tiene el foco (acepta eventos de ratón o de teclado).
- ✓ :**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).

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.

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

En este ejemplo se elimina el subrayado del enlace cuando se sitúa el ratón sobre él.

Pseudo-elementos

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.
- ✓ El pseudo-elemento :**first-letter** permite aplicar estilos a la primera letra del texto.

CSS también define dos pseudo-elementos :**before** y :**after** que nos permiten insertar contenidos antes o después de un elemento determinado. Para ello utilizaremos la propiedad CSS content, en la que indicaremos los contenidos que serán insertados.

```
a:after { content: " (" attr(href) ") "; }
```

El código CSS anterior añade después de cada enlace de la página un texto formado por la dirección a la que apunta el enlace mostrada entre paréntesis. Si se quiere añadir las direcciones antes de cada enlace, se puede utilizar el pseudo- elemento :before:

```
a:before { content: " (" attr(href) ") "; }
```

Propiedades CSS

La potencia de CSS la encontramos en las propiedades disponibles para modificar el aspecto de los elementos a los que se las aplicamos. Evidentemente, no podemos ver en detalle todas las propiedades de CSS, pero sí que indicaremos a continuación, las propiedades de uso más habitual.

CSS que afecta al texto	**color**: red; **font-family**: 20px; **font-size**: 20px; **font-weight**: bold; **text-align**: center; **line-height**: 50px; **letter-spacing**: 2px; **word-spacing**: 5px;	Color del texto Tipografía utilizada. Tamaño del texto (expresado en píxeles "10px", en tamaño original de la fuente "2em"..) Estilo de la fuente ("bold", "normal", "italic"..) Alineación de un texto dentro de un bloque (center, left, right o justify). Altura de una línea de texto. Sirve para centrar un texto verticalmente, indicando como valor la altura del bloque. Espacio que hay entre cada letra (se puede definir en píxeles "px" o en cantidades relativas "em"). Espacio que hay entre palabras (se puede definir en píxeles "px" o en cantidades relativas "em").
CSS que afecta a los bloques	**background-color**: red; **padding**: 20px; **margin**: 20px; **position**: relative; **left**: 10px; **top**: 10px; **float**: left **clear**: both	Color de fondo de un bloque. Espacio que hay entre el contenido de un bloque y su borde (sería el espacio interno). Espacio que hay entre bloques (espacio externo). Define el tipo de posicionamiento de un bloque (relative, absolute o fixed). Posición horizontal de un elemento. Posicion vertical de un elemento. Posiciona bloques a la izquierda (left) o derecha (right) de otros. Elimina los float declarados con anterioridad y que aún perduran.

CSS que afectan a los enlaces	**text-decoration**: none;
	cursor: pointer;
	a {
	a:hover {
		Elimina el subrayado de los enlaces (o la viñetas en las listas).
		Transforma el cursor en la mano con el dedo extendido (al usar junto a **a:hover**).
		Selector de CSS que identifica a los enlaces que contenga la página.
		Selector de CSS que identifica el momento en el que el cursor se coloca encima de un enlace.

Imagen: Propiedades CSS más utilizadas

Validador CSS

Al igual que ocurre con html, disponemos de un validador para css que nos informará de los errores que tenemos en nuestros archivos de estilos. Este validador se encuentra en la url http://jigsaw.w3.org/css-validator/. Es altamente recomendable pasar el validador antes de probar nuestro sitio web, ya que, muchas veces los estilos no se aplicarán como nosotros esperamos porque tenemos errores, y el validador nos puede ahorrar mucho tiempo a la hora de encontrar dónde está el problema.

Página web de ejemplo

A continuación se muestra un ejemplo de una página web real a la cual le aplicaremos los estilos necesarios para que se visualice como nos interesa. En concreto, el ejemplo que veremos a continuación contiene el diseño de un blog muy simple.

```
<!doctype html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Blog</title>
  <meta name="description" content="HTML5">
  <meta name="author" content="Maribel Campos”>
  <link rel="stylesheet" href="css/estilos.css">
</head>
<body>
  <div id="page">
    <div id="header">
      <h1>Este es el título principal del sitio web</h1>
    </div>
    <div id="nav">
      <ul>
          <li>principal</li>
          <li><a href=”#”>fotos</a></li>
          <li><a href=”#”>videos</a></li>
          <li><a href=”#”>contacto</a></li>
     </ul>
    </div>
    <div id="content">
      <div class="article">
          <div class="article_header">
                  <h2>Título del mensaje uno</h2>
                  <h3>Subtítulo del mensaje uno</h3>
                  <p class="time">publicado 10-10-2020</p>
          </div><p>Este es el texto de mi primer mensaje</p>
          <div class="figure">
                  <img src="imgs/figure1.png" alt="W3C WAI-ARIA">
                  <p class="figcaption">Esta es la imagen del primer mensaje</p>
          </div>
          <div class="article_footer">
          <p>comentarios (0)</p>
       </div>
      </div>
   <div class="article">
          <div class="article_header">
                  <h2>Título del mensaje dos</h2>
                  <h3>Subtítulo del mensaje dos</h3>
                  <p class="time">publicado 15-10-2020</p>
          </div>
          <p>Este es el texto de mi segundo mensaje</p>
          <div class="article_footer">
          <p>comentarios (0)</p>
        </div>
       </div>
```

```
    </div>
    <div id="aside">
      <blockquote>Mensaje número uno</blockquote>
      <blockquote>Mensaje número dos</blockquote>
    </div>
    <div id="footer">
          <p>Derechos Reservados &copy; 2018-2020</p>
    </div>
  </div>
</body>
</html>
```

Copia y pega el código en atom. Muestra el contenido con un navegador.

Estilos y estructura

A pesar de que cada navegador garantiza estilos por defecto para cada uno de los elementos HTML, estos estilos no necesariamente satisfacen los requerimientos de cada diseñador. Por lo tanto, los diseñadores y desarrolladores deben aplicar sus propios estilos para obtener la organización y el efecto visual que realmente desean.

Con respecto a la estructura, básicamente, cada navegador ordena los elementos por defecto de acuerdo a su tipo: block (bloque) o inline (en línea). Como hemos comentado anteriormente, esta clasificación está asociada con la forma en que los elementos son mostrados en pantalla. Los elementos block son posicionados uno sobre otro hacia abajo en la página, mientras que, los elementos inline son posicionados, uno al lado del otro en la misma línea, sin ningún salto de línea, a menos que ya no haya más espacio horizontal para ubicarlos.

La página web de ejemplo está basada en el diseño web típico anterior, el cual incluía barras horizontales y dos columnas en el medio. Debido a la forma en que los navegadores muestran estos elementos por defecto, el resultado en la pantalla está muy lejos de nuestras expectativas.

Crea un en tu estrucura de carpetas, dentro de la carpeta css un nuevo archivo llamado “estilos.css”. De momento estará en blanco pero poco a poco le iremos dando forma.

Modelos de caja

Para aprender cómo podemos crear nuestra propia organización de los elementos en pantalla, primero debemos entender cómo los navegadores procesan el código HTML.

Los navegadores consideran cada elemento como una caja. Una página web es en realidad un grupo de cajas ordenadas siguiendo ciertas reglas. Estas reglas se establecen por los estilos por defecto que aplican los navegadores a cada elemento de la página, o por los estilos que aplican los diseñadores mediante CSS. Combinando las propiedades CSS en los elementos de la página podemos obtener la organización deseada. Estas propiedades, tienen que ser combinadas para formar reglas que luego serán usadas para obtener la correcta disposición en pantalla. A la combinación de estas reglas la llamaremos disposición de la página (Layout). Todas estas reglas aplicadas juntas, constituyen lo que se llama un modelo de caja.

Gracias a las etiquetas y a los estilos CSS fue posible reemplazar la maquetación mediante tablas y separar la estructura HTML de la presentación. Con elementos 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 dispone de propiedades específicas que nos permiten organizar las cajas. Estas propiedades son lo suficientemente poderosas como para crear un modelo de caja que se conoce como Modelo de Caja Tradicional. A continuación, aplicaremos los estilos necesarios a nuestro diseño básico para que adopte la forma que nos interesa.

Creando la disposición de nuestra página con CSS

P2.1. Mi primera hoja de estilos

Copia cada uno de los estilos que se muestran en este punto del tema. Ve comprobando cada uno de los cambios que se producen en la página web
Después de aplicar todos los estilos, la página debe de mostrarse como se indica al final de este documento.
Sube a aules tu archivo html y tu css y una captura de pantalla del resultado.

Márgenes por defecto

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

Utilizando el selector universal establecemos en 0px los márgenes por defecto de todos los elementos de nuestra página. De esta forma, sólo necesitaremos modificar los márgenes de los elementos que queremos que sean mayores que cero.

Es importante recordar que en HTML cada elemento es considerado como una caja. El

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. 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 píxeles, por ejemplo, la caja tendría un espacio de 5 píxeles de ancho en todo su contorno. Esto significa que la caja estaría separada de sus vecinas por 5 píxeles.

Nueva jerarquía para cabeceras

En nuestro documento usamos elementos <h1>, <h2> y <h3> para declarar títulos de diferentes niveles. Para personalizar los estilos de estos elementos utilizaremos las siguientes reglas CSS:

```
h1 { font: bold 2em verdana, sans-serif; }
h2 { font: bold 1.5em verdana, sans-serif; }
h3 { font: bold 1em verdana, sans-serif; }
```

La propiedad **font**, asignada a los elementos <h1>, <h2> y <h3>, 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 y font-family**. Con las reglas anteriores estamos cambiando el grosor, tamaño y el tipo de letra del texto.

Centrando el cuerpo

El primer elemento que forma 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; }
```

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 de estilo anterior, 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. En nuestro documento básico agregamos un elemento para agrupar todas las cajas dentro del cuerpo. Este será la caja principal para la construcción de nuestro modelo de caja. De este modo, modificando el tamaño de este elemento lo hacemos al mismo tiempo para todos los demás:

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

Esta regla aplica tres estilos a la caja principal:

- El primer estilo establece un valor fijo de 960 píxeles para el ancho de la caja.
- El segundo estilo es parte de lo que llamamos el **Modelo de Caja Tradicional**. En la regla anterior, especificamos que el contenido del cuerpo sería centrado horizontalmente con el estilo *text-align: center*. Pero esto sólo afecta al 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 sólo escribimos los primeros dos parámetros, el resto tomará los mismos valores. En nuestro ejemplo, estamos usando esta técnica. El estilo margin: 15px auto asigna 15 píxeles al margen superior e inferior del elemento que está afectando, y declara como automático el tamaño de los márgenes de izquierda y derecha. De esta manera, habremos generado un espacio de 15 píxeles 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 , centrando el contenido en pantalla.
- El último estilo ***(text-align: left***) lo necesitamos para 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> será asignado a cada uno de sus hijos. Debemos retornar este estilo a su valor por defecto para el resto del documento. El resultado final es que el contenido del cuerpo está centrado, pero el contenido de la caja principal estará alineado nuevamente a la izquierda, por lo tanto, todo el resto del código HTML dentro de esta caja hereda este estilo.

La cabecera

A continuación del elemento page, tenemos el . Este elemento contiene el título principal de nuestra página web y estará ubicado en la parte superior de la pantalla.

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 píxeles, porque su padre es la caja principal que fue previamente configurada con este tamaño. Por lo tanto, lo único que haremos será asignar estilos que nos permitirán reconocer el elemento cuando es presentado en pantalla.

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

En esta regla le otorgamos al un fondo amarillo, un borde sólido de 1 píxel y un margen interior de 20 píxeles.

Barra de navegación

Siguiendo al elemento se encuentra el elemento , 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 lo tanto, lo único que nos queda por hacer es mejorar su aspecto en pantalla. Agregaremos un fondo gris y un pequeño margen interno para separar las opciones del menú del borde del elemento:

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

Dentro de la barra de navegación hay una lista desordenada. 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 usando el selector #nav li, y luego asignamos a todos ellos el estilo display: inline para convertirlos en elementos inline. A diferencia de los elementos block, los elementos afectados por el parámetro inline no generan ningún salto de línea.

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*.

En cada elemento de la lista aparece un enlace que nos llevará a otra sección de la página, pero en la sección actual no existe enlace, esto hace que al colocar el ratón sobre el elemento de menú de la sección actual aparezca un cursor inadecuado. Para solucionar esto, utilizamos la propiedad cursor: default, que hará que en los elementos <li> aparezca siempre el cursor default.
Sólo nos queda mejorar un poco el aspecto de los enlaces del menú:

```
#nav a { text-decoration:none; }
#nav a:hover { color: white; }
```

Por defecto, los enlaces aparecen subrayados, con el estilo **text-decoration:none** eliminamos el subrayado de los mismos. Además, utilizamos la pseudoclase **a:hover** para cambiar el estilo de los enlaces cuando el ratón está sobre ellos. En este caso, cambiamos el color de la fuente a blanco **color: white**.

Sección principal de contenidos y barra lateral

Los siguientes elementos estructurales en nuestro ejemplo son dos cajas que nos interesa que se dispongan 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 para crear estas cajas son y .

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

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 anteriores, 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 estará situado a la izquierda de la pantalla con un tamaño de 660 píxeles, más 40 píxeles de margen, ocupando un espacio total de 700 píxeles de ancho.

La propiedad float del elemento 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 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 así la segunda columna de nuestro diseño.

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

Es importante que tengamos en cuenta que el tamaño de un elemento y sus márgenes se suman para obtener el valor real ocupado en pantalla. Si tenemos un elemento de 200 píxeles de ancho y un margen de 10 píxeles a cada lado, el área real ocupada por el elemento será de 220 píxeles.

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 se sumarán al ancho del elemento para obtener el valor real cuando el elemento es mostrado en pantalla.

El tamaño real de un elemento se calculará con la fórmula: tamaño + márgenes + márgenes internos + bordes.

Pie de página

Para finalizar la aplicación del Modelo de Caja Tradicional, tenemos que aplicar otra propiedad CSS al elemento .

Esta propiedad devuelve al documento su flujo normal y nos permite posicionar este elemento debajo del anterior, en lugar de a su lado.

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

La regla anterior declara un borde de 2 píxeles en la parte superior del elemento , un margen interno (padding) de 20 píxeles, y centra el texto dentro del elemento. Así mismo, restaura el flujo normal 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**, que 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.

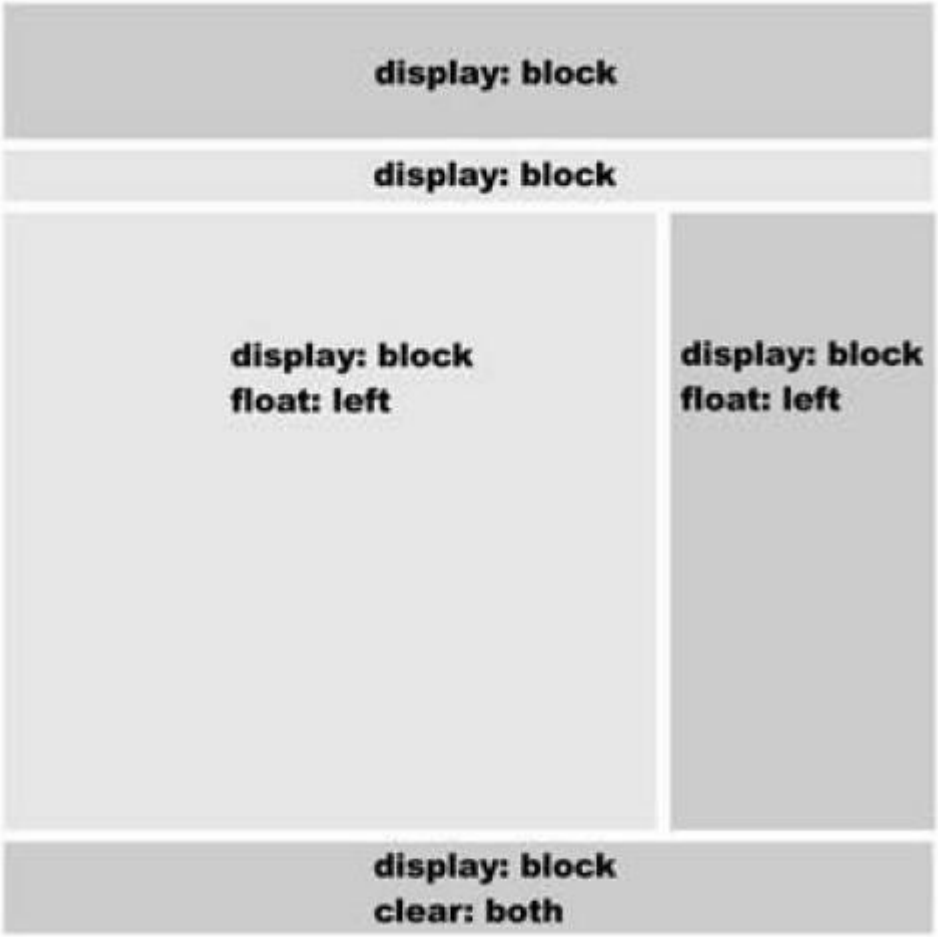

Imagen: Representación visual del modelo de caja tradicional

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 imagen anterior 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 a continuación 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 contigua 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.

```
.article { background: #FFFBCC; border: 1px solid #999999; padding: 20px;
margin-bottom: 15px; }
.article_header { border-bottom: 1px solid #999999; }
.article_footer { text-align: right; }
.time { color: #999999; }
.figure { border: 1px double #999999; padding: 5px; }
.figure .figcaption { font: italic 0.6em verdana, sans-serif; }
```

La primera regla referencia los elementos de la clase article y les otorga algunos estilos básicos (color de fondo, un borde sólido de 1 píxel, margen interno y margen inferior). El margen inferior de 15 píxeles tiene el propósito de separar un elemento article del siguiente verticalmente.

Cada elemento article cuenta con un elemento article_header, que muestra los títulos y fecha de publicación del artículo, y un elemento article_footer, que muestra los comentarios recibidos. El texto de los elementos article_footer aparecerá alineado a la derecha, mientras que los elementos article_header tendrán un borde inferior.

Los elementos time tendrán el color de la fuente #999999.En el primer artículo, además, tenemos un elemento que contiene una imagen y un elemento que nos muestra un texto explicativo de la misma. Por lo tanto, crearemos dos reglas para aplicar estilos a dichos elementos. En concreto, se pondrá un borde a los elementos figure y se cambiará la fuente a los elementos figcaption.

Si hemos creado bien la hoja de estilos, el resultado de la página web creada al principio del documento tiene que ser como se muestra a continuación: