

Front-end

Estructura básica HTML5 y CSS3



Instructor: Gustavo Adolfo Rodríguez Q.
garodriguez335@misena.edu.co
Front-end

Componentes básicos

HTML5 provee básicamente tres características: estructura, estilo y funcionalidad. Nunca fue declarado oficialmente pero, incluso cuando algunas APIs (Interface de Programación de Aplicaciones) y la especificación de CSS3 por completo no son parte del mismo, HTML5 es considerado el producto de la combinación de HTML, CSS y Javascript. Estas tecnologías son altamente dependientes y actúan como una sola unidad organizada bajo la especificación de HTML5. HTML está a cargo de la estructura, CSS presenta esa estructura y su contenido en la pantalla y Javascript hace el resto que (como veremos más adelante) es extremadamente significativo. Más allá de esta integración, la estructura sigue siendo parte esencial de un documento. La misma provee los elementos necesarios para ubicar contenido estático o dinámico, y es también una plataforma básica para aplicaciones. Con la variedad de dispositivos para acceder a Internet y la diversidad de interfaces disponibles para interactuar con la web, un aspecto básico como la estructura se vuelve parte vital del documento. Ahora la estructura debe proveer forma, organización y flexibilidad, y debe ser tan fuerte como los fundamentos de un edificio.

Para trabajar y crear sitios webs y aplicaciones con HTML5, necesitamos saber primero cómo esa estructura es construida. Crear fundamentos fuertes nos ayudará más adelante a aplicar el resto de los componentes para aprovechar completamente estas nuevas tecnologías. Por lo tanto, empecemos por lo básico, paso a paso. En este primer capítulo aprenderá cómo construir una plantilla para futuros proyectos usando los nuevos elementos HTML introducidos en HTML5.

Hágalo usted mismo: Cree un archivo de texto vacío utilizando su editor de textos favorito para probar cada código presentado en este capítulo. Esto lo ayudará a recordar las nuevas etiquetas HTML y acostumbrarse a ellas.

Conceptos básicos: Un documento HTML es un archivo de texto. Si usted no posee ningún programa para desarrollo web, puede simplemente utilizar el Bloc de Notas de Windows o cualquier otro editor de textos. El archivo debe ser grabado con la extensión *.html* y el nombre que desee (*por ejemplo, micodigo.html*).

Estructura global

Los documentos HTML se encuentran estrictamente organizados. Cada parte del documento está diferenciada, declarada y determinada por etiquetas específicas. En este documento vamos a ver cómo construir la estructura global de un documento HTML y los nuevos elementos semánticos incorporados en HTML5.

<!DOCTYPE>

En primer lugar necesitamos indicar el tipo de documento que estamos creando. Esto en HTML5 es extremadamente sencillo:

```
<!DOCTYPE html> //primera línea de código
```

IMPORTANTE: Esta línea debe ser la primera línea del archivo, sin espacios o líneas que la precedan. De esta forma, el modo estándar del navegador es activado y las incorporaciones de HTML5 son interpretadas siempre que sea posible, o ignoradas en caso contrario.

Hágalo usted mismo: Puede comenzar a copiar el código en su archivo de texto y agregar los próximos a medida que los vamos avanzando.

```
<html>
```

Luego de declarar el tipo de documento, debemos comenzar a construir la estructura HTML. Como siempre, la estructura tipo árbol de este lenguaje tiene su raíz en el elemento <html>. Este elemento envolverá al resto del código:

```
<!DOCTYPE html>
<html lang="es">
</html>
```

El atributo lang en la etiqueta de apertura <html> es el único atributo que necesitamos especificar en HTML5. Este atributo define el idioma humano del contenido del documento que estamos creando, en este caso es por español.

Conceptos básicos: HTML usa un lenguaje de etiquetas para construir páginas web. Estas etiquetas HTML son palabras clave y atributos rodeados de los signos mayor y menor (por ejemplo, <html lang="es">). En este caso, html es la palabra clave y lang es el atributo con el valor es. La mayoría de las etiquetas HTML se utilizan en pares, una etiqueta de apertura y una de cierre, y el contenido se declara entre ellas. En nuestro ejemplo, <html lang="es"> indica el comienzo del código HTML y </html> indica el

final. Compare las etiquetas de apertura y cierre y verá que la de cierre se distingue por una barra invertida antes de la palabra clave (por ejemplo, </html>). El resto de nuestro código será insertado entre estas dos etiquetas: <html> ... </html>.

IMPORTANTE: HTML5 es extremadamente flexible en cuanto a la estructura y a los elementos utilizados para construirla. El elemento <html> puede ser incluido sin ningún atributo o incluso ignorado completamente. Con el propósito de preservar compatibilidad (y por algunas razones extras que no vale la pena mencionar aquí) le recomendamos que siga algunas reglas básicas. En este libro vamos a enseñarle cómo construir documentos HTML de acuerdo a lo que nosotros consideramos prácticas recomendadas. Para encontrar otros lenguajes para el atributo lang puede visitar el siguiente enlace: www.w3schools.com/tags/ref_language_codes.asp.

<head>

Continuemos construyendo nuestra plantilla. El código HTML insertado entre las etiquetas <html> tiene que ser dividido entre dos secciones principales. Al igual que en versiones previas de HTML, la primera sección es la cabecera y la segunda el cuerpo. El siguiente paso, por lo tanto, será crear estas dos secciones en el código usando los elementos <head> y <body> ya conocidos. El elemento <head> va primero, por supuesto, y al igual que el resto de los elementos estructurales tiene una etiqueta de apertura y una de cierre:

```
<!DOCTYPE html>
<html lang="es">
  <head>
  </head>
</html>
```

La etiqueta no cambió desde versiones anteriores y su propósito sigue siendo exactamente el mismo. Dentro de las etiquetas <head> definiremos el título de nuestra página web, declararemos el set de caracteres correspondiente, proveeremos información general acerca del documento e incorporaremos los archivos externos con estilos, códigos Javascript o incluso imágenes necesarias para generar la página en la pantalla. Excepto por el título y algunos íconos, el resto de la información incorporada en el documento entre estas etiquetas es invisible para el usuario.

<body>

La siguiente gran sección que es parte principal de la organización de un documento HTML es el cuerpo. El cuerpo representa la parte visible de todo documento y es especificado entre etiquetas <body>. Estas etiquetas tampoco han cambiado en relación con versiones previas de HTML:

```

<!DOCTYPE html>
<html lang="es">
  <head>
  </head>

  <body>
  </body>
</html>

```

Conceptos básicos: Hasta el momento tenemos un código simple pero con una estructura compleja. Esto es porque el código HTML no está formado por un conjunto de instrucciones secuenciales. HTML es un lenguaje de etiquetas, un listado de elementos que usualmente se utilizan en pares y que pueden ser anidados (totalmente contenidos uno dentro del otro). En la primera línea del código del Listado 1-4 tenemos una etiqueta simple con la definición del tipo de documento e inmediatamente después la etiqueta de apertura <html lang="es">. Esta etiqueta y la de cierre </html> al final del listado están indicando el comienzo del código HTML y su final. Entre las etiquetas <html> insertamos otras etiquetas especificando dos importantes partes de la estructura básica: <head> para la cabecera y <body> para el cuerpo del documento. Estas dos etiquetas también se utilizan en pares. Más adelante en este capítulo veremos que más etiquetas son insertadas entre estas últimas conformando una estructura de árbol con <html> como su raíz.

<meta>

Es momento de construir la cabecera del documento. Algunos cambios e innovaciones fueron incorporados dentro de la cabecera, y uno de ellos es la etiqueta que define el juego de caracteres a utilizar para mostrar el documento. Ésta es una etiqueta <meta> que especifica cómo el texto será presentado en pantalla:

```

<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="iso-8859-1">
  </head>

  <body>
  </body>
</html>

```

La innovación de este elemento en HTML5, como en la mayoría de los casos, es solo simplificación. La nueva etiqueta <meta> para la definición del tipo de caracteres es más corta y simple. Por supuesto, podemos cambiar el tipo iso-8859-1 por el necesario para nuestros documentos y agregar otras etiquetas <meta> como description o keywords para definir otros aspectos de la página web, como es mostrado en el siguiente ejemplo:

```
<!DOCTYPE html>
<html lang="es">
  <head>

    <meta charset="utf-8">
    <meta name="description" content="Ejemplo de
HTML5">
    <meta name="keywords" content="HTML5, CSS3,
Javascript">

  </head>

  <body>
  </body>
</html>
```

Conceptos básicos: Hay varios tipos de etiqueta <meta> que pueden ser incluidas para declarar información general sobre el documento, pero esta información no es mostrada en la ventana del navegador, es solo importante para motores de búsqueda y dispositivos que necesitan hacer una vista previa del documento u obtener un sumario de la información que contiene. Como comentamos anteriormente, aparte del título y algunos íconos, la mayoría de la información insertada entre las etiquetas <head> no es visible para los usuarios. En el código del Listado 1-6, el atributo name dentro de la etiqueta <meta> especifica su tipo y content declara su valor, pero ninguno de estos valores es mostrado en pantalla. Para aprender más sobre la etiqueta <meta>, visite nuestro sitio web y siga los enlaces proporcionados para este capítulo. En HTML5 no es necesario cerrar etiquetas simples con una barra al final, pero recomendamos utilizarlas por razones de compatibilidad. El anterior código se podría escribir de la siguiente manera:

```
<!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" />
    </head>

    <body>
    </body>
</html>

```

<title>

La etiqueta <title>, como siempre, simplemente especifica el título del documento, y no hay nada nuevo para comentar:

```

<!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>
    </head>

    <body>
    </body>
</html>

```

Conceptos básicos: El texto entre las etiquetas <title> es el título del documento que estamos creando. Normalmente este texto es mostrado en la barra superior de la ventana del navegador.

<link>

Otro importante elemento que va dentro de la cabecera del documento es <link>. Este elemento es usado para incorporar estilos, códigos Javascript, imágenes o iconos es de archivos externos. Uno de los usos más comunes para <link> es la incorporación de archivos con estilos CSS, los cuales trabajaremos más adelante.

<!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>
  </body>
</html>

```

En HTML5 ya no se necesita especificar qué tipo de estilos estamos insertando, por lo que el atributo `type` fue eliminado. Solo necesitamos dos atributos para incorporar nuestro archivo de estilos: `rel` y `href`. El atributo `rel` significa “relación” y es acerca de la relación entre el documento y el archivo que estamos incorporando por medio de `href`. En este caso, el atributo `rel` tiene el valor `stylesheet` que le dice al navegador que el archivo `misestilos.css` es un archivo CSS con estilos requeridos para presentar la página en pantalla (en el próximo capítulo estudiaremos cómo utilizar estilos CSS). Conceptos básicos: Un archivo de estilos es un grupo de reglas de formato que ayudarán a cambiar la apariencia de nuestra página web (por ejemplo, el tamaño y color del texto). Sin estas reglas, el texto y cualquier otro elemento HTML sería mostrado en pantalla utilizando los estilos estándar provistos por el navegador. Los estilos son reglas simples que normalmente requieren solo unas pocas líneas de código y pueden ser declarados en el mismo documento. Como veremos más adelante, no es estrictamente necesario obtener esta información de archivos externos pero es una práctica recomendada. Cargar las reglas CSS desde un documento externo (otro archivo) nos permitirá organizar el documento principal, incrementar la velocidad de carga y aprovechar las nuevas características de HTML5. Con esta última inserción podemos considerar finalizado nuestro trabajo en la cabecera. Ahora es tiempo de trabajar en el cuerpo, donde la magia ocurre.

Importante: sigue paso a paso la guía y en el documento HTML que crees, agrega comentarios describiendo lo que entiendes de cada etiqueta.

Las etiquetas en HTML se agregan de la siguiente manera

```
<!-- Tu comentario -->
```


Estructura del cuerpo HTML5

La estructura del cuerpo, lo que hay entre las etiquetas `<body></body>` contiene la parte visible de un documento. Este es el código que producirá la página web.

En un inicio las tablas fueron una revolución un gran paso para la visualización de documentos y experiencia ofrecida a los usuarios. Gradualmente otros elementos fueron remplazando su función permitiendo lograr lo mismo con menos código, facilitando de este modo la creación, permitiendo portabilidad y ayudando al mantenimiento de sitios web.

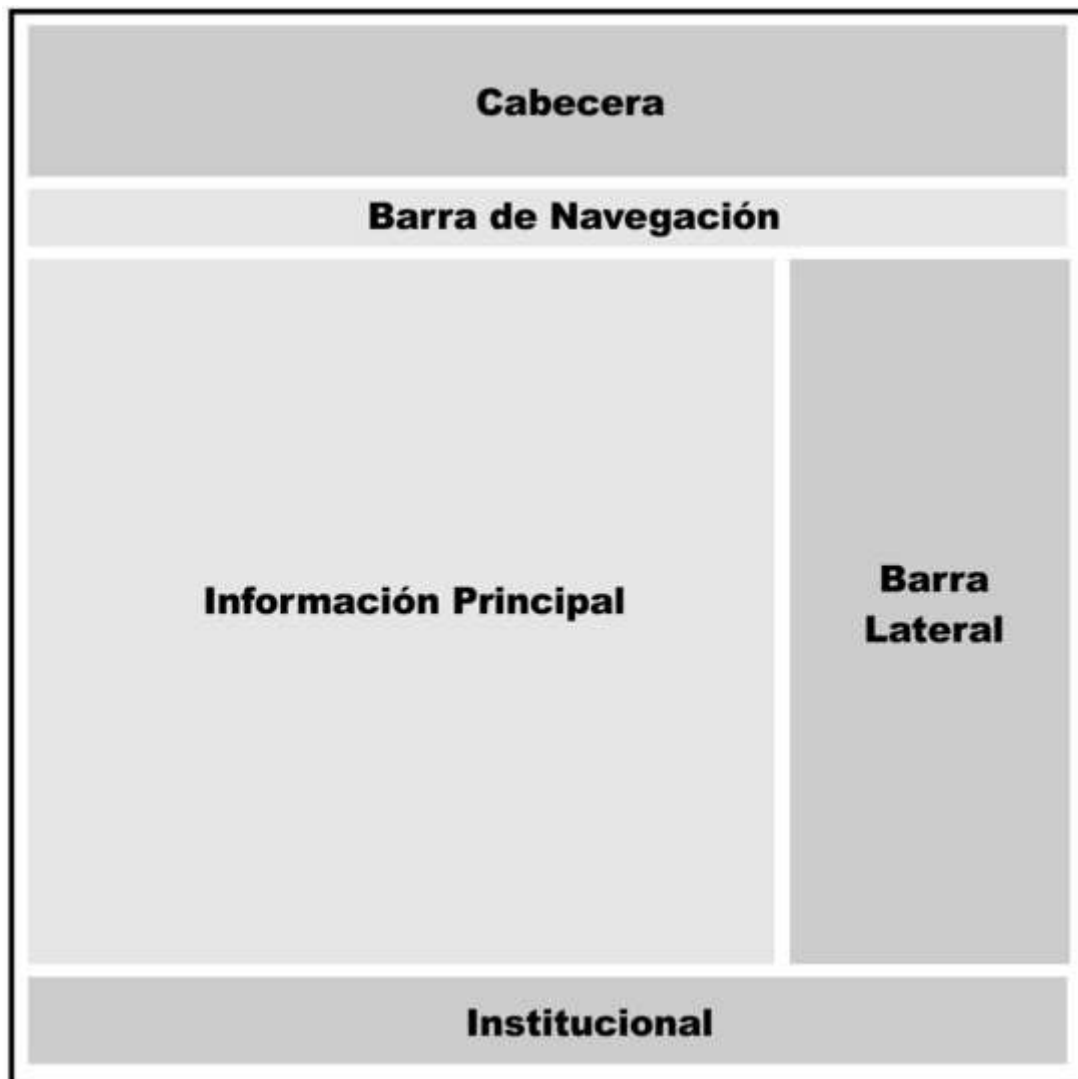
Una de las etiquetas que comenzó a dominar fue el `<div></div>`. Con el surgimiento de webs más interactivas y la integración de HTML, CSS y JavaScript, el uso del div se volvió una práctica común. El elemento div no informa más que una división en el documento, sin especificar qué clase de división es, cuál es su propósito o que contiene.

Luego de la revolución de los dispositivos móviles y los diferentes accesos a la web, la identificación de cada parte que compone un sitio se volvió crucial y relevante. Teniendo en cuenta todo lo expuesto anteriormente, **HTML5** incorpora nuevos elementos que ayudan a identificar cada sección del documento, permitiendo una mejor organización del mismo. En este momento comprendemos que la estructura principal ya no depende más de los elementos `<div>` o `<table>`.

Normalmente una página o aplicación web está dividida entre varias áreas visuales que mejoran la experiencia de usuario y facilitan su interactividad, los nuevos elementos HTML5 están directamente relacionados con estas áreas como lo veremos a continuación.

Organización

La siguiente imagen representa un diseño común encontrado en la mayoría de sitios web. A pesar de que cada diseñador crea sus propios diseños, como una generalidad podremos encontrar las siguientes secciones de un sitio.



- **Cabecera** usualmente encontramos el logo, títulos, subtítulos y una corta descripción de la página o el sitio web.
- **Barra de Navegación** lugar donde se ofrece el menú o lista de enlaces, que tienen como objetivo mejorar la navegación a través del sitio.
- El contenido más importante o relevante siempre se encuentra en la parte central del sitio. Este espacio puede estar dividido en varias columnas, como lo vemos en la imagen: **Información Principal** y la **Barra Lateral**, este espacio es extremadamente flexible y normalmente los diseñadores lo adaptan a sus necesidades, añadiendo más columnas o bloques, pequeños, largos, etc. Eso si siguiendo una diagramación.
- **Institucional** regularmente en los sitios web encontramos una sección, es denominada de esta manera ya que es el sitio donde se encuentra la información del sitio, el autor, la empresa, además de enlaces relacionados con reglas, términos, condiciones, y toda la información que se considere pertinente. Esta sección es el complemento de la cabecera.

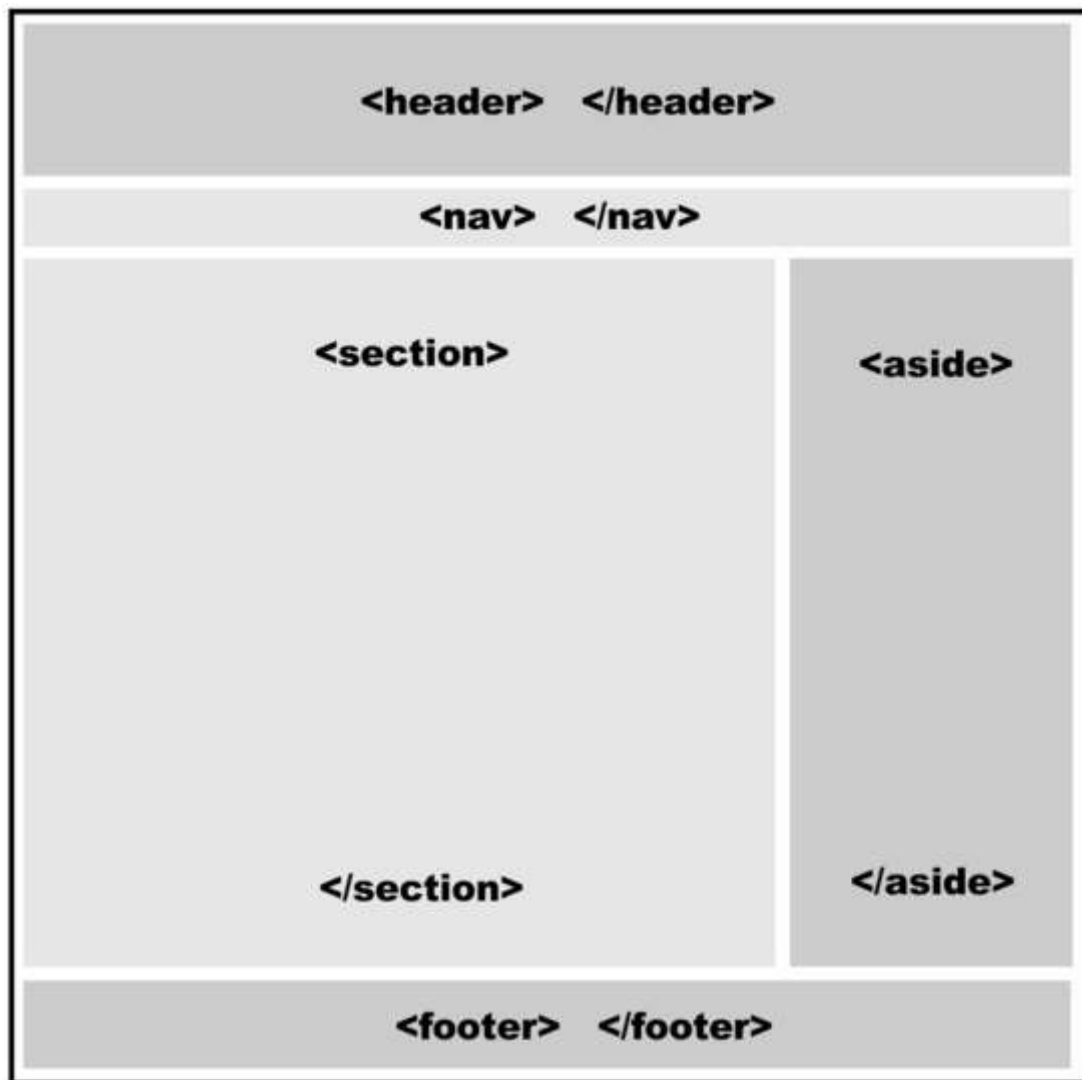
Un ejemplo de sitio web que usa esta estructura es la siguiente imagen:



En la anterior imagen podemos observar la representación de un blog normal, en este diseño podemos observar claramente el diseño considerado anteriormente.

Esta simple representación nos puede ayudar a entender que cada sección definida en un sitio web tiene un propósito y no está ahí solo por estar. HTML5 considera esta estructura básica y provee de nuevos elementos para diferenciar y aclarar cada una de sus partes.

A partir de ahora le podemos decir al navegador para qué es cada una de sus partes. En la siguiente imagen veremos el uso de las nuevas etiquetas para cada una de las secciones.



En la anterior imagen podemos observar la representación de un blog normal, en este diseño.

<header>

No debe ser confundido con la etiqueta **<head>** (usado antes para construir la cabecera del documento). Del mismo modo que el **<head>** el **<header>** pretende introducir información introductoria (títulos, subtítulos, logos), pero difiere con respecto a su alcance. Mientras el **<head>** tiene el propósito de proveer la información de todo el documento, el **<header>** es usado para proveer solo la información del cuerpo o secciones específicas dentro del cuerpo.

```
<!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>
  <header>
    <h1>Este es el título principal del sitio web</h1>
  </header>

</body>
</html>
```

Conceptos básicos: Entre el **<header>** podemos ver una etiqueta que probablemente no conoce, el **<h1>** este es un elemento viejo HTML usado para definir títulos, El número indica la importancia del título el **<h1>** es el más importante y el **<h6>** es el de menor importancia, por lo tanto el **<h1>** será usado para mostrar el título principal y los demás para los subtítulos o subtítulos internos. Más adelante veremos su uso en HTML5.

h1 example

h2 example

h3 example

h4 example

h5 example

h6 example

<nav>

Siguiendo con nuestro ejemplo, la siguiente sección es la barra de navegación, esta barra es generada en HTML5 con la etiqueta **<nav>**:

```
<!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>
  <header>
    <h1>Este es el título principal del sitio web</h1>
  </header>
  <nav>
    <ul>
      <li>principal</li>
      <li>fotos</li>
      <li>videos</li>
      <li>contacto</li>
    </ul>
  </nav>

</body>
</html>
```

La etiqueta **<nav>** está dentro de la etiqueta **<body>**, y esta después del cierre de la etiqueta **<header>**, esto es porque **<nav>** no es parte de la cabecera si no otra sección. Sabemos que la estructura y el orden de colocar los elementos HTML5 dependen de nosotros, esto significa que podemos crear un código versátil donde **<nav>** pueda estar en el interior del **<header>** o en cualquier otra parte del cuerpo. Sin embargo debemos recordar que estas etiquetas fueron creadas para brindar información a los navegadores y ayudar a cada nuevo programa o dispositivo a identificar las partes más relevantes del documento. Para ofrecer un código portable y comprensible. El elemento **<nav>** fue creado para ofrecer ayuda para la navegación, como menús principales o grandes bloques de enlaces y debería ser usado de esa manera.

Conceptos básicos: Entre la etiqueta **<nav>** hay dos elementos que son utilizados para crear una lista. El propósito de **** es definir la lista y anidado en ella están los elementos ****, estos son usados para definir los ítems de la lista.

<section>

Siguiendo con nuestro ejemplo encontramos las columnas **Información principal** y **Barra lateral**. Como comentamos anteriormente **Información principal** contiene la información más relevante del documento y puede ser encontrada en diferentes formas como por ejemplo dividida en bloques o columnas. Debido a que el propósito de esta área es más general se denomina **<section>**:

```
<!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>
  <header>
    <h1>Este es el título principal del sitio web</h1>
  </header>
  <nav>
    <ul>
      <li>principal</li>
      <li>fotos</li>
      <li>videos</li>
      <li>contacto</li>
    </ul>
  </nav>
  <section>

  </section>

</body>
</html>
```

Al igual que la **Barra de navegación**, la columna de **Información principal** es una sección aparte, por este motivo va debajo de la etiqueta de cierre **</nav>**.

Importante: usted podrá ver el documento en el navegador y en un principio solo vera los elementos apilados secuencialmente, lo cual se entiende que todo el diseño será delegado al CSS.

<aside>

En un típico diseño la llamada **Barra lateral** se ubica de lado de la columna de **Información principal**. Esta columna normalmente contiene datos relacionados con la información personal pero que no son relevantes o igual de importantes. Por ejemplo esta zona en un blog contendrá enlaces y ofrece información adicional del autor.

```
<!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>
  <header>
    <h1>Este es el título principal del sitio web</h1>
  </header>
  <nav>
    <ul>
      <li>principal</li>
      <li>fotos</li>
      <li>videos</li>
      <li>contacto</li>
    </ul>
  </nav>
  <section>

    </section>
    <aside>
      <blockquote>Mensaje número uno</blockquote>
      <blockquote>Mensaje número dos</blockquote>
    </aside>

  </body>
</html>
```

<footer>

Para finalizar la plantilla básica y estructura elemental, contamos con la zona que nos proporciona el cierre de nuestro diseño y otorgarle un final al cuerpo del documento, para ello tenemos la etiqueta **<footer>**, que representa la parte institucional ya que representa la parte final o pie.

```
<!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>
  <header>
    <h1>Este es el título principal del sitio web</h1>
  </header>
  <nav>
    <ul>
      <li>principal</li>
      <li>fotos</li>
      <li>videos</li>
      <li>contacto</li>
    </ul>
  </nav>
  <section>

  </section>
  <aside>
    <blockquote>Mensaje número uno</blockquote>
    <blockquote>Mensaje número dos</blockquote>
  </aside>
  <footer>
    Derechos Reservados &copy; 2010-2011
  </footer>

</body>
</html>
```

Dentro del cuerpo HTML5

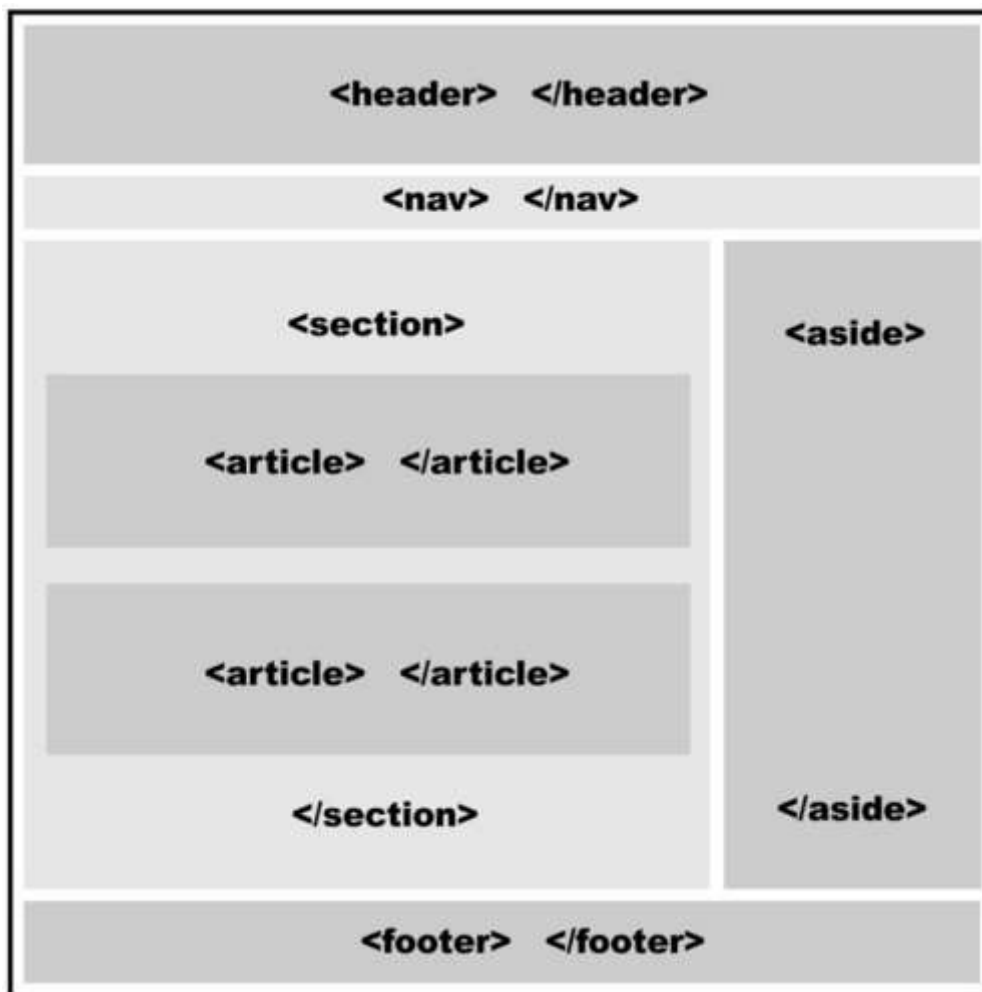
La estructura básica de nuestro sitio fue finalizada, pero queda trabajar en el contenido, hasta el momento los elementos vistos nos ayudan a identificar las secciones del documento y asignar un significado intrínseco, pero lo realmente importante es lo que contiene cada una de esas secciones. Esta información estará compuesta por diferentes elementos visuales como títulos, textos, imágenes, videos y aplicaciones interactivas.

<article>

Los sitios web presentan información relevante dividida importante que presenta las mismas características, el elemento **<article>** nos ayuda a identificar cada una de esas partes.

```
<section>
  <article>
    Este es el texto de mi primer mensaje
  </article>
  <article>
    Este es el texto de mi segundo mensaje
  </article>
</section>
```

Estas etiquetas se encuentran dentro del elemento **<section>** es decir, las etiquetas **<article>** son hijos del elemento **<section>**, del mismo modo cada elemento dentro de la etiqueta **<body>**.



Este elemento no se restringe por su nombre, quiere decir que no solamente tendrá artículos de noticias; el mismo fue creado con la intención de representar unidades independientes de contenido, como entradas de blog, comentarios de usuario, etc...

Aprovechando la versatilidad de los elementos **<header>** y **<footer>** puedes usarlos para definir los límites no solo del documento, sino también de cualquier sección del documento.

```

<section>
  <article>
    <header>
      <h1>Título del mensaje uno</h1>
    </header>
    Este es el texto de mi primer mensaje
    <footer>
      <p>comentarios (0)</p>
    </footer>
  </article>
  <article>
    <header>
      <h1>Titulo del mensaje dos</h1>
    </header>
    Este es el texto de mi segundo mensaje
    <footer>
      <p>comentarios (0)</p>
    </footer>
  </article>
</section>

```

<hgroup>

Es un elemento que permite agrupar el contenido de una etiqueta **<header>** esto con el propósito de que el navegador no interprete múltiples secciones en las diferentes secciones de cabecera.

```

<article>
  <header>
    <hgroup>
      <h1>Título del mensaje dos</h1>
      <h2>Subtítulo del mensaje dos</h2>
    </hgroup>
    <p>publicado 15-12-2011</p>
  </header>
  Este es el texto de mi segundo mensaje
  <footer>
    <p>comentarios (0)</p>
  </footer>
</article>

```

<figure> y <figcaption>

La etiqueta **<figure>** nos ayuda a identificar el contenido que es parte de la información pero a la vez independiente como lo son ilustraciones, fotos, videos, etc.

```
<figure>
  
  <figcaption>
    Esta es la imagen del primer mensaje
  </figcaption>
</figure>
```

El **<figcaption>** encierra el texto que tiene que ver con el **<figure>** y establece una relación entre ambos elementos y su contenido.

Referencia rápida

Acá la descripción básica de otras etiquetas HTML

<mark>: Este elemento resalta un texto que tiene relevancia en una situación en particular o que ha sido mostrando en la actividad de un usuario.

<small>: Este elemento representa contenido al margen, como letra pequeña.

<cite>: Este elemento es usado para mostrar el título de un trabajo.

<address>: Este elemento encierra información de contacto para un elemento.

<time>: Este elemento es usado para mostrar fechas y horas en formatos comprensibles por los usuarios y el navegador.

<video>: Inserta videos sin necesidad de plugins es muy fácil usarla pero cada navegador soporta codecs distintos.

<audio>: Inserta elementos de audio, se puede usar multiples formatos en especial .mp3

<input *>: Este elemento ya existía como elemento, pero ahora es más potente con la capacidad de definir diferentes tipos de caja debido a su contenido.

<canvas>: Área de dibujo vectorial y bitmaps con JavaScript.

<svg>: Es una etiqueta igual que el **** para insertar imágenes vectoriales. Basado en el estándar SVG.

Atributos

Los atributos dan ciertas características a un elemento, unos muy importantes que hasta definirán como interactúan como el **ID**, y que le darán su forma como el **CLASS** en conjunto con una hoja de estilos, el **STYLE** y los estilos que se aplican directamente. No en todas las etiquetas o elementos existen los mismos atributos, como por ejemplo está el caso del elemento **<A>** y el atributo **HREF**, este atributo define la página a la cual se debe direccionar. Otro ejemplo es el del **** y el atributo **SRC** que definirá la ruta de la imagen que se debe mostrar en pantalla.

Tag	Description	Attributes
A	Defines a hyperlink	Href, hreflang, media, name, rel, target, type
Abbr	Defines an abbreviation	
address	Defines contact information for the author/owner of a document/article	
Área	Defines an area inside an image-map	alt, coords, href, hreflang, media, rel, shape, target, type
article	Defines an article	
aside	Defines content aside from the page content	
audio	Defines sound content	autoplay, controls, loop, preload, src
B	Defines bold text	
Base	Specifies the base URL/target for all relative URLs in a document	href, target
Bdi	Isolates a part of text that might be formatted in a different direction from other text outside it	
bdo	Overrides the current text direction	dir
blockquote	Defines a section that is quoted from another source	cite
body	Defines the document's body	
Br	Defines a single line break	
button	Defines a clickable button	autofocus, disabled, form, formaction, formenctype, formmethod, formnovalidate, formtarget, name, type, value
id	Used to draw graphics, on the fly, via scripting (usually	id, height, width

	JavaScript)	
caption	Defines a table caption	
cite	Defines the title of a work	
code	Defines a piece of computer code	
col	Specifies column properties for each column within a <colgroup> element	span
colgroup	Specifies a group of one or more columns in a table for formatting	span
command	Defines a command button that a user can invoke	
datalist	Specifies a list of pre-defined options for input controls	
Dd	Defines a description of an item in a definition list	
del	Defines a text that has been deleted from a document	cite, datetime
details	Defines additional details that the user can view or hide	open
dfn	Defines a definition term	
Div	Defines a section in a document	
DI	Defines a definition list	
Dt	Defines a term (an item) in a definition list	
em	Defines emphasized text	
embed	Defines a container for an external application or interactive content (a plug-in)	height, src, type, width
fieldset	Groups related elements in a form	disabled, form, name
figcaption	Defines a caption for a <figure> element	
figure	Specifies self-contained content	
footer	Defines a footer for a document or section	
form	Defines an HTML form for user input	Accept-charset, action, autocomplete, enctype, method, name, novalidate, target
h1	Defines HTML headings	
h2	Defines HTML headings	
h3	Defines HTML headings	
h4	Defines HTML headings	
h5	Defines HTML headings	

h6	Defines HTML headings	
head	Defines information about the document	
header	Specifies an introduction, or a group of navigation elements for a document	
hgroup	Groups a set of <h1> to <h6> elements when a heading has multiple levels	
hr	Defines a thematic change in the content	
html	Defines the root of an HTML document	manifest, xmlns
i	Defines a part of text in an alternate voice or mood	
iframe	Defines an inline frame	height, name, sandbox, seamless, src, srcdoc, width
img	Defines an image	src, alt, height, ismap, usemap, width
input	Defines an input control	accept, alt, autocomplete, autofocus, checked, disabled, form, formaction, formenctype, formmethod, formnovalidate, formtarget, height, list, max, maxlength, min, multiple, name, pattern, placeholder, readonly, required, size, src, step, type, value, width
ins	Defines a text that has been inserted into a document	cite, datetime
keygen	Defines a key-pair generator field (for forms)	autofocus, challenge, disabled, form, keytype, name
kbd	Defines keyboard input	
label	Defines a label for an input element	for, form
legend	Defines a caption for a <fieldset>, <figure>, or <details> element	
li	Defines a list item	value
link	Defines the relationship between a document and an external resource (most used to link to style sheets)	href, hreflang, media, rel, sizes, type
map	Defines a client-side image-map	name

mark	Defines marked/highlighted text	
menu	Defines a list/menu of commands	label, type
meta	Defines metadata about an HTML document	charset, content, http-equiv, name
meter	Defines a scalar measurement within a known range (a gauge)	form, high, low, max, min, optimum, value
nav	Defines navigation links	
noscript	Defines an alternate content for users that do not support client-side scripts	
object	Defines an embedded object	data, form, height, name, type, usemap, width
Ol	Defines an ordered list	reversed, start, type
optgroup	Defines a group of related options in a drop-down list	label, disabled
option	Defines an option in a drop-down list	disabled, label, selected, value
output	Defines the result of a calculation	for, form, name
P	Defines a paragraph	
param	Defines a parameter for an object	name, value
pre	Defines preformatted text	
progress	Represents the progress of a task	max, value
Q	Defines a short quotation	cite
Rp	Defines what to show in browsers that do not support ruby annotations	
Rt	Defines an explanation/pronunciation of characters (for East Asian typography)	
ruby	Defines a ruby annotation (for East Asian typography)	
S	Defines text that is no longer correct	
samp	Defines sample output from a computer program	
script	Defines a client-side script	async, defer, type, charset, src
section	Defines a section in a document	
select	Defines a drop-down list	autofocus, disabled, form, multiple, name, size
small	Defines smaller text	
source	Defines multiple media resources	media, src, type

	for media elements (<video> and <audio>)	
span	Defines a section in a document	
strong	Defines important text	
style	Defines style information for a document	type, media, scoped
sub	Defines subscripted text	
summary	Defines a visible heading for a <details> element	
sup	Defines superscripted text	
table	Defines a table	border
tbody	Groups the body content in a table	
Td	Defines a cell in a table	colspan, headers
textarea	Defines a multiline input control (text area)	autofocus, cols, disabled, form, maxlength, name, placeholder, readonly, required, rows, wrap
tfoot	Groups the footer content in a table	
Th	Defines a header cell in a table	colspan, headers, rowspan, scope
thead	Groups the header content in a table	
time	Defines a date/time	datetime, pubdate
title	Defines a title for the document	
Tr	Defines a row in a table	
track	Defines text tracks for media elements (<video> and <audio>)	default, kind, label, src, srclang
Ul	Defines an unordered list	
var	Defines a variable	
video	Defines a video or movie	autoplay, controls, height, loop, muted, poster, preload, src, width
wbr	Defines a possible line-break	

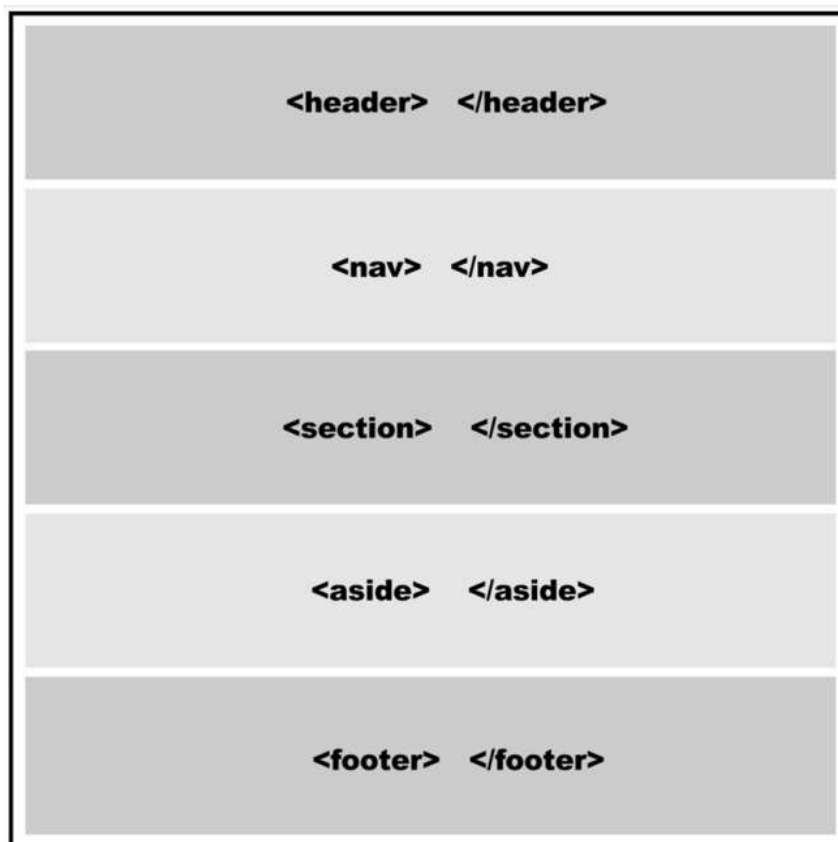
CSS y HTML

Como hemos venido comentando la web demanda diseño y funcionalidad, y no solo organización estructural o definición de sesiones. El este nuevo paradigma se presenta como HTML+CSS+JS como un único instrumento integrado.

Estilos y estructuras

A pesar de que cada navegador define estilos por defecto a cada elemento HTML, diseñadores y desarrolladores deben aplicar sus propios estilos para obtener organización y el efecto visual que realmente desean.

Con respecto a esto cada navegador ordena elementos por defecto de acuerdo a su tipo **block** o **inline**, los elementos se asocian a esta clasificación de acuerdo a la forma como se ordenan en la pantalla. Los elementos estructurales están clasificados como elementos **block**, teniendo en cuenta lo anterior lo que veríamos en el navegador con el código HTML desarrollado a través de la guía es lo siguiente:



Modelos de caja

Para poder crear nuestra propia organización de elementos, debemos comprender como los navegadores procesan el código HTML, aquí comprendemos que los navegadores consideran cada elemento como una caja. De la misma forma determinamos que una página web no es más que un montón de cajas ordenadas y este orden lo determinamos con el CSS.

CSS tiene un set de propiedades destinadas a sobrescribir los estilos definidos por los navegadores y permitiendo obtener la organización deseada, estas propiedades no son específicas y tienen que ser combinadas para agrupar cajas (elementos) y obtener su correcta disposición en pantalla. La combinación de estas reglas es normalmente llamado modelo o sistema de disposición. Todas estas reglas juntas constituyen lo que se denomina modelo de caja.

Conceptos básicos

Antes de insertar reglas CSS en nuestro archivo de estilos y aplicar un modelo de caja, debemos entender los conceptos básicos sobre CSS. Para aplicar un estilo podemos usar diferentes técnicas las cuales describiremos a continuación.

Estilos en línea: es la técnica más simple para incorporar CSS al HTML, consiste en asignar los estilos dentro de los elementos por medio del atributo **STYLE**.

En un documento .html comprueba el siguiente ejemplo y adicionalmente crea una etiqueta **<p>** sin los estilos aplicados en el atributo **STYLE** y compara.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <title>Este es el título del documento</title>
</head>
<body>
  <p style="font-size: 20px">Mi texto</p>
</body>
</html>
```

Estilos embebidos: es una mejor alternativa, consiste en insertar los estilos en la cabecera del documento luego usar los referencias para afectar los elementos HTML.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <title>Este texto es el título del documento</title>
  <style>
    p { font-size: 20px }
  </style>
</head>
<body>
  <p>Mi texto</p>
</body>
</html>
```

El elemento **<style>** incrustado en el **<head>** permite agrupar los estilos CSS dentro del documento, en HTML5 no es necesario determinar ningún atributo ya que por defecto determina que son CSS los que se van a incrustar.

Archivos externos: declarar los estilos en la cabecera del documento nos ahorra espacio, pero en este caso sería necesario copiar este estilo en todos los documentos de nuestro sitio web. La solución para ello es poder mover todos estos estilos en un archivo externo. En la cabecera podemos usar la etiqueta **<link>**, esta etiqueta tiene la función de insertar una hoja de estilos externa. Su uso podemos ver en el siguiente ejemplo **<link rel='stylesheet' href='miestilo.css' >**. El realizar los estilos en un archivo externo nos ahorra espacio en el documento HTML y además nos permite modificar fácilmente los estilos de un sitio web, sin la necesidad de modificar archivo por archivo del sitio web.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <title>Este texto es el título del documento</title>
  <link rel="stylesheet" href="misestilos.css">
</head>
<body>
  <p>Mi texto</p>
</body>
</html>
```

Referencias CSS

Almacenar nuestros estilos en un archivo externo e insertarlo en el archivo HTML que se necesite es muy conveniente, sin embargo no sería posible hacer esto sin mecanismos que nos ayuden a establecer relación entre los estilos y los elementos, de lo cual hay varios métodos para seleccionar cuales elementos serán afectados por las reglas CSS:

- Referencia por **clave** del elemento.
- Referencia por el atributo **ID**.
- Referencia por atributo **CLASS**.

Referencia por palabra CLAVE del elemento: utilizamos la palabra clave del elemento y afectamos todos los elementos de un documento que pertenezcan a ese tipo.

```
p { font-size: 20px }
```

Referencia con el atributo ID: este es el nombre único de cada elemento, esto significa que este atributo no puede ser duplicado y es único en cada documento, para referenciar este elemento desde nuestro CSS lo declaramos con el símbolo # al frente el valor que usamos para identificar el elemento.

```
#texto1 { font-size: 20px }
```

En el HTML usamos un elemento de cualquier tipo con el valor **id='texto1'**, Ejm:
<div id='texto1'></div>.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <title>Este texto es el título del documento</title>
  <link rel="stylesheet" href="misestilos.css">
</head>
<body>
  <p id="texto1">Mi texto</p>
</body>
</html>
```

Referencia con el atributo CLASS: para propósito de estilo es mejor usar el atributo CLASS, este atributo es más flexible y puede ser asignado a cada elemento contenido en el documento HTML, por decirlo así a elemento que comparten un diseño similar. Para trabajar con CLASS debemos declarar las reglas con un **punto** antes del nombre o valor del atributo:

```
.texto1 { font-size: 20px }
```

La ventaja de este método es que insertar el atributo CLASS con el valor **texto1** (en nuestro ejemplo) será suficiente para signar los estilos a cualquier elementos que queramos:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <title>Este texto es el título del documento</title>
  <link rel="stylesheet" href="misestilos.css">
</head>
<body>
  <p class="texto1">Mi texto</p>
  <p class="texto1">Mi texto</p>
  <p>Mi texto</p>
</body>
</html>
```

Como vemos en la anterior imagen el elemento **<p>** esta dos veces y tienen el atributo CLASS con el mismo valor y serán afectados por la misma regla CSS.

Referencia con cualquier atributo: acá podemos asignar una regla de estilo a cualquier elemento teniendo en cuanto cualquier atributo del elemento por ejemplo:

```
p[name] { font-size: 20px }
```

En este ejemplo podemos ver que se le asigna una regla a un elemento **<p>** directamente relacionado con su atributo name. Para limitar esta asignación se utiliza el valor que lleva el atributo como por ejemplo:

Estructura básica HTML5 y CSS3 | garodriguez335@misena.edu.co.

Página 31

de asignación podríamos de la misma manera asignar un estilo que solo seleccione los elementos pares o impares.

Para identificar su funcionalidad realice los siguientes estilos para el mismo documento HTML:

1

```
p:nth-child(4) {  
    background: #999999;  
}
```

2

```
*{  
    margin: 0px;  
}  
  
p:nth-child(1) {  
    background: #999999;  
}  
p:nth-child(2) {  
    background: #CCCCCC;  
}  
p:nth-child(3) {  
    background: #999999;  
}  
p:nth-child(4) {  
    background: #CCCCCC;  
}
```

3

```
*{
  margin: 0px;
}
p:nth-child(odd){
  background: #999999;
}
p:nth-child(even){
  background: #CCCCCC;
}
```

4

```
*{
  margin: 0px;
}
p:last-child{
  background: #999999;
}
```

5

```
:not(p){
  margin: 0px;
}
```

```
:not(.mitexto2) {  
  margin: 0px;  
}
```

Aplicando CSS, Modelo de Caja Tradicional

Como comprendimos todo elemento estructural es una caja y esas cajas agrupadas constituyen un modelo de caja. Actualmente el modelo de caja tradicional está directamente referenciado a los DIVS y su organización en CSS. En HTML5 no vamos a usar la etiqueta DIV para el modelo de caja ya que tenemos elementos dedicados para las diferentes secciones tal como vimos anteriormente.

Inicialmente ya tenemos una plantilla en HTML5 ahora vamos a realizar algunas añadiduras para poder implementar el modelo de caja con CSS. Lo primero que debemos hacer agregar el elemento **<div>**, con el fin de agrupar, centrar y dar un tamaño específico a las cajas. La nueva plantilla deberá lucir de la siguiente forma.

```

<!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>
        
        <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>

```

Ahora ya provee de una plantilla lista para recibir los estilos CSS, el primer template para nuestro CSS.

Sintaxis CSS

Ahora pasamos a la sintaxis de cada estilo CSS, esta es muy fácil e intuitiva, consiste en una serie de reglas que describen la forma de visualizar cada uno de los elementos del documento del siguiente modo:

```
SELECTOR { PROPIEDAD : VALOR; }
```

Ejm:

```
H1 { align: left; }
```

La primera parte de la regla es el **SELECTOR** y que indica que elementos del HTML afectara (esto está más arriba del documento donde indicamos referencias CSS), lo que hay entre corchetes es lo que se llama **DECLARACION** y donde indicamos como se va a ver el selector.

La declaración está formada por una o más **PROPIEDADEDES** a la que se le asigna un **VALOR**.

Plantilla CSS

Iniciamos la construcción de nuestra plantilla CSS vamos a ir agregando cada parte de nuestra hoja de estilos, recomiendo que paso a paso vayan refrescando el navegador para ver reflejados los estilos:

Selector universal *: comencemos con algunas reglas básicas que nos ayudaran a proveer consistencia al diseño:

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

Normalmente necesitaremos personalizar los márgenes o simplemente mantenerlos al mínimo como el estilo anterior.

Nueva jerarquía para cabeceras: en nuestra plantilla utilizamos elementos `<h1>` y `<h2>` para títulos y subtítulos, pero inicialmente estos elementos están lejos de lo que necesitamos:

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

Declaramos nuevos elementos HTML5: otra regla básica es declarar desde el comienzo los elementos HTML5 ya que algunos navegadores con las reglas por defecto los tratan como elementos *inline*. De esta manera estos elementos serán tratados como normalmente serían tratados los elementos `<div>`:

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

Centrando el cuerpo: por defecto la etiqueta `<body>` tiene un valor de ancho de 100% lo que significa que tendrá el valor total de la pantalla del navegador, por lo tanto para centrar la página necesitamos centrar el contenido del `<body>`:

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

Creando la caja principal: siguiendo nuestro diseño debemos especificar el tamaño máximo del contenido de nuestro cuerpo como anteriormente recuerda agregamos un elemento `<div>` que agrupa todo el contenido del body, ese elemento será considerado nuestra caja principal:

```
#agrupar {  
    width: 960px;  
    margin: 15px auto;  
    text-align: left;  
    border: 1px solid #000;  
}
```

Nos referenciamos al elemento **<div>** por medio de su atributo **ID**.

La cabecera: siguiendo con el resto de elementos estructurales nos encontramos con la cabecera de nuestra página el primer elemento **<header>**. El elemento contiene el título principal de nuestra página:

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

Barra de navegación: después de nuestra cabecera encontramos la barra de navegación los elementos agrupados representaran el menú de nuestro sitio:

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

En la primera regla tenemos los estilos para nuestro contenedor **<nav id='menu'>** en el segundo tenemos la configuración para los elementos **** que están en su interior.

Section y aside: los siguientes elementos por asignar las reglas de estilo son los elementos que van organizados de manera horizontal. Para ello es importante tener en cuenta la propiedad **float**.

Utilizando la propiedad **float** podemos hacer que el elemento flote a un lado por ejemplo que los elementos se agrupen al lado izquierdo o derecho respectivamente del elemento padre. Para ello también necesitamos definir los tamaños de los elementos que estarán flotando. Como el ancho y alto:

```
#seccion {
    float: left;
    width: 660px;
    margin: 20px;
    background-color: #E6E6E6;
}
#columna {
    float: left;
    width: 220px;
    margin: 20px 0px;
    padding: 20px;
    background: #CCCCCC;
}
```

Footer: para finalizar la aplicación de modelo de caja vamos a agregar un valor que le permitirá a nuestro modelo tomar su flujo normal; esta propiedad permite ubicar al elemento **<footer>** posicionarlo debajo del último elemento en lugar de su lado. Esta es la propiedad **clear**, esta simplemente permite restaurar las condiciones normales del elemento en el área ocupada. El valor de esta propiedad es **both**, esta propiedad también empuja los elementos verticalmente haciendo que los elementos flotantes ocupen un espacio real en pantalla.

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

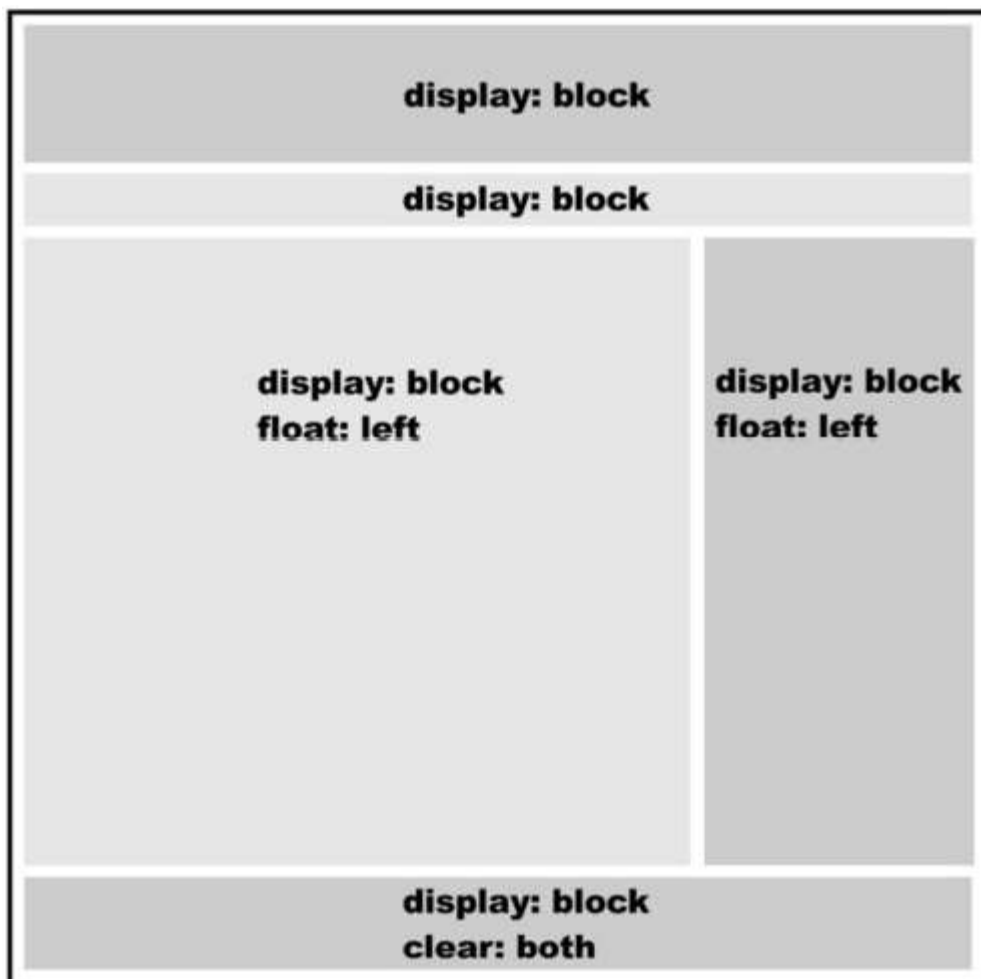
Últimos retoques: lo único que nos queda por hacer es trabajar un poco del diseño del contenido. Para ello solo necesitamos configurar un poco los elementos HTML5 restantes, esos elementos son el **<article>**, el **<footer>** de los article, el **<time>** y el **<figcaption>**:


```

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;
}

```

Al final nuestro documento HTML5 tendrá la siguiente forma y sabremos cómo están ubicados los elementos según las reglas CSS que definimos.





Página finalizada



Actividad

- 1- Debe entregar la página maquetada durante todo el documento.
- 2- Realice una página html que contenga una tabla con la siguiente información:
 - Lista de etiquetas html
 - Descripción de la etiqueta
 - Ejemplo funcional de la etiqueta

Pdta: puede usar un elemento <table> con tres columnas para desarrollar el ejercicio.

Etiqueta	Descripción	Ejemplo
<code></code>	Etiqueta representa el elemento de imágenes.	
<code><audio src='cancion.mp3' /></code>	Etiqueta que agrega un elemento de reproducción de audio.	

Referencia

<http://www.aulaclie.es/html/index.htm>

<http://roble.pntic.mec.es/apuente/html/paginas/resumen.htm>

<http://www.w3schools.com/html/>

<http://validator.w3.org/>