POLITÉCNICO COLOMBIANO
JAIME ISAZA CADAVID

Educación para vivir mejor

Construcción de elementos de software web 1

Semana 3

Fundamentos de HTML y CSS - Intermedio

Media Técnica en Programación de Sistemas de Información
Politécnico Jaime Isaza Cadavid - 2020

ISO 9001 icontec SC 5206-1
IQNet
Politécnico Colombiano Jaime Isaza Cadavid
@PolitecnicoJIC

Profundizando etiquetas HTML

Hemos visto que las etiquetas HTML nos permiten estructurar nuestra página web y en ella el contenido que se encuentra materializado en dicha página, pero algo importante es que se estructura de acuerdo a su función o carga semántica. Es esencial que profundicemos en etiquetas que nos ayuden a:

1. Definir nuestra página.
2. Agrupar secciones.
3. Identificar semánticamente el contenido.
4. Crear Tablas de datos.

En este punto debemos recordar que las etiquetas que actúan como contenedoras es normal que lo que vive dentro de ellas se vaya anidando, es decir que metamos etiquetas dentro de otras etiquetas y sepamos cual es la etiqueta padre de ese conjunto de etiquetas.

```html
<html>
  <body>
    <header>
      <h1>Título</h1>
    </header>
    <main>
      <section>
        <h2>Subtítulo</h2>
        <p>Contenido y más contenido</p>
        <p>Contenido con <a href="">enlaces</a></p>
        <ul>
          <li>lista</li>
          <li>de</li>
          <li>cosas</li>
        </ul>
      </section>
    </main>
  </body>
</html>
```

Secciones

Algo que normalmente en nuestras vidas hacemos, es buscar la forma de agrupar distintos elementos que tienen algo en común y que nos permita identificarlos fácilmente o poderlos encontrar de acuerdo a dicha clasificación. Es por ello que a medida que avanzamos en la estructura de nuestros proyectos de software web, evidenciamos que deseamos agrupar esta estructura en bloques de código.

Para materializar estos bloques podemos hacer uso de la etiqueta **<section>**, es decir, vamos a usar la etiqueta <section> para agrupar contenidos de acuerdo a una temática, por ejemplo, imagina una página de productos, donde por cada producto vamos a agrupar la descripción del producto por un lado y los comentarios que han realizado los usuarios o clientes por el otro.

Entonces empezamos a tener una estructura semántica de nuestra página web, existe una serie de secciones especiales que tienen asignado un significado semántico predeterminado:

- **<header> :** es una cabecera o sección de presentación de un bloque.
- **<main> :** indica la sección principal del contenido.
- **<footer>** : un pie de página o sección final de un bloque.
- **<nav> :** indica un bloque de navegación, normalmente para un menú.
- **<aside> :** es un bloque de contenido de menor importancia o con contenido relacionado.
- **<article> :** se consideran independientes del sitio web, permitiendo ser vistos o utilizados por separado, tales como: artículos, entradas de blogs o mensajes de un foro.

Estas etiquetas contenedoras especiales se pueden usar unas dentro de otras siempre y cuando esto tenga sentido: por ejemplo, un <article> puede tener cabecera y pié de página, mientras que una cabecera no debería tener en su interior un pie de página.

En este punto debemos hacer especial énfasis en que si usamos mal estos elementos, el navegador no va a arrojar ningún error, pero lo que sí es cierto es que estaríamos brindando muy poco valor a aquellos usuarios que necesiten un valor semántico extra en nuestro sitio web para poder navegar, por ejemplo, una persona con discapacidad visual.

Contenido

Teniendo claro que gracias a las distintas secciones que podemos empezar a identificar de forma semántica en nuestra página web, debemos profundizar en más etiquetas de contenido además de los encabezados, párrafos y listas.

Enlaces

Uno de los conceptos básicos de HTML es el uso de los enlaces, estos nos van a permitir vincular páginas o partes de ellas de manera que la información no quede de forma aislada sino que se vaya generando distintas interconexiones que permiten navegar por nuestro sitio web.

Un claro ejemplo, donde se evidencian muchos enlaces, es la página de Wikipedia, en esta a cada artículo se añaden enlaces relacionados que permiten profundizar o complementar la información a medida que vamos consultando sobre el tema que deseamos aprender.

Los enlaces se escriben haciendo uso de la etiqueta **<a>** y con un atributo **href=””** el cual indica el enlace o a dónde enlaza.

De esta manera podemos enlazar a:

- Una página interna
- Una página externa
- Un archivo
- Una posición específica dentro de la misma página u otra página.

Entonces un primer tipo de enlace, expresado de una forma muy sencilla, es simplemente poner la dirección de nuestra página o archivo como valor del atributo href:

```
<a href="https://www.wikipedia.org">Wikipedia.com</a>
```

El segundo tipo de enlace, va a necesitar de un atributo especial denominado **id=”” .** Cualquier elemento de nuestra página puede poseer este atributo, pero al tratarse de un atributo que permite identificar a dicho elemento **debe ser único** y no debe existir dos elementos o más con el mismo id en la misma página.

En el siguiente ejemplo vamos a identificar la cabecera y el contenido principal:

```
<!doctype html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Mi página</title>
</head>
<body>
  <header id="top">
    <h1>Título de mi página</h1>
  </header>
  <main id="main">
    <h2>Texto en latín</h2>
    <p>Lorem ipsum dolor sit amet, magna aliqua.</p>
  </main>
</body>
</html>
```

Ahora podemos añadir un <footer> donde indiquemos al usuario en un pie de página un enlace que le permita ahorrar tener que desplazarse hacia arriba para volver al contenido inicio del contenido, esto lo logramos en el href haciendo uso del símbolo de numeral **#,** seguido del id que queremos enlazar:

```
<!doctype html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Mi página</title>
</head>
<body>
  <header id="top">
    <h1>Título de mi página</h1>
  </header>
  <main id="main">
    <h2>Texto en latín</h2>
    <p>Lorem ipsum dolor sit amet, magna aliqua.</p>
  </main>
  <footer>
    <p><a href="#top">Volver arriba</a></p>
  </footer>
</body>
</html>
```

Si se quisiera enlazar el contenido principal de la página desde otra página, por ejemplo desde un archivo product.html se usaría de la siguiente forma:

```
<a href="index.html#top">Volver arriba de la página principal</a>
```

En estos dos casos podemos afirmar que las rutas son **relativas**, ya que apuntan dentro de nuestro proyecto. Si incluimos el dominio donde está alojada la página o archivo, aunque sea en nuestro propio dominio, diremos que estamos utilizando una ruta **absoluta.**

Al hacer uso de la etiqueta <a> podemos incluir otros atributos que debemos conocer:

- **title = “ ”** : En este atributo podemos añadir un texto complementario que el navegador mostrará en un pequeño tooltip (mensaje de ayuda) cuando pongamos el cursor sobre el enlace. Este tipo de atributo es interesante usar cuando tengamos un enlace del tipo descargar y quiero asociar que lo que se va a descargar es un archivo pdf con el mensaje “Descargar archivo PDF”.

- **target = “ ” :** En este atributo podemos especificar donde se abre el enlace, si quisiéramos que al darle click al enlace se nos abra en una nueva pestaña utilizaremos el valor **_blank.**

Negritas, cursivas

Tradicionalmente se usaban las etiquetas **<b> y <i>** para poner un texto en negrita (*bold*) o en cursivas o itálicas (*italic*). Estas etiquetas se mantienen aunque no tienen carga semántica, no significan nada más allá de que muestran el texto visualmente en negrita o cursiva.

Las nuevas etiquetas, ** y **, aunque visualmente hacen lo mismo (strong muestra el texto en negrita y em, en cursiva) sí que tienen una carga semántica, para indicar el nivel de énfasis o de importancia. Con **** resaltar un texto importante, y con resaltar un texto más importante.

```
<p>Dentro de este párrafo tenemos <em>un texto importante</em> y <strong>uno muy importante</strong></p>
```

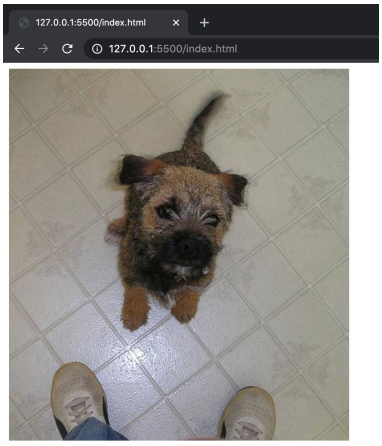
Imágenes

Muchas veces queremos acompañar nuestro contenido con imágenes, ya sea para acompañarlo, o como un motivo principal en el cual queremos hacer un énfasis especial.

Para ello tenemos la etiqueta **<img>**, que tiene varios atributos:

- **src = " " :** Aquí indicamos la ruta de nuestro archivo de imagen.
- **alt = " " :** El atributo alt es el texto que va a mostrar el navegador en caso de la imagen no se pueda cargar, pero también es muy importante para la accesibilidad, cuando la imagen es parte del contenido debemos usarlo añadiendo un texto descriptivo. Cuando la imagen sea decorativa, se recomienda dejar el valor vacío, pero no omitir dicho valor.

```
<img src="https://images.dog.ceo/breeds/terrier-border/n02093754_6992.jpg" alt="Mi mascota">
```



Saltos de línea

Tenemos unas etiquetas que fuerzan un salto de línea: **
.** Desde que empezamos a separar el contenido y el diseño esta etiqueta ha quedado relegada a un lugar muy secundario pero todavía hay veces en los que vas a querer forzar una línea nueva en mitad de un texto y está bien conocerla.

```
<h1>Mi título genial <br>en dos líneas</h1>
```

Contenedores generales

Aparte de las secciones tenemos un par de contenedores sin propósito específico que nos sirven para hacer agrupaciones sin carga semántica, estas etiquetas son ** y el **.

Mientras que el **div** es para bloques de contenido el **span** está indicado para partes del texto o elementos en línea.

Tablas

Hubo un tiempo en el que las tablas eran la base sobre la que se creaba cualquier página web. Hoy se utilizan para lo que son: presentar datos tabulados.

La tabla básica tiene una estructura bastante simple y tiene tres etiquetas principales:

- una etiqueta que marque que se va a escribir una tabla
- una etiqueta para las filas
- una etiqueta para las celdas

```
<table>
  <tr>
    <td></td>
    <td></td>
    <td></td>
  </tr>
  <tr>
    <td></td>
    <td></td>
    <td></td>
  </tr>
  <tr>
    <td></td>
    <td></td>
    <td></td>
  </tr>
</table>
```

Selectores en CSS

Los navegadores ofrecen este aspecto por defecto pero nosotros lo podemos cambiar con CSS, creando estilos para definir la apariencia de nuestras página.

Para cambiar el aspecto de un elemento usamos un selector, hay varios tipos de selectores:

- la propia etiqueta del elemento: h1, a, p.
- una clase que hayamos incluido con el atributo **class=""**
- a través de un identificador en el atributo **id=""**
- a través de una pseudo clase, que son unas palabras claves que añadidas al selector especifican un estado especial del elemento.
- a través de una mezcla de los anteriores

Vamos a ver cada uno de los casos.

El propio elemento como selector

Esto no es lo ideal y se aplica los estilos a cada elemento de este tipo que aparezca en la página, así que hay que usarlo con tiento.

Podemos, por ejemplo, hacer que todos los enlaces sean rojos.

```
1   a {
2       color: red;
3   }
```

Clases como selectores

Las clases son palabras claves que atribuimos a elementos HTML para poder agruparlos por función o apariencia y diferenciarlos del resto de elementos de su mismo tipo.

Por ejemplo: La clase "text-link" nos permite aplicar estilos particulares a los enlaces que lleven dicha clase sin afectar al resto de etiquetas.

```
<a href="#" class="text-link">Enlace de texto</a>
```

En CSS creamos clases para aplicar a grupos de elementos como pueden ser todos los enlaces de texto, los elementos del listado de ingredientes o a los párrafos del pié de página. La manera de indicar en css que se trata de una clase es escribiendo un **.** primero:

```
1  .text-link {
2      color: red;
3  }
```

Id como selector

Ya habíamos visto que los *id*, eran una palabra clave que usábamos como identificador para un único elemento. En CSS también los podemos usar como selector, pero al no poder haber más de uno por página no es recomendable usarlo salvo en casos muy excepcionales.

En una lista de acciones, por ejemplo, podemos tener unas clases para añadir estilos a los elementos del bloque y, además, añadir un identificador único para cada elemento.

A nivel de código tendríamos un resultado como el siguiente:

```html
<ul class="actions">
  <li class="action">
    <a id="add-user" href="" class="button">Nuevo usuario</a>
  </li>
  <li class="action">
    <a id="rename-user" href="" class="button">Renombrar usuario</a>
  </li>
  <li class="action">
    <a id="delete-user" href="" class="button">Eliminar usuario</a>
  </li>
</ul>
```

Y ahora podríamos usar el *id* para cambiar el tamaño del texto de uno de los elementos. Para ello, usamos la # seguida de la id como selector.

```css
#add-user {
    font-size: 24px;
}
```

Pseudo clase como selector

Las pseudo clases son palabras claves que añadidas a alguno de los selectores anteriores especifican un estado concreto del elemento. El más usado es el estado de **hover**, que ocurre cuando colocamos el ratón encima del elemento.

Las pseudo clases se escriben usando el selector, seguido de **:** y la palabra clave para el estado.

Por ejemplo, como en uno de los ejemplos anteriores, tenemos un enlace que vamos al cual le vamos a poner el texto de color rojo, pero cuando coloques el cursor encima invertiremos los colores y lo mostraremos con fondo blanco y color rojo. Partimos del mismo html que anteriormente.

```html
<a href="#" class="text-link">Enlace de texto</a>
```

Y la clase sería:

```
.text-link {
    color: red;
}
.text-link:hover {
    background-color: red;
    color: white;
}
```

Los selectores se pueden mezclar

De los ejemplos de las pseudo clases vemos que los selectores se pueden mezclar. Esto nos ayuda a contemplar casos particulares sin tener que usar las ID.

Un elemento HTML puede tener tantas clases como queramos y las indicamos en su atributo class separadas cada una por un espacio.

Por ejemplo, si tenemos una lista de botones como la anterior:

```
<ul class="actions">
    <li class="action">
       <a href="" class="button button--new">Nuevo usuario</a>
    </li>
    <li class="action">
      <a href="" class="button button--rename">Renombrar usuario</a>
    </li>
    <li class="action">
       <a href="" class="button button--delete">Eliminar usuario</a>
    </li>
</ul>
```

Hemos eliminado el atributo **id** y hemos añadido una clase extra para cada tipo de botón. De esta manera tenemos por cada "botón" una clase general **.button** donde colocaremos los estilos comunes a todos los botones y luego una particular **(.button--new, .button--rename y .button--delete)** donde solo pondremos los ajustes particulares.

Digamos que queremos que los botones tengan una caja con bordes redondeados pero que el de añadir usuario tenga fondo verde, el de renombrar azul y el de borrar, claramente, rojo muerte.

```css
.button {
  background-color: grey;
  border-radius: 20px;
  color: white;
  display: inline-block;
  margin-bottom: 1em;
  padding: 10px 20px;
  text-decoration: none;
}

.button--new {
  background-color: green;
}

.button--rename {
  background-color: blue;
}

.button--delete {
  background-color: red;
}
```

Herencia en CSS

Hay una serie de estilos que se heredan, es decir, que se transmiten a los hijas. Entonces, si aplicamos una de estas propiedades a una etiqueta, todos las etiquetas anidadas en ella la heredarán también.

Cascada y especificidad de selectores

CSS es, en español, "hojas de estilo en cascada". La "cascada" se refiere al proceso de combinación y aplicación de estilos en CSS y cómo se resuelven los conflictos entre ellos.

Acabamos de ver que a veces varios selectores se aplican al mismo elemento, es el algoritmo de la cascada lo que decide qué propiedades se aplicarán.

La cascada depende de 3 factores:

1. La **importancia**: hay una palabra clave **(!important)** que hace que nuestra propiedad se aplique siempre.
2. La **especificidad**: es un arma de doble filo porque cuanto más específico sea un selector más fuerza tendrán sus reglas sobre las demás. Pero es una buena práctica escribir los selectores lo menos específicos posibles.
3. El **orden** en el archivo CSS: si varios selectores tienen la misma "fuerza" ganarán los que estén más abajo porque el CSS se aplica en orden de escritura.

Taller: Ejercicios Práctico

1. **Organizando la semana:** Hacer una tabla con la comida de cada día de la semana usando **<th>** para las celdas que contienen los días.
2. **Una página clásica:** Realizar una página semántica con: un título principal, tres párrafos con el contenido, dos párrafos con anuncios secundarios, y un texto de derechos de autor - copyright - (© 2020).
3. **Coloreando unos enlaces:** El color es una de la propiedades que se heredan así que si tenemos esta estructura:

```
<article>
    <h2>Título</h2>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit</p>
    <p>Sed do eiusmod tempor incididunt ut labore et dolore </p>
    <aside class="links">
        <h3>Enlaces relaccionados</h3>
        <ul>
            <li><a href="#">Enlace 1</a></li>
            <li><a href="#">Enlace 2</a></li>
            <li><a href="#">Enlace 3</a></li>
            <li><a href="#">Enlace 4</a></li>
        </ul>
    </aside>
</article>
```

y al **<aside>** con clase **.links** le aplicamos una regla que ponga el texto rojo,

Pregunta: ¿Qué quedará en rojo?

4. Partiendo del **ejercicio 2** vamos a hacer uso de la herencia, y utilizando un solo selector de css vamos a hacer que:

    - El tamaño de fuente de la página sea de **18px**
    - El tipo de fuente sea **Tahoma**
    - El color de fuente sea **rebeccapurple**
    - El color de fondo sea **mediumpurple**
    - El interlineado sea de **1.5**

Nota: Tenemos que trabajar con la primera etiqueta del contenido de **html** que es **<body>**, y utilizar el selector de etiqueta para que todas sus hijas anidadas hereden de ella.

5. Partiendo del ejemplo que se encuentra en el siguiente enlace https://codepen.io/oneeyedman/pen/vWEBex

Responda:

- Por qué los enlaces son verdes y no rojos
- Hacer que los enlaces sean rojos
- Rehacer el HTML usando en lugar de <ul> y <li>. ¿Qué pasa?
- Comentar el CSS que no se puede tocar y reescribir este CSS usando una clase por selector para que se vea igual.