1.- Declara el DOCTYPE

A través de la declaración del tipo de documento, o DOCTYPE, indicamos el estándar usado al construir un documento HTML. De esta forma, los navegadores que lean esta declaración interpretarán el documento con el lenguaje y versión definidos.

Cabe destacar que, estando en 2017, no debemos usar un DOCTYPE obsoleto, y utilizar siempre la versión recomendada de HTML, que a día de hoy es HTML5.

Mal (por usar versiones antiguas):

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"

"http://www.w3.org/TR/html4/strict.dtd">

Bien (se declara el DOCTYPE de una versión actual y recomendada de HTML):

<!DOCTYPE html>

Como dato de interés, si quieres saber por qué la declaración del DOCTYPE de HTML5 es tan sencilla comparada con la de versiones anteriores, puedes consultar el siguiente enlace de la W3C.

2.- Cierra las etiquetas

Una de las primeras cosas que se estudian cuando se está aprendiendo HTML es que las etiquetas deben cerrarse, y en el mismo orden en el que se abrieron.

Mal:

<div>

<p>Contenido

<p>Contenido</p>

</div>

Bien:

<div>

<p>Contenido</p>

<p>Contenido</p>

</div>

Cabe destacar que en HTML5 se introdujeron las etiquetas vacías (void tags en inglés) que son un caso especial que no necesitan etiqueta de cierre.

Mal:

<p>Este es<br></br>un mal ejemplo</p>

Bien:

<p>Este es<br>un buen ejemplo</p>

Dichas etiquetas son:

<area>, <base>, <br>, <col>, <embed>, <hr>, <img>, <input>, <keygen>, <link>, <menuitem>, <meta>, <param>, <source>, <track>, <wbr>

Puedes consultar más información sobre las etiquetas vacías en el siguiente enlace.

3.- Usa minúsculas en los nombres de etiquetas, atributos y valores

A pesar de no afectar al funcionamiento de la página, es una recomendación escribir siempre las etiquetas, atributos y valores en minúsculas, ya que mejora la legibilidad del código.

Mal:

<SECTION>

<h1>Título</H1>

<P>Contenido</P>

<section>

Bien:

<section>

<h1>Título</h1>

<p>Contenido</p>

<section>

4.- Usa el atributo ALT en imágenes

Las imágenes deben incluir siempre dicho atributo, que proporciona, como su nombre indica, un texto alternativo a la imagen. Su uso mejora la accesibilidad ya que, por ejemplo, los lectores de pantalla se basan en este atributo para proporcionar un contexto a las imágenes.

Dicho valor debe ser lo suficientemente descriptivo al contenido. Si la imagen no tiene ninguna relevancia (como un icono, por ejemplo), es recomendable seguir incluyendo el atributo, aunque esté vacío.

Mal:

<img src="img/logo.png">

Bien:

<img src="img/logo.png" alt="Logo">

5.- Usa una indentación consistente

Esto es aplicable a cualquier lenguaje: un código con mala indentación o sin ella carece en gran medida de legibilidad.

Indentar es utilizar espacios a la derecha de cada línea con el objetivo de mejorar la legibilidad del código, por lo que es un aspecto importante y que deberías tener siempre en cuenta al programar.

En el modo de hacerlo tenemos varias opciones, mediante tabulaciones (Tab) o espacios (1, 2, o los que se prefieran). Esto ya es cuestión personal (yo personalmente prefiero Tabs), aunque podríamos tomar como referencia la guía de estilos de Google que recomienda dos espacios.

Sea como sea (con tabulaciones, dos, tres espacios,…) lo importante es que sea consistente en todo el código.

Mal:

<div>

<h1>Título</h1>

<ul>

<li>Elemento 1</li>

<li>Elemento 2</li>

<li>Elemento 3</li>

</ul>

</div>

Bien:

<div>

<h1>Título</h1>

<ul>

<li>Elemento 1</li>

<li>Elemento 2</li>

<li>Elemento 3</li>

</ul>

</div>

6.- Separa el contenido de la presentación

No uses estilos en línea. HTML es el contenido, y CSS proporciona la presentación visual de dicho contenido. No los mezcles.

Al mezclarlos, se consiguen páginas más lentas, difíciles de leer y mantener. Tener el estilo y el contenido separado ayudará al equipo de desarrollo a realizar cambios y mantenimiento de una forma mucho más eficiente.

Por ello, siempre es recomendable utilizar hojas de estilo externas, junto con clases para aplicar estilos según sea necesario.

Mal:

<p style="color: red;">Información</p>

Bien:

<p class="alerta">Información</p>

7.- Usa etiquetas semánticas y evita la Divitis

Se podría definir Divitis como esa mala práctica de usar divs para organizar todo el contenido de la página. Aunque funciona, empeora la legibilidad y sobrecarga el código.

Es más recomendable mantener nuestro código lo más limpio y reducido posible, reduciendo el uso de clases e id innecesarios para vincular estilos a un solo elemento.

Para ello, podemos utilizar los elementos estructurales de HTML5 de forma adecuada, como:

<header>, <aside>, <section>, <article>, <menu>, <footer>...

Mal:

<div id="encabezado"></div>

<div id="navegacion"></div>

<div id="contenido">

<div class="articulo"></div>

<div class="articulo"></div>

</div>

<div id="pie"></div>

Bien:

<header></header>

<nav></nav>

<main>

<article></article>

<article></article>

</main>

<footer></footer>

<https://www.juanmacivico87.com/buenas-practicas-html-escribir-codigo-limpio-mantenible/>