301 HTML - Introducción

¿Qué es un lenguaje de marcado?

Markup Languages son lenguajes diseñados para el procesamiento, definición y presentación de texto. En pocas palabras, podríamos decir que **los lenguajes de marcado se utilizan para presentar información a los usuarios en un *formato* específico, utilizando diferentes diseños y estilos.** Los elementos utilizados para especificar el *formato* se denominan **etiquetas**.

En resumen, defines elementos como etiquetas, y esas etiquetas se formatean en un nuevo y elegante documento.

¿Qué es HTML?

Definición

Según *MDN - Mozilla Developer Network* (sabrás mucho más sobre MDN a medida que avances en el curso, ya que contiene todos los documentos oficiales a los que nos referiremos), HTML - HyperText Markup Language **es el bloque de construcción más fundamental de la Web. Define el significado y la estructura del contenido web. Utiliza *etiquetas* como principales bloques de construcción para crear páginas web.**

Hemos mencionado las etiquetas varias veces hasta ahora, así que vamos a tratar este concepto a continuación.

Etiquetas y atributos

Etiquetas

Una **página web** es una colección de diferentes *tipos* de contenido. Algunos de los **tipos básicos de contenido** que casi cualquier página web tiene son: **texto**, **imágenes**, **vídeo** y **audio**. Todos los tipos de contenido están **representados por las etiquetas HTML correspondientes y se muestran en los navegadores**.

Podemos llegar a la siguiente conclusión: **Un documento HTML es una colección de etiquetas. Las etiquetas se utilizan para describir tipos específicos de contenido, como imágenes, párrafos, encabezados, pies de página y muchos más.

Las etiquetas suelen ir en pares: una etiqueta de apertura define el inicio de un bloque de contenido, y una etiqueta de finalización define el final de ese bloque de contenido.

Nota: En el siguiente párrafo, verás algunas explicaciones/**comentarios** colocados entre `<!-- Los comentarios van aquí -->`. Esta es la forma estándar de escribir comentarios en HTML. Más adelante hablaremos más sobre los comentarios, pero por ahora, ten en cuenta que **cualquier cosa colocada entre `<!-- -->` no es interpretada por un navegador, lo que significa que es visible para ti pero no para el navegador**.

```
<html>
<!-- aquí empieza la etiqueta (o tag) de html -->
...
</html>
<!-- aquí acaba el tag de html -->

<!-- ejemplo - paragraph tag -->
<p>some text will go here</p>
```

Te hemos dado sólo dos de las muchas etiquetas HTML, pero ya lo entiendes, ¿verdad? Todas las etiquetas se **colocan entre `<` y `/>` y la etiqueta de cierre también tiene la barra invertida**.

Recuerda esto también: *toda regla tiene una excepción*.

No todas las etiquetas tienen parte de cierre ya que algunas de ellas son **etiquetas de autocierre**.

Se representan en la siguiente estructura:

```
<img />
<!-- image tag -->

<br />
<!-- tag de salto de línea-->

<input />
<!-- input tag -->
```

Para recapitular, tenemos dos tipos diferentes de etiquetas:

- La mayoría de las etiquetas HTML tienen una parte de apertura y otra de cierre:

```
-- Opening Tag --              --Closing Tag--
<a href="google.com">Go to Google</a>
```

- sin embargo, algunas etiquetas son **autocierre** como es:

```
<img />
```

Atributos

Los Atributos HTML **modifican o definen una característica particular de una etiqueta**.

Según los recursos de Wikipedia, los atributos HTML son **palabras especiales utilizadas dentro de la etiqueta de apertura para controlar el comportamiento del elemento**. Un atributo **modifica la funcionalidad por defecto** de un tipo de elemento **o proporciona funcionalidad** a ciertos tipos de elementos que no pueden funcionar correctamente sin ellos.

Dicho esto, a nuestra etiqueta `img` le falta el atributo `src` (fuente) para poder funcionar correctamente. La parte `src` tiene que estar junto a la `img` colocada dentro de `<` y `>` y apuntará a la fuente específica donde se guarda una imagen. En el siguiente ejemplo, estamos cogiendo la imagen disponible online y pasándosela al `src`; en algún otro caso, podríamos utilizar una imagen almacenada en nuestro disco duro, en nuestro ordenador y mostraremos cómo hacerlo en una de las siguientes lecciones. Abre el siguiente CodePen y echa un vistazo al código.

```
<img src="https://devs.lol/uploads/2021/11/meme-dev-humor-when-you-are-in-love-with-programming-176.jpg">
```

Como podemos ver, en la parte que representa el navegador, está la imagen que queríamos que se mostrara.

Las etiquetas pueden tener **múltiples atributos**. Las imágenes también tienen un atributo `alt` en caso de que la imagen no se pueda mostrar. Los atributos también se utilizan para que los lectores de pantalla entiendan cuál es el contenido.

```
<img src="https://devs.lol/uploads/2021/11/meme-dev-humor-when-you-are-in-love-with-programming-176.jpg" alt="Programming meme">
```

Algunos atributos sólo funcionan con las etiquetas correspondientes. Por ejemplo, si intentas usar un atributo `src` en una etiqueta `a` (que usamos para mostrar enlaces) o `p` (para mostrar párrafos), no funcionará.

Vale, parece que hemos aprendido a escribir etiquetas correctamente.

La estructura básica del documento HTML

Primero echemos un vistazo a la estructura del siguiente documento HTML y luego desmitifiquémosla un poco.

```
<!DOCTYPE html>
<html>
        <head>
                <title>My first document</title>
                <meta charset="UTF-8" />
        </head>
        <body>
                <p>Hey!</p>
                ...
        </body>
</html>
```

- `<!DOCTYPE html>` Esta etiqueta indica que el lenguaje de marcado para el contenido de tu documento es HTML5. Como puedes ver, esta etiqueta es un poco diferente, no sigue la estructura estándar de etiquetas desde entonces. No te preocupes por ello; te acostumbrarás.
- Etiqueta `<html>`. La columna vertebral del documento HTML es la etiqueta `<html>`. Cada documento *HTML* tiene que tener esta etiqueta, y tiene que haber SOLO UNA etiqueta `<html>` por documento.
- La etiqueta `<head>`. La etiqueta `<head>` contiene información general sobre el documento. En esta etiqueta, más adelante, colocarás el enlace a tus *hojas de estilo* (que te permitirán dar estilo a tu documento). Suele contener las etiquetas *title* y *metadata*.
- Etiqueta `<title>`. Define el título del documento. Sólo hay un elemento title en el elemento head de un HTML. Este título sólo contiene texto y se muestra en la barra de título del navegador o en la pestaña de la página. - Etiqueta `<meta>` - se utiliza para definir metadatos. Su finalidad es ayudar a los navegadores a representar la página. En nuestro ejemplo, esta etiqueta contiene información sobre los conjuntos de caracteres permitidos en el documento (el conjunto de caracteres define qué caracteres especiales están permitidos, como á o ñ).
- Etiqueta `<body>` La etiqueta `<body>` contiene todos los elementos visibles que se presentarán a los usuarios. **Sólo puede haber un elemento `<body>` en un documento HTML.**

Estructura HTML a través del DOM tree

Ya sabes lo que es CodePen y cómo usarlo, y durante este módulo lo usaremos intensamente. Sin embargo, te animamos encarecidamente a que pruebes a utilizar tu entorno local para empezar a practicar, así que aquí te daremos un paso opcional: puedes crear tu primer miniproyecto práctico.

```
$ mkdir my-first-page
$ cd my-first-page
$ touch index.html
$ code .
```

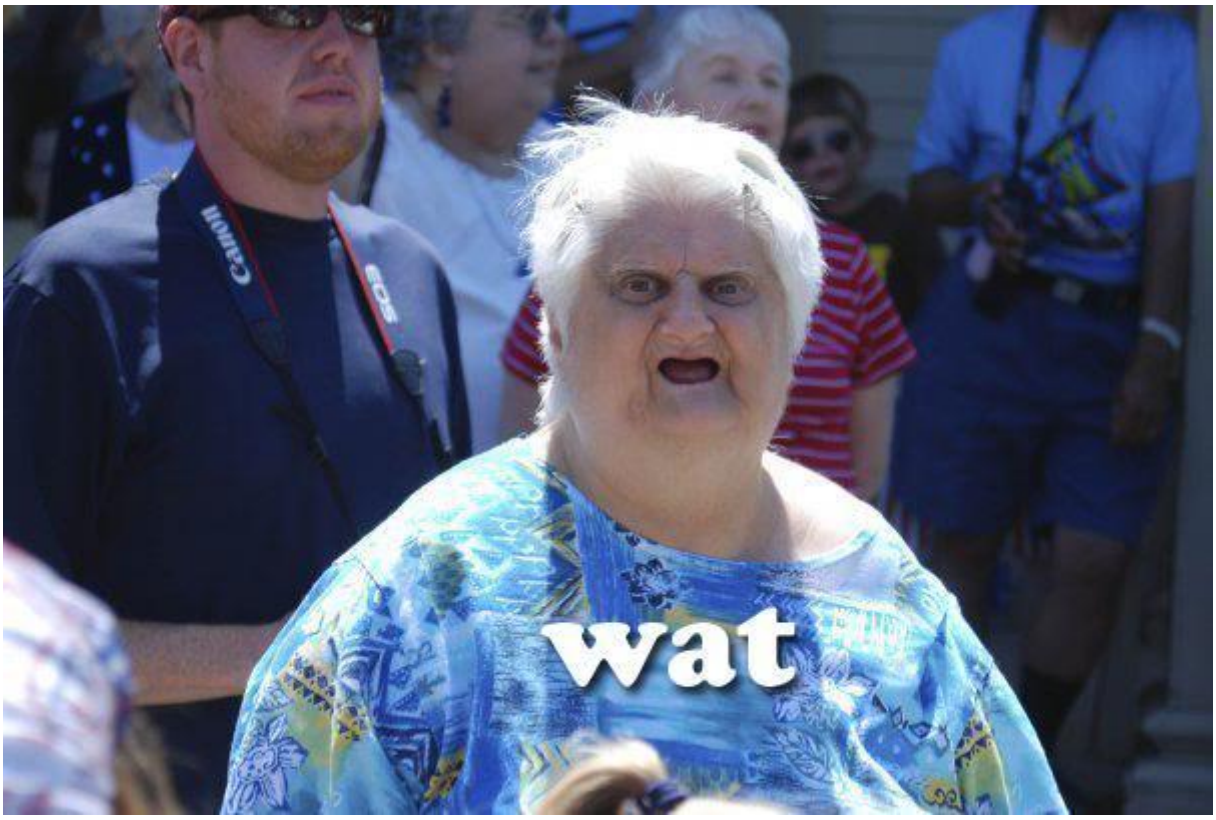
Para resumir, `mkdir` significa *make directory* y es el comando que usamos para crear una carpeta vacía en la terminal. `cd` significa *cambiar directorio* y la ruta que le sigue indica a dónde "vamos" a continuación - en este caso, estamos entrando en la carpeta vacía que hemos creado (`my-first-page`). `touch` se usa para *crear archivos* y lo que sigue después de este comando es el nombre del archivo recién creado_.

Después de eso, escribe el siguiente código dentro de `index.html` (no te recomendamos copiar-pegar ya que necesitas practicar):

```
<!DOCTYPE html>
<html lang="en">
        <head>
                <meta charset="UTF-8" />
                <title>Simple Web Page</title>
        </head>
        <body>
                <div>
                        <p>Hello!</p>
                </div>
        </body>
</html>
```

Ahora te daremos una explicación super complicada del Document Object Model (DOM) y después te la simplificaremos:

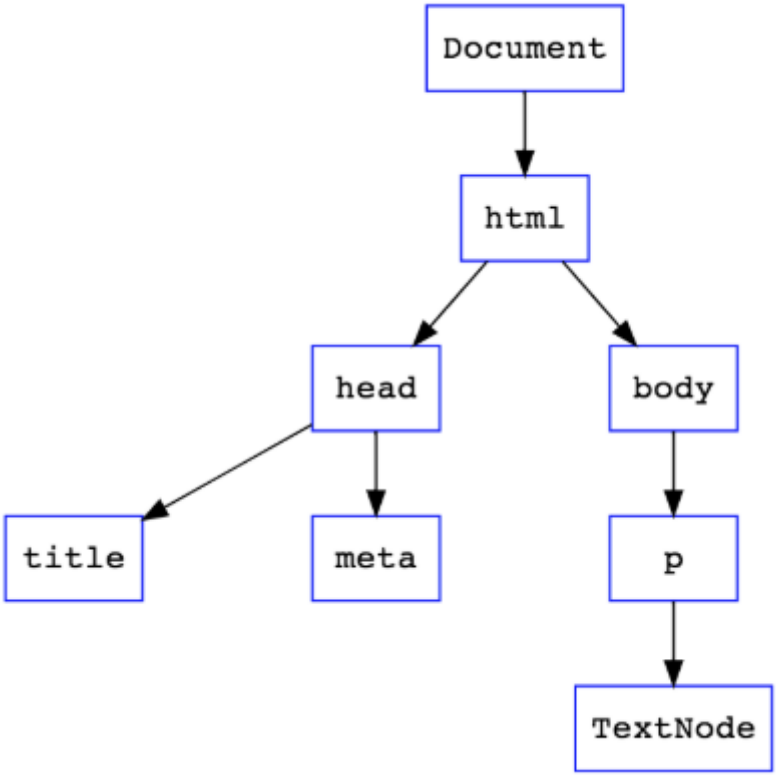
> El Modelo de Objetos de Documento (DOM) es una interfaz de programación de aplicaciones multiplataforma e independiente del lenguaje que trata un documento HTML, XHTML o XML como una estructura de árbol en la que cada nodo es un objeto que representa una parte del documento. Los objetos pueden manipularse mediante programación, y cualquier cambio visible que se produzca como resultado puede reflejarse en la visualización del documento.



Como te prometimos:

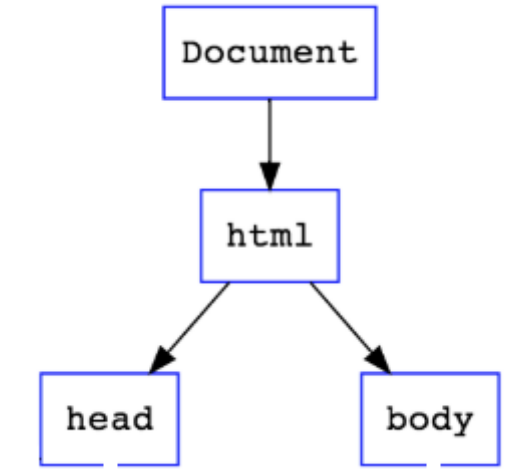
El **DOM (Document Object Model)** representa cómo el navegador lee el documento HTML. Después de que el navegador lee el documento HTML, crea un árbol representacional llamado Modelo de Objetos del Documento y define cómo se puede acceder a ese árbol.

Basándonos en el fragmento de código anterior, nuestro navegador construiría el siguiente árbol DOM:



En el DOM, todo es un **nodo**. El documento, el cuerpo, todos los demás elementos, los atributos de los elementos, el contenido dentro de los elementos, son todos *nodos*. Cubriremos esto en profundidad en las próximas lecciones.

El árbol DOM



Hemos visto que sólo hay un elemento `<html>` lo que implica que todos los demás elementos están anidados dentro de él. Esto hace que la etiqueta `html` sea el **padre** de sus *descendientes directos*, que son las etiquetas *head* y *body* y el *ancestro* de todos los demás elementos. Al mismo tiempo, puedes observar, siguiendo la estructura del árbol genealógico, que las etiquetas `<head>` y `<body>` son elementos **hijos** de su elemento padre `html` pero, al

mismo tiempo, son **hermanos** ya que tienen *padre* mutuo. Ten en cuenta estas relaciones; las utilizarás más adelante.

DOM y la estructura de tu código

Es esencial escribir correctamente tu código. No basta con cerrar todas las etiquetas que deban cerrarse; es **necesario respetar la indexación correcta**. Por indexación, nos referimos a lo siguiente:

Cuando miras en tu documento HTML, cada etiqueta que se abre y cada etiqueta que se cierra necesitan estar alineadas, y necesitan respetar el orden - algunas están anidadas dentro de otras.

Dicho esto, fíjate en la indexación en el fragmento de código que proporcionamos antes - no tener indentación no romperá tu código, pero es una práctica súper mala ya que **laindexaciónindentación debería ser la imagen perfecta de tu DOM**.