HTML5

Introducción a HTML5

HTML son las siglas de ***HyperText Markup Language*** o Lenguaje de Marcado de Hipertexto. Es un lenguaje de basado en etiquetas.

El lenguaje HTML se introduce en un documento tipo texto con extensión **.html**. Este HTML es interpretado por un *"user agent"*, que en la mayoría de ocasiones se corresponde con un navegador web. Sin embargo existen otro tipo de *"user agent"* como podría ser los robots de indexación de los motores de búsqueda, etcétera…

El objetivo del HTML es describir la estructura del documento e indicar el **contenido semántico** de cada elemento que forma parte del documento.

Para ello se emplea el uso de **etiquetas**, que son la base del lenguaje HTML. Existen muchas etiquetas y cada una se utiliza para contener información y darle un significado a dicha información.

Esta es la razón por la que el lenguaje HTML se considera semántico. Describe el contenido de la información con el uso de las etiquetas. Para gestionar el aspecto visual de esa información se debe utilizar el lenguaje CSS.

El estándar HTML contiene un número concreto de etiquetas, algunas plenamente utilizables y otras marcadas como obsoletas. Sin embargo, aunque no se genere ningún error, en **HTML** no se debe utilizar cualquier palabra como etiqueta:

```
<!-- INCORRECTO -->
<etiqueta>contenido</etiqueta>

<!-- Correcto -->
<p>Párrafo</p>
```

En el documento HTML debe aparecer información correctamente individualizada, de modo que al leer una página HTML comprendamos su significado, y si queremos cambiar la apariencia, lo hagamos en el documento CSS. Esto es lo que comunmente se conoce como **separación de la presentación del contenido**.

```
Hola, quiero resaltar esta <b>palabra</b>.
```

En este caso se está utilizando la etiqueta `<b>` de HTML4 y anteriores para poner en negrita un texto. Se está utilizando una propiedad de presentación (visual) en el HTML, algo que no se debe hacer en HTML5. La misión de HTML5 es mantener sólo contenido e información semántica en HTML5. Por dicha razón, la forma de hacerlo en HTML5 es utilizando una etiqueta que añade esa información semántica como es `<strong>`:

```
Hola, quiero resaltar esta <strong>palabra</strong>.
```

En los navegadores, hay varias formas de acceder al código HTML de la página:

- Pulsando la combinación de teclas `Ctrl + U`. Te aparecerá el código fuente tal cuál lo recibe el navegador.
- Pulsando `Ctrl + Shift + I` aparecerá la consola del navegador.

Estructura de una etiqueta HTML

La estructura de las etiquetas HTML tiene el siguiente formato:

```
<etiqueta atributo="valor">contenido</etiqueta>
```

Por norma general, las etiquetas deben cerrarse para indicar donde finaliza su contenido. Sin embargo, aquellas etiquetas sin contenido textual como por ejemplo `<hr>` o `<img>` o sin elementos anidados no tienen etiqueta de cierre.

Formato de una etiqueta

La parte esencial de una etiqueta HTML es lo que se denomina la **etiqueta de apertura**. Se trata de escribir el nombre de la etiqueta en cuestión, colocándola entre los carácteres `<` y `>`. Aunque el navegador lo permita si no es así, las etiquetas HTML se deberían escribir siempre en minúsculas.

Para el código HTML sea válido no se debe utilizar cualquier palabra como etiqueta, pese a que el lenguaje HTML es permisivo en ese sentido.

La mayoría de las etiquetas requieren que se especifique un **cierre de etiqueta** para saber donde termina de actuar. Se caracteriza en que se escribe igual que la etiqueta de apertura, pero con la barra `/` inmediatamente después del `<`.

Atributos

En algunas etiquetas HTML, existen algunos atributos específicos (que pueden ser **opcionales u obligatorios**) y otros que son atributos globales.

Los atributos determinan cierta información sobre la etiqueta y generalmente van asociados a un **valor determinado**. Este atributo se escribe después del nombre de la etiqueta, separándola por espacio y antes del carácter `>`:

```
<strong id="dato">Contenido</strong>

<strong id="dato" class="clase1" lang="es">Contenido de texto</strong>
```

Una etiqueta puede tener uno o varios pares de *atributo-valor*, pero nunca se debe repetir el mismo atributo en una misma etiqueta varias veces, ya que **sobreescribiría** al anterior. El orden de los atributos no es importante.

Aunque los valores pueden ir rodeados por comillas simples, se recomienda escribir el valor siempre entre **comillas dobles**.

Existen 3 tipos de atributos dependiendo de sus valores:

1. **Conjunto de valores**: Son aquellos atributos en los que sólo se permiten unos valores concretos. Cualquier otro valor diferente, no será válido.

2. **Valores libres**: Son los atributos en los que puedes especificar un valor libremente, como una dirección URL o un texto, y no existe una serie de valores específicos para escribir.

3. **Valores booleanos**: Son los atributos que deben tener un valor verdadero (true) o un valor falso (false). En HTML5 un atributo sin valor (solo el atributo) significa verdadero y si se omite el atributo se considera falso.

Contenido de la etiqueta

Una etiqueta puede contener desde un fragmento de texto hasta un grupo de etiquetas:

```
<div id="pagina">
  <!-- Bloque de etiquetas -->
```

```
    <strong>Contenido importante</strong>
  </div>
```

Comentarios en HTML

Para introducir estos comentarios en el código HTML, basta con escribir los fragmentos de texto `<!--` y `-->` entre el comentario en cuestión que queramos incluir.

⚠ **NOTA**: Los comentarios HTML son **visibles** si se accede al código fuente del documento.

Elementos en bloque vs elementos en línea

La distinción entre **elementos en bloque** frente a **elementos en línea** se utiliza en las especificaciones de HTML hasta la 4.01. HTML5 hereda esta noción, en la que los elementos de estructura son elementos de bloque (como por ejemplo `<div>` o `<p>`) mientras que los elementos de formato de texto (como `<strong>` o `<span>`) son elementos en línea.

Un **elemento de bloque** ocupa todo el espacio de su elemento padre (el contenedor), creando así un *"bloque"*. Los navegadores suelen mostrar el elemento a nivel de bloque con un salto de línea antes y después del elemento.

⚠ **NOTA:** Los elementos de bloque sólo deben aparecer dentro del elemento `<body>`.

En lo que respecta a anidación, los elementos de tipo bloque pueden contener otros elementos de tipo bloque, elementos de tipo en línea y de tipo texto.

Un **elemento en línea** ocupa sólo el espacio delimitado por las etiquetas que definen el elemento en línea. De forma predeterminada, los elementos en línea no comienzan con la nueva línea.

Los elementos de tipo en línea pueden contener otros elementos de tipo en línea y de tipo texto pero no elementos de tipo bloque.

Lista de elementos en bloque y lista de elementos en línea.

Etiquetas obsoletas

Con el paso del tiempo y la transición desde versiones anteriores a HTML5 (por ejemplo, desde HTML4 o XHTML), hay muchas etiquetas HTML que han sido marcadas como obsoletas.

Además de las etiquetas, hay comportamientos que se van marcando como obsoletos aunque no son incorrectos si se utilizan como por ejemplo indicar el atributo de tipo `type="text/css"` en un elemento `<style>`.

Atributos comunes en HTML

Los atributos son palabras clave de texto que modifican ligeramente el comportamiento de la etiqueta que lo contiene. Los atributos comunes son atributos que pueden estar prácticamente en **cualquier etiqueta HTML**.

Más información en el documento "HTML: The Living Standard"

Atributos CSS

- **id**: Establece un identificador único a la etiqueta HTML. Sólo el mismo nombre una vez por documento.
- **class**: Establece una clase (género) a una etiqueta HTML. Puede repetirse por documento.
- **style**: Aplica propiedades CSS directamente al elemento HTML en cuestión.

El atributo *id* (identificador)

En HTML, podemos darle un identificador a una etiqueta HTML y de esta forma darle un nombre. Simplemente, añadimos el atributo `id` y colocamos el nombre como valor de ese atributo.

```
<div id="header">
  <div>Aquí irá un anuncio</div>
  <div id="articulo">Aquí irá el contenido de texto del artículo</div>
  <div>Aquí irá un anuncio</div>
</div>
```

Ese identificador único tiene unas normas:

- No debe empezar nunca por un número, pero puede contenerlos más adelante.
- El texto debe estar en *kebab-case*, es decir, minúsculas y separados por un guión.
- Es preferible que no contenga carácteres raros, acentuados o emojis.
- En un documento HTML no pueden existir dos elementos con el mismo id.

Los identificadores se suele utilizar para zonas específicas que no se van a repetir, como **header** o **footer**.

La recomendación es utilizar clases en vez de identificadores **siempre que sea posible**.

El atributo *class* (clase)

Las clases funcionan de una forma muy similar a los identificadores, pero son mucho más flexibles ya que no tiene la limitación de ser únicos. La idea de las clases es establecer géneros o tipos de etiquetas, a los que les asociemos características comunes.

```
<div id="pagina">
  <div class="anuncio">Aquí irá un anuncio</div>
  <div id="articulo">Aquí irá el contenido de texto del artículo</div>
  <div class="anuncio ultimo">Aquí irá un anuncio</div>
</div>
```

La ventaja de utilizar clases es que mediante CSS se puede aplicar un estilo concreto a todas las etiquetas HTML que tengan esa clase.

Además, a diferencia de los identificadores, una etiqueta puede tener múltiples clases diferentes separadas por un espacio. Si se indican múltiples atributos `class` en la misma etiqueta, como ya se ha indicado el navegador ignorará los primeros y sólo utilizará el valor del último `class`.

El atributo *style* (estilos en línea)

El atributo `style` es un atributo que se utiliza en las etiquetas HTML para incrustar código CSS directamente en la propia etiqueta.

```
<p style="background: indigo; color: white;">Esto es un mensaje con estilos CSS</p>
```

En la mayoría de los casos, no se recomienda añadir los estilos de esta forma ya que suele ser considerado una mala práctica. Es mejor colocar el código CSS en un documento .css separado.

Atributos de idioma

- **lang**: Indica el idioma del contenido de la etiqueta HTML.
- **translate**: Indica si el contenido de la etiqueta se debería traducir o no.

- **dir**: Establece la direccionalidad del texto.

El atributo *lang* (idioma)

Mediante el atributo lang podemos indicar el **idioma** del contenido de la etiqueta. El valor de dicho atributo tendrá que ser el código ISO 639-1 del idioma al que queremos hacer referencia.

Aunque en principio podemos hacer esto en cualquier etiqueta HTML, es **obligatorio** hacerlo en la etiqueta `<html>` que es la etiqueta que abarca todo el documento, estableciendo el idioma del mismo:

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

El atributo *translate* (traducción)

En las etiquetas HTML se puede indicar el atributo `translate`, el cuál acepta los valores 'yes' y 'no'. Por defecto, aunque este atributo no sea añadido en una etiqueta, el valor por **defecto** que tiene es 'yes'. Por lo tanto, todas las etiquetas están marcadas como «traducibles».

```
<p>
  Hace algunos días fuí a ver la nueva película de <span translate="no">StarWars</span>.
</p>
```

Por ejemplo, mediante este atributo podemos indicar a herramientas como 'Google Translate' que no traduzcan una etiqueta que puede estar indicando el título de un libro, que debe conservarse tal cual.

El atributo *dir* (direccionalidad)

Existe un atributo `dir` que permite al desarrollador indicar la **direccionalidad** del texto en el documento. Esto es ideal para idiomas en los que se escribe o lee de derecha a izquierda, en lugar de izquierda a derecha.

El valor por **defecto** de este atributo es 'ltr' (left to right, de izquierda a derecha), pero podemos modificarlo y establecer el valor 'rtl' (right to left, de derecha a izquierda).

Otros atributos comunes

- **title**: Mensaje mostrado en un tooltip (aviso emergente) al mover el ratón encima.
- **data-***: Metadatos en la propia etiqueta. Se puede usar cualquier nombre con prefijo 'data-'.
- **accesskey**: Combinación de teclas que puede pulsar el usuario para activar el elemento.

El atributo *title* (título)

Aunque se suele hacer sobre todo en las etiquetas de imágenes `<img>`, en la mayoría de las etiquetas HTML podemos indicar el atributo `title` para especificar un mensaje de texto que aparezca cuando el usuario detenga el ratón sobre el elemento un instante.

```
<figure>
  <img src="https://www.w3.org/html/logo/img/html5-display.png"
       alt="HTML5 Logo"
       title="¡Este es el logo de HTML5!" />
  <figcaption title="Pie de foto">One Web W3C for All</figcaption>
</figure>
```

Aunque los atributos `alt` y `title` en las etiquetas `<img>` tienen objetivos parecidos:

- El atributo `alt` debe ser un texto alternativo que describa la imagen (en el caso de que no se pueda ver visualmente)
- El atributo `title` puede describir la imagen, pero no tiene porque ser una descripción alternativa.

El atributo *data-* (metadatos)

En un documento HTML, la mayoría de los **metadatos** (información adicional) se incluyen en el interior de la etiqueta `<head>` del documento HTML. Sin embargo, también se pueden incluir metadatos en una etiqueta HTML a través de un atributo que comienza con el prefijo `data-`.

Los atributos de datos personalizados están destinados a almacenar datos personalizados, estado, anotaciones y similares, privados a la página o aplicación, para los que no hay atributos o elementos más apropiados.

Desde Javascript, si queremos hacer referencia a estos elementos, podemos hacerlo de varias formas:

```
<!-- HTML5 -->
<div class="spaceship" data-ship-id="92432"
     data-weapons="laser 2" data-shields="50%"
     data-x="30" data-y="10" data-z="90">
</div>
```

```
// JavaScript
const spaceship = document.getElementsByClassName("spaceship");

console.log(spaceship[0].dataset.weapons) // "laser 2"
console.log(spaceship[0].getAttribute("data-shields")); // "50%"
```

En la primera, accedemos a la propiedad `.dataset` donde tendremos una lista de propiedades dependiendo de los diferentes atributos con prefijo `data-` que tenga el elemento. Por otro lado, podemos también utilizar el método `.getAttribute()` del DOM.

El atributo *accesskey* (atajo)

En HTML es posible añadir el atributo `accesskey` para indicar un atajo de teclado que puede pulsar el usuario para activar ese elemento. De esta forma, al pulsar la combinación `Alt + <tecla>` se activa ese elemento:

```
<form>
  <input accesskey="N" placeholder="Nombre (ALT+N)" />          <!-- Campo de datos -->
  <input accesskey="A" placeholder="Apellidos (ALT+A)" />       <!-- Campo de datos -->
</form>
```

⚠ Debido a que no está estandarizado entre sistemas operativos y navegadores no se aconseja su uso.

Estructura de un documento HTML

Un **documento HTML** debe estar bien formado sintácticamente para que el navegador pueda leerlo correctamente:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Título del documento</title>
```

```
</head>
<body>
  <!-- ... -->
</body>
</html>
```

El `<!DOCTYPE>` (tipo de documento)

HTML es una aplicación **SGML** o *Standard Generalized Markup Language*. Es por esto que es necesario que la primera línea de un documento contenga la indicación del lenguaje de etiquetas utilizado.

El `!DOCTYPE` o **tipo de documento** es una etiqueta especial que se escribe en la primera línea del documento HTML y que realiza esa función de indicar el tipo de etiquetas utilizado. Esta etiqueta no se considera una etiqueta HTML.

Es una etiqueta **obligatoria**, pero de no indicarla, la página web probablemente continuaría visualizándose de forma correcta. En el caso de no indicar el tipo de documento en una página HTML, el navegador entra en lo que se llama **'Quirk mode'** (modo peculiar o modo no estándar), donde se activa un modo de retrocompatibilidad con páginas antiguas, por lo que procesará de forma diferente muchas etiquetas HTML o propiedades CSS.

Para los documentos **HTML5** tiene esta forma:

```
<!DOCTYPE html>
```

En versiones anteriores, como **HTML4 o XHTML**, el tipo de documento se especificaba en la primera línea de una forma más compleja:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

La etiqueta `<html>`

El elemento `<html>` es el elemento raíz de un documento HTML. Se coloca después de la declaración del tipo de documento y abarca todo el documento.

Entre los atributos globales, el uso del atributo `lang` **no es obligatorio** pero sí recomendable para indicar al navegador el idioma empleado en el documento. Este atributo también es utilizado por los motores de búsqueda y por los navegadores con síntesis de voz para personas con discapacidad visual.

La etiqueta `<head>` (metadatos)

Cuando hablamos de la etiqueta `<head>` de un documento HTML, muchas veces la denominamos **cabecera HTML**. Sin embargo, esta cabecera no es una parte visual de la página web, sino una parte de nuestro código HTML donde se incluyen ciertas etiquetas de metadatos, es decir, unas etiquetas que establecen ciertos datos que no tienen que verse necesariamente de forma visual.

Algunos metadatos son el **título**, la **descripción** de la página, el **icono** o favicon, etcétera…

Más información en el documento "HTML: The Living Standard"

Etiqueta `<title>`

Esta etiqueta es **obligatoria** y sólo puede haber **una etiqueta** de este tipo en un mismo documento. Es el título del documento y aparece como título en la ventana o en las pestañas del navegador.

Además, este título se utiliza como enlace y en los resultados de los motores de búsqueda.

Más información en el documento "HTML: The Living Standard"

Etiqueta `<meta>`

El elemento `<meta>` permite guardar metadatos del documento. Puede haber varios elementos, uno o ningún elemento de este tipo.

El más importante es el metadato que indica el tipo de codificación. Es importante indicar la codificación **justo después de la etiqueta de apertura** `<head>` porque esa codificación va a afectar a todos los elementos posteriores. Actualmente la codificación utilizada es **UTF-8**.

```
<meta charset="UTF-8">
<meta name="description" content="Descripción del document">
<meta name="author" content="Autor del documento">
```

Más información en el documento "HTML: The Living Standard"

Etiqueta `<link>`

El elemento `<link>` permite crear enlaces a lugares externos del documento. También permite mostrar un icono en la barra de direcciones del navegador.

```
<link rel="stylesheet" href="/home.css">
<link rel="icon" href="favicon.ico">
```

Más información en el documento "HTML: The Living Standard"

Etiqueta `<style>`

El elemento `<style>` permite declarar los estilos CSS que solo se aplicarán al documento actual. No es necesario indicar el atributo `type="text/css"` porque se considera que CSS es el tipo por defecto.

```
<style>
  .autor {
    text-transform: uppercase;
  }
</style>
```

Más información en el documento "HTML: The Living Standard"

Etiqueta `<script>`

El elemento `<script>` permite declarar los script JavaScript que solo se aplicarán en el documento actual. No es necesario indicar el atributo `type="text/javascript"` porque se considera que JavaScript es el tipo por defecto.

```
<script>
  alert ("Hello World!");
</script>
```

Más información en el documento "HTML: The Living Standard"

La etiqueta `<body>` (cuerpo)

El elemento `<body>` incluye todos los elementos del contenido del documento. Su apertura se sitúa justo después del cierre del encabezado `</head>`.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Título del documento</title>
</head>
<body>
  <!-- ... -->
</body>
</html>
```

Contenedores semánticos

Un contenedor sirve para estructurar un documento HTML. Dentro de un contenedor se puede incluir contenido muy variado como texto, imágenes, formularios, etcétera…

Por un lado hay contenedores más genéricos sin contenido semántico definido, como `<div>` o `<span>` y luego hay contenedores "semánticos" que están dedicados a un contenido específico como `<header>`, `<footer>` o `<article>`.

Elemento `<div>`

El elemento `<div>` es uno de los contenedores más antiguos de HTML. Permite incluir en su interior todo tipo de contenido, incluido otros elementos `<div>`.

Es un contenedor neutro, y aunque se puede utilizar para estructurar un documento, es preferible utilizar contenedores "semánticos" acordes a la información que contienen.

```
<!-- Contenedores neutros  -->
<div id="article">
  <div id="header">
    <!-- Título -->
  </div>
  <div id="main">
    <!-- Cuerpo -->
  </div>
  <div id="footer">
    <!-- Pie -->
  </div>
</div>

<!-- Contenedores semánticos  -->
<article>
  <header>
    <!-- Título -->
  </header>
  <p><!-- Cuerpo --></p>
  <footer>
    <!-- Pie -->
  </footer>
</article>
```

Más información en el documento "HTML: The Living Standard"

Elemento `<span>`

El elemento `<span>` normalmente se utiliza para formatear de manera particular un texto dentro de un párrafo de texto, como por ejemplo:

```
<style>
  .fondo-gris{
    background-color: #eee;
  }
</style>

<p>Unde consequatur amet itaque. Velit <span class="fondo-gris">rerum sed iusto</span> quae
consectetur voluptas temporibus.</p>
```

Más información en el documento "HTML: The Living Standard"

Elemento `<header>`

El elemento `<header>` permite insertar una zona de visualización para los encabezados:

- A nivel de **página**: situado normalmente en la parte superior para contener un logo, una barra de navegación, etcétera…
- A nivel de **contenido**: como podría ser el encabezado de un artículo.

```
<article>
  <header>
    <h2>....</h2>
  </header>
  <p>....</p>
  <footer>
    <p>....</p>
  </footer>
</article>
```

Desde HTML5.1, se pueden anidar elementos `<header>` y `<footer>` dentro de otro elemento `<header>` si los dos primeros elementos se incluyen en el mismo elemento padre:

```
<article>
  <header>
    <h2>....</h2>
    <aside>
      <header>
        <h3>....</h3>
      </header>
      <p>....</p>
      <footer>
        <p>....</p>
      </footer>
    </aside>
  </header>
  <p>....</p>
</article>
```

Más información en el documento "HTML: The Living Standard"

Elemento `<footer>`

El elemento `<footer` permite insertar una zona de visualización para los pies de página. Se puede utilizar en los mismos niveles que un elemento `<header>`.

Sin embargo, la utilización de `<footer>` no implica forzosamente el uso de `<header>`.

Más información en el documento "HTML: The Living Standard"

Elemento `<aside>`

El elemento `<aside>` permite mostrar un contenido relacionado con un contenido principal al que se le asocia, como podría ser una barra de desplazamiento, una zona para *advertising*, etcétera…

Más información en el documento "HTML: The Living Standard"

Elemento `<nav>`

El elemento `<nav>` se utiliza para visualizar una barra de navegación con enlaces a otras secciones del propio documento o enlaces a otros documentos.

No es obligatorio incluir una barra de navegación ni tampoco está prohibido que haya varios elementos `<nav>` en un mismo documento HTML. Por ejemplo es posible tener un elemento `<nav>` dentro de un elemento `<header>` y otro elemento `<nav>` dentro de un elemento `<footer>`.

Más información en el documento "HTML: The Living Standard"

Elemento `<main>`

El elemento `<main>` permite indicar el contenido principal del documento. Este contenido debe ser único y no repetirse en el documento.

Además, no debe utilizarse en el interior, como elemento incluido, de elementos como `<article>`, `<aside>`, `<footer>`, `<header>` o `<nav>`.

Más información en el documento "HTML: The Living Standard"

Elemento `<section>`

El elemento `<section` permite agrupar los elementos que comparten una misma temática.

Esto permite agrupar en un mismo elemento un contenido estructurado, con su encabezado y su pie de página. La utilización de varios elementos `<section>` facilita estructurar un documento en secciones distintas.

Más información en el documento "HTML: The Living Standard"

Elemento `<article>`

El elemento `<article` permite insertir un contenido autónomo, es decir, un contenido reutilizable en cualquier parte del documento. Como indica su nombre, su uso más habitual es la creación de artículos en un blog.

Más información en el documento "HTML: The Living Standard"

Validación de errores en HTML

En los lenguajes de marcas como HTML, los navegadores son más permisivos, ya que en el caso de encontrar un error, intentan «deducir» lo que realmente se quería indicar y continuan con la carga del documento.

En nuestro código HTML podemos tener varios tipos de problemas:

- **Sintaxis**: El código está mal escrito y es incorrecto. El navegador podría no mostrar correctamente ciertos detalles.
- **Accesibilidad**: El código no tiene porque estar mal escrito o ser incorrecto, sin embargo, tiene problemas que harán que no se vean bien en determinados dispositivos o que un usuario invidente o similar no pueda utilizarla.
- **Posicionamiento (SEO)**: El código no tiene porque estar mal escrito o ser incorrecto, sin embargo, los buscadores como Google no aceptarán favorablemente la página y podría tener un rendimiento menor de posicionamiento SEO.
- **Rendimiento**: El código no tiene porque estar mal escrito o ser incorrecto, sin embargo, el navegador puede ser lento a la hora de cargarlo o tardar en realizar sus tareas.
- **Usabilidad**: El código no tiene porque estar mal escrito o ser incorrecto, sin embargo, la forma en la que se disponen los elementos o está pensado puede frustrar al usuario final que utiliza la web.

Para asegurarnos de que nuestro código está correctamente escrito, podemos utilizar un **Validador HTML**, que no es más que un sistema que analiza nuestro código y nos dice el número de errores que tenemos, junto a una breve descripción del mismo para facilitar su corrección.

Este proceso de validación se puede realizar mediante la herramienta oficial HTML Validator de W3C u otras vías de validación mediante plugins para el IDE o paquetes NPM.

Enlaces de interés

- https://html.spec.whatwg.org/
- https://www.w3.org/TR/
- https://lenguajehtml.com/html/
- https://developer.mozilla.org/es/docs/Web/HTML
- https://w3schools.com/html/
- https://htmlreference.io/
- https://html.com/
- https://cheatsheets.shecodes.io/html
- https://overapi.com/html
- https://htmlcheatsheet.com/
- https://devhints.io/
- https://internetingishard.netlify.app/html-and-css/
- https://theodinproject.com/
- https://roadmap.sh/frontend
- https://caniuse.com/
- https://jsfiddle.net

Licencia