HTML5

Introducción a HTML5

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

El código HTML se escribe en un documento de 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 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 emplean las **etiquetas**, que son la base del lenguaje HTML. Existen muchas etiquetas y cada una se utiliza para contener información y darle un **significado semántico** 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>.
```

Del mismo modo, en lugar de usar `<i>` para poner un texto en cursiva, en HTML5 se debería usar `<em>` para marcar énfasis:

```
Hola, quiero enfatizar esta <em>palabra</em>.
```

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, existen algunas etiquetas que no requieren un cierre explícito, como `<hr>` o `<img>`, ya que no contienen contenido textual o elementos anidados. Estas etiquetas se conocen como **etiquetas vacías o etiquetas autocontenidas**.

En versiones más antiguas de HTML, era común ver estas etiquetas con un cierre automático, como `<img />`, pero en HTML5 esto no es necesario.

Formato de una etiqueta

La parte esencial de una etiqueta HTML es lo que se denomina la **etiqueta de apertura**.

Consiste en escribir el nombre de la etiqueta en cuestión, colocándolo entre los caracteres `<` y `>`. Aunque los navegadores pueden procesar etiquetas que no siguen este formato estrictamente, las etiquetas HTML deben escribirse siempre en **minúsculas** para cumplir con las buenas prácticas y con la especificación del estándar HTML.

Para que el código HTML sea válido, no se puede utilizar cualquier palabra como etiqueta, aunque el lenguaje HTML es permisivo en algunos aspectos y los navegadores puedan interpretar etiquetas no estándar.

La mayoría de las etiquetas requieren un **cierre de etiqueta** para indicar dónde termina su efecto. El cierre de la etiqueta se caracteriza por el uso del mismo nombre de la etiqueta de apertura, pero precedido por una barra `/`, inmediatamente después del `<`:

```
<p>Este es un párrafo</p>
```

Atributos

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

Los atributos proporcionan **información adicional** sobre una etiqueta y generalmente van acompañados de un **valor determinado**. Los atributos se colocan después del nombre de la etiqueta, separados por un 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, ya que el valor posterior **sobreescribiría** al anterior. El orden de los atributos no es importante.

Aunque los valores de los atributos pueden ir entre comillas simples o dobles, la convención es utilizar **comillas dobles** para mayor consistencia y legibilidad.

Existen 3 tipos principales de atributos según el tipo de valores que aceptan:

1. **Conjunto de valores**: atributos que sólo aceptan un conjunto predefinido de valores. Cualquier otro valor será inválido.
2. **Valores libres**: atributos donde puedes especificar un valor libremente, como una URL o texto, sin un conjunto predefinido de valores.
3. **Valores booleanos**: atributos que representan un valor verdadero o falso. En HTML5, si el atributo está presente sin un valor, se considera como verdadero (`true`), y si se omite, se interpreta como falso (`false`).

```
<!-- Ejemplo de atributo con un conjunto de valores válidos -->
<input type="email" placeholder="Introduce tu correo">

<!-- Ejemplo de atributo con valores libres -->
<a href="https://www.ejemplo.com">Visita nuestro sitio web</a>

<!-- Ejemplo de atributo booleano -->
<input type="checkbox" checked> Opción seleccionada por defecto
<br>
<button disabled>Botón deshabilitado</button>
```

Contenido de la etiqueta

Una etiqueta HTML puede contener desde un simple fragmento de texto hasta un grupo de etiquetas anidadas. Este contenido puede incluir tanto texto plano como otras etiquetas HTML que estructuren o den formato al contenido:

```
<div id="pagina">
  <!-- Bloque de etiquetas anidadas -->
  <strong>Contenido importante</strong>
</div>
```

Comentarios en HTML

Para introducir comentarios en el código HTML, se utilizan los delimitadores `<!-- y -->`. Todo lo que se encuentre entre estos delimitadores será considerado un comentario y no será visible en la página web, aunque sí estará presente en el código fuente.

```
<!-- Este es un comentario en HTML -->
<p>Este es un párrafo visible.</p>
```

> ⚠️ Los comentarios en HTML son **visibles** si se accede al código fuente del documento a través del navegador.

Elementos en bloque vs elementos en línea

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

Un **elemento de bloque** ocupa todo el ancho de su elemento padre (el contenedor), creando un *"bloque"*. Los navegadores suelen mostrar los elementos de bloque con un salto de línea antes y después de ellos.

En términos de anidación, los elementos de tipo bloque pueden contener otros elementos de tipo bloque, elementos en línea y texto.

Por otro lado, un **elemento en línea** sólo ocupa el espacio delimitado por el contenido de sus etiquetas. De forma predeterminada, los elementos en línea no causan un salto de línea.

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

Elementos de bloque

- `<address>`
- `<article>`
- `<aside>`
- `<audio>`
- `<blockquote>`
- `<canvas>`
- `<dd>`
- `<div>`
- `<dl>`
- `<fieldset>`
- `<figcaption>`
- `<figure>`
- `<footer>`
- `<form>`
- `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`, `<h6>`
- `<header>`
- `<hgroup>`
- `<hr>`
- `<li>`
- `<main>`
- `<nav>`
- `<noscript>`
- `<ol>`
- `<output>`
- `<p>`
- `<pre>`
- `<section>`
- `<table>`
- `<tfoot>`
- `<ul>`
- `<video>`

Elementos en línea

- `<a>`
- `<abbr>`
- `<acronym>`
- `<b>`
- `<bdo>`
- `<big>`
- `<button>`
- `<cite>`
- `<code>`
- `<dfn>`
- `<em>`
- `<g>`
- `<i>`
- `<img>`

- `<input>`
- `<kbd>`
- `<label>`
- `<map>`
- `<object>`
- `<samp>`
- `<script>`
- `<select>`
- `<small>`
- `<span>`
- `<strong>`
- `<sub>`
- `<sup>`
- `<textarea>`
- `<time>`
- `<tt>`
- `<var>`

Etiquetas obsoletas

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

Además de las etiquetas, también hay comportamientos que se consideran obsoletos. Aunque su uso no es incorrecto,se desaconseja, como es el caso de indicar el atributo `type="text/css"` en un elemento `<style>`, ya que este tipo se infiere automáticamente en HTML5.

Atributos comunes en HTML

Los atributos son palabras clave que modifican ligeramente el comportamiento de la etiqueta que los contienen.

Los **atributos comunes** son atributos que pueden utilizarse en prácticamente **cualquier etiqueta HTML**.

Atributos CSS

- **id**: establece un identificador único para la etiqueta HTML. Debe ser un nombre único en todo el documento.
- **class**: establece una clase a una etiqueta HTML. Puede repetirse a lo largo del documento.
- **style**: aplica propiedades CSS directamente al elemento HTML en cuestión.

El atributo *id* (identificador)

En HTML, podemos asignar un identificador a una etiqueta y, de este modo, 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 contener números más adelante.

- El texto debe estar en *kebab-case*, es decir, en minúsculas y separado por guiones.
- Es preferible que no contenga carácteres especiales, acentuados o emojis.
- En un documento HTML no pueden existir dos elementos con el mismo id.

Los identificadores suelen utilizarse 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 únicas. La idea de las clases es establecer géneros o tipos de etiquetas a las que les asociamos 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, el navegador no ignorará los primeros; cada clase se aplicará y se combinará en el estilo final.

El atributo *style* (estilos en línea)

El atributo `style` 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 estilos de esta forma, ya que suele considerarse una mala práctica. Es mejor colocar el código CSS en un archivo `.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 este atributo debe ser el código ISO 639-1 correspondiente al idioma que queremos especificar.

Aunque en principio podemos usar este atributo en cualquier etiqueta HTML, es **obligatorio** hacerlo en la etiqueta `<html>`, ya que esta etiqueta abarca todo el documento y establece el idioma del mismo:

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

```
  ...
</html>
```

Es completamente válido utilizar este atributo en un etiqueta específica para indicar un idioma diferente al especificado en la etiqueta `<html>`:

```
<!DOCTYPE html>
<html lang="es">
  ...
  <p lang="it">È perfettamente valido utilizzare questo attributo su un'etichetta specifica</p>
  ...
</html>
```

El atributo *translate* (traducción)

El atributo `translate` se puede utilizar en las etiquetas HTML y acepta los valores 'yes' y 'no'. Por defecto, si este atributo no se añade a una etiqueta, su valor es 'yes', lo que significa que todas las etiquetas son consideradas como «traducibles».

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

Con este atributo, podemos indicar a herramientas como *Google Translate* que no traduzcan una etiqueta que por ejemplo puede contener el título de un libro, el cual debe conservarse tal cual.

El atributo *dir* (direccionalidad)

El atributo `dir` permite al desarrollador indicar la **direccionalidad del texto** en el documento. Esto es especialmente útil para idiomas que se escriben o leen 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). Sin embargo, podemos modificarlo y establecer el valor `rtl` (*right to left*, de derecha a izquierda) o `auto` para permitir que el *user agent* detecte automáticamente la dirección de escritura.

```
<p lang="it" dir="auto">I contenuti dell'elemento sono esplicitamente isolati dal testo</p>
```

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 utiliza principalmente en las etiquetas de imágenes `<img>`, la mayoría de las etiquetas HTML pueden tener el atributo `title`, que especifica un mensaje de texto que aparece cuando el usuario detiene 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 caso de que no se pueda ver visualmente)
- El atributo `title` puede describir la imagen, pero no necesariamente debe ser una descripción alternativa.

El atributo *data-* (metadatos)

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

Los atributos de datos personalizados están destinados a almacenar datos personalizados, estados, anotaciones y similares, que son privados a la página o aplicación y para los cuales 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 el ejemplo, 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, también podemos 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>
```

> ⚠ Su **uso está desaconsejado** debido a que no está estandarizado entre sistemas operativos y navegadores.

Estructura de un documento HTML

Un documento HTML debe estar **bien formado sintácticamente** para que el navegador pueda leerlo correctamente. A continuación se presenta un ejemplo básico de la estructura de un documento HTML:

```
<!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** (_Standard Generalized Markup Language), por lo 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 declaración especial que se escribe en la primera línea del documento HTML y que indica el tipo de etiquetas que se están utilizando. Esta declaración no se considera una etiqueta HTML en sí.

Es una **declaración obligatoria**, aunque si no se incluye, la página web probablemente continuará visualizándose de manera correcta. Sin embargo, omitirla puede hacer que el navegador entre en lo que se llama **'Quirks Mode'** (modo peculiar o modo no estándar), que activa un modo de retrocompatibilidad con páginas antiguas, procesando muchas etiquetas HTML o propiedades CSS de manera diferente.

Para los **documentos HTML5**, la declaración se escribe de la siguiente manera:

```
<!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">
```

Etiqueta `<html>`

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

Aunque el uso del atributo `lang` no es obligatorio, es altamente recomendable, ya que permite indicar al navegador el idioma del contenido. Este atributo es útil para los motores de búsqueda y para navegadores con síntesis de voz, facilitando el acceso a personas con discapacidad visual.

Etiqueta `<head>`

La etiqueta `<head>` de un documento HTML se conoce comúnmente como **cabecera HTML**. Sin embargo, es importante destacar que esta cabecera no es una parte visual de la página web; más bien, es una sección del código HTML donde se incluyen etiquetas de **metadatos**. Estos metadatos establecen información sobre el documento que no necesariamente se visualiza en la página.

Algunos ejemplos de metadatos que se pueden incluir en la etiqueta `<head>` son:

- Título de la página (definido con `<title>`).

- Descripción de la página (generalmente a través de la etiqueta `<meta name="description">`).
- Icono o *favicon* (definido con `<link rel="icon">`).
- Configuración de la codificación de caracteres (definida con `<meta charset="UTF-8">`).
- Configuraciones de *viewport* para dispositivos móviles (definida con `<meta name="viewport">`).

```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Título de la Página</title>
  <link rel="icon" href="favicon.ico" type="image/x-icon">
  <meta name="description" content="Descripción breve de la página.">
</head>
```

Etiqueta `<title>`

La etiqueta `<title>` es **obligatoria** en un documento HTML y **sólo puede haber una** de este tipo por documento.

Esta etiqueta define el **título del documento**, que aparece en la barra de título de la ventana del navegador o en las pestañas. Además, el contenido de la etiqueta `<title>` se utiliza como enlace en los resultados de los motores de búsqueda, lo que la convierte en un elemento crucial para la optimización en motores de búsqueda (SEO).

Etiqueta `<meta>`

La etiqueta `<meta>` se utiliza para almacenar metadatos del documento. Se pueden incluir varios elementos `<meta>` en la sección `<head>`, y no es obligatorio que haya al menos uno.

Uno de los metadatos más importantes es el que indica el tipo de codificación del documento. Es crucial definir la codificación justo después de la etiqueta de apertura `<head>`, ya que esta afectará a todos los elementos subsiguientes.

En HTML5, la **codificación por defecto es UTF-8**.

Unicode es un estándar que asigna un número único a cada carácter, lo que permite representar texto de manera uniforme. Por ejemplo, la letra "A" se representa con el número 65. Por su parte, **UTF-8 es un estándar de codificación** que especifica cómo estos números se traducen a un formato binario que puede ser almacenado y procesado por computadoras.

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

Etiqueta `<link>`

La etiqueta `<link>` se utiliza para crear enlaces a recursos externos que el navegador procesará, como hojas de estilo y *favicons*. Esta etiqueta se coloca dentro de la sección `<head>` del documento HTML:

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

La etiqueta `<link>` también permite especificar diferentes tamaños y tipos de *favicon* mediante el atributo `rel`:

```
<head>
  <title>Forums — Inbox</title>
  <link rel=icon href=favicon.png sizes="16x16" type="image/png">
  <link rel=icon href=windows.ico sizes="32x32 48x48" type="image/vnd.microsoft.icon">
  <link rel=icon href=mac.icns sizes="128x128 512x512 8192x8192 32768x32768">
  <link rel=icon href=iphone.png sizes="57x57" type="image/png">
  <link rel=icon href=gnome.svg sizes="any" type="image/svg+xml">
  <!-- ... -->
 </head>
```

Etiqueta `<style>`

La etiqueta `<style>` se utiliza para declarar estilos CSS que se aplican exclusivamente al documento actual. Se debe incluir dentro de la sección `<head>` del documento HTML.

```
<head>
  <style>
    .autor {
      text-transform: uppercase; /* Convierte el texto a mayúsculas */
    }
  </style>
</head>
```

No es necesario especificar el atributo `type="text/css"` en la etiqueta `<style>`, ya que CSS es considerado el tipo de contenido por defecto.

Aunque se pueden definir estilos directamente en el documento HTML, se recomienda utilizar hojas de estilo externas para mantener una mejor organización y reutilización del código.

Etiqueta `<script>`

La etiqueta `<script>` se utiliza para incluir scripts de JavaScript que se aplican exclusivamente al documento actual. Esta etiqueta puede colocarse en cualquier parte del documento HTML, aunque se recomienda incluirla al final del cuerpo (`<body>`) para asegurar que el navegador procese el contenido HTML antes de ejecutar el script.

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

No es necesario especificar el atributo `type="text/javascript"` en la etiqueta `<script>`, ya que JavaScript es considerado el tipo de contenido por defecto.

Para utilizar un script desde un archivo externo, puede usarse el atributo `src`:

```
<script src="script.js"></script>
```

Etiqueta `<body>`

La etiqueta `<body>` es esencial en un documento HTML, ya que incluye todos los elementos que constituyen el contenido visible de la página.

La apertura de la etiqueta `<body>` se sitúa justo después del cierre de la etiqueta 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>
  <h1>Bienvenido a mi página web</h1>
  <p>Este es un párrafo de ejemplo.</p>
  <img src="imagen.jpg" alt="Descripción de la imagen">
  <a href="https://www.ejemplo.com">Visita nuestro sitio</a>
</body>
</html>
```

Contenedores semánticos

Los contenedores en HTML permiten **organizar y estructurar el contenido de una página** de forma lógica. Estos contenedores pueden incluir una amplia gama de elementos como texto, imágenes, formularios, etcétera…

- **Contenedores genéricos**: no tienen un significado específico en cuanto al contenido que agrupan.
  - `<div>`: se utiliza para agrupar bloques de contenido. No tiene significado semántico y se usa principalmente para aplicar estilos o scripts.
  - `<span>`: similar a `<div>`, pero se utiliza para agrupar contenido en línea, como partes de texto, dentro de un párrafo.
- **Contenedores semánticos**: proporcionan información adicional sobre el tipo de contenido que contienen, lo que mejora la accesibilidad y la comprensión tanto para los navegadores como para los motores de búsqueda.
  - `<header>`: representa un encabezado de una página o sección.
  - `<footer>`: representa el pie de una página o sección.
  - `<article>`: define un contenido independiente y autónomo, como un artículo de una revista o una entrada de blog.
  - `<section>`: agrupa contenido temáticamente relacionado.
  - `<nav>`: define un conjunto de enlaces de navegación.
  - `<aside>`: contiene información relacionada pero tangencial al contenido principal, como una barra lateral.

Elemento `<div>`

El elemento `<div>` es un contenedor genérico y uno de los elementos más antiguos en HTML. Se utiliza para agrupar otros elementos y contenido, permitiendo incluir cualquier tipo de elementos, incluidos otros `<div>`. Sin embargo, su uso no proporciona información semántica sobre el contenido.

Aunque `<div>` es útil para estructurar un documento, es preferible utilizar contenedores semánticos como `<header>`, `<footer>`, o `<section>`, que describen mejor la naturaleza del contenido.

```
<!-- 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>
```

> !! El elemento `<div>` es un elemento de **bloque**.

Elemento `<span>`

El elemento `<span>` es un contenedor genérico que se utiliza normalmente 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>
```

> !! El elemento `<span>` es un elemento **en línea**.

Elemento `<header>`

El elemento `<header>` es un contenedor semántico que se se utiliza para definir una sección introductoria o de encabezado, y suele contener elementos como títulos, logotipos o enlaces de navegación. Se puede usar en diferentes niveles:

- A nivel de **página**: para mostrar encabezados generales como un logotipo o una barra de navegación.
- A nivel de **contenido**: dentro de un artículo u otra sección para definir un encabezado específico.

```
<article>
  <header>
    <h2>Encabezado del artículo</h2>
  </header>
  <p>Cuerpo del artículo.</p>
  <footer>
    <p>Pie del artículo.</p>
  </footer>
</article>
```

Desde HTML5.1, es posible 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>Título del artículo</h2>
    <aside>
      <header>
        <h3>Título del contenido secundario</h3>
      </header>
      <p>Texto adicional dentro del artículo.</p>
      <footer>
        <p>Pie del contenido secundario.</p>
      </footer>
    </aside>
  </header>
  <p>Texto principal del artículo.</p>
</article>
```

> !! El elemento `<header>` es un elemento de **bloque**.

Elemento `<footer>`

El elemento `<footer>` es un contenedor genérico que se utiliza para definir una sección de pie de página en un documento o en una parte específica del contenido, como un artículo o una sección. Al igual que el `<header>`, puede emplearse a diferentes niveles:

- A nivel de **página**: contiene información como derechos de autor, enlaces de contacto o políticas de privacidad.
- A nivel de **contenido**: por ejemplo, al final de un artículo, se puede utilizar para mostrar información adicional relacionada con ese contenido.

```
<article>
  <header>
    <h2>Título del artículo</h2>
  </header>
  <p>Contenido del artículo.</p>
  <footer>
    <p>© 2024 Autor del artículo</p>
  </footer>
</article>
```

El elemento `<footer>` se puede utilizar independientemente del uso de un `<header>`, y es común verlo al final de la página o en la conclusión de una sección de contenido.

> !! El elemento `<footer>` es un elemento de **bloque**.

Elemento `<aside>`

El elemento `<aside>` es un contenedor semántico que se utiliza para definir contenido relacionado o complementario al contenido principal. Es común emplearlo en barras laterales, anuncios, citas destacadas o cualquier información que apoye el contenido principal sin ser parte directa de él.

```
<article>
  <h2>Artículo principal</h2>
  <p>Este es el contenido principal del artículo.</p>
  <aside>
    <h3>Nota relacionada</h3>
    <p>Esta es una información adicional o relacionada con el artículo principal.</p>
```

```
  </aside>
</article>
```

> !! El elemento `<aside>` es un elemento de **bloque**.

Elemento `<nav>`

El elemento `<nav>` es un contenedor semántico que define una sección del documento destinada a la navegación. Suele contener enlaces a diferentes secciones del mismo documento o a otros sitios web.

```
<nav>
  <ul>
    <li><a href="#inicio">Inicio</a></li>
    <li><a href="#servicios">Servicios</a></li>
    <li><a href="#contacto">Contacto</a></li>
  </ul>
</nav>
```

Es importante destacar que no todo conjunto de enlaces tiene que estar dentro de un elemento `<nav>`. Sólo aquellos que son parte de la navegación principal o estructural del sitio deberían usarlo.

> !! El elemento `<nav>` es un elemento de **bloque**.

Elemento `<main>`

El elemento `<main>` es un contenedor semántico que se utiliza para contener el **contenido principal** de un documento HTML, es decir, el contenido que es único y específico de la página. Este contenido debe ser el más relevante y significativo para la estructura del sitio.

```
<main>
  <h1>Título Principal</h1>
  <p>Este es el contenido más importante de la página.</p>
</main>
```

Este elemento debe ser único en el documento. Sólo puede haber un elemento `<main>` en el documento.

Además, no debe estar dentro de otros elementos de estructura como `<header>`, `<footer>`, `<nav>`, `<aside>` o `<article>`.

> !! El elemento `<main>` es un elemento de **bloque**.

Elemento `<section>`

El elemento `<section` es un contenedor semántico que se utiliza para **agrupar contenido relacionado** que comparte un tema o propósito común. Cada `<section>` debe contener un encabezado (generalmente un `<h1>`, `<h2>`, etc.) que identifique el tema de esa sección, así como la posibilidad de incluir un pie de página.

```
<section>
  <h2>Título de la Sección</h2>
  <p>Este es el contenido de la sección que trata sobre un tema específico.</p>
  <footer>
    <p>Información adicional o referencias.</p>
  </footer>
</section>
```

Este elemento facilita la organización del documento en partes claramente definidas y ayuda a los lectores de pantalla a entender mejor la estructura del contenido.

Es recomendable que cada `<section>` contenga su propio encabezado, para que quede claro cuál es el tema que abarca.

La utilización de varios elementos `<section>` facilita estructurar un documento en secciones distintas.

> !! El elemento `<section>` es un elemento de **bloque**.

Elemento `<article>`

El elemento `<article>` es un contenedor semántico diseñado para representar un **contenido autónomo y reutilizable**. Como indica su nombre, su uso más habitual es la creación de artículos en un blog.

```
<article>
  <header>
    <h2>Título del Artículo</h2>
    <p>Fecha de publicación: <time datetime="2024-09-24">24 de septiembre de 2024</time></p>
  </header>
  <p>Este es el contenido, que es completamente autónomo y tiene su propio sentido.</p>
  <footer>
    <p>Escrito por: Autor del artículo</p>
  </footer>
</article>
```

> !! El elemento `<article>` es un elemento de **bloque**.

Elemento `<details>`

El elemento `<details>` es un contenedor semántico que muestra **información adicional** de forma interactiva. Este elemento es útil para revelar o esconder contenido cuando el usuario lo solicita, mejorando así la experiencia del usuario al interactuar con la página.

El elemento `<details>` contiene un elemento `<summary>`, que actúa como el título o encabezado del contenido que puede expandirse. Al hacer clic en el `<summary>`, se despliega el contenido adicional, que permanece oculto hasta que el usuario decide visualizarlo.

```
<section class="progress window">
  <h1>Copying "Really Achieving Your Childhood Dreams"</h1>
  <details>
    <summary>Copying... <progress max="100" value="25"></progress> 25%</summary>
    <dl>
      <dt>Transfer rate:</dt> <dd>452KB/s</dd>
      <dt>Local filename:</dt> <dd>/home/rpausch/raycd.m4v</dd>
      <dt>Remote filename:</dt> <dd>/var/www/lectures/raycd.m4v</dd>
      <dt>Duration:</dt> <dd>01:16:27</dd>
      <dt>Color profile:</dt> <dd>SD (6-1-6)</dd>
      <dt>Dimensions:</dt> <dd>320×240</dd>
    </dl>
  </details>
</section>
```

> !! El elemento `<details>` es un elemento de **bloque**.

Contenedores de texto

Los contenedores de texto son un tipo de contenedores semánticos que se utilizan para visualizar textos.

Estos contenedores son todos de tipo **bloque**, lo que implica que utilizarán toda la longitud del contenedor padre. Por lo tanto, el contenedor siguiente se mostrará en una nueva línea.

Además, los navegadores insertar un espacio antes y después del bloque. Este comportamiento se puede modificar mediante reglas CSS.

Elementos de título `<hx>`

Los elementos de título `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`, `<h6>` permiten insertar hasta seis niveles de títulos jerárquicos en un documento.

Estos títulos tienen un fuerte significado semántico, siendo `<h1>` el más importante y `<h6>` el menos importante. Es aconsejable no saltarse ningún nivel.

Además, es perfectamente válido repetir niveles de jerarquía en contenedores distintos, es decir, es posible utilizar un título `<h1>` en varios contenedores `<article>` por ejemplo.

```
<body>
  <h1>Let's call it a draw(ing surface)</h1>
  <section>
    <h2>Diving in</h2>
  </section>
  <section>
    <h2>Simple shapes</h2>
  </section>
  <section>
    <h2>Canvas coordinates</h2>
    <section>
      <h3>Canvas coordinates diagram</h3>
    </section>
  </section>
  <section>
    <h2>Paths</h2>
  </section>
</body>
```

> !! Este elemento es un elemento de **bloque**.

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

Elemento de párrafo `<p>`

El elemento `<p>` permite insertar el texto actual en un párrafo.

```
<p>The little kitten gently seated herself on a piece of carpet.</p>

<fieldset>
  <legend>Personal information</legend>
  <p>
    <label>Name: <input name="n"></label>
    <label><input name="anon" type="checkbox"> Hide from other users</label>
  </p>
  <p><label>Address: <textarea name="a"></textarea></label></p>
</fieldset>
```

> !! Este elemento es un elemento de **bloque**.

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

Elemento de citas `<blockquote>`

El elemento `<blockquote>` permite mostrar un texto extraído de un origen externo con un formato que lo distingue del texto normal. Este elemento sirve de contenedor para otros elementos como títulos, párrafo, imagen, etcétera…

```
<blockquote>
  <p>[Jane] then said she liked [...] fish.</p>
</blockquote>
```

> !! Este elemento es un elemento de **bloque**.

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

Elemento de direcciones `<address>`

El elemento `<address>` se utiliza para mostrar direcciones de todo tipo dentro de un documento. Este elemento permite anidar otros elementos en su interior.

```
<footer>
  <address>
    For more details, contact
    <a href="mailto:js@example.com">John Smith</a>.
  </address>
  <p><small>© copyright 2038 Example Corp.</small></p>
</footer>
```

> !! Este elemento es un elemento de **bloque**.

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

Elemento de texto preformateado `<pre>`

El texto preformateado, insertado con el elemento `<pre>`, permite insertar texto que se formateará con las convenciones tipográficas usuales y no con elementos HTML.

Por ejemplo, para representar código, el elemento `<pre>` puede ser utilizado junto al elemento `<code>`:

```
<p>This is the <code>Panel</code> constructor:</p>
<pre>
  <code>
    function Panel(element, canClose, closeHandler) {
      this.element = element;
      this.canClose = canClose;
      this.closeHandler = function () { if (closeHandler) closeHandler() };
    }
  </code>
</pre>
```

> !! Este elemento es un elemento de **bloque**.

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

Elemento `<hr>`

Este elemento `<hr>` no contiene texto y solo muestra una línea horizontal que permite separar diferentes partes de un contenido.

> !! Este elemento es un elemento de **bloque**.

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

Elemento de lista no ordenada `<ul>`

Las listas no ordenadas o *"unordered list"* permite listar los datos que se mostrarán con una **viñeta** delante de cada ítem. Para ello se utiliza el elemento `<ul>` para definir la lista.

Cada ítem de la lista o *"list item"* se ubica en un elemento `<li>`. Se suele emplear para barras de navegación ya que puede considerarse que son listas de enlaces.

> ⚠ La etiqueta de cierre `</li>` puede omitirse si a continuación hay otro elemento `<li>` o no hay más contenido en el elemento padre.

```
<p>I have lived in the following countries:</p>
<ul>
  <li>Norway
  <li>Switzerland
  <li>United Kingdom
  <li>United States
</ul>
```

> !! Este elemento es un elemento de **bloque**.

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

Elemento de lista no ordenada `<menu>`

Este elemento `<menu>` es una alternativa semántica al uso de elementos `<ul>` para representar un menú o barra de herramientas donde cada elemento representa un comando o acción que el usuario puede ejecutar o activar.

```
<menu>
  <li><button onclick="copy()"><img src="copy.svg" alt="Copy"></button></li>
  <li><button onclick="cut()"><img src="cut.svg" alt="Cut"></button></li>
  <li><button onclick="paste()"><img src="paste.svg" alt="Paste"></button></li>
</menu>
```

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

Elemento de lista ordenada `<ol>`

Las listas ordenadas o *"ordered list"* permiten listar datos que se mostrarán con una **cifra** delante de cada ítem. Para ello se utiliza el elemento `<ol>` para definir la lista.

Cada ítem de la lista o *"list item"* se ubica en un elemento `<li>`. Este elemento `<li>` puede utilizar el atributo *"value"* para especificar el valor de visualización, es decir, el número o cifra que aparecerá delante del ítem en la lista.

> ⚠ La etiqueta de cierre `</li>` puede omitirse si a continuación hay otro elemento `<li>` o no hay más contenido en el elemento padre.

```
<p>I have lived in the following countries:</p>
<ol>
 <li>Switzerland
 <li>United Kingdom
 <li value="5">United States
 <li value="8">Norway
</ol>
```

El elemento `<ol>` tiene varios atributos:

- *"start"*: permite indicar el valor inicial de la numeración
- *"reversed"*: valor booleano que da la posibilidad de invertir el orden de la lista
- *"type"*: permite cambiar el tipo de la enumeración:
  - "1": para números decimales
  - "a": para letras en minúsculas
  - "A": para letras en mayúsculas
  - "i": números romanos en minúsculas
  - "I": números romanos en mayúsculas

```
<p>I have lived in the following countries</p>
<ol reversed start="5" type="a">
 <li>Switzerland
 <li>United Kingdom
 <li>United States
 <li>Norway
</ol>
```

> !! Este elemento es un elemento de **bloque**.

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

Elementos de lista de definiciones `<dl>`

Las listas de definiciones permiten mostrar las definiciones de palabras o términos. Para crear una lista de definiciones hay tres elementos disponibles:

- `<dl>` *"description list"*: permite definir la lista de definición
- `<dt>` *"description term"*: indica el término que se va a definir
- `<dd>` *"description definition"*: da la definición del término, identada respecto al término.

```
<dl>
  <dt lang="en-US"> <dfn>color</dfn> </dt>
  <dt lang="en-GB"> <dfn>colour</dfn> </dt>
  <dd> A sensation which (in humans) derives from the ability of
  the fine structure of the eye to distinguish three differently
  filtered analyses of a view. </dd>
</dl>
```

> !! Este elemento es un elemento de **bloque**.

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

Formateo del texto

Para el formateo de texto con etiquetas HTML, existen una serie de etiquetas, algunas de las cuales son semánticas y proporcionan información del contenido y otras son etiquetas no semánticas que no proporcionan ninguna información y que provienen de versiones anteriores de HTML.

Formateo semántico del texto

El formateo semántico permite resaltar palabras en un contenedor de tipo bloque con elementos de tipo en línea.

- `<strong>` : marcado semántica de especial énfasis marcando en negrita (y no de resaltado en negrita como popularmente se cree)
- `<em>` : marcado semántico de énfasis sencillo aplicando una cursiva.
- `<ins>` : marcado semántico para subrayar un texto
- `<del>` : marcado semántico para tachar un texto
- `<sub>` y `<sup>` : permite poner caracteres como índice o exponente respectivamente
- `<small>` : marcado semántico para mostrar un texto más pequeño
- `<cite>` y `<q>` : marcado para indicar el título de una obra (en cursiva) y una cita al mismo (entre comillas dobles)
- `<dfn>` : marcado semántico para una definición
- `<abbr>` : marcado semántico para una abreviatura
- `<code` >: marcado semántica para código informático
- `<mark>` : marcado semántico que destaca un texto
- `<br>` : permite cambiar de línea dentro de un párrafo permaneciendo estructuralmente en el mismo párrafo

Formateo no semántico

Etiquetas que permiten un formateo del texto pero no proporcionan ninguna información semántica. Es conveniente siempre utilizar sus alternativas semánticas para proporcionar más información al navegador y al usuario.

Por ejemplo, es recomendable utilizar la etiqueta `<strong>` antes que la etiqueta `<b>` para resaltar un texto.

- `<b>` : resaltado en negrita sin información semántica
- `<i>` : resaltado en cursiva sin información semántica
- `<u>` : resaltado con subrayado sin información semántica
- `<s>` : resaltado con tachado sin información semántica

Caracteres especiales

Los caracteres especiales como flechas o símbolos tienen la forma `&{code};` .

Por ejemplo, un caracter muy utilizado es el espacio de no separación o *Non-breaking space* que se denota como ` `. Este tipo de separación no incluye un salto de línea.

HTML Entities, HTML Symbols o Using Emojis in HTML

Los enlaces

La Web esta formada por enlaces de hipertexto, que representan una conexión entre dos recursos. En un documento HTML hay dos tipos de enlaces:

- Enlaces a **recursos externos** como hojas de estilo CSS o ficheros JavaScript que serán procesados por el *"user agent"* o navegador.

- Enlaces de **hipertexto** que son enlaces a otros recursos y que son accesibles mediante la interacción del usuario (pulsando sobre ellos, por ejemplo)

Los enlaces son un elemento de tipo **en línea**.

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

Elemento `<a>`

Los enlaces se crean con la etiquetas `<a>` aunque hay otras etiquetas como `<area>`, `<form>` y `<link>` que son enlaces especiales.

Algunas atributos aplicables a los enlaces:

- **href**: obligatorio para indicar la URL de destino
- **hreflang**: especifica el idioma de destino
- **rel**: permite indicar el tipo de relación del enlace que se establece
- **target**: da el contexto de apertura del enlace
  - _blank: el destino se abre en una nueva ventana o una nueva pestaña
  - _parent: el destino se abre en el elemento padre.
  - _self: el destino se abre en el mismo navegador por lo que no hay cambio de contexto
  - _top: el destino se abre en el padre superior en la jerarquía de ventanas

Los enlaces son **relativos** cuando apuntan a un documento del mismo sitio web mientras que un enlace es **absoluto** si apunta a otro sitio web.

```
<nav>
 <ul>
  <li><a href="/">Home</a></li> <!--Enlace relativo -->
  <li><a href="/news">News</a></li> <!--Enlace relativo -->
  <li><a href="/legal">Legal</a></li> <!--Enlace relativo -->
  <li><a href="#sectionA">Section A</a></li> <!--Enlace interno -->
  <li><a href="#sectionB">Section B</a></li> <!--Enlace interno -->
  <li><a href="http://google.es">Buscador</a></li> <!--Enlace absoluto -->
 </ul>
</nav>
<section id="sectionA">
  <h2>Section A</h2>
  <p>Lorem ipsum dolor sit amet consectetur adipisicing elit.</p>
</section>
<section id="sectionB">
  <h2>Section B</h2>
  <p>Lorem ipsum dolor sit amet consectetur adipisicing elit.</p>
</section>
```

Los enlaces **internos** al propio documento facilitan la navegación al usurario cuando el documento tiene una longitud elevada. Para implementar un enlace interno es necesario que el destino del enlace tenga el atributo `id`

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

Tablas

Las tablas se utilizan para mostrar datos tabulares. Cualquier otro uso, como por ejemplo estructurar un documento, no es correcto.

El elemento padre para contener una tabla es `<table>`.

Otros elementos son:

- `<tr>` - *"table row"*: permite insertar filas en la tabla
- `<td>` - *"table data"*: permite crear celdas en cada fila
- `<th>` - *"table header"*: permite crear celdas con encabezado
- `<caption>`: permite añadir un título a la tabla

```
<table>
  <caption>Resultados primer trimestre</caption>
  <tr>
    <th> </th>
    <th>Enero</th>
    <th>Febrero</th>
    <th>Marzo</th>
  </tr>
  <tr>
    <th>Nantes</th>
    <td>1.84M</td>
    <td>2.33M</td>
    <td>1.39M</td>
  </tr>
  <tr>
    <th>París</th>
    <td>4.24M</td>
    <td>2.29M</td>
    <td>5.17M</td>
  </tr>
</table>
```

Las etiquetas de cierre `</tr>`, `</td>`, `</th>` y `</caption>` se pueden omitir en determinados supuestos. La tabla anterior quedaría:

```
<table>
  <caption>Resultados primer trimestre
  <tr>
    <th> 
    <th>Enero
    <th>Febrero
    <th>Marzo
  <tr>
    <th>Nantes
    <td>1.84M
    <td>2.33M
    <td>1.39M
  <tr>
```

```
    <th>París
    <td>4.24M
    <td>2.29M
    <td>5.17M
</table>
```

Para fusionar celdas horizontalmente se utiliza el atributo `colspan` mientras que para fusionar celdas verticalmente se utiliza el atributo `rowspan`. En ambos casos se indica el número de celdas a fusionar.

Para estructurar la tabla se pueden usar las etiquetas `<thead>`, `<tbody>` y `<tfoot>` de forma que se puedan agrupar las filas según su significado.

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

Imágenes

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

Elemento `<img>`

Para insertar una imagen en un documento HTML se utiliza la etiqueta `<img>`.

El atributo `src` es **obligatorio** para indicar la ruta de acceso a la imagen a mostrar. Esta ruta puede ser relativa al propio sitio web o absoluta.

Con el atributo `alt` se puede mostrar un texto alternativo a la imagen si no se puede cargar.

Los atributos `width` y `height` permiten indicar el espacio asignado a la visualización de la imagen. Si no se informan, el navegador debe esperar a la carga del archivo de imagen para determinar sus dimensiones y reservar el espacio. En caso de que el tamaño de alto o ancho de la imagen difieran de los indicados en los atributos, éstos tendrán preferencia.

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

Elemento `<picture>`

El elemento `<picture>` permite mostrar diferentes imágenes para diferentes dispositivos o tamaños de pantalla.

Este elemento actua como contenedor. Dentro de este contenedor podemos utilizar etiquetas `<img>` o `<source>` para indicar diferentes imágenes a través del atributo `srcset`. El navegador utilizará el primer elemento que mejor se ajuste a la pantalla del dispositivo.

Es recomendable especifiar una etiqueta `<img>` en último lugar dentro del contenedor `<picture>` que será utilizada por los navegadores que no soporten la etiqueta `<picture>`.

```
 <picture>
  <source media="(min-width: 650px)" srcset="img_food.jpg">
  <source media="(min-width: 465px)" srcset="img_car.jpg">
  <img src="img_girl.jpg">
</picture>
```

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

Formularios

Un formulario es un componente de una página web que tiene controles de formulario, tales como texto, botones, casillas de verificación, rango o controles de selector de color.

Un usuario puede interactuar con un formulario de este tipo, proporcionando datos que luego se pueden enviar al servidor para su posterior procesamiento.

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

Atributos comunes

Los elementos HTML dedicados a los formularios comparten un gran número de atributos comunes:

- `autocomplete` : permite el autocompletado de los campos.
- `autofocus` : para activar inmediatamente el campo
- `disabled` : es un atributo booleano. Desactiva el campo y por tanto el usuario no lo puede utilizar:
  - elementos de tipo `button` , `input` , `select` , `textarea`
  - elementos contenidos en `fieldset`
- `id` : permite identificar de forma única el campo
- `name` :da nombre al campo
- `placeholder` : muestra un texto como indicación del campo
- `readonly` : indica que el campo es de solo lectura
- `required` : indica que el campo es obligatorio
- `name` : da nombre al campo. Este nombre se utilizará al recuperar los datos enviados por el formulario.

Elemento `<form>`

El elemento `<form>` es el contenedor de un formulario.

Este elemento tiene varios atributos:

- `action` : la URL del script que va a hacerse cargo de los datos introducidos en el formulario
- `method` : especifica si los datos se enviarán usando HTTP con el método 'POST' o 'GET'.
- `name` : asigna un nombre a un formulario
- `enctype` : indica el tipo MIME de los datos enviados:
  - *"application/x-www-form-urlencoded"*: formato por defecto. Los datos se codifican como pares clave-valor.
  - *"multipart/form-data"*: para el envío de archivos

```
<form action="backed.php" method="post" name="inscription"
  enctype="application/x-www-form-urlencoded">
  ...
</form>
```

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

Elemento `<fieldset>`

Para mejorar la visibilidad de los campos de un formulario se pueden agrupar mediante el elemento `<fieldset>`.

Mediante el elemento `<legend>` se puede una leyenda o título.

```
<form method="post" action="order.cgi">
  <fieldset>
    <legend>Personal data</legend>
    <p><label>Full name: <input name="fn"> <small>Format: First Last</small></label></p>
    <p><label for="age">Age: <input min="0" name="age" type="number"></label></p>
    <p><label>Post code: <input name=pc> <small>Format: AB12 3CD</small></label></p>
  </fieldset>
</form>
```

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

Elemento `<label>`

El elemento `<label>` permite asociar una etiqueta a un campo. Esta etiqueta se mostrará delante del campo y permite que se active el campo al pulsar sobre la etiqueta.

Este elemento facilita su utilización y facilita la accesibilidad a las personas con discapacidad visual.

La forma de asociar el elemento `<label>` con el campo es medidante el atributo `for` y el campo `id` del campo o simplemente poniendo el campo dentro de la etiqueta `<label>`. Ambas formas son aceptables:

```
<form method="post" action="order.cgi">
  <p><label for="fullName">Full name: <input id="fullName" name="fn"></label></p>
  <p><label>Age: <input name="age" type="number" min="0"></label></p>
  <p><label>Post code: <input name="pc"> <small>Format: AB12 3CD</small></label></p>
</form>
```

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

Elemento `<input>`

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

Elemento `<button>`

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

Elemento `<select>`

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

Elemento `<datalist>`

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

Elemento `<optgroup>`

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

Elemento `<option>`

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

Elemento `<textarea>`

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

Elemento `<output>`

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

Elemento `<progress>`

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

Elemento `<meter>`

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

Recursos multimedia embebidos

Los recursos multimedia embebidos pueden ser tanto audio como video.

Para comprimir un fichero multimedia se necesita un **codec**, que es un acrónimo de COdificador-DEcodificador como por ejemplo VP9, MO3, AAC, H.265, etcétera…

Para distribuir estos ficheros, es necesario "empaquetarlos" en un formato de transporte como por ejemplo .ogg, .mp4 o .webm.

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

Elemento `<source>`

Este elemento `<source>` se puede utilizar con elementos `<img>`, `<video>` y `<audio>`.

Este elemento permite definir diferentes fuentes de datos para un mismo elemento. Será el navegador el que elija el origen que mejor se ajuste.

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

Elemento `<video>`

El elemento `<video>` permite insertar archivos de vídeo en un documento web. El atributo `src` del elemento indica el archivo a utilizar.

Sin embargo, es recomendable utilizar la etiqueta `<source>` para indicar distintas fuentes para un mismo clip, por ejemplo indicando diferentes formatos. Además, permite indicar un texto que será mostrado si el navegador no tiene soporte para esta etiqueta:

```
<video controls poster="intro.jpg">
  <source src="clipA.mp4">
  <source src="clipA.webm">
  <p>El navegador no tiene soporte para reproducir estos clips</p>
</video>
```

El atributo booleano `autoplay` permite reproducir automáticamente el vídeo.

Para controlar la reproducción está disponible el atributo booleano `controls` cuyo aspecto varía en función del navegador.

Con el atributo `poster` se puede indicar una imagen de apertura.

Se pueden especifiar las dimensiones del vídeo con los atributos `width` y/o `height`.

Además, tenemos el atributo `loop` para reproducir el vídeo en bucle y el atributo `muted` para desactivar el sonido.

Los archivos de vídeo son archivos pesados por lo que es recomendable precargarlos para que la reproducción sea más fluida:

- `preload="auto"` : indica que es el navegador el encargado de descargar los datos necesarios
- `preload="metadata"` : especifica al navegador que es necesario descargar los metadatos del vídeo para obtener la información sobre tamaño, duración, etcétera…
- `preload="none"` : indica al navegador que no hay que precargar el vídeo

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

Elemento `<audio>`

El elemento `<audio>` permite insertar archivos de audio en un documento web. El atributo `src` del elemento indica el archivo a utilizar.

Sin embargo, es recomendable utilizar la etiqueta `<source>` para indicar distintas fuentes para un mismo clip, por ejemplo indicando diferentes formatos. Además, permite indicar un texto que será mostrado si el navegador no tiene soporte para esta etiqueta:

```
<audio controls>
  <source src="audioA.mp3">
  <source src="audioA.webm">
  <p>El navegador no soporte esta etiqueta</p>
</audio>
```

Para controlar la reproducción está disponible el atributo booleano `controls` cuyo aspecto varía en función del navegador.

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.

Referencias

- https://html.spec.whatwg.org/
- https://www.w3.org/TR/
- https://htmlreference.io/
- https://cheatsheets.zip/

Licencia