

# Teoría HTML

¡Hola! 🖐️ Te damos la bienvenida al módulo de HTML y CSS.

Te guiaremos paso a paso en el aprendizaje de los conceptos fundamentales de la programación web. Exploraremos los pilares principales de la creación de sitios web atractivos: HTML, CSS.

Comenzaremos por entender el papel crucial de HTML (Hypertext Markup Language) en la estructuración lógica y semántica del contenido de una página web.

Aprenderás a crear encabezados, párrafos, listas y enlaces, y cómo utilizar las etiquetas adecuadas para organizar la información de manera eficiente.

A continuación, nos sumergiremos en CSS (Cascading Style Sheets), el lenguaje que permite dar estilo y diseño a nuestras páginas web. Descubrirás cómo aplicar colores, fuentes, márgenes y efectos visuales para lograr un aspecto visualmente atractivo y coherente en tus proyectos web.

Estamos emocionados por todo lo que aprenderás en este curso y estamos seguros de que te sorprenderá la versatilidad y el poder que adquirirás al dominar HTML y CSS.

**¡Así que empecemos juntos esta increíble aventura en el fascinante mundo de la programación web!**

## Ambiente de trabajo

Antes de empezar a trabajar es muy importante poder establecer el ambiente de trabajo, que comprende tanto la organización del trabajo como las herramientas a utilizar.

### Editor de texto

El editor de texto es una herramienta que posibilita la creación, modificación o eliminación de texto. Muchos editores incluyen características más especializadas para la programación, razón por la cual se les conoce como **editores de código**.

Además de facilitar la edición de texto, un editor de texto también es útil al momento de programar, ya que resalta errores, subraya y ofrece la función de resaltado de sintaxis, lo que significa que asigna colores a cada tipo de elemento en el código para una lectura más sencilla.

Existen diferentes tipos de editores de texto, desde opciones básicas como el bloc de notas de tu computadora, hasta opciones más avanzadas y profesionales como Sublime Text, Vim o Visual Studio Code.

Como editor de código, seguiremos empleando 👉 [Visual Studio Code](#), que nos brinda una gran cantidad de herramientas y extensiones para facilitar nuestro trabajo. Además, te recomendamos utilizar la extensión **“Live Server”** que nos permitirá ver los cambios en tiempo real mientras desarrollamos nuestras páginas web.

Prepara tu ambiente de trabajo siguiendo esta estructura de carpetas y asegúrate de tener instalado Visual Studio Code junto con la extensión “Live Server”. De esta manera podrás comenzar a crear tus proyectos frontend de manera eficiente y efectiva.

💡 Te recordamos a continuación los pasos para instalar “Live Server”

1. Abre Visual Studio Code en tu computadora.
2. Haz clic en la pestaña "Extensions" en la barra lateral izquierda (o presiona Ctrl+Shift+X en Windows/Linux o Cmd+Shift+X en macOS) para abrir el administrador de extensiones.
3. En el campo de búsqueda, escribe "Live Server" y presiona Enter.
4. La extensión "Live Server" debería aparecer en los resultados de búsqueda. Haz clic en el botón "Install" para comenzar la instalación.
5. Una vez que se complete la instalación, verás un botón "Reload" para reiniciar Visual Studio Code. Haz clic en ese botón para activar la extensión.

Después de seguir estos pasos, la extensión "Live Server" estará instalada en Visual Studio Code y podrás usarla para iniciar un servidor local y obtener una vista previa en tiempo real de tus proyectos web. Para utilizarla, simplemente abre el archivo HTML que desees visualizar en el servidor local, haz clic derecho en el editor y selecciona la opción "Open with Live Server" para abrirlo en tu navegador predeterminado.

Recuerda que para emplear correctamente la extensión "Live Server", debes asegurarte de tener instalado un navegador web en tu computadora.

---

## Introducción a HTML

HTML (Hypertext Markup Language) es el lenguaje empleado para crear páginas web. Utilizando HTML, el contenido de una página web se puede estructurar lógica y semánticamente.

Comenzaremos tu aprendizaje con las etiquetas HTML fundamentales que se emplean para generar contenido en la web, como encabezados, párrafos, listas y enlaces. Además, adquirirás habilidades en el uso de imágenes y tablas para mejorar la funcionalidad y apariencia de las páginas web.

## HTML: Lenguaje de Marcado

**HTML** no es un lenguaje de programación; **es un lenguaje de marcado que define la estructura de tu contenido**. Basa su sintaxis en un elemento base al que llamamos marca, tag o simplemente etiqueta. A través de las etiquetas vamos definiendo los elementos del documento, como enlaces, párrafos, imágenes, etc. Así pues, un documento HTML estará constituido por texto y un **conjunto de etiquetas** para definir la función que juega cada contenido dentro de la página. Todo eso le servirá al navegador para saber cómo se tendrá que presentar el texto y otros elementos en la página.

Existen etiquetas para crear negritas, párrafos, imágenes, tablas, listas, enlaces, etc. Así pues, aprender HTML es básicamente aprenderse una serie de etiquetas, sus funciones, sus usos y saber un poco sobre cómo debe de construirse un documento básico.

Es una tarea muy sencilla de afrontar, al alcance de cualquier persona, puesto que el lenguaje es muy entendible para los seres humanos.

Por ejemplo, toma la siguiente línea de texto:

***Mi gato es muy gruñón***

Si quieres especificar que se trata de un párrafo, podrías encerrar el texto con la etiqueta de párrafo (<p>):

**<p> Mi gato es muy gruñón </p>**

---

# Anatomía de un documento HTML

Hasta ahora has visto lo básico de elementos HTML individuales, pero estos no son muy útiles por sí solos. Ahora verás cómo los elementos individuales son combinados para formar una **página HTML entera**.

Los documentos HTML van a ser archivos de texto con la extensión **.html** y tienen la siguiente anatomía:

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

  <link rel="stylesheet" href="styles.css">
</head>
<body>

</body>
</html>
```

Tienes:

- **<!DOCTYPE html>**: el tipo de documento. Es un preámbulo requerido.

Anteriormente, cuando HTML era joven (cerca de 1991/1992), los tipos de documento actuaban como vínculos a un conjunto de reglas que el código HTML de la página debía seguir para ser considerado bueno, lo que podía significar la verificación automática de errores y algunas otras cosas de utilidad. Sin embargo, hoy día es simplemente un artefacto antiguo que a nadie le importa, pero que debe ser incluido para que todo funcione correctamente. Por ahora, eso es todo lo que necesitas saber.

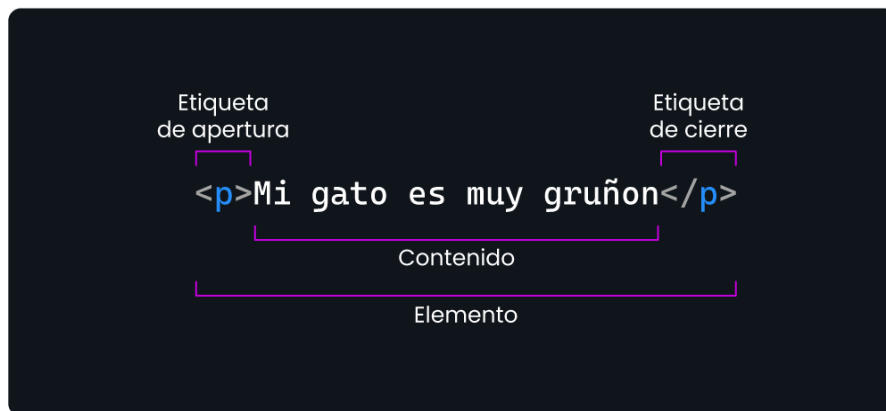
- **<html></html>**: la etiqueta <html>. Esta etiqueta encierra toda la página entera y, a veces, se la llama raíz (root element).

- **<head></head>**: la etiqueta <head>. Esta etiqueta actúa como un contenedor de todo aquello que quieres incluir en la página HTML que no es contenido visible por los visitantes de la página. Incluye cosas como palabras clave (keywords), una descripción de la página que quieres que aparezca en resultados de búsquedas, código CSS para dar estilo al contenido, declaraciones del juego de caracteres, etc.
- **<meta charset="utf-8">**: esta etiqueta establece el juego de caracteres que tu documento usará en utf-8, que incluye casi todos los caracteres de todos los idiomas humanos. Básicamente, puede manejar cualquier contenido de texto que puedas incluir. No hay razón para no incluirlo, y puede evitar problemas en el futuro.
- **<title></title>**: la etiqueta <title> establece el título de tu página, que es el título que aparece en la pestaña o en la barra de título del navegador cuando la página es cargada, y se emplea para describir la página cuando es añadida a los marcadores o como favorita.
- **<body></body>**: la etiqueta <body>. Encierra todo el contenido que desees mostrar a los usuarios web que visiten tu página, ya sea texto, imágenes, videos, juegos, pistas de audio reproducibles, y demás. Estos, delimitados a su vez por otras etiquetas como las que hemos visto.

## Elementos y etiquetas HTML

**<p> Mi gato es muy gruñón </p>** esta línea es un elemento HTML que incluye la etiqueta <p>. Veremos a continuación la anatomía de un elemento HTML y las diferentes etiquetas que existen.

## Anatomía de un elemento HTML



Las partes principales de un elemento conformado por una son:

1. **La etiqueta de apertura:** consiste en el nombre de la etiqueta (en este caso, `p`), encerrado por paréntesis angulares (`<` `>`) de apertura y cierre. Establece dónde comienza o empieza a tener efecto la etiqueta, en este caso, dónde es el comienzo del párrafo.
2. **La etiqueta de cierre:** es igual que la etiqueta de apertura, excepto que incluye una barra de cierre (`/`) antes del nombre de la etiqueta. Establece dónde termina la etiqueta, en este caso dónde termina el párrafo.
3. **El contenido:** este es el contenido de la etiqueta, que en este caso es solo texto.
4. **El elemento:** la etiqueta de apertura, más la etiqueta de cierre, más el contenido equivale al elemento.

## Anidamiento de etiquetas

Puedes también colocar etiquetas dentro de otras etiquetas. Esto se llama anidamiento. Si, por ejemplo, quieres resaltar una palabra del texto (en el ejemplo, la palabra «muy»), podemos encerrarla en una etiqueta `<strong>`, que significa que dicha palabra se debe enfatizar:

`<p>` Mi gato es `<strong>` muy `</strong>` gruñón `</p>`

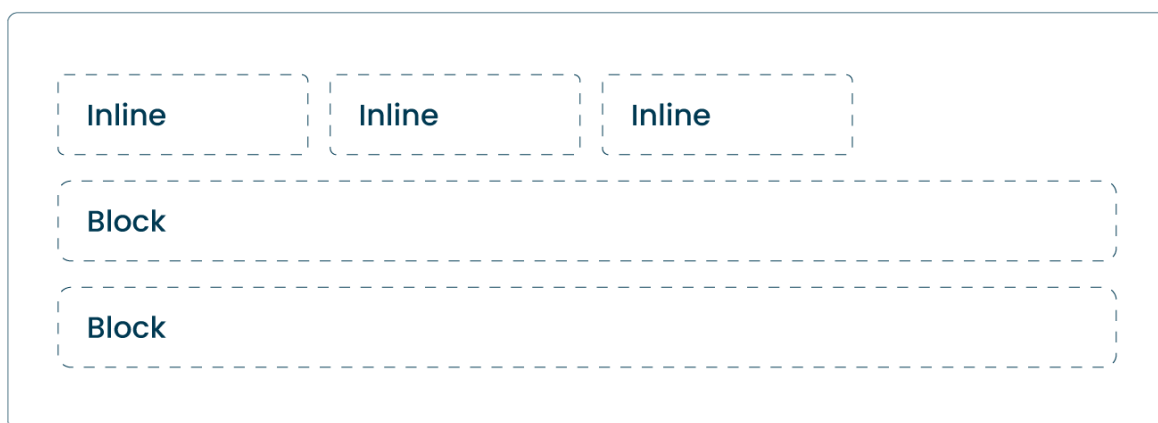
Debes asegurarte de que las etiquetas estén correctamente anidadas: en el ejemplo, creaste la etiqueta de apertura del elemento `<p>` primero, luego la del elemento `<strong>`, por lo tanto, debes cerrar esta etiqueta primero, y luego la de `<p>`.

Las etiquetas deben **abrirse y cerrarse ordenadamente**, de forma tal que se encuentren claramente dentro o fuera el uno del otro. Si estos se encuentran solapados, el navegador web tratará de adivinar lo que intentas decirle, pero puede que obtengas resultados inesperados.

## Elementos en bloque y en línea

El lenguaje HTML clasifica a todos los elementos en dos grupos: **elementos en línea o “inline elements” y elementos en bloque o “block elements”**. La diferencia entre ambos viene dada por el modelo de contenido, por el formato y la dirección.

Los **elementos en bloque** siempre empiezan en una nueva línea y ocupan todo el espacio disponible hasta el final de la línea, mientras que los **elementos en línea** solo ocupan el espacio necesario para mostrar sus contenidos sin realizar saltos de línea.



Debemos entender que los bloques jamás van a permitir que haya nada a su lado aunque haya espacio. Nunca podremos colocar nada junto a un elemento en



bloque. Por ello, en una web sin formato todos los elementos suelen ordenarse de forma vertical. Los elementos en bloque crean saltos de línea y esto es debido a que siempre ocuparán el 100% del ancho, normalmente son utilizados como contenedores. Si le dices a un bloque que ocupe un ancho determinado en píxeles (width: 200px; por ejemplo), dará lo mismo, generará un margen a su lado que impedirá que cualquier elemento se ponga junto a él.

Por otro lado, tenemos los elementos en línea. Si los elementos en bloque eran el contenedor, debemos tratar a los elementos en línea como el contenido. Cuando colocamos un elemento en línea junto a otro se colocará a su lado siempre que el ancho de pantalla lo permita, es decir, siempre que haya espacio a su lado. Podemos tratar a los elementos en línea como si fuera texto. No están pensados para ser contenedores, sino que son contenidos por ellos. Entonces, a diferencia de los elementos en bloque, un elemento en línea solo ocupa lo estrictamente necesario, no el 100% del ancho disponible.

Debemos entender las diferencias entre ambos elementos para poder crear nuestras estructuras luego con CSS. No podemos tratar por igual a un elemento en bloque y en línea, es uno de los errores más comunes cuando se comienza a maquetar. Y muchas veces, esta es la parte más difícil, diferenciarlos.

Etiquetas en línea tenemos muchas. Y etiquetas en bloque, aún más. Te dejamos un enlace con elementos en línea y en bloque [👉 elementos en bloque y en línea](#) (se puede traducir la página, pero la lista de elementos recomendamos verla en inglés).

## Atributos de las etiquetas

Las etiquetas son la estructura básica del HTML. Estas etiquetas se componen y contienen otras propiedades, como son los atributos y el contenido.

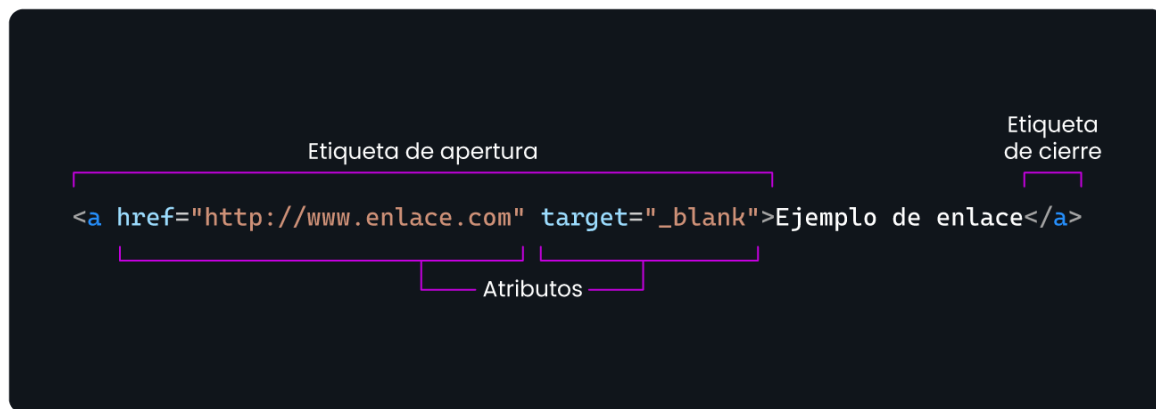
Una etiqueta por sí sola a veces no contiene la suficiente información para estar completamente definida. Para ello contamos con **los atributos**.

Estos están compuestos de un par nombre-valor que se encuentran separados por "=" y escritos en la etiqueta inicial de un elemento después del nombre del

elemento. El valor puede estar encerrado entre "comillas dobles" o 'simples'. Existen, también, algunos atributos que afectan al elemento por su presencia en la etiqueta de inicio.

Esta sería la estructura general de una línea de código en lenguaje HTML:

```
<etiqueta atributo1="valor1" atributo2="valor2">contenido</etiqueta>  
<a href="http://www.enlace.com" target="_blank">Ejemplo de enlace</a>
```



Donde:

- **<a>** es la etiqueta inicial y **</a>** la etiqueta de cierre.
- **href** y **target** son los atributos.
- **http://www.enlace.com** y **\_blank** son las variables o valores de los atributos. \*
- **Ejemplo de enlace** es el contenido.

💡 **Nota:** : las etiquetas **<img>** y **<a>** las veremos en mayor profundidad más adelante.

Los atributos definen el comportamiento, los vínculos y la funcionalidad de los elementos. Algunos atributos son globales, lo que significa que pueden aparecer dentro de la etiqueta de apertura de cualquier elemento. Otros atributos se aplican a varios elementos, pero no a todos, mientras que otros atributos son específicos del

elemento, relevantes solo para un elemento. En HTML, todos los atributos, excepto los booleanos y, hasta cierto punto, los atributos enumerados, requieren un valor.

## Tipos de atributos

Aunque cada una de las etiquetas HTML define sus propios atributos, encontramos algunos comunes a muchas o casi todas las etiquetas, que se dividen en grupos según su funcionalidad:

- **Atributos básicos**
- **Atributos booleanos**

### Atributos básicos

Los atributos básicos se utilizan en la mayoría de las etiquetas HTML y XHTML, aunque adquieren mayor sentido cuando se utilizan hojas de estilo en cascada (CSS):

Atributo	Descripción
id="texto"	Establece un indicador único a cada elemento
class="texto"	Establece la clase CSS que se aplica a los estilos del elemento
style="texto"	Aplica de forma directa los estilos CSS de un elemento
title="texto"	Establece el título del elemento (Mejora la accesibilidad)

 **Nota:** Los atributos de *id*, *class* y *style* los veremos en mayor profundidad en la parte de CSS.

### Atributos booleanos

Si un atributo booleano está presente, siempre es verdadero. Los atributos booleanos incluyen ***autofocus***, ***inert***, ***checked***, ***disabled***, ***required***, ***reversed***, ***allowfullscreen***, ***default***, ***loop***, ***autoplay***, ***controls***, ***muted***, ***readonly***, ***multiple***,

**selected.** Si uno (o más) de estos atributos está presente, el elemento está deshabilitado. Si no está presente, no lo está.

Estas tres etiquetas son equivalentes:

```
<input required>  
<input required="">  
<input required="required">
```

Si el valor del atributo es falso, omita el atributo. Si el atributo es verdadero, incluya el atributo, pero no proporcione un valor. Por ejemplo, `required="required"` no es un valor válido en HTML; pero como `required` es booleano, los valores inválidos se resuelven como verdaderos.

## Sintaxis HTML

En HTML existen ciertas reglas de sintaxis que aprenderemos a utilizar y diferenciar, para sacarle mayor provecho al lenguaje y tener más herramientas como desarrolladores.

### ***Las mayúsculas o minúsculas son indiferentes al escribir etiquetas***

En HTML las mayúsculas y minúsculas son indiferentes. Quiere decir que las etiquetas pueden ser escritas con cualquier tipo de combinación de mayúsculas y minúsculas. Resulta, sin embargo, aconsejable acostumbrarse a escribirlas en minúscula, ya que otras tecnologías que pueden convivir con nuestro HTML (XML, por ejemplo) no son tan permisivas y nunca viene mal hacernos a las buenas costumbres desde el principio, para evitar fallos triviales en un futuro.

### ¿Cómo hacer comentarios en HTML?

En un documento HTML, los comentarios se escriben entre los caracteres "`<!--`" y "`-->`". Por ejemplo:

```
<!-- Esto es un comentario en HTML -->
```

## ¿Cómo hacer un salto de línea en HTML?

Otra de las cosas importantes de conocer sobre la sintaxis básica del HTML es que los saltos de línea no importan a la hora de interpretar una página. Un salto de línea será simplemente interpretado como un separador de palabras, un espacio en blanco. Es por ello que para separar líneas necesitamos usar la etiqueta de párrafo, o la etiqueta BR que significa un salto de línea simple.

```
<p> Esto es una línea </p>  
<br>  
<p> Esto es otra línea </p>
```

💡 **Nota:** La etiqueta **<br>** no tiene su correspondiente cierre. Es un detalle que quizás te haya llamado la atención. Y es porque existen las etiquetas que se “autocierran”, en inglés, “self closing tags”.

Lista completa de etiquetas de cierre automático para HTML5

Etiqueta	Descripción
<b>&lt;area&gt;</b>	La etiqueta HTML <area> especifica un área dentro de un mapa de imagen con zonas predeterminadas en las que se puede hacer clic según las coordenadas, que posteriormente acepta una URL y se comporta como un hipervínculo. Este elemento solo se puede usar dentro de un elemento <map>.
<b>&lt;base&gt;</b>	La etiqueta HTML <base> especifica un URI base, a menudo conocido como URL base, para enlaces relativos en un documento. Un documento solo puede incluir un elemento <base>. Por ejemplo, puede especificar la URL base una vez en el área de encabezado de su página, y todos los enlaces relativos posteriores utilizarán esa URL como punto de partida.
<b>&lt;br&gt;</b>	La etiqueta HTML   se emplea para crear un salto de línea en el texto. Por lo general, se emplea en poemas o direcciones

	<p>donde se requiere la división de líneas. Es una etiqueta vacía, lo que significa que no contiene contenido y se denomina elemento vacío. Incluir la etiqueta <code>&lt;br&gt;</code> en el código HTML funciona de manera similar a presionar la tecla Intro en un procesador de textos.</p>
<b><code>&lt;col&gt;</code></b>	<p>La etiqueta HTML <code>&lt;col&gt;</code> especifica los atributos de las columnas contenidas dentro de la etiqueta <code>&lt;colgroup&gt;</code>. Esto le permite formatear o agregar una clase a una columna o grupo de columnas en lugar de cada celda individual. Se encuentra más comúnmente dentro de un elemento <code>&lt;colgroup&gt;</code>. Este elemento especifica la propiedad de estilo para cada columna.</p>
<b><code>&lt;embed&gt;</code></b>	<p>La etiqueta HTML <code>&lt;embed&gt;</code> se usa para incrustar aplicaciones externas, que suelen ser elementos multimedia como audio o video, en el lugar especificado en un documento HTML. Sirve como contenedor para complementos como animaciones flash. Esta es una nueva etiqueta en HTML 5 y solo requiere la etiqueta de inicio.</p>
<b><code>&lt;hr&gt;</code></b>	<p>La etiqueta HTML <code>&lt;hr&gt;</code> se utiliza para insertar una regla horizontal o un salto temático a nivel de párrafo en un documento HTML para dividir o separar secciones del documento. Se usa cuando el tema de su contenido HTML cambia abruptamente. Los divide dibujando una línea horizontal. La etiqueta <code>&lt;hr&gt;</code> es una etiqueta vacía que no requiere una etiqueta de cierre. Por ejemplo, un cambio de escena en una historia o un cambio de tema dentro de un segmento.</p>
<b><code>&lt;img&gt;</code></b>	<p>La etiqueta HTML <code>&lt;img&gt;</code> se utiliza para mostrar o incrustar una imagen en la página web. El elemento de imagen HTML es un elemento en línea y vacío que solo incluye atributos; las etiquetas de cierre no se usan en el elemento de imagen.</p>
<b><code>&lt;input&gt;</code></b>	<p>La etiqueta HTML <code>&lt;input&gt;</code> se usa para crear controles interactivos para formularios basados en web para aceptar datos del usuario. Según el dispositivo y el agente de usuario,</p>

	<p>se puede acceder a una amplia variedad de datos de entrada y widgets de control. El elemento se encuentra entre los más potentes y complejos de todas las etiquetas HTML debido a la gran cantidad de tipos de entrada y combinaciones de atributos. Se usa dentro del elemento <code>&lt;form&gt;</code> para declarar controles de entrada que permiten a los usuarios ingresar datos. <code>&lt;label&gt;</code> se puede utilizar para definir etiquetas para el elemento de entrada.</p>
<b><code>&lt;link&gt;</code></b>	<p>La etiqueta HTML <code>&lt;link&gt;</code> se utiliza para establecer una conexión entre un documento actual y un recurso externo. La etiqueta de enlace se usa principalmente para conectarse a hojas externas y establecer íconos del sitio (tanto íconos de estilo "favicon" como íconos para la pantalla de inicio y aplicaciones en dispositivos móviles), entre otras cosas. Este elemento puede aparecer más de una vez, pero solo aparece en la sección de encabezado. Los valores del elemento de enlace indican cómo se vincula el elemento y cómo se relaciona con la página que lo contiene.</p>
<b><code>&lt;meta&gt;</code></b>	<p>La etiqueta HTML <code>&lt;meta&gt;</code> le permite agregar metadatos: información adicional esencial sobre un documento de varias maneras. Los elementos <code>&lt;meta&gt;</code> se pueden usar para incorporar pares de nombre/valor que especifican las características del documento HTML, como la fecha de caducidad, el autor, una lista de palabras clave, el autor del documento, etc. Puede incluir más de una metaetiqueta en su documento según la información desea mantener. Aun así, en general, las etiquetas meta no afectan la apariencia física del documento. Por lo tanto, no importa si los incluye o no.</p>
<b><code>&lt;param&gt;</code></b>	<p>La etiqueta HTML <code>&lt;param&gt;</code> se usa para pasar un parámetro al objeto asociado con el elemento <code>&lt;object&gt;</code> para complementos. Podemos emplear varias etiquetas <code>&lt;param&gt;</code> dentro de un elemento <code>&lt;object&gt;</code> en cualquier orden, pero cada etiqueta debe tener un atributo de nombre y valor y debe insertarse al principio del contenido. La etiqueta de parámetro rige el comportamiento del elemento <code>&lt;object&gt;</code> al especificar un par distinto de atributos de nombre y valor,</p>

	como reproducción automática, controlador, etc.
<b>&lt;source&gt;</b>	La etiqueta HTML <source> se utiliza como elemento secundario para definir varios recursos multimedia para los elementos <audio>, <video> e <image>. Se usa ampliamente para proporcionar el mismo material multimedia en varios formatos de archivo, como mp3, mp4, etc., para permitir la compatibilidad con una amplia gama de navegadores debido a su soporte variable para formatos de archivo de imagen y multimedia. Básicamente, se utiliza para adjuntar activos multimedia como audio, video e imágenes.
<b>&lt;track&gt;</b>	La etiqueta HTML <track> se utiliza como elemento secundario de los elementos <audio> y <video> para definir pistas de texto basadas en el tiempo para un archivo multimedia. Se utiliza para incluir un subtítulo, una leyenda o cualquier otro tipo de texto que se representará cuando se muestre un archivo multimedia. Por ejemplo, le permite configurar pistas de texto cronometradas (o datos basados en el tiempo) para manejar los subtítulos automáticamente. El formato WebVTT (Web Video Text Tracks) (archivos .vtt) se utiliza para las pistas.
<b>&lt;wbr&gt;</b>	La etiqueta HTML <wbr> significa oportunidad de salto de palabra. Esta etiqueta denota un lugar dentro del texto donde el navegador puede opcionalmente dividir una línea, aunque sus reglas de división de línea no causarían una interrupción en esa ubicación. Por lo general, se usa cuando el término empleado es demasiado largo y existe el riesgo de que el navegador rompa las líneas en la ubicación incorrecta para ajustarse al contenido.

## Formato de párrafo en HTML

Previamente en nuestra guía habíamos visto la etiqueta <strong> que nos permitía darle formato a nuestro texto, más concreto ponerlo en negrita. Ahora veremos con más detalle las más ampliamente utilizadas y ejemplificaremos algunas de ellas posteriormente.



Formatear un texto pasa por tareas tan evidentes como definir los párrafos, justificarlos, introducir viñetas, numeraciones o bien poner en negrita, itálica, etc.

Hemos visto que para definir los párrafos nos servimos de la etiqueta **<p>** que introduce un salto y deja una línea en blanco antes de continuar con el resto del documento.

Podemos también usar la etiqueta **<br>**, de la cual no existe su cierre correspondiente, para realizar un simple salto de línea con lo que no dejamos una línea en blanco, sino que solo cambiamos de línea. Cabe destacar que la etiqueta **<br>**, no es la única etiqueta sin cierre.

Puedes comprobar que cambiar de línea en nuestro documento HTML sin introducir alguna de estas u otras etiquetas no implica en absoluto un cambio de línea en la página visualizada. En realidad, el navegador introducirá el texto y no cambiará de línea a no ser que esta llegue a su fin o bien lo especifiquemos con la etiqueta correspondiente.

### **¿Qué son las etiquetas de texto?**

Las etiquetas de texto nos permiten dar formato a los textos, algunas poseen relevancia semántica y otras solo se usan para cambiar el estilo. Veamos algunas:

Etiqueta	Ejemplo	Uso
<code>&lt;b&gt;</code>	<b>Bold</b>	No tiene importancia semántica, sólo le da estilo negrita al texto.
<code>&lt;strong&gt;</code>	<b>Strong</b>	Tiene importancia semántica ya que nos permite subrayar texto importante
<code>&lt;i&gt;</code>	<i>Italic</i>	No tiene importancia semántica, sólo permite cambiar el formato a Italic
<code>&lt;em&gt;</code>	<i>Énfasis</i>	Es Italic con importancia semántica, permite hacer énfasis en una parte del texto
<code>&lt;small&gt;</code>	small	No tiene importancia semántica, permite agregar texto pequeño

## ¿Cómo alinear un texto?

Los párrafos delimitados por etiquetas **<p>** pueden ser fácilmente justificados a la izquierda, centro o derecha, especificando dicha justificación en el interior de la etiqueta por medio de un atributo "align". Recordemos que los atributos no son más que un parámetro incluido en el interior de la etiqueta que ayudan a definir el funcionamiento de la etiqueta de una forma más personalizada.

Es importante tener muy en cuenta lo siguiente, que ya hemos comentado anteriormente. El HTML se usa para definir el contenido. Por lo tanto, los atributos align que vamos a conocer a continuación se estarían metiendo en un terreno que no le corresponde al HTML, porque están definiendo la forma en la que un párrafo debe de representarse, su estilo, y no el contenido. Es importante señalarlo para aprender que estas cosas se deben hacer mediante el lenguaje CSS, que sirve para

definir el estilo, la forma. Usamos este ejemplo también para reforzar el uso de los atributos de una manera más práctica.

Así, si deseamos introducir un texto alineado a la izquierda escribiríamos:

```
<p align="left">Texto alineado a la izquierda</p>
```

Para una justificación al centro:

```
<p align="center">Texto alineado al centro</p>
```

Para alinear a la derecha:

```
<p align="right">Texto alineado a la derecha</p>
```

Esto en una página se vería así:

Texto alineado a la izquierda

Texto alineado al centro

Texto alineado a la derecha

Como vemos, en cada caso el atributo align toma determinados valores que son escritos entre comillas. En algunas ocasiones necesitamos especificar algunos atributos para el correcto funcionamiento de la etiqueta. En otros casos, el propio navegador toma un valor definido por defecto. Para el caso de align, el valor por defecto es left.

## Formato de letra en HTML

Además de todo lo relativo a la organización de los párrafos, uno de los aspectos primordiales del formateo de un texto es el de la propia letra. Resulta muy común y práctico presentar texto resaltado en negrita, itálica y otros. Todo esto y mucho más

es posible por medio del HTML a partir de multitud de etiquetas entre las cuales vamos a destacar algunas.

No obstante, antes de comenzar es necesario reflexionar sobre por qué son interesantes estas etiquetas y siguen usándose, a pesar de que están entrando prácticamente en el terreno de CSS, ya que en la práctica están directamente formateando el aspecto de las fuentes. Son relevantes porque las etiquetas no se utilizan para establecer un estilo en concreto, sino como una función de ciertas palabras en un contenido.

## Negrita

Podemos escribir texto en negrita, incluyendo las etiquetas strong y su cierre. Recordemos que ya la habíamos visto previamente.

```
<p> <strong> Texto en negrita </strong> y texto normal</p>
```

Esto en una página se vería así:

**Texto en negrita y texto normal**

## Itálica

Existe dos opciones, una corta: i y su cierre (italic), y otra un poco más larga: EM y su cierre. En esta guía vamos a usar la primera manera de escribir y acordarse.

```
<p> <i> Texto en itálica </i> y texto normal</p>
```

Esto en una página se vería así:

*Texto en italica y texto normal*

## Subrayado

El HTML nos propone también para el subrayado la etiqueta: U (underlined). Sin embargo, el uso de subrayados debe realizarse con mucha precaución, ya que los

enlaces hipertexto serán (a pesar de que se indique lo contrario) subrayados con lo que podemos confundir al lector y apartarlo del verdadero interés de nuestro texto.

Además, cabe decir que la etiqueta U se ha quedado obsoleta, debido a que es algo que realmente se debe hacer del lado del CSS, al ser básicamente un estilo.

```
<p> <u> Texto subrayado </u> y texto normal</p>
```

Esto en una página se vería así:

Texto subrayado y texto normal

## Encabezados en HTML

Existen otras etiquetas para definir párrafos especiales, que funcionaran como títulos de nuestra página. Son los encabezados o *headings* en inglés. Son etiquetas que forman el texto como un título, pero el hecho de que cambien el formato no es lo que nos tiene que preocupar, sino el significado en sí de la etiqueta. Es cierto que los navegadores asignan un mayor tamaño de letra y colocan el texto en negrita, pero lo esencial es que sirven para definir la estructura del contenido de un documento HTML. Asimismo, los motores de búsqueda podrán analizar mejor el contenido de una página en función de los títulos y subtítulos.

Existen diversos tipos de encabezados, que se diferencian visualmente en el tamaño de la letra utilizada. En el ámbito de la etiqueta se encuentra la H1, para los encabezados más grandes, H2 para los de segundo nivel y, de esta forma, hasta el H6, el encabezado más pequeño.

Los encabezados se verán de esta manera en la página:

<h1>Encabezado 1</h1>

<h2>Encabezado 2</h2>

...

`<h6>Encabezado 6</h6>`

Los encabezados implican también una separación en párrafos, así que todo lo que escribamos dentro de H1 y su cierre (o cualquier otro encabezado) se colocará en un párrafo independiente.

Podemos ver cómo se presentan algunos encabezados a continuación.

```
<h1> Encabezado de nivel 1 </h1>
```

Los encabezados, como otras etiquetas de HTML, soportan el atributo align. Veremos un ejemplo de encabezado de nivel 2 alineado al centro.

```
<h2 align="center"> Encabezado de nivel 2 </h2>
```

## Listas en HTML

Las posibilidades que nos ofrece el HTML en cuanto al tratamiento de texto son realmente notables. No se limitan a lo observado hasta el momento, sino que se encuentran aún más allá. En numerosos casos, se encuentran las listas, que permiten enumerar y definir elementos.

**Las listas** originalmente están pensadas para citar, enumerar y definir cosas a través de características. Las listas se utilizan finalmente para mucho más que enumerar una serie de puntos, en realidad son un recurso muy útil para poder manipular elementos diversos, como barras de navegación, pestañas, etc.

En este momento, examinaremos las listas desde el punto de vista de su estructura y examinaremos los diversos tipos que se presentan, y que podemos emplear para satisfacer nuestras diversas necesidades al redactar textos en formato HTML.

Podemos distinguir dos tipos de listas HTML:

- Listas desordenadas
- Listas ordenadas

## Listas desordenadas

Son delimitadas por las etiquetas `<ul>` y su cierre `</ul>` (unordered list). Cada uno de los elementos de la lista es citado por medio de una etiqueta `<li></li>` (list item).

💡 **Nota:** La etiqueta `<li></li>` tiene su respectiva etiqueta de cierre, aunque si no lo colocas, el navegador al ver el siguiente `<li>` interpretará que estás cerrando el anterior.

```
<p>Países del mundo</p>
<ul>
  <li>Argentina</li>
  <li>Perú</li>
  <li>Chile</li>
</ul>
```

Esto renderizado (desplegado en la web) se vería así:

Países del mundo

- Argentina
- Perú
- Chile

## Tipos de viñetas

Podemos definir el tipo de viñeta empleada para cada elemento. Para ello debemos especificarlo por medio del atributo **type**. Si queremos que el estilo sea válido para toda la lista lo incluiremos dentro de la etiqueta de apertura `<ul>`, mientras que si queremos hacerlo específico a un solo elemento lo incluiremos dentro de la etiqueta `<li>`.

La sintaxis es del siguiente tipo:

```
<ul type="tipo de viñeta">
```

Donde tipo de viñeta puede ser uno de los siguientes:

- circle
- disc
- square

Por ejemplo:

```
<ul type="square">  
  <li>Elemento 1 </li>  
  <li>Elemento 2 </li>  
  <li>Elemento 3 </li>  
  <li type="circle">Elemento 4 </li>  
</ul>
```

Esto en una página web se vería así:



- Elemento 1
- Elemento 2
- Elemento 3
- Elemento 4

En este ejemplo nos encontramos con una lista desordenada donde uno de los elementos tiene una viñeta circular en lugar de un cuadrado como se especifica para el resto de los elementos en la etiqueta `<ul>`.

## Listas ordenadas

Las listas ordenadas sirven también para presentar información en diversos elementos, con la particularidad que estos estarán precedidos de un número o una letra para enumerarlos manteniendo un orden.

Para implementar listas ordenadas usaremos la etiqueta `<ol>` (ordered list) y su cierre. Cada elemento será igualmente indicado por la etiqueta `<li>` (list item), que ya vimos en las listas desordenadas.

```
<p> Reglas de convivencia</p>
<ol>
  <li> Respeta a tu compañero y escucha su opinión </li>
  <li> Trata amablemente a tu entorno </li>
</ol>
```

Esto en una página se vería así:

## Reglas de convivencia

1. Respeta a tu compañero y escucha su opinión
2. Trata amablemente a tu entorno

### Tipos de numeración

De la misma manera que para las listas desordenadas, las listas ordenadas brindan la posibilidad de modificar el estilo. En concreto, podemos especificar el tipo de numeración utilizado seleccionando entre números (1, 2, 3,...), letras (a, b, c, ...) y sus mayúsculas (A, B, C, ...) y números romanos en sus versiones mayúsculas (I, II, III, ...) y minúsculas (i, ii, iii,

Para realizar dicha selección hemos de utilizar, como para el caso precedente, el atributo `type`, el cual será situado dentro de la etiqueta `<ol>`.

Los valores que puede tomar el atributo en este caso son:

- 1** → para ordenar por números
- a** → por letras del alfabeto
- A** → por letras mayúsculas del alfabeto
- i** → ordenación por números romanos en minúsculas
- I** → ordenación por números romanos en mayúsculas

Puede que en algún caso deseemos comenzar nuestra enumeración por un número o letra que no tiene por qué ser necesariamente el primero de todos. Para solventar esta situación, podemos usar un segundo atributo, **start**, que tendrá como valor un número. Este número, que por defecto es 1, corresponde al valor a partir del cual

comenzamos a definir nuestra lista. Para el caso de las letras o los números romanos, el navegador se encarga de hacer la traducción del número a la letra correspondiente.

```
<p> Ordenamos por números </p>
<ol type="1">
  <li> Elemento 1 </li>
  <li> Elemento 2 </li>
</ol>
<p> Ordenamos por letras </p>
<ol type="a">
  <li> Elemento a </li>
  <li> Elemento b </li>
</ol>
<p>Ordenamos por números romanos empezando por el 10</p>
<ol type="i" start="10">
  <li> Elemento x </li>
  <li> Elemento xi </li>
</ol>
```

Esto en una página se vería así:

Ordenamos por números

1. Elemento 1

2. Elemento 2

Ordenamos por letras

a. Elemento a

b. Elemento b

Ordenamos por números romanos empezando por el 10

X. Elemento x

XI. Elemento xi

## Anidando listas

Nada nos impide utilizar todas estas etiquetas de forma anidada como hemos visto en otros casos. De esta forma, podemos conseguir listas mixtas.

```
<p>Ciudades del mundo</p>
<ul>
  <li>Argentina </li>
  <ol>
    <li>Buenos Aires </li>
    <li>Bariloche </li>
  </ol>
  <li>Uruguay </li>
  <ol>
    <li>Montevideo </li>
```

```
<li>Punta del Este </li>
</ol>
</ul>
```

Esto en una página se vería así:

## Ciudades del mundo

- Argentina
  1. Buenos Aires
  2. Bariloche
- Uruguay
  1. Montevideo
  2. Punta del Este

## Enlaces en HTML

Hasta aquí, hemos podido ver que una página web es un archivo HTML en el que podemos incluir, entre otras cosas, textos formateados a nuestro gusto e imágenes (las veremos con detalle enseguida). Del mismo modo, un sitio web podrá ser considerado como el conjunto de archivos, principalmente páginas HTML e imágenes, que constituyen el contenido al que el navegante tiene acceso.

Sin embargo, no podríamos hablar de navegación si estos archivos HTML no estuviesen debidamente conectados entre ellos y con el exterior de nuestro sitio por medio de **enlaces hipertexto**. En efecto, el atractivo original del HTML radica en la posible puesta en relación de los contenidos de los archivos introduciendo

referencias bajo forma de enlaces que permitan un acceso rápido a la información deseada. De poco serviría en la red tener páginas aisladas a las que la gente no puede acceder y desde las que la gente no puede saltar a otras.

Un enlace puede ser fácilmente detectado por el usuario en una página. Basta con deslizar el puntero del ratón sobre las imágenes o el texto y ver cómo cambia su forma original transformándose por regla general en una mano con un dedo señalador.

Adicionalmente, estos enlaces suelen ir, en el caso de los textos, coloreados y subrayados para que el usuario no tenga dificultad en reconocerlos.

## Sintaxis de un enlace

Para colocar un enlace, nos serviremos de las etiquetas `<a>` y su cierre `</a>`. Dentro de la etiqueta de apertura deberemos especificar asimismo el destino del enlace. Este destino será introducido bajo forma de atributo, el cual lleva por nombre **"href"**.

La sintaxis general de un enlace es, por tanto:

```
<a href="destino">contenido</a>
```

Siendo el "contenido" un texto o una imagen. Es la parte de la página que se colocará activa y donde deberemos pulsar para acceder al enlace. Por su parte, "destino" será una página.

Veamos un ejemplo con un enlace al home de Egg Live:

```
<a href="https://egg.live/es-ar/">Home de Egg Live</a>
```

Renderizado se vería de la siguiente manera:

[Home de EggEducación](https://egg.live/es-ar/)

## Enlaces con imágenes

Ahora, si queremos que el contenido del enlace sea una imagen y no un texto, deberemos colocar la etiqueta **<img>** dentro de la etiqueta **<a>**.

```
<a href="https://egg.live/es-ar/"></a>
```

💡 **Nota:** veremos la etiqueta de imágenes más adelante.

## El aspecto de los enlaces

Utilizando HTML, y las hojas de estilo CSS, podremos definir el aspecto que tendrán los enlaces de una página. Sin embargo, de manera predeterminada el navegador los destaca para que los podamos distinguir. Generalmente, encontraremos a los enlaces subrayados y coloreados en azul, aunque esta regla depende del navegador del usuario y de sus estilos definidos como predeterminados.

## Tipos de enlaces

Para estudiar en profundidad los enlaces tenemos que clasificarlos por su tipo, ya que, dependiendo del mismo, algunas características pueden cambiar.

En función del destino, los enlaces son clásicamente agrupados del siguiente modo:

- **Enlaces locales:** los que se dirigen a otras páginas del mismo sitio web.
- **Enlaces remotos:** los dirigidos hacia páginas de otros sitios web. Estos son los que vimos en el ejemplo anterior.

## Enlaces locales

Como hemos dicho, un sitio web está constituido de páginas interconectadas, que se relacionan mediante enlaces de hipertexto. Para lograr esto, vamos a utilizar los enlaces locales.

Los enlaces locales son un tipo de enlace mucho más habitual en el día a día del desarrollo. De hecho, es el tipo de enlace que más se produce en general. Los enlaces locales nos permiten establecer conexiones entre diversos documentos HTML que conforman un sitio web. Podemos usar enlaces locales para crear un sitio web completo con varios documentos.

Para crear este tipo de enlaces, hemos de usar la etiqueta `<a>` que ya conocemos de la siguiente forma:

```
<a href="archivo.html">contenido</a>
```

## Rutas de los enlaces

Como rutas, nos referimos al destino del enlace, o sea lo que hemos puesto en el atributo "href".

Por lo general, para una mejor organización, los sitios web suelen estar ordenados por directorios. Estos directorios suelen contener diferentes secciones de la página como imágenes, scripts, estilos, etc. Por consiguiente, en numerosos casos, no será factible definir el nombre del archivo, sino que será necesario también definir el directorio en el que nuestro archivo.html se encuentra alojado.

Cómo mostrar la ruta de un archivo:

1. Hay que situarse **en el directorio** en el que se encuentra la página donde vamos a crear el enlace.
2. Si la página destino está en **el mismo directorio** que el archivo desde donde vamos a enlazar, podemos colocar simplemente **el nombre del archivo de destino**, ya que no hay necesidad de cambiar de directorio.
3. Si la página de destino está en una carpeta o subdirectorio **interior al directorio** donde está el archivo de origen, hemos de marcar la ruta

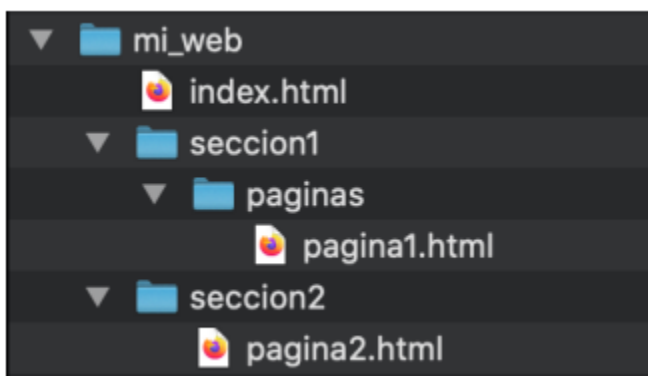


**enumerando cada uno de los directorios** por los que pasamos hasta llegar al archivo de destino. Separándolos por el símbolo barra **"/"**. Al final, escribimos el **nombre del archivo destino**.

4. Si la página destino se encuentra en un **directorio padre** (superior al de la página del enlace), hemos de escribir dos puntos y una barra **"../"** tantas veces como niveles subamos en la arborescencia hasta dar con el directorio donde está emplazado el archivo destino. Finalmente, también agregaremos el **nombre del archivo destino**.
5. Si la página se encuentra en otro directorio no incluido ni incluyente del archivo origen, tendremos que subir como en el punto 3 por medio de **".."** hasta encontrar un directorio que englobe el directorio que contiene a la página destino. A continuación, haremos como en la regla 2. Escribiremos todos los directorios por los que pasamos hasta llegar al archivo.

💡 **Nota:** puede parecer complicado en la lectura. Pero te aseguramos que cuando empieces a practicarlo vas a comprenderlo mejor. Recuerda que siempre que tengas alguna duda puedes volver a este material y releer la información.

Imagina que tienes la siguiente estructura de carpetas y archivos:



1. Para hacer un enlace a **index.html** desde **index.html**

```
<a href="index.html">Ir a index.html</a>
```

2. Para hacer un enlace desde **index.html** hacia **pagina1.html**:

```
<a href="seccion1/paginas/pagina1.html">Ir a pagina1.html</a>
```

3. Para hacer un enlace desde **pagina2.html** hacia **pagina1.html**:

```
<a href="../../seccion1/paginas/pagina1.html">Ir (también) a pagina1.html</a>
```

4. Para hacer un enlace desde **pagina1.html** hacia **pagina2.html**:

```
<a href="../../seccion2/pagina2.html">Ir ahora a pagina2.html</a>
```

## Imágenes en HTML

Insertar imágenes en nuestros textos puede ayudar a explicar más fácilmente los objetivos y organizar la información al mismo tiempo.

Cuando una imagen agrega contexto a un documento, tiene el carácter de contenido (esqueleto) y debe estar incrustado con HTML.

Las imágenes decorativas, como las imágenes de fondo de las secciones de contenido o de la página completa, son de presentación y se deben aplicar con CSS.

El método principal para incluir imágenes es la etiqueta **<img>** con el atributo **src (source)** que hace referencia al origen de la imagen y el atributo **alt** que describe la imagen.

La sintaxis queda entonces de la siguiente forma:

```

```

En el código anterior estamos enlazando un archivo con extensión **.jpg** que se encuentra en nuestra computadora, aunque también podría ser de otro tipo como **.gif** o **.png**.

## Atributo alt

Dentro de las comillas de este atributo colocaremos una breve descripción de la imagen. Esta etiqueta no es indispensable, pero presenta varias utilidades. La sintaxis quedaría de esta manera:

```

```

En primer lugar, puede ser utilizado para el posicionamiento de la página en buscadores. El buscador puede extraer las palabras clave determinadas en el atributo ALT y ayudar a comprender la función o contenido de la imagen, lo que le permite comprender la función o contenido de la página en sí mismo. Esta es una de las maneras en las que se trabaja el correcto posicionamiento web en buscadores (SEO, Search Engine Optimization).

Además, cuando la página no ha sido todavía cargada, el navegador podría mostrar esta descripción, lo que permitirá al usuario comprender lo que está sucediendo en ese sitio.

## Tablas en HTML

Una tabla es un conjunto de celdas organizadas dentro de las cuales podemos alojar distintos contenidos. HTML dispone de una gran variedad de etiquetas para crear tablas con sus atributos.

Si quieres mostrar datos de forma sencilla de leer, dispuestos en filas y columnas, tarde o temprano veremos que las tablas un gran recurso.

### Etiquetas básicas para tablas en HTML

Las tablas son definidas por las etiquetas **<table>** y su cierre **</table>**.

Dentro de estas dos etiquetas colocaremos todas las otras etiquetas de las tablas, hasta llegar a las celdas. En las celdas ya se pueden colocar textos e imágenes que proporcionarán el contenido de la tabla.

Las tablas son descritas por líneas de arriba a abajo (y luego por filas de izquierda a derecha). Cada una de estas líneas, llamadas columnas, es definida por otra etiqueta y su cierre: **<tr></tr>** (table row: columna de tabla).

Asimismo, dentro de cada línea, habrá diferentes celdas. Cada una de estas celdas será definida por otra etiqueta: **<td></td>** (table data). Dentro de ésta y su cierre será donde coloquemos nuestro contenido, el contenido de cada celda.

```
<table>
<tr>
  <td> Celda 1, línea 1 </td>
  <td> Celda 2, línea 1 </td>
</tr>
<tr>
  <td> Celda 1, línea 2 </td>
  <td> Celda 2, línea 2 </td>
</tr>
</table>
```

Esto en una página se vería así:

Celda 1, línea 1  
Celda 1, línea 2

Celda 2, línea 1  
Celda 2, línea 2

También existe la etiqueta **<th></th>** (table header), que sirve para crear una celda cuyo contenido esté formateado como un título o cabecera de la tabla. En la práctica, lo que hace es poner en negrita y centrado el contenido de esa celda.

```
<table>
<tr>
  <th>Titulo Celda 1</th>
```

```

<th> Titulo Celda 2</th>
</tr>
<tr>
  <td>Celda 1, línea 1</td>
  <td> Celda 2, línea 1</td>
</tr>
<tr>
  <td> Celda 1, línea 2</td>
  <td> Celda 2, línea 2</td>
</tr>
</table>

```

Esto en una página se vería así:

	Titulo Celda 1		Titulo Celda 2
Celda 1, línea 1		Celda 2, línea 1	
Celda 1, línea 2		Celda 2, línea 2	

## Formularios en HTML

Hasta ahora hemos visto la forma en la que el HTML gestiona y muestra la información, esencialmente mediante texto, imágenes y enlaces. Nos queda por ver de qué forma podemos intercambiar información con nuestro visitante. Desde luego, este nuevo aspecto resulta primordial para gran cantidad de acciones que se pueden llevar a cabo mediante la Web: comprar un artículo, rellenar una encuesta, enviar un comentario al autor, registrar un usuario, etc.

**Los formularios son esas famosas cajas de texto y botones que podemos encontrar en muchas páginas web.** Son muy utilizados para realizar búsquedas o bien para introducir datos personales por ejemplo en sitios de comercio electrónico. Los datos que el usuario introduce en estos campos son enviados al correo electrónico del administrador del formulario o bien a un programa que se encarga de procesarlo automáticamente.

## ¿Qué se puede hacer con un formulario?

Usando HTML podemos únicamente enviar el contenido del formulario a un correo electrónico, es decir, construir un formulario con diversos campos y, a la hora pulsar el botón de enviar, generar un mensaje de que se ha registrado con éxito la información.

Pero para todo lo que sea manejar esa información y guardarla, por ejemplo, en una base de datos vamos a tener que utilizar un lenguaje de backend, como por ejemplo Java.

Así pues, en resumen, con HTML podremos construir los formularios, con diversos tipos de campos, como cajas de texto, botones de radio, cajas de selección, menús desplegables, etc. Sin embargo, debe quedar claro que desde HTML no se puede manejar esta información para guardarla o enviarla a algún correo, etc. Eso será trabajo de Java.

## ¿Cómo hacer un formulario en HTML?

Los formularios son definidos por medio de las etiquetas **<form>** y su cierre **</form>**. Entre estas dos etiquetas colocaremos todos los campos y botones que componen el formulario. Dentro de esta etiqueta **<form> </form>** debemos especificar algunos atributos:


- **action:** define el tipo de acción a llevar a cabo con el formulario. Como ya hemos dicho, existen dos posibilidades:
  - El formulario es enviado a una dirección de correo electrónico. Para esto hay que poner el mail en el action.
  - El formulario es enviado a un programa o script que procesa su contenido.

```
<form action="ruta del método que va a manejar la información"></form>
```

- **method:** Este atributo se encarga de especificar la forma en la que el formulario es enviado. Los dos valores posibles que puede tomar este atributo

son **POST** y **GET**. A efectos prácticos y, salvo que se diga lo contrario, daremos siempre el valor **POST**.

- **enctype**: Se utiliza para indicar la forma en la que viajará la información que se mande por el formulario. En el caso más corriente, enviar el formulario por correo electrónico, el valor de este atributo debe de ser "**text/plain**". Así conseguimos que se envíe el contenido del formulario como texto plano dentro del email. Si fuéramos a enviar una imagen dentro del formulario, este atributo debería ser "**multipart/form-data**". También todos estos conceptos vamos a detallarlos más adelante.

 **Nota:** Este último atributo no es indispensable. Sin embargo, si quisiéramos guardar la información de nuestro formulario en Java, requeriremos indispensablemente de los primeros dos atributos. Estos conceptos los abordaremos y profundizaremos en otro módulo.

Entonces con todo lo anterior ya explicado, la etiqueta <form> con los atributos nos quedaría así:

```
<form action="ruta del método que va a manejar la información"
method="POST" enctype="multipart/form-data">
<!-- contenido del formulario -->
</form>
```

Entre esta etiqueta y su cierre colocaremos el resto de las etiquetas que darán forma a nuestro formulario.

## Campos de texto

El lenguaje HTML nos propone una gran diversidad de alternativas a la hora de crear nuestros formularios, es decir, una gran variedad de elementos para diferentes propósitos. Estas van desde la clásica caja de texto, hasta la lista de opciones en un menú desplegable, pasando por las cajas de validación, etc.

Las etiquetas que tenemos que utilizar para crear campos de texto, pueden ser de dos tipos. Veamos en qué consiste cada una de estas modalidades y cómo podemos implementarlas en nuestro formulario.

### Etiqueta input

Las cajas de texto son colocadas por medio de la etiqueta **<input>**. Dentro de esta etiqueta hemos de especificar el valor de dos atributos: **type** y **name**.

```
<input type="text" name="nombre">
```

Al colocarlo dentro de la etiqueta **<form>** se verá así:

De este modo expresamos nuestro deseo de crear una caja de texto cuyo contenido será llamado "nombre" (por ejemplo, en el caso de la etiqueta anterior, pero podemos poner distintos nombres a cada uno de los campos de texto que habrá en los formularios).

### **Atributos comunes entre componentes de formularios**

Todos los inputs que utilizaremos en los formularios permiten la utilización de diferentes atributos. Algunos de ellos son propios del tipo de input específico, pero también tenemos algunos atributos que son comunes a todos los inputs que insertemos en nuestros formularios, los más utilizados son:

- **type**
- **name**



- **value**

### Atributo type

Como hemos visto el atributo type nos sirve para especificar el tipo de dato que se va a ingresar en nuestro input, en el ejemplo anterior lo habíamos puesto como tipo text, para que sea una caja de texto y poder ingresar texto. Pero existen otros tipos de valores para el atributo type.

- **Text**
- **Number**
- **Date**
- **Email**
- **Texto oculto**
- **Entre otros...**

💡 **Nota:** Pueden ver una lista completa de atributos en el siguiente link de W3Schools [!\[\]\(c3d993ca47bfe2a953c700506ce31fa0\_img.jpg\) type values](#)

Valor	Descripción
<b>Text</b>	Este tipo permite al usuario ingresar una línea e texto.

<b>Number</b>	<p>Este tipo permite al usuario ingresar números. Los navegadores vienen con validaciones para evitar que el usuario ingrese algo que no sea números.</p> <p>Además, en los navegadores modernos, los campos numéricos suelen venir con controles que permiten a los usuarios cambiar su valor de forma gráfica.</p>
<b>Date</b>	<p>Este le permite al usuario ingresar una fecha, ya sea mediante una caja de texto o una interfaz gráfica con selector de fecha.</p>
<b>Email</b>	<p>Este tipo permite al usuario ingresar un mail. Los navegadores vienen con validaciones para validar que se esté ingresando con el formato correcto de un mail. Este input se va a ver como un input de texto común y corriente.</p>
<b>Password</b>	<p>Hay determinados casos en los que podemos desear esconder el texto escrito en el campo input, por medio de círculos negros, de manera que aporte una cierta confidencialidad. Para esto vamos a usar el type password.</p>

**Veamos un ejemplo de cómo se verían estos atributos en la web.**

```
<form action="ruta del método que va a manejar la información"
method="POST" enctype="multipart/form-data">
<!-- contenido del formulario -->
  <label>Tipo Texto
    <input type="text" name="name">
  </label>
  <label>Tipo Number
    <input type="number" name="name">
  </label>
  <label>Tipo Date
    <input type="date" name="name">
  </label>
  <label>Tipo Email
    <input type="email" name="name">
  </label>
  <label>Tipo Password
    <input type="password" name="name">
  </label>
</form>
```

Tipo Texto	<input type="text" value="Texto"/>
Tipo Number	<input type="number" value="123"/>
Tipo Date	<input type="date" value="06/06/2023"/>
Tipo Email	<input type="email" value="prueba@gmail.com"/>
Tipo Password	<input type="password" value="....."/>

### **Atributo name**

El nombre del elemento del formulario es de gran importancia para poder identificarlo en nuestro programa de procesamiento.

Si **name** se omite, el valor del campo de entrada no se enviará en absoluto.

### Atributos prescindibles

Además de estos dos atributos, esenciales para el correcto funcionamiento de nuestra etiqueta, existen otra serie de atributos que pueden resultarnos de utilidad pero que no son imprescindibles:

- **size**: define el tamaño de la caja de texto, en número de caracteres visibles. Si al escribir el usuario llega al final de la caja, el texto que escriba a continuación también cabrá dentro del campo, pero irá corriéndose a medida que se escribe, haciendo desaparecer la parte de texto que queda a la izquierda.
- **maxlength**: indica el tamaño máximo del texto, en número de caracteres, que puede ser escrito en el campo. En caso de que el campo de texto tenga definido el atributo maxlength, el navegador no permitirá escribir más caracteres en ese campo que los que hayamos indicado.
- **value**: en algunos casos puede resultarnos interesante asignar un valor predefinido al campo en cuestión. Esto puede ayudar al usuario a rellenar más rápidamente el formulario o darle alguna idea sobre la naturaleza de datos que se requieren. Este valor inicial del campo puede ser expresado mediante el atributo value.

```
<input type="text" name="instituto" value="Egg Educación">
```

Con la línea de especificada anteriormente se genera un campo de texto como el siguiente:

- **placeholder**: este atributo especifica una pequeña pista que describe el valor esperado para el campo (input).

La pequeña sugerencia se muestra en el campo de entrada antes de que el usuario ingrese un valor. Una vez que escriba, ese valor va a desaparecer.

```
<input type="text" name="nombre" placeholder="Nombre del usuario">
```

Nombre del usuario

💡 **Nota:** recordemos que todos estos ejemplos de input deben ir entre las etiquetas de apertura y de cierre form.

### Etiqueta textarea

Si deseamos poner a disposición del usuario un campo de texto donde pueda escribir cómodamente sobre un espacio compuesto de varias líneas, hemos de invocar una nueva etiqueta: **<textarea>** y su cierre correspondiente **</textarea>**.

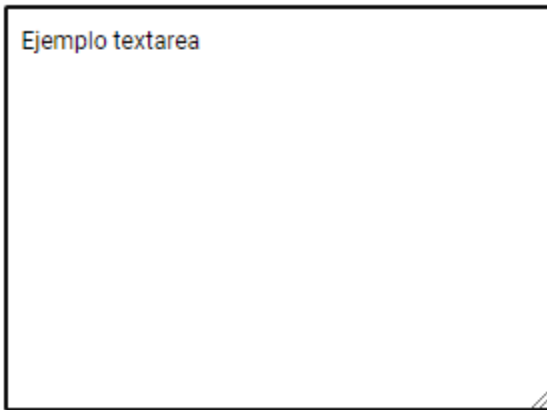
Este tipo de campos son prácticos cuando el contenido a enviar no es un nombre, teléfono, edad o cualquier otro dato breve, sino más bien, un comentario, opinión, etc. En los que existe la posibilidad de que el usuario desee rellenar varias líneas.

Dentro de la etiqueta **textarea** deberemos indicar, como para el caso visto anteriormente, el atributo name para asociar el contenido a un nombre que será asemejado a una variable en un lenguaje de programación. Además, podemos definir las dimensiones del campo a partir de los atributos siguientes:

- **rows:** define el número de líneas del campo de texto.
- **cols:** define el número de columnas del campo de texto.

La etiqueta queda por tanto de esta forma:

```
<textarea name="comentario" rows="10" cols="40"></textarea>
```



Asimismo, es posible predefinir el contenido del campo. Para ello, no usaremos el atributo value, sino que escribiremos dentro de la etiqueta el contenido que deseamos atribuirle.

```
<textarea name="comentario" rows="10" cols="40">Escribe tu comentario  
...</textarea>
```

### Etiqueta label

El elemento **<label>** y su etiqueta de cierre **</label>**, provee una descripción corta que acompaña al campo de texto. Su función es indicarle al usuario qué información debería ingresar en el campo o input.

También podemos asociar una etiqueta label a un campo para que el usuario pueda acceder al campo de texto con solo clicar el label.

La etiqueta podría verse de esta forma:

```
<label>Nombre del Usuario</label>  
<input type="text" name="nombre">
```

Esto en una página se vería así:

Nombre del Usuario

## Atributo for

La etiqueta label solo consta del atributo **for**. Mediante la utilización del atributo for podemos asociar el label con el input. Para lograr esto vamos a tener que utilizar también el atributo id, este atributo lo explicamos previamente y lo vamos a ver más en detalle en la parte de CSS.

La manera que asociamos un label a un input es la siguiente:

- En la etiqueta label agregaremos un **atributo for** con un valor, este va a representar el dato que se va a ingresar en el input.
- En el input vamos a poner el mismo valor, pero en el **atributo ID**.

```
<label for="nombre">Nombre del Usuario</label>  
<input type="text" id="nombre" name="nombre">
```

El label y el input se verán igual pero ahora cuando el usuario clickee el label, se va a activar el campo de texto del input para poder ingresar el valor que el usuario necesite. Después vamos a ver un ejemplo más útil con las cajas de opciones.

## Otros elementos de formularios

Seguramente hayan notado que los inputs son una manera muy práctica de hacernos llegar la información del navegante. No obstante, en muchos casos, permitir al usuario que escriba cualquier texto permite demasiada libertad y puede que la información que éste escriba no sea la que nosotros estamos necesitando.

Por ejemplo, pensemos que queremos que el usuario indique su país de residencia. En ese caso podríamos ofrecer una lista de países para que seleccione el que sea. Este mismo caso se puede aplicar a gran variedad de informaciones, como el tipo de tarjeta de crédito para un pago, la puntuación que da a un recurso, si quiere recibir o no un boletín de novedades, etc...

Este tipo de opciones predefinidas por nosotros pueden ser expresadas por medio de diferentes campos de formulario. Veamos a continuación cuales son:

## Listas de opciones

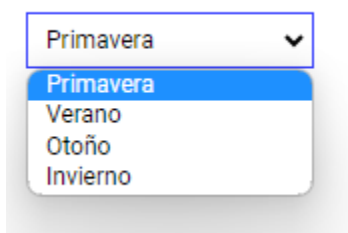
Las listas de opciones son ese tipo de menús desplegables que nos permiten elegir una (o varias) de las múltiples opciones que nos proponen. Para construirlas emplearemos una etiqueta **SELECT**, con su respectivo cierre.

Como para los casos ya vistos, dentro de esta etiqueta definiremos su nombre por medio del atributo name. Cada opción será incluida en una línea precedida de la etiqueta **OPTION**.

Podemos ver, a partir de estas explicaciones, la forma más típica y sencilla de esta etiqueta:

```
<select name="estacion">
  <option>Primavera</option>
  <option>Verano</option>
  <option>Otoño</option>
  <option>Invierno</option>
</select>
```

Esto en una página se vería así:



## Atributos para la etiqueta select

Esta estructura puede verse modificada principalmente a partir de otros dos atributos:

- **size:** indica el número de valores mostrados a la vez en la lista. Lo típico es que no se incluya ningún valor en el atributo size, en ese caso tendremos un



campo de opciones desplegable, pero si indicamos un valor para el atributo `size` aparecerá un campo donde veremos las opciones definidas por `size` y el resto podrán ser vistos por medio de la barra lateral de desplazamiento.

- **multiple:** permite la selección de más elementos de la lista. Este atributo se expresa sin valor alguno, es decir, no se utiliza con el igual, simplemente se pone para conseguir el efecto, o no se pone si queremos una lista desplegable común.

```
<select name="estacion" size="3" multiple>
  <option>Primavera</option>
  <option>Verano</option>
  <option>Otoño</option>
  <option>Invierno</option>
</select>
```

Esto renderizado se vería así:



Vemos que se pueden seleccionar múltiples opciones (en este caso, "Primavera" y "Verano") y que además sólo son visibles 3 opciones.

### Atributos para la etiqueta `option`

La etiqueta `<option>` puede asimismo ser modificada por medio de otro atributo.

- **selected:** del mismo modo que `multiple`, este atributo no toma ningún valor, sino que simplemente indica que la opción que lo presenta está elegida por defecto.

```
<option selected>Otoño</option>
```

## Botones de radio

Existe otra alternativa para plantear una elección, en este caso, obligamos al usuario a elegir únicamente una de las opciones que se le proponen.

La etiqueta empleada en este caso es **<input>** en la cual usaremos el atributo type con el valor radio. Este atributo colocará una casilla pinchable al lado del valor del input.

```
<input type="radio" name="estacion" value="1">Primavera  
<br>  
<input type="radio" name="estacion" value="2">Verano  
<br>  
<input type="radio" name="estacion" value="3">Otoño  
<br>  
<input type="radio" name="estacion" value="4">Invierno
```

Esto en una página web se vería así:

☒ Primavera  
☐ Verano  
☐ Otoño  
☐ Invierno

En este tipo de input para elegir una opción debemos tocar en la casilla clickeable, pero habíamos explicado previamente en la etiqueta label, que podíamos hacer que la etiqueta label al clickearla se active la caja de texto del input. Ahora, podemos usar eso para que cuando el usuario clickee la palabra primavera se seleccione esa opción. Esto se vería así:

```
<input type="radio" id="primavera" name="estacion" value="1">  
<label for="primavera">Primavera</label>  
<br>  
<input type="radio" id="verano" name="estacion" value="2">  
<label for="verano">Verano</label>  
<br>  
<input type="radio" id="otonio" name="estacion" value="3">
```

```
<label for="otoño">Otoño</label>
<br>
<input type="radio" id="invierno" name="estacion" value="4">
<label for="invierno">Invierno</label>
```

Esto en la página se verá igual que el anterior. La única diferencia va a ser que el usuario va a poder clicar el nombre de la estación que quiere para seleccionar esa opción, además de poder clicar la casilla.

### Cajas de validación

Este tipo de elementos pueden ser activados o desactivados por el visitante por un simple click sobre la caja en cuestión. Para esto vamos a usar la etiqueta INPUT con el valor checkbox en el atributo type.

```
<input type="checkbox" name="estacion" value="1">Primavera
```

Esto se verá así:

☐ Primavera

## Botones

Ha llegado el momento de explicar cómo podemos hacer un botón para provocar el envío del formulario, entre otras cosas.

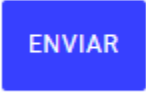
Como podremos imaginarnos, en formularios no solamente habrá elementos o campos donde solicitar información del usuario, sino también habrá que implementar otra serie de funciones. Concretamente, han de permitirnos su envío mediante un botón. También puede resultar práctico poder proponer un botón de borrado o bien acompañar el formulario de datos ocultos que puedan ayudarnos en su procesamiento.

### Botón de envío (submit)

Para dar por finalizado el proceso de relleno del formulario y hacerlo llegar a su gestor, el usuario ha de enviarlo por medio de un botón previsto a tal efecto. Para esto vamos a utilizar la etiqueta **<button>** y su respectivo cierre. Dentro el elemento button se puede poner texto (y etiquetas como <i>, <b>, <strong>, <br>, <img>, etc.).

```
<button type="submit">Enviar</button>
```

Esto en la página se verá así:



Como puede verse, tan solo hemos de especificar que se trata de un botón de envío (type="submit") y hemos de definir el mensaje que queremos que aparezca escrito en el botón.

### Botón de borrado (reset)

Este botón nos permitirá borrar el formulario por completo, en el caso de que el usuario desee rehacerlo desde el principio. Su estructura sintáctica es parecida a la anterior:

```
<button type="reset">Borrar</button>
```

A diferencia del botón de envío, indispensable en cualquier formulario, el botón de borrado resulta meramente optativo y no es utilizado frecuentemente. Hay que tener cuidado de no ponerlo muy cerca del botón de envío y de distinguir claramente el uno del otro, para que ningún usuario borre el contenido del formulario que acaba de rellenar por error.

## Botones normales

Dentro de los formularios también podemos colocar botones normales, pulsables como cualquier otro botón. Estos botones por si solos no tienen mucha utilidad, pero podremos necesitarlos para realizar acciones en el futuro. Su sintaxis es la siguiente:

```
<button type="button">Borrar</button>
```

## Datos ocultos (hidden)

En algunos casos, aparte de los propios datos rellenos por el usuario, puede resultar práctico enviar datos definidos por nosotros mismos que ayuden al programa en su procesamiento del formulario. Este tipo de datos, que no se muestran en la página, pero sí pueden ser detectados solicitando el código fuente, no son frecuentemente utilizados por páginas construidas en HTML, son más bien usados por páginas que emplean tecnologías de servidor. No se asusten, veremos más adelante qué quiere decir esto. Tan solo queremos dar constancia de su existencia y de su modo de creación.

He aquí un ejemplo:

```
<input type="hidden" name="instituto" value="Egg Educación">
```

Esta etiqueta, incluida dentro de nuestro formulario, enviará un dato adicional al programa encargado de la gestión del formulario.

## Ejemplo completo de formulario

Ya hemos visto todo lo respectivo a formularios. Pasemos ahora a ejemplificar todo lo aprendido a partir de la creación de un formulario.

Si tienes alguna duda, te recomendamos complementar el aprendizaje con los videos respectivos.


```
<form action="ruta del método que va a manejar la información"
method="POST" enctype="multipart/form-data">
  <label>Nombre del usuario</label> <br>
  <input type="text" name="nombre"> <br>
  <label>Edad del usuario</label> <br>
  <input type="number" name="edad "> <br>
  <label>Fecha de nacimiento del usuario</label> <br>
  <input type="date" name="fechanac"> <br>
  <label>Sexo del usuario</label> <br>
  <input type="radio" name="sexo" value="Hombre"> Hombre <br>
  <input type="radio" name="sexo" value="Mujer"> Mujer <br>
  <label>País nacimiento del usuario</label> <br>
  <select name="pais">
    <option>Argentina</option>
    <option>Brasil</option>
    <option>Chile</option>
    <option>Uruguay</option>
  </select>
  <br>
  <button type="submit">Enviar</button>
  <button type="reset">Borrar</button>
</form>
```

Esto se vería de la siguiente manera:

Nombre del usuario

Edad del usuario

Fecha de nacimiento del usuario


 

Sexo del usuario

☐ Hombre

☐ Mujer

País nacimiento del usuario

ENVIAR

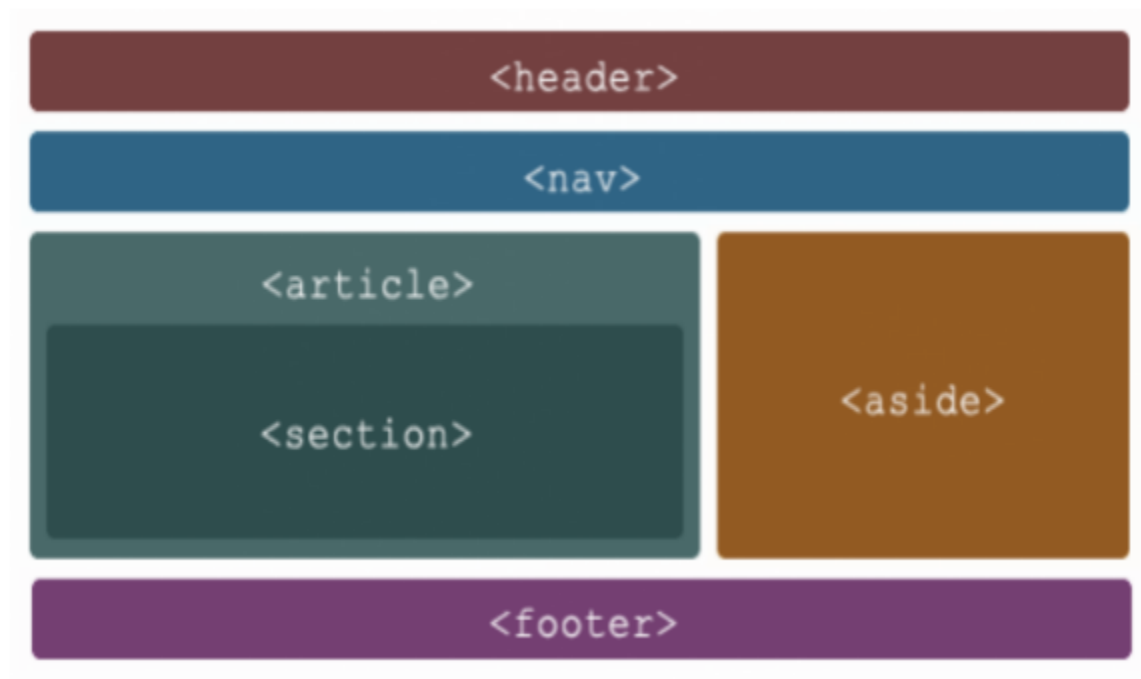
BORRAR

## HTML semántico

Las páginas web se trabajan con lo que se conoce como un esquema. El esquema (outline) de una página web es un índice de los apartados de una página web que muestra la relación de jerarquía entre los diferentes apartados y subapartados. El concepto de esquema se formalizó en HTML 5 con más precisión que en HTML 4 / XHTML 1 y explica algunas características y formas de utilización de las etiquetas de secciones y bloques de contenido.

En relación con esto se pensó que las páginas de HTML se pueden dividir en secciones y en HTML 5 se introdujo una serie de etiquetas que nos van a ayudar con la división de nuestra página en secciones. Dentro de cada sección van a haber más etiquetas, esto es simplemente para que podemos tener un índice de los apartados de la página web.

La página dividida en secciones con sus respectivas etiquetas se ve así:





```

<body>
  <header>
    <a href="/"><img src=logo.png alt="home"></a>
    <hgroup>
      <h1>Title</h1>
      <h2 class="tagline">
        A lot of effort went into making this effortless.
      </h2>
    </hgroup>
  </header>
  <nav>
    <ul>
      <li><a href="#">home</a></li>
      <li><a href="#">blog</a></li>
      <li><a href="#">gallery</a></li>
      <li><a href="#">about</a></li>
    </ul>
  </nav>
  <section class="articles">
    <article>
      <time datetime="2009-10-22">October 22, 2009</time>
      <h2>
        <a href="#" title="link to this post">Travel day</a>
      </h2>
      <div class="content">
        Content goes here...
      </div>
      <section class="comments">
        <p><a href="#">3 comments</a></p>
      </section>
    </article>
  </section>
  <aside>
    <div class="related"></div>
  </aside>

```

- **<section>**: se utiliza para representar una sección "general" dentro de un documento o aplicación, como un capítulo de un libro. Puede contener subsecciones y si lo acompañamos de h1-h6 podemos estructurar mejor toda la página creando jerarquías del contenido, algo muy favorable para el buen posicionamiento web.

- **«article»**: representa un componente de una página que consiste en una composición autónoma en un documento, página, aplicación, o sitio web con la intención de que pueda ser reutilizado y repetido.
- **«aside»**: representa una sección de la página que abarca un contenido relacionado con el contenido que lo rodea, por lo que se le puede considerar un contenido independiente. Este elemento puede utilizarse para efectos tipográficos, barras laterales, elementos publicitarios u otro contenido que se considere separado del contenido principal de la página.
- **«header»**: representa un grupo de artículos introductorios o de navegación. Está destinado a contener por lo general la cabecera de la sección (un elemento h1-h6 o un elemento hgroup).
- **«nav»**: representa una sección de una página que enlaza a otras páginas o a otras partes dentro de la página. No todos los grupos de enlaces en una página necesita estar en un elemento nav, sólo las secciones que constan de bloques de navegación principales son apropiadas para el elemento de navegación.
- **«footer»**: representa el pie de una sección, con información acerca de la página/sección que poco tiene que ver con el contenido de la página, como el autor, el copyright o el año.
- **«hgroup»**: representa el encabezado de una sección. El elemento se utiliza para agrupar un conjunto de elementos h1-h6 cuando el título tiene varios niveles, tales como subtítulos o títulos alternativos.

## Etiquetas extras

En este apartado que va a ser el último de nuestra parte de HTML vamos a ver unas etiquetas que no hemos visto todavía y que son importantes.

### Etiqueta div

La etiqueta div se conoce como etiqueta de división. Esta etiqueta se usa en HTML para hacer divisiones de contenido en la página web como (texto, imágenes, encabezado, pie de página, barra de navegación, etc.). La etiqueta div tiene

etiquetas de apertura **<div>** y de cierre **</div>** y es obligatorio cerrar la etiqueta. Div es la etiqueta más útil en el desarrollo web porque nos ayuda a separar datos en la página web y podemos crear una sección particular para datos o funciones particulares en las páginas web. Cabe aclarar que la etiqueta div genera un salto de línea.

- La etiqueta Div es una etiqueta de nivel de bloque
- Es una etiqueta de contenedor genérica
- Se utiliza para agrupar varias etiquetas de HTML para que se puedan crear secciones y aplicarles estilo en conjunto.

Supongamos que tenemos 3 párrafos que queremos alinear a la izquierda, esto recordemos lo haríamos con el atributo align.

Crearíamos algo así:

```
<p align="left">Párrafo 1</p>
<p align="left">Párrafo 2</p>
<p align="left">Párrafo 3</p>
```

Una forma de simplificar nuestro código anterior y de evitar introducir continuamente el atributo align sobre cada una de nuestras etiquetas es utilizando la **etiqueta div**. Por lo tanto, usaremos un div para generar una sección de todos los párrafos y le agregaremos el atributo align a ese div.

Con div se vería así:

```
<div>
<p align="left">Párrafo 1</p>
<p align="left">Párrafo 2</p>
<p align="left">Párrafo 3</p>
</div>
```

Como hemos visto, la etiqueta DIV marca divisiones en las que definimos un bloque de contenido, y a los que podríamos aplicar estilo de manera global, aunque lo correcto sería aplicar ese estilo del lado del CSS.

## Etiqueta span

El elemento span HTML es un contenedor en línea genérico para elementos y contenido en línea. Se usa para agrupar elementos con fines de compartir un determinado estilo (mediante el uso de los atributos de clase o id).

La mejor manera de usarlo es cuando no hay ningún otro elemento semántico disponible. Esta etiqueta es muy similar a la etiqueta div, pero se diferencian en que div es una etiqueta a nivel de bloque y span es una etiqueta en línea.

La etiqueta span es una etiqueta emparejada, lo que significa que tiene una etiqueta de apertura (<) y de cierre (>), y es obligatorio cerrar la etiqueta.

La etiqueta span se utiliza para agrupar elementos en línea y no realiza ningún cambio visual por sí misma.


En el siguiente caso vemos que tenemos un párrafo donde necesitamos agregar estilos para una palabra. Por lo que agregaremos una etiqueta span para poder implementar ese estilo deseado.

```
<p>Yo estudio en <span style="color: yellow">Egg</span> Programación FullStack </p>
```

Esto en una página se vería así:

Yo estudio en Egg Programación FullStack

La etiqueta span no crea un salto de línea, sino que permite al usuario elegir cuando separar un elemento de otros dentro de la misma línea.

 **Nota:** en el apartado de CSS vamos a ver mejor el atributo style y el atributo color. Ahora los estamos usando sólo para ejemplificar.

Hemos llegado al final de **HTML** y has adquirido un amplio conocimiento, desde los fundamentos del lenguaje hasta numerosos aspectos más avanzados.