**HTML**

**1. Introducción**

**1.1.** **¿Qué significan las siglas HTML?**

**HTML** es la abreviatura de **"HyperText Mark-up Language"**, es decir, "Lenguaje de marcado hipertextual".

**Hiper** es lo contrario de lineal. En los viejos tiempos (cuando un ratón era un animalillo que perseguía un gato) los programas de ordenador se ejecutaban de forma lineal: cuando el programa había ejecutado una acción seguía hasta la siguiente línea, y después de ésta a la siguiente, y a la siguiente. HTML, sin embargo, es diferente: uno puede ir donde quiera, cuando quiera. Por ejemplo, no es necesario visitar MSN.como antes de visitar HTML.net.

**Texto** se explica por sí solo.

**Marcado** es lo que haces con el texto. Se marca el texto del mismo modo que en un programa de edición de textos con encabezados, viñetas, negrita, etc.

**Lenguaje** es lo que es HTML. Este lenguaje hace uso de muchos términos en inglés.

**1.2. ¿Cómo instalar HTML en mi proyecto?**

Los archivos HTML que producen páginas web son **archivos de texto simple**. El texto es una forma universal de representación de datos para ordenadores.

El html no **necesita ser instalado** ni preparar ningún entorno de trabajo especial para usarlo.

Para usar html solo necesitamos crear un archivo con **la terminación “.html”**. Este archivo puede ser ejecutado por cualquier **navegador**. (tanto html, como css y javascript pueden ejecutarse y son entendidos perfectamente por los navegadores, también conocidos como clientes).

Para crear este archivo nos basta con cualquier **editor de textos plano**, como el blog de notas (sí, con algo tan simple se puede construir un html), aunque, por regla general utilizaremos editores de textos especializados que nos permiten escribir mucho más rápido y de forma profesional (por ejemplo, Atom).

**2. Elementos y etiquetas**

Los **elementos** proporcionan la estructura al documento HTML e indican al navegador cómo quieres que se presente tu sitio web. Por lo general, los elementos están formados por una **etiqueta de inicio,** el contenido, y una **etiqueta de cierre**.

Las etiquetas son marcas que se usan para señalar el inicio y el fin de un elemento.

Todas las etiquetas comparten el mismo formato: empiezan con el signo menor que "**<**" y terminan con el signo mayor que "**>**".

Por lo general, hay dos tipos de etiquetas: la etiquetas de inicio, por ejemplo, <html>, y las etiquetas de cierre, por ejemplo, </html>. La única diferencia entre la etiqueta de inicio y la de cierre es la barra oblicua "**/**". El contenido queda etiquetado al colocarlo entre una etiqueta de inicio y una etiqueta de cierre.

HTML trata esencialmente de elementos. Aprender HTML consiste en aprender a usar diferentes etiquetas.

**2.1. ¿Siempre son necesarias las etiqueta de inicio y de cierre?**

Como se dice, siempre hay una excepción que confirma la regla, y en HTML la excepción son unos cuantos elementos que se abren y cierran con la misma etiqueta. Estos elementos que se denominan elementos vacíos no se asocian a un pasaje específico del texto sino que son más bien etiquetas aisladas; por ejemplo, un salto de línea tiene este formato: <br/>.

**2.3. ¿Las etiquetas hay que escribirlas en mayúscula o en minúscula?**

A la mayoría de los navegadores les da igual si se escriben en mayúscula, minúscula o una mezcla de las dos. <HTML>, <html> or <HtMl> normalmente tendrán el mismo resultado. Sin embargo, el modo correcto es escribir las etiquetas en minúscula. Así que acostúmbrate a escribir las etiquetas en minúscula.

**2.3. ¿Y dónde coloco todas estas etiquetas?**

Las etiquetas se escriben dentro de un documento HTML. Todo sitio web contiene uno o más documentos HTML. Cuando navegas por la web, no haces otra cosa que abrir diferentes documentos HTML.

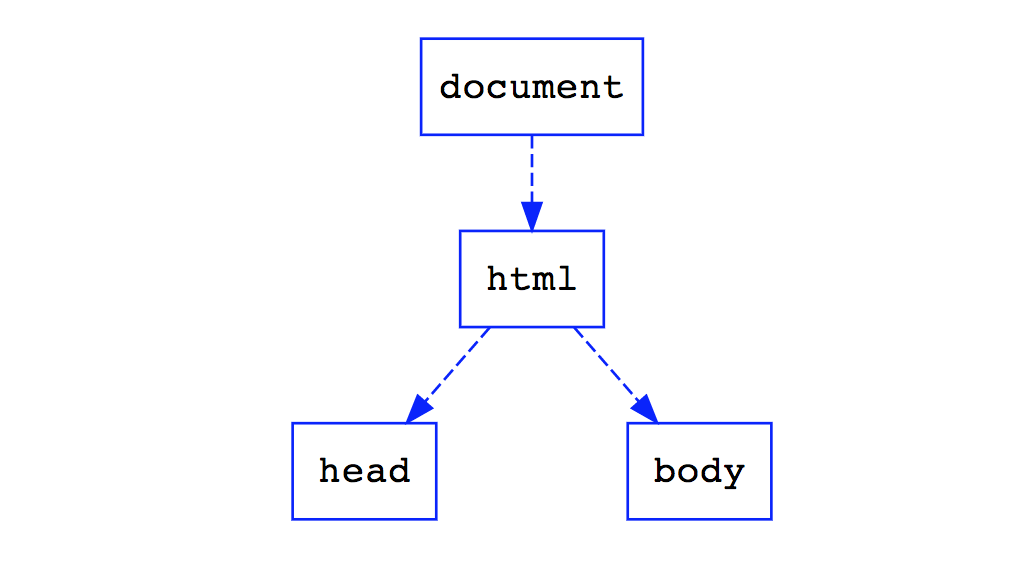
RESUMEN DE TODAS LAS ETIQUETAS HTML PARA ANEXAR AL PDF

|  |
| --- |
| <https://drive.google.com/open?id=1Wi8I4XwarnDQ7ZjnJa3KlFkkrWrpYXM8hDYjsj6at-Y> |

**3. HTML DOM**

Cada documento HTML puede ser representado como un árbol utilizando el **Document object Model (DOM)**, que contiene todos los elementos del documento HTML, el formato y el estado del navegador en un momento específico.

La estructura de DOM se suele representar como un **árbol de nodos**, donde podemos ver qué nodos (como <html>) tenemos, y qué otros nodos están dentro de nodos, por ejemplo <head>.



En el **DOM HTML**, todo es un nodo. El document es un nodo. Pero los elementos, atributos y el contenido en el interior de los elementos son nodos, también. Todos son nodos.

Vamos a echar un vistazo al código en el ejemplo dado y vamos a explicar algunos de la sintaxis:

**EJEMPLO**

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

**<!DOCTYPE html>** → DOCTYPE Indica que el lenguaje de marcado para su contenido del documento es HTML5.

**<html>** → html representa la raíz (root) de un documento HTML. Todos los demás elementos deben ser descendientes de este elemento. Es el primer nodo en nuestra DOM. Es obligatorio cerrar la etiqueta en la parte final del documento escribiendo </html>.

**<head>** → Head define un elemento que proporciona información general (metadatos) sobre el documento, incluyendo su título y links a sus scripts y hojas de estilo.

**<title>** → Define el título del documento, que se mostrará en la pestaña del navegador. Hay uno por documento html.

**<meta>** → Se utiliza para definir los metadatos. Esto incluye información sobre styles, scripts y datos para ayudar a utilizar navegadores y representar la página. Uno de los elementos más comunes es la <meta charset="UTF-8"> de nuestro ejemplo. Esto especifica la codificación de caracteres para el documento HTML como [UTF-8](https://en.wikipedia.org/wiki/UTF-8) .

**<body>** → Body es el elemento que contiene todo el contenido de un documento HTML. Cada componente HTML debe ser escrito entre la apertura y el cierre de la etiqueta body. Solo puede haber un elemento body en un document.

**4. Tipos de elementos en html**

**4.1. Hide elements**

Son elementos que no se muestran en absoluto. Estos elementos proporcionan información sobre la página, pero no se muestran cuando se representa en un navegador web (como todos los elementos de nuestra sección <head>). El resto de elementos los veremos a continuación con mayor detalle.

**4.2. Block elements**

Dentro del **<body>**, vamos a escribir todos los elementos que queremos mostrar en nuestra página web. Estos elementos serán o block-elements o inline-elements.

Los **block elements** comienzan en una nueva línea de la página web y, si no se establece la anchura, se extienden por todo el ancho del espacio horizontal disponible de su elemento padre.

**Div <div></div>**

Sirve para crear secciones o agrupar contenidos. Es un elemento de bloque que crea una caja. Puede contener texto y otros elementos en bloque o en línea.

|  |  |
| --- | --- |
| <div style="color: blue;">  <h2> Ejemplo de div y span </h2>  <p>  Esto es un párrafo dentro de un div,  <span style="color: red;"> y esto un span dentro de un párrafo.</span>   </p> </div> | |

**Los párrafos <p></p>**

Representa un párrafo de texto.

**Ejemplo**

Un párrafo <p> es un ejemplo de elemento de bloque.

|  |  |
| --- | --- |
| <p>An ongoing collaboration between the classroom staff and the development team allow us to iterate our content and give our students the coding 'fruits and vegetables' -mainly made of Javascript- that they need to grow as a healthy developer.</p> | |

**Texto preformateado**

El elemento **<pre>** representa el texto preformateado. El texto dentro de este elemento por lo general se muestra en una fuente no proporcional (“monovolumen”).

**Headings <h1></h1> - <h6></h6>**

Los **elementos de encabezado** tienen seis niveles de encabezados de documentos (títulos): <h1> es el más importante y <h6> es el menos. Un título describe brevemente el tema de la sección que introduce.

**Ejemplo**

El código siguiente muestra todos los niveles de título, en uso.

|  |  |
| --- | --- |
| <h1>Heading level 1</h1> <h2>Heading level 2</h2> <h3>Heading level 3</h3> <h4>Heading level 4</h4> <h5>Heading level 5</h5> <h6>Heading level 6</h6> | |

**Listas**

**Lista ordenada <ol></ol>**

Representa una lista ordenada de elementos. Cada elemento de la lista está numerada.

**Lista desordenada <ul></ul>**

Representa una lista de elementos también. Sin embargo, el orden de los elementos no es relevante.

No hay limitación a la profundidad y la superposición de las listas definidas con <ol> y <ul>.

**Elementos de la Lista <li></li>**

Los elementos dentro de las listas.

**Ejemplo**

Este sería un ejemplo de código para una lista ordenada:

|  |  |
| --- | --- |
| <ol>  <li>first item</li>  <li>second item</li>  <li>third item</li> </ol> | |

**Tabla <table></table>**

Se utiliza para expresar la información en una tabla de datos de dos dimensiones, pero se requiere de varios otros elementos en el interior para que tenga sentido:

**<tr></tr>** define una fila de celdas en una tabla. y dentro de esta etiqueta:

**<th></th>** Si el elemento dentro de una fila define una cabecera (head de la tabla).

**<td>** (table data), para datos normales.

**Ejemplo**

Este sería un ejemplo sencillo de una tabla:

|  |  |
| --- | --- |
| <table>  <tr>  <th>Name</th>  <th>Bootcamp</th>  <th>City</th>  </tr>  <tr>  <td>Jose Perez</td>  <td>FullStack WebDev</td>  <td>Madrid</td>  </tr>  <tr>  <td>Michael Smith</td>  <td>FrontEnd WebDev</td>  <td>Barcelona</td>  </tr>  <tr>  <td>Ariel Quinones</td>  <td>MasterChef WebDev</td>  <td>Miami</td>  </tr> </table> | |

**Formularios <form></form>**

Es una sección que permite al usuario introducir información en la página. Los elementos <form> incluyen entradas, botones radio, cuadros de selección, etc. Por lo general, vamos a enviar esta información a un **servidor web**.

El elemento <form> requiere dos **atributos** muy importantes para que funcione correctamente. Esos son los atributos **action** y **method**.

**Atributo action.** Este atributo establece el [URI](https://developer.mozilla.org/en-US/docs/Glossary/URI) para el programa que procesa la información del formulario. En otras palabras, se le dice al formulario a donde enviar la información.

**Atributo method.** Este atributo establece el método HTTP que el navegador utilizará para enviar el formulario. Vamos a aprender sólo el método post por ahora. El método post se corresponde con el [método POST HTTP](https://en.wikipedia.org/wiki/POST_(HTTP)). Este método envía los datos insertados en el formulario al servidor.

**Ejemplo**

Un ejemplo de un simple form con una solicitud posterior sería la siguiente:

|  |  |
| --- | --- |
| <form action="" method="post">  <label for="name">Name:</label>  <input id="name" type="text">  <input type="submit" value="Save"> </form> | |

**Fieldset <fieldset></fieldset>**

Se utiliza para varios controles de grupo así como etiquetas (<label>) dentro de un formulario web.

Habrá distintos elementos fieldset dentro de un form por diferentes razones: efectos de estilo, organización y características de [accesibilidad web](https://en.wikipedia.org/wiki/Web_accessibility)

**Ejemplo**

Este es un ejemplo formulario con fieldset. Sólo incluye un elemento que puede hacer clic.

|  |  |
| --- | --- |
| <form action="test.php" method="post">  <fieldset>  <input type="radio" id="radio"> <label for="radio">Click me</label>  </fieldset> </form> | |

**Crear un formulario simple**

Para construir nuestro formulario de contacto, vamos a utilizar los siguientes elementos de HTML: [<form>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/form), [<label>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/label), [<input>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/input), [<textarea>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/textarea), y [<button>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/button).

**1. El elemento** [**<form>**](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/form)

Todos los formularios HTML comienzan con el elemento [<form>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/form) de la siguiente forma:

|  |  |
| --- | --- |
| <form action="/my-handling-form-page" method="post"> </form> | |

Este elemento define formalmente un formulario. Es un contenedor como lo son [<div>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/div) o [<p>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/p), pero también soporta algunos atributos específicos para configurar la forma en que el formulario se comporta. Todos sus atributos son opcionales pero se considera una buena práctica que siempre al menos los atributos action y method se encuentren presentes.

* El atributo action define la locación (una URL) desde donde la información recolectada por el formulario debería enviarse.
* El atributo method define con que método HTTP se enviará la información (puede ser "get" o "post").

Si quieres introducirte en como estos atributos funcionan, lo veras detallado en el artículo [Enviando y recuperando información de formularios](https://developer.mozilla.org/en-US/docs/HTML/Forms/Sending_and_retrieving_form_data).

**Agrega widgets con los elementos** [**<label>**](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/label)**,** [**<input>**](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/input)**, y [<textarea>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/textarea)**

Nuestro formulario de contacto es muy simple y contiene tres campos de texto, cada uno con un label. El campo de texto para el nombre será de linea simple; el del e-mail también será de linea simple que aceptará solamente una dirección de correo electrónico; el campo de texto para el mensaje será multilínea.

En términos de código HTML vamos a tener algo como lo siguiente:

|  |  |
| --- | --- |
| <form action="/my-handling-form-page" method="post">  <div>  <label for="name">Name:</label>  <input type="text" id="name" />  </div>  <div>  <label for="mail">E-mail:</label>  <input type="email" id="mail" />  </div>  <div>  <label for="msg">Message:</label>  <textarea id="msg"></textarea>  </div> </form> | |

Los elementos [<div>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/div) están allí para estructurar nuestro código de forma conveniente y poder darles estilos de una forma más sencilla (ver abajo). Puedes notar el uso de atributo for en todos los elementos [<label>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/label); es la forma correcta de unir un label a un widget en un formulario. Este atributo referencia el id del widget correspondiente. Hay algunos beneficios de hacer esto. El más obvio es permitir al usuario hacer click en el label para activar el widget correspondiente. Si quieres aprender otros beneficios, tienes todo detallado en el artículo: [Como estructurar un formulario HTML](https://developer.mozilla.org/en-US/docs/HTML/Forms/How_to_structure_an_HTML_form).

En el elemento [<input>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/input), el atributo mas importante es type. Este atributo es sumamente importante porque define la forma en que el elemento [<input>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/input) se comporta. Puede cambiar radicalmente el elemento, asi que presta atención a esto. Si quieres saber más sobre esto, lee el artículo [Widgets nativos de formularios](https://developer.mozilla.org/en-US/docs/HTML/Forms/The_native_form_widgets). En nuestro ejemplo usamos únicamente el valor text - el valor por defecto de este atributo. Representa un campo de texto básico de una linea que acepta cualquier tipo de texto sin ningún control o validación.

Por último pero no menos importante, veamos la sintaxis del <input /> vs. <textarea></textarea>. Esta es una de las extrañezas del HTML. La etiqueta <input> se cierra a si misma, lo que significa que si quieres cerrar formalmente el elemento, tienes que agregar un " / " al final del mismo y no una etiqueta de cierre. Por el contrario, [<textarea>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/textarea) no es un elemento que se cierre a si mismo, por lo que tienes que cerrarlo con su etiqueta correspondiente. Esto tiene impacto sobre una característica específica de los formularios HTML: la forma en que defines el valor por defecto. Para definir el valor por defecto de un elemento [<input>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/input) debes usar el atributo value de la siguiente forma:

|  |  |
| --- | --- |
| <input type="text" value="by default this element is filled with this text" /> | |

Por el contrario, si quieres definir el valor por defecto de un [<textarea>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/textarea), solamente tienes que poner el valor por defecto entre el comienzo y el final de las etiquetas del elemento [<textarea>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/textarea), así:

|  |  |
| --- | --- |
| <textarea>by default this element is filled with this text</textarea> | |

**Y un** [**<button>**](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/button) **para finalizar**

Nuestro formulario está casi listo; tenemos que agregar un botón para permitir al usuario enviar la información que llenó en el formulario. Esto se hace simplemente usando el elemento [<button>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/button):

|  |  |
| --- | --- |
| <form action="/my-handling-form-page" method="post">  <div>  <label for="name">Name:</label>  <input type="text" id="name" />  </div>  <div>  <label for="mail">E-mail:</label>  <input type="email" id="mail" />  </div>  <div>  <label for="msg">Message:</label>  <textarea id="msg"></textarea>  </div>    <div class="button">  <button type="submit">Send your message</button>  </div> </form> | |

Un botón puede ser de tres tipos: submit, reset, o button.

* Un click en un botón submit envía la información del formulario a una pagina web definida por defecto en el atributo action del elemento [<form>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/form).
* Un click en un botón reset reinicia inmediatemente todos los widgets del formulario a sus valores por defecto. Desde un punto de vista de UX, esto se considera una mala práctica.
* Un click en un botón button... ¡no hace nada! Puede sonar tonto, pero es muy útil para construir botones customizados con JavaScript.

Puedes también usar el elemento [<input>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/input) con el type correspondiente para producir un botón. La diferencia principal con el elemento [<button>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/button) es que el elemento [<input>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/input)únicamente permite texto plano como su label mientras que el elemento [<button>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/button)permite contenido HTML en su label.

**Listado de elementos block**

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Descripción** | **Ejemplo de uso / Output** |
| <p></p> | Párrafo | Bienvenido a su sitio web |
| <pre></pre> | texto preformateado | ¡Hola Mundo! |
| <h1></h1> | La partida (título) | Hola Mundo! |
| <ol></ol> | Lista ordenada | 1. Uno 2. Dos |
| <ul></ul> | Lista desordenada | * Uno * Dos |
| <li></li> | Producto en la lista | Uno |
| <table></table> | 2-D de datos | [Ir a codepen](http://codepen.io/ironhack/pen/BLLWwY) |
| <form></form> | presentar información | [Ir a codepen](http://codepen.io/ironhack/pen/dppvqv) |
| <fieldset></fieldset> | Grupo de elementos de formulario | [Ir a codepen](http://codepen.io/ironhack/pen/xEEAwz) |

**4.3. Inline Elements**

Los **elementos en línea** fluyen como texto. No comienzan una nueva línea y se muestran justo al lado del elemento anterior sin borrar el contenido anterior.

**Enfatizando el texto <em></em>**

**Ejemplo**

Hacer hincapié en las palabras “impecable” y “calidad” envolviéndolas en <em>:

|  |  |
| --- | --- |
| ... commitment is to use **<em>**impeccable**</em>** sourcing and **<em>**quality**</em>** ingredients... | |

**¿Cuándo usar <i> y cuándo usar <em>?**

El resultado visual es, por defecto, el mismo (ambas marcas generan sus contenidos en cursiva). Pero el significado semántico es diferente.

Un ejemplo para **<em>** podría ser: “Hemos tenido que hacer algo al respecto”. Una persona o un software de lectura del texto entenderían que tienen que pronunciar las palabras en cursiva con énfasis.

Un ejemplo para **<i>** podría ser: “El Queen Mary navegó ayer por la noche”. Aquí, no hay énfasis añadido a la importancia de la palabra “Queen Mary”. Meramente se indica que el objeto en cuestión no es una reina llamada María, es un barco llamado Queen Mary. Otro ejemplo para <i> podría ser: “La palabra el es un artículo”.

**Strong <strong></strong> o <b></b>**

Cuando se da una gran importancia al texto por lo general se muestra en bold.

**Enlaces (anclajes) <a></a**

Define un hipervínculo a una ubicación en la misma página o cualquier otra página en la Web. Por lo general, seguido por el atributo 'href'.

**Ejemplo**

|  |
| --- |
| <a href = "<http://www.example.com>">This is a example</a> |

**Puntos de fijación: anclas**

Cuando se tienen documentos extensos, divididos en varios apartados, es preferible poder ir directamente al apartado deseado, en lugar de ir al comienzo del documento. Para ello se utilizan las **anclas**, o puntos de fijación, muy útiles a la hora de crear índices.

Un ancla necesita que se inserten las <a> y </a>, con el atributo name, que puede tomar cualquier valor inventado por el usuario, se recomienda no utilizar caracteres especiales.

**Ejemplo**

Por ejemplo, para insertar un punto de fijación delante del siguiente texto:

|  |
| --- |
| <a name="miancla"></a>TEXTO ANCLA |

Como puedes ver, no es necesario insertar nada entre las etiquetas <a> y </a>, y que sin mirar el código no hay nada que indique que delante del texto haya un ancla.

Teniendo en cuenta que el documento actual se llama mypage.htm, y que el ancla anterior se llama miancla, podríamos crear un enlace que nos llevara directamente a la línea de texto en la que se encuentra el ancla. Por ejemplo:

|  |
| --- |
| <a href="mypage.htm#miancla">Enlace al ancla</a> |

Si pulsas sobre el enlace verás como se vuelve a cargar el documento actual, pero en lugar de cargarse desde el principio, la primera línea de texto será la línea en la que hemos insertado el ancla.

Cuando el ancla se encuentra en el mismo documento que el enlace, como ocurre en este caso, podemos prescindir de poner el nombre de la página en el atributo href. En el ejemplo anterior podríamos haber escrito:

|  |
| --- |
| <a href="#miancla">Enlace al ancla</a> |

**Break <br>**

Produce un salto de línea en el texto (back return) y no tiene cierre. No es muy aconsejable su uso en el html cuando tiene fines de decoración. Porque todo lo que tenga que ver con la apariencia debe tratarse con CSS. El uso incorrecto de la etiqueta <br> se penaliza en el SEO.

**Imágenes <img></img>**

Define una imagen. Es seguido por el atributo 'src' y es obligatorio añadir la URL de la imagen.

**Ejemplo**

|  |
| --- |
| <img src = "http://www.example.com"> |

**Script <script></script>**

Se utiliza para insertar o hacer referencia a un script ejecutable dentro de un archivo HTML.

**Spans <span></span>**

Es un contenedor en línea genérico para contenido estático, que no representa nada inherentemente. Puede ser utilizado para agrupar elementos para fines de estilo. Se debe utilizar sólo cuando ningún otro elemento semántico es apropiado. No recomendado.

**Buttons <button></button>**

Representa un botón que se pulsa.

**Input <input></input>**

Campo de entrada de datos. Es el elemento básico para introducir datos en cualquier formulario. El input admite varios atributos. Los más importantes son name (nombre del campo) y type (tipo de datos que vamos a introducir). Si el atributo type de la etiqueta input es submit estaremos creando un botón para enviar la información.

**Etiqueta Label <label></label>**

Leyenda para un campo. Esta etiqueta es la que va a mostrar el nombre del campo al usuario y también nos servirá para definir mejor el tipo de dato que vamos a introducir. Su atributo es for, que asocia la etiqueta a un id.

**Selects <select></select>**

Representa un control que presenta un menú de opciones. Las opciones del menú están representadas por elementos <option>, que pueden ser agrupados por elementos <optgroup>.

**Ejemplo**

|  |
| --- |
| <select name=“select">  <option value="one"><b>First option</b></option>  <option value="two">Second option</option>  <option value="three">Third option</option>  </select> |

**Textarea <textarea></textarea>**

Representa un control de edición de texto sin formato multi-línea. Puede indicar el tamaño especificando número de columnas y filas.

**Ejemplo**

|  |
| --- |
| <textarea>Write here </textarea> |

**Listado de elementos inline**

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Descripción** | **Ejemplo de uso / Output** |
| <em></em> | Destaca el texto | Somos el mejor en la ciudad |
| <i></i> | El texto en cursiva | *Javascript* es un lenguaje |
| <strong></strong> | Texto en negrita | Este es el **último día** para aplicar |
| <b></b> | Texto en negrita | Este es el **último día** para aplicar |
| <time></time> | Las fechas y horarios | El concierto comienza a las 12:00 |
| <a></a> | Enlace | Ir a [Google](http://materials.ironhack.com/s/rkCNjQ6Sg) |
| <br> | salto de línea | Para ser  o no ser |
| <img> | Imagen | [Ir a codepen](http://codepen.io/ironhack/pen/VKjPPk) |
| <script></script> | El código incrustado | [Ir a codepen](http://codepen.io/ironhack/pen/JRKEWV) |
| <span></span> | recipiente inline Genérico | Algún texto |
| <button></button> | botón que se pulse | [Ir a codepen](http://codepen.io/ironhack/pen/WGxRjE) |
| <input></input> | campo de entrada de datos | [Ir a codepen](http://codepen.io/ironhack/pen/VKjPWw) |
| <label></label> | Leyenda para el artículo | [Ir a codepen](http://codepen.io/ironhack/pen/bwZgRW) |

**5. HTML5**

HTML5 es la última versión de HTML. El término representa dos conceptos diferentes:

* Se trata de una **nueva versión de HTML**, con nuevos elementos, atributos y comportamientos.
* Contiene un **conjunto más amplio de tecnologías** que permite a los sitios Web y a las aplicaciones ser más diversas y de gran alcance.

Diseñado para ser utilizable por todos los desarrolladores de Open Web, referenciamos numerosos recursos sobre las tecnologías de HTML5, clasificados en varios grupos según su función.

* **Semántica**: Permite describir con mayor precisión cuál es su contenido.
* **Conectividad**: Permite comunicarse con el servidor de formas nuevas e innovadoras.
* **Sin conexión y almacenamiento**: Permite a las páginas web almacenar datos localmente en el lado del cliente y operar sin conexión de manera más eficiente.
* **Multimedia**: Nos otorga un excelente soporte para utilizar contenido multimedia como lo son audio y video nativamente.
* **Gráficos y efectos 2D/3D:** Proporciona una amplia gama de nuevas características que se ocupan de los gráficos en la web como lo son canvas 2D, WebGL, SVG, etc.
* **Rendimiento e Integración**: Proporciona una mayor optimización de la velocidad y un mejor uso del hardware.
* **Acceso al dispositivo**: Proporciona APIs para el uso de varios componentes internos de entrada y salida de nuestro dispositivo.
* **CSS3**: Nos ofrece una nueva gran variedad de opciones para hacer diseños más sofisticados.

Una de las más importantes novedades está relacionada con la inserción de multimedia en los sitios web, que ahora contarán con etiquetas HTML especiales para poder ser incluidos.

Por otro lado, algunos aspectos de diseño también son incluidos en el lenguaje, así como también algunos detalles de navegación.

Con el uso de HTML5, se puede reducir la dependencia de los plug-ins que tenemos que tener instalados para poder ver una determinada web. Caso emblemático, el de Adobe Flash, que se ve claramente perjudicado por la instauración de este estándar. Por otro lado, fue un avance importante para dispositivos que de forma nativa no soportaban Flash, y que no soportaban tampoco plug-ins necesarios para hacerlo.

Pero además, con HTML5 se amplía el horizonte del desarrollo de aplicaciones que pueden ser usadas en una multiplicidad de dispositivos.

Gracias a HTML5, los usuarios pueden acceder a sitios web de manera offline, sin estar conectados a internet. Se suma también la funcionalidad de drag and drop, y también la edición online de documentos ampliamente popularizada por Google Docs.

La geolocalización es uno de sus puntos fuertes, pero por otro lado, las etiquetas diseñadas especialmente para el audio y el video ahorran la necesidad de tener que tener un plug-in de Flash y, al mismo tiempo, asestan un golpe mortal al producto de Adobe, que cada vez se está usando menos.

**Las nuevas etiquetas**

Con HTML5, se agregan nuevas etiquetas para utilizar que nos ahorran el uso de otros productos que se usaban para complementar y hacer cosas que con el simple HTML no se podían hacer.

La idea detrás de HTML5 es que podamos visualizar el contenido multimedia variado que podemos encontrar en internet aún cuando nos encontramos en dispositivos de gama baja que no podrían soportarlo cuando tienen que instalar infinidad de plug-ins. No solamente contamos con etiquetas especiales como audio, video y canvas, sino también integración con contenidos de gráficos en vectores (que anteriormente se conocía como la etiqueta object). Con estas etiquetas, los usuarios pueden consumir videos y canciones, por ejemplo, sin necesidad de instalar nada de forma adicional.

Las más importantes de las nuevas etiquetas creadas son:

* **article**: esta etiqueta sirve para definir un artículo, un comentario de usuario o una publicación independiente dentro del sitio.
* **header**, **footer**: estas etiquetas individuales ahorran tener que insertar IDs para cada uno, como se solía hacer anteriormente. Además, se pueden insertar headers y footers para cada sección, en lugar de tener que hacerlo únicamente en general.
* **nav**: la navegación puede ser insertada directamente en el markup, entre estas etiquetas, que nos permitirán hacer que nuestras listas oficien de navegación.
* **section**: con esta etiqueta, una de las más importantes de las novedades, se puede definir todo tipo de secciones dentro de un documento. Por ponerlo de forma sencilla, funciona de una forma similar a la etiqueta div que nos separa también diferentes secciones.
* **audio** y **video**: estas son las dos más importantes etiquetas de HTML5, dado que nos permiten acceder de forma más simple a contenido multimedia que puede ser reproducido por casi todo tipo de dispositivos; marcan el tipo de contenido que estará en su interior.
* **embed**: con esta etiqueta se puede marcar la presencia de un contenido interactivo o aplicación externa.
* **canvas**: finalmente, esta etiqueta nos permite introducir un “lienzo” dentro de un documento, para poder dibujar gráficos por vectores; será necesario el uso de JavaScript.