

# Introducción

## Información General

El Lenguaje de Marcas de Hipertexto o “HyperText Markup Language” (HTML), es el principal lenguaje de marcas para todos los sitios web. Los elementos de HTML son los componentes básicos de internet.

### Objetivos

- Entender los componentes teóricos de la web y cómo interactúan.
- Comprender el modelo cliente-servidor.
- Entender las partes más importantes de lo que se muestra en el navegador.
- Entender el rol HTML vs CSS vs Javascript.
- Comprender cómo descomponer gradualmente la construcción de una página web en sus elementos fundamentales.

### Qué es HTML

HTML es el lenguaje de la Web. Significa Lenguaje de Marcas de Hipertexto.

Su propósito es permitirnos comunicar por medio del navegador el significado del contenido que deseamos poner en una página web. Para lograr esto, HTML define un número de **etiquetas (tags)** organizando el contenido de acuerdo a ellas.

- Headings (títulos) y párrafos del texto
- Imágenes
- Links (enlaces)
- Listas
- Tablas
- Formularios

HTML tiene una etiqueta predefinida para cada uno de estos, tema que cubriremos a continuación.

Las etiquetas de HTML **describen el contenido que contienen**. Por ejemplo, existen etiquetas para párrafos, etiquetas de anclaje para links, etc. **El navegador no muestra las etiquetas de HTML**, pero las usa para interpretar el contenido de la página.

## Por qué HTML y CSS

Los cursos de HTML y CSS están diseñados para guiarte a través de los componentes teóricos de la web y dar el primer paso en crear una aplicación web (desde el lado del cliente). En este contexto, creemos que la mejor forma de comenzar es con un prototipo “clicable” involucrando HTML, CSS y Javascript, por lo que es normal enseñar estas tecnologías primero.

### *Crea un Documento Simple en HTML*

- Crear un nuevo archivo en el Editor de Texto de tu elección
- Escribe “¡Hola Mundo!” en el archivo
- Guarda el archivo como **hola\_mundo.html**
- Arrastra el archivo a tu navegador para abrirlo

¡Eso es todo! Deberías ver el texto “Hola Mundo” en tu navegador.

Tal vez hayas notado que no agregamos ninguna etiqueta a nuestro texto. Para ver cómo el navegador maneja esta situación, presiona el botón derecho de tu mouse en tu navegador y selecciona **Inspeccionar**. Debiera abrirse un nuevo módulo en la base de tu navegador con varias etiquetas.

Lo que nos interesa es la pestaña **Elementos**. Cuando haces clic en ella, debieras ver:

```
<html>

  <head></head>

  <body>¡Hola Mundo!</body>

</html>
```

En el caso que tu documento en HTML no sea válido, el navegador intentará agregar las etiquetas correctas para arreglarlo.

**<html>**, **<head>** y **<body>** son necesarios para cada documento válido en HTML.

Unos cuantos términos:

- **Abriendo y cerrando etiquetas**

Puesto que las etiquetas debieran involucrar determinadas cosas, **la mayoría de ellas vienen de a pares**: una etiqueta de apertura y otra de cierre para señalar el comienzo y el fin del

contenido. Las etiquetas sin un “forward slash” (/) adelante son llamadas etiquetas de apertura, mientras que las etiquetas que sí los incluyen se denominan etiquetas de cierre.

En el ejemplo de más arriba, <body> es una etiqueta de apertura (es el equivalente a que le digas al navegador “oye, voy a poner contenido en body ahora”). </body> es una etiqueta de cierre, lo que significa “ok, ya terminé con el contenido en body”.

- **Nesting**

Fíjate que entre las etiquetas de apertura y cierre en HTML, además tenemos las etiquetas “head” y “body”.

**Las etiquetas pueden contener otras etiquetas.** A esto se le denomina nesting (anidamiento). Los elementos anidados **están indentados (tienen sangría)** para que el documento sea más fácil de leer.

La jerarquía que surge **del** nesting es llamada “Document Object Model” (DOM), es decir, Modelo de Objetos del Documento.

- **Comentarios**

El formato para comentar en un archivo HTML es el siguiente:

```
<p> Contenido de algún párrafo. </p>

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

## Secciones de HTML

### Video de grabación de Secciones de HTML

Veamos nuestro ejemplo anterior línea por línea:

```
<!doctype html>

<html>

  <head></head>

  <body>;Hola Mundo!</body>

</html>
```

- Tal vez hayas notado que agregamos la línea **<!doctype html>**. El propósito de esta línea es decirle al navegador que queremos que nuestro documento sea leído en modo estándar. Esto significa que estamos usando la última convención aceptada de HTML, por lo tanto, esta línea debe ser la primera en cada documento de HTML que crees.
- **<html>** indica que todo lo que esté alrededor de este y su etiqueta de cierre **</html>**, será marcada utilizando las convenciones de HTML.
- **<head>** eventualmente contendrá metadata sobre documento, las hojas de estilo externas y las librerías de JavaScript que el documento utilizará, y el título del documento.
- **<body>** incorporará el **contenido del usuario** (todo lo que esté en el documento y que será **visible en la ventana del navegador**).
- Si son necesarias separaciones adicionales, usaremos la etiqueta **<div>** que significa división.

## Relación Padre, Hijo, Hermano



En esta lección, cubriremos 2 conceptos muy importantes: “Proper indentation” (indentación correcta) y “Parent, Child, Sibling Relationship” (Relación Padre, Hijo, Hermano).

Considera el siguiente ejemplo:

### ***Bacon Ipsum, Dolor Amet?***

*Bacon ipsum dolor amet tail salami ball tip leberkas venison. Pig pork loin shoulder pork fatback corned beef chuck shank drumstick cow doner cupim capicola. Swine beef ground round, kielbasa meatball doner jowl rump chuck pastrami venison spare ribs turducken sirloin sausage. Sausage venison doner brisket, andouille pork pastrami strip steak drumstick tri-tip cupim.*

Nota cómo el encabezado (header) está centrado en la mitad de la página. Además, fíjate en que la primera palabra al comienzo del párrafo tiene algunos espacios de indentación. Como

programadores, queremos que nuestros co-programadores distingan donde termina nuestro **bloque de código** y comienza el otro. Aplicando este concepto a la programación, nuestro párrafo de “Bacon Ipsum” debiera ser traducido en HTML de la siguiente forma:

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8">

    <title>Document</title>

  </head>

  <body>

    <h3>Bacon Ipsum, Dolor Amet?</h3>

    <p> Bacon ipsum dolor amet tail salami ball tip leberkas venison.
Pig pork loin shoulder pork

    fatback corned beef chuck shank drumstick cow doner cupim capicola
. Swine before ground, kiel

    basa meatball doner jowl rump chuck pastrami venison spare ribs tu
rducken sirloin. Sausage

    venison doner brisket, andouille pork pastrami strip steak drumsti
ck tri-tip cupim.</p>

  </body>

</html>
```

Espera, ¿no debería haber estado indentado como aquí abajo?:

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8">

    <title>Document</title>

  </head>

  <body>
```

```
<h3>Bacon Ipsum, Dolor Amet?</h3>

<p> Bacon ipsum dolor amet tail salami ball tip leberkas v
enison. Pig pork

    loin shoulder pork fatback corned beef chuck shank drumstick cow d
oner cupim capicola. Swine

    beef ground round, kielbasa meatball doner jowl rump chuck pastram
i venison spare ribs turdu

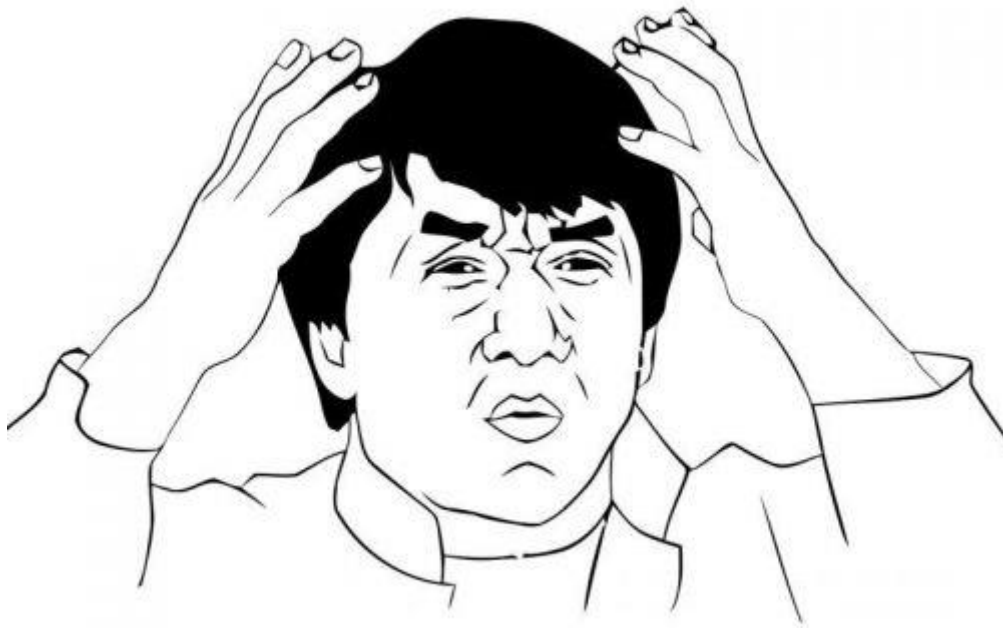
    ken sirloin sausage. Sausage venison doner brisket, andouille pork
pastrami strip steak drums

    tick tri-tip cupim.</p>

</body>

</html>
```

Respuesta simple, NO



Recuerda, no indentamos nuestro código por el contenido de nuestro sitio web, lo hacemos a través de la etiqueta HTML **relación padre, hijo, hermano (PCS)**. Pero entonces, ¿cómo obtuvimos el título centrado? ¡Buena pregunta! Esto necesitará algo de CSS y aprenderemos todo sobre posicionamiento más adelante. Por ahora, simplemente tienes que entender que

esa indentación está determinada por la relación PCS, *no* por cómo situemos el párrafo en el código HTML.

## ¿Qué es la relación PCS en HTML?

La relación PCS es solo un término para describir la relación etiquetas y elementos dentro de un documento HTML. Para conocer la relación entre etiquetas, primero debemos determinar los padres, hijos y hermanos dentro de nuestro documento HTML.

Ejemplo:

```
<!DOCTYPE html>

<html>

  <head>

    <title> ¡Hola Mundo! </title>

  </head>

  <body>

    <h1> Esta es una etiqueta de encabezado </h1>

    <p> Ahora una etiqueta de párrafo </p>

  </body>

</html>
```

De este HTML puedes leer:

- <html> es el elemento raíz
- <html> no tiene padres
- <html> es el padre de <head> y <body>
- <head> es el hijo mayor de <html>
- <body> es el hijo menor de <html>

y:

- <head> tiene un hijo: <title>
- <title> tiene un contenido (texto): "¡Hola Mundo!"
- <body> tiene dos hijos: <h1> and <p>

- `<h1>` tiene un contenido (texto): "Esta es una etiqueta de encabezado"
- `<p>` tiene un contenido (texto): "Ahora una etiqueta de párrafo"
- `<h1>` y `<p>` son hermanos
- `<head>` y `<body>` son hermanos

## ¿Es realmente necesario indentar el código HTML?

No es necesario, pero es *altamente recomendado*. He aquí las razones:

- Un código que se vea bien te puede hacer feliz, y una forma de que se vea bien es indentándolo correctamente.
- El código indentado correctamente significa menos tiempo en la resolución de problemas. Imagina que estás estancado con un problema y te enfureces con el creador de HTML (¡maldito Tim Berners-Lee!), hasta que te das cuenta que olvidaste una etiqueta de cierre ¡No pudiste verla rápidamente porque no indentaste!
- Es fácil navegar a través de tu propio código. Con solo un vistazo ya sabes que parte es el encabezado (header) y que parte es el pie de página (footer).
- Es un buen hábito de programación. Los lenguajes populares de programación como PHP, Java o Ruby recomiendan una correcta indentación del código.

### Cómo Indentar

- Simplemente presiona la tecla TAB. Un TAB es lo mismo que 4 espacios. Algunas personas prefieren 2 espacios, por lo que es posible cambiar esa opción con tu editor favorito. Eso sí, recomendamos tabs, no espacios.
- Para una demostración rápida, mira el siguiente video:

En este punto, tal vez no has apreciado la importancia de la relación PCS en HTML, pero no hay problema. Solo considera que desde el comienzo tendrás que lidiar con ella (a través de la correcta indentación del código), pero no trabajarás directamente con dicha relación (te enfocarás y manipularás elementos relacionados via CSS o jQuery).

## Práctica de Indentación

### Video de indentación de Relación padre, hijo, hermano HTML



## Configura tu editor:

Nosotros recomendamos VS Code: <https://code.visualstudio.com/download>.

Otros prefieren Atom: <https://atom.io/>

## ¡Empecemos a programar!

Por favor, corrige este código (¡está horrible!).

Crea un nuevo archivo en tu Editor de Texto (VS Code, Atom, Sublime...) e indenciona correctamente el siguiente código.

```
<!DOCTYPE html>

<html>

<head>

<title>

Basic I

</title>

</head>

<body>

<h1>

¿Qué lenguaje amas?

</h1>

<p>

¡Amo HTML!

</p>

</body>

</html>
```

Ten en cuenta que la mayoría de las etiquetas (tags) en el código a continuación aún no se han discutido. Pero, en este punto, ya sabes cómo se ven las etiquetas de apertura y cierre.

Sin conocer las etiquetas HTML, determina cuáles son los padres, hijos y hermanos indentando correctamente el siguiente código:

```
<!DOCTYPE html>

<html>

  <head>

    <title>

      Basic II

    </title>

  </head>

  <body>

    <table>

      <thead>

        <tr><th>Nombre</th><th>Apellido</th><th>Email</th><th>Password</th></tr>

      </thead>

      <tbody>

        <tr>

          <td>

            Sadie

          </td>

          <td>

            Valerio

          </td>

          <td>

            sadierocks@gmail.com

          </td><td> FakePassword123 </td></tr>

      </tbody>

    </table>
```

```
<h1>Esta es una lista de mis cosas favoritas:</h1>

  <ul><li>Comida</li>

  <li>Gatos</li>

    <li>Café</li>

    <li>Música

    </li>

  </ul>

</body>

</html>
```

## Repaso de Indentación

Este es un concepto tan importante de entender para el resto de tu vida como programador, que repasaremos juntos la última actividad.

```
<!DOCTYPE html>

<html>

  <head>

    <title>Basic I</title>

  </head>

  <body>

    <h1>¿Qué lenguaje amas?</h1>

    <p>¡Amo HTML!</p>

  </body>

</html>
```

Si analizamos nuestro HTML, nos daremos cuenta que tenemos una etiqueta de apertura html en la segunda línea que no cierra hasta el final del documento. Esto significa que cada elemento dentro de dichas etiquetas deben estar indentadas puesto que son todos elementos- hijo de la etiqueta HTML. Lo mismo se aplica a la etiqueta de título, que está anidado en el

elemento head. Asimismo, las etiquetas h1 y p son elementos-hijo de la etiqueta body, y nietas de la etiqueta html. Ahora veamos el segundo ejemplo.

```
<!DOCTYPE html>

<html>

  <head>

    <title>Basic II</title>

  </head>

  <body>

    <table>

      <thead>

        <tr>

          <th>Nombre</th>

          <th>Apellido</th>

          <th>Email</th>

          <th>Password</th>

        </tr>

      </thead>

      <tbody>

        <tr>

          <td>Sadie</td>

          <td>Valerio</td>

          <td>sadierocks@gmail.com</td>

          <td>FakePassword123</td>

        </tr>

      </tbody>

    </table>

    <h1>Esta es una lista de mis cosas favoritas:</h1>
```

```
<ul>

  <li>Comida</li>

  <li>Gatos</li>

  <li>Café</li>

  <li>Música</li>

</ul>

</body>

</html>
```

Revisa este código y aplica la relación padre, hijo, hermano para entender el contenido un poco más fácil.

¡Que disfrutes programando!

## Head

Aquí tienes un ejemplo de la sección head:

```
<head>

  <meta charset="utf-8">

  <title>Mi gran página Web</title>

  <meta name="description" content="Este texto describe de qué trata la
página web. Es lo que se mostrará en los resultados de los motores de búsq
ueda como google, bajo el título de la página web ¡Es importante que sea r
elevante para tu página y que esté bien escrito!" >

  <link rel="stylesheet" href="mi_archivo_css.css">

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

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

</head>
```

Revisemos este código línea por línea:

<head>

Este es la **etiqueta de apertura head**, la cual indica que empezaremos a hablar sobre las propiedades del documento.

## Metaetiquetas (Meta Tags)

Las metaetiquetas hacen a tu página web más relevante para los motores de búsqueda como Google.

El atributo *content* de la metaetiqueta *description* describe el propósito básico de tu página web (un resumen de lo que contiene). Para cada página web, deberías redactar un resumen conciso y relevante en esta sección.

Por ejemplo, esta descripción:

```
<title>Coding Dojo: Coding Bootcamp in Seattle and San Francisco</title>

<meta name="description" content="Coding Dojo Founder, Michael Choi, developed and refined the dynamic learning curriculum after years of hands-on experience as an executive in fast growing...">
```

Esto es lo que aparece en la página de resultados del motor de búsqueda de Google:

About 425,000 results (0.28 seconds)

**Coding Dojo: Coding Bootcamp in Seattle and San Francisco**

[www.codingdojo.com/](http://www.codingdojo.com/) ▼

**Coding Dojo** Founder, Michael Choi, developed and refined the dynamic learning curriculum after years of hands-on experience as an executive in fast growing ...

[Program](#) - [Program Options](#), [Dates](#), and ... - [Apply Now](#) - [FAQ](#)

"Coding Dojo: Coding Bootcamp in Seattle and San Francisco" viene de la etiqueta <title>.

- **<meta charset="utf-8">**

Las páginas Web adecuadamente codificadas, declaran la codificación a un navegador a través de una metaetiqueta en el encabezado. **Sin esta etiqueta**, puede que el navegador no sepa cambiar a la codificación correcta y entonces aparezcan caracteres sin sentido.

- **<title>Mi gran página Web</title>**

Este es el título de tu página web, lo que significa que cuando abres esta página en el navegador, la pestaña que se abre leerá "Mi gran página Web". Este es el nombre que se usará como marcador y que se mostrará como resultado en el motor de búsqueda.

- **<meta name="description" content="contenido de la descripción">**

La metaetiqueta de descripción es utilizada por los motores de búsqueda cuando muestran los resultados.

- `<link rel="stylesheet" href="mi_archivo_css.css">`

Esta línea enlaza una **stylesheet** a nuestra página, lo que determinará cómo nuestros elementos en HTML son **visualmente desplegados** en la página. Aprenderemos más sobre lo que debe incluirse en `my_css_file.css` en el curso de CSS.

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

Esta línea enlaza un archivo **JavaScript** o **jQuery** a nuestro documento. JavaScript hace que nuestras páginas sean **interactivas**. Aprenderemos más sobre estos archivos en el curso de jQuery.

**NOTA: Puedes enlazar todas las hojas de estilo o archivos JavaScript que quieras dentro de las etiquetas head.**

`</head>`

Esta es la **etiqueta head de cierre**. Significa que terminamos de hablar sobre las propiedades de nuestra página y podemos avanzar hacia el body.

## Etiquetas Body Comunes

### Video de etiquetas body comunes

\* Una copia del código discutido anteriormente puede ser descargado [aquí \(common-tags.html\)](#).

Es tiempo de aprender sobre las etiquetas más comunes que usarás dentro de la etiqueta **body**. Recuerda que en la lección “información general” hablamos de los elementos comunes que tal vez quisieras poner en una página web, estos son:

- Headings (títulos) y párrafos del texto
- Imágenes
- Links (enlaces)
- Listas
- Tablas
- Formularios

En esta lección cubriremos todo excepto los formularios.

## Headings (títulos)

Un **heading** es un **título de sección**, lo que significa comúnmente (¡pero no siempre!) que cada sección **<div>** tendrá un título. Existen 6 niveles de headings que puedes usar (desde **<h1>** hasta **<h6>**), donde cada uno indica la importancia de su sección.

Usemos esta página como ejemplo. En la parte de arriba puedes leer “Etiquetas Body Comunes” en letra grande y negrita. Esta es una **etiqueta <h1>**, escrita como...

```
<h1>Etiquetas Body Comunes</h1>
```

...siendo el **heading principal** de toda la página.

## Párrafos

Cualquier trozo de texto es un párrafo, por lo que necesita estar dentro de las **etiquetas <p>**.

Este es un ejemplo de cómo usar etiquetas **<p>**:

```
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse c  
ursus  
  
velit lectus, ut congue ligula molestie nec. Fusce a facilisis risus. Nul  
lam  
  
id magna semper, semper eros quis, varius velit. Duis sagittis porta enim  
ac  
  
mattis. Cum sociis natoque penatibus et magnis dis parturient montes, nas  
cet  
  
ur ridiculus mus. Donec sodales lorem id orci blandit, ac tincidunt lorem  
po  
  
rta. Sed euismod a arcu sed mollis.</p>  
  
<p>Maecenas imperdiet risus at nisl aliquet, eu ullamcorper enim imperdie  
t. Sed  
  
id metus consectetur, sollicitudin eros at, dapibus ipsum. Morbi cursus n  
ibh  
  
sit amet porta fringilla. Nam egestas nisi dui, a varius lectus egestas n  
on.  
  
Nulla facilisi. Donec sed mauris vitae sem volutpat auctor nec eu nisi. M  
aec
```



```
enas quis diam consequat, semper felis ullamcorper, mollis nisi. Morbi tu  
rpi  
  
s nulla, pellentesque sit amet erat eu, pretium sagittis mi. Fusce rhoncu  
s i  
  
mperdiet eros, ac porta ligula ullamcorper in. Suspendisse nulla urna, fa  
cil  
  
isis non nunc ut, faucibus condimentum leo.</p>
```

## Imágenes

Hay dos formas de usar imágenes en una página web: como **elementos de la página** (ej: album art en Pandora, o las fotos en tu feed de Facebook), o como **imágenes de fondo** (esto será tratado en el curso de CSS).

```

```

Por otro lado, tiene dos atributos requeridos: **src** y **alt**. El atributo **src** significa **source (fuente)** y es el link de donde proviene la imagen. El atributo **alt** significa **alternate (sustituto)** y es un texto para describir la imagen en caso de que no se cargue (aparece en el lugar de la imagen). También es utilizado por lectores de pantalla para personas con discapacidad visual.

## Links (enlaces)

Los links nos redireccionan a otras páginas. Normalmente vienen en formato de texto, pero también pueden ser imágenes.

La etiqueta utilizada para los links es **<a>**, que significa etiqueta de **anclaje (anchor tag)**. De manera similar a las imágenes, los links también necesitan un atributo (denominado **href**) que informa al navegador hacia donde apunta el link.

**Posibles valores para el atributo href son:**

- Un URL absoluto que redirige a otro sitio web (como href="http://www.example.com/default.html")

- Un URL relativo que conduce a un archivo dentro de un sitio web (como href="default.html")
- Un URL de anclaje que dirige a un anclaje dentro de una página (como href="#top")

Ejemplo:

```
<a href="https://www.google.com">Haz clic aquí para ir a Google</a>

<a href="https://www.google.com">

    <img src="">

</a>
```

## Listas

Cómo pensamos en listas en HTML es un tanto diferente a como pensamos en ella en nuestra vida diaria.

¿Qué tipo de cosas consideramos como listas normalmente?

- *Listas de compras*
- *Nombres de capítulos o temas*
- *etc..*

Entonces, ¿qué es una lista en HTML?

Es cualquier conjunto de elementos del mismo tipo. El uso más común de listas en HTML es para links de navegación.

Hay dos tipos de listas en HTML: **listas ordenadas** (aquellas que están numeradas) y **listas no ordenadas**. Las listas ordenadas usan la **etiqueta <ol>** y las no ordenadas utilizan la **etiqueta <ul>**. Ambas usan la **etiqueta <li>** para describir cada **ítem de la lista**.

```
<ul>

    <li>

        <a href="home.html">Home</a>

    </li>

    <li>

        <a href="contáctanos.html">Acerca de</a>
```

```
</li>

<li>

    <a href="contact_us.html">Contáctanos</a>

</li>

</ul>
```

## Tablas

Normalmente usaremos tablas para mostrar data. No te asustes con la palabra “data”, simplemente significa información.

Las tablas tienen muchas etiquetas asociadas puesto que están compuestas por muchas partes diferentes. Incluyen:

- Una tabla **head** (<thead>), que contiene **filas** (<tr>) y columnas de **nombre** (<th>).
- Una tabla **body** (<tbody>), que contiene **filas** (<tr>) completadas con **data** de la tabla (<td>).

Por lo que las etiquetas que necesitamos son:

**<table>, <thead>, <th>, <tbody>, <tr> y <td>**

Ejemplo:

```
<table>

    <thead>

        <tr>

            <th>Nombre</th>

            <th>Email</th>

            <th>Número Telefónico</th>

        </tr>

    </thead>

    <tbody>

        <tr>

            <td>Nombre de Ejemplo</td>
```

```
<td>un_email@gmail.com</td>

<td>555-5555</td>

</tr>

<tr>

<td>Otro Nombre</td>

<td>otro_email@gmail.com</td>

<td>444-4444</td>

</tr>

</tbody>

</table>
```

## Formularios

Los formularios son **una de las etiquetas HTML más importantes** que aprenderás. Son las responsables de **todos los intercambios de información** entre el usuario (front-end) y el servidor (back-end), por lo que es muy importante que entiendas bien cómo construirlas.

Un trabajo del formulario es **tomar el input del usuario y enviarlo al back-end para ser procesado**. Un formulario se declara utilizando la **etiqueta <form>**, que tendrá **atributos de acción y método** que deciden **dónde** y **cómo** debe enviarse la información del formulario (todavía no te preocupes de los valores que estamos usando para estos atributos, esto será cubierto a lo largo de tu primer stack).

Respecto al input, este se realiza por medio de **campos input**, normalmente designados con la etiqueta **<input>**. Dependiendo del tipo de información requerida, la forma en que tomamos la información puede ser diferente. Esto a veces es designado como un **atributo type**, y en otras ocasiones se utiliza una etiqueta distinta. Comúnmente, cada input también incluirá un *label* (**<label>**), es decir, el nombre del campo. Para asegurarse que un label específico está enlazado/asociado a un elemento input específico, debemos agregar un **atributo for** en el label con el atributo de input denominado **id**. Incluir una etiqueta label alrededor del campo de

input es una convención que nos permite clicar en el label para enfocarnos en el campo del input.

Un **atributo name**, también irá mano a mano con tus etiquetas de input. Son utilizadas principalmente para enviar información del formulario hacia el back-end, pero no te preocupes por ahora del atributo name. Será explicado más adelante en tu primer stack.

Veamos qué tipos de input podrían ser utilizados en las siguientes circunstancias:

- **Cuando los usuarios necesitan incorporar un texto corto, como una dirección de email o un nombre.**

El tipo de input apropiado acá es **text**.

```
<label for="nombre">Nombre:</label>

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

<label for="apellido">Apellido:</label>

<input type="text" id="apellido" name="apellido">

<label for="email">Email:</label>

<input type="text" id="email" name="email">
```

- **Un campo de contraseña (password)**

El tipo de input aquí es **password**.

```
<label for="password">Contraseña</label>

<input type="password" id="password" name="password">
```

- **Cuando el usuario puede elegir una de varias opciones. Un buen ejemplo es un selector género.**

El input apropiado acá serían los botones **radio**.

```
<label for="masculino">Masculino</label>

<input type="radio" id="masculino" name="genero" value="masculino">

<label for="femenino">Femenino</label>

<input type="radio" id="femenino" name="genero" value="femenino">

<label for="prefiero-no-decir">Prefiero no decir</label>
```

```
<input type="radio" id="prefiero-no-decir" name="genero" value="prefiero-no-decir">
```

Otra opción es un menú desplegable que usa las etiquetas `<select>` y `<option>`.

```
<select name="genero">

  <option value="masculino">Masculino</option>

  <option value="femenino">Femenino</option>

  <option value="prefiero-no-decir">Prefiero no decir</option>

</select>
```

- Cuando los usuarios pueden escoger entre múltiples opciones, como sus 3 colores favoritos entre 5 alternativas.

El input apropiado acá es **checkboxes**.

```
<label for="azul">Azul</label>

<input type="checkbox" id="azul" name="color" value="azul">

<label for="verde">Verde</label>

<input type="checkbox" id="verde" name="color" value="verde">

<label for="rojo">Rojo</label>

<input type="checkbox" id="rojo" name="color" value="rojo">

<label for="negro">Negro</label>

<input type="checkbox" id="negro" name="color" value="negro">

<label for="morado">Morado</label>

<input type="checkbox" id="morado" name="color" value="morado">
```

- Cuando el usuario quiere ingresar un texto más largo. Puede ser utilizado para comentar en foros o para descripciones de perfil.

En este caso usaremos la etiqueta `<textarea>`.

```
<textarea name="description"></textarea>
```

- Cuando un formulario debe enviar el input de más de un usuario

El tipo de input **hidden** es similar a los campos de texto, con la diferencia de que no se muestra en la página y los usuarios no pueden agregar nada en ellos. Esto es útil para la autenticación back-end y pasar datos.

```
<input type="hidden" name="id" value="7">
```

- Finalmente, para crear un botón de enviar (submit)

El input apropiado acá es submit.

```
<input type="submit" value="Enviar">
```

Veamos un ejemplo de formulario de registro completo:

```
<form action="process.php" method="post">

  <p>Por favor regístrese</p>

  <label for="nombre">Nombre:</label>

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

  <label for="apellido">Apellido:</label>

  <input type="text" id="apellido" name="apellido">

  <label for="email">Email:</label>

  <input type="text" id="email" name="email">

  <p>Selecciona tu género:</p>

  <label for="masculino">Masculino</label>

  <input type="radio" id="masculino" name="genero" value="masculino">

  <label for="femenino">Female</label>

  <input type="radio" id="femenino" name="genero" value="femenino">
```

```
<label for="prefiero-no-decir">Prefiero no decir</label>

<input type="radio" id="prefiero-no-decir" name="genero" value="prefie
ro-no-decir">

<p>Selecciona 3 de tus colores favoritos:</p>

<label for="azul">Azul</label>

<input type="checkbox" id="azul" name="color" value="azul">

<label for="verde">Verde</label>

<input type="checkbox" id="verde" name="color" value="verde">

<label for="rojo">Rojo</label>

<input type="checkbox" id="rojo" name="color" value="rojo">

<label for="negro">Negro</label>

<input type="checkbox" id="negro" name="color" value="negro">

<label for="morado">Morado</label>

<input type="checkbox" id="morado" name="color" value="morado">

<p>Escribe unas palabras sobre ti mismo:</p>

<textarea name="descripcion"></textarea>

<label for="password">Contraseña:</label>

<input type="password" id="password" name="password">

<label for="pw_confirm">Confirmación de Contraseña:</label>

<input type="password" id="pw_confirm" name="password_confirmacion">
```



```
<input type="submit" value="Presiona aquí para registrarte">

</form>
```

## Otra declaración *label-input*

La mayoría de los frameworks de CSS (especialmente Twitter Bootstrap), usan la relación label-input mostrada anteriormente, pero tal vez encontrarás un formato diferente en cómo el conjunto label-input es declarado. Aquí un ejemplo:

```
<form>

  <p>Por favor Regístrese</p>

  <label>

    Nombre:<input type="text" name="nombre">

  </label>

  <p>Seleccione su género:</p>

  <label>

    Masculino<input type="radio" name="genero" value="masculino">

  </label>

  <label>

    Femenino<input type="radio" name="genero" value="femenino">

  </label>

  <label>

    Prefiero no decirlo<input type="radio" name="genero" value="prefiero-no-decir">

  </label>

  <p>Por seguridad, ingrese su contraseña.</p>

  <label>

    Contraseña:<input type="password" name="password">

  </label>

  <input type="submit" value="Presiona aquí para registrarte">
```

```
</form>
```

Fíjate cómo el elemento ***input*** ahora está anidado dentro del elemento label y, por ende, ya no necesitamos enlazar ambas utilizando el atributo for de ***label*** y el atributo id de input.

## Formulario de Registro

Ahora que hemos aprendido todos los diferentes tipos de etiquetas de formulario, creemos un formulario de registro utilizando solo HTML. Cuando te registras en cualquier sitio web, ya sea una cuenta de correo, una red social o incluso este bootcamp, lo que estás haciendo es enviar un formulario. Usarás muchísimos formulario para enviar datos a tu backend en los siguientes meses, por lo que sintámonos cómodos creando formularios.

### Mi Formulario de Registro


Nombre:

Apellido:

Email:


Contraseña:

Confirmar:

Cumpleaños:   

☐ Hombre ☐ Mujer ☐ Otro

Una breve descripción de ti:

Lenguaje Favorito  

Recrea todos los elementos de HTML que aparecen en la imagen. Como aún no has aprendido a usar css para que el formulario se vea bien, por ahora no te preocupes por su apariencia.

Tu HTML debería contener las siguientes etiquetas:

- `<input>` con estos valores en el atributo "type":
  - text
  - password
  - date
  - radio
  - submit
- `<textarea>`
- `<select>`

Utilizarás formularios en casi todas las actividades del bootcamp, por lo que no trates de memorizar hoy todas sus etiquetas y atributos. Siempre puedes volver a esta sección como referencia. En la siguiente actividad, crearemos un blog falso utilizando tanto los formularios como todas las otras etiquetas HTML que aprendimos hoy.

**NOTA:** Recuerda validar tu código antes de enviarlo. Los servicios de validación de HTML como [W3C Markup Validation](https://validator.w3.org/) (gratis) son depuradores útiles que te ayudan a identificar errores de procesamiento.

## Consejos Importantes para Evitar Dolores de Cabezas

HTML permite que un código deficiente se ejecute y procese con diferentes niveles de precisión. Sin embargo, un procesamiento exitoso no significa que nuestro código sea correcto o garantiza que se validará de acuerdo a los estándares. El código deficiente es impredecible, y no puedes tener certeza qué obtendrás cuando se procese. De esta manera, tenemos que poner bastante atención cuando escribimos en HTML, asegurándonos de anidar/indentar y cerrar todos los elementos correctamente y siempre validar nuestro código.

### Usa una Estructura de Documentos Apropiaada

Es posible procesar las páginas sin utilizar el doctype `<!DOCTYPE html>` o los elementos `<html>`, `<head>` y `<body>`. Sin embargo, sin el doctype y dichos elementos estructurales, no se procesarán correctamente en todos los navegadores.

#### Código incorrecto

```
<html>

  <h1> ¡Hola Mundo! </h1>

  <p> Este es mi primer sitio web </p>
```

```
</html>
```

## Código correcto

```
<!DOCTYPE html>

<html>

  <head>

    <title> Mi primer sitio web </title>

  </head>

  <body>

    <h1> ¡Hola Mundo! </h1>

    <p> Este es mi primer sitio web </p>

  </body>

</html>
```

## ¡Valida tu Código Constantemente!

Mientras escribes en HTML, haz de validar frecuentemente un hábito. Esto te evitará problemas que son difíciles de localizar (o rehacer) cuando tu trabajo está completo.

Los servicios de validación de HTML como la opción gratuita [W3C Markup Validation Service](#), son depuradores que te ayudan a identificar errores de procesamiento.

## Organiza la sintaxis de HTML

A medida que tu HTML se hace más grande, gestionarlo puede ser una tarea complicada. Más abajo verás reglas rápidas que pueden ayudarte a mantener limpia tu sintaxis limpia y organizada:

- Usa minúsculas para los elementos de nombre, atributos y valores.
- Indenta los elementos anidados.
- Utiliza doble comillas (no simples o inexistentes) para el contenido de los atributos en HTML. Un buen ejemplo sería:

```
<h1 id="titulo_de_pagina">¡Mi Primer Sitio Web!</h1>
```

```
<p class="subtitulo">¡Esto es <span class="enfaticar">Genial!</span></p>
```

## ¡Evita usar demasiados *divs*!

Cuando programas en HTML, es fácil dejarse llevar agregando elementos `<div>` aquí y allá para construir los estilos necesarios. Aunque esto funciona, puede dejar muy sobrecargada una página, y al poco tiempo no estaremos seguros qué hace cada elemento `<div>`.

### Código Incorrecto

```
<div class="contenedor">

  <div class="articulo">

    <div class="titular"> HTML Rocks! </div>

  </div>

</div>
```

### Código Correcto

```
<div class="contenedor">

  <article>

    <h1>HTML Rocks!</h1>

  </article>

</div>
```

## Utiliza los Elementos Semánticos

Decidir qué elementos utilizar para describir distintos contenidos puede ser complejo, pero estos elementos son la columna vertebral de la semántica.

*Nota: La semántica de HTML es el uso del marcado de HTML para reforzar la semántica (o el **significado**) de la información en los sitios web en lugar de, simplemente, definir su presentación o aspecto.*

ver: [en.wikipedia.org/wiki/Semantic\\_HTML](https://en.wikipedia.org/wiki/Semantic_HTML)

Aquí el HTML no utiliza apropiadamente los elementos heading y paragraph. En cambio, usa elementos sin sentido para agrupar contenido.

## Código Incorrecto

```
<span class="titular"><strong>Bienvenido Devuelta</span></strong>

<br><br>

Tanto tiempo ¿Qué ha sido de ti?

<br><br><br>
```

## Código Correcto

```
<h1>Bienvenido Devuelta</h1>

<p> Tanto tiempo ¿Qué ha sido de ti?</p>
```

## Utiliza Minúsculas para los Nombres de las Etiquetas

Técnicamente, puede usarse mayúsculas.

```
<DIV>

  <P> ¡Ya no me quedan temas! </P>

</DIV>
```

Habiendo dicho eso, la mejor práctica es mantener todas las etiquetas en minúscula.

## Usa el Atributo *alt* con Imágenes

La utilización de los atributos *alt* con elementos `<img>` es imprescindible para escribir código semántico y válido. La información en alt es una ayuda cuando un usuario no puede ver tu imagen, ya sea por un problema de conexión, una imagen perdida o porque el usuario está utilizando un lector de pantalla.

## Código Incorrecto

```

```

## Código Correcto

```

```

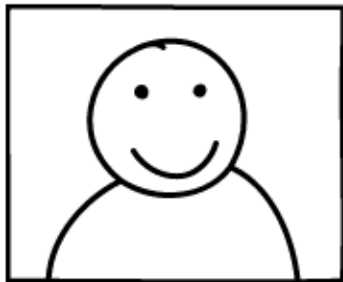
## Fake Blog

Crea un blog ficticio utilizando solo HTML ¡Puede ser sobre lo que quieras! Tus hobbies, tu equipo o jugador favorito, tu programa de televisión preferido, etc. Tu blog puede verse como quieras, pero tu documento debe utilizar al menos las siguientes etiquetas:

- HTML
- head, title, meta description,
- body, div, a, p, h1, ul, li, table (incluyendo thead, td, tbody, tr)  
Al final de la página, crea un formulario ficticio de encuesta que utilice las siguientes etiquetas:
  - - form, select, input type="text", input type="checkbox", input type="radio",  
input type="submit".

Puedes utilizar también los [generadores Lorem Ipsum](#) para texto ficticio. Acá un ejemplo de cómo podría verse tu blog.

# Michael Jordan



Michael Jordan is a professional basketball player who has won six NBA championships with the Chicago Bulls. He is widely regarded as one of the greatest players in the history of the sport. Jordan played for the Bulls from 1984 to 1998 and for the Washington Wizards from 2001 to 2003.

Año	PPG	MPV
1990-91	31.5	Sí
1991-92	30.1	Sí
1992-93	32.6	No
1995-96	30.4	Sí
1996-97	29.6	No
1997-98	28.7	Sí

- 1984 - 1983 Chicago Bulls
- 1995 - 1998 Chicago Bulls
- 2001 - 2003 Washington Wizards

[Haz clic acá por un increíble video destacado](#)

Por favor, contesta esta breve encuesta

Email:

- ☐ Sí, por favor envíame montones de correos
- ☐ No por favor, no me mandes ningún correo

¿Cómo supiste de nosotros?

- ☐ Facebook
- ☐ El periódico
- ☐ Un contacto



Durante el resto del bootcamp, utilizarás mucho estas etiquetas HTML. No te sientas presionado a memorizarlas y entender cómo funcionan exactamente, puesto que siempre puedes volver a esta sección y, además, cuando aprendas más sobre CSS y otros temas del currículum, te sentirás más cómodo con HTML. Cuando hayas terminado las actividades básicas, avanza a la siguiente sección y obtendrás un mejor entendimiento de estas etiquetas HTML que acabas de aprender.

**NOTA: Recuerda validar tu código antes de enviarlo.**

## Etiquetas Adicionales

Hay montones de etiquetas en HTML. Aquí te presentamos algunas etiquetas adicionales que tal vez te ayuden durante tu carrera de desarrollador.

### DOCTYPE

Todo este tiempo, hemos estado utilizando muchos aspectos de HTML5. Por ejemplo, el doctype para HTML4.01 se ve así:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
```

Esto no es muy comprensible. En términos simples está diciendo “este documento está escrito en HTML 4.01”. En cambio, el doctype para HTML5 se ve así:

```
<!DOCTYPE html>
```

Quiere decir que “este documento está escrito en HTML5”. Fíjate que en esta nueva declaración no estamos especificando la versión de HTML. Esto es porque HTML5 admite contenido existente (el doctype puede ser aplicado a un documento existente en HTML4.01). Cualquier versión futura de HTML, también admitirá el contenido existente en HTML5, por lo que esta forma de declarar tu doctype es el camino a seguir.

*Navegadores compatibles: Todas las versiones de Chrome, IE, Firefox, Safari, Opera.*

### Codificación de Caracteres

En HTML 4.01 tenías que especificar la codificación de caracteres de tu documento HTML con la siguiente metadecларación:

```
<meta http-equiv="content-type" content="text/html; charset=UTF8">
```

En HTML5:

```
<meta charset="utf-8">
```

*Navegadores compatibles: Todas las versiones de Chrome, IE, Firefox, Safari, Opera.*

### Etiquetas Script y Link

Las etiquetas `<script>` y `<link>` son otro caso donde HTML5 nos permite escribir menos. Antes, tenías que especificar un `type` denominado “text/javascript” para hacer un script de elementos o un “text/css” si usabas elementos de link para referencias tus stylesheets.

```
<script type="text/javascript" src="main.js"></script>
<link rel="stylesheet" type="text/css" href="style.css">
```

Con HTML5 puedes escribir:

```
<script src="main.js"></script>
<link rel="stylesheet" href="style.css">
```

*Navegadores compatibles: Todas las versiones de Chrome, IE, Firefox, Safari, Opera.*

---

## Rich Media

A menudo tenemos que usar alguna tecnología plug-in como flash o Silverlight para publicar audio o video en la web. Dichas tecnologías tapan (“plug-in”) los hoyos de la web. Sin embargo, estas tecnologías no son abierta y, por ende, no son creadas por la comunidad, sino que están bajo el control de empresas individuales. Aquí HTML5 sale al rescate compitiendo

directamente con tecnologías como Flash y Silverlight, pero en vez de requerir un plug-in, los elementos de rich media en HTML5 son nativos para navegador.

## Canvas

Una vez que una imagen ha sido insertada en un navegador, su contenido no puede ser actualizado. Puedes crear gifs animados o actualizar los estilos de imagen, pero su contenido no puede ser actualizado. El elemento canvas puede ser utilizado para crear imágenes dinámicas.

```
<canvas id="obra_de_arte" width="400" height="200">
  <p>Tu navegador no es compatible con canvas</p>
</canvas>
```

Puedes aprender cómo interactuar con canvas [aquí](#). Canvas ofrece muchas de las herramientas que puedes encontrar en un programa gráfico como Illustrator: tipo de trazo, rellenos, degradados, sombras, formas, y curvas de Bézier. La diferencia es que debes especificar todo utilizando JavaScript en vez de usar una interfaz gráfica de usuario (GUI).

*Navegadores Compatibles: Chrome 4.0+, IE 9.0+, Firefox 2.0+, Safari 9.0+, Opera 9.0+*

## Audio

Insertar un archivo de audio en un documento HTML5 es simple:

```
<audio src="desperado.mp3">
</audio>
```

Tal vez molestes a tus visitantes de esta forma, pero puedes incluso crear un atributo de autoplay.

```
<audio src="desperado.mp3" autoplay>
</audio>
```

Y puedes molestarlos aún más con un loop...

```
<audio src="desperado.mp3" autoplay loop>

</audio>
```

O puedes darle el control solicitando al navegador controles nativos para dar play o pausa al audio, además de poder ajustar el volumen.

```
<audio src="desperado.mp3" controls>

</audio>
```

El elemento audio parece demasiado bueno para ser verdad, pero hay una trampa. El problema con este no está en la especificación, está en los formatos de audio. El formato MP3 es ampliamente usado en todas partes, pero no es un formato abierto. Esto significa que los navegadores no puede decodificar los archivos MP3 sin pagar una tarifa. La tarifa no es un gran problema para empresas grandes como Apple, pero sí lo es para empresas más pequeñas como Mozilla. De esta manera, Safari reproducirá los archivos MP3, mientras que Firefox no lo hará.

El codec Vorbis, que generalmente viene como un archivo .ogg, es un formato abierto. Firefox es compatible con Ogg Vorbis, pero Safari no. Chrome es compatible con ambos formatos.

Afortunadamente, hay una forma de usar el elemento audio para ser compatible con todos los navegadores que aceptan elementos de audio.

```
<audio controls>

  <source src="desperado.ogg">

  <source src="desperado.mp3">

</audio>
```

Un navegador que puede reproducir archivos Ogg Vorbis no buscará más allá del primer elemento fuente. Un navegador que puede reproducir archivos MP3, se saltará el primer elemento fuente y reproducirá el archivo en el segundo elemento fuente. Puedes aprender más del la etiqueta de audio [acá](#).

*Navegadores Compatibles: Chrome 4.0+, IE 9.0+, FireFox 3.5+, Safari 4.0+, Opera 10.5+*

## Video

El plugin de Flash es uno de las formas más populares de reproducir contenido de video en la Web. HTML5 podría cambiar esto. El elemento video funciona muy similar al elemento audio. La mayor diferencia es que tal vez querrás agregar dimensiones a tu elemento de video.

```
<video src="caddyshack.mp4" width="400" height="800" controls>
</video>
```

Una vez más, existe una batalla entre los formatos de video. Algunos de los más grandes son MP4 (patentado) y Theora Video (licencia gratuita). Safari no es compatible con archivos .ogv y FireFox no es compatible archivos .mp4. Chrome es compatible con ambos. Afortunadamente, puedes usar el elemento video para ser compatible con todos los navegadores que aceptan el elemento de video.

```
<video width="400" height="200" poster="picture.jpg" controls>
  <source src="caddyshack.ogv">
  <source src="movie.mp4">
</video>
```

Una de las limitaciones de confiar en un plug-in para rich media, es que el contenido del plug-in está apartado del resto de la página web. Tener elementos nativos de rich media en HTML, significa que podemos hacer que nuestro contenido multimedia se reproduzca bien con otras tecnologías de navegador como CSS y JavaScript. Puedes aprender más sobre el elemento video [here](#).

*Browser Support: Chrome 4.0+, IE 9.0+, FireFox 3.5+, Safari 4.0+, Opera 10.5+*

---

## Elementos Semánticos

HTML nos entrega elementos que nos dicen exáctamente con qué estamos trabajando, tales como paragraphs, lists y headlines. Asimismo, proporciona elementos más genéricos como

<div> y <span>. Por otro parte, además de las etiquetas básicas headline, paragraph y lists, podemos usar etiquetas más descriptivas como <header>, <footer> y <nav>. Puedes aprender más sobre los elementos semánticos [acá](#).

## FAQ

- **¿Por qué no vimos las nuevas etiquetas de HTML5 como section, header, footer, main, etc?**

HTML5 tiene más etiquetas nuevas para ayudar a los desarrolladores a estructurar mejor los contenidos del sitio web. Para conocer la lista completa entra a: [http://www.w3schools.com/html/html5\\_new\\_elements.asp](http://www.w3schools.com/html/html5_new_elements.asp). Aunque estas etiquetas pueden ser útiles, los navegadores antiguos todavía no son totalmente compatibles con ellas, por eso decidimos focalizarnos en las etiquetas HTML originales/antiguas por ahora. Cuando seas un experto en estas etiquetas, será mucho más fácil comenzar a usar las nuevas etiquetas de HTML5. Obviamente ya sabes que los navegadores antiguos tal vez no puedan acceder a las nuevas etiquetas HTML y tu sitio web no pueda ser visualizado.

- **¿Cómo puedo saber si mi HTML es válido?**

Puedes (y deberías) usar <http://validator.w3.org/> para verificar tu HTML.

- **¿Cuál es la diferencia entre un atributo id y un atributo name?**

El **atributo id** identifica tu elemento para el **front-end** (CSS y JavaScript). Puede ser utilizada en **CUALQUIER elemento**.

El **atributo name** pertenece **solo a los elementos de un formulario** y es utilizado en el back-end (PHP, Ruby on Rails, etc) los valores de tu formulario.

- **¿Qué es el DOM?**

El Document Object Model (DOM) o Modelo de Objetos del Documento, es una representación estructurada de tu HTML generado por el navegador, permitiendo el acceso a los elementos de tu página web para que pueden ser manipulados. Generalmente, es un JavaScript el que hace

dicha manipulación. El DOM es muy difícil de definir, por lo que usaremos la siguiente metáfora:

Digamos que tenemos un pedazo de papel que dice “¡Coding Dojo es lo máximo!”. Entonces, tomamos unas letras magnéticas, vamos a nuestro refrigerador y escribimos “¡Coding Dojo es lo máximo! de acuerdo a lo que aparece en el papel. **Ese papel es tu HTML y las letras magnéticas pueden ser consideradas nuestro DOM.**

En muchos casos, las letras magnéticas serán lo mismo que lo está escrito en nuestro papel, por lo que no habrá diferencia entre HTML y el DOM. En otros casos, queremos cambiar las palabras en el refrigerador, o cambiar el color de las letras magnéticas o agregar otra funcionalidad. **Dichos cambios significan manipular el DOM, entonces nuestro papel en HTML con el que comenzamos no tendrá cambios, pero el DOM ya no será idéntico.**

Cuando veas JavaScript y sus cambios al DOM o escuches hablar sobre manipulación del DOM, básicamente estarán hablando de **interactuar con esos magnetos de refrigerador luego de haber copiado nuestro papel en HTML.**