# Manual de HTML5 en español

Alejandro Castillo Cantón

www.theproc.es

# Primera Parte

El HTML5 (HyperText Markup Language, versión 5) es la quinta revisión del lenguaje de programación "básico" de la World Wide Web, el HTML. Esta nueva versión pretende remplazar al actual (X)HTML, corrigiendo problemas con los que los desarrolladores web se encuentran, así como rediseñar el código actualizandolo a nuevas necesidades que demanda la web de hoy en día.

Debido a que estos cambios afectaran la forma de desarrollar la web en un futuro inmediato, desde The Process, plantearemos una serie de artículos donde desvelaremos los cambios más importantes.

Actualmente el HTML5 está en un estado BETA, aunque ya algunas empresas están desarrollando sus sitios webs en esta versión del lenguaje. A diferencia de otras versiones de HTML, los cambios en HTML5 comienzan añadiendo semántica y accesibilidad implícitas, especificando cada detalle y borrando cualquier ambigüedad. Se tiene en cuenta el dinamismo de muchos sitios webs (facebook, twenti, etc), donde su aspecto y funcionalidad son más semejantes a aplicaciones webs que a documentos.

## Mejor estructura

Actualmente es abusivo el uso de elementos DIV para estructurar una web en bloques. El HTML5 nos brinda varios elementos que perfeccionan esta estructuración estableciendo qué es cada sección, eliminando así DIV innecesarios. Este cambio en la semántica hace que la estructura de la web sea más coherente y fácil de entender por otras personas y los navegadores podrán darle más importancia a según qué secciones de la web facilitándole además la tarea a los buscadores, así como cualquier otra aplicación que interprete sitios web. Las webs se dividirán en los siguientes elementos:

- **<section></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 mu favorable para el buen posicionamiento web.

- **<article></article>** - El elemento de artículo 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. Podría utilizarse en los artículos de los foros, una revista o el artículo de periódico, una entrada de un blog, un comentario escrito por un usuario, un widget interactivo o gadget, o cualquier otro artículo independiente de contenido.

  Cuando los elementos de <article> son anidados, los elementos de <article> interiores representan los artículos que en principio son relacionados con el contenido del artículo externo. Por ejemplo, un artículo de un blog que permite comentarios de usuario, dichos comentarios se podrían representar con <article>.

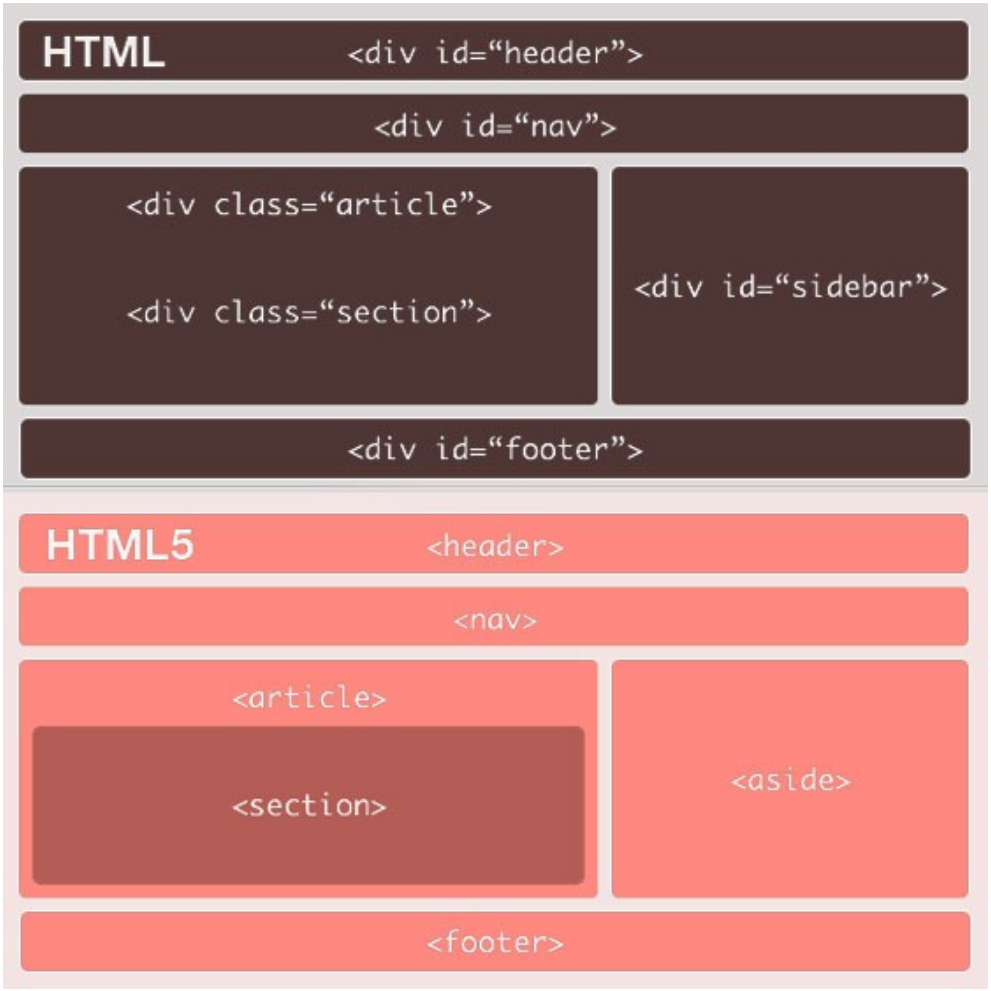
- **<aside></aside>** - Representa una sección de la página que abarca un contenido tangencialmente relacionado con el contenido que lo rodea, por lo que se le puede considerar un contenido independiente. Este elemento puede utilizarse para efectos tipográficos, barras laterales, elementos publicitarios, para grupos de elementos de la navegación, u otro contenido que se considere separado del contenido principal de la página.

- **<header></header>** - Elemento <header> representa un grupo de artículos introductorios o de navegación.

- **<nav></nav>** - El elemento <nav> representa una sección de una página que es un link a otras páginas o a partes dentro de la página: una sección con links de navegación.

  No todos los grupos de enlaces en una página tienen que estar en un elemento <nav>, sólo las secciones que consisten en bloques principales de la navegación son apropiadas para ser utilizadas con el elemento <nav>. Puede utilizarse particularmente en el pie de página para tener un menú con un listado de enlaces a varias páginas de un sitio, como el Copyright; home page, política de uso y privacidad. No obstante, el elemento <footer> es plenamente suficiente sin necesidad de tener un elemento <nav>.

- **<footer></footer>** - El elemento <footer> representa el pié 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.




*Diferencias entre HTML y HTML5*

## Mejoras en los formularios

El elemento input adquiere gran relevancia al ampliarse los elementos que se permitiran en el "type".

- <input type="search"> para cajas de búsqueda.
- <input type="number"> para adicionar o restar números mediante botones.
- <input type="range"> para seleccionar un valor entre dos valores predeterminados.
- <input type="color"> seleccionar un color.
- <input type="tel"> números telefónicos.
- <input type="url"> direcciones web.
- <input type="email"> direcciones de email.

- <input type="date"> para seleccionar un día en un calendario.
- <input type="month"> para meses.
- <input type="week"> para semanas.
- <input type="time"> para fechas.
- <input type="datetime"> para una fecha exacta, absoluta y tiempo.
- <input type="datetime-local"> para fechas locales y frecuencia.

## Otros elementos muy interesantes

**<audio> y <video>** - Nuevos elementos que permitirán incrustar un contenido multimedia de sonido o de vídeo, respectivamente. Es una de las novedades más importantes e interesantes en este HTML5, ya que permite reproducir y controlas vídeos y audio sin necesidad de plugins como el de Flash.

El comportamiento de estos elementos multimedia será como el de cualquier elemento nativo, y permitirá insertar en un video, enlaces o imágenes, por ejemplo. Youtube, ya ha anunciado que deja el Flash y comienza a proyectar con HTML5.

**<embed>** - Se emplea para contenido incrustado que necesita plugins como el Flash. Es un elemento que ya reconocen los navegadores, pero ahora al formar parte de un estándar, no habrá conflicto con <object>.

**<canvas>** - Este es un elemento complejo que permite que se generen gráficos al hacer dibujos en su interior. Es utilizado en Google Maps y en un futuro permitirá a los desarrolladores crear aplicaciones muy interesantes.

# Segunda Parte

Una pregunta muy común en estos tiempos es: "¿Cómo puedo empezar a utilizar HTML5 si existen navegadores antiguos que no lo soportan?" Pero la pregunta en sí se ha formulado de forma errónea. El HTML5 no es una cosa grande como un todo, sino una colección de elementos individuales, por consiguiente lo que sí se podrá será detectar si los navegadores soportan cada elemento por separado.

Cuando los navegadores realizan un render de una página, construyen un "**Modelo de Objeto de Documento**" (Document Object Model - DOM), una colección de objetos que representan los elementos del HTML en la página. Cada elemento - ***<p>***, *** ***, ****** - es representado en el **DOM** por un objeto diferente.

Todos los objetos **DOM** comparten unas características comunes, aunque algunos tienen más que otros. En los navegadores que soportan rasgos del HTML5, algunos objetos tienen una única propiedad y con una simple ojeada al **DOM** podremos saber las características que soporta el navegador.

Existen cuatro técnicas básicas para saber cuando un navegador soporta una de estas particulares características, desde las más sencillas a las más complejas.

1. Comprueba si determinadas propiedades existen en objetos genéricos o globales (como ***window*** o ***navigator***).
   Ejemplo: comprobar soporte para la "Geolocalización".

2. Crear un elemento, luego comprobar si determinadas propiedades existen en ese elemento.
   Ejemplo: comprobar soporte para ***canvas***.

3. Crear un elemento, comprobar si determinados métodos existen en ese elemento, llamar el método y comprobar los valores que devuelve.
   Ejemplo: comprobar qué formatos de video soporta.

4. Crear un elemento, asignar una propiedad a determinado valor, entonces comprobar si la propiedad mantiene su valor.
   Ejemplo: comprobar que tipo de ***<input>*** soporta.

## MODERNIZR, una biblioteca para detectar HTML5.

**Modernizr** es una librería de *JavaScript* con licencia MIT de código abierto que detecta si son compatibles muchos elementos para HTML5 y CSS3. Dicha librería se irá actualizando y para utilizarla solo hay que incluir en ***<head><script>*** de tu página.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Dive Into HTML5</title>
  <script src="modernizr.min.js"></script>
</head>
<body>
  ...
</body>
</html>
```

**Modernizr** se ejecuta automáticamente, no es necesario llamar a ninguna función tipo: **modernizr_init()**. Cuando se ejecuta, se crea una objeto global llamado Modernizr, que contiene un set de propiedades Boleanas para cada elemento que detecta. Por ejemplo si su navegador soporta elementos canvas, la propiedad de la librería **Modernizr.canvas** será ***"true"***. Si tu navegador no soporta los elementos canvas, la propiedad **Modernizr.canvas** será ***"false"***.

```
if (Modernizr.canvas) {
  // a crear formas!!
} else {
  // no hay soporte para canvas, los siento
```

## Canvas

HTML 5 define el elemento <canvas> como un rectángulo en la página donde se puede utilizar Java Script para dibujar cualquier cosa. También determina un grupo de funciones (canvas API) para dibujar formas, crear gradientes y aplicar transformaciones.

## Texto Canvas

Si tu navegador soporta las API de canvas no quiere decir que pueda soportar las API para texto-canva. Las API de canva se han ido generando con el tiempo y las funciones de texto se han añadido posteriormente,

por lo que algunos navegadores puede que no tengan integrado las API para texto.

## Video

El HTML5 ha definido un nuevo elemento llamado <video> para incrustar video en las páginas de la web. Actualmente insertar un video en la web era imposible sin determinados plugins como el QuickTime o el Flash.

El elemento <video> ha sido diseñado para utilizarlo sin la necesidad de que tenga que detectar ningún script. Se pueden especificar múltiples ficheros de video y los navegadores que soporten el video en HTML5 escogerán uno basado en el formato que soporte.

## Formatos de video

Los formatos de video son como los lenguajes escritos. Un periódico en inglés contiene la misma información que un periódico en español, aunque solo uno le será útil. Con los navegadores pasa lo mismo, necesitan saber en qué "idioma" está escrito el video.

Los lenguajes de los videos se llaman "codecs" un algoritmo utilizado para compactar un video. Existen docenas de codecs en uso en todo el mundo, aunque dos son los más relevantes. Uno de estos codecs cuesta dinero por la licencia de la patente, y funciona en safari y los iphones. El otro codec es gratis y de código abierto y funciona en navegadores como Chromium y Firefox.

## Aplicaciones OFFline

Leer página webs offline es relativamente sencillo. Te conectas a Internet, cargas una web, te desconectas y puedes sentarte tranquilamente a leer. ¿Pero qué sucede cuando son aplicaciones como Gmail o Google Docs? Gracias al HTML5 cualquiera puede crear una aplicación web que funcione offline.

Las aplicaciones web offline se ejecutan como una aplicación online. La primera vez que se visita una web offline que esté disponible, el servidor web le dirá a al navegador los ficheros que necesita para poder trabajar desconectado. Estos ficheros pueden ser, HTML, JavaScript, imágenes y hasta videos. Una vez que el navegador ha descargado los ficheros necesarios podrás volver a visitar la web aunque no estés conectado a Internet. El navegador reconocerá que estás desconectado de Internet y utilizará los ficheros que había descargado con anterioridad. La próxima

vez que te conectes, si has realizado cambios en la web offline, estos se subirán al servidor actualizándolo.

### Geolocalización

La geolocalización es la forma de suponer donde te encuentras en el mundo y si quieres, compartir información con gente de confianza. Existen muchas maneras de descubrir donde te encuentras, por tu dirección IP, la conexión de red inalámbrica, la torre de telefonía móvil por la que habla tu teléfono móvil (celular), o GPS específicos que reciben las coordenadas de longitud y la latitud de satélites que están ene el cielo.

## Tercera Parte

En el primer capítulo de esta charla sobre HTML5, mencionamos por arriba sobre los cambios en los formularios que incluye el HTML5.

Como norma para todos, un formulario es una etiqueta <form> y en su interior algunos elementos <input type="text"> o <input type="password"> finalizado con un botón <input type="submit"> y ya está. A continuación explicaremos algunas de estas novedades.

## Texto como PLACEHOLDER

***Placeholder*** es un nuevo atributo que se utiliza dentro de los campos input. Sirve para mostrar un texto dentro del input siempre y cuando el campo esté vacío o no esté señalado. En cuanto se haga clic dentro del campo (o se llegue por el TAB), el texto desaparecerá.

Seguramente ha visto la propiedad ***Placeholder*** antes. Por ejemplo, Mozilla Firefox 3.5 incluye textos ***placeholder*** en la barra de localización.


Cuando se hace clic sobre la barra de búsqueda o se llega por un tab, el texto preestablecido desaparece.

Irónicamente Firefox no da soporte a esta propiedad, al igual que IE y Opera, solo es compatible (a día de hoy) con Safari y Chrome. Aquellos

navegadores que no soporten ***placeholder*** simplemente lo ignorarán y no mostrarán nada.

Aquí hay un ejemplo de cómo se puede incluir ***placeholder*** en un formulario:

Código:

```
<form>
  <input placeholder="Buscar en la base de datos">
  <input value="Buscar">
</form>
```

## Campos con autofocus

El atributo de ***autofoco*** permite al usuario decidir y controlar qué campo de texto debe ser enfocado (señalado, activado) en cuanto la página es cargada o se esté cargando, permitiendo al usuario comenzar a escribir sin tener él que especificar cual es su campo de texto principal en su página. El atributo de ***autofoco*** es un atributo boleano (respuesta true - false) y no deberá haber más de un elemento en la página.

Muchos sitios utilizan JavaScript para focalizar y dirigir el cursor automáticamente al campo de texto. Por ejemplo Google cuando comienza a cargar su página dirigirá el cursos a su input de búsqueda automáticamente para que puedas empezar a escribir tus palabras de búsqueda en su navegador. Esto puede ser conveniente para algunos y para otros que pueden tener una necesidad específica no tanto. Si aprietas la barra de espacio esperando que la página baje haciendo un scroll, esto no sucederá porque está enfocado el input del formulario.

HTML5 introduce un atributo de control de ***autofoco*** en los formularios. El atributo autofoco hace exactamente lo que suena, en cuanto la web se comienza a cargar, mueve el cursor y así la atención del usuario a un campo <input> particular.

A día de hoy, Autofocus solo lo soportan Safari, Chrome y Opera. Firefox e IE, lo ignorarán.

Código:

```
<form>
 <input name="b" autofocus>
 <input type="submit" value="Search">
</form>
```

En el primer tutorial mencionamos varios de los nuevos atributos de los formularios, y en próximos artículos iremos profundizando en ellos. Ahora les dejo una galería de sitios hechos con HTML5 que podrán inspeccionar viendo el código fuente.