Tutorial HTML

autor:	Víctor Cuervo

versión:	v1.0
fecha:	2017-04-12
url:	http://www.manualweb.net/tutorial-html/
github:	https://github.com/manualweb/manualweb/tree/master/html

Índice

Introducción a la World Wide Web

Hoy en día la tecnología de la red, va avanzando a pasos agigantados. ¿Por qué? La razón es que tanto las empresas como a nivel individual la sociedad se está dando cuenta de todas las posibilidades que la 'red de redes' nos ofrece.

Antes de adentrarnos en el HTML deberemos de tener una serie de conceptos básicos sobre lo que es Internet y el mundo que lo rodea.

¿Qué es Internet?

Internet es un conjunto de ordenadores o red de ordenadores interconectados entre mediante algún soporte, ya sea físico o inalámbrico. El lenguaje que utilizan los ordenadores para hablarse entre ellos se llama TCP/IP. Mediante el lenguaje TCP/IP las máquinas se intercambian la información.

El Inicio: Comunicando máquinas

En 1969 nace un proyecto de índole militar en EEUU llamado ARPANET, este proyecto es el primero que permite conectar máquinas de diferentes universidades entre sí.

Poco a poco el proyecto fue cogiendo un matiz de investigación hasta separarse del propio proyecto militar, conectando más máquinas, y llegando a conceptualizar lo que es Internet Así hasta llegar a los día de hoy, donde las máquinas que están conectadas a la red son de empresas, universidades, personales,... dónde las máquinas ya no son solo ordenadores, sino que pueden ser móviles, elementos de domótica, coches,...

La World Wide Web y el HTML

Un poco más tarde, sobre 1990, en el CERN, Tim Berners Lee y una serie de colaboradores definió un sistema de almacenamiento y recuperación de información. De tal manera que la información pudiese ser consultada de una forma estándar desde cualquier sitio. Estaba naciendo el lenguaje que define la información, nacía HTML. Era la gestación de la **World Wide Web**, información enlazada una con otra creando el tejido que es la red.

Actualmente el lenguaje HTML tiene como especificación oficial HTML 5.0 *(28 de octubre de 2014)* y se está trabajando en el borrador de HTML 5.1 *(última actualización el 10 de marzo de 2016)*.

El lenguaje **HTML (HyperText Markup Language)** o lenguaje de etiquetas de hipertexto es un lenguaje que permite definir la estructura de una página web mediante un conjunto de elementos o etiquetas. La base de los documentos web o documentos HTML es que unos se van enlazando con otros creando una gran malla o red de documentos que es lo que conocemos como World Wide Web, “la red” o más comúnmente como “la web”.

Los Navegadores Web y las URL

Solo faltaban los programas que nos permitieran visualizar dicha información. De esta forma en 1993 se creaba el **navegador web** Mosaic.

Herramienta imprescindible para poder visualizar el contenido HTML. Los navegadores web son las herramientas que nos permiten movernos por la red, visualizar y acceder a los contenidos de la misma.

Muy atrás han quedado los tiempos en los que solo existía el navegador web Mosaic y ahora en el mercado podemos encontrar una gran cantidad de navegadores web: Chrome, Firefox, Opera, Safari, Internet Explorer,...

Para identificar cada uno de los contenidos que hay en la red se ha definido el concepto de **URL (o URI)**. Mediante una URL podemos acceder a cualquier contenido de la red utilizando un agente de usuario, más conocido como navegador web o browser.

Una URL se compone, a modo general, de diferentes partes:

- **Protocolo**, es el tipo de comunicación que queremos establecer con la máquina. Por ejemplo, si queremos visualizar un documento utilizaremos http, si queremos recuperar un fichero indicaremos ftp,...
- **Máquina**, será la máquina que tiene el contenido. Las máquinas se identifican mediante un código numérico llamado IP, pero como esos códigos con complejos de aprender las damos nombres. Así podemos conocer una máquina (o máquinas) como google.com, lineadecodigo.com, manualweb.net,...
- **Contenido**, será el contenido al cual queremos acceder. Este puede ser el nombre de un fichero, el nombre de una página web, el nombre de un vídeo,... El contenido suele estar ubicado en un directorio, por lo cual dicho nombre suele venir antepuesto de la ruta de un directorio.

La estructura general de una URL será la siguiente:

```
protocolo://máquina/contenido
```

Si nos fijamos en los nombres de las máquinas, estos suelen tener dos partes:

- **Nombre de la Máquina**, este suele ser el nombre del servicio al que intentamos acceder. Por ejemplo: google, manualweb, elpais,...
- **Dominio**, es la extensión que nos permite identificar el tipo o ubicación del servicio al que accedemos. Por ejemplo: .com, .net, .gov,... Aunque hay que tener cuidado ya que los dominios, en ciertos casos, no están reglados y pueden referirse a otra cosa distinta.

Algunos dominios que podemos encontrarnos son:

- **.com**, para empresas o compañías. - http://www.google.com
- **.org**, para organizaciones. - http://www.greenpeace.org
- **.edu**, para recursos de educación y universidades. - http://www.mit.edu
- **.mil**, para estamentos militares. - http://www.navy.mil
- **.gov**, para elementos gubernamentales. - http://www.whitehouse.gov
- **.es, .fr, .co, .pe, .ar,...** son dominios de países (españa, francia, colombia, perú, argentina,...) - http://www.red.es

Introducción al HTML

¿Qué es el HTML?

HTML (HyperText Markup Language) es un lenguaje que nos permite crear páginas web mediante etiquetas o elementos.

Mediante HTML podremos incluir dentro de nuestras páginas web *texto, imágenes, enlaces, tablas, vídeos, música,...* cualquier elemento que te puedas imaginar. Todo elemento es representado por una etiqueta.

Las páginas web que creemos **serán visualizadas mediante un navegador web o browser**, los cuales interpretan en contenido del documento y lo presentan de forma visual. De manera forman es conocido como **agente de usuario.** Ya que a parte de los navegadores web pueden existir programas que exploten de otra manera el contenido, por ejemplo los lectores de contenido adaptados a discapacitados.

La característica principal del HTML son los enlaces. Mediante los enlaces podremos realizar navegaciones entre páginas web, movernos de una a otra. Esta característica es la que hace diferente al lenguaje HTML de otros formatos como Microsoft Word, PDF,...

¿Quién define el HTML?

El organismo encargado de la definición, supervisión y evolución del lenguaje estándar HTML es el W3C (World Wide Web Consortium). Si bien, son las diferentes empresas que construyen software relacionado con el lenguaje HTML quienes implementan el lenguaje.

Esto puede dar en algunos casos que haya implementaciones de elementos por fuera del lenguaje, o que el mismo elemento se comporte de forma diferente dependiendo del navegador que lo visualice.

La especificación oficial del lenguaje HTML es HTML 5.0 *(28 de octubre de 2014) *y se está trabajando en el borrador de HTML 5.1 *(última actualización *el 10 de marzo de 2016).*

Herramientas para crear código HTML

Las páginas HTML son nada más que texto plano. Es por ello que, para crearlas, simplemente necesitemos de la ayuda de un editor o procesador de textos.

Editores de Texto

Puedes utilizar cualquier editor de texto. El Notepad o WordPad de Microsoft Windows, UltraEdit, NoteTab, el TextEdit de Mac, o un editor avanzado como Sublime Text o Atom.

Herramientas WYSIWYG

Si bien, existe otra alternativa que es el utilizar herramientas WYSIWYG *(Why You See Is What You*

Get). Estas herramientas nos proporcionan un interface gráfico para poder montar la página web, ocultándose el código HTML que se genera por debajo.

Si bien, estas herramientas, casi siempre, nos dejarán editar el código HTML generado.

Algunos ejemplos de herramientas *HTML WYSIWYG*:

- CoffeCup HTML Editor
- HoTMetal Pro
- DreamWeaver
- Microsoft Expression

Herramientas On-Line

Ahora está muy de moda el crear la página web desde una herramienta on-line. Es decir, desde una página web.

Este tipo de sitios te permiten dos cosas: por un lado *crear tu página web de forma gráfica* y por otro *almacenarla* (en muchos casos de forma gratuita) para que la pueda visualizar la gente.
Algunos de estos sitios son:

- Google Sites
- IM Creator
- WIX
- Sitios de blogs: WordPress, Blogger,...

La historia del HTML

La historia del HTML ha sido muy cambiante desde que Tim Berners-Lee y Robert Cailliau trabajasen en el **CERN** y tuviesen la necesidad de establecer una forma de **compartir la información.** Daba lugar al nacimiento del hipertexto, enlaces y protocolo http, todo lo que acabo siendo el lenguaje HTML.

Ha pasado por épocas de definición en sus primeras versiones **HTML 2** y **HTML 3.2**. Ha permitido la creación del W3C para la gestión de estándares. Ha vivido la ***"guerra de los navegadores"*** entre *Internet Explorer* y *Netscape*. Se ha conseguido estandarizar con gran aceptación en **HTML 4.01**.
Y ha visto como "unos disidentes" abandonaron el W3C para generar un evolucionado y moderno **HTML 5.**

Así podríamos resumir **la historia del HTML en las siguientes épocas**:

- Inicios del HTML
- HTML Tags y Mosaic
- HTML 2.0
- Creación de la W3C y HTML 3.0
- HTML 3.2

- HTML 4.01
- XML y XHTML 1.0
- HTML 5
- HTML.next

La Historia del HTML: Los Inicios

Los inicios del HTML

Los inicios de HTML se deben a Tim Berners-Lee cuando trabajaba en el **CERN (Centro Europeo de Investigación Nuclear)**. Y es que estando como trabajador del CERN se encontró con la problemática de poder facilitar el acceso a la información con la que trabajaban desde cualquier ordenador del centro o de otras instituciones que trabajaban con ellos.

Buscaban una forma sencilla y estándar de acceder a toda la información. Es en ese momento cuando nace el **protocolo HTTP** (hypertext transfer protocol) y las páginas HTML.

Además ideó que las páginas estarían unidas entre sí, estarían enlazadas. Era el concepto de **hipertexto.**

Para la definición del estándar HTML, Tim Berners-Lee se basó en el lenguaje de marcado **SGML (Standard General Markup Language)**. Este lenguaje define reglas de etiquetado y estructura generales. A partir de SGML se han definido lenguajes como HTML, Postscript, RTF,...

Tras tener el desarrollo del sistema de Hipertexto interno, Tim Berners-Lee lo presentó a una convocatoria para desarrollar el sistema Hipertexto en Internet junto con el ingeniero de sistemas Robert Cailliau. La propuesta que presentaron la llamaron **World Wide Web (W3)**.

HTML Tags y Mosaic

La primera versión del HTML nace hacía 1989 como un subconjunto de SGML y es especificada mediante un documento que se denomina HTML Tags.

En el documento HTML Tags ya podemos ver los conceptos básicos del lenguaje HTML ya que en él se define como insertar texto, títulos, enlaces y listas. Y algún elemento que acabó perdiéndose con el paso del tiempo como la identificación de índice del documento.

Podemos observar que intentan aglutinar en un lenguaje conceptos como estructura, formato y semántica, los cuales han ido derivando a la creación de otros lenguajes como CSS y XML.

Junto con HTML Tags **aparece el primer navegador para poder visualizar las páginas que se llamó "WorldWideWeb".**

Posteriormente se crearían otros navegadores web (Samba, Erwise, y Viola) además del que puede ser considerado como el primer navegador web global que fue Mosaic, desarrollado por NCSA.

El grupo de gente que creo Mosaic (Mark Andreessen entre ellos) abandonó NCSA para crear

posteriormente Netscape.

HTML 2.0

El proyecto World Wide Web se empieza a extender por el mundo, siendo varios proveedores de servicios (entre ellos AOL) los que empiezan a dar acceso a la red.

En paralelo a esta creciente entrada colaboradores aparece, en noviembre de 1995, HTML 2.0. La versión HTML 2.0 es desarrollada por el **IETF (Internet Engineering Task Force)**.

HTML 2.0 es la primera versión que podríamos considerar como estándar, o al menos definida por un organismo oficial.

En HTML 2.0 con respecto al inicial HTML Tags podíamos encontrar cosas como imágenes, mapas de imágenes, formularios, barras separadoras... así como una definición inicial del DTD HTML.

Creación de la W3C y HTML 3.0

Tras diferentes conversaciones entre Tim Berners-Lee y el MIT, el 1 de octubre de 1994 es creado el consorcio W3. Más conocido como W3C (World Wide Web Consortium) con la idea de definir estándares para Internet.

Dentro de la W3C se empieza a trabajar en noviembre 1995 sobre el borrador de HTML 3.0, el cual nunca llegará a ser recomendación y se quedará en borrador.

Es **HTML 3.2** la versión que pasa a ser la recomendación. El borrador de HTML 3.0 es interesante ya que se empieza a hablar de elementos como tablas, textos alrededor de las imágenes, y un elemento llamado MATH que permite crear fórmulas dentro del documento HTML.

El elemento MATH acabó cayéndose de posteriores revisiones del lenguaje HTML y conformando un nuevo lenguaje en sí llamado MathML.

Una de las cosas que más buscaba HTML 3.0 es que fuese compatible hacia atrás con todo lo que se había creado hasta entonces. Quizás un trabajo demasiado teórico.

HTML 3.2

Podríamos decir que HTML 3.2 es la primera versión de HTML ampliamente extendida y utilizada en la red.

HTML 3.2 dejaba de ser una especificación teórica, buscaba ser más práctica. **HTML 3.2 veía la luz en enero de 1997**.

En HTML 3.2 se pasaban a utilizar elementos que habían nacido fuera de la especificación y que habían sido definidos por los fabricantes como Netscape e Internet Explorer.

Así podemos encontrar en HTML 3.2 la *capacidad de crear código script, capas, formularios, posibilidad de meter Applets de Java, modificar el tamaño metiendo fuentes,..*

Quizás la especificación HTML 3.2 quedó demasiado abierta y empezó una **"guerra entre navegadores"** en la que cada fabricante quería presionar porque se aceptasen sus elementos. Así Internet Explorer presionaba para tener el **elemento MARQUEE **y Netscape para tener el elemento **BLINK**.

Historia del HTML: El estándar

HTML 4.01

Después de años de lucha entre los fabricantes la W3C intentaba poner un poco de orden con las versiones de HTML 4 y HTML 4.01 en los años 1998 y 1999 respectivamente.

Por el momento la especificación HTML 4.01 (24 de diciembre de 1999) es la más longeva del estándar.

En HTML 4.01 la W3C empieza con la separación de la estructura del documento con la de la representación visual. Así crea un lenguaje paralelo al HTML 4.01 llamado CSS.

Los elementos nuevos que aparecen en HTML 4.01 son las hojas de estilo (CSS), los objetos (para poder insertar elementos externos como vídeo y música) y los frameset para dividir la pantalla en partes.

Aunque HTML 4.01 era un lenguaje que definía de forma clara el estándar HTML y lo consensuaba entre los diferentes navegadores web, todavía dejaba abierta la posibilidad de interpretación de la estructura. Esta interpretación venía derivada de que la base del HTML seguía siendo el esquema de SGML.

XML y XHTML 1.0

Debido a la falta de coherencia en la definición de los esquemas derivados del SGML, la W3C decide definir un nuevo subesquema de SGML denominado XML. Este esquema sería un esquema bien definido y cerrado, lo cual podría proporcionarnos documentos coherentes y sin posibilidad de dobles interpretaciones.

El lenguaje XML 1.0 es creado el 10 de febrero de 1998, principalmente para la compartición de datos entre computadoras. Si bien, la W3C ve al XML como una posible solución a sus problemas de interpretación del HTML 4.01 creando el lenguaje XHTML 1.0 (eXtensible HyperText Markup Language).

XHTML 1.0 viene a ser los mismo que HTML 4.01 pero con una definición de documento basada sobre XML en vez de sobre SGML.

Así se dictan una serie de normas de las cuales la principal es que el documento XHTML 1.0 tiene que ser un **documento bien formado.** Para ello se establecen normas como que todo elemento que tenga una etiqueta de inicio debe de tener obligatoriamente una etiqueta de fin, que los nombres de los elementos y de los atributos deben de ir en minúsculas, no podrán existir atributos

que vayan sin valor,...

La definición de XHTML 1.0 pasa a ser bastante estricta para el lenguaje HTML.

HTML5

Una vez publicado XHTML 1.0 los esfuerzos de la W3C se dirigen a la creación de XHTML 2.0 como evolución del lenguaje.

Nuevamente la W3C se mete en un trabajo teórico, lo que hace que algunos "disidentes" (oportunistas para otros), como Ian Hickson, monten un grupo paralelo conocido como WHATWG (Web Hypertext Application Technology Working Group) que empieza a trabajar en HTML5 el 4 junio de 2004.

HTML5 empieza su definición apoyándose en dos puntos. Por un lado la compatibilidad hacía atrás de todo lo que hay creado y por otro la capacidad de absorber todas las funcionalidades que los nuevos fabricantes de la web habían ido construyendo, véase Google, Apple u Opera.

Además WHATWG siempre ha acusado a la W3C de tener unos procesos de estandarización muy largos y no alineados a las necesidades del mercado.

Entre las nuevas funcionalidades se encuentran la simplicidad para reproducir audio y vídeo, el disponer de un lienzo de dibujo denominado Canvas,.... Además alrededor de HTML5 nacen una gran cantidad de especificaciones para la mejora de las Webapps como son Websockets, Geolocalización, Webstorage,..

La W3C, en 2007, decide aparcar el trabajo realizado con XHTML 2.0 y empezar a trabajar con HTML5. Si bien el grupo WHATWG sigue con su trabajo y estudio sobre diferentes API, los cuales irán cayendo en el estándar.

La especificación de HTML5 es publicada oficialmente el 28 de octubre de 2014. Actualmente el HTML Working Group está trabajando en el borrador de HTML 5.1 *(cuya última actualización es del 10 de marzo de 2016).*

El futuro del HTML: HTML.next

Tanto la W3C como el WHATWG están trabajando sobre la evolución del HTML5. Cada uno a su ritmo.

Es por ello que es posible que muchas de las cosas en las que está trabajando WHATWG no acaben dentro del estándar HTML5. Así un subconjunto del trabajo sobre HTML5 pasará a ser recomendación del W3C y la otra parte pasará para las siguientes versiones. Esto es lo que se ha dado a conocer como HTML.next

En HTML.next están propuestos elementos como RECO, para el reconocimiento de voz, áreas de texto WYSIWYG, DATAGRID para representar datos tabulares, DATA para insertar elementos sólo reconocibles para máquinas,...

Tipo de documentos HTML

Lo primero que tiene que hacer un documento HTML es definir su **DOCTYPE o tipo de documento HTLM que es**. Para ello la W3C ha definido una serie de tipos de documentos.

La declaración del DOCTYPE tiene que ser la primera línea de nuestro documento HTML.

En el caso de que no indiquemos el DOCTYPE del documento el navegador interpretará el documento como quirks o modo de compatibilidad. Buscando que el contenido de la página pueda ser visualizado.

Algunos de los tipos de DOCTYPE que define la W3C son:

- HTML 4.01 transitorio
- HTML 4.01 frameset
- HTML Estricto
- HTML5

HTML 4.01 Estricto

Implica que el documento que generemos tiene que ser compatible con la definición del estándar HTML 4.01. Es por ello que todos los elementos que utilicemos en el documento tienes que estar dentro de la definición.

Este modo es interpretado correctamente por todos los navegadores y es compatible con el estándar CSS. Además que conseguiremos tener documentos accesibles.

Aunque no contempla las reglas del estándar XHTML sobre anidación, escritura,... es un punto sencillo para hacer una migración hacia XHTML.

La cabecera que utilizaremos en este caso será:

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

Hay que tener cuidado con el modo estricto ya que algunos navegadores antiguos pueden no interpretar este tipo de documentos.

HTML 4.01 Transitorio

El nombre de transitorio le viene porque no llega a ser un documento 100% estricto, pero está en camino o en transición a conseguirlo.

De igual manera que en el modo estricto los elementos deben de ser de la especificación HTML 4.01, si bien se permiten elementos que estén obsoletos o deprecados.

La cabecera que utilizaremos en estos casos será:

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

HTML 4.01 Frameset

En el caso de que nuestra página esté compuesta por frames deberemos de utilizar un tipo de DOCTYPE frameset.

La cabecera que utilicemos en estos casos será:

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

HTML5

En HTML5 han dado un giro a la cabecera ya que han visto que era demasiado compleja para los desarrolladores. Así que lo han reducido a la máxima expresión.

La cabecera que utilizaremos en los documentos HTML5 será:

```
<!DOCTYPE html>
```

Una de las cosas que vemos en este DOCTYPE es que ya no existe una dependencia del DTD del lenguaje.

Validando documentos

Una vez que hayamos construido un documento lo mejor que podemos es validarlo. Para ello la W3C nos proporciona un validador en http://validator.w3.org/

Mediante esta sencilla herramienta podremos validar un documento a partir de una URL de Internet, subiendo un fichero o escribiendo directamente en una caja de texto.

Así podremos ver si el documento que hemos generado es compatible con el DOCTYPE que hayamos definido.

Sintaxis del HTML

Dentro del HTML vamos a encontrar diferentes estructuras que deberemos de diferenciar. En primer lugar están **los elementos** que es la principal estructura del lenguaje y los que conforman las páginas web.

A su vez los elementos contendrán **atributos**. Los atributos dan más especificidad al comportamiento de los elementos, permitiendo parametrizarlos.

Los elementos HTML

Los elementos HTML son los que configuran la estructura de la página. También son llamados en algunos sitios como etiquetas, derivado del tema del lenguaje de marcado. Si bien su nombre correcto es el de elementos.

Todo elemento se encierra entre los símbolos **menor que** < y **mayor que** >:

```
<elemento>
```

Dentro de los elementos HTML encontramos dos tipos:

- Elementos que tienen un inicio y un cierre
- Elementos únicos

Los **elementos que tienen un inicio y un cierre** permiten tener a otros elementos u otro contenido anidado, es decir, a otros elementos o directamente texto. La estructura de los elementos de inicio y cierre es la siguiente:

```
<elemento> contenido| subelementos </elemento>
```

Como podemos apreciar el elemento de cierre se precede de una barra invertida.

Algunos de estos elementos son p, div, ul,...

En el caso de los elementos únicos, estos no permiten anidar contenido y aparecen de forma aislada. Su estructura es la siguiente:

```
<elemento />
```

Como podemos apreciar la barra invertida se encuentra al final y dentro del elemento. Algún ejemplo de estos elementos es img, br,...

Atributos en HTML

Los elementos pueden ser parametrizados mediante los atributos. Los atributos siguen la estructura nombre/valor y se deberán de poner antes del cierre del elemento.

La estructura de definición de los atributos es la siguiente:

```
<elemento atributo="valor">
```

El valor del atributo estará delimitado mediante comillas simples o comillas dobles.

Entidades en HTML

Otra estructura que nos podemos encontrar dentro de una página HTML son las entidades. Las entidades **empiezan por un ampersand &**, seguidas del **código de la entidad** -que puede ser numérico o alfanumérico- y **finalizan en un punto y coma ;**

La estructura de una entidad en HTML será la siguiente:

```
&código;
```

Las entidades nos sirven para representar símbolos que son parte de la estructura del lenguaje, como es el caso de los símbolos mayor y menor. O símbolos específicos de un determinado juego de caracteres, cómo podrían ser símbolos de monedas o caracteres especiales.

```
&lt;      representa <
&gt;      representa >
&quot;    representa '
&eur;     representa €
```

Ya hablaremos en detalle sobre ellas más adelante.

Comentarios en HTML

Para poder insertar un comentario dentro de las páginas HTML deberemos de utilizar la siguiente estructura:

```
<!-- comentario -->
```

El comentario puede tener varias líneas de texto.

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

Normas de codificación en un documento HTML

Existen una serie de normas de codificación que hay que conocer y seguir dentro del lenguaje HTML a la hora de crear nuestras páginas web.

No sensible a mayúsculas

El lenguaje HTML no es sensible a mayúsculas. Es decir, que da igual que escribamos nuestros elementos y atributos en mayúsculas y/o minúsculas. Si bien, por convenio, **se recomienda que tanto los elementos como los atributos sean escritos en minúsculas**.

Espacios en blanco

Si insertamos un espacio en blanco dentro de nuestra página se generará un espacio en blanco dentro de la visualización. Si bien si juntamos varios espacios en blancos, **estos solo generarán un único espacio en blanco**.

Lo mismo ocurre si insertamos una o varias tabulaciones. Estas solo generan un único espacio en blanco. Si queremos crear un conjunto de espacios en blanco seguidos deberemos de utilizar la entidad:

```
 
```

Cada vez que insertemos esta entidad generamos un espacio en blanco.

Saltos de línea

Los saltos de línea que insertemos dentro de la página web no tienen ningún efecto de cara a la renderización del contenido. Por lo tanto, un salto de línea en el código no genera nada.

Si queremos insertar un salto de línea dentro de nuestra página web deberemos de utilizar el elemento

```
<br />
```

Mi primera página web

Con lo que ya llevamos aprendido en el Manual de HTML es un buen momento para crear nuestra primera página HTML o mejor dicho, nuestra primera página web.

Herramientas básicas

Lo primero que vamos a necesitar es un editor de texto instalado en tu ordenador. Ya hemos visto que nos pueden valer el UltraEdit, NoteTab, el TextEdit de Mac, o un avanzado editor como Sublime Text o Atom. Dentro del editor de texto crea un documento de texto el cual llamaremos

`miprimeraweb.html`

Es importante saber que los documentos HTML tienen la **extensión .html o .htm**. Es más común la primera de ellas.

Crear la página HTML

Una vez tengamos nuestro documento de texto vamos a crear la estructura del documento HTML, con sus elementos html, head y body.

```
<! doctype html>
<html>
<head>
  <title>Mi Primera Página</title>
  <meta charset=”utf-8”/>
</head>
<body>
  <h1>Mi Primera Página</h1>
  Mi primera página en HTML.
</body>
</html>
```

Vemos que dentro del elemento body hemos insertado un elemento h1 con un texto y directamente texto que indica *“Esta es mi primera página web”*.

No te preocupes por los elementos meta y h1 que aparecen nuevos, ya que los veremos en detalle más adelante.

Visualizar la página HTML

Una vez guardada la página HTML vamos a visualizarla. Para poder visualizarla necesitamos un navegador web. Lo más normal es que tu ordenador ya venga con alguno instalado por defecto, si no puedes instalarte alguno como Google Chrome, Mozilla Firefox u Opera.

Una vez arrancado el navegador web simplemente tiene que abrir la página creada anteriormente, es decir, el archivo **miprimeraweb.html**

Verás que el navegador carga algo parecido a lo siguiente:

Metadados: Las metatags de HTML

Las metatags de HTML nos permiten dotar de metadatos a la página HTML. Es decir de información relativa al contenido de la página, pero que no se representa de ninguna forma. **Si bien son muy importantes ya que será una parte de información que leerán los buscadores web y un correcto uso de los metadatos harán que nuestra página sea mejor o peor indexada**.

Por ejemplo podremos encontrar dentro de los metadatos, la descripción de la página, un conjunto palabras que describan la página, el tipo de codificación de la página, información relativa al autor de la página o con qué herramienta ha sido construida, entre otras más.

La estructura general de las meta-tags se define mediante el elemento meta:

```
<meta name=”metadato” content=”valor” http-equiv=”cabecera-http” schema=”esquema”/>
```

Dentro de los metadatos podríamos diferenciarlos de tres tipos:

- **Metadatos generales**, que proporcionan información relativa al documento HTML.
- **Metadatos http**, son aquellos que modifican alguna propiedad de las cabeceras http.

Metadatos Generales

Nos permiten definir información de metadatos generales del documento: autor, descripción palabras clave,... Es estándar HTML 4.01 no define un perfil de metadatos a utilizar y deja abierto su uso. Si bien hace referencia al Dublin Core Profile para la descripción de documentos electrónicos. Algunos de los metadatos más utilizados son:

Author

Para hacer referencia al autor del documento. La estructura sería:

```
<meta name="author" content="Manual Web" />
```

Description

Define la descripción del contenido del documento a forma de resumen. Su uso sería:

```
<meta name="description" content="Manual Web que nos explica el uso del lenguaje
HTML" />
```

Keywords

Conjunto de palabras que describen el documento. Las palabras van separadas por comas. Se escribiría de la siguiente forma:

```
<meta name="keywords" content="manual,html,elementos,atributos,ejemplos" />
```

Revised

Información relativa a cuándo el documento fue revisado por última vez. Se utilizará de la siguiente forma:

```
<meta name="revised" content="24/03/2016" />
```

Metadatos Cabeceras HTTP

Las cabeceras http suelen ser establecidas en el servidor, si bien podemos modificarlas en el cliente mediante las meta-tags. El navegador realizará alguna acción al interpretar estas cabeceras. Por ejemplo podemos decirle al navegador cada cuanto tiempo tiene que recargar la página, o durante cuánto tiempo debe de cachear la página.

Estos metadatos se apoyan en el **atributo http-equiv**.

Refresh

Especifica cada cuanto tiempo tiene que recargar la página el navegador web. El tiempo es en segundos.

```
<meta http-equiv=”refresh” content=”5” />
```

Incluso podemos utilizar este tipo para hacer una redirección a otra página.

```
<meta http-equiv="refresh" content="2; http://lineadecodigo.com" />
```

Content-type

Nos sirve para identificar el tipo de documento, que será un documento de tipo text/html y la codificación del contenido. Si es ISO 8859-1, UTF-8,... Esto servirá al navegador a interpretar los caracteres que vayan dentro del contenido.

```
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
```

Cookie

Nos sirve para guardar una cookie. Los datos guardados son clave/valor

```
<meta http-equiv="cookie" content="clave=valor; expires=Saturday, 25-Mar-16
23:59:59 GMT;" />
```

Metadatos y el Open Graph Protocol

Como hemos comentado anteriormente la especificación HTML 4.01 no habla de un perfil de metadatos. Es por ello que hay diferentes perfiles que están proliferando diferentes perfiles que nos permiten metadatar el documento con algún fin.

Uno de esos perfiles es el Open Graph Protocol, este perfil es utilizado, entre otros, por Facebook para poder interpretar el documento de una forma más sencilla.

Algunos de los metadatos que define el Open Graph Protocol son:

og:title

Define el título del documento.

```
<meta name=”og:title” content=”Manual Web. Manuales en Español” />
```

og:type

Indica el tipo de objeto que representa la página. Si es un artículo, un vídeo, una imagen,...

```
<meta name="og:type" content="article" />
```

og:image

Permite establecer la imagen más representativa del documento

```
<meta name="og:image" content="http://www.manualweb.net/logo.png" />
```

og:url

Nos ayuda a definir la URL asociada al documento. Esto es por si queremos utilizar alguna diferente a la URL que ya tenga.

```
<meta name="og:url" content="http://www.manualweb.net" />
```

Texto en html

Si bien deberemos de tener en cuenta algunos condicionantes. Sobre todo si queremos insertar caracteres o símbolos particulares de nuestro idioma. Por ejemplo, cuando añadimos una ñ o el símbolo del copyright. Para ello tenemos las entidades, que veremos más adelante.

Así podemos tener nuestro documento HTML y escribir lo siguiente.

```
Este es mi primer texto en una web
```

Elementos HTML para trabajar con texto

Para poder enriquecer el texto básico que tengamos en una web, HTML nos ofrece una serie de **elementos de estructura** para poder ***dotar de semántica al contenido***. Ya que **la parte del estilo**: *texto en negrita, colores, modificar el tamaño,...* lo haremos con CSS.

Respecto a los elementos estructurales de texto de HTML los hemos dividido en dos partes:

- Elementos básicos para texto en HTML
  - Saltos de línea
  - Organizando el texto en párrafos
  - Elementos de cabecera o titulares
  - Crear una cita
  - Subíndices y superíndices
  - Enfatizando un texto

    - Resaltando un texto
- Elementos avanzados para texto en HTML
    - Abreviaturas y acrónimos
    - Textos preformateados
    - Notas de cambios en los documentos
    - Uso de código fuente
    - Fuentes o referencias a citas
    - Definiciones

Texto Básico en HTML

Saltos de línea en el texto

Está claro que escribir el texto todo seguido, sin darle un orden o una separación, sería un caos. Así que lo primero que vamos a aprender es el crear un salto de línea en el texto.

Y es que si escribimos lo siguiente:

```
Esta es una línea


Y esta otra línea.
```

Veremos que al cargar nuestra página web no se producirá el efecto deseado y que no existe la separación entre las dos líneas.

Y es que el salto de línea en un documento web equivale a un espacio en blanco a la hora de visualizarse. Y si ponemos muchos saltos de línea seguidos solo se contabilizará el primero, que será el que equivalga a un espacio en blanco.

Para poder añadir un salto de línea deberemos de utilizar el elemento br. El elemento br es un elemento simple. Es decir, no tiene un elemento de inicio y un elemento de cierre.

Así, para representar dos saltos de línea utilizaremos el siguiente código:

```
Esta es una línea <br/><br/>
Y esta otra línea.
```

Organizando el texto en párrafos

Ahora que conocemos el elemento br vemos su potencial, si bien, si queremos dar estilo a un documento a base de datos de línea, veremos que es bastante complicado.

Es por eso que el lenguaje HTML nos ofrece otra serie de elementos para organizar el contenido del texto. Así contamos con párrafos, los cuales son representados mediante el elemento p. En este

caso el elemento p tiene una elemento de inicio y un elemento de cierre. Y el contenido de dentro será lo que representa al párrafo.

```
<p>Parrafo</p>
```

De esta manera si queremos generar dos párrafos crearemos el siguiente código.

```
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed a felis non sem elementum tempor in at urna. Suspendisse auctor libero ut nibh consequat sed sagittis dolor iaculis. Donec condimentum mauris nec eros auctor sed vestibulum tellus consequat. Pellentesque tincidunt hendrerit neque, tincidunt tempus mauris consequat non.</p>


<p>Nullam interdum, enim sed ultrices sagittis, nibh tortor viverra lacus, eu tristique risus sapien et eros. Cras gravida, felis sed sagittis convallis, nulla ante vehicula justo, id imperdiet enim nisi id mauris. Nunc egestas volutpat congue. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed vehicula purus eu enim vulputate rhoncus.</p>
```

Elementos de cabecera o titulares

Si seguimos revisando el poder dar formato al texto nos encontramos con los elementos de cabecera o titulares.

Estos elementos nos servirán para poner títulos a las diferentes secciones que tenga nuestra web. Son dos elementos, el de inicio y el de cierre y el contenido será el nombre de la cabecera o titular.

El elemento es de tipo hx, donde la x va desde el 1 hasta el 6. Por ejemplo, si queremos poner un elemento de cabecera de tipo 1 escribiríamos lo siguiente:

```
<h1>Cabecera o Titulo</h1>
```

Así veríamos todos los elementos de cabecera:

```
<h1>Cabecera de tipo H1</h1>
<h2>Cabecera de tipo H2</h2>
<h3>Cabecera de tipo H3</h3>
<h4>Cabecera de tipo H4</h4>
<h5>Cabecera de tipo H5</h5>
<h6>Cabecera de tipo H6</h6>
```

Crear una cita

Otro elemento que nos servirá para organizar nuestro documento HTM será el elemento blockquote o q. Este elemento nos permite reseñar una cita.

La diferencia entre el elemento blockquote y elemento q es qué blockquote será un elemento de párrafo en si mismo, mientras que q será autocontenida dentro de otro texto, lo que se conoce como un elemento en línea.

De esta forma, una cita con blockquote será:

```
<blockquote>"Muchas veces me moría pensando que no iba verte. Pero moría la muerte cada vez que te veía". Eduardo Galeano</blockquote>
```

Y una cita con q puede ser:

```
El primer ministro afirmó que <q>"era el mejor momento económico para el páis"</q> el pasado día 8.
```

> Hay que tener cuidado con el elemento blockquote ya que en el pasado se utilizó de forma incorrecta para crear indentaciones o tabulaciones en los documentos web.

Los elementos blockquote y q nos permiten indicar el origen de la cita mediante el **atributo cite**.

```
El primer ministro afirmó que <q cite="http://elpais.com/">"era el mejor momento económico para el páis"</q> el pasado día 8.
```

Y el idioma en el que está escrita la cita, mediante el atributo lang:

```
El primer ministro afirmó que <q lang="ES-es">"era el mejor momento económico para el páis"</q> el pasado día 8.
```

Algo para las fórmulas: Subíndices y Superíndices

Aunque existen lenguajes de marcado especiales para la representación de fórmulas matemáticas, como puede ser MathML, en HTML encontramos un par de elementos que nos permitirán crear subíndices y superíndices. Estos son los elementos sub y sup respectivamente.

La representación del contenido irá entre el elemento de inicio y el elemento de cierre.

```
<SUP>superíndice</SUP>
<SUB>subíndice</SUB>
```

Podemos escribir algunas notas matemáticas como:

```
Como indica el teorema de Pitágoras:
hipotenusa<sup>2</sup> = cateto1<sup>2</sup>+cateto2<sup>2</sup>
```

O químicas como:

```
El símbolo del agua es H<sub>2</sub>O.
```

Enfatizando un texto

Una de las opciones que encontramos dentro del HTML es la de enfatizar un texto. Para ello el lenguaje HTML nos ofrece el elemento em.

El texto que vaya dentro de los elementos de inicio y de cierre del elemento em será el texto que aparece enfatizado en nuestro navegador.

Así podemos escribir el siguiente texto:

```
Más vale <em>pájaro en mano</em> que ciento volando.
```

Resaltando un texto, más énfasis

Si el énfasis que nos proporciona el elemento em no es suficiente para nuestro cometido, si queremos resaltar todavía más el texto, podemos utilizar el elemento strong.

El elemento strong tiene un comportamiento similar al elemento em.

Así, si queremos resaltar más un texto, podemos escribir.

```
A enemigo que huye, <strong>puente de plata</strong>.
```

Aunque **la mayoría de los navegadores representan el resaltado del texto elemento strong en negrita** no debemos de confundir esto con el fin del elemento strong, el cual es meramente estructural.

Texto Avanzado en HTML

Abreviaturas y acrónimos

Cuando estamos escribiendo un texto es posible que nos encontremos con partes que sean abreviadas o partes que sean acrónimos.

Aunque abreviatura y acrónimo suenan similares, estas, tienen su pequeña diferencia. **Una abreviatura es un tipo de abreviación que consiste en la representación gráfica reducida de una palabra mediante la supresión de letras finales o centrales**, y que suele cerrarse con punto; p. ej., afmo. por afectísimo; Dir.a por directora; íd. por ídem; SS. MM. por Sus Majestades; D. por don.

Mientras que **acrónimo es un tipo de sigla que se pronuncia como una palabra**; p. ej., o(bjeto) v(olante) n(o) i(dentificado).

De esta forma en el lenguaje HTML nos encontramos con el elemento abbr para las abreviaturas y el elemento acronym para los acrónimos. Si queremos insertar una abreviatura:

```
<abbr title="Director">Dir.</abbr> Juan de la Espina
<abbr title="Sus Majestades">SS. MM.</abbr> los Reyes de España
<abbr title="Calle">C.</abbr> del Pez nº40
```

En el caso de la abreviatura **se suele acompañar por un atributo title**, el cual nos ofrece el texto completo que representa la abreviatura.

Y si queremos insertar un acrónimo podemos escribir lo siguiente:

```
<acronym title="Objeto Volante No Identificado">OVNI</acronym></pre>
```

De igual forma que con el elemento abbr, en el elemento acronym encontramos el atributo title, el cual, en este caso, nos dirá que significan las siglas del acrónimo.

Otro atributo que solemos encontrar en estos elementos es el atributo lang, el cual hace referencia al idioma en el que está escrita la abreviatura o el acrónimo.

Textos preformateados

Ya hemos visto que a la hora de insertar texto en un documento HTML da igual poner muchos espacios o saltos de línea, ya que siempre serán ignorados.

Si bien, el lenguaje HTML nos ofrece el elemento pre. El elemento pre permite representar el texto tal cual es escrito, respetando sus espacios y saltos de línea. Acaba representando fielmente lo que hayamos insertado.

La estructura del elemento pre es:

```
<pre>Texto Preformateado</pre>
```

De esta forma podríamos escribir lo siguiente:

```
<pre>Esto es un texto
que está preformateado

y por lo tanto  --->   mantiene los espacios
                       y saltos de línea.</pre>
```

Que en pantalla nos mostrará:

```
Esto es un texto
que está preformateado

y por lo tanto  --->   mantiene los espacios
                       y saltos de línea.
```

Notas de cambios en los documentos

Hay que pensar que HTML fue pensado para compartir documentos electrónicos. Es por ello que el contenido de dichos documentos electrónicos iría avanzando en las revisiones que fuesen teniendo. Por lo tanto habría contenido nuevo y contenido eliminado.

A tal respecto el lenguaje HTML nos ofrece dos elementos. El primero es el elemento ins, este elemento sirve para indicar que el contenido es nuevo y que sustituye a un contenido que hayamos definido mediante el elemento del.

La estructura del elemento ins y del elemento del es sencilla:

```
<del>contenido eliminado</del>
<ins>contenido nuevo</ins>
```

Por ejemplo podríamos encontrarnos el siguiente código en un documento HTML utilizando el elemento ins y el elemento del:

```
Los hechos sucedieron en Madrid, el pasado día <del>26</del><ins>28</ins> de
febrero.
```

Manejando código fuente en HTML

Otra de las cosas para las que el lenguaje HTML fue pensando es el compartir código fuente, es decir, código informático.

Para poder cubrir esta necesidad nos ofrece un conjunto de elementos que representan semántica relacionada con el mundo de la computación.

Así tenemos.

- **code**, para insertar código fuente.
- **kbd**, para representar entradas de información por teclado.
- **samp**, para mostrar las salidas por consola de información.
- **var**, para definir las variables de un programa.

La estructura de los elementos code, kbd, samp y var es la misma:

```
<code> código fuente </code>
```

```
<kbd> entrada teclado </kbd>
<samp> salida por consola </samp>
<var> variable </var>
```

Así podríamos encontrarnos el siguiente ejemplo HTML que usase estos cuatro elementos:

```
El programa en Java se ejecuta mediante <kbd>java Saludo</kbd>. Lo que hará este
código es ejecutar el siguiente programa.

<code>public class Saludo
{
  public static void main(String[] args)
  {
    System.out.println("Hola"+ args[1]);
  }
}</code>

Dependiendo del valor que le demos a la variable <var>args</var> nos aparecerá un
saludo u otro. Así si ejecutamos como <kbd>java Saludo Esther</kbd> por pantalla
nos mostrará <samp>Hola Esther</samp>
```

Fuente o referencia de una cita

Aunque ya hemos visto que tenemos los elementos blockquote y q para crear citas en HTML, el lenguaje HTML nos ofrece otro elemento para establecer una referencia o fuente de una cita. Este es el elemento cite.

La estructura del elemento cite sería:

```
<cite>Fuente</cite>
```

Hay que indicar que este elemento no es para representar la cita, si no la fuente o referencia origen de la cita.

Por ejemplo podríamos escribir lo siguiente con el elemento cite:

```
<cite>La sombra del ciprés es alargada</cite>, empieza diciendo "Yo nací en Ávila,
la vieja ciudad de las murallas…"
```

Definiciones

Otro de los elementos que nos ofrece el lenguaje HTML es el elemento dfn, este elemento es utilizado para marcar un término que va a ser definido dentro de un contenido.

La estructura del elemento dfn es la siguiente:

```
<dfn>término</dfn>
```

Podemos utilizarlo de la siguiente forma:

```
Un <dfn>gabán</dfn> es capote con mangas, y a veces con capilla.
```

Colores en HTML

Colores RGB

Los colores en HTML se especifican mediante el estándar RGB (Red Green and Blue). Este estándar indica que una combinación de los tres colores básicos: rojo, verde y azul puede dar lugar a cualquier otro color.

Los valores que se les da al RGB son valores hexadecimales que van desde el 00 hasta el FF. Al valor del color se le antepone una almohadilla.

De esta forma un color rojo sería aquel que solo tiene activado el Red, el verde solo la parte del Green y el azul la parte del Blue. Así los colores básicos en HTML serán:

```
Rojo    #FF0000
Verde   #00FF00
Azul    #0000FF
```

Otras combinaciones generales de colores serían el negro (activando todos los colores), el blanco (desactivando todos los colores), amarillo (combinando Red y Blue), fucsia (combinando todo el rojo y todo el azul)

```
Negro     #FFFFFF
Blanco    #000000
Amarillo  #FFFF00
Fucsia    #FF00FF
```

Y luego ya combinaciones de los múltiples valores hexadecimales

```
Gris Plata    #C0C0C0
Azul Marino   #000080
Verde Oliva   #808000
…
```

Aunque lo más recomendable es utilizar el código hexadecimal para representar un valor, tenemos la alternativa de referirnos a los colores por su nombre en inglés. Este valor también será entendido por el navegador web. Así tendremos:

- Rojo - red
- Verde - green
- Azul - blue
- Blanco - white
- Negro - black
- Naranja - orange
- ...

Utilizar los colores en HTML

Los colores al ser elementos de estilo son utilizados en las CSS. Ya que HTML solo define la estructura de la página.

Pero vamos a ver, por encima, cómo podríamos cambiar el color de un texto. Para ello vamos a utilizar el atributo style. El atributo style nos permite asignar un estilo CSS a un elemento HTML.

El estilo que vamos a manipular es color. A ese atributo color le asignaremos un valor RGB.

```
<elemento style="color:#RGB" />
```

Por ejemplo si queremos cambiar el color a un header podríamos hacer lo siguiente.

```
<h1 style="color:#FF0000">Cabecera 1</h1>
```

Otro atributo CSS al que podríamos dar un colo es el color de fondo. Para ello deberemos de manipular el atributo background-color. De esta forma si queremos poner un color de fondo a una capa podríamos hacer lo siguiente.

```
<div style="background-color:#FFFF00">Mi Capa</div>
```

Es interesante que le eches un ojo al Manual de CSS para aprender más sobre el uso de colores en páginas web.

Imágenes en HTML

Si solo insertamos texto en nuestras páginas webs, estas quedarán muy sencillas y poco lucidas. Es por ello que en HTML tenemos la capacidad de insertar imágenes.

Las imágenes podrán ser elementos representativos de página web o elementos decorativos. Si bien, en el caso de ser elementos decorativos deberíamos de utilizar CSS para insertarlos en nuestra página web.

El uso de imágenes dentro de una página web tiene que hacerse con cautela, ya que cada imagen que pongamos en nuestra web incrementará el tamaño (peso) de la página. Por lo cual se verá

afectado el tiempo de descarga de la página.

Insertar una imagen: El elemento img

Para insertar una imagen en HTML tenemos el elemento img, cuya sintaxis básica es:

```
<img src=""nombreimagen.jpg"" alt="" />
```

Como podemos comprobar, el elemento img es un elemento sencillo, con lo cual no tiene elemento de cierre y termina con la barra invertida.

El atributo principal del elemento img es src nos indica la ruta de la imagen que queremos cargar. Así, si la imagen se encuentra en la misma ruta que nuestra página web pondremos:

```
<img src=""foto.jpg"" alt="" />
```

En el caso de que la imagen esté en otro directorio, por ejemplo en *"/multimedia/imagenes"* pondremos lo siguiente:

```
<img src=""/multimedia/imagenes/foto.jpg"" alt="" />
```

Incluso la imagen puede residir en otro servidor. En ese caso la URL que contenga la imagen deberá de indicar el protocolo y el servidor que alberga la imagen. Por ejemplo podremos tener el siguiente código:

```
<img src="//lineadecodigo.com/imagenes/logo.jpg"" alt="" />
```

Dimensiones de la imagen: width y height

Si no indicamos más sobre el elemento img, la imagen que le hayamos pasado en su campo src se cargará con su tamaño original.

Si bien disponemos de los atributos width para el ancho de la imagen y height para el alto de la imagen. De esta forma, si queremos que nuestra imagen se vea en 100x100 pixels, podemos insertar el siguiente código:

```
<img src=""foto.jpg"" alt="" width=""100"" height=""100"" />
```

El tamaño de la imagen puede ser especificado en pixels o en porcentajes. En caso de omitir la unidad se utilizará el pixel.

```
<img src=""foto.jpg"" alt="" width=""100"" height=""100"" />
```

```
<img src=""”foto.jpg”"" alt="" width=""”100px”"" height=""”100px”"" />
<img src=""”foto.jpg”"" alt="" width=""”50%”"" height=""”50%”"" />
```

Por cuestiones de rendimiento en la carga de las webs siempre es bueno el indicar sus valores width y height.

Los valores del alto y el ancho que indiquemos en la elemento img no tienen porqué coincidir con el tamaño real de la imagen. Por ejemplo, reduciendo los valores de estos atributos podremos conseguir una previsualización de la misma, lo que se conoce como thumbnail.

Texto alternativo de la imagen: el atributo alt y longdesc

Sobre una imagen podemos indicar un texto alternativo o descriptivo de la misma. Para ello tenemos el atributo alt.

```
<img src=""”foto.jpg”"" alt=""”texto" />
```

Pero, ¿por qué quiero poner un texto, cuando realmente es una imagen gráfica?

Piensa que esto es útil en varios casos. Por ejemplo, si por algún problema técnico no se puede cargar la imagen, el navegador mostrará en su espacio el texto alternativo, lo cual dará al usuario una idea de lo que iba en ese sitio.

Otra cosa útil es para cuando nuestra página sea utilizada por personas discapacitadas con problemas de visibilidad. En este caso estas personas disponen de herramientas que le van leyendo la página y cuando llegan a una imagen leen el contenido que encuentran en el atributo alt.

Es por ello que el texto alternativo que insertemos en la imagen debe de ser bastante descriptivo de la misma. En algunos casos se llega hasta indicar el tamaño de la imagen.

```
<img src=""”foto.jpg”"" alt=""”Fotografía" />
```

Tipos de formatos de imágenes

Hemos aprendido mucho de cómo podemos insertar nuestras imágenes en el documento HTML. Pero a la hora de insertar imágenes, qué tipo de imágenes puedo insertar en el documento HTML.

En este punto tenemos un pequeño problema y es que los estándares de HTML no definen los tipos de formato de imagen que se pueden ver en un navegador web.

Los formatos de imágenes más comúnmente aceptados son:

- **JPEG**, son imágenes digitales comprimidas con pérdida de información. Pero que nos permiten tener imágenes digitales que ocupen poco espacio.
- **GIF**, es un formato para imágenes de mapas de bits las cuales soportan 8 bits por pixel. El

formato GIF soporta imágenes animadas.
- **PNG**, es un formato de imagen en mapa de bits que emplea compresión de datos sin pérdida de información. No requieren de licencia de patente. Es un formato creado para utilizar imágenes en Internet con un tamaño adecuado.

Otros formatos que también son aceptados:

- **APNG**, son imágenes PNG animadas. Intenta evolucionar los gráficos animados en GIF.
- **SVG**, gráficos vectoriales escalables. Son gráficos especificados mediante texto, lo cual hace que sean interpretables por los dispositivos y puedan escalar a través de diferentes resoluciones.
- **BMP**, son imágenes de mapas de bits. Se pueden encontrar con extensión .bmp y .dib
- **BMP ICO y PNG ICO**, formato para representar iconos en el sistema operativo Windows. Los iconos suelen contener diferentes tamaños y densidad de pixels. De esta forma pueden ser escalados.

Mapas de imágenes

Los mapas de imágenes nos permiten crear un conjunto de enlaces dentro de una imagen o bien enlazar una parte en concreto de una imagen. Para ello HTML nos ofrece el elemento map.

La estructura del elemento map es la siguiente:

```
<map name="nombreMapa">
  <area/>
  <area/>
  ...
</map>
```

Lo que vemos es que el elemento map anida un conjunto de elementos area. Los elementos area serán los que establezcan las zonas enlazables dentro de la imagen.

Es importante saber que el mapa en sí no tiene una imagen asociada, si no que tendremos que asociar un elemento img al mapa para conseguir tener los áreas enlazables.

Nombre del Mapa

Una de las cosas más importantes en los mapas de imágenes es darle un nombre. Ya que este nombre será el que enlacemos sobre la imagen para poder usar el mapa de imágenes.

El nombre del mapa de imágenes se da mediante el atributo name.

```
<map name="nombreMapa"></map>
```

Tipos de Áreas

Dentro de los tipos de áreas que podemos crear dentro de una imagen tenemos diferentes formas:

- **Círculo**, define una región mediante un círculo.
- **Rectángulo**, define la región mediante un rectángulo.
- **Polígono**, define una región mediante un conjunto de puntos que representan un polígono.
- **Por defecto**, sería el resto de zonas no referenciada por ninguna zona.

El elemento area tiene la siguiente estructura:

```
<area shape="forma" coords="coordenadas" href="enlace" alt="texto alternativo" />
```

Dónde shape es la forma a utilizar, coords el conjunto de coordenadas que define la forma. Dependiendo de la forma utilizada serán unas coordenadas u otras. El atributo href contendrá el enlace y alt el texto alternativo a ese enlace.

Circle

Esta forma define un área circular dentro del mapa. En este caso las coordenadas son x,y como centro del círculo y radio.

```
<map>
  <area shape="circle" coords="x,y,radio"/>
</map>
```

Rect

Representa una forma rectangular en el mapa de imágenes. Las coordenadas son x1,y1 de la esquina superior izquierda seguido de x2,y2 de la esquina inferior derecha.

```
<map>
  <area shape="rect" coords="x1,y1,x2,y2"/>
</map>
```

Poly

Representa una forma de un polígono definido por un conjunto de puntos. Las coordenadas son x1,y1, x2, y2, x3, y3,..., xN, yN. Dónde la primer coordenada y la última deben de coincidir para poder cerrar el polígono.

```
<map>
  <area shape="poly" coords="x1,y1,x2,y2,x3,y3,...,xN,yN"/>
</map>
```

Default

Representa el resto de zonas del mapa que no hayan sido referenciadas por ninguna forma. En este caso no hay coordenadas.

Asociar Mapa a Imagen

Lo siguiente que tenemos que hacer es definir la imagen mediante el elemento img.

```
<img src=”imagen.jpg” />
```

Y asociarla el mapa de imágenes. Para ello utilizamos el atributo usemap al cual asignaremos el valor indicado en el atributo name del mapa.

```
<img src=”imagen.jpg” usemap=”#mapa” />
```

Ejemplo de Mapa de Imágenes

Para ver el uso de los mapas de imágenes analizaremos el siguiente caso. Vamos a partir de la siguiente imagen y vamos a definir tres mapas. El primero será un área circular sobre el segundo logo, el segundo será un área rectangular sobre el tercer logo y el tercero será un polígono sobre el último logo. En caso de que pinche en otro sitio se irá al enlace del HTML5 que será un área por defecto.

Así tendremos el siguiente mapa:

```
<map name="mapalogos">
 <area shape="rect" coords="405,73,520,166" href="#"/>
 <area shape="rect" coords="748,248,750,250" href="#"/>
 <area shape="poly" coords="571,119,626,59,687,118,628,177" href="#"/>
 <area shape="default" href="#"/>
</map>
```

Y el siguiente uso del mapa desde la imagen:

```
<img src="logos.png" usemap="#mapalogos" alt="Mapa de Logos" />
```

Enlaces en HTML

¿Qué son los enlaces en HTML?

Lo más importante de los documentos HTML son los enlaces. Ya que mediante los enlaces en HTML podemos comunicar una página con otra. De esta forma, enlazando documentos HTML podemos acabar tejiendo lo que es Internet.

Para crear un enlace en HTML utilizamos el elemento A con la siguiente sintaxis:

```
<a>Contenido del enlace</a>
```

Pero solo con esto el enlace no tendrá mucha utilidad ya que el principal objetivo del enlace es enlazar a un destino. Para poder indicar el destino de un enlace utilizamos el atributo href. En valor del atributo href puede ser cualquier URI que represente un recurso. Es decir, una página, una parte de una página, una URL genérica, un archivo,... De esta forma el enlace en HTML lo crearemos con la sintaxis:

```
<a href="URI">Contenido del enlace</a>
```

A modo de ejemplo podríamos tener los siguientes enlaces:

```
<a href="documento.html">Enlace a un documento</a>
<a href="documento.html#resumen">Enlace a una parte de un documento</a>
<a href="http://www.manualweb.net">Enlace a una web</a>
<a href="http://www.manualweb.net/tutorial-html/">Enlace a un directorio</a>
<a href="miimagen.png">Enlace a una imagen</a>
<a href="mimusica.mp3">Enlace a un archivo de sonido</a>
```

Destino del enlace

Pero, ¿dónde se abre el enlace? Pues por defecto y si no hemos configurado nada en el navegador web que estemos utilizando el enlace se abre en la misma ventana en la que tengamos el enlace.

Si bien, en el enlace, podemos indicar el destino que queremos darle a dicho enlace. Eso lo podemos hacer mediante el atributo target. Los posibles valores que admite el atributo target son:

- **_blank**, el agente de usuario intentará abrir el enlace en una nueva ventana. La nueva ventana no tendrá nombre.
- **_self**, el agente de usuario intentará abrir el enlace en el mismo marco donde está en código actual.
- **_parent**, el agente de usuario intentará abrir el enlace en el frameset inmediatamente superior al que se encuentra la página. Esto suele suceder cuando tenemos el enlace en un área de frames.
- **_top**, el agente de usuario intentará abrir el enlace en la ventana padre. En el caso de que exista un frameset lo eliminará y se hará con toda la ventana.
- **nombre_marco**, se podrá indicar el nombre de un frame. En este caso el agente de usuario intentará abrir el enlace en el frame que coincida con el nombre. En el caso de no existir un frame con ese nombre lo abrirá en una nueva ventana, asignándole dicho nombre.

Así podremos tener el siguiente código:

```
<a href="enlace.html" target="_blank">Abrir enlace en una nueva ventana</a>
<a href="enlace.html" target="_top">Abrir enlace en la ventana superior</a>
```

Título de los enlaces

El enlace en HTML, tal y como lo hemos visto hasta ahora, sirve para enlazar contra un recurso de la web: servidor, directorio, dominio,... Y lo que en mayor o menor medida describe lo que enlazamos es el contenido que encontramos entre las etiquetas A.

Si bien el elemento A nos ofrece un atributo llamado title. En el atributo title podemos describir de una forma textual el destino del enlace. Esto servirá no solo al usuario para que pueda obtener más información de dónde va, si no a las máquinas a la hora de asignar un nombre a la URI sobre la que estamos enlazando.

Un ejemplo sería:

```
<a href="http://www.manual.net" title="Web de Manuales y Tutoriales de
Programación">Ir a Manual Web</a>
```

Enlaces en HTML a una parte del documento

Hasta el momento lo que hemos visto es como montar enlaces en HTML a documentos. Ya sea porque enlazamos directamente al documento o bien porque enlazamos a un servidor o directorio que nos dará un documento.

Pero otra capacidad que tenemos en HTML es la de enlazar a una parte concreta del documento. Imagina que en un documento tenemos un título y varios capítulos. Y lo que queremos hacer desde otro documento HTML o bien desde el mismo documento es enlazar directamente al inicio de un capítulo.

Marcando una parte del documento

Para poder enlazar a una parte concreta de un documento lo primero que tenemos que hacer es marcar esa parte del documento. Para ello contamos con el atributo name. Si creamos un enlace con el atributo name, este enlace se definirá como posición y no como enlace de navegación.

La sintaxis será:

```
<a name="partedocumento">Contenido</a>
```

En este caso el contenido puede ser vacío ya que no se ofrecerá nada para navegar. Es por ello que podemos ponerlo antes de nuestro capítulo.

```
<p>Parrafo</p>
<a name="cap2"></a><h2>Nuevo Capítulo</2>
```

Es muy importante el contenido que le demos al atributo name, ya que dicho contenido vamos a utilizarlo para acceder desde un enlace.

Enlazando a una parte del documento

Una vez que hemos creado el marcaje del enlace en HTML en un documento es hora de acceder a esa parte del documento. Para ello solo tenemos que poner el nombre que le hayamos dado al atributo name precedido de una almohadilla.

La sintaxis será:

```
<a href="#name">Enlace a parte del documento</a>
```

Es decir, en el caso de que la parte marcada en el documento la hayamos marcado con “cap2” el enlace que tenemos que conformar será:

```
<a href="#cap2">Enlace al capítulo 2</a>
```

La parte del documento al que accedemos no tiene porqué estar en el mismo documento del enlace, puede estar en otro documento o servidor. De esta forma podemos tener enlaces a partes de otros documentos de la siguiente forma:

```
<a href="documento2.html#cap2">Enlace al capítulo 2 del documento2</a>
<a href="http://servidor.com/#cap2">Enlace al capítulo 2 del servidor</a>
```

El **utilizar la almohadilla** como valor del href nos puede servir para acceder a la parte superior del documento. Es por ello que esto es utilizado como enlace en las partes inferiores de los documentos HTML para subir a la parte de arriba.

```
<a href="#">Ir Arriba</a>
```

Enlaces en HTML con imágenes

En lo que va de capítulo sólo hemos visto enlaces en HTML cuyo contenido era texto. Si bien como contenido de los enlaces podemos poner imágenes.

```
<a href="http://www.victorcuervo.com"><img src=”victor.jpg” alt=”Foto de
Victor”/></a>
```

De esta forma toda la imagen será un elemento enlazable que nos llevará, en el caso de pinchar sobre ella, al destino indicado en el atributo href.

Esta técnica se suele utilizar para presentar una imágen de baja resolución y tamaño, que al

pulsarla nos cargue una imagen con mayor resolución y tamaño. Un código que podría ser de la siguiente forma:

```
<a href="foto.png"><img src="thumbnail_foto.png" alt="Mi foto"/></a>
```

Enlaces en HTML para descargar fichero

Otro de los usos que se les da a los enlaces en HTML es el de habilitar la descarga de ficheros. En este caso el destino indicado por el atributo href debe de ser el fichero que queremos descargar. En estos casos es bueno que el fichero a descargar este comprimido.

El código quedaría de la siguiente forma:

```
<a href="fichero.zip">Descargar el fichero</a>
```

Es importante que sepas que el navegador siempre va a intentar abrir el fichero enlazado en el atributo href para poder mostrarlo visualmente. En el caso de que no encuentre ningún programa para abrirlo nos mostrará un menú emergente en el que nos dará la posibilidad de guardar el fichero.

Protocolos en la URL del enlace

Hasta ahora hemos visto que todos los enlaces en HTML se apoyan en el protocolo http, pero el enlace especificado en el atributo href no tiene porqué ser http, si no que podría ser otro protocolo como ftp, mailto,...

```
<a href="ftp://servidorftp.com">Servidor FTP</a>
```

Lo bueno de utilizar el protocolo mailto dentro de los enlaces en HTML es que el navegador web nos va a abrir directamente el programa de correo electrónico que tengamos predeterminado en el ordenador.

La estructura del protocolo mailto en un enlace a sería la siguiente:

```
<a href="mailto:usuario@dominio.com">Email para usuario@dominio.com</a>
```

Lo bueno es que además podemos ponerle parámetros al modelo de mailto y se autorellenarán campos como el tema del email, campos CC, BCC,...

```
<a href="mailto:usuario@dominio.com?subject='Tema del Mensaje'">Email para
usuario@dominio.com</a>
```

Relaciones entre documentos: el elemento link

Hasta ahora hemos visto enlaces explícitos entre diferentes recursos. Si bien el lenguaje HTML nos da la posibilidad de establecer relaciones entre documentos sin tener que explicitar directamente un enlace. Para ello HTML nos ofrece el elemento link.

La estructura del elemento link es:

```
<link href="destino" rel="relacion" rev="relacion-inversa"/>
```

Es importante saber que el elemento link solo aparece dentro de la cabecera o head del documento.

El atributo rel establece la relación que hay con el documento destino, mientras que el atributo rev define la relación del documento destino con el documento actual. Es decir, la relación inversa.

Por otro lado el atributo href es el que contiene la URI del documento destino.

Uno de los usos del elemento link es para establecer las relaciones de documentos que formen una publicación completa. De esta forma manejando los valores del atributo rel podemos establecer dónde está el índice, cuál es artículo anterior y cual es el próximo artículo.

```
<link rel=”index” href=”indice.html”>
<link rel=”prev” href=”c2.html”>
<link rel=”next” href=”c4.html”>
```

Aunque los enlaces de tipo link no son renderizados por el navegador pueden ser interpretados para añadir la información al navegador.

Tipos de relaciones entre documentos

Según el lenguaje HTML se definen los siguientes tipos de relaciones entre documentos. Estos valore pueden ser utilizados por los atributos rel y rev.

- **alternate,** indica una versión alternativa del documento.
- **stylesheet,** hace referencia a una hoja de estilo externa para el documento.
- **start**, primer documento de un conjunto de documento.
- **next**, siguiente documento al actual.
- **prev**, documento anterior al actual.
- **contents**, documento que contiene la tabla de contenidos.
- **index**, documento que contiene el índice.
- **glossary**, documento que contiene el glosario.
- **copyright**, información del copyright del documento actual.
- **chapter,** documento que actúa como capítulo en una colección de documentos.
- **section,** documento que actúa como sección en un conjunto de documentos.

- **subsection,** documento que actúa como subsección en un conjunto de documentos.
- **appendix**, documento que actúa como apéndice de una colección de documentos.
- **help,** documento de ayuda.
- **bookmark,** para marcar un punto concreto del documento.

Enlaces en HTLM para hojas de estilo

En elemento link nos servirá para cargar las hojas de estilo del documento HTML. Ya veremos más adelante qué son, pero digamos que nos sirven para darle estilo gráfico a la página.

Las hojas de estilos se almacenan en ficheros .css. Así que podemos utilizar el elemento link para enlazarlas indicando que **su tipo es "text/css"**, mediante el atributo type.

```
<link href="style.css" rel="style" type="text/css"/>
```

Agrupaciones en HTML

Hasta ahora hemos visto cómo insertar diferentes elementos sobre un documento HTML. Estos elementos se irán mostrando según la secuencia en la que hayamos escrito el documento HTML.

Una de las cosas que tenemos que saber de los elementos html es si son elementos de bloque o elementos de línea.

Un **elemento de bloque** es aquél que una vez utilizado aparece en la siguiente línea y ocupa todo el ancho. Elementos de tipo bloque son los párrafos p, los formularios form, o las cabeceras hx.

Un **elemento en línea** es aquel que se muestra justo a continuación del anterior elemento. Estos elementos serían los enlaces a, imágenes img,...

El lenguaje HTML nos permite agrupar un conjunto de elementos mediante una agrupación en bloque o una agrupación en línea.

Agrupaciones en Bloque

Un elemento en bloque siempre empieza con una línea y su tamaño será igual al ancho disponible. El ancho disponible inicialmente es el de la página.

El elemento que nos permite realizar agrupaciones en bloque es el elemento div o más conocidos como capas. La estructura del elemento div es:

```
<div>
<!-- Contenido de la Capa -->
</div>
```

Los elementos en bloque pueden contener a otros elementos en bloque o bien a otros elementos en línea.

Por ejemplo podríamos agrupar en un bloque el siguiente contenido.

```
<div id=”micapa”>
  <h2>Título del Contenido</h2>
  Este es el contenido del artículo
  <img src=”logo.jpg” />
  <p>Más contenido del artículo</p>
</div>
```

Agrupaciones en Línea

Para poder realizar agrupaciones en línea tenemos el elemento span. La estructura del elemento span será:

```
<span> <!-- Contenido --></span>
```

Las agrupaciones en línea sólo pueden contener a otros elementos en línea, no a elementos de tipo bloque.

Por ejemplo podríamos tener la siguiente agrupación en línea.

```
<span id=”entrada”>
  <strong>Articulo Nuevo</strong>,
  <em>,12 de marzo de 2016</em>
</span>
```

Es muy normal que los agrupadores, ya sean o bien div, o bien span lleven el atributo id o class, ya que a posteriori serán manipulados mediante hojas de estilo CSS utilizando dichos identificadores.

Listas en HTML

Las listas en HTML nos permite crear conjuntos de elementos en forma de lista dentro de una página, todos los cuales irán precedidos, generalmente, por un guión o número.

Los tipos de listas en HTML son los siguientes:

- Listas ordenadas
- Listas desordenadas
- Listas de definiciones

Listas Ordenadas

Las listas en HTML ordenadas son aquellas que nos muestran los elementos de la lista en orden. Para representar el orden tendremos los elementos numerados. Es decir, cada uno de los elementos irá precedido de un número o letra que establezca su orden.

Las listas en HTML ordenadas se representan mediante el elemento OL.

```
<ol> ... </ol>
```

Cada uno de los elementos de la lista ordenada se representará mediante el elemento LI.

```
<ol>
  <li>Elemento 1</li>
  <li>Elemento 2</li>
  ...
  <li>Elemento N</li>
</ol>
```

Un ejemplo de lista ordenada sería el siguiente:

```
<ol>
  <li>Julio</li>
  <li>Carmen</li>
  <li>Ignacio</li>
  <li>Elena</li>
</ol>
```

Que produciría el siguiente efecto:

1. Julio
2. Carmen
3. Ignacio
4. Elena

Número de inicio de la lista: start

El atributo start nos permite indicar el número por el cual queremos que empiece la lista, ya que por defecto las listas ordenadas en HTML empiezan por el número 1.

```
<ol start="numero"> … </ol>
```

De esta forma, si queremos que la lista empiece por el número 5, escribiremos el siguiente código:

```
<ol start="5">
  <li>Julio</li>
  <li>Carmen</li>
  <li>Ignacio</li>
  <li>Elena</li>
</ol>
```

Que produciría el siguiente efecto:

1. Julio
2. Carmen
3. Ignacio
4. Elena

Tipo de lista ordenada: type

De igual manera podemos indicar el tipo de lista ordenada en HTML que queremos representar.

Entre los tipos de listas que podemos representar tenemos:

- 1 - Listas decimales
- a - Listas alfabéticas en minúsculas
- A - Listas alfabéticas en mayúsculas
- i - Listas de números romanos en minúsculas
- I - Listas de números romanos en mayúsculas

Los valores indicados al principio son los que le podemos asignar al atributo type de la lista ordenada en HTML.

```
<ol type="tipo"> … </ol>
```

Si queremos que nuestra lista aparezca ordenada mediante letras en mayúsculas escribimos lo siguiente:

```
<ol type=”A”>
  <li>Julio</li>
  <li>Carmen</li>
  <li>Ignacio</li>
  <li>Elena</li>
</ol>
```

En este caso la lista se representará de la siguiente forma:

1. Julio
2. Carmen
3. Ignacio
4. Elena

Atributos start/type en HTML 4.01

Aunque en HTML 5 han vuelto a la especificación los atributos type y start hay que tener cuidado con ellos, ya que en ciertas versiones como HTML 4.01 fueron declarados obsoletos, por lo cual si usamos tipos de documentos HTML 4.01 strict podríamos tener un problema en su definición.

Para estos casos (y para todos en general) podemos utilizar las hojas de estilo CSS de cara a dar estilo a las listas en HTML.

De esta forma en CSS podemos escribir lo siguiente:

```
<style type="text/css">
ol {
  list-style-type: lower-roman;
}
</style>
```

Lo cual hará que las listas en HTML ordenadas se muestren en números romanos y en minúsculas.

Listas en orden inverso: reversed

En HTML 5 aparece el atributo reversed para las listas ordenadas. El atributo reversed nos permite invertir el orden de la lista. Es decir, realiza la numeración de la lista de forma inversa.

Para utilizarlo simplemente indicamos el atributo reversed sobre el elemento OL.

```
<ol reversed> … </ol>
```

En este caso, si escribimos la siguiente lista:

```
<ol reversed>
  <li>Julio</li>
  <li>Carmen</li>
  <li>Ignacio</li>
  <li>Elena</li>
</ol>
```

Lo que obtendremos por pantalla será lo siguiente:

1. Julio
2. Carmen
3. Ignacio
4. Elena

Listas Desordenadas

Las listas desordenadas en HTML nos sirven para mostrar los elementos sin ningún tipo de orden, simplemente precedidos por una viñeta que puede ser un punto, un cuadrado,...

Para definir una lista desordenada en HTML utilizamos el elemento ul.

```
<ul> … </ul>
```

Para representar los elementos de la lista desordenada utilizamos el mismo elemento que con las listas ordenadas, es decir, el elemento li.

```
<ul>
  <li>Elemento 1</li>
  <li>Elemento 2</li>
  ...
  <li>Elemento N</li>
</ul>
```

De esa forma podríamos tener el siguiente código HTML:

```
<ul>
  <li>FC. Barcelona</li>
  <li>Real Madrid</li>
  <li>Real Betis</li>
  <li>Atlético de Madrid</li>
</ul>
```

Lo que nos mostrará por pantalla:

- FC. Barcelona
- Real Madrid
- Real Betis
- Atlético de Madrid

Tipos de lista desordenada

En el caso de las listas en HTML desordenadas en HTML no podemos indicarle el tipo de lista mediante HTML. En este caso tenemos que recurrir a CSS para poder indicar el tipo de lista utilizando el atributo list-style-type

```
<style type="text/css">
ul {
  list-style-type: circle;
}
</style>
```

Listas de Definiciones

Las listas en HTML de definiciones en HTML nos sirven para montar listas en las que tenemos la estructura valor y definición. Suelen ser listas para definir términos, como si fuese un diccionario, si bien pueden ser cualquier par valor-definición.

Las listas en HTML de definiciones en HTML se construyen mediante el elemento dl.

```
<dl> … </dl>
```

En este caso, dentro de las listas en HTML de definiciones tenemos dos elementos anidados, el que

representa al valor dt y el que representa a la definición dd. De esta forma la estructura de las listas en HTML de definiciones es la siguiente:

```
<dl>
  <dt>Término1</dt>
  <dd>Definición 1</dd>
  <dt>Término 2</dt>
  <dd>Definición 2</dd>
  …
  <dt>Término N</dt>
  <dd>Definición N</dd>
</dl>
```

Si queremos crear una lista en HTML con definiciones de palabras, podemos escribir lo siguiente:

```
<dl>
  <dt>Pizpireta</dt>
  <dd>Dicho de una mujer: Viva, pronta y aguda.</dd>
  <dt>Pulular</dt>
  <dd>Dicho de las personas, animales o cosas: Abundar y bullir en un lugar.</dd>
  <dt>Concupiscencia</dt>
  <dd>En la moral católica, deseo de bienes terrenos y, en especial, apetito
desordenado de placeres deshonestos.</dd>
</dl>
```

Lo cual nos acabará mostrando por pantalla lo siguiente:

Pizpireta
: Dicho de una mujer: Viva, pronta y aguda.

Pulular
: Dicho de las personas, animales o cosas: Abundar y bullir en un lugar.

Concupiscencia
: En la moral católica, deseo de bienes terrenos y, en especial, apetito desordenado de placeres deshonestos.

Por defecto los navegadores dejan el término y en la siguiente líne, junto con un tabulador, la definición.

Listas anidadas

Cuando estemos manejando listas podemos anidar unas en otras independientemente del tipo de lista que estemos anidando.

Para crear listas anidadas en HTML simplemente tenemos que hacer que el elemento de una de las listas sea a su vez una lista. Es decir, el esquema de listas sería parecido al siguiente:

```
<ul>
  <li>Elemento 1</li>
```

```
  <li>Elemento 2</li>
  <li>
    <ol>
      <li>Elemento 2.1</li>
      <li>Elemento 2.2</li>
      …
      <li>Elemento 2.N</li>
    </ol>
  </li>
  <li>Elemento 3</li>
  …
  <li>Elemento N</li>
</ul>
```

En este caso hemos anidado una lista ordenada dentro del tercer elemento li de una lista desordenada.

Hay que tener cuidado de poner el elemento li dentro de la lista anidada, ya que si no lo ponemos estaremos generando un código incorrecto.

Las anidaciones de listas puede ser de cualquier tipo de lista y sin límite de anidación.

Un ejemplo de listas anidadas sería una clasificación de animales como la siguiente:

```
<ul>
  <li>Carnívoro
    <ul>
        <li>León</li>
        <li>Buitre</li>
        <li>Hiena</li>
    </ul>
  </li>
  <li>Herbívoro
    <ul>
        <li>Cabra</li>
        <li>Vaca</li>
    </ul>
  </li>
  <li>Omnívoro
    <ul>
        <li>Oso</li>
        <li>Urraca</li>
    </ul>
  </li>
</ul>
```

El efecto que obtendríamos sería el siguiente:

- Carnívoro
  - León
  - Buitre
  - Hiena

- Herbívoro
  - Cabra
  - Vaca
- Omnívoro
  - Oso
  - Urraca

Tablas en HTML

¿Qué son las tablas en HTML?

Las tablas es el elemento del lenguaje HTML que utilizaremos para mostrar información de forma agrupada. Ya sea texto, imágenes, vídeos,...

El elemento table será el que nos ayudará para crear las tablas en HTML.

Un mal uso de las tablas en HTML es el motivo de maquetación, uso que se dió a las tablas en los principios del lenguaje HTML. Algo que hoy en día es una mala práctica. Cambiando por un modelo de maquetación apoyado en capas.

Crear una tabla simple

Para crear una tabla vamos a necesitar conocer, al menos, tres elementos: table, tr y td. Si bien existen otros cuantos que nos permitirán ampliar la funcionalidad de las tablas.

El primer elemento es table. table es el elemento que representa las tablas y será el que agrupe al resto de elementos. Por lo tanto tiene sus etiquetas de inicio y cierre.

```
<table> … </table>
```

Siguiendo con la construcción de la tabla el siguiente elemento que necesitamos es tr. El elemento tr representa a una fila de la tabla. Por lo tanto tendremos tantos elementos tr como filas tenga la tabla.

Así, si queremos tener una tabla de tres filas tendremos un código como el que sigue:

```
<table>
  <tr> … </tr>
  <tr> … </tr>
  <tr> … </tr>
</table>
```

El último elemento que necesitamos conocer es td. El elemento td es el que representa de una forma unitaria a una celda. Por lo tanto los elementos tr tendrán tantos elementos td en su interior como celdas contenga la fila.

El contenido que haya entre los elementos td será el contenido de la celda.

Una tabla de tres filas por cuatro columnas quedaría de la siguiente forma:

```
<table>
  <tr>
    <td>Fila 1 Columna 1</td>
    <td>Fila 1 Columna 2</td>
    <td>Fila 1 Columna 3</td>
    <td>Fila 1 Columna 4</td>
  </tr>
  <tr>
    <td>Fila 2 Columna 1</td>
    <td>Fila 2 Columna 2</td>
    <td>Fila 2 Columna 3</td>
    <td>Fila 2 Columna 4</td>
  </tr>
  <tr>
    <td>Fila 3 Columna 1</td>
    <td>Fila 3 Columna 2</td>
    <td>Fila 3 Columna 3</td>
    <td>Fila 3 Columna 4</td>
  </tr>
</table>
```

Así con los tres elementos table, tr y td tenemos construida nuestra tabla.

Visualmente tendríamos algo así:

Fila 1 Columna 1	Fila 1 Columna 2	Fila 1 Columna 3	Fila 1 Columna 4
Fila 2 Columna 1	Fila 2 Columna 2	Fila 2 Columna 3	Fila 2 Columna 4
Fila 3 Columna 1	Fila 3 Columna 2	Fila 3 Columna 3	Fila 3 Columna 4

Tablas Básicas en HTML

Poner título a la tabla

Ahora que conocemos cómo se construye una tabla básica con HTML vamos a ir viendo qué posibilidades adicionales tenemos sobre las tablas.

Lo primero que haremos será poner un título a la tabla. Para ello vamos a utilizar el elemento caption. El contenido del elemento caption será el título que le asignemos a la tabla.

Añadimos el elemento caption antes de cualquier definición de filas dentro de la tabla.

Así, el código para poner el título a la tabla sería:

```
<table>
<caption>Mi tabla de ejemplo</caption>
  <tr>
```

```
    <td>Fila 1 Columna 1</td>
    <td>Fila 1 Columna 2</td>
    <td>Fila 1 Columna 3</td>
    <td>Fila 1 Columna 4</td>
  </tr>
</table>
```

Resumen de la tabla

Aunque el título suele ser el elemento descriptivo de la tabla existe un atributo a nivel del elemento table de especial importancia. Este es el **atributo summary**. El **atributo summary** nos permite adjuntar un resumen de la información que contiene la tabla.

Este atributo será de gran interés para cuando nuestras páginas web sean interpretadas por programas para discapacitados, ya que la información que contiene una tabla suele ser de difícil interpretación.

Es por ello que es muy recomendable que siempre añadamos un resumen a nuestra tabla.

El código es tan sencillo como este:

```
<table summary="Tabla que contiene datos de ejemplo para poder explicar como
construir tablas con el lenguaje HTML">
<caption>Mi tabla de ejemplo</caption>
  <tr>
    <td>Fila 1 Columna 1</td>
    <td>Fila 1 Columna 2</td>
    <td>Fila 1 Columna 3</td>
    <td>Fila 1 Columna 4</td>
  </tr>
</table>
```

Definiendo Una Cabecera en la Tabla

Una cosa que vemos es que las tablas suelen tener una primera fila de encabezado. Dentro de las tablas en HTML podemos identificar esta fila mediante el elemento th. Es decir, para las celdas de la fila de cabecera en vez de utilizar un elemento td utilizaremos un elemento th.

Así, la cabecera de una tabla quedará de la siguiente forma:

```
<table>
  <tr>
    <th>Cabecera 1</th>
    <th>Cabecera 2</th>
    <th>Cabecera 3</th>
  </tr>
  <tr>
    <td>Fila 1 Columna 1</td>
    <td>Fila 1 Columna 2</td>
    <td>Fila 1 Columna 3</td>
```

```
  </tr>
</table>
```

El atributo scope

Hay celdas de cabecera que necesiten una pequeña explicación sobre si la información que representan es la de las columnas o la de las filas. Suele suceder, normalmente, con la primera celda.

Columna o Fila	□	□
□		
□		

Para resolver este problema tenemos el atributo scope. El atributo scope solo se puede aplicar a las celdas de una cabecera. Y sus valores son: “col”, “row”, “colgroup” o “rowgroup”.

De esta forma si queremos que esta celda sea representativa de columnas la definiremos como:

```
<td scope=”col”>Contenido de la Celda</td>
```

Agrupando Celdas

A la hora de presentar datos en una tabla nos puede surgir la necesidad de agrupar celdas, ya que el contenido a presentar refleja una agrupación de datos.

Imaginemos una tabla que nos saca los ingresos y gastos por meses. Existirá una columna con el mes, la cual agrupará dos columnas: ingresos y gastos.

Algo como lo siguiente:

Enero		Febrero	
Ingresos	Gastos	Ingresos	Gastos
1.000€	700€	1.100€	580€
1.800€	920€	1.750€	920€

En este caso lo que estamos diciendo es que una celda ocupa dos espacios. Para ello vamos a utilizar el atributo colspan sobre el elemento td de la celda.

Así indicaremos que el colspan de esa celda es 2. Ya que ocupa dos celdas.

```
<td colspan="2">Enero</td>
```

El código completo sería:

```
<table>
```

```
  <tr>
    <td colspan="2">Enero</td>
    <td colspan="2">Febrero</td>
  </tr>
  <tr>
    <td>Ingresos</td>
    <td>Gastos</td>
    <td>Ingresos</td>
    <td>Gastos</td>
  </tr>
  <tr>
    <td>1.000€</td>
    <td>700€/td>
    <td>1.100€</td>
    <td>580€</td>
  </tr>
  <tr>
    <td>1.800€</td>
    <td>920€</td>
    <td>1.750€</td>
    <td>920€</td>
  </tr>
</table>
```

De igual manera nos puede suceder en sentido horizontal. Es decir, que queramos que una celda ocupe dos filas.

Si lo vemos sobre nuestro ejemplo veremos que podemos añadir una columna que simplemente ponga que los valores numéricos tengan el literal "Datos Económicos". En este caso tendremos que indicar que esa celda ocupa dos filas.

	Enero		Febrero	
	Ingresos	Gastos	Ingresos	Gastos
Datos Económicos	1.000€	700€	1.100€	580€
	1.800€	920€	1.750€	920€

Para la agrupación de filas tenemos otro atributo que es rowspan. Este atributo, al igual que colspan se aplica sobre la celda td

```
<td rowspan="3">Datos Económicos</td>
```

El código completo de la tabla sería el siguiente:

```
<table>
  <tr>
    <td></td>
    <td colspan="2">Enero</td>
    <td colspan="2">Febrero</td>
  </tr>
  <tr>
```

```
    <td rowspan="3">Datos Económicos</td>
    <td>Ingresos</td>
    <td>Gastos</td>
    <td>Ingresos</td>
    <td>Gastos</td>
  </tr>
  <tr>
    <td>1.000€</td>
    <td>700€/td>
    <td>1.100€</td>
    <td>580€</td>
  </tr>
  <tr>
    <td>1.800€</td>
    <td>920€</td>
    <td>1.750€</td>
    <td>920€</td>
  </tr>
</table>
```

Tablas Avanzadas en HTML

Grupos de Filas

Ya hemos visto que las tablas se van definiendo por filas mediante el elemento tr. Pues dentro de HTML podemos agrupar estas filas por funcionalidad.

Para ello podemos agrupar las filas en tres partes:

- **Cabecera**, representada por el elemento thead.
- **Cuerpo**, representada por el elemento tbody.
- **Pie**, representada por el elemento tfoot.

Cada uno de estos elementos tendrá una o n filas, dependiendo de las que queramos agrupar. La estructura es la misma para los tres casos:

```
<thead>
  <tr> <!-- fila(s) --></tr>
</thead>

<tbody>
  <tr> <!-- fila(s) --></tr>
</tbody>

<tfoot>
  <tr> <!-- fila(s) --></tr>
</tfoot>
```

Es importante saber que no es necesario que aparezcan en ese orden dentro de la tabla, este podría ser alterados. Si bien el navegador si que las representará en dicho orden.

De esta forma podríamos tener la siguiente tabla con agrupaciones:

```
<table>
  <thead>
    <tr>
    <td scope=”row”>Mes</td>
      <td>Enero</td>
      <td>Febrero</td>
    </tr>
  </thead>
  <tfoot>
    <tr>
    <td>Total</td>
      <td>15</td>
      <td>25</td>
    </tr>
  </tfoot>
  <tbody>
    <tr>
    <td>Agua</td>
      <td>10</td>
      <td>15</td>
    </tr>
    <tr>
    <td>Gas</td>
      <td>5</td>
      <td>10</td>
    </tr>
  </tbody>
</table>
```

Que representaría lo siguiente:

Agua	10	15
Gas	5	10
Total	15	25

Grupos de columnas

Ya hemos visto que las tablas se definen por filas. Pero una de las cosas que nos ofrece HTML es la posibilidad de definir, sobre dichas filas, grupos de columnas que tengan una relación semántica entre ellas.

Por ejemplo en la siguiente tabla vemos que hay una relación semántica de las columnas relativa a los meses.

Enero		Febrero	
Ingresos	Gastos	Ingresos	Gastos
1.000€	700€	1.100€	580€
1.800€	920€	1.750€	920€

Para poder definir estas relaciones semánticas entre las columnas HTML nos ofrece el elemento colgroup.

El elemento colgroup se define al principio de la tabla y tiene la siguiente estructura.

```
<colgroup span=”numero-columnas” width=”ancho”></colgroup>
```

Dónde el **atributo span** indica el número de columnas que representa la agrupación. Empezando de izquierda a derecha.

Además contamos con el atributo width el cual nos permite especificar un ancho para la columna.

En la tabla que hemos visto el código HTML quedaría de la siguiente forma:

```
<table>
  <colgroup span=”2” width=”100”></colgroup>
  <colgroup span=”2” width=”100”></colgroup>
  <tr>
    <td colspan=”2”>Enero</td>
    <td colspan=”2”>Febrero</td>
  </tr>
  <tr>
    <td>Ingresos</td>
    <td>Gastos</td>
    <td>Ingresos</td>
    <td>Gastos</td>
  </tr>
  <tr>
    <td>1.000€</td>
    <td>700€/td>
    <td>1.100€</td>
    <td>580€</td>
  </tr>
  <tr>
    <td>1.800€</td>
    <td>920€</td>
    <td>1.750€</td>
    <td>920€</td>
  </tr>
</table>
```

Si no queremos utilizar el atributo span o si bien queremos manipular los estilos gráficos de las columnas, tenemos otro elemento, este es el elemento col.

El elemento col aparecerá dentro del elemento colgroup tantas veces como columnas queramos agrupar.

La estructura del elemento col es:

```
<col span=”numero-columnas” width=”ancho-columna” />
```

Es decir que también permite agrupar columnas mediante su atributo width y darles un ancho mediante el atributo width.

El anterior ejemplo utilizando el elemento col quedaría de la siguiente forma:

```
<table>
  <colgroup>
    <col width=”100”>
    <col width=”100”>
  </colgroup>
  <colgroup>
    <col width=”100”>
    <col width=”100”>
  </colgroup>
  <tr>
    <td colspan=”2”>Enero</td>
    <td colspan=”2”>Febrero</td>
  </tr>
  <tr>
    <td>Ingresos</td>
    <td>Gastos</td>
    <td>Ingresos</td>
    <td>Gastos</td>
  </tr>
  <tr>
    <td>1.000€</td>
    <td>700€/td>
    <td>1.100€</td>
    <td>580€</td>
  </tr>
  <tr>
    <td>1.800€</td>
    <td>920€</td>
    <td>1.750€</td>
    <td>920€</td>
  </tr>
</table>
```

Tablas para agentes de usuario no visuales

Las tablas no siempre serán representadas por un navegador web o agente de usuario visual. Hay otro tipo de agentes de usuario que son no visuales y que suelen estar adaptados para discapacitados.

Asociar celdas a cabeceras

En este sentido tenemos que saber cómo dar formato a las tablas para que estos agentes de usuario no visuales puedan interpretar la información de forma correcta.

El elemento sobre el que nos podemos apoyar es el atributo header. El atributo header relaciona una celda con una celda de la cabecera, para poder establecer esta relación semántica.

Partamos de la siguiente tabla...

Nombre	Edad	Localidad
Víctor	38	Madrid
Esther	25	Salamanca

Para ello lo primero que hay que hacer es darle un atributo id a las celdas de cabecera.

```
<tr>
  <th id="nombre">Nombre</th>
  <th id="edad">Edad</th>
  <th id="localidad">Localidad</th>
</tr>
```

Ahora, para cada celda deberemos de asociar el identificador, atributo id, de la cabecera que les aplique en el atributo headers.

```
<tr>
  <th headers="nombre">Víctor</th>
  <th headers="edad">38</th>
  <th headers="localidad">Madrid</th>
</tr>
<tr>
  <th headers="nombre">Esther</th>
  <th headers="edad">25</th>
  <th headers="localidad">Salamanca</th>
</tr>
```

Así, el agente de usuario no visual, cuando vaya leyendo la fila hará lo siguiente:

```
"Nombre, Víctor. Edad, 38. Localidad, Madrid.
Nombre, Esther. Edad, 25. Localidad, Salamanca."
```

Categorizar Celdas

Otra de las cosas que podemos hacer para los agentes de usuario no visuales es categorizar las celdas. En HTML tenemos un atributo que es axis, de esta manera podemos establecer ejes de agrupación.

El **atributo axis** aplica a las celdas td y celdas de cabecera th. Y permite darle una categoría textual.

La estructura sería:

```
<td axis="categoria">...</td>
```

	Comida	Hotel	Transporte
Madrid			
1 de marzo	15	120	-
2 de marzo	18	120	34
3 de marzo	25	120	-
Ávila			
4 de marzo	10	75	12
5 de marzo	12	75	14

En esta tabla podemos establecer que haya 3 tipos de categorías. Los gastos (comida, hotel y transporte), las ciudades (Madrid y Ávila) y las fechas.

Así que deberemos de marcar esas celdas con la categoría correspondiente, mediante el atributo axis. El código en HTML nos quedará de la siguiente forma:

```
<table>
  <tr>
    <th></th>
    <th axis="gasto">Comida</th>
    <th axis="gasto">Hotel</th>
    <th axis="gasto">Transporte</th>
  </tr>
  <tr>
    <td axis="ciudad">Madrid</td>
    <td></td>
    <td></td>
    <td></td>
  </tr>
  <tr>
    <td axis="fecha">1 de marzo</td>
    <td>15</td>
    <td>120</td>
    <td>-</td>
  </tr>
  <tr>
    <td axis="fecha">2 de marzo</td>
    <td>18</td>
    <td>120</td>
    <td>34</td>
  </tr>
  <tr>
    <td axis="fecha">3 de marzo</td>
    <td>25</td>
    <td>120</td>
    <td>-</td>
  </tr>
  <tr>
    <td axis="ciudad">Avila</td>
    <td></td>
```

```
    <td></td>
    <td></td>
  </tr>
  <tr>
    <td axis="fecha">4 de marzo</td>
    <td>10</td>
    <td>75</td>
    <td>12</td>
  </tr>
  <tr>
    <td axis="fecha">5 de marzo</td>
    <td>12</td>
    <td>75</td>
    <td>14</td>
  </tr>
</table>
```

El atributo axis se combina con el atributo headers para poder dar suficiente información a los agentes de usuario no visuales sobre las tablas.

Anchos de las tablas y columnas

Aunque el formato de las tablas, tanto para la tabla en sí, como para sus filas y celdas se hará mediante hojas de estilo CSS, tenemos dos formas básicas de poder establecer el ancho de la tabla y el ancho de las columnas.

En primer lugar podemos utilizar el atributo width del elemento table.

```
<table width="ancho">...</table>
```

De esta forma podremos establecer en cualquier medida el ancho que queremos que ocupe la tabla. Ya sean px, em, tantos por ciento,...

Por ejemplo podemos definir que ocupe todo el ancho de una página asignándole un valor de width del 100%.

```
<table width="100%">...</table>
```

Para el caso de las columnas ya hemos visto que tanto los elementos colgroup como col también tenían el atributo width. Así que serán con estos elemento con los que de forma básica podamos establecer el ancho de una columna.

De esta forma si tenemos 4 columnas y queremos que se distribuyan de forma igual, podríamos escribir el siguiente código.

```
<table>
  <colgroup span="4" width="25%">
```

```
    ....
</table>
```