



ELEARNING TOTAL

Programador Web / Nivel 1 – Unidad 2

# Programador Web – Nivel 1

Unidad 2: Introducción a HTML





## Indice

### Unidad 1: Introducción a HTML

Texto

Secciones

Enlaces

Listas

Imágenes

Tablas



## Objetivos

### Que el alumno logre:

- Reconocer e implemente las diferentes etiquetas de HTML.





# Etiquetas principales

## TEXTO

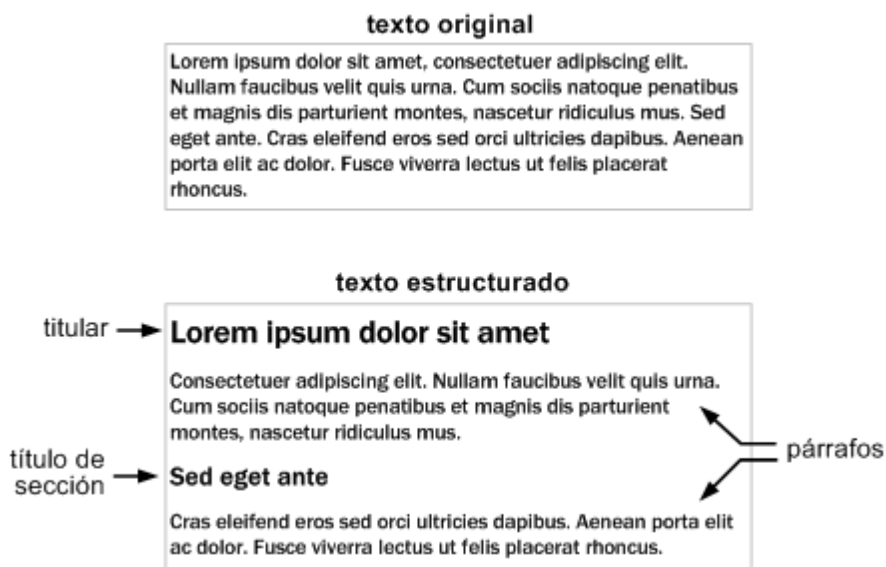
Gran parte del contenido de las páginas HTML habituales está formado por texto.

Por este motivo, es muy importante conocer los elementos y etiquetas que define HTML para el manejo del texto.

El lenguaje HTML incorpora al tratamiento del texto muchas de las ideas y normas establecidas en otros entornos de publicación de contenidos.

De esta forma, HTML define etiquetas para estructurar el contenido en secciones y párrafos y define otras etiquetas para marcar elementos importantes dentro del texto.

La tarea inicial del editor de contenidos HTML consiste en estructurar el texto original definiendo sus párrafos, titulares y títulos de sección, como se muestra en la siguiente imagen:





El anterior ejemplo muestra la transformación de un párrafo con un texto simple en un párrafo cuyo texto contiene elementos marcados de forma especial. Así, algunas palabras del texto se muestran en negrita porque se consideran importantes; otras palabras aparecen en cursiva, ya que se han marcado como destacadas e incluso una frase aparece tabulada y entre comillas, indicando que es una cita textual de otro contenido.

En las secciones siguientes se muestran todas las etiquetas que define HTML para estructurar y marcar el texto. Además, se hace una mención especial al tratamiento que hace HTML de los espacios en blanco y las nuevas líneas.

La forma más sencilla de estructurar un texto consiste en separarlo por párrafos. Además, HTML permite incluir títulos que delimitan cada una de las secciones.

### **Párrafos**

Una de las etiquetas más utilizadas de HTML es la etiqueta `<p>`, que permite definir los párrafos que forman el texto de una página.

Para delimitar el texto de un párrafo, se encierra ese texto con la etiqueta `<p>`, como muestra el siguiente ejemplo:

`<p>`Este es el texto que forma el primer párrafo de la página.

Los párrafos pueden ocupar varias líneas y el navegador se encarga de ajustar su longitud al tamaño de la ventana. `</p>`

`<p>`El segundo párrafo de la página también se define encerrando su texto con la etiqueta `p`. El navegador también se encarga de

separar automáticamente cada párrafo. `</p>`

Los párrafos creados con HTML son elementos de bloque, por lo que siempre ocupan toda la anchura de la ventana del navegador. Además, no tienen atributos específicos, pero sí que se les pueden asignar los atributos comunes de HTML básicos, de internacionalización y de eventos.



## Secciones

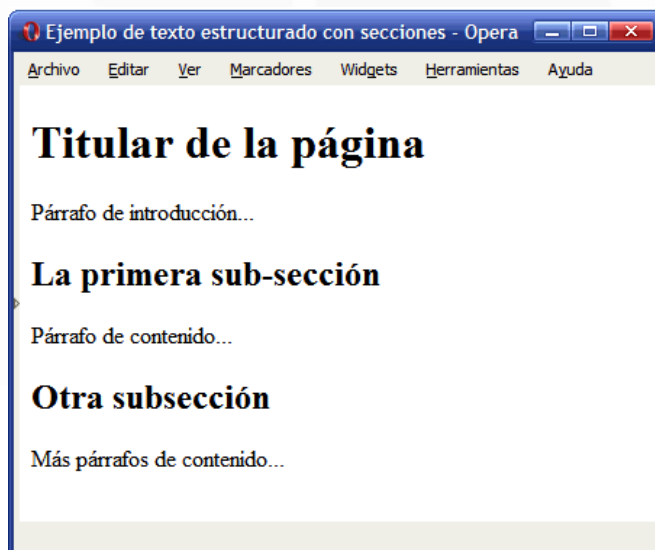
Las páginas HTML habituales suelen tener una estructura más compleja que la que se puede crear solamente mediante párrafos. De hecho, es habitual que las páginas se dividan en diferentes secciones jerárquicas.

Los títulos de sección se utilizan para delimitar el comienzo de cada sección de la página. HTML permite crear secciones de hasta seis niveles de importancia. De esta forma, aunque una página puede definir cualquier número de secciones, sólo puede incluir seis niveles jerárquicos.

Las etiquetas que definen los títulos de sección son `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>` y `<h6>`. La etiqueta `<h1>` es la de mayor importancia y por tanto se utiliza para definir los titulares de la página. La importancia del resto de etiquetas es descendiente, de forma que la etiqueta `<h6>` es la que se utiliza para delimitar las secciones menos importantes de la página.

Las etiquetas `<h1>`, ..., `<h6>` definen títulos de sección, no secciones completas. Por este motivo, no es necesario encerrar los contenidos de una sección con su etiqueta correspondiente. Solamente se debe encerrar con las etiquetas `<h1>`, ..., `<h6>` los títulos de cada sección.

El siguiente ejemplo muestra el uso de las etiquetas de título de sección (ver ejemplo4.html):





`<h1>`Titular de la página`</h1>`

`<p>`Párrafo de introducción...`</p>`

`<h2>`La primera sub-sección`</h2>`

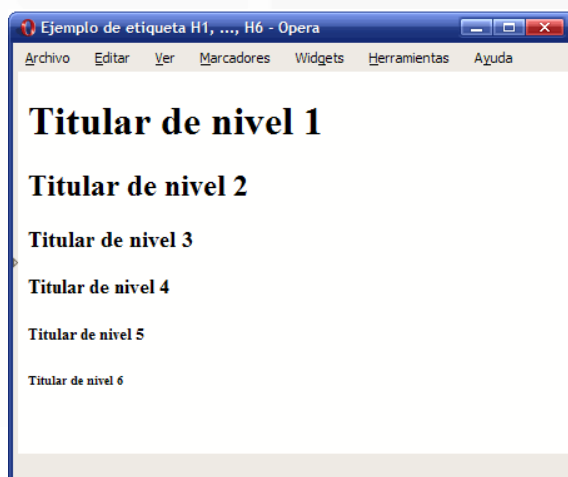
`<p>`Párrafo de contenido...`</p>`

`<h2>`Otra subsección`</h2>`

`<p>`Más párrafos de contenido...`</p>`

Los navegadores asignan de forma automáticamente el tamaño del título de cada sección en función de su importancia. Así, los títulos de sección `<h1>` se muestran con el tamaño de letra más grande, ya que son el nivel jerárquico superior, mientras que los títulos de sección `<h6>` se visualizan con un tamaño de letra muy pequeño, adecuado para el nivel jerárquico de menor importancia.

Evidentemente, el aspecto que los navegadores aplican por defecto a los títulos de sección se puede modificar utilizando las hojas de estilos de CSS. La siguiente imagen muestra el tamaño por defecto con el que los navegadores muestran cada titular:





### *Marcado básico de texto*

Una vez estructurado el texto en párrafos y secciones, el siguiente paso es el marcado de los elementos que componen el texto.

Los textos habituales están formados por elementos como palabras en negrita o cursiva, anotaciones y correcciones, citas a otros documentos externos, etc.

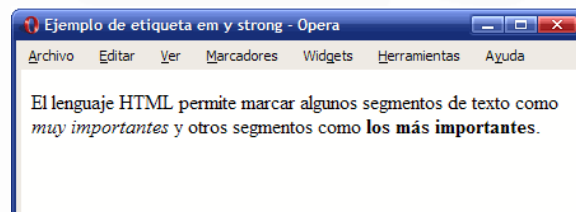
HTML proporciona varias etiquetas para marcar cada uno de los diferentes tipos de texto.

Entre las etiquetas más utilizadas para marcar texto se encuentran `<em>` y `<strong>`.

La etiqueta `<em>` marca un texto indicando que su importancia es mayor que la del resto del texto. La etiqueta `<strong>` indica que un determinado texto es de la mayor importancia dentro de la página.

`<p>`El lenguaje HTML permite marcar algunos segmentos de texto como `<em>`muy importantes`</em>` y otros segmentos como `<strong>`los más importantes`</strong>`.`</p>`

Por defecto, los navegadores muestran los elementos `<em>` en cursiva para hacer evidente su importancia y muestran los elementos `<strong>` en negrita, para indicar que son los más importantes:







### *Espacios en blanco y nuevas líneas*

El aspecto más sorprendente del lenguaje HTML cuando se desarrollan los primeros documentos es el tratamiento especial de los “**espacios en blanco**” del texto. HTML considera espacio en blanco a los espacios en blanco, los tabuladores, los retornos de carro y el carácter de nueva línea (ENTER o Intro).

`<p>Este primer párrafo no contiene saltos de línea ni otro tipo de espaciado.</p>`

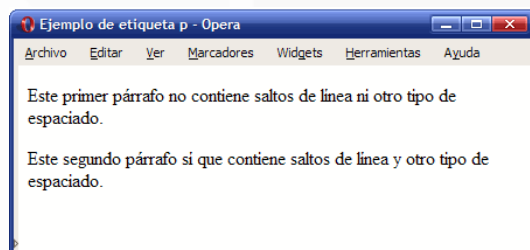
`<p>Este segundo párrafo sí que contiene saltos`

`de`

`línea`

`y otro tipo de espaciado.</p>`

El anterior código HTML se visualiza en cualquier navegador de la siguiente manera:



Los dos párrafos de la imagen anterior se ven idénticos, aunque el segundo párrafo incluye varios espacios en blanco y está escrito en varias líneas diferentes. La razón de este comportamiento es que HTML ignora todos los espacios en blanco sobrantes, es decir, todos los espacios en blanco que no son el espacio en blanco que separa las palabras.

No obstante, HTML proporciona varias alternativas para poder incluir tantos espacios en blanco y tantas nuevas líneas como sean necesarias dentro del contenido textual de las páginas.



## Nuevas líneas

Para incluir una nueva línea en un punto y forzar a que el texto que sigue se muestre en la línea inferior, se utiliza la etiqueta `<br>`. En cierta manera, insertar la etiqueta `<br>` en un determinado punto del texto equivale a presionar la tecla ENTER (o Intro) en ese mismo punto.

La etiqueta `<br>` es una de las pocas etiquetas especiales de HTML. La particularidad de `<br>` es que es una etiqueta vacía, es decir, no encierra ningún texto. De esta forma, la etiqueta debe abrirse y cerrarse de forma consecutiva: `<br></br>`.

En estos casos, HTML permite utilizar un atajo para indicar que una etiqueta se está abriendo y cerrando de forma consecutiva: `<br/>` (también se puede escribir como `<br />`).

Utilizando la etiqueta `<br>` se puede rehacer el ejemplo anterior para que respete las líneas que forman el segundo párrafo:

`<p>Este primer párrafo no contiene saltos de línea ni otro tipo de espaciado.</p>`

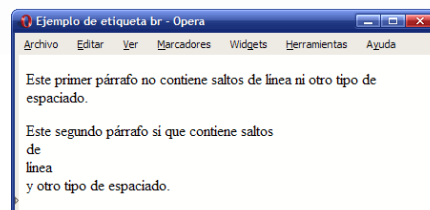
`<p>Este segundo párrafo sí que contiene saltos <br/>`

de `<br/>`

línea `<br/>`

y otro tipo de espaciado.</p>

El navegador ahora sí que muestra correctamente las nuevas líneas que se querían insertar:





## ENLACES

El lenguaje de marcado HTML se definió teniendo en cuenta algunas de las características que existían en ese momento para la publicación digital de contenidos. Entre los conceptos utilizados en su creación, se encuentra el mecanismo de “hipertexto”.

De hecho, las letras “HT” de la sigla HTML significan “hipertexto” (hypertext en inglés), por lo que el significado completo de HTML podría traducirse como “lenguaje de marcado para hipertexto”.

La incorporación del hipertexto fue una de las claves del éxito del lenguaje HTML, ya que permitió crear documentos interactivos que proporcionan información adicional cuando se solicita. El elemento principal del hipertexto es el “hiperenlace”, también llamado “enlace web” o simplemente “enlace”.

Los enlaces se utilizan para establecer relaciones entre dos recursos. Aunque la mayoría de enlaces relacionan páginas web, también es posible enlazar otros recursos como imágenes, documentos y archivos.

Una característica que no se suele tener en cuenta en los enlaces es que están formados por dos extremos y un sentido. En otras palabras, el enlace comienza en un recurso y apunta hacia otro recurso. Cada uno de los dos extremos se llaman “anchors” en inglés, que se puede traducir literalmente como “anclas”.

### URL

Antes de empezar a crear enlaces, es necesario comprender y dominar el concepto de URL.

El acrónimo URL (del inglés Uniform Resource Locator) hace referencia al identificador único de cada recurso disponible en Internet. Las URL son esenciales para crear los enlaces, pero también se utilizan en otros elementos HTML como las imágenes y los formularios.

La URL de un recurso tiene dos objetivos principales:

- .: Identificar de forma única a ese recurso
- .: Permitir localizar de forma eficiente ese recurso

En primer lugar, las URL permiten que cada página HTML publicada en Internet tenga un nombre único que permita diferenciarla de las demás. De esta forma es posible crear enlaces que apunten de forma inequívoca a una determinada página.



La cadena de texto <http://www.google.com> es la URL completa de la página principal de Google. La URL de las páginas es imprescindible para crear los enlaces, ya que permite distinguir una página de otra.

El segundo objetivo de las URL es el de permitir la localización eficiente de cada recurso de Internet. Para ello es necesario comprender las diferentes partes que forman las URL.

Las partes que componen la URL anterior son:

**.: Protocolo (<http://>):** el mecanismo que debe utilizar el navegador para acceder a ese recurso. Todas las páginas web utilizan <http://>. Las páginas web seguras (por ejemplo las de los bancos y las de los servicios de email) utilizan <https://> (se añade una letra s).

**.: Servidor ([www.frre.utn.edu.ar/](http://www.frre.utn.edu.ar/)):** simplificando mucho su explicación, se trata del ordenador en el que se encuentra guardada la página que se quiere acceder. Los navegadores son capaces de obtener la dirección de cada servidor a partir de su nombre.

**.: Ruta ([/home.html](http://home.html)):** camino que se debe seguir, una vez que se ha llegado al servidor, para localizar el recurso específico que se quiere acceder.

Por tanto, las URL no sólo identifican de forma única a cada recurso de Internet, sino que también proporcionan a los navegadores la información necesaria para poder llegar hasta ese recurso.

La mayoría de URL son tan sencillas como la URL mostrada anteriormente. No obstante, existen URL complejas formadas por más partes.

<http://www.elearning-total.com/mod/forum/view.php?f=7#42>

Las cinco partes que forman la URL anterior son:

**.: Protocolo (<http://>)**

**.: Servidor ([www.elearning-total.com](http://www.elearning-total.com))**

**.: Ruta ([/mod/forum/view.php](http://mod/forum/view.php))**

**.: Consulta ([?f=7](http://?f=7)):** información adicional necesaria para que el servidor localice correctamente el recurso que se quiere acceder. Siempre comienza con el carácter ? y contiene una sucesión de palabras separadas por = y &



∴ **Sección (#42)**: permite que el navegador se posicione automáticamente en una sección de la página web. Siempre comienza con el carácter #

Como las URL utilizan los caracteres :, =, & y / para separar sus partes, estos caracteres están reservados y no se pueden utilizar libremente. Además, algunos caracteres no están reservados pero pueden ser problemáticos si se utilizan en la propia URL.

Si es necesario incluir estos caracteres reservados y especiales en una URL, se sustituyen por combinaciones de caracteres seguros. Esta sustitución se denomina codificación de caracteres y el servidor realiza el proceso inverso (decodificación) cuando le llega una URL con los caracteres codificados.

### *Enlaces relativos y absolutos*

Las páginas web habituales suelen contener decenas de enlaces de diferentes tipos.

En una página, cuando se pincha sobre algunos enlaces, el navegador abandona el sitio web para acceder a páginas que se encuentran en otros sitios.

Estos enlaces se conocen como “enlaces externos”. Sin embargo, la mayoría de enlaces de un sitio web apuntan a páginas del propio sitio web, por lo que se denominan “enlaces internos”.

Además de internos/externos, la otra característica que diferencia a los enlaces (y por tanto, también a las URL) es si el enlace es absoluto o relativo.

Las URL absolutas incluyen todas las partes de la URL (protocolo, servidor y ruta) por lo que no se necesita más información para obtener el recurso enlazado.

Las URL relativas prescinden de algunas partes de las URL para hacerlas más breves. Como se trata de URL incompletas, es necesario disponer de información adicional para obtener el recurso enlazado. En concreto, para que una URL relativa sea útil es imprescindible conocer la URL del origen del enlace.

Las URL relativas se construyen a partir de las URL absolutas y prescinden de la parte del protocolo, del nombre del servidor e incluso de parte o toda la ruta del recurso enlazado. Aunque las URL relativas pueden ser difíciles de entender para los que comienzan con HTML, son tan útiles que todos los sitios web las utilizan.

Imagina que dispones de una página publicada en <http://www.ejemplo.com/ruta1/ruta2/pagina1.html> y quieres incluir en ella un enlace a otra página que se encuentra en <http://www.ejemplo.com/ruta1/ruta2/pagina2.html>. Como las URL identifican de forma única a los recursos



de Internet y proporcionan la información necesaria para llegar hasta ellos, el enlace debería utilizar la URL completa de la segunda página.

Las URL completas también se llaman URL absolutas, ya que el navegador no necesita disponer de información adicional para localizar el recurso enlazado. Si se utilizan siempre las URL absolutas, los enlaces están completamente definidos.

Sin embargo, escribir siempre las URL completas es bastante aburrido, cuesta mucho tiempo y hace imposible cambiar la ubicación de los contenidos de un sitio web. Por ese motivo, casi todos los sitios web de Internet utilizan URL relativas siempre que es posible.

**Una URL relativa es una versión abreviada de una URL absoluta.** Su objetivo es eliminar todas las partes de la URL absoluta que se pueden adivinar a partir de la información de contexto de la página web. En otras palabras, las URL relativas aprovechan la inteligencia de los navegadores para crear URL incompletas que los navegadores pueden completar deduciendo la información que falta

Considerando de nuevo el ejemplo anterior, la URL a la que se quiere enlazar utiliza el mismo protocolo y se encuentra en el mismo servidor que la página origen, por lo que la URL relativa puede prescindir de esas partes:

**URL absoluta:** <http://www.ejemplo.com/ruta1/ruta2/pagina2.html>

**URL relativa:** </ruta1/ruta2/pagina2.html>

En el ejemplo anterior, las dos URL son equivalentes porque cuando no se indica el protocolo y el servidor de una URL, los navegadores suponen que son los mismos que los de la página origen. Por lo tanto, cuando el navegador encuentra la URL </ruta1/ruta2/pagina2.html>, realiza el siguiente proceso:

1. La URL no es absoluta, por lo que se debe determinar la URL absoluta a partir de la URL relativa para poder cargar el recurso enlazado.
2. A la URL relativa le falta el protocolo y el servidor, por lo que se supone que son los mismos que los de la página origen (<http://> y [www.ejemplo.com](http://www.ejemplo.com)).



3. Se añaden las partes que faltan a la URL relativa para obtener la URL absoluta: `http:// + www.ejemplo.com + /ruta1/ruta2/pagina2.html = http://www.ejemplo.com/ruta1/ruta2/pagina2.html`.

Aunque el ejemplo mostrado es el caso más sencillo de URL relativa, existen otros casos más avanzados en los que se prescinde de parte o toda la ruta del recurso que se enlaza. A continuación se muestran los cuatro tipos diferentes de URL relativas:

#### *1) El origen y el destino del enlace se encuentran en el mismo directorio*

Si desde una página web se quiere enlazar un recurso que se encuentra en el mismo directorio del servidor, la URL relativa puede prescindir de todas las partes de la URL absoluta salvo el nombre del recurso enlazado.

**URL absoluta:** `http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina2.html`

**URL relativa:** `pagina2.html`

Cuando el navegador encuentra una URL relativa que sólo consiste en el nombre de un recurso, supone que el protocolo, servidor y directorio del recurso enlazado son los mismos que los del origen del enlace.

#### *2) El destino del enlace se encuentra cerca de su origen y en un nivel superior*

En este caso, el recurso que se enlaza no está en el mismo directorio que el origen del enlace pero sí que está cerca y en algún directorio superior. La URL relativa debe indicar de alguna manera que es necesario subir un nivel en la jerarquía de directorios para llegar hasta el recurso.

Para indicar al navegador que debe subir un nivel, se incluyen dos puntos y una barra (`../`) en la ruta del recurso enlazado. De esta forma, cada vez que aparece `../` en una URL relativa, significa que se debe subir un nivel.

**URL absoluta:** `http://www.ejemplo.com/ruta1/ruta2/pagina2.html`

**URL relativa:** `../pagina2.html`

Cuando el navegador encuentra la URL relativa `../pagina2.html`, sabe que para encontrar el recurso enlazado (`pagina2.html`) tiene que subir un nivel desde el lugar en el que se encuentra esa URL relativa. La página que incluye esa URL se encuentra en el directorio `ruta1/ruta2/ruta3`, por lo que subir un nivel equivale entrar en el directorio `ruta1/ruta2`.



### *3) El destino del enlace se encuentra cerca de su origen y en un nivel inferior*

Este caso es muy similar al anterior, pero más sencillo. Si el recurso enlazado se encuentra en algún directorio inferior al que se encuentra el origen, sólo es necesario indicar el nombre de los directorios a los que debe entrar el navegador.

**URL absoluta:** `http://www.ejemplo.com/ruta1/ruta2/ruta3/ruta4/pagina2.html`

**URL relativa:** `ruta4/pagina2.html`

### *4) El origen y el destino del enlace se encuentran muy alejados*

Cuando el origen y el destino de un enlace se encuentran muy alejados (pero en el mismo servidor) las URL relativas se pueden complicar en exceso.

Aunque es posible utilizar `../` para subir por la jerarquía de directorios y se puede entrar en cualquier directorio indicando su nombre, las URL relativas que se obtienen son demasiado largas y complicadas.

En estos casos, lo más sencillo es indicar la ruta completa hasta el recurso enlazado comenzando desde la raíz del servidor web. Por lo tanto, estas URL relativas sólo omiten el protocolo y el nombre del servidor.

**URL absoluta:** `http://www.ejemplo.com/ruta7/pagina2.html`

**URL relativa:** `/ruta7/pagina2.html`

### *Enlaces básicos*

Los enlaces en HTML se crean mediante la etiqueta `<a>` (su nombre viene del inglés “anchor”, literalmente traducido como “ancla”).

El atributo más importante de la etiqueta `<a>` es `href`, que se utiliza para indicar la URL a la que apunta el enlace. Cuando el usuario pincha sobre un enlace, el navegador se dirige a la URL del recurso indicado mediante `href`. Las URL de los enlaces pueden ser absolutas, relativas, internas y externas.

Con la definición anterior, para crear un enlace que apunte a la página principal de Google solamente habría que incluir lo siguiente en un documento HTML:

```
<a href="http://www.google.com">Página principal de Google</a>
```

Como el atributo `href` indica una URL, un enlace puede apuntar a cualquier tipo de recurso al que pueda acceder el navegador. El siguiente enlace apunta a una imagen, que se mostrará en el navegador cuando el usuario pinche sobre el enlace:





`<a href="http://www.ejemplo.com/fondo_escritorio.jpg">Imagen interesante para un fondo de escritorio</a>`

De la misma forma, los enlaces pueden apuntar directamente a documentos PDF, Word, etc.

`<a href="http://www.ejemplo.com/informe.pdf">Descargar informe completo [PDF]</a>`

## Listas

En ocasiones, es posible agrupar determinadas palabras o frases en un conjunto de elementos que tienen más significado de forma conjunta.

El lenguaje HTML define tres tipos diferentes de listas para agrupar los elementos: listas no ordenadas (se trata de una colección simple de elementos en la que no importa su orden), listas ordenadas (similar a la anterior, pero los elementos están numerados y por tanto, importa su orden) y listas de definición (un conjunto de términos y definiciones similar a un diccionario).

### Listas no ordenadas

Las listas no ordenadas son las más sencillas y las que más se utilizan. Una lista no ordenada es un conjunto de elementos relacionados entre sí pero para los que no se indica un orden o secuencia determinados. La etiqueta `<ul>` encierra todos los elementos de la lista y la etiqueta `<li>` cada uno de sus elementos.

El siguiente código HTML muestra un ejemplo sencillo de lista no ordenada:

```
<h1>Menú</h1>
```

```
<ul>
```

```
<li>Inicio</li>
```

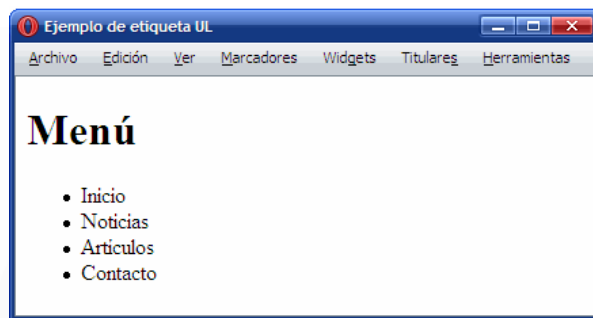
```
<li>Noticias</li>
```



```
<li>Artículos</li>
```

```
<li>Contacto</li>
```

```
</ul>
```



## Listas ordenadas

Las listas ordenadas son casi idénticas a las listas no ordenadas, salvo que en este caso los elementos relacionados se muestran siguiendo un orden determinado. Cuando se crea por ejemplo una lista con las instrucciones de un producto, es importante el orden en el que se realiza cada paso. Cuando se muestra un índice o tabla de contenidos en un libro, es importante el orden de cada elemento del índice.

En todos estos casos, la lista más adecuada es la lista ordenada, que se define mediante la etiqueta `<ol>`. Los elementos de la lista se definen mediante la etiqueta `<li>`, la misma que se utiliza en las listas no ordenadas.

```
<h1>Instrucciones</h1>
```

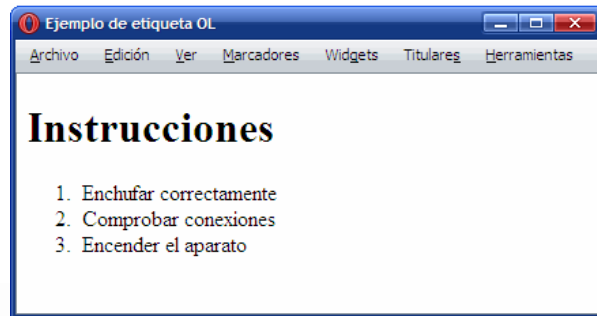
```
<ol>
```

```
<li>Enchufar correctamente</li>
```

```
<li>Comprobar conexiones</li>
```

```
<li>Encender el aparato</li>
```

```
</ol>
```



## Listas de definición

Las listas de definición apenas se utilizan en la mayoría de páginas HTML. Su funcionamiento es similar al de un diccionario, ya que cada elemento de la lista está formado por términos y definiciones. La etiqueta `<dl>` crea la lista de definición y las etiquetas `<dt>` y `<dd>` definen respectivamente el término y la descripción de cada elemento de la lista.



## IMÁGENES

Las imágenes son uno de los elementos más importantes de las páginas web. De hecho, prácticamente todas las páginas web contienen alguna imagen y la mayoría incluyen decenas de imágenes. Dentro de las imágenes que se pueden incluir en una página HTML se deben distinguir dos tipos: las imágenes de contenido y las imágenes de adorno.

Las imágenes de contenido son las que proporcionan información y complementan la información textual. Las imágenes de adorno son las que se utilizan para hacer bordes redondeados, para mostrar pequeños iconos en las listas de elementos, para mostrar fondos de página, etc. Las imágenes de contenido se incluyen directamente en el código HTML mediante la etiqueta `<img>` y las imágenes de adorno no se deberían incluir en el código HTML, sino que deberían emplearse hojas de estilos CSS para mostrarlas.

### Atributos específicos

.: `src = "url"` - Indica la URL de la imagen que se muestra

.: `alt = "texto"` - Descripción corta de la imagen

.: `longdesc = "url"` - Indica una URL en la que puede encontrarse una descripción más detallada de la imagen

.: `name = "texto"` - Nombre del elemento imagen

.: `height = "unidad_de_medida"` - Indica la altura con la que se debe mostrar la imagen (no es obligatorio que coincida con la altura original de la imagen)

.: `width = "unidad_de_medida"` - Indica la anchura con la que se debe mostrar la imagen (no es obligatorio que coincida con la anchura original de la imagen)

Los dos atributos requeridos son `src` y `alt`. El atributo `src` es similar al atributo `href` de los enlaces, ya que establece la URL de la imagen que se va a mostrar en la página. Las URL indicadas pueden ser absolutas o relativas.



El atributo alt permite describir el contenido de la imagen mediante un texto breve. Las descripciones deben tener una longitud inferior a 1024 caracteres y son útiles para las personas y dispositivos discapacitados que no pueden acceder a las imágenes.

Ejemplo sencillo para incluir una imagen:

```

```

Como <img> es una etiqueta vacía, no tiene etiqueta de cierre. No obstante, para que la página XHTML sea válida, todas las etiquetas deben estar cerradas. Como ya se explicó anteriormente, para cerrar una etiqueta vacía se incluyen los caracteres /> al final de la etiqueta.

HTML no impone ninguna restricción sobre el formato gráfico que se puede utilizar en las imágenes, por lo que en principio la etiqueta <img> puede incluir cualquier formato gráfico existente. Sin embargo, si la imagen utiliza un formato poco habitual, todos o algunos navegadores no serán capaces de mostrar esa imagen.

La recomendación es utilizar uno de los tres siguientes formatos gráficos que entienden todos los navegadores modernos: GIF, JPG y PNG. El formato PNG presenta el inconveniente de que los navegadores obsoletos como Internet Explorer 6 no muestran correctamente las imágenes con transparencias de 24 bits.

Si el valor del atributo width o height se indica mediante un número entero, el navegador supone que hace referencia a la unidad de medida píxel. Por tanto, en el ejemplo anterior, la primera foto se muestra con una anchura de 200 píxel y una altura de 350 píxel.

También es posible indicar la anchura y altura en forma de porcentaje. En este caso, el porcentaje hace referencia a la altura/anchura del elemento en el que está contenida la imagen. Si la imagen no se encuentra dentro de ningún otro elemento, hace referencia a la anchura/altura total de la página.



## TABLAS

Desde sus primeras versiones, HTML incluyó el soporte para crear tablas de datos en las páginas web. Además de ser sencillo, el modelo definido por HTML es muy flexible y bastante completo.

Las tablas en HTML utilizan los mismos conceptos de filas, columnas, cabeceras y títulos que los que se utilizan en cualquier otro entorno de publicación de documentos:

Nombre	Horas	Plazas	Horario
Introducción a XHTML	20	20	09:00 – 13:00
CSS avanzado	40	15	16:00 – 20:00
Taller de usabilidad	40	10	16:00 – 20:00
Introducción a AJAX	60	20	08:30 – 12:30

Las tablas de HTML pueden contener elementos simples, agrupaciones de filas y de columnas, cabeceras y pies de tabla, subdivisiones, cabeceras múltiples y otros elementos complejos.

A pesar de que las tablas HTML son fáciles de comprender y utilizar, son uno de los elementos más polémicos de HTML. El problema de las tablas es que no siempre se utilizan adecuadamente. Aunque parezca obvio, las tablas se deben utilizar para mostrar información tabular.

Hasta hace unos años, las tablas también se utilizaban para definir la estructura de las páginas web. La cabecera de la página era una fila de una gran tabla, el pie de página era otra fila de esta tabla y la zona de contenidos estaba formada por varias columnas dentro de esa gran tabla.



## Tablas básicas

Las tablas más sencillas de HTML se definen con tres etiquetas: `<table>` para crear la tabla, `<tr>` para crear cada fila y `<td>` para crear cada columna.

A continuación se muestra el código HTML de una tabla sencilla:

```
<h1>Listado de cursos</h1>

<table>

<tr>

<td><h3>Curso</h3></td>

<td><h3>Horas</h3></td>

<td><h3>Horario</h3></td>

</tr>

<tr>

<td>CSS</td>

<td>20</td>

<td>16:00 - 20:00</td>

</tr>

<tr>

<td>HTML</td>

<td>20</td>

<td>16:00 - 20:00</td>

</tr>

<tr>

<td>Dreamweaver</td>

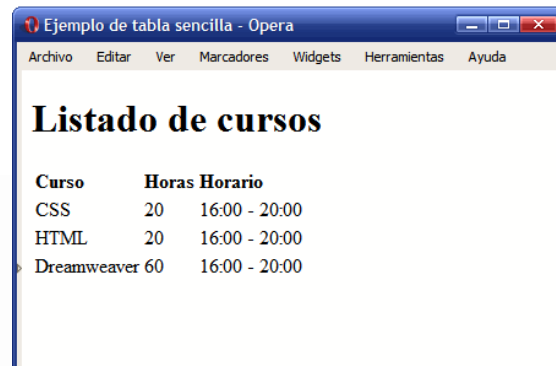
<td>60</td>

<td>16:00 - 20:00</td>

</tr>
```



`</table>`



Curso	Horas	Horario
CSS	20	16:00 - 20:00
HTML	20	16:00 - 20:00
Dreamweaver	60	16:00 - 20:00

Si se visualiza el código anterior en cualquier navegador, se obtiene una tabla como la que muestra la siguiente imagen:

La etiqueta `<table>` encierra todas las filas y columnas de la tabla. Las etiquetas `<tr>` (del inglés “table row”) definen cada fila de la tabla y encierran todas las columnas.

Por último, la etiqueta `<td>` (del inglés “table data cell”) define cada una de las columnas de las filas, aunque realmente HTML no define columnas sino celdas de datos.

Al definir una tabla, se debe pensar en primer lugar en las filas que la forman y a continuación en las columnas. El motivo es que HTML procesa primero las filas y por eso las etiquetas `<tr>` aparecen antes que las etiquetas `<td>`.

De todos los atributos disponibles para las celdas, los más utilizados son `rowspan` y `colspan`, que se emplean para construir tablas complejas como las que se ven más adelante. Entre los demás atributos, sólo se utiliza de forma habitual el atributo `scope`, sobre todo con las celdas de cabecera que se ven a continuación.

Normalmente, algunas de las celdas de la tabla se utilizan como cabecera de las demás celdas de la fila o de la columna. En este caso, HTML define la etiqueta `<th>` (del inglés “table header cell”) para indicar que una celda es cabecera de otras celdas.

Las tablas complejas suelen disponer de una estructura irregular que junta varias columnas para formar una columna ancha o une varias filas para formar una fila más alta que las demás. Para fusionar filas o columnas, se utilizan los atributos `rowspan` y `colspan` respectivamente.





La siguiente imagen muestra una tabla compleja que ha fusionado dos columnas simples para formar una columna más ancha:

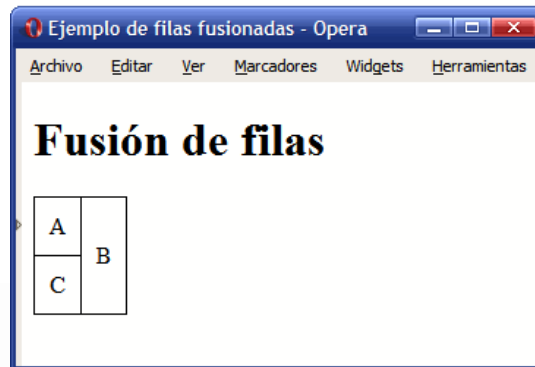
A	
B	C

Para obtener una tabla como la de la imagen anterior, se debe utilizar el siguiente código:

```
<table>
<tr>
<td colspan="2">A</td>
</tr>
<tr>
<td>B</td>
<td>C</td>
</tr>
</table>
```

La primera fila de la tabla está formada sólo por una columna, mientras que la segunda fila está formada por dos columnas.

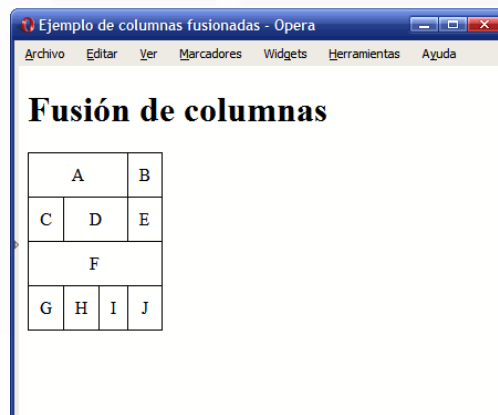
De forma equivalente, si se quiere diseñar una tabla HTML que fusiona filas como la de la siguiente imagen:



A	B
C	

```
<table>
<tr>
<td>A</td>
<td rowspan="2">B</td>
</tr>
<tr>
<td>C</td>
</tr></table>
```

Utilizando los atributos rowspan y colspan, es posible diseñar tablas tan complejas como las que se muestran en los siguientes ejemplos.



Ejemplo de columnas fusionadas - Opera				—	□	✕
Archivo	Editar	Ver	Marcadores	Widgets	Herramientas	Ayuda
<h1>Fusión de columnas</h1>						
A		B				
C	D	E				
F						
G	H	I	J			

El código HTML necesario para crear esta tabla es:



```
<table border="1"><tr>
<td colspan="3">A</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td colspan="2">D</td>
<td>E</td>
</tr>
<tr>
<td colspan="4">F</td>
</tr>
<tr>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
</tr>
</table>
```

Las tablas pueden ser formateadas a partir de los atributos que nos ofrece la etiqueta <table>. Estos atributos son:

**align**

Alinea horizontalmente la tabla con respecto a su entorno. (El mismo atributo aplicado sobre la etiqueta td (fila) alinea el contenido de la celda).

Los valores posibles son:

- "right"
- "left"
- "center"

Ejemplo: <table align="center">

**background** Nos permite colocar un fondo para la tabla a partir de un enlace a una imagen.



Ejemplo: `<table background="imagen.jpg">`

**bgcolor** Da color de fondo a la tabla.

Los colores en html se manejan a través del código de color hexadecimal (profundizaremos sobre esto en futuras unidades)

Por ejemplo: `<table bgcolor="#ff0000">` o `<table bgcolor="red">`

**border** Define el número de pixeles del borde principal.

Las tablas por defecto no muestran el borde entre una celda y la otra, para hacer visible esa propiedad, debemos utilizar el atributo border.

Ejemplo: `<table border="1">`

**bordercolor** Define el color del borde.

Una vez definido el atributo de borde, podemos darle color utilizando el atributo bordercolor.

Ejemplo: `<table border="1" bordercolor="blue">`

**cellpadding** Define, en pixeles, el espacio entre los bordes de la celda y el contenido de la misma.

Esta separación es uniforme en toda la tabla, es decir, no se puede definir un cellpadding diferente para cada separación interna de celda dentro de la tabla.

Ejemplo: `<table cellpadding="2">`

**cellspacing** Define el espacio entre los bordes (en pixeles).



Al igual que el cellpadding, la propiedad cellspacing es uniforme en toda la tabla.

Ejemplo: `<table cellspacing="5">`

**height**

Define la altura de la tabla en pixeles o porcentaje.

Hay que tener en cuenta al utilizar este atributo que las tablas siempre priorizarán la estructura de contenido, es decir, si mi tabla tiene una altura de 100 px, pero el contenido supera este tamaño, la tabla se estirará para mostrar el contenido completo.

Ejemplo: `<table height="200px">`

**width**

Define la anchura de la tabla en pixeles o porcentaje.

Ejemplo: `<table width="80%">`

## Tablas anidadas

De la misma forma que podemos anidar listas dentro de otras listas, el lenguaje HTML nos permite trabajar con tablas dentro de otras tablas.

Esto nos permite, por ejemplo, trabajar con diferentes atributos de cellpadding, uno para cada tabla.

A continuación veremos un código de anidación de tablas.

```
<table cellspacing="10" cellpadding="10" border="3">
<tr>
  <td align="center">
    Celda de la tabla principal
  </td>
  <td align="center">
```



```
<table cellspacing="2" cellpadding="2" border="1">
<tr>
  <td>Tabla anidada, celda 1</td>
  <td>Tabla anidada, celda 2</td>
</tr>
<tr>
  <td>Tabla anidada, celda 3</td>
  <td>Tabla anidada, celda 4</td>
</tr>
</table>
</td>
</tr>
</table>
```

Celda de la tabla principal	<table><tr><td>Tabla anidada, celda 1</td><td>Tabla anidada, celda 2</td></tr><tr><td>Tabla anidada, celda 3</td><td>Tabla anidada, celda 4</td></tr></table>	Tabla anidada, celda 1	Tabla anidada, celda 2	Tabla anidada, celda 3	Tabla anidada, celda 4
Tabla anidada, celda 1	Tabla anidada, celda 2				
Tabla anidada, celda 3	Tabla anidada, celda 4				

Esta posibilidad nos permite trabajar atributos de tabla independientes y nos brinda una forma más flexible de organizar el contenido.

OBSERVACIÓN: Si bien vamos a profundizar sobre el tema de color en HTML en las próximas unidades, para que puedan realizar las prácticas con mayor comodidad, pueden obtener los códigos de color en la página oficial de W3C:

[http://www.w3schools.com/Html/html\\_colors.asp](http://www.w3schools.com/Html/html_colors.asp)



## Resumen

### En esta Unidad...

En la presente unidad comenzamos a introducirnos en el lenguaje de creación de sitios web: HTML.

Con las etiquetas propuestas podemos comenzar a plantear la estructura semántica de una página web.

### En la próxima Unidad...

En la próxima unidad vamos a comenzar a trabajar con los nuevos elementos incorporados en la versión HTML5.