

# **Unidad 2.1 Creación de interfaces web utilizando estilos.**

## **Repaso de HTML**

<b>1. LENGUAJE HTML.....</b>	<b>3</b>
<b>2. ESTRUCTURA DE UN DOCUMENTO.....</b>	<b>3</b>
2.1. ELEMENTOS DE LA CABECERA.....	4
2.2 ELEMENTOS DE CUERPO DE DOCUMENTO.....	5
2.2.1 <MAIN>.....	9
2.2.1 <header>.....	9
2.2.2. <nav>.....	10
2.2.3. <section>.....	11
2.2.4. <aside>.....	12
2.2.5. <footer>.....	13
2.3. DENTRO DEL CONTENIDO.....	14
2.3.1. <article>.....	14
2.3.2. <hgroup>.....	17
2.3.3. <figure> y <figcaption>.....	20
<b>3. COLOR.....</b>	<b>21</b>
3.1. CODIFICACIÓN DE COLORES.....	21
<b>4. TEXTO.....</b>	<b>22</b>
4.1 ENCABEZADOS. JERARQUÍA Y ESTRUCTURA DEL CONTENIDO DE UN DOCUMENTO.....	23
4.2. PÁRRAFOS.....	25
4.3. SEPARADORES DE TEXTO.....	26
4.4. ETIQUETAS ESPECÍFICAS PARA EL MARCADO DE TEXTO. ESTILOS LÓGICOS.....	27
4.4.1. <i>Etiquetas de estilos lógicos aplicadas a un conjunto de caracteres</i> .....	30
<b>5. ENLACES DE HIPERTEXTO.....</b>	<b>31</b>
5.1. ESTRUCTURA DE UN ENLACE: LA DIRECCIÓN DE INTERNET O URL.....	33
5.2. DIFERENCIAS ENTRE ENLACES ABSOLUTOS Y RELATIVOS.....	33
5.3. ENLACES INTERNOS.....	39
5.4. ENLACES ESPECIALES: CORREO ELECTRÓNICO. ENLACES DE DESCARGA.....	41
5.4.1. <i>Enlace a un correo electrónico</i> .....	41
5.4.2. <i>Enlaces de descarga</i> .....	41
5.5. ATRIBUTOS ESPECÍFICOS: TÍTULOS, DESTINO, ATAJO DE TECLADO, ETC.....	41
5.5.1. <i>Tipo de contenido (type)</i> .....	41
5.5.2. <i>Tipo de relación (rel y rev)</i> .....	42
<b>6. IMÁGENES.....</b>	<b>43</b>
6.1. FORMATOS DE IMÁGENES.....	44
6.2. CARACTERÍSTICAS DE IMÁGENES: TAMAÑO, TÍTULO, TEXTOS ALTERNATIVOS.....	45
<i>Tamaño</i> .....	45
<i>Título</i> .....	46
<i>Textos alternativos</i> .....	46
6.3. ENLACES EN IMÁGENES.....	46
<b>7. LISTAS.....</b>	<b>47</b>
7.1. ORDENACIÓN DE LISTAS.....	48
7.2. ANIDAMIENTO EN LISTAS.....	48
<b>8. TABLAS.....</b>	<b>49</b>
8.1. ESTRUCTURA BÁSICA.....	49

8.2. FORMATO DE CONTENIDO DE CELDAS.....	50
8.3. TABLAS ANIDADAS.....	52
<b>9. MARCOS (FRAMES).....</b>	<b>54</b>
9.1. CREACIÓN DE MARCOS.....	54
9.1. MARCOS INCRUSTADOS (IFRAMES).....	55
<b>10. FORMULARIOS.....</b>	<b>56</b>
10.1 NUEVOS ATRIBUTOS EN HTML5.....	58
<i>Atributo placeholder HTML5</i> .....	58
<i>Atributo autofocus HTML5</i> .....	59
<i>Atributo autocomplete HTML5</i> .....	59
<i>Atributos heigh y width HTML5</i> .....	59
<i>Atributo list HTML5</i> .....	60
<i>Atributos min y max HTML5</i> .....	60
<i>Atributo multiple HTML5</i> .....	60
<i>Atributo novalidate HTML5</i> .....	60
<i>Atributo pattern HTML5</i> .....	60
<i>Atributo required HTML5</i> .....	61
<i>Atributo step HTML5</i> .....	61
10.2 NUEVOS TIPOS DE INPUT EN HTML5.....	61
<i>Ejemplos de los nuevos atributos</i> .....	62

## 1. LENGUAJE HTML

HTML (HypertextMarkupLanguage) o lenguaje de marcas de hipertexto, es el lenguaje que permite la generación de hipertextos en la World Wide Web. El lenguaje HTML deriva de SGML pero, a su vez, ha tenido unos desarrollos posteriores. Antes de seguir con HTML aclararemos los conceptos de hipertexto y el funcionamiento de la Web.

Podemos definir el hipertexto como una manera de organizar la información de forma no secuencial. Mediante el modelo de hipertexto un sistema puede organizar y presentar los documentos en un medio informático, basado en la vinculación de documentos o fragmentos documentales digitales (textuales o gráficos) a otros fragmentos o documentos, lo que permite acceder a la información no necesariamente de forma secuencial.

Una web donde puedes encontrar todas las etiquetas html5 para consultar es:

<http://tablaelementoshtml.esy.es>

## 2. ESTRUCTURA DE UN DOCUMENTO

Todo documento HTML tiene la siguiente estructura básica:

```
<html>
```

```
<head>
...
</head>
<body>
...
</body>
</html>
```

Las etiquetas `<html>` y `</html>` nos indican que todo el texto contenido entre ambas es código HTML. Por tanto, la primera debe abrir este tipo de documentos y la segunda, cerrarlos. La cabecera del documento, delimitada por `<head>` y `</head>`, proporciona información acerca de él.

El contenido, propiamente dicho, se aloja dentro del cuerpo del documento y viene señalado con las etiquetas `<body>` y `</body>`. Esa estructura debe ser antecedida por una declaración de tipo de documento. Por ejemplo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Esta declaración indica que las definiciones que describen la estructura de un documento XHTML se exponen en un documento DTD en la Red cuyo URL se especifica. De esta forma, los clientes web conocerán sin ambigüedad con qué reglas deben procesar el documento.

O simplemente:

```
<!DOCTYPE html> (Esta es para HTML5)
```

## 2.1. ELEMENTOS DE LA CABECERA

Lo primero que diseñamos en la página web, es la cabecera. En la cabecera de un documento HTML se incluyen las siguientes etiquetas que dan información de la página o soporte para ella:

a) **<base>** Permite indicar la URL por defecto para las URL relativas, así como el lugar donde se abrirán las páginas enlazadas.

b) **<link>** Sirve para relacionar un documento con los recursos externos que utiliza, como por ejemplo la hoja de estilos aplicable al documento. Ejemplo:

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

c) **<title> ... </title>** Como ya hemos visto sirve especificar el texto que aparecerá en la barra de título de la ventana, es importante no confundir con que al hacer una página web la información que presentemos lleve un título, el cual deberemos poner con una etiqueta de encabezado `<h1>` antes de los párrafos de información.

d) **<meta name="..." content="..." />** Permite proporcionar metainformación al navegador sobre la página. Se escriben tantas etiquetas meta, como valores de name queramos utilizar. Algunos de los valores de name son:

- *author*, permite indicar el autor de la página.
- *keywords*, lista de palabras clave separadas por comas.
- *description*, permite incluir una breve descripción de la página.

Por ejemplo:

```
<head>
  <meta name="author" content="Paquito" />
  <meta name="keywords" lang="es" content="instituto, granada, clase, ciclo
  formativo" />
  <meta name="description" lang="es" content="Página de ejemplo para la
  explicación" />
</head>
```

**Cosas a tener en cuenta:**

- El atributo de escheme no es compatible con HTML5.
- HTML5 tiene un nuevo atributo, **charset**, que hace más fácil definir el conjunto de caracteres:
  - **HTML 4.01:** `<meta http-equiv = "Content-Type" content = "text / html; charset = UTF-8">`
  - **HTML5:** `<charset meta = "UTF-8">`

Valores más comunes:

- UTF-8 – Conjunto de caracteres Unicode.
- ISO-8859-1 – Conjunto de caracteres para el alfabeto latino.

e) **<script>**, con esta etiqueta podemos incluir código en algún lenguaje cliente, por ejemplo, JavaScript. Esta etiqueta también puede aparecer una o más veces en la sección body.

Por ejemplo:

```
<script type="text/javascript">
alert("hola")
</script>
```

f) **<style type="text/css">** Al igual que con `<link>` podemos especificar los estilos para etiquetas dentro del mismo documento, por ejemplo:

```
<style type="text/css">
body {background: black; color:white;}
div {background: red; width:300px;}
</style>
```

## 2.2 ELEMENTOS DE CUERPO DE DOCUMENTO

La estructura del cuerpo (el código entre las etiquetas <body>) generará la parte visible del documento. Este es el código que producirá nuestra página web.

HTML ofrece diferentes formas de construir y organizar la información dentro del cuerpo de un documento. Uno de los primeros elementos provistos para este propósito fue <table>. Las **tablas** permitían a los diseñadores acomodar datos, texto, imágenes y herramientas dentro de filas y columnas de celdas, incluso sin que hayan sido concebidas para este propósito.

En los primeros días de la web, las tablas fueron una revolución, un gran paso hacia adelante con respecto a la visualización de los documentos y la experiencia ofrecida a los usuarios. Más adelante, gradualmente, otros elementos reemplazaron su función, permitiendo lograr lo mismo con menos código, facilitando de este modo la creación, permitiendo portabilidad y ayudando al mantenimiento de los sitios web.

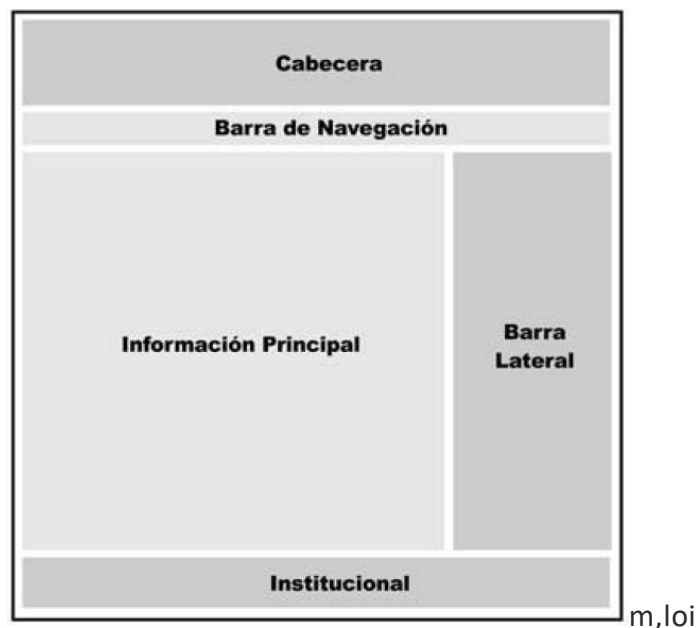
El elemento <div> comenzó a dominar la escena. Con el surgimiento de webs más interactivas y la integración de HTML, CSS y Javascript, el uso de <div> se volvió una práctica común. Pero este elemento, así como <table>, no provee demasiada información acerca de las partes del cuerpo que está representando. Desde imágenes a menús, textos, enlaces, códigos, formularios, cualquier cosa puede ir entre las etiquetas de apertura y cierre de un elemento <div>. En otras palabras, la palabra clave div solo especifica una división en el cuerpo, como la celda de una tabla, pero no ofrece indicio alguno sobre qué clase de división es, cuál es su propósito o qué contiene.

Para los usuarios estas claves o indicios no son importantes, pero para los navegadores la correcta interpretación de qué hay dentro del documento que se está procesando puede ser crucial en muchos casos. Luego de la revolución de los dispositivos móviles y el surgimiento de diferentes formas en que la gente accede a la web, la identificación de cada parte del documento es una tarea que se ha vuelto más relevante que nunca.

Considerando todo lo expuesto, **HTML5 incorpora nuevos elementos que ayudan a identificar cada sección del documento** y organizar el cuerpo del mismo. En HTML5 las secciones más importantes son diferenciadas y la estructura principal ya no depende más de los elementos <div> o <table>.

Cómo usamos estos nuevos elementos depende de nosotros, pero las palabras clave otorgadas a cada uno de ellos nos dan ayuda a entender sus funciones. Normalmente una página o aplicación web está dividida entre varias áreas visuales para mejorar la experiencia del usuario y facilitar la interactividad. Las palabras claves que representan cada nuevo elemento de HTML5 están íntimamente relacionadas con estas áreas, como veremos pronto.

La siguiente imagen representa un diseño común encontrado en la mayoría de los sitios webs estos días. A pesar del hecho de que cada diseñador crea sus propios diseños, en general podremos identificar las siguientes secciones en cada sitio web estudiado:



Representación visual de un clásico diseño web.

En la parte superior, descrito como **Cabecera**, se encuentra el espacio donde usualmente se ubica el logo, título, subtítulos y una corta descripción del sitio web o la página.

Inmediatamente debajo, podemos ver la **Barra de Navegación** en la cual casi todos los desarrolladores ofrecen un menú o lista de enlaces con el propósito de facilitar la navegación a través del sitio. Los usuarios son guiados desde esta barra hacia las diferentes páginas o documentos, normalmente pertenecientes al mismo sitio web.

El **contenido** más relevante de una página web se encuentra, en casi todo diseño, ubicado en su centro. Esta sección presenta información y enlaces valiosos. La mayoría de las veces es dividida en varias filas y columnas. En el ejemplo de la figura se utilizaron solo dos columnas: **Información Principal y Barra Lateral**, pero esta sección es extremadamente flexible y normalmente diseñadores la adaptan acorde a sus necesidades insertando más columnas, dividiendo cada columna entre bloques más pequeños o generando diferentes distribuciones y combinaciones. El contenido presentado en esta parte del diseño es usualmente de alta prioridad. En el diseño de ejemplo, Información Principal podría contener una lista de artículos, descripción de productos, entradas de un blog o cualquier otra información importante, y la Barra Lateral podría mostrar una lista de enlaces apuntando hacia cada uno de esos ítems. En un blog, por ejemplo, esta última columna ofrecerá una lista de enlaces apuntando a cada entrada del blog, información acerca del autor, etc...

En la base de un diseño web clásico siempre nos encontramos con una barra más que aquí llamamos **Institucional**. La nombramos de esta manera porque esta es el área en donde normalmente se muestra información acerca del sitio web, el autor o la empresa, además de algunos enlaces con respecto a reglas, términos y condiciones y toda información adicional que el desarrollador considere importante compartir. La barra Institucional es un complemento de la

Cabecera y es parte de lo que se considera estos días la estructura esencial de una página web, como podemos apreciar en el siguiente ejemplo:



Representación visual de un clásico diseño para blogs.

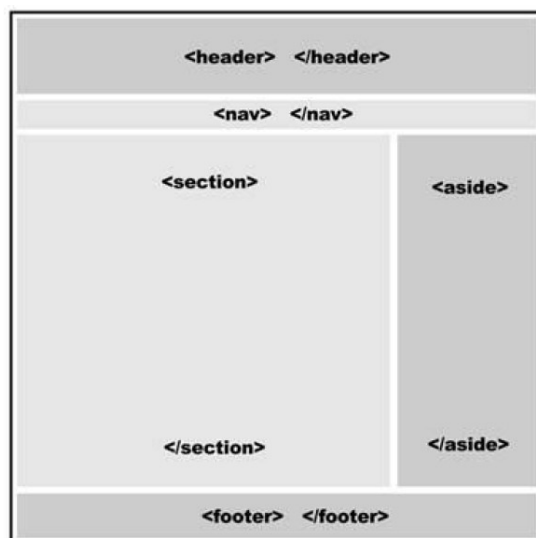
La figura es una representación de un blog normal. En este ejemplo se puede claramente identificar cada parte del diseño considerado anteriormente.

- Cabecera
- Barra de Navegación
- Sección de Información Principal
- Barra Lateral
- El pie o la barra Institucional

Esta simple representación de un blog nos puede ayudar a entender que cada sección definida en un sitio web tiene un propósito. A veces este propósito no es claro pero en esencia se encuentra siempre allí, ayudándonos a reconocer cualquiera de las secciones descritas anteriormente en todo diseño.

HTML5 considera esta estructura básica y provee nuevos elementos para diferenciar y declarar cada una de sus partes. A partir de ahora podemos decir al navegador para qué es cada sección:





Representación visual de un diseño utilizando elementos HTML5.

La figura muestra el típico diseño presentado anteriormente, pero esta vez con los correspondientes elementos HTML5 para cada sección (incluyendo etiquetas de apertura y cierre).

### 2.2.1 <MAIN>

El propósito principal de esta etiqueta es desde un punto de vista de accesibilidad, ya que ayuda a que los *screen readers* (lectores de pantalla) y otras tecnologías asistenciales puedan identificar donde comienza el contenido principal de la página y donde termina.

Según la especificación: el elemento *main* representa el contenido principal del cuerpo (*body*) de un documento.

Una característica importante a tener en cuenta sobre este elemento, es que **puede ser usado una sola vez por página**, algo que parece bastante obvio si pensamos su propósito. Usar más de un elemento *main*, hará que nuestro HTML sea inválido para la W3C.

Este elemento **no puede ser hijo de ninguno de los siguientes**: *header*, *nav*, *article*, *aside* y *footer*. Puede incluirla sección de cabecera (*header*) y pie (*footer*) principal o no.

### 2.2.1 <HEADER>

Uno de los nuevos elementos incorporados en HTML5 es <header>. El elemento <header> no debe ser confundido con <head> usado antes para construir la cabecera del documento. Del mismo modo que <head>, la intención de <header> es proveer información introductoria (títulos, subtítulos, logos), pero difiere con respecto a <head> en su alcance. Mientras que el elemento <head> tiene el propósito de proveer información acerca de todo el documento, <header> es usado solo para el cuerpo o secciones específicas dentro del cuerpo:

```
<!DOCTYPE html>
<html lang="es">
```

```

<head>
  <meta charset="iso-8859-1">
  <meta name="description" content="Ejemplo de HTML5">
  <meta name="keywords" content="HTML5, CSS3, JavaScript">
  <title>Este texto es el título del documento</title>
  <link rel="stylesheet" href="misestilos.css">
</head>
<body>
  <header>
    <h1>Este es el título principal del sitio web</h1>
  </header>
</body>
</html>

```

En el código definimos el título de la página web utilizando el elemento `<header>`. Recuerde que esta cabecera no es la misma que la utilizada previamente para definir el título del documento. La inserción del elemento `<header>` representa el comienzo del cuerpo y por lo tanto de la parte visible del documento. De ahora en más será posible ver los resultados de nuestro código en la ventana del navegador.

*Conceptos básicos: Entre las etiquetas `<header>` en el código hay un elemento que probablemente no conoce. El elemento `<h1>` es un viejo elemento HTML usado para definir títulos. El número indica la importancia del título. El elemento `<h1>` es el más importante y `<h6>` el de menor importancia, por lo tanto `<h1>` será utilizado para mostrar el título principal y los demás para subtítulos o subtítulos internos. Más adelante veremos cómo estos elementos trabajan en HTML5.*

### 2.2.2. `<NAV>`

Siguiendo con nuestro ejemplo, la siguiente sección es la Barra de Navegación. Esta barra es generada en HTML5 con el elemento `<nav>`:

```

<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="iso-8859-1">
    <meta name="description" content="Ejemplo de HTML5">
    <meta name="keywords" content="HTML5, CSS3, JavaScript">
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    <header>
      <h1>Este es el título principal del sitio web</h1>
    </header>
    <nav>
      <ul>
        <li>principal</li>
        <li>fotos</li>
        <li>videos</li>
        <li>contacto</li>
      </ul>
    </nav>
  </body>
</html>

```

Como se puede apreciar en este código, el elemento `<nav>` se encuentra dentro de las etiquetas `<body>` pero es ubicado después de la etiqueta de cierre de la cabecera (`</header>`), no dentro de las etiquetas `<header>`. Esto es porque `<nav>` no es parte de la cabecera sino una nueva sección.

Anteriormente dijimos que la estructura y el orden que elegimos para colocar los elementos HTML5 dependen de nosotros. Esto significa que HTML5 es versátil y solo nos otorga los parámetros y elementos básicos con los que trabajar, pero cómo usarlos será exclusivamente decisión nuestra. Un ejemplo de esta versatilidad es que el elemento `<nav>` podría ser insertado dentro del elemento `<header>` o en cualquier otra parte del cuerpo. Sin embargo, siempre se debe considerar que estas etiquetas fueron creadas para brindar información a los navegadores y ayudar a cada nuevo programa y dispositivo en el mercado a identificar las partes más relevantes del documento. Para conservar nuestro código portable y comprensible, recomendamos como buena práctica seguir lo que marcan los estándares y mantener todo tan claro como sea posible. El elemento `<nav>` fue creado para ofrecer ayuda para la navegación, como en menús principales o grandes bloques de enlaces, y debería ser utilizado de esa manera.

*Conceptos básicos: En el ejemplo generamos las opciones del menú para nuestra página web. Entre las etiquetas `<nav>` hay dos elementos que son utilizados para crear una lista. El propósito del elemento `<ul>` es definir la lista. Anidado entre las etiquetas `<ul>` encontramos varias etiquetas `<li>` con diferentes textos representando las opciones del menú. Las etiquetas `<li>`, como probablemente ya se ha dado cuenta, son usadas para definir cada ítem de la lista.*

### 2.2.3. `<SECTION>`

Siguiendo nuestro diseño estándar nos encontramos con las columnas que llamamos Información Principal y Barra Lateral. Como explicamos anteriormente, la columna Información Principal contiene la información más relevante del documento y puede ser encontrada en diferentes formas (por ejemplo, dividida en varios bloques o columnas). Debido a que el propósito de estas columnas es más general, el elemento en HTML5 que especifica estas secciones se llama simplemente `<section>`:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="iso-8859-1">
    <meta name="description" content="Ejemplo de HTML5">
    <meta name="keywords" content="HTML5, CSS3, JavaScript">
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    <header>
      <h1>Este es el título principal del sitio web</h1>
    </header>

    <nav>
      <ul>
        <li>principal</li>
```

```

        <li>fotos</li>
        <li>videos</li>
        <li>contacto</li>
    </ul>
</nav>

    <section>
</section>

</body>
</html>

```

Al igual que la Barra de Navegación, la columna Información Principal es una sección aparte. Por este motivo, la sección para Información Principal va debajo de la etiqueta de cierre </nav>.

---

#### 2.2.4. <ASIDE>

En un típico diseño web la columna llamada Barra Lateral se ubica al lado de la columna Información Principal. Esta es una columna o sección que normalmente contiene datos relacionados con la información principal pero que no son relevantes o igual de importantes.

En el diseño de un blog, por ejemplo, la Barra Lateral contendrá una lista de enlaces. En el ejemplo, los enlaces apuntan a cada una de las entradas del blog y ofrecen información adicional sobre el autor (número 4). La información dentro de esta barra está relacionada con la información principal pero no es relevante por sí misma. Siguiendo el mismo ejemplo podemos decir que las entradas del blog son relevantes pero los enlaces y las pequeñas reseñas sobre esas entradas son solo una ayuda para la navegación, pero no lo que al lector realmente le interesa.

En HTML5 podemos diferenciar esta clase secundaria de información utilizando el elemento <aside>:

```

<!DOCTYPE html>
<html lang="es">
    <head>
        <meta charset="iso-8859-1">
        <meta name="description" content="Ejemplo de HTML5">
        <meta name="keywords" content="HTML5, CSS3, JavaScript">
        <title>Este texto es el título del documento</title>
        <link rel="stylesheet" href="misestilos.css">
    </head>
    <body>
        <header>
            <h1>Este es el título principal del sitio web</h1>
        </header>

        <nav>
            <ul>
                <li>principal</li>
                <li>fotos</li>
                <li>videos</li>
                <li>contacto</li>
            </ul>
        </nav>

        <section>
</section>

```

```

        <aside>
        <blockquote>Mensaje número uno</blockquote>
        <blockquote>Mensaje número dos</blockquote>
        </aside>

    </body>
</html>

```

El elemento `<aside>` podría estar ubicado del lado derecho o izquierdo de nuestra página de ejemplo, la etiqueta no tiene una posición predefinida. El elemento `<aside>` solo describe la información que contiene, no el lugar dentro de la estructura. Este elemento puede estar ubicado en cualquier parte del diseño y ser usado siempre y cuando su contenido no sea considerado como el contenido principal del documento. Por ejemplo, podemos usar `<aside>` dentro del elemento `<section>` o incluso insertado entre la información relevante, como en el caso de una cita.

---

### 2.2.5. <FOOTER>

Para finalizar la construcción de la plantilla o estructura elemental de nuestro documento HTML5, solo necesitamos un elemento más. Ya contamos con la cabecera del cuerpo, secciones con ayuda para la navegación, información importante y hasta una barra lateral con datos adicionales, por lo tanto, lo único que nos queda por hacer es cerrar nuestro diseño para otorgarle un final al cuerpo del documento. HTML5 provee un elemento específico para este propósito llamado `<footer>`:

```

<!DOCTYPE html>
<html lang="es">
    <head>
        <meta charset="iso-8859-1">
        <meta name="description" content="Ejemplo de HTML5">
        <meta name="keywords" content="HTML5, CSS3, JavaScript">
        <title>Este texto es el título del documento</title>
        <link rel="stylesheet" href="misestilos.css">
    </head>
    <body>
        <header>
            <h1>Este es el título principal del sitio web</h1>
        </header>

        <nav>
            <ul>
                <li>principal</li>
                <li>fotos</li>
                <li>videos</li>
                <li>contacto</li>
            </ul>
        </nav>

        <section>
        </section>

        <aside>
        <blockquote>Mensaje número uno</blockquote>
        <blockquote>Mensaje número dos</blockquote>
        </aside>
    </body>
</html>

```

```
<footer>
Derechos Reservados &copy; 2010-2011
</footer>

</body>
</html>
```

En el típico diseño de una página web la sección llamada Institucional será definida por etiquetas `<footer>`.

Esto es debido a que la barra representa el final (o pie) del documento y esta parte de la página web es normalmente usada para compartir información general sobre el autor o la organización detrás del proyecto.

Generalmente, el elemento `<footer>` representará el final del cuerpo de nuestro documento y tendrá el propósito descrito anteriormente. Sin embargo, `<footer>` puede ser usado múltiples veces dentro del cuerpo para representar también el final de diferentes secciones (del mismo modo que la etiqueta `<header>`).

### 2.3. DENTRO DEL CONTENIDO

El cuerpo de nuestro documento está listo. La estructura básica de nuestro sitio web fue finalizada, pero aún tenemos que trabajar en el contenido. Los elementos HTML5 estudiados hasta el momento nos ayudan a identificar cada sección del diseño y asignar un propósito intrínseco a cada una de ellas, pero lo que es realmente importante para nuestro sitio web se encuentra en el interior de estas secciones.

La mayoría de los elementos ya estudiados fueron creados para construir una estructura para el documento HTML que pueda ser identificada y reconocida por los navegadores y nuevos dispositivos. Aprendimos acerca de la etiqueta `<body>` usada para declarar el cuerpo o parte visible del documento, la etiqueta `<header>` con la que agrupamos información importante para el cuerpo, la etiqueta `<nav>` que provee ayuda para la navegación del sitio web, la etiqueta `<section>` necesaria para contener la información más relevante, y también `<aside>` y `<footer>` para ofrecer información adicional de cada sección y del documento mismo. Pero ninguno de estos elementos declara algo acerca del contenido. Todos tienen un específico propósito estructural.

Cuanto más nos introducimos dentro del documento más cerca nos encontramos de la definición del contenido. Esta información estará compuesta por diferentes elementos visuales como títulos, textos, imágenes, videos y aplicaciones interactivas, entre otros. Necesitamos poder diferenciar estos elementos y establecer una relación entre ellos dentro de la estructura.

---

#### 2.3.1. `<ARTICLE>`

El diseño considerado anteriormente es el más común y representa una estructura esencial para los sitios web estos días, pero es además ejemplo de cómo el contenido clave es mostrado en pantalla. Del mismo modo que los blogs están divididos en entradas, los sitios web normalmente

presentan información relevante dividida en partes que comparten similares características. El elemento `<article>` nos permite identificar cada una de estas partes:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="iso-8859-1">
    <meta name="description" content="Ejemplo de HTML5">
    <meta name="keywords" content="HTML5, CSS3, JavaScript">
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    <header>
      <h1>Este es el título principal del sitio web</h1>
    </header>

    <nav>
      <ul>
        <li>principal</li>
        <li>fotos</li>
        <li>videos</li>
        <li>contacto</li>
      </ul>
    </nav>

    <section>
      <article>
        Este es el texto de mi primer mensaje
      </article>
      <article>
        Este es el texto de mi segundo mensaje
      </article>
    </section>

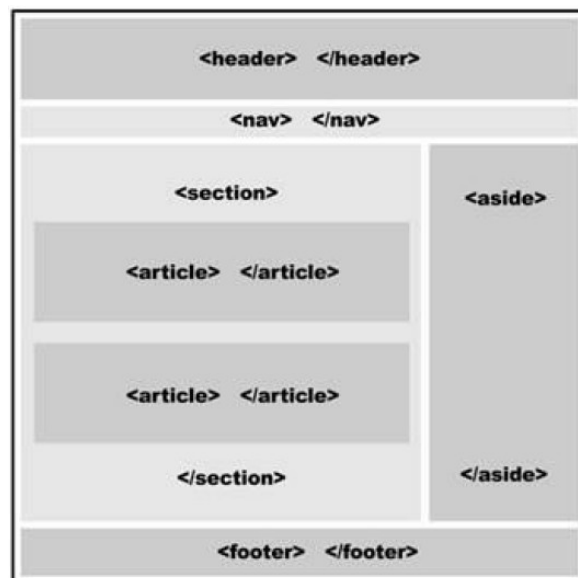
    <aside>
      <blockquote>Mensaje número uno</blockquote>
      <blockquote>Mensaje número dos</blockquote>
    </aside>

    <footer>
      Derechos Reservados &copy; 2010-2011
    </footer>
  </body>
</html>
```

Como puede observarse en el código las etiquetas `<article>` se encuentran ubicadas dentro del elemento `<section>`. Las etiquetas `<article>` en nuestro ejemplo pertenecen a esta sección, son sus hijos, del mismo modo que cada elemento dentro de las etiquetas `<body>` es hijo del cuerpo. Y al igual que cada elemento hijo del cuerpo, las etiquetas `<article>` son ubicadas una sobre otra, como es mostrado en la siguiente figura.

*Conceptos básicos: La estructura de un documento HTML puede ser descripta como un árbol, con el elemento `<html>` como su raíz. Otra forma de describir la relación entre elementos es nombrarlos como padres, hijos y hermanos, de acuerdo a la posición que*

ocupan dentro de esa misma estructura. Por ejemplo, en un típico documento HTML el elemento `<body>` es hijo del elemento `<html>` y hermano del elemento `<head>`. Ambos, `<body>` y `<head>`, tienen al elemento `<html>` como su padre.



En la imagen vemos la representación visual de las etiquetas `<article>` que fueron incluidas para contener información relevante de la página web.

El elemento `<article>` no está limitado por su nombre (no se limita, por ejemplo, a artículos de noticias). Este elemento fue creado con la intención de contener unidades independientes de contenido, por lo que puede incluir mensajes de foros, artículos de una revista digital, entradas de blog, comentarios de usuarios, etc... Lo que hace es agrupar porciones de información que están relacionadas entre sí independientemente de su naturaleza.

Como una parte independiente del documento, el contenido de cada elemento `<article>` tendrá su propia estructura.

Para definir esta estructura, podemos aprovechar la versatilidad de los elementos `<header>` y `<footer>` estudiados anteriormente. Estos elementos son portables y pueden ser usados no solo para definir los límites del cuerpo sino también en cualquier sección de nuestro documento:

```

<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="iso-8859-1">
    <meta name="description" content="Ejemplo de HTML5">
    <meta name="keywords" content="HTML5, CSS3, JavaScript">
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    <header>
      <h1>Este es el título principal del sitio web</h1>
    </header>
  
```



```

    <nav>
    <ul>
    <li>principal</li>
    <li>fotos</li>
    <li>videos</li>
    <li>contacto</li>
    </ul>
    </nav>

    <section>
        <article>
            <header>
            <h1>Titulo del mensaje uno</h1>
            </header>
            Este es el texto del primer mensaje
            <footer>
            <p>comentarios (0)</p>
            </footer>
        </article>

        <article>
            <header>
            <h1>Titulo del mensaje dos</h1>
            </header>
            Este es el texto de mi segundo mensaje
            <footer>
            <p>comentarios (0)</p>
            </footer>
        </article>
    </section>

    <aside>
    <blockquote>Mensaje número uno</blockquote>
    <blockquote>Mensaje número dos</blockquote>
    </aside>

    <footer>
    Derechos Reservados &copy; 2010-2011
    </footer>
</body>
</html>

```

Los dos mensajes insertados en el código han sido contruidos con el elemento `<article>` y tienen una estructura específica. En la parte superior de esta estructura incluimos las etiquetas `<header>` conteniendo el título definido con el elemento `<h1>`, debajo se encuentra el contenido mismo del mensaje y sobre el final, luego del texto, vienen las etiquetas `<footer>` especificando la cantidad de comentarios recibidos.

---

### 2.3.2. `<HGROUP>`

Dentro de cada elemento `<header>`, en la parte superior del cuerpo o al comienzo de cada `<article>`, incorporamos elementos `<h1>` para declarar un título. Básicamente, las etiquetas `<h1>` son todo lo que necesitamos para crear una línea de cabecera para cada parte del documento, pero es normal que necesitemos también agregar subtítulos o más información que especifique de qué se trata la página web o una sección en particular. De hecho, el elemento `<header>` fue

creado para contener también otros elementos como tablas de contenido, formularios de búsqueda o textos cortos y logos.

Para construir este tipo de cabeceras, podemos aprovechar el resto de las etiquetas H, como <h1>, <h2>, <h3>, <h4>, <h5> y <h6>, pero siempre considerando que, por propósitos de procesamiento interno, y para evitar generar múltiples secciones durante la interpretación del documento por parte del navegador, estas etiquetas deben ser agrupadas juntas. Por esta razón, HTML5 provee el elemento **<hgroup>**:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="iso-8859-1">
    <meta name="description" content="Ejemplo de HTML5">
    <meta name="keywords" content="HTML5, CSS3, JavaScript">
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    <header>
      <h1>Este es el título principal del sitio web</h1>
    </header>
    <nav>
      <ul>
        <li>principal</li>
        <li>fotos</li>
        <li>videos</li>
        <li>contacto</li>
      </ul>
    </nav>
    <section>
      <article>
        <header>
          <hgroup>
            <h1>Título del mensaje uno</h1>
            <h2>Subtítulo del mensaje uno</h2>
          </hgroup>
          <p>publicado 10-12-2011</p>
        </header>
        Este es el texto de mi primer mensaje
        <footer>
          <p>comentarios (0)</p>
        </footer>
      </article>

      <article>
        <header>
          <hgroup>
            <h1>Título del mensaje dos</h1>
            <h2>Subtítulo del mensaje dos</h2>
          </hgroup>
          <p>publicado 15-12-2011</p>
        </header>
        Este es el texto de mi segundo mensaje
        <footer>
          <p>comentarios (0)</p>
        </footer>
      </article>
    </section>
```

```
<aside>
  <blockquote>Mensaje número uno</blockquote>
  <blockquote>Mensaje número dos</blockquote>
</aside>

<footer>
  Derechos Reservados &copy; 2010-2011
</footer>
</body>
</html>
```

Las etiquetas H deben conservar su jerarquía, lo que significa que debemos primero declarar la etiqueta <h1>, luego usar <h2> para subtítulos y así sucesivamente. Sin embargo, a diferencia de anteriores versiones de HTML, HTML5 nos deja reusar las etiquetas H y construir esta jerarquía una y otra vez en cada sección del documento. En el ejemplo agregamos un subtítulo y datos adicionales a cada mensaje. Los títulos y subtítulos fueron agrupados juntos utilizando <hgroup>, recreando de este modo la jerarquía <h1> y <h2> en cada elemento <article>.

*IMPORTANTE: El elemento <hgroup> es necesario cuando tenemos un título y subtítulo o más etiquetas H juntas en la misma cabecera. Este elemento puede contener solo etiquetas H y esta fue la razón por la que en nuestro ejemplo dejamos los datos adicionales afuera. Si solo dispone de una etiqueta <h1> o la etiqueta <h1> junto con datos adicionales, no tiene que agrupar estos elementos juntos. Por ejemplo, en la cabecera del cuerpo (<header>) no usamos este elemento porque solo tenemos una etiqueta H en su interior. Siempre recuerde que <hgroup> fue creado solo con la intención de agrupar etiquetas H, exactamente como su nombre lo indica.*

Navegadores y programas que ejecutan y presentan en la pantalla sitios webs leen el código HTML y crean su propia estructura interna para interpretar y procesar cada elemento. Esta estructura interna está dividida en secciones que no tienen nada que ver con las divisiones en el diseño o el elemento <section>. Estas son secciones conceptuales generadas durante la interpretación del código. El elemento <header> no crea una de estas secciones por sí mismo, lo que significa que los elementos dentro de <header> representarán diferentes niveles e internamente pueden generar diferentes secciones. El elemento <hgroup> fue creado con el propósito de agrupar las etiquetas H y evitar interpretaciones incorrectas por parte de los navegadores.

*Conceptos básicos: lo que llamamos “información adicional” dentro de la cabecera en nuestra descripción previa es conocido como Metadata. Metadata es un conjunto de datos que describen y proveen información acerca de otro grupo de datos. En nuestro ejemplo, Metadata es la fecha en la cual cada mensaje fue publicado.*

### 2.3.3. <FIGURE> Y <FIGCAPTION>

La etiqueta <figure> fue creada para ayudarnos a ser aún más específicos a la hora de declarar el contenido del documento. Antes de que este elemento fuera introducido, no podíamos identificar el contenido que era parte de la información, pero a la vez independiente, como ilustraciones, fotos, videos, etc... Normalmente estos elementos son parte del contenido relevante, pero pueden ser extraídos o movidos a otra parte sin afectar o interrumpir el flujo del documento. Cuando nos encontramos con esta clase de información, las etiquetas <figure> pueden ser usadas para identificarla:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="iso-8859-1">
    <meta name="description" content="Ejemplo de HTML5">
    <meta name="keywords" content="HTML5, CSS3, JavaScript">
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    <header>
      <h1>Este es el título principal del sitio web</h1>
    </header>

    <nav>
      <ul>
        <li>principal</li>
        <li>fotos</li>
        <li>videos</li>
        <li>contacto</li>
      </ul>
    </nav>

    <section>
      <article>
        <header>
          <hgroup>
            <h1>Título del mensaje uno</h1>
            <h2>Subtítulo del mensaje uno</h2>
          </hgroup>
          <p>publicado 10-12-2011</p>
        </header>
        Este es el texto de mi primer mensaje
        <figure>
          
          <figcaption>
            Esta es la imagen del primer mensaje
          </figcaption>
        </figure>
        <footer>
          <p>comentarios (0)</p>
        </footer>
      </article>
      <article>
        <header>
          <hgroup>
            <h1>Título del mensaje dos</h1>
            <h2>Subtítulo del mensaje dos</h2>
          </hgroup>
          <p>publicado 15-12-2011</p>
```

```
        </header>
        Este es el texto de mi segundo mensaje
        <footer>
        <p>comentarios (0)</p>
        </footer>
        </article>
    </section>

    <aside>
    <blockquote>Mensaje número uno</blockquote>
    <blockquote>Mensaje número dos</blockquote>
    </aside>

    <footer>
    Derechos Reservados &copy; 2010-2011
    </footer>
</body>
</html>
```

En el código, en el primer mensaje, luego del texto insertamos una imagen (). Esta es una práctica común, a menudo el texto es enriquecido con imágenes o videos. Las etiquetas <figure> nos permiten envolver estos complementos visuales y diferenciarlos así de la información más relevante. También en el código se puede observar un elemento extra dentro de <figure>. Normalmente, unidades de información como imágenes o videos son descriptas con un corto texto debajo. HTML5 provee un elemento para ubicar e identificar estos títulos descriptivos. Las etiquetas <figcaption> encierran el texto relacionado con <figure> y establecen una relación entre ambos elementos y su contenido.

## 3. COLOR

### 3.1. CODIFICACIÓN DE COLORES

Los colores en HTML son construidos utilizando una combinación de tres colores básicos: rojo, verde y azul. Cada uno de estos colores puede tomar un valor desde 0 a 255 y son representados en notación hexadecimal (00 a FF). Dicho esto, un color está compuesto por seis caracteres donde los dos primeros representan al color rojo, los dos segundos al verde y los dos terceros al azul. Por ejemplo, el color rojo está formado en HTML con "FF" para el rojo (los dos primeros caracteres) y "00" para el resto: "FF0000".

Al mezclar estos valores se mezclarán los colores resultantes, de modo que puedes crear un amarillo al mezclar el rojo y el verde (por ejemplo, "FFFF00"), violeta al mezclar rojo y azul (por ejemplo, "FF00FF") ó cyan al mezclar verde y azul (por ejemplo, "00FFFF"). Esto funciona exactamente como la paleta de un pintor.

Cuanto mayor sea el valor del par, tanto mayor será la intensidad de color de ese par.

La escala cromática de intensidad de color es:

mínima (nulo = 00)

media (mediano = 80)

máxima (saturado = FF)

El tono del color puede también ser cambiado incrementando (por ejemplo, violeta claro "FF66FF") ó disminuyendo (por ejemplo, violeta oscuro "AA00AA") los tres valores proporcionalmente.

Un valor de color puede ser o bien un número hexadecimal (anteponiendo un signo "#") o uno de los siguientes dieciséis nombres de colores. En los nombres de colores no se distingue entre mayúsculas y minúsculas. A continuación, se listan los colores básicos en HTML:

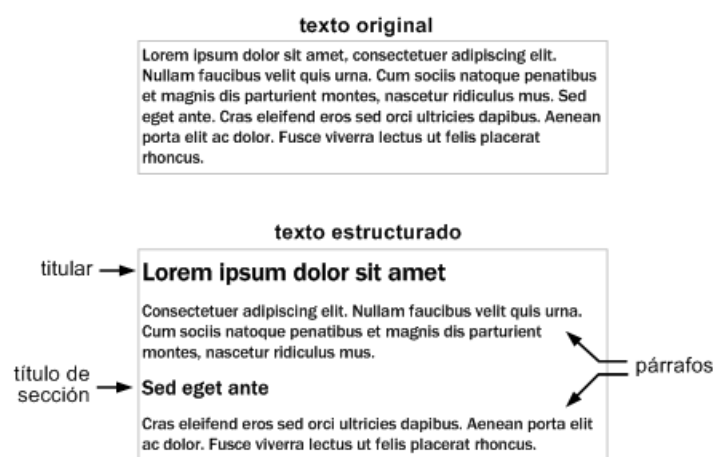
Nombres de colores y valores sRGB	
 Black = "#000000" (Negro)	 Green = "#008000" (Verde)
 Silver = "#C0C0C0" (Plateado)	 Lime = "#00FF00" (Verde lima)
 Gray = "#808080" (Gris)	 Olive = "#808000" (Verde oliva)
 White = "#FFFFFF" (Blanco)	 Yellow = "#FFFF00" (Amarillo)
 Maroon = "#800000" (Marrón)	 Navy = "#000080" (Azul marino)
 Red = "#FF0000" (Rojo)	 Blue = "#0000FF" (Azul)
 Purple = "#800080" (Púrpura)	 Teal = "#008080" (Azul verdoso)
 Fuchsia = "#FF00FF" (Fucsia)	 Aqua = "#00FFFF" (Celeste)

#### 4. TEXTO

La mayor parte del contenido de las páginas HTML habituales está formado por texto, llegando a ser más del 90% del código de la página.

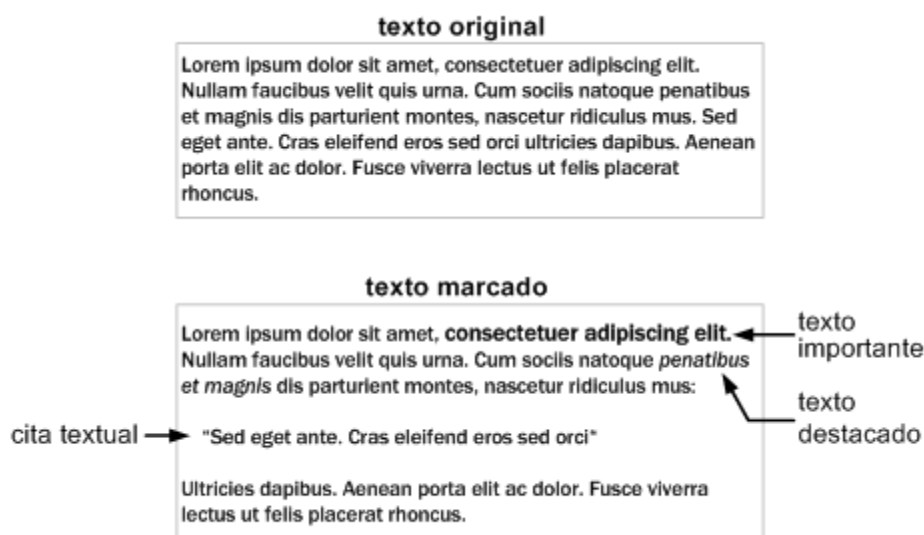
El lenguaje HTML incorpora al tratamiento del texto muchas de las ideas y normas establecidas en otros entornos de publicación de contenidos. De esta forma, HTML define etiquetas para estructurar el contenido en secciones y párrafos y define otras etiquetas para marcar elementos importantes dentro del 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 proceso de estructurar un texto simple consiste en indicar las diferentes zonas o secciones que componen el texto. De esta forma, los textos estructurados utilizan etiquetas para delimitar cada párrafo y títulos de sección para delimitar cada una de las secciones que forman el texto.

Una vez definida la estructura básica de los contenidos de la página, el siguiente paso consiste en marcar los diferentes elementos dentro del propio texto: definiciones, abreviaturas, textos importantes, textos modificados, citas a otras referencias, etc.



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.

#### 4.1 ENCABEZADOS. JERARQUÍA Y ESTRUCTURA DEL CONTENIDO DE UN DOCUMENTO

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.

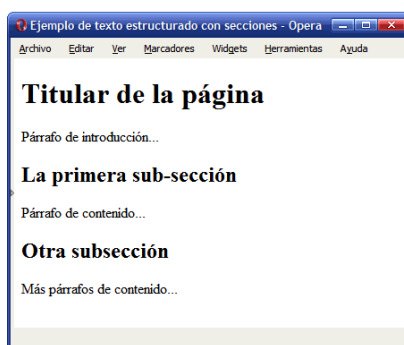
Las etiquetas que definen los títulos de sección son <h1>, <h2>, <h3>, <h4>, <h5> y <h6>. La etiqueta <h1> es la de mayor importancia y por tanto se utiliza para definir los titulares de la página. La importancia del resto de etiquetas es descendiente, de forma que la etiqueta <h6> es la que se utiliza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

Al igual que la etiqueta <p>, las etiquetas de título de sección son elementos de bloque y no tienen atributos específicos.

El siguiente ejemplo muestra el uso de las etiquetas de título de sección:

```
<html>
  <head>
    <title>Ejemplo de texto estructurado con secciones</title>
  </head>
  <body>
    <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>
  </body>
</html>
```

Los navegadores muestran el ejemplo anterior de la siguiente manera:

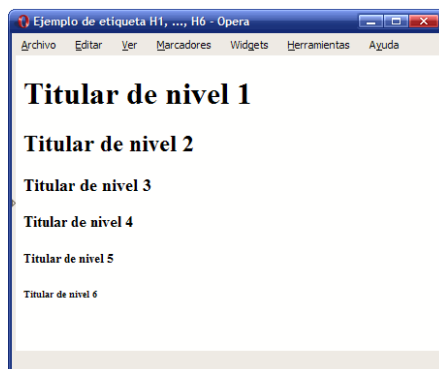


### Ejemplo de texto HTML estructurado con párrafos y secciones

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:





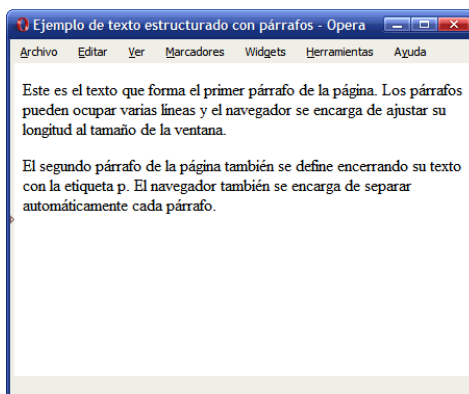
Ejemplo de uso de las etiquetas h1, h2, h3, h4, h5 y h6

## 4.2. 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:

```
<html>
  <head>
    <title>Ejemplo de texto estructurado con párrafos</title>
  </head>
  <body>
    <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>
  </body>
</html>
```

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



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.

### 4.3. SEPARADORES DE TEXTO

Para separar un texto de otro o un párrafo de otro podemos utilizar una línea horizontal de un tamaño o un grosor determinado por nosotros. Esa franja la escribimos con la etiqueta `<hr>`. Lo contrario que muchas etiquetas html, ésta no necesita ser cerrada. Sólo con escribir la etiqueta anterior ya la escribimos.

La etiqueta `<hr>`, como muchas otras etiquetas, puede variar de aspecto dependiendo de una serie de caracteres o parámetros que podemos predefinir. Por ejemplo, podemos definir su grosor mediante el parámetro `size`.

Para escribir este parámetro en la etiqueta escribiremos `"size=x"`, siendo `"x"` el número del grosor de la franja. A continuación te vamos a ofrecer dos ejemplos de franjas con diferentes grosores, siendo la primera 10 y la segunda dos. La diferencia es abismal

```
<hr size="20" />  
<hr size="2" />
```

Y el resultado sería el siguiente:



Otro parámetro que podemos definir es la anchura de la franja mediante el parámetro `width`. El parámetro será `"width=x %"`, siendo `"x"` el tanto por cien que queremos que ocupe nuestra franja.

En el caso de no escribir nada (como en los ejemplos anteriores), la franja será predeterminada del 100%. Si queremos que ocupe más o menos tendremos que escribirlo con el parámetro `width`.

A continuación, vamos a escribir una franja de 75% de ancho:

```
<hr width=75% />
```

Que quedaría de la siguiente manera:



Muchos navegadores hacen esta franja con una sombra exterior que transforma la franja en tres dimensiones. Si quieres que la franja sea simple, sin sombrita deberás escribir el parámetro "noshade".

También podemos predefinir el color que le queremos dar a la franja con el parametro "color". Es muy sencillo. Por ejemplo, si queremos que nuestra franja sea de color rojo sólo debemos ponerle la etiqueta: `<hr color="#FF0000"/>`.

Por último, puedes hacer que la franja quede alineada a un lado, a otro, o al centro del párrafo mediante el parámetro que ya vimos con anterioridad: "align". "align = center" para el centro, "align = right" para la derecha y "align = left" para la izquierda.

Como ejemplos finales vamos a hacer dos franjas. La primera va a ser una franja de grosor 3, de un ancho del 50% y alineada al centro.

```
<hr size=3 width= 50% align=center/>
```

La siguiente va a ser una franja de grosor 2, de ancho 80%, sin sombra y alineada a la derecha.

```
<hr size=2 width=80% noshade="noshade" align= right />
```

#### 4.4. ETIQUETAS ESPECÍFICAS PARA EL MARCADO DE TEXTO. ESTILOS LÓGICOS

Etiquetas propias de bloques de texto.

`<abbr>`

Pensada para etiquetar abreviaturas. El significado de la abreviatura debe escribirse mediante el atributo title.

<p><code>&lt;p&gt;Sitúa el cursor del ratón sobre la abreviatura</code></p> <p><code>&lt;abbr title="cónfer (compárese, véase)"&gt;cf&lt;/abbr&gt;</code></p> <p><code>para ver su significado.&lt;/p&gt;</code></p>	<p>Sitúa el cursor del ratón sobre la abreviatura cf para ver su significado.</p>
--	---

`<acronym>`

Pensada para etiquetar acrónimos (siglas que se pronuncian como palabras). El significado del acrónimo debe escribirse mediante el atributo title.

<p><code>&lt;p&gt;Sitúa el cursor del ratón sobre el acrónimo</code></p>	<p>Sitúa el cursor del ratón sobre el acrónimo RENFE para ver su significado.</p>
--	---

<pre>&lt;acronym title="Rogamos Empujen Nuestros Ferrocarriles Estropeados (es broma)"&gt;RENFE&lt;/acronym&gt; para ver su significado.&lt;/p&gt;</pre>	
--	--

HTML



La etiqueta <tt> no está incluida en HTML5. En su lugar se recomienda utilizar la etiqueta <abbr>.

<cite>

Pensada para identificar una cita o referencia a otras fuentes. Ahora las etiquetas <cite> encierran el título de un trabajo, como un libro, una película, una canción, etc... Los navegadores suelen mostrar la etiqueta <cite> en cursiva.

<pre>&lt;p&gt;Como escribió Robert Browning, &lt;cite&gt;The best is yet to be&lt;/cite&gt;.&lt;/p&gt;</pre>	<p>Como escribió Robert Browning, <i>The best is yet to be.</i></p>
--	---

<code>

Pensada para etiquetar fragmento de código de ordenador.

Los navegadores suelen mostrar la etiqueta <code> en tipo de letra monospace (normalmente Courier).

<pre>&lt;p&gt;El primer programa que se enseña en un lenguaje de programación suele ser algo así como &lt;code&gt;main() { printf ("Hola, mundo"); }&lt;/code&gt;&lt;/p&gt;</pre>	<p>El primer programa que se enseña en un lenguaje de programación suele ser algo así como</p> <pre>main() { printf ("Hola, mundo"); }</pre>
---	--

<del>

Pensada, junto con la etiqueta <ins>, para etiquetar modificaciones en un texto. Esta etiqueta debería etiquetar el texto que se ha eliminado de una página al revisarlo (si se quiere que se sepa que se ha eliminado, claro).

Los navegadores suelen mostrar la etiqueta <del> tachada.

<pre>&lt;p&gt;Internet Explorer 6 era un navegador</pre>	<p>Internet Explorer 6 era un navegador bastante <del>male</del> mejorable. Internet Explorer 7</p>
--	---

bastante  <del>malo</del> mejorable. Internet Explorer 7 cumplía mejor las recomendaciones del W3C.</p>	cumplía mejor las recomendaciones del W3C.
--	--

### <dfn>

Pensada para identificar la primera aparición de un término en un texto.

Los navegadores suelen mostrar la etiqueta <dfn> en cursiva.

<p>En 2005 se empezaron a comercializar discos duros que empleaban la <dfn>grabación magnética perpendicular</dfn> para aumentar la densidad de información.</p>	En 2005 se empezaron a comercializar discos duros que empleaban la <i>grabación magnética perpendicular</i> para aumentar la densidad de información.
--	---

### <em>

Pensada para resaltar una porción de texto dándole énfasis, aunque no tanto como con la etiqueta <strong>.

Los navegadores suelen mostrar la etiqueta <em> en cursiva.

<p>Es la <em>última</em> vez que te lo digo.</p>	Es la <i>última</i> vez que te lo digo.
--	---

### <ins>

Pensada, junto con la etiqueta <del>, para etiquetar modificaciones en un texto. Esta etiqueta debería etiquetar el texto que se ha añadido a una página al revisarlo (si se quiere que se sepa que se ha añadido, claro).

Los navegadores suelen mostrar la etiqueta <ins> subrayada.

<p>La verdad es que Microsoft podría mejorar su navegador<ins>, respetando las recomendaciones del W3C</ins>.</p>	La verdad es que Microsoft podría mejorar su navegador, <u>respetando las recomendaciones del W3C</u> .
---	---

### `<kbd>`

Pensada para identificar el texto que debe teclear el usuario.

Los navegadores suelen destacar la etiqueta `<kbd>` cambiando el tipo de letra.

<code>&lt;p&gt;En Amaya, para seleccionar un elemento, debes pulsar la tecla <code>&lt;kbd&gt;F2&lt;/kbd&gt;</code>.</code>	En Amaya, para seleccionar un elemento, debes pulsar la tecla F2.
---	---

### `<samp>`

Pensada para identificar un ejemplo de la salida de un programa o de un script.

Los navegadores suelen destacar la etiqueta `<samp>` cambiando el tipo de letra.

<code>&lt;p&gt;Al ejecutar tu primer programa, en la pantalla podrás leer <code>&lt;samp&gt;Hola, mundo&lt;/samp&gt;</code>.</code>	Al ejecutar tu primer programa, en la pantalla podrás leer Hola, mundo.
---	---

### `<strong>`

Pensada para resaltar una porción de texto dándole énfasis, aún más que con la etiqueta `<em>`.

Los navegadores suelen mostrar la etiqueta `<strong>` en negrita.

<code>&lt;p&gt;Es la <code>&lt;strong&gt;última&lt;/strong&gt;</code> vez que te lo digo.</code>	Es la última vez que te lo digo.
--	----------------------------------

### `<var>`

Pensada para identificar una instancia de una variable o de un argumento de programa.

Los navegadores suelen mostrar la etiqueta `<var>` en cursiva.

<code>&lt;p&gt;<code>&lt;var&gt;\$saludo&lt;/var&gt;</code> = "Hola, mundo";</code>	<code>\$saludo = "Hola, mundo";</code>
---	--

## 4.4.1. ETIQUETAS DE ESTILOS LÓGICOS APLICADAS A UN CONJUNTO DE CARACTERES

**<bdo>**

Pensada para elegir la dirección del texto (de izquierda a derecha o de derecha a izquierda). Es obligatorio especificar el atributo dir, con el valor rtl (de derecha a izquierda) o ltr (de izquierda a derecha).

`<p><bdo dir="rtl">Esta frase está  
escrita al derecho, pero debería  
leerse al revés</bdo>.</p>`

séver la esreel airebed orep ,ohcered la atircse átse esarf atsE.

**<q>**

Pensada para identificar una cita o referencia a otras fuentes. La recomendación HTML 4.0 especifica que los navegadores deben añadir automáticamente comillas al texto marcado.

`<p>Como escribió Robert Browning,  
<q>The best  
is yet to be</q>.</p>`

Como escribió Robert Browning, "The best is yet to be".

**<sub>**

Pensada para identificar texto en subíndice.

`<p>2H<sub>2</sub> + O<sub>2</sub>&rarr;  
2H<sub>2</sub>O</p>`

2H<sub>2</sub> + O<sub>2</sub> → 2H<sub>2</sub>O

**<sup>**

Pensada para identificar texto en superíndice.

`<p>El último teorema de Fermat dice que la  
ecuación  $x^{n} + y^{n} = z^{n}$  no tiene soluciones enteras  
para  $x$ ,  $y$  y  $z$  cuando  $n > 2$ .</p>`

El último teorema de Fermat dice que la ecuación  $x^n + y^n = z^n$  no tiene soluciones enteras para  $x, y$  y  $z$  cuando  $n > 2$ .

## 5. ENLACES DE HIPERTEXTO

Las principales etiquetas para enlaces son:

a) **<a> ...</a>**, con atributos básicos, de internacionalización, de eventos y de foco, Además, cuenta con los siguientes atributos específicos:

- **href="url"** indica la dirección a la que apunta el enlace. (Esta dirección puede ser relativa o absoluta si es del servidor, si no, en general se puede escribir mediante el protocolo y la dirección completa.)
- **name="valor"**, pone un nombre a la etiqueta y crea anclas a las que otro enlace puede hacer referencia usando #valor como valor del atributo href.
- **rel="..."**, indica la relación de la página actual con la página enlazada.
- **rev="..."**, indica la relación de la página enlazada con la actual.
- **target="..."**, señala el nombre de la ventana o el marco donde se mostrará el documento apuntado por el enlace.
- **hreflang="..."**, especifica el código de internacionalización de la página enlazada.

Los atributos **rel** y **rev** pueden tomar los valores: alternate, start, next, prev, chapter, help, etc.

El valor de **target** puede ser el nombre de una ventana o marco, o bien, \_blank, si se quiere abrir una ventana nueva o \_self (valor por defecto), si queremos visualizar el documento en el mismo marco o ventana. Si se pone un nombre que no existe abre una ventana nueva.

La etiqueta **<a>** es de tipo en línea.

b) **<script> ...</script>**, sólo tiene atributos específicos, que son:

- **src="..."**, indica la URL de un archivo externo que contiene el código del script.
- **type="text/javascript"**, especifica el tipo MIME (tipo de archivo/contenido).
- **defer = "defer"**, indica al navegador que no pare la carga de la página por la ejecución del script.

c) **<link>** establece una relación entre el documento y otros recursos que descarga. Esta etiqueta sólo puede aparecer en la cabecera del documento. Tiene atributos básicos, de internacionalización, de eventos (aunque no se pueden usar al estar en la cabecera) y los específicos siguientes:

- **href="url"** del documento a enlazar.
- **type = "..."** indica el tipo (tipo MIME) del documento enlazado. (Ver la lista de tipos MIME en <http://www.htmlquick.com/es/reference/mime-types.html>)
- **rel="..."** indica la relación entre el documento actual y el enlazado
- **rev="..."** indica la relación entre el documento enlazado y el actual
- **target="..."** indica dónde se presentará el documento enlazado
- **hreflang="..."** especifica el código de internacionalización del documento enlazado.

Ejemplos:

```
<link rel="stylesheet" type="text/css" href="estilos/llamativo.css"/>
<link rel="shortcut icon" type="image/ico" href="favicon.ico" />
```



### 5.1. ESTRUCTURA DE UN ENLACE: LA DIRECCIÓN DE INTERNET O URL

Un enlace se puede definir sobre un texto, una imagen, un elemento de una lista, etc. y hace referencia a una URL concreta. Ésta se compone de las siguientes partes:

Protocolo	Servidor	[Ruta]	[Archivo]	[Consulta]	[Secciones]
http:// https:// ftp:// file://	nombre.subdominio... dominio	Camino de directorios	nombre.extensión	?variable=valor	#ID (ID es el lugar concreto dentro de la página web)

Los caracteres que aparecen en los enlaces deben poder expresarse en ASCII. Sin embargo, en ocasiones, las URL contienen caracteres que no se pueden representar mediante este código y, por tanto, éstos deben ser convertidos. La llamada codificación URL convierte una URL en un formato ASCII válido.

La codificación URL sustituye estos caracteres especiales por un "%" seguido por los dos dígitos hexadecimales correspondientes al código ISO-8859-1. Como las URL no pueden contener espacios en blanco, se sustituyen por el signo +.

He aquí la codificación de algunos de ellos:

/	?	@	~	ñ	Ñ	á	Ç
%2F	%3F	%40	%7E	%F1	%D1	%E1	%E7

### 5.2. DIFERENCIAS ENTRE ENLACES ABSOLUTOS Y RELATIVOS

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:

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.

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)).

Se añaden las partes que faltan a la URL relativa para obtener la URL absoluta: <http://> + [www.ejemplo.com](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.

Elemento	Valor
Página origen	<a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</a>
Página enlazada	Página web llamada pagina2.html y que se encuentra en el mismo directorio
URL absoluta	<a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina2.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina2.html</a>
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.

Elemento	Valor
Página origen	<a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</a>
Página enlazada	Página web llamada pagina2.html y que se encuentra en el directorio superior llamadoruta2
URL absoluta	<a href="http://www.ejemplo.com/ruta1/ruta2/pagina2.html">http://www.ejemplo.com/ruta1/ruta2/pagina2.html</a>

Elemento	Valor
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.

De la misma forma, si el destino se encuentra un par de niveles por encima, se debe incluir ../ dos veces seguidas:

Elemento	Valor
Página origen	<a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</a>
Página enlazada	Página web llamada pagina2.html y que se encuentra en el directorio superior llamadoruta1
URL absoluta	<a href="http://www.ejemplo.com/ruta1/pagina2.html">http://www.ejemplo.com/ruta1/pagina2.html</a>
URL relativa	../../pagina2.html

Además de subir niveles, también se puede entrar en otros directorios para obtener los recursos:

Elemento	Valor
Página origen	<a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</a>
Página enlazada	Página web llamada pagina2.html y que se encuentra en un directorio llamado ruta4 que se encuentra en la raíz del servidor
URL absoluta	<a href="http://www.ejemplo.com/ruta4/pagina2.html">http://www.ejemplo.com/ruta4/pagina2.html</a>
URL relativa	../../ruta4/pagina2.html

Si se intentan subir más niveles de los que es posible, el navegador ignora todos los ../ sobrantes. Si la página que tiene el enlace es <http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html> y la URL relativa que se incluye es [../../../../pagina2.html](#), el navegador realmente la interpreta como [../../../pagina2.html](#).

Como el objetivo de las URL relativas es crear URL más cortas y sencillas que las URL absolutas, este método sólo se puede utilizar cuando el origen y el destino se encuentran cerca, porque de otro modo la URL relativa se complica demasiado.

### 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.

Elemento	Valor
Página origen	<a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</a>
Página enlazada	Página web llamada pagina2.html y que se encuentra en un directorio inferior llamado ruta4
URL absoluta	<a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/ruta4/pagina2.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/ruta4/pagina2.html</a>
URL relativa	<a href="#">ruta4/pagina2.html</a>

De la misma forma, se pueden indicar varios directorios seguidos para que el navegador descienda jerárquicamente por la estructura de directorios:

Elemento	Valor
Página origen	<a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</a>
Página enlazada	Página web llamada pagina2.html y que se encuentra en un directorio inferior llamado ruta6 que está dentro del directorio ruta5 y que a su vez está dentro del directorio ruta4
URL	<a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/ruta4/ruta5/ruta6/pagina2.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/ruta4/ruta5/ruta6/pagina2.html</a>

Elemento	Valor
absoluta	
URL relativa	ruta4/ruta5/ruta6/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.

Elemento	Valor
Página origen	<a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</a>
Página enlazada	Página web llamada pagina2.html y que se guarda en un directorio llamado ruta7 que se encuentra en la raíz del servidor
URL absoluta	<a href="http://www.ejemplo.com/ruta7/pagina2.html">http://www.ejemplo.com/ruta7/pagina2.html</a>
URL relativa	/ruta7/pagina2.html

Cuando la URL relativa comienza por /, el navegador considera que es la ruta completa comenzando desde la raíz del servidor, por lo que sólo le añade el protocolo y el nombre del servidor origen.

A continuación se resumen los cuatro posibles casos de URL relativas y el procedimiento que sigue el navegador para convertirlas en URL absolutas:

Si la URL relativa...	El navegador la transforma en URL absoluta...
...sólo consiste en el nombre de un recurso	...añadiendo el protocolo, servidor y ruta completa del origen del enlace
...comienza por ../	...añadiendo el protocolo y servidor del origen del enlace, subiendo un nivel en la jerarquía de directorios y añadiendo el resto de la ruta incluida en la URL relativa
...comienza por /	...añadiendo el protocolo y servidor del origen del enlace
En cualquier otro caso	...añadiendo el protocolo, servidor y ruta completa del origen del enlace, a la que se añade la ruta incluida en la URL relativa

### 5.3. ENLACES INTERNOS

El otro atributo básico de la etiqueta `<a>` es `name`, que permite definir enlaces dentro de una misma página web. Si una página es muy larga, puede ser útil mostrar enlaces de tipo "Saltar hasta la segunda sección", "Volver al principio de la página", etc.

Este tipo de enlaces son especiales, ya que la URL de la página siempre es la misma para todas las secciones y por tanto, debe añadirse otra parte a las URL para identificar cada sección.

```
<a name="primera_seccion"></a>
```

```
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris id ligula eu felis adipiscing ultrices. Duis gravida leo ut lectus. Praesent condimentum mattis ligula.</p>
```

...

```
<a name="segunda_seccion"></a>
```

```
<p>Pellentesque malesuada. In in lacus. Phasellus erat erat, lacinia a, convallis eu, nonummy et, odio. Aenean urna elit, ultrices id, placerat varius, facilisis eget, dolor.</p>
```

...

El atributo `name` permite crear "enlaces vacíos" que hacen referencia a secciones dentro de una misma página. Una vez definidos los "enlaces vacíos", es posible crear un enlace que apunte directamente a una sección concreta de una página:

```
<!-- Enlace normal a la página -->
<a href="http://www.ejemplo.com/pagina1.html">Enlace a la página 1</a>
```

```
<!-- Enlace directo a la segunda sección de la página -->
<a href="http://www.ejemplo.com/pagina1.html#segunda_seccion">Enlace a la
sección 2 de la página 1</a>
```

La sintaxis que se utiliza con estos enlaces es la misma que con los enlaces normales, salvo que se añade el símbolo # seguido del nombre de la sección a la que se apunta. Cuando el usuario pincha sobre uno de estos enlaces, el navegador accede a la página apuntada por la URL y baja directamente a la sección cuyo nombre se indica después del símbolo #.

También es posible utilizar este tipo de enlaces con URL relativas en una misma página. El siguiente ejemplo añade enlaces de tipo "Volver al inicio de la página" en varias secciones:

```
<a name="inicio"></a>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris id ligula eu
felis adipiscing ultrices. Duis gravida leo ut lectus. Praesent condimentum
mattis ligula.</p>
```

```
<a href="#inicio">Volver al inicio de la página</a>
```

...

```
<p>Pellentesque malesuada. In in lacus. Phasellus erat erat, lacinia a,
convallis eu, nonummy et, odio. Aenean urna elit, ultrices id, placerat varius,
facilisis eget, dolor.</p>
```

```
<a href="#inicio">Volver al inicio de la página</a>
```

...

Los enlaces directos a secciones también funcionan con el atributo id de cualquier elemento. El siguiente ejemplo es equivalente al ejemplo anterior:

```
<h1 id="inicio">Título de la página</h1>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris id ligula eu
felis adipiscing ultrices. Duis gravida leo ut lectus. Praesent condimentum
mattis ligula.</p>
```

```
<a href="#inicio">Volver al inicio de la página</a>
```

...

```
<p>Pellentesque malesuada. In in lacus. Phasellus erat erat, lacinia a,
convallis eu, nonummy et, odio. Aenean urna elit, ultrices id, placerat varius,
facilisis eget, dolor.</p>
```

```
<a href="#inicio">Volver al inicio de la página</a>
```

...

El nombre de la sección que se indica después del símbolo # puede utilizar el valor de los atributos id de cualquier elemento. De hecho, se recomienda utilizar los atributos id de los elementos ya existentes en la página en vez de crear "enlaces vacíos" de tipo `<a name="nombre_seccion"></a>`.



## 5.4. ENLACES ESPECIALES: CORREO ELECTRÓNICO. ENLACES DE DESCARGA

### 5.4.1. ENLACE A UN CORREO ELECTRÓNICO

```
<a href="mailto:nombre@direccion.com" title="Dirección de email para solicitar
más información">
Solicita más información

</a>
```

Al pinchar sobre el enlace anterior, se abre automáticamente el programa de correo electrónico del ordenador del usuario y se establece la dirección de envío al valor indicado después de mailto: La sintaxis es la misma que la de un enlace normal, salvo que se cambia el prefijo http:// por mailto:

La sintaxis de mailto: permite utilizarlo para otros ejemplos más complejos:

```
<!-- Envío del correo electrónico a varias direcciones a la vez -->
<a href="mailto:nombre@direccion.com,otro_nombre@direccion.com">Solicita más
información</a>
<!-- Añadir un "asunto" inicial al correo electrónico -->
<a href="mailto:nombre@direccion.com?subject=Solicitud de más
información">Solicita más información</a>

<!-- Añadir un texto inicial en el cuerpo del correo electrónico -->
<a href="mailto:nombre@direccion.com?body=Estaría interesado en solicitar más
información sobre sus productos">Solicita más información</a>
```

Todas las opciones anteriores se pueden combinar entre sí para realizar ejemplos más avanzados. Aunque el uso de mailto: puede parecer una ventaja, su uso está desaconsejado. Si se incluye una dirección de correo electrónico directamente en una página web, es muy probable que en poco tiempo esa dirección de email se encuentre llena de correo electrónico basura o "spam", ya que existen programas automáticos encargados de rastrear sistemáticamente todas las páginas web de Internet para encontrar direcciones de correo electrónico válidas.

### 5.4.2. ENLACES DE DESCARGA

Para enlazar un archivo almacenado en un servidor FTP, la parte del protocolo de la URL debe cambiar de http:// a ftp://:

```
<a href="ftp://ftp.ejemplo.com/ruta/archivo.zip" title="Archivo comprimido de
los contenidos">
Descarga un ZIP con todos los contenidos

</a>
```

## 5.5. ATRIBUTOS ESPECÍFICOS: TÍTULOS, DESTINO, ATAJO DE TECLADO, ETC

### 5.5.1. TIPO DE CONTENIDO (TYPE)

Se utiliza para notificar al navegador sobre el tipo de contenido que se enlaza. Se indica mediante una cadena de texto cuyos posibles valores también están estandarizados. Los valores de los contenidos más utilizados son los siguientes: **"text/html"** (páginas

HTML), "**image/png**" (imágenes con formato PNG), "**image/gif**" (imágenes con formato GIF), "**text/css**" (hojas de estilo CSS), "**application/rss+xml**" (archivos RSS).

### *Destino*

Se especifica mediante la etiqueta target. La etiqueta sirve para definir la forma en la que se accederá a la nueva url. Esta etiqueta la escribimos dentro del enlace para predefinirla. Puede ser de diferentes tipos:

- **\_self**: Es el valor por defecto del parámetro target. El enlace se abrirá en el mismo marco en el que está alojado.
- **\_blank**: Para hacer que ese enlace se abra en una ventana a parte.
- **\_top**: Elimina todos los marcos existente y muestra la nueva página en la ventana original sin marcos.
- **\_parent** : Muestra la nueva página en el <frameset> que contiene al marco donde se encuentra alojado el enlace.

---

#### 5.5.2. TIPO DE RELACIÓN (REL Y REV)

Los enlaces pueden proporcionar información adicional muy útil para los navegadores y para los motores de búsqueda como Google. Los atributos rel y rev permiten indicar la relación que la página actual tiene con la página a la que se enlaza (atributo rel) y la relación que tiene la página enlazada con la página actual (atributo rev).

Los tipos de relación definidos son los siguientes:

- **alternate** – Indica que es una versión alternativa al documento actual (puede ser una versión en otro idioma o una versión preparada para otro medio, como una impresora o un dispositivo móvil)
- **stylesheet** – Indica que se ha enlazado una hoja de estilos
- **start** – Indica que se trata del primer documento de una colección de documentos (por ejemplo el primer capítulo de un libro)
- **next** – Indica que es el documento que sigue al actual dentro de una secuencia lógica de documentos (por ejemplo, los capítulos de un libro)
- **prev** - Indica que es el documento que precede al actual dentro de una secuencia lógica de documentos (por ejemplo, los capítulos de un libro)
- **contents** – Indica que el recurso enlazado es el documento que contiene la tabla de contenidos de la colección de documentos (por ejemplo, el índice de un libro).
- **bookmark** – Establece el enlace actual como un "marcador" o "favorito". Un marcador es un enlace que constituye un punto de entrada muy importante dentro del documento.

### *Atributos del teclado*

Atributos	Valor	Descripción
<b>accesskey</b>	Carácter	Asigna una tecla de acceso a un elemento. Al pulsar esa tecla el foco se dirige hacia ese elemento.
<b>tabindex</b>	Número	Especifica el orden de tabulación del elemento dentro del documento actual

## 6. IMÁGENES

En un sitio web, se pueden usar imágenes de contenido e imágenes de adorno y diseño. Las primeras deben incluirse mediante HTML y las segundas, en la medida de lo posible, con hojas de estilo CSS.

Los formatos de imagen más estandarizados en páginas web son JPG, PNG y GIF.

Existen diferentes etiquetas que permiten insertar imágenes y multimedia:

a) **<img>**, para insertar una imagen. Posee atributos básicos, de internacionalización, de eventos y los siguientes específicos:

- **src="url"**, para indicar la localización de la imagen.
- **alt="texto descriptivo"**, texto que aparece si no se carga la imagen.
- **title="texto descriptivo"**, título de la imagen.
- **width="valor"**, especifica la anchura de la imagen en píxeles o porcentaje.
- **height="valor"**, especifica la altura de la imagen en píxeles o porcentaje.
- **usemap="#nedelmap"**, mapa de imágenes para la imagen.

Es importante tener en cuenta que las imágenes tienen un comportamiento en línea y que es obligatorio incluir los atributos src y alt cuando se crea una imagen.

b) **<map>**, para crear un mapa de imágenes. Cuenta con atributos básicos, de internacionalización y de eventos y puede ser en línea o de bloque. Se usa junto a la etiqueta **<area>**.

c) **<area>**, para definir los distintos trozos del mapa. Cuenta con atributos básicos, de internacionalización, de eventos y los siguientes atributos específicos:

- **shape="..."**, para indicar la forma geométrica del trozo (rect, circle, poly).
- **coords="x..."**, para señalar las coordenadas del trozo de imagen.
- **href="url"**, especifica la dirección enlazada desde el trozo de imagen.
- **target="..."**, indica dónde se presentará el contenido apuntado en la url.

Ejemplo:

```

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

```
<area shape="circle" coords="x, y, r"/>
<area shape="poly" coords="x1, y1, x2, y2, x3, y3, ..."/>
</map>
```

d) Aparte de éstas, en HTML existían, para multimedia, diversos tipos de etiquetas que, posteriormente, se estandarizaron en una sola, llamada **<object>**. Se trata de una etiqueta contenedora de objetos (video, audio, applets Java, Activex, PDF y Flash), la cual incluye otra, llamada **<param/>**, que permite pasar los parámetros adecuados al objeto correspondiente. Es de tipo bloque.

Ejemplo:

```
<object classid="CLSID:XXXXXXXXXX" width="800" height="600" type="video/wmv">
<param name="FileName" value="archive.wmv" />
<param name="autoStart" value="true" />
</object>
```

La etiqueta **<object>** tenía la intención de sustituir los elementos **<img>** y **<applet>**. Sin embargo, a causa de errores y la falta de soporte de navegadores, esto no ha sucedido. El soporte para la etiqueta en los navegadores depende del tipo de objeto, y lamentablemente, los principales navegadores utilizan códigos diferentes para cargar el mismo tipo de objeto.

## 6.1. FORMATOS DE IMÁGENES

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.

- **Formato GIF:** El Graphics Interchange Format (GIF) es comúnmente usado para la transferencia de documentos entre diferentes plataformas por su versatilidad, tanto en tamaño como en aplicaciones que lo reconocen. Es un formato de alta compresión por lo que minimiza los tiempos de transferencia a través de líneas telefónicas. Este formato soporta resoluciones de imagen de 1 bit a 8 bits, es decir hasta 256 colores. Se recomienda comprimir arte de línea y otras imágenes que utilicen pocos colores con este formato, debido a su modo de actuar. Mientras más zonas de color similar continuas (horizontalmente) existan en un documento, la compresión será mejor, y el archivo ocupará muy poco espacio.
- **Formato JPEG:** Las siglas de este formato significan Joint Photographic Experts Group. Este formato economiza la manera en que la información de una imagen es guardada, descartando información basura. Cuando se comprime una imagen con este formato, la

imagen ya no será la misma, habrá perdido valiosa información por lo que no es recomendable editar una imagen después de haberla comprimido con este formato. Generalmente la diferencia entre la imagen original y la imagen comprimida no es perceptible para el ojo humano, pero si existe. Este método de compresión ofrece diferentes opciones en cuanto a la relación calidad y tamaño que ocupa en disco, comprimiendo de ratios desde 5:1 hasta 15:1. Este formato es capaz de guardar información para imágenes de 24 bits, es decir para 16 millones de colores, por lo que produce una calidad más alta con respecto al formato GIF, aunque no siempre un menor tamaño. Se recomienda comprimir fotografías y dibujos complejos con este formato.

- **Formato png:** El formato PNG permite almacenar imágenes en blanco y negro (una profundidad de color de 16 bits por píxel) y en color real (una profundidad de color de 48 bits por píxel), así como también imágenes indexadas, utilizando una paleta de 256 colores. Además, soporta la transparencia de canal alfa, es decir, la posibilidad de definir 256 niveles de transparencia, mientras que el formato GIF permite que se defina como transparente sólo un color de la paleta. También posee una función de entrelazado que permite mostrar la imagen de forma gradual. La compresión que ofrece este formato es (compresión sin pérdida) de 5 a 25% mejor que la compresión GIF. Por último, el PNG almacena información gama de la imagen, que posibilita una corrección de gama y permite que sea independiente del dispositivo de visualización. Los mecanismos de corrección de errores también están almacenados en el archivo para garantizar la integridad.
- **Formato SVG:** para la representación de gráficos vectoriales y animaciones en dispositivos de pantalla pequeña, que cuenta con dos perfiles, SVG Tiny para la representación de gráficos vectoriales en teléfonos móviles y SVG Basic para PDAs.

## 6.2. CARACTERÍSTICAS DE IMÁGENES: TAMAÑO, TÍTULO, TEXTOS ALTERNATIVOS

### TAMAÑO

Los atributos **width** y **height** se utilizan para indicar la anchura y altura con la que se muestran las imágenes, por lo que son los más contradictorios. Como ya se ha comentado, HTML estructura de forma correcta los contenidos de la página y CSS define el aspecto gráfico con el que se muestran los contenidos. En principio, la anchura y la altura con la que se muestra una imagen es parte de su aspecto gráfico, por lo que debería ser propio de CSS y no de (X)HTML.

Sin embargo, en la práctica no es viable establecer la anchura y altura de todas las imágenes de contenidos mediante CSS. Si el sitio web dispone de muchas imágenes, la sobrecarga de estilos diferentes que debería definir CSS sería contraproducente. Por este motivo, los atributos **width** y **height** son la excepción a la norma de que el código HTML no haga referencia al aspecto de los contenidos.

Ejemplo:

```

```

```

```

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.

```
<div>
```

```

```

```
</div>
```

El ejemplo anterior mezcla los dos tipos de medidas que se pueden utilizar, para indicar que la foto tiene una anchura igual al 30% de la anchura del elemento `<div>` que la contiene y una altura de 350 píxel.

La anchura/altura con la que se muestra una imagen no tiene que coincidir obligatoriamente con la anchura/altura real de la imagen. Sin embargo, cuando estos valores no coinciden, las imágenes se muestran deformadas y el aspecto final es muy desagradable.

Si solamente se establece la altura de la imagen, el navegador calcula la anchura necesaria para que se mantenga la proporción de la imagen. De la misma forma, si sólo se establece la anchura de la imagen, el navegador calcula la altura que hace que la imagen se siga viendo con las mismas proporciones.

---

## TÍTULO

El atributo **title** permite añadir información a la imagen. Si se pasa el ratón por encima de la imagen y se deja un momento se podrá leer en un cuadro de texto el contenido de este atributo.

---

## TEXTOS ALTERNATIVOS


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.

### 6.3. ENLACES EN IMÁGENES

Con frecuencia se emplean imágenes como enlaces. A veces son enlaces a las mismas imágenes con más resolución, como en el caso de fotos. Otras veces nos llevan a una localización cualquiera.

Para que una imagen sea un enlace hay que colocarla en el lugar del texto del enlace, si además queremos que vaya a otra imagen, ésta la debemos poner en el lugar de la URL.

Veamos unos ejemplos Supongamos para ello, que la estructura de nuestras páginas es la indicada en la imagen. Si escribo en la página actual localizada en Aprendiendo Html

 `<A HREF="#inicio"></A>`  
 obtengo una imagen que al pulsarla me lleva al inicio de esta página.

Para que una imagen sea un enlace a otra imagen se debe escribir:

```
<a href="gifs/sky.jpg"></a>
```

## 7. LISTAS

HTML ofrece a los autores varios mecanismos para especificar listas de información. Todas las listas deben contener uno o más objetos de lista. Las listas pueden contener:

- Información no ordenada.
- Información ordenada.
- Definiciones.

La lista anterior, por ejemplo, es una lista no ordenada, creada con el elemento UL:

```
<ul>
<li>información no ordenada.</li>
<li>información ordenada. </li>
<li>definiciones. </li>
</ul>
```

Las listas son estructuras que permiten enumerar elementos por lo que, habitualmente, se usan en clasificaciones, instrucciones y barras de navegación (menús).

Las etiquetas que se utilizan para trabajar con listas son:

- `<ul>`: Lista sin indicador de orden.
- `<ol>`: Lista ordenadas (numeradas).
- `<li>`: Elemento de una lista

Los tres son elementos de bloque.

He aquí un ejemplo de lista formada por dos elementos:

```
<ul>
<li>Elemento1</li>
<li>Elemento2</li>
</ul>
```

El atributo **type** especifica las opciones de representación en el navegador.

Para el elemento UL, los valores posibles del atributo type son disc (disco), square (cuadrado), y circle (círculo). El valor por defecto depende del nivel de anidamiento de la lista actual. Estos valores no distinguen entre mayúsculas y minúsculas.

### 7.1. ORDENACIÓN DE LISTAS

Una lista ordenada, creada por medio del elemento OL, debería contener información para la cual debe ponerse énfasis en el orden, como en una receta de cocina:

1. Mezcle los ingredientes secos íntimamente.
2. Vierta los ingredientes líquidos.
3. Remueva durante 10 minutos.
4. Hornee durante una hora a 300 grados.

El anterior ejemplo se traduce a html como:

```
<ol>
<li>Mezcle los ingredientes secos íntimamente.</li>
<li>Vierta los ingredientes líquidos.</li>
<li>Remueva durante 10 minutos.</li>
<li>Hornee durante una hora a 300 grados.</li>
</ol>
```

Para el elemento OL, los valores posibles del atributo **type** se resumen en la siguiente tabla (la diferencia entre mayúsculas y minúsculas es importante):

Tipo	Estilo de numeración	
1	números arábigos	1, 2, 3, ...
a	alfabética en minúsculas	a, b, c, ...
A	alfabética en mayúsculas	A, B, C, ...
i	números romanos en minúsculas	i, ii, iii, ...
I	números romanos en mayúsculas	I, II, III, ...

Obsérvese que el atributo type está desaprobado y que los estilos de las listas deberían especificarse mediante hojas de estilo.

### 7.2. ANIDAMIENTO EN LISTAS

Las listas se pueden anidar, es decir, un elemento de una lista puede ser, a su vez, otra lista. En este caso, habrá que poner especial cuidado al colocar las etiquetas de cierre y apertura para que no se entrecrucen.



Las listas además de estar anidadas, pueden usar al mismo tiempo tipos diferentes de listas, como en el ejemplo siguiente, que es una lista de definiciones que contiene una lista no ordenada (los ingredientes) y una lista ordenada (los pasos a seguir):

**Ingredientes:**

- 100 g de harina
- 10 g de azúcar
- 1 taza de agua
- 2 huevos
- sal, pimienta

**Pasos a seguir:**

1. Mezcle los ingredientes secos íntimamente.
2. Vierta los ingredientes líquidos.
3. Remueva durante 10 minutos.
4. Hornear durante una hora a 300 grados.

**Notas:**

Puede añadir uvas para mejorar la receta.

## 8. TABLAS

Las tablas permiten representar datos en filas y columnas y, aunque durante cierto tiempo se usaron para maquetar información, esta práctica está totalmente desaconsejada.

Es responsabilidad de quien escribe la tabla asegurarse de que el número de datos se ajusta al de filas y columnas. Si no se hace así, el navegador no informará de los errores sino que, tan sólo, mostrará huecos en determinados lugares o presentará un comportamiento extraño.

### 8.1. ESTRUCTURA BÁSICA

Una tabla se organiza en filas y columnas y se representa en html definiendo, en primer lugar, la tabla en sí, a continuación, las filas de la tabla y para cada fila los elementos que contiene, que son las celdas que forman la tabla. Además, se pueden añadir elementos de encabezado y pie de tabla para usar estilos.

Las etiquetas para tablas son:

- **<table>**, que posee atributos básicos, de internacionalización y de eventos. Además, posee el atributo específico **summary="..."**, para escribir un breve resumen de los contenidos de la tabla. Es de tipo bloque.
- **<tr>**, señala cada fila de la tabla. Tiene atributos básicos, de internacionalización y de eventos.
- **<td>**, señala cada elemento de una fila. Tiene atributos básicos, de internacionalización, de eventos y los atributos específicos:
  - **abbr="..."**, para escribir un resumen de la celda.

- **colspan**="número", expande el número de celdas.
- **rowspan**="número", expande el número de filas.

Se puede poner una tabla dentro de otra, siempre que se coloque dentro de una celda.

- **<th>**, posee atributos básicos, de internacionalización, de eventos y los mismos atributos específicos que **<td>**. Permite indicar las celdas que forman la cabecera de la tabla.
- **<caption>**, posee atributos básicos, de internacionalización y de eventos. Sólo puede aparecer una vez y se escribe, justo, después de la etiqueta **<table>**. Se utiliza para poner un título o leyenda a la tabla.
- **<thead>**, **<tfoot>**, **<tbody>** son etiquetas para crear tablas de forma avanzada. Su principal utilidad es agrupar bloques de la tabla y aplicar estilos a cada bloque, sin tener que ir poniendo el estilo en cada fila o celda. Las tres poseen los atributos básicos, de internacionalización y de eventos. Además, son de tipo bloque.

Una tabla tiene tres secciones distintas: **encabezado, cuerpo y pie**. La sección (o celda) de encabezado se define mediante un elemento **thead**, el cual contiene información correspondiente al encabezado, como los nombres de las columnas. Cada elemento **tr** define una sola fila de la tabla. Las columnas en la sección de encabezado se definen mediante los elementos **th**. La mayoría de los exploradores centran el texto que tiene el formato de los elementos **th** (columna de encabezado de la tabla) y lo muestran en negritas. Los elementos de encabezado de una tabla se anidan dentro de los elementos de fila. La sección del pie se define mediante un elemento **tfoot** (pie de la tabla). Por lo general, el texto que se coloca en el pie incluye los resultados de cálculos y las notas al pie. Al igual que las demás secciones, el pie de la tabla puede contener filas, y cada fila puede contener columnas.

La sección del cuerpo, o cuerpo de la tabla, contiene los datos principales de la misma. El cuerpo de la tabla se define en un elemento **tbody**. En el cuerpo, cada elemento **tr** especifica una fila. Las celdas de datos contienen piezas de datos individuales, y se definen con elementos **td** (datos de tabla) dentro de cada fila.

## 8.2. FORMATO DE CONTENIDO DE CELDAS

La etiqueta **<col>** se utiliza para asignar los mismos atributos a varias columnas de forma simultánea.

De esta forma, la etiqueta **<col>** no agrupa columnas, sino que sólo asigna atributos comunes a varias columnas.

La siguiente imagen muestra una tabla que hace uso de la etiqueta **<col>**:



AÑO	Producto A	Producto B	Producto C	Producto D
N-3	-	-	-	-
N-2	3	5	8	4
N-1	4	4	7	3
N	5	7	6	2

Ejemplo de tabla avanzada que usa la etiqueta col

El código HTML necesario para crear la tabla anterior se muestra a continuación:

```
<table summary="Análisis de ventas anuales">

<caption>Análisis de ventas anuales</caption>
<col style="width:10%;" />
<col style="width:30%;" />

<thead>
<tr>
<th scope="col">AÑO</th>
<th scope="col">Producto A</th>
<th scope="col">Producto B</th>
<th scope="col">Producto C</th>
<th scope="col">Producto D</th>
</tr>
</thead>

<tbody>
<tr>
<th scope="row">N-3</th><td>-</td><td>-</td><td>-</td><td>-</td>
</tr>
<tr>
<th scope="row">N-2</th><td>3</td><td>5</td><td>8</td><td>4</td>
</tr>
<tr>
<th scope="row">N-1</th><td>4</td><td>4</td><td>7</td><td>3</td>
</tr>
<tr>
<th scope="row">N</th><td>5</td><td>7</td><td>6</td><td>2</td>
</tr>
</tbody>
</table>
```

Por otra parte, la etiqueta `<colgroup>` se emplea para agrupar de forma estructural varias columnas de la tabla. La forma habitual de indicar el número de columnas que abarca la agrupación es utilizar el atributo `span`, que establece el número de columnas de cada agrupación.

La siguiente imagen muestra una tabla avanzada con una agrupación de columnas realizada con la etiqueta `<colgroup>`:

AÑO	Producto A	Producto B	Producto C	Producto D
N-3	-	-	-	-
N-2	3	5	8	4
N-1	4	4	7	3
N	5	7	6	2

Ejemplo de tabla avanzada que usa la etiqueta colgroup

El código HTML necesario para crear la tabla anterior se muestra a continuación:

```
<table summary="Análisis de ventas anuales">

<caption>Análisis de ventas anuales</caption>
<colgroup span="1" style="color:red;" />
<colgroup span="3" style="color:blue;" />

<thead>
<tr>
<th scope="col">AÑO</th>
<th scope="col">Producto A</th>
<th scope="col">Producto B</th>
<th scope="col">Producto C</th>
<th scope="col">Producto D</th>
</tr>
</thead>
<tbody>
<tr>
<th scope="row">N-3</th><td>-</td><td>-</td><td>-</td><td>-</td>
</tr>
<tr>
<th scope="row">N-2</th><td>3</td><td>5</td><td>8</td><td>4</td>
</tr>
<tr>
<th scope="row">N-1</th><td>4</td><td>4</td><td>7</td><td>3</td>
</tr>
<tr>
<th scope="row">N</th><td>5</td><td>7</td><td>6</td><td>2</td>
</tr>
</tbody>
</table>
```

El uso de las etiquetas <col> y <colgroup> no está muy extendido, debido a que la mayoría de navegadores no soportan muchas de sus funcionalidades.

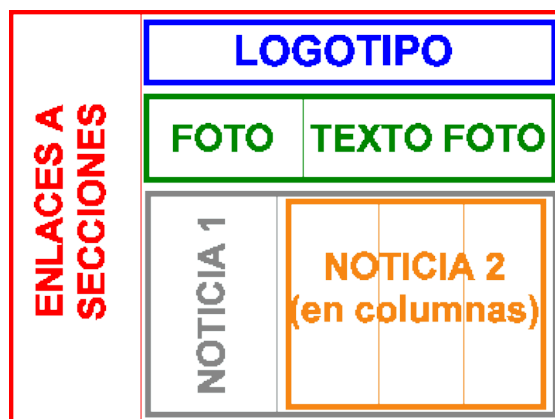
### 8.3. TABLAS ANIDADAS

En ocasiones, especialmente cuando utilizamos las tablas como recurso de formato para distribuir la información dentro de la página, nos interesa introducir una tabla dentro de una

celda perteneciente a una tabla creada anteriormente. Esto nos permitirá, por ejemplo, hacer que convivan tablas con borde junto a tablas sin borde.

Supongamos, por ejemplo, que estamos diseñando una página para introducir en ella los contenidos de una revista escolar. En la siguiente imagen podemos contemplar una posible estructura en la que vamos anidando sucesivas tablas.

Cada color corresponde a una tabla que incluye en su interior las que son de otro color diferente.



Posible estructura de una página de revista electrónica

Como ya conocemos los códigos básicos que constituyen una tabla entenderemos rápidamente la estructura general. Para facilitar la interpretación se han recogido las etiquetas correspondientes a cada tabla con el mismo color con el que se presentan en el gráfico y se han ido sangrando las etiquetas correspondientes a cada tabla, de forma que se pueda ver con facilidad en qué celda se encuentra insertada cada una de ellas.

Introduciendo el texto correspondiente a cada zona de la hipotética pantalla obtendríamos una presentación coincidente con el ejemplo gráfico.

```
<table border=0>
<tr>
<td>Enlaces a secciones</td>
<td>
    <table border=0>
    <tr>
    <td>Logotipo </td>
    </tr>
    </table>

    <table border=0>
    <tr>
    <td>Foto </td>
    <td>Texto foto </td>
    </tr>
    </table>

    <table border=0>
    <tr>
    <td>Noticia 1 </td>
    <td>
        <table border=0>
        <tr>
```

```

        <td>Noticia en columnas </td>
        <td>Columna 2 </td>
        <td>Columna 3 </td>
    </tr>
</table>
</td>
</tr>
</table>
</td>
</tr>
</table>

```

## 9. MARCOS (FRAMES)

Podríamos decir que una estructura de marcos es una artimaña que hace posible dividir la pantalla en varias zonas que nos permiten presentar en cada una de ellas una página independiente. Los marcos permiten organizar la presentación de varios documentos HTML simultáneamente en la ventana del navegador. La forma de conseguirlo es dividiendo ésta en diferentes áreas y asignando, a cada una de ellas, uno de los documentos a mostrar.

### 9.1. CREACIÓN DE MARCOS

Su declaración tiene esta estructura:

```

<frameset [cols="..." | rows="..."]>
<frame src="url1" name="nombre_marco1"/>
<frame src="url2" name="nombre_marco2"/>
...
<noframes> Texto que aparece si el navegador no acepta marcos </noframes>
</frameset>

```

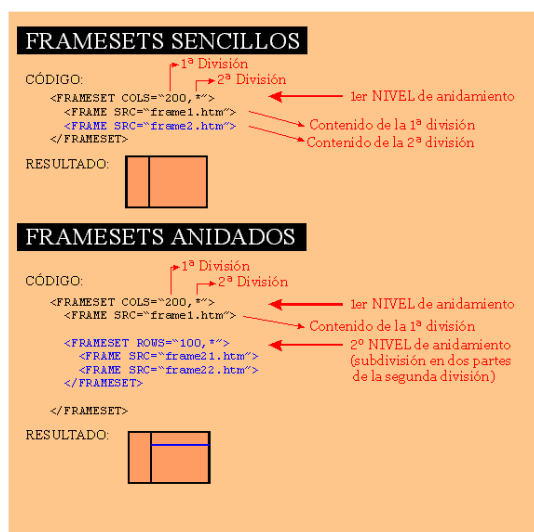
El tamaño de una fila o columna se puede indicar por el tanto por ciento que ocupa en el espacio total de las ventanas o por el número de píxeles. Por ejemplo `rows="20%,*,40%"`.

La etiqueta `<frame>` que sustituye a `<body>`, aparecerá tantas veces como filas o columnas se hayan indicado y, en ella, se especificará toda la información relativa a un marco. El atributo `src` se usa para indicar la página que se va a cargar en él y el atributo `name` para nombrar o identificar el marco.

Los marcos no son muy utilizados hoy en día, además, sólo son válidos en los documentos especificados como tipo `frameset` en su DTD. En la actualidad se usa otro tipo de marco que permite insertar un documento dentro de otro. El marco insertado se denomina "flotante" y puede considerarse como un agujero que se abre en una página web para mostrar otra. Admite atributos básicos, de internacionalización, de eventos y los siguientes específicos:

- **src**="url", página que se carga en el marco.
- **width**="número", anchura en píxeles o en %.
- **height**="número", altura en píxeles o en %.
- **scrolling**="yes" | "no" | "auto". Indica la aparición o no de barras de desplazamiento laterales.
- **frameborder**="1" | "0". Indica si el marco tiene borde o no.
- **name**="...", nombre para el marco.

- **marginwidth**="número", anchura en píxeles de los márgenes superior e inferior.
- **marginheight**="número", altura en píxeles de los márgenes izquierdo y derecho.



## 9.1. MARCOS INCRUSTADOS (IFRAMES)

Además de poder crear una estructura de marcos convencional disponemos también de una posibilidad adicional que consiste en insertar dentro del flujo de la página una ventana en la cual podremos hacer que se muestren los contenidos de una página web. Es lo que se denomina marco en línea, in line frame o, utilizando el nombre de la etiqueta que sirve para crearlo, iframe.

Para crear un marco en línea utilizaremos la etiqueta **<iframe>** y su correspondiente cierre **</iframe>**

Lógicamente lo primero que tendremos que hacer será adjudicarle unas medidas al marco e indicar cual es la página que se mostrará en su interior.

- **width**: será el atributo que nos indicará la anchura del marco.
- **height**: indicará la altura
- **src**: será el atributo con el que indiquemos cuál es la página que se mostrará en el hueco que hemos definido
- **name**: Servirá para identificar el marco en línea, de forma que podamos indicar que un enlace se cargue en dicho espacio.
- **frameborder** : Indicará si el marco lleva o no lleva borde. Sus valores pueden ser 0 y 1, siendo este último el valor por defecto
- **marginwidth** y **marginheight**: indican, mediante un entero positivo la distancia que separará el contenido del marco de los bordes internos del mismo.
- **scrolling**: utilizando el valor no haríamos que no se presentaran barras de desplazamiento, por lo que habría que haber comprobado que el contenido se visualiza en cualquier plataforma sin necesidad de ellas.

Para hacer que el destino de un vínculo se cargue dentro de un iframe lo único que tendremos que hacer al crear el vínculo es especificar el valor target con el nombre que le hayamos adjudicado al iframe. Lógicamente es imprescindible que el iframe lleve el atributo name.

Cuando se crea un vínculo desde una página para cargar un contenido en un iframe insertado en ella, el atributo target deberá llevar como valor el nombre adjudicado al iframe mediante el atributo name.

Cuando quieras utilizar vínculos desde dentro de un iframe de forma que se cargue otra página dentro del propio iframe el atributo target deberá indicar esta situación adoptando el valor target="\_self"

Como el iframe es un elemento que, independientemente del tamaño que ocupe, se inserta como un elemento en línea dentro del flujo de la página se comporta de forma muy similar a cualquier otro elemento en línea, tal como una palabra.

Si algún navegador o en términos más genéricos, agente de usuario, no fuera capaz de mostrar el contenido del iframe podríamos paliar fácilmente este error. El texto que se introduzca antes del cierre `</iframe>` se ignorará siempre que se pueda mostrar el contenido del marco en línea, pero se mostrará en caso contrario, así que bastaría con que ese texto fuera el enlace a la página que mostramos en el iframe como veremos en el ejemplo.

Lo que tienes a continuación es, exactamente, el código que hace que se muestre el iframe que se puso como ejemplo al principio de la página.

```
<iframe name="muestra" src="muestraiframe.htm" width=200px height="100px">
```

```
Ver un <a href="muestraiframe.htm">ejemplo</a>
```

```
</iframe>
```

El texto que aparece dentro del iframe sólo lo habrás visto en el caso de que tu navegador no soporte esta etiqueta y está ahí para que todos los visitantes pudieran acceder al contenido del archivo que pretendemos que se muestre en ese espacio.

## 10. FORMULARIOS

HTML es un lenguaje de marcado cuyo propósito principal consiste en estructurar los contenidos de los documentos y páginas web. Sin embargo, HTML también incluye elementos para crear aplicaciones web. El estándar HTML/XHTML permite crear formularios para que los usuarios interactúen con las aplicaciones web.

Un formulario HTML es una sección de un documento que contiene contenido normal, código, elementos especiales llamados controles (casillas de verificación (checkboxes), radiobotones (radio buttons), menús, etc.), y rótulos (labels) en esos controles. Los usuarios normalmente "completan" un formulario modificando sus controles (introduciendo texto, seleccionando objetos de un menú, etc.), antes de enviar el formulario a un agente para que lo procese (p.ej., a un servidor web, a un servidor de correo, etc.)

Por lo general, los datos que introducen los usuarios en una página Web se envían a un servidor Web, el cual proporciona acceso a los recursos de un sitio (por ejemplo, documentos en XHTML,



imagenes). Estos recursos se ubican en el mismo equipo que el servidor Web, o en un equipo al que el servidor Web puede acceder a través de la red. Cuando un explorador solicita una página Web o un archivo que se encuentra en un servidor, este procesa la solicitud y devuelve el recurso solicitado. Una solicitud contiene el nombre y la ruta del recurso deseado, junto con el método de comunicación (conocido como protocolo). Los documentos en XHTML utilizan el Protocolo de transferencia de hipertexto (HTTP).

Dentro de la etiqueta form, se ponen los controles, cuyas etiquetas se nombran a continuación.

- **<input>**, con atributos básicos, de internacionalización, de eventos, de foco y los siguientes:
  - **type**="...", con diez posibles valores (text, password, checkbox, radio, submit, reset, file, hidden, image, button) y es de tipo en línea. Dependiendo del valor, se pueden utilizar otros atributos.
  - **value**="texto...", indica el valor del control.
  - **size**="número", especifica el ancho del control en número de caracteres
  - **maxlength**="número", En los controles text y password indica la longitud máxima de caracteres que se puede teclear.
  - **checked**="checked" indica si el control está marcado en los controles radio y checkbox.
  - **disabled**="disabled" desactiva el control y, por tanto, no se envía el valor al servidor.
  - **readonly**="readonly" pone el control en sólo lectura. (sólo para los controles text o password)
  - **src**="url", para indicar la ruta de la imagen que actuará como un botón de envío.
  - **alt**="texto", descripción o texto alternativo que aparece cuando no se carga la imagen.
- **<fieldset>** sirve para hacer agrupaciones lógicas de controles para lo cual dibuja un marco que engloba varios de ellos. Es de tipo bloque y tiene atributos básicos, de internacionalización y de eventos.
- **<legend>**, se escribe justo debajo de la apertura del fieldset y le pone un título al grupo.
- **<label>**, tiene atributos básicos, de internacionalización y de eventos. Define una etiqueta que puede asociarse con un control de formulario mediante el atributo específico for="id", siendo "id" el identificador del control del formulario al que se asocia. Mejora la entrada de usuario, ya que si se hace un clic de ratón en el texto que muestra <label> conmuta el estado del control asociado.
- **<textarea>**, de tipo línea, con atributos básicos, de internacionalización, de eventos, de foco y los siguientes específicos:
  - **cols**="número", número de columnas.
  - **rows**="número", número de filas.
  - **disabled**="disabled", desactiva el textarea con lo que no se envía información.
  - **readonly**="readonly", de sólo lectura por lo que no se puede escribir.
- **<select>**, muestra listas desplegables o desplegadas. Es de tipo línea y admite atributos básicos, de internacionalización, de eventos, de foco y los siguientes específicos:
  - **size**="número", fija el tamaño inicial de la lista. Si es mayor que uno la lista aparece desplegada.
  - **multiple**="multiple" permite elegir más de una opción.
  - **disabled**="disabled", desactiva el control.

- **<option>**, indica cada una de las opciones de la lista. Admite los siguientes atributos específicos:
  - **selected**="selected", opción marcada por defecto.
  - **value**="value", valor de la variable de la opción correspondiente que se envía al servidor.
  - **disabled**="disabled", desactiva esa opción de la lista.
- **<optgroup>**, sirve para englobar etiquetas **<option>** y hacer subgrupos. Tiene atributos básicos, de internacionalización, de eventos y los siguientes específicos:
  - **label**="...", texto para la cabecera del grupo.
  - **disabled**="disabled", desactiva el grupo.

Ejemplo:

```
<select>
  <optgroup label="vehículos suecos">
    <option value="Volvo">Volvo</option>
    <option value="Saab">Saab</option>
  </optgroup>
  <optgroup label="vehículos alemanes">
    <option value="Opel">Opel</option>
    <option value="Audi">Audi</option>
  </optgroup>
</select>
```

## 10.1 NUEVOS ATRIBUTOS EN HTML5

### ATRIBUTO PLACEHOLDER HTML5

Este **nuevo atributo** permite al usuario ver una especie de leyenda o label dentro del input. Cuando el usuario comience a escribir sobre él este **placeholder** desaparecerá.

Es muy útil para guiar al usuario sobre el tipo de contenido a rellenar en el formulario.

Podéis saber más sobre el atributo en su post '[Placeholder HTML5: Un atributo para inputs de formularios](#)'.

Os dejo una imagen de ejemplo:

```
<form action="https://www.dominio.com/atributo-placeholder-html5.php"
name="formulario">
  <input type="text" name="nombre" placeholder="inserta tu nombre">
  <input type="submit" value="enviar">
</form>
```



inserta tu nombre enviar

---

### ATRIBUTO AUTOFOCUS HTML5

La función de este atributo es poner el cursor de manera activa en un input del formulario sin necesidad de hacer click en él.

Podéis saber más sobre el atributo en su post '[Autofocus HTML5: Un atributo para seleccionar inputs de formularios](#)'.

Imagen de ejemplo:

Introduzca su nombre:

<br><br>

Introduzca su edad:

Introduzca su nombre:

Introduzca su edad:

---

### ATRIBUTO AUTOCOMPLETE HTML5

Este nuevo elemento nos va ayudar a recordar, autocompletar y/o **sugerir los valores insertados anteriormente** en el mismo formulario.

Podéis saber más sobre el atributo en su post '[Autocomplete HTML5: Atributo para autocompletar datos en formularios](#)'.

```
<form action="#" autocomplete="on">
  Nombre: <input type="text" name="nombre" autocomplete="on"/>
  Apellidos: <input type="text" name="apellidos" autocomplete="off" />
</form>
```

---

### ATRIBUTOS HEIGHT Y WIDTH HTML5

Estos atributos solo se pueden aplicar a los [input](#) de tipo 'image', así de esta manera podemos jugar con las dimensiones de la imagen que sustituye al botón 'submit' para enviar [formularios](#).

```
<form action="/demos/2014/030-ejemplo-atributos-height-width-html5.php"
method="POST">
  Nombre: <input type="text" name="nombre"><br><br>
  <input type="image" src="/wp-content/uploads/2013/04/html5.png"
alt="Enviar" width="100" height="50">
</form>
```

Podéis saber más sobre el atributo en su post '[Atributos height y width HTML5 para inputs de tipo image](#)'.

---

### ATRIBUTO LIST HTML5

Este atributo sirve para asociar el elemento de tipo 'datalist' con un input, de esta manera el input tendrá establecido por defecto una serie de valores para facilitar la inserción de datos.

```
<form name="formulario" id="formulario" action="032-atributo-list-html5.php" method="POST">
```

```
<!-- La datalist con id 'listas' y sus diferentes valores -->
```

Escribe un color (Azul, rojo, amarillo, negro, verde):

```
<datalist id="listas">
  <option value="azul">
  <option value="rojo">
  <option value="amarillo">
  <option value="negro">
  <option value="verde">
</datalist>
```

```
<!-- Asociamos al input la datalist 'listas' -->
  <input name="color" list="listas">
  <input type="submit" value="Enviar">
</form>
```

Podéis saber más sobre el atributo en su post '[Atributos height y width HTML5 para inputs de tipo image](#)'.

---

### ATRIBUTOS MIN Y MAX HTML5

Sirven para fijar los valores mínimos y máximos de los [input](#) con valor numérico o de fecha.

Podéis saber más sobre el atributo en su post '[Atributos min y max HTML5 para los input numéricos y de fecha](#)'.

---

### ATRIBUTO MULTIPLE HTML5

Con el **atributo multiple HTML5** se pueden seleccionar varios ficheros o emails en el envío de formularios, es compatible con los [input](#) de tipo **file** o [email](#).

Podéis saber más sobre el atributo en su post '[Atributo multiple HTML5: Seleccionar varios archivos o emails en formularios](#)'.

---

### ATRIBUTO NOVALIDATE HTML5

El **atributo novalidate HTML5** hace que el elemento en cuestión no se valide al enviar el formulario, es decir, si el [formulario](#) tiene varios input que disponen de validación automática por parte del navegador podemos anular esta validación por cada [input](#).

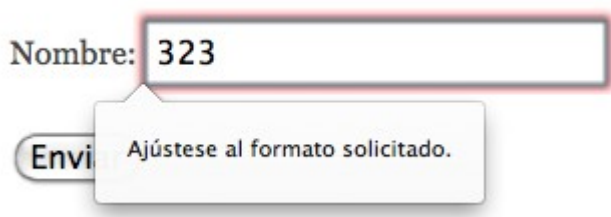
Podéis saber más sobre el atributo en su post '[Atributo novalidate HTML5: Anular la validación de inputs de formularios](#)'.

---

### ATRIBUTO PATTERN HTML5

Con el **atributo pattern HTML5** podremos delimitar mediante [expresiones regulares](#) el contenido insertado por el usuario en los inputs de los formularios.

De no cumplirse la **expresión regular** sale un mensaje similar a este:



Nombre: 323

Envíajústese al formato solicitado.

Podéis saber más sobre el atributo en su post '[Atributo pattern HTML5: Expresiones regulares para la validación de inputs](#)'.

#### ATRIBUTO REQUIRED HTML5

Con el **atributo required HTML5** podemos establecer que un input de [formulario](#) sea de rellenado obligatorio.

En caso de no rellenar el campo obligatorio sale un mensaje automático de aviso.



Fecha de nacimiento: dd/mm/aaaa

Enviar

Completa este campo

Podéis saber más sobre el atributo en su post '[Atributo required HTML5: Establecer campos obligatorios en formularios](#)'.

#### ATRIBUTO STEP HTML5

Vamos a repasar el funcionamiento del **atributo step HTML5**, este atributo permite establecer el rango de intervalos en los que se pueden mover los valores de los [input de fecha](#) y numéricos.

Podéis saber más sobre el atributo en su post '[Atributo step HTML5: Establecer intervalo de valores en inputs](#)'.

### 10.2 NUEVOS TIPOS DE INPUT EN HTML5

Dentro de la gran cantidad de novedades que nos ofrece el HTML5, una muy interesante son los nuevos valores para el atributo type del elemento input.

Estos nuevos tipos de campos hacen que los navegadores adopten distintos comportamientos que, sin dudas, nos van a hacer la vida más fácil a los desarrolladores de sitios web.

Así como actualmente tenemos el conocido password que oculta la contraseña con asteriscos o círculos (dependiendo del navegador); ahora contamos con el nuevo search que presenta una pequeña cruz para poder borrar su contenido; o también el nuevo campo numérico number que muestra dos flechas (hacia arriba y hacia abajo) para aumentar o disminuir el valor del número.

Algo muy importante a tener en cuenta es que si bien estos elementos todavía no son soportados por todos los navegadores modernos, el uso de los mismos no afectará de ningún modo en los navegadores que no los soporten, actuarán simplemente como si fuesen del tipo text.

### EJEMPLOS DE LOS NUEVOS ATRIBUTOS

A continuación los distintos valores, recuerda que solo verás los nuevos comportamientos si estás utilizando algún navegador moderno (últimas versiones de Chrome, Firefox, Safari, Opera o IE).

#### Búsqueda (search):

```
<input type="search" name="busqueda">
```

Al ingresar texto en el campo, el navegador muestra una cruz a la derecha para borrar todo lo que hemos escrito.



En Chrome, el campo de búsqueda de htmlcinco.com

#### Teléfono (tel):

```
<input type="tel" name="telefono">
```

A la hora de completar un input de tipo tel, un smartphone como el iPhone convierte su teclado a números de teléfono.



El teclado del iPhone se adapta para completar un input de tipo tel.

#### Dirección url (url):

```
<input type="url" name="url">
```

En este campo, el teclado del iPhone es qwerty pero en modo "url", ya que ofrece teclas fundamentales para escribir una dirección web como son el punto, la barra "/" o la tecla ".com".



El teclado del iPhone para completar un campo de tipo url.

### Correo electrónico (email):

```
<input type="email" name="correo">
```

Esta vez, el teclado del smartphone es qwerty pero también tenemos la tecla "@".



Teclado para escribir una dirección de correo electrónico.

### Fecha y hora (datetime):

```
<input type="datetime" name="fechahora">
```

Si estás viendo esta página con la última versión de Opera, al clicar en el campo verás un calendario muy completo que el navegador dispone de forma totalmente nativa.



El campo de tipo fecha y hora en Opera

### Fecha (date):

```
<input type="date" name="fecha">
```

Si estamos usando Opera, el calendario es el mismo que el de la imagen anterior.

### Mes (month):

```
<input type="month" name="mes">
```

El calendario aquí permite seleccionar el número de mes.

### Semana (week):

```
<input type="week" name="semana">
```

El calendario que nos muestra Opera para el campo de semana, nos permite elegir el número de semana del año.

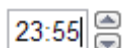


El input de tipo week en Opera.

### Hora (time):

```
<input type="time" name="hora">
```

Aquí el input está con el formato de hora, con los dos puntos “:” correspondientes y las flechas para subir o bajar el horario.



El input de tipo hora.

### Número (number):

```
<input type="number" name="num" min="0" max="50">
```

Para este input tenemos los atributos min y max para establecer el máximo y el mínimo que acepta el campo.

En un smartphone vemos el teclado numérico:



Teclado numérico del iPhone.

### Rango (range):

```
<input type="range" name="rango" min="0" max="50">
```

El input de tipo range se presenta como un control para arrastrar con el mouse (o con el dedo en un móvil con pantalla táctil). Este campo también acepta los atributos min y max.



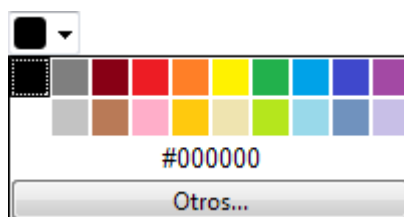


Input de tipo range.

### Color (color):

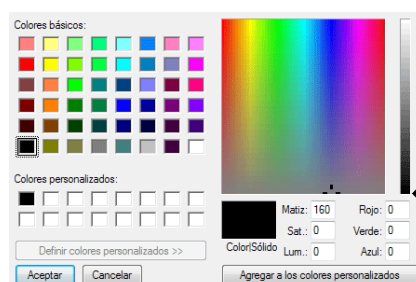
```
<input type="color" name="color">
```

Si estás viendo este campo con la última versión de Opera te vas a sorprender, porque el navegador presenta de forma nativa un selector de color... otra funcionalidad que comunmente tendríamos que hacer con javascript.



Primera ventana para seleccionar un color en Opera.

Pero algo que sorprende aún más, es que al clicar en “otros...” de la ventana anterior, Opera muestra una ventana con un selector de color mucho más avanzado como el de la siguiente captura de pantalla:



Una opción más avanzada para seleccionar color en Opera.