TEMA 2.
HTML. PÁGINAS WEB ESTÁTICAS. HTML (II)

Lenguajes de Marcas
CFGS DAW 1

Autor: Pascual Ligero

Revisado por:

Fco. Javier Valero – franciscojavier.valero@ceedcv.es

2019/2020

Versión:190920.2254

Licencia

Nomenclatura

A lo largo de este tema se utilizarán distintos símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:

 Importante

 Atención

 Interesante

ÍNDICE DE CONTENIDO

UD02. HTML (II)

La teoría de este capítulo está sacada íntegramente de http://librosweb.es/libro/xhtml/ os la pongo en este formato para que sea más cómoda de estudiar y por que quitaré cosas que sean obvias o ya no sean imprescindibles.

1. ENLACES

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

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

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

1.1 URL

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

La URL de un recurso tiene dos objetivos principales:

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

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

Si se accede a la página principal de Google, la dirección que muestra el navegador es:
`http://www.google.com`

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

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

Una URL sencilla siempre está formada por las mismas tres partes. Si por ejemplo se considera la siguiente URL:
`http://www.librosweb.es/xhtml/capitulo4.html`

Las partes que componen la URL anterior son:

- **Protocolo** (`http://`): el mecanismo que debe utilizar el navegador para acceder a ese recurso. Todas las páginas web utilizan `http://`. Las páginas web *seguras* (por ejemplo las de los bancos y las de los servicios de email) utilizan `https://` (se añade una letra `s`).
- **Servidor** (`www.librosweb.es`): simplificando mucho su explicación, se trata del ordenador en el que se encuentra guardada la página que se quiere acceder. Los navegadores son capaces de obtener la dirección de cada servidor a partir de su nombre.
- **Ruta** (`/xhtml/capitulo4.html`): *camino* que se debe seguir, una vez que se ha llegado al servidor, para localizar el recurso específico que se quiere acceder.

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

La mayoría de URL son tan sencillas como la URL mostrada anteriormente. No obstante, existen URL complejas formadas por más partes.
`http://www.alistapart.com/comments/webstandards2008?page=5#42`

Las cinco partes que forman la URL anterior son:

- Protocolo (`http://`)
- Servidor (`www.alistapart.com`)
- Ruta (`/comments/webstandards2008`)
- Consulta (`?page=5`): información adicional necesaria para que el servidor localice correctamente el recurso que se quiere acceder. Siempre comienza con el carácter `?` y contiene una sucesión de palabras separadas por = y &
- Sección (`#42`): permite que el navegador se posicione automáticamente en una sección de la página web. Siempre comienza con el caracter #

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

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

A continuación se muestra la tabla para codificar los caracteres más comunes:

Carácter original	Carácter codificado	Carácter original	Carácter codificado
/	%2F	?	%3F
:	%3A	@	%40
=	%3D	&	%26
"	%22	\	%5C

Carácter original	Carácter codificado	Carácter original	Carácter codificado
'	%60	~	%7E
(espacio en blanco)	%20	#	%23

Por otra parte, aunque desde hace tiempo ya es posible incluir en las URL caracteres de otros idiomas que no sean el inglés, aún no es completamente seguro utilizar estos caracteres en las URL. Si se utilizan letras como ñ, á, é o ç, es posible que algunos navegadores no las interpreten de forma correcta.

La solución consiste en codificar todos los caracteres que no existen en inglés. La siguiente tabla muestra la codificación de los caracteres más utilizados:

Carácter original	Carácter codificado	Carácter original	Carácter codificado
ñ	%F1	Ñ	%D1
á	%E1	Á	%C1
é	%E9	É	%C9
í	%ED	Í	%CD
ó	%F3	Ó	%D3
ú	%FA	Ú	%DA
ç	%E7	Ç	%C7

Teniendo en cuenta las dos tablas anteriores de codificación de caracteres, es fácil crear las URL correctas sin caracteres problemáticos:

```
<!-- URL problemática -->
http://www.ejemplo.com/estaciones/otoño.html

<!-- URL correcta -->
http://www.ejemplo.com/estaciones/oto%F1o.html

<!-- URL problemática -->
http://www.ejemplo.com/ruta/nombre página.html

<!-- URL correcta -->
http://www.ejemplo.com/ruta/nombre%20p%E1gina.html
```

1.2 Enlaces relativos y absolutos

Las páginas web habituales suelen contener decenas de enlaces de diferentes tipos. La siguiente imagen muestra algunos de los tipos de enlaces de la página principal del sitio web www.thinkvitamin.com:

Figura 1.1.1 Ejemplo de diferentes tipos de enlaces en la página 456BereaStreet.com

En esa página, cuando se pincha sobre algunos enlaces, el navegador abandona el sitio web para acceder a páginas que se encuentran en otros sitios. Estos enlaces se conocen como "enlaces externos". Sin embargo, la mayoría de enlaces de un sitio web apuntan a páginas del propio sitio web, por lo que se denominan "enlaces internos".

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

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

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

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

Imagina que dispones de una página publicada en:

http://www.ejemplo.com/ruta1/ruta2/pagina1.html y quieres incluir en ella un enlace a otra página que se encuentra en http://www.ejemplo.com/ruta1/ruta2/pagina2.html. Como las URL identifican de forma única a los recursos de Internet y proporcionan la información necesaria para llegar hasta ellos, el enlace debería utilizar la URL completa de la segunda página.

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

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

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

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

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

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

1. La URL no es absoluta, por lo que se debe determinar la URL absoluta a partir de la URL relativa para poder cargar el recurso enlazado.
2. A la URL relativa le falta el protocolo y el servidor, por lo que se supone que son los mismos que los de la página origen (`http://` y `www.ejemplo.com`).
3. Se añaden las partes que faltan a la URL relativa para obtener la URL absoluta: `http://` + `www.ejemplo.com` + `/ruta1/ruta2/pagina2.html` = http://www.ejemplo.com/ruta1/ruta2/pagina2.html.

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

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

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

Elemento	Valor
Página origen	`http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html`
Página enlazada	Página web llamada `pagina2.html` y que se encuentra en el mismo directorio
URL absoluta	`http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina2.html`
URL relativa	`pagina2.html`

Cuando el navegador encuentra una URL relativa que solo 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	`http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html`
Página enlazada	Página web llamada `pagina2.html` y que se encuentra en el directorio superior llamado `ruta2`
URL absoluta	`http://www.ejemplo.com/ruta1/ruta2/pagina2.html`
URL relativa	`../pagina2.html`

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

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	`http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html`
Página enlazada	Página web llamada `pagina2.html` y que se encuentra en el directorio superior llamado `ruta1`
URL absoluta	`http://www.ejemplo.com/ruta1/pagina2.html`
URL relativa	`../../pagina2.html`

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

Elemento	Valor
Página origen	`http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html`
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	`http://www.ejemplo.com/ruta4/pagina2.html`
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 solo 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, solo es necesario indicar el nombre de los directorios a los que debe entrar el navegador.

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

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	`http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html`
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 absoluta	`http://www.ejemplo.com/ruta1/ruta2/ruta3/ruta4/ruta5/ruta6/pagina2.html`
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 solo omiten el protocolo y el nombre del servidor.

Elemento	Valor
Página origen	`http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html`
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	`http://www.ejemplo.com/ruta7/pagina2.html`
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 solo 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...
...solo 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

1.3 Enlaces básicos

Los enlaces en HTML se crean mediante la etiqueta <a> (su nombre viene del inglés *"anchor"*, literalmente traducido como "ancla"). A continuación se muestra la definición simplificada de <a> y más adelante se muestra su definición completa:

Etiqueta	**<a>**
Atributos comunes	básicos, internacionalización, eventos y foco
Atributos propios	• `name = "texto"` - Permite nombrar al enlace para que se pueda acceder desde otros enlaces • `href = "url"` - Indica la URL del recurso que se quiere enlazar
Tipo de elemento	En línea
Descripción	Se emplea para enlazar todo tipo de recursos

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

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

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

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

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

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

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

Un truco muy útil con los enlaces es el uso de URL relativas para poder volver al inicio del sitio web desde cualquier página web interior:

```
<a href="/">Volver al inicio</a>
```

El enlace anterior utiliza una URL relativa con una ruta que apunta directamente a la raíz del servidor. Para obtener la URL absoluta, el navegador añade el mismo protocolo y el mismo nombre de servidor de la página en la que se encuentra el enlace. El resultado es que cuando se pincha ese enlace, siempre se vuelve al inicio del sitio web, independientemente de la página en la que se incluya el enlace.

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, consectetuer 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, consectetuer 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, consectetuer 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>`.

1.4 Otros tipos de enlaces

Los enlaces mostrados en las secciones anteriores son los más utilizados por las páginas web. Los enlaces creados con la etiqueta `<a>` permiten enlazar cualquier tipo de recurso desde cualquier página. La característica más importante de estos enlaces es que el usuario debe activar la carga de los recursos. En otras palabras, el navegador no carga ningún recurso enlazado con la etiqueta `<a>` a menos que el usuario pinche sobre el enlace.

Además de estos enlaces, las páginas HTML pueden incluir otro tipo de enlaces que cargan los recursos automáticamente. Si una página HTML utiliza archivos CSS para aplicar estilos a sus contenidos, no es lógico que los enlace con la etiqueta `<a>` y espere a que el usuario pinche sobre el enlace para así cargar los archivos CSS. De la misma forma, muchas páginas web dinámicas necesitan que el navegador cargue varios archivos JavaScript para funcionar correctamente.

HTML define las etiquetas `<script>` y `<link>` para enlazar recursos que se deben cargar automáticamente. Cuando el navegador encuentra alguna de estas dos etiquetas, **descarga los recursos enlazados y los aplica a la página web**.

La etiqueta `<script>` tiene dos modos de funcionamiento, ya que se emplea tanto para insertar un bloque de código JavaScript en la página como para enlazar un archivo JavaScript externo.

Etiqueta	**<script>**
Atributos comunes	-
Atributos propios	• `src = "url"` - Indica la dirección del archivo que contiene el código • `type = "tipo_de_contenido"` - Permite "avisar" al navegador sobre el tipo de código que se incluye (normalmente JavaScript) • `defer = "defer"` - El código no va a modificar el contenido de la página web • `charset = "tipo_de_charset"` - Describe la codificación del código enlazado
Tipo de elemento	Bloque y en línea (también puede ser una etiqueta vacía)
Descripción	Se emplea para enlazar o definir un bloque de código (normalmente JavaScript)

Aunque la etiqueta `<script>` permite enlazar código de varios lenguajes de programación, el uso habitual de `<script>` consiste en enlazar un archivo JavaScript externo:

```
<head>
  <script type="text/javascript" src="http://www.ejemplo.com/js/inicializar.js">
  </script>
</head>
```

El atributo `type` utilizado habitualmente para los archivos JavaScript es `"text/javascript"`. El atributo `src` es equivalente al atributo `href` de los enlaces creados con la etiqueta `<a>`. La URL indicada en el atributo `src` puede ser absoluta o relativa y externa o interna.

Además de enlazar un archivo JavaScript externo, la misma etiqueta `<script>` también permite incluir en la página web un bloque de código JavaScript:

```
<html>
<head>
  <script type="text/javascript">
  //<![CDATA[
      window.onload = function() {
            alert("La página se ha cargado completamente");
      }
  //]]>
  </script>
</head>
<body>
...
</body>
</html>
```

Cuando se incluye código JavaScript en la propia página XHTML, se debe insertar dentro de una sección especial llamada CDATA. Para ello, el código JavaScript se debe encerrar entre `<![CDATA[` y `]]>`.

Cuando el navegador encuentra una sección de este tipo, no procesa su contenido como si fuera XHTML y por tanto no tiene en cuenta los posibles errores de validación de XHTML.

De esta forma, se pueden construir páginas XHTML válidas y código JavaScript correcto. Los caracteres `//` al comienzo y al final de la sección CDATA son necesarios para los navegadores que no son capaces de procesar correctamente estas secciones.

La etiqueta `<script>` (tanto cuando enlaza, como cuando incluye directamente el código) puede aparecer en cualquier parte del documento HTML, aunque normalmente se incluye dentro de la cabecera de la página (`<head>...</head>`).

La segunda etiqueta de XHTML para enlazar recursos es `<link>`, que permite enlazar y relacionar la página con otros recursos externos.

Etiqueta	**<link>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	• Los siguientes con el mismo significado que para la etiqueta "a": charset, href, hreflang, type, rel y rev • `media = "tipo_de_medio"` - Indica el medio para el que debe aplicarse la relación
Tipo de elemento	Etiqueta vacía
Descripción	Se emplea para enlazar y establecer relaciones entre el documento y otros recursos

Al contrario que `<script>`, la etiqueta `<link>` solamente se puede incluir dentro de la cabecera del documento.

Se pueden añadir tantas etiquetas `<link>` como sean necesarias, pero siempre dentro de `<head>...</head>`.

El atributo `media` hace referencia al medio para el que es válida la relación con el recurso enlazado. Los medios disponibles también están estandarizados, siendo los más comunes `screen` para los contenidos mostrados en pantalla, `print` para las impresoras y `handheld` para los dispositivos móviles.

El uso habitual de la etiqueta `<link>` es el de enlazar las hojas de estilos CSS utilizadas por las páginas web:

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

En este caso, es habitual establecer los atributos `rel` y `type` para indicar el tipo de recurso enlazado y su relación con la página web. La URL del recurso enlazado se indica en el atributo `href`, que admite tanto URL absolutas como relativas.

1.5 Ejemplos de enlaces habituales

1.5.1 Enlace al inicio del sitio web

```
<a href="/">Inicio</a>
```

Al pulsar el enlace anterior desde cualquier página web, se vuelve directamente a la página de inicio, *home* o página principal del sitio web.

1.5.2 Enlace a un email

```
<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.

La forma de mostrar las direcciones de correo electrónico en las páginas web consiste en incluir la dirección en una imagen o indicarla de forma que solamente los usuarios puedan entenderlo:

```
<p>La dirección de correo es <strong>nombre (arroba) direccion.com</strong></p>
<p>La dirección de correo es <strong>nombre_arroba_direccion.com</strong></p>
<p>La dirección de correo es <strong>nombreQUITAESTO@direccion.com</strong></p>
```

```
<p>La dirección de correo es <strong>nombre(ARROBA)direccion.com</strong></p>
<p>La dirección de correo es <strong>nombre @ direccion . com</strong></p>
```

1.5.3 Enlace a un archivo FTP

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>
```

1.5.4 Enlazar varias hojas de estilos CSS

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

1.5.5 Enlazar hojas de estilos CSS para diferentes medios

```
<link rel="stylesheet" type="text/css" href="/css/comun.css" media="screen, projection" />
<link rel="stylesheet" type="text/css" href="/css/impresora.css" media="print" />
<link rel="stylesheet" type="text/css" href="/css/movil.css" media="handheld" />
```

1.5.6 Enlazar el favicon

El *favicon* o **icono** para favoritos es el pequeño icono que muestran las páginas en varias partes del navegador. Dependiendo del navegador que se utilice, este icono se muestra en la barra de direcciones, en la barra de título del navegador y/o en el menú de favoritos/marcadores.

```
<link rel="shortcut icon" href="/favicon.ico" type="image/ico" />
```

Aunque en principio la imagen debería ser de tipo `.ICO` (formato gráfico de los iconos), algunos navegadores soportan favicons en otros formatos gráficos más habituales (como por ejemplo `.PNG`).

1.5.7 Enlazar un archivo RSS

```
<link rel="alternate" type="application/rss+xml" title="Resumen de todos los artículos del blog" href="/feed.xml" />
```

1.5.8 Enlazar hojas de estilos, favicon y RSS

En una misma página se pueden incluir varias etiquetas `<link>`, por lo que es habitual que las páginas enlacen hojas de estilos, favicon y archivos RSS de forma conjunta:

```
<head>
...
<link rel="stylesheet" type="text/css" href="/css/impresora.css" media="print" />
<link rel="stylesheet" type="text/css" href="/css/movil.css" media="handheld" />
<style type="text/css" media="screen,projection">
  @import '/css/main.css';
```

```
</style>
<link rel="shortcut icon" href="/favicon.ico" type="image/ico" />
<link rel="alternate" type="application/rss+xml" title="Resumen de todos los artículos del blog" href="/feed.xml" />
...
</head>
```

1.5.9 Indicar que existe una versión de la página en otro idioma

```
<head>
<title>English tutorial</title>
<link lang="es" xml:lang="es" title="El tutorial en español" type="text/html" rel="alternate" hreflang="es" href="http://www.ejemplo.com/tutorial/espanol.html" />
</head>
```

1.5.10 Indicar que existe una versión de la página preparada para imprimir

```
<head>
<link media="print" title="El tutorial en PDF" type="application/pdf" rel="alternate" href="http://www.ejemplo.com/tutorial/documento.pdf" />
</head>
```

1.5.11 Indicar que existe una página que es índice de la página actual

```
<head>
<title>Tutorial – Capítulo 5</title>
<link rel="start" title="El índice del tutorial" type="text/html" href="http://www.ejemplo.com/tutorial/indice.html" />
</head>
```

2. LISTAS

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

El menú de navegación de un sitio web por ejemplo está formado por un grupo de palabras. Aunque cada palabra por separado tiene sentido, de forma conjunta constituyen el menú de navegación de la página, por lo que su significado conjunto es mayor que por separado.

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

2.1 Listas no ordenadas

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

Etiqueta	** **
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	-
Tipo de elemento	Bloque
Descripción	Se emplea para definir listas no ordenadas
Etiqueta	** **
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	-
Tipo de elemento	Bloque
Descripción	Se emplea para definir los elementos de las listas (ordenadas y no ordenadas)

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

```
<html>
<head><title>Ejemplo de etiqueta UL</title></head>
<body>

<h1>Menú</h1>

<ul>
  <li>Inicio</li>
  <li>Noticias</li>
  <li>Artículos</li>
  <li>Contacto</li>
</ul>

</body>
</html>
```

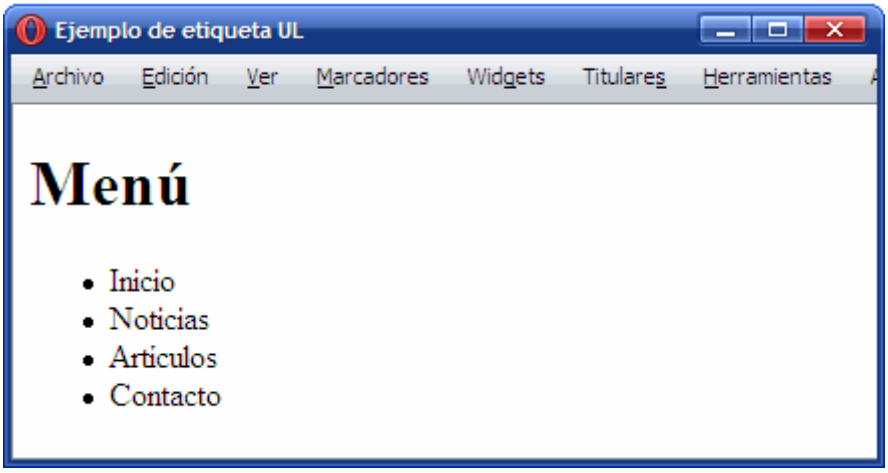

Figura 2.1 Ejemplo de uso de la etiqueta ul

El navegador por defecto muestra los elementos de la lista tabulados y con una pequeña viñeta formada por un círculo negro. Como ya se sabe, el aspecto con el que se muestran los elementos de las listas se puede modificar mediante las hojas de estilos CSS.

2.2 Listas ordenadas

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

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

Etiqueta	** **
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	-
Tipo de elemento	Bloque
Descripción	Se emplea para definir listas ordenadas

El siguiente código HTML muestra un ejemplo sencillo de lista ordenada:

```
<html>
<head><title>Ejemplo de etiqueta OL</title></head>
<body>

<h1>Instrucciones</h1>

<ol>
  <li>Enchufar correctamente</li>
  <li>Comprobar conexiones</li>
  <li>Encender el aparato</li>
</ol>

</body>
</html>
```

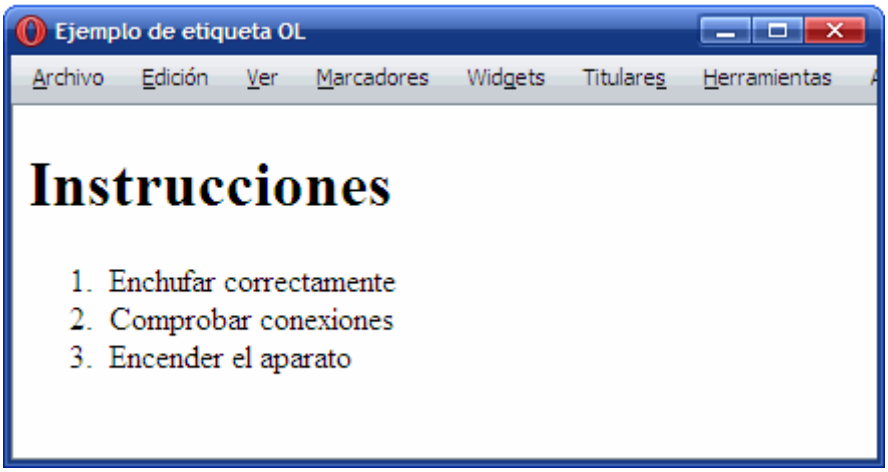

Figura 2.2 Ejemplo de uso de la etiqueta ol

El navegador muestra la lista de forma muy parecida a las listas no ordenadas, salvo que en este caso no se emplean viñetas gráficas en los elementos, sino que se numeran de forma consecutiva. El tipo de numeración empleada también se puede modificar aplicando hojas de estilos CSS a los elementos de la lista.

2.3 Listas de definición

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

Etiqueta	**<dl>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	-
Tipo de elemento	Bloque
Descripción	Se emplea para definir listas de definición

Etiqueta	**<dt>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	-
Tipo de elemento	Bloque
Descripción	Se emplea para definir los términos de los elementos de una lista de definición
Etiqueta	**<dd>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	-
Tipo de elemento	Bloque
Descripción	Se emplea para indicar las definiciones de los elementos de una lista de definición

El siguiente código HTML muestra un ejemplo sencillo de lista de definición:

```
<html>
<head><title>Ejemplo de etiqueta DL</title></head>
<body>
<h1>Metalenguajes</h1>

<dl>
  <dt>SGML</dt>
  <dd>Metalenguaje para la definición de otros lenguajes de marcado</dd>

  <dt>XML</dt>
  <dd>Lenguaje basado en SGML y que se emplea para describir datos</dd>

  <dt>RSS</dt>
  <dt>GML</dt>
  <dt>XHTML</dt>
  <dt>SVG</dt>
  <dt>XUL</dt>
  <dd>Lenguajes derivados de XML para determinadas aplicaciones</dd>
</dl>

</body>
</html>
```

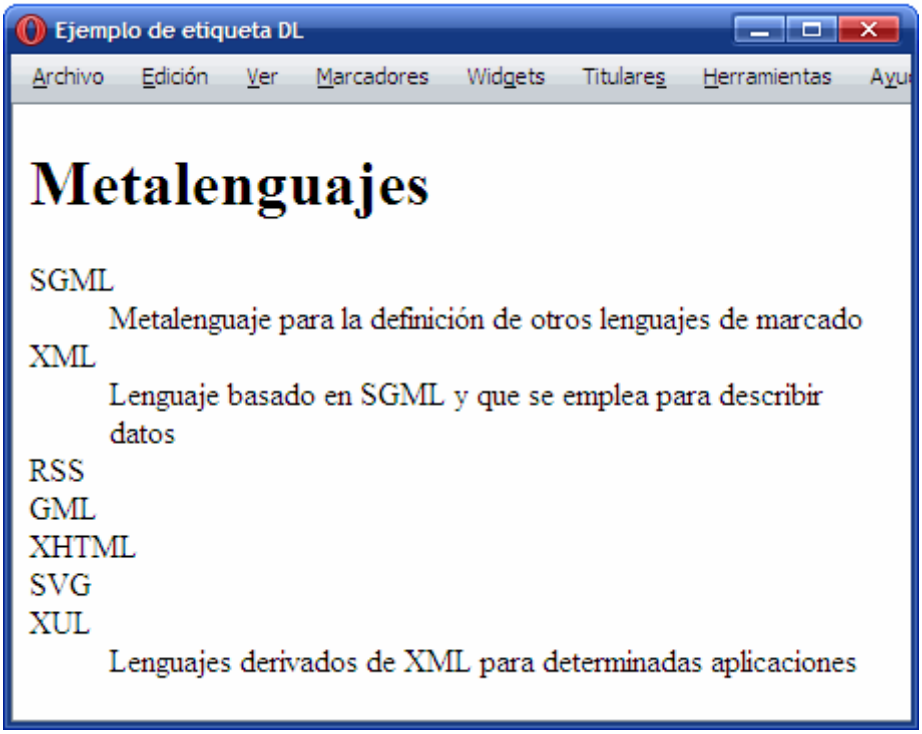

Figura 2.3 Ejemplo de uso de la etiqueta dl

Los navegadores formatean las listas de definición de forma similar a las otras listas, tabulando la definición y alineando a la izquierda los términos. Aunque no es habitual, cada término puede tener asociada más de una definición y cada definición puede tener asociada varios términos.

3. IMÁGENES Y OBJETOS

3.1 Imágenes

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

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

A continuación se muestra la definición de la etiqueta `<img>`, utilizada para incluir las imágenes en las páginas HTML:

Etiqueta	**<img>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	• `src = "url"` - Indica la URL de la imagen que se muestra • `alt = "texto"` - Descripción corta de la imagen • `longdesc = "url"` - Indica una URL en la que puede encontrarse una descripción más detallada de la imagen • `name = "texto"` - Nombre del elemento imagen • `height = "unidad_de_medida"` - Indica la altura con la que se debe mostrar la imagen (no es obligatorio que coincida con la altura original de la imagen) • `width = "unidad_de_medida"` - Indica la anchura con la que se debe mostrar la imagen (no es obligatorio que coincida con la anchura original de la imagen)
Tipo de elemento	En línea y etiqueta vacía
Descripción	Se emplea para incluir imágenes en los documentos

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

Las URL indicadas pueden ser absolutas o relativas. El atributo `alt` permite describir el contenido

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

Ejemplo sencillo para incluir una imagen:

```
<img src="logotipo.gif" alt="Logotipo de Mi Sitio" />
```

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

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

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

El atributo `longdesc` no se utiliza de forma habitual, pero permite indicar la URL en la que se puede encontrar más información sobre la imagen. Como el atributo `alt` solo permite incluir descripciones de hasta 1024 caracteres, el atributo `longdesc` se emplea con las imágenes complejas que necesitan mucha información para ser descritas:

```
<img src="/imagenes/proyecto1.png" alt="Imagen del Proyecto 1"
longdesc="/portfolio/proyecto1.html" />

<img src="/imagenes/proyecto2.jpg" alt="Imagen del Proyecto 2"
longdesc="/mas_informacion.html" />
```

En el ejemplo anterior, las dos imágenes se encuentran en el mismo directorio del servidor (`/imagenes/`). Se trata de una estrategia habitual en la mayoría de sitios web: guardar todas las imágenes de contenido en un directorio especial independiente del resto de contenidos HTML. Además, el directorio siempre suele llamarse de la misma manera: "`imagenes`" o "`images`" en inglés.

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 XHTML.

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.

```
<img src="/imagenes/foto1.jpg" alt="Fotografía de un paisaje" width="200"
```

```
height="350" />

<img src="/imagenes/foto2.jpg" alt="Fotografía de un atardecer en la playa"
width="330" height="220" />
```

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>
   <img src="/imagenes/foto1.jpg" alt="Fotografía de un paisaje" width="30%"
height="350" />
</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 solo se establece la anchura de la imagen, el navegador calcula la altura que hace que la imagen se siga viendo con las mismas proporciones.

4. MAPAS DE IMAGEN

Aunque el uso de los mapas de imagen se ha reducido drásticamente en los últimos años, aún se utilizan en algunos sitios especializados. Muchas agencias de viaje y sitios relacionados utilizan mapas geográficos para seleccionar el destino del viaje. La mayoría de mapas se realiza hoy en día mediante Flash, aunque algunos sitios siguen recurriendo a los mapas de imagen.

Un mapa de imagen permite definir diferentes zonas *"pinchables"* dentro de una imagen. El usuario puede pinchar sobre cada una de las zonas definidas y cada una de ellas puede apuntar a una URL diferente. Siguiendo el ejemplo anterior, una sola imagen que muestre un mapa de todos los continentes puede definir una zona diferente para cada continente. De esta forma, el usuario puede pinchar sobre la zona correspondiente a cada continente para que el navegador muestre la página que contiene los viajes disponibles a ese destino.

Las zonas o regiones que se pueden definir en una imagen se crean mediante rectángulos, círculos y polígonos. Para crear un mapa de imagen, en primer lugar se inserta la imagen original mediante la etiqueta `<img>`. A continuación, se utiliza la etiqueta `<map>` para definir las zonas o regiones de la imagen. Cada zona se define mediante la etiqueta `<area>`.

Etiqueta	**<map>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	• `name = "texto"` - Nombre que identifica de forma única al mapa definido (es obligatorio indicar un nombre único)
Tipo de elemento	Bloque y en línea
Descripción	Se emplea para definir mapas de imagen
Etiqueta	**<area>**
Atributos comunes	básicos, internacionalización, eventos y foco
Atributos propios	• `href = "url"` - URL a la que se accede al pinchar sobre el área • `nohref = "nohref"` - Se emplea para las áreas que no son seleccionables • `shape = "default | rect | circle | poly"` - Indica el tipo de área que se define (toda la imagen, rectangular, circular o poligonal) • `coords = "lista de números"` - Se trata de una lista de números separados por comas que representan las coordenadas del área. Rectangular = X1,Y1,X2,Y2 (coordenadas X e Y del vértice superior izquierdo y coordenadas X e Y del vértice inferior derecho). Circular = X1,Y1,R (coordenadas X e Y del centro y radio del círculo). Poligonal =

Etiqueta	**<area>**
	X1,Y1,X2,Y2,...,XnYn (coordenadas de los vértices del polígono. Si las últimas coordenadas no son iguales que las primeras, se cierra automáticamente el polígono uniendo ambos vértices)
Tipo de elemento	Etiqueta vacía
Descripción	Se emplea para definir las distintas áreas que forman un mapa de imagen

Si una imagen utiliza un mapa de imagen, debe indicarlo mediante el atributo `usemap`. El valor del atributo debe ser el nombre del mapa de imagen definido en otra parte del mismo documento HTML:

```
<img src="mapa_mundo.gif" usemap="#continentes" />
...
<map name="continentes">
  ...
</map>
```

Las áreas se definen mediante el atributo `shape` que indica el tipo de área y `coords` que es una lista de coordenadas cuyo significado depende del tipo de área definido. El enlace de cada área se define mediante el atributo `href`, con la misma sintaxis y significado que para los enlaces normales.

El siguiente ejemplo muestra una imagen sencilla que contiene cuatro figuras geométricas:

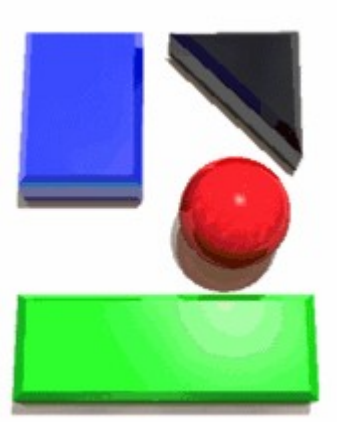

Figura 4.1 Ejemplo de imagen que incluye un mapa de imagen

Utilizando un círculo, dos rectángulos y un polígono se pueden definir fácilmente cuatro zonas *pinchables* en la imagen mediante el siguiente código HTML:

```
<img src="imagen.gif" usemap="#mapa_zonas" />

<map name="mapa_zonas">

      <area shape="rect" coords="20,25,84,113" href="rectangulo.html" />

      <area shape="polygon" coords="90,25,162,26,163,96,89,25,90,24"
      href="triangulo.html" />

      <area shape="circle" coords="130,114,29" href="circulo.html" />

      <area shape="rect" coords="19,156,170,211"
      href="mailto:rectangulo@direccion.com" />

      <area shape="default" nohref="nohref" />

</map>
```

5. OBJETOS

Además de las imágenes, HTML permite incluir en las páginas web otros elementos mucho más complejos, como *applets* de Java y vídeos en formato QuickTime o Flash. La mayoría de este tipo de contenidos no los interpreta el navegador directamente, sino que hace uso de pequeños programas llamados *plugins* y que se encargan de tratar con este tipo de elementos complejos.

La etiqueta `<object>` es la que permite *"embeber"* o incluir en las páginas HTML cualquier tipo de contenido complejo:

Etiqueta	**<object>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	• `data = "url"` - Indica la URL de los datos que utiliza el objeto • `classid, codebase, codetype` - Información específica que depende del tipo de objeto • `type` - Indica el tipo de contenido de los datos • `height = "unidad_de_medida"` - Indica la altura con la que se debe mostrar el objeto • `width = "unidad_de_medida"` - Indica la anchura con la que se debe mostrar el objeto
Tipo de elemento	Bloque y en línea
Descripción	Se emplea para embeber objetos en los documentos

El atributo `data` se emplea para indicar la URL del recurso que se va a incluir. El atributo `type` indica el tipo de contenido de los datos del objeto.

Los posibles valores de `type` están estandarizados y coinciden con los del atributo `type` de la etiqueta `<a>` que se explicó anteriormente.

El propio estándar de HTML incluye ejemplos de uso de esta etiqueta. Incluir un vídeo en formato MPEG:

```
<object data="PlanetaTierra.mpeg" type="application/mpeg" />
```

También se pueden incluir varias versiones alternativas de un mismo contenido. Así, si el navegador no es capaz de interpretar el formato por defecto, puede optar por cualquiera de los otros formatos alternativos:

```
<object title="La Tierra vista desde el espacio"
classid="http://www.observer.mars/TheEarth.py">

  <!-- Formato alternativo en forma de vídeo -->
  <object data="PlanetaTierra.mpeg" type="application/mpeg">
  <!-- Otro formato alternativo mediante una imagen GIF -->
      <object data="PlanetaTierra.gif" type="image/gif">
      <!-- Si el navegador no soporta ningún formato, se muestra el siguiente
      texto -->
      La <strong>Tierra</strong> vista desde el espacio.
    </object>
  </object>
</object>
```

A los objetos también se les puede pasar información adicional en forma de parámetros mediante la etiqueta <param>:

Etiqueta	**<param>**
Atributos comunes	id
Atributos propios	• `name = "texto"` - Indica el nombre del parámetro • `value = "texto"` - Indica el valor del parámetro
Tipo de elemento	Etiqueta vacía
Descripción	Se emplea para indicar el valor de los parámetros del objeto

Las etiquetas <param> siempre se incluyen en el interior de las etiquetas <object>:

```
<object data="..." type="...">
  <param name="parametro1" value="40" />
  <param name="parametro2" value="20" />
  <param name="parametro3" value="texto de prueba" />
</object>
```

Uno de los principales inconvenientes de <object> es la forma de incluir vídeos en formato Flash en las páginas HTML. Si se utiliza el siguiente código:

```
<object data="nombre_video.swf" type="application/x-shockwave-flash"></object>
```

El elemento anterior es correcto desde el punto de vista técnico, pero provoca que algunos navegadores como Internet Explorer no visualicen el vídeo hasta que se ha descargado completamente. Si se trata de un vídeo largo, esta solución no es válida para el usuario.

Por este motivo, se utiliza una solución alternativa para incluir vídeos Flash en las páginas HTML: el uso de la etiqueta <embed>. Aunque esta solución funciona correctamente, no se trata de una solución válida desde el punto de vista del estándar de XHTML, por lo que las páginas que incluyan esta solución no pasarán correctamente el proceso de validación.

Etiqueta	**<embed>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	• `src = "url"` - Indica la URL del archivo u objeto que se incluye en la página • `type = "tipo_de_contenido"` - Indica el tipo de contenido del objeto (flash, quicktime, java, etc.) • `height = "unidad_de_medida"` - Indica la altura con la que se debe mostrar el objeto • `width = "unidad_de_medida"` - Indica la anchura con la que se debe mostrar el objeto
Tipo de elemento	Bloque
Descripción	Se emplea para embeber objetos en los documentos

Este es el motivo por el que los sitios web más populares de vídeos en formato Flash proporcionan un código similar al siguiente para incluir sus vídeos en las páginas HTML:

```
<object width="425" height="350">

  <param name="movie" value="http://www.youtube.com/v/MsH0rBWCYjs"></param>
  <param name="wmode" value="transparent"></param>
  <embed src="http://www.youtube.com/v/MsH0rBWCYjs" type="application/x-
      shockwave-flash" wmode="transparent" width="425" height="350">
  </embed>
</object>
```

Una vez más, se debe tener en cuenta que la solución anterior de utilizar la etiqueta <embed> es correcta desde el punto de vista del usuario (no tiene que esperar a que el vídeo se descargue completamente para poder verlo) pero no es una solución técnicamente válida, ya que la etiqueta <embed> no es parte del estándar XHTML.

6. TABLAS

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

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

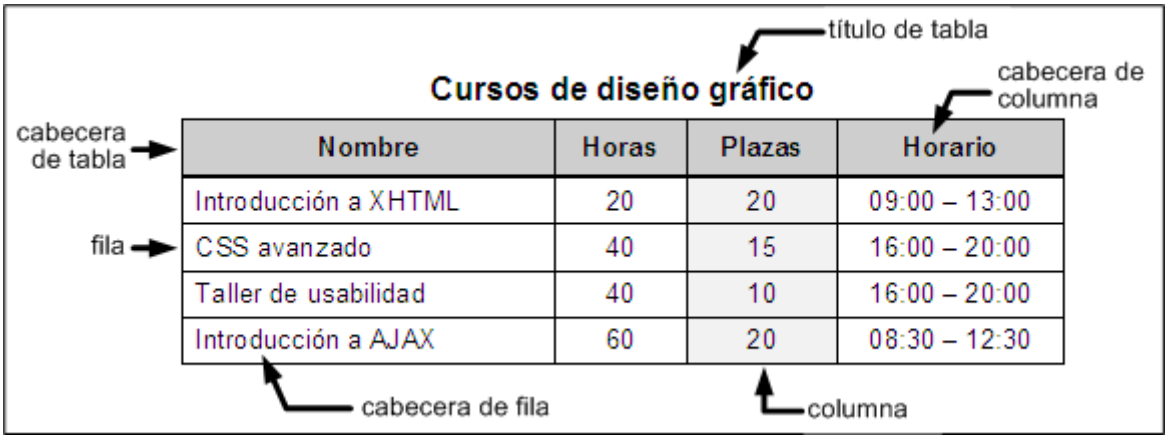

Figura 3.2.6.1 Partes que componen una tabla compleja

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

A pesar de que las tablas HTML son fáciles de comprender y utilizar, son uno de los elementos más polémicos de HTML.

El problema de las tablas es que no siempre se utilizan adecuadamente. Aunque parezca obvio, las tablas se deben utilizar para mostrar información tabular.

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

Aunque algunos malos diseñadores siguen utilizando hoy en día las tablas para definir la estructura completa de las páginas web, se trata de una técnica obsoleta y nada recomendable. El motivo es que se complica en exceso el código HTML y su mantenimiento es muy complejo. La solución correcta para definir la estructura de las páginas consiste en la utilización de hojas de estilos CSS.

6.1 Tablas básicas

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

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

```
<html>
<head><title>Ejemplo de tabla sencilla</title></head>
<body>

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

```
<table>
<tr>
  <td><strong>Curso</strong></td>
  <td><strong>Horas</strong></td>
  <td><strong>Horario</strong></td>
</tr>

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

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

<tr>
  <td>Dreamweaver</td>
  <td>60</td>
  <td>16:00 - 20:00</td>
</tr>
</table>

</body>
</html>
```

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

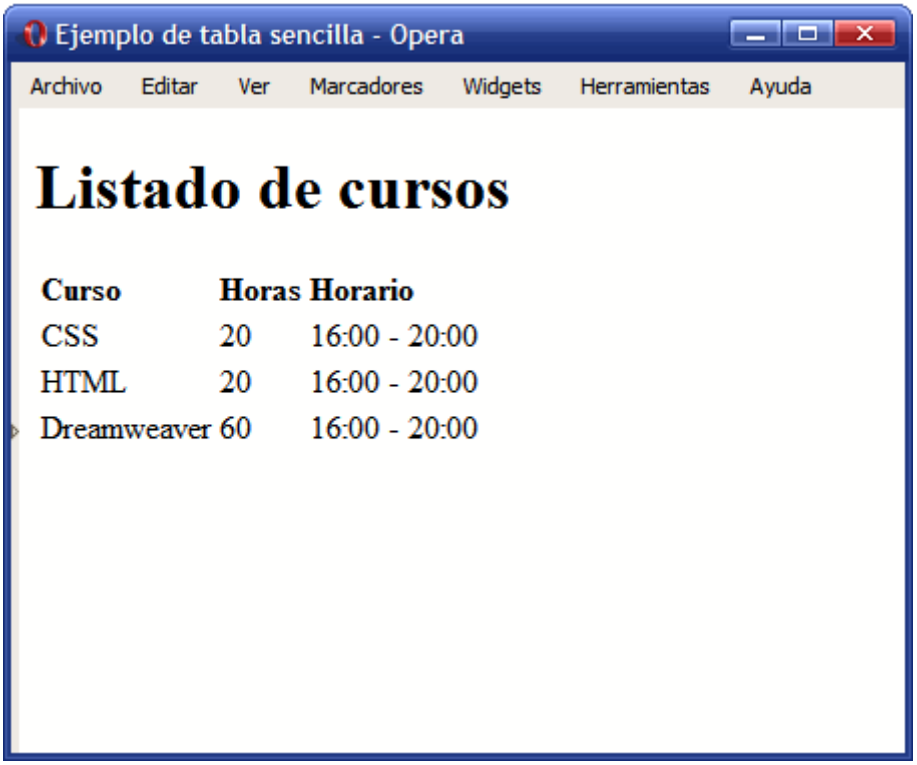

Figura 6.1 Ejemplo de tabla sencilla creada con las etiquetas table, tr y td

La etiqueta `<table>` encierra todas las filas y columnas de la tabla. Las etiquetas `<tr>` (del inglés *"table row"*) definen cada fila de la tabla y encierran todas las columnas. Por último, la etiqueta `<td>` (del inglés *"table data cell"*) define cada una de las columnas de las filas, aunque realmente HTML no define columnas sino *celdas de datos*.

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

Etiqueta	**<table>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	• `summary = "texto"` - Permite describir el contenido de la tabla (lo utilizan los buscadores y las personas discapacitadas)
Tipo de elemento	Bloque
Descripción	Se emplea para definir tablas de datos
Etiqueta	**<tr>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	-
Tipo de elemento	Bloque
Descripción	Se emplea para definir cada fila de las tablas de datos
Etiqueta	**<td>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	• `abbr = "texto"` - Permite describir el contenido de la celda (empleado sobre todo en los navegadores utilizados por personas discapacitadas) • `headers = "lista_de_id"` - Indica las celdas que actúan como cabeceras para esta celda (los títulos de las columnas y filas). Se indica como una lista de valores del atributo "id" de celdas • `scope = "col, row, colgroup, rowgroup"` - Indica las celdas para las que esta celda será su cabecera. Ej: scope="col" indica que esta celda es la cabecera de todas las demás celdas que están en la misma columna • `colspan = "numero"` - Número de columnas que ocupa esta celda • `rowspan = "numero"` - Número de filas que ocupa esta celda
Tipo de elemento	Bloque

Etiqueta	**<td>**
Descripción	Se emplea para definir cada una de las celdas que forman las filas de una tabla, es decir, las columnas de la tabla

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

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

Etiqueta	**<th>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	• `abbr = "texto"` - Permite describir el contenido de la celda (empleado sobre todo en los navegadores utilizados por personas discapacitadas) • `headers = "lista_de_id"` - Indica las celdas que actúan como cabeceras para esta celda (los títulos de las columnas y filas). Se indica como una lista de ID de celdas • `scope = "col, row, colgroup, rowgroup"` - Indica las celdas para las que esta celda será su cabecera. Ej: scope="col" indica que esta celda es la cabecera de todas las demás celdas que están en la misma columna • `colspan = "numero"` - Número de columnas que ocupa esta celda • `rowspan = "numero"` - Número de filas que ocupa esta celda
Tipo de elemento	Bloque
Descripción	Se emplea para definir las celdas que son cabecera de una fila o de una columna de la tabla

Los atributos de la etiqueta `<th>` son idénticos que los atributos definidos para la etiqueta `<td>`. En este caso, el atributo más utilizado es `scope`, que permite indicar si la celda es cabecera de la fila o de la columna (`<th scope="row">` y `<th scope="col">` respectivamente).

Por otra parte, HTML define la etiqueta `<caption>` para establecer la leyenda o título de una tabla. La etiqueta debe colocarse inmediatamente después de la etiqueta `<table>` y cada tabla solo puede incluir una etiqueta `<caption>`.

Etiqueta	**<caption>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	-
Tipo de elemento	En línea

Etiqueta	**<caption>**
Descripción	Se emplea para definir la leyenda o título de una tabla

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

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

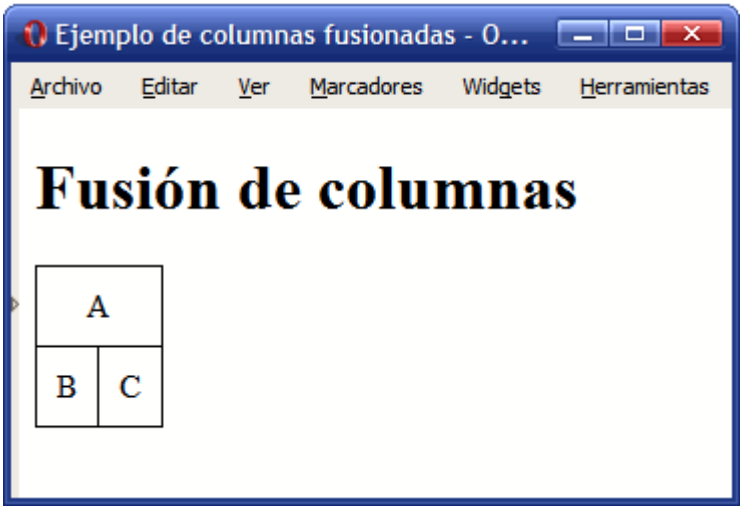

Figura 3.2.6.5 Ejemplo sencillo de fusión de columnas

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

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

<tr>
  <td>B</td>
  <td>C</td>
</tr>
</table>
```

La primera fila de la tabla está formada solo por una columna, mientras que la segunda fila está formada por dos columnas. En principio, podría pensarse en utilizar el siguiente código HTML para definir la tabla:

```
<table>
<tr>
  <td>A</td>
</tr>

<tr>
  <td>B</td>
  <td>C</td>
</tr>
</table>
```

Sin embargo, si se utiliza el código anterior, el navegador visualiza de forma incorrecta la tabla, ya que las tablas en HTML deben disponer de una estructura regular. En otras palabras, todas las filas de una tabla HTML deben tener el mismo número de columnas. Por lo tanto, si se quieren mostrar menos columnas en una fila, se fusionan mediante el atributo `colspan`, que indica el número de columnas simples que va a ocupar una determinada celda.

En el ejemplo anterior, la celda de la primera fila debe ocupar el espacio de dos columnas simples, por lo que el código HTML debe ser `<td colspan="2">A</td>`.

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

Figura 6.2 Ejemplo sencillo de fusión de filas

El código HTML que se debe utilizar para obtener la tabla de la imagen anterior es:

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

<tr>
  <td>C</td>
</tr>
</table>
```

De forma análoga a la fusión de columnas del ejemplo anterior, la fusión de filas debe indicarse de forma especial. Como las tablas HTML tienen que ser regulares, todas las columnas deben tener el mismo número de filas. Así, si en el ejemplo anterior se utilizara el siguiente código:

```
<table>
<tr>
  <td>A</td>
  <td>B</td>
</tr>

<tr>
  <td>C</td>
</tr>
</table>
```

La tabla anterior no se visualizaría correctamente. Como la segunda columna de la tabla ocupa el espacio de las dos filas, el código HTML debe indicar claramente que esa celda va a ocupar dos filas, de manera que todas las columnas de la tabla cuenten con el mismo número de filas.

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

Ejemplo de fusión de múltiples columnas:

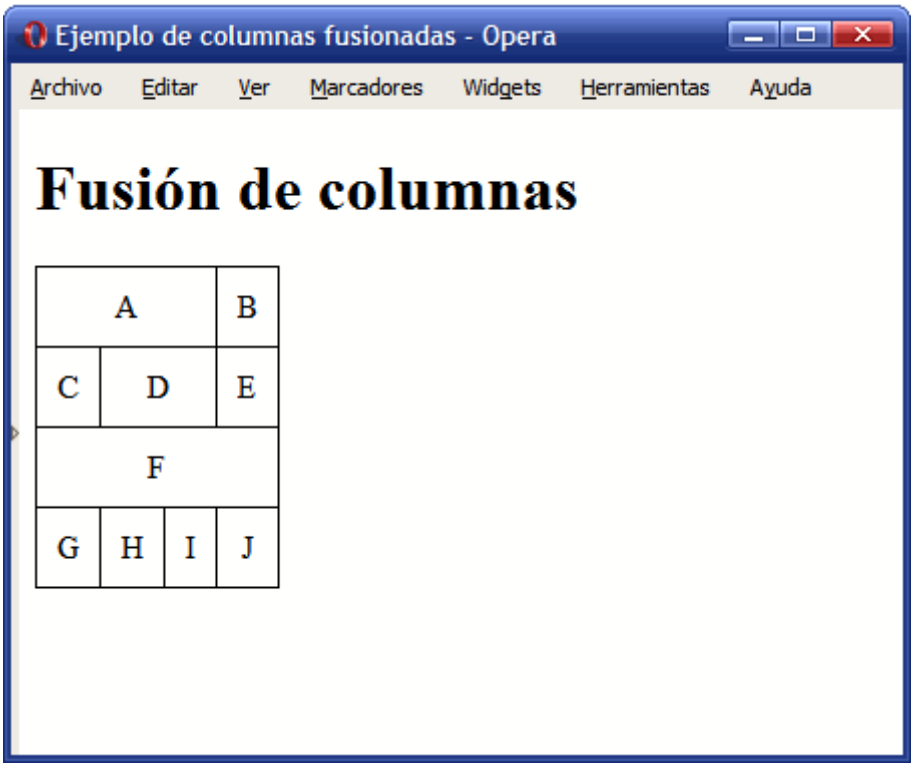

Figura 6.3 Ejemplo complejo de fusión de columnas

El código HTML necesario para fusionar las columnas de la tabla anterior se muestra a continuación:

```
<html>
<head><title>Ejemplo de columnas fusionadas</title></head>
<body>

<h1>Fusión de columnas</h1>

<table>
<tr>
  <td colspan="3">A</td>
  <td>B</td>
</tr>

<tr>
  <td>C</td>
  <td colspan="2">D</td>
  <td>E</td>
</tr>

<tr>
```

```
  <td colspan="4">F</td>
</tr>

<tr>
  <td>G</td>
  <td>H</td>
  <td>I</td>
  <td>J</td>
</tr>
</table>

</body>
</html>
```

Ejemplo de fusión de múltiples filas:

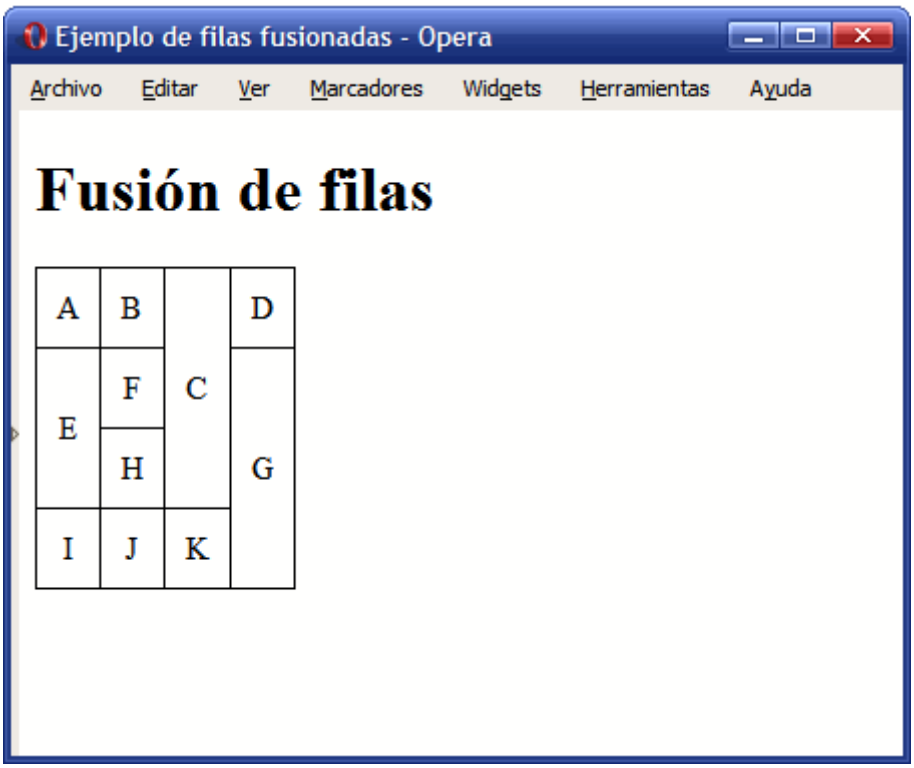

Figura 6.4 Ejemplo complejo de fusión de filas

El código HTML necesario para fusionar las filas de la tabla anterior se muestra a continuación:

```
<html>
<head><title>Ejemplo de filas fusionadas</title></head>
<body>

<h1>Fusión de filas</h1>

<table>
  <tr>
    <td>A</td>
    <td>B</td>
    <td rowspan="3">C</td>
    <td>D</td>
  </tr>
```

```
  <tr>
    <td rowspan="2">E</td>
    <td>F</td>
    <td rowspan="3">G</td>
  </tr>
  <tr>
    <td>H</td>
  </tr>
  <tr>
    <td>I</td>
    <td>J</td>
    <td>K</td>
  </tr>
</table>

</body>
</html>
```

6.2 Tablas avanzadas

Algunas tablas complejas están formadas por más elementos que filas y celdas de datos. Así, es común que las tablas más avanzadas dispongan de una sección de cabecera, una sección de pie y varias secciones de datos. Además, también es posible agrupar varias columnas de forma lógica para poder aplicar estilos similares a un determinado grupo de columnas.

Un ejemplo clásico de tablas avanzadas es el de las tablas utilizadas en contabilidad, como por ejemplo la tabla que muestra el balance de una empresa:

CONSOLIDATED BALANCE SHEET	2008			
	March	June	September	December
Non-current assets	**87.249**	**87.126**	**90.426**	**91.269**
Intangible assets	21.810	21.145	20.986	20.758
Goodwill	17.914	19.660	21.828	21.739
Property, plant and equipment and Investment property	33.245	32.332	33.428	33.888
Long-term financial assets and other non-current assets	5.723	5.687	5.981	6.183
Deferred tax assets	8.557	8.303	8.202	8.702
Current assets	**18.042**	**17.979**	**19.128**	**17.713**
Inventories	1.154	1.134	1.052	1.012
Trade and other receivables	9.244	9.495	9.709	9.666
Current tax receivable	1.288	1.565	1.468	1.555
Short-term financial investments	1.877	1.803	1.788	1.679
Cash and cash equivalents	4.468	3.557	5.101	3.792
Non-current assets classified as held for sale	11	425	9	9
Total Assets = Total Equity and Liabilities	**105.291**	**105.106**	**109.554**	**108.982**
Equity	**15.714**	**15.072**	**19.185**	**20.001**
Equity attributable to equity holders of the parent	11.932	12.085	16.397	17.178
Minority interest	3.782	2.987	2.788	2.823
Non-current liabilities	**54.053**	**66.406**	**63.908**	**62.644,0**
Long-term financial debt	41.665	54.263	51.647	50.675
Deferred tax liabilities	4.868	4.617	4.727	4.700
Long-term provisions	6.466	6.507	6.545	6.287
Other long-term liabilities	1.054	1.020	988	982
Current liabilities	**35.523**	**23.628**	**26.462**	**26.337,0**
Short-term financial debt	19.507	7.466	8.975	8.382
Trade and other payables	8.792	8.259	8.782	8.533
Current tax payable	2.007	2.324	2.529	2.841
Short-term provisions and other liabilities	5.218	5.212	6.176	6.580
Liabilities associated with non-current assets classified as held for sale	0	367	0	0
Financial Data				
Net Financial Debt (1)	53.510	54.922	52.239	52.145

Figura 6.5 Ejemplo de tabla compleja correspondiente al balance de una empresa

Las partes que componen las tablas complejas se definen mediante las etiquetas `<thead>`, `<tbody>` y `<tfoot>`. La cabecera de la tabla se define con la etiqueta `<thead>`, el pie de la tabla se define mediante `<tfoot>` y cada sección de datos se define con una etiqueta `<tbody>`.

Etiquetas	**<thead><tbody><tfoot>**
Atributos comunes	básicos, internacionalización y eventos
Atributos propios	-
Tipo de elemento	Bloque
Descripción	Se emplean para agrupar varias filas en una cabecera (thead) un pie (tfoot) o una sección (tbody) de una tabla

Cada tabla puede contener solamente una cabecera y un pie, pero puede incluir un número ilimitado de secciones. Si se define una cabecera y/o un pie, las etiquetas `<thead>` y/o `<tfoot>` deben colocarse inmediatamente antes que cualquier etiqueta `<tbody>`.

La siguiente imagen muestra una tabla avanzada con cabecera, pie y una sección de datos:

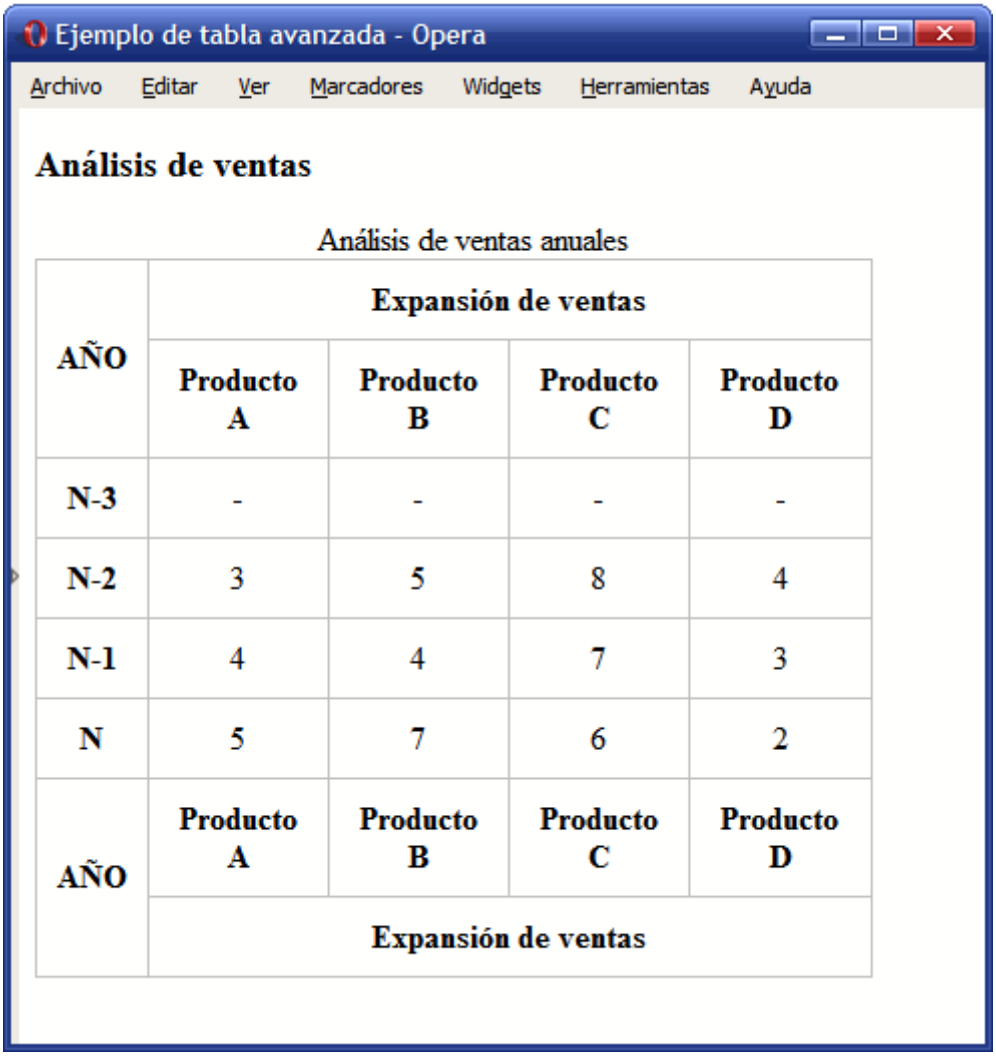

Figura 6.6 Ejemplo de tabla avanzada con cabecera, pie y secciones

El código HTML necesario para crear la tabla de la imagen anterior hace uso de las etiquetas `<thead>`, `<tbody>` y `<tfoot>`:

```
<html>
<head><title>Ejemplo de tabla avanzada</title></head>
<body>

<h3>Análisis de ventas</h3>

<table summary="Análisis de ventas anuales">
  <caption>Análisis de ventas anuales</caption>
  <thead>
    <tr>
      <th rowspan="2" scope="col">AÑO</th>
      <th colspan="4" scope="col">Expansión de ventas</th>
    </tr>
    <tr>
      <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>

  <tfoot>
    <tr>
      <th rowspan="2" 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>
    <tr>
      <th colspan="4" scope="col">Expansión de ventas</th>
    </tr>
  </tfoot>

  <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>

</body>

</html>
```

Aunque al principio resulta extraño, el elemento `<tfoot>` siempre se escribe antes que cualquier elemento `<tbody>` en el código HTML. De hecho, si la etiqueta `<tfoot>` aparece después de un elemento `<tbody>`, la página no se considera válida.

La etiqueta `<tbody>` permite realizar agrupaciones de filas, pero en ocasiones se necesitan agrupar columnas. Aunque su uso no es muy común, HTML define dos etiquetas similares para agrupar columnas: `<col>` y `<colgroup>`.

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 solo asigna atributos comunes a varias columnas.

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

Figura 6.7 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>`:

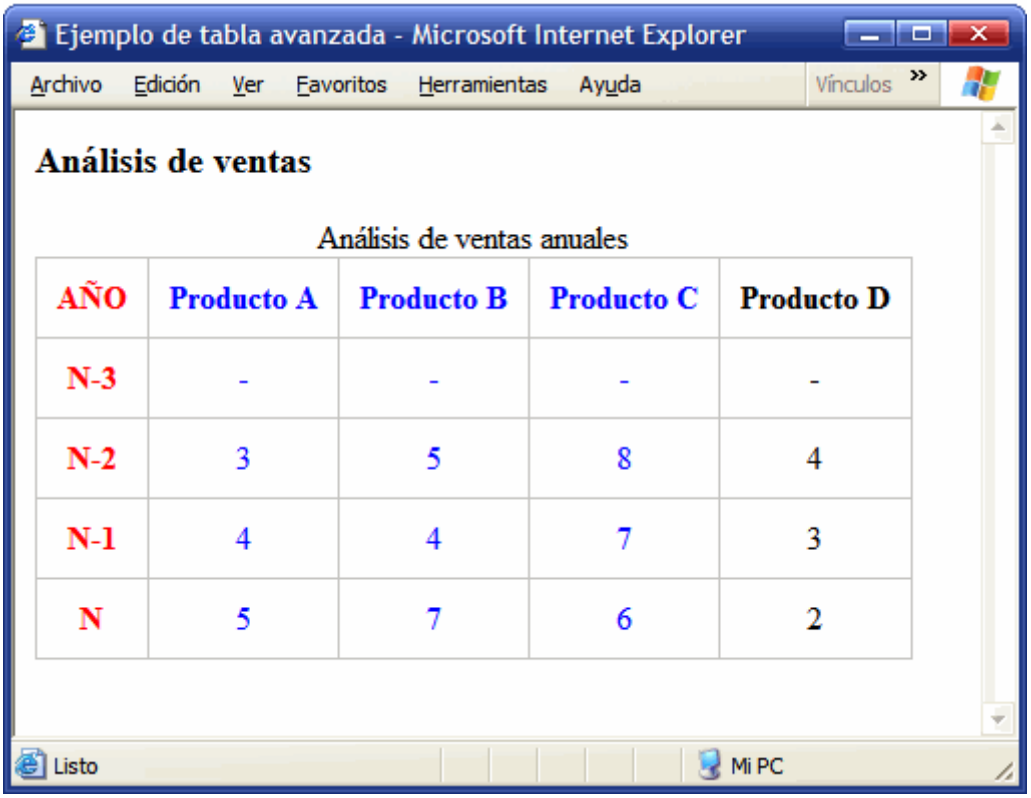

Figura 6.8 Ejemplo de tabla avanzada que usa la etiqueta colgroup

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.

7. BIBLIOGRAFÍA

http://librosweb.es/libro/xhtml/