



Arkaitz Garro

---

XHTML

Esta página se ha dejado vacía a propósito

## XHTML

**Publication date:** 14/06/2013

This book was published with *easybook v5.0-DEV*, a free and open-source book publishing application developed by Javier Eguiluz (<http://javiereguiluz.com>) using several Symfony components (<http://components.symfony.com>) .

Esta página se ha dejado vacía a propósito

Esta obra se publica bajo la licencia *Creative Commons Reconocimiento - No Comercial - Compartir Igual 3.0*, cuyos detalles puedes consultar en <http://creativecommons.org/licenses/by-nc-sa/3.0/es/>.

Esta obra está basada en el trabajo previo de Javier Eguiluz, Introducción a XHTML, publicada en la siguiente dirección: <http://www.librosweb.es/xhtml/>. Puedes copiar, distribuir y comunicar públicamente la obra, incluso transformándola, siempre que cumplas todas las condiciones siguientes:

- Reconocimiento: debes reconocer siempre la autoría de la obra original, indicando tanto el nombre del autor (Arkaitz Garro) como el nombre del sitio donde se publicó originalmente ([www.arkaitzgarro.com](http://www.arkaitzgarro.com)). Este reconocimiento no debe hacerse de una manera que sugiera que el autor o el sitio apoyan el uso que haces de su obra.
- No comercial: no puedes utilizar esta obra con fines comerciales de ningún tipo. Entre otros, no puedes vender esta obra bajo ningún concepto y tampoco puedes publicar estos contenidos en sitios web que incluyan publicidad de cualquier tipo.
- Compartir igual: si alteras o transformas esta obra o si realizas una obra derivada, debes compartir tu trabajo obligatoriamente bajo esta misma licencia.

Esta página se ha dejado vacía a propósito

|   |           |
|---|-----------|
| <b>Capítulo 1 ¿Qué es HTTP? .....</b>                   | <b>9</b>  |
| 1.1 HTTP es simple.....                                 | 9         |
| 1.2 Paso 1: el cliente manda una Petición.....          | 10        |
| 1.3 Paso 2: el servidor devuelve una Respuesta .....    | 11        |
| <b>Capítulo 2 Introducción.....</b>                     | <b>13</b> |
| 2.1 ¿Qué es HTML? .....                                 | 13        |
| 2.2 Estructura y primer documento .....                 | 15        |
| <b>Capítulo 3 Etiquetas, atributos y elementos.....</b> | <b>17</b> |
| 3.1 Tipos de atributos.....                             | 18        |
| 3.2 Elementos .....                                     | 21        |
| <b>Capítulo 4 Texto.....</b>                            | <b>27</b> |
| 4.1 Cómo estructurar un texto .....                     | 29        |
| 4.2 Marcado básico de un texto.....                     | 31        |
| 4.3 Marcado avanzado de un texto .....                  | 36        |
| 4.4 Etiqueta <span> .....                               | 39        |
| 4.5 Espacios en blanco y nuevas líneas.....             | 39        |
| 4.6 Codificación de caracteres .....                    | 44        |
| <b>Capítulo 5 Enlaces.....</b>                          | <b>47</b> |
| 5.1 URL.....  | 47        |
| 5.2 Enlaces básicos .....                               | 51        |
| 5.3 Enlaces avanzados .....                             | 52        |
| 5.4 Otros tipos de enlaces .....                        | 54        |
| 5.5 Ejemplos prácticos de enlaces .....                 | 56        |
| <b>Capítulo 6 Listas .....</b>                          | <b>59</b> |
| 6.1 Listas no ordenadas .....                           | 59        |
| 6.2 Listas ordenadas.....                               | 61        |
| 6.3 Listas de definiciones .....                        | 63        |
| 6.4 Listas anidadas.....                                | 65        |
| <b>Capítulo 7 Imágenes y objetos.....</b>               | <b>67</b> |
| 7.1 Imágenes.....                                       | 67        |

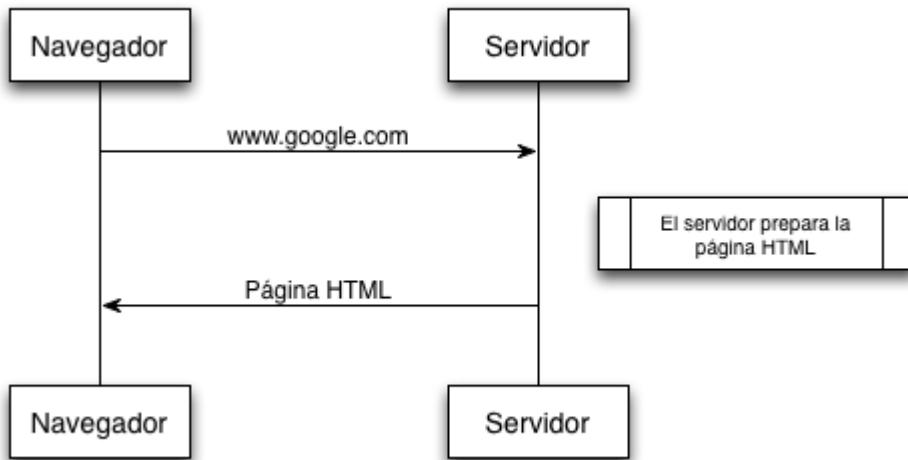
|  |            |
|--|------------|
| 7.2 Mapas . . . . .                                      | 70         |
| 7.3 Objetos . . . . .                                    | 72         |
| <b>Capítulo 8 Tablas . . . . .</b>                       | <b>77</b>  |
| 8.1 Tablas básicas . . . . .                             | 78         |
| 8.2 Tablas avanzadas . . . . .                           | 87         |
| <b>Capítulo 9 Formularios . . . . .</b>                  | <b>95</b>  |
| 9.1 Formularios básicos . . . . .                        | 95         |
| 9.2 Elementos de formulario . . . . .                    | 98         |
| 9.3 Formularios avanzados . . . . .                      | 106        |
| 9.4 Otros elementos de formulario . . . . .              | 110        |
| <b>Capítulo 10 Estructura y layout . . . . .</b>         | <b>117</b> |
| <b>Capítulo 11 Metainformación . . . . .</b>             | <b>121</b> |
| 11.1 Estructura de la cabecera . . . . .                 | 121        |
| 11.2 Metadatos . . . . .                                 | 123        |
| 11.3 DOCTYPE . . . . .                                   | 125        |
| <b>Capítulo 12 Otras etiquetas importantes . . . . .</b> | <b>129</b> |
| 12.1 Comentarios . . . . .                               | 129        |
| 12.2 JavaScript . . . . .                                | 130        |
| <b>Capítulo 13 Validación . . . . .</b>                  | <b>133</b> |
| 13.1 Validador del W3C . . . . .                         | 134        |
| <b>Capítulo 14 Ejercicios de HTML . . . . .</b>          | <b>137</b> |
| 14.1 Capítulo 3 . . . . .                                | 137        |
| 14.2 Capítulo 4 . . . . .                                | 137        |
| 14.3 Capítulo 5 . . . . .                                | 141        |
| 14.4 Capítulo 6 . . . . .                                | 142        |
| 14.5 Capítulo 7 . . . . .                                | 144        |
| 14.6 Capítulo 8 . . . . .                                | 145        |
| 14.7 Capítulo 9 . . . . .                                | 148        |
| 14.8 Ejercicio final . . . . .                           | 151        |

# Capítulo 1

**HTTP** (Hypertext Transfer Protocol) es un conjunto de reglas acordadas para transferir texto con atributos propios de Internet, que permite a dos máquinas comunicarse entre sí.

Se trata de un protocolo de transferencia de texto que opera a través de solicitudes entre un **cliente** y un **servidor**. HTTP es el término utilizado para describir este sencillo lenguaje basado en texto. No importa cómo se desarrolle, el objetivo del servidor será siempre entender y devolver respuestas de texto sencillo.

Es también uno de los protocolos que cierra la brecha entre los grupos de creación de redes y desarrollo de aplicaciones, ya que contiene información que es utilizada por los dos en la entrega y el desarrollo de aplicaciones basadas en web.



**Figura 1.1** Proceso de una petición web

Cada conversación en la web comienza con una petición. Esta petición es un mensaje de texto creado por un cliente (navegador, app) en un formato especial conocido como HTTP. El cliente envía esta petición a un servidor, y entonces espera la respuesta.

Una petición en lenguaje HTTP sería algo así:

```
GET / HTTP/1.1
Host: xkcd.com
Accept: text/html
User-Agent: Mozilla/5.0 (Macintosh)
```

Este simple mensaje comunica todo lo necesario acerca de qué recurso exactamente está solicitando el cliente.

La primera línea de una petición HTTP es la más importante y contiene dos cosas:

- el **URI** (Uniform Resource Identifier)
- el **método HTTP**

El URI (e.g. /, /contact, etc) es la única dirección o ubicación que identifica el recurso que el cliente quiere.

El método HTTP (e.g. GET) define lo que quieres hacer con el recurso. Los métodos HTTP son los verbos de la petición y definen las pocas maneras comunes que pueden actuar sobre el recurso:

- GET Recuperar el recurso del servidor
- POST Crear un recurso en el servidor
- PUT Actualizar el recurso en el servidor
- DELETE Borrar el recurso del servidor

Teniendo esto en cuenta, es muy fácil imaginarse cómo debería ser una petición HTTP para eliminar una entrada específica de un blog, por ejemplo:

```
| DELETE /blog/15 HTTP/1.1
```

Aunque hay nueve métodos HTTP definidos por la especificación HTTP, muchos de ellos no son muy utilizados. En realidad, muchos navegadores modernos no son compatibles con los métodos PUT y DELETE.

Además de la primera línea, una petición HTTP contiene invariablemente otras líneas de datos llamadas *request headers* o cabeceras. Estas cabeceras pueden suministrar una amplia gama de información como el servidor (*Host*), los formatos de respuesta que acepta el cliente (*Accept*) y la aplicación que utiliza el cliente para realizar la solicitud (*User-Agent*).

Una vez que el servidor ha recibido la petición, sabe exactamente qué recurso necesita el cliente (vía URI) y qué es lo que el cliente quiere hacer con ese recurso (vía método HTTP).

Traducido a HTTP, la respuesta enviada al navegador será algo como esto:

```
| HTTP/1.1 200 OK  
| Date: Sat, 02 Apr 2011 21:05:05 GMT  
| Server: lighttpd/1.4.19  
| Content-Type: text/html  
  
<html>  
|   <!-- ... -->  
</html>
```

La respuesta HTTP contiene el recurso solicitado (el contenido HTML en este caso), así como otra información acerca de la respuesta. La primera línea es especialmente importante y contiene el código de estado de

respuesta HTTP (200 en este caso). El código de estado comunica el resultado global de la solicitud. Existen diferentes códigos de estado que nos indican éxito, error o que el cliente necesita hacer algo (e.g. redirigir a otra página). Los códigos de estado más comunes son:

- 200 OK. Indica éxito.
- 304 Not Modified. Esto muestra que el recurso en cuestión no ha cambiado y que el navegador debe cargarlo desde su caché.
- 404 Not Found. Esto sugiere que el recurso no se encuentra en el servidor.
- 401 Authorization Required. Esto indica que el recurso está protegido y quiere unos credenciales válidos antes de que el servidor pueda conceder el acceso.
- 500 Internal Error. Esto significa que el servidor ha tenido un problema procesando la petición.

Al igual que la petición, una respuesta HTTP contiene piezas de información adicionales conocidas como cabeceras HTTP o *headers*. Por ejemplo, una importante cabecera de respuesta HTTP es *Content-Type*. El cuerpo o *body* de un mismo recurso puede ser devuelto en múltiples formatos diferentes como HTML, XML o JSON y la cabecera *Content-Type* utiliza *Internet Media Types* como `text/html` para decirle al cliente qué formato está devolviendo.

## Capítulo 2

**HTML** (*HyperText Markup Language*) es un lenguaje de marcas usado, de manera predominante, para la elaboración de páginas web. El lenguaje **HTML** es utilizado en forma de texto para describir, traducir y crear la estructura de estas páginas, tanto por programas, que traducen un diseño de página en código HTML, como por parte de las personas de manera directa.

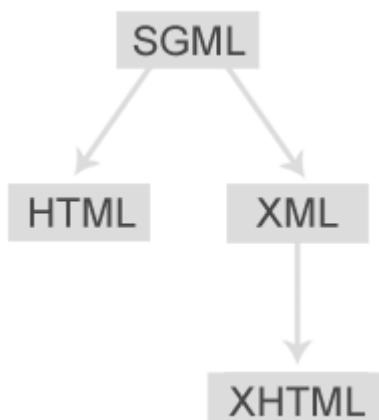
Este lenguaje es un estándar reconocido mundialmente y cuyas normas vienen dadas por el *World Wide Web Consortium* o **W3C** (<http://www.w3.org/TR/html401/>) - consorcio internacional que produce recomendaciones para la *World Wide Web* o Red Informática Mundial -, lo que hace que una misma página HTML sea visualizada de forma *similar* en diferentes navegadores y sistemas operativos.

Este organismo **W3C** elabora las normas a seguir para la creación de las páginas HTML o XHTML. Sin embargo, no es necesario conocer todas estas especificaciones, escritas es un lenguaje bastante formal, para diseñar páginas con este lenguaje. Las normas oficiales están escritas en inglés y se pueden consultar de forma gratuita en las siguientes direcciones:

- Especificación oficial de HTML 4.01 (<http://www.w3.org/TR/html401/>)

- Especificación oficial de XHTML 1.0 (<http://www.w3.org/TR/xhtml1/>)

El estándar XHTML 1.0 incluye el 95% del estándar HTML 4.01, ya que sólo añade pequeñas mejoras y modificaciones menores. Este sería un sencillo gráfico del esquema de relación y evolución de los lenguajes **HTML** y **XHTML** (*eXtensible HyperText Markup Language*) - que es básicamente HTML expresado como **XML** válido, una versión más estricta a nivel técnico - :

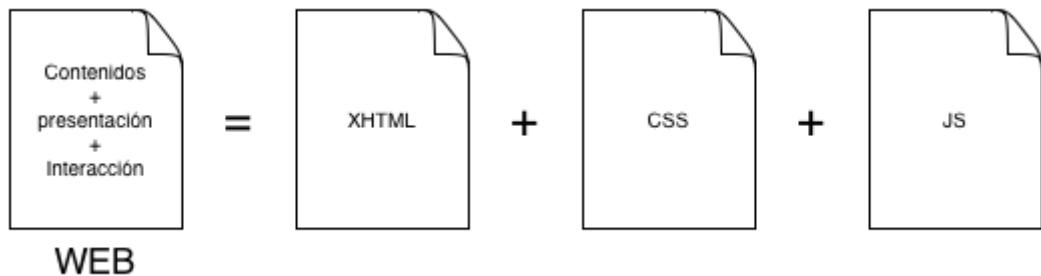


**Figura 2.1** Esquema de la evolución de HTML y XHTML

Originalmente, las páginas HTML sólo incluían información sobre sus contenidos de texto e imágenes. Con el desarrollo del **estándar HTML**, las páginas empezaron a incluir también información sobre el aspecto de sus contenidos.

La posterior aparición de tecnologías como **JavaScript** provocó que las páginas HTML también incluyeran el código de las aplicaciones (llamadas *scripts*) que se utilizan para crear páginas web dinámicas.

Una página web está compuesta, como vemos en este gráfico, por **contenidos, presentación e interacción**. Esto es, está subdividida de manera que, en vez de incluir en una misma página todo, se utilizan mecanismos para separar dichos contenidos: como son **CSS** y **JavaScript**.



**Figura 2.2 Separación de contenidos**

Este lenguaje de marcado se construye en forma de **etiquetas** o *tags* redondeadas por corchetes angulares (como <html>) dentro del contenido de la página web. Muchas de las etiquetas vienen 'en pareja', como pueden ser <h1> y </h1>, siendo la primera la etiqueta o **tag inicial o de apertura** y la segunda **de cierre**; pero también existen algunos *tags* conocidos como elementos vacíos, por ejemplo <img>.

**HTML** puede incluir o incrustar *scripts* escritos en lenguajes como **JavaScript** que afectan al comportamiento de las páginas en HTML y crean webs dinámicas, y puede también describir en cierta manera la apariencia de un documento, aunque para dar estilo a las páginas web escritas con este lenguaje, lo más habitual es la utilización de las **hojas de estilo en cascada**, *Cascading Style Sheets*, o su nombre más común, **CSS**.

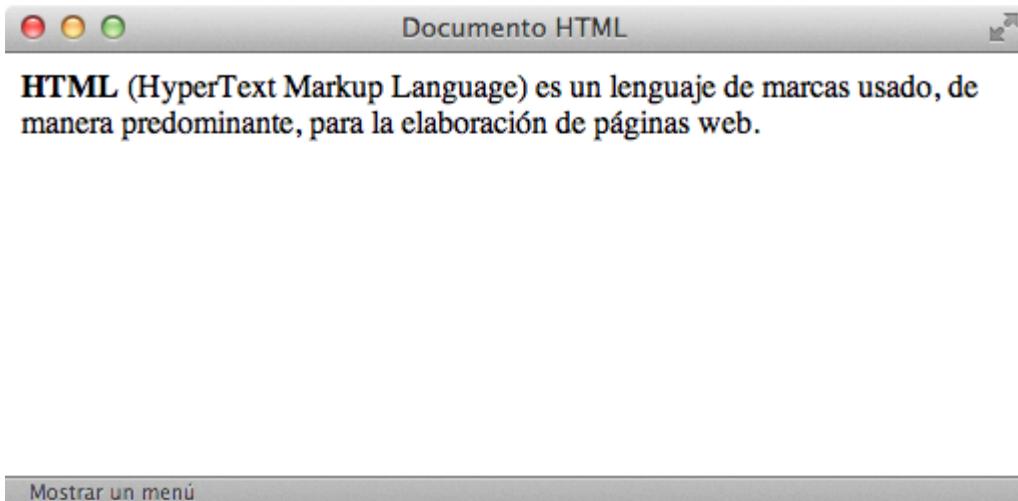
Las páginas HTML se dividen en:

- **cabecera** o *head*: incluye información sobre dicha página (título, idioma, hojas de estilos) - todo lo que el usuario no ve, a excepción del título.
- **cuerpo** o *body*: incluye todos sus contenidos (párrafos de texto, imágenes) - todo lo que ve el usuario.

```
<html>
  <head>
    <title>Documento HTML</title>
  </head>
  <body>
    <p><b>HTML</b> (HyperText Markup Language) es un lenguaje
       de marcas usado, de manera predominante, para la elaboración
```

```
    de páginas web.</p>
  </body>
</html>
```

Si guardásemos este código HTML con un editor de texto sin formato y la extensión **.html**, y lo abriésemos con cualquier navegador, esto sería lo que veríamos:



**Figura 2.3** Aspecto que muestra el primer documento HTML en cualquier navegador

En este ejemplo anterior, hemos utilizado los *tags* `<html>`, `<head>` y `<body>`, las tres etiquetas principales de un documento HTML:

- `<html>`: es el primer y último *tag* (`</html>`) de un documento HTML, lo que significa que ninguna etiqueta o contenido debe colocarse antes o después de éstas (a excepción del *doctype*).
- `<head>`: delimita la parte de la cabecera del documento y contiene información sobre éste que no se muestran al usuario (a excepción del *tag* `<title>` que muestra el título de la página en la parte superior izquierda de la ventana del navegador).
- `<body>`: delimita el cuerpo del documento HTML y encierra todos sus contenidos visibles.

## Capítulo 3

Las **etiquetas** son la estructura básica del HTML. Estas etiquetas o *tags* se componen y contienen otras propiedades, como son los **atributos** y el **contenido**.

HTML define un total de 91 etiquetas, de las cuales 10 se consideran obsoletas. Sin embargo, una etiqueta por sí sola a veces no contiene la suficiente información para estar completamente definida. Para ello contamos con los **atributos**: pares *nombre-valor* separados por "=" y escritos en la etiqueta inicial de un elemento después del nombre del elemento. El valor puede estar encerrado entre "comillas dobles" o 'simples'. Existen, también, algunos atributos que afectan al elemento por su presencia en la etiqueta de inicio, como puede ser el atributo *ismap* para el elemento *<img>*.

Esta sería la **estructura general de una línea de código en lenguaje HTML**:

```
| <tag attribute1="value1" attribute2="value2">content</tag>
```

O lo que es lo mismo, con un ejemplo:

```
| <a href="http://www.enlace.com" target="_blank">Ejemplo de enlace</a>
```

Donde:

- <a> es la etiqueta o *tag* inicial y </a> la etiqueta de cierre.
- href y target son los atributos.
- http://www.enlace.com y \_blank son las variables.
- Ejemplo de enlace es el contenido.

Aunque también existen elementos vacíos que no necesitan *tag* de cierre, cuya estructura sería ésta:

```
| <tag attribute1="value1" attribute2="value2" />
```

Estos elementos vacíos no constan de contenido, como por ejemplo, los *tags* <br> o <img>.

Aunque cada una de las etiquetas HTML define sus propios atributos, encontramos algunos comunes a muchas o casi todas las etiquetas, que se dividen en cuatro grupos según su funcionalidad:

- Atributos **básicos**
- Atributos de **internacionalización**
- Atributos de **eventos**
- Atributos de **foco**

Los atributos básicos se utilizan en la mayoría de etiquetas **HTML** y **XHTML**, aunque adquieren mayor sentido cuando se utilizan hojas de estilo en cascada (**CSS**):

|                            |   |
|----------------------------|---|
| <code>id="texto"</code>    | Establece un indicador único a cada elemento                    |
| <code>class="texto"</code> | Establece la clase CSS que se aplica a los estilos del elemento |
| <code>style="texto"</code> | Aplica de forma directa los estilos CSS de un elemento          |
| <code>title="texto"</code> | Establece el título del elemento (Mejora la accesibilidad)      |

## i18n

Estos atributos se utilizan en aquellas páginas que muestran sus contenidos en varios **idiomas** y las que quieran indicar de forma explícita el idioma de sus contenidos:

| lang="codigo"     | Indica el idioma del elemento  |
|-------------------|--|
| xml:lang="codigo" | Indica el idioma del elemento, aunque tiene más prioridad que el atributo anterior y es obligatorio si se incluye el atributo lang |
| dir               | Indica la dirección del texto  |

Estos atributos se utilizan en las páginas web que incluyen código **JavaScript** para realizar acciones dinámicas sobre los elementos de la página.

**Pueden ser utilizados por: *todos los elementos***

| onclick     | Ejecuta la acción cuando se realiza un clic sobre el elemento                 |
|-------------|---|
| ondblclick  | Ejecuta la acción cuando se realiza un doble clic sobre el elemento           |
| onmousedown | Ejecuta la acción cuando se detecta el botón pulsado del ratón                |
| onmouseup   | Ejecuta la acción cuando se detecta que se ha soltado el botón del ratón      |
| onmousemove | Ejecuta la acción cuando se detecta el movimiento del ratón sobre el elemento |
| onmouseout  | Ejecuta la acción cuando el ratón abandona el elemento                        |
| onmouseover | Ejecuta la acción cuando se detecta que el ratón se sitúa sobre el elemento   |

**Pueden ser utilizados por: <body>**

|          |   |
|----------|---|
| onload   | Ejecuta la acción cuando se carga el documento                                  |
| onunload | Ejecuta la acción cuando se abandona el documento                               |
| onresize | Ejecuta la acción cuando se ha modificado el tamaño de la ventana del navegador |

### Pueden ser utilizados por: *elementos de formulario y <body>*

|            |  |
|------------|--|
| onkeydown  | Ejecuta la acción cuando se detecta que la tecla esta pulsada          |
| onkeyup    | Ejecuta la acción cuando se detecta que se ha soltado la tecla pulsada |
| onkeypress | Ejecuta la acción cuando se pulsa una tecla                            |

### Pueden ser utilizados por: *varios*

|          |  |  |
|----------|--|--|
| onblur   | Ejecuta la acción cuando el elemento pierde el foco bien sea a través del ratón o por navegación tabulada  | <button>, <input>, <label>, <select>, <textarea>, <body> |
| onfocus  | Ejecuta la acción cuando el elemento obtiene el foco bien sea a través del ratón o por navegación tabulada | <button>, <input>, <label>, <select>, <textarea>, <body> |
| onchange | Ejecuta la acción cuando el valor de un control ha sido modificado   | <input>, <select>, <textarea>                            |
| onreset  | Ejecuta la acción cuando el formulario es reestablecido a sus valores por defecto                          | <form>   |
| onselect | Ejecuta la acción cuando un usuario selecciona texto en un campo de texto                                  | <input>, <textarea>                                      |
| onsubmit | Ejecuta la acción cuando el formulario es enviado  | <form>   |

Se le denomina **foco** o *focus*, cuando un control o elemento del documento ha sido seleccionado. Cuando ese elemento deja de estar seleccionado, "pierde el foco" y es el nuevo elemento seleccionado el que se dice que tiene "el foco".

|                   |  |
|-------------------|--|
| accesskey="letra" | Establece una tecla de acceso rápido a un elemento HTML  |
| tabindex="numero" | Establece la posición del elemento en el orden de tabulación de la página (valor entre 0 y 32.767) |
| onfocus, onblur   | Controlan los eventos JavaScript que se ejecutan cuando el elemento obtiene o pierde el foco       |

Además de etiquetas y atributos, HTML define el término **elemento** para referirse a las partes que componen los documentos HTML. Como ya hemos mencionado antes, la estructura general de una línea de código en lenguaje HTML sería ésta:

```
<tag attribute1="value1" attribute2="value2">content</tag>
```

El lenguaje HTML clasifica a todos los elementos en dos grupos: **elementos en línea** o *inline elements* y **elementos en bloque** o *block elements*.

La diferencia entre ambos viene dada por el **modelo de contenido**, por el **formato** y la **direccionalidad**. La principal diferencia entre los dos tipos de elementos es la forma en la que ocupan el espacio disponible en la página: los **elementos en bloque** siempre empiezan en una nueva línea y ocupan todo el espacio disponible hasta el final de la línea, mientras que los **elementos en línea** sólo ocupan el espacio necesario para mostrar sus contenidos.

---

---

|   |   |
|---|---|
| a | Define un anchor (anclaje o hipervínculo) |
|---|---|

|          |  |
|----------|--|
| abbr     | Marca las abreviaturas del texto y proporciona el significado de esas abreviaturas                   |
| acronym  | Marca las siglas o acrónimos del texto y proporciona el significado de esas siglas                   |
| b        | Indica que el texto debe ser representado en <i>bold</i> (o negrita)                                 |
| basefont | Permite cambiar algunas propiedades del texto  |
| bdo      | Anulación del algoritmo bidireccional (en referencia a la dirección de la escritura)                 |
| big      | Muestra el texto marcado con un tamaño de fuente más grande  |
| br       | <i>line break</i> - ruptura (o salto) de línea   |
| cite     | Se emplea para marcar una cita o una referencia a otras fuentes                                      |
| code     | Delimita el texto considerado un fragmento de código fuente  |
| dfn      | Marca las definiciones de ciertos términos y proporciona el significado de éstos                     |
| em       | <i>emphasis</i> – énfasis  |
| font     | Indica el tamaño, color, o fuente del texto que contiene   |
| i        | Muestra el texto marcado con un estilo en <i>cursiva</i> o itálica                                   |
| img      | Imagen   |
| input    | Posibilita y define la introducción de datos en el formulario  |
| kbd      | Indicar al usuario el texto que debe introducir  |
| label    | Asocia un rótulo o etiqueta a un campo de un formulario  |
| q        | <i>short quotations</i> - cita corta   |
| s        | <i>strike-through</i> - tachado  |
| samp     | Sirve identificar una muestra de los caracteres que forman la salida o el resultado de algún proceso |

| select   | Crea un contenedor mediante el cual el usuario puede seleccionar de una lista de opciones           |
|----------|---|
| small    | Aplica al texto marcado un tamaño de fuente más pequeño   |
| span     | Es un contenedor genérico en línea. Sirve para aplicar estilo al texto o agrupar elementos en línea |
| strike   | Muestra el texto tachado con una linea horizontal   |
| strong   | Marca con especial énfasis las partes más importantes de un texto                                   |
| sub      | Crea un subíndice posicionando el texto marcado por debajo de la linea                              |
| sup      | Crea un superíndice posicionando el texto marcado por encima de la linea                            |
| textarea | Crea un control de entrada de texto multilínea  |
| tt       | Representa como texto de teletipo o ancho fijo  |
| u        | Muestra el texto subrayado  |
| var      | Marca variables de programas y similares  |

| address    | Contiene la información de contacto con los autores del documento                   |
|------------|---|
| blockquote | Indica que el texto que encierra es una cita textual de otro texto externos         |
| center     | Crea una caja en bloque con el contenido centrado                                   |
| dir        | Crea listas multicolumna de directorios   |
| div        | Es un elemento en bloque genérico y sirve para crear secciones o agrupar contenidos |
| d1         | Crea una lista de definiciones  |
| fieldset   | Permite organizar en grupos los campos de un formulario                             |

|                        |   |
|------------------------|---|
| form                   | Actúa como contenedor de controles. Representa un formulario  |
| h1, h2, h3, h4, h5, h6 | Crea un encabezado o título de primer, segundo, tercer, cuarto, quinto o sexto nivel para una sección del documento respectivamente |
| hr                     | Crea una linea de separación horizontal   |
| isindex                | Crea un control de entrada de texto de una línea  |
| menu                   | Crea un menú  |
| noframes               | (sin marcos) - aporta contenidos alternativos a los marcos  |
| nos-cript              | Contenedor de contenido alternativo para la representación no basada en <i>scripts</i>  |
| ol                     | Crea una lista ordenada   |
| p                      | Párrafo   |
| pre                    | Permite que el texto conserve el formato y sea mostrado tal cual  |
| table                  | Tabla   |
| ul                     | Crea una lista no ordenada  |

|           |                               |
|-----------|-------------------------------|
| dd        | Descripción de una definición |
| dt        | Término definido              |
| frame-set | Subdivisión en ventanas       |
| li        | Objeto de lista               |
| tbody     | Cuerpo de tabla               |
| td        | Celda de datos de una tabla   |
| tfoot     | Pie de tabla                  |
| th        | Celda de encabezado de tabla  |

|        |   |
|--------|---|
| thead  | Cabecera de tabla                       |
| tr     | Fila de una tabla                       |
| button | Botón                                   |
| del    | Texto borrado                           |
| iframe | Subventana en línea                     |
| ins    | Texto insertado                         |
| map    | Mapa de imágenes en el lado del cliente |
| object | Objeto genérico incluído                |
| script | Sentencias de <i>script</i>             |

Esta página se ha dejado vacía a propósito

## Capítulo 4

La mayor parte de las **páginas HTML** están formadas por **texto** (llegando a ser más del 90% del código de la página). Este lenguaje define *tags* para **estructurar** el contenido y otros para **marcar** elementos importantes dentro del texto.

El proceso de **estructuración** de un texto simple consiste en indicar las diferentes zonas o secciones que componen un texto: como son los párrafos o títulos de sección.

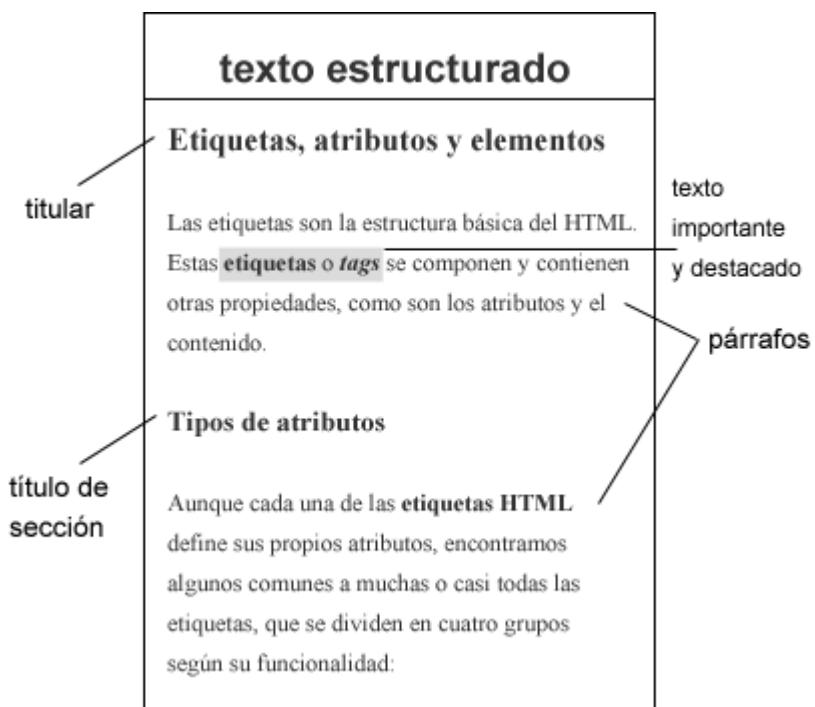
El proceso posterior a estructurar el texto consiste en **marcar** los diferentes elementos dentro de éste: definiciones, abreviaturas, textos importantes, textos modificados, citas a otras referencias, etc.

Un **ejemplo** de texto original y otro de texto estructurado sería:

| texto original  | texto estructurado   |
|---|--|
| <p>Etiquetas, atributos y elementos. Las etiquetas son la estructura básica del HTML. Estas etiquetas o tags se componen y contienen otras propiedades, como son los atributos y el contenido. Tipos de atributos. Aunque cada una de las etiquetas HTML define sus propios atributos, encontramos algunos comunes a muchas o casi todas las etiquetas, que se dividen en cuatro grupos según su funcionalidad:</p> | <p><b>Etiquetas, atributos y elementos</b></p> <p>Las etiquetas son la estructura básica del HTML. Estas <b>etiquetas o tags</b> se componen y contienen otras propiedades, como son los atributos y el contenido.</p> <p><b>Tipos de atributos</b></p> <p>Aunque cada una de las <b>etiquetas HTML</b> define sus propios atributos, encontramos algunos comunes a muchas o casi todas las etiquetas, que se dividen en cuatro grupos según su funcionalidad:</p> |

**Figura 4.1** Resultado de estructurar un texto sencillo

Este ejemplo muestra una transformación de un párrafo con un **texto simple** a un **texto estructurado y marcado** donde encontramos:



**Figura 4.2** Resultado de marcar un texto sencillo

|                          |  |
|--------------------------|--|
|                          | <p>  |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos       |
| <b>Atributos propios</b> | -  |
| <b>Tipo de elemento</b>  | En bloque                                    |
| <b>Descripción</b>       | Delimita el contenido de un párrafo de texto |

Esta etiqueta <p> permite definir los **párrafos** que forman el texto de una página. Como se puede ver en la tabla anterior, estos párrafos son **elementos en bloque**, por lo que ocupan toda la anchura del navegador. No presentan **atributos** específicos pero sí se les pueden asignar los atributos comunes de HTML básicos, de internacionalización y de eventos.

Un ejemplo de **código HTML** con la etiqueta <p> sería:

```
<html>
  <head>
    <title>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>
```

Y un navegador lo visualizaría de esta manera:



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.

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.



**Figura 4.3** Ejemplo de texto HTML estructurado con párrafos

|                          |   |
|--------------------------|---|
|                          | <h1> <h2> <h3> <h4> <h5> <h6>   |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos                                |
| <b>Atributos propios</b> | -   |
| <b>Tipo de elemento</b>  | En bloque   |
| <b>Descripción</b>       | Define los títulos de las secciones de mayor importancia de la página |

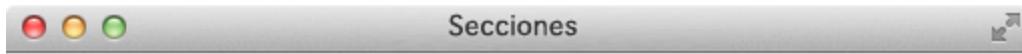
Estas etiquetas <h1>...<h6> definen **títulos de sección**, no secciones completas. Como se puede ver en la tabla anterior y al igual que el tag <p>, estas secciones son **elementos en bloque**, por lo que ocupan toda la anchura del navegador y tampoco presentan **atributos** específicos pero sí se les pueden asignar los atributos comunes de HTML básicos, de internacionalización y de eventos.

Los navegadores asignan de forma automática el tamaño del título de cada sección en función de su importancia, que se puede modificar utilizando las hojas de estilos **CSS**.

Un ejemplo de **código HTML** con la etiqueta <p> más las etiquetas <h1> y <h2> sería:

```
<html>
  <head>
    <title>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>
```

Y un navegador lo visualizaría de esta manera:



## Titular de la página

Párrafo de introducción

### La primera sub-sección

Párrafo de contenido

### Otra subsección

Más párrafos de contenido

**Figura 4.4** Ejemplo de texto HTML estructurado con párrafos y secciones

El siguiente paso posterior a la **estructuración** del texto consistiría en el **marcado** de éste. Para el marcado básico de un texto encontraríamos estas etiquetas:

|                          |   |
|--------------------------|---|
|                          | <em>  |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos                                  |
| <b>Atributos propios</b> | -   |
| <b>Tipo de elemento</b>  | En línea  |
| <b>Descripción</b>       | Realza la importancia del texto que encierra - <i>cursiva</i>           |
|                          | <strong>  |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos                                  |
| <b>Atributos propios</b> | -   |
| <b>Tipo de elemento</b>  | En línea  |
| <b>Descripción</b>       | Realza con la máxima importancia el texto que encierra - <b>negrita</b> |

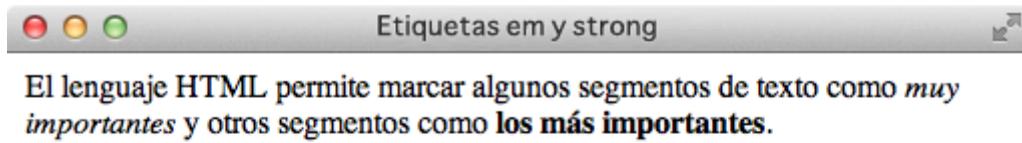
Un ejemplo de **código HTML** con las etiquetas <em> y <strong> sería:

```

<html>
  <head>
    <title>Etiquetas em y strong</title>
  </head>
  <body>
    <p>El lenguaje HTML permite marcar algunos segmentos de texto
      como <em>muy importantes</em> y otros segmentos como <strong>los
      más importantes</strong>. </p>
  </body>
</html>

```

Y un navegador lo visualizaría de esta manera:



**Figura 4.5** Ejemplo de uso de etiquetas em y strong

HTML también permite marcar de forma adecuada las modificaciones realizadas en el contenido de una página: el texto que **ha sido eliminado** y el texto que **ha sido añadido**:

| <ins>                    |   |
|--------------------------|---|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | <code>cite="url"</code> Indica la URL de la página en la que se puede obtener más información sobre el motivo por el que se realizó la modificación.<br><code>datetime="fecha"</code> Especifica la fecha y hora en la que se realizó el cambio |
| <b>Tipo de elemento</b>  | En bloque y en línea  |
| <b>Descripción</b>       | Se emplea para marcar una modificación en los contenidos originales consistente en la inserción de un nuevo contenido   |
| <del>                    |   |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | <code>cite="url"</code> Indica la URL de la página en la que se puede obtener más información sobre el motivo por el que se   |

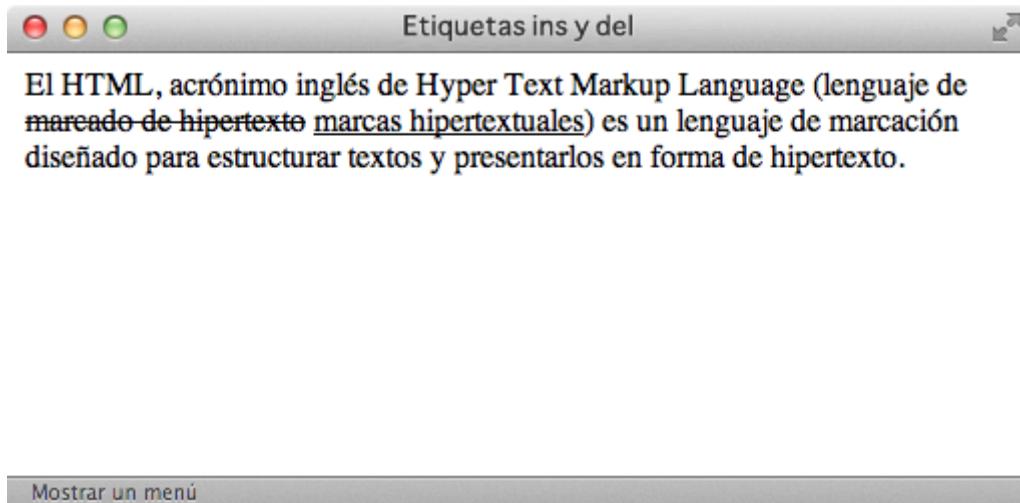
|                         |   |
|-------------------------|---|
|                         | <del>   |
|                         | realizó la modificación.<br>datetime="fecha" Especifica la fecha y hora en la que se realizó el cambio            |
| <b>Tipo de elemento</b> | En bloque y en línea  |
| <b>Descripción</b>      | Se emplea para marcar una modificación en los contenidos originales consistente en el borrado de cierto contenido |

Por defecto, el texto eliminado (<del>) se muestra tachado y el texto insertado (<ins>) se muestra subrayado.

Un ejemplo de **código HTML** con las etiquetas <ins> y <del> sería:

```
<html>
  <head>
    <title>Etiquetas ins y del</title>
  </head>
  <body>
    <p>El HTML, acrónimo inglés de Hyper Text Markup Language
(lenguaje
      de <del datetime="20091025" cite="http://www.enlace.com">marcado
      de
        hipertexto</del> <ins datetime="20091026"
      cite="http://enlace.com">
        marcas hipertextuales</ins>) es un lenguaje de marcación diseñado
        para estructurar textos y presentarlos en forma de
      hipertexto.</p>
  </body>
</html>
```

Y un navegador lo visualizaría de esta manera:



**Figura 4.6** Ejemplo de uso de etiquetas ins y del

Además de estas dos, encontramos la etiqueta <blockquote> para incluir citas textuales en las páginas web.

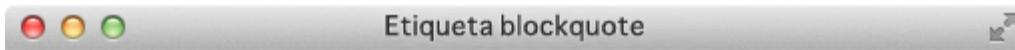
| <blockquote>             |  |
|--------------------------|--|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos   |
| <b>Atributos propios</b> | cite="url" Indica la URL de la página en la que se puede obtener más información sobre el motivo por el que se realizó la modificación |
| <b>Tipo de elemento</b>  | En bloque  |
| <b>Descripción</b>       | Se emplea para indicar que el texto que encierra es una cita textual de otro texto externo   |

Un ejemplo de **código HTML** con la etiqueta <blockquote> sería:

```
<html>
  <head>
    <title>Etiqueta blockquote</title>
  </head>
  <body>
    <p>Según el W3C, el valor del atributo <em>cite</em> en las
       etiquetas <strong>blockquote</strong> tiene el siguiente
       significado:</p>
    <blockquote cite="http://www.w3.org/TR/html401/struct/text.html">
      "El valor de este atributo es una dirección URL que indica el
```

```
documento original de la cita."</blockquote>
</body>
</html>
```

Y un navegador lo visualizaría de esta manera:



Según el W3C, el valor del atributo *cite* en las etiquetas **blockquote** tiene el siguiente significado:

"El valor de este atributo es una dirección URL que indica el documento original de la cita."

**Figura 4.7** Ejemplo de uso de etiqueta **blockquote**

La etiqueta `<abbr>` marca las abreviaturas de un texto y la etiqueta `<acronym>` se emplea para marcar las siglas o acrónimos del texto. En ambos casos, el atributo `title` se puede utilizar para incluir el significado completo de la abreviatura o sigla. La mayoría de navegadores muestran por defecto un borde inferior punteado para estos elementos.

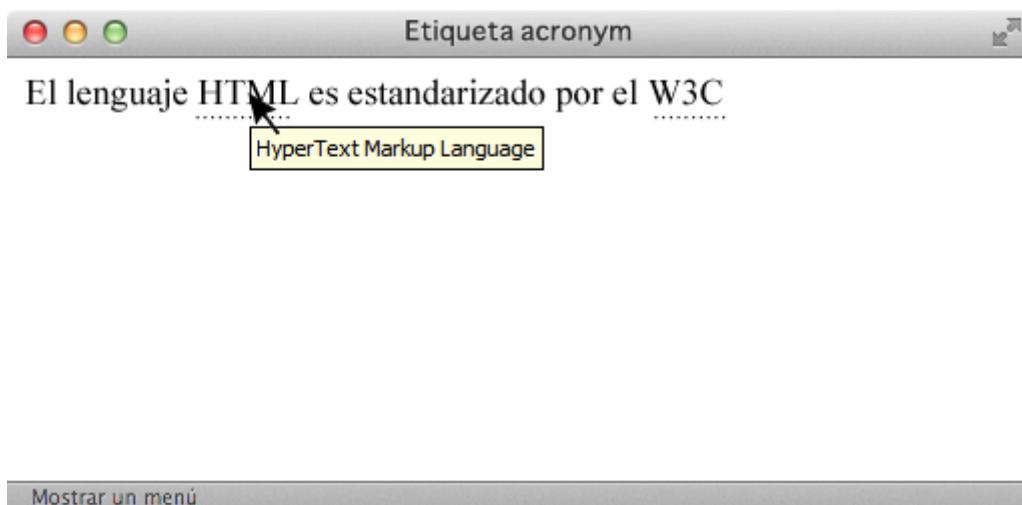
|                          | <code>&lt;abbr&gt;</code>   |
|--------------------------|---|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | <code>title="texto"</code> Indica el significado completo de la abreviatura                         |
| <b>Tipo de elemento</b>  | En línea  |
| <b>Descripción</b>       | Se emplea para marcar las abreviaturas del texto y proporcionar el significado de esas abreviaturas |

|                          |   |
|--------------------------|---|
|                          | <acronym>   |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | title="texto" Indica el significado completo del acrónimo o sigla                                   |
| <b>Tipo de elemento</b>  | En línea  |
| <b>Descripción</b>       | Se emplea para marcar las siglas o acrónimos del texto y proporcionar el significado de esas siglas |

Un ejemplo de **código HTML** con la etiqueta <acronym> sería:

```
<html>
  <head>
    <title>Etiqueta acronym</title>
  </head>
  <body>
    <p>El lenguaje <acronym title="HyperText Markup Language">
      HTML</acronym> es estandarizado por el <acronym title="World
      Wide Web Consortium">W3C</acronym>.</p>
  </body>
</html>
```

Y un navegador lo visualizaría de esta manera:



**Figura 4.8** Ejemplo de uso de etiqueta acronym

La etiqueta `<dfn>` proporciona al usuario la definición de todas las palabras para las que se considere apropiado, y la etiqueta `<cite>` se utiliza para marcar un texto como una citación:

|                          |   |
|--------------------------|---|
|                          | <code>&lt;dfn&gt;</code>  |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | <code>title="texto"</code> Indica el significado completo del término                                     |
| <b>Tipo de elemento</b>  | En línea  |
| <b>Descripción</b>       | Se emplea para marcar las definiciones de ciertos términos y proporcionar el significado de esos términos |

Un ejemplo de **código HTML** con la etiqueta `<dfn>` sería:

```
<p>Con estos síntomas, podría tratarse de un caso de <dfn title="Imagen o sensación subjetiva, propia de un sentido, determinada por otra sensación que afecta a un sentido diferente">sinestesia</dfn></p>
```

|                          |   |
|--------------------------|---|
|                          | <code>&lt;cite&gt;</code>                                       |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos                          |
| <b>Atributos propios</b> | -   |
| <b>Tipo de elemento</b>  | En línea  |
| <b>Descripción</b>       | Se emplea para marcar una cita o una referencia a otras fuentes |

La diferencia entre `<cite>` y `<blockquote>` está en que el elemento `<cite>` marca el autor de la cita (persona, documento, etc.) y `<blockquote>` marca el contenido de la propia cita.

Un ejemplo de **código HTML** con ambas etiquetas sería:

Como dijo <cite>Mahatma Gandhi</cite>:  
<blockquote>Vive como si fueras a morir mañana y aprende como si fueras a vivir para siempre.</blockquote>

La etiqueta <span> permite agrupar varios elementos en línea seguidos dentro de un mismo bloque (por ejemplo, varias palabras seguidas en un párrafo), para después darles formato con las hojas de estilo **CSS**. Se emplea para marcar cualquier elemento que no se puede marcar con las otras etiquetas definidas.

La etiqueta <span> se visualiza por defecto con el mismo aspecto que el texto normal. Por tanto, es habitual utilizar esta etiqueta junto con los atributos *id* y *class* para modificar posteriormente su aspecto con **CSS**.

Este *tag* sólo se puede utilizar, como ya hemos mencionado, para encerrar **contenidos y etiquetas en línea**. Cuando se quieren estructurar elementos en bloque, se utiliza la etiqueta <div>.

Un ejemplo de **código HTML** con la etiqueta <span> sería:

Importante: Si quiere ponerse en contacto con nuestra empresa, puede hacerlo en el teléfono <span class="telefono">0034 900 000 000</span> o a través de la dirección de correo <span class="email">contacto@empresa.com</span>

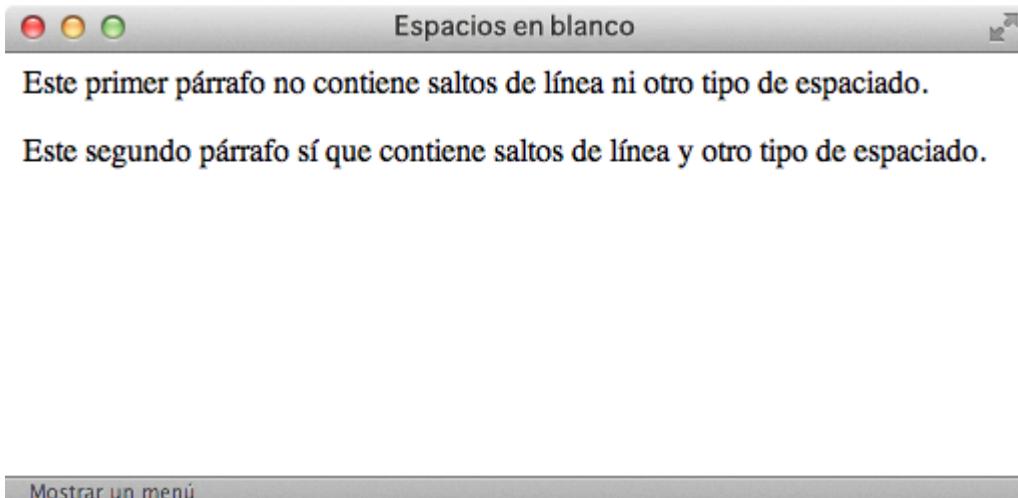
El lenguaje **HTML** considera **espacio en blanco** a: los espacios en blanco, los tabuladores, los retornos de carro y el carácter de nueva línea. HTML ignora todos los espacios en blanco sobrantes, es decir, todos los espacios en blanco que no son el espacio en blanco que separa las palabras.

Un ejemplo de este comportamiento sería:

```
<html>
  <head>
    <title>Espacios en blanco</title>
  </head>
  <body>
    <p>Este primer párrafo no contiene saltos de línea ni otro tipo
```

```
de espaciado.</p>
    <p>Este segundo párrafo sí que contiene saltos
    de
    línea
    y otro tipo de espaciado.</p>
</body>
</html>
```

Que un navegador visualizaría de esta manera:



**Figura 4.9** Ejemplo de uso de la etiqueta p con espacios en blanco y nuevas líneas

Sin embargo, existen alternativas para incluir **espacios en blanco adicionales**. Esto se consigue sustituyendo cada nuevo espacio en blanco por el texto &nbsps;.

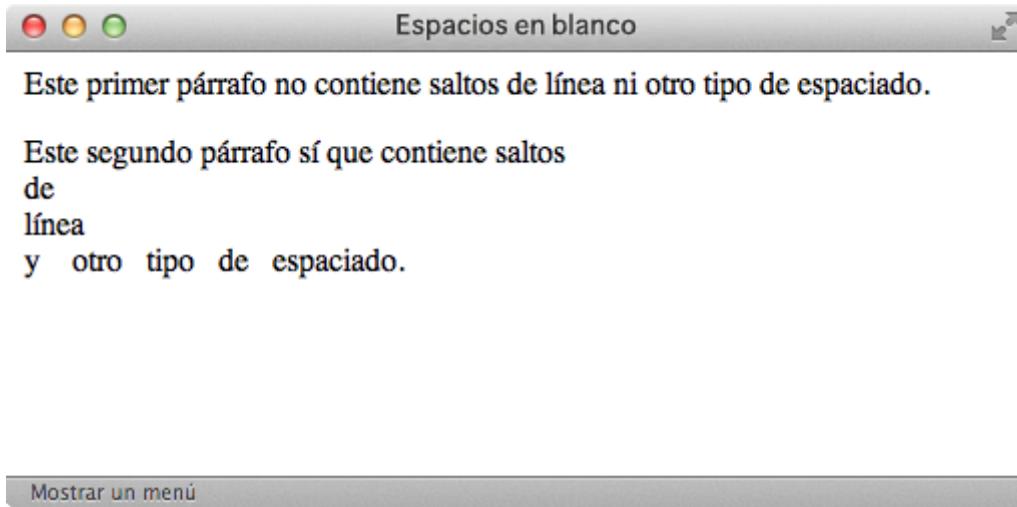
Cada texto &nbsps; equivale a un sólo espacio en blanco, por lo que se deben escribir tantos &nbsps; seguidos como espacios en blanco se quieran conseguir.

Si quisiéramos visualizar el texto del ejemplo anterior con todos los espacios adicionales, debería escribirse de esta manera:

```
<html>
    <head>
        <title>Espacios en blanco</title>
    </head>
    <body>
        <p>Este primer párrafo no contiene saltos de línea ni otro tipo
        de espaciado.</p>
        <p>Este segundo párrafo sí que contiene saltos <br/>
```

```
de <br/>
línea <br/>
y &nbsp;&nbsp; otro &nbsp; tipo &nbsp; de &nbsp; espaciado.</p>
</body>
</html>
```

Que un navegador visualizaría de esta manera:



**Figura 4.10** Ejemplo de uso &nbsp;

Este ejemplo de **código HTML** incluye la etiqueta `<br/>`, que se explica a continuación.

Para forzar una nueva línea, o lo que es lo mismo, lo equivalente a presionar la tecla *Enter* o *Intro* escribiendo un texto, se utiliza el *tag* `<br>`. Se trata de una **etiqueta vacía**, es decir, no encierra ningún texto.

| <br>                     |  |
|--------------------------|--|
| <b>Atributos comunes</b> | básicos  |
| <b>Atributos propios</b> | -  |
| <b>Tipo de elemento</b>  | En línea y etiqueta vacía                      |
| <b>Descripción</b>       | Fuerza al navegador a insertar una nueva línea |

Existen varias formas de expresar esta etiqueta. Si bien se puede abrir y cerrar de forma consecutiva (`<br></br>`), la forma de uso más común es abriendo y cerrando un único *tag* de esta forma: `<br/>` o `<br />`.

Existe una manera de mostrar el **texto tal y como está escrito**, respetando los espacios en blanco y las nuevas líneas. Se utiliza, por ejemplo, cuando una página debe mostrar directamente el texto generado por alguna aplicación.

| <pre>                    |   |
|--------------------------|---|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos                |
| <b>Atributos propios</b> | -   |
| <b>Tipo de elemento</b>  | En bloque   |
| <b>Descripción</b>       | Muestra el texto que encierra tal y como está escrito |

Los elementos <pre> son algo especiales, ya que los navegadores les aplican algunas reglas:

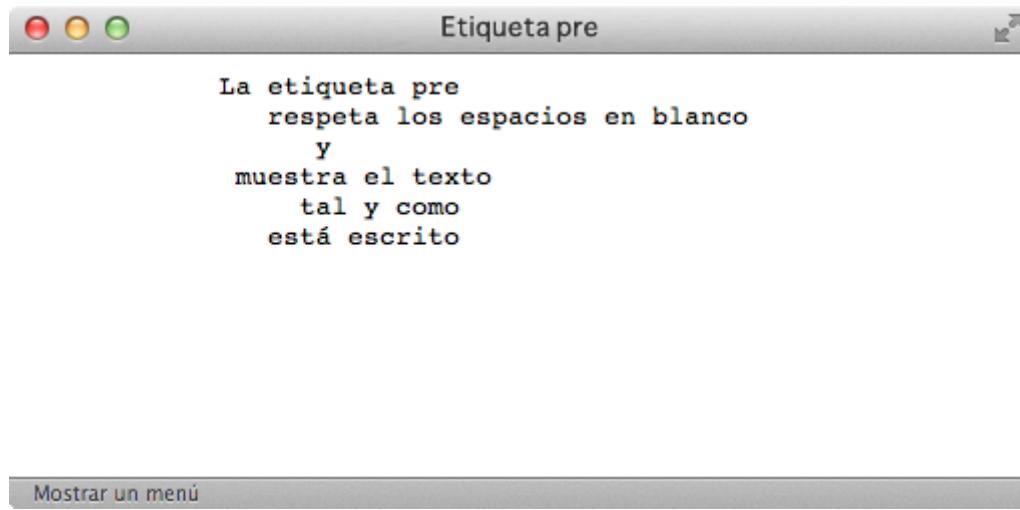
- Mantienen todos los espacios en blanco (tabuladores, espacios y nuevas líneas)
- Muestra el texto con un tipo de letra de ancho fijo (todas las letras de la misma anchura)
- No se ajusta la longitud de las líneas (las líneas largas producen un scroll horizontal), lo que provoca que la anchura de la página sea superior a la anchura de la ventana del navegador.

Un ejemplo de **código HTML** con la etiqueta <pre> sería:

```
<html>
  <head>
    <title>Etiqueta pre</title>
  </head>
  <body>
    <pre>
      La etiqueta pre
      respeta los espacios en blanco
      y
      muestra el texto
      tal y como
      está escrito
    </pre>
  </body>
</html>
```

```
</p>
</pre>
</body>
</html>
```

Y un navegador lo visualizaría de esta manera:



**Figura 4.11** Ejemplo de uso de la etiqueta pre

También existe la etiqueta `<code>` que se utiliza para mostrar código fuente de cualquier lenguaje de programación.

| <code>                   |   |
|--------------------------|---|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos                      |
| <b>Atributos propios</b> | -   |
| <b>Tipo de elemento</b>  | En bloque   |
| <b>Descripción</b>       | Delimita el texto considerado un fragmento de código fuente |

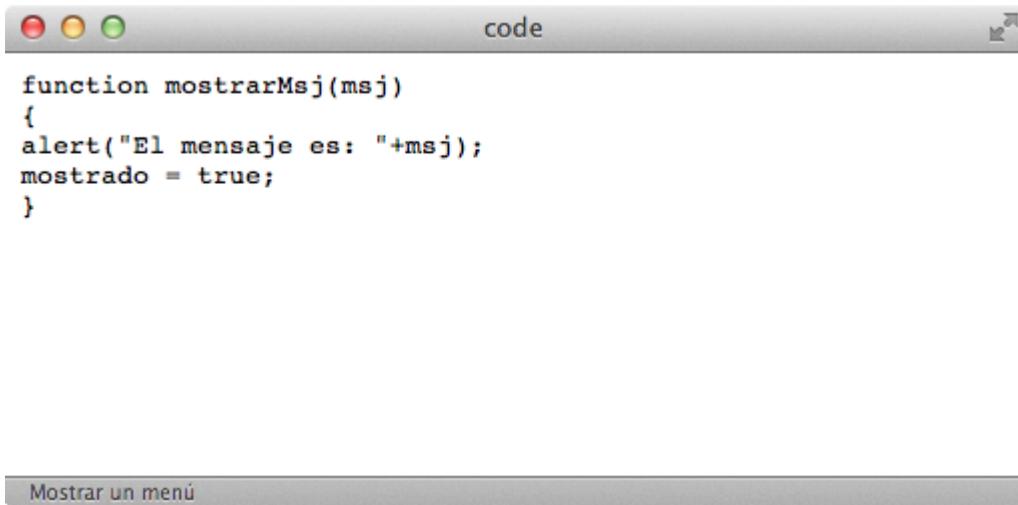
Las reglas que los navegadores aplican a las etiquetas `<code>` son:

- Muestra el texto con un tipo de letra de ancho fijo (todas las letras de la misma anchura)

Sin embargo, no respeta los espacios en blanco ni las líneas, por lo que, en ese sentido, su comportamiento es similar a la etiqueta `<p>`.

Un ejemplo de **código HTML** con la etiqueta `<code>` sería:

```
<html>
  <head>
    <title>code</title>
  </head>
  <body>
    <code>
      function mostrarMsj(msj)
      {
        alert("El mensaje es: "+msj);
        mostrado = true;
      }
    </code>
  </body>
</html>
```



**Figura 4.12** Ejemplo de uso de la etiqueta `code`

Existen **caracteres** que se utilizan habitualmente en los textos que no se pueden incluir directamente en las páginas web o que pueden darnos problemas. Estos son:

- Los caracteres que utiliza el lenguaje **HTML** para definir sus etiquetas.
- Los caracteres **propios de idiomas que no son el inglés**.

Para los caracteres propios del lenguaje HTML existen algunas expresiones o **entidades HTML** que los sustituyen:

|        |                   |
|--------|-------------------|
| &lt;   | <                 |
| &gt;   | >                 |
| &amp;  | &                 |
| &quot; | "                 |
| &nbsp; | espacio en blanco |
| &apos; | '                 |

En cuanto a los caracteres propios de idiomas diferentes al inglés, cuando todos los procesos involucrados (entorno de desarrollo, servidor web y navegador) utilizan la misma codificación de caracteres (por ejemplo, **UTF-8**), el texto se verá correctamente en el navegador. Pero si la codificación cambia sin realizar una conversión correcta, el navegador mostrará caracteres extraños.

La solución más correcta y sencilla es sustituir estos caracteres potencialmente problemáticos por su **entidad HTML**, ya que si se utilizan las entidades HTML en vez de los caracteres problemáticos, es indiferente pasar de una codificación de caracteres a otra diferente.

|          |   |
|----------|---|
| &ntilde; | ñ |
| &Ntilde; | Ñ |
| &aacute; | á |
| &eacute; | é |
| &iacute; | í |
| &oacute; | ó |
| &uacute; | ú |
| &Aacute; | Á |
| &Eacute; | É |

|          |   |
|----------|---|
| &Iacute; | Í |
| &Oacute; | Ó |
| &Uacute; | Ú |
| &euro;   | € |

Puedes consultar la tabla completa de entidades HTML en Wikipedia:

[http://en.wikipedia.org/wiki/  
List\\_of\\_XML\\_and\\_HTML\\_character\\_entity\\_references](http://en.wikipedia.org/wiki/List_of_XML_and_HTML_character_entity_references)

## Capítulo 5

El **lenguaje HTML** (*HyperText Markup Language*), como ya hemos mencionado anteriormente, es un lenguaje de marcas. Literalmente, podría ser traducido como "lenguaje de marcado para hipertexto". El elemento principal del **hipertexto** es el hiperenlace, también llamado enlace web o simplemente **enlace**.

Un enlace es una conexión desde un recurso web a otro. Establece **relaciones entre dos recursos** (principalmente páginas web, pero también imágenes, documentos o archivos). Un enlace comienza en un recurso y apunta hacia otro.

Para poder comprender el funcionamiento y creación de los **enlaces**, es importante conocer y dominar el concepto de **URL**. El acrónimo **URL** (*Uniform Resource Locator*) es un localizador de recursos uniformes y hace referencia a un único identificador de cada recurso en Internet. Esta secuencia de caracteres nombra recursos en Internet para su localización o identificación.

La URL de un recurso tiene dos objetivos principales:

- **Identificar** de forma única a ese recurso: cada página en Internet tiene un nombre único, lo que posibilita la creación de enlaces que apunten de forma inequívoca a una página determinada.
- **Localizar** de forma eficiente ese recurso.

El **formato general** de una URL es: esquema://máquina/directorio/archivo

Por ejemplo: <http://www.arkaitzgarro.com/xhtml/capitulo-5.html>

Cuyas partes son:

|   |  |   |
|---|--|---|
| http://   | www.arkaitzgarro.com                                     | /xhtml/<br>capitulo-5.html                  |
| Mecanismo que debe utilizar el navegador para acceder a ese recurso | Nombre del dominio, hace referencia a una dirección I.P. | Recurso específico al que se quiere acceder |

También podemos encontrar URL más complejas como:

<http://www.arkaitzgarro.com/xhtml/capitulo-5.html?page=5#url>

Donde además de las partes anteriores encontramos:

- ?page=5 **Consulta**: Información adicional para acceder al recurso. Comienza con el carácter ? seguido de una sucesión de palabras y/o números separados por = y &.
- #url **Sección**: el navegador se posiciona en dicha sección de la página. Comienza con el carácter #.

Al igual que en el texto, las URL también presentan problemas con algunos caracteres, por lo que se utiliza una **codificación de caracteres** que asegura su correcto funcionamiento.

|   |     |   |     |
|---|-----|---|-----|
| / | %2F | ñ | %F1 |
| : | %3A | á | %E1 |
| = | %3D | é | %E9 |
| " | %22 | í | %ED |

|                     |     |   |     |
|---------------------|-----|---|-----|
| '                   | %60 | ó | %F3 |
| (espacio en blanco) | %20 | ú | %2F |
| ?                   | %3F | ç | %FA |
| @                   | %40 | Ñ | %D1 |
| &                   | %26 | Á | %C1 |
| \                   | %5C | É | %C9 |
| ~                   | %7E | Í | %CD |
| #                   | %23 | Ó | %D3 |
|                     |     | Ú | %DA |
|                     |     | Ç | %C7 |

Dentro de una página web podemos encontrar decenas de **enlaces** y éstos pueden ser de diferentes tipos. Como principal distinción encontramos URL completas, absolutas y relativas:

- **URL completas:** `http://www.dominio.com/directorio/recurso` Incluyen todas las partes de la URL (protocolo, dominio y ruta) y no se necesita más información para obtener el recurso enlazado. El enlace está completamente definido.
- **URL absolutas:** `/directorio/recurso` Se parte de la base que el recurso al que queremos llegar esta en el mismo servidor del que partimos.
- **URL relativas:** `../recurso` Son URL incompletas, prescinden de algunas partes de las URL para hacerlas más breves (de la parte del protocolo, del nombre del servidor e incluso de parte o toda la ruta del recurso enlazado). Por esto, es necesario tener información adicional, es decir, se debe conocer la URL del origen del enlace. Es una versión abreviada de una URL absoluta.

Dentro de las **URL relativas**, existen distintos tipos:

- El origen y el destino del enlace se encuentran en el **mismo directorio**:

|                 |   |
|-----------------|---|
| 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 | pagina2.html - mismo directorio   |
| URL Completa    | <a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina2.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina2.html</a> |
| URL Relativa    | <a href="#">pagina2.html</a>  |

- El destino del enlace se encuentra **en un nivel superior**:

|                 |   |
|-----------------|---|
| 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 | pagina2.html - directorio superior llamado ruta2  |
| URL Completa    | <a href="http://www.ejemplo.com/ruta1/ruta2/pagina2.html">http://www.ejemplo.com/ruta1/ruta2/pagina2.html</a>             |
| URL Relativa    | <a href="#">../pagina2.html</a>   |

Si el destino se encuentra **dos niveles por encima**, se debe incluir ../  
dos veces seguidas: [../../pagina2.html](#)

Y si el destino está en **otro directorio**, llamado por ejemplo ruta4 que se encuentra en la **raíz del servidor**, la URL relativa sería:  [../../../../../../ruta4/pagina2.html](#)

- El destino del enlace se encuentra **en un nivel inferior**:

|                 |   |
|-----------------|---|
| 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 | pagina2.html - directorio inferior llamado ruta4  |
| URL Completa    | <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>  |

También se pueden indicar varios directorios seguidos: ruta4/ruta5/ruta6/pagina2.html

- El destino del enlace se encuentra **en algún lugar en la raíz del servidor:**

| Página origen   | http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html   |
|-----------------|---|
| Página enlazada | pagina2.html - directorio en la raíz del servidor ruta7 |
| URL Completa    | 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 sólo le añade el protocolo y el nombre del servidor origen.

Son los creados mediante el *tag* <a>:

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

Un ejemplo de **código HTML** con la etiqueta <a> sería:

```
<a href="http://www.google.com">Enlace básico a una web</a>
```

Este enlace apunta a una web, en concreto, a la página principal de Google. Pero de la misma manera, podríamos enlazarlo con una imagen, un

documento, etc., ya que el atributo `href` puede apuntar a cualquier tipo de recurso al que pueda acceder el navegador.

Por ejemplo:

```
<a href="http://www.ejemplo.com/imagen.jpg">Enlace básico a una  
imagen</a>
```

o

```
<a href="http://www.ejemplo.com/informe.pdf">Enlace básico a un archivo  
pdf</a>
```

El atributo `name` permite definir enlaces dentro de una misma página web. Esto es muy útil cuando se trata de documentos largos divididos en secciones.

Por ejemplo, si creas el "enlace vacío"

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

y en otro lugar de la página creas el enlace

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

al pinchar en éste, el navegador accede a la página apuntada por la URL y baja directamente a la sección cuyo nombre se indica después del símbolo `#`. Los enlaces directos a secciones también funcionan con el atributo `id` de cualquier elemento.

La definición anterior del *tag* `<a>` no es su definición completa, ya que dispone de otros atributos específicos también muy importantes que nos servirán para crear enlaces muchos más complejos. Estos **atributos específicos**, además de `name="texto"` y `href="url"` son:

- `hreflang="codigo"` Indica el idioma y la variación idiomática en función del país del recurso enlazado que se establece a través de unos códigos estandarizados de dos letras. Algunos de ellos son:

---

en

Inglés

|       |                     |
|-------|---------------------|
| en-AU | Inglés - Australia  |
| en-US | Inglés - EE.UU.     |
| es    | Español             |
| es-AR | Español - Argentina |
| es-ES | Español - España    |
| es-MX | Español - México    |

La lista completa de códigos se define en el estándar ISO 639 (<http://xml.coverpages.org/iso639a.html>) .

- `type="tipo"` Da información al navegador acerca del tipo de contenido que se enlaza. Se indica mediante una cadena de texto cuyos posibles valores también están estandarizados. Algunos de los más utilizados son "text/html" (páginas HTML), "image/png" (imágenes con formato PNG) o "text/css" (hojas de estilo CSS).

La lista completa de tipos de contenido se define en los estándares RFC 2045 y RFC 2046 (<http://www.iana.org/assignments/media-types>) .

Los atributos `rel` y `rev` describen la relación que la página actual tiene con la página a la que se enlaza.

- `rel="tipo"` Describe la relación del documento actual con el recurso enlazado.
- `rev="tipo"` Describe la relación del recurso enlazado con el documento actual.

Los tipos de relación pueden ser muy variados, como por ejemplo `alternate` (indica que es una versión alternativa al documento actual), `stylesheet` (indica que se ha enlazado una hoja de estilos) o `start` (indica que se trata del primer documento de una colección de documentos).

La lista completa de tipos de relaciones se define en la especificación oficial de HTML (<http://www.w3.org/TR/1999/REC-html401-19991224/types.html#type-links>) .

- `charset="tipo"` Describe la codificación del recurso enlazado.

Los valores que se pueden utilizar también están estandarizados y las codificaciones más utilizadas son UTF-8 y ISO-8859-1, aunque existen decenas de códigos definidos.

Aquí encontramos una lista completa de codificaciones (<http://www.iana.org/assignments/character-sets/character-sets.xml>) .

Un ejemplo de código HTML con un enlace avanzado sería:

```
<a href="http://www.google.com" hreflang="en"
    type="text/html" charset="UTF-8">Página principal de Google</a>
```

Además de los **enlaces** creados por la etiqueta `<a>`, las **páginas HTML** pueden incluir otro tipo de enlaces que cargan los recursos automáticamente.

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.

| <script>                 |  |
|--------------------------|--|
| <b>Atributos comunes</b> | -  |
| <b>Atributos propios</b> | <p><code>src="url"</code> Indica la dirección del archivo que contiene el código. La URL puede ser absoluta o relativa y externa o interna</p> <p><code>type="tipo"</code> Permite avisar al navegador sobre el tipo de código que se incluye (normalmente JavaScript, expresado como "text/javascript")</p> <p><code>defer="defer"</code> El código no va a modificar el contenido de la página web</p> <p><code>charset="tipo"</code> Describe la codificación del código enlazado</p> |
| <b>Tipo de elemento</b>  | En línea, en bloque y etiqueta vacía   |

```
<script>
```

|                    |   |
|--------------------|---|
| <b>Descripción</b> | Se emplea para enlazar o definir un bloque de código (normalmente JavaScript) |
|--------------------|---|

Este *tag* permite enlazar código de **varios lenguajes de programación**, aunque su uso más habitual consiste tanto en insertar un **bloque de código JavaScript** en la página como en enlazar un **archivo JavaScript externo**. Normalmente se incluye dentro de la cabecera (`<head>...</head>`), aunque puede aparecer en cualquier parte del documento HTML.

Dos ejemplos de **código html** con la etiqueta `<script>` serían:

- Enlazar un archivo **JavaScript** externo:

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

- Incluir en la página web un bloque de código **JavaScript**: Este bloque de código se debe encerrar entre `<![CDATA[ y ]]>`, de esta forma, se pueden construir páginas **HTML** válidas y código **JavaScript** correcto.

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

La etiqueta `<link>`, sin embargo, sólo se puede incluir dentro de la cabecera (`<head>...</head>`) del documento HTML. Su uso habitual es el de enlazar las hojas de estilos **CSS** utilizadas por las páginas web.

|                          |  |
|--------------------------|--|
|                          | <link>   |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos   |
| <b>Atributos propios</b> | charset, href, hreflang, type, rel y rev - Todos ellos con el mismo significado que para la etiqueta <a><br>media="tipo" Indica el medio para el que debe aplicarse la relación. Los medios disponibles también están estandarizados |
| <b>Tipo de elemento</b>  | Etiqueta vacía   |
| <b>Descripción</b>       | Se emplea para enlazar y establecer relaciones entre el documento y otros recursos   |

En cuanto al atributo media, como ya hemos mencionado, estos medios están estandarizados. Sin embargo, existen algunos más utilizados como son:

- screen para los contenidos mostrados en pantalla.
- print para las impresoras.
- handheld para los dispositivos móviles.

Un ejemplo de **código html** con la etiqueta <link> para enlazar hojas de estilo CSS sería:

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

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

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

Al pinchar sobre el enlace anterior, se abre automáticamente el **programa de correo electrónico** por defecto del ordenador del usuario. Además, el protocolo *mailto* permite el paso de parámetros por método GET. De esta manera podemos mandar un email especificando el asunto y el mensaje:

```
<a href="mailto:nombre@direccion.com?body=Me gustaría conocer más acerca  
de su negocio&subject=Info">Solicita más información</a>
```

Los parametros que reconocerá el programa de correo son:

- subject Asunto del mensaje
- body Cuerpo del mensaje
- cc Direccion de copia
- bcc Direccion de copia oculta

Aunque el uso de *mailto* puede parecer una ventaja, su uso está desaconsejado. La forma más correcta 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.

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

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

```
<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>
```

```
<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>
```

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

```
<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>
```

## Capítulo 6

Las listas son otro de los elementos **HTML** más comunes. Este lenguaje incorpora unas listas con viñetas sencillas o también letras o números. Dentro de una lista se puede incluir cualquiera de los elementos HTML, y éstas a su vez pueden incluirse dentro de formularios, tablas, etc.

Las listas ofrecen la posibilidad de presentar información de una manera útil, simple y fácilmente comprensible. No sólo para ordenarla sino también para jerarquizarla, o numerarla.

El **lenguaje HTML** define **tres tipos** diferentes de listas para agrupar los elementos: **listas no ordenadas** (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). Además, podemos anidar varias listas a otras y de diferentes tipos sin restricciones.

Estas listas son las más utilizadas. Se componen de elementos relacionados entre sí pero para los que no se indica un orden. Las **listas no ordenadas** van dentro de las etiquetas `<ul>...</ul>` y cada punto que queramos añadir dentro de las etiquetas `<li>...</li>`.

|                          |  |
|--------------------------|--|
|                          | <code>&lt;ul&gt;</code>                |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos |

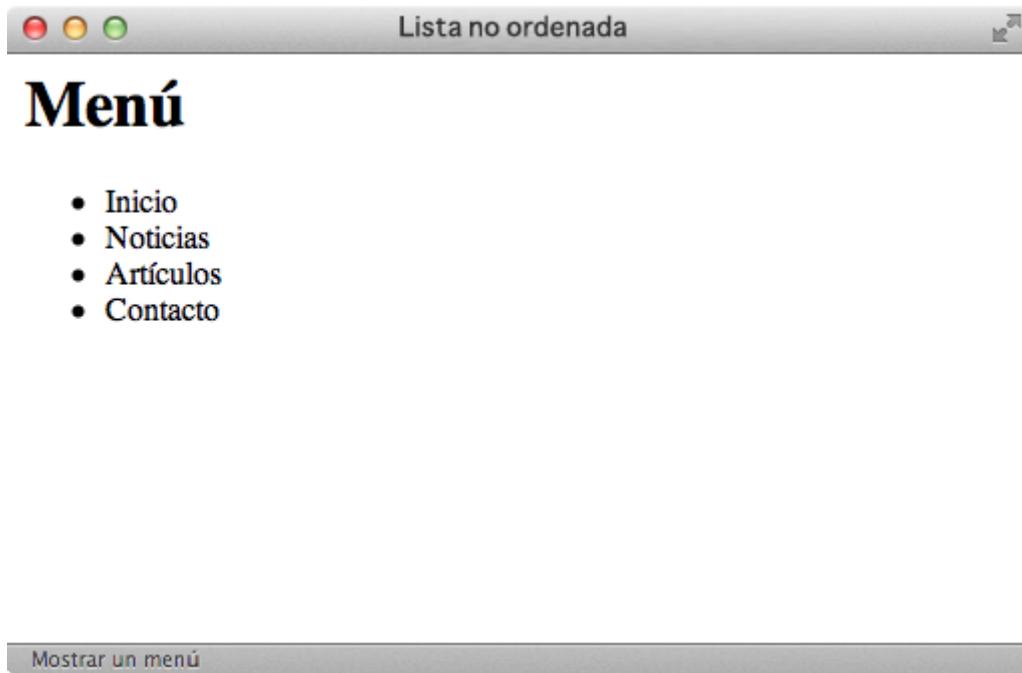
|                          |   |
|--------------------------|---|
|                          | <ul>  |
| <b>Atributos propios</b> | -   |
| <b>Tipo de elemento</b>  | En bloque   |
| <b>Descripción</b>       | Se emplea para definir listas no ordenadas                                    |
|                          | <li>  |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | -   |
| <b>Tipo de elemento</b>  | En bloque   |
| <b>Descripción</b>       | Se emplea para definir los elementos de las listas (ordenadas y no ordenadas) |

El navegador por defecto muestra los elementos de la lista tabulados y con una pequeña **viñeta** formada por un círculo negro. Esto puede cambiarse mediante el atributo `type`, el cual indica cómo debe ser representado la viñeta o numeración en una lista, aunque su uso está desaconsejado. La mejor opción es la modificándolo con las hojas de estilo **CSS**.

Un ejemplo de **código HTML** con las etiquetas `<ul>` y `<li>` sería:

```
<html>
  <head>
    <title>Lista no ordenada</title>
  </head>
  <body>
    <h1>Menú</h1>
    <ul>
      <li>Inicio</li>
      <li>Noticias</li>
      <li>Artículos</li>
      <li>Contacto</li>
    </ul>
  </body>
</html>
```

Que un navegador visualizaría de esta manera:



**Figura 6.1** Ejemplo lista no ordenada

Estas listas son iguales que las anteriores, salvo por que en este caso los elementos relacionados se muestran siguiendo **un orden determinado**.

El uso de esta lista es el más adecuado cuando existe una mayor **importancia en el orden** de los elementos (índice de un libro, instrucciones, etc.), ya que los símbolos que preceden a los elementos serán números y éstos se irán generando automáticamente por orden.

Las **listas ordenadas** van dentro de las etiquetas `<ol>...</ol>` y cada punto que queramos añadir dentro de las etiquetas `<li>...</li>`.

|                          |   |
|--------------------------|---|
|                          | <code>&lt;ol&gt;</code>                 |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | -                                       |
| <b>Tipo de elemento</b>  | En bloque                               |
| <b>Descripción</b>       | Se emplea para definir listas ordenadas |

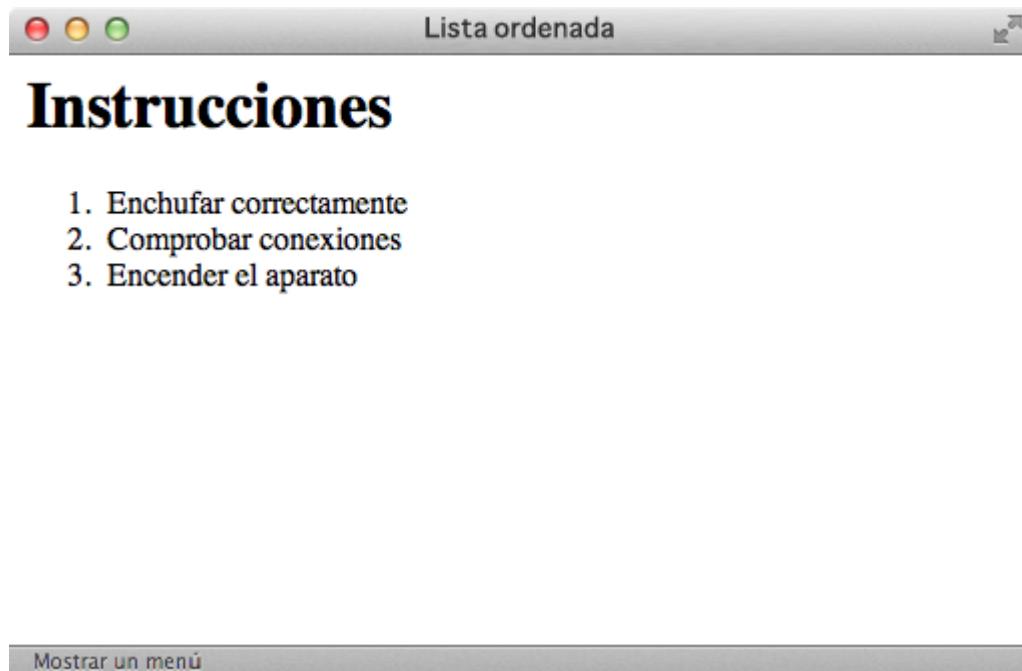
De la misma manera que en las listas no ordenadas, el navegador por defecto muestra los elementos de la lista tabulados y con una consecución de numeración arábiga. Esto puede cambiarse mediante el atributo

type pero, como ya hemos comentado, no sería correcto; se debe modificar con las hojas de estilo **CSS**. También se puede modificar el primer número con el que queremos que comience la lista (si nos interesa que comience con un número distinto al 1).

Un ejemplo de **código HTML** con las etiquetas <ol> y <li> sería:

```
<html>
  <head>
    <title>Lista ordenada</title>
  </head>
  <body>
    <h1>Instrucciones</h1>
    <ol>
      <li>Enchufar correctamente</li>
      <li>Comprobar conexiones</li>
      <li>Encender el aparato</li>
    </ol>
  </body>
</html>
```

Que un navegador visualizaría de esta manera:



**Figura 6.2** Ejemplo lista ordenada

Estas **listas** son las menos utilizadas, su funcionamiento es similar al de un **diccionario**, ya que cada elemento de la lista está formado por términos y definiciones.

Éstas se componen de tres etiquetas:

- <dl> “definiton list” - entre ella y su cierre va a ir una definición.
- <dt> “definition term” - entre ella y su cierre irá el término que vamos a definir.
- <dd> “definition description” - entre ella y su cierre irá la definición en cuestión.

| <dl>                     |   |
|--------------------------|---|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos      |
| <b>Atributos propios</b> | -   |
| <b>Tipo de elemento</b>  | En bloque                                   |
| <b>Descripción</b>       | Se emplea para definir listas de definición |

| <dt>                     |   |
|--------------------------|---|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | -   |
| <b>Tipo de elemento</b>  | En bloque   |
| <b>Descripción</b>       | Se emplea para definir los términos de los elementos de una lista de definición |

| <dd>                     |  |
|--------------------------|--|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos |
| <b>Atributos propios</b> | -                                      |

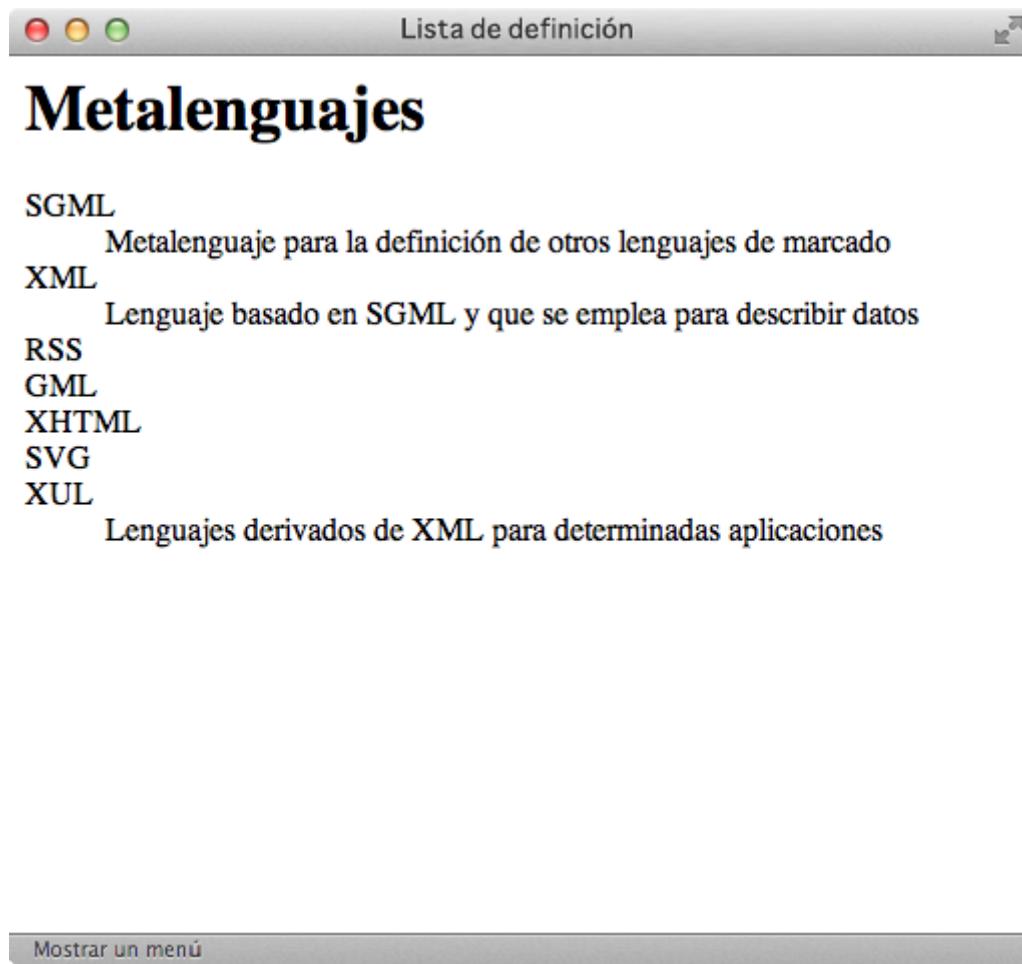
|                         |   |
|-------------------------|---|
|                         | <dd>  |
| <b>Tipo de elemento</b> | En bloque   |
| <b>Descripción</b>      | Se emplea para indicar las definiciones de los elementos de una lista de definición |

Aunque no es habitual, cada término puede tener asociada más de una definición y cada definición puede tener asociados varios términos.

Un ejemplo de **código HTML** con estas tres etiquetas sería:

```
<html>
  <head>
    <title>Lista de definición</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>
```

Que un navegador visualizaría de esta manera:



**Figura 6.3** Ejemplo lista definiciones

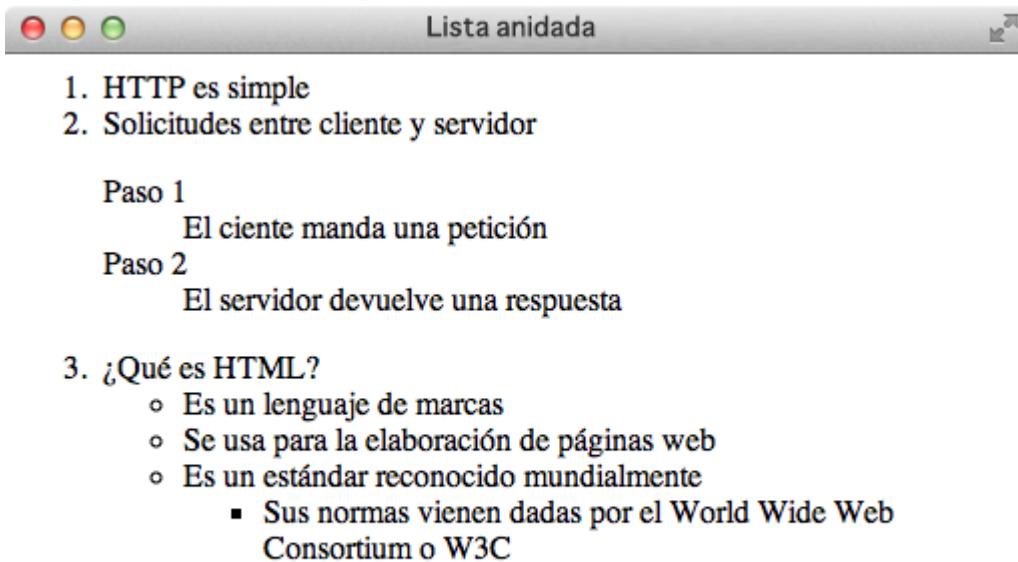
Como ya hemos mencionado, podemos anidar varias listas a otras y crear así listas compuestas a nuestro gusto.

Un ejemplo de **código HTML** de una lista anidada sería:

```
<html>
  <head>
    <title>Lista anidada</title>
  </head>
  <body>
    <ol>
      <li>HTTP es simple</li>
      <li>Solicitudes entre cliente y servidor
        <ol>
          <li>Paso 1</li>
          <li>El cliente manda una petición</li>
        </ol>
        <li>Paso 2</li>
        <li>El servidor devuelve una respuesta</li>
      </ol>
    </li>
  </ol>
</body>
</html>
```

```
</dl>
</li>
<li>¿Qué es HTML?
    <ul>
        <li>Es un lenguaje de marcas</li>
        <li>Se usa para la elaboración de páginas web</li>
        <li>Es un estándar reconocido mundialmente
            <ul>
                <li>Sus normas vienen dadas por el
                    World Wide Web Consortium o W3C</li>
            </ul>
        </li>
    </ul>
</li>
</ol>
</body>
</html>
```

Que un navegador visualizaría de esta manera:



**Figura 6.4** Ejemplo lista anidada

## Capítulo 7

Las imágenes son **uno de los elementos más importantes** de las páginas web. Podríamos decir que existen dos tipos de imagen, las imágenes de **contenido** de una web, que proporcionan y complementan la información y otras imágenes (pequeños iconos en listas, fondos de página, etc.) que aunque también se pueden incluir en el código HTML mediante la etiqueta `<img>` que explicaremos a continuación, es más correcto emplear hojas de estilo **CSS** para mostrarlas.

La etiqueta `<img>` tiene varios atributos:

| <code>&lt;img&gt;</code> |  |
|--------------------------|--|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos   |
| <b>Atributos propios</b> | <code>src="url"</code> Indica la URL de la imagen que se muestra.<br><code>alt="texto"</code> Descripción corta de la imagen<br><code>longdesc="url"</code> Indica una URL en la que puede encontrarse una descripción más detallada de la imagen. Se emplea con las imágenes complejas que necesitan mucha información para ser descritas<br><code>name="texto"</code> Nombre del elemento imagen<br><code>height="unidad_de_medida"</code> Indica la altura con la que se debe mostrar la imagen (no es obligatorio que coincida |

|                         |   |
|-------------------------|---|
|                         | <img>   |
|                         | con la altura original de la imagen)<br>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) |
| <b>Tipo de elemento</b> | En línea y etiqueta vacía   |
| <b>Descripción</b>      | Se emplea para incluir imágenes en los documentos   |

De todos estos atributos, sólamente hay dos requeridos, que son `src` y `alt`, que proporcionan la información suficiente para establecer la URL de la imagen y describir su contenido mediante un texto breve (menos de 1024 caracteres).

Aunque `<img>` es una etiqueta vacía (por lo que no necesita etiqueta de cierre), lo más correcto (y válido para una página XHTML) sería incluir `/>` al final de ésta, ya que de esta manera se considera una etiqueta cerrada.

En teoría no existe ninguna restricción sobre el **formato gráfico** de la imagen. No obstante, 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.

Un ejemplo sencillo para incluir una imagen sería:

```
| 
```

En este ejemplo, encontramos que la imagen `logotipo.gif` se encuentra en el directorio (`/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. Como ya hemos comentado, HTML estructura de forma correcta los contenidos de la página y CSS define el

aspecto gráfico con el que se muestran los contenidos. Por esta razón, 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 **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.

```

>
```

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. Si solamente se establece la altura o anchura de la imagen, el navegador calcula la el otro valor necesario para que se mantenga la proporción de la imagen.

Aunque el uso de los **mapas de imagen** se ha reducido mucho en los últimos años, aún se utilizan en algunos sitios especializados. 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**. Una 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.

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>`.

| <map>                    |   |
|--------------------------|---|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | <code>name="texto"</code> Nombre que identifica de forma única al mapa definido (es obligatorio indicar un nombre único)  |
| <b>Tipo de elemento</b>  | En bloque y en línea  |
| <b>Descripción</b>       | Se emplea para definir mapas de imagen  |
| <area>                   |   |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos y foco   |
| <b>Atributos propios</b> | <code>href="url"</code> URL a la que se accede al pinchar sobre el área<br><code>nohref="nohref"</code> Se emplea para las áreas que no son seleccionables<br><code>shape="default   rect   circle   poly"</code> Indica el tipo de |

|                         |  |
|-------------------------|--|
|                         | <p>&lt;area&gt;</p> <p>área que se define (toda la imagen, rectangular, circular o poligonal)</p> <p>coords="lista de números" Se trata de una lista de números separados por comas que representan las coordenadas del área.</p> <p>Rectangular = X1,Y1,X2,Y2 (coordenadas X e Y del vértice superior izquierdo y coordenadas X e Y del vértice inferior derecho)</p> <p>Circular = X1,Y1,R (coordenadas X e Y del centro y radio del círculo)</p> <p>Poligonal = 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)</p> |
| <b>Tipo de elemento</b> | Etiqueta vacía   |
| <b>Descripción</b>      | Se emplea para definir las distintas áreas que forman un mapa de imagen  |

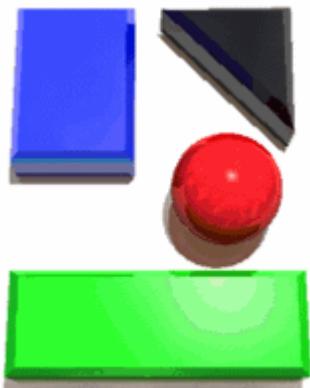
Si una imagen utiliza un mapa de imagen, se debe indicar mediante el atributo `usemap`. El valor del atributo debe ser el nombre del mapa de imagen definido en otra parte del mismo documento HTML:

```

...
<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 7.1** Mapa imagen

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

```

<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>
```

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:

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

Los posibles valores de type están estandarizados y coinciden con los del atributo type de la etiqueta <a> que explicamos 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>`:

| <code>&lt;param&gt;</code> |  |
|----------------------------|--|
| <b>Atributos comunes</b>   | <code>id</code>  |
| <b>Atributos propios</b>   | <code>name="texto"</code> Indica el nombre del parámetro<br><code>value="texto"</code> Indica el valor del parámetro |
| <b>Tipo de elemento</b>    | Etiqueta vacía   |
| <b>Descripción</b>         | 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>`. Pero 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.

| <embed>                  |  |
|--------------------------|--|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos   |
| <b>Atributos propios</b> | <p>src="url" Indica la URL del archivo u objeto que se incluye en la página</p> <p>type="tipo_de_contenido" Indica el tipo de contenido del objeto (<i>flash</i>, quicktime, java, etc.)</p> <p>height="unidad_de_medida" Indica la altura con la que se debe mostrar el objeto</p> <p>width="unidad_de_medida" Indica la anchura con la que se debe mostrar el objeto</p> |
| <b>Tipo de elemento</b>  | En bloque  |
| <b>Descripción</b>       | 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.

Esta página se ha dejado vacía a propósito

## Capítulo 8

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

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

**Figura 8.1** Partes que componen una tabla compleja

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 algún tiempo, las tablas también se utilizaban para definir la estructura de las páginas web y aunque hoy en día hay diseñadores que siguen utilizando este método, se trata de una técnica poco recomendable ya que se complica en exceso el código HTML. La solución correcta para definir la **estructura de las páginas** consiste en la utilización de hojas de estilos **CSS**.

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.

Un ejemplo de **código HTML** con una tabla sencilla sería:

```
<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>
```

Y un navegador lo visualizaría de esta manera:

The screenshot shows a web browser window with a title bar reading "Ejemplo de tabla sencilla". Below the title bar is a menu bar with "Mostrar un menú" highlighted. The main content area displays a heading "Listado de cursos" and a table with three rows. The table has two columns: "Curso" and "Horas Horario". The data rows are: "CSS 20 16:00 - 20:00", "HTML 20 16:00 - 20:00", and "Dreamweaver 60 16:00 - 20:00".

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

**Figura 8.2** 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>` (*table row*) definen cada fila de la tabla y encierran todas las columnas. Por último, la etiqueta `<td>` (*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>`.

| <table>                  |   |
|--------------------------|---|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | summary="texto" Permite describir el contenido de la tabla (lo utilizan los buscadores y las personas discapacitadas) |
| <b>Tipo de elemento</b>  | En bloque   |

|                          |  |
|--------------------------|--|
|                          | <table>  |
| <b>Descripción</b>       | Se emplea para definir tablas de datos   |
|                          | <tr>   |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos   |
| <b>Atributos propios</b> | -  |
| <b>Tipo de elemento</b>  | En bloque  |
| <b>Descripción</b>       | Se emplea para definir cada fila de las tablas de datos  |
|                          | <td>   |
| <b>Atributos comunes</b> | básicos, internacionalización y eventos  |
| <b>Atributos propios</b> | <p>abbr="texto" Permite describir el contenido de la celda (empleado sobre todo en los navegadores utilizados por personas discapacitadas)</p> <p>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</p> <p>scope="col, row, colgroup, rowgroup" Indica las celdas para las que esta celda será su cabecera</p> <p>colspan="numero" Número de columnas que ocupa esta celda</p> <p>rowspan="numero" Número de filas que ocupa esta celda</p> |
| <b>Tipo de elemento</b>  | En bloque  |
| <b>Descripción</b>       | 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, sólo se utiliza de forma habitual el atributo scope, sobre todo con las celdas de cabecera que se ven a continuación.

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

| <th>                     |   |
|--------------------------|---|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | abbr="texto" Permite describir el contenido de la celda (empleado sobre todo en los navegadores utilizados por personas discapacitadas)<br>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<br>scope="col, row, colgroup, rowgroup" Indica las celdas para las que esta celda será su cabecera<br>colspan="numero" Número de columnas que ocupa esta celda<br>rowspan="numero" Número de filas que ocupa esta celda |
| <b>Tipo de elemento</b>  | En bloque   |
| <b>Descripción</b>       | 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 sólo puede incluir una etiqueta `<caption>`.

| <code>&lt;caption&gt;</code> |   |
|------------------------------|---|
| <b>Atributos comunes</b>     | básicos, internacionalización, eventos                  |
| <b>Atributos propios</b>     | -   |
| <b>Tipo de elemento</b>      | En línea  |
| <b>Descripción</b>           | 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.

Un ejemplo de **código HTML** y una tabla compleja que ha fusionado dos columnas simples para formar una columna más ancha sería:

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

Y un navegador lo visualizaría de esta manera:

|   |   |
|---|---|
| A |   |
| B | C |
| D |   |

Mostrar un menú

### Figura 8.3 Ejemplo sencillo de fusión de columnas

La primera fila de la tabla está formada sólo 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**. 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.

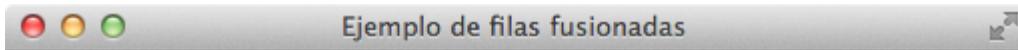
En el ejemplo anterior, la celda de la primera fila debe ocupar el espacio de dos columnas simples, por lo que el código HTML debe ser `<td colspan="2">A</td>`.

Un ejemplo de **código HTML** y na tabla HTML que fusiona filas sería:

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

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

Y un navegador lo visualizaría de esta manera:



## Fusión de filas

|   |   |
|---|---|
| A | B |
| C |   |

Mostrar un menú

**Figura 8.4** Ejemplo sencillo de fusión de filas

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:

The screenshot shows a web browser window with a title bar "Ejemplo de columnas fusionadas". The main content area displays a table with the following structure:

|   |   |   |   |
|---|---|---|---|
| A |   | B |   |
| C | D | E |   |
| F |   |   |   |
| G | H | I | J |

A menu bar at the bottom has an item "Mostrar un menú".

**Figura 8.5** Ejemplo complejo de fusión de columnas

El **código HTML** necesario para fusionar las columnas de la tabla anterior sería:

```
<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:

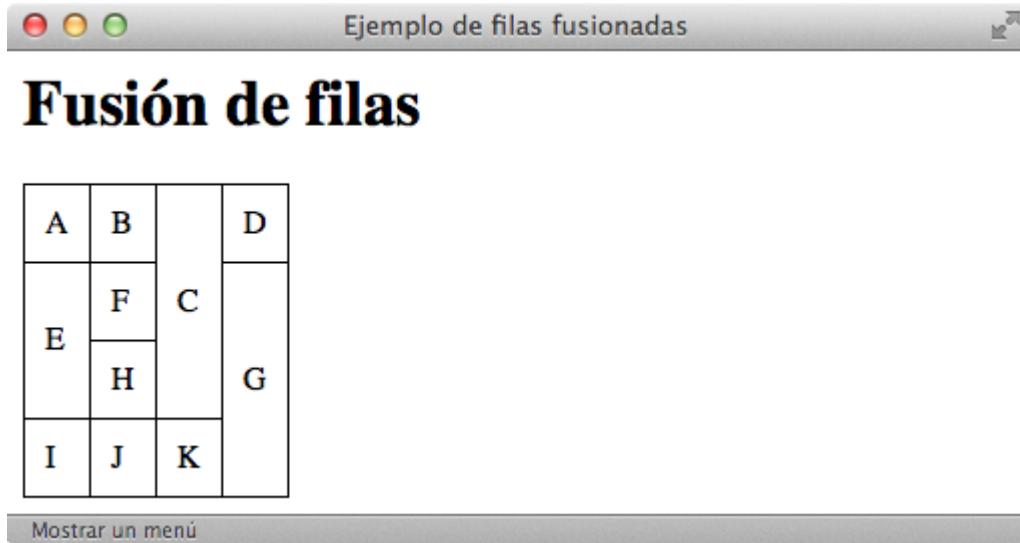


Figura 8.6 Ejemplo complejo de fusión de filas

El **código HTML** necesario para fusionar las filas de la tabla anterior sería:

```

<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>
      </tr>
      <tr>
        <td>G</td>
      </tr>
      <tr>
        <td>H</td>
        <td>I</td>
        <td>J</td>
        <td>K</td>
      </tr>
      <tr>
        <td>L</td>
        <td>M</td>
        <td>N</td>
        <td>O</td>
      </tr>
    </table>
  </body>
</html>

```

```
<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>
```

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:

|  | 2008           |                |                |                 |
|--|----------------|----------------|----------------|-----------------|
|  | March          | June           | September      | December        |
| <b>Non-current assets</b>  | <b>87.249</b>  | <b>87.126</b>  | <b>90.426</b>  | <b>91.269</b>   |
| 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           |
| <b>Current assets</b>  | <b>18.042</b>  | <b>17.979</b>  | <b>19.128</b>  | <b>17.713</b>   |
| 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               |
| <b>Total Assets = Total Equity and Liabilities</b>                         | <b>105.291</b> | <b>105.106</b> | <b>109.554</b> | <b>108.982</b>  |
| <b>Equity</b>  | <b>15.714</b>  | <b>15.072</b>  | <b>19.185</b>  | <b>20.001</b>   |
| 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           |
| <b>Non-current liabilities</b>   | <b>54.053</b>  | <b>66.406</b>  | <b>63.908</b>  | <b>62.644,0</b> |
| 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             |
| <b>Current liabilities</b>   | <b>35.523</b>  | <b>23.628</b>  | <b>26.462</b>  | <b>26.337,0</b> |
| 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               |
| <b>Financial Data</b>  |                |                |                |                 |
| Net Financial Debt (1)   | 53.510         | 54.922         | 52.239         | 52.145          |

**Figura 8.7** 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>`.

|                          | <thead> <tbody> <tfoot>   |
|--------------------------|---|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | -   |
| <b>Tipo de elemento</b>  | En bloque   |
| <b>Descripción</b>       | Se emplean para agrupar varias filas en una cabecera ( <code>thead</code> ) un pie ( <code>tfoot</code> ) o una sección ( <code>tbody</code> ) 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>`.

Un ejemplo de **código HTML** con las etiquetas `<thead>`, `<tbody>` y `<tfoot>` sería:

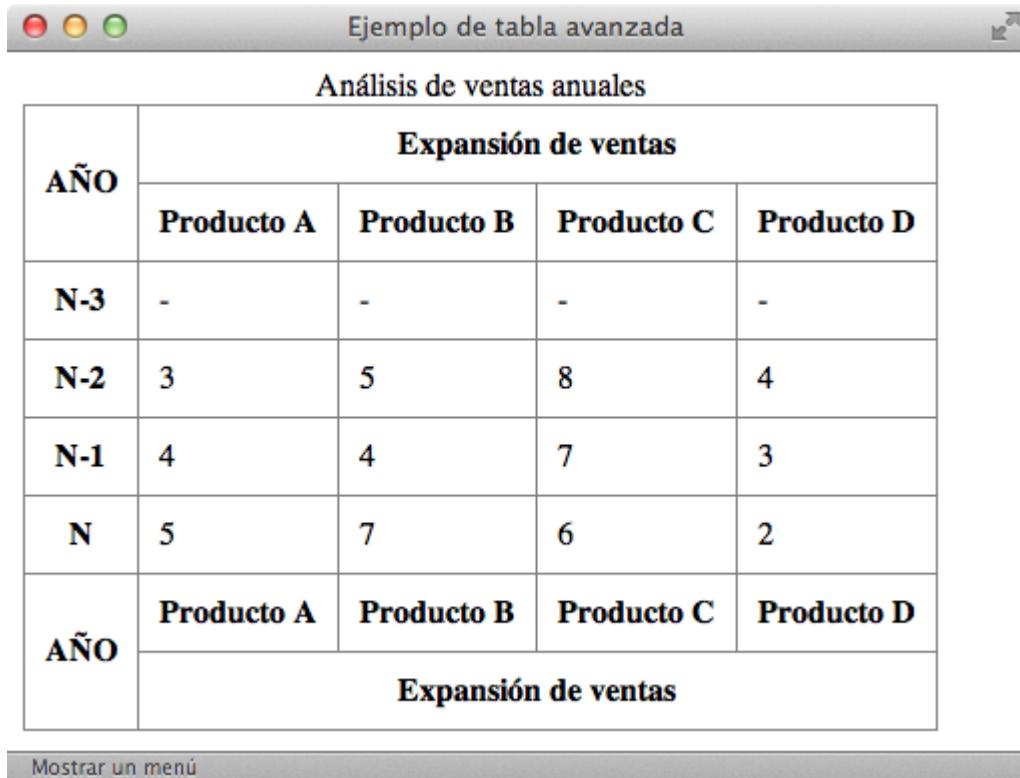
```
<html>
  <head>
    <title>Ejemplo de tabla avanzada</title>
  </head>
  <body>
    <h3>Análisis de ventas</h3>
    <table summary="Análisis de ventas anuales">
      <caption>Análisis de ventas anuales</caption>
      <thead>
        <tr>
          <th rowspan="2" scope="col">AÑO</th>
          <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>
      </tbody>
    </table>
  </body>
</html>
```

```

</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>

```

Y un navegador lo visualizaría de esta manera:



The screenshot shows a web browser window with a title bar reading "Ejemplo de tabla avanzada". The main content area displays a table titled "Análisis de ventas anuales". The table has a complex structure with multiple sections and rows. The columns represent products (A, B, C, D) and the rows represent years (N-3, N-2, N-1, N). The data values are: N-3: All cells are empty. N-2: Product A: 3, Product B: 5, Product C: 8, Product D: 4. N-1: Product A: 4, Product B: 4, Product C: 7, Product D: 3. N: Product A: 5, Product B: 7, Product C: 6, Product D: 2.

| AÑO                 | Expansión de ventas |            |            |            |
|---------------------|---------------------|------------|------------|------------|
|                     | 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          |
| AÑO                 | Producto A          | Producto B | Producto C | Producto D |
| Expansión de ventas |                     |            |            |            |

**Figura 8.8** Ejemplo de tabla avanzada con cabecera, pie y secciones

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.

Esta 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 sólo asigna atributos comunes a varias columnas.

Un ejemplo de **código HTML** con la etiqueta <col> sería:

```
<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>
```

Y un navegador lo visualizaría de esta manera:

Ejemplo de tabla avanzada

Análisis de ventas anuales

| AÑO | Expansión de ventas |            |            |            |
|-----|---------------------|------------|------------|------------|
|     | 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          |
| AÑO | Producto A          | Producto B | Producto C | Producto D |
|     | Expansión de ventas |            |            |            |

Figura 8.9 Ejemplo de tabla avanzada que usa la etiqueta col

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.

Un ejemplo de **código HTML** con la etiqueta `<colgroup>` sería:

```
<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>

```

Y un navegador lo visualizaría de esta manera:

Ejemplo de tabla avanzada

Análisis de ventas anuales

| 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          |

Figura 8.10 Ejemplo de tabla avanzada que usa la etiqueta colgroup

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

Esta página se ha dejado vacía a propósito

## Capítulo 9

Como ya hemos mencionado en numerosas ocasiones, **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.

**HTML/XHTML** incluye los suficientes elementos de formulario para crear desde los formularios sencillos que utilizan los buscadores hasta los formularios complejos de las aplicaciones más avanzadas.

Los formularios más sencillos se pueden crear utilizando solamente dos etiquetas: <form> y <input>.

Un ejemplo de **código HTML** con ambas etiquetas sería:

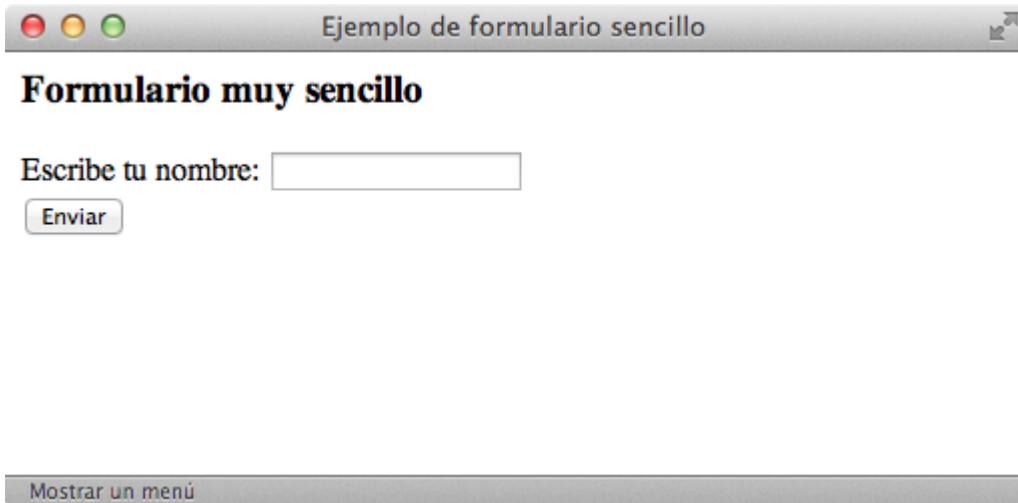
```
<html>
  <head>
    <title>Ejemplo de formulario sencillo</title>
  </head>
  <body>
    <h3>Formulario muy sencillo</h3>
    <form action="http://www.enlace.es/formulario.php" method="post">
      Escribe tu nombre:
      <input type="text" name="nombre" value="" />
      <br/>
```

```

<input type="submit" value="Enviar" />
</form>
</body>
</html>

```

Y un navegador lo visualizaría de esta manera:



**Figura 9.1** Formulario sencillo definido con las etiquetas form e input

La etiqueta `<form>` encierra todos los contenidos del formulario (botones, cuadros de texto, listas desplegables) y la etiqueta `<input>` permite definir varios tipos diferentes de elementos (botones y cuadros de texto).

|                          |  |
|--------------------------|--|
|                          | <code>&lt;form&gt;</code>  |
| <b>Atributos comunes</b> | <code>action="url"</code> Indica la URL que se encarga de procesar los datos del formulario<br><code>method="POST o GET"</code> Método HTTP empleado al enviar el formulario<br><code>enctype="application/x-www-form-urlencoded o multipart/form-data"</code> Tipo de codificación empleada al enviar el formulario al servidor (sólo se indica de forma explícita en los formularios que permiten adjuntar archivos)<br><code>accept="tipo"</code> Lista separada por comas de todos los tipos de archivos aceptados por el servidor (sólo para los formularios que permiten adjuntar archivos)<br>Otros: <code>accept-charset</code> , <code>onsubmit</code> , <code>onreset</code> |
| <b>Atributos propios</b> | -  |

|                         |  |
|-------------------------|--|
|                         | <form>   |
| <b>Tipo de elemento</b> | En bloque  |
| <b>Descripción</b>      | Se emplea para insertar un formulario en la página |

La mayoría de formularios utilizan sólo los atributos `action` y `method`. El atributo `action` indica la URL de la aplicación del servidor que se encarga de **procesar los datos** introducidos por los usuarios. Esta aplicación también se encarga de generar la **respuesta** que muestra el navegador.

El atributo `method` establece la **forma en la que se envian los datos** del formulario al servidor. Este atributo hace referencia al **método HTTP**, por lo que no es algo propio de HTML. Los dos valores que se utilizan en los formularios son GET y POST. De esta forma, casi todos los formularios incluyen el atributo `method="get"` o el atributo `method="post"`.

Al margen de otras diferencias técnicas, el método POST permite el envío de mucha más información que el método GET. En general, el método GET admite como máximo el envío de unos 500 bytes de información. La otra gran limitación del método GET es que no permite el envío de archivos adjuntos con el formulario. Además, los datos enviados mediante GET se ven en la barra de direcciones del navegador (se añaden al final de la URL de la página), mientras que los datos enviados mediante POST no se pueden ver tan fácilmente.

Si no sabes qué método elegir para un formulario, existe una regla general que dice que el **método GET** se debe utilizar en los **formularios que no modifican la información** (por ejemplo en un formulario de búsqueda). Por su parte, el **método POST** se debería utilizar cuando el **formulario modifica la información original** (insertar, modificar o borrar alguna información).

El ejemplo más común de formulario con método GET es el de los buscadores. Si realizas una búsqueda con tu buscador favorito, verás que las palabras que has introducido en tu búsqueda aparecen como parte de la URL de la página de resultados.

Del resto de atributos de la etiqueta `<form>`, el único que se utiliza ocasionalmente es `enctype`. Como se explica más adelante, este atributo es imprescindible en los formularios que permiten adjuntar archivos.

**Los elementos de formulario** como botones y cuadros de texto también se denominan "campos de formulario" y "controles de formulario". La mayoría de controles se crean con la etiqueta <input>, por lo que su definición formal y su lista de atributos es muy extensa:

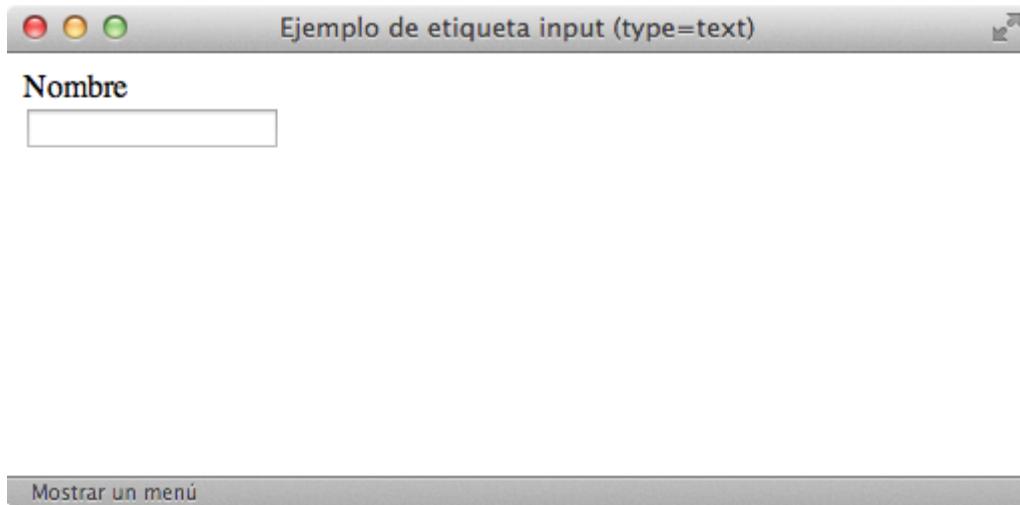
|                          |  |
|--------------------------|--|
|                          | <input>  |
| <b>Atributos comunes</b> | <p>type="text   password   checkbox   radio   submit   reset   file   hidden   image   button" Indica el tipo de control que se incluye en el formulario</p> <p>name="texto" Asigna un nombre al control (es imprescindible para que el servidor pueda procesar el formulario)</p> <p>value="texto" Valor inicial del control</p> <p>size="unidad" Tamaño inicial del control (para los campos de texto y de password se refiere al número de caracteres, en el resto de controles se refiere a su tamaño en píxel)</p> <p>maxlength="numero" Máximo número de caracteres para los controles de texto y de password</p> <p>checked="checked" Para los controles checkbox y radiobutton permite indicar qué opción aparece preseleccionada</p> <p>disabled="disabled" El control aparece deshabilitado y su valor no se envía al servidor junto con el resto de datos</p> <p>readonly="readonly" El contenido del control no se puede modificar</p> <p>src="url" Para el control que permite crear botones con imágenes, indica la URL de la imagen que se emplea como botón de formulario</p> <p>alt="texto" Descripción del control</p> |
| <b>Atributos propios</b> | -  |
| <b>Tipo de elemento</b>  | En línea y etiqueta vacía  |

<input>

**Descripción** Se emplean para insertar un control en un formulario

A continuación se muestran ejemplos para los diez controles que se pueden crear con la etiqueta <input>.

Se trata del elemento más utilizado en los formularios. En el caso más sencillo, se muestra un cuadro de texto vacío en el que el usuario puede escribir cualquier texto:



**Figura 9.2 Ejemplo de etiqueta input (type=text)**

Nombre <br/>  
<input type="text" name="nombre" value="" />

**El atributo** type diferencia a cada uno de los diez controles que se pueden crear con la etiqueta <input>. Para los cuadros de texto, su valor es text. El atributo name es el más importante en los campos del formulario. De hecho, si un campo no incluye el atributo name, sus datos no se envían al servidor. El valor que se indica en el atributo name es el nombre que utiliza la aplicación del servidor para obtener el valor de este campo de formulario.

Cuando el usuario pulsa el **botón de envío** del formulario, el navegador envía los datos a una aplicación del servidor para que procese la información y genere una **respuesta adecuada**. En el servidor, la aplicación que procesa los datos debe obtener en primer lugar toda la información introducida por el usuario. Para ello, utiliza el valor del atributo name para obtener los datos de cada control del formulario.

Como el valor del atributo `name` se utiliza en aplicaciones programadas, es esencial ponerse de acuerdo con el programador de la aplicación, no se debe modificar su valor sin modificar la aplicación y no se deben utilizar caracteres problemáticos en programación (espacios en blanco, acentos y caracteres como ñ o ç).

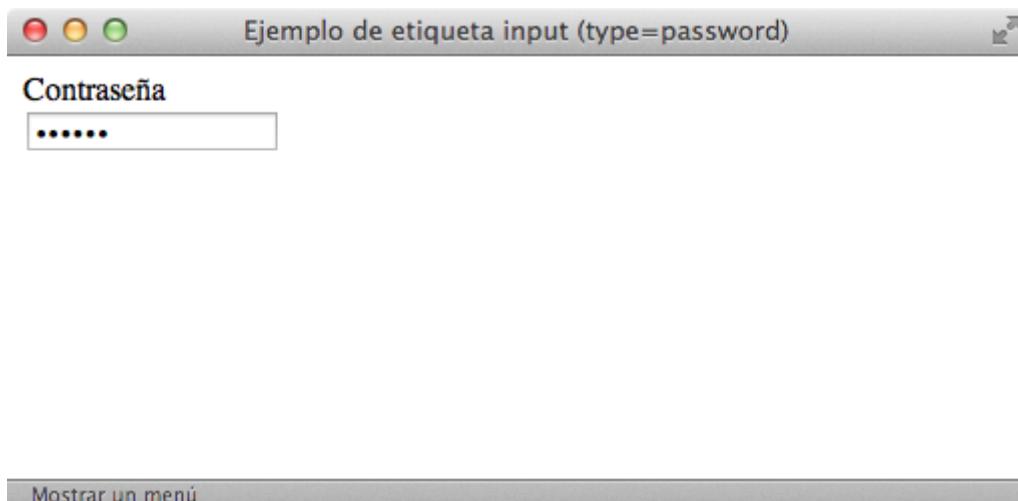
**El atributo** `value` se emplea para establecer el valor inicial del cuadro de texto. Si se crea un formulario para insertar datos, los cuadros de texto deberían estar vacíos. Por lo tanto, o no se añade el atributo `value` o se incluye con un valor vacío `value=""`. Si por el contrario se crea un formulario para modificar datos, lo lógico es que se muestren inicialmente los datos guardados en el sistema. En este caso, el atributo `value` incluirá el valor que se desea mostrar: `<input type="text" name="nombre" value="Juan Pérez" />`

Si no se especifica un tamaño, el navegador muestra el cuadro de texto con un tamaño predeterminado. **El atributo** `size` permite establecer el tamaño, en caracteres, con el que se muestra el cuadro de texto. Su uso es imprescindible en muchos formularios, en los que algunos campos como la dirección deben mostrar más caracteres de lo normal (`<input size="100" ...>`) y otros campos como el código postal deben mostrar menos caracteres de lo normal (`<input size="5" ...>`).

Además de controlar el tamaño con el que se muestra un cuadro de texto, también se puede limitar el tamaño del texto introducido. El **atributo** `maxlength` permite establecer el máximo número de caracteres que el usuario puede introducir en un cuadro de texto. Su uso es imprescindible para campos como el código postal, el número de la Seguridad Social y cualquier otro dato con formato predefinido y limitado.

Por último, el **atributo** `readonly` permite que el usuario pueda ver los contenidos del cuadro de texto pero no pueda modificarlos y el atributo `disabled` deshabilita un cuadro de texto de forma que el usuario no pueda modificarlo y además, el navegador no envía sus datos al servidor.

La única diferencia entre este control y el cuadro de texto normal es que el texto que el usuario escribe en un cuadro de contraseña no se ve en la pantalla. En su lugar, los navegadores ocultan el texto utilizando asteriscos o círculos, por lo que es ideal para escribir contraseñas y otros datos sensibles.

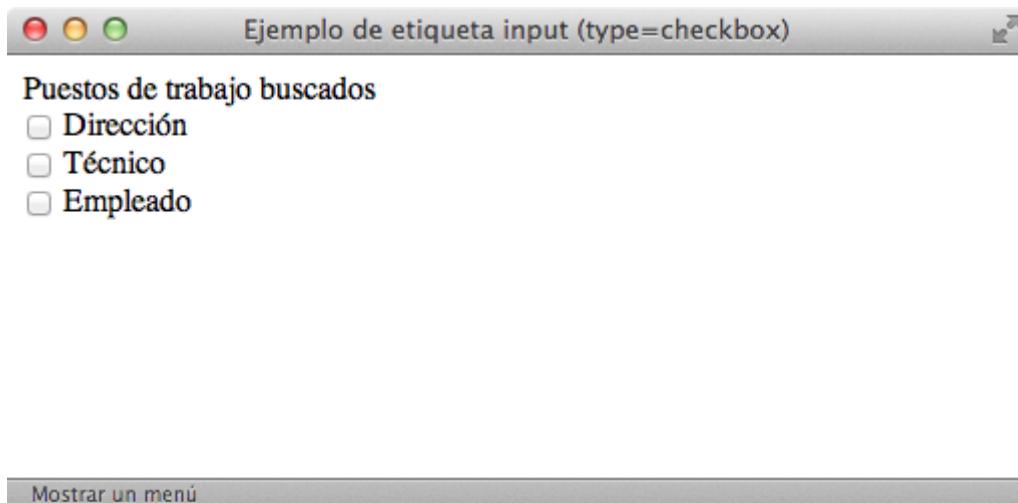


**Figura 9.3 Ejemplo de etiqueta input (type=password)**

```
Contraseña <br/>
<input type="password" name="contrasena" value="" />
```

Cambiando el valor del atributo `type` por `password` se transforma el cuadro de texto normal en un cuadro de contraseña. Todos los demás atributos se utilizan de la misma forma y tienen el mismo significado.

Los *checkbox* o "casillas de verificación" son controles de formulario que permiten al usuario seleccionar y deseleccionar opciones individualmente. Aunque en ocasiones se muestran varios *checkbox* juntos, cada uno de ellos es completamente independiente del resto. Por este motivo, se utilizan cuando el usuario puede activar y desactivar varias opciones relacionadas pero no excluyentes.



**Figura 9.4 Ejemplo de etiqueta input (type=checkbox)**

```
Puestos de trabajo buscados <br/>
<input name="puesto_direct" type="checkbox" value="direccion"/> Dirección
<input name="puesto_tecnico" type="checkbox" value="tecnico"/> Técnico
<input name="puesto_empleado" type="checkbox" value="empleado"/> Empleado
```

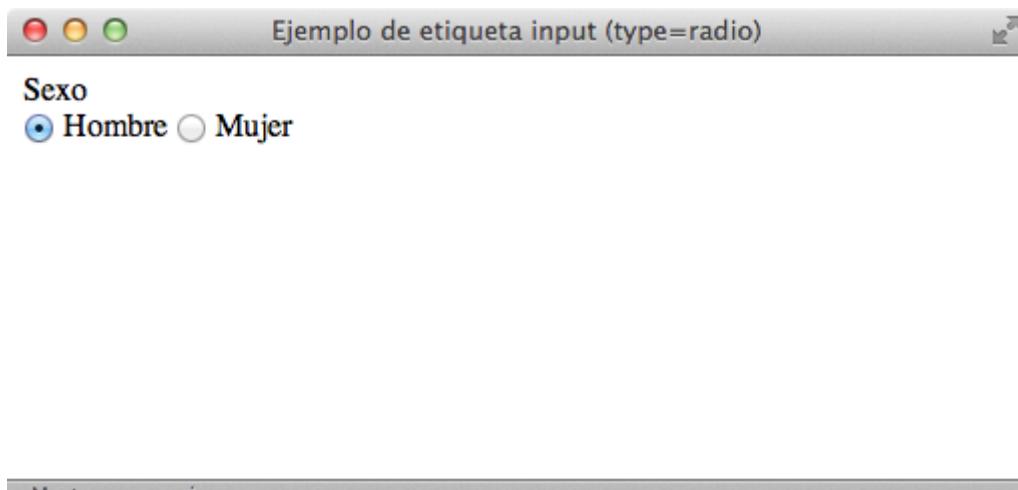
El valor del atributo `type` para estos controles de formulario es `checkbox`. Como se muestra en el ejemplo anterior, el texto que se encuentra al lado de cada `checkbox` no se puede establecer mediante ningún atributo, por lo que es necesario añadirlo manualmente fuera del control del formulario. Si no se añade un texto al lado de la etiqueta `<input />` del `checkbox`, el usuario sólo ve un pequeño cuadrado sin ninguna información relativa a la finalidad de ese `checkbox`.

El valor del atributo `value`, junto con el valor del atributo `name`, es la información que llega al servidor cuando el usuario envía el formulario.

Si se quiere mostrar un `checkbox` seleccionado por defecto, se utiliza el atributo `checked`. Si el valor del atributo es `checked`, el `checkbox` se muestra seleccionado. En cualquier otro caso, el `checkbox` permanece sin seleccionar. Aunque resulta redundante que el nombre y el valor del atributo sean idénticos, es obligatorio indicarlo de esta forma porque los atributos en XHTML no pueden tener valores vacíos:

```
<input type="checkbox" checked="checked" /> Checkbox seleccionado
```

Los controles de tipo `radiobutton` son similares a los controles de tipo `checkbox`, pero presentan una diferencia muy importante: son mutuamente excluyentes. Los `radiobutton` se utilizan cuando el usuario solamente puede escoger una opción entre las distintas opciones relacionadas que se le presentan. Cada vez que se selecciona una opción, automáticamente se deselecciona la otra opción que estaba seleccionaba.

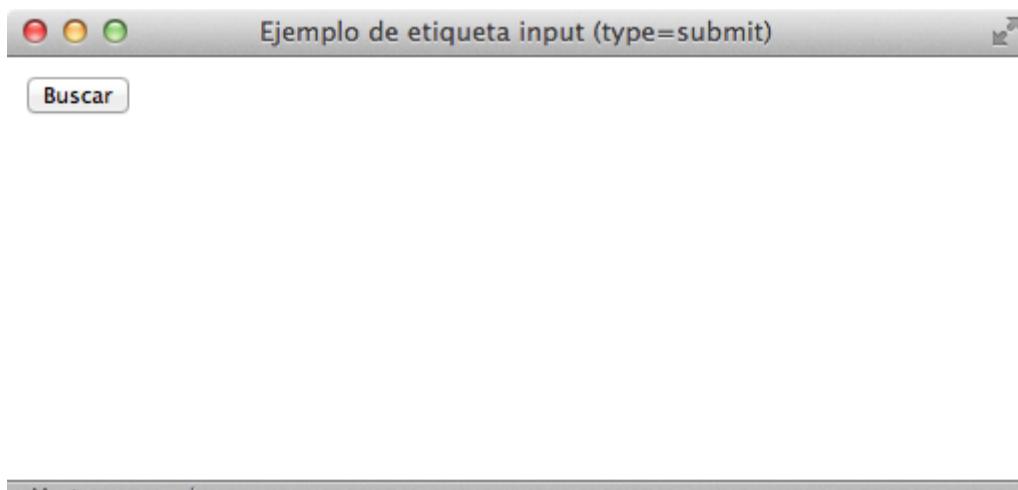


**Figura 9.5** Ejemplo de etiqueta input (type=radio)

```
Sexo <br/>
<input type="radio" name="sexo" value="hombre" checked="checked" />
Hombre
<input type="radio" name="sexo" value="mujer" /> Mujer
```

El valor del atributo `type` para estos controles de formulario es `radio`. El atributo `name` se emplea para indicar los *radiobutton* que están relacionados. Por lo tanto, cuando varios *radiobutton* tienen el mismo valor en su atributo `name`, el navegador sabe que están relacionados y puede deseleccionar una opción del grupo de *radiobutton* cuando se seleccione otra opción.

La mayoría de formularios dispone de un botón para enviar al servidor los datos introducidos por el usuario:

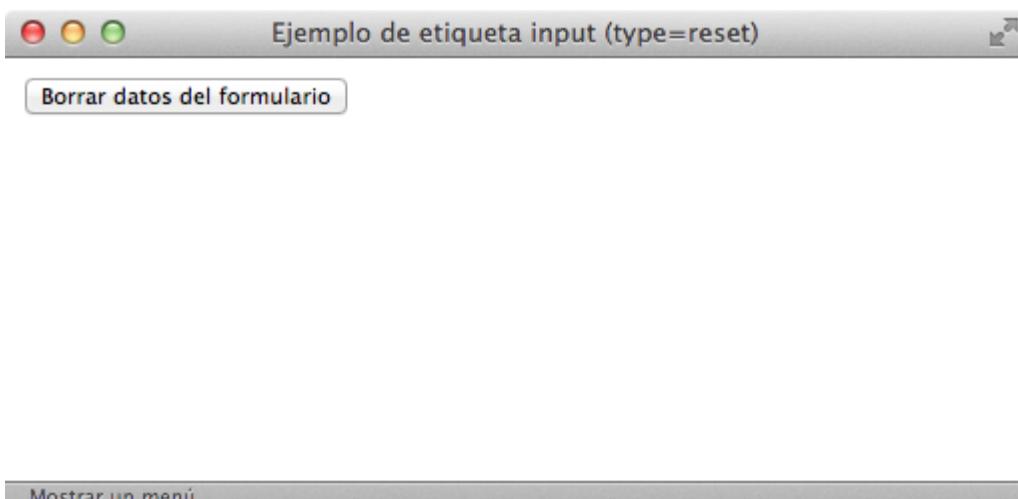


**Figura 9.6** Ejemplo de etiqueta input (type=submit)

```
<input type="submit" name="buscar" value="Buscar" />
```

El valor del atributo `type` para este control de formulario es `submit`. El navegador se encarga de enviar automáticamente los datos cuando el usuario pincha sobre este tipo de botón. El valor del atributo `value` es el texto que muestra el botón. Si no se establece el atributo `value`, el navegador muestra el texto predefinido Enviar consulta.

Aunque su uso era muy popular hace unos años, la mayoría de formularios modernos ya no utilizan este tipo de botón. Se trata de un botón especial que borra todos los datos introducidos por el usuario y devuelve el formulario a su estado original:



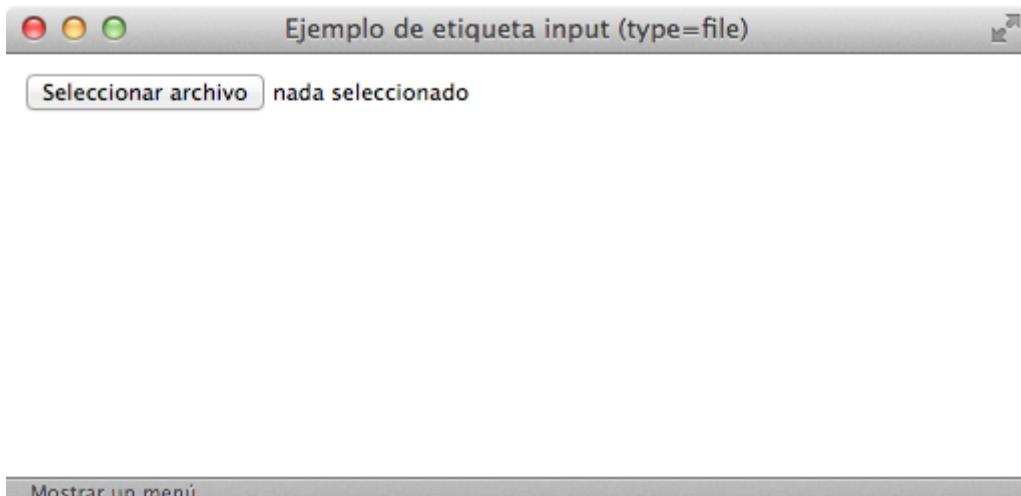
**Figura 9.7 Ejemplo de etiqueta input (type=reset)**

```
<input type="reset" name="limpiar" value="Borrar datos del formulario" />
```

El valor del atributo `type` para este control de formulario es `reset`. Cuando el usuario pulsa este botón, el navegador borra toda la información introducida y muestra el formulario en su estado original. Si el formulario no contenía originalmente ningún valor, el botón de `reset` lo vuelve a mostrar vacío. Si el formulario contenía información, el botón `reset` vuelve a mostrar la misma información original.

Como es habitual en los botones de formulario, el atributo `value` permite establecer el texto que muestra el botón. Si no es utiliza este atributo, el navegador muestra el texto predefinido del botón, que en este caso es Restablecer.

Los formularios también permiten adjuntar archivos para subirlos al servidor. Aunque desde el punto de vista de HTML y del navegador no existe ninguna limitación sobre el número, tipo o tamaño total de los archivos que se pueden adjuntar, todos los servidores añaden restricciones por motivos de seguridad.



**Figura 9.8** Ejemplo de etiqueta input (type=file)

```
Fichero adjunto  
<input type="file" name="adjunto" />
```

El valor del atributo `type` para este control de formulario es `file`. El navegador se encarga de mostrar un cuadro de texto donde aparece el nombre del archivo seleccionado y un botón que permite navegar por los directorios y archivos del ordenador del usuario.

Si se incluye un control para adjuntar archivos, es obligatorio añadir el atributo `enctype` en la etiqueta `<form>` del formulario. El valor del atributo `enctype` debe ser `multipart/form-data`, por lo que la etiqueta `<form>` de los formularios que permiten adjuntar archivos siempre es:

```
<form action="..." method="post" enctype="multipart/form-data">  
...  
</form>
```

Los campos ocultos se emplean para añadir información oculta en el formulario:

```
<input type="hidden" name="url_previa" value="/articulo/primer.html" />
```

El valor del atributo `type` para este control de formulario es `hidden`. Los campos ocultos no se muestran por pantalla, de forma que el usuario desconoce que el formulario los incluye. Normalmente los campos ocultos se utilizan para incluir información que necesita el servidor pero que no es necesario o no es posible que la establezca el usuario.

El aspecto de los botones de formulario se puede personalizar por completo, ya que incluso es posible utilizar una imagen como botón:

```
| <input type="image" name="enviar" src="accept.png" />
```

El valor del atributo `type` para este control de formulario es `image`. El atributo `src` indica la URL de la imagen que debe mostrar el navegador en lugar del botón normal.

Su principal ventaja es que permite personalizar por completo la estética de los botones y mostrarlos con un aspecto homogéneo en todos los navegadores. El principal inconveniente es que ralentiza la carga del formulario y que si se quiere modificar su aspecto, es necesario crear una nueva imagen.

Algunos formularios complejos necesitan botones más avanzados que los de enviar datos (`type="submit"`) y resetear el formulario (`type="reset"`). Por ese motivo, el estándar HTML/XHTML define un botón de tipo genérico:

```
| <input type="button" name="guardar" value="Guardar Cambios" />
```

El valor del atributo `type` para este control de formulario es `button`. Si pruebas a pulsar un botón de este tipo, verás que el navegador no hace nada: no envía los datos al servidor y no borra los datos introducidos. Este tipo de botones sólo son útiles si se utilizan junto con el lenguaje de programación JavaScript. Si la página incluye código JavaScript, los botones de este tipo se pueden programar para que realicen cualquier tarea compleja cuando se pulsa sobre ellos.

Utilizando solamente las etiquetas `<form>` y `<input>` es posible diseñar la mayoría de formularios de las aplicaciones web. No obstante, HTML de-

fine algunos elementos adicionales para mejorar la estructura de los formularios creados.

La etiqueta `<fieldset>` agrupa campos del formulario y la etiqueta `<legend>` asigna un nombre a cada grupo.

| <code>&lt;fieldset&gt;</code> |   |
|-------------------------------|---|
| <b>Atributos comunes</b>      | básicos, internacionalización, eventos                                |
| <b>Atributos propios</b>      | -   |
| <b>Tipo de elemento</b>       | En bloque   |
| <b>Descripción</b>            | Se emplea para agrupar de forma lógica varios campos de un formulario |

| <code>&lt;legend&gt;</code> |   |
|-----------------------------|---|
| <b>Atributos comunes</b>    | básicos, internacionalización, eventos  |
| <b>Atributos propios</b>    | -   |
| <b>Tipo de elemento</b>     | En línea  |
| <b>Descripción</b>          | Se emplea para definir el título o leyenda de un conjunto de campos de formulario agrupados con la etiqueta <code>fieldset</code> |

Un ejemplo de **código HTML** con las etiquetas `<fieldset>` y `<legend>` sería:

```
<form action="maneja_formulario.php" method="post">
<fieldset>
    <legend>Datos personales</legend>
    Nombre <br/>
    <input type="text" name="nombre" value="" />
    <br/>
```

```
Apellidos <br/>
<input type="text" name="apellidos" value="" />
<br/>
DNI <br/>
<input type="text" name="dni" value="" size="10" maxlength="9" />
</fieldset>
<fieldset>
<legend>Datos de conexión</legend>
Nombre de usuario<br/>
<input type="text" name="nombre" value="" maxlength="10" />
<br/>
Contraeña<br/>
<input type="password" name="password" value="" maxlength="10" />
<br/>
Repite la contraeña<br/>
<input type="password" name="password2" value="" maxlength="10" />
</fieldset>
</form>
```

Y un navegador lo visualizaría de esta manera:

The screenshot shows a web browser window with a title bar reading "Ejemplo de etiqueta fieldset y legend". The main content area displays a form titled "Formulario estructurado". The form is divided into two sections: "Datos personales" and "Datos de conexión".

**Datos personales**

- Nombre:
- Apellidos:
- DNI:

**Datos de conexión**

- Nombre de usuario:
- Contraeña:
- Repite la contraeña:

**Figura 9.9** Ejemplo de uso de las etiquetas fieldset y legend

La etiqueta `<fieldset>` agrupa todos los controles de formulario a los que encierra. El navegador muestra por defecto un borde resaltado para cada agrupación. La etiqueta `<legend>` se incluye dentro de cada etiqueta `<fieldset>` y establece el título que muestra el navegador para cada agrupación de elementos.

Por otra parte, todos los controles de formulario salvo los botones presentan una carencia muy importante: no disponen de la opción de establecer el título o texto que se muestra junto al control. En el código HTML del ejemplo anterior, el nombre de cada campo se incluye en forma de texto normal, sin ninguna relación con el campo al que hace referencia.

Afortunadamente, el lenguaje HTML incluye una etiqueta denominada `<label>` y que se utiliza para establecer el título de cada campo del formulario. Su definición formal es la siguiente:

| <code>&lt;label&gt;</code> |  |
|----------------------------|--|
| <b>Atributos comunes</b>   | básicos, internacionalización, eventos   |
| <b>Atributos propios</b>   | <code>for="id"</code> Indica el ID del campo del formulario para el que este elemento es su título<br>Otros: <code>accesskey</code> , <code>onfocus</code> y <code>onblur</code> |
| <b>Tipo de elemento</b>    | En línea   |
| <b>Descripción</b>         | Se emplea para definir el título o leyenda de un conjunto de campos de formulario agrupados con la etiqueta <code>fieldset</code>  |

El único atributo que suele utilizarse con la etiqueta `<label>` es `for`, que indica el identificador (atributo `id`) del campo de formulario para el que esta etiqueta hace de título.

En el anterior ejemplo, el nombre de los campos de formulario se incluía mediante un texto normal:

```
Nombre <br/>
<input type="text" name="nombre" value="" />
Apellidos <br/>
<input type="text" name="apellidos" value="" />
```

```
DNI <br/>
<input type="text" name="dni" value="" size="10" maxlength="9" />
```

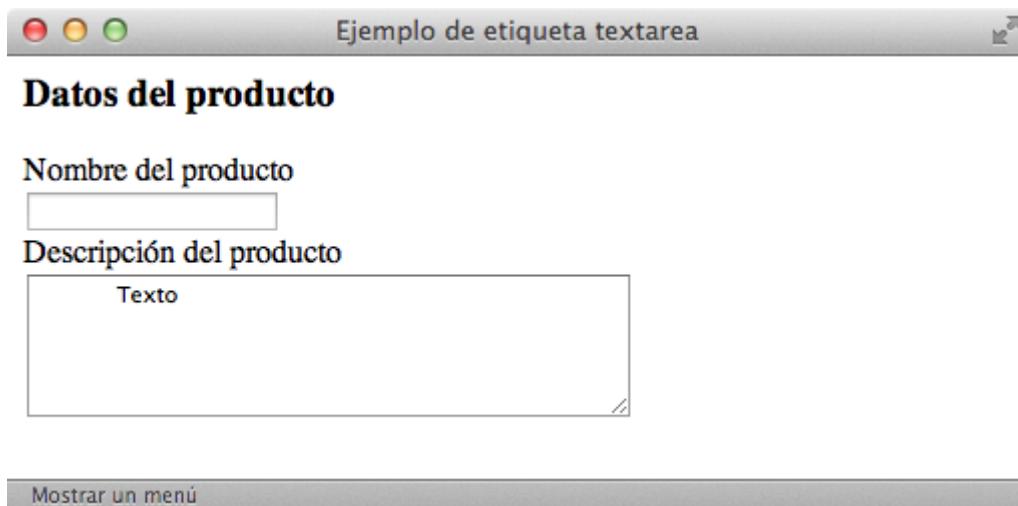
Utilizando la etiqueta `<label>`, cada campo de formulario puede disponer de su propio título:

```
<label for="nombre">Nombre</label> <br/>
<input type="text" id="nombre" name="nombre" value="" />
<label for="apellidos">Apellidos</label> <br/>
<input type="text" id="apellidos" name="apellidos" value="" />
<label for="dni">DNI</label> <br/>
<input type="text" id="dni" name="dni" value="" size="10" maxlength="9"
/>
```

La principal ventaja de utilizar `<label>` es que el código HTML está mejor estructurado y se mejora su accesibilidad. Además, al pinchar sobre el texto del `<label>`, el puntero del ratón se posiciona automáticamente para poder escribir sobre el campo de formulario asociado. Este comportamiento es especialmente útil para los campos de tipo `radio`button y `checkbox`.

La etiqueta `<input>` permite crear diez tipos diferentes de controles de formulario. Sin embargo, algunas aplicaciones web utilizan otros elementos de formulario que no se pueden crear con `<input>`. Las listas desplegables y las áreas de texto disponen de sus propias etiquetas (`<select>` y `<textarea>` respectivamente).

Las áreas de texto son útiles cuando se debe introducir una gran cantidad de texto, ya que es mucho más cómodo de introducir que en un campo de texto normal:



**Figura 9.10** Ejemplo de uso de la etiqueta textarea

```
<form action="insertar_producto.php" method="post">
    <label for="nombre">Nombre del producto</label> <br/>
    <input type="text" id="nombre" name="nombre" value="" />
    <label for="descripcion">Descripción del producto</label> <br/>
    <textarea id="descripcion" name="descripcion" cols="40" rows="5">
        Texto
    </textarea>
</form>
```

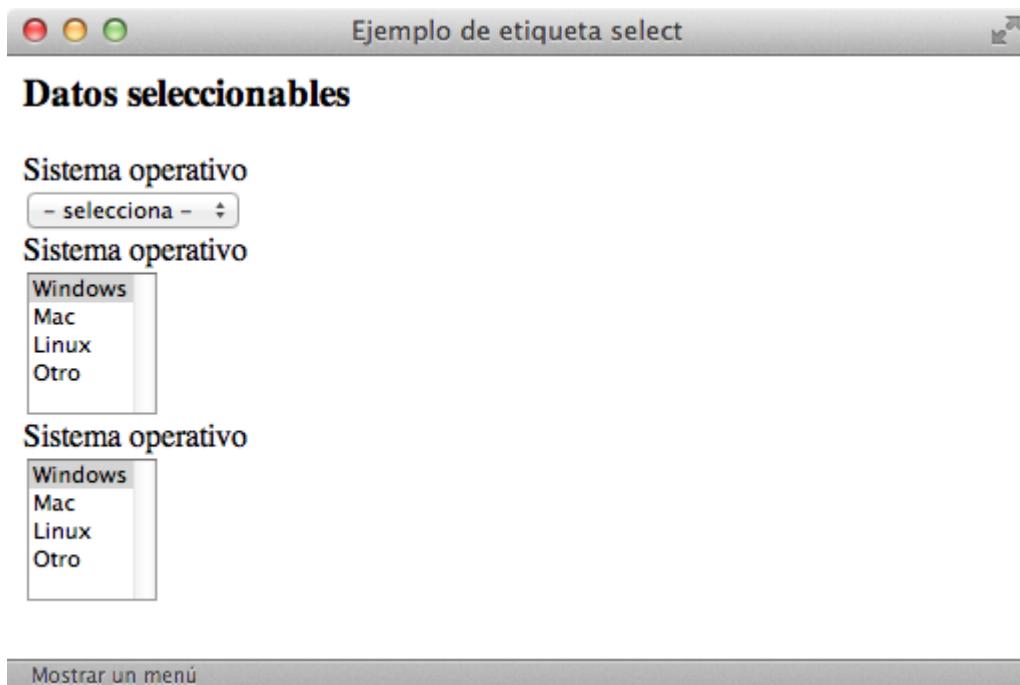
La definición formal de la etiqueta <textarea> es:

| <textarea>               |   |
|--------------------------|---|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | rows="numero" Número de filas de texto que mostrará el textarea<br>cols="numero" Número de caracteres que se muestran en cada fila del textarea<br>Otros: name, disabled, readonly, onselect, onchange, onfocus, onblur |
| <b>Tipo de elemento</b>  | En línea  |
| <b>Descripción</b>       | Se emplea para incluir un área de texto en un formulario  |

Los atributos más utilizados en las etiquetas <textarea> son los que controlan su anchura y altura. La anchura del área de texto se controla mediante el atributo cols, que indica las columnas o número de caracteres que se podrán escribir como máximo en cada fila. La altura del área de texto se controla mediante rows, que indica directamente las filas de texto que serán visibles.

El principal inconveniente de los elementos <textarea> es que el lenguaje HTML no permite limitar el número máximo de caracteres que se pueden introducir. Mientras los elementos <input type="text"> disponen del atributo maxlength, las áreas de texto no disponen de un atributo equivalente, por lo que sólo es posible limitar el número de caracteres mediante su programación con JavaScript.

Por otra parte, el otro control disponible para los formularios es el de las listas desplegables:



**Figura 9.11 Ejemplo de uso de la etiqueta select**

La imagen anterior muestra los tres tipos de listas desplegables disponibles. El primero es el de las listas más utilizadas que sólo muestran un valor cada vez y sólo permiten seleccionar un valor. El segundo tipo de lista es el que sólo permite seleccionar un valor pero muestra varios a la vez. Por último, el tercer tipo de lista desplegable es aquella que muestra varios valores y permite realizar selecciones múltiples.

El **código HTML** del ejemplo anterior sería:

```
<label for="so">Sistema operativo</label> <br/>
<select id="so" name="so">
    <option value="" selected="selected">- selecciona -</option>
    <option value="windows">Windows</option>
    <option value="mac">Mac</option>
    <option value="linux">Linux</option>
    <option value="otro">Otro</option>
</select>
<label for="so2">Sistema operativo</label> <br/>
<select id="so2" name="so2" size="5">
    <option value="windows" selected="selected">Windows</option>
    <option value="mac">Mac</option>
    <option value="linux">Linux</option>
    <option value="otro">Otro</option>
</select>
<label for="so3">Sistema operativo</label> <br/>
<select id="so3" name="so3" size="5" multiple="multiple">
    <option value="windows" selected="selected">Windows</option>
    <option value="mac">Mac</option>
    <option value="linux">Linux</option>
    <option value="otro">Otro</option>
</select>
```

Los tres tipos de listas desplegables se definen con la misma etiqueta `<select>` y cada elemento de la lista se define mediante la etiqueta `<option>`:

| <code>&lt;select&gt;</code> |  |
|-----------------------------|--|
| <b>Atributos comunes</b>    | básicos, internacionalización, eventos   |
| <b>Atributos propios</b>    | <code>size="numero"</code> Número de filas que se muestran de la lista (por defecto sólo se muestra una)<br><code>multiple="multiple"</code> Si se incluye, se permite seleccionar más de un elemento<br>Otros: <code>name</code> , <code>disabled</code> , <code>onchange</code> , <code>onfocus</code> , <code>onblur</code> |
| <b>Tipo de elemento</b>     | En línea   |
| <b>Descripción</b>          | Se emplea para incluir una lista desplegable en un formulario  |

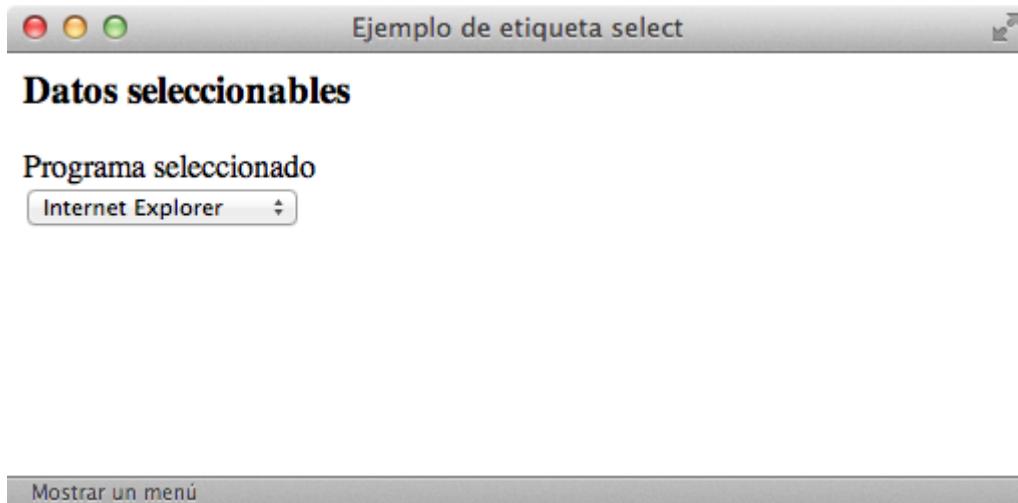
|                          |   |
|--------------------------|---|
|                          | <option>  |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos  |
| <b>Atributos propios</b> | <p>selected="selected" Indica si el elemento aparece seleccionado por defecto al cargarse la página</p> <p>value="texto" El valor que se envía al servidor cuando el usuario elige esa opción</p> <p>Otros: label, disabled</p> |
| <b>Tipo de elemento</b>  | -   |
| <b>Descripción</b>       | Se emplea para definir cada elemento de una lista desplegable   |

La inmensa mayoría de listas desplegables que utilizan las aplicaciones web son simples, por lo que el código HTML habitual de las listas desplegables es:

```
<label for="so">Sistema operativo</label> <br/>
<select id="so" name="so">
    <option value="" selected="selected">- selecciona -</option>
    <option value="windows">Windows</option>
    <option value="mac">Mac</option>
    <option value="linux">Linux</option>
    <option value="otro">Otro</option>
</select>
```

La etiqueta <select> define la lista y encierra todas las opciones que muestra la lista. Cada una de las opciones de la lista se define mediante una etiqueta <option>. El atributo value de cada opción es obligatorio, ya que es el dato que se envía al servidor cuando el usuario envía el formulario. Para seleccionar por defecto una opción al mostrar la lista, se añade el atributo selected a la opción deseada.

Por otra parte, las listas desplegables permiten agrupar sus opciones de forma que el usuario pueda encontrar fácilmente las opciones cuando la lista es muy larga:



**Figura 9.12** Ejemplo de uso de la etiqueta optgroup

El **código HTML** correspondiente a la imagen anterior sería:

```
<form id="formulario" method="post" action="">
<label for="programa">Programa seleccionado</label> <br/>
<select id="programa" name="programa">
    <optgroup label="Sistemas Operativos">
        <option value="Windows" selected="selected">Windows</option>
        <option value="Mac">Mac</option>
        <option value="Linux">Linux</option>
        <option value="Other">Otro</option>
    </optgroup>
    <optgroup label="Navegadores">
        <option value="Internet Explorer" selected="selected">Internet
        Explorer</option>
        <option value="Firefox">Firefox</option>
        <option value="Safari">Safari</option>
        <option value="Opera">Opera</option>
        <option value="Other">Otro</option>
    </optgroup>
</select>
</form>
```

La etiqueta `<optgroup>` permite agrupar opciones relacionadas dentro de una lista desplegable. Su definición formal se muestra a continuación:

|                          |  |
|--------------------------|--|
|                          | <code>&lt;optgroup&gt;</code>          |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos |

|                          |  |
|--------------------------|--|
|                          | <optgroup>   |
| <b>Atributos propios</b> | label="texto" Texto que se muestra como título de la agrupación de opciones<br>Otros: disabled, selected |
| <b>Tipo de elemento</b>  | -  |
| <b>Descripción</b>       | Se emplea para definir una agrupación lógica de opciones de una lista desplegable                        |

El único atributo que suele utilizarse con la etiqueta <optgroup> es label, que indica el nombre de cada agrupación. Los navegadores muestran de forma destacada el título de cada agrupación, de forma que el usuario pueda localizar más fácilmente la opción deseada.

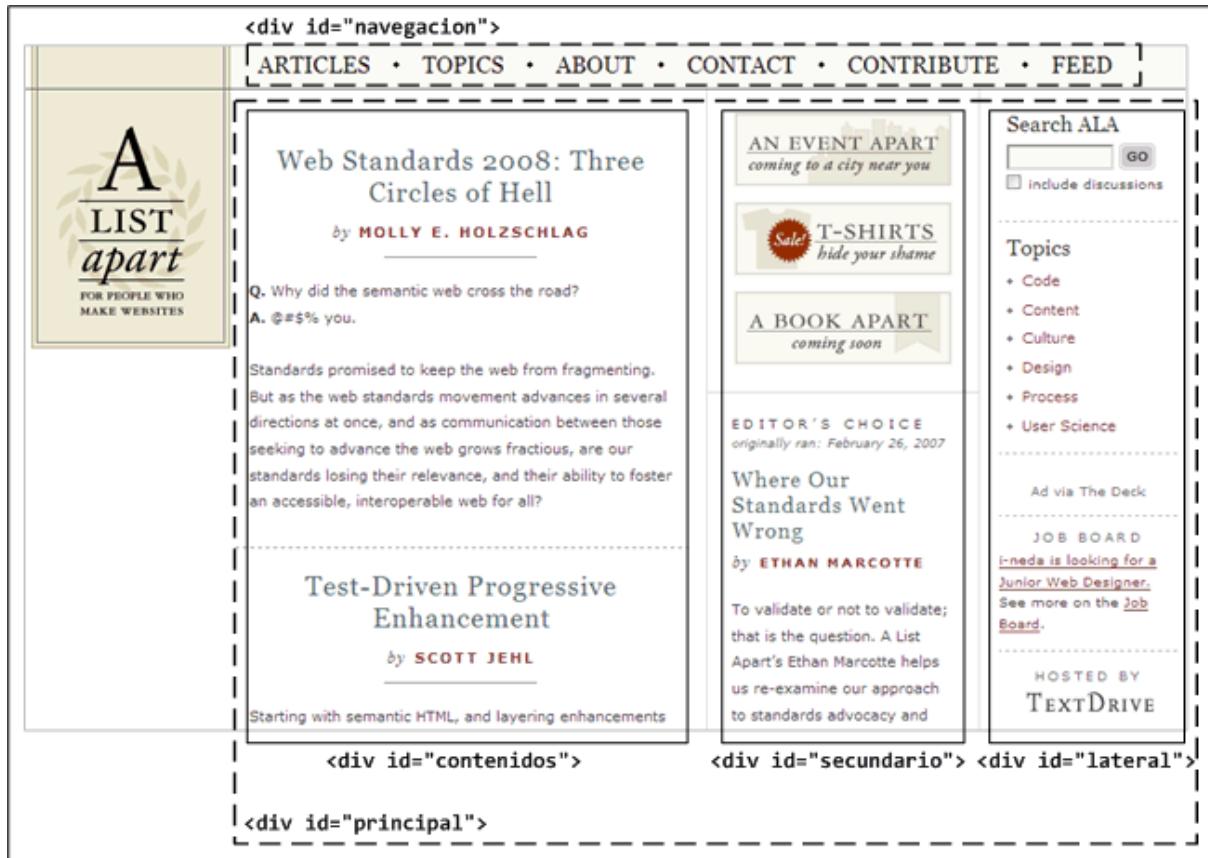
## Capítulo 10

Los capítulos anteriores muestran las decenas de etiquetas XHTML disponibles para marcar y estructurar cada elemento individual de las páginas web: tablas, listas, enlaces, párrafos, imágenes, etc. Aunque combinando esas etiquetas es posible crear cualquier página web, no es posible hacer que las páginas muestren **estructuras complejas**.

La mayoría de **páginas HTML** disponen de estructuras complejas formadas por varias columnas de contenidos y otro tipo de divisiones. Utilizando exclusivamente HTML no es posible crear estas estructuras complejas, ya que es imprescindible emplear las **hojas de estilos CSS**.

No obstante, los estilos de CSS necesitan la ayuda de HTML/XHTML para crear los diseños más avanzados. En concreto, el código HTML se encarga de **agrupar** los elementos de la página en diferentes divisiones en función de su finalidad: la zona de la cabecera de la página, la zona de contenidos, una zona lateral para el menú y otras secciones menores, la zona del pie de página, etc.

La siguiente imagen muestra algunas de las zonas definidas en la página principal del sitio [www.alistapart.com](http://www.alistapart.com):



**Figura 10.1** Ejemplo de página compleja estructurada con etiquetas div

Para agrupar los elementos que forman cada zona o división de la página se utiliza la etiqueta <div>:

| <div>                    |  |
|--------------------------|--|
| <b>Atributos comunes</b> | básicos, internacionalización, eventos |
| <b>Atributos propios</b> | -                                      |
| <b>Tipo de elemento</b>  | En bloque                              |
| <b>Descripción</b>       | Agrupa elementos de bloque             |

El nombre de la etiqueta `div` tiene su origen en la palabra división, ya que esta etiqueta define zonas o divisiones dentro de una página HTML. En cualquier caso, casi todos los diseñadores web utilizan la palabra "capa" para referirse a una "división". Aunque se trata de un error grave (las capas se crean mediante una propiedad de CSS llamada `z-index`) es preferible seguir llamando "capas" a las zonas definidas con la etiqueta `<div>` para poder entenderse con el resto de diseñadores.

Las páginas web complejas que están bien diseñadas utilizan decenas de etiquetas `<div>`. Con mucha diferencia, los atributos más utilizados con esta etiqueta son `id` (para identificar la capa de forma única) y `class` (para aplicar a la capa estilos CSS).

No se va a profundizar en el proceso de diseñar una página web mediante `<div>`, ya que no es posible diseñar una página web compleja utilizando elementos `<div>` sin utilizar hojas de estilos CSS.

Por último, si observas el código HTML de algunas páginas web complejas, verás que la mayoría utilizan los mismos nombres para identificar sus divisiones. Los nombres más comunes, y sus equivalentes en inglés, se muestran a continuación:

- **contenedor** (wrapper) suele encerrar la mayor parte de los contenidos de la página y se emplea para definir las características básicas de la página: su anchura, sus bordes, imágenes laterales, si se centra o no respecto de la ventana del navegador, etc.
- **cabecera** (header) que incluye todos los elementos invariantes de la parte superior de la página (logotipo, imagen o banner, cuadro de búsqueda superior, etc.)
- **contenido** (content) engloba el contenido principal del sitio (la zona de noticias, la zona de artículos, la zona de productos, etc. dependiendo del tipo de sitio web)
- **menu** (menu) se emplea para agrupar todos los elementos del menú lateral de navegación de la página
- **pie** (footer) que incluye todos los elementos invariantes de la parte inferior de la página (aviso de copyright, política de privacidad, términos de uso, etc.)
- **lateral** (sidebar) se emplea para agrupar los elementos de las columnas laterales y secundarias de la página.

De esta forma, el esqueleto de una página HTML compleja suele ser similar al siguiente:

```
...
<div id="contenedor">
  <div id="cabecera">
    ...
  </div>
```

```
<div id="contenido">
  <div id="menu">
    ...
  </div>
  ...
</div>
<div id="pie">
  ...
</div>
</div>
...
```

El equivalente para las páginas en inglés sería el siguiente:

```
...
<div id="wrapper">
  <div id="header">
    ...
  </div>
  <div id="content">
    <div id="menu">
      ...
    </div>
    ...
  </div>
  <div id="footer">
    ...
  </div>
</div>
```

# Capítulo 11

Las páginas y documentos HTML incluyen más información de la que los usuarios ven en sus pantallas. Estos **datos adicionales** siempre están relacionados con la propia página, por lo que se denominan **metainformación** o metadatos. La metainformación siempre se incluye en la sección de la cabecera, es decir, dentro de la etiqueta <head>.

Aunque la metainformación más conocida y utilizada es el título de la propia página, se puede incluir mucha otra información útil para los navegadores y para los **buscadores**. En las próximas secciones se explica cómo incluir la metainformación y se introduce un concepto relacionado llamado DOCTYPE.

Como ya se explicó anteriormente, las páginas XHTML se dividen en dos partes denominadas cabecera y cuerpo. La sección de la cabecera está formada por todas las etiquetas encerradas por la etiqueta <head>:

| <head>                   |   |
|--------------------------|---|
| <b>Atributos comunes</b> | internacionalización  |
| <b>Atributos propios</b> | profile="url" Especifica la URL del perfil o perfiles que utilizan los metadatos<br>lang="codigo" Especifica el idioma principal de los contenidos de la página |

|                         |                                       |
|-------------------------|---------------------------------------|
|                         | <head>                                |
| <b>Tipo de elemento</b> | En bloque                             |
| <b>Descripción</b>      | Define la cabecera del documento HTML |

La cabecera típica de una página HTML completa presenta la siguiente estructura:

```
<head>
  <!-- Zona de etiquetas META -->
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <!-- Zona de título -->
  <title>El título del documento</title>
  <!-- Zona de recursos enlazados (CSS, RSS, JavaScript) -->
  <link rel="stylesheet" href="#" type="text/css" media="screen" />
  <link rel="stylesheet" href="#" type="text/css" media="print" />
  <link rel="alternate" type="application/rss+xml"
        title="RSS 2.0" href="#" />
  <script src="#" type="text/javascript"></script>
</head>
```

La **etiqueta** `<title>` establece el **título** de la página. Los navegadores muestran este título como título de la propia ventana del navegador. Los buscadores utilizan este título como título de sus resultados de búsqueda.

Por tanto, el valor de `<title>` no sólo es importante para los usuarios, sino que también es importante para que los usuarios encuentren las páginas a través de los buscadores. Un error común de muchos sitios web consiste en mostrar un mismo título genérico en todas sus páginas. **Cada página debe mostrar un título corto**, adecuado, único y que describa inequívocamente los contenidos de la página.

Las páginas XHTML deben tener definido un título y sólo uno, por lo que todas las páginas web deben incluir obligatoriamente una etiqueta `<title>`, cuya definición formal se muestra a continuación:

|                          |                      |
|--------------------------|----------------------|
|                          | <title>              |
| <b>Atributos comunes</b> | internacionalización |

|                          |  |
|--------------------------|--|
|                          | <title>  |
| <b>Atributos propios</b> | lang="codigo" Especifica el idioma principal del título de la página |
| <b>Tipo de elemento</b>  | -  |
| <b>Descripción</b>       | Define el título del documento HTML                                  |

```

<head profile="http://gmpg.org/xfn/11">
  ...
</head>

```

El documento <http://gmpg.org/xfn/11> es un perfil que define atributos adicionales para establecer la relación entre sitios web.

Una de las partes más importantes de la metainformación de la página son los **metadatos**, que permiten incluir cualquier información relevante sobre la propia página.

La especificación oficial de HTML no define la lista de metadatos que se pueden incluir, por lo que las páginas tienen **libertad absoluta** para definir los metadatos que consideren adecuados. La etiqueta empleada para la definición de los metadatos es <meta>.

|                          |   |
|--------------------------|---|
|                          | <meta>  |
| <b>Atributos comunes</b> | internacionalización  |
| <b>Atributos propios</b> | <p>name="texto" El nombre de la propiedad que se define (no existe una lista oficial de propiedades)</p> <p>content="texto" El valor de la propiedad definida (no existe una lista de valores permitidos)</p> <p>http-equiv="texto" En ocasiones, reemplaza al atributo name y lo emplean los servidores para adaptar sus respuestas al documento</p> <p>scheme="texto" Indica el esquema que se debe emplear para interpretar el valor de la propiedad</p> |

| <meta>                  |   |
|-------------------------|---|
| <b>Tipo de elemento</b> | -   |
| <b>Descripción</b>      | Permite definir el valor de los metadatos que forman la metainformación del documento |

Los metadatos habituales utilizan solamente los atributos name y content para definir el nombre y el valor del metadato:

```
<meta name="autor" content="Juan Pérez" />
```

No obstante, algunas etiquetas <meta> muy utilizadas hacen uso del atributo http-equiv. Este atributo se utiliza para indicar que el valor establecido por este metadato puede ser utilizado por el servidor al entregar la página al navegador del usuario. El siguiente metadato indica al servidor que el contenido de la página es código HTML y su codificación de caracteres es UTF-8:

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

El atributo scheme no suele utilizarse, aunque permite proporcionar información de contexto para que el navegador interprete correctamente el valor del metadato. En el siguiente ejemplo, el atributo scheme indica al navegador que el valor del metadato hace referencia al código ISBN:

```
<meta scheme="ISBN" name="identificador" content="789-1392349610">
```

Aunque no existe una lista oficial con los metadatos que se pueden definir, algunos de ellos se utilizan en tantas páginas que se han convertido prácticamente en un estándar. A continuación se muestran los metadatos más utilizados:

Definir el autor del documento:

```
<meta name="author" content="Juan Pérez" />
```

Definir el programa con el que se ha creado el documento:

```
<meta name="generator" content="WordPress 2.8.4" />
```

Definir la codificación de caracteres del documento:

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"
/>
```

Definir el copyright del documento:

```
<meta name="copyright" content="www.arkaitzgarro.com" />
```

Definir el comportamiento de los buscadores:

```
<meta name="robots" content="index, follow" />
```

Definir las palabras clave que definen el contenido del documento:

```
<meta name="keywords" content="diseño, css, hojas de estilos, web, html"
/>
```

Definir una breve descripción del sitio:

```
<meta name="description" content="Artículos sobre diseño web, usabilidad
y accesibilidad" />
```

La etiqueta que define la codificación de los caracteres (`http-equiv="Content-Type"`) se emplea prácticamente en todas las páginas y las etiquetas que definen la descripción (`description`) y las palabras clave (`keywords`) también son muy utilizadas.

**El estándar XHTML deriva de XML**, por lo que comparte con el muchas de sus normas y sintaxis. Uno de los conceptos fundamentales de XML es la utilización del **DTD** o *Document Type Definition* ("Definición del Tipo de Documento").

Un DTD es un **documento que recoge el conjunto de normas y restricciones** que deben cumplir los documentos de un determinado tipo. Si por ejemplo se define un DTD para los documentos relacionados con libros, se puede fijar como norma que cada libro tenga un título y sólo uno, que tenga uno o más autores, que la información sobre el número de páginas pueda ser opcional, etc.

El conjunto de normas, obligaciones y restricciones que se deben seguir al crear un documento de un determinado tipo, se recogen en su correspondiente DTD. El estándar XHTML define el DTD que deben seguir las

páginas y documentos XHTML. En este documento se definen las etiquetas que se pueden utilizar, los atributos de cada etiqueta y el tipo de valores que puede tener cada atributo.

En realidad, la versión 1.0 del estándar de XHTML define tres DTD diferentes. Para indicar el DTD utilizado al crear una determinada página, se emplea una etiqueta especial llamada doctype. La etiqueta doctype es el único elemento que se incluye fuera de la etiqueta <html> de la página. De hecho, la declaración del doctype es lo primero que se debe incluir en una página web, antes incluso que la etiqueta <html>.

Como veremos más adelante, para que una **página XHTML sea correcta y válida** es imprescindible que incluya el correspondiente doctype que indica el DTD utilizado. A continuación se muestran los tres DTD que se pueden utilizar al crear páginas XHTML:

## XHTML 1.0 Estricto

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

Se trata de la variante con las normas más estrictas y las restricciones más severas. Las páginas web que incluyan este doctype, no pueden utilizar atributos relacionados con el aspecto de los contenidos, por lo que requiere una separación total de código HTML y estilos CSS.

## XHTML 1.0 Transitorio

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

Se trata de una variante menos estricta que la anterior, ya que permite el uso de algunos atributos HTML relacionados con el aspecto de los elementos.

## XHTML 1.0 Frameset

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

Esta última variante la utilizan las páginas que están formadas por frames, una práctica completamente desaconsejada y que hoy en día sólo utilizan los sitios web obsoletos.

Si no tienes claro el DTD que más te conviene, deberías utilizar el XHTML 1.0 transitorio, ya que es más fácil crear páginas web válidas. Si tienes conocimientos avanzados de XHTML, puedes utilizar XHTML 1.0 estricto.

Por otra parte, además del DOCTYPE apropiado, también es necesario que las páginas web indiquen el namespace asociado. Un namespace en un documento XML permite diferenciar las etiquetas y atributos que pertenecen a cada lenguaje.

Si en un mismo documento se mezclan etiquetas de dos o más lenguajes derivados de XML (XHTML y SVG por ejemplo) y que tienen el mismo nombre, no se podría determinar a qué lenguaje pertenece cada etiqueta y por tanto, no se podría interpretar esa etiqueta o ese atributo. Los namespaces se indican mediante una URL.

El namespace que utilizan todas las páginas XHTML (independientemente de la versión y del DOCTYPE) es <http://www.w3.org/1999/xhtml> y se indica de la siguiente manera:

```
<html xmlns="http://www.w3.org/1999/xhtml">
...
</html>
```

De esta forma, es habitual que las páginas XHTML comiencen con el siguiente código:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="es" xml:lang="es">
```

Aunque el código anterior es mucho más complicado que una simple etiqueta <html>, es imprescindible para que las páginas XHTML creadas sean correctas y superen satisfactoriamente el proceso de validación que se muestra en los capítulos siguientes.

Afortunadamente, si utilizas un editor avanzado como Dreamweaver para crear las páginas, todo el código anterior se incluye de forma automática. Si creas las páginas a mano, sólo tienes que copiar y pegar ese código en cada nueva página.

Esta página se ha dejado vacía a propósito

## Capítulo 12

Al igual que la mayoría de lenguajes de marcado, HTML permite incluir **comentarios dentro de su código** para añadir **información que no se debe mostrar por pantalla**.

Normalmente, los diseñadores y programadores incluyen comentarios para marcar el comienzo y el final de las secciones de las páginas, para incluir avisos y notas para otros diseñadores o para incluir explicaciones sobre la forma en la que se ha creado el código HTML.

Aunque los comentarios no se muestran por pantalla y por tanto son invisibles para los usuarios, **sí que se descargan con el código HTML de la página**. Por este motivo, nunca debe incluirse información sensible o confidencial en los comentarios.

La sintaxis de los comentarios es la siguiente:

- Apertura del comentario: <!--
- Contenido del comentario: (cualquier texto)
- Cierre del comentario: -->

El siguiente ejemplo muestra el uso de los comentarios HTML para indicar el comienzo y final de cada sección. Recuerda que los comentarios

no se muestran por pantalla y que no influyen en la forma en la que se ven las páginas:

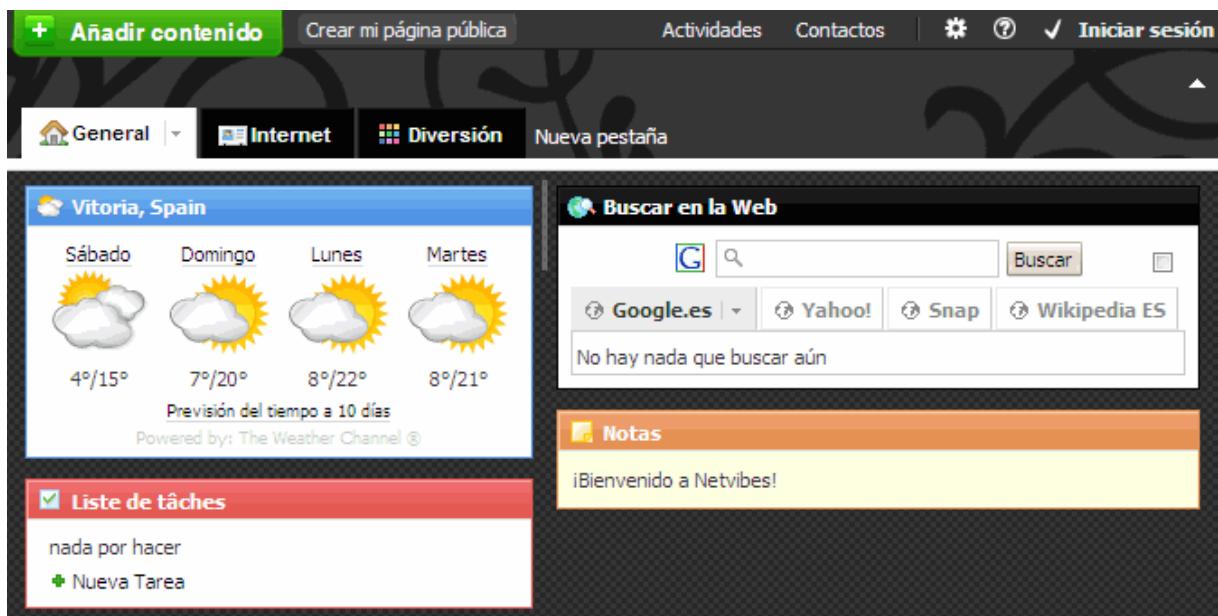
```
<!-- Inicio del menú -->
<div id="menu">
<ul>
  <li>...</li>
  <li>...</li>
  <li>...</li>
  <li>...</li>
</ul>
<!-- Fin del menú -->
<!-- Inicio de la publicidad -->
<div id="publicidad"> ... </div>
<!-- Fin de la publicidad -->
```

Los **comentarios de HTML** puede ocupar tantas líneas como sea necesario. Sin embargo, los comentarios no se pueden anidar, es decir, no se puede incluir un comentario dentro de otro comentario.

Como ya se explicó en los capítulos anteriores, la **etiqueta** `<script>` se utiliza para enlazar **archivos JavaScript externos** y para incluir **bloques de código JavaScript en las páginas**. Sin embargo, algunos navegadores no disponen de soporte completo de JavaScript, otros navegadores permiten bloquearlo parcialmente e incluso algunos usuarios bloquean completamente el uso de JavaScript porque creen que así navegan de forma más segura.

Si JavaScript está bloqueado o deshabilitado y la página web requiere su uso para un correcto funcionamiento, es habitual incluir un mensaje de aviso al usuario indicándole que debería activar JavaScript para disfrutar completamente de la página.

El siguiente ejemplo muestra una misma página web que requiere JavaScript tanto cuando se accede con JavaScript activado y como cuando se accede con JavaScript completamente desactivado.



**Figura 12.1** Ejemplo de página compleja con JavaScript activado



**Figura 12.2** Ejemplo de página compleja con JavaScript desactivado

HTML define la etiqueta <noscript> para incluir un mensaje que los navegadores muestran cuando JavaScript se encuentra bloqueado o desabilitado.

|                          |  |
|--------------------------|--|
|                          | <noscript>                             |
| <b>Atributos comunes</b> | básicos, internacionalización, eventos |
| <b>Atributos propios</b> | -                                      |

|                         |   |
|-------------------------|---|
|                         | <noscript>  |
| <b>Tipo de elemento</b> | En bloque   |
| <b>Descripción</b>      | Define un mensaje alternativo que se muestra al usuario cuando su navegador no soporta la ejecución de scripts. De esta forma, incluir un mensaje de aviso que solamente sea visible en los navegadores que tienen bloqueado JavaScript es tan sencillo como incluir la etiqueta <noscript> dentro del <body> |

```
<head> ... </head>
<body>
  <noscript>
    <p>Bienvenido a Mi Sitio</p>
    <p>La página que estás viendo requiere para su funcionamiento el uso de JavaScript. Si lo has deshabilitado intencionadamente, por favor vuelve a activarlo.</p>
  </noscript>
</body>
```

## Capítulo 13

La **validación** es el proceso que asegura que un **documento** escrito en un determinado lenguaje (por ejemplo XHTML) **cumple con las normas y restricciones de ese lenguaje**. Las normas y restricciones de los documentos escritos en XML (y en sus lenguajes derivados, como XHTML) se definen en el **DTD** o *Document Type Definition* ("Definición del Tipo de Documento").

El concepto de validación es objeto de controversia en el ámbito del diseño web. Por una parte, la validación **no es obligatoria** y las páginas web se pueden ver bien sin que sean válidas. Por otra parte, una página válida es **más correcta** que otra página que no lo sea, ya que cumple con las normas y restricciones impuestas por XHTML.

Debido a esto, algunos diseñadores consideran que se da demasiada importancia a la validación de las páginas y a la creación de páginas válidas. El resto de diseñadores argumentan que si la especificación de XHTML impone una serie de normas y restricciones, lo más correcto es que las páginas web las cumplan, aunque no sea obligatorio.

En cualquier caso, el proceso de validación consiste en **probar página a página si su código HTML pasa la prueba de validación**. Los validadores son las herramientas que se utilizan para validar cada página. Algunos editores de páginas web incluyen sus propios validadores y el organismo **W3C** ha creado una herramienta gratuita para la validación de las páginas.

En las próximas secciones de este capítulo se muestran las diferentes herramientas de validación disponibles para validar las páginas web.

La validación de las páginas web no requiere el uso de editores avanzados como Dreamweaver, ya que el organismo W3C ha creado una herramienta que se puede utilizar gratuitamente a través de Internet: <http://validator.w3.org/>

The screenshot shows the W3C Validator interface. At the top, there are three tabs: 'Validate by URI' (selected), 'Validate by File Upload', and 'Validate by Direct Input'. Below the tabs, the 'Validate by URI' section is active. It contains a sub-section titled 'Validate by URI' with the sub-instruction 'Validate a document online:'. A text input field labeled 'Address:' is present, along with a 'More Options' link. At the bottom of the section is a blue 'Check' button.

**Figura 13.1** Validador W3C

Aunque la herramienta sólo está disponible en inglés, su uso es muy intuitivo:

- Validate by URI, permite escribir la URL de la página que se quiere validar. Esta opción es la más sencilla para validar las páginas que ya están publicadas en Internet.
- Validate by File Upload, muestra un formulario mediante el que se puede subir el archivo HTML correspondiente a la página que se quiere validar. Esta opción es la mejor para validar las páginas web que has desarrollado y que aún no has publicado en Internet.
- Validate by Direct Input, permite validar código HTML de forma directa. Se trata de la opción más rápida para validar trozos o páginas HTML completas. Esta opción es la mejor cuando estás desarrollando las páginas y quieres asegurarte que el código sea válido.

La siguiente imagen muestra el **resultado de la validación** de la página principal de Google realizada mediante la opción Validate by URI:

| Errors found while checking this document as HTML5! |   |   |
|---|---|---|
| <b>Result:</b>                                      | 25 Errors, 4 warning(s)                             |   |
| <b>Address :</b>                                    | <input type="text" value="http://www.google.com/"/> |   |
| <b>Encoding :</b>                                   | iso-8859-1  | (detect automatically) <input type="button" value="▼"/> |
| <b>Doctype :</b>                                    | HTML5   | (detect automatically) <input type="button" value="▼"/> |
| <b>Root Element:</b>                                | html  |   |

**Figura 13.2** Resultado de validar www.google.com

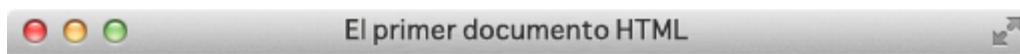
Si la página no pasa correctamente la prueba de validación, se muestra el listado completo de fallos junto con la ayuda necesaria para resolver cada uno de los errores.

Como se observa en la imagen anterior, incluso una página tan sencilla como la portada de Google contiene decenas de errores que impiden considerarla válida. Por lo tanto, la página principal de Google no es una página válida, aunque eso no impide que se vea bien en todos los navegadores y que los usuarios la consideren correcta.

Esta página se ha dejado vacía a propósito

## Capítulo 14

Determinar el código HTML correspondiente a la siguiente página:



El lenguaje HTML es **tan sencillo** que prácticamente se entiende sin estudiar el significado de sus etiquetas principales.

Además de textos en **negrita**, también se pueden poner *en cursiva* o tachados.



**Figura 14.1** Página HTML sencilla que resalta algunas partes del texto

Estructurar y marcar el siguiente texto extraído de la Wikipedia ([http://es.wikipedia.org/wiki/Exploraci%C3%B3n\\_espacial](http://es.wikipedia.org/wiki/Exploraci%C3%B3n_espacial)) para que el navegador lo muestre con el aspecto de la siguiente imagen:



## La exploración espacial

**La exploración espacial** designa los esfuerzos del hombre en estudiar el espacio y sus astros desde el punto de vista científico y de su explotación económica. Estos esfuerzos pueden involucrar tanto seres humanos viajando en naves espaciales como satélites con recursos de telemetría o sondas teleguiadas enviadas a otros planetas (orbitando o aterrizando en la superficie de estos cuerpos celestes).

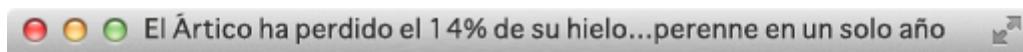
Las personas que pilotan naves espaciales, o son pasajeros en ellas, se llaman astronautas (en Rusia: *cosmonautas*; en China: *taikonautas*). Técnicamente se considera astronauta a todo aquel que emprenda un vuelo sub-orbital (sin entrar en órbita) u orbital a como mínimo 100 km de altitud (considerado el límite externo de la atmósfera).

El cielo siempre ha atraído la atención y los sueños del hombre. Ya en 1634 se publicó la que se considera primera novela de ciencia ficción, *Somnium*, de **Johannes Kepler**, que narra un hipotético viaje a la Luna. Más tarde, en 1865, en una famosa obra de ficción titulada "*De la Terre à la Lune*", **Julio Verne** escribe sobre un grupo de hombres que viajó hasta la Luna usando un gigantesco cañón.

En Francia, **Georges Méliès**, uno de los pioneros del cine, tomaba la novela de Verne para crear "*Le voyage dans la Lune*" (1902), una de las primeras películas de ciencia ficción en la que describía un increíble viaje a la Luna. En obras como "*The War of the Worlds*" (1898) y "*The First Men in The Moon*" (1901), **Herbert George Wells** también se concibieron ideas de exploración del espacio y de contacto con civilizaciones extraterrestres.

Mostrar un menú  
**Figura 14.2** Resultado de estructurar y marcar el texto original

Estructurar y marcar el siguiente texto para que el navegador lo muestre con el aspecto de la siguiente imagen:



# El Ártico ha perdido el 14% de su hielo marino perenne en un solo año

**WASHINGTON.**- El hielo perenne del Ártico se redujo en un 14% durante el último año, al perder **720.000 kilómetros cuadrados**, una superficie superior a la Península Ibérica, según datos de la NASA.

Según el JPL, la pérdida del hielo perenne, que debiera mantenerse durante todo el verano, fue todavía mayor y se acercó a un 50% en el momento en que ese hielo se desplazaba desde el Ártico oriental hacia el oeste.

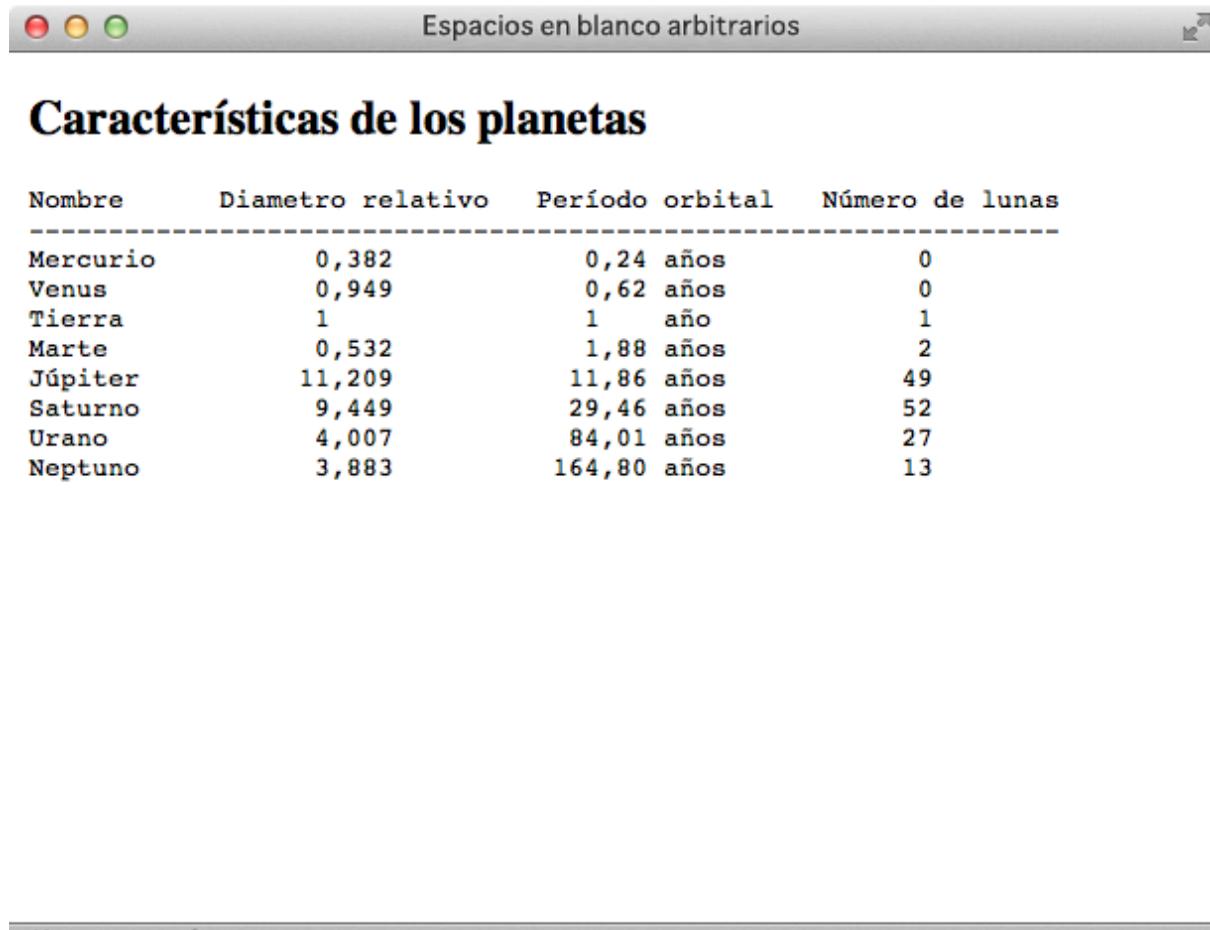
Son Nghiem, investigador del JPL ha declarado que:

*"Los cambios registrados en esos años en el hielo ártico son rápidos y espectaculares. De mantenerse la situación, ésta tendrá un impacto profundo en el ambiente, así como en el transporte marino y el comercio."*

Mostrar un menú

**Figura 14.3** Texto HTML correctamente estructurado y marcado

Determinar el código HTML que corresponde al siguiente documento:



The screenshot shows a table titled "Características de los planetas" (Characteristics of the planets) with the following data:

| Nombre   | Diametro relativo | Período orbital | Número de lunas |
|----------|-------------------|-----------------|-----------------|
| Mercurio | 0,382             | 0,24 años       | 0               |
| Venus    | 0,949             | 0,62 años       | 0               |
| Tierra   | 1                 | 1 año           | 1               |
| Marte    | 0,532             | 1,88 años       | 2               |
| Júpiter  | 11,209            | 11,86 años      | 49              |
| Saturno  | 9,449             | 29,46 años      | 52              |
| Urano    | 4,007             | 84,01 años      | 27              |
| Neptuno  | 3,883             | 164,80 años     | 13              |

**Figura 14.4** Texto HTML con espacios en blanco y nuevas líneas

Determinar el código HTML que corresponde al siguiente documento:



The screenshot shows the following text displayed in a browser:

**Sintaxis de la etiqueta <blockquote>**

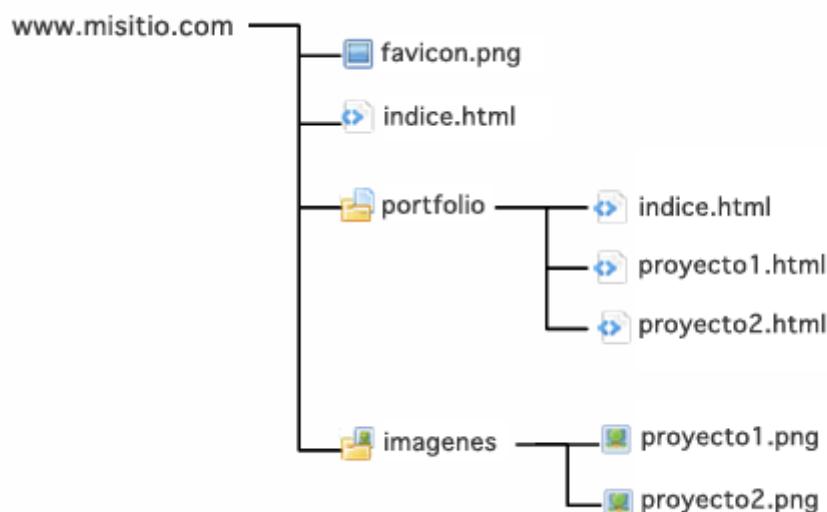
La sintaxis de la etiqueta <blockquote> se muestra a continuación:

<blockquote cite="...dirección original de la cita...">Texto que se cita</blockquote>

Mostrar un menú

**Figura 14.5 Texto HTML que incluye caracteres especiales**

A partir de la estructura de directorios y archivos indicada en la siguiente imagen:

**Figura 14.6 Estructura de archivos y directorios de un sitio web de ejemplo**

1. Crear la siguiente página llamada `indice.html` que sirva como página principal del sitio:

Mi Sitio

# Mi Sitio

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Donec iaculis posuere justo. Nam vel neque. Proin sagittis mauris sit amet nisl. Sed ipsum. Aliquam vitae justo.

## Ultimos proyectos

Etiam consectetuer, mauris vitae cursus scelerisque, dui turpis dignissim justo, et euismod enim odio sit amet erat. Aliquam dui ligula, porttitor eu, facilisis vitae, ornare sed, tortor.

[Acceder a los ultimos proyectos de Mi Sitio](#)

Mostrar un menú

**Figura 14.7** Página principal del sitio web de ejemplo

1. Crear la página de índice del portfolio:



## Ultimos proyectos

  Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Donec iaculis posuere justo. Nam vel neque.

### Proyecto 1

  Etiam consectetuer, mauris vitae cursus scelerisque, dui turpis dignissim justo, et euismod enim odio sit amet erat.

[Ver imagen del Proyecto 1](#)

### Proyecto 2

  Etiam consectetuer, mauris vitae cursus scelerisque, dui turpis dignissim justo, et euismod enim odio sit amet erat.

[Ver imagen del Proyecto 2](#)

**Figura 14.8** Página con la información sobre los proyectos realizados

Enlazar el favicon en todas las páginas del ejercicio 6 y añadir todos los atributos posibles a los enlaces.

Determinar el código HTML que corresponde a la siguiente lista anidada simple:



**Figura 14.9** Ejemplo de lista anidada simple de dos niveles

Determinar el código HTML que corresponde a la siguiente lista anidada compleja:

The screenshot shows a web page with a header bar containing three colored circles (red, yellow, green) and the text "Lista compleja anidada". On the right side of the header is a small icon with a double arrow. Below the header, the word "Menú" is displayed in large, bold, black font. A nested list follows:

- Inicio
- Noticias
  - Recientes
  - Más leídas
  - Más valoradas
- Artículos
  1. XHTML
  2. CSS
  3. JavaScript
  4. Otros
- Contacto

*Email*  
nombre@direccion.com

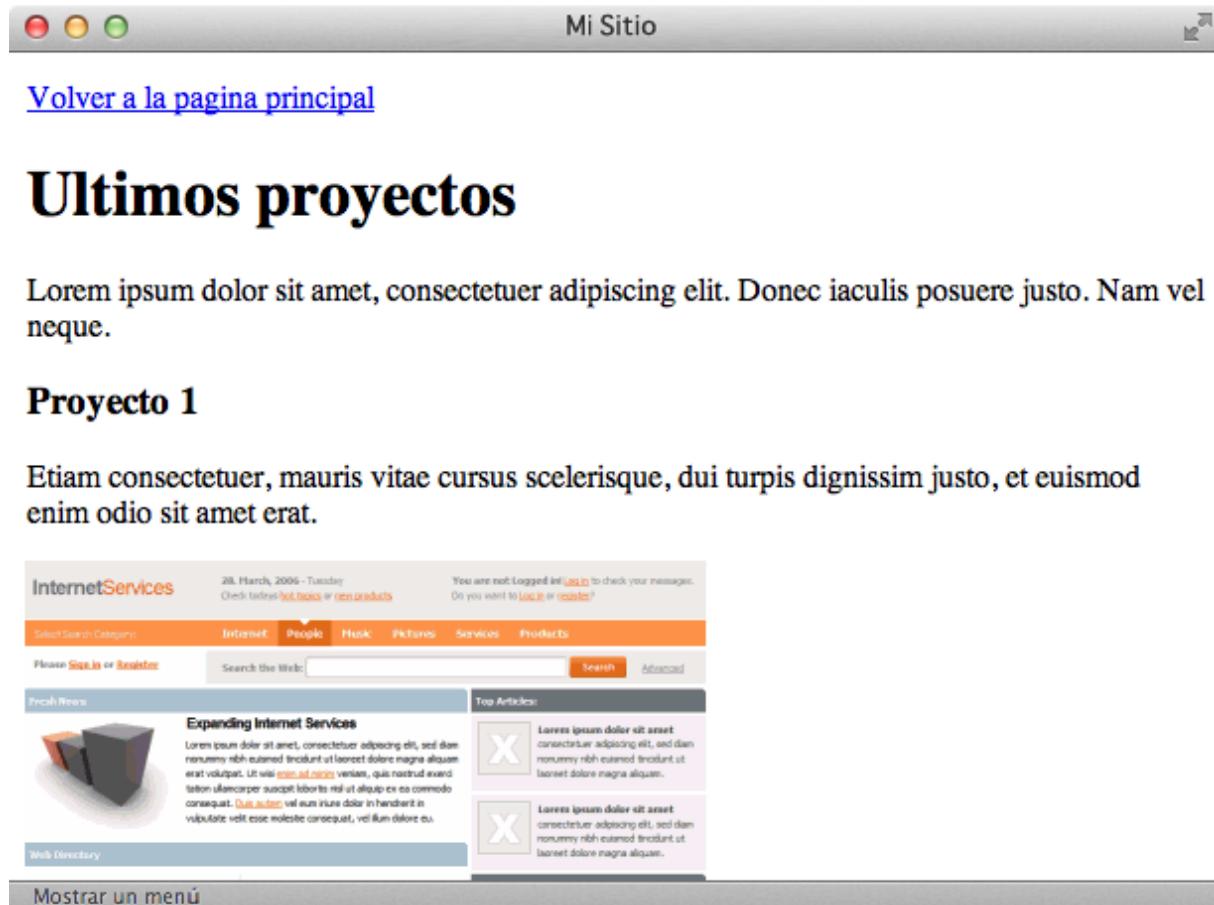
*Teléfono*  
900 900 900

*Fax*  
900 900 900

Mostrar un menú

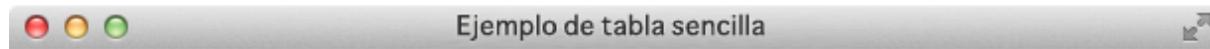
**Figura 14.10** Ejemplo de lista anidada compleja de dos niveles

Modificar la página de índice del portfolio de los ejercicios 6 y 7 para mostrar directamente las imágenes de los proyectos.



**Figura 14.11** Nueva página del portfolio que muestra la imagen de cada uno de los proyectos

Determinar el código HTML necesario para crear la tabla que se muestra en la siguiente imagen:



## Su pedido

| Nombre producto                       | Precio unitario | Unidades | Subtotal      |
|---------------------------------------|-----------------|----------|---------------|
| Reproductor MP3 (80 GB)               | 192.02          | 1        | 192.02        |
| Fundas de colores                     | 2.50            | 5        | 12.50         |
| Reproductor de radio & control remoto | 12.99           | 1        | 12.99         |
| <b>TOTAL</b>                          | -               | 7        | <b>207.51</b> |

Mostrar un menú

**Figura 14.12** Tabla sencilla con celdas de cabecera

Utilizar las celdas de cabecera donde sea necesario y añadir todos los atributos posibles.

Determinar el código HTML necesario para crear la tabla que se muestra en la siguiente imagen. Utilizar las celdas de cabecera donde sea necesario y añadir todos los atributos posibles.

Ejemplo de tabla avanzada

| Imagen | Datos   |
|--------|---|
|        | <u><a href="#">Portátil - 3 GHz - 4 GB RAM</a></u><br><u><a href="#">Comprar: 2.990€ 2.599€</a></u>               |
|        | <u><a href="#">Videocámara - Alta definición 1080p - 60 GB</a></u><br><u><a href="#">Comprar: 1.099€ 999€</a></u> |
|        | <u><a href="#">Televisor - 46" - Full HD</a></u><br><u><a href="#">Comprar: 1.999€ 1.799€</a></u>                 |
|        | <u><a href="#">Móvil - 3G - Wi-Fi - 8 GB</a></u><br><u><a href="#">Comprar: 399€ 349€</a></u>                     |

Mostrar un menú  
**Figura 14.13** Tabla con los resultados de una búsqueda

Determinar el código HTML necesario para crear la tabla que se muestra en la siguiente imagen:

**Ejemplo de tabla compleja**

### Comparativa de reproductores MP3

Tabla comparativa de las características técnicas de los reproductores MP3

|                             |  MP3 mini |                          |                           |  MP3 grande |                            |
|-----------------------------|--|--------------------------|---------------------------|---|----------------------------|
| Capacidad de almacenamiento | 4GB<br>(1.000 canciones)   | 8GB<br>(2.000 canciones) | 16GB<br>(4.000 canciones) | 30GB<br>(7.500 canciones)   | 80GB<br>(20.000 canciones) |
| Colores                     |  |                          |                           |   |                            |
| Pantalla                    | LCD de 3 cm (diagonal) con retroiluminación  |                          |                           | LCD de 6 cm (diagonal) con retroiluminación   |                            |
| Tiempo de carga             | Unas 3 horas   |                          |                           | Unas 4 horas  |                            |
|                             |  |                          |                           | Unas 2 horas para alcanzar el 80% de la capacidad   |                            |

Mostrar un menú

**Figura 14.14** Ejemplo de tabla con una estructura compleja

Emplear las etiquetas `<table>`, `<tr>`, `<td>`, `<th>`, `<caption>` y los atributos `colspan`, `rowspan`, `abbr`, `scope`, `summary`.

Determinar el código HTML necesario para crear el formulario que se muestra en la siguiente imagen:

Rellena tu CV

Nombre

Apellidos

Contraseña

DNI

Sexo  
 Hombre  
 Mujer

Incluir mi foto  nada seleccionado

Suscribirme al boletín de novedades

**Figura 14.15** Formulario con controles de varios tipos

1. Elegir el método más adecuado para el formulario (GET o POST) y cualquier otro atributo necesario.
2. La aplicación que se encarga de procesar el formulario se encuentra en la raíz del servidor, carpeta "php" y archivo "insertar\_cv.php" .
3. El nombre puede tener 30 caracteres como máximo, los apellidos 80 caracteres y la contraseña 10 caracteres como máximo.
4. Asignar los atributos adecuados al campo del DNI.
5. Por defecto, debe estar marcada la casilla de suscripción al boletín de novedades.

Determinar el código HTML necesario para crear el formulario que se muestra en la siguiente imagen:

Rellena tu CV

Datos personales

Provincia

- selecciona -

Fecha de nacimiento

\_\_\_\_\_ de Enero \_\_\_\_\_ de \_\_\_\_\_

Temas de interés

Administración de bases de datos  
Análisis y programación  
Arquitectura  
Calidad  
ERP, CRM, Business Intelligence

Mostrar un menú

**Figura 14.16** Formulario con controles de tipo lista desplegable

Determinar el código HTML necesario para crear el formulario que se muestra en la siguiente imagen:

The screenshot shows a complex product information form titled "Información sobre el producto". The form is divided into two main sections: "Datos básicos" and "Datos económicos".

**Datos básicos:**

- Nombre:** An input field.
- Descripción:** A large text area.
- Foto:** A button labeled "Seleccionar archivo" with the placeholder "nada seleccionado".
- Añadir contador de visitas**

**Datos económicos:**

- Precio:** An input field with a € symbol and a dropdown menu set to 4%.
- Promoción:**
  - Ninguno
  - Transporte gratuito
  - Descuento 5%

**Figura 14.17** Formulario complejo

Determinar el código HTML necesario para crear las siguientes páginas, utilizando el máximo de atributos posibles que añadan semántica al contenido. Crear así mismo, la estructura de directorios necesaria para almacenar imágenes (comunes y de post), hojas de estilos, scripts y los propios documentos HTML.



## [Travel day](#)

October 22, 2011

[John Doe, 3 comments](#)

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec hendrerit felis accumsan turpis pretium tempor. Duis eu turpis nunc, ut euismod nisl. Aliquam erat volutpat. Proin eu eros mollis dui fringilla sodales. Curabitur venenatis tincidunt felis ac congue.

Maecenas at odio dui, sit amet congue sapien. Proin placerat feugiat eros, non mollis quam pharetra at. Duis gravida eleifend ligula nec auctor. Fusce nulla diam, fringilla non ultrices in, iaculis eu tellus. Sed mollis consequat turpis sit amet facilisis. Donec pretium luctus aliquet. Curabitur placerat varius purus vel congue. Aliquam erat volutpat.

Curabitur vitae eros sed turpis sollicitudin mattis. Morbi venenatis pulvinar nunc, at vulputate massa placerat a. Nam et tortor id nisi consequat tempor eget sit amet risus.

Praesent bibendum, velit eu hendrerit porttitor, elit mauris posuere nisl, non pellentesque est leo a quam.

## [I'm going to Prague!](#)

October 17, 2011

[John Doe, 5 comments](#)

Sed ante mi, sagittis sed euismod sit amet, convallis et nibh. Etiam sit amet odio dui, id semper turpis. Mauris risus mauris, imperdiet pulvinar vehicula et, hendrerit vitae dui.

Phasellus ultrices lacus rhoncus purus posuere rutrum. Maecenas mattis eleifend scelerisque. Nulla quam sem, facilisis ac ultrices et, tincidunt eu dolor. Mauris arcu est, porttitor eu blandit nec, pulvinar sed enim. Praesent diam felis, cursus at facilisis eu, mollis ut elit. Praesent rutrum porta euismod. Nulla facilisi. Suspendisse potenti. In auctor ultricies eleifend. Proin erat dolor, malesuada non tempus nec, tincidunt in mi.

### October

[5 Vos vestros servate, meos mihi linquite mores](#)

### September

[30 Ut sementem feceris ita metes](#)

### August

[4 Risu inepto res ineptior nulla est](#)

### July

[6 Vitanda est improba siren desidia](#)

**Figura 14.18 Portada del blog**

The screenshot shows a dark-themed website layout. At the top center is the title "My Weblog". Below it is a subtitle "A lot of effort went into making this effortless.". A horizontal navigation bar at the bottom contains four items: "home", "blog", "gallery", and "about".

## Travel day

October 22, 2011

[John Doe](#), 3 comments

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec hendrerit felis accumsan turpis pretium tempor. Duis eu turpis nunc, ut euismod nisl. Aliquam erat volutpat. Proin eu eros mollis dui fringilla sodales. Curabitur venenatis tincidunt felis ac congue.

Maecenas at odio dui, sit amet congue sapien. Proin placerat feugiat eros, non mollis quam pharetra at. Duis gravida eleifend ligula nec auctor. Fusce nulla diam, fringilla non ultrices in, iaculis eu tellus. Sed mollis consequat turpis sit amet facilisis. Donec pretium luctus aliquet. Curabitur placerat varius purus vel congue. Aliquam erat volutpat.

Curabitur vitae eros sed turpis sollicitudin mattis. Morbi venenatis pulvinar nunc, at vulputate massa placerat a. Nam et tortor id nisi consequat tempor eget sit amet risus.

Praesent bibendum, velit eu hendrerit porttitor, elit mauris posuere nisl, non pellentesque est leo a quam.

| Column 1       | Column 2              | Column 3 | Column 4 | Column 5        | Column 6    |
|----------------|-----------------------|----------|----------|-----------------|-------------|
| Lorem ipsum    | <a href="#">John</a>  | 5/6/2010 | Lorem    | sed in porta    | 125€        |
| Dignissim enim | <a href="#">Admin</a> | 5/6/2010 | Ipsum    | duis nec rutrum | 75€         |
| Maecenas velit | <a href="#">Admin</a> | 5/6/2010 | Lorem    | porta lectus    | 35€         |
| Maecenas velit | <a href="#">Admin</a> | 5/6/2010 | Ipsum    | porta lectus    | 85€         |
| Maecenas velit | <a href="#">Admin</a> | 5/6/2010 | Lorem    | porta lectus    | 50€         |
|                |                       |          |          |                 | Total: 370€ |

### 3 Comments



[stylearua](#) February 22, 2013 at 11:34 am

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec hendrerit felis accumsan turpis pretium tempor. Duis eu turpis nunc, ut euismod nisl.



[stylearua](#) February 22, 2013 at 11:34 am

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec hendrerit felis accumsan turpis pretium tempor. Duis eu turpis nunc, ut euismod nisl.

**Figura 14.19** Vista del post