

## UNIDAD 2 III HTML

---

- Enlaces
- Agrupación de Contenido
- Formularios

### HIPERENLACES

---

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

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

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

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

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

### URL

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

La URL de un recurso tiene dos objetivos principales:

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

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

Si se accede a la página principal de Google, la dirección que muestra el navegador es:

<http://www.google.com>

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

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

Una URL sencilla siempre está formada por las mismas tres partes. Si por ejemplo se considera la siguiente URL:

<http://www.alistapart.com/comments/webstandards2008>

Las partes que componen la URL anterior son:

- **Protocolo (<http://>):** el mecanismo que debe utilizar el navegador para acceder a ese recurso. Todas las páginas web utilizan <http://>. Las páginas web *seguras* (por ejemplo las de los bancos y las de los servicios de email) utilizan <https://> (se añade una letra s).
- **Servidor ([www.alistapart.com](http://www.alistapart.com) ):** simplificando mucho su explicación, se trata del ordenador en el que se encuentra guardada la página que se quiere acceder. Los navegadores son capaces de obtener la dirección de cada servidor a partir de su nombre.
- **Ruta ([comments/webstandards2008](http://www.alistapart.com/comments/webstandards2008)):** *camino* que se debe seguir, una vez que se ha llegado al servidor, para localizar el recurso específico que se quiere acceder.
- Por tanto, las URL no sólo identifican de forma única a cada recurso de Internet, sino que también proporcionan a los navegadores la información necesaria para poder llegar hasta ese recurso.

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

<http://www.alistapart.com/comments/webstandards2008?page=5#42>

Las cinco partes que forman la URL anterior son:

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

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

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

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

| Carácter original   | Carácter codificado | Carácter original | Carácter codificado |
|---------------------|---------------------|-------------------|---------------------|
| /                   | %2F                 | ?                 | %3F                 |
| :                   | %3A                 | @                 | %40                 |
| =                   | %3D                 | &                 | %26                 |
| "                   | %22                 | \                 | %5C                 |
| '                   | %27                 | ~                 | %7E                 |
| {espacio en blanco} | %20                 |                   | %7C                 |

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

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

#### Carácter original Carácter codificado

|   |     |
|---|-----|
| ñ | %F1 |
| á | %E1 |
| é | %E9 |
| í | %ED |
| ó | %F3 |
| ú | %FA |
| ç | %E7 |

#### Carácter original Carácter codificado

|   |     |
|---|-----|
| Ñ | %D1 |
| Á | %C1 |
| É | %C9 |
| Í | %CD |
| Ó | %D3 |
| Ú | %DA |
| Ç | %C7 |

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

**<!-- URL problemática -->**

<http://www.ejemplo.com/estaciones/otoño.html>

**<!-- URL correcta -->**

<http://www.ejemplo.com/estaciones/oto%F1o.html>

**<!-- URL problemática -->**

<http://www.ejemplo.com/ruta/nombre página.html>

**<!-- URL correcta -->**

<http://www.ejemplo.com/ruta/nombre%20p%E1gina.html>

## Enlaces absolutos y enlaces relativos

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

*Cuando se pincha sobre algún enlace y el navegador abandona el sitio web para acceder a páginas que se encuentran en otros sitios, estos enlaces se conocen como "enlaces externos".*

*Sin embargo, la mayoría de enlaces de un sitio web apuntan a páginas del propio sitio web, por lo que se denominan "enlaces internos".*

*Además de internos/externos, la otra característica que diferencia a los enlaces (y por tanto, también a las URL) es si el enlace es absoluto o relativo. Las URL absolutas incluyen todas las partes de la URL (protocolo, servidor y ruta) por lo que no se necesita más información para obtener el recurso enlazado.*

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

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

*Imagina que dispones de una página publicada en <http://www.ejemplo.com/ruta1/ruta2/pagina1.html> y quieres incluir en ella un enlace a otra página que se encuentra en <http://www.ejemplo.com/ruta1/ruta2/pagina2.html>. Como las URL identifican de forma única a los recursos de Internet y proporcionan la información necesaria para llegar hasta ellos, el enlace debería utilizar la URL completa de la segunda página. Sin embargo como la segunda URL utiliza el mismo protocolo y se encuentra en el mismo servidor que la página origen la URL relativa puede prescindir de esas partes:*

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

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

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

- *Se debe determinar la URL absoluta a partir de la URL relativa para poder cargar el recurso enlazado.*
- *A la URL relativa le falta el protocolo y el servidor, por lo que se supone que coincide con los de la página origen ([http://, www.ejemplo.com](http://www.ejemplo.com)).*

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

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

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

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

|                  |  |
|------------------|--|
| Origen           | <code>http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</code>                     |
| Recurso enlazado | Página web llamada <code>pagina2.html</code> y que se encuentra en el mismo directorio |
| URL absoluta     | <code>http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina2.html</code>                     |
| URL relativa     | <code>pagina2.html</code>  |

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

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

La URL relativa debe indicar de alguna manera que es necesario *subir* un nivel en la jerarquía de directorios para llegar hasta el recurso.

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

|                  |  |
|------------------|--|
| Origen           | <code>http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</code>   |
| Recurso enlazado | Página web llamada <code>pagina2.html</code> y que se encuentra en el directorio superior llamado <code>ruta2</code> |
| URL absoluta     | <code>http://www.ejemplo.com/ruta1/ruta2/pagina2.html</code>   |
| URL relativa     | <code>../pagina2.html</code>   |

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

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

|                  |  |
|------------------|--|
| Origen           | <a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</a>  |
| Recurso enlazado | Página web llamada <a href="#">pagina2.html</a> y que se encuentra en el directorio superior llamado <a href="#">ruta1</a> |
| URL absoluta     | <a href="http://www.ejemplo.com/ruta1/pagina2.html">http://www.ejemplo.com/ruta1/pagina2.html</a>                          |
| URL relativa     | <a href="#">../../pagina2.html</a>   |

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

|                  |  |
|------------------|--|
| Origen           | <a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</a>                                  |
| Recurso enlazado | Página web llamada <a href="#">pagina2.html</a> y que se encuentra en un directorio llamado <a href="#">ruta4</a> que se encuentra en la raíz del servidor |
| URL absoluta     | <a href="http://www.ejemplo.com/ruta4/pagina2.html">http://www.ejemplo.com/ruta4/pagina2.html</a>  |
| URL relativa     | <a href="#">../../../../ruta4/pagina2.html</a>   |

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

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

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

Si el recurso enlazado se encuentra en algún directorio inferior al que se encuentra el origen, sólo es necesario indicar el nombre de los directorios a los que debe entrar el navegador.

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

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



|                  |   |
|------------------|---|
| Origen           | <a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</a>   |
| Recurso enlazado | Página web llamada pagina2.html y que se encuentra en un directorio inferior llamado ruta6 que está dentro del directorio ruta5 y que a su vez está dentro del directorio ruta4 |
| URL absoluta     | <a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/ruta4/ruta5/ruta6/pagina2.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/ruta4/ruta5/ruta6/pagina2.html</a>                   |
| URL relativa     | <a href="#">ruta4/ruta5/ruta6/pagina2.html</a>  |

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

Cuando el origen y el destino de un enlace se encuentran muy alejados (pero en el mismo servidor) las URL relativas se pueden complicar en exceso. Aunque es posible utilizar ../ para subir por la jerarquía de directorios y se puede entrar en cualquier directorio indicando su nombre, las URL relativas que se obtienen son demasiado largas y complicadas.

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

|                  |   |
|------------------|---|
| Origen           | <a href="http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html">http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</a> |
| Recurso enlazado | Página web llamada pagina2.html y que se guarda en un directorio llamado ruta7 que se encuentra en la raíz del servidor   |
| URL absoluta     | <a href="http://www.ejemplo.com/ruta7/pagina2.html">http://www.ejemplo.com/ruta7/pagina2.html</a>                         |
| URL relativa     | <a href="#">/ruta7/pagina2.html</a>   |

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.

### Enlaces en HTML

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

| Etiqueta                   | Atributo      | Valor                               | Significado   |
|----------------------------|---------------|-------------------------------------|---|
| <b>&lt;a&gt;&lt;/a&gt;</b> | <b>href</b>   | <b>URL</b>                          | Indica la URL de la página que se cargará   |
|                            | <b>name</b>   | <b>Nombre</b>                       | Establece el nombre del ancla al que llevará el enlace..  |
|                            | <b>target</b> | <b>_blank, _self, _top, _parent</b> | Indica al navegador dónde debe abrir la nueva página : En una ventana nueva ( <b>_blank</b> ), en la misma ventana ( <b>_self</b> ), en el marco primario ( <b>_parent</b> ) o en toda la ventana ( <b>_top</b> ) |

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

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

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

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

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

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

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

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

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

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

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

## Enlaces dentro de la misma página “anclas”

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

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

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

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

...



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

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

...

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

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

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

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

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

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

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

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

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

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

Los enlaces directos a secciones también funcionan con el atributo id de cualquier elemento.

El siguiente ejemplo es equivalente al ejemplo anterior:

```
<h1 id="inicio">Título de la página</h1>
```

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

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

```
<p>Pellentesque malesuada. In in lacus. Phasellus erat erat, lacinia a, convallis eu,nonummy et, odio. Aenean urna elit, ultrices id, placerat varius, facilisis eget,dolor.</p><a href="#inicio">Volver al inicio de la página</a>...
```

El nombre de la sección que se indica después del símbolo # puede utilizar el valor de los atributos id de cualquier elemento. De hecho, se recomienda utilizar los atributos id

de los elementos ya existentes en la página en vez de crear "enlaces vacíos" de tipo `<a name="nombre_seccion"></a>`.

### Enlace a un e-mail

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

Solicita más información

```
</a>
```

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

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

```
<!-- Envío del correo electrónico a varias direcciones a la vez -->
```

```
<a href="mailto:nombre@direccion.com,otro_nombre@direccion.com">Solicita más información</a>
```

```
<!-- Añadir un "asunto" inicial al correo electrónico -->
```

```
<a href="mailto:nombre@direccion.com?subject=Solicitud de más información">Solicita más información</a>
```

```
<!-- Añadir un texto inicial en el cuerpo del correo electrónico -->
```

```
<a href="mailto:nombre@direccion.com?body=Estaría interesado en solicitar más información sobre sus productos">Solicita más información</a>
```

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

## AGRUPACIÓN DE CONTENIDO <div> Y <span>

La diferencia fundamental entre ambos elementos está en que el DIV forma estructuras más grandes que los SPAN. Los DIV suelen estar contenidos en varias líneas.

### <div>

La etiqueta **<div>** permite agrupar los elementos que forman cada zona o división

de una página web.

La etiqueta `<div>` define una división. Esta etiqueta permite agrupar varios elementos de bloque (párrafos, encabezados, listas, tablas, divisiones, etc). En principio, los navegadores no muestran nada especial cuando se crea una división, salvo que se dé formato a la división con la hoja de estilo.

Hay gente y programas que llaman "capas" a las divisiones.

Una división no puede insertarse dentro de una etiqueta en-línea (`<strong>`, `<em>`, etc.) o de un bloque de texto (párrafo `<p>`, encabezado `<h1> ... <h6>`, dirección `<address>`, pre-formateado `<pre>`, lista, etc), pero si puede insertarse dentro de una tabla, de un bloque de cita `<blockquote>` o de una división `<div>`.

| Etiqueta                             | Atributo     | Valor                               | Significado  |
|--------------------------------------|--------------|-------------------------------------|--|
| <code>&lt;div&gt;&lt;/div&gt;</code> | <b>id</b>    | <b>identificador</b>                | El atributo <code>id</code> designa un nombre a un elemento ( <code>id="name"</code> ). No puede haber otro con el mismo nombre en el mismo documento. Un <code>id</code> puede tener múltiples funciones, entre ellas, un identificador de estilo, un ancla para los enlaces, un nombre de objeto, etc. |
|                                      | <b>align</b> | <b>justify, center, left, right</b> | Justificación del texto o de los objetos contenidos  |

### Ejemplo div.html

```

<html>
<head>
  <title>Untitled Document</title>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
</head>

<body>
<div id="primera" align="justify" >
  <p>El hombre es fuego; la mujer, estopa; llega el diablo y sopla.</p>
  <p>Para el amor y la muerte, no hay cosa fuerte.</p>
  <p>Viejo el pajar, malo de encender y peor de apagar.</p>
</div>

<div align="center">
  <p>Enseñas sin saber? Como no sea el culo, no sé qué.</p>
  <p>Practicar hace maestro; que no leer en el cuaderno.</p>
  <p>Lo que natura no da, Salamanca no presta</p>
</div>

</body>

```

&lt;/html&gt;

## <span>

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 la hoja de estilo.

### Ejemplo span.html

---

&lt;html&gt;

&lt;head&gt;

&lt;title&gt;Untitled Document&lt;/title&gt;

&lt;meta http-equiv="Content-Type" content="text/html; charset=utf-8"&gt;

&lt;/head&gt;

&lt;body&gt;

&lt;p&gt;El primer servidor web de la historia se instaló en el CERN en

&lt;span&gt;diciembre de 1990&lt;/span&gt; (

&lt;a href="http://en.wikipedia.org/wiki/Image:First\_Web\_Server.jpg"&gt;foto&lt;/a&gt;&lt;/p&gt;

&lt;/body&gt;

&lt;/html&gt;

## CREACIÓN DE FORMULARIOS

---

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

HTML 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>. Si se considera el formulario que muestra la siguiente imagen:



El código HTML necesario para definir el formulario anterior se muestra a continuación:

```
<html>
<head>
  <title>Ejemplo de formulario sencillo</title>
</head>
<body>

  <h3>Formulario muy sencillo</h3>

  <form action="http://www.librosweb.es/maneja_formulario.php" method="post">
    Escribe tu nombre:
    <input type="text" name="nombre" value="">

    <br/>

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

</body>
</html>
```

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).

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 envían 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 que 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.

## Elementos de formulario

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:

| Etiqueta | Atributo  | Valor   | Significado   |
|----------|-----------|---|---|
| <input>  | alt       | Texto   | Descripción del control   |
|          | checked   | checked   | Para los controles checkbox y radiobutton permite indicar qué opción aparece preseleccionada  |
|          | disabled  | disabled  | El control aparece deshabilitado y su valor no se envía al servidor junto con el resto de datos.  |
|          | maxlength | Número  | Determina el número máximo de caracteres que se pueden insertar en un campo de texto.   |
|          | name      | Texto   | Atributo obligatorio. Define un nombre unívoco para el elemento.  |
|          | readonly  | readonly  | El contenido del control no se puede modificar  |
|          | size      | Número  | Define el tamaño del elemento. Para los campos de texto se refiere al número de caracteres, en el resto de controles se refiere a su tamaño en pixels.  |
|          | src       | URL   | Para el control que permite crear botones con imágenes, indica la URL de la imagen que se emplea como botón de formulario.  |
|          | type      | text, password, checkbox, radio, submit, reset, file, hidden, image, button | Indica el tipo de control que se incluye en el formulario   |
|          | value     | Texto   | Valor inicial del control. Si type="button", "reset" o "submit", define el texto del botón.<br><br>Si type="checkbox" o "radio" este atributo es obligatorio. Define el resultado del elemento input al ser pulsado. El |



|  |  |  |   |
|--|--|--|---|
|  |  |  | resultado se envía a la URL indicada en <form><br>Si type="hidden", "text" o "password" define el valor por defecto del elemento. |
|--|--|--|---|

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

### Caja de texto: <input type="text" />, <input type="password" />

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. Existen cajas de una o varias líneas.

Nombre

### Caja de texto de una sola línea: <input type="text" /> y <input type="password" />

Las cajas de texto de una sola línea se crean mediante la etiqueta <input /> cuyo atributo type tiene el valor text.

A continuación se muestra el código HTML correspondiente al ejemplo anterior:

**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">**

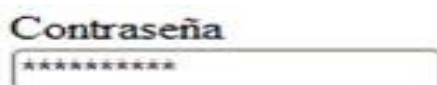
Si no se especifica un tamaño, el navegador muestra el cuadro de texto con un tamaño predeterminado. El atributo `size` permite establecer el tamaño, en caracteres, con el que se muestra el cuadro de texto. Su uso es imprescindible en muchos formularios, en los que algunos campos como la dirección deben 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.

### Cuadro de contraseña

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.

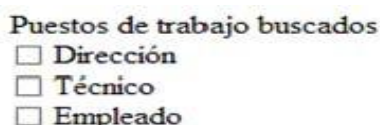


**Contraseña** `<br><input type="password" name="contrasena" value="">`

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.

## Checkbox

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.



**Puestos de trabajo buscados** `<br>`

`<input name="puesto_directivo" 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.

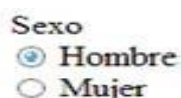
Si se quiere mostrar un *checkbox* seleccionado por defecto, se utiliza el atributo checked. Si el valor del atributo es checked, el *checkbox* se muestra seleccionado. En cualquier otro caso, el *checkbox* permanece sin seleccionar.

`<input type="checkbox" checked="checked"... >` **Checkbox seleccionado por defecto**

## Radiobutton

---

Los controles de tipo radiobutton son similares a los controles de tipo checkbox, pero presentan una diferencia muy importante: son mutuamente excluyentes. **Lo botones radio que tienen el mismo atributo name forman un grupo, es decir, que si se marca uno de ellos se desmarca automáticamente el resto.**



**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. Recordar que el atributo name 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.

Si uno de los botones tiene el atributo **checked** con el valor checked, el botón aparece marcado.

## Botones de envío y reseto de formulario

---

### **Botón de envío**

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



```
<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** el texto que muestra el botón. Si no se establece el atributo value, el navegador muestra el texto predefinido Enviar consulta.

### **Botón de reseteo de formulario**

Se trata de un botón especial que borra todos los datos introducidos por el usuario y devuelve el formulario a su estado original:



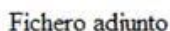
```
<input type="reset" name="limpiar" value="Borrar datos del formulario" >
```

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 se utiliza este atributo, el navegador muestra el texto predefinido del botón, que en este caso es Restablecer.

## **Ficheros adjuntos**

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.




```
Fichero adjunto<br><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>

## Campos ocultos

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

\*Los campos ocultos  
 no se ven en pantalla

**<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.

## Botón de imagen

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.

## Botón

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:

Guardar Cambios

**<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.

## Otros elementos de formulario

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 desplegadas y las áreas de texto disponen de sus propias etiquetas (`<select>` y `<textarea>` respectivamente).

### Áreas de texto

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:



El código HTML del ejemplo anterior se muestra a continuación:

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

La definición formal de la etiqueta `<textarea>` es:

| Etiqueta                                       | Atributo | Valor  | Significado   |
|--|----------|--------|---|
| <code>&lt;textarea&gt;&lt;/textarea&gt;</code> | cols     | Número | Atributo obligatorio. Indica el número de columnas visibles |

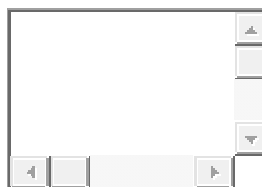


|  |          |        |  |
|--|----------|--------|--|
|  | rows     | Número | Atributo obligatorio. Indica el número de filas visibles           |
|  | name     | Nombre | Especifica el nombre unívoco del elemento                          |
|  | readonly |        | Indica que el usuario no puede modificar el contenido del elemento |

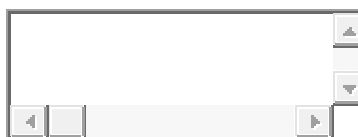
Los atributos más utilizados en las etiquetas `<textarea>` son los que controlan su anchura y altura. La anchura del área de texto se controla mediante el atributo `cols`, que indica las columnas o número de caracteres que se podrán escribir como máximo en cada fila. La altura del área de texto se controla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

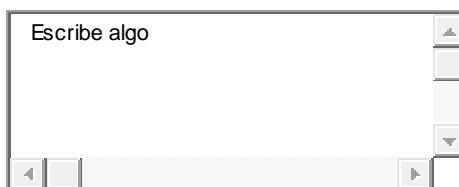
```
<textarea name="texto" rows="4"
cols="20"></textarea>
```



```
<textarea name="texto" rows="3"
cols="30"></textarea>
```



```
<textarea name="texto" rows="4" cols="40">Escribe
algo</textarea>
```



### Listas Desplegables

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



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 se muestra a continuación:

```
<span>Sistema operativo</span> <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>

<span>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>
```

```

<span>Sistema operativo</span> <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>**:

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:

```

<span>Sistema operativo</span> <br>
<select 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.

| Etiqueta                             | Atributo        | Valor           | Significado  |
|--------------------------------------|-----------------|-----------------|--|
| <b>&lt;option&gt;&lt;/option&gt;</b> | <b>selected</b> | <b>Selected</b> | Indica que la opción debe aparecer seleccionada por defecto.   |
|                                      | <b>value</b>    | <b>Texto</b>    | Define el valor de la opción que debe ser enviada al servidor. |

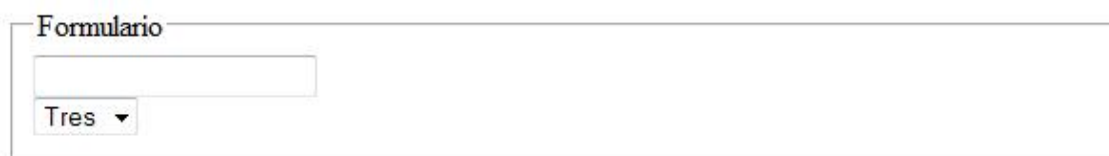
| Etiqueta                             | Atributo        | Valor           | Significado   |
|--------------------------------------|-----------------|-----------------|---|
| <b>&lt;select&gt;&lt;/select&gt;</b> | <b>name</b>     | <b>Nombre</b>   | Especifica un nombre unívoco para el menú desplegable.        |
|                                      | <b>multiple</b> | <b>Multiple</b> | Indica que es posible seleccionar diversas opciones a la vez. |
|                                      | <b>size</b>     | <b>Número</b>   | Define el número de opciones visibles en el menú desplegable  |

## Grupos de controles: **<fieldset>**

La etiqueta **<fieldset>** permite agrupar un conjunto de controles. Los navegadores muestran una caja alrededor de cada grupo de controles.

La etiqueta **<legend>** permite añadir una leyenda al **<fieldset>**. Los navegadores muestran la leyenda sobre el borde que rodea el grupo de controles.

```
<fieldset>
  <legend>Formulario</legend>
  <input type="text" name="texto"/><br/>
  <select name="menu">
    <option>Uno</option>
    <option>Dos</option>
    <option selected="selected">Tres</option>
  </select>
</fieldset>
```



## Etiqueta label

La etiqueta **<label>** permite asociar un control con un texto, de manera que mejore la accesibilidad de los formularios.

La asociación entre el control y la etiqueta **<label>** puede ser implícita o explícita. Se dice que la relación es implícita cuando el control se encuentra en el interior de la etiqueta. Se dice que la relación es explícita cuando la etiqueta **<label>** contiene el atributo **for**, que indica el control afectado (el control tiene entonces que tener establecido el atributo **id**).

Por ejemplo, en el caso de una casilla de verificación, la etiqueta **<label>** permite que la casilla se marque o desmarque haciendo clic en el texto, como se muestra en los ejemplos siguientes:

```
<input type="checkbox" name="casilla" />Casilla 1
```

```
<label><input type="checkbox" name="casilla" />Casilla 1</label>
```

```
<input type="checkbox" name="casilla" id="casilla1" />
<label for="casilla1">Casilla 1</label>
```

