Unidad Didáctica 4

Formularios y otros en XHTML 1.0

1. Formularios

El estándar HTML/XHTML permite crear formularios para que los usuarios interactúen con las aplicaciones web.

1.1. Formularios básicos

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

<form>	Formulario
Atributos comunes	básicos, i18n y eventos
Atributos específicos	▪ `action = "url"` - Indica la URL que se encarga de procesar los datos del formulario ▪ `method = "POST o GET"` - Método HTTP empleado al enviar el formulario ▪ `enctype = "application/x-www-form-urlencoded o multipart/form-data"` - Tipo de codificación empleada al enviar el formulario al servidor (sólo se indica de forma explícita en los formularios que permiten adjuntar archivos) ▪ `accept = "tipo_de_contenido"` - Lista separada por comas de todos los tipos de archivos aceptados por el servidor (sólo para los formularios que permiten adjuntar archivos) ▪ Otros: accept-charset, onsubmit, onreset
Tipo de elemento	Bloque
Descripción	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 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.

1.2. Elementos de formulario

Los elementos de formulario como botones, cuadros de texto, etc también se denominan *"campos de formulario"* y/o *"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>	Control de un formulario
Atributos comunes	básicos, i18n, eventos y foco
Atributos específicos	▪ type = "text | password | checkbox | radio | submit | reset | file | hidden | image | button" - Indica el tipo de control que se incluye en el formulario ▪ name = "texto" - Asigna un nombre al control (es imprescindible para que el servidor pueda procesar el formulario) ▪ value = "texto" - Valor inicial del control ▪ size = "unidad_de_medida" - 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) ▪ maxlength = "numero" - Máximo número de caracteres para los controles de texto y de password ▪ 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 ▪ readonly = "readonly" - El contenido del control no se puede modificar ▪ src = "url" - Para el control que permite crear botones con imágenes, indica la URL de la imagen que se emplea como botón de formulario ▪ alt = "texto" - Descripción del control
Tipo de elemento	En línea y etiqueta vacía
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>.

1.2.1. Cuadro de texto

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:

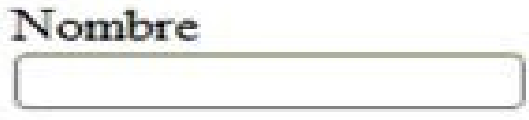

Figura . Ejemplo de etiqueta input (type=text)

- 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.

1.2.2. 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.

Figura . Ejemplo de etiqueta input (type=password)

```
<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.

1.2.3. 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.

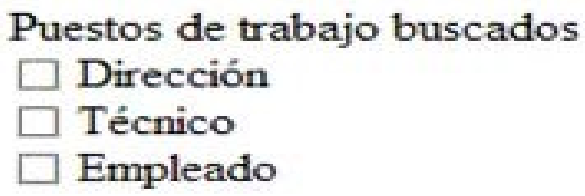

Figura. Ejemplo de etiqueta input (type=checkbox)

- 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 por defecto
```

1.2.4. Radiobutton

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 . Ejemplo de etiqueta input (type=radio)

- 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.

1.2.5. Botón de envío de formulario

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

Figura . Ejemplo de etiqueta input (type=submit)

- 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.

1.2.6. Botón de reseteo del formulario

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 . Ejemplo de etiqueta input (type=reset)

- 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.

1.2.7. 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.

Figura . Ejemplo de etiqueta input (type=file)

- 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>
```

1.2.8. Campos ocultos

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

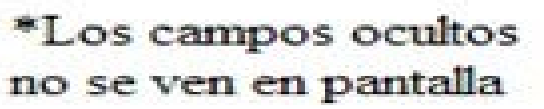

Figura . Ejemplo de etiqueta input (type=hidden)

```
<input type="hidden" name="url_previa" value="/articulo/primero.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.

1.2.9. 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:

Figura. Ejemplo de etiqueta input (type=image)

```
<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.

1.2.10. 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:

Figura . Ejemplo de etiqueta input (type=button)

```
<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.

1.3. Formularios avanzados

Utilizando solamente las etiquetas <form> y <input> es posible diseñar la mayoría de formularios de las aplicaciones web. No obstante, HTML define algunos elementos adicionales para mejorar la estructura de los formularios creados.

La siguiente imagen muestra un formulario que agrupa sus elementos y añade etiquetas a cada campo para mejorar su estructura:

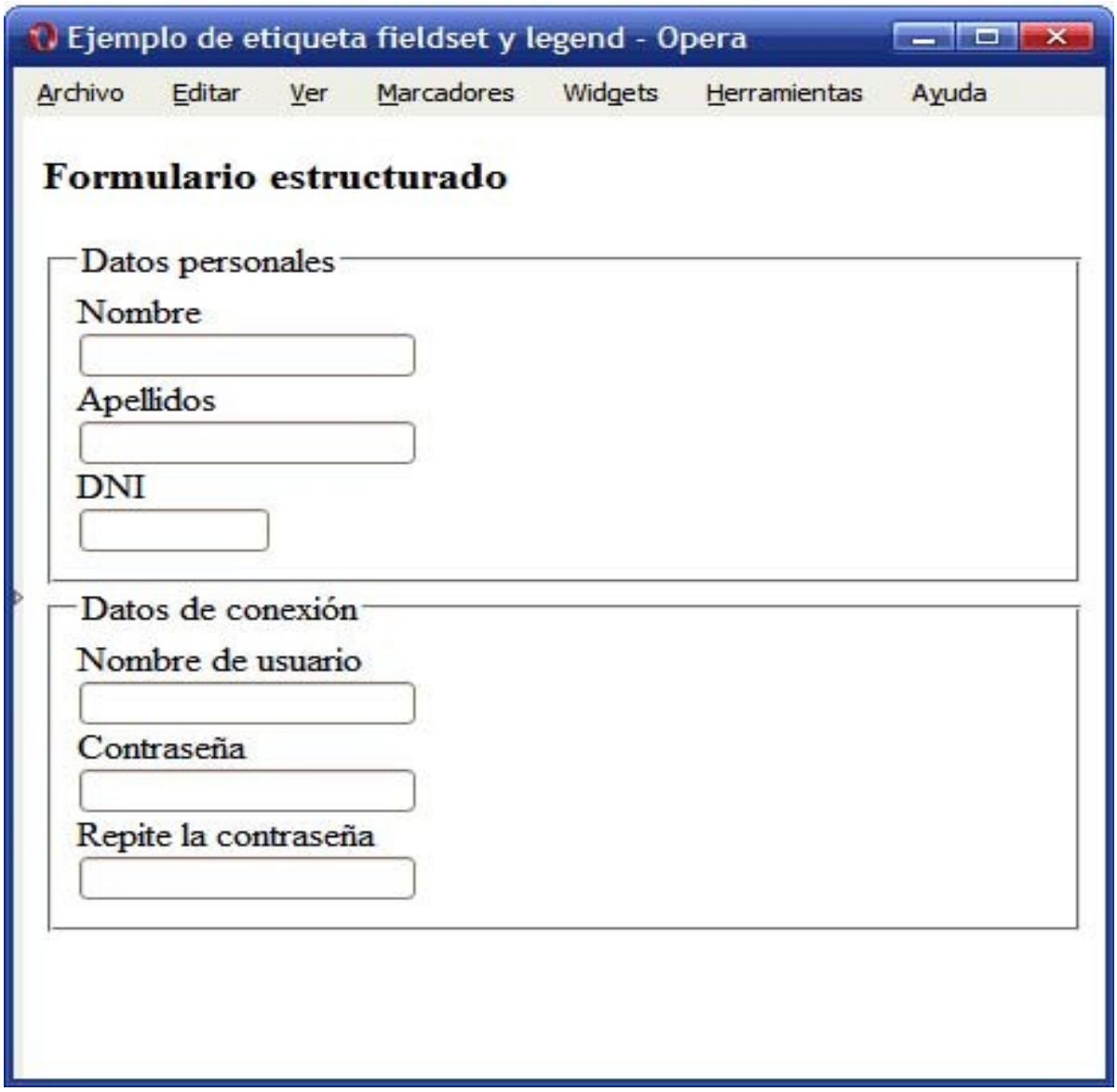

Figura . Ejemplo de uso de las etiquetas fieldset y legend

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

<fieldset>	Agrupación de campos
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Se emplea para agrupar de forma lógica varios campos de un formulario

<legend>	Título o leyenda de un fieldset
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	En línea
Descripción	Se emplea para definir el título o leyenda de un conjunto de campos de formulario agrupados con la etiqueta fieldset

- 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:

<label>	Título o leyenda de un campo de formulario
Atributos comunes	básicos, i18n y eventos
Atributos específicos	▪ for = "id_de_elemento" - Indica el ID del campo del formulario para el que este elemento es su título ▪ Otros: accesskey, onfocus y onblur
Tipo de elemento	En línea
Descripción	Se emplea para definir el título o leyenda de los campos definidos en un formulario

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 radiobutton y checkbox.

1.4. 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 deplegables 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:

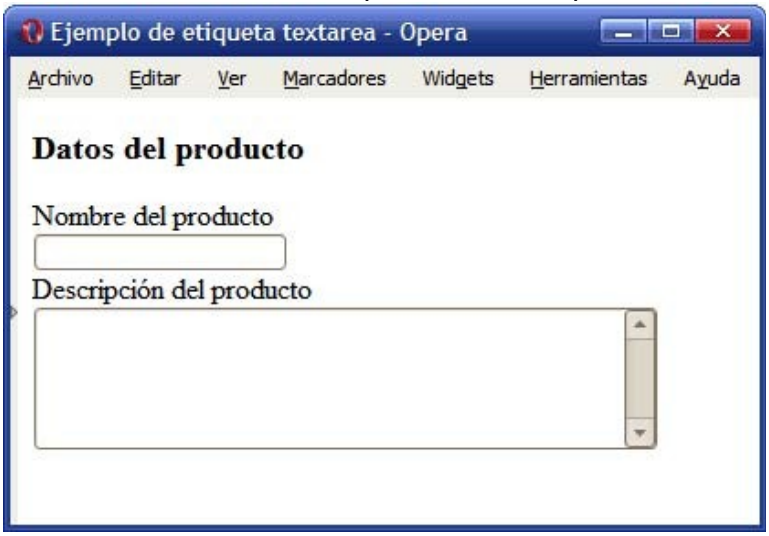

Figura. Ejemplo de uso de la etiqueta textarea

La definición formal de la etiqueta <textarea> es:

<textarea>	Área de texto
Atributos comunes	básicos, i18n, eventos y foco
Atributos específicos	• rows = "numero" - Número de filas de texto que mostrará el textarea • cols = "numero" - Número de caracteres que se muestran en cada fila del textarea • Otros: name, disabled, readonly, onselect, onchange, onfocus, onblur
Tipo de elemento	En línea
Descripción	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:

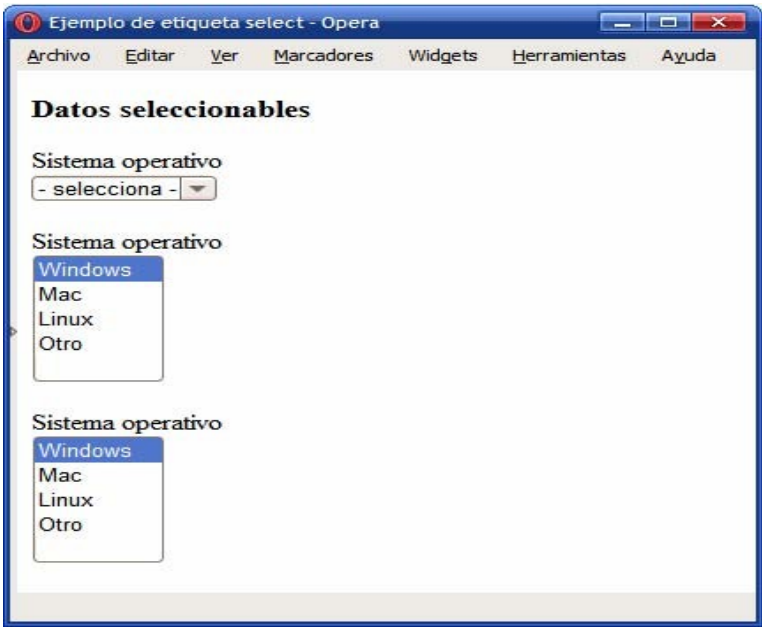

Figura . Ejemplo de uso de la etiqueta select

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.

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>:

<select>	Lista desplegable
Atributos comunes	básicos, i18n y eventos
Atributos específicos	▪ `size = "numero"` - Número de filas que se muestran de la lista (por defecto sólo se muestra una) ▪ `multiple = "multiple"` - Si se incluye, se permite seleccionar más de un elemento ▪ Otros: name, disabled, onchange, onfocus, onblur
Tipo de elemento	En línea
Descripción	Se emplea para incluir una lista desplegable en un formulario

<option>	Elemento de una lista desplegable
Atributos comunes	básicos, i18n y eventos
Atributos específicos	▪ `selected = "selected"` - Indica si el elemento aparece seleccionado por defecto al cargarse la página ▪ `value = "texto"` - El valor que se envía al servidor cuando el usuario elige esa opción ▪ Otros: label, disabled
Tipo de elemento	-
Descripción	Se emplea para definir cada elemento de una lista desplegable

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 selectede 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:

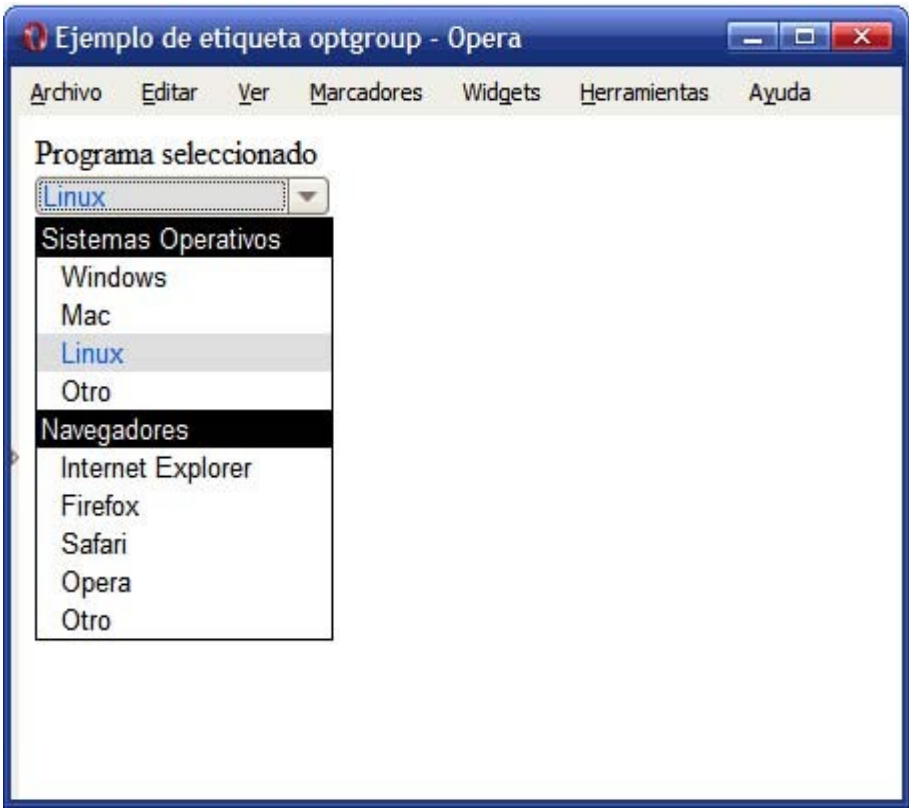

Figura . Ejemplo de uso de la etiqueta optgroup

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

<optgroup>	Agrupación de elementos de una lista desplegable
Atributos comunes	básicos, i18n y eventos
Atributos específicos	▪ `label = "texto"` - Texto que se muestra como título de la agrupación de opciones ▪ Otros: disabled, selected
Tipo de elemento	-
Descripción	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.

3. Otras etiquetas importantes

3.1. JavaScript

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.

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

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>.

<noscript>	Sin soporte de scripts
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Define un mensaje alternativo que se muestra al usuario cuando su navegador no soporta la ejecución de scripts

3.2. CSS

Elementos para uso de hojas

Existen algunos elementos ***HTML*** que tienen una especial utilidad de cara a las hojas de estilo, como ***DIV*** y ***SPAN***. Son elementos de bloque y línea respectivamente que no imponen un significado presentacional. Son ideales para crear estructura artificial que será referenciada desde CSS.

Ejemplo:

```
<H1>Título</H1>
<DIV class="resumen"> <P>----------</P> <P>----------</P>
</DIV> <P>-----
```

Con este elemento ***DIV*** conseguimos que el conjunto que forman los dos primeros párrafos estén recogidos bajo un único elemento.

```
<P>Se ha estrenado <SPAN class="pelicula">Millenium</SPAN> en TV</P>
```

Con este elemento ***SPAN*** hacemos que una o varias palabras sean recogidas en un elemento ***HTML*** que podrá ser referenciado posteriormente.

Otro elemento importante de cara a las hojas de estilo es LINK que nos permite enlazar el documento a una hoja de estilo y establecer el lenguaje de hojas de estilo por defecto.

EJEMPLO:

```
<LINK href="hoja.css" rel="stylesheet" type="text/css">
```

indica que el documento actual debería visualizarse según las reglas que se encuentran en la hoja de estilo (***stylesheet***) ***hoja.css***, que está descrita con el lenguaje ***CSS***.

EJEMPLO:

```
<META http-equiv="Content-Style-Type" content="text/css">
```

indica que lenguaje usado para indicar reglas de estilo (incrustadas en el propio documento ***HTML***) será el ***CSS***.

Atributos para uso de hojas y scripts

Algunos de los atributos más utilizados en la creación de páginas web son id, class y style. Los tres atributos están muy relacionados con CSS, sobre todo class y style.

- El atributo **id** se emplea para asignar un identificador único a cada elemento de la página, lo que es útil tanto para aplicar estilos CSS a ese elemento como para programar aplicaciones con JavaScript.
- Por otra parte, el atributo **class** se emplea para definir la clase CSS que se aplica a un elemento. La clase CSS es el nombre de un conjunto de estilos que se definen en la hoja de estilos y que se quieren aplicar a un elemento:

```
<p class="resumen">Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas at diam id enim viverra semper. Nulla id urna. Donec sodales.</p>
```

El párrafo del ejemplo anterior se muestra por pantalla con el aspecto definido por el conjunto de estilos llamado resumen y que se define en la hoja de estilos CSS enlazada por la página web.

- El atributo **style** se emplea para definir estilos CSS directamente sobre los elementos HTML, tal y como se muestra en el siguiente ejemplo:

```
<p>Algunas palabras de esta frase se muestran de <span style="color:red">color rojo</span></p>
```

No se debe confundir el atributo style con la etiqueta **<style>** que se explicó anteriormente. La etiqueta <style> se utiliza para incluir bloques de código CSS:

```
<head>
...
<style type="text/css"> span {color:red;}
   </style>
</head>
```

3.3. Iframes

Aunque su uso no es muy común, la etiqueta <iframe> puede ser muy útil en determinadas ocasiones, ya que permite insertar un documento HTML dentro de otro documento HTML. Un iframe puede considerarse como un *agujero* que se abre en una página web y a través del cual se muestra otra página web.

En ocasiones se utiliza para mostrar contenidos externos al sitio web como si fueran parte del mismo sitio. Otra veces se emplea para incluir una aplicación común a varios sitios web de una misma empresa.

La página principal de **Google Analytics** emplea un <iframe> para incluir en un pequeño recuadro la página correspondiente a la validación de usuario.

El siguiente ejemplo define la altura y anchura del iframe, indica la URL que se debe mostrar y mediante el atributo scrolling se indica que el iframe no debe mostrar barras de *scroll* ni siquiera en el caso de que el contenido mostrado no quepa en el iframe definido:

```
<iframe src="/ruta/documento.html" width="250" height="250" scrolling="no" />
```

<iframe>	Marco (frame) en línea
Atributos comunes	básicos
Atributos específicos	▪ `src = "url"` - URL del documento HTML que se visualiza en el iframe ▪ `height = "longitud"` - Altura que ocupará el iframe en el documento ▪ `width = "longitud"` - Anchura que ocupará el iframe en el documento ▪ `name = "texto"` - Nombre que identifica al iframe ▪ `longdesc = "url"` - Dirección en la que puede encontrarse una descripción larga del contenido del iframe ▪ `scrolling = "yes | no | auto"` - Indica si el iframe debe mostrar barras de scroll (horizontal y vertical) cuando el contenido incluido no cabe en el iframe
Tipo de elemento	Bloque y en línea
Descripción	Se emplea para incluir en la página un marco que muestra otro documento HTML

3.4. Otras etiquetas

La etiqueta **<address>** es una de las etiquetas más desconocidas de HTML, por lo que uso no está muy extendido. La etiqueta <address> se utiliza para proporcionar información de contacto.

El siguiente ejemplo sencillo muestra directamente el nombre, dirección y teléfono de contacto de una empresa:

```
<address> Nombre de la empresa Dirección completa Teléfono y Fax
</address>
```

La especificación oficial de HTML muestra un ejemplo complejo del uso de la etiqueta <address>:

<address>	Direcciones
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Define la información de contacto de un documento
<hr>	Línea horizontal
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Permite incluir una línea horizontal de separación

4. Accesibilidad

El principal objetivo de la accesibilidad web es el de permitir a cualquier usuario, independientemente del tipo de discapacidad que sufra, el acceso a los contenidos del sitio y permitirle la navegación necesaria para realizar las acciones deseadas.

Los sitios web accesibles no solamente facilitan el acceso de sus contenidos a los usuarios discapacitados, sino que también permiten ofrecer la misma funcionalidad con dispositivos muy limitados (dispositivos sin pantalla o con pantallas minúsculas, dispositivos sin teclado ni ratón, etc.).

Las cuatro principales ventajas de diseñar un sitio web completamente accesible son las siguientes:

- Los sitios accesibles separan completamente diseño y contenido.
- Un sitio accesible puede ser accedido por multitud de dispositivos diferentes sin necesidad de reescribir el código HTML.
- Los sitios accesibles son los únicos con una audiencia potencial global, ya que permiten el acceso a todos los usuarios y a todos los dispositivos.
- Generalmente, los sitios accesibles son más fáciles de utilizar también para los usuarios sin discapacidades.

La creación de sitios accesibles puede realizarse siguiendo las recomendaciones establecidas por el W3C y que se recogen en el documento *Web Content Accessibility Guidelines* (http://www.w3.org/WAI/intro/wcag.php) (WCAG).

La versión WCAG 1.0 (http://www.w3.org/TR/WCAG10/) que se utiliza en la actualidad se publicó en 1999, mientras que la versión WCAG 2.0 (http://www.w3.org/TR/WCAG20/) se encuentra todavía en borrador y se actualizó por última vez el día 30 de abril de 2008.

Las recomendaciones del WCAG 1.0 están formadas por 65 requisitos que un sitio web debe cumplir para considerarse accesible. Los requerimientos se agrupan en prioridades.

Los requisitos de prioridad 1 son de obligado cumplimiento, los de prioridad 2 son recomendables y los de prioridad 3 son deseables. Si un sitio web cumple con todos los requisitos de prioridad 1, se considera que el sitio es conforme al nivel A de accesibilidad.

El nivel AA de accesibilidad está reservado para los sitios que cumplan todos los requisitos de prioridad 1 y prioridad 2. Finalmente, los sitios que cumplen los 65 requisitos, son conformes al nivel AAA de accesibilidad.

4.1. Requisitos del nivel A de accesibilidad

Los requisitos de accesibilidad que exige el nivel A son los siguientes:

4.1.1. Generales

- Proporcionar un texto alternativo para todas las imágenes, objetos y otros elementos no textuales (mediante los atributos alty longdesc).
- Asegurar que toda la información que utilice el color como elemento informativo pueda ser entendida por las personas o dispositivos que no pueden distinguir los colores.
- Marcar claramente (mediante los atributos lang y xml:lang) las variaciones del idioma del texto o de los elementos textuales (<caption>) respecto del idioma principal de la página.
- El documento debe poder leerse completamente cuando no se utilicen hojas de estilos.
- La información equivalente para los contenidos dinámicos debe adaptarse a los cambios de los contenidos dinámicos.
- Ningún elemento debe parpadear en la pantalla.
- El contenido del sitio se debe escribir con un lenguaje sencillo y limpio.

4.1.2. Si se utilizan mapas de imagen

- Proporcionar un enlace textual por cada una de las regiones del mapa de imagen.
- Utilizar mapas de imagen en el cliente, en vez de mapas de imagen de servidor.

4.1.3. Si se utilizan tablas

- Utilizar cabeceras de fila y de columna.
- Si la tabla tiene varios niveles de cabeceras, utilizar las agrupaciones disponibles (<thead>, <tfoot>).

4.1.4. Si se utilizan frames

- Indicar un título a cada *frame* para su identificación y facilitar la navegación.

4.1.5. Si se utilizan applets y scripts

- Asegurar que la página también se pueda utilizar cuando no se ejecutan los applets y los scripts. Si no es posible, proporcionar informaciones equivalente o páginas alternativas que sean accesibles.

4.1.6. Si se utilizan contenidos multimedia (audio y vídeo)

- Incluir una descripción textual del contenido multimedia.
- Para los contenidos basados en vídeo o animaciones, sincronizar las alternativas textuales con la presentación.