

TEMA 2.

HTML. PÁGINAS WEB ESTÁTICAS. HTML (III)

Lenguajes de Marcas
CFGS DAW 1

Autor: Pascual Ligeró

Revisado por:

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

2019/2020

Versión:191004.2217

Licencia



CC BY-NC-SA 3.0 ES Reconocimiento - NoComercial - Compartirlgual (by-nc-sa)

No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

NOTA: Esta es una obra derivada de la original realizada por Pascual Ligeró.

Nomenclatura

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



Importante



Atención



Interesante

ÍNDICE DE CONTENIDO

1. Metainformación.....	4
1.1 Estructura de la cabecera.....	4
1.2 Metadatos.....	5
1.3 DOCTYPE.....	7
2. Formularios.....	8
2.1 Formularios básicos.....	8
2.2 Elementos de formulario.....	10
2.2.1 Cuadro de texto.....	11
2.2.2 Cuadro de contraseña.....	12
2.2.3 Checkbox.....	12
2.2.4 Radiobutton.....	13
2.2.5 Botón de envío de formulario.....	13
2.2.6 Botón de reseteo del formulario.....	14
2.2.7 Ficheros adjuntos.....	14
2.2.8 Campos ocultos.....	15
2.2.9 Botón de imagen.....	15
2.2.10 Botón.....	15
2.3 Otros elementos de formulario.....	17
3. Bibliografía.....	27
http://librosweb.es/libro/xhtml/	27

UD02. HTML (III)

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

1. META INFORMACIÓN

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**.

1.1 Estructura de la cabecera

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>:

Etiqueta <head>

Atributos comunes [internacionalización](#)

- **profile** = "url" - Especifica la URL del perfil o perfiles que utilizan los metadatos

Atributos propios • **lang** = "codigo_de_idioma" - Especifica el idioma principal de los contenidos de la página

Tipo de elemento -

Descripción 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" />
</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:

Etiqueta	<title>
Atributos comunes	internacionalización
Atributos propios	<ul style="list-style-type: none"> • lang = "codigo_de_idioma" - Especifica el idioma principal del título de la página

Tipo de elemento -

Descripción Define el título del documento HTML

1.2 Metadatos

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>**.

Etiqueta	<meta>
Atributos comunes	internacionalización
Atributos propios	<ul style="list-style-type: none"> • name = "texto" - El nombre de la propiedad que se define (no existe una lista oficial de propiedades) • content = "texto" - El valor de la propiedad definida (no existe una lista de valores permitidos) • http-equiv = "texto" - En ocasiones, reemplaza al atributo "name" y lo emplean los servidores para adaptar sus respuestas al documento • scheme = "texto" - Indica el esquema que se debe emplear para interpretar el valor de la propiedad
Tipo de elemento	-

Etiqueta	<meta>
Descripción	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" />
```

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="librosweb.es" />
```

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.

1.3 DOCTYPE

El estándar XHTML deriva de XML, por lo que comparte con él 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.

Como se verá 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.

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.

Si creas las páginas a mano, sólo tienes que copiar y pegar ese código en cada nueva página.

2. FORMULARIOS

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.

2.1 Formularios básicos

Los formularios más sencillos se pueden crear utilizando solamente dos etiquetas: `<form>` y `<input>`. Si se considera el formulario que muestra la siguiente imagen:



Figura 2.1 Formulario sencillo definido con las etiquetas form e input

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).

Etiqueta `<form>`

Atributos comunes [básicos](#), [internacionalización](#) y [eventos](#)

Atributos propios

- `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).

OJO: Algo que no comenta el texto original es que usando GET cualquiera puede ver la información que se envía al servidor ya que aparece en la URL de petición, por lo que si incluyes campos “secretos” que no quieres que vea el usuario dejarán de ser “secretos”.

2.2 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	<code><input></code>
Atributos comunes	básicos , internacionalización , eventos y foco
Atributos propios	<ul style="list-style-type: none"> • <code>type = "text password checkbox radio submit reset file hidden image button"</code> - Indica el tipo de control que se incluye en el formulario • <code>name = "texto"</code> - Asigna un nombre al control (es imprescindible para que el servidor pueda procesar el formulario) • <code>value = "texto"</code> - Valor inicial del control • <code>size = "unidad_de_medida"</code> - 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) • <code>maxlength = "numero"</code> - Máximo número de caracteres para los controles de texto y de password • <code>checked = "checked"</code> - Para los controles checkbox y radiobutton

Etiqueta `<input>`

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>`.

2.2.1 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:

Nombre

Figura 2.2.1 Ejemplo de etiqueta input (type=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.

2.2.2 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.

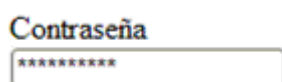


Figura 2.2.2 Ejemplo de etiqueta `input (type=password)`

Contraseña `
`

`<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.

2.2.3 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.

Puestos de trabajo buscados

- ☐ Dirección
- ☐ Técnico
- ☐ Empleado

Figura 2.2.3 Ejemplo de etiqueta input (type=checkbox)

```
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.

2.2.4 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.

Sexo

- ☒ Hombre
- ☐ Mujer

Figura 2.2.4 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.

2.2.5 Botón de envío de formulario

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

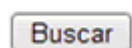


Figura 2.2.4 Ejemplo de etiqueta input (type=submit)

```
<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`.

2.2.6 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 2.2.6 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 se utiliza este atributo, el navegador muestra el texto predefinido del botón, que en este caso es `Restablecer`.

2.2.7 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.

Fichero adjunto

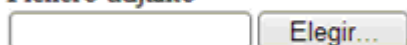


Figura 2.2.6 Ejemplo de etiqueta input (type=file)

Fichero adjunto

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

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>
```

2.2.8 Campos ocultos

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

**Los campos ocultos
no se ven en pantalla*

Figura 2.2.7 Ejemplo de etiqueta `input (type=hidden)`

```
<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.

2.2.9 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 2.2.9 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.

2.2.10 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:

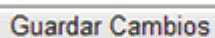


Figura 2.2.10 Ejemplo de etiqueta input (type=button)

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

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.

Ejercicio 1

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

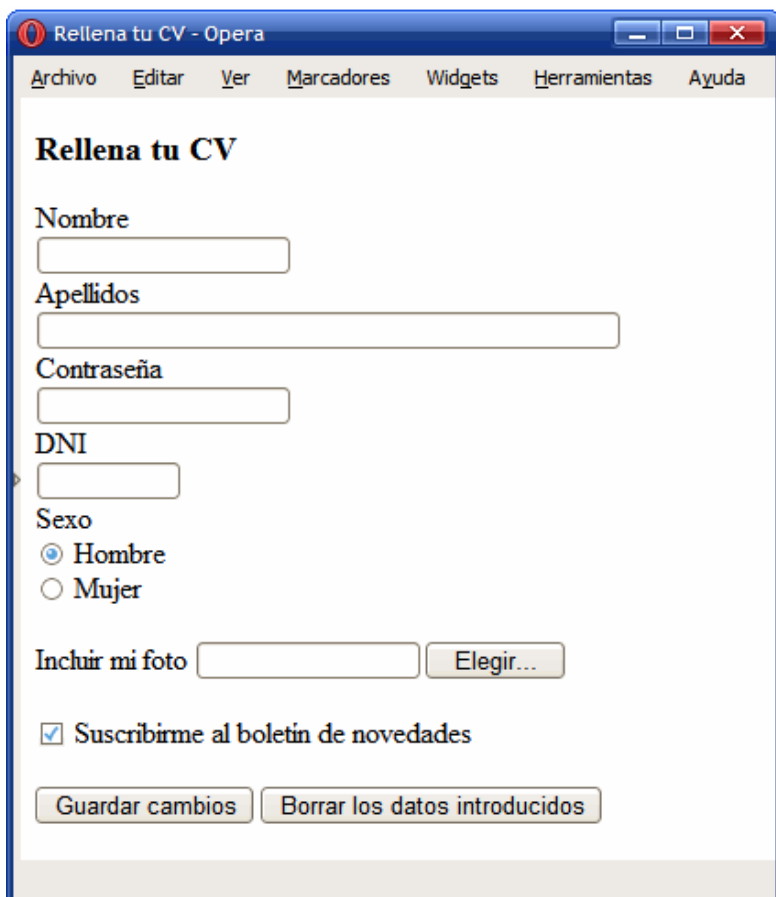


Figura 2.2.11 Formulario con controles de varios tipos

1. El nombre puede tener 30 caracteres como máximo, los apellidos 80 caracteres y la contraseña 10 caracteres como máximo.
2. Asignar los atributos adecuados al campo del DNI.

3. Por defecto, debe estar marcada la casilla de suscripción al boletín de novedades.

Ver solución

Solución:

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

<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Rellena tu CV</title>
</head>

<body>
  <h3>Rellena tu CV</h3>
  <form action="/php/insertar_cv.php" method="post" enctype="multipart/form-
data">
    Nombre <br/>
    <input type="text" name="nombre" value="" size="20" maxlength="30" />
    <br/>
    Apellidos <br/>
    <input type="text" name="apellidos" value="" size="50" maxlength="80" />
    <br/>
    Contraseña <br/>
    <input type="password" name="contrasena" value="" maxlength="10" />
    <br/>
    DNI <br/>
    <input type="text" name="dni" value="" size="10" maxlength="9" />
    <br/>
    Sexo <br/>
    <input type="radio" name="sexo" value="hombre" checked="checked" /> Hombre
    <br/>
    <input type="radio" name="sexo" value="mujer" /> Mujer
    <br/><br/>
    Incluir mi foto <input type="file" name="foto" />
    <br/><br/>
    <input name="suscribir" type="checkbox" value="suscribir"
checked="checked"/> Suscribirme al boletín de novedades
    <br/><br/>
    <input type="submit" name="enviar" value="Guardar cambios" />
    <input type="reset" name="limpiar" value="Borrar los datos introducidos" />
    </form>
  </body>
</html>
```

2.3 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 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:

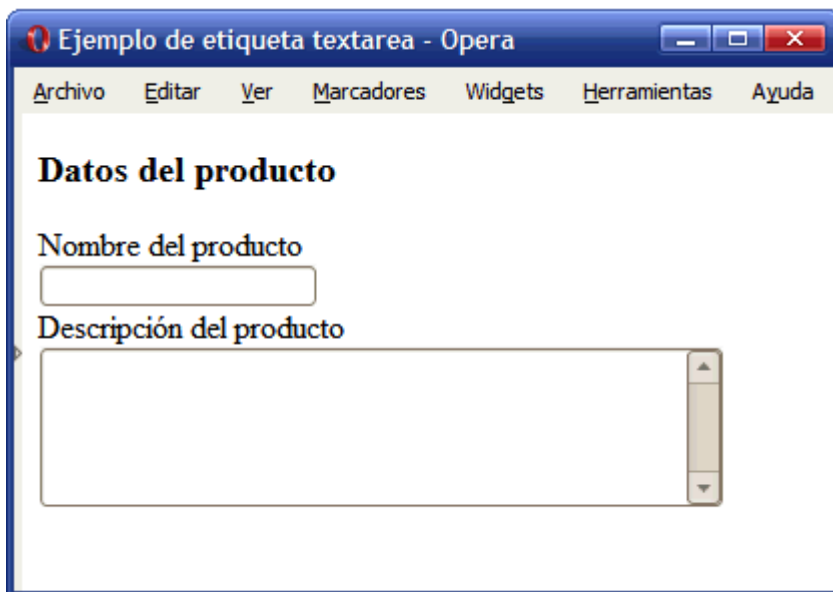


Figura 2.4 Ejemplo de uso de la etiqueta textarea

El código HTML del ejemplo anterior se muestra a continuación:

```
<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"></textarea>

</form>
```

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

Etiqueta	<code><textarea></code>
Atributos comunes	básicos , internacionalización , eventos y foco
Atributos propios	<ul style="list-style-type: none"> <code>rows</code> = "numero" - Número de filas de texto que mostrará el textarea <code>cols</code> = "numero" - Número de caracteres que se muestran en cada fila del textarea Otros: <code>name</code>, <code>disabled</code>, <code>readonly</code>, <code>onselect</code>, <code>onchange</code>, <code>onfocus</code>, <code>onblur</code>

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 2.5 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 se muestra a continuación:

```
<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>`:

Etiqueta	<code><select></code>
Atributos comunes	básicos , internacionalización y eventos
	<ul style="list-style-type: none"> <code>size = "numero"</code> - Número de filas que se muestran de la lista (por defecto sólo se muestra una)
Atributos propios	<ul style="list-style-type: none"> <code>multiple = "multiple"</code> - Si se incluye, se permite seleccionar más de un elemento Otros: <code>name</code>, <code>disabled</code>, <code>onchange</code>, <code>onfocus</code>, <code>onblur</code>

Tipo de elemento En línea

Descripción Se emplea para incluir una lista desplegable en un formulario

Etiqueta `<option>`

Atributos comunes [básicos](#), [internacionalización](#) y [eventos](#)

- `selected = "selected"` - Indica si el elemento aparece seleccionado por defecto al cargarse la página

Atributos propios

- `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 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:

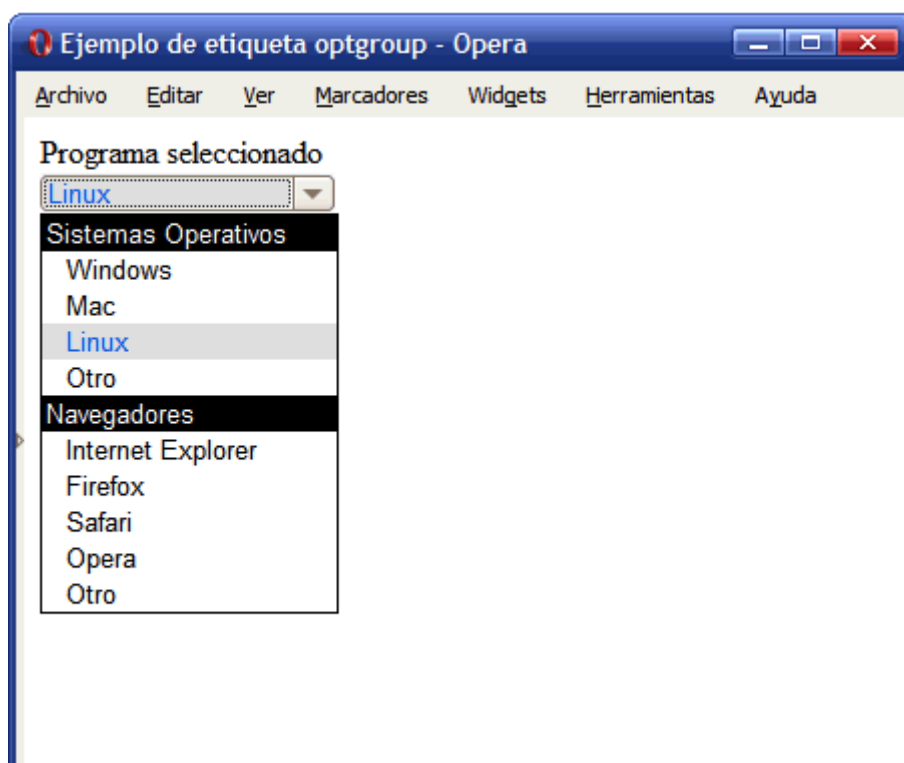


Figura 8.16 Ejemplo de uso de la etiqueta `optgroup`

El código HTML correspondiente a la imagen anterior se muestra a continuación:

```
<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:

Etiqueta `<optgroup>`

Atributos comunes [básicos](#), [internacionalización](#) y [eventos](#)

- `label = "texto"` - Texto que se muestra como título de la agrupación de opciones

Atributos propios

- 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.

Ejercicio 1

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

Figura 8.18 Formulario complejo

[Ver solución](#)

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

<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Información sobre el producto</title>
</head>

<body>

<h3>Información sobre el producto</h3>

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

[illegible]

3. BIBLIOGRAFÍA

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