

UD1_ Aplicacións web e linguaxes de marcas: HTML e CSS

ACTIVIDADE 6: Formularios

- 1. Formularios.....3
 - 1.1 Elementos de los formularios.....3
 - 1.1.1 input4
 - 1.1.2 Etiqueta label.....9
 - 1.1.3 Agrupación de controles.....9
 - 1.1.4 Expresiones regulares.....10
 - 1.1.5 Errores de validación11

1. Formularios

Los formularios son un elemento presente en muchas páginas web cuya labor es la de recabar información del usuario.

Con Html5 se han incluido nuevos elementos y se han mejorado mucho las posibilidades de los formularios. Pero debemos tener en cuenta que la nueva especificación de HTML es soportada de diferente manera por los distintos navegadores y no todas sus características se comportan igual. La página <http://www.wufoo.com.mx/html5/> contiene una tabla muy interesante para saber, sobre cada nueva característica de los formularios HTML 5 qué navegadores la soportan (e incluso desde qué versión).

A pesar de todo debemos comenzar a diseñar formularios con HTML5 ya que hasta ahora, la forma que se utilizaba para validar los formularios “del lado del cliente” era usando javascript. Las nuevas características para manejo de formularios html5 permiten prescindir de javascript para realizar validaciones “del lado del cliente”, esto me permite como diseñador crear un formulario completo sin recurrir a un desarrollador, aumentando la productividad, sobre todo en equipos pequeños de trabajo.

1.1 Elementos de los formularios

form

Es la etiqueta que delimita un formulario, dentro de ella se colocan todos los controles del formulario.

Todos sus atributos son opcionales. Los más importantes son:

- `action`

Es el atributo que contiene la URL de la página web o servicios que procesará los datos del formulario. Si este atributo está vacío, es la página que contiene el formulario la que se recargará con los datos como parámetros.

- `method`

Cuando un usuario rellena un formulario en una página web los datos hay que enviarlos de alguna manera. Las dos formas posibles de envío son `get` y `post`.

- `get`: El concepto de `get` es obtener información del servidor. Los valores enviados se añaden a la dirección indicada en el atributo `action`.
- `post`: El concepto de `post` es enviar información desde el cliente para que sea procesada y actualice la información disponible en el servidor.

Las llamadas por método `get` pueden ser cacheadas (historial navegador), indexadas por buscadores, agregar los enlaces a nuestros favoritos o hasta pasar una URL completa a otra persona para que ingresa a dicha página. Con el método `post` sin embargo no se puede hacer esto.

Elegir entre un método y otro depende de la aplicación concreta que se esté desarrollando y es algo que dentro de las empresas de desarrollos web suelen decidir los encargados del diseño de las aplicaciones. A nosotros en este curso simplemente nos interesa conocer la existencia de ambos métodos.

1.1.1 input

La mayoría de controles de un formulario se inserta mediante la etiqueta `input`. Esta tiene un atributo `type` que indica cual es el tipo de control.

Independientemente del tipo de control que se indique, `input` tiene los siguientes atributos:

Atributo	Descripción
<code>type</code>	Indica el tipo de control
<code>name</code>	Indica el nombre del control. Este nombre es el que se pasa, junto con el valor que el usuario haya introducido, al servicio receptor del formulario
<code>value</code>	Permite dar valor al elemento.
<code>id</code>	Identifica un elemento en HTML5. Como veremos a continuación se usa con la etiqueta <code>label</code> para mejorar el funcionamiento y accesibilidad del formulario. Además es usada para acceder desde JavaScript a un control del formulario.
<code>autofocus</code>	Hace que el foco esté en el <code>input</code> que tenga asociado este valor al cargar la página.
<code>placeholder</code>	Para ofrecer una pista de lo que el usuario debe introducir.
<code>required</code>	Atributo booleano que determina si el elemento es obligatorio o no.
<code>autocomplete</code>	Permite que el navegador rellene automáticamente el contenido del control en base a la información que posee del usuario. Los valores posibles son <code>on</code> u <code>off</code> .
<code>maxlength</code>	Número máximo de caracteres que se admitirá al rellenar el control.
<code>size</code>	Tamaño, en caracteres, que tendrá el cuadro (especialmente útil en los cuadros de texto y numéricos) Anchura, en caracteres, del cuadro de texto. No tiene sentido que sea mayor que <code>maxlength</code> (sí que sea menor).
<code>disabled</code>	El control aparecerá deshabilitado para su edición
<code>readonly</code>	El control es de solo lectura; es decir, no se podrá escribir en él.
<code>form</code>	Recibe el identificador del formulario al que pertenece el control. Se usa cuando el elemento está fuera de la etiqueta <code>form</code> . Internet Explorer no es compatible con este atributo.
<code>formnovalidated</code>	Hace que el control no sea validado cuando se envíen los datos
<code>pattern</code>	Permite colocar una expresión regular en un cuadro de texto. De esa forma no se admitirán los valores que no cumplan esa expresión (salvo que indiquemos <code>novalidate</code>)

Cuadro de texto

Los cuadros de texto permiten recoger texto que escriba el usuario. Permite usar los atributos de la tabla anterior. Su sintaxis es:

```
<input type="text" name="nombre" id="nombre" />
```

Cuadro de Contraseña

Funcionan como los cuadros de texto, solo que el texto que se introduce se oculta, mostrando solo puntos o asteriscos. Usa los mismos atributos que los cuadros de texto. La sintaxis es:

```
<input type="password" name="pass" id="pass" />
```

Si usamos el método GET, la contraseña es visible en la parte superior del navegador. Con POST esto no ocurre, pero aun así podríamos averiguarla. Por ello lo ideal es pasar cifrada la contraseña a través, por ejemplo, del protocolo https.

Botones

Los botones son controles del formulario en los que no se puede escribir, sino que, simplemente al hacer clic sobre ellos provocan diversos efectos. El más común de ellos es enviar los datos del formulario.

En HTML5 se ha sustituido el `input` con `type="submit"` para crear botones por el

elemento `button`. Muchos diseñadores siguen usando el primero porque es compatible con los navegadores antiguos.

Hay varios tipos de botones:

- `submit` → Botón de envío

Sirve para llevar a cabo la comunicación entre el formulario y la página que recoge sus datos. En cuanto se pulsa este botón, los datos del resto de controles se envían a la página receptora del formulario. La sintaxis es:

```
<input type="submit" value="Texto botón" id="identificador"/>
```

- `reset` → Botón restablecer

Al pulsar este botón todos los componentes del formulario vuelven a tener sus valores predeterminados. La sintaxis básica es:

```
<input type="reset" value="Texto botón" id="identificador"/>
```

- `button` → Botón genérico

Un botón genérico se marca indicando `type="button"` en la etiqueta `type`. En los formularios no se usa para enviar o configurar la información, sino que se utiliza normalmente para capturar su pulsación (mediante JavaScript es lo más habitual) y responder en consecuencia.

Casillas de verificación

Una casilla de verificación está definido mediante la configuración de un elemento `input` con el atributo `type` tiene valor `checkbox`. Por defecto está sin seleccionar. El atributo `checked` permite que aparezca seleccionado inicialmente poniendo su valor igual a `checked`.

```
<input type="checkbox" name="acepto" id="acepto">
<label for="acepto">Acepto las condiciones</label>
```

que se vería:

☐ Acepto las condiciones

Botones de radio

Los botones de radio son parecidos a las casillas de verificación en el sentido de que pueden estar seleccionados o no, pero normalmente están agrupados juntos para que únicamente uno de ellos pueda estar seleccionado.

En el siguiente ejemplo usaremos los botones de radio para preguntar al usuario si quiere entrar en nuestro sitio Web como invitado, si quiere configurar una nueva cuenta de usuario, o si quiere iniciar la sesión utilizando una cuenta existente. El nombre que le damos a los tres botones es el mismo: `acceso`; de este modo lo trataremos como un grupo de modo que solo uno de ellos pueda estar seleccionado cada vez. El atributo `value` define el valor que se enviará al servidor cuando esté seleccionado un botón de radio determinado una vez que se envíe el formulario. Por defecto marcamos el botón de login usando el atributo `checked`.

```
<input type="radio" name="acceso" id="invitado" value="invitado">
<label for="invitado">Acceder al sitio como invitado</label>
<input type="radio" name="acceso" id="nuevo" value="nuevo">
<label for="nuevo">Configurar unha nova conta de usuario</label>
<input type="radio" name="acceso" id="login" value="login"
checked="checked">
<label for="login">Iniciar sesión empregando unha conta
existente</label>
```

Cuadros numéricos

Con `type="num"` en el elemento `input`, podemos introducir números (decimales o no). Los navegadores que reconocen este tipo de control presentan un cuadro de texto con botones para subir y bajar el valor del número. Además no permiten enviar los datos (salvo que se use el atributo `novalidate`) si se intentan introducir datos no numéricos.

Con `max`, `min` y `step` podemos delimitar el rango de valores numéricos máximos, mínimos y múltiplos permitidos dentro de un rango.

```
<input id="num" name="num" type="number" min="0" max="100" step="5">
```

Cuadros de fecha y hora

Tenemos varios modos de seleccionar una fecha en función de la parte de fecha que queramos seleccionar. Al indicar los distintos valores al atributo `type` nos abrirá un calendario que permitirá seleccionar cómodamente la fecha. Los posibles valores son:

- `date`: Calendario para seleccionar un día.

```
<input type="date" max="2032-04-20">
```

- `month`: Calendario para seleccionar los meses.

```
<input type="month">
```

- `week`: Fecha para ingresar la semana del año

```
<input type="week">
```

- `time`: Campo para las horas y minutos

```
<input type="time" >
```

- `datetime`: Para indicar una fecha exacta

```
<input type="datetime">
```

Cuadros deslizantes

`type="range"` presenta un control para seleccionar un valor entre dos valores predeterminados. Los atributos `max` y `min` establecen el rango máximo y mínimo del control. El atributo `step` indica cuánto se mueve el control (si de uno en uno, de dos en dos,...)

```
<input type="range" min="0" max="10" step="2" value="6">
```

Cuadro de selección de archivo

Cuando el elemento `input` tiene el valor `"file"` en su atributo `type`, representa un control para seleccionar una lista de uno o más archivos para ser subidos al servidor. Cuando el formulario es enviado, los archivos seleccionados son subidos al servidor, junto con su nombre y tipo.

En primer lugar, debemos saber que si queremos adjuntar archivos a nuestro formulario, es necesario indicar en la etiqueta `form` el atributo `enctype="multipart/form-data"`. Con esto nos aseguramos que las cabeceras del formulario indican que estamos enviando archivos adjuntos.

El atributo `accept` permite indicar, a modo de sugerencia, los formatos de archivos permitidos al usuario en el campo de selección de archivos.

Con el atributo `multiple` se le da la opción al usuario de pulsar la tecla `CTRL` para adjuntar varios archivos a la vez, y no uno solo.

```
<form name="formulario" method="post"
      action="http://mipagina.com/envio.php"
      enctype="multipart/form-data">
```

```
<!-- ;Importante no olvidar el enctype! -->
<input type="file" name="adjunto" accept=".jpg,.png" multiple>
</form>
```

Datalist

Se trata de un elemento HTML 5 muy potente. Permite añadir entradas a un control de cuadro de texto (y también a cuadros especializados como los de email, url,...). La forma de utilizarlo consiste en usar el atributo HTML5 `list` existente en la etiqueta `input`. Ese atributo asociará el cuadro de texto a la lista de valores a través del `id` del `datalist`. Luego dentro de `datalist` se colocan etiquetas `option` para cada opción en la lista (al estilo de los cuadros de tipo `select`). Al hacer foco en ese `input` aparece un dropdown mostrando el contenido del elemento `datalist`, pero podremos escribir lo que queramos (sin elegir ninguna opción de la lista).


```
<label for="como">Cómo nos conociste?</label>
<input type="text" id="como" name="como" list="origen">
<datalist id="origen">
  <select name="select">
    <option value="television">Televisión</option>
    <option value="radio">Radio</option>
    <option value="periodico">Periódico</option>
    <option>Otros</option>
  </select>
</datalist>
```

El elemento `select` no es necesario en los navegadores modernos (que lo ignorarán). Se mete por compatibilidad con los antiguos.

1.1.2 Otros controles

Tenemos controles para:

- **color**: Para seleccionar un color de una paleta


- **tel**: Para números de teléfono. (En realidad, no prueba que sea un número, para validar un formato en particular habría que complementarlo con `pattern`).
- **search**: Para utilizarlo en las cajas de búsqueda. En algunos navegadores no se notará la diferencia con el `input` de texto mientras que en otros se mostrará con las esquinas redondeadas.
- **url**: Para escribir direcciones web
- **email**: Para valores únicos o múltiples de direcciones de correo

Cuadro de texto multilínea

La etiqueta `textarea` permite colocar un cuadro de texto de varias líneas para que el usuario puede introducir un texto largo. El atributo `rows` permite indicar la altura en líneas de texto del cuadro y el atributo `cols`, la anchura en caracteres (los demás atributos son como los de la etiqueta `input type="text"`). Entre la etiqueta `textarea` se puede colocar texto que aparecerá inicialmente dentro del cuadro.

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

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



Cuadros combinados (listas de selección)

Las listas de selección se pueden usar para seleccionar una o varias opciones de una lista. La lista puede ser desplegable, con barra de desplazamiento o únicamente un conjunto de elementos que pueden ser seleccionados.

Una lista de selección comprende un elemento `select` y uno o varios elementos `option`. Cada uno de ellos tiene un atributo `value` que especifica el valor que se enviará al servidor si seleccionamos esa opción. El cuerpo del elemento `option` contiene el texto que se mostrará para esa opción de la lista. Si queremos que una opción aparezca marcada por defecto debemos añadir `selected`.

El siguiente ejemplo, contiene una lista de selección para seleccionar entre varios idiomas. El formato predeterminado es la lista desplegable:

```
<select name="idioma">
  <option value="gl">Gallego</option>
  <option value="es">Castellano</option>
  <option value="eu">Euskera</option>
  <option value="ct">Catalán</option>
  <option SELECTED value="po">Portugués</option>
  <option value="ru">Ruso</option>
</select>
```

Para crear una lista de desplazamiento en lugar de una lista desplegable usaremos el atributo `size` del elemento `select`. Si lo configuramos con valor 1, creará la lista desplegable. Cualquier valor mayor que 1 creará:


- una lista con barra de desplazamiento, si el tamaño es menor que el total de opciones
- una lista sencilla, si el tamaño es mayor o igual al número de opciones.

Para el ejemplo anterior, dando un valor 3 al atributo `size` tendríamos una lista con barra de desplazamiento:

```
<select name="idioma" size="3">
```

Y dando el valor 6 al atributo `size`, tendríamos una lista sin barra de desplazamiento ya que hay seis opciones:

```
<select name="idioma" size="6">
```

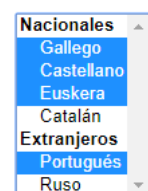


En la imagen vemos como quedarían las tres opciones.

Con el atributo `multiple` podemos seleccionar varios valores, y si queremos que seleccionen al menos una opción del cuadro pondremos `required`.

Se pueden agrupar opciones dentro del cuadro usando el elemento `optgroup`.

```
<label for="idioma">Idiomas que conoces</label>
<select name="idioma" size="8" multiple>
  <optgroup label="Nacionales">
    <option value="gl">Gallego</option>
    <option value="es">Castellano</option>
    <option value="eu">Euskera</option>
    <option value="ct">Catalán</option>
  </optgroup>
  <optgroup label="Extranjeros">
    <option SELECTED value="po">Portugués</option>
    <option value="ru">Ruso</option>
  </optgroup>
</select>
```




```
</optgroup>
</select>
```

Botones HTML5

La recomendación de HTML 5 para los botones es utilizar el elemento `button`. La diferencia práctica está en el funcionamiento, esta etiqueta tiene apertura y cierre y el texto que aparece en el botón está contenido por elemento `button`; de esta forma se permite colocar código HTML en el texto del botón como texto formateado, imágenes, ... La pega es que algunos navegadores antiguos no la soportan y por ello se sigue utilizando `input`.

Cuando el elemento `button` es declarado con el valor "button" en su atributo `type`, la acción que lleva a cabo cuando es presionado, es usualmente provista por un programa. Si el botón no es asociado a un programa, no llevará a cabo ninguna acción al ser presionado. Los atributos de `button` son `id`, `name`, `value`, `type` (con los mismos valores que para `input`), `disabled`, `form` y `formnovalidate`. Además tiene los siguientes:

- `formaction`: Destino de los datos del formulario cuando se pulsa este botón (suponiendo que sea de tipo `submit`).
- `formmethod`: Método de paso (GET o POST) de los datos (si el botón es tipo `submit`).
- `formenctype`: Tipo de codificación (si el botón es de tipo `submit`).
- `formtarget`: El destino de los datos se mostrarán en una ventana aparte si se le da el valor `_blank` a este atributo.

1.1.3 Etiqueta label

Cada control tiene una etiqueta de texto asociada a este mediante el uso del elemento `label`. Este elemento es el más importante en cuanto a la accesibilidad cuando estás construyendo un formulario.

Para asociar la etiqueta a un control se usa el atributo `for`, cuyo valor debe coincidir con el del atributo `id` del control del formulario al que la queremos asociar.

Asociar una etiqueta con un control implica que si hacemos clic en la etiqueta, el control asociado recibe el foco. Esta característica se soporta en todos los navegadores, y es especialmente útil para checkboxes y radio buttons.

El elemento `label` debe usarse con los elementos `textarea`, `select` y los `inputs` de tipo `text`, `radio`, `checkbox`, `file` y `password`.

A continuación vemos una etiqueta para un cuadro de texto de tipo `password` que tiene como identificador `pass`:

```
<label for="pass">Password:</label>
```

Para ello, el valor del atributo `for` debe coincidir con el atributo `id` del elemento `password`:

```
<input type="password" name="pass" id="pass">
```

1.1.4 Agrupación de controles

El elemento `fieldset` permite agrupar controles que comparten un mismo propósito. Visualmente los controles aparecerán encuadrados y debe ir acompañada de la etiqueta `legend` que contiene el texto que encabezará al grupo de controles.

Muchos asistentes usarán el elemento `legend` como si fuera parte de la etiqueta del control. Por ejemplo, algunos lectores de pantalla como Jaws, pronunciarán el contenido

del elemento `legend` antes de leer la etiqueta de cada control.

Vamos a ver un ejemplo:

```
<form>
  <fieldset>
    <legend>Estudios previos</legend>
    <input type="radio" name="estudios" id="estudios_1" value="eso" />
    <label for="estudios_1">ESO</label>

    <input type="radio" name="estudios" id="estudios_2" value="bac" />
    <label for="estudios_2">Bachillerato</label>

    <input type="radio" name="estudios" id="estudios_3" value="cm" />
    <label for="estudios_3">Ciclo medio</label>

    <input type="radio" name="estudios" id="estudios_4" value="cs" />
    <label for="estudios_4">Ciclo superior</label>

    <input type="radio" name="estudios" id="estudios_5" value="grado" />
    <label for="estudios_5">Grado</label>
  </fieldset>
</form>
```

Con este ejemplo, un lector de pantalla leería “Estudios previos ESO” para el primer control, “Estudios previos Bachillerato para el segundo, y así sucesivamente.

El elemento `fieldset` es clave para construir formularios accesibles, y es recomendable incluir dentro de un `fieldset` todos los botones de `radio` que sean excluyentes entre sí. También se recomienda su uso con los controles de tipo `checkbox`, y aunque a veces se use para delimitar secciones, debemos tener cuidado para no abusar de ellos, ya que entonces la lectura del formulario sería tediosa.

1.1.5 Expresiones regulares

Las expresiones regulares son patrones de búsqueda que se pueden utilizar para encontrar el texto que coincide con un patrón determinado.

Patrón	Descripción	Ejemplo		No validaría
^	Busca la coincidencia al inicio de la cadena	<code>^Esto</code>	<i>Esto es una cadena</i>	<i>Es esto de aquí</i>
\$	Coincidencia con el final del input o del texto	<code>final\$</code>	<i>Llegamos al final</i>	
*	el carácter que le precede puede aparecer cero, una, o más veces	<code>mi*</code> coincidiría con: “miiii” y también con “mi”	<i>miiii ó mi ó m</i>	
[...]	Cualquier caracter dentro de los parentesis. Pueden indicarse rangos.	<code>/a[px]e</code> <code>[a-z]5</code>	<i>ape ó axe</i> <i>g5</i>	<i>ale</i> <i>g4</i>
[^...]	Cualquier caracter menos los que están entre paréntesis	<code>/a[^px]</code>	<i>al</i>	<i>ax</i>
?	el carácter que le precede puede aparecer como mucho una vez (0 Ó 1 VEZ)	<code>A[0-9]?</code>	<i>A</i> <i>A9</i>	<i>A99</i>
+	el carácter que le precede debe aparecer al menos una vez (UNA O MÁS VECES)	<code>A[0-9]+</code>	<i>A9</i> <i>A99</i>	<i>A</i>
{n}	Coincidiría exactamente n veces	<code>bua{2}</code>	<i>buaa</i>	<i>bua</i>
{n,}	Coincidiría n o mas veces	<code>bu{2,}/</code>	<i>buuuuu</i>	<i>bu</i>
{n,m}	Coincidiría al menos n y como máximo m veces	<code>[0-9]{2,4}</code>	<i>324</i>	<i>1 ó 989896</i>
.	Para cualquier caracter salvo salto de linea			
\d	Dígito	<code>\d{2}</code>	<i>23</i>	<i>Sd</i>

Patrón	Descripción	Ejemplo		No validaría
	Una barra vertical separa las alternativas.	M[0-9]MP	MP M9	MA 9M

A continuación se muestran algunos ejemplos:

- Letras → [a-zA-Z]+
- Caracteres alfanuméricos → [a-zA-Z0-9]+
- Código postal de 5 caracteres numéricos → [0-9]{5} ó [0-9][0-9][0-9][0-9][0-9] ó \d{5}
- Dirección válida de IPV4 → \d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}
- Fecha (dd/mm/YYYY) → (0[1-9]|[12][0-9]|3[01])[- /.](0[1-9]|1[012])[- /.](19|20)\d\d

1.1.6 Errores de validación

En el momento en que se encuentra un error de validación se ejecuta la acción por defecto del navegador, normalmente aparece un pequeño globo emergente (tooltip) conteniendo un texto de advertencia.

Podemos crear mensajes personalizados para que se muestren cuando el elemento no cumple los criterios de validación.

Para ellos debemos usar:

- El atributo `invalid`, que es un atributo de evento que ejecuta un script de javascript cuando el usuario escribe algo que no cumple los requisitos de validación en un campo `input`.
- El método `setCustomValidity()`, que es un método que establece un mensaje de error personalizado que es mostrado antes de hacer el submit de un formulario, si el campo es inválido.

Por ejemplo:

```
<input type="text" id="alias" name="alias" required pattern="[a-z]+"
oninvalid="setCustomValidity('El alias es obligatorio y debe tener sólo letras
minúsculas')">
```



Realiza las tareas 3 y 4 teniendo en cuenta que debes introducir un mensaje propio cuando se produce un error de validación.

