UD 05. FORMULARIOS.

Índice de contenido

UD 5. FORMULARIOS

1. INTRODUCCIÓN

En esta unidad trataremos algunas de las operaciones más habituales en los formularios. En primer lugar, veremos cómo acceder a elementos de formularios de distinto tipo. También veremos cómo validar envíos de formularios a mano, mediante el evento “onsubmit” y el uso de expresiones regulares.

El elemento HTML form (<form>) representa una sección de un documento que contiene controles interactivos que permiten a un usuario enviar información a un servidor web.

https://developer.mozilla.org/es/docs/Web/HTML/Elemento/form

Al hacer una aplicación Web, el cliente debe validar la información introducida, pero ello no quita el servidor también deba validarla.

2. ACCEDER A ELEMENTOS DE FORMULARIOS.

Independientemente del método utilizado para obtener la referencia a un elemento de formulario, cada elemento dispone de las siguientes propiedades útiles para el desarrollo de las aplicaciones:

- type: indica el tipo de elemento que se trata. Para los elementos de tipo <input> (text, button, checkbox, etc.) coincide con el valor de su atributo type. Para las listas desplegables normales (elemento <select>) su valor es select-one, lo que permite diferenciarlas de las listas que permiten seleccionar varios elementos a la vez y cuyo tipo es select-multiple. Por último, en los elementos de tipo <textarea>, el valor de type es textarea.
- form: es una referencia directa al formulario al que pertenece el elemento. Así, para acceder al formulario de un elemento, se puede utilizar document.getElementById("id_del_elemento").form
- name: obtiene el valor del atributo name de XHTML. Solamente se puede leer su valor, por lo que no se puede modificar.
- value: permite leer y modificar el valor del atributo value de XHTML. Para los campos de texto (<input type="text"> y <textarea>) obtiene el texto que ha escrito el usuario. Para los botones obtiene el texto que se muestra en el botón. Para los elementos checkbox y radiobutton no es muy útil, como se verá más adelante.

Ejemplo:

```
<input type="date" name="fecha" id="fecha">
```

Por último, los eventos más utilizados en el manejo de los formularios son los siguientes:

- onclick: evento que se produce cuando se pincha con el ratón sobre un elemento. Normalmente se utiliza con cualquiera de los tipos de botones que permite definir XHTML (<input type="button">, <input type="submit">, <input type="image">).

- onchange: evento que se produce cuando el usuario cambia el valor de un elemento de texto (<input type="text"> o <textarea>). También se produce cuando el usuario selecciona una opción en una lista desplegable (<select>). Sin embargo, el evento sólo se produce si después de realizar el cambio, el usuario pasa al siguiente campo del formulario, lo que técnicamente se conoce como que "el otro campo de formulario ha perdido el foco".
- onfocus: evento que se produce cuando el usuario selecciona un elemento del formulario.
- onblur: evento complementario de onfocus, ya que se produce cuando el usuario ha deseleccionado un elemento por haber seleccionado otro elemento del formulario. Técnicamente, se dice que el elemento anterior "ha perdido el foco".

2.1 text

Para acceder al valor de un input de tipo texto, simplemente debemos referenciar el atributo “value”.

Ejemplo:

```
<input type=”text” id=”miTexto”>
```

Javascript asociado

```
var elemento=document.getElementById(“miTexto”);
alert(elemento.value);
```

2.2 Radio button

Radio button son elementos del formulario, que, ante varias entradas, te dejan seleccionar una de ellas. Se agrupan teniendo un “name” común.

Para acceder a ellos, se accede como un array, donde se tiene el atributo “value” y el atributo “checked” que es true si está seleccionado, false en caso contrario.

Ejemplo:

```
<input type=”radio” id=”preguntaSI” name=”pregunta” value=“si” />SI
<input type=”radio” id=”preguntaNO” name=”pregunta” value=”no” />NO
```

Javascript asociado

```
var elementos=document.getElementsByName(“pregunta”);
for(i=0;i<elementos.length;i++){
      if(elementos[i].checked==true)
            alert(“Valor del elemento marcado “+elementos[i].value);
}
```

2.3 checkbox

Similar al radio button, salvo que puede haber más de uno marcado.

Ejemplo:

```
<input type=”checkbox” id=”preguntaAS” name=”pregunta” value=“asc” />Piso con ascensor
<input type=”radio” id=”preguntaAM” name=”pregunta” value=”amb” />Piso amueblado
```

Javascript asociado

```
var elementos=document.getElementsByName(“pregunta”);
```

```
for(i=0;i<elementos.length;i++){
      if(elementos[i].checked==true){
            alert("Valor del elemento marcado "+elementos[i].value);
      }
}
```

2.4 select

Elemento que muestra un desplegable y nos permite elegir una opción del mismo. Aquí destaca el atributo "options", que es un atributo que contiene un array con las opciones disponibles y el atributo "selectedIndex" que contiene … (y se puede modificar) la posición del array "options" seleccionada actualmente (o la primera si se permite multiselección) o -1 si no está seleccionada ninguna opción (o queremos des-seleccionarlas).

Dentro de cada "options", "value" almacena el valor y "text" el texto mostrado.

Ejemplo:

```
<select id="aprobar" >
      <option value="10">Saco 10 en DWEC</option>
      <option value="9">Saco 9 en DWEC</option>
      <option value="8">Saco 8 en DWEC</option>
</select>
```

Javascript asociado

```
var elemento=document.getElementById("aprobar");
for(i=0;i<elemento.options.length;i++){
      alert("Valor de la opcion "+elemento.options[i].value);
}
var sel=elemento.selectedIndex;
alert("El valor de la opcion seleccionada es "+elemento.options[sel].value+" y el
texto asociado es "+elemento.options[sel].text);
// Cambiamos el indice seleccionado
elemento.selectedIndex=0;
```

3. VALIDACIÓN DE FORMULARIOS: EVENTO ONSUBMIT

3.1 Validar un formulario

Si un manejador de un evento devuelve true (o no devuelve nada), se realiza el evento asociado. Si el manejador devuelve false, se cancela el evento.

Existe un evento asociado a un formulario completo llamado “onsubmit”.

Aprovechando esto, podemos a nuestro antojo permitir él envió de información al servidor mediante el formulario devolviendo true o cancelarlo devolviendo false.

La estructura típica es la siguiente:

```
<form onsubmit=”return validar();”>
```

Si la función validar devuelve true, se realiza el envío. Si devuelve false, se cancela.

En la función validar podemos hacer las validaciones que estimemos convenientes.

3.2 Deshabilitar enviar un formulario dos veces

A veces un usuario pulsa enviar un formulario más de una vez por error. Si queremos evitar esto, podemos usar “this.disabled=true”;

Ejemplo:

```
<form onsubmit=”this.disabled=true;”>
```

Otra forma:

```
<form id="formulario"action="#">
      ...
      <input type="button" value="Enviar" onclick="this.disabled=true";

      this.value='Enviando...';
      this.form.submit()"/>
</form>
```

Cuando se pulsa sobre el botón de envío del formulario, se produce el evento onclick sobre el botón y por tanto, se ejecutan las instrucciones JavaScript contenidas en el atributo onclick:

1. En primer lugar, se deshabilita el botón mediante la instrucción this.disabled = true;. Esta es la única instrucción necesaria si sólo se quiere deshabilitar un botón.
2. A continuación, se cambia el mensaje que muestra el botón. Del original "Enviar" se pasa al más adecuado "Enviando..."
3. Por último, se envía el formulario mediante la función submit() en la siguiente instrucción: this.form.submit()

El botón del ejemplo anterior está definido mediante un botón de tipo <input type="button" />, ya que el código JavaScript mostrado no funciona correctamente con un botón de tipo <input type="submit" />. Si se utiliza un botón de tipo submit, el botón se deshabilita antes de enviar el formulario y por tanto el formulario acaba sin enviarse.

3.3 Enviar un formulario desde código

En algunas aplicaciones por motivos estéticos o de funcionalidad es deseable que el "enviar un formulario" no se haga desde un botón "submit", sino desde cualquier otro evento que permita la ejecución de código. Esto se puede hacer recogiendo el elemento del formulario y aplicándole el método submit();

Un ejemplo de esto en Javascript.

Ejemplo:

```
<form id="formulario">
```

Javascript asociado

```
var elemento=document.getElementById("formulario");
elemento.submit();
```

4. VALIDACIÓN DE FORMULARIOS: EXPRESIONES REGULARES

4.1 Funciones Javascript para el uso de expresiones regulares

Javascript posee dos formar de crear expresiones regulares:

- Literal con expresión regular. Ejemplo `var re = /ab+c/;`
- El objeto RegExp. Ejemplo `var re = new RegExp("ab+c");`

Entre distintos métodos, uno de los más usado es test. Test recibe una cadena y devuelve true si la cadena cumple esa expresión regular, false en caso contrario.

Ejemplo:

```
var re = new RegExp("ab+c");
var cadena=prompt("Dime una cadena");
if(re.test(cadena)){
      alert("La cadena cumple el patron de una a, entre 1 e infinitas b y al final una c");
}
```

Sobre el uso de expresiones regulares tenéis mas información en
https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Regular_Expressions

Podéis utilizarlas siempre que queráis para cualquier ejercicio.

4.2 Validar un correo.

Se trata de obligar al usuario a introducir una dirección de email con un formato válido. Por tanto, lo que se comprueba es que la dirección parezca válida, ya que no se comprueba si se trata de una cuenta de correo electrónico real y operativa. La condición JavaScript consiste en:

```
valor =document.getElementById("campo").value;
if(!(/\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)/.test(valor))){
      return false;
}
```

La comprobación se realiza nuevamente mediante las expresiones regulares, ya que las direcciones de correo electrónico válidas pueden ser muy diferentes. Por otra parte, como el estándar que define el formato de las direcciones de correo electrónico es muy complejo, la expresión regular anterior es una simplificación. Aunque esta regla valida la mayoría de direcciones de correo electrónico utilizadas por los usuarios, no soporta todos los diferentes formatos válidos de email.

4.3 Validar un número de DNI

Se trata de comprobar que el número proporcionado por el usuario se corresponde con un número válido de Documento Nacional de Identidad o DNI. Aunque para cada país o región los requisitos del documento de identidad de las personas pueden variar, a continuación, se muestra un ejemplo genérico fácilmente adaptable. La validación no sólo debe comprobar que el número esté formado por ocho cifras y una letra, sino que también es necesario comprobar que la letra indicada es correcta para el número introducido:

```
if(!(/^\d{8}[A-Z]$/.test(valor))){
      return false;
}
```

La primera comprobación asegura que el formato del número introducido es el correcto, es decir, que está formado por 8 números seguidos y una letra. Si la letra está al principio de los números, la comprobación sería /^[A-Z]\d{8}$/. Si en vez de ocho números y una letra, se requieren diez números y dos letras, la comprobación sería /^\d{10}[A-Z]{2}$/ y así sucesivamente.

```
var letras =
['T','R','W','A','G','M','Y','F','P','D','X','B','N','J','Z','S','Q','V','H','L',
'C','K','E','T'];
if(valor.charAt(8)!= letras[(valor.substring(0,8))%23]){
      return false;
}
```

La segunda comprobación aplica el algoritmo de cálculo de la letra del DNI y la compara con la letra proporcionada por el usuario. El algoritmo de cada documento de identificación es diferente, por lo que esta parte de la validación se debe adaptar convenientemente.

4.4 Validar un número de teléfono

Los números de teléfono pueden ser indicados de formas muy diferentes: con prefijo nacional, con prefijo internacional, agrupado por pares, separando los números con guiones, etc. El siguiente script considera que un número de teléfono está formado por nueve dígitos consecutivos y sin espacios ni guiones entre las cifras:

```
valor =document.getElementById("campo").value;
      if(!(/^\d{9}$/.test(valor))){
            return false;
}
```

5. BIBLIOTECAS PARA VALIDACIÓN FORMULARIOS JAVASCRIPT

Algunas de las bibliotecas más populares para la validación de formularios Javascript son:

- Validate.js http://rickharrison.github.io/validate.js/
- jQuery Validation Plugin https://jqueryvalidation.org/documentation/

6. MATERIAL ADICIONAL

[1] Curso de Javascript en Udacity validate form jquery
https://www.udacity.com/course/javascript-basics--ud804
[2] Enlaces Libros Web validación

http://librosweb.es/libro/javascript/capitulo_7/validacion.html

7. BIBLIOGRAFÍA

[1] Referencia Javascript
http://www.w3schools.com/jsref/
[2] Expresiones regulares
https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Regular_Expressions