Control de Formularios 14

En este capítulo finalizamos el estudio de los objetos del navegador viendo cómo manipular formularios. Este punto es especialmente importante: si aprendemos correctamente a manipular todos los objetos de un formulario, podremos hacer funciones que nos permitan validarlo antes de enviar estos datos a un servidor, ahorrándole el trabajo de tener que verificar la corrección de los datos enviados.

Antes de pasar a ver los elementos de un formulario, es importante indicar que tiene dos funciones que podemos activar en cualquier momento. Tales funciones son `submit` y `reset`. Por tanto, podríamos tomar los datos de un formulario y poner un botón ACEPTAR que en realidad no fuera un `submit`. Al pulsar el botón de ACEPTAR el programa verificaría los datos introducidos y, si son correctos, llamaría a la función `submit` que enviaría los datos.

También hemos de resaltar que todos los elementos del formulario disponen de una función que se denomina `focus`. Cuando esta función se llama, el cursor pasa al elemento que la ha llamado. Asimismo, todos los elementos de un formulario responden a los eventos `onFocus` y `onBlur`. El primero de ellos se activa cuando el elemento es el seleccionado para escribir o hacer algo sobre él. El segundo se activa cuando deja de ser seleccionado.

14.1 El Objeto FORM

Este objeto es el contenedor de todos los elementos del formulario. Un formulario no es, ni más ni menos, que un elemento de mi página que tendrá sus atributos y su correspondientes propiedades CSS para su maquetación. Lo importante de este elemento es que, a su vez, engloba a otros elementos necesarios para la finalidad de esta etiqueta (solicitar o mostrar información al usuario).

Como con cualquier otro elemento de la página, para poder visualizar o modificar sus atributos (propiedades) o hacer uso de sus métodos, lo primero es acceder al elemento. En capítulos anteriores ya se han mostrado varias formas de acceder a los distintos elementos HTML (usando DOM o el array correspondiente del objeto Document).

14.1.1 Propiedades

Aparte de `id` y/o `class`, las propiedades más utilizadas a las que se pueden acceder cuando se usa una etiqueta `<form>` son:

- `action`: cadena modificable que indica la URL a la que el formulario debe enviar los datos para su procesamiento.
- `method`: cadena modificable que indica el método de envío de los datos del formulario (GET o POST)
- `encoding`: cadena modificable que indica la codificación mime para el atributo `enctype` del formulario.
- `elements`: Es un **array** que contiene todos los elementos del formulario, en el mismo orden en el que se definen en el documento HTML.

Ejemplo: para el formulario que se muestra a continuación:

```
<form method="POST" action="...">
    <input type="text" name="cajaTexto">
    <input type="checkbox" name= "cuadradito">
    <select name="lista">
        …
    </select>
</form>
```

Tendríamos los siguientes valores en el array `elements`:

```
var formu = document.querySelector("form");
console.log(formu.elements[0]); //la caja de texto
console.log(formu.elements[1]); //el cuadradito (checkbox)
console.log(formu.elements[2]);//la lista (select)
```

14.1.2 Métodos

- `reset()`: Resetea el formulario: tiene el mismo efecto que si pulsáramos un botón de tipo RESET dispuesto en el formulario.
- `Submit()`: Envía el formulario: tiene el mismo efecto que si pulsáramos un botón de tipo SUBMIT dispuesto en el formulario.

14.1.3 Acceso a un formulario

Existen varias formas de acceder a un nodo formulario:

La forma más intuitiva es **usando funciones de acceso a nodos.** Sin embargo, esta forma es la menos común de usar debido a que es la más larga de escribir.

Se suelen usar las funciones `document.getElementsByTagname("form")` o mejor, `document.querySelector("form")`

A través del array[14] `forms` del documento: `document.forms[0]`

Normalmente una página web suele tener 1 o, como mucho dos formularios, por lo que esta forma suele ser la más usada.

Usando el atributo `name` del formulario: si la etiqueta `form` del formulario posee el atributo `name` con su valor correspondiente, podríamos acceder al formulario de la siguiente forma: `document.valor_del_name`

Esta forma se denomina notación punto y es propia del manejo de objetos (tanto el *document* como el formulario son objetos). Sin embargo, debido a que aún no sabemos nada sobre el manejo de objetos en Javascript, es la menos intuitiva de las tres.

Ejemplo:

```
!DOCTYPE html>
<html>
    <head>
        <title>Test de Formularios</title>
        <meta charset="utf-8">
    </head>
```

14 Recuerda que en realidad es un objeto NodeList, aunque nosotros lo tratamos como un array.

```
    <body>
         <form action="" id="formu" name="formulario">
              . . .
         </form>
    </body>
</html>
```

Podríamos acceder al formulario del ejemplo de las siguientes formas:

- `var formu = document.getElementsByTagName("form")[0];`
- `var formu = document.getElementById("formu");`
- `var formu = document.forms[0];`
- `var formu = document.formulario;`

14.1.4 Acceso a elementos del formulario

Antes de ver las propiedades y métodos de cada elemento de formulario por separado, vamos a mostrar como acceder a ellos.

En los elementos de formulario es muy importante el atributo `name` dado que, a través de él, vamos a poder capturar los valores que tengan en el servidor a través de lenguajes como PHP. Así que es muy importante colocar correctamente dicho atributo.

Además de lo indicado, a través de este atributo se nos va a facilitar el acceso a dichos elementos.

Independientemente del elemento de formulario que sea, siempre vamos a poder acceder a él usando una de las siguientes formas:

- **funciones de acceso a nodos:** con diferencia la menos usada en estos casos. Como veremos más adelante en este capítulo, la mayoría de los elementos de formulario comparten la misma etiqueta (`input`) lo cual hace tedioso el acceso a elementos usando esta forma.
- **A través del array `elements` del formulario:** *(necesitamos tener una referencia al formulario como se vio en el apartado anterior)*: `formu.elements[posicion]`

También es una forma un poco tediosa porque necesitamos saber la posición que ocupa el elemento que buscamos dentro del formulario. Es fácil equivocarse si el formulario es grande.

- Usando el valor del atributo `name` del elemento: *(necesitamos tener una referencia al formulario como se vio en el apartado anterior)*: `formu.valor_de_name` o `formu[valor_de_name]`

  Notación punto de los objetos en Javascript. Es la más usada con diferencia por su sencillez.

Ejemplo:

```
<body>
   <form action="" id="formu" name="formulario">
     Nombre: <input type="text" name="nombre"> <br>
     Clave: <input type="password" name="clave"> <br>
     Sexo: <br>
     <input type="radio" name="sexo"> Masculino <br>
     <input type="radio" name="sexo"> Femenino <br>
     Condiciones de uso:
    <input type="checkbox" name="condiciones">
    Marca para aceptar  <br>

     <select name="lista">
         <option>Una</option>
         <option>Dos</option>
         <option>Tres</option>
     </select>
    <br>
   <input type="button" name="benv" value="Enviar"
onclick="accesos()">
    <input type="reset" name="borrar" value="Borrar">
</form>
</body>
```

```
function accesos(){

    //acceso al formulario
     let formu = document.forms[0];

    //Accesos a Nombre
    let nombre = formu.elements[0];
    //nombre = formu.nombre;
    //nombre = formu['nombre'];

    console.log(nombre);

   //Acceso al Sexo
    sexo = formu.sexo;
    console.log(sexo);



    //Lista de opciones
    lista = formu.lista;
    console.log(lista);

}
```

14.2 Los objetos TEXT, TEXTAREA y PASSWORD

Estos objetos representan los campos de texto dentro de un formulario. El elemento `password` es exactamente igual que el elemento `text` salvo oculta los caracteres introducidos por el usuario, poniendo asteriscos (*) en su lugar.

14.2.1 Propiedades

`id`	Equivale al atributo id del elemento.
`defaultValue`	Es una cadena contiene el valor que se le da por defecto al atributo `value`. Aunque el atributo `value` cambie durante la ejecución, `defaultValue` sigue devolviendo su valor inicial.
`type`	Cadena que contiene el valor del atributo `type`.
`name`	Es una cadena que contiene el valor del atributo `name`.
`value`	Es una cadena que contiene el valor del elemento, es decir, su texto. Puede coincidir con el atributo `value` si este está definido por defecto.
`size`	Tamaño del campo de texto.
`disabled`	Devuelve el valor del atributo `disabled`. Mediante `elemento.disabled=true` se desactiva el elemento en cuestión.

14.2.2 Métodos

`blur()`	Pierde el foco del ratón sobre el objeto especificado.
`focus()`	Obtiene el foco del ratón sobre el objeto especificado.
`select()`	Selecciona todo el texto dentro del objeto dado.

Ejemplo:

```
<html>
<head>
  <title>Ejemplo de JavaScript</title>

  <script type="text/javaScript">

    function Mostrar() {
      var nombre = document.getElementById("formulario");
      alert('Su nombre: ' + formulario.nombre.value);
      alert('El password: ' + formulario.pass.value);
    }
  </script>
</head>
<body>
  <form name="formulario" id="formulario">
      Nombre:
      <input type="text" name="nombre" value="Tu nombre"><br />
      Password:
      <input type="password" name="pass" maxlength="10"><br />


      <input type="button" name="b" value="Aceptar"
                onclick="Mostrar()">
  </form>

</body>
</html>
```

12.3 El objeto BUTTON

Este objeto hace referencia a los botones de formulario definidos con la etiqueta `<input type= "...">`. Por tanto, tendremos tres tipos de botones: un botón genérico, `button`, que no tiene acción asignada, y dos botones específicos, `submit` y `reset`. Estos dos últimos sí que tienen una acción asignada al ser pulsados: el primero envía el formulario y el segundo limpia los valores del formulario.

12.3.1 Propiedades

Además de los mencionados anteriormente : `id, disabled, type`.

`name`	Es una cadena que contiene el valor del atributo `name`.
`value`	Es una cadena que contiene el valor del atributo `value`.
`form`	Devuelve una referencia al formulario en el que se encuentra el botón.

12.3.2 Métodos

`click()`	Realiza la acción de pulsado del botón
`blur()`	Pierde el foco del ratón sobre el objeto especificado.
`focus()`	Obtiene el foco del ratón sobre el objeto especificado.

Ejemplo:

```
<html>
<head>
  <title>Ejemplo de JavaScript</title>

  <script type="text/javaScript">

   function Mostrar(boton) {
     alert('Ha hecho click sobre el boton: ' + boton.name+', de
            valor:'+boton.value);
     return true;
   }

  </script>
 </head>
 <body>
   <form  name="formulario" id="formulario" >
```

```
      <input type="button" name="Boton1" value="El boton 1"
          OnClick="Mostrar(this);"><br>
      <input type="button" name="Boton2" value="El boton 2"
          OnClick="Mostrar(this);"><br>
      <input type="button" name="Boton3" value="El boton 3"
          OnClick="Mostrar(this);"><br>
    </form>

  </body>
</html>
```

12.4 El objeto CHECKBOX

Los elementos checkbox nos permiten seleccionar varias opciones marcando el *cuadrito* que aparece a su izquierda. Si se marca el *cuadradito*, equivale a un "sí" (o true) y si no se marca, a un "no" (o false).

12.3.1 Propiedades

Además de los mencionados anteriormente : `id, disabled, type`.

`checked`	Valor booleano que nos dice si el checkbox está pulsado o no. Es modificable.
`defaultChecked`	Valor booleano que nos dice si el checkbox debe estar seleccionado por defecto o no.
`name`	Es una cadena que contiene el valor del atributo `name`.
`value`	Es una cadena que contiene el valor del atributo `value`.

12.4.2 Métodos

`click()`	Realiza la acción de pulsado del botón
`blur()`	Pierde el foco del ratón sobre el objeto especificado.
`focus()`	Obtiene el foco del ratón sobre el objeto especificado.

```
<html>
 <head>
   <title>Ejemplo de JavaScript</title>

   <script type="text/javaScript">

     function Mostrar() {
        msg="Opcion 1:"+formulario.check1.checked+"\n"
        msg+="Opcion 2:"+formulario.check2.checked+"\n"
        msg+="Opcion 3:"+formulario.check3.checked+"\n"
        alert(msg);
     }
   </script>
 </head>
 <body>
    <form name="formulario" id="formulario">

     <input type="checkbox" name="check1" checked='checked'> Opcion 1<br>
     <input type="checkbox" name="check2"> Opcion 2<br>
     <input type="checkbox" name="check3"> Opcion 3<br>
     <input type="button" name="b" value="Ver Valores" onclick="Mostrar()">


    </form>
 </body>
</html>
```

12.5 El objeto RADIO

Al contrario que con los `checkbox`, que nos permiten elegir varias posibilidades entre las mostradas, los objetos `radio` sólo nos permiten elegir una de entre todas las que hay. Están pensados para posibilidades mutuamente excluyentes (Pej: no se puede ser a la vez mayor de 18 años y menor de 18 años o no se puede estar a la vez soltero y casado).

12.5.1 Propiedades

Además de los mencionados anteriormente : `id, disabled, type`.

`checked`	Valor booleano que nos dice si el checkbox está pulsado o no. Es modificable.
`defaultChecked`	Valor booleano que nos dice si el checkbox debe estar seleccionado por defecto o no.
`name`	Es una cadena que contiene el valor del atributo `name`.
`value`	Es una cadena que contiene el valor del atributo `value`.
`length`	Valor numérico que nos dice el número de opciones dentro de un grupo de elementos radio.

RECUERDA: Para agrupar elementos de tipo radio, todos ellos deben tener el mismo valor en el atributo `name`.

12.5.2 Métodos

`click()`	Realiza la acción de pulsado del botón
`blur()`	Pierde el foco del ratón sobre el objeto especificado.
`focus()`	Obtiene el foco del ratón sobre el objeto especificado.

```
<html>
<head>
  <title>Ejemplo de JavaScript</title>
  <script type="text/JavaScript"">

    function Mostrar() {
      msg="Elementos:"+formulario.edad.length+"\n";
     msg+="Menor de 18 años:"+formulario.edad[0].checked+"\n";
     msg+="Entre 18 y 60 años:"+formulario.edad[1].checked+"\n";
     msg+="Mayor de 60 años:"+formulario.edad[2].checked+"\n";
     alert(msg);
    }
  </script>
</head>


<body>
   <form name="formulario" id="formulario">

       Edad:<br>
       <input type="radio" name="edad" value="<18">
        Menor de 18 años.<br>
      <input type="radio" name="edad" value=">18 y <60" checked>
        Entre 18 y 60 años.<br>
      <input type="radio" name="edad" value=">60">
        Mayor de 60 años.<br>

      <input type="button" name="b" value="Ver Valores"
             onclick="Mostrar()">
    </form>

</body>
</html>
```

12.6 El objeto SELECT

Este objeto representa una lista de opciones dentro de un formulario. Normalmente se trata de una lista desplegable de la que podremos escoger alguna (o algunas) de sus opciones.

Lo más importante que se debe saber de un `select` en lo respectivo a Javascript es:

1. Para cada `select` que existe en la página web, se crea en Javascript un array denominada: `select_accedido.options`.
2. Cada `select` tiene una propiedad llamada `selectedIndex` que indica qué opción del `select` está seleccionada. Hay que tener en cuenta que empieza contando en *0*.
3. Un evento muy usado en este tipo de elementos es evento `onChange` que se produce cuando el usuario elige una opción del `select` distinta a la que estaba seleccionada en ese momento.

12.6.1 Propiedades

Al ser un elemento compuesto por dos etiquetas distintas (`select` y `option`), vamos a tener propiedades para cada una de esas etiquetas

Propiedades de `<select>`

Además, de los ya comentados: `id, disabled`**.**

`length`	Valor numérico que nos indica cuántas opciones tiene la lista .
`name`	Es una cadena que contiene el valor del atributo `name`. Es modificable.
`selectedIndex`	Valor numérico que nos dice cuál de todas las opciones disponibles está actualmente seleccionada.
`value`	Mostrará el valor de la opción que esté seleccionada.
`multiple`	Permite conocer y establecer cuando la lista admite selección múltiple.

`size`	Nos permite saber el valor del atributo `size`, que viene a ser el número de opciones visibles al mismo tiempo en la lista desplegable.

Propiedades de `<option>`

Como ya se ha comentado, cada elemento `select` tiene asociada un array que guarda las posibles opciones de del elemento. Para acceder a ese array hay que colocar `select_accedido.options`

Las propiedades **de cada elemento de ese array** son:

`defaultSelected`	Valor booleano que nos indica si la opción está seleccionada por defecto.
`index`	Valor numérico que nos da la posición de la opción dentro de la lista.
`selected`	Valor booleano que nos dice si la opción está actualmente seleccionada o no. Es modificable
`text`	Cadena con el texto mostrado en la lista de una opción concreta. Es modificable.
`value`	Es una cadena que contiene el valor del atributo `value` de la opción concreta de la lista.

12.6.2 Métodos

Los métodos más comunes para `<select>` son los ya comentados:

`click()`	Realiza la acción de pulsado del botón
`blur()`	Pierde el foco del ratón sobre el objeto especificado.
`focus()`	Obtiene el foco del ratón sobre el objeto especificado.

```
<html>
 <head>
   <title>Ejemplo de JavaScript</title>

   <script type="text/javaScript">

     function Mostrar() {
         msg="Elementos:"+formulario.edad.length+"\n";
         msg+="Edad:"+
         formulario.edad.options[formulario.edad.selectedIndex].value+"\n";

        alert(msg);
     }
   </script>


 </head>
 <body>
  <form name="formulario" id="formulario" >

    Edad:<br>
    <select name="edad">
       <option value="<18" SELECTED>Menor de 18 años</option>
       <option value=">18 y <60">Entre 18 y 60 años</option>
       <option value=">60">Mayor de 60 años</option>
    </select>
   <input type="button" name="b" value="Ver Valores" onclick="Mostrar()">
  </form>

 </body>
</html>
```