VALIDACIÓN DE FORMULARIOS

Contenido

1. OBJETO form

Para definir un formulario es necesario introducir una primera instrucción con la siguiente *sintaxis*:

```
<FORM
      NAME="NombreDelFormulario"
      TARGET="NombreDeVentana"
      ACTION="URLDelServidor"
      METHOD=GET|POST
      ENCTYPE="encodingType">
       ..........................
</FORM>
```

Donde:

- *NAME* es el nombre del formulario.
- *TARGET* es el nombre de la ventana que debe activarse después del envío, en caso de trabajar con frames o simplemente con dos ventanas separadas. En caso de que un programa *CGI* devuelva código ***HTML*** al navegador, por ejemplo un mensaje de aceptación del formulario, su emisión se producirá en la ventana indicada en el *TARGET*.
- *ACTION* es la URL del servidor al cual es enviada la información. Este atributo puede especificar una aplicación en el servidor, para que gestione el formulario; también puede especificar una dirección de correo, si el formulario se quiere recibir a través de correo. La forma sería: *ACTION=*[*"mailto:usuario@dominio"*]
- *METHOD* puede contener dos valores: *GET|POST* indica cómo se enviará la información hacia el servidor especificado en *ACTION*. Para el protocolo HTTP, el más común, puede tomar los valores *GET* o *POST*. Para los *mailto:* se debe utilizar el método *POST*, de esta forma, la URL se envía como un attatchment MIME del correo.
- *ENCTYPE* especifica el MIME codificado del bloque enviado.

Para acceder con JavaScript a un formulario determinado, se puede utilizar básicamente uno de los siguientes métodos:

- ***Método1***: A los formularios se puede acceder por su número de formulario o índice. La numeración empieza por 0. Al asignar los índices se sigue el mismo orden en que se han escrito los tags <FORM> en el archivo.

```
document.forms[numform].Propiedad
document.forms[numform].Método()
destino=document.forms[0].action;
document.forms[0].reset()
```

- ***Método2***: También se puede acceder a los formularios utilizando el id definido en el atributo **ID** de su definición

```
document.getElementById("identificador").Propiedad
document.getElementById("identificador").Método()
destino= document.getElementById("identificador").action;
document.getElementById("identificador").reset()
```

- ***Método3***: Se puede escribir directamente el nombre del formulario, definido en el atributo **NAME** de su definición, para hacer referencia a su contenido, propiedades y métodos.

```
document.nombreform.Propiedad
document.nombreform.Método()
destino=document.nombreform.action;
document.nombreform.reset()
```

Propiedades

action	Permite consultar el valor del atributo *ACTION* establecido en la definición.
encoding	Permite leer el valor del atributo *ENCTYPE* establecido en la definición.
length	Permite leer el número de elementos del formulario.
method	Contiene el valor del atributo *METHOD* establecido en la definición.
name	Contiene el nombre del formulario.
target	Contiene el valor de *TARGET* establecido en la definición.

Los objetos del formulario también son considerados propiedades.

Métodos

reset()	Borra todas las entradas de un formulario. Su efecto es el mismo que el botón definido como type=reset.
submit()	Valida el formulario y éste es enviado al receptor. Su efecto es el mismo que el botón definido como type=submit.

Ejemplo 1: En las líneas siguientes se define un formulario y dos enlaces para borrar las entradas y enviar el formulario.

```
<FORM NAME="Profesional"
    ACTION="mailto:mio@servidor.com"
    METHOD=POST >
    <p>ESTUDIOS: <input type="text" size="10" name="dato1"> </p>
    <p>PROFESIÓN: <input type="text" size="15" name="dato2"> </p>
</FORM >
<a href="javascript:document.Profesional.reset()"> Borrar entradas
</a><p>
<a href="javascript:document.Profesional.submit()"> Enviar entradas
</a>
```

Subobjetos

Con el objeto elements, que en la jerarquía JavaScript se encuentra por debajo del objeto forms, se tiene acceso a los elementos del formulario. Para acceder con JavaScript a un elemento determinado de un formulario se pueden seguir varios métodos:

- ***Método1***: Por medio de un índice. La numeración empieza por 0. Al asignar los índices se sigue el mismo orden en que se han escrito los distintos elementos en el archivo.

```
document.forms[numform].elements[numelemento].Propiedad
document.forms[numform].elements[numelemento].Método()
document.forms[0].elements[0].value="dirección";
document.forms[0].elements[0].focus();
```

- ***Método2***: También se puede acceder a los elementos de los formularios utilizando el nombre del formulario y de los elementos que se definió en el atributo ***name***.

```
document.NombreDelFormulario.NombreDelElemento.Propiedad
document.NombreDelFormulario.NombreDelElemento.Método()
document.DatosProfesion.titulo.value;
document.DatosProfesion.titulo.focus();
```

- ***Método3:*** También se puede acceder a los elementos de los formularios utilizando el identificador del formulario y de los elementos que se definió en el atributo **id.**

```
document.getElementById("identificador").value
```

Propiedades

checked	Contiene el valor *true* si el elemento está activo y *false* si no lo está. Se puede aplicar a botones pulsables, cuadros de control y botones de opción
name	Almacena el nombre del elemento de formulario, según se definió en la propiedad *NAME*. Se puede utilizar con todos los elementos de formulario.
form	Aquí se guarda el formulario en el que se encuentra el elemento en cuestión. Se puede utilizar con todos los elementos de formulario.
type	Guarda el tipo al que pertenece un elemento de formulario. Se puede utilizar con todos los elementos de formulario.
value	Guarda el valor introducido en un elemento de formulario o bien el valor asignado a este elemento. Se puede aplicar a todos los elementos de formulario.
defaultValue	Contiene el texto definido en *VALUE*. Esta propiedad es aplicable los cuadros de texto.

Métodos

blur()	Elimina el foco del elemento en el que se active. Se puede aplicar a todos los elementos de formulario.
focus()	Activa el foco del elemento que corresponda. Se puede aplicar a todos los elementos de formulario.
click()	Simula una pulsación en el botón correspondiente. Se puede aplicar a cualquier tipo de botón y cuadro de control.
select()	Selecciona el texto completo del cuadro. Se puede aplicar a cuadros de texto.

2. OBJETO button

Este objeto se muestra en pantalla como un **botón** rectangular con un texto en su interior y es susceptible de ser pulsado, normalmente, para activar la acción que indica el texto. Sintaxis:

```
<INPUT TYPE="button"
    NAME¨="NombreDelBotón"
    VALUE="TextoDelBotón">
```

Donde:

- *NAME*: nombre del objeto
- *VALUE*: texto del botón

Ejemplo 2: El siguiente formulario muestra un botón que al pulsarlo lanza una ventana de información.

```
<FORM>
    <input type="button"
         name="bAutor" value="Autor"
     onClick="alert('Rosa Fernández')">
</FORM>
```

3. OBJETO text

Este objeto muestra una **caja** en la que se pueden introducir todo tipo de caracteres.

Sintaxis:

```
<INPUT TYPE="text"
    NAME¨="NombreObjeto"
    VALUE="TextoProDefecto"
    SIZE=NumEntEro
    MAXLENGTH=NumMaximoCaracteres>
```

Donde:

- *NAME*: nombre del objeto.
- *VALUE*: valor inicial del texto.
- *SIZE*: capacidad máxima de caracteres del objeto.
- *MAXLENGTH*: número máximo de caracteres que acepta el cuadro. En el caso de no introducir este dato, el número máximo es ilimitado.

Ejemplo 3: Se crean tres campos de texto y un botón. Se introducen el nombre y el apellido, cada uno en su campo y al pulsar el botón debe aparecer el nombre completo en el tercer campo. Observar que el nombre completo aparece en mayúsculas debido a la utilización de la función *toUpperCase()*.

```
<script language="JavaScript">
  function nombreCompleto(){
    datos.resultado.value=datos.nombre.value.toUpperCase()+" "+
        datos.apellido.value.toUpperCase();
  }
</script>
<FORM NAME="datos">
    <p>Nombre: <input type="text" size="10" name="nombre">
     Apellidos: <input type="text" size="10" name="apellido"><p>
     <input type="button" name="bresult" value="Nombre completo"
              onclick="nombreCompleto()"> <p>
     <input type="text" size="20" name="resultado"> </p>
</FORM>
```

4. OBJETO textarea

Este objeto muestra una **caja** en la que se pueden introducir todo tipo de caracteres en **varias líneas**.

Sintaxis:

```
<TEXAREA
    NAME="NombreObjeto"
    ROWS= "Entero"
    COLS="Entero"
    WRAP=="off | virtual | physical">
   Texto
</TEXAREA>
```

Donde:

- *NAME*: nombre del objeto.
- *ROWS*: número de filas visibles para el objeto TexArea.
- *COLS*: número de caracteres por fila.
- *WRAP*: Controla como se almacena el texto. Las posibilidades son:
    - *off*: es el modo por defecto, las líneas serán enviadas exactamente como se han tecleado
    - *virtual*: el texto será enviado como una línea continua
    - *physical*: envía el texto con saltos de línea que coinciden con los del objeto.

Ejemplo 4: Hacer una caja de texto que compruebe que se han introducido datos cuando pierde el foco. En caso de que no se introduzcan datos se devuelve el foco al campo. El código que se encuentra a continuación del formulario se ejecuta inmediatamente y presupone la existencia de un formulario. En esta área se inicializa el foco en el campo definido en el formulario.

```
<script language="JavaScript">
  function CompContenido(caja){
    if (caja.value.length==0){
              alert ("tiene que introducir datos");
              caja.focus();
    }
  }
</script>
<FORM NAME="primero">
    <p>Introduce un comentario:<br>
    <textarea name="caja" rows="4" cols="11"
         onblur="CompContenido(caja)">
    </textarea> </p>
</FORM>
<script language="JavaScript">
  primero.caja.focus();
</script>
```

5. OBJETO checkbox

Este objeto muestra una **casilla de verificación**.

Sintaxis:

```
<INPUT TYPE="checkbox"
    NAME¨="NombreObjeto"
    VALUE="ValorDevuelto"
    [CHECKED]>
 "Mensaje"
```

Donde:

- *NAME*: nombre del objeto.
- *VALUE*: valor que se devuelve al servidor si la casilla está activada y el formulario es validado.
- *Mensaje*: es el texto que se muestra al lado de la casilla

Ejemplo 5: Tendremos un botón que activa a la vez tres casillas de verificación. En la función se pasa como parámetro el objeto this.form, el cual hace referencia al formulario actual.

```
<script language="JavaScript">
  function Marcar (formulario){
    formulario.caja1.checked=true;
    formulario.caja2.checked=true;
    formulario.caja3.checked=true;
  }
</script>
<FORM>
    <p><input type="button" name="botón1"
                value="Marcar las 3 opciones" onClick="Marcar
(this.form)"><br>
        <input type="checkbox" name="caja1" value="Opción1">
            Primera Opción<br>
        <input type="checkbox" name="caja2" value="Opción2">
            Segunda Opción<br>
        <input type="checkbox" name="caja3" value="Opción3">
       Tercera Opción<br></p>
</FORM>
```

6. OBJETO radio

Este objeto muestra varios **botones circulares** que muestran distintas opciones, de las cuales se debe escoger una.

Sintaxis:

```
<INPUT TYPE="radio"
    NAME¨="NombreObjeto"
    VALUE="ValorObjeto"
    [CHECKED]>
"Mensaje"
```

Donde:

- *NAME*: nombre del objeto. Es corriente utilizar series de botones con el mismo name, de forma que se traten como grupos. Solo puede haber un elemento seleccionado del mismo grupo. En caso de tener que gestionar un único elemento se utiliza la notación array.
- *VALUE*: valor que es devuelto al servidor cuando se confirma un formulario y la casilla está activada. Contiene el atributo value de la definición del objeto.
- *Mensaje*: es el texto que se muestra al lado de botón.

Ejemplo 6: El siguiente ejemplo consta de un cuadro de texto, tres botones de radio y un botón. Al pulsar el botón se ejecuta una función que comprueba cuál de los tres botones de radio está activado, y actúa convirtiendo el contenido del cuadro de texto en mayúsculas, minúsculas o dejándola tal y como está.

```
<script>
  function Cambiar (formulario){
     if (formulario.converTexto[0].checked==true){
        formulario.cuadroTexto.value=formulario.cuadroTexto.value.toUpp
erCase();
    }else{
        if (formulario.converTexto[1].checked==true){

        formulario.cuadroTexto.value=formulario.cuadroTexto.value.toLow
erCase();
        }
    }
 }
</script>
<FORM NAME="FormTexto">
    <p><input type="text" size="25" name="cuadroTexto"><br>
    <input type="radio" name="converTexto" value="May">Mayúsculas<br>
    <input type="radio" name="converTexto" value="Min">Minúsculas<br>
    <input type="radio" name="converTexto" value="Igual">Sin
Cambios<br>
    <input type="button" name="boton" value="convertir"
           onclick="Cambiar(this.form)"> </p>
</FORM>
```

7. OBJETO select

Defina una **lista de selección**.

Sintaxis:

```
<SELECT
     NAME¨="Nombre"
     [SIZE="Numero"]
     [MULTIPLE]
     <OPTION VALUE="valor" [SELECTED]>TextoElemento1
          ..............
     <OPTION>Textoelemento
</SELECT>
```

Donde:

- *NAME*: contiene el atributo name de la definición select.
- *SIZE*: número que determina la cantidad de elementos visibles de la lista. Opcional.
- *MULTIPLE*: activando esta opción se podrá seleccionar más de un elemento de la lista.
- *OPTION*: especifica un elemento para la lista. El mensaje que muestra el elemento debe ponerse a continuación (TextoElemento). Las option pueden gestionarse mediante notación de array.
- *VALUE*: especifica el valor que se devuelve al servidor (o al programa) cuando se valida el formulario.
- *SELECTED*: especifica que elemento aparecerá seleccionado por defecto. Opcional.
- *TextoElemento*: texto que se visualizará en el formulario acompañado a cada elemento de la lista.

Propiedades

length	Contiene el número de elementos de la lista.
name	Contiene el atributo name de la definición del select.
selectedIndex	Contiene el índice de la opción seleccionada. Si la lista es de selección múltiple contiene el índice del primer elemento seleccionado.
options	Esta propiedad contiene un array al cual podemos acceder con las siguientes propiedades.
defaultSelected	Contiene el índice seleccionado por defecto en la lista.
index	Contiene el índice de una opción.
length	Contiene el número de elementos de la lista.
selected	Contiene true o false, dependiendo de sí el elemento está o no seleccionado.

value	Contiene el valor del elemento seleccionado en la lista.
texto	Contiene el texto que acompaña al elemento de la lista (aquel que sigue a cada <option>).

Ejemplo 7-1: Se muestra una lista de selección en la cual al elegir una montaña se muestra en otra lista el país donde se encuentra

```
<script language="JavaScript">
  function mostrarpais(lista,texto){
    if (lista.selectedIndex==0) texto.value="Rusia";
    else
        if (lista.selectedIndex==1) texto.value="Suiza";
        else
                if (lista.selectedIndex==2) texto.value="Italia";
                else
                    if (lista.selectedIndex==3) texto.value="Chile";
  }
</script>
  <FORM NAME="viajes">
    <p>Selecciona una montaña
    <select name="Montaña" size="1"
         onchange="mostrarpais(this,viajes.pais)">
            <option>URALES </option>
            <option>ALPES </option>
            <option>APENINOS </option>
            <option>ANDES </option>
    </select> El pais de la montaña es:
    <input type="text" size="20" name="pais"> <br>
</FORM>
```

Ejemplo 7-2. Muestra todas las propiedades de la opción seleccionada

```
<html>
<head>
<script>
function dimePropiedades(){
 var texto ;
 texto = "El numero de opciones del select: " +
document.formulario1.miSelect.length;
 var indice = document.formulario1.miSelect.selectedIndex;
 texto += "\nIndice de la opcion escogida: " + indice;
 var valor = document.formulario1.miSelect.value;
 texto += "\nValor de la opcion escogida: " + valor
 var textoEscogido =
document.formulario1.miSelect.options[indice].text;
 texto += "\nTexto de la opcion escogida: " + textoEscogido;
 alert(texto);
}
</script>
</head>
<form name="formulario1">
Valoración sobre este web:
<select name="miSelect">
<option value="10">Muy bien
```

```
<option value="5" selected>Regular
<option value="0">Muy mal
</select>
<br>
<br>
<input type=button value="Dime propiedades"
onclick="dimePropiedades()">
</form>
</html>
```

8. OBJETO password

Este tipo se utiliza generalmente para la introducción de las claves de acceso. Este objeto presenta un cuadro de texto en el cual no se ve el texto que se escribe no es legible.

Ejemplo: El siguiente ejemplo pide la clave de usuario permitiendo ver el contendio con un botón

```
Introduzca la clave de usuario:
<input type="password" name="clave">
```

La información que contiene el campo es accesible desde el código *JavaScript* utilizando la propiedad *VALUE* del campo

9. EVENTOS: onReset, onSubmit, onSelect

Cuando el formulario se rellena, es necesario que la información que contiene sea enviada al servidor. Esta operación se realiza con el método submit. Para ello se debe utilizar la siguiente instrucción dentro del formulario:

```
<input type=submit value="Enviar">
```

Es conveniente comprobar la correcta entrada de los datos del formulario para una mejor comunicación entre el emisor y el receptor. En la comprobación, además de rechazar las entradas incorrectas, se impiden las modificaciones en los cuadros de entrada, se confirma el envío del formulario por correo electrónico, se evalúan los formularios de pedidos, o se limita la longitud de los textos. Para ello podemos utilizar una cabecera de formulario de este tipo:

```
<FORM ACTION="mailto:ppp@aaaa.com"
    METHOD="post"
    ENCTYPE="text/plain"
    onSumit="return test()">
```

Donde ***test()*** es una función que comprueba la correcta introducción de los textos y devuelve *false* en el caso de que las entradas tengan algún error. El formulario es enviado solamente si onSumit devuelve el valor true, sino se emite un mensaje de error.

El evento ***onReset*** se ejecuta cuando se pulsa el botón de formulario Reset; el evento onSubmit cuando se pulsa el botón de enviar formulario. Estos eventos son aplicables al objeto form.

El evento ***onSelect*** se ejecuta cuando el usuario selecciona un texto en un área o un cuadro de texto de un formulario. Es aplicable a los elementos de formulario Text y Textarea.

Ejemplo 9: Muestra un formulario con un campo de texto y dos botones, uno para borrar el contenido (reset) y otro para enviar la información a la dirección de correo indicada en la cabecera del formulario (submit). El formulario detecta la pulsación de los botones correspondientes y activa el evento correspondiente y ejecuta la acción que tiene asignada cada uno de ellos. Si se realiza una selección en el campo de texto definido se activa el evento onSelect y se ejecuta la acción que tiene asignada.

```
<FORM
    ACTION="mailto:ppp@tala.com"
    METHOD="post"
    ENCTYPE="text/plain"
   onReset="return confirm('¿Quiere borrar todas las entradas?')"
   onSubmit="return confirm('¿Desea enviar la información?')">
   Dirección de correo:
   <INPUT TYPE="text" NAME="direccion" onSelect="alert('Si pulsas
Contro+C podrás copiar la información')"<p>
   <INPUT TYPE="reset" VALUE="Borrar el contenido">
   <INPUT TYPE="submit" VALUE="Enviar">
</FORM>
```