# Javascript

5. Funciones y Validaciones

Pablo Garramone R

# Formularios en Javascript

- JavaScript dispone de numerosas propiedades y funciones que facilitan la programación de aplicaciones que manejan formularios.

- En este documento básicamente vamos a ver:

1. Como acceder a los distintos elementos de un Formulario.
2. Como manipular los elementos del formulario
3. Como validar los elementos del formulario

# Acceder a Formularios

- Tenemos 3 formas de acceder a un formulario:

1. A través de los arrays forms, con todos los formularios que haya en la página, y elements, un array con cada uno de los elementos de dichos arrays, que el navegador genera en el momento que se carga la página web.
2. A través de los atributos NAME de las etiquetas del formulario.
3. Usando el DOM.

# Acceder con Arrays forms y elements

- De las 3 formas de acceso **es la menos recomendable**, porque depende totalmente del orden en el que esten dispuestas las etiuqetas en el HTML, y si hay algún cambio haría que todo el código quedara inservible, lo cual es un  muy grave problema.
- Siempre partimos del objeto document.
- **document.forms:** Es un array donde cada posición del mismo es cada una de las etiquetas FORM que haya en el html.
  - ***Ejemplo: document.forms[0] sería el primer formulario.***
- **document.forms[0].elements:** Es un array con cada uno de las etiquetas que haya dentro del FORM.
  - ***Ejemplo: document.forms[0].elements[0] sería el primer elemento del primer formulario que hubiera en el HTML.***

# Ejemplo con Arrays forms y elements

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8">
    <title>Ejemplo Formulacio con Forms y Elements</title>
    <script>
     window.onload = function(){
      let primerElemento=document.forms[0].elements[0];
      console.log(primerElemento.nodeName+" "+primerElemento.name);
      //IMPRIME: input nombre
     }
    </script>
</head>
<body>
    <h2>EJEMPLO FORMULACIO</h2>
    <form name="formulario" method="get" action="pagina2.html">
     <p>
      <label>Nombre: <input type="text" name="nombre" maxlength="50"></label>
     </p>
     <p>
      <label>Apellidos: <input type="text" name="apellidos" maxlength="50"></label>
     </p>
    </form>
</body>
</html>
```

# Acceder con los atributos NAME

- La forma más sencilla e intutiva.
- Se trata de acceder a través de los atributos NAME de cada elemento del formulario.
- Partimos como siempre desde document.
- Accederiamos desde el formulario y luego el elemento o control que queremos acceder:
- ***Ejemplo: document.<valor name formulario>.<valor name del control>***

# Ejemplo con atributos NAME

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8">
    <title>Ejemplo Formulacio con Forms y Elements</title>
    <script>
     window.onload = function(){
      let primerElemento=document.formulario.nombre;
      console.log(primerElemento.nodeName+" "+primerElemento.name);
      //IMPRIME: input nombre
     }
    </script>
</head>
<body>
    <h2>EJEMPLO FORMULACIO</h2>
    <form name="formulario" method="get" action="pagina2.html">
     <p>
      <label>Nombre: <input type="text" name="nombre" maxlength="50"></label>
     </p>
     <p>
      <label>Apellidos: <input type="text" name="apellidos" maxlength="50"></label>
     </p>
    </form>
</body>
</html>
```

# Acceder con el DOM

- Exactamente igual que accedíamos en el tema 3.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8">
    <title>Ejemplo Formulacio con Forms y Elements</title>
    <script>
     window.onload = function(){
      let primerElemento=document.getElementsByTagName("input")[0];
      console.log(primerElemento.nodeName+" "+primerElemento.name);
      //IMPRIME: input nombre
     }
    </script>
</head>
<body>
    <h2>EJEMPLO FORMULACIO</h2>
    <form name="formulario" method="get" action="pagina2.html">
     <p>
      <label>Nombre: <input type="text" name="nombre" maxlength="50"></label>
     </p>
     <p>
      <label>Apellidos: <input type="text" name="apellidos" maxlength="50"></label>
     </p>
    </form>
</body>
</html>
```

# EJERCICIO 1

- Usando el HTML del aula virtual.

1. Imprime por el console.log el TYPE del input del correo electrónico y el ID del desplegable ROL usando las 3 formas de acceso.
2. Imprime por el console.log el NAME de todos los controles del formulario, usando las 3 formas de acceso.

# Validar Formularios

- La forma más habitual para validar formularios es en el evento SUBMIT que es el momento que se envía el formulario entero.
- Se pueden realizar validaciones en el momento de grabar un control (con onfocus, onblur, onchange…) pero eso suele ser más engorroso y siempre es conveniente en el onsubmit para asegurarse.
- El evento SUBMIT cuando salta este evento, tiene 2 opciones que devuelva TRUE entonces enviará el formulario a la página del ACTION del FORM o FALSE y no envía nada. Por esta razón cuando validemos y queramos que no se envíe un formulario deberemos devolver FALSE.

# Validar Formularios

- Si ponemos el evento directamente en el FORM, es importante no olvidar el return:

```
<form action="" method="" id="" name="" onsubmit="return validacion()">
  ...
</form>
```

- Ya que si lo olvidamos aunque hagamos un return false el formulario se enviará siempre.
- Luego en la función validar, iremos elemento a elemento accediendo y comprobando el VALUE de cada control si es correcto o no, poniendo return false en el caso que no.

# Validar Formularios

```
13          function validar(){
14              let f=document.formulario;
15              if(f.nombre.value==""){
16                  alert("Error el nombre no puede estar vacío");
17                  return false;
18              }
19              if(f.apellidos.value==""){
20                  alert("Error los apellidos no pueden estar vacíos");
21                  return false;
22              }
23          }
24      </script>
25  </head>
26  <body>
27      <h2>EJEMPLO FORMULACIO</h2>
28      <form name="formulario" method="get" action="pagina2.html"
29      onsubmit="return validar()">
30          <p>
31              <label>Nombre: <input type="text" name="nombre" maxlength="50"></label>
32          </p>
33          <p>
34              <label>Apellidos: <input type="text" name="apellidos" maxlength="50"></label>
35          </p>
36          <p>
37              <input type="submit" value="GRABAR">
38          </p>
39      </form>
40  </body>
41  </html>
```

# Validar Formularios con el DOM

- Como siempre es más conveniente usar el DOM y no tocar el HTML

```
    <script>
        window.onload = function(){
            let f=document.formulario;
            f.onsubmit=validar;
        }

        function validar(){
            let f=document.formulario;
            if(f.nombre.value==""){
                alert("Error el nombre no puede estar vacío");
                return false;
            }
            if(f.apellidos.value==""){
                alert("Error los apellidos no pueden estar vacíos");
                return false;
            }
        }
    </script>
</head>
<body>
    <h2>EJEMPLO FORMULACIO</h2>
    <form name="formulario" method="get" action="pagina2.html">
        <p>
            <label>Nombre: <input type="text" name="nombre" maxlength=
```

# Elementos Especiales Formulario

- A parte del VALUE para acceder a los input text, password, radio, desplegables… tenemos otros controles que el VALUE no nos valdrá.
- Para saber si se ha seleccionado un CHECKBOX se utiliza **checked**:

```
elemento = document.getElementById("campo");
if( !elemento.checked ) {
  return false;
}
```

- También vale para el RADIOBUTTON, pero como solo se puede seleccionar 1 valor, también podemos ir por el VALUE.
- El desplegable (SELECT) si solo admite 1 resultado podemos ir por el VALUE, pero tiene una propiedad interesante: **selectedIndex**, que nos indicara el elemento/s que se ha seleccionado.

```
indice = document.getElementById("opciones").selectedIndex;
if( indice == null || indice == 0 ) {
  return false;
}
```

# Elementos Especiales Formulario

- Desde javaScript podemos provocar con código un HREF o un SUBMIT.

```
if(<determinada circunstancia>)
    f.submit();

if(<determinada circunstancia>)
    location.href="pagina2.html";
```

- FOCUS: Podemos hacer que el foco se ponga en un control determinado. Muy útil cuando ha habido errores y queremos que el control se quede en el error.

```
if(f.nombre.value==""){
    alert("Error el nombre no puede estar vacío");
    f.nombre.focus();
    return false;
}
if(f.apellidos.value==""){
    alert("Error los apellidos no pueden estar vacíos");
    f.apellidos.focus();
    return false;
}
```

# Validar con API html5

- Podemos crear nuestras propias validaciones con mensajes de validación personalizados usando el método setCustomValidity(mensaje).
- Este método funciona solo funciona con el evento SUBMIT.
- La idea es sencilla, si un control tiene esta «variable» de Validity distinto de "" , el submit fallará automáticamente y no irá a la siguiente pantalla. Por lo que la técnica es sencilla cuando pongamos un mensaje al control indicaremos que ese control fallará con ese mensaje y si queremos que no falle porque ha superado la validación lo dejaremos a vacío.

# Validar con API html5

- Como se puede observar ya no se realiza la valicación al dar a SUBMIT, debemos hacerlo antes, y ya cuando se le de a submit como haya algún validity que no sea vacío fallará solo.

```
6       <script>
7           window.onload = function(){
8               let f=document.formulario;
9               if(f.nombre.value=="")
10                  f.nombre.setCustomValidity("Error el nombre no puede estar vacío");
11              else
12                  f.nombre.setCustomValidity("");
13
14              if(f.apellidos.value=="")
15                  f.apellidos.setCustomValidity("Error los apellidos no pueden estar vacíos"
16              else
17                  f.apellidos.setCustomValidity("");
18          }
19      </script>
20  </head>
21  <body>
22      <h2>EJEMPLO FORMULACIO</h2>
23      <form name="formulario" method="get" action="pagina2.html">
```

# Validar con Expresiones Regulares

- Ver anexo.

# EJERCICIO 2

1. Modifica el ejemplo del aula virtual en otro fichero pero esta vez que el SUBMIT se realice en la etiqueta FORM
2. Modificalo otra vez en otro fichero para que todas las validaciones se realicen con serCustomValidity.

# Ejercicio 3

- Ve al documento EJERCICIO HTML DE TABLAS y FORMULARIOS del tema 2 HTML y vuelve a hacer el ejercicio 3 de formularios, añadiendo 3 controles más de contraseña y repetir contraseña y una fecha de nacimiento.
- Debes hacer todas las validaciones con javaScript nada con HTML. Y a las que ya se solicitan añade:

1. Todos los controles deben ser rellenados usando expresiones regulares.
2. La contraseña debe tener mínimo 8 caracteres, y debe tener al menos 1 minúscula, 1 mayúscula y 1 número. Usa expresiones regulares.
3. La contraseña y repetir contraseña deben coincidir.
4. La fecha debes asegurarte que los días, mes y año son números y son correctos (el año debe ser de este siglo). Debes hacerlo con expresiones regulares