I.E.S. PALOMERAS VALLECAS

FORMULARIOS Y EXPRESIONES REGULARES

UT5.- INTERACCIÓN CON EL USUARIO EN JAVASCRIPT

Susana López Luengo

Utilización de formularios desde código

- Los formularios se definen con las etiquetas **<form> </form>** y los atributos **action** y **method**
- Los formularios tienen entre otros los siguientes atributos
  - **action**, URL donde se redirigen los datos
  - **method** método por el cual lo envía (POST o GET)
  - **enctype**, tipo de codificación que empleamos para los archivos adjuntos
  - **accept** de archivos que acepta el servidor.
  - **accept-charset, onsubmit u onreset**, amplían algunas posibilidades para los formularios aunque son algo menos utilizados.

Utilización de formularios desde código

Etiqueta input

- Esta etiqueta se transforma en distintos elementos que a su vez almacenan un tipo de dato.
- El atributo **type** indica el tipo de elemento
  - **text** (cuadro de texto).
  - **password** (cuadro de texto para contraseñas)
  - **checkbox** (casilla de verificación)
  - **radio** (opción de entre dos o más)
  - **submit** (botón de envío del formulario)
  - **reset** (botón de vaciado de campos).
  - **file** (botón para buscar ficheros).
  - **hidden** (campo oculto)
  - **image** (botón de imagen en el formulario).
  - **button** (botón del formulario).

Utilización de formularios desde código

Etiqueta input

- Con HTML5 se introdujeron nuevos tipos pero no son soportados por todos los navegadores
  - **color** Debería mostrar una rueda de color
  - **date** Debería mostrar un selector de fecha
  - **datetime-local** Muestra el día y hora local
  - **email** Debería validar automáticamente al enviarlo
  - **month** Debería mostrar un selector de fecha
  - **number** Permite restricciones
  - **range** Define un control tipo slider
  - **search** Para campos de búsqueda
  - **tel** Debería validar automáticamente al enviarlo
  - **time** Debería mostrar un selector de hora
  - **url** Debería validar automáticamente al enviarlo
  - **week** Debería mostrar un selector de fecha
  - **datalist:** Muestra una lista desplegable

Utilización de formularios desde código

Propiedades de la etiqueta input

- **Name**: asigna un nombre al elemento.
- **Value**: valor del elemento.
- **Size**: asigna el tamaño inicial del elemento en pixeles o caracteres.
- **Maxlength**: máximo de caracteres que pueden contener los elementos text y password
- **Checked**. opción por defecto de checkbox y radio
- **Disabled**. El elemento aparece deshabilitado
- **Readonly**. Solo lectura
- **Src**. URL de la imagen de un botón del formulario.
- **Alt**. descripción del elemento.
- **required**. El elemento es obligatorio

https://www.w3schools.com/tags/tag_input.asp

Utilización de formularios desde código

Ejemplo de formulario

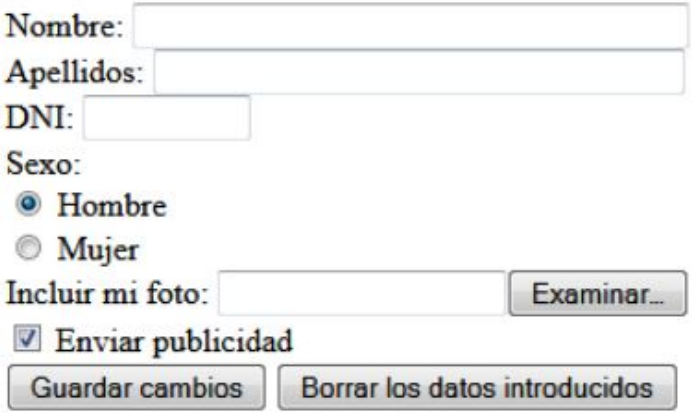
```
<form action="/formulario.php" method="post">
    <div>
        <label for="nombre">Nombre:</label>
        <input type="text" id="nombre" />
    </div>
    <div>
        <label for="correo">E-mail:</label>
        <input type="email" id="correo" />
    </div>
    <div>
        <label for="msg">Mensaje:</label>
        <textarea id="msg"></textarea>
    </div>

    <div class="button">
        <button type="submit">Envia tu mensaje</button>
    </div>
</form>
```

Utilización de formularios desde código

elemento input

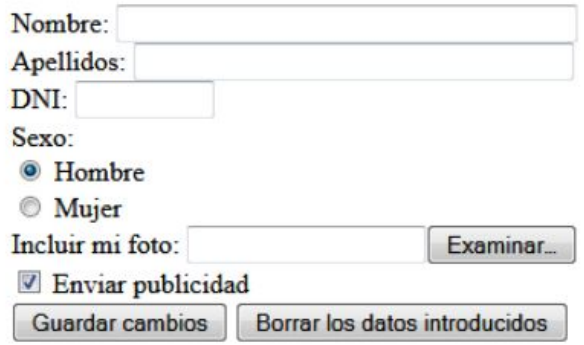
- El elemento **<input>** es el elemento más importante de un formulario
- Puede mostrarse de muchas formas, dependiendo del atributo **type**
  - button
  - checkbox
  - text
  - password
  - ...



- https://developer.mozilla.org/es/docs/Web/HTML/Elemento/input
- https://www.w3schools.com/tags/tag_input.asp

Utilización de formularios desde código

Ejemplo input




```
<form action="pagina.php" method="post"
 enctype="multipart/form-data" /> <br/>
 Nombre:<input type="text" name= "nombre" value="" size="42 maxlength="30”/>
 Apellidos:<input type="text" name="ape" value="” size="40” maxlength="80”/>
 DNI:<input type="text" name="dni" value="" size="10” maxlength="9" />
 Sexo:
 <input type="radio" name="sexo" value="hombre” checked="checked”/>Hombre
 <input type="radio" name="sexo" value="mujer" />Mujer
 Incluir mi foto: <input type="file" name="foto" /> <br/>
 <input name="publ" type="checkbox" value="publicidad” checked="checked"/>
 Enviar publicidad
 <input type="submit" name="enviar" value="Guardar cambios” />
 <input type="reset" name="limpiar" value="Borrar los datos introducidos”/>
</form>
```

Más etiquetas y elementos para formularios

- Etiqueta textarea -- campo de texto - párrafo
  - Ejemplo
  - Documentación
- Etiqueta select -- lista desplegable
  - Ejemplo
  - Documentación
  - Ejemplo con optgroup
- output -- guarda el resultado de una ejecución (HTML5)
  - Ejemplo
  - Documentación

Estilizando formularios

- A algunos elementos de formulario se les puede dar estilo con pocos o ningún problema independientemente de la plataforma
  - <form>
  - <fieldset>
  - <label>
  - <output>
- Esto también incluye todos los campos de texto (tanto los de línea simple como los de línea múltiple) y los botones.

Estilizando formularios

Problemas para utilizar CSS en los formularios

- Hay otros elementos a los que raramente se les puede aplicar estilos y pueden llegar a requerir técnicas complejas y ocasionalmente necesitan conocimientos avanzados de CSS3.
- El elemento **<legend>** no puede posicionarse adecuadamente en todas las plataformas
- Los elementos **checkbox y los botones de radio** no permiten que se le apliquen estilos directamente con CSS2
- Al contenido de **placeholder** no se le puede aplicar estilo de ninguna forma convencional

Estilizando formularios

Problemas para utilizar CSS en los formularios

Elementos en los que no se puede utilizar CSS:

- Elementos avanzados de interfaces de usuario tales como los controles **range, color, o date**
- Widgets desplegables como **select, option, optgroup y datalist**
- Los nuevos elementos **progress, meter y selector de archivos**

Esto es debido a su estructura muy compleja y CSS no es lo suficientemente expresivo para estilizarlas

Para personalizar estos widgets se deberá recurrir a javaScript para construir un árbol DOM que permita acceder a ellos

Estilizando formularios

Cajas de texto

- Se pueden aplicar estilos a las cajas de texto sin problemas a excepción de las cajas de búsqueda
- En los navegadores basados en webkit (Chrome, Safari, etc.) se debe lidiar con la propiedad -webkit

```html
<form>
  <input type="search">
</form>
```

```css
input[type=search] {
  border: 1px dotted #999;
  border-radius: 0;

  -webkit-appearance: none;
}
```

Estilizando formularios

Fuentes y texto

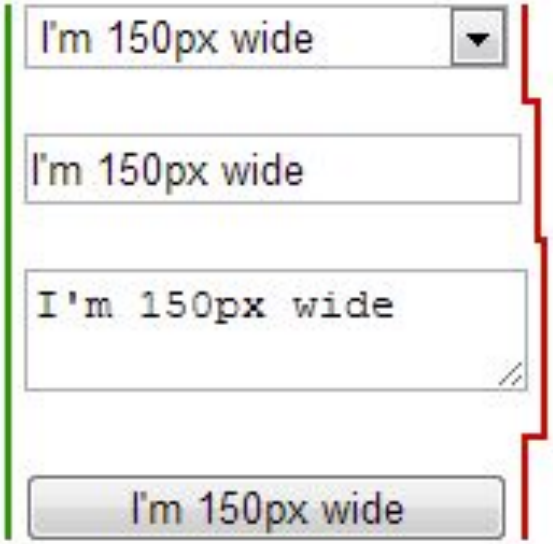
- Las fuentes y capacidades de texto de CSS se pueden utilizar sin problemas en cualquier widget
- Hay que tener cuidado que el comportamiento de los navegadores no es siempre consistente
- Por defecto, algunos widgets no heredan font-family ni font-size de sus antecesores y utilizan la apariencia por defecto.
- Para corregirlo debemos añadir las siguientes reglas de normalización:

```
button, input, select, textarea {
  font-family : inherit;
  font-size   : 100%;
}
```

Estilizando formularios

Modelo de cajas

- Todos los campos de texto tienen soporte completo para las propiedades relacionadas con el modelo de cajas de CSS (width, height, padding, margin y border).
- Los navegadores utilizan los estilos por defecto del sistema cuando muestran estos widgets
- Si se mantiene el aspecto nativo de los widgets, se pueden producir pequeñas inconsistencias de tamaño.
- Esto es porque cada widget tiene sus propias reglas para el orden, margen y padding



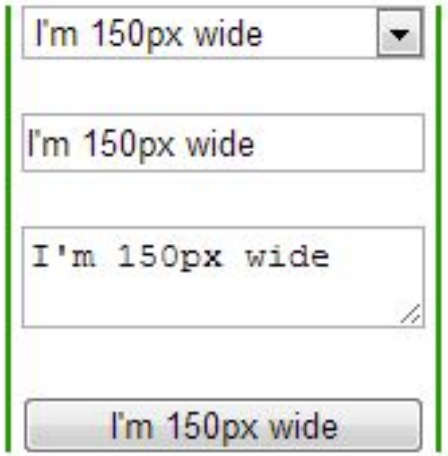
Estilizando formularios

Modelo de cajas

- Para dar el mismo tamaño a varios widgets diferentes se debe usar la propiedad **box-sizing**:

```
input, textarea, select, button {
  width : 150px;
  margin: 0;

  -webkit-box-sizing: border-box; /* Para navegadores
antiguos basados en WebKit */
       -moz-box-sizing: border-box; /* Para navegadores
antiguos basados en Gecko */
            box-sizing: border-box;
}
```



Estilizando formularios

Posicionado legend

- En los navegadores el elemento legend se posiciona encima del borde superior de su antecesor fieldset
- No existe ninguna posibilidad de colocarlo dentro del flujo HTML más allá del borde superior
- Se puede posicionar de forma relativa o absoluta mediante la propiedad position, pero seguirá siendo parte del borde de fieldset.
- Como el elemento legend es muy importante por razones de accesibilidad, se suele emparejar con un título que se oculta pero siendo aún accesible

```
<fieldset>
  <legend>Hi!</legend>
  <h1>Hello</h1>
</fieldset>
```

```
legend {
  width: 1px;
  height: 1px;
  overflow: hidden;
}
```

Validación y envío

- Cuando un formulario es enviado, podemos procesar el contenido con javascript para asegurarnos de que todos los datos está bien:
  - Campos obligatorios
  - Emails válidos
  - Fortaleza de una contraseña
  - Habilitar o deshabilitar campos según lo seleccionado
  - Etc
- En el formulario <form> añadiremos el atributo onsubmit con el resultado de la validación

```
<form action="URL" method="post" name="formValidado"
  onsubmit="return validar()">
  ...
</form>
```

Validación y envío

```
<script type="text/javascript">

    function validar()    {
      if( comprobación--)
      {
         <!-- mensaje de error !>
         return false;
      }
      //…
      return true;
}
</script>
```

evento.preventDefault()

- preventDefault() **previene la acción por defecto de un elemento**.
- Los formularios normalmente después de enviar los datos al servidor por defecto recargan la página
- Para evitar este efecto pasaremos el argumento event y llamaremos al método preventDefault

```
formulario.addEventListener('submit', (e) => {
    e.preventDefault();
    ... instrucciones de validación de formulario
   }
```

Expresiones regulares

- Las expresiones regulares describen un conjunto de elementos que siguen un **patrón**.
- Es una regla que identifica a una serie de elementos que tiene algo en común
- Un ejemplo podría ser todas las palabras que comienzan por la letra ‘a’ minúscula
- JavaScript implementa expresiones regulares y facilita las comprobaciones de ciertos datos que deben seguir una estructura concreta.

Expresiones regulares. Objeto RegExp

- Una expresión regular es un objeto que describe un patrón de caracteres.
- Las expresiones regulares se utilizan para realizar reconocimiento de patrones y funciones "buscar y reemplazar" en el texto.
- Sintaxis: `/patron/modificadores;`
- Modificadores:
  - **i:** no distingue mayúsculas de minúsculas
  - **m:** multilínea
  - **g:** global. Localiza todas las cadenas de caracteres que cumplan el patrón (No se limita al primero)

Expresiones regulares. patrones

caracteres especiales

\b	Fin de palabra. Indica que tiene que haber un fin de palabra o retorno de carro.
\B	No fin de palabra. Indica cualquiera que no sea un límite de palabra.
\d	Cualquier carácter dígito. Indica que puede haber cualquier carácter numérico, de 0 a 9.
\D	Carácter que no es dígito. Indica que puede haber cualquier carácter no numérico
\f	Salto de página. Este símbolo indica que tiene que haber un salto de página.
\n	Salto de línea. Este símbolo indica que tiene que haber un salto de línea.
\r	Retorno de carro indica que indica que tiene que haber un retorno de carro.
\s	Cualquier espacio en blanco. Indica que tiene que haber un carácter individual de espacio en blanco: espacios, tabulaciones, saltos de página o saltos de línea.
\S	Carácter que no sea blanco. indica que tiene que haber cualquier carácter individual que no sea un espacio en blanco.
\t	Este símbolo indica que tiene que haber cualquier tabulación.
\w	Carácter alfanumérico. Indica que puede haber cualquier carácter alfanumérico.
\W	Carácter que no sea alfanumérico. indica un carácter no alfanumérico.

Expresiones regulares. patrones

caracteres especiales

^	Principio de entrada o línea: indica que las cadenas deberán comenzar por el siguiente carácter
$	Fin de entrada o línea: Indica que la cadena debe terminar por el elemento precedido al dólar.
*	Indica que el carácter anterior se puede repetir en la cadena 0 o más veces.
+	indica que el carácter anterior se puede repetir en la cadena una o más veces.
?	indica que el carácter anterior se puede repetir en la cadena cero o una vez.
.	El punto indica que puede haber cualquier carácter individual salvo el de salto de línea
x|y	La barra vertical indica que puede ser el carácter x o el y.
{n,[m]}	El carácter anterior a las llaves tiene que aparecer exactamente n veces, o entre n y m veces
[abc]	Puede aparecer cualquier carácter entre corchetes
[^abc]	Cualquier carácter que no sea el que está entre los corchetes

Sitio web para practicar con expresiones regulares https://regexr.com/

Expresiones regulares. Métodos

Método	Descripción
`exec`	Un método `RegExp` que ejecuta una búsqueda por una coincidencia en una cadena. Devuelve un array de información.
`test`	Un método `RegExp` que verifica una coincidencia en una cadena. Devuelve `true` o `false`.
`match`	Un método `String` que ejecuta una búsqueda por una coincidencia en una cadena. Devuelve un array de información o `null` si no existe coincidencia alguna.
`search`	Un método `String` que verifica una coincidencia en una cadena. Devuelve el índice de la coincidencia, o -1 si la búsqueda falla.
`replace`	Un método `String` que ejecuta una búsqueda por una coincidencia en una cadena, y reemplaza la subcadena encontrada con una subcadena de reemplazo.
`split`	Un método `String` que utiliza una expresión regular o una cadena fija para cortar una cadena y colocarlo en un array de subcadenas.

Expresiones regulares. Métodos

test

Sintaxis: ExpReg.test(cadena)

Devuelve true si existe coincidencia entre la expresión regular y la cadena

```
1 | var cadena = "hello world!";
2 | var result = /^hello/.test(cadena);
3 | console.log(result); // true
```

Expresiones regulares. Métodos

match

Sintaxis: cadena.match(ExpReg)

Devuelve un array con todas las ocurrencias de una expresión regular dentro de una cadena.

```
1  var cadena = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz";
2  var expresion = /[A-E]/gi;
3  var array_emparejamientos = cadena.match(expresion);
4  console.log(array_emparejamientos);
```

`array_emparejamientos` será `['A', 'B', 'C', 'D', 'E', 'a', 'b', 'c', 'd', 'e']`

https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Regular_Expressions#special-lookahead

https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_match_regexp

Expresiones regulares. Métodos

replace

Sintaxis: string.replace(cadena/ExprReg, cadenanueva)

Ejemplos:

```
1  var cadena = "Twas the night before Xmas...";
2  var nuevaCadena = cadena.replace(/xmas/i, "Christmas");
```

```
1  var str = "Apples are round, and apples are juicy.";
2  var newstr = str.replace("apples", "oranges", "gi");
```

```
1  var expresion = /(\w+)\s(\w+)/;
2  var cadena = "John Smith";
3  var nuevaCadena = cadena.replace(expresion, "$2, $1");
```

$1 y $2 son patrones de reemplazo nuevaCadena = “Smith, John”

https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos_globales/String/replace

https://www.w3schools.com/jsref/jsref_replace.asp


ESCUELA PÚBLICA:
DE TOD@S
PARA TOD@S
Vallecas