

Parte 2: Formularios

UD3: Interacción con el usuario y modelo de objetos del documento

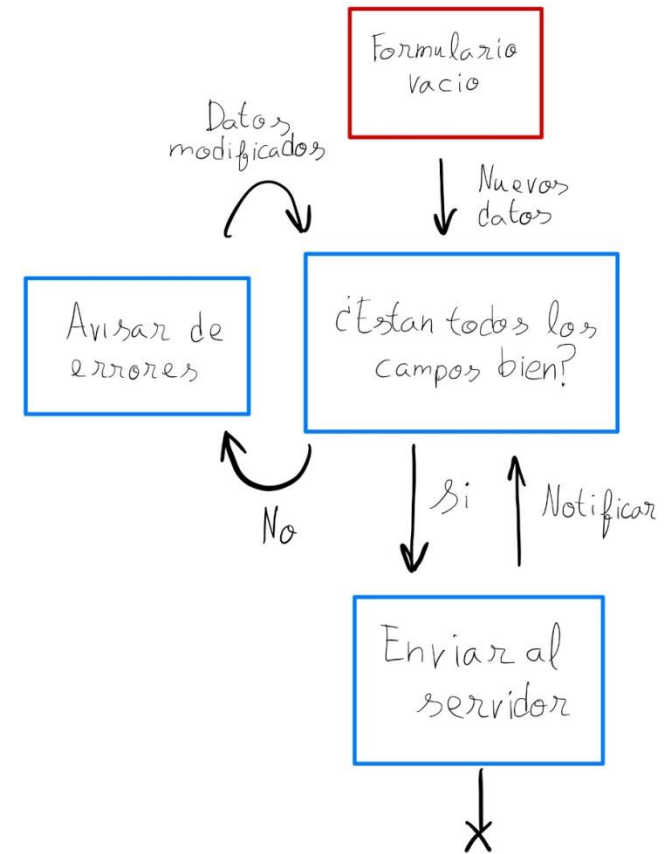
María Rodríguez Fernández mariafer@educastur.org

Al final de este documento...

- Habrás recordado los distintos elementos que puede haber en un formulario
 - Conectarás con tus conocimientos sobre HTML5
- Conocerás la forma de realizar la validación de estos elementos con JS
 - Para cada uno, repasaremos particularidades (sus principales propiedades y métodos)

Envío y validación de formularios

- Comprobar si todos los datos de un formulario han sido introducidos correctamente
 - Lado servidor (PHP, ASP, JSP...)
 - Más segura
 - Lado cliente (JS, HTML5)
 - **Más fácil y eficiente**



Importante

- **La validación con JavaScript no garantiza que los datos enviados son correctos**
 - Evitar mandar datos erróneos al servidor y hacer operaciones innecesarias
 - Pero es posible ...
 - **Deshabilitar el JavaScript del navegador**
 - Utilizar la consola del desarrollador para **cambiar estos datos**, por lo que no se puede asegurar su validez
- Por eso, **la seguridad en la validación de un formulario debe realizarse en el lado del servidor y en la base de datos**

Validación en JS

- ¿Cuándo se hace la validación?
 - A medida que vamos introduciendo datos en el formulario (campo a campo, evento `blur`, `keyUp...`)
 - Al pulsar el botón de envío (evento `submit`, `click...`)
- Tipos de validaciones:
 - Existencia
 - Tipo de datos
 - Patrones (email, fecha, DNI...)



Formularios HTML5

- Se facilita mucho el tratamiento de formularios
 - Se añaden 12 nuevos tipos de input y elementos de formulario que mejoran la experiencia de usuario y añaden validaciones (evitando el uso intensivo de JavaScript)
 - Si el navegador no los acepta se cambia por un campo de texto normal.
 - Los dispositivos móviles ofrecen teclados virtuales adaptados al tipo de campo.

Puedes comprobar qué navegadores admiten cada tipo en w3Schools:
https://www.w3schools.com/html/html_form_input_types.asp

Objeto Form

- Los formularios son el principal medio de **introducción de datos** en una aplicación Web, y el principal punto de **interactividad** con el usuario
- Los formularios y sus controles son objetos del **DOM** con propiedades únicas
 - Se representan mediante un objeto de tipo `Form` que se corresponde con la etiqueta `<form>` de HTML
 - Podemos acceder a los formularios a través de las funciones del DOM

Acceso a formularios

```
<form id="contactar">...</form>
```

- Algunos métodos de acceso:

```
/* Opción 1: conociendo el id */
```

```
var formulario=document.getElementById("contactar");
```

```
var formulario=document.querySelector("#contactar");
```

```
var formulario=document.forms["contactar"];
```

```
/* Opción 2: conociendo la posición */
```

```
var formulario=document.getElementsByTagName("form")[0];
```

```
var formulario=document.forms[0];
```

```
var formulario=document.querySelectorAll("form")[0];
```

```
var formulario=document.querySelector("form");
```


Objeto Form: Propiedades y métodos

- Propiedades:

| | |
|--------------------|---|
| action | Atributo action del formulario |
| elements [] | Arrays con los elementos del formulario |
| length | Número de elementos del formulario |
| name | Atributo name del formulario |

- Métodos:

| | |
|-----------------|-----------------------|
| reset() | Resetea el formulario |
| submit() | Envía el formulario |

Entradas de texto

```
<input type="text, password..." />
```

```
<textarea> </textarea>
```

- Propiedades básicas:

| | |
|--------------------|---|
| name | Nombre |
| readOnly | Atributo de sólo lectura |
| autofocus | Elemento que tiene el foco al cargar (sólo 1) |
| placeholder | Texto de ayuda |
| form | Referencia al formulario que lo contiene <input type="text" value="usuario@dominio.com"/> |
| type | Tipo (type= text, password, etc.) |
| value | Valor del campo de texto |

- Métodos:

| | |
|-----------------|-------------------------------|
| select() | Selecciona el valor del campo |
| focus() | Sitúa el foco en el campo |

Casillas de verificación y botones de radio

```
<input type="checkbox" />
```

```
<input type="radio" />
```

| | |
|----------------|---|
| checked | Estado, indica con un booleano si está marcado |
| name | Atributo name del checkbox . Si es un radio button : <ul style="list-style-type: none">- Todos los botones del grupo deben tener el mismo atributo name |
| value | Valor del checkbox/radio (atributo value) Es el texto asociado que se envía al procesar el formulario |

Ejemplo con casillas de verificación

Me gusta

```
<form >  
  <input type="checkbox" id="gusta" value="si"/>Me gusta  
  <input type="button" value="Enviar" />  
</form>
```

```
/* Para que el siguiente código se ejecute debe ser asignado a  
un evento */  
var gusta=document.querySelector("#gusta");  
//Equivalente a document.querySelector("input[name='gusta']");  
  
if(gusta.checked)  
  ...;  
else  
  ...;
```

Ejemplo con botones de radio

¿Qué quieres de comer: _____

Fabada Arroz con leche Cachopo

```
<form name="formulario">
  <fieldset>
    <legend>¿Qué quieres de comer:</legend>

    <input type="radio" name="comida" id="fabada" value="fabada" checked>
    <label for="fabada">Fabada</label>

    <input type="radio" name="comida" id="arroz-leche" value="arroz-leche">
    <label for="arroz-leche">Arroz con leche</label>

    <input type="radio" name="comida" id="cachopo" value="cachopo">
    <label for="cachopo">Cachopo</label>
  </fieldset>
</form>
```

```
document.querySelectorAll("input[name='comida']").forEach(
  (opcion)=>{if (opcion.checked) alert(opcion.value)}
);
```

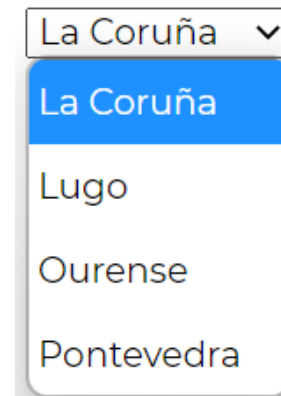
Lista de selección

```
<select> </select>
```

- Lista desplegable para seleccionar una o varias opciones
 - Opciones definidas con la etiqueta **option**
- En JS array de objetos **option** llamado **options**
 - `selectedIndex`
 - Índice (base 0) de la opción seleccionada.
 - Para cada objeto `option`:
 - **text**: Texto de la selección
 - **value**: Valor interno de la opción

Lista de selección: Ejemplo

```
<select name="provincias" id="provincias">
  <option value="CO">La Coruña</option>
  <option value="LU">Lugo</option>
  <option value="OU">Ourense</option>
  <option value="PO">Pontevedra</option>
</select>
```



```
var provincias=document.querySelector("#provincias");
var texto=provincias.options[provincias.selectedIndex].text;
var valor=provincias.options[provincias.selectedIndex].value;

console.log("Datos de la opción seleccionada:\n\nTexto:
"+texto+"\nValor:"+valor);
```

Input con opciones

- Se usa en campos de texto en conjunción con un elemento de tipo `<datalist>`
 - Atributo `list="id_datalist"` en el campo de texto
- Las opciones definidas en el `<datalist>` se muestran como una lista desplegable:

```
<input type="text" list="sugerencias" />
```

```
<datalist id="sugerencias">
```

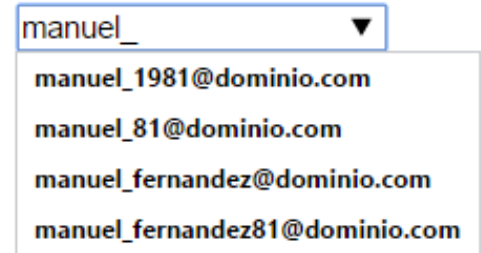
```
<option>manuel_1981@dominio.com</option>
```

```
<option>manuel_81@dominio.com</option>
```

```
<option>manuel_fernandez@dominio.com</option>
```

```
<option>manuel_fernandez81@dominio.com</option>
```

```
</datalist>
```



manuel_ ▼

- manuel_1981@dominio.com
- manuel_81@dominio.com
- manuel_fernandez@dominio.com
- manuel_fernandez81@dominio.com

EJERCICIO PROPUESTO-IV: Formulario (1 de 5)

- Dado el formulario proporcionado (**propuestoFormulario.html**)
 - Haz que los campos de texto se pasen a mayúscula al terminar de escribirlos
 - Comprueba:
 - Las contraseñas coinciden
 - Muestra por consola los datos introducidos al enviar el formulario mediante el botón

Formulario

Nombre completo

Fecha de nacimiento

DNI

Email

Contraseña

Contraseña (repetición)

Género Hombre Mujer

Suscribirme al boletín de novedades

Informarme sobre ofertas

Producto favorito

Comentario

EJERCICIO PROPUESTO-IV: Formulario (2 de 5)

| Elemento escuchado | Evento | Manejador |
|--------------------|--------|--|
| window | load | cargaPagina(): <ul style="list-style-type: none">• Asigna los eventos |
| Campo de texto | blur | salCampoTexto() <ul style="list-style-type: none">• Pasa a mayúsculas el campo |
| Campo contraseña 2 | blur | salPassword2() <ul style="list-style-type: none">• Mira que los passwords sean iguales |
| Botón de envío | click | validaFormulario(): lanza las validaciones y muestra los datos |

EJERCICIO PROPUESTO-IV: Formulario (3 de 5)

- **RETO EXTRA:** Diseña una clase para guardar objetos de la clase que encapsula el formulario e instancia un objeto de dicha clase cada vez que se pulse el botón de enviar datos.



Validación de formularios

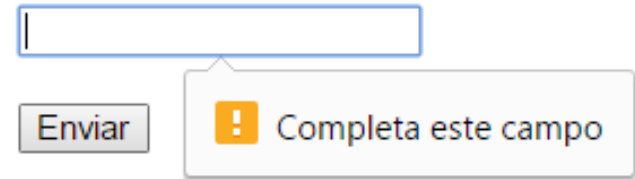
- Tradicionalmente era una de las principales funciones de JavaScript
 - Se accedía a los valores introducidos y se hacían comprobaciones "a mano"
- Validación avanzada desde HTML5
 - Se usa JS si se quiere controlar la apariencia de los mensajes de error



Validar obligatoriedad

- `required` [boolean]
 - El navegador no permite enviar este elemento si está vacío

```
<form>
  <label for="name">Nombre: </label>
  <input type="text" name="name"
        id="name" required
        autocomplete="off" >
  <button>Enviar</button>
</form>
```



- `autocomplete` [boolean]
 - Es recomendable usar `autocomplete="off"` para evitar que el autocompletador de los navegadores tape las validaciones
 - Por defecto activado

Validar tamaño y contenido

- Tamaño:
 - minlength / maxlength
 - Tamaño mínimo / máximo de caracteres del campo
- Contenido:
 - min, max [numérico]
 - En campos numéricos indica mínimo y máximo valor permitidos
- pattern [texto]
 - Permite introducir una **expresión regular** (usando sintaxis JavaScript) que el valor del campo debe cumplir
 - *Ejemplo: Alfanuméricos incluyendo espacios:*

```
<input type="text" pattern="^[a-zA-Z0-9 ]*$" />
```

Validación con JavaScript

- **checkValidity()**: método del input que comprueba si se cumplen los atributos. Si no se cumplen se puede comprobar cuál falla:
 - **valueMissing**
 - **rangeOverflow**
 - **tooLong**
 - **patternMismatch**
 - Otros: https://www.w3schools.com/js/js_validation_api.asp
- El atributo **validationMessage** ayuda al usuario
 - **setCustomValidity()**: permite personalizar el mensaje de validación
- El evento “**invalid**” se dispara cuando se ha comprobado la validez de un elemento que se puede enviar y no satisface sus restricciones

```
miInput.addEventListener('invalid', () => {  
    if(this.validity.valueMissing === '') {  
        // Campo vacío  
        miInput.setCustomValidity('No olvides tu nombre');  
    }  
    else {  
        // Patrón  
        miInput.setCustomValidity('Sólo caracteres alfanuméricos');  
    }  
});
```

Ejemplo: Validar con `checkValidity` (I)

- En el manejador del elemento que queremos validar llamamos a `checkValidity()`, y si devuelve falso, podemos ver qué error ha sucedido y tratarlo de forma diferente (*función error definida en la página siguiente*)

```
function validaEdad() {
    var elemento = document.querySelector("#edad");
    if (!elemento.checkValidity()) {
        if (elemento.validity.valueMissing)
            error(elemento)
        if (elemento.validity.rangeOverflow)
            error(elemento)
        if (elemento.validity.rangeUnderflow)
            error(elemento);
        return false;
    }
    return true;
}
```


Ejemplo: Validar con checkValidity (II)

```
<p id="mensajeError"></p>  
<p>
```

Párrafo vacío a continuación del formulario en el HTML para contener el mensaje de error

- La función "error":
 - Mostrará el mensaje correspondiente (`validationMessage` o mensaje personalizado) en el párrafo destinado a los errores
 - Se podría personalizar pasándolo como parámetro
 - La clase del elemento que dio el fallo pasa a ser "error" y para adquirir el estilo definido con CSS (ej. Rojo)
 - El elemento que dio el fallo recibe el foco

```
function error(elemento) {  
    document.querySelector("#mensajeError").innerHTML =  
                                                elemento.validationMessage;  
    elemento.className = "error";  
    elemento.focus();  
}
```


EJERCICIO PROPUESTO-IV: Formulario (4 de 5)

- Continua validando campos en el formulario:
 - Haz que el nombre y el email sea obligatorio añadiendo **required**
 - Personaliza el mensaje en el caso del nombre con la frase *"No seas tímido, dinos tu nombre"*
 - Haz que la contraseña tenga una longitud mínima de 8

Nombre

Apellidos

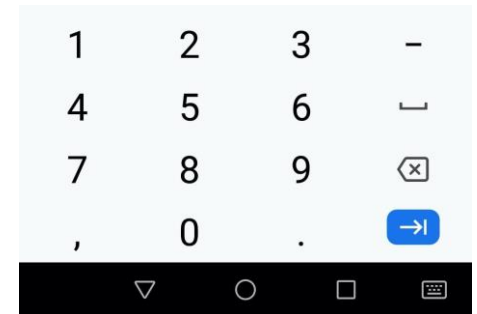
Fecha de

FER  Completa este campo

Tipos numéricos

```
<input type="number" />
```

- Indica al navegador que no debemos permitir el envío de un formulario si no hemos introducido un número
- `step [numérico]`
 - Indica el valor en que un campo aumenta/disminuye su valor
 - También compatible con otros tipos (date, time, range...)



```
<input type="range" />
```

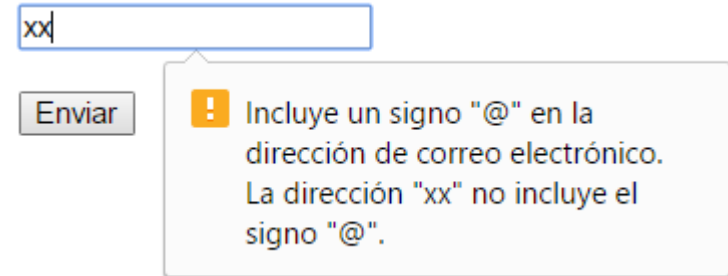
- Permite añadir de manera sencilla un deslizador
- Se suele combinar con los atributos `max`, `min` y `step`



Tipo email

```
<input type="email" />
```

- Indica al navegador que no debemos permitir el envío de un formulario si no hemos introducido una dirección válida.
 - No comprueba si la dirección existe



Tipo URL

```
<input type="url" />
```

- No permito el envío del formulario si no introducimos una URL correcta
 - Ojo, esto incluye el prefijo http:// [algunos navegadores lo añaden]



Tipos fecha, hora...

```
<input type="date" />
```

```
<input type="month" />
```

```
<input type="time" />
```

- Interfaces de usuario para introducir y validar fechas, meses, horas...
 - Formato [ISO](#)
 - Unificar códigos (habitualmente a criterio del programador)

| lu. | ma. | mi. | ju. | vi. | sá. | do. |
|-----|-----|-----|-----|-----|-----|-----|
| 26 | 27 | 28 | 29 | 30 | 1 | 2 |
| 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 17 | 18 | 19 | 20 | 21 | 22 | 23 |
| 24 | 25 | 26 | 27 | 28 | 29 | 30 |
| 31 | 1 | 2 | 3 | 4 | 5 | 6 |

sábado
1
OCT.
2016

octubre de 2016

| L | M | X | J | V | S | D |
|----|----|----|----|----|----|----|
| | | | | | 1 | 2 |
| 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 17 | 18 | 19 | 20 | 21 | 22 | 23 |
| 24 | 25 | 26 | 27 | 28 | 29 | 30 |
| 31 | | | | | | |

ELIMINAR CANCELAR ESTABLECER

octubre de 2016 X ▾

Establecer mes

| | |
|-------|------|
| sept. | 2015 |
| oct. | 2016 |
| nov. | 2017 |

ELIMINAR CANCELAR ESTABLECER

Existe un tipo week para escoger la semana del año (muy usado en países anglosajones)

23:-- X ▾

Establecer hora

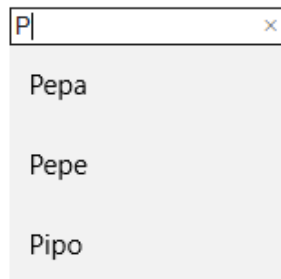
| | |
|---|----|
| 7 | 08 |
| 8 | 09 |
| 9 | 10 |

ELIMINAR CANCELAR ESTABLECER

Tipo búsqueda y barra de progreso

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

- Diferencia estética con una caja de texto corriente
- Añade un historial de búsquedas



```
<progress max="100"  
value="70">70%  
</progress>
```

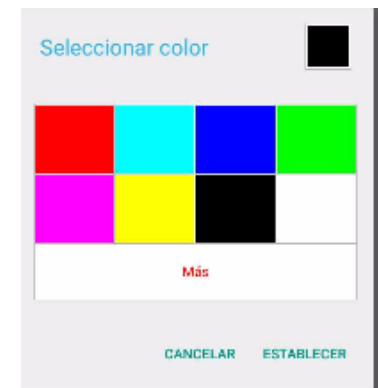
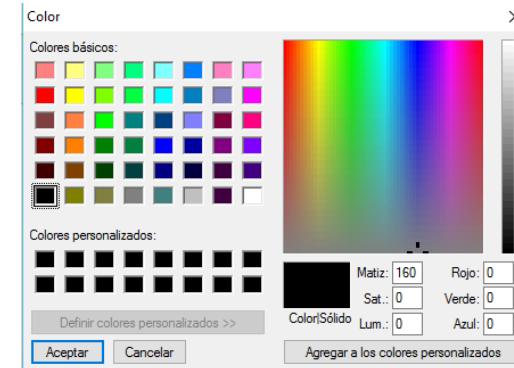
- Para representar un avance o progreso en la ejecución de una tarea que lleva tiempo
 - No tiene animación, habitualmente usaremos JavaScript para animarla
 - Atributos:
 - max (por defecto 1.0)
 - value (entre 0 y max 1.0)



Tipo color

```
<input type="color" />
```

- Muestra un cuadro de diálogo para escoger un color



EJERCICIO PROPUESTO-IV: Formulario (5 de 5)

- Continúa validando campos en el formulario:
 - Añade los elementos HTML5:
 - Email
 - Date
- Organiza el código para mejorar la gestión de errores del formulario
 - Almacena los códigos de error en una constante
 - Método "trataError" que reciba el campo y el código de error



Fecha de nacimiento dd/mm/aaaa

DNI

Email

Contraseña

Contraseña (repetición)

Género

Suscribirme

Informarme

Calendar: enero de 2022

| L | M | X | J | V | S | D |
|----|----|----|----|----|----|----|
| 27 | 28 | 29 | 30 | 31 | 1 | 2 |
| 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 17 | 18 | 19 | 20 | 21 | 22 | 23 |
| 24 | 25 | 26 | 27 | 28 | 29 | 30 |
| 31 | 1 | 2 | 3 | 4 | 5 | 6 |

Borrar Hoy

```
const CODIGOS_ERROR={  
  NOMBRE_VACIO:1,  
  PASSWORD_CORTO:2,  
  PASSWORDS_DISTINTOS:3,  
  EMAIL_TIPO:4  
};
```