

Comunicación Persona-Máquina

Formularios y Javascript

¿Qué es *Javascript*?

- Es un **lenguaje de programación** que permite en una página **interactuar con el usuario**
- Las instrucciones **se ejecutan en el navegador**, no en el servidor web,...
 - ... aunque se pueden complementar con instrucciones que se ejecutan en el lado del servidor,...
 - ... utilizando otros lenguajes como *PHP, ASP, etc,* ...
 - ... así el servidor es capaz de crear páginas sobre la marcha, que se transmiten al usuario.

Formularios: ¿Qué son?

- Es una forma de interactuar con el usuario
- Sirven para recoger datos que el usuario introduce...
 - ... que serán almacenados en una base datos para procesarlos posteriormente
- Todo el código ha de ir entre las etiquetas

<form></form>

- Con el parámetro **name** especificamos el nombre del formulario
- Para insertar elementos se utilizan la etiqueta **<input>**
 - Con las propiedades **name** y **type** especificamos el nombre y el tipo del elemento

Formularios: Ejemplo

```
<form name="miformulario">  
  <h1>Bienvenido</h1>  
  Introduce tu nombre: <input name="nombre" type="text" />  
  <br />Mensaje:  
  <input name="mensaje" type="textarea" size="60" />  
  <br /><input name="boton" type="submit" value="Probar"  
        />  
</form>
```

Formularios: Ejemplo

A screenshot of a web browser window titled "Formulario". The address bar shows the URL "file:///D:/ITEHTML/14formularios/eje/fig1401.html?nombre=Pepe". The page content displays the text "Bienvenido" in large bold letters. Below it are two input fields: one for "Introduce tu nombre:" and another for "Mensaje:". A "Probar" button is located at the bottom left.

Bienvenido

Introduce tu nombre:

Mensaje:

Probar

Parámetros (I)

- Se pueden añadir comportamientos a los formularios a través de los parámetros
 - La propiedad **action** indica dónde se enviará la información (generalmente una página .asp o .php del servidor o bien una aplicación del servidor).
 - Ejemplo
 - <form name="miformulario" **action="mailto:milogin@correo.es"** method="post" enctype="text/plain" target="_blank">
 - El formulario hará saltar la aplicación que el usuario tenga configurada por defecto para el correo

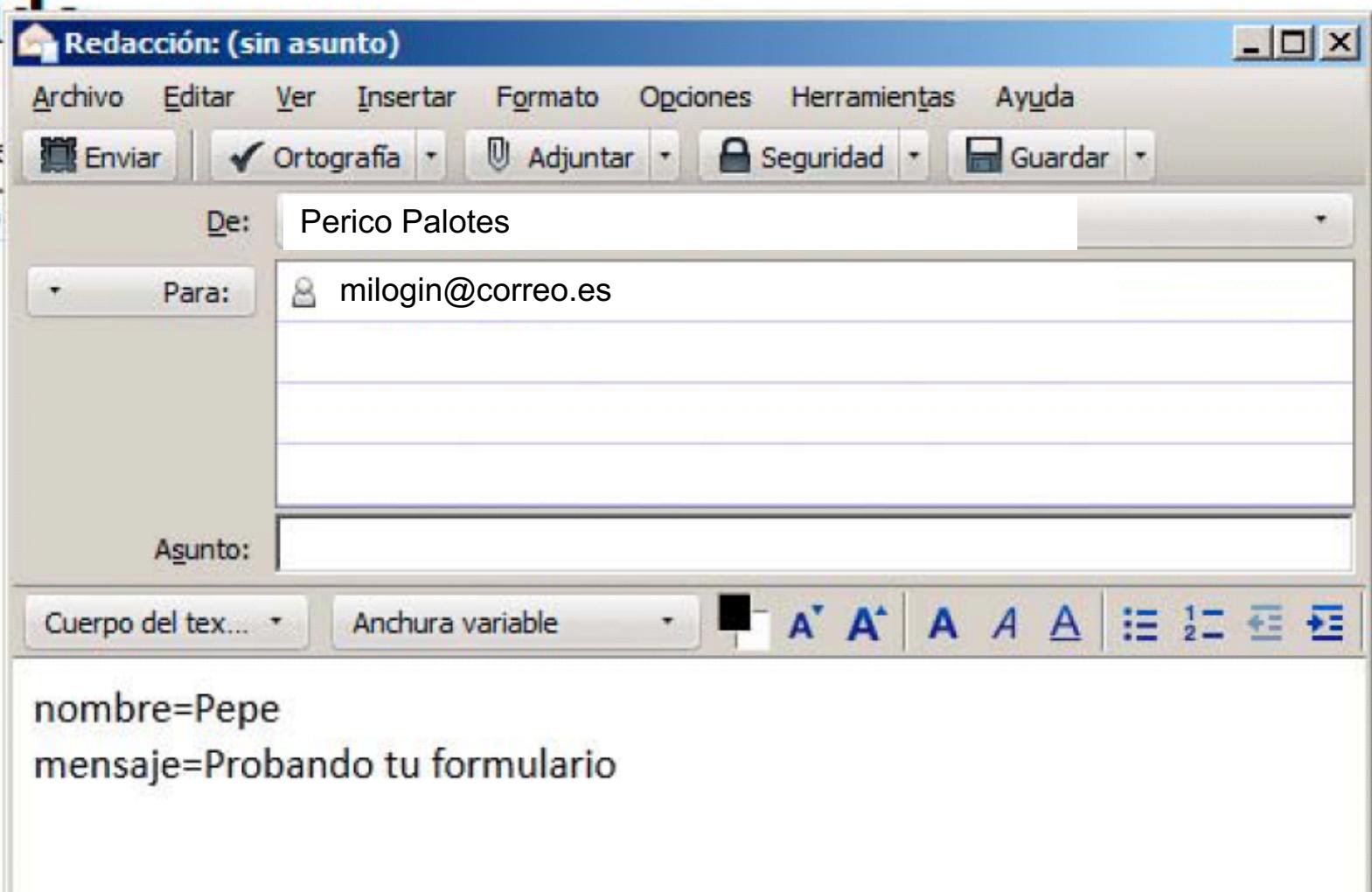
Parámetros (II)

Bienvenido

Introduce tu nombre

Mensaje: Probando

Probar



Parámetros (III)

- Junto con la propiedad **action** aparecen otras dos propiedades
 - **method**, que indica el formato de envío que se va a emplear para el formulario.
 - **target**, que indica donde se mostrará el resultado del formulario
 - **enctype**, se usa para indicar el formato de codificación de los datos que estamos remitiendo.

Parámetros (IV)

- La propiedad **method**, puede tener dos valores
 - El valor **post** remite la información de forma oculta y sin limitaciones de tamaño (es más recomendable)
 - El valor **get** transmite los datos como parte de la URL de la página. Es adecuado cuando se envían pocos datos.

Parámetros (V)

- La propiedad **enctype**, tiene tres valores:
 - El valor **application/x-www-form-urlencoded** hace que se codifiquen todos los caracteres antes de enviar (por defecto).
 - Se reemplacen los espacios por el signo +
 - Los caracteres especiales se convierten a valores hexadecimales de ASCII
 - El valor **multipart/form-data** es para enviar archivos, fotos, etc... hace que no se codifique ningún carácter
 - El valor **text/plain** hace que se codifiquen algunos caracteres.
 - Los espacios se reemplazan por el signo +
 - No se codifican los caracteres especiales

Parámetros (VI)

- Otras propiedades
 - La propiedad **autocomplete** (con valores **on** u **off**) que decide si al llenar el formulario se nos harán sugerencias (por defecto) o no.
 - Ponerlo en off es útil para formularios en los que se introducen datos personales
 - La propiedad **novalidate**, para que el formulario no se compruebe antes de enviarlo.

Campos

- Se introducen con la etiqueta **<input/>**...
 - ... y con las propiedades **name** y **type** especificamos su nombre y su tipo
 - Además, dependiendo del valor de **type** la etiqueta input tendrá o no otros parámetros

Ejemplos:

```
<input name="nombre" type="text" />  
<input name="mensaje" type="textarea" />  
<input name="boton" type="submit"  
value="Probar" />
```

Cuadros

- **Cuadro de texto (text)...**

- ... espacio para que el usuario introduzca texto libre
- Ejemplo:

```
<input name="nombre" type="text" />
```

- **Cuadro de contraseña (password)...**

- ... como los de texto pero se ven * al escribir
- Ejemplo:

```
<input name="clave" type="password" />
```

Opciones excluyentes (I)

- Para seleccionar uno de los valores del bloque, pero solo uno (son excluyentes) (**radio**)
- Ejemplo:

Selecciona el color:


```
<input name="elcolor" type="radio" value="rojo"  
checked> Rojo<br />
```

```
<input name="elcolor" type="radio" value="verde"/>Verde<br/>  
<input name="elcolor" type="radio" value="azul"/> Azul <br />
```

Opciones excluyentes (II)

- El valor del parámetro **name** ha de ser el mismo para todas las opciones
- El parámetro **value** recoge el valor que tendrá el campo cuando se marque una opción y se envíe el formulario
- El parámetro **checked** indica si inicialmente está activado

Casillas de verificación (I)

- Para seleccionar uno o más de los valores del bloque, (no son excluyentes) (**checkbox**)
- Ejemplo:

Selecciona tus colores favoritos:


```
<input type="checkbox" name="favorito" value="r" checked>  
Rojo<br />  
<input type="checkbox" name="favorito" value="v"/>  
Verde<br />  
<input type="checkbox" name="favorito" value="a" checked>  
Azul<br />
```

Casillas de verificación (II)

- El valor del parámetro **name** ha de ser el mismo para todas las opciones
- Se recogen los valores del parámetro **value** de las opciones marcadas cuando se envíe el formulario
- También goza del parámetro **checked**, pero en este caso es posible para más de una opción

Botones de envío y limpieza

- **Botón de envío (submit o image)**
 - Un botón que al ser pulsado se realizará la acción configurada en el formulario
 - Ejemplo:

```
<input type="submit" value="Enviar" />
<input type="image" src="miimagen.png" />
```
- **Botón de limpieza (reset)**
 - Un botón que al ser pulsado elimina toda la información rellenada en el formulario
 - Ejemplo:

```
<input type="reset" value="Borrar" />
```

Botones estándar

- Son botones genéricos en los que necesitamos definir un evento (típicamente **onclick**)
- Ejemplo:

```
<input type="button" value="Dale"  
onclick="JavaScript:window.location.assign('http://www.google.es')"/>
```

Notas:

- En este caso al pulsar el botón se cargará la página web de Google
- window.location es un objeto Javascript que se indica anteponiendo la palabra “JavaScript” y “:” y tiene el método assign para indicar una url.

Subir archivos

- Es posible subir archivos al servidor (**file**)
- Ejemplo:

Sube tu foto <input type="file" name="mifoto"/>

Valores ocultos

- Es posible enviar información al servidor pero sin que se muestre en pantalla (**hidden**)...
 - ... generalmente porque no se necesita que el usuario la proporcione (fecha del sistema, configuración del teclado, dirección IP, datos que se calculan a partir de otros...)
 - Ejemplo:
`<input type="hidden" name="pais" value="España"/>`

Tipos HTML5 (I)

- Con HTML5 aparecen más opciones para el parámetro **type**
 - date, datetime, datetime-local, month, week, time
 - ... para introducir fechas, fechas/horas, etc...
 - ... tiene el atributo step para saltar de step en step
 - number
 - ... para introducir un número
 - ... también con el atributo step
 - range
 - ...una barra indicando una escala donde se señala la posición, se puede indicar el mínimo y el máximo. También tiene step

Tipos HTML5(II)

- Con HTML5 aparecen más opciones para el parámetro **type**
 - color
 - ... para introducir un color
 - email
 - ... para introducir email
 - search
 - ... para introducir querys de búsqueda
 - tel
 - ... para introducir un teléfono
 - url
 - ... para introducir una url

Sin <input>: Áreas de texto

- Con las etiquetas **<textarea></textarea>**...
 - ... para escribir texto
 - ... con los parámetros **cols** y **rows** para indicar el tamaño visual
 - ... similar a **<input type="text">**
 - Tiene su propia etiqueta de cierre
 - Ejemplo:
`<textarea name="Sugerencias" cols="60" rows="6" /> Introduce
aquí tu sugerencia
</textarea>`

Sin <input>: Listas desplegables (I)

- Con las etiquetas **<select></select>**...
 - ... para seleccionar una opción
 - ... cada opción con la etiqueta **<option></option>**
 - ... similar a **<input type=“radio”/>**
 - ... pero más adecuado cuando hay muchas opciones
 - También tiene su propia etiqueta de cierre
 - Ejemplo:

```
<select> <option value=“r”> Rojo </option>
<option value=“v”> Verde </option>
<option value=“a”> Azul </option>
<option value=“n”> Naranja </option></select>
```

Sin <input>: Listas desplegables (II)

- Podemos establecer la opción predeterminada diferente de la primera
 - ... con el parámetro **selected**
 - Ejemplo:
 - `<option value="v" selected> Verde </option>`
- También podemos añadir un texto en blanco o indicaciones al principio
 - ... con el valor del parámetro value en blanco
 - Ejemplo:
 - `<option value=""> Selecciona un color </option>`

Sin <input>: Grupos de opciones

- Con las etiquetas **<optgroup></optgroup>...**
 - Se trata de agrupar las opciones de una lista desplegable
 - Al igual que el resto tiene su propia etiqueta de cierre
 - Ejemplo:

Selecciona un color

```
<select> <optgroup label="Básicos">
  <option value="ro"> Rojo </option>
  <option value="am"> Amarillo </option>
  <option value="az"> Azul </option></optgroup>
<optgroup label="Mezcla">
  <option value="na"> Naranja </option>
  <option value="vi"> Violeta </option>
  <option value="ve"> Verde </option></optgroup></select>
```

Paneles

- Con las etiquetas **<fieldset></fieldset>**...
 - Para agrupar varios elementos de forma visual
 - También con etiqueta de cierre
 - Con las etiquetas **<legend></legend>** podemos añadir un nombre al grupo
 - Ejemplo:
<fieldset>
<legend>Datos personales</legend>
Nombre: **<input name="nombre" type="text" />
**
Dirección: **<textarea name="direccion" cols="60" rows="2"></textarea>
**
</fieldset>

Parámetros comunes (I)

- Los más interesantes son:
 - **id** y **name**
 - **size**: define el tamaño
 - **maxlength**: establece la máxima longitud
 - **value**: valor por defecto
 - **required**: determina si el valor es obligatorio
 - **tabindex**: para numerar los elementos y establecer un orden con el tabulador

Parámetros comunes (II)

- Seguimos con más:
 - **form**: el id del form al que pertenece
 - **novalidate**: no se valida al enviar el formulario
 - **autocomplete**: valores on u off
 - **readonly**: no se puede modificar
 - **disabled**: deshabilita el elemento
 - **placeholder**: añade texto explicativo dentro del cuadro que desaparece al hacer clic sobre él

Parámetros comunes (III)

- El atributo **pattern=“patrón”**
 - Para los elementos cuyo *type* es: *text*, *email*,...
 - Donde *patrón* es una cadena que tiene los siguientes elementos especiales:
 - []: definen un conjunto de caracteres
 - { } : definen un rango de repeticiones
 - ?: opcional
 - *: o más
 - +: 1 o más
 - Más información:
 - https://www.w3schools.com/tags/att_input_pattern.asp
 - https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Regular_Expressions

Parámetros comunes (III)

Ejemplos de **pattern=“patron”**

- Uso de rangos ‘-’ en conjuntos:
 - 4 letras y 3 dígitos: “[A-Z] {4} [0-9] {3}”
- Uso de paréntesis (como en programación):
 - Matrícula (como la anterior sin vocales):
“([B-D] [F-H] [J-N] [P-T] [V-Z]) {4} [0-9] {3}”
- Cuantificadores:
Nº de teléfono de 9 a 12 dígitos con posibles espacios en blanco en medio y opcionalmente empieza por ‘+’: “\+? ([0-9] *) {9,12}”

\+ es el carácter ‘+’ en vez de un operador

“\+? ([0-9] *) {9,12}”

El operador '*' se aplica al espacio en blanco que le precede

Parámetros comunes: Ejemplo

- Ejemplo:

```
<input type="text" name="Sugerencias"  
maxlenght="9"  
size="9"  
readonly="readonly"  
disabled="disabled"  
placeholder="Introduce tus sugerencias"  
tabindex=1/>
```

También vale poner **readonly** y **disabled** sin valor

Parámetros comunes: Ejemplo

- Ejemplo:

```
<input type="text" name="Sugerencias"  
maxlenght="9"  
size="9"  
readonly="readonly"  
disabled="disabled"  
placeholder="Introduce tus sugerencias"  
tabindex=1/>
```

También vale poner **readonly** y **disabled** sin valor

JavaScript 1: estructura

La **estructura** del código JS se puede incluir en el head y es como la del ejemplo:

```
<script language="JavaScript">  
    function writeNumber(Sid,Pid) {  
        var Sel=document.getElementById(Sid);  
        var P=document.getElementById(Pid);  
        P.innerHTML=Sel.value;  
    }  
</script>
```

Si el código está en un fichero .js se enlaza a otro HTML usando:

```
<script language="JavaScript" src="fichero.js"></script>
```

Se asocia una **función(writeNumber)** JS a un **evento(onchange)** de un **componente(SelectNumber)** la siguiente manera:

```
<select id="SelectNumber"  
onchange="JavaScript:writeNumber(this.id,'Number')">
```

Suponiendo que hay un párrafo cuyo id es 'Number'.

JavaScript 2: DOM y variables

Se puede acceder a todo el DOM de la página actual:

- https://www.w3schools.com/js/js_htmldom.asp
- <https://www.w3.org/2005/03/DOM3Core-es/introduccion.html>

document.getElementById: es el modo de acceder al objeto DOM a partir de su *id*.

innerHTML es el campo por el que se puede acceder al contenido HTML de un elemento.

Las **variables** se declaran con la palabra reservada **var**.

Solo se indica su nombre, el tipo y el valor varían en tiempo de ejecución.

El operador **==** compara si dos elementos tienen el mismo tipo y valor.

El operador **==** compara si dos elementos tienen el mismo valor.

Se suelen utilizar comillas dobles en HTML y simples en JS.

JavaScript 3: números

Todos los parámetros que se pasan a JS en HTML son de tipo cadena. Si se ha de pasar un número se pasará éste como cadena.

Para convertir un número a cadena se utiliza: `new Number()`.

Ejemplo: `Var num=new Number('123');`

Si `new Number(cadena)` encuentra una cadena que no representa un número retorna `NaN`.

Ejemplo: `Var n=new Number(cadena);
If(isNaN(n)) { // Error`

Los `value` de los `input type="number"` solo retornan cadenas de números válidos. Si las cadenas introducidas por el usuario son inválidas (o vacías) retorna la cadena vacía. En este caso hay que comprobarlo antes, pues `new Number('')` retorna cero.

Ejemplo: `Var cad=document.getElementById('inputNumber').value;
if(cad) { // cad no es vacía
var n=new Number(cad)`

JavaScript 4: estilos

Se puede cambiar el **estilo** de los componentes usando el campo *style* del objeto.

Hay dos maneras:

- Usando un campo de *style*: `Obj.style.borderColor='blue'`
- Asignando a *style* una cadena tipo CSS: `Obj.style='border-color:blue'`

Los campos de *style* se pueden consultar en:

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

La asignación inicial de estilos en una web suele hacerse mediante CSS, pero si necesario hacerla mediante JS puede utilizarse el evento *onload* del objeto *body* para enlazar una función que ponga los estilos adecuados.

Javascript: Formularios y elementos

- En el vector **document.forms** se guardan todos los formularios de la página y en el vector **document.forms[i].elements** se guardan todos los elementos del formulario i-ésimo
- Aunque es mejor utilizar el nombre (**name**) o el identificador (**id**) del formulario o elemento

```
<form name="miformulario">  
  <input name="mielemento" type="text"/>  
</form>  
  
document.miformulario  
document.miformulario.mielemento
```

- O bien las funciones **document.getElementById(elId)** o **document.getElementsByName(elName)**

Utilidades: Valores de los campos(I)

- Cuadro de texto y textarea...
 - ... mediante el atributo **value**

```
<input type="text" id="cuadro"/>  
document.getElementById("cuadro").value
```

```
<textarea id="area"></textarea>  
document.getElementById("area").value
```

Utilidades: Valores de los campos (II)

- Radio...
 - ... mediante los atributos **value** y **checked**

```
<input type="radio" value="rojo" name="micolor" id="r"/> Rojo  
<input type="radio" value="verde" name="micolor" id="v"/> Verde  
<input type="radio" value="azul" name="micolor" id="a"/> Azul
```

```
radios=document.getElementsByName("micolor");  
radios[i].value                   radios[i].checked
```

```
radio=document.getElementById("r")  
radio.value       radio.checked
```

Utilidades: Valores de los campos(III)

- Checkbox...
 - ... mediante los atributos **value** y **checked**

```
<input type="checkbox" value="rojo" name="micolor" id="r"/> Rojo  
<input type="checkbox" value="verde" name="micolor" id="v"/> Verde  
<input type="checkbox" value="azul" name="micolor" id="a"/> Azul
```

```
casillas=document.getElementsByName("micolor");  
casillas[i].value                                   casillas[i].checked
```

```
casilla=document.getElementById("r")  
casilla.value    casilla.checked
```

Utilidades: Valores de los campos(IV)

- Listas desplegables...
 - ... mediante los atributos **selectedIndex**, **options**, **value** y **text**

```
<select id="micolor" name="micolor">  
    <option value="r"> Rojo </option>  
    <option value="v"> Verde </option>  
    <option value="a"> Azul </option></select>
```

```
lista=document.getElementById("micolor");  
indice=lista.selectedIndex;  
opcion=lista.options[indice];  
valor=opcion.value;  
texto=opción.text;
```

Utilidades: Establecer foco

- Con la función **focus()**...
 - ...siempre y cuando no sea de tipo **hidden**

```
If(elemento.type!="hidden"){  
    elemento.focus();  
}
```

Utilidades: Evitar el envío por duplicado

- Con un botón genérico...
 - ...y el evento **onclick** debe deshabilitarlo y...
 - ... enviar el formulario

```
<form name="miformulario">  
...  
<input name="boton" type="button" value="Probar"  
onclick="deshabilitaYenvia(this)"/>  
</form>
```

```
function deshabilitaYenvia(objeto){  
    objeto.disabled="true";  
    objeto.value="Enviando...";  
    objeto.form.submit();  
}
```

Eventos

- Es necesario definir dos cosas:
 - El evento
 - Ejemplo: Eventos de la etiqueta <body>
 - Carga y descarga: **onload**, **onbeforeunload** y **onunload**
 - Cambio de tamaño de una ventana: **onresize**
 - Impresión de una página: **onprint** y **onbeforeprint**
 - etc...
 - Qué hacer cuando tiene lugar el evento
 - Ejemplo: Mostrar un mensaje de alerta (con la función **alert** de JavaScript)

```
<body onload="JavaScript:alert('La página está cargada);">
```

Eventos de foco (I)

- Son dos:
 - **onfocus** (con foco)
 - **onblur** (sin foco)
- Ejemplo:

Nombre: <input name="nombre" id="nombre" type="text"
onfocus="JavaScript:EstaConFoco(this.id);"
onblur="JavaScript:EstaSinFoco (this.id);">

Notas:

- En este caso se llama a las funciones de JavaScript **EstaConFoco** y **EstaSinFoco** que previamente hemos de programar
- El parámetro **this.id** se refiere al id del objeto en cuestión

Eventos de foco (II)

- Ejemplo:

Nombre: <input name="nombre" id="nombre" type="text" onfocus="JavaScript:EstaConFoco(this.id); onblur="JavaScript:EstaSinFoco (this.id);">

- El código entre las etiquetas <**script**></**script**> dentro de las etiquetas <**head**></**head**> o en un fichero de extensión .js que se enlaza mediante la etiqueta <**link**>

```
<head>
...
<script language="JavaScript">
function EstaConFoco(id_objeto) {
document.getElementById(id_objeto).style.border="2px solid red";
}
function EstaSinFoco(id_objeto) {
document.getElementById(id_objeto).style.border="inherit";
}
</script>
</head>
```

Eventos de teclado (I)

- Cuando una tecla está presionada: **onkeydown**
- Liberar una tecla: **onkeyup**
- Presionar una tecla: **onkeypress**
- Ejemplo:

```
<body onkeypress="mostrar_tecla(event);">
```

Prueba de teclas:
</body>

Notas:

- Cada vez que pulsemos una tecla se mostrará en la parte identificada como “zonadeteclas”

Eventos de teclado (II)

- Ejemplo:

```
<body onkeypress="mostrar_tecla(event);">
```

Prueba de teclas:
</body>

- El código entre las etiquetas **<script></script>** dentro de las etiquetas **<head></head>** o en un fichero de extensión .js que se enlaza mediante la etiqueta **<link/>**

```
<head>...<script language="JavaScript">
function mostrar_tecla(e) {
if (e.keyCode) keycode=e.keyCode;
else keycode=e.which;
document.getElementById("zonadeteclas").innerHTML =
document.getElementById("zonadeteclas").innerHTML +
String.fromCharCode(keycode);
}</script></head>
```

Eventos de ratón

- Click, dobleclick o mover la rueda
 - **onclick, ondblclick, onmousesheel**
- Arrastrar y soltar un elemento de la página web
 - **ondrag, ondragend, ondragenter, ondragleave, ondragover, ondragstart, ondrop**
- Movimientos de ratón
 - **onmousedown, onmousemove, onmouseout, onmouseover, onmouseup**
- Desplazamiento vertical
 - **onscroll**

Más eventos

- Pulsar el botón de envío
 - **onsubmit**
- Pulsar botón de limpieza
 - **onreset**
- Introducir información en un cuadro
 - **oninput**
- Cambiar un valor
 - **onchange**

Ejemplo:

- Cuando cambie la edad, se comprueba si es mayor de 18 años

```
<p>Edad: <input name="edad" id="edad" type="number"
onchange="JavaScript:CalculaMayorEdad();"
      >
    Mayor de edad <span id="MayorEdad"></span></p>
```

- El código entre las etiquetas **<script></script>** dentro de las etiquetas **<head></head>** o en un fichero de extensión .js que se enlaza mediante la etiqueta **<link/>**

```
<head>...<script language="JavaScript"> function CalculaMayorEdad() {
  Txt="NO";
  E=new Number(document.getElementById("edad").value);
  if(isNaN(E)) Txt="ERROR";
  if(E>=18) Txt="SI";
  document.getElementById("MayorEdad").innerHTML=Txt+" "+E;
</script></head>
```