

Unidad 5.- Interacción con el usuario. Eventos y formularios:

- a) Modelo de gestión de eventos.
- b) Utilización de formularios desde código.
- c) Modificación de apariencia y comportamiento.
- d) Validación y envío.
- e) Expresiones regulares.
- f) Utilización de cookies.

a) Modelo de gestión de eventos.

Eventos en HTML

Los eventos son una característica de los documentos HTML (presente en otros lenguajes de programación también) que permite a los autores agregar interactividad entre el sitio web y el visitante, al ejecutar programas del lado cliente cuando el visitante (u otro programa) realiza una acción. Por ejemplo, el autor puede hacer que un párrafo cambie de color de su texto cuando el visitante pose el puntero del mouse sobre el mismo.

Así como los atributos, los eventos pueden ser definidos fácilmente en la etiqueta de apertura del elemento, con el mismo formato:

`<nombre-etiquetaevento="código">`

El contenido del evento es el código que se ejecutará, y debe ser creado utilizando un lenguaje del lado cliente (por ejemplo, JavaScript) que debe ser soportado por el navegador para que funcione. En el ejemplo siguiente, definimos un párrafo que cambia el color de su texto a rojo cuando el mouse pasa por encima, y lo devuelve a negro cuando se retira.

001	<code><p onmouseover="this.style.color='red'" onmouseout="this.style.color='black'"></code>
002	<code>Este es un texto que cambia de color. Pruébalo!</p></code>

Lista de eventos

A continuación, hay una lista de todos los eventos disponibles para los estándares HTML y XHTML.

- ❖ **onload**: el evento "onload" es lanzado cuando el agente de usuario termina de cargar una página o todos los marcos en un ser de marcos. Este evento es exclusivo de los elementos HTML body y HTML frame.
- ❖ **onunload**: el evento "onunload" es disparado cuando el agente de usuario retira el documento de una ventana o marco. este evento es exclusivo de los elementos HTML body y HTML frameset.
- ❖ **onclick**: el evento "onclick" ocurre cuando se realiza un click sobre el elemento.
- ❖ **ondblclick**: el evento "ondblclick" es ejecutado cuando se hace un doble click sobre el elemento.
- ❖ **onmousedown**: el evento "onmousedown" es lanzado cuando el botón del mouse es presionado sobre el elemento (independientemente de que sea soltado o no).
- ❖ **onmouseup**: el evento "onmouseup" es disparado cuando el botón del mouse se suelta sobre el elemento.
- ❖ **onmouseover**: el evento "onmouseover" ocurre cuando el mouse es puesto sobre el elemento.
- ❖ **onmousemove**: el evento "onmousemove" es ejecutado cuando el mouse es movido mientras está sobre el elemento.
- ❖ **onmouseout**: el evento "onmouseout" es lanzado cuando el mouse se quita de encima de un documento.
- ❖ **onfocus**: el evento "onfocus" es disparado cuando un elemento recibe el enfoque, bien sea a través del mouse o por navegación tabulada. Este evento es exclusivo de aquellos elementos que pueden recibir el enfoque: HTML a, HTML area, HTML label, HTML input, HTML select, HTML textarea, y HTML button.
- ❖ **onblur**: el evento "onblur" ocurre cuando el elemento pierde el enfoque bien sea a través del mouse o por navegación tabulada. Este evento es exclusivo de aquellos elementos que pueden recibir el enfoque.

- ❖ **onkeypress:** el evento "onkeypress" es ejecutado cuando una tecla es presionada y luego soltada mientras el elemento tiene el enfoque. Este evento es exclusivo de aquellos elementos que pueden recibir el enfoque.
- ❖ **onkeydown:** el evento "onkeydown" es lanzado cuando una tecla es presionada (independientemente de que sea soltada o no) mientras el elemento tiene el enfoque. Este evento es exclusivo de aquellos elementos que pueden recibir el enfoque.
- ❖ **onkeyup:** el evento "onkeyup" es disparado cuando una tecla es soltada mientras el elemento tiene el enfoque. Este evento es exclusivo de aquellos elementos que pueden recibir el enfoque.
- ❖ **onsubmit:** el evento "onsubmit" ocurre cuando el formulario es enviado. Este evento es exclusivo del elemento HTML form.
- ❖ **onreset:** el evento "onreset" es ejecutado cuando el formulario es reestablecido a sus valores por defecto. Este evento es exclusivo del elemento HTML form.
- ❖ **onselect:** el evento "onselect" es lanzado cuando un usuario selecciona texto en un campo de texto. Este evento es exclusivo de los elementos HTML input y HTML textarea.
- ❖ **onchange:** el evento "onchange" es disparado cuando un control pierde el enfoque y su valor ha sido modificado desde que recibió el enfoque por última vez. Este evento es exclusivo de los elementos HTML input, HTML select y HTML textarea.

Evento	Elemento sobre el que se produce
onblur	a, area, button, input, label, select, textarea
onchange	input, select, textarea
onclick	Todos los elementos except applet, base, basefont, bdo, br, font, frame, frameset, head, html, iframe, isindex, meta, param, script, style, title
ondblclick	Todos los elementos except applet, base, basefont, bdo, br, font, frame, frameset, head, html, iframe, isindex, meta, param, script, style, title
onfocus	a, area, button, input, label, select, textarea
onkeydown	Todos los elementos except applet, base, basefont, bdo, br, font, frame, frameset, head, html, iframe, isindex, meta, param, script, style, title
onkeypress	Todos los elementos except applet, base, basefont, bdo, br, font, frame, frameset, head, html, iframe, isindex, meta, param, script, style, title
onkeyup	Todos los elementos except applet, base, basefont, bdo, br, font, frame, frameset, head, html, iframe, isindex, meta, param, script, style, title
onload	frameset, body
onmousedown	Todos los elementos except applet, base, basefont, bdo, br, font, frame, frameset, head, html, iframe, isindex, meta, param, script, style, title
onmousemove	Todos los elementos except applet, base, basefont, bdo, br, font, frame, frameset, head, html, iframe, isindex, meta, param, script, style, title
onmouseout	Todos los elementos except applet, base, basefont, bdo, br, font, frame, frameset, head, html, iframe, isindex, meta, param, script, style, title
onmouseover	Todos los elementos except applet, base, basefont, bdo, br, font, frame, frameset, head, html, iframe, isindex, meta, param, script, style, title

Evento	Elemento sobre el que se produce
onmouseup	Todos los elementos except applet, base, basefont, bdo, br, font, frame, frameset, head, html, iframe, isindex, meta, param, script, style, title
onreset	form
onselect	input, textarea
onsubmit	form
onunload	frameset, body

Además de estos eventos que son de HTML 4, HTML 5 incorpora otra serie de eventos, que la mayoría de los navegadores aún no han incluido en su especificación, pero aun así vamos a ponerles a continuación, y hay algún evento que se desaprueba su utilización:

Eventos de atributos de ventana

Eventos activados por el objeto de ventana (se aplica a la etiqueta <body>):

Evento	Descripción
onafterprint	Después de que el documento se imprima
onbeforeprint	Antes de que el documento se imprima
onbeforeunload	Antes de que el documento se descarge
onerror	cuando se produce un error
onhaschange	cuando el documento ha cambiado
onload	cuando se ha cargado la página
onmessage	cuando el mensaje se activa
onoffline	cuando el documento se queda sin conexión.
ononline	cuando el documento se pone en línea
onpagehide	cuando la ventana está oculta
onpageshow	cuando la ventana se hace visible
onpopstate	cuando se modifica el fichero histórico de la página.
onredo	cuando el documento ejecuta un rehacer
onresize	cuando la ventana del navegador cambia de tamaño
onstorage	cuando un área de almacenamiento web se actualiza
onundo	cuando el documento realiza un deshacer
onunload	cuando se cierra la ventana del navegador

Eventos del formulario

Eventos activados por acciones dentro de un formulario HTML (se aplica a casi todos los elementos HTML, pero es la más utilizada en los elementos de formulario):

Evento	Descripción
onblur	cuandopierde el foco

Evento	Descripción
onchange	cuando se modifica el valor del elemento
oncontextmenu	cuando se active un menú contextual
onfocus	cuando el elemento obtiene el foco
onformchange	cuando cambia un formulario
onforminput	cuando un formulario consigue entrada del usuario
oninput	cuando un elemento obtiene la entrada del usuario
oninvalid	cuando un elemento no es valido
onreset	No soportado en HTML5 cuando se hace click en reset
onselect	cuando se ha seleccionado texto de un elemento
onsubmit	cuando se envía un formulario

Eventos del teclado

Evento	Descripción
onkeydown	cuando se ha hundido una tecla
onkeypress	cuando se ha pulsado una tecla
onkeyup	cuando se ha soltado una tecla

Eventos del ratón o acciones similares del usuario.

Evento	Descripción
onclick	cuando se hace click en un elemento
ondblclick	cuando se hace doble click en un elemento
ondrag	cuando un elemento se arrastra
ondragend	cuando finaliza una operación de arrastrar
ondragenter	cuando un elemento ha sido arrastrado a un destino válido
ondragleave	cuando un elemento deja un destino válido
ondragover	cuando un elemento está siendo arrastrado sobre un destino válido
ondragstart	cuando se inicia una operación de arrastrar
ondrop	cuando un elemento arrastrado es soltado
onmousedown	cuando se hunde el botón del ratón en un elemento
onmousemove	cuando se mueve o desplaza el ratón sobre un elemento.
onmouseout	cuando el puntero del ratón sale del elemento
onmouseover	cuando el puntero del ratón entra en el elemento.
onmouseup	cuando se suelta el botón del ratón en un elemento
onmousewheel	cuando se mueve la rueda del ratón.

Evento	Descripción
onscroll	cuando se está desplazando un elemento scrooll

Eventos multimedia

Eventos activados por medios como videos, imágenes y audio (se aplica a todos los elementos HTML, pero es más común en los elementos de los medios de comunicación, como <audio>, <embed>, , <object> y <video>):

Evento	Descripción
onabort	cuando se produce una interrupción o se aborta una acción.
oncanplay	cuando un archivo está listo para empezar a mostrarse o reproducirse.
oncanplaythrough	cuando un archivo se puede reproducir, sin pausas por el almacenamiento temporal.
ondurationchange	cuando cambia la duración de los medios
onemptied	cuando se produce un error y el archive no está disponible
onended	cuando ha finalizado la reproducción.
onerror	cuando se produce un error al cargar el archivo.
onloadeddata	cuando se lee un medio multimedia.
onloadedmetadata	cuando los metadatos del fichero se cargar (duración,...)
onloadstart	justo en el momento en que el fichero empieza a leerse
onpause	cuando se detiene la reproducción por el usuario o por programación.
onplay	cuando los medios están listos para reproducirse
onplaying	cuando ha comenzado la reproducción
onprogress	cuando el navegador está obteniendo los datos.
onratechange	cuando se cambia la velocidad de reproducción.
onreadystatechange	cuando cambia el estado de reproducción.
onseeked	cuando la búsqueda ha finalizado
onseeking	cuando se está realizando la búsqueda
onstalled	cuando el navegador no es capaz de recuperar los daros por cualquier motivo
onsuspend	cuando la búsqueda de los datos se detiene antes de que finalice.
ontimeupdate	cuando el punto de reproducción cambia
onvolumechange	cuando se cambia el volumen, incluso cuando se deja sin sonido
onwaiting	cuando se ha pausado o detenido de manera momentanea la reproducción.

Cada vez que se produce un evento se genera un objeto **event**

Objeto «event»

El objeto «event» permite obtener las características del evento que acaba de ocurrir. Sólo existe durante el tiempo en que se examinan sus consecuencias. Si se trata de acceder a él en otro momento se obtiene la constante especial «**null**». Se recomienda acceder al objeto event únicamente en el procedimiento asociado al evento.

Principales propiedades del objeto «event», vamos a ver algunas propiedades, existen más pero solo vemos unas cuantas, además en función del navegador utilizado nos podemos encontrar diferentes propiedades.

Propiedad	Tipo de datos	Elementos implicados	Descripción
type	Cadena	Todos	Nombre del evento: Uno de los enumerados en la lista que encabeza este capítulo, sin el prefijo «on».
target	Objeto	Todos	Elemento HTML que ha producido el evento.
cancelBubble	Buleano	Todos	Esta propiedad es modificable por el usuario. Su valor es false cuando un evento se produce, lo que significa que el ámbito es global. Si el usuario pone esta propiedad a true limita el ámbito. Esta propiedad es modificable por el usuario. Su valor es true cuando un evento se produce, esto significa que el tratamiento estándar del evento será efectuado después del tratamiento usuario. Si el usuario pone esta propiedad a false el tratamiento estándar no será efectuado. NO USAR. SE DEBE USAR event.preventDefault() ;
offsetX,offsetY	Entero	Ratón	Coordenadas del cursor dentro del elemento HTML que haya provocado el evento.
clientX, clientY	Entero	Ratón	Coordenadas del cursor dentro del área del navegador, sin tener en cuenta las barras de desplazamiento.
screenX,screenY	Entero	Ratón	Coordenadas del cursor dentro de la pantalla.
button	Entero	Ratón	Botones del ratón pulsados al ocurrir el evento: 1=botón izquierdo (principal), 2 = botón derecho, (secundario o alternativo), 4 = botón central. El valor de event.buttones la suma de los valores correspondientes a los botones pulsados. Por ejemplo, si se pulsa simultáneamente los botones izquierdo y central se obtiene 5, 0 si no se pulsa ninguno.
fromElement	Objeto	Ratón	Elemento HTML anterior a

Propiedad	Tipo de datos	Elementos implicados	Descripción
			(mouseoveromouseout).
toElement	Objeto	Ratón	Elemento HTML posterior a (mouseoveromouseout).
shiftKey	Buleano	Ratón o teclado	Pulsación tecla «Shift» (mayúsculas).
ctrlKey	Buleano	Ratón o teclado	Pulsación tecla «Control».
altKey	Buleano	Ratón o teclado	Pulsación tecla «Alt».
keyCode	Entero	Teclado	Valor ASCII de la tecla pulsada.
charCode	Entero	Teclado	Valor Tecla
key	Cadena	Teclado	Carácter de la tecla pulsada
char	Cadena	Teclado	Carácter de la tecla pulsada

Los valores correspondientes al teclado pueden varias de un navegador a otro:

001	Nombre <input name="nombre" onkeypress="controlar(event);" type="text"/>
002	Primer Apellido <input name="papellido" onkeyup="controlar(event);" type="text"/>
003	Segundo Apellido <input name="segape" onkeydown="controlar(event);" type="text"/>

001	function controlar(dato){
002	let variable="";
003	variable+="type "+dato.type+"\n";
004	variable+="keyCode "+dato.keyCode+"\n";
005	variable+="charCode "+dato.charCode+"\n";
006	variable+="char "+dato.char+"\n";
006	variable+="key "+dato.key+"\n";
007	console.log(variable);
008	}

Vamos a ver la ejecución de cada una de las estas líneas en cada uno de los navegadores principales.

En Firefox 17.0.1 veremos al pulsar la letrea "a"

letra "A"

type keypress	type keydown	type keyup	type keypress
keyCode 0	keyCode 65	keyCode 65	keyCode 0
charCode 97	charCode 0	charCode 0	charCode65
char undefined	char undefined	char undefined	char undefined
key undefined	key undefined	key undefined	key undefined

En Google Chrome23.0.1297.97 m veremos al pulsar la letrea "a" letra "A"

type keypress	type keydown	type keyup	type keypress
keyCode 97	keyCode 65	keyCode 65	keyCode65
charCode 97	charCode 0	charCode 0	charCode65
char undefined	char undefined	char undefined	char undefined
key undefined	key undefined	key undefined	key undefined

En Internet Explorer9.0.8112.16421 veremos al pulsar la letrea "a" letra "A"

typekeypress	type keydown	type keyup	type keypress
keyCode 97	keyCode 65	keyCode 65	KeyCode65
charCode 97	charCode 0	charCode 0	charCode65
char a	char a	char a	char A
key a	key a	key a	key A

En Opera 12.12 veremos al pulsar la letra "a"

letra "A"

typekeypress	type keydown	type keyup	type keypress
keyCode 97	keyCode 65	keyCode 65	KeyCode 65
charCode 97	charCode 0	charCode 0	charCode65
char a	char a	char a	char A
key a	key a	key a	key A

En Safari 5.1.7 veremos al pulsar la letra "a"

letra "A"

type keypress	type keydown	type keyup	type keypress
keyCode 97	keyCode65	keyCode 65	KeyCode 65
charCode 97	charCode 0	charCode 0	charCode65
char undefined	char undefined	char undefined	char undefined
key undefined	key undefined	key undefined	key undefined

Ámbito de un evento:

Cuando un evento ocurre en un elemento HTML, ocurre igualmente en los elementos que le engloban. Si, por ejemplo, un elemento <DIV> contiene un elemento , cuando el usuario hace clic sobre la imagen, ocurre un evento click que puede ser tratado a nivel del elemento como también a nivel del <DIV> que le contiene, y, por supuesto, a nivel <BODY> que engloba a ambos. La propiedad modificable **cancelBubble** trunca esta escalada, limitando el ámbito al del elemento en el que se coloque a true su valor, es la forma antigua de utilización hoy en día desaconsejada; en su lugar se debe utilizar **event.preventDefault();**

El evento es tratado en primer lugar por el elemento donde se desencadena. Existirán, pues, en nuestro ejemplo, los tratamientos consecutivos ligados a , <DIV> y <BODY>. Podríamos denominar a esto como «escalada o ámbito» del evento. Se podría llamar también «burbujeo».

A los métodos que tratan los eventos, cuando se incluye el evento en el documento HTML como un atributo de la etiqueta HTML, les vamos a poder pasar una serie de parámetros, que son **this** para hacer referencia al elemento sobre el que se produce el evento, **event** para hacer referencia al evento que se ha producido y en los elementos del formulario podemos pasarles también **this.form** para hacer referencia al formulario que contiene el elemento sobre el que se ha producido el evento.

Si deseamos desde javascript asignar un procedimiento a un evento correspondiente al formulario o a cualquier otro elemento del formulario deberemos poner.

window. *on*evento = nombre-metodo;

Observación solo se pone el nombre del método, se omiten los paréntesis ya que sino se ejecuta el método.

001	window.onload =cargado;
002	function cargado(){
003	alert ("página cargada correctamente");
004	}

window. *on*evento = function([*parámetro*]) {
 cuerpo de la función
}

001	<code>window.onload=function(){</code>
002	<code> alert("página cargada correctamente");</code>
003	<code>}</code>

También se puede poner

`document.nombre-formulario.nombre-elemento.onsuceso=metodo;`

O bien

```
document.nombre-formulario.nombre-elemento.onsuceso=function([parámetro])
{
    cuerpo de la función
}
```

Deberemos tener en cuenta que en estos casos vamos a poder poner un parámetro, que se va a corresponder con el objeto event correspondiente a ese evento.

En Internet Explorer 8 y versiones anteriores no entiende que se envíe el evento y en su lugar hay que utilizar para hacer referencia al evento **window.event**.

Con lo cual dentro de la función pondríamos:

001	<code>function tratar(evento){</code>
002	<code> let eventoNuevo= evento window.event;</code>
003	<code>}</code>

Con lo cual dentro de la función usaremos eventoNuevo.

ejemplo-05-100.html.

001	<code><!doctype html></code>
002	<code><html lang="es"></code>
003	<code> <head></code>
004	<code> <title>ejemplo-05-100 </title></code>
005	<code> <meta charset="utf-8"/></code>
006	<code> <script src="js/ejemplo-05-100.js" type="text/javascript"></script></code>
007	<code> <script type="text/javascript"></code>
008	<code> </script></code>
009	<code> <style type="text/css"></code>
010	<code> </style></code>
011	<code> </head></code>
012	<code> <body></code>
013	<code> <header></code>
014	<code> </header></code>
015	<code> <nav></code>
016	<code> </nav></code>
017	<code> <main></code>
018	<code> <section></code>
019	<code> <article></code>
020	<code> <div></code>
021	<code> <label for="codigo">Código Postal</label></code>
022	<code> <input type="text" name="codigo" maxlength="5"</code>
	<code> onkeypress="return solodigitos(event)" onfocus="activar(this)"</code>
	<code> onblur="valida(this)"/>
</code>
023	<code> <label for="ape">Apellidos</label></code>
024	<code> <input type="text" name="ape" onfocus="activar(this)"</code>
	<code> onblur="desactivar(this)" onkeypress="return sololetras(event)"/></code>
025	<code> </div></code>
026	<code> </article></code>
027	<code> </section></code>
028	<code> </main></code>
029	<code> <footer></code>
030	<code> </footer></code>
031	<code> <aside></code>
032	<code> </aside></code>

033	</body>
034	</html>

ejemplo-05-100.js.

001	function solodigitos(evento){
002	let valido=true;
003	let valor=String.fromCharCode(evento.charCode);
004	if(valor <"0" valor >"9")
005	valido=false;
006	return valido;
007	}
008	function sololetras(evento){
009	let valido=true;
010	let otros=new Array("ñ","á","é","í","ó","ú","ü");//Ñ vocales acentuadas y u con diéresis.
011	let valor=String.fromCharCode(evento.charCode).toLowerCase();
012	if(valor <"a" valor >"z")//letras alfabeto inglés
013	if(evento.keyCode!=8 && !otros.includes(valor))//tecla retroceso y letras españolas
014	valido=false;
015	return valido;
016	}
017	function valida(dato){
018	let numero=parseInt(dato.value,10);
019	if(numero <1000 numero >52999){
020	dato.value="dato no valido";
021	dato.style.backgroundColor="red";
022	dato.style.color="yellow";
023	dato.focus();
024	}else{
025	dato.style.backgroundColor="white";
026	dato.style.color="black";
027	}
028	}
029	function activar(elemento){
030	elemento.value="";
031	elemento.style.backgroundColor="blue";
032	elemento.style.color="white";
033	}
034	function desactivar(elemento){
035	elemento.style.backgroundColor="white";
036	elemento.style.color="black";
037	}

ejemplo-05-101.html.

001	<!doctype html>
002	<html lang="es">
003	<head>
004	<title>ejemplo-5-101</title>
005	<meta charset="utf-8"/>
006	<script src="js/ejemplo-05-101.js" type="text/javascript"></script>
007	<script type="text/javascript">
008	</script>
009	<style type="text/css">
010	</style>
011	</head>
012	<body>
013	<header>
014	</header>
015	<nav>
016	</nav>
017	<main>
018	<section>
019	<article>
020	<form name="formulario">
021	<label for="codigo">Código Postal</label>
022	<input type="text" name="codigo" maxLength="5" tabIndex="0" />
023	
023	<label for="ape">Apellidos</label>
024	<input type="text" name="ape" tabIndex="1"/>

025	</form>
026	<article>
027	<section>
028	</main>
029	<footer>
030	</footer>
031	<aside>
032	</aside>
033	</body>
034	</html>

Para este código HTML tenemos varias posibilidades de código javascript. Vamos a ver algunas de ellas.

ejemplo-05-101.js.

001	window.onload=cargado;
002	function cargado(){
003	document.formulario.codigo.onkeypress=solodigitos;
004	document.formulario.ape.onkeypress=sololettras;
005	document.formulario.codigo.onfocus=activar;
006	document.formulario.ape.onfocus=activar;
007	document.formulario.codigo.onblur=valida;
008	document.formulario.ape.onblur=desactivar;
009	}
010	function solodigitos(evento){
011	let valido=true;
012	let valor=String.fromCharCode(evento.charCode);
013	if(valor <"0" valor >"9")
014	valido=false;
015	return valido;
016	}
017	function sololettras(evento){
018	let valido=true;
019	let otros=new Array("ñ","á","é","í","ó","ú","ü");//Ñ vocales acentuadas y u con diéresis.
020	let valor=String.fromCharCode(evento.charCode).toLowerCase();
021	if(valor <"a" valor >"z")//letras alfabeto inglés
022	if(evento.keyCode!=8 && !otros.includes(valor))//tecla retroceso y letras españolas
023	valido=false;
024	return valido;
025	}
026	function valida(evento){
027	let elemento=evento.target;
028	let numero=parseInt(elemento.value,10);
029	if(numero <1000 numero >52999){
030	elemento.value="dato no valido";
031	elemento.style.backgroundColor="red";
032	elemento.style.color="yellow";
033	elemento.focus();
034	}else{
035	elemento.style.backgroundColor="white";
036	elemento.style.color="black";
037	}
038	}
039	function activar(evento){
040	let elemento=evento.target;
041	elemento.value="";
042	elemento.style.backgroundColor="blue";
043	elemento.style.color="white";
044	}
045	function desactivar(evento){
046	let elemento=evento.target;
047	elemento.style.backgroundColor="white";
048	elemento.style.color="black";
049	}

ejemplo-05-103.js

001	window.onload=function(){
002	document.formulario.codigo.onkeypress=solodigitos;
003	document.formulario.ape.onkeypress=sololettras;
004	document.formulario.codigo.onfocus=activar;

005	document.formulario.ape.onfocus=activar;
006	document.formulario.codigo.onblur=valida;
007	document.formulario.ape.onblur=desactivar;
008	}
009	function solodigitos(evento){
010	let valido=true;
011	let valor=String.fromCharCode(evento.charCode);
012	if(valor <"0" valor >"9")
013	valido=false;
014	return valido;
015	}
016	
017	function sololetras(evento){
018	let valido=true;
019	let otros=new Array("ñ","á","é","í","ó","ú","ü");//Ñ vocales acentuadas y u con diéresis.
020	let valor=String.fromCharCode(evento.charCode).toLowerCase();
021	if(valor <"a" valor >"z")//letras alfabeto inglés
022	if(evento.keyCode!=8 && !otros.includes(valor))//tecla retroceso y letras españolas
023	valido=false;
024	return valido;
025	}
026	function valida(evento){
027	let elemento=evento.target;
028	let numero=parseInt(elemento.value,10);
029	if(numero <1000 numero >52999){
030	elemento.value="dato no valido";
031	elemento.style.backgroundColor="red";
032	elemento.style.color="yellow";
033	elemento.focus();
034	}else{
035	elemento.style.backgroundColor="white";
036	elemento.style.color="black";
037	}
038	}
039	function activar(evento){
040	let elemento=evento.target;
041	elemento.value="";
042	elemento.style.backgroundColor="blue";
043	elemento.style.color="white";
044	}
045	function desactivar(evento){
046	let elemento=evento.target;
047	elemento.style.backgroundColor="white";
048	elemento.style.color="black";
049	}

ejemplo-05-105.js

001	window.onload=function(){
002	document.formulario.codigo.onkeypress=function(evento){
003	let valido=true;
004	let valor=String.fromCharCode(evento.charCode);
005	if(valor <"0" valor >"9")
006	valido=false;
007	return valido;
008	}
009	document.formulario.ape.onkeypress=function(evento){
010	let valido=true;
011	let otros=new Array("ñ","á","é","í","ó","ú","ü");//Ñ vocales acentuadas y u con diéresis.
012	let valor=String.fromCharCode(evento.charCode).toLowerCase();
013	if(valor <"a" valor >"z")//letras alfabeto inglés
014	if(evento.keyCode!=8 && !otros.includes(valor))//tecla retroceso y letras españolas
015	valido=false;
016	return valido;
017	}
018	document.formulario.codigo.onfocus=function(evento){
019	let elemento=evento.target;
020	elemento.value="";

021	elemento.style.backgroundColor="blue";
022	elemento.style.color="white";
023	}
024	document.formulario.ape.onfocus=function(evento){
025	let elemento=evento.target;
026	elemento.value="";
027	elemento.style.backgroundColor="blue";
028	elemento.style.color="white";
029	}
030	document.formulario.codigo.onblur=function(evento){
031	let elemento=evento.target;
032	let numero=parseInt(elemento.value,10);
033	if(numero <1000 numero >52999){
034	elemento.value="dato no valido";
035	elemento.style.backgroundColor="red";
036	elemento.style.color="yellow";
037	elemento.focus();
038	}else{
039	elemento.style.backgroundColor="white";
040	elemento.style.color="black";
041	}
042	}
043	document.formulario.ape.onblur=function(evento){
044	let elemento=evento.target;
045	elemento.style.backgroundColor="white";
046	elemento.style.color="black";
047	}
048	}

ejemplo-05-107.js

001	window.onload=cargado;
002	function cargado(){
003	document.formulario.codigo.onkeypress=function(evento){
004	let valido=true;
005	let valor=String.fromCharCode(evento.charCode);
006	if(valor <"0" valor >"9")
007	valido=false;
008	return valido;
009	}
010	document.formulario.ape.onkeypress=function(evento){
011	let valido=true;
012	let otros=new Array("ñ","á","é","í","ó","ú","ü");//Ñ vocales acentuadas y u con diéresis.
013	let valor=String.fromCharCode(evento.charCode).toLowerCase();
014	if(valor <"a" valor >"z")//letras alfabeto inglés
015	if(evento.keyCode!=8&&!otros.includes(valor))//tecla retroceso y letras españolas
016	valido=false;
017	return valido;
018	}
019	document.formulario.codigo.onfocus=function(evento){
020	let elemento=evento.target;
021	elemento.value="";
022	elemento.style.backgroundColor="blue";
023	elemento.style.color="white";
024	}
025	document.formulario.ape.onfocus=function(evento){
026	let elemento=evento.target;
027	elemento.value="";
028	elemento.style.backgroundColor="blue";
029	elemento.style.color="white";
030	}
031	document.formulario.codigo.onblur=function(evento){
032	let elemento=evento.target;
033	let numero=parseInt(elemento.value,10);
034	if(numero <1000 numero >52999){
035	elemento.value="dato no valido";
036	elemento.style.backgroundColor="red";
037	elemento.style.color="yellow";
038	elemento.focus();
039	}else{

040	elemento.style.backgroundColor="white";
041	elemento.style.color="black";
042	}
043	}
044	document.formulario.ape.onblur=function(evento){
045	let elemento=evento.target;
046	elemento.style.backgroundColor="white";
047	elemento.style.color="black";
048	}
049	}

ejemplo-05-102.html

001	<!doctype html>
002	<html lang="es">
003	<head>
004	<title>ejemplo 05-102 </title>
005	<meta charset="utf-8"/>
006	<meta name="author" value="Félix Ángel Muñoz Bayón"/>
007	<script type="text/javascript">
008	</script>
009	<style type="text/css">
010	</style>
011	</head>
012	<body>
013	<header>
014	</header>
015	<nav>
016	</nav>
017	<main>
018	<form name="formulario">
019	<label for="codigo">Código Postal</label>
020	<input type="text" name="codigo" maxlength="5" tabindex="0"/>
021	<label for="ape">Apellidos</label>
022	<input type="text" name="ape" tabindex="1"/>
023	</form>
024	</main>
025	<footer>
026	</footer>
027	<aside>
028	</aside>
029	<script src="js/ejemplo-05-102.js" type="text/javascript"></script>
030	</body>
031	</html>

Para este documento HTML vamos a poder utilizar los ficheros javascript anterior y además los siguientes ficheros.

ejemplo-05-102.js

001	document.formulario.codigo.onkeypress=function(evento){
002	let valido=true;
003	let valor=String.fromCharCode(evento.charCode);
004	if(valor <"0" valor >"9")
005	valido=false;
006	return valido;
007	}
008	document.formulario.ape.onkeypress=function(evento){
009	let valido=true;
010	let otros=new Array("ñ","á","é","í","ó","ú","ü");//Ñ vocales acentuadas y u con diéresis.
011	let valor=String.fromCharCode(evento.charCode).toLowerCase();
012	if(valor <"a" valor >"z")//letras alfabeto inglés
013	if(evento.keyCode!=8&&!otros.includes(valor))//tecla retroceso y letras españolas
014	valido=false;
015	return valido;
016	}
017	document.formulario.codigo.onfocus=function(evento){
018	let elemento=evento.target;
019	elemento.value="";
020	elemento.style.backgroundColor="blue";
021	elemento.style.color="white";
022	}
023	

024	document.formulario.ape.onfocus=function(evento){
025	let elemento=evento.target;
026	elemento.value="";
027	elemento.style.backgroundColor="blue";
028	elemento.style.color="white";
029	}
030	document.formulario.codigo.onblur=function(evento){
031	let elemento=evento.target;
032	let numero=parseInt(elemento.value,10);
033	if(numero <1000 numero >52999){
034	elemento.value="dato no valido";
035	elemento.style.backgroundColor="red";
036	elemento.style.color="yellow";
037	elemento.focus();
038	}else{
039	elemento.style.backgroundColor="white";
040	elemento.style.color="black";
041	}
042	}
043	document.formulario.ape.onblur=function(evento){
044	let elemento=evento.target;
045	elemento.style.backgroundColor="white";
046	elemento.style.color="black";
047	}
ejemplo-05-104.js	
001	document.formulario.codigo.onkeypress=solodigitos;
002	document.formulario.ape.onkeypress=sololettras;
003	document.formulario.codigo.onfocus=activar;
004	document.formulario.ape.onfocus=activar;
005	document.formulario.codigo.onblur=valida;
006	document.formulario.ape.onblur=desactivar;
007	function solodigitos(evento){
008	let valido=true;
009	let valor=String.fromCharCode(evento.charCode);
010	if(valor <"0" valor >"9")
011	valido=false;
012	return valido;
013	}
014	function sololettras(evento){
015	let valido=true;
016	let otros=new Array("ñ","á","é","í","ó","ú","ü");//ñ vocales acentuadas y u con diéresis.
017	let valor=String.fromCharCode(evento.charCode).toLowerCase();
018	if(valor <"a" valor >"z")//letras alfabeto inglés
019	if(evento.keyCode!=8&&!otros.includes(valor))//tecla retroceso y letras españolas
020	valido=false;
021	return valido;
022	}
023	function valida(evento){
024	let elemento=evento.target;
025	let numero=parseInt(elemento.value,10);
026	if(numero <1000 numero >52999){
027	elemento.value="dato no valido";
028	elemento.style.backgroundColor="red";
029	elemento.style.color="yellow";
030	elemento.focus();
031	}else{
032	elemento.style.backgroundColor="white";
033	elemento.style.color="black";
034	}
035	}
036	function activar(evento){
037	let elemento=evento.target;
038	elemento.value="";
039	elemento.style.backgroundColor="blue";
040	elemento.style.color="white";
041	}
042	function desactivar(evento){
043	let elemento=evento.target;
044	elemento.style.backgroundColor="white";

045	<code>elemento.style.color="black";</code>
046	<code>}</code>

Deberemos tener en cuenta que en el último documento HTML (ejemplo-05-102.html), al poner la inclusión del fichero javascript al final, todos los elementos del documento ya se han cargado con lo cual podemos hacer referencia a los mismos directamente para establecer los eventos que se van a controlar. Mientras que en el documento HTML anterior (ejemplo-05-101.html) como la inclusión del fichero javascript se hace dentro de la etiqueta HEAD, el documento HTML aún no se ha cargado y por consiguiente no se reconoce a los elementos del mismo, con lo cual deberemos esperar a que se cargue el documento para hacer referencia a los mismos y asignarles los eventos que queremos tratar. En nuestro caso, para que nos sirva para cualquier documento HTML independientemente en donde se incluya el código javascript vamos a utilizar alguna de las cuatro primeras formas.

b) Utilización de formularios desde código.

Cuando queremos hacer referencia a un formulario vamos a poner

```
document.forms.nombre-formulario
document.nombre-formulario
nombre-formulario
document.forms[posición]
document.forms["nombre-formulario"]
document.forms.item(posición)
document.forms.namedItem("nombre-formulario")
```

En cada uno de los casos para hacer referencia a los distintos elementos que forma parte del formulario deberemos agregar a la referencia del formulario punto y el nombre del elemento del formulario correspondiente, el valor del atributo **name**. Además, vamos a poder acceder a los elementos o atributos de los elementos del formulario.

Para acceder a los elementos de un formulario también tenemos la posibilidad de utilizar **elements**, que es un array con todos los elementos del formulario.

Cada uno de los elementos del array **elements** va a tener las siguientes propiedades:

1. **type** nos indique el tipo del elemento del formulario, los valores que puede tener son: checkbox, radio, text, textarea, password, button, reset, submit, image, select-one, select-multiplefile, hidden, file.
2. **name** nombre asignado al elemento.
3. **value** valor asignado al elemento.
4. cualquier otra propiedad del elemento indicado.

Para acceder a los diferentes elementos que constituyen el formulario vamos a poner

```
nombre-formulario.nombre-elemento
referencia-formulario.nombre-elemento
nombre-formulario.elements["nombre-elemento"]
referencia-formulario.elements["nombre-elemento"]
```

Y una vez que tenemos la referencia al elemento del formulario ya tenemos acceso a todos los atributos del mismo, poniendo punto y el nombre del atributo.

c) Modificación de apariencia y comportamiento.

Para modificar la apariencia del formulario vamos a poder utilizar los estilos mediante:

formulario.nombre-elemento.style.tipoestilo=valor

En donde tipo estilo va a ser el estilo que vamos a aplicar teniendo en cuenta que va todo junto, sin guiones y que las palabras compuestas van a estar en minúsculas excepto la inicial de la segunda palabra y sucesivas.

También se puede hacer modificando los atributos de un elemento del formulario para los cual podemos poner

formulario.nombre-elemento.nombre-atributo=valor

d) Validación y envío.

Cuando se valida un formulario se deben comprobar los campos, para ellos deberemos acceder al valor de los campos e ir comprobando si tiene valor y además ese valor es el adecuado, cuando todo está bien se envían los datos del formulario, sino no se deben enviar dichos datos.

Deberemos tener en cuenta que para validar las cajas de texto (**input type="text"**, **input type="password"** y **textarea**) vamos a utilizar el atributo **value**. Para validar los botones de selección (**input type="radio"**) y las casillas de verificación (**input type="checkbox"**) vamos a utilizar el atributo **checked**; si tienen el mismo nombre se utilizan como un array. Para una lista de valores (**select**), si solo se puede seleccionar uno utilizaremos el atributo **value**, mientras que si hay selección **multiple** se utiliza como un array y se consulta **selected**, para ver si está seleccionado y para obtener el valor usaremos **value**.

Para realizar la validación de los datos de un formulario vamos a poder elegir entre dos tipos de validaciones:

- Validaciones exhaustivas
- Validaciones con expresiones regulares.

En las validaciones exhaustivas vamos a ir comprobando carácter a carácter que el contenido del campo se adecua a las especificaciones que se nos han indicado.

En las validaciones con expresiones regulares, que veremos un poco más adelante, vamos a crear una expresión regular y mediante un método vamos a comprobar que se adecua al patrón establecido en la expresión regular.

Para el envío del formulario en el evento **onsubmit** podemos poner **onsubmit="return metodo([this])"**, en donde this hace referencia al formulario con lo cual en el método pondremos

```
function metodo([parámetro]) {  
}
```

Dentro de la función haremos referencia al formulario con el parámetro.

También se puede poner cuando hacemos referencia al evento de un elemento del documento **onevento="return método(this.form)"**

Si la función del evento **onsubmit** devuelve true se envía, mientras que cuando devuelve false no se envían.

Vamos a ver un formulario sencillo y su validación

ejemplo-05-120.html

001	<!doctype html>
002	<html lang="es">
003	<head>

004	<title>ejemplo-05-120 </title>
005	<meta charset="utf-8"/>
006	<meta name="author" value="Félix Ángel Muñoz Bayón"/>
007	<link href="css/ejemplo-05-120.css" rel="stylesheet" type="text/css"/>
008	<script src="js/ejemplo-05-120.js" type="text/javascript"></script>
009	<script type="text/javascript">
010	
011	</script>
012	<style type="text/css">
013	
014	</style>
015	</head>
016	<body>
017	<header>
018	</header>
019	<nav>
020	</nav>
021	<main>
022	<section>
023	<article>
024	<div id="datos">
025	<form name="formulario">
026	<label for="apellidos">Apellidos</label>
027	<input type="text" name="apellidos"/>
028	<label for="nombre">Nombre</label>
029	<input type="text" name="nombre"/>
030	<label for="direccion">Dirección</label>
031	<input type="text" name="direccion"/>
032	<label for="localidad">Localidad</label>
033	<input type="text" name="localidad"/>
034	<label for="codigopostal">Código Postal</label>
035	<input type="text" name="codigopostal" maxlength="5"/>
036	<label for="provincia">Provincia</label>
037	<input type="text" name="provincia"/>
038	<label for="comunidad">Comunidad Autónoma</label>
039	<select name="comunidad">
040	<option>Andalucía</option>
041	<option>Aragón</option>
042	<option>Asturias</option>
043	<option>Balears</option>
044	<option>Canarias</option>
045	<option>Cantabria</option>
046	<option>Castilla-La Mancha</option>
047	<option>Castilla-León</option>
048	<option>Cataluña</option>
049	<option>Extremadura</option>
050	<option>Galicia</option>
051	<option>Madrid</option>
052	<option>Murcia</option>
053	<option>Navarra</option>
054	<option>La Rioja</option>
055	<option>Valencia</option>
056	<option>País Vasco</option>
057	<option>Ceuta</option>
058	<option>Melilla</option>
059	</select>
060	<fieldset>
061	<legend>Estado civil</legend>
062	Soltero
063	<input type="radio" value="soltero" name="estadocivil">
064	Casado
065	<input type="radio" value="casado" name="estadocivil">
066	Viudo
067	<input type="radio" value="viudo" name="estadocivil">
068	Separado
069	<input type="radio" value="separado" name="estadocivil">
070	Divorciado
071	<input type="radio" value="divorciado" name="estadocivil">
072	Otros
073	<input type="radio" value="otros" name="estadocivil">

074	</fieldset>
075	<fieldset>
076	<legend>Aficiones</legend>
077	Bailar
078	<input type="checkbox" value="bailar" name="bailar"/>
079	HacerDeporte
080	<input type="checkbox" value="deporte" name="deporte"/>
081	Leer
082	<input type="checkbox" value="leer" name="leer"/>
083	Viajar
084	<input type="checkbox" value="viajar" name="viajar"/>
085	Escuchar música
086	<input type="checkbox" value="musica" name="musica"/>
087	Pintar
088	<input type="checkbox" value="pintar" name="pintar"/>
089	Escribir
090	<input type="checkbox" value="escribir" name="escribir"/>
091	Jugar a los videojuegos
092	<input type="checkbox" value="jugarvideojuegos" name="jugarvideojuegos"/>
093	</fieldset>
094	<input type="submit" value="Enviar" width="40px"/>
095	<input type="reset" value="Limpiar" width="40px"/>
096	</form>
097	<div>
098	</article>
099	</section>
100	</main>
101	<footer>
102	</footer>
103	<aside>
104	</aside>
105	</body>
106	</html>

ejemplo-05-120.css

001	#datos{
002	margin-top:10%;
003	margin-left:15%;
004	line-height:2em;
005	}
006	#datoslabel,filedset{
007	position:absolute;
008	left:50px;
009	}
010	#datos input[type="text"],select{
011	position:absolute;
012	left:300px;
013	}

ejemplo-05-120.js

001	window.onload= iniciar;
002	function iniciar(){
003	document.formulario.onsubmit=comprobar;
004	}
005	function comprobar(){
006	let enviar=true;
007	let mensaje="";
008	let masLetras=new Array("á","é","í","ó","ú","ñ","ü");
009	let adicionales=newArray("à","ò","_"," ");
010	let masDirec=newArray("a","e","-"," ","");
011	let vnom=document.formulario.nombre.value.toLowerCase().trim();
012	let vapell=document.formulario.apellidos.value.toLowerCase().trim();
013	let vdir=document.forms["formulario"].direccion.value.toLowerCase().trim();
014	let vloc=document.forms.elements["localidad"].value.toLowerCase().trim();
015	let vpos=document.forms.namedItem("formulario").codigopostal.value.trim();
016	let vpro=document.forms.item(0).elements["provincia"].value.toLowerCase().trim();
017	//comprobar nombre, empieza y termina por letra y en medio letras y blancos. Longitud mínima 3.
018	if(vnom.length<3){
019	enviar=false;
020	mensaje+="El nombre introducido no cumple los requisitos \n";

021	}else{
022	letra=vnom.charAt(0);
023	if((letra <"a" letra >"z") && (!masLetras.includes(letra))){
024	enviar=false;
025	mensaje+="El nombre introducido no cumple los requisitos \n";
026	}else{
027	letra=vnom.charAt(vnom.charAt(vnom.length-1));
028	if((letra <"a" letra >"z")&&(!masLetras.includes(letra))){
029	enviar=false;
030	mensaje+="El nombre introducido no cumple los requisitos \n";
031	}else{
032	let indice=1;
033	while(enviar && indice<vnom.length-1){
034	letra=vnom.charAt(indice);
035	if((letra <"a" letra >"z") && (!masLetras.includes(letra)) && (!adicionales.includes(letra))){
036	enviar=false;
037	mensaje+="El nombre introducido no cumple los requisitos \n";
038	}
039	indice+=1;
040	}
041	}
042	}
043	}
044	//comprobar apellidos
045	if(vapell.length<6){
046	enviar=false;
047	mensaje+="Los apellidos introducidos no cumplen los requisitos \n";
048	}else{
049	letra=vapell.charAt(0);
050	if((letra <"a" letra >"z")&&(!masLetras.includes(letra))){
051	enviar=false;
052	mensaje+="Los apellidos introducidos no cumplen los requisitos \n";
053	}else{
054	letra=vapell.charAt(vapell.charAt(vapell.length-1));
055	if((letra <"a" letra >"z") && (!masLetras.includes(letra))&&(letra <"0" letra >"9")){
056	enviar=false;
057	mensaje+="Los apellidos introducidos no cumplen los requisitos \n";
058	}else{
059	let indice=1;
060	while(enviar && indice<vapell.length-1){
061	letra=vapell.charAt(indice);
062	if((letra <"a" letra >"z") && (!masLetras.includes(letra)) && (!masDirec.includes(letra)) && (letra <"0" letra >"9")){
063	enviar=false;
064	mensaje+="Los apellidos introducidos no cumplen los requisitos \n";
065	}
066	indice+=1;
067	}
068	}
069	}
070	}
071	//comprobar dirección
072	if(vdir.length<6){
073	enviar=false;
074	mensaje+="La dirección introducida no cumple los requisitos \n";
075	}else{
076	letra=vdir.charAt(0);
077	if((letra <"a" letra >"z")&&(!masLetras.includes(letra))){
078	enviar=false;
079	mensaje+="La dirección introducida no cumple los requisitos \n";
080	}else{
081	letra=vdir.charAt(vdir.charAt(vdir.length-1));
082	if((letra <"a" letra >"z") && (!masLetras.includes(letra)) && (!adicionales.includes(letra))){
083	enviar=false;
084	mensaje+="La dirección introducida no cumple los requisitos \n";
085	}else{

086	let indice=1;
087	while(Enviar && indice<vdir.length-1){
088	letra=vdir.charAt(indice);
089	if((letra <"a" letra >"z")&&(!masLetras.includes(letra))&& letra!="
090	"){
091	Enviar=false;
092	mensaje+="La dirección introducida no cumple los requisitos \n";
093	}
094	indice+=1;
095	}
096	}
097	}
098	//comprobar localidad, empieza y termina por letra y en medio letras y blancos. Longitud
099	mínima 3.
100	if(vloc.length<3){
101	Enviar=false;
102	mensaje+="La localidad introducida no cumple los requisitos \n";
103	}else{
104	letra=vloc.charAt(0);
105	if((letra <"a" letra >"z")&&(!masLetras.includes(letra))){
106	Enviar=false;
107	mensaje+="La localidad introducida no cumple los requisitos \n";
108	}else{
109	letra=vloc.charAt(vloc.charAt(vloc.length-1));
110	if((letra <"a" letra >"z")&&(!masLetras.includes(letra))){
111	Enviar=false;
112	mensaje+="La localidad introducida no cumple los requisitos \n";
113	}else{
114	let indice=1;
115	while(Enviar && indice<vloc.length-1){
116	letra=vloc.charAt(indice);
117	if((letra <"a" letra >"z") && (!masLetras.includes(letra)) &&
118	letra!=" "){
119	Enviar=false;
120	mensaje+="La localidad introducida no cumple los requisitos \n";
121	}
122	indice+=1;
123	}
124	}
125	//comprobar provincia, empieza y termina por letra y en medio letras y blancos. Longitud
126	mínima 5.
127	if(vpro.length<5){
128	Enviar=false;
129	mensaje+="La provincia introducida no cumple los requisitos \n";
130	}else{
131	letra=vpro.charAt(0);
132	if((letra <"a" letra >"z")&&(!masLetras.includes(letra))){
133	Enviar=false;
134	mensaje+="La provincia introducida no cumple los requisitos \n";
135	}else{
136	letra=vpro.charAt(vpro.charAt(vpro.length-1));
137	if((letra <"a" letra >"z")&&(!masLetras.includes(letra))){
138	Enviar=false;
139	mensaje+="La provincia introducida no cumple los requisitos \n";
140	}else{
141	let indice=1;
142	while(Enviar && indice<vpos.length-1){
143	letra=vpro.charAt(indice);
144	if((letra <"a" letra >"z") && (!masLetras.includes(letra)) &&
145	letra!=" "){
146	Enviar=false;
147	mensaje+="La provincia introducida no cumple los requisitos \n";
148	}
149	indice+=1;
150	}

151	}
152	// comprobar código postal, cuatro o cinco dígitos y valor comprendido entra 1000 y 52999.
153	let escodigo=true;
154	if(vpos.length<4 vpos.length>5){
155	escodigo=false;
156	mensaje+="El código postal introducido no es correcto \n";
157	}else{
158	let indice=0;
159	let digito;
160	while(escodigo&&indice<vpos.length){
161	digito=vpos.charAt(indice);
162	if(digito<"0" digito >"9"){
163	escodigo=false;
164	mensaje+="El código postal introducido no es correcto \n";
165	}
166	indice++;
167	}
168	if(escodigo){
169	let vcopos=parseInt(vpos,10);
170	if(vcopos<1000 vcopos>52999){
171	escodigo=false;
172	mensaje+="El código postal introducido no es correcto \n";
173	}
174	}
175	}
176	enviar = enviar && escodigo
177	// comprobar que se ha seleccionaado una comunidad autonómica.
178	if(formulario.comunidad.value.length==0)
179	enviar=false;
180	// comprobar que se ha seleccionado un estado civil
181	let estado=false;
182	for(let i=0;i <formulario.estadocivil.length;i++)
183	estado = estado formulario.estadocivil[i].checked;
184	if(!estado)
185	mensaje+="Debe seleccionar al menos un estado civil \n";
186	enviar = enviar && estado;
187	// comprobar que se han seleccionado dos aficiones
188	let contador=0;
189	if(formulario.bailar.checked)contador++;
190	if(formulario.deporte.checked)contador++;
191	if(formulario.leer.checked)contador++;
192	if(formulario.viajar.checked)contador++;
193	if(formulario.musica.checked)contador++;
194	if(formulario.pintar.checked)contador++;
195	if(formulario.escribir.checked) contador++;
196	if(formulario.jugarvideojuegos.checked) contador++;
197	if(contador <2){
198	enviar=false;
199	mensaje+="Debe seleccionar al menos dos aficiones \n";
200	}
201	if(!enviar)
202	alert(mensaje);
203	return enviar;
204	}

Otro formulario y otra validación

ejemplo-05-121.html

001	<!doctypehtml>
002	<html lang="es">
003	<head>
004	<title>ejemplo-05-121 </title>
005	<meta charset="utf-8"/>
006	<meta name="author" value="Félix Ángel Muñoz Bayón" />
007	<link href="css/ejemplo-05-121.css" rel="stylesheet" type="text/css" />
008	<script src="js/ejemplo-05-121.js" type="text/javascript"></script>
009	<script type="text/javascript">
010	
011	</script>
012	<style type="text/css">

013	
014	</style>
015	</head>
016	<body>
017	<header>
018	</header>
019	<nav>
020	</nav>
021	<main>
022	<section>
023	<article>
024	<div id="datos">
025	<form name="formulario" onsubmit="return comprobar();">
026	<label for="apellidos">Apellidos</label>
027	<div>
028	<input type="text" name="apellidos"/>
029	<input type="text" name="errorapellidos" readonly
	class="error"/>
030	</div>
031	<label for="nombre">Nombre</label>
032	<div>
033	<input type="text" name="nombre"/>
034	<input type="text" name="errornombre" readonly
	class="error"/>
035	</div>
036	<label for="direccion">Dirección</label>
037	<div>
038	<input type="text" name="direccion"/>
039	<input type="text" name="errordireccion" readonly
	class="error" />
040	</div>
041	<label for="localidad">Localidad</label>
042	<div>
043	<input type="text" name="localidad"/>
044	<input type="text" name="errorlocalidad" readonly
	class="error"/>
045	</div>
046	<label for="codigopostal">Código Postal</label>
047	<div>
048	<input type="text" name="codigopostal" maxlength="5"/>
049	<input type="text" name="errorcodigopostal" readonly
	class="error"/>
050	</div>
051	<label for="provincia">Provincia</label>
052	<div>
053	<input type="text" name="provincia"/>
054	<input type="text" name="errorprovincia" readonly
	class="error"/>
055	</div>
056	<label for="comunidad">Comunidad Autónoma</label>
057	<select name="comunidad">
058	<option>Andalucia</option>
059	<option>Aragon</option>
060	<option>Asturias</option>
061	<option>Balears</option>
062	<option>Canarias</option>
063	<option>Cantabria</option>
064	<option>Castilla-La Mancha</option>
065	<option>Castilla-León</option>
066	<option>Cataluña</option>
067	<option>Extremadura</option>
068	<option>Galicia</option>
069	<option>Madrid</option>
070	<option>Murcia</option>
071	<option>Navarra</option>
072	<option>La Rioja</option>
073	<option>Valencia</option>
074	<option>País Vasco</option>
075	<option>Ceuta</option>
076	<option>Melilla</option>

077		</select>
078		<fieldset>
079		<legend>Estado civil</legend>
080	Soltero 	<input type="radio" value="soltero" name="estadocivil" />
081	Casado 	<input type="radio" value="casado" name="estadocivil" />
082	Viudo 	<input type="radio" value="viudo" name="estadocivil" />
083	Separado 	<input type="radio" value="separado" name="estadocivil" />
084	Divorciado 	<input type="radio" value="divorciado" name="estadocivil" />
085	Otros	<input type="radio" value="otros" name="estadocivil" />
086		</fieldset>
087		<fieldset>
088		<legend>Aficiones</legend>
089	Bailar 	<input type="checkbox" value="bailar" name="bailar" />
090	HacerDeporte 	<input type="checkbox" value="deporte" name="deporte" />
091	Leer 	<input type="checkbox" value="leer" name="leer" />
092	Viajar 	<input type="checkbox" value="viajar" name="viajar" />
093	Escucharmúsica 	<input type="checkbox" value="musica" name="musica" />
094	Pintar 	<input type="checkbox" value="pintar" name="pintar" />
095	Escribir 	<input type="checkbox" value="escribir" name="escribir" />
096	Jugar a los videojuegos	<input type="checkbox" value="jugarvideojuegos" name="jugarvideojuegos" />
097		</fieldset>
098		<input type="text" name="errorestadocivil" readonly class="error-2" />
099		<input type="text" name="erroraficiones" readonly class="error- 2" />
100		<input type="submit" value="Enviar" width="40px" />
101		<input type="reset" value="Limpiar" width="40px" />
102		</form>
103		</div>
104		</article>
105		</section>
106		</main>
107		<footer>
108		</footer>
109		<aside>
110		</aside>
111		</body>
112		</html>

ejemplo-05-121.css

001	#datos{
002	line-height:2em;
003	display:grid;
004	grid-template-columns:30% 60%;
005	grid-template-rows:repeat(10,1tr);
006	grid-auto-flow:row;
007	}
008	div input {
009	width:48%;
010	}
011	.error{
012	border:none;
013	color:red;
014	}
015	.error-2{
016	border:none;
017	color:red;
018	width:98%;

019	}
	<i>ejemplo-05-121.js</i>
001	function comprobar(){
002	let enviar= true ;
003	let masLetras= new Array("á","é","í","ó","ú","ñ","ü");
004	let adicionales= new Array("a","o","-"," ");
005	let masDirec= new Array("a","o","-"," ","");
006	let vnom=document.formulario.nombre.value.toLowerCase().trim();
007	let vapell=document.formulario.apellidos.value.toLowerCase().trim();
008	let vdir=document.forms["formulario"].direccion.value.toLowerCase().trim();
009	let vloc=document.formulario.elements["localidad"].value.toLowerCase().trim();
010	let vpos=document.forms.namedItem("formulario").codigopostal.value.trim();
011	let vpro=document.forms.item(0).elements["provincia"].value.toLowerCase().trim();
012	document.formulario.errornombre.value="";
013	document.formulario.errorapellidos.value="";
014	document.formulario.errordireccion.value="";
015	document.formulario.errorlocalidad.value="";
016	document.formulario.errorprovincia.value="";
017	document.formulario.errorcodigopostal.value="";
018	document.formulario.errorestadocivil.value="";
019	document.formulario.erroraficiones.value="";
020	//comprobar nombre, empieza y termina por letra y en medio letras y blancos. Longitud mínima 3.
021	if (vnom.length<3){
022	enviar= false ;
023	document.formulario.errornombre.value="Nombre no valido";
024	} else {
025	letra=vnom.charAt(0);
026	if ((letra <"a" letra >"z") && (!masLetras.includes(letra))){
027	enviar= false ;
028	document.formulario.errornombre.value="Nombre no valido";
029	} else {
030	letra=vnom.charAt(vnom.charAt(vnom.length-1));
031	if ((letra <"a" letra >"z") && (!masLetras.includes(letra))){
032	enviar= false ;
033	document.formulario.errornombre.value="Nombre no valido";
034	} else {
035	let indice=1;
036	while (enviar && indice<vnom.length-1){
037	letra=vnom.charAt(indice);
038	if ((letra <"a" letra >"z") && (!masLetras.includes(letra)) && (!adicionales.includes(letra))){
039	enviar= false ;
040	document.formulario.errornombre.value="Nombre no valido";
041	}
042	indice+=1;
043	}
044	}
045	}
046	}
047	//comprobar apellidos
048	if (vapell.length<6){
049	enviar= false ;
050	document.formulario.errorapellidos.value="Apellidos no validos";
051	} else {
052	letra=vapell.charAt(0);
053	if ((letra <"a" letra >"z") && (!masLetras.includes(letra))){
054	enviar= false ;
055	document.formulario.errorapellidos.value="Apellidos no validos";
056	} else {
057	letra=vapell.charAt(vapell.charAt(vapell.length-1));
058	if ((letra <"a" letra >"z") && (!masLetras.includes(letra)) && (letra <"0" letra >"9")){
059	enviar= false ;
060	document.formulario.errorapellidos.value="Apellidos no validos";
061	} else {
062	let indice=1;
063	while (enviar && indice<vapell.length-1){
064	letra=vapell.charAt(indice);
065	if ((letra <"a" letra >"z") && (!masLetras.includes(letra)) &&

	(!masDirec.includes(letra)) && (letra <"0" letra >"9")){
066	enviar=false;
067	document.formulario.errorapellidos.value="Apellidos no validos";
068	}
069	indice+=1;
070	}
071	}
072	}
073	}
074	//comprobar dirección
075	if(vdir.length<6){
076	enviar=false;
077	document.formulario.errordireccion.value="Dirección no valida";
078	}else{
079	letra=vdir.charAt(0);
080	if((letra <"a" letra >"z") && (!masLetras.includes(letra))){
081	enviar=false;
082	document.formulario.errordireccion.value="Dirección no valida";
083	}else{
084	letra=vdir.charAt(vdir.charAt(vdir.length-1));
085	if((letra <"a" letra >"z") && (!masLetras.includes(letra)) && (!adicionales.includes(letra))){
086	enviar=false;
087	document.formulario.errordireccion.value="Dirección no valida";
088	}else{
089	let indice=1;
090	while(enviar && indice<vdir.length-1){
091	letra=vdir.charAt(indice);
092	if((letra <"a" letra >"z") && (!masLetras.includes(letra)) && letra!=" "){
093	enviar=false;
094	document.formulario.errordireccion.value="Dirección no valida";
095	}
096	indice+=1;
097	}
098	}
099	}
100	}
101	//comprobar localidad, empieza y termina por letra y en medio letras y blancos. Longitud mínima 3.
102	if(vloc.length<3){
103	enviar=false;
104	document.formulario.errorlocalidad.value="Localidad no valida";
105	}else{
106	letra=vloc.charAt(0);
107	if((letra <"a" letra >"z") && (!masLetras.includes(letra))){
108	enviar=false;
109	document.formulario.errorlocalidad.value="Localidad no valida";
110	}else{
111	letra=vloc.charAt(vloc.charAt(vloc.length-1));
112	if((letra <"a" letra >"z") && (!masLetras.includes(letra))){
113	enviar=false;
114	document.formulario.errorlocalidad.value="Localidad no valida";
115	}else{
116	let indice=1;
117	while(enviar && indice<vloc.length-1){
118	letra=vloc.charAt(indice);
119	if((letra <"a" letra >"z") && (!masLetras.includes(letra)) && letra!=" "){
120	enviar=false;
121	document.formulario.errorlocalidad.value="Localidad no valida";
122	}
123	indice+=1;
124	}
125	}
126	}
127	}
128	//comprobar provincia, empieza y termina por letra y en medio letras y blancos. Longitud mínima 5.
129	if(vpro.length<5){

130	enviar=false;
131	document.formulario.errorprovincia.value="Provincia no valida";
132	}else{
133	letra=vpro.charAt(0);
134	if((letra <"a" letra >"z") && (!masLetras.includes(letra))){
135	enviar=false;
136	document.formulario.errorprovincia.value="Provincia no valida";
137	}else{
138	letra=vpro.charAt(vpro.charAt(vpro.length-1));
139	if((letra <"a" letra >"z") && (!masLetras.includes(letra))){
140	enviar=false;
141	document.formulario.errorprovincia.value="Provincia no valida";
142	}else{
143	let indice=1;
144	while(enviar && indice<vpos.length-1){
145	letra=vpro.charAt(indice);
146	if((letra <"a" letra >"z") && (!masLetras.includes(letra)) && letra!=" "){
147	enviar=false;
148	document.formulario.errorprovincia.value="Provincia no valida";
149	}
150	indice+=1;
151	}
152	}
153	}
154	}
155	// comprobar codigo postal, cuatro o cinco dígitos y valor comprendido entra 1000 y 52999.
156	let escodigo=true;
157	if(vpos.length<4 vpos.length>5){
158	escodigo=false;
159	document.formulario.errorcodigopostal.value="Código Postal no valido";
160	}else{
161	let indice=0;
162	let digito;
163	while(escodigo && indice<vpos.length){
164	digito=vpos.charAt(indice);
165	if(digito<"0" digito >"9"){
166	escodigo=false;
167	document.formulario.errorcodigopostal.value="Código Postal no valido";
168	}
169	indice+=1;
170	}
171	if(escodigo){
172	let vcopos=parseInt(vpos,10);
173	if(vcopos<1000 vcopos>52999){
174	escodigo=false;
175	document.formulario.errorcodigopostal.value="Código Postal no valido";
176	}
177	}
178	}
179	enviar = enviar && escodigo;
180	// comprobar que se ha seleccionad una comunidad autonómica.
181	if(formulario.comunidad.value.length==0)
182	enviar=false;
183	// comprobar que se ha seleccionado un estado civil
184	let estado=false;
185	for(let i=0 ; i <formulario.estadocivil.length ; i++){
186	estado = estado formulario.estadocivil[i].checked;
187	if(!estado)
188	document.formulario.errorestadocivil.value="Debe seleccionar un estado civil";
189	enviar = enviar && estado;
190	// comprobar que se han seleccionado dos aficiones
191	let contador=0;
192	if(formulario.bailar.checked) contador+=1;
193	if(formulario.deporte.checked) contador+=1;
194	if(formulario.leer.checked) contador+=1;
195	if(formulario.viajar.checked) contador+=1;
196	if(formulario.musica.checked) contador+=1;
197	if(formulario.pintar.checked) contador+=1;
198	if(formulario.escribir.checked) contador+=1;

199	<code>if(formulario.jugarvideojuegos.checked) contador+=1;</code>
200	<code>if(contador <2){</code>
201	<code> enviar=false;</code>
202	<code> document.formulario.erroraficiones.value="Debe seleccionar al menos dos aficiones";</code>
203	<code>}</code>
204	<code>return enviar;</code>
205	<code>}</code>

e) Expresiones regulares.

Las expresiones regulares las vamos a poder indicar de dos maneras, que son:

Creando un objeto de tipo RegExp, con la expresión regular.

Asignar la expresión regular, delimitada por el carácter "/" a una variable.

➔ Objeto **RegExp** crea expresiones regulares:

Las expresiones regulares son modelos que describen las combinaciones de caracteres en el texto. Se podrían definir como una serie de caracteres que forman un patrón, que representan a otro grupo de caracteres mayor, de tal forma que podemos comparar el patrón con otros conjuntos de caracteres para ver las coincidencias. Las expresiones regulares pueden utilizarse en múltiples lenguajes de programación, pero en esta entrada vamos a ver un ejemplo de validación de formularios mediante Javascript y haciendo uso de expresiones regulares.

Las expresiones regulares permiten realizar búsquedas complejas dentro de las cadenas de caracteres. Estas expresiones son, ellas mismas, unas cadenas balizadas por unos caracteres particulares según una sintaxis precisa. Esta sintaxis no es propia de JavaScript, encontramos variantes, por ejemplo, en comandos DOS de manejo de ficheros y Windows (por ejemplo «*.htm») y en comandos Unix como «grep» y «sed».

La tabla siguiente contiene los caracteres especiales de las expresiones regulares.

Carácter	Texto buscado
.	Cualquier carácter individual, salvo el de salto de línea.
x y	x o y.
[abc]	Cualquiera de los caracteres entre corchetes. Especifique un rango de caracteres con un guión (por ejemplo, [a-f] es equivalente a [abcdef]).
[^abc]	Cualquier carácter que no esté entre corchetes. Especifique un rango de caracteres con un guión (por ejemplo, [^a-f] es equivalente a [^abcdef]).
\b	Límite de palabra (como un espacio o un retorno de carro).
\B	Cualquiera que no sea un límite de palabra.
\d	Cualquier carácter de dígito. Equivalente a [0-9].
\D	Cualquier carácter que no sea de dígito. Equivalente a [^0-9].
\f	Salto de página.
\n	Salto de línea.
\r	Retorno de carro.
\s	Cualquier carácter individual de espacio en blanco (espacios, tabulaciones, saltos de página o saltos de línea).
\S	Cualquier carácter individual que no sea un espacio en blanco.
\t	Tabulación.
\w	Cualquier carácter alfanumérico, incluido el de subrayado. Equivalente a [A-Za-z0-9_].
\W	Cualquier carácter que no sea alfanumérico. Equivalente a [^A-Za-z0-9_].
\0	carácter nulo
\cX	Siendo X una letra, e índice control+letra.
\xhh	El carácter cuyo código en hexadecimal es xx.

Carácter	Texto buscado
\uxxxx	El carácter de código utf-16 hhhh, en hexadecimal
\u{hhhh}	El carácter de código utf-16 hhhh, en hexadecimal. Solo con el flag u in set.
\u{hhhhh}	igual anterior
^	Principio de entrada o línea.
\$	Fin de entrada o línea.
*	El carácter anterior 0 o más veces.
+	El carácter anterior 1 o más veces.
?	El carácter anterior una vez como máximo (es decir, indica que el carácter anterior es opcional).
{n}	Exactamente n apariciones del carácter anterior.
{n,m}	Como mínimo n y como máximo m apariciones del carácter anterior.
{n,}	Como mínimo n
(x)	Encuentra x y recuerda la búsqueda. Se llaman paréntesis de captura. Por ejemplo, /(foo)/ encuentra y recuerda 'foo' en "foo bar." El substring encontrado puede ser vuelto a llamar desde el array de elementos resultante [1], ..., [n] o desde las propiedades predefinidas del objeto RegExp\$1, ..., \$9.
(?:x)	Encuentra x pero no recuerda la búsqueda. Estos son llamados los paréntesis de no-captura. El substring encontrado no puede ser re-llamado desde el array de elementos resultante [1], ..., [n] o desde las propiedades predefinidas del objeto RegExp\$1, ..., \$9.
x(?:y)	Encuentra x solo si x es seguido de y. Por ejemplo, /Jack(?:=Sprat)/ encuentra 'Jack' solo si está seguido de 'Sprat'. /Jack(?:=Sprat Frost)/ encuentra 'Jack' solo si este es seguido de 'Sprat' o 'Frost'. Sin embargo, ni 'Sprat' ni 'Frost' forman parte del resultado encontrado.
x(?:!y)	Encuentra x solo si x no va seguido de y. Por ejemplo, /\d+(?!\.)/ encuentra un numero si esto no va seguido de un punto decimal. /\d+(?!\.).exec("3.141") encuentra 141 pero no 3.141.

La clase de objetos «RegExp»

Creación de una expresión regular

Un objeto de la clase RegExp puede ser creado y compilado de dos maneras:

Por «new», es decir con ayuda del constructor de objetos de la clase

RegExp: **let *nomExpresion* = new RegExp(*cadena* [, *claves*]);**

En este caso en la cadena cada vez que debemos poner el carácter “\” se debe poner duplicado.

ejemplo

```
001 var reg2=newRegExp("^\\d{1,2}\\d{3}$");
```

Por constante insertada:

```
var nomExpresion= /cadena/[claves];
```

«cadena» es una cadena de caracteres que constituye el cuerpo de la expresión regular.

«claves» es un cadena de caracteres que puede contener «g», «i» o los dos.

```
001 var reg2=/^\\d{1,2}\\d{3}$/;
```

«g» («global»): La expresión regular se aplica a toda la cadena objeto. Si no existe se detiene en la primera ocurrencia.

«i» («ignoreCase»): No diferenciar mayúsculas de minúsculas.

«m» multilinea, los caracteres de inicio y final se van a aplicar a varias líneas creadas con `\r` o `\n`.

«u» («unicode»): el patrón es una secuencia de códigos Unicode.

«y» («sticky»): compara solo en el índice buscado con `indexOf`.

La expresión regular constituida con la ayuda de cadena y claves es compilada y colocada en la variable `nomExpresion`.

«cadena» y «claves» pueden ser constantes o variables String normales en la primera sintaxis, pero son obligatoriamente constantes en la segunda sintaxis, además no necesitan estar encerradas entre comillas (ni dobles, ni simples) como otras cadenas JavaScript.

Una expresión regular está compuesta de cadenas a buscar y de caracteres que sirven de operadores entre las cadenas.

Si una cadena citada en la forma «new» es una constante, las eventuales «\» deberán estar duplicadas. En caso contrario JavaScript las interpretará inmediatamente sin efectuar el tratamiento de la expresión regular.

Si uno de los caracteres que puede servir de operador está presente con su significado real en una cadena citada con la sintaxis «constante insertada» o en una variable, deberá estar precedida de «\» para que no sea interpretado como un operador.

Métodos de las expresiones regulares

Método / Ejemplo	Resultado
<code>expReg.test(cadena)</code>	Valor lógico iguala true si es hallada una correspondencia en cadena mediante la expresión regular <code>expReg</code> .
<code>cadena.match(expReg)</code>	Matriz con las ocurrencias halladas en cadena con la expresión <code>expReg</code> indexadas a partir de cero. Al terminar la ejecución del método esta matriz contiene - en otra - ciertas propiedades del objeto <code>specialRegExp</code> (ver más abajo). Estas propiedades son: <code>input</code> , <code>index</code> , <code>lastIndex</code> .
<code>expReg.exec(cadena)</code>	Realiza una búsqueda en la cadena de la expresión regular y devuelve una matriz con la cadena encontradas.
<code>cadena1.replace(expReg, cadena2)</code>	Sustitución de las ocurrencias halladas en <code>cadena1</code> , mediante la expresión <code>expReg</code> , por <code>cadena2</code> .
<code>cadena.search(expReg)</code>	Índice (partiendo del índice cero) de la primera correspondencia hallada en cadena mediante la expresión <code>expReg</code> .

Método / Ejemplo	Resultado
<code>cadena.split(expReg)</code>	Matriz cuyos elementos son los trozos de cadena delimitados por las correspondencias halladas con la expresión <code>expReg</code> - excluidas.
<code>expReg.compile(cadena[,claves])</code>	Modificación de la expresión regular compilada <code>expReg(expRegmodificada)</code> es también devuelta como resultado del método.
<code>expReg.source</code>	Origen, fuente de la expresión regular compilada <code>expReg</code> , sin las claves.
<code>expReg.toSource()</code>	Devuelve una cadena con la expresión regular.
<code>expReg.lastIndex</code>	Índice donde empieza la expresión regular, después de una comprobación realizada con <code>exec</code> o <code>test</code> .
<code>expReg.flags</code>	Devuelve una cadena con los flags ordenados alfabéticamente
<code>expReg.global</code>	Indica si tiene el flag global
<code>expReg.ignoreCase</code>	Indica si tiene el flag <code>ignoreCase</code>
<code>expReg.multiline</code>	indica si tiene el flag <code>multilinea</code>
<code>expReg.sticky</code>	Indica si tiene el flag <code>multilinea</code>
<code>expReg.unicode</code>	Indica si tiene el flag Unicode
<code>expReg.source</code>	valor de la Expresión regular

De entre todos los métodos que hemos visto para validar formularios vamos a utilizar el primero de ellos, `test`. Vamos a ver un ejemplo de formulario validado con expresiones regulares

ejemplo-05-122.html

001	<code><!doctypehtml></code>
002	<code><html lang="es"></code>
003	<code><head></code>
004	<code><title>ejemplo-05-122 </title></code>
005	<code><meta charset="utf-8"/></code>
006	<code><meta name="author" value="Félix Ángel Muñoz Bayón"/></code>
007	<code><link href="css/ejemplo-05-122.css" rel="stylesheet" type="text/css"/></code>
008	<code><script src="js/ejemplo-05-122.js" type="text/javascript"></script></code>
009	<code><script type="text/javascript"></code>
010	
011	<code></script></code>
012	<code><style type="text/css"></code>
013	
014	<code></style></code>
015	<code></head></code>

016	<body>
017	<header>
018	</header>
019	<nav>
020	</nav>
021	<main>
022	<section>
023	<article>
024	<div id="datos">
025	<form name="formulario">
026	<label for="apellidos">Apellidos</label>
027	<input type="text" name="apellidos"/>
028	<label for="nombre">Nombre</label>
029	<input type="text" name="nombre"/>
030	<label for="direccion">Dirección</label>
031	<input type="text" name="direccion"/>
032	<label for="localidad">Localidad</label>
033	<input type="text" name="localidad"/>
034	<label for="codigopostal">Código Postal</label>
035	<input type="text" name="codigopostal" maxlength="5"/>
036	<label for="provincia">Provincia</label>
037	<input type="text" name="provincia"/>
038	<label for="comunidad">Comunidad Autónoma</label>
039	<select name="comunidad">
040	<option>Andalucía</option>
041	<option>Aragón</option>
042	<option>Asturias</option>
043	<option>Balears</option>
044	<option>Canarias</option>
045	<option>Cantabria</option>
046	<option>Castilla-La Mancha</option>
047	<option>Castilla-León</option>
048	<option>Cataluña</option>
049	<option>Extremadura</option>
050	<option>Galicia</option>
051	<option>Madrid</option>
052	<option>Murcia</option>
053	<option>Navarra</option>
054	<option>La Rioja</option>
055	<option>Valencia</option>
056	<option>País Vasco</option>
057	<option>Ceuta</option>
058	<option>Melilla</option>
059	</select>
060	<fieldset>
061	<legend>Estado civil</legend>
062	<input type="radio" value="soltero" name="estadocivil">
063	Soltero
064	<input type="radio" value="casado" name="estadocivil">
065	Casado
066	<input type="radio" value="viudo" name="estadocivil">
067	Viudo
068	<input type="radio" value="separado" name="estadocivil">
069	Separado
070	<input type="radio" value="divorciado" name="estadocivil">
071	Divorciado
072	<input type="radio" value="otros" name="estadocivil">
073	Otros
074	</fieldset>
075	<fieldset>
076	<legend>Aficiones</legend>
077	<input type="checkbox" value="bailar" name="bailar"/>
078	Bailar
079	<input type="checkbox" value="deporte" name="deporte"/>
080	HacerDeporte
081	<input type="checkbox" value="leer" name="leer"/>
082	Leer
083	<input type="checkbox" value="viajar" name="viajar"/>
084	Viajar
085	<input type="checkbox" value="musica" name="musica"/>

	Escucharmúsica
076	<input type="checkbox" value="pintar" name="pintar"/>
	Pintar
077	<input type="checkbox" value="escribir" name="escribir"/>
	Escribir
078	<input type="checkbox" value="jugarvideojuegos" name="jugarvideojuegos"/> Jugar a los videojuegos
079	</fieldset>
080	<input type="submit" value="Enviar" width="40px"/>
081	<input type="reset" value="Limpiar" width="40px"/>
082	</form>
083	</div>
084	</article>
085	</section>
086	</main>
087	<footer>
088	</footer>
089	<aside>
090	</aside>
091	</body>
092	</html>

ejemplo-05-122.css

001	#datos{
002	line-height:2em;
003	display:grid;
004	grid-template-columns:30% 60%;
005	grid-template-rows:repeat(9,1tr);
006	grid-auto-flow:row;
007	}

ejemplo-05-122.js

001	window.onload= iniciar;
002	function iniciar(){
003	document.formulario.onsubmit=comprobar;
004	document.formulario.nombre.onfocus=inicio;
005	document.formulario.apellidos.onfocus=inicio;
006	document.formulario.direccion.onfocus=inicio;
007	document.formulario.localidad.onfocus=inicio;
008	document.formulario.provincia.onfocus=inicio;
009	document.formulario.codigopostal.onfocus=inicio;
010	}
011	function inicio(evento){
012	let elemento=evento.target;
013	elemento.style.backgroundColor="white";
014	elemento.style.color="black";
015	elemento.value="";
016	}
017	function comprobar(){
018	let enviar=true;
019	let vnom=document.formulario.nombre.value.trim();
020	let vapell=formulario.apellidos.value.trim();
021	let vdir=document.forms["formulario"].direccion.value.trim();
022	let vloc=formulario.elements["localidad"].value.trim();
023	let vpos=document.forms.namedItem("formulario").codigopostal.value.trim();
024	let vpro=document.forms.item(0).elements["provincia"].value.trim();
025	let ernombre=/^[a-zAÉÍÓÜÑ][a-zAÉÍÓÜÑ* \-]+[a-zAÉÍÓÜÑ]\$/i
026	let erape=new RegExp("^[a-zAÉÍÓÜÑ][a-zAÉÍÓÜÑ* \-]{4,}[a-zAÉÍÓÜÑ]\$", "i")
027	let ercp=new RegExp("^(0?[1-9]) ([1-4]\\d) ([50-2])\\d{3}\$");
028	let erdir=/^[a-zAÉÍÓÜÑ][a-zAÉÍÓÜÑ* 0-9\\-]{4,}[a-zAÉÍÓÜÑ0-9]\$/i
029	let erloc=new RegExp("^[a-zAÉÍÓÜÑ][a-zAÉÍÓÜÑ* \-]+[a-zAÉÍÓÜÑ]\$", "i");
030	let erpro=/^[a-zAÉÍÓÜÑ][a-zAÉÍÓÜÑ* \-]{3,}[a-zAÉÍÓÜÑ]\$/i;
031	//comprobar nombre, empieza y termina por letra y en medio letras y blancos. Longitud mínima 3.
032	if(!ernombre.test(vnom)){
033	enviar=false;
034	formulario.nombre.style.backgroundColor="red";
035	formulario.nombre.style.color="yellow";
036	formulario.nombre.value="Nombre no valido";
037	}
038	//comprobar apellidos
039	if(!erape.test(vapell)){

040	enviar=false;
041	formulario.apellidos.style.backgroundColor="red";
042	formulario.apellidos.style.color="yellow";
043	formulario.apellidos.value="Apellidos no validos";
044	}
045	//comprobar dirección
046	if(!erdir.test(vdir)){
047	enviar=false;
048	formulario.direccion.style.backgroundColor="red";
049	formulario.direccion.style.color="yellow";
050	formulario.direccion.value="Dirección no valida";
051	}
052	//comprobar localidad, empieza y termina por letra y en medio letras y blancos. Longitud mínima 3.
053	if(!erloc.test(vloc)){
054	enviar=false;
055	formulario.localidad.style.backgroundColor="red";
056	formulario.localidad.style.color="yellow";
057	formulario.localidad.value="Localidad no valida";
058	}
059	
060	//comprobar provincia, empieza y termina por letra y en medio letras y blancos. Longitud mínima 5.
061	if(!erpro.test(vpro)){
062	enviar=false;
063	formulario.provincia.style.backgroundColor="red";
064	formulario.provincia.style.color="yellow";
065	formulario.provincia.value="Provincia no valida";
066	}
067	// comprobar codigo postal, cuatro o cinco dígitos y valor comprendido entra 1000 y 52999.
068	if(!erpc.test(vpos)){
069	enviar=false;
070	formulario.codigopostal.style.backgroundColor="red";
071	formulario.codigopostal.style.color="yellow";
072	formulario.codigopostal.value="Código Postal no valido";
073	}
074	// comprobar que se ha seleccinado una comunidad autonómica.
075	if(formulario.comunidad.value.length==0)
076	enviar=false;
077	// comprobar que se ha seleccionado un estado civil
078	let estado=false;
079	for(let i=0 ; i <formulario.estadocivil.length ; i++)
080	estado = estado formulario.estadocivil[i].checked;
081	enviar = enviar && estado;
082	// comprobar que se han seleccionado dos aficiones
083	let contador=0;
084	if(formulario.bailar.checked) contador+=1;
085	if(formulario.deporte.checked) contador+=1;
086	if(formulario.leer.checked) contador+=1;
087	if(formulario.viajar.checked) contador+=1;
088	if(formulario.musica.checked) contador+=1;
089	if(formulario.pintar.checked) contador+=1;
090	if(formulario.escribir.checked) contador+=1;
091	if(formulario.jugarvideojuegos.checked) contador+=1;
092	if(contador <2)
093	enviar=false;
094	return enviar;
095	}

f) Utilización de cookies.

Una cookie es un pequeño fichero de texto, almacenado por nuestro navegador en nuestro sistema, el cual almacena información que transmite al servidor en cada petición.

Su objetivo es dar información sobre nosotros al servidor web, para el cual, cada vez que accedemos a un recurso suyo somos un visitante distinto. Normalmente se suelen usar para guardar las preferencias de usuario.

Desventajas de usar cookies

- El límite de almacenamiento de cookies en los navegadores web se limita a alrededor de 4 KB.
- Las cookies se envían con cada petición HTTP, retrasando así el rendimiento de las aplicaciones web.

El funcionamiento de las cookies es sencillo. Al acceder a un recurso web se comprueba si poseemos una cookie para dicho dominio y path. Si existe se la enviamos al servidor en los encabezados de la petición. El servidor lee los datos almacenados en ella y de esta forma obtiene de nosotros la información que antes no tenía y, según el caso, puede enviar contenido personalizado para nuestro usuario.

Elementos de una cookie

Una cookie consiste de la siguiente información:

1. Un par nombre-valor que contiene el dato o datos almacenados.
2. Una fecha de expiración a partir de la cual la cookie será invalidada (y eliminada).
3. El dominio y el path del servidor a cuál enviará dicho dato.

1. nombre-valor

Cada cookie contiene un par nombre-valor que contiene el dato o datos que se desea almacenar. El nombre se utiliza como referencia para obtener el valor.

2. Fecha de expiración

Las cookies tienen una duración. Si no se especifica nada la cookie será válida mientras el navegador esté abierto.

3. Dominio y path

El dominio y el path indican el ámbito de validez de la cookie. Si no se especifica nada el ámbito será el dominio actual. En este caso www.mipagina.com.

Es importante tener en cuenta que no podemos crear una cookie en un dominio distinto del que nos encontremos.

Las cookies creadas en este dominio sólo se enviarán mientras visitemos páginas o recursos de este dominio. No se enviarán en el caso de otros subdominios, por ejemplo: catalogo.3sellers.com o promocion.3sellers.com, los cuales mantendrán sus propias cookies.

Las cookies en Chrome y Opera solo se pueden utilizar si estamos en un servidor, como localhost.

Con el path pasa lo mismo, podemos restringir desde que paths enviar nuestras cookies.

Por ejemplo, `/store/page/blog`.

Esta cookie sólo se enviaría en este path.

`document.cookie`

Las cookies se pueden leer usando Javascript, mediante el uso de la propiedad `"document.cookie"`.

Las operaciones básicas sobre una cookie son:

- crear
- actualizar
- leer
- borrar

Para crear una cookie deberemos, al menos, asignar valor a la clave e indicar la fecha de expiración de la cookie.


```
001 document.cookie="usuario=yo;expires=true, 30 Jan 2030 00:00:00 GMT;";
```

Para actualizar una cookie se hace igual que si creáramos una cookie.

```
001 document.cookie="usuario=yo;expires=true, 21 Jan 2030 00:00:00 GMT;";
```

Para leer debemos acceder a document.cookie que contiene el nombre y el valor de todas las cookies y buscar donde empieza la cookie que queremos y luego ya buscar donde termina. Con lo cual se debe poner para buscar el valor de la cookie usuario.

Leer una cookie, aquí si se incluye una cookie y luego se lee con la parte final de la cookie nos la reconoce.

```
001 var cadena=document.cookie;
002 var posicion=cadena.indexOf("usuario=");
003 var pos2=cadena.indexOf("=",posicion+1);
004 var pos3=cadena.indexOf(";",pos2+1);
005 if(pos3== -1){
006     pos3=cadena.length;
007 }
008 var usu=cadena.substring(pos2+1,pos3);
```

Mejora de leer una cookie, ya que aquí se comprueba exactamente la cookie que se quiere mirar

```
001 document.forms[0].mensaje.value="";
002 var nom=document.forms[0].nombre.value;
003 var cadena=document.cookie;
004 if (document.cookie.length>=0){
005     let posicion=cadena.indexOf(nom+"=");
006     if(posicion== -1){
007         document.forms[0].mensaje.value="no existe esa clave";
008     }else{
009         if(posicion!=0)
010             posicion=cadena.indexOf(";", "+nom+=");
011         if(posicion== -1){
012             document.forms[0].mensaje.value="no existe esa clave";
013         }else{
014             let pos2=cadena.indexOf("=",posicion+1);
015             let pos3=cadena.indexOf(";",pos2+1);
016             if(pos3== -1){
017                 pos3=cadena.length;
018             }
019             var usu=cadena.substring(pos2+1,pos3);
020             document.forms[0].valor.value=usu;
021         }
022     }
023 }else{
024     document.forms[0].mensaje.value="No hay claves";
025 }
```

Para borrar una cookie deberemos actualizar una cookie con la fecha de expiración anterior a la fecha de hoy.

```
001 document.cookie="usuario=yo;expires=true, 20 Jan 2013 00:00:00 GMT;";
```