Interacción con el usuario. Eventos y formularios

SOFIA VALERA FERNÁNDEZ

Contenido

[1. INTRODUCCION](#_Toc505166311)

[2. FORMULARIO REGISTRO](#_Toc505166312)

[3. FORMULARIO DE LOG IN](#_Toc505166313)

[4. COOKIES](#_Toc505166314)

# INTRODUCCION

La práctica trata sobre la creación de un formulario de registro y de log In, con validaciones a través de **HTML5**, **CSS3** y **JavaScript**. Para comprobar si el usuario esta registrado o no se hará a través de cookies, a continuación, explicare como lo he llevado a cabo:

# FORMULARIO REGISTRO

El formulario tiene que tener unos determinados campos, algunos obligatorios y otros opcionales, dependiendo de cada caso se comprobaran los datos.

Al final del **<form>,** habrá un input con type **submit** (enviar), que enviará todos los datos.

* **CAMPOS OBLIGATORIOS**
* Se dividen en: Nombre y Apellidos (Obligatorio). Compuesto como máximo por dos palabras
* Correo electrónico (Obligatorio)
* Contraseña de acceso (Obligatorio). Para que sea válida debe tener al menos ocho caracteres y contener al menos una letra minúscula, una mayúscula, un número y un símbolo
* Confirmación de contraseña (Obligatorio)

**HTML**

Me creare un **registro.html**, que contendrá el formulario de registro. Antes de crear el form, crearemos un div con grid\_8 que será donde va a estar contenido. Una vez hecho esto creamos la etiqueta **<form>** con **id=registro**, **name=registro**, y con el evento **onsubmit**, con “return comprobarDatos()” para que al enviarse se ejecute la función **comprobarDatos()**.

El formulario contendrá una cabecera, un separador y los inputs, a través de **CSS** daré estilos a cada elemento, para que tenga una apariencia mas atractiva para el usuario.

Cada input estará contenido dentro de las etiquetas **<label>,** con un **tipo de datos**, un **id**, un **name**, **placeholder** que será lo que aparezca dentro del input antes de clickear dentro de él, **pattern** para validar las expresiones regulares y un title para cuando falle al escribirlo le muestre el porqué.

**CSS**

A través de CSS3 se validará en el momento de escribir, añadiendo esto en nuestro css creado, registro.css:

/\*Si el valor que el usuario escribe es válido, obtendrá un color verde\*/

input:valid{

border:2px solid green;

}

/\*caso contrario, el color será rojo\*/

input:invalid{

border:2px solid red;

}

**JAVASCRIPT**

Para comprobar que los datos son válidos, crearemos un script JS, que contendrá una función, comprobarDatos(), no hay que olvidarse de importar el script a registro.html.

Nuestra función ira extrayendo los valores del formulario mediante esta sentencia:

var nombreId = document.registro.nombre.value;

Y mediante expresiones regulares, comprobara todos los campos obligatorios si todos los valores son true, mostrara un mensaje indicando “Los datos se han registrado correctamente.” y creara las cookies con email y contraseña. Si por el contrario sale false, mostrara que las contraseñas son incorrectas, ya que los demás campos se comprueban a medida que el usuario va escribiendo, por el contrario, las contraseñas mediante un IF comprobamos si son iguales y si no lo son devuelve false, con el respectivo ALERT.

A parte de estas validaciones en el nombre y apellidos, al escribirlo en mayúsculas, la función minsuculas(), llamada en cada campo del registro nombre y apellidos, mediante **onkeyup="minusculas()"**. Nos va a ir convirtiendo a medida que se escribe a minúsculas los nombres y apellidos.

* **CAMPOS OPCIONALES**
* Sexo (Opcional)
* Fecha de nacimiento (Opcional)
* Dirección (Opcional)
* País (Opcional, implementado como un selector)
* Número de tarjeta de crédito (Opcional, aparecerá únicamente cuando se indique una dirección completa: campos dirección y país)
* Notificación de novedades (Casilla de verificación)
* Recepción de la revista digital (Casilla de verificación)

**HTML**

Al igual que los campos obligatorios estarán entre etiquetas **<label>.** Y aparecen en color verde, ya que no es obligatorio escribirlos.

El campo sexo es un **select**, con dos **option** masculino o femenino. El campo fecha, es un input, con **type date**, para que se muestre en formato fecha.

En el campo dirección al escribir aparecerá más abajo la opción de tarjeta de crédito, con el formato que tienen todas las tarjetas de crédito **XXXX-XXXX-XXXX-XXXX**. Ya que es un campo oculto y aparecerá solo al escribir la dirección, si se borra la dirección se quitará el campo tarjeta de crédito. Además, el campo tarjeta de crédito aparece en rojo, ya que lo he puesto como campo obligatorio ya que no tiene sentido que te pida la tarjeta de crédito, si no es para rellenarla obligatoriamente en el formato solicitado.

El campo países, es un **select** formado por todos los países del mundo. Luego explicare como he hecho esto mediante **JavaScript**.

Los campos de notificación de novedades y recepción de la revista digital aparecen como casilla de verificación.

**JAVASCRIPT**

Para *mostrar la tarjeta de crédito*, lo haremos a través de una función llamada **mostrar(),** en la que saco el valor de la dirección y si es distinta de vacía, es decir, si han escrito algo, muéstramelo y pónmelo obligatorio el campo. Si no ocúltamelo.

Llamaremos a la función en el campo dirección, de esta manera, **onkeyup="mostrar()”.**

Para *generar todos los países*, los guardare todos en un array llamado países, recorreré el array con los países, pondré por defecto que muestre “Spain”, a través de un IF. Y mostrare los países a través de **idSelect.appendChild(variable creada).**

Llamo a la función arriba del Script *validar.js* para que cuando se ejecute el Script me haga la función.

* **HASH CONTRASEÑA**

Para **hashear** la contraseña lo llevo a cabo a través de una librería, copio en un Js el script que me da la librería, lo llamo **encript.js** y lo importo en mis dos html.

En la función **validar.js:**

Para llamarlo y que me la encripte, escribo **sha251(nombreVariablePassword)** y guardo esto en una variable que se llama hashContrasenia y la meto en la creación de mi cookie.

Para que quede mas claro lo que he escrito en mi archivo Js es:

var hasContrasenia = sha256(password); //hashear contraseña

comprobarCookie("contrasenia", hasContrasenia, 1);

Y en mi **validarLogin.js**:

var contraseniaUser = sha256(document.signup.contrasenia.value);

# FORMULARIO DE LOG IN

El formulario de log In está compuesto por dos campos: **email y contraseña**. Los cuales se validan para que el usuario no meta un formato incorrecto. Además de que es obligatorio aceptar los términos de uso. Si el usuario previamente se ha registrado sus datos de email y contraseña, están guardados en una cookie. Al hacer log In se comprobará si el usuario es correcto, es decir, si está registrado. Y se creara una cookie de sesión para que, si lo está, deje la sesión abierta, **cambie de log in** a **log out**.

**HTML**

Tiene la misma estructura que registro.html, la diferencia es que tiene menos campos, ambos son con formato input, requeridos. El campo de términos será una casilla de verificación, también requerida.

Y al final tiene un input con **type=”submit”.** Para enviar los datos introducidos. Ya que el <form>, tiene un **onsubmit** llamando a la función comprobarDatos(). Pero el script no es el mismo que el de registro.html, se llama v**alidarLogin.js.**

**JAVASCRIPT**

En la función comprobarDatos(), sacamos los valores introducidos en los campos del LogIn y comprobamos si coinciden con el valor de las cookies. Si coinciden, nos aparece un mensaje de “Hola ” con el email del usuario, para interactuar con el usuario. Y creamos una cookie de sesión.

Y comprobamos con un IF si existe la cookie de sesión, si existe cambiamos log In por log out, mediante innerHTML.

# COOKIES

Las cookies las voy a usar para almacenar el email y la contraseña.

* **CREACION DE COOKIES**

La creación de cookies se va a dividir en tres funciones, obtenerCookie, crearCookie y comprobarCookie.

Si todos los datos son correctos del registro se llamara a la función comprobarCookie**("nombreusuario", correoId, 1),** de esta manera se creara la cookie de email, el 1 significa que expira en 1 día. De la misma forma creamos otra cookie que será la contraseña, pero cambiando la clave y el valor por: comprobarCookie**("contrasenia", password, 1);**

En la función antes de crear la cookie en la variable resultado, se comprueba si existe la cookie llamando a la función obtenerCookie, que nos separa la cookie y nos extrae el valor que hay después del nombre.

Si la cookie no existe, se crea la cookie, en la función crearCookie, con los días de expiración que hayamos puesto.

**document.cookie = clave + "=" + valor + "; " + expires;**

De igual manera en el Lpg In se comprueba si existe esa cookie creada, a través de la función obtenerCookie(). Si existe se crea:

**document.cookie="sesion=creada";**

* **MENSAJE COOKIES**

Al abrir la página saldrá un mensaje con la política de cookies, que al pulsar en OK se quitará el mensaje. Este mensaje es obligatorio ponerlo ya que si es una multa mínima de 30.000€.

**HTML**

El mensaje estará dentro de:

**<div id="barraaceptacion" style="display: block;">**

Con el mensaje dentro correspondiente que queramos poner. Y un enlace con OK y con más información.

Para llamar a la función que hace que se quite el mensaje cuando pulsamos OK es:

**<a href="javascript:void(0);" class="ok" onclick="ok();"><b>OK</b></a>**

**CSS**

El mensaje tiene estilos CSS3 que van a hacer que tenga esa apariencia superpuesta al texto, con un fondo semitransparente:

display: none;

position: fixed;

background-color: rgba(0, 0, 0, 0.5);

color: #fff;

z-index: 99999;

**JAVASCRIPT**

Dentro de la función ok() tenemos:

**document.getElementById("barraaceptacion").style.display="none";**

Con el estilo *display: none* desaparece el mensaje al pulsar en Ok.

* **BORRAR COOKIE DE SESION**

Para comprobar si el usuario esta logueado se crea una cookie de sesión, y nos muestra **log out**, cuando el usuario se ha autenticado correctamente, quien previamente se debe a ver registrado.

Si el usuario quiere cerrar la sesión creada, debe pulsar en **log out**, y este cambiara a **log in**, borrando la cookie de sesión. Por lo tanto, si vuelve a loguearse, mostrara otra vez log out y asi sucesivamente. Para llevar esto a cabo he escrito la función cerrarSesion en mi **validarLogin.js**:

function cerrarSesion(){

if(obtenerCookie("sesion") != ""){

document.cookie = 'sesion' + '=; expires=Thu, 01 Jan 1970 00:00:01 GMT;';

}

}

Esta función borra la cookie sesión si existe.