

Entorno Cliente

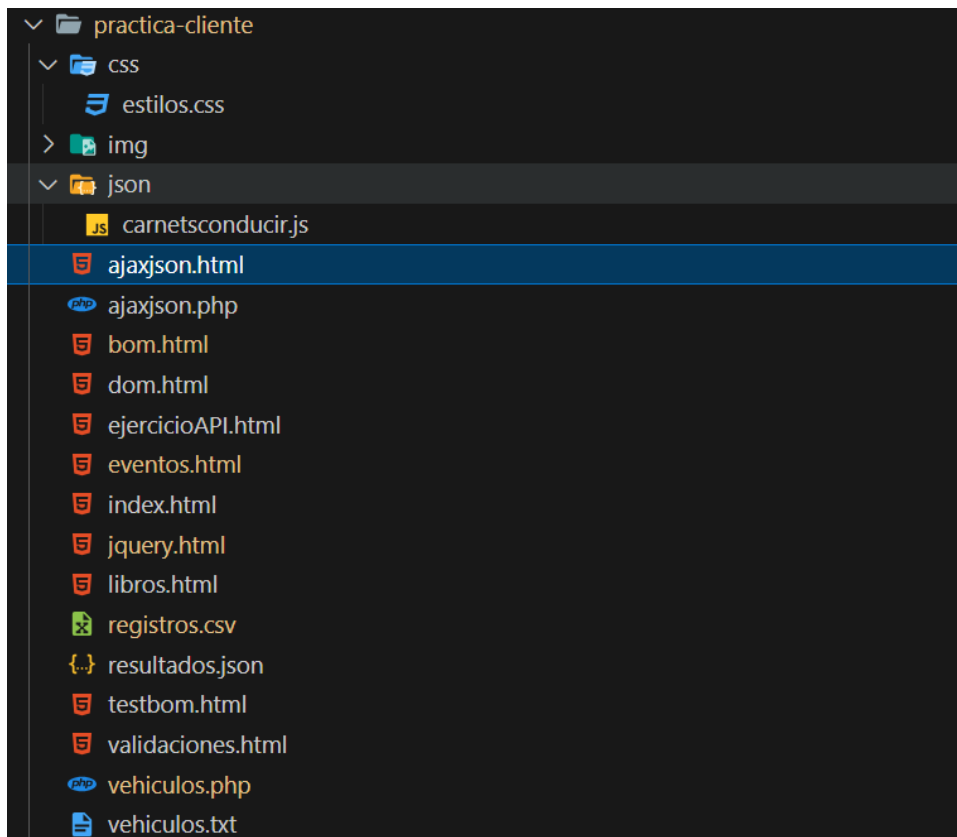
Ejercicios Javascript

David Herrero Estévez

Tabla de contenidos

Descripción del ejercicio	3
Manipulación del DOM	4
Manipulación del BOM	7
Validaciones	10
Eventos	11
Ajax	14
JQuery	17
Consumo de API	19
Repositorio	22

Ficheros



Descripción del ejercicio

El ejercicio consiste en demostrar el uso de diferentes métodos de javascript para la manipulación del DOM y BOM.

Se deben realizar las siguientes manipulaciones:

- Manipulación del DOM
 - Buscar elementos
 - Movernos entre los elementos del DOM , acceder a padre, hijo
 - Crear y borrar elementos
- Manipulación del BOM
 - Window
 - Control del tiempo
- Validación
 - ReportValidity o check validity
- Eventos
 - AddEventListener
- Ajax
- jquery
- Consumición de API

Yo he decidido utilizar una página por ejercicio, aunque hay muchos ejercicios que están utilizados en varias páginas.

He creado una página *index* desde donde se puede acceder a cada uno de los ejercicios.

Ejercicios Entorno Cliente

Topic
Manipulación del DOM
Manipulación del BOM
Validación (formulario)
Eventos
Ajax
Jquery
Consumo API (con fetch)

Manipulación del DOM

localhost/practica-cliente/dom.html

1- Manipulación del DOM

- Buscar elementos
- movernos entre los elementos del DOM Acceder a padre, hijo...
- crear y borrar elementos

Número	Nombre	Apellido	Edad	Modificar	Eliminar
1	DAVID	HERRERO	47		✖
2	NATALIA	GARCÍA	41		✖
3	ADRIANA	HERRERO	9		✖
4	SOFÍA	HERRERO	8		✖

Nombre Apellido Edad

Lista de elementos (para hacer referencia al elemento parent)

- disc
- circle
- square
- decimal
- decimal-leading-zero
- lower-roman
- upper-roman
- lower-alpha
- upper-alpha
- none

[Volver](#)

Esta página consta de 2 ejercicios.

El primero cuenta con un formulario y el segundo hace uso de parent / child.

Inicialmente tenemos una tabla HTML con la cabecera de los elementos que se añadirán posteriormente.

```
<table id="tabla">
  <thead>
    <tr>
      <th>Número</th>
      <th>Nombre <span id="ordenanombre" style="font-size: 0.8em;">&#x1F53C;</span></th>
      <th>Apellido<span id="ordenaapellidos" style="font-size: 0.8em;">&#x1F53C;</span></th>
      <th>Edad</th>
      <th>Modificar <img alt="pencil icon" /></th>
      <th>Eliminar <img alt="X icon" /></th>
    </tr>
  </thead>
  <tbody>
  </tbody>
</table>
```

A continuación se describe el funcionamiento de la página

Haciendo uso de eventos, añadimos al botón el evento submit (onsubmit)

```
form.addEventListener('submit', function(event){
    event.preventDefault();
    if (!form.reportValidity()) {
        return;
    }
    const nombre = document.getElementById('nombre').value.toUpperCase();
    const apellido = document.getElementById('apellido').value.toUpperCase();
    const edad = document.getElementById('edad').value;
    if (nombre === '' || apellido === '' || edad === '') {
        alert('Todos los campos son obligatorios');
        return;
    }
});
```

Usando preventdefault evitamos que el formulario se envíe de forma standard.

Mediante !form.reportValidity nos aseguramos de que si hay algún error no enviemos el formulario.

Mediante el uso de constantes (no vamos a cambiar el contenido) capturamos los campos del formulario, y los hacemos mayúsculas (toUpperCase())

```
let botonsubmit=document.querySelector('button'); //accedemos al botón mediante la propiedad query selector

if (botonsubmit.textContent === 'Editar') { //editar
    let tr = tbody.querySelector('tr:nth-child('+filaeditada+1)'); //accedemos a la fila que queremos editar
    tr.querySelector('td:nth-child(2)').textContent = nombre; //el nombre está en la segunda columna
    tr.querySelector('td:nth-child(3)').textContent = apellido; //el apellido está en la tercera columna
    tr.querySelector('td:nth-child(4)').textContent = edad; //la edad está en la cuarta columna
    botonsubmit.textContent = 'Agregar'; //modificamos el texto del botón a "Agregar"
    botonsubmit.style.backgroundColor = 'grey'; //modificamos el color de fondo del botón a gris
    botonsubmit.style.color = 'white'; //modificamos el color del texto del botón a blanco
    document.getElementById('nombre').value = ''; //limpiamos el campo nombre
    document.getElementById('apellido').value = ''; //limpiamos el campo apellido
    document.getElementById('edad').value = ''; //limpiamos el campo edad
    filaeditada=-1; //reseteamos la variable filaeditada
    return;
}else{ //en caso contrario, añadimos una nueva fila
```

Hemos capturado el evento submit, pero vamos a realizar dos acciones diferentes dentro del mismo evento.

Primero preguntamos si el valor del botón es editar.

En caso afirmativo modificamos el valor de fila elegida (filaeditada+1) de la tabla con el contenido del formulario.

Posteriormente modificamos el texto del botón así como el color de fondo y color.

Finalmente limpiamos el valor del formulario.

```
}else{ //en caso contrario, añadimos una nueva fila

    const tr = document.createElement('tr'); //creamos un elemento tr
    let numeroTablas=document.querySelector('tbody').rows?document.querySelector('tbody').rows.length:0; //obtenemos el número de filas de la tabla
    console.log(document.querySelector('tbody').rows);
    console.info(numeroTablas);
    numeroTablas++; //incrementamos el número de filas para que empiece en 1 y no en 0
    tr.innerHTML = `
        <td>${numeroTablas}</td>
        <td>${nombre}</td>
        <td>${apellido}</td>
        <td>${edad}</td>
        <td>👤</td>
        <td>✖</td>
    `; //añadimos el contenido de las celdas
    tbody.appendChild(tr); //añadimos el tr a la tabla

    let td = tr.querySelector('td:last-child');
    td.addEventListener('click', function(){ //eliminar
        const tr = td.parentElement;

        tr.remove();
    });
};
```

en caso contrario (no es editar), agregamos una nueva fila a la tabla.

Creamos dinámicamente una fila de la tabla con `createElement`, le añadimos contenido mediante el uso del “`” y los valores de las variables `${var1}`, y finalmente añadimos la nueva fila creada al cuerpo de la tabla (`tbody.appendChild`).

Una vez añadida la fila, mediante `addEventListener` le añadimos el comportamiento para la ✕.

Irá al objeto padre y lo eliminará.

```
let tdnumero = tr.querySelector('td:first-child');
tdnumero.addEventListener('mouseover', function(){ //
    const tr = tdnumero.parentElement;

    tr.style.backgroundColor = 'red';
    tr.style.color = 'white';
});
tdnumero.addEventListener('mouseout', function(){ //
    const tr = tdnumero.parentElement;
    tr.style.backgroundColor = 'white';
    tr.style.color = 'black';
});
```

Para dar más funcionalidad a la tabla, añadimos un evento `mouseover` y `mouseout`, donde vamos a colorear la fila actual.

Finalmente vamos a dotar de funcionalidad al botón de editar la fila

```
let tdEdit = tr.querySelector('td:nth-child(5)'); //editar
tdEdit.addEventListener('click', function(){
    const tr = tdEdit.parentElement;
    const tds = tr.querySelectorAll('td');
    document.getElementById('nombre').value = tds[1].textContent;
    document.getElementById('apellido').value = tds[2].textContent;
    document.getElementById('edad').value = tds[3].textContent;
    filaeditada=tr.rowIndex-1;
    console.log(filaeditada);
    let botonsubmit=document.querySelector('button');
    botonsubmit.textContent = 'Editar';
    botonsubmit.style.backgroundColor = 'green';
    botonsubmit.style.color = 'white';
});
```

Si pulsamos sobre el botón editar, se cargarán los datos de la fila en el formulario y se cambiará el texto y color del botón submit del mismo.

También vamos a darle valor a la fila (`filaeditada`), para poder modificar la tabla posteriormente.

La tabla cuenta junto al nombre y apellido de unas flechitas que sirven para ordenar ascendente y descendientemente la tabla

```
let ordenanombre = document.getElementById('ordenanombre');
let ordenaapellidos = document.getElementById('ordenaapellidos');
let sentido = 1;
ordenanombre.addEventListener('click', function(){
    let filas = Array.from(tbody.querySelectorAll('tr'));
    if (sentido==1){
        filas.sort((a, b) => {
            if (a.querySelector('td:nth-child(2)').textContent > b.querySelector('td:nth-child(2)').textContent) {
                return 1;
            }
            if (a.querySelector('td:nth-child(2)').textContent < b.querySelector('td:nth-child(2)').textContent) {
                return -1;
            }
            return 0;
        });
        sentido=1;
    }
});
```

Mediante la función `sort(a,b)` ordenamos la tabla en función de que el sentido sea ascendente o descendente.

La página también cuenta con una pequeña lista para hacer uso del DOM

```
var elementosLista = [
  'disc',
  'circle',
  'square',
  'decimal',
  'decimal-leading-zero',
  'lower-roman',
  'upper-roman',
  'lower-alpha',
  'upper-alpha',
  'none'
];

const lista = document.getElementById('lista');
lista.style
for (let elemento of elementosLista) {
  const li = document.createElement('li');
  li.textContent = elemento;
  lista.appendChild(li);
}

let elementosDeLaLista = document.querySelectorAll('li');
for (let elemento of elementosDeLaLista) {
  elemento.addEventListener('click', function(){
    console.log('click en ' + elemento.textContent);
    elemento.parentElement.style.listStyleType=elemento.textContent;
  });
}
const form = document.querySelector('form');
```

Creo un array con los diferentes estilos que puede tener una UL

Recorro los elementos del array, creo un elemento LI y lo añado a la lista.

Cuando se hace click en cualquiera de los elementos de la lista, cambio el elemento padre (UL) utilizando el texto del elemento seleccionado.

○ disc	■ circle	1. disc
○ circle	■ square	2. circle
○ square	■ decimal	3. square
		4. decimal

Manipulación del BOM

Manipulación del BOM

Temporizador ☐

He creado otra página para la manipulación del BOM (browser object model).

Dispone de una serie de botones y un checkbox.

Los botones hacen lo que indica el botón.

En el caso de haber seleccionado el temporizador, la acción se llevará a cabo en 3 segundos (o en ½ segundo en el caso de mover ventana).

```
<body>
  <h2> Manipulación del BOM</h2>
  <br>
  <div id="botones">
    <button id="abrir" onclick="abrirVentana()">Abrir ventana</button>
    <button id="cerrar" onclick="cerrarVentana()">Cerrar ventana</button>
    <button id="moverVentana" onclick="moverVentana()">Mover ventana</button>
    <button id="cambiarUrl" onclick="cambiarUrl()">Cambiar URL Ventana</button>
    <label for="temporizador">Temporizador</label> <input type="checkbox" id="temporizador" >
  </div>
  <footer>
    <a href="#" onclick="window.href=window.history.back();" >Volver</a>
  </footer>
</body>
</html>
```

Al hacer click en abrir ventana, haciendo uso del BOM abrimos una ventana y asignamos el controlador a la variable ventana.

En función de si el checkbox temporizador está seleccionado o no, la ventana se abre inmediatamente o tras 3 segundos, usando setTimeout.

```
var temporizador=false;
document.getElementById('temporizador').addEventListener('change',function(){
  if(this.checked){
    temporizador = true;
  }else{
    temporizador = false;
    console.log('El temporizado está '+temporizador);
  }
});
/*
@description: Abrir una ventana. Si el checkbox temporizador está seleccionado, lo hará con un delay de 3 segundos*/
function abrirVentana(){
  console.log('abrir ventana');
  cambiarDisponibilidadExcepto('abrir',false);
  if (temporizador){
    window.setTimeout(function(){
      ventana = window.open("", "_blank", "width=400,height=400");
    },3000);
  }else{
    ventana = window.open("", "_blank", "width=400,height=400");
  }
  //cuando la ventana se cierre, habilita todos los botones
  ventana.addEventListener('close',function(){
    cambiarDisponibilidadExcepto('abrir',true);
  });
}
```


Cerrar ventana cierra la ventana.

Lo he metido dentro de un try catch, porque si la ventana abierta se cambia la URL a una predefinida en la función abrirUrl, da una excepción de seguridad.

```
/*
@description: Cierra la ventana inmediatamente o en 3 segundos, si el checktemporizador está seleccionado */
function cerrarVentana(){
    try{
        console.log('cerrar ventana' + ventana);
        if (temporizador){
            window.setTimeout(function(){
                ventana.close();
            },3000);
        } else{
            ventana.close();
        }
        cambiarDisponibilidadExcepto('abrir',true);
    } catch(e){
        alert('Ha ocurrido un error al cerrar la ventana' + e);
    }
}
```

Mover la ventana mueve la ventana 100 pixels a la derecha y 100 pixels abajo.

```
/*
@description: Mueve la ventana.
Lo hará inmediatamente o con un pequeño delay si el temporizador está seleccionado o no*/
function moverVentana(){
    console.log('mover ventana');
    if (temporizador){
        window.setTimeout(function(){
            ventana.moveBy(100,100);
        },500);
    }else{
        ventana.moveBy(100,100);
    }
}
```

Por último, el botón cambiarURL cambia la url de la ventana a una dirección aleatoria de las 5 que almacena el array listadoURL

```
/*usando BOM cambia la url de la ventana abierta*/
function cambiarUrl(){
    console.log('cambiar url');
    const listadoUrls = [
        'https://www.google.com',
        'https://www.youtube.com',
        'https://www.facebook.com',
        'https://www.twitter.com',
        'https://www.instagram.com'
    ];
    let random = Math.floor(Math.random()*listadoUrls.length);
    console.log(listadoUrls[random]);
    if (temporizador){
        window.setTimeout(function(){
            ventana.location.href=listadoUrls[random];
        },3000);}else{
        ventana.location.href=listadoUrls[random];
    }
}
```

Validaciones

Formulario de Validación

Nombre:	<input type="text"/>
Apellidos:	<input type="text"/>
Email:	<input type="text"/>
Edad:	<input type="text"/>
Contraseña:	<input type="password"/>
Confirmar Contraseña:	<input type="password"/>
URL:	<input type="text"/>
Teléfono:	<input type="text"/>
<input type="button" value="Enviar"/>	

Nombre:	<input type="text" value="d"/>
Apellidos:	<input type="text"/>
Email:	<input type="text"/>
Email:	<input type="text" value="dd@"/>
Edad:	<input type="text" value="1"/>
Contraseña:	<input type="password"/>
Contraseña:	<input type="password"/>
Teléfono:	<input type="text" value="333"/>

Enviar

! Prolonga este texto a 3 caracteres o más (en este momento tiene 1 carácter).

! Rellene este campo.

! Escribe una parte después de '@'. 'dd@' está incompleto.

! El valor debe ser mayor o igual que 18.

! Prolonga este texto a 6 caracteres o más (en este momento tiene 1 carácter).


! Busque la coincidencia con el formato solicitado.
El teléfono debe tener el formato XXX-XXX-XXX

Todas las validaciones anteriores se consiguen mediante el método checkValidity y reportValidity (el primero evalúa si existe un error o no, y el último muestra el mensaje de error).

```
<script>
    document.getElementById('myForm').addEventListener('submit', function(event) {
        var form = event.target;
        if (!form.checkValidity()) {
            event.preventDefault();
            form.reportValidity();
        }
        if (form.password.value != form.confirm_password.value) {
            event.preventDefault();
            if (form.password.value != form.confirm_password.value) {
                event.preventDefault();
                alert('Las contraseñas no coinciden');
            }
        }
    });
</script>
```

Eventos

Eventos en JavaScript

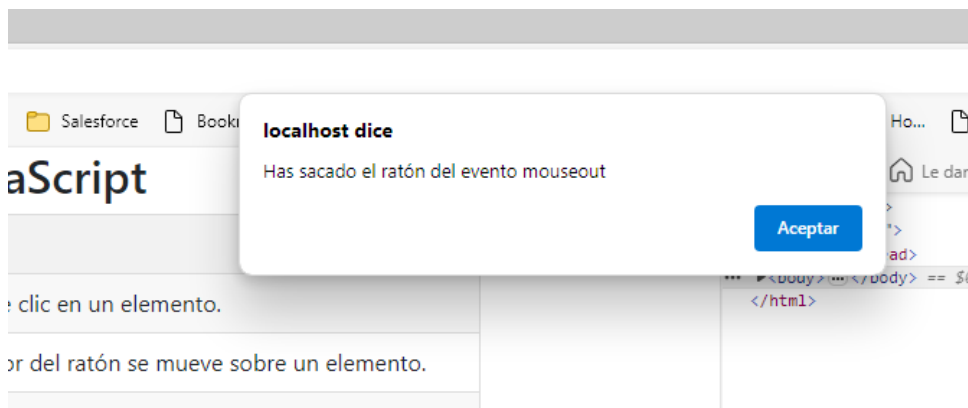
Evento	Descripción
click	Se dispara cuando se hace clic en un elemento.
mouseover	Se dispara cuando el cursor del ratón se mueve sobre un elemento.
mouseout	Se dispara cuando el cursor del ratón se mueve fuera de un elemento.
keydown <input type="text"/>	Se dispara cuando una tecla del teclado está siendo presionada.
keyup <input type="text"/>	Se dispara cuando una tecla del teclado es liberada.
change <input type="text"/>	Se dispara cuando el valor de un elemento cambia.
submit <input type="submit" value="submit"/>	Se dispara cuando se envía un formulario.
focus 	Se dispara cuando un elemento recibe el foco.
blur <input type="text"/>	Se dispara cuando un elemento pierde el foco.
scrollscroll scroll scroll scroll	Se dispara cuando se desplaza un elemento (por ejemplo, una ventana o un elemento con scroll).

Hasta ahora las páginas que hemos visto hacían uso de varios eventos mediante addEventListener.

Esta página demuestra el uso de todos los eventos disponibles:

Click , mouseover, mouseout, keydown, keyup, change, submit, focus, blur y scroll.

Algunos eventos son verificados mediante el uso de un alert



Y otros eventos, para prevenir que el formulario se bloquee, se hace uso de la consola.

```
Has hecho blur en el campo de texto blur
```

Se ha dispuesto una tabla html con el listado de los eventos disponibles

```
<table>
  <tr>
    <th>Evento</th>
    <th>Descripción</th>
  </tr>
  <tr>
    <td>click</td>
    <td>Se dispara cuando se hace clic en un elemento.</td>
  </tr>
  <tr>
    <td>mouseover</td>
    <td>Se dispara cuando el cursor del ratón se mueve sobre un elemento.</td>
  </tr>
  <tr>
    <td>mouseout</td>
    <td>Se dispara cuando el cursor del ratón se mueve fuera de un elemento.</td>
  </tr>
  <tr>
    <td>keydown <input type="text" id="keydowntext"></td>
    <td>Se dispara cuando una tecla del teclado está siendo presionada.</td>
  </tr>
  <tr>
    <td>keyup <input type="text" id="keyupintext"></td>
    <td>Se dispara cuando una tecla del teclado es liberada.</td>
  </tr>
</table>
```

```

<tr>
  <td>change <select id="eventoChange">
    <option value="1">Opción 1</option>
    <option value="2">Opción 2</option>
    <option value="3">Opción 3</option>
    <option value="4">Opción 4</option>
    <option value="--" selected</option>
  </select></td>
  <td>Se dispara cuando el valor de un elemento cambia.</td>
</tr>
<tr>
  <td><form><button type="submit">submit</button></form></td>
  <td>Se dispara cuando se envía un formulario.</td>
</tr>
<tr>
  <td>focus <button id="botonFocus" class="boton-con-imagen"></button>
  </td>
  <td>Se dispara cuando un elemento recibe el foco.</td>
</tr>
<tr>
  <td>blur <input type="text" id="botonblur" ></input></td>
  <td>Se dispara cuando un elemento pierde el foco.</td>
</tr>
<tr>
  <td><textarea id="eventoscroll">scroll</textarea></td>
  <td>Se dispara cuando se desplaza un elemento (por ejemplo, una ventana o un elemento con scroll).</td>
</tr>

```

Los eventos que se ejecutan directamente sobre la celda de la tabla, se han añadido mediante addEventListener haciendo uso del texto de la celda y un switch

```

eventos.forEach(evento => { //recorremos cada uno de los elementos
  console.log(evento.textContent); //mostramos el texto de cada uno de los elementos
  switch(evento.textContent){
    case "click": //si el evento es click
      evento.addEventListener('click', function() {
        alert("Has hecho clic en el evento click");
      });
      break;
    case "mouseover": //si el evento del texto es mouseover
      evento.addEventListener('mouseover', function() {
        alert("Has pasado el ratón por encima del evento mouseover");
      });
      break;
    case "mouseout": //si el evento es mouseout
      evento.addEventListener('mouseout', function() {
        alert("Has sacado el ratón del evento mouseout");
      });
      break;
    case "keydown":
      evento.addEventListener('keydown', function() {
        alert("Has pulsado una tecla en el evento keydown");
      });
      break;
    case "focus":
      break;
  }
}

```

Otros eventos se controlan haciendo uso de getElementById

```

//eventos de teclado
const inputtext = document.getElementById('keyupintext');
inputtext.addEventListener('keyup', function() {
    alert("Has pulsado una tecla en el evento keyup");
});
//eventos de teclado : keydown. Si pulsamos una tecla, se mostrará en consola
document.getElementById('keydownintext').addEventListener('keydown', function(e) {
    console.log("Has pulsado una tecla en el evento keydown (" + e.key + ")");
});
document.getElementById('eventoChange').addEventListener('change', (e) => { //evento change
    alert("Has cambiado el valor del evento change a: " + e.target.value);
});
//evento submit
document.querySelector('form').addEventListener('submit', function(e) {
    e.preventDefault();
    alert("Has enviado el formulario");
});
//evento focus
document.getElementById('botonFocus').addEventListener('focus', function(e) {
    alert("Has hecho foco en el botón focus");
    //para evitar un bucle, quitamos el foco
    e.target.blur();
});
document.getElementById('botonblur').addEventListener('blur', (e)=>{
    console.log("%cHas hecho blur en el campo de texto blur", "color: green");
});
for (let i = 0; i < 10; i++) {
    document.getElementById('eventoscroll').textContent += 'scroll\n';
}
document.getElementById('eventoscroll').addEventListener('scroll', function() {
    console.log("%cHas hecho scroll en el evento scroll", "color: orange");
});

```

Ajax


Uso de Ajax

Dar de alta registro de usuario

Nombre:

Apellidos:

Email:

Fecha de Nacimiento: 

Buscar registros de usuario

Nombre:

Apellidos:

Para la demostración del uso de ajax se ha utilizado una página html junto con una página php

La página php debe ser ejecutada en un servidor web que soporte PHP.

La página cuenta con 2 formularios. El primero realiza un alta de usuarios y el segundo buscará los registros introducidos.

```
<!-- formulario para dar de alta usuarios en la "base de datos"-->
<form id="formularioAlta">
  <label for="nombre">Nombre:</label>
  <input type="text" id="nombre" name="nombre" required ><br><br>
  <label for="apellidos">Apellidos:</label>
  <input type="text" id="apellidos" name="apellidos" required><br><br>
  <label for="email">Email:</label>
  <input type="email" id="email" name="email" required><br><br>
  <label for="fechaNacimiento">Fecha de Nacimiento:</label>
  <input type="date" id="fechaNacimiento" name="fechaNacimiento" required><br><br>
  <input type="submit" value="Dar de alta">
</form>
```

El funcionamiento del botón de submit “Dar de Alta” se puede ver en la siguiente función.

```
// Evento para dar de alta un nuevo registro de usuario
document.getElementById('formularioAlta').addEventListener('submit', function(event) {
  event.preventDefault(); // evita que se envíe el formulario
  console.log('Enviando formulario...');
  let formulario = document.querySelector('form#formularioAlta');

  var data = new FormData(formulario); // crea un objeto FormData con los datos del formulario

  enviarSolicitud('POST', 'ajaxjson.php', data, function(response) {
    alert(response); // Muestra la respuesta del servidor
  });
});
```

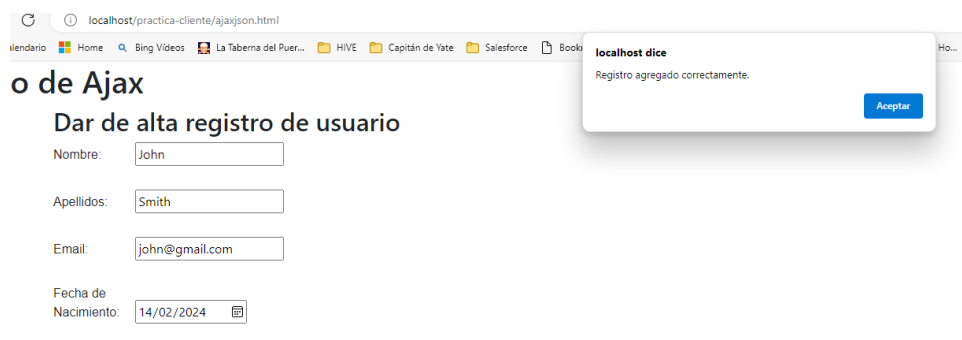
```
// Función para enviar una solicitud AJAX
function enviarSolicitud(method, url, data, callback) {
  var xhr = new XMLHttpRequest();
  xhr.open(method, url, true); // abre la conexión
  // xhr.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
  xhr.onreadystatechange = function() { // función que se ejecuta cuando cambia el estado de la conexión
    if (xhr.readyState === 4 && xhr.status === 200) {
      callback(xhr.responseText); // ejecuta la función callback pasándole la respuesta del servidor
    }
  };
  xhr.send(data); // envía la solicitud
}
```

La función `enviarSolicitud` recibe por parámetro el método a usar (GET o POST), la url de la llamada, los datos y la función de callback.

Creamos un objeto `FormData` con los datos del formulario.

Llamamos a la función `enviarSolicitud`.

En la función callback mostramos la respuesta del servidor.



The screenshot shows a web browser window with the address `localhost/practica-cliente/ajaxjson.html`. The page title is "o de Ajax" and the main heading is "Dar de alta registro de usuario". The form contains the following fields:

- Nombre:
- Apellidos:
- Email:
- Fecha de Nacimiento:

A modal alert box is displayed over the form with the text "Registro agregado correctamente." and an "Aceptar" button.

La página php está fuera de esta práctica, pero como resumen diré que recibe los parámetros POST y los escribe en un fichero .csv

```
if ($SERVER['REQUEST_METHOD'] === 'POST') {
    // Verificar si los datos requeridos se han recibido

    if (isset($_POST['nombre']) && isset($_POST['apellidos']) && isset($_POST['email']) && isset($_POST['fechaNacimiento'])) {
        // Agregar un nuevo registro

        agregarRegistro($_POST['nombre'], $_POST['apellidos'], $_POST['email'], $_POST['fechaNacimiento']);
        echo "Registro agregado correctamente.";
        http_response_code(200);
    } else {
        // Datos incompletos
        http_response_code(400);
        echo "Todos los campos son obligatorios.";
    }
}

// Función para agregar un nuevo registro de usuario al archivo CSV
function agregarRegistro($nombre, $apellidos, $email, $fechaNacimiento) {
    global $archivo;
    $nuevoRegistro = array($nombre, $apellidos, $email, $fechaNacimiento);
    $manejadorArchivo = fopen($archivo, 'a');
    fputcsv($manejadorArchivo, $nuevoRegistro);
    fclose($manejadorArchivo);
}
```

El formulario de búsqueda es el siguiente

Buscar registros de usuario

Nombre:

Apellidos:

Podemos realizar búsquedas por nombre y / o apellidos.

```
// Evento para buscar registros de usuario
document.getElementById('formularioBusqueda').addEventListener('submit', function(e) {
    //borramos el resultado
    document.getElementById('resultadosBusqueda').innerHTML='';
    e.preventDefault();
    var nombre = document.getElementById('busquedaNombre').value;
    var apellidos = document.getElementById('busquedaApellidos').value;
    var parametros = 'nombre=' + encodeURIComponent(nombre) + '&apellidos=' + encodeURIComponent(apellidos);
    enviarSolicitud('GET', 'ajaxjson.php?' + parametros, null, function(response) { // envia una solicitud GET al servidor
        var resultados = '';
        var registros = JSON.parse(response);
        registros.forEach(function(registro) {
            resultados += '<p>Nombre: ' + registro[0] + '</p>'; // crea un párrafo con el nombre del usuario
            resultados += '<p>Apellidos: ' + registro[1] + '</p>';
            resultados += '<p>Email: ' + registro[2] + '</p>';
            resultados += '<p>Fecha de Nacimiento: ' + registro[3] + '</p>';
            resultados += '<hr>';
        });
        document.getElementById('resultadosBusqueda').innerHTML = resultados; // muestra los resultados en el div
    });
});
```


Usamos la misma función que en alta y en la función de callback vamos a pintar los resultados obtenidos .

Los resultados que vienen en el array “registros” están divididos en un array bidimensional.

Siendo la dimensión X un array de resultados y la dimensión Y cada uno de los resultados de la fila.

JQuery

Uso de JQuery

Nombre:	<input type="text"/>
Apellidos:	<input type="text"/>
Email:	<input type="text"/>
Fecha de nacimiento:	<input type="text" value="dd/mm/aaaa"/> 
Carnet de conducir:	<input type="text" value="Clase B"/> ▼
Marca vehículo:	<input type="text" value="Audi"/> ▼ >
Modelo:	<input type="text" value="A1"/> ▼ >
Acabado:	<input type="text"/>
<input type="button" value="Enviar"/>	

Buscar registros de usuario

[Mostrar todos los registros guardados](#)

Para el uso de jquery he creado un formulario simulando el alta de usuarios con datos de vehículos.

El carnet de conducir así como los vehículos (marcas y modelos), se encuentran en un fichero json

```
<script src="./json/carnetsconducir.js"></script>
```

Realizamos la carga de los valores de los select

```

$(document).ready(function(){
    //document ready se ejecuta cuando la página ha cargado completamente
    $.each(carnets, function(index, value){
        $("#carnet").append("<option value='"+value.tipo+"'>"+value.tipo+"</option>");
        //recorremos el array de carnets y añadimos cada uno al select
    });
    $.each(marcas, function(index, value){
        $("#marca").append("<option value='"+value.marca+"'>"+value.marca+"</option>");
        //recorremos el array de marcas, obtenido a través del fichero json y añadimos cada uno al select
    });
    $("#marca").change(function(){
        //cada vez que cambie el valor del select de marca, se ejecutará esta función
        $("#modelo").empty(); //vaciamos el select de modelos
        $.each(marcas, function(index, value){
            if(value.marca==$("#marca").val()){
                $.each(value.modelos, function(index, value){
                    //recorremos el array de modelos de la marca seleccionada y los añadimos al select
                    $("#modelo").append("<option value='"+value+"'>"+value+"</option>");
                });
            }
        });
    });
    $("#marca").val("Audi").change(); //seleccionamos la marca Audi y ejecutamos el evento change

```

También añadimos la funcionalidad de modificar los modelos disponibles en función de la marca seleccionada mediante el evento change en el select con id marca.

Al hacer submit en el formulario, enviamos los datos asíncronos mediante la función de jquery .ajax

```

$("#form").submit(function(event){
    //cuando se envíe el formulario
    event.preventDefault(); //evitamos que se envíe el formulario
    // Obtener los datos del formulario
    var formData = $(this).serialize();
    console.log(formData);
    $.ajax({
        type: "POST",
        url: "vehiculos.php",
        data: formData,
        success: function(response){ //si la petición se ha realizado correctamente
            console.log(response);
        },
        error: function(xhr, status, error){
            console.error(xhr.responseText); // Imprimir mensaje de error en la consola
        }
    });
});

```

Para mostrar los datos guardados, he creado el botón “mostrar datos”

```

$("button").click(function(){
    //cuando se haga clic en el botón
    $.ajax({
        type: "GET",
        url: "vehiculos.php",
        success: function(response){ //si la petición se ha realizado correctamente

            const vehiculos = JSON.parse(response); //parseamos la respuesta a JSON
            $("#resultadosBusqueda").empty(); //vaciamos el div resultadosBusqueda

            for (let vehiculo of vehiculos) {
                //recorremos el array de vehículos y mostramos cada registro en el div resultadosBusqueda
                console.log('vehiculo->' + vehiculo);
                let nombre=vehiculo[0]; //obtenemos el nombre del vehículo

                let apellidos=vehiculo[1];
                let email=vehiculo[2];
                let fecha_de_nacimiento=vehiculo[3];
                let carnet=vehiculo[4];
                let marca=vehiculo[5];
                let modelo=vehiculo[6];
                let acabado=vehiculo[7];
                console.debug("nombre : " + nombre);
                $("#resultadosBusqueda").append("<p>Nombre: " + nombre + "</p>");
                $("#resultadosBusqueda").append("<p>Apellidos: " + apellidos + "</p>");
                $("#resultadosBusqueda").append("<p>Email: " + email + "</p>");
                $("#resultadosBusqueda").append("<p>Fecha de Nacimiento: " + fecha_de_nacimiento + "</p>");
                $("#resultadosBusqueda").append("<p>Carnet de conducir: " + carnet + "</p>");
                $("#resultadosBusqueda").append("<p>Marca: " + marca + "</p>");
                $("#resultadosBusqueda").append("<p>Modelo: " + modelo + "</p>");
                $("#resultadosBusqueda").append("<p>Acabado: " + acabado + "</p>");
                $("#resultadosBusqueda").append("<hr>");
            }
        }
    });
}

```

Haciendo uso de la misma función asíncrona .ajax , en este caso con método GET mostramos en un div todos los datos recibidos.

Al hacer la llamada a la página vehículos, en caso de que se devuelva un código 200 (success) , parseamos el json recibido y lo guardamos en la variable vehículos.

Mediante un for recorreremos el objeto y añadimos al div resultadosBusqueda cada uno de los elementos recibidos.

Consumo de API

Buscador de Libros

Autor




Libro

juan gomez jurado -

Autor

Libro

Buscar

Autor	Libro	Imagen
Juan Gómez-Jurado	Cicatriz	
Juan Gómez-Jurado	El paciente	
Juan Gómez-Jurado	Todo vuelve (Todo arde 2)	

También mediante jquery he consumido una API de google, “google books”

URL : <https://www.googleapis.com/books/v1/volumes>

Esta API no necesita clave ni usuario , por lo que es más sencillo su uso.

```
$(document).ready(function() {
  $('#form').submit(function(event) {
    event.preventDefault();
    var author = $('#author').val();
    var book = $('#book').val();
    const search = `${author} - ${book}`;
    searches.add(search);

    var url = 'https://www.googleapis.com/books/v1/volumes';
    if (author) {
      url += '?q=inauthor:' + author;
    }
    if (book) {
      url += (author ? '+' : '?q=intitle:') + book;
    }

    $.get(url, function(data) {
      var books = data.items;
      var html = '';
      if (books.length > 0) {
        $('#json').html(JSON.stringify(books, null, 2));
        $('#resultados').html('');
        html += '<table class="table"><thead><tr><th>Autor</th><th>Libro</th><th>Imagen</th></tr></thead><tbody>';
        for (var i = 0; i < books.length; i++) {
          var bookInfo = books[i].volumeInfo;
          var author = bookInfo.authors ? bookInfo.authors[0] : 'Desconocido';
          var title = bookInfo.title ? bookInfo.title : 'Desconocido';
          var image = bookInfo.imageLinks ? bookInfo.imageLinks.thumbnail : 'https://via.placeholder.com/128x200';
          var id = books[i].id;
          var description = bookInfo.description ? bookInfo.description : 'Sin descripción';
          var pageCount = bookInfo.pageCount ? bookInfo.pageCount : 'Desconocido';
          var price = bookInfo.saleInfo && bookInfo.saleInfo.listPrice ? bookInfo.saleInfo.listPrice.amount : 'Desconocido';

          var bookData = [title, author, description, pageCount, image, price];
          bookMap.set(id, bookData);
        }
      }
    });
  });
});
```

Al hacer click en el botón, se comprueba primero si el autor y el libro tienen valor, en ese caso se concatena el autor y el libro a la URL de búsqueda

Se puede comprobar el funcionamiento de la API poniendo la url en el navegador y viendo el json devuelto

```

1 {
2   "kind": "books#volumes",
3   "totalItems": 131,
4   "items": [
5     {
6       "kind": "books#volume",
7       "id": "ZrIGAAAQAAJ3",
8       "etag": "281ng6Zrrik",
9       "selfLink": "https://www.googleapis.com/books/v1/volumes/ZrIGAAAQAAJ3",
10      "volumeInfo": {
11        "title": "El ingenioso hidalgo don Quixote de la Mancha",
12        "authors": [
13          "Miguel de Cervantes Saavedra"
14        ],
15        "publishedDate": "1797",
16        "industryIdentifiers": [
17          {
18            "type": "OTHER",
19            "identifier": "OXFORD-911186078"
20          }
21        ],
22        "readingModes": {
23          "text": false,
24          "image": true
25        },
26        "pageCount": 828,
27        "printType": "BOOK",
28        "categories": [
29          "Don Quixote (Fictitious character)"
30        ],
31        "maturityRating": "NOT_MATURE",
32        "allowAnnotating": false,
33        "contentVersion": "0.6.4.0.full.1",
34        "panelizationSummary": {
35          "containsImageBubbles": false,
36          "containsImageBubbles": false
37        },
38        "imageLinks": {
39          "smallThumbnail": "http://books.google.com/books/content?id=ZrIGAAAQAAJ3&printsec=frontcover&img=1&source=5&edgecur1&source=ghs_api",
40          "thumbnail": "http://books.google.com/books/content?id=ZrIGAAAQAAJ3&printsec=frontcover&img=1&source=5&edgecur1&source=ghs_api",
41          "language": "es",
42          "infoLink": "http://books.google.es/books?id=ZrIGAAAQAAJ3&printsec=frontcover&img=1&source=5&edgecur1&source=ghs_api",
43          "infoLink": "https://play.google.com/store/books/details?id=ZrIGAAAQAAJ3&source=ghs_api",
44          "canonicalVolumeLink": "https://play.google.com/store/books/details?id=ZrIGAAAQAAJ3"
45        },
46        "saleInfo": {
47          "country": "ES",
48          "availability": "FREE",
49          "isBook": true,
50          "buyLink": "https://play.google.com/store/books/details?id=ZrIGAAAQAAJ3&source=ghs_api"
51        },
52        "accessInfo": {
53          "country": "ES",
54          "availability": "ALL_PAGES",
55          "embeddable": true,
56          "publicDomain": true,
57          "textToSpeechPermission": "ALLOWED",
58          "isAvailable": false,
59          "downloadLink": "http://books.google.es/books/download/El_ingenioso_hidalgo_don_Quixote_de_la_Mancha_epub?id=ZrIGAAAQAAJ3&output=epub&source=ghs_api"
60        },
61        "pdf": {
62          "isAvailable": true,
63          "downloadLink": "http://books.google.es/books/download/El_ingenioso_hidalgo_don_Quixote_de_la_Mancha_pdf?id=ZrIGAAAQAAJ3&output=pdf&sig=4CfU3U2kq93BMB--9evhIP_mPOT6gk&source=ghs_api"
64        },
65        "accessViewStatus": "FULL_PUBLIC_DOMAIN",
66        "quoterSharingAllowed": false
67      }
68    ]
69  }
70 }
71
72 {
73   "kind": "books#volume",
74   "id": "ZrIGAAAQAAJ3",
75   "etag": "281ng6Zrrik",
76   "selfLink": "https://www.googleapis.com/books/v1/volumes/ZrIGAAAQAAJ3",
77   "volumeInfo": {
78     "title": "El ingenioso hidalgo don Quixote de la Mancha",
79     "authors": [
80       "Miguel de Cervantes Saavedra"
81     ],
82     "publishedDate": "1842",
83     "industryIdentifiers": [
84       {
85         "type": "OTHER",
86         "identifier": "OXFORD-911186078"
87       }
88     ],
89     "readingModes": {
90       "text": false,
91       "image": true
92     },
93     "pageCount": 828,
94     "printType": "BOOK",
95     "categories": [
96       "Don Quixote (Fictitious character)"
97     ],
98     "maturityRating": "NOT_MATURE",
99     "allowAnnotating": false,
100    "contentVersion": "0.6.4.0.full.1",
101    "panelizationSummary": {
102      "containsImageBubbles": false,
103      "containsImageBubbles": false
104    },
105    "imageLinks": {
106      "smallThumbnail": "http://books.google.com/books/content?id=ZrIGAAAQAAJ3&printsec=frontcover&img=1&source=5&edgecur1&source=ghs_api",
107      "thumbnail": "http://books.google.com/books/content?id=ZrIGAAAQAAJ3&printsec=frontcover&img=1&source=5&edgecur1&source=ghs_api",
108      "language": "es",
109      "infoLink": "http://books.google.es/books?id=ZrIGAAAQAAJ3&printsec=frontcover&img=1&source=5&edgecur1&source=ghs_api",
110      "infoLink": "https://play.google.com/store/books/details?id=ZrIGAAAQAAJ3&source=ghs_api",
111      "canonicalVolumeLink": "https://play.google.com/store/books/details?id=ZrIGAAAQAAJ3"
112    },
113    "saleInfo": {
114      "country": "ES",
115      "availability": "FREE",
116      "isBook": true,
117      "buyLink": "https://play.google.com/store/books/details?id=ZrIGAAAQAAJ3&source=ghs_api"
118    },
119    "accessInfo": {
120      "country": "ES",
121      "availability": "ALL_PAGES",
122      "embeddable": true,
123      "publicDomain": true,
124      "textToSpeechPermission": "ALLOWED",
125      "isAvailable": false,
126      "downloadLink": "http://books.google.es/books/download/El_ingenioso_hidalgo_don_Quixote_de_la_Mancha_epub?id=ZrIGAAAQAAJ3&output=epub&source=ghs_api"
127    },
128    "pdf": {
129      "isAvailable": true,
130      "downloadLink": "http://books.google.es/books/download/El_ingenioso_hidalgo_don_Quixote_de_la_Mancha_pdf?id=ZrIGAAAQAAJ3&output=pdf&sig=4CfU3U2kq93BMB--9evhIP_mPOT6gk&source=ghs_api"
131    },
132    "accessViewStatus": "FULL_PUBLIC_DOMAIN",
133    "quoterSharingAllowed": false
134  }
135 }

```

Llamamos a la API mediante jquery , get

```

$.get(url, function(data) {
  var books = data.items;
  var html = '';
  if (books.length > 0) {
    $('resultados').html('');
    html += '<table class="table"><thead><tr><th>Autor</th><th>Libro</th><th>Imagen</th></tr></thead><tbody>';
    for (var i = 0; i < books.length; i++) {
      var bookInfo = books[i].volumeInfo;
      var author = bookInfo.authors ? bookInfo.authors[0] : 'Desconocido';
      var title = bookInfo.title ? bookInfo.title : 'Desconocido';
      var image = bookInfo.imageLinks ? bookInfo.imageLinks.thumbnail : 'https://via.placeholder.com/128x200';
      var id = books[i].id;
      var description = bookInfo.description ? bookInfo.description : 'Sin descripción';
      var pageCount = bookInfo.pageCount ? bookInfo.pageCount : 'Desconocido';
      var price = bookInfo.saleInfo && bookInfo.saleInfo.listPrice ? bookInfo.saleInfo.listPrice.amount : 'Desconocido';

      var bookData = [title, author, description, pageCount, image, price];
      bookMap.set(id, bookData);

      html += '<tr><td>' + author + '</td><td>' + title + '</td><td></td></tr>';

      console.log(JSON.stringify(bookInfo, null, 2));
    }
    html += '</tbody></table>';
  } else {
    html = '<p>No se encontraron resultados</p>';
  }
}

```

Con solo dos parámetros.

El primero es la URL de búsqueda y el segundo la función de callback, en este caso , hacemos uso de una función anónima que recibe los datos en formato json.

Creamos una tabla html y vamos añadiendo filas por cada elemento contenido en el objeto items.

He añadido un evento onclick cuando se pulsa sobre las imágenes, para que muestre la información del libro en una ventana modal.

```
const images = document.querySelectorAll('.masinfo');
images.forEach(image => {
  image.addEventListener('click', function(event) {
    const id = event.target.id;
    muestraInformacionLibro(id);
  });
});
```

Repositorio

Puedes consultar la mayoría de funcionalidades (excepto las que hacen uso de php) en el siguiente enlace:

<https://sfdcdauidesi.github.io/EntornoServidor/practica-cliente/>

Y puedes consultar el código fuente en este otro enlace:

<https://github.com/SFDCdauidesi/EntornoServidor/tree/main/practica-cliente>