Tabla Dinámica

body { font-family: Arial, sans-serif; margin: 20px; text-align: center; } h1 { color: #333; } table { width: 50%; margin: 20px auto; border-collapse: collapse; box-shadow: 0 0 20px rgba(0, 0, 0, 0.15); } th, td { padding: 12px 15px; border: 1px solid #ddd; } th { background-color: #4CAF50; color: white; font-weight: bold; } tr:nth-child(even) { background-color: #f2f2f2; } tr:hover { background-color: #ddd; } document.addEventListener('DOMContentLoaded', () => { const tableContainer = document.getElementById('tableContainer'); // String de ejemplo const dataString = "Nombre,Edad,Ciudad\nJuan,25,Madrid\nAna,30,Barcelona\nLuis,22,Valencia"; // Función para crear la tabla dinámica function createTableFromData(data) { const rows = data.split('\n'); const table = document.createElement('table'); const tbody = document.createElement('tbody'); rows.forEach((row, index) => { const cells = row.split(','); const tr = document.createElement('tr'); cells.forEach(cell => { const element = index === 0 ? document.createElement('th') : document.createElement('td'); element.textContent = cell; tr.appendChild(element); }); tbody.appendChild(tr); }); table.appendChild(tbody); tableContainer.appendChild(table); } // Crear la tabla dinámica createTableFromData(dataString); }); ------- 9 -------

Lista Dinámica

Agregar elemento Agregar Eliminar Último

body { font-family: Arial, sans-serif; margin: 20px; } button { margin: 5px; padding: 10px; font-size: 16px; color: #07ae25; } input { padding: 10px; font-size: 16px; color: blue; } ul { list-style-type: none; padding: 0; } li { background-color: #87abf2; padding: 10px; margin: 5px 0; border: 1px solid #ddd; border-radius: 5px; } document.addEventListener('DOMContentLoaded', () => { const itemInput = document.getElementById('itemInput'); const addItemButton = document.getElementById('addItem'); const removeItemButton = document.getElementById('removeItem'); const itemList = document.getElementById('itemList'); // Función para agregar un elemento a la lista function addItem() { const itemText = itemInput.value.trim(); if (itemText) { const li = document.createElement('li'); li.textContent = itemText; itemList.appendChild(li); itemInput.value = ''; } } // Función para eliminar el último elemento de la lista function removeItem() { const items = itemList.getElementsByTagName('li'); if (items.length > 0) { itemList.removeChild(items[items.length - 1]); } } // Evento para agregar un elemento addItemButton.addEventListener('click', addItem); // Evento para eliminar el último elemento removeItemButton.addEventListener('click', removeItem); // Evento para agregar un elemento al presionar Enter itemInput.addEventListener('keypress', (event) => { if (event.key === 'Enter') { addItem(); } }); }); -----

Formulario de Registro

Nombre:
Email:
Contraseña:
Registrar

title { font-family: Arial, sans-serif; color: #721c24; margin: 20px; } body { font-family: Arial, sans-serif; margin: 20px; } form { max-width: 300px; color: aqua; margin: 0 auto; } label { display: block; margin-top: 10px; color: blueviolet; } input { width: 100%; padding: 10px; margin-top: 5px; font-size: 16px; color: burlywood; } button { margin-top: 20px; padding: 10px; font-size: 16px; width: 100%; } #message { margin-top: 20px; padding: 10px; text-align: center; font-size: 18px; color: brown; } .success { background-color: #d4edda; color: #155724; border: 1px solid #c3e6cb; } .error { background-color: #f8d7da; color: #721c24; border: 1px solid #f5c6cb; } document.addEventListener('DOMContentLoaded', () => { const form = document.getElementById('registrationForm'); const message = document.getElementById('message'); // Función para validar el formulario function validateForm(event) { event.preventDefault(); // Evita que el formulario se envíe automáticamente const name = document.getElementById('name').value.trim(); const email = document.getElementById('email').value.trim(); const password = document.getElementById('password').value.trim(); if (name === '' || email === '' || password === '') { showMessage('error', 'Todos los campos son obligatorios.'); return; } if (!validateEmail(email)) { showMessage('error', 'Por favor, introduce un email válido.'); return; } // Simulación de envío del formulario setTimeout(() => { showMessage('success', 'Registro exitoso. ¡Bienvenido, ' + name + '!'); form.reset(); }, 1000); } // Función para validar el formato de email function validateEmail(email) { const re = /^[^\s@]+@[^\s@]+\.[^\s@]+$/; return re.test(email); } // Función para mostrar mensajes function showMessage(type, text) { message.className = type; message.textContent = text; } // Evento para validar el formulario al enviar form.addEventListener('submit', validateForm); }); ----------

Contador Simple

- 0 +

body { font-family: Arial, sans-serif; margin: 20px; text-align: center; } #counter { margin-top: 20px; } button { padding: 10px 20px; font-size: 16px; margin: 0 10px; } span { font-size: 24px; font-weight: bold; } const countElement = document.getElementById('count'); const incrementButton = document.getElementById('increment'); const decrementButton = document.getElementById('decrement'); let count = 0; // Función para actualizar el contador en la página function updateCount() { countElement.textContent = count; } // Función para incrementar el contador function increment() { count++; updateCount(); } // Función para decrementar el contador function decrement() { if (count > 0) { count--; updateCount(); } } // Evento para incrementar el contador incrementButton.addEventListener('click', increment); // Evento para decrementar el contador decrementButton.addEventListener('click', decrement); }); ----------

Añadir, Borrar y Ordenar Datos

Nuevo dato Añadir Dato Borrar Dato Ordenar Ascendente Ordenar Descendente
body { font-family: Arial, sans-serif; margin: 20px; text-align: center; } button { margin: 10px; padding: 10px 20px; font-size: 16px; cursor: pointer; background-color: #4CAF50; color: white; border: none; border-radius: 5px; } button:hover { background-color: #45a049; } button:disabled { background-color: #cccccc; cursor: not-allowed; } input { padding: 10px; font-size: 16px; margin-right: 10px; } #dataContainer { display: flex; flex-direction: column; /* Cambia la dirección de la disposición a vertical */ align-items: center; margin-top: 20px; } .dataItem { width: 150px; height: 50px; margin: 10px; display: flex; align-items: center; justify-content: center; font-size: 20px; color: white; background-color: #f44336; border-radius: 5px; } document.addEventListener('DOMContentLoaded', () => { const addDataButton = document.getElementById('addData'); const clearDataButton = document.getElementById('clearData'); const sortAscButton = document.getElementById('sortAsc'); const sortDescButton = document.getElementById('sortDesc'); const newDataInput = document.getElementById('newData'); const dataContainer = document.getElementById('dataContainer'); let dataArray = []; // Función para añadir un nuevo dato function addData() { const newData = newDataInput.value.trim(); if (newData) { dataArray.push(newData); const div = document.createElement('div'); div.className = 'dataItem'; div.textContent = `${dataArray.length - 1}: ${newData}`; // Mostrar el número y el dato en el div dataContainer.appendChild(div); newDataInput.value = ''; updateButtons(); } else { alert('Por favor, ingresa un dato válido.'); } } // Función para borrar un dato function clearData() { if (dataContainer.firstChild) { dataContainer.removeChild(dataContainer.lastChild); dataArray.pop(); updateButtons(); } else { alert('No hay datos para borrar.'); } } // Función para ordenar los datos de forma ascendente function sortAsc() { dataArray.sort((a, b) => a.localeCompare(b)); updateDataContainer(); } // Función para ordenar los datos de forma descendente function sortDesc() { dataArray.sort((a, b) => b.localeCompare(a)); updateDataContainer(); } // Función para actualizar el contenedor de datos function updateDataContainer() { dataContainer.innerHTML = ''; dataArray.forEach((data, index) => { const div = document.createElement('div'); div.className = 'dataItem'; div.textContent = `${index}: ${data}`; dataContainer.appendChild(div); }); } // Función para actualizar el estado de los botones de ordenamiento function updateButtons() { if (dataArray.length > 0) { sortAscButton.disabled = false; sortDescButton.disabled = false; } else { sortAscButton.disabled = true; sortDescButton.disabled = true; } } // Evento para añadir un nuevo dato addDataButton.addEventListener('click', addData); // Evento para borrar un dato clearDataButton.addEventListener('click', clearData); // Evento para ordenar de forma ascendente sortAscButton.addEventListener('click', sortAsc); // Evento para ordenar de forma descendente sortDescButton.addEventListener('click', sortDesc); }); ----------

Tabla Dinámica con Filtros

Filtrar por: Todos
body { font-family: Arial, sans-serif; margin: 20px; text-align: center; } h1 { color: #333; } table { width: 80%; margin: 20px auto; border-collapse: collapse; box-shadow: 0 0 20px rgba(0, 0, 0, 0.15); } th, td { padding: 12px 15px; border: 1px solid #ddd; cursor: pointer; } th { background-color: #4CAF50; color: white; font-weight: bold; } tr:nth-child(even) { background-color: #f2f2f2; } tr:hover { background-color: #ddd; } select { margin: 10px; padding: 10px; font-size: 16px; } document.addEventListener('DOMContentLoaded', () => { const tableContainer = document.getElementById('tableContainer'); const filterSelect = document.getElementById('filter'); let todos = []; let sortColumn = null; let sortOrder = 'asc'; // Función para cargar los datos desde la API async function fetchTodos() { try { const response = await fetch('https://jsonplaceholder.typicode.com/todos'); todos = await response.json(); createTable(todos); } catch (error) { console.error('Error fetching todos:', error); } } // Función para crear la tabla dinámica function createTable(data) { tableContainer.innerHTML = ''; const table = document.createElement('table'); const thead = document.createElement('thead'); const tbody = document.createElement('tbody'); // Crear encabezados de la tabla const headers = ['ID', 'Título', 'Completado']; const tr = document.createElement('tr'); headers.forEach(header => { const th = document.createElement('th'); th.textContent = header; th.addEventListener('click', () => sortTable(header.toLowerCase())); tr.appendChild(th); }); thead.appendChild(tr); table.appendChild(thead); // Crear filas de la tabla data.forEach(todo => { const tr = document.createElement('tr'); tr.innerHTML = ` ${todo.id} ${todo.title} ${todo.completed ? 'Sí' : 'No'} `; tbody.appendChild(tr); }); table.appendChild(tbody); tableContainer.appendChild(table); } // Función para filtrar la tabla function filterTable() { const filterValue = filterSelect.value; let filteredData = todos; if (filterValue === 'completed') { filteredData = todos.filter(todo => todo.completed); } else if (filterValue === 'notCompleted') { filteredData = todos.filter(todo => !todo.completed); } createTable(filteredData); } // Función para ordenar la tabla function sortTable(column) { if (sortColumn === column) { sortOrder = sortOrder === 'asc' ? 'desc' : 'asc'; } else { sortColumn = column; sortOrder = 'asc'; } todos.sort((a, b) => { let valueA, valueB; if (column === 'id') { valueA = a.id; valueB = b.id; } else if (column === 'title') { valueA = a.title.toLowerCase(); valueB = b.title.toLowerCase(); } else if (column === 'completed') { valueA = a.completed ? 1 : 0; valueB = b.completed ? 1 : 0; } if (valueA < valueB) { return sortOrder === 'asc' ? -1 : 1; } if (valueA > valueB) { return sortOrder === 'asc' ? 1 : -1; } return 0; }); createTable(todos); } // Cargar los datos y crear la tabla inicial fetchTodos(); // Evento para filtrar la tabla filterSelect.addEventListener('change', filterTable); }); ----------