



Somos  
F5



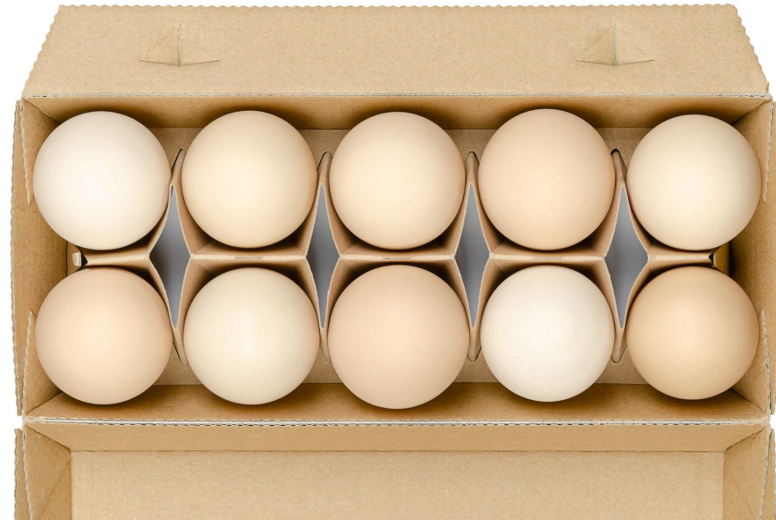
Iberdrola  
España  
Fundación

# Listas en Javascript

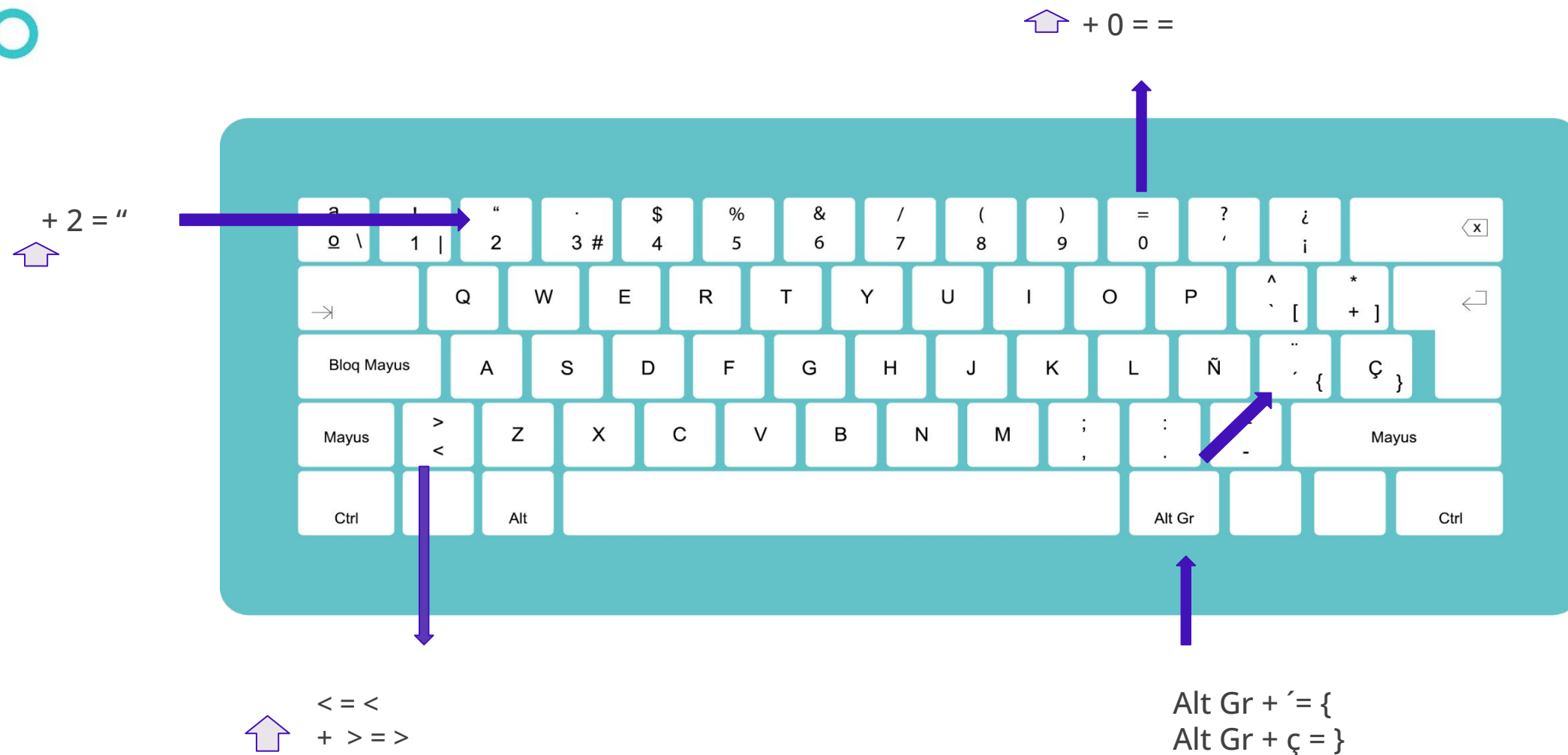
ABRIMOS RUTAS **INCLUSIVAS**  
AL TALENTO DIGITAL

# Objetivos generales y contenidos del taller

- ¿Qué es un Array?
- Para qué usamos listas en JS
- Probar por consola el resultado
- ¡Sumérgete en el código!



# Antes de empezar...



# Conceptos importantes sobre HTML

## Etiquetas

```
<etiqueta>Hola</etiqueta>
```

Para abrir

Para cerrar

```
<h1 class="claseSaludo" id="idSaludo">Hola</h1>
```

Atributos

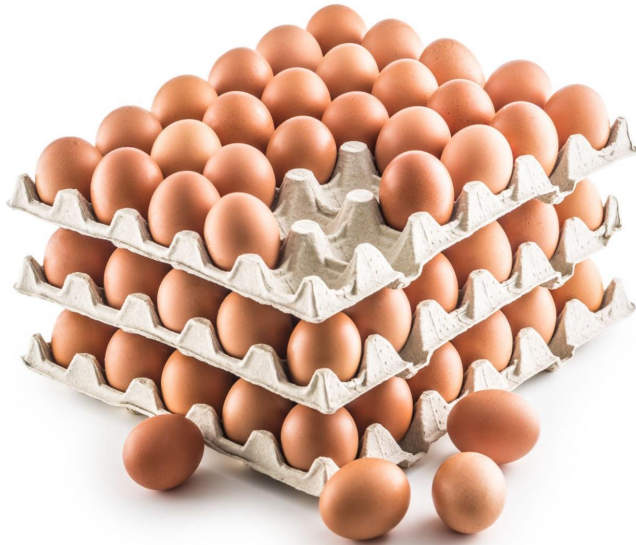
## HTML

```
<body>  
  <h1>Hola Mundo!!</h1>  
</body>
```

Hola Mundo!!



# ¿Qué es un Array?



# Declaración de un Array

para crear un array, usamos corchetes [ ]

```
let colores = ["rojo", "azul", "verde"];  
let numeros = [10, 20, 30, 40, 50];  
let mezclado = [true, "hola", 5]; // ¡Pueden tener diferentes tipos!  
let vacio = []; // Un array vacío
```



# Acceder a los Elementos por su Índice

Explicar que cada "hueco" o posición en el array tiene un número llamado **índice**. ¡Importante! Los índices empiezan desde **cero**.

**Analogía:** En nuestra caja de huevos, el primer hueco es el número 0, el segundo es el número 1, y así sucesivamente.

```
let colores = ["rojo", "azul", "verde"];  
console.log(colores[0]); // Mostrará "rojo" (el elemento en el índice 0)  
console.log(colores[1]); // Mostrará "azul" (el elemento en el índice 1)
```





# La Propiedad length (Longitud del Array):

Explicar que podemos saber cuántos elementos hay en un array usando la propiedad `.length`.

**Analogía:** Es como contar cuántos huevos hay en la caja.

```
let frutas = ["manzana", "banana", "naranja"];  
console.log(frutas.length); // Mostrará 3
```






# Agregar Elementos al Final (push()):

Explicar que `push()` es como añadir un nuevo huevo al final de la caja.

```
let listaSuper = ["pan", "leche"];  
listaSuper.push("huevos"); // Ahora listaSuper es ["pan", "leche", "huevos"]  
listaSuper.push("manzanas"); // Ahora listaSuper es ["pan", "leche", "huevos", "manzanas"]
```



# Eliminar el Último Elemento (pop()):



Explicar que `pop()` es como quitar el último huevo de la caja. Además, `pop()` nos dice cuál fue el huevo que quitamos.

```
let tareasPendientes = ["lavar platos", "hacer la cama", "estudiar"];  
let ultimaTarea = tareasPendientes.pop(); // ultimaTarea será "estudiar", y tareasPendientes ahora es ["lavar platos", "hacer la cama"]  
console.log("Tarea completada:", ultimaTarea);
```



# Actividades Prácticas Sencillas:

- "Mi Lista de Deseos":** Pide a cada alumna que cree un array con tres cosas que deseen. Luego, pídeles que muestren el primer y el último elemento de su lista.
- "Registro de Asistencia":** Simula un registro de asistencia. Comienza con un array vacío. Cada vez que una alumna diga "presente", agrega su nombre al array usando `push()`. Al final, muestra la lista de asistentes y cuántos hay usando `length`.
- "Última Canción Escuchada":** Crea un array con las últimas 5 canciones que "sonaron". Usa `pop()` para simular que la última canción terminó y mostrar cuál era.



# Pasos para visualizar el resultado de console.log()

1. Usando Node.js: <https://nodejs.org/>.
2. Extension en VSC : Code runner



# Pasos

- Guarda tu archivo JavaScript: Asegúrate de que el archivo que contiene tus `console.log()` (por ejemplo, `script.js`) esté guardado.
- Abre el Terminal en VS Code: Ve al menú superior de VS Code y selecciona Terminal > Nuevo Terminal. Esto abrirá un panel de terminal en la parte inferior de la ventana.
- Navega a la carpeta de tu proyecto: Si el terminal no se abre directamente en la carpeta donde está tu archivo JavaScript, usa el comando `cd` (change directory) para navegar hasta allí. Por ejemplo, si tu archivo está en una carpeta llamada `mi_proyecto` en tu escritorio:

```
cd Escritorio/mi_proyecto
```

**Ejecuta el archivo JavaScript con Node.js:** En el terminal, escribe el comando `node` seguido del nombre de tu archivo JavaScript y presiona Enter:

```
node script.js
```

**Visualiza la salida:** Cualquier `console.log()` que esté en tu archivo `script.js` mostrará su resultado directamente en el terminal de VS Code.



# Usando la Consola del Navegador

## Pasos:

1. **Abre el archivo HTML en tu navegador:** Haz doble clic en el archivo `.html` o haz clic derecho y selecciona "Abrir con..." y elige tu navegador preferido (Chrome, Firefox, etc.).
2. **Abre las herramientas de desarrollador del navegador:** La forma más común de hacerlo es:
  - **Chrome/Edge/Brave:** Presiona la tecla **F12** o haz clic derecho en cualquier parte de la página y selecciona "Inspeccionar" o "Inspeccionar elemento". Luego, busca la pestaña "**Consola**".
  - **Firefox:** Presiona la tecla **F12** o haz clic derecho en cualquier parte de la página y selecciona "Inspeccionar elemento". Luego, busca la pestaña "**Consola**".
  - **Safari:** Necesitas habilitar el menú "Desarrollo" en las preferencias de Safari. Luego, puedes acceder a la consola desde el menú **Desarrollo > Mostrar consola JavaScript**.
3. **Visualiza la salida:** Cualquier `console.log()` en tu código JavaScript se mostrará en la pestaña "Consola" de las herramientas de desarrollador del navegador.



# Proyecto: "Lista de Tareas Interactiva"

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Lista de Tareas</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <h1>Mi Lista de Tareas</h1>

  <div class="input-container">
    <input type="text" id="nuevaTarea" placeholder="Escribe una nueva tarea">
    <button id="agregarBtn">Agregar Tarea</button>
  </div>

  <ul id="listaDeTareas">
    </ul>

  <script src="script.js"></script>
</body>
</html>
```





# CSS

```
body {
  font-family: sans-serif;
  margin: 20px;
  background-color: #f4f4f4;
  color: #333;
}

h1 {
  text-align: center;
  margin-bottom: 20px;
  color: #007bff;
}

.input-container {
  display: flex;
  margin-bottom: 20px;
}

#nuevaTarea {
  flex-grow: 1;
  padding: 10px;
  border: 1px solid #ccc;
  border-radius: 5px 0 0 5px;
  font-size: 16px;
}
```

```
#agregarBtn {
  background-color: #007bff;
  color: white;
  border: none;
  padding: 10px 15px;
  border-radius: 0 5px 5px 0;
  cursor: pointer;
  font-size: 16px;
}

#agregarBtn:hover {
  background-color: #0056b3;
}

ul {
  list-style-type: none;
  padding: 0;
  background-color: white;
  border: 1px solid #ddd;
  border-radius: 5px;
  padding: 10px;
}
```

```
li {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 8px 10px;
  border-bottom: 1px solid #eee;
}

li:last-child {
  border-bottom: none;
}

li button {
  background-color: #dc3545;
  color: white;
  border: none;
  padding: 5px 10px;
  border-radius: 3px;
  cursor: pointer;
  font-size: 14px;
  margin-left: 10px;
}

li button:hover {
  background-color: #c82333;
}
```



// 1. Declarar un array vacío para almacenar las tareas

```
let tareas = [];
```

// 2. Obtener referencias a los elementos del DOM

```
const inputNuevaTarea = document.getElementById('nuevaTarea');
```

```
const agregarBtn = document.getElementById('agregarBtn');
```

```
const listaDeTareasUL = document.getElementById('listaDeTareas');
```




```
// Función para renderizar la lista de tareas en el HTML
function renderizarLista() {
  listaDeTareasUL.innerHTML = ""; // Limpiar la lista anterior

  // Iterar sobre el array de tareas y crear elementos <li> para cada una
  tareas.forEach((tarea, index) => {
    const nuevaLi = document.createElement('li');
    nuevaLi.textContent = tarea;

    // Opcional: Agregar un botón de eliminar a cada tarea
    const eliminarBtn = document.createElement('button');
    eliminarBtn.textContent = 'Eliminar';
    eliminarBtn.addEventListener('click', () => {
      eliminarTarea(index); // Llamar a la función para eliminar la tarea
    });
    nuevaLi.appendChild(eliminarBtn);

    listaDeTareasUL.appendChild(nuevaLi);
  });
}
```






```
// 3. Evento para agregar una nueva tarea
agregarBtn.addEventListener('click', () => {
  const nuevaTareaTexto = inputNuevaTarea.value.trim(); // Obtener el texto y eliminar espacios en blanco

  if (nuevaTareaTexto !== "") {
    tareas.push(nuevaTareaTexto); // Agregar la nueva tarea al final del array
    inputNuevaTarea.value = ""; // Limpiar el campo de entrada
    renderizarLista(); // Volver a renderizar la lista para mostrar la nueva tarea
  }
});
```





```
// 4. Función para eliminar una tarea (opcional)
function eliminarTarea(indice) {
  tareas.splice(indice, 1); // Eliminar 1 elemento en el índice especificado
  renderizarLista(); // Volver a renderizar la lista
}

// Renderizar la lista inicial (estará vacía al principio)
renderizarLista();
```





**Somos  
F5**

[www.somosf5.org](http://www.somosf5.org)

Síguenos en

