# Arrays en JavaScript: Guía de Partes Importantes

## 1. Cómo crear un array

**Explicación:** Un array es una estructura que permite almacenar varios valores en una sola variable.

**Ejemplo:**

const numeros = [1, 2, 3, 4];  
const frutas = ["manzana", "pera", "uva"];

## 2. Cómo acceder a los elementos de un array

**Explicación:** Los elementos de un array se acceden por su índice (empieza en 0).

**Ejemplo:**

console.log(frutas[0]); // Imprime: manzana

## 3. Cómo modificar elementos en un array

**Explicación:** Puedes cambiar el valor de un elemento accediendo por su índice.

**Ejemplo:**

frutas[1] = "banana";  
console.log(frutas); // ["manzana", "banana", "uva"]

## 4. Cómo agregar elementos a un array

**Explicación:** Puedes usar métodos como .push() para agregar al final, y .unshift() para agregar al inicio.

**Ejemplo:**

frutas.push("kiwi"); // Al final  
frutas.unshift("mango"); // Al inicio

## 5. Cómo eliminar elementos de un array

**Explicación:** Usa .pop() para quitar el último, .shift() para quitar el primero, o .splice() para eliminar en cualquier posición.

**Ejemplo:**

frutas.pop(); // Quita el último  
frutas.shift(); // Quita el primero  
frutas.splice(1,1); // Quita el elemento en el índice 1

## 6. Métodos comunes: .push(), .pop(), .shift(), .unshift()

* .push(elemento): Agrega al final
* .pop(): Quita el último
* .shift(): Quita el primero
* .unshift(elemento): Agrega al inicio

**Ejemplo:**

const arr = [1,2,3];  
arr.push(4); // [1,2,3,4]  
arr.pop(); // [1,2,3]  
arr.shift(); // [2,3]  
arr.unshift(0); // [0,2,3]

## 7. Método .length

**Explicación:** Retorna la cantidad de elementos del array.

**Ejemplo:**

console.log(frutas.length); // 3

## 8. Recorrer arrays con bucles (for, forEach)

**Explicación:** Puedes usar distintos tipos de bucles para procesar cada elemento del array.

**Ejemplo:**

for (let i = 0; i < frutas.length; i++) {  
 console.log(frutas[i]);  
}  
  
frutas.forEach(fruta => console.log(fruta));

## 9. Métodos de búsqueda: .find(), .findIndex(), .some(), .includes()

* .find(fn): Primer elemento que cumple condición
* .findIndex(fn): Índice del primer elemento que cumple
* .some(fn): Si al menos uno cumple
* .includes(valor): Si contiene el valor

**Ejemplo:**

const numeros = [5, 12, 8, 130, 44];  
numeros.find(x => x > 10); // 12  
numeros.findIndex(x => x > 100); // 3  
numeros.some(x => x < 0); // false  
numeros.includes(44); // true

## 10. Métodos de filtrado y transformación: .filter(), .map(), .reduce()

* .filter(fn): Nuevo array con los que cumplen
* .map(fn): Nuevo array transformado
* .reduce(fn, acumulador): Acumula un resultado

**Ejemplo:**

const pares = numeros.filter(x => x % 2 === 0);  
const dobles = numeros.map(x => x \* 2);  
const suma = numeros.reduce((acc, x) => acc + x, 0);

## 11. Arrays anidados (arrays de arrays)

**Explicación:** Un array puede contener otros arrays.

**Ejemplo:**

const matriz = [  
 [1, 2],  
 [3, 4],  
 [5, 6]  
];  
console.log(matriz[1][0]); // 3

## 12. Métodos para ordenar: .sort(), .reverse()

* .sort(): Ordena los elementos (por defecto, alfabéticamente)
* .reverse(): Invierte el orden

**Ejemplo:**

const letras = ['b','a','d','c'];  
letras.sort(); // ['a','b','c','d']  
letras.reverse(); // ['d','c','b','a']

## 13. Copiar arrays: spread operator (…), .slice()

**Explicación:** Copiar arrays evita modificar el original.

**Ejemplo:**

const copia1 = [...numeros];  
const copia2 = numeros.slice();

## 14. Métodos para combinar arrays: .concat(), spread operator (…)

**Ejemplo:**

const arr1 = [1,2];  
const arr2 = [3,4];  
const combinado1 = arr1.concat(arr2);  
const combinado2 = [...arr1, ...arr2];

## 15. Diferencias entre arrays y objetos

* Un array es una lista ordenada con índices numéricos.
* Un objeto es una colección de pares clave-valor, sin orden.

**Ejemplo:**

const persona = { nombre: "Ana", edad: 28 };  
const notas = [7, 9, 10];

## 16. Arrays de objetos

**Explicación:** Es común guardar objetos dentro de arrays.

**Ejemplo:**

const alumnos = [  
 { nombre: "Ana", edad: 20 },  
 { nombre: "Luis", edad: 22 }  
];

## 17. Métodos para buscar y modificar objetos dentro de arrays

**Explicación:** Usar .find(), .filter(), y modificar propiedades.

**Ejemplo:**

const alumno = alumnos.find(a => a.nombre === "Luis");  
alumno.edad = 23;  
  
const mayores = alumnos.filter(a => a.edad > 21);