

DEV.F.:

Clase 3

Métodos con Arreglos

Comenzamos en 10 min

6:40pm (hora CDMX)



| Temas de la clase (180 min)

Métodos

`reduce()`, `some()`, `includes()`, `every()`

Objetos

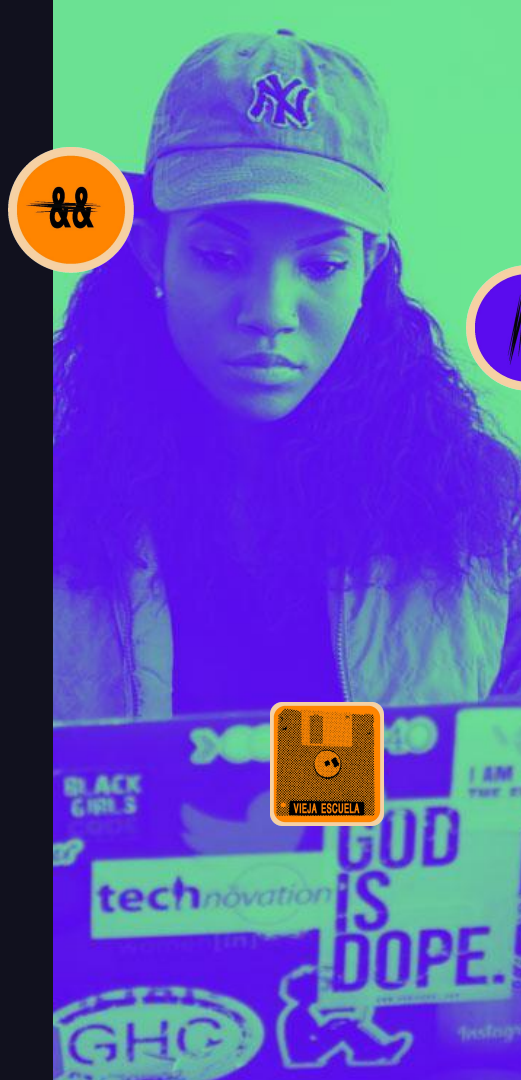
¿Qué es un objeto?

¿Para qué sirven?

Practica

Reto de la semana

DEV.F.

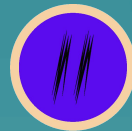


| reduce()

Función: Reduce el arreglo a un solo valor.

Cuándo usarlo: Cuando necesitas realizar un cálculo acumulativo sobre los elementos del arreglo, como sumar todos los números o encontrar el valor máximo.

```
const numeros = [1, 2, 3, 4];  
const suma = numeros.reduce((acumulador, elemento) => acumulador + elemento, 0);  
console.log(suma); // Output: 10
```



| every()

Función: Comprueba si todos los elementos del arreglo cumplen con una condición.

Cuándo usarlo: Cuando necesitas saber si todos los elementos en el arreglo satisfacen un criterio.

```
const numeros = [2, 4, 6, 8];  
const todosPares = numeros.every(elemento => elemento % 2 === 0);  
console.log(todosPares); // Output: true
```



| includes()

Función: Determina si un arreglo incluye un determinado elemento.

Cuándo usarlo: Cuando necesitas saber si un elemento específico existe en el arreglo.

```
const nombres = ["Ana", "Juan", "Pedro"];  
const incluyeJuan = nombres.includes("Juan");  
console.log(incluyeJuan); // Output: true
```



| some()

Función: Comprueba si al menos un elemento del arreglo cumple con una condición.

Cuándo usarlo: Cuando necesitas saber si existe algún elemento en el arreglo que satisface un criterio.

```
const numeros = [1, 2, 3, 4, 5];  
const hayPares = numeros.some(elemento => elemento % 2 === 0);  
console.log(hayPares); // Output: true
```



| Objetos

En JavaScript, un objeto es una colección de propiedades. Imagina un objeto como un contenedor que guarda información en forma de pares clave-valor.

Clave: Es un identificador para la propiedad, como un nombre.

Valor: Es el dato que se asocia a la clave. Puede ser de cualquier tipo: un número, una cadena de texto, un booleano, otro objeto, una función, etc.

```
let person = {  
  firstName: 'John',  
  lastName: 'Doe'  
};
```

```
person.firstName = 'Jane';
```

```
console.log(person);
```

```
//{ firstName: 'Jane', lastName: 'Doe' }
```

| ¿Para qué sirven los objetos?

Los objetos son muy útiles para:

- **Representar entidades del mundo real:** Puedes usar objetos para modelar personas, animales, productos, etc.
- **Organizar el código:** Los objetos te permiten agrupar datos y funciones relacionadas, haciendo tu código más modular y fácil de mantener.
- **Crear estructuras de datos complejas:** Puedes crear objetos que contienen otros objetos, formando estructuras jerárquicas.

Diagram illustrating the structure of a JavaScript object literal:

```
const miAuto = {  
  color: "rojo",  
  marca: "Toyota",  
  modelo: "Prius",  
}
```

Annotations:

- Nombre del objeto:** Points to the variable name `miAuto`.
- Propiedades:** A bracket on the right side of the object, indicating the collection of properties.
- llave:** Points to the property name `color`.
- valor:** Points to the property value `"rojo"`.

Ejemplo 1

```
let estudiante = {  
  nombre: "Ana",  
  calificaciones: [90, 85, 92],  
  materias: ["Matemáticas", "Historia", "Ciencias"]  
};  
  
console.log(estudiante.calificaciones.length); // 3  
console.log(estudiante.materias[0]); // "Matemáticas"
```

En este ejemplo, el **objeto estudiante** tiene una **propiedad calificaciones** que almacena un **arreglo de números** y una propiedad **materias** que **almacena un arreglo de cadenas**.



Ejemplo 2

```
let carritoCompras = {  
  productos: [],  
  agregarProducto: function(producto) {  
    this.productos.push(producto);  
  },  
  eliminarProducto: function(indice) {  
    this.productos.splice(indice, 1);  
  }  
};  
  
carritoCompras.agregarProducto("Manzanas");  
carritoCompras.agregarProducto("Plátanos");  
console.log(carritoCompras.productos); // ["Manzanas", "Plátanos"]  
  
carritoCompras.eliminarProducto(0);  
console.log(carritoCompras.productos); // ["Plátanos"]
```

En este caso, el **objeto carritoCompras** tiene un **arreglo productos** y dos métodos:

- **agregarProducto** (que usa **push()** para agregar elementos al arreglo)
- **eliminarProducto** (que usa **splice()** para eliminar elementos)

Puedes crear arreglos donde cada elemento es un objeto:

```
let estudiantes = [  
  { nombre: "Ana", edad: 20 },  
  { nombre: "Juan", edad: 22 },  
  { nombre: "María", edad: 21 }  
];  
  
console.log(estudiantes[0].nombre); // "Ana"
```

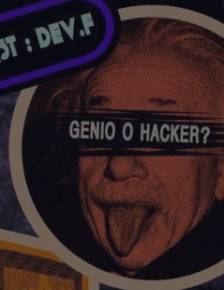
En este ejemplo, `estudiantes` es un arreglo que contiene objetos, cada uno con las propiedades `nombre` y `edad`.



Hora de codear



CONST : DEV.F



DEV.F FUNCTION ()



No olviden



Las lecturas de **edu.devf.la**

