



Colecciones de Datos

Centro de Servicios y Gestión Empresarial
SENA Regional Antioquia

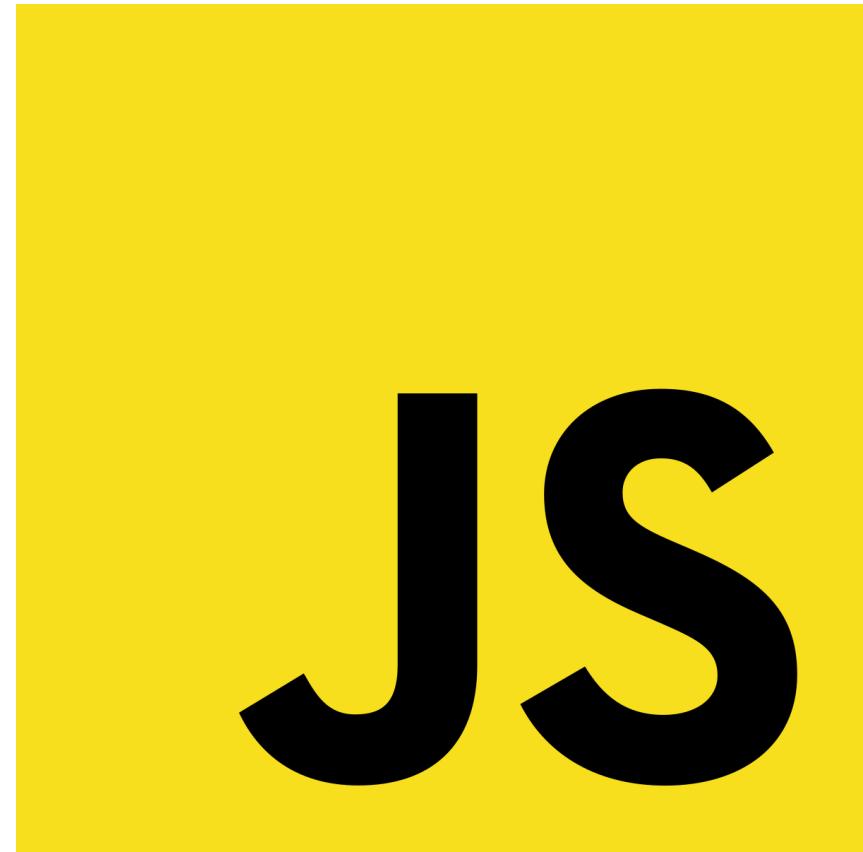
Contextualización

Colecciones de Datos



Las colecciones de datos en JavaScript nos permiten almacenar y gestionar conjuntos de valores de manera eficiente.

Existen cuatro tipos principales de colecciones.



Tipos de Colecciones



Arrays ([])

- Listas ordenadas de elementos indexados numéricamente.

Objects ({})

- Conjuntos de pares clave-valor.

Sets (new Set())

- Colección de valores únicos sin duplicados.

Maps (new Map())

- Conjuntos clave-valor con claves únicas y cualquier tipo de dato.

Arrays ([])

*Listas ordenadas de elementos indexados
numéricamente*

Arrays ([])

Los arrays almacenan valores en una lista ordenada y permiten acceder a los elementos por su índice numérico (empezando en 0).

🔨 Declaración de un Array

```
var frutas = ["Manzana", "Plátano", "Cereza", "Fresa"];
console.log(frutas);

let objetos = ["Mesa", "Silla", "Ordenador", "Lámpara"];
console.log(objetos[2]);
```

Métodos - Arrays ([])

Método	Descripción	Ejemplo
push()	Agrega un elemento al final	frutas.push("Mango")
pop()	Elimina el último elemento	frutas.pop()
unshift()	Agrega un elemento al inicio	frutas.unshift("Uva")
shift()	Elimina el primer elemento	frutas.shift()
length	Devuelve el tamaño del array	frutas.length
indexOf()	Busca un elemento y devuelve su índice	frutas.indexOf("Banana")
splice()	Elimina elementos en una posición	frutas.splice(1, 1)

Array

```
// Declaración del arreglo o Definición del arreglo
const autos = ["BMW", "Volvo", "Toyota"];
// Declaración del arreglo o Definición del arreglo
const autosUno = [
    "BMW",
    "Volvo",
    "Toyota"
];
// Instanciar el arreglo
const autosDos = new Array("BMW", "Volvo", "Toyota");
// Inicializar el arreglo
const autosTres = [];
```

Array - Métodos

```
let objetos = ["Mesa", "Silla", "Ordenador", "Lámpara"];  
  
// Reconocer el tipo de dato de un arreglo  
// typeof --> Devuelve el tipo de dato de la variable  
console.log(typeof objetos);  
// Array.isArray() --> Devuelve true si la variable es un arreglo  
console.log(Array.isArray(objetos));  
// instanceof --> Devuelve true si la variable es un arreglo  
console.log(objetos instanceof Array);  
  
// Longitud de un arreglo  
console.log(objetos.length);  
  
// Acceder a un elemento del arreglo  
console.log(objetos[1]);
```

Array - Métodos

```
// Añadir un elemento al arreglo
console.log(objetos.push("Sofá"));
console.log(objetos);

// Añadir un elemento al inicio del arreglo
console.log(objetos.unshift("Estantería"));
console.log(objetos);

// Añadir un elemento en una posición específica del arreglo
console.log(objetos.splice(2, 0, "Mesa de Centro"));

// Modificar un elemento del arreglo
objetos[1] = "Sillón";
console.log(objetos);
```

Array - Métodos

```
// Eliminar un elemento al final del arreglo  
console.log(objetos.pop());  
console.log(objetos);  
  
// Eliminar un elemento al inicio del arreglo  
console.log(objetos.shift());  
console.log(objetos);  
  
// Eliminar un elemento en una posición específica del arreglo  
console.log(objetos.splice(2, 1));  
console.log(objetos);  
  
// Buscar un elemento en el arreglo  
console.log(objetos.indexOf("Silla"));  
// Buscar un elemento en el arreglo  
console.log(objetos.includes("Silla"));
```

Array - Métodos

```
// Ordenar un arreglo
console.log(objetos.sort());
console.log(objetos);

// Invertir el orden de los elementos del arreglo
console.log(objetos.reverse());
console.log(objetos);

// Ciclos en arreglos
// For
for (let i = 0; i < objetos.length; i++) {
    console.log(objetos[i]);
}

// ForEach
objetos.forEach(elemento => {
    console.log(elemento);
});
```

Array - Métodos

```
// Concatenar arreglos
let objetosDos = ["Mesa de Centro", "Sofá"];
let objetosTres = objetos.concat(objetosDos);
console.log(objetosTres);

// toString() --> Convierte un arreglo en una cadena de
// texto
console.log(objetos.toString());

// join() --> Convierte un arreglo en una cadena de texto
console.log(objetos.join(" - "));
```

Array - Métodos

```
// Métodos estadísticos
// Math.max() --> Devuelve el número más grande de un
// arreglo
let numeros = [32, 45, 29, 15, 78, 56];
console.log(Math.max(...numeros));
// Math.min() --> Devuelve el número más pequeño de un
// arreglo
console.log(Math.min(...numeros));
```

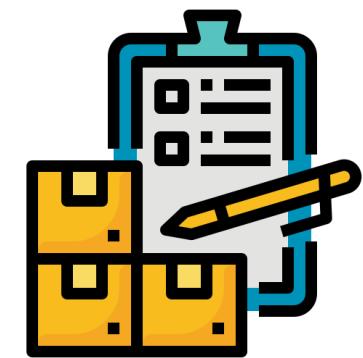
Array - Ejercicio

🔨 Ejercicio: Gestión de Inventario

Una tienda necesita gestionar su inventario de productos. La aplicación debe permitir al usuario gestionar las siguientes acciones:

- Ver la lista de productos disponibles: "*Laptop*", "*Mouse*", "*Teclado*", "*Monitor*"
- Agregar un nuevo producto.
- Eliminar un producto.
- Ordenar la lista de productos alfabéticamente.
- Salir del programa.

El uso de funciones y array es obligatorio.



Objects ({})

Conjuntos de pares clave-valor.

Objects ({ })

Los objetos permiten almacenar datos como pares clave-valor, facilitando la representación de información estructurada.

🔨 Declaración de un Objects

```
let persona = {  
    nombre: "Carlos",  
    edad: 30,  
    ciudad: "Medellín"  
};  
console.log(persona.nombre); // "Carlos"  
console.log(persona["edad"]); // 30
```

Métodos - Objects ({})



Método / Operación	Descripción	Ejemplo
Object.keys(obj)	Retorna un array con todas las claves del objeto.	Object.keys(persona) → ["nombre", "edad", "ciudad"]
Object.values(obj)	Retorna un array con todos los valores del objeto.	Object.values(persona) → ["Carlos", 30, "Medellín"]
Object.entries(obj)	Retorna un array de pares [clave, valor] .	Object.entries(persona) → [["nombre", "Carlos"], ["edad", 30]]
Object.assign(obj, nuevoObj)	Copia propiedades de otro objeto al existente.	Object.assign(persona, { país: "Colombia" })
delete obj.clave	Elimina una propiedad del objeto.	delete persona.edad
Object.hasOwnProperty("clave") (ES2022)	Verifica si el objeto tiene una clave específica .	Object.hasOwnProperty("edad") → true
Object.freeze(obj)	Bloquea un objeto para que no pueda modificarse.	Object.freeze(persona)
Object.seal(obj)	Permite modificar valores, pero no agregar ni eliminar propiedades.	Object.seal(persona)
Object.fromEntries(array)	Convierte un array de pares clave-valor en un objeto.	Object.fromEntries([["nombre", "Ana"], ["edad", 25]])
Object.getOwnPropertyNames(obj)	Devuelve todas las claves , incluyendo propiedades no enumerables.	Object.getOwnPropertyNames(persona)

Objects ({ })

```
let persona = {  
    nombre: "Carlos",  
    edad: 30,  
    ciudad: "Medellín"  
};  
  
// 1. Obtener Claves y Valores del Objeto  
  
// ["nombre", "edad", "ciudad"]  
console.log(Object.keys(persona));  
// ["Carlos", 30, "Medellín"]  
console.log(Object.values(persona));  
// [["nombre", "Carlos"], ["edad", 30], ["ciudad", "Medellín"]]  
console.log(Object.entries(persona));
```

Objects ({ })

```
// Agregar, Modificar y Eliminar Propiedades
persona.apellido = "Gómez";
persona.edad = 31;
delete persona.ciudad;
console.log(persona);

// Verificar si una Propiedad Existe
console.log("nombre" in persona); // true
console.log(Object.hasOwnProperty(persona, "telefono")); // false

// Convertir un Array en Objeto
let datos = [["nombre", "Ana"], ["edad", 25]];
let objetoDesdeArray = Object.fromEntries(datos);
console.log(objetoDesdeArray); // { nombre: "Ana", edad: 25 }
```

Objects ({ })

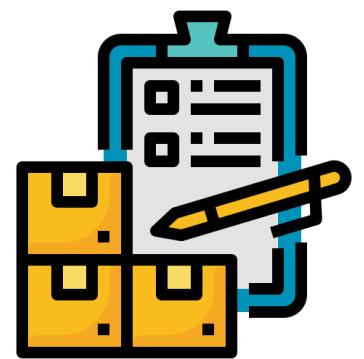
```
// Proteger un Objeto Contra Modificaciones  
  
Object.freeze(persona);  
  
persona.edad = 40; // No se modifica  
persona.ciudad = "Bogotá"; // No se añade  
delete persona.pais; // No se elimina  
  
// { nombre: "Carlos", edad: 31, pais: "Colombia" }  
console.log(persona);
```

Objects - Ejercicio

📌 **Ejercicio:** Agenda Telefónica

Realice una aplicación que permita gestionar los siguientes procedimientos:

- Crear nuevos contactos en una agenda (contacto y teléfono).
- Editar el número de un contacto existente.
- Eliminar contactos.
- Buscar contactos por nombre.
- Ver todos los contactos almacenados.



Sets (new Set())

Colección de valores únicos sin duplicados.

Sets (new Set())

Los Sets almacenan valores únicos, eliminando duplicados automáticamente.

🔨 Declaración de un Objects

```
// Variables Colecciones Datos - Set
let numeros = new Set([10, 20, 30, 10, 20]);
console.log(numeros); // {10, 20, 30}
(elimina duplicados)
```

Métodos - Sets (new Set())

Método	Descripción	Ejemplo
add()	Agrega un elemento	numeros.add(40)
delete()	Elimina un elemento	numeros.delete(10)
has()	Verifica si existe un valor	numeros.has(20)
size	Devuelve el tamaño del Set	numeros.size

Sets (new Set())

```
let colores = new Set();
// Agregar elementos
colores.add("Rojo");
colores.add("Azul");
colores.add("Rojo"); // No se duplica
// Eliminar elementos
colores.delete("Azul");
// Verificar si un elemento existe
console.log(colores.has("Rojo")); // true
// Tamaño del conjunto
console.log(colores.size); // 2
// Vaciar el conjunto
colores.clear();
console.log(colores); // {}
```



G R A C I A S

Presentó: Alvaro Pérez Niño

Instructor Técnico

Correo: aperezn@sena.edu.co

<http://centrodesserviciosygestionempresarial.blogspot.com/>

Línea de atención al ciudadano: 01 8000 910270

Línea de atención al empresario: 01 8000 910682



www.sena.edu.co