DEV.F::

Clase 2

Repaso: Arreglos

Comenzamos en 10 min

6:40pm (hora CDMX)

Temas de la clase (180 min)

Repaso de Arreglos

Declaración
Acceso
Otras formas de acceso

Metodos

Función VS Métodos

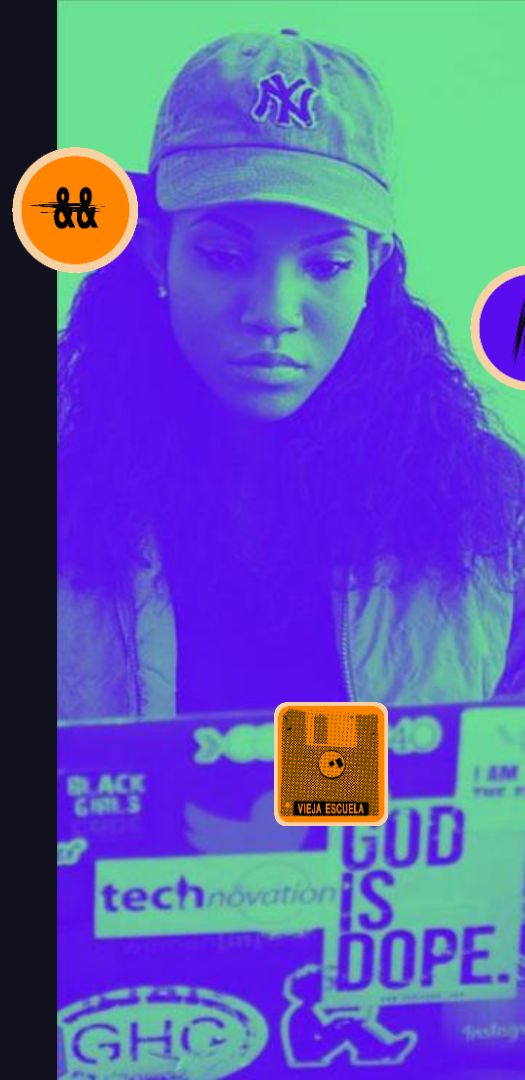
Métodos para arreglos

map(), forEach(), some(), filter(), find()

Practica

Ejercicios

Reto de la semana



DEV.F::

¿Qué son los arreglos en JavaScript?

Un arreglo (array) es una estructura de datos que te permite almacenar una colección ordenada de elementos. Estos elementos pueden ser de cualquier tipo de dato en JavaScript, como números, cadenas de texto, objetos, e incluso otros arreglos.

Piensa en un arreglo como una lista de compras. Cada elemento de la lista es un elemento del arreglo, y puedes acceder a ellos por su posición en la lista (índice).

¿Cómo se declaran?

- Usa `const` cuando no necesites reasignar el arreglo a una nueva referencia, pero sí modificar su contenido. Esta es la opción preferida en la mayoría de los casos, ya que promueve la inmutabilidad y reduce la posibilidad de errores.
- Usa `let` cuando necesites reasignar la variable a un arreglo completamente diferente.

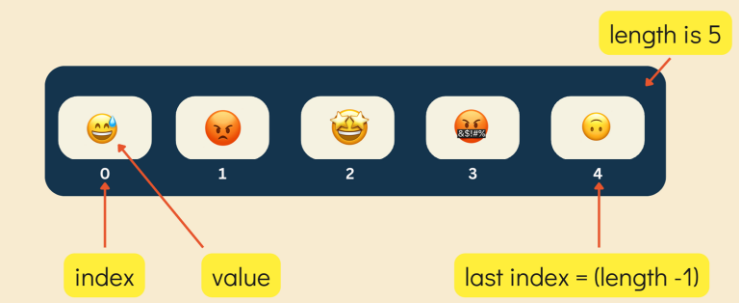
Recuerda que la elección entre `let` y `const` depende de cómo planeas usar el arreglo en tu código.

```
let mountains = ['Everest', 'Fuji',
'Nanga Parbat'];
console.log(mountains[0]); // 'Everest'
console.log(mountains[1]); // 'Fuji'
console.log(mountains[2]); // 'Nanga Parbat'
mountains.push('Popocatepetl');
mountains.indexOf('Fuji');
```

DEV.F::

Acceso

- Los índices en JavaScript comienzan en 0. Es decir, el primer elemento tiene índice 0, el segundo tiene índice 1, y así sucesivamente.
- Puedes usar el índice para acceder directamente al elemento que se encuentra en esa posición específica del arreglo.



DEV.F::

Otras formas

Además de acceder a los elementos de un arreglo por su índice numérico (como `miArreglo[0]`), existen otras formas de recorrer y acceder a cada elemento en JavaScript.

Bucles `for`:

El bucle `for` tradicional te permite iterar sobre los elementos del arreglo utilizando un contador.

Puedes acceder a cada elemento usando el contador como índice.

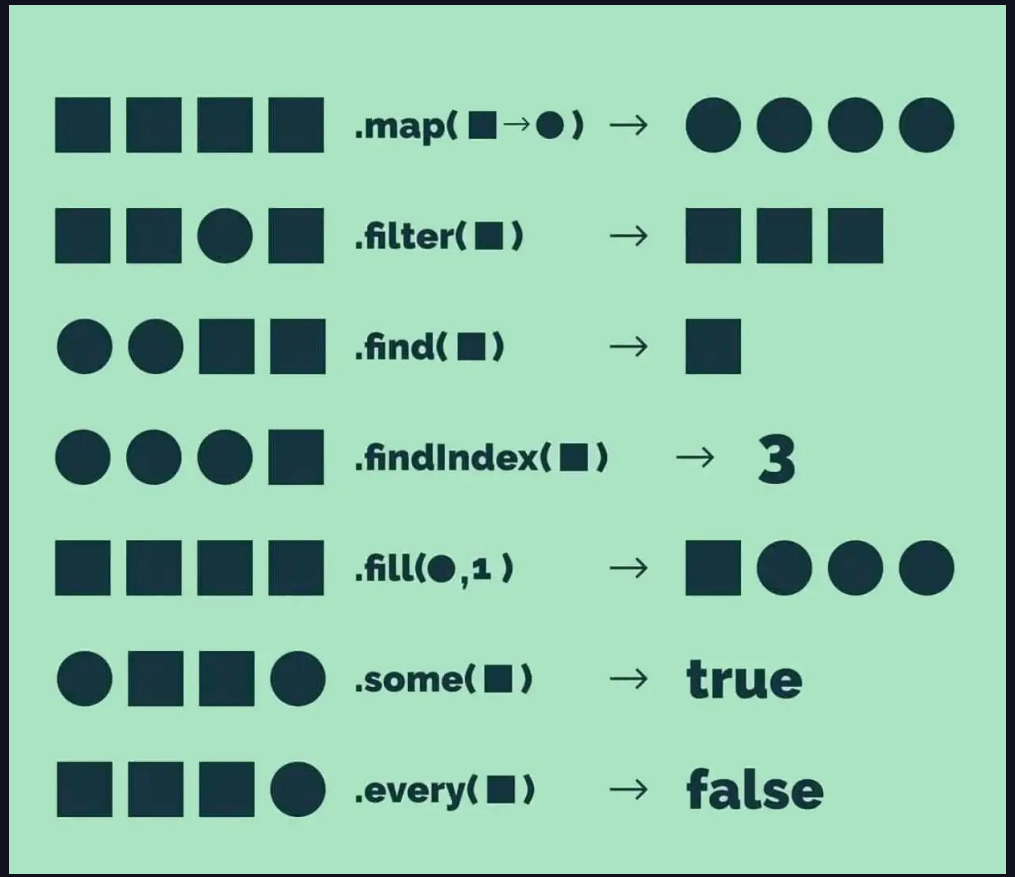
```
const numeros = [10, 20, 30, 40];

for (let i = 0; i < numeros.length; i++) {
  console.log(numeros[i]); // Accede al elemento en la posición i
}
```

DEV.F::

Metodos

¿Los métodos y las funciones son lo mismo?



Función VS Método

- **Martillo:** Sirve para clavar clavos.
- **Destornillador:** Sirve para atornillar y desatornillar tornillos.
- **Llave inglesa:** Sirve para ajustar tuercas.

En este caso, cada **herramienta** sería como una **función** en JavaScript. Son independientes y realizan una tarea específica.

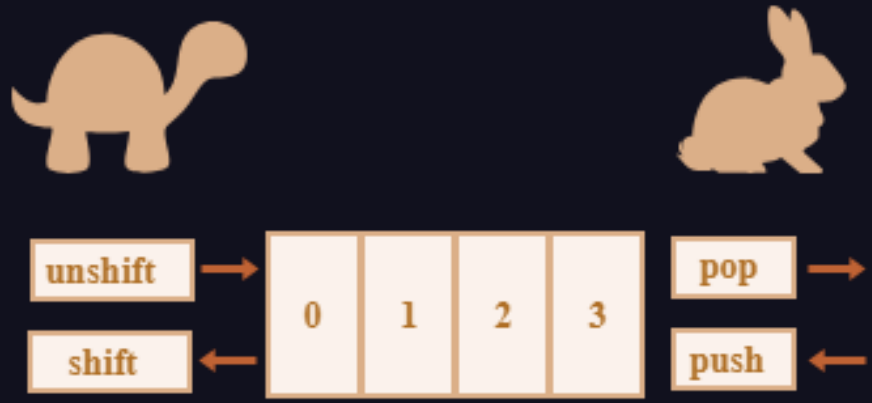
- **Acelerar:** Hace que el coche aumente su velocidad.
- **Frenar:** Hace que el coche disminuya su velocidad.
- **Girar el volante:** Hace que el coche cambie de dirección.

En este caso, "acelerar", "frenar" y "girar el volante" serían como **métodos** en JavaScript. Son funciones, pero están asociadas al objeto "coche" y operan sobre él. No puedes "frenar" sin un coche, ¿verdad?

Métodos básicos

Imagina un arreglo como una fila de personas:

- `unshift()` agrega a alguien al principio de la fila.
- `shift()` hace que la primera persona de la fila se retire.
- `push()` agrega a alguien al final de la fila.
- `pop()` hace que la última persona de la fila se retire.
- `length`: Devuelve la longitud (número de elementos) de un arreglo.



DEV.F::

Metodos

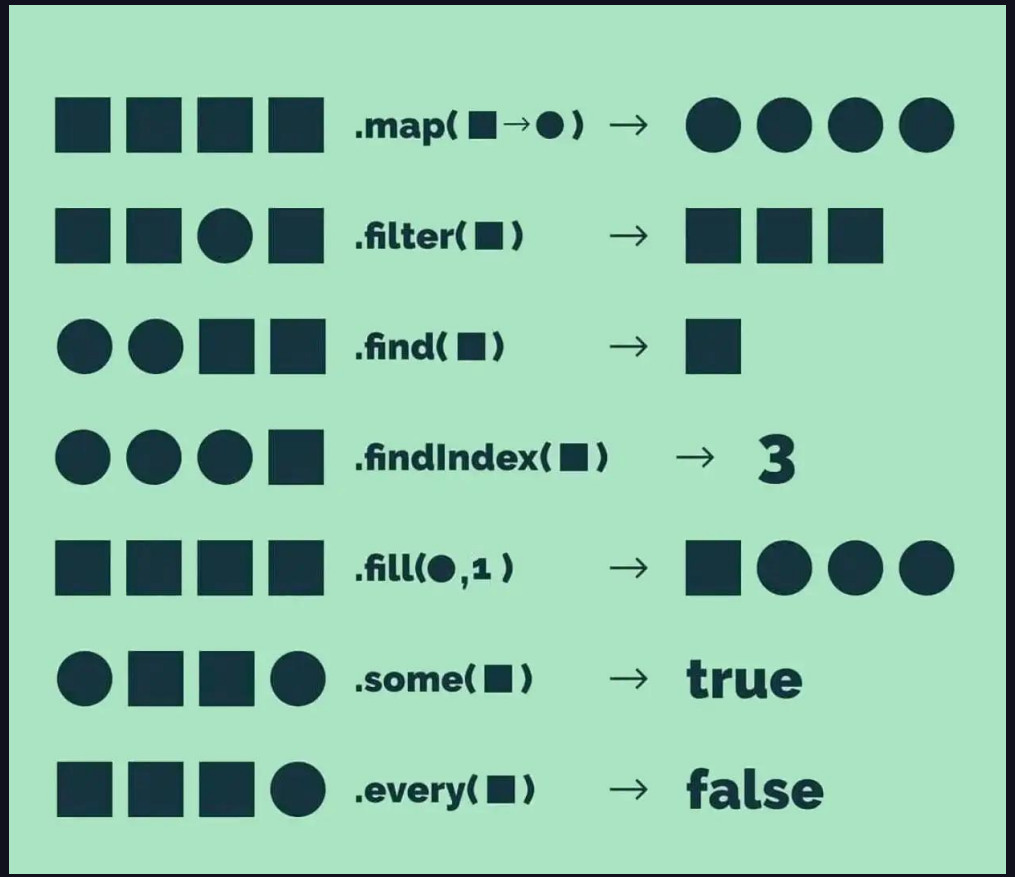
¿Los métodos y las funciones son lo mismo?

Entonces, la diferencia clave es:

Función: Es como una herramienta independiente que puedes usar en diferentes situaciones.

Método: Es como una función que está integrada en un objeto específico y solo se puede usar con ese objeto.

Los método están asociado al objeto `Array` (el objeto que representa a los arreglos en JavaScript).



find()

Función: Encuentra el primer elemento en el arreglo que cumple con una condición dada.

Cuándo usarlo: Cuando necesitas obtener un elemento específico del arreglo que satisface un criterio.

```
const numeros = [1, 5, 10, 15];
const encontrado = numeros.find(elemento => elemento > 9);
console.log(encontrado); // Output: 10
```

DEV.F::

map()

Función: Crea un nuevo arreglo con los resultados de aplicar una función a cada elemento del arreglo original.

Cuándo usarlo: Cuando necesitas transformar cada elemento del arreglo de alguna manera.

```
const numeros = [1, 2, 3, 4];
const dobles = numeros.map(elemento => elemento * 2);
console.log(dobles); // Output: [2, 4, 6, 8]
```

DEV.F::

filter()

Función: Crea un nuevo arreglo que contiene solo los elementos del arreglo original que cumplen con una condición.

Cuándo usarlo: Cuando necesitas obtener un subconjunto de elementos del arreglo que satisfacen un criterio

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

DEV.F::

forEach()

Función: Ejecuta una función proporcionada una vez por cada elemento del arreglo.

Cuándo usarlo: Cuando necesitas realizar una acción para cada elemento del arreglo, como imprimirlos en la consola o modificarlos.

```
const nombres = ["Ana", "Juan", "Pedro"];
nombres.forEach(nombre => console.log("Hola, " + nombre + "!"));
```

DEV.F::

sort()

Función: Ordena los elementos del arreglo.

Cuándo usarlo: Cuando necesitas ordenar los elementos del arreglo en un orden específico, ya sea ascendente o descendente.

```
const numeros = [3, 1, 4, 2];
numeros.sort((a, b) => a - b); // Orden ascendente
console.log(numeros); // Output: [1, 2, 3, 4]
```

DEV.F::

Hora de codear

No olviden

Las lecturas de edu.devf.la