Clase 16

JavaScript Parte 4

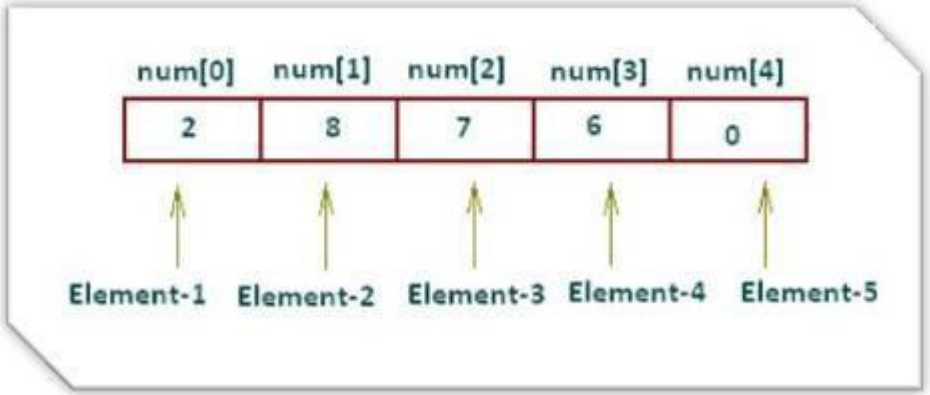
Arrays

Los Arrays son colecciones que nos permiten añadir varios valores dentro de un elemento. Dicho de otra manera, permiten almacenar varios valores en una sola referencia.

```
var array_name = [item1, item2, ...];
```

Arrays

Un arreglo es una colección de elementos contiguos, más bien una estructura que nos permite almacenar secuencialmente datos, no necesariamente en orden. ¿Qué tipo de datos?, en el caso de Javascript, cualquiera.



Arrays

Los elementos pueden ser de cualquier tipo permitido e incluso contener otros arrays.

```
var days = ['Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes', 'Sábado', 'Domingo'];
```

```
var hodgepodge = [100, "paint", [200, "brush"], false];
```

Arrays

Para acceder a uno de los elementos dentro de un array, se utilizan los corchetes [ ] y un número que especifica la posición dentro del mismo, por ejemplo:
myArray [3]
La primer posición es 0, por lo que el primer elemento siempre estará dentro de myArray[0].

```
var sisters = ["Tia", "Tamera"];

sisters[0];
```

```
OUTPUT
"Tia"
```

Arrays

Poseen distintas propiedades y métodos:

- ✔ **lenght:** cantidad de elementos del array.
- ✔ **concat:** retorna un nuevo array que combina los valores de dos arrays.
- ✔ **pop:** remueve el último elemento del array y lo retorna.

```
["Jupiter", "Saturn", "Uranus", "Neptune", "Pluto"].pop();
```

OUTPUT

```
"Pluto"
```

```
["tortilla chips"].concat(["salsa", "queso", "guacamole"]);
```

OUTPUT

```
["tortilla chips", "salsa", "queso", "guacamole"]
```

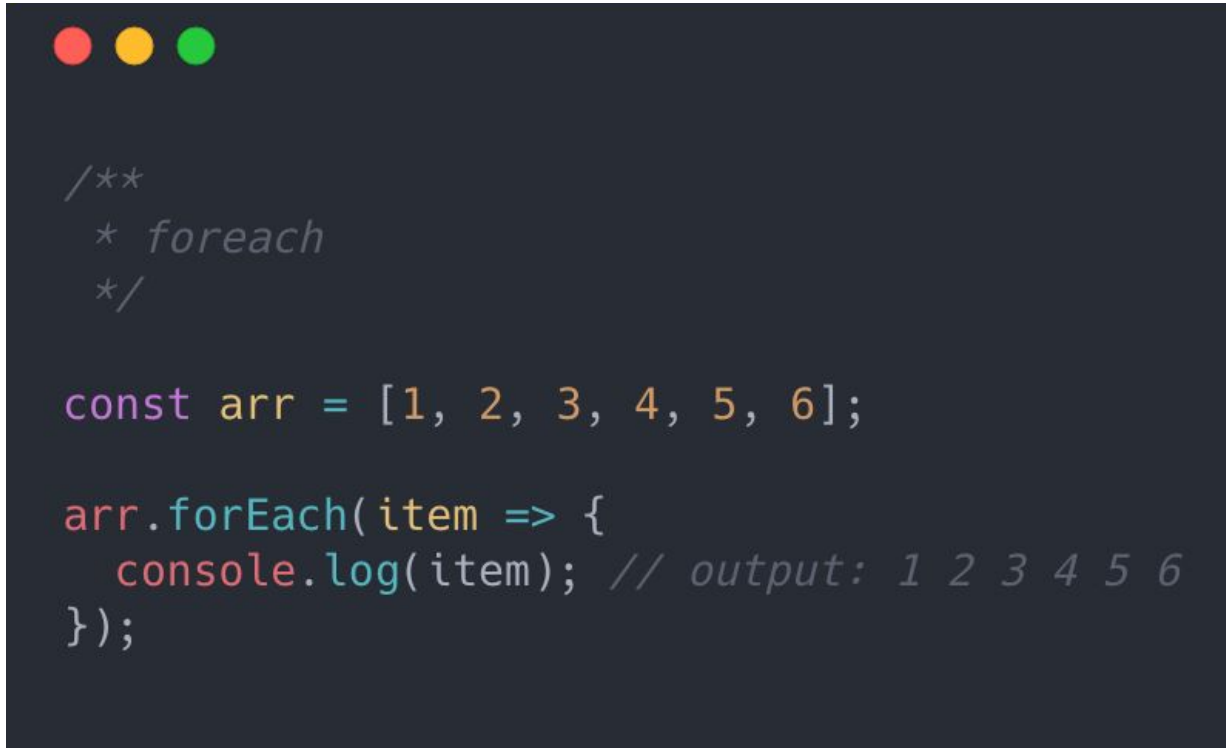
```
["a", "b", "c", 1, 2, 3].length;
```

OUTPUT

```
6
```

forEach

El método forEach() ejecuta la función indicada una vez por cada elemento del array. Este método ayuda a recorrer los elementos de un array.



```
/**
 * foreach
 */

const arr = [1, 2, 3, 4, 5, 6];

arr.forEach(item => {
  console.log(item); // output: 1 2 3 4 5 6
});
```

Every

El método every() verifica si todos los elementos en el arreglo pasan la prueba implementada por la función dada.

```
/**
 * every
 */

const arr = [1, 2, 3, 4, 5, 6];

// all elements are greater than 4
const greaterFour = arr.every(num => num > 4);
console.log(greaterFour); // output: false

// all elements are less than 10
const lessTen = arr.every(num => num < 10);
console.log(lessTen); // output: true
```

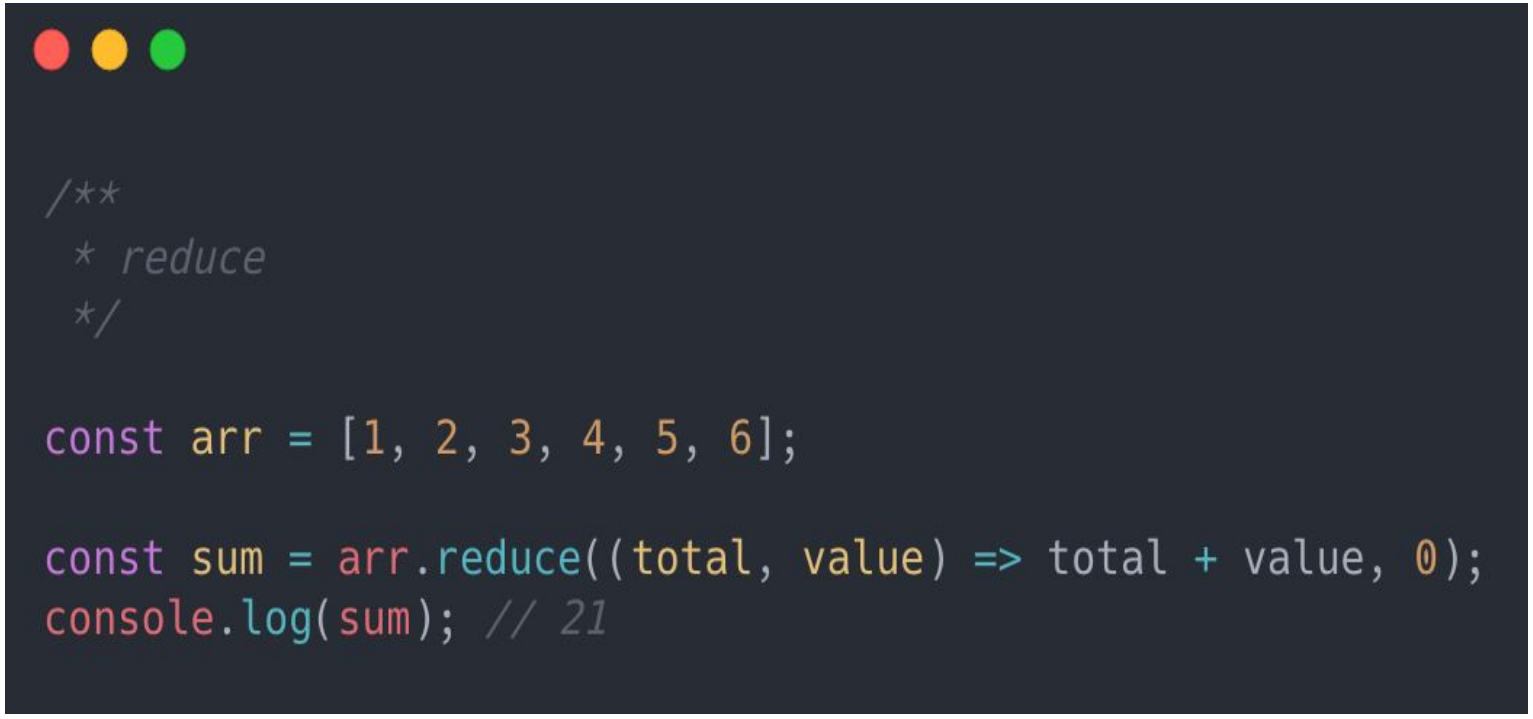
Some

El método some() verifica si algún elemento de un array cumple con el test implementado por la función brindada. Si se aprueba, devuelve “true” de lo contrario “false”.

```
/**
 * some
 */

const arr = [1, 2, 3, 4, 5, 6];

// at least one element is greater than 4?
const largeNum = arr.some(num => num > 4);
console.log(largeNum); // output: true

// at least one element is less than or equal to 0?
const smallNum = arr.some(num => num <= 0);
console.log(smallNum); // output: false
```

Reduce

Es una función que se va a ejecutar para cada elemento del array y luego devolver un único valor conocido como acumulado. Se podría utilizar cuando necesitas acumular cosas, por ejemplo sacar el total de las compras, también se puede utilizar para sacar promedios.



```
/**
 * reduce
 */

const arr = [1, 2, 3, 4, 5, 6];

const sum = arr.reduce((total, value) => total + value, 0);
console.log(sum); // 21
```

Map

Se ejecuta por cada elemento del array, y crea un nuevo array a partir del resultado de cada ejecución por elemento. Se podría utilizar cuando tenemos un array y sabemos que por cada elemento vamos a realizar un cambio.

```
/**
 * map
 */

const arr = [1, 2, 3, 4, 5, 6];

// add one to every element
const oneAdded = arr.map(num => num + 1);
console.log(oneAdded); // output [2, 3, 4, 5, 6, 7]

console.log(arr); // output: [1, 2, 3, 4, 5, 6]
```

Filter

Se ejecuta por cada elemento de array, y crea un nuevo array con los elementos que cumplen cierta condición. Podríamos usarlo cuando tenemos un array y necesitamos filtrar datos.

```
/**
 * filter
 */

const arr = [1, 2, 3, 4, 5, 6];

// item(s) greater than 3
const filtered = arr.filter(num => num > 3);
console.log(filtered); // output: [4, 5, 6]

console.log(arr); // output: [1, 2, 3, 4, 5, 6]
```