Métodos de un array (Parte 1)

DigitalHouse >
Coding School

“

Para JavaScript los arrays son un tipo especial de objetos.

Por esta razón disponemos de muchos **métodos** muy útiles a la hora de trabajar con la información que hay adentro.

”

Ya vimos antes que una función es un bloque de código que nos permite agrupar funcionalidad para usarla muchas veces.

Cuando una **función le pertenece a un objeto**, en este caso nuestro array, la llamamos **método**.

.push()

Agrega uno o varios elementos al final del array.

- **Recibe** uno o más elementos como parámetros.
- **Retorna** la nueva longitud del array.

```
let colores = ['Rojo','Naranja','Azul'];
colores.push('Violeta'); // retorna 4
console.log(colores); // ['Rojo','Naranja','Azul','Violeta']
{}

colores.push('Gris','Oro');
console.log(colores);
// ['Rojo','Naranja','Azul','Violeta','Gris','Oro']
```

.pop()

Elimina el último elemento de un array.

- **No recibe** parámetros.
- **Devuelve** el elemento eliminado.

```
let series = ['Mad Men','Breaking Bad','The Sopranos'];

// creamos una variable para guardar lo que devuelve .pop()
let ultimaSerie = series.pop();

console.log(series); // ['Mad men', 'Breaking Bad']
console.log(ultimaSerie); // ['The Sopranos']
```

.shift()

Elimina el primer elemento de un array.

- **No recibe** parámetros.
- **Devuelve** el elemento eliminado.

```
let nombres = ['Frida','Diego','Sofía'];

// creamos una variable para guardar lo que devuelve .shift()
let primerNombre = nombres.shift();

console.log(nombres); // ['Diego', 'Sofia']
console.log(primerNombre); // ['Frida']
```

.unshift()

Agrega uno o varios elementos al principio de un array.

- **Recibe** uno o más elementos como parámetros.
- **Retorna** la nueva longitud del array.

```
let marcas = ['Audi'];

marcas.unshift('Ford');
console.log(marcas); // ['Ford', 'Audi']

marcas.unshift('Ferrari','BMW');
console.log(marcas); // ['Ferrari','BMW','Ford', 'Audi']
```

.join()

Une los elementos de un array utilizando el separador que le especifiquemos. Si no lo especificamos, utiliza comas.

- **Recibe** un separador (string), **es opcional**.
- **Retorna** un string con los elementos unidos.

```
let dias = ['Lunes','Martes','Jueves'];

let separadosPorComa = dias.join();
console.log(separadosPorComa); // 'Lunes,Martes,Jueves'

let separadosPorGuion = dias.join(' - ');
console.log(separadosPorGuion); // 'Lunes - Martes - Jueves'
```

.indexOf()

Busca en el array el elemento que recibe como parámetro.

- **Recibe** un elemento a buscar en el array.
- **Retorna** el primer índice donde encontró lo que buscábamos. Si no lo encuentra, retorna un -1.

```
{}
let frutas = ['Manzana','Pera','Frutilla'];
frutas.indexOf('Frutilla');
// Encontró lo que buscaba. Devuelve 2, el índice del elemento

frutas.indexOf('Banana');
// No encontró lo que buscaba. Devuelve -1
```

.lastIndexOf()

Similar a `.indexOf()`, con la salvedad de que empieza buscando el elemento por el **final del array** (de atrás hacia adelante).

En caso de haber elementos repetidos, devuelve la posición del primero que encuentre (o sea el último si miramos desde el principio).

```
let clubes = ['Racing','Boca','Lanús','Boca'];

clubes.lastIndexOf('Boca');
{}  // Encontró lo que buscaba. Devuelve 3

clubes.lastIndexOf('River');
// No encontró lo que buscaba. Devuelve -1
```

.includes()

También similar a `.indexOf()`, con la salvedad que retorna un booleano.

- **Recibe** un elemento a buscar en el array.
- **Retorna** *true* si encontró lo que buscábamos, *false* en caso contrario.

```
let frutas = ['Manzana','Pera','Frutilla'];

frutas.includes('Frutilla');
{} // Encontró lo que buscaba. Devuelve true

frutas.includes('Banana');
// No encontró lo que buscaba. Devuelve false
```


DigitalHouse>
Coding School