Para continuar con otros métodos para recorrer arrays, aprenderás find, forEach y some.Utilizaremos el siguiente array de objetos para los ejemplos de los métodos:

```
var articulos = [
{ nombre: "Bici", costo: 3000 },
{ nombre: "TV", costo: 2500 },
{ nombre: "Libro", costo: 320 },
{ nombre: "Celular", costo: 10000 },
{ nombre: "Laptop", costo: 20000 },
{ nombre: "Teclado", costo: 500 },
{ nombre: "Audifonos", costo: 1700 },
]
```

Cómo utilizar el método find

El método find consiste en encontrar el primer elemento de un array que cumpla con la condición especificada en la función (callback). Si ningún elemento cumpla con la condición, retornará undefined.

El método find recibe dos argumentos:

```
La función que itera y evalúa cada elemento del array hasta encuentre uno que
cumpla con la condición especificada (obligatorio).
Un objeto al que puede hacer referencia el contexto this en la función. Si se
lo omite, será undefined.
```

array.find(function(), thisArg)

La función, que recibe como argumento, utiliza tres parámetros opcionales:

```
El valor actual del elemento iterado. Es decir, si es la primera iteración,
será el primer elemento, y así sucesivamente.
El índice del elemento iterado. Es decir, si es la primera iteración, será el
índice 0, y así sucesivamente.
El array que está iterando.
```

array.find(function(element, index, array))

Practiquemos el uso del método find

Utilicemos el array articulos que definimos para encontrar algún artículo que su nombre sea Laptop.

Entonces utilizamos el método find que retorne la condición que necesitamos. Recuerda que el primer parámetro de la función callback es cada uno de los elementos del array.

```
var algunArticulo = articulos.find(function (articulo) {
return (articulo.nombre = "Laptop")
```

```
})
console.log(algunArticulo)
/*
{ nombre: 'Laptop', costo: 3000 }
*/
```

Cómo utilizar el método forEach

El método forEach de los arrays consiste en ejecutar una función (callback) para cada uno de los elementos iterados. Iterar significa repetir una acción varias veces. Este método no retorna ningún valor.

Este método recibe dos argumentos:

```
La función que itera cada elemento del array (obligatorio).
Un objeto al que puede hacer referencia el contexto this en la función. Si se
lo omite, será undefined.
```

array.forEach(function(), thisArg)

La función, que recibe como argumento el método forEach, utiliza tres parámetros opcionales:

```
El valor actual del elemento iterado. Es decir, si es la primera iteración,
será el primer elemento, y así sucesivamente.
El índice del elemento iterado. Es decir, si es la primera iteración, será el
índice 0, y así sucesivamente.
El array que está iterando.
```

array.forEach(function(element, index, array))

Practiquemos el uso del método forEach

Utilicemos el array articulos que definimos para mostrar todos los artículos.

Entonces utilizamos el método forEach y que ejecute la función console.log para cada uno de los elementos. Recuerda que el primer parámetro de la función callback es cada uno de los elementos del array.

```
articulos.forEach(function (articulo) {
console.log(articulo)
})
/*
{ nombre: 'Bici', costo: 3000 }
{ nombre: 'TV', costo: 2500 }
...
{ nombre: 'Audifonos', costo: 1700 }
*/
```

Cómo utilizar el método some

El método somees inmutable y consiste retornar un valor lógico verdadero si existe al menos un elemento que cumpla la condición establecida en la función (callback).

El método some recibe dos argumentos:

```
La función que itera y evalúa cada elemento del array hasta que al menos uno
cumpla con la condición especificada (obligatorio).
Un objeto al que puede hacer referencia el contexto this en la función. Si se
lo omite, será undefined.
```

array.some(function(), thisArg)

La función, que recibe como argumento el método some, utiliza tres parámetros:

```
El valor actual del elemento iterado. Es decir, si es la primera iteración,
será el primer elemento, y así sucesivamente.
El índice del elemento iterado. Es decir, si es la primera iteración, será el
índice 0, y así sucesivamente.
El array que está iterando.
```

array.some(function(element, index, array))

Practiquemos el uso del método some

Utilicemos el array articulos que definimos para saber si existe al menos un artículo con el costo menor o igual que 700.

Entonces utilizamos el método some que retorne la condición que necesitamos. Recuerda que el primer parámetro de la función callback es cada uno de los elementos del array.

```
var existeArticulo = articulos.some(function (articulo) {
return articulo.costo <= 700
})
console.log(existeArticulo) // true
```

Próximos pasos

Existen más métodos mutables e inmutables de arrays. Si quieres aprender y profundizar sobre este tema, que es fundamental como desarrollador de JavaScript, toma el Curso de Manipulación de Arrays en JavaScript.