Existen métodos de arrays para recorrerlos, y devolver un valor o un array con nuevos resultados. Entre estos están dos muy importantes: map y filter.

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 filter

El método filter consiste en crear un nuevo array a partir de los elementos originales filtrados mediante una función (callback) que indica la condición a cumplir y es inmutable. Si la condición se cumple, retorna el elemento completo.

El método filter recibe dos argumentos:

```
La función que itera y evalúa si cada elemento del array si cumple con la
condición especificada (obligatorio).
Un objeto al que puede hacer referencia el contexto this en la función. Si se
lo omite, será undefined. Recuerde que this es diferente según el lugar donde
sea invocado.
```

var otherArray = array.filter(function(), thisArg)

La función, que recibe como argumento el método filter, 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.
```

const other = array.filter(function(element, index, array))

Practiquemos el uso del método filter

Utilicemos el array articulos que definimos para filtrar en un nuevo array los artículos cuyo costo sea menor o igual que 500.

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

```
var articulosFiltrados = articulos.filter(function (articulo) {
return articulo.costo <= 500
})

console.log(articulosFiltrados)
/* [
{ nombre: 'Libro', costo: 320 },
{ nombre: 'Teclado', costo: 500 }
] */
```

Cómo utilizar el método map

El método map es inmutable y consiste en crear un nuevo array a partir de los elementos originales transformados mediante una función (callback).

El método map recibe dos argumentos:

```
La función que itera y transforma cada elemento del array (obligatorio).
Un objeto al que puede hacer referencia el contexto this en la función. Si se
lo omite, será undefined. Recuerde que this es diferente según el lugar donde
sea invocado.
```

```
var otherArray = array.map(function(), thisArg)
```

La función, que recibe como argumento el método map, 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.
```

```
var otherArray = array.map(function(element, index, array))
```

Practiquemos el uso del método map

Utilicemos el array articulos que definimos para crear un nuevo array con el nombre de cada uno de los artículos.

Entonces utilizamos el método map que retorne el nombre de cada artículo. Recuerda que el primer parámetro de la función callback es cada uno de los elementos del array.

```
var nombreArticulos = articulos.map(function (articulo) {
return articulo.nombre
```

```
})
console.log(nombreArticulos)
/*
[ 'Bici', 'TV', 'Libro', 'Celular', 'Laptop', 'Teclado', 'Audifonos' ]
*/
```