Apunte sobre métodos vistos en clase

Fuente

https://developer.mozilla.org/es/docs/Web/JavaScript/Reference

Foreach

El método **`forEach()`** ejecuta la función indicada una vez por cada elemento del array.

Ejemplo

```
const array1 = ['a', 'b', 'c'];

array1.forEach(element => console.log(element));

// expected output: "a"

// expected output: "b"

// expected output: "c"
```

Fuente

https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/Array/forEach

Sort

El método **`sort()`** ordena los elementos de un arreglo (array) *localmente* y devuelve el arreglo ordenado

Ejemplo

```
var items = [
  { name: 'Edward', value: 21 },
  { name: 'Sharpe', value: 37 },
  { name: 'And', value: 45 },
  { name: 'The', value: -12 },
  { name: 'Magnetic', value: 13 },
  { name: 'Zeros', value: 37 }
];
items.sort(function (a, b) {
  if (a.name > b.name) {
    return 1;
  }
  if (a.name < b.name) {
    return -1;
  }
  // a must be equal to b
  return 0;
});
```

Fuente:

https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/Array/sort

Filter

El método **`filter()`** crea un nuevo array con todos los elementos que cumplan la condición implementada por la función dada.

```
var arr = [
  { id: 15 },
  { id: -1 },
  { id: 0 },
  { id: 3 },
  { id: 12.2 },
  { },
  { id: null },
  { id: NaN },
  { id: 'undefined' }
];
var entradasInvalidas = 0;
// Si el elemento tiene un atributo id, y su valor correspondiente es un numero
// Y no es el valor NaN, entonces es una entrada válida
function filtrarPorID(obj) {
  if ('id' in obj && typeof(obj.id) === 'number' && !isNaN(obj.id)) {
    return true;
  } else {
    entradasInvalidas++;
    return false;
  }
}
```

Fuente

https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/Array/filter

Map

El método **`map()`** crea un nuevo array con los resultados de la llamada a la función indicada aplicados a cada uno de sus elementos.

Ejemplo

El siguiente código toma un array de objetos y crea un nuevo array que contiene los nuevos objetos formateados.

```
var kvArray = [{clave:1, valor:10},
               {clave:2, valor:20},
               {clave:3, valor: 30}];


var reformattedArray = kvArray.map(function(obj){
  var rObj = {};
  rObj[obj.clave] = obj.valor;
  return rObj;
});
```

Fuente

https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/Array/map