Listado de arrays en Javascript

Fecha de entrega: 04-11-2016
Autor: Miguel Ángel López Moyano

1. Indica los tres argumentos del método forEach a un array. Demuestra su uso mediante un ejemplo.

Sintaxis:

array.forEach(function(currentValue,index,arr), thisValue)

- currentValue: el valor del elemento actual (obligatorio).
- index: índice del elemento actual (opcional).
- arr: objeto al que pertenece el elemento actual (opcional).
- thisValue: Un valor que se pasará a la función que se utilizará como su valor "this". Si este parámetro está vacío, el valor "undefined" se pasará como su valor "this" (opcional).

Ejemplo:

```
{
    let suma = 0;
    let array = [1,2,3,4];

    let sumar=function (elemento) {
        suma += elemento;
    }

    array.forEach(sumar);
    console.log(suma);
}
10
undefined
```

2. Indica la utilidad del operador in con los arrays. Demuestra su uso mediante un ejemplo.

El operador in nos permite buscar dentro de un array a través del índice.

Ejemplo:

```
array= [9,8,7,6,5];
▶ [9, 8, 7, 6, 5]
for(let i in array){
    console.log(i+" "+array[i]);
}
0 9
1 8
2 7
3 6
4 5
undefined
```

3. Indica la función que comprueba si un objeto es o no un Array. Demuestra su uso mediante un ejemplo.

Array.isArray()

Ejemplo:

```
> let array=[1,2,3];
< undefined
> let cadena="hola";
< undefined
> Array.isArray(array);
< true
> Array.isArray(cadena);
< false
```

4. Crea una función que cree un array de la dimensión indicada, cuyo contenido sean los números naturales comenzando desde 0

```
{
    let array;

    let crearArray=function(numero){
        array=[numero];
        for(let i=0;i<=numero;i++){
            array[i]=i;
        }
    }

    crearArray(10);
    console.log(array);
}
▶ [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

5. Crea una función que devuelva un array con cada uno de los argumentos.

```
{
    let devuelveArrayArgumentos=function(){
        let array=[];
        for (i=0;i< arguments.length;i++)
            array.push(arguments[i]);

        return array;
    }

    let array=devuelveArrayArgumentos("hola","que","ase");
    console.log(array);
}
▶ ["hola", "que", "ase"]
undefined
```

6. Crea una función que devuelva un array con cada uno de los argumentos. En caso de que alguno de sus argumentos sea un array, que introduzca sus elementos uno a uno.

```
{
    let devuelveArrayArgumentos=function(){
        let array=[];
        for (i=0;i< arguments.length;i++){
            if(arguments[i] instanceof Array){
                for(j=0;j<arguments[i].length;j++){
                    array.push((arguments[i])[j]);
                }
            }
            else
                array.push(arguments[i]);
        }

        return array;
    }

    let array=devuelveArrayArgumentos("hola",[1,2,3],"que","ase");
    console.log(array);
}
▶ ["hola", 1, 2, 3, "que", "ase"]                                    V
```

7. Crea una función que elimine todos los undefined de un array.

```
{
    let eliminaUndefined=function(array){
        for (i=0;i< array.length;i++){
            if(array[i] == undefined)
                array.splice(i,1);
        }
        return array;
    }

    let array=eliminaUndefined(["hola",undefined,"que",undefined,"ase"]);
    console.log(array);
}
▶ ["hola", "que", "ase"]                                    VM8513:11
undefined
```

8. Indica la diferencia entre los siguientes métodos, y demuestra su uso con algunos arrays: Array.prototype.forEach(), Array.prototype.every(), Array.prototype.some() y Array.prototype.filter()

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

```
{
    let suma = 0;
    let array = [1,2,3,4];

    let sumar=function (elemento) {
        suma += elemento;
    }

    array.forEach(sumar);
    console.log(suma);
}
10
undefined
```

El método every() verifica si todos los elementos del array pasan la prueba implementada por la función dada.

```
{
    let menoresDe10=function(elemento,indice,array){
        return elemento<10;
    }

    console.log([1,2,3,4].every(menoresDe10));
}
true
```

El método some() verifica si algún elemento de un array cumple con el test implementado por la función brindada.

```
{
    let existeUnMenorDe10=function(elemento,indice,array){
        return elemento<10;
    }

    console.log([111,222,333,4].some(existeUnMenorDe10));
}
true
```

El metodo filter() crea un nuevo array con todos los elementos que pasen la prueba implementada por la función dada.

```
{
    let nuevoArrayConMenoresDe10=function(elemento,indice,array){
        return elemento<10;
    }

    console.log([111,222,333,4,5,666].filter(nuevoArrayConMenoresDe10));
}
▶ [4, 5]                                                    VM12155:6
undefined
```

9. Averigua qué método es el más eficiente para manejarse con arrays. Compruébalo mediante performance.now() o similares

Introduce 10 elementos en un array mediante push(), unshift(), directamente, fijando tamaño en new Array...

```
{
    let array=[];

    let eficienciaPush=function(){
        let t0 = performance.now();
        array.push(1,2,3,4,5,6,7,8,9,10);
        let t1 =performance.now();
        console.log("Eficiencia de push: "+(t1 - t0));
    }

    eficienciaPush();
}
Eficiencia de push: 0.004999999888241291
undefined
```

```
{
    let array=[];

    let eficienciaUnshift=function(){
        let t0=performance.now();
        array.unshift(1,2,3,4,5,6,7,8,9,10);
        let t1=performance.now();
        console.log("Eficiencia de unshift: "+(t1-t0));
    }

    eficienciaUnshift();
}
```

```
Eficiencia de unshift: 0.010000000000218279
undefined
```

```
{
    let eficienciaNew=function(){
        let t0=performance.now();
        let array= new Array(1,2,3,4,5,6,7,8,9,10);
        let t1=performance.now();
        console.log("Eficiencia de new: "+(t1-t0));
    }

    eficienciaNew();
}
```

```
Eficiencia de new: 0.005000000004656613
undefined
```

Eliminar 10 elementos en un array mediante pop(), shift(), directamente, fijando tamaño...

```
{
    let array=[1,2,3,4,5,6,7,8,9,10];

    let eficienciaPop=function(){
        let t0=performance.now();
        for(let i=0;i<10;i++){
            array.pop();
        }
        let t1=performance.now();
        console.log("Eficiencia de pop: "+(t1-t0));
    }

    eficienciaPop();
}
```

```
Eficiencia de pop: 0.014999999897554517
undefined
```

```
{
    let array=[1,2,3,4,5,6,7,8,9,10];

    let eficienciaShift=function(){
        let t0=performance.now();
        for(let i=0;i<10;i++){
            array.shift();
        }
        let t1=performance.now();
        console.log("Eficiencia de shift: "+(t1-t0));
    }

    eficienciaShift();
}
```

```
Eficiencia de shift: 0.014999999897554517
```

```
{
    let array=[1,2,3,4,5,6,7,8,9,10];

    let eficienciaSplice=function(){
        let t0=performance.now();
        for(let i=0;i<10;i++){
            array.splice(0,9);
        }
        let t1=performance.now();
        console.log("Eficiencia de splice: "+(t1-t0));
    }

    eficienciaSplice();
}
```

```
Eficiencia de splice: 0.02500000037252903
```