Xaxiry Magaly Gonzalez Ramos

```
JS script.js > [@] arreglosNumeros
1   // Xaxiry Magaly Gonzzalez Ramos   30/10/24
2   //Arreglos (Arrays), matrices o colecciones.
3   // Los arrays son estructuras utilizadas para almacenar y organizar datos.
4   // Los arreglos en JavaScript se pueden declarar utilizando let, const o var, pero cada uno tiene un
    comportamiento diferente.
5   /*Usa let si necesitas cambiar el arreglo completamente.
6   Usa const si quieres que el arreglo no cambie de referencia, pero aún puedes modificar su contenido.
7   Var no se recomienda en el contexto moderno de JavaScript. */
8   // ***** Creando,modificando, accediendo, arreglos en JS *****
9   // creando arreglos en JS. JavaScript Array literal. Estrcutura: let array_name = [item1, item2, ...];
10  let taquitos=[ "pastor ", "canasta ", "dorados ", "barbacoa ", "carnitas ", "guisado ", "carne asada ",
    "suadero "];
11  console.log(taquitos);
12  // Un arreglo puede usar varias líneas, el espacio y los saltos de línea no interfieren, mientras se respete
    la estructura.
13  let arreglosNumeros = [
14    1 ,
15    2 ,
16    3 ,
17    4 ,
18    5 ,
19    6 ,
20    7
21  ];
22  console.log(arreglosNumeros);
23  // Usando la palabra clave new.
24  let arreglosLogicos = new Array ( true,  false ,true );
25  console.log(arreglosLogicos);
26  // *Se recomienda utilizar la primera estructura JavaScript Array literal.*
```

```
▼ Array(8) i                                                    script.js:11
    0: "pastor "
    1: "canasta "
    2: "dorados "
    3: "barbacoa "
    4: "carnitas "
    5: "guisado "
    6: "carne asada "
    7: "suadero "
    length: 8
  ▶ [[Prototype]]: Array(0)
▼ Array(7) i                                                    script.js:22
    0: 1
    1: 2
    2: 3
    3: 4
    4: 5
    5: 6
    6: 7
    length: 7
  ▶ [[Prototype]]: Array(0)
▼ Array(3) i                                                    script.js:25
    0: true
    1: false
    2: true
    length: 3
  ▶ [[Prototype]]: Array(0)
```

Xaxiry Magaly Gonzalez Ramos

```
27  // modificando Array taquitos.
28  //Cambiando un elemento del arreglo.
29  taquitos[5] = "alambre"; // Cambiando guisado por alambre
```

```
▼ Array(8) i                                                    script.js:11
    0: "pastor "
    1: "canasta "
    2: "dorados "
    3: "barbacoa "
    4: "carnitas "
    5: "alambre"
    6: "carne asada "
    7: "suadero "
    length: 8
  ▶ [[Prototype]]: Array(0)
```

```
31  // Ejemplos y para que sirve push y pop en los arreglos
32  // Los métodos .push() y .pop() actuan al final del array.
33  // .push() Añade uno o varios elementos al final del array.
34  let agua = ["Jamaica ", "horchata ", "Naranja ", "Limon "];
35  document.getElementById("agua1").innerHTML = agua; //Para visualizar en html
36  agua.push("Uva "); //Agrega Uva al final de la lista
37  // let length = agua.push("Uva"); // Para añadir uno o varios elementos al final del array. Devuelve el tamaño
    del array.
38  document.getElementById("agua2").innerHTML = agua ; // Para visualizar en html en parrafo
39  //.pop()  elimina el último elemento del array.
40  let calcetin = ["Rojos ", "Verdes ", "Naranjas ", "Amarillo "];
41  document.getElementById("calcetines1").innerHTML = calcetin; //Para visualizar en html
42  calcetin.pop(); // elimina el ultimo elemento del array
43  //document.getElementById("calcetines2").innerHTML = calcetin.pop();//Devuelve dicho elemento, eliminado.
44  document.getElementById("calcetines2").innerHTML = calcetin ; // Para visualizar en html en parrafo
45  //Los métodos .unshift() y .shift() actuan al inicio del array.
46  // .unshift Añade uno o varios elementos al inicio del array.
47  document.getElementById("agua3").innerHTML = agua; //Para visualizar en html
48  agua.unshift("Piña "); //Agrega Uva al inicio de la lista
49  document.getElementById("agua4").innerHTML = agua ; // Para visualizar en html en parrafo
50  // .shift() Elimina el primer elemento del array. Devuelve dicho elemento.
51  document.getElementById("calcetines3").innerHTML = calcetin; //Para visualizar en html
52  calcetin.shift(); // elimina un elemento al inicio del array
53  document.getElementById("calcetines4").innerHTML = calcetin ; // Para visualizar en html en parrafo
```

Xaxiry Magaly Gonzalez Ramos

Metodo push()

Añade uno o varios elementos al final del array.

Jamaica ,horchata ,Naranja ,Limon

Jamaica ,horchata ,Naranja ,Limon ,Uva

Metodo pop()

elimina el último elemento del array.

Rojos ,Verdes ,Naranjas ,Amarillo

Rojos ,Verdes ,Naranjas

Metodo .unshift ()

Añade uno o varios elementos al inicio del array.

Jamaica ,horchata ,Naranja ,Limon ,Uva

Piña ,Jamaica ,horchata ,Naranja ,Limon ,Uva

Metodo .shift()

Elimina el primer elemento del array.

Rojos ,Verdes ,Naranjas

Verdes ,Naranjas

```
55  // ----- accediendo a los arreglos -----
56  let animales =["conejo ", "perro ", "caballo ", "gato ", "loro "];
57  document.getElementById("arregloAni").innerHTML = animales; //console.log(animales);
58  document.getElementById("unElemento").innerHTML = animales[1]; // console.log(animales[1]); // accedemos a
    perro.
```

Accediendo al array

Al arreglo completo

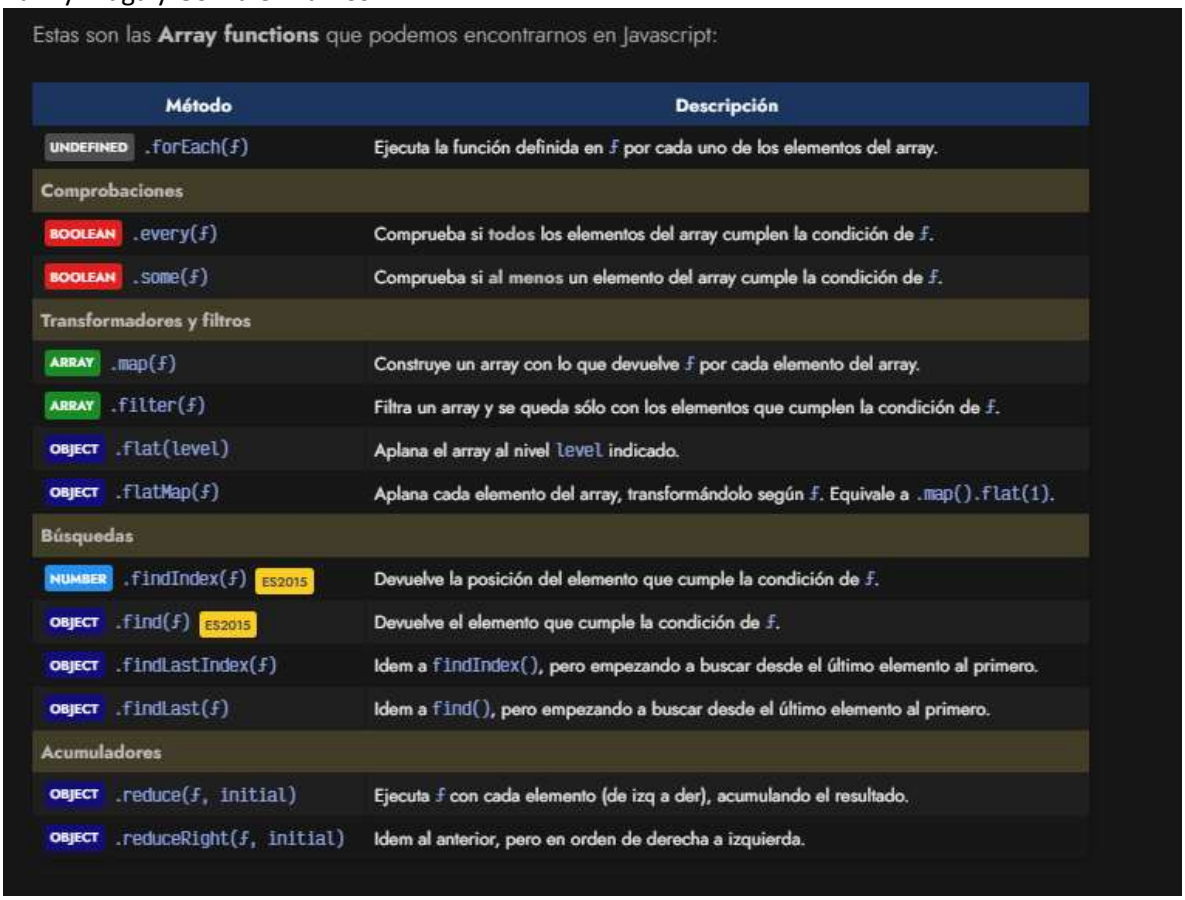
conejo ,perro ,caballo ,gato ,loro

A un elemento

perro

Funciones de arreglos más usadas en desarrollo JS

Xaxiry Magaly Gonzalez Ramos



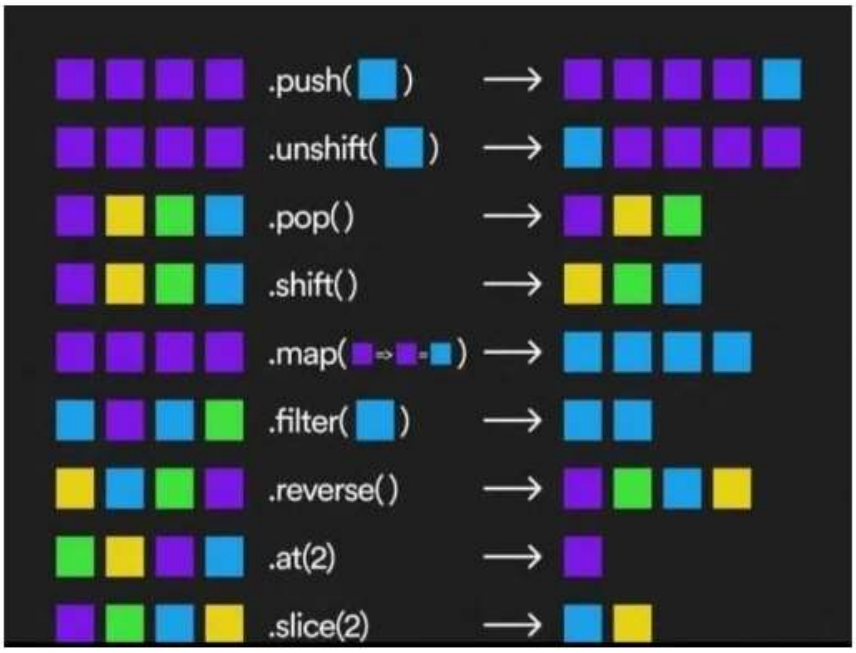
Estas son las **Array functions** que podemos encontrarnos en Javascript:

Método	Descripción
UNDEFINED `.forEach(f)`	Ejecuta la función definida en `f` por cada uno de los elementos del array.
Comprobaciones	
BOOLEAN `.every(f)`	Comprueba si todos los elementos del array cumplen la condición de `f`.
BOOLEAN `.some(f)`	Comprueba si al menos un elemento del array cumple la condición de `f`.
Transformadores y filtros	
ARRAY `.map(f)`	Construye un array con lo que devuelve `f` por cada elemento del array.
ARRAY `.filter(f)`	Filtra un array y se queda sólo con los elementos que cumplen la condición de `f`.
OBJECT `.flat(level)`	Aplana el array al nivel `level` indicado.
OBJECT `.flatMap(f)`	Aplana cada elemento del array, transformándolo según `f`. Equivale a `.map().flat(1)`.
Búsquedas	
NUMBER `.findIndex(f)` ES2015	Devuelve la posición del elemento que cumple la condición de `f`.
OBJECT `.find(f)` ES2015	Devuelve el elemento que cumple la condición de `f`.
OBJECT `.findLastIndex(f)`	Idem a `findIndex()`, pero empezando a buscar desde el último elemento al primero.
OBJECT `.findLast(f)`	Idem a `find()`, pero empezando a buscar desde el último elemento al primero.
Acumuladores	
OBJECT `.reduce(f, initial)`	Ejecuta `f` con cada elemento (de izq a der), acumulando el resultado.
OBJECT `.reduceRight(f, initial)`	Idem al anterior, pero en orden de derecha a izquierda.

https://lenguajejs.com/javascript/arrays/array-functions/

Xaxiry Magaly Gonzalez Ramos

A continuación se muestran en la siguiente imagen los métodos más comunes para recorrer un array:



https://configuroweb.com/metodos-mas-comunes-para-recorrer-un-array/

```
// Funciones de arreglos mas usadas en desarrollo js
// forEach, slice y shift
// .forEach(f): Ejecuta la función definida en f por cada uno de los elementos del array.
// Ejemplo de .forEach
let dulces = ["paletas ", "chicles ", "chocolate ", "caramelo ", "palanquetas "];
// .forEach con funcion arrow https://www.youtube.com/watch?v=SDaZXXbHI_U
dulces.forEach(dulces => {
console.log(dulces); // Imprime cada fruta, recorre el array.
 });
 // Ejemplo de slice
/*Es una función integrada en JavaScript que se utiliza para crear una copia superficial de una parte de un array existente. La parte del array
que se copia se especifica mediante el índice inicial y final. Este método no modifica el array original y devuelve un nuevo array.*/
let slicedulces = dulces.slice(1, 3);
console.log(slicedulces); // [2, 3]
 // Ejemplo de shift
 /*Es una función integrada en JavaScript que se utiliza para eliminar el primer elemento de un array y devolver ese elemento eliminado. Este
 método modifica el array original en el que se llama. */
let numbers = [1, 2, 3, 4, 5];
let firstNumber = numbers.shift();
console.log(numbers); // [2, 3, 4, 5]
console.log(firstNumber); // 1
```

Xaxiry Magaly Gonzalez Ramos

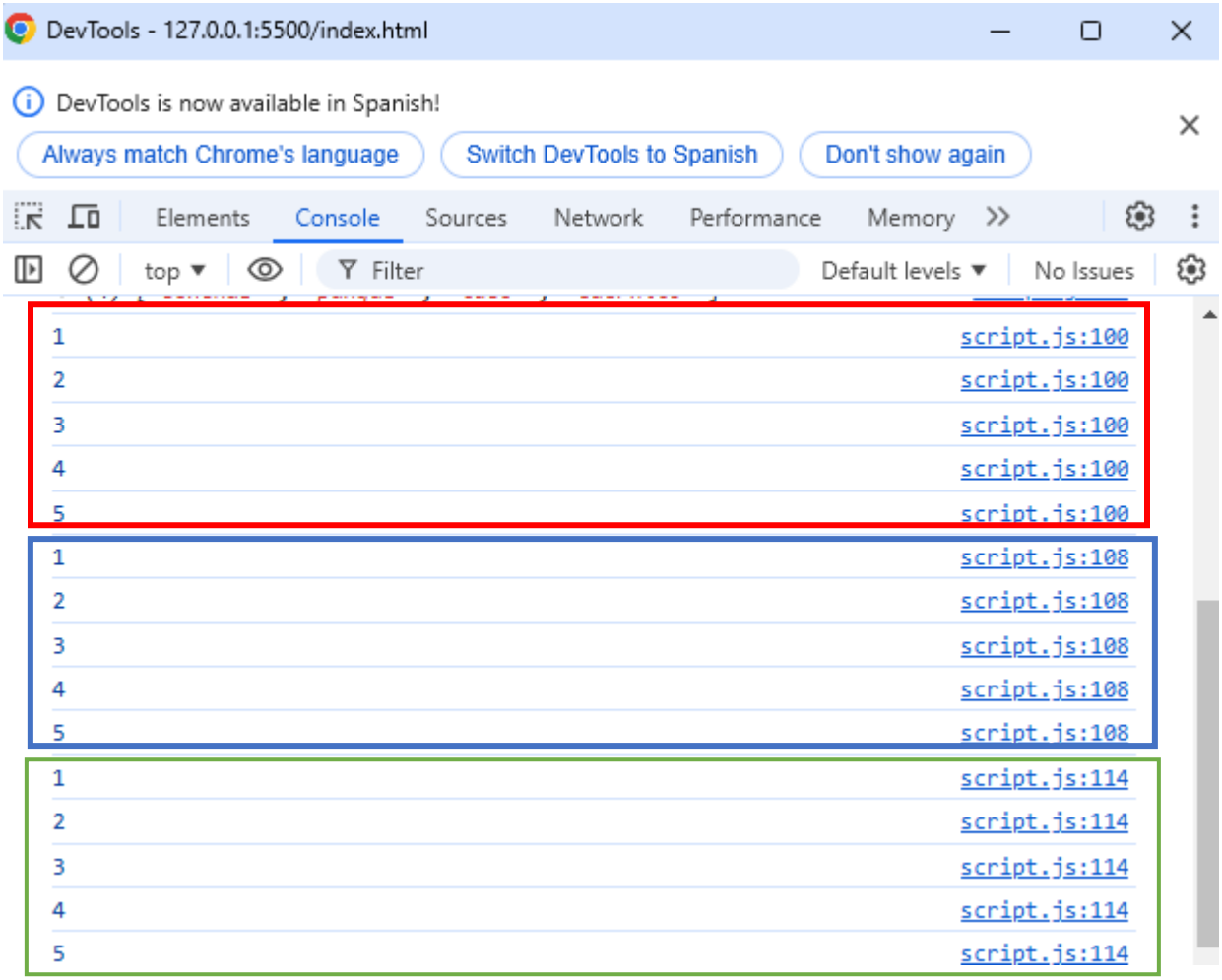
```
paletas                                                    script.js:67
chicles                                                    script.js:67
chocolate                                                  script.js:67
caramelo                                                   script.js:67
palanquetas                                                script.js:67
▼ Array(2) i                                               script.js:72
    0: "chicles "
    1: "chocolate "
    length: 2
  ▶ [[Prototype]]: Array(0)
▼ Array(4) i                                               script.js:77
    0: 2
    1: 3
    2: 4
    3: 5
    length: 4
  ▶ [[Prototype]]: Array(0)
1                                                          script.js:78
```

```
80  // Ejemplo y paso a paso de como crear una cola o pila con arreglos JS
81  // Cola FIFO (First In, First Out), lo que significa que el primer elemento en entrar es el primero en salir,
    y los nuevos que se añaden van al final.
82  //Primero se declara la matriz
83  let cola = ["donas ", "conchas ", "panque "]
84  console.log(cola);
85  // luego se agregan mas elementos al arreglo con push.
86  cola.push("taco ");
87  cola.push("cuernito "); // Utilizamos push porque añade uno o varios elementos al final del array.
88  // Para eliminar la primera en la cola se utiliza shift
89  cola.shift();
90  console.log(cola);
91  // Explica For, Do while, while y cuando usar uno en cada caso
92  // 5 ejemplos de For anidados
```

```
▶ (3) ['donas ', 'conchas ', 'panque ']                    script.js:84
▶ (4) ['conchas ', 'panque ', 'taco ', 'cuernito ']        script.js:90
```

Xaxiry Magaly Gonzalez Ramos

```
JS script.js > ...
92  // Explica For, Do while, while y cuando usar uno en cada caso
93  // Do - While
94  //Es una combinación de instrucciones define un bloque de código que se ejecutará una vez y se repetirá
    mientras se cumpla una condición true.
95  /* Estructura
96  do {  instrucción de codigo
97  } while (condition); */
98  let contar = 1;
99  do {
100     console.log(contar);
101     contar++;
102 } while (contar <= 5); // Para contar del 1 al 5
103 // While
104 // La while de claración crea un bucle (alrededor de un bloque de código) que se ejecuta mientras se cumple
    una condición true.
105 let contarWhile = 1;
106
107 while (contarWhile <= 5) {
108     console.log(contarWhile);
109     contarWhile++;
110 }
111 //For
112 //Se usa cuando conoces el número de iteraciones.
113 for (let cont = 1; cont <= 5; cont++) {
114   console.log(cont);
115 }
```



Xaxiry Magaly Gonzalez Ramos

```
117  // Ejemplo de For anidados
118  const filas = 5;
119  // Para hacer una piramide triangulo.
120  for (let i = 1; i <= filas; i++) { //Este bucle se ejecuta desde i = 1 hasta i = 5
121      let linea = ""; // Se inicializa una cadena vacía llamada linea. Esta cadena se irá llenando con
     asteriscos en cada iteración del bucle interior.
122      for (let j = 1; j <= i; j++) { // Este bucle anidado se ejecuta desde j = 1 hasta j = i. Esto significa
     que en la primera iteración (cuando i = 1), el bucle interior se ejecutará una vez; en la segunda
     iteración (cuando i = 2), se ejecutará dos veces, y así sucesivamente hasta que i = 5.
123          linea += '* '; // En cada iteración del bucle interior, se añade un asterisco seguido de un espacio a
     la cadena linea. Esto construye la línea de asteriscos para esa fila.
124      }// se cierra el for anidado.
125      console.log(linea); // imprimer en consola
126  } //Se cierra el primero for
```

```
*                                                    script.js:125
* *                                                  script.js:125
* * *                                                script.js:125
* * * *                                              script.js:125
* * * * *                                            script.js:125
```