**Array functions**

const fruits = ["Banana", "Orange", "Apple", "Mango"];

1. fruits.length; The length property **returns the length** (size) of an array

fruits.length = 2; ex : Banana,Orange

1. toString() method **returns the elements of an array as a comma separated string**. Ex: fruits.toString();
2. fruits.at(2); fruits[2]  element of fruits using at():
3. The join() method also **joins all array elements into a string.  separator**: fruits.join(" \* ");
4. The pop() **method removes the last element from an array**: fruits.pop();
5. The push() method **adds a new element to an array (at the end):** fruits.push("Kiwi");
6. The shift() method **removes the first array element and "shifts" all other elements to a lower index**. fruits.shift();
7. The unshift() method adds a **new element to an array (at the beginning), and "unshifts" older elements**: fruits.unshift("Lemon");
8. Using delete() leaves undefined holes in the array.
9. delete fruits[0]; , show undefined
10. The concat() method creates a new array by merging (concatenating) existing arrays:
11. Array1.concat(array2);
12. **console.log(names.concat([1,2,3]))**

for

**For …in => get the array index**

**For …of => get the array elements**

**let len =  names.length**

**//  console.log(len)**

**// iteration**

**for(let i=3;i<len;i++){**

**document.writeln(names[i],"<br>")**

**}**

**/\***

**for(let a  in array){**

**}**

**\*/**

**for(let i  in names){**

**document.writeln(names[i])**

**}**

**/\***

**for(let e of array){**

**}**

**\*/**

**for(let ele of names){**

**console.log(ele)**

**}**

 let num = [10,20,30,40,50];

1. **sum of array**
2. **count the element**
3. let studentName = "Denish@12345";
4. console.log(studentName[0])
5. console.log(studentName[1])
6. console.log(studentName[2])
7. console.log(studentName[3])
8. console.log(studentName[4])
9. //  a-z count the letters  6
10. //  0-9 count the numbers  5
11. //   count special charactors in 1
12. //  a,e,i,o,u
13. //  count the vowels 2

----------------------------------------------------------------------

1.The splice() method adds new items and also remove and update to an array.

2.The slice() method slices out a piece of an array.

Must add the two parameters:

const fruits = ["Banana", "Orange", "Apple", "Mango"];

fruits.splice(0,1); //

fruites.splice(1,2);//

fruites.splice(1,0,”Sweet”,”jaggery”) // Banana,Sweet,Jaggery,Orange,Apple, Mango

fruites.splice(1,2,”Sweet”,”jaggery”) // Banana,Sweet,Jaggery,Mango

===================================================================================================

The method then selects elements from the start argument, and up to (but not including) the end argument.

const citrus = fruits.slice(1); / /Orange,Lemon,Apple,Mango

const citrus = fruits.slice(1,3); // Orange,Lemon

const fruits = ["Apple", "Orange", "Apple", "Mango"];

numbers = [3,5,7,8,10,20,60,23]

const numbers.find((value,index,array)=>{return value>18})

|  |  |  |  |
| --- | --- | --- | --- |
| [Array indexOf()](https://www.w3schools.com/js/js_array_search.asp#mark_indexof) | Returns the **first position** of an element value | Ex: const index = fruits.indexOf("Apple"); => console.log(index); // 0 |  |
| [Array lastIndexOf()](https://www.w3schools.com/js/js_array_search.asp#mark_lastindexof) | Returns the **last position** of an element value | Ex: const index = fruits.lastIndexOf("Apple"); => console.log(index); // 2 |  |
| [Array includes()](https://www.w3schools.com/js/js_array_search.asp#mark_includes) | Returns **true** if an element value is present in an array | Ex: fruits.includes("Mango"); // true or false |  |
| [Array find()](https://www.w3schools.com/js/js_array_search.asp#mark_find) | Returns the value of the **first element** that passes a test | Ex: let first = numbers.find((value, index, array)=>{return value > 18; }); // element ex 30 |  |
| [Array findIndex()](https://www.w3schools.com/js/js_array_search.asp#mark_findindex) | Returns **the index of the first element** that passes a test | Ex: let first = numbers. findIndex((value, index, array)=>{return value > 18; }); index ex: 0 |  |
| [Array findLast()](https://www.w3schools.com/js/js_array_search.asp#mark_findlast) | Returns the value of the **last element that passes** a test | const temp = [27, 28, 30, 40, 42, 35, 30];  let high = temp.findLast(x => x > 20); // ex : 30 |  |
| [Array findLastIndex()](https://www.w3schools.com/js/js_array_search.asp#mark_findlastindex) | Returns the **index of the last element** that passes a test | const temp = [27, 28, 30, 40, 42, 35, 30]; let pos = temp.findLastIndex(x => x > 40); // index : 4 |  |

## Array Sort Methods

const fruits = ["Banana", "Orange", "Apple", "Mango"];

sort() method sorts an array alphabetically: only string

fruits.sort();

The reverse() method reverses the elements in an array:

fruits.reverse();

toSort(); toReversed();

Number sort

points.sort(function(a, b){return a - b});

array.forEach(()=>{

})

**forEach()**  
Executes a callback on each array element and returns undefined. It’s ideal for **side effects** (like logging, updating variables, DOM actions). Do not use it if you want to produce a new array

**filter()**

Returns a new array with only those items that satisfy a given condition. Keeps or removes each element based on true/false result

**map()**

each element and returns a **new array** of the same length. Use it when you want to transform values. The original array is not mutated.

**reduce()**

Processes all elements to produce a **single output value**, using an accumulator that carries across iterations. **Useful for summarizing arrays into sums, counts, or aggregated objects.**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <!--

 // foreach

     array.foreach((element,index,array)=>{

     })

    es 6

    ecma script 6

     element=> mandatory

     index,array => optional

    map

    ====

    it will return the new array without change

    the array length

    array.map(()=>{

    })

    reduce

    =======

    it will return only one value

    array.reduce((star,iterate)=>{

    })

    let sum =0

    -->

    <script>

let num =[50,40,50,60,80,1,4,6,7,8,400,500];

const finalPriceList =  num.filter((val)=>{

    return val <=50

  })

  console.log(finalPriceList)

//   {

//     "productName":"soap",

//     "price":300

//     "desc":"sdfsd",

//     "dis":10

//   }

//   objects

//  let final= num.reduce((sum,iterate)=>{

//     // console.log("sum: ", sum , "iterate:", iterate, "total :", sum+iterate)

//     return sum-iterate;

// })

// -50-40 => -90

// console.log(final)

        // let num = [1,2,3,5,6];

        // let names = ['a','b','c','d','f']

    //   const finalnum = num.map((el)=>{

    //         return el\*2;

    //     })

    //  const fn  = num.map((e)=>{

    //             if(e>2){

    //                 return 1

    //             }

    //             return e\*2;

            // if(e == 'a'){

            //     return '';

            // }else if(e == 'b'){

            //      return 'denis';

            // }else{

            // }

        // })

        // console.log(fn)

    // document.writeln(finalnum)

    // console.log(finalnum)

//     let names = ['surendhar','Denish','arula','kabil','Dina','Deepa'];

// //    document.writeln(names)

// let n =1;

//    names.forEach((ele)=>{

//     document.writeln(n,ele,"<br>")

//     n++

//    })

    </script>

</body>

</html>