

Array, Set and Map

By CODEMIND Technology

Contact us 966 5044 698
966 5044 598

JS Collections

- Array
 - Set
 - Map
-

Array in JavaScript

- Array is reference or non primitive data type which can store multiple values
- Array allows to duplicate elements
- Each value in an array has a numeric position, known as its index and it starts from 0
- Array can contain data of any data type-numbers, strings, booleans, functions, objects, and even other arrays

Syntax:

```
var array_name = [ element1, element2, element3, elementN ];
```

Or

```
var array_name = new Array[ element1, element2, element3, elementN ];
```

Note: It is a common practice to declare arrays with the const keyword.

What is typeof Array? What is reindexing?

Accessing and Updating array element

Accessing array element

Syntax:


```
array_name[index_value];
```

Changing or Updating an array element

```
array_name[index_value] = "new value";
```

includes() method

It check whether the array contains the given value or not



```
const array = [1, 2, 3, 4, 5];  
  
array.includes(3); // output: true  
array.includes(9); // output: false
```

indexOf() method

The `indexOf()` method returns the index of the first occurrence of the substring or element that we specify as the argument



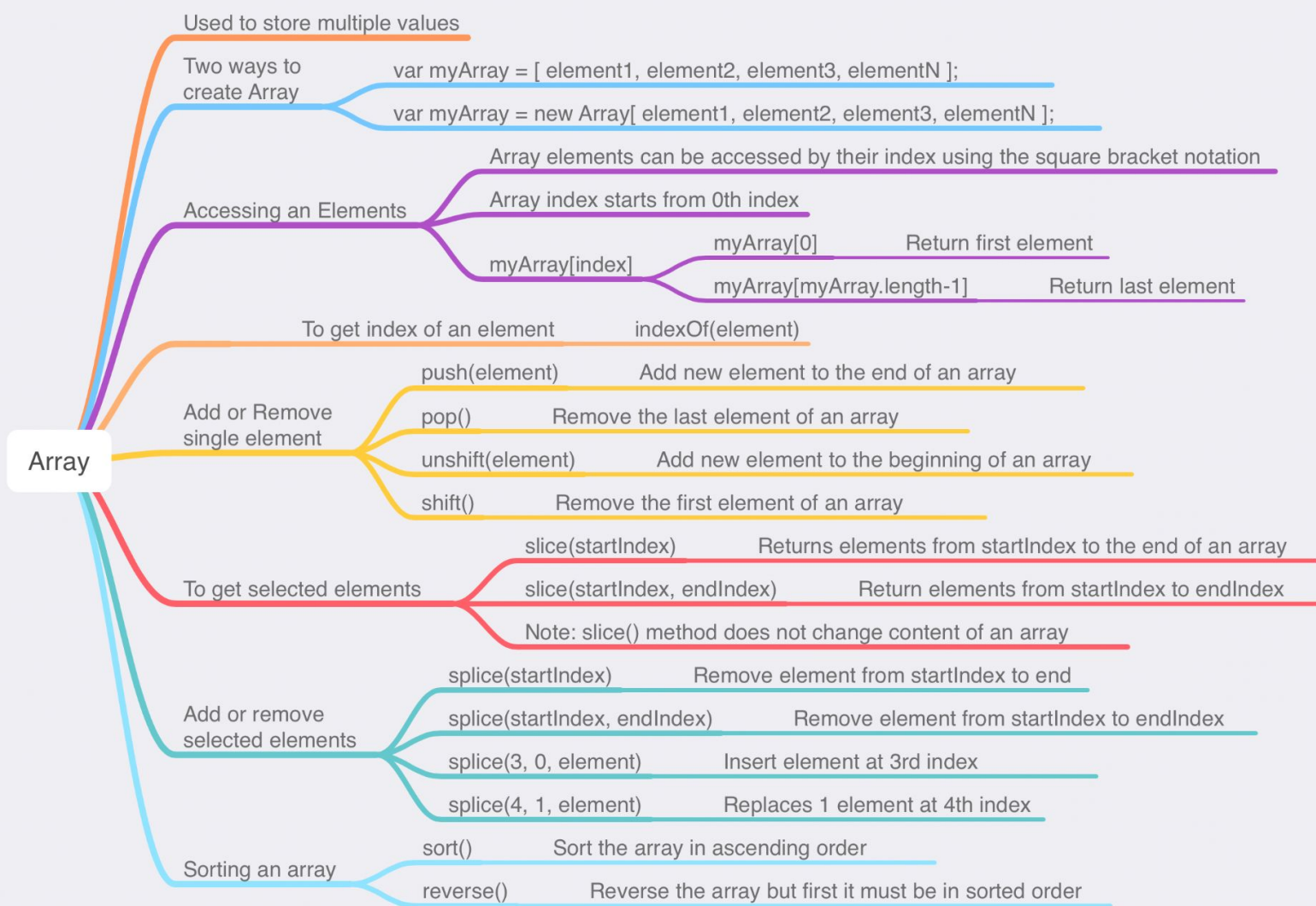
```
const numbers = [1, 3, 4, 6, 10];  
const indexOf10 = numbers.indexOf(10); // 4  
const indexOf16 = numbers.indexOf(16); // -1
```

IndexOf()

Traverse the array in reverse order ?

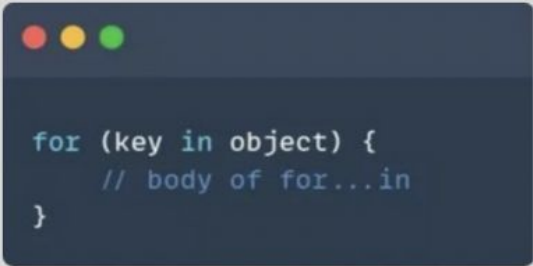
```
const array = [4, 6, 7, 8, 3, 2];  
for (let index = (array.length-1); index >= 0; index-- ){  
    const element = array[index];  
    console.log(element);  
}
```

WAP to find the even positioned value ?



for in loop

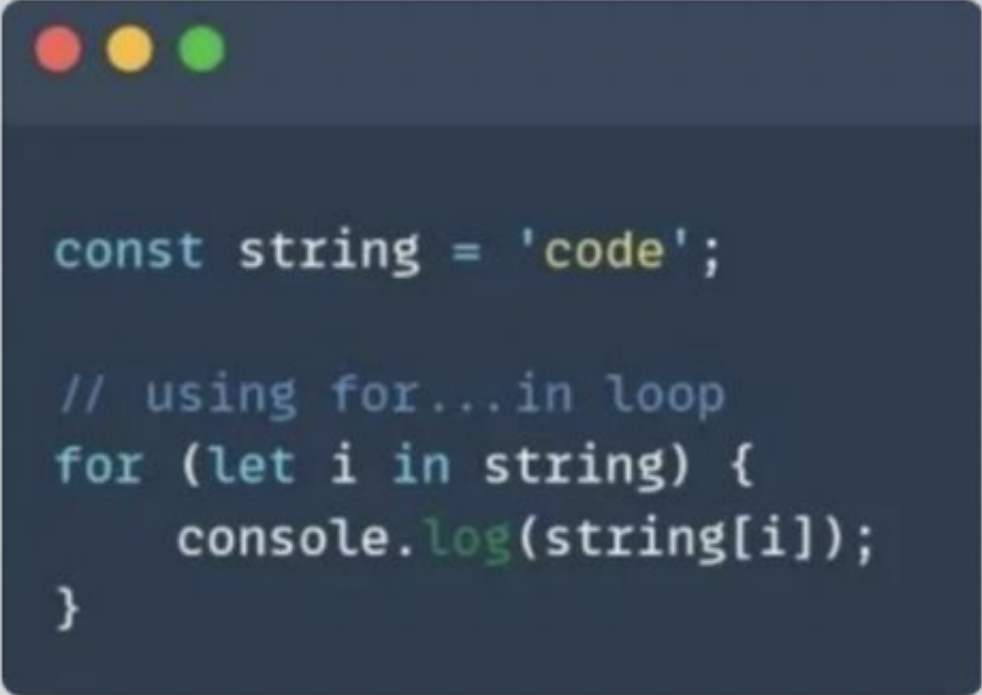
- The for...in loop in JavaScript allows you to iterate over all property keys of an object.
- The syntax of the for...in loop is:

A dark-themed code editor window with three colored window control buttons (red, yellow, green) in the top-left corner. It contains JavaScript code for a for...in loop.

```
for (key in object) {  
    // body of for...in  
}
```

- In each iteration of the loop, a key is assigned to the key variable.

- for...in loop to iterate over string values.



```
const string = 'code';  
  
// using for...in loop  
for (let i in string) {  
    console.log(string[i]);  
}
```

for loop: To traverse over an array using traditional for loop

```
// Traversing an array
const fruits = ["Apple", "Mango", "Orange", "Strawberry", "Grapes", "Mango", "Orange"];
console.log("----- Total Elements in the array - fruits -----")
for (let index = 0; index < fruits.length; index++) {
    const element = fruits[index];
    console.log(`${element}`);
}
```

for in loop: To traverse over an array

```
const fruits = ["Apple", "Mango", "Orange", "Strawberry", "Grapes"];
for (const element in fruits) {
    console.log(fruits[element]);
}
```

What is iterable

Iterable is an object which can be looped over or iterated over with the help of a for loop.

String, Array, Set, Map these are all built-in iterables, because each of their prototype objects implements an `@iterator` method.

for of loop

- The for...of loop in JavaScript allows you to iterate over iterable objects.
- The syntax of the for...of loop is:

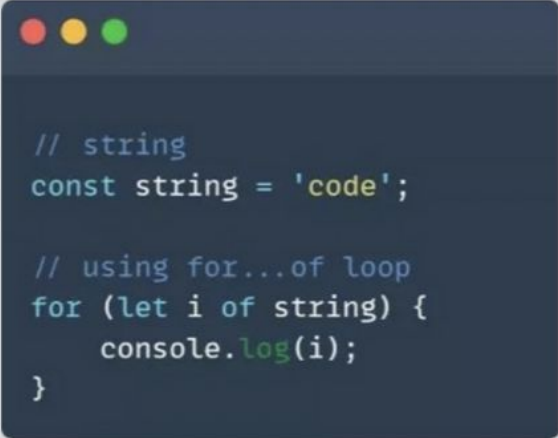
```
for (element of iterable) {  
  // body of for...of  
}
```

- iterable - an iterable object (array, set, strings, etc).
- element - items in the iterable.

For of loop to traverse over an array

```
let fruits = ["Apple", "Mango", "Orange", "Strawberry", "Grapes"];  
for (const element of fruits) {  
  console.log(element);  
}
```


- for...of loop to iterate over string values.



```
// string  
const string = 'code';  
  
// using for...of loop  
for (let i of string) {  
  console.log(i);  
}
```

join() method

- join() method returns an array as a string & does not change the original array
- The default separator is a comma (,) separator, We can also specify any separator



```
const words = ["follow", "for", "more"];
```

```
console.log(words.join());
```

```
// Output - follow,for,more
```

```
console.log(words.join(" "));
```

```
// Output - follow for more
```

concat() method

This method is used for concatenation

1. This method concat two strings
2. This method can be used to concat or merge two arrays

Example:

```
let arr1= [1,2,3];  
let arr2 = [4,5];  
let arr3 = arr1.concat(arr2);  
console.log(arr3); // [ 1, 2, 3, 4, 5 ]
```


Resize an array



JS main.js

```
var entries = [ 1,2,3,4,5,6,7]
console.log(entries.length)
// 7
entries.length = 4
```

```
console.log(entries.length)
// 4
console.log(entries)
```

Output:

```
[1, 2, 3, 4]
```

How to add Elements in an Array

- At the time of array creation we can add elements
 - `const my_array = [obj1, obj2, obj3, 1, 2, 3]`
- Also we can use `push()` or `splice()` method
 - `const array = [];`
 - `array.push(23);` or `array.splice();`

Array of objects

- How to add objects inside array
- How to access object one by one
- How to traverse array of object using for of loop

How to create the new array object from existing object

```
const array_original = [1, 2, 3, 4, 6, 77, 88 ];  
// let array_copy = array_original; // Shallow copy - Not recommended  
let array_copy = [...array_original];  
console.log(array_original);  
console.log(array_copy);
```

How to merge two array

```
const array1 = [1, 3, 4];  
const array2 = [5,4,6,7];  
// First way to merge array using concat() method  
const array3 = array1.concat(array2);  
console.log(array3);  
// Second way to merge array using ... spread operator  
const array4 = [...array1, ...array2];  
console.log(array4);
```

Spread Operator: ... (3 dots)

The spread operator ... is used to expand an array.

```
const fruits = ["Apple", "Mango", "Orange", "Strawberry", "Grapes"];  
console.log(fruits); // (5) ['Apple', 'Mango', 'Orange', 'Strawberry', 'Grapes']  
console.log(...fruits); // Apple Mango Orange Strawberry Grapes
```

In this code: `console.log(...fruits);`

Is equivalent to: `console.log("Apple", "Mango", "Orange", "Strawberry", "Grapes");`

Array: Performance

- Why push(), pop() methods are faster than shift() and unshift()?
- What is reindexing?

The push() and pop() methods runs faster than unshift() and shift(). Because push() and pop() methods simply add and remove elements at the end of an array therefore the elements do not move, whereas unshift() and shift() add and remove elements at the beginning of the array that require re-indexing of whole array

Assignment 01:

```
const fruits_seasonal = ["Banana", "Orange", "Apple", "Mango", "Water Melon"];
```

For a given array fruits perform below operations as:

1. Log the first and last element on console
2. Add element → Papaya before the element 'Banana' and then log array on console
3. Remove 'Mango' from the array
4. Add element or insert an element 'Pineapple' at the last position
5. Insert an element - 'Dragon Fruit' before "Water Melon"
6. Replace an element 'Orange' with 'Kiwi'
7. Log the elements starting from index 1 to 4
8. Only select last 3 element and log on console: Use the length property

Assignment 02: Please before posting on whatsapp group verify your result once.

```
const array_numbers = [ 20, 31, 40, 25, 23, 11, 29, 9, 60, 2, 11 ];
```

1. Find the total elements available or length and log on console
2. Log the first element and last element in array_numbers and log on console
3. Log the thirst last element using length property and log on console
4. Find the all even numbers and log on console
5. Find the odd numbers and log on console
6. Find all the even positioned elements from array_numbers, for loop
7. Find all the odd positioned elements from array_numbers, log on console
8. Find the sum of all elements from array_numbers, log on console
9. Find the numbers which are multiple of 5
10. Is number 115 available in array_numbers ?
11. Is number 23 available in array_numbers ?

Set: Set is a collection of unique elements or values

How to create Set?

We can use 'new' keyword to create set

```
const set_name = new Set();
```

How to add element ?

We can use add() method:

```
set_name.add(element)
```

What will happen in case we will add duplicate element?

Ans: Will maintain only unique element

How to get the total elements count

We can use size property

How to clear set

clear() method is used to clear the set

Delete an element from Set

delete(element) is used

How to see whether particular element is present or not

has(element)

How to traverse set?

We can use The 'for of loop' to traverse set

Syntax:

```
for(const element of set_name) {  
    console.log(element);  
}
```

Example: Set of objects

Programs:

Program to remove duplicates from array [5, 6, 7, 34, 23, 6, 7, 89]

```
const array_numbers = [1, 2, 3, 4, 6, 1, 88, 3, 2, 6 ];  
const array_new = [...new Set(array_numbers)];  
console.log(array_new);
```

Map: Map is used to store collection of elements in key-value pairs

- Key should be unique
- Map contains the insertion order

How to create Map?

'new' keyword is used to create Map in JS

Syntax:

```
const map_name = new Map();
```

Adding elements in key-value pair format

Syntax: `map_name.set(key, value);`

How to get value using key

Syntax: `map_name(key);` → Will get the value associated to it's key

What will happen if we will add duplicate key and not the value ?

Ans: In that case value new value will be overridden with the existing key

What will happen if we will add unique key and but duplicate value ?

Ans: In that case brand new entry will be created with given key and value

How to get the total number of elements available inside Map

'size' property is used

How to delete element from the Map

Syntax: `map_name.delete(key);`

How to check whether key exists or not?

Syntax: `map_name.has(key)`

How to traverse or iterate map

We can use the for of loop

```
const allKeys = map_name.keys();  
  
for(const key of allKeys ) {  
    console.log(map_name.get(key));  
}
```


Object vs Map

Key type of an object can be only string

Iteration method : for in loop

Creation way

Checking if key exist or not use - in operator

Map allows Keys of any type

Iteration method: for of loop

Creation Way

Checking if key exist or not use - has()

Assignment 0B: Set Of Objects, 11_SetObjectAssignmnet.js

Create a class 'Bank' with all possible data members in such as way that all properties should be initialized using constructor.

- A. Data members: bank_name, location, account_no, ifsc, interest_rate
- B. Create objects - axis_bank, sbi_bank, icici_bank, kotak_bank, hdfc_bank, panjab_bank
- C. Add all object elements in a Set and Traverse this Set using for of loop then just log on console - Bank name and Location

Assignment 0A: Map hands on for objects: 11_MapObjectAssignmnet.js

Create a class 'Bank' with all possible data members in such a way that all properties should be initialized using constructor.

- A. Data members: bankName, location, accountNo, ifsc, interestRate
- B. Create objects - axis_bank, sbi_bank, icici_bank, kotak_bank, hdfc_bank, panjab_bank
- C. Create a Map in such way that key should be accountNo and value is object that is created in step B.
- D. Traverse the map, Log bankName, accountNo and interestRate for each object