@Code.Clash

JavaScript Map

Next →

Hey Everyones 👋

In this Post, you will learn about JavaScript Maps with the help of examples.

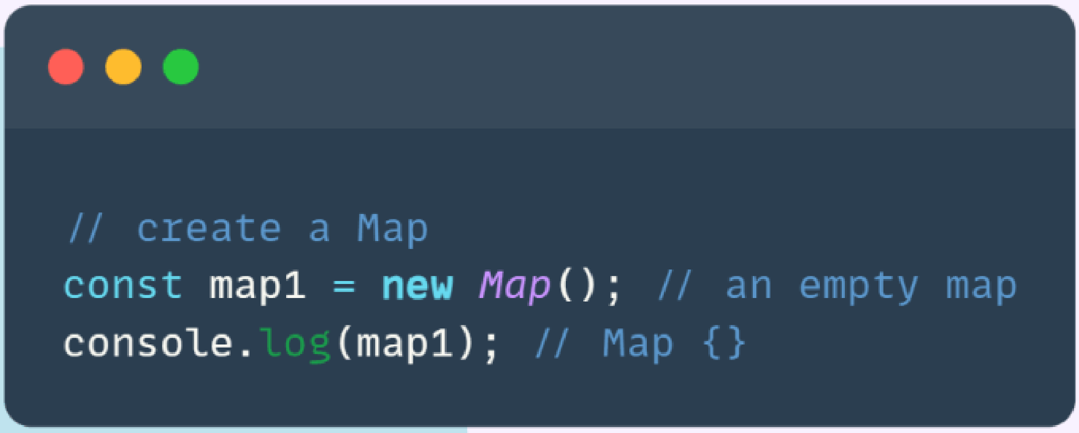
The JavaScript ES6 has introduced two new data structures, i.e Map and WeakMap.

Do Like, save and Share This Post If You Found This Helpful.

Next →

JavaScript Map

- Map is similar to **objects** in JavaScript that allows us to store elements in a **key/value** pair.
- Unlike an object, a map can contain objects, functions and other data types as key.
- To create a Map, we use the new Map() constructor.

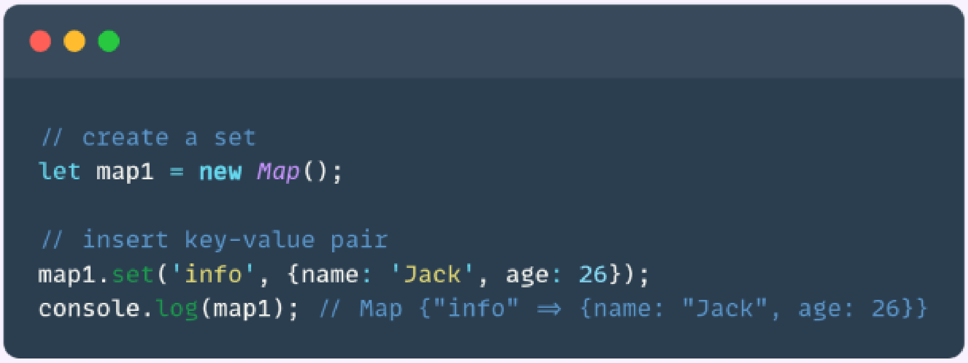


```javascript
// create a Map
const map1 = new Map(); // an empty map
console.log(map1); // Map {}
```

NEXT →

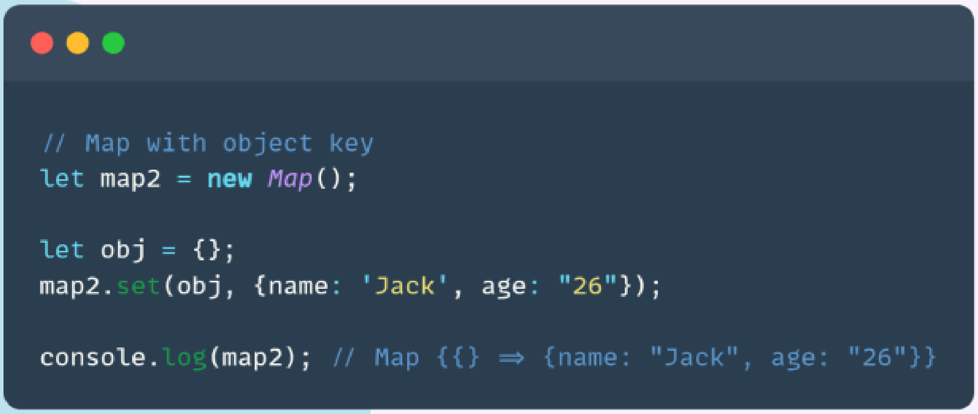
Insert Item to Map

- After you create a map, you can use the set() method to insert elements to it.



```javascript
// create a set
let map1 = new Map();

// insert key-value pair
map1.set('info', {name: 'Jack', age: 26});
console.log(map1); // Map {"info" => {name: "Jack", age: 26}}
```

- You can also use objects or functions as keys.

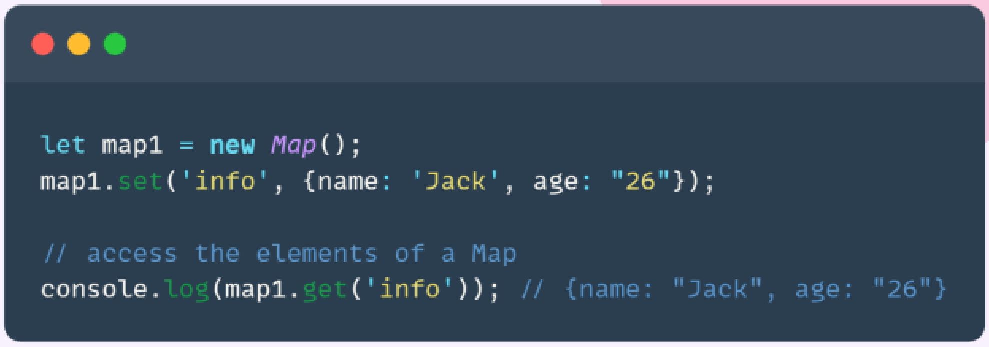


```javascript
// Map with object key
let map2 = new Map();

let obj = {};
map2.set(obj, {name: 'Jack', age: "26"});

console.log(map2); // Map {{} => {name: "Jack", age: "26"}}
```

Next →

1. Access Map Elements

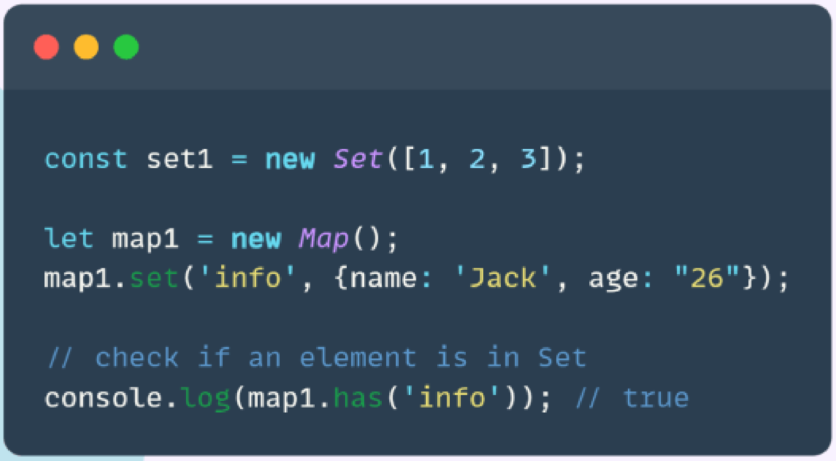
- You can access Map elements using the get() method.



```
let map1 = new Map();
map1.set('info', {name: 'Jack', age: "26"});

// access the elements of a Map
console.log(map1.get('info')); // {name: "Jack", age: "26"}
```

2. Check Map Elements

- You can use the has() method to check if the element is in a Map.

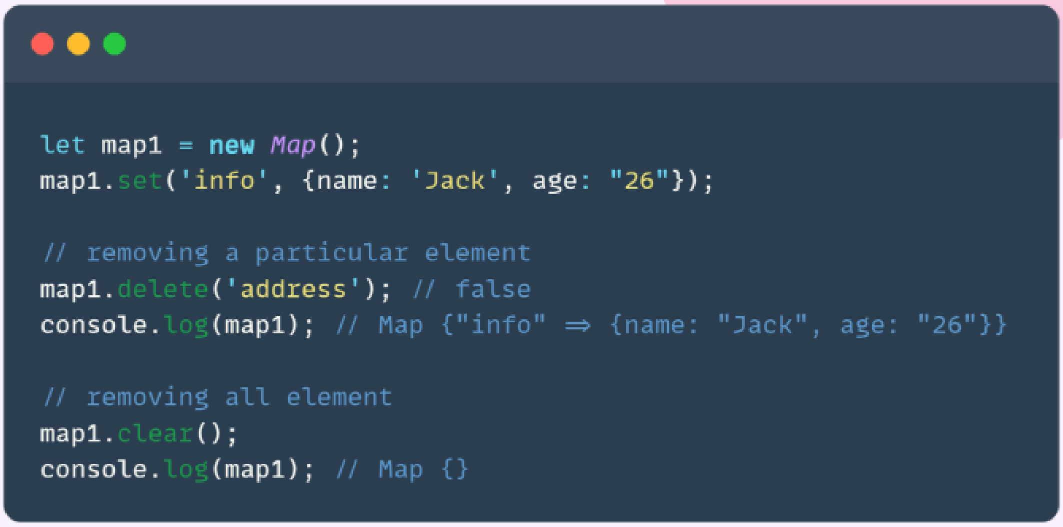


```
const set1 = new Set([1, 2, 3]);

let map1 = new Map();
map1.set('info', {name: 'Jack', age: "26"});

// check if an element is in Set
console.log(map1.has('info')); // true
```

NEXT →

3. Removing Elements

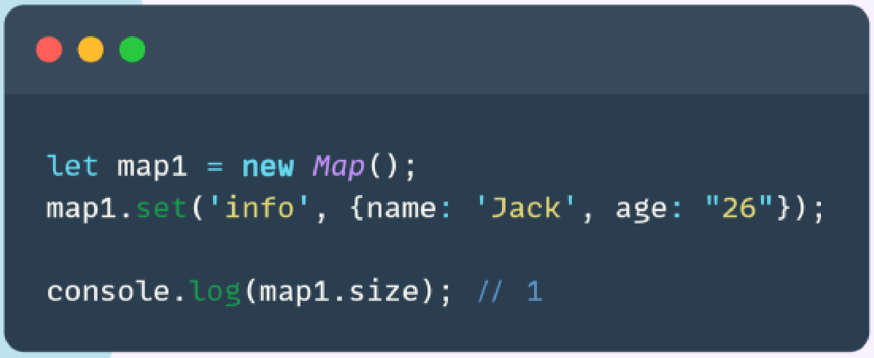
- You can use the clear() and the delete() method to remove elements from a Map.



```
let map1 = new Map();
map1.set('info', {name: 'Jack', age: "26"});

// removing a particular element
map1.delete('address'); // false
console.log(map1); // Map {"info" => {name: "Jack", age: "26"}}

// removing all element
map1.clear();
console.log(map1); // Map {}
```

4. JavaScript Map Size

- You can get the number of elements in a Map using the size property.



```
let map1 = new Map();
map1.set('info', {name: 'Jack', age: "26"});

console.log(map1.size); // 1
```

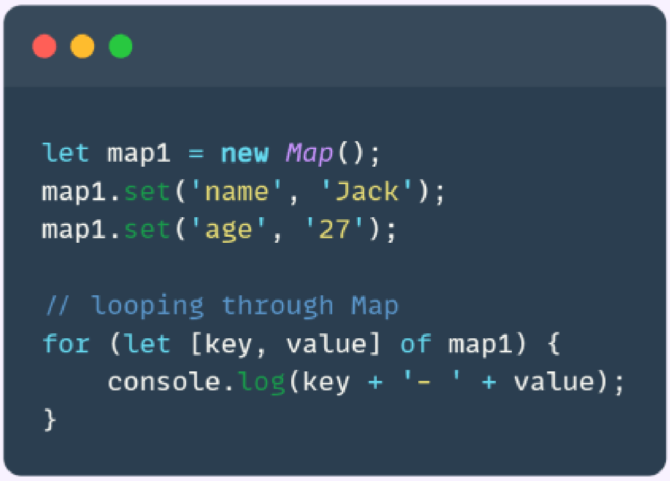
NEXT →

Iterate a Map

1. Using the for...of loop or forEach() method.

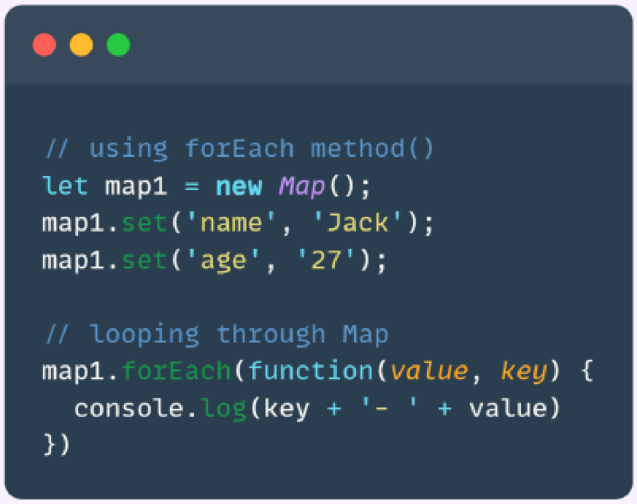
- The elements are accessed in the insertion order.

- For...of loop

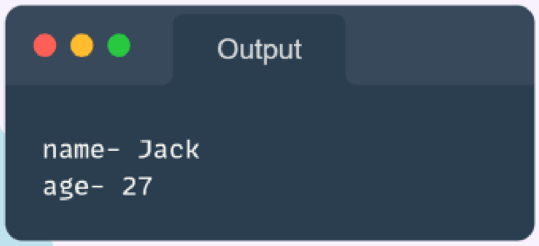


```
let map1 = new Map();
map1.set('name', 'Jack');
map1.set('age', '27');

// looping through Map
for (let [key, value] of map1) {
    console.log(key + '- ' + value);
}
```

- Using forEach



```
// using forEach method()
let map1 = new Map();
map1.set('name', 'Jack');
map1.set('age', '27');

// looping through Map
map1.forEach(function(value, key) {
  console.log(key + '- ' + value)
})
```

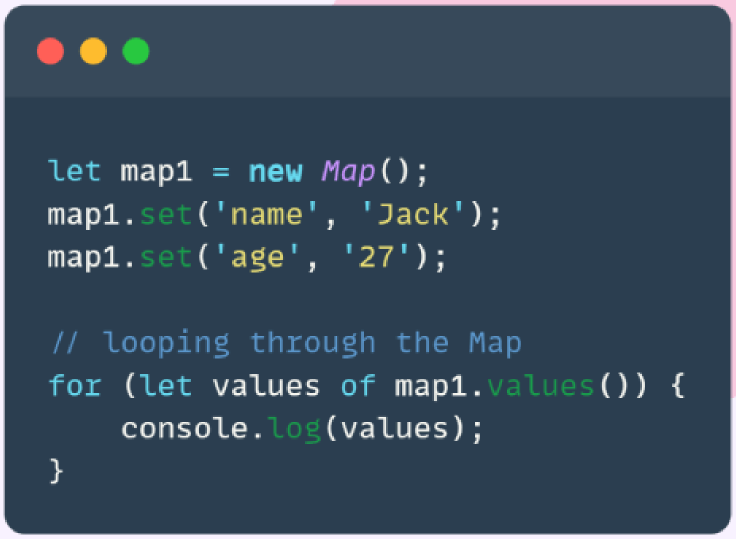


```
name- Jack
age- 27
```

NEXT →

2. Over Map Values

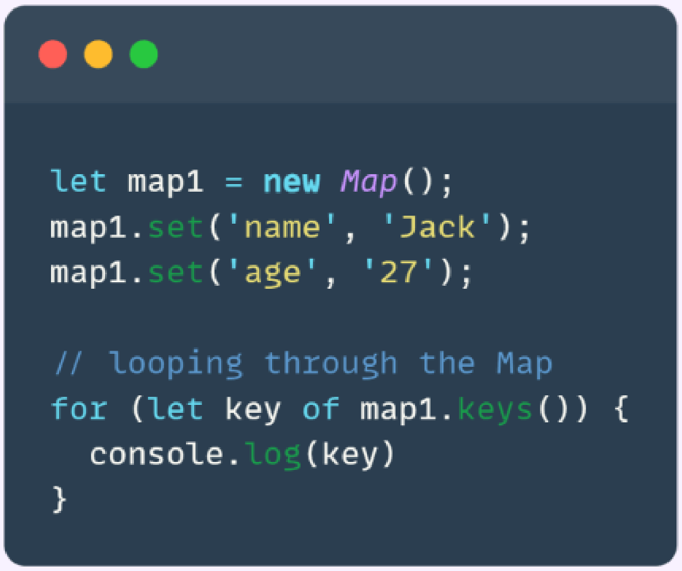
- You can iterate over the Map and get the values using the values() method.



```
let map1 = new Map();
map1.set('name', 'Jack');
map1.set('age', '27');

// looping through the Map
for (let values of map1.values()) {
    console.log(values);
}
```

3. Over Map Keys

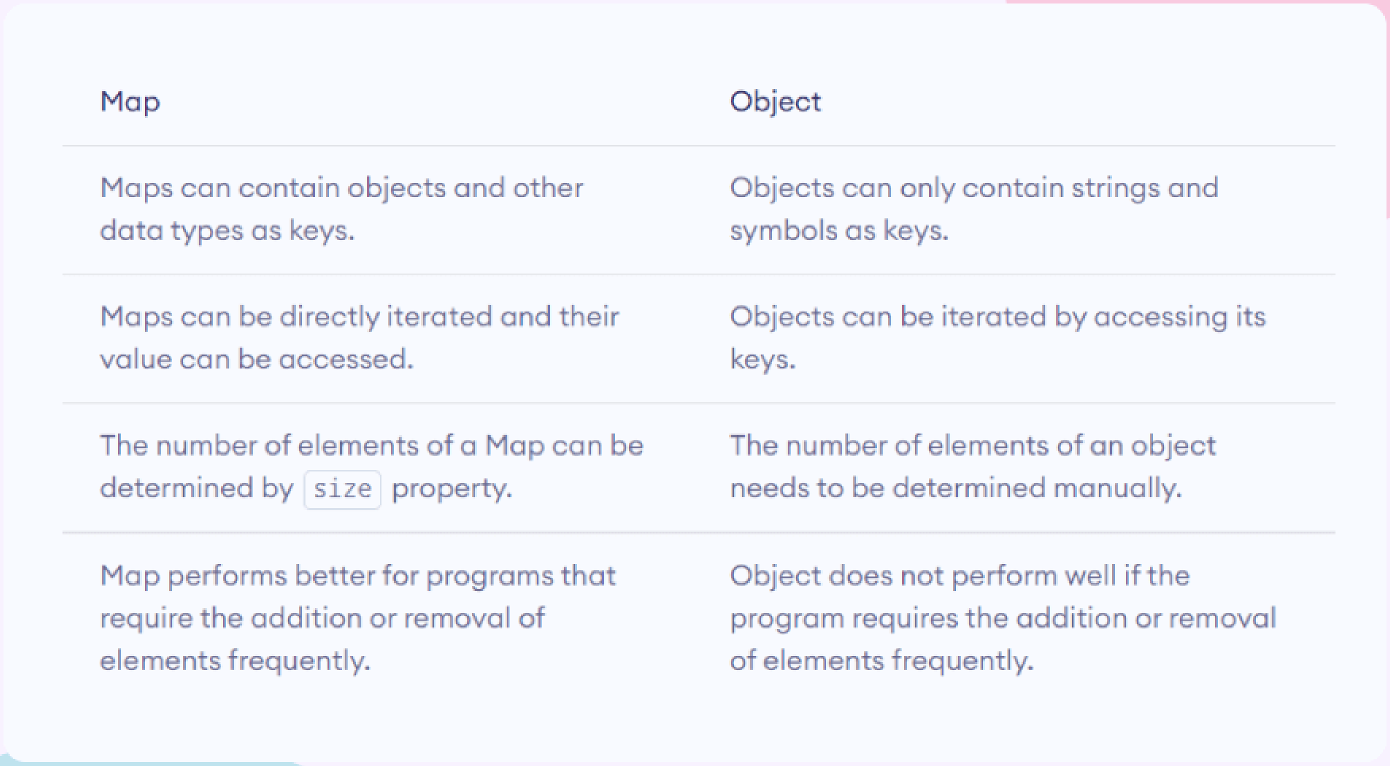
- You can iterate over the Map and get the key using the keys() method.



```
let map1 = new Map();
map1.set('name', 'Jack');
map1.set('age', '27');

// looping through the Map
for (let key of map1.keys()) {
  console.log(key)
}
```

Next →

Map vs Object



Map	Object
Maps can contain objects and other data types as keys.	Objects can only contain strings and symbols as keys.
Maps can be directly iterated and their value can be accessed.	Objects can be iterated by accessing its keys.
The number of elements of a Map can be determined by `size` property.	The number of elements of an object needs to be determined manually.
Map performs better for programs that require the addition or removal of elements frequently.	Object does not perform well if the program requires the addition or removal of elements frequently.

Best Of Luck :)

Next →

@Code.Clash

THANKS FOR YOUR ATTENTION

Code.Clash

Imtiyaz Nandasaniya

LIKE AND SAVE IT

FOR LATER