JS

map

forEach

Both map and forEach are array methods that are used to iterate the elements of the array.

Both methods take a callback function as an argument.





The main difference between map and forEach lies in their return values.





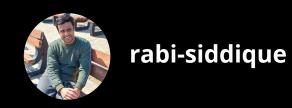
The map method returns a new array from the existing array, after applying the callback function to each element of the existing array.





```
1 let nums = [1, 33, 44, 55];
2
3 const mapResult = nums.map((element) => {
4   return element + 2;
5 });
6
7 console.log(mapResult); // [3, 35, 46, 57]
```





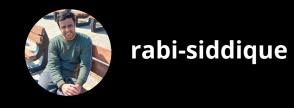
The **forEach** method on the other hand does not return anything. It is just used to iterate the elements of an array.





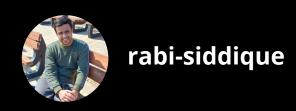
```
1 let nums = [1, 33, 44, 55];
2
3 const forEachResult = nums.forEach((element) => {
4    return element + 2;
5 });
6
7 // prints undefined
8 console.log(forEachResult);
9
```





The other difference between these two methods is based on **chaining**. We can chain other array methods with the **map** method as it has a return value of an array.





```
let nums = [1, 33, 44, 55];
2
3
   const mapResult = nums
     .map((element) => {
4
       return element + 2;
5
6
     })
     .filter((element) => {
       return element > 3;
8
9
     });
10
11
   console.log(mapResult); // [35, 46, 57]
```





While, this is not possible with forEach, as it does not return anything. Instead, chaining other array methods with forEach results in a TypeError.





```
let nums = [1, 33, 44, 55];
2 // This snipped will give:
   // TypeError: Cannot read property 'filter' of undefined
3
   const forEachResult = nums
4
     .forEach((element) => {
5
       return element + 2;
6
     })
7
     .filter((element) => {
8
       return element > 3;
9
10
     });
11
   console.log(forEachResult);
```





Thanks for Reading!:)

let's connect!



