+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

Reduce Method

IN

JAVASCRIPT

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

Hello!

Among all the methods available in JavaScript, reduce it was one of those that didn't seem to be compatible, and unlike other methods like filter and, map at first glance, you could never understand what it was doing.

The solution was to scrutinize this method carefully and share it with you.

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

What does the word reduce means?

One of the meanings of the word reduce is to simplify or crush. Familiarity with this meaning gives us a great understanding of this method.

What does the reduce method do?

We should know that this method is accessible for arrays. According to the meaning of this word, whenever we want to simplify an array, we use this method. What does it mean to simplify the array? It means to make the array so simple and small to reach a single value. Simplification happens to one execution of an arbitrary function.

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

For example, we have an array of numbers and what we want is the sum of the numbers. A large array, and our desired result is a single value.
To reach a single value, we must read the array items from left to right to reach the desired result, which is the sum of the numbers.

The question will surely arise that this task can be written with normal loops as well:

```
var numbers = [1, 2, 3, 4]
var sum = 0

for (var i = 0; i < numbers.length; i++) {
  sum += numbers[i]
}

console.log(sum) // 10
```

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

There are two points:

1. In such a situation, we must always define a variable in advance (sum in line 2)
2. One can think that most problems can be solved with loops and there was no reason to introduce these methods.
Case 2 can be true; But using such methods makes our code more readable and organized, and if it is used in the right place, it shows the intelligence and level of the programmer in recognizing the appropriate method

First, let's check this method and get familiar with its arguments, and then we'll try to understand it better with an example.

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

The method reduce takes two arguments:

```
arr.reduce(callback, initialValue)
```

The first argument (callback) is our desired function, which is used for simplicity. This function takes 4 arguments:

```
function (accumulator, currentValue, index, array) {
    ...
}
```

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

accumulator:

This word means accumulator. This argument stores the output of the callback function with each navigation on the array members and finally outputs it. We can consider the name of this argument as total. That is, the output of all surveys.

currentValue:

There is a member of the array that is being processed now.

Index:

ilt is optional. The index is an item that is being processed now.

array:

It is optional. There is an array that we are working on now.

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

The second argument (initialValue)

is optional and the value we give to it is considered as the initial value for the accumulator. The accumulator must always have an initial value. If we omit this argument, the initial value of the accumulator is equal to the first item of the array.

Now with an example, we can better understand this method. We write the example of adding numbers with this method to get more familiar with its function:

```
var numbers = [1, 2, 3, 4]

var sum = numbers.reduce(function(accumulator, currentValue) {
  return accumulator + currentValue
})

console.log(sum) // 10
```

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

In line 4, we wrote that every time the function is executed, add the current value with the accumulator. simply! Keep in mind that in the example above, since we did not specify initialValue, the initial value of the accumulator is equal to the first value of the array, i.e. 1. In the following code with console.loglines 4 to 6, we want to check this method in more detail. Note the outputs (lines 12 to 22) with each iteration:

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

```
var numbers = [1, 2, 3, 4]

var sum = numbers.reduce((acc, cur, index) => {
  console.log('Accumulator is:' + acc)
  console.log('currentValue is:' + cur)
  console.log('index is:' + index)

  return acc + cur
})

/*
Accumulator is:  1
currentValue is: 2
index is       : 1

Accumulator is:  3
currentValue is: 3
index is       : 2

Accumulator is:  6
currentValue is: 4
index is       : 3
*/
```

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

With the first traversal of the array, the value of the accumulator is equal to 1, the value of currentValue is equal to 2, and the index is equal to 1.
- Why is the first value of currentValue equal to 2, when the first item of the array is 1?
- Why was the first item of the array ignored?
- Why was it only traversed 3 times when the array has 4 items?

The answer is simple. Because the first value of the array is intended for the accumulator and the accumulator must not be re-added to this value.

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

Therefore, in such a situation, the index value starts from 1.
- What if we need the first item of the array?
- What should we do to make the current Value equal to the first value of the array with the first navigation?
- How to have index number 0?

As we said above, we must specify the second argument of the method reduce, which is the initial value of the accumulator. By doing this, the first item of the array is placed in the scrolling flow:

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

```
var numbers = [1, 2, 3, 4]
var initialValue = 50

var sum = numbers.reduce((acc, cur) => {
  console.log('Accumulator is:' + acc)
  console.log('currentValue is:' + cur)
  console.log('index is:' + index)

  return acc + cur
}, initialValue)

/*
Accumulator is:  50
currentValue is: 1
index is       : 0

Accumulator is:  51
currentValue is: 2
index is       : 1

Accumulator is:  53
currentValue is: 3
index is       : 2

Accumulator is:  56
currentValue is: 4
index is       : 3

*/
```

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

As we can see:

- This time 4 surveys were done
- The initial value was equal to the first value of the array
- The initial value of the accumulator was equal to what we specified (50).

Note : In the above examples, because it was written console.log before, the last value of the array has not yet been accumulated with the accumulator. return

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

More examples
Collecting values in an object (ages in the example below)

```
var users = [{ age: 10 }, { age: 20 }, { age: 30 }]
var initialValue = 0

var ages = users.reduce((total, item) => {
  return total + item.age
}, initialValue)

console.log(ages) // 60
```

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

Flatten an array of arrays:

```
var arrays = [[1, 2], [3, 4], [5, 6]]

var flatten = arrays.reduce((acc, arr) => {
    return acc.concat(arr)
    // or
    // return [...acc, ...arr] not recommended
}, [])

console.log(flatten) // [ 1, 2, 3, 4, 5, 6 ]
```

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

Character length of the items of an array:

```
var users = ['Ali', 'John', 'Sarah', 'Napleon']

var len = users.reduce((total, current) => {
  return total + current.length
}, 0)

console.log(len) // 19
```

+92 315 980 7707

https://ismail.vercel.app

ismaeel.kheshgi@gmail.com

Count the number of times an array member is repeated:

```
var chars = ['a', 'a', 'a', 'b', 'b', 'c', 'a']

var repeats = chars.reduce((acc, item) => {
  if (! (item in acc)) {
    acc[item] = 0
  }
  acc[item]++

  return acc
}, {})

console.log(repeats) // { a: 4, b: 2, c: 1 }
```

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

Doubling the number of members of an array:

```
var numbers = [1, 2, 3, 4, 4, 1.5]

var doubled = numbers.reduce((output, item, index) => {
  output[index] = item * 2

  return output
}, [])

console.log(doubled) // [ 2, 4, 6, 8, 8, 3 ]
```

+92 315 980 7707
https://ismail.vercel.app
ismaeel.kheshgi@gmail.com

Of course, using the method mapfor the above example is more appropriate. To learn about the Map and Filter method , see this post .

Well, friends, I hope I was able to reduce the Reduce method :)))
Have a nice day 😉 ✌