Daniyal Ahmed

Web Developer

Master

Map Filter And Reduce

For Your Next Interview

Swipe for more

Daniyal Ahmed

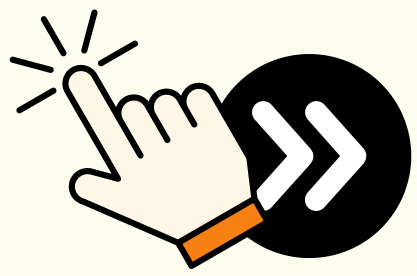
Web Developer

map()

Transforms every element and returns a new array.

```
const nums = [1, 2, 3];
const doubled = nums.map(num => num * 2);
console.log(doubled); // [2, 4, 6]
```

Swipe for more



Daniyal Ahmed

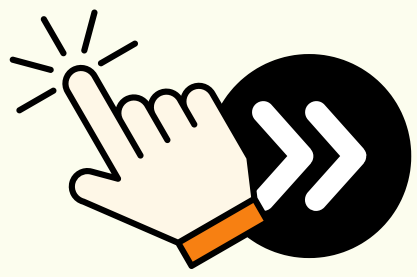
Web Developer

filter()

keeps only the elements that match a condition.

```
const nums = [1, 2, 3, 4];
const evens = nums.filter(num => num % 2 === 0);
console.log(evens); // [2, 4]
```

Swipe for more



Daniyal Ahmed Web Developer

reduce()

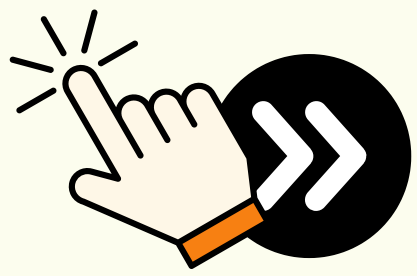
Combines all elements into a single value.

It processes each element in the array one by one, carrying along a result (called the accumulator).

```
array.reduce((accumulator, currentValue, index, array) => {
  // return updated accumulator
}, initialValue);
```

accumulator = The result of previous computation
currentValue = Current item in the array
index = (Optional) Index of current item
array = (Optional) Original array
initialValue = Starting value for the accumulator

Swipe for more



Daniyal Ahmed Web Developer

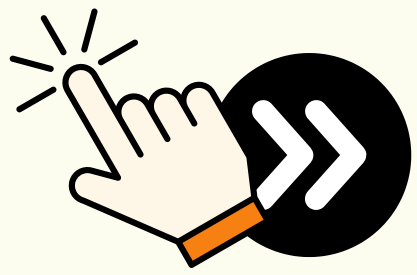
```
const numbers = [1, 2, 3, 4];

const sum = numbers.reduce((acc, curr) => {
  return acc + curr;
}, 0);

console.log(sum); // Output: 10

// 0 + 1 = 1
// 1 + 2 = 3
// 3 + 3 = 6
// 6 + 4 = 10
// Return 10
```

Swipe for more



Daniyal Ahmed

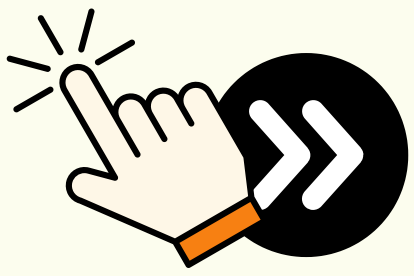
Web Developer

Polyfill for map()

Polyfills help you understand how built-in JavaScript functions really work internally.

```
Array.prototype.myMap = function(cb) {
  let temp = [];
  for (let i = 0; i < this.length; i++) {
    temp.push(cb(this[i], i, this));
  }
  return temp;
};
```

Swipe for more

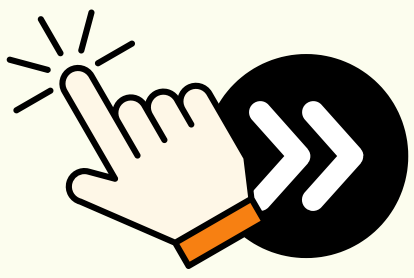


Daniyal Ahmed

Web Developer

Array.prototype.myMap = function(cb) { ... }	You are adding a new method called myMap to all arrays. It accepts a callback function (cb).
let temp = []	Create an empty array to store the results.
for (let i = 0; i < this.length; i++) { ... }	Loop through each element of the array (this refers to the array calling myMap).
temp.push(cb(this[i], i, this));	For each element: → call the callback function cb, passing: - the current element this[i] - the index i - the whole array this Then push the result into temp.
return temp;	After the loop, return the new array!

Swipe for more



Daniyal Ahmed

Web Developer

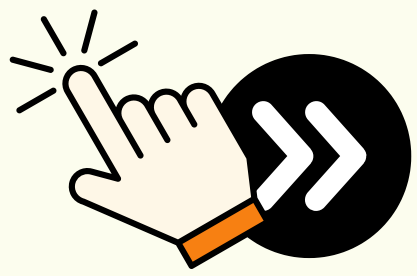
Example Usage

```
const arr = [1, 2, 3];

const doubled = arr.myMap((num, index, array) => {
  return num * 2;
});

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

Swipe for more



Daniyal Ahmed Web Developer

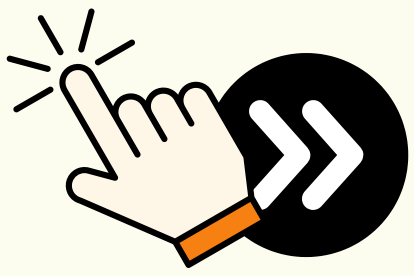
Polyfill for filter()

The only change is now we add element in the newArray if true is returned from the callback

```
Array.prototype.myFilter = function (callback){
  let newArray = [];
  for (let i=0;i<this.length;i++){
    if (callback(this[i],i,this))
      newArray.push(this[i])
  }
  return newArray;
}


const nums = [1, 2, 3, 4];
const evens = nums.myFilter(num => num % 2 === 0);
console.log(evens); // [2, 4]
```

Swipe for more



Daniyal Ahmed Web Developer

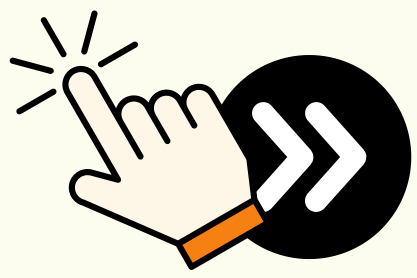
Polyfill for reduce()

It mimics the native Array.prototype.reduce() method — it reduces an array to a single value by applying a callback function.

```
Array.prototype.myReduce = function(callback, initialValue) {
  let accumulator = initialValue;
  for (let i = 0; i < this.length; i++) {
    accumulator = accumulator !== undefined
      ? callback(accumulator, this[i], i, this)
      : this[i];
  }
  return accumulator;
}

const nums = [1, 2, 3, 4];
const sum = nums.myReduce((acc, num) => acc + num, 0);
console.log(sum); // 10
```

Swipe for more

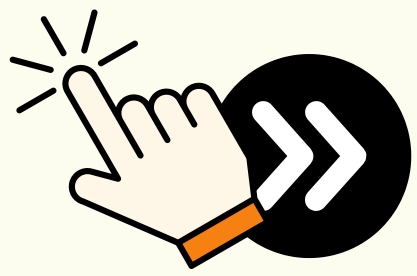


Daniyal Ahmed Web Developer

How It Works ?

1. Starts with initialValue if provided.
2. Loops through the array.
3. If accumulator is defined, applies callback(acc, currentValue).
4. If not, uses the current element as the initial accumulator.
5. Returns the final result.

Swipe for more



Daniyal Ahmed

Web Developer

Practice Questions

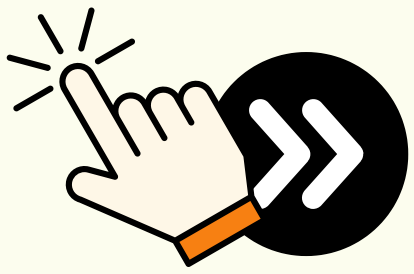
```
// map, filter and reduce - Output Based Questions
// Question 1 (Return only names of students who scored more than 60))
let students = [
  { name: "Piyush", rollNumber: 31, marks: 80 },
  { name: "Jenny", rollNumber: 15, marks: 69 },
  { name: "Kaushal", rollNumber: 16, marks: 35 },
  { name: "Dilpreet", rollNumber: 7, marks: 55 },
];

let passedStudents = students.filter((student) => student.marks > 60);
let names = passedStudents.map((student) => student.name);
console.log(names)
```

```
// Question 2: Calculate the Sum of marks of all students.

const sum = students.reduce((acc, curr) => acc + curr.marks, 0);
console.log("Sum of all marks:", sum); // Output: Sum of all marks: 239
```

Swipe for more



Daniyal Ahmed Web Developer

Practice Questions

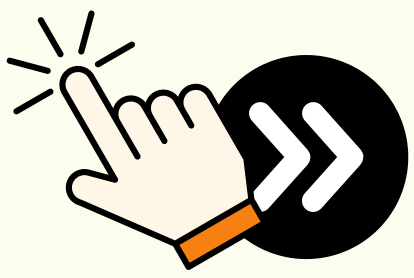
```
// Question 3: Return total marks for students with
// marks greater than 60, after 20 grace marks have been
// added to those who scored less than 60.

const students = [
  { name: "Piyush", rollNumber: 31, marks: 80 },
  { name: "Jenny", rollNumber: 15, marks: 69 },
  { name: "Kaushal", rollNumber: 16, marks: 35 },
  { name: "Dilpreet", rollNumber: 7, marks: 55 },
];

const totalMarks = students
  .map(({ marks }) => (marks < 60 ? marks + 20 : marks))
  .filter((updatedMarks) => updatedMarks > 60)
  .reduce((sum, marks) => sum + marks, 0);

console.log("Total marks:", totalMarks); // Output: 224
```

Swipe for more



Daniyal Ahmed Web Developer

Question 3 Explaination

The solution calculates the total marks of students who score more than 60 after giving grace marks. First, the .map() function is used to iterate over each student's marks. If a student scored less than 60, 20 marks are added as grace; otherwise, the original marks are kept. This results in an updated array of marks. Next, the .filter() function is applied to this new array to keep only those students whose marks are now greater than 60. Finally, the .reduce() function is used to sum up the filtered marks, resulting in the total marks of all eligible students.

Swipe for more

Daniyal Ahmed

Web Developer

If you find this helpful, please like and share it with your connections

@daniyalahmed-dev