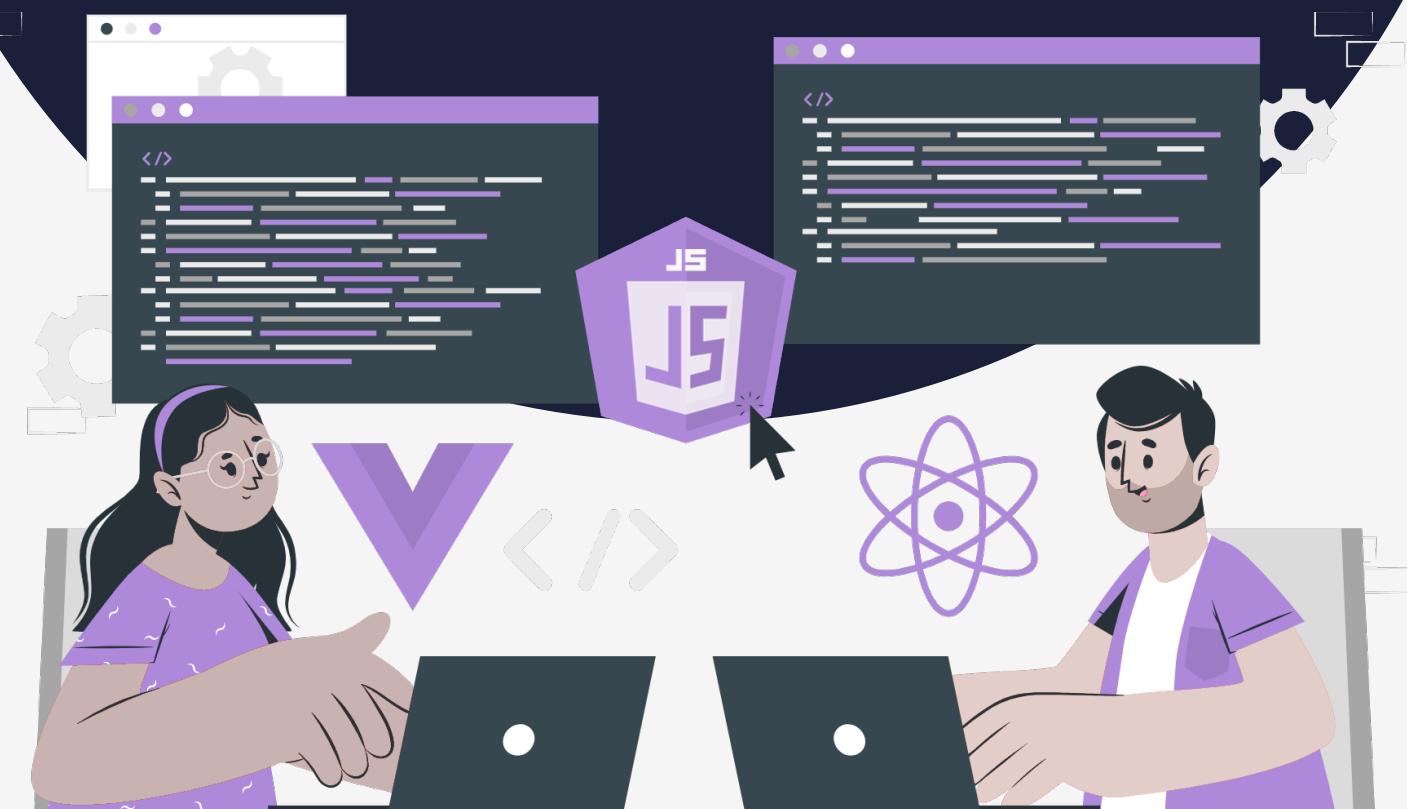


Lesson:

every, find, sort



Topics Covered

1. Introduction to every.
2. Features of reduce.
3. Syntax.
4. Implementation.
5. Introduction to find.
6. Features of find.
7. Syntax.
8. Implementation.
9. Introduction to sort
10. Features of the sort.
11. Syntax.
12. Compare Function.
13. Implementation.

The every method in javascript checks if all the items in an array pass the specified condition or not.

Features of reduce.

1. The condition specified in every method is checked for all the items in the array.
2. If all the array items satisfy the condition, every method returns true.
3. If any of the array items doesn't satisfy the condition, every method returns false.
4. The every method is not valid for an empty array.
5. The every method doesn't change the original array.

Syntax:

```
// Syntax
array.every(function(currentValue, currentIndex, array))
```

The parameters passed:

1. currentValue: It is a required parameter and is used to specify the value of the current element.
2. currentIndex: It is an optional parameter and is used to specify the array index of the current element.
3. array: It is an optional parameter and is used to specify the array the current element belongs to.

Let's look at the implementation of the every method. The simplest example would be to check if all the array elements are of the same data type.

```
// Check if all the array elements are strings
// Given Array

let arr1 = ["PW Skills", "HTML", "CSS", "JavaScript"];
```

```
// Check if all the array elements are strings

// Given Array
let arr1 = ["PW Skills", "HTML", "CSS", "JavaScript"];

// Apply every method
let result = arr1.every((curr) => typeof curr == "string");

// Print the result to console
console.log(result);
```

We are passing a callback function (`curr`) => `typeof curr == "string"` to the `every()` method, which takes in the current element as an argument and checks if its type is "string" using the `typeof` operator.

The `every()` method then applies this callback function to each element in the array `arr1` and returns true if all elements pass the test (if the type is "string"), and false otherwise.

In this case, since all elements in the array `arr1` are strings, the result will be true.

Let's add a number to the array and check what the result would be:

```
// Check if all the array elements are strings

// Given Array
let arr1 = ["PW Skills", "HTML", "CSS", "JavaScript", 200];

// Apply every method
let result = arr1.every((curr) => typeof curr == "string");

// Print the result to console
console.log(result); // OUTPUT: false
```

The above example returns false as the array item 200 is not a string.

find:

The `find()` function in JavaScript is a method used to find the first element in an array that satisfies the condition specified.

It returns the value of the found element, and if no element is found that satisfies the test, it returns `undefined`.

Features of find:

1. The find condition is applied to every array element.
2. The find method returns the first element satisfied by the condition passed.
3. If no element satisfies the find condition, `undefined` is returned.
4. `find` method is not valid on an empty array.
5. The original array is not changed by applying the `find` method.

Syntax:

```
// Syntax

array.find(function (currentValue, currentIndex, array) {
/* Function Body */

});
```

The parameters passed:

1. currentValue: It is a required parameter and is used to specify the value of the current element.
2. currentIndex: It is an optional parameter and is used to specify the array index of the current element.
3. array: It is an optional parameter and is used to specify the array the current element belongs to.

Let's look at the implementation of the find method. We can use the find method to check if the element is present in the array or not.

```
// Check if the element "Javascript" is present in the array.

let techStack = ["HTML", "CSS", "Javascript", "NodeJS", "React JS"];

// Apply find method.

let result = techStack.find((curr) => curr === "Javascript");

// Print the result to the console.

result
? console.log(`The element is present in the techStack`)
: console.log(`The element is not present in the techStack`);
```

We are passing a callback function (`curr`) => `curr === "Javascript"` to the `find()` method, which takes in the current element as an argument and checks if the current element is equal to "Javascript" using the comparison operator (`==`).

The `find()` method then applies this callback function to each element in the array `techStack` and returns the first element that passes the test, which is "Javascript".

The ternary operator checks if the value returned is true or false. This will print "The element is present in the `techStack`" to the console since the result is true. If the element is not present in the array, `find()` will return undefined it will print "The element is not present in the `techStack`"

Sort:

The sort function in javascript is used to sort the string values in an array. It converts the array items into strings, compares them, and sorts the values.

Features of the sort:

1. The elements are compared according to their sequences of UTF-16 code unit values.
2. By default, the arrays are sorted in ascending order.
3. The sorting operation happens on the same array and no new array is created.

Syntax:

```
// Without any parameter
arr.sort();

// Compare function method
arr.sort(compareFunc)

// Arrow function method
arr.sort((x, y) => { /* Function Body */ } )

// Inline compare function method
arr.sort(function compareFunc(x, y) { /* Function Body */ })
```

Parameters:

A compare function in JavaScript is a function that is used to compare two elements of an array when sorting the array using the `sort()` method. The function takes two arguments, which are the two elements being compared, and returns a value indicating the relative order of the elements.

The `compareFunction` takes two parameters `a` and `b` and returns three possible values: 0, negative or positive.

<code>compareFunction(a, b)</code> return value	sort order
<code>> 0</code>	sort <code>b</code> before <code>a</code>
<code>< 0</code>	sort <code>a</code> before <code>b</code>
<code>== 0</code>	keep original order of <code>a</code> and <code>b</code>

```
// Compare Function

function compareFunc(a, b) {
  if (a < b) {
    return -1;
  } if (a > b) {
    return 1;
  }

  // When a and b are equal
  return 0;
}
```

We will look into the implementation of compare function in below examples.

Let's look at the implementation of the sort function. First, let's sort the string elements in an array.

```
// Sort a list of string items

// Given an array of strings
let names = ["Anurag", "Mithun", "Alka", "Prabir", "Vinay"];

// Apply sort methodnames.sort();

// Print the result

console.log(names); // [ 'Alka', 'Anurag', 'Mithun', 'Prabir', 'Vinay' ]
```

Let's now sort an array of numeric values.

```
// Sort Numeric values

// Given Array
let numbers = [18, 155, 78, 97, 86];

// Apply sort
numbers.sort();

console.log(numbers); // OUTPUT: [ 155, 18, 78, 86, 97 ]
```

The code you provided sorts the numbers array in ascending order, however, it is not sorting the array as expected because the `sort()` method sorts the elements of an array in ascending order as strings, which is not suitable for sorting numeric values.

So, we need to use a compare function to sort the numbers.

```
// Sort Numeric values

// Given Array
let numbers = [18, 155, 78, 97, 86];

// Apply sort
numbers.sort((a, b) => a - b);

console.log(numbers); // OUTPUT: [ 18, 78, 86, 97, 155 ]
```

This is the correct way to sort an array of numeric values in ascending order.

The `sort()` method takes an optional compare function as a parameter, which is used to determine the sort order of the elements in the array.

In this case, you provided a comparator function that takes two arguments, `a` and `b`, and subtracts `b` from `a`.

This will sort the array in ascending order, so the smallest number will be at the first position, and the largest number will be at the last position.