JS JavaScript TUTORIAL

Array.prototype.indexOf()

Summary: in this tutorial, you will learn how to use JavaScript Array `indexOf()` method to find the index of the first matching element in an array

Introduction to the JavaScript Array indexOf() method

The Array `indexOf()` method returns the index of the first matching element in an array or `-1` if there is no matching element.

Here's the syntax of the `indexOf()` method:

```
const index = array.indexOf(searchElement, fromIndex)
```

The `indexOf()` method accepts two arguments:

- `searchElement` is the element to locate in the array.
- `fromIndex` is a zero-based index at which the method starts searching.

If you omit the `fromIndex`, the `indexOf()` method starts searching from the beginning of the array.

The `fromIndex` argument can be a positive or negative integer.

If `fromIndex` is positive, the method starts searching from the `fromIndex` toward the end of the array. If `fromIndex >= array.length`, then the method returns `-1` without carrying a search.

A negative `fromIndex` counts back from the end of the array and the `indexOf()` method still searches from the *front to the back* of the array.

If `fromIndex` is negative and `>= -array.length`, the method starts searching from `fromIndex` to the end of the array.

If `fromIndex < - array.length` , the method searches the entire array.

By default, the `indexOf()` method starts searching from the `fromIndex` to the end of the string. If you omit the `fromIndex` , the `indexOf()` method starts searching from the beginning of the string.

Notice that the `indexOf()` method uses the strict equality comparison algorithm that is similar to the triple-equals operator (`===`) when comparing the `searchElement` with the elements in the array.

JavaScript Array indexOf() method examples

Let's take some examples of using the `indexOf()` method.

Basic JavaScript Array indexOf() method example

The following example uses the `indexOf()` method to locate the number `20` in the `scores` array:

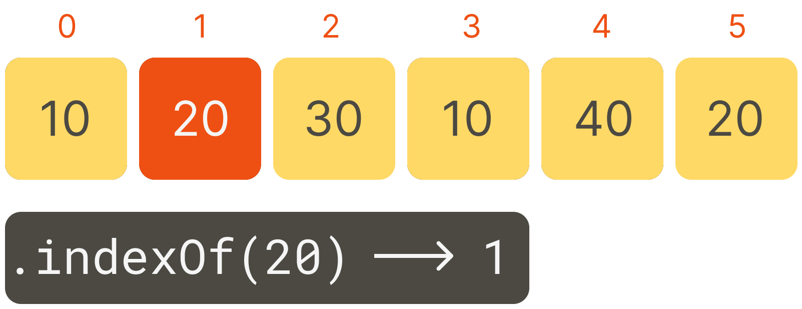
```
const scores = [10, 20, 30, 10, 40, 20];
const index = scores.indexOf(20);

console.log({ index });
```

Output:

```
{ index: 1 }
```

In this example, the `indexOf()` method returns `1` which is the second position in the array:



Using the fromIndex argument

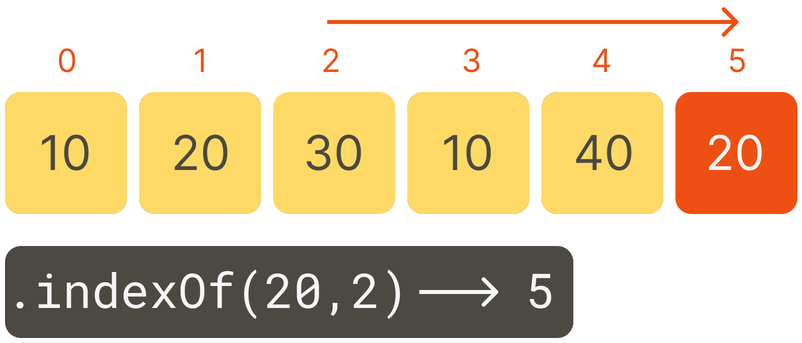
The following example uses the `indexOf()` method to locate the number `20` in the scores array starting from the index 2:

```
const scores = [10, 20, 30, 10, 40, 20];
const index = scores.indexOf(20, 2);

console.log({ index });
```

Output:

```
{ index: 5 }
```



Using a negative fromIndex argument

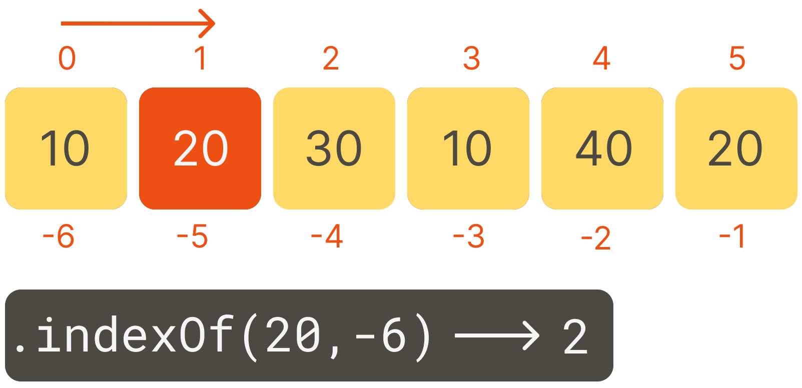
The following example uses the `indexOf()` method to locate the number 20 in the `scores` array starting from a negative index `-6`, which is starting from the beginning of the array:

```
const scores = [10, 20, 30, 10, 40, 20];
const index = scores.indexOf(20, -6);

console.log({ index });
```

Output:

```
{ index: 1 }
```



Finding indices of all occurrences

To find all the indexes of an element in an array, you can use the indexOf with a do-while loop:

```
const scores = [10, 20, 30, 10, 40, 20];

const results = [];

let index;
let fromIndex = 0;

do {
  index = scores.indexOf(20, fromIndex);

  if (index !== -1) {
    results.push(index);
    fromIndex = index + 1;
  }
} while (index !== -1);

console.log({ indexes: results });
```

Output:

```
{ indexes: [ 1, 5 ] }
```

The output indicates that the number `20` appears at index 1 and 5, or 2nd and 6th position in the `scores` array.

Using the indexOf() method with a negative fromIndex argument

The following example uses the `fromIndex()` with the negative values:

```
const scores = [10, 20, 30, 10, 40, 20];
const index = scores.indexOf(20, -1);

console.log({ index });
```

Output:

```
{ index: 5 }
```

The following `allIndexOf()` function returns an array of indexes of all occurrences of an element in an array:

```
function allIndexOf(needle, haystack) {
  const results = [];
  let index = haystack.indexOf(needle);
  while (index != -1) {
    results.push(index);
    index = haystack.indexOf(needle, index + 1);
  }
  return results;
}
```

Locating an object in an array

The following example attempts to locate an object in an array of objects using the `indexOf()` method:

```
const guests = [
  { name: 'John Doe', age: 30 },
  { name: 'Lily Bush', age: 20 },
  { name: 'William Gate', age: 25 },
];
```

```
const index = guests.indexOf({
  name: 'John Doe',
  age: 30,
});

console.log({ index });
```

The two objects are different despite having the same properties and values.

To locate an object in an array of objects by some properties, you can use the findIndex() method like this:

```
const guests = [
  { name: 'John Doe', age: 30 },
  { name: 'Lily Bush', age: 20 },
  { name: 'William Gate', age: 25 },
];

const guestIndex = guests.findIndex((g) => g.name == 'John Doe' && g.age == 30);

console.log({ guestIndex });
```

Output:

```
{ guestIndex: 0 }
```

Summary

- Use the JavaScript array `indexOf()` method to locate an element in the array.