Javascript array search and remove string?

Asked 7 years, 6 months ago Active 3 months ago Viewed 148k times

102

17

I have:

```
var array = new Array();
array.push("A");
array.push("B");
array.push("C");
```

I want to be able to do something like:

```
array.remove("B");
```

but there is no remove function. How do I accomplish this?

javascript arrays

edited Mar 20 '12 at 18:48 Rob W 284k 55 671 597

asked Mar 20 '12 at 18:40 Rolando 13.3k 69 187 311

4 A combination of `.indexOf()` and `.splice()` should do the trick. Or maybe, alternatively, `.filter()` . – Marc B Mar 20 '12 at 18:42

1 see here: stackoverflow.com/questions/3954438/… – benedict_w Mar 20 '12 at 18:44

Possible duplicate of How to remove item from array by value? – totymedli Mar 6 '18 at 17:24

10 Answers

I'm actually updating this thread with a more recent 1-line solution:

```
let arr = ['A', 'B', 'C'];
arr = arr.filter(e => e !== 'B'); // will return ['A', 'C']
```

The idea is basically to filter the array by selecting all elements different to the element you want to remove.

Note: will remove all occurrences.

edited Feb 7 '18 at 9:39

answered Jun 8 '17 at 10:21

Tyrannas

1,231 1 7 7

1 This solution returns a copy of the array, whereas using splice removes the element(s) in place. Which you choose depends on context. – twhitehead Dec 18 '17 at 9:39

5 This is perfect for Redux stuff where you need to return a new state. – colinwong Jan 11 '18 at 20:30

@Regis actually not, arr.filter returns a new array. So arr.filter(e => e!== 'B') won't modify arr. Or maybe I didn't understand your comment correctly? – Tyrannas Dec 5 '18 at 21:51

168

Loop through the list in reverse order, and use the `.splice` method.

```
var array = ['A', 'B', 'C']; // Test
var search_term = 'B';

for (var i=array.length-1; i>=0; i--) {
    if (array[i] === search_term) {
        array.splice(i, 1);
        // break;       //<-- Uncomment  if only the first term has to be removed
    }
}
```

The reverse order is important when **all** occurrences of the search term has to be removed. Otherwise, the counter will increase, and you will skip elements.

When only the first occurrence has to be removed, the following will also work:

```
    array.splice(index, 1);
}
```

answered Mar 20 '12 at 18:41

Rob W
284k 55 671 597

1 i'm guessing because it's meant to be slightly faster to iterate in reverse. – Ben Clayton Mar 20 '12 at 18:43

1 @BenClayton: Thanks. FWIW, in JavaScript, that's not reliably true. Counting down to `0` isn't automatically faster like it is in, say, C. So long as you cache the limit, of course, which would complicate things if you keep going after the first match (but not if you stop on it). – T.J. Crowder Mar 20 '12 at 18:44

If we're going for speed why not use while --? :D – Snuffleupagus Mar 20 '12 at 18:47

11 It's not about speed, he even says so in his answer. It's about SKIPPING elements. If you're at position 5 and you splice that position, the element formelry located at position 6 *is now at 5*. Still, your loop-counter increases, next iteration is position 6 and that is where you skipped an item. That's why it's in reverse order. – amenthes Aug 21 '15 at 22:09

1 If you remove items in a forward loop, and an item gets removed, the last iteration can throw null pointer exceptions as it will be referencing an index that does not exist – Drenai Feb 15 '18 at 19:03

DEMO

19 You need to find the location of what you're looking for with `.indexOf()` then remove it with `.splice()`

```
function remove(arr, what) {
    var found = arr.indexOf(what);

    while (found !== -1) {
        arr.splice(found, 1);
        found = arr.indexOf(what);
    }
}

var array = new Array();
array.push("A");
array.push("B");
```

```
remove(array, 'B');
alert(array);
```

This will take care of all occurrences.

answered Mar 20 '12 at 19:16

qwertymk
19.5k 22 99 175

For browsers that don't support `.indexOf()` you can add **this** to your javascript file. – qwertymk Mar 20 '12 at 19:23

yep, elegant. If you need an option to remove only some elements, e.g. only the first: the same updated: jsfiddle.net/qpZFd/9 – sebilasse Jul 19 '15 at 15:11

I always get the following error: `Uncaught ReferenceError: array is not defined` . What is wrong? – Pathros Oct 15 '15 at 17:27

If you are going this route, you can easily take advantage of `.indexOf()` just a little more. If you pass `found` as the second argument to the `.indexOf()` call **within the while-loop**, the elements in the array that were already checked and ended up not being equal are not checked again: `found = arr.indexOf(what, found);` – pimmhogeling Apr 11 '16 at 14:07

18

List of One Liners

Let's solve this problem for this array:

```
var array = ['A', 'B', 'C'];
```

1. Remove only the first: Use If you are sure that the item exist, Not supported in <IE9

```
array.splice(array.indexOf('B'), 1);
```

2. Remove only the last: Use If you are sure that the item exist, Not supported in <IE9

```
array.splice(array.lastIndexOf('B'), 1);
```

3. Remove all occurrences: Not supported in <IE9

answered May 26 '18 at 21:45

enesn

393 3 8

13

Simply

```
array.splice(array.indexOf(item), 1);
```

answered Jul 27 '18 at 18:36

MM

Matt

1,037 4 15

2

Simple solution (ES6)

If you don't have duplicate element

```
Array.prototype.remove = function(elem) {
  var indexElement = this.findIndex(el => el === elem);
  if (indexElement != -1)
    this.splice(indexElement, 1);
  return this;
};
```

[Online demo (fiddle)]

edited Feb 5 '18 at 8:22

answered Dec 2 '17 at 8:51

Ali Soltani

6,699 2 14 34

This solution always removes the last element if NO match is found. – markus s Feb 5 '18 at 7:33

1

You have to write you own remove. You can loop over the array, grab the index of the item you want to remove, and use `splice` to remove it.

Alternatively, you can create a new array, loop over the current array, and if the current object doesn't match what you want to remove, put it in a new array.

answered Mar 20 '12 at 18:42

hvgotcodes
97.1k 22 180 226

1

use:

```
array.splice(2, 1);
```

This removes one item from the array, starting at index 2 (3rd item)

edited Feb 10 '18 at 20:26

answered Mar 20 '12 at 18:42

Ben Clayton
65.6k 23 111 122

1 actually it'll remove second item from the array, index starts from zero. this statement has ambiguity, more simple example could be like `array.splice(2,1)` which removes 1 item at index 2 from array. check https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/splice for more details – imdzeeshan Jan 18 '18 at 16:54

0

use array.splice

```
/*array.splice(index , howMany[, element1[, ...[, elementN]]])

array.splice(index) // SpiderMonkey/Firefox extension*/

array.splice(1,1)
```

Source: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/splice


chepe263
2,198 17 35

The comma between `array` and `splice` has to be a dot. – Rob W Mar 20 '12 at 18:50

4 w3fools.com – jbabey Mar 20 '12 at 19:02

Tried to correct but SO's policy states that edits must be 6 characters or more :/ – ben_nuttall Feb 19 '14 at 14:58

0

```
const changedArray = array.filter( function(value) {
  return value !== 'B'
});
```

or you can use :

```
const changedArray = array.filter( (value) => value === 'B');
```

The changedArray will contain the without value 'B'

answered Jun 12 at 13:52


siva gopi
1 1