How to create an array of unique values in JavaScript using Sets

Claire Parker-Jones May 8

#javascript #webdev

TL;DR

```
let uniqueArray = [...new Set([5,5,2,2,2,4,2])];
// [5,2,4]
```

Pardon?

Sorry for the verbose title - sometimes things can be explained better with a code example.

Imagine you have a JavaScript array that contains many elements, some of which are duplicated:

```
let dupeArray = [1,1,4,5,4,4,2,1,5];
```

Your goal is to remove the duplicates and leave only one entry per value:

```
let uniqueArray = [1,4,5,2];
```

result. There are a million different ways to tackle a problem in computer programming (which is what makes it so fun!). ES6 introduces a bunch of new features, including Sets, which we can also use to solve this problem in one line of code.

Note: the element type could be strings, or a mixture of numbers and strings, but I'm using integers here because they are quicker to type!

What's a Set?

A Set is a new data structure introduced in ES6. Quoting directly from MDN:

> Set objects are collections of values. You can iterate through the elements of a set in insertion order. A value in the Set

search 1

The rule for unique values is one we can use to our advantage here.

Let's create a Set, add some values to it and query the size. (You can follow along in the developer tools console of a modern browser):

```
let mySet = new Set().add(1).add(3).add(2).add(1);
// Set(3) {1, 3, 2}
mySet.size
// 3
```

remained at 3 instead of 4. In an array, it would have been added and the length would be 4.

There are two ways to add values to a Set. Firstly by using the `add` method as above. Secondly by passing an array to the constructor (`new Set()`):

```
let arraySet1 = new Set([3,2,1]);
// Set(3) {3, 2, 1}
```

Duplicated values are also removed if they are included in the initial array:

```
let arraySet2 = new Set([8,8,9,2]);
// Set(3) {8,9,2}
arraySet2.size;
// 3
```

You can probably see where this is going. All that's left to do is to convert this Set into an array and we've achieved our original goal. There are two ways to do this: both using ES6 methods.

`Array.from()`

The `Array.from` method creates a new array from an array-like structure:

```
let uniqueArray = Array.from(new Set(dupeArray));
```

Spread operator ...

Those three dots are ubiquitous in ES6. They crop up everywhere and have several uses (and they're a right pain to google). When we use them as the spread operator they can be used to create an array:

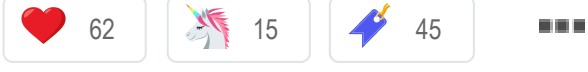
```
let uniqueArray = [...new Set(dupeArray)];
```

Which of these two methods should you use? The spread syntax looks cool, but `Array.from` is more explicit in its purpose and easier to read. They both accomplish the same thing here so the choice is yours!

Conclusion

Something that would have took many lines of code and variables can now be executed in a one-liner in ES6. What a time to be alive.

```
let uniqueArray = [...new Set([5,5,2,4,2])];
// [5,2,4]
```



Claire Parker-Jones + FOLLOW

Frontend web developer and dog mum

ClaireParkerPen claireparker www.claireparker.co.uk

Add to the discussion

PREVIEW SUBMIT

Xing Wang

May 9

Good highlight this feature.

However, it maybe worth pointing out that with Set, you can't control the equality operator.

It uses '===', which only work only off same actual object or same value for a primitive.

Which limits the usage somewhat compares to other methods such as map or filter.

2 REPLY

Sebastian Kolind Sørensen

May 9

I will implement this right away in a project I am working on. Although I like the `Array.from()` better, since it's easier to read. I like easy-to-read code :) Thank you for sharing!

3 REPLY

Ashish Mehra

May 9

I used to filter and map over an array to remove duplicates. Thanks for sharing this method, Mam.

4 REPLY

Hey there, we see you aren't signed in. (Yes you, the reader. This is a fake comment.)

Please consider creating an account on dev.to. It literally takes a few seconds and ***we'd appreciate the support so much***. ♡

Plus, no fake comments when you're signed in. 🙃

JOIN THE DEV COMMUNITY

Seth T

May 13

Is there any difference in performance between `Array.from` and using the spread operator?

♡ 1

REPLY

JavaScriptErika

May 30

Great article and explanations! Thank you for your awesome insight and clear examples!

♡ 1

REPLY

Sai gowtham

May 9

I have posted a similar thing

How to remove Duplicate elements from the Array

Sai gowtham

#javascript #beginners #showdev #programming

♡ 1

REPLY

Marcelo Faundez

May 9

Great tip. Thanks

♡ 1

REPLY

What if the elements are deep objects themselves rather than just basic elements like integers and alphabets.

REPLY

nuton.dev

May 10

We will read about it in your post, right? :)

2

REPLY

code of conduct - report abuse

Classic DEV Post from Mar 16

Are you an introvert, extrovert, or somewhere in between?

Ben Halpern

...

71 63

READ POST SAVE FOR LATER

Classic DEV Post from May 24

Choose Your Own Coding Adventure!

Cat Carbonell

Where to start? There are so many paths to take!

124 17

READ POST SAVE FOR LATER

Twitter bot that generates an image with your quote!


Rizqan Arief

Learning Javascript by making a Twitter bot.

84 6

READ POST SAVE FOR LATER


Building a Pomodoro Timer with Vue.js on CodePen
Tori Pugh - May 31


Play With the React Router
Sai gowtham - Jun 1


Under the Hood of the Most Powerful Video JavaScript API
The JW Player Team - Jun 1


How To Build Your First Chrome Extension
Himashi Hettege Dona - May 31