Most popular (last 30 days)

ECMAScript 2017: the final feature set
ECMAScript 6 modules: the final syntax
ES proposal: import() – dynamically importing ES modules
Making transpiled ES modules more spec-compliant
ES proposal: Shared memory and atomics
Classes in ECMAScript 6 (final semantics)
Communicating between Web Workers via MessageChannel

Most popular (all time)

ECMAScript 6 modules: the final syntax
Classes in ECMAScript 6 (final semantics)
Iterating over arrays and objects in JavaScript
ECMAScript 6's new array methods
The final feature set of ECMAScript 2016 (ES7)
WebAssembly: a binary format for the web
Basic JavaScript for the impatient programmer
Google Dart to "ultimately ... replace JavaScript"
Six nifty ES6 tricks
Google's Polymer and the future of web UI frameworks

Blog archive

► 2017 (4)
► 2016 (38)

2012-12-25

Two ways of clearing an array in JavaScript

Labels: dev, javascript, jslang

Update 2012-12-27: Better example for aliasing in Sect. 1.

In a blog post, David Walsh mentions two approaches for emptying (clearing) an array. This blog post explains the pros and cons of both approaches. In order to understand them, we first need to know about *aliasing*.

1. Background: aliasing

Aliasing means that the same piece of mutable data can be accessed from several locations (variables, properties, etc.) in a program. Sometimes aliasing is useful, sometimes it is harmful. In this section we examine an example where its effects are harmful. Take the following object, `fruitBasket`, a data structure for fruits:

```
var fruitBasket = {
    _fruits: [ 'Apple', 'Orange' ],
    getFruits: function () {
        return this._fruits;
    }
};
```

Obviously, this is not very good code, but it illustrates a general risk related to aliasing: Letting a reference to an internal data structure escape to the outside. Code that uses `fruitBasket` might look like this:

```
var thingsIHaveEaten = fruitBasket.getFruits();  // (*)
thingsIHaveEaten.push('Cheese');
console.log('Things I have eaten: '+thingsIHaveEaten);
```

At (*), `thingsIHaveEaten` has become an alias for `fruitBasket._fruits`. Thus, the outside has gained access to something that should be hidden inside `fruitBasket`. And after the above three lines of code have been executed, the integrity of fruitBasket has indeed been breached:

```
> fruitBasket.getFruits()
[ 'Apple', 'Orange', 'Cheese' ]
```

How can this be fixed? `fruitBasket` simply has to return a copy of `this._fruits`, instead of returning the original:

```
var fruitBasket = {
    _fruits: [ 'Apple', 'Orange' ],
    getFruits: function () {
        return this._fruits.slice();  // shallow copy
    }
};
```

We have performed a *shallow copy*: the copy is new, but the elements are the same.

2. Clearing an array

Let's say we want to clear the array in the following variable `myArray`:

Dr. Axel Rauschmayer

Free online books by Axel

Speaking JavaScript [up to ES5]
Exploring ES6

JavaScript training: Ecmanauten

```
var myArray = [ elem0, elem1, ... ];
```

First, you can replace the current value with an empty array. Second, you can set the array’s length to zero.

2.1. Replace an array with an empty array

The first way of clearing `myArray` works like this:

```
myArray = [];
```

Depending on where the old value of `myArray` came from, this is the safer alternative, because you don’t change that value. You do create extra garbage: you don’t reuse the existing array, you create a new one. However, garbage collection is quite fast, so that it rarely matters (it also helps avoid an effect that incurs even more of a performance penalty, as we shall see below).

2.2. Set the array’s length to zero

The second way of clearing `myArray` works like this:

```
myArray.length = 0;
```

If the value of `myArray` is shared and all participants have to see the effect of clearing then this is the approach you need to take. However, JavaScript semantics dictate that, if you decrease the length of an array, all elements at the new length and above have to be deleted [1]. And that costs time (unless an engine has an optimization for the special case of setting a length to zero). Indeed, a performance test (mentioned by Kris Zyp) shows that the previous way of clearing is faster on all current JavaScript engines.

3. Reference

[1] Arrays in JavaScript

10 Comments **The 2ality blog** 1 **Login**

Recommend 1 **Share** **Sort by Best**

Join the discussion…

Dmitri Pavlutin • a year ago

Indeed myArray = []; does not clear the array, but creates a new reference. Another way to clear an array: myArray.splice(0). And this may be the preferable one.

1 • Reply • Share ›

Jerome Covington • 2 years ago

myArray = []; does not seem to really "clear" anything. It just reassigns the variable, without any context as to the variable's previous state or "type". Though faster, it is as meaningful as the (absurd) myArray = '';

• Reply • Share ›

Axel Rauschmayer Mod → Jerome Covington • 2 years ago

You seem to understand what sharing is about. Many people don’t.

• Reply • Share ›

D8 • 2 years ago

but your

fruitBasket._fruits is available to the 'outside world' already, there aren't any private variables in classes

• Reply • Share ›

Axel Rauschmayer Mod → D8 • 2 years ago

Yes. Privacy in these examples is via naming convention only. This is a different kind of danger: If a method returns something and you change it, you don’t expect the state of the object to change.

▶ 2015 (65)
▶ 2014 (55)
▶ 2013 (98)
▼ 2012 (177)
  ▼ December (14)
    Looking back on 2012
    The problem with blogs
    Two ways of clearing an array in JavaScript
    “Skyfall” plot holes
    Language complexity: C++, Ruby, CoffeeScript and J...
    The future of JavaScript: a CoffeeScript-like work...
    New Samuel: a font derived from Morse code
    JavaScript puzzle: equal, but not the same
    A few thoughts on iTunes 11
    Using ES6 template strings for regular expressions...
    I’m the new JavaScript Weekly editor
    A new way of delivering Retina images on the web
    Arrays in JavaScript
    Next for iOS: better sharing
  ▶ November (14)
  ▶ October (15)
  ▶ September (14)
  ▶ August (17)
  ▶ July (15)
  ▶ June (16)
  ▶ May (1)
  ▶ April (14)
  ▶ March (18)
  ▶ February (20)
  ▶ January (19)
▶ 2011 (380)
▶ 2010 (174)
▶ 2009 (68)
▶ 2008 (46)

Tweets by @rauschma

Axel Rauschm… @rauschma

Enjoying “Troll Hunters”:
– “Let’s call him ‘Gnome Chomsky’”
– “Juliet dies in this? Nooo!”

10h

Axel Rauschmayer Retweeted

Jonathan Creamer @jcreamer898

A nice little shout out to @rauschma… infoworld.com/article/316483 … #es2017 #async

JavaScr…
ECMASc…
infoworld…

14h

Axel Rauschmayer @rauschma

Not a physics book! twitter.com/ManningBooks/s …

02 Feb

Axel Rauschmayer Retweeted

Jordan Harband @ljharb

Making our React components forbid extra props has caught SO many bugs. I highly recommend it.

npmjs.com/airbnb-prop-ty…

npm: air…
Custom …
npmjs.com

02 Feb

Axel Rauschmayer Retweeted

Seth Petry-Johnson @spetryjohnson

Open strong. Be bold. Tell a story. Do something to get me interested. Opening w/ the "obligatory 'about me' slide" puts me to sleep :)

02 Feb

Google

▶ 2007 (12)

▶ 2006 (1)

▶ 2005 (2)

Labels

dev (592)
javascript (400)
computers (316)
life (194)
jslang (179)
esnext (156)
apple (107)
webdev (95)
mobile (83)
scitech (50)
hack (49)
mac (47)
google (39)
java (37)
ios (33)
business (32)
video (32)
clientjs (31)
hci (27)
entertainment (26)
nodejs (26)
society (26)
browser (25)
firefox (25)
html5 (24)
ipad (24)
movie (23)
psychology (22)
2ality (18)
tv (18)
android (17)
social (17)
chrome (16)
fun (16)
jsmodules (16)
tablet (16)
humor (15)
politics (15)
web (15)
cloud (14)
hardware (14)
microsoft (14)
software engineering (13)
blogging (12)
gaming (12)
eclipse (11)
gwt (11)
numbers (11)

• Reply • Share ›

huang47 • 4 years ago
the perf test cases are not created equally
"arr.length = 0" and "arr = []" both keep arr an instance of array
but "arr = null" and "delete arr" we can't operate arr like an array anymore
• Reply • Share ›

andres • 4 years ago
what is the performace for shallow copy using concat instead slice??...
• Reply • Share ›

Axel Rauschmayer Mod ➦ andres • 4 years ago
I had forgotten about slice (which is also the canonical way of turning `arguments` into an array). I've replaced concat with slice in the post, as I prefer the latter. My guess: it is faster.
• Reply • Share ›

emmecinque • 4 years ago
If you've got an API to which an array is passed, and one of the things the function may do is render the passed array empty, then you can't solve that by setting it to a new array - you have to set the length to zero.
• Reply • Share ›

Axel Rauschmayer Mod ➦ emmecinque • 4 years ago
Agreed! This is was I meant by “if all participants have to see the effect of clearing then this is the approach that you need to take”.
1 • Reply • Share ›

Subscribe **Add Disqus to your site Add Disqus Add** **Privacy**

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)

programming languages (11)
app store (10)
media (10)
nature (10)
security (10)
semantic web (10)
software (10)
twitter (10)
webos (10)
12quirks (9)
education (9)
jstools (9)
photo (9)
webcomponents (9)
windows 8 (9)
async (8)
idea (8)
iphone (8)
itunes (8)
scifi-fantasy (8)
app (7)
babel (7)
bookmarklet (7)
chromeos (7)
english (7)
es proposal (7)
fringe (7)
html (7)
jsint (7)
jsshell (7)
thunderbolt (7)
webapp (7)
advancedjs (6)
blogger (6)
crowdsourcing (6)
latex (6)
lion (6)
promises (6)
ted (6)
book (5)
environment (5)
gadget (5)
googleio (5)
intel (5)
jsarrays (5)
jshistory (5)
layout (5)
light peak (5)
michael j. fox (5)
music (5)
pdf (5)
polymer (5)
shell (5)
tc39 (5)

template literals (5)
underscorejs (5)
vlc (5)
__proto__ (4)
coffeescript (4)
concurrency (4)
dart (4)
facebook (4)
gimp (4)
googleplus (4)
health (4)
howto (4)
hp (4)
javafx (4)
kindle (4)
leopard (4)
macbook (4)
motorola (4)
münchen (4)
occupy (4)
pl fundamentals (4)
presenting (4)
publishing (4)
series (4)
textbook (4)
web design (4)
amazon (3)
asmjs (3)
back to the future (3)
bitwise_ops (3)
css (3)
es2016 (3)
flattr (3)
fluentconf (3)
food (3)
foreign languages (3)
house (3)
icloud (3)
info mgmt (3)
jsfuture (3)
jsstyle (3)
linux (3)
mozilla (3)
python (3)
regexp (3)
samsung (3)
tizen (3)
traffic (3)
typedjs (3)
unix (3)
adobe (2)
angry birds (2)
angularjs (2)
astronomy (2)

audio (2)

comic (2)

design (2)

dom (2)

ecommerce (2)

eval (2)

exploring es6 (2)

facebook flow (2)

facets (2)

flash (2)

free (2)

futurama (2)

guide (2)

history (2)

hyena (2)

internet explorer (2)

iteration (2)

journalism (2)

jquery (2)

jsengine (2)

jslib (2)

law (2)

lightning (2)

markdown (2)

math (2)

meego (2)

month (2)

nike (2)

nokia (2)

npm (2)

programming (2)

raffle (2)

repl (2)

servo (2)

sponsor (2)

steve jobs (2)

travel (2)

typescript (2)

usb (2)

winphone (2)

wwdc (2)

airbender (1)

amdefine (1)

aol (1)

app urls (1)

architecture (1)

atscript (1)

basic income (1)

biology (1)

blink (1)

bluetooth (1)

canada (1)

clip (1)

coding (1)

community (1)

cross-platform (1)

deutsch (1)

diaspora (1)

distributed-social-network (1)

dsl (1)

dvd (1)

dzone (1)

emacs (1)

emberjs (1)

energy (1)

esnext news (1)

esprop (1)

example (1)

facetator (1)

feedback (1)

firefly (1)

firefoxos (1)

fritzbox (1)

german (1)

git (1)

guest (1)

guice (1)

h.264 (1)

home entertainment (1)

hosting (1)

htc (1)

ical (1)

jsdom (1)

jsmyth (1)

library (1)

location (1)

marketing (1)

mars (1)

meta-data (1)

middle east (1)

mpaa (1)

msl (1)

mssurface (1)

netflix (1)

nsa (1)

obama (1)

openoffice (1)

opinion (1)

oracle (1)

organizing (1)

philosophy (1)

pixar (1)

pnacl (1)

prism (1)

privacy (1)

proxies (1)

puzzle (1)

raspberry pi (1)

- read (1)
- rodney (1)
- rust (1)
- safari (1)
- sponsoring (1)
- star trek (1)
- static generation (1)
- talk (1)
- technique (1)
- theora (1)
- thunderbird (1)
- typography (1)
- unicode (1)
- v8 (1)
- voice control (1)
- webassembly (1)
- webkit (1)
- webm (1)
- webpack (1)
- yahoo (1)