②ality – JavaScript and more

About | Donate | Subscribe | ES2017 | Books (free online!)

Free email newsletter: “ES.next News”

2013-05-30

JavaScript quirk 8: array-like objects

Labels: 12quirks, dev, javascript, jslang

[This post is part of a series on JavaScript quirks.]

Some objects in JavaScript look like arrays, but aren’t. They are called *array-like*. This blog post looks at what exactly that means and how to best work with those objects.

1. Array-like objects

An array-like object

- has: indexed access to elements and the property `length` that tells us how many elements the object has.
- does not have: array methods such as `push`, `forEach` and `indexOf`.

Two examples of array-like objects is the result of the DOM method `document.getElementsByClassName()` (many DOM methods return array-like objects) and the special variable `arguments` [1]. You can determine the number of arguments via

```
arguments.length
```

And you can access a single argument, e.g. read the first argument:

```
arguments[0]
```

Array methods, however, have to be borrowed. You can do that, because most of those methods are *generic*.

2. Generic methods

A generic method does not require `this` to be an array, it only requires `this` to have `length` and indexed element access. Normally, you invoke a method `m` on an array `arr` as follows.

```
arr.m(arg0, arg1, ...)
```

All functions have a method `call` that allows you to perform the above invocation differently:

```
Array.prototype.m.call(arr, arg0, arg1, ...)
```

The first argument of `call` is the value for `this` that `m` receives (in this case, `arr`). Because we access `m` directly and not via `arr`, we can now hand any `this` to that method. For example, `arguments`:

```
Array.prototype.m.call(arguments, arg0, arg1, ...)
```

2.1. Examples

Let’s continue with a concrete example. The following function `printArgs` logs all arguments that it receives.

```
function printArgs() {
    Array.prototype.forEach.call(arguments,
        function (arg, i) {
            console.log(i+'. '+arg);
        });
}
```

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)

Dr. Axel Rauschmayer

Free online books by Axel

Speaking JavaScript [up to ES5]
Exploring ES6

JavaScript training: Ecmanauten

- ► 2015 (65)
- ► 2014 (55)
- ▼ 2013 (98)
  - ► December (6)
  - ► November (3)
  - ► October (4)
  - ► September (6)
  - ► August (6)
  - ► July (8)
  - ► June (11)
  - ▼ May (11)
    - JavaScript quirk 8: array-like objects
    - The beginning of infinity in JavaScript
    - Plans for supporting Web Components in AngularJS a...
    - JavaScript quirk 7: inadvertent sharing of variabl...
    - Google's Polymer and the future of web UI framewor...
    - JavaScript quirk 6: the scope of variables
    - Netflix's technology and hiring practices
    - JavaScript history: undefined
    - Beyond "always on"
    - Pets in wheelchairs
    - JavaScript quirk 5: parameter handling
  - ► April (12)
  - ► March (8)
  - ► February (10)
  - ► January (13)
- ► 2012 (177)
- ► 2011 (380)
- ► 2010 (174)
- ► 2009 (68)
- ► 2008 (46)
- ► 2007 (12)
- ► 2006 (1)
- ► 2005 (2)

Labels

dev (592)

We have used method `forEach` generically. `printArgs` in use:

```
> printArgs()
> printArgs('a')
0. a
> printArgs('a', 'b')
0. a
1. b
```

You can even apply generic methods to ordinary objects:

```
> var obj = {};
> Array.prototype.push.call(obj, 'a');
1
> obj
{ '0': 'a', length: 1 }
```

In the above case, property `length` did not exist and was automatically created, with the initial value zero.

2.2. Converting an array-like object to an array

Sometimes the best way to work with an array-like object is to convert it to an array. That can also be done via a generic method:

```
Array.prototype.slice.call(arguments)
```

Compare: to create a copy of an array `arr`, you make the method call

```
arr.slice()
```

3. Reference

[1] JavaScript quirk 5: parameter handling

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

- 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)
- programming languages (11)
- app store (10)
- media (10)
- nature (10)
- security (10)
- semantic web (10)

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

Recommend 2 **Share** **Sort by Best**

Join the discussion…

Priit Pirita • 4 years ago

There's more quirks :-)

Some array-like objects, like NodeList, have one method that Array does not have - item(). And it has it's own quirks - foo[1000] if not exist, returns undefined, but foo.item(1000) if not exists, returns null.
And there's more, in IE8 all array-like objects can not be converted into Array with slice() - getElementsByTagName() for example.

5 ^ | v • Reply • Share ›

Matthias • 3 years ago

Another way to convert an array-like object to an array (and easier to remember in my opinion) is to use the Array constructor and apply it with the object:

Array.apply(null, arguments);

The use of Array.prototype.slice always seems like a hack to me. From its name you would not guess that it creates an Array. The Array constructor, on the other hand, is just meant to do that. And you save nine characters too!

^ | v • Reply • Share ›

Axel Rauschmayer Mod → Matthias • 3 years ago

I agree with you on the “seem like a hack”, but your approach does not work if arguments has a single element that is a positive integer:

```
$ function arrify() { return Array.apply(null, arguments) }
$ arrify(3)
[ , ,  ]
```

Reason: `Array(3)` creates an empty array of length 3 and not `[3]`.

Also: you can write `[].slice.call(arguments)` if you want to save characters.

2 ^ | v • Reply • Share ›

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

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)

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)