

# ②ality – JavaScript and more

| [About](#) | [Donate](#) | [Subscribe](#) | [ES2017](#) | [Books \(free online!\)](#) |

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)

Free email newsletter: “[ES.next News](#)”

2015-11-03

## Synchronous and asynchronous sequential execution of functions

Labels: [async](#), [dev](#), [esnext](#), [javascript](#)

This blog post examines three ways of executing function sequentially:

- Synchronously
- Asynchronously, via Promises
- Asynchronously, via the library co

### 1. Synchronously

Synchronous sequential execution is built into JavaScript and looks like this:

```
function func() {  
    foo();  
    bar();  
    baz();  
}
```

### 2. Asynchronously, via Promises

To execute Promise-based functions sequentially, you need to chain function calls via `.then()`, which is the Promise equivalent of the semicolon:

```
/**  
 * Returns a Promise that resolves to `undefined`.  
 */  
function func() {  
    return foo()  
        .then(() => bar())  
        .then(() => baz());  
}
```

If you are OK with executing the functions in an arbitrary order (the single-threaded version of “in parallel”), you can use `Promise.all()`:

```
/**  
 * Returns a Promise that resolves to  
 * [undefined, undefined, undefined].  
 */  
function func() {  
    return Promise.all([foo(), bar(), baz()]);  
}
```

### 3. Asynchronously, via the library co

The [library co](#) also works with Promise based functions. Its helper method, `co.wrap()`, converts a generator function `gen` into a function that returns a promisification of whatever `gen` returns. The neat thing is that `gen` can yield a Promise (e.g. the result of a function call) and is suspended while the Promise is pending. If the Promise is fulfilled,

(Ad, please don't block.)



90% Unlimited  
Downloads Choose from  
Over 300,000 Vectors,  
Graphics & Photos.  
ads via Carbon



Dr. Axel Rauschmayer

Free online  
books by Axel

[Speaking JavaScript](#)  
[up to ES5]

[Exploring ES6](#)

[JavaScript training:](#)  
Ecmascripten

- ▼ 2015 (65)
    - December (7)
    - ▼ November (6)
      - Configuring Babel 6
    - ES proposal: Trailing commas in function parameter...
    - ES proposal: string padding
    - ES proposal: Object.entries() and Object.values()
  - The TC39 process for ECMAScript features
  - Synchronous and asynchronous sequential execution ...
  - October (13)
  - September (5)
  - August (6)
  - July (3)
  - June (3)
  - April (4)
  - March (4)
  - February (8)
  - January (6)
  - 2014 (55)
  - 2013 (98)
  - 2012 (177)
  - 2011 (380)
  - 2010 (174)
  - 2009 (68)
  - 2008 (46)
  - 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)

the fulfillment value becomes the result of `yield`. If the Promise is rejected, the rejection value is thrown as an exception.

Let's look at an example:

```
/**  
 * The callback implicitly returns `undefined`.  
 * Therefore, `func` returns a Promise that resolves to  
 * `undefined`.  
 */  
  
const func = co.wrap(function* () {  
    yield foo();  
    yield bar();  
    yield baz();  
});
```

`co` works very much like [async functions](#), a proposed ECMAScript feature (to appear in ES2016 or later), without being much more verbose. I prefer it to transpiling `async` functions via Babel, because their syntax may still change.

## 4. Further reading

- [Async Functions for ECMAScript](#) (ECMAScript proposal)
- “[Promises for asynchronous programming](#)” (chapter in “Exploring ES6”)
  - “[Asynchronous programming \(background\)](#)” (background for the chapter on Promises)

8 Comments    The 2ality blog

Login ▾

Recommend 2    Share

Sort by Best ▾

Join the discussion...

Slava Ganzin • a year ago  
Maybe add <https://github.com/caolan/asyn...>

2 ⌂ ⌄ • Reply • Share ↗

bradleymeck • a year ago  
A good note is that task runners like `'co'` allow for more flexibility than `'async'` functions since you can specify custom changes in scheduling like how `'galaxy'` has interrupts and `'generator-runner'` has out of band aborts.

1 ⌂ ⌄ • Reply • Share ↗

jaya krishna • 7 months ago  
var num = [1,2,3,4,5,6,7,8,9];  
  
var stop = false;  
  
function countFront (){  
  
for(var i=0;i<num.length;i++){ console.log(num[i]);="" if(num[i]==="3="" ||="" stop=="true"){="" stop="true;" break;="" }="" }="" }="" countfront="" ()="" ;="" function="" countrev="" ()="" {="" for(var="" j="num.length-1;j>=0;j--){"}  
  
console.log(num[j]);  
  
if(num[j] == 3 || stop == true){  
  
stop = true;  
  
break;  
  
}

[see more](#)

## Tweets by @rauschma

Axel Rauschmayer  
@rauschma

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

9h

Axel Rauschmayer  
Retweeted

Jonathan Creamer  
@jcreamer898

A nice little shout out to  
@rauschma...  
[infoworld.com/article/316483](http://infoworld.com/article/316483)  
... #es2017 #async

JavaScr...  
ECMASc...  
infoworl...

13h

Axel Rauschmayer  
@rauschma

Not a physics book!  
[twitter.com/ManningBooks/s/](http://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...](http://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

[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)  
[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)

^ | v • Reply • Share >



**Piter Kater** • a year ago

Hi Axel. This is not about this blog post. Is related to recent post I do on stackoverflow. Here is: <http://stackoverflow.com/quest...>

Can you make me any advice on this or better, make a post about it ??  
Thanks

^ | v • Reply • Share >



**Matteo Ronchi** • a year ago

co is great but often I prefer to use Bluebird coroutines, mostly because I'm already using bluebird for promises. Have you tried it?

<http://bluebirdjs.com/docs/api...>

^ | v • Reply • Share >



**Fayabomb** • a year ago

what about setTimeout(function,ms)?  
seems to be the easiest and quickest way to execute asynchronously in javascript.

^ | v • Reply • Share >



**Alex** [ ✯ ] • a year ago

foo.pipe(bar).pipe(baz)

Would that work too?

^ | v • Reply • Share >



**Sebastian Schürmann** • a year ago

Didnt knew co. Will give it a spin.

^ | v • Reply • Share >



Subscribe



Add Disqus to your site

Add Disqus Add



Privacy

[Newer Post](#)

[Home](#)

[Older Post](#)

Subscribe to: [Post Comments \(Atom\)](#)

-----  
**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)  
pi 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)

Powered by [Blogger](#).