What’s Next?
The Future of Node.js

•••

Michael Dawson | Beth Griggs | Joe Sepi

Joe Sepi

- Open Source Engineer & Advocate, IBM
- Active in OpenJS foundation
  - Cross Project Council Chairperson
  - Node.js voting representative in CPC
- Active Node.js community member
  - Node.js Community Committee member
  - Currently active in the Tooling group/work

@joesepi

IBM joesepi@ibm.com

@joe_sepi

Michael Dawson

- Node.js lead for Red Hat and IBM
- Active Node.js community member
  - Node.js Collaborator, Node.js Technical Steering Committee, Community Committee member
  - Active in a number of Working group(s)
- Active in OpenJS foundation
  - Voting Cross Project Council Member
  - Community Director 2020-2021

@mhdawson midawson@redhat.com @mhdawson1

Bethany Griggs

- Senior Software Engineer at Red Hat
- Active Node.js Community Member
  - Node.js Technical Steering Committee Member
  - Node.js Release Working Group
  - Node.js Package Maintenance Working Group

@BethGriggs bgriggs@redhat.com @BethGriggs_

Agenda

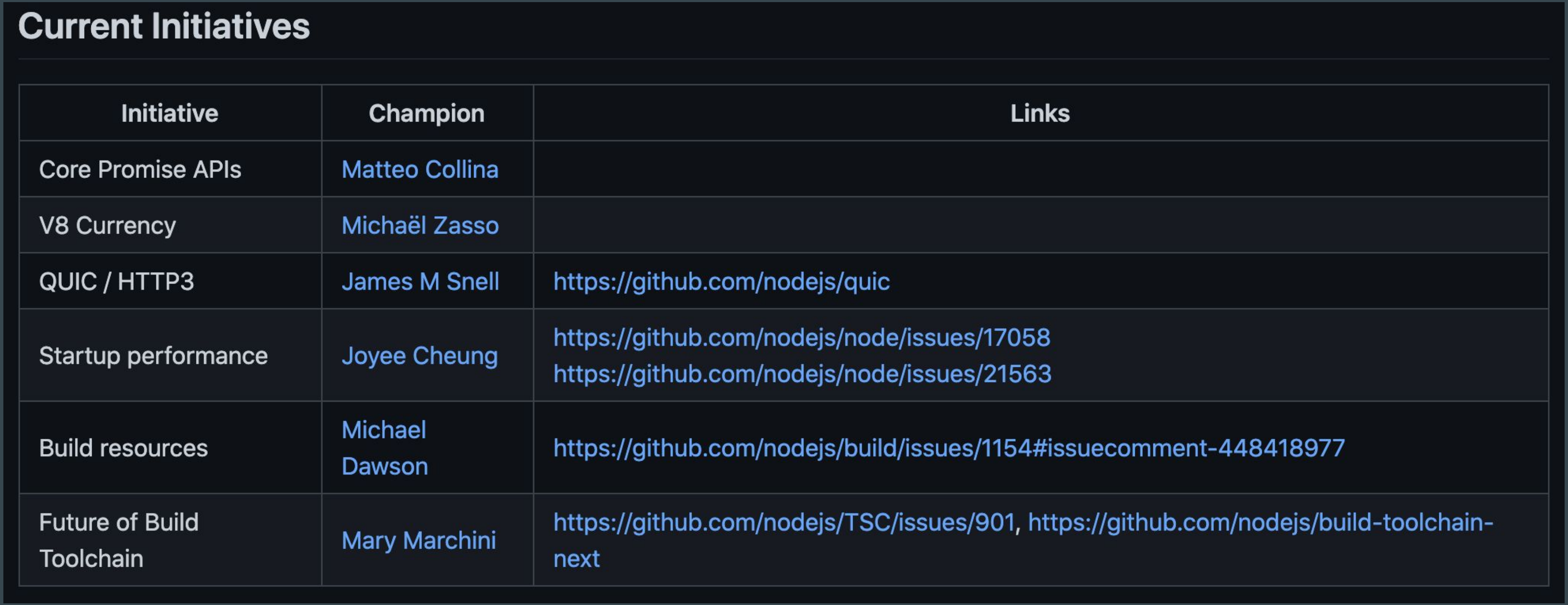
- How can you track ‘What’s next’?
    - Strategic Initiatives and Working Groups
    - Next-10
    - Node.js Releases
- New and Experimental Features
- Will Fetch happen?
- Getting Involved

What’s next?

Working Groups and Teams

- Build Working Group
- Diagnostics Working Group
- Docker Working Group
- I18n Working Group
- Issue Triage Team
- Next-10 Team
- Node-API Team
- Release Working Group
- Package Maintenance Working Group
- Streams Working Group
- Tooling Team

Strategic Initiatives - Technical Steering Committee

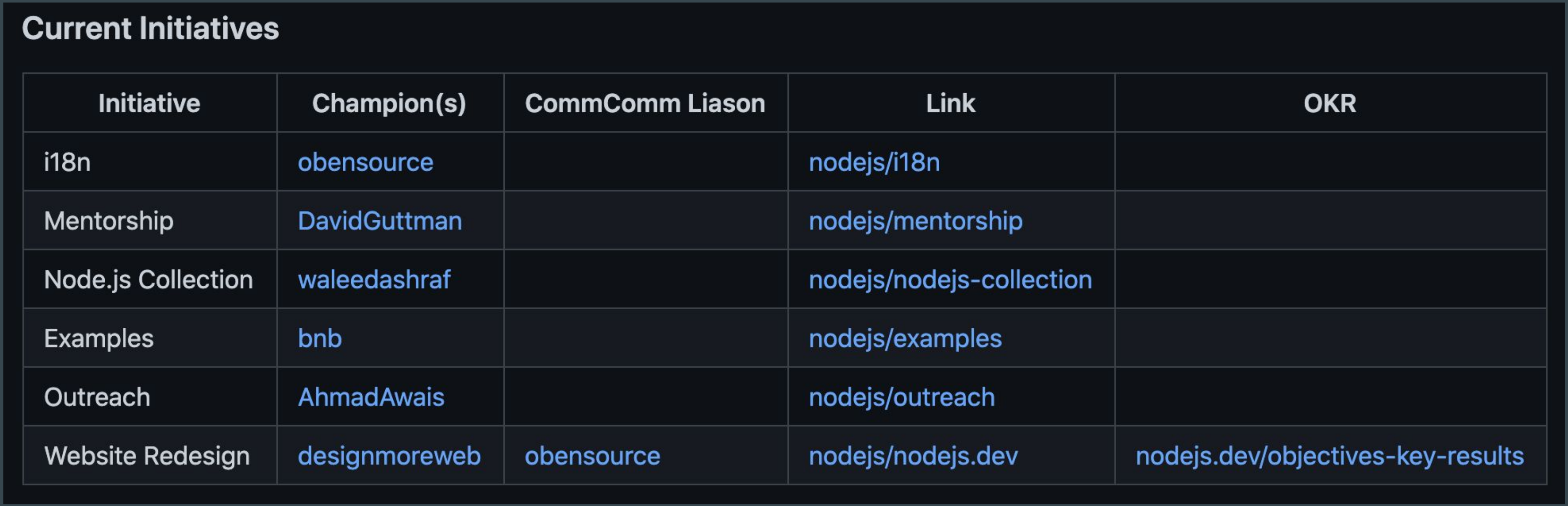


Current Initiatives

Initiative	Champion	Links
Core Promise APIs	Matteo Collina	
V8 Currency	Michaël Zasso	
QUIC / HTTP3	James M Snell	https://github.com/nodejs/quic
Startup performance	Joyee Cheung	https://github.com/nodejs/node/issues/17058 https://github.com/nodejs/node/issues/21563
Build resources	Michael Dawson	https://github.com/nodejs/build/issues/1154#issuecomment-448418977
Future of Build Toolchain	Mary Marchini	https://github.com/nodejs/TSC/issues/901, https://github.com/nodejs/build-toolchain-next

https://github.com/nodejs/TSC/blob/main/Strategic-Initiatives.md

Strategic Initiatives - Community Committee



Current Initiatives

Initiative	Champion(s)	CommComm Liason	Link	OKR
i18n	obensource		nodejs/i18n	
Mentorship	DavidGuttman		nodejs/mentorship	
Node.js Collection	waleedashraf		nodejs/nodejs-collection	
Examples	bnb		nodejs/examples	
Outreach	AhmadAwais		nodejs/outreach	
Website Redesign	designmoreweb	obensource	nodejs/nodejs.dev	nodejs.dev/objectives-key-results

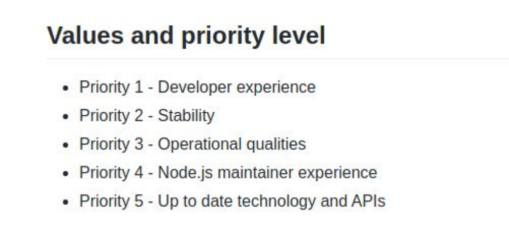
https://github.com/nodejs/community-committee#current-initiatives

Next 10

- Strategic directions for the next 10 years of Node.js (Building on the successful first 10)
- Laying the Foundation
  - Technical Values
  - Constituencies
  - Needs of the Constituencies
- Next - dive into Technical priorities

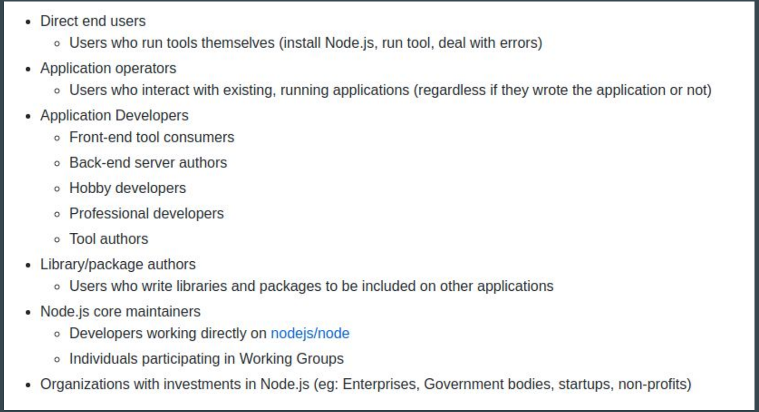
Join us in

https://github.com/nodejs/next-10



Values and priority level

- Priority 1 - Developer experience
- Priority 2 - Stability
- Priority 3 - Operational qualities
- Priority 4 - Node.js maintainer experience
- Priority 5 - Up to date technology and APIs



- Direct end users
  - Users who run tools themselves (install Node.js, run tool, deal with errors)
- Application operators
  - Users who interact with existing, running applications (regardless if they wrote the application or not)
- Application Developers
  - Front-end tool consumers
  - Back-end server authors
  - Hobby developers
  - Professional developers
  - Tool authors
- Library/package authors
  - Users who write libraries and packages to be included on other applications
- Node.js core maintainers
  - Developers working directly on nodejs/node
  - Individuals participating in Working Groups
- Organizations with investments in Node.js (eg: Enterprises, Government bodies, startups, non-profits)

Next 10

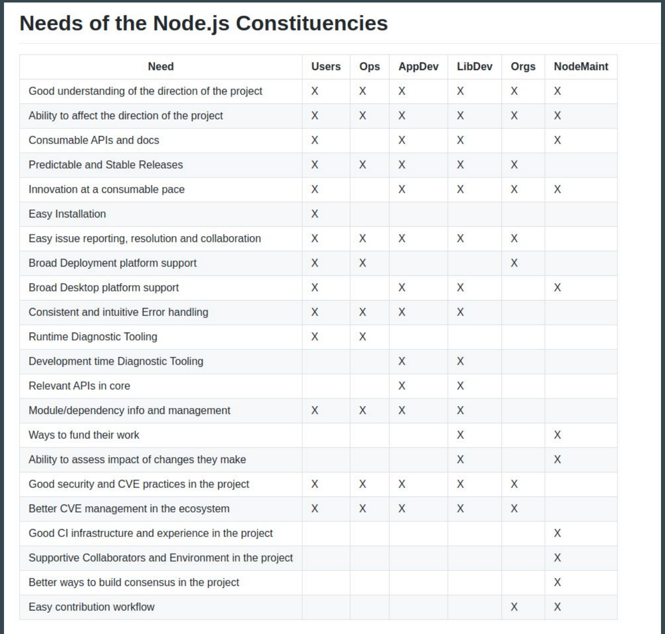
Help us get the foundation right

Survey

https://www.surveymonkey.com/r/86SSY9Q

Background

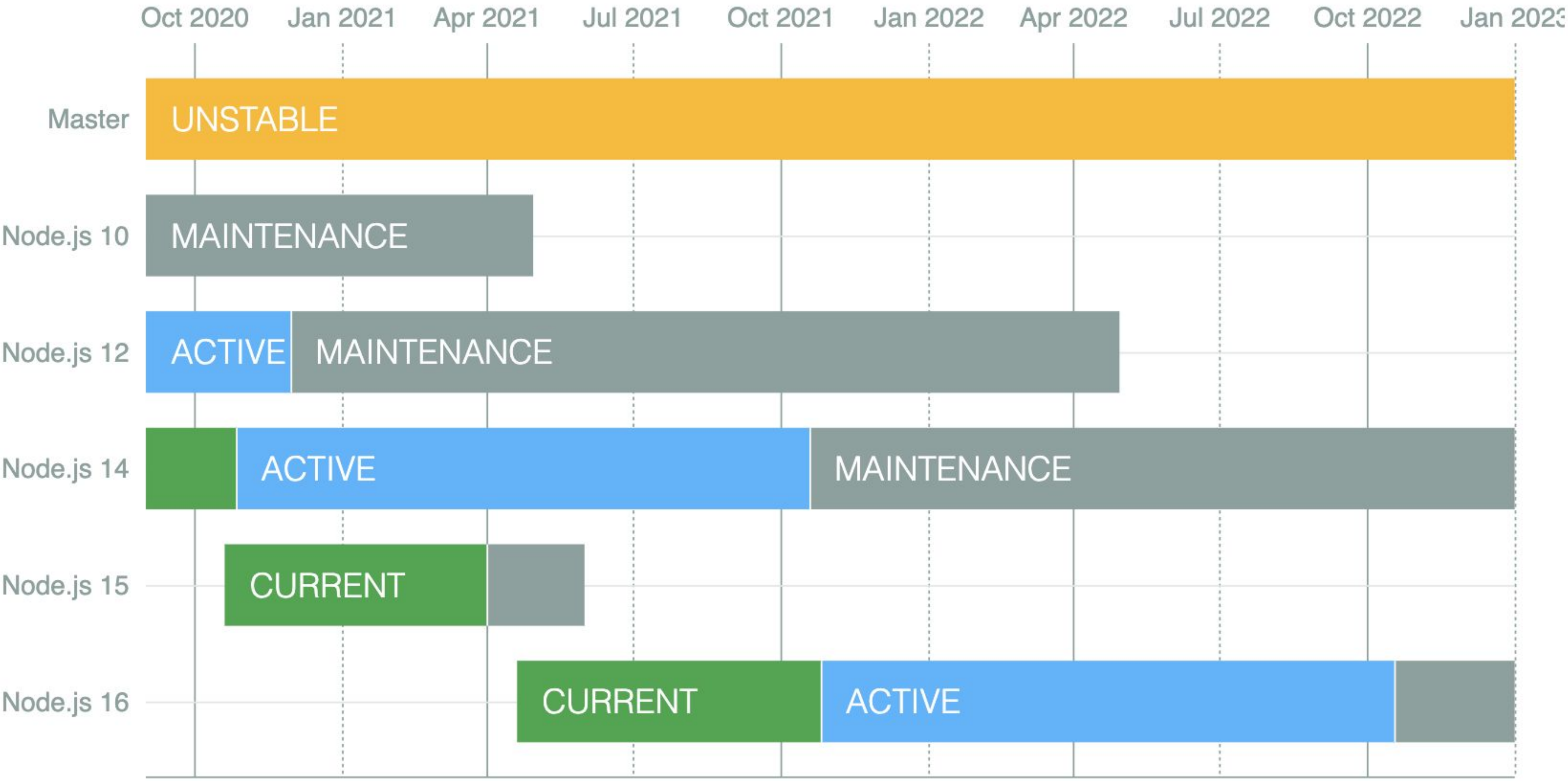
https://nodejs.medium.com/next-10-years-of-node-js-understanding-the-needs-of-the-node-js-constituencies-2f95a1df6a6f

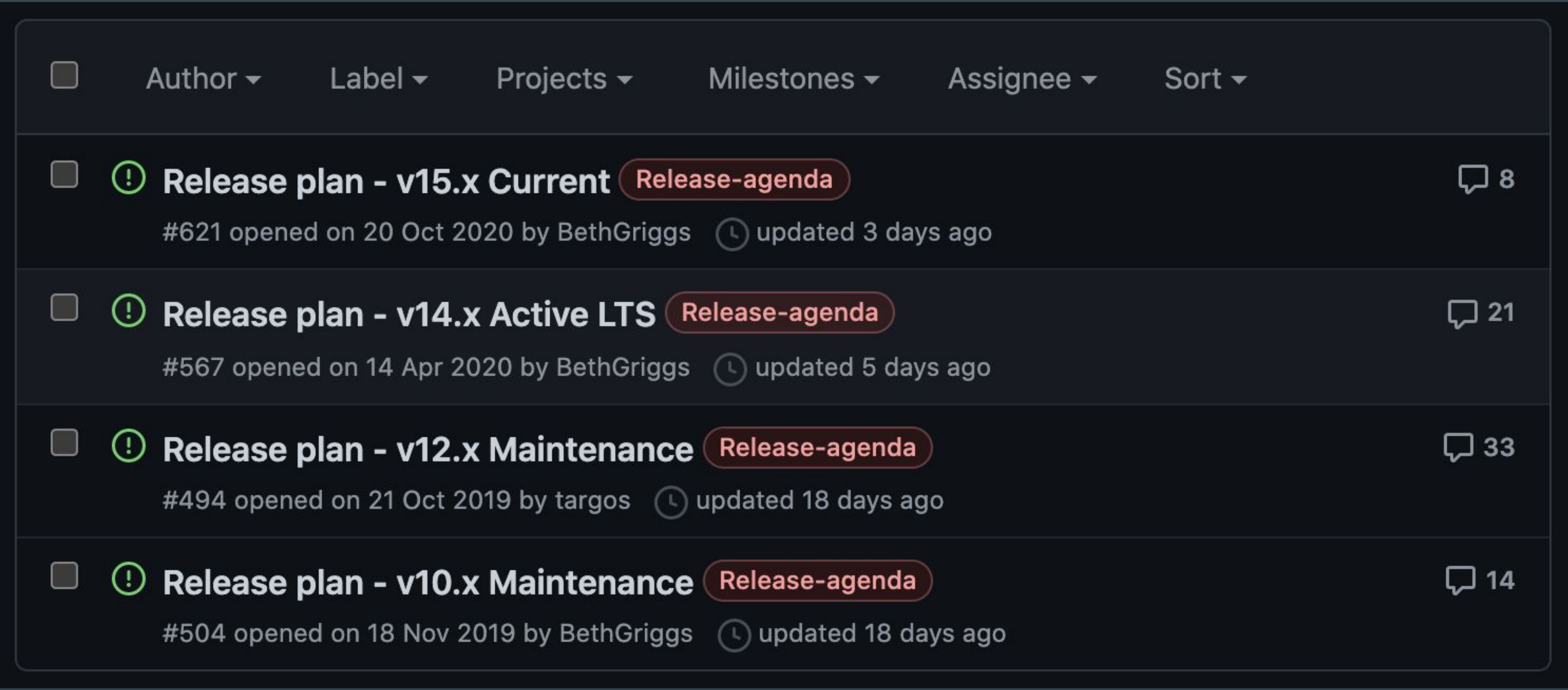


Needs of the Node.js Constituencies

Need	Users	Ops	AppDev	LibDev	Orgs	NodeMaint
Good understanding of the direction of the project	X	X	X	X	X	X
Ability to affect the direction of the project	X	X	X	X	X	X
Consumable APIs and docs	X		X	X		X
Predictable and Stable Releases	X	X	X	X	X	
Innovation at a consumable pace	X		X	X	X	X
Easy Installation	X					
Easy issue reporting, resolution and collaboration	X	X	X	X	X	
Broad Deployment platform support	X	X			X	
Broad Desktop platform support	X		X	X		X
Consistent and intuitive Error handling	X	X	X	X		
Runtime Diagnostic Tooling	X	X				
Development time Diagnostic Tooling			X	X		
Relevant APIs in core			X	X		
Module/dependency info and management	X	X	X	X		
Ways to fund their work				X		X
Ability to assess impact of changes they make				X		X
Good security and CVE practices in the project	X	X	X	X	X	
Better CVE management in the ecosystem	X	X	X	X	X	
Good CI infrastructure and experience in the project						X
Supportive Collaborators and Environment in the project						X
Better ways to build consensus in the project						X
Easy contribution workflow					X	X

Releases


Oct 2020
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Master
UNSTABLE
Node.js 10
MAINTENANCE
Node.js 12
ACTIVE
MAINTENANCE
Node.js 14
ACTIVE
MAINTENANCE
Node.js 15
CURRENT
Node.js 16
CURRENT
ACTIVE



□ Release plan - v15.x Current Release-agenda 8
#621 opened on 20 Oct 2020 by BethGriggs updated 3 days ago

□ Release plan - v14.x Active LTS Release-agenda 21
#567 opened on 14 Apr 2020 by BethGriggs updated 5 days ago

□ Release plan - v12.x Maintenance Release-agenda 33
#494 opened on 21 Oct 2019 by targos updated 18 days ago

□ Release plan - v10.x Maintenance Release-agenda 14
#504 opened on 18 Nov 2019 by BethGriggs updated 18 days ago

https://github.com/nodejs/Release

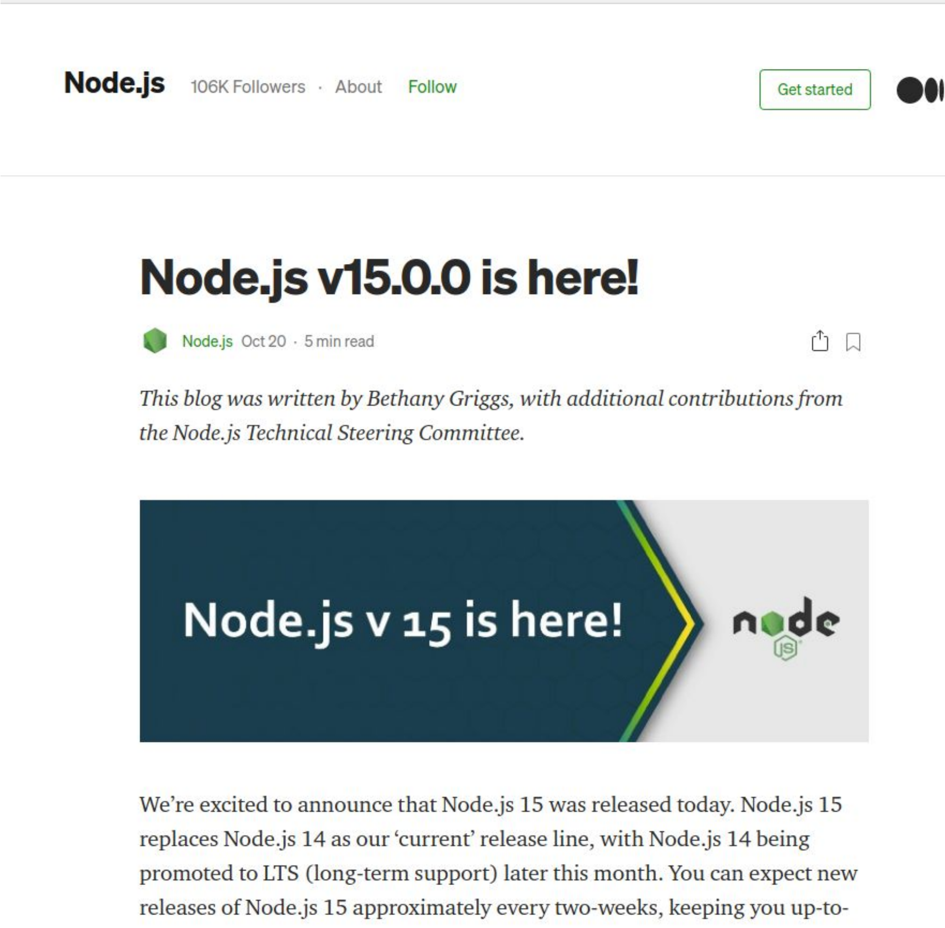
Node.js 16

20th April 2021

https://nodejs.org/download/rc/v16.0.0-rc.0/

Major Releases?

- Why do “Majors” seem so boring?
- Interesting features often Minor
- Features flow backwards quickly
- By the time we have a Major they are “old” news
- Still important -> Transition to LTS !



Node.js v15.0.0 is here!

This blog was written by Bethany Griggs, with additional contributions from the Node.js Technical Steering Committee.

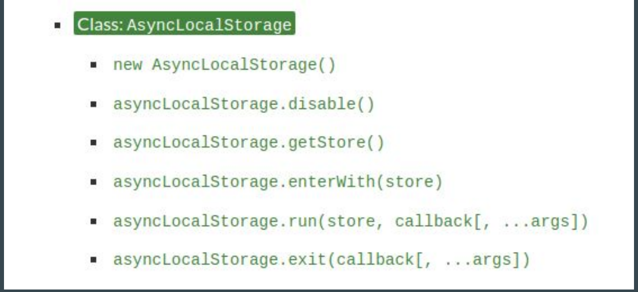
We’re excited to announce that Node.js 15 was released today. Node.js 15 replaces Node.js 14 as our ‘current’ release line, with Node.js 14 being promoted to LTS (long-term support) later this month. You can expect new releases of Node.js 15 approximately every two-weeks, keeping you up-to-

https://nodejs.medium.com/node-js-v15-0-0-is-here-deb00750f278

New Features

AsyncLocalStorage (Experimental)

- Need to share data across asynchronous calls
  - Transaction traces, APMs
- Easier for synchronous thread based runtimes (ex ThreadLocal)
- PR to move key parts to stable (new, run, getStore)
  - https://github.com/nodejs/node/pull/37675



- Class: `AsyncLocalStorage`
  - `new AsyncLocalStorage()`
  - `asyncLocalStorage.disable()`
  - `asyncLocalStorage.getStore()`
  - `asyncLocalStorage.enterWith(store)`
  - `asyncLocalStorage.run(store, callback[, ...args])`
  - `asyncLocalStorage.exit(callback[, ...args])`

AsyncLocalStorage (Experimental)

```
const AsyncLocalStorage = require('async_hooks').AsyncLocalStorage;
const asyncLocal1 = new AsyncLocalStorage();
let   sharedFlowId = "";

function asyncflow(flowid) {
  setTimeout( () => {
    asyncLocal1.run(new Object(), () => {
      sharedFlowId = flowid;
      asyncLocal1.getStore()['flowid'] = flowid;
      setTimeout(() => {
        console.log(flowid + ':' + sharedFlowId + ':' +
asyncLocal1.getStore()['flowid']);
      }, Math.random()*500);
    });
  }, Math.random()*500);
}

asyncflow('flow1');
asyncflow('flow2');
asyncflow('flow3');
```

Same object as in run

Output - sharedFlowID varies

```
------------------------------
flow2:flow1:flow2
flow3:flow3:flow3
flow1:flow3:flow1
```

https://nodejs.org/dist/latest-v15.x/docs/api/async_hooks.html

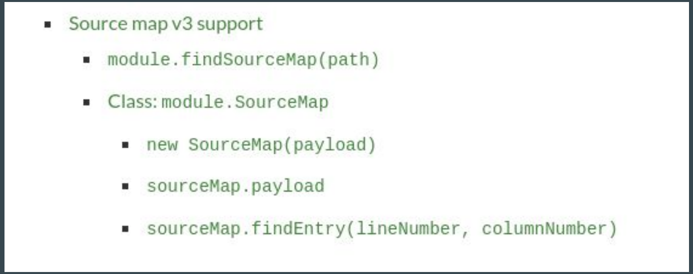
Source Maps

- Source code does not always match what’s running
  - Transpilers (ex Babel)
  - Flow, TypeScript, CoffeeScript, etc.
- Causes problems for
  - Observability
  - Code Coverage
  - …
- Source Map V3
  - - https://sourcemaps.info/spec.html
  - **Embedded in source with special comment -** `//# sourceMappingURL=test2-babel.js.map`

Same file with running source + .map

Source Maps

- What support does Node.js have so far?
  - Environment variable - NODE_V8_COVERAGE (v12.11.0)
  - --enable-source-maps (v12.12.0) - incomplete
  - Public API - https://nodejs.org/api/module.html (v13.7.0)

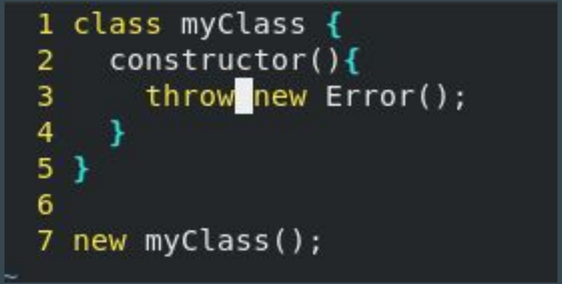


- Source map v3 support
  - `module.findSourceMap(path)`
  - Class: `module.SourceMap`
    - `new SourceMap(payload)`
    - `sourceMap.payload`
    - `sourceMap.findEntry(lineNumber, columnNumber)`

To read more, great blog by @bcoe
https://nodejs.medium.com/source-maps-in-node-js-482872b56116

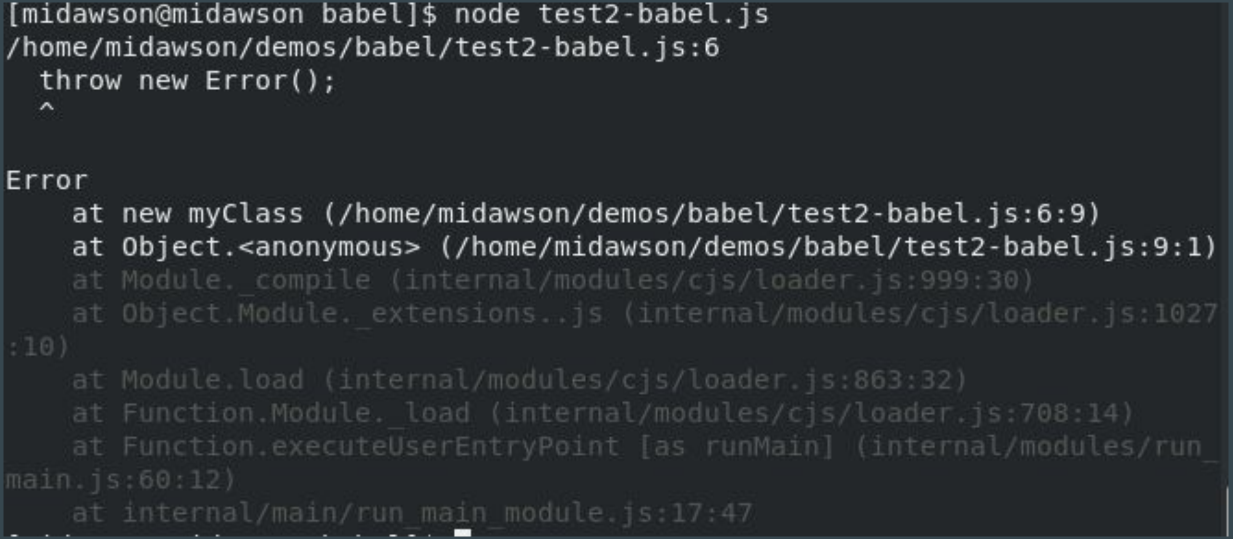
Source maps - Experimental

By default



```
class myClass {
  constructor(){
    throw new Error();
  }
}

new myClass();
```

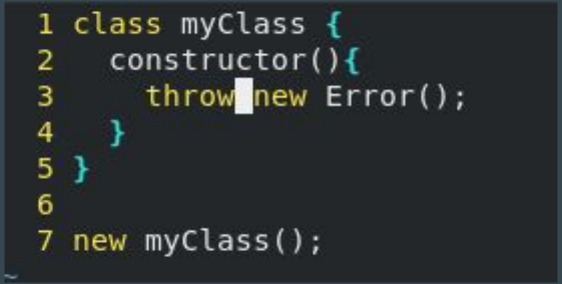


```
[midawson@midawson babel]$ node test2-babel.js
/home/midawson/demos/babel/test2-babel.js:6
  throw new Error();
  ^

Error
    at new myClass (/home/midawson/demos/babel/test2-babel.js:6:9)
    at Object.<anonymous> (/home/midawson/demos/babel/test2-babel.js:9:1)
    at Module._compile (internal/modules/cjs/loader.js:999:30)
    at Object.Module._extensions..js (internal/modules/cjs/loader.js:1027
:10)
    at Module.load (internal/modules/cjs/loader.js:863:32)
    at Function.Module._load (internal/modules/cjs/loader.js:708:14)
    at Function.executeUserEntryPoint [as runMain] (internal/modules/run_
main.js:60:12)
    at internal/main/run_main_module.js:17:47
```

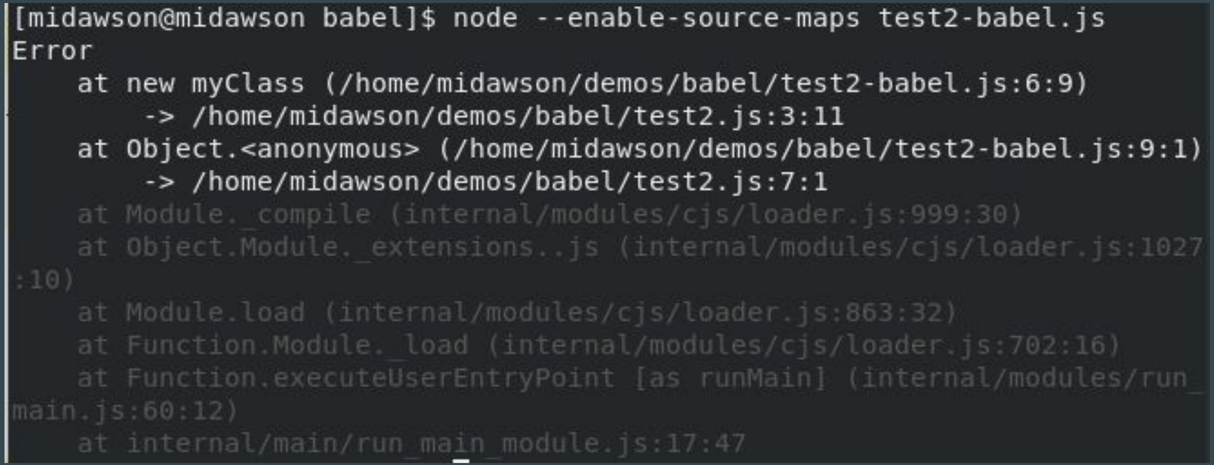
Source maps - Experimental

With --enable-source-maps



```
class myClass {
  constructor(){
    throw new Error();
  }
}

new myClass();
```



```
[midawson@midawson babel]$ node --enable-source-maps test2-babel.js
Error
    at new myClass (/home/midawson/demos/babel/test2-babel.js:6:9)
        -> /home/midawson/demos/babel/test2.js:3:11
    at Object.<anonymous> (/home/midawson/demos/babel/test2-babel.js:9:1)
        -> /home/midawson/demos/babel/test2.js:7:1
    at Module._compile (internal/modules/cjs/loader.js:999:30)
    at Object.Module._extensions..js (internal/modules/cjs/loader.js:1027
:10)
    at Module.load (internal/modules/cjs/loader.js:863:32)
    at Function.Module._load (internal/modules/cjs/loader.js:702:16)
    at Function.executeUserEntryPoint [as runMain] (internal/modules/run_
main.js:60:12)
    at internal/main/run_main_module.js:17:47
```

ECMAScript Modules

CommonJS

```
module.exports = function() {
  console.log('Hello');
}

const test = require('./helloTest.js');
test();
```

ES6 Modules

```
export function test() {
  console.log('Hello ES6');
}

import { test } from './helloTest.mjs';
test();
```

`package.json`

```
{
  "name": "test",
  "type": "module"
}
```

⇨

```
import { test } from './helloTest2.js';
test();
```

https://nodejs.org/api/esm.html

ECMAScript Modules - Experimental APIs

- Loaders API:

```
$ node --experimental-loader ./custom-loader.mjs ./main.mjs
```

- JSON Modules:

```
import packageConfig from './package.json';
```

- WASM Modules:

```
import * as M from './module.wasm';
```

ECMAScript Modules - Top-level await

- Dynamic dependency pathing:

```
const strings = await import(`/i18n/${navigator.language}`);
```

- Resource initialization:

```
const connection = await dbConnector();
```

- Dependency fallbacks:

```
try {
 jQuery = await import('https://cdn-a.com/jQuery');
} catch {
 jQuery = await import('https://cdn-b.com/jQuery');
}
```

Timers Promises API

```
import { setTimeout } from 'timers/promises';

async function run() {
   await setTimeout(5000);
   console.log('Hello from NearForm Presents!');
}

run();
```

https://nodejs.org/api/timers.html#timers_timers_promises_api

What about Fetch?

James Snell's talk earlier today touches on Fetch, but in summary:

- Fetch is a lot of work (!)
- Fetch needs underlying work to be done first

However...

- Fetch is being planned
- Fetch will get done! (perhaps even this year)

OpenJS Foundation

openjsf.org

How to get involved?

- NodeTodo.org
- Triager
- Build helper
- Teams, working groups and initiatives
  - Website Redesign
  - i18n
  - Tooling
  - Release team and backporting
  - Web frameworks
  - Package Maintenance
  - Etc...
- Slack / IRC

What’s next?

It’s what you make it!

Next 10 Survey

https://www.surveymonkey.com/r/86SSY9Q

@nodejs