NearForm RESEARCH

The State of Node.js

October 2019

James M Snell / jasnell
Head of Research, NearForm

A Quick Review

There have been just under 50 Node.js Releases in 2019

Node.js 12.0.0 was released in April

Node.js 12.13.0 moved to Active LTS this week.

Node.js 13.0.0 was released this week.

A Quick Review

There have been 3,186 Pull Requests opened so far this year (averaging about 74 per week)

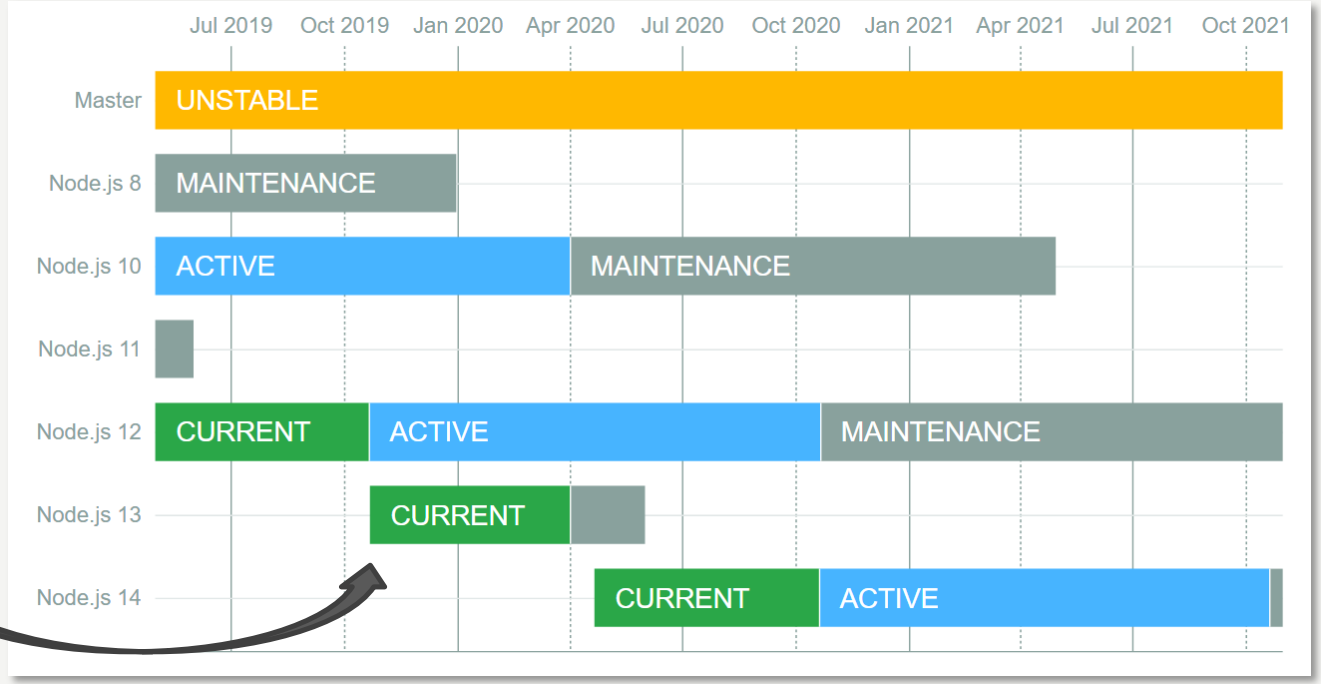
There are now over 2,500 contributors, 116 maintainers, and 20 Technical Steering Committee Members

The Node.js Foundation merged with the JS Foundation to form the new OpenJS Foundation

Node.js Certification

The OpenJS Foundation, in partnership with NodeSource and NearForm, has officially launched the Node.js Certification.

Long Term Support Updates

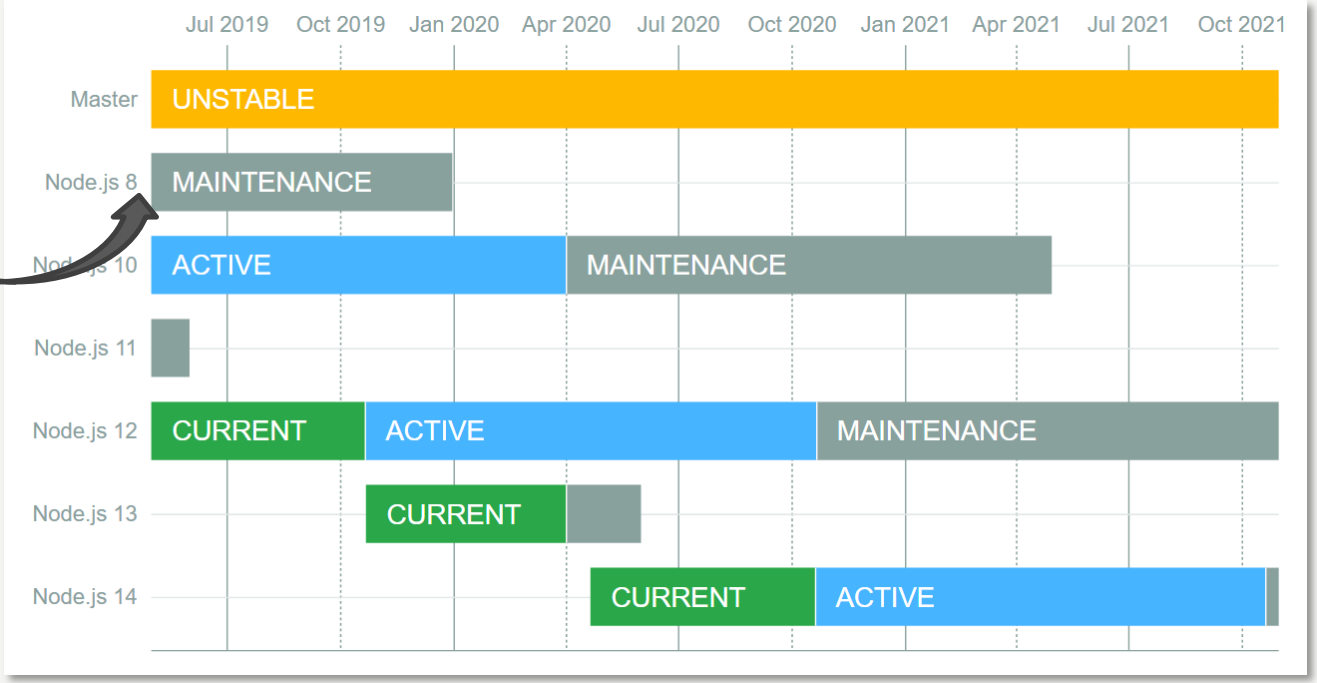


We are here

Long Term Support Updates

If you are using Node.js 8, upgrade soon!

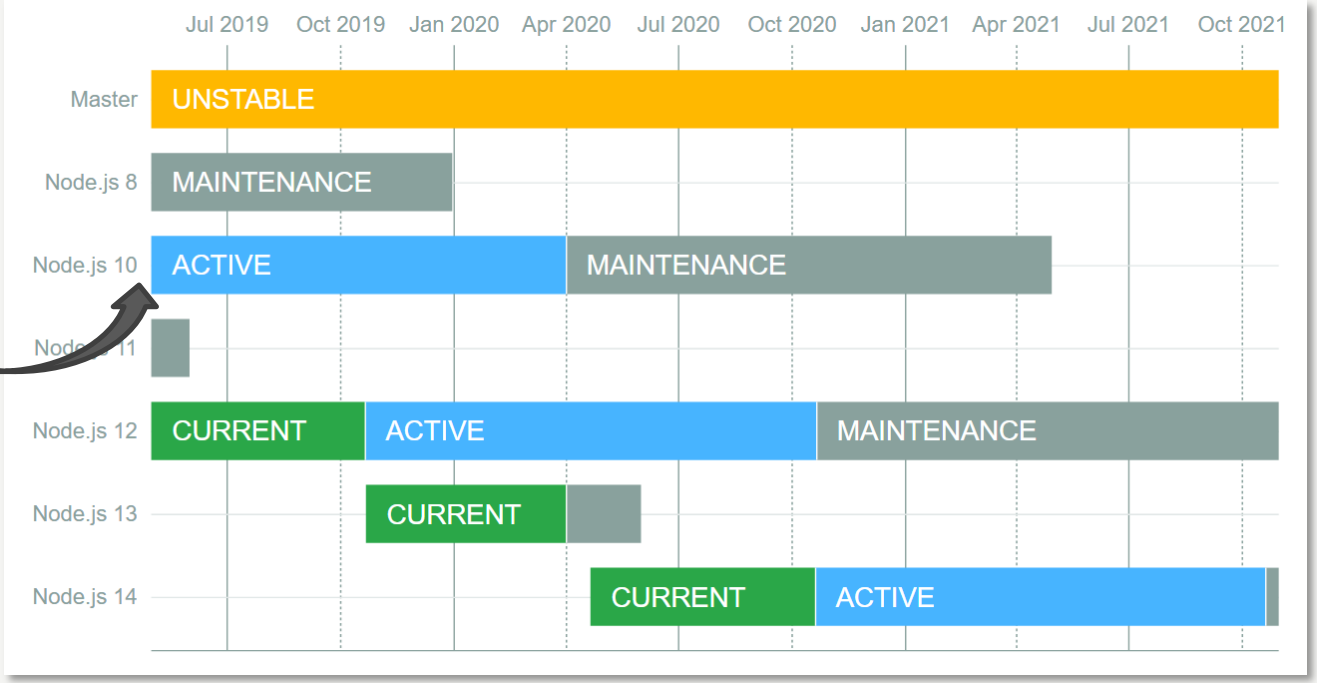
End-of-Life at the end of year.



Long Term Support Updates

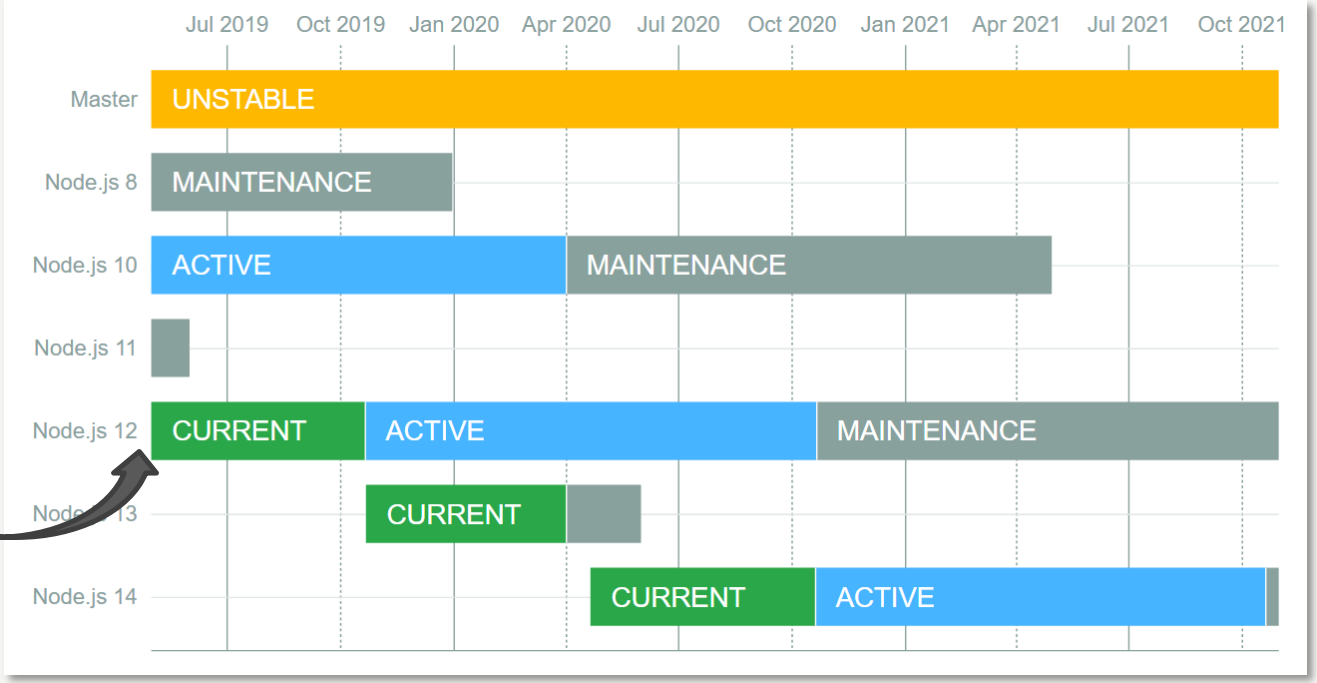
If you are using Node.js 10 Start Upgrading soon

Maintenance only in April 2020



Long Term Support Updates

Yeah, Node.js 13 is cool, but if you're a business, Node.js 12 should be your target.



NearForm RESEARCH

Let’s talk about Node.js 13

Node.js now includes Full ICU/Intl Data

```
const january = new Date(9e8);
const spanish = new Intl.DateTimeFormat('es', { month: 'long' });
console.log(spanish.format(january))  // 'enero';
```

Worker Threads are no longer experimental.

```
const {
  Worker, isMainThread, parentPort, workerData
} = require('worker_threads');

if (isMainThread) {
  module.exports = function parseJSAsync(script) {
    return new Promise((resolve, reject) => {
      const worker = new Worker(__filename, {
        workerData: script
      });
      worker.on('message', resolve);
      worker.on('error', reject);
      worker.on('exit', (code) => {
        if (code !== 0)
          reject(new Error(`Worker stopped with exit code ${code}`));
      });
    });
  };
} else {
  const { parse } = require('some-js-parsing-library');
  const script = workerData;
  parentPort.postMessage(parse(script));
}
```

Workers are effectively new Node.js instances running in a separate thread.

Promises supported in EventEmitter

```
const { once, EventEmitter } = require('events');

async function run() {
  const ee = new EventEmitter();

  process.nextTick(() => {
    ee.emit('myevent', 42);
  });

  const [value] = await once(ee, 'myevent');
  console.log(value);

  const err = new Error('kaboom');
  process.nextTick(() => { ee.emit('error', err); });

  try {
    await once(ee, 'myevent');
  } catch (err) {
    console.log('error happened', err);
  }
}

run();
```

Currently only waiting for one-time events is supported.

Support for Repeated events using an async generator is under development.

TLS Keylogging and Tracing Enabled

(Use Wireshark!)

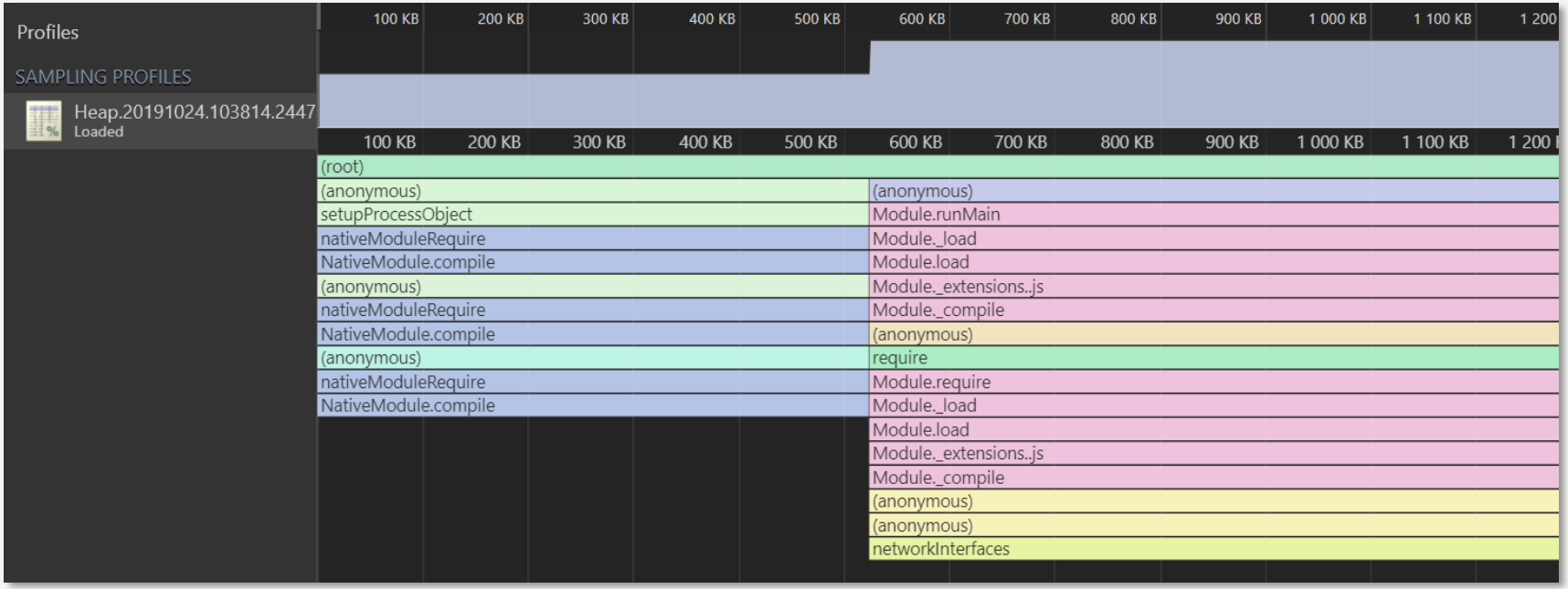
```
const logFile = fs.createWriteStream('/tmp/ssl-keys.log', { flags: 'a' });

server.on('keylog', (line, tlsSocket) => {
  if (tlsSocket.remoteAddress !== '...')
    return; // Only log keys for a particular IP
  logFile.write(line);
});
```

Expanded Diagnostics Reporting

Experimental heap profiling (--heap-prof)

Network interfaces in diagnostic report

Event Loop Delay Monitoring


Profiles
SAMPLING PROFILES
Heap.20191024.103814.2447
Loaded
100 KB
200 KB
300 KB
400 KB
500 KB
600 KB
700 KB
800 KB
900 KB
1 000 KB
1 100 KB
1 200
(root)
(anonymous)
setupProcessObject
nativeModuleRequire
NativeModule.compile
(anonymous)
nativeModuleRequire
NativeModule.compile
(anonymous)
nativeModuleRequire
NativeModule.compile
(anonymous)
Module.runMain
Module._load
Module.load
Module._extensions..js
Module._compile
(anonymous)
require
Module.require
Module._load
Module.load
Module._extensions..js
Module._compile
(anonymous)
(anonymous)
networkInterfaces

Recursive mkdir and rmdir

```
fs.mkdir('/tmp/a/apple', { recursive: true }, (err) => {
  if (err) throw err;
});

fs.rmdir('/tmp/a/apple', { recursive: true }, (err) => {
  if (err) throw err;
});
```

Iterative Directory Listing with fs.Dir

```
async function print(path) {
  const dir = await fs.promises.opendir(path);
  for await (const dirent of dir) {
    console.log(dirent.name);
  }
}
print('./').catch(console.error);
```

V8 7.8

Faster Parsing, Faster Destructuring, WebAssembly Improvements, Intl.NumberFormat improvements, JSON.parse improvements, Promise.allSettled, BigInt improvements, top level await (but only in Modules)

NearForm RESEARCH

QUIC/HTTP3

What’s QUIC?

QUIC is the new UDP-based Transport Protocol for HTTP

Currently under development at the IETF

Currently being implemented for Node.js thanks to NearForm and Protocol Labs

```
const { createSocket } = require('quic');

const sock = createSocket({ port: 8000 });

sock.listen({ key, cert, ca, alpn: 'xyz' });

sock.on('session', (session) => {
  session.on('stream', (stream) => {
    stream.end('Hello World');
  });
});

const req = sock.connect({
  key, cert, ca,
  address: 'localhost',
  port: 1234,
  servername: 'example.com',
});
req.on('secure', () => {
  const stream = req.openStream();
  stream.end('Say Hello');
  stream.on('data', console.log);
});
```

The QUIC API in Node.js is designed to be as familiar as possible while giving full access to every feature of the protocol.

Developers can use HTTP/3, or they can create their own QUIC Applications using familiar primitives.

The goal is to land QUIC/HTTP3 as experimental in Node.js 13.x by the end of 2019.

NearForm
RESEARCH

United States: + 1 916 235 6459
United Kingdom: + 44 773 370 0655
International: + 353 1 514 3545

nearform.com
sales@nearform.com