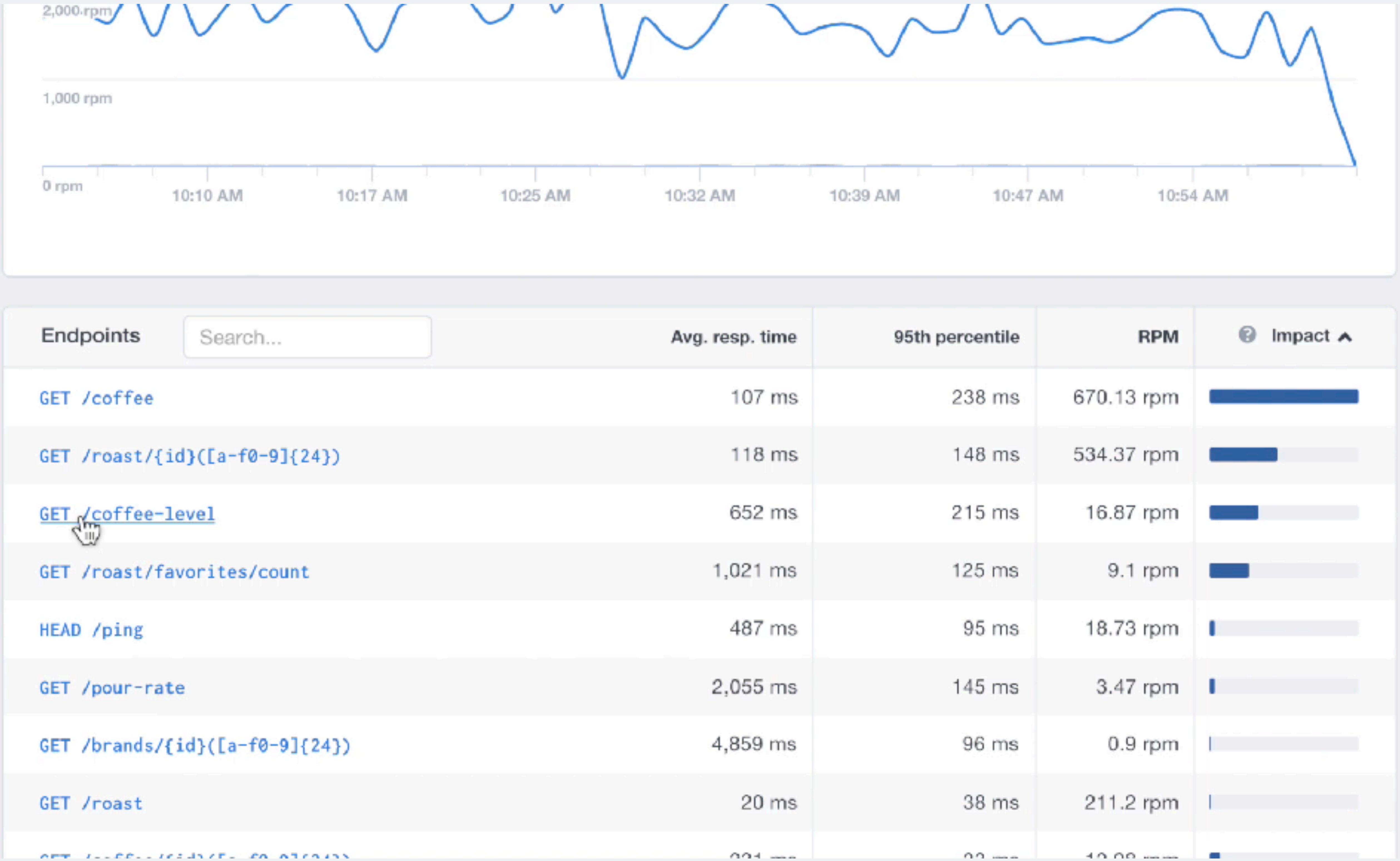
Thomas Watson

@wa7son

github.com/watson

opbeat

@wa7son

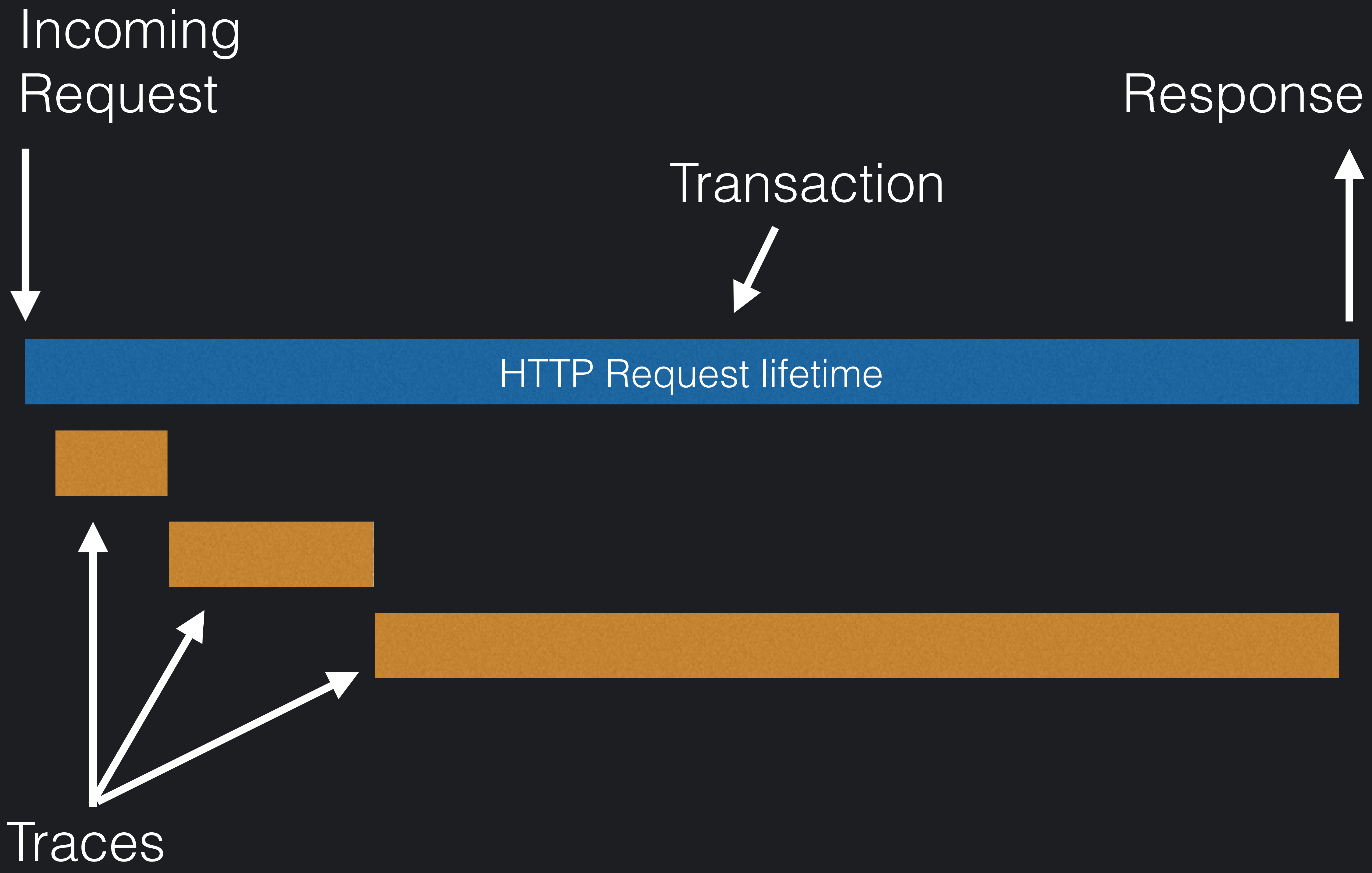


Endpoints	Avg. resp. time	95th percentile	RPM	Impact
GET /coffee	107 ms	238 ms	670.13 rpm	
GET /roast/{id}([a-f0-9]{24})	118 ms	148 ms	534.37 rpm	
GET /coffee-level	652 ms	215 ms	16.87 rpm	
GET /roast/favorites/count	1,021 ms	125 ms	9.1 rpm	
HEAD /ping	487 ms	95 ms	18.73 rpm	
GET /pour-rate	2,055 ms	145 ms	3.47 rpm	
GET /brands/{id}([a-f0-9]{24})	4,859 ms	96 ms	0.9 rpm	
GET /roast	20 ms	38 ms	211.2 rpm	

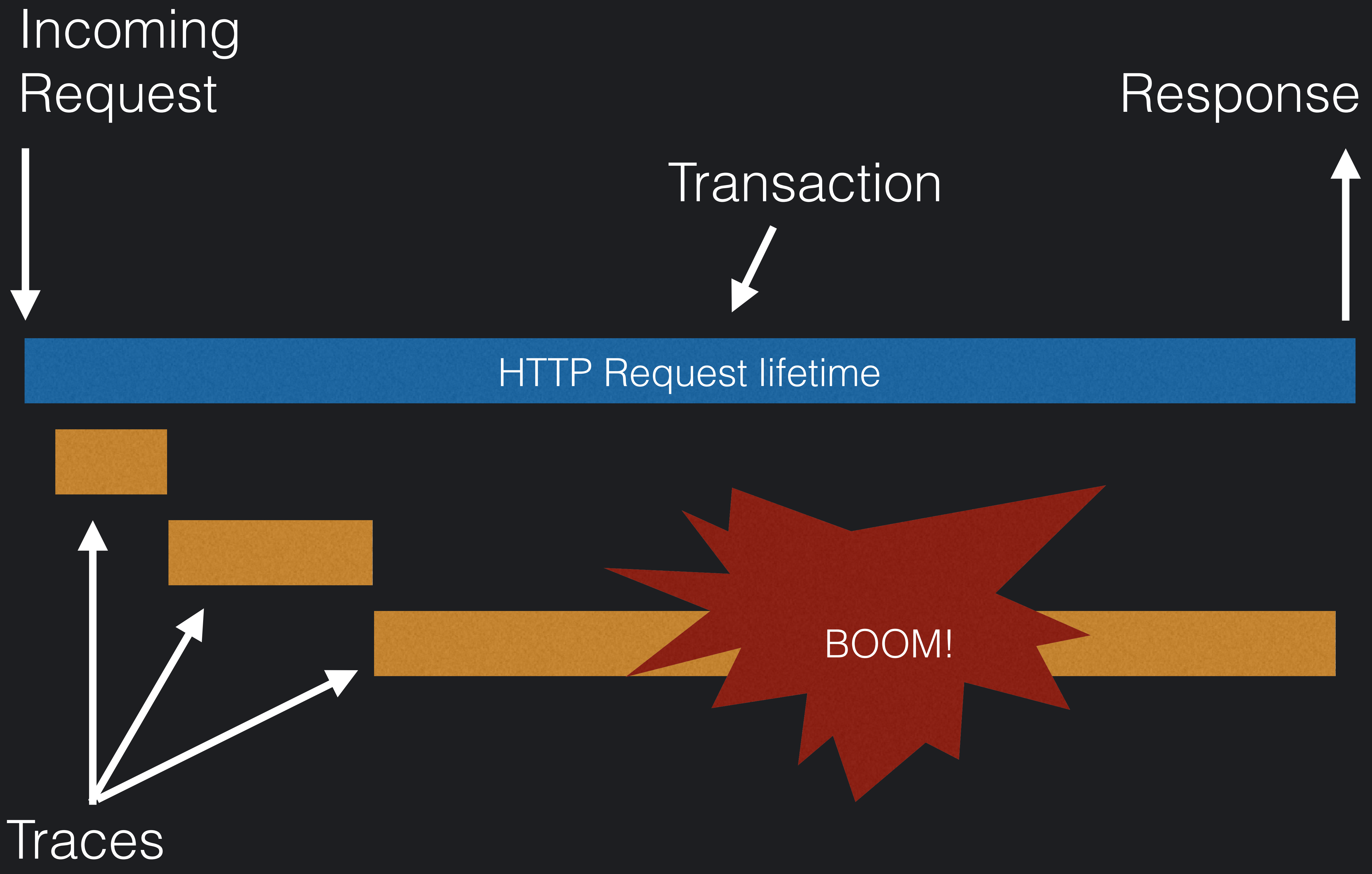
@wa7son

AsyncHooks: A new API coming to Node.js core

@wa7son


Incoming Request
Response
Transaction
HTTP Request lifetime
Traces

@wa7son


Incoming Request
Response
Transaction
HTTP Request lifetime
BOOM!
Traces

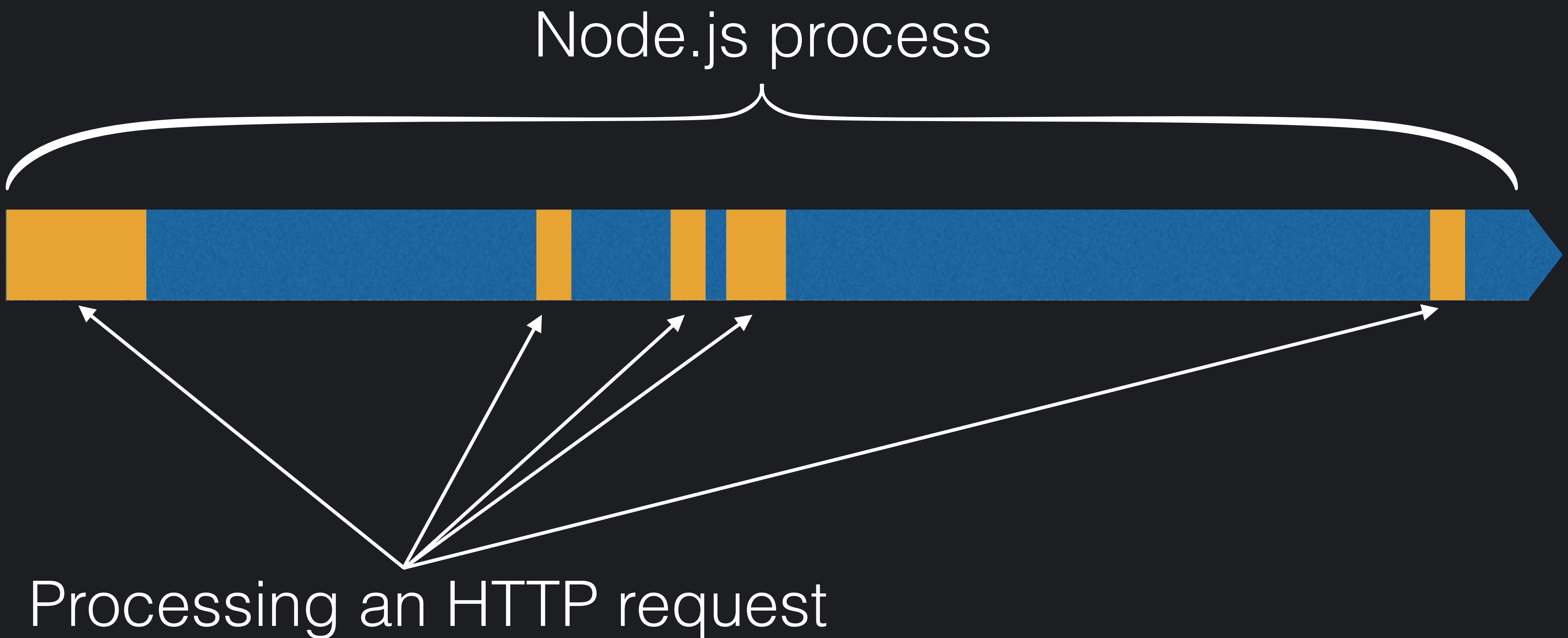
@wa7son

The Event Loop

@wa7son

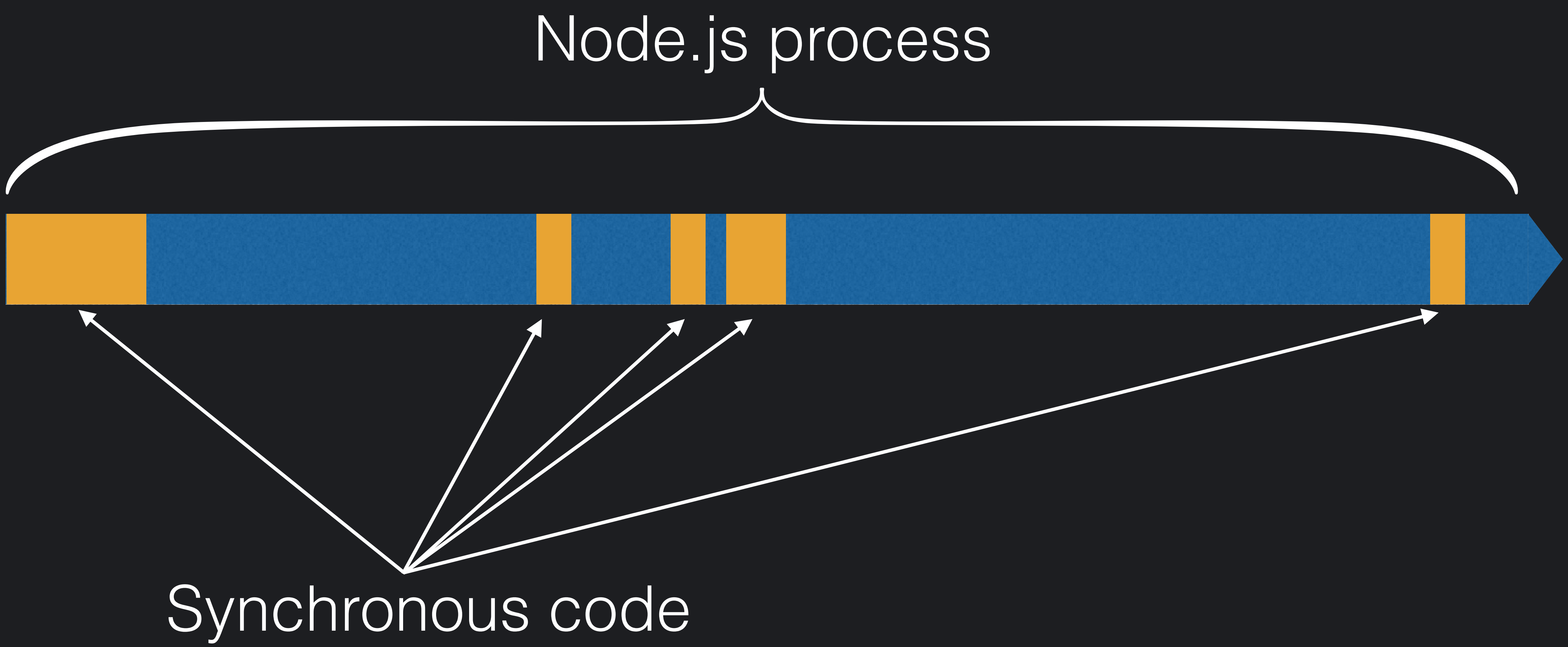
nodejsreactions.tumblr.com

The Event Loop



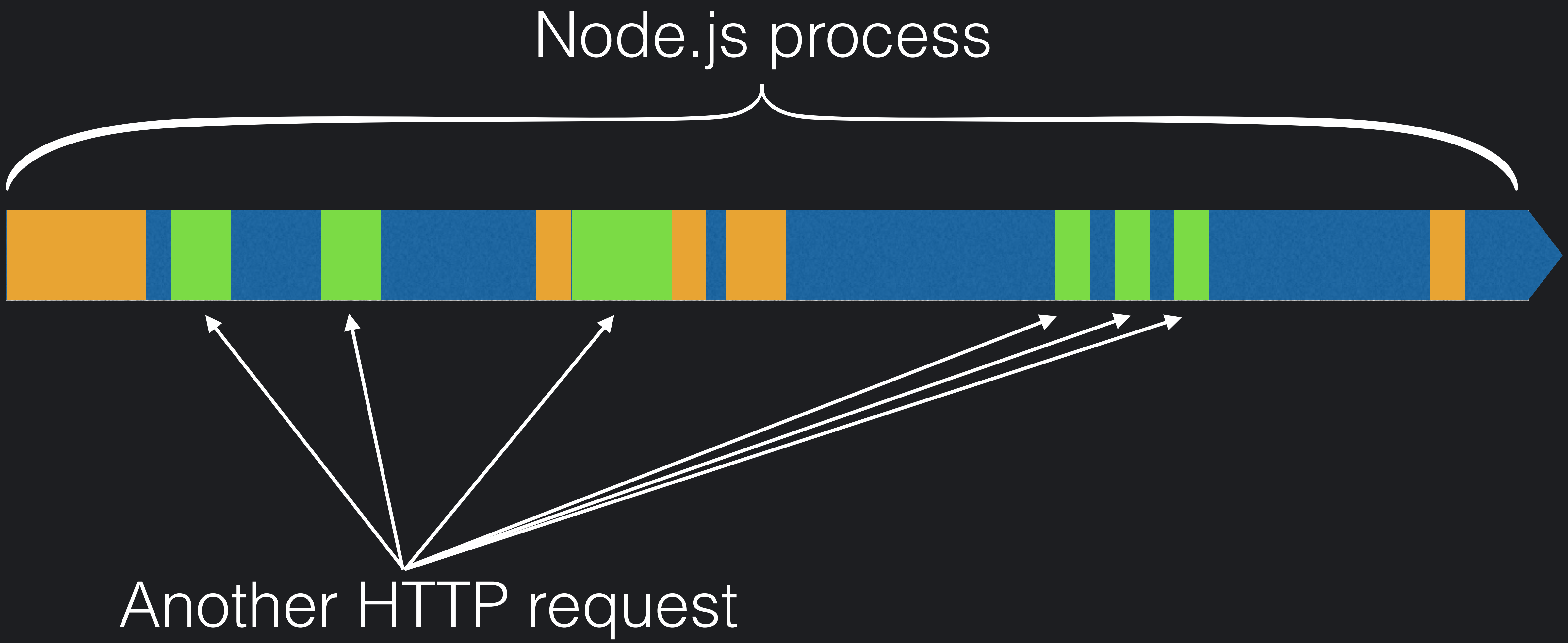
@wa7son

The Event Loop



@wa7son

The Event Loop

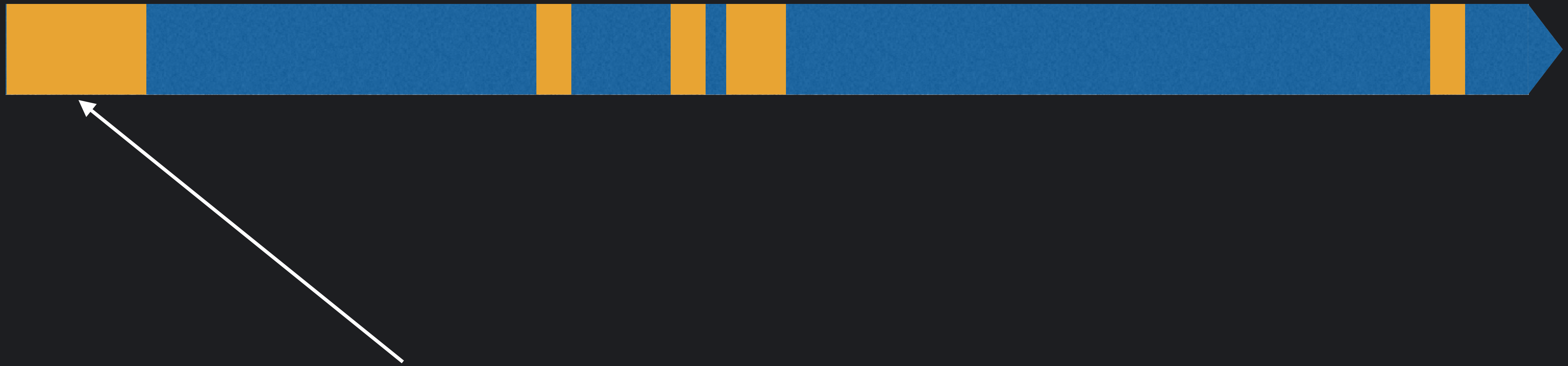


@wa7son

```
http.createServer(function (req, res) {
  global.currentRequest = req
  // ... continue as normal
})
```

@wa7son

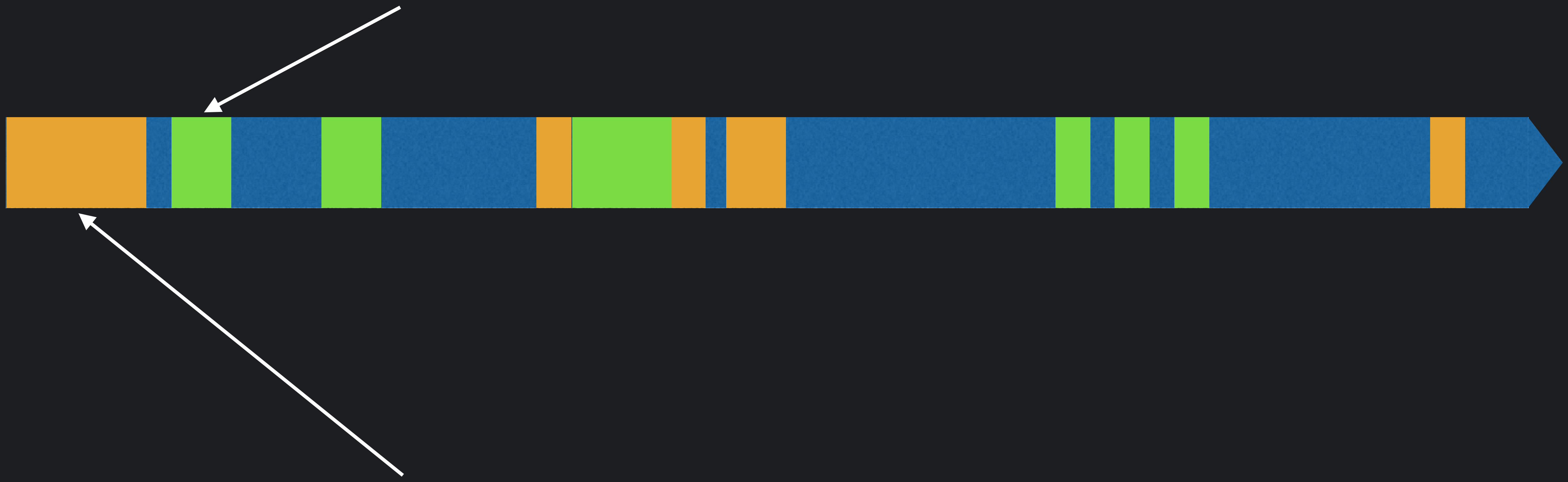
Request Lifecycle



global.currentRequest == Req #1

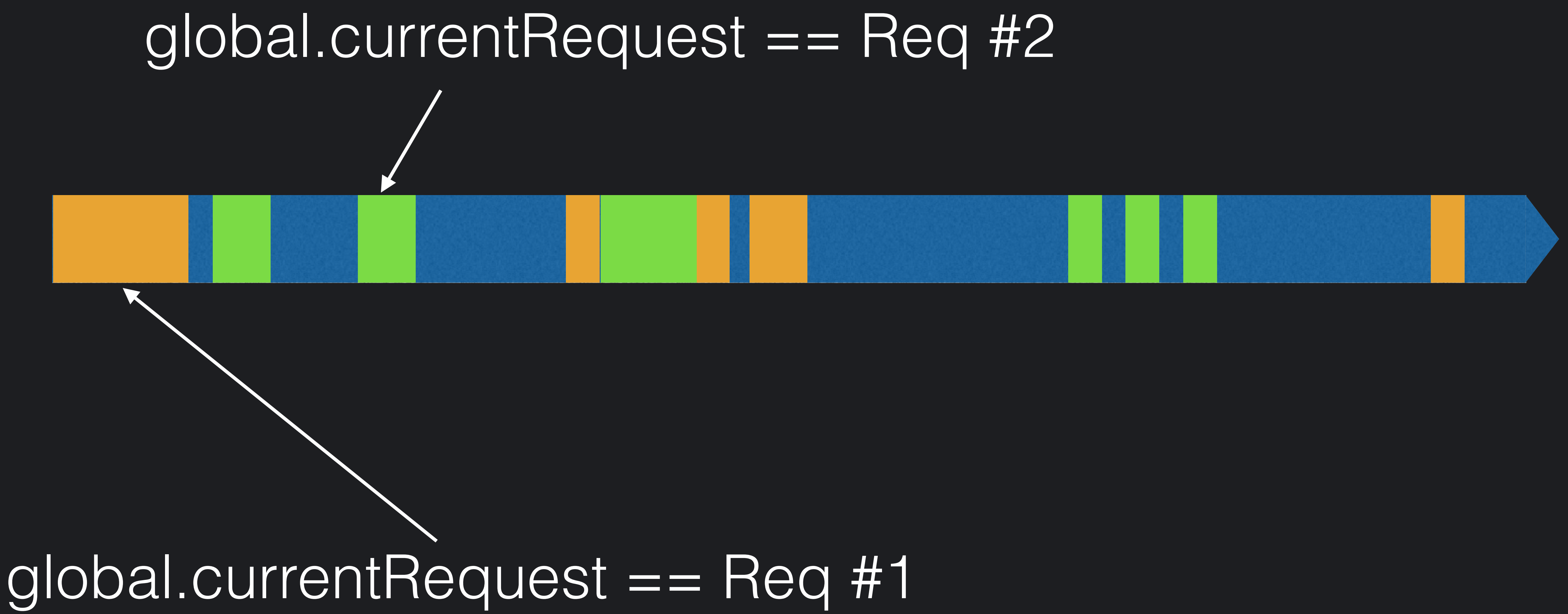
@wa7son

Request Lifecycle


global.currentRequest == Req #2
global.currentRequest == Req #1

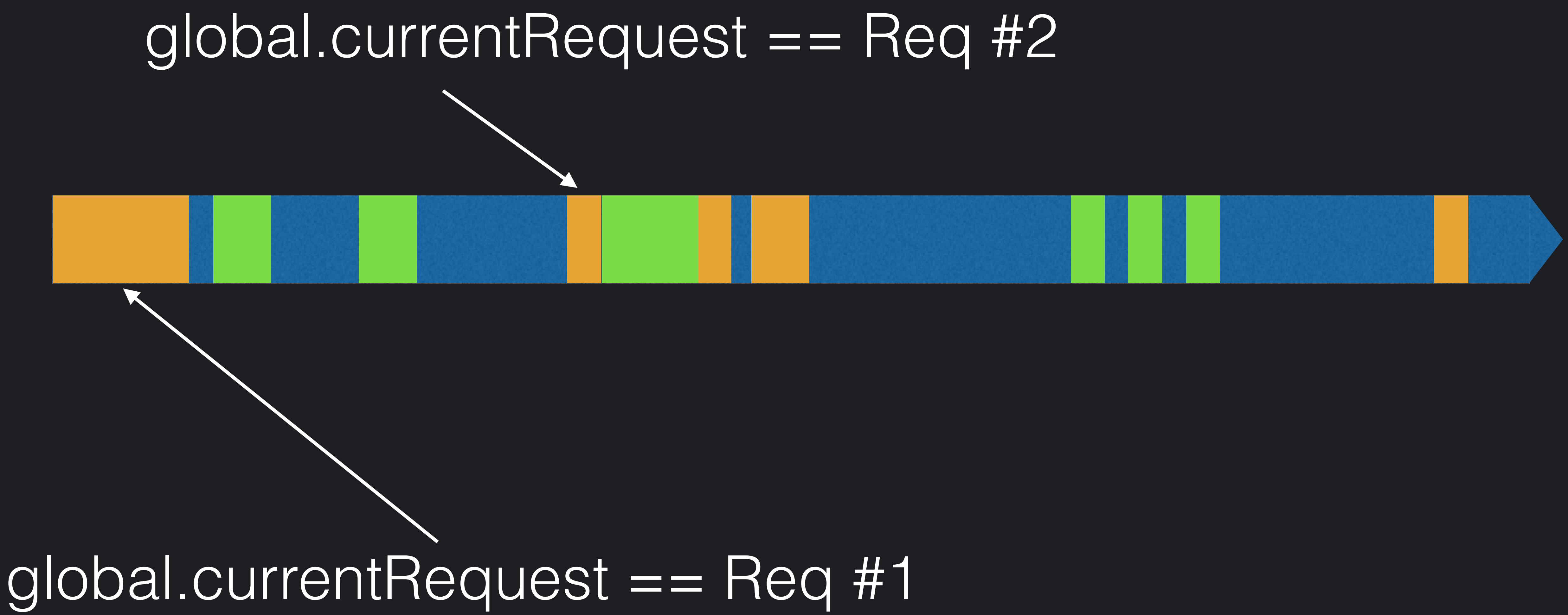
@wa7son

Request Lifecycle


global.currentRequest == Req #2
global.currentRequest == Req #1

@wa7son

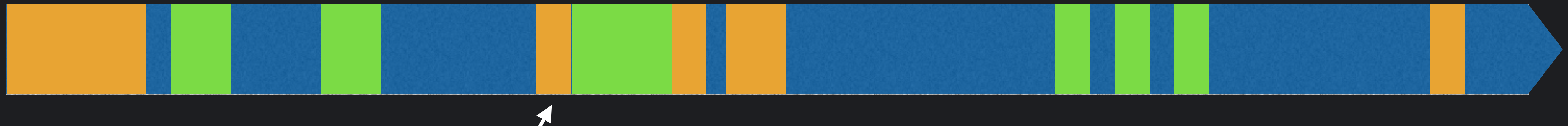
Request Lifecycle


global.currentRequest == Req #2
global.currentRequest == Req #1

@wa7son

Request Lifecycle

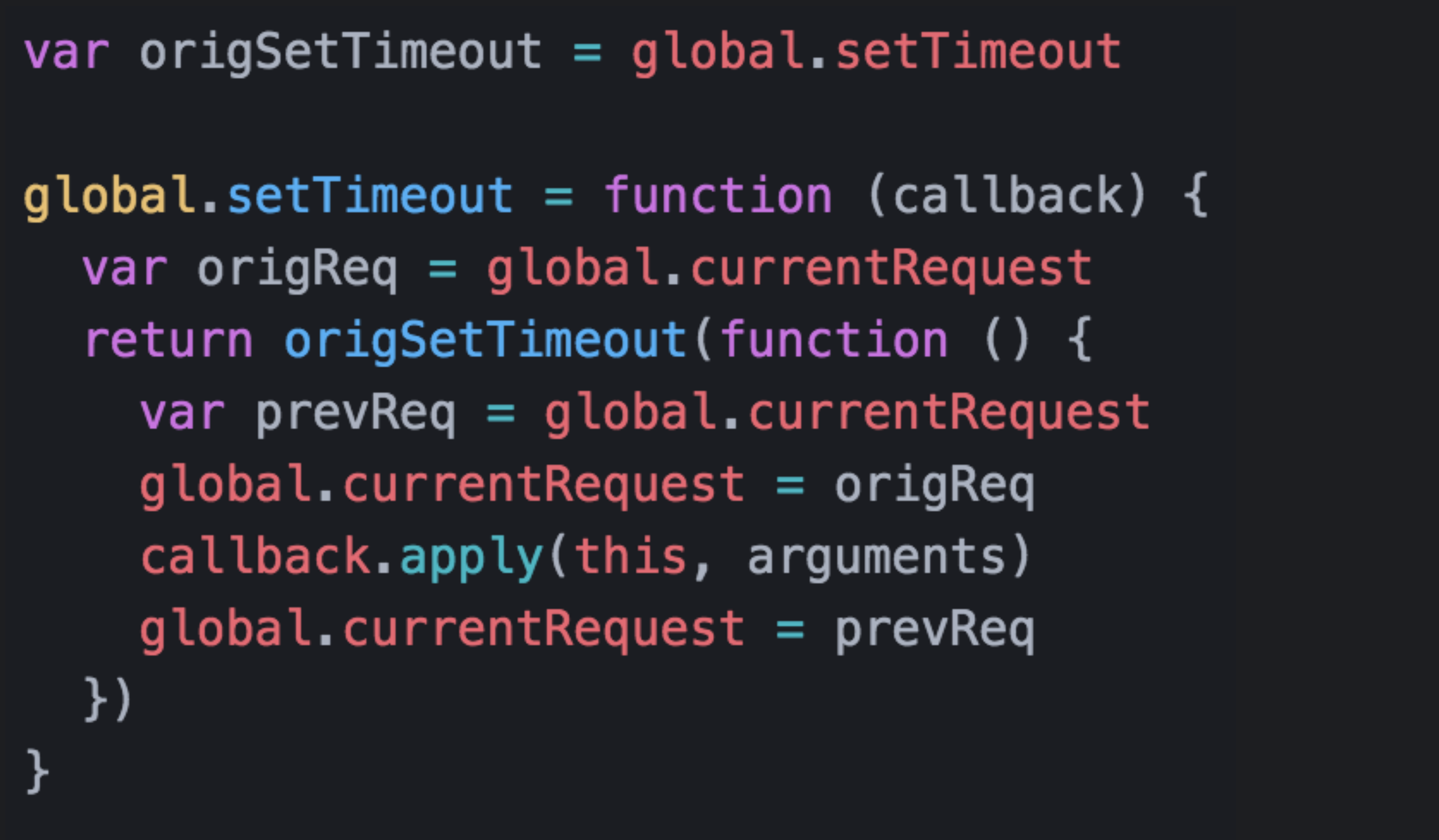
global.currentRequest == Req #2



global.currentRequest == Req #1

@wa7son

@wa7son



```
var origSetTimeout = global.setTimeout

global.setTimeout = function (callback) {
  var origReq = global.currentRequest
  return origSetTimeout(function () {
    var prevReq = global.currentRequest
    global.currentRequest = origReq
    callback.apply(this, arguments)
    global.currentRequest = prevReq
  })
}
```

@wa7son

@wa7son

Patch the world

- Patch **every** async operation in core
  - timers
  - process.nextTick
  - Promise (native)
  - libuv
- Patch certain 3rd party modules

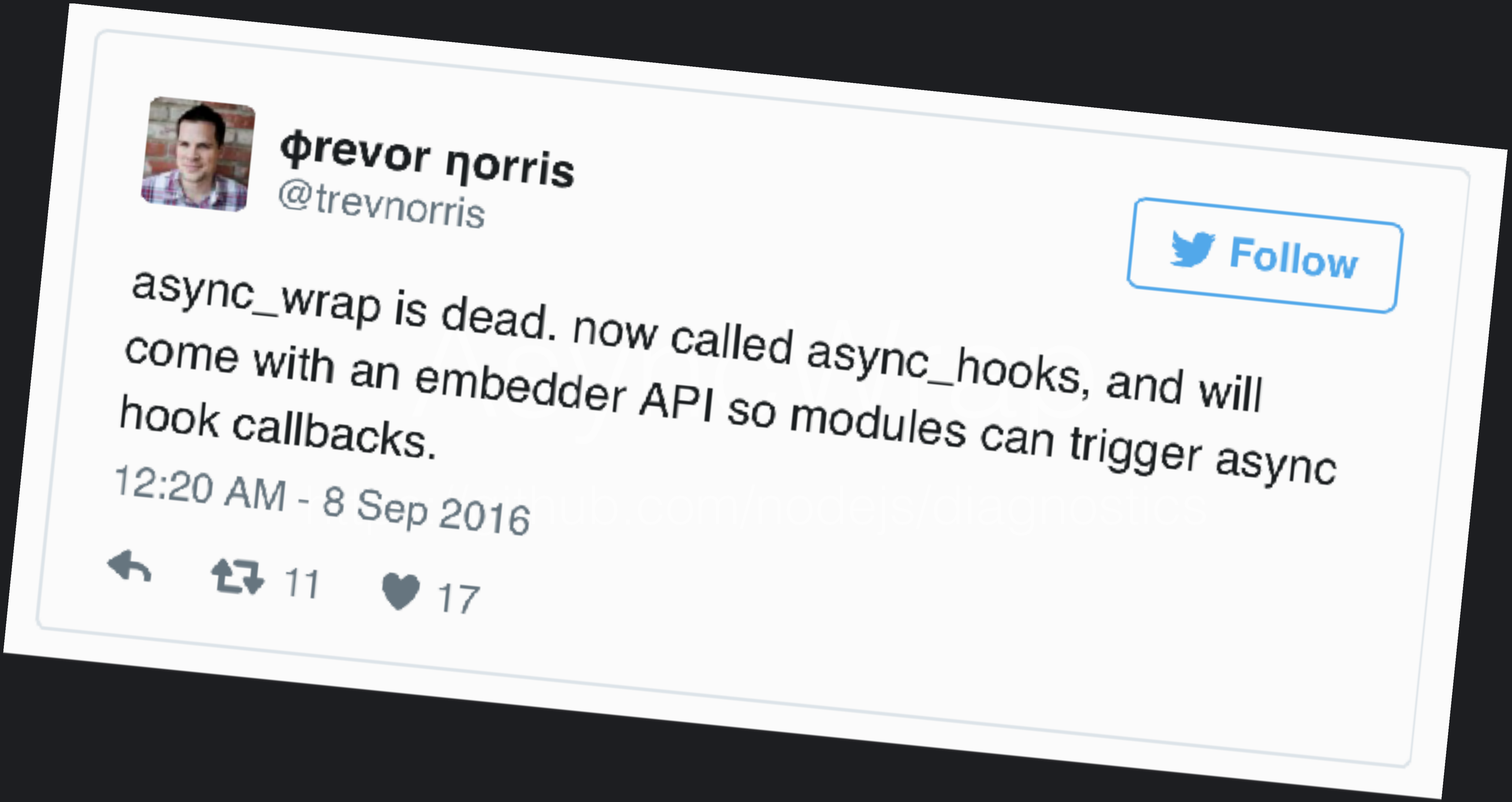
AsyncWrap

https://github.com/nodejs/diagnostics

@wa7son

Async Wrap

https://github.com/nodejs/diagnostics



ɸrevor ɳorris
@trevnorris

async_wrap is dead. now called async_hooks, and will come with an embedder API so modules can trigger async hook callbacks.

12:20 AM - 8 Sep 2016

@wa7son

AsyncWrap

https://github.com/nodejs/diagnostics

@wa7son

Async Hooks

https://github.com/nodejs/diagnostics

@wa7son

```
var asyncWrap = process.binding('async_wrap')
```

@wa7son

```
var asyncWrap = process.binding('async_wrap')

asyncWrap.setupHooks({init, pre, post, destroy})
asyncWrap.enable()
```

@wa7son

```
var asyncWrap = process.binding('async_wrap')

asyncWrap.setupHooks({init, pre, post, destroy})
asyncWrap.enable()

function init (uid, provider, parentUid, parentHandle) {
  log('async_wrap: init')
}
function pre (uid) {
  log('async_wrap: pre')
}
function post (uid) {
  log('async_wrap: post')
}
function destroy (uid) {
  log('async_wrap: destroy')
}
```

@wa7son

```
}
function post (uid) {
  log('async_wrap: post')
}
function destroy (uid) {
  log('async_wrap: destroy')
}

var fs = require('fs')

log('user: before')
fs.open(__filename, 'r', function (err, fd) {
  log('user: done')
})
log('user: after')
```

@wa7son

```
var asyncWrap = process.binding('async_wrap')

asyncWrap.setupHooks({init, pre, post, destroy})
asyncWrap.enable()

function init (uid, provider, parentUid, parentHandle) {
  log('async_wrap: init')
}
function pre (uid) {
  log('async_wrap: pre')
}
function post (uid) {
  log('async_wrap: post')
}
function destroy (uid) {
  log('async_wrap: destroy')
}

var fs = require('fs')

log('user: before')
fs.open(__filename, 'r', function (err, fd) {
  log('user: done')
})
log('user: after')
```

@wa7son

```
var asyncWrap = process.binding('async_wrap')

asyncWrap.setupHooks({init, pre, post, destroy})
asyncWrap.enable()

function init (uid, provider, parentUid, parentHandle) {
  log('async_wrap: init')
}
function pre (uid) {
  log('async_wrap: pre')
}
function post (uid) {
  log('async_wrap: post')
}
function destroy (uid) {
  log('async_wrap: destroy')
}

var fs = require('fs')

log('user: before')
fs.open(__filename, 'r', function (err, fd) {
  log('user: done')
})
log('user: after')
```

```
user: before
async_hooks: init
user: after
async_hooks: pre
user: done
async_hooks: post
async_hooks: destroy
```

@wa7son

```javascript
var asyncWrap = process.binding('async_wrap')

asyncWrap.setupHooks({init, pre, post, destroy})
asyncWrap.enable()

function init (uid, provider, parentUid, parentHandle) {
  console.log('async_wrap: init')
}
function pre (uid) {
  console.log('async_wrap: pre')
}
function post (uid) {
  console.log('async_wrap: post')
}
function destroy (uid) {
  console.log('async_wrap: destroy')
}

var fs = require('fs')

console.log('user: before')
fs.open(__filename, 'r', function (err, fd) {
  console.log('user: done')
})
console.log('user: after')
```

@wa7son

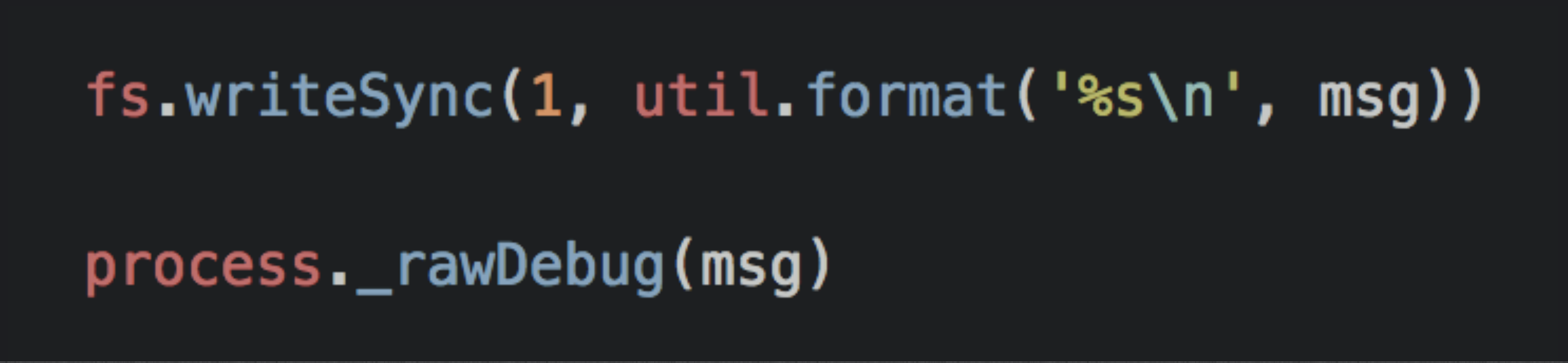
```
var asyncWrap = process.binding('async_wrap')

asyncWrap.setupHooks({init, pre, post, destroy})
asyncWrap.enable()

function init (uid, provider, parentUid, parentHandle) {
  console.log('async_wrap: init')
}
function pre (uid) {
  console.log('async_wrap: pre')
}
fun
  c
}
function destroy (uid) {
  console.log('async_wrap: destroy')
}

var fs = require('fs')

console.log('user: before')
fs.open(__filename, 'r', function (err, fd) {
  console.log('user: done')
})
console.log('user: after')
```

```
FATAL ERROR: node::AsyncWrap::AsyncWrap init hook threw
```

@wa7son

```
fs.writeSync(1, util.format('%s\n', msg))
```

@wa7son



```
fs.writeSync(1, util.format('%s\n', msg))

process._rawDebug(msg)
```

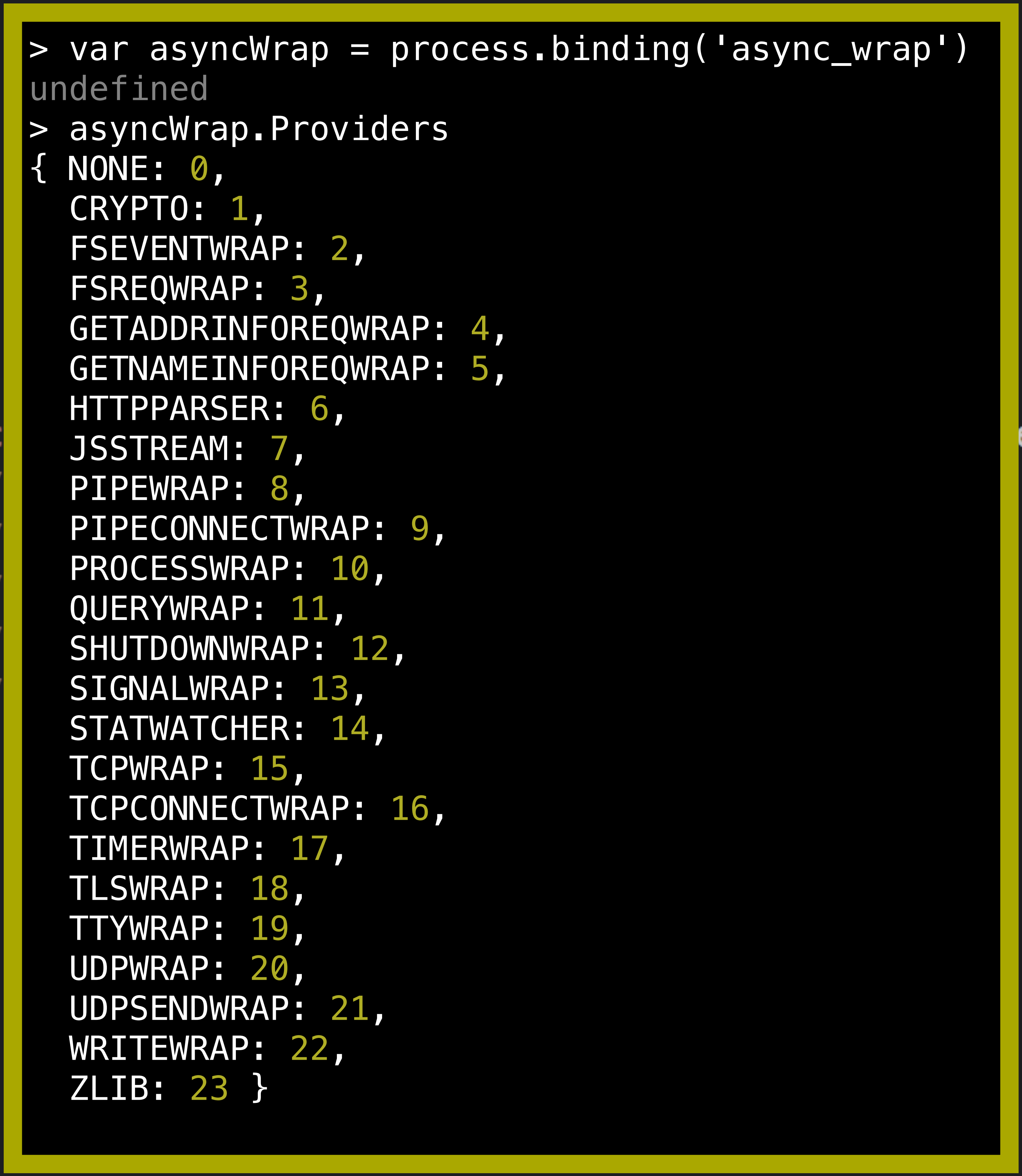
@wa7son

```
var asyncWrap = process.binding('async_wrap')

asyncWrap.setupHooks({init, pre, post, destroy})
asyncWrap.enable()

function init (uid, provider, parentUid, parentHandle) {
  process._rawDebug('async_wrap: init')
}
function pre (uid) {
  process._rawDebug('async_wrap: pre')
}
function post (uid) {
  process._rawDebug('async_wrap: post')
}
function destroy (uid) {
  process._rawDebug('async_wrap: destroy')
}

var fs = require('fs')

process._rawDebug('user: before')
fs.open(__filename, 'r', function (err, fd) {
  process._rawDebug('user: done')
})
process._rawDebug('user: after')
```

```
user: before
async_hooks: init
user: after
async_hooks: pre
user: done
async_hooks: post
async_hooks: destroy
```

@wa7son

```
function init (uid, provider, parentUid, parentHandle) {
  // this         => current handle
  // uid          => 1, 2, 3...
  // provider     => 0 - 23 (asyncWrap.Providers)
  // parentUid    => 1, 2, 3...
  // parentHandle => parent `this`
}
```



```
> var asyncWrap = process.binding('async_wrap')
undefined
> asyncWrap.Providers
{ NONE: 0,
  CRYPTO: 1,
  FSEVENTWRAP: 2,
  FSREQWRAP: 3,
  GETADDRINFOREQWRAP: 4,
  GETNAMEINFOREQWRAP: 5,
  HTTPPARSER: 6,
  JSSTREAM: 7,
  PIPEWRAP: 8,
  PIPECONNECTWRAP: 9,
  PROCESSWRAP: 10,
  QUERYWRAP: 11,
  SHUTDOWNWRAP: 12,
  SIGNALWRAP: 13,
  STATWATCHER: 14,
  TCPWRAP: 15,
  TCPCONNECTWRAP: 16,
  TIMERWRAP: 17,
  TLSWRAP: 18,
  TTYWRAP: 19,
  UDPWRAP: 20,
  UDPSENDWRAP: 21,
  WRITEWRAP: 22,
  ZLIB: 23 }
```

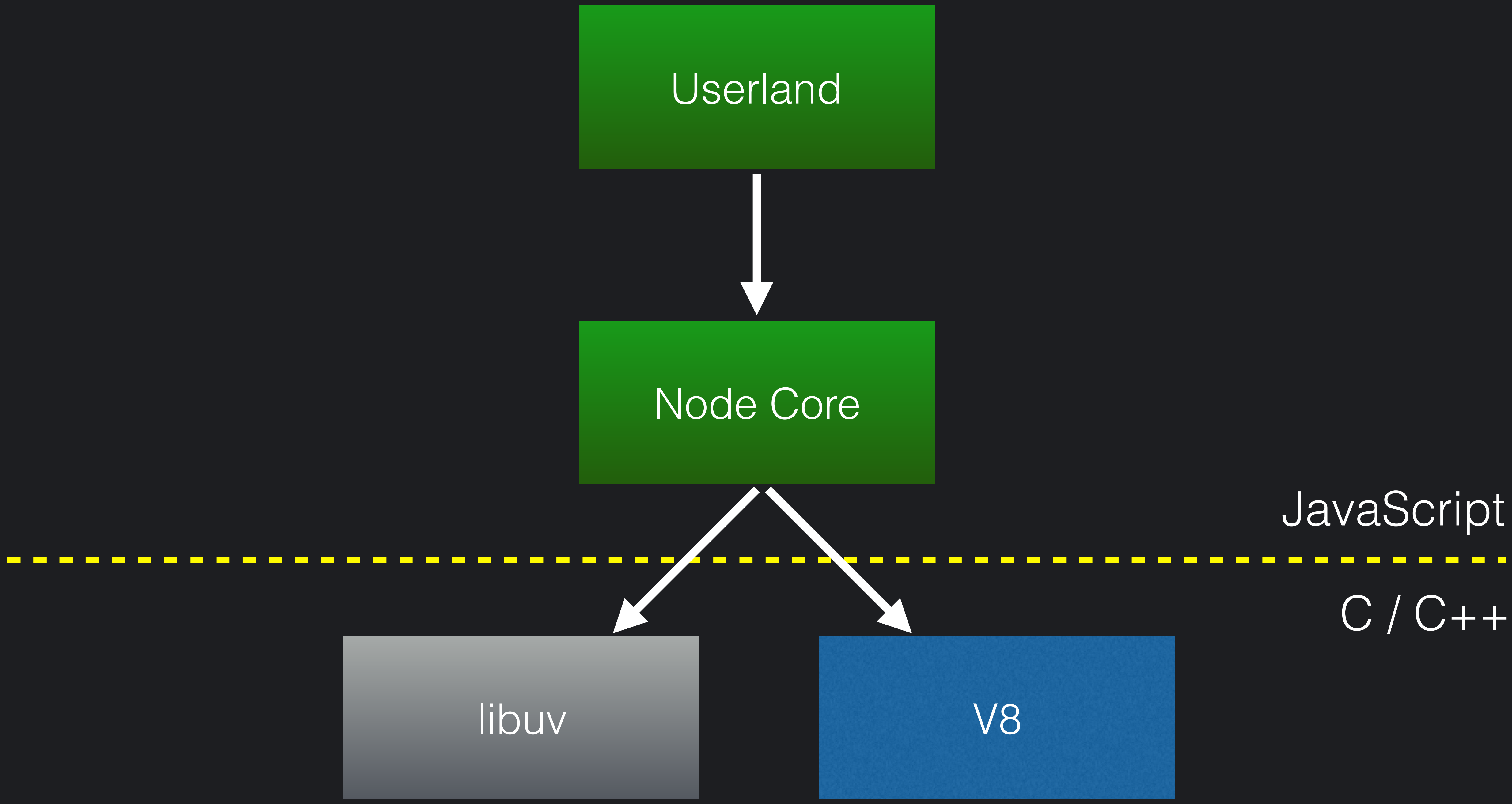
```
func                                            e) {
  //
  //
  //
  //
  //
}
```

@wa7son

```
function init (uid, provider, parentUid, parentHandle) {
  // this         => current handle
  // uid          => 1, 2, 3...
  // provider     => 0 - 23 (asyncWrap.Providers)
  // parentUid    => 1, 2, 3...
  // parentHandle => parent `this`
}
```

@wa7son

Handle Objects



@wa7son

Handle Objects

```
const TCPConnectWrap = process.binding('tcp_wrap').TCPConnectWrap;
const TCP = process.binding('tcp_wrap').TCP;

const req = new TCPConnectWrap();
req.oncomplete = oncomplete;
req.address = address;
req.port = port;

const socket = new TCP();
socket.onread = onread;
socket.connect(req, address, port);

// later
socket.destroy();
```

Handle Objects

```
const TCPConnectWrap = process.binding('tcp_wrap').TCPConnectWrap;
const TCP = process.binding('tcp_wrap').TCP;

const req = new TCPConnectWrap();
req.oncomplete = oncomplete;
req.address = address;
req.port = port;

const socket = new TCP();
socket.onread = onread;
socket.connect(req, address, port);

// later
socket.destroy();
```

Handle objects

@wa7son

Handle Objects

```
const TCPConnectWrap = process.binding('tcp_wrap').TCPConnectWrap;
const TCP = process.binding('tcp_wrap').TCP;

const req = new TCPConnectWrap();
req.oncomplete = oncomplete;
req.address = address;
req.port = port;

const socket = new TCP();
socket.onread = onread;
socket.connect(req, address, port);

// later
socket.destroy();
```

req === this

socket === this

Timers

```
log('user: before #1')
setTimeout(function () {
  log('user: done #1')
}, 2000)
log('user: after #1')

log('user: before #2')
setTimeout(function () {
  log('user: done #2')
}, 2000)
log('user: after #2')
```

Timers

```
log('user: before #1')
setTimeout(function () {
  log('user: done #1')
}, 2000)
log('user: after #1')

log('user: before #2')
setTimeout(function () {
  log('user: done #2')
}, 2000)
log('user: after #2')
```

```
user: before #1
async_hooks: init
user: after #1
user: before #2
user: after #2
```

Timers

```
log('user: before #1')
setTimeout(function () {
  log('user: done #1')
}, 2000)
log('user: after #1')

log('user: before #2')
setTimeout(function () {
  log('user: done #2')
}, 2000)
log('user: after #2')
```

```
user: before #1
async_hooks: init
user: after #1
user: before #2
user: after #2

async_hooks: pre
user: done #1
user: done #2
async_hooks: post
async_hooks: destroy
```

@wa7son

Real World Example

@wa7son

```
const asyncWrap = process.binding('async_wrap')
```

@wa7son

```
const asyncWrap = process.binding('async_wrap')
const TIMER = asyncWrap.Providers.TIMERWRAP
const asyncWrap = process.binding('async_wrap')
const TIMER = asyncWrap.Providers.TIMERWRAP

asyncWrap.setupWrap({init, before, after, destroy})
```

```
const asyncWrap = process.binding('async_wrap')
const TIMER = asyncWrap.Providers.TIMERWRAP

asyncWrap.setupWrap({init, before, after, destroy})
asyncWrap.enable()
```

@wa7son

```
const asyncWrap = process.binding('async_wrap')
const TIMER = asyncWrap.Providers.TIMERWRAP

asyncWrap.setupWrap({init, before, after, destroy})
asyncWrap.enable()

const initState = new Map()
const prevState = new Map()
```

@wa7son

```javascript
const asyncWrap = process.binding('async_wrap')
const TIMER = asyncWrap.Providers.TIMERWRAP

asyncWrap.setupWrap({init, before, after, destroy})
asyncWrap.enable()

const initState = new Map()
const prevState = new Map()

function init (uid, provider, parentUid, parentHandle) {
  if (provider === TIMER) return // timers share handles, manage manually
  initState.set(uid, global.currentRequest)
}
```

@wa7son

```javascript
const asyncWrap = process.binding('async_wrap')
const TIMER = asyncWrap.Providers.TIMERWRAP

asyncWrap.setupWrap({init, before, after, destroy})
asyncWrap.enable()

const initState = new Map()
const prevState = new Map()

function init (uid, provider, parentUid, parentHandle) {
  if (provider === TIMER) return // timers share handles, manage manually
  initState.set(uid, global.currentRequest)
}

function before (uid) {
  if (!initState.has(uid)) return // in case provider === TIMER
  prevState.set(uid, global.currentRequest)
  global.currentRequest = initState.get(uid)
}
```

@wa7son

```
asyncWrap.setupWrap({init, before, after, destroy})
asyncWrap.enable()

const initState = new Map()
const prevState = new Map()

function init (uid, provider, parentUid, parentHandle) {
  if (provider === TIMER) return // timers share handles, manage manually
  initState.set(uid, global.currentRequest)
}

function before (uid) {
  if (!initState.has(uid)) return // in case provider === TIMER
  prevState.set(uid, global.currentRequest)
  global.currentRequest = initState.get(uid)
}

function after (uid) {
  if (!initState.has(uid)) return // in case provider === TIMER
  global.currentRequest = prevState.get(uid)
}
```

@wa7son

```
function init (uid, provider, parentUid, parentHandle) {
  if (provider === TIMER) return // timers share handles, manage manually
  initState.set(uid, global.currentRequest)
}

function before (uid) {
  if (!initState.has(uid)) return // in case provider === TIMER
  prevState.set(uid, global.currentRequest)
  global.currentRequest = initState.get(uid)
}

function after (uid) {
  if (!initState.has(uid)) return // in case provider === TIMER
  global.currentRequest = prevState.get(uid)
}

function destroy (uid) {
  if (!initState.has(uid)) return // in case provider === TIMER
  initState.delete(uid)
  prevState.delete(uid)
}
```

@wa7son

AsyncHooks Gotchas

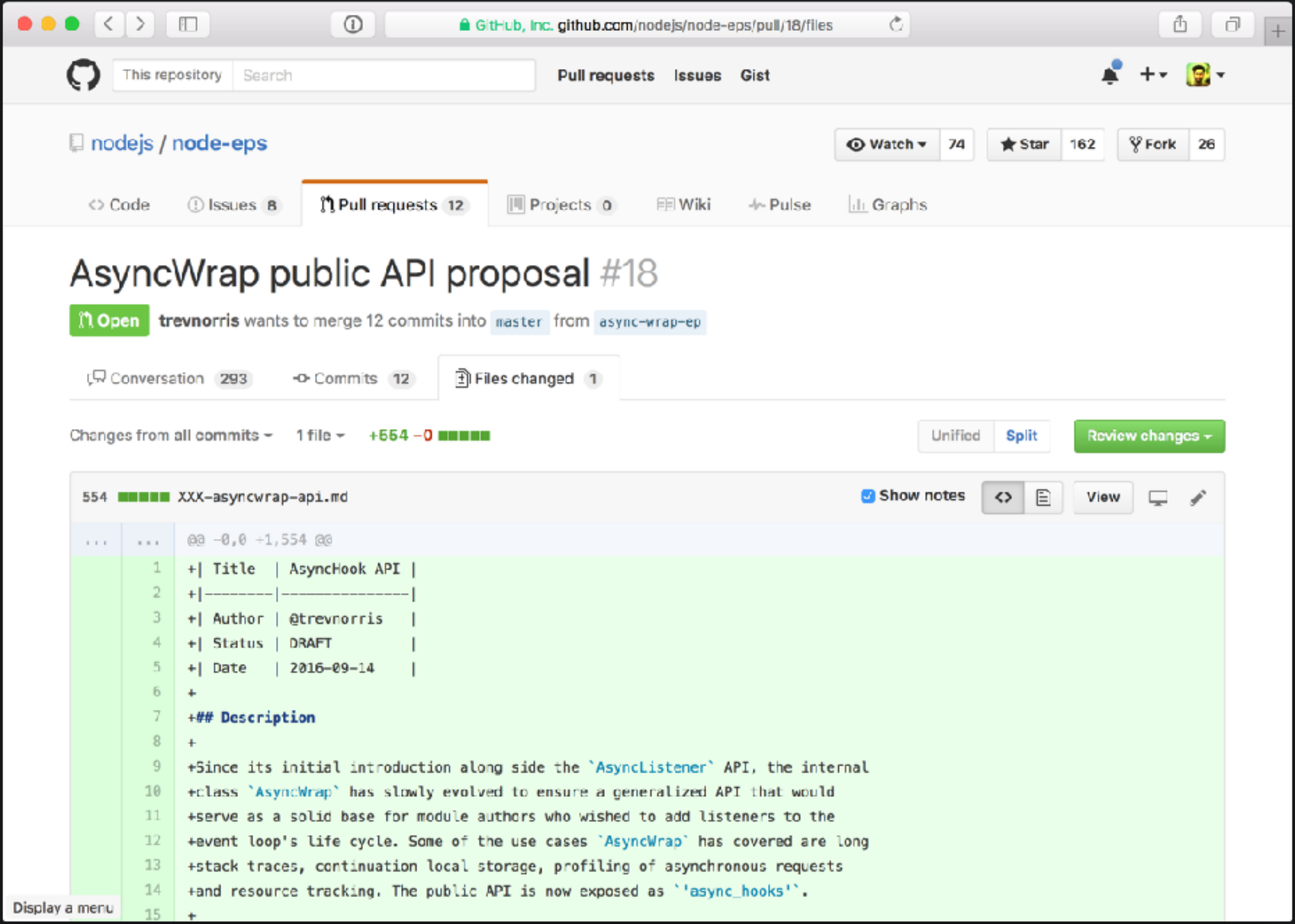
- Handle creation time
- console.log
- (process.nextTick)
- Timers
- Promises
- Multiple AsyncHooks

Callback queues in user-land

@wa7son

ES Modules

@wa7son



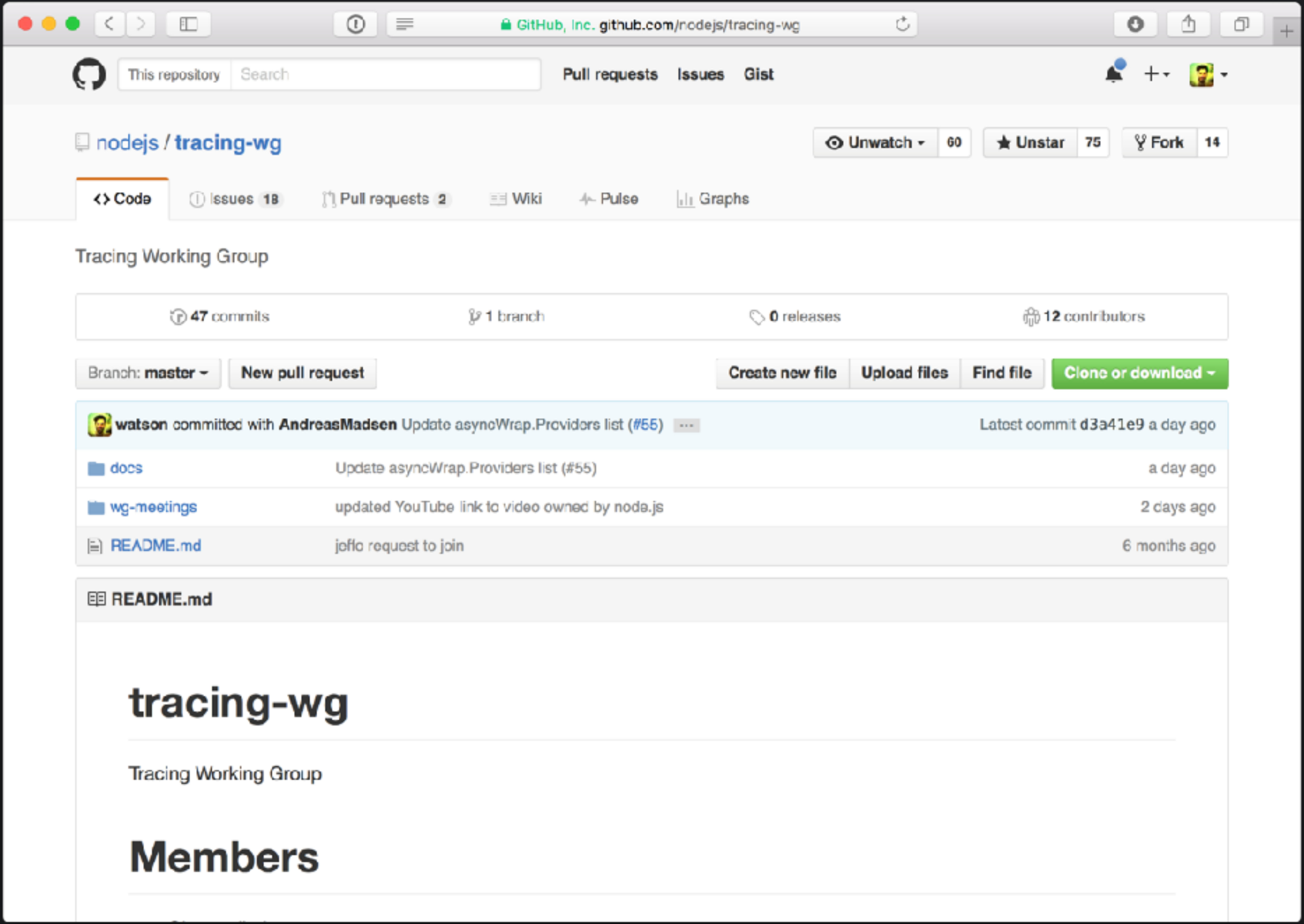
AsyncWrap public API proposal #18

Open trevnorris wants to merge 12 commits into `master` from `async-wrap-ep`

```
@@ -0,0 +1,554 @@
+| Title  | AsyncHook API |
+|--------|---------------|
+| Author | @trevnorris   |
+| Status | DRAFT         |
+| Date   | 2016-09-14    |
+
+## Description
+
+Since its initial introduction along side the `AsyncListener` API, the internal
+class `AsyncWrap` has slowly evolved to ensure a generalized API that would
+serve as a solid base for module authors who wished to add listeners to the
+event loop's life cycle. Some of the use cases `AsyncWrap` has covered are long
+stack traces, continuation local storage, profiling of asynchronous requests
+and resource tracking. The public API is now exposed as `'async_hooks'`.
+
```

@wa7son

https://github.com/nodejs/node-eps/pull/18



Tracing Working Group

47 commits · 1 branch · 0 releases · 12 contributors

watson committed with AndreasMadsen Update asyncWrap.Providers list (#55) … Latest commit d3a41e9 a day ago

docs	Update asyncWrap.Providers list (#55)	a day ago
wg-meetings	updated YouTube link to video owned by node.js	2 days ago
README.md	jofio request to join	6 months ago

README.md

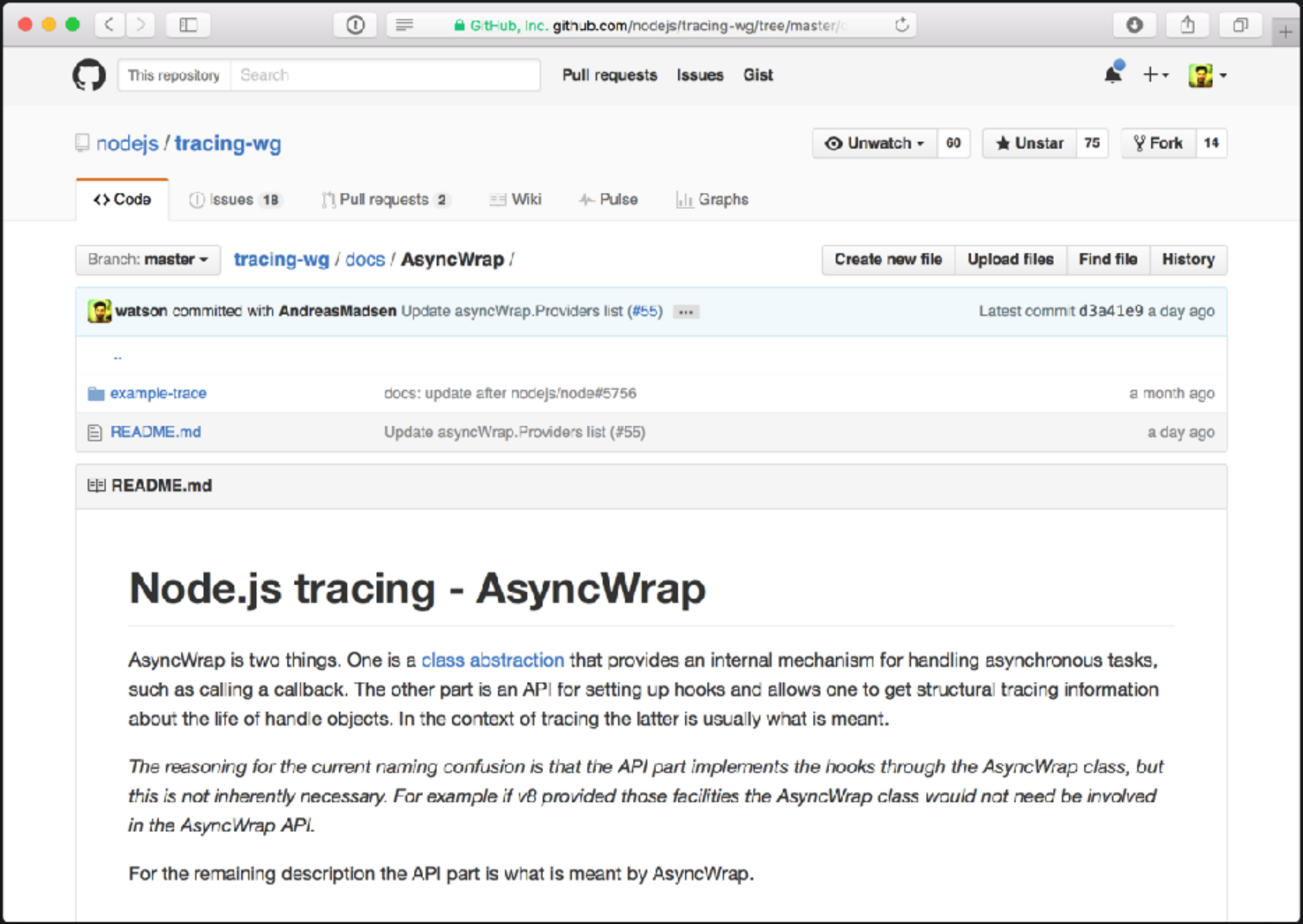
tracing-wg

Tracing Working Group

Members

@wa7son

https://github.com/nodejs/diagnostics



tracing-wg / docs / AsyncWrap /

watson committed with AndreasMadsen Update asyncWrap.Providers list (#55) … Latest commit d3a41e9 a day ago

..

example-trace	docs: update after nodejs/node#5756	a month ago
README.md	Update asyncWrap.Providers list (#55)	a day ago

README.md

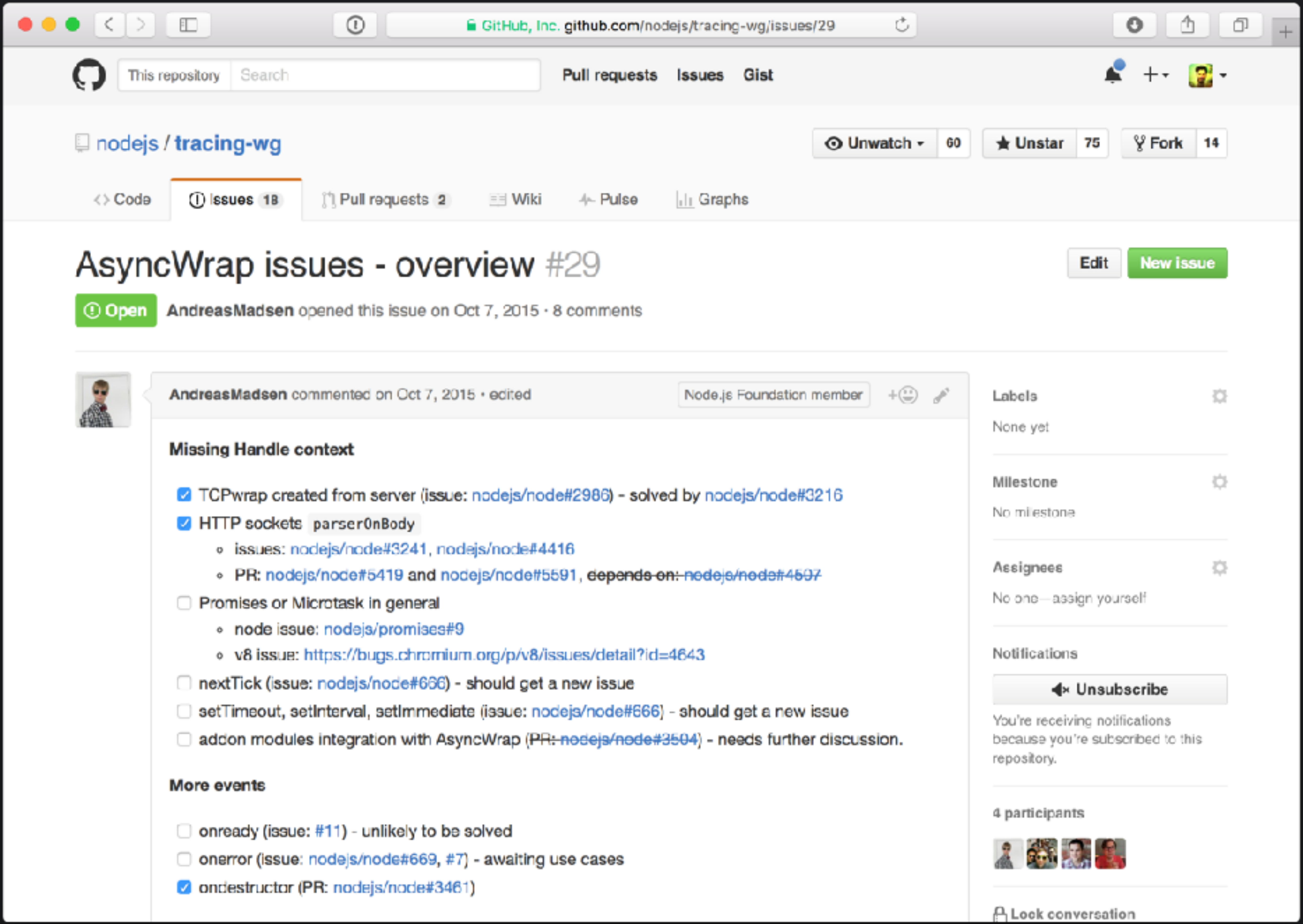
Node.js tracing - AsyncWrap

AsyncWrap is two things. One is a class abstraction that provides an internal mechanism for handling asynchronous tasks, such as calling a callback. The other part is an API for setting up hooks and allows one to get structural tracing information about the life of handle objects. In the context of tracing the latter is usually what is meant.

The reasoning for the current naming confusion is that the API part implements the hooks through the AsyncWrap class, but this is not inherently necessary. For example if v8 provided those facilities the AsyncWrap class would not need be involved in the AsyncWrap API.

For the remaining description the API part is what is meant by AsyncWrap.

https://github.com/nodejs/diagnostics/tree/master/tracing/AsyncWrap



GitHub, Inc. github.com/nodejs/tracing-wg/issues/29

This repository Search Pull requests Issues Gist

nodejs / tracing-wg

Unwatch 60 Unstar 75 Fork 14

Code Issues 18 Pull requests 2 Wiki Pulse Graphs

AsyncWrap issues - overview #29

Edit New issue

Open AndreasMadsen opened this issue on Oct 7, 2015 · 8 comments

AndreasMadsen commented on Oct 7, 2015 • edited Node.js Foundation member

Missing Handle context

- [x] TCPwrap created from server (issue: nodejs/node#2986) - solved by nodejs/node#3216
- [x] HTTP sockets `parserOnBody`
  - issues: nodejs/node#3241, nodejs/node#4416
  - PR: nodejs/node#5419 and nodejs/node#5591, ~~depends on: nodejs/node#4507~~
- [ ] Promises or Microtask in general
  - node issue: nodejs/promises#9
  - v8 issue: https://bugs.chromium.org/p/v8/issues/detail?id=4643
- [ ] nextTick (issue: nodejs/node#666) - should get a new issue
- [ ] setTimeout, setInterval, setImmediate (issue: nodejs/node#666) - should get a new issue
- [ ] addon modules integration with AsyncWrap (~~PR: nodejs/node#3504~~) - needs further discussion.

More events

- [ ] onready (issue: #11) - unlikely to be solved
- [ ] onerror (issue: nodejs/node#669, #7) - awaiting use cases
- [x] ondestructor (PR: nodejs/node#3461)

Labels
None yet

Milestone
No milestone

Assignees
No one—assign yourself

Notifications
Unsubscribe
You're receiving notifications because you're subscribed to this repository.

4 participants

Lock conversation

@wa7son https://github.com/nodejs/diagnostics/issues/29

Grazie

@wa7son

github.com/watson

opbeat