DEVIEW
2019

console.warn('좀 불안하지?')

// node.js 모니터링을 위한 APM 개발기

배근배
SHOPPING DISPLAY

NAVER

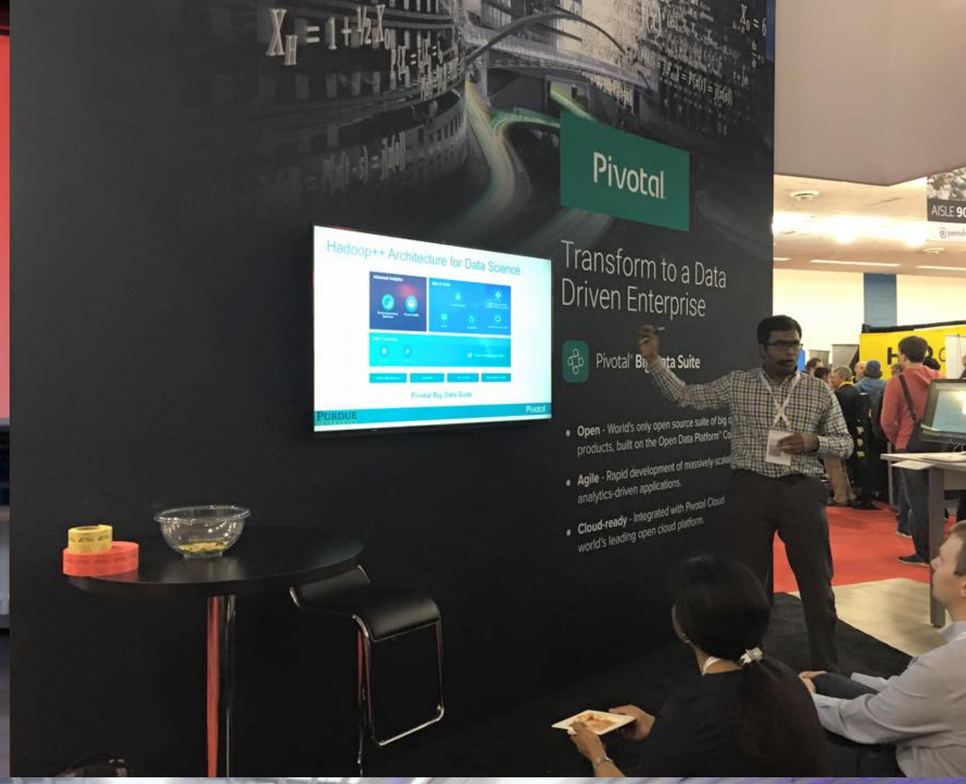
CONTENTS

DEVIEW
2019

1. From Java to JS
2. 쇼핑개발자가 A.P.M Library 를 만든 이유
3. 모니터링 맛집, 3가지 비밀 레시피
4. Sharing

DEVIEW
2019

Introduction

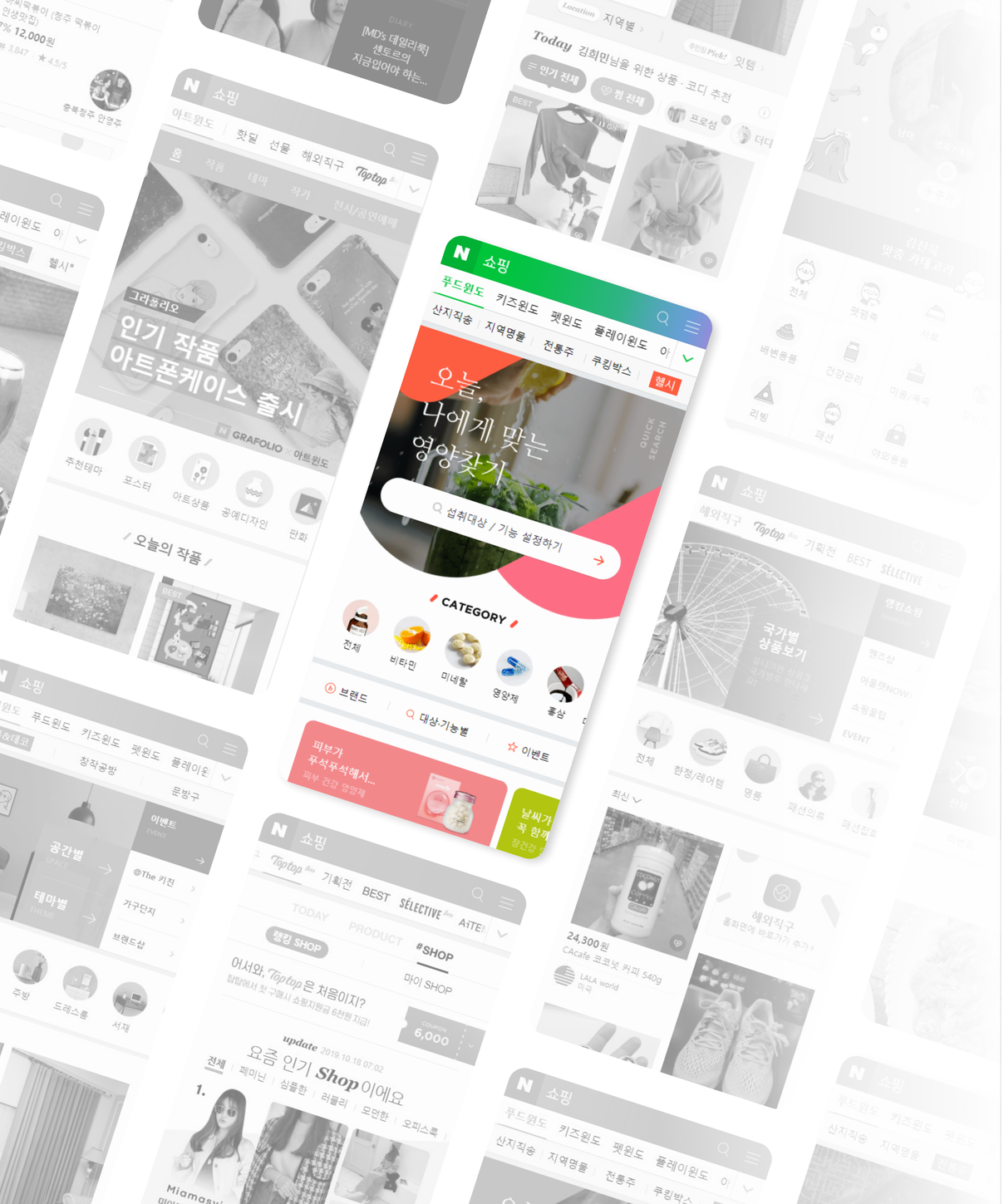


DEVELOPER

KEUN BAE BAE.

keunbae.bae@navercorp.com

1.
From Java to JS



DEVIEW
2019

NAVER SHOPPING DISPLAY

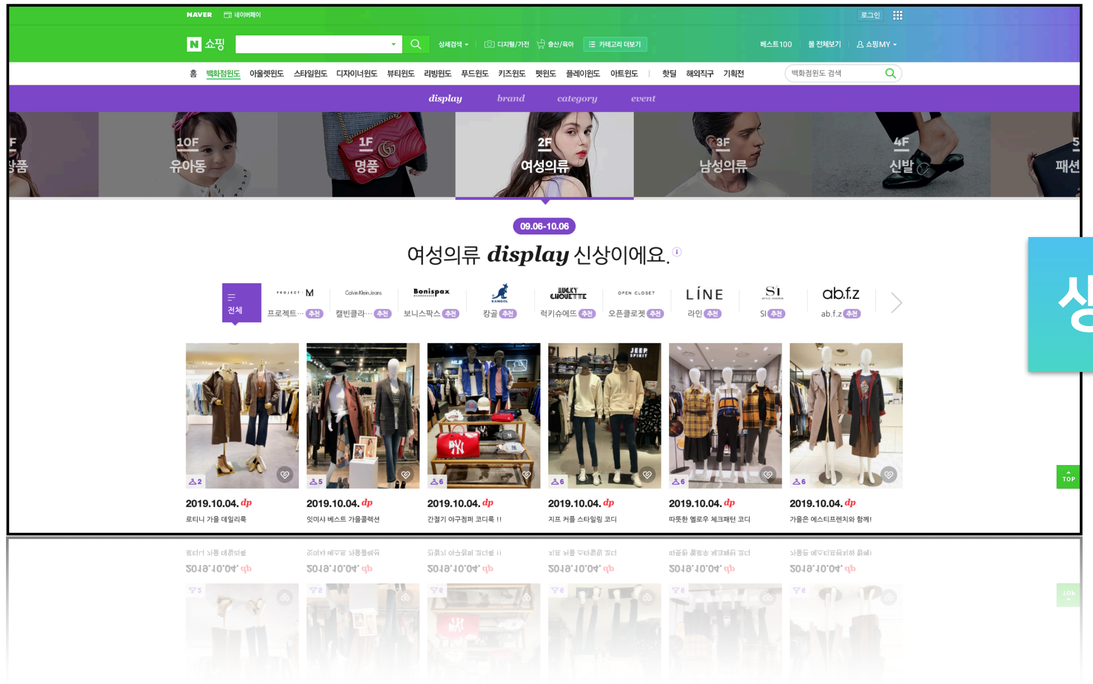
상품 수 약 1억개

약 50만개의 스토어

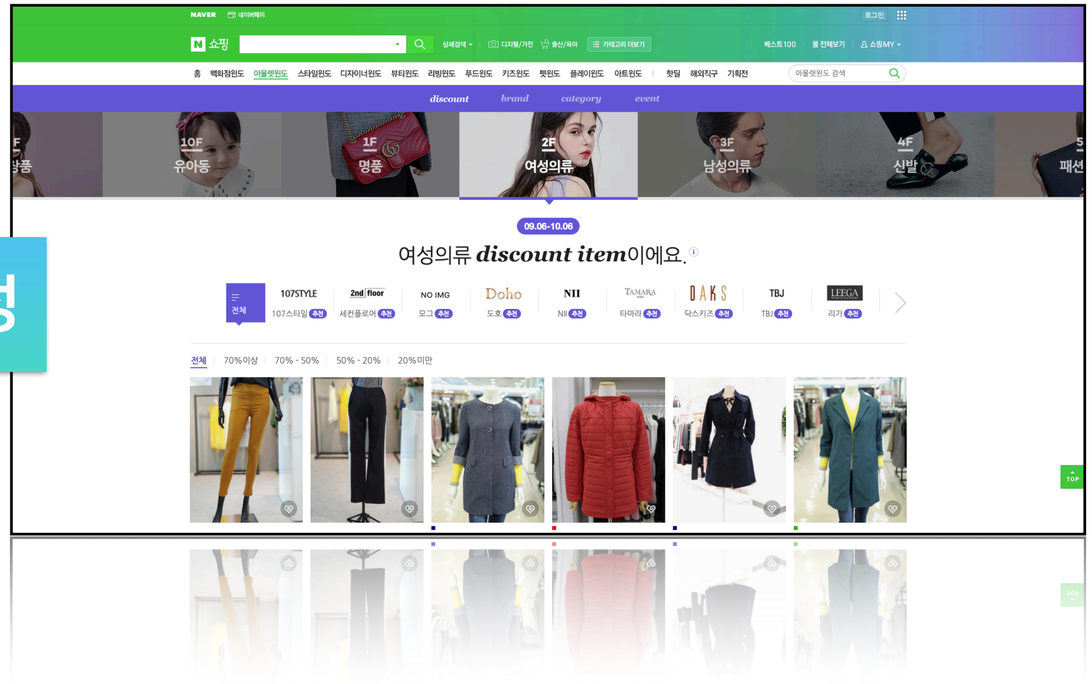
월간 약 15억명의 PV

NAVER SHOPPING

DEVIEW
2019



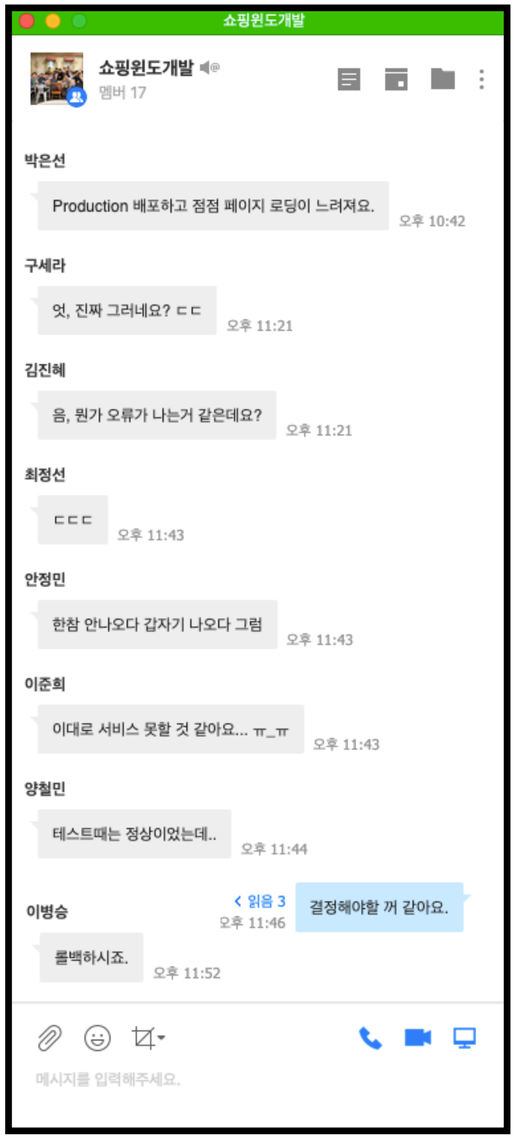
생산성



[이미지 출처] 타짜 곽철용좌 / 곽철용 짤 생성기
https://wormwlrm.github.io/kwakcheolyong/

From Java to JS

DEVIEW 2019



2018.06.28

D+15

2018.07.12

What's wrong?

Debug Monitoring Log

What's wrong?

NAVER SHOPPING

Change Platform

Visibility

We will find a way. We always have.

우리는 답을 찾을 것이다. 늘 그랬듯이

DEVIEW
2019

2.
쇼핑개발자가 A.P.M Library 를 만든 이유

What is APM?

Application Performance Management

Compare between node apm

DEVIEW 2019

New Relic

세계적으로 유명한 APM 명가

Elastic

Elastic 에서 제공하는 Open Source

Skyapm

Apache SkyWalking 을 사용

[이미지 출처] Google Image Thumb

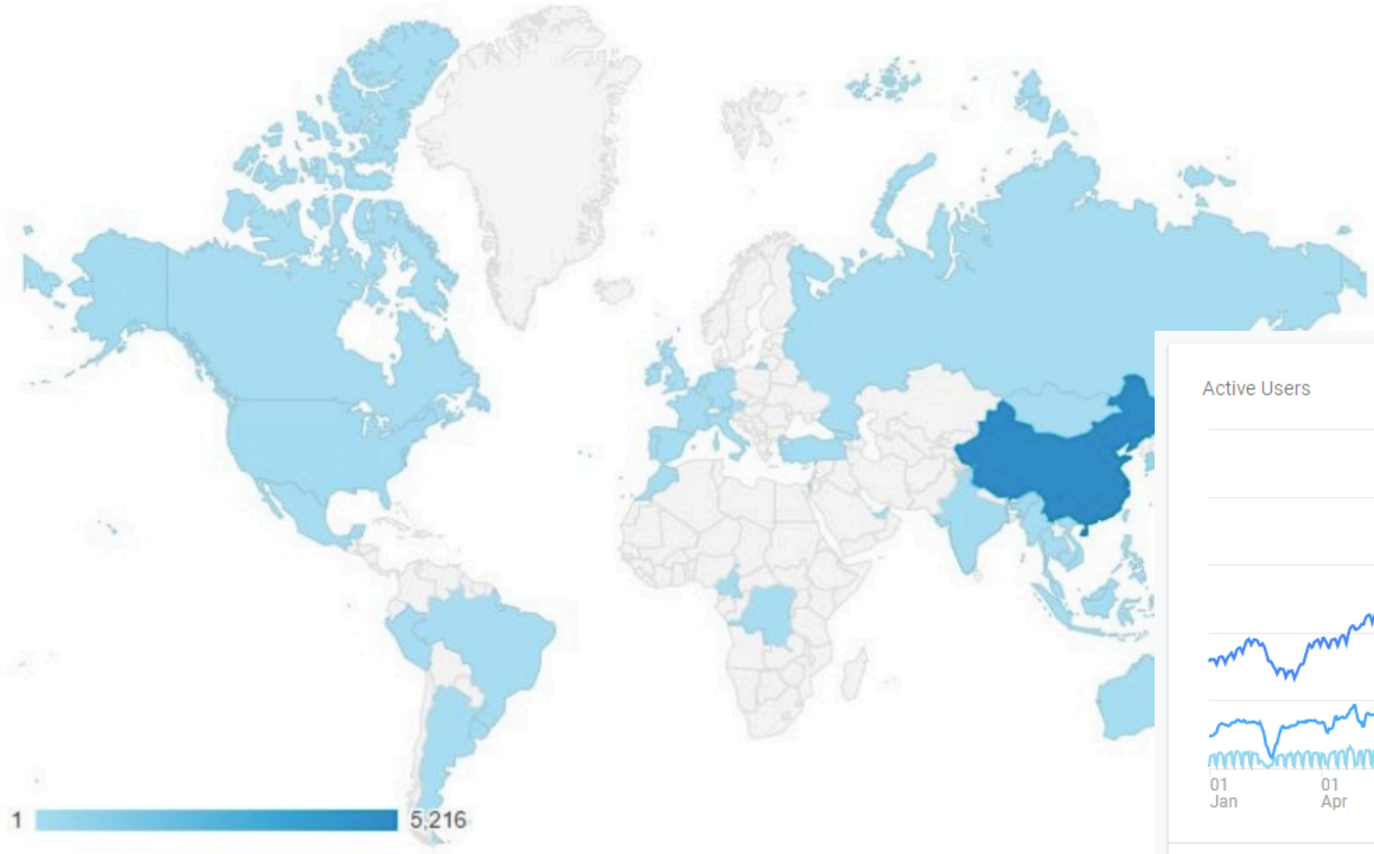
new relic : twitter / elastic : elastic-apm-pacakge / skyapm : GitHub thumb

About Pinpoint

DEVIEW
2019

About Pinpoint

DEVIEW 2019

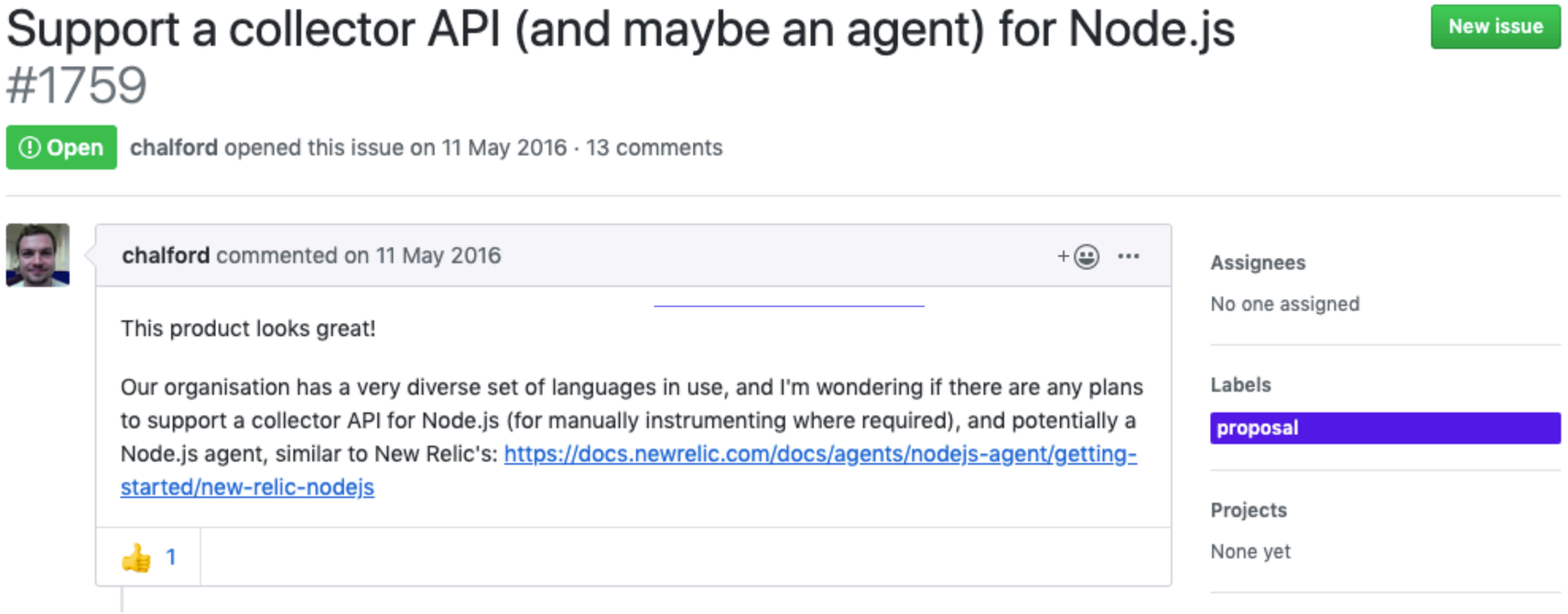


DEVIEW
2019

3.
모니터링 맛집,
3가지 비밀 레시피

Why did we develop it?

DEVIEW 2019



[출처] GitHub / pinpoint
https://github.com/naver/pinpoint/issues/1759

Recipe. 1

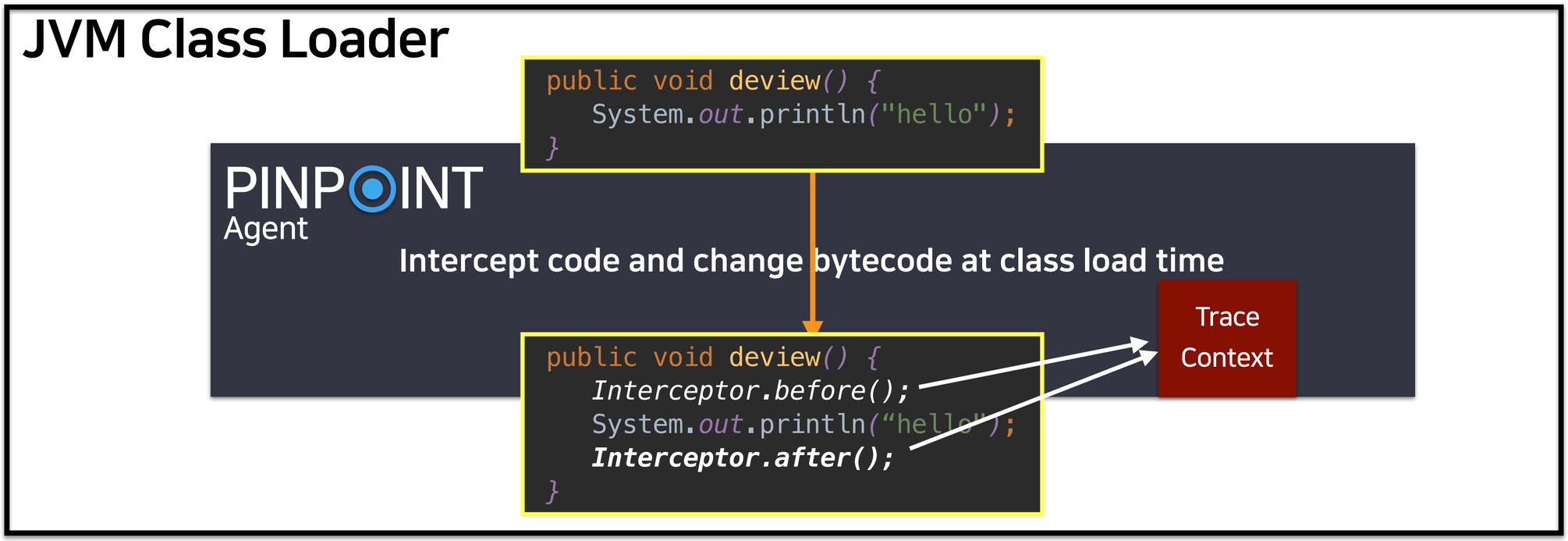
DEVIEW
2019

Monkey patching

Asynchronous tracking

Apply modules

Monkey Patching



BCI (bytecode instrumentation)

Monkey Patching

Code Coverage in V8

```
function foo (deview) {
  if (deview) {
  // do something with ‘deview'.
  } else {
  // do something else.
  }
}
```

```
function foo(deview) {
  cov_2mofekog2n.f[0]++;
  cov_2mofekog2n.s[0]++;
  if (deview) {
    // do something with 'deview'.
    cov_2mofekog2n.b[0][0]++;
  } else {
    // do something else.
    cov_2mofekog2n.b[0][1]++;
  }
}
```

[이미지 출처] v8
https://v8.dev

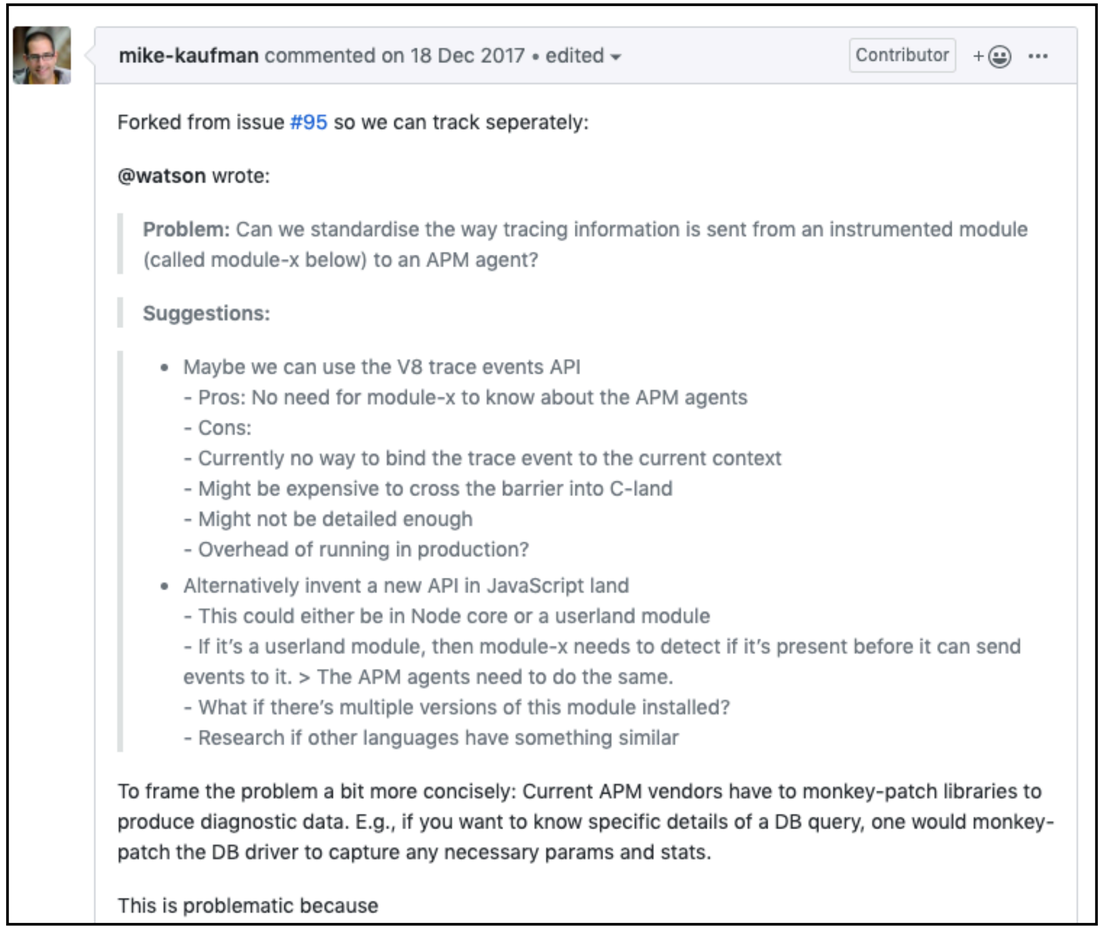
Monkey Patching

Many A.P.M used Monkey Patching

[이미지 출처] Google Image Thumb
new relic : twitter / elastic : elastic-apm-pacakge

Monkey Patching

DEVIEW 2019



mike-kaufman commented on 18 Dec 2017 • edited ▾ Contributor +😀 ···

Forked from issue #95 so we can track seperately:

@watson wrote:

> **Problem:** Can we standardise the way tracing information is sent from an instrumented module (called module-x below) to an APM agent?
>
> **Suggestions:**
>
> - Maybe we can use the V8 trace events API
>   - Pros: No need for module-x to know about the APM agents
>   - Cons:
>   - Currently no way to bind the trace event to the current context
>   - Might be expensive to cross the barrier into C-land
>   - Might not be detailed enough
>   - Overhead of running in production?
> - Alternatively invent a new API in JavaScript land
>   - This could either be in Node core or a userland module
>   - If it's a userland module, then module-x needs to detect if it's present before it can send events to it. > The APM agents need to do the same.
>   - What if there's multiple versions of this module installed?
>   - Research if other languages have something similar

To frame the problem a bit more concisely: Current APM vendors have to monkey-patch libraries to produce diagnostic data. E.g., if you want to know specific details of a DB query, one would monkey-patch the DB driver to capture any necessary params and stats.

This is problematic because

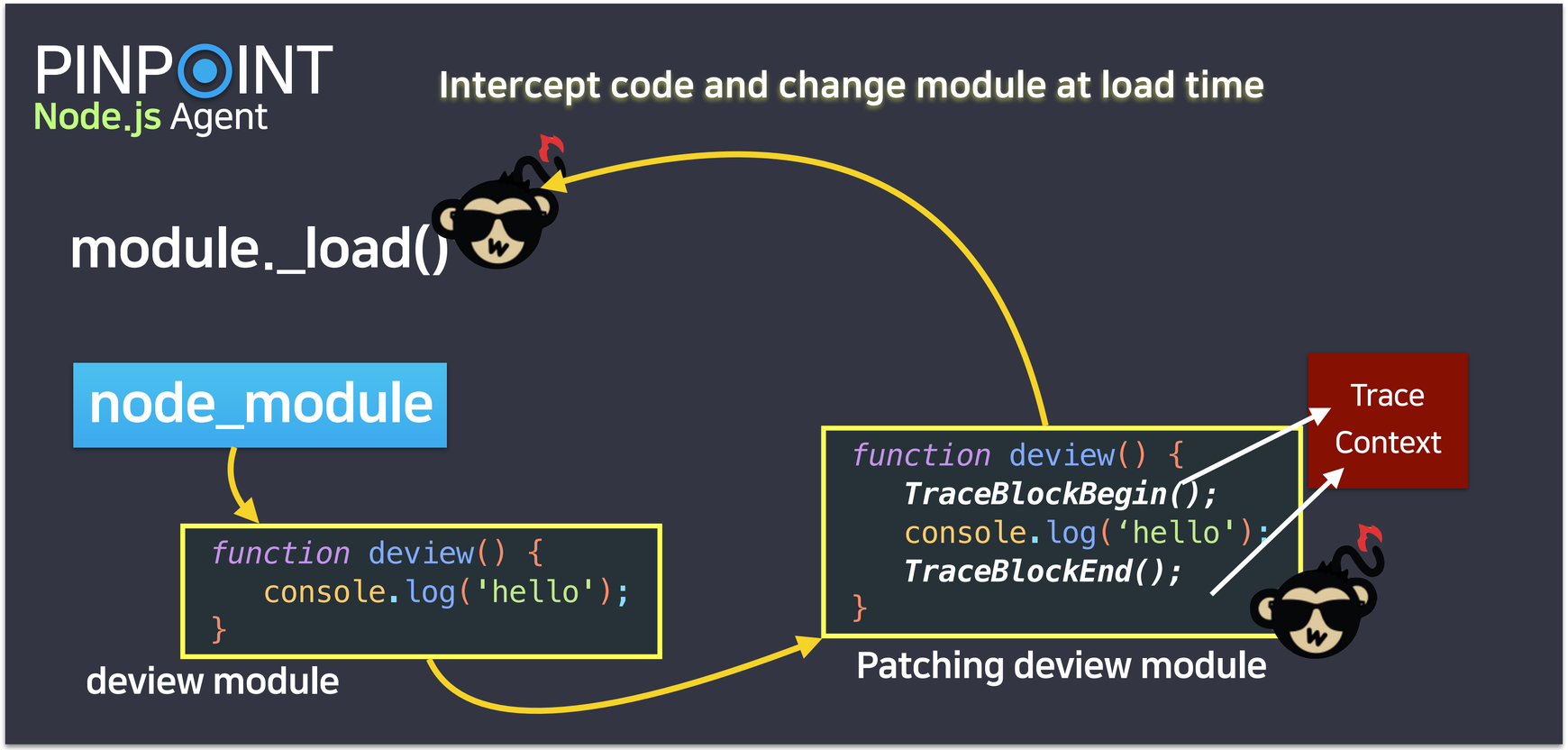
[출처] GitHub / node.js
https://github.com/nodejs/diagnostics/issues/134#issue-282734020

Monkey Patching

A monkey patch is a way for a program to extend or modify supporting system software locally.
(affecting only the running instance of the program)

Module

Monkey Patching



Monkey Patching

Monkey Patching

DEVIEW
2019

```
function init(agent) {
  try {
    const Module = require('module')
    shimmer.wrap(Module, '_load', function (original) {
      return function (name) {
        const m = original.apply(this, arguments)
        if (MODULES.includes(name) && agent.includedModules(name)) {
          try {
            const version = _getModuleVersion(name)
            require('./module/' + name)(agent, version, m)
            agent.setModules(name)

            log.info('loader ==> %s (%s)', name, version)
          } catch (e) {
            log.error('fail to load:', e)
          }
        }
        return m
      }
    })
  } catch (e) {
    log.error('error occurred', e)
  }
}
```

Monkey Patching

DEVIEW
2019

```
function init(agent) {
  try {
    const Module = require('module')
    shimmer.wrap(Module, '_load', function (original) {
      return function (name) {
        const m = original.apply(this, arguments)
        if (MODULES.includes(name) && agent.includedModules(name)) {
          try {
            const version = _getModuleVersion(name)
            require('./module/' + name)(agent, version, m)
            agent.setModules(name)
          } catch (e) {
            log.error('fail to load:', e)
          }
        }
        return m
      }
    })
  } catch (e) {
    log.error('error occurred', e)
  }
}
```

Recipe. 2

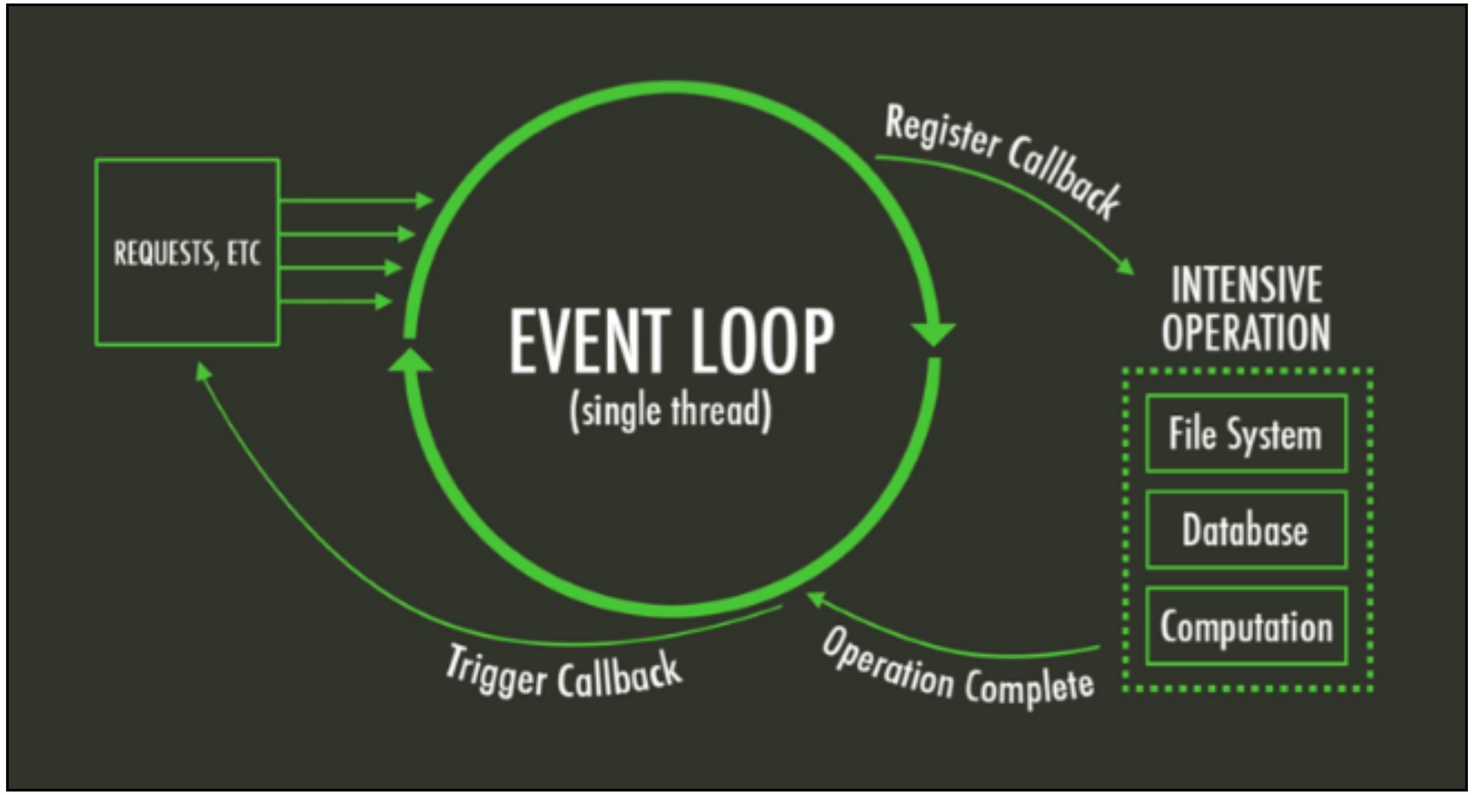
DEVIEW
2019

Monkey patching

Asynchronous tracking

Apply modules

Asynchronous Tracking



[이미지 출처] Google Image
https://codeburst.io/asynchronous-adventures-with-node-js-5c7463970efd

async_hooks

Asynchronous Tracking

DEVIEW 2019

```
const fs = require('fs')
const http = require('http')
const async_hooks = require('async_hooks')

const log = (str) => fs.writeSync(1, `${str}\n`)

async_hooks.createHook({
  init(asyncId, type, triggerAsyncId) {
    log(`asyncId: ${asyncId} / triggerAsyncId: ${triggerAsyncId}`)
  },
}).enable()

const deviewResponse = (res) => {
  setTimeout(() => {
    log(`Inside deviewResponse1. [executionAsyncId] :${async_hooks.executionAsyncId()}`)
    setTimeout(() => {
      log(`Inside deviewResponse2. [executionAsyncId] :${async_hooks.executionAsyncId()}`)
    }, 1);
  }, 0);
  res.end('console.warn(\'are you worried?\')')
}

const requestHandler = (req, res) => {
  log(`>> Inside request: [executionAsyncId]: ${async_hooks.executionAsyncId()}`)
  deviewResponse(res)
}

const server = http.createServer(requestHandler)

server.listen(8080)
```

Asynchronous Tracking

DEVIEW 2019

```
const fs = require('fs')
const http = require('http')
const async_hooks = require('async_hooks')

const log = (str) => fs.writeSync(1, `${str}\n`)

async_hooks.createHook({
  init(asyncId, type, triggerAsyncId) {
    log(`asyncId: ${asyncId} / triggerAsyncId: ${triggerAsyncId}`)
  },
}).enable()

const deviewResponse = (res) => {
  setTimeout(() => {
    log(`Inside deviewResponse1. [executionAsyncId] :${async_hooks.executionAsyncId()}`)
    setTimeout(() => {
      log(`Inside deviewResponse2. [executionAsyncId] :${async_hooks.executionAsyncId()}`)
    }, 1);
  }, 0);
}

const requestHandler = (req, res) => {
  log(`>> Inside request: [executionAsyncId]: ${async_hooks.executionAsyncId()}`)
  deviewResponse(res)
}

const server = http.createServer(requestHandler)

server.listen(8080)
```

19

33

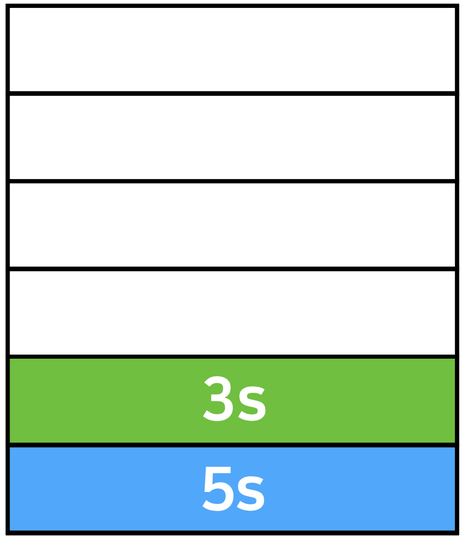
11

```
>> Inside request: [executionAsyncId]: 11
asyncId: 19 / triggerAsyncId: 11
Inside deviewResponse1. [executionAsyncId] :19
asyncId: 33 / triggerAsyncId: 19
Inside deviewResponse2. [executionAsyncId] :33
```

Asynchronous Tracking Issue

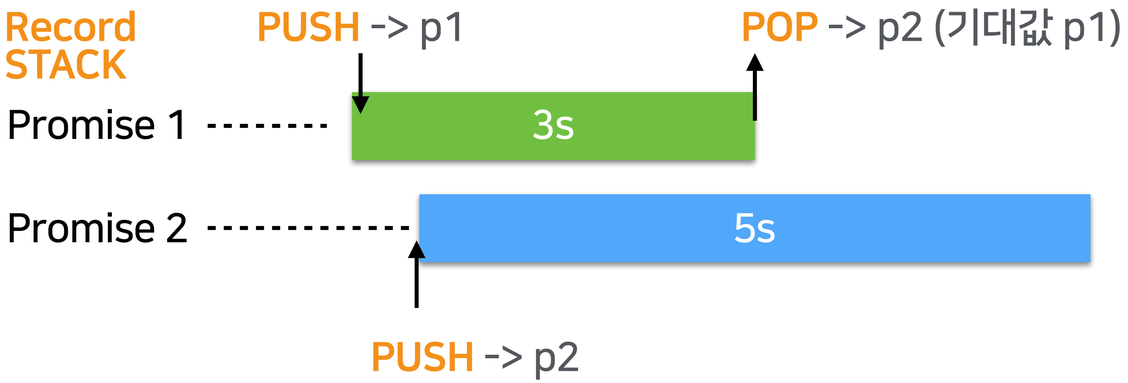
```
app.get('/test/express', function (req, res) {

    let p1 = rp('http://localhost:9998/wait/3s');
    let p2 = rp('http://localhost:9998/wait/5s');

    Promise.all([p1, p2]).then(() => {
        res.send('Hello Deview2019!’);
    })
});
```

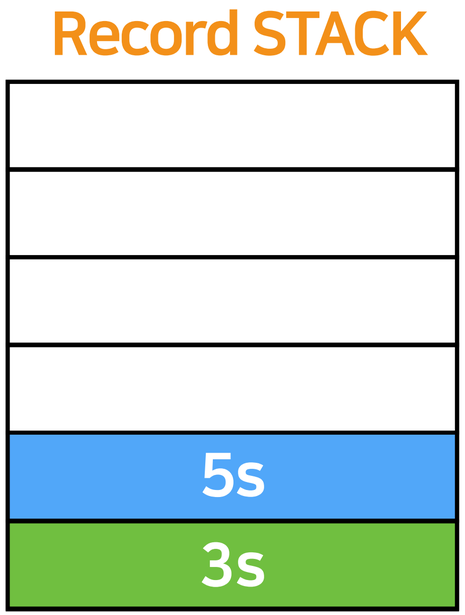
Record STACK



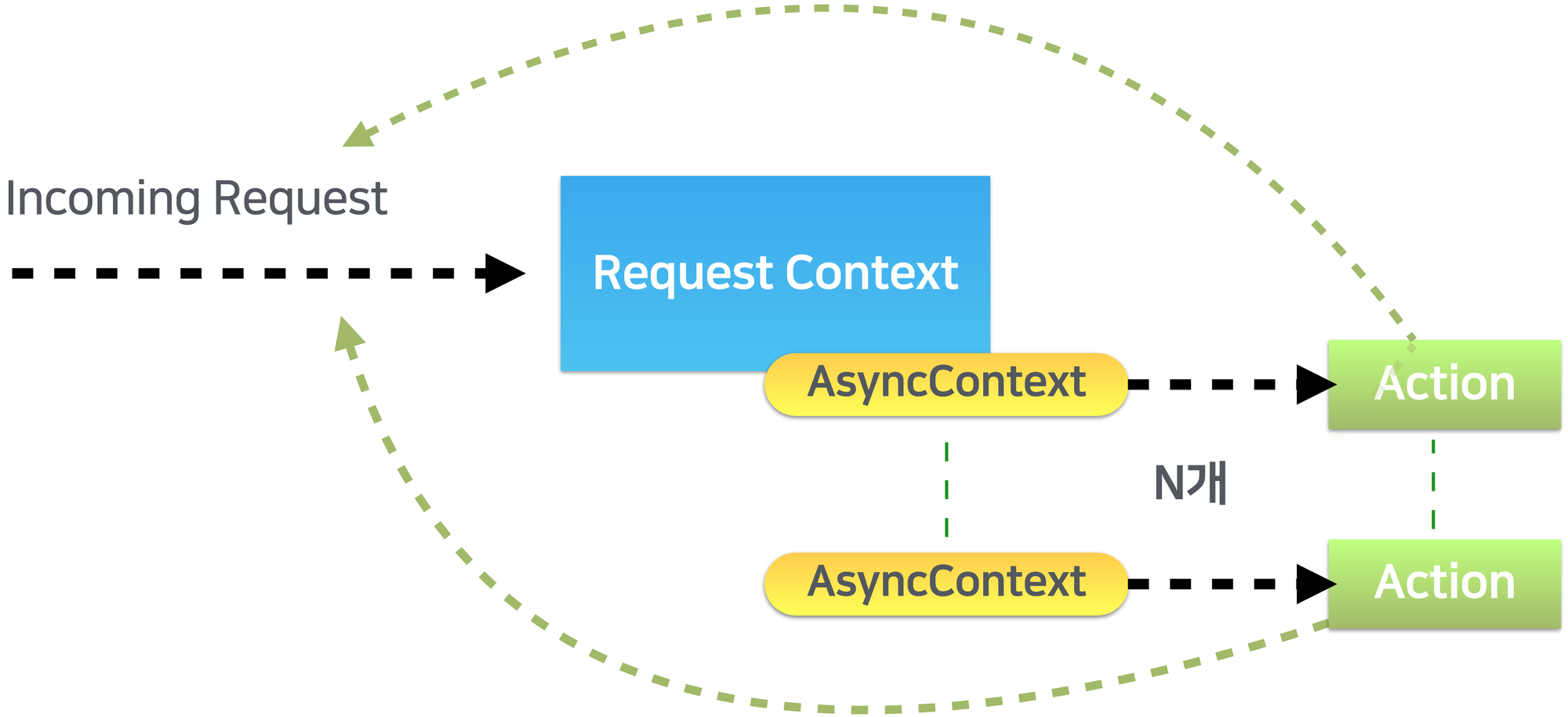
Asynchronous Tracking


```
app.get('/test/express', function (req, res) {

    let p1 = rp('http://localhost:9998/wait/3s');
    let p2 = rp('http://localhost:9998/wait/5s');

    Promise.all([p1, p2]).then(() => {
        res.send('Hello Deview2019!');
    })
});
```



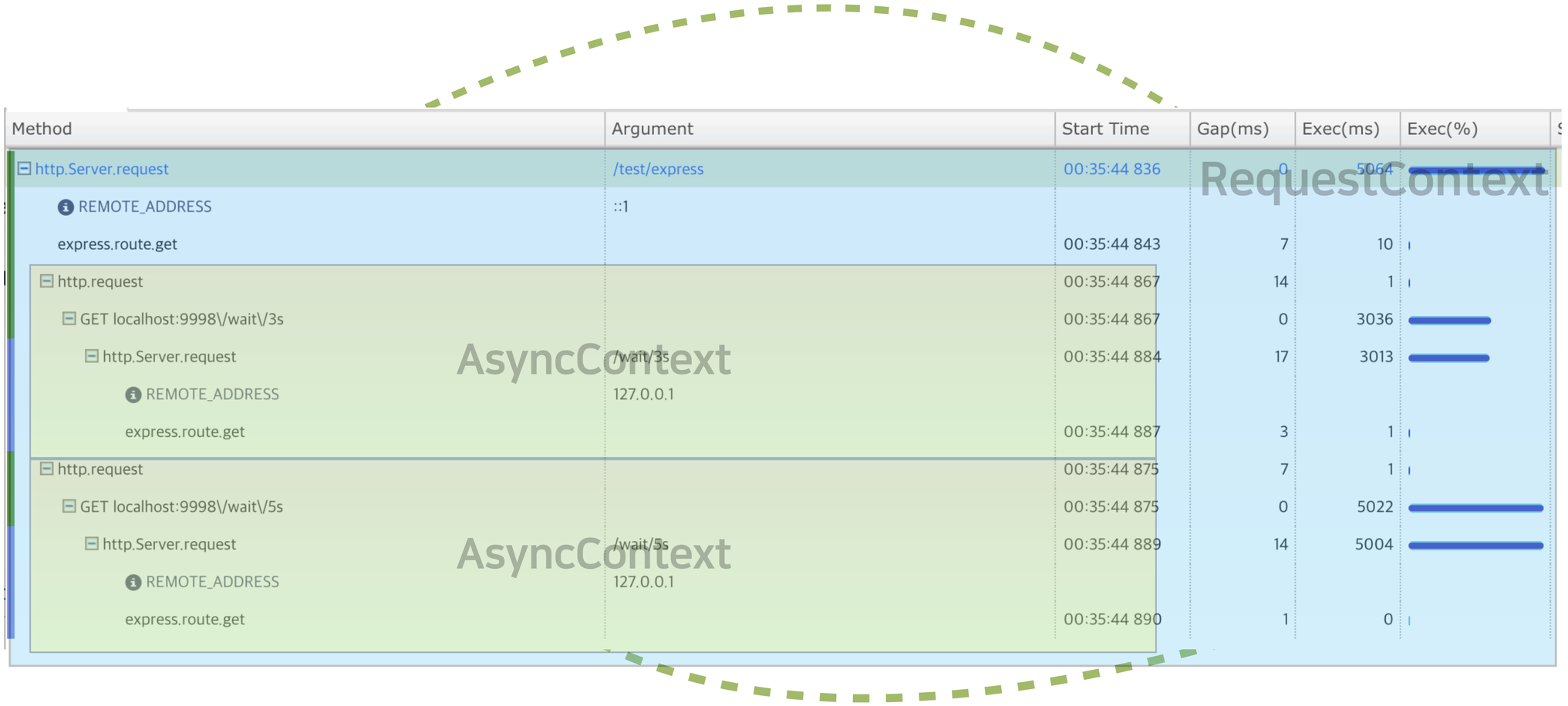


Asynchronous Tracking



Asynchronous Tracking

DEVIEW 2019



Asynchronous Tracking

DEVIEW
2019

```
      … (중략) …
    if ( name !== 'next' || !this[firstTrace]) {
      spanEventRecorder = trace.traceBlockBegin()
      … (중략) …
      trace.traceBlockEnd(spanEventRecorder)
      asyncTrace = trace.newAsyncTrace(spanEventRecorder)
    }
    if (trace && asyncEventRecorder) {
      arguments[0] = wrappedCallback
      … (중략) …
    }
  }
}
return original.apply(this, arguments)

function wrappedCallback () {
  if (asyncTrace) {
    asyncTrace.traceAsyncEnd(asyncEventRecorder)
  }
  return cb.apply(this, arguments)
}
```

Asynchronous Tracking

DEVIEW 2019

```
        (중략)
  spanEventRecorder = trace.traceBlockBegin()
  … (중략) …
  trace.traceBlockEnd(spanEventRecorder)
      asyncTrace = trace.newAsyncTrace(spanEventRecorder)
  asyncTrace = trace.newAsyncTrace(spanEventRecorder)
    if (trace && asyncEventRecorder) {
      arguments[0] = wrappedCallback
      … (중략) …
    }
  }
}
return original.apply(this, arguments)

function wrappedCallback () {
  if (asyncTrace) {
    asyncTrace.traceAsyncEnd(asyncEventRecorder)
  }
}
```

Recipe. 3

DEVIEW 2019

Monkey patching

Asynchronous tracking

Apply modules

Apply modules

STEP1

개발해야 할 NPM module 을 선택합니다.

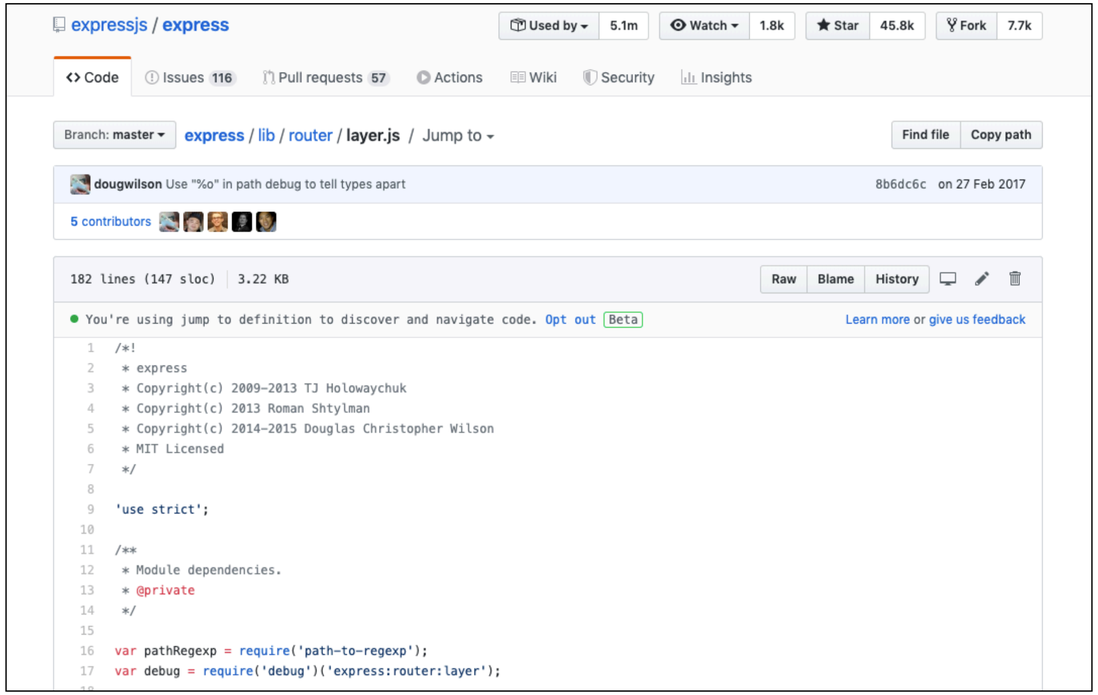
[출처] 40 NPM Modules We Can't Live Without
https://medium.com/startup-frontier/40-npm-modules-we-can-t-live-without-36e29e352e3a

Apply modules

DEVIEW 2019

STEP2

tracing 할 수 있게 module 를 분석합니다.

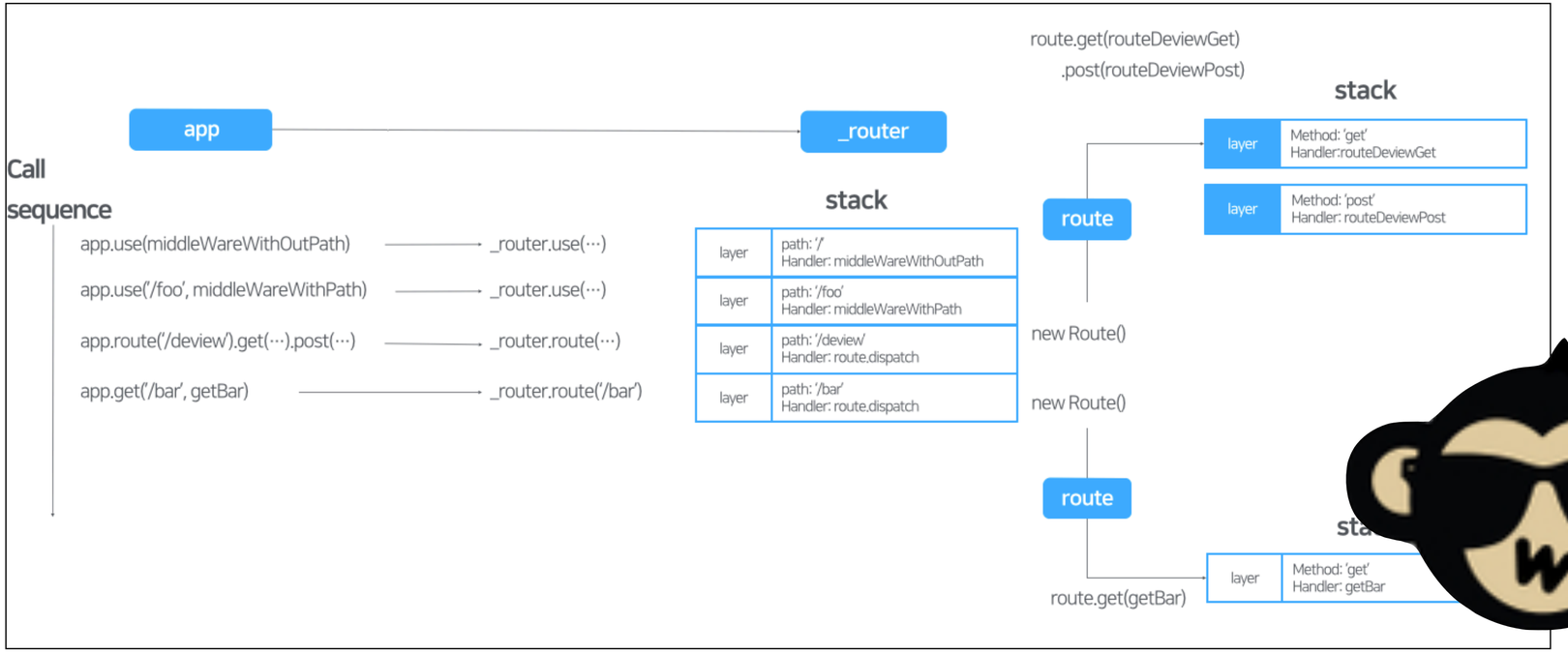


Apply modules

DEVIEW 2019

STEP3

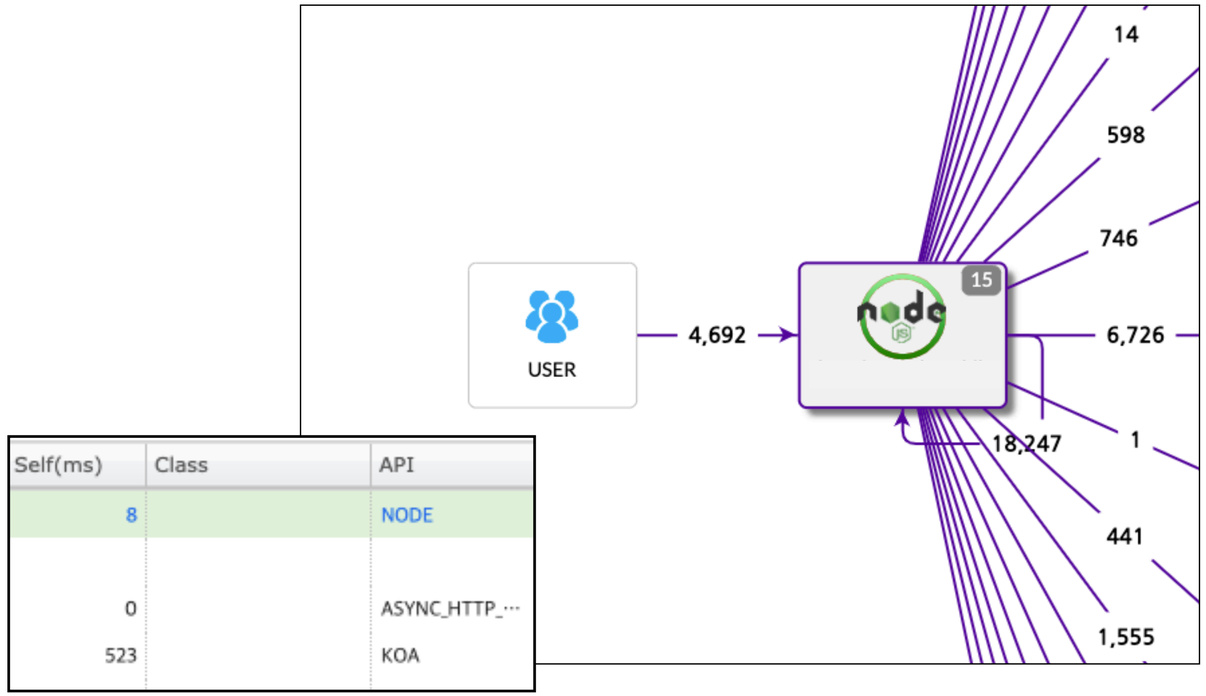
분석이 완료된 module 에 적절히 Monkey Patching!



Apply modules

DEVIEW
2019

짠!

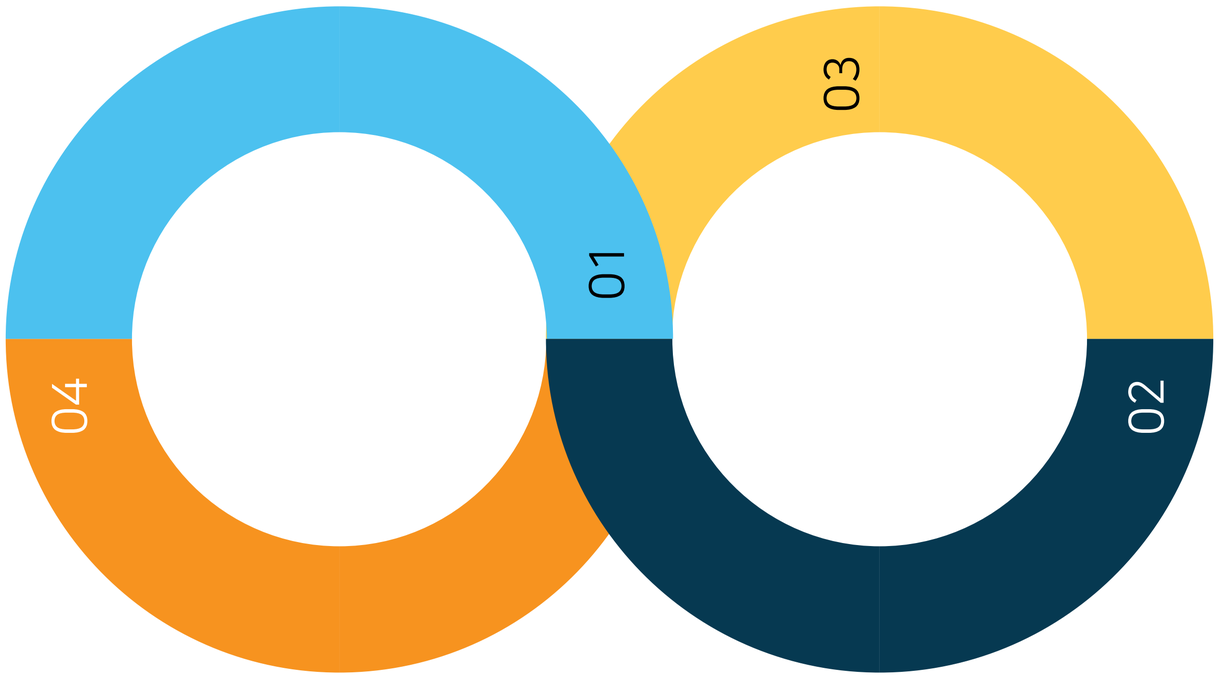


Apply modules

DEVIEW
2019

정리하자면,

NPM 을
결정하고



Monkey
Patching

Test &
Test

Tracing 을 위해
module 를 분석하고

Apply modules

DEVIEW 2019

Express 분석과정

two main api

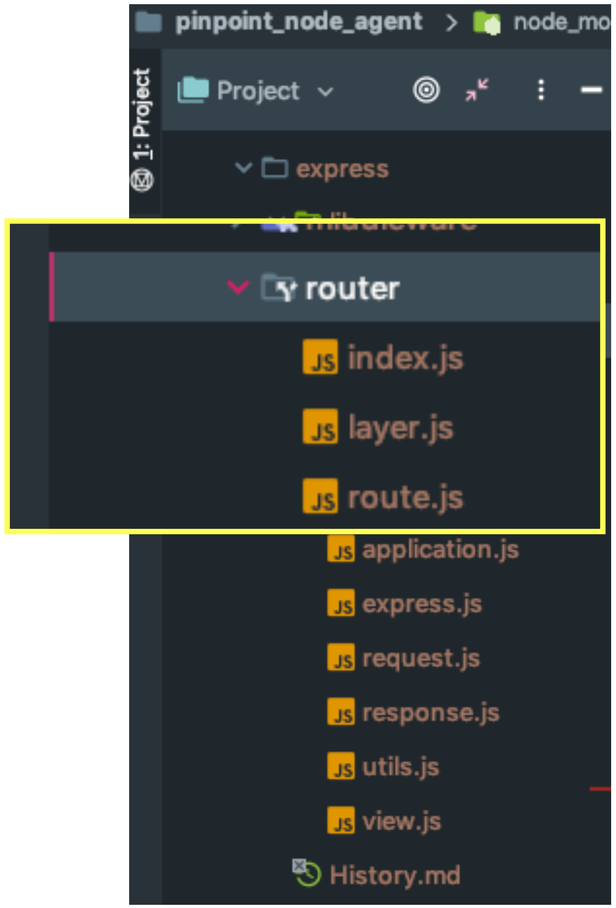
middleware

```
const app = new express()
// third party middleware
app.use(express.json())
// 사용자 정의 미들웨어
app.use(function (req, res, next) {
  console.log('Time:', Date.now())
  next()
})
// 특별한 url path 를 사용한 미들웨어
app.use('/user/:id', function (req, res, next) {
  validUser(req.params.id)
  next()
})
```

routing

```
const app = new express()
// GET /foo 정의
app.get('/foo', function (req, res, next) {
  res.end('foo')
})
// 라우트의 그룹 핸들러 정의
app.route('/book')
    .get(function (req, res) {
      res.send('Get a random music')
    })
    .post(function (req, res) {
      res.send('Add a music')
    })
    .put(function (req, res) {
      res.send('Update the music')
    })
```

Apply modules



Express 분석과정

express 의 structure

All internal logic is inside the **lib** folder

application.js defines the express application

the **middleware** and **routing** logic is in the **router** folder.

Apply modules

DEVIEW
2019

Express 분석과정

sample source

```
const express = require('express')
const app = express()
// 사용자 정의 미들웨어
app.use(function middlewareWithoutPath(req, res, next) {
  console.log('use middleware without path')
  next()
})
// 특별한 url path 를 사용한 미들웨어
app.use('/foo', function middlewareWithPath(req, res, next) {
  console.log('use middleware with path')
  next()
})
// 라우트의 그룹 핸들러 정의
app.route('/deview')
    .get(function routeDeviewGet(req, res) {
      res.send('Good')
    })
    .post(function routeDeviewPost(req, res) {
      res.send('Very Good')
    })
```

Apply modules

DEVIEW
2019

Express 분석과정

sample source

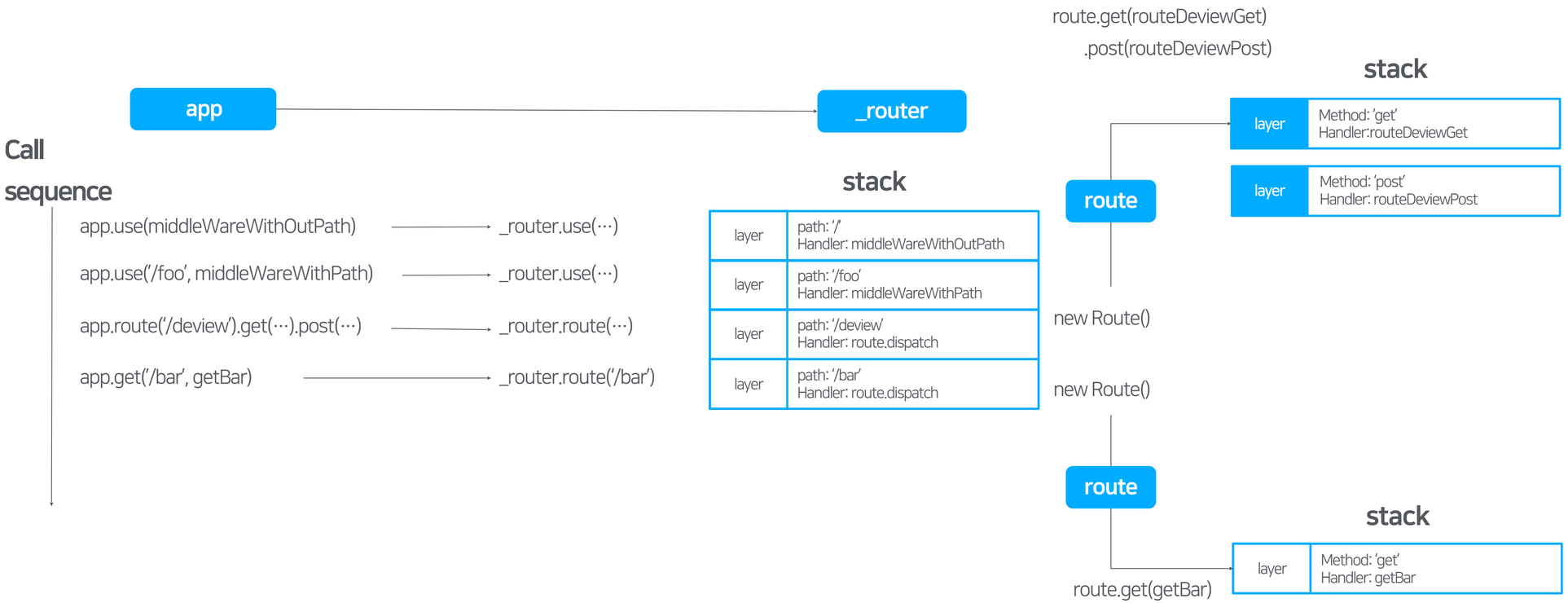
```
// 라우트의 그룹 핸들러 정의
app.route('/deview')
    .get(function routeDeviewGet(req, res) {
      res.send('Good')
    })
    .post(function routeDeviewPost(req, res) {
      res.send('Very Good')
    })
// GET /bar 정의
app.get('/bar', function getZoo(req, res) {
  res.send('ok')
})

app.listen(3000, () => {
  console.log('listen on 3000...')
})
```

Apply modules

DEVIEW 2019

Express 분석과정

inside application



Apply modules

Express 분석과정

three **key** point

Route

http method 일치 시 처리
경로 자체에 **Stack** 이 있음

The Router

app.use(path, fn)
_router는
새로운 Layer 로 Stack Push

Layer

가장 간단한 구성 요소
Handle 존재

Apply modules

DEVIEW 2019

Express 분석과정

middleware

```
shimmer.wrap(express.Router, 'use', function (original) {
  return function () {
    const result = original.apply(this, arguments)
    const fn = arguments && arguments[1]
    if (fn && fn.name !== 'router' && this.stack && this.stack.length) {
      log.debug('>> [Express] express.Router.use ', this.stack[this.stack.length - 1])
      const layer = this.stack[this.stack.length - 1]
      doPatchLayer(layer, 'middleware', layer.name)
    }
    return result
  }
})
```

Apply modules

DEVIEW 2019

Express 분석과정

middleware

```
shimmer.wrap(express.Router, 'use', function (original) {
  return function () {
    const result = original.apply(this, arguments)
    const fn = arguments && arguments[1]
    if (fn && fn.name !== 'router' && this.stack && this.stack.length) {
      log.debug('>> [Express] express.Router.use ', this.stack[this.stack.length - 1])
      const layer = this.stack[this.stack.length - 1]
      doPatchLayer(layer, 'middleware', layer.name)
    }
    return result
  }
})
```

Apply modules

DEVIEW 2019

Express 분석과정

route

```
shimmer.wrap(express.Router, 'route', function (original) {
  return function () {
    const result = original.apply(this, arguments)
    if (this.stack && this.stack.length) {
      log.debug('>> [Express] express.Router.route ', this.stack[this.stack.length - 1])
      const layer = this.stack[this.stack.length - 1]
      doPatchLayer(layer, 'route', layer.name)
    }
    return result
  }
})
```

Apply modules

DEVIEW 2019

Express 분석과정

route

```
shimmer.wrap(express.Router, 'route', function (original) {
  return function () {
    const result = original.apply(this, arguments)
    if (this.stack && this.stack.length) {
      log.debug('>> [Express] express.Router.route ', this.stack[this.stack.length - 1])
      const layer = this.stack[this.stack.length - 1]
      doPatchLayer(layer, 'route', layer.name)
    }
    return result
  }
})
```

Apply modules

DEVIEW
2019

Express 분석과정

layer

```
function doPatchLayer(layer, moduleName, methodName) {
  shimmer.wrap(layer, 'handle', function(original) {
    return (original.length === 4)
      ? recordErrorHandle(original, moduleName, methodName)
      : recordHandle(original, moduleName, methodName)
  })
}
```

Apply modules

DEVIEW
2019

Express 분석과정

layer

```
function doPatchLayer(layer, moduleName, methodName) {
  shimmer.wrap(layer, 'handle', function(original) {
    return (original.length === 4)
      ? recordErrorHandle(original, moduleName, methodName)
      : recordHandle(original, moduleName, methodName)
  })
}
```

Problem & Resolution

Mission1: Duplicate error in middleware.

Mission2: Anonymous function.

Mission3: a lot of modules.

DEVIEW
2019

4. Sharing

사용법

ES6

```
import 'pinpoint-node-agent'
```

CommonJS

```
require('pinpoint-node-agent')
```

사용법

```
env: {
        "PINPOINT_ENABLE": "true",
        "PINPOINT_LOG_LEVEL": "INFO",
        "PINPOINT_APPLICATION_NAME": “{ name }”,
        "PINPOINT_COLLECTOR_IP": “{ collector_IP}”,
}
```

Sharing

DEVIEW 2019

```
import * as Sentry from '@sentry/node'

import { createServer } from 'http'

import { getFormattedDate } from '@shopping/common'
import { redisConnect, disconnectMongo, mongoConnect } from '@shopping/server'
import logger from 'common/logger'
import { initKoaMiddleware } from 'common/server'
import hostConfig from 'common/config'
import projectConfig from 'common/config/project'
import urlConfig from '../server/config'

let server
export let redisClient
```

Sharing

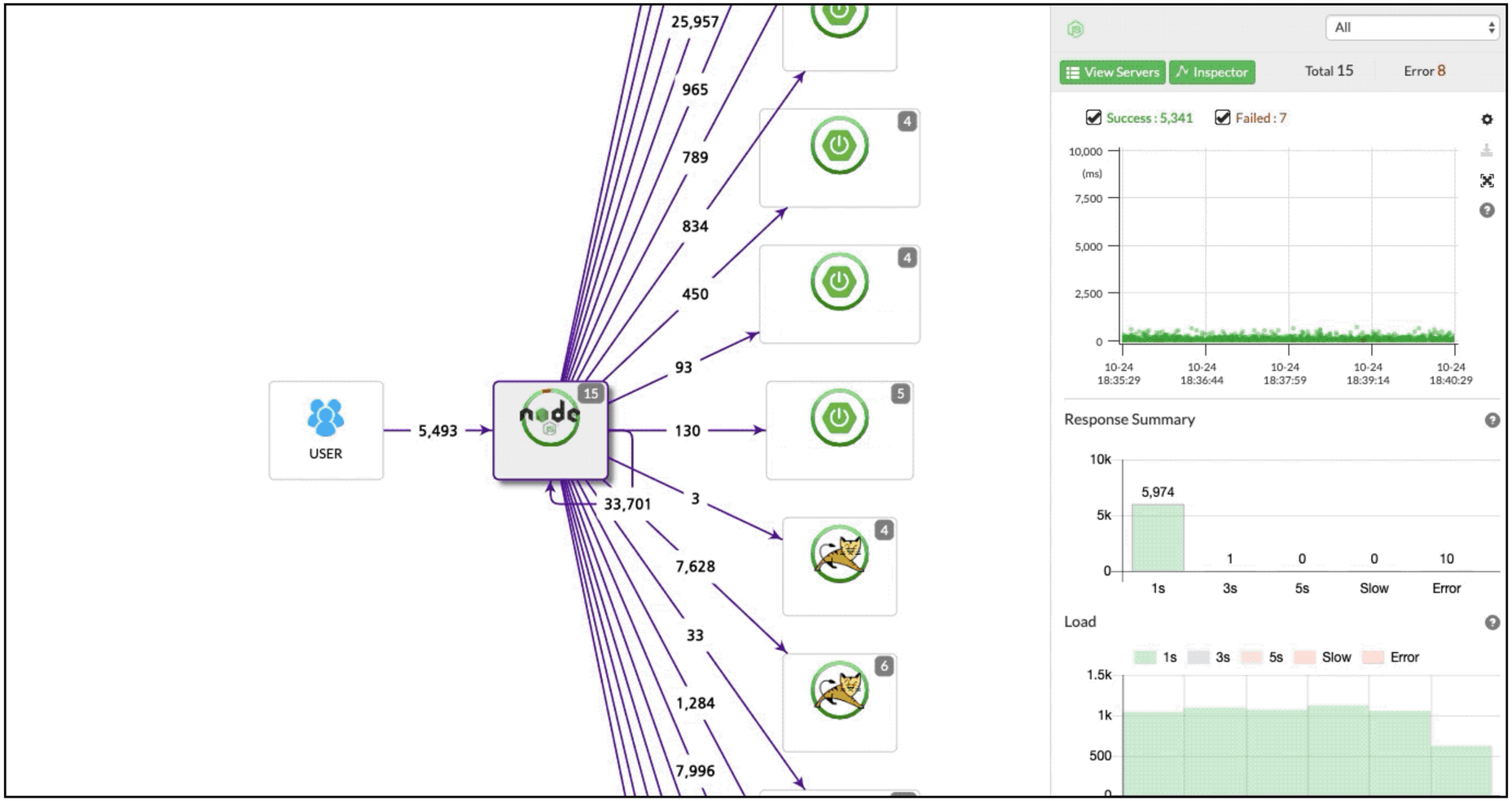
DEVIEW 2019

```
import 'pinpoint-node-agent'
import * as Sentry from '@sentry/node'

import { createServer } from 'http'

import { getFormattedDate } from '@shopping/common'
import { redisConnect, disconnectMongo, mongoConnect } from '@shopping/server'
import logger from 'common/logger'
import { initKoaMiddleware } from 'common/server'
import hostConfig from 'common/config'
import projectConfig from 'common/config/project'
import urlConfig from '../server/config'

let server
export let redisClient
```

Let’s Go 🏃‍♂️🏃‍♀️

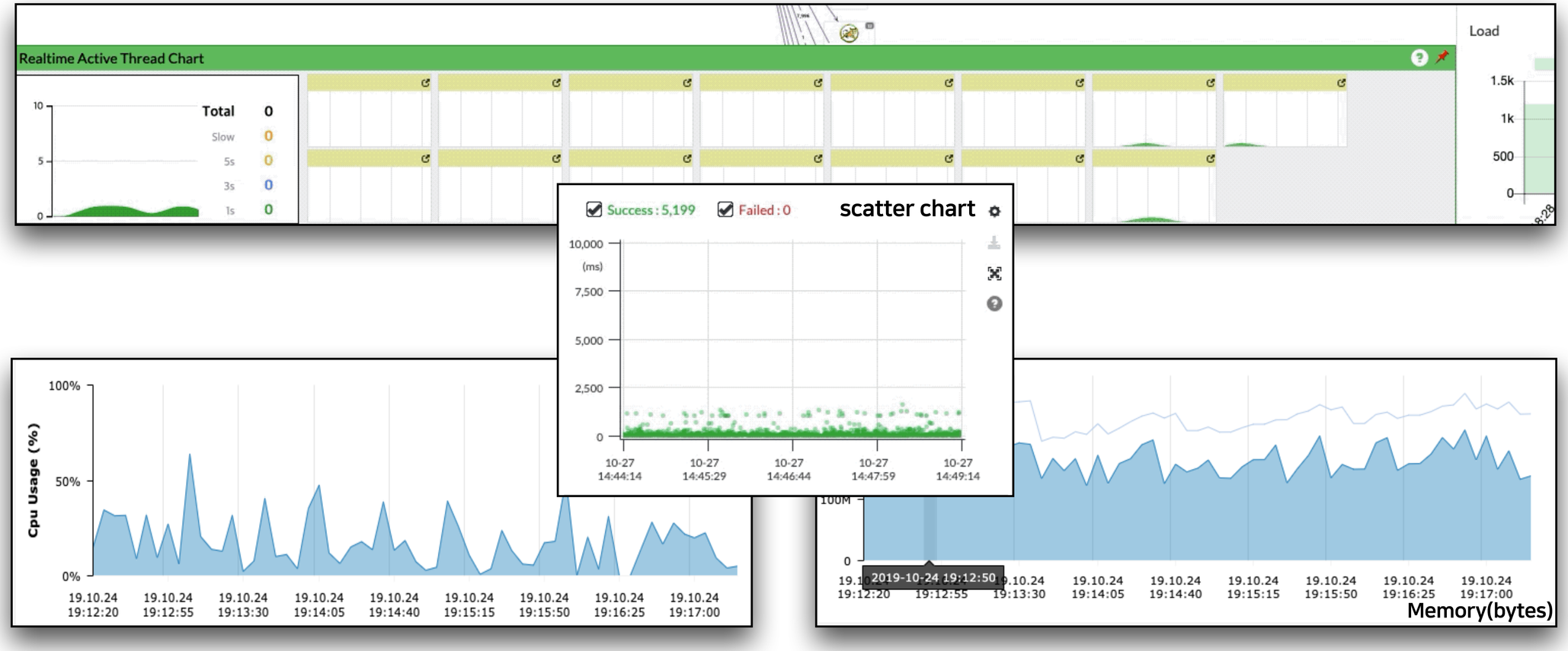
Sharing

DEVIEW 2019



Sharing

DEVIEW 2019



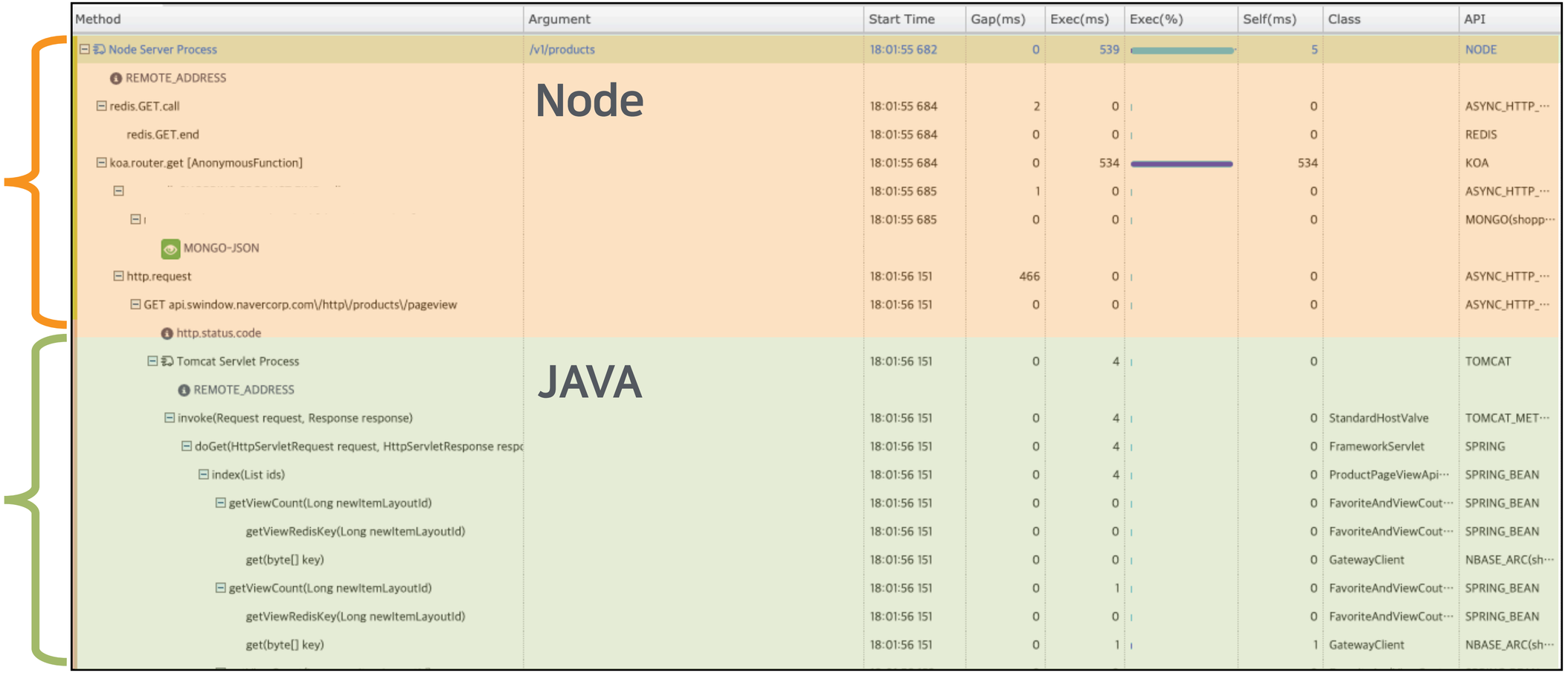
Sharing / Find Error

DEVIEW 2019

Node Server Process	/v1/products/4301959119	17:38:50 732	0	6		2		NODE
REMOTE_ADDRESS								
koa.router.get [AnonymousFunction]		17:38:50 733	1	4		4		KOA
Error	Error: Product with id 4301959119 not found-							
		17:38:50 734	1	0		0		ASYNC_HTTP_···
		17:38:50 734	0	0		0		MONGO(shopp···
MONGO-JSON	{"_id":"4301959119"}							

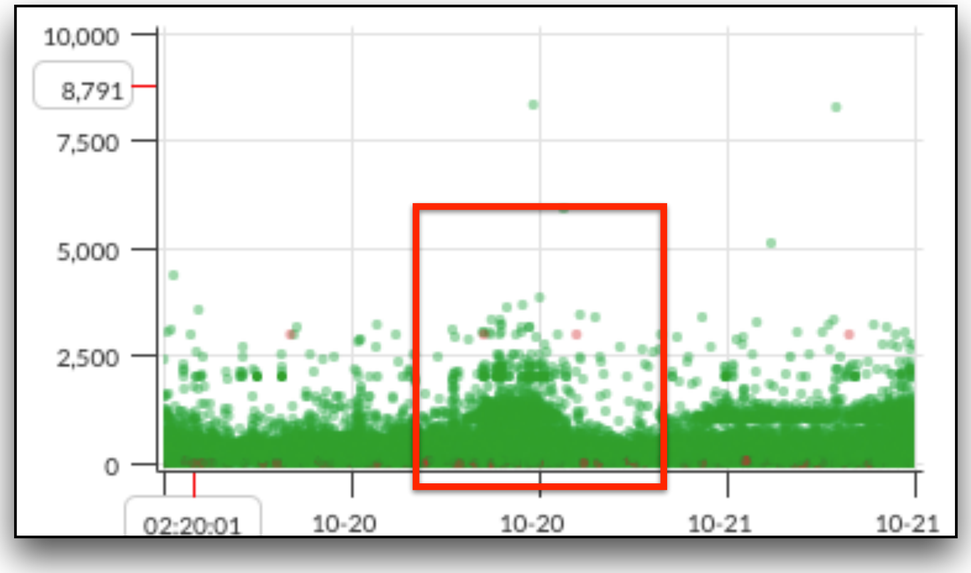
Sharing / Tracking of Distributed Environments

DEVIEW 2019



Method	Argument	Start Time	Gap(ms)	Exec(ms)	Exec(%)	Self(ms)	Class	API
Node Server Process	/v1/products	18:01:55 682	0	539		5		NODE
REMOTE_ADDRESS								
redis.GET.call		18:01:55 684	2	0		0		ASYNC_HTTP_…
redis.GET.end		18:01:55 684	0	0		0		REDIS
koa.router.get [AnonymousFunction]		18:01:55 684	0	534		534		KOA
		18:01:55 685	1	0		0		ASYNC_HTTP_…
		18:01:55 685	0	0		0		MONGO(shopp…
MONGO-JSON								
http.request		18:01:56 151	466	0		0		ASYNC_HTTP_…
GET api.swindow.navercorp.com\/http\/products\/pageview		18:01:56 151	0	0		0		ASYNC_HTTP_…
http.status.code								
Tomcat Servlet Process		18:01:56 151	0	4		0		TOMCAT
REMOTE_ADDRESS								
invoke(Request request, Response response)		18:01:56 151	0	4		0	StandardHostValve	TOMCAT_MET…
doGet(HttpServletRequest request, HttpServletResponse respc		18:01:56 151	0	4		0	FrameworkServlet	SPRING
index(List ids)		18:01:56 151	0	4		0	ProductPageViewApi…	SPRING_BEAN
getViewCount(Long newItemLayoutId)		18:01:56 151	0	0		0	FavoriteAndViewCout…	SPRING_BEAN
getViewRedisKey(Long newItemLayoutId)		18:01:56 151	0	0		0	FavoriteAndViewCout…	SPRING_BEAN
get(byte[] key)		18:01:56 151	0	0		0	GatewayClient	NBASE_ARC(sh…
getViewCount(Long newItemLayoutId)		18:01:56 151	0	1		0	FavoriteAndViewCout…	SPRING_BEAN
getViewRedisKey(Long newItemLayoutId)		18:01:56 151	0	0		0	FavoriteAndViewCout…	SPRING_BEAN
get(byte[] key)		18:01:56 151	0	1		1	GatewayClient	NBASE_ARC(sh…

그래서 뭐가 좋을까요?



빠른 문제해결!

Summary

Monkey Patching

Module

Asynchronous tracking

async_hooks

Apply modules

S.A.P

DEVIEW
2019

Thank You

dl_valhalla@navercorp.com