Omni-channel
Personalization
Platform
www.dable.io

시간당 수백만 요청을 처리하는 node.js 서버 운영기

김군우 (CPO, co-founder at Dable)

2015-11-12

Dable 김군우 (miya.pe.kr)

NCSOFT NHN SK planet

lemonpen

Game SNS(for AION)

Front-end 유틸들
(N-MET, N-WAX, OpenWAX)

댓글 플러그인 Affogato

RecoPick

1. 수백만 요청?
2. 어떻게 운영?
3. 어떻게 개발?
4. 삽질좀 했지?

?

1. 데이블이 뭐하는 회산데 수백만 요청씩이나?

겨우 6개월 밖에 안된 회사지만...

거대한 사이트들의 로그를 수집하고 있는 B2B 기업

한겨레
HANI.CO.KR

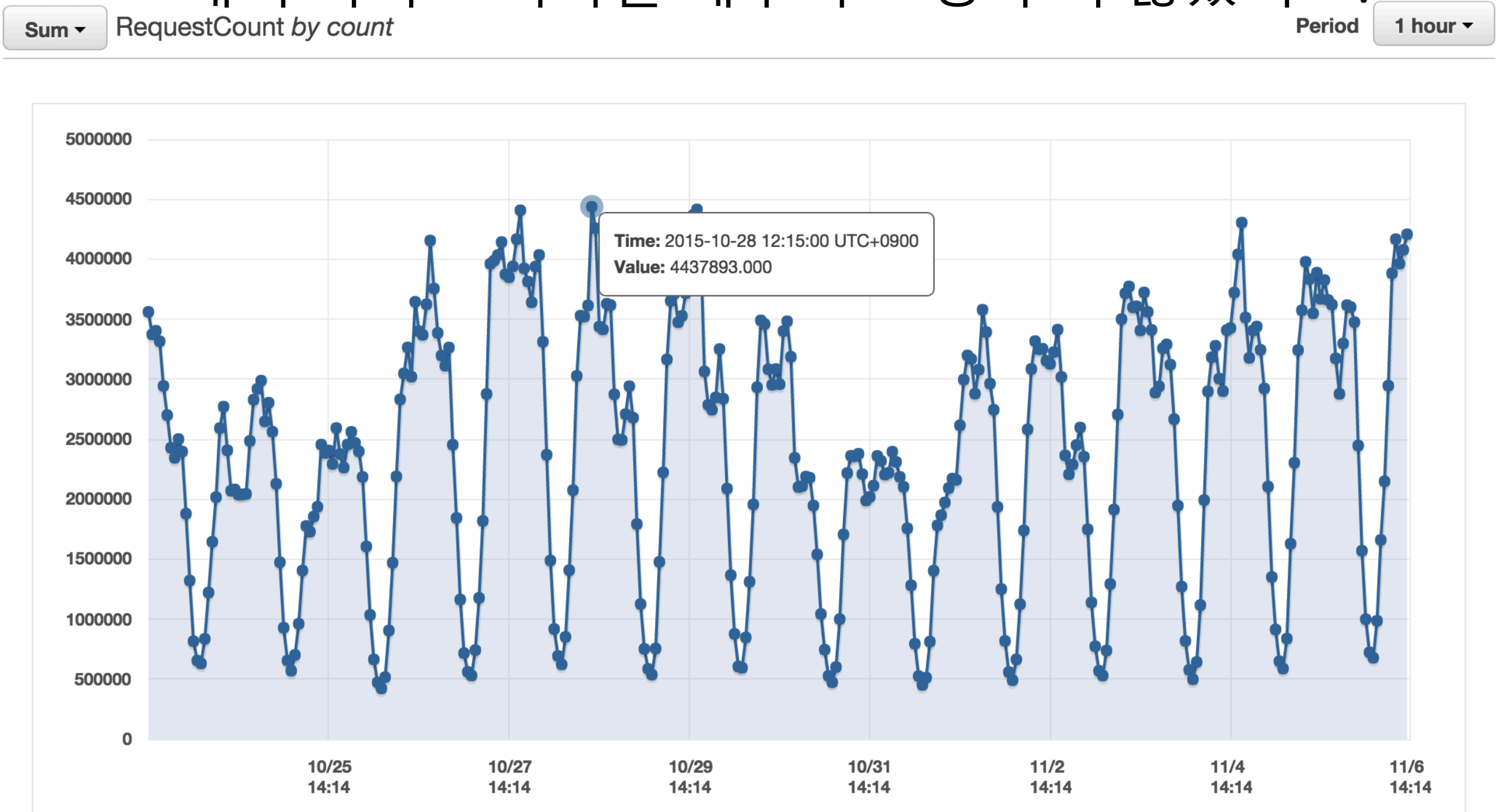
STYLE NANDA
www.stylenanda.com
since 2004

Dable

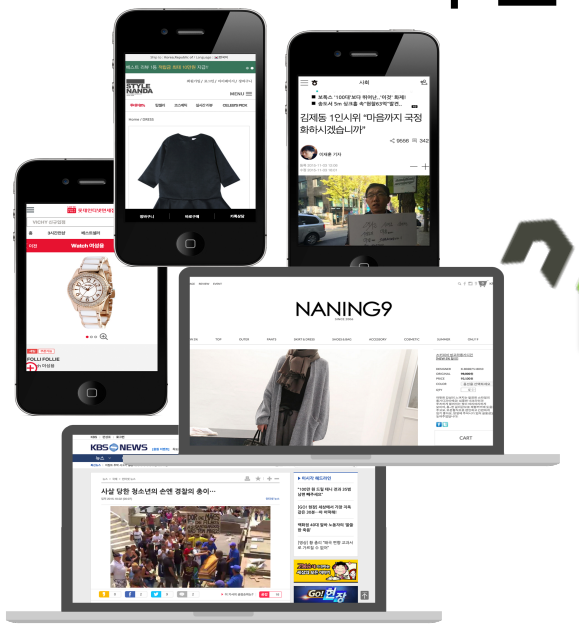
KBS

LOTTE DUTY FREE

그래서 서비스 시작할 때부터 요청 수가 많았어요.

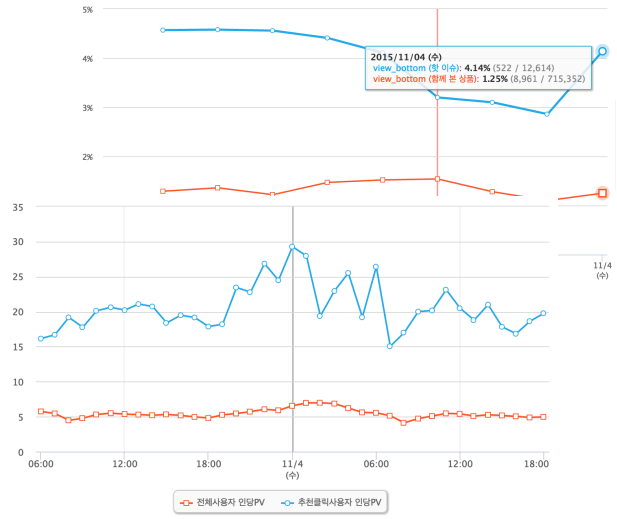


이런 형태로 서비스 하고 있습니다.



로그 수집

빅데이터 처리



통계 시각화

추천 성과 계산

추천 제공

이런 형태로 서비스 하고 있습니다.

- Node.js Packages in Dable
  - Web Servers
    - **API**: 로그 수집, 추천 제공(위젯 렌더링)
      Recommendation(s): 추천 제공(알고리즘 별)
    - Dashboard: 통계 시각화 (고객사들에게 성과 제공)
    - HQ: 내부용 관리자
  - scripts
    - admin scripts (각종 관리용 스크립트 모음)
    - meta crawler
      (API, Recommendation 등에서 모듈 형태로 포함)

?

2. 그래서 운영은 어떻게?

기승전…

Good

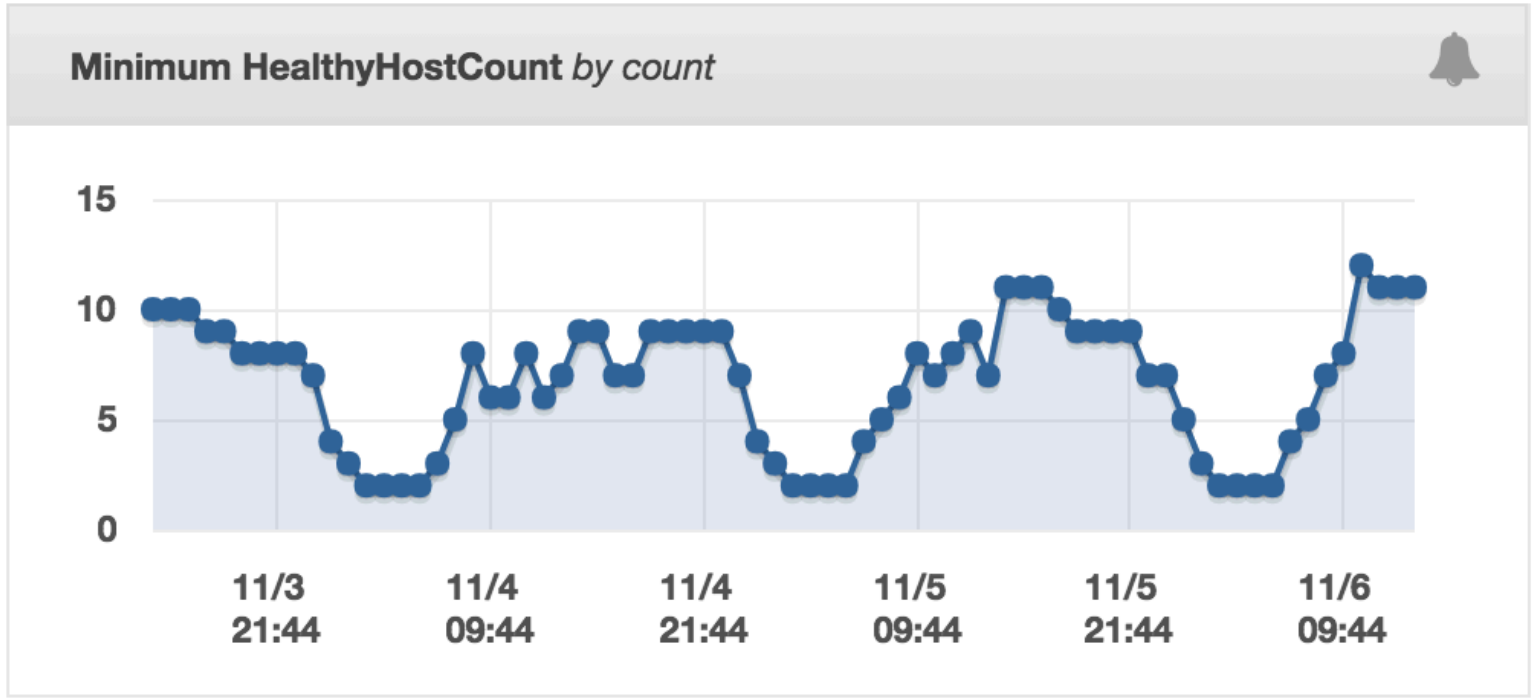
- Auto Scailing, Load Balancing 등 알아서 처리
- 서버 모니터링도 쉽게 가능하도록 지원된다.

Bad

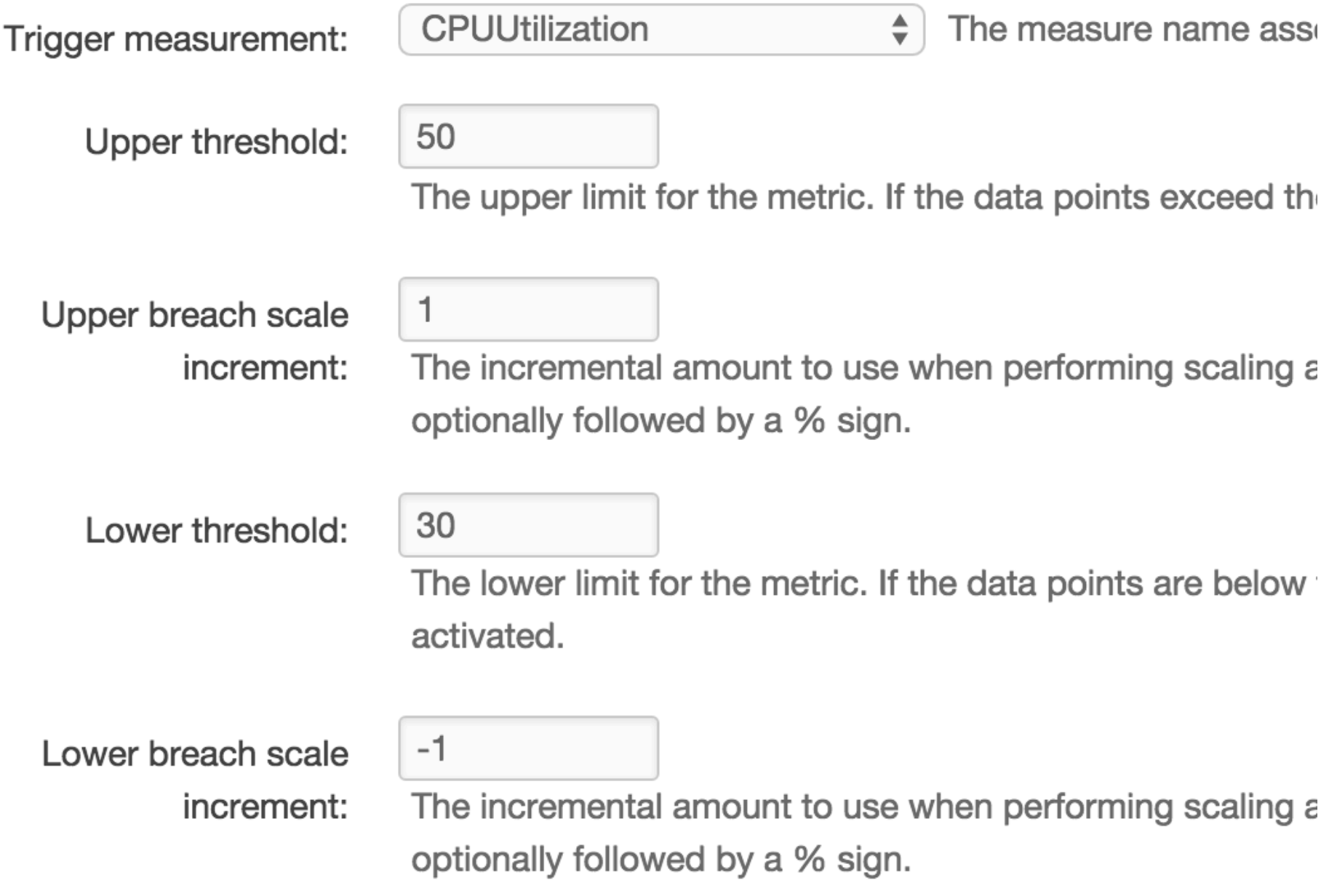
- 플랫폼 대응이 쪼금 느리긴 하다.
  - 1주일 전에 node v4.2.1 버전업
  - 그 전까진 v0.12.6을 쓰다가;;
- 서버 환경 자유도가 좀 떨어진다.

Google App Engine, Heroku 같은 Cloud Paas Platform

Instance 수는 요청 수에 비례하여 새벽에 2대, 피크 때 10대 남짓



CPU 사용률에 따라 Instance를 넣고 빼고 합니다.



v0.12.6 -> v4.2.1

- 메모리 사용량이 반으로 줄었다!
  (3.75G 중 30% -> 15%)
- CPU 사용량, 응답 속도는 체감상 크게 달라지지는 않았다.
- ~~근자감이 생겼다.~~

그 밖에 대량 트래픽 관리를 위해...

- **캐시로서의 redis 서버 활용**
  - 시간당 최대 백만건의 로그를 저장해야 하는 API 서버
  - 현재 MySQL(legacy), AWS kinesis에 동시에 저장
  - 매번 INSERT 하면 부담
  - 그래서 각 서버 Instance에 local redis 서버를 띄워놓고
  - local redis로 로그를 Queue처럼 쌓은 후
  - 20건 정도씩 Dequeue 해다가 두 군데에 몰아넣는 구조

그 밖에 대량 트래픽 관리를 위해...

- **로그 저장 HTTP 요청은 최대한 빠르게 응답**
  - 로그 저장 요청은 특성 상 응답 시 성공/실패 여부 필요없음
  - 일단 HTTP 요청은 먼저 응답하고 (200 ok)
  - 저장에 필요한 일련의 동작들은 async하게 처리
  - HTTP 요청의 동작과 무관한 응답을 하는 경우라면
    이 방법으로 응답 시간을 줄일 수 있을 것 같다.

?

3. 그래서 개발은 어떻게?

node.js 웹 서버 디렉토리 구조: express(v4.13.x) 기반

```
app.js
route.js
controller/
lib/
front/
test/
views/
public/
```

app.js: 서버 실행 파일. cluster, web server 실행

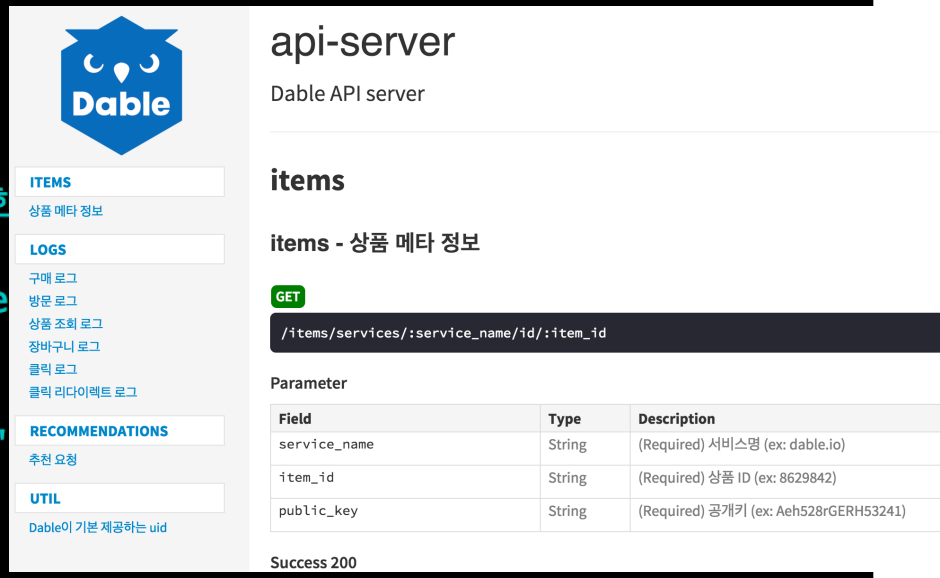
app.js
route.js
controller/
lib/
front/
test/
views/
public/

Cluster count == CPU count

route.js - 라우팅 맵, Documentation (with apidoc)

app.js
route.js
controller/
lib/
front/
test/
views/
public/

```coffee
route = (app) ->
  app.get "/", (req, res) -> res.send "DABLE API SERVER WORKS"

  # log APIs
  ###
  @api {get, post} /logs/services/:service_name/users/:uid/visit 방문 로그
  @apiName visit_log
  @apiGroup logs
  @apiUse common_log_api
  ###
  app.get "/logs/services/:service_name/users/:uid/visit", c.logs.visit
  ###
  @api {get} /logs/services/:service_name/users/:uid/view 상품 조회 로그
  @apiName view_log
  @apiGroup logs
  @apiUse common_log_api

  @apiParam {Object[]} items (Required) 조회한
  @...
  @apiParamExample {json} items param example
      {
        "items": [
          {"id": "ITEM_ID", "c1": "CATEGORY1"
        ]
      }
  ###
  app.get "/logs/services/:service_name/users/:uid/view", c.logs.view
```



express middleware는 controller로, 복잡한 로직, 공통 로직은 lib으로

```
app.js
route.js
controller/
lib/
front/
test/
views/
public/
```

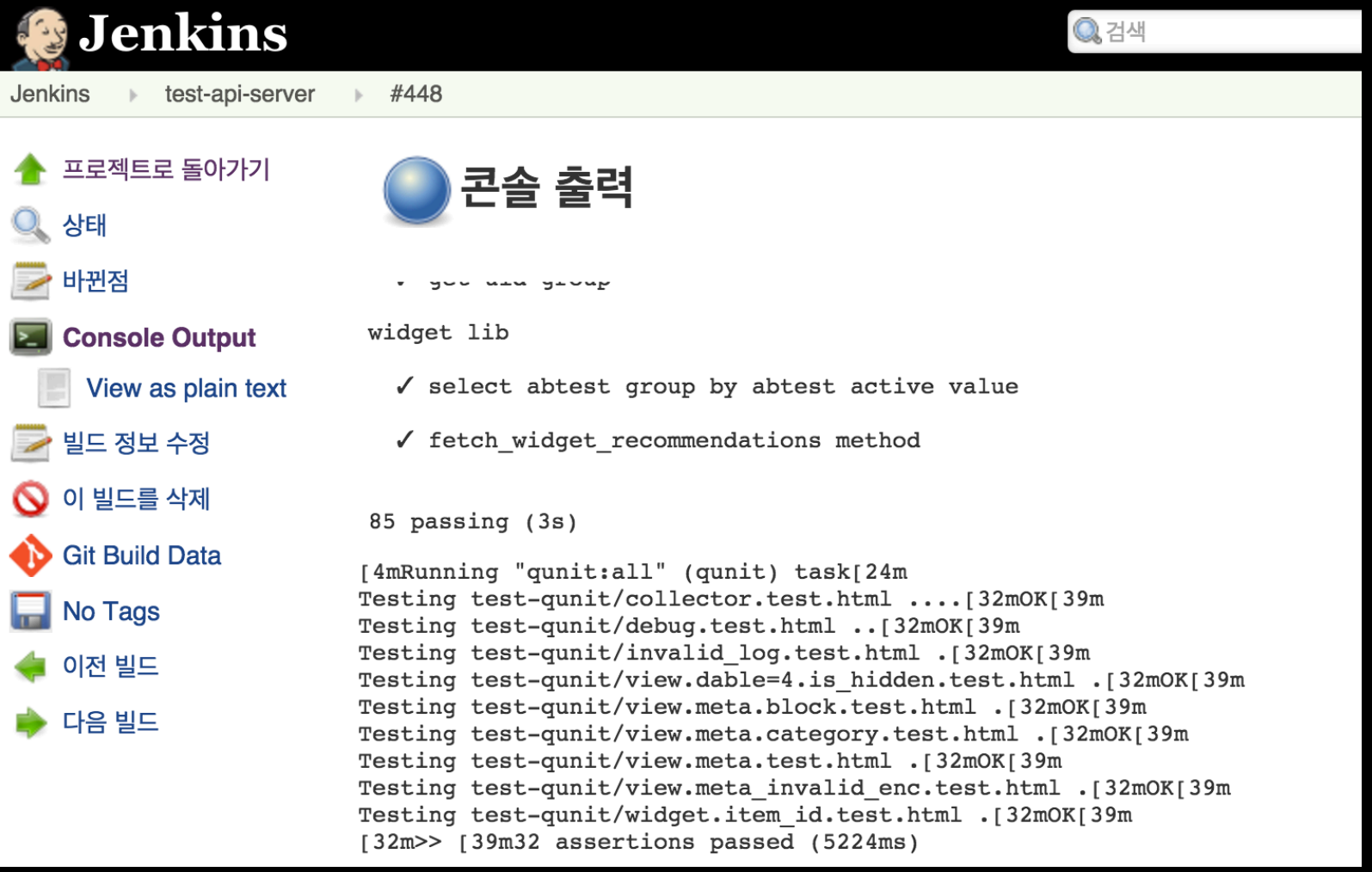
Front end 코드들. public/dist/로 concat & uglify

```
app.js
route.js
controller/
lib/
front/
  style/
  script/
test/
views/
public/
```

- CSS: stylus, JS: coffeescript
- 모듈별로 쪼갠 여러개의 파일로 개발
- JS의 경우 Browserify (CommonJS Pattern) 사용하여 서버 로직을 함께 쓰기도...
- 배포 시 concat & uglify 하여 하나의 파일로
- 기타
  - 개발시에는 grunt watch 이용
  - source map을 활용하여 디버깅

Unit tests. mocha+should+rewire(for Server), qunit/karma(for Front-end

app.js
route.js
controller/
lib/
front/
test/
test-front/
views/
public/



Unit tests. rewire 사용 예제

```
app.js
package.json
routes.js
controller/
lib/
front/
test/
test-front/
views/
public/
```

```
var fsMock = {
    readFile: function (path, encoding, cb) {
        expect(path).to.equal("/somewhere/on/the/disk");
        cb(null, "Success!");
    }
};
myModule.__set__("fs", fsMock);

myModule.readSomethingFromFileSystem(function (err, data) {
    console.log(data); // = Success!
});
```

기타 즐겨쓰는 라이브러리

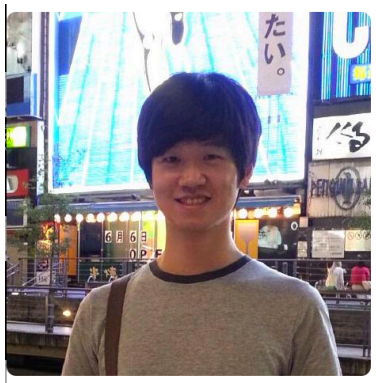
- moment: 날짜, 시간 다루기
- async -> Q: Promise!
- lodash(underscore): Super Utility
- sanitizer: 문자열에서 안전하지 않은 HTML 태그들을 걸러주는 라이브러리.

기타 즐겨쓰는 라이브러리

- lz-string: Front/Server 양쪽에서 사용 가능한 문자열 압축 라이브러리. Front에서 만든 URL의 긴 파라미터를 줄이기 위해 사용. (2000자 -> 800자)

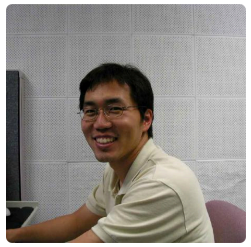
```
// compress (보내는 쪽)
reco_list = [{id:"121", method:'1.1.1'}, {id:"A42", method:'1.2.1'}, ... ];
reco_list_lz = lz.compressToEncodedURIComponent(reco_list);

url = "http://api.dable.io/logs/blahblah?reco_list_lz=NobwRAlgJmBcYCYAsBWAHAZgSsAaMAtgKYAuAF
gPYzwCMADAHQ1NgC%2Bu40ciSAbCnRoB2PIVKVqYekxbtOk5Cl4BOJElHFyVbtOY02HSAr5ohQuhvHbajPQfndkvGs-3
5NEnbdmGu8ZEIY6MqWWpK6Pg7%2BSJhICG5iYV4y%2BnJGjjE0CAhooZ42Kfbp0YFoKDjuVuHeqb7GKOa8vHnWUjVFfj
xoNMqBLdWFaZ0BKJn9yXZDxvzlFYn5bYN1GbwYvH2VSQWTALpAA";

// decompress (받는 쪽)
reco_list = JSON.parse(lz.decompressFromEncodedURIComponent(reco_list_lz));
```

협업?



Module A (Node.js)

HTTP



Module B (Node.js)

npm package

Module C (Node.js)

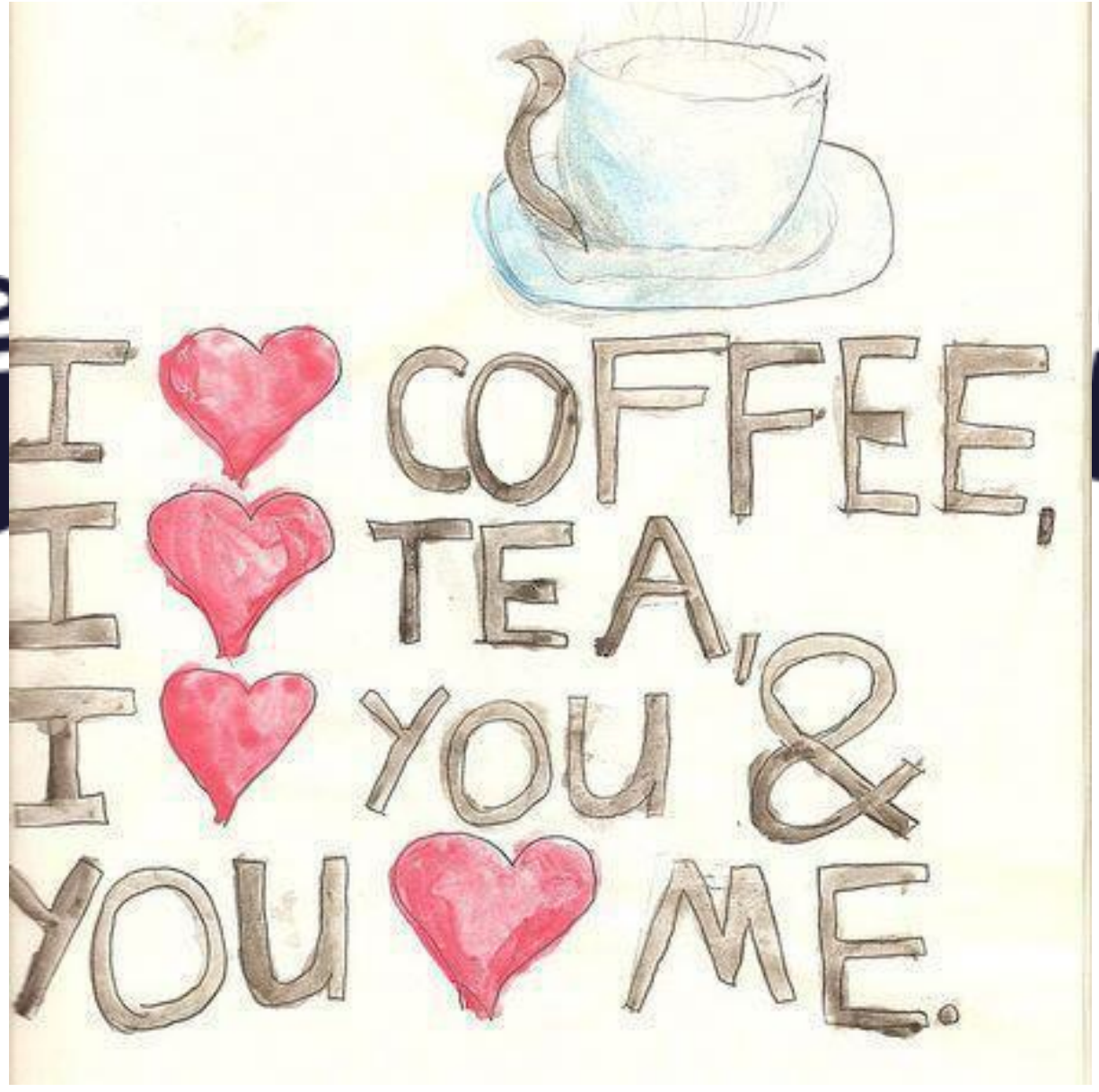
HTTP

Module D (Python)

- 한 repo를 같이 작성하는 경우는 드묾
- 각자 project를 생성하여
  - npm package 형태로 가져다 쓰거나
  - HTTP로 통신한다.
  - HTTP 통신
    - 성능은 별로지만
    - B 수정 시 A 재배포가 필요없어 관리에 좋다.

CoffeeScript를 즐겨쓰는데요...

왜냐하면 커피를 좋아해서...



CoffeeScript

- 들여쓰기, 띄어쓰기, brace 어디다 열지, jslint/jshint... 피곤
- 늙으니 타이핑 속도가 느려짐. IDE 취향 아님 -_-;;
  - 물론 이건 es6에 익숙해지는 해결책도...
  - 근데 front end JavaScript 코딩도 같은 언어로 하다보니 es6 쓰면 context swiching이 잘 안됨 ㅠ

```
var that = this;
function callback(){
  that.something();
}
```

```
# in coffeescript
callback = => @something()
```

CoffeeScript

> less typing, bad readability
> Why CoffeeScript Isn't the Answer
> Replace CoffeeScript with ES6

네... CoffeeScript를 추천할 심산은 아니에요.
ES6, Babeljs. 조금 더 나은 해답들이 많죠.

CoffeeScript

그런데 IE7, IE8를 버릴 수 없는 상황이라면? (...)

거기다 저같은 불편함을 느끼신다면 추천!

4. 몇가지 삽질기

삽질 1: 자기 서버에 HTTP로 요청을 날리는 로직을 만들었다가...

일단 설명 전에... 이것들, 익숙하시죠?

삽질 1: 자기 서버에 HTTP로 요청을 날리는 로직을 만들었다가...

그럼 request를 날린 서버가 502 Bad Gateway라면?


```
request.get("http://some.where/in/the/world",
    function (err, response, body) {
  // err가 있을까 없을까?
});
```

없더라고요.....

삽질 1: 자기 서버에 HTTP로 요청을 날리는 로직을 만들었다가...

```
var fetch_widget_data = function (options) {
  request({
    url: 'http://api.dable.io/recommendations/playnode.io/...',
    form: options
  }, function (err, resp, body) {
    if (err) return res.json({result: []});
    // BLAHBLAH
    return res.json(
      JSON.parse(body)
    );
  });
};
```

1. 무언가 다른 원인으로 서버 뻗음
2. 서버 리스타트 (리스타트 도중
   일부 nginx의 502 Bad Gateway 에러)
3. 여기서 날린 Request가
   그 에러에 해당
4. err는 null이고 body는 502 Bad Gateway에 해당하는
   HTML 코드라 JSON.parse 시도 중 서버 뻗음
5. 무한루프

삽질 1: 자기 서버에 HTTP로 요청을 날리는 로직을 만들었다가...

```
var fetch_widget_data = function (options) {
  request({
    url: 'http://api.dable.io/recommendations/playnode.io/...',
    form: options
  }, function (err, resp, body) {
    if (err) return res.json({result: []});
    // BLAHBLAH
    try { return res.json(JSON.parse(body)); }
    catch (e) { return res.json({result: []}); }
  });
};
```

JSON.parse 로직을 try, catch 처리.
후에 HTTP 요청을 function call 형태로 변경. (진작 이랬어야...)

삽질 2: 평화로운 휴일. 서버가 뻗었다. 뻗은 서버는 살아나지 않았다.

2015-08-15 16:20:52 UTC+0900	INFO	Adding instance 'i-be95cc4b' to your environment.
2015-08-15 16:20:49 UTC+0900	INFO	Added EC2 instance 'i-be95cc4b' to Auto Scaling Group 'awseb-e-efbczggwxa-stack-AWSEBAutoScalingGroup-GTE9RB9AZ49Q'.
2015-08-15 16:18:11 UTC+0900	INFO	Removed instance 'i-71396784' from your environment. (Reason: Instance is in 'shutting-dow
2015-08-15 16:09:58 UTC+0900	INFO	restartAppServer is starting.
2015-08-15 10:34:11 UTC+0900	WARN	Environment health has transitioned from YELLOW to RED
2015-08-15 10:32:11 UTC+0900	WARN	Environment health has transitioned from GREEN to YELLOW
2015-08-15 10:32:11 UTC+0900	WARN	Elastic Load Balancer awseb-e-e-AWSEBLoa-1NXX5H3MA73NK has zero healthy instances.
2015-08-14 15:29:25 UTC+0900	INFO	Removed instance 'i-a64f1053' from your environment. (Reason: Instance is in 'terminated' st
2015-08-14 15:21:55 UTC+0900	INFO	Adding instance 'i-71396784' to your environment.

삽질 2: 평화로운 휴일. 서버가 뻗었다. 뻗은 서버는 살아나지 않았다.

원인은 Memory Leak

서버 구동 후 Memory는 야금야금 상승

휴일의 한가운데에서 뻗어버렸다.

삽질 2: 평화로운 휴일. 서버가 뻗었다. 뻗은 서버는 살아나지 않았다.

메모리가 왜 올라가는지 몰라서...

서버 로그도 보고

코드도 거듭 정독하고

디버그 코드도 잔뜩 심어보고

그래도 몰라서...

삽질 2: 평화로운 휴일. 서버가 뻗었다. 뻗은 서버는 살아나지 않았다.

Heap Snapshot을 찍어봤다. (with https://github.com/bnoordhuis/node-heapdump)

크롬 개발자도구의 Profiles 탭에서 두 snapshot을 비교해봤다.

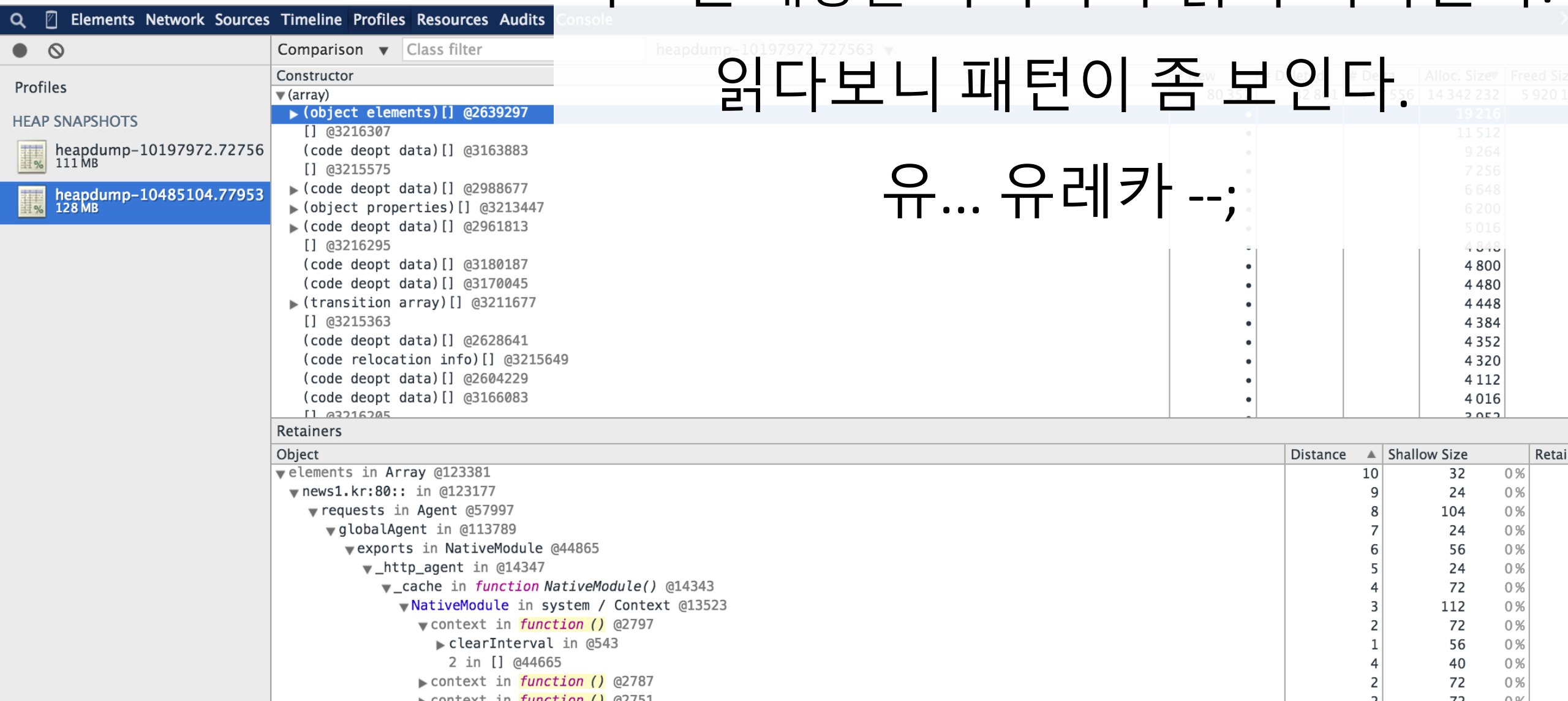
이제 어쩌지 싶었다.

삽질 2: 평화로운 휴일. 서버가 뻗었다. 뻗은 서버는 살아나지 않았다.

비교한 내용을 하나하나 읽기 시작한다.

읽다보니 패턴이 좀 보인다.

유... 유레카 --;



삽질 2: 평화로운 휴일. 서버가 뻗었다. 뻗은 서버는 살아나지 않았다.

아래가 문제의 코드. 뭐가 문제였을까요?

```
var crawl = function (url, callback) {
  request({
    url: url,
    headers: {'User-Agent': 'Mozilla/5.0 (Macintosh; ...'}
  }, function (err, res, body) {
    if (err) return callback(err);

    var result = body;
    // !@!%#$AT#T@#%
    callback(null, result);
  });
};

// examples (callback은 생략)
crawl("http://news.kbs.co.kr/news/view.do?ncd=3107610");
crawl("http://news1.kr/articles/?2377986");
```

삽질 2: 평화로운 휴일. 서버가 뻗었다. 뻗은 서버는 살아나지 않았다.

```
var crawl = function (url, callback) {
  request({
    url: url,
    headers: {'User-Agent': 'Mozilla/5.0 (Macintosh; ...'}
  }, function (err, res, body) {
    if (err) return callback(err);

    var result = body;
    // !@!%#$AT#T@#%
    callback(null, result);
  });
};

// examples (callback은 생략)
crawl("http://news.kbs.co.kr/news/view.do?ncd=3107610");
crawl("http://news1.kr/articles/?2377986");
```

원인은 timeout 없는 HTTP request
고객사 중 한곳이 API Server IP를 Block
Block 방식이 응답을 주지 않는 식
응답이 없으니 linux 기본 timeout(2시간) 적용
요청만큼의 request가 메모리에 쌓여갔던 것

삽질 2: 평화로운 휴일. 서버가 뻗었다. 뻗은 서버는 살아나지 않았다.

```
var crawl = function (url, callback) {
  request({
    url: url,
    timeout: 5000,
    headers: {'User-Agent': 'Mozilla/5.0 (Macintosh; ...'}
  }, function (err, res, body) {
    if (err) return callback(err);

    var result = body;
    // !@!%#$AT#T@#%
    callback(null, result);
  });
};

// examples (callback은 생략)
crawl("http://news.kbs.co.kr/news/view.do?ncd=3107610");
crawl("http://news1.kr/articles/?2377986");
```

timeout을 추가하여 문제 해결

마치며... SPECIAL THANKS TO NODE.JS

저에게 node.js란...

서버 개발 지식이 부족한 저같은 프론트 엔드 개발자도
큰 규모의 서비스를 운영할 수 있게 해준 존재입니다.

같이 삽질하실 분... 찾습니다. +_+

WANTED: node.js 개발자 (빅데이터 흥미있으신 분 대환영!)

dable@dable.io

THANK YOU