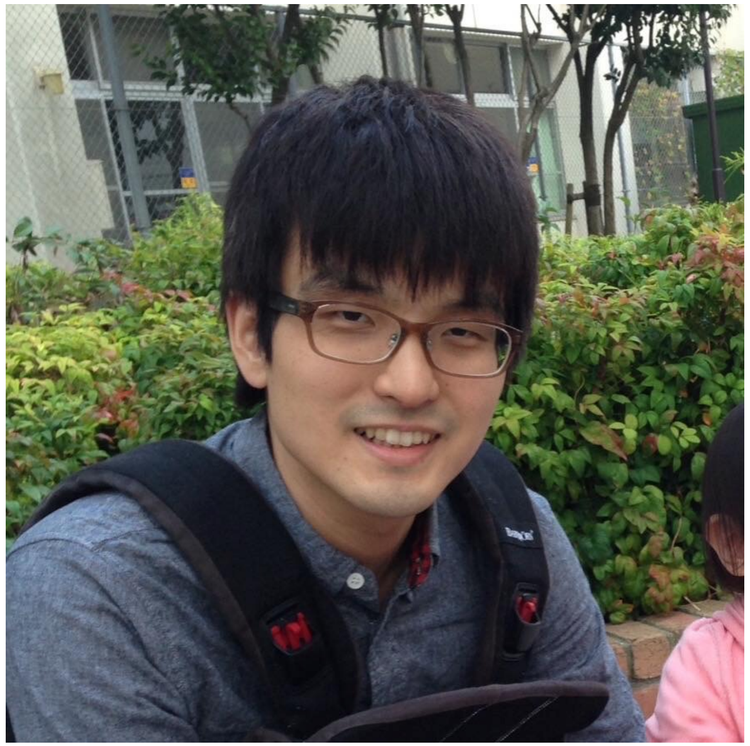
Ruby Way OpenStack

ペパボにおけるOpenStack基盤開発でのRuby活用について

GMO Pepabo, Inc.
Uchio Kondo

2015/11/12 RubyWorld Conference 2015

me



GMOペパボ

盤

近藤うちお

- GMOペパボ所属
  - 技術基盤チーム
  - 福岡支社勤務
- Fukuoka.rb
- RailsGirls Fukuoka #1 総合雑用/コーチまとめ

出自

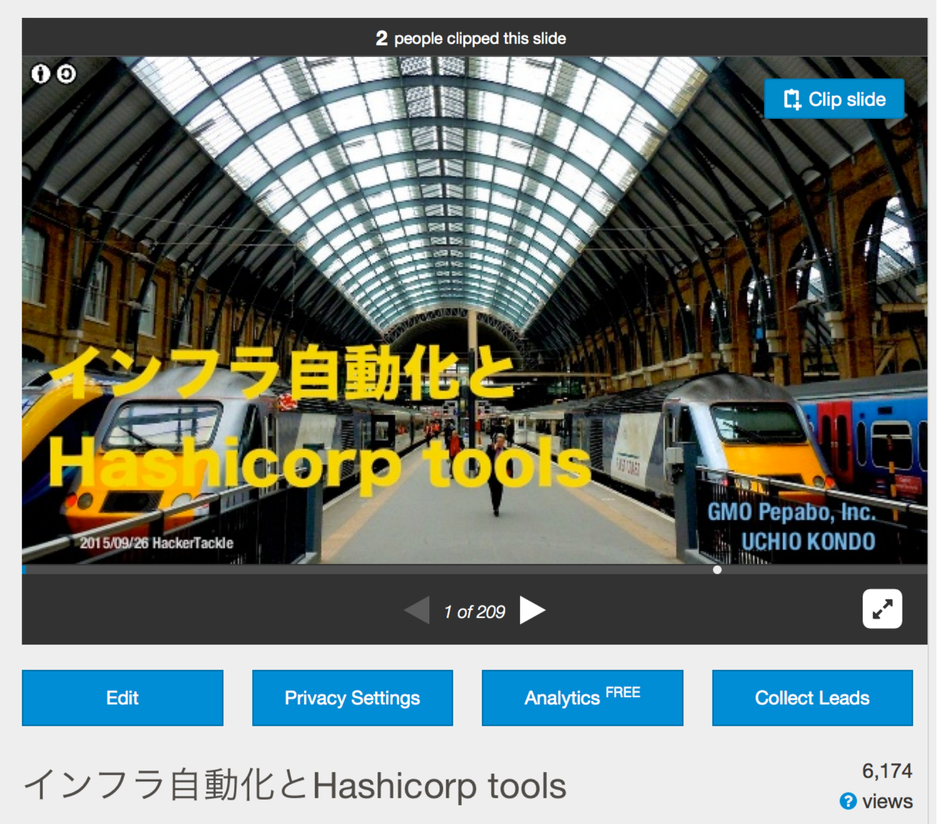
- 東三河の人
- 大学入学とともに東京へ
- 2013年から福岡
- 名古屋じゃないもんだいね

島根とのご縁

- 母方の実家は旧邇摩郡仁摩町⏳
  - 松江は学生以来です
  - 家によく隠岐島のウニの瓶詰がありました

興味

- Ruby / Golangを少々
- Fluentd / InfluxDB
- Puppet / Docker
- Hashicorp tools
- OpenStack

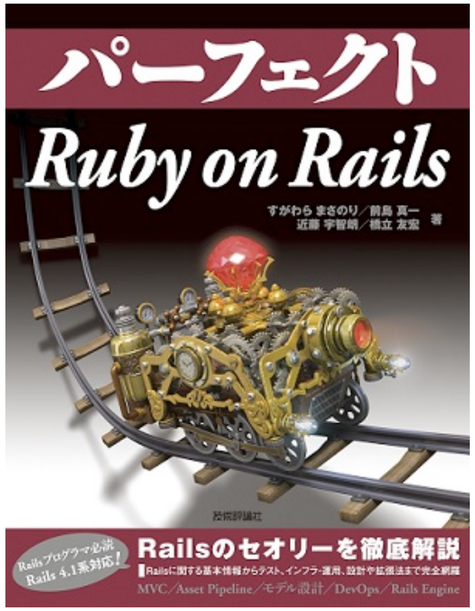


http://www.slideshare.net/udzura/hashicorp-tools

7 years old Rubyist

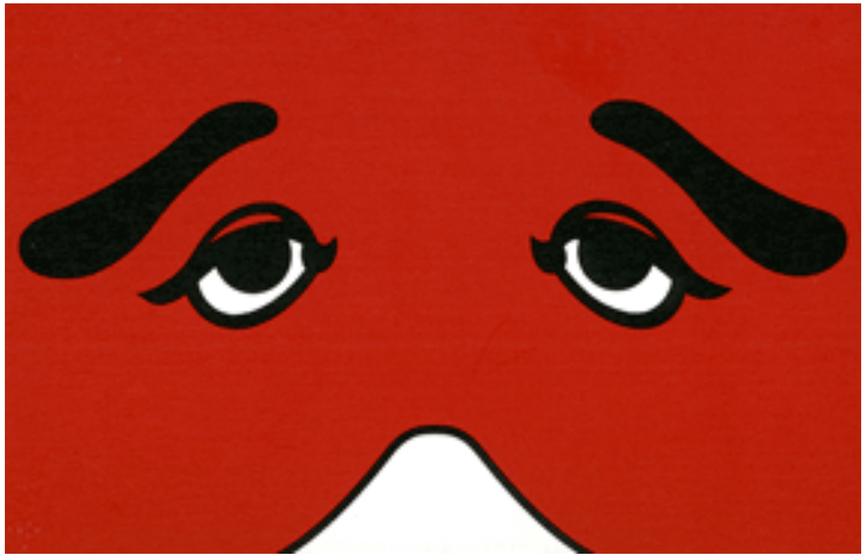
- Rails 2.1.0 ごろからのルビースト(2008 ~)
- Rubyをこじらせて著作あり
  - Web+DB Press Ruby連載(2012 ~ 2014)
  - パーフェクトRuby (2013)
  - パーフェクトRails (2014)


WEB+DB PRESS
新バージョン2.1登場!
VOL 79
Ruby最深動向
主要コミッタが明かす新機能と高速化の舞台裏
これからはじめる
iOS7開発
緊急対応、計画停止、メンテフリー化
実践メンテナンス
Dockerによる軽量な仮想環境
Parseでらくらくスマートフォン開発
AngularJS／mruby／Bitcoin


パーフェクト
Ruby on Rails
すがわら まさのり／前島 真一
近藤 宇智朗／横立 友宏 著
技術評論社
Railsプログラマ必読
Rails 4.1系対応!
Railsのセオリーを徹底解説
Railsに関する基本情報からテスト、インフラ・運用、設計や拡張法まで完全網羅
MVC／Asset Pipeline／モデル設計／DevOps／Rails Engine


パーフェクト
Ruby
Rubyサポーターズ 著
技術評論社
Rubyist必読の1冊
Ruby 2.0対応!
Rubyのセオリーを徹底解説
基本からgemパッケージの作成方法やWebアプリケーション開発まで、最新の技術を完全網羅
RubyGems／Bundler／Rack／Sinatra／YARD

Fukuoka.rb



福岡

店舗検索：福岡県　福岡市中央区で検索した結果（13 件）

福岡県

福岡市中央区

天神地下街店

福岡県 福岡市中央区 天神2丁目地下０号 天神地下街

詳細

メディアモール天神店

福岡県 福岡市中央区 天神1-10-13 天神MMＴビル 1F

福岡パルコ店

福岡県 福岡市中央区 天神2-11-1 福岡パルコ新館

IWATAYA本店本館店

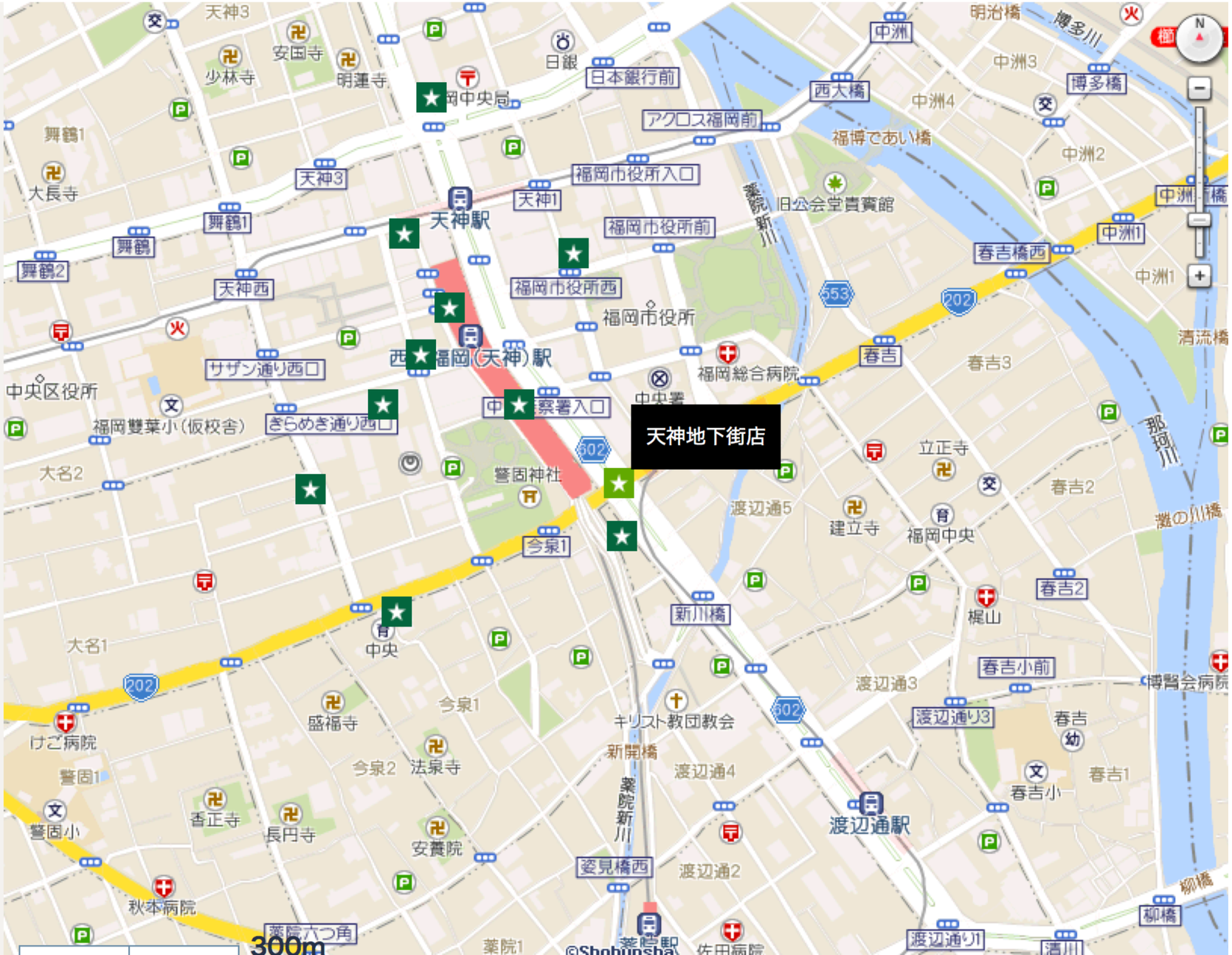
福岡県 福岡市中央区 天神2-5-35 岩田屋本店本館１Ｆ

ミーナ天神店

福岡県 福岡市中央区 天神4-3-8 ミーナ天神

天神VIORO店

福岡県 福岡市中央区 天神2-10-3 天神VIORO



店舗検索：福岡県　福岡市中央区で検索した結果（13 件）

福岡県

福岡市中央区

天神地下街店

福岡県 福岡市中央区 天神2丁目地下０号 天神地下街

詳細

メディアモール天神店

福岡県 福岡市中央区 天神1-10-13 天神MMＴビル 1F

福岡パルコ店

福岡県 福岡市中央区 天神2-11-1 福岡パルコ新館

IWATAYA本店本館店

福岡県 福岡市中央区 天神2-5-35 岩田屋本店本館１Ｆ

ミーナ天神店

福岡県 福岡市中央区 天神4-3-8 ミーナ天神

天神VIORO店

福岡県 福岡市中央区 天神2-10-3 天神VIORO

管理者パネル
コミュニティ管理　イベント編集　イベント管理

Fukuoka.rb #33 #fukuokarb

2015-04-16（木）19:00 - 21:00

Google カレンダーに追加　#fukuokarb

締め切り	参加費
募集終了	無料

イベントのお申し込みは終了しました。

イベントのお申し込みは完了しています。

お申し込みの確認

参加者 4

- jue
- 近永 智之
- MATSUMOTO, Ryos...
- 近藤 うちお

Fukuoka.rb メンバー

- @nagachika (Rubyコミッタ, 2.2リリースマネージャ)
- @matsumotory (mod&ngx_mruby)
- @k1Low (awspec)
- @morygonzalez (清貧会所属)
- @h_demon (Ember.jsアイドル)
- @udzura (Puppetおじさん)
- などなど…

RailsGirls Fukuoka #1

http://blog.railsgirls.jp/post/128691554417/rails-girls-fukuoka-1

ペパボ

all of services by GMOペパボ

インフラでは

fluentd

VAGRANT

Chef

Capistrano

OpenStack


openstack™
CLOUD SOFTWARE

python™

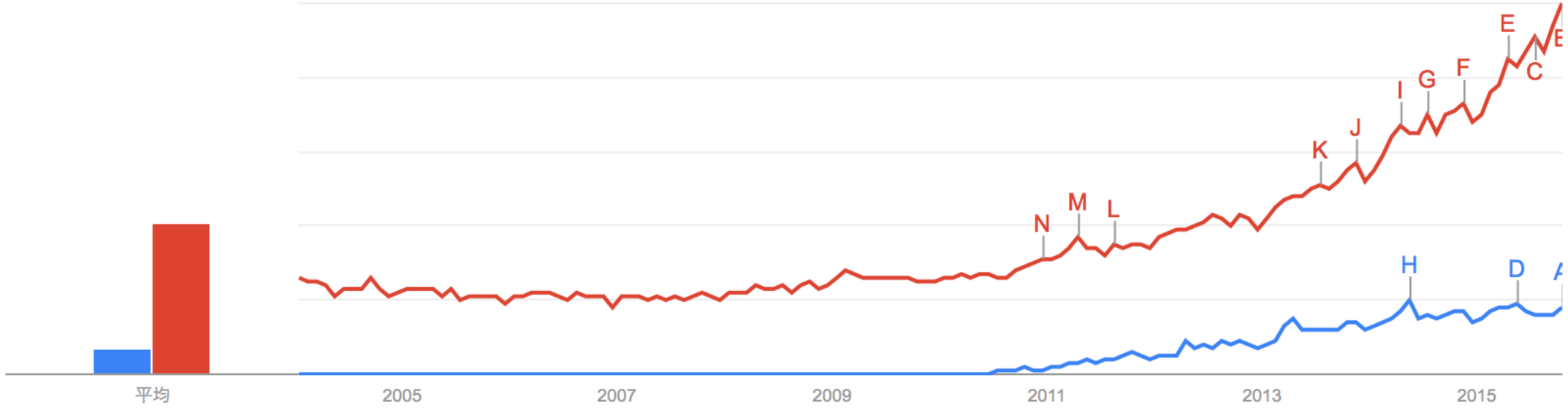
openstack
検索キーワード

aws
検索キーワード

+キーワードを追加

人気度の動向

ニュースのヘッドライン　予測



google trend™

今日の発表の
ターゲット

ターゲットのまえに

- OpenStack利用者といっても
- OpenStack自体を運用提供するひと
  - Rackspace社、GMOインターネットなど
- OpenStack基盤を開発者として利用するひと
  - mixi、drecom、CyberAgentなど

ペパボの立ち位置

- 基盤提供側として:
  - 社内のインフラチームで運用保守している
- 利用者として:
  - アプリケーションエンジニアがサーバ基盤として使っていく
  - 社内の基盤/インフラチームも関わる
- 「自社で」「自分たちのために」運用、利用するひととしてお話しします。
- ただ、若干、「利用者」寄りになると思います

なのでターゲット

- OpenStack利用者として、フツーにクラウドらしく利用したいひと
  - →参考になる部分は多いかもしれません
- OpenStack基盤運用するひと
  - →利用者はこうなんだな〜という感じでお聞きください

OpenStack x Pepabo

NyAh

ポチポチするために ニャーニャーしよう

> http://www.slideshare.net/ume3_/pb-tc01-bob001

ペパボにおけるOpenStackHacks!

> https://speakerdeck.com/pyama86/pepaboniokeruopenstackhacks

cf. GMOインターネットにおける OpenStack Swiftのサービス化

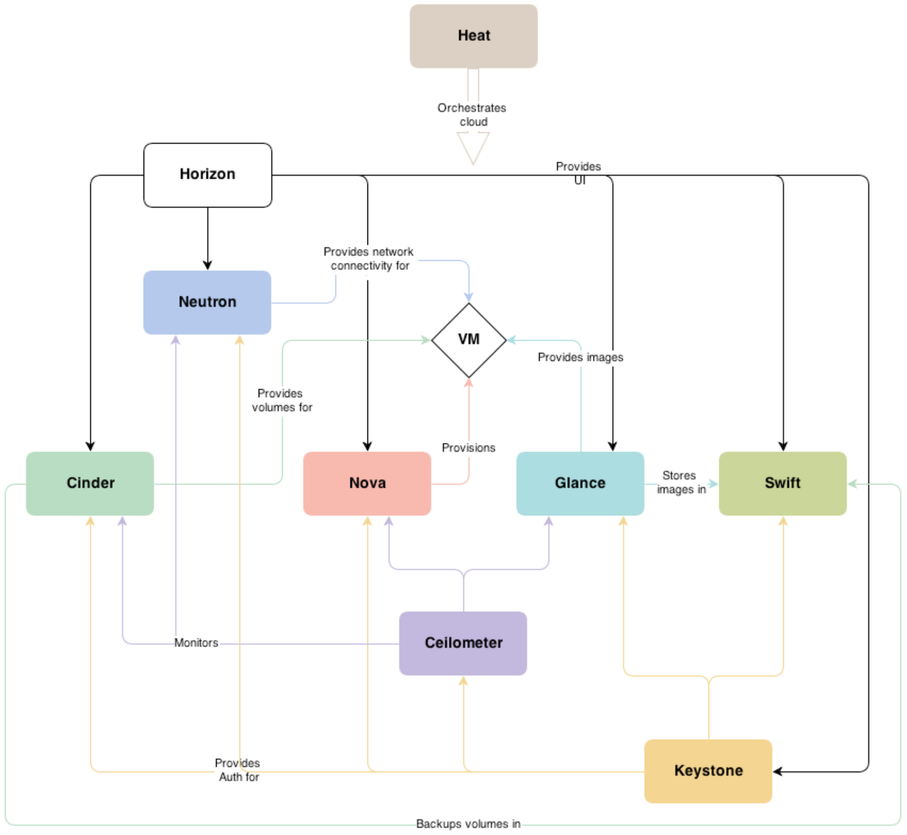
> http://www.slideshare.net/VirtualTech-JP/conoha-44760338

弊社構成詳細

- Havanaを利用
  - グループ内で運用実績があり、アドバイス等聞ける
  - とはいえ、2年半以上前のリリースということでアップグレードは検討に入っている
- ネットワークに癖
  - 主にパフォーマンス的な観点から、SDNにしていない
  - 組み込みのDHCP/L3 agentは利用不可

構成図

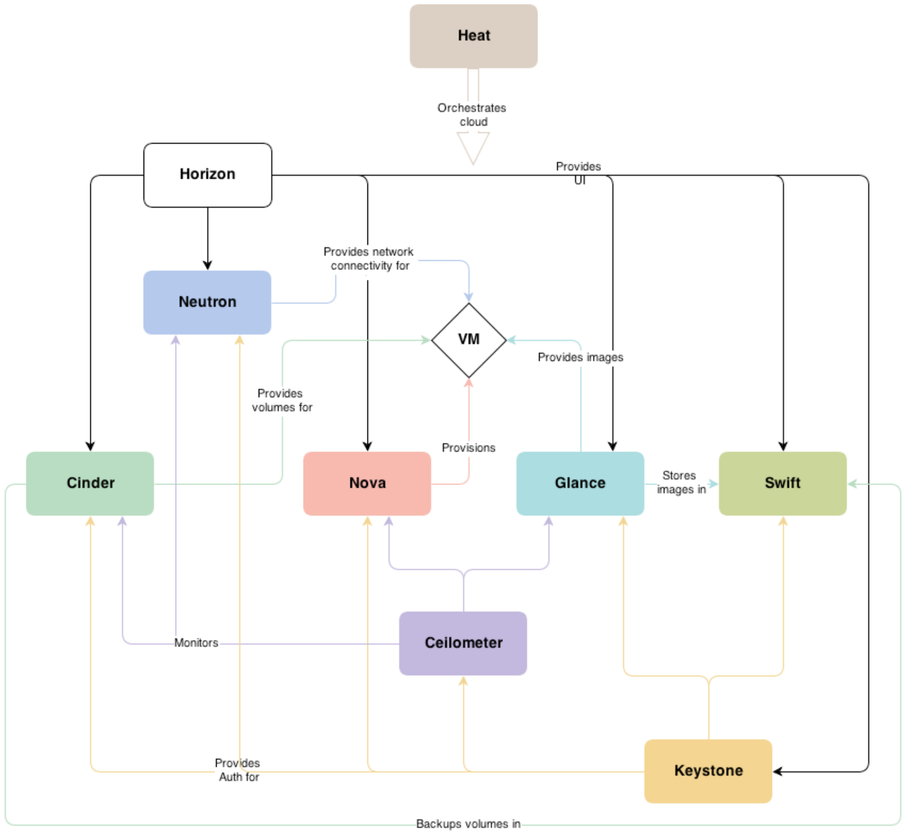
- Nova: コンピュート/インスタンス管理
- Neutron: ネットワーク
- Glance: イメージ管理
- Keystone: ユーザなどの管理
- (Cinder: ブロックストレージ)
- (Swift: オブジェクトストレージ)
- Horizon: 管理画面UI
- Ceilometer: リソース利用状況の管理
- (Heat: オーケストレーション層)



http://thinkit.co.jp/story/2014/06/10/4999

ざっくり言うと

> Nova:	EC2
> Neutron:	(VPC network)
> Glance:	AMI
> Keystone:	IAM
> (Cinder:)	EBS
> (Swift:)	S3
> Horizon:	Web console
> Ceilometer:	(Billing)
> (Heat:)	CloudFormation



運用に乗って
はや10ヶ月

実際どうなの？

ぶつかった問題

ユーザの管理をどうする…

- ユーザの管理は、セキュリティや監査にも関わる重要なポイント
- 共通ユーザ？ それはちょっと…
- 個々人でユーザを作るのが理想。でも、それをいい感じのフローにしたい

SecurityGroupの管理を…

- クラウドなので（？）
  自分でiptables書きたくない。宣言的なセキュリティグループに任せたい。
- これも割とカオスになりがち。構成管理、アドホックな対応の後始末、分割ルール…

ネットワークの癖

- いろいろあるが、大きいのが、DHCPは原則使わない運用
- あらかじめIPとMac addressがわかっていないとネットに繋がらない

userdataの活用

- AWS/GCEなどと同様、OpenStackでも cloud-init(*) は利用可能！
- userdataの概念が利用できる
- なので
  - cloud-config形式の設定を生成、そこにネットワークを直書き
  - 流し込む
- これでもいい。何も困らない。けど…

* https://cloudinit.readthedocs.org/en/latest/

userdataの手製が
クラウドっぽくない……

※個人の考えを含みます

問題、
先達を見てみる

Codenize.tools

codenize any services, and manage by DSL

About

Codenize.tools manege any services by DSL.

For example, It is possible to manage the Route53 by the following Ruby code:

```
hosted_zone "example.com." do
  rrset "example.com.", "A" do
    ttl 300
    resource_records(
      "127.0.0.1",
      "127.0.0.2"
    )
```

codenize.tools

Why?

“**Why?**

Idempotency. The service is set according to the definition of code.

It is easy to grasp the service configuration.

The service configuration can be managed by VCS.”

– http://codenize.tools/

“Codification is the belief that all processes should be written as code, stored, and versioned. Operations teams have historically relied on oral tradition to pass along the knowledge of how to build, upgrade and triage infrastructure. But **information was easily lost or hidden** from the people who needed it most. **Codification of critical knowledge promotes information sharing and prevents data loss**, as any changes to process are automatically stored and versioned.”

–Tao of Hashicorp (https://hashicorp.com/blog/tao-of-hashicorp.html)

Codinize/Codification

- 設定にべき等性が生まれる/自動化できる
- VCSで変更管理ができる
- GitHubなどで、設定の全容が容易に把握/検索できる

AWS の例

- miam.gem
  - IAMをRuby DSLで管理/変更できる
- piculet.gem
  - Security Group(AWS)をDSLで管理/変更できる

IAMfile example

```
require 'other/iamfile'

user "bob", :path => "/developer/" do
  login_profile :password_reset_required=>true

  groups(
    "Admin"

  policy "bob-policy" do
    {"Version"=>"2012-10-17",
     "Statement"=>
      [{"Action"=>
        ["s3:Get*",
         "s3:List*"],
       "Effect"=>"Allow",
       "Resource"=>"*"}]}
  end

  attached_managed_policies(
    # attached_managed_policy
  )
end

user "mary", :path => "/staff/" do
  # login_profile :password_reset_required=>true
```

tools are for AWS……

よろしい、ならば
書きましょう

kaname

kaname

- https://github.com/yaocloud/kaname
- @hsbt 作
- keystoneのユーザと、テナントごとの権限をyamlで記述可能になる

```yaml
antipop:
  email: "antipop@example.com"
  tenants:
    production: "cto"
hsbt:
  email: "hsbt@example.com"
  tenants:
    development: "admin"
    production: "member"
```

kakine

kakine

- https://github.com/yaocloud/kakine
- これも @hsbt 作……
- テナント別にSecurityGroupをyamlで記述
  それをソースに設定を適用できる

```yaml
app:
  rules:
    - direction: ingress
      protocol: tcp
      port: 443
      remote_ip: 0.0.0.0/0
    - direction: ingress
      protocol: tcp
      port: 80
      remote_ip: 0.0.0.0/0
  description: app rules
rails:
  rules:
    - direction: ingress
      protocol: tcp
      port: 3000
      remote_ip: 0.0.0.0/0
```

変更管理

- GitHub(Enterprise)のリポジトリで
- kanameはnyah管理全般プロジェクトを用意、そのリポジトリで
- kakineはサービスごとに

@udzura のユーザ権限追加 #145

Edit

Merged **udzura** merged 1 commit into `master` from `add-udzura2` on 14 Apr

Conversation 2 | Commits 1 | Files changed 1 | +2 −0

Showing 1 changed file with **2 additions** and **0 deletions**.

Unified | Split

2 keystone.yml

Show notes | View

```
@@ -17,6 +17,8 @@ udzura:
17  17    tenants:
18  18      admin: admin
19  19      kiban: admin
    20  +   devnyah: _member_
```

udzura added a note on 14 Apr — Owner

ここ、みんな `_member_` で not admin なんですがそれで日常生活に問題は無いと言うことですかね。

udzura added a note on 14 Apr — Owner

adminは逆に強すぎ、memberで良さそう。

Add line no

```
    21  +   mini  _member
20  22  lamanotrama:
21  23    email: kuroda@pepabo.com
22  24    tenants:
```

プルリクエストによる承認

Codinize/Codification

- 設定にべき等性が生まれる/自動化できる
- VCSで変更管理ができる
- GitHubなどで、設定の全容が容易に把握/検索できる

Codinize/Codification

- 設定にべき等性が生まれる/自動化
- VCSで変更管理ができる
- GitHubなどで、設定の全容が容易に把握/検索できる

コマンドで標準の手順になり、間違えない！

誰が誰を変えたかすぐわかる！

PRになるので検索もできる！

バリューだ！

インスタンス立ち上げ

立ち上げ、具体的流れ

- NeutronのPort（仮想NIC）を作成する
  - そこに、MACアドレスとIPがアサイン
- そのPortからMACアドレスとIP情報を取得
- その情報を元にifcfg-eth0などを作成
  - userdataのwrite_filesで流し込む
- Portを紐付けて、nova boot

立ち上げ、具体的流れ

- NeutronのPort（仮想NIC）を作成する
  - そこに、MACアドレスとIPがアサ
- そのPortからMACアドレスとIP情報を取得
- その情報を元にifcfg-eth0などを作成
  - userdataのwrite_filesで流し込む
- Portを紐付けて、nova boot

OpenStack APIがある

APIで取れる

プログラムでつくる

APIで起動できる

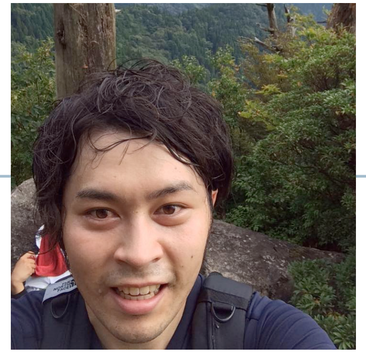
OpenStackにも
APIがある

APIがあるから、
プログラムで自動化できる

とりあえず

- Goで作ってみた
- nyah-cliという名前（非公開）

一方その頃



- 福岡の暑男（？） @pyama86 が
  - 自分のプロジェクトのためにツールを作っていた……
- それが pec

https://github.com/yaocloud/pec

README.md

Pec

build passing | code climate 3.5 | coverage 88%

OpenStack Vm create wrapper

Install

```
$ gem install pec
```

Usage

```
# set up
$ pec init
```

```
create - /Pec.yaml
create - /user_data/web_server.yaml.sample
```

pec

- 作成するインスタンスの情報をyamlで記述すると、その通りに作ってくれる
- ネットワークの情報を記述した場合、Port作成まで自動でやってくれる

Pec.yaml

```yaml
# merge of yaml
_default_: &def
  tenant: your_tenant
  image: centos-7.1_chef-12.3_puppet-3.7
  flavor: m1.small
  availability_zone: nova

# vm config
pyama-test001:
  <<: *def
  networks:
    eth0:
      bootproto: static
      allowed_address_pairs:
      - 10.1.1.5/24
      ip_address: 10.1.1.1/24
      gateway: 10.1.1.254
      dns1: 8.8.8.8
      dns2: 8.8.8.8
    eth1:
      bootproto: dhcp
  security_group:
  - default
  - ssh
  templates:
  - base.yaml
  - webserver.yaml
  user_data:
    hostname: pyama-test001
    fqdn: pyama-test001.ikemen.com
    repo_releasever: 7.1.1503
```

Demo

```
[udzura] ~/.ghq/
(*'-') <

[udzura] ~/.ghq/
(*'-') < pec up manage
make start manage001.
image is centos-6.6_chef-12.3_puppet-3.7
flavor is m1.medium
availability_zone is nova
port create start : eth0
assgin ip :
port create start : eth1
assgin ip : 10.51.97.15
keypair is sample001
load template manage.yaml
create success! manage001.

[udzura] ~/.ghq/
(*'-') < pec status manage
Current machine status:
 manage001.                    BUILD      sqale      m1.medium  nova       sample001  comp-node006

[udzura] ~/.ghq/
(*'-') <
```

pecの付加価値

- 立ち上げるだけじゃなく、
  ロールごとのスペックを構成管理できる
  - 簡易的なCloudFormation/Terraformといった趣
  - ロールがわかればだいたいアーキテクチャはわかる
- やるじゃん……

各プロジェクトに広まった

- ※筆者は最初は自作のnyah-cliで頑張ってました
- 今は使ってみて便利を感じてます…

さらに

自社の知っている人が
作ったOSSなので

貢献のモチベーションが
上がる！

pyama86 / pec

Watch 3 Star 5 Fork 5

社員間プルリク

manage availability_zone. #8

Merged **pyama86** merged 3 commits into `pyama86:master` from `sugy:add-az` on 21 Jul

Conversation 4 Commits 3 Files changed 4 +17 −8

sugy commented on 21 Jul

インスタンス作成時に availability_zone を指定出来るようにします。
また、status 表示で availability_zone を表示します。

SUGIYAMA Kaz... added some commits on 21 Jul

- `add availability_zone on status view.` dbcfbd2
- `add availability_zone in config.` c2970be
- `set availability_zone in option.` 7806ee6

pyama86 commented on 21 Jul Owner

うおーーー！！
有難うございます！
確認後リリースします。
マージ！

pyama86 merged commit **5a09558** into `pyama86:master` on 21 Jul View details Revert

Labels
None yet

Milestone
No milestone

Assignee
No one assigned

Notifications
Subscribe
You're not receiving notifications from this thread.

2 participants

OSSへの貢献？

- あるいはソーシャルコーディング
- ただやるには結構敷居が高い……
- なんか意識高そう……

「業務で必要なんで」

- ちゃんと、オープンソースで公開して
  - それを自分たちで使って
  - さらにそれを自分たちで直す
- Nyah周りで色々作ったことで、この流れを、後押しできるようになった。
- これはエンジニアにとって重要なことだと思う

共有の文化
コラボレーションの文化

GitHub:Enterprize

- ペパボはハードユーザー
- スーパーハードユーザー
  - >= 250 users
  - >= 40 organizations

- http://www.slideshare.net/hsbt/github-enterprise-with-gmo-pepabo

実は、Nyah自身も

様々な検証、相談、質問を

GHのプロジェクトのIssueで行っている

tech / nyah

Unwatch 245 Star 8 Fork 0

Issues Pull requests Labels

58 Open 220 Closed

Author Labels Milestones Assignee Sort

Mackerelのコミット台数を増やす(500→?)
#497 opened 3 hours ago by antipop

10/27 nyah ぽよん
#496 opened 3 days ago by hsbt

【MTG】Nyah 第34回 / 2015/10/21.水 キープMTG 1
#495 opened 8 days ago by ume3 1 of 3

ディスク容量が多くて CPU、メモリが程々の flavor がほしい 10
#494 opened 10 days ago by mickey

booklog-lan と subnets が被ってる謎のネットワークがある 4
#493 opened 11 days ago by okkun

glanceのイメージを整理する 6
#491 opened 14 days ago by pyama 6 of 125

[検証]リソースQuotaが使用できるか検証する 知見共有 12
#485 opened 22 days ago by pyama

db-nodeのメモリ使用率が高い 9
#454 opened on 3 Sep by pyama

We are 社内Social

ここまで、まとめ

- OpenStackを乗りこなすために:
  - ツールを作る。AWSのものを真似るのも手。
  - オープンソースで開発する。
  - 情報を共有する。GHを全力で使う。
- Be Open to utilize OpenStack!!

end

One More Thing

yaocloud/yao

Yet Another OpenStack API Wrapper

yaocloud / yao

Unwatch 4 Star 14 Fork 4

The OpenStack API wrapper — Edit

95 commits 2 branches 10 releases 4 contributors

Branch: master yao / +

udzura Bump Latest commit 69dbd39 6 days ago

lib	Bump	6 days ago
test	Refine config stubbing and handler matching	2 months ago
.gitignore	Rename oslo to YAO	2 months ago
.travis.yml	Update token	2 months ago
CODE_OF_CONDUCT.md	Add code of conduct	2 months ago
Gemfile	Update token	2 months ago
LICENSE.txt	Init	4 months ago
README.md	Make belonging to organization 'yaocloud'	2 months ago
Rakefile	Add minimal test and test task	2 months ago
yao.gemspec	Make belonging to organization 'yaocloud'	2 months ago

Code
Issues 0
Pull requests 0
Wiki
Pulse
Graphs
Settings

SSH clone URL
git@github.com:y

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop
Download ZIP

なぜ作ったか？

- OpenStack系のRubyライブラリ、フル機能のものは二つ
  - fog
  - ruby-openstack

どちらも、不満がある

- fog
  - http://fog.io/
  - 複数クラウドのラッパーなので依存が大
  - しかも、OpenStackの機能だけの切り出しがない(動きはあるが…
  - そもそも規模が大きく、動きも遅いように見える
    (PR取り込み後のリリースが遅いなど)
- ruby-openstack
  - https://github.com/ruby-openstack/ruby-openstack
  - 2013年で更新停止、16のPRが放置

```
50    s.add_dependency("fog-core", "~> 1.32")
51    s.add_dependency("fog-json")
52    s.add_dependency("fog-xml", "~> 0.1.1")
53
54    s.add_dependency("ipaddress", "~> 0.5")
55    s.add_dependency("nokogiri")
56
57    # Modular providers (please keep sorted)
58    s.add_dependency("fog-atmos")
59    s.add_dependency("fog-aws", ">= 0.6.0")
60    s.add_dependency("fog-brightbox", "~> 0.4")
61    s.add_dependency("fog-dynect", "~> 0.0.2")
62    s.add_dependency("fog-ecloud", "~> 0.1")
63    s.add_dependency("fog-google", ">= 0.1.1")
64    s.add_dependency("fog-local")
65    s.add_dependency("fog-powerdns", ">= 0.1.1
66    s.add_dependency("fog-profitbricks")
67    s.add_dependency("fog-radosgw", ">= 0.0.2")
68    s.add_dependency("fog-riakcs")
69    s.add_dependency("fog-sakuracloud", ">= 0.0
70    s.add_dependency("fog-serverlove")
71    s.add_dependency("fog-softlayer")
72    s.add_dependency("fog-storm_on_demand")
73    s.add_dependency("fog-terremark")
74    s.add_dependency("fog-vmfusion")
75    s.add_dependency("fog-voxel")
76    s.add_dependency("fog-xenserver")
77    s.add_dependency("fog-aliyun")
78
79    s.add_development_dependency("docker-api",
80    s.add_development_dependency("fission")
```

ところでOpenStackのAPI

- 非常に美しいRESTful API
  - ```
    GET    /v2/{tenant_id}/servers
    POST   /v2/{tenant_id}/servers
    DELETE /v2/{tenant_id}/servers/{server_id}
    …
    ```
  - see http://developer.openstack.org/api-ref.html && https://www.conoha.jp/docs/
- クラウドベンダによっては、かなり厳しいAPIのところもあるが、これなら……
    ```

RUBYGEMS.ORG

欲しいものは、作ればいい。

自分だけのオリジナルgemを作れる。OSSにできる。

オリジナル gemを作る

作ってみた所感

設計方針

- 低依存
  - 原則、faradayのみに依存
  - faradayは高機能で、コード削減量が素晴らしいので使う
  - Golangを書いていた影響はある
- RESTに沿ってリソースを一気に定義
- コンポーネントの違いを気にしないで済むように
  (nova or neutron or ……)

動くところまで、割とすぐできた

> こういうリソースを定義すればいい

1 lib/yao/resources.rb View

```
@@ -13,6 +13,7 @@ module Resources
     autoload :Network,          "yao/resources/network"
     autoload :Subnet,           "yao/resources/subnet"
     autoload :Port,             "yao/resources/port"
+    autoload :Tenant,           "yao/resources/tenant"

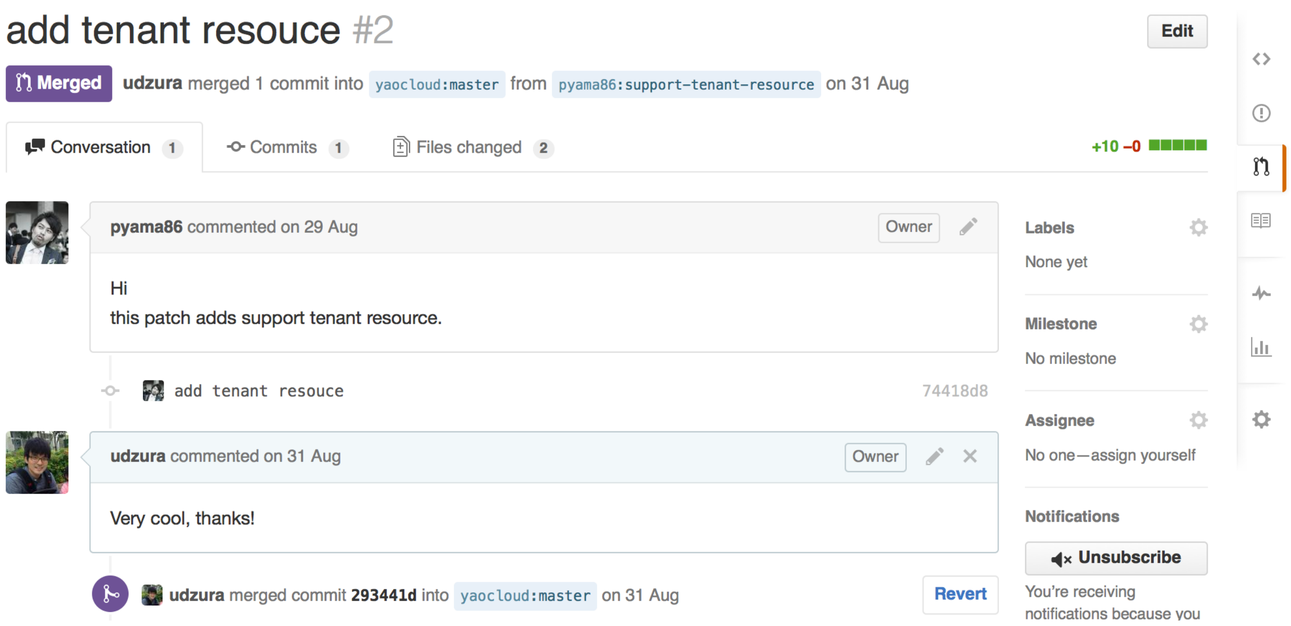
   end
```

9 lib/yao/resources/tenant.rb View

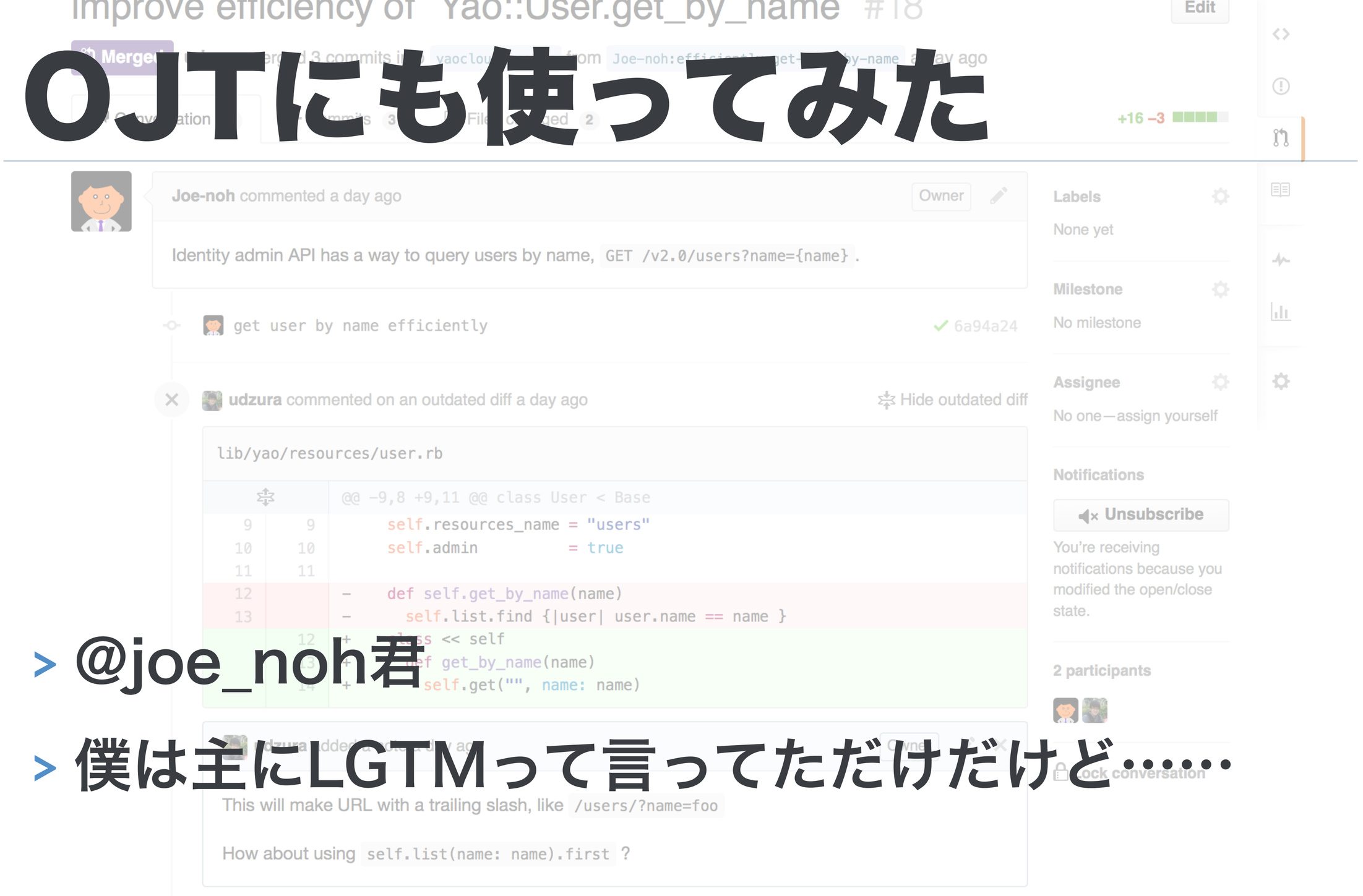
```
@@ -0,0 +1,9 @@
+module Yao::Resources
+  class Tenant < Base
+    friendly_attributes :id, :name, :description, :enabled
+
+    self.service        = "identity"
+    self.resource_name  = "tenant"
+    self.resources_name = "tenants"
+  end
+end
```

結果

- pecに採用された（fog -> yao）
  - いくつかPRをしてもらった（主にリソース追加）
  - 問題なく使えている様子
  - 依存はもちろん減った
  - 動作も少し速い？
    （体感、未検証）



OJTにも使ってみた



> @joe_noh君

> 僕は主にLGTMって言ってただけだけど……

学び

- あえての車輪の再発明も、条件が整えばアリ。
  - サポートしてくれるコミュニティ（この場合は社の皆）
  - ドッグフーディングする気概と体制
- テストはもっと頑張る必要がある……
  - APIのテストはwebmockが大変
  - VCR……社の内部API名が出てしまうし、など

総括

OpenStackを乗りこなすために

- 先人から学ぶ
  - AWS向けのツールを見てみる
- 社内コミュニティを使う
  - みんなでツールをOSSで開発する
  - 相談をワイワイと、バザール的に行う

Rubyはどう絡んだか？

- 1) Ops向けのツールにもやっぱりRubyは便利
  - Thorの存在
  - Faradayの存在
  - その他gem、特にテストスイートが充実している点
  - 内部DSLの設計と利用（今回はyamlだが）
  - 何より、（環境によるが）会社にRuby遣いが多い

Rubyはどう絡んだか？

- 2) Railsの開発＋GitHubが切り開いたソーシャルコーディングスタイル
  - みんなフラットに議論する
  - なんでも共有する、相談する
  - ツールをオープンにする
  - コラボレーションの文化を醸成する
  - これらが「当たり前」の組織を作れるとスピード⤴️

今後の課題

- 自力構築
  - 最新のバージョンにキャッチアップし、現在困っている箇所を頑張って解決
- OpenStackコミュニティ自体とのつながり
  - 社内→社外へ
  - 地域Ruby meetupほど頻繁じゃない印象
  - 同時に、地道にツールとワークフローを作ってOSSにしてく

Rubyの力を使って
今後もOpenStackを
乗りこなします！

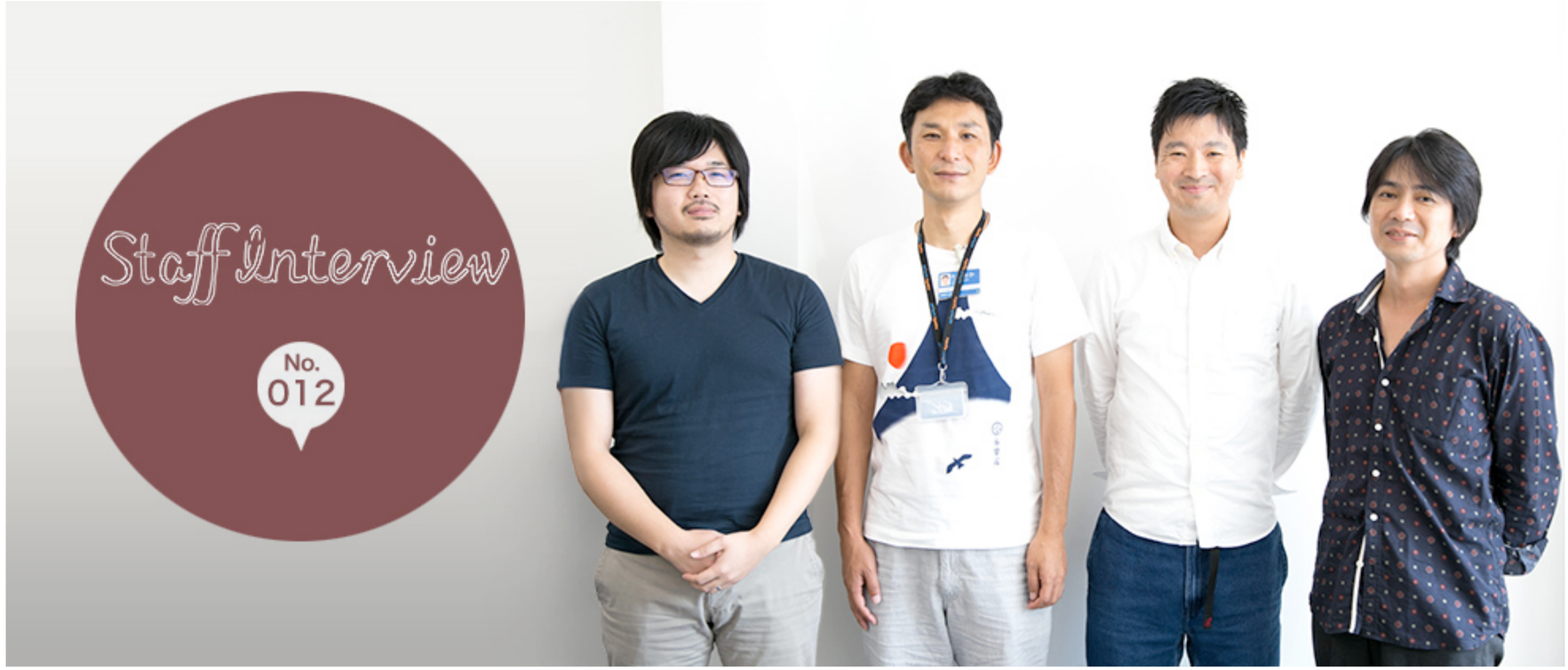
※ もちろんPythonも
勉強します

[PR]

ペパボにはOpenStack/Rubyを使う仕事があります

- 東京/福岡で特にインフラ絶賛募集！
- UIターン大歓迎！
- ペパランチョンでお話を（会場でも是非）。
  - https://pepabo.com/recruit/pepaluncheon/?fukuoka

インタビュー、おすすめです



- mrubyの話も出てきます！
- http://pepabo.com/recruit/interview/201510_01

afterword

- 本スライドに出てくる各社/各サービス/各プロダクトについて、ロゴ、名称等それぞれの団体に権利が帰属する場合があります。
- タイトル画像: public domain
  - https://pixabay.com/p-492412/?no_redirect