個人開発者のための
aws-serverless-express

HANATANI Takuma(@potato4d) 2018.03.23
#kyotojs

ア

- 自己紹介
- 個人開発におけるインフラ負担の辛さ
- PaaSによる解決
- aws-serverless-expressの良さ
- 個人開発者にとってのNode.jsのインフラのつらさ
- aws-serverless-expressのはじめかた

自己紹介

めぐみん

声 - 内田真礼 / 高橋李依 紅魔族の少女。13歳→14歳（WEB版では17歳前後）。スピンオフ「この素晴らしい世界に爆焔を！」の主人公。 黒髪に紅い目。黒マントに黒ローブ、トンガリ帽子を被り、左目に眼帯（別に目が悪いわけではなく、ただのファッション）をつけている。カズマとアクアの出したパーティーメンバー募集の張り紙を見て仲間に加わる。体格は小柄で細身、身長は

@potato4d

- 東京でフリーランスやってる
- だいたいフロントエンド
  - 最近は殆どNuxtの開発とかフロント技術相談
- Vue.js JP / VueFes Staff / Vue.js translator
- この間久々にNode.jsを生で書いたらdeep copyのためにobject-rest-spreadを使ってCIを落とした人

本題

いきなりですが皆さん！

個人開発してますか？

WebApp開発してますか？

個人開発の「インフラ」
つらくないですか？

「メンテナンス」に
新しいことをやる時間を
奪われていませんか？

本質的でない作業で
モチベを下げていませんか？

例えば「こんなこと」
起こってないですか？

個人開発のインフラメンテのつらみ

- IaaSを借りているとセキュリティアップデートなどに確実に追われてどこかで対応するハメになる
- オートスケールの構成にしないと負荷に対処できないけど、個人でオートスケールはお財布がつらい
- 何故かDockerが壊れて対応で趣味開発の時間が溶ける
- この苦しみがWebAppの数だけ増えていく

ありがち

作れば作るほど新しいものを
作る時間が減っていく

メンテしたくない

どうにかして解決したい

今日はこれを解決しよう

よくある解決法

PaaSを使う


heroku

Heroku使いがち

- 言語を特に問わずに使えるので便利
- 基本無料だし有料で24h起動にしてもたかだか月800円ぐらい
- Let’s Encrypt の自動更新もできるしAddonもあって死ぬほど便利
- 大体これでよくなる

けどつらい時もある

Herokuでつらいとき

- レイテンシがキツイ
  - 個人開発でFastlyを使えるわけがないんだよな
- Naked domainのSSLに対応してない
  - ルートドメインでSSLしたいならCloudFront噛ませろ
- 大量に動かしだすと無料枠が2秒で枯渇する
- スケーリング性能が終了している

GAE Standard使いがち

- Googleが好きな言語だとLambda的な実行時間課金でアプリケーションを動かせるので安い
- cronとかもついてて意外に高性能
- 日本リージョンが勿論あるのではやくて最高っぽい
- Naked domainのSSL対応もバッチリ
- デフォでオートスケールする
- 大体これでよくなる

これもつらい時がある

GAE Standardでつらいとき

- Googleが好きな言語以外は高いプラン(Flexible)しかない
  - その好きな言語はPython/Go/Javaあたりでつらい
- FlexibleはDockerコンテナを動かすだけなので普通に月額でGCEの料金がかかって高くてつらい
- FlexibleはElasticBeanstalkみたいなもん

Nodeでの開発者に人権がない

そこで

もう一つの解決方法

FaaSに載せる

Lambdaとか CloudFunctions

FaaS

- Node.jsで書ける(LambdaはPython/Javaも)
- Lambdaはイベントの汎用性を、CloudFunctionsは Express LikeなI/Fが魅力な形で提供されている
- レイテンシも初回起動が微妙ぐらい
- 勿論実行時間課金で安い
- 逆に無限にリクエストがきたら金をかけてオートスケール無限にしてくれて便利

けどつらさもある

FaaSのつらさ

- 結局一定以上独自記法に落ち着く
  - 既存のNode.jsアプリケーションの作り方で作ること
    ができるわけではない
- 技術的に汎用性がない（ロックイン）
- ローカル実行がとにかくめんどくさい
  - API Gatewayから渡ってくる event の構造が.....
- 完成品はよくても開発途中がつらい

つらい

FaaSの特徴を持った

GAE/SE for Node.jsがあれば……

そんな時の救世主

aws-serverless-express

ところで

aws-serverless-express
を知っている人？

aws-serverless-express

使ったことある人？

aws-serverless-express

- Amazonがノリで作っちゃった

  “AWS LambdaでExpressが動く” ライブラリ

- 既存のExpressアプリケーションにMiddlewareを追加するだけで何故かLambdaで動く

- サーバーレスの恩恵を受けながら慣れ親しんだExpress記法でアプリケーションを記述できるのが魅力

Source Code

```
const serverlessExpress = require('aws-serverless-express/middleware')

app.use(serverlessExpress.eventContext())

app.get('/', (req, res) => {

  res.json(req.apiGateway.event)

})
```

実際に使ってるWebApp

https://connpassport.com

良さ

どこが良いのか

- **普通にNode.jsで開発しているのに**何故か無限にスケールする
- **普通にNode.jsで開発しているのに**何故か実行時課金
- **普通にNode.jsで開発しているのに**なぜかメンテナンスフリー
- Naked domainのSSL、リージョンによるレイテンシ、ロックイン……さっきまでの全部の課題が解消

欲しかったもの全部入り

aws-serverless-expressは
全てを解決してくれる

つらさのおさらい

- そもそもIaaSはメンテがつらい
- Herokuはレイテンシがつらい
- GAE/SEにはない
- FaaSはロックインがつらい

解決できているか

- レイテンシはないか

  ->ランタイムの初回起動の遅さだけ

- マネージドPaaS Likeに使えるか

  ->使えて最高

- 独自記法はないか

  ->全てExpressそのまま

解決できているか

- スケーリング
  - めちゃめちゃするし逆に人がいない時はお金がかからないのでどっちの意味でも最高
- 運用コスト
  - 基本的にFaaSなので皆無、証明書もACM
  - API GW + ACM + Route53になるのでNaked OK

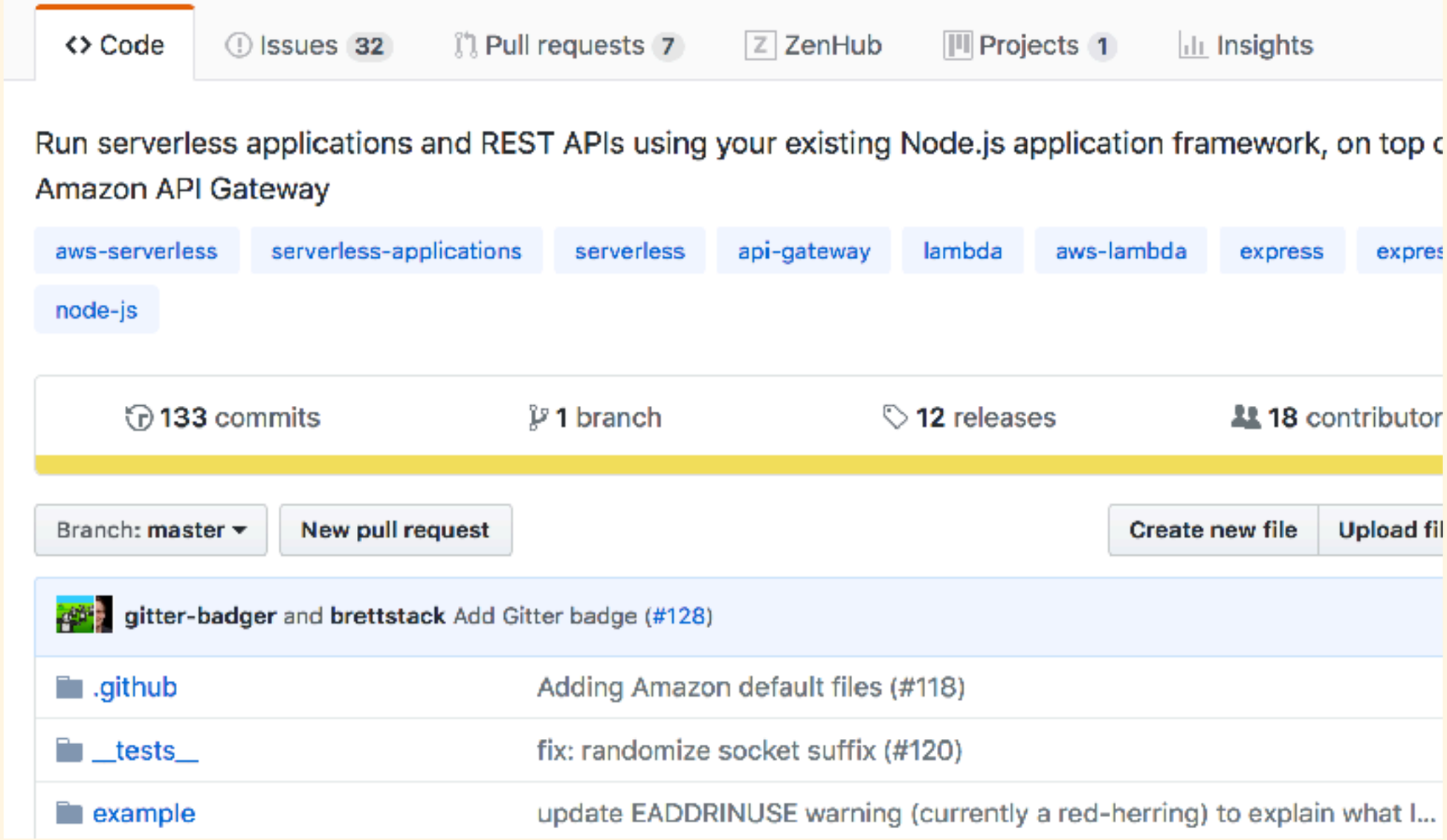
素敵！

使いたい！

はじめかた

awslabs/aws-serverless-express

にアクセス



おもむろにexampleをクリック

awslabs / aws-serverless-express

Watch 68 | Unstar 1,499 | Fork 192

Code | Issues 32 | Pull requests 7 | ZenHub | Projects 1 | Insights

Branch: master ▾ aws-serverless-express / example / Create new file | Upload files | Find file | History

brettstack update EADDRINUSE warning (currently a red-herring) to explain what l... Latest commit 649eaf2 on Jun 9, 2017

..		
scripts	update EADDRINUSE warning (currently a red-herring) to explain what l...	10 months ago
views	fix: decodes base64 requests (fixes #84); added form to index view to...	11 months ago
.gitignore	update to use SAM, and add deconfigure script	a year ago
README.md	fix: decodes base64 requests (fixes #84); added form to index view to...	11 months ago
api-gateway-event.json	Eaddrinuse (#71)	11 months ago
app.js	fix: decodes base64 requests (fixes #84); added form to index view to...	11 months ago
app.local.js	fix: decodes base64 requests (fixes #84); added form to index view to...	11 months ago
cloudformation.yaml	fix: decodes base64 requests (fixes #84); added form to index view to...	11 months ago
lambda.js	fix: decodes base64 requests (fixes #84); added form to index view to...	11 months ago
package.json	update example dependencies including aws-serverless-express to 3.0.0	11 months ago
sam-logo.png	add pug template and example of using middleware	a year ago
simple-proxy-api.yaml	fix: decodes base64 requests (fixes #84); added form to index view to...	11 months ago

Win/Mac/Linux対応のテンプレ

Example

In addition to a basic Lambda function and Express server, the `example` directory includes a Swagger file, CloudFormation template with Serverless Application Model (SAM), and helper scripts to help you set up and manage your application.

Steps for running the example

This guide assumes you have already set up an AWS account and have the latest version of the AWS CLI installed.

1. From your preferred project directory: `git clone https://github.com/awslabs/aws-serverless-express.git && cd aws-serverless-express/example`.
2. Run `npm run config -- --account-id="<accountId>" --bucket-name="<bucketName>" [--region="<region>" --function-name="<functionName>"]` to configure the example, eg. `npm run config -- --account-id="123456789012" --bucket-name="my-unique-bucket"`. This modifies `package.json`, `simple-proxy-api.yaml` and `cloudformation.yaml` with your account ID, bucket, region and function name (region defaults to `us-east-1` and function name defaults to `AwsServerlessExpressFunction`). If the bucket you specify does not yet exist, the next step will create it for you. This step modifies the existing files in-place; if you wish to make changes to these settings, you will need to modify `package.json`, `simple-proxy-api.yaml` and `cloudformation.yaml` manually.
3. Run `npm run setup` (Windows users: `npm run win-setup`) - this installs the node dependencies, creates an S3 bucket (if it does not already exist), packages and deploys your serverless Express application to AWS Lambda, and creates an API Gateway proxy API.
4. After the setup command completes, open the AWS CloudFormation console

READMEがめっちゃ丁寧

既存プロジェクトの移行

yarn add aws-serverless-express

lambda.jsを作成

```
const awsServerlessExpress = require('aws-serverless-express')
const app = require('./app')
const server = awsServerlessExpress.createServer(app)


exports.handler = (event, context) => awsServerlessExpress.proxy(server, event, context)
```

Express側(e.g. app.js)に追記

```
const serverlessExpress = require('aws-serverless-express/middleware')

app.use(serverlessExpress.eventContext())
```

このままLambdaにデプロイで

OK

まとめ

aws-serverless-express

- LambdaでExpressを動かすライブラリ
- Expressミドルウェアとして開発されており、既存プロジェクトへの導入も可能
- Expressのかき心地でLambdaのPros/Consを引き継ぐアプリケーションを作ることが可能

よさ

- 現状のPaaS事情では満たせない要件を満たせる
  - 低レイテンシ
  - 実行時間課金かつオートスケーリング
  - Node.js環境が動く
  - Naked domainのSSL対応
  - の全部入り環境

はじめるには

- 公式レポジトリのexampleがボイラープレートなのでそのまま使える
- 既存プロジェクトへの導入もREADMEにあるとおりにmiddlewareを追加するだけ

使おう

おわりに

GAE Standard Node.js 8.x Early Access

We are looking for teams and individuals with experience developing Node.js web applications.

If you are a hobbyist regularly experimenting with new technology or part of a team deploying enterprise applications we want to talk to you.

You will get access to an experimental runtime with scale-to-zero capabilities, <1 minute deploy times, and Google-grade security. In return we ask that you provide thoughtful feedback and are available for infrequent video calls.

Spots are limited so please fill out this form to be considered.

****IMPORTANT**: You will need to sign our Trusted Tester Agreement for access: https://cloud-tta.appspot.com/**

*必須

First Name *

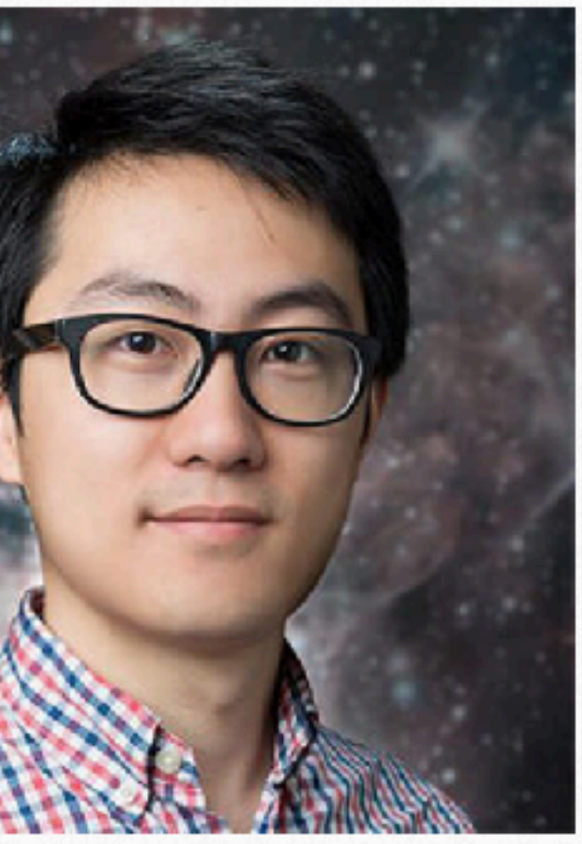
Takuma

GAE/SE for Node 来るやんけ！

宣伝


Vue Fes Japan
2018.11.3 [Sat.]

Speaker



Vue.js クリエーター

Evan You

Evan は開発者、デザイナー、そしてクリエイティブコーダーです。彼は、リアクティブなコンポーネントでモダンな Web インターフェイスを構築するための JavaScript フレームワーク、Vue.js の作者です。

かつて、GitHub で最もスターを集めたフルスタック JavaScript フレームワークである Meteor の開発グループでも働いていました。Google Creative Lab で、さまざまなプロダクト向けの実験的な UI プロトタイプに、2年間取り組んでいた経験もあり

他スピーカーは決定次第、更新予定です。

ご清聴ありがとうございました