AWS re:Invent

DVO209

JAWS: The Monstrously Scalable Serverless Framework

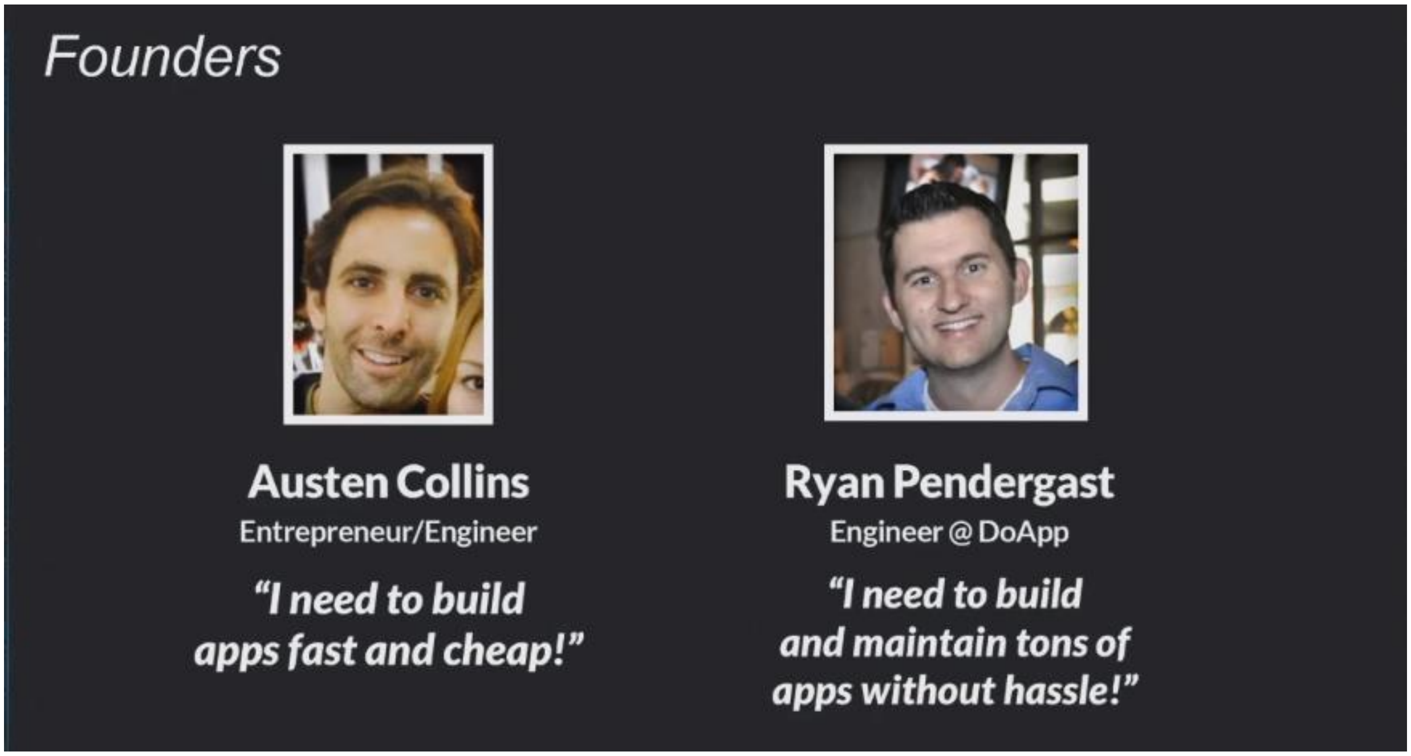
AWS Lambda, API Gateway & More!

Austen Collins, JAWS
Ryan Pendergast, DoApp, Inc.

October 2015

You can now build entire applications without servers using AWS Lambda and Amazon API Gateway - JAWS just makes it easier. JAWS is an open-source application framework that provides structure, best practices, and optimizations for serverless applications through its powerful command line tool and module ecosystem. Join us as we build an application with JAWS and discuss tips and tricks for building serverless apps in general.

Github: https://github.com/jaws-stack/JAWS



Founders

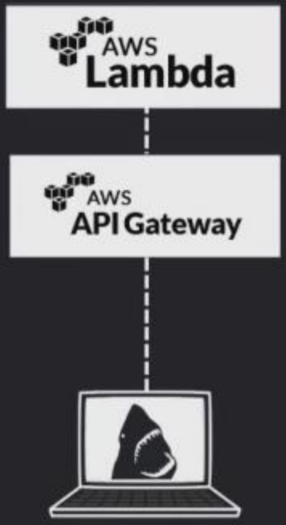
Austen Collins
Entrepreneur/Engineer
"I need to build apps fast and cheap!"

Ryan Pendergast
Engineer @ DoApp
"I need to build and maintain tons of apps without hassle!"

SERVERLESS AWS IS HERE!

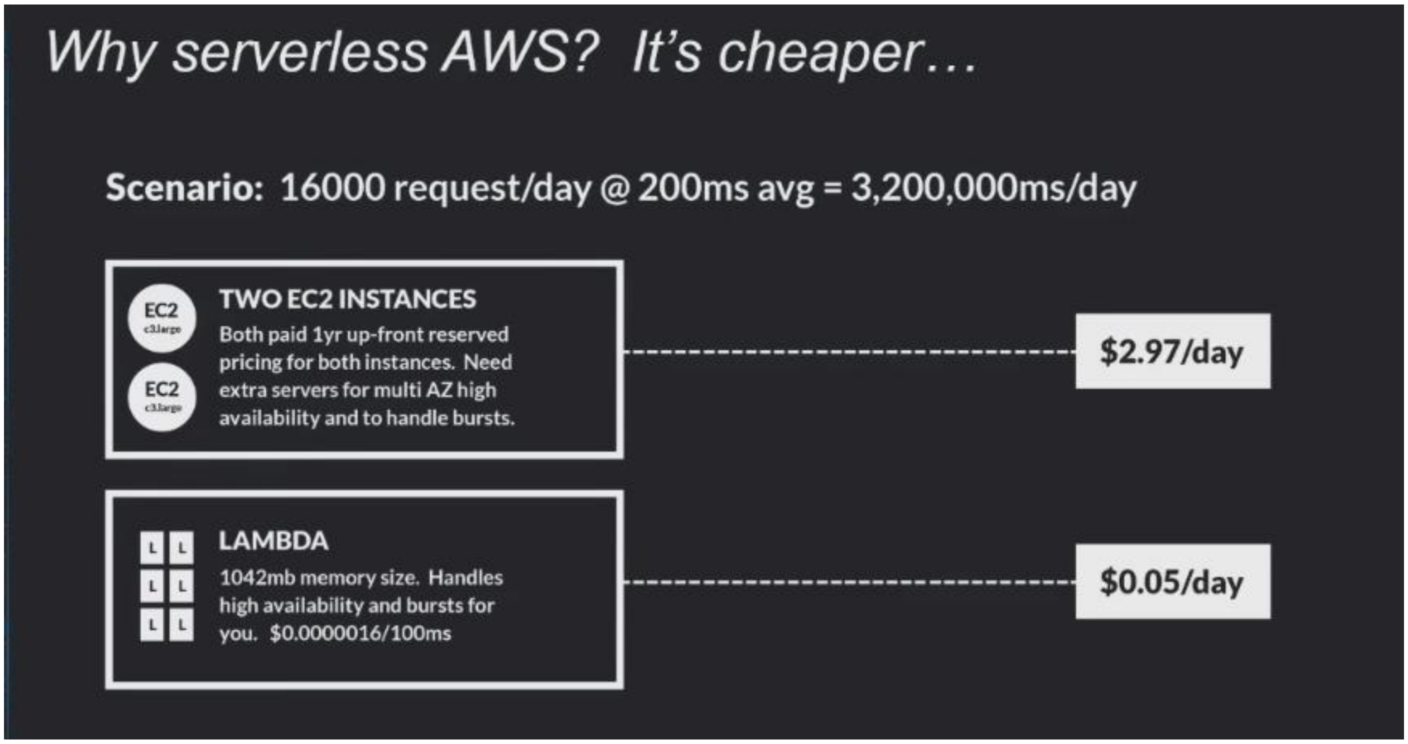
What is serverless AWS?

- Serverless = Not having to think about servers
- Lambda: Event-driven compute resources
- API Gateway: Build REST APIs for Lambdas
- Serverless AWS = Lambda + API Gateway
- Allow you to cut out PaaS/BaaS middle men
- *“Lambda has the potential to be the focal point AWS cloud”* – Janakiram MSV (janakiram.com)



Why serverless AWS? It’s easy to scale…

- Though Lambda runs in containers, you don’t have to deal with containers!
- Orchestration/autoscale of containers handled for you
- API Gateway handles DDoS + rate limiting
- Distributed application logging and metrics are built in



Why serverless AWS? It's cheaper…

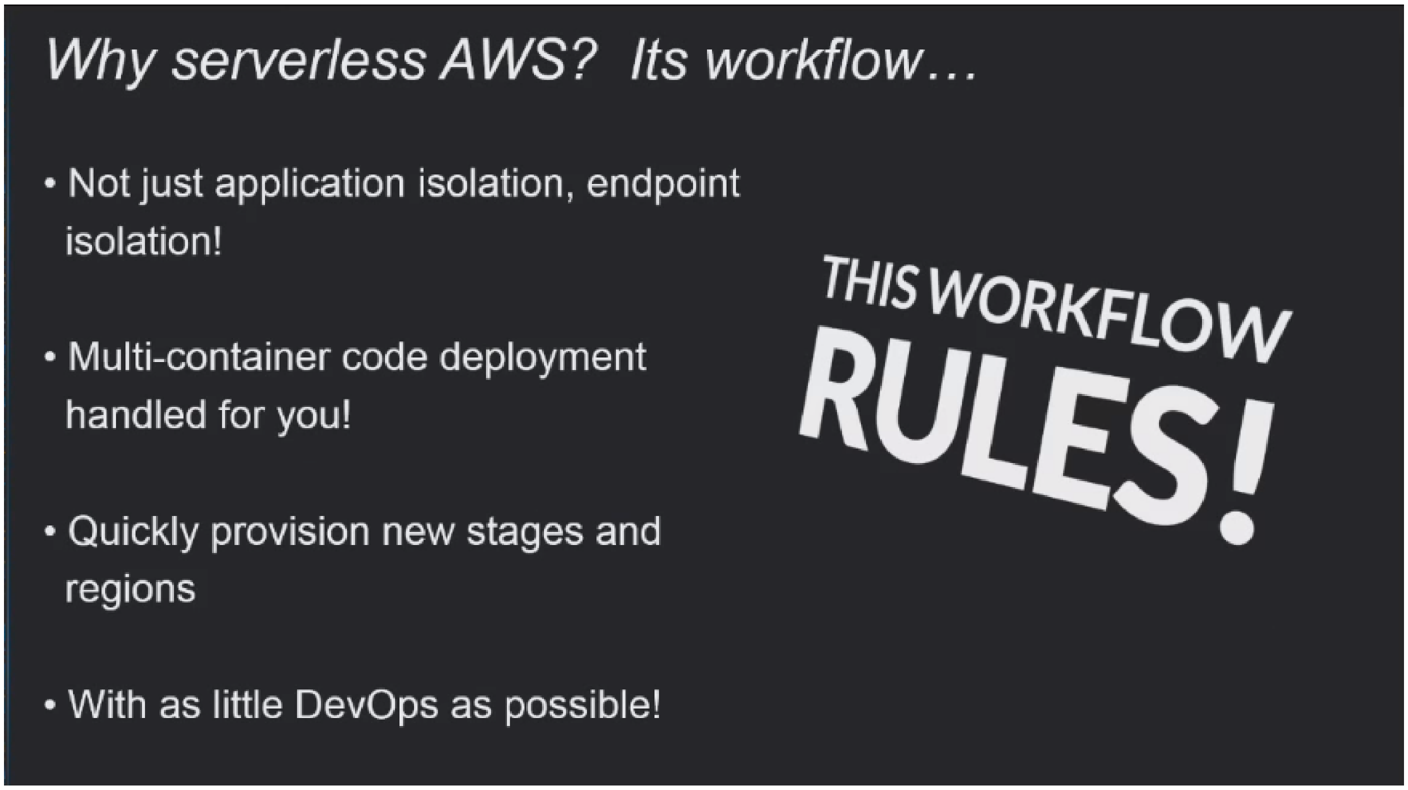
Scenario: 16000 request/day @ 200ms avg = 3,200,000ms/day

TWO EC2 INSTANCES
Both paid 1yr up-front reserved pricing for both instances. Need extra servers for multi AZ high availability and to handle bursts.

$2.97/day

LAMBDA
1042mb memory size. Handles high availability and bursts for you. $0.0000016/100ms

$0.05/day

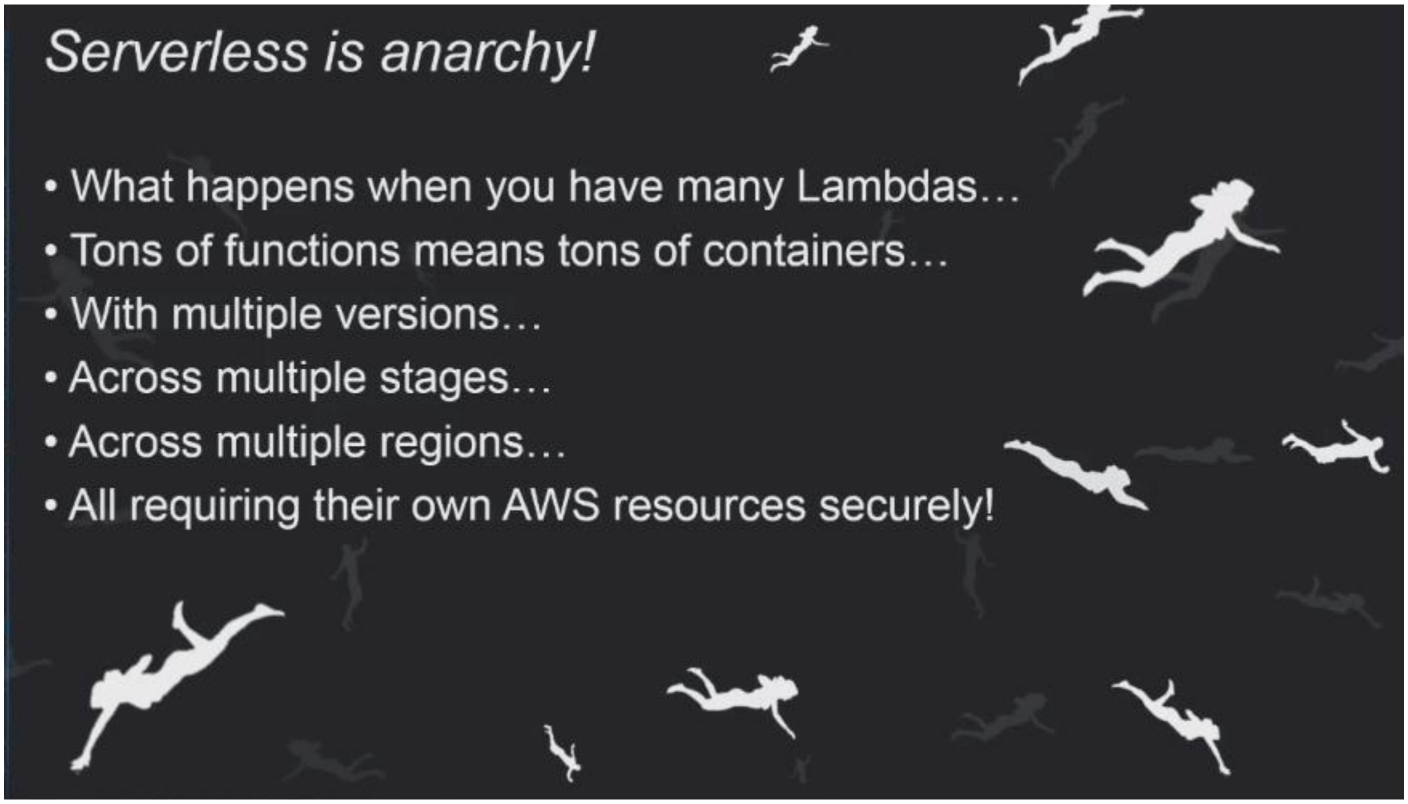


Why serverless AWS? Its workflow…

- Not just application isolation, endpoint isolation!
- Multi-container code deployment handled for you!
- Quickly provision new stages and regions
- With as little DevOps as possible!

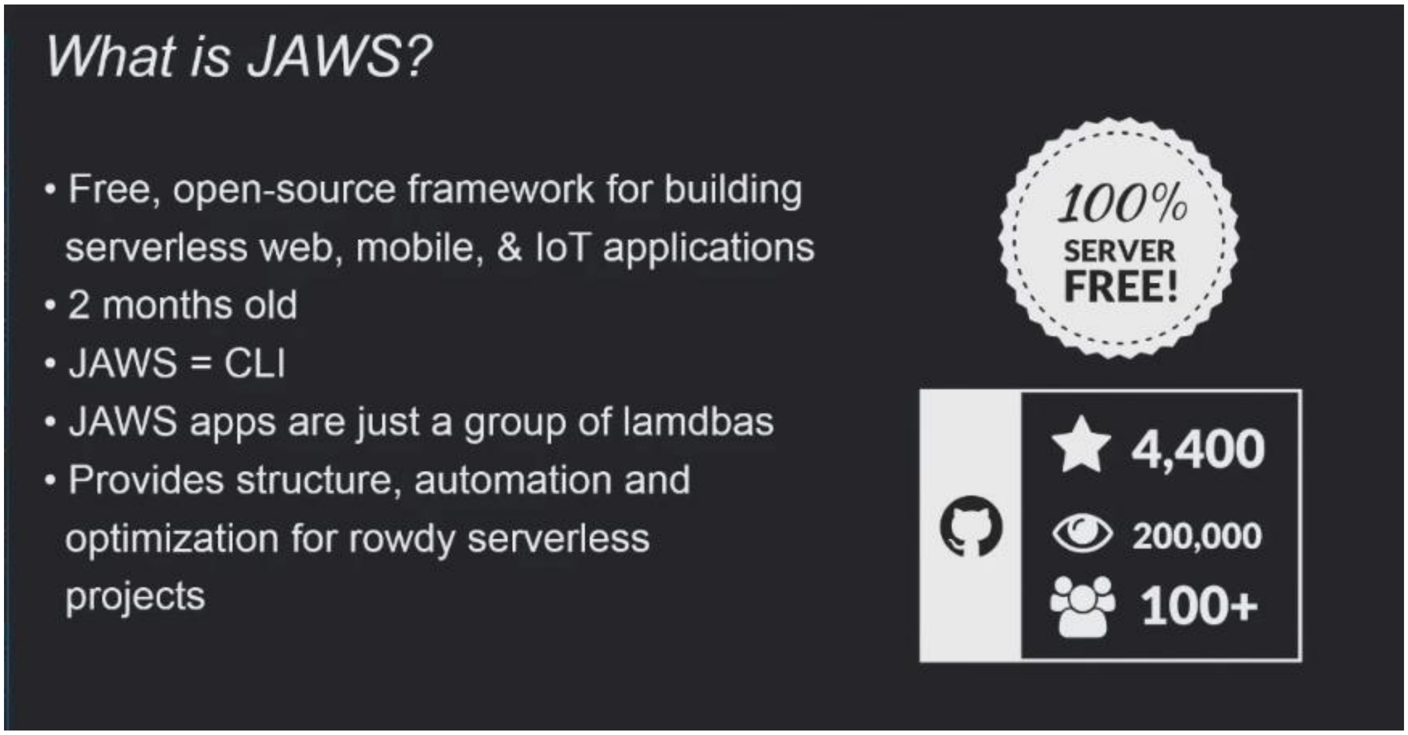
THIS WORKFLOW **RULES!**

BUT SERVERLESS IS ALSO…



Serverless is anarchy!

- What happens when you have many Lambdas…
- Tons of functions means tons of containers…
- With multiple versions…
- Across multiple stages…
- Across multiple regions…
- All requiring their own AWS resources securely!



What is JAWS?

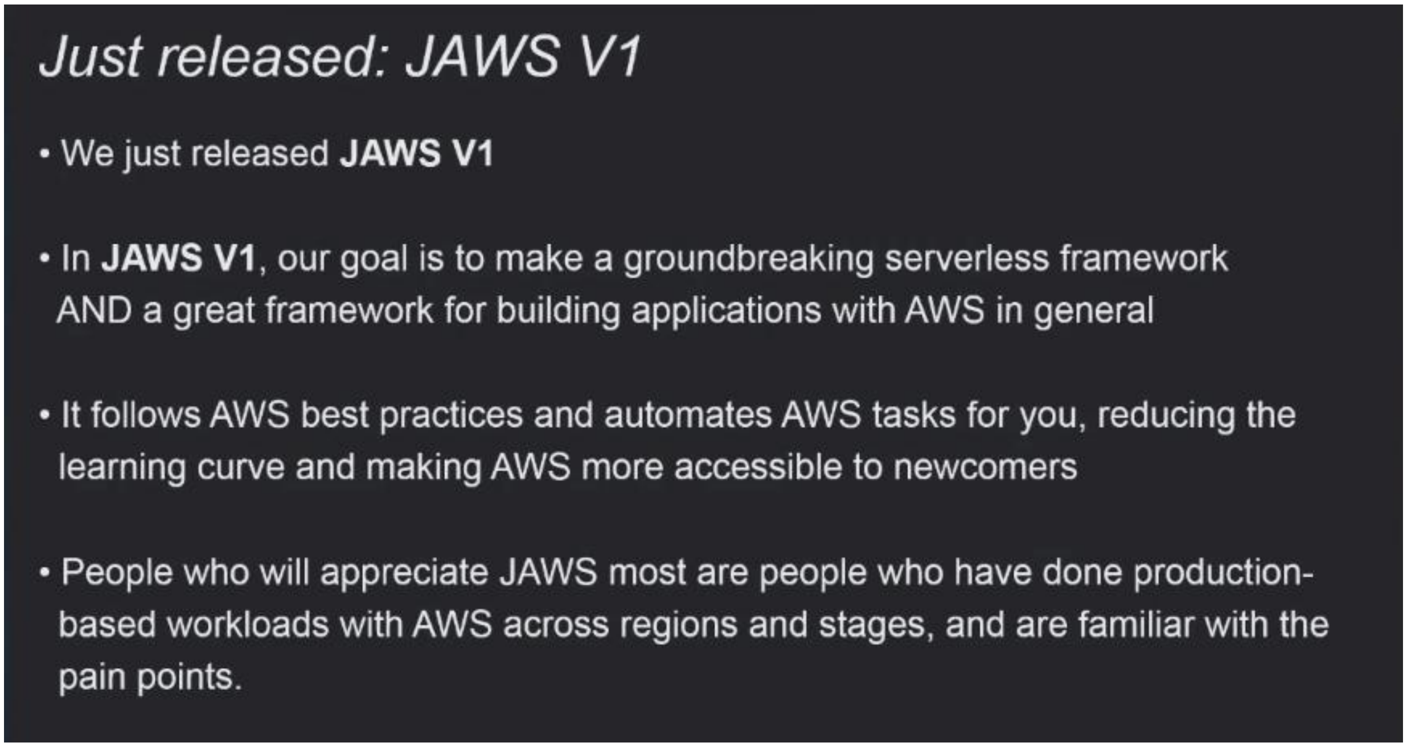
- Free, open-source framework for building serverless web, mobile, & IoT applications
- 2 months old
- JAWS = CLI
- JAWS apps are just a group of lamdbas
- Provides structure, automation and optimization for rowdy serverless projects

100% SERVER FREE!

★ 4,400

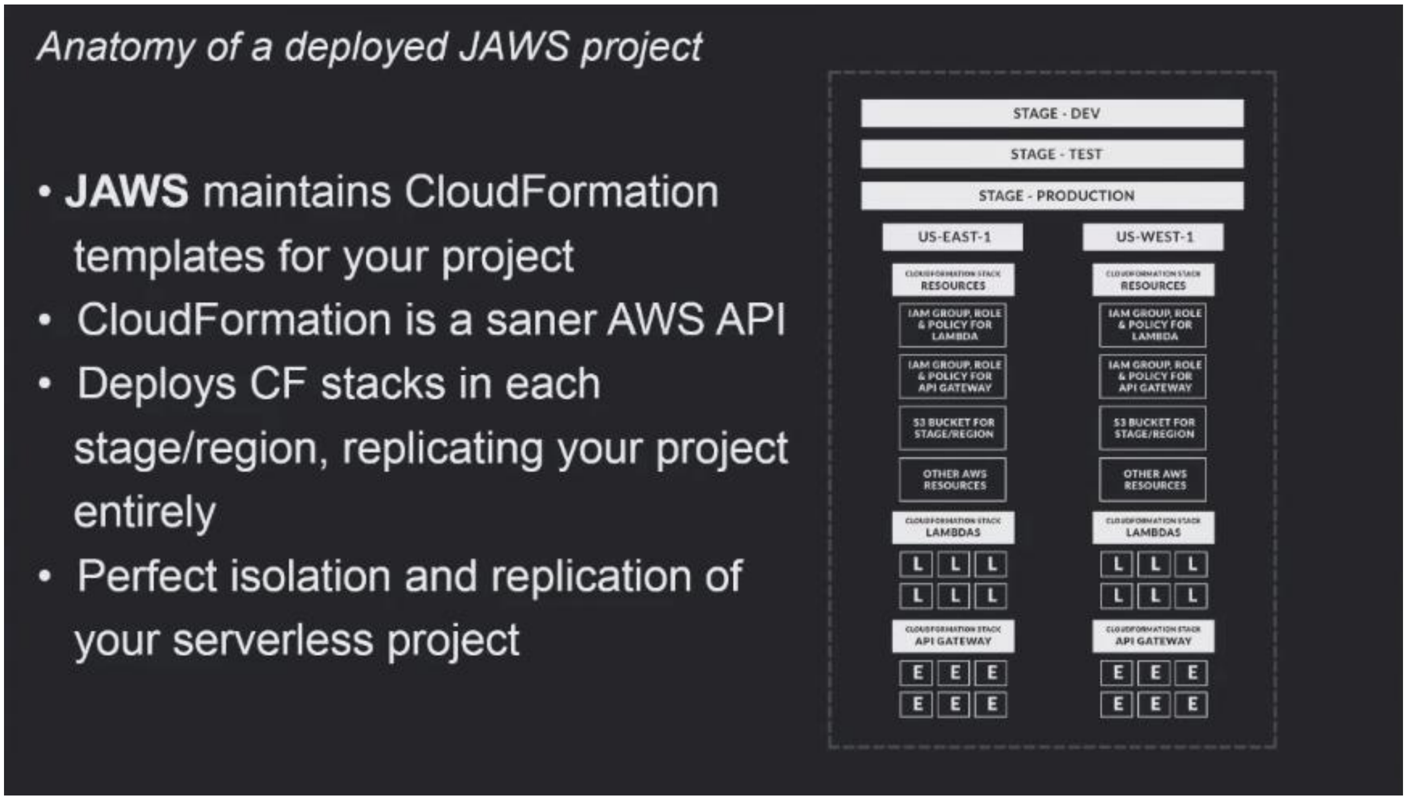
200,000

100+



Just released: JAWS V1

- We just released **JAWS V1**
- In **JAWS V1**, our goal is to make a groundbreaking serverless framework AND a great framework for building applications with AWS in general
- It follows AWS best practices and automates AWS tasks for you, reducing the learning curve and making AWS more accessible to newcomers
- People who will appreciate JAWS most are people who have done production-based workloads with AWS across regions and stages, and are familiar with the pain points.



Anatomy of a deployed JAWS project

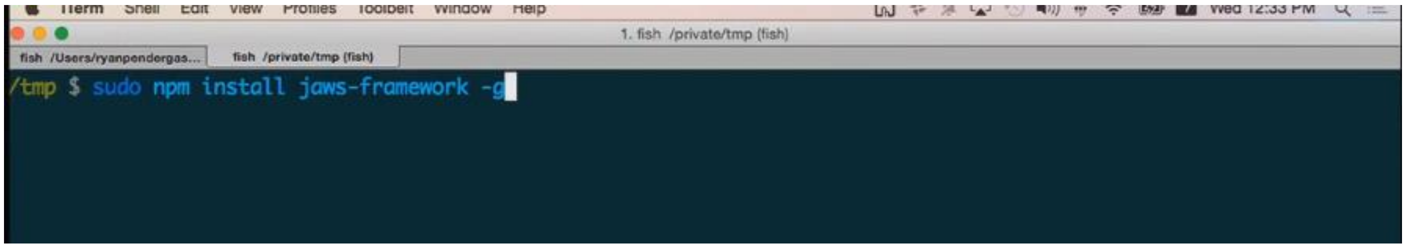
- **JAWS** maintains CloudFormation templates for your project
- CloudFormation is a saner AWS API
- Deploys CF stacks in each stage/region, replicating your project entirely
- Perfect isolation and replication of your serverless project

STAGE - DEV
STAGE - TEST
STAGE - PRODUCTION
US-EAST-1
US-WEST-1
CLOUDFORMATION STACK RESOURCES
IAM GROUP, ROLE & POLICY FOR LAMBDA
IAM GROUP, ROLE & POLICY FOR API GATEWAY
S3 BUCKET FOR STAGE/REGION
OTHER AWS RESOURCES
CLOUDFORMATION STACK LAMBDAS
L
CLOUDFORMATION STACK API GATEWAY
E

This is what a JAWS application looks like when deployed on AWS, every JAWS project comes with 3 CloudFormation templates because CF has a 200-resource limit. JAWS will deploy a CF stack in each stage and region to replicate your project entirely. Stages are like environments, you can easily replicate all your resources within each stage across regions.

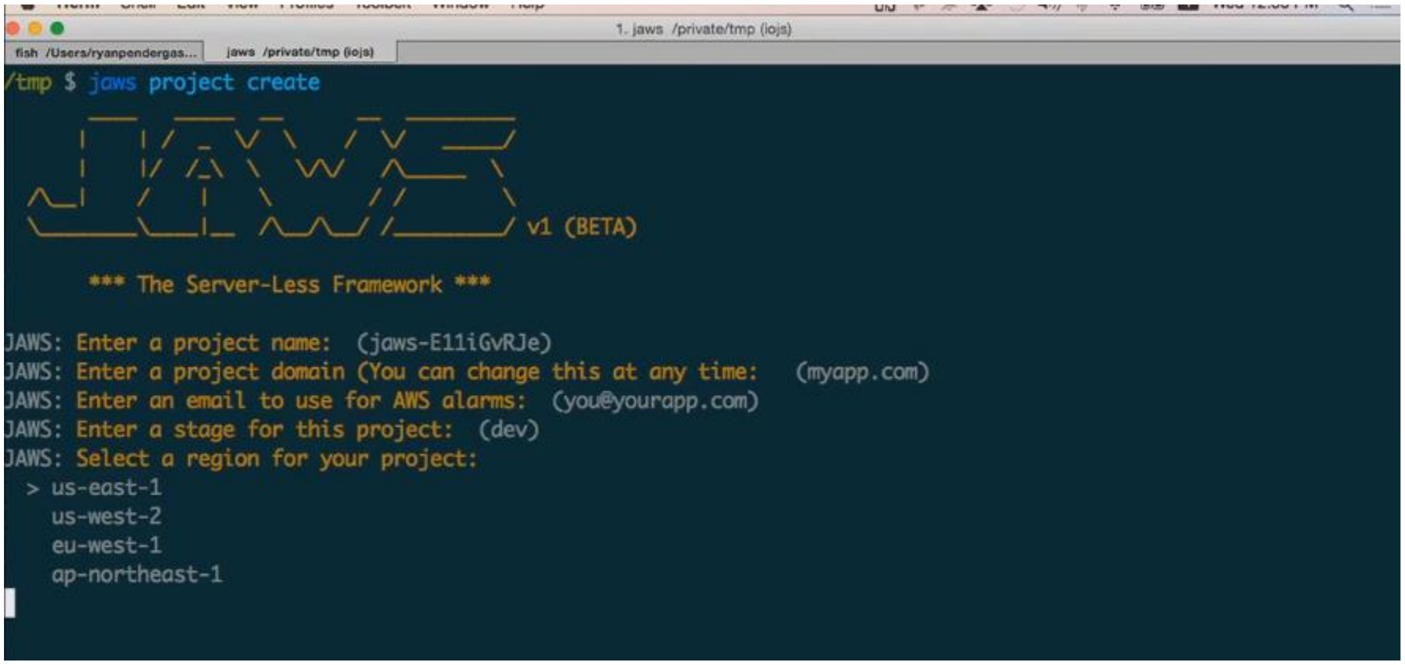
DEMO

INSTALLING JAWS

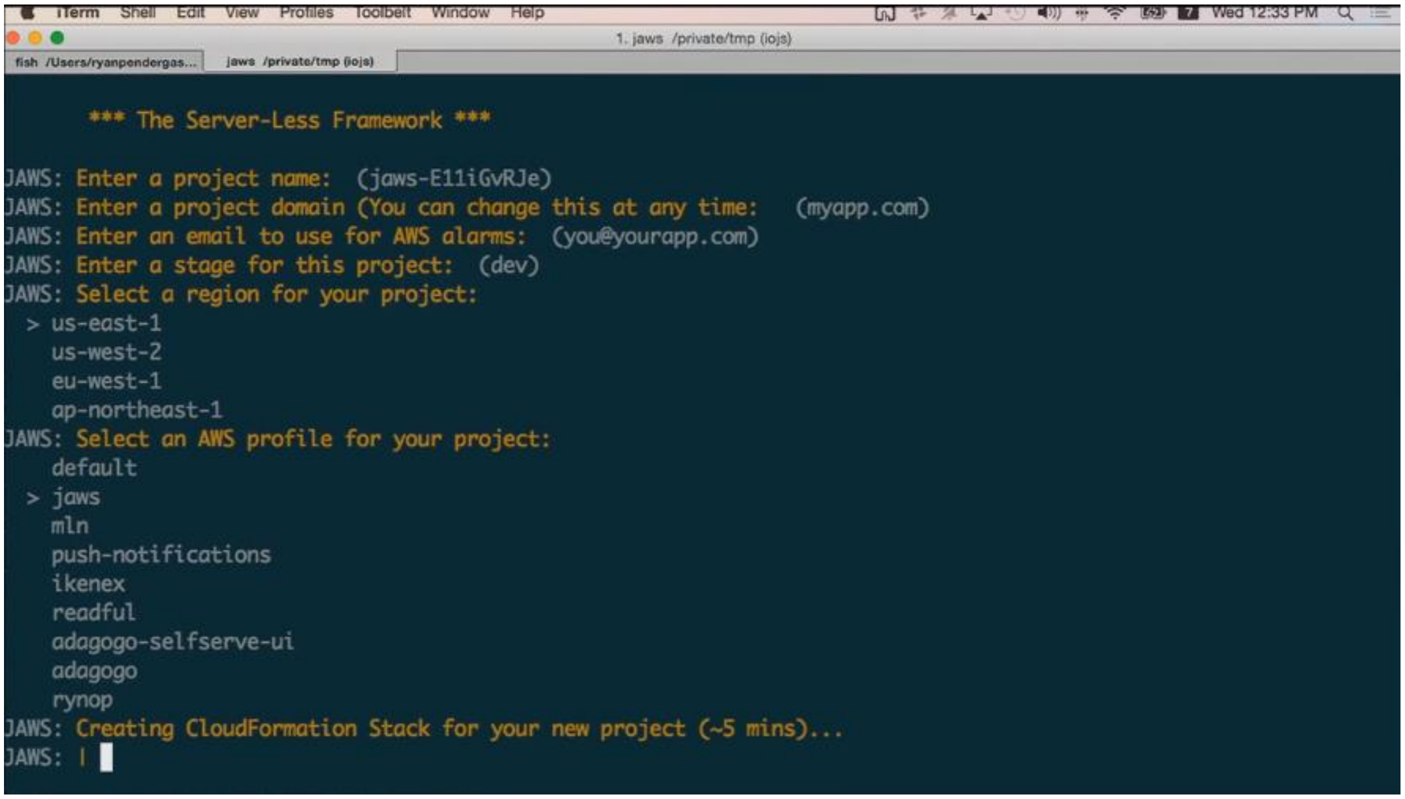

iTerm Shell Edit View Profiles Toolbelt Window Help
Wed 12:33 PM
1. fish /private/tmp (fish)
fish /Users/ryanpendergas...
fish /private/tmp (fish)

```
/tmp $ sudo npm install jaws-framework -g
```

JAWS is just an NPM module and can be installed using the command ***$ npm install jaws-framework -g***

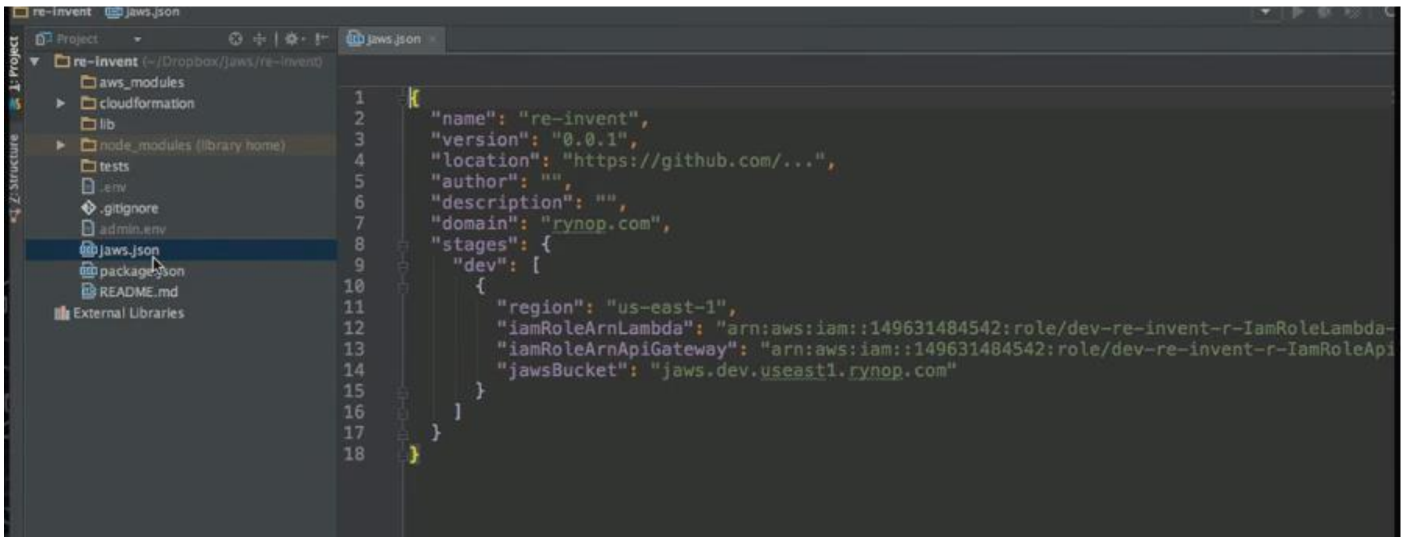


```
/tmp $ jaws project create
        ____  _____  __      __  _________
       |    |/ _   \/  \    /  \/   _____/
       |    |/ /_\  \   \/\/   /\_____  \
   /\__|    /    |    \        //        \
   \________\____|__  /\__/\__/ /_________/ v1 (BETA)

       *** The Server-Less Framework ***

JAWS: Enter a project name:  (jaws-E11iGvRJe)
JAWS: Enter a project domain (You can change this at any time:   (myapp.com)
JAWS: Enter an email to use for AWS alarms:  (you@yourapp.com)
JAWS: Enter a stage for this project:  (dev)
JAWS: Select a region for your project:
  > us-east-1
    us-west-2
    eu-west-1
    ap-northeast-1
```


iTerm Shell Edit View Profiles Toolbelt Window Help
Wed 12:33 PM
1. jaws /private/tmp (iojs)
fish /Users/ryanpendergas...
jaws /private/tmp (iojs)

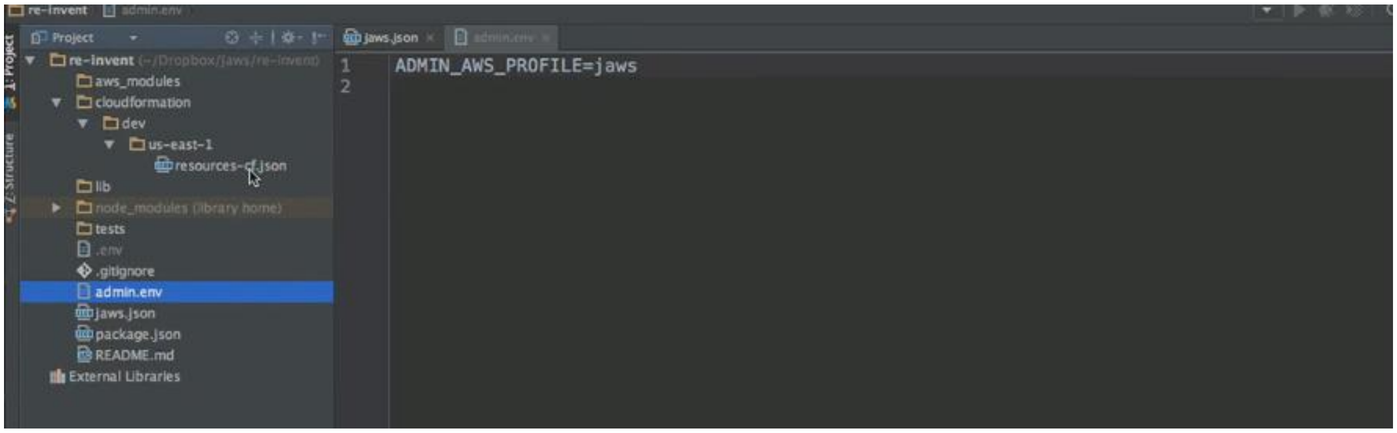
```
      *** The Server-Less Framework ***

JAWS: Enter a project name:  (jaws-E11iGvRJe)
JAWS: Enter a project domain (You can change this at any time:   (myapp.com)
JAWS: Enter an email to use for AWS alarms:  (you@yourapp.com)
JAWS: Enter a stage for this project:  (dev)
JAWS: Select a region for your project:
  > us-east-1
    us-west-2
    eu-west-1
    ap-northeast-1
JAWS: Select an AWS profile for your project:
    default
  > jaws
    mln
    push-notifications
    ikenex
    readful
    adagogo-selfserve-ui
    adagogo
    rynop
JAWS: Creating CloudFormation Stack for your new project (~5 mins)...
JAWS: |
```

This is the part that will use CF to create the IAM Roles, provisioning the S3 bucket to store all your templates for this project. This takes about 5 mins to set up per project per region per stage.



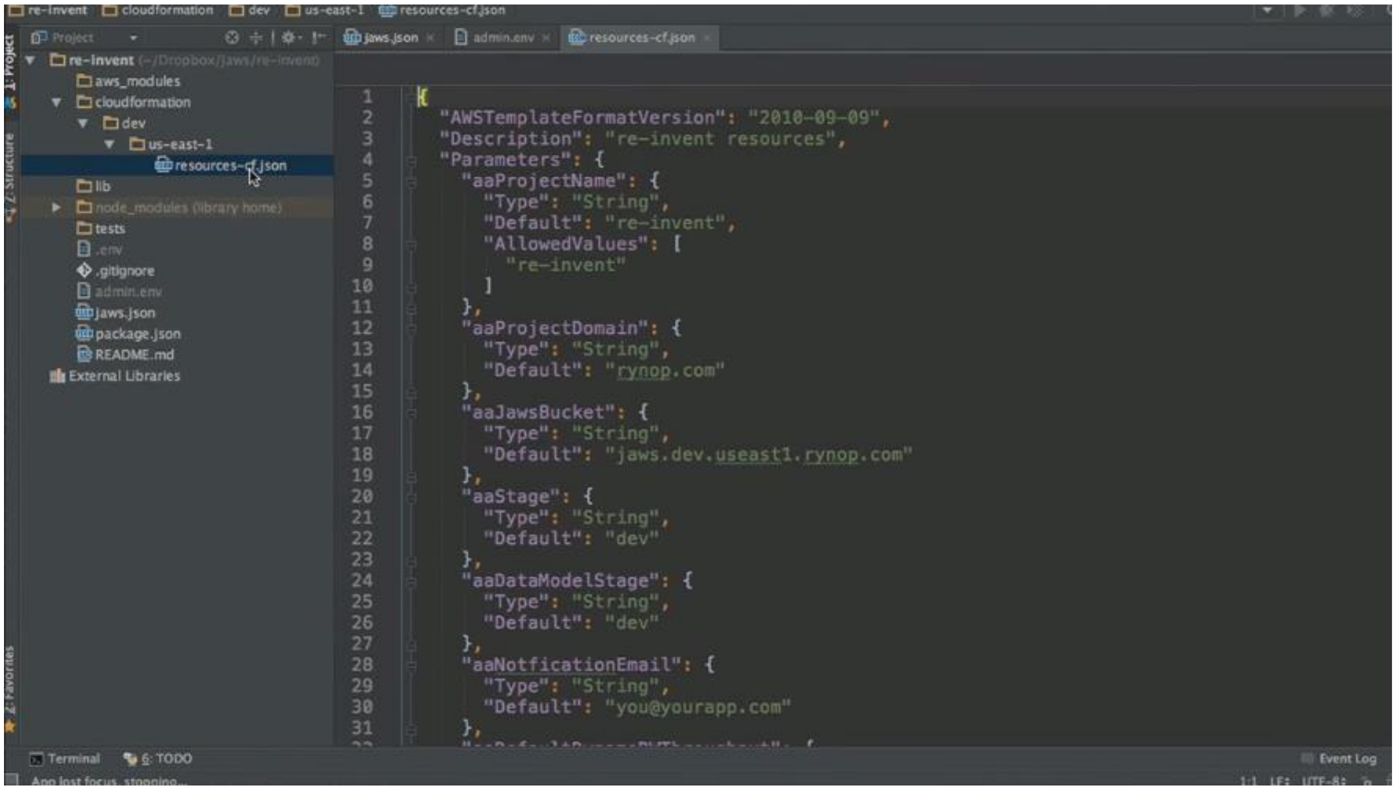
```
{
  "name": "re-invent",
  "version": "0.0.1",
  "location": "https://github.com/...",
  "author": "",
  "description": "",
  "domain": "rynop.com",
  "stages": {
    "dev": [
      {
        "region": "us-east-1",
        "iamRoleArnLambda": "arn:aws:iam::149631484542:role/dev-re-invent-r-IamRoleLambda-
        "iamRoleArnApiGateway": "arn:aws:iam::149631484542:role/dev-re-invent-r-IamRoleApi
        "jawsBucket": "jaws.dev.useast1.rynop.com"
      }
    ]
  }
}
```

This is the scaffolding that JAWS creates for a new project. The jaws.json file contains metadata about your project, it includes every stage and region.



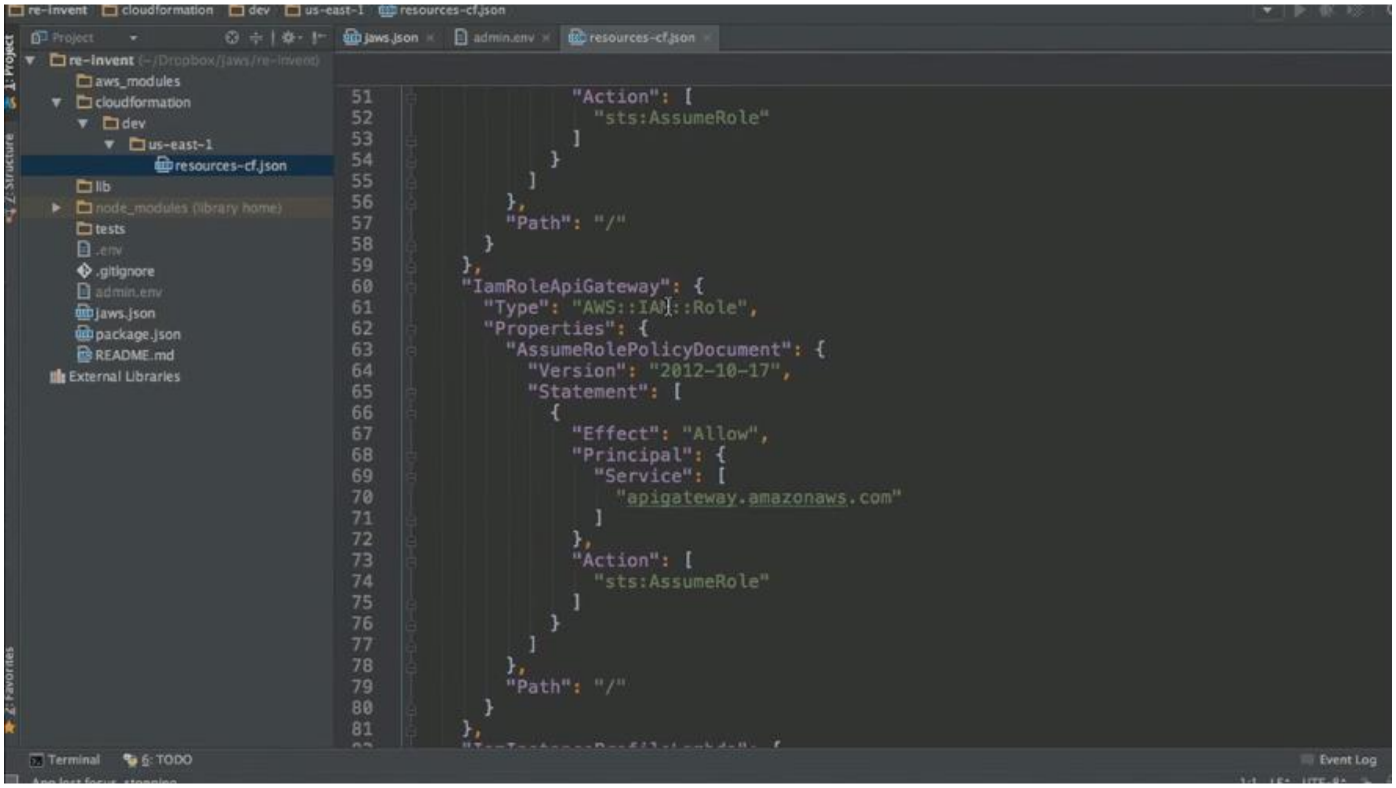
```
ADMIN_AWS_PROFILE=jaws
```

This is the way to tie your project back to the AWS profile you have on your file system,

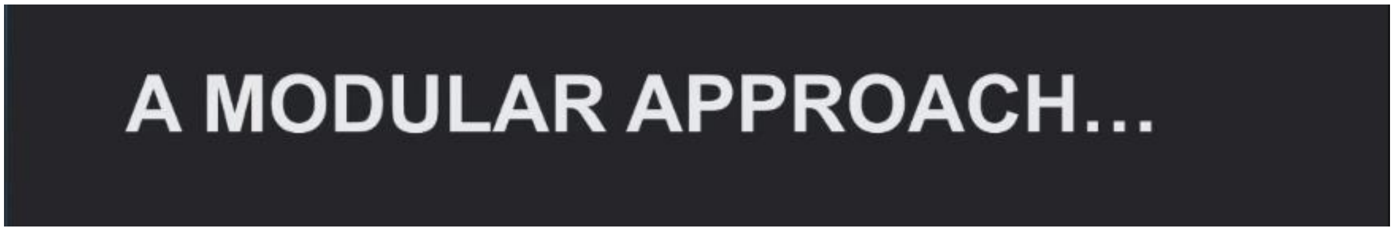


```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "re-invent resources",
  "Parameters": {
    "aaProjectName": {
      "Type": "String",
      "Default": "re-invent",
      "AllowedValues": [
        "re-invent"
      ]
    },
    "aaProjectDomain": {
      "Type": "String",
      "Default": "rynop.com"
    },
    "aaJawsBucket": {
      "Type": "String",
      "Default": "jaws.dev.useast1.rynop.com"
    },
    "aaStage": {
      "Type": "String",
      "Default": "dev"
    },
    "aaDataModelStage": {
      "Type": "String",
      "Default": "dev"
    },
    "aaNotficationEmail": {
      "Type": "String",
      "Default": "you@yourapp.com"
    },
```

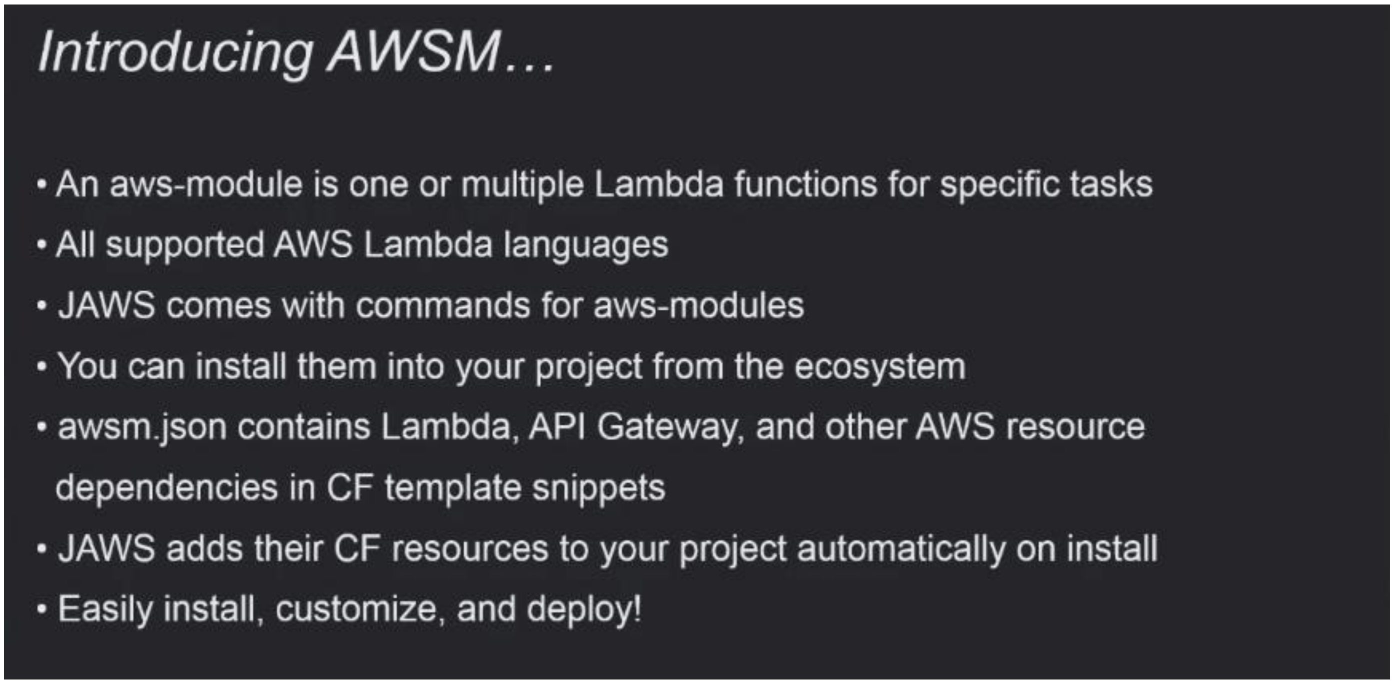
This is one of the CF files that is created for you, it contains all your resources, stages, projects, domains, IAM Roles, S3 buckets,



```
                "Action": [
                  "sts:AssumeRole"
                ]
              }
            ]
          },
          "Path": "/"
        }
      },
      "IamRoleApiGateway": {
        "Type": "AWS::IAM::Role",
        "Properties": {
          "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
              {
                "Effect": "Allow",
                "Principal": {
                  "Service": [
                    "apigateway.amazonaws.com"
                  ]
                },
                "Action": [
                  "sts:AssumeRole"
                ]
              }
            ]
          },
          "Path": "/"
        }
      },
```



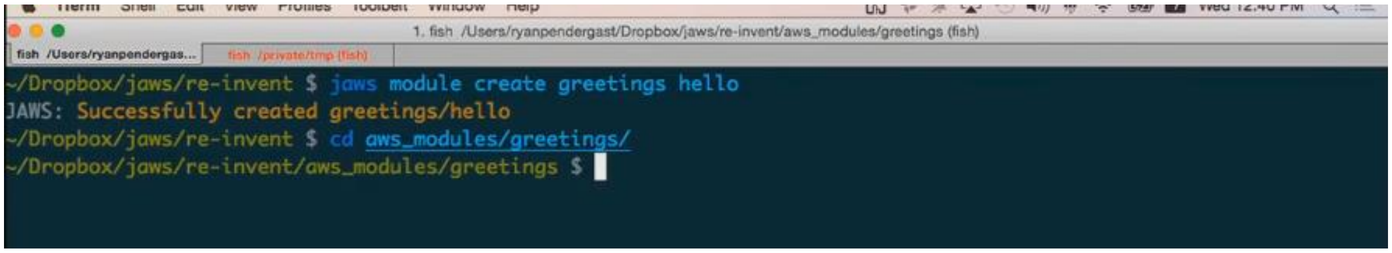
A MODULAR APPROACH…


Introducing AWSM…

- An aws-module is one or multiple Lambda functions for specific tasks
- All supported AWS Lambda languages
- JAWS comes with commands for aws-modules
- You can install them into your project from the ecosystem
- awsm.json contains Lambda, API Gateway, and other AWS resource dependencies in CF template snippets
- JAWS adds their CF resources to your project automatically on install
- Easily install, customize, and deploy!

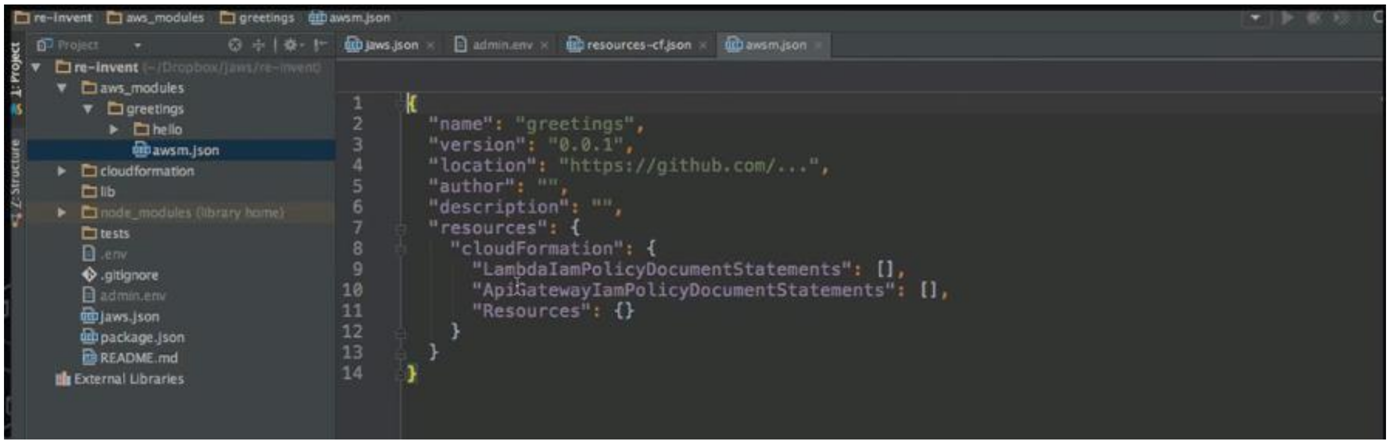
DEMO

CREATING AN AWS-MODULE


1. fish /Users/ryanpendergast/Dropbox/jaws/re-invent/aws_modules/greetings (fish)
fish /Users/ryanpendergas...
fish /private/tmp (fish)

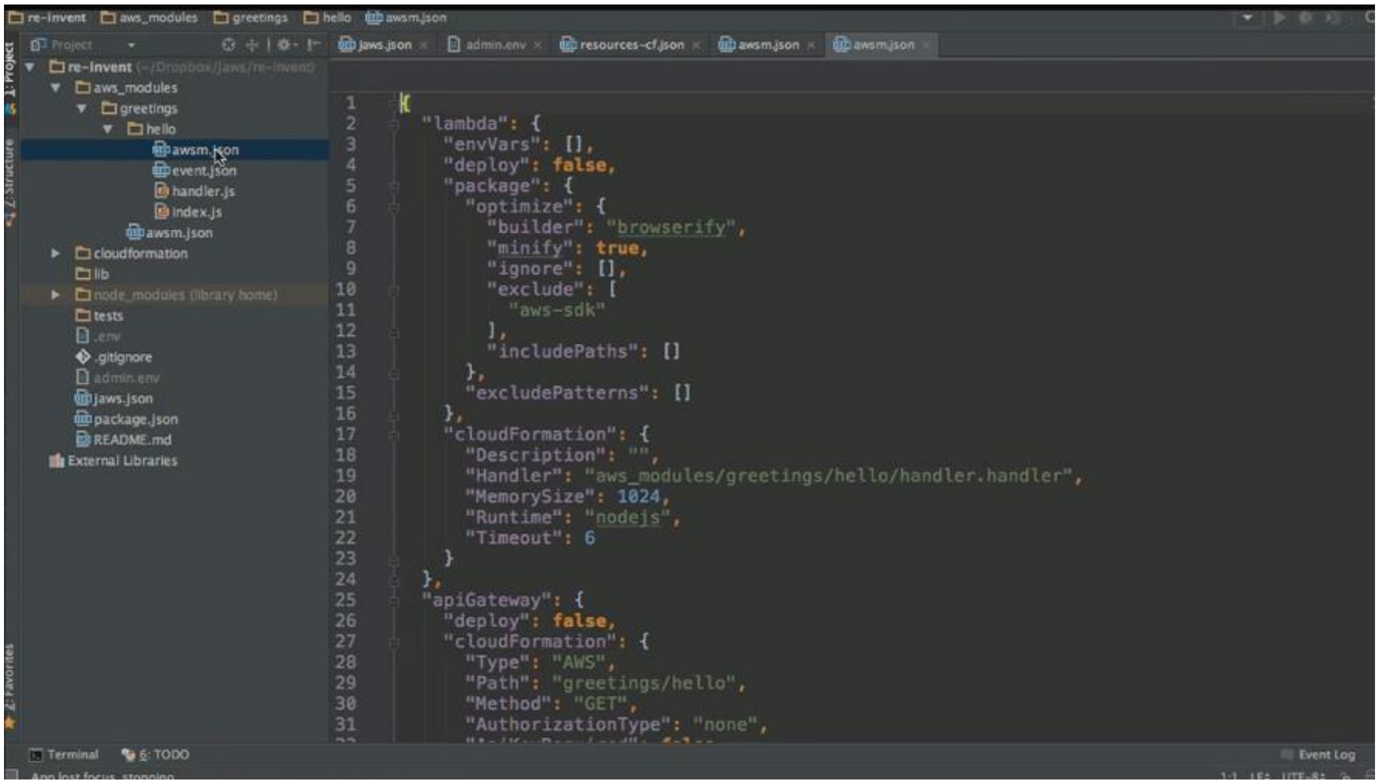
```
~/Dropbox/jaws/re-invent $ jaws module create greetings hello
JAWS: Successfully created greetings/hello
~/Dropbox/jaws/re-invent $ cd aws_modules/greetings/
~/Dropbox/jaws/re-invent/aws_modules/greetings $
```

The name of the module is ***greetings*** and ***hello*** will be the name of the first lambda function in that module, since a module can contain one or more lambda functions.



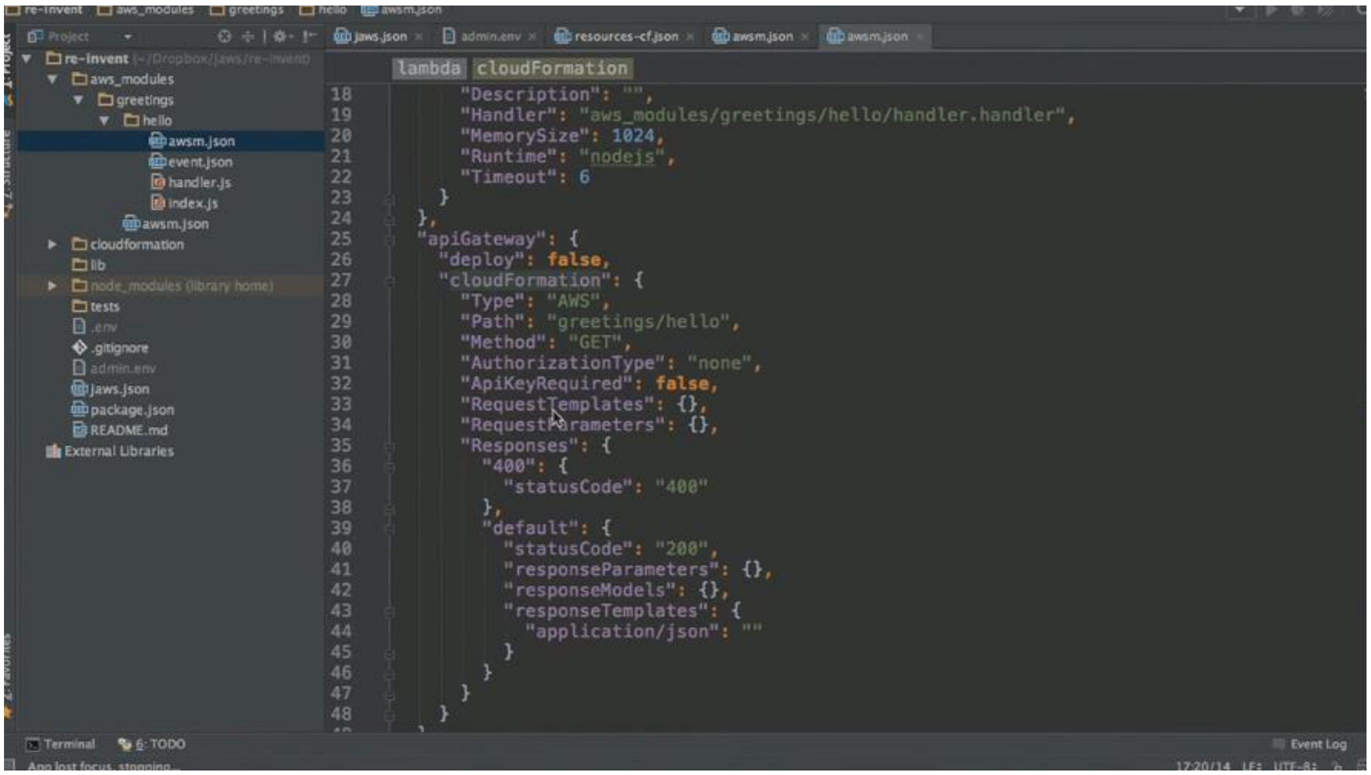
```
{
  "name": "greetings",
  "version": "0.0.1",
  "location": "https://github.com/...",
  "author": "",
  "description": "",
  "resources": {
    "cloudFormation": {
      "LambdaIamPolicyDocumentStatements": [],
      "ApiGatewayIamPolicyDocumentStatements": [],
      "Resources": {}
    }
  }
}
```

This is the ***aswm.json*** for the module



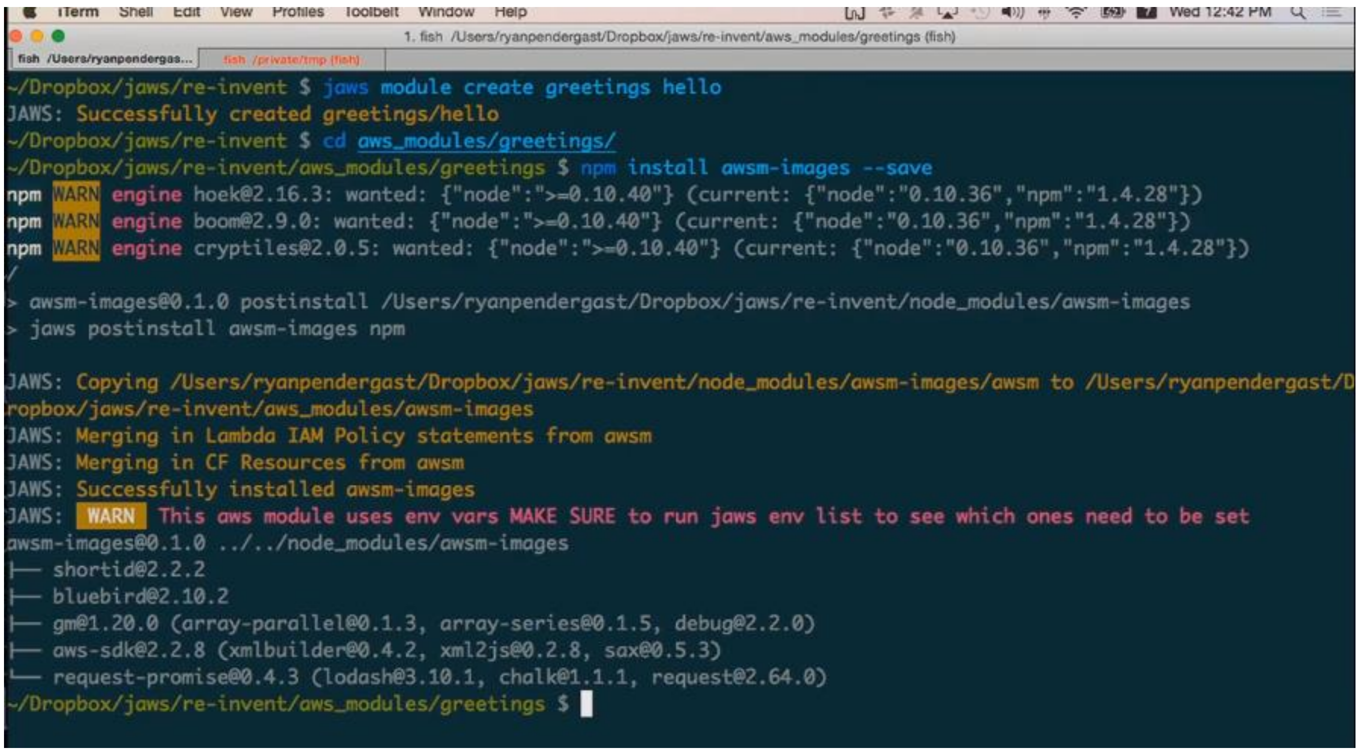
```
{
  "lambda": {
    "envVars": [],
    "deploy": false,
    "package": {
      "optimize": {
        "builder": "browserify",
        "minify": true,
        "ignore": [],
        "exclude": [
          "aws-sdk"
        ],
        "includePaths": []
      },
      "excludePatterns": []
    },
    "cloudFormation": {
      "Description": "",
      "Handler": "aws_modules/greetings/hello/handler.handler",
      "MemorySize": 1024,
      "Runtime": "nodejs",
      "Timeout": 6
    }
  },
  "apiGateway": {
    "deploy": false,
    "cloudFormation": {
      "Type": "AWS",
      "Path": "greetings/hello",
      "Method": "GET",
      "AuthorizationType": "none",
```

This is where a lot of the heavy lifting is done for you like setting env variables, using a package object for NodeJS that browserify's your code into small size, etc.



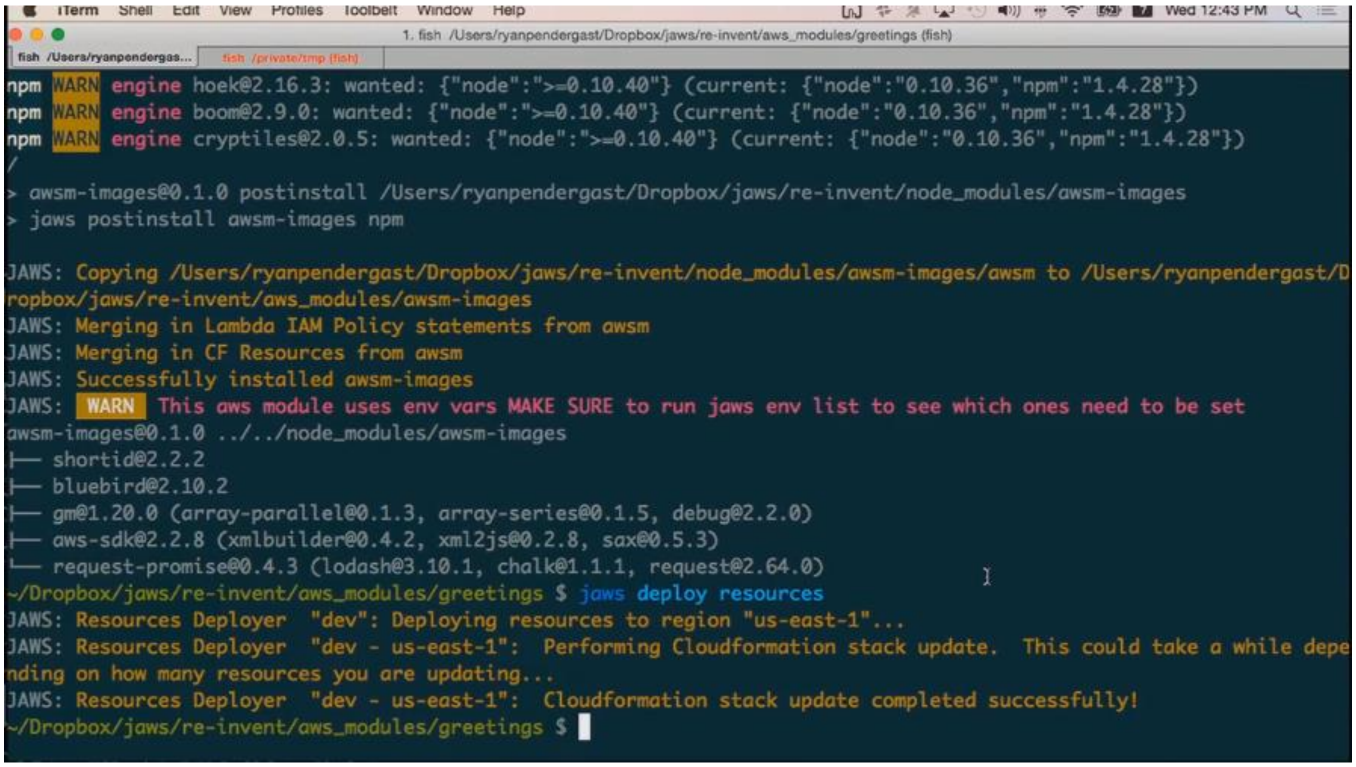
```
        "Description": "",
        "Handler": "aws_modules/greetings/hello/handler.handler",
        "MemorySize": 1024,
        "Runtime": "nodejs",
        "Timeout": 6
      }
    },
    "apiGateway": {
      "deploy": false,
      "cloudFormation": {
        "Type": "AWS",
        "Path": "greetings/hello",
        "Method": "GET",
        "AuthorizationType": "none",
        "ApiKeyRequired": false,
        "RequestTemplates": {},
        "RequestParameters": {},
        "Responses": {
          "400": {
            "statusCode": "400"
          },
          "default": {
            "statusCode": "200",
            "responseParameters": {},
            "responseModels": {},
            "responseTemplates": {
              "application/json": ""
            }
          }
        }
      }
```

It also includes the API Gateway metadata object shown above


iTerm Shell Edit View Profiles Toolbelt Window Help
Wed 12:42 PM
1. fish /Users/ryanpendergast/Dropbox/jaws/re-invent/aws_modules/greetings (fish)
fish /Users/ryanpendergas...
fish /private/tmp (fish)

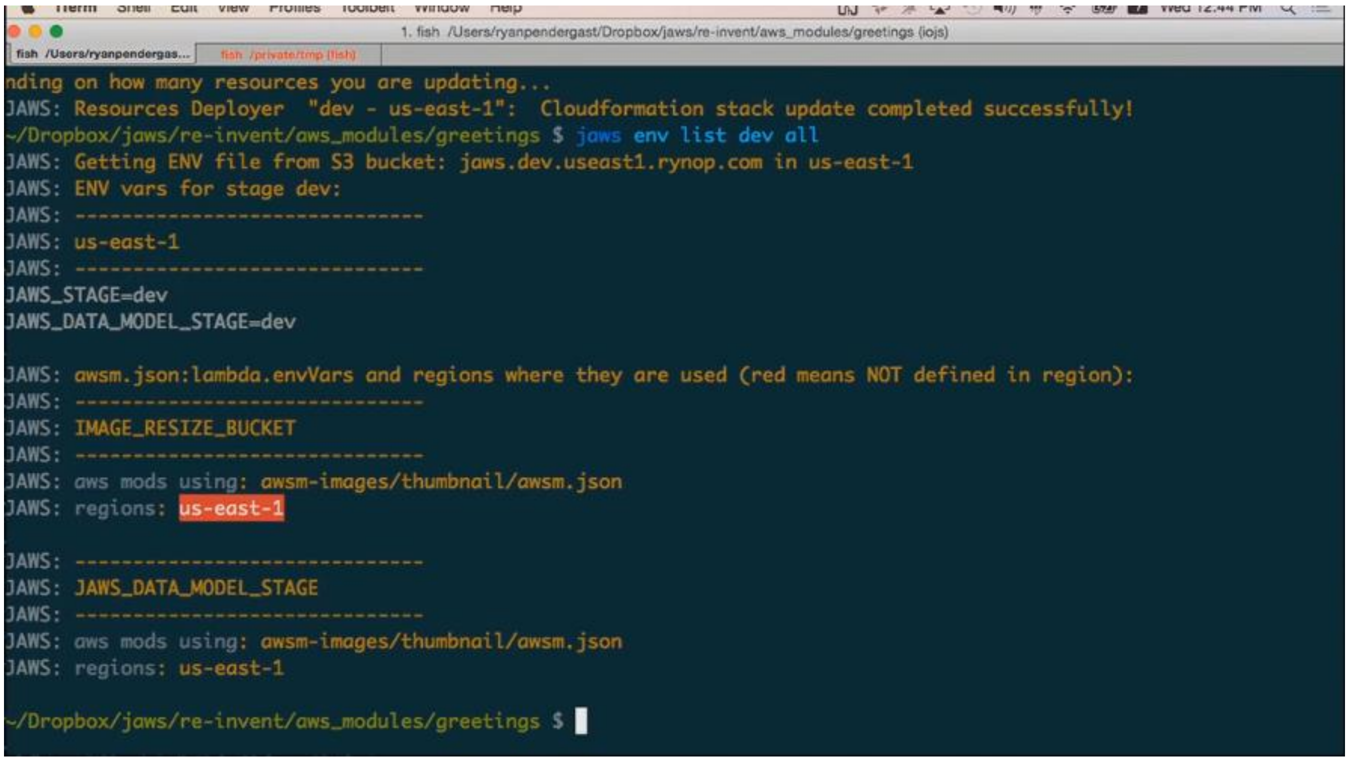
```
~/Dropbox/jaws/re-invent $ jaws module create greetings hello
JAWS: Successfully created greetings/hello
~/Dropbox/jaws/re-invent $ cd aws_modules/greetings/
~/Dropbox/jaws/re-invent/aws_modules/greetings $ npm install awsm-images --save
npm WARN engine hoek@2.16.3: wanted: {"node":">=0.10.40"} (current: {"node":"0.10.36","npm":"1.4.28"})
npm WARN engine boom@2.9.0: wanted: {"node":">=0.10.40"} (current: {"node":"0.10.36","npm":"1.4.28"})
npm WARN engine cryptiles@2.0.5: wanted: {"node":">=0.10.40"} (current: {"node":"0.10.36","npm":"1.4.28"})
/
> awsm-images@0.1.0 postinstall /Users/ryanpendergast/Dropbox/jaws/re-invent/node_modules/awsm-images
> jaws postinstall awsm-images npm

JAWS: Copying /Users/ryanpendergast/Dropbox/jaws/re-invent/node_modules/awsm-images/awsm to /Users/ryanpendergast/D
ropbox/jaws/re-invent/aws_modules/awsm-images
JAWS: Merging in Lambda IAM Policy statements from awsm
JAWS: Merging in CF Resources from awsm
JAWS: Successfully installed awsm-images
JAWS: WARN This aws module uses env vars MAKE SURE to run jaws env list to see which ones need to be set
awsm-images@0.1.0 ../../node_modules/awsm-images
├── shortid@2.2.2
├── bluebird@2.10.2
├── gm@1.20.0 (array-parallel@0.1.3, array-series@0.1.5, debug@2.2.0)
├── aws-sdk@2.2.8 (xmlbuilder@0.4.2, xml2js@0.2.8, sax@0.5.3)
└── request-promise@0.4.3 (lodash@3.10.1, chalk@1.1.1, request@2.64.0)
~/Dropbox/jaws/re-invent/aws_modules/greetings $
```

We now install a separate ***aswm NPM module*** called image-resizing as above. A post-install hook is used to install JAWS hooks into the aswm module code.



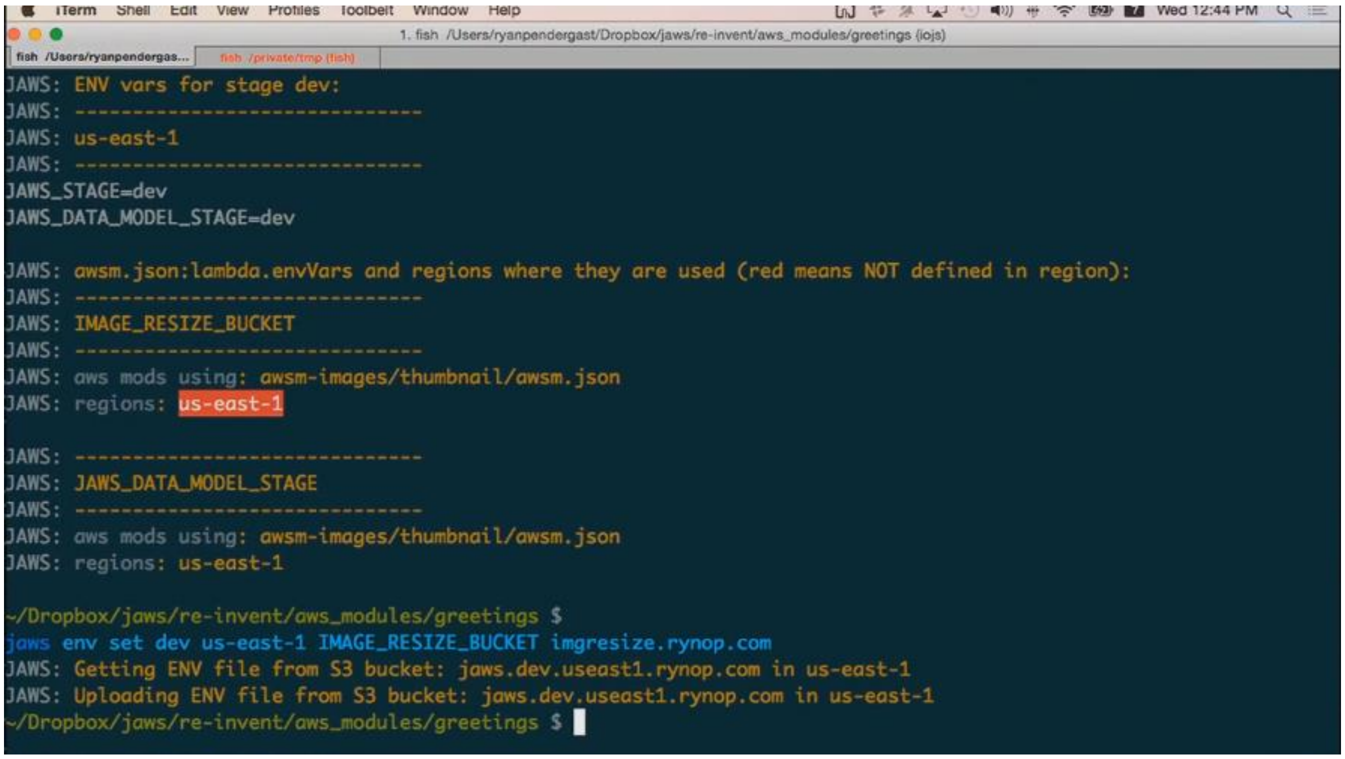
```
npm WARN engine hoek@2.16.3: wanted: {"node":">=0.10.40"} (current: {"node":"0.10.36","npm":"1.4.28"})
npm WARN engine boom@2.9.0: wanted: {"node":">=0.10.40"} (current: {"node":"0.10.36","npm":"1.4.28"})
npm WARN engine cryptiles@2.0.5: wanted: {"node":">=0.10.40"} (current: {"node":"0.10.36","npm":"1.4.28"})
/
> awsm-images@0.1.0 postinstall /Users/ryanpendergast/Dropbox/jaws/re-invent/node_modules/awsm-images
> jaws postinstall awsm-images npm

JAWS: Copying /Users/ryanpendergast/Dropbox/jaws/re-invent/node_modules/awsm-images/awsm to /Users/ryanpendergast/D
ropbox/jaws/re-invent/aws_modules/awsm-images
JAWS: Merging in Lambda IAM Policy statements from awsm
JAWS: Merging in CF Resources from awsm
JAWS: Successfully installed awsm-images
JAWS:  WARN  This aws module uses env vars MAKE SURE to run jaws env list to see which ones need to be set
awsm-images@0.1.0 ../../node_modules/awsm-images
├── shortid@2.2.2
├── bluebird@2.10.2
├── gm@1.20.0 (array-parallel@0.1.3, array-series@0.1.5, debug@2.2.0)
├── aws-sdk@2.2.8 (xmlbuilder@0.4.2, xml2js@0.2.8, sax@0.5.3)
└── request-promise@0.4.3 (lodash@3.10.1, chalk@1.1.1, request@2.64.0)
~/Dropbox/jaws/re-invent/aws_modules/greetings $ jaws deploy resources
JAWS: Resources Deployer  "dev": Deploying resources to region "us-east-1"...
JAWS: Resources Deployer  "dev - us-east-1":  Performing Cloudformation stack update.  This could take a while depe
nding on how many resources you are updating...
JAWS: Resources Deployer  "dev - us-east-1":  Cloudformation stack update completed successfully!
~/Dropbox/jaws/re-invent/aws_modules/greetings $
```

Using the ***$ jaws deploy*** resources to provision your resources within the specified stage, this will now execute your CF resource file



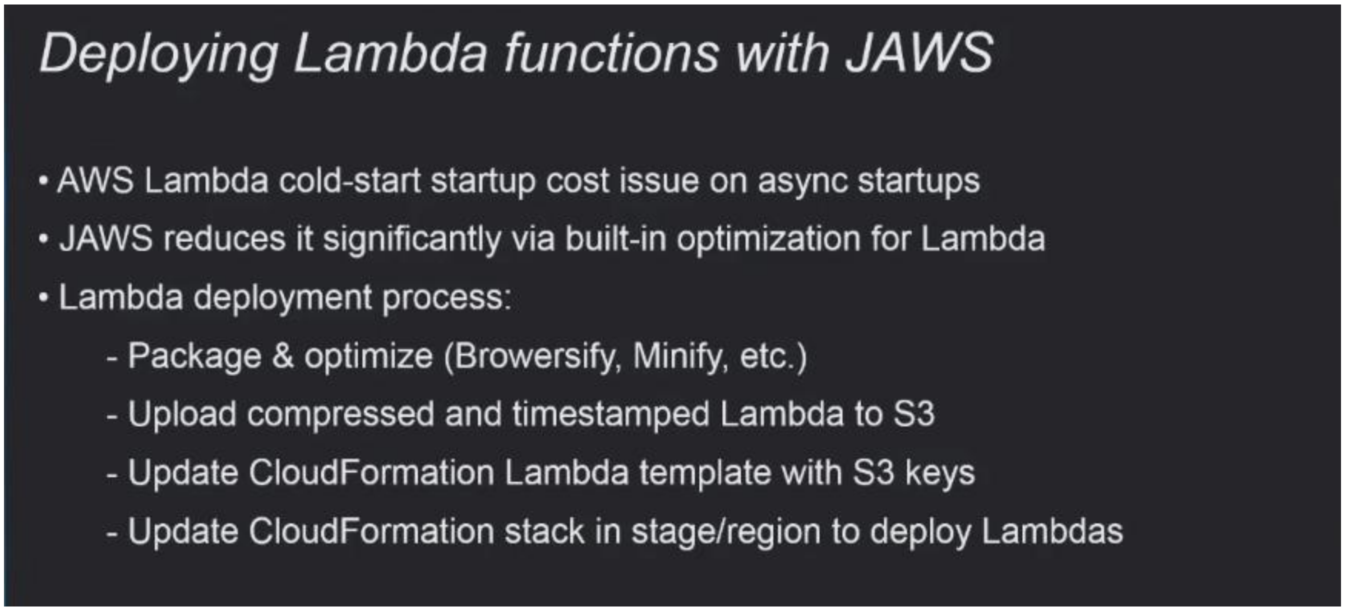
```
nding on how many resources you are updating...
JAWS: Resources Deployer  "dev - us-east-1":  Cloudformation stack update completed successfully!
~/Dropbox/jaws/re-invent/aws_modules/greetings $ jaws env list dev all
JAWS: Getting ENV file from S3 bucket: jaws.dev.useast1.rynop.com in us-east-1
JAWS: ENV vars for stage dev:
JAWS: ------------------------------
JAWS: us-east-1
JAWS: ------------------------------
JAWS_STAGE=dev
JAWS_DATA_MODEL_STAGE=dev

JAWS: awsm.json:lambda.envVars and regions where they are used (red means NOT defined in region):
JAWS: ------------------------------
JAWS: IMAGE_RESIZE_BUCKET
JAWS: ------------------------------
JAWS: aws mods using: awsm-images/thumbnail/awsm.json
JAWS: regions: us-east-1

JAWS: ------------------------------
JAWS: JAWS_DATA_MODEL_STAGE
JAWS: ------------------------------
JAWS: aws mods using: awsm-images/thumbnail/awsm.json
JAWS: regions: us-east-1

~/Dropbox/jaws/re-invent/aws_modules/greetings $
```

The ***$ jaws env list dev all*** command will list all our Dev environment variables for this stage across all regions. The red color highlight is reminding us to set the IMAGE_RESIZE_BUCKET variable, we will do this below


iTerm Shell Edit View Profiles Toolbelt Window Help
Wed 12:44 PM
1. fish /Users/ryanpendergast/Dropbox/jaws/re-invent/aws_modules/greetings (iojs)
fish /Users/ryanpendergas...
fish /private/tmp (fish)

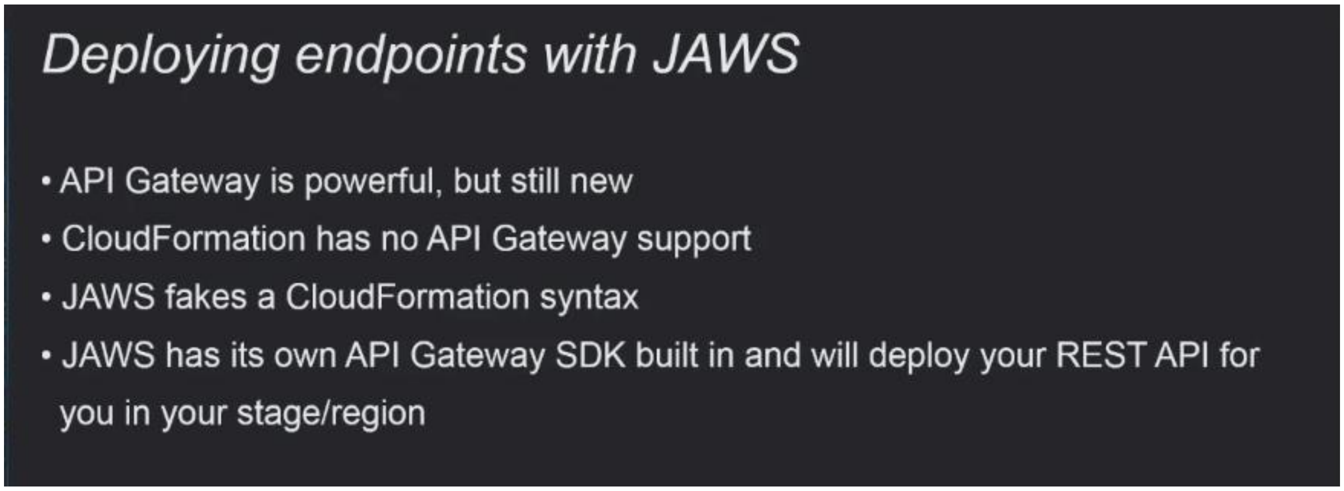
```
JAWS: ENV vars for stage dev:
JAWS: ------------------------------
JAWS: us-east-1
JAWS: ------------------------------
JAWS_STAGE=dev
JAWS_DATA_MODEL_STAGE=dev

JAWS: awsm.json:lambda.envVars and regions where they are used (red means NOT defined in region):
JAWS: ------------------------------
JAWS: IMAGE_RESIZE_BUCKET
JAWS: ------------------------------
JAWS: aws mods using: awsm-images/thumbnail/awsm.json
JAWS: regions: us-east-1

JAWS: ------------------------------
JAWS: JAWS_DATA_MODEL_STAGE
JAWS: ------------------------------
JAWS: aws mods using: awsm-images/thumbnail/awsm.json
JAWS: regions: us-east-1

~/Dropbox/jaws/re-invent/aws_modules/greetings $
jaws env set dev us-east-1 IMAGE_RESIZE_BUCKET imgresize.rynop.com
JAWS: Getting ENV file from S3 bucket: jaws.dev.useast1.rynop.com in us-east-1
JAWS: Uploading ENV file from S3 bucket: jaws.dev.useast1.rynop.com in us-east-1
~/Dropbox/jaws/re-invent/aws_modules/greetings $
```

We set the IMAGE_RESIZE_BUCKET variable for that region and stage using the command above



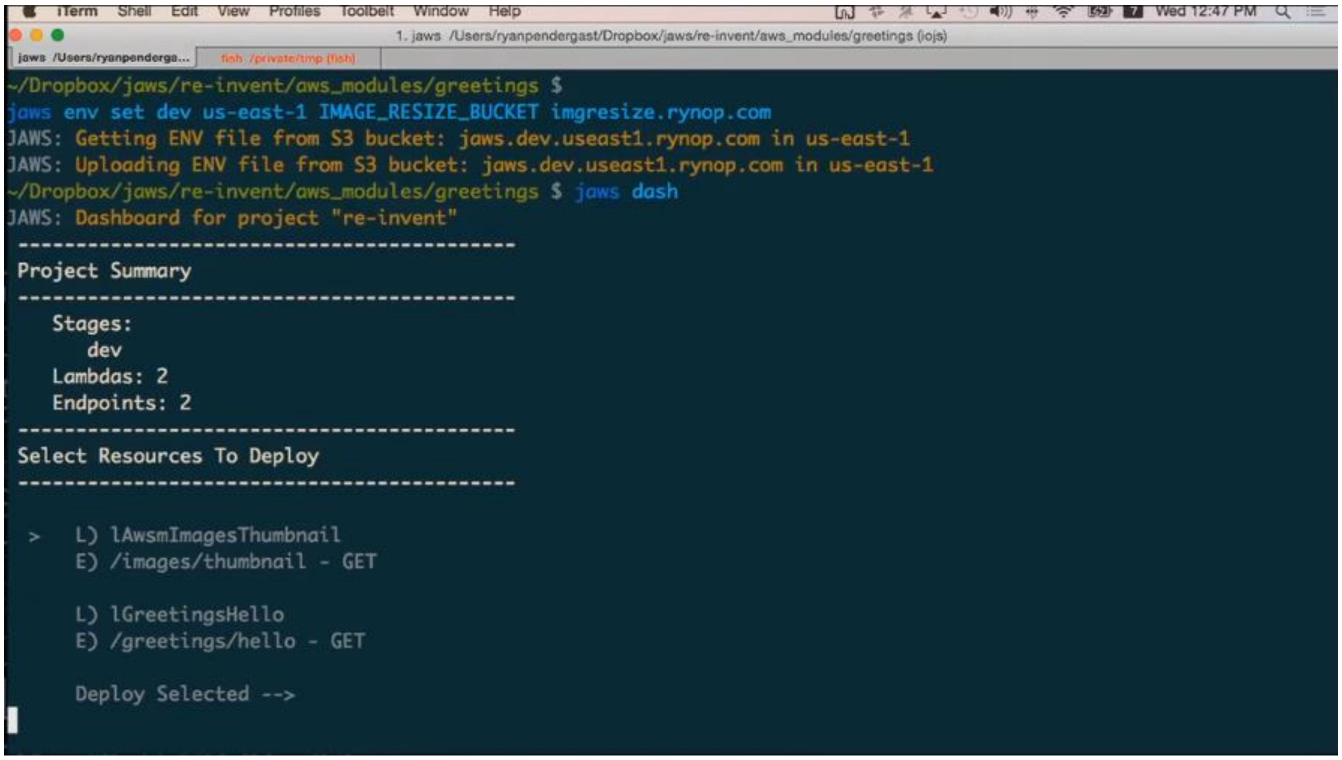
Deploying Lambda functions with JAWS

- AWS Lambda cold-start startup cost issue on async startups
- JAWS reduces it significantly via built-in optimization for Lambda
- Lambda deployment process:
  - Package & optimize (Browersify, Minify, etc.)
  - Upload compressed and timestamped Lambda to S3
  - Update CloudFormation Lambda template with S3 keys
  - Update CloudFormation stack in stage/region to deploy Lambdas



Deploying endpoints with JAWS

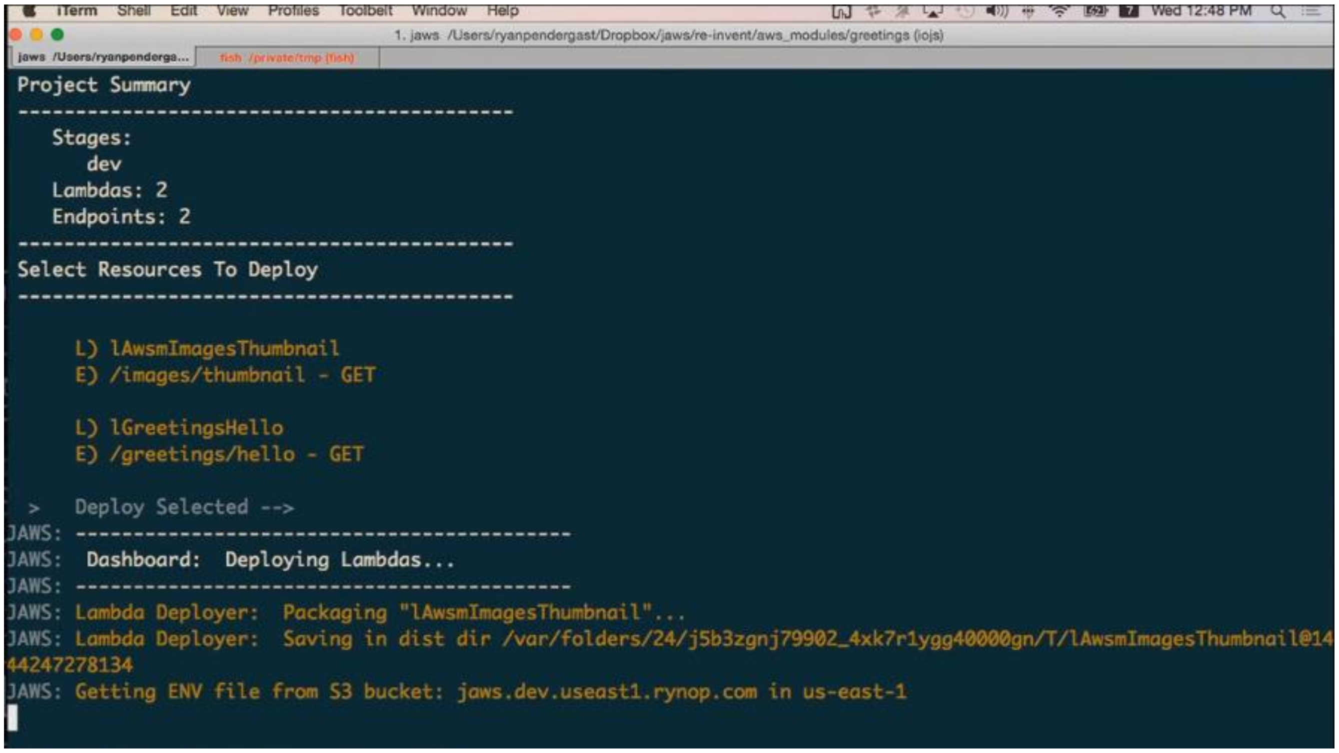
- API Gateway is powerful, but still new
- CloudFormation has no API Gateway support
- JAWS fakes a CloudFormation syntax
- JAWS has its own API Gateway SDK built in and will deploy your REST API for you in your stage/region



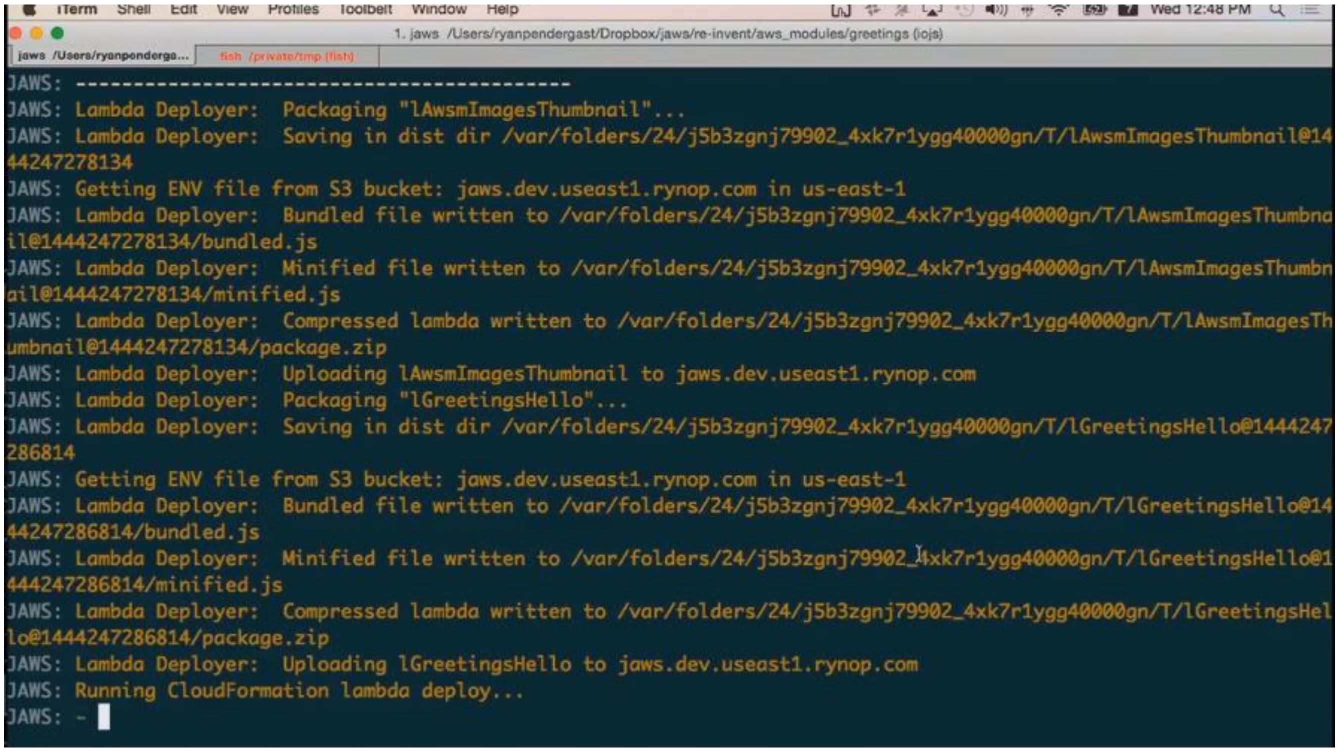
iTerm Shell Edit View Profiles Toolbelt Window Help
Wed 12:47 PM
1. jaws /Users/ryanpendergast/Dropbox/jaws/re-invent/aws_modules/greetings (iojs)

```
~/Dropbox/jaws/re-invent/aws_modules/greetings $
jaws env set dev us-east-1 IMAGE_RESIZE_BUCKET imgresize.rynop.com
JAWS: Getting ENV file from S3 bucket: jaws.dev.useast1.rynop.com in us-east-1
JAWS: Uploading ENV file from S3 bucket: jaws.dev.useast1.rynop.com in us-east-1
~/Dropbox/jaws/re-invent/aws_modules/greetings $ jaws dash
JAWS: Dashboard for project "re-invent"
 -------------------------------------------
 Project Summary
 -------------------------------------------
    Stages:
       dev
    Lambdas: 2
    Endpoints: 2
 -------------------------------------------
 Select Resources To Deploy
 -------------------------------------------

  >   L) lAwsmImagesThumbnail
      E) /images/thumbnail - GET

      L) lGreetingsHello
      E) /greetings/hello - GET

      Deploy Selected -->
```

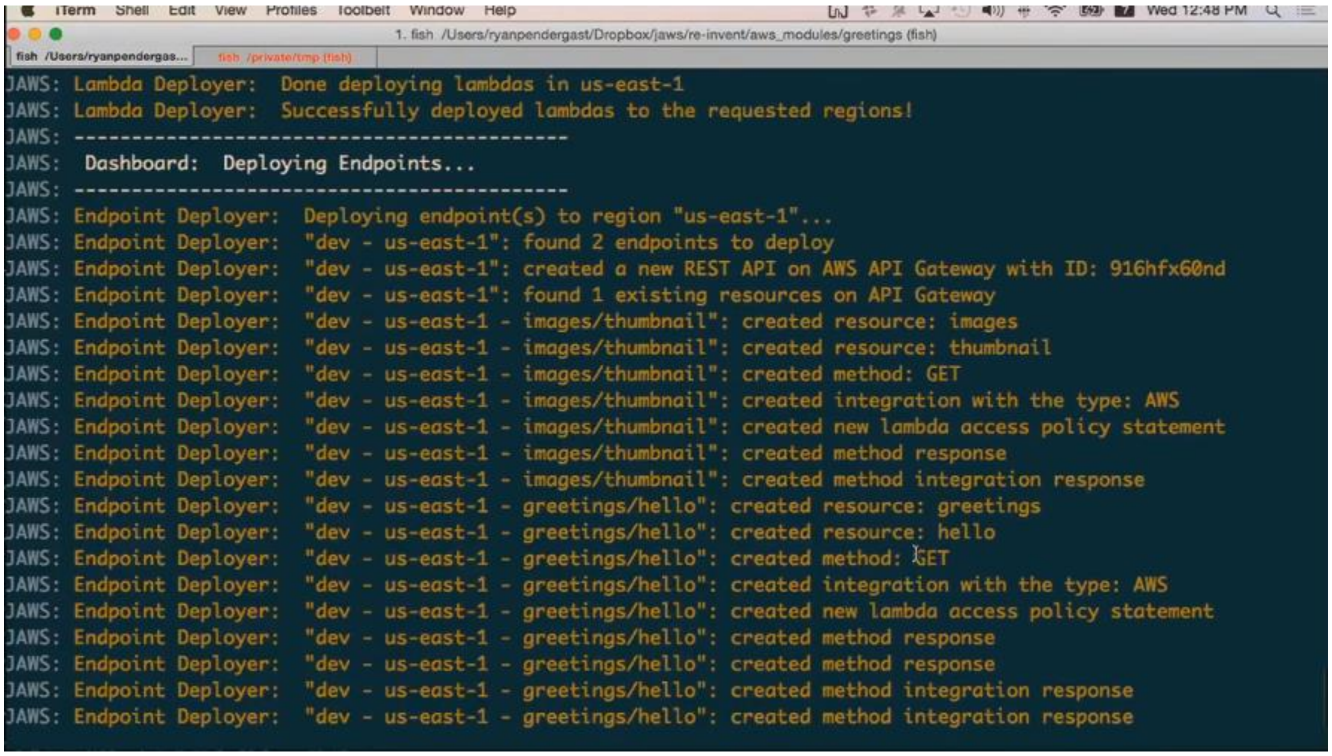
The ***$ jaws dash*** command will display all your Lambdas and the L and API Gateways as the E. we are going to deploy all


iTerm Shell Edit View Profiles Toolbelt Window Help
Wed 12:48 PM
1. jaws /Users/ryanpendergast/Dropbox/jaws/re-invent/aws_modules/greetings (iojs)
jaws /Users/ryanpenderga...
fish /private/tmp (fish)

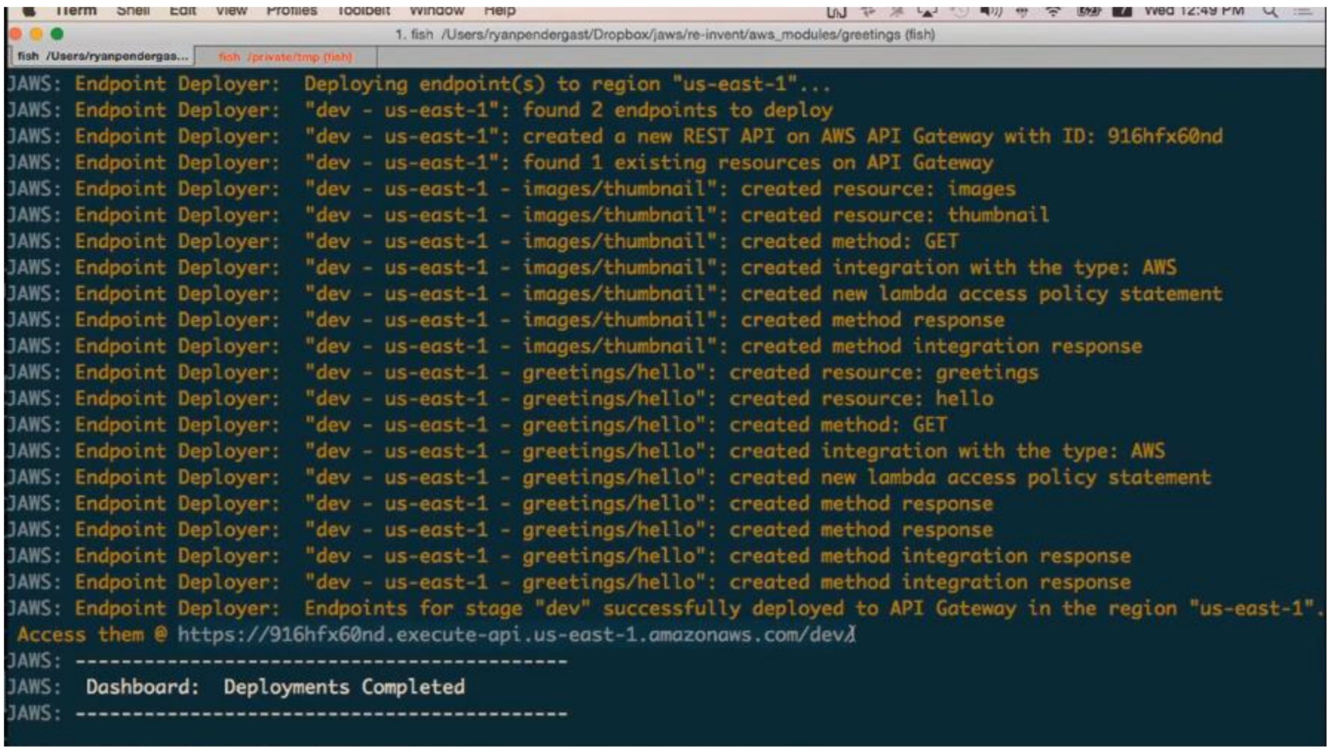
```
Project Summary
-------------------------------------------
   Stages:
      dev
   Lambdas: 2
   Endpoints: 2
-------------------------------------------
Select Resources To Deploy
-------------------------------------------

     L) lAwsmImagesThumbnail
     E) /images/thumbnail - GET

     L) lGreetingsHello
     E) /greetings/hello - GET

  >  Deploy Selected -->
JAWS: -------------------------------------------
JAWS:  Dashboard:  Deploying Lambdas...
JAWS: -------------------------------------------
JAWS: Lambda Deployer:  Packaging "lAwsmImagesThumbnail"...
JAWS: Lambda Deployer:  Saving in dist dir /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lAwsmImagesThumbnail@14
44247278134
JAWS: Getting ENV file from S3 bucket: jaws.dev.useast1.rynop.com in us-east-1
```


iTerm Shell Edit View Profiles Toolbelt Window Help
Wed 12:48 PM
1. jaws /Users/ryanpendergast/Dropbox/jaws/re-invent/aws_modules/greetings (iojs)
jaws /Users/ryanpenderga...
fish /private/tmp (fish)

```
JAWS: -------------------------------------------
JAWS: Lambda Deployer:  Packaging "lAwsmImagesThumbnail"...
JAWS: Lambda Deployer:  Saving in dist dir /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lAwsmImagesThumbnail@14
44247278134
JAWS: Getting ENV file from S3 bucket: jaws.dev.useast1.rynop.com in us-east-1
JAWS: Lambda Deployer:  Bundled file written to /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lAwsmImagesThumbna
il@1444247278134/bundled.js
JAWS: Lambda Deployer:  Minified file written to /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lAwsmImagesThumbn
ail@1444247278134/minified.js
JAWS: Lambda Deployer:  Compressed lambda written to /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lAwsmImagesTh
umbnail@1444247278134/package.zip
JAWS: Lambda Deployer:  Uploading lAwsmImagesThumbnail to jaws.dev.useast1.rynop.com
JAWS: Lambda Deployer:  Packaging "lGreetingsHello"...
JAWS: Lambda Deployer:  Saving in dist dir /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lGreetingsHello@1444247
286814
JAWS: Getting ENV file from S3 bucket: jaws.dev.useast1.rynop.com in us-east-1
JAWS: Lambda Deployer:  Bundled file written to /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lGreetingsHello@14
44247286814/bundled.js
JAWS: Lambda Deployer:  Minified file written to /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lGreetingsHello@1
444247286814/minified.js
JAWS: Lambda Deployer:  Compressed lambda written to /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lGreetingsHel
lo@1444247286814/package.zip
JAWS: Lambda Deployer:  Uploading lGreetingsHello to jaws.dev.useast1.rynop.com
JAWS: Running CloudFormation lambda deploy...
JAWS: -
```

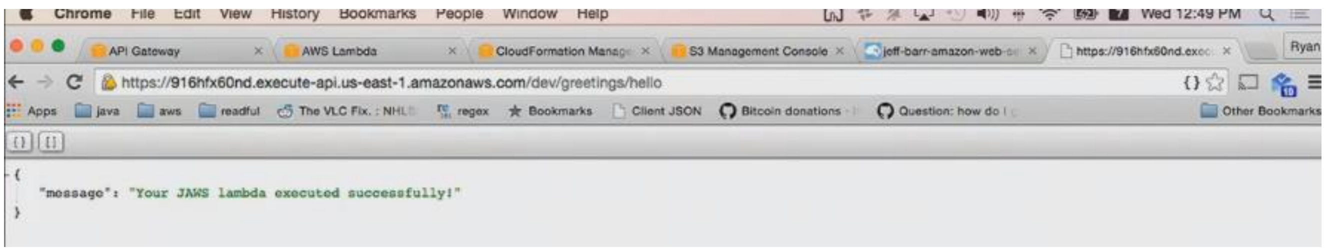


```
JAWS: Lambda Deployer:  Done deploying lambdas in us-east-1
JAWS: Lambda Deployer:  Successfully deployed lambdas to the requested regions!
JAWS: -------------------------------------------
JAWS:  Dashboard:  Deploying Endpoints...
JAWS: -------------------------------------------
JAWS: Endpoint Deployer:  Deploying endpoint(s) to region "us-east-1"...
JAWS: Endpoint Deployer:  "dev - us-east-1": found 2 endpoints to deploy
JAWS: Endpoint Deployer:  "dev - us-east-1": created a new REST API on AWS API Gateway with ID: 916hfx60nd
JAWS: Endpoint Deployer:  "dev - us-east-1": found 1 existing resources on API Gateway
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created resource: images
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created resource: thumbnail
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created method: GET
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created integration with the type: AWS
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created new lambda access policy statement
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created method response
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created method integration response
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created resource: greetings
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created resource: hello
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created method: GET
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created integration with the type: AWS
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created new lambda access policy statement
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created method response
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created method response
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created method integration response
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created method integration response
```



```
JAWS: Endpoint Deployer:  Deploying endpoint(s) to region "us-east-1"...
JAWS: Endpoint Deployer:  "dev - us-east-1": found 2 endpoints to deploy
JAWS: Endpoint Deployer:  "dev - us-east-1": created a new REST API on AWS API Gateway with ID: 916hfx60nd
JAWS: Endpoint Deployer:  "dev - us-east-1": found 1 existing resources on API Gateway
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created resource: images
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created resource: thumbnail
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created method: GET
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created integration with the type: AWS
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created new lambda access policy statement
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created method response
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created method integration response
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created resource: greetings
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created resource: hello
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created method: GET
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created integration with the type: AWS
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created new lambda access policy statement
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created method response
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created method response
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created method integration response
JAWS: Endpoint Deployer:  "dev - us-east-1 - greetings/hello": created method integration response
JAWS: Endpoint Deployer:  Endpoints for stage "dev" successfully deployed to API Gateway in the region "us-east-1".
 Access them @ https://916hfx60nd.execute-api.us-east-1.amazonaws.com/dev/
JAWS: -------------------------------------------
JAWS:  Dashboard:  Deployments Completed
JAWS: -------------------------------------------
```

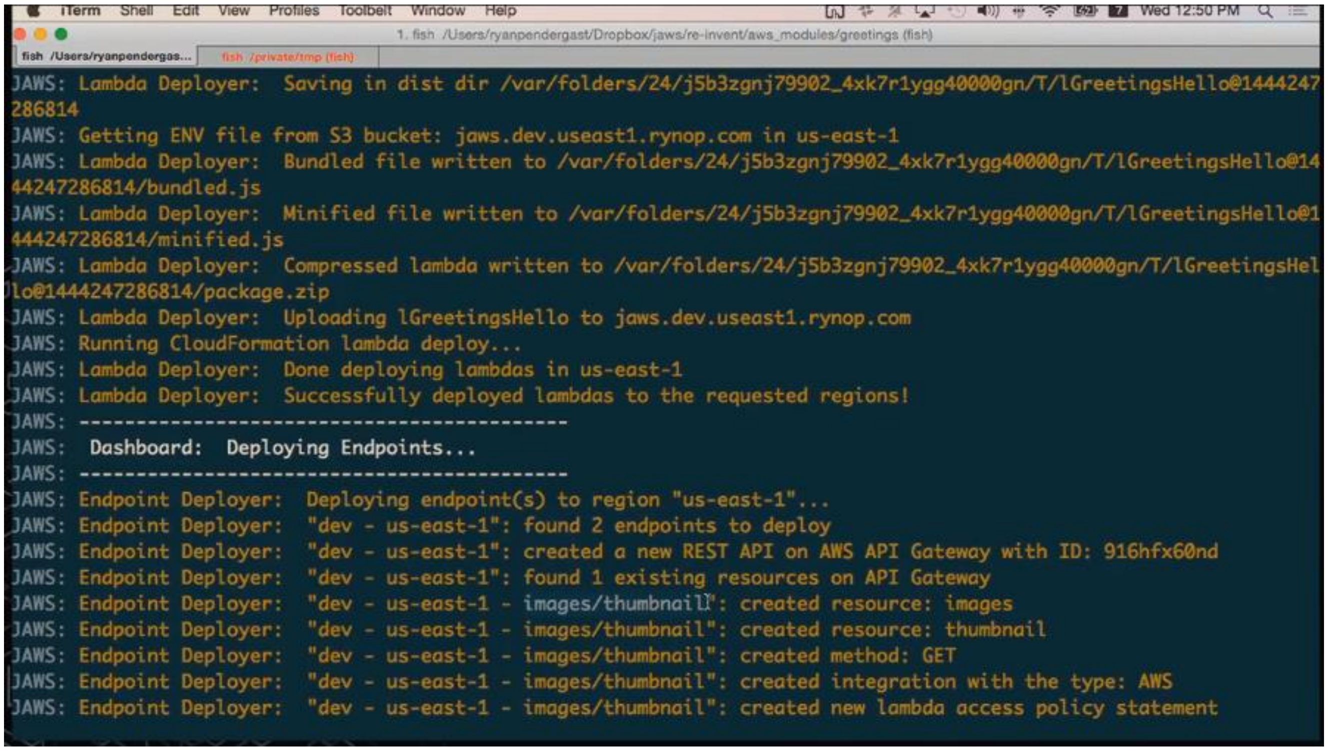
We then get back the URL for the REST API created for us



https://916hfx60nd.execute-api.us-east-1.amazonaws.com/dev/greetings/hello

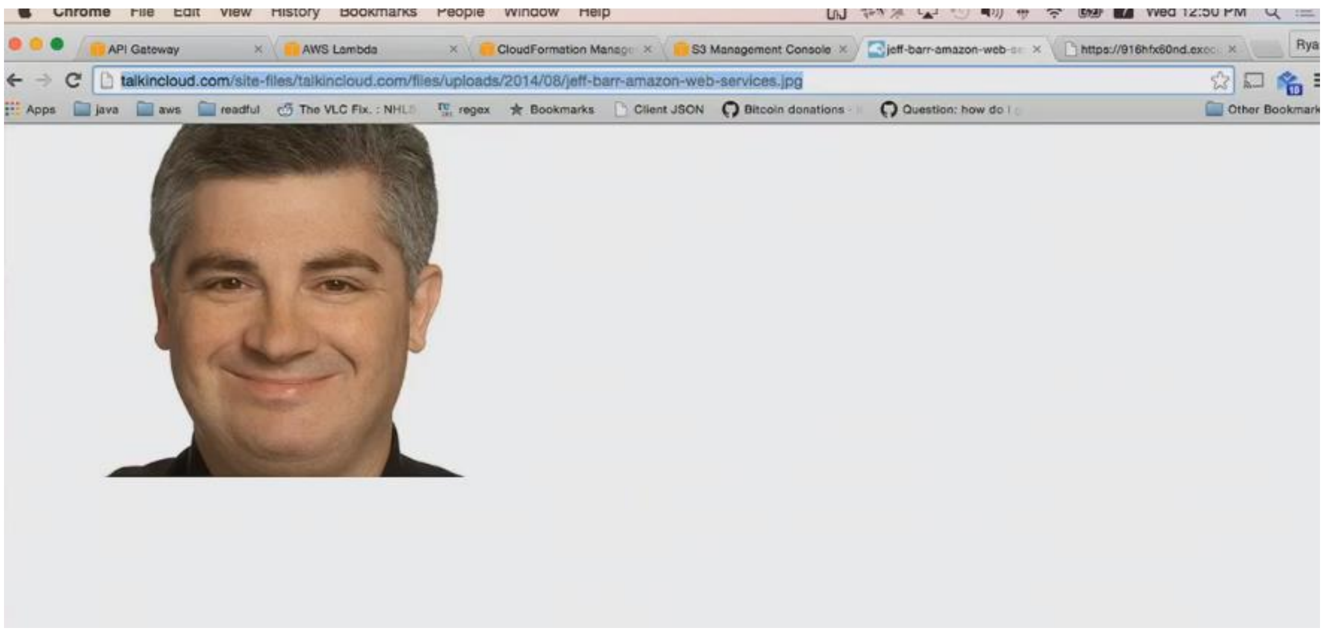
```
{
  "message": "Your JAWS lambda executed successfully!"
}
```

This shows that our API endpoint now works successfully using JAWS

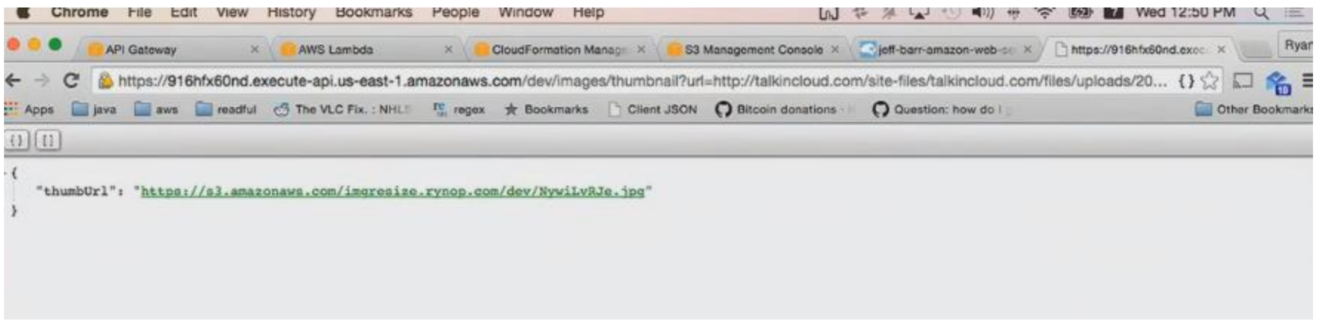


```
JAWS: Lambda Deployer:  Saving in dist dir /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lGreetingsHello@1444247
286814
JAWS: Getting ENV file from S3 bucket: jaws.dev.useast1.rynop.com in us-east-1
JAWS: Lambda Deployer:  Bundled file written to /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lGreetingsHello@14
44247286814/bundled.js
JAWS: Lambda Deployer:  Minified file written to /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lGreetingsHello@1
444247286814/minified.js
JAWS: Lambda Deployer:  Compressed lambda written to /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lGreetingsHel
lo@1444247286814/package.zip
JAWS: Lambda Deployer:  Uploading lGreetingsHello to jaws.dev.useast1.rynop.com
JAWS: Running CloudFormation lambda deploy...
JAWS: Lambda Deployer:  Done deploying lambdas in us-east-1
JAWS: Lambda Deployer:  Successfully deployed lambdas to the requested regions!
JAWS: -----------------------------------------
JAWS:  Dashboard:  Deploying Endpoints...
JAWS: -----------------------------------------
JAWS: Endpoint Deployer:  Deploying endpoint(s) to region "us-east-1"...
JAWS: Endpoint Deployer:  "dev - us-east-1": found 2 endpoints to deploy
JAWS: Endpoint Deployer:  "dev - us-east-1": created a new REST API on AWS API Gateway with ID: 916hfx60nd
JAWS: Endpoint Deployer:  "dev - us-east-1": found 1 existing resources on API Gateway
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created resource: images
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created resource: thumbnail
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created method: GET
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created integration with the type: AWS
JAWS: Endpoint Deployer:  "dev - us-east-1 - images/thumbnail": created new lambda access policy statement
```

We can also see image resizing lambda in action, we need to pass the URL of an image we want to resize as a querystring parameter as below



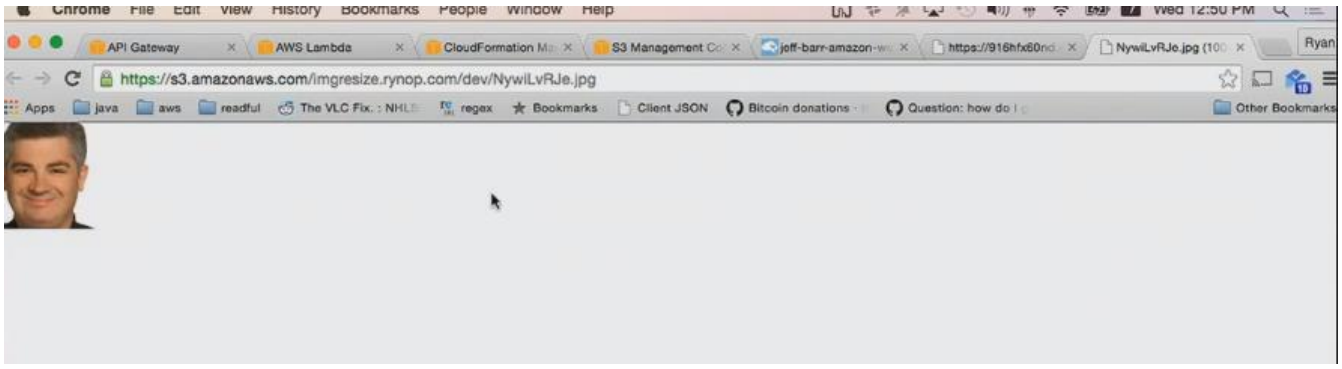
talkincloud.com/site-files/talkincloud.com/files/uploads/2014/08/jeff-barr-amazon-web-services.jpg



https://916hfx60nd.execute-api.us-east-1.amazonaws.com/dev/images/thumbnail?url=http://talkincloud.com/site-files/talkincloud.com/files/uploads/20...

```
{
    "thumbUrl": "https://s3.amazonaws.com/imgresize.rynop.com/dev/NywiLvRJe.jpg"
}
```

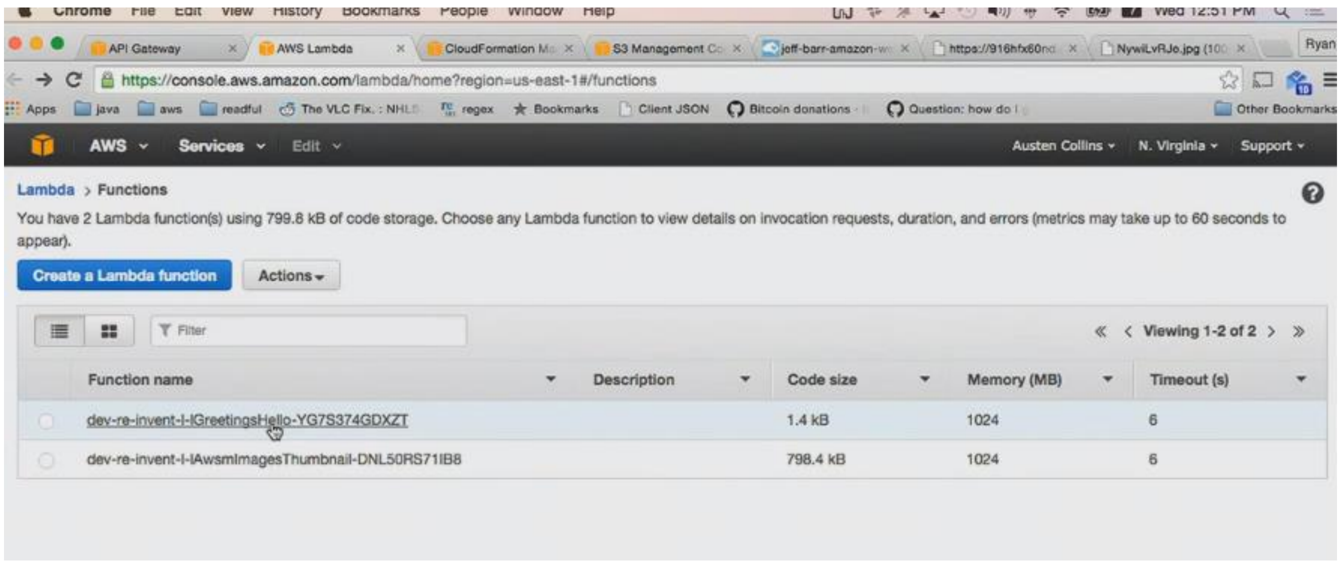
It downloaded the image specified in the ***url*** querystring, then it resized the image and uploaded it to an S3 bucket and returned back the URL for the S3 bucket as above


https://s3.amazonaws.com/imgresize.rynop.com/dev/NywiLvRJe.jpg

We now can see the resized image in S3 above

DEMO

THE EFFECT OF OPTIMIZING

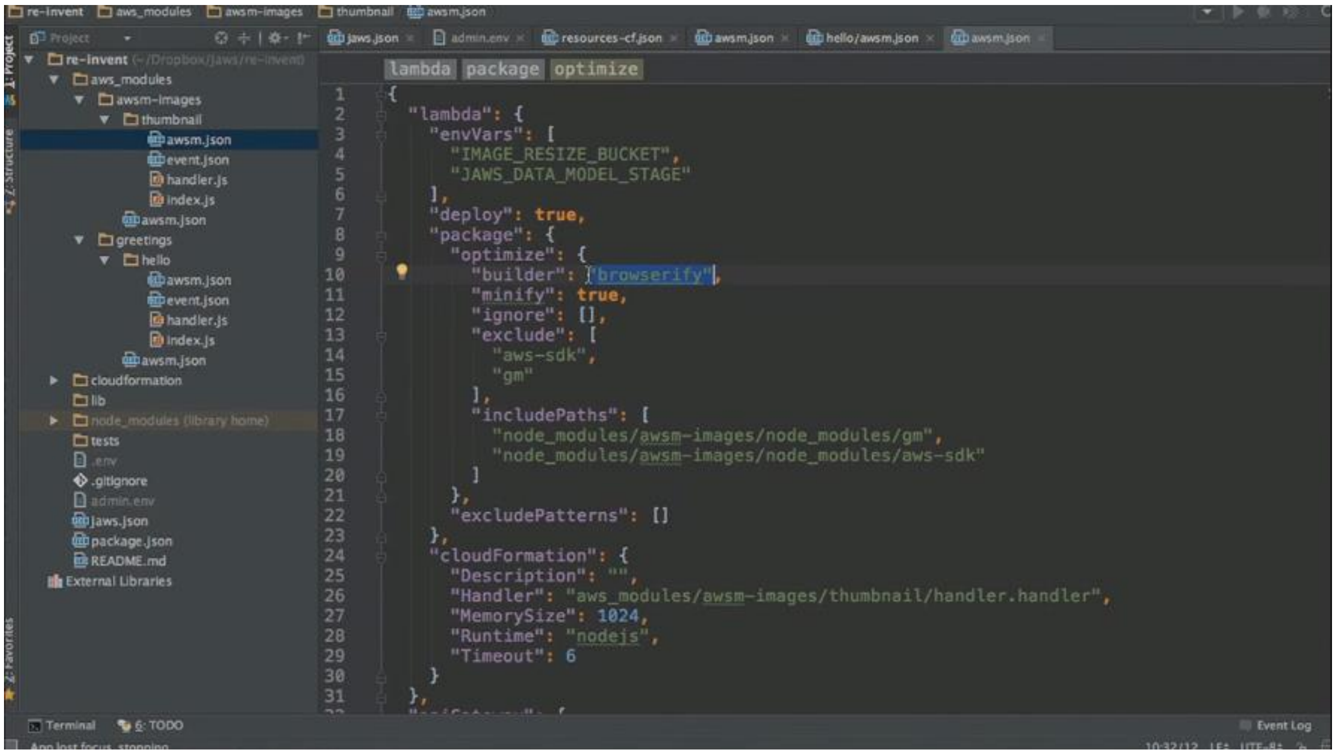


Lambda > Functions

You have 2 Lambda function(s) using 799.8 kB of code storage. Choose any Lambda function to view details on invocation requests, duration, and errors (metrics may take up to 60 seconds to appear).

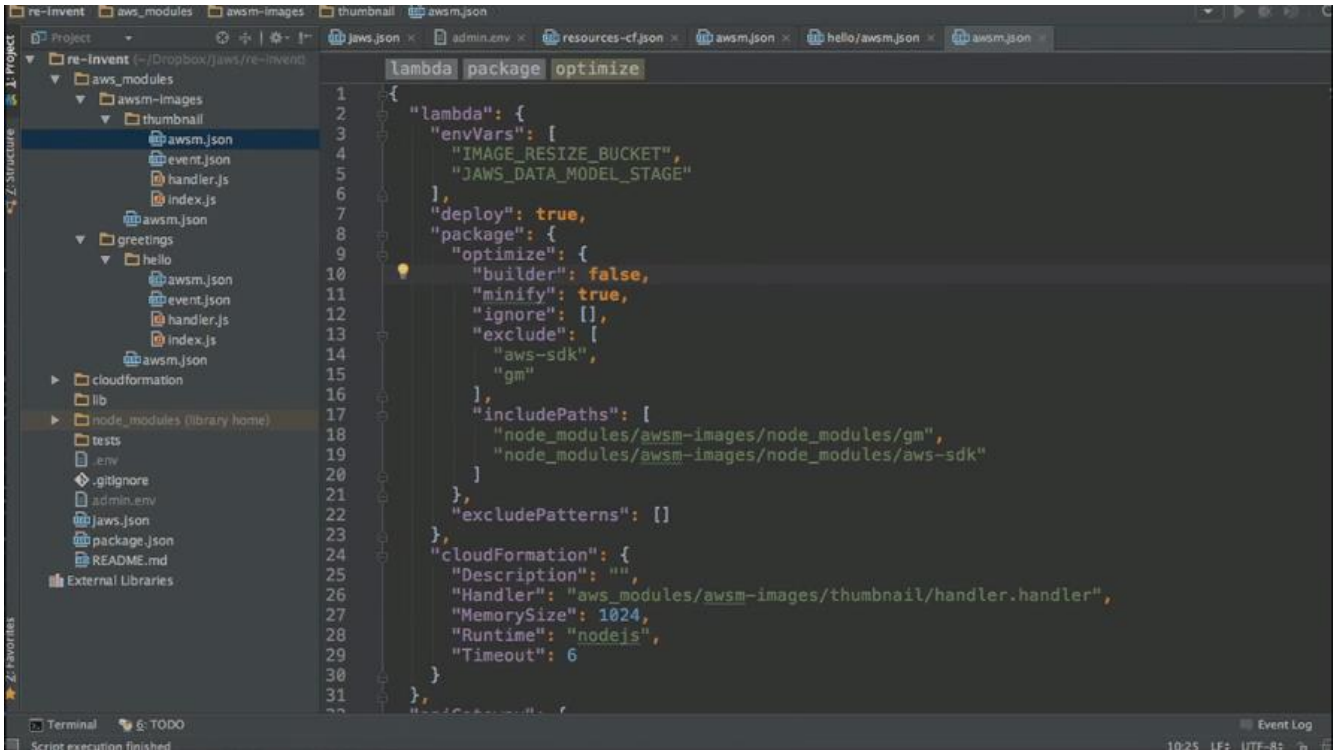
Function name	Description	Code size	Memory (MB)	Timeout (s)
dev-re-invent-l-lGreetingsHello-YG7S374GDXZT		1.4 kB	1024	6
dev-re-invent-l-lAwsmImagesThumbnail-DNL50RS71IB8		798.4 kB	1024	6

The optimized code package created by JAWS is just 798.4kB in size.

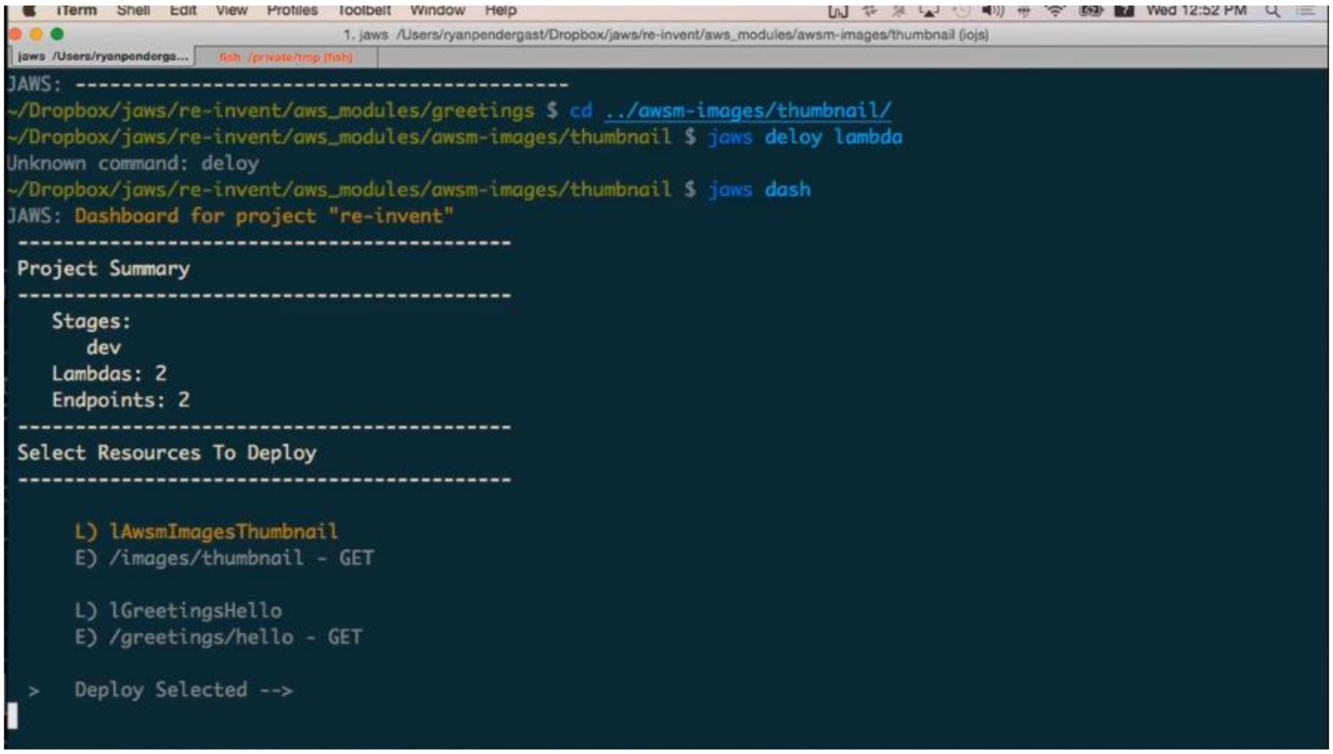


```
{
  "lambda": {
    "envVars": [
      "IMAGE_RESIZE_BUCKET",
      "JAWS_DATA_MODEL_STAGE"
    ],
    "deploy": true,
    "package": {
      "optimize": {
        "builder": "browserify",
        "minify": true,
        "ignore": [],
        "exclude": [
          "aws-sdk",
          "gm"
        ],
        "includePaths": [
          "node_modules/awsm-images/node_modules/gm",
          "node_modules/awsm-images/node_modules/aws-sdk"
        ]
      },
      "excludePatterns": []
    },
    "cloudFormation": {
      "Description": "",
      "Handler": "aws_modules/awsm-images/thumbnail/handler.handler",
      "MemorySize": 1024,
      "Runtime": "nodejs",
      "Timeout": 6
    }
  },
```

The optimizer used for minify and packaging the code into a small size is Browserify, let us remove it and see what the actual un-optimized code size is below

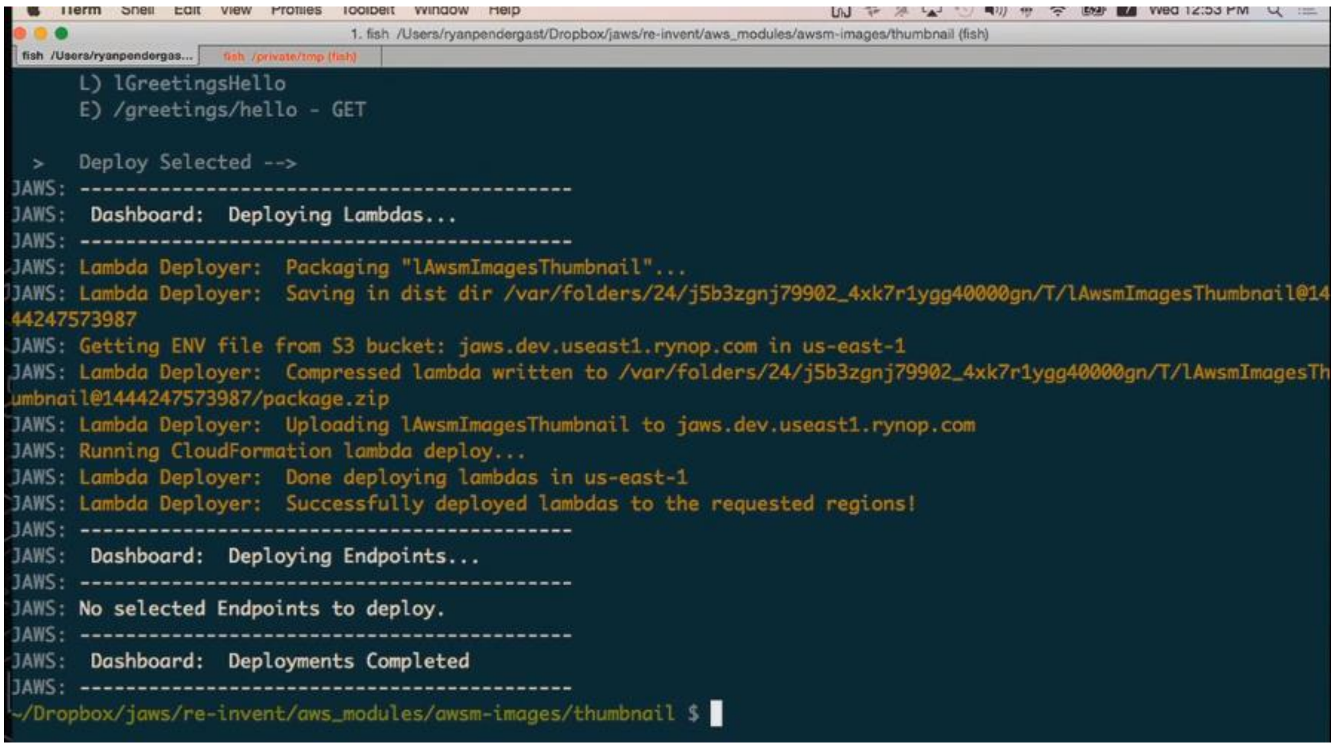


```
{
  "lambda": {
    "envVars": [
      "IMAGE_RESIZE_BUCKET",
      "JAWS_DATA_MODEL_STAGE"
    ],
    "deploy": true,
    "package": {
      "optimize": {
        "builder": false,
        "minify": true,
        "ignore": [],
        "exclude": [
          "aws-sdk",
          "gm"
        ],
        "includePaths": [
          "node_modules/awsm-images/node_modules/gm",
          "node_modules/awsm-images/node_modules/aws-sdk"
        ]
      },
      "excludePatterns": []
    },
    "cloudFormation": {
      "Description": "",
      "Handler": "aws_modules/awsm-images/thumbnail/handler.handler",
      "MemorySize": 1024,
      "Runtime": "nodejs",
      "Timeout": 6
    }
  },
```

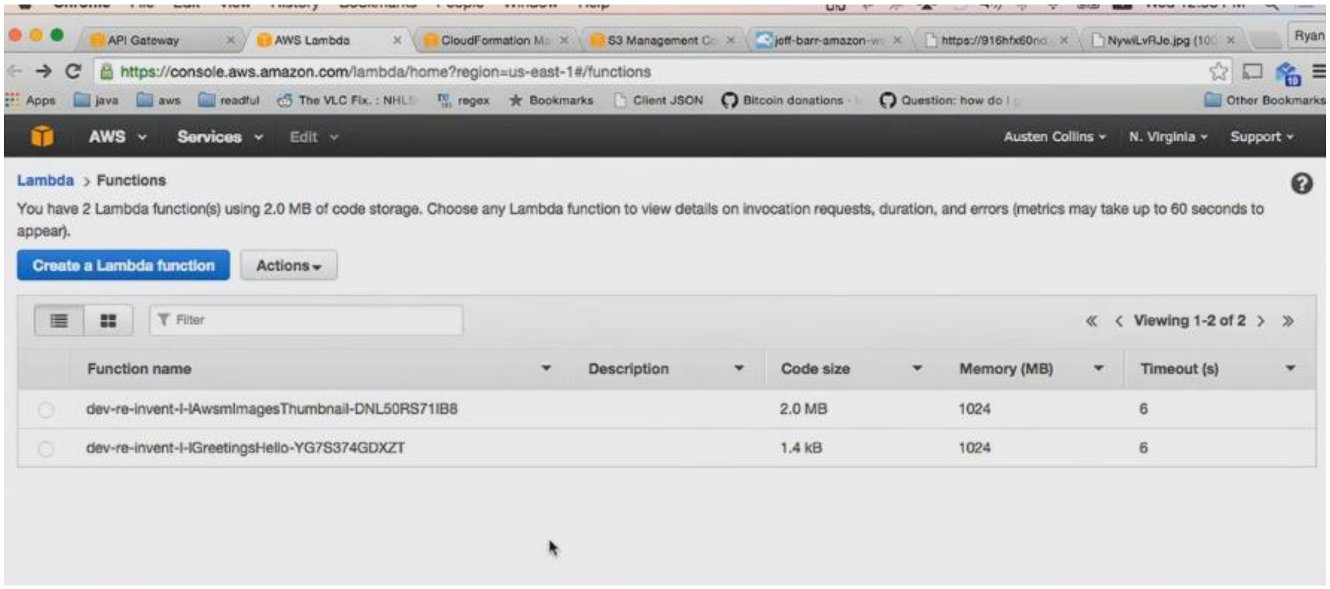

iTerm Shell Edit View Profiles Toolbelt Window Help
Wed 12:52 PM
1. jaws /Users/ryanpendergast/Dropbox/jaws/re-invent/aws_modules/awsm-images/thumbnail (iojs)
jaws /Users/ryanpenderga...
fish /private/tmp (fish)

```
JAWS: -------------------------------------------
~/Dropbox/jaws/re-invent/aws_modules/greetings $ cd ../awsm-images/thumbnail/
~/Dropbox/jaws/re-invent/aws_modules/awsm-images/thumbnail $ jaws deloy lambda
Unknown command: deloy
~/Dropbox/jaws/re-invent/aws_modules/awsm-images/thumbnail $ jaws dash
JAWS: Dashboard for project "re-invent"
 -------------------------------------------
 Project Summary
 -------------------------------------------
    Stages:
       dev
    Lambdas: 2
    Endpoints: 2
 -------------------------------------------
 Select Resources To Deploy
 -------------------------------------------

     L) lAwsmImagesThumbnail
     E) /images/thumbnail - GET

     L) lGreetingsHello
     E) /greetings/hello - GET

  >  Deploy Selected -->
```

We are just going to deploy the lambda that we changed only


iTerm Shell Edit View Profiles Toolbelt Window Help
Wed 12:53 PM
1. fish /Users/ryanpendergast/Dropbox/jaws/re-invent/aws_modules/awsm-images/thumbnail (fish)
fish /Users/ryanpendergas...
fish /private/tmp (fish)

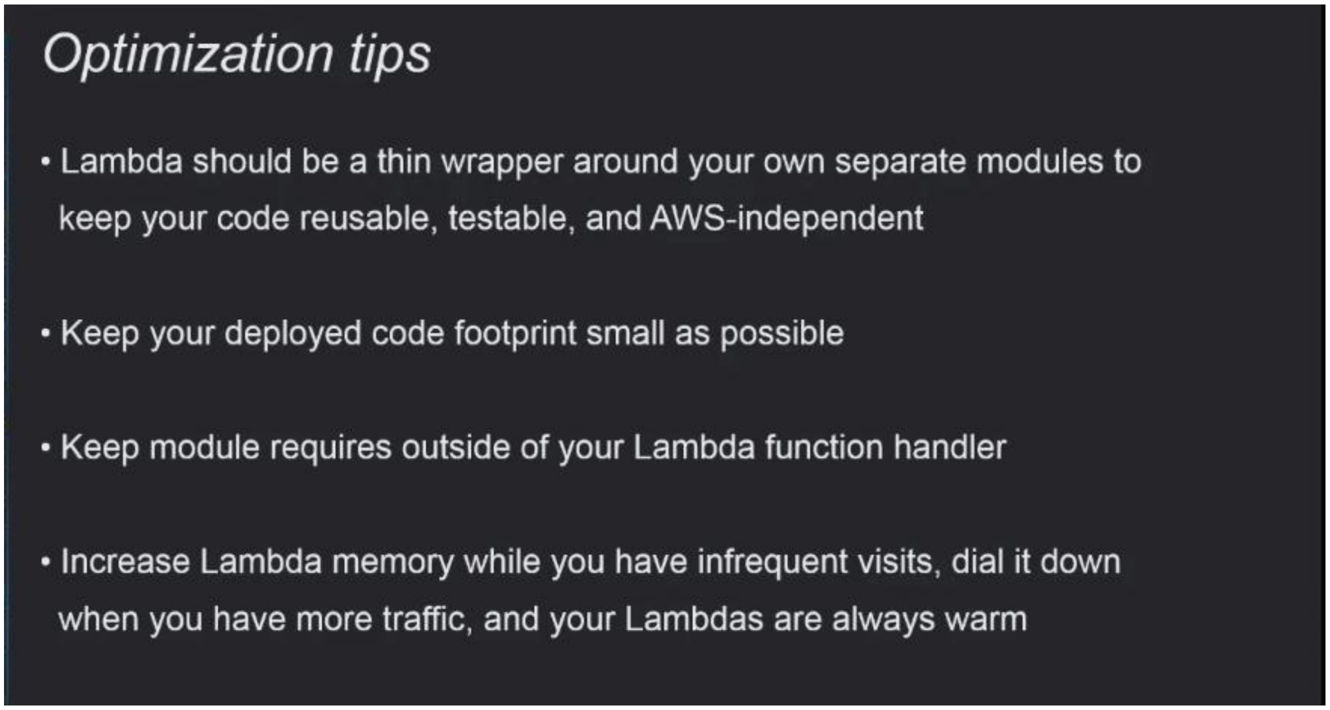
```
     L) lGreetingsHello
     E) /greetings/hello - GET

  >  Deploy Selected -->
JAWS: -------------------------------------------
JAWS:  Dashboard:  Deploying Lambdas...
JAWS: -------------------------------------------
JAWS: Lambda Deployer:  Packaging "lAwsmImagesThumbnail"...
JAWS: Lambda Deployer:  Saving in dist dir /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lAwsmImagesThumbnail@14
44247573987
JAWS: Getting ENV file from S3 bucket: jaws.dev.useast1.rynop.com in us-east-1
JAWS: Lambda Deployer:  Compressed lambda written to /var/folders/24/j5b3zgnj79902_4xk7r1ygg40000gn/T/lAwsmImagesTh
umbnail@1444247573987/package.zip
JAWS: Lambda Deployer:  Uploading lAwsmImagesThumbnail to jaws.dev.useast1.rynop.com
JAWS: Running CloudFormation lambda deploy...
JAWS: Lambda Deployer:  Done deploying lambdas in us-east-1
JAWS: Lambda Deployer:  Successfully deployed lambdas to the requested regions!
JAWS: -------------------------------------------
JAWS:  Dashboard:  Deploying Endpoints...
JAWS: -------------------------------------------
JAWS: No selected Endpoints to deploy.
JAWS: -------------------------------------------
JAWS:  Dashboard:  Deployments Completed
JAWS: -------------------------------------------
~/Dropbox/jaws/re-invent/aws_modules/awsm-images/thumbnail $
```


Lambda > Functions

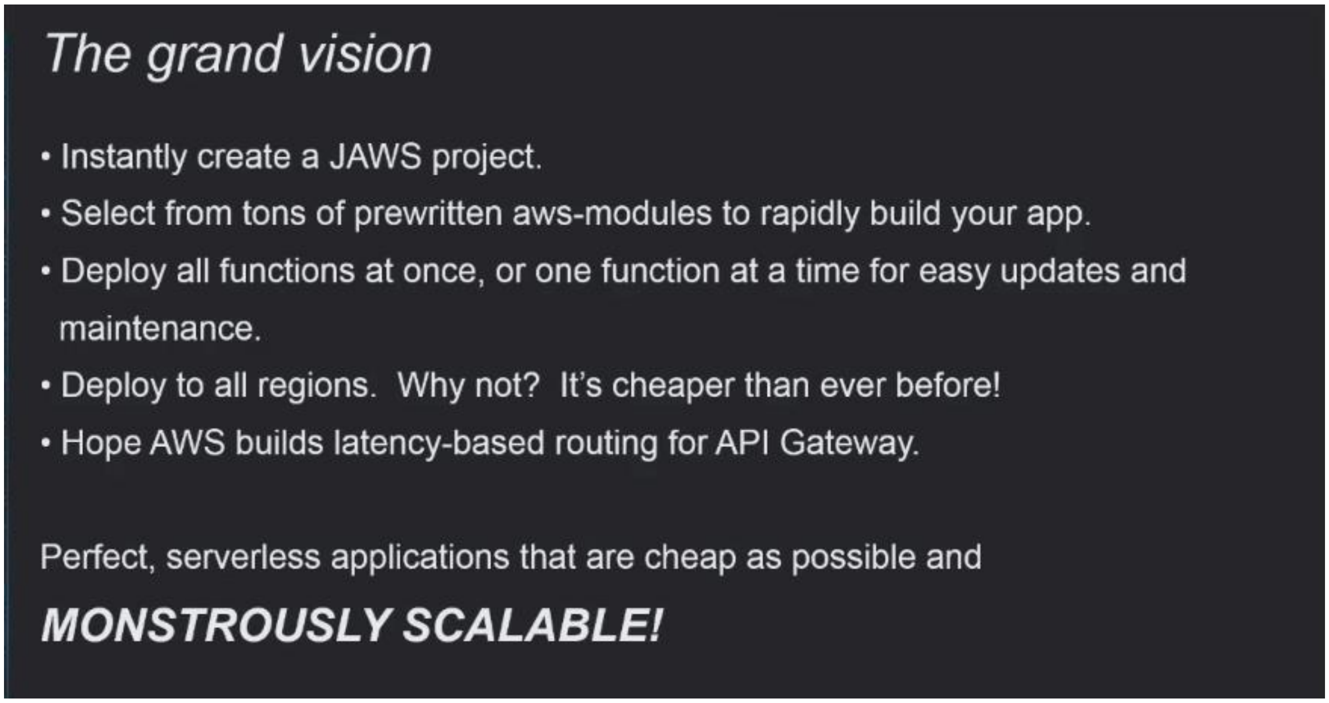
You have 2 Lambda function(s) using 2.0 MB of code storage. Choose any Lambda function to view details on invocation requests, duration, and errors (metrics may take up to 60 seconds to appear).

Function name	Description	Code size	Memory (MB)	Timeout (s)
dev-re-invent-l-lAwsmImagesThumbnail-DNL50RS71IB8		2.0 MB	1024	6
dev-re-invent-l-lGreetingsHello-YG7S374GDXZT		1.4 kB	1024	6

The original, un-optimized version is up to 2.0MB in size.


Optimization tips

- Lambda should be a thin wrapper around your own separate modules to keep your code reusable, testable, and AWS-independent
- Keep your deployed code footprint small as possible
- Keep module requires outside of your Lambda function handler
- Increase Lambda memory while you have infrequent visits, dial it down when you have more traffic, and your Lambdas are always warm

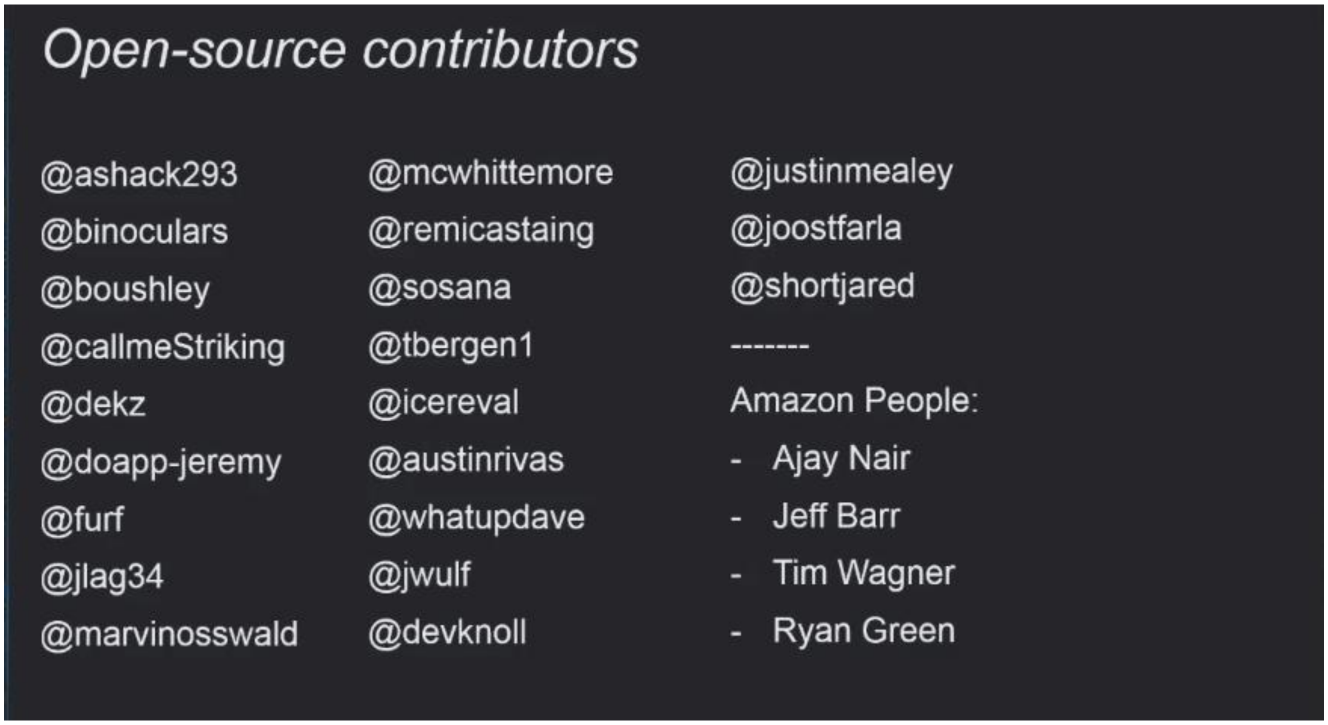


The grand vision

- Instantly create a JAWS project.
- Select from tons of prewritten aws-modules to rapidly build your app.
- Deploy all functions at once, or one function at a time for easy updates and maintenance.
- Deploy to all regions. Why not? It's cheaper than ever before!
- Hope AWS builds latency-based routing for API Gateway.

Perfect, serverless applications that are cheap as possible and

MONSTROUSLY SCALABLE!



Open-source contributors

@ashack293
@binoculars
@boushley
@callmeStriking
@dekz
@doapp-jeremy
@furf
@jlag34
@marvinosswald

@mcwhittemore
@remicastaing
@sosana
@tbergen1
@icereval
@austinrivas
@whatupdave
@jwulf
@devknoll

@justinmealey
@joostfarla
@shortjared

Amazon People:
- Ajay Nair
- Jeff Barr
- Tim Wagner
- Ryan Green

Thank You!

Questions?

jawsframework.com

Austen Collins
Oakland, CA.
Twitter: @austencollins
Github: @ac360
Email: austen@servant.co

Ryan Pendergast
Rochester, MN.
DoApp, Inc.
Twitter: @rynop
Github: @doapp-ryanp
Email: ryan.pendergast@gmail.com