Ask Django

여러분의 파이썬/장고 페이스메이커가 되겠습니다.

EP 16. Serverless 배포 - AWS Lambda

AWS Lambda에 Django 애플리케이션을 간략히 배포해보겠습니다.

- NO STATIC FILE 서빙
- SQLITE 3 데이터베이스 사용 : 실서비스에서는 RDS 사용 권장

배포준비

장고 - 서비스 배포하기 코스를 참고

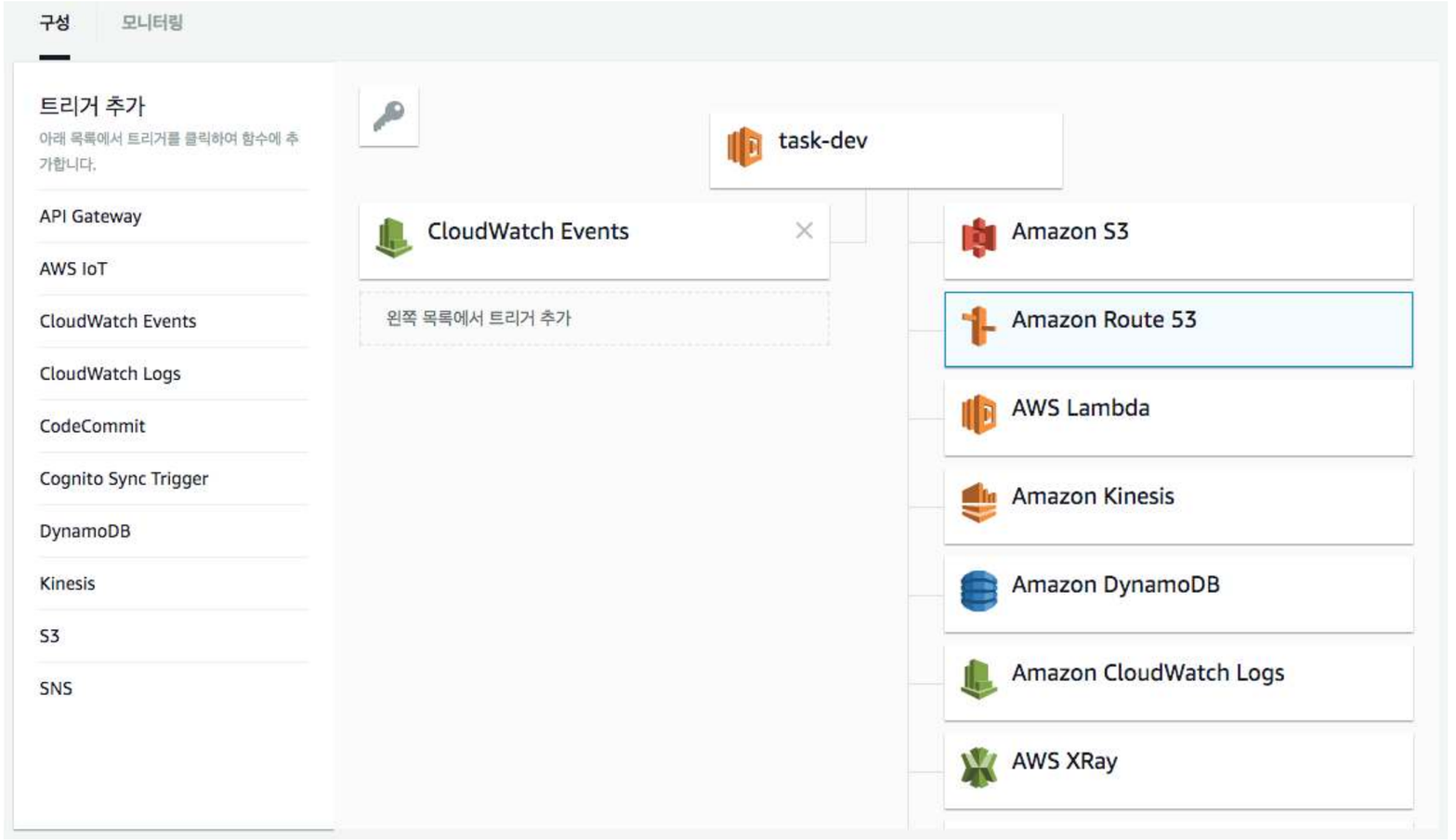
- 구동환경별 requirements.txt 분기
- 구동환경별 settings 분기

ZAPPA

Serverless Python Web Services - Powered by AWS Lambda and API Gateway

- https://www.zappa.io
- 장점
  - **WSGI Application을 지원**하기에, 파이썬 웹애플리케이션을 그대로 배포 가능
- 단점
  - Pure 파이썬 라이브러리만 지원
  - 혹은 Lambda 구동환경(Amazon Linux)에 맞춘 팩키지를 써도 됩니다. #lib
- AWS Lambda에서는 Python 2.7와 3.6 만을 지원합니다.
  - 3.6을 쓰기 위해서는 로컬머신에 3.6이 있어야 합니다.
- 필수조건
  - 파이썬 3.6
  - zappa는 배포 시에 호스트 머신의 `virtualenv`내 팩키지를 가져와서 압축하기 때문에, **virtualenv를 통한 실행이 필수**입니다.

ZAPPA에서 구성



사용하는 명령 요약

```
pip install zappa
zappa init
zappa deploy
```

Credential 환경변수 설정

IAM 유저 생성 필요한 최소한의 권한에 대한 토론

1. 인증정보를 환경변수 AWS_ACCESS_KEY_ID와 환경변수 AWS_SECRET_ACCESS_KEY에 저장해서 활용
2. ~/.aws/config 에 키 설정 저장하여, 환경변수 AWS_PROFILE을 통한 접근

```
[profile zappa]
aws_access_key_id = your_access_key_id_specific_to_zappa
aws_secret_access_key = your_secret_access_key_specific_to_zappa
region = ap-northeast-2
```

선택1) 로컬 파이썬이 3.6일 경우, 로컬머신에 셋업

```
mkdir myproj
cd myproj

# 각자에 맞게 가상환경을 생성해주시고,

pip install -r requirements.txt  # zappa 필히 포함시킬 것.

zappa init
```

선택 2) Python 3.6이 없을 경우, 직접 설치

- Anaconda를 쓰는 경우, conda를 통한 가상환경 생성 시에 파이썬 버전을 3.6으로 지정
  - anaconda는 여러 버전을 설치할 필요없습니다. 하나의 버전만 설치되어있으면 됩니다.
- 맥/리눅스일 경우, pyenv를 통한 3.6 설치

선택 3) (추천) Docker 환경에 셋업 #doc 1 #doc 2

Docker가 설치되어 있지 않으신분은 Docker Community Edition을 설치해주세요.

로컬에 AWS Lambda 구동환경에 동일한 환경을 만들 Dockerfile

```
FROM lambci/lambda:build-python3.6

WORKDIR /var/task

# Fancy prompt to remind you are in zappashell
RUN echo 'export PS1="\[\e[36m\]zappashell>\[\e[m\] "' >> /root/.bashrc

# Additional RUN commands here
# RUN yum clean all && \
#    yum -y install <stuff>

CMD ["bash"]
```

Docker 실행

```
호스트쉘> docker run -ti \
    -e AWS_PROFILE=zappa \
    -v $(pwd):/var/task \
    -v ~/.aws/:/root/.aws \
    --rm myzappa

zappashell> virtualenv ve
zappashell> source ve/bin/activate
zappashell> pip install -r requirements.txt
```

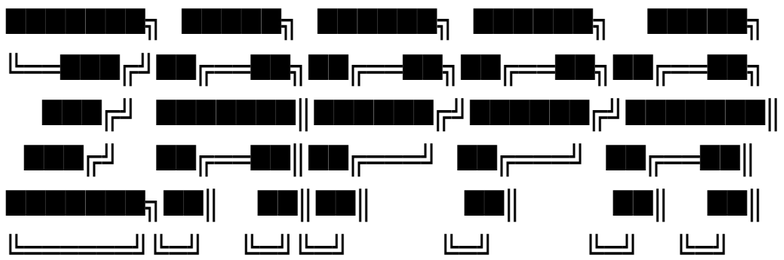
Tip: AWS_PROFILE 대신에 AWS_SECRET_ACCESS_KEY, AWS_ACCESS_KEY_ID, AWS_DEFAULT_REGION 환경변수를 대신 설정하셔도 됩니다.

zappa init

zappa init 실행

```
(ve) zappashell> zappa init
```

웰컴 메세지



```
Welcome to Zappa!

Zappa is a system for running server-less Python web applications on AWS Lambda and AWS API Gateway.
This `init` command will help you create and configure your new Zappa deployment.
Let's get started!
```

배포 구분 : dev (개발), staging (비프로덕션 테스트), production (실서비스)

```
Your Zappa configuration can support multiple production stages, like 'dev', 'staging', and 'production'.
What do you want to call this environment (default 'dev'):
```

AWS Credential - Profile 선택

```
AWS Lambda and API Gateway are only available in certain regions. Let's check to make sure you have a profile set up in one that will work.
Okay, using profile zappa!
```

zappa 이름의 *profile*이 있어서, 디폴트 지정되었습니다.

배포코드를 저장할 S3 버킷명

```
Your Zappa deployments will need to be uploaded to a private S3 bucket.
If you don't have a bucket yet, we'll create one for you too.
What do you want call your bucket? (default 'zappa-1i7vt2z5p'):
```

구동할 장고 애플리케이션 settings 경로

```
It looks like this is a Django application!
What is the module path to your projects's Django settings?
We discovered: askdjango.settings
Where are your project's settings? (default 'askdjango.settings'):
```

글로벌 배포 여부

```
You can optionally deploy to all available regions in order to provide fast global service.
If you are using Zappa for the first time, you probably don't want to do this!
Would you like to deploy this application globally? (default 'n') [y/n/(p)rimary]: n
```

생성된 zappa_settings.json 내역

```
Okay, here's your zappa_settings.json:

{
    "dev": {
        "aws_region": "ap-northeast-2",
        "django_settings": "askdjango.settings",
        "profile_name": "zappa",
        "project_name": "task",
        "runtime": "python3.6",
        "s3_bucket": "zappa-1i7vt2z5p"
    }
}

Does this look okay? (default 'y') [y/n]: y
```

나머지 안내

```
Done! Now you can deploy your Zappa application by executing:

    $ zappa deploy dev

After that, you can update your application code with:

    $ zappa update dev

To learn more, check out our project page on GitHub here: https://github.com/Miserlou/Zappa
and stop by our Slack channel here: https://slack.zappa.io

Enjoy!,
 ~ Team Zappa!
```

첫 배포 수행

```
(ve) zappashell> zappa deploy dev
(botocore 1.8.6 (/var/task/ve/lib/python3.6/site-packages), Requirement.parse('botocore<1.8.0,>=1.7.0'), {'boto3'})
Calling deploy for stage dev..
Downloading and installing dependencies..
 - pillow==4.3.0: Downloading
100%|████████████████████████████████████████| 5.82M/5.82M [00:01<00:00, 5.35MB/s]
 - sqlite==python36: Using precompiled lambda package
Packaging project as zip.
Uploading task-dev-1512376218.zip (21.9MiB)..
100%|████████████████████████████████████████| 23.0M/23.0M [00:03<00:00, 5.99MB/s]
Scheduling..
Scheduled task-dev-zappa-keep-warm-handler.keep_warm_callback with expression rate(4 minutes)!
Uploading task-dev-template-1512376324.json (1.6KiB)..
100%|████████████████████████████████████████| 1.61K/1.61K [00:00<00:00, 50.7KB/s]
Waiting for stack task-dev to create (this can take a bit)..
100%|████████████████████████████████████████| 4/4 [00:12<00:00,  4.31s/res]
Deploying API Gateway..
Deployment complete!: https://rw8qusuhq7.execute-api.ap-northeast-2.amazonaws.com/dev
```

접속 URL

```
https://rw8qusuhq7.execute-api.ap-northeast-2.amazonaws.com/dev/
        ^^^^^^^^^^^^^^^^^^^^^                                  ^^^
       Auto Generated API Gateway                  Your Zappa Environment
```

추후 업데이트 시

```
(ve) zappashell> zappa update dev

(botocore 1.8.6 (/var/task/ve/lib/python3.6/site-packages), Requirement.parse('botocore<1.8.0,>=1.7.0'), {'boto3'})
Calling update for stage dev..
Downloading and installing dependencies..
 - pillow==4.3.0: Using locally cached manylinux wheel
 - sqlite==python36: Using precompiled lambda package
Packaging project as zip.
Uploading task-dev-1512376455.zip (21.9MiB)..
100%|████████████████████████████████████████| 23.0M/23.0M [00:03<00:00, 6.20MB/s]
Updating Lambda function code..
Updating Lambda function configuration..
Uploading task-dev-template-1512376560.json (1.6KiB)..
100%|████████████████████████████████████████| 1.61K/1.61K [00:00<00:00, 44.0KB/s]
Deploying API Gateway..
Scheduling..
Unscheduled task-dev-zappa-keep-warm-handler.keep_warm_callback.
Scheduled task-dev-zappa-keep-warm-handler.keep_warm_callback with expression rate(4 minutes)!
Your updated Zappa deployment is live!: https://rw8qusuhq7.execute-api.ap-northeast-2.amazonaws.com/dev
```

로그 확인

```
zappashell> zappa tail

zappashell> zappa tail --http         # HTTP 로그만 보기

zappashell> zappa tail --non-http     # 비 HTTP 로그만 보기

zappashell> zappa tail --since 4h     # 4시간 이내 로그부터
zappashell> zappa tail --since 1m     # 1분 이내 로그부터
zappashell> zappa tail --since 1mm    # 1달 이내 로그부터

zappashell> zappa tail --http --filter "POST"
```

장고 manage 명령

표준입력이 불가합니다. 단방향 명령만 가능합니다.

```
# showmigrations 명령 수행
zappashell> zappa manage dev showmigrations

# 옵션이 필요한 Management 명령일 경우
zappashell> zappa manage dev "shell --version"
```

참고

- Github 공식 저장소
- edgarroman.github.io/zappa-django-guide/setup/
  - zappa-django-utils
- songyunseop.github.io/post/2017/09/serverless-microservice-with-zappa-2-/

Advanced Topic

- 커스텀 도메인 설정

zappa-django-utils

데이터베이스

고성능을 위해서는 AWS RDS를 쓰는 것이 정답이나, 방문자가 거의 없는 사이트는 SQLite3 데이터베이스를 활용해볼 수도 있겠습니다. zappa-django-utils를 활용하여 S3에 동기화된 SQLite3 데이터베이스를 손쉽게 연동하실 수 있습니다.

```
# settings
INSTALLED_APPS += ['zappa_django_utils']

DATABASES = {
    'default': {
        'ENGINE': 'zappa_django_utils.db.backends.s3sqlite',
        'NAME': 'db.sqlite3',     # 원하는 DB파일명
        'BUCKET': 'zappa-dist',   # 원하는 버킷명
    }
}
```

명령

```
zappashell> zappa manage dev migrate
```

superuser 생성

표준입력이 안 되기에 `zappa manage dev createsuperuser` 명령은 사용불가합니다. `zappa-django-utils`에서 지원하는 `create_admin_user` 명령을 사용합니다. 랜덤 암호를 생성하여 superuser계정을 생성해줍니다.

```
(ve) zappashell> zappa manage dev create_admin_user

(botocore 1.8.6 (/var/task/ve/lib/python3.6/site-packages), Requirement.parse('botocore<1.8.0,>=1.7.0'), {'boto3'})
[DEBUG] 2017-12-04T10:19:05.162Z 8e5e2ad0-d8dc-11e7-ab32-8f51a9665759 Zappa Event: {'manage': 'create_admin_user'}
Creating new admin user..
Created user "admin", email: "admin@admin.com", password: Z1V9W4PP70
```

lambda-packages 지원 라이브러리 #github

MySQL-Python, OpenCV, Pillow, PyNACl, bcrypt, cffi, cryptography, datrie_extended, lxml, misaka, mysqlclient, numpy, pycopg2, pycrypto, pylibmc, pyproj, python-Levenshtein, python-ldap, regex, sqlite3, xmlsec

하지만, `lambci/lambda:build-python3.6` 기반의 Docker 이미지가 Lambda 환경과 동일하기에, `pip install`한 라이브러리를 Lambda에 올려 사용할 수 있습니다.

AWS Zappa에 대해서는
장고 - 서비스 배포하기 코스에서
더 자세히 다뤄보겠습니다.

Life is short,
use Python3/Django.