

boto3

1. 크롤링 서버 생성 및 설정

auto.py

```
import requests, json
```

```
import datetime
```

```
from bs4 import BeautifulSoup
```

```
WEBHOOK_URL = "https://hooks.slack.com/services/TNREL1KJR/BQHDMJ9TM/  
NkAc2UDpQemyH2oCkSbYie10"
```

```
def send_slack(msg, channel="#rada", username="슬랙봇"):
```

```
    payload = {  
        "channel": channel,  
        "username": username,  
        "text": msg,  
    }
```

```
    response = requests.post(  
        WEBHOOK_URL,  
        data = json.dumps(payload),  
    )
```

```
# 날씨 정보 함수
```

```
def forecast(lat, lng, TOKEN="52b4cd56c007d2863acc0c1a9c8de16c"):
```

```
    url = "https://api.darksky.net/forecast/{}/{},{}".format(TOKEN, lat, lng)
```

```

response = requests.get(url)
json_obj = response.json()
return json_obj["currently"]["summary"]

```

```

lat, lng = 37.5665, 126.9780
msg = forecast(lat, lng)
send_slack(msg, username="날씨봇")

```

```

crontab -e

```

```

@reboot /home/ubuntu/.pyenv/versions/python3/bin/python /home/ubuntu/auto.py

```

인스턴스를 중지 시켰다가 다시 시작해서 슬랙으로 메시지가 전송되는지 확인

2. 컨트롤 서버 설정

boto3 설치

- pip install boto2

IAM 서비스로 부터 접속키 보안키 획득 후 아래 코드 작성

ec2.py

```

Import sys

```

```

import boto3

```

```
aws_access_key_id = "<본인키>"
```

```
aws_secret_access_key = "<본인키>"
```

```
instance_id = "<본인키>"
```

```
client = boto3.client(
    'ec2',
    aws_access_key_id=aws_access_key_id,
    aws_secret_access_key=aws_secret_access_key,
    region_name='ap-northeast-2',
)
```

```
command = sys.argv[1]
```

```
If command == "stop":
```

```
    client.stop_instances(InstanceIds=[instance_id], DryRun=False)
    print("stop")
```

```
elif command == "start":
```

```
    client.start_instances(InstanceIds=[instance_id], DryRun=False)
    print("start")
```

인스턴스 stop과 start가 잘되는지 확인

시작 : python ec2.py start

정지 : python ec2.py stop

```
crontab -e
```

```
56 * * * * /home/ubuntu/.pyenv/versions/python3/bin/python /home/ubuntu/ec2.py
start
```

```
57 * * * * /home/ubuntu/.pyenv/versions/python3/bin/python /home/ubuntu/ec2.py
stop
```

3. AWS Lambda 사용

AWS의 Lambda는 코드를 함수로 실행시켜주는 서비스 입니다. AWS의 Lambda를 이용하면 위에 있는 컨트롤 서버를 사용할 필요가 없습니다.

AWS Lambda의 사용방법

1. AWS Console에서 Lambda 서비스로 이동
2. 함수 생성 클릭 후 아래와 같이 설정 하여 함수 생성 (런타임 Python 3.6 설정)

The screenshot shows the AWS Lambda console interface for creating a new function. The 'Basic information' section is expanded, showing the following details:

- Function name:** naverCrawlingStart
- Runtime:** Python 3.6
- Permissions:** Create new role

3. lambda function 코드 추가

```

import json
import boto3

def lambda_handler(event, context):

    aws_access_key_id="<본인의 키값>"
    aws_secret_access_key="<본인의 키값>"
    instance_id = "<시작할 인스턴스 아이디>"

    client = boto3.client('ec2',
                           aws_access_key_id=aws_access_key_id,
                           aws_secret_access_key=aws_secret_access_key,
                           region_name='ap-northeast-2',
                           )


    client.start_instances(InstanceIds=[instance_id], DryRun=False)
    # 인스턴스 종료시 아래의 코드 사용
    # client.stop_instances(InstanceIds=[instance_id], DryRun=False)

    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }

```

4. 제한시간을 1분으로 수정
5. 저장 후 테스트 생성
6. 테스트 버튼 클릭 후 인스턴스가 start 되는지 확인

트리거 구성


CloudWatch Events
aws events management-tools

규칙
기존의 규칙을 선택하거나 새로운 규칙을 생성합니다.

새 규칙 생성

새 규칙 선택 또는 생성

규칙 이름*
규칙을 고유하게 식별하려면 이름을 입력합니다.

test_stop

규칙 설명
규칙에 대한 선택적 설명을 제공합니다.

규칙 유형
이벤트 패턴이나 자동 일정에 따라 대상을 트리거합니다.

☐ 이벤트 패턴
☒ 예약 표현식

예약 표현식*
cron 또는 rate 표현식을 사용하여 자동 일정에 따라 대상을 자체 트리거합니다. cron 표현식은 UTC 기준입니다.

cron(55|02 9 8 ? 2019)

예: rate(1 day), cron(0 17 ? * MON-FRI *)

Lambda는 Amazon CloudWatch Events이(가) 이 트리거에서 Lambda 함수를 호출하는 데 필요한 권한을 추가합니다. Lambda 권한 모델에 대해 [자세히 알아보기](#).

☒ **트리거 활성화**

지금 트리거를 활성화하거나 테스트를 위해 비활성화된 상태로 생성합니다(권장).

7. 왼쪽의 버튼에서 CloudWatch Events 추가

8. 아래와 같이 트리거 규칙 추가

예약 표현식은 아래와 같이 사용 (UTC 기준)

cron(15 12 8 1 ? 2019) - 2019년 1월 8일 12시 15분

- https://docs.aws.amazon.com/ko_kr/lambda/latest/dg/tutorial-scheduled-events-schedule-expressions.html

9. 추가 버튼 클릭 후 저장
10. 설정한 시간에 인스턴스가 실행되는지 확인
11. 시간 텀을 주어 정지도 같은 방법으로 설정