

클라우드 컴퓨팅 과제 #1

: AWS Lambda 실습

2016130927 박준영

1. 시나리오 구성

AWS Lambda 실습 영상과 동일한 시나리오를 자바스크립트가 아닌 파이썬 언어를 통해 구현하였다. 자세한 시나리오 구성은 다음과 같다. 서버리스 환경에서 트리거로 연결된 S3 버킷에 이미지 파일을 업로드할 때, Lambda 함수는 이 파일을 다운로드 한 뒤 환경변수에 설정된 세 가지 함수(흑백, 투명, 반전) 중 한 가지를 실행한다. 이후 결과물이 설정된 버킷에 자동으로 업로드된다.

기존 실습에서는 업로드되는 파일 확장자명과 입력되는 환경변수에 해당 사항이 없으면 call back하는 방식으로 구현되었으나, 본 시나리오는 이중 확장자명만 검사하여 jpeg 또는 png만 받아들이고, 나머지는 return None으로 그대로 종료한다. 반면 환경변수 값은 설정한 변수 중에 존재하지 않더라도 가공 없이 업로드되도록 구현하였다.

2. Lambda 함수 코드

Lambda 함수 CL-ASSIGNMENT1의 구성은 다음과 같다.

(1). Lambda_handler: AWS Lambda 함수의 메인 함수로 이벤트 트리거로 입력된 event를 통해 버킷과 키, 파일 정보 등을 인식하고, S3 버킷에서 파일을 다운로드한다. Splitext 함수를 통해 다운로드한 파일의 확장자명이 jpeg 혹은 png인지 확인한다. 이후 환경변수 입력값이 설정된 세 가지 함수(흑백, 투명, 반전) 중에 존재하면 각 함수를 실행, 다운로드한 이미지 파일을 변형시키고, 존재하지 않으면 아무 가공도 하지 않는다. 마지막으로 트리거로 연결된 버킷 이름에 '-output'이 추가된 이름을 가진 버킷에 위의 과정을 거친 이미지 파일을 업로드한다.

(2). gray_image: 가공할 이미지 파일을 받아 convert 함수를 이용해 grayscale, 즉 흑백으로 만든다. 가공한 파일은 이후 업로드할 곳에 저장한다.

(3). transparent_image: 가공할 이미지 파일을 받아 RGBA가 아니라 RGB로 convert한다. RGB 값을 가지고 있는 이 이미지 파일은 다시 흑백 반전되어 흰색 값이 검은색 값이 된다. 이 상태를 흑백 모드로 convert하면, 이 이미지를 기존의 RGB에 alpha(투명도를 결

정하는 값)로 putalpha시켜줄 수 있다. 즉, Convert("L")을 통해 검정색이 된 배경을 투명하게 설정한 이미지를 업로드한다.

(4). negative_image: transparent_image가 알파 값을 절반으로 하여 투명화하는 과정이라면, negative 즉 흑백 반전은 각 픽셀이 가지고 있는 RGB 값을 완전히 반전(즉 255 - 각 RGB 색상값)한 것이다.

(5). Others: PIL로부터 Image를 Import해야 하므로, Lambda에 추가할 계층에 업로드할 필요가 있다. 또한 어떤 종류의 이미지 가공을 할지 사용자의 입장에서 결정해야 하므로, 환경변수 transform을 고려해야 한다(세 가지 설정값 이외는 Default로 가공 없음)

다음은 AWS Lambda 함수 코드를 캡처한 스크린샷이다.

```
lambda_function x (+)
1 import boto3
2 import os
3 import sys
4 import uuid
5
6 from PIL import Image, ImageOps, ImageFilter
7
8 transform = os.environ.get('TRANSFORM_FUNC')
9 s3_client = boto3.client('s3')
10
11 def gray_image(image_path, gray_path):
12     with Image.open(image_path) as image:
13         gray = image.convert("L")
14         gray.save(gray_path)
15         # grayscale image
16
17 def transparent_image(image_path, transparent_path):
18     with Image.open(image_path) as image:
19         transparent = image.convert("RGBA")
20         image_invert = ImageOps.invert(transparent)
21         image_invert_L = image_invert.convert("L")
22         transparent.putalpha(image_invert_L)
23         transparent.save(transparent_path, "png")
24         #transparent image
25
26 def negative_image(image_path, negative_path):
27     with Image.open(image_path) as image:
28         negative = image.point(lambda i: 255 - i)
29         negative.save(negative_path)
30         # negativize image
31
32 def raw_image(image_path, raw_path):
33     with Image.open(image_path) as image:
34         raw = image
35         raw.save(raw_path)
36         # raw(original) image
37
38 def lambda_handler(event, context):
39     for record in event['Records']:
40         bucket = record['s3']['bucket']['name']
41         key = record['s3']['object']['key']
42         download_path = '/tmp/{}'.format(uuid.uuid4(), key)
43         upload_path = '/tmp/resized-{}'.format(key)
44         s3_client.download_file(bucket, key, download_path)
45
46         if (os.path.splitext(download_path)[1] == '.jpeg'): print("Image type jpeg")
47         elif (os.path.splitext(download_path)[1] == '.png'): print("Image type png")
48         else:
49             print("Unsupported type detected. jpeg and png only allowed.")
50             return None
51         #check whether its format jpeg/png or not. otherwise return.
52
53         if (transform == 'gray'): gray_image(download_path, upload_path), print("grayscale transform")
54         elif (transform == 'transparent'): transparent_image(download_path, upload_path), print("transparent transform")
55         elif (transform == 'negative'): negative_image(download_path, upload_path), print("negativie transform")
56         else: raw_image(download_path, upload_path), print("No transform")
57         s3_client.upload_file(upload_path, '{}-output'.format(bucket), key)
58         # Upload processed file to output bucket
```

3. 레이어 설정

계층 추가

함수 런타임 설정

런타임
Python 3.8

아키텍처
x86_64

계층 선택

계층 소스 [Info](#)
호환 런타임 및 명령 세트 아키텍처가 있는 계층에서 선택하거나 계층 버전의 Amazon 리소스 이름(ARN)을 지정합니다. 새로운 계층을 생성할 수도 있습니다.

☐ AWS 계층
AWS에서 제공하는 계층 목록에서 계층을 선택합니다.

☐ 사용자 지정 계층
AWS 계정 또는 조직에서 생성한 계층 목록에서 계층을 선택합니다.

☒ ARN 지정
ARN을 제공하여 계층을 지정합니다.

ARN 지정

Amazon 리소스 이름(ARN)을 제공하여 계층을 지정합니다.

확인

설명
Pillow==8.4.0 | 0fd903094d8ac7efb6130dc6da17f442df50dfd3e9ea0d9eaf8e2dcdcedca0

호환 런타임
Python 3.6, Python 3.7, Python 3.8

호환 아키텍처
-

취소

추가

계층 Info					
병합 주문	이름	Layer 버전	호환 런타임	호환 아키텍처	버전 ARN
1	Klayers-python38-Pillow	14	python3.6, python3.7, python3.8	-	arn:aws:lambda:us-east-1:770693421928:layer:Klayers-python38-Pillow:14

실습 영상에서 imageMagic, node module 계층을 설정하여 사용하였으나, 본 시나리오에서는 파이썬 언어를 사용하며, 이미지 가공을 위한 도구로 PIL을 이용하고자 하였다. 위 두 경우는 직접 계층에 zip 파일을 업로드해 Lambda 함수에 추가하여 사용했는데, PIL 라이브러리는 ARN을 통해 다른 사용자가 업로드한 계층을 사용 가능했다. 따라서 AWS 계층이나 사용자 지정 계층이 아닌 ARN만 지정해서 곧바로 레이어를 추가할 수 있었다.

추가한 레이어 Klayers-python38-Pillow를 통해 Lambda 함수에서 PIL.Image 등을 import할 수 있었다.

4. S3 구성 및 이벤트 트리거 설정

Lambda 함수에 이벤트 트리거로 연결된 S3 버킷 cl-assignment1은 ObjectCreated 유형에 반응하도록 설정되었다. 즉 특정 오브젝트가 만들어졌을 때(따라서 파일 업로드와 본

시나리오에서는 그 의미가 상동), lambda_handler를 통해 그 event를 관리할 수 있다.

모든 퍼블릭 액세스 차단을 비활성화하여 퍼블릭 상태가 가능하도록 만든 S3 버킷을 총 두 개(트리거용, 업로드용) 만들어 각각 cl-assignment1, cl-assignment1-output으로 이름 붙였다. 트리거 버킷에 '-output' 문자가 추가된 것으로 Lambda 함수에서 업로드용 버킷을 판별하기 때문이다. 이후 cl-assignment1을 Lambda 함수에 트리거로 추가하였다.

버킷 만들기 [Info](#)
버킷은 S3에 저장되는 데이터의 컨테이너입니다. 자세히 알아보기 [▶](#)

일반 구성

버킷 이름
cl-assignment1

이름은 S3에 저장되는 데이터의 컨테이너입니다. 자세한 내용은 [버킷 이름 규칙 참조](#)를 참조하십시오.

AWS 리전
미국 동부(버지니아 북부) us-east-1

기존 버킷에서 설정 복사 - 선택 사항
다음 구성의 버킷 설정만 복사됩니다.

버킷 선택

이 버킷의 퍼블릭 액세스 차단 설정

퍼블릭 액세스는 ACL(액세스 제어 목록), 버킷 정책, 액세스 지점 정책 또는 모두를 통해 버킷 및 객체에 부여됩니다. 이 버킷 및 객체와 관련된 퍼블릭 액세스가 차단되었는지 확인하려면 모든 퍼블릭 액세스 차단을 활성화합니다. 이 설정은 이 버킷 및 객체에 대한 퍼블릭 액세스 지점에만 적용됩니다. AWS에서는 모든 퍼블릭 액세스 차단을 활성화하도록 권장하지만, 이 설정을 적용하기 전에 퍼블릭 액세스가 없어도 애플리케이션이 올바르게 작동하는지 확인합니다. 이 버킷 또는 내부 객체에 대한 어느 정도 수준의 퍼블릭 액세스가 필요한 경우 특정 스토리지 사용 사례에 맞게 여러 개별 설정을 사용자 지정할 수 있습니다. 자세히 알아보기 [▶](#)

- ☐ 모든 퍼블릭 액세스 차단
이 설정을 활성화하면 아래 4개의 설정을 모두 활성화한 것과 같습니다. 다음 설정 각각은 서로 독립적입니다.
- ☐ 새 ACL(액세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단
S3은 새로 추가된 버킷 또는 객체에 적용되는 퍼블릭 액세스 권한을 차단하여 기존 버킷 및 객체에 대한 새 퍼블릭 액세스 ACL 생성을 금지합니다. 이 설정은 ACL을 사용하여 S3 리소스에 대한 퍼블릭 액세스를 허용하는 기존 권한을 변경하지 않습니다.
- ☐ 임의의 ACL(액세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단
S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 모든 ACL을 무시합니다.
- ☐ 새 퍼블릭 버킷 또는 액세스 지점 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단
S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 새 버킷 및 액세스 지점 정책을 차단합니다. 이 설정은 S3 리소스에 대한 퍼블릭 액세스를 허용하는 기존 정책을 변경하지 않습니다.
- ☐ 임의의 퍼블릭 버킷 또는 액세스 지점 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 및 교차 계정 액세스 차단
S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 정책을 사용하는 버킷 또는 액세스 지점에 대한 퍼블릭 및 교차 계정 액세스를 무시합니다.

모든 퍼블릭 액세스 차단을 비활성화하면 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있습니다.
정적 웹 사이트 호스팅과 같은 구체적인 확인된 사용 사례에서 퍼블릭 액세스가 필요한 경우가 아니면 모든 퍼블릭 액세스 차단을 활성화하는 것이 좋습니다.

☒ 현재 설정으로 인해 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있음을 알고 있습니다.


버킷 (4) [Info](#) [ARN 복사](#) [비어 있음](#) [삭제](#) [버킷 만들기](#)

버킷은 S3에 저장되는 데이터의 컨테이너입니다. 자세히 알아보기 [▶](#)

이름	AWS 리전	액세스	생성 날짜
cl-assignment1	미국 동부(버지니아 북부) us-east-1	객체를 퍼블릭으로 설정할 수 있음	2021. 10. 31. pm 3:31:41 PM KST
cl-assignment1-output	미국 동부(버지니아 북부) us-east-1	객체를 퍼블릭으로 설정할 수 있음	2021. 10. 31. pm 3:44:45 PM KST

트리거 (1) [활성화](#) [비활성화](#) [오류 수정](#) [삭제](#) [트리거 추가](#)

1결과

트리거
<div> S3: cl-assignment1 arn:aws:s3:::cl-assignment1</div> <div>세부 정보 알림 이름: 7c4cd589-e9a9-44fa-ba07-a64516ddd40 이벤트 유형: ObjectCreated</div>

5. 이벤트 발생 이후 결과

(1). JPEG 파일 업로드/환경변수 'raw', 'gray', 'transparent', 'negative':

환경 변수 (1)
아래의 환경 변수는 기본 Lambda 서비스 키를 이용해 저장 중 암호화됩니다.

키

값

TRANSFORM_FUNC

raw

환경 변수 (1)
아래의 환경 변수는 기본 Lambda 서비스 키를 이용해 저장 중 암호화됩니다.

키

값

TRANSFORM_FUNC

gray

환경 변수 (1)
아래의 환경 변수는 기본 Lambda 서비스 키를 이용해 저장 중 암호화됩니다.

키

값

TRANSFORM_FUNC

transparent

환경 변수 (1)
아래의 환경 변수는 기본 Lambda 서비스 키를 이용해 저장 중 암호화됩니다.

키

값

TRANSFORM_FUNC

negative

위는 환경 변수 값에 각각 'raw(사실 이후 세 값을 제외한 그 어떤 값을 넣어도 작동한다)', 'gray', 'transparent', 'negative'로 준 모습이다.

이름	유형	마지막 수정	크기	스토리지 클래스
SAMPLE_GRAY.jpeg	jpeg	2021. 10. 31. pm 10:19:47 PM KST	750.5KB	Standard
SAMPLE_NEGATIVE.jpeg	jpeg	2021. 10. 31. pm 10:21:06 PM KST	750.5KB	Standard
SAMPLE_RAW.jpeg	jpeg	2021. 10. 31. pm 10:30:32 PM KST	750.5KB	Standard
SAMPLE_TRANSPARENT.jpeg	jpeg	2021. 10. 31. pm 10:19:24 PM KST	750.5KB	Standard

이름	유형	마지막 수정	크기	스토리지 클래스
SAMPLE_GRAY.jpeg	jpeg	2021. 10. 31. pm 10:19:51 PM KST	627.5KB	Standard
SAMPLE_NEGATIVE.jpeg	jpeg	2021. 10. 31. pm 10:21:10 PM KST	685.4KB	Standard
SAMPLE_RAW.jpeg	jpeg	2021. 10. 31. pm 10:30:34 PM KST	685.5KB	Standard
SAMPLE_TRANSPARENT.jpeg	jpeg	2021. 10. 31. pm 10:19:28 PM KST	28.4KB	Standard

Lambda 함수의 트리거로 설정한 S3 버킷 cl-assignment1에 SAMPLE 파일(jpeg 확장자)을 각각 네 번 업로드한 모습이다. 모두 동일한 SAMPLE이며 업로드 시 퍼블릭 환경을 허용하도록 설정하였다.

Lambda 함수 코드 내에서 자동적으로 트리거 이벤트 버킷 'cl-assignmnet1'에 '-output'을 추가한 이름을 가진 버킷에 가공한 이미지 파일을 업로드하였으므로, 자동적으로 본 버킷에 파일 업로드가 이루어진다. 다음은 업로드할 때 사용한 SAMPLE 이미지 파일과 각 결과물이다.



SAMPLE.jpeg



(1). SAMPLE_RAW.jpeg



(2). SAMPLE_GRAY.jpeg



(3). SAMPLE_TRANSPARENT.jpeg



(4). SAMPLE_NEGATIVE.jpeg

SAMPLE 이미지 파일은 각각 아무런 가공도 거치지 않은 (1) raw, 흑백으로 표현한 (2). Gray, 배경에 투명도를 더한 (3). Transparent, 각 RGB 값을 반전한 (4). Negative 과정이 적용된 이미지 파일로 S3 버킷(cl-assignment1-output)으로 업로드되었다.

(2). PNG 파일 업로드/환경변수 'raw', 'gray', 'transparent', 'negative':

(1)과 마찬가지로 PNG 이미지 파일을 각각 환경 변수 네 가지 조건을 준 상태로 이벤트 트리거 S3 버킷 cl-assignment1에 업로드하였다.

The screenshot shows the AWS S3 console interface. The top bucket, 'cl-assignment1', contains four PNG files. A red box highlights these files, and a red arrow points to the bottom bucket, 'cl-assignment1-output', which also contains the same four files. The files are: SAMPLE2_GRAY.png, SAMPLE2_NEGATIVE.png, SAMPLE2_RAW.png, and SAMPLE2_TRANSPARENT.png.

이름	유형	마지막 수정	크기	스토리지 클래스
SAMPLE2_GRAY.png	png	2021. 11. 1. am 9:59:02 AM KST	2.1MB	Standard
SAMPLE2_NEGATIVE.png	png	2021. 11. 1. am 9:59:45 AM KST	2.1MB	Standard
SAMPLE2_RAW.png	png	2021. 11. 1. am 10:00:02 AM KST	2.1MB	Standard
SAMPLE2_TRANSPARENT.png	png	2021. 11. 1. am 9:59:24 AM KST	2.1MB	Standard

이름	유형	마지막 수정	크기	스토리지 클래스
SAMPLE2_GRAY.png	png	2021. 11. 1. am 9:59:07 AM KST	621.6KB	Standard
SAMPLE2_NEGATIVE.png	png	2021. 11. 1. am 9:59:56 AM KST	2.1MB	Standard
SAMPLE2_RAW.png	png	2021. 11. 1. am 10:00:14 AM KST	2.1MB	Standard
SAMPLE2_TRANSPARENT.png	png	2021. 11. 1. am 9:59:34 AM KST	2.4MB	Standard

JPEG 이미지 파일을 업로드한 경우와 마찬가지로, PNG 파일을 업로드하였을 때에도 자동으로 트리거 함수의 이름에 매핑된 다른 S3 버킷 cl-assignment1-output에 자동으로 Lambda 함수가 적용된 SAMPLE2 이미지 파일이 업로드된다.



SAMPLE2.png



(1). SAMPLE2_RAW.png



(2). SAMPLE2_GRAY.png



31). SAMPLE2_TRANSPARENT.png



(4). SAMPLE2_NEGATIVE.png

(3). JPG 파일 업로드/환경변수 'gray' (설정된 확장자명이 아님):



SAMPLE3.jpg

CloudWatch > Log groups > /aws/lambda/CL-ASSIGNMENT1 > 2021/11/01/[\$LATEST]1f2937aff2d4acba53e00b42e4cd7f5

로그 이벤트
You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

이벤트 필터링

타임스탬프	메시지
2021-11-01T10:12:48.715+09:00	START RequestId: b3fb751d-9cc7-4c36-bb86-e485cf088c3e Version: \$LATEST
2021-11-01T10:12:48.867+09:00	Unsupported type detected. .jpeg and .png only allowed. Unsupported type detected. .jpeg and .png only allowed.
2021-11-01T10:12:48.888+09:00	END RequestId: b3fb751d-9cc7-4c36-bb86-e485cf088c3e
2021-11-01T10:12:48.888+09:00	REPORT RequestId: b3fb751d-9cc7-4c36-bb86-e485cf088c3e Duration: 173.62 ms Billed Duration: 174 ms Memory Size: 256 MB Max Memory Used: 73 MB Init Duration: 431.16 ms

Lambda 함수 코드에서 jpeg와 png 확장자명만 허용했으므로 본 jpg 파일은 가공하지 않는다고 설정하였다. 따라서 곧바로 return None 처리되어, 비록 트리거인 버킷에는 파일이 업로드되었으나 업로드 과정은 이루어지지 않았다.

cl-assignment1 info

객체 (1)

객체는 Amazon S3에 저장되어 있는 기본 엔티티입니다. [Amazon S3 인벤토리](#)를 사용하여 버킷에 있는 모든 객체의 목록을 얻을 수 있습니다. 다른 사용자가 객체에 액세스할 수 있게 하려면 명시적으로 권한을 부여해야 합니다. [자세히 알아보기](#)

업로드

이름	유형	마지막 수정	크기	스토리지 클래스
SAMPLE3.jpg		2021. 11. 1. am 10:12:46 AM KST	64.2KB	Standard

cl-assignment1-output info

객체 (0)

객체는 Amazon S3에 저장되어 있는 기본 엔티티입니다. [Amazon S3 인벤토리](#)를 사용하여 버킷에 있는 모든 객체의 목록을 얻을 수 있습니다. 다른 사용자가 객체에 액세스할 수 있게 하려면 명시적으로 권한을 부여해야 합니다. [자세히 알아보기](#)

업로드

객체 없음
이 버킷에 객체가 없습니다.