

## [Q1] AWS Lambda(람다) 를 사용하여 Server-Less 서비스 개발: 람다 서비스 체험하기

AWS Free Tier에서 람다 서비스를 무료로 사용할 수 있는지 여부 확인: 매 월 1백만건의 무료 요청 사용 가능

The screenshot shows the AWS Free Tier page for AWS Lambda. The top navigation bar includes links for products, solutions, pricing, documentation, learning, partner network, AWS Marketplace, and customer support. Below this, a secondary navigation bar highlights 'AWS 프리 티어' (AWS Free Tier) along with '개요' (Overview), 'FAQ', and '이용 약관' (Terms of Use).

### 프리 티어 세부 정보

**필터링 기준:**  
[모든 필터 지우기](#)

**▼ 티어 유형**

- ☐ 추천
- ☐ 12개월 무료
- ☐ 언제나 무료
- ☐ 평가판

**▼ 제품 카테고리**

- ☐ 분석
- ☐ 애플리케이션 통합
- ☐ 비즈니스 생산성
- ☐ 컴퓨팅
- ☐ 컨테이너
- ☐ 고객 참여
- ☐ 데이터베이스
- ☐ 개발자 도구
- ☐ 최종 사용자 컴퓨팅
- ☐ 프런트 엔드 웹 및 모바일
- ☐ Game Tech

**컴퓨팅**

프리 티어 언제나 무료

## AWS Lambda

# 1백만

월별 무료 요청

이벤트에 응답하여 코드를 실행하고 자동으로 컴퓨팅 리소스를 관리하는 컴퓨팅 서비스입니다.

월별 무료 요청 1,000,000건

월별 최대 320만 초의 컴퓨팅 시간

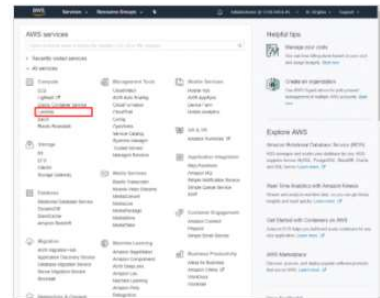
^

<https://aws.amazon.com/ko/getting-started/hands-on/run-serverless-code/>

위 URL을 참고하여 hello-world-python 예제를 실행하고, 테스트

## 1단계: Lambda 콘솔로 이동

여기를 클릭하면 AWS Management Console이 새 브라우저 창에서 열리므로 이 단계별 안내서를 계속 열어 놓을 수 있습니다. [컴퓨팅] 아래에 있는 **Lambda**를 찾아 클릭하여 AWS Lambda 콘솔을 엽니다.



(확대하려면 클릭)

## 2단계: Lambda Blueprint 선택

블루프린트는 일부 최소한의 처리를 수행할 수 있는 예제 코드를 제공합니다. 대부분 블루프린트는 Amazon S3, DynamoDB 또는 사용자 정의 애플리케이션과 같은 특정 이벤트 소스의 이벤트를 처리합니다.

a. AWS Lambda 콘솔에서 [함수 선택]을 선택합니다.

참고:

생성한 Lambda 함수가 없는 경우에만 콘솔에 이 페이지가 표시됩니다. 함수를 이미 생성했다면 [Lambda > 함수] 페이지가 표시됩니다. 목록 페이지에서 [함수 생성]을 선택하여 [함수 생성] 페이지로 이동합니다.



(확대하려면 클릭)

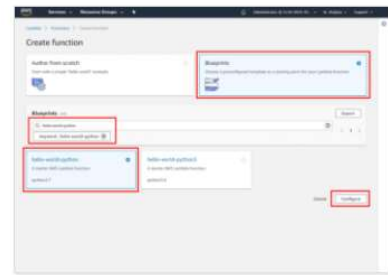
(실제 화면)



b. [블루프린트]를 선택합니다.

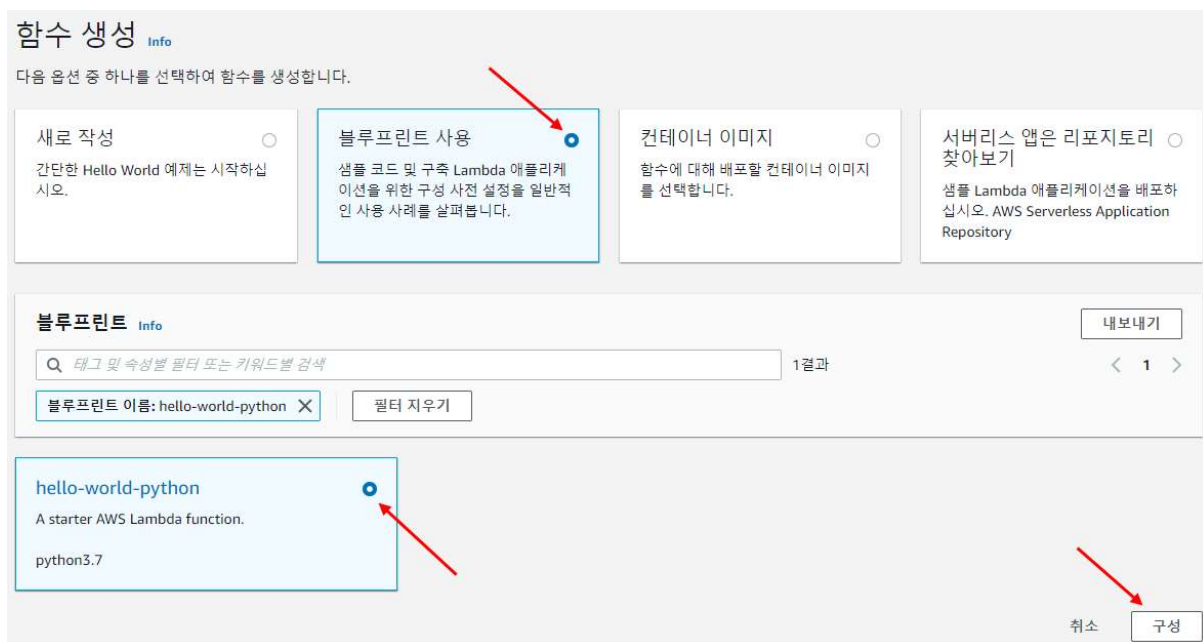
c. 필터 상자에 *hello-world-python*을 입력하고 hello-world-python 블루프린트를 선택합니다.

d. 그런 다음 [구성]을 클릭합니다.



(확대하려면 클릭)

(실제 화면)



### 3단계: Lambda 함수를 구성 및 생성

Lambda 함수는 사용자가 제공한 코드, 관련 종속성 및 구성으로 이루어집니다. 사용자가 제공한 구성 정보에는 할당하려는 컴퓨팅 리소스(메모리 등), 실행 제한 시간, 그리고 사용자 대신 Lambda 함수를 실행할 수 있도록 AWS Lambda가 맡는 IAM 역할이 포함됩니다.

a. Lambda 함수에 대한 [기본 정보]를 입력합니다.

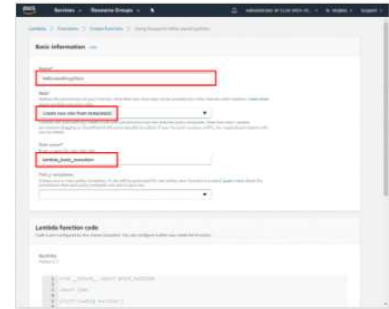
기본 정보:

- **이름:** 여기에서 Lambda 함수의 이름을 지정할 수 있습니다. 본 자습서에서는 *hello-world-python*을 입력합니다.
- **Role:** 사용자 대신 Lambda 함수를 호출하기 위해 AWS Lambda가 맡을 수 있는 필요한 권한을 보유한 IAM 역할(실행 역할이라고 부름)을 생성합니다. [템플릿의 새 역할 생성]을 선택합니다.
- **역할 이름:** *lambda\_basic\_execution*을 입력합니다.

Lambda 함수 코드:

- 이 섹션에서는 Python으로 작성된 예제 코드를 검토할 수 있습니다.

b. 페이지 하단으로 이동하여 [함수 생성]을 선택합니다.

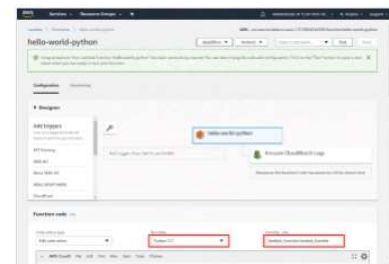


(확대하려면 클릭)

c. **Runtime:** 현재 Lambda 함수 코드를 Java, Node.js, C#, Go 또는 Python으로 작성할 수 있습니다. 이 자습서에서는 *Python 2.7*을 런타임으로 그대로 둡니다.

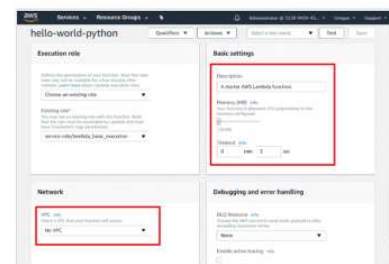
d. **Handler:** AWS Lambda가 코드 실행을 시작할 수 있는 핸들러(코드의 메서드/함수)를 지정할 수 있습니다. AWS Lambda는 이벤트를 처리하는 이 핸들러에 이벤트 데이터를 입력값으로 제공합니다.

본 예제에서는 Lambda가 코드 샘플에서 이를 확인하며, 이는 *lambda\_function.lambda\_handler*를 통해 미리 구성되어 있어야 합니다.



(확대하려면 클릭)

e. 아래로 스크롤하여 메모리, 제한 시간, VPC 설정을 구성합니다. 본 자습서에서는 기본 Lambda 함수 구성 값을 그대로 둡니다.

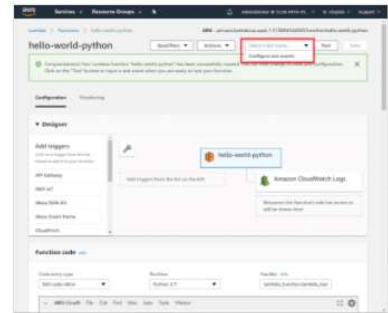


(확대하려면 클릭)

## 4단계: Lambda 함수 호출 및 결과 확인

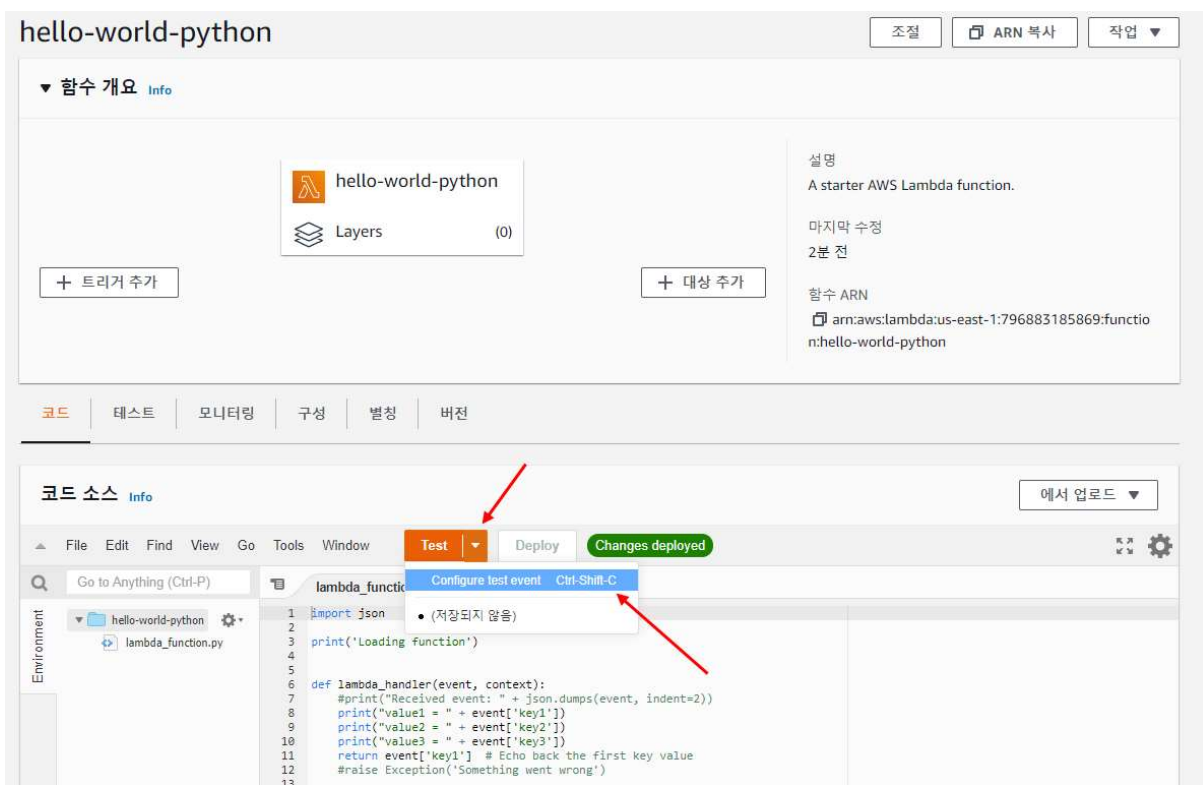
콘솔에 hello-world-python Lambda 함수가 표시됩니다. 이제 함수를 테스트하고, 결과를 확인하며, 로그를 검토할 수 있습니다.

- a. "테스트 이벤트 선택..."이라는 드롭다운 메뉴에서 [테스트 이벤트 구성]을 선택합니다.



(확대하려면 클릭)

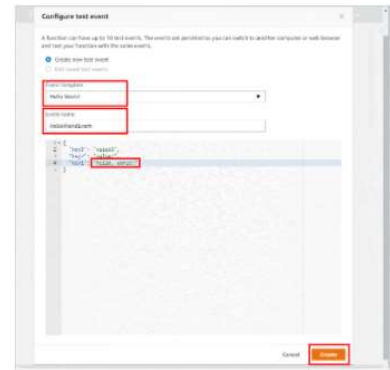
(실제 화면)



b. 이벤트를 입력하여 함수를 테스트할 수 있도록 편집기가 표시됩니다.

- 입력 테스트 이벤트 페이지의 샘플 이벤트 템플릿 목록에서 *Hello World*를 선택합니다.
- 이벤트 이름을 *HelloWorldEvent*처럼 입력합니다.
- 샘플 JSON의 값을 변경할 수 있지만, 이벤트 구조는 변경하지 마십시오. 본 자습서에서는 *value1*을 *hello, world!*로 대체합니다.

[생성]을 선택합니다.



(확대하려면 클릭)

(실제 화면)

테스트 이벤트 구성

함수는 최대 10개의 테스트 이벤트를 가질 수 있습니다. 이 이벤트는 지속되므로 다른 컴퓨터 또는 웹 브라우저로 전환하고 동일한 이벤트로 함수를 테스트할 수 있습니다.

새로운 테스트 이벤트 생성

저장된 테스트 이벤트 편집

이벤트 템플릿

hello-world

이벤트 이름

HelloWorldEvent

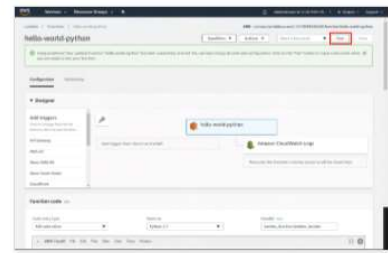
1 {  
2 "key1": "hello, world!",  
3 "key2": "value2",  
4 "key3": "value3"  
5 }

취소

JSON 형식 지정

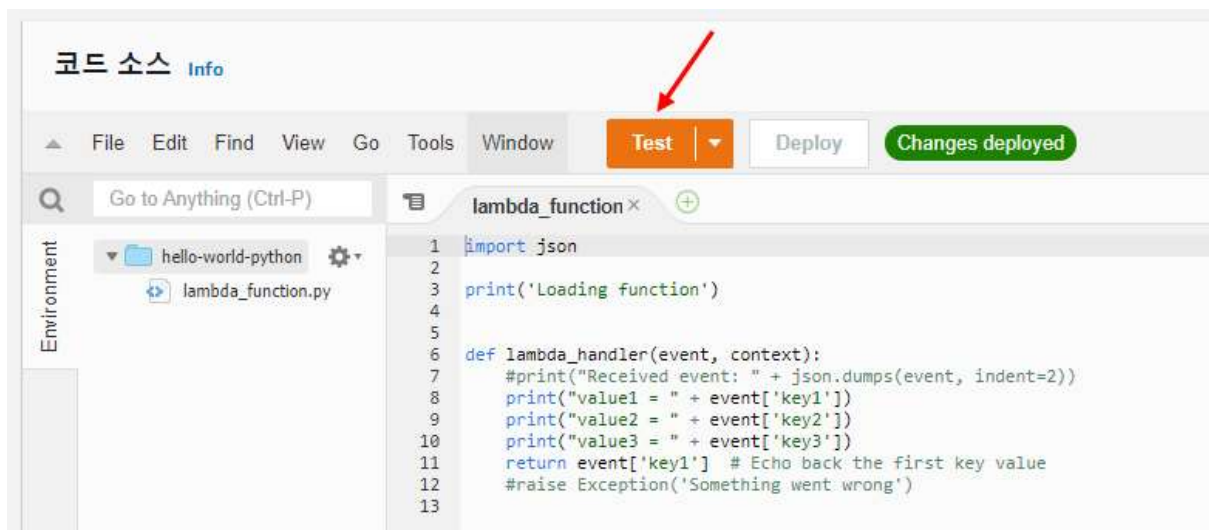
생성

c. [테스트]를 선택합니다.



(확대하려면 클릭)

(실제 화면)





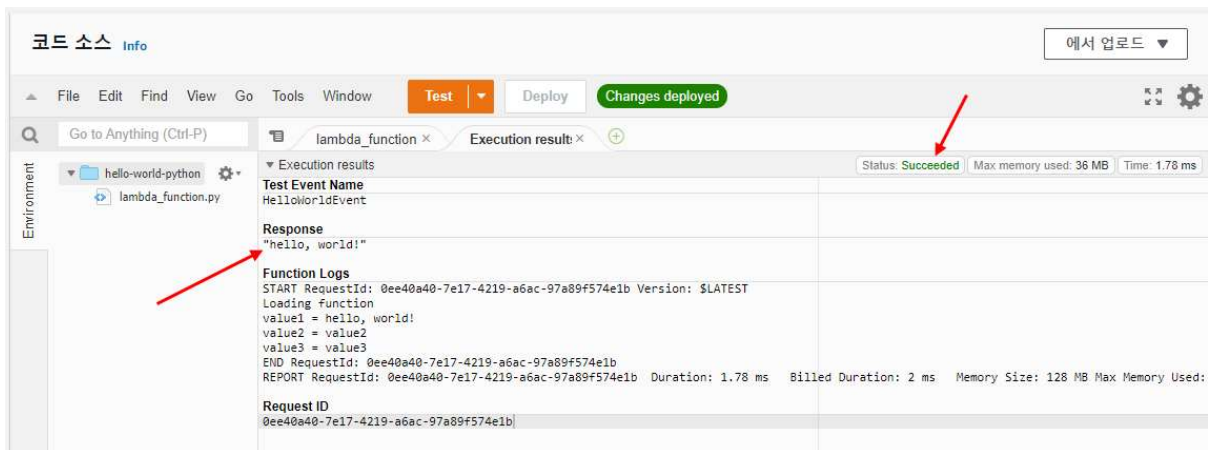
d. 함수가 성공적으로 실행되면, 콘솔에서 결과를 확인합니다.

- [실행 결과] 섹션에서는 실행에 성공했는지 확인합니다.
- [요약] 섹션은 로그 출력에 보고된 주요 정보를 보여줍니다.
- [로그 출력] 섹션은 Lambda 함수 실행으로 생성된 로그를 보여줍니다.



(확대하려면 클릭)

(실제 화면)






문제 1) 답변으로 제출할 캡처 화면: 아래와 같이 화면을 캡처하고 보고서에 첨부하세요


Lambda > 함수 > hello-world-python

## hello-world-python

조절 ARN 복사 작업 ▼

▼ 함수 개요 Info

 hello-world-python

 Layers (0)

+ 트리거 추가 + 대상 추가

설명  
A starter AWS Lambda function.

마지막 수정  
7분 전

함수 ARN  
arn:aws:lambda:us-east-1:796883185869:function:hello-world-python

코드 테스트 모니터링 구성 별칭 버전

### 코드 소스 Info

에서 업로드 ▼

File Edit Find View Go Tools Window Test Deploy Changes deployed

Go to Anything (Ctrl-P)

Environment

- hello-world-python
- lambda\_function.py

Execution results

Status: Succeeded Max memory used: 36 MB Time: 1.78 ms

Test Event Name  
HelloWorldEvent

Response  
"hello, world!"

Function Logs  
START RequestId: 0ee40a40-7e17-4219-a6ac-97a89f574e1b Version: \$LATEST  
Loading function  
value1 = hello, world!  
value2 = value2  
value3 = value3  
END RequestId: 0ee40a40-7e17-4219-a6ac-97a89f574e1b  
REPORT RequestId: 0ee40a40-7e17-4219-a6ac-97a89f574e1b Duration: 1.78 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used:

Request ID  
0ee40a40-7e17-4219-a6ac-97a89f574e1b

## 5단계: 지표 모니터링

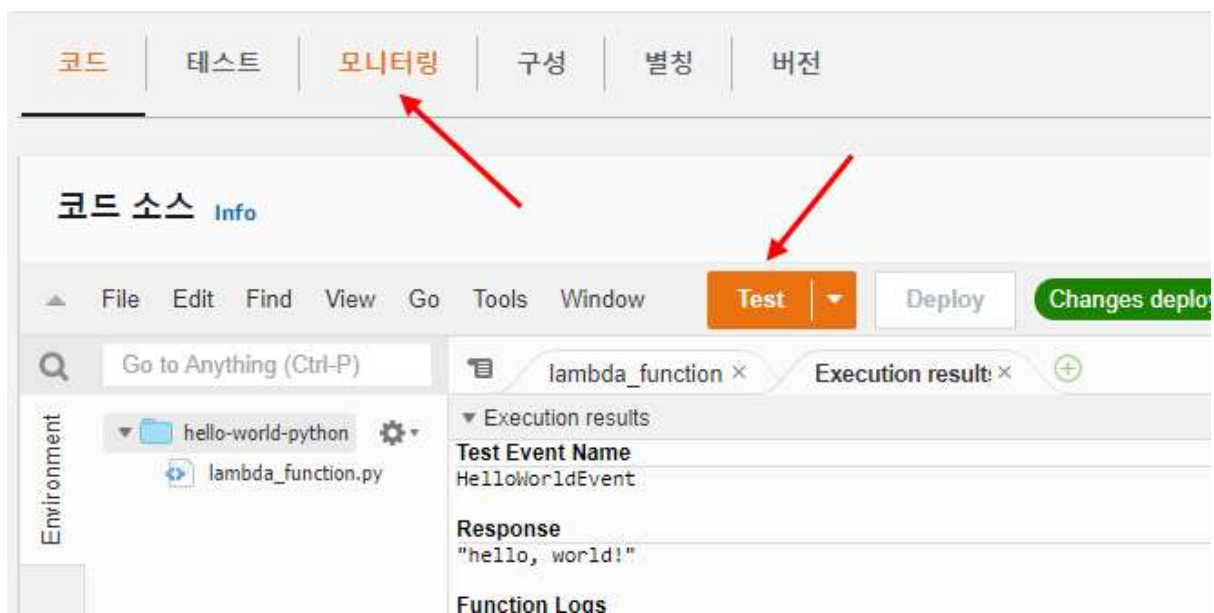
AWS Lambda는 Lambda 함수를 자동으로 모니터링하고 Amazon CloudWatch를 통해 지표를 보고합니다. 코드가 실행됨에 따라 이를 모니터링하는 데 도움이 되도록, Lambda에서는 요청 수, 요청당 지연 시간 및 오류가 발생한 요청 수를 자동으로 추적하고, 관련 지표를 게시합니다.

- [테스트] 버튼을 반복해서 클릭하여 Lambda 함수를 몇 번 더 호출합니다. 이를 통해 지표가 생성되며, 다음 단계에서 확인할 수 있습니다.
- [모니터링]을 선택하여 결과를 확인합니다.



(확대하려면 클릭)

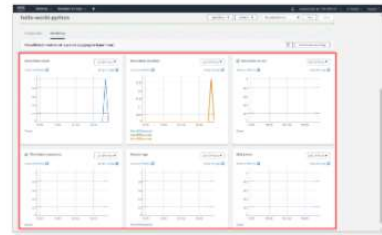
'Test' 버튼을 두 번 더 클릭하고, '모니터링' 탭 클릭



c. 아래로 스크롤하여 Lambda 함수의 지표를 확인합니다. Lambda 지표는 Amazon CloudWatch를 통해 보고됩니다. 이 지표를 활용하여 사용자 정의 경보를 설정할 수 있습니다. CloudWatch에 대한 자세한 내용은 [Amazon CloudWatch 개발자 안내서](#)를 참조하십시오.

모니터링 탭에서 CloudWatch 지표, 즉 호출 수, 호출 기간, 호출 오류, 제한된 호출, 반복자 경과 시간 및 DLQ 오류를 볼 수 있습니다.

AWS Lambda를 사용하면 사용한 만큼만 비용을 지불합니다. AWS Lambda 프리 티어 한도를 초과하면, 함수에 대한 요청 수(호출 수)와 코드가 실행된 시간(호출 기간)을 기준으로 비용이 부과됩니다. 자세한 내용은 [AWS Lambda 요금](#)을 참조하십시오.



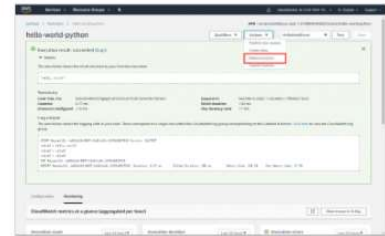
(확대하려면 클릭)

\*\* 참고: AWS Free Tier 계정인 경우, 매월 100만건까지 무료로 람다 요청을 보낼 수 있음

## 6단계: Lambda 함수 삭제

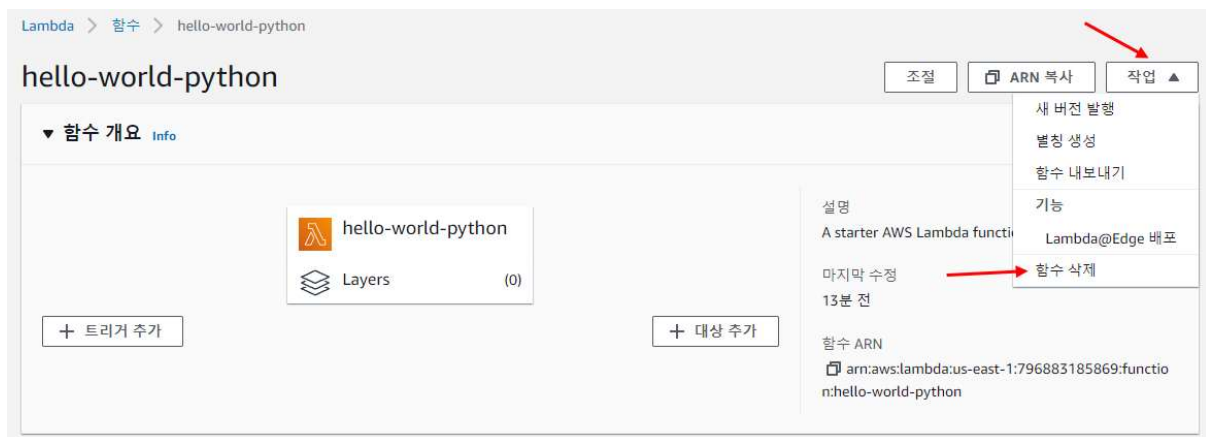
Lambda 함수를 유지하는 데 비용이 부과되지 않으므로 AWS Lambda 콘솔에서 함수를 손쉽게 삭제할 수 있습니다.

- a. [작업] 버튼을 선택하고, [함수 삭제]를 클릭합니다.



(확대하려면 클릭)

(실제화면)



- b. 종료할지 여부를 묻는 메시지가 표시되면 [삭제]를 선택합니다.



(확대하려면 클릭)

## [Q1] AWS Lambda(람다) 를 사용하여 Server-Less 서비스 개발: 파파고 연동하기

<https://developers.naver.com/docs/papago/papago-nmt-overview.md> 를 참고하여 '오픈 API 이용 신청'

→ ↺ ↻

https://developers.naver.com/docs/papago/papago-nmt-overview.md

🔍 📄 📌 ⬇

NAVER Developers

Products Documents Application NAVER D2 Support Forum

API 상태

Search Here

로그인

파파고

Papago 번역

개요

API 레퍼런스

구현 예제

언어 감지

한글 인명-로마자 변환

사전 준비 사항

Papago 번역을 사용하려면 먼저 [네이버 개발자 센터](#)에서 애플리케이션을 등록하고 클라이언트 아이디와 클라이언트 시크릿을 발급받아야 합니다.

클라이언트 아이디와 클라이언트 시크릿은 인증된 사용자인지를 확인하는 수단이며, 애플리케이션이 등록되면 발급됩니다. 클라이언트 아이디와 클라이언트 시크릿을 네이버 오픈API를 호출할 때 HTTP 헤더에 포함해서 전송해야 API를 호출할 수 있습니다. API사용량은 클라이언트 아이디별로 합산됩니다.

주의

네이버에 로그인한 사용자 계정으로 애플리케이션이 등록됩니다. 애플리케이션을 등록한 네이버 아이디는 '관리자' 권한을 가지게 되므로 네이버 계정의 보안에 각별히 주의해야 합니다.

회사나 단체에서 애플리케이션을 등록할 때는 추후 키 관리 등이 용이하도록 네이버 단체 회원으로 로그인해 이용할 것을 권장합니다.

- [네이버 단체 회원 가입하기](#)

애플리케이션 등록

네이버 개발자 센터에서 애플리케이션을 등록하는 방법은 다음과 같습니다.

- 네이버 개발자 센터의 메뉴에서 [Application > 애플리케이션 등록](#)을 선택합니다.
- 이용약관 동의 단계에서 [이용약관에 동의합니다](#).를 선택한 다음 [확인](#)을 클릭합니다.
- 계정 정보 등록 단계에서 휴대폰 인증을 완료하고 회사 이름을 입력한 다음 [확인](#)을 클릭합니다. 휴대폰 인증은 담당자 연락처 확인을 위해 필요한 과정이며, 애플리케이션을 처음 등록할 때 한 번만 인증받으면 됩니다.
- 애플리케이션 등록 (API이용신청) 페이지에서 [애플리케이션 등록 세부 정보](#)를 입력한 다음 등록하기를 클릭합니다.

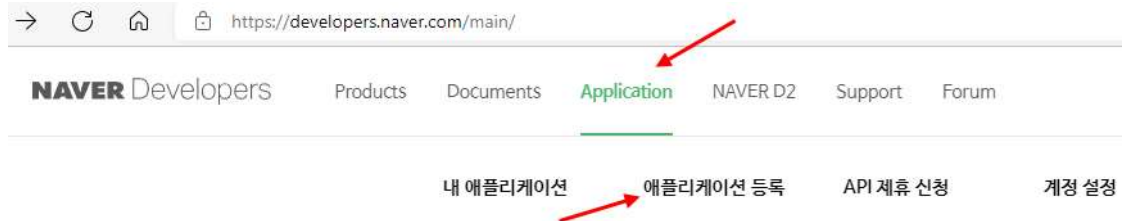
애플리케이션 등록 세부 정보

애플리케이션 등록 (API이용신청) 페이지에서 애플리케이션 세부 정보를 입력하는 방법은 다음과 같습니다.

- 등록하려는 애플리케이션의 이름을 [애플리케이션 이름](#)에 입력합니다. 최대 40자까지 입력할 수 있습니다.
- 사용 API에서 [Papago 번역](#)을 선택해 추가합니다.
- [비로그인 오픈 API 서비스 환경](#)에서 애플리케이션을 서비스할 환경을 추가하고 필요한 상세 정보를 입력합니다.

<방법>

1. 네이버 개발자 센터(<https://developers.naver.com/main/>) 로그인
2. 화면 상단의 'Application' 메뉴에서 '애플리케이션 등록' 클릭



3. 아래와 같이 애플리케이션 등록하기  
**애플리케이션 등록 (API 이용신청)**

애플리케이션의 기본 정보를 등록하면, 좌측 **내 애플리케이션** 메뉴의 서브 메뉴에 등록하신 애플리케이션 이름으로 서브 메뉴가 만들어집니다.

애플리케이션 이름	<input type="text" value="papago-trans"/> <ul style="list-style-type: none"><li>• 네이버 아이디로 로그인할 때 사용자에게 표시되는 이름이므로 서비스 브랜드를 대표할 수 있는 이름으로 가급적 10자 이내로 간결하게 설정해주세요.</li><li>• 40자 이내의 영문, 한글, 숫자, 공백문자, "-", "_" 만 입력 가능합니다.</li></ul>
사용 API	<div>선택하세요. ▼ ✓</div> <div>Papago 번역 ×</div>
비로그인 오픈 API 서비스 환경	<div>환경 추가 ▼</div> <div>WEB 설정 × ^</div> <div>웹 서비스 URL (최대 10개)</div> <div><input type="text" value="https://console.aws.amazon.com"/> + ✓</div> <ul style="list-style-type: none"><li>• 텍스트 폼 우측 끝의 '+' 버튼을 누르면 행이 추가되며, '-' 버튼을 누르면 행이 삭제됩니다.</li><li>• http와 https는 구분하지 않습니다.</li><li>• www는 빼고 입력해 주세요. 예) http://naver.com</li><li>• 서브 도메인이 있으면 대표 도메인명만 입력해 주세요. (예: http://naver.com)</li><li>• 하이브리드 앱은 location.href 객체 출력 값을 입력하면 됩니다. (예: file://로컬 URI)</li></ul>

4. '등록하기' 클릭 후, Client ID와 Client Secret 확인하기  
papago-trans

개요	API 설정	멤버관리
----	--------	------

#### 애플리케이션 정보

Client ID	ZBkJAhBpB9E
Client Secret	..... <a href="#">보기</a>



파파고 번역 구현예제 코드를 <https://developers.naver.com/docs/papago/papago-nmt-example-code.md> 에서 확인

→ ↺ 🏠 📑

https://developers.naver.com/docs/papago/papago-nmt-example-code.md#python

NAVER Developers

Products Documents Application NAVER D2 Support Forum

파파고

Papago 번역

개요

API 레퍼런스

구현 예제

언어 감지

한글 인명-로마자 변환

Python

```
import os
import sys
import urllib.request
client_id = "YOUR_CLIENT_ID" # 개발자센터에서 발급받은 Client ID 값
client_secret = "YOUR_CLIENT_SECRET" # 개발자센터에서 발급받은 Client Secret 값
encText = urllib.parse.quote("반갑습니다")
data = "source=ko&target=en&text=" + encText
url = "https://openapi.naver.com/v1/papago/n2mt"
request = urllib.request.Request(url)
request.add_header("X-Naver-Client-Id",client_id)
request.add_header("X-Naver-Client-Secret",client_secret)
response = urllib.request.urlopen(request, data=data.encode("utf-8"))
rescode = response.getcode()
if(rescode==200):
    response_body = response.read()
    print(response_body.decode('utf-8'))
else:
    print("Error Code:" + rescode)
```

- [GitHub에서 보기](#)

직전의 람다 사용 예제와 같이 'hello-world-python' 블루 프린트를 생성하고, 함수 이름을 papago-trans 로 설정, 그리고 함수 생성

Lambda > 함수 > 함수 생성 > 블루프린트 hello-world-python 구성

### 기본 정보 [Info](#)

함수 이름

실행 역할

함수에 대한 권한을 정의하는 역할을 선택합니다. 사용자 지정 역할을 생성하려면 [IAM 콘솔](#)로 이동합니다.

☐ 기본 Lambda 권한을 가진 새 역할 생성

☐ 기존 역할 사용

☒ AWS 정책 템플릿에서 새 역할 생성

**i** 역할을 생성하는 데 몇 분 정도 걸릴 수 있습니다. 역할을 삭제하거나 이 역할에서 신뢰 또는 권한 정책을 편집하지 마십시오.

역할 이름

새 역할의 이름을 입력합니다.

공백 없이 문자, 숫자, 하이픈 또는 밑줄만 사용합니다.

정책 템플릿 - 선택 사항 [Info](#)

정책 템플릿을 하나 이상 선택합니다.

함수 생성 후, 코드 소스를 아래와 같이 수정, 그리고 'Deploy' 버튼 클릭

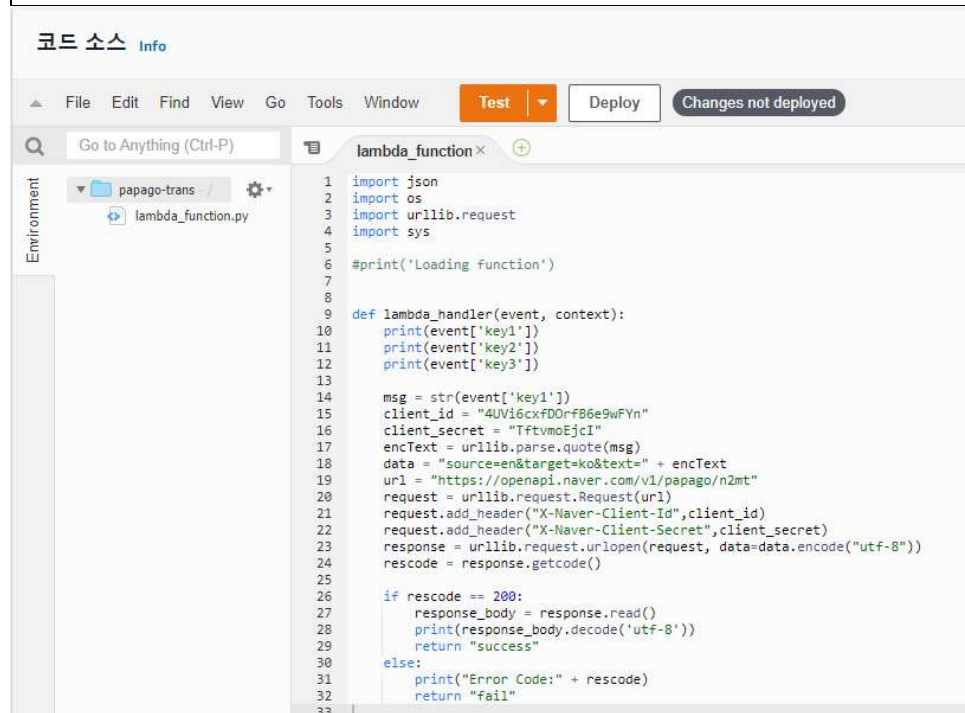
```
import json
import os
import urllib.request
import sys

#print('Loading function')

def lambda_handler(event, context):
    print(event['key1'])
    print(event['key2'])
    print(event['key3'])

    msg = str(event['key1'])
    client_id = "YOUR_CLIENT_ID" # 개발자센터에서 발급받은 Client ID 값
    client_secret = "YOUR_CLIENT_SECRET" # 개발자센터에서 발급받은 Client Secret 값
    encText = urllib.parse.quote(msg)
    data = "source=en&target=ko&text=" + encText
    url = "https://openapi.naver.com/v1/papago/n2mt"
    request = urllib.request.Request(url)
    request.add_header("X-Naver-Client-Id",client_id)
    request.add_header("X-Naver-Client-Secret",client_secret)
    response = urllib.request.urlopen(request, data=data.encode("utf-8"))
    rescode = response.getcode()

    if rescode == 200:
        response_body = response.read()
        print(response_body.decode('utf-8'))
        return "success"
    else:
        print("Error Code:" + rescode)
        return "fail"
```



'Translate'라는 이름으로 새로운 테스트 이벤트 생성

## 테스트 이벤트 구성

함수는 최대 10개의 테스트 이벤트를 가질 수 있습니다. 이 이벤트는 지정된 이벤트로 함수를 테스트할 수 있습니다.

- ☒ 새로운 테스트 이벤트 생성
- ☐ 저장된 테스트 이벤트 편집

이벤트 템플릿

hello-world

이벤트 이름

Translate

```
1 {  
2   "key1": "I like an apple",  
3   "key2": "Taewoon Kim",  
4   "key3": "Hallym Univ"  
5 }
```

'테스트' 탭으로 이동하고 '테스트' 버튼 클릭하여 테스트 실행

The screenshot shows a web interface for testing. At the top, there is a navigation bar with tabs: '코드' (Code), '테스트' (Test), '모니터링' (Monitoring), '구성' (Configuration), '별칭' (Alias), and '버전' (Version). The '테스트' tab is selected, indicated by a red arrow. Below the navigation bar, a green banner displays a success message: '실행 결과: 성공 (로그)' (Execution result: Success (Log)), with a link to '세부 정보' (Detailed information). To the right of the banner is a close button 'X'. Below the banner, the '테스트 이벤트' (Test Event) section contains buttons for '삭제' (Delete), '형식' (Format), '변경 사항 저장' (Save changes), and '테스트' (Test). The '테스트' button is highlighted with a red arrow. Below these buttons, there is a text instruction: '테스트 이벤트를 사용하여 함수를 호출합니다. 함수를 트리거하는 서비스와 일치하는 템플릿을 선택하거나 JSON에 이벤트 문서를 입력합니다.' (Use test events to call functions. Select a template that matches the service that triggers the function, or enter an event document in JSON). Below this instruction are two radio buttons: '새 이벤트' (New event) and '저장된 이벤트' (Saved event), with the latter being selected. Below the radio buttons is a dropdown menu labeled 'Translate' and a refresh icon. At the bottom, there is a code editor showing a JSON object: 

```
1 {  
2   "key1": "I like an apple.",  
3   "key2": "Taewoon Kim",  
4   "key3": "Hallym Univ"  
5 }
```

로그 '세부정보' 클릭하여 상세 로그 확인하고, 영어 문장이 한글로 번역 되었는지 확인

코드 | **테스트** | 모니터링 | 구성 | 별칭 | 버전

✓ 실행 결과: 성공 (로그) ✕

▶ 세부 정보

**테스트 이벤트** [삭제] [형식] [변경 사항 저장] **테스트**

테스트 이벤트를 사용하여 함수를 호출합니다. 함수를 트리거하는 서비스와 일치하는 템플릿을 선택하거나 JSON에 이벤트 문서를 입력합니다.

☐ 새 이벤트  
☒ 저장된 이벤트

저장된 이벤트

Translate [▼] ↻

```
1 {  
2   "key1": "I like an apple.",  
3   "key2": "Taewoon Kim",  
4   "key3": "Hallym Univ"  
5 }
```

문제 2) 답변으로 제출할 캡처 화면: 아래와 같이 화면을 캡처하고 보고서에 첨부하세요

✓ 실행 결과: 성공 (로그) ✕

▼ 세부 정보

아래 영역은 함수 실행에서 반환한 결과를 보여줍니다. 함수에서 반환하는 결과에 대해 자세히 알아보세요.

"success"

**요약**

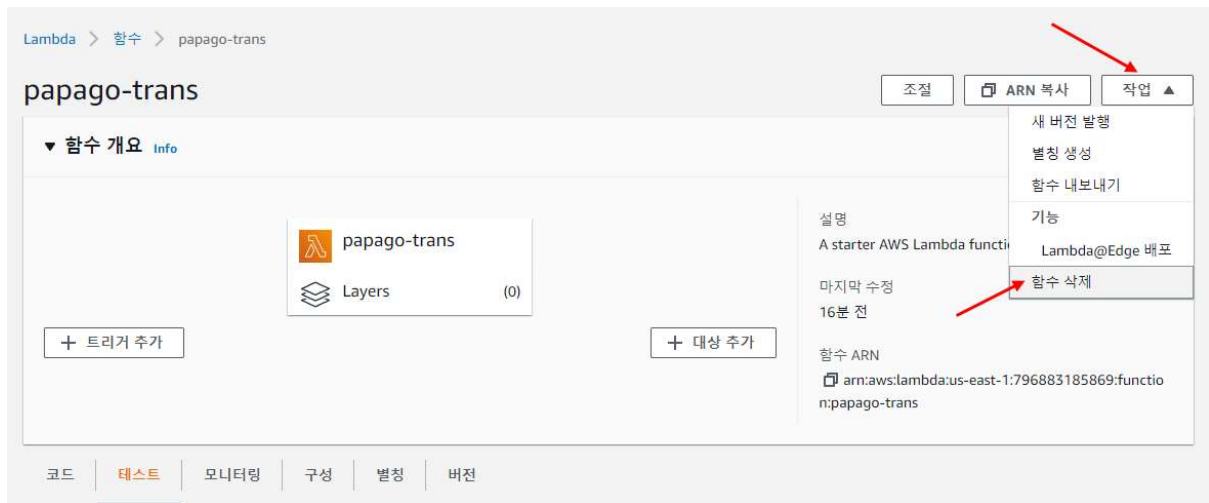
코드 SHA-256	요청 ID
y6pa+pKYpLoaK5BKlbXgePgMsT58X9nwEaNPV3HEr4o=	aaa9ce5e-cd8f-4328-b63c-94a943048bb9
시작 시간	기간
144.30 ms	1105.70 ms
청구 시간	리소스 구성
1106 ms	128 MB
사용된 최대 메모리	
39 MB	

**로그 출력**

아래 영역은 코드의 로깅 호출을 보여줍니다. 연관된 CloudWatch 로그 그룹을 보려면 [여기를 클릭](#)(을) 하십시오.

```
START RequestId: aaa9ce5e-cd8f-4328-b63c-94a943048bb9 Version: $LATEST  
I like an apple.  
Taewoon Kim  
Hallym Univ  
RETURN : {"message":{"@type":"response","@service":"naverservice.nmt.proxy","@version":"1.0.0","result":{"srcLangType":"en","tarLangType":"ko","translatedText":"나는 사과를 좋아한다.","engineType":"N2MT","pivot":null}}}  
END RequestId: aaa9ce5e-cd8f-4328-b63c-94a943048bb9  
REPORT RequestId: aaa9ce5e-cd8f-4328-b63c-94a943048bb9 Duration: 1105.70 ms Billed Duration: 1106 ms Memory Size: 128 MB  
Max Memory Used: 39 MB Init Duration: 144.30 ms
```

## 람다 함수를 삭제





## 지금까지 사용한 비용 확인

The screenshot displays the AWS Billing console interface. At the top, the navigation bar includes a search bar, the user's name 'Taewoon', and a dropdown menu. The dropdown menu is open, showing options: '내 계정 796883185869', '내 조직', '내 Service Quotas', '내 결제 대시보드' (highlighted with a red arrow), '내 보안 자격 증명', and '로그아웃'. The main heading is '대금 및 비용 관리 대시보드'. Below this, there's a section for 'AWS 대금 및 비용 관리 시작하기' with several links. A '소비 요약' section shows a 'Cost Explorer' button. Below that, a message states that the bill is \$0. A large '0 KRW' is displayed with a red arrow pointing to it, and a note mentions a total of 1,180.08 KRW across various currencies. On the right, a donut chart shows '\$0'. At the bottom, a bar chart shows '지불할 금액 없음' (No amount to be paid) with a red arrow pointing to it, and a value of '\$0.00' is shown on the right.

서비스별 이번 달

아래 차트로는 각 서비스

내 계정 796883185869

내 조직

내 Service Quotas

내 결제 대시보드

내 보안 자격 증명

로그아웃

AWS 대금 및 비용 관리 시작하기

- AWS Budgets을(를) 이용하여 비용 및 사용량 관리
- Cost Explorer을(를) 통해 비용 요인 및 사용 추세 시각화
- Athena 통합에서 비용 및 사용 보고서를(를) 사용하여 비용 심층 분석
- 자세히 알아보기: AWS 새로운 소식 페이지에 대해 자세히 알아보십시오.

예약 인스턴스(RI)가 있습니까?

- Cost Explorer.을(를) 통해 RI 사용률 및 범위 보고서(RI 구매 추천 포함)에 액세스

소비 요약

Cost Explorer

AWS 계정 결제 콘솔입니다. 지난달 비용과 이번 달 현재까지 비용이 아래에 나타납니다.

현재 당월 누적잔액 산정 대상: 2021년 11월, 결제 통화의 환율로 산출되었습니다. 0.00 USD 청구서에 표시된 것과 같이

0 KRW

여러 가지 통화의 1,180.08

\$4

\$3

지불할 금액 없음

\$0.00