



사용자 가이드

# AWS VS Code용 툴킷



# AWS VS Code용 툴킷: 사용자 가이드

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon의 상표 및 브랜드 디자인은 Amazon 외 제품 또는 서비스와 함께, 브랜드 이미지를 떨어뜨리거나 고객에게 혼동을 일으킬 수 있는 방식으로 사용할 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 Amazon 계열사, 관련 업체 또는 Amazon의 지원 업체 여부에 상관없이 해당 소유자의 자산입니다.

# Table of Contents

AWS Toolkit for Visual Studio Code .....	1
무엇입니까 AWS Toolkit for Visual Studio Code .....	1
관련 정보 .....	1
아마존 Q 디벨로퍼와 아마존 CodeWhisperer .....	2
Toolkit 다운로드 .....	3
VS Code Marketplace에서 도구 키트 다운로드 .....	3
AWS 추가 IDE 도구 키트 .....	3
시작하기 .....	4
Toolkit for VS Code 설치 .....	4
필수 조건 .....	4
AWS Toolkit for Visual Studio Code 다운로드 및 설치 .....	4
추가 필수 조건 .....	5
연결 대상 AWS .....	6
필수 조건 .....	6
로그인 패널 열기 .....	6
AWS 툴킷에서 연결 .....	7
아마존용 인증 CodeCatalyst .....	8
AWS 지역 변경 .....	9
AWS 탐색기에 지역 추가 .....	9
AWS 탐색기에서 지역 숨기기 .....	9
툴체인 구성 .....	9
.NET Core 툴체인 구성 .....	10
Node.js 툴체인 구성 .....	10
Python 툴체인 구성 .....	10
Java 툴체인 구성 .....	11
Go 툴체인 구성하기 .....	11
툴체인 사용 .....	11
인증 및 액세스 .....	12
IAM Identity Center .....	12
IAM 보안 인증 .....	12
IAM 사용자 생성 .....	13
에서 공유 자격 증명 파일 생성 AWS Toolkit for Visual Studio Code .....	14
자격 증명 프로필 추가 .....	15
AWS 빌더 ID .....	15

외부 보안 인증 프로세스 사용 .....	16
다음과 함께 작업하기 AWS .....	17
실험 기능 .....	18
AWS 익스플로러 .....	18
Amazon CodeCatalyst .....	19
Amazon CodeCatalyst란? .....	19
CodeCatalyst 시작하기 .....	20
CodeCatalyst 작업 .....	22
개발 환경과 작업 .....	26
문제 해결 .....	29
Amazon API Gateway .....	30
AWS App Runner .....	31
필수 조건 .....	31
요금 .....	34
App Runner 서비스 생성 .....	34
App Runner 서비스 관리 .....	37
AWS Application Composer .....	40
AWS Application Composer로 작업 .....	40
AWS CDK .....	41
AWS CDK 애플리케이션 .....	42
AWS CloudFormation 스택 .....	44
AWS CloudFormation 스택 삭제 .....	44
CloudFormation 템플릿 만들기 .....	45
Amazon CloudWatch Logs .....	46
CloudWatch 로그 그룹 및 로그 스트림 보기 .....	47
CloudWatch 로그 이벤트 작업 .....	48
로그 그룹 검색 .....	50
Amazon ECR .....	52
필수 조건 .....	52
VS Code용 도구 키트와 함께 Amazon ECR 사용 .....	54
Amazon ECS .....	62
작업 정의 파일에 IntelliSense 사용 .....	63
Amazon ECS Exec .....	64
Amazon EventBridge .....	66
Amazon EventBridge 스키마로 작업하기 .....	66
AWS IAM 액세스 분석기 .....	68

AWS IAM 액세스 분석기 사용 .....	69
AWS IoT .....	72
AWS IoT 필수 조건 .....	72
AWS IoT 사물 .....	72
AWS IoT 인증서 .....	74
AWS IoT 정책 .....	77
AWS Lambda 함수 .....	80
원격 Lambda 함수와 상호 작용 .....	80
Amazon Redshift .....	85
Amazon Redshift 작업 .....	86
Amazon S3 .....	90
S3 리소스 사용 .....	90
객체 작업 .....	92
AWS 서버리스 애플리케이션 .....	96
시작하기 .....	96
코드에서 Lambda 함수 실행 및 디버깅 .....	103
로컬 Amazon API Gateway 리소스 실행 및 디버깅 .....	107
서버리스 애플리케이션 디버깅을 위한 구성 옵션 .....	110
문제 해결 .....	117
AWS Systems Manager .....	119
전제 조건 및 필수 조건 .....	119
Systems Manager Automation 문서에 대한 IAM 권한 .....	120
새 Systems Manager 자동화 문서 생성 .....	120
기존 Systems Manager 자동화 문서 열기 .....	121
Systems Manager 자동화 문서 수정 .....	122
Systems Manager 자동화 문서 게시 .....	122
Systems Manager 자동화 문서 삭제 .....	123
Systems Manager 자동화 문서 실행 .....	123
문제 해결 .....	124
AWS Step Functions .....	124
AWS Step Functions 및 VS 코드 .....	125
위협 컴포저 .....	134
위협 컴포저 사용 .....	135
리소스 .....	136
리소스 액세스를 위한 IAM 권한 .....	137
기존 리소스 추가 및 상호 작용 .....	137

---

리소스 생성 및 업데이트 .....	139
보안 .....	141
데이터 보호 .....	141
사용 설명서 기록 .....	142
.....	cxlvi

# AWS Toolkit for Visual Studio Code

이것은 VS Code용 AWS 툴킷 사용 설명서입니다. AWS Toolkit for Visual Studio를 찾는 경우 [사용 설명서](#)를 참조하세요.

## 무엇입니까 AWS Toolkit for Visual Studio Code

VS Code용 도구 키트는 Visual Studio Code (VS Code) 편집기용 오픈 소스 확장 프로그램입니다. 이 확장 프로그램을 사용하면 개발자는 Amazon Web Services (AWS)를 사용하는 서버리스 애플리케이션을 쉽게 개발하고 로컬 디버깅 및 배포할 수 있습니다.

주제

- [AWS Toolkit for Visual Studio Code 시작하기](#)
- [AWS 서비스 및 도구 사용](#)

## 관련 정보

도구 키트의 소스 코드가 필요하거나 현재 해결되지 않은 문제를 보려면 다음 리소스를 이용하십시오.

- [소스 코드](#)
- [문제 추적기](#)

Visual Studio Code 편집기에 대해 자세히 알아보려면 <https://code.visualstudio.com/>을 방문하세요.

# 아마존 Q 디벨로퍼와 아마존 CodeWhisperer

2024년 4월 30일부터 CodeWhisperer Amazon은 이제 Amazon Q Developer의 일원이 되었으며, 여기에는 인라인 코드 제안 및 Amazon Q 개발자 보안 스캔이 포함됩니다. 시작하려면 [VS Code Marketplace](#)에서 [Amazon Q Developer IDE 확장 프로그램](#)을 다운로드하십시오.

Amazon Q 개발자 서비스에 대한 자세한 내용은 [Amazon Q 개발자](#) 사용 설명서를 참조하십시오. Amazon Q 요금제 및 가격에 대한 자세한 내용은 [Amazon Q 요금](#) 안내서를 참조하세요.



# Toolkit for VS Code 다운로드

IDE의 VS Code Marketplace를 통해 AWS Toolkit for Visual Studio Code를 다운로드, 설치 및 설정할 수 있습니다. 자세한 지침은 본 사용자 설명서의 시작하기에 있는 [다운로드 및 설치](#)를 참조하세요.

## VS Code Marketplace에서 도구 키트 다운로드

또는 웹 브라우저에서 [VS Code Marketplace](#)로 이동하여 AWS Toolkit for Visual Studio Code 설치 파일을 다운로드할 수 있습니다.

## AWS 추가 IDE 도구 키트

AWS Toolkit for Visual Studio Code 뿐만 아니라 AWS 또한 JetBrains 및 Visual Studio용 IDE Toolkits를 제공합니다.

### AWS Toolkit for JetBrains 링크

- 이 링크를 통해 JetBrains Marketplace로 가서 [AWS Toolkit for JetBrains를 다운로드하세요](#).
- AWS Toolkit for JetBrains에 대한 자세한 내용은 [사용 설명서](#)를 참조하세요.

### Toolkit for Visual Studio 링크

- 이 링크를 통해 Visual Studio Marketplace로 가서 [Toolkit for Visual Studio를 다운로드](#)하세요.
- Toolkit for Visual Studio에 대한 자세한 내용은 [Toolkit for Visual Studio](#) 사용자 설명서를 참조하세요.

# AWS Toolkit for Visual Studio Code 시작하기

AWS Toolkit for Visual Studio Code를 통해 VS Code 통합 개발 환경(IDE)에서 AWS 서비스와 리소스를 사용할 수 있습니다.

다음 항목에서는 AWS Toolkit for Visual Studio Code 설정, 설치 및 구성 방법을 설명합니다.

## 주제

- [AWS Toolkit for Visual Studio Code 설치](#)
- [연결 대상 AWS](#)
- [AWS 지역 변경](#)
- [툴체인 구성](#)

## AWS Toolkit for Visual Studio Code 설치

### 필수 조건

VS Code에서 AWS Toolkit for Visual Studio Code로 작업하려면 다음 전제 조건을 충족해야 합니다. AWS Toolkit for Visual Studio Code에서 사용할 수 있는 모든 AWS 서비스와 리소스에 액세스하는 방법에 대한 자세한 내용은 본 설명서에서 [the section called “추가 필수 조건”](#) 섹션을 참조하십시오.

- Windows, macOS, Linux 운영 체제에서 VS Code를 사용할 수 있습니다.
- VS Code 1.42.0 이상 버전에서 AWS Toolkit for Visual Studio Code를 사용할 수 있습니다.

VS Code에 대한 자세한 정보를 확인하거나 VS Code 최신 버전을 다운로드하려면 [VS Code downloads](#) 웹 사이트를 참조하십시오.

### AWS Toolkit for Visual Studio Code 다운로드 및 설치

IDE의 VS Code Marketplace에서 AWS Toolkit for Visual Studio Code를 다운로드, 설치 및 설정할 수 있습니다. 또는 웹 브라우저에서 [VS Code Marketplace](#)로 이동하여 AWS Toolkit for Visual Studio Code 설치 파일을 다운로드할 수 있습니다.

VS Code IDE Marketplace에서 AWS Toolkit for Visual Studio Code 설치

1. [VS Code 마켓플레이스](#) 링크를 클릭하여 VS Code IDE에서 AWS Toolkit for Visual Studio Code 확장 프로그램을 여세요.

**Note**

VS Code가 컴퓨터에서 실행되지 않다면 VS Code가 로드되는 데 잠시 시간이 걸릴 수 있습니다.

2. VS Code Marketplace의 AWS Toolkit for Visual Studio Code 확장 프로그램에서 설치를 선택하여 설치를 시작하세요.
3. 메시지가 표시된 후, VS Code를 재시작하면 설치 프로세스가 완료됩니다.

## 추가 필수 조건

AWS Toolkit for Visual Studio Code의 특정 기능을 사용하려면 다음이 있어야 합니다.

- Amazon Web Services (AWS) 계정: AWS 계정이 있어야 AWS Toolkit for Visual Studio Code를 사용할 수 있는 것은 아니지만 계정이 없으면 사용할 수 있는 기능이 크게 제한됩니다. AWS 계정을 생성하려면 [AWS 홈페이지](#)로 이동하세요. (이전에 사이트를 방문한 적이 있는 경우) AWS Account 만들기 또는 가입 완료를 선택합니다.
- Code Development - 사용할 언어 관련 SDK입니다. 다음 링크에서 다운로드하거나 선호하는 패키지 관리자를 사용하세요.
  - .NET SDK: <https://dotnet.microsoft.com/download>
  - Node.js SDK: <https://nodejs.org/en/download>
  - Python SDK: <https://www.python.org/downloads>
  - Java SDK: <https://aws.amazon.com/corretto/>
  - Go SDK: <https://golang.org/doc/install>
- AWS SAM CLI - AWS CLI 도구로 로컬 서버리스 애플리케이션 개발, 테스트 및 분석을 할 수 있습니다. 이는 도구 키트를 설치하는 데 필요하지 않습니다. 그러나 [새 서버리스 애플리케이션 생성 \(로컬\)](#)과 같은 AWS Serverless Application Model(AWS SAM) 기능에 필요하므로 CLI(및 도구, 다음에 설명)를 설치하는 것이 좋습니다.

자세한 내용은 [AWS Serverless Application Model 개발자 가이드](#)에서 AWS SAM CLI 설치를 참조하세요.

- Docker - AWS SAM CLI는 이 오픈 소스 소프트웨어 컨테이너 플랫폼에서 실행합니다. 자세한 정보를 참조하거나 지침을 다운로드하려면 [Docker](#)를 참조하세요.
- Package Manager — 애플리케이션 코드를 다운로드하고 공유할 수 있는 패키지 관리자입니다.

- .NET: [NuGet](#)
- Node.js: [npm](#)
- Python: [pip](#)
- Java: [Gradle](#) 또는 [Maven](#)

## 연결 대상 AWS

대부분의 Amazon Web Services (AWS) 리소스는 AWS 계정을 통해 관리됩니다. 를 사용하는 데 AWS 계정이 꼭 필요한 것은 아니지만 연결 없이는 툴킷 기능이 제한됩니다. AWS Toolkit for Visual Studio Code

이전에 AWS 계정을 설정하고 다른 AWS 서비스 (예: AWS Command Line Interface) 를 통해 인증한 경우 에서 자격 증명을 AWS Toolkit for Visual Studio Code 자동으로 감지합니다.

### 필수 조건

계정을 처음 AWS 사용하거나 아직 만들지 않은 경우 다음과 같은 세 가지 기본 단계를 AWS Toolkit for Visual Studio Code 통해 AWS 계정에 연결할 수 있습니다.

1. AWS 계정 가입: 가입 [포털에서 AWS 계정에 AWS 가입할](#) 수 있습니다. 새 AWS 계정 설정에 대한 자세한 내용은 AWS 설치 사용 안내서의 [개요](#) 항목을 참조하십시오.
2. 인증 설정: 에서 AWS 계정을 사용하여 인증하는 세 가지 기본 방법이 있습니다. AWS Toolkit for Visual Studio Code 3가지 방법에 대해 자세히 알아보려면 사용 설명서의 [Authentication and Access](#)(인증 및 액세스)를 참조하세요.
3. AWS 툴킷에서 인증: 이 사용 설명서의 다음 섹션에 있는 절차를 완료하여 툴킷에서 AWS 계정에 연결할 수 있습니다.

### 로그인 패널 열기

다음 절차 중 하나를 완료하여 AWS 툴킷 로그인 패널을 엽니다.

탐색기에서 AWS 툴킷 로그인 패널을 열려면: AWS

1. 에서 AWS Toolkit for Visual Studio Code EXPLORER를 확장합니다.
2. 추가 작업 펼치기... 메뉴를 선택하여... 아이콘.
3. 더 많은 액션에서... 메뉴에서 연결 대상을 AWS 선택하여 AWS 툴킷 로그인 패널을 엽니다.

VS Code 명령 팔레트를 사용하여 AWS 툴킷 로그인 패널을 열려면:

1. **Shift+Command+P(Ctrl+Shift+PWindows)** 를 눌러 명령 팔레트를 엽니다.
2. 검색 **AWS: Add a New Connection** 필드에 입력합니다.
3. **AWS: Add a New Connection** 선택하여 AWS 툴킷 로그인 패널을 엽니다.

## AWS 툴킷에서 연결

### SSO를 통한 인증 및 연결

를 AWS 사용하여 AWS IAM Identity Center 인증하고 연결하려면 다음 절차를 완료하십시오.

#### Note

AWS 빌더 ID 또는 IAM Identity Center를 사용한 인증은 기본 웹 브라우저에서 AWS 인증 포털을 시작합니다. 자격 증명이 만료될 때마다 이 프로세스를 반복하여 계정과 AWS 계정 간의 연결을 갱신해야 합니다. AWS Toolkit for Visual Studio Code

IAM ID 센터를 인증하고 연결합니다. AWS

1. AWS 툴킷 로그인 패널에서 Workforce 탭을 선택한 다음 계속 버튼을 선택하여 계속 진행하십시오.
2. IAM Identity Center로 로그인 패널에서 조직의 시작 URL을 입력합니다. 이 URL은 회사의 관리자 또는 헬프 데스크에서 제공합니다.
3. 드롭다운 메뉴에서 AWS 지역을 선택합니다. ID 디렉터리를 호스팅하는 AWS 지역입니다.
4. 계속 버튼을 선택하고 기본 웹 브라우저에서 AWS 권한 부여 요청 웹 사이트를 열 것인지 확인합니다.
5. 기본 웹 브라우저의 메시지에 따라 인증 프로세스가 완료되면 알림을 받게 되며, 브라우저를 닫고 VS Code로 돌아가도 안전합니다.

### IAM 보안 인증으로 인증 및 연결

IAM 자격 증명을 AWS 사용하여 인증하고 연결하려면 다음 절차를 완료하십시오.

## IAM 보안 인증으로 인증 및 연결

1. AWS 툴킷 로그인 패널에서 IAM 자격 증명을 선택한 다음 계속 버튼을 선택하여 계속하십시오.
2. 제공된 필드에 AWS 계정의 **Profile Name** **Access Key**, 및 **Secret Key** 를 입력한 다음 계속 버튼을 선택하여 구성 파일에 프로필을 추가하고 툴킷을 계정에 연결합니다. AWS
3. 인증이 완료되고 연결이 설정되면 Toolkit AWS 탐색기가 업데이트되어 AWS 서비스 및 리소스를 표시합니다.

## 아마존용 인증 CodeCatalyst

CodeCatalyst 툴킷에서 작업을 시작하려면 AWS 빌더 ID 또는 IAM ID 센터 자격 증명으로 인증하고 연결하세요.

다음 절차에서는 Toolkit을 인증하고 AWS 계정과 연결하는 방법을 설명합니다.

### 빌더 ID로 인증 및 연결 AWS

1. AWS 툴킷 로그인 패널에서 Workforce 탭을 선택한 다음 계속 버튼을 선택하여 계속 진행하십시오.
2. SSO로 로그인 패널 상단에서 로그인으로 건너뛰기 링크를 선택합니다.
3. 기본 웹 브라우저의 메시지에 따라 인증 프로세스가 완료되면 알림을 받게 되며, 브라우저를 닫고 VS Code로 돌아가도 안전합니다.

### IAM ID Center로 인증 및 연결

1. AWS 툴킷 로그인 패널에서 Workforce 탭을 선택한 다음 계속 버튼을 선택하여 계속 진행하십시오.
2. IAM Identity Center로 로그인 패널에서 조직의 시작 URL을 입력합니다. 이 URL은 회사의 관리자 또는 헬프 데스크에서 제공합니다.
3. 드롭다운 메뉴에서 AWS 지역을 선택합니다. ID 디렉터리를 호스팅하는 AWS 지역입니다.
4. 계속 버튼을 선택하고 기본 웹 브라우저에서 AWS 권한 부여 요청 웹 사이트를 열 것인지 확인합니다.
5. 기본 웹 브라우저의 메시지에 따라 인증 프로세스가 완료되면 알림을 받게 되며, 브라우저를 닫고 VS Code로 돌아가도 안전합니다.

## AWS 지역 변경

AWS 지역은 AWS 리소스가 관리되는 위치를 지정합니다. 에서 AWS 계정에 연결하면 기본 AWS 지역이 감지되어 AWS 탐색기에 자동으로 표시됩니다. AWS Toolkit for Visual Studio Code

다음 섹션에서는 AWS Explorer에서 리전을 추가하고 숨기는 방법을 설명합니다.

### AWS 탐색기에 지역 추가

AWS 탐색기에 지역을 추가하려면 다음 절차를 완료하세요.

1. VS Code 기본 메뉴에서 보기를 확장한 다음 명령 팔레트를 선택하여 명령 팔레트를 여세요. 또는 아래의 바로 가기 키를 사용하세요.
  - Windows 및 Linux - **Ctrl+Shift+P** 키를 누릅니다.
  - macOS - **Shift+Command+P** 키를 누릅니다.
2. 명령 팔레트에서 지역 표시 또는 숨기기를 검색하여 사용 가능한 지역 목록을 표시합니다. **AWS: Show or Hide Regions**
3. 목록에서 AWS 탐색기에 추가할 AWS 지역을 선택합니다.
4. 확인 버튼을 선택하여 선택 사항을 확인하고 AWS 탐색기를 업데이트합니다.

### AWS 탐색기에서 지역 숨기기

AWS 탐색기 보기에서 지역을 숨기려면 다음 절차를 완료하세요.

1. AWS 탐색기에서 숨기려는 AWS 지역을 찾습니다.
2. 숨기려는 리전에 대한 컨텍스트(마우스 우클릭) 메뉴를 엽니다.
3. 지역 표시 또는 숨기기를 선택하여 VS Code에서 지역 표시 또는 숨기기 옵션을 엽니다. **AWS**
4. AWS 탐색기 보기에서 숨기려는 지역을 선택 취소합니다.

## 툴체인 구성

AWS Toolkit for Visual Studio Code은 모든 AWS 서비스에 다양한 언어를 지원합니다. 다음 섹션은 언어에 맞게 툴체인을 구성하는 방법을 설명합니다.

## .NET Core 툴체인 구성.

1. AWS VS Code용 도구 키트가 [설치](#)되어 있는지 확인하세요.
2. [C# 확장 프로그램](#)을 설치합니다. 확장 프로그램을 설치하면 VS Code로 .NET Core 애플리케이션을 디버깅할 수 있습니다.
3. AWS Serverless Application Model(AWS SAM) 애플리케이션을 열거나 [새로 생성하세요](#).
4. `template.yaml`가 포함된 폴더를 엽니다.

## Node.js 툴체인 구성

1. AWS Toolkit for VS Code가 [설치](#)되어 있는지 확인하세요.
2. AWS SAM 애플리케이션을 열거나 [새로 생성하세요](#).
3. `template.yaml`이 포함된 폴더를 엽니다.

### Note

소스 코드("target": "code"가 포함된 시작 구성)에서 TypeScript Lambda 함수를 디버깅하려면 TypeScript 컴파일러를 전역적으로 또는 `package.json` 프로젝트에 설치하세요.

## Python 툴체인 구성

1. AWS VS Code용 도구 키트가 [설치](#)되어 있는지 확인하세요.
2. [Python extension for Visual Studio Code](#)을 설치합니다. 확장 프로그램을 설치하면 VS Code로 Python 애플리케이션을 디버깅할 수 있습니다.
3. AWS SAM 애플리케이션을 열거나 [새로 생성하세요](#).
4. `template.yaml`이 포함된 폴더를 엽니다.
5. 애플리케이션 루트에서 터미널을 열고 `python -m venv ./venv`를 실행하여 `virtualenv`를 구성하세요.

### Note

`virtualenv` 시스템당 한 번만 구성하면 됩니다.



6. 다음 중 하나를 실행하여 `virtualenv`를 활성화합니다.

- Bash shell: `./venv/Scripts/activate`
- PowerShell: `./venv/Scripts/Activate.ps1`

## Java 툴체인 구성

1. AWS Toolkit for VS Code가 [설치](#)되어 있는지 확인하세요.
2. [Java 확장 프로그램과 Java 11](#)을 설치합니다. 확장 프로그램을 설치하면 VS Code로 Java 함수를 인식할 수 있습니다.
3. [Java 디버거 확장 프로그램](#)을 설치합니다. 확장 프로그램을 설치하면 VS Code로 Java 애플리케이션을 디버깅할 수 있습니다.
4. AWS SAM 애플리케이션을 열거나 [새로 생성하세요](#).
5. `template.yaml`이 포함된 폴더를 엽니다.

## Go 툴체인 구성하기

1. AWS VS Code용 도구 키트가 [설치](#)되어 있는지 확인하세요.
2. Go Lambda 함수를 디버깅하려면 Go 1.14 이상이어야 합니다.
3. [Go extension](#)을 설치합니다.

### Note

Go1.15+ 런타임을 디버깅하려면 버전 0.25.0 이상이어야 합니다.

4. [command palette](#)로 Go 도구를 설치하세요.
  - a. `command palette`에서 `Go: Install/Update Tools`을 선택합니다.
  - b. 체크 박스에서 `dlv` 및 `gopls`를 선택합니다.
5. AWS SAM 애플리케이션을 열거나 [새로 생성하세요](#).
6. `template.yaml`이 포함된 폴더를 엽니다.

## 툴체인 사용

툴체인을 설정하면 AWS SAM 애플리케이션을 [실행하거나 디버깅](#)할 수 있습니다.

# AWS Toolkit for Visual Studio Code에 대한 인증 및 액세스

사용을 시작하기 AWS 위해 인증을 받을 필요는 없습니다. AWS Toolkit for Visual Studio Code 하지만 대부분의 AWS 리소스는 AWS 계정을 통해 관리됩니다. 모든 AWS Toolkit for Visual Studio Code 서비스와 기능에 액세스하려면 AWS Builder ID 또는 IAM 자격 증명으로 AWS IAM Identity Center 인증해야 합니다.

다음 항목에는 각 자격 증명 유형에 대한 추가 세부 정보가 포함되어 있습니다.

기존 자격 증명을 AWS Toolkit for Visual Studio Code 사용하여 AWS 에 연결하는 방법에 대한 자세한 내용은 이 사용 설명서의 [연결](#) 대상 AWS 항목을 참조하십시오.

주제

- [AWS IAM 아이덴티티 센터](#)
- [AWS IAM 자격 증명](#)
- [AWS 개발자용 빌더 ID](#)
- [외부 보안 인증 프로세스 사용.](#)

## AWS IAM 아이덴티티 센터

AWS IAM Identity Center AWS 계정 인증을 관리하기 위한 권장 모범 사례입니다.

소프트웨어 개발 키트(SDK)용 IAM Identity Center를 설정하는 방법에 대한 자세한 지침은 AWS SDKs 와 도구 참조 안내서의 [IAM Identity Center 인증](#) 섹션을 참조하세요.

기존 IAM Identity Center 자격 증명으로 AWS 툴킷을 인증하고 연결하는 방법에 대한 자세한 내용은 이 사용 설명서의 [Connect to AWS](#) 항목을 참조하십시오.

## AWS IAM 자격 증명

AWS 로컬에 저장된 액세스 키를 통해 AWS 계정을 통한 IAM 자격 증명 인증.

기존 AWS IAM 자격 증명을 사용하여 AWS 툴킷을 인증하고 연결하는 방법에 대한 자세한 내용은 이 사용 설명서의 [Connect to AWS](#) 항목을 참조하십시오.

다음 섹션에서는 에서 계정을 인증하도록 IAM 자격 증명을 설정하는 방법을 설명합니다. AWS AWS Toolkit for Visual Studio Code

### ⚠ Important

AWS 계정 인증을 위해 IAM 자격 증명을 설정하기 전에 다음 사항을 참고하십시오.

- 다른 AWS 서비스 (예: AWS CLI) 를 통해 이미 IAM 자격 증명을 설정한 경우는 해당 자격 증명을 AWS Toolkit for Visual Studio Code 자동으로 탐지하여 VS Code에서 사용할 수 있도록 합니다.
- AWS IAM ID 센터 인증을 사용할 것을 권장합니다. AWS IAM 모범 사례에 대한 추가 정보는 AWS Identity 및 Access Management 사용 설명서의 [IAM 섹션의 보안 모범 사례](#)를 참조하십시오.
- 보안 사고를 방지하려면 특수 목적 소프트웨어 개발 또는 실제 데이터로 작업할 때 IAM 사용자 인증하지 마세요. 대신 AWS IAM Identity Center 사용 설명서의 [What is IAM Identity Center?](#)와 같은 ID 제공자의 페더레이션을 사용합니다.

## IAM 사용자 생성

AWS 계정을 사용하여 AWS Toolkit for Visual Studio Code 인증하도록 설정하려면 먼저 AWS SDK 및 도구 참조 안내서의 [장기 자격 증명을 사용하여 인증 항목에서 1단계: IAM 사용자 생성 및 2단계: 액세스 키 가져오기를 완료해야 합니다.](#)

### 📌 Note

3단계: AWS SDKs and Tools Reference Guide에서 공유 보안 인증 파일 업데이트하는 것은 선택 사항입니다.

3단계를 완료하면 아래 단계에 따라 자격 증명에 AWS Toolkit for Visual Studio Code 자동으로 검색됩니다. [the section called “에서 공유 자격 증명 파일 생성 AWS Toolkit for Visual Studio Code”](#)

3단계를 완료하지 않은 경우 아래 [the section called “에서 공유 자격 증명 파일 생성 AWS Toolkit for Visual Studio Code”](#) 위치에 설명된 credentials file 대로 생성 프로세스를 단계별로 AWS Toolkit for Visual Studio Code 안내합니다.

## 에서 공유 자격 증명 파일 생성 AWS Toolkit for Visual Studio Code

공유 구성 파일 및 공유 자격 증명 파일에는 계정의 구성 및 자격 증명 정보가 저장됩니다. AWS 공유 구성 및 보안 인증에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 [Where are configuration settings stored?](#)(구성 설정은 어디에 저장됩니까?) 섹션을 참조하세요.

를 통한 공유 자격 증명 파일 생성 AWS Toolkit for Visual Studio Code

1. **Shift+Command+P(Ctrl+Shift+PWindows)** 를 눌러 명령 팔레트를 엽니다.
2. 검색 **AWS: Add a New Connection** 필드에 입력합니다.
3. **AWS: Add a New Connection** 선택하여 AWS 툴킷 로그인 패널을 엽니다.
4. AWS 툴킷 로그인 패널에서 IAM 자격 증명을 선택한 다음 계속 버튼을 선택하여 계속하십시오.
5. 제공된 필드에 AWS 계정의 **Profile Name** **Access Key**, 및 **Secret Key** 를 입력한 다음 계속 버튼을 선택하여 구성 파일에 프로필을 추가하고 툴킷을 계정에 연결합니다. AWS
6. 인증이 완료되고 연결이 설정되면 Toolkit AWS 탐색기가 업데이트되어 AWS 서비스 및 리소스를 표시합니다.

### Note

이것은 `[Profile_Name]`의 구문 오류로 인해 인증이 실패한 예시입니다.

```
[Profile_Name]
xaws_access_key_id= AKIAI44QH8DHBEXAMPLE
xaws_secret_access_key= wJalrXUtnFEMI/K7MDENG/bPxrRfiCYEXAMPLEKEY
```

다음은 인증 오류로 생성된 로그 메시지의 예입니다.

```
2022-11-02 22:01:54 [ERROR]: Profile [Profile_Name] is not a valid Credential
Profile: not supported by the Toolkit
2022-11-02 22:01:54 [WARN]: Shared Credentials Profile [Profile_Name] is not
valid. It will not be used by the toolkit.
```

## 자격 증명 프로필 추가

구성 파일에 여러 자격 증명을 추가할 수 있습니다. 추가하려면 Command Palette를 열고 AWS Toolkit Create Credentials Profile을 선택하세요. 그러면 자격 증명 파일이 열립니다. 다음 예와 같이 이 페이지에서 첫 번째 프로필 아래에 새 프로필을 추가할 수 있습니다.

```
# Amazon Web Services Credentials File used by AWS CLI, SDKs, and tools
# This file was created by the AWS Toolkit for Visual Studio Code extension.
#
# Your AWS credentials are represented by access keys associated with IAM users.
# For information about how to create and manage AWS access keys for a user, see:
# https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_access-keys.html
#
# This credential file can store multiple access keys by placing each one in a
# named "profile". For information about how to change the access keys in a
# profile or to add a new profile with a different access key, see:
# https://docs.aws.amazon.com/cli/latest/userguide/cli-config-files.html
#
[Profile1_Name]
# The access key and secret key pair identify your account and grant access to AWS.
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
# Treat your secret key like a password. Never share your secret key with anyone. Do
# not post it in online forums, or store it in a source control system. If your secret
# key is ever disclosed, immediately use IAM to delete the access key and secret key
# and create a new key pair. Then, update this file with the replacement key details.
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
[Profile2_Name]
aws_access_key_id = AKIAI44QH8DHBEXAMPLE
aws_secret_access_key = je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
```

## AWS 개발자용 빌더 ID

AWS 빌더 ID는 선택 사항이거나 특정 AWS 서비스에 필요한 추가 AWS 계정입니다. AWS 빌더 ID 인증 방법에 대한 자세한 내용은 AWS 로그인 사용 안내서의 [AWS 빌더 ID로](#) 로그인 항목을 참조하십시오.

AWS 툴킷을 인증하고 기존 AWS 빌더 ID로 연결하는 방법에 대한 자세한 내용은 이 사용 설명서의 [Connect to AWS](#) 항목을 참조하십시오.

## 외부 보안 인증 프로세스 사용.

shared config file을 수정하여 AWS가 지원하지 않는 보안 인증 프로세스에 맞게 AWS Toolkit for Visual Studio Code를 구성할 수 있습니다.

shared config file의 보안 인증 프로세스를 수정하는 것은 AWS Toolkit for Visual Studio Code와 AWS Command Line Interface 모두 동일합니다. 외부 보안 인증을 설정하는 방법에 대한 자세한 내용은 AWS Command Line Interface사용 설명서의 [Sourcing credentials with an external process](#)(외부 프로세스로 보안 인증 소싱) 항목을 참조하십시오.

# AWS 서비스 및 도구 사용

를 AWS Toolkit for Visual Studio Code 사용하면 VS Code에서 직접 AWS 서비스, 도구 및 리소스를 사용할 수 있습니다. 다음은 각 Toolkit for VS Code 서비스와 해당 기능을 다루는 가이드 항목 목록입니다. 서비스 또는 도구의 기능, 설정 방법, 기본 기능 사용에 대한 자세한 내용을 보려면 서비스 또는 도구를 선택하십시오.

## 주제

- [실험 기능 작업](#)
- [AWS 탐색기에서 AWS 서비스 사용하기](#)
- [VS Code용 Amazon CodeCatalyst](#)
- [Amazon API Gateway로 실행하기](#)
- [AWS App Runner 와 함께 사용 AWS Toolkit for Visual Studio Code](#)
- [AWS Application Composer](#)
- [AWS CDK VS 코드](#)
- [AWS CloudFormation 스택 작업](#)
- [AWS Toolkit for Visual Studio Code를 사용하여 CloudWatch Logs 작업](#)
- [Amazon Elastic Container Registry 서비스 사용](#)
- [Amazon Elastic Container Service 작업](#)
- [Amazon EventBridge 작업](#)
- [AWS IAM 액세스 분석기](#)
- [AWS Toolkit for Visual Studio Code에서 AWS IoT로 작업](#)
- [AWS Lambda 함수 작업](#)
- [Toolkit for VS Code의 Amazon Redshift](#)
- [Amazon S3 작업](#)
- [서버리스 애플리케이션 작업](#)
- [Systems Manager 자동화 설명서로 작업](#)
- [작업 대상 AWS Step Functions](#)
- [위협 컴포저와 함께 작업하기](#)
- [리소스 작업](#)

## 실험 기능 작업

실험 기능을 통해 공식 출시 전 AWS Toolkit for Visual Studio Code 기능에 미리 액세스할 수 있습니다.

### Warning

실험 기능을 계속 테스트하고 업데이트하므로 사용성 문제가 있을 수 있습니다. 실험 기능은 AWS Toolkit for Visual Studio Code에서 예고 없이 삭제될 수도 있습니다.

VS Code IDE 설정 창의 AWS 도구 키트 섹션에서 특정 AWS 서비스에 대한 실험 기능을 활성화할 수 있습니다.

1. VS Code AWS 설정을 변경하려면 파일, 환경설정, 설정을 선택합니다.
2. 설정 창에서 확장을 클릭한 다음 AWS도구 키트를 선택합니다.
3. AWS: 확장에서 출시 전 액세스하고 싶은 실험 기능의 확인란을 선택합니다. 실험 기능을 끄려면 관련 확인란을 선택 취소합니다.

## AWS 탐색기에서 AWS 서비스 사용하기

AWS Explorer에서는 를 사용할 때 사용할 수 있는 일부 AWS 서비스를 볼 수 AWS Toolkit for Visual Studio Code 있습니다.

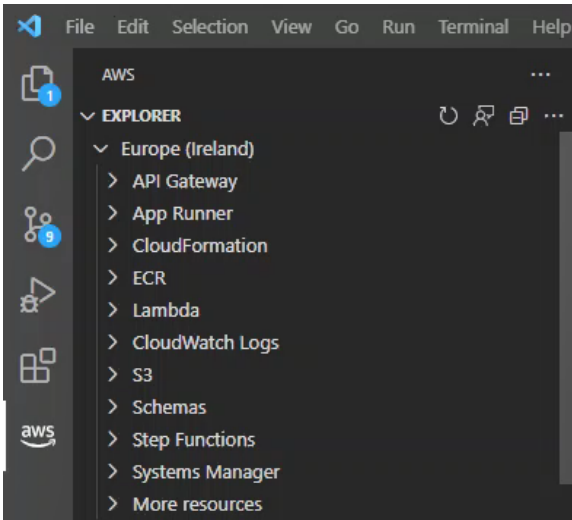
본 섹션은 VS Code AWS 탐색기에 액세스하고 사용하는 방법에 대한 정보를 제공합니다. 여기에서는 사용자 시스템에 VS Code 도구 키트가 [설치 및 구성](#)되어 있다고 가정합니다.

### 주의 사항

- 도구 키트가 올바르게 설치 및 구성되었다면 AWS 탐색기에 항목이 표시됩니다. 활동 표시줄에서 AWS 아이콘을 선택하면 AWS 탐색기를 볼 수 있습니다.

예:





- 특정 기능에는 특정 AWS 권한이 필요합니다. 예를 들어, AWS 계정에서 AWS Lambda 함수를 보려면 구성된 자격 증명에 최소한 읽기 전용 Lambda 권한이 [인증 및 액세스](#) 포함되어야 합니다. 각 기능에 필요한 권한에 대한 자세한 내용은 다음 주제를 참조하세요.
- AWS 탐색기에 바로 표시되지 않는 AWS 서비스와 상호 작용하려는 경우 추가 리소스로 이동하여 인터페이스에 추가할 수 있는 수백 개의 리소스에서 선택할 수 있습니다.

예를 들어, 사용 가능한 리소스 유형 선택에서 AWS ToolkitCodeArtifact:::Repository를 선택할 수 있습니다. 이 리소스 유형을 추가 리소스에 추가한 후에는 항목을 확장하여 고유한 속성과 속성을 가진 다양한 CodeArtifact 리포지토리를 생성하는 리소스 목록을 볼 수 있습니다. 또한 JSON 형식의 템플릿으로 리소스의 속성과 속성을 설명할 수 있으며, 이를 저장하여 클라우드에서 새 리소스를 생성할 수 있습니다. AWS

## VS Code용 Amazon CodeCatalyst

### Amazon CodeCatalyst란?

Amazon CodeCatalyst는 소프트웨어 개발 팀을 위한 클라우드 기반 협업 공간입니다. AWS Toolkit for Visual Studio Code를 통해 VS Code에서 직접 CodeCatalyst 리소스를 보고 관리할 수 있습니다. AWS 도구 키트를 사용하여 VS Code를 사용하는 개발 환경 가상 컴퓨팅 환경을 실행하여 클라우드에서 직접 작업할 수도 있습니다. CodeCatalyst 서비스에 대한 자세한 내용은 [Amazon CodeCatalyst 사용 설명서](#)를 참조하세요.

다음 항목에서 CodeCatalyst와 VS Code를 연결하는 방법과 VS Code용 도구 키트에서 CodeCatalyst를 사용하는 방법을 설명합니다.

### 주제

- [CodeCatalyst와 VS Code용 툴킷 시작하기](#)
- [VS Code에서 Amazon CodeCatalyst 리소스 작업](#)
- [개발 환경 및 VS Code에서 작업하기](#)
- [Amazon CodeCatalyst 및 VS Code 문제 해결](#)

## CodeCatalyst와 VS Code용 툴킷 시작하기

VS Code에서 CodeCatalyst를 사용하려면 다음 절차를 따르세요.

주제

- [VS 코드 설치](#)
- [VS Code용 도구 키트 설치](#)
- [CodeCatalyst 계정 생성](#)
- [AWS 도구 키트와 CodeCatalyst 연결](#)

### VS 코드 설치

시작하기 전에 최신 VS Code 버전이 있어야 합니다. 최신 VS Code 버전을 다운로드하려면 VS Code 웹 사이트에서 [Download VS Code](#)(VS Code 다운로드)를 참조하세요.

### VS Code용 도구 키트 설치

VS Code용 도구 키트 최신 확장 프로그램 버전을 설치하여 VS Code용 도구 키트를 CodeCatalyst 계정에 연결합니다. VS Code Extensions: Marketplace에서 최신 버전을 찾아 AWS 도구 키트를 설치하세요.

VS Code Extensions: Marketplace를 열어 VS Code용 도구 키트를 설치하려면 다음 단계를 완료하세요.

1. 기본 화면 왼쪽 가장자리에 있는 VS Code Activity Bar(활동 표시줄)에서 Extensions(확장 프로그램) 아이콘을 선택하여 EXTENSIONS: MARKETPLACE(확장 프로그램: 마켓플레이스) 보기를 엽니다.

#### Note

VS Code Command Palette(명령 팔레트)를 사용하여 확장 프로그램: 마켓플레이스 보기를 열 수도 있습니다. 이렇게 하려면 Command+Shift+P (macOS) 또는 Ctrl+Shift+P

(Windows)를 눌러 명령 팔레트를 엽니다. 그런 다음 제공된 검색 필드에 **Extensions: Focus on Marketplace View**를 입력하고 **Extensions: Focus on Marketplace View**(확장 프로그램: 마켓플레이스 보기)를 선택하면 마켓플레이스 보기가 열립니다.

- EXTENSIONS: MARKET PLACE(확장 프로그램: 마켓플레이스) 검색 창에 AWS Toolkit를 입력합니다.
- 입력한대로 검색 결과가 필터링되며, AWS도구 키트 확장 프로그램이 있는 경우 설치를 선택하여 최신 버전의 AWS 도구 키트를 설치합니다.

## CodeCatalyst 계정 생성

VS Code용 도구 키트에 CodeCatalyst를 연결하려면 최신 버전의 AWS 설치뿐만 아니라 활성 AWS Builder ID 계정이 필요합니다. 활성 AWS Builder ID 또는 CodeCatalyst 계정이 없는 경우 CodeCatalyst 사용 설명서의 [Setting up with CodeCatalyst](#)(CodeCatalyst로 설정하기) 섹션을 참조하세요.

### Note

AWS Builder ID는 IAM 자격 증명과 다릅니다. IAM 자격 증명은 AWS 도구 키트에서 액세스하는 대부분의 AWS 서비스에 필요합니다. 반대로 AWS Builder ID는 새 CodeCatalyst 계정을 생성하고 이를 AWS 도구 키트와 연결하는 데 필요합니다.

## AWS 도구 키트와 CodeCatalyst 연결

AWS 도구 키트를 CodeCatalyst 계정과 연결하려면 다음 단계를 완료하세요.

- VS Code용 도구 키트에서 DEVELOPER TOOLS(개발자 도구) 탐색기를 확장합니다.
- DEVELOPER TOOLS(개발자 도구) 탐색기에서 CodeCatalyst를 확장하고 Start(시작)을 선택합니다.
- VS Code에 메시지가 표시되면 선호하는 웹 브라우저에서 AWS 빌더 ID 포털을 열고 진행하세요.

### Note

AWS Builder ID가 없는 경우 제공된 링크를 통해 만드세요. 그런 다음 다음 단계를 완료하여 AWS 빌더 ID 자격 증명을 인증하고 VS Code용 도구 키트에 연결합니다.

4. AWS Builder ID 포털에서 다음 단계에 따라 AWS Builder ID를 인증하세요. 메시지가 표시되면 VS Code용 도구 키트에서 데이터에 액세스할 수 있도록 Allow(허용)을 선택합니다.
5. AWS Builder ID 자격 증명이 VS Code용 도구 키트에 연결되면 브라우저가 업데이트됩니다. 이는 브라우저를 닫고 VS Code용 도구 키트로 돌아가도 안전하다는 것을 나타냅니다.

#### Note

VS Code용 도구 키트가 IAM 자격 증명의 인증받았다면 VS Code로 다음 프롬프트를 열 수 있습니다.

사용중인 일부 도구는 AWS Builder ID와 호환되지 않습니다. AWS 보안 인증을 사용하는 동안 백그라운드에서 AWS Builder ID를 계속 사용하겠습니까?

- AWS Builder ID를 사용하여 CodeCatalyst 서비스를 인증하고 IAM 보안 인증을 사용하여 다른 VS Code용 도구 키트 서비스를 인증하려면 예, 다른 서비스에서는 IAM 보안 인증을 사용하고 CodeCatalyst에 연결된 AWS Builder ID를 계속 사용하겠다는 것을 선택하세요.
- 계AWS Builder ID로만 모든 VS Code용 도구 키트 서비스를 인증하려면 아니오, AWS Builder ID로만 인증하기로 전환하겠다는 것을 선택하세요.

6. AWS Builder ID가 VS Code용 도구 키트에 연결되면 개발자 도구 탐색기에 CodeCatalyst 기능이 업데이트되고 AWS Builder ID Connected(빌더 ID 연결됨) 상태를 표시합니다.

## VS Code에서 Amazon CodeCatalyst 리소스 작업

다음 섹션에서는 VS Code용 도구 키트에서 사용할 수 있는 Amazon CodeCatalyst 리소스 관리 기능에 대한 개요를 설명합니다.

개발 환경에 대한 자세한 내용과 CodeCatalyst에서 개발 환경에 액세스하는 방법은 Amazon CodeCatalyst 사용 설명서의 [Dev Environments](#)(개발 환경) 섹션을 참조하세요.

다음 섹션에서는 VS Code에서 개발 환경을 생성하고 열고 작업하는 방법을 설명합니다.

### 주제

- [리포지토리 복제](#)
- [개발 환경 열기](#)
- [CodeCatalyst 개발 환경 생성](#)
- [타사 리포지토리에서 개발 환경 생성](#)

## • [VS Code의 CodeCatalyst 명령](#)

### 리포지토리 복제

CodeCatalyst는 클라우드 기반 서비스이며, CodeCatalyst 프로젝트를 수행하려면 클라우드에 연결되어 있어야 합니다. 컴퓨터에서 프로젝트를 작업하는 것을 선호하는 경우, CodeCatalyst 리포지토리를 로컬 시스템에 복제하고 나중에 클라우드에 연결하면 CodeCatalyst 프로젝트와 동기화할 수 있습니다.

AWS 도구 키트를 사용하여 CodeCatalyst 계정의 리포지토리를 VS Code로 복제하려면 다음 단계를 완료하세요.

1. VS Code용 도구 키트에서 DEVELOPER TOOLS(개발자 도구) 탐색기를 확장합니다.
2. CodeCatalyst를 확장하고 Clone Repository(리포지토리 복제)를 선택합니다.
3. Select a CodeCatalyst Repository(CodeCatalyst 리포지토리 선택) 대화 상자에서 복제하려는 리포지토리를 검색한 다음 해당 리포지토리를 선택하여 Choose a folder to clone(복제할 폴더 선택) 대화 상자를 엽니다.
4. Select Repository Location(리포지토리 위치 선택)을 클릭하여 메시지를 닫고 리포지토리 복제를 시작합니다.

#### Note

타사 서비스에서 리포지토리를 복제하는 경우 해당 서비스의 자격 증명으로 인증하라는 메시지가 표시될 수 있습니다.

리포지토리가 복제되는 동안 VS Code 상태 창에 Cloning Repository(리포지토리 복제) 진행 상황이 표시됩니다. 리포지토리가 복제되면 Would you like to open the cloned repository?(복제된 리포지토리를 여시겠습니까?)라는 메시지가 나타납니다.

5. 대화 상자 창에서 다음 중 하나를 선택하여 복제 프로세스를 완료합니다.
  - 현재 VS Code 창에서 리포지토리를 열려면 Open(열기)을 선택합니다.
  - 새 VS Code 창에서 리포지토리를 열려면 Open in new window(새 창에서 열기)를 선택합니다.
  - 리포지토리를 열지 않고 복제 프로세스를 완료하려면 대화 상자를 닫으세요.

### 개발 환경 열기

VS Code에서 기존 개발 환경을 열려면

1. VS Code용 도구 키트에서 DEVELOPER TOOLS(개발자 도구) 탐색기를 확장합니다.
2. CodeCatalyst를 확장하고 Open Dev Environment(개발 환경 열기)를 선택하여 VS Code에서 Select a CodeCatalyst Dev Environment(CodeCatalyst 개발 환경 선택) 대화 상자를 엽니다.
3. Select a CodeCatalyst Dev Environment(CodeCatalyst 개발 환경 선택) 대화 상자에서 열리는 Dev Environment(개발 환경)를 선택합니다.

#### Note

개발 환경을 선택하면 개발 환경이 열리고 VS Code를 CodeCatalyst에 연결하는 프로세스가 시작됩니다. 프로세스가 진행되는 동안 VS Code 상태 창에 CodeCatalyst 업데이트 상황이 표시됩니다. 상태 창이 업데이트되었다면 업데이트가 완료된 것입니다.

- 개발 환경이 열리지 않으면 업데이트가 실패한 이유에 대한 정보와 프로세스 로그를 열 수 있는 링크가 포함된 상태로 변경됩니다.
- 업데이트가 완료되면 VS Code 새 창에서 개발 환경이 열립니다.

## CodeCatalyst 개발 환경 생성

### 새 개발 환경 생성

1. VS Code용 도구 키트에서 DEVELOPER TOOLS(개발자 도구) 탐색기를 확장합니다.
2. CodeCatalyst를 확장하고 Create Dev Environment(개발 환경 생성) 옵션을 선택한 다음 VS Code에서 Create a CodeCatalyst Dev Environment(CodeCatalyst 개발 환경 생성) 메뉴를 여세요.
3. Source Code(소스 코드) 섹션에서 다음 옵션 중 하나를 선택합니다.
  - 기존 CodeCatalyst 리포지토리 사용: 기존 CodeCatalyst 리포지토리에서 개발 환경을 만듭니다. CodeCatalyst Project와 Branch를 선택하세요.
  - 빈 개발 환경 만들기: 빈 개발 환경을 만듭니다.
4. (선택 사항) Alias 섹션에서 개발 환경의 대체 이름을 입력합니다.

#### Note

필수는 아니지만 이름을 지정하면 개발 환경의 구성 및 검색이 간소화되므로 이름을 지정하는 것이 좋습니다.

5. (선택 사항) Dev Environments Configuration(개발 환경 구성) 섹션에서 특정 요구 사항에 맞게 다음 설정을 변경하세요.
- Compute: Edit Compute를 선택하여 시스템에 할당된 처리 능력과 RAM의 용량을 변경합니다.
  - Timeout: Edit Timeout을 선택하여 개발 환경이 중지되기 전에 허용되는 시스템 유휴 시간을 변경합니다.

**Note**

개발 환경은 작업을 영구 저장합니다. 따라서 작업 손실 없이 개발 환경을 중지할 수 있습니다. 개발 환경을 중지하면 개발 환경을 가동하고 운영하는 데 필요한 비용이 줄어듭니다.

- Storage: Edit Storage Size 선택하여 시스템에 할당된 스토리지 공간의 크기를 변경합니다.

**Note**

Storage는 개발 환경이 생성된 후에 변경이 불가능합니다.

6. 새 클라우드 개발 환경을 생성하려면 Create Dev Environment(개발 환경 생성)을 선택합니다.

**Note**

VS Code 상태 창에 개발 환경 생성 진행 상황이 표시됩니다. 개발 환경이 생성되면 VS Code에 개발 환경 새 창이 열리고 이 폴더에 있는 Do you trust the authors of the files in this folder?(이 폴더의 파일 작성자를 신뢰합니까?) 메시지가 나타납니다. 개발 환경에서 계속 작업하려면 이용 약관에 동의하세요.

## 타사 리포지토리에서 개발 환경 생성

리포지토리에 소스로 연결하여 타사 리포지토리에서 개발 환경을 생성할 수 있습니다.

타사 리포지토리에 소스로 연결은 CodeCatalyst의 프로젝트 수준에서 처리됩니다. 개발 환경에 타사 리포지토리를 연결하는 방법에 대한 지침과 자세한 정보는 Amazon CodeCatalyst 사용 설명서의 [Linking a source repository](#)(소스 리포지토리 연결)를 참조하세요.

## VS Code의 CodeCatalyst 명령

AWS 도구 키트에 직접 표시되지 않는 CodeCatalyst 관련 기능 및 역할에 할당되는 추가 VS Code 명령이 있습니다.

명령 팔레트에서 CodeCatalyst에 할당된 명령 목록 보기

1. VS Code용 도구 키트에서 DEVELOPER TOOLS(개발자 도구) 탐색기를 확장합니다.
2. Show CodeCatalyst Commands(CodeCatalyst 명령 표시)를 선택하면 CodeCatalyst 검색어가 포함된 명령 팔레트가 열립니다.
3. 목록에서 CodeCatalyst 명령을 선택하여 활성화합니다.

## 개발 환경 및 VS Code에서 작업하기

VS Code용 도구 키트에서 개발 환경이라는 VS Code 가상 컴퓨팅 환경을 시작할 수 있습니다. 개발 환경은 스페이스의 여러 팀원들이 복사하고 공유할 수 있는 사용자 지정이 가능한 클라우드 개발 환경입니다.

다음 섹션에서는 로컬 및 원격으로 개발 환경을 구성하는 방법과 개발 환경에서 실행할 때 VS Code용 도구 키트에서 액세스할 수 있는 기본 기능에 대해 설명합니다. Dev Environment에 대한 자세한 내용은 CodeCatalyst 설명서의 [Dev Environment](#)(개발 환경)를 참조하세요.

주제

- [devfiles로 개발 환경 구성](#)
- [AWS IAM 자격 증명과 개발 환경 연결](#)
- [개발 환경에서 VS Code용 도구 키트로 작업](#)

### devfiles로 개발 환경 구성

개발 환경 devfile에서 개발 환경에 설치된 프로젝트별 도구 및 애플리케이션 라이브러리를 정의할 수 있습니다. devfile 사양은 YAML 문서에 작성할 수 있는 공개 표준입니다.

VS Code에서 로컬 및 원격으로 devfile을 수정할 수 있습니다. 프로세스는 VS Code 호환 파일 생성 또는 편집하는 것과 비슷합니다. CodeCatalyst 프로젝트의 소스 리포지토리에서 Devfile을 편집할 수도 있습니다. CodeCatalyst에서 devfile을 편집하는 방법에 대한 지침은 CodeCatalyst 가이드의 [Configuring your Dev Environment](#)(개발 환경 구성) 섹션을 참조하세요.



**⚠ Important**

VS Code에서 Devfile을 편집하는 경우 다음 사항에 유의하세요.

- devfile 이름 또는 devfile 구성 요소 이름을 변경하면 루트 디렉터리 콘텐츠가 바뀝니다. 이전의 모든 콘텐츠는 사라지고 복구할 수 없습니다.
- 루트 폴더에 devfile이 없는 개발 환경이나 소스 리포지토리와 연결되지 않은 개발 환경을 만드는 경우, 기본 구성 설정이 포함된 devfile이 생성됩니다.
- Devfile 정의 및 구성하는 방법에 대한 지침은 [devfile.io](https://devfile.io) 웹 사이트의 [Adding Commands](#)(명령 추가) 설명서를 참조하세요.

**VS Code의 로컬 인스턴스에서 Devfile 편집**

1. VS Code 탐색기에서 devfile.yaml 파일을 찾아 VS Code 편집기에서 엽니다.
2. 문서를 변경한 다음 저장합니다.
3. 변경한 다음 변경 내용을 소스 리포지토리에 저장합니다.
4. 다음에 개발 환경을 시작하면 구성이 Devfile에 정의된 사양과 동일하게 변경됩니다.

**AWS IAM 자격 증명과 개발 환경 연결**

개발 환경에서 모든 Toolkit for VS Code 서비스에 액세스하려면 IAM 자격 증명을 개발 환경에 AWS 연결해야 합니다. IAM 자격 증명을 개발 환경에 연결하려면 다음을 완료하세요.

1. 개발 환경의 VS Code용 도구 키트 EXPLORER에서 Select IAM Credentials to View Resources(리소스를 볼 수 있는 IAM 자격 증명 선택)를 클릭하면 Switch Connection(스위치 연결) 대화 상자가 열립니다.
2. 목록에서 사용하려는 IAM 자격 증명을 선택하거나 Add New Connection(새 연결 추가)을 선택하면 새 자격 증명이 추가됩니다.
3. 다음 옵션 중에서 선택하고 안내에 따라 새 연결을 생성하세요.
  - 개인 이메일로 AWS Builder ID 생성 및 로그인: 개인용 AWS Builder ID를 생성합니다. 이 옵션은 CodeCatalyst와 같은 AWS Builder ID 전용 서비스 사용만 지원합니다.
  - AWS IAM ID 센터를 통한 연결: 회사 IAM 액세스 포털에 로그인할 수 있습니다. 이 옵션은 IAM ID 센터 액세스 권한이 있는 회사를 통해서만 사용할 수 있습니다. 이 계정으로 액세스할 수 있는 AWS 리소스와 서비스는 계정을 소유한 회사에서 관리합니다.

- IAM 자격 증명 사용: VS Code용 도구 키트의 AWS 서비스 및 리소스를 사용할 수 있습니다. 이 옵션만 사용하는 경우 CodeCatalyst와 같은 AWS Builder ID 전용 서비스를 사용할 수 없습니다.
4. IAM 자격 증명이 개발 환경의 VS Code용 도구 키트와 연결되면 AWS EXPLORER는 AWS 서비스 및 리소스로 업데이트됩니다.

## 개발 환경에서 VS Code용 도구 키트로 작업

VS Code에서 개발 환경을 열거나 만든 후에는 VS Code의 로컬 인스턴스에서 작업하는 것과 마찬가지로 VS Code용 도구 키트에서 작업할 수 있습니다. VS Code에서 운영되는 개발 환경에 AWS 도구 키트를 설치하고 AWS Builder ID로 연결되도록 구성됩니다.

### 개발 환경 중지

#### 현재 개발 환경 중지

1. VS Code용 도구 키트에서 DEVELOPER TOOLS(개발자 도구) 탐색기를 확장합니다.
2. CodeCatalyst를 확장하고 Stop Dev Environment(개발 환경 중지)를 선택합니다.
3. VS Code에 메시지가 표시되면 개발 환경 중지를 선택합니다. 개발 환경을 중지하면 VS Code 원격 연결을 끊고 로컬 개발 인스턴스로 돌아갑니다.

### 개발 환경 설정 열기

#### 현재 개발 환경 설정 열기

1. VS Code용 도구 키트에서 DEVELOPER TOOLS(개발자 도구) 탐색기를 확장합니다.
2. CodeCatalyst를 확장하고 Open Settings(설정 열기)를 선택하여 현재 개발 환경에 대한 Dev Environment Settings(개발 환경 설정) 보기를 엽니다.
3. Dev Environment Settings(개발 환경 설정) 보기의 다음 섹션에는 개발 환경 관련 옵션이 포함되어 있습니다.
  - Alias(별칭): 개발 환경에 할당된 별칭을 보고 변경할 수 있습니다.
  - Status(상태): 현재 개발 환경 상태와 개발 환경에 할당된 프로젝트를 확인하고 개발 환경을 중지할 수 있습니다.
  - Devfile: 개발 환경용 Devfile의 이름과 위치를 볼 수 있습니다. Open in Editor(편집기에서 열기) 버튼을 선택하여 Devfile을 엽니다.
  - Compute Settings(컴퓨팅 설정): 개발 환경의 크기 및 기본 Timeout Length(제한 시간)을 변경할 수 있습니다.

**Note**

개발 환경이 생성되면 개발 환경에 할당된 스토리지 스페이스를 변경할 수 없습니다.

## 개발 환경 구성

개발 환경 devfile을 열고 현재 개발 환경을 구성하려면 다음 단계를 완료하세요. devfiles 수정에 대한 자세한 내용은 설명서의 [Configuring your Dev Environment with devfiles](#)(devfiles로 개발 환경 구성하기) 섹션에 있는 리소스를 참조하세요.

1. VS Code용 도구 키트에서 DEVELOPER TOOLS(개발자 도구) 탐색기를 확장합니다.
2. CodeCatalyst를 확장하고 Open Devfile(Devfile 열기)를 선택하여 현재 개발 환경의 새 편집기 창에서 devfile.yaml을 엽니다.

## Amazon CodeCatalyst 및 VS Code 문제 해결

다음 항목에서는 Amazon CodeCatalyst 및 VS Code 사용 시 발생할 수 있는 기술적 문제를 다룹니다.

### 주제

- [VS Code 버전](#)
- [Amazon CodeCatalyst 권한](#)
- [VS Code용 도구 키트에서 개발 환경로 연결](#)

### VS Code 버전

사용 중인 VS Code 버전은 시스템에서 vscode:// URI에 대한 핸들러를 설정해야 합니다. 이 핸들러가 없으면 AWS Toolkit은 모든 CodeCatalyst 기능에 액세스할 수 없습니다. VS Code Insiders에서 개발 환경을 시작할 때 오류가 발생할 수 있습니다. VS Code Insiders가 vscode-insiders:// URI는 처리하고 vscode:// URI는 처리하지 않기 때문입니다.

### Amazon CodeCatalyst 권한

AWS Toolkit for Visual Studio Code에서 CodeCatalyst를 사용하기 위한 파일 권한 요구 사항은 다음과 같습니다.

- `~/.ssh/config` 파일에 대한 액세스 권한을 `read` 및 `write` 로 설정합니다. 다른 모든 사용자에게 `write` 권한만 부여하세요.
- `~/.ssh/id_dsa` 및 `~/.ssh/id_rsa` 파일에 대한 액세스 권한을 `read`로만 설정합니다. 다른 모든 사용자에게 `read`, `write` 및 `execute` 권한만 부여하세요.
- `globals.context.globalStorageUri.fsPath` 파일은 쓰기 가능한 위치에 있어야 합니다.

## VS Code용 도구 키트에서 개발 환경로 연결

AWS Toolkit for Visual Studio Code에서 개발 환경에 연결하려고 시도할 때 다음 오류가 나타나는 경우

`~/.ssh/config`에 오래된 `aws-devenv-*` 섹션이 있을 수 있습니다.

- Open config. .(구성 열기) 버튼을 클릭하여 VS Code 편집기에서 `~/.ssh/config` 파일을 엽니다.
- 편집기에서 Host `aws-devenv-*` 섹션의 콘텐츠를 선택하고 삭제합니다.
- `~/.ssh/config`의 Host `aws-devenv-*`에 변경 사항을 저장합니다. 그런 다음에 파일을 닫습니다.
- VS Code용 도구 파일에서 개발 환경 연결을 다시 시도하세요.

## Amazon API Gateway로 실행하기

AWS Toolkit for Visual Studio Code를 사용하여 연결된 AWS 계정에서 원격 API Gateway 리소스를 검색하고 실행할 수 있습니다.

### Note

이 기능은 디버깅을 지원하지 않습니다.

원격 API Gateway 리소스 검색 및 실행하는 방법.

1. AWS탐색기에서 API Gateway를 선택하여 메뉴를 확장합니다. 원격 API Gateway 리소스가 표시됩니다.
2. 호출하려는 API Gateway 리소스를 찾아 컨텍스트 (마우스 오른쪽 클릭) 메뉴를 열고 호출하기 AWS를 선택하세요.
3. 파라미터 양식에서 호출 파라미터를 지정하세요.

- 원격 API Gateway 리소스를 실행하려면 호출을 선택하세요. 결과는 VS 코드 출력 보기에 표시됩니다.

## AWS App Runner 와 함께 사용 AWS Toolkit for Visual Studio Code

[AWS App Runner](#) 소스 코드 또는 컨테이너 이미지를 AWS 클라우드의 확장 가능하고 안전한 웹 애플리케이션으로 직접 배포할 수 있는 빠르고 간단하며 비용 효율적인 방법을 제공합니다. 이를 사용하면 새로운 기술을 배우거나, 사용할 컴퓨팅 서비스를 결정하거나, AWS 리소스를 프로비저닝하고 구성하는 방법을 몰라도 됩니다.

를 AWS App Runner 사용하여 소스 이미지 또는 소스 코드를 기반으로 서비스를 만들고 관리할 수 있습니다. 소스 이미지를 사용하는 경우 이미지 리포지토리에 저장된 퍼블릭 또는 프라이빗 컨테이너 이미지를 선택할 수 있습니다. App Runner는 다음 이미지 저장소 제공업체를 지원합니다.

- Amazon Elastic 컨테이너 레지스트리 (Amazon ECR): 계정에 프라이빗 이미지를 저장합니다. AWS
- Amazon Elastic Container Registry Public(Amazon ECR Public): 공개적으로 읽을 수 있는 이미지를 저장합니다.

소스 코드 옵션을 선택하면 지원되는 리포지토리 공급자가 유지 관리하는 소스 코드 리포지토리에서 배포할 수 있습니다. 현재 App Runner는 소스 코드 리포지토리 [GitHub](#) 공급자로 지원합니다.

### 필수 조건

를 사용하여 App Runner와 상호 작용하려면 AWS Toolkit for Visual Studio Code 다음이 필요합니다.

- 계정 AWS
- AWS Toolkit for Visual Studio Code 해당 기능의 한 버전 AWS App Runner

이러한 핵심 요구 사항 외에도 모든 관련 IAM 사용자에게 App Runner 서비스와 상호 작용할 수 있는 권한이 있는지 확인하십시오. 또한 컨테이너 이미지 URI 또는 GitHub 리포지토리 연결과 같은 서비스 소스에 대한 특정 정보를 얻어야 합니다. App Runner 서비스를 생성할 때 이 정보가 필요합니다.

### App Runner에 대한 IAM 권한 구성

App Runner에 필요한 권한을 부여하는 가장 쉬운 방법은 기존 AWS 관리형 정책을 관련 AWS Identity and Access Management (IAM) 개체, 특히 사용자 또는 그룹에 연결하는 것입니다. App Runner는 IAM 사용자에게 연결할 수 있는 2개의 관리형 정책을 제공합니다.

- `AWSAppRunnerFullAccess`: 사용자가 모든 App Runner 작업을 수행할 수 있도록 허용합니다.
- `AWSAppRunnerReadOnlyAccess`: 사용자가 App Runner 리소스에 대한 세부 정보를 나열하고 볼 수 있도록 허용합니다.

또한 Amazon Elastic Container Registry(Amazon ECR)에서 프라이빗 리포지토리를 서비스 소스로 선택하는 경우 App Runner 서비스에 대해 다음 액세스 역할을 생성해야 합니다.

- `AWSAppRunnerServicePolicyForECRAccess`: App Runner가 계정에서 Amazon Elastic Container Registry(Amazon ECR) 이미지에 액세스할 수 있도록 허용합니다.

Command Palette를 사용하여 서비스 인스턴스를 구성할 때 이 역할을 자동으로 생성할 수 있습니다.

#### Note

`AWSAppRunnerServicePolicyForECRAccess` 서비스 연결 역할을 통해 다음 작업을 AWS App Runner 완료할 수 있습니다.

- CloudWatch 로그를 Amazon Logs 로그 그룹에 푸시합니다.
- Amazon CloudWatch Events 규칙을 생성하여 Amazon Elastic Container 레지스트리(Amazon ECR) 이미지 푸시를 구독하십시오.

서비스 연동 역할을 수동으로 생성하지 않아도 됩니다. AWS App Runner 에서 호출하는 API 작업을 사용하여 AWS Management Console OR을 AWS App Runner 생성하면 이 서비스 연결 역할이 자동으로 생성됩니다. AWS Toolkit for Visual Studio Code

자세한 내용은 AWS App Runner 개발자 안내서에서 [App Runner ID 및 액세스 관리](#)를 참조하세요.

## App Runner에 대한 서비스 소스 얻기

AWS App Runner를 사용하여 소스 이미지 또는 소스 코드에서 서비스를 배포할 수 있습니다.

### Source image

소스 이미지에서 배포하는 경우 프라이빗 또는 퍼블릭 AWS 이미지 레지스트리에서 해당 이미지의 리포지토리로 연결되는 링크를 얻을 수 있습니다.

- Amazon ECR 프라이빗 레지스트리: <https://console.aws.amazon.com/ecr/repositories>에서 Amazon ECR 콘솔을 사용하는 프라이빗 리포지토리의 URI를 복사합니다.
- Amazon ECR 퍼블릭 레지스트리: <https://gallery.ecr.aws/>에서 Amazon ECR 퍼블릭 갤러리를 사용하는 퍼블릭 리포지토리의 URI를 복사합니다.

#### Note

VS Code용 도구 키트의 AWS 탐색기에서 비공개 Amazon ECR 리포지토리 URI도 얻을 수 있습니다.

- AWS Explorer를 열고 ECR 노드를 확장하여 해당 지역의 리포지토리 목록을 확인합니다. AWS
- 리포지토리를 마우스 오른쪽 클릭하고 리포지토리 URI 복사를 선택하면 링크가 클립보드에 복사됩니다.

VS Code의 Command Palette로 서비스 인스턴스를 구성할 때 이미지 리포지토리 URI를 지정합니다.

자세한 내용은 AWS App Runner 개발자 안내서의 [소스 이미지 기반의 App Runner 서비스](#)를 참조하세요.

#### Source code

소스 코드를 AWS App Runner 서비스에 배포하려면 지원되는 리포지토리 공급자가 유지 관리하는 Git 리포지토리에 해당 코드를 저장해야 합니다. App Runner는 하나의 소스 코드 리포지토리 제공자를 지원합니다. [GitHub](#)

GitHub 리포지토리 설정에 대한 자세한 내용은 [의 시작하기 설명서를 참조하십시오](#). GitHub

GitHub 리포지토리에서 App Runner 서비스에 소스 코드를 배포하기 위해 App Runner는 연결을 설정합니다. GitHub 저장소가 비공개인 경우 (즉, 공개적으로 액세스할 수 없는 경우 GitHub) 인 경우 App Runner에 연결 세부 정보를 제공해야 합니다.

#### Important

GitHub 연결을 만들려면 앱 러너 콘솔 (<https://console.aws.amazon.com/apprunner>) 을 사용하여 연결되는 GitHub 연결을 만들어야 합니다. AWS VS Code의 명령 팔레트를 사용하

여 서비스 인스턴스를 구성할 때 GitHub연결 페이지에서 사용 가능한 연결을 선택할 수 있습니다.

자세한 내용은 AWS App Runner 개발자 안내서의 [App Runner 연결 관리](#)를 참조하세요.

App Runner 서비스 인스턴스는 코드를 빌드하고 실행할 수 있는 관리형 런타임을 제공합니다. AWS App Runner 현재 다음과 같은 런타임을 지원합니다.

- Python 관리형 런타임
- Node.js 관리형 런타임

서비스 구성의 일부로 App Runner 서비스가 서비스를 빌드하고 시작하는 방법에 대한 정보를 제공합니다. 명령 팔레트를 사용하여 이 정보를 입력하거나 YAML 형식의 [App Runner 구성 파일](#)을 지정할 수 있습니다. 이 파일의 값은 App Runner에 서비스를 빌드 및 시작하고 런타임 컨텍스트를 제공하는 방법을 지시합니다. 여기에는 관련 네트워크 설정 및 환경 변수가 포함됩니다. 구성 파일의 이름이 `apprunner.yaml`로 지정되었습니다. 애플리케이션 리포지토리의 루트 디렉토리에 자동으로 추가됩니다.

## 요금

애플리케이션에서 사용하는 컴퓨팅 및 메모리 리소스에 대한 요금이 청구됩니다. 또한 배포를 자동화하는 경우 해당 월의 모든 자동화된 배포를 포함하는 각 애플리케이션에 대해 설정된 월별 요금도 지불합니다. 소스 코드에서 배포하기로 선택한 경우 App Runner가 소스 코드에서 컨테이너를 빌드하는 데 걸리는 시간만큼 빌드 비용을 추가로 지불합니다.

자세한 내용은 [AWS App Runner 요금](#)을 참조하세요.

### 주제

- [App Runner 서비스 생성](#)
- [App Runner 서비스 관리](#)

## App Runner 서비스 생성

AWS 탐색기와 VS Code의 Command Palette를 사용하여 VS Code용 도구 키트에서 App Runner 서비스를 생성할 수 있습니다. 특정 AWS 지역에서 서비스를 생성하기로 선택한 후에는 명령 팔레트에서



제공하는 번호가 매겨진 단계를 통해 애플리케이션이 실행되는 서비스 인스턴스를 구성하는 프로세스를 안내합니다.

App Runner 서비스를 생성하려면 [필수 조건](#)을 충족해야 합니다. 여기에는 관련 IAM 권한을 제공하고 배포하려는 특정 소스 리포지토리를 확인하는 작업이 포함됩니다.

## App Runner 서비스 생성

1. AWS 탐색기가 아직 열려 있지 않으면 탐색기를 엽니다.
2. App Runner 노드를 마우스 오른쪽 버튼으로 클릭하고 Create Service(서비스 생성)를 선택합니다.

명령 팔레트가 나타납니다.

3. Select a source code location type(소스 코드 위치 유형 선택)에서 ECR 또는 리포지토리를 선택합니다.

ECR을 선택하는 경우 Amazon Elastic Container Registry에서 유지 관리하는 리포지토리의 컨테이너 이미지를 지정합니다. 리포지토리(Repository)를 선택하는 경우 지원되는 리포지토리 공급자가 유지 관리하는 소스 코드 리포지토리를 지정합니다. 현재 App Runner는 소스 코드 리포지토리 [GitHub](#)공급자로 지원합니다.

## ECR에서 배포

1. 이미지 리포지토리 선택 또는 입력(Select or enter an image repository)에서 Amazon ECR 프라이빗 레지스트리 또는 Amazon ECR 퍼블릭 갤러리에서 유지 관리하는 이미지 리포지토리의 URL을 선택하거나 입력합니다.

### Note

Amazon ECR 퍼블릭 갤러리에서 리포지토리를 지정하는 경우 App Runner는 ECR 퍼블릭 리포지토리의 이미지에 대한 자동 배포를 지원하지 않으므로 자동 배포 기능이 꺼져 있는지 확인하세요.

자동 배포는 기본적으로 꺼져 있으며 Command Palette 헤더 아이콘에 대각선으로 표시됩니다. 자동 배포를 사용하기로 한 경우 추가 비용이 발생할 수 있다는 메시지가 표시됩니다.

2. Command Palette 단계에서 No tags found(태그를 찾을 수 없음)가 보고되면 태그가 지정된 컨테이너 이미지가 있는 리포지토리를 선택하는 단계로 돌아가세요.

3. Amazon ECR 사설 레지스트리를 사용하는 경우 앱 러너가 계정의 Amazon Elastic Container Registry (Amazon AppRunnerECR AccessRole) 이미지에 액세스할 수 있도록 허용하는 ECR 액세스 역할인 ECR이 필요합니다. Command Palette 헤더에서 “+” 아이콘을 선택하면 이 역할이 생성됩니다. (이미지가 공개적으로 제공되는 Amazon ECR 퍼블릭에 이미지가 저장되어 있는 경우에는 액세스 역할이 필요하지 않습니다.)
4. 포트에서 서비스에서 사용하는 IP 포트(예를 들어 포트 8000)를 입력합니다.
5. 환경 변수 구성(Configure environment variables)에서 서비스 인스턴스의 동작을 사용자 지정하는데 사용되는 환경 변수가 포함된 파일을 지정할 수 있습니다. 혹은 이 단계를 건너뛸 수 있습니다.
6. 서비스 이름 지정(Name your service)에서 공백 없이 고유한 이름을 입력하고 Enter를 누릅니다.
7. 인스턴스 구성 선택(Select instance configuration)에서 서비스 인스턴스의 CPU 유닛과 메모리(GB) 조합을 선택합니다.

서비스가 생성되면 상태가 생성(Creating)에서 실행(Running)으로 변경됩니다.

8. 서비스 실행을 시작한 후 서비스를 마우스 오른쪽 버튼으로 클릭하고 서비스 URL 복사(Copy Service URL)를 선택합니다.
9. 배포된 애플리케이션에 액세스하려면 복사한 URL을 웹 브라우저의 주소 표시줄에 붙여넣습니다.

#### 원격 리포지토리에서 배포

1. 연결 선택에서 연결되는 연결을 선택합니다. GitHub AWS선택 가능한 연결은 App Runner 콘솔의 GitHub 연결 페이지에 나열되어 있습니다.
2. 원격 GitHub 리포지토리 선택에서 원격 리포지토리의 URL을 선택하거나 입력합니다.

Visual Studio Code의 소스 제어 관리(SCM)로 구성되어 있는 원격 리포지토리를 선택할 수 있습니다. 목록에 없는 경우 리포지토리에 대한 링크를 붙여넣을 수도 있습니다.

3. 분기 선택(Select a branch)에서 배포할 소스 코드의 Git 분기를 선택합니다.
4. 구성 소스 선택(Choose configuration source)에서 런타임 구성을 정의하는 방식을 지정합니다.

구성 파일 사용(Use configuration file)을 선택한 경우 서비스 인스턴스는 `apprunner.yaml` 구성 파일에 의해 정의된 설정으로 구성됩니다. 이 파일은 애플리케이션 리포지토리의 루트 디렉터리에 있습니다.

여기서 모든 설정 구성(Configure all settings here)을 선택한 경우 Command palette을 사용하여 다음을 지정합니다.

- 런타임(Runtime): Python 3 또는 Nodejs 12를 선택합니다.

- 빌드 명령(Build command): 서비스 인스턴스의 런타임 환경에서 애플리케이션을 빌드하는 명령을 입력합니다.
  - 시작 명령(Start command): 서비스 인스턴스의 런타임 환경에서 애플리케이션을 시작하는 명령을 입력합니다.
5. 포트(Port)에서 서비스에서 사용하는 IP 포트(예를 들어 포트 8000)를 입력합니다.
  6. 환경 변수 구성(Configure environment variables)에서 서비스 인스턴스의 동작을 사용자 지정하는데 사용되는 환경 변수가 포함된 파일을 지정할 수 있습니다. 혹은 이 단계를 건너뛸 수 있습니다.
  7. 서비스 이름 지정(Name your service)에서 공백 없이 고유한 이름을 입력하고 Enter를 누릅니다.
  8. 인스턴스 구성 선택(Select instance configuration)에서 서비스 인스턴스의 CPU 유닛과 메모리(GB) 조합을 선택합니다.

서비스가 생성되면 상태가 생성(Creating)에서 실행(Running)으로 변경됩니다.

9. 서비스 실행을 시작한 후 서비스를 마우스 오른쪽 버튼으로 클릭하고 서비스 URL 복사(Copy Service URL)를 선택합니다.
10. 배포된 애플리케이션에 액세스하려면 복사한 URL을 웹 브라우저의 주소 표시줄에 붙여넣습니다.

#### Note

App Runner 서비스를 만들지 못했다면 AWS 탐색기에 서비스 생성 실패(Create failed) 상태가 표시됩니다. 문제 해결 팁은 App Runner 개발자 안내서에서 [서비스 생성을 실패한 경우\(When service creation fails\)](#)를 참조하세요.

## App Runner 서비스 관리

App Runner 서비스를 만든 후에는 AWS 탐색기 창에서 다음 활동을 수행하여 서비스를 관리할 수 있습니다.

- [App Runner 서비스 일시 중지 및 다시 시작](#)
- [App Runner 서비스 배포](#)
- [App Runner에 대한 로그 스트림 보기](#)
- [App Runner 서비스 삭제](#)

## App Runner 서비스 일시 중지 및 다시 시작

웹 애플리케이션을 일시적으로 사용하지 않도록 설정하고 코드 실행을 중지해야 하는 경우 AWS App Runner 서비스를 일시 중지할 수 있습니다. App Runner는 서비스에 대한 컴퓨팅 용량을 0으로 줄입니다. 애플리케이션을 다시 실행할 준비가 되면 App Runner 서비스를 다시 시작합니다. App Runner는 새로운 컴퓨팅 파워를 프로비저닝하고, 애플리케이션을 배포한 후 애플리케이션을 실행합니다.

### Important

App Runner가 실행 중일 때만 요금이 청구됩니다. 따라서 비용을 관리하는 데 필요한 경우 애플리케이션을 일시 중지했다가 다시 시작할 수 있습니다. 이는 개발 및 테스트 시나리오에서 특히 유용합니다.

### App Runner 서비스 일시 중지

1. AWS 탐색기가 아직 열려 있지 않으면 탐색기를 여십시오.
2. App Runner를 확장하여 서비스 목록을 봅니다.
3. 서비스를 마우스 오른쪽 버튼으로 클릭하고 Pause(일시 중지)를 선택합니다.
4. 표시되는 대화 상자에서 확인(Confirm)을 선택합니다.

서비스가 일시 중지되는 동안 서비스 상태는 실행 중(Running)에서 일시 중지 중(Pausing)으로 변한 다음 일시 중지됨(Paused)으로 변경됩니다.

### App Runner 서비스 다시 시작

1. AWS 탐색기가 아직 열려 있지 않으면 해당 탐색기를 엽니다.
2. App Runner를 확장하여 서비스 목록을 봅니다.
3. 서비스를 마우스 오른쪽 버튼으로 클릭하고 다시 시작(Resume)을 선택합니다.

서비스가 다시 시작되는 동안 서비스 상태가 다시 시작 중(Resuming)에서 실행 중(Running)으로 변경됩니다.

### App Runner 서비스 배포

서비스에 대한 수동 배포 옵션을 선택하는 경우 서비스에 대한 각 배포를 명시적으로 시작해야 합니다.

1. AWS 탐색기가 아직 열려 있지 않으면 해당 탐색기를 엽니다.

2. App Runner를 확장하여 서비스 목록을 봅니다.
3. 서비스를 마우스 오른쪽 버튼으로 클릭하고 배포 시작(Start Deployment)을 선택합니다.
4. 애플리케이션이 배포되는 동안 서비스 상태가 배포 중(Deploying)에서 실행 중(Running)으로 변경됩니다.
5. 애플리케이션이 성공적으로 배포되었는지 확인하려면 동일한 서비스를 마우스 오른쪽 버튼으로 클릭하고 서비스 URL 복사(Copy Service URL)를 선택합니다.
6. 배포한 웹 애플리케이션에 액세스하려면 복사한 URL을 웹 브라우저의 주소 표시줄에 붙여넣습니다.

## App Runner에 대한 로그 스트림 보기

CloudWatch 로그를 사용하여 App Runner와 같은 서비스의 로그 스트림을 모니터링, 저장 및 액세스할 수 있습니다. 로그 스트림은 동일한 소스를 공유하는 로그 이벤트 시퀀스입니다.

1. App Runner를 확장하여 서비스 인스턴스 목록을 봅니다.
2. 특정 서비스 인스턴스를 확장하여 로그 그룹 목록을 봅니다. (로그 그룹은 동일한 보존, 모니터링 및 액세스 제어 설정을 공유하는 로그 스트림 그룹입니다.)
3. 로그 그룹을 마우스 오른쪽 버튼으로 클릭하고 로그 스트림 보기(View Log Streams)를 선택합니다.
4. Command Palette에서 그룹의 로그 스트림을 선택합니다.

VS Code 편집기에 스트림을 구성하는 로그 이벤트 목록이 나타납니다. 이전 이벤트 또는 최신 이벤트를 편집기에 로드할 수 있습니다.

## App Runner 서비스 삭제

### Important

App Runner 서비스를 삭제하면 영구적으로 제거되고 저장된 데이터가 삭제됩니다. 서비스를 다시 생성해야 하는 경우 App Runner는 소스를 다시 가져와 코드 리포지토리인 경우 빌드해야 합니다. 웹 애플리케이션은 새로운 App Runner 도메인을 가져옵니다.

1. AWS 탐색기가 아직 열려 있지 않으면 탐색기를 여십시오.
2. App Runner를 확장하여 서비스 목록을 봅니다.

3. 서비스를 마우스 오른쪽 단추로 클릭하고 서비스 삭제(Delete Service)를 선택합니다.
4. Command Palette에서 삭제(delete)를 입력한 다음 Enter 키를 누릅니다.

삭제된 서비스는 삭제 중(Deleting)상태로 표시된 후 목록에서 사라집니다.

## AWS Application Composer

AWS Toolkit for Visual Studio Code을 사용하여 AWS Application Composer 서비스로 작업할 수 있습니다. AWS Application Composer는 애플리케이션 아키텍처를 설계하고 AWS CloudFormation 인프라를 시각화하는 데 도움을 주는 AWS 애플리케이션용 비주얼 빌더입니다.

AWS Application Composer 서비스에 대한 자세한 내용은 [AWSApplication Composer](#) 사용 설명서를 참조하세요.

다음 주제에서는 AWS Toolkit for Visual Studio Code에서 AWS Application Composer로 작업하는 방법을 설명합니다.

### 주제

- [Toolkit에서 AWS Application Composer로 작업](#)

## Toolkit에서 AWS Application Composer로 작업

AWS Toolkit for Visual Studio Code용 AWS Application Composer를 사용하면 대화형 캔버스를 통해 애플리케이션을 시각적으로 디자인할 수 있습니다. 또한 Application Composer를 사용하여 AWS CloudFormation 및 AWS Serverless Application Model(AWS SAM) 템플릿을 시각화하고 수정할 수 있습니다. Application Composer로 작업하는 동안 변경 사항은 영구적으로 저장되므로 VS Code 편집기에서 직접 파일을 편집하거나 대화형 캔버스를 사용하여 원활하게 전환할 수 있습니다.

AWS Application Composer 서비스에 대한 자세한 정보, 시작하기 정보 및 자습서는 [AWS Application Composer 서비스](#) 사용 설명서를 참조하세요.

다음 섹션에서는 AWS Toolkit for Visual Studio Code에서 AWS Application Composer 서비스에 액세스하는 방법을 설명합니다.

## Toolkit에서 AWS Application Composer에 액세스

Toolkit에서 AWS Application Composer에 액세스할 수 있는 방법은 크게 3가지가 있습니다.

기존 템플릿에서 AWS Application Composer에 액세스

1. VS Code 편집기에서 기존 템플릿 파일을 엽니다.
2. 편집기 창에서 편집기 창 오른쪽 상단에 있는 AWS Application Composer 버튼을 클릭합니다.
3. AWS Application Composer가 VS Code 편집기 창에서 템플릿 파일을 열고 시각화합니다.

컨텍스트 메뉴(마우스 오른쪽 클릭)에서 AWS Application Composer에 액세스

1. VS Code에서 AWS Application Composer로 열리는 템플릿 파일을 마우스 오른쪽 버튼으로 클릭합니다.
2. 컨텍스트 메뉴에서 App Composer로 열기 옵션을 선택합니다.
3. AWS Application Composer가 새 VS Code 편집기 창에서 템플릿 파일을 열고 시각화합니다.

명령 팔레트에서 AWS Application Composer에 액세스

1. VS Code에서 **Cmd + Shift + P** 또는 **Ctrl + Shift + P**(Windows)를 눌러 명령 팔레트를 엽니다.
2. 검색 필드에 **AWS Application Composer**를 입력하고 결과가 표시되면 AWS Application Composer를 선택합니다.
3. 열리는 템플릿 파일을 선택하면 AWS Application Composer가 새 VS Code 편집기 창에서 템플릿 파일을 열고 시각화합니다.

## AWS CDK VS 코드

이 시험판 설명서는 미리 보기 버전 기능에 관한 것입니다. 이 시험판 설명서는 변경될 수 있습니다.

AWS CDK 서비스를 사용하면 [AWS Cloud Development Kit \(AWS CDK\)](#) 애플리케이션 또는 앱 작업을 수행할 수 있습니다. AWS CDK에 대한 자세한 내용은 [AWS Cloud Development Kit \(AWS CDK\) 개발자 안내서](#)에서 확인할 수 있습니다.

AWS CDK 앱은 AWS CloudFormation 스택과 AWS 리소스에 대한 정의를 포함하는 [구성](#)라는 빌딩 블록으로 구성됩니다. AWS CDK 탐색기를 사용하면 AWS CDK 구성에서 정의된 [스택](#) 및 [리소스](#)를 시각화할 수 있습니다. 이 시각화는 Visual Studio 코드 (VS 코드) 편집기의 개발자 도구 창에 있는 트리 보기로 볼 수 있습니다.

이 섹션에서는 VS 코드 편집기에서 AWS CDK 를 액세스하고 사용하는 방법에 대한 정보를 제공합니다. 로컬 IDE에 Toolkit for VS Code를 [설치 및 구성](#)했다고 가정합니다.

주제

- [AWS CDK 애플리케이션 작업](#)

## AWS CDK 애플리케이션 작업

이 시험판 설명서는 미리 보기 버전 기능에 관한 것입니다. 이 시험판 설명서는 변경될 수 있습니다.

AWS Toolkit for VS Code AWS CDK탐색기를 사용하여 AWS CDK 애플리케이션을 시각화하고 작업할 수 있습니다.

### 필수 조건

- [Toolkit for VS Code 설치](#)에 명시된 필수 조건에 맞는 시스템인지 확인합니다.
- AWS Cloud Development Kit (AWS CDK)개발자 안내서의 [AWS CDK 시작하기](#) 처음 몇 섹션에 설명된대로 AWS CDK 명령줄 인터페이스를 설치합니다.

#### Important

AWS CDK 버전은 1.17.0 이상이어야 합니다. 명령줄의 **cdk --version**을 사용하여 실행 중인 버전을 확인합니다.

## AWS CDK 애플리케이션 시각화

AWS VS 코드 AWS CDK 탐색기 도구 키트를 사용하면 애플리케이션의 CDK 구조에 저장된 [스택](#) 및 [리소스](#)를 관리할 수 있습니다. AWS CDK탐색기는 **cdk synth** 명령을 실행할 때 생성되는 tree.json 파일에 정의된 정보를 사용하여 리소스를 트리 보기로 표시합니다. 기본적으로 애플리케이션의 cdk.out 디렉터리에 tree.json 파일이 있습니다.

도구 키트 AWS CDK 탐색기를 사용하려면 CDK 애플리케이션을 생성하세요.

1. [AWS CDK 개발자 안내서](#)에 수록된 [Hello World 자습서](#) 처음 몇 단계를 완료하세요.



**⚠ Important**

스택 배포 단계에서 중지하고 개발자 안내서를 확인하세요.

**ℹ Note**

자습서에 있는 명령어(예: `mkdir` 및 `cdk init`)를 운영 체제 명령줄이나 VS 코드 편집기의 터미널창에서 실행할 수 있습니다.

2. CDK 자습서의 필수 단계를 완료한 후 VS 코드 편집기에서 생성한 CDK 콘텐츠를 여세요.
3. AWS 탐색 창에서 CDK (미리보기) 제목을 펼치세요. 이제 CDK 애플리케이션 및 관련 리소스가 CDK 탐색기 트리 보기에 표시됩니다.

**중요 정보**

- CDK 애플리케이션을 편집기로 열 때, 여러 폴더를 한꺼번에 불러올 수 있습니다. 앞의 이미지와 같이 폴더는 CDK 앱을 여러 개 포함할 수 있습니다. AWS CDK 탐색기는 프로젝트 루트 디렉터리 및 하위 디렉터리에서 앱을 찾습니다.
- 이 자습서의 처음 몇 단계를 수행할 때 마지막으로 실행하는 명령이 `cdk synth`이므로 `tree.json` 파일이 생성된다는 사실을 알 수 있습니다. 예를 들어, 리소스 추가 등 CDK 앱 관련 사항을 변경하는 경우 해당 명령을 다시 실행하여 트리 보기에 반영된 변경 사항을 확인해야 합니다.

**AWS CDK 앱에서 기타 작업 수행**

운영 체제의 명령줄이나 다른 도구를 사용하는 것처럼 VS Code 편집기를 사용하여 CDK 앱에서 다른 작업을 수행할 수 있습니다. 예를 들어, 편집기에서 코드 파일을 업데이트하고 VS Code 터미널 창을 사용하여 앱을 설치할 수 있습니다.

이러한 유형의 작업을 시험해 보려면 VS Code 편집기를 사용하여 AWS CDK 개발자 안내서에 수록된 [Hello World 자습서](#)를 계속 수행하세요. 사용자 AWS 계정에 예상치 못한 비용이 발생하지 않도록 앱 리소스를 최종적으로 삭제하세요.

## AWS CloudFormation 스택 작업

AWS Toolkit for Visual Studio Code는 [AWS CloudFormation](#) 스택을 지원합니다. VS Code 도구 키트를 사용하면 스택 삭제 같은 특정 AWS CloudFormation 스택 작업을 수행할 수 있습니다.

주제

- [AWS CloudFormation 스택 삭제](#)
- [틀 사용하여 AWS CloudFormation 템플릿 만들기 AWS Toolkit for Visual Studio Code](#)

## AWS CloudFormation 스택 삭제

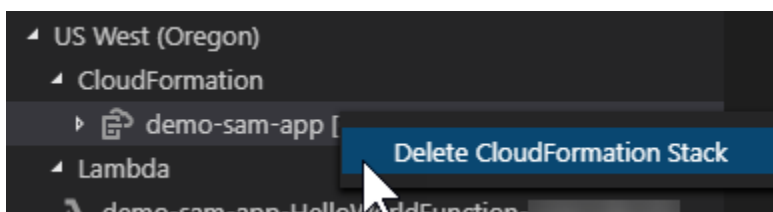
AWS Toolkit for Visual Studio Code를 사용하여 AWS CloudFormation 스택을 삭제할 수 있습니다.

필수 조건

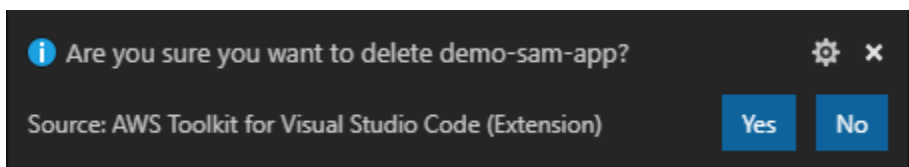
- [Toolkit for VS Code 설치](#)에서 명시한 조건을 충족하는 시스템인지 확인하세요.
- [인증 및 액세스](#)에서 구성한 자격 증명에 AWS CloudFormation 서비스에 대한 적절한 읽기/쓰기 액세스 권한이 포함되어 있는지 확인하세요. AWS Explorer의 CloudFormation에서 'CloudFormation 리소스를 로드하는 동안 오류가 발생했습니다(Error loading CloudFormation resources)'와 유사한 메시지가 표시되면 해당 자격 증명에 연결된 권한을 확인합니다. 권한을 변경한 경우 VS 코드 AWS Explorer에 적용되는 데 몇 분 정도 걸립니다.

## CloudFormation 스택 삭제

1. AWS Explorer에서 삭제할 AWS CloudFormation 스택의 컨텍스트 메뉴를 엽니다.



2. CloudFormation 스택 삭제를 선택합니다.
3. 나타나는 메시지에서 예를 선택하여 삭제되었는지 확인합니다.



스택이 삭제되면 더 이상 AWS Explorer에 나열되지 않습니다.

## 를 사용하여 AWS CloudFormation 템플릿 만들기 AWS Toolkit for Visual Studio Code

템플릿을 AWS CloudFormation 작성하고 SAM 템플릿을 작성하는 데 도움이 AWS Toolkit for Visual Studio Code 될 수 있습니다.

### 필수 조건

#### VS Code용 도구 키트 및 자격 증명 필수 요건

- VS Code용 Toolkit에서 CloudFormation 서비스에 액세스하려면 먼저 [사용자 안내서 VS Code용 툴킷 설치에 설명된 요구 사항을 충족해야 합니다.](#)
- 에서 생성한 자격 증명에는 서비스에 대한 적절한 읽기/쓰기 권한이 [인증 및 액세스](#) 포함되어야 합니다. AWS CloudFormation

#### Note

CloudFormation 서비스에 CloudFormation 리소스 로드 중 오류 메시지가 표시되면 해당 자격 증명에 연결한 권한을 확인하세요. 또한 권한 변경 사항이 AWS Explorer에서 업데이트되는 데 몇 분 정도 걸릴 수 있습니다.

#### CloudFormation 템플릿 사전 요구 사항

- [Redhat Developer YAML VS Code](#) 확장 프로그램을 설치하고 실행하세요.
- Redhat Developer YAML VS Code 확장 프로그램을 사용할 때는 인터넷에 연결되어 있어야 합니다. 이 확장 프로그램은 컴퓨터에 JSON 스키마를 다운로드하고 캐시화하는 데 사용되기 때문입니다.

## YAML 스키마 Support를 통한 CloudFormation 템플릿 작성

이 툴킷은 YAML 언어 지원 및 JSON 스키마를 사용하여 SAM 템플릿 작성 및 SAM 템플릿 작성 프로세스를 간소화합니다. CloudFormation 구문 검증 및 자동 완성과 같은 기능은 프로세스를 더 빠르게 할 뿐만 아니라 템플릿 품질을 개선하는 데도 도움이 됩니다. 다음은 템플릿의 스키마를 선택할 때 권장하는 모범 사례입니다.

## CloudFormation 템플릿

- 파일 확장자는 .yaml 또는 .yml입니다.
- 파일에는 최상위 AWSTemplateFormatVersion 또는 리소스 노드가 있습니다.

## SAM 템플릿

- 에 대한 모든 기준은 이미 설명되어 있습니다. CloudFormation
- 파일에는 AWS::Serverless로 시작하는 값을 포함한 최상위 변환 노드가 있습니다.

스키마는 파일 수정 시 적용됩니다. 예를 들어, 템플릿에 서버리스 변환을 추가하고 파일을 저장한 후에 SAM CloudFormation 템플릿 스키마가 적용됩니다.

## 구문 유효성 검사

YAML 확장 프로그램은 템플릿에 형식 유효성 검사를 자동으로 실행합니다. 이렇게 하면 지정된 속성의 형식이 잘못된 항목이 강조 표시됩니다. 강조 표시된 항목 위로 마우스를 가져가면 확장 프로그램에 수정 조치가 표시됩니다.

## 자동 완성

새 필드, 나열된 값 또는 기타 [리소스 유형](#)을 추가할 때 Ctrl + space로 YAML 확장 프로그램의 자동 완성 기능을 사용할 수 있습니다.

# AWS Toolkit for Visual Studio Code를 사용하여 CloudWatch Logs 작업

Amazon CloudWatch Logs는 확장성이 뛰어난 단일 서비스에서 사용하는 모든 시스템, 애플리케이션 및 AWS 서비스에서 로그를 중앙 집중화할 수 있습니다. 그런 다음 로그를 쉽게 보고, 특정 오류 코드 또는 패턴이 있는지 검색하고, 특정 필드를 기반으로 필터링하거나, 향후 분석을 위해 안전하게 보관할 수 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서에서 [Amazon CloudWatch Logs란 무엇입니까?](#)를 참조하세요.

다음 주제에서는 AWS 계정에서 AWS Toolkit for Visual Studio Code를 사용하여 CloudWatch Logs 작업을 수행하는 방법을 설명합니다.

## 주제

- [AWS Toolkit for Visual Studio Code를 사용하여 CloudWatch 로그 그룹 및 로그 스트림 보기](#)
- [AWS Toolkit for Visual Studio Code를 사용하여 로그 스트림의 CloudWatch 로그 이벤트 작업](#)
- [CloudWatch 로그 그룹 검색](#)

## AWS Toolkit for Visual Studio Code를 사용하여 CloudWatch 로그 그룹 및 로그 스트림 보기

로그 스트림은 동일한 소스를 공유하는 로그 이벤트 시퀀스입니다. CloudWatch Logs의 개별 로그 소스는 각각의 로그 스트림을 구성합니다.

로그 그룹은 동일한 보존 기간, 모니터링 및 액세스 제어 설정을 공유하는 로그 스트림 그룹입니다. 로그 그룹을 정의하고 각 그룹에 배치할 스트림을 지정할 수 있습니다. 하나의 로그 그룹이 가질 수 있는 로그 스트림의 수는 제한이 없습니다.

자세한 내용은 Amazon CloudWatch 사용 설명서에서 [Log Groups 및 Log Streams 작업](#)을 참조하세요.

주제

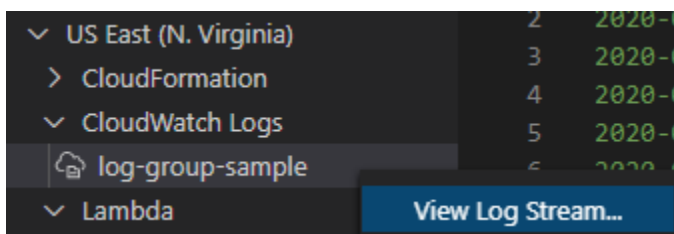
- [CloudWatch Logs 노드를 사용하여 로그 그룹 및 로그 스트림 보기](#)

### CloudWatch Logs 노드를 사용하여 로그 그룹 및 로그 스트림 보기

1. VS Code에서 보기, 탐색기를 선택하여 AWS Explorer를 엽니다.
2. CloudWatch Logs 노드를 클릭하여 로그 그룹 목록을 확장합니다.

CloudWatch Logs 노드 아래에 현재 AWS 리전 로그 그룹이 표시됩니다.

3. 로그 그룹 이름을 마우스 우클릭 클릭한 다음 View Log Streams(로그 스트림 보기)를 선택하면 로그 그룹의 로그 스트림을 볼 수 있습니다.



4. Command Palette에서 보려고 하는 그룹의 로그 스트림을 선택합니다.

**Note**

Command Palette에 각 스트림의 최근 이벤트에 대한 타임스탬프가 표시됩니다.

[Log Stream 편집기](#)가 실행되어 스트림의 로그 이벤트가 표시됩니다.

## AWS Toolkit for Visual Studio Code를 사용하여 로그 스트림의 CloudWatch 로그 이벤트 작업

Log Stream 편집기를 열면 각 스트림의 로그 이벤트에 액세스할 수 있습니다. 로그 이벤트는 모니터링 중인 애플리케이션 또는 리소스에 의해 기록된 활동의 기록입니다.

주제

- [로그 스트림 정보 보기 및 복사](#)
- [로그 스트림 편집기의 콘텐츠를 로컬 파일에 저장](#)

### 로그 스트림 정보 보기 및 복사

로그 스트림을 열면 Log Stream 편집기에 스트림의 로그 이벤트 시퀀스가 표시됩니다.

1. Log Stream 편집기를 열면 보려는 로그 스트림을 찾을 수 있습니다([CloudWatch 로그 그룹 및 로그 스트림 보기](#) 참조).

이벤트가 나열된 각 줄에는 기록된 시점을 알 수 있는 타임스탬프가 있습니다.

2. 다음 옵션을 사용하여 스트림 이벤트에 대한 정보를 보고 복사할 수 있습니다.
  - 시간순으로 이벤트 보기: Load newer events(최근 이벤트 로드) 또는 Load older events(이전 이벤트 로드)를 선택하여 최근 로그 이벤트 및 이전 로그 이벤트를 표시합니다.

**Note**

Log Stream 편집기는 처음에 최근 10,000개 줄의 로그 이벤트 또는 1MB의 로그 데이터 (둘 중 용량이 작은 것)의 배치를 로드합니다. Load newer events(최근 이벤트 로드)를 선택하면 편집기에 마지막 배치가 로드된 이후 기록된 이벤트가 표시됩니다. Load older

events(이전 이벤트 로드)를 선택하면 편집기에 현재 표시된 이벤트보다 이전에 발생한 이벤트 배치가 표시됩니다.

- 로그 이벤트 복사: 복사할 이벤트를 선택한 다음 마우스 우클릭하고 메뉴에서 Copy(복사)를 선택합니다.
- 로그 스트림 이름 복사: Log Stream 편집기의 탭을 마우스 우클릭하고 Log Stream Name(로그 스트림 이름)을 선택합니다.

#### Note

Command Palette를 사용하여 AWSToolkit Copy Log Stream Name을 실행해도 됩니다.

## 로그 스트림 편집기의 콘텐츠를 로컬 파일에 저장

CloudWatch 로그 스트림 편집기의 콘텐츠를 로컬 시스템의 log 파일로 다운로드할 수 있습니다.

#### Note

이 옵션을 사용할 경우 현재 로그 스트림 편집기에 표시된 로그 이벤트만 파일에 저장할 수 있습니다. 예를 들어 로그 스트림의 총 크기가 5MB인데 2MB만 편집기에 로드된 경우 2MB 로그 데이터만 파일에 저장됩니다. 저장할 데이터를 더 표시하려면 편집기에서 Load newer events(최근 이벤트 로드) 또는 Load older events(이전 이벤트 로드)를 선택하세요.

1. Log Streams 편집기를 열면 복사할 로그 스트림을 찾을 수 있습니다([CloudWatch 로그 그룹 및 로그 스트림 보기](#) 참조).
2. 로그 스트림 이름이 표시된 탭 옆에 있는 Save(저장) 아이콘을 선택하세요.

#### Note

Command Palette를 사용하여 AWS Toolkit Save Current Log Stream Content( 도구 키트로 현재 로그 스트림 내용 저장)를 실행합니다.

3. 이 대화 상자에서 로그 파일의 다운로드 폴더를 선택하거나 생성한 다음 Save(저장)를 클릭하세요.

## CloudWatch 로그 그룹 검색

Search Log Group을 사용하여 로그 그룹의 모든 로그 스트림을 검색할 수 있습니다.

Amazon CloudWatch Logs 서비스에 대한 자세한 내용을 알아보려면 Amazon CloudWatch 사용 설명서에서 [Working with Log Groups and Log Streams](#)(로그 그룹 및 로그 스트림 작업)를 참조하세요.

### VS Code Command Palette에서 로그 그룹 검색

다음 단계를 완료하여 VS Code Command Palette에서 로그 그룹을 검색하세요.

Amazon CloudWatch Logs 필터 및 패턴 구문에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 [Filter and pattern syntax](#)(필터 및 패턴 구문) 섹션을 참조하세요.

1. VS Code에서 **cmd+shift+p** (windows:**ctrl+shift+p**)를 눌러 Command Palette를 엽니다.
2. Command Palette에 **AWS: Search Log Group** 명령을 입력하여 search log group dialog(로그 그룹 대화상자 검색)를 열고 지시에 따라 계속 진행하세요.

#### Note

첫 번째 프롬프트에는 다음 단계로 진행하기 전에 AWS 리전을 전환할 수 있는 옵션이 있습니다.

3. Select Log Group (1/3) 프롬프트에서 검색하려는 로그 그룹을 선택합니다.
4. Select Time Filter (2/3) 프롬프트에서 검색에 적용할 시간 필터를 선택합니다.
5. Search Log Group... (3/3) 프롬프트에서 제공된 필드에 검색 패턴 구문을 입력한 다음 **Enter** 키를 눌러 계속 진행하거나 **ESC** 키를 눌러 검색을 취소하세요.
6. 검색을 완료하면 VS Code 편집기에 검색 결과가 나타납니다.

### AWS Explorer에서 로그 그룹 검색

다음 단계를 완료하여 AWS Toolkit for Visual Studio Code Explorer에서 로그 그룹을 검색하세요.

1. AWS Toolkit for Visual Studio Code Explorer에서 CloudWatch를 확장합니다.
2. 마우스 우클릭으로 검색할 로그 그룹 검색 컨텍스트 메뉴를 열고 Search Log Group(로그 그룹 검색)을 선택하여 검색 프롬프트를 엽니다.
3. 타임 프레임을 선택하여 프롬프트를 진행하세요.



4. 메시지가 표시되면 제공된 필드에 검색 패턴 구문을 입력한 다음 **Enter** 키를 눌러 계속하거나 **ESC** 키를 눌러 검색을 취소하세요.
5. 검색을 완료하면 VS Code 편집기에서 검색 결과가 나타납니다.

## 검색 로그 결과 사용

CloudWatch 로그 그룹 검색을 성공적으로 완료하면 VS Code 편집기에 검색 결과가 나타납니다. 다음 절차는 검색 로그 결과를 사용하는 방법을 설명합니다.

### Note

단일 로그 스트림을 볼 때 다음 기능은 현재 활성 로그 스트림의 결과로 한정합니다.

## 검색 로그 그룹 결과 저장

다음 단계를 완료하여 검색 로그 그룹 결과를 로컬에 저장하세요.

1. 검색 로그 그룹 결과에서 VS Code 편집기의 오른쪽 상단에 있는 Save Log to File(파일에 로그 저장) 아이콘 버튼을 선택합니다.
2. Save As 프롬프트에서 저장할 파일의 이름과 위치를 지정합니다.
3. OK를 선택하면 파일은 로컬 시스템에 저장됩니다.

## 시간 범위 변경

검색 로그 그룹 결과의 활성 상태인 시간 범위를 변경하려면 다음 단계를 완료하세요.

1. 검색 로그 그룹 결과에서 VS Code 편집기의 우측 상단 가장자리에 있는 Search by date...(날짜별 검색) 아이콘을 선택합니다.
2. Select Time Filter 프롬프트에서 검색 로그 결과의 새 시간 범위를 선택합니다.
3. Select Time Filter 프롬프트를 닫으면 결과가 업데이트됩니다.

## 검색 패턴 변경

검색 로그 그룹 결과의 활성화된 검색 패턴을 변경하려면 다음 단계를 완료하세요.

1. 검색 로그 그룹 결과에서 VS Code 편집기의 우측 상단 가장자리에 있는 Search by Pattern...(패턴별 검색) 아이콘을 선택합니다.

2. Search Log Group(로그 그룹 검색) 프롬프트 필드에 새 검색 패턴을 입력합니다.
3. **Enter** 키를 눌러 프롬프트를 닫으면 새 검색 패턴으로 결과가 업데이트됩니다.

## Amazon Elastic Container Registry 서비스 사용

Amazon Elastic Container Registry(Amazon ECR)는 안전하고 확장 가능한 AWS 관리형 컨테이너 레지스트리 서비스입니다. VS Code용 도구 키트 탐색기에서 여러 Amazon ECR 서비스 기능을 액세스할 수 있습니다.

- 리포지토리 생성
- 리포지토리 또는 태그가 지정된 이미지를 위한 AWS App Runner 서비스 생성
- 이미지 태그 및 리포지토리 URI 또는 ARN에 액세스
- 이미지 태그 및 리포지토리 삭제

또한 AWS CLI 및 다른 플랫폼과 VS Code를 통합하여 VS Code 콘솔을 통해 모든 Amazon ECR 기능에 액세스할 수 있습니다.

Amazon ECR에 대한 자세한 내용은 Amazon Elastic Container Registry 사용 설명서의 [Amazon ECR 이란?](#)을 참조하세요.

### 필수 조건

이 방법으로 VS Code 탐색기에서 Amazon ECR 서비스에 액세스할 수 있습니다.

#### IAM 사용자 생성

Amazon ECR과 같은 AWS 서비스에 액세스하려면 자격 증명을 제공하세요. 서비스 사용에 필요한 리소스 액세스 권한이 있는지 알 수 있습니다. 루트 AWS 계정의 자격 증명을 사용하여 AWS에 액세스하지 않는 것이 좋습니다. 대신 AWS Identity and Access Management (IAM)으로 IAM 사용자를 생성하여 관리자 권한이 있는 IAM 그룹에 추가하세요. 그러면 IAM 사용자의 특정 URL과 자격 증명을 사용하여 AWS에 액세스할 수 있습니다.

AWS에 가입했지만 IAM 사용자를 생성하지 않았다면 IAM 콘솔에서 생성할 수 있습니다.

관리자 사용자를 생성하려면 다음 옵션 중 하나를 선택합니다.

관리자를 관리하는 방법 한 가지 선택	To	By	다른 방법
IAM Identity Center에서 (권장)	단기 보안 인증 정보를 사용하여 AWS에 액세스합니다.  이는 보안 모범 사례와 일치합니다. 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 <a href="#">IAM 보안 모범 사례</a> 를 참조하세요.	AWS IAM Identity Center 사용 설명서의 <a href="#">시작하기</a> 지침을 따르세요.	AWS Command Line Interface 사용 설명서의 <a href="#">AWS IAM Identity Center 사용할 AWS CLI 구성</a> 을 통해 프로그래밍 방식의 액세스를 구성합니다.
IAM에서 (권장되지 않음)	장기 보안 인증 정보를 사용하여 AWS에 액세스합니다.	IAM 사용 설명서의 <a href="#">첫 IAM 관리 사용자 및 사용자 그룹 만들기</a> 에 나온 지침을 따릅니다.	IAM 사용 설명서에 나온 <a href="#">IAM 사용자의 액세스 키 관리</a> 단계를 수행하여 프로그래밍 방식의 액세스를 구성합니다.

AWS 콘솔에서 로그아웃한 후 다음 URL에서 새 IAM 사용자로 로그인하세요. 다음 URL의 `your_aws_account_id`에서 하이픈을 제외한 것이 AWS 계정 번호가입니다(예: AWS 계정 번호가 1234-5678-9012이면 AWS 계정 ID는123456789012).

```
https://your_aws_account_id.signin.aws.amazon.com/console/
```

방금 생성한 IAM 사용자 이름과 암호를 입력합니다. 로그인하면 네비게이션 바에 "your\_user\_name @ your\_aws\_account\_id"가 표시됩니다.

로그인 페이지 URL에 AWS 계정 ID를 보이지 않게 하려면 계정 별칭을 생성하세요. IAM 대시보드에서 Customize를 선택하고 Account Alias를 입력합니다. 회사 이름을 입력해도 됩니다. 자세한 내용은 IAM 사용 설명서의 [AWS 계정 ID 및 별칭](#)을 참조하세요.

계정 별칭 생성 후에는 다음 URL에서 로그인하세요.

```
https://your_account_alias.signin.aws.amazon.com/console/
```

사용자 계정의 IAM 사용자 로그인 링크를 확인하려면 IAM 콘솔을 열고 대시보드의 IAM 사용자 로그인 링크에서 확인합니다.

IAM에 대한 자세한 내용은 [AWS Identity and Access Management 사용 설명서](#)를 참조하십시오.

## Docker 설치 및 구성

[Docker Engine 설치 사용 설명서](#)에서 원하는 운영 체제를 선택하고 지침에 따라 Docker를 설치 및 구성합니다.

AWS CLI 버전 2를 설치하고 구성합니다.

[AWS CLI 버전 2 설치, 업데이트 및 제거 사용 설명서](#)에서 원하는 운영 체제를 선택하고 AWSCLI 버전 2를 설치하고 구성합니다.

## 주제

- [VS Code에서 Amazon Elastic Container Registry 서비스 사용](#)

## VS Code에서 Amazon Elastic Container Registry 서비스 사용

Amazon Elastic Container Registry (Amazon ECR)는 VS Code AWS Explorer에서 액세스할 수 있으며, Amazon ECR 리포지토리로 프로그램 이미지를 푸시할 수 있습니다. 시작하기 전에 다음을 준비하세요.

1. 이미지 빌드에 필요한 정보가 포함된 Dockerfile을 생성합니다.
2. 해당 Dockerfile에서 이미지를 빌드하고 처리할 이미지에 태그를 지정합니다.
3. Amazon ECR 인스턴스 내부에 리포지토리를 생성합니다.
4. 리포지토리에 태그가 지정된 이미지를 푸시합니다.

## 섹션

- [필수 조건](#)
- [1. Dockerfile 생성](#)
- [2. Dockerfile에서 이미지 빌드](#)
- [3. 새 리포지토리 생성](#)
- [4. 이미지 푸시, 폴 및 삭제](#)

## 필수 조건

VS Code용 도구 키트의 Amazon ECR 기능을 사용하려면 먼저 이러한 다음 조건을 갖춰야 합니다.

### 1. Dockerfile 생성

Docker는 Dockerfile이라는 파일을 사용하여 원격 리포지토리에 푸시하고 저장할 수 있는 이미지를 정의합니다. 이미지를 ECR 리포지토리에 업로드하려면 먼저 Dockerfile을 생성한 다음 Dockerfile에서 이미지를 빌드하세요.

#### Dockerfile 생성

1. Toolkit for VS Code 탐색기를 사용하여 Dockerfile을 저장할 디렉토리로 이동합니다.
2. Dockerfile 이름으로 새 파일을 생성합니다.

#### Note

VS Code에서 파일 형식 또는 파일 확장자를 선택하라는 메시지가 표시될 수 있습니다. 이 경우에는 일반 텍스트를 선택합니다. Vs Code에는 'dockerfile' 확장자가 있습니다. 그러나 사용하지 않는 것이 좋습니다. 이 확장자가 특정 버전의 Docker 또는 기타 관련 애플리케이션과 충돌을 일으킬 수 있기 때문입니다.

#### VS Code를 사용하여 Dockerfile 편집

Dockerfile에 파일 확장자가 있는 경우 파일에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 해당 파일 확장자를 제거합니다.

Dockerfile에서 파일 확장자를 제거한 후 다음을 수행합니다.

1. VS Code에서 빈 Dockerfile을 엽니다.
2. 다음 예제의 내용을 Dockerfile에 복사합니다.

#### Example Dockerfile 이미지 템플릿

```
FROM ubuntu:18.04

# Install dependencies
RUN apt-get update && \
    apt-get -y install apache2
```

```
# Install apache and write hello world message
RUN echo 'Hello World!' > /var/www/html/index.html

# Configure apache
RUN echo '. /etc/apache2/envvars' > /root/run_apache.sh && \
  echo 'mkdir -p /var/run/apache2' >> /root/run_apache.sh && \
  echo 'mkdir -p /var/lock/apache2' >> /root/run_apache.sh && \
  echo '/usr/sbin/apache2 -D FOREGROUND' >> /root/run_apache.sh && \
  chmod 755 /root/run_apache.sh

EXPOSE 80

CMD /root/run_apache.sh
```

이것은 Ubuntu 18.04 이미지를 사용하는 Dockerfile입니다. RUN 명령은 패키지 캐시를 업데이트합니다. 웹 서버용 소프트웨어 패키지를 설치하고 'Hello World!'를 웹 서버의 문서 루트에 작성합니다. EXPOSE 명령은 컨테이너에 포트 80을 노출하고 CMD 명령은 웹 서버를 시작합니다.

### 3. Dockerfile을 저장합니다.

#### Important

Dockerfile 이름에 확장자가 첨부되어 있지 않은지 확인하세요. 확장자가 있는 Dockerfile은 특정 버전의 Docker 또는 기타 관련 애플리케이션과 충돌을 일으킬 수 있기 때문입니다.

## 2. Dockerfile에서 이미지 빌드

생성한 Dockerfile에는 프로그램의 이미지를 빌드하는 데 필요한 정보가 포함되어 있습니다. 해당 이미지를 Amazon ECR 인스턴스로 푸시하려면 먼저 이미지를 빌드해야 합니다.

### Dockerfile에서 이미지 빌드

1. Dockerfile이 포함된 디렉터리로 이동하려면 Docker CLI 또는 Docker 인스턴스와 통합된 CLI를 사용합니다.
2. Dockerfile에 정의된 이미지를 빌드하려면 Docker build 명령을 실행합니다.

```
docker build -t hello-world .
```

3. 이미지가 올바르게 생성되었는지 확인하려면 Docker images 명령을 실행합니다.

```
docker images --filter reference=hello-world
```

Example 출력 예:

REPOSITORY SIZE	TAG	IMAGE ID	CREATED
hello-world 241MB	latest	e9ffedc8c286	4 minutes ago

- 4.

**Note**

이 단계는 이미지를 생성하거나 푸시하는 데 필요하지 않지만 프로그램 이미지가 실행되면 어떻게 작동하는지 확인할 수 있습니다.

Docker run 명령으로 새로 빌드된 이미지를 실행할 수 있습니다.

```
docker run -t -i -p 80:80 hello-world
```

위 예제에서 지정한 -p 옵션으로 컨테이너의 노출된 포트 80을 호스트 시스템의 포트 80에 매핑할 수 있습니다. Docker가 로컬에서 실행되고 있다면, 웹 브라우저에서 <http://localhost:80>로 이동하세요. 프로그램이 제대로 실행되면 “Hello, World!” 문구가 표시됩니다.

Docker run 명령에 대한 자세한 내용은 Docker 웹 사이트에서 [Docker run reference](#)를 참조하세요.

### 3. 새 리포지토리 생성

이미지를 Amazon ECR 인스턴스에 업로드하려면 이미지를 저장할 수 있는 새 리포지토리를 생성합니다.

Amazon ECR 리포지토리를 생성합니다.

1. VS Code Activity Bar에서 AWS도구 키트 아이콘을 선택합니다.
2. AWS Explorer 메뉴를 확장합니다.
3. AWS 계정에 연결된 기본 AWS 리전을 찾습니다. 리전을 선택하면 Toolkit for VS Code가 제공하는 서비스 목록이 표시됩니다.
4. ECR+ 옵션을 선택하여 새 리포지토리 생성 프로세스를 시작하세요.
5. 프롬프트의 메시지를 따라 프로세스를 완료합니다.
6. 프로세스가 완료되면 AWS Explorer 메뉴의 ECR 섹션에서 새 리포지토리에 액세스할 수 있습니다.

#### 4. 이미지 푸시, 풀 및 삭제

Dockerfile에서 이미지를 빌드하고 리포지토리를 생성한 후에는 Amazon ECR 리포지토리로 이미지를 푸시할 수 있습니다. 또한 Docker 및 AWS CLI와 함께 AWS Explorer를 사용하여 다음 작업을 수행할 수 있습니다.

- 리포지토리에서 이미지를 가져옵니다.
- 리포지토리에 저장된 이미지를 삭제합니다.
- 리포지토리를 삭제합니다.

#### 기본 레지스트리에 대해 Docker 인증

Amazon ECR과 Docker 인스턴스 간에 데이터를 교환하려면 인증이 필요합니다. 레지스트리에 대해 Docker를 인증하려면 다음을 수행합니다.

1. AWS CLI 인스턴스에 연결된 명령줄 운영 체제를 실행하세요.
2. get-login-password 메서드를 사용하여 프라이빗 ECR 레지스트리에 대해 인증합니다.

```
aws ecr get-login-password --region region | docker login --username AWS --password-stdin AWS_account_id.dkr.ecr.region.amazonaws.com
```



**⚠ Important**

이전 명령에서 **region**과 **AWS\_account\_id** 모두 AWS 계정에 맞는 정보로 업데이트해야 합니다.

이미지를 리포지토리에 태그 지정 및 푸시

AWS 인스턴스로 Docker 인증 후 이미지를 리포지토리에 푸시합니다.

1. Docker images 명령을 사용하면 로컬에 저장된 이미지를 보고 태그 지정할 이미지를 식별할 수 있습니다.

```
docker images
```

Example 출력 예:

REPOSITORY SIZE	TAG	IMAGE ID	CREATED
hello-world 241MB	latest	e9ffedc8c286	4 minutes ago

2. Docker tag 명령을 사용하여 이미지에 태그를 지정합니다.

```
docker tag hello-world:latest AWS_account_id.dkr.ecr.region.amazonaws.com/hello-world:latest
```

3. Docker push 명령을 사용하여 태그가 지정된 이미지를 리포지토리에 푸시합니다.

```
docker push AWS_account_id.dkr.ecr.region.amazonaws.com/hello-world:latest
```

Example 출력 예:

```
The push refers to a repository [AWS_account_id.dkr.ecr.region.amazonaws.com/hello-world] (len: 1)
e9ae3c220b23: Pushed
```

```
a6785352b25c: Pushed
0998bf8fb9e9: Pushed
0a85502c06c9: Pushed
latest: digest:
sha256:215d7e4121b30157d8839e81c4e0912606fca105775bb0636b95aed25f52c89b size: 6774
```

태그가 지정된 이미지를 리포지토리에 업로드되면 AWS Explorer 메뉴에 해당 이미지가 표시됩니다.

### Amazon ECR에서 이미지 가져오기

- Docker tag 명령의 로컬 인스턴스로 이미지를 가져올 수 있습니다.

```
docker pull AWS_account_id.dkr.ecr.region.amazonaws.com/hello-world:latest
```

Example 출력 예:

```
The push refers to a repository [AWS_account_id.dkr.ecr.region.amazonaws.com/hello-world] (len: 1)
e9ae3c220b23: Pushed
a6785352b25c: Pushed
0998bf8fb9e9: Pushed
0a85502c06c9: Pushed
latest: digest:
sha256:215d7e4121b30157d8839e81c4e0912606fca105775bb0636b95aed25f52c89b size: 6774
```

### Amazon ECR 리포지토리에서 이미지 삭제

VS Code에서 이미지를 삭제하는 방법은 두 가지입니다. 첫 번째 방법은 AWS Explorer를 사용하는 것입니다.

1. AWS 탐색기에서 ECR 메뉴를 확장합니다.
2. 삭제할 이미지의 리포지토리를 확장합니다.
3. 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 클릭하여 삭제하고 싶은 이미지와 연결된 이미지 태그를 선택하세요.
4. 해당 태그와 연결된 이미지를 모두 삭제하려면 Delete Tag...를 선택합니다.

## AWS CLI로 이미지 삭제

- AWS `ecr batch-delete-image` 명령을 사용하여 리포지토리에서 이미지를 삭제할 수도 있습니다.

```
AWS ecr batch-delete-image \  
  --repository-name hello-world \  
  --image-ids imageTag=latest
```

Example 출력 예:

```
{  
  "failures": [],  
  "imageIds": [  
    {  
      "imageTag": "latest",  
      "imageDigest":  
"sha256:215d7e4121b30157d8839e81c4e0912606fca105775bb0636b95aed25f52c89b"  
    }  
  ]  
}
```

## Amazon ECR 인스턴스에서 리포지토리 삭제

VS Code에서 리포지토리를 삭제하는 방법은 두 가지입니다. 첫 번째 방법은 AWS 탐색기를 사용하는 것입니다.

1. AWS Explorer에서 ECR 메뉴를 확장합니다.
2. 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 삭제할 리포지토리를 선택합니다.
3. 선택한 리포지토리에서 Delete Repository... 옵션을 선택합니다.

## AWS CLI에서 Amazon ECR 리포지토리 삭제

- AWS `ecr delete-repository` 명령을 사용하여 리포지토리를 삭제할 수 있습니다.

**Note**

기본적으로, 이미지가 들어 있는 리포지토리는 삭제할 수 없습니다. 하지만 `--force` 플래그로 삭제할 수 있습니다.

```
AWS ecr delete-repository \  
--repository-name hello-world \  
--force
```

Example 출력 예:

```
{  
  "failures": [],  
  "imageIds": [  
    {  
      "imageTag": "latest",  
      "imageDigest":  
"sha256:215d7e4121b30157d8839e81c4e0912606fca105775bb0636b95aed25f52c89b"  
    }  
  ]  
}
```

## Amazon Elastic Container Service 작업

AWS Toolkit for Visual Studio Code은 [Amazon Elastic Container Service\(Amazon ECS\)](#)를 일부 지원합니다. VS Code용 도구 키트는 작업 정의 생성과 같은 Amazon ECS 관련 작업을 지원합니다.

주제

- [Amazon ECS 작업 정의 파일에 IntelliSense 사용](#)
- [Amazon Elastic 컨테이너 서비스 담당 임원 AWS Toolkit for Visual Studio Code](#)

## Amazon ECS 작업 정의 파일에 IntelliSense 사용

Amazon Elastic Container Service (Amazon ECS)를 사용하면 Amazon Elastic Container Service 개발자 설명서에 설명된 [작업 정의 생성](#) 방법으로 작업 정의를 생성할 수 있습니다. AWS Toolkit for Visual Studio Code를 설치하면 Amazon ECS 작업 정의 파일에 대한 IntelliSense 기능도 설치됩니다.

### 필수 조건

- 시스템이 [VS Code용 도구 키트 설치](#)에 있는 조건에 부합하는지 확인하세요.

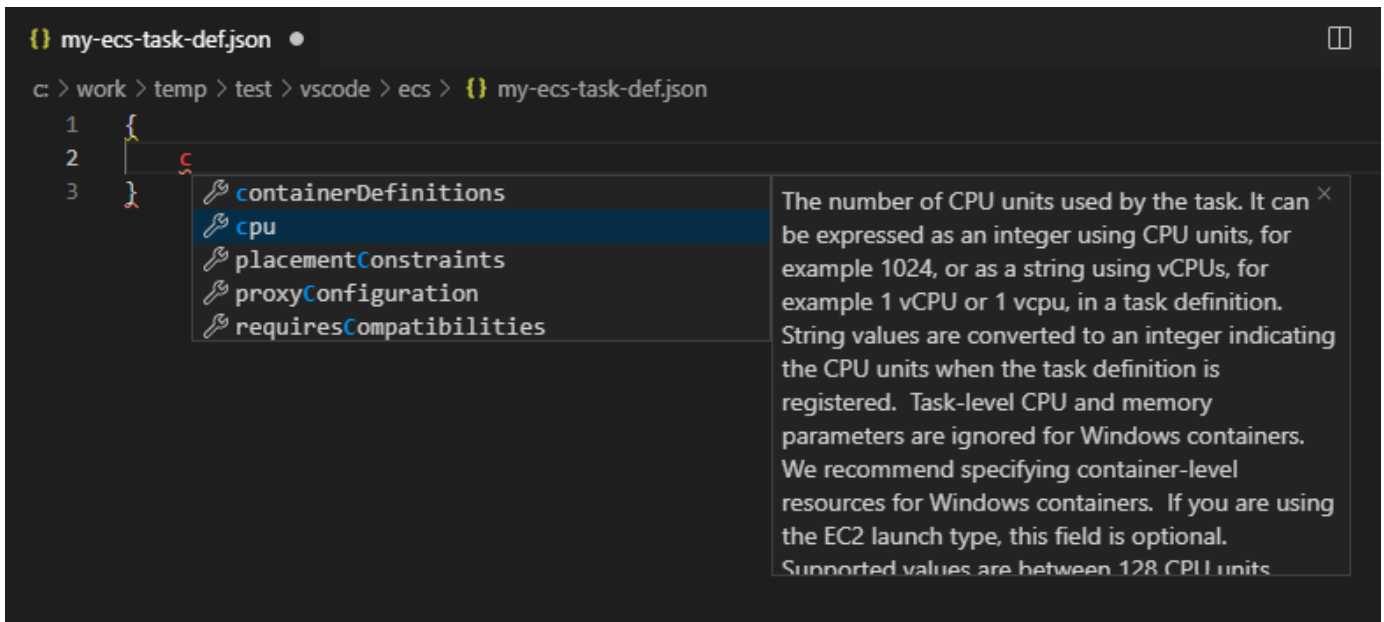
### Amazon ECS 작업 정의 파일에서 IntelliSense 사용

다음 예시는 Amazon ECS 작업 정의 파일에서 IntelliSense를 활용하는 방법을 보여 줍니다.

1. Amazon ECS 작업 정의에 대한 JSON 파일을 생성합니다. 파일 이름은 끝에 `ecs-task-def.json`이 있어야 합니다. 파일 이름 앞에 문자를 추가해도 됩니다.

이름이 `my-ecs-task-def.json`인 파일을 생성합니다.

2. VS Code 편집기에서 파일을 열고 종괄호를 입력합니다.
3. 정의에 `cpu`를 추가할 것처럼 문자 `"c"`를 입력합니다. 다음과 비슷하게 열리는 IntelliSense 대화 상자를 살펴보세요.



## Amazon Elastic 컨테이너 서비스 담당 임원 AWS Toolkit for Visual Studio Code

Amazon ECS Exec 기능을 사용하여 Amazon Elastic Container Service (Amazon ECS) 컨테이너에서 단일 명령을 실행할 수 있습니다. AWS Toolkit for Visual Studio Code

### Important

Amazon ECS Exec을 활성화 및 비활성화하면 계정의 리소스 상태가 변경됩니다. AWS 서비스 중지 및 재시작을 해도 바뀝니다. Amazon ECS Exec이 활성화된 상태에서 리소스 상태를 변경하면 예상치 못한 결과가 발생할 수 있습니다. Amazon ECS Exec에 대한 자세한 내용은 개발자 안내서의 [Using Amazon ECS Exec for Debugging](#)(Amazon ECS Exec로 디버깅하기)을 참조하세요.

## Amazon ECS Exec 필수 조건

Amazon ECS Exec 기능을 사용하기 위한 필수 조건을 충족해야 합니다.

### Amazon ECS 사전 조건

작업이 Amazon EC2에서 호스팅되는지 아니면 호스팅되는지에 따라 AWS Fargate (Fargate) Amazon ECS Exec의 버전 요구 사항은 다릅니다.

- Amazon EC2를 사용하는 경우 2021년 1월 20일 이후에 출시된 에이전트 버전 1.50.2 이상의 Amazon ECS optimized AMI를 사용하세요. 추가 정보는 개발자 안내서의 [Amazon ECS optimized AMIs](#)에서 확인할 수 있습니다.
- 를 사용하는 AWS Fargate 경우 플랫폼 버전 1.4.0 이상을 사용해야 합니다. Fargate 요구 사항에 대한 자세한 정보는 개발자 안내서의 [AWS Fargate platform versions](#)에서 확인하세요.

### AWS 계정 구성 및 IAM 권한

Amazon ECS Exec 기능을 사용하려면 계정에 연결된 기존 Amazon ECS 클러스터가 있어야 합니다. AWS Amazon ECS Exec은 Systems Manager를 통해 클러스터의 컨테이너에 연결되며 특정 Task IAM 역할 권한이 있어야 SSM 서비스와 통신할 수 있습니다.

Amazon ECS Exec과 관련된 IAM 역할 및 정책 정보는 개발자 안내서의 [IAM permissions required for ECS Exec](#)(ECS Exec에 필요한 IAM 권한)에서 확인할 수 있습니다.

## Amazon ECS Exec 작업

AWS 탐색기에서 VS Code용 도구 키트에서 직접 Amazon ECS Exec을 활성화하거나 비활성화할 수 있습니다. Amazon ECS Exec을 사용한다면 Amazon ECS 메뉴에서 컨테이너를 선택하고 해당 컨테이너에 대한 명령어를 실행합니다.

### Amazon ECS Exec 사용

1. AWS 탐색기에서 Amazon ECS 메뉴를 찾아 확장합니다.
2. 수정할 서비스가 포함된 클러스터를 확장합니다.
3. 서비스의 컨텍스트 메뉴를 열고(마우스 우클릭) Enable Command Execution(명령 실행)을 선택합니다.

#### Important

새로운 서비스가 설치되며 몇 분 정도 걸릴 수 있습니다. 자세한 내용은 이 섹션의 시작 부분에 나오는 참고를 참조하세요.

### Amazon ECS Exec 끄기

1. AWS 탐색기에서 Amazon ECS 메뉴를 찾아 확장합니다.
2. 원하는 서비스가 포함된 클러스터를 확장합니다.
3. 서비스의 컨텍스트 메뉴를 열고(마우스 우클릭) Disable Command Execution(명령 실행 취소)을 선택합니다.

#### Important

새로운 서비스가 설치되며 몇 분 정도 걸릴 수 있습니다. 자세한 내용은 이 섹션의 시작 부분에 나오는 참고를 참조하세요.

### 컨테이너에 대한 명령어 실행

AWS 탐색기를 사용하여 컨테이너에 대해 명령을 실행하려면 Amazon ECS Exec를 활성화해야 합니다. 활성화되지 않은 경우 이 섹션의 Enabling ECS Exec(ECS Exec 활성화) 방법을 참조하세요.

1. AWS 탐색기에서 Amazon ECS 메뉴를 찾아 확장합니다.

2. 원하는 서비스가 포함된 클러스터를 확장합니다.
3. 서비스를 확장하여 연결된 컨테이너를 나열합니다.
4. 컨테이너의 컨텍스트 메뉴를 열고(마우스 우클릭) Run Command in Container(컨테이너에서 명령 실행)를 선택합니다.
5. 실행 중인 작업 목록이 포함된 프롬프트가 나타나면 원하는 Task ARN을 선택합니다.

#### Note

해당 서비스에 Task 한 개만 실행 중이라면 Task가 자동으로 선택되며 이 단계는 생략됩니다.

6. 프롬프트가 표시되면 실행할 명령어를 입력하고 Enter 키를 눌러 계속 진행합니다.

## Amazon EventBridge 작업

AWS Toolkit for Visual Studio Code (VS Code)는 [Amazon EventBridge](#)를 지원합니다. Toolkit for VS Code를 사용하면 스키마를 비롯한 EventBridge의 특정 부분을 작업할 수 있습니다.

### 주제

- [Amazon EventBridge 스키마로 작업하기](#)

## Amazon EventBridge 스키마로 작업하기

AWS Toolkit for Visual Studio Code (VS Code) 를 사용하여 [Amazon EventBridge 스키마에서](#) 다양한 작업을 수행할 수 있습니다.

### 필수 조건

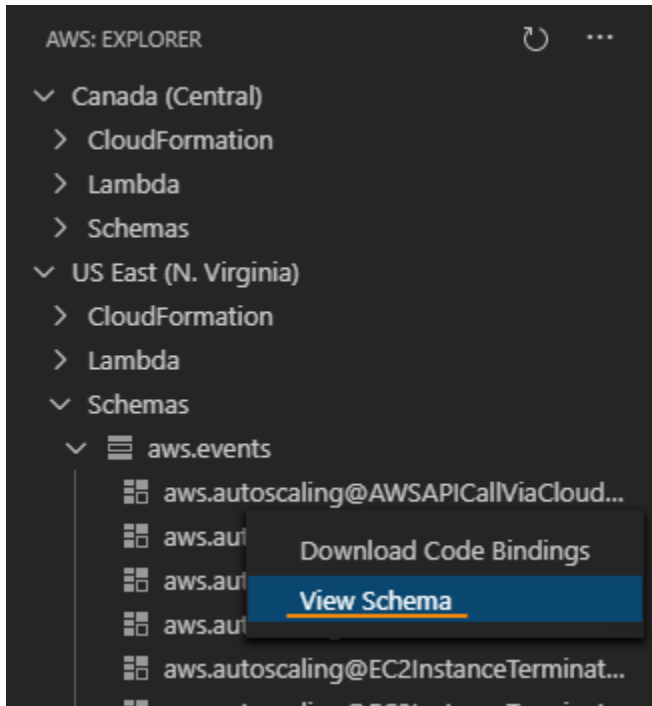
- 시스템이 [Toolkit for VS Code 설치](#)에 나온 조건에 맞는지 확인하세요.
- 사용하려는 EventBridge 스키마는 AWS 계정에서 사용할 수 있어야 합니다. 사용할 수 없다면 스키마를 생성하거나 업로드하세요. [Amazon EventBridge 사용 설명서의 Amazon EventBridge 스키마를 참조하십시오.](#)

### 사용 가능한 스키마 보기

1. AWS Explorer에서 스키마를 확장합니다.



2. 보려는 스키마가 포함된 레지스트리 이름을 확장합니다. 예를 들어, AWS 제공하는 대부분의 스키마는 aws.events 레지스트리에 있습니다.
3. 스키마의 컨텍스트 메뉴를 연 다음 View Schema(스키마 보기)를 클릭하면 편집기에서 스키마를 볼 수 있습니다.

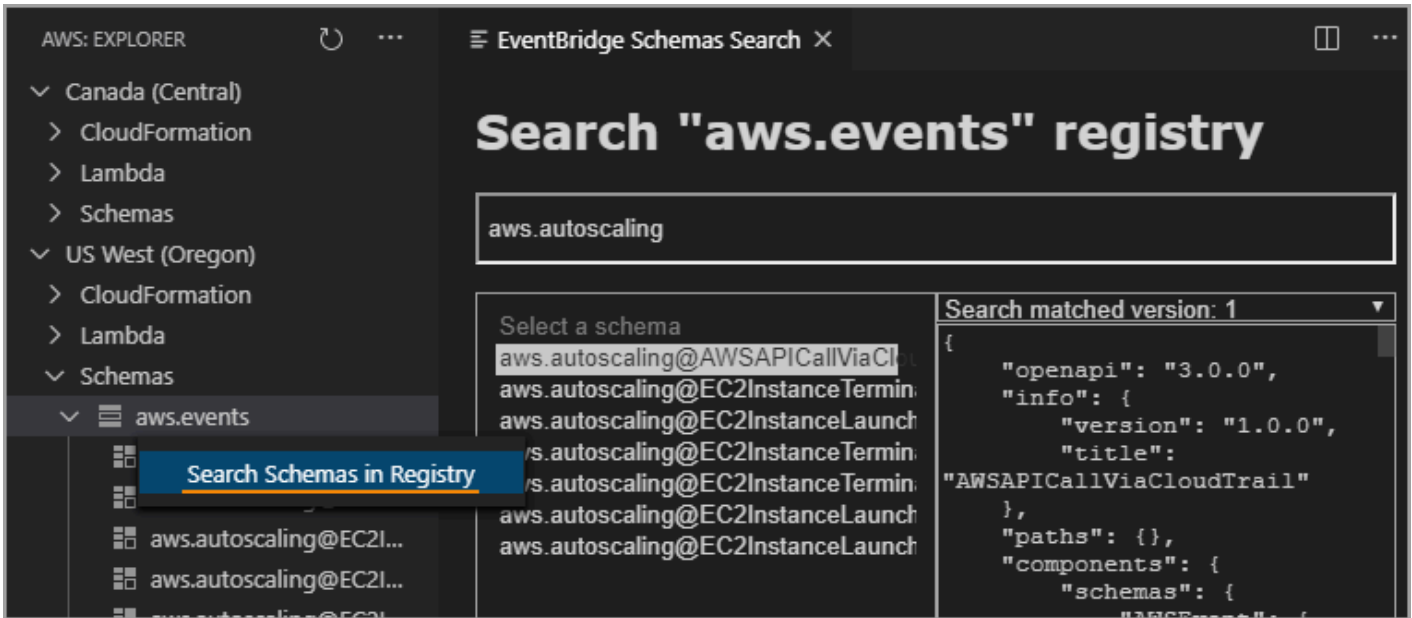


## 사용 가능한 스키마 찾기

AWS Explorer에서 다음 중 하나 이상을 수행합니다.

- 찾고 싶은 스키마 이름을 입력하세요. AWS Explorer에서 일치되는 스키마 제목이 강조 표시됩니다. (강조 표시된 제목을 보려면 레지스트리를 확장하세요.)
- Schemas 컨텍스트 메뉴를 열고 Search Schemas(스키마 찾기)를 클릭합니다. 또는 Schemas를 확장하고 찾고 싶은 스키마가 포함된 레지스트리 컨텍스트 메뉴를 연 다음 Search Schemas in Registry(레지스트리에서 스키마 찾기)를 선택합니다. EventBridge 스키마 검색 대화 상자에서 찾으려는 스키마의 제목을 입력하기 시작합니다. 대화 상자에 일치하는 스키마 제목이 표시됩니다.

스키마의 제목을 선택하면 대화 상자에 스키마가 표시됩니다.



## 사용 가능한 스키마 코드 생성

1. AWS Explorer에서 스키마를 확장합니다.
2. 코드 생성할 스키마가 포함된 레지스트리 이름을 확장합니다.
3. 스키마 이름을 마우스 우클릭한 다음 Download code bindings를 선택합니다.
4. 결과 마법사 페이지에서 다음을 선택합니다.
  - 스키마 버전
  - 코드 바인딩 언어
  - 로컬 개발 시스템에 생성된 코드를 저장할 작업 영역 폴더

## AWS IAM 액세스 분석기

의 [IAM 액세스 분석기를 사용하여 AWS CloudFormation 템플릿, Terraform 계획 및 JSON 정책 문서에서 작성된 IAM 정책에 대해 IAM \(AWS ID 및 액세스 관리\) 액세스 분석기 정책 검사를 실행할 수 있습니다.](#) AWS Toolkit for Visual Studio Code

IAM 액세스 분석기 정책 검사에는 정책 검증 및 사용자 지정 정책 검사가 포함됩니다. 정책 검증은 사용 설명서에 있는 IAM [JSON 정책 언어 문법 및 IAM 주제의 AWS보안 모범 사례에 자세히 설명된 표준에 따라 IAM 정책을](#) 검증하는 데 도움이 됩니다. AWS Identity and Access Management 정책 검증 결과에는 보안 경고, 오류, 일반 경고 및 정책 제안이 포함됩니다.

보안 표준에 따라 새 액세스에 대한 사용자 지정 정책 검사를 실행할 수도 있습니다. 새 액세스에 대한 각 사용자 지정 정책 검사에는 요금이 부과됩니다. 요금에 대한 자세한 내용은 [AWS IAM Access Analyzer 요금](#) 사이트를 참조하십시오. IAM Access Analyzer 정책 검사에 [대한 자세한 내용은 사용 설명서의 정책 검증 검사](#) 항목을 참조하십시오. AWS Identity and Access Management

다음 항목에서는 IAM 액세스 분석기 정책 체크인을 사용하는 방법을 설명합니다. AWS Toolkit for Visual Studio Code

주제

- [AWS IAM 액세스 분석기 사용](#)

## AWS IAM 액세스 분석기 사용

다음 섹션에서는 에서 IAM 정책 검증 및 사용자 지정 정책 검사를 수행하는 방법을 설명합니다. AWS Toolkit for Visual Studio Code 자세한 내용은 AWS Identity and Access Management 사용 설명서의 [IAM 액세스 분석기 정책 검증 및 IAM 액세스 분석기 사용자 지정 정책 검사](#) 항목을 참조하십시오.

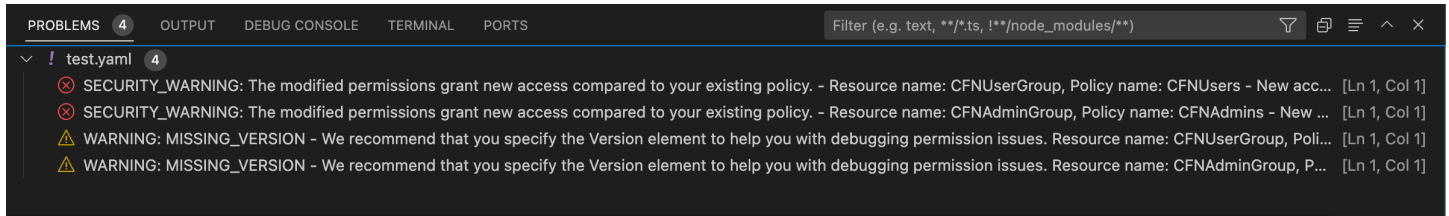
### 필수 조건

툴킷에서 IAM 액세스 분석기 정책 검사를 사용하려면 먼저 다음 사전 요구 사항을 충족해야 합니다.

- Python 버전 3.6 이상을 설치합니다.
- Python CLI [도구에 AWS CloudFormation](#) 필요하고 [IAM 정책 검사 창에 지정된 Terraform용 IAM 정책 검사기 또는 IAM 정책 검사기](#)를 설치합니다.
- 역할 자격 증명을 구성하십시오. AWS

### IAM Access Analyzer 정책 확인

를 사용하여 AWS CloudFormation 템플릿, Terraform 계획 및 JSON 정책 문서에 대한 정책 검사를 수행할 수 있습니다. AWS Toolkit for Visual Studio Code 확인 결과는 VS 코드 문제 패널에서 확인할 수 있습니다. 다음 이미지는 VS 코드 문제 패널을 보여줍니다.



IAM 액세스 분석기는 세 가지 유형의 검사를 제공합니다.

- 검증 정책
- CheckAccessNotGranted
- CheckNoNewAccess

다음 섹션에서는 각 유형의 검사를 실행하는 방법을 설명합니다.

### Note

모든 유형의 검사를 실행하기 전에 AWS 역할 자격 증명을 구성하십시오. 지원되는 파일에는 AWS CloudFormation 템플릿, Terraform 플랜, JSON 정책 문서 등의 문서 유형이 포함됩니다.

## 검증 정책 실행

정책 검증이라고도 하는 Validate Policy 검사는 IAM 정책 문법 및 AWS 모범 사례에 따라 정책을 검증합니다. 자세한 내용은 사용 설명서에 있는 [IAM JSON 정책 언어의 문법](#) 및 [IAM 주제의 AWS 보안 모범 사례](#)를 참조하십시오. AWS Identity and Access Management

1. VS Code에서 VS Code 편집기에서 AWS IAM 정책이 포함된 지원 파일을 엽니다.
2. IAM Access Analyzer 정책 검사를 열려면 **CTRL+Shift+P** VS Code 명령 팔레트를 열고 **IAM Policy Checks** 검색한 다음 클릭하여 VS Code 편집기에서 IAM 정책 검사 창을 엽니다.
3. IAM 정책 검사 창의 드롭다운 메뉴에서 문서 유형을 선택합니다.
4. 정책 검증 섹션에서 정책 검증 실행 버튼을 선택하여 정책 검증 검사를 실행합니다.
5. VS Code의 문제 패널에서 정책 검사 결과를 검토하십시오.
6. 정책을 업데이트하고 이 절차를 반복하여 정책 검사 결과에 더 이상 보안 경고나 오류가 표시되지 않을 때까지 Validate Policy 검사를 다시 실행하십시오.

## 실행 CheckAccessNotGranted

CheckAccessNotGranted 정책에서 특정 IAM 작업을 허용하지 않는지 확인하기 위한 사용자 지정 정책 검사입니다.

### Note

각 사용자 지정 정책 확인에는 요금이 부과됩니다. 사용자 지정 정책 확인 요금에 대한 자세한 내용은 [AWS IAM Access Analyzer](#) 요금 가이드를 참조하십시오.

1. VS Code에서 VS Code 편집기에서 AWS IAM 정책이 포함된 지원 파일을 엽니다.
2. IAM Access Analyzer 정책 검사를 열려면 를 눌러 **CRTL+Shift+P** VS Code 명령 팔레트를 열고 **IAM Policy Checks** 검색한 다음 클릭하여 VS Code 편집기에서 IAM 정책 검사 창을 엽니다.
3. IAM 정책 검사 창의 드롭다운 메뉴에서 문서 유형을 선택합니다.
4. 사용자 지정 정책 검사 섹션에서 을 선택합니다 CheckAccessNotGranted.
5. 텍스트 입력 필드에 심표로 구분된 작업 목록을 입력합니다. 또는 작업 목록을 가리키는 파일 경로를 제공할 수 있습니다.

#### Note

파일 경로는 일반적으로 관리자 또는 보안 팀이 제공하며, 시스템 파일 경로 또는 Amazon S3 버킷 URI일 수 있습니다. Amazon S3 버킷 URI를 사용하려면 현재 역할에 Amazon S3 버킷에 대한 액세스 권한이 있어야 합니다.

6. 사용자 지정 정책 검사 실행 버튼을 선택합니다.
7. VS Code의 문제 패널에서 정책 점검 결과를 검토하십시오. 사용자 지정 정책 검사는 PASS OR FAIL 결과를 반환합니다.
8. 정책을 업데이트하고 이 절차를 반복하여 결과가 PASS 반환될 때까지 CheckAccessNotGranted 검사를 다시 실행하십시오.

### 실행 중 CheckNoNewAccess

CheckNoNewAccess 정책이 참조 정책과 비교하여 새 액세스를 허용하는지 확인하기 위한 사용자 지정 정책 검사입니다.

#### Note

각 사용자 지정 정책 확인에는 요금이 부과됩니다. 사용자 지정 정책 확인 요금에 대한 자세한 내용은 [AWS IAM Access Analyzer 요금 가이드](#)를 참조하십시오.

1. VS Code에서 VS Code 편집기에서 AWS IAM 정책이 포함된 지원 파일을 엽니다.
2. IAM Access Analyzer 정책 검사를 열려면 를 눌러 **CRTL+Shift+P** VS Code 명령 팔레트를 열고 **IAM Policy Checks** 검색한 다음 클릭하여 VS Code 편집기에서 IAM 정책 검사 창을 엽니다.
3. IAM 정책 검사 창의 드롭다운 메뉴에서 문서 유형을 선택합니다.
4. 사용자 지정 정책 검사 섹션에서 을 선택합니다 CheckNoNewAccess.

5. 참조 JSON 정책 문서를 입력합니다. 또는 JSON 정책 문서를 참조하는 파일 경로를 제공할 수도 있습니다.

#### Note

파일 경로는 일반적으로 관리자 또는 보안 팀이 제공하며, 시스템 파일 경로 또는 Amazon S3 버킷 URI일 수 있습니다. Amazon S3 버킷 URI를 사용하려면 현재 역할에 Amazon S3 버킷에 대한 액세스 권한이 있어야 합니다.

6. 참조 문서 유형과 일치하는 참조 정책 유형을 선택합니다.
7. 사용자 지정 정책 검사 실행 버튼을 선택합니다.
8. VS Code의 문제 패널에서 정책 점검 결과를 검토하십시오. 사용자 지정 정책 검사는 PASS OR FAIL 결과를 반환합니다.
9. 정책을 업데이트하고 이 절차를 반복하여 결과가 PASS 반환될 때까지 CheckNoNewAccess 검사를 다시 실행하십시오.

## AWS Toolkit for Visual Studio Code에서 AWS IoT로 작업

AWS Toolkit for Visual Studio Code에서 AWS IoT를 사용하면 AWS IoT 서비스와 상호 작용하면서 VS Code의 작업 흐름 방해가 최소화될 수 있습니다. 이 설명서는 AWS Toolkit for Visual Studio Code에서 제공하는 AWS IoT 서비스 기능을 사용하는 방법을 설명합니다. AWS IoT 서비스에 대한 자세한 내용은 개발자 안내서 [What is AWS IoT?](#) 를 참조하십시오.

### AWS IoT 필수 조건

Toolkit for VS Code에서 AWS IoT를 사용하려면 AWS 계정과 VS Code가 가이드의 조건에 맞는지 확인하십시오.

- AWS IoT 서비스에 대한 AWS 계정 조건과 AWS 사용자 권한에 대한 자세한 내용은 AWS IoT Core 시작하기 개발자 설명서를 참조하십시오.
- Toolkit for VS Code 요구 사항은 [VS Code용 도구 키트 설정](#) 사용 설명서를 참조하십시오.

### AWS IoT 사물

AWS IoT는 장치와 AWS 클라우드 서비스와 리소스를 연결합니다. 사물이라는 객체를 사용하여 장치를 AWS IoT에 연결할 수 있습니다. 사물이란 특정 장치 또는 논리적 엔터티를 의미합니다. 사물은 물

리적 장치 또는 센서일 수 있습니다(예: 전구 또는 벽면 스위치). AWS IoT 사물에 대한 자세한 내용은 [Managing devices with AWS IoT](#)(로 기기 관리)를 참조하세요.

## AWS IoT 사물 관리

VS Code용 도구 키트는 AWS IoT 사물을 효율적으로 관리할 수 있는 몇 가지 기능을 제공합니다. VS Code용 도구 키트를 사용하여 AWS IoT 사물을 관리할 수 있는 방법은 다음과 같습니다.

- [Create a thing](#)
- [Attach a certificate to a thing](#)
- [Detach a certificate from a thing](#)
- [Delete a thing](#)

### 사물 생성

1. AWS Explorer에서 IoT 서비스 제목을 확장하고 Things(사물) 컨텍스트 선택 (마우스 오른쪽 버튼 클릭) 합니다.
2. 대화 상자를 열려면 컨텍스트 메뉴에서 Create Thing(사물 생성)을 선택합니다.
3. 메시지에 따라 Thing Name 필드에 IoT 사물 이름을 입력합니다.
4. 이 단계가 완료되면 사용자가 지정한 사물 아이콘 이름이 Thing 섹션에 표시됩니다.

### 사물에 인증서 첨부

1. AWS Explorer에서 IoT 서비스 섹션을 확장합니다.
2. Things 섹션에서 인증서를 첨부할 사물을 찾습니다.
3. Thing에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Attach Certificate(인증서 첨부)를 선택하면 인증서 목록이 포함된 입력 선택기가 열립니다.
4. 목록에서 사물에 첨부할 인증서에 맞는 certificate ID를 선택합니다.
5. 선택하면 AWS Explorer에서 사물의 항목에 첨부한 인증서에 액세스할 수 있습니다.

### 사물에서 인증서 분리

1. AWS Explorer에서 IoT 서비스 섹션을 확장합니다.

2. Things 섹션에서 분리할 인증서를 사물에서 찾으세요.
3. 사물 컨텍스트 (마우스 오른쪽 버튼으로 클릭)를 선택하고 Detach Certificate(인증서 분리)를 클릭합니다.
4. 분리된 인증서는 해당 사물 AWS 탐색기에서 더 이상 표시되지 않지만 Certificates에서 계속 액세스할 수 있습니다.

## 사물 삭제

1. AWS Explorer에서 IoT 서비스 섹션을 확장합니다.
2. Things 섹션에서 삭제할 사물을 찾습니다.
3. 사물 컨텍스트(마우스 오른쪽 버튼 클릭)를 선택하고 컨텍스트 메뉴에서 Delete Thing(사물 삭제)를 클릭하여 삭제합니다.
4. 삭제된 사물은 Things에서 더 이상 사용할 수 없습니다.

### Note

주의: 인증서가 첨부되지 않은 항목만 삭제할 수 있습니다.

## AWS IoT 인증서

인증서는 AWS IoT 서비스와 장치 간 보안 연결을 만드는 일반적인 방법입니다. X.509 인증서는 X.509 퍼블릭 키 인프라 표준을 사용하여 퍼블릭 키를 인증서에 포함된 ID와 연결하는 디지털 인증서입니다. AWS IoT 인증에 대한 자세한 내용은 개발자 설명서의 [Authentication \(IoT\)](#)을 참조하세요.

## 인증서 관리

VS Code 도구 키트는 AWS Explorer에서 바로 AWS IoT 인증서를 관리하는 다양한 방법을 제공합니다.

- [Create a certificate](#)
- [Change a certificate status](#)
- [Attach a policy to a certificate](#)
- [Delete a certificate](#)



## AWS IoT 인증서 생성

X.509 인증서를 이용하여 AWS IoT 인스턴스에 연결할 수 있습니다.

1. AWS Explorer에서 IoT 서비스 섹션을 확장하고 (마우스 오른쪽 버튼 클릭으로) 인증서를 엽니다.
2. 대화 상자를 열려면 컨텍스트 메뉴에서 Create Certificate(인증서 생성)를 선택합니다.
3. RSA 키 페어와 X.509 인증서를 저장하려면, 로컬 파일 시스템에서 디렉토리를 선택합니다.

### Note

- 기본 파일 이름에 인증서 ID가 접두사로 포함됩니다.
- X.509 인증서만 AWS IoT 서비스를 통해 AWS 계정과 함께 저장됩니다.
- RSA 키 페어는 한 번만 발급할 수 있습니다. 메시지가 표시되면 파일 시스템 내 안전한 위치에 저장하세요.
- 인증서나 키 페어를 파일 시스템에 저장할 수 없다면, AWS 도구 키트로 AWS 계정에서 인증서를 삭제합니다.

## 인증서 상태 수정

개별 인증서의 상태는 AWS 탐색기의 인증서 ID 옆에 표시되며 active(활성), inactive(비활성) 또는 revoked(취소됨)으로 설정할 수 있습니다.

### Note

- 인증서를 사용하여 장치를 AWS IoT 서비스에 연결하려면 인증서가 active(활성) 상태여야 합니다.
- 이전에 비활성화했거나 기본적으로 비활성 상태인 inactive(비활성) 인증서를 활성화할 수 있습니다.
- revoked(취소됨) 상태인 인증서는 활성화할 수 없습니다.

1. AWS Explorer에서 IoT 서비스 섹션을 확장합니다.
2. Certificates(인증서) 섹션에서 수정할 인증서를 찾습니다.

3. 인증서에 사용할 수 있는 상태 변경 옵션을 표시하는 인증서의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 엽니다.
  - 인증서의 상태가 inactive(비활성)인 경우 activate(활성)를 선택하여 activate(활성)으로 상태를 변경합니다.
  - 인증서의 상태가 active(활성)인 경우 deactivate(비활성)를 선택하여 inactivate(비활성)로 상태를 변경합니다.
  - 인증서의 상태가 active(활성) 또는 inactive(비활성)인 경우 revoke(해지)를 선택하여 revoked(취소됨)로 상태를 변경합니다.

#### Note

이러한 상태 변경 작업은 Things(사물) 섹션에 있는 사물에 첨부된 인증서를 선택하면 수행할 수 있습니다.

### 인증서에 IoT 정책 첨부

1. AWS Explorer에서 IoT 서비스 섹션을 확장합니다.
2. Certificates(인증서) 섹션에서 수정할 인증서를 찾습니다.
3. 인증서의 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) Attach Policy(정책 첨부)를 선택하여 사용 가능한 정책 목록이 포함된 입력 선택기를 엽니다.
4. 인증서에 첨부할 정책을 선택합니다.
5. 이 단계를 완료하면 선택한 정책은 인증서의 하위 메뉴 항목으로 추가됩니다.

### 인증서에서 IoT 정책 분리

1. AWS Explorer에서 IoT 서비스 섹션을 확장합니다.
2. Certificates(인증서) 섹션에서 수정할 인증서를 찾습니다.
3. 인증서를 확장하고 분리할 정책을 찾습니다.
4. 정책의 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) Detach(분리)를 선택합니다.
5. 이 단계를 완료하면 인증서에서 더 이상 해당 정책에 액세스할 수 없으며, Policy(정책) 섹션에서 액세스할 수 있습니다.

## 인증서 삭제

1. AWS Explorer에서 IoT 서비스 이름을 확장합니다.
2. Certificates(인증서) 섹션에서 삭제할 인증서를 찾습니다.
3. 인증서의 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) Delete Certificate(인증서 삭제)를 선택합니다.

### Note

사물에 첨부되었거나 활성 상태인 인증서는 삭제할 수 없습니다. 정책에 첨부된 인증서는 삭제할 수 있습니다.

## AWS IoT 정책

AWS IoT Core 정책은 정책 문구가 하나 이상 포함된 JSON 문서를 통해 정의됩니다. 정책은 AWS IoT, AWS, 및 장치가 서로 상호 작용하는 방법을 정의합니다. 정책 문서를 만드는 방법에 대한 자세한 내용은 개발자 안내서의 [IoT Policies](#)(IoT 정책)을 참조하세요.

### Note

이름이 있는 정책은 롤백할 수 있도록 버전이 지정됩니다. AWS Explorer에서 IoT 정책은 IoT 서비스의 Policies(정책) 섹션에 나열됩니다. 정책을 확장하면 정책 버전을 볼 수 있습니다. 기본 버전은 별표(\*)로 표시됩니다.

## 정책 관리

Toolkit for VS Code의 여러 방법을 통해 AWS IoT 서비스 정책을 관리할 수 있습니다. 다음은 VS Code AWS Explorer에서 바로 정책을 관리하거나 수정하는 방법입니다.

- [Create a policy](#)
- [Upload a new policy version](#)
- [Edit a policy version](#)
- [Change the policy version default](#)
- [Change the policy version default](#)

## AWS IoT 정책 생성

### Note

AWS 탐색기에서 새 정책을 생성할 수 있으며 정책을 정의하는 JSON 문서가 파일 시스템에 있어야 합니다.

1. AWS 탐색기에서 IoT 서비스 섹션을 확장합니다.
2. Policies(정책) 하위 섹션의 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) Create Policy from Document(문서에서 정책 만들기)를 선택하여 Policy Name(정책 이름) 입력 필드를 엽니다.
3. 이름을 입력하고 파일 시스템에서 JSON 문서를 선택하라는 지시에 따라 대화 상자를 여세요.
4. 정책 정의가 포함된 JSON 파일을 선택합니다. 이 작업이 완료되면 AWS Explorer에서 정책에 액세스할 수 있습니다.

## 새 AWS IoT 정책 버전 업로드

JSON 문서를 정책에 업로드하면 새 버전의 정책을 만들 수 있습니다.

### Note

AWS Explorer를 사용하여 새 버전을 만들려면 파일 시스템에 새 JSON 문서가 있어야 합니다.

1. AWS Explorer에서 IoT 서비스 섹션을 확장합니다.
2. Policies(정책) 섹션을 확장하여 AWS IoT 정책을 봅니다.
3. 업데이트할 정책의 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) Create new version from Document(문서에서 새 버전 생성)를 선택합니다.
4. 대화 상자가 열리면 정책 정의에 대한 업데이트가 포함된 JSON 파일을 선택합니다.
5. AWS Explorer의 정책에서 새 버전에 액세스할 수 있습니다.

## AWS IoT 정책 버전 편집

VS Code를 사용하여 정책 문서를 열고 편집할 수 있습니다. 문서 편집이 끝나면 문서를 파일 시스템에 저장합니다. 그런 다음 AWS 탐색기에서 문서를 AWS IoT 서비스에 업로드합니다.

1. AWS Explorer에서 IoT 서비스 섹션을 확장합니다.
2. Policies(정책) 섹션을 확장하여 업데이트하려는 정책을 찾습니다. Create Policy from Document(문서에서 정책 생성)을 클릭하여 Policy Name(정책 이름) 입력 필드를 엽니다.
3. 업데이트할 정책을 확장한 후 편집할 정책 버전의 컨텍스트 메뉴를 엽니다(마우스 오른쪽 버튼 클릭).
4. 컨텍스트 메뉴에서 View(보기)를 선택하여 VS Code의 정책 버전을 엽니다.
5. 정책 문서가 열렸다면 변경 사항을 편집하고 저장합니다.

#### Note

이때 정책에 적용된 변경 사항은 로컬 파일 시스템에만 저장됩니다. 버전을 업데이트하고 AWS Explorer에서 추적하려면 [Upload a new policy version](#)에 나오는 단계를 반복합니다.

### 새 정책 기본 버전 선택

1. AWS Explorer에서 IoT 서비스 섹션을 확장합니다.
2. Policies(정책) 하위 섹션을 확장하고 업데이트할 정책을 찾습니다.
3. 업데이트할 정책을 확장한 후 설정할 정책 버전의 컨텍스트 메뉴를 연 다음(마우스 오른쪽 버튼 클릭) Set as Default(기본값으로 설정)를 선택합니다.
4. 이 작업을 완료하면 선택한 새 기본 버전 옆에 별표가 표시됩니다.

### 정책 삭제

#### Note

정책 또는 정책 버전을 삭제하기 전에 다음 조건이 충족되어야 합니다.

- 정책이 인증서에 첨부된 경우 삭제할 수 없습니다.
- 기본 버전이 아닌 정책은 삭제할 수 없습니다.
- 새 기본 버전을 선택하거나 정책 전체를 삭제하면 정책의 기본 버전만 삭제할 수 있습니다.
- 정책 전체를 삭제하기 전에 기본이 아닌 버전 모두 삭제해야 합니다.

1. AWS Explorer에서 IoT 서비스 섹션을 확장합니다.
2. Policies(정책) 하위 섹션을 펼치고 업데이트할 정책을 찾습니다.
3. 업데이트할 정책을 확장한 후 삭제할 정책 버전의 컨텍스트 메뉴를 연 다음(마우스 오른쪽 버튼 클릭) Delete(삭제)를 선택합니다.
4. 삭제된 버전은 탐색기에서 볼 수 없습니다.
5. 정책 기본 버전만 남아 있는 경우 상위 정책의 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) Delete(삭제)를 선택합니다.

## AWS Lambda 함수 작업

AWS Toolkit for Visual Studio Code는 [AWS Lambda](#) 함수를 지원합니다. VS Code용 도구 키트를 사용하여 [서버리스 애플리케이션](#) 일부인 Lambda 함수 코드를 작성할 수 있습니다. 또한 Lambda 함수를 로컬 또는 AWS에서 호출할 수 있습니다.

Lambda는 사용자 지정 코드로 생성된 이벤트 또는 Amazon Simple Storage Service(Amazon S3), Amazon DynamoDB, Amazon Kinesis, Amazon Simple Notification Service(Amazon SNS), Amazon Cognito 등 다양한 AWS 서비스에서 코드를 실행하는 완전관리형 컴퓨팅 서비스입니다.

### 주제

- [원격 Lambda 함수와 상호 작용](#)

## 원격 Lambda 함수와 상호 작용

Toolkit for VS Code를 사용하면 이 주제의 뒷부분에 설명된 다양한 방법으로 [AWS Lambda](#) 함수와 상호 작용할 수 있습니다.

Lambda에 대한 자세한 내용은 [AWS Lambda 개발자 안내서](#)를 참조하세요.

### Note

AWS Management Console 또는 다른 방법을 사용하여 Lambda 함수를 생성했다면 도구 키트에서 해당 함수를 호출할 수 있습니다. AWS Lambda에 배포할 수 있는 새 함수(VS Code 사용)를 생성하려면 먼저 [서버리스 애플리케이션](#)을 생성하세요.

### 주제

- [필수 조건](#)
- [Lambda 함수 호출](#)
- [Lambda 함수 삭제](#)
- [Lambda 함수 가져오기](#)
- [Lambda 함수 업로드](#)

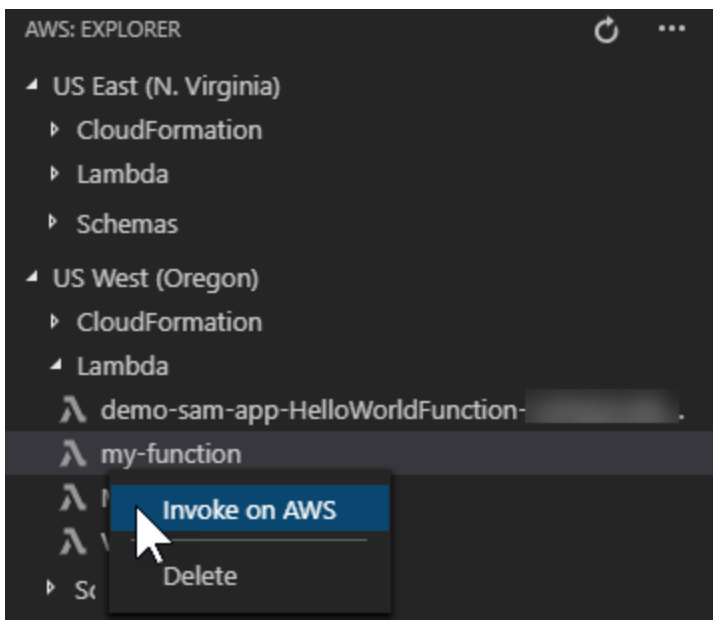
## 필수 조건

- 시스템이 [Toolkit for VS Code 설치](#)에 명시된 필수 조건에 맞는지 확인합니다.
- [인증 및 액세스](#)에서 구성한 자격 증명에 AWS Lambda 서비스에 대한 적절한 읽기/쓰기 액세스 권한이 포함되어 있는지 확인하세요. AWS Explorer의 Lambda 아래에서 'Lambda 리소스를 로드하는 동안 오류가 발생했습니다(Error loading Lambda resources)' 같은 메시지가 표시되면 해당 자격 증명에 연결된 권한을 확인하세요. 권한을 변경한 경우 VS Code AWS Explorer에 적용되는 데 몇 분 정도 걸립니다.

## Lambda 함수 호출

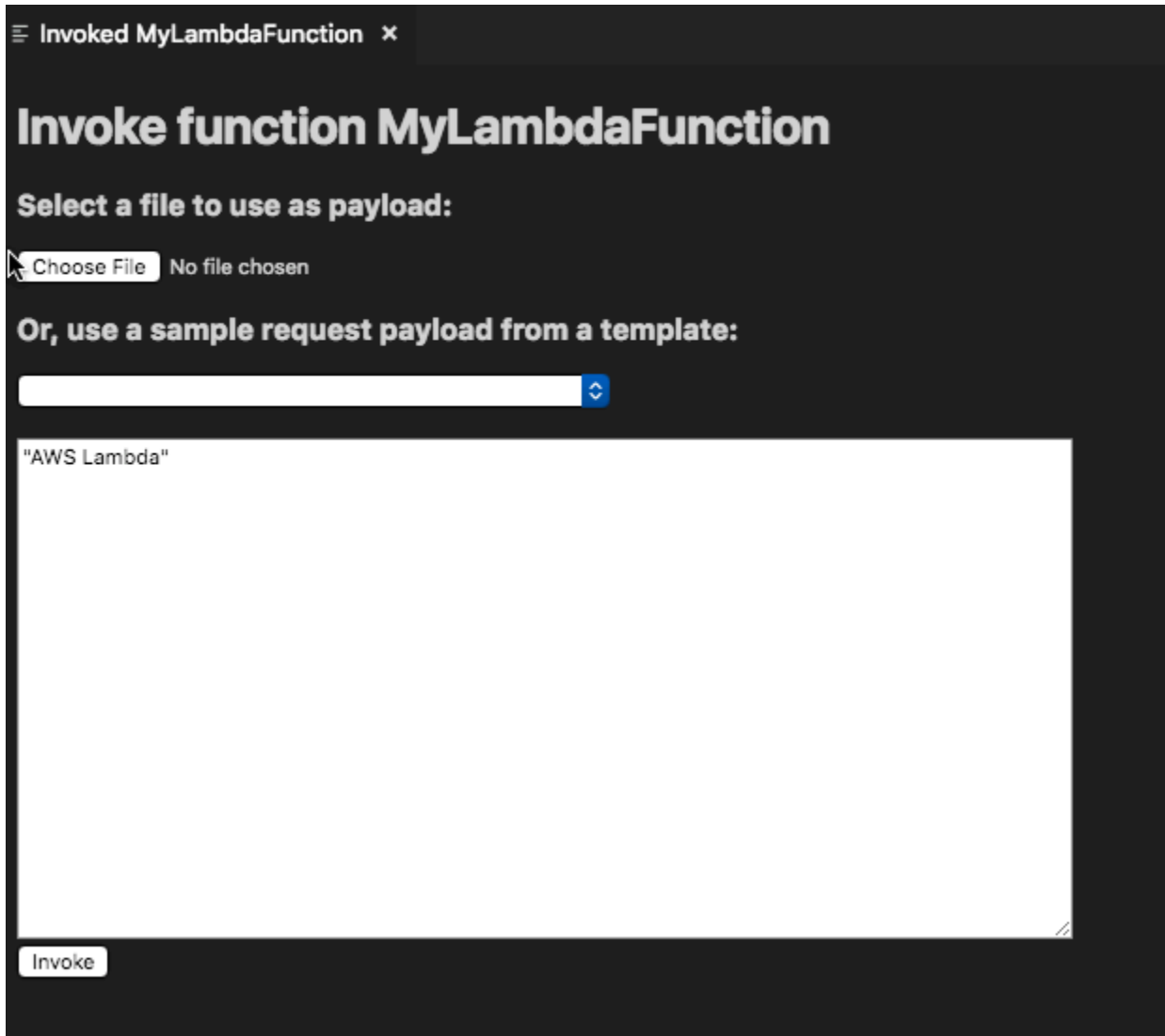
VS Code용 도구 키트를 사용하여 AWS에서 Lambda 함수를 호출할 수 있습니다.

1. AWS Explorer에서 호출하려는 Lambda 함수의 이름을 선택한 다음 컨텍스트 메뉴를 엽니다.



2. AWS에서 호출을 선택합니다.

- 호출 창이 열리면 Lambda 함수에 필요한 것을 입력합니다. 예를 들어 텍스트 상자에 표시된 것처럼 Lambda 함수에 문자열을 입력할 수도 있습니다.



VS Code를 사용하는 프로젝트와 마찬가지로 Lambda 함수의 출력을 확인할 수 있습니다.

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL AWS Lambda
Loading response...
Invocation result for arn:aws:lambda:us-west-2:282147211633:function:MyLambdaFunction
Logs:
START RequestId: 04c5f09a-508f-4251-bc6d-663023153d7c Version: $LATEST
Input: AWS LambdaEND RequestId: 04c5f09a-508f-4251-bc6d-663023153d7c
REPORT RequestId: 04c5f09a-508f-4251-bc6d-663023153d7c Duration: 253.98 ms Billed Duration: 300 ms Memory Size: 128 MB
Max Memory Used: 86 MB

Payload:
"Hello, AWS Lambda!"
  
```



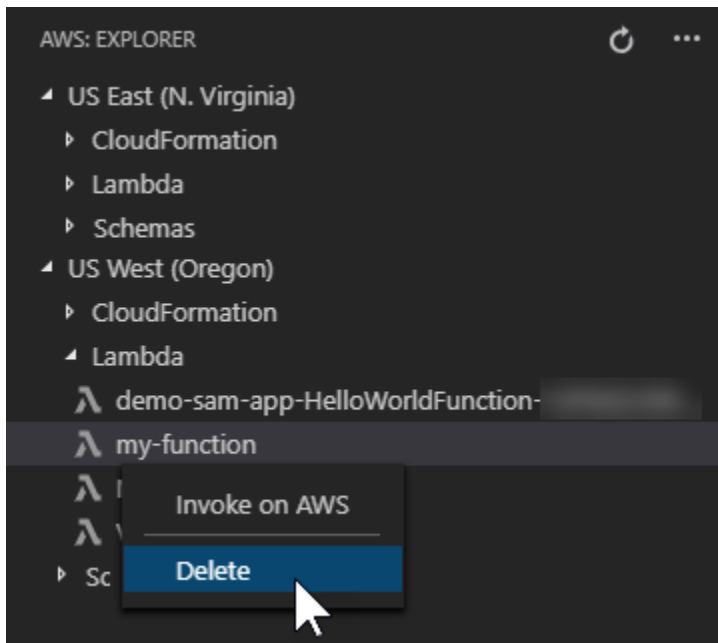
## Lambda 함수 삭제

동일한 컨텍스트 메뉴를 사용하여 Lambda 함수를 삭제할 수도 있습니다.

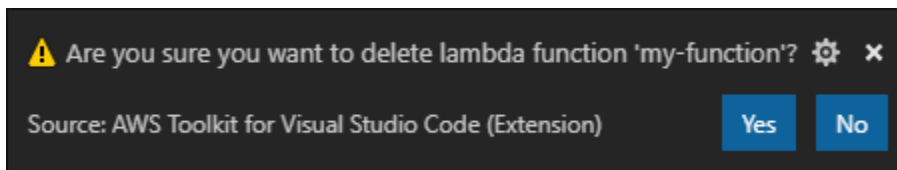
### ⚠ Warning

이 절차를 사용하여 [AWS CloudFormation](#)에 연결된 Lambda 함수를 삭제하지 마세요(예: 이 가이드의 앞부분에서 [서버리스 애플리케이션을 만들때](#) 생성된 Lambda 함수). 이러한 함수는 AWS CloudFormation 스택을 통해 삭제해야 합니다.

1. AWS Explorer에서 삭제할 Lambda 함수의 이름을 선택한 다음 컨텍스트 메뉴를 엽니다.



2. Delete를 선택합니다.
3. 보고 있는 메시지에서 Yes를 선택하여 삭제합니다.



함수가 삭제되면 더 이상 AWS Explorer에 보이지 않습니다.

## Lambda 함수 가져오기

원격 Lambda 함수 코드를 VS Code 작업 공간으로 가져와 수정 및 디버깅을 할 수 있습니다.

### Note

도구 키트로 지원되는 Node.js 및 Python 런타임으로 Lambda 함수 가져오기만 가능합니다.

1. AWS Explorer에서 가져올 Lambda 함수 이름을 선택한 다음 컨텍스트 메뉴를 엽니다.
2. Import...(가져오기)를 선택합니다.
3. Lambda 코드를 가져올 폴더를 선택합니다. 현재 작업 폴더 밖에 있는 폴더가 현재 작업 폴더에 추가됩니다.
4. 다운로드 후 도구 키트로 작업 폴더에 코드를 추가하고 Lambda 핸들러 코드가 들어 있는 파일을 엽니다. 또한 도구 키트는 VS Code 실행 패널에 보이는 시작 구성을 생성하기 때문에 AWS Serverless Application Model을 사용하여 Lambda 함수를 로컬에서 실행 및 디버깅할 수 있습니다. AWS SAM 사용에 관한 자세한 내용은 [the section called “템플릿\(로컬\)에서 서버리스 애플리케이션 실행 및 디버깅”](#) 부분을 참조하세요.

## Lambda 함수 업로드

로컬 코드로 기존 Lambda 함수를 업데이트할 수 있습니다. 이 방법으로 코드를 업데이트하면 AWS SAM CLI를 사용하여 배포하지 않으며 AWS CloudFormation 스택이 생성되지 않습니다. 이 기능을 통해 Lambda가 지원하는 모든 런타임을 사용하여 Lambda 함수를 업로드할 수 있습니다.

### Warning

도구 키트로 코드 작동 여부를 확인할 수 없습니다. 프로덕션 Lambda 함수를 업데이트하기 전에 코드가 작동하는지 확인하세요.

1. AWS Explorer에서 가져오려는 Lambda 함수 이름을 선택한 다음 컨텍스트 메뉴를 엽니다.
2. Upload Lambda...(Lambda 업로드)를 선택합니다.
3. 세 가지 Lambda 함수 업로드 옵션 중에서 선택합니다. 옵션은 다음과 같습니다.

미리 만든 .zip 아카이브 업로드


- 메뉴에서 ZIP Archive를 선택합니다.

- 파일 시스템에서 .zip 파일을 선택하고 모달 대화상자를 사용하여 업로드를 확인합니다. 이렇게 .zip 파일을 업로드하고 설치하여 Lambda를 업데이트합니다.

디렉토리를 있는 그대로 업로드


- Quick Pick 메뉴에서 디렉토리를 선택합니다.
- 파일 시스템에서 디렉토리를 선택합니다.
- 디렉토리를 생성하라는 메시지가 표시되면 No를 선택한 다음 모달 대화 상자를 사용하여 업로드합니다. 디렉토리가 있는 그대로 업로드되고 설치되면 Lambda가 즉시 업데이트됩니다.

디렉터리 구축 및 업로드

 Note

AWS SAM CLI가 필요합니다.

- Quick Pick 메뉴에서 디렉토리를 선택합니다.
- 파일 시스템에서 디렉토리를 선택합니다.
- 디렉토리를 생성하라는 메시지가 표시되면 Yes를 선택한 다음, 모달 대화 상자를 사용하여 업로드합니다. AWS SAM CLI `aws sam build` 명령어로 디렉터리에 코드를 생성하고 배포하면 Lambda가 즉시 업데이트됩니다.

 Note

업로드 전에 일치하는 핸들러가 감지되지 않으면 도구 키트에 경고가 나타납니다. Lambda 함수에 연결된 핸들러 변경은 AWS Management Console 또는 AWS CLI를 통해 할 수 있습니다.

## Toolkit for VS Code의 Amazon Redshift

Amazon Redshift는 클라우드에서 완전히 관리되는 페타바이트급 데이터 웨어하우스 서비스입니다. Amazon Redshift 서비스에 대한 자세한 내용은 [Amazon Redshift](#) 사용 설명서 목차를 참조하세요.

다음 항목은 AWS Toolkit for Visual Studio Code에서 Amazon Redshift를 사용하는 방법을 설명합니다.

주제

- [Toolkit for VS Code에서 Amazon Redshift 작업](#)

## Toolkit for VS Code에서 Amazon Redshift 작업

다음 섹션에서는 AWS Toolkit for Visual Studio Code Amazon Redshift로 작업하는 방법을 설명합니다.

Amazon Redshift 서비스에 대한 자세한 내용은 [Amazon Redshift](#) 사용 설명서를 참조하세요.

### 시작하기

AWS Toolkit for Visual Studio Code에서 Amazon Redshift로 작업하기 전에 다음 사항을 충족해야 합니다.

1. 도구 키트에서 AWS 계정에 연결되었습니다. 도구 키트에서 AWS 계정에 연결하는 방법에 대한 자세한 내용은 사용 설명서의 [Connecting to AWS](#)(연결)를 참조하세요.
2. 프로비저닝된 데이터 웨어하우스 또는 서버리스 데이터 웨어하우스가 생성되었습니다.

Amazon Redshift 서버리스 또는 Amazon Redshift 프로비저닝 클러스터를 아직 생성하지 않은 경우, AWS 콘솔에서 샘플 데이터셋을 사용하여 데이터 웨어하우스를 생성하는 방법에 대한 설명입니다.

### 프로비저닝된 데이터 웨어하우스 생성

Amazon Redshift 프로비저닝 클러스터 데이터 웨어하우스를 생성하는 방법에 대한 자세한 내용은 Amazon Redshift 시작하기 사용 설명서의 [샘플 Amazon Redshift 클러스터 생성](#) 항목을 참조하세요.

1. 선호하는 인터넷 브라우저에서 <https://console.aws.amazon.com/redshift/>로 이동하여 AWS 관리 콘솔에 로그인하고 Amazon Redshift 콘솔을 엽니다.
2. Amazon Redshift 콘솔에서 프로비저닝된 클러스터 대시보드를 선택합니다.
3. 프로비저닝된 클러스터 대시보드에서 클러스터 생성 버튼을 선택하여 클러스터 생성 창을 엽니다.
4. 클러스터 구성 섹션의 필수 필드를 작성합니다.
5. 샘플 데이터 섹션에서 샘플 데이터 로드 상자를 선택하여 샘플 데이터셋을 **public** 스키마가 **Dev** 있는 기본 데이터베이스의 **Tickit**에 로드합니다.

6. 데이터베이스 구성 섹션에서 관리자 이름 및 관리자 암호를 입력합니다.
7. 클러스터 생성을 선택하여 프로비저닝된 데이터 웨어하우스를 생성합니다.

## 서버리스 데이터 웨어하우스 생성

Amazon Redshift 서버리스 데이터 웨어하우스를 생성하는 방법에 대한 자세한 내용은 Amazon Redshift 시작하기 사용 설명서의 [Amazon Redshift 서버리스로 데이터 웨어하우스 생성](#) 섹션을 참조하세요.

1. 선호하는 인터넷 브라우저에서 <https://console.aws.amazon.com/redshift/>로 이동하여 AWS 관리 콘솔에 로그인하고 Amazon Redshift 콘솔을 엽니다.
2. Amazon Redshift 콘솔에서 Amazon Redshift 서버리스 사용해 보기 버튼을 선택하여 Amazon Redshift 서버리스 시작 창을 엽니다.
3. 구성 섹션에서 기본 설정 사용 (방사형)을 선택합니다.
4. Amazon Redshift Serverless 시작하기 창 하단의 구성 저장을 선택하여 기본 작업 그룹, 네임스페이스, 자격 증명 및 암호 설정이 있는 서버리스 데이터 웨어하우스를 생성합니다.

## 도구 키트에서 데이터 웨어하우스에 연결

도구 키트에서 데이터베이스에 연결하는 방법 세 가지가 있습니다.

- 데이터베이스 사용자 이름 및 암호
- AWS Secrets Manager
- 임시 보안 인증

도구 키트에서 기존의 프로비저닝된 클러스터 또는 서버리스 데이터 웨어하우스에 있는 데이터베이스에 연결하려면 다음 단계를 완료하세요.

### Important

사용 설명서의 필수 조건 섹션에 있는 단계를 완료했지만 도구 키트에 탐색기에 데이터 웨어하우스가 보이지 않는 경우 탐색기에서 올바른 AWS 리전에서 작업하고 있는지 확인하세요.

데이터베이스 사용자 이름 및 암호 메서드를 사용하여 데이터 웨어하우스에 연결

1. 도구 키트 탐색기에서 데이터 웨어하우스가 있는 AWS 리전을 확장하세요.

2. Redshift를 확장하고 데이터 웨어하우스를 선택한 후 VS Code에서 연결 유형 선택 대화 상자를 엽니다.
3. 연결 유형 선택 대화 상자에서 데이터베이스 사용자 이름 및 암호를 선택하고 각 프롬프트에 필요한 정보를 제공합니다.
4. 도구 키트가 데이터 웨어하우스에 연결되어 절차가 완료되면 사용 가능한 데이터베이스, 테이블 및 스키마가 도구 키트 탐색기에 표시됩니다.

AWS Secrets Manager을 사용하여 데이터 웨어하우스에 연결

#### Note

이 절차를 완료하려면 AWS secrets manager 데이터베이스 암호가 필요합니다. 데이터베이스 암호를 설정하는 방법에 대한 지침은 AWS Secrets Manager 사용 설명서의 [AWS Secrets Manager 데이터베이스 암호 만들기를](#) 참조하세요.

1. 도구 키트 탐색기에서 데이터 웨어하우스가 있는 AWS 리전을 확장하세요.
2. Redshift를 확장하고 데이터 웨어하우스를 선택하여 VS Code에서 연결 유형 선택 대화 상자를 엽니다.
3. 연결 유형 선택 대화 상자에서 Secrets Manager를 선택하고 각 프롬프트에 필요한 정보를 입력합니다.
4. 도구 키트가 데이터 웨어하우스에 연결되고 절차가 완료되면 사용 가능한 데이터베이스, 테이블 및 스키마가 도구 키트 탐색기에 표시됩니다.

임시 자격 증명으로 데이터 웨어하우스에 연결

1. 도구 키트 탐색기에서 데이터 웨어하우스가 있는 AWS 리전을 확장합니다.
2. Redshift를 확장하고 데이터 웨어하우스를 선택하여 VS Code에서 연결 유형 선택 대화 상자를 엽니다.
3. 연결 유형 선택 대화 상자에서 임시 자격 증명을 선택하고 각 프롬프트에 필요한 정보를 입력합니다.
4. 도구 키트가 데이터 웨어하우스에 연결되고 절차가 완료되면 사용 가능한 데이터베이스, 테이블 및 스키마가 도구 키트 탐색기에 표시됩니다.

## 데이터 웨어하우스 연결 수정

데이터 웨어하우스 연결을 수정하여 연결할 데이터베이스를 변경할 수 있습니다.

1. 도구 키트 탐색기에서 데이터 웨어하우스가 있는 AWS 리전을 확장하세요.
2. Redshift를 확장하고 연결된 데이터 웨어하우스를 마우스 우클릭한 다음 연결 수정을 선택하고 연결하려는 데이터베이스의 이름을 입력합니다.
3. 도구 키트가 데이터 웨어하우스에 연결되고 절차가 완료되면 사용 가능한 데이터베이스, 테이블 및 스키마가 도구 키트 탐색기에 표시됩니다.

## 데이터 웨어하우스 연결 삭제

1. 도구 키트 탐색기에서 데이터 웨어하우스가 있는 AWS 리전을 확장합니다.
2. Redshift를 확장하고 삭제하려는 연결된 데이터 웨어하우스를 마우스 우클릭한 다음 연결 삭제를 선택합니다. 이렇게 하면 도구 키트 탐색기에서 사용 가능한 데이터베이스, 테이블 및 스키마가 삭제됩니다.
3. 데이터 웨어하우스에 다시 연결하려면 클릭하여 연결을 선택하고 각 프롬프트에 필요한 정보를 입력합니다. 기본적으로 재연결은 이전 인증 방법을 사용하여 데이터 웨어하우스에 연결합니다. 다른 방법을 사용하려면 인증 프롬프트가 나올 때까지 대화 상자에서 뒤쪽 화살표를 누릅니다.

## SQL 명령문 실행

다음은 AWS Toolkit for Visual Studio Code의 데이터베이스에 SQL 명령문을 생성 및 실행하는 방법에 대한 설명입니다.

### Note

각 단계를 완료하려면 우선 사용 설명서에 있는 도구 키트에서 데이터 웨어하우스에 연결 섹션을 완료하세요.

1. 도구 키트 탐색기에서 Redshift에서 쿼리하려는 데이터베이스가 포함된 데이터 웨어하우스를 확장합니다.
2. Create-Notebook을 선택하여 노트북에 저장할 파일 이름과 위치를 지정한 다음 OK를 선택하면 VS Code 편집기에서 노트북을 열 수 있습니다.
3. VS Code 편집기에서 노트북에 저장하려는 SQL 명령문을 입력합니다.

4. 모두 실행 버튼을 선택하여 입력한 SQL 명령문을 실행합니다.
5. SQL 명령문의 출력이 입력한 명령문 아래에 표시됩니다.

### 노트북에 마크다운 추가

1. VS Code 편집기의 노트북에서 마크다운 버튼을 선택하여 노트북에 마크다운 셀을 추가합니다.
2. 셀에 마크다운을 입력합니다.
3. 마크다운 셀은 마크다운 셀의 오른쪽 상단 가장자리에 있는 편집기 도구를 사용하여 편집할 수 있습니다.

### 노트북에 코드 추가

1. VS Code 편집기의 노트북에서 코드 버튼을 선택하여 노트북에 코드 셀을 추가합니다.
2. 셀에 코드를 입력합니다.
3. 코드 셀의 오른쪽 상단 가장자리에 있는 셀 편집기 도구에 있는 버튼을 선택하여 코드 셀 위 또는 아래에서 코드를 실행하세요.

## Amazon S3 작업

Amazon Simple Storage Service (Amazon S3)는 확장 가능한 데이터 스토리지 서비스입니다. AWS Toolkit for Visual Studio Code를 사용하면 VS Code에서 직접 Amazon S3 객체 및 리소스를 관리할 수 있습니다.

Amazon S3 서비스에 대한 자세한 내용은 [Amazon DynamoDB](#) 사용 설명서를 참조하세요.

다음 항목에서는 AWS Toolkit for Visual Studio Code에서 Amazon S3 객체 및 리소스 작업을 수행하는 방법을 설명합니다.

### 주제

- [Amazon S3 리소스 사용](#)
- [Amazon S3 객체 작업](#)

## Amazon S3 리소스 사용

AWS Toolkit for Visual Studio Code의 Amazon S3에서 버킷 및 기타 리소스를 보고, 관리하고, 수정할 수 있습니다.



다음 섹션에서는 AWS Toolkit for Visual Studio Code의 Amazon S3 리소스를 활용하는 방법을 설명합니다. AWS Toolkit for Visual Studio Code의 Amazon S3 객체 (예: 폴더 및 파일) 활용에 대한 자세한 내용은 이 사용 설명서의 [S3 객체 활용](#)을 참조하세요.

## Amazon S3 버킷 생성

1. 도구 키트 탐색기에서 S3 서비스의 컨텍스트 메뉴를 연 다음(마우스 우클릭) Create Bucket...(버킷 생성)을 선택합니다. 또는 Create Bucket 아이콘을 선택하여 Create Bucket 대화 상자를 열 수도 있습니다.
2. Bucket Name 필드에 유효한 버킷 이름을 입력합니다.

Enter 키를 눌러 버킷을 생성하고 대화 상자를 닫습니다. 이제 새 버킷이 도구 키트의 S3 서비스 아래에 표시됩니다.

### Note

Amazon S3에서 공개 액세스 URL로 버킷을 사용할 수 있기 때문에 선택한 버킷 이름이 전역적으로 고유해야 합니다. 다른 계정이 사용하려는 이름으로 버킷을 생성한 경우 다른 이름을 사용해야 합니다.

새 버킷을 만들 수 없다면 Output 탭에서 AWS Toolkit Logs를 확인하세요. 잘못된 버킷 이름을 사용하려고 하면 BucketAlreadyExists 오류가 발생합니다.

자세한 내용은 Amazon Simple Storage Service 사용 설명서의 [Bucket restrictions and limitations](#)을 참조하세요.

## Amazon S3 버킷에 폴더 추가

객체를 폴더로 그룹화하여 S3 버킷 콘텐츠를 정리할 수 있습니다. 폴더 안에 폴더를 만들 수도 있습니다.

1. 도구 키트 탐색기에서 S3 서비스를 확장하여 S3 리소스를 확인하세요.
2. Create Folder icon(폴더 아이콘 만들기)을 선택하면 Create Folder(폴더 만들기) 대화 상자가 나타납니다. 버킷 또는 폴더 컨텍스트 메뉴를 연 다음(마우스 우클릭) Create Folder(폴더 만들기)를 선택합니다.
3. Folder Name 필드에 값을 입력하고 Enter 키를 누르면 폴더가 생성되고 대화 상자가 닫힙니다. 새 폴더는 도구 키트 메뉴의 해당 S3 리소스 아래에 표시됩니다.

## Amazon S3 버킷 삭제

S3 버킷을 삭제하면 S3 버킷에 포함된 폴더와 객체도 삭제됩니다. S3 버킷을 삭제하려고 하면 삭제 의사를 묻는 메시지가 나타납니다.

1. 도구 키트 기본 메뉴에서 Amazon S3 서비스를 확장하여 S3 리소스 목록을 확인합니다.
2. 버킷 또는 폴더의 컨텍스트메뉴를 연 다음(마우스 우클릭) Delete S3 Bucket(S3 Bucket 삭제)을 선택합니다.
3. 메시지가 나타나면 텍스트 필드에 버킷 이름을 입력한 다음 Enter 키를 눌러 버킷을 삭제하고 확인 메시지를 받으세요.

### Note

버킷에 객체가 포함된 경우 삭제되기 전에 객체가 비워집니다. 한 번에 많은 리소스 또는 객체를 삭제하면 삭제 시간이 조금 걸릴 수 있습니다. 삭제되면 성공적으로 삭제되었다는 알림이 표시됩니다.

## Amazon S3 객체 작업

S3 리소스 버킷에 저장된 파일, 폴더 및 기타 데이터를 S3 객체라고 합니다.

다음 섹션에서 AWS Toolkit for Visual Studio Code에서 Amazon S3 작업을 수행하는 방법을 설명합니다. AWS Toolkit for Visual Studio Code에서 Amazon S3 리소스 (예: S3 버킷)로 작업하는 방법에 대한 자세한 내용은 사용 설명서의 [Working with S3 resources](#)(S3 리소스 작업)를 참조하세요.

### 객체 페이지 매기기

많은 Amazon S3 객체 및 폴더로 작업하는 경우 페이지 매김 기능을 통해 페이지에 표시할 항목 수를 지정할 수 있습니다.

1. VS Code Activity Bar로 이동하여 Extensions을 선택합니다.
2. AWS Toolkit 확장 프로그램에서 설정 아이콘을 선택한 다음 Extension Settings을 선택합니다.
3. Settings 페이지에서 AWS > S3: Max Items Per Page 설정으로 내려갑니다.
4. '추가 항목 로드'가 표시되기 전에, 기본값을 표시하려는 S3 항목 수로 변경합니다.

**Note**

유효한 값은 3과 1,000 사이의 숫자입니다. 이 설정은 한 번에 표시되는 객체 또는 폴더 수에만 적용됩니다. 생성한 모든 버킷이 한 번에 표시됩니다. 기본적으로 AWS 계정 별로 최대 100개의 버킷을 만들 수 있습니다.

5. Settings 페이지를 닫고 변경 사항을 확인하세요.

또한 Settings 페이지 오른쪽 상단의 Open Settings (JSON) 아이콘을 선택하여 JSON 형식 파일의 설정을 업데이트할 수 있습니다.

## Amazon S3에서 객체 업로드 및 다운로드

로컬에 저장된 파일을 Amazon S3 버킷에 업로드하거나 AWS Toolkit for Visual Studio Code에서 로컬 시스템으로 원격 Amazon S3 객체를 다운로드할 수 있습니다.

도구 키트를 사용하여 버킷에 파일 업로드

1. 도구 키트 탐색기에서 Amazon S3 서비스를 확장하면 S3 리소스 목록이 나타납니다.
2. 버킷 또는 폴더 옆에 있는 파일 업로드 아이콘을 선택하여 파일 업로드 대화 상자를 엽니다. 또는 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) 파일 업로드를 선택합니다.

**Note**

객체의 상위 폴더 또는 리소스에 파일을 업로드하려면 S3 객체의 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) 상위 폴더에 업로드를 선택합니다.

3. 시스템의 파일 관리자를 통해 파일을 선택한 다음 파일 업로드를 선택하면 대화 상자가 닫히고 파일이 업로드됩니다.

Command Palette로 파일 업로드

도구 키트 인터페이스 또는 Command Palette를 사용하여 버킷에 파일을 업로드할 수 있습니다.

1. VS Code에서 해당 파일의 탭을 클릭하여 업로드할 파일을 선택합니다.
2. Ctrl+Shift+P를 누르면 Command Palette가 나타납니다.
3. Command Palette 해당 문구 `upload file`를 입력하면 권장 명령이 나타납니다.

4. AWS: 파일 업로드를 선택하여 AWS: 파일 업로드 대화 상자를 엽니다.
5. 메시지가 표시되면 업로드할 파일을 선택한 다음 파일을 업로드할 버킷을 선택합니다.
6. 대화 상자를 닫으면 업로드가 시작됩니다. 업로드가 완료되면 객체 크기, 최근 수정 날짜, 경로가 포함된 메타데이터와 함께 도구 키트 메뉴에 객체가 표시됩니다.

## Amazon S3 객체 다운로드

1. 도구 키트 탐색기에서 S3 서비스를 확장합니다.
2. 버킷 또는 폴더에서 다운로드하려는 객체의 컨텍스트 (마우스 오른쪽 버튼 클릭) 메뉴를 엽니다. 그런 다음 Download As(다른 이름으로 다운로드)를 선택하여 다른 이름으로 다운로드 대화 상자를 엽니다. 또는 개체 옆에 있는 Download As(다른 이름으로 다운로드) 아이콘을 선택해도 됩니다.
3. 시스템의 파일 관리자를 사용하여 대상 폴더를 선택하고 파일 이름을 입력한 다음 다운로드를 선택하면 대화 상자가 사라지고 다운로드를 시작합니다.

## 원격 객체 편집

AWS Toolkit for Visual Studio Code를 사용하면 원격 Amazon S3 리소스에 저장된 Amazon S3 객체를 편집할 수 있습니다.

1. 도구 키트 탐색기를 확장하면 S3 서비스가 나타납니다.
2. 편집할 파일이 포함된 S3 리소스를 확장합니다.
3. 연필 아이콘 (파일 편집) 을 선택하여 파일을 편집합니다.
4. 읽기 전용 모드로 연 파일을 편집하려면 VS Code 편집기에서 파일을 확인한 다음 UI의 오른쪽 상단에 있는 연필 아이콘을 선택합니다.

### Note

- VS Code를 다시 시작하거나 종료하면 IDE와 S3 리소스 연결이 끊깁니다. 원격 S3 파일 편집 중에 연결이 끊기면 편집이 중단됩니다. VS Code를 다시 시작하여 편집 탭을 열어 편집을 계속할 수 있습니다.
- 파일 편집 버튼은 UI의 오른쪽 상단에 있습니다. 해당 버튼은 VS Code 편집기에서 읽기 전용 파일을 보고 있는 경우에만 표시됩니다.

- 텍스트가 아닌 파일은 읽기 전용 모드에서 열리지 않습니다. 파일은 편집 모드에서만 열립니다.
- 편집 전용 모드에서 읽기 전용 모드로 전환할 수는 없으며 반대로는 가능합니다.

## Amazon S3 객체의 경로 복사

다음은 AWS Toolkit for Visual Studio Code에서 Amazon S3 객체의 경로를 복사하는 방법을 설명합니다.

1. 도구 키트를 확장하면 S3 서비스가 나타납니다.
2. 경로 복사하려는 객체가 포함된 리소스 버킷을 확장합니다.
3. 경로를 복사하려는 객체의 컨텍스트 메뉴 (마우스 오른쪽 버튼 클릭)를 연 다음, 경로 복사를 선택하여 객체 경로를 로컬 클립보드에 복사합니다.

## 미리 서명된 Amazon S3 객체 URL 생성

미리 서명된 URL에 다운로드 시간 제한을 부여하여 비공개 Amazon S3 객체를 다른 사용자와 공유할 수 있습니다. 자세한 내용은 [Sharing an object with a presigned URL](#)(미리 서명된 URL로 객체 공유)을 참조하세요.

1. 도구 키트를 확장하면 S3 서비스가 나타납니다.
2. 버킷 또는 폴더에서 공유하려는 객체의 컨텍스트 (마우스 오른쪽 버튼 클릭) 메뉴를 엽니다. 그런 다음 Generate Presigned URL(미리 서명된 URL 만들기)을 선택하여 Command palette를 엽니다.
3. Command Palette에서 URL을 통해 객체에 액세스할 수 있는 시간(분)을 입력합니다. 그런 다음 Enter를 선택하고 대화 상자를 닫습니다.
4. 미리 서명된 URL이 생성되면 VS 코드 상태 표시줄에 로컬 클립보드에 복사된 객체의 미리 서명된 URL이 표시됩니다.

## Amazon S3 객체 삭제

버전 관리 대상이 아닌 버킷에 있는 객체를 영구적으로 삭제할 수 있습니다. 하지만 버전 관리를 사용하는 버킷에서 객체를 삭제해도 영구 삭제되지 않습니다. 대신 Amazon S3에서 버킷에 삭제 마커를 삽입하세요. 자세한 내용은 [Deleting object versions](#)(객체 버전 삭제)를 참조하세요.

1. 도구 키트 탐색기에서 S3 서비스를 확장하여 S3 리소스 목록을 확인합니다.

2. 삭제하려는 객체의 컨텍스트 메뉴 (마우스 오른쪽 버튼 클릭) 를 연 다음, Delete를 선택하면 확인 대화 상자가 나타납니다.
3. Delete. . .를 선택하면 삭제하려는 S3 객체가 삭제됩니다. 삭제되면 대화 상자를 닫습니다.

## 서버리스 애플리케이션 작업

AWS Toolkit for Visual Studio Code은 [AWS 서버리스 애플리케이션](#)을 지원합니다. 다음 항목에서는 AWS Toolkit for Visual Studio Code에서AWS Serverless Application Model (AWS SAM) 애플리케이션을 생성하고 사용하는 방법을 설명합니다.

### 주제

- [서버리스 애플리케이션 시작하기](#)
- [코드에서 Lambda 함수 실행 및 디버깅](#)
- [로컬 Amazon API Gateway 리소스 실행 및 디버깅](#)
- [서버리스 애플리케이션 디버깅을 위한 구성 옵션](#)
- [서버리스 애플리케이션 문제 해결](#)

## 서버리스 애플리케이션 시작하기

다음 섹션에서는 AWS Serverless Application Model (AWS SAM) 및 AWS CloudFormation 스택을 사용하여 AWS Toolkit for Visual Studio Code에서 AWS 서버리스 애플리케이션을 생성하는 방법을 설명합니다.

### 필수 조건

AWS 서버리스 애플리케이션을 생성하거나 사용하려면 먼저 다음 조건을 충족해야 합니다.

#### Note

다음 작업을 수행하려면 변경이 완료되기 전에 VS Code를 종료하거나 재시작해야 합니다.

- AWS SAM 명령줄 인터페이스(CLI)를 설치합니다. AWS SAM CLI 설치 방법에 대한 추가 정보 및 지침은 AWS Serverless Application Model사용 설명서의 [AWS SAMCLI 설치](#)를 참조하세요.
- AWS 구성 파일의 기본 AWS 리전을 확인합니다. 구성 파일에 대한 자세한 내용은 AWS Command Line Interface사용 설명서의 [구성 및 자격 증명 파일 설정](#)을 참조하세요.

- 언어 SDK를 설치하고 툴체인을 구성합니다. AWS Toolkit for Visual Studio Code에서 툴체인을 구성하는 방법에 대한 자세한 내용은 사용 설명서의 [툴체인 구성](#) 항목을 참조하세요.
- VS Code 마켓플레이스에서 [YAML 언어 지원 확장 프로그램](#)을 설치하세요. AWS SAM 템플릿 파일의 CodeLens 기능에 액세스하는 데 필요합니다. CodeLens에 대한 추가 정보는 VS Code 설명서의 [CodeLens](#) 섹션을 참조하세요.

## 서버리스 애플리케이션에 대한 IAM 권한

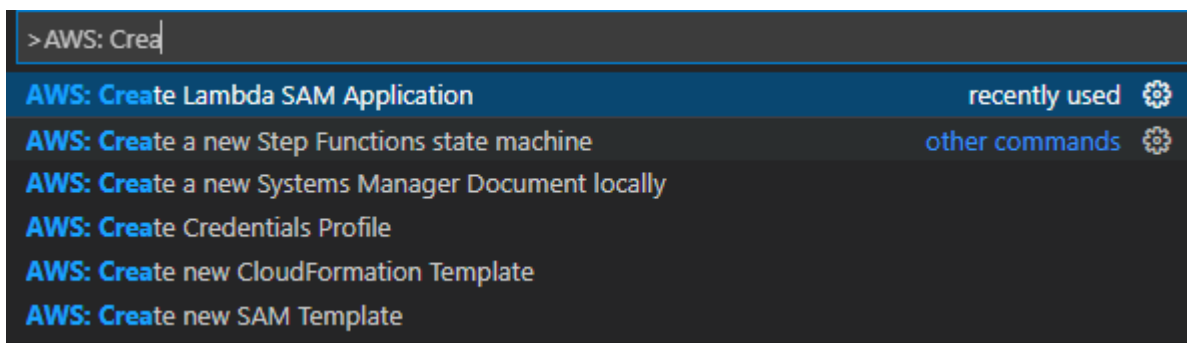
VS Code용 도구 키트에는 서버리스 애플리케이션을 배포 및 실행하는 데 필요한 AWS Identity and Access Management (IAM) 권한이 포함된 자격 증명 프로파일이 있어야 합니다. AWS CloudFormation, IAM, Lambda, Amazon API Gateway, Amazon Simple Storage Service (Amazon S3), Amazon Elastic Container Registry (Amazon ECR)에 대한 읽기/쓰기 액세스 권한이 있어야 합니다.

서버리스 애플리케이션을 배포하고 실행하는 데 필요한 인증을 설정하는 방법에 대한 추가 정보는 AWS Serverless Application Model 개발자 안내서의 [리소스 액세스 및 권한 관리](#)를 참조하세요. 자격 증명 설정 방법에 대한 자세한 내용은 사용 설명서의 [AWS IAM 자격 증명](#) 섹션을 참조하세요.

## 새 서버리스 애플리케이션 생성 (로컬)

AWS SAM을 사용하여 VS Code용 도구 키트로 서버리스 애플리케이션을 만드는 방법을 알 수 있습니다. 이 프로시저의 출력은 샘플 서버리스 애플리케이션이 포함된 개발 호스트의 로컬 디렉토리로, 이를 빌드, 로컬 테스트, 수정 및 배포를 AWS 클라우드에 할 수 있습니다.

1. Command Palette를 열려면 View, Command Palette를 선택한 다음 AWS를 입력합니다.
2. AWS Toolkit Create Lambda SAM Application을 선택합니다.



**Note**

AWS SAM CLI가 설치되어 있지 않으면 VS Code 편집기 오른쪽 아래 가장자리에 오류 메시지가 표시됩니다. 이 경우 모든 [가정과 필수 조건](#)을 충족하는지 확인하세요.

3. AWS SAM 애플리케이션의 런타임을 선택합니다.

**Note**

'(Image)'라는 표시가 있는 런타임 중 하나를 선택하면 애플리케이션의 패키지 유형이 Image가 됩니다. '(Image)'라는 표시가 없는 런타임 중 하나를 선택하면 애플리케이션의 유형이 Zip이 됩니다. Image 패키지 유형과 Zip 패키지 유형의 차이점에 대한 자세한 내용은 AWS Lambda 개발자 가이드에서 [Lambda 배포 패키지](#)를 참조하세요.

4. 선택한 런타임에 따라 SAM 애플리케이션의 의존성 관리자와 런타임 아키텍처를 선택하라는 메시지가 표시될 수 있습니다.

### Dependency Manager

Gradle 또는 Maven 중에서 선택하세요.

**Note**

선택한 빌드 자동화 도구는 Java 런타임에만 사용할 수 있습니다.

### Architecture

x86\_64 또는 arm64중에서 선택하세요.

x86\_64 기반 기본 환경 대신 ARM64 기반 에뮬레이션 환경에서 서버리스 애플리케이션을 실행하는 옵션은 다음 런타임에서 사용할 수 있습니다.

- nodejs12.x (ZIP 파일 및 이미지)
- nodejs14.x (ZIP 파일 및 이미지)
- python3.8 (ZIP 파일 및 이미지)
- python3.9 (ZIP 파일 및 이미지)
- python3.10 (ZIP 파일 및 이미지)



- python3.11 (ZIP 파일 및 이미지)
- java8.a2을 지원하는 Gradle (ZIP 파일 및 이미지)
- java8.a2을 지원하는Maven (ZIP 파일만)
- java11을 지원하는 Gradle (ZIP 파일 및 이미지)
- java11을 지원하는 Maven (ZIP 파일만)

#### Important

ARM64 기반 환경에서 애플리케이션을 실행하려면 AWS CLI 버전 1.33.0 이상을 설치해야 합니다. 자세한 내용은 [필수 조건](#) 섹션을 참조하세요.

5. 새 프로젝트의 위치를 선택합니다. 기존의 작업 폴더가 있으면 그중에서 폴더를 선택하거나, 새 폴더를 생성하고 사용하세요. 예시에서 이름이 **MY-SAM-APP**인 폴더를 생성할 작업 폴더가 없습니다를 선택합니다.
6. 프로젝트의 이름을 입력합니다. `my-sam-app-nodejs`를 입력하세요. Enter 키를 누르면 VS Code용 도구 키트로 프로젝트를 생성하는 데 몇 분 정도 걸립니다.

프로젝트가 생성되면 현재 작업 폴더에 애플리케이션이 추가됩니다. 탐색기 창에 해당 프로젝트가 보여야 합니다.

## 서버리스 애플리케이션 열기 (로컬)

로컬 개발 호스트에서 서버리스 애플리케이션을 열려면 애플리케이션의 템플릿 파일이 들어 있는 폴더를 여세요.

1. File에서 Open Folder...를 선택합니다.
2. Open Folder 대화 상자에서 열리는 서버리스 애플리케이션 폴더를 찾습니다.
3. Select Folder 버튼을 선택합니다.

애플리케이션 폴더를 열면 탐색기 창에 나타납니다.

## 템플릿(로컬)에서 서버리스 애플리케이션 실행 및 디버깅

VS Code용 도구 키트를 사용하면 서버리스 애플리케이션을 디버깅하고 로컬 개발 환경에서 실행하는 환경을 설정할 수 있습니다.

VS Code [CodeLens](#) 기능으로 적절한 Lambda 함수를 식별하여 디버깅 동작 구성을 시작합니다. CodeLens를 사용하면 소스 코드와의 콘텐츠 인식 상호 작용이 가능합니다. CodeLens 기능에 액세스할 수 있는지 확인하는 방법은 이 항목 앞부분의 [필수 조건](#) 섹션을 확인하세요.

### Note

여기서는 JavaScript를 사용하여 애플리케이션을 디버깅합니다. 하지만 VS Code용 도구 키트에서 제공되는 디버깅 기능을 다음 언어 및 런타임에 사용할 수 있습니다.

- C# – .NET Core 2.1, 3.1; .NET 5.0
- JavaScript/TypeScript – Node.js 12.x, 14.x
- Python — 3.6, 3.7, 3.8, 3.9, 3.10, 3.11
- Java – 8, 8.al2, 11
- Go – 1.x

선택한 언어는 CodeLens가 Lambda 핸들러를 탐지하는 방법에도 영향을 미칩니다. 자세한 내용은 [코드에서 Lambda 함수 실행 및 디버깅](#) 섹션을 참조하세요.

여기에서는 이 항목의 [새 서버리스 애플리케이션 생성 \(로컬\)](#) 섹션 앞부분에서 생성한 예제 애플리케이션을 사용합니다.

1. VS Code 파일 탐색기에서 애플리케이션 파일을 보려면 View, Explorer를 선택합니다.
2. 애플리케이션 폴더(예: my-sample-app)에서 template.yaml 파일을 엽니다.

### Note

template.yaml과 다른 이름의 템플릿을 사용하는 경우 YAML 파일에서 CodeLens 표시기를 사용할 수 없습니다. 즉, 디버깅 구성을 수동으로 추가해야 합니다.

3. template.yaml 편집기에서 서버리스 리소스를 정의하는 템플릿의 Resources 섹션으로 이동하세요. 이 경우 AWS::Serverless::Function 유형의 HelloWorldFunction 리소스입니다.

이 리소스의 CodeLens 표시기에서 디버그 구성 추가를 선택합니다.

4. Command Palette에서 AWS SAM 애플리케이션을 실행할 런타임을 선택합니다.
5. launch.json 파일 편집기에서 다음의 구성 속성 값을 편집하거나 선택합니다.

- "name" - Run 보기의 Configuration 드롭다운 필드에서 알아보기 쉬운 이름을 입력합니다.
- "target" - AWS SAM 템플릿이 디버깅 세션의 진입점이 되도록 값이 "template" 인지 확인합니다.
- "templatePath" - template.yaml 파일의 상대 경로 또는 절대 경로를 입력합니다.
- "logicalId" - 이름이 AWS SAM 템플릿의 Resources 섹션에 지정된 이름과 일치하는지 확인합니다. 이 예에서는 AWS::Serverless::Function 유형의 HelloWorldFunction입니다.

launch.json 파일의 이들 항목과 기타 항목에 대한 자세한 내용은 [서버리스 애플리케이션 디버깅을 위한 구성 옵션](#) 섹션을 참조하세요.

6. 디버그 구성이 만족스럽다면 launch.json을 저장합니다. 그런 다음 RUN 보기에서 녹색 '재생' 버튼을 선택하면 디버깅이 시작됩니다.

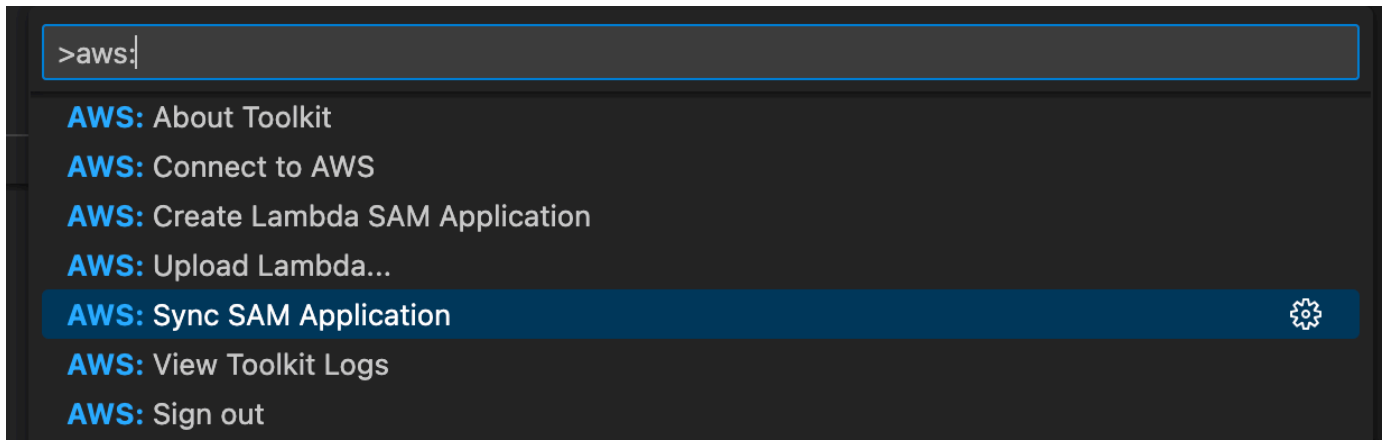
디버깅 세션이 시작되면 DEBUG CONSOLE 패널에 디버깅 결과가 표시되고 Lambda 함수에서 반환된 모든 값이 표시됩니다. (AWS SAM 애플리케이션을 디버깅할 때, Output 패널의 Output 채널을 AWS 도구 키트로 선택하세요.)

## 애플리케이션 동기화

AWS Toolkit for Visual Studio Code로 AWS SAM CLI 명령 `sam sync`를 실행하여 서버리스 애플리케이션을 AWS 클라우드에 배포합니다. AWS SAM 동기화에 대한 자세한 방법은 AWS Serverless Application Model 개발자 안내서의 [AWS SAM CLI command reference](#)를 참조하세요.

다음은 VS Code용 도구 키트에서 `sam sync`를 사용하여 AWS 클라우드에 서버리스 애플리케이션을 배포하는 방법을 설명합니다.

1. VS Code의 기본 메뉴에서 View를 확장한 다음 Command Palette를 선택하여 Command Palette 팔레트를 엽니다.
2. Command Palette에서 AWS를 검색하고 Sync SAM Application을 선택하여 동기화 설정을 시작합니다.



3. 서버리스 애플리케이션을 동기화할 AWS 리전을 선택합니다.
4. 배포에 사용할 `template.yaml` 파일을 선택합니다.
5. 기존 Amazon S3 버킷을 선택하거나 애플리케이션을 배포할 새 Amazon S3 버킷 이름을 입력합니다.

#### Important

Amazon S3 버킷은 다음 요구 사항을 충족해야 합니다.


- 버킷은 동기화하려는 리전에 있어야 합니다.
- Amazon S3 버킷 이름은 Amazon S3의 모든 기존 버킷 이름에서 전역적으로 고유해야 합니다.

6. 서버리스 애플리케이션에 패키지 유형이 Image인 함수가 포함되어 있는 경우, 이 배포에서 사용할 수 있는 Amazon ECR 리포지토리 이름을 입력합니다. 이 리포지토리는 배포하려는 리전에 있어야 합니다.
7. 이전 배포 목록에서 배포 스택을 선택하거나 새 스택 이름을 입력하여 새 배포 스택을 생성합니다. 그런 다음 동기화 프로세스를 시작합니다.

#### Note

이전 배포에서 사용한 스택은 작업 폴더 및 리전별로 리콜됩니다.

8. 동기화 프로세스 중에는 배포 상태는 VS Code의 Terminal 탭에 캡처됩니다. 터미널 탭에서 동기화가 성공했는지 확인하세요. 오류가 발생하면 알림을 받게 됩니다.

 Failed to deploy SAM application.

**Note**

동기화에 대한 자세한 정보를 보려면 Command Palette에서 AWS Toolkit for Visual Studio Code 로그에 액세스하세요.

명령 팔레트에서 AWS Toolkit for Visual Studio Code 로그에 액세스하려면 View를 확장하고 Command Palette를 선택한 다음 **AWS: View AWS Toolkits Logs**를 검색하여 결과 목록에서 선택하세요.

배포가 완료되면 AWS Explorer에서 애플리케이션을 볼 수 있습니다. 애플리케이션으로 생성된 Lambda 함수를 호출하는 방법에 대한 자세한 내용은 사용 설명서의 [원격 Lambda 함수와 상호 작용](#) 항목을 참조하세요.

## AWS 클라우드에서 서버리스 애플리케이션 삭제

서버리스 애플리케이션을 삭제하려면 이전에 AWS 클라우드에 배포한 AWS CloudFormation 스택을 삭제해야 합니다. 이 방법으로 로컬 호스트에서 애플리케이션 디렉터리를 삭제할 수 없습니다.

1. [AWS 익스플로러](#)를 엽니다.
2. AWS Toolkit Explorer 창에서 삭제하려는 배포된 애플리케이션이 포함된 리전을 확장한 다음 AWS CloudFormation을 확장합니다.
3. 삭제할 서버리스 애플리케이션에 해당하는 AWS CloudFormation 스택 이름의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴에서 Delete AWS CloudFormation Stack(스택 삭제)을 선택합니다.
4. 선택한 스택을 삭제하려면 Yes를 선택합니다.

스택이 삭제되면 AWS Explorer AWS CloudFormation 목록에 Toolkit for VS Code가 보이지 않습니다.

## 코드에서 Lambda 함수 실행 및 디버깅

AWS SAM 애플리케이션을 테스트 할 때 AWS SAM 템플릿으로 정의된 다른 리소스는 제외하고 Lambda 함수만 실행 및 디버깅하세요. 이는 [CodeLens](#) 기능을 사용하여 직접 호출할 수 있는 소스 코드에서 Lambda 함수 핸들러를 식별하는 방법입니다.

CodeLens로 감지되는 Lambda 핸들러는 애플리케이션에 사용 중인 언어와 런타임에 따라 다릅니다.

언어/런타임	CodeLens로 식별되는 Lambda 함수 기준
C# (dotnetcore2.1, 3.1; .NET 5.0)	<p>함수에는 다음 기능을 포함합니다.</p> <ul style="list-style-type: none"> <li>• 퍼블릭 클래스의 퍼블릭 함수입니다.</li> <li>• 하나 또는 두 개의 파라미터가 있습니다. 파라미터가 두 개인 경우 두 번째 파라미터가 <code>ILambdaContext</code> 인터페이스를 구현합니다.</li> <li>• VS Code WorkSpace 폴더 내 상위 폴더에 <code>*.csproj</code> 파일이 있습니다.</li> </ul> <p><a href="#">ms-dotnettools.csharp 확장 프로그램</a> (또는 C# 언어를 지원하는 모든 확장 프로그램)을 설치하고 활성화합니다.</p>
JavaScript/TypeScript (Node.js 12.x, 14.x)	<p>함수에는 다음 기능도 포함합니다.</p> <ul style="list-style-type: none"> <li>• 최대 세 개의 파라미터가 있는 추출 함수입니다.</li> <li>• VS Code WorkSpace 폴더 내 상위 폴더에 <code>package.json</code> 파일이 있습니다.</li> </ul>
Python (3.7, 3.8, 3.9, 3.10, 3.11)	<p>함수에는 다음 기능도 포함합니다.</p> <ul style="list-style-type: none"> <li>• 최상위 함수입니다.</li> <li>• VS Code WorkSpace 폴더 내 상위 폴더에 <code>requirements.txt</code> 파일이 있습니다.</li> </ul> <p><a href="#">ms-python.python 확장 프로그램</a> (또는 Python 언어를 지원하는 모든 확장 프로그램)을 설치하고 활성화합니다.</p>
Java (8, 8.al2, 11)	<p>함수에는 다음 기능도 포함합니다.</p> <ul style="list-style-type: none"> <li>• 추상이 아닌 공개 클래스의 공개 함수입니다.</li> </ul>

언어/런타임	CodeLens로 식별되는 Lambda 함수 기준
	<ul style="list-style-type: none"> <li>• 한 개, 두 개 또는 세 개의 파라미터가 있습니다. <ul style="list-style-type: none"> <li>• 파라미터 1개: 무엇이든 파라미터가 될 수 있습니다.</li> <li>• 파라미터 2개: 파라미터는 <code>java.io.InputStream</code> 과 <code>java.io.OutputStream</code> 이 되거나 마지막 파라미터가 <code>com.amazonaws.services.lambda.runtime.Context</code> 여야 합니다.</li> <li>• 파라미터 3개: 파라미터는 <code>java.io.InputStream</code> 과 <code>java.io.OutputStream</code> 이여야 하고 마지막 매개변수는 <code>com.amazonaws.services.lambda.runtime.Context</code> 여야 합니다.</li> </ul> </li> <li>• VS Code WorkSpace 폴더 내 상위 폴더에 <code>build.gradle</code> (Gradle) 또는 <code>pom.xml</code> (Maven) 파일이 있습니다.</li> </ul> <p><a href="#">redhat.java 확장 프로그램</a> (또는 Java 언어를 지원하는 모든 확장 프로그램)을 설치하고 활성화합니다. 사용 중인 Java 런타임이 무엇이든 확장 프로그램에 Java 11이 설치되어 있어야 합니다.</p> <p><a href="#">vscjava.vscode-java-debug</a> 확장 프로그램 (또는 Java debugger를 지원하는 모든 확장 프로그램)을 설치하고 활성화합니다.</p>

언어/런타임	CodeLens로 식별되는 Lambda 함수 기준
Go (1.x)	<p>이 함수에는 다음 기능도 포함합니다.</p> <ul style="list-style-type: none"> <li>• 최상위 함수입니다.</li> <li>• 0~2개의 인수를 사용합니다. 인수가 2개일 경우, 첫 번째 인수는 <code>context.Context</code> 를 구현합니다.</li> <li>• 0~2개의 인수를 반환합니다. 인수가 0개 이상인 경우, 마지막 인수는 <code>error</code>를 구현합니다.</li> <li>• VS Code 작업 폴더 내에 <code>go.mod</code> 파일이 있습니다.</li> </ul> <p><a href="#">golang.go 확장 프로그램</a>이 설치, 구성 및 활성화되어 있습니다.</p>

애플리케이션 코드에서 직접 서버리스 애플리케이션을 실행하고 디버깅하려면

1. VS Code 파일 탐색기에서 애플리케이션 파일을 보려면 View, Explorer를 선택합니다.
2. 애플리케이션 폴더(예: my-sample-app)에서 함수 폴더(hello-world)를 확장하여 `app.js` 파일을 엽니다.
3. 적합한 Lambda 함수 핸들러를 식별하는 CodeLens에서 Add Debug Configuration을 선택합니다.
4. Command Palette AWS SAM 애플리케이션을 실행할 런타임을 선택합니다.
5. `launch.json` 파일의 편집기에서 다음 구성의 속성 값을 수정하거나 확인합니다.
  - `"name"` - Run 보기의 Configuration 드롭다운 필드에 표시할 알아보기 쉬운 이름을 입력합니다.
  - `"target"` - Lambda 함수 핸들러가 바로 호출되도록 값이 `"code"`인지 확인합니다.
  - `"lambdaHandler"` - Lambda가 함수를 호출하는 코드 내에 메서드 이름을 입력합니다. 예를 들어 JavaScript로 작성된 애플리케이션의 경우 기본값은 `app.lambdaHandler`입니다.
  - `"projectRoot"` - Lambda 함수가 포함된 애플리케이션 파일의 경로를 입력합니다.
  - `"runtime"` - Lambda 실행 환경에 유효한 런타임을 입력하거나 확인합니다(예: `"nodejs.12x"`).



- "payload" - 다음 옵션 중 하나를 선택하여 Lambda 함수에 입력으로 제공할 이벤트 페이로드를 정의합니다.
- "json": 이벤트 페이로드를 정의하는 JSON 형식의 키 값 페어를 정의합니다.
- "path": 이벤트 페이로드로 사용되는 파일의 경로입니다.

아래 예제에서 "json" 옵션은 페이로드를 정의합니다.

launch.json 파일의 해당 항목 및 기타 항목에 대한 자세한 내용은 [서버리스 애플리케이션 디버깅을 위한 구성 옵션](#) 섹션을 참조하세요.

6. 디버그 구성이 만족스럽다면 RUN 옆에 있는 녹색 재생 화살표를 선택하여 디버깅을 시작합니다.

디버깅 세션이 시작되면 DEBUG CONSOLE 패널에 디버깅 출력이 표시되고 Lambda 함수에서 반환된 모든 값이 표시됩니다. (AWS SAM 애플리케이션을 디버깅할 때, Output 패널의 Output 채널을 AWS 도구 키트로 선택합니다.)

## 로컬 Amazon API Gateway 리소스 실행 및 디버깅

invokeTarget.target=api로 type=aws-sam의 VS Code 시작 구성을 실행하면 template.yaml에 지정된 AWS SAM API Gateway 로컬 리소스를 실행하거나 디버깅할 수 있습니다.

### Note

API Gateway는 REST와 HTTP 두 가지 유형의 API를 지원합니다. 그러나 AWS Toolkit for Visual Studio Code를 사용한 API Gateway 기능은 REST API만 지원합니다. HTTP API를 'API Gateway V2 API'라고 부르기도 합니다.

### 로컬 API Gateway 리소스 실행 및 디버깅하기

1. 다음 방법 중 하나를 선택하여 AWS SAM API Gateway 리소스의 시작 구성을 생성하세요.
  - 옵션 1: AWS SAM 프로젝트에 있는 핸들러 소스 코드(.js, .cs, or .py 파일)로 이동하여 Lambda 핸들러 위로 마우스를 가져간 다음 디버그 구성 추가 CodeLens를 선택합니다. 그런 다음 메뉴에서 API Event를 선택합니다.

- 옵션 2: `launch.json`을 수정하고 다음 구문을 사용하여 새 시작 구성을 생성합니다.

```
{
  "type": "aws-sam",
  "request": "direct-invoke",
  "name": "myConfig",
  "invokeTarget": {
    "target": "api",
    "templatePath": "n12/template.yaml",
    "logicalId": "HelloWorldFunction"
  },
  "api": {
    "path": "/hello",
    "httpMethod": "post",
    "payload": {
      "json": {}
    }
  },
  "sam": {},
  "aws": {}
}
```

2. VS Code Run 패널에서 시작 구성 (위 예시의 `myConfig`에서)을 선택합니다.
3. (선택 사항) Lambda 프로젝트 코드에 중단점을 추가할 수 있습니다.
4. F5를 입력하거나 Run 패널에서 Play를 선택합니다.
5. 출력 창에 결과가 나타납니다.

## 구성

`invokeTarget.target` 속성 값 `api`를 사용하면 도구 키트로 `api` 필드를 지원하는 시작 구성 검증 및 동작을 변경할 수 있습니다.

```
{
  "type": "aws-sam",
  "request": "direct-invoke",
  "name": "myConfig",
  "invokeTarget": {
```

```

    "target": "api",
    "templatePath": "n12/template.yaml",
    "logicalId": "HelloWorldFunction"
  },
  "api": {
    "path": "/hello",
    "httpMethod": "post",
    "payload": {
      "json": {}
    },
    "queryString": "abc=def&qrs=tuv",
    "headers": {
      "cookie": "name=value; name2=value2; name3=value3"
    }
  },
  "sam": {},
  "aws": {}
}

```

예시의 값을 다음과 같이 변경하세요.

`invokeTarget.logicalId`

API 리소스.

경로

시작 구성이 요청하는 API 경로(예: `"path": "/hello"`).

`invokeTarget.templatePath`로 지정된 `template.yaml`에서 확인한 유효한 API 경로여야 합니다.

`httpMethod`

동사 'delete', 'get', 'head', 'options', 'patch', 'post', 'put' 중 하나입니다.

payload

요청에 보낼 JSON 페이로드(HTTP 본문)로, [lambda.payload](#) 필드와 구조 및 규칙이 같습니다.

`payload.path`는 JSON 페이로드가 포함된 파일을 가리킵니다.

`payload.json`은 JSON 페이로드를 인라인으로 지정합니다.

## headers

이름-값 쌍의 옵션 맵은 다음 예와 같은 요청에 포함할 HTTP 헤더를 지정하는 데 사용됩니다.

```
"headers": {
  "accept-encoding": "deflate, gzip;q=1.0, *;q=0.5",
  "accept-language": "fr-CH, fr;q=0.9, en;q=0.8, de;q=0.7, *;q=0.5",
  "cookie": "name=value; name2=value2; name3=value3",
  "user-agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.198 Safari/537.36",
}
```

## querystring

요청의 querystring을 설정하는 옵션 문자열(예: "querystring": "abc=def&ghi=jk1").

## AWS

AWS 연결 정보를 제공하는 방법. 자세한 내용은 [서버리스 애플리케이션 디버깅을 위한 구성 옵션](#) 섹션에서 AWS connection ("aws") properties 표를 참조하세요.

## sam

AWS SAM CLI로 애플리케이션을 빌드하는 방법. 자세한 내용은 [서버리스 애플리케이션 디버깅을 위한 구성 옵션](#) 섹션에서 AWS SAM CLI ("sam") 속성 표를 참조하세요.

## 서버리스 애플리케이션 디버깅을 위한 구성 옵션

디버그 구성을 수정하기 위해 launch.json 파일을 열고 VS Code [IntelliSense](#) 기능을 사용하여 유효한 속성을 보고 자동으로 완성할 수 있습니다. 편집기에서 Ctrl+Spacebar를 눌러 IntelliSense를 실행합니다.



```
"lambda": {
  "runtime": "nodejs12.x",
  "event": {
    "json": {}
  }
}
```

IntelliSense를 사용하면 바로 또는 AWS SAM 템플릿을 사용하여 Lambda 함수 호출의 속성을 찾아 정의할 수 있습니다. "lambda"(함수 실행 방법), "sam"(AWS SAM CLI가 애플리케이션을 빌드하는 방법) 및 "aws"(AWS 연결 정보를 제공하는 방법)에 대한 속성을 정의할 수도 있습니다.

### AWS SAM Lambda 핸들러 직접 호출/템플릿 기반 Lambda 호출

속성	설명
type	시작 구성을 관리하는 확장 프로그램을 지정합니다. AWS SAM CLI를 사용하여 로컬에서 빌드하고 디버그하려면 항상 aws-sam으로 설정하세요.
name	Debug launch configuration 목록에 표시할 알아보기 쉬운 이름을 정합니다.
request	지정된 확장 프로그램(aws-sam)으로 실행할 구성의 유형을 정합니다. 항상 Lambda 함수를 시작하도록 direct-invoke 로 설정합니다.
invokeTarget	<p>리소스를 호출하기 위한 진입점을 지정합니다.</p> <p>Lambda 함수를 직접 호출하려면 다음 invokeTarget 필드의 값을 설정합니다.</p> <ul style="list-style-type: none"> <li>target - code로 설정합니다.</li> <li>lambdaHandler - 호출할 Lambda 함수 핸들러의 이름입니다.</li> <li>projectRoot - Lambda 함수 핸들러가 포함된 애플리케이션 파일의 경로입니다.</li> <li>architecture — 로컬 SAM Lambda 애플리케이션이 실행되는 에뮬레이트된 환경의 프로세서 아키텍처입니다. 특정 런타임의 경우 기본 x86_64 아키텍처 대신 arm64를 선택할 수 있습니다. 자세한 내용은 <a href="#">새 서버리스 애플리케이션 생성 (로컬)</a> 섹션을 참조하세요.</li> </ul> <p>AWS SAM 템플릿을 사용하여 Lambda 리소스를 호출하려면 다음 invokeTarget 필드 값을 설정합니다.</p>

속성	설명
	<ul style="list-style-type: none"> <li>• <code>target - template</code>으로 설정합니다.</li> <li>• <code>templatePath</code> - AWS SAM 템플릿 파일의 경로입니다.</li> <li>• <code>logicalId</code> - 호출할 <code>AWS::Lambda::Function</code> 또는 <code>AWS::Serverless::Function</code> 의 리소스 이름입니다. 리소스 이름은 YAML 형식의 AWS SAM 템플릿에서 확인할 수 있습니다. 참고로 AWS Toolkit는 AWS SAM 템플릿에서 <code>PackageType: Image</code> 로 정의된 함수를 <a href="#">이미지 기반 Lambda 함수</a>로 인식합니다. 자세한 내용은 AWS Lambda개발자 안내서의 <a href="#">Lambda deployment packages</a>를 참조하세요.</li> </ul>

## Lambda("lambda") 속성

속성	설명
<code>environmentVariables</code>	<p>연산 파라미터를 Lambda 함수에 전달합니다. 예를 들어, Amazon S3 버킷에 기록하는 경우 기록하고 있는 버킷 이름을 하드 코딩하는 대신 환경 변수로 구성합니다.</p> <div style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p><b>Note</b></p> <p>서버리스 애플리케이션의 환경 변수를 지정할 때는 AWS SAM 템플릿 (<code>template.yaml</code>)과 <code>launch.json</code> 파일의 구성을 모두 추가해야 합니다.</p> <p>AWS SAM 템플릿의 환경 변수 형식 지정 예:</p> <pre>Resources:   HelloWorldFunction:     Type: AWS::Serverless::Function     Properties:       CodeUri: hello-world/       Handler: app.lambdaHandlerN10       Runtime: nodejs10.x       Environment:         Variables:           SAMPLE1: Default Sample 1 Value</pre> </div>

속성	설명
	<p>launch.json 파일의 환경 변수 형식 지정 예:</p> <pre data-bbox="673 283 1474 445"> "environmentVariables": {   "SAMPLE1": "My sample 1 value" } </pre>
payload	<p>Lambda 함수에 입력으로 제공할 이벤트 페이로드에 두 가지 옵션이 있습니다.</p> <ul style="list-style-type: none"> <li>• "json": 이벤트 페이로드를 정의하는 JSON 형식의 키 값 페어를 정의합니다.</li> <li>• "path": 이벤트 페이로드로 사용되는 파일의 경로입니다.</li> </ul>
memoryMB	<p>호출된 Lambda 함수의 실행하기 위한 메모리의 용량(메가바이트)을 지정합니다.</p>
runtime	<p>Lambda 함수가 사용하는 런타임을 지정합니다. 자세한 내용은 <a href="#">AWS Lambda 런타임</a>을 참조하세요.</p>
timeoutSec	<p>디버그 세션이 시간 초과 전까지 허용되는 시간(초)을 설정합니다.</p>

속성	설명
pathMappings	<p>컨테이너에서 실행되는 위치와 관련하여 로컬 코드의 위치를 지정합니다.</p> <p>기본적으로 Toolkit for VS Code는 localRoot 를 로컬 작업 폴더에 있는 Lambda 함수의 코드 루트로 설정하고 remoteRoot 를 Lambda에서 실행되는 코드의 기본 작업 디렉토리인 /var/task 로 설정합니다. Dockerfile에서 또는 AWS CloudFormation 템플릿 파일의 WorkingDirectory 파라미터로 작업 디렉토리를 변경했다면 디버거가 로컬로 설정된 중단점을 Lambda 컨테이너에서 실행 중인 코드에 성공적으로 매핑할 수 있도록 최소한 하나의 pathMapping 항목을 지정해야 합니다.</p> <p>launch.json 파일에서 pathMappings 의 형식 지정하기 예시:</p> <pre data-bbox="592 940 1507 1255">"pathMappings": [   {     "localRoot": " \${workspaceFolder}/sam-app/HelloWorldFunction ",     "remoteRoot": " /var/task "   } ]</pre> <p><b>경고</b></p> <ul style="list-style-type: none"> <li>.NET 이미지 기반 Lambda 함수의 경우 remoteRoot 항목은 빌드 디렉터리입니다.</li> <li>Node.js 기반 Lambda 함수의 경우 단일 경로 매핑 항목만 지정할 수 있습니다.</li> </ul>

VS Code용 도구 키트는 AWS SAM CLI를 사용하여 서버리스 애플리케이션을 로컬에 빌드하고 디버그합니다. launch.json 파일에 있는 "sam" 구성의 속성을 사용하여 AWS SAM CLI 명령어의 동작을 구성할 수 있습니다.



## AWS SAM CLI("sam") 속성

속성	설명	기본값
buildArguments	<p> <code>sam build</code> 명령으로 Lambda 소스 코드를 빌드하는 방법을 알아보세요. 빌드 옵션을 보려면 AWS Serverless Application Model 개발자 가이드에서 <a href="#">sam build</a>를 참조하세요.         </p>	빈 문자열
containerBuild	<p>           Lambda와 유사한 Docker 컨테이너 내부에 함수에 빌드할지 여부를 나타냅니다.         </p>	false
dockerNetwork	<p>           Lambda Docker 컨테이너에 연결되어 있는 기존 Docker 네트워크의 이름 또는 ID와 더불어 기본 브리지 네트워크를 지정합니다. 지정하지 않으면 Lambda 컨테이너는 기본 브리지 Docker 네트워크에만 연결됩니다.         </p>	빈 문자열
localArguments	<p>           추가 로컬 호출 인수 지정.         </p>	빈 문자열
skipNewImageCheck	<p>           명령어로 Lambda 런타임의 최신 Docker 이미지를 가져오는 단계를 건너뛰지 여부를 지정합니다.         </p>	false
template	<p>           고객 값 입력 파라미터로 AWS SAM 템플릿을 사용자 지정합니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 <a href="#">Parameters</a>를 참조하세요.         </p>	"parameters": {}

## AWS 연결("aws") 속성

속성	설명	기본값
credentials	자격 증명 파일에서 프로파일 (예:profile:default )을 선택하여 AWS 자격 증명을 가져옵니다.	VS Code용 도구 키트에도 적용된 기존의 <a href="#">공유 AWS 구성 파일 또는 공유 AWS자격 증명 파일</a> 의 AWS 자격 증명입니다.
region	서비스 AWS 리전(예: us-east-1)을 설정합니다.	활성 자격 증명 프로파일과 연결된 기본 AWS 리전.

### 예: 템플릿 시작 구성

다음은 AWS SAM 템플릿 대상의 시작 구성 파일 예시입니다.

```
{
  "configurations": [
    {
      "type": "aws-sam",
      "request": "direct-invoke",
      "name": "my-example:HelloWorldFunction",
      "invokeTarget": {
        "target": "template",
        "templatePath": "template.yaml",
        "logicalId": "HelloWorldFunction"
      },
      "lambda": {
        "payload": {},
        "environmentVariables": {}
      }
    }
  ]
}
```

### Code 시작 파일 구성 예

다음은 Lambda 함수 타겟의 시작 구성 파일 예시입니다.

```
{
  "configurations": [
```

```

    {
      "type": "aws-sam",
      "request": "direct-invoke",
      "name": "my-example:app.lambda_handler (python3.7)",
      "invokeTarget": {
        "target": "code",
        "projectRoot": "hello_world",
        "lambdaHandler": "app.lambda_handler"
      },
      "lambda": {
        "runtime": "python3.7",
        "payload": {},
        "environmentVariables": {}
      }
    }
  ]
}

```

## 서버리스 애플리케이션 문제 해결

Toolkit for VS Code를 사용하여 서버리스 애플리케이션을 개발할 때 발생할 수 있는 일반적인 오류와 이러한 오류를 해결하는 방법에 대해 자세히 설명합니다.

### 주제

- [SAM 시작 구성에서 samconfig.toml을 어떻게 사용할 수 있을까요?](#)
- ["RuntimeError: Container does not exist" 오류가 발생합니다.](#)
- [Error: "docker.errors.APIError: 500 Server Error ... 폴 레이트 리미트에 도달했습니다."](#)
- [Error: "500 Server Error: Mounting C:\Users\..."](#)
- [WSL을 사용하면 웹 뷰 \(예: "AWS의 Invoke" 형식\) 가 깨집니다.](#)
- [TypeScript 애플리케이션을 디버깅하고 있지만 중단점이 작동하지 않습니다.](#)

### SAM 시작 구성에서 samconfig.toml을 어떻게 사용할 수 있을까요?

sam.localArguments 시작 구성의 속성에 --config-file 인수를 구성하여 SAM CLI [samconfig.toml](#)의 위치를 지정합니다. 예를 들어 samconfig.toml 파일이 작업 공간의 상위 레벨에 있다면

```

"sam": {
  "localArguments": ["--config-file", "${workspaceFolder}/samconfig.toml"],

```

```
}

```

"RuntimeError: Container does not exist" 오류가 발생합니다.

시스템에 Docker 컨테이너 디스크 공간이 충분하지 않은 경우 `sam build` 명령이 이 오류를 표시합니다. 시스템의 사용 가능한 저장공간이 1~2GB라면 빌드가 시작되기 전에 시스템 저장공간이 가득 차지 않았더라도 처리 중에 `sam build`가 실패할 수 있습니다. 자세한 내용은 [GitHub issue](#)를 참조하세요.

Error: "docker.errors.APIError: 500 Server Error ... 풀 레이트 리미트에 도달했습니다."

Docker Hub는 익명 사용자의 요청을 제한합니다. 시스템이 리미트에 도달하면 Docker는 실행되지 않고 VS Code의 OUTPUT 보기에 다음 오류가 나타납니다.

```
docker.errors.APIError: 500 Server Error: Internal Server Error ("toomanyrequests: You
have
reached your pull rate limit. You may increase the limit by authenticating and
upgrading:
https://www.docker.com/increase-rate-limit")
```

시스템 Docker 서비스가 Docker Hub 자격 증명의 인증을 받았는지 확인하세요.

Error: "500 Server Error: Mounting C:\Users\..."

Windows 사용자는 AWS SAM 애플리케이션을 디버깅할 때 다음과 같은 Docker 마운트 오류를 볼 수 있습니다.

```
Fetching lambci/lambda:nodejs10.x Docker container image.....
2019-07-12 13:36:58 Mounting C:\Users\\AppData\Local\Temp\ ... as /var/
task:ro,delegated inside runtime container
Traceback (most recent call last):
...
requests.exceptions.HTTPError: 500 Server Error: Internal Server Error ...
```

(Docker 설정에서) 공유 드라이브의 자격 증명을 새로 고침 하세요.

WSL을 사용하면 웹 뷰 (예: "AWS의 Invoke" 형식) 가 깨집니다.

Cisco VPN 사용자라면 아는 VS Code 문제입니다. 자세한 내용은 [GitHub issue](#)를 참조하세요.

[WSL 트래킹 문제](#)에 해결 방법이 있습니다.

TypeScript 애플리케이션을 디버깅하고 있지만 중단점이 작동하지 않습니다.

컴파일된 JavaScript 파일과 소스 TypeScript 파일을 연결하는 소스 맵이 없는 경우에 발생합니다. 이 문제를 해결하려면 `tsconfig.json` 파일을 열고 다음 옵션과 값이 설정되어 있는지 확인하세요:  
`"inlineSourceMap": true.`

## Systems Manager 자동화 설명서로 작업

AWS Systems Manager은 AWS 인프라에 대한 가시성과 제어를 제공합니다. Systems Manager는 통합된 사용자 인터페이스를 제공하므로 여러 AWS 서비스의 운영 데이터를 보고 AWS 리소스 전체에서 운영 태스크를 자동화할 수 있습니다.

[Systems Manager 문서](#)는 Systems Manager가 관리형 인스턴스에서 실행하는 작업을 정의합니다. 자동화 문서는 Amazon Machine Image(AMI) 생성 또는 업데이트와 같은 일반적인 유지 관리 및 배포 태스크를 수행하는 데 사용하는 Systems Manager 문서 유형입니다. 여기에서는 AWS Toolkit for Visual Studio Code를 사용하여 자동화 문서를 생성, 편집, 게시 및 삭제하는 방법을 간략하게 설명합니다.

주제

- [전제 조건 및 필수 조건](#)
- [Systems Manager Automation 문서에 대한 IAM 권한](#)
- [새 Systems Manager 자동화 문서 생성](#)
- [기존 Systems Manager 자동화 문서 열기](#)
- [Systems Manager 자동화 문서 수정](#)
- [Systems Manager 자동화 문서 게시](#)
- [Systems Manager 자동화 문서 삭제](#)
- [Systems Manager 자동화 문서 실행](#)
- [VS Code용 도구 키트에서 Systems Manager 자동화 문서 문제 해결](#)

## 전제 조건 및 필수 조건

시작하기 전에 다음을 확인하세요.

- Visual Studio Code와 AWS Toolkit for Visual Studio Code 최신 버전이 설치되어 있어야 합니다. 자세한 내용은 [AWS Toolkit for Visual Studio Code 설치](#) 섹션을 참조하세요.
- Systems Manager에 대해 잘 알고 있어야 합니다. 자세한 정보는 [AWS Systems Manager 사용 설명서](#)를 참조하세요.

- Systems Manager 자동화 사용 사례에 대해 잘 알고 있어야 합니다. 자세한 내용은 AWS Systems Manager 사용 설명서의 [AWS Systems Manager 자동화](#)을 참조하세요.

## Systems Manager Automation 문서에 대한 IAM 권한

VS Code용 도구 키트에 Systems Manager 자동화 문서를 생성, 편집, 게시 및 삭제하는 데 필요한 AWS Identity and Access Management (IAM) 권한이 포함된 자격 증명 프로파일이 있어야 합니다. 다음 정책 문서는 주요 정책에서 사용할 수 있는 필수 IAM 권한을 정의합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:ListDocuments",
        "ssm:ListDocumentVersions",
        "ssm:DescribeDocument",
        "ssm:GetDocument",
        "ssm:CreateDocument",
        "ssm:UpdateDocument",
        "ssm:UpdateDocumentDefaultVersion",
        "ssm>DeleteDocument"
      ],
      "Resource": "*"
    }
  ]
}
```

IAM 정책을 업데이트하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 [IAM 정책 생성](#)을 참조하세요. 보안 인증 프로파일 설정 방법에 대한 자세한 내용은 [AWS IAM 자격 증명 단원](#)을 참조하세요.

## 새 Systems Manager 자동화 문서 생성

Visual Studio Code로 JSON 또는 YAML에서 새 자동화 문서를 만들 수 있습니다. 새 자동화 문서를 생성하면 제목 없는 파일로 표시됩니다. 파일 이름을 정하고 VS Code에 저장할 수 있지만 파일 이름은 AWS에 표시되지 않습니다.

새 자동화 문서를 생성하는 방법

1. VS Code를 엽니다.

2. View 메뉴에서 Command Palette를 선택합니다.
3. 명령 팔레트에 AWS Toolkit Create a new Systems Manager Document Locally를 입력합니다.
4. Hello World 예제 스타터 템플릿 중 하나를 선택합니다.
5. JSON 또는 YAML을 선택합니다.

새 자동화 문서가 생성됩니다.

#### Note

VS Code의 새 자동화 문서는 AWS에 표시되지 않습니다. 자동화 문서를 실행하려면 먼저 AWS에 게시해야 합니다.

## 기존 Systems Manager 자동화 문서 열기

AWS Explorer에서 기존 Systems Manager 자동화 문서를 찾습니다. 기존 자동화 문서를 열면 VS Code에 제목 없는 파일로 나타납니다.

### 자동화 문서 열기

1. VS Code를 엽니다.
2. 왼쪽 탐색창에서 AWS를 클릭하여 AWS 탐색기를 엽니다.
3. Systems Manager AWS탐색기에서 보려는 문서의 다운로드 아이콘을 선택한 다음 문서 버전을 선택합니다. 파일은 해당 버전 형식으로 열립니다. 그렇지 않으면 JSON으로 다운로드 또는 YAML로 다운로드를 선택합니다.

#### Note

자동화 문서를 VS Code 파일로 컴퓨터에 저장하면 AWS에서 볼 수 없습니다. 실행하기 전 AWS에 게시해야 합니다.

## Systems Manager 자동화 문서 수정

자동화 문서를 소유하고 있다면 AWS 탐색기에서 Systems Manager 문서의 Owned by Me(내 소유) 카테고리에서 볼 수 있습니다. AWS에 있는 자동화 문서를 소유할 수 있으며 이전에 VS Code에서 AWS에 게시한 새 문서나 업데이트된 문서를 소유할 수 있습니다.

VS Code에서 자동화 문서를 편집하면 AWS Management Console에서 하는 것보다 더 많은 작업을 수행할 수 있습니다. 예:

- JSON 및 YAML 형식 모두에 대한 스키마 검증이 있습니다.
- 문서 편집기에는 모든 유형의 자동화 단계를 만들 수 있는 스니펫이 있습니다.
- JSON 및 YAML의 다양한 옵션에 자동 완성 기능이 지원됩니다.

### 버전 작업

Systems Manager 자동화 문서는 변경 관리를 위해 버전을 사용합니다. VS Code에서 자동화 문서의 기본 버전을 선택할 수 있습니다.

#### 기본 버전 설정

- AWS 탐색기에서 기본 버전을 설정할 문서로 이동하고, 문서에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 연 다음 기본 버전 설정을 선택합니다.

#### Note

선택한 문서에 버전이 하나만 있는 경우 기본값을 변경할 수 없습니다.

## Systems Manager 자동화 문서 게시

VS Code에서 자동화 문서를 편집한 후 AWS에 게시할 수 있습니다.

#### 자동화 문서 게시

1. [기존 Systems Manager 자동화 문서 열기](#)에 설명된 절차에 따라 게시하려는 자동화 문서를 엽니다.
2. 게시할 내용을 변경합니다. 자세한 내용은 [Systems Manager 자동화 문서 수정](#) 섹션을 참조하세요.



- 열린 파일의 오른쪽 상단에서 업로드 아이콘을 선택합니다.
- 게시 워크플로 대화 상자에서 자동화 문서를 게시할 AWS 리전을 선택합니다.
- 새 문서를 게시하려면 빠른 생성을 선택합니다. 또는 빠른 업데이트를 선택하여 해당 AWS 리전의 자동화 문서를 업데이트합니다.
- 자동화 문서의 이름을 입력합니다.

기존 자동화 문서에 대한 업데이트를 AWS에 게시하면 새 버전이 문서에 추가됩니다.

## Systems Manager 자동화 문서 삭제

VSCode에서 자동화 문서를 삭제할 수 있습니다. 자동화 문서를 삭제하면 문서와 문서의 모든 버전이 삭제됩니다.

### Important

- 삭제는 실행 취소할 수 없습니다.
- 실행된 자동화 문서를 삭제해도 문서 시작 시 생성되거나 수정된 AWS 리소스는 삭제되지 않습니다.

### 자동화 문서 삭제

- VS Code를 엽니다.
- 왼쪽 탐색창에서 AWS를 클릭하여 AWS Explorer를 엽니다.
- Systems Manager AWS Explorer에서 삭제할 문서의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 문서 삭제를 선택합니다.

## Systems Manager 자동화 문서 실행

자동화 문서가 AWS에 게시된 후 이를 실행하여 AWS 계정에서 사용자를 대신하여 태스크를 수행할 수 있습니다. AWS Management Console, Systems Manager API, AWS CLI 또는 AWS Tools for PowerShell을 사용하여 자동화 문서를 실행합니다. 자동화 문서를 실행하는 방법에 대한 지침은 AWS Systems Manager 사용 설명서의 [단순 자동화 실행](#)을 참조하세요.

또는 AWS SDK 중 하나를 Systems Manager API와 함께 사용하여 자동화 문서를 실행하려는 경우 [AWS SDK 참조 자료](#)를 참조하세요.

**Note**

자동화 문서를 실행하면 AWS에서 새 리소스가 생성되며 청구 비용이 발생할 수 있습니다. 자동화 문서를 실행하기 전에 계정에서 생성되는 자동화 문서가 무엇인지 파악하는 것이 좋습니다.

## VS Code용 도구 키트에서 Systems Manager 자동화 문서 문제 해결

자동화 문서를 VS Code에 저장했지만 AWS Management Console에서 볼 수 없습니다.

VS Code에 자동화 문서를 저장해도 자동화 문서가 AWS에 게시되지 않습니다. 자동화 문서를 게시하는 방법에 대한 자세한 내용은 [Systems Manager 자동화 문서 게시](#) 섹션을 참조하세요.

권한 문제로 인해 자동화 문서를 게시할 수 없습니다.

AWS 자격 증명 프로파일에 자동화 문서를 게시하는 데 필요한 권한이 있는지 확인하세요. 권한 정책 예시는 [Systems Manager Automation 문서에 대한 IAM 권한](#) 단원을 참조하십시오.

자동화 문서를 AWS에 게시했지만 AWS Management Console에서 해당 문서가 보이지 않습니다.

AWS Management Console 탐색기 창에 보이는 AWS 리전에 문서를 게시했는지 확인하세요.

자동화 문서를 삭제했지만 생성된 리소스에 대한 요금이 계속 청구되고 있습니다.

자동화 문서를 삭제해도 해당 문서가 생성하거나 수정한 리소스는 삭제되지 않습니다. [AWS 결제 관리 콘솔](#)에서 생성한 AWS 리소스를 확인하고 요금을 살펴본 후 삭제할 리소스를 클릭하세요.

## 작업 대상 AWS Step Functions

AWS Toolkit for Visual Studio Code (VS Code) 는 에 대한 지원을 제공합니다 [AWS Step Functions](#). VS Code용 도구 키트를 사용하면 Step Functions 상태의 머신을 생성, 업데이트 및 실행할 수 있습니다.

주제

- [함께 작업하기 AWS Step Functions](#)

## 함께 작업하기 AWS Step Functions

AWS Toolkit for Visual Studio Code (VS Code) 를 사용하여 [상태 머신으로](#) 다양한 작업을 수행할 수 있습니다.

### 주제

- [필수 조건](#)
- [VS Code에서 상태 머신 작업](#)
- [상태 머신 템플릿](#)
- [상태 머신 그래프 화면](#)
- [코드 조각](#)
- [코드 완성 및 유효성 검사](#)

### 필수 조건

- 시스템이 [VS Code용 도구 키트 설치](#)에 필요한 필수 조건을 충족하는지 확인한 다음 해당 도구 키트를 설치하세요.
- AWS 탐색기를 열기 전, 자격 증명이 구성되어 있는지 확인하세요.

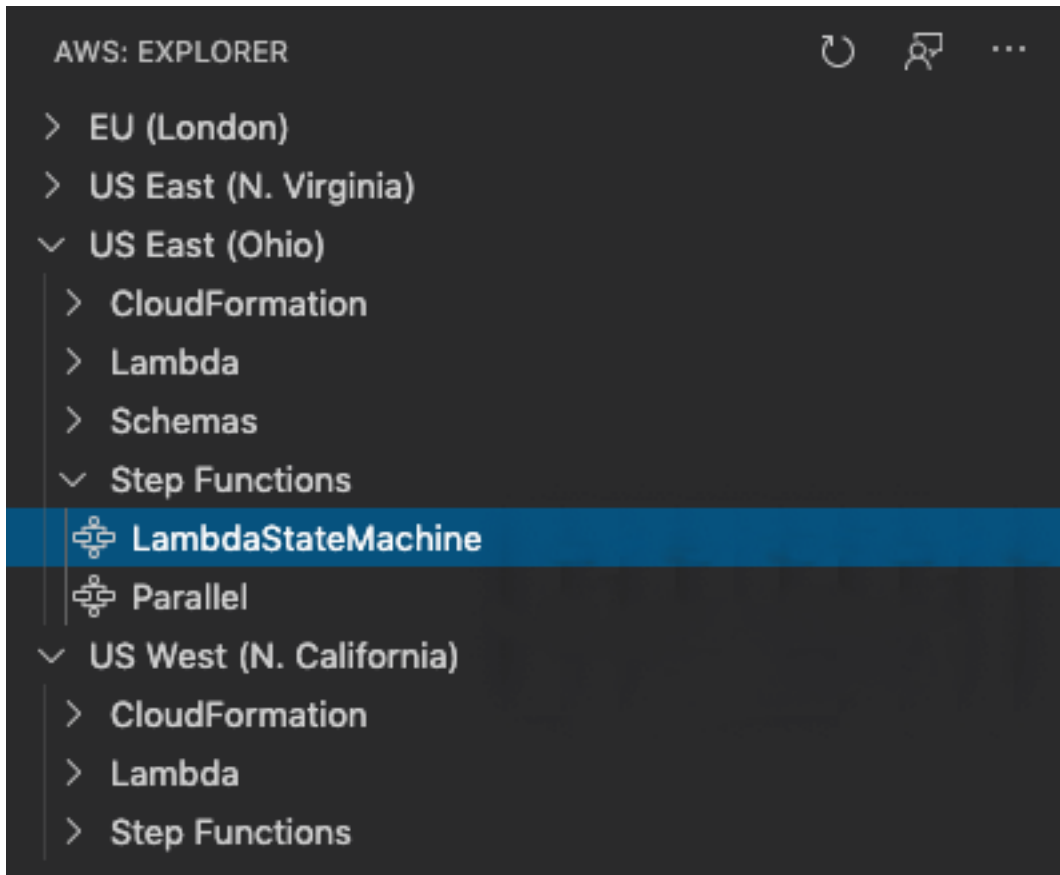
### VS Code에서 상태 머신 작업

VS 코드를 사용하여 원격 상태 머신과 상호 작용하고 상태 머신을 JSON 이나 YAML 형식으로 개발할 수 있습니다. 상태 머신을 생성 또는 업데이트하여 기존 상태 머신 목록을 나열한 후 실행하고 다운로드할 수 있습니다. 또한 VS Code를 통해 템플릿에서 새 상태 머신을 생성하고 상태 머신의 화면을 보고 코드 조각, 코드 완성 및 코드 유효성 검사를 할 수 있습니다.

### 기존 상태 머신 목록

상태 시스템을 이미 생성한 경우 해당 상태 시스템의 목록을 볼 수 있습니다.

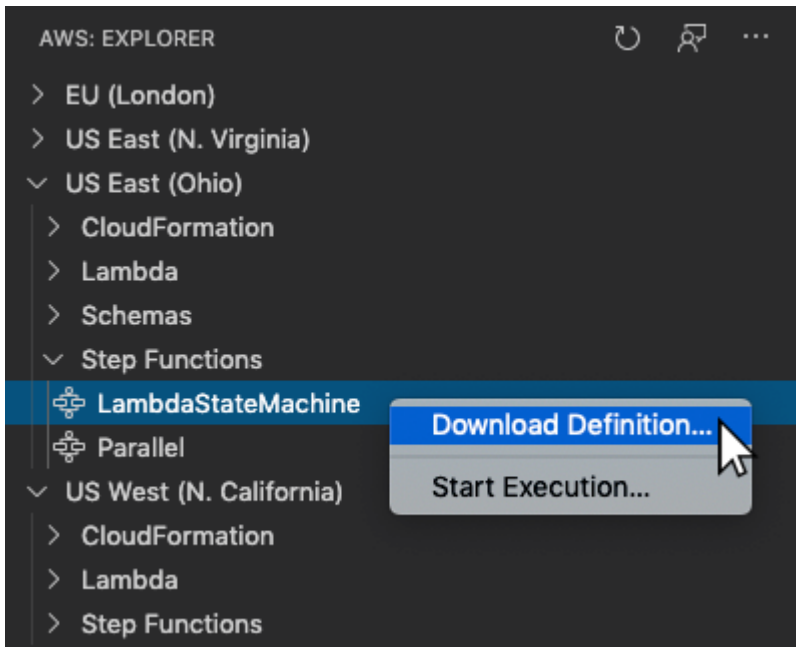
1. AWS Explorer를 엽니다.
2. Step Functions 선택
3. 계정에 있는 모든 상태 머신 목록이 나열되는지 확인합니다.



## 상태 머신 다운로드

### 상태 머신 다운로드하기

1. AWS Explorer에서 다운로드할 상태 머신을 마우스 오른쪽 버튼으로 클릭합니다.
2. 다운로드를 선택한 다음 상태 머신을 다운로드할 위치를 선택합니다.
3. 해당 상태 머신이 올바르게 다운로드되었는지 확인합니다.



## 상태 시스템 생성

새 상태 머신을 직접 생성하거나 템플릿을 사용할 수 있습니다. 템플릿에서 상태 머신을 생성하는 방법에 대한 자세한 내용은 상태 머신 템플릿(State Machine Templates) 섹션을 참조하세요. 새 상태 머신 생성하기

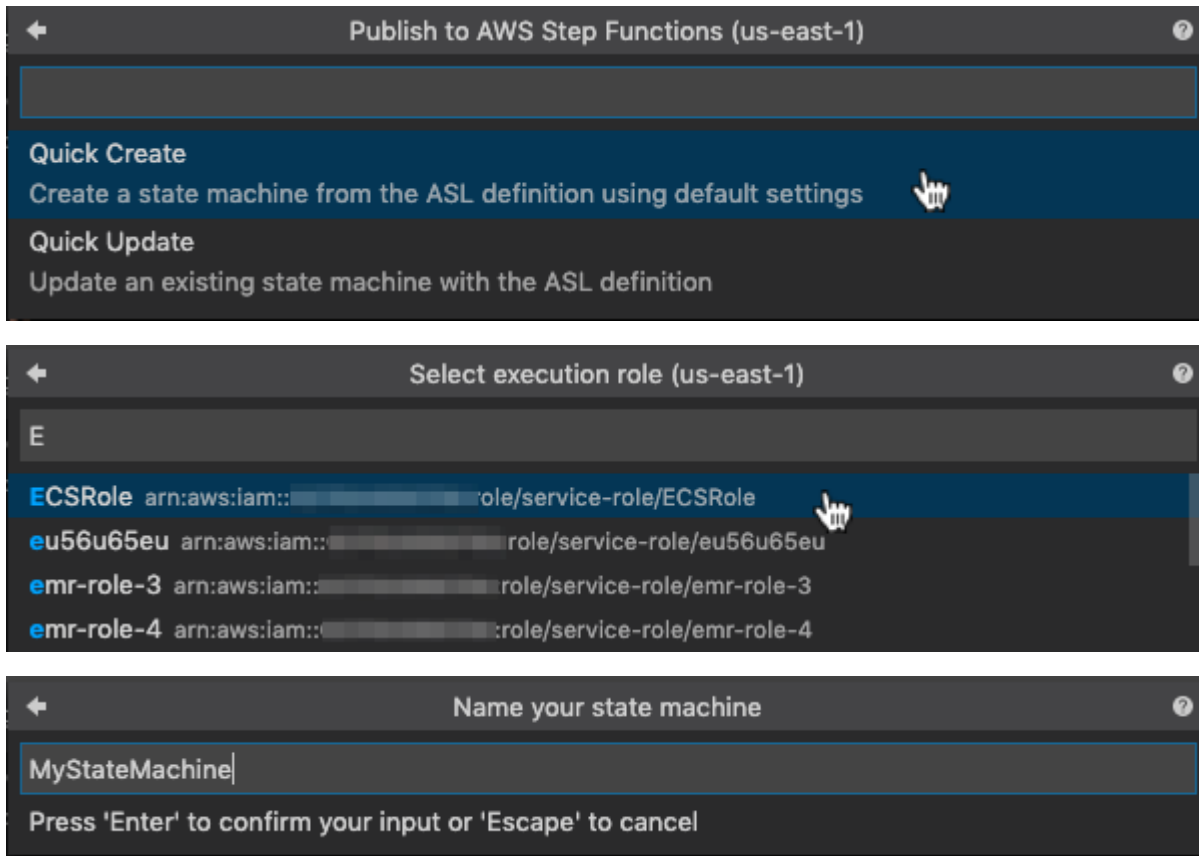
1. 상태 머신 정의를 사용하여 새 [Amazon States Language](#)(ASL) 파일을 생성합니다. 오른쪽 하단의 메뉴를 사용하여 Amazon States Language로 설정합니다.
2. Step Functions에 게시(Publish to Step Functions)를 선택합니다.

```

Publish to Step Functions | Render graph
1  {
2    "StartAt": "FirstState",
3    "States": {
4      "FirstState": {
5        "Type": "Task",
6        "Resource": "arn:aws:lambda:us-east-1:123456789012:function:Function",
7        "Next": "ChoiceState"
8      },
9      "ChoiceState": {
10     "Type": "Choice",

```

3. 빠른 생성(Quick Create)을 클릭하고 역할을 선택한 다음 상태 머신의 이름을 정합니다.



## 상태 머신 업데이트

### 상태 시스템 업데이트하기

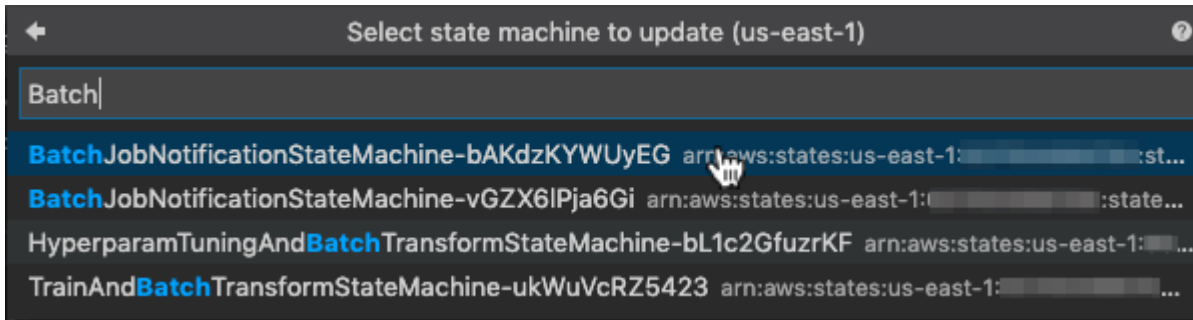
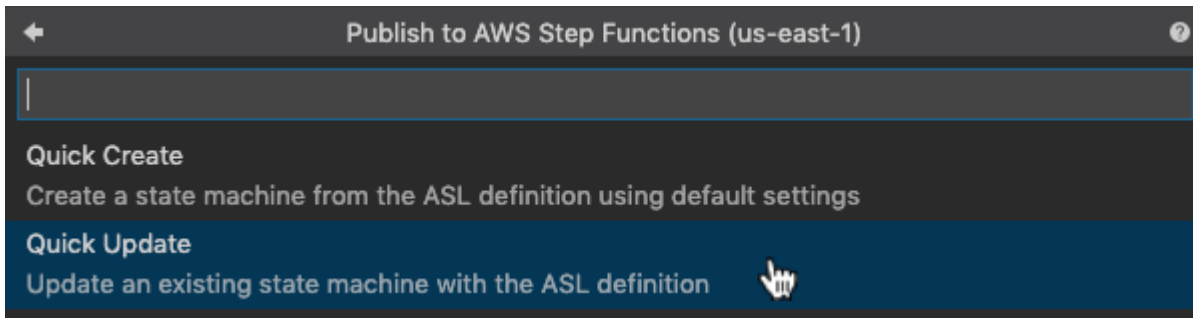
1. 해당 상태 시스템 정의를 사용하여 ASL 파일을 편집합니다.
2. Step Functions에 게시(Publish to Step Functions)를 선택합니다.

```

Publish to Step Functions | Render graph
1  {
2    "StartAt": "FirstState",
3    "States": {
4      "FirstState": {
5        "Type": "Task",
6        "Resource": "arn:aws:lambda:us-east-1:123456789012:function:Function",
7        "Next": "ChoiceState"
8      },
9      "ChoiceState": {
10     "Type": "Choice",

```

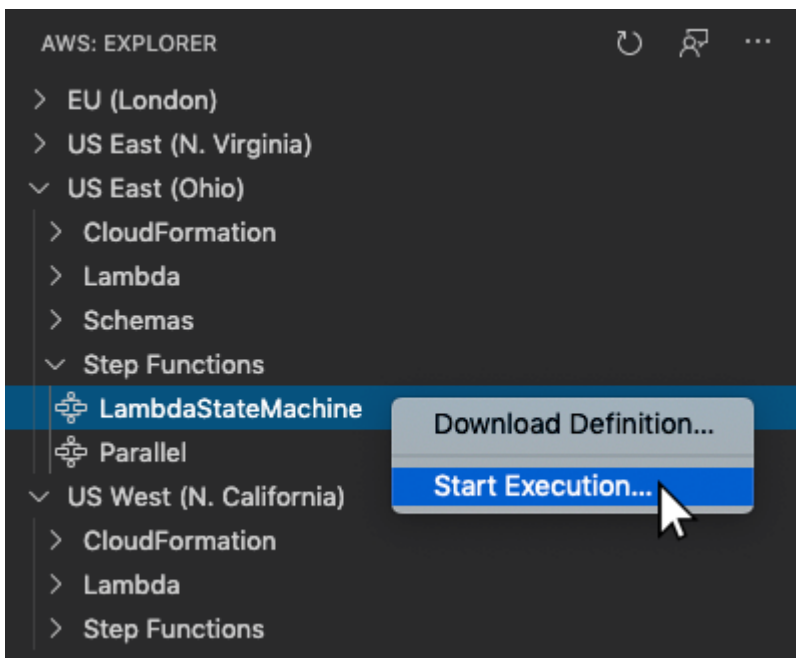
3. 빠른 업데이트(Quick Update)를 선택한 다음 업데이트할 상태 머신을 선택합니다.

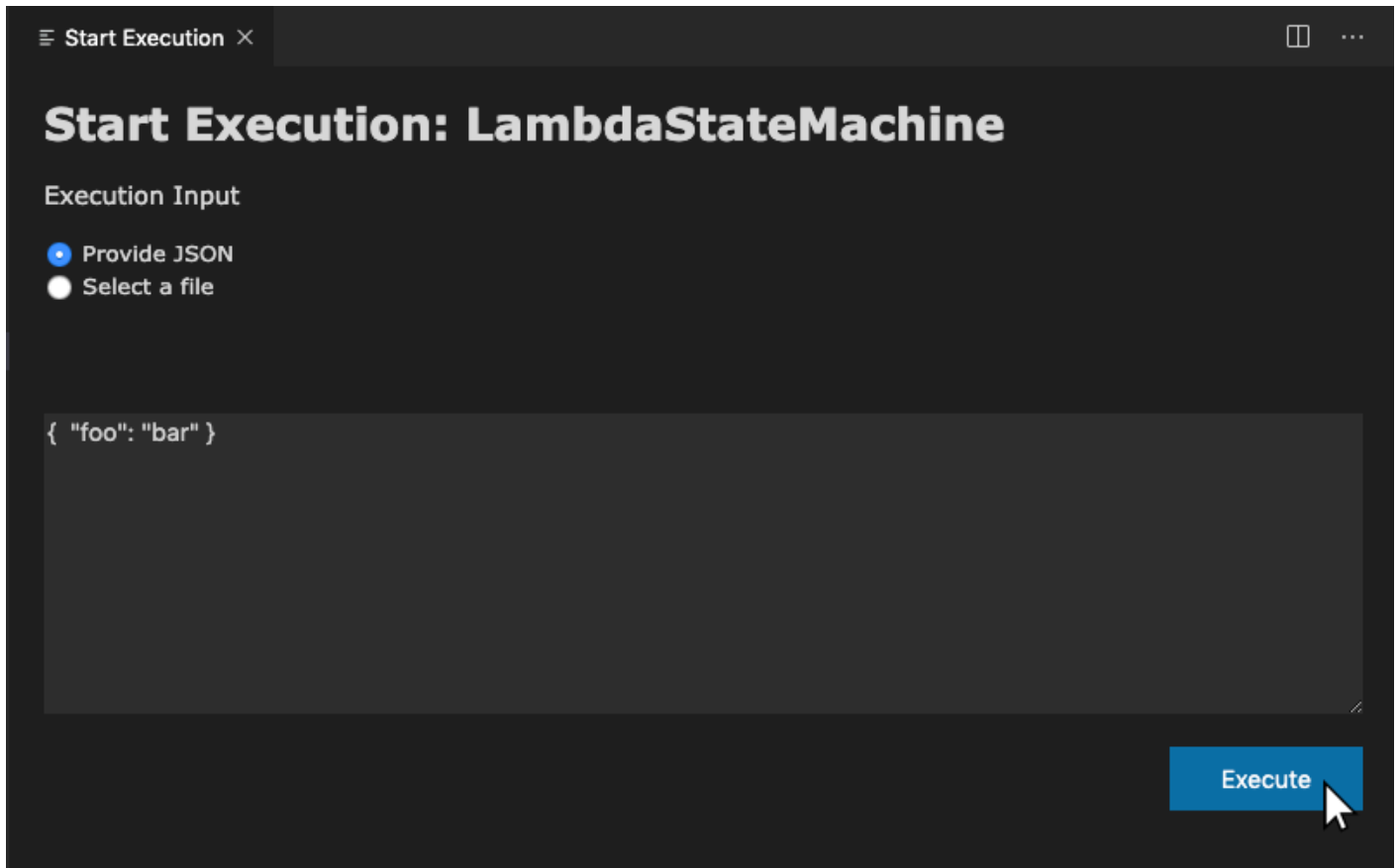


## 상태 머신 실행

### 상태 머신 실행하기

1. AWS Explorer에서 실행할 상태 머신 마우스 오른쪽 버튼으로 클릭합니다.
2. 상태 머신에 입력합니다. 파일에서 입력과 텍스트 상자에 모두 입력을 할 수 있습니다.
3. 상태 머신을 시작하고 잘 실행되는지 확인합니다.



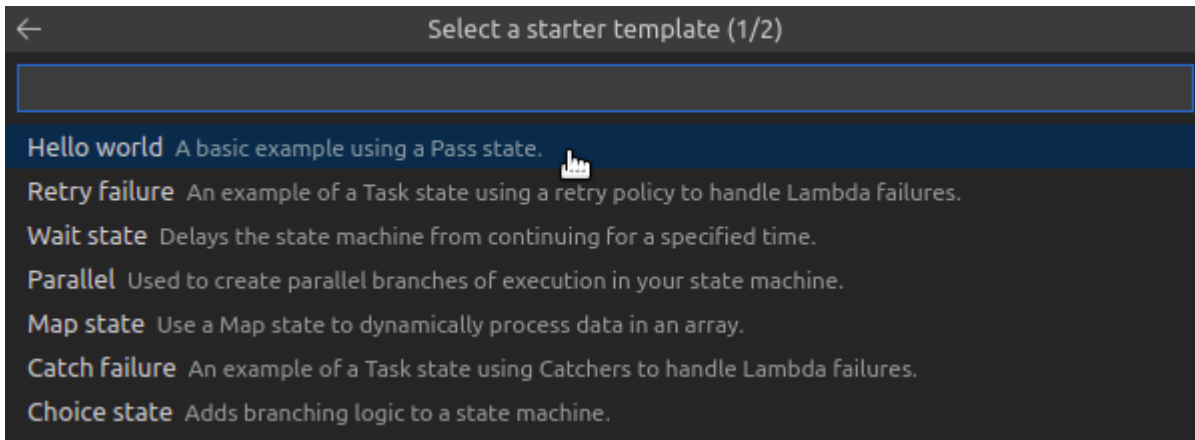


## 상태 머신 템플릿

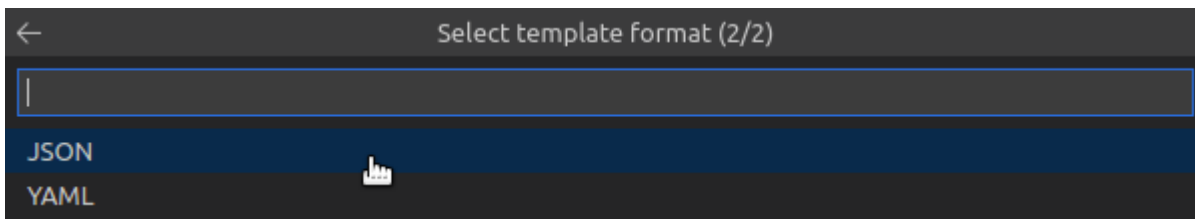
템플릿에서 상태 머신을 생성 할 수 있는 옵션이 있습니다. 템플릿에는 일반적으로 사용되는 여러 상태가 포함된 샘플 상태 머신 정의가 포함되어 있으며 이를 통해 시작할 수 있습니다. 상태 머신 템플릿 사용하기

1. VS Code에서 Command Palette를 엽니다.
2. AWS Toolkit로 새 단계 기능 상태 머신 생성(Toolkit Create a new Step Functions state machine)을 선택합니다.
3. 사용할 템플릿을 선택합니다.





4. JSON 또는 YAML 템플릿 형식 사용 여부를 선택합니다.



## 상태 머신 그래프 화면

그래프 화면을 사용하면 상태 머신의 모양을 그래픽 형식으로 볼 수 있습니다. 그래프 화면을 생성하면 탭이 열리고 상태 머신 JSON 또는 YAML 화면이 나타납니다. 작성 중인 상태 머신 정의를 해당 화면과 비교할 수 있습니다. 상태 머신 정의를 변경하면 화면도 변경됩니다.

### Note

상태 머신 정의 화면을 생성하려면 정의가 활성 편집기에 열려 있어야 합니다. 정의 파일을 닫거나 이름을 바꾸면 화면이 닫힙니다.

## 상태 머신 그래프 화면 생성하기

1. 상태 머신을 정의합니다.
2. VS Code에서 Command Palette를 엽니다.
3. 화면을 생성하려면 오른쪽 상단 가장자리에 있는 시각화 버튼을 사용하거나 AWS 렌더 그래프 (Render graph)를 선택합니다.

```

1  {
2    "Comment": "An example of the Amazon States Language using a choice
3    state.",
4    "StartAt": "FirstState",
5    "States": {
6      "FirstState": {
7        "Type": "Task",
8        "Resource":
9        "arn:aws:lambda:us-east-2:637554956784:function:Function",
10       "Next": "ChoiceState"
11     },
12     "ChoiceState": {
13       "Type": "Choice",
14       "Choices": [
15         {
16           "Variable": "$.foo",
17           "NumericEquals": 1,
18           "Next": "FirstMatchState"
19         },
20         {
21           "Variable": "$.foo",
22           "NumericEquals": 2,
23           "Next": "SecondMatchState"
24         }
25       ],
26       "Default": "DefaultState"
27     },
28     "FirstMatchState": {
29       "Type": "Task",
30       "Resource":
31       "arn:aws:lambda:us-east-2:637554956784:function:Function",
32       "Next": "NextState"
33     },
34     "SecondMatchState": {
35       "Type": "Task",
36       "Resource":
37       "arn:aws:lambda:us-east-2:637554956784:function:Function",
38       "Next": "NextState"
39     },
40     "NextState": {
41       "Type": "Task",
42       "Resource":
43       "arn:aws:lambda:us-east-2:637554956784:function:Function",
44       "Next": "End"
45     },
46     "End": {}
47   }
48 }

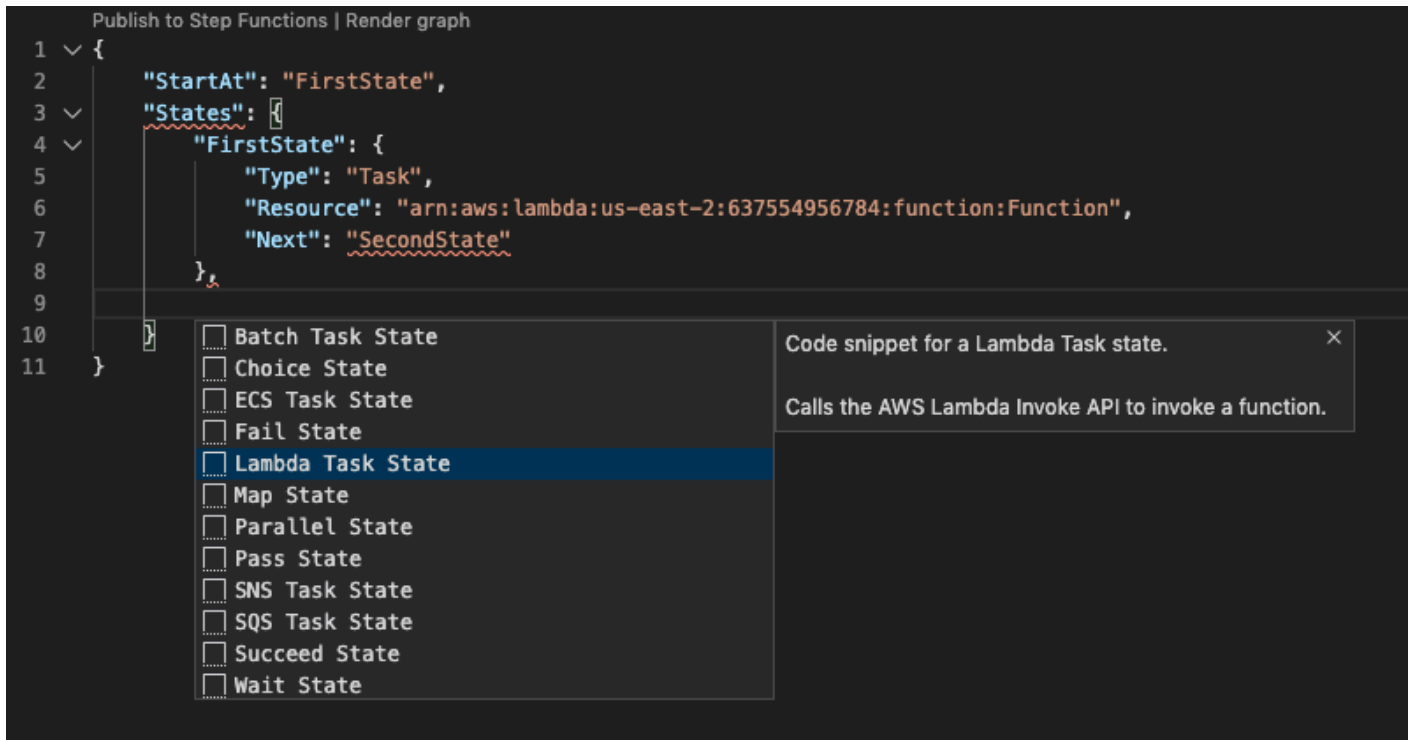
```

Ln 2, Col 81 Spaces: 4 UTF-8 LF Amazon States Language AWS Credentials: profile:default

## 코드 조각

코드 조각을 사용하면 짧은 코드 섹션을 삽입할 수 있습니다. 코드 조각 사용하기

1. 파일을 열고 JSON 형식의 `.asl.json` 확장자 또는 YAML 형식의 `.asl.yaml` 확장자로 저장합니다.
2. 상태(States) 속성을 사용하여 새 상태 머신을 생성합니다.
3. 상태(States) 내에 커서를 둡니다.
4. `Control + Space` 키 조합을 사용하여 선호하는 원하는 코드 조각을 선택합니다.
5. `Tab`을 사용하여 코드 조각에서 변수 및 파라미터를 트래버스합니다.
6. 관련 상태에 커서를 둔 다음 `Retry` 및 `Catch` 조각을 테스트합니다.



## 코드 완성 및 유효성 검사

### 코드 완성의 작동 방식 확인

1. 여러 상태를 생성합니다.
2. 다음 StartAt, 또는 기본 속성 뒤에 커서를 놓습니다.
3. Control + Space 키 조합을 사용하여 사용 가능한 완성을 나열합니다. Control + Space를 사용하여 추가 속성에 액세스할 수 있으며 State의 Type을 기반으로 합니다.
4. 작업할 때 다음에 대해 다음에 대한 코드 유효성 검사가 수행됩니다.
  - 누락된 속성
  - 잘못된 값
  - 터미널 상태 없음
  - 포인트가 지정되지만 존재하지 않는 상태

```

"FirstMatchState": {
  "Type": "Task",
  "Resource": "arn:aws:lambda:us-east-2:637554956784:function:Function",
  "Next": ""
},
"SecondMatchState": {
  "Type": "Task",
  "Resource": "arn:aws:lambda:us-east-2:637554956784:function:Function",
  "Next": ""
},
"DefaultState": {
  "Type": "Fail",
  "Error": "DefaultStateError",
  "Cause": "No Matches!"
},
"NextState": {
  "Type": "Task",
  "Resource": "arn:aws:lambda:us-east-2:637554956784:function:Function",
  "End": true
}
    
```

```

"FirstMatchState": {
  "Type": "Task",
  "Resource": "arn:aws:lambda:us-east-2:637554956784:function:Function",
  "Catch": [
    {
      "Error": "DefaultStateError",
      "Cause": "No Matches!"
    }
  ],
  "End": true
}
    
```

An array of objects, called Catchers, that define a fallback state. This state is executed if the state encounters runtime errors and its retry policy is exhausted or isn't defined.

## 위협 컴포저와 함께 작업하기

를 AWS Toolkit for Visual Studio Code 사용하여 위협 컴포저 도구를 사용할 수 있습니다. Threat Composer는 위협 모델링 프로세스를 단순화할 수 있는 위협 모델링 도구입니다.

[위협 컴포저 도구에 대한 자세한 내용은 위협 컴포저 리포지토리를 참조하십시오. GitHub](#)

다음 항목에서는 Threat Composer를 에서 사용하는 방법을 설명합니다. AWS Toolkit for Visual Studio Code

## 주제

- [툴킷에서 위협 컴포저로 작업하기](#)

## 툴킷에서 위협 컴포저로 작업하기

위협 컴포저를 사용하면 VS Code에서 직접 위협 컴포저 위협 모델을 만들고, 보고, 편집할 수 있습니다. [위협 컴포저 도구에 대한 자세한 내용은 위협 컴포저 리포지토리를 참조하십시오. GitHub](#)

다음 섹션에서는 에서 위협 컴포저 도구에 액세스하는 방법을 설명합니다. AWS Toolkit for Visual Studio Code

### 툴킷에서 위협 컴포저에 액세스

툴킷에서 위협 컴포저에 액세스할 수 있는 세 가지 주요 방법이 있습니다.

#### 기존 위협 모델을 통해 위협 컴포저에 액세스

위협 컴포저를 열려면 VS Code에서 기존 위협 모델 파일 (확장자 `.tc.json`) 을 여십시오. Threat Composer는 VS Code 편집기 창에서 위협 모델 파일을 자동으로 열고 시각화합니다.

#### 새 위협 컴포저 위협 모델 생성

1. VS Code 기본 메뉴에서 파일을 확장한 다음 새 파일을 선택합니다.
2. 새 파일 대화상자에서 Threat Composer 파일... 을 선택합니다. .
3. 메시지가 표시되면 a를 file name 입력한 다음 **enter** 키를 눌러 Threat Composer를 열고 새 VS Code 편집기 창에서 빈 위협 모델 파일을 시각화하십시오.

#### 명령 팔레트에서 새 위협 컴포저 위협 모델 생성

1. VS Code에서 **Cmd + Shift + P** 또는 **Ctrl + Shift + P** (Windows) 를 눌러 명령 팔레트를 엽니다.
2. 검색 필드에 **Threat Composer** 입력하고 결과에 새 위협 컴포저 파일 만들기가 표시되면 [새 위협 컴포저 파일 만들기] 를 선택합니다.
3. 메시지가 표시되면 a를 file name 입력한 다음 **enter** 키를 눌러 Threat Composer를 열고 새 VS Code 편집기 창에서 빈 위협 모델 파일을 시각화하십시오.

## 리소스 작업

AWS 탐색기에서 기본 AWS 서비스에 액세스할 뿐만 아니라 Resources에서도 수백 개의 리소스 중에서 선택하여 인터페이스에 추가할 수 있습니다. AWS에서 리소스는 작업할 수 있는 엔터티입니다. 추가할 수 있는 리소스는 Amazon AppFlow, Amazon Kinesis Data Streams, AWS IAM 역할, Amazon VPC 및 Amazon CloudFront입니다.

선택한 후 리소스로 이동하고 리소스 유형을 확장하여 해당 유형에 사용 가능한 리소스를 나열할 수 있습니다. 예를 들어 AWS Toolkit:Lambda::Function 리소스 유형을 선택하면 다양한 기능, 속성 및 특성을 정의하는 리소스에 액세스할 수 있습니다.

리소스 유형을 리소스에 추가한 후 다음과 같은 방법으로 해당 리소스와 상호 작용할 수 있습니다.

- 이 리소스 유형에 대해 현재 AWS 리전에서 사용할 수 있는 기존 리소스 목록을 확인합니다.
- 리소스를 설명하는 JSON 파일의 읽기 전용 버전을 확인합니다.
- 리소스의 리소스 식별자를 복사합니다.
- 리소스 모델링을 위한 리소스 유형 및 스키마(JSON 및 YAML 형식)의 목적을 설명하는 AWS 설명서를 확인합니다.
- 스키마에 맞는 JSON 형식의 템플릿을 편집하고 저장하여 새 리소스를 생성합니다.\*
- 기존 리소스를 업데이트하거나 삭제합니다.\*

### Important

\*AWS Toolkit for Visual Studio Code 현재 릴리스에서는 리소스 생성, 편집 및 삭제하는 옵션이 실험 기능으로 제공됩니다. 실험 기능은 계속 테스트되고 업데이트되므로 사용성 문제가 있을 수 있습니다. AWS Toolkit for Visual Studio Code에서 예고 없이 삭제될 수도 있습니다.

리소스에 실험 기능을 사용하려면 VS Code IDE의 설정에서 확장을 클릭하여 AWS도구 키트를 선택합니다.

리소스 생성, 업데이트 및 삭제하려면 AWS Toolkit Experiments에서 jsonResourceModification을 선택하세요.

자세한 내용은 [실험 기능 작업](#) 섹션을 참조하세요.

## 리소스 액세스를 위한 IAM 권한

AWS 서비스와 연결된 리소스에 액세스하려면 특정 AWS Identity and Access Management 권한이 필요합니다. 예를 들어 사용자 또는 역할과 같은 IAM 엔터티는 Lambda 권한이 있어야 AWS Toolkit:Lambda::Function 리소스에 액세스할 수 있습니다.

IAM 엔터티는 서비스 리소스에 대한 권한뿐만 아니라 VS Code용 도구 키트로 Cloud Control API 작업을 호출할 수 있는 권한도 있어야 합니다. Cloud Control API 작업을 통해 IAM 사용자 또는 역할이 원격 리소스에 액세스하고 업데이트할 수 있습니다.

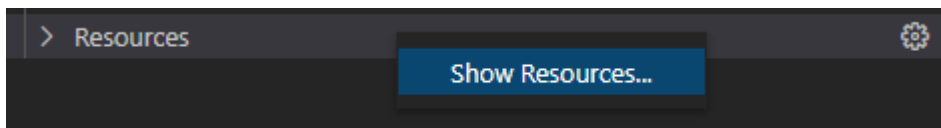
권한을 부여하는 가장 쉬운 방법은 AWS 관리형 정책, PowerUserAccess를 도구 키트 인터페이스를 사용하여 이러한 API 작업을 호출하는 IAM 엔터티에 연결하는 것입니다. 이 [관리형 정책](#)은 API 작업 호출을 포함하여 애플리케이션 개발 작업을 수행할 수 있는 다양한 권한을 부여합니다.

원격 리소스에서 허용 가능한 API 작업을 정의하는 특정 권한은 [AWS Cloud Control API 사용 설명서를 참조하세요](#).

## 기존 리소스 추가 및 상호 작용

1. AWS Explorer에서 리소스를 마우스 오른쪽 버튼으로 클릭하고 리소스 보기를 선택합니다.

이 창에는 사용가능한 리소스 유형 목록이 표시됩니다.



2. 선택 창에서 AWS Explorer에 추가할 리소스 유형을 선택하고 Return 키를 누르거나 OK를 선택합니다.

선택한 리소스 유형이 리소스 아래에 나열됩니다.

### Note

AWS Explorer에 리소스 유형을 추가했다면 해당 유형에 대한 확인란 선택을 취소하세요. OK를 선택한 후에는 해당 유형이 더 이상 리소스 아래에 나열되지 않습니다. 현재 선택된 리소스 유형만 AWS Explorer에 표시됩니다.

3. 리소스 유형에 속한 리소스를 보려면 해당 유형의 항목을 확장합니다.

사용 가능한 리소스 목록이 해당 리소스 유형 아래에 표시됩니다.

4. 특정 리소스와 상호 작용하려면 해당 이름을 마우스 오른쪽 버튼으로 클릭하고 다음 옵션 중 하나를 선택합니다.

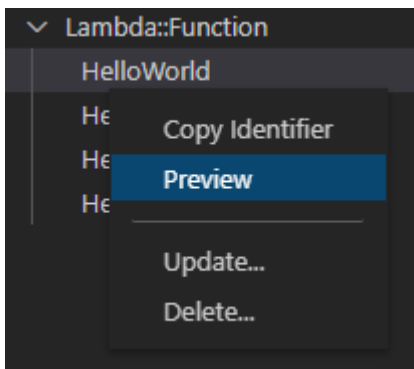
- 리소스 식별자 복사: 특정 리소스의 식별자를 클립보드에 복사합니다. (예를 들어 TableName 속성으로 AWS Toolkit:DynamoDB::Table 리소스를 식별할 수 있습니다.)
- 미리 보기: 리소스를 설명하는 JSON 형식 템플릿의 읽기 전용 버전을 확인합니다.

리소스 템플릿이 나타나면 편집기 탭 오른쪽에 있는 업데이트 아이콘을 선택하여 수정할 수 있습니다. 리소스를 업데이트하려면 필요한 [???](#) 항목을 활성화하세요.

- 업데이트: VS Code 편집기에서 리소스의 JSON 형식 템플릿을 편집합니다. 자세한 내용은 [리소스 생성 및 업데이트](#) 섹션을 참조하세요.
- 삭제: 표시된 대화 상자에서 삭제를 클릭하여 리소스를 삭제합니다. 리소스 삭제는 현재 이 버전의 AWS Toolkit for Visual Studio Code에서 [???](#)입니다.

#### Warning

리소스를 삭제하면 리소스를 사용하는 AWS CloudFormation 스택은 업데이트되지 않습니다. 이 업데이트 문제를 해결하려면 리소스를 다시 생성하거나 스택의 AWS CloudFormation 템플릿에서 리소스에 대한 참조를 제거해야 합니다. 자세한 내용은 이 [지식 센터 문서](#)를 참조하세요.





## 리소스 생성 및 업데이트

### ⚠ Important

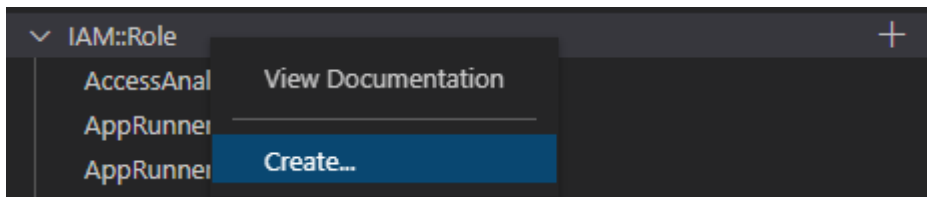
리소스 생성 및 업데이트는 현재 이 버전의 AWS Toolkit for Visual Studio Code에서 [???](#)입니다.

새 리소스를 생성하려면 리소스 목록에 리소스 유형을 추가한 다음 리소스, 속성 및 특성을 정의하는 JSON 형식의 템플릿을 편집하세요.

예를 들어, `AWS Toolkit:SageMaker::UserProfile` 리소스 유형에 속하는 리소스는 Amazon SageMaker Studio용 사용자 프로파일을 생성하는 템플릿으로 정의됩니다. 이 사용자 프로파일 리소스를 정의하는 템플릿은 `AWS Toolkit:SageMaker::UserProfile`의 리소스 유형 스키마에 맞아야 합니다. 예를 들어, 속성이 누락되거나 올바르지 않아 템플릿이 스키마를 준수하지 않는 경우 리소스를 생성하거나 업데이트할 수 없습니다.

1. 리소스를 마우스 오른쪽 버튼으로 클릭하고 리소스 보기를 선택하여 생성하려는 리소스의 유형을 추가합니다.
2. 리소스에서 리소스 유형을 추가한 후 더하기 (“+”) 아이콘을 선택하여 새 편집기에서 템플릿 파일을 엽니다.

또는 리소스 유형의 이름을 마우스 오른쪽 버튼으로 클릭하고 생성을 선택할 수 있습니다. 설명서 보기를 선택하면 리소스를 모델링하는 방법에 대한 정보에 액세스할 수 있습니다.



3. 편집기에서 리소스 템플릿을 구성하는 속성을 정의합니다. 자동 완성 기능은 템플릿의 스키마에 맞는 속성 이름을 제안합니다. 속성 유형에 마우스를 가져가면 속성 용도에 대한 설명이 창에 표시됩니다. 스키마에 대한 자세한 내용을 보려면 설명서 보기를 선택합니다.

리소스 스키마에 맞지 않는 텍스트는 물결 모양의 빨간색 밑줄로 표시됩니다.

```

1  {
2    "Role": "arn:aws:iam::488247187723:role/service-role/HelloWorld-role",
3    "":
4    " Code
5    " CodeSigningConfigArn
6    " DeadLetterConfig
7    " Environment
8    " ImageConfig
9    " KmsKeyArn
10   " Layers
11   } ReservedConcurrentExecutions
12   " Tags
13   " VpcConfig
14   "Handler": "index.handler",
15   "Arn": "arn:aws:lambda:us-west-2:488247187723:function:HelloWorld"
16 }

```

4. 리소스 선언 후 저장 아이콘을 선택하여 템플릿을 검증하고 원격 AWS 클라우드에 리소스를 저장합니다.

템플릿이 해당 스키마에 따라 리소스를 정의하는 경우 리소스가 생성되었음을 확인하는 메시지가 표시됩니다. (리소스가 이미 있는 경우 리소스가 업데이트되었음을 확인하는 메시지가 표시됩니다.)

생성된 리소스는 리소스 유형 이름에 추가됩니다.

5. 파일에 오류가 있는 경우 리소스를 생성하거나 업데이트할 수 없다는 메시지가 표시됩니다. 보기 로그를 열어 수정해야 할 템플릿 요소를 확인합니다.

# AWS Toolkit for Visual Studio Code의 보안

## 주제

- [AWS Toolkit for Visual Studio Code의 데이터 보호](#)

## AWS Toolkit for Visual Studio Code의 데이터 보호

AWS [공동 책임 모델](#)의 데이터에 AWS Toolkit for Visual Studio Code의 보안이 적용됩니다. 이 모델이 설명하는 것처럼 AWS는 모든 AWS 클라우드를 실행하는 글로벌 인프라를 보호할 책임이 있습니다. 사용자는 인프라에서 호스팅되는 콘텐츠를 관리해야 합니다. 이 콘텐츠에는 사용하는 AWS 서비스 서비스의 보안 구성과 관리 작업이 포함되어 있습니다. 데이터 프라이버시에 대한 자세한 내용은 [Data Privacy FAQ](#)(데이터 프라이버시 FAQ)를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS 보안 블로그의 [AWS Shared Responsibility Model and GDPR](#) 블로그 게시물을 참조하세요.

데이터를 보호하려면 AWS 계정 보안 인증 정보를 보호하고 AWS IAM Identity Center 또는 AWS Identity and Access Management(IAM)를 통해 개별 사용자 계정을 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 다중 인증(MFA)을 사용합니다.
- SSL/TLS를 사용하여 AWS 리소스와 통신합니다. TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- AWS CloudTrail로 API 및 사용자 활동 로깅을 설정합니다.
- AWS 암호화 솔루션을 AWS 서비스 내의 모든 기본 보안 컨트롤과 함께 사용하세요.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용하세요.
- 명령줄 인터페이스 또는 API를 통해 AWS에 액세스할 때 FIPS 140-2 인증 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용하세요. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 [Federal Information Processing Standard\(FIPS\) 140-2](#)를 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 Name 필드와 같은 자유 양식 필드에 입력하지 않는 것이 좋습니다. 이때 콘솔, API, AWS CLI 또는 AWS SDK를 사용하여 AWS Visual Studio Code 도구 키트 또는 기타 AWS 서비스를 사용하는 경우도 포함합니다. 이름에 사용되는 태그 또는 자유 형식 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버로 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 자격 증명 정보를 URL에 포함해서는 안 됩니다.

# AWS Toolkit for Visual Studio Code 사용 설명서 기록

기본 설명서 최신 업데이트: 2021년 10월 11일

다음 표는 AWS Toolkit for Visual Studio Code의 릴리스 별로 변경된 중요 사항을 안내합니다. 이 설명서에 대한 업데이트 알림을 받으려면 [RSS feed](#)를 구독하세요.

변경 사항	설명	날짜
<a href="#">생성된 사용 설명서: CloudFormation 템플릿 생성</a>	VS Code용 도구 키트를 사용한 CloudFormation 템플릿 생성 방법을 설명하는 새 사용 설명서 제작	2021년 12월 17일
<a href="#">간단한 UI 업데이트</a>	“Preview Machine State”라는 텍스트를 UI에 맞게 “Render graph”로 변경.	2021년 12월 14일
<a href="#">Amazon Elastic Container Service Exec 사용 설명서 작성</a>	다음은 Amazon ECS Exec의 개요입니다.	2021년 12월 13일
<a href="#">VS Code용 AWS IoT 도구 키트 사용 설명서 작성</a>	이 사용 설명서는 VS Code용 도구 키트 AWS IoT 서비스를 시작하는 데 도움을 주기 위해 작성했습니다.	2021년 11월 22일
<a href="#">실험 기능 지원</a>	AWS 서비스의 실험 기능을 활성화하는 지원이 추가되었습니다.	2021년 10월 14일
<a href="#">AWS 리소스 지원</a>	리소스 생성, 편집 및 삭제 옵션과 더불어 리소스 유형 액세스에 대한 지원이 추가되었습니다.	2021년 10월 14일
<a href="#">AWS Toolkit for Visual Studio Code IDE용 Amazon ECR 서비스 개요</a>	VS Code에서 액세스할 수 있는 Amazon ECR 서비스의 특	2021년 10월 14일

	징과 기능에 대한 개요 및 검토 회가 추가되었습니다.	
<a href="#">ARM64 환경 지원</a>	이제 ARM64 기반 에뮬레이션 환경뿐 아니라 x86_64 기반 환경에서도 서버리스 애플리케이션을 실행할 수 있습니다.	2021년 10월 1일
<a href="#">AWS 서버리스 애플리케이션</a>	ARM64 플랫폼에서 AWS SAM 애플리케이션을 실행할 수 있습니다.	2021년 9월 30일
<a href="#">포맷 업데이트 Node.js 섹션</a>	고객 피드백에 따라 Node.js/ TypeScript 포맷을 업데이트했습니다.	2021년 8월 12일
<a href="#">App Runner 지원</a>	AWS App Runner 지원을 AWS Toolkit for Visual Studio Code에 추가했습니다.	2021년 8월 11일
<a href="#">Go 함수 디버깅</a>	로컬 Go 함수 디버깅 지원이 추가되었습니다.	2021년 5월 10일
<a href="#">Java 함수 디버깅</a>	로컬 Java 함수 디버깅 지원이 추가되었습니다.	2021년 4월 22일
<a href="#">AWS Step Functions에 YAML 지원</a>	AWS Step Functions에 YAML 지원이 추가되었습니다.	2021년 3월 4일
<a href="#">Amazon API Gateway 리소스 디버깅</a>	로컬 Amazon API Gateway 리소스 디버깅 지원이 추가되었습니다.	2020년 12월 1일
<a href="#">Amazon API Gateway</a>	Amazon API Gateway 지원이 추가되었습니다.	2020년 12월 1일
<a href="#">AWS 서버리스 애플리케이션</a>	서버리스 애플리케이션에 Lambda 컨테이너 이미지 지원이 추가되었습니다.	2020년 12월 1일

<a href="#">AWS Systems Manager 지원</a>	Systems Manager Automation 설명서 지원이 추가되었습니다.	2020년 9월 30일
<a href="#">CloudWatch Logs</a>	CloudWatch Logs 지원이 추가되었습니다.	2020년 8월 24일
<a href="#">Amazon S3</a>	Amazon S3 지원이 추가되었습니다.	2020년 7월 30일
<a href="#">AWS Step Functions 지원</a>	AWS Step Functions 지원이 추가되었습니다.	2020년 3월 31일
<a href="#">Security Content</a>	보안 콘텐츠를 추가했습니다.	2020년 2월 6일
Amazon EventBridge 스키마 사용	Amazon EventBridge 스키마 지원이 추가되었습니다.	2019년 12월 1일
<a href="#">AWS CDK</a>	AWS CDK 릴리스 미리 보기	2019년 11월 25일
<a href="#">외부 자격 증명 프로세스 사용</a>	외부 자격 증명 프로세스를 사용하여 AWS 자격 증명을 얻는 방법에 대한 정보가 추가되었습니다.	2019년 9월 25일
<a href="#">작업 정의 파일의 IntelliSense 사용</a>	Amazon ECS 작업 정의 파일에 IntelliSense를 사용할 수 있습니다.	2019년 9월 24일
<a href="#">AWS Toolkit for Visual Studio Code 사용 설명서</a>	일반 공개용 릴리스입니다.	2019년 7월 11일
<a href="#">AWS Toolkit for Visual Studio Code 사용 설명서</a>	명확하고 사용하기 쉽도록 설명서 구성을 변경했습니다.	2019년 7월 3일
<a href="#">AWS Toolkit for Visual Studio Code 설치</a>	다양한 툴체인을 지원하기 위해 언어 SDK를 설치하는 방법에 대한 정보가 추가되었습니다.	2019년 6월 12일

[툴체인 구성](#)

다양한 툴체인 구성에 대한 정보가 추가되었습니다.

2019년 6월 12일

[최초 릴리스](#)

AWS Toolkit for Visual Studio Code 사용 설명서의 최초 릴리스입니다.

2019년 3월 28일

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.