사용자 가이드

AWS Cloud9

AWS Cloud9: 사용자 가이드

Table of Contents

AWS Cloud9란 무엇인가요?

AWS Cloud9은 통합 개발 환경, 즉 IDE입니다.

AWS Cloud9 IDE는 여러 프로그래밍 언어와 런타임 디버거 및 기본 제공 터미널을 갖춘 강력한 코드 편집 환경을 제공합니다. 소프트웨어를 코딩, 빌드, 실행, 테스트 및 디버깅하기 위한 도구 모음을 갖추고 있으며 클라우드에 소프트웨어를 릴리스하는 데에도 도움이 됩니다.

웹 브라우저를 통해 AWS Cloud9 IDE에 액세스합니다. IDE를 원하는 대로 구성할 수 있습니다. 색상 테마를 전환하고, 바로 가기 키를 결합하고, 프로그래밍 언어별 구문 색상 지정 및 코드 서식 지정 등을 사용할 수 있습니다.

(알았습니다! AWS Cloud9을 사용할 준비가 되었습니다.어떻게 시작할 수 있습니까?)

AWS Cloud9 작동 방식

다음은 AWS Cloud9의 작동 방식에 대한 종합적인 개요를 보여주는 다이어그램입니다.

다이어그램에서 보면(아래쪽부터), 로컬 컴퓨터의 웹 브라우저를 실행하는 AWS Cloud9 IDE를 사용하여 AWS Cloud9 환경과 상호 작용합니다. 컴퓨팅 리소스(예: Amazon EC2 인스턴스 또는 자체 서버)가 해당 환경에 연결됩니다. 끝으로, AWS CodeCommit 리포지토리 또는 다른 유형의 원격 리포지토리에 작업이 저장됩니다.

AWS Cloud9 환경

AWS Cloud9 환경은 프로젝트의 파일을 저장하고 도구를 실행하여 애플리케이션을 개발하는 곳입니다.

AWS Cloud9 IDE를 사용하면 다음을 수행할 수 있습니다.

- 인스턴스나 서버에 로컬로 프로젝트 파일을 저장합니다.
- 원격 코드 리포지토리(예: AWS CodeCommit의 리포지토리)를 환경에 복제합니다.
- 환경에서 로컬 파일과 복제 파일의 조합 작업.

각 환경이 특정 개발 프로젝트용으로 설정된 여러 환경을 생성하고 전환할 수 있습니다. 클라우드에 환경을 저장함으로써 프로젝트를 더 이상 단일 컴퓨터 또는 서버 설정에 제한시키지 않아도 됩니다. 이를 통해 컴퓨터 간에 손쉽게 전환하고 개발자를 팀에 더 빠르게 온보딩하는 등 여러 이점을 누릴 수 있습니다.

환경 및 컴퓨팅 리소스

드러나 있지는 않지만 컴퓨팅 리소스에 환경을 연결할 수 있는 몇 가지 방법이 있습니다.

- Amazon EC2 인스턴스를 생성한 후 새로 만든 EC2 인스턴스에 환경을 연결하도록 AWS Cloud9에 지시할 수 있습니다. 이 유형의 설정을 EC2 환경이라고 합니다.
- 환경을 기존 클라우드 컴퓨팅 인스턴스 또는 자체 서버에 연결하도록 AWS Cloud9에 지시할 수 있습니다. 이 유형의 설정을 SSH 환경이라고 합니다.

EC2 환경과 SSH 환경에는 몇 가지 유사점과 차이점이 있습니다. AWS Cloud9을 처음 사용하는 경우 AWS Cloud9이 여러 구성을 자동으로 관리하므로 EC2 환경을 사용하는 것이 좋습니다. AWS Cloud9을 자세히 배울 때 이 유사성과 차이를 보다 깊이 알아보려면 AWS Cloud9에서 EC2 환경과 SSH 환경 비교 단원을 참조하십시오.

AWS Cloud9 작동 방법에 대한 자세한 내용은 관련 동영상과 웹 페이지를 참조하십시오.

AWS Cloud9으로 할 수 있는 작업은 무엇입니까?

AWS Cloud9을 사용하여 여러 가지 흥미로운 시나리오에서 다양하게 소프트웨어를 코딩, 빌드, 실행, 테스트, 디버깅 및 릴리스할 수 있습니다. 다음과 같은 방법이 여기에 포함됩니다(이에 국한되지 않음).

- 여러 프로그래밍 언어 및 AWS Cloud Development Kit (AWS CDK)에서 코드 작업
- 실행 중인 도커 컨테이너에서 코드 작업
- 온라인 코드 리포지토리 사용
- 실시간으로 다른 사용자들과 협력
- 여러 데이터베이스 및 웹 사이트 기술과 상호 작용
- AWS Lambda, Amazon API Gateway 및 AWS 서버리스 애플리케이션을 대상으로 지정
- Amazon Lightsail, AWS CodeStar, AWS CodePipeline 등의 다른 AWS 제품 활용

자세한 내용은 AWS Cloud9으로 할 수 있는 작업은 무엇입니까? 단원을 참조하십시오.

어떻게 시작할 수 있습니까?

AWS Cloud9을 사용하려면 AWS Cloud9 설정의 단계를 수행한 다음 기본 자습서를 살펴보십시오.

추가 주제

- AWS Cloud9으로 할 수 있는 작업은 무엇입니까?
- AWS Cloud9에 대한 추가 정보

AWS Cloud9으로 할 수 있는 작업은 무엇입니까?

다음 리소스에서 일반적인 몇 가지 시나리오에서의 AWS Cloud9 사용에 대해 알아보십시오.

주요 시나리오

시나리오	리소스
AWS Lambda 함수와 서버리스 애플리케이션에서 AWS 도구 키트를 사용하여 코드를 생성, 실행 및 디버그합니다.	AWS 도구 키트를 사용한 AWS Lambda 함수 작업
WordPress, LAMP(Linux, Apache, MySQL, PHP), Node.js, Nginx, Drupal, Joomla 등 널리 사용되는 애플리케이션 및 프레임워크, 그리고 Amazon Linux, Ubuntu, Debian, FreeBSD, openSUSE 등의 Linux 배포판으로 사전 구성된 Amazon Lightsail 인스턴스로 작업합니다.	AWS Cloud9 통합 개발 환경(IDE)의 Amazon Lightsail 인스턴스
AWS CodeStar의 AWS 소프트웨어 개발 프로젝트 및 도구 체인에서 코드를 작업합니다.	AWS Cloud9 통합 개발 환경(IDE)의 AWS CodeStar 프로젝트로 작업
AWS CodePipeline의 지속적 배포 솔루션에서 코드를 작업합니다.	AWS Cloud9 통합 개발 환경(IDE)의 AWS CodePipeline 작업
AWS CLI 및 AWS CloudShell을 사용하여 AWS 서비스를 자동화합니다.	AWS Cloud9용 AWS Command Line Interface 및 aws-shell 자습서

시나리오	리소스
AWS CodeCommit에서 소스 코드 리포지토리를 작업합니다.	AWS Cloud9용 AWS CodeCommit 자습서
GitHub에서 Git 패널 인터페이스를 사용하여 소스 코드 리포지토리를 작업합니다.	Git 패널로 시각적 소스 제어
Amazon DynamoDB에서 NoSQL 데이터베이스를 작업합니다.	AWS Cloud9용 Amazon DynamoDB 자습서
LAMP(Linux, Apache HTTP Server, MySQL, PHP) 스택을 작업합니다.	AWS Cloud9용 LAMP 자습서
WordPress 웹 사이트를 작업합니다.	AWS Cloud9용 WordPress 자습서
Java 및 AWS SDK for Java에 대한 코드로 작업합니다.	AWS Cloud9용 Java 자습서
C++ 및 AWS SDK for C++에 대한 코드로 작업합니다.	AWS Cloud9용 C++ 자습서
Python 및 AWS SDK for Python (Boto)에 대한 코드로 작업합니다.	AWS Cloud9용 Python 자습서
.NET Core 및 AWS SDK for .NET에 대한 코드로 작업합니다.	AWS Cloud9용 .NET 자습서
Node.js 및 AWS SDK for JavaScript에 대한 코드로 작업합니다.	Node.js 튜토리얼 AWS Cloud9
PHP 및 AWS SDK for PHP에 대한 코드로 작업합니다.	에 대한 PHP 튜토리얼 AWS Cloud9
Ruby 및 AWS SDK for Ruby에 대한 코드로 작업합니다.	AWS Cloud9의 Ruby
Go 및 AWS SDK for Go에 대한 코드로 작업합니다.	AWS Cloud9용 Go 자습서

시나리오	리소스
TypeScript 및 AWS SDK for JavaScript에 대한 코드로 작업합니다.	AWS Cloud9용 TypeScript 자습서
AWS Cloud Development Kit (AWS CDK)용 코드로 작업합니다.	AWS Cloud9용 AWS CDK 자습서
실행 중인 도커 컨테이너에서 코드를 작업합니다.	에 대한 도커 튜토리얼 AWS Cloud9
나와 함께 환경을 사용하도록 텍스트 채팅 지원을 통해 실시간으로 다른 사람을 초대합니다.	AWS Cloud9의 공유 환경 작업
AWS RoboMaker에서 지능형 로봇 애플리케이션용 코드로 작업합니다.	AWS RoboMaker 개발자 가이드의 AWS Cloud9으로 개발

AWS Cloud9에 대한 추가 정보

이 항목에서는 AWS Cloud9에 대해 알아보는 데 도움이 되는 자세한 정보를 제공합니다.

주제

- 관련 비디오
- AWS 사이트의 관련 항목
- 요금
- 추가 질문이 있거나 도움이 필요합니다.

관련 비디오

- AWS re:Invent 2017 - Introducing AWS Cloud9: Werner Vogels Keynote(9분, YouTube 웹 사이트)
- AWS re:Invent Launchpad 2017 - AWS Cloud9, (15분, YouTube 웹 사이트)
- Introducing AWS Cloud9 - AWS Online Tech Talks(33분, YouTube 웹 사이트)
- AWS Sydney Summit 2018: AWS Cloud9 and AWS CodeStar(25분, YouTube 웹 사이트)

AWS 사이트의 관련 항목

- AWS Cloud9 소개
- AWS Cloud9 – 클라우드 개발자 환경
- AWS Cloud9 개요
- AWS Cloud9 기능
- AWS Cloud9 FAQ

요금

AWS Cloud9에 대한 추가 요금은 없습니다. AWS Cloud9 개발 환경에 Amazon EC2 인스턴스를 사용하는 경우 코드를 실행 및 저장하는 데 사용되는 컴퓨팅 및 스토리지 리소스(예: Amazon EC2 인스턴스, Amazon EBS 볼륨)에 대해서만 요금을 지불합니다. 또한 추가 비용 없이 SSH를 통해 환경을 기존 Linux 서버(예: 온프레미스 서버)에 연결할 수도 있습니다.

사용한 만큼만 비용을 지불하고 최소 요금 및 사전 약정은 없습니다. 환경 내에서 생성 또는 사용하는 AWS 리소스(예: AWS Lambda 함수)에 대해 일반 AWS 요금이 청구됩니다.

AWS 프리 티어 이용 자격이 있는 AWS 신규 고객은 AWS Cloud9을 무료로 사용할 수 있습니다. AWS에서 프리 티어를 초과하는 리소스를 활용하는 경우, 그러한 리소스에 대해 일반 AWS 요금이 청구됩니다.

자세한 내용은 다음을 참조하세요.

- AWS Cloud9 요금: AWS Cloud9 요금 단원을 참조하십시오.
- AWS 서비스 요금: Amazon EC2 요금, Amazon EBS 요금, AWS Lambda 요금 및 AWS 요금을 참조하세요.
- AWS 프리 티어: AWS Billing and Cost Management 사용 설명서에서 AWS 프리 티어 사용 및 프리 티어 사용량 추적을 참조하세요.
- 교육 요금: AWS 교육 프로그램을 참조하십시오.

추가 질문이 있거나 도움이 필요합니다.

AWS Cloud9 커뮤니티에 질문하거나 도움을 구하려면 AWS Cloud9 Discussion Forum을 참조하십시오. (이 포럼에 들어갈 때 AWS에서 로그인을 요청할 수 있습니다.)

또한 FAQ를 참조하거나 직접 당사에 문의하십시오.

AWS Cloud9 설정

AWS Cloud9 사용을 시작하려면 AWS Cloud9을 사용하려고 하는 방법에 따라 다음 절차 중 하나를 따르십시오.

사용 패턴	따라야 할 절차
내 AWS 계정을 사용하는 유일한 개인이지만 학생은 아님	개별 사용자 설정
단일 AWS 계정에 사용자가 여러 명 있는 팀에 속해 있음	팀 설정
단일 조직 내 AWS 계정이 하나 이상 있는 기업에 속해 있음	엔터프라이즈 설정

AWS Cloud9에 대한 일반 정보는 AWS Cloud9란 무엇입니까?를 참조하세요.

주제

- 개별 사용자 설정 AWS Cloud9
- 팀 설정 대상 AWS Cloud9
- AWS Cloud9에 대한 엔터프라이즈 설정
- AWS Cloud9에 대한 추가 설정 옵션(팀 및 엔터프라이즈)

개별 사용자 설정 AWS Cloud9

이 항목에서는 학생이 아닐 AWS 계정 때 유일한 AWS Cloud9 사용자로 설정하고 사용하는 방법을 설명합니다. 다른 AWS Cloud9 사용 패턴으로 설정할 수 있습니다. 지침은 AWS Cloud9 설정을 참조하세요.

사용자 중 유일한 AWS Cloud9 사용자로 사용하려면 아직 계정이 없는 AWS 계정 AWS 계정 경우 가입하세요. 다음으로 AWS Cloud9 콘솔에 로그인합니다.

주제

- 등록하세요. AWS 계정

- 관리자 액세스 권한이 있는 사용자 생성
- 다른 인증 방법
- 다음 단계

등록하세요. AWS 계정

계정이 없는 경우 다음 단계를 완료하여 계정을 만드세요. AWS 계정

가입하려면 AWS 계정

1. https://portal.aws.amazon.com/billing/signup을 여세요.
2. 온라인 지시 사항을 따르세요.

   등록 절차 중에는 전화를 받고 키패드로 인증 코드를 입력하는 과정이 있습니다.

   에 AWS 계정가입하면 AWS 계정 루트 사용자a가 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스 액세스 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 루트 사용자 액세스 권한이 필요한 작업을 수행하는 것입니다.

AWS 가입 절차가 완료된 후 확인 이메일을 보냅니다. 언제든지 https://aws.amazon.com/으로 가서 내 계정(My Account)을 선택하여 현재 계정 활동을 보고 계정을 관리할 수 있습니다.

관리자 액세스 권한이 있는 사용자 생성

등록한 AWS 계정후에는 일상적인 작업에 루트 사용자를 사용하지 않도록 관리 사용자를 보호하고 AWS IAM Identity Center활성화하고 생성하십시오 AWS 계정 루트 사용자.

보안을 유지하세요. AWS 계정 루트 사용자

1. 루트 사용자를 선택하고 AWS 계정 이메일 주소를 입력하여 계정 AWS Management Console소유자로 로그인합니다. 다음 페이지에서 비밀번호를 입력합니다.

   루트 사용자를 사용하여 로그인하는 데 도움이 필요하면AWS 로그인 사용 설명서의 루트 사용자로 로그인을 참조하세요.
2. 루트 사용자의 다중 인증(MFA)을 활성화합니다.

지침은 IAM 사용 설명서의 AWS 계정 루트 사용자 (콘솔) 에 대한 가상 MFA 디바이스 활성화를 참조하십시오.

관리자 액세스 권한이 있는 사용자 생성

1. IAM Identity Center를 활성화합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 AWS IAM Identity Center설정을 참조하세요.
2. IAM Identity Center에서 사용자에게 관리 액세스 권한을 부여합니다.

   를 ID 소스로 사용하는 방법에 대한 자습서는 사용 설명서의 기본값으로 IAM Identity Center 디렉터리사용자 액세스 구성을 참조하십시오. IAM Identity Center 디렉터리 AWS IAM Identity Center

관리 액세스 권한이 있는 사용자로 로그인

- IAM IDentity Center 사용자로 로그인하려면 IAM IDentity Center 사용자를 생성할 때 이메일 주소로 전송된 로그인 URL을 사용합니다.

  IAM Identity Center 사용자를 사용하여 로그인하는 데 도움이 필요하면 사용 설명서의 AWS 액세스 포털에 로그인을 참조하십시오.AWS 로그인

추가 사용자에게 액세스 권한 할당

1. IAM Identity Center에서 최소 권한 적용 모범 사례를 따르는 권한 세트를 생성합니다.

   지침은AWS IAM Identity Center 사용 설명서의 Create a permission set를 참조하세요.
2. 사용자를 그룹에 할당하고, 그룹에 Single Sign-On 액세스 권한을 할당합니다.

   지침은AWS IAM Identity Center 사용 설명서의 Add groups를 참조하세요.

다른 인증 방법

> **Warning**
>
> 보안 위험을 방지하려면 목적별 소프트웨어를 개발하거나 실제 데이터로 작업할 때 IAM 사용자를 인증에 사용하지 마십시오. 대신 AWS IAM Identity Center과 같은 보안 인증 공급자를 통한 페더레이션을 사용하십시오.

전체 액세스 관리 AWS 계정

보안 모범 사례로서, IAM Identity Center를 AWS Organizations 사용하여 모든 사용자의 AWS 계정액세스를 관리하는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 IAM의 보안 모범 사례를 참조하십시오.

IAM ID 센터에서 사용자를 만들거나, Microsoft Active Directory를 사용하거나, SAML 2.0 ID 공급자 (IdP) 를 사용하거나, IdP를 개별적으로 페더레이션할 수 있습니다. AWS 계정이러한 접근 방식 중 하나를 사용하면 사용자에게 Single Sign-On 경험을 제공할 수 있습니다. 또한 멀티 팩터 인증 (MFA) 을 적용하고 액세스를 위한 임시 자격 증명을 사용할 수 있습니다. AWS 계정 이는 공유할 수 있는 장기 보안 인증 정보이며 AWS 리소스에 대한 보안 위험을 증가시킬 수 있는 IAM 사용자와는 다릅니다.

샌드박스 환경에서만 사용할 IAM 사용자 생성

처음 사용하는 경우 테스트 IAM 사용자를 만든 다음 이를 사용하여 자습서를 실행하고 제공되는 기능을 탐색할 수 있습니다. AWS AWS 학습 중에는 이러한 유형의 보안 인증 정보를 사용해도 괜찮지만 샌드박스 환경 밖에서는 사용하지 않는 것이 좋습니다.

다음과 같은 사용 사례에서는 IAM 사용자와 함께 시작하는 것이 합리적일 수 있습니다. AWS

- AWS SDK 또는 도구를 시작하고 샌드박스 AWS 서비스 환경에서 탐색하기.
- 사람이 직접 진행하는 로그인 프로세스를 지원하지 않는 예약된 스크립트, 작업 및 기타 자동화된 프로세스를 학습의 일부로 실행하세요.

이러한 사용 사례 이외의 IAM 사용자를 사용하는 경우 가능한 한 빨리 IAM Identity Center로 전환하거나 ID 공급자를 페더레이션하십시오. AWS 계정 자세한 내용은 AWS에서 자격 증명 공급자 및 페더레이션을 참조하세요.

보안 IAM 사용자 액세스 키

IAM 사용자 액세스 키는 정기적으로 교체해야 합니다. IAM 사용자 설명서의 액세스 키 교체에 있는 지침을 따르세요. 실수로 IAM 사용자 액세스 키를 공유했다고 생각되면 액세스 키를 교체하세요.

IAM 사용자 액세스 키는 로컬 시스템의 공유 AWS `credentials` 파일에 저장해야 합니다. 코드에 IAM 사용자 액세스 키를 저장하지 마세요. IAM 사용자 액세스 키가 포함된 구성 파일을 소스 코드 관리 소프트웨어에 포함시키지 마세요. 오픈 소스 프로젝트 git-secrets와 같은 외부 도구를 사용하면 중요한 정보를 실수로 Git 리포지토리에 커밋하는 것을 방지할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 IAM 자격 증명(사용자, 사용자 그룹 및 역할)을 참조하세요.

다음 단계

학습을 위한 작업	주제
AWS Cloud9 IDE 사용 방법을 알아보십시오.	시작하기: 기본 자습서 및 IDE 작업

추가 고급 작업	주제
AWS Cloud9 개발 환경을 만든 다음 AWS Cloud9 IDE를 사용하여 새 환경에서 코드 작업을 수행하십시오.	환경 생성
나와 함께 다른 사람이 텍스트 채팅 지원을 통해 실시간으로 새 환경을 사용할 수 있도록 초대합니다.	공유 환경 사용

팀 설정 대상 AWS Cloud9

이 항목에서는 한 AWS 계정 명 내에 있는 여러 사용자가 사용할 수 있도록 AWS IAM Identity Center하는 방법을 설명합니다 AWS Cloud9. 다른 사용 AWS Cloud9 패턴에 사용하도록 설정하려면 올바른 지침을 AWS Cloud9 설정 참조하십시오.

다음 지침에서는 사용자에게 단일 AWS 계정에 대한 관리 권한이 이미 있거나 생길 것이라고 가정합니다. 자세한 내용은 IAM 사용 설명서의 AWS 계정 루트 사용자 및 첫 번째 관리자 및 그룹 생성을 참조

하십시오. 이미 계정이 AWS 계정 있지만 계정에 대한 AWS 계정 관리자 액세스 권한이 없는 경우 관리자에게 문의하십시오.

> **Warning**
>
> 보안 위험을 방지하려면 목적별 소프트웨어를 개발하거나 실제 데이터로 작업할 때 IAM 사용자를 인증에 사용하지 마십시오. 대신 AWS IAM Identity Center과 같은 보안 인증 공급자를 통한 페더레이션을 사용하십시오.

> **Note**
>
> IAM 대신 IAM Identity Center를 사용하여 단일 사용자 내의 여러 사용자가 사용할 수 있도록 AWS 계정 할 수 있습니다. AWS Cloud9이 사용 패턴에서는 단일 계정이 조직의 관리 계정 AWS 계정 역할을 합니다. AWS Organizations또한 해당 조직에는 멤버 계정이 없습니다. IAM Identity Center를 사용하려면 이 주제를 건너뛴 다음 대신 엔터프라이즈 설정의 지침을 따르세요. 관련 내용은 다음 리소스를 참조하십시오.
>
> - 사용 AWS Organizations 설명서의 AWS 조직이란 무엇입니까 (IAM ID 센터에서는 다음을 사용해야 함) AWS Organizations
> - AWS IAM Identity Center User Guide의 What is AWS IAM Identity Center
> - 4분짜리 동영상 AWS 지식 센터 동영상: on을 시작하려면 어떻게 해야 하나요? AWS Organizations YouTube
> - 7분짜리 동영상: IAM Identity Center를 사용하여 여러 AWS 계정에 대한 사용자 액세스를 관리합니다. YouTube
> - 9분짜리 동영상: 온프레미스 Active Directory 사용자를 위해 IAM ID 센터를 설정하는 방법은 다음과 같습니다. YouTube

한 번에 여러 사용자가 사용을 AWS Cloud9시작할 수 있게 AWS 계정 하려면 보유하고 있는 리소스에 대한 단계를 시작하십시오. AWS

AWS 계정이 있으신가요?	해당 계정에 IAM 그룹과 사용자가 하나 이상 있습니까?	이 단계부터 시작하십시오.
아니요	—	1단계: 계정 등록하기 AWS 계정

AWS 계정이 있으신가요?	해당 계정에 IAM 그룹과 사용자가 하나 이상 있습니까?	이 단계부터 시작하십시오.
예	아니요	2단계: IAM 그룹 및 사용자를 생성하고, 그룹에 사용자 추가
예	예	3단계: 그룹에 AWS Cloud9 액세스 권한 추가

주제

등록하세요. AWS 계정

계정이 없는 경우 다음 단계를 완료하여 계정을 만드세요. AWS 계정

가입하려면 AWS 계정

1. https://portal.aws.amazon.com/billing/signup을 여세요.
2. 온라인 지시 사항을 따르세요.

   등록 절차 중에는 전화를 받고 키패드로 인증 코드를 입력하는 과정이 있습니다.

   에 AWS 계정가입하면 AWS 계정 루트 사용자a가 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스 액세스 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 루트 사용자 액세스 권한이 필요한 작업을 수행하는 것입니다.

AWS 가입 절차가 완료된 후 확인 이메일을 보냅니다. 언제든지 https://aws.amazon.com/으로 가서 내 계정(My Account)을 선택하여 현재 계정 활동을 보고 계정을 관리할 수 있습니다.

관리자 액세스 권한이 있는 사용자 생성

등록한 AWS 계정후에는 일상적인 작업에 루트 사용자를 사용하지 않도록 관리 사용자를 보호하고 AWS IAM Identity Center활성화하고 생성하십시오 AWS 계정 루트 사용자.

보안을 유지하세요. AWS 계정 루트 사용자

1. 루트 사용자를 선택하고 AWS 계정 이메일 주소를 입력하여 계정 AWS Management Console소유자로 로그인합니다. 다음 페이지에서 비밀번호를 입력합니다.

   루트 사용자를 사용하여 로그인하는 데 도움이 필요하면AWS 로그인 사용 설명서의 루트 사용자로 로그인을 참조하세요.
2. 루트 사용자의 다중 인증(MFA)을 활성화합니다.

   지침은 IAM 사용 설명서의 AWS 계정 루트 사용자 (콘솔) 에 대한 가상 MFA 디바이스 활성화를 참조하십시오.

관리자 액세스 권한이 있는 사용자 생성

1. IAM Identity Center를 활성화합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 AWS IAM Identity Center설정을 참조하세요.
2. IAM Identity Center에서 사용자에게 관리 액세스 권한을 부여합니다.

   를 ID 소스로 사용하는 방법에 대한 자습서는 사용 설명서의 기본값으로 IAM Identity Center 디렉터리사용자 액세스 구성을 참조하십시오. IAM Identity Center 디렉터리 AWS IAM Identity Center

관리 액세스 권한이 있는 사용자로 로그인

- IAM IDentity Center 사용자로 로그인하려면 IAM IDentity Center 사용자를 생성할 때 이메일 주소로 전송된 로그인 URL을 사용합니다.

  IAM Identity Center 사용자를 사용하여 로그인하는 데 도움이 필요하면 사용 설명서의 AWS 액세스 포털에 로그인을 참조하십시오.AWS 로그인

추가 사용자에게 액세스 권한 할당

1. IAM Identity Center에서 최소 권한 적용 모범 사례를 따르는 권한 세트를 생성합니다.

지침은AWS IAM Identity Center 사용 설명서의 Create a permission set를 참조하세요.

2. 사용자를 그룹에 할당하고, 그룹에 Single Sign-On 액세스 권한을 할당합니다.

   지침은AWS IAM Identity Center 사용 설명서의 Add groups를 참조하세요.

2단계: IAM 그룹 및 사용자를 생성하고, 그룹에 사용자 추가

이 단계에서는 AWS Identity and Access Management (IAM) 에서 그룹과 사용자를 생성하고, 사용자를 그룹에 추가한 다음, 해당 사용자를 사용하여 액세스합니다. AWS Cloud9이는 AWS 보안 모범 사례입니다. 자세한 설명은 IAM 사용자 가이드의 IAM 모범 사례 섹션을 참조하십시오.

필요한 모든 IAM 그룹과 사용자가 이미 있다면 3단계: 그룹에 AWS Cloud9 액세스 권한 추가로 건너뛰십시오.

> **Note**
>
> 귀하의 조직에 귀하를 위해 설정된 IAM 그룹 및 사용자가 이미 있을 수도 있습니다. 조직에 AWS 계정 관리자가 있는 경우 다음 절차를 시작하기 전에 해당 담당자에게 문의하세요.

이러한 태스크는 AWS Management Console 또는 AWS 명령줄 인터페이스(AWS CLI)를 사용하여 완료할 수 있습니다.

다음 콘솔 절차와 관련된 9분짜리 동영상을 보려면 IAM 사용자를 설정하고 IAM 자격 증명을 AWS Management Console 사용하여 로그인하려면 어떻게 해야 하나요? 를 참조하십시오. YouTube

2.1단계: 콘솔을 사용하여 IAM 그룹 생성

1. 아직 로그인하지 않은 AWS Management Console경우 https://console.aws.amazon.com/codecommit 에서 로그인하십시오.

   > **Note**
   >
   > 를 만들 때 제공한 이메일 주소와 AWS Management Console AWS 계정 암호로 에 로그인할 수 있습니다. 이를 일컬어 '루트 사용자로 서명'이라고 합니다. 하지만 이는 AWS 보안 모범 사례는 아닙니다. 앞으로는 AWS 계정의 관리자 사용자 자격 증명을 사용하여 로그인하는 것이 좋습니다. 관리자 사용자는 AWS 계정 루트 사용자와 비슷한 AWS 액세스 권한을 가지므로 일부 관련 보안 위험을 피할 수 있습니다. 관리자 사용자로 로그인할 수 없는 경우

관리자에게 AWS 계정 문의하세요. 자세한 내용은 IAM 사용 설명서에서 첫 번째 IAM 사용자 및 그룹 만들기를 참조하세요.

2. IAM 콘솔(IAM console)을 엽니다. 이렇게 하려면 AWS 탐색 표시줄에서 서비스를 선택합니다. 그런 다음 IAM을 선택합니다.
3. IAM 콘솔의 탐색 창에서[그룹(Groups)]을 선택합니다.
4. 새 그룹 생성을 선택합니다.
5. Set Group Name(그룹 이름 설정) 페이지의 Group Name(그룹 이름)에 새 그룹의 이름을 입력합니다.
6. 다음 단계를 선택합니다.
7. 정책 연결 페이지에서 정책을 연결하지 않고 다음 단계를 선택합니다. 3단계: 그룹에 AWS Cloud9 액세스 권한 추가에서 정책을 연결합니다.
8. 그룹 생성을 선택합니다.

> **Note**
>
> 이 절차를 반복하여 적어도 두 개의 그룹 (AWS Cloud9 사용자용 그룹과 AWS Cloud9 관리자용 그룹 하나) 을 생성하는 것이 좋습니다. 이 AWS 보안 모범 사례는 AWS 리소스 액세스 관련 문제를 더 효과적으로 제어, 추적 및 해결하는 데 도움이 될 수 있습니다.

2.2단계: 콘솔을 사용하여 IAM 사용자를 생성하고 이 사용자를 그룹에 추가로 건너뜁니다.

2.1단계: 다음을 사용하여 IAM 그룹 생성 AWS CLI

> **Note**
>
> AWS 관리형 임시 자격 증명을 사용하는 경우 AWS Cloud9 IDE의 터미널 세션을 사용하여 이 섹션의 일부 또는 모든 명령을 실행할 수 없습니다. AWS 보안 모범 사례를 해결하기 위해 AWS 관리형 임시 자격 증명에서는 일부 명령을 실행할 수 없습니다. 대신 AWS Command Line Interface (AWS CLI) 를 별도로 설치하여 해당 명령을 실행할 수 있습니다.

1. 컴퓨터에 설치하고 구성하십시오 (아직 설치하지 않았다면). AWS CLI 이를 위해 AWS Command Line Interface 사용 설명서에서 다음 지침을 참조하세요.
   - AWS 명령줄 인터페이스 설치

- 빠른 구성

> **Note**
>
> 생성 시 제공한 전자 메일 주소 및 암호와 관련된 자격 증명을 AWS CLI 사용하여 구성할 수 있습니다. AWS 계정 이를 일컬어 '루트 사용자로 서명'이라고 합니다. 하지만 이는 AWS 보안 모범 사례는 아닙니다. 대신 AWS 계정의 IAM 관리자 사용자에 대한 자격 증명을 AWS CLI 사용하도록 구성하는 것이 좋습니다. IAM 관리자 사용자는 AWS 계정 루트 사용자와 비슷한 AWS 액세스 권한을 가지므로 일부 관련 보안 위험을 피할 수 있습니다. 를 IAM 관리자 AWS CLI 사용자로 구성할 수 없는 경우 관리자에게 문의하십시오. AWS 계정 자세한 내용은 IAM 사용 설명서에서 첫 번째 IAM 관리자 및 그룹 만들기를 참조하세요.

2. 새 그룹의 이름(예: `MyCloud9Group`)을 지정하여 IAM `create-group` 명령을 실행합니다.

```
aws iam create-group --group-name MyCloud9Group
```

> **Note**
>
> 이 절차를 반복하여 적어도 두 개의 그룹을 생성하는 것이 좋습니다. 하나는 AWS Cloud9 사용자용 그룹이고 다른 하나는 AWS Cloud9 관리자용 그룹입니다. 이 AWS 보안 모범 사례는 AWS 리소스 액세스 관련 문제를 더 효과적으로 제어, 추적 및 해결하는 데 도움이 될 수 있습니다.

2.2단계로 건너뛰십시오. IAM 사용자를 생성하고 AWS CLI를 사용하여 그룹에 사용자를 추가합니다.

2.2단계: 콘솔을 사용하여 IAM 사용자를 생성하고 이 사용자를 그룹에 추가

1. 이전 절차에서 IAM 콘솔을 연 상태로 탐색 창에서[사용자(Users)]를 선택합니다.
2. 사용자 추가를 선택합니다.
3. 사용자 이름에 새 사용자의 이름을 입력합니다.

> **Note**
>
> 다른 사용자 추가를 선택하여 동시에 여러 사용자를 만들 수 있습니다. 이 절차의 다른 설정이 이러한 각 새 사용자에게 적용됩니다.

4. Programmatic access(프로그래밍 방식 액세스) 및 AWS Management Console access(액세스) 확인란을 선택합니다. 그러면 새 사용자가 다양한 AWS 개발자 도구 및 서비스 콘솔을 사용할 수 있습니다.
5. 자동 생성된 비밀번호의 기본 선택 항목을 그대로 둡니다. 그러면 새 사용자가 콘솔에 로그인할 수 있는 암호가 임의로 생성됩니다. 또는 사용자 지정 비밀번호를 선택하고 새 사용자의 특정 암호를 입력합니다.
6. 비밀번호 재설정 필요의 기본 선택 항목을 그대로 둡니다. 그러면 새 사용자가 콘솔에 처음 로그인한 후 암호를 변경하라는 메시지가 표시됩니다.
7. 다음: 권한을 선택합니다.
8. 기본 설정인 그룹에 사용자 추가를 그대로 둡니다. 또는 여러 사용자에 대해 그룹에 사용자 추가를 선택합니다.
9. 그룹 목록에서 사용자를 추가하려는 그룹 옆의 확인란(이름 아님)을 선택합니다.
10. 다음: 검토를 선택합니다.
11. 사용자 생성을 선택합니다. 또는 다중 사용자를 위한 사용자를 생성합니다.
12. 마법사의 마지막 페이지에서 다음 중 하나를 수행합니다.
    - 각 새 사용자 옆의 이메일 전송을 선택하고, 화면 지침에 따라 새 사용자에게 콘솔 로그인 URL 및 사용자 이름을 이메일로 보냅니다. 그런 다음 각 신규 사용자에게 콘솔 로그인 암호, AWS 액세스 키 ID, AWS 보안 액세스 키를 개별적으로 전달하십시오.
    - .csv 다운로드를 선택합니다. 그런 다음 각 새 사용자에게 콘솔 로그인 URL, 콘솔 로그인 암호, AWS 액세스 키 ID, 다운로드한 파일에 있는 AWS 보안 액세스 키를 알려주십시오.
    - 각 새 사용자 옆에 있는 [비밀 액세스 키(Secret access key)] 및 [암호(Password)]에 대해 [표시(Show)]를 선택합니다. 그런 다음 각 새 사용자에게 콘솔 로그인 URL, 콘솔 로그인 암호, AWS 액세스 키 ID 및 AWS 비밀 액세스 키를 전달합니다.

> **Note**
>
> .csv 다운로드를 선택하지 않은 경우 새 사용자의 AWS 보안 액세스 키와 콘솔 로그인 암호를 볼 수 있는 유일한 시간입니다. 새 사용자를 위한 새 AWS 보안 액세스 키 또는 콘솔 로그인 암호를 생성하려면 IAM 사용 설명서의 다음 내용을 참조하십시오.
>
> - 액세스 키 생성, 수정 및 확인(콘솔)
> - IAM 사용자 암호 생성, 변경 또는 삭제(콘솔)

13. 생성하려는 각각의 추가 IAM 사용자에 대해 이 절차를 반복해서 수행한 다음 3단계: 그룹에 AWS Cloud9 액세스 권한 추가로 이동합니다.

2.2단계: IAM 사용자를 생성하고 다음을 사용하여 그룹에 사용자를 추가합니다. AWS CLI

> **Note**
>
> AWS 관리형 임시 자격 증명을 사용하는 경우 AWS Cloud9 IDE의 터미널 세션을 사용하여 이 섹션의 일부 또는 모든 명령을 실행할 수 없습니다. AWS 보안 모범 사례를 해결하기 위해 AWS 관리형 임시 자격 증명에서는 일부 명령을 실행할 수 없습니다. 대신 AWS Command Line Interface (AWS CLI) 를 별도로 설치하여 해당 명령을 실행할 수 있습니다.

1. 새 사용자의 이름(예: `MyCloud9User`)을 지정하고 IAM `create-user` 명령을 실행하여 사용자를 생성합니다.

   ```
   aws iam create-user --user-name MyCloud9User
   ```

2. 사용자의 이름과 초기 로그인 암호(예: `MyC10ud9Us3r!`)를 지정하고 IAM `create-login-profile` 명령을 실행하여 사용자의 새 콘솔 로그인 암호를 생성합니다. 사용자가 로그인하면 AWS가 해당 사용자에게 로그인 암호를 변경하라고 요청합니다.

   ```
   aws iam create-login-profile --user-name MyCloud9User --password MyC10ud9Us3r! --
   password-reset-required
   ```

   나중에 사용자를 위한 대체 콘솔 로그인 암호를 생성해야 하는 경우 IAM 사용 설명서의 IAM 사용자 암호 (API, CLI, PowerShell) 생성, 변경 또는 삭제를 참조하십시오.

3. IAM `create-access-key` 명령을 실행하여 사용자를 위한 새 AWS 액세스 키와 해당 AWS 비밀 액세스 키를 생성합니다.

   ```
   aws iam create-access-key --user-name MyCloud9User
   ```

   표시되는 `AccessKeyId` 및 `SecretAccessKey` 값을 적어 둡니다. IAM `create-access-key` 명령을 실행한 후 이 시간에만 사용자의 AWS 보안 액세스 키를 볼 수 있습니다. 나중에 사용자를 위한 새 AWS 보안 액세스 키를 생성해야 하는 경우 IAM 사용 설명서의 액세스 키 (API, CLI,) 생성 PowerShell, 수정 및 보기를 참조하십시오.

4. 그룹 및 사용자의 이름을 지정하고 IAM `add-user-to-group` 명령을 실행하여 사용자를 그룹에 추가합니다.

```
aws iam add-user-to-group --group-name MyCloud9Group --user-name MyCloud9User
```

5. 사용자에게 콘솔 로그인 URL, 초기 콘솔 로그인 암호, AWS 액세스 키 ID 및 비밀 액세스 키를 알려 주십시오. AWS
6. 생성하려는 각 추가 IAM 사용자에 대해 이 절차를 반복해서 수행합니다.

3단계: 그룹에 AWS Cloud9 액세스 권한 추가

기본적으로 대부분의 IAM 그룹과 사용자는 다음을 포함하여 어떤 것에도 AWS 서비스액세스할 수 없습니다 (단 AWS Cloud9, AWS 계정 기본적으로 모든 AWS 서비스 그룹에 액세스할 수 있는 IAM 관리자 그룹과 IAM 관리자 사용자는 예외). 이 단계에서는 IAM을 사용하여 한 명 이상의 사용자가 속한 IAM 그룹에 AWS Cloud9 액세스 권한을 직접 추가합니다. 이렇게 하면 해당 사용자가 AWS Cloud9에 액세스할 수 있습니다.

> **Note**
>
> 귀하의 조직에 귀하를 위해 설정된 적절한 액세스 권한을 보유한 그룹이 이미 있을 수도 있습니다. 조직에 AWS 계정 관리자가 있는 경우 다음 절차를 시작하기 전에 해당 담당자에게 확인하십시오.

이 태스크는 AWS Management Console 또는 AWS CLI를 사용하여 수행할 수 있습니다.

콘솔을 사용하여 그룹에 AWS Cloud9 액세스 권한을 추가합니다.

1. 아직 로그인하지 않은 AWS Management Console경우 https://console.aws.amazon.com/codecommit 에서 로그인하십시오.

> **Note**
>
> 를 만들 때 제공한 이메일 주소와 AWS Management Console AWS 계정 암호로 에 로그인할 수 있습니다. 이를 일컬어 '루트 사용자로 서명'이라고 합니다. 하지만 이는 AWS 보안 모범 사례는 아닙니다. 앞으로 AWS 계정의 IAM 관리자 사용자용 보안 인증 정보를 사용하여 로그인하는 것이 좋습니다. 관리자 사용자는 AWS 계정 루트 사용자와 비슷한 AWS 액세스 권한을 가지므로 일부 관련 보안 위험을 피할 수 있습니다. 관리자 사용자로 로그인할 수

없는 경우 관리자에게 AWS 계정 문의하세요. 자세한 내용은 IAM 사용 설명서에서 첫 번째 IAM 관리자 및 그룹 만들기를 참조하세요.

2. IAM 콘솔(IAM console)을 엽니다. 이렇게 하려면 AWS 탐색 표시줄에서 서비스를 선택합니다. 그런 다음, IAM을 선택합니다.
3. 그룹을 선택합니다.
4. 그룹의 이름을 선택합니다.
5. 그룹에 AWS Cloud9 사용자 액세스 권한을 추가할지 AWS Cloud9 관리자 액세스 권한을 추가할지 결정합니다. 이러한 권한은 그룹의 각 사용자에게 적용됩니다.

   AWS Cloud9 사용자 액세스 권한을 통해 그룹의 각 사용자는 자신의 그룹 내에서 다음과 같은 작업을 수행할 수 있습니다 AWS 계정.

   - 자체 AWS Cloud9 개발 환경을 만드세요.
   - 자신의 환경과 관련한 정보를 확인합니다.
   - 자신의 환경 설정을 변경합니다.

   AWS Cloud9 관리자 액세스 권한을 통해 그룹의 각 사용자는 AWS 계정다음과 같은 추가 작업을 수행할 수 있습니다.

   - 자신 또는 다른 사람의 환경 생성
   - 자신 또는 다른 사람의 환경에 대한 정보 확인
   - 자신 또는 다른 사람의 환경 삭제
   - 자신 또는 다른 사람의 환경 설정 변경

   > **Note**
   >
   > 제한된 수의 사용자만 AWS Cloud9 관리자 그룹에 추가하는 것이 좋습니다. 이 AWS 보안 모범 사례는 AWS 리소스 액세스 관련 문제를 더 잘 제어, 추적 및 해결하는 데 도움이 될 수 있습니다.

6. 권한 탭의 관리형 정책에서 정책 연결을 선택합니다.
7. 정책 이름 목록에서 AWS Cloud9 사용자 액세스 권한 또는 AWSCloud9Administrator AWS Cloud9 관리자 액세스 권한 AWSCloud9User옆의 상자를 선택합니다. 목록에서 이러한 정책 이름 중 하나가 보이지 않으면 표시할 정책 이름을 Filter(필터) 상자에 입력합니다.
8. 정책 연결을 선택하세요.

> **Note**
>
> AWS Cloud9 액세스 권한을 추가하려는 그룹이 두 개 이상 있는 경우 각 그룹에 대해 이 절차를 반복합니다.

이러한 AWS 관리형 정책이 그룹에 부여하는 액세스 권한 목록을 보려면 AWS 관리형 (사전 정의된) 정책을 참조하십시오.

필요한 AWS 액세스 권한 외에 그룹에 추가할 수 있는 액세스 권한에 대해 알아보려면 IAM 사용 설명서의 관리형 정책 및 인라인 정책 및 정책에 의해 부여된 권한 이해를 참조하십시오. AWS Cloud9

4단계: AWS Cloud9 콘솔에 로그인으로 건너뜁니다.

다음을 사용하여 그룹에 AWS Cloud9 액세스 권한을 추가합니다. AWS CLI

> **Note**
>
> AWS 관리형 임시 자격 증명을 사용하는 경우 AWS Cloud9 IDE의 터미널 세션을 사용하여 이 섹션의 일부 또는 모든 명령을 실행할 수 없습니다. AWS 보안 모범 사례를 해결하기 위해 AWS 관리형 임시 자격 증명에서는 일부 명령을 실행할 수 없습니다. 대신 AWS Command Line Interface (AWS CLI) 를 별도로 설치하여 해당 명령을 실행할 수 있습니다.

1. 컴퓨터에 설치하고 구성하십시오 (아직 설치하지 않았다면). AWS CLI 이를 위해 AWS Command Line Interface 사용 설명서에서 다음 지침을 참조하세요.
   - AWS 명령줄 인터페이스 설치
   - 빠른 구성

   > **Note**
   >
   > 생성 시 제공한 전자 메일 주소 및 암호와 관련된 자격 증명을 AWS CLI 사용하여 구성할 수 있습니다. AWS 계정 이를 일컬어 '루트 사용자로 서명'이라고 합니다. 하지만 이는 AWS 보안 모범 사례는 아닙니다. 대신 에서 IAM 관리자 사용자의 자격 증명을 AWS CLI 사용하도록 구성하는 것이 좋습니다. AWS 계정 IAM 관리자 사용자는 AWS 계정 루트 사용자와 비슷한 AWS 액세스 권한을 가지므로 관련 보안 위험을 피할 수 있습니다. 관리자 AWS CLI 사용

자로 구성할 수 없는 경우 관리자에게 문의하십시오. AWS 계정 자세한 내용은 IAM 사용 설명서에서 첫 번째 IAM 관리자 및 그룹 만들기를 참조하세요.

2. 그룹에 AWS Cloud9 사용자 액세스 권한을 추가할지 AWS Cloud9 관리자 액세스 권한을 추가할지 결정합니다. 이러한 권한은 그룹의 각 사용자에게 적용됩니다.

   AWS Cloud9 사용자 액세스 권한을 통해 그룹의 각 사용자는 자신의 그룹 내에서 다음과 같은 작업을 수행할 수 있습니다 AWS 계정.

   - 자체 AWS Cloud9 개발 환경을 만드세요.
   - 자신의 환경과 관련한 정보를 확인합니다.
   - 자신의 환경 설정을 변경합니다.

   AWS Cloud9 관리자 액세스 권한을 통해 그룹의 각 사용자는 AWS 계정다음과 같은 추가 작업을 수행할 수 있습니다.

   - 자신 또는 다른 사람의 환경 생성
   - 자신 또는 다른 사람의 환경에 대한 정보 확인
   - 자신 또는 다른 사람의 환경 삭제
   - 자신 또는 다른 사람의 환경 설정 변경

   > **Note**
   >
   > 제한된 수의 사용자만 AWS Cloud9 관리자 그룹에 추가하는 것이 좋습니다. 이 AWS 보안 모범 사례는 AWS 리소스 액세스 관련 문제를 더 잘 제어, 추적 및 해결하는 데 도움이 될 수 있습니다.

3. IAM `attach-group-policy` 명령을 실행하여 그룹 이름과 추가할 액세스 권한 정책의 Amazon 리소스 이름 (ARN) AWS Cloud9 을 지정합니다.

   AWS Cloud9 사용자 액세스 권한에 대해 다음 ARN을 지정하십시오.

   ```
   aws iam attach-group-policy --group-name MyCloud9Group --policy-arn
    arn:aws:iam::aws:policy/AWSCloud9User
   ```

   AWS Cloud9 관리자 액세스 권한의 경우 다음 ARN을 지정하십시오.

   ```
   aws iam attach-group-policy --group-name MyCloud9Group --policy-arn
    arn:aws:iam::aws:policy/AWSCloud9Administrator
   ```

> **Note**
>
> AWS Cloud9 액세스 권한을 추가하려는 그룹이 두 개 이상 있는 경우 각 그룹에 대해 이 절차를 반복합니다.

이러한 AWS 관리형 정책이 그룹에 부여하는 액세스 권한 목록을 보려면 AWS 관리형 (사전 정의된) 정책을 참조하십시오.

필요한 AWS 액세스 권한 외에 그룹에 추가할 수 있는 액세스 권한에 대해 알아보려면 IAM 사용 설명서의 관리형 정책 및 인라인 정책 및 정책에 의해 부여된 권한 이해를 참조하십시오. AWS Cloud9

4단계: 콘솔에 로그인 AWS Cloud9

이 항목의 이전 단계를 완료했으면 사용자와 함께 AWS Cloud9 콘솔에 로그인할 수 있습니다.

1. 이미 AWS 계정 루트 AWS Management Console 사용자로 로그인한 경우 콘솔에서 로그아웃하십시오.
2. https://console.aws.amazon.com/cloud9/ 에서 AWS Cloud9 콘솔을 엽니다.
3. 이전에 생성하거나 식별한 IAM 사용자의 AWS 계정 번호를 입력하고 다음을 선택합니다.

> **Note**
>
> AWS 계정 번호를 입력하는 옵션이 보이지 않으면 다른 계정으로 로그인을 선택합니다. 다음 페이지에서 AWS 계정 번호를 입력한 후, Next(다음)를 선택합니다.

4. 이전에 생성했거나 확인한 IAM 사용자의 로그인 보안 인증 정보를 입력한 후, Sign In(로그인)을 선택합니다.
5. 메시지가 표시되면 화면 지침에 따라 사용자의 초기 로그인 암호를 변경합니다. 새 로그인 암호를 안전한 위치에 저장합니다.

AWS Cloud9 콘솔이 표시되고 사용을 시작할 수 있습니다 AWS Cloud9.

다음 단계

작업	다음 주제 참조
비용을 통제하기 위해 내 다른 사용자의 AWS Cloud9 사용을 제한하십시오 AWS 계정.	추가 설정 옵션
AWS Cloud9 개발 환경을 만든 다음 AWS Cloud9 IDE를 사용하여 새 환경에서 코드 작업을 수행하십시오.	환경 생성
AWS Cloud9 IDE 사용 방법을 알아보십시오.	시작하기: 기본 자습서 및 IDE 작업
나와 함께 다른 사람이 텍스트 채팅 지원을 통해 실시간으로 새 환경을 사용할 수 있도록 초대합니다.	공유 환경 사용

AWS Cloud9에 대한 엔터프라이즈 설정

이 주제에서는 AWS IAM Identity Center을 사용하여 엔터프라이즈 내에서 하나 이상의 AWS 계정이 AWS Cloud9를 사용하도록 설정하는 방법을 설명합니다. 기타 모든 사용 패턴에 맞춰 AWS Cloud9을 사용하도록 설정하기 위한 올바른 지침은 AWS Cloud9 설정 단원을 참조하십시오.

> **Warning**
>
> 보안 위험을 방지하려면 목적별 소프트웨어를 개발하거나 실제 데이터로 작업할 때 IAM 사용자를 인증에 사용하지 마세요. 대신 AWS IAM Identity Center과 같은 보안 인증 공급자를 통한 페더레이션을 사용하세요.

다음 지침에서는 AWS Organizations에서 조직에 대한 관리 권한이 사용자에게 이미 있거나 생길 것이라고 가정합니다. AWS Organizations에서 조직에 대한 관리 권한이 아직 없는 경우 AWS 계정 관리자에게 문의하세요. 자세한 정보는 다음 자료를 참조하세요.

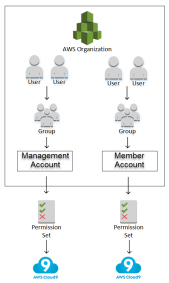
- AWS Organizations 사용 설명서(IAM Identity Center에는 AWS Organizations를 사용해야 함)의 AWS 조직에 대한 액세스 권한 관리
- AWS IAM Identity Center 사용 설명서의 IAM Identity Center 리소스에 대한 액세스 권한 관리 개요

- AWS 다중 계정 환경을 설정하고 관리하는 데 사용할 수 있는 서비스인 AWS Control Tower을 사용합니다. AWS Control Tower는 1시간 이내에 랜딩 존을 빌드하기 위해 AWS Organizations, AWS Service Catalog 및 AWS IAM Identity Center을 포함한 다른 AWS 서비스의 기능을 활용합니다.

이 주제와 관련된 소개 내용은 다음 리소스를 참조하십시오.

- AWS Organizations 사용 설명서(IAM Identity Center에는 AWS Organizations를 사용해야 함)의 AWS Organizations란 무엇입니까?
- AWS IAM Identity Center User Guide의 What is AWS IAM Identity Center
- AWS Control Tower 사용 설명서의 AWS Control Tower 시작하기
- AWS Knowledge Center Videos: How do I get started with AWS Organizations(4분짜리 동영상, YouTube 웹 사이트)
- Manage user access to multiple AWS accounts using AWS IAM Identity Center(7분짜리 동영상, YouTube 웹 사이트)
- How to set up AWS Single Sign On for your on-premise Active Directory users(9분짜리 동영상, YouTube 웹 사이트)

다음 개념 다이어그램은 어떤 결과가 나오는지 보여줍니다.



하나 이상의 AWS 계정이 엔터프라이즈 내에서 AWS Cloud9 사용을 시작하도록 설정하려면 이미 가지고 있는 AWS 리소스에 따라 단계별로 진행합니다.

AWS Organizations에서 조직에 대한 관리 계정으로 사용할 수 있거나 사용하는 AWS 계정가 있습니까?	해당 관리 계정에 대한 조직이 AWS Organizations에 있나요?	필요한 AWS 계정 멤버 모두가 해당 조직에 속해 있나요?	해당 조직이 IAM Identity Center를 사용하도록 설정되어 있습니까?	해당 조직이 AWS Cloud9을 사용하려는 모든 그룹 및 사용자로 설정되어 있습니까?	이 단계부터 시작하십시오.
아니요	—	—	—	—	1단계: 조직의 관리 계정 생성
예	아니요	—	—	—	2단계: 관리 계정의 조직 생성
예	예	아니요	—	—	3단계: 조직에 멤버 계정 추가
예	예	예	아니요	—	4단계: 조직 내에서 IAM Identity Center 사용
예	예	예	예	아니요	단계 5. 조직 내 그룹 및 사용자 설정
예	예	예	예	예	6단계. 그룹 및 사용자가 조직 내에서 AWS Cloud9을 사용하도록 지정

1단계: 조직의 관리 계정 생성

> **Note**
>
> 엔터프라이즈에 여러분을 위해 설정된 관리 계정이 이미 있을 수도 있습니다. 엔터프라이즈에 AWS 계정 관리자가 있는 경우, 다음 절차를 시작하기 전에 관리자에게 확인해 보세요. 관리 계정이 이미 있는 경우 2단계: 관리 계정의 조직 생성으로 진행하세요.

AWS IAM Identity Center(IAM ID 센터)을 사용하려면 AWS 계정가 필요합니다. AWS 계정는 AWS Organizations에서 조직의 관리 계정으로 사용됩니다. 자세한 내용은 AWS Organizations 사용 설명서의 AWS Organizations 용어 및 개념에서 관리 계정에 대한 설명을 참조하세요.

다음 절차와 관련된 4분짜리 동영상을 시청하려면 YouTube 웹 사이트에서 Creating an Amazon Web Services account를 참조하세요.

관리 계정을 생성하려면

1. https://aws.amazon.com/으로 이동합니다.
2. Sign In to the Console(콘솔에 로그인)을 선택합니다.
3. 새 AWS 계정 생성을 선택합니다.
4. 화면 지침에 따라 프로세스를 완료합니다. 여기에는 AWS에 이메일 주소와 신용카드 정보를 제공하는 것이 포함됩니다. 또한 휴대폰을 사용하여 AWS가 제공하는 코드를 입력해야 합니다.

계정 생성을 마치면 AWS가 확인 이메일을 보냅니다. 이 확인을 받을 때까지 다음 단계로 이동하지 마십시오.

2단계: 관리 계정의 조직 생성

> **Note**
>
> 엔터프라이즈에 관리 계정을 사용하도록 설정된 AWS Organizations가 이미 있을 수 있습니다. 엔터프라이즈에 AWS 계정 관리자가 있는 경우, 다음 절차를 시작하기 전에 관리자에게 확인해 보세요. 관리 계정을 사용하도록 이미 설정된 AWS Organizations가 있는 경우에는 3단계: 조직에 멤버 계정 추가로 진행하세요.

IAM Identity Center를 사용하려면 AWS Organizations에 관리 계정을 사용하는 조직이 있어야 합니다. 자세한 내용은 AWS Organizations 사용 설명서의 AWS Organizations 용어 및 개념에서 조직에 대한 설명을 참조하세요.

AWS Organizations에서 관리 AWS 계정에 대한 조직을 생성하려면 AWS Organizations 사용 설명서의 다음 지침을 따르세요.

1. 조직 생성
2. 조직 내 모든 기능 활성화

이러한 절차와 관련된 4분짜리 동영상을 보려면 YouTube 웹 사이트에서 AWS Knowledge Center Videos: How do I get started with AWS Organizations를 참조하세요.

3단계: 조직에 멤버 계정 추가

> **Note**
>
> 엔터프라이즈에 필요한 멤버 계정으로 설정된 AWS Organizations가 이미 있을 수 있습니다. 엔터프라이즈에 AWS 계정 관리자가 있는 경우, 다음 절차를 시작하기 전에 관리자에게 확인해 보세요. 필요한 멤버 계정으로 설정된 AWS Organizations가 이미 있는 경우에는 4단계: 조직 내에서 IAM Identity Center 사용으로 진행하세요.

이 단계에서는 AWS Organizations에서 조직의 멤버 계정 역할을 할 AWS 계정를 모두 추가합니다. 자세한 내용은 AWS Organizations 사용 설명서의 AWS Organizations 용어 및 개념에서 멤버 계정에 대한 설명을 참조하세요.

> **Note**
>
> 여기서 멤버 계정을 조직에 반드시 추가할 필요는 없습니다. 조직 내에서는 단일 관리 계정에만 IAM Identity Center를 사용할 수 있습니다. 이후에 원할 경우 조직에 멤버 계정을 추가할 수 있습니다. 지금 멤버 계정을 추가하지 않으려는 경우 4단계: 조직 내에서 IAM Identity Center 사용으로 진행하세요.

AWS Organizations에서 조직에 멤버 계정을 추가하려면 AWS Organizations 사용 설명서의 다음 지침 중 하나 또는 둘 다를 수행합니다. 조직의 멤버로 필요한 AWS 계정를 모두 추가할 때까지 이러한 지침을 필요한만큼 반복해서 수행하세요.

- 조직 내 AWS 계정 생성
- 조직에 가입하도록 AWS 계정 초대

4단계: 조직 내에서 IAM Identity Center 사용

> **Note**
>
> 엔터프라이즈에 IAM Identity Center를 사용하도록 설정된 AWS Organizations가 이미 있을 수 있습니다. 엔터프라이즈에 AWS 계정 관리자가 있는 경우, 다음 절차를 시작하기 전에 관리자에게 확인해 보세요. IAM Identity Center를 사용하도록 설정된 AWS Organizations가 이미 있는 경우 5단계.로 진행하세요. 조직 내 그룹 및 사용자 설정으로 건너뛰세요.

이 단계에서는 AWS Organizations의 조직이 IAM Identity Center를 사용하도록 지정합니다. 이를 위해 AWS IAM Identity Center 사용 설명서에서 다음 지침을 따르세요.

1. IAM Identity Center 사전 조건
2. IAM Identity Center 활성화

5단계. 조직 내 그룹 및 사용자 설정

> **Note**
>
> 엔터프라이즈에는 IAM Identity Center 디렉터리, AWS Managed Microsoft AD 또는 AWS Directory Service에서 관리되는 AD Connector 디렉터리의 그룹 및 사용자로 설정된 AWS Organizations이 이미 존재할 수 있습니다. 엔터프라이즈에 AWS 계정 관리자가 있는 경우, 다음 절차를 시작하기 전에 관리자에게 확인해 보세요. IAM Identity Center 디렉터리 또는 AWS Directory Service의 그룹 및 사용자로 설정된 AWS Organizations가 이미 있는 경우 6단계로 진행하세요. 그룹 및 사용자가 조직 내에서 AWS Cloud9을 사용하도록 지정으로 건너뛰세요.

이 단계에서는 조직의 IAM Identity Center 디렉터리에 그룹 및 사용자를 생성합니다. 또는 조직의 AWS Directory Service에서 관리하는 AWS Managed Microsoft AD 또는 AD Connector 디렉터리에 연결할 수도 있습니다. 후반 단계에서는 그룹에 AWS Cloud9 사용을 위한 필수 액세스 권한을 부여합니다.

- 조직에 IAM Identity Center 디렉터리를 사용하는 경우 AWS IAM Identity Center 사용 설명서의 지침을 따릅니다. 필요한 그룹 및 사용자가 모두 지정될 때까지 다음 단계를 반복해서 수행하십시오.
  1. 그룹을 추가합니다. 조직 전체의 모든 AWS Cloud9 관리자를 위한 그룹을 적어도 하나 이상 만드는 것이 좋습니다. 그런 다음, 이 단계를 반복하여 조직 전체의 모든 AWS Cloud9 사용자를 위한 또 다른 그룹을 생성합니다. 경우에 따라서는 이러한 단계를 반복해 조직 전체에서 기존 AWS Cloud9 개발 환경을 공유하려는 모든 사용자를 위한 세 번째 그룹을 생성할 수도 있습니다. 하지만 이러한 사용자가 환경을 직접 생성하도록 허용하지는 마십시오. 간단하게 사용할 수 있도록 이러한 그룹은 각각 `AWSCloud9Administrators`, `AWSCloud9Users` 및 `AWSCloud9EnvironmentMembers`로 이름을 지정하는 것이 좋습니다. 자세한 내용은 AWS Cloud9에 대한 AWS 관리형(미리 정의된) 정책을 참조하세요.
  2. 사용자를 추가합니다.
  3. 그룹에 사용자를 추가합니다. `AWSCloud9Administrators` 그룹에 AWS Cloud9 관리자를 추가하고 이 단계를 반복하여 `AWSCloud9Users` 그룹에 AWS Cloud9 사용자를 추가합니다. 필요에 따라 이 단계를 반복하여 나머지 사용자를 `AWSCloud9EnvironmentMembers` 그룹에 추가할 수도 있습니다. 그룹에 사용자 추가는 AWS 보안 모범 사례로, AWS 리소스 액세스와 관련된 문제를 더 효과적으로 제어, 추적 및 해결하는 데 도움이 됩니다.
- 조직을 위해 AWS Directory Service에서 관리하는 AWS Managed Microsoft AD 또는 AD Connector 디렉터리를 사용 중인 경우 AWS IAM Identity Center 사용 설명서에서 Microsoft AD 디렉터리에 연결을 참조하세요.

6단계. 그룹 및 사용자가 조직 내에서 AWS Cloud9을 사용하도록 지정

기본적으로 AWS Organizations에서 조직 내 대부분의 사용자 및 그룹은 AWS Cloud9을 포함해 모든 AWS 서비스에 액세스할 수 없습니다. 이 단계에서는 IAM Identity Center를 사용하여 AWS Organizations의 조직 내 그룹 및 사용자가 참여 중인 계정의 모든 조합 내에서 AWS Cloud9을 사용하도록 허용합니다.

1. IAM Identity Center 콘솔의 서비스 탐색 창에서 AWS 계정를 선택합니다.
2. Permission sets(권한 세트) 탭을 선택합니다.
3. Create permission set(권한 세트 생성)를 선택합니다.
4. Create a custom permission set(사용자 지정 권한 세트 생성)를 선택합니다.
5. 이 권한 세트의 이름을 입력합니다. 조직 전체의 모든 AWS Cloud9 관리자를 위한 권한 세트를 적어도 하나 이상 만드는 것이 좋습니다. 그런 다음, 이 절차의 3단계부터 10단계까지 반복하여 조직 전체의 모든 AWS Cloud9 사용자를 위한 또 다른 권한 세트를 생성합니다. 경우에 따라서는 3~10단계를 반복해 조직 전체에서 기존 AWS Cloud9 개발 환경을 공유하려는 모든 사용자를 위한 세 번째 권

한 세트를 생성할 수도 있습니다. 하지만 이러한 사용자가 환경을 직접 생성하도록 허용하지는 마십시오. 간단하게 사용할 수 있도록 이러한 권한 세트는 각각 `AWSCloud9AdministratorsPerms`, `AWSCloud9UsersPerms` 및 `AWSCloud9EnvironmentMembersPerms`로 이름을 지정하는 것이 좋습니다. 자세한 내용은 AWS Cloud9에 대한 AWS 관리형(미리 정의된) 정책을 참조하세요.

6. 권한 세트에 대한 설명을 선택적으로 입력합니다.
7. 권한 세트에 대한 Session duration(세션 기간)을 선택하거나 1시간의 기본 세션 기간을 그대로 둡니다.
8. Attach AWS managed policies(AWS 관리형 정책 연결)를 선택합니다.
9. 정책 목록에서 올바른 Policy name(정책 이름) 항목 옆에 있는 다음 상자 중 하나를 선택합니다. (정책 이름 자체를 선택하지 마세요. 목록에 정책 이름이 보이지 않으면 정책 이름을 [검색(Search)] 상자에 입력해 표시합니다.)
   - `AWSCloud9AdministratorsPerms` 권한 세트의 경우 AWSCloud9Administrator를 선택합니다.
   - `AWSCloud9UsersPerms` 권한 세트의 경우 AWSCloud9User를 선택합니다.
   - 경우에 따라 `AWSCloud9EnvironmentMembersPerms` 권한 세트의 경우 AWSCloud9EnvironmentMember를 선택합니다.

   > **Note**
   >
   > AWS Cloud9에 필요한 정책 외에 추가할 수 있는 정책에 대해 알아보려면 IAM 사용 설명서에서 관리형 정책과 인라인 정책 및 정책에 의해 부여된 권한 이해를 참조하세요.

10. 생성을 선택합니다.
11. 필요한 권한 세트 생성을 모두 마친 후에는 AWS organization(AWS 조직) 탭에서 AWS Cloud9 액세스 권한을 할당하려는 AWS 계정를 선택합니다. AWS organization(AWS 조직) 탭이 보이지 않으면 서비스 탐색 창에서 AWS 계정를 선택합니다. 그러면 AWS organization(AWS 조직) 탭이 표시됩니다.
12. Assign users(사용자 배정)를 선택합니다.
13. Groups(그룹) 탭에서 AWS Cloud9 액세스 권한을 할당하려는 그룹의 이름 옆 상자를 선택합니다. 그룹 이름 자체를 선택하지 마십시오.
    - 조직을 위해 IAM Identity Center 디렉터리를 사용 중인 경우, AWS Cloud9 관리자를 위해 AWSCloud9Administrators라는 그룹을 만들었을 수도 있습니다.
    - 조직을 위해 AWS Directory Service에서 관리하는 AWS Managed Microsoft AD 또는 AD Connector 디렉터리를 사용 중인 경우 디렉터리의 ID를 선택합니다. 그런 다음, 그룹 이름의 일부

또는 전체를 입력하고 Search connected directory(연결된 디렉터리 검색)을 선택합니다. 마지막으로 AWS Cloud9 액세스 권한을 할당하려는 그룹의 이름 옆 상자를 선택합니다.

> **Note**
>
> 개인 사용자 대신 그룹에 AWS Cloud9 액세스 권한을 할당하는 것이 좋습니다. 이 AWS 보안 모범 사례는 AWS 리소스 액세스와 관련된 문제를 더 효과적으로 제어, 추적 및 해결하는 데 도움이 됩니다.

14. Next: Permission sets(다음: 권한 세트)를 선택합니다.
15. 이 그룹에 할당하려는 권한 세트 이름 옆의 상자를 선택합니다(예를 들어 AWS Cloud9 관리자 그룹의 경우 AWSCloud9AdministratorsPerms를 선택할 것). 권한 세트 이름 자체를 선택하지 마십시오.
16. [마침]을 클릭합니다.
17. AWS 계정로 진행을 선택합니다.
18. 조직 전체에서 AWS 계정에 할당하려는 모든 추가 AWS Cloud9 액세스 권한에 대해 이 절차의 11~17단계를 반복합니다.

7단계: AWS Cloud9 사용 시작

이 주제의 이전 단계를 완료하면 여러분과 사용자가 IAM Identity Center에 로그인하여 AWS Cloud9 사용을 시작할 준비가 됩니다.

1. 이미 AWS 계정 또는 IAM Identity Center에 로그인한 경우 로그아웃합니다. 이렇게 하려면 AWS Support 웹 사이트의 AWS 계정에서 로그아웃하는 방법을 참조하거나 AWS IAM Identity Center 사용 설명서에서 사용자 포털에서 로그아웃하는 방법을 참조하세요.
2. IAM Identity Center에 로그인하려면의 AWS IAM Identity Center 사용 설명서에서 IAM Identity Center 가입 초대를 수락하는 방법의 지침을 따르세요. 여기에는 고유한 로그인 URL로 이동하고 고유한 로그인 보안 인증 정보로 로그인하는 등의 작업이 포함됩니다. AWS 계정 관리자는 이러한 정보를 이메일로 보내거나 직접 제공할 수 있습니다.

> **Note**
>
> 제공된 고유한 로그인 URL을 북마크에 꼭 추가하세요. 이렇게 하면 나중에 쉽게 되돌아갈 수 있습니다. 또한 이 URL의 고유한 로그인 보안 인증 정보를 안전한 위치에 저장해야 합니다.

URL, 사용자 이름 및 암호의 조합은 AWS 계정 관리자가 사용자에게 부여하는 AWS Cloud9 액세스 권한의 수준에 따라 바뀔 수 있습니다. 예를 들어 하나의 URL, 사용자 이름 및 암호를 사용하여 하나의 계정에 대한 AWS Cloud9 관리자 액세스 권한을 얻을 수 있습니다. AWS Cloud9 사용자만 다른 계정에 액세스할 수 있도록 상이한 URL, 사용자 이름 및 암호를 사용할 수 있습니다.

3. IAM Identity Center에 로그인한 후에는 AWS 계정 타일을 선택합니다.
4. 표시되는 드롭다운 목록에서 사용자의 표시 이름을 선택합니다. 이름이 두 개 이상 표시되면 AWS Cloud9 사용을 시작하려는 이름을 선택합니다. 어떤 이름을 선택해야 할지 모르겠다면 AWS 계정 관리자에게 문의하세요.
5. 사용자의 표시 이름 옆에 Management console(관리 콘솔) 링크를 선택합니다. Management console(관리 콘솔) 링크가 두 개 이상 표시되면 올바른 권한 세트 옆에 있는 링크를 선택합니다. 어떤 링크를 선택해야 할지 모르겠다면 AWS 계정 관리자에게 문의하세요.
6. AWS Management Console에서 다음 중 하나를 수행합니다.
   - 이미 표시된 경우에는 Cloud9을 선택합니다.
   - All services(모든 서비스)를 확장한 다음 Cloud9을 선택합니다.
   - Find services(서비스 찾기) 상자에 Cloud9을 입력한 후 `Enter` 키를 누릅니다.
   - AWS 탐색 모음에서 [서비스(Services)]를 선택한 후 [Cloud9]을 선택합니다.

AWS Cloud9 콘솔이 표시되면 AWS Cloud9 사용을 시작할 수 있습니다.

다음 단계

작업	다음 주제 참조
AWS Cloud9 개발 환경을 만든 다음 AWS Cloud9 IDE를 사용하여 새 환경에서 코드를 작업합니다.	환경 생성
AWS Cloud9 IDE를 사용하는 방법을 알아봅니다.	시작하기: 기본 자습서 및 IDE 작업
나와 함께 다른 사람이 텍스트 채팅 지원을 통해 실시간으로 새 환경을 사용할 수 있도록 초대합니다.	공유 환경 사용

AWS Cloud9에 대한 추가 설정 옵션(팀 및 엔터프라이즈)

이 주제에서는 Team Setup(팀 설정) 또는 Enterprise Setup(엔터프라이즈 설정)의 설정 단계를 이미 완료했다고 가정합니다.

Team Setup(팀 설정) 또는 Enterprise Setup(엔터프라이즈 설정)에서 그룹을 생성하고 그러한 그룹에 직접 AWS Cloud9 액세스 권한을 추가했습니다. 이는 해당 그룹의 사용자가 AWS Cloud9에 액세스할 수 있도록 하기 위한 것입니다. 이 주제에서는 그러한 그룹의 사용자가 생성할 수 있는 환경 종류를 제한하는 액세스 권한을 추가합니다. 그러면 AWS 계정 및 조직의 AWS Cloud9과 관련된 비용을 통제하는 데 도움이 됩니다.

이러한 액세스 권한을 추가하려면 적용하려는 AWS 액세스 권한을 정의하는 고유 정책 집합을 에 만듭니다. 이를 일컬어 각각 고객 관리형 정책이라고 합니다. 그런 다음, 사용자가 속한 그룹에 그러한 고객 관리형 정책을 연결합니다. 일부 시나리오에서는 그러한 그룹에 이미 연결된 기존 AWS 관리형 정책도 분리해야 합니다. 이를 설정하려면 이 주제의 절차를 따릅니다.

> **Note**
>
> 다음 절차에서는 AWS Cloud9 사용자 그룹에 대해서만 정책을 연결 및 분리하는 방법을 다룹니다. 이 절차에서는 이미 별도의 AWS Cloud9 사용자 그룹과 AWS Cloud9 관리자 그룹이 있다고 가정합니다. 또한 AWS Cloud9 관리자 그룹의 사용자 수가 제한되어 있다고 가정합니다. 이 AWS 보안 모범 사례는 AWS 리소스 액세스와 관련된 문제를 더 효과적으로 제어, 추적 및 해결하는 데 도움이 됩니다.

1단계: 고객 관리형 정책 만들기

AWS Management Console 또는 AWS Command Line Interface(AWS CLI)를 사용하여 고객 관리형 정책을 만들 수 있습니다.

> **Note**
>
> 이 단계에서는 IAM 그룹에 대한 고객 관리형 정책 생성만 다룹니다. AWS IAM Identity Center에서 그룹에 대한 사용자 지정 권한 세트를 만들려면 이 단계를 건너뛰고 대신 AWS IAM

Identity Center 사용 설명서에서 권한 세트 생성의 지침을 따르세요. 이 주제에서는 이러한 지침에 따라 사용자 지정 권한 세트를 생성합니다. 관련 사용자 지정 권한 정책은 이 주제 뒷부분에 나오는 AWS Cloud9을 사용한 팀용 고객 관리형 정책 예를 참조하세요.

콘솔을 사용하여 고객 관리형 정책 만들기

1. 아직 로그인하지 않았다면 AWS Management Console에 로그인합니다.

   AWS 계정의 관리자 사용자용 보안 인증 정보를 사용하여 로그인하는 것이 좋습니다. 이렇게 할 수 없으면 AWS 계정 관리자에게 문의하세요.
2. IAM 콘솔(IAM console)을 엽니다. 이렇게 하려면 콘솔의 탐색 모음에서 서비스를 선택합니다. 그런 다음 IAM을 선택합니다.
3. 서비스의 탐색 창에서 정책을 선택합니다.
4. [정책 생성(Create policy)]을 선택합니다.
5. JSON 탭에 제안된 고객 관리형 정책 예 중 하나를 붙여 넣습니다.

   > **Note**
   >
   > 고유한 고객 관리형 정책을 만들 수도 있습니다. 자세한 내용은 IAM 사용 설명서의 IAM JSON 정책 참조와 AWS 서비스의 문서를 참조하세요.

6. Review policy(정책 검토)를 선택합니다.
7. 정책 검토 페이지에 정책의 이름과 설명(선택 영역)을 입력한 후, 정책 생성을 선택합니다.

생성하려는 각 추가 고객 관리형 정책에 대해 이 단계를 반복합니다. 그런 다음, 콘솔을 사용하여 그룹에 고객 관리형 정책 추가로 건너뜁니다.

AWS CLI를 사용하여 고객 관리형 정책 만들기

1. AWS CLI를 실행하는 컴퓨터에 정책을 설명하는 파일을 만듭니다(예: `policy.json`).

   다른 파일 이름으로 파일을 만들 경우, 이 절차 전체에서 해당 이름으로 바꿉니다.
2. 제안된 고객 관리형 정책 예 중 하나를 `policy.json` 파일에 붙여 넣습니다.

> **Note**
>
> 고유한 고객 관리형 정책을 만들 수도 있습니다. 자세한 내용은 IAM 사용 설명서의 IAM JSON 정책 참조와 AWS 서비스의 문서를 참조하세요.

3. 터미널 또는 명령 프롬프트에서 `policy.json` 파일이 포함된 디렉터리로 전환합니다.
4. 정책 이름과 `policy.json` 파일을 지정하여 IAM `create-policy` 명령을 실행합니다.

```
aws iam create-policy --policy-document file://policy.json --policy-name MyPolicy
```

앞의 명령에서 `MyPolicy`를 해당 정책의 이름으로 바꿉니다.

AWS CLI를 사용하여 그룹에 고객 관리형 정책 추가로 건너뜁니다.

2단계: 그룹에 고객 관리형 정책 추가

AWS Management Console 또는 AWS Command Line Interface(AWS CLI)를 사용하여 그룹에 고객 관리형 정책을 추가할 수 있습니다.

> **Note**
>
> 이 단계에서는 IAM 그룹에 고객 관리형 정책을 추가하는 것만 다룹니다. AWS IAM Identity Center에서 그룹에 사용자 지정 권한 세트를 추가하려면 이 단계를 건너뛰고 대신 AWS IAM Identity Center 사용 설명서에서 사용자 액세스 권한 할당의 지침을 따르세요.

콘솔을 사용하여 그룹에 고객 관리형 정책 추가

1. 이전 절차에서 IAM 콘솔을 연 상태로 서비스의 탐색 창에서 [그룹(Groups)]을 선택합니다.
2. 그룹의 이름을 선택합니다.
3. 권한 탭의 관리형 정책에서 정책 연결을 선택합니다.
4. 정책 이름 목록에서 그룹에 연결하려는 각 고객 관리형 정책 옆의 상자를 선택합니다. 목록에서 특정 정책 이름이 보이지 않으면 표시할 정책 이름을 Filter(필터) 상자에 입력합니다.
5. 정책 연결(Attach Policy)을 선택합니다.

AWS CLI를 사용하여 그룹에 고객 관리형 정책 추가

> **Note**
>
> AWS 관리형 임시 자격 증명을 사용하는 경우 AWS Cloud9 IDE의 터미널 세션을 사용하여 이 섹션의 일부 또는 모든 명령을 실행할 수 없습니다. AWS 보안 모범 사례를 따르기 위해 AWS 관리형 임시 자격 증명은 일부 명령의 실행을 허용하지 않습니다. 대신에 별도의 AWS Command Line Interface(AWS CLI) 설치에서 이러한 명령을 실행할 수 있습니다.

그룹의 이름과 정책의 Amazon 리소스 이름(ARN)을 지정하여 IAM `attach-group-policy` 명령을 실행합니다.

```
aws iam attach-group-policy --group-name MyGroup --policy-arn
 arn:aws:iam::123456789012:policy/MyPolicy
```

앞의 명령에서 `MyGroup`을 그룹의 이름으로 바꿉니다. `123456789012`를 AWS 계정 ID로 바꿉니다. `MyPolicy`를 해당 고객 관리형 정책의 이름으로 바꿉니다.

AWS Cloud9를 사용한 팀용 고객 관리형 정책 예

다음은 그룹의 사용자가 AWS 계정에 생성할 수 있는 환경을 제한할 때 사용할 수 있는 정책의 일부 예입니다.

- 그룹의 사용자가 환경을 생성하지 못하게 차단
- 그룹의 사용자가 EC2 환경을 생성하지 못하게 차단
- 그룹의 사용자가 특정 Amazon EC2 인스턴스 유형이 있는 EC2 환경만 생성하도록 허용
- 그룹의 사용자가 AWS 리전당 하나의 EC2 환경만 생성하도록 허용

그룹의 사용자가 환경을 생성하지 못하게 차단

다음 고객 관리형 정책은 AWS Cloud9 사용자 그룹에 연결할 경우 해당 사용자가 AWS 계정에 환경을 생성하지 못하게 합니다. 이는 AWS 계정의 관리자 사용자가 환경 생성을 관리하게 하려는 경우에 유용합니다. 그렇지 않으면 AWS Cloud9 사용자 그룹의 사용자가 이 작업을 수행합니다.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "cloud9:CreateEnvironmentEC2",
        "cloud9:CreateEnvironmentSSH"
      ],
      "Resource": "*"
    }
  ]
}
```

위의 고객 관리형 정책은 AWS Cloud9 사용자 그룹에 이미 연결된 AWSCloud9User 관리형 정책에 있는 "Resource": "*"의 "Action": "cloud9:CreateEnvironmentEC2" 및 "cloud9:CreateEnvironmentSSH"에 대한 "Effect": "Allow"를 명시적으로 재정의합니다.

그룹의 사용자가 EC2 환경을 생성하지 못하게 차단

다음 고객 관리형 정책은 AWS Cloud9 사용자 그룹에 연결할 경우 해당 사용자가 AWS 계정에 EC2 환경을 생성하지 못하게 합니다. 이는 AWS 계정의 관리자 사용자가 EC2 환경 생성을 관리하게 하려는 경우에 유용합니다. 그렇지 않으면 AWS Cloud9 사용자 그룹의 사용자가 이 작업을 수행합니다. 또한 이 정책은 해당 그룹의 사용자가 SSH 환경을 생성하지 못하게 하는 정책을 연결하지 않았다고 가정합니다. 그렇지 않으면 해당 사용자가 환경을 만들 수 없습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*"
    }
  ]
}
```

위의 고객 관리형 정책은 AWS Cloud9 사용자 그룹에 이미 연결된 AWSCloud9User 관리형 정책에 있는 "Resource": "*"의 "Action": "cloud9:CreateEnvironmentEC2"에 대한 "Effect": "Allow"를 명시적으로 재정의합니다.

그룹의 사용자가 특정 Amazon EC2 인스턴스 유형이 있는 EC2 환경만 생성하도록 허용

다음 고객 관리형 정책은 AWS Cloud9 사용자 그룹에 연결할 경우 해당 사용자가 AWS 계정에서 t2로 시작하는 인스턴스 유형만 사용하는 EC2 환경을 생성하도록 허용합니다. 이 정책은 해당 그룹의 사용자가 EC2 환경을 생성하지 못하게 하는 정책을 연결하지 않았다고 가정합니다. 그렇지 않으면 해당 사용자가 EC2 환경을 만들 수 없습니다.

다음 정책의 "t2.*"를 다른 인스턴스 클래스(예: "m4.*")로 바꿀 수 있습니다. 또는 여러 인스턴스 클래스 또는 인스턴스 유형(예: ["t2.*", "m4.*"] 또는 ["t2.micro", "m4.large"])으로 제한할 수 있습니다.

AWS Cloud9 사용자 그룹의 경우, 그룹에서 AWSCloud9User 관리형 정책을 분리합니다. 그런 다음, 그 자리에 다음 고객 관리형 정책을 추가합니다. AWSCloud9User 관리형 정책을 분리하지 않으면 다음 고객 관리형 정책이 작동하지 않습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloud9:CreateEnvironmentSSH",
        "cloud9:ValidateEnvironmentName",
        "cloud9:GetUserPublicKey",
        "cloud9:UpdateUserSettings",
        "cloud9:GetUserSettings",
        "iam:GetUser",
        "iam:ListUsers",
        "ec2:DescribeVpcs",
        "ec2:DescribeSubnets"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "cloud9:InstanceType": "t2.*"
        }
```

```
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloud9:DescribeEnvironmentMemberships"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "Null": {
          "cloud9:UserArn": "true",
          "cloud9:EnvironmentId": "true"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": "cloud9.amazonaws.com"
        }
      }
    }
  ]
}
```

위의 고객 관리형 정책은 해당 사용자가 SSH 환경도 생성하도록 허용합니다. 해당 사용자가 SSH 환경을 생성하지 못하게 하려면 앞의 고객 관리형 정책에서 `"cloud9:CreateEnvironmentSSH",`를 제거합니다.

그룹의 사용자가 AWS 리전당 하나의 EC2 환경만 생성하도록 허용

다음 고객 관리형 정책은 AWS Cloud9 사용자 그룹에 연결할 경우 각 해당 사용자가 AWS Cloud9를 사용할 수 있는 AWS 리전당 최대 하나의 EC2 환경을 생성하도록 허용합니다. 이를 위해 환경 이름을 해당 AWS 리전의 특정 이름으로 제한합니다. 이 예에서는 환경이 `my-demo-environment`로 제한됩니다.

> **Note**
>
> AWS Cloud9는 특정 AWS 리전에 대한 환경이 생성되지 않도록 제한할 수 없습니다. 또한 AWS Cloud9는 만들 수 있는 전체 환경 수를 제한할 수 없습니다. 유일한 예외는 게시된 서비스 한도입니다.

AWS Cloud9 사용자 그룹의 경우, 그룹에서 `AWSCloud9User` 관리형 정책을 분리한 후 그 자리에 다음 고객 관리형 정책을 추가합니다. `AWSCloud9User` 관리형 정책을 분리하지 않으면 다음 고객 관리형 정책이 작동하지 않습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloud9:CreateEnvironmentSSH",
        "cloud9:ValidateEnvironmentName",
        "cloud9:GetUserPublicKey",
        "cloud9:UpdateUserSettings",
        "cloud9:GetUserSettings",
        "iam:GetUser",
        "iam:ListUsers",
        "ec2:DescribeVpcs",
        "ec2:DescribeSubnets"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloud9:CreateEnvironmentEC2"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "cloud9:EnvironmentName": "my-demo-environment"
        }
      }
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "cloud9:DescribeEnvironmentMemberships"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "Null": {
          "cloud9:UserArn": "true",
          "cloud9:EnvironmentId": "true"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": "cloud9.amazonaws.com"
        }
      }
    }
  ]
}
```

위의 고객 관리형 정책은 해당 사용자가 SSH 환경도 생성하도록 허용합니다. 해당 사용자가 SSH 환경을 생성하지 못하게 하려면 앞의 고객 관리형 정책에서 `"cloud9:CreateEnvironmentSSH",`를 제거합니다.

더 많은 예제는 고객 관리형 정책 예를 참조하세요.

다음 단계

작업	다음 주제 참조
생성AWS Cloud9개발 환경을 만든 다음AWS Cloud9IDE를 사용하여 새 환경에서 코드를 작업합니다.	환경 생성

작업	다음 주제 참조
AWS Cloud9 IDE를 사용하는 방법을 알아봅니다.	시작하기: 기본 자습서 및 IDE 작업
나와 함께 다른 사람이 텍스트 채팅 지원을 통해 실시간으로 새 환경을 사용할 수 있도록 초대합니다.	공유 환경 사용

시작하기: AWS Cloud9에 대한 기본 자습서

AWS Cloud9를 처음 사용하시나요? 아직 살펴보지 않은 경우 란 무엇입니까?AWS Cloud9에서 AWS Cloud9에 대한 일반 정보를 참조하세요.

다음 자습서에서는 AWS Cloud9에서 환경을 생성한 다음 해당 환경을 사용하여 간단한 애플리케이션을 생성합니다. 두 자습서의 입력과 결과는 동일하지만 각각 AWS Cloud9 콘솔 및 AWS Command Line Interface(AWS CLI)을 사용합니다. 둘 중 하나 또는 모두를 수행할 수 있습니다.

자습서를 완료한 후 AWS Cloud9 IDE 둘러보기에서 AWS Cloud9 IDE에 대해 자세히 알아볼 수 있습니다.

주제

- 자습서: Hello AWS Cloud9(콘솔)
- 자습서: Hello AWS Cloud9(CLI)

자습서: Hello AWS Cloud9(콘솔)

이 자습서는 AWS Cloud9을 처음 접하는 사용자를 위한 것이며, AWS Cloud9 콘솔을 사용하고 탐색하는 방법을 다룹니다.

이 자습서에서는 AWS Cloud9 개발 환경을 설정한 다음 AWS Cloud9 IDE를 사용하여 첫 번째 애플리케이션을 코딩, 실행 및 디버깅합니다.

이 자습서는 완료하는데 약 1시간이 걸립니다.

> **Warning**
>
> 이 자습서를 완료하면 AWS 리전에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금을 참조하세요.

사전 조건

이 자습서를 완료하려면 먼저 AWS Cloud9 설정의 단계를 완료해야 합니다.

단계

단계 1: 환경 조성

(자습서: Hello AWS Cloud9(콘솔)의 첫 단계)

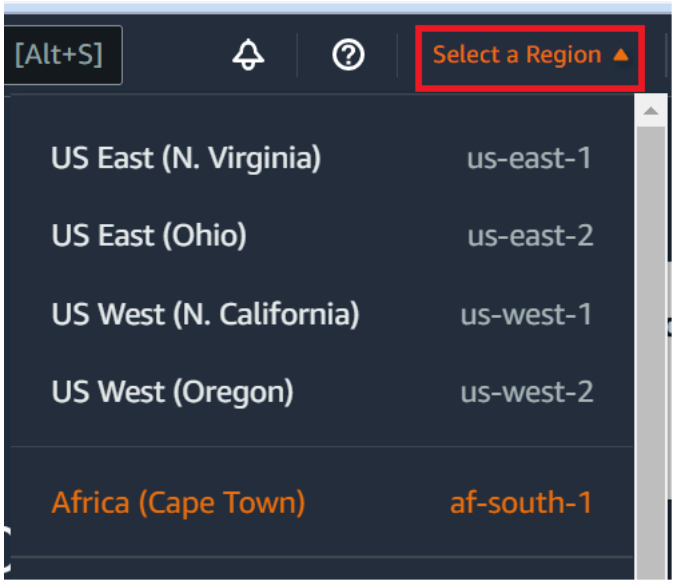
이 단계에서는 AWS Cloud9 콘솔을 사용하여 AWS Cloud9 개발 환경을 생성한 후 엽니다.

> **Note**
>
> 이 자습서에 사용할 환경을 이미 만든 경우 해당 환경을 열고 2단계: IDE의 기본 사항 둘러보기로 건너뜁니다.

AWS Cloud9에서 개발 환경 또는 환경은 개발 프로젝트의 파일을 저장하고 도구를 실행하여 애플리케이션을 개발하는 곳입니다. 이 자습서에서는 EC2 환경을 생성하고 이 환경에서 파일과 도구를 작업합니다.

콘솔을 사용한 EC2 환경 생성

1. AWS Cloud9 콘솔에 로그인합니다.
   - AWS 계정를 혼자만 사용하는 경우 또는 단일 AWS 계정의 IAM 사용자인 경우 https://console.aws.amazon.com/cloud9/으로 이동합니다.
   - 조직에서 AWS IAM Identity Center을 사용하는 경우 로그인 지침은 AWS 계정 관리자에게 문의하세요.
   - 교실의 학생인 경우 로그인 지침은 강사에게 문의하세요.
2. AWS Cloud9 콘솔에 로그인한 후 상단 탐색 모음에서 환경을 생성할 AWS 리전을 선택합니다. 사용 가능한 AWS 리전 목록은 AWS 일반 참조의 AWS Cloud9 단원을 참조하세요.



3. 여기에 나와 있는 위치 중 한 곳에서 큰 [환경 생성(Create environment)] 버튼을 선택합니다.

   AWS Cloud9 환경이 없는 경우 이 버튼이 시작 페이지에 표시됩니다.

   AWS Cloud9 환경이 이미 있는 경우 이 버튼이 다음과 같이 표시됩니다.

4. Create environment(환경 생성) 페이지의 Name(이름)에 환경의 이름을 입력합니다.
5. Description(설명)에 환경에 대한 설명을 입력합니다. 본 자습서에서는 `This environment is for the AWS Cloud9 tutorial.`을 사용합니다.
6. Environment type(환경 유형)에서 New EC2 instance(새 EC2 인스턴스)를 선택하여 Amazon EC2 환경을 만듭니다.

   - New EC2 instance새 (EC2 인스턴스) - AWS Cloud9이 SSH를 통해 직접 연결할 수 있는 새 Amazon EC2 인스턴스를 시작합니다. Systems Manager를 사용하여 새 Amazon EC2 인스턴스와 상호 작용할 수 있습니다. 자세한 내용은 AWS Systems Manager를 사용하여 수신하지 않는 EC2 인스턴스에 액세스 섹션을 참조하세요.

- Existing compute(기존 컴퓨팅) - SSH 로그인 세부 정보가 필요한 기존 Amazon EC2 인스턴스를 시작합니다. 이 경우 Amazon EC2 인스턴스에 인바운드 보안 그룹 규칙이 있어야 합니다.
  - Existing compute(기존 컴퓨팅) 옵션을 선택하면 서비스 역할이 자동으로 생성됩니다. 설정 화면 하단에 표시되는 메모에서 서비스 역할의 이름을 확인할 수 있습니다.

> **Note**
>
> 기존 컴퓨팅을 사용하는 Amazon EC2 인스턴스를 사용하여 생성된 AWS Cloud9 환경에서는 자동 종료를 사용할 수 없습니다.

> **Warning**
>
> 환경의 Amazon EC2 인스턴스를 생성하면 AWS 계정에 Amazon EC2 요금이 발생할 수 있습니다. Systems Manager를 사용하여 EC2 인스턴스에 대한 연결을 관리하는 데 따른 추가 비용은 없습니다.

7. Instance type(인스턴스 유형)의 New EC2 인스턴스 패널에서 기본 선택을 그대로 유지합니다. 이 옵션은 RAM과 vCPU가 더 적을 수 있습니다. 하지만 이 자습서에서는 이 정도 메모리로도 충분합니다.

> **Warning**
>
> 더 많은 RAM 및 vCPU가 있는 인스턴스 유형을 선택하면 Amazon EC2의 AWS 계정에 추가 비용이 발생할 수 있습니다.

8. 플랫폼에서 원하는 Amazon EC2 인스턴스 유형으로 Amazon Linux 2, Amazon Linux 또는 Ubuntu 22.04 LTS를 선택합니다. AWS Cloud9에서 인스턴스를 생성한 다음 환경을 인스턴스에 연결합니다.

> **Important**
>
> EC2 환경에 대해 Amazon Linux 2023 옵션을 선택하는 것이 좋습니다. Amazon Linux 2023 AMI는 안전하고 안정적인 고성능 런타임 환경을 제공할 뿐만 아니라 2024년까지 장기적인 지원을 제공합니다.

자세한 내용은 AL2023 페이지를 참조하십시오.

9. Timeout(제한 시간)을 선택합니다. 이 옵션은 AWS Cloud9이 자동 최대 절전 모드로 전환되기 전까지 비활성 상태를 유지하는 시간을 결정합니다. 환경의 IDE에 연결된 모든 웹 브라우저 인스턴스가 닫히면 AWS Cloud9이 지정된 시간 동안 대기했다가 환경의 Amazon EC2 인스턴스를 종료합니다.

   Warning

   긴 기간을 선택할수록 AWS 계정에 더 많은 요금이 발생할 수 있습니다.

10. Network settings(네트워크 설정) 패널에서, 환경에 액세스하는 방법으로 다음 두 가지 옵션 중 하나를 선택합니다.

    - AWS System Manager(SSM) – 이 방법은 인바운드 포트를 열지 않고 SSM을 사용하여 환경에 액세스합니다.

    - SSH(Secure Shell) – 이 방법은 SSH를 사용하여 환경에 액세스하며 열린 인바운드 포트가 필요합니다.

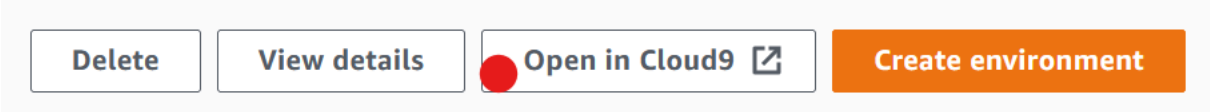
11. VPC Settings(VPC 설정)를 선택하여 환경의 Amazon Virtual Private Cloud 및 서브넷을 표시합니다. AWS Cloud9은 Amazon Virtual Private Cloud(Amazon VPC)를 사용하여, 새로 생성된 Amazon EC2 인스턴스와 통신합니다. 이 자습서에서는 미리 선택된 기본 설정을 변경하지 않는 것이 좋습니다. 기본 설정을 사용하면 AWS Cloud9이 새 환경과 동일한 AWS 계정 및 리전에서 단일 서브넷이 있는 기본 VPC를 자동으로 사용하려고 시도합니다.

    Amazon VPC 대한 자세한 내용은 콘솔을 사용하여 EC2 환경 생성과 개발 환경을 위한 AWS Cloud9 VPC 설정 섹션에서 확인할 수 있습니다.

12. 각 태그에 키와 값을 지정하여 최대 50개의 태그를 추가합니다. Add new tag(새 태그 추가)를 선택하면 됩니다. 태그는 리소스 태그로 AWS Cloud9 환경에 연결되며, AWS CloudFormation 스택, Amazon EC2 인스턴스 및 Amazon EC2 보안 그룹과 같은 기본 리소스에 전파됩니다. 태그에 대해 자세히 알아보려면 IAM 사용 설명서의 AWS 리소스 태그를 사용한 액세스 제어와 이 설명서의 고급 정보를 참조하세요.

> **Warning**
>
> 태그를 생성한 후 이러한 태그를 업데이트하면 변경 사항이 기본 리소스에 전파되지 않습니다. 자세한 내용은 태그에 대한 고급 정보에서 기본 리소스에 태그 업데이트 전파 섹션을 참조하세요.

13. Create(생성)를 선택하여 환경을 만들면 홈 페이지로 리디렉션됩니다. 계정이 성공적으로 생성되면 AWS Cloud9 콘솔 상단에 녹색 플래시바가 나타납니다. 새 환경을 선택하고 Open in Cloud9(Cloud9에서 열기)을 선택하여 IDE를 시작할 수 있습니다.



계정이 생성되지 못하면 AWS Cloud9 콘솔 상단에 적색 플래시바가 나타납니다. 웹 브라우저,AWS 액세스 권한, 인스턴스 또는 연결된 네트워크 관련 문제 때문에 계정이 생성되지 않을 수 있습니다. AWS Cloud9 문제 해결 섹션에서 해결 방법 관련 정보를 확인할 수 있습니다.

> **Note**
>
> AWS Cloud9는 IMDSv1 및 IMDSv2를 모두 지원합니다. IMDSv1과 비교하여 향상된 보안 수준을 제공하는 IMDSv2를 채택하는 것이 좋습니다. IMDSv2의 이점에 대한 자세한 내용은 AWS 보안 블로그 섹션을 참조하십시오. IMDSv1에서 IMDSv2로의 전환에 대한 자세한 내용은 Linux 인스턴스용 Amazon EC2 사용 설명서의 인스턴스 메타데이터 서비스 버전 2 사용으로 전환 섹션을 참조하십시오.

> **Note**
>
> 환경이 프록시를 사용하여 인터넷에 액세스하는 경우, 종속 구성 요소를 설치할 수 있도록 AWS Cloud9에 프록시 세부 정보를 제공해야 합니다. 자세한 내용은 종속성을 설치하지 못함 섹션을 참조하세요.

다음 단계

2단계: IDE의 기본 사항 둘러보기

2단계: IDE의 기본 사항 둘러보기

(이전 단계: 단계 1: 환경 조성)

자습서의 이 부분에서는 AWS Cloud9 IDE를 사용하여 애플리케이션을 생성하고 테스트할 수 있는 몇 가지 방법을 소개합니다.

- editor(편집기) 창을 사용하여 코드를 생성하고 편집할 수 있습니다.
- terminal(터미널) 창 또는 Run Configuration(실행 구성) 창을 사용하여 코드를 디버깅하지 않고 실행할 수 있습니다.
- Debugger(디버거) 창을 사용하여 코드를 디버깅할 수 있습니다.

JavaScript 및 Node.js 엔진을 사용하여 이 세 작업을 수행합니다. 다른 프로그래밍 언어 사용 관련 지침은 AWS Cloud9에 대한 자습서 섹션을 참조하세요.

주제

환경 준비

JavaScript 코드를 실행 및 디버깅하는 데 필요한 대부분의 도구가 이미 설치되어 있습니다. 하지만 이 자습서를 사용하려면 추가 Node.js 패키지가 필요합니다. 패키지를 다음과 같이 설치하십시오.

1. AWS Cloud9 IDE의 맨 위에 있는 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택하거나 기존 터미널 창을 사용합니다.
2. IDE의 맨 아래에 있는 탭 중 하나인 터미널 창에 다음을 입력합니다.

   ```
   npm install readline-sync
   ```

   그 결과는 다음과 비슷합니다. `npm WARN` 메시지도 표시된다면 무시하세요.

   ```
   + readline-sync@1.4.10
   ```

```
added 1 package from 1 contributor and audited 5 packages in 0.565s
found 0 vulnerabilities
```

코드 작성

먼저 일부 코드를 작성합니다.

1. 메뉴 모음에서 File(파일)과 New File(새 파일)을 선택합니다.
2. 다음 JavaScript를 새 파일에 추가합니다.

```
var readline = require('readline-sync');
var i = 10;
var input;

console.log("Hello Cloud9!");
console.log("i is " + i);

do {
    input = readline.question("Enter a number (or 'q' to quit): ");
    if (input === 'q') {
        console.log('OK, exiting.')
    }
    else{
        i += Number(input);
        console.log("i is now " + i);
    }
} while (input != 'q');

console.log("Goodbye!");
```

3. File(파일), Save(저장)를 선택한 다음 파일을 `hello-cloud9.js`로 저장합니다.

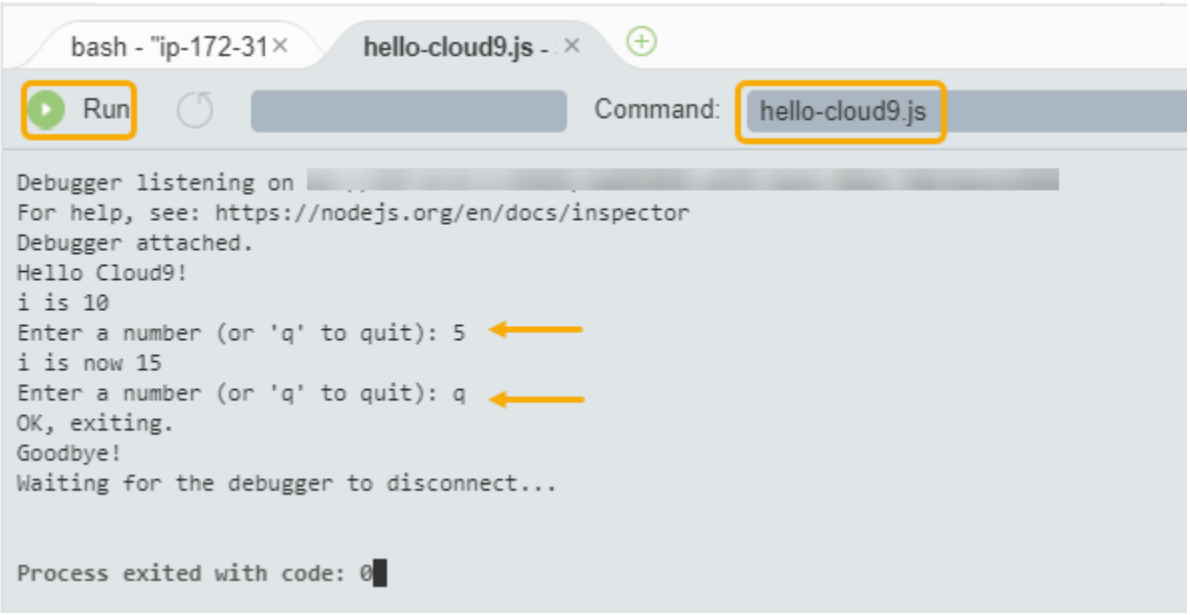
코드 실행

그런 다음 코드를 실행할 수 있습니다.

사용 중인 프로그래밍 언어에 따라 다양한 방법으로 코드를 실행할 수 있습니다. 이 자습서에서는 터미널 창 또는 [실행 구성(Run Configuration)] 창에서 실행할 수 있는 JavaScript를 사용합니다.

실행 구성(Run Configuration) 창을 사용하여 코드를 실행하려면

1. 메뉴 표시줄에서 실행, 실행 구성, 새 실행 구성을 선택합니다.
2. 새 [실행 구성(Run Configuration)] 창(IDE의 맨 아래에 있는 탭 중 하나)에서 [명령(Command)] 필드에 `hello-cloud9.js`를 입력한 다음 [실행(Run)]을 선택합니다.
3. Run Configuration(실행 구성) 프롬프트가 활성화되는지 확인한 다음 프롬프트에 번호를 입력하여 애플리케이션과 상호 작용합니다.
4. Run Configuration(실행 구성) 창에서 코드의 출력을 확인합니다. 다음과 유사하게 표시됩니다.



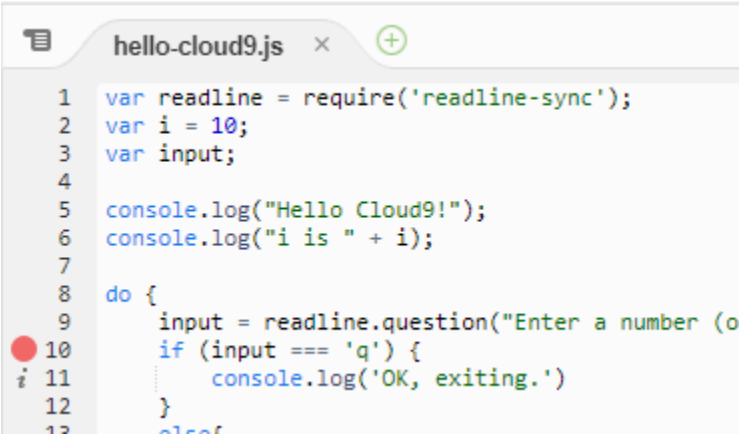
터미널 창을 사용하여 코드를 실행하려면

1. 앞에서 사용한 터미널 창으로 이동하거나 새 창을 엽니다.
2. 터미널 창의 터미널 프롬프트에 `ls`를 입력하고 코드 파일이 파일 목록에 있는지 확인합니다.
3. 프롬프트에 `node hello-cloud9.js`를 입력하여 애플리케이션을 시작합니다.
4. 프롬프트에 번호를 입력하여 애플리케이션과 상호 작용합니다.
5. 터미널 창에서 코드의 출력을 확인합니다. 다음과 유사하게 표시됩니다.

코드 디버그

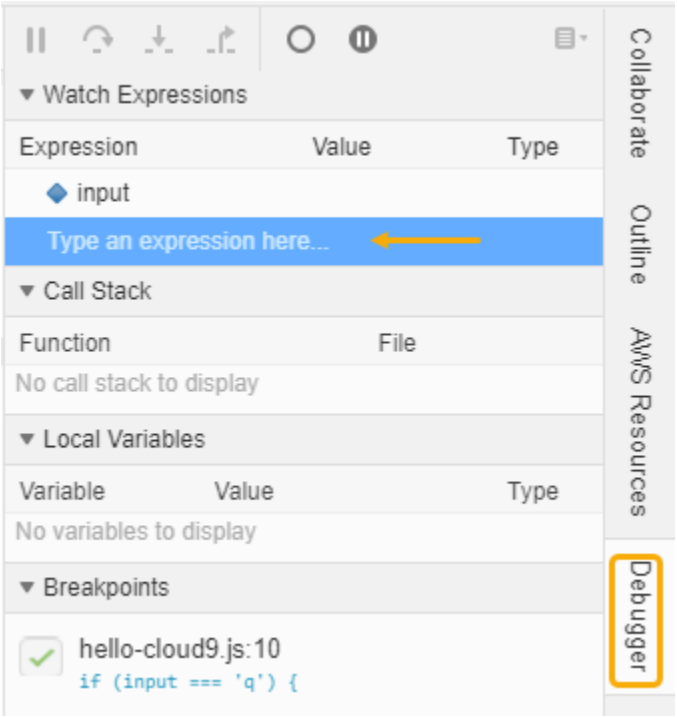
마지막으로 Debugger(디버거) 창을 사용하여 코드를 디버깅할 수 있습니다.

1. 10번 행의 옆에 있는 여백을 선택하여 코드의 10번 행(`if (input === 'q')`)에 중단점을 추가합니다. 줄 번호 옆에는 다음과 같이 빨간색 원이 표시됩니다.



2. IDE의 오른쪽에 있는 [디버거(Debugger)] 버튼을 선택하여 [디버거(Debugger)] 창을 엽니다. 또는 메뉴 모음에서 Window(창), Debugger(디버거)를 선택합니다.

   그런 다음 디버거(Debugger) 창의 조사식(Watch Expressions) 섹션에서 여기에 표현식 입력(Type an expression here)을 선택하여 `input` 변수에 조사식을 넣습니다.

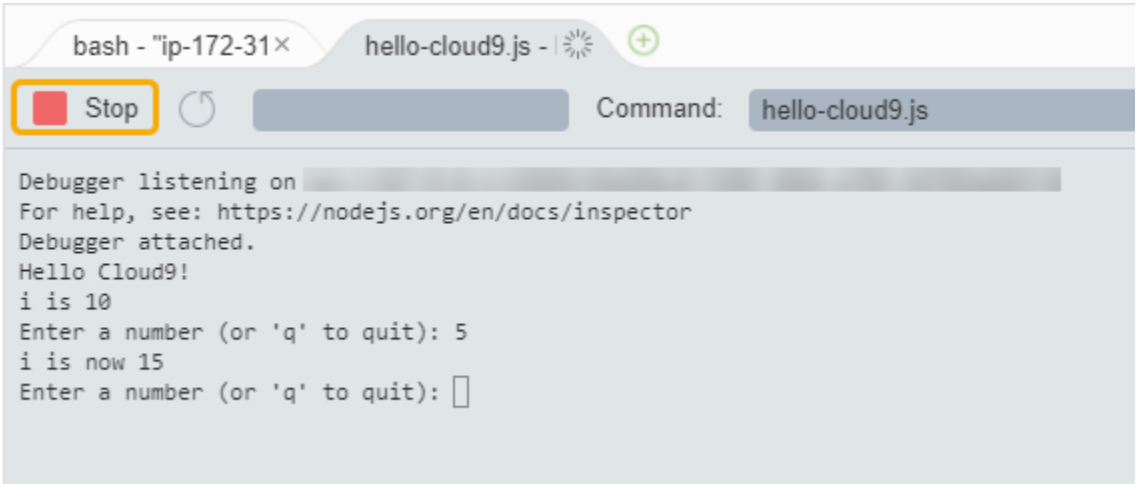


3. 앞에서 코드를 실행하는 데 사용한 Run Configuration(실행 구성) 창으로 이동합니다. Run(실행)을 선택합니다.

   또는 새 Run Configuration(구성 실행) 창을 열고 코드 실행을 시작합니다. 메뉴 표시줄에서 Run(실행), Run With(다음으로 실행), Node.js를 선택하면 됩니다.

4. Run Configuration(실행 구성) 프롬프트에 번호를 입력하고 코드가 10번 행에서 일시 중지되는지 확인합니다. Watch Expressions(조사식)에 입력한 값이 Debugger(디버거) 창에 표시됩니다.

5. Debugger(디버거) 창에서 Resume(재개)를 선택합니다. 이전 스크린샷에서 강조 표시된 파란색 화살표 아이콘입니다.
6. Run Configuration(실행 구성) 창에서 Stop(중지)을 선택하여 디버거를 중지합니다.



다음 단계

3단계: 정리

3단계: 정리

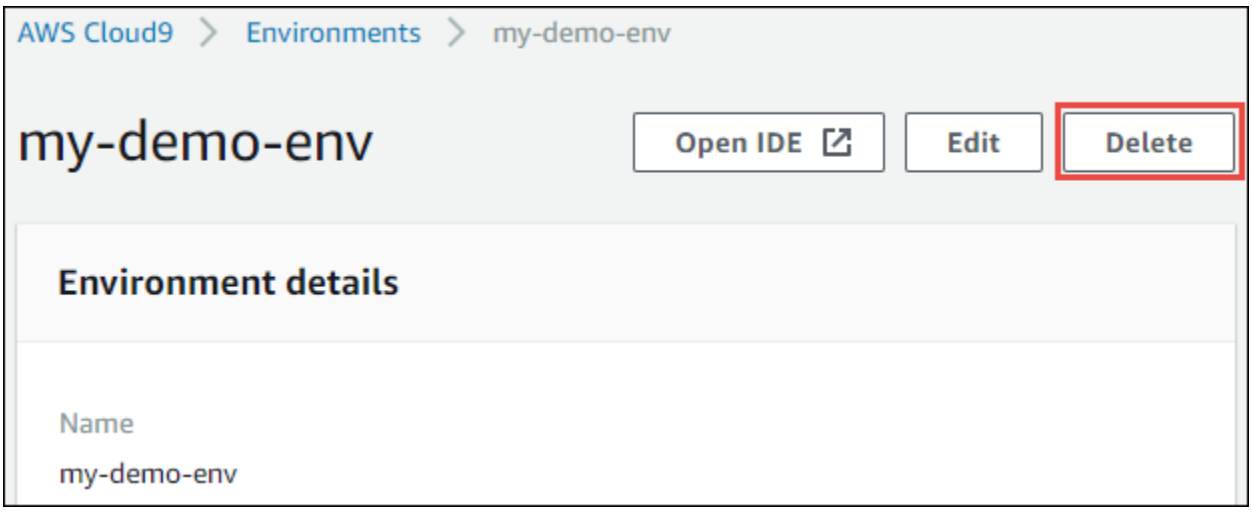
(이전 단계: 2단계: IDE의 기본 사항 둘러보기)

이 자습서와 관련된 요금이 AWS 계정에 지속적으로 부과되지 않도록 하려면 환경을 삭제합니다.

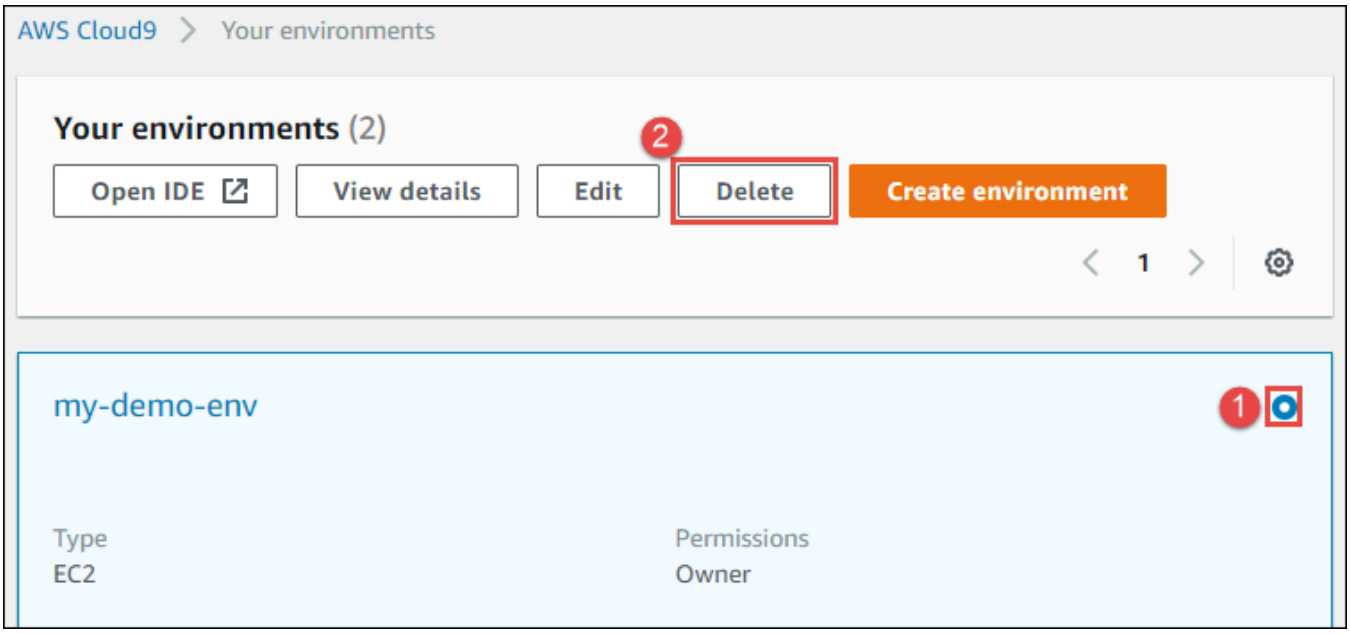
> **Warning**
>
> 삭제한 환경은 복구할 수 없습니다.

AWS Cloud9 콘솔을 사용하여 환경 삭제

1. 대시보드를 열려면 IDE의 메뉴 모음에서 [AWS Cloud9], [대시보드로 이동(Go To Your Dashboard)]을 선택합니다.
2. 다음 중 하나를 수행합니다.
   - my-demo-environment 카드 내부에서 제목을 선택한 다음 Delete(삭제)를 선택합니다.



- my-demo-environment 카드를 선택한 다음 Delete(삭제)를 선택합니다.



3. Delete(삭제) 대화 상자에 `Delete`를 입력한 후 Delete(삭제)를 선택합니다. 삭제 작업에 몇 분 정도 걸립니다.

> **Note**
>
> 이 자습서를 정확히 따랐다면 환경이 EC2 환경이고 AWS Cloud9이 해당 환경에 연결된 Amazon EC2 인스턴스를 종료합니다.
> 하지만 자습서를 따르는 대신 SSH 환경을 사용하고 환경이 Amazon EC2 인스턴스에 연결되었으면 AWS Cloud9이 인스턴스를 종료하지 않습니다. 나중에 해당 인스턴스를 종료하지 않으

면, AWS 계정에 해당 인스턴스와 관련된 Amazon EC2에 대한 요금이 계속해서 부과될 수 있습니다.

다음 단계

관련 정보

관련 정보

다음은 자습서: Hello AWS Cloud9(콘솔)에 대한 추가 정보입니다.

- EC2 환경을 생성할 때 환경에는 기본적으로 샘플 코드가 포함되지 않습니다. 샘플 코드와 함께 환경을 생성하려면 다음 주제 중 하나를 참조하세요.
  - AWS Cloud9 통합 개발 환경(IDE)의 Amazon Lightsail 인스턴스스
  - AWS Cloud9 통합 개발 환경(IDE)의 AWS CodeStar 프로젝트로 작업
- AWS Cloud9 개발 환경을 생성하는 중에 Amazon EC2 인스턴스를 생성하도록 AWS Cloud9에 지시했습니다. AWS Cloud9은 인스턴스를 생성한 다음 환경을 인스턴스에 연결했습니다. 기존 클라우드 컴퓨팅 인스턴스나 (SSH 환경이라고 하는) 자체 서버를 사용할 수도 있습니다. 자세한 내용은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.

다음 단계(선택 사항)

다음 주제 중 하나 또는 전부를 탐색하여 AWS Cloud9를 계속해서 익힙니다.

작업	다음 주제 참조
환경에서 할 수 있는 일을 자세히 알아봅니다.	환경 관련 작업 AWS Cloud9
다른 컴퓨터 언어를 사용해 보십시오.	AWS Cloud9에 대한 자습서
AWS Cloud9 IDE에 대해 알아봅니다.	AWS Cloud9 IDE 둘러보기의 IDE 작업
텍스트 채팅 지원을 통해 실시간으로 새 환경을 사용할 수 있도록 다른 사람을 초대합니다.	AWS Cloud9의 공유 환경 작업
SSH 환경 생성 AWS Cloud9이 자동으로 생성한 Amazon EC2 인스턴스 대신, 사용자가 직접 생	에서 환경 만들기 AWS Cloud9 및 SSH 환경 호스트 요구 사항

작업	다음 주제 참조
성한 클라우드 컴퓨팅 인스턴스 또는 서버를 사용하는 환경입니다.	
AWS Lambda 함수와 서버리스 애플리케이션에서 AWS 도구 키트를 사용하여 코드를 생성, 실행 및 디버그합니다.	AWS 도구 키트를 사용한 AWS Lambda 함수 작업
AWS Cloud9 및 Amazon Lightsail 함께 사용.	AWS Cloud9 통합 개발 환경(IDE)의 Amazon Lightsail 인스턴스스
AWS Cloud9 및 AWS CodeStar 함께 사용.	AWS Cloud9 통합 개발 환경(IDE)의 AWS CodeStar 프로젝트로 작업
AWS Cloud9 및 AWS CodePipeline 함께 사용.	AWS Cloud9 통합 개발 환경(IDE)의 AWS CodePipeline 작업
AWS CLI, AWS CloudShell, AWS CodeCommit 및 AWS Cloud Development Kit(AWS CDK), GitHub 또는 Amazon DynamoDB 및 Node.js, Python 또는 기타 프로그래밍 언어와 함께 AWS Cloud9을 사용합니다.	AWS Cloud9에 대한 자습서
AWS RoboMaker에서 지능형 로봇 애플리케이션용 코드로 작업합니다.	AWS RoboMaker 개발자 가이드의 AWS Cloud9으로 개발

커뮤니티에서 AWS Cloud9에 대한 도움을 받으려면 AWS Cloud9 토론 포럼을 참조하세요. (이 포럼에 들어갈 때 AWS에서 로그인을 요청할 수 있습니다.)

AWS에서 직접 AWS Cloud9에 대한 도움을 받으려면 AWS Support 페이지의 지원 옵션을 참조하세요.

자습서: Hello AWS Cloud9(CLI)

이 자습서는 AWS Cloud9을 처음 접하는 사용자를 위한 것이며, AWS Command Line Interface(AWS CLI)를 사용합니다. CLI를 사용하면 그래픽 사용자 인터페이스 대신 명령줄을 사용하여 필요한 리소스를 설정 및 삭제할 수 있습니다.

이 자습서에서는 AWS Cloud9 개발 환경을 설정한 다음 AWS Cloud9 IDE를 사용하여 첫 번째 애플리케이션을 코딩, 실행 및 디버깅합니다.

이 자습서를 완료하는 데 1시간 가량 걸립니다.

> **Warning**
>
> 이 자습서를 완료하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금을 참조하세요.

필수 조건

이 자습서를 완료하려면 먼저 AWS Cloud9 설정의 단계를 완료해야 합니다.

단계

- 단계 1: 환경 조성
- 2단계: IDE의 기본 사항 둘러보기
- 3단계: 정리
- 관련 정보

단계 1: 환경 조성

(자습서: Hello AWS Cloud9(CLI)의 첫 단계)

이 단계에서는 AWS CLI를 사용하여 AWS Cloud9 개발 환경을 생성합니다.

AWS Cloud9에서 개발 환경 또는 환경은 개발 프로젝트의 파일을 저장하고 도구를 실행하여 애플리케이션을 개발하는 곳입니다. 이 자습서에서는 EC2 환경을 생성하고 이 환경에서 파일과 도구를 작업합니다.

AWS CLI를 사용한 EC2 환경 생성

1. 아직 하지 않은 경우 AWS CLI를 설치하고 구성합니다. 이를 위해 AWS Command Line Interface 사용 설명서에서 다음 지침을 참조하세요.

   - AWS 명령줄 인터페이스 설치

- 빠른 구성

다음 중 하나에 대한 자격 증명을 사용하여 AWS CLI를 구성할 수 있습니다.

- 팀 설정 대상 AWS Cloud9에서 생성한 IAM 사용자.
- 여러 계정의 여러 사용자에 대해 AWS Cloud9 리소스를 사용하여 정기적으로 작업하는 경우 AWS 계정의 IAM 관리자. IAM 관리자로 AWS CLI를 구성할 수 없으면 AWS 계정 관리자에게 문의하세요. 자세한 내용은 IAM 사용 설명서에서 첫 번째 IAM 관리자 및 그룹 만들기를 참조하세요.
- 항상 혼자만 AWS 계정을 사용하고 환경을 다른 사람과 공유할 필요가 없는 경우에만 AWS 계정 루트 사용자. AWS 보안 모범 사례에 부합하지 않으므로 이 옵션은 사용하지 않는 것이 좋습니다. 자세한 내용은 Amazon Web Services 일반 참조에서 AWS 계정에 대한 액세스 키 생성, 비활성화 및 삭제를 참조하세요.
- 기타 옵션은 AWS 계정 관리자 또는 강의실 강사에게 문의하십시오.

2. 다음 AWS Cloud9 명령에서는 `--region` 및 `--subnet-id`에 대한 값을 제공합니다. 그런 다음 명령을 실행하고 나중에 정리할 수 있도록 `"environmentId"` 값을 기록해 둡니다.

```
aws cloud9 create-environment-ec2 --name my-demo-environment --description "This
 environment is for the AWS Cloud9 tutorial." --instance-type t2.micro --image-id
 resolve:ssm:/aws/service/cloud9/amis/amazonlinux-2-x86_64 --region MY-REGION --
connection-type CONNECT_SSM --subnet-id subnet-12a3456b
```

앞의 명령에서:

- `--name`은 환경의 이름을 나타냅니다. 이 자습서에서는 `my-demo-environment` 이름을 사용합니다.
- `--description`은 환경의 선택적 설명을 나타냅니다.
- `--instance-type`는 AWS Cloud9이 시작하고 새 환경에 연결하는 Amazon EC2 인스턴스의 유형을 나타냅니다. 이 예제에서는 `t2.micro`를 지정합니다. 이 항목은 RAM과 vCPU가 비교적 낮지만 이 자습서에는 충분합니다. 더 많은 RAM 및 vCPU가 있는 인스턴스 유형을 지정하면 AWS 계정에 Amazon EC2의 추가 비용이 발생할 수 있습니다. 사용 가능한 인스턴스 유형의 목록은 AWS Cloud9 콘솔의 환경 생성 마법사를 참조하세요.
- `--image-id`는 EC2 인스턴스를 생성하는 데 사용되는 Amazon Machine Image(AMI)의 식별자를 지정합니다. 인스턴스에 대한 AMI를 선택하려면 유효한 AMI 별칭 또는 유효한 AWS Systems Manager(SSM) 경로를 지정해야 합니다. 위의 예에서는 Amazon Linux 2 AMI에 대한 SSM 경로가 지정되었습니다.

자세한 내용은 AWS CLI명령 참조서의 create-environment-ec 2를 참조하십시오.

- `--region`은 AWS Cloud9이 환경을 생성할 AWS 리전의 ID를 나타냅니다. 사용할 수 있는 AWS 리전 목록은 Amazon Web Services 일반 참조에서 AWS Cloud9 섹션을 참조하세요.
- `--connection-type CONNECT_SSM`은 AWS Cloud9이 Systems Manager를 통해 Amazon EC2 인스턴스에 연결하도록 지정합니다. 이 옵션을 사용하면 인스턴스에 대한 인바운드 트래픽이 허용되지 않습니다. 자세한 설명은 AWS Systems Manager를 사용하여 수신하지 않는 EC2 인스턴스에 액세스 섹션을 참조하세요.

  > **Note**
  >
  > 이 옵션을 사용할 때 아직 생성되지 않은 경우에는 `AWSCloud9SSMAccessRole` 서비스 역할과 `AWSCloud9SSMInstanceProfile`을 생성해야 합니다. 자세한 설명은 AWS CLI를 사용하여 Systems Manager의 인스턴스 프로파일 관리 섹션을 참조하세요.

- `--subnet-id`는 AWS Cloud9에서 사용할 서브넷을 나타냅니다. `subnet-12a3456b`를 Amazon Virtual Private Cloud(VPC)의 서브넷 ID로 바꿉니다. 이 서브넷은 AWS Cloud9과 호환 가능해야 합니다. 자세한 내용은 개발 환경을 위한 AWS Cloud9 VPC 설정에서 VPC 및 기타 VPC 리소스 생성 섹션을 참조하세요.
- AWS Cloud9은 환경의 IDE에 연결된 모든 웹 브라우저 인스턴스가 닫히면 환경의 Amazon EC2 인스턴스를 종료합니다. 이 기간을 구성하려면 `--automatic-stop-time-minutes` 및 분을 추가합니다. 기간이 짧으면 AWS 계정에 청구되는 요금이 적어질 수 있습니다. 마찬가지로, 긴 시간은 더 많은 요금이 발생할 수 있습니다.
- 기본적으로 이 명령을 호출하는 엔터티는 환경을 소유합니다. 이 설정을 변경하려면 `--owner-id`와 소유 엔터티의 Amazon 리소스 이름(ARN)을 추가합니다.

3. 이 명령을 성공적으로 실행한 수 새로 생성된 환경의 AWS Cloud9 IDE를 엽니다. 이렇게 하려면 단원을 참조하세요AWS Cloud9에서 환경 열기 그런 다음 이 주제로 돌아와서 2단계: IDE의 기본 사항 둘러보기을(를) 계속 진행하고 AWS Cloud9 IDE를 사용하여 새 환경을 작업하는 방법을 알아봅니다.

   환경을 열려고 하지만 최소 5분 후에도 AWS Cloud9에서 IDE가 표시되지 않는 경우 웹 브라우저, AWS 액세스 권한, 인스턴스 또는 연결된 VPC에 문제가 있을 수 있습니다. 가능한 해결 방법은 환경을 열 수 없음 섹션을 참조하세요.

다음 단계

2단계: IDE의 기본 사항 둘러보기

2단계: IDE의 기본 사항 둘러보기

(이전 단계: 단계 1: 환경 조성)

자습서의 이 부분에서는 AWS Cloud9 IDE를 사용하여 애플리케이션을 생성하고 테스트할 수 있는 몇 가지 방법을 소개합니다.

- editor(편집기) 창을 사용하여 코드를 생성하고 편집할 수 있습니다.
- terminal(터미널) 창 또는 Run Configuration(실행 구성) 창을 사용하여 코드를 디버깅하지 않고 실행할 수 있습니다.
- Debugger(디버거) 창을 사용하여 코드를 디버깅할 수 있습니다.

JavaScript 및 Node.js 엔진을 사용하여 이 세 가지 작업을 수행합니다. 다른 프로그래밍 언어 사용 관련 지침은 AWS Cloud9에 대한 자습서 섹션을 참조하세요.

주제

환경 준비

JavaScript 코드를 실행하고 디버깅하는 데 필요한 대부분의 도구가 이미 설치되어 있습니다. 하지만 이 자습서를 사용하려면 추가 Node.js 패키지가 필요합니다. 패키지를 다음과 같이 설치하십시오.

1. AWS Cloud9 IDE의 맨 위에 있는 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택하거나 기존 터미널 창을 사용합니다.
2. IDE의 맨 아래에 있는 탭 중 하나인 터미널 창에 다음을 입력합니다.

   ```
   npm install readline-sync
   ```

   그 결과는 다음과 비슷합니다. `npm WARN` 메시지도 표시된다면 무시하세요.

   ```
   + readline-sync@1.4.10
   ```

```
added 1 package from 1 contributor and audited 5 packages in 0.565s
found 0 vulnerabilities
```

코드 작성

먼저 일부 코드를 작성합니다.

1. 메뉴 모음에서 File(파일)과 New File(새 파일)을 선택합니다.
2. 새 파일에 다음을 JavaScript 추가합니다.

```
var readline = require('readline-sync');
var i = 10;
var input;

console.log("Hello Cloud9!");
console.log("i is " + i);

do {
    input = readline.question("Enter a number (or 'q' to quit): ");
    if (input === 'q') {
        console.log('OK, exiting.')
    }
    else{
        i += Number(input);
        console.log("i is now " + i);
    }
} while (input != 'q');

console.log("Goodbye!");
```

3. File(파일), Save(저장)를 선택한 다음 파일을 `hello-cloud9.js`로 저장합니다.

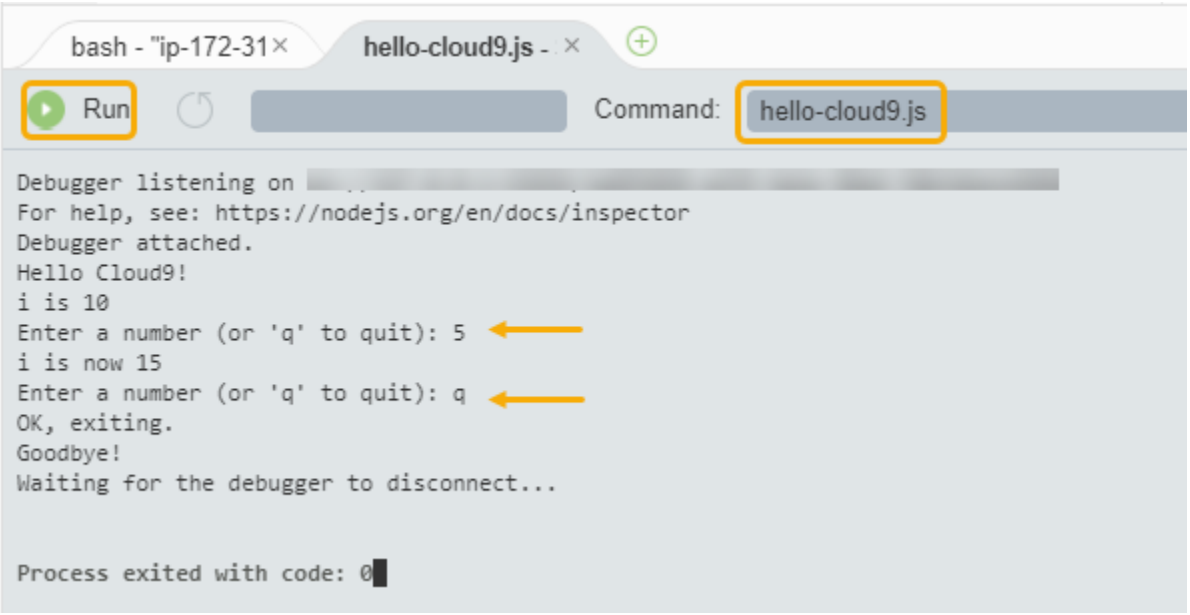
코드 실행

그런 다음 코드를 실행할 수 있습니다.

사용 중인 프로그래밍 언어에 따라 다양한 방법으로 코드를 실행할 수 있습니다. 이 자습서에서는 터미널 창이나 구성 실행 창을 사용하여 실행할 수 있는 방법을 사용합니다 JavaScript.

실행 구성(Run Configuration) 창을 사용하여 코드를 실행하려면

1. 메뉴 표시줄에서 실행, 실행 구성, 새 실행 구성을 선택합니다.
2. 새 [실행 구성(Run Configuration)] 창(IDE의 맨 아래에 있는 탭 중 하나)에서 [명령(Command)] 필드에 `hello-cloud9.js`를 입력한 다음 [실행(Run)]을 선택합니다.
3. Run Configuration(실행 구성) 프롬프트가 활성화되는지 확인한 다음 프롬프트에 번호를 입력하여 애플리케이션과 상호 작용합니다.
4. Run Configuration(실행 구성) 창에서 코드의 출력을 확인합니다. 다음과 유사하게 표시됩니다.



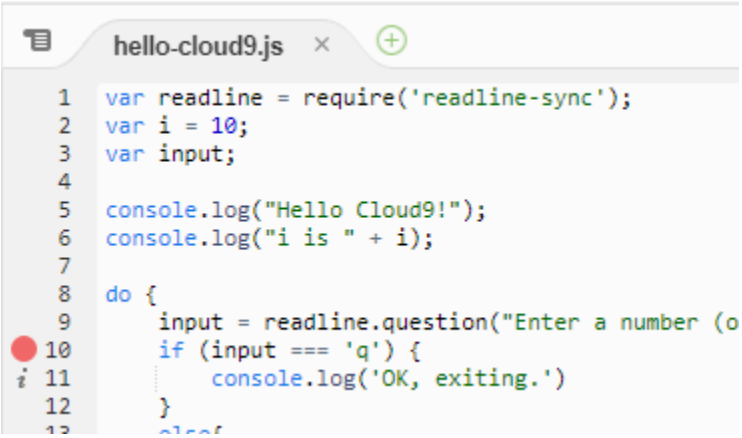
터미널 창을 사용하여 코드를 실행하려면

1. 앞에서 사용한 터미널 창으로 이동하거나 새 창을 엽니다.
2. 터미널 창의 터미널 프롬프트에 `ls`를 입력하고 코드 파일이 파일 목록에 있는지 확인합니다.
3. 프롬프트에 `node hello-cloud9.js`를 입력하여 애플리케이션을 시작합니다.
4. 프롬프트에 번호를 입력하여 애플리케이션과 상호 작용합니다.
5. 터미널 창에서 코드의 출력을 확인합니다. 다음과 유사하게 표시됩니다.

코드 디버그

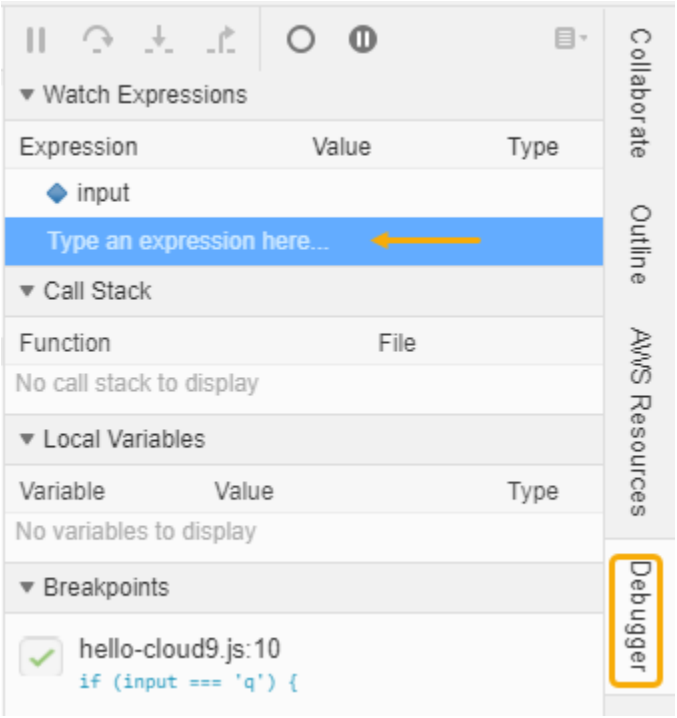
마지막으로 Debugger(디버거) 창을 사용하여 코드를 디버깅할 수 있습니다.

1. 10번 행의 옆에 있는 여백을 선택하여 코드의 10번 행(`if (input === 'q')`)에 중단점을 추가합니다. 줄 번호 옆에는 다음과 같이 빨간색 원이 표시됩니다.



2. IDE의 오른쪽에 있는 [디버거(Debugger)] 버튼을 선택하여 [디버거(Debugger)] 창을 엽니다. 또는 메뉴 모음에서 Window(창), Debugger(디버거)를 선택합니다.

   그런 다음 디버거(Debugger) 창의 조사식(Watch Expressions) 섹션에서 여기에 표현식 입력(Type an expression here)을 선택하여 `input` 변수에 조사식을 넣습니다.

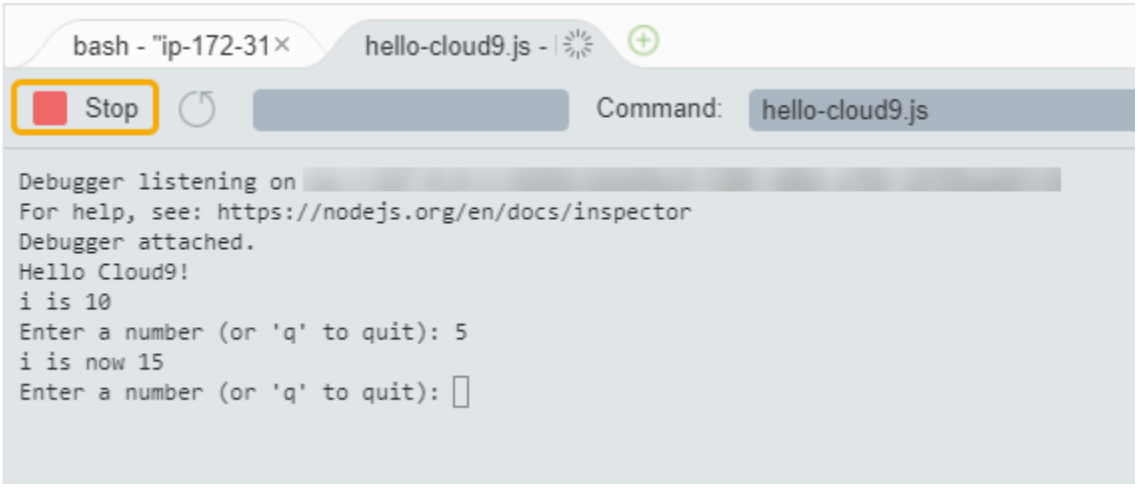


3. 앞에서 코드를 실행하는 데 사용한 Run Configuration(실행 구성) 창으로 이동합니다. Run(실행)을 선택합니다.

   또는 새 Run Configuration(구성 실행) 창을 열고 코드 실행을 시작합니다. 메뉴 표시줄에서 Run(실행), Run With(다음으로 실행), Node.js를 선택하면 됩니다.

4. Run Configuration(실행 구성) 프롬프트에 번호를 입력하고 코드가 10번 행에서 일시 중지되는지 확인합니다. Watch Expressions(조사식)에 입력한 값이 Debugger(디버거) 창에 표시됩니다.

5. Debugger(디버거) 창에서 Resume(재개)를 선택합니다. 이전 스크린샷에서 강조 표시된 파란색 화살표 아이콘입니다.
6. Run Configuration(실행 구성) 창에서 Stop(중지)을 선택하여 디버거를 중지합니다.



다음 단계

3단계: 정리

3단계: 정리

(이전 단계: 2단계: IDE의 기본 사항 둘러보기)

이 자습서와 관련된 요금이 AWS 계정에 지속적으로 부과되지 않도록 하려면 환경을 삭제해야 합니다.

> **Warning**
> 환경 삭제는 실행 취소할 수 없습니다.

AWS CLI를 사용하여 환경 삭제

1. 삭제할 환경의 ID를 지정하여 AWS Cloud9 `delete-environment` 명령을 실행합니다.

```
aws cloud9 delete-environment --region MY-REGION --environment-id
 12a34567b8cd9012345ef67abcd890e1
```

이전 명령에서 `MY-REGION`을 환경이 생성된 AWS 리전으로 바꾸고, `12a34567b8cd9012345ef67abcd890e1`을 삭제할 환경의 ID로 바꿉니다.

환경을 생성할 때 ID를 저장하지 않은 경우 AWS Cloud9 콘솔을 사용하여 ID를 확인할 수 있습니다. 콘솔에서 환경의 이름을 선택한 다음 Environment ARN(환경 ARN)의 마지막 부분을 확인합니다.

2. 이 자습서를 위해 Amazon VPC 만들었는데 더 이상 필요하지 않은 경우 Amazon VPC 콘솔(https://console.aws.amazon.com/vpc)을 사용하여 VPC를 삭제합니다.

다음 단계

관련 정보

관련 정보

다음은 자습서: Hello AWS Cloud9(CLI)에 대한 추가 정보입니다.

- EC2 환경을 생성할 때 환경에는 기본적으로 샘플 코드가 포함되지 않습니다. 샘플 코드와 함께 환경을 생성하려면 다음 주제 중 하나를 참조하세요.
  - AWS Cloud9 통합 개발 환경(IDE)의 Amazon Lightsail 인스턴스스
  - AWS Cloud9 통합 개발 환경(IDE)의 AWS CodeStar 프로젝트로 작업
- AWS Cloud9 개발 환경을 생성하는 중에 Amazon EC2 인스턴스를 생성하도록 AWS Cloud9에 지시했습니다. AWS Cloud9은 인스턴스를 생성한 다음 환경을 인스턴스에 연결했습니다. 기존 클라우드 컴퓨팅 인스턴스나 (SSH 환경이라고 하는) 자체 서버를 사용할 수도 있습니다. 자세한 설명은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.

다음 단계(선택 사항)

다음 주제 중 하나 또는 전부를 탐색하여 AWS Cloud9를 계속해서 익힙니다.

작업	다음 주제 참조
환경에서 할 수 있는 일을 자세히 알아봅니다.	환경 관련 작업 AWS Cloud9
다른 컴퓨터 언어를 사용해 보십시오.	AWS Cloud9에 대한 자습서
AWS Cloud9 IDE에 대해 알아봅니다.	IDE 작업의 AWS Cloud9 IDE 둘러보기
텍스트 채팅 지원을 통해 실시간으로 새 환경을 사용할 수 있도록 다른 사람을 초대합니다.	AWS Cloud9의 공유 환경 작업

작업	다음 주제 참조
SSH 환경 생성 AWS Cloud9이 자동으로 생성한 Amazon EC2 인스턴스 대신, 사용자가 직접 생성한 클라우드 컴퓨팅 인스턴스 또는 서버를 사용하는 환경입니다.	에서 환경 만들기 AWS Cloud9 및 SSH 환경 호스트 요구 사항
AWS Lambda 함수와 서버리스 애플리케이션에서 AWS 도구 키트를 사용하여 코드를 생성, 실행 및 디버그합니다.	AWS 도구 키트를 사용한 AWS Lambda 함수 작업
Amazon Lightsail에 AWS Cloud9 사용	AWS Cloud9 통합 개발 환경(IDE)의 Amazon Lightsail 인스턴스스
AWS Cloud9 및 AWS CodeStar 함께 사용.	AWS Cloud9 통합 개발 환경(IDE)의 AWS CodeStar 프로젝트로 작업
AWS Cloud9 및 AWS CodePipeline 함께 사용.	AWS Cloud9 통합 개발 환경(IDE)의 AWS CodePipeline 작업
AWS CLI,,, AWS 클라우드 개발 키트 (AWSCDK)AWS CodeCommit, Amazon DynamoDB GitHub, Node.js, Python 또는 기타 프로그래밍 AWS Cloud9 언어와 함께 사용하십시오. AWS CloudShell	AWS Cloud9에 대한 자습서
에서 지능형 로보틱스 애플리케이션을 위한 코드를 사용해 보세요. AWS RoboMaker	AWS RoboMaker 개발자 안내서를 사용하여 AWS Cloud9 개발하기

커뮤니티에서 AWS Cloud9에 대한 도움을 받으려면 AWS Cloud9 토론 포럼을 참조하세요. (이 포럼에 들어갈 때 AWS에서 로그인을 요청할 수 있습니다.)

AWS에서 직접 AWS Cloud9에 대한 도움을 받으려면 AWS Support 페이지의 지원 옵션을 참조하세요.

환경 관련 작업 AWS Cloud9

개발 환경은 프로젝트 파일을 저장하고 도구를 실행하여 애플리케이션을 개발하는 곳입니다. AWS Cloud9

AWS Cloud9 EC2 환경과 SSH 환경이라는 두 가지 유형의 개발 환경을 제공합니다. 이들 개발 환경 간의 주요 유사성과 차이를 이해하려면 AWS Cloud9에서 EC2 환경과 SSH 환경 비교 섹션을 참조하세요.

다음 주제 중 하나 이상을 읽고 환경을 활용하는 방법을 알아보십시오. AWS Cloud9

주제

- 에서 환경 만들기 AWS Cloud9
- AWS Systems Manager를 사용하여 수신하지 않는 EC2 인스턴스에 액세스
- AWS Cloud9에서 환경 열기
- AWS Cloud9의 환경에서 AWS 서비스 호출
- AWS Cloud9에서 환경 설정 변경
- AWS Cloud9의 공유 환경 작업
- 환경 이동 및 Amazon EBS 볼륨 크기 조정 또는 암호화
- AWS Cloud9에서 환경 삭제

에서 환경 만들기 AWS Cloud9

AWS Cloud9 개발 환경을 만들려면 사용 계획에 따라 제공된 절차 중 하나를 따르십시오 AWS Cloud9.

무엇을 선택해야 할지 확실하지 않으면 EC2 환경 생성하는 것이 좋습니다.

빠른 설정을 위해 EC2 환경을 생성하십시오. AWS Cloud9 에서 새 Amazon EC2 인스턴스를 자동으로 생성하고 설정합니다. AWS 계정 AWS Cloud9 또한 새 인스턴스를 환경에 자동으로 연결합니다.

개발 환경 간의 주요 유사성과 차이를 이해하려면 AWS Cloud9에서 EC2 환경과 SSH 환경 비교 섹션을 참조하세요.

소스 코드 제공자	개발 환경 호스트 제공자	관련 절차
사용자	AWS Cloud9	EC2 환경 생성
사용자	사용자	SSH 환경 생성
Amazon Lightsail 또는 사용자	사용자(Lightsail 사용)	AWS Cloud9 통합 개발 환경(IDE)의 Amazon Lightsail 인스턴스
AWS CodeStar 또는 사용자	AWS Cloud9 (사용 AWS CodeStar)	AWS Cloud9 통합 개발 환경(IDE)의 AWS CodeStar 프로젝트로 작업
사용자(AWS CodePipeline 사용)	AWS Cloud9 아니면 당신	EC2 또는 SSH 환경 및 AWS Cloud9 통합 개발 환경(IDE)의 AWS CodePipeline 작업 생성
사용자(AWS CodeCommit 사용)	AWS Cloud9 아니면 당신	AWS Cloud9용 AWS CodeCommit 자습서
사용자(GitHub 사용)	AWS Cloud9 아니면 당신	EC2 또는 SSH 환경을 생성하고 Git 패널 인터페이스 사용

주제

EC2 환경 생성

이 절차에서는 EC2 환경과 새 Amazon EC2 인스턴스를 AWS Cloud9 만들고 환경을 이 인스턴스에 연결합니다. AWS Cloud9 필요에 따라 인스턴스를 시작, 중지 및 재시작하는 등 이 인스턴스의 수명 주기를 관리합니다. 이 환경을 삭제하면 AWS Cloud9 은 이 인스턴스를 자동으로 종료합니다.

AWS Cloud9 콘솔에서 또는 코드를 사용하여 AWS Cloud9 EC2 개발 환경을 만들 수 있습니다.

> **Note**
>
> 이 절차를 완료하면 요금이 부과될 수 있습니다. AWS 계정여기에는 Amazon EC2 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금을 참조하세요.

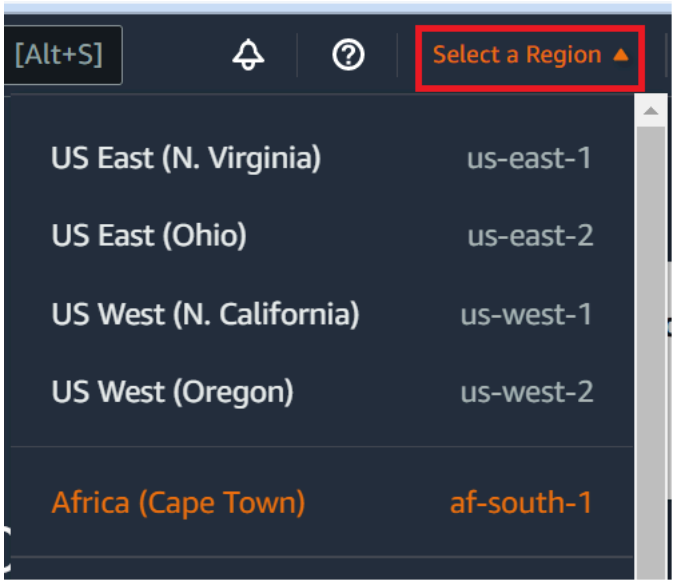
> **Warning**
>
> AWS Control Tower 사전 예방 제어 AWS Cloud9 CT.EC2.PR.8에 호환성 문제가 있습니다. 이 컨트롤이 활성화되면 AWS Cloud9에서 EC2 환경을 만들 수 없습니다. 이 문제에 대한 자세한 내용은 문제 해결을 참조하십시오. AWS Cloud9

필수 조건

AWS Cloud9 콘솔에 로그인하고 환경을 만들 수 AWS Cloud9 설정 있도록 의 단계를 완료하십시오.

콘솔을 사용한 EC2 환경 생성

1. AWS Cloud9 콘솔에 로그인:
   - 본인을 사용하는 사람이 AWS 계정 본인뿐이거나 IAM AWS 계정사용자인 경우 https://console.aws.amazon.com/cloud9/ 으로 이동하십시오.
   - 조직에서 사용하는 AWS IAM Identity Center경우 AWS 계정 관리자에게 로그인 지침을 문의하세요.
   - 교실의 학생인 경우 로그인 지침은 강사에게 문의하세요.
2. AWS Cloud9 콘솔에 로그인한 후 상단 탐색 표시줄에서 환경을 만들 항목을 AWS 리전 선택합니다. 사용 가능한 목록은 AWS 리전AWS Cloud9을 참조하십시오 AWS 일반 참조.



3. 여기에 나와 있는 위치 중 한 곳에서 큰 [환경 생성(Create environment)] 버튼을 선택합니다.

   AWS Cloud9 환경이 아직 없는 경우 시작 페이지에 버튼이 표시됩니다.

   이미 AWS Cloud9 환경이 있는 경우 버튼은 다음과 같이 표시됩니다.

4. Create environment(환경 생성) 페이지의 Name(이름)에 환경의 이름을 입력합니다.
5. 환경에 설명을 추가하려면 Description(설명) 필드에 내용을 입력합니다.
6. Environment type(환경 유형)에서 New EC2 instance(새 EC2 인스턴스)를 선택하여 Amazon EC2 환경을 만듭니다.

   - New EC2 instance새 (EC2 인스턴스) - AWS Cloud9 이 SSH를 통해 직접 연결할 수 있는 새 Amazon EC2 인스턴스를 시작합니다. Systems Manager를 사용하여 새 Amazon EC2 인스턴스와 상호 작용할 수 있습니다. 자세한 내용은 AWS Systems Manager를 사용하여 수신하지 않는 EC2 인스턴스에 액세스 섹션을 참조하세요.

- Existing compute(기존 컴퓨팅) - SSH 로그인 세부 정보가 필요한 기존 Amazon EC2 인스턴스를 시작합니다. 이 경우 Amazon EC2 인스턴스에 인바운드 보안 그룹 규칙이 있어야 합니다.
  - Existing compute(기존 컴퓨팅) 옵션을 선택하면 서비스 역할이 자동으로 생성됩니다. 설정 화면 하단에 표시되는 메모에서 서비스 역할의 이름을 확인할 수 있습니다.

> **Note**
>
> 기존 컴퓨팅을 사용하는 Amazon EC2 인스턴스를 사용하여 만든 AWS Cloud9 환경에서는 자동 종료 기능을 사용할 수 없습니다.

> **Warning**
>
> 환경의 Amazon EC2 인스턴스를 생성하면 AWS 계정 에 Amazon EC2 요금이 발생할 수 있습니다. Systems Manager를 사용하여 EC2 인스턴스에 대한 연결을 관리하는 데 따른 추가 비용은 없습니다.

7. Instance type(인스턴스 유형)에는 수행하려는 작업 유형에 필요하다고 생각되는 양의 RAM 및 vCPU가 있는 인스턴스 유형을 선택합니다.

> **Warning**
>
> RAM 및 vCPU가 더 많은 인스턴스 유형을 선택하면 AWS 계정 Amazon EC2에 대한 추가 요금이 부과될 수 있습니다. 워크로드에 적합한 인스턴스 유형에 대한 자세한 내용은 Amazon EC2 인스턴스 유형 페이지를 참조하세요.

8. 플랫폼에서는 아마존 리눅스 2023, 아마존 리눅스 2 또는 우분투 22.04 LTS 중에서 원하는 Amazon EC2 인스턴스 유형을 선택합니다. AWS Cloud9 인스턴스를 생성한 다음 환경을 인스턴스에 연결합니다.

> **Important**
>
> EC2 환경에 대해 Amazon Linux 2023 옵션을 선택하는 것이 좋습니다. Amazon Linux 2023 AMI는 안전하고 안정적인 고성능 런타임 환경을 제공할 뿐만 아니라 2024년까지 장기적인 지원을 제공합니다.

자세한 내용은 AL2023 페이지를 참조하십시오.

9. Timeout(제한 시간)을 선택합니다. 이 옵션은 AWS Cloud9 이 자동 최대 절전 모드로 전환되기 전까지 비활성 상태를 유지하는 시간을 결정합니다. 해당 환경의 IDE에 연결된 모든 웹 브라우저 인스턴스가 닫히면 지정된 시간만큼 AWS Cloud9 기다린 다음 해당 환경의 Amazon EC2 인스턴스를 종료합니다.

   Warning

   긴 기간을 선택할수록 AWS 계정에 더 많은 요금이 발생할 수 있습니다.

10. Network settings(네트워크 설정) 패널에서, 환경에 액세스하는 방법으로 다음 두 가지 옵션 중 하나를 선택합니다.

    - AWS Systems Manager (SSM) - 이 메서드는 인바운드 포트를 열지 않고 SSM을 사용하여 환경에 액세스합니다.

    - SSH(Secure Shell) – 이 방법은 SSH를 사용하여 환경에 액세스하며 열린 인바운드 포트가 필요합니다.

11. VPC 설정을 선택하여 환경에 맞는 Amazon Virtual Private 클라우드 및 서브넷을 표시합니다. AWS Cloud9 Amazon VPC (가상 사설 클라우드) 를 사용하여 새로 생성된 Amazon EC2 인스턴스와 통신합니다. 이 자습서에서는 미리 선택된 기본 설정을 변경하지 않는 것이 좋습니다. 기본 설정을 사용하면 새 환경과 동일한 지역 AWS 계정 및 지역에 있는 단일 서브넷이 있는 기본 VPC를 AWS Cloud9 사용하려고 합니다. Amazon VPC 설정 방식에 따라 다음 지침 중 하나를 따릅니다.

    무엇을 선택해야 하는지 확실하지 않으면 이 절차의 다음 단계로 건너뛰는 것이 좋습니다.

    네트워크 설정 (고급) 을 건너뛰고 미리 선택된 기본 설정을 그대로 두면 단일 서브넷에서 기본 VPC를 AWS Cloud9 사용하려고 시도합니다. AWS Cloud9 선택한 인스턴스 유형에 따라 서브넷을 선택합니다. 이들은 새 환경과 동일한 AWS 계정 및 AWS 지역에 있습니다.

    Important

    환경 유형으로 Existing compute(기존 컴퓨팅)를 선택한 경우 퍼블릭 또는 프라이빗 서브넷에서 인스턴스를 시작할 수 있습니다.

- 퍼블릭 서브넷(Public subnet): 인스턴스 SSM 에이전트가 Systems Manager와 통신할 수 있도록 인터넷 게이트웨이를 서브넷에 연결합니다.
- 프라이빗 서브넷(Private subnet): 인스턴스가 인터넷 및 다른 AWS 서비스와 통신할 수 있도록 NAT 게이트웨이를 생성합니다.

현재는 AWS 관리형 임시 자격 증명을 사용하여 IAM 사용자와 같은 AWS 엔티티를 AWS 서비스 대신하여 EC2 환경에 액세스하도록 허용할 수 없습니다.
서브넷 구성에 대한 자세한 내용은 개발 환경을 위한 AWS Cloud9 VPC 설정 섹션을 참조하세요.

Amazon VPC에 액세스할 수 AWS 계정 있습니까?	VPC가 새 환경과 동일한 지역 AWS 계정 및 지역에 있습니까?	해당 VPC가 AWS 계정의 기본 VPC입니까?	해당 VPC에 단일 서브넷이 포함되어 있습니까?	다음 지침을 따릅니다.
아니요	—	—	—	VPC가 없다면, 하나 만드십시오. 새 환경과 동일한 지역 AWS 계정 및 지역에 VPC를 만들려면 Create new VPC를 선택한 다음 화면의 지침을 따릅니다. 자세한 정보는 VPC 및 기타 VPC 리소스 생성을 참조하세요. 새 환경이 AWS 계정 아닌 다른 환경에서 VPC를 만들려면 Amazon VPC 사용 설명서의 공유 VPC 사용을 참조하십시오.

Amazon VPC에 액세스할 수 AWS 계정 있습니까?	VPC가 새 환경과 동일한 지역 AWS 계정 및 지역에 있습니까?	해당 VPC가 AWS 계정의 기본 VPC입니까?	해당 VPC에 단일 서브넷이 포함되어 있습니까?	다음 지침을 따릅니다.
예	예	예	예	이 절차의 다음 단계로 건너뜁니다. 네트워크 설정 (고급) 을 건너뛰고 미리 선택된 기본 설정을 변경하지 않으면 새 환경과 동일한 계정 및 지역에 있는 단일 서브넷이 있는 기본 VPC를 AWS Cloud9 사용하려고 시도합니다.

Amazon VPC에 액세스할 수 AWS 계정 있습니까?	VPC가 새 환경과 동일한 지역 AWS 계정 및 지역에 있습니까?	해당 VPC가 AWS 계정의 기본 VPC입니까?	해당 VPC에 단일 서브넷이 포함되어 있습니까?	다음 지침을 따릅니다.
예	예	예	아니요	기본 VPC에 여러 개의 서브넷이 있으면 Network settings (advanced)(네트워크 설정(고급))를 확장합니다. Subnet(서브넷)에서 AWS Cloud9 가 사전 선택된 기본 VPC에서 사용할 서브넷을 선택합니다. 기본 VPC에 서브넷이 없는 경우 하나를 생성합니다. 이렇게 하려면 Create new subnet(새 서브넷 생성)을 선택한 다음 화면의 지시를 따릅니다. 자세한 내용은 에 대한 서브넷을 생성하십시오. AWS Cloud9 섹션을 참조하세요.

Amazon VPC에 액세스할 수 AWS 계정 있습니까?	VPC가 새 환경과 동일한 지역 AWS 계정 및 지역에 있습니까?	해당 VPC가 AWS 계정의 기본 VPC입니까?	해당 VPC에 단일 서브넷이 포함되어 있습니까?	다음 지침을 따릅니다.
예	예	아니요	예	네트워크 설정을 확장합니다. Network(VPC)에서 AWS Cloud9 이 사용할 VPC를 선택합니다.

Amazon VPC에 액세스할 수 AWS 계정 있습니까?	VPC가 새 환경과 동일한 지역 AWS 계정 및 지역에 있습니까?	해당 VPC가 AWS 계정의 기본 VPC입니까?	해당 VPC에 단일 서브넷이 포함되어 있습니까?	다음 지침을 따릅니다.
예	예	아니요	아니요	네트워크 설정을 확장합니다. Network(VPC)에서 AWS Cloud9 이 사용할 VPC를 선택합니다. 선택한 VPC에 여러 개의 서브넷이 있으면 Network settings (advanced)(네트워크 설정(고급))를 확장합니다. Subnet의 경우 선택한 VPC에서 사용할 AWS Cloud9 서브넷을 선택합니다. 선택한 VPC에 서브넷이 없는 경우 하나를 생성합니다. 이렇게 하려면 Create new subnet(새 서브넷 생성)을 선택한 다음 화면의

Amazon VPC에 액세스할 수 AWS 계정 있습니까?	VPC가 새 환경과 동일한 지역 AWS 계정 및 지역에 있습니까?	해당 VPC가 AWS 계정의 기본 VPC입니까?	해당 VPC에 단일 서브넷이 포함되어 있습니까?	다음 지침을 따릅니다.
				지시를 따릅니다. 자세한 내용은 에 대한 서브넷을 생성하십시오. AWS Cloud9 섹션을 참조하세요.
예	아니요	예	—	AWS Cloud9 새 환경의 계정과 다른 기본 VPC를 사용할 수 없습니다. AWS 계정 이 목록에서 다른 옵션을 선택합니다.

Amazon VPC에 액세스할 수 AWS 계정 있습니까?	VPC가 새 환경과 동일한 지역 AWS 계정 및 지역에 있습니까?	해당 VPC가 AWS 계정의 기본 VPC입니까?	해당 VPC에 단일 서브넷이 포함되어 있습니까?	다음 지침을 따릅니다.
예	아니요	아니요	예	네트워크 설정을 확장합니다. Network(VPC)에서 AWS Cloud9 이 사용할 VPC를 선택합니다. **Note** VPC가 다른 계정에 있더라도 VPC는 새 환경과 동일한 리전에 있어야 합니다.

Amazon VPC에 액세스할 수 AWS 계정 있습니까?	VPC가 새 환경과 동일한 지역 AWS 계정 및 지역에 있습니까?	해당 VPC가 AWS 계정의 기본 VPC입니까?	해당 VPC에 단일 서브넷이 포함되어 있습니까?	다음 지침을 따릅니다.
예	아니요	아니요	아니요	네트워크 설정을 확장합니다. Network(VPC)에서 AWS Cloud9 이 사용할 VPC를 선택합니다. Subnet(서브넷)에서 AWS Cloud9 이 선택한 VPC에 사용할 서브넷을 선택합니다. 선택한 VPC에 서브넷이 없는 경우 새 환경이 AWS 계정 아닌 다른 VPC에 서브넷을 만들려면 Amazon VPC 사용 설명서의 공유 VPC 사용을 참조하십시오. **Note** VPC와 서브넷이 다른 계정에

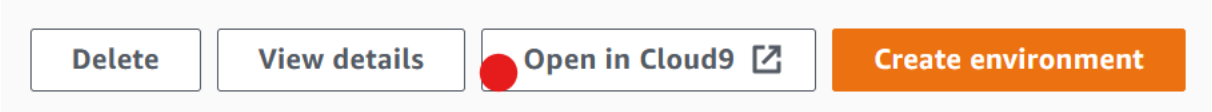
Amazon VPC에 액세스할 수 AWS 계정 있습니까?	VPC가 새 환경과 동일한 지역 AWS 계정 및 지역에 있습니까?	해당 VPC가 AWS 계정의 기본 VPC입니까?	해당 VPC에 단일 서브넷이 포함되어 있습니까?	다음 지침을 따릅니다.
				있더라도 VPC와 서브넷은 새 환경과 동일한 리전에 있어야 합니다.

어떤 것을 활용할지에 대해서는 개발 환경을 위한 AWS Cloud9 VPC 설정 단원을 참조하세요.

12. 각 태그에 키와 값을 지정하여 최대 50개의 태그를 추가합니다. Add new tag(새 태그 추가)를 선택하면 됩니다. 태그는 AWS Cloud9 환경에 리소스 태그로 연결되며 AWS CloudFormation 스택, Amazon EC2 인스턴스, Amazon EC2 보안 그룹과 같은 기본 리소스에 전파됩니다. 태그에 대한 자세한 내용은 IAM 사용 설명서의 AWS 리소스 태그를 사용한 액세스 제어 및 이 안내서의 고급 정보를 참조하십시오.

> **Warning**
>
> 태그를 생성한 후 이러한 태그를 업데이트하면 변경 사항이 기본 리소스에 전파되지 않습니다. 자세한 내용은 태그에 대한 고급 정보에서 기본 리소스에 태그 업데이트 전파 섹션을 참조하세요.

13. Create(생성)를 선택하여 환경을 만들면 홈 페이지로 리디렉션됩니다. 계정이 성공적으로 생성되면 AWS Cloud9 콘솔 상단에 녹색 플래시 바가 나타납니다. 새 환경을 선택하고 Open in Cloud9(Cloud9에서 열기)을 선택하여 IDE를 시작할 수 있습니다.



계정 생성에 실패하면 AWS Cloud9 콘솔 상단에 빨간색 플래시 바가 나타납니다. 웹 브라우저, AWS 액세스 권한, 인스턴스 또는 연결된 네트워크 관련 문제 때문에 계정이 생성되지 않을 수 있습니다. AWS Cloud9 문제 해결 섹션에서 해결 방법 관련 정보를 확인할 수 있습니다.

> **Note**
>
> AWS Cloud9 IMDSv1과 IMDSv2를 모두 지원합니다. IMDSv1과 비교하여 향상된 보안 수준을 제공하는 IMDSv2를 채택하는 것이 좋습니다. IMDSv2의 이점에 대한 자세한 내용은 AWS 보안 블로그 섹션을 참조하십시오. IMDSv1에서 IMDSv2로의 전환에 대한 자세한 내용은 Linux 인스턴스용 Amazon EC2 사용 설명서의 인스턴스 메타데이터 서비스 버전 2 사용으로 전환 섹션을 참조하십시오.

> **Note**
>
> 프록시를 사용하여 인터넷에 액세스하는 환경인 경우 종속 항목을 설치할 수 AWS Cloud9 있도록 프록시 세부 정보를 제공해야 합니다. 자세한 정보는 종속성을 설치하지 못함을 참조하세요.

코드를 사용하여 환경 생성

코드를 사용하여 EC2 환경을 생성하려면 다음과 같이 EC2 환경 AWS Cloud9 생성 작업을 호출하십시오. AWS Cloud9

AWS CLI	create-environment-ec2.
AWS SDK for C++	CreateEnvironmentEC2 요청, EC2 결과 CreateEnvironment
AWS SDK for Go	CreateEnvironmentEC2, EC2 요청, EC2 CreateEnvironment CreateEnvironment WithContext
AWS SDK for Java	CreateEnvironment CreateEnvironmentEC2 요청, EC2 결과

AWS SDK for JavaScript	createEnvironmentEC2
AWS SDK for .NET	CreateEnvironmentEC2 요청, CreateEnvironmentEC2 응답
AWS SDK for PHP	createEnvironmentEC2
AWS SDK for Python (Boto)	create_environment_ec2
AWS SDK for Ruby	create_environment_ec2
AWS Tools for Windows PowerShell	New-C9EnvironmentEC2
AWS Cloud9 API	CreateEnvironmentEC2

> **Note**
>
> 환경에서 프록시를 사용하여 인터넷에 액세스하는 경우 종속성을 설치할 수 AWS Cloud9 있도록 프록시 세부 정보를 제공해야 합니다. 자세한 정보는 종속성을 설치하지 못함을 참조하세요.

SSH 환경 생성

콘솔을 사용하여 AWS Cloud9 SSH 개발 환경을 생성합니다. AWS Cloud9 CLI를 사용하여 SSH 환경을 만들 수는 없습니다.

필수 조건

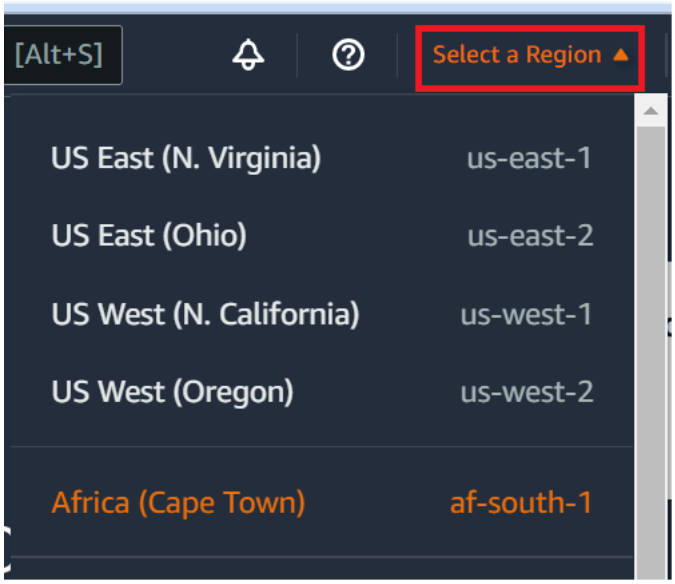
- 먼저 AWS Cloud9 설정의 단계를 완료했는지 확인합니다. 이렇게 해야 AWS Cloud9 콘솔에 로그인하여 환경을 생성할 수 있습니다.
- 환경에 AWS Cloud9 연결하려는 기존 클라우드 컴퓨팅 인스턴스 (예: Amazon EC2 인스턴스 AWS 계정) 또는 자체 서버를 식별합니다.
- 기존 인스턴스 또는 자체 서버가 모든 SSH 호스트 요구 사항을 충족해야 합니다. 이러한 요구 사항에는 특정 버전의 Python, Node.js 및 기타 구성 요소 설치, 로그인 후 AWS Cloud9 이 시작하도록 할 디렉터리에 대한 특정 권한 설정, 연결된 Amazon Virtual Private Cloud 설정이 포함됩니다.

SSH 환경 생성

1. 위의 사전 조건을 완료해야 합니다.
2. 아직 연결되지 않은 경우 SSH 클라이언트를 사용하여 기존 인스턴스 또는 자체 서버에 연결합니다. 이렇게 하면 필요한 공개 SSH 키 값을 인스턴스나 서버에 추가할 수 있습니다. 이 절차의 뒷부분에서 자세하게 설명합니다.

   > **Note**
   >
   > 기존 AWS 클라우드 컴퓨팅 인스턴스에 연결하려면 다음 리소스 중 하나 이상을 참조하십시오.
   >
   > - Amazon EC2의 경우 Amazon EC2 사용 설명서의 Linux 인스턴스에 연결을 참조하십시오.
   > - Amazon Lightsail의 경우 Amazon Lightsail 문서에서 Linux/Unix 기반 Lightsail 인스턴스에 연결을 참조하세요.
   > - 에 대해서는 AWS Elastic BeanstalkAWS Elastic Beanstalk 개발자 안내서의 서버 인스턴스 나열 및 연결을 참조하십시오.
   > - 에 대해서는 AWS OpsWorks사용 설명서의 SSH를 사용하여 Linux 인스턴스에 로그인하는 방법을 참조하십시오.AWS OpsWorks
   > - 기타 AWS 서비스정보는 해당 서비스의 설명서를 참조하십시오.
   >
   > 자체 서버에 연결하려면 SSH를 사용하세요. SSH는 macOS 및 Linux 운영 체제에서는 이미 설치되어 있습니다. 윈도우에서 SSH를 사용하여 서버에 연결하려면 PuTTY를 설치해야 합니다.

3. AWS Cloud9 콘솔의 https://console.aws.amazon.com/cloud9/ 에 로그인합니다.
4. AWS Cloud9 콘솔에 로그인한 후 상단 탐색 표시줄에서 환경을 만들 항목을 AWS 리전 선택합니다. 사용 가능한 목록은 AWS 리전AWS Cloud9을 참조하십시오 AWS 일반 참조.



5. 개발 환경을 처음 생성하는 경우 시작 페이지가 표시됩니다. 새 AWS Cloud9 환경 패널에서 환경 만들기를 선택합니다.

   이전에 개발 환경을 생성한 경우 화면 왼쪽의 창을 확장합니다. Your environments(사용자 환경)를 선택하고 Create environment(환경 생성)를 선택합니다.

   시작 페이지에서:

   또는 Your environments(환경) 페이지에서:

6. Create environment(환경 생성) 페이지에 환경의 이름을 입력합니다.
7. Description(설명)에 환경에 대한 설명을 입력합니다. 본 자습서에서는 `This environment is for the AWS Cloud9 tutorial.`을 사용합니다.
8. Environment type(환경 유형)에서는 다음 옵션 중에서 Existing Compute(기존 컴퓨팅)를 선택합니다.

- 새 EC2 인스턴스 — SSH 또는 SSM을 통해 직접 연결할 수 AWS Cloud9 있는 Amazon EC2 인스턴스를 시작합니다.

- 기존 컴퓨팅 — SSH 로그인 세부 정보와 포트 22를 열어야 하는 기존 Amazon EC2 인스턴스를 시작합니다. AWS Cloud9 를 통해 인스턴스에 연결합니다. AWS Systems Manager
  - Existing compute(기존 컴퓨팅) 옵션을 선택하면 서비스 역할이 자동으로 생성됩니다. 인터페이스의 더 아래에 있는 Systems Manager 액세스를 위한 서비스 역할 및 인스턴스 프로파일 섹션에서 서비스 역할의 이름을 확인할 수 있습니다. 자세한 정보는 AWS Systems Manager를 사용하여 수신하지 않는 EC2 인스턴스에 액세스을 참조하세요.

> **Warning**
>
> 사용자 환경에 맞게 EC2 인스턴스를 생성하면 Amazon EC2에 AWS 계정 대한 요금이 부과될 수 있습니다. Systems Manager를 사용하여 EC2 인스턴스에 대한 연결을 관리하는 데 따른 추가 비용은 없습니다.

> **Warning**
>
> AWS Cloud9 SSH 퍼블릭 키를 사용하여 서버에 안전하게 연결합니다. 보안 연결을 설정하려면 다음 단계에서 공개 키를 `~/.ssh/authorized_keys` 파일에 추가하고 로그인 보안 인증 정보를 제공하십시오. Copy key to clipboard(클립보드에 키 복사)를 선택하여 SSH 키를 복사하거나, View public SSH key(공개 SSH 키 보기)를 선택하여 키를 표시합니다.

9. Existing compute(기존 컴퓨팅) 패널의 User(사용자)에 이 절차의 앞부분에서 인스턴스 또는 서버에 연결하는 데 사용한 로그인 이름을 입력합니다. 예를 들어 AWS 클라우드 컴퓨팅 인스턴스의 경우 이 값은 `ec2-user`, `ubuntu` 또는 `root`일 수 있습니다.

> **Note**
>
> 로그인 이름이 인스턴스 또는 서버의 관리 권한 또는 관리자 사용자와 연결되는 것이 좋습니다. 더 자세히 말하자면, 이 로그인 이름은 인스턴스 또는 서버에서 Node.js 설치를 소유하는 것이 좋습니다. 이를 확인하려면 인스턴스 또는 서버의 터미널에서 **`ls -l $(which node)`** (또는 `nvm`을 사용하는 경우 **`ls -l $(nvm which node)`**) 명령을

실행합니다. 이 명령은 Node.js 설치의 소유자 이름을 표시합니다. 설치의 권한, 그룹 이름과 위치도 표시합니다.

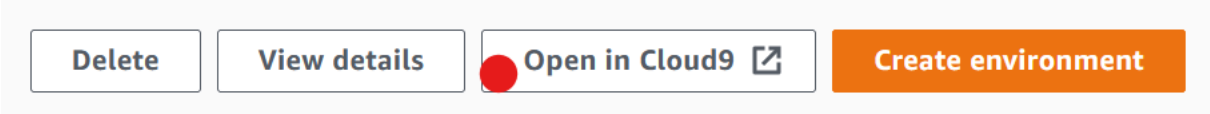
10. Host(호스트)에 인스턴스 또는 서버의 퍼블릭 IP 주소(기본) 또는 호스트 이름을 입력합니다.
11. Port에는 인스턴스 또는 서버에 연결하는 AWS Cloud9 데 사용할 포트를 입력합니다. 또는 기본 포트를 그대로 유지합니다.
12. Additional details - optional(추가 상세 정보 - 선택 사항)을 선택하여 환경 경로, node.js 바이너리 경로 및 SSH 점프 호스트 정보를 표시합니다.
13. 환경 경로에는 AWS Cloud9 시작하려는 인스턴스 또는 서버의 디렉터리 경로를 입력합니다. 이 절차의 사전 요구 사항에서 이를 이미 확인했습니다. 이 항목을 비워 두면 AWS Cloud9 은 로그인 후 인스턴스 또는 서버가 일반적으로 시작되는 디렉터리를 사용합니다. 이 디렉터리는 일반적으로 홈 또는 기본 디렉터리입니다.
14. Path to Node.js binary path(Node.js 바이너리 경로로 가는 경로)에 경로 정보를 입력하여, 인스턴스 또는 서버의 Node.js 바이너리로 가는 경로를 지정합니다. 경로를 가져오려면 인스턴스 또는 서버에서 **`which node`**(또는 `nvm`을 사용하는 경우 **`nvm which node`**) 명령을 실행할 수 있습니다. 예를 들어 이 경로는 `/usr/bin/node`일 수 있습니다. 이 항목을 비워 두면 AWS Cloud9 은 연결을 시도할 때 Node.js 바이너리의 위치를 추측합니다.
15. SSH jump host(SSH 점프 호스트)에 인스턴스 또는 서버가 사용하는 점프 호스트에 대한 정보를 입력합니다. 형식 `USER_NAME@HOSTNAME:PORT_NUMBER`(예:`ec2-user@ip-192-0-2-0:22`)를 사용합니다.

    점프 호스트는 다음 요구 사항을 충족해야 합니다.

    - SSH를 사용하여 퍼블릭 인터넷을 통해 접근 가능해야 합니다.
    - 지정된 포트를 통해 어떤 IP 주소에서든 인바운드 액세스를 허용해야 합니다.
    - 기존 인스턴스 또는 서버의 `~/.ssh/authorized_keys` 파일에 복사된 퍼블릭 SSH 키 값을 점프 호스트의 `~/.ssh/authorized_keys` 파일에도 복사해야 합니다.
    - Netcat이 설치되어 있어야 합니다.
16. 각 태그에 키와 값을 지정하여 최대 50개의 태그를 추가합니다. Add new tag(새 태그 추가)를 선택하면 됩니다. 태그는 AWS Cloud9 환경에 리소스 태그로 연결되며 AWS CloudFormation 스택, Amazon EC2 인스턴스, Amazon EC2 보안 그룹과 같은 기본 리소스에 전파됩니다. 태그에 대한 자세한 내용은 IAM 사용 설명서의 AWS 리소스 태그를 사용한 액세스 제어 및 이 안내서의 태그에 대한 고급 정보를 참조하십시오.

> **Warning**
>
> 태그를 생성한 후 이러한 태그를 업데이트하면 변경 사항이 기본 리소스에 전파되지 않습니다. 자세한 내용은 태그에 대한 고급 정보에서 기본 리소스에 태그 업데이트 전파 섹션을 참조하세요.

17. Create(생성)를 선택하여 환경을 만들면 홈 페이지로 리디렉션됩니다. 계정이 성공적으로 생성되면 AWS Cloud9 콘솔 상단에 녹색 플래쉬 바가 나타납니다. 새 환경을 선택하고 Open in Cloud9(Cloud9에서 열기)을 선택하여 IDE를 시작할 수 있습니다.



계정이 생성되지 못하면 AWS Cloud9 콘솔 상단에 적색 플래시바가 나타납니다. 웹 브라우저, AWS 액세스 권한, 인스턴스 또는 관련 네트워크에 문제가 있어 계정을 만들지 못할 수 있습니다. 계정 장애를 일으킬 수 있는 문제를 해결하는 방법 관련 정보는 AWS Cloud9 문제 해결 섹션에서 확인할 수 있습니다.

> **Note**
>
> 프록시를 사용하여 인터넷에 액세스하는 환경인 경우 종속 항목을 설치할 수 AWS Cloud9 있도록 프록시 세부 정보를 제공해야 합니다. 자세한 내용은 종속성을 설치하지 못함을(를) 참조하세요.

AWS Systems Manager를 사용하여 수신하지 않는 EC2 인스턴스에 액세스

EC2 환경용으로 생성된 '수신하지 않는 EC2 인스턴스'는 AWS Cloud9이 해당 인스턴스의 인바운드 포트를 열 필요 없이 Amazon EC2 인스턴스에 연결할 수 있도록 합니다. 콘솔, 명령줄 인터페이스 또는 AWS CloudFormation 스택을 사용하여 EC2 환경을 생성할 때 수신 안 함 옵션을 선택할 수 있습니다

> **Important**
>
> Systems Manager 세션 관리자를 사용하여 EC2 인스턴스에 대한 연결을 관리하는 데 따른 추가 요금은 없습니다.

콘솔의 Create environment(환경 생성) 섹션에서 환경 유형을 선택할 때, 인바운드 연결이 필요한 새 EC2 인스턴스 또는 다음 사항이 필요 없는 수신되지 않는 새 EC2 인스턴스를 선택할 수 있습니다.

- 새 EC2 인스턴스 – 이 설정을 사용하면 인스턴스의 보안 그룹에 수신 네트워킹 트래픽을 허용하는 규칙이 포함됩니다. 수신 네트워크 트래픽은 AWS Cloud9 연결용으로 승인된 IP 주소로 제한됩니다. AWS Cloud9은 열려 있는 인바운드 포트를 사용하여 SSH를 통해 해당 인스턴스에 연결할 수 있습니다. AWS Systems Manager Session Manager를 사용하는 경우 인바운드 포트를 열지 않고도 (수신하지 않음) SSM을 통해 Amazon EC2 인스턴스에 액세스할 수 있습니다. 이 방법은 새 Amazon EC2 인스턴스에만 적용됩니다. 자세한 내용은 EC2 환경에 Systems Manager 사용하는 데 따른 이점 섹션을 참조하세요.

- 기존 컴퓨팅 - 이 설정을 사용하면 SSH 로그인 세부 정보가 필요한 기존 Amazon EC2 인스턴스에 액세스할 수 있으며, 이 경우 인스턴스에 인바운드 보안 그룹 규칙이 있어야 합니다. 이 옵션을 선택하면 서비스 역할이 자동으로 생성됩니다. 설정 화면 하단에 표시되는 메모에서 서비스 역할의 이름을 확인할 수 있습니다.

AWS CLI를 사용하여 환경을 생성하는 경우 `create-environment-ec2` 명령을 호출할 때 `--connection-type CONNECT_SSM` 옵션을 설정하여 수신하지 않는 EC2 인스턴스를 구성할 수 있습니다. 필요한 서비스 역할 및 인스턴스 프로파일을 생성하는 방법에 대한 자세한 내용은 AWS CLI를 사용하여 Systems Manager의 인스턴스 프로파일 관리 섹션을 참조하세요.

수신하지 않는 EC2 인스턴스를 사용하는 환경의 생성을 완료한 후 다음을 확인합니다.

- Systems Manager Session Manager에는 사용자를 대신하여 EC2 인스턴스에 대한 작업을 수행할 수 있는 권한이 있습니다. 자세한 내용은 Systems Manager 권한 관리 섹션을 참조하세요.
- AWS Cloud9는 Session Manager에서 관리하는 인스턴스에 액세스할 수 있습니다. 자세한 내용은 사용자에게 세션 관리자에서 관리하는 인스턴스에 대한 액세스 권한 부여 섹션을 참조하세요.

EC2 환경에 Systems Manager 사용하는 데 따른 이점

세션 관리자가 AWS Cloud9과 EC2 인스턴스 간의 보안 연결을 처리하도록 허용하면 다음과 같은 두 가지 주요 이점이 있습니다.

- 인스턴스를 위해 인바운드 포트를 열어야 하는 요구 사항 없음
- 퍼블릭 또는 프라이빗 서브넷으로 인스턴스를 시작하는 옵션

No open inbound ports

AWS Cloud9과 해당 EC2 인스턴스 간의 보안 연결은 세션 관리자가 처리합니다. 세션 관리자는 완전관리형 Systems Manager 기능으로, AWS Cloud9이 인바운드 포트를 열 필요 없이 해당 EC2 인스턴스에 연결할 수 있게 합니다.

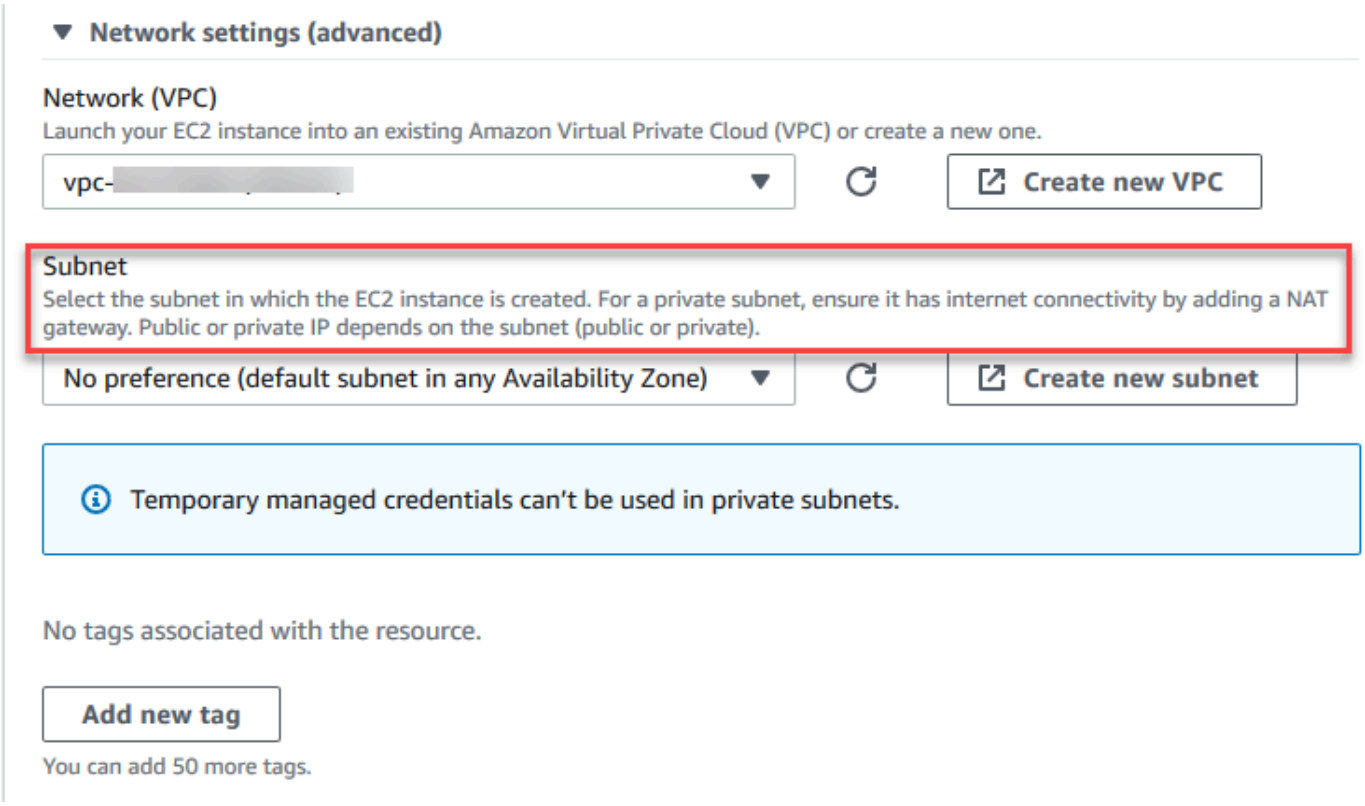
> **Important**
>
> 수신하지 않는 연결에 Systems Manager를 사용하는 옵션은 현재 새 EC2 환경을 생성할 때만 사용할 수 있습니다.

세션 관리자 세션이 시작되면 대상 인스턴스에 대한 연결이 설정됩니다. 연결이 설정되면 이제 환경이 Systems Manager 서비스를 통해 인스턴스와 상호 작용할 수 있습니다. Systems Manager 서비스는 Systems Manager Agent(SSM Agent)를 통해 인스턴스와 통신합니다.

SSM Agent는 EC2 환경에서 사용되는 모든 인스턴스에 기본적으로 설치됩니다.

Private/public subnets

[네트워크 설정(고급)(Network settings (advanced))] 섹션에서 인스턴스의 서브넷을 선택할 때, 환경의 인스턴스가 Systems Manager를 통해 액세스되는 경우 프라이빗 또는 퍼블릭 서브넷을 선택할 수 있습니다.



프라이빗 서브넷

프라이빗 서브넷의 경우 인스턴스가 SSM 서비스에 계속 연결할 수 있는지 확인합니다. 이는 퍼블릭 서브넷에 NAT 게이트웨이를 설정하거나 Systems Manager를 위한 VPC 엔드포인트를 구성하는 방법으로 수행할 수 있습니다.

NAT 게이트웨이를 사용하면 인터넷이 프라이빗 서브넷의 인스턴스에 대한 연결을 시작하지 못한다는 이점이 있습니다. 사용자 환경의 인스턴스에는 퍼블릭 IP 주소 대신 프라이빗 IP 주소가 할당됩니다. NAT 게이트웨이는 인스턴스에서 인터넷 또는 기타 AWS 서비스로 트래픽을 전달한 후 인스턴스에 응답을 다시 보냅니다.

VPC 옵션의 경우 Systems Manager를 위한 최소 3개의 필수 인터페이스 엔드포인트, 즉 com.amazonaws.region.ssm, com.amazonaws.region.ec2messages, com.amazonaws.region.ssmmessages를 생성합니다. 자세한 내용은 AWS Systems Manager 사용 설명서에서 Systems Manager를 위한 VPC 엔드포인트 생성을 참조하세요.

> **Important**
>
> 현재, 환경의 EC2 인스턴스가 프라이빗 서브넷에서 시작된 경우 AWS 관리형 임시 자격 증명을 사용하여 EC2 환경이 AWS엔터티(예: IAM 사용자)를 대신하여 AWS 서비스에 액세스하도록 허용할 수 없습니다.

퍼블릭 서브넷

개발 환경에서 SSM을 사용하여 EC2 인스턴스에 액세스하는 경우 인스턴스가 시작된 퍼블릭 서브넷에 의해 인스턴스에 퍼블릭 IP 주소가 할당되었는지 확인합니다. 이렇게 하려면 고유한 IP 주소를 지정하거나 퍼블릭 IP 주소의 자동 할당을 사용하도록 설정할 수 있습니다. IP 설정 자동 할당 수정과 관련된 단계는 Amazon VPC 사용 설명서에서 VPC의 IP 주소 지정을 참조하세요.

환경 인스턴스의 프라이빗 및 퍼블릭 서브넷을 구성하는 방법에 대한 자세한 내용은 에 대한 서브넷을 생성하십시오. AWS Cloud9 섹션을 참조하세요.

Systems Manager 권한 관리

Systems Manager는 기본적으로 EC2 인스턴스에서 작업을 수행할 권한이 없습니다. 액세스는 AWS Identity and Access Management(IAM) 인스턴스 프로파일을 통해 제공됩니다. (인스턴스 프로파일은 시작할 때 IAM 역할 정보를 EC2 인스턴스에 전달하는 컨테이너입니다.)

AWS Cloud9 콘솔을 사용하여 수신하지 않는 EC2 인스턴스를 생성 때, 서비스 역할(`AWSCloud9SSMAccessRole`) 및 IAM 인스턴스 프로파일(`AWSCloud9SSMInstanceProfile`)이 자동으로 생성됩니다. (`AWSCloud9SSMAccessRole`을 IAM 관리 콘솔에서 확인할 수 있습니다. 인스턴스 프로파일은 IAM 콘솔에 표시되지 않습니다.)

> **Important**
>
> 처음으로 AWS CLI를 사용하여 수신하지 않는 EC2 환경을 생성하는 경우 필요한 서비스 역할과 인스턴스 프로파일을 명시적으로 정의해야 합니다. 자세한 내용은 AWS CLI를 사용하여 Systems Manager의 인스턴스 프로파일 관리 섹션을 참조하세요.

> **Important**
>
> AWS Cloud9 환경을 생성하고 `AWSCloud9Administrator` 또는 `AWSCloud9User` 정책이 연결된 Amazon EC2 Systems Manager를 사용하는 경우 특정 IAM 권한이 있는 사용자 지정 정책도 연결해야 합니다. SSM 환경 생성을 위한 사용자 지정 IAM 정책을 참조하세요. 이는 `AWSCloud9Administrator` 및 `AWSCloud9User` 정책의 권한 문제 때문입니다.

보안을 더 강화하기 위해 AWS Cloud9 서비스 연결 역할인 `AWSServiceRoleforAWSCloud9`의 `AWSCloud9ServiceRolePolicy` 정책에는 `PassRole` 제한이 포함되어 있습니다. IAM 역할을 서비스에 전달(pass)할 경우, 해당 서비스가 역할을 수임하고 사용자 대신 작업을 수행할 수 있습니다. 이 경우 `PassRole` 권한은 AWS Cloud9이 `AWSCloud9SSMAccessRole` 역할(및 해당 권한)만 EC2 인스턴스에 전달할 수 있도록 합니다. 이렇게 하면 EC2 인스턴스에 대해 수행할 수 있는 작업이 AWS Cloud9에 필요한 작업만으로 제한됩니다.

> **Note**
>
> 인스턴스에 액세스하기 위해 Systems Manager를 더 이상 사용할 필요가 없는 경우 `AWSCloud9SSMAccessRole` 서비스 역할을 삭제할 수 있습니다. 자세한 내용은 IAM 사용 설명서에서 역할 또는 인스턴스 프로필 삭제를 참조하세요.

AWS CLI를 사용하여 Systems Manager의 인스턴스 프로파일 관리

AWS CLI를 사용하여, 수신하지 않는 EC2 환경을 생성할 수도 있습니다. `create-environment-ec2`를 호출할 때 `--connection-type` 옵션을 `CONNECT_SSM`으로 설정합니다.

이 옵션을 사용하면 `AWSCloud9SSMAccessRole` 서비스 역할 및 `AWSCloud9SSMInstanceProfile`이 자동으로 생성되지 않습니다. 따라서 다음 중 하나를 수행하여 필요한 서비스 프로파일과 인스턴스 프로파일을 생성합니다.

- 이후 `AWSCloud9SSMAccessRole` 서비스 역할과 `AWSCloud9SSMInstanceProfile`이 자동으로 생성되고 나면 콘솔을 사용하여 EC2 환경을 생성합니다. 생성된 서비스 역할 및 인스턴스 프로파일은 AWS CLI를 사용하여 생성한 모든 추가 EC2 환경에 사용할 수 있습니다.
- 다음 AWS CLI 명령을 실행하여 서비스 역할 및 인스턴스 프로파일을 생성합니다.

```
aws iam create-role --role-name AWSCloud9SSMAccessRole --path /service-role/ --
assume-role-policy-document '{"Version": "2012-10-17","Statement": [{"Effect":
```

```
 "Allow","Principal": {"Service": ["ec2.amazonaws.com","cloud9.amazonaws.com"]
  },"Action": "sts:AssumeRole"}]}'
aws iam attach-role-policy --role-name AWSCloud9SSMAccessRole --policy-arn
 arn:aws:iam::aws:policy/AWSCloud9SSMInstanceProfile
aws iam create-instance-profile --instance-profile-name AWSCloud9SSMInstanceProfile
 --path /cloud9/
aws iam add-role-to-instance-profile --instance-profile-name
 AWSCloud9SSMInstanceProfile --role-name AWSCloud9SSMAccessRole
```

사용자에게 세션 관리자에서 관리하는 인스턴스에 대한 액세스 권한 부여

Systems Manager를 통해 EC2 인스턴스에 연결된 AWS Cloud9 환경을 열려면, 사용자에게 `StartSession` API 작업에 대한 권한이 있어야 합니다. 이 작업은 세션 관리자 세션을 위해 관리형 EC2 인스턴스에 대한 연결을 시작합니다. AWS Cloud9 전용 관리형 정책을 사용(권장)하거나 IAM 정책을 편집하고 필요한 권한을 추가하여 사용자에게 액세스 권한을 부여할 수 있습니다.

방법	설명
AWS Cloud9 전용 관리형 정책 사용	AWS 관리형 정책을 사용하여 사용자가 Systems Manager에서 관리되는 EC2 인스턴스에 액세스하도록 허용하는 것이 좋습니다. 관리형 정책은 표준 AWS Cloud9 사용 사례를 위한 일련의 권한을 제공하며 IAM 엔터티에 쉽게 연결할 수 있습니다. 또한 모든 관리형 정책에는 `StartSession` API 작업을 실행할 권한이 포함되어 있습니다. AWS Cloud9 전용 관리형 정책은 다음과 같습니다. • `AWSCloud9Administrator` (`arn:aws:iam::aws:policy/AWSCloud9Administrator`) • `AWSCloud9User` (`arn:aws:iam::aws:policy/AWSCloud9User`)

방법	설명
	• AWSCloud9EnvironmentMember (arn:aws:iam::aws:policy/AWSCloud9EnvironmentMember) **Important** AWS Cloud9 환경을 생성하고 AWSCloud9Administrator 또는 AWSCloud9User 정책이 연결된 Amazon EC2 Systems Manager를 사용하는 경우 특정 IAM 권한이 있는 사용자 지정 정책도 연결해야 합니다. SSM 환경 생성을 위한 사용자 지정 IAM 정책을 참조하세요. 이는 AWSCloud9Administrator 및 AWSCloud9User 정책의 권한 문제 때문입니다. 자세한 내용은 AWS 관리형 정책은 다음과 같습니다. AWS Cloud9 섹션을 참조하세요.
IAM 정책 편집 및 필요한 정책 문 추가	기존 정책을 편집하려면 StartSession API에 대한 권한을 추가하면 됩니다. AWS Management Console 또는 AWS CLI를 사용하여 정책을 편집하려면 IAM 사용 설명서의 IAM 정책 편집에서 제공하는 지침을 따릅니다. 정책을 편집할 때 ssm:startSession API 작업을 실행되도록 하는 policy statement(다음 참조)을 추가합니다.

다음 권한을 사용하여 StartSession API 작업을 실행할 수 있습니다. ssm:resourceTag 조건 키는 인스턴스가 AWS Cloud9 EC2 개발 환경(aws:cloud9:environment)인 것을 조건으로, 모든 인

스턴스(Resource: arn:aws:ec2:*:*:instance/*)에 대해 세션 관리자 세션을 시작할 수 있도록 지정합니다.

> **Note**
>
> 다음 관리형 정책에는 AWSCloud9Administrator, AWSCloud9User 및 AWSCloud9EnvironmentMember 정책 문도 포함되어 있습니다.

```
{
            "Effect": "Allow",
            "Action": "ssm:StartSession",
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:cloud9:environment": "*"
                },
                "StringEquals": {
                    "aws:CalledViaFirst": "cloud9.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:document/*"
            ]
        }
```

AWS CloudFormation을 사용하여 수신하지 않는 EC2 환경 생성

AWS CloudFormation 템플릿를 사용하여 수신하지 않는 Amazon EC2 개발 환경을 정의할 경우, 스택을 생성하기 전에 다음을 수행합니다.

1. AWSCloud9SSMAccessRole 서비스 역할 및 AWSCloud9SSMInstanceProfile 인스턴스 프로파일을 생성합니다. 자세한 내용은 AWS CloudFormation 템플릿을 사용하여 서비스 역할 및 인스턴스 프로파일 생성 섹션을 참조하세요.

2. AWS CloudFormation을 호출하는 IAM 엔터티에 대한 정책을 업데이트합니다. 이렇게 하면 해당 엔터티는 EC2 인스턴스에 연결하는 Session Manager 세션을 시작할 수 있습니다. 자세한 내용은 IAM 정책에 Systems Manager 권한 추가 섹션을 참조하세요.

AWS CloudFormation 템플릿을 사용하여 서비스 역할 및 인스턴스 프로파일 생성

Systems Manager가 개발 환경을 지원하는 EC2 인스턴스를 관리할 수 있도록 서비스 역할 `AWSCloud9SSMAccessRole` 및 인스턴스 프로파일 `AWSCloud9SSMInstanceProfile`을 생성해야 합니다.

이전에 with the console하거나 AWS CLI 명령을 실행하여 수신하지 않는 EC2 환경을 생성함으로써 `AWSCloud9SSMAccessRole` 및 `AWSCloud9SSMInstanceProfile`을 생성한 경우, 서비스 역할과 인스턴스 프로파일을 이미 사용할 수 있습니다.

> **Note**
>
> 수신하지 않는 EC2 환경에 대한 AWS CloudFormation 스택을 생성하려고 하지만 먼저 필요한 서비스 역할과 인스턴스 프로파일을 생성하지 않았다고 가정합니다. 그러면 스택이 생성되지 않고 다음과 같은 오류 메시지가 표시됩니다.
> Instance profile AWSCloud9SSMInstanceProfile does not exist in account(인스턴스 프로파일 AWSCloud9SSMInstanceProfile이 계정에 없습니다.)

AWS CloudFormation을 사용하여 수신하지 않는 EC2 환경을 처음으로 생성하는 경우, 템플릿에서 `AWSCloud9SSMAccessRole` 및 `AWSCloud9SSMInstanceProfile`을 IAM 리소스로 정의할 수 있습니다.

샘플 템플릿에서 발췌한 이 코드는 이러한 리소스를 정의하는 방법을 보여줍니다. `AssumeRole` 작업은 AWS Cloud9 환경과 그 EC2 인스턴스 모두에 대한 액세스를 제공하는 보안 인증 정보를 반환합니다.

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  AWSCloud9SSMAccessRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
```

```
          - Effect: Allow
            Principal:
              Service:
              - cloud9.amazonaws.com
              - ec2.amazonaws.com
            Action:
              - 'sts:AssumeRole'
      Description: 'Service linked role for AWS Cloud9'
      Path: '/service-role/'
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/AWSCloud9SSMInstanceProfile
      RoleName: 'AWSCloud9SSMAccessRole'

  AWSCloud9SSMInstanceProfile:
    Type: "AWS::IAM::InstanceProfile"
    Properties:
      InstanceProfileName: AWSCloud9SSMInstanceProfile
      Path: "/cloud9/"
      Roles:
        -
          Ref: AWSCloud9SSMAccessRole
```

IAM 정책에 Systems Manager 권한 추가

AWS CloudFormation 템플릿에 서비스 역할과 인스턴스 프로파일을 정의한 후에는 스택을 생성하는 IAM 엔터티에 Session Manager 세션을 시작할 권한이 있는지 확인해야 합니다. 세션은 Session Manager를 사용한 EC2 인스턴스에 대한 연결입니다.

> **Note**
>
> 수신하지 않는 EC2 환경을 위한 스택을 생성하기 전에 세션 관리자 세션을 시작할 권한을 추가하지 않으면 `AccessDeniedException` 오류가 반환됩니다.

AWS CloudFormation을 호출하여 IAM 엔터티에 대한 정책에 다음 권한을 추가합니다.

```
{
            "Effect": "Allow",
            "Action": "ssm:StartSession",
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
```

```
                "StringLike": {
                    "ssm:resourceTag/aws:cloud9:environment": "*"
                },
                "StringEquals": {
                    "aws:CalledViaFirst": "cloudformation.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:document/*"
            ]
        }
```

종속 구성 요소를 다운로드하도록 Amazon S3의 VPC 엔드포인트 구성

AWS Cloud9 환경의 EC2 인스턴스가 인터넷에 액세스할 수 없는 경우, 지정된 Amazon S3 버킷의 VPC 엔드포인트를 생성합니다. 이 버킷에는 IDE를 최신 상태로 유지하는 데 필요한 종속 구성 요소가 포함되어 있습니다.

또한 Amazon S3 VPC 엔드포인트를 설정하려면 액세스 정책을 사용자 지정해야 합니다. 액세스 정책에서는 다운로드할 종속 구성 요소가 포함된 신뢰할 수 있는 S3 버킷에 대한 액세스만 허용하고자 합니다.

> **Note**
>
> AWS Management Console, AWS CLI 또는 Amazon VPC API를 사용하여 VPC 엔드포인트를 생성하고 구성할 수 있습니다. 아래의 절차에서는 콘솔 인터페이스를 사용하여 VPC 엔드포인트를 생성하는 방법을 보여줍니다.

Amazon S3의 VPC 엔드포인트 생성 및 구성

1. AWS Management Console에서 Amazon VPC의 콘솔 페이지로 이동합니다.
2. 탐색 모음에서 [엔드포인트(Endpoints)]를 선택합니다.
3. [엔드포인트(Endpoints)] 페이지에서 [엔드포인트 생성(Create Endpoint)]을 선택합니다.

4. Create Endpoint(엔드포인트 생성) 페이지에서 검색 필드에 ‘s3’를 입력하고 Return 키를 눌러 현재 AWS 리전의 Amazon S3에 사용 가능한 엔드포인트를 나열합니다.
5. 반환된 Amazon S3 엔드포인트 목록에서 [게이트웨이(Gateway)] 유형을 선택합니다.
6. 그런 다음, 환경의 EC2 인스턴스가 포함된 VPC 선택합니다.
7. 이제 VPC의 라우팅 테이블을 선택합니다. 이렇게 하면 연결된 서브넷이 엔드포인트에 액세스할 수 있습니다. 환경의 EC2 인스턴스는 이들 서브넷 중 하나에 있습니다.
8. Policy(정책) 섹션에서 Custom(사용자 지정) 옵션을 선택하고 표준 정책을 다음으로 바꿉니다.

```
{
  "Version": "2008-10-17",
  "Statement": [
      {
          "Sid": "Access-to-C9-bucket-only",
          "Effect": "Allow",
          "Principal": "*",
          "Action": "s3:GetObject",
          "Resource": "arn:aws:s3:::{bucket_name}/content/dependencies/*"
      }
  ]
}
```

`Resource` 요소의 경우 `{bucket_name}`을 AWS 리전에서 사용 가능한 버킷의 실제 이름으로 바꿉니다. 예를 들어 EU(아일랜드) 리전에서 AWS Cloud9을 사용하는 경우 `"Resource": "arn:aws:s3:::static-eu-west-1-prod-static-hld3vzaf7c4h/content/dependencies/`를 지정합니다.

다음 표에는 AWS Cloud9를 사용할 수 있는 AWS 리전의 버킷 이름이 나열되어 있습니다.

AWS Cloud9 리전의 Amazon S3 버킷

AWS 리전	Bucket name
미국 동부(오하이오)	`static-us-east-2-prod-static-1c3sfcvf9hy4m`
미국 동부(버지니아 북부)	`static-us-east-1-prod-static-mft1klnkc4hl`

AWS 리전	Bucket name
미국 서부(오레건)	static-us-west-2-prod-static-p21mksqx9zlr
미국 서부(캘리포니아 북부)	static-us-west-1-prod-static-16d59zrrp01z0
아프리카(케이프타운)	static-af-south-1-prod-static-v6v7i5ypdppv
아시아 태평양(홍콩)	static-ap-east-1-prod-static-171xhpfkrorh6
아시아 태평양(뭄바이)	static-ap-south-1-prod-static-ykocre202i9d
아시아 태평양(오사카)	static-ap-northeast-3-prod-static-ivmxqzrx2ioi
아시아 태평양(서울)	static-ap-northeast-2-prod-static-1wxyctlhwiajm
아시아 태평양(싱가포르)	static-ap-southeast-1-prod-static-13ibpyrx4vk6d
아시아 태평양(시드니)	static-ap-southeast-2-prod-static-1cjsl8bx27rfu
아시아 태평양(도쿄)	static-ap-northeast-1-prod-static-4fwvbdisquj8
캐나다(중부)	static-ca-central-1-prod-static-g80lpejy486c
유럽(프랑크푸르트)	static-eu-central-1-prod-static-14lbgls2vrkh
유럽(아일랜드)	static-eu-west-1-prod-static-hld3vzaf7c4h

AWS 리전	Bucket name
유럽(런던)	static-eu-west-2-prod-static-36lbg202837x
유럽(밀라노)	static-eu-south-1-prod-static-1379tzkd3ni7d
유럽(파리)	static-eu-west-3-prod-static-1rwpkf766ke58
유럽(스톡홀름)	static-eu-north-1-prod-static-1qzw982y7yu7e
중동(바레인)	static-me-south-1-prod-static-gmljex38qtqx
남아메리카(상파울루)	static-sa-east-1-prod-static-1cl8k0y7opidt
이스라엘(텔아비브)	static-il-central-1-prod-static-k02vrnhcesue

9. 엔드포인트 생성을 선택합니다.

   올바른 구성 정보를 제공한 경우, 생성된 엔드포인트의 ID가 메시지에 표시됩니다.

10. IDE가 Amazon S3 버킷에 액세스할 수 있는지 확인하려면 메뉴 모음에서 [창(Window)], [새 터미널(New Terminal)]을 선택하여 터미널 세션을 시작합니다. 그리고 나서 다음 명령을 실행하여 `{bucket_name}`을 해당 리전용 버킷의 실제 이름으로 바꿉니다.

```
ping {bucket_name}.s3.{region}.amazonaws.com.
```

   예를 들어 S3 버킷에 대한 엔드포인트를 미국 동부(버지니아 북부) 리전에서 생성한 경우 다음 명령을 실행합니다.

```
ping static-us-east-1-prod-static-mft1klnkc4hl.s3.us-east-1.amazonaws.com
```

   ping 응답을 받으면 이는 IDE가 버킷 및 해당 종속 구성 요소에 액세스할 수 있음을 의미합니다.

이 기능에 관한 자세한 내용은 AWS PrivateLink 가이드에서 Amazon S3의 엔드포인트를 참조하세요.

프라이빗 연결을 위한 VPC 엔드포인트 구성

Systems Manager를 이용한 액세스 옵션을 사용하여 서브넷에서 인스턴스를 시작할 경우, 해당 보안 그룹에는 들어오는 네트워크 트래픽을 허용하는 인바운드 규칙이 없습니다. 하지만 이 보안 그룹에는 인스턴스의 아웃바운드 트래픽을 허용하는 아웃바운드 규칙이 있습니다. 이는 AWS Cloud9 IDE를 최신 상태로 유지하는 데 필요한 패키지와 라이브러리를 다운로드하는 데 필요합니다.

인스턴스에 대한 아웃바운드 및 인바운드 트래픽을 방지하려면 Systems Manager용 Amazon VPC 엔드포인트를 생성하고 구성하세요. 인터페이스 VPC 엔드포인트(인터페이스 엔드포인트)를 사용하면 AWS PrivateLink에 의해 구동되는 서비스에 연결할 수 있습니다. AWS PrivateLink는 프라이빗 IP 주소를 사용하여 Amazon EC2 및 Systems Manager API에 비공개로 액세스할 수 있는 기술입니다. Systems Manager를 사용하도록 VPC 엔드포인트를 구성하려면 이 지식 센터 리소스에서 제공하는 지침을 따르세요.

> **Warning**
>
> 인바운드 또는 아웃바운드 네트워킹 트래픽을 허용하지 않는 보안 그룹을 구성한다고 가정해 보겠습니다. 그러면 AWS Cloud9 IDE를 지원하는 EC2 인스턴스가 인터넷에 액세스할 수 없습니다. VPC용 Amazon S3 엔드포인트를 생성하여 신뢰할 수 있는 S3 버킷에 포함된 종속 구성 요소에 대한 액세스를 허용해야 합니다. 또한 인터넷에 액세스할 수 없는 경우 AWS Lambda와 같은 일부 AWS 서비스도 정상적으로 작동하지 않을 수 있습니다.
>
> AWS PrivateLink를 사용하면 VPC 엔드포인트를 통해 처리된 기가바이트당 데이터 처리 요금이 부과됩니다. 이는 트래픽의 소스 또는 대상에 관계없이 적용됩니다. 자세한 내용은 AWS PrivateLink 요금을 참조하세요.

AWS Cloud9에서 환경 열기

이 절차에서는 AWS Cloud9에서 환경을 여는 방법을 설명합니다.

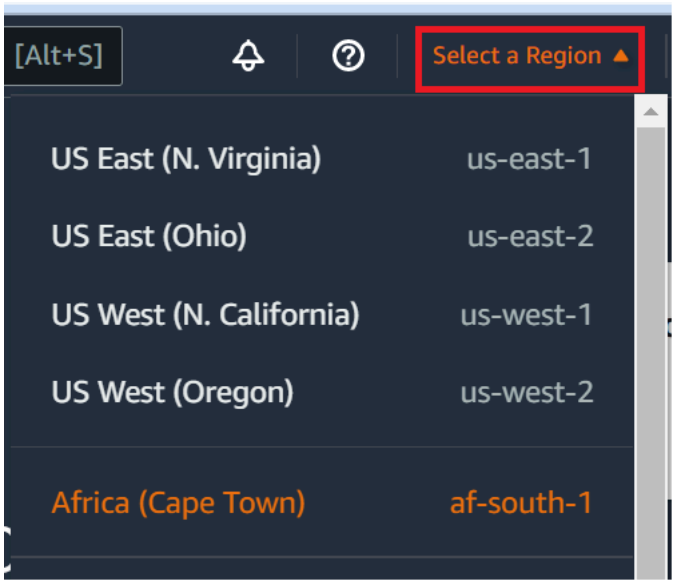
> **Note**
>
> 이 절차에서는 AWS Cloud9 개발 환경을 이미 생성했다고 가정합니다. 환경을 생성하려면 환경 생성을 참조하세요.

1. 다음과 같이 AWS Cloud9 콘솔에 로그인합니다.

- AWS 계정를 혼자만 사용하는 경우 또는 단일 AWS 계정의 IAM 사용자인 경우 https://console.aws.amazon.com/cloud9/로 이동합니다.
- 조직에서 AWS IAM Identity Center을 사용하는 경우 로그인 지침은 AWS 계정 관리자에게 문의하세요.

> **Important**
>
> AWS 계정에서 로그아웃하더라도 이후 최대 5분간 AWS Cloud9 IDE에 계속 액세스할 수 있습니다. 필요한 권한이 만료되면 액세스가 거부됩니다.

2. 상단 탐색 모음에서 환경이 위치한 AWS 리전을 선택합니다.



3. 환경 목록에서 열려는 환경에 대해 다음 작업 중 하나를 수행합니다.
   - 카드 내에서 Open in Cloud9(Cloud9에서 열기) 링크를 선택합니다.
   - 카드를 선택하고 Open in Cloud9(Cloud9에서 열기) 버튼을 선택합니다.

콘솔에 환경이 표시되지 않으면 다음 작업 중 하나 이상을 수행하여 표시해 보세요.

- Environments(환경) 페이지의 드롭다운 메뉴 표시줄에서 다음 중 하나 이상을 선택합니다.
  - My environments(내 환경)를 선택하여 선택한 AWS 리전 및 AWS 계정 내에서 AWS 엔터티가 소유하고 있는 모든 환경을 표시합니다.

  - Shared with you(사용자와 공유)를 선택하여 선택한 AWS 리전 및 AWS 계정 내에서 AWS 엔터티가 초대된 모든 환경을 표시합니다.
  - All account environments(모든 계정 환경)를 선택하여 AWS 엔터티가 표시할 권한을 가지고 있는 선택한 AWS 리전 및 AWS 계정 내의 모든 환경을 표시합니다.
- 본인이 환경의 멤버라고 생각하지만 Shared with you(사용자와 공유) 목록에 환경이 표시되지 않는 경우 환경 소유자에 문의하여 확인하세요.
- 상단 탐색 모음에서 다른 AWS 리전을 선택합니다.

AWS Cloud9의 환경에서 AWS 서비스 호출

AWS Cloud9 개발 환경에서 AWS 서비스를 호출할 수 있습니다. 예를 들어 다음 작업을 수행할 수 있습니다.

- Amazon Simple Storage Service(Amazon S3) 버킷에서 데이터를 업로드하고 다운로드합니다.
- Amazon Simple Notification Service(Amazon SNS) 주제를 통해 브로드캐스트 알림을 보냅니다.
- Amazon DynamoDB(DynamoDB) 데이터베이스에서 데이터를 읽고 씁니다.

여러 가지 방법으로 환경에서 AWS 서비스를 호출할 수 있습니다. 예를 들어, AWS Command Line Interface(AWS CLI) 또는 AWS CloudShell을 사용하여 터미널 세션에서 명령을 실행할 수 있습니다. 환경 내에서 실행하는 코드로부터 AWS 서비스를 호출할 수도 있습니다. JavaScript, Python, Ruby, PHP, Go 및 C++ 등의 프로그래밍 언어용 AWS SDK를 사용하여 이 작업을 수행할 수 있습니다. 자세한 내용은 AWS Command Line Interface 사용 설명서의 AWS CLI 및 aws-shell 샘플과 AWS SDK를 참조하세요.

AWS CLI, AWS CloudShell 또는 코드가 AWS 서비스를 호출할 때마다 AWS CLI, AWS CloudShell 또는 코드는 호출과 함께 AWS 액세스 보안 인증 정보 세트를 제공해야 합니다. 이러한 자격 증명은 호출자가 호출을 수행할 적절한 권한을 가지고 있는지 여부를 확인합니다. 자격 증명으로 적절한 권한이 확인되지 않는 경우 호출이 실패합니다.

환경에 자격 증명을 제공하는 여러 가지 방법이 있습니다. 다음 표에서는 몇 가지 접근 방식에 대해 설명합니다.

환경 유형	접근 방식
EC2	AWS 관리형 임시 자격 증명 사용

환경 유형	접근 방식
	EC2 환경에는 이 접근 방식을 사용하는 것이 좋습니다. AWS 관리형 임시 자격 증명은 AWS 보안 모범 사례를 따르면서 사용자를 대신하여 EC2 환경에서 AWS 액세스 자격 증명을 관리합니다. EC2 환경을 사용하는 경우 이 주제의 나머지 부분은 건너뛸 수 있습니다. AWS 관리형 임시 자격 증명이 환경에 이미 설정되어 있기 때문입니다. 자세한 내용은 AWS 관리형 임시 자격 증명을 참조하세요.
EC2	IAM 인스턴스 프로파일을 인스턴스에 연결합니다. 어떠한 이유로든 AWS 관리형 임시 보안 인증 정보를 사용할 수 없는 경우에만 이 접근 방식을 사용하세요. AWS 관리형 임시 자격 증명과 마찬가지로, 인스턴스 프로파일은 사용자를 대신하여 AWS 액세스 자격 증명을 관리합니다. 하지만 사용자가 직접 인스턴스 프로파일을 생성하고 관리하고 Amazon EC2 인스턴스에 연결해야 합니다. 지침은 인스턴스 프로파일을 생성하고 사용하여 임시 자격 증명 관리를 참조하세요.

환경 유형	접근 방식
EC2 또는 SSH	영구 AWS 액세스 자격 증명을 환경 내에 저장합니다. 이 접근 방식은 임시 AWS 액세스 자격 증명을 사용하는 것보다 안전하지 않습니다. 하지만 이 방식은 SSH 환경에 지원되는 유일한 접근 방식입니다. 지침은 영구 액세스 자격 증명을 생성하여 환경에 저장을 참조하십시오.
EC2 또는 SSH	영구 AWS 액세스 자격 증명을 코드에 직접 삽입합니다. 이 접근 방식은 AWS 보안 모범 사례를 준수하지 않으므로 사용하지 않는 것이 좋습니다. 이 접근 방식은 사용하지 않는 것이 좋으므로 이 주제에서 다루지 않습니다.

인스턴스 프로파일을 생성하고 사용하여 임시 자격 증명 관리

> **Note**
>
> AWS Cloud9 SSH 개발 환경에는 이 절차를 사용할 수 없습니다. 그 대신, 영구 액세스 자격 증명을 생성하여 환경에 저장으로 건너뛰십시오.
> 인스턴스 프로파일 대신 AWS 관리형 임시 자격 증명을 사용하는 것이 좋습니다. 어떠한 이유로든 AWS 관리형 임시 자격 증명을 사용할 수 없는 경우에만 이러한 지침을 따르세요. 자세한 내용은 AWS 관리형 임시 자격 증명을 참조하세요.

이 절차에서는 IAM 및 Amazon EC2를 사용하여 IAM 인스턴스 프로파일을 생성하고 환경에 연결되는 Amazon EC2 인스턴스에 연결합니다. 이 인스턴스 프로파일은 사용자를 대신하여 임시 자격 증명을 관리합니다. 이 절차에서는 AWS Cloud9에 환경을 이미 생성했다고 가정합니다. 환경을 생성하려면 환경 생성을 참조하세요.

IAM 및 Amazon EC2 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 이러한 작업을 완료할 수 있습니다.

IAM 콘솔을 사용하여 인스턴스 프로파일 생성

> **Note**
>
> 인스턴스 프로파일이 포함된 IAM 역할이 이미 있는 경우 Amazon EC2 콘솔을 사용하여 인스턴스 프로파일을 인스턴스에 연결로 건너뜁니다.

1. https://console.aws.amazon.com/iam에서 IAM 콘솔에 로그인합니다.

   이 단계에서는 AWS 계정의 관리자 수준 보안 인증 정보를 사용하여 로그인하는 것이 좋습니다. 이렇게 할 수 없으면 AWS 계정 관리자에게 문의하세요.

2. 탐색 모음에서 [Roles]를 선택합니다.

   > **Note**
   >
   > 자체적으로 IAM 콘솔을 사용하여 인스턴스 프로파일을 생성할 수 없습니다. 인스턴스 프로파일이 포함된 IAM 역할을 생성해야 합니다.

3. 역할 생성을 선택합니다.
4. Select type of trusted entity(신뢰할 수 있는 엔터티 유형 선택) 페이지에서 AWS 서비스가 이미 선택된 상태에서 Choose the service that will use this role(이 역할을 사용할 서비스 선택)에서 EC2를 선택합니다.
5. Select your use case(사용 사례 선택)에서 EC2를 선택합니다.
6. 다음: 권한을 선택합니다.
7. Attach permissions policies(권한 정책 연결) 페이지의 정책 목록에서 AdministratorAccess 옆에 있는 상자를 선택한 다음 Next: Review(다음: 검토)를 선택합니다.

   > **Note**
   >
   > AdministratorAccess 정책은 AWS 계정 전반에 걸쳐 모든 AWS 작업과 리소스에 대한 무제한 액세스를 허용합니다. 이 정책은 실험 용도로만 사용하세요. 자세한 내용은 IAM 사용 설명서에서 IAM 정책을 참조하세요.

8. Review(검토) 페이지의 Role Name(역할 이름)에 역할의 이름을 입력합니다(예: `my-demo-cloud9-instance-profile`).
9. 역할 생성을 선택합니다.

Amazon EC2 콘솔을 사용하여 인스턴스 프로파일을 인스턴스에 연결로 건너뛰십시오.

AWS CLI를 사용하여 인스턴스 프로파일 생성

> **Note**
>
> 인스턴스 프로파일이 포함된 IAM 역할이 이미 있는 경우 AWS CLI를 사용하여 인스턴스 프로파일을 인스턴스에 연결로 건너뜁니다.
> 이 주제의 경우 AWS 계정의 관리자 수준 보안 인증 정보를 사용하여 AWS CLI를 구성하는 것이 좋습니다. 이렇게 할 수 없으면 AWS 계정 관리자에게 문의하세요.

> **Note**
>
> AWS 관리형 임시 자격 증명을 사용하는 경우 AWS Cloud9 IDE의 터미널 세션을 사용하여 이 섹션의 일부 또는 모든 명령을 실행할 수 없습니다. AWS 보안 모범 사례를 따르기 위해 AWS 관리형 임시 자격 증명은 일부 명령의 실행을 허용하지 않습니다. 대신에 별도의 AWS Command Line Interface(AWS CLI) 설치에서 이러한 명령을 실행할 수 있습니다.

1. AWS에서 인스턴스 프로파일의 필수 IAM 역할에 대한 신뢰 관계를 정의합니다. 이렇게 하려면 다음 내용이 포함된 파일을 생성한 다음 저장합니다(예: `my-demo-cloud9-instance-profile-role-trust.json`).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
```

```
  ]
}
```

2. 터미널 또는 명령 프롬프트를 사용하여 방금 이 파일을 저장한 디렉터리로 전환합니다.
3. 인스턴스 프로파일에 대한 IAM 역할을 생성합니다. 이렇게 하려면 IAM `create-role` 명령을 실행합니다. 이 명령을 실행할 때에는 새 IAM 역할의 이름(예: `my-demo-cloud9-instance-profile-role`)과 방금 저장한 파일의 이름을 지정합니다.

```
aws iam create-role --role-name my-demo-cloud9-instance-profile-role --assume-role-policy-document file://my-demo-cloud9-instance-profile-role-trust.json
```

4. AWS 액세스 권한을 인스턴스 프로파일의 IAM 역할에 연결합니다. 이렇게 하려면 IAM `attach-role-policy` 명령을 실행합니다. 기존 IAM 역할의 이름과 `AdministratorAccess`라는 AWS 관리형 정책의 Amazon 리소스 이름(ARN)을 지정합니다.

```
aws iam attach-role-policy --role-name my-demo-cloud9-instance-profile-role --policy-arn arn:aws:iam::aws:policy/AdministratorAccess
```

> **Note**
>
> AdministratorAccess 정책은 AWS 계정 전반에 걸쳐 모든 AWS 작업과 리소스에 대한 무제한 액세스를 허용합니다. 이 정책은 실험 용도로만 사용하세요. 자세한 내용은 IAM 사용 설명서에서 IAM 정책을 참조하세요.

5. 인스턴스 프로파일을 생성합니다. 이렇게 하려면 새 인스턴스 프로파일의 이름(예: `my-demo-cloud9-instance-profile`)을 지정하여 IAM `create-instance-profile` 명령을 실행합니다.

```
aws iam create-instance-profile --instance-profile-name my-demo-cloud9-instance-profile
```

6. IAM 역할을 인스턴스 프로파일에 추가합니다. 이렇게 하려면 기존 IAM 역할과 인스턴스 프로파일의 이름을 지정하여 IAM `add-role-to-instance-profile`을 실행합니다

```
aws iam add-role-to-instance-profile --role-name my-demo-cloud9-instance-profile-role
 --instance-profile-name my-demo-cloud9-instance-profile
```

AWS CLI를 사용하여 인스턴스 프로파일 생성으로 건너뜁니다.

Amazon EC2 콘솔을 사용하여 인스턴스 프로파일을 인스턴스에 연결

1. https://console.aws.amazon.com/ec2에서 Amazon EC2 콘솔에 로그인합니다.

   이 단계에서는 AWS 계정의 관리자 수준 보안 인증 정보를 사용하여 로그인하는 것이 좋습니다. 이렇게 할 수 없으면 AWS 계정 관리자에게 문의하세요.
2. 탐색 모음에서 리전 선택기에 환경의 리전과 일치하는 AWS 리전이 표시되는지 확인합니다. 예를 들어 미국 동부(오하이오) 리전에서 환경을 생성한 경우 여기서도 리전 선택기에서 US East (Ohio)(미국 동부(오하이오))를 선택합니다.
3. Running Instances(실행 중인 인스턴스) 링크를 선택하거나, 탐색 창에서 Instances(인스턴스)를 확장한 다음 Instances(인스턴스)를 선택합니다.
4. 인스턴스 목록에서 [이름(Name)]에 해당 환경의 이름이 포함된 인스턴스를 선택합니다. 예를 들어 환경 이름이 `my-demo-environment`인 경우 [이름(Name)]에 my-demo-environment가 포함된 인스턴스를 선택합니다.
5. Actions(작업), Security(보안), Modify IAM role(IAM 역할 수정)을 선택합니다.

   > **Note**
   >
   > 역할을 인스턴스에 연결하고 있지만 역할에는 인스턴스 프로파일이 포함됩니다.

6. Modify IAM role(IAM 역할 수정) 페이지의 IAM role(IAM 역할)에서 식별한 역할의 이름 또는 이전 절차 생성한 역할의 이름을 선택한 다음 Apply(적용)를 선택합니다.
7. 다시 환경으로 돌아와 AWS CLI를 사용하여 `aws configure` 명령을 실행하거나 AWS CloudShell을 사용하여 `configure` 명령을 실행합니다. AWS Access Key ID(AWS 액세스 키 ID) 또는 AWS Secret Access Key(AWS 비밀 액세스 키)에는 값을 지정하지 않습니다(각 프롬프트가 나타나면 `Enter` 키를 누름). Default Region name(기본 리전 이름)에서 사용자와 가장 가까운 AWS 리전 또는 AWS 리소스가 위치한 리전을 지정합니다. 예를 들어, 미국 동부(오하이오) 리전의 경우 `us-east-2`입니다. 리전 목록은 Amazon Web Services 일반 참조의 AWS 리전 및 엔드포인트를 참조하세요. 선택적으로 Default output format(기본 출력 형식)의 값을 지정합니다(예: `json`).

이제 환경에서 AWS 서비스 호출을 시작할 수 있습니다. AWS CLI, aws-shell 또는 이 두 가지를 모두 사용하여 AWS 서비스를 호출하려면 AWS CLI 및 aws-shell 샘플을 참조하세요. 코드에서 AWS 서비스를 호출하려면 다른 자습서 및 샘플을 참조하세요.

AWS CLI를 사용하여 인스턴스 프로파일을 인스턴스에 연결

> **Note**
>
> AWS 관리형 임시 자격 증명을 사용하는 경우 AWS Cloud9 IDE의 터미널 세션을 사용하여 이 섹션의 일부 또는 모든 명령을 실행할 수 없습니다. AWS 보안 모범 사례를 따르기 위해 AWS 관리형 임시 자격 증명은 일부 명령의 실행을 허용하지 않습니다. 대신에 별도의 AWS Command Line Interface(AWS CLI) 설치에서 이러한 명령을 실행할 수 있습니다.

1. Amazon EC2 `associate-iam-instance-profile` 명령을 실행합니다. 환경에 대해 인스턴스 프로파일의 이름과 Amazon EC2 인스턴스의 ID 및 AWS 리전 ID를 지정합니다.

   ```
   aws ec2 associate-iam-instance-profile --iam-instance-profile Name=my-demo-cloud9-instance-profile --region us-east-2 --instance-id i-12a3b45678cdef9a0
   ```

   위의 명령에서 `us-east-2`를 인스턴스의 AWS 리전 ID로 바꾸고 `i-12a3b45678cdef9a0`을 인스턴스 ID로 바꿉니다.

   인스턴스 ID를 가져오려면 예를 들어 환경의 이름과 AWS 리전 ID를 지정하여 Amazon EC2 `describe-instances` 명령을 실행할 수 있습니다.

   ```
   aws ec2 describe-instances --region us-east-2 --filters Name=tag:Name,Values=*my-environment* --query "Reservations[*].Instances[*].InstanceId" --output text
   ```

   위의 명령에서 `us-east-2`를 인스턴스의 AWS 리전 ID로 바꾸고 `my-environment`를 환경의 이름으로 바꿉니다.

2. 다시 환경으로 돌아와 AWS CLI를 사용하여 `aws configure` 명령을 실행하거나 aws-shell을 사용하여 `configure` 명령을 실행합니다. AWS 액세스 키 ID 또는 AWS 비밀 액세스 키에 값을 지정하지 마십시오. 각 프롬프트가 표시된 후 `Enter` 키를 누릅니다. Default Region name(기본 리전 이름)에서 사용자와 가장 가까운 AWS 리전 또는 AWS 리소스가 위치한 리전을 지정합니다. 예를 들어, 미국 동부(오하이오) 리전의 경우 `us-east-2`입니다. 리전 목록은 Amazon Web Services 일반 참조의 AWS 리전 및 엔드포인트를 참조하세요. 선택적으로 Default output format(기본 출력 형식)의 값을 지정합니다(예: `json`).

이제 환경에서 AWS 서비스 호출을 시작할 수 있습니다. AWS CLI, aws-shell 또는 이 두 가지를 모두 사용하여 AWS 서비스를 호출하려면 AWS CLI 및 aws-shell 샘플을 참조하세요. 코드에서 AWS 서비스를 호출하려면 다른 자습서 및 샘플을 참조하세요.

영구 액세스 자격 증명을 생성하여 환경에 저장

> **Note**
>
> AWS Cloud9 EC2 개발 환경을 사용하는 경우 AWS 영구 액세스 자격 증명 대신 AWS 관리형 임시 자격 증명을 사용하는 것이 좋습니다. AWS 관리형 임시 자격 증명으로 작업하려면 AWS 관리형 임시 자격 증명 섹션을 참조하세요.

이 섹션에서는 AWS Identity and Access Management(IAM)을 사용하여 영구 보안 인증 정보 세트를 생성합니다. AWS CLI, aws-shell 또는 사용자 코드는 AWS 서비스 호출 시 이 보안 인증 정보 세트를 사용할 수 있습니다. 이 세트에는 AWS 계정에서 사용자에게 고유한 AWS 액세스 키 ID와 AWS 비밀 액세스 키가 포함됩니다. AWS 액세스 키 ID와 AWS 비밀 액세스 키가 이미 있는 경우 해당 자격 증명을 기록한 다음 환경에 영구 액세스 자격 증명 저장으로 건너뜁니다.

IAM 콘솔 또는 AWS CLI를 사용하여 영구 자격 증명 세트를 생성할 수 있습니다.

프로그래밍 방식 액세스 권한 부여

사용자가 AWS Management Console 외부에서 AWS와 상호 작용하려면 프로그래밍 방식의 액세스가 필요합니다. 프로그래밍 방식으로 액세스를 부여하는 방법은 AWS에 액세스하는 사용자 유형에 따라 다릅니다.

사용자에게 프로그래밍 방식 액세스 권한을 부여하려면 다음 옵션 중 하나를 선택합니다.

프로그래밍 방식 액세스가 필요한 사용자는 누구인가요?	To	By
작업 인력 ID (IAM Identity Center에서 관리되는 사용자)	임시 보안 인증 정보로 AWS CLI, AWS SDK 또는 AWS API에 대한 프로그래밍 요청에 서명합니다.	사용하고자 하는 인터페이스에 대한 지침을 따릅니다. • AWS CLI에 대해서는 AWS Command Line Interface 사용 설명서에서 AWS IAM Identity Center을 사용하도

프로그래밍 방식 액세스가 필요한 사용자는 누구인가요?	To	By
		록 AWS CLI 구성을 참조하세요. • AWS SDK, 도구,AWS API에 대해서는 AWS SDK 및 도구 참조 가이드에서 IAM Identity Center 인증을 참조하세요.
IAM	임시 보안 인증 정보로 AWS CLI, AWS SDK 또는 AWS API에 대한 프로그래밍 요청에 서명합니다.	IAM 사용 설명서의 AWS 리소스와 함께 임시 보안 인증 정보 사용에 나와 있는 지침을 따르세요.
IAM	(권장되지 않음) 장기 보안 인증 정보로 AWS CLI, AWS SDK 또는 AWS API에 대한 프로그래밍 요청에 서명합니다.	사용하고자 하는 인터페이스에 대한 지침을 따릅니다. • AWS CLI에 대해서는 AWS Command Line Interface 사용 설명서에서 IAM 사용자 보안 인증 정보를 사용한 인증을 참조하세요. • AWS SDK와 도구에 대해서는 AWS SDK 및 도구 참조 가이드에서 장기 보안 인증 정보를 사용한 인증을 참조하세요. • AWS API에 대해서는 IAM 사용 설명서에서 IAM 사용자의 액세스 키 관리를 참조하세요.

AWS CLI를 사용하여 영구 액세스 자격 증명 생성

> **Note**
>
> 이 섹션에서는 AWS 계정의 관리자 수준 보안 인증 정보를 사용하여 AWS CLI를 구성하는 것이 좋습니다. 이렇게 할 수 없으면 AWS 계정 관리자에게 문의하세요.

> **Note**
>
> AWS 관리형 임시 자격 증명을 사용하는 경우 AWS Cloud9 IDE의 터미널 세션을 사용하여 이 섹션의 일부 또는 모든 명령을 실행할 수 없습니다. AWS 보안 모범 사례를 따르기 위해 AWS 관리형 임시 자격 증명은 일부 명령의 실행을 허용하지 않습니다. 대신에 별도의 AWS Command Line Interface(AWS CLI) 설치에서 이러한 명령을 실행할 수 있습니다.

IAM `create-access-key` 명령을 실행하여 사용자에 대한 새 AWS 액세스 키 및 해당 AWS 비밀 액세스 키를 생성합니다.

```
aws iam create-access-key --user-name MyUser
```

위의 명령에서 `MyUser`를 사용자의 이름으로 바꿉니다.

표시되는 `AccessKeyId` 및 `SecretAccessKey` 값을 안전한 위치에 저장합니다. IAM `create-access-key` 명령을 실행한 후에는 이 시점에서만 AWS CLI를 사용하여 사용자의 AWS 비밀 액세스 키를 볼 수 있습니다. 나중에 필요에 따라 사용자의 새 AWS 비밀 액세스 키를 생성하려면 IAM 사용 설명서에서 액세스 키 생성, 수정 및 확인(API, CLI, PowerShell)을 참조하세요.

환경에 영구 액세스 자격 증명 저장

이 절차에서는 AWS Cloud9 IDE를 사용하여 영구 AWS 액세스 자격 증명을 환경에 저장합니다. 이 절차에서는 AWS Cloud9에 환경을 이미 생성하고 환경을 열었으며 웹 브라우저에 AWS Cloud9 IDE를 표시한 상태라고 가정합니다. 자세한 내용은 환경 생성 및 환경 열기를 참조하십시오.

> **Note**
>
> 다음 절차에서는 환경 변수를 사용하여 영구 액세스 자격 증명을 저장하는 방법을 보여 줍니다. AWS CLI 또는 aws-shell이 환경에 설치되어 있다면 AWS CLI의 **`aws configure`** 명령 또는 aws-shell의 **`configure`** 명령을 대신 사용하여 영구 액세스 보안 인증 정보를 저장할 수

있습니다. 자세한 내용은 AWS Command Line Interface 사용 설명서에서 빠른 구성을 참조하세요.

1. 터미널이 아직 시작되지 않은 경우 환경이 열려 있는 상태로 AWS Cloud9 IDE에서 새 터미널 세션을 시작합니다. 터미널 세션을 새로 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다.
2. 다음 각 명령을 한 번에 한 명령씩 실행하여 영구 액세스 자격 증명을 나타내는 로컬 환경 변수를 설정합니다. 이러한 명령에서 `AWS_ACCESS_KEY_ID:` 뒤에 AWS 액세스 키 ID를 입력합니다. `AWS_SECRET_ACCESS_KEY` 뒤에 AWS 비밀 액세스 키를 입력합니다. `AWS_DEFAULT_REGION_ID` 다음에는 가장 가까운 AWS 리전(또는 선호하는 AWS 리전)과 관련된 AWS 리전 식별자를 입력하세요. 사용할 수 있는 식별자 목록은 Amazon Web Services 일반 참조의 및 엔드포인트를 참조하세요. 예를 들어, 미국 동부(오하이오) 리전의 경우 `us-east-2`를 사용합니다.

```
export AWS_ACCESS_KEY_ID=
export AWS_SECRET_ACCESS_KEY=
export AWS_DEFAULT_REGION=
```

3. 위의 환경 변수는 현재 터미널 세션에만 유효합니다. 이러한 환경 변수를 여러 터미널 세션 간에 사용할 수 있도록 하려면 다음과 같이 이러한 환경 변수를 셸 프로파일 파일에 사용자 환경 변수로 추가해야 합니다.
   a. IDE의 [환경(Environment)] 창에서 기어 모양 아이콘을 선택한 다음 [즐겨찾기에 홈 표시(Show Home in Favorites)]를 선택합니다. 이 단계를 반복하고 Show Hidden Files(숨겨진 파일 표시)도 선택합니다.
   b. `~/.bashrc` 파일을 엽니다.
   c. 다음 코드를 파일의 끝 부분에 입력하거나 붙여 넣습니다. 이러한 명령에서 `AWS_ACCESS_KEY_ID:` 뒤에 AWS 액세스 키 ID를 입력합니다. `AWS_SECRET_ACCESS_KEY` 뒤에 AWS 비밀 액세스 키를 입력합니다. `AWS_DEFAULT_REGION_ID` 다음에는 가장 가까운 AWS 리전(또는 선호하는 AWS 리전)과 관련된 AWS 리전 식별자를 입력하세요. 사용할 수 있는 식별자 목록은 Amazon Web Services 일반 참조의 및 엔드포인트를 참조하세요. 예를 들어, 미국 동부(오하이오) 리전의 경우 `us-east-2`를 사용합니다.

```
export AWS_ACCESS_KEY_ID=
export AWS_SECRET_ACCESS_KEY=
export AWS_DEFAULT_REGION=
```

   d. 파일을 저장합니다.

e. `~/.bashrc` 파일을 소싱하여 이러한 새 환경 변수를 로드합니다.

```
. ~/.bashrc
```

이제 환경에서 AWS 서비스 호출을 시작할 수 있습니다. AWS CLI 또는 aws-shell를 사용하여 AWS 서비스를 호출하려면 AWS CLI 및 aws-shell 샘플을 참조하세요. 코드에서 AWS 서비스를 호출하려면 다른 자습서 및 샘플을 참조하세요.

AWS Cloud9에서 환경 설정 변경

AWS Cloud9 개발 환경의 기본 설정 또는 설정을 변경할 수 있습니다.

- 환경 기본 설정 변경
- 콘솔을 사용하여 환경 설정 변경
- 코드를 사용하여 환경 설정 변경

환경 기본 설정 변경

1. 설정을 변경하려는 환경을 엽니다. 환경을 열려면 환경 열기를 참조하세요.
2. AWS Cloud9 IDE의 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
3. [기본 설정(Preferences)] 창에서 [프로젝트 설정(Project Settings)]을 선택합니다.
4. 사용 가능한 프로젝트 설정을 원하는 대로 변경합니다. 여기에는 [코드 편집기(Ace)(Code Editor (Ace))] 및 [파일에서 찾기(Find in Files)]와 같은 설정이 포함됩니다.

> **Note**
>
> 자세한 내용은 사용자가 변경할 수 있는 프로젝트 설정을 참조하세요.

AWS Cloud9 IDE에서 환경의 제한 시간 조정하기

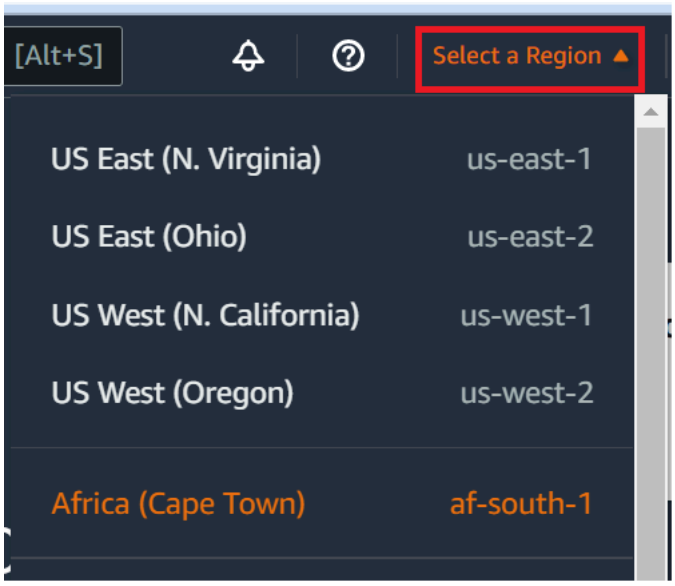
다음 단계에서는 AWS Cloud9 IDE에서 Amazon EC2 환경의 제한 시간을 업데이트하는 방법을 간략하게 설명합니다. 이 시간은 환경이 중지되기까지의 시간입니다.

1. 구성하려는 환경을 엽니다.

2. AWS Cloud9 IDE의 메뉴 모음에서 AWS Cloud9 기본 설정을 선택합니다.
3. 기본 설정 창에서 Amazon EC2 인스턴스 섹션으로 스크롤합니다.
4. 사용 가능한 목록에서 제한 시간 값을 선택하고 업데이트합니다.

콘솔을 사용하여 환경 설정 변경

1. 다음과 같이 AWS Cloud9 콘솔에 로그인합니다.
   - AWS 계정를 혼자만 사용하는 경우 또는 단일 AWS 계정의 IAM 사용자인 경우 https://console.aws.amazon.com/cloud9/로 이동합니다.
   - 조직에서 AWS IAM Identity Center을 사용하는 경우 로그인 지침은 AWS 계정 관리자를 참조하세요.
2. 상단 탐색 모음에서 환경이 위치한 AWS 리전을 선택합니다.



3. 환경 목록에서 설정을 변경하려는 환경에 대해 다음 중 하나를 수행합니다.
   - 환경에 대한 카드의 제목을 선택합니다. 다음 페이지에서 View details(상세 정보 보기)를 선택합니다.
   - 환경에 대한 카드를 선택한 다음 View details(상세 정보 보기) 버튼을 선택합니다.
4. 변경한 후 [변경 내용 저장(Save changes)]을 선택합니다.

   AWS Cloud9 콘솔을 사용하여 다음 설정을 변경할 수 있습니다.

   - EC2 환경의 경우: [이름(Name)] 및 [설명(Description)]

- SSH 환경의 경우: [이름(Name)], [설명(Description)] [사용자(User)], [호스트(Host)], [포트(Port)], [환경 경로(Environment path)], [Node.js 바이너리 경로(Node.js binary path)] 및 [SSH 점프 호스트(SSH jump host)]

다른 설정을 변경하려면 다음을 수행합니다.

- EC2 환경에 대해 다음을 수행합니다.
  - [유형(Type)], [보안 그룹(Security groups)], [VPC], [서브넷(Subnet), [환경 경로(Environment path) 또는 [환경 ARN(Environment ARN]은 변경할 수 없습니다.
  - Permissions(권한) 또는 Number of members(멤버 수)의 경우 환경 멤버의 액세스 역할 변경, 사용자 제거, IAM 사용자 초대 및 다른 환경 멤버 제거를 참조하세요.
  - [EC2 인스턴스 유형(EC2 instance type)], [메모리(Memory)] 또는 [vCPU]의 경우 환경 이동 또는 크기 조정을 참조하세요.
- SSH 환경에 대해 다음을 수행합니다.
  - [유형(Type)] 또는 [환경 ARN(Environment ARN)]은 변경할 수 없습니다.
  - [권한(Permissions)] 또는 [멤버 수(Number of members)]의 경우 환경 멤버의 액세스 역할 변경, 사용자 제거, IAM 사용자 초대 및 다른 환경 멤버 제거를 참조하세요.

콘솔에 환경이 표시되지 않으면 다음 작업 중 하나 이상을 수행하여 표시해 보세요.

- Environments(환경) 페이지의 드롭다운 메뉴 표시줄에서 다음 중 하나 이상을 선택합니다.
  - My environments(내 환경)를 선택하여 선택한 AWS 리전 및 AWS 계정 내에서 AWS 엔터티가 소유하고 있는 모든 환경을 표시합니다.
  - Shared with you(사용자와 공유)를 선택하여 선택한 AWS 리전 및 AWS 계정 내에서 AWS 엔터티가 초대된 모든 환경을 표시합니다.
  - All account environments(모든 계정 환경)를 선택하여 AWS 엔터티가 표시할 권한을 가지고 있는 선택한 AWS 리전 및 AWS 계정 내의 모든 환경을 표시합니다.
- 본인이 환경의 멤버라고 생각하지만 Shared with you(사용자와 공유) 목록에 환경이 표시되지 않는 경우 환경 소유자에 문의하여 확인하세요.
- 상단 탐색 모음에서 다른 AWS 리전을 선택합니다.

코드를 사용하여 환경 설정 변경

코드를 사용하여 AWS Cloud9에서 환경의 설정을 변경하려면 다음과 같이 AWS Cloud9 update environment 작업을 호출합니다.

AWS CLI	update-environment
AWS SDK for C++	UpdateEnvironmentRequest, UpdateEnvironmentResult
AWS SDK for Go	UpdateEnvironment, UpdateEnvironmentRequest, UpdateEnvironmentWithContext
AWS SDK for Java	UpdateEnvironmentRequest, UpdateEnvironmentResult
AWS SDK for JavaScript	updateEnvironment
AWS SDK for .NET	UpdateEnvironmentRequest, UpdateEnvironmentResponse
AWS SDK for PHP	updateEnvironment
AWS SDK for Python (Boto)	update_environment
AWS SDK for Ruby	update_environment
AWS Tools for Windows PowerShell	Update-C9Environment
AWS Cloud9 API	UpdateEnvironment

AWS Cloud9의 공유 환경 작업

공유 환경은 여러 사용자가 참여하도록 초대된 AWS Cloud9 개발 환경입니다. 이 항목에서는 AWS Cloud9에서 환경을 공유하기 위한 지침과 공유 환경에 참여하는 방법에 대한 지침을 제공합니다.

소유한 환경에 참여하도록 사용자를 초대하려면 다음과 같은 일련의 절차 중 하나를 수행합니다. 초대하려는 사용자의 유형에 따라 선택합니다.

- 환경과 동일한 AWS 계정 사용자인 경우 환경과 동일한 계정의 사용자를 초대해야 합니다.
- 환경과 동일한 AWS 계정의 AWS Cloud9 관리자(특히 AWS 계정 루트 사용자, 관리자 사용자 또는 AWS 관리 정책 `AWSCloud9Administrator`가 연결된 사용자)인 경우 AWS Cloud9 관리자를 직접 초대해야 합니다. 환경과 동일한 계정에 사용자 초대를 참조하거나 AWS Cloud9 관리자가 자신

(또는 동일한 AWS 계정의 다른 사람)을 초대하도록 하세요. 환경과 동일한 계정에 AWS Cloud9 관리자가 자신 또는 다른 사람을 초대를 참조하세요.

공유 환경 사용 사례

공유 환경은 다음 사용 사례에서 유용합니다.

- 페어 프로그래밍(피어 프로그래밍이라고도 함): 두 명의 사용자가 단일 환경에서 동일한 코드를 함께 작업하는 것입니다. 페어 프로그래밍에서는 일반적으로 한 사용자가 코드를 작성하고, 다른 사용자가 그 코드를 관찰합니다. 관찰자는 코드 작성자에게 바로 조언과 피드백을 제공합니다. 이러한 역할은 프로젝트 진행 중 종종 바뀝니다. 공유 환경에서 페어 프로그래머로 구성된 팀은 대체로 단일 머신에서 작업합니다. 한 번에 한 명의 사용자만 코드를 작성할 수 있습니다. 공유 환경을 사용하면 두 사용자는 각자의 컴퓨터에서 동시에 작업할 수 있습니다. 또한 다른 물리적인 사무실에 있더라도 각자의 위치에서 동시에 코드를 작성할 수 있습니다.
- 컴퓨터 공학 수업: 이 기능은 교사 또는 조교가 학생의 환경에 액세스하려고 할 때 유용합니다. 이렇게 하는 이유는 학생의 과제를 검토하거나 실시간으로 환경과 관련된 문제를 해결하기 위한 것일 수 있습니다. 또한 학생은 학습 친구와 함께 공유 과제 프로젝트를 진행하고 단일 환경에서 실시간으로 함께 코드를 작성할 수 있습니다. 학생들은 운영 체제와 웹 브라우저 유형이 다른 위치에 있더라도 이와 같이 함께 작업이 가능합니다.
- 여러 사용자가 실시간으로 동일한 코드에 대해 공동 작업해야 하는 기타 모든 상황

환경 멤버 액세스 역할 정보

AWS Cloud9에서 환경을 공유하거나 공유 환경에 참여하려면 먼저 공유 환경에 대한 액세스 권한 수준을 이해해야 합니다. 이러한 권한 수준을 환경 멤버 액세스 역할이라고 합니다.

AWS Cloud9의 공유 환경은 소유자, 읽기/쓰기, 읽기 전용이라는 세 가지 환경 멤버 액세스 역할을 제공합니다.

- 소유자는 환경에 대한 전체 제어 권한을 가집니다. 각 환경에는 환경 작성자인 소유자가 한 명 있습니다. 소유자는 다음 작업을 수행할 수 있습니다.
  - 환경의 멤버 추가, 변경 및 제거
  - 파일 열기, 보기 및 편집
  - 코드 실행
  - 환경 설정 변경
  - 다른 멤버와 채팅

  - 기존 채팅 메시지 삭제

  AWS Cloud9 IDE에서 환경 소유자는 읽기+쓰기 액세스 권한으로 표시됩니다.
- 읽기/쓰기 멤버는 다음 작업을 수행할 수 있습니다.
  - 파일 열기, 보기 및 편집
  - 코드 실행
  - AWS Cloud9 IDE 내에서 다양한 환경 설정 변경
  - 다른 멤버와 채팅
  - 기존 채팅 메시지 삭제

  AWS Cloud9 IDE에서 읽기/쓰기 멤버는 읽기+쓰기 액세스 권한으로 표시됩니다.
- 읽기 전용 멤버는 다음 작업을 수행할 수 있습니다.
  - 파일 열기 및 보기
  - 다른 멤버와 채팅
  - 기존 채팅 메시지 삭제

  AWS Cloud9 IDE에서 읽기 전용 멤버는 읽기 전용 액세스 권한으로 표시됩니다.

사용자가 환경 소유자 또는 멤버가 되기 위해서는 다음 기준 중 하나를 충족해야 합니다.

- 사용자가 AWS 계정 루트 사용자입니다.
- 사용자가 관리자 사용자입니다. 자세한 내용은 IAM 사용 설명서에서 첫 번째 IAM 관리자 및 그룹 만들기를 참조하세요.
- IAM 그룹에 속한 사용자, 역할을 맡은 사용자 또는 역할을 맡은 페더레이션 사용자. 그리고 이러한 역할 또는 그룹에는 AWS 관리형 정책 `AWSCloud9Administrator`나 `AWSCloud9User`(또는 멤버만 되려면 `AWSCloud9EnvironmentMember`)가 연결되어 있어야 합니다. 자세한 내용은 AWS 관리형(미리 정의된) 정책을 참조하세요.
  - 앞서 언급한 정책 중 하나를 IAM 그룹에 연결하기 위해 다음 절차에서 설명하는 대로 AWS Management Console 또는 AWS 명령줄 인터페이스(AWS CLI)를 사용할 수 있습니다.
  - 사용자 또는 페더레이션 사용자가 수임할 이전 관리형 정책 중 하나를 사용하여 IAM에서 역할을 만들 수 있습니다. 자세한 내용은 IAM 사용 설명서의 역할 생성을 참조하세요. 사용자 또는 페더레이션 사용자가 역할을 수임하게 하려면 IAM 사용 설명서의 IAM 역할 사용에서 역할 수임 범위를 참조하세요.

콘솔을 사용하여 그룹에 AWS Cloud9에 대한 AWS 관리형 정책 연결

다음 절차에서는 콘솔을 사용하여 AWS Cloud9에 대한 AWS 관리형 정책을 그룹에 연결하는 방법을 간략하게 설명합니다.

1. 아직 로그인하지 않았다면 AWS Management Console에 로그인합니다.

   이 단계에서는 AWS 계정의 IAM 관리자 수준 보안 인증 정보를 사용하여 로그인하는 것이 좋습니다. 이렇게 할 수 없으면 AWS 계정 관리자에게 문의하세요.
2. IAM 콘솔(IAM console)을 엽니다. 이렇게 하려면 콘솔 탐색 모음에서 서비스를 선택합니다. 그런 다음 IAM을 선택합니다.
3. 그룹을 선택합니다.
4. 그룹의 이름을 선택합니다.
5. 권한 탭의 관리형 정책에서 정책 연결을 선택합니다.
6. 정책 이름 목록에서 다음 상자 중 하나를 선택합니다.
   - AWSCloud9User(기본 설정) 또는 AWSCloud9Administrator: 그룹의 각 사용자가 환경 소유자가 되도록 합니다.
   - AWSCloud9EnvironmentMember: 그룹의 각 사용자가 멤버만 되도록 합니다.

   목록에 이러한 정책 이름 중 하나가 보이지 않으면 표시할 정책 이름을 검색 상자에 입력합니다.
7. 정책 연결(Attach policies)을 선택합니다.

AWS CLI를 사용하여 그룹에 AWS Cloud9에 대한 AWS 관리형 정책 연결

> **Note**
>
> AWS 관리형 임시 자격 증명을 사용하는 경우 AWS Cloud9 IDE의 터미널 세션을 사용하여 이 섹션의 일부 또는 모든 명령을 실행할 수 없습니다. AWS 보안 모범 사례를 따르기 위해 AWS 관리형 임시 자격 증명은 일부 명령의 실행을 허용하지 않습니다. 대신에 별도의 AWS Command Line Interface(AWS CLI) 설치에서 이러한 명령을 실행할 수 있습니다.

IAM `attach-group-policy` 명령을 실행하여 AWS Cloud9에 대한 AWS 관리형 정책을 그룹에 연결합니다. 그룹 이름과 정책의 Amazon 리소스 이름(ARN)을 다음과 같이 지정합니다.

```
aws iam attach-group-policy --group-name MyGroup --policy-arn arn:aws:iam::aws:policy/
POLICY_NAME
```

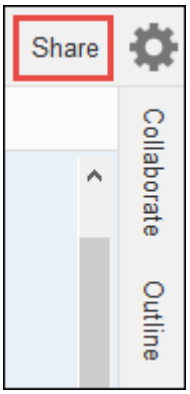
앞의 명령에서 `MyGroup`을 그룹의 이름으로 바꿉니다. `POLICY_NAME`은 다음 AWS 관리형 정책 중 하나의 이름으로 바꿉니다.

- `AWSCloud9User`(기본 설정) 또는 `AWSCloud9Administrator`: 그룹의 각 사용자가 환경 소유자가 되도록 합니다.
- `AWSCloud9EnvironmentMember`: 그룹의 각 사용자가 멤버만 되도록 합니다.

환경과 동일한 계정의 사용자 초대

이 섹션의 지침을 사용하여 AWS 계정에서 소유한 AWS Cloud9 개발 환경을 동일한 계정의 사용자와 공유합니다.

1. 초대하려는 사용자가 다음 유형의 사용자 중 하나가 아니라고 가정해 보겠습니다. 초대하려는 사용자에게 해당 환경 멤버 액세스 역할이 이미 있는지 확인합니다. 지침은 환경 멤버 액세스 역할 정보를 참조하십시오.
   - AWS 계정 루트 사용자.
   - 관리자 사용자.
   - IAM 그룹에 속한 사용자, 역할을 맡은 사용자 또는 역할을 맡은 페더레이션 사용자. 그리고 이러한 역할 또는 그룹에는 AWS 관리형 정책 `AWSCloud9Administrator`가 연결되어 있어야 합니다.
2. 소유하고 있고 사용자를 초대하려는 환경을 아직 열지 않은 경우 해당 환경을 엽니다.
3. AWS Cloud9 IDE의 메뉴 모음에서 다음 중 하나를 수행합니다.
   - Window, Share(창, 공유)를 선택합니다.
   - [기본 설정(Preferences)] 기어 아이콘 옆에 있는 [공유(Share)]를 선택합니다.



4. Share this environment(이 환경 공유) 대화 상자에서 Invite Members(멤버 초대)에 다음 중 하나를 입력합니다.
   - IAM 사용자를 초대하려면 사용자의 이름을 입력합니다.
   - AWS 계정 루트 사용자를 초대하려면 `arn:aws:iam::123456789012:root`를 입력합니다. `123456789012`를 사용자의 AWS 계정 ID로 바꿉니다.
   - 위임된 역할을 가진 사용자 또는 위임된 역할을 가진 페더레이션 사용자를 초대하려면 `arn:aws:sts::123456789012:assumed-role/MyAssumedRole/MyAssumedRoleSession`을 입력하세요. `123456789012`를 AWS 계정 ID로 바꾸고 `MyAssumedRole`을 위임된 역할의 이름으로 바꿉니다. `MyAssumedRoleSession`을 위임된 역할의 세션 이름으로 바꿉니다.
5. 이 사용자를 읽기 전용 멤버로 설정하려면 [R]을 선택합니다. 이 사용자를 읽기/쓰기 사용자로 설정하려면 [RW]를 선택합니다.
6. [Invite]를 선택합니다.

> **Note**
>
> 이 사용자를 읽기/쓰기 멤버로 설정하면 AWS 보안 자격 증명이 위험해 질 수 있는 정보가 포함된 대화 상자가 표시됩니다. 다음 정보는 이 문제에 대해 보다 자세한 배경 지식을 제공합니다.
> 환경은 신뢰할 수 있는 사람하고만 공유해야 합니다.
> 읽기/쓰기 멤버는 환경에서 AWS CLI, AWS CloudShell 또는 AWS SDK 코드를 사용하여 대신 에서 조치를 취할 수 있습니다. 게다가 환경 내에서 영구적 AWS 액세스 자격 증명을 저장한 경우 해당 멤버는 이러한 자격 증명을 복사해 환경 외부에서 사용할 수 있습니다.
> 환경에서 영구적 AWS 액세스 자격 증명을 제거하고 대신 임시 AWS 액세스 자격 증명을 사용한다고 해서 이 문제가 완전히 해결되는 것은 아닙니다. 하지만 멤버(임시 자격 증명은 제한된 시간 동안에만 유효하기 때문에)가 임시 자격 증명을 복사해 환경 외부에서 사용할 수 있는 가능성이 줄어듭니다. 그러나 임시 자격 증명을 통해 읽기/쓰기 멤버는 여전히 환경에서 대신 AWS 내 조치를 취할 수 있습니다.

7. 사용자에게 연락하여 이 환경을 열고 사용하기 시작할 수 있다고 알려줍니다.

환경과 동일한 계정의 AWS Cloud9 관리자가 자신 또는 다른 사용자를 초대하도록 하기

> **Note**
>
> AWS 관리형 임시 자격 증명을 사용하는 경우 AWS Cloud9 IDE의 터미널 세션을 사용하여 이 섹션의 일부 또는 모든 명령을 실행할 수 없습니다. AWS 보안 모범 사례를 따르기 위해 AWS 관리형 임시 자격 증명은 일부 명령의 실행을 허용하지 않습니다. 대신에 별도의 AWS Command Line Interface(AWS CLI) 설치에서 이러한 명령을 실행할 수 있습니다.

다음 유형의 사용자는 자신(또는 동일한 AWS 계정의 다른 사용자)을 동일 계정 내 모든 환경에 초대할 수 있습니다.

- AWS 계정 루트 사용자.
- 관리자 사용자.
- IAM 그룹에 속한 사용자, 역할을 맡은 사용자 또는 역할을 맡은 페더레이션 사용자. 그리고 이러한 역할 또는 그룹에는 AWS 관리형 정책 `AWSCloud9Administrator`가 연결되어 있어야 합니다.

초대된 사용자가 이전 유형의 사용자 중 하나가 아니라고 가정해 보겠습니다. 사용자에게 해당 환경 멤버 액세스 역할이 이미 있는지 확인하세요. 지침은 환경 멤버 액세스 역할 정보를 참조하십시오.

사용자를 초대하려면 AWS CLI 또는 AWS CloudShell을 사용하여 AWS Cloud9 `create-environment-membership` 명령을 실행합니다.

```
aws cloud9 create-environment-membership --environment-id
 12a34567b8cd9012345ef67abcd890e1 --user-arn USER_ARN --permissions PERMISSION_LEVEL
```

앞서 언급한 명령에서 `12a34567b8cd9012345ef67abcd890e1`을 환경의 ID로 바꿉니다. `PERMISSION_LEVEL`을 `read-write` 또는 `read-only`로 바꿉니다. `USER_ARN`은 다음 중 하나로 바꿉니다.

- IAM 사용자를 초대하려면 `arn:aws:iam::123456789012:user/MyUser`를 입력합니다. `123456789012`를 AWS 계정 ID로 바꾸고 `MyUser`를 사용자의 이름으로 바꿉니다.
- AWS 계정 루트 사용자를 초대하려면 `arn:aws:iam::123456789012:root`를 입력합니다. `123456789012`를 사용자의 AWS 계정 ID로 바꿉니다.

- 위임된 역할을 가진 사용자 또는 위임된 역할을 가진 페더레이션 사용자를 초대하려면 `arn:aws:sts::123456789012:assumed-role/MyAssumedRole/MyAssumedRoleSession`을 입력하세요. `123456789012`를 사용자의 AWS 계정 ID로 바꿉니다. `MyAssumedRole`을 위임된 역할의 이름으로 바꿉니다. 그리고 `MyAssumedRoleSession`을 위임된 역할의 세션 이름으로 바꿉니다.

예를 들어 계정 ID `123456789012`에 대한 AWS 계정 루트 사용자를 환경(ID: `12a34567b8cd9012345ef67abcd890e1`)에 읽기/쓰기 멤버로 초대하려면 다음 명령을 실행합니다.

```
aws cloud9 create-environment-membership --environment-id
 12a34567b8cd9012345ef67abcd890e1 --user-arn arn:aws:iam::123456789012:root --
permissions read-write
```

> **Note**
>
> AWS CloudShell을 사용하는 경우에는 앞서 언급한 명령에서 접두사 `aws`를 생략합니다.

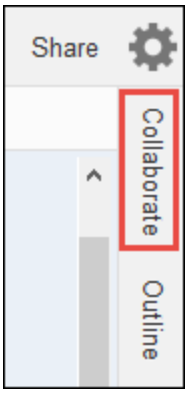
공유 환경 열기

공유 환경을 열려면 AWS Cloud9 대시보드를 사용하면 됩니다. AWS Cloud9 IDE를 사용하여 공유 환경에서 작업을 수행하고 완료할 수 있습니다. 예를 들면 파일 작업 및 다른 팀 구성원과의 채팅이 있습니다.

1. 해당하는 액세스 정책이 사용자에 대한 그룹 또는 역할에 연결되어 있는지 확인합니다. 자세한 내용은 환경 멤버 액세스 역할 정보를 참조하십시오.
2. 다음과 같이 AWS Cloud9 콘솔에 로그인합니다.
   - AWS 계정를 혼자만 사용하는 경우 또는 단일 AWS 계정의 IAM 사용자인 경우 https://console.aws.amazon.com/cloud9/로 이동합니다.
   - 조직에서 IAM Identity Center를 사용하는 경우, 로그인 지침은 AWS 계정 관리자에게 문의하세요.
   - 교실의 학생인 경우 로그인 지침은 강사에게 문의하십시오.
3. AWS Cloud9 대시보드에서 공유 환경을 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기를 참조하세요.

[협업(Collaborate)] 창에서 이 주제의 나머지 부분에서 설명하는 것처럼 다른 멤버와 상호 작용합니다.

> **Note**
>
> Collaborate(협업) 창이 보이지 않으면 Collaborate(협업)를 선택합니다. [협업(Collaborate)] 버튼이 표시되지 않으면 메뉴 모음에서 [창, 협업(Window, Collaborate)]을 선택합니다.

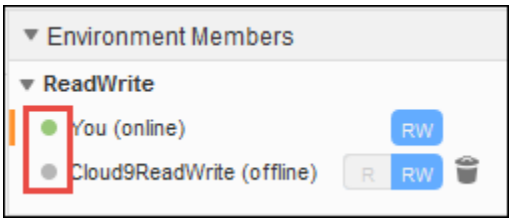


환경 멤버 목록 보기

공유 환경이 열려 있는 상태에서협업창에서 확장멤버 환경, 구성원 목록이 표시되지 않는 경우.

각 멤버 옆에 있는 원은 다음과 같이 멤버의 온라인 상태를 나타냅니다.

- 활성 멤버에는 녹색 원이 표시되어 있습니다.
- 오프라인 멤버에는 회색 원이 표시되어 있습니다.
- 유휴 멤버에는 주황색 원이 표시되어 있습니다.



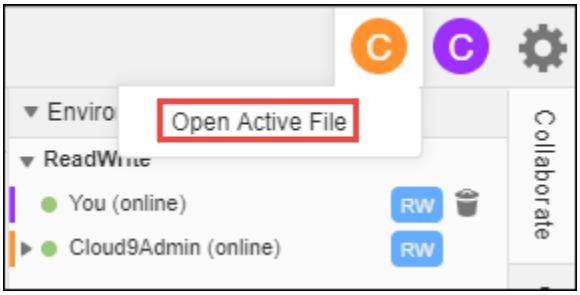
코드를 사용해 환경 멤버의 목록을 가져오려면 다음과 같이 AWS Cloud9 describe environment memberships 작업을 호출합니다.

AWS CLI	describe-environment-memberships

AWS SDK for C++	DescribeEnvironmentMembershipsRequest, DescribeEnvironmentMembershipsResult
AWS SDK for Go	DescribeEnvironmentMemberships, DescribeEnvironmentMembershipsRequest, DescribeEnvironmentMembershipsWithContext
AWS SDK for Java	DescribeEnvironmentMembershipsRequest, DescribeEnvironmentMembershipsResult
AWS SDK for JavaScript	describeEnvironmentMemberships
AWS SDK for .NET	DescribeEnvironmentMembershipsRequest, DescribeEnvironmentMembershipsResponse
AWS SDK for PHP	describeEnvironmentMemberships
AWS SDK for Python (Boto)	describe_environment_memberships
AWS SDK for Ruby	describe_environment_memberships
AWS Tools for Windows PowerShell	Get-C9EnvironmentMembershipList
AWS Cloud9 API	DescribeEnvironmentMemberships

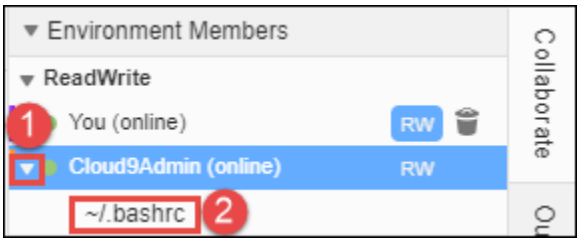
환경 멤버의 활성 파일 열기

공유 환경이 열려 상태에서 메뉴 모음에서 멤버 이름을 선택합니다. Open Active File(활성 파일 열기)을 선택합니다.



환경 멤버의 열어 놓은 파일 열기

1. 멤버 목록이 표시되지 않을 경우 공유 환경이 열려 있는 상태로 [협업(Collaborate)] 창에서 [환경 멤버(Environment Members)]를 확장합니다.
2. 환경에서 열려고 하는 열린 파일을 소유한 사용자의 이름을 확장합니다.
3. 열려는 파일의 이름을 엽니다(두 번 클릭).

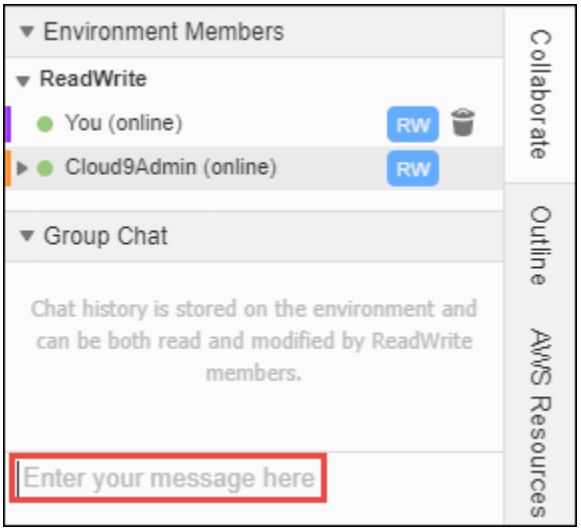


환경 멤버의 활성 커서로 이동

1. 멤버 목록이 표시되지 않을 경우 공유 환경이 열려 있는 상태로 [협업(Collaborate)] 창에서 [환경 멤버(Environment Members)]를 확장합니다.
2. 멤버 이름에 대한 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) Show Location(위치 표시)을 선택합니다.

다른 환경 멤버와 채팅

공유 환경이 열려 있는 상태에서, [협업(Collaborate)] 창 하단에서 [여기에 메시지 입력(Enter your message here)]에 채팅 메시지를 입력하고 `Enter` 키를 누릅니다.



공유 환경에서 채팅 메시지 보기

채팅 메시지 목록이 표시되지 않는 경우, 공유 환경을 연 상태로 Collaborate(협업) 창에서 Group Chat(그룹 채팅)을 확장합니다.

공유 환경에서 채팅 메시지 삭제

공유 환경이 열려 있을 때 Collaborate(협업) 창에서 Group Chat(그룹 채팅)의 채팅 메시지에 대한 컨텍스트 메뉴(마우스 오른쪽 버튼으로 클릭)를 엽니다. 그런 다음, Delete Message(메시지 삭제)를 선택합니다.

> **Note**
>
> 채팅 메시지를 삭제하면 환경에서 모든 멤버에 대해 삭제됩니다.

공유 환경에서 모든 채팅 메시지 삭제

공유 환경이 열려 있는 상태에서, Collaborate(협업) 창에서 Group Chat(그룹 채팅)의 아무 곳이나 컨텍스트 메뉴(마우스 오른쪽 버튼으로 클릭)를 선택합니다. 그런 다음, Clear history(기록 삭제)를 선택합니다.

> **Note**
>
> 채팅 메시지를 모두 삭제하면 모든 멤버에 대한 환경에서 삭제됩니다.

환경 멤버의 액세스 역할 변경

1. 환경을 아직 열지 않은 경우 현재 소유하고 있고 변경하려는 액세스 역할을 가진 멤버가 포함된 환경을 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기를 참조하세요.
2. 멤버 목록이 표시되지 않을 경우 Collaborate(협업) 창에서 Environment Members(환경 멤버)를 확장합니다.
3. 다음 작업 중 하나를 수행합니다.
   - 변경하려는 액세스 역할을 가진 멤버 이름 옆에서 R 또는 RW를 선택하여 이것을 멤버 소유자 또는 읽기/쓰기로 각각 설정합니다.
   - 읽기/쓰기 멤버를 읽기 전용 멤버로 변경하려면 멤버 이름에 대한 컨텍스트 메뉴(마우스 오른쪽 버튼으로 클릭)를 연 다음 Revoke Write Access(쓰기 액세스 취소)를 선택합니다.

- 읽기 전용 멤버를 읽기/쓰기 멤버로 변경하려면 멤버 이름에 대한 컨텍스트 메뉴(마우스 오른쪽 버튼으로 클릭)를 연 다음, Grant Read+Write Access(읽기+쓰기 액세스 부여)를 선택합니다.

> **Note**
>
> 이 사용자를 읽기/쓰기 멤버로 설정하면 AWS 보안 자격 증명이 위험해 질 수 있는 정보가 포함된 대화 상자가 표시됩니다. AWS에서 해당 사용자가 여러분을 대신해 조치를 취할 수 있을 것으로 신뢰하지 않는다면 사용자를 읽기/쓰기 멤버로 지정하지 마세요. 자세한 내용은 동일한 계정의 사용자를 환경으로 초대의 관련 참고 사항을 참조하십시오.

코드를 사용하여 환경 멤버의 액세스 역할을 변경하려면 다음과 같이 AWS Cloud9 update environment membership 작업을 호출합니다.

AWS CLI	update-environment-membership
AWS SDK for C++	UpdateEnvironmentMembershipRequest, UpdateEnvironmentMembershipResult
AWS SDK for Go	UpdateEnvironmentMembership, UpdateEnvironmentMembershipRequest, UpdateEnvironmentMembershipWithContext
AWS SDK for Java	UpdateEnvironmentMembershipRequest, UpdateEnvironmentMembershipResult
AWS SDK for JavaScript	updateEnvironmentMembership
AWS SDK for .NET	UpdateEnvironmentMembershipRequest, UpdateEnvironmentMembershipResponse
AWS SDK for PHP	updateEnvironmentMembership
AWS SDK for Python (Boto)	update_environment_membership
AWS SDK for Ruby	update_environment_membership
AWS Tools for Windows PowerShell	Update-C9EnvironmentMembership

AWS Cloud9 API	UpdateEnvironmentMembership

공유 환경에서 사용자 제거

> **Note**
> 환경 소유자가 아니라면 환경에서 사용자를 제거할 수 없습니다.
> 환경에서 사용자를 제거해도 IAM에서 제거되는 것은 아닙니다.

1. 멤버 목록이 표시되지 않을 경우 공유 환경이 열려 있는 상태로 [협업(Collaborate)] 창에서 [환경 멤버(Environment Members)]를 확장합니다.
2. 다음 작업 중 하나를 수행합니다.
   - You(현재 사용자) 옆에서 휴지통 아이콘을 선택합니다.
   - 여러분을 위한 컨텍스트(마우스 오른쪽 버튼으로 클릭) 메뉴를 선택한 후 Leave environment(환경 나가기)를 선택합니다.
3. 메시지가 나타나면 Leave(나가기)를 선택합니다.

코드를 사용하여 공유 환경에서 사용자를 제거하려면 다음과 같이 AWS Cloud9 delete environment membership 작업을 호출합니다.

AWS CLI	delete-environment-membership
AWS SDK for C++	DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipResult
AWS SDK for Go	DeleteEnvironmentMembership, DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipWithContext
AWS SDK for Java	DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipResult
AWS SDK for JavaScript	deleteEnvironmentMembership

AWS SDK for .NET	DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipResponse
AWS SDK for PHP	deleteEnvironmentMembership
AWS SDK for Python (Boto)	delete_environment_membership
AWS SDK for Ruby	delete_environment_membership
AWS Tools for Windows PowerShell	Remove-C9EnvironmentMembership
AWS Cloud9 API	DeleteEnvironmentMembership

다른 공유 멤버 제거

> **Note**
>
> 환경에서 내 사용자가 아닌 멤버를 제거하려면 환경 소유자의 보안 인증 정보를 사용하여 AWS Cloud9에 로그인해야 합니다.
> 멤버를 제거한다고 해서 IAM에서 해당 사용자가 제거되는 것은 아닙니다.

1. 환경을 아직 열지 않은 경우 제거하려는 멤버가 포함된 환경을 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기를 참조하세요.
2. 멤버 목록이 표시되지 않을 경우 [협업(Collaborate)] 창에서 [환경 멤버(Environment Members)]를 확장합니다.
3. 다음 중 하나를 수행하세요.
   - 삭제할 멤버의 이름 옆에 있는 휴지통 아이콘을 선택합니다.
   - 삭제하려는 멤버의 이름에 대한 컨텍스트 메뉴(마우스 오른쪽 버튼으로 클릭)를 연 다음, Revoke Access(액세스 취소)를 선택합니다.
4. 메시지가 나타나면 Remove Member(멤버 제거)를 선택합니다.

코드를 사용하여 환경에서 멤버를 제거하려면 다음과 같이 AWS Cloud9 delete environment membership 작업을 호출합니다.

AWS CLI	delete-environment-membership
AWS SDK for C++	DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipResult
AWS SDK for Go	DeleteEnvironmentMembership, DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipWithContext
AWS SDK for Java	DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipResult
AWS SDK for JavaScript	deleteEnvironmentMembership
AWS SDK for .NET	DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipResponse
AWS SDK for PHP	deleteEnvironmentMembership
AWS SDK for Python (Boto)	delete_environment_membership
AWS SDK for Ruby	delete_environment_membership
AWS Tools for Windows PowerShell	Remove-C9EnvironmentMembership
AWS Cloud9 API	DeleteEnvironmentMembership

환경 공유 모범 사례

환경 공유 시 다음과 같은 모범 사례를 권장합니다.

- 신뢰할 수 있는 읽기/쓰기 멤버만 환경에 초대합니다.
- EC2 환경의 경우 읽기/쓰기 멤버는 환경 소유자의 AWS 액세스 보안 인증 정보를 사용하여 환경에서 AWS 서비스를 호출할 수 있습니다. 이것은 자신의 보안 인증 정보 대신 사용됩니다. 이를 방지하기 위해 환경 소유자는 환경에 대해 AWS 관리형 임시 자격 증명을 사용 중지할 수 있습니다. 그러나 이렇게 하면 환경 소유자도 호출할 수 없습니다. 자세한 내용은 AWS 관리형 임시 자격 증명을 참조하세요.

- AWS CloudTrail을 켜 환경에서의 활동을 추적합니다. 자세한 정보는 AWS CloudTrail 사용 설명서를 참조하세요.
- 환경을 생성 및 공유할 때 AWS 계정 루트 사용자를 사용하지 마세요. 대신 계정 내 IAM 사용자를 사용합니다. 자세한 내용은 IAM 사용 설명서에서 첫 액세스에만 해당: 루트 사용자 보안 인증 정보 및 IAM 사용자를 참조하세요.

환경 이동 및 Amazon EBS 볼륨 크기 조정 또는 암호화

Amazon EC2 인스턴스 간에 AWS Cloud9 개발 환경을 이동할 수 있습니다. 예를 들어, 다음 작업을 수행할 수 있습니다.

- 정상적인 인스턴스와 비교하여 장애가 발생하거나 예기치 않은 방식으로 작동하는 Amazon EC2 인스턴스에서 환경을 이전합니다.
- 기존 인스턴스에서 최신 시스템 업데이트가 설치된 인스턴스로 환경을 이전합니다.
- 현재 인스턴스에서 환경이 과도하게 사용되거나 잘 사용되지 않으므로 인스턴스의 컴퓨팅 리소스를 늘리거나 줄입니다.

프로젝트 파일은 유지한 채 새 AWS Cloud9 EC2 환경으로 마이그레이션하여 AWS Cloud9 지원되는 AMI를 다른 AMI로 업그레이드할 수 있습니다. 다음과 같은 이유로 AMI의 다른 버전으로 업그레이드해야 할 수 있습니다.

- 현재 환경의 AMI가 end-of-life 도달했으며 더 이상 지원되지 않습니다.
- 필요한 패키지는 현재 AMI에서 구형입니다.

또한 환경의 Amazon EC2 인스턴스와 연결되어 있는 Amazon Elastic Block Store(Amazon EBS) 볼륨의 크기를 조정할 수도 있습니다. 예를 들어, 다음 작업 중 하나 또는 모두를 수행할 수 있습니다.

- 해당 인스턴스에 스토리지 공간이 부족하므로 볼륨의 크기를 늘립니다.
- 사용하고 있지 않은 추가 스토리지 공간에 대해 요금을 지불하기를 원치 않으므로 볼륨의 크기를 줄입니다.

환경을 이동하거나 크기를 조정하려면 먼저 환경에서 실행 중인 프로세스를 중지하거나 스왑 파일을 환경에 추가해 볼 수 있습니다. 메모리 부족 또는 높은 CPU 사용률 문제를 해결하는 방법에 관한 자세한 내용은 문제 해결을 참조하세요.

> **Note**
>
> 이 주제에서는 Amazon EC2 인스턴스에서 다른 인스턴스로 환경을 이동하거나 Amazon EBS 볼륨의 크기를 조정하는 작업에 대해서만 설명합니다. 고유 서버 중 하나에서 환경의 크기를 조정하거나 고유 서버 중 하나에 대한 스토리지 공간을 변경하려면 서버 설명서를 참조하세요.

마지막으로 Amazon EBS 리소스를 암호화하여 data-at-rest data-in-transit 인스턴스와 연결된 EBS 스토리지 모두의 보안을 보장할 수 있습니다.

주제

환경 이동

이동 프로세스를 시작하기 전에 다음 조건을 유의하세요.

- 환경을 동일한 유형의 Amazon EC2 인스턴스로 이동할 수 없습니다. 이동 시 새 인스턴스에 대해 다른 Amazon EC2 인스턴스를 선택해야 합니다.

  > **Important**
  >
  > 환경을 다른 Amazon EC2 인스턴스 유형으로 이동하는 경우 현재 Amazon EC2 인스턴스 유형도 지원되어야 합니다. AWS Cloud9 AWS 리전각 리전에서 사용할 수 있는 인스턴스 유형을 확인하려면 콘솔을 사용하여 EC2 환경을 생성할 때 표시되는 Configure settings(설정 구성) 페이지로 이동하세요. 인스턴스 유형 섹션의 선택은 콘솔 오른쪽 상단에서 선택한 유형에 따라 결정됩니다. AWS 리전

- 인스턴스 유형을 변경하려면 먼저 환경과 연결된 Amazon EC2 인스턴스를 중지해야 합니다. 인스턴스가 중지된 상태에서는 본인을 비롯한 어떤 멤버도 중지된 인스턴스와 연결된 환경을 사용할 수 없습니다.
- AWS 인스턴스를 새 하드웨어로 이동하지만 인스턴스의 ID는 변경되지 않습니다.

- 인스턴스가 Amazon VPC에서 실행 중이고 퍼블릭 IPv4 주소가 있는 경우, 주소를 AWS 해제하고 인스턴스에 새 퍼블릭 IPv4 주소를 제공합니다. 인스턴스는 프라이빗 IPv4 주소와 모든 탄력적 IP 주소 또는 IPv6 주소를 유지합니다.
- 인스턴스가 중단된 동안의 가동 중지 시간을 계획합니다. 이 프로세스는 완료하는 데 몇 분이 걸릴 수 있습니다.

환경을 이동하려면

1. (선택 사항) 기존 인스턴스에 설치되지 않은 드라이버가 새로운 인스턴스 유형에 필요한 경우, 인스턴스에 연결하여 해당 드라이버를 설치합니다. 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 크기 조정 호환성을 참조하십시오.
2. 환경이 현재 표시된 모든 웹 브라우저 탭을 닫습니다.

   > **Important**
   >
   > 현재 환경을 표시하고 있는 웹 브라우저 탭을 모두 닫지 않으면 이 절차를 완료하는 데 방해가 AWS Cloud9 될 수 있습니다. 특히, 이 절차 중에 잘못된 시간에 환경과 연결된 Amazon EC2 인스턴스를 다시 시작하려고 할 AWS Cloud9 수 있습니다. 이 절차의 마지막 단계에 도달할 때까지 인스턴스를 중지된 상태로 유지해야 합니다.

3. 아직 로그인하지 않은 AWS Management Console경우 https://console.aws.amazon.com 에서 로그인하십시오.

   에서 관리자 수준 자격 증명을 사용하여 로그인하는 것이 좋습니다. AWS 계정이렇게 할 수 없는 경우 관리자에게 문의하세요. AWS 계정
4. Amazon EC2 콘솔을 엽니다. 이렇게 하려면 서비스 목록에서 EC2를 선택합니다.
5. AWS 탐색 표시줄에서 이동하려는 환경이 AWS 리전 포함된 환경 (예: 미국 동부 (오하이오)) 을 선택합니다.
6. 서비스 탐색 모음에서 [인스턴스(Instances)]를 확장하고 나서 [인스턴스(Instances)]를 선택합니다.
7. 인스턴스 목록에서 이동할 환경과 연결된 인스턴스를 선택합니다. EC2 환경의 경우 인스턴스 이름은 `aws-cloud9-`으로 시작하고 뒤에 환경 이름이 붙습니다. 예를 들어 환경 이름이 `my-demo-environment`인 경우, 인스턴스 이름은 `aws-cloud9-my-demo-environment`로 시작합니다.
8. 인스턴스 상태가 중지되지 않은 경우 [작업], [인스턴스 상태], [중지] 를 선택합니다. 메시지가 표시되면 Yes, Stop(예, 중지)을 선택합니다. 인스턴스가 중지하는 데 몇 분 정도 걸릴 수 있습니다.

9. [인스턴스 상태(Instance State)]가 [중지됨(stopped)]으로 설정되고 나면 해당 인스턴스가 여전히 선택된 상태에서 [작업(Actions)], [인스턴스 설정(Instance Settings)], [인스턴스 유형 변경(Change Instance Type)]을 선택합니다.
10. [인스턴스 유형 변경(Change Instance Type)] 대화 상자에서 환경에 사용할 새 [인스턴스 유형(Instance Type)]을 선택합니다.

> **Note**
>
> 원하는 인스턴스 유형이 목록에 없으면 해당 인스턴스의 구성과 호환되지 않는 것입니다. 예를 들어 가상화 유형 때문에 인스턴스가 호환되지 않을 수 있습니다.

11. (선택 영역) 선택한 인스턴스 유형이 EBS 최적화를 지원하는 경우 EBS-optimized(EBS 최적화)를 선택하여 EBS 최적화를 활성화하거나, EBS-optimized(EBS 최적화)의 선택을 취소하여 EBS 최적화를 비활성화합니다.

> **Note**
>
> 선택한 인스턴스 유형이 기본적으로 EBS에 최적화된 경우 [EBS 최적화(EBS-optimized)]가 선택되고 이 선택을 취소할 수 없습니다.

12. 적용을 선택하여 새로운 설정을 승인합니다.

> **Note**
>
> 이 절차의 앞부분에서 [인스턴스 유형(Instance Type)]으로 다른 인스턴스 유형을 선택하지 않은 경우 [적용(Apply)]을 선택해도 아무런 동작이 발생하지 않습니다.

13. 환경을 다시 엽니다. 자세한 정보는 AWS Cloud9에서 환경 열기을 참조하세요.

위 절차에 대한 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 유형 변경을 참조하십시오.

AWS Cloud9 EC2 환경을 다른 Amazon 머신 이미지 (AMI) 로 이동

이 주제에서는 한 Amazon Linux AMI에서 AWS Cloud9 지원되는 다른 AMI로 AWS Cloud9 EC2 환경을 마이그레이션하는 방법을 설명합니다.

> **Note**
>
> OS 버전을 업데이트하지 않고 환경을 새 인스턴스로 이동하려는 경우 을 참조하십시오the section called “환경 이동”.

다음 절차 중 하나를 사용하여 환경 간에 데이터를 마이그레이션할 수 있습니다.

아카이브를 로컬 시스템에 다운로드하여 환경을 이동하려면

1. 동일한 가용 영역에 다른 기본 이미지를 사용하여 새 환경을 생성하십시오.

   a. the section called “EC2 환경 생성”섹션의 단계를 완료하여 새 환경을 생성하십시오.

   > **Note**
   >
   > 플랫폼을 선택할 때 환경을 마이그레이션하려는 플랫폼을 선택하십시오.

   b. 기본적으로 환경은 10GiB 볼륨으로 생성됩니다. 새 환경에 아카이브를 업로드하거나 압축을 풀 공간이 충분하지 않은 경우 the section called “환경에서 사용하는 Amazon EBS 볼륨 크기 조정” 절차의 단계를 완료하여 Amazon EBS 볼륨 크기를 조정하십시오.
2. 마이그레이션하려는 환경을 IDE에서 엽니다. AWS Cloud9
3. AWS Cloud9 IDE가 로드되면 메뉴에서 파일 > 프로젝트 다운로드를 선택하여 환경 프로젝트 디렉토리의 내용이 포함된 아카이브를 다운로드합니다.
4. 새 환경에서 AWS Cloud9 IDE를 엽니다.
5. 파일 > 로컬 파일 업로드... 를 선택합니다. 아카이브를 업로드하려면
6. (선택 사항) 환경 터미널에서 이전 `.c9` 디렉터리를 백업하려면 다음 명령을 실행합니다. `.c9.backup`

   ```
   cp .c9 .c9.backup
   ```

   나중에 구성 파일을 복원하려는 경우 이러한 백업 파일이 필요할 수 있습니다.
7. 아카이브의 압축을 풀려면 다음 명령을 실행합니다.

   ```
   tar xzvf <old_environment_name>.tar.gz -C ~/
   ```
8. 프로젝트 디렉터리에서 아카이브를 삭제하려면 다음 명령을 실행합니다.

```
rm <old_environment_name>.tar.gz
```

새 환경이 예상대로 작동하는지 확인하십시오.

9. 이제 이전 환경을 삭제할 수 있습니다.

Amazon EBS 볼륨을 사용하여 환경을 이동하려면

아카이브를 다운로드할 수 없거나 결과 아카이브가 너무 큰 경우 Amazon EBS 볼륨을 사용하여 마이그레이션할 수 있습니다. 또한 이 방법을 사용하면 `~/environment` 디렉터리 외부에 있는 파일을 복사할 수 있습니다.

1. 기존 환경에 열려 있는 모든 AWS Cloud9 IDE 탭을 닫습니다.
2. 기존 인스턴스를 중지하려면 다음 단계를 완료하십시오.
   a. AWS Cloud9 콘솔에서 탐색할 환경을 선택하여 세부 정보를 확인합니다.
   b. 환경 세부 정보 페이지의 EC2 인스턴스 탭에서 EC2 인스턴스 관리를 선택합니다.
   c. EC2 콘솔에서 인스턴스를 선택하여 인스턴스 세부 정보로 이동합니다.
   d. 인스턴스 상태가 중지됨으로 설정되어 있는지 확인하십시오. 그렇지 않은 경우 인스턴스 상태 드롭다운 목록에서 인스턴스 중지를 선택합니다. 메시지가 표시되면 [Stop] 을 선택합니다. 인스턴스가 중지하는 데 몇 분 정도 걸릴 수 있습니다.
3. 동일한 가용 영역에 다른 기본 이미지를 사용하여 새 환경을 생성하십시오.
   a. the section called “EC2 환경 생성”섹션의 단계를 완료하여 새 환경을 생성하십시오.

      > **Note**
      > 플랫폼을 선택할 때 환경을 마이그레이션하려는 플랫폼을 선택하십시오.

   b. 기본적으로 환경은 10GiB 볼륨으로 생성됩니다. 소스 볼륨에서 새 환경으로 파일을 이동할 공간이 충분하지 않은 경우 the section called “환경에서 사용하는 Amazon EBS 볼륨 크기 조정” 절차의 단계를 완료하여 Amazon EBS 볼륨 크기를 조정하십시오.
4. 다음 단계를 완료하여 기존 인스턴스에서 볼륨을 분리하십시오.
   a. 인스턴스 요약 페이지에서 스토리지 탭을 선택하고 볼륨을 선택합니다. 선택한 볼륨의 디바이스 이름은 루트 디바이스 세부 정보 섹션의 루트 디바이스 이름에 지정된 이름과 같아야 합니다.

b. 볼륨 세부 정보 페이지에서 작업 > 볼륨 분리를 선택합니다.

c. 볼륨을 성공적으로 분리한 후 [작업] > [볼륨 연결] 을 선택한 다음 드롭다운 목록에서 새 환경의 인스턴스를 찾아 선택합니다. 선택하는 Amazon EC2 인스턴스의 이름에는 접두사가 붙은 AWS Cloud9 환경 이름이 포함되어야 합니다. `aws-cloud9`

5. 새 환경에서 AWS Cloud9 IDE를 엽니다.

6. 환경이 로드된 후 새로 연결된 볼륨의 장치를 식별하려면 터미널에서 다음 명령을 실행합니다.

```
lsblk
```

다음 샘플 출력에서는 루트 `nvme0n1` 디바이스의 `nvme0n1p1` 파티션이 이미 마운트되어 있으므로 `nvme1n1p1` 파티션도 마운트해야 합니다. 해당 디바이스의 전체 경로는 `/dev/nvme1n1p1` 다음과 같습니다.

```
Admin:~/environment $ lsblk
NAME          MAJ:MIN RM SIZE RO TYPE MOUNTPOINTS
nvme0n1       259:0    0  10G  0 disk
##nvme0n1p1   259:2    0  10G  0 part /
##nvme0n1p127 259:3    0   1M  0 part
##nvme0n1p128 259:4    0  10M  0 part /boot/efi
nvme1n1       259:1    0  10G  0 disk
##nvme1n1p1   259:5    0  10G  0 part
##nvme1n1p128 259:6    0   1M  0 part
```

> **Note**
>
> 터미널에서 이 명령을 실행하면 출력이 달라집니다.

7. 환경 터미널에서 다음 단계를 완료하여 기존 볼륨을 마운트하십시오.

a. 볼륨의 파티션을 마운트할 임시 디렉터리를 만들려면 다음 명령을 실행합니다.

```
MOUNT_POINT=$(mktemp -d)
```

b. `lsblk`명령의 샘플 출력에 따라 마운트할 디바이스의 경로를 다음과 같이 지정합니다.

```
MOUNT_DEVICE=/dev/nvme1n1p1
```

> **Note**
>
> 터미널에서 이 명령을 실행하면 출력이 달라집니다.

c. 기존 볼륨을 마운트하려면 다음 명령을 실행합니다.

```
sudo mount $MOUNT_DEVICE $MOUNT_POINT
```

d. 다음 단계를 완료하여 기존 볼륨이 올바르게 마운트되었는지 확인하십시오.

i. 볼륨이 출력에 포함되는지 확인하려면 다음 명령을 실행합니다.

```
df -h
```

ii. 볼륨의 내용을 확인하려면 다음 명령을 실행합니다.

```
ls $MOUNT_POINT/home/ec2-user/environment/
```

8. (선택 사항) 환경 터미널에서 이전 `.c9` 디렉터리를 백업하려면 다음 명령을 실행합니다. `.c9.backup`

```
cp .c9 .c9.backup
```

나중에 구성 파일을 복원하려는 경우 이러한 백업 파일이 필요할 수 있습니다.

9. 기존 볼륨에서 이전 환경을 복사하려면 다음 명령을 실행합니다.

```
cp -R $MOUNT_POINT/home/ec2-user/environment ~
```

> **Note**
>
> 필요한 경우 이전 명령을 사용하여 환경 디렉터리 외부의 파일이나 디렉터리를 복사할 수도 있습니다.

새 환경이 예상대로 작동하는지 확인하십시오.

10. 이전 디바이스를 마운트 해제하려면 다음 두 명령 중 하나를 실행합니다.

```
sudo umount $MOUNT_DEVICE
```

```
sudo umount $MOUNT_POINT
```

11. 작업 드롭다운 목록에서 볼륨 분리를 선택하여 3단계에서 연결한 볼륨을 분리합니다.
12. 이제 이전 환경과 해당 볼륨을 삭제할 수 있습니다.

> **Note**
>
> 볼륨이 더 이상 환경의 Amazon EC2 인스턴스에 연결되어 있지 않으므로 수동으로 제거해야 합니다. 볼륨 세부 정보 페이지에서 삭제를 선택하여 이 작업을 수행할 수 있습니다.

환경에서 사용하는 Amazon EBS 볼륨 크기 조정

1. 크기를 조정할 Amazon EBS 볼륨에 대해 Amazon EC2 인스턴스와 연결된 환경을 엽니다.
2. 환경용 AWS Cloud9 IDE에서 다음 내용으로 파일을 만든 다음 확장명을 사용하여 파일을 저장합니다 `.sh` (예:`resize.sh`).

> **참고**
>
> 이 스크립트는 AL2023, Amazon Linux 2, Amazon Linux 또는 Ubuntu 서버를 실행하는 EC2 인스턴스에 연결된 Amazon EBS 볼륨에 적용되며 IMDSv2를 사용하도록 구성됩니다.
> 또한 이 스크립트는 Nitro 기반 인스턴스에 NVMe 블록 디바이스로 표시된 Amazon EBS 볼륨의 크기를 조정합니다. Nitro 시스템 기반 인스턴스 목록은 Amazon EC2 사용 설명서의 Nitro기반 인스턴스를 참조하십시오.

```
#!/bin/bash

# Specify the desired volume size in GiB as a command line argument. If not
 specified, default to 20 GiB.
SIZE=${1:-20}

# Get the ID of the environment host Amazon EC2 instance.
```

```
TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-
metadata-token-ttl-seconds: 60")
INSTANCEID=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" -v
 http://169.254.169.254/latest/meta-data/instance-id 2> /dev/null)
REGION=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/
latest/meta-data/placement/region 2> /dev/null)

# Get the ID of the Amazon EBS volume associated with the instance.
VOLUMEID=$(aws ec2 describe-instances \
  --instance-id $INSTANCEID \
  --query "Reservations[0].Instances[0].BlockDeviceMappings[0].Ebs.VolumeId" \
  --output text \
  --region $REGION)

# Resize the EBS volume.
aws ec2 modify-volume --volume-id $VOLUMEID --size $SIZE

# Wait for the resize to finish.
while [ \
  "$(aws ec2 describe-volumes-modifications \
    --volume-id $VOLUMEID \
    --filters Name=modification-state,Values="optimizing","completed" \
    --query "length(VolumesModifications)"\
    --output text)" != "1" ]; do
sleep 1
done

# Check if we're on an NVMe filesystem
if [[ -e "/dev/xvda" && $(readlink -f /dev/xvda) = "/dev/xvda" ]]
then
# Rewrite the partition table so that the partition takes up all the space that it
 can.
  sudo growpart /dev/xvda 1
# Expand the size of the file system.
# Check if we're on AL2 or AL2023
  STR=$(cat /etc/os-release)
  SUBAL2="VERSION_ID=\"2\""
  SUBAL2023="VERSION_ID=\"2023\""
  if [[ "$STR" == *"$SUBAL2"* || "$STR" == *"$SUBAL2023"* ]]
  then
    sudo xfs_growfs -d /
  else
    sudo resize2fs /dev/xvda1
  fi
```

```
else
# Rewrite the partition table so that the partition takes up all the space that it
 can.
  sudo growpart /dev/nvme0n1 1

# Expand the size of the file system.
# Check if we're on AL2 or AL2023
  STR=$(cat /etc/os-release)
  SUBAL2="VERSION_ID=\"2\""
  SUBAL2023="VERSION_ID=\"2023\""
  if [[ "$STR" == *"$SUBAL2"* || "$STR" == *"$SUBAL2023"* ]]
  then
    sudo xfs_growfs -d /
  else
    sudo resize2fs /dev/nvme0n1p1
  fi
fi
```

3. IDE의 터미널 세션에서 `resize.sh` 파일을 포함하는 디렉터리로 전환합니다. 그리고 나서 다음 명령 중 하나를 실행하여 Amazon EBS Volume의 크기를 조정하기 위해 `20`을 원하는 크기(단위: GIB)로 바꿉니다.

   - ```
     bash resize.sh 20
     ```
   - ```
     chmod +x resize.sh
     ./resize.sh 20
     ```

사용하는 Amazon EBS 볼륨을 암호화합니다. AWS Cloud9

Amazon EBS 암호화는 다음 데이터를 암호화합니다.

- 볼륨의 저장된 데이터
- 볼륨과 인스턴스 사이에서 이동하는 모든 데이터
- 볼륨에서 생성된 모든 스냅샷
- 그런 스냅샷에서 생성된 모든 볼륨

AWS Cloud9 EC2 개발 환경에 사용되는 Amazon EBS 볼륨에는 두 가지 암호화 옵션이 있습니다.

- 기본적으로 암호화 – 사용자의 AWS 계정 를 구성하여 생성하는 새 EBS 볼륨 및 스냅샷 사본의 암호화를 적용할 수 있습니다. 기본적으로 암호화는 AWS 리전의 수준에서 활성화됩니다. 따라서 해당 리전의 개별 볼륨 또는 스냅샷에 대해 이 기능을 활성화할 수 없습니다. 또한 Amazon EBS는 인스턴스를 시작할 때 생성된 볼륨을 암호화합니다. 따라서 EC2 환경을 생성하기 전에 이 설정을 활성화해야 합니다. 자세한 내용은 Amazon EC2 사용 설명서의 기본 암호화를 참조하십시오.
- EC2 환경에서 사용하는 기존 Amazon EBS 볼륨의 암호화 - EC2 인스턴스용으로 이미 생성된 특정 Amazon EBS 볼륨을 암호화할 수 있습니다. 이 옵션에는 AWS Key Management Service (AWS KMS) 를 사용하여 암호화된 볼륨에 대한 액세스를 관리하는 작업이 포함됩니다. 관련 절차는 AWS Cloud9 가 사용하는 기존 Amazon EBS 볼륨을 암호화 섹션을 참조하세요.

> **Important**
>
> AWS Cloud9 IDE에서 기본적으로 암호화된 Amazon EBS 볼륨을 사용하는 경우 AWS Identity and Access Management 서비스 연결 역할에 대한 EBS 볼륨에 대한 액세스 권한이 AWS Cloud9 필요합니다. AWS KMS key 액세스가 제공되지 않으면 AWS Cloud9 IDE가 시작되지 않고 디버깅이 어려울 수 있습니다.
> 액세스를 제공하려면 Amazon EBS 볼륨에서 사용하는 KMS 키에 AWS Cloud9`AWSServiceRoleForAWSCloud9`, 의 서비스 연결 역할을 추가하십시오. 이 작업에 대한 자세한 내용은 AWS 규범적 지침 패턴의 기본 암호화와 함께 Amazon EBS 볼륨을 사용하는 AWS Cloud9 IDE 생성을 참조하십시오.

AWS Cloud9 가 사용하는 기존 Amazon EBS 볼륨을 암호화

기존 Amazon EBS 볼륨을 AWS KMS 암호화하려면 를 사용하여 KMS 키를 생성해야 합니다. 교체할 볼륨의 스냅샷을 생성한 후 KMS 키를 사용하여 스냅샷의 사본을 암호화합니다.

다음으로, 해당 스냅샷으로 암호화된 볼륨을 생성합니다. 그런 다음, 암호화되지 않은 볼륨을 EC2 인스턴스에서 분리하고 암호화된 볼륨을 연결하여 교체합니다.

마지막으로, 고객 관리형 키의 키 정책을 업데이트하여 AWS Cloud9 서비스 역할에 대한 액세스를 사용하도록 설정해야 합니다.

> **Note**
>
> 다음 절차에서는 고객 관리 키를 사용하여 볼륨을 암호화하는 데 중점을 둡니다. AWS 서비스 계정에서 AWS 관리형 키 for an을 사용할 수도 있습니다. Amazon EBS의 별칭은 `aws/ebs`입

니다. 암호화에 대해 이 기본 옵션을 선택한 경우 고객 관리형 키를 만드는 1단계를 건너뜁니다. 또한 키 정책을 업데이트하는 8단계를 건너뜁니다. 의 키 정책을 변경할 수 없기 때문입니다 AWS 관리형 키.

기존 Amazon EBS 볼륨을 암호화하려면

1. AWS KMS 콘솔에서 대칭 KMS 키를 생성합니다. 자세한 내용은 AWS Key Management Service 개발자 가이드에서 대칭 KMS 키 생성을 참조하세요.
2. Amazon EC2 콘솔에서 환경에 사용되는 Amazon EBS 지원 인스턴스를 중지합니다. 콘솔이나 명령줄을 사용하여 인스턴스를 중지할 수 있습니다.
3. Amazon EC2 콘솔의 탐색 창에서 Snapshots(스냅샷)를 선택하여 암호화하려는 기존 볼륨의 스냅샷을 생성합니다.
4. Amazon EC2 콘솔의 탐색 창에서 [스냅샷(Snapshots)]을 선택하여 스냅샷을 복사합니다. [스냅샷 복사(Copy snapshot)] 대화 상자에서 다음을 수행하여 암호화를 사용하도록 설정합니다.
   - [이 스냅샷 암호화(Encrypt this snapshot)]를 선택합니다.
   - Master Key(마스터 키)에서 앞서 생성한 KMS 키를 선택합니다. (를 사용하는 경우 (기본) aws/ebs 설정을 유지하십시오.) AWS 관리형 키
5. 암호화된 스냅샷에서 새 볼륨을 생성합니다.

   > **Note**
   >
   > 암호화된 스냅샷에서 생성된 새 Amazon EBS 볼륨은 자동으로 암호화됩니다.

6. Amazon EC2 인스턴스에서 이전 Amazon EBS 볼륨을 분리합니다.
7. 암호화된 새 데이터 볼륨을 Amazon EC2 인스턴스에 연결합니다.
8. AWS Management Console 기본 보기, 정책 보기 또는 API를 사용하여 KMS 키의 키 AWS Management Console 정책을 업데이트하십시오. AWS KMS 다음 주요 정책 설명을 추가하여 AWS Cloud9 서비스`AWSServiceRoleForAWSCloud9`, 가 KMS 키에 액세스할 수 있도록 허용하십시오.

   > **Note**
   >
   > 를 AWS 관리형 키사용하는 경우 이 단계를 건너뛰세요.

```
{
    "Sid": "Allow use of the key",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:{Partition}:iam::{AccountId}:role/aws-service-role/
cloud9.amazonaws.com/AWSServiceRoleForAWSCloud9"
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*"
   },
   {
    "Sid": "Allow attachment of persistent resources",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:{Partition}:iam::{AccountId}:role/aws-service-role/
cloud9.amazonaws.com/AWSServiceRoleForAWSCloud9"
    },
    "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
    ],
    "Resource": "*",
    "Condition": {
        "Bool": {
            "kms:GrantIsForAWSResource": "true"
        }
    }
}
```

9. Amazon EC2 인스턴스를 다시 시작합니다. Amazon EC2 인스턴스 재시작에 대한 자세한 내용은 인스턴스 중지 및 시작을 참조하십시오.

AWS Cloud9에서 환경 삭제

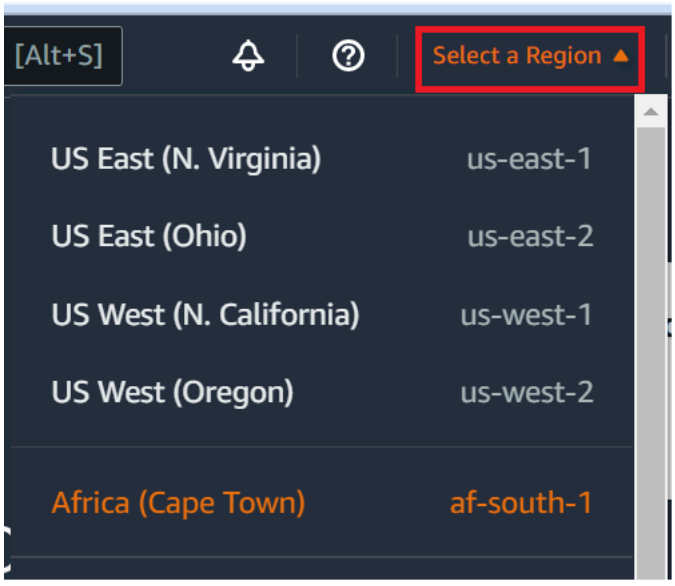
더 이상 사용하지 않는 AWS Cloud9 개발 환경과 관련된 AWS 계정에 계속해서 요금이 부과되지 않도록 하려면 환경을 삭제하세요.

- 콘솔을 사용하여 환경 삭제
- 코드를 사용하여 환경 삭제

콘솔을 사용하여 환경 삭제

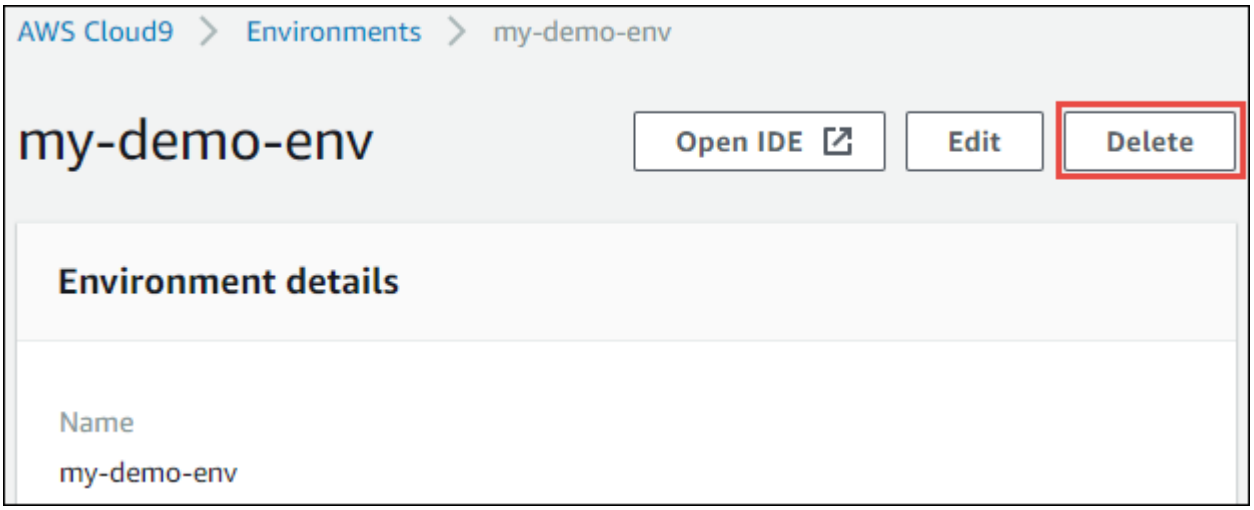
> **Warning**
>
> 환경을 삭제하면 AWS Cloud9이 환경을 영구적으로 삭제합니다. 관련된 모든 설정, 사용자 데이터, 커밋하지 않은 코드도 영구적으로 삭제됩니다. 삭제된 환경은 다시 복구할 수 없습니다.

1. AWS Cloud9 콘솔에 로그인합니다.
   - AWS 계정를 혼자만 사용하는 경우 또는 단일 AWS 계정의 IAM 사용자인 경우 https://console.aws.amazon.com/cloud9/로 이동합니다.
   - 조직에서 AWS IAM Identity Center을 사용하는 경우 로그인 지침은 AWS 계정 관리자에게 문의하세요.
2. 상단 탐색 모음에서 환경이 위치한 AWS 리전을 선택합니다.

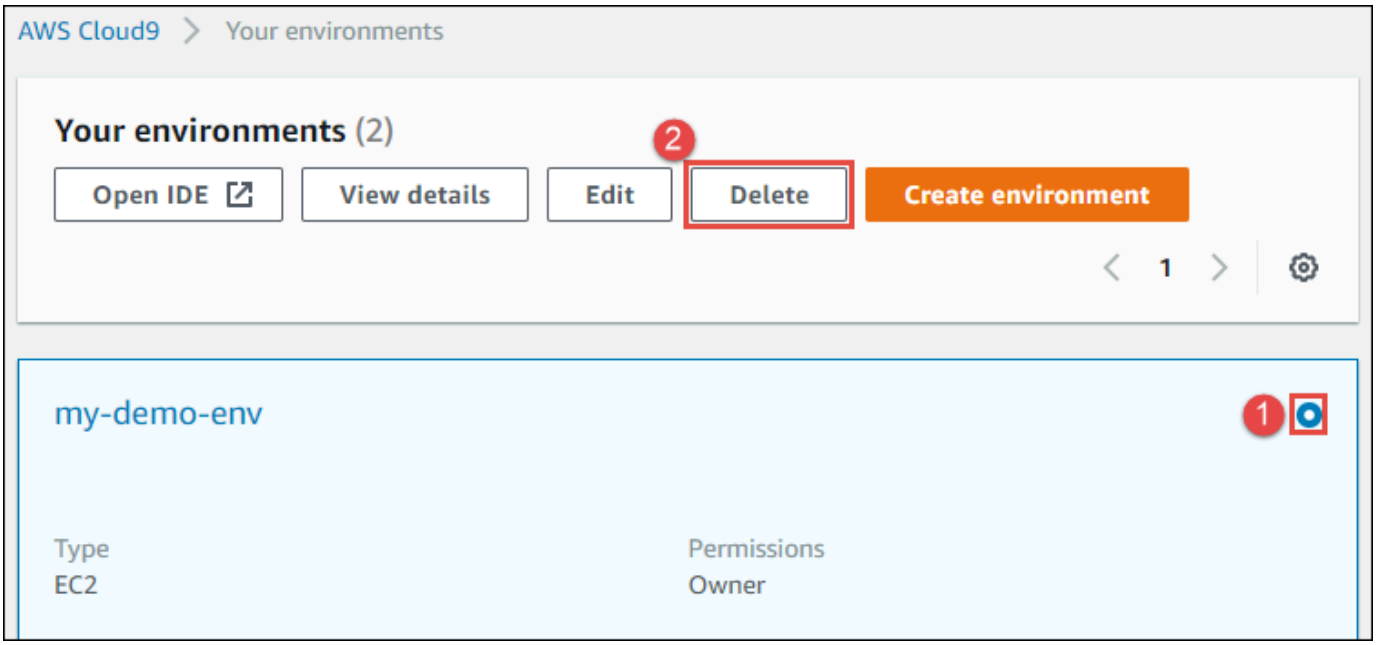


3. 환경 목록에서 삭제하려는 환경에 대해 다음 작업 중 하나를 수행합니다.

- 환경에 대한 카드의 제목을 선택합니다. 그런 후 다음 페이지에서 Delete(삭제)를 선택합니다.



- 환경에 대한 카드를 선택한 다음 [삭제(Delete)] 버튼을 선택합니다.



4. 삭제 대화 상자에 `Delete`를 입력한 후 삭제를 선택합니다.

- EC2 환경

  AWS Cloud9이 해당 환경에 연결된 Amazon EC2 인스턴스도 종료합니다.

> **Note**
>
> 계정 삭제에 실패하면 콘솔 웹 페이지 상단에 배너가 표시됩니다. 또한 환경에 대한 카드(있는 경우)는 환경 삭제에 실패했음을 나타냅니다.

- SSH 환경

  환경이 Amazon EC2 인스턴스에 연결된 경우 AWS Cloud9은 해당 인스턴스를 종료하지 않습니다. 나중에 해당 인스턴스를 종료하지 않으면, AWS 계정에 해당 인스턴스와 관련된 Amazon EC2에 대한 요금이 계속해서 부과될 수 있습니다.

5. 환경이 SSH 환경인 경우, AWS Cloud9은 해당 환경에 연결되었던 자체 서버 또는 클라우드 컴퓨팅 인스턴스에 숨겨진 하위 디렉터리를 남겨 둡니다. 이제 필요하다면 해당 하위 디렉터리를 안전하게 삭제할 수 있습니다. 이 하위 디렉터리의 이름은 `.c9`입니다. 환경을 생성할 때 하위 디렉터리는 지정한 환경 경로 디렉터리에 위치해 있습니다.

   콘솔에 환경이 표시되지 않으면 다음 작업 중 하나 이상을 수행하여 표시해 보세요.

   - Environments(환경) 페이지의 드롭다운 메뉴 표시줄에서 다음 중 하나 이상을 선택합니다.
     - My environments(내 환경)를 선택하여 선택한 AWS 리전 및 AWS 계정 내에서 AWS 엔터티가 소유하고 있는 모든 환경을 표시합니다.
     - Shared with you(사용자와 공유)를 선택하여 선택한 AWS 리전 및 AWS 계정 내에서 AWS 엔터티가 초대된 모든 환경을 표시합니다.
     - All account environments(모든 계정 환경)를 선택하여 AWS 엔터티가 표시할 권한을 가지고 있는 선택한 AWS 리전 및 AWS 계정 내의 모든 환경을 표시합니다.
   - 본인이 환경의 멤버라고 생각하지만 Shared with you(사용자와 공유) 목록에 환경이 표시되지 않는 경우 환경 소유자에 문의하여 확인하세요.
   - 상단 탐색 모음에서 다른 AWS 리전을 선택합니다.

코드를 사용하여 환경 삭제

> **Warning**
>
> 환경을 삭제하면 AWS Cloud9이 환경을 영구적으로 삭제합니다. 관련된 모든 설정, 사용자 데이터, 커밋하지 않은 코드도 영구적으로 삭제됩니다. 삭제된 환경은 다시 복구할 수 없습니다.

코드를 사용하여 AWS Cloud9에서 환경을 삭제하려면 다음과 같이 AWS Cloud9 delete environment 작업을 호출합니다.

AWS CLI	delete-environment

AWS SDK for C++	DeleteEnvironmentRequest, DeleteEnvironmentResult
AWS SDK for Go	DeleteEnvironment, DeleteEnvironmentRequest, DeleteEnvironmentWithContext
AWS SDK for Java	DeleteEnvironmentRequest, DeleteEnvironmentResult
AWS SDK for JavaScript	deleteEnvironment
AWS SDK for .NET	DeleteEnvironmentRequest, DeleteEnvironmentResponse
AWS SDK for PHP	deleteEnvironment
AWS SDK for Python (Boto)	delete_environment
AWS SDK for Ruby	delete_environment
AWS Tools for Windows PowerShell	Remove-C9Environment
AWS Cloud9 API	DeleteEnvironment

AWS Cloud9 통합 개발 환경(IDE) 작업

통합 개발 환경(IDE)은 소스 코드 편집기, 디버거, 빌드 도구 등의 코딩 생산성 도구 세트를 제공합니다.

Important

다음은 AWS Cloud9 사용에 대한 권장 모범 사례입니다.

- 소스 제어를 사용하고 환경을 자주 백업하세요. AWS Cloud9은 자동 백업을 수행하지 않습니다.
- 환경에서 소프트웨어를 정기적으로 업데이트하세요. AWS Cloud9은 자동 소프트웨어 업데이트를 수행하지 않습니다.
- AWS 계정에서 AWS CloudTrail을 활성화하여 환경에서의 활동을 추적하세요. 자세한 내용은 AWS CloudTrail을 사용하여 AWS Cloud9 API 호출 로깅 섹션을 참조하세요.
- 신뢰할 수 있는 사용자와만 환경을 공유하세요. 환경을 공유할 경우 AWS 액세스 자격 증명이 훼손될 위험이 있습니다. 자세한 내용은 AWS Cloud9의 공유 환경 작업 섹션을 참조하세요.

다음 주제 중 하나 이상을 읽어 AWS Cloud9 IDE로 작업하는 방법을 알아보세요.

주제

- AWS Cloud9 통합 개발 환경(IDE)의 프로젝트 설정 작업
- AWS Cloud9 IDE의 사용자 설정 작업
- AWS Cloud9 통합 개발 환경(IDE)의 AWS 프로젝트 및 사용자 설정 작업
- AWS Cloud9 통합 개발 환경(IDE)의 키 바인딩 작업
- AWS Cloud9 통합 개발 환경(IDE)의 테마 작업
- AWS Cloud9 통합 개발 환경(IDE)에서 초기화 스크립트 관리
- AWS Cloud9 통합 개발 환경(IDE)에 대한 macOS 기본 키 바인딩 참조
- AWS Cloud9 통합 개발 환경(IDE)에 대한 macOS Vim 키 바인딩 참조
- AWS Cloud9 통합 개발 환경(IDE)에 대한 macOS Emacs 키 바인딩 참조
- AWS Cloud9 통합 개발 환경(IDE)에 대한 macOS Sublime 키 바인딩 참조
- AWS Cloud9 통합 개발 환경(IDE)에 대한 Windows/Linux 기본 키 바인딩 참조
- AWS Cloud9 통합 개발 환경(IDE)에 대한 Windows/Linux Vim 키 바인딩 참조
- AWS Cloud9 통합 개발 환경(IDE)에 대한 Windows/Linux Emacs 키 바인딩 참조
- AWS Cloud9 통합 개발 환경(IDE)에 대한 Windows/Linux Sublime 키 바인딩 참조
- AWS Cloud9 통합 개발 환경(IDE)의 명령 참조

AWS Cloud9 IDE 둘러보기

이 주제에서는 AWS Cloud9 통합 개발 환경(IDE)의 기본 둘러보기를 제공합니다. 이 둘러보기를 최대한 활용하려면 아래 표시된 단계를 순서대로 수행하십시오.

주제

- 필수 조건
- 1단계: 메뉴 모음
- 2단계: 대시보드
- 3단계: 환경 창
- 4단계: 편집기, 탭 및 창
- 5단계: 콘솔
- 6단계: 파일 열기 섹션
- 7단계: 거터
- 8단계: 상태 표시줄
- 9단계: 개요 창

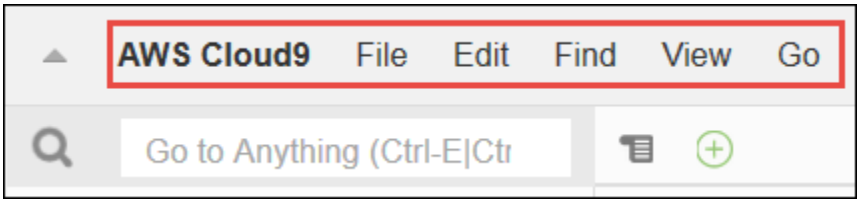
필수 조건

이 둘러보기를 계속 진행하려면 AWS 계정과 열려 있는 AWS Cloud9 개발 환경이 있어야 합니다. 작업 방법을 알아보려면 시작하기: AWS Cloud9에 대한 기본 자습서의 단계를 수행할 수 있습니다. 또는 AWS Cloud9 설정 및 환경 관련 작업 AWS Cloud9 등과 같은 개별 관련 항목을 살펴볼 수 있습니다.

> **Warning**
>
> AWS Cloud9 개발 환경이 있으면 AWS 계정에 요금이 발생할 수 있습니다. EC2 환경을 사용 중인 경우 Amazon EC2 요금이 포함될 수 있습니다. 자세한 내용은 Amazon EC2 요금을 참조하세요.

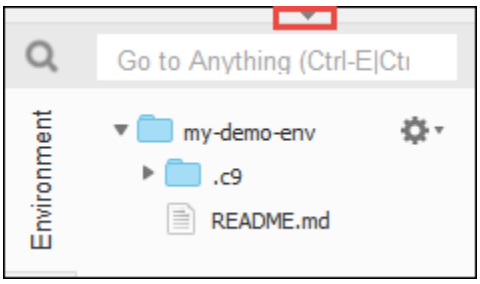
1단계: 메뉴 모음

IDE 상단 가장자리에 있는 menu bar(메뉴 모음)에는 파일과 코드로 작업하고 IDE 설정을 변경하기 위한 일반적인 명령이 포함되어 있습니다. 메뉴 모음에서 코드를 미리 보고 실행할 수도 있습니다.

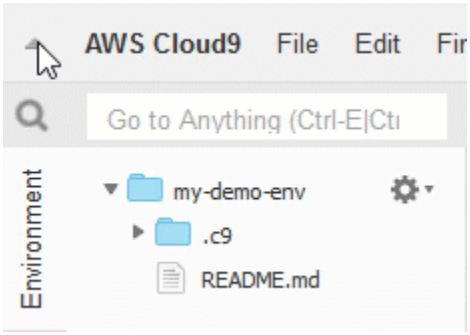


다음과 같이 가장자리의 화살표를 선택하여 메뉴 모음을 숨길 수 있습니다.

다음과 같이 전에 메뉴 모음이 있던 위치의 중간에 있는 화살표를 선택하여 메뉴 모음을 다시 표시할 수 있습니다.



결과를 다음과 비교합니다.



이 자습서의 이후 여러 단원에서 IDE를 사용하여 파일 세트를 작업할 수 있습니다. 이러한 파일을 설정하려면 File(파일), New File(새 파일)을 선택합니다.

다음에는, 다음 텍스트를 `Untitled1` 편집기 탭으로 복사합니다.

```
fish.txt
--------
A fish is any member of a group of organisms that consist of
all gill-bearing aquatic craniate animals that lack limbs with
digits. They form a sister group to the tunicates, together
forming the olfactores. Included in this definition are
lampreys and cartilaginous and bony fish as well as various
extinct related groups.
```

파일을 저장하려면 File(파일), Save(저장)를 선택합니다. 파일에 `fish.txt`라는 이름을 지정한 다음 Save(저장)를 선택합니다.

이러한 지침을 반복하여 다음 콘텐츠가 있는 두 번째 파일을 `cat.txt`로 저장합니다.

```
cat.txt
-------
The domestic cat is a small, typically furry, carnivorous mammal.
They are often called house cats when kept as indoor pets or
simply cats when there is no need to distinguish them from
```

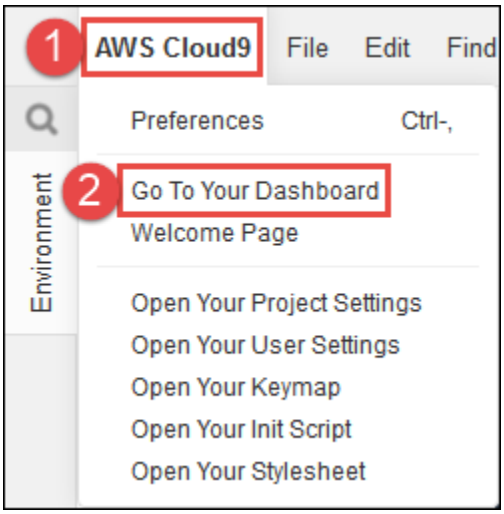
```
other felids and felines. Cats are often valued by humans for
companionship and for their ability to hunt.
```

흔히 여러 가지 방법으로 IDE에서 작업을 수행할 수 있습니다. 예를 들어, 메뉴 모음을 숨기려면 가장 자리에 있는 화살표를 선택하는 대신 View(보기), Menu Bar(메뉴 모음)를 선택할 수 있습니다. 새 파일을 생성하려면 File, New File(파일, 새 파일)을 선택하는 대신 `Alt-N`(Windows/Linux의 경우) 또는 `Control-N`(MacOS의 경우)을 누를 수 있습니다. 이 자습서의 길이를 줄이기 위해 여기서는 작업을 수행하는 한 가지 방법만 설명합니다. IDE에 더 익숙해지면 자유롭게 실험하여 자신에게 가장 효과적인 방법을 알아내십시오.

2단계: 대시보드

dashboard(대시보드)에서는 각 환경에 빠르게 액세스할 수 있습니다. 대시보드에서 환경에 대한 설정을 생성하고 열고 변경할 수 있습니다.

대시보드를 열려면 메뉴 모음에서 AWS Cloud9, Go To Your Dashboard(대시보드로 이동)를 선택합니다.



환경에 대한 설정을 보려면 my-demo-environment 카드 내부에서 제목을 선택합니다. 대시보드로 돌아가려면 웹 브라우저의 뒤로 버튼이나 환경이라는 탐색 이동 경로를 사용합니다.

사용자 환경의 IDE를 열려면 my-demo-environment 카드 내부에서 [IDE 열기(Open IDE)]를 선택합니다.

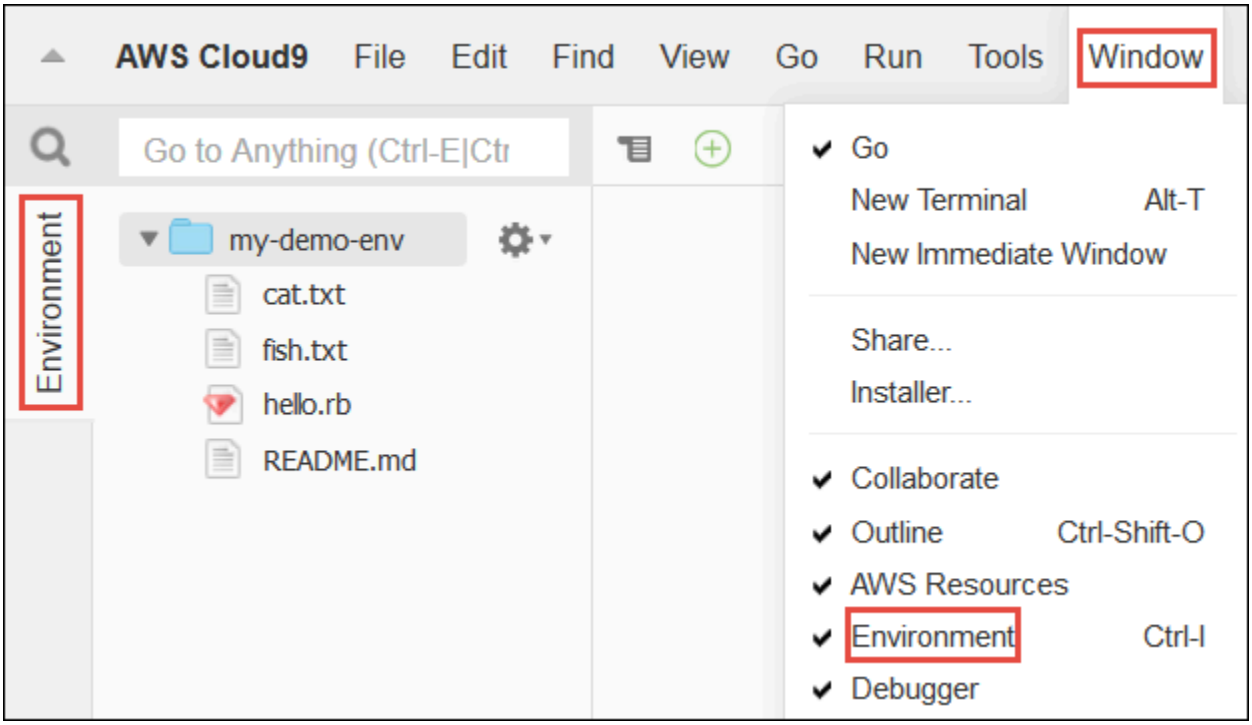
> **Note**
>
> IDE가 다시 표시되려면 몇 분 정도 걸릴 수 있습니다.

3단계: 환경 창

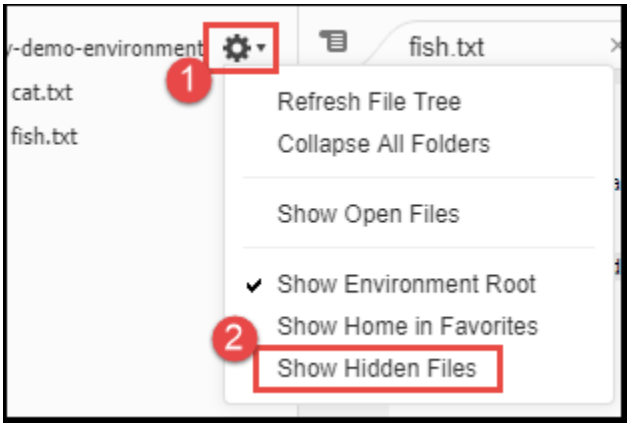
[환경(Environment)] 창에는 환경에 있는 폴더와 파일의 목록이 표시됩니다. 숨겨진 파일과 같은 다양한 유형의 파일도 표시할 수 있습니다.

[환경(Environment)] 창을 표시하거나 숨기려면 [환경(Environment)] 버튼을 선택합니다.

환경(Environment) 창과 환경(Environment) 버튼을 표시하거나 숨기려면 메뉴 모음에서 창(Window), 환경(Environment)을 선택합니다.



숨김 파일을 표시하거나 숨기려면 [환경(Environment)] 창에서 기어 모양 아이콘을 선택한 다음 [숨김 파일 표시(Show Hidden Files)]를 선택합니다.

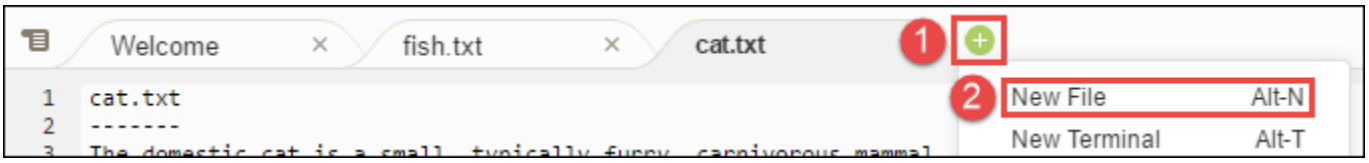


4단계: 편집기, 탭 및 창

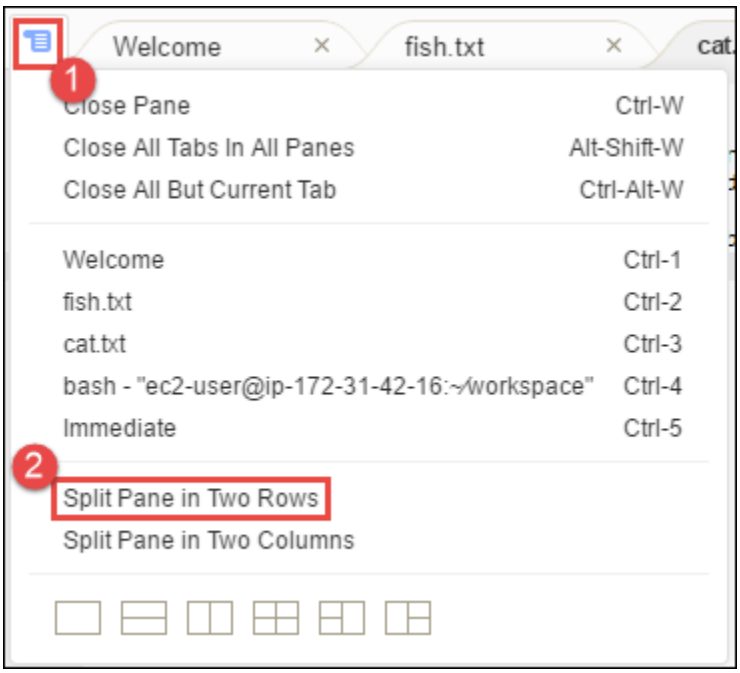
편집기에서는 코드 작성, 터미널 세션 실행, IDE 설정 변경과 같은 작업을 수행할 수 있습니다. 열려 있는 파일, 터미널 세션 등의 각 인스턴스는 탭으로 표현됩니다. 탭은 창으로 그룹화할 수 있습니다. 탭은 각 창의 가장자리에 표시됩니다.

탭을 표시하거나 숨기려면 메뉴 모음에서 보기, Tab Buttons(탭 버튼)를 선택합니다.

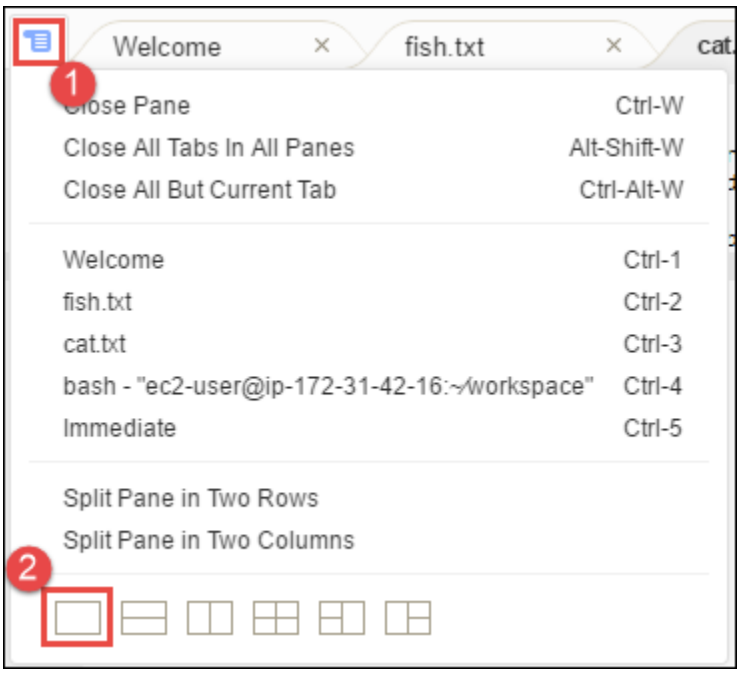
새 탭을 열려면 탭 행의 가장자리에 있는 + 아이콘을 선택합니다. 그런 다음, 아래와 같이 사용 가능한 명령 중 하나를 선택합니다. 예를 들면 New File(새 파일)을 선택할 수 있습니다.



두 개의 창을 표시하려면 탭 행의 가장자리에 있는 드롭다운 메뉴 모양의 아이콘을 선택합니다. 그런 다음, 아래와 같이 Split Pane in Two Rows(두 행으로 창 분할)을 선택합니다.

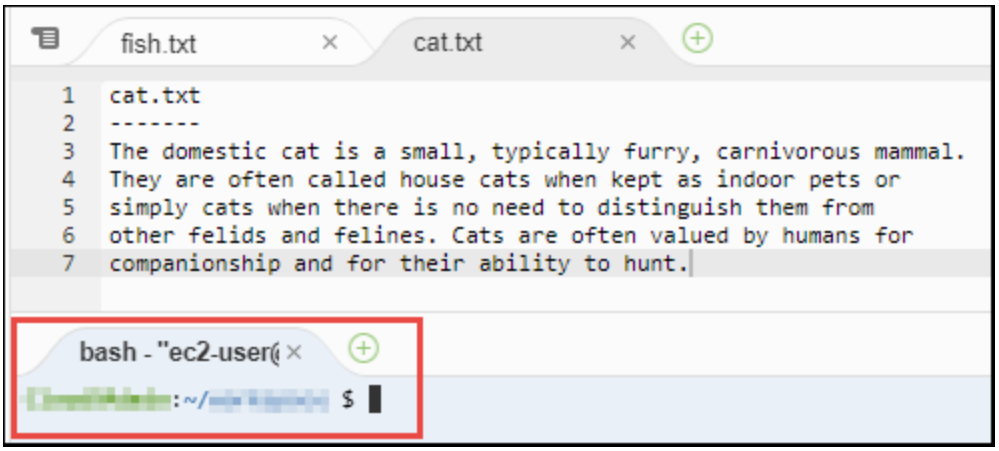


단일 창으로 돌아가려면 다음과 같이 드롭다운 메뉴 아이콘을 다시 선택한 다음 단일 사각형 아이콘을 선택합니다.



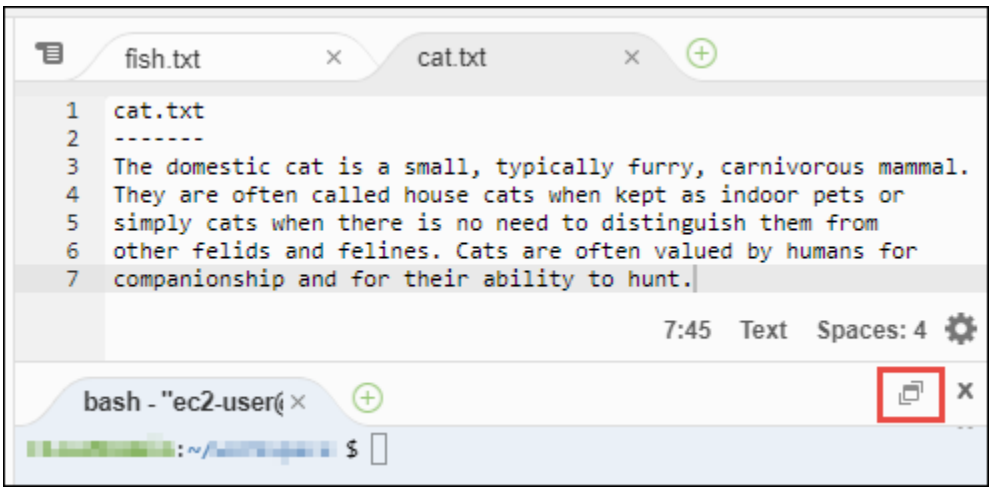
5단계: 콘솔

콘솔에서도 탭을 생성하고 관리할 수 있습니다. 기본적으로 터미널 탭이 있지만 다른 탭 유형도 콘솔에 포함할 수 있습니다.



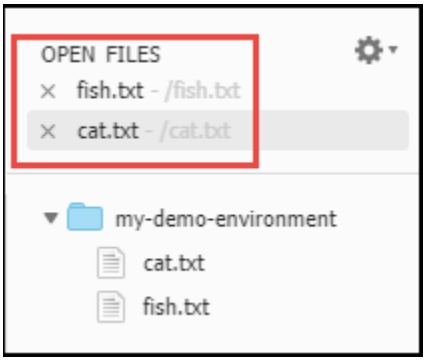
콘솔을 표시하거나 숨기려면 메뉴 모음에서 보기, Console(콘솔)을 선택합니다.

콘솔을 확장하거나 축소하려면 다음과 같이 콘솔 가장자리에 있는 크기 조정 아이콘을 선택합니다.



6단계: 파일 열기 섹션

Open Files(열려 있는 파일) 섹션에는 편집기에 현재 열려 있는 모든 파일의 목록이 표시됩니다. [파일 열기(Open Files)]는 [환경(Environment)] 창의 일부입니다.

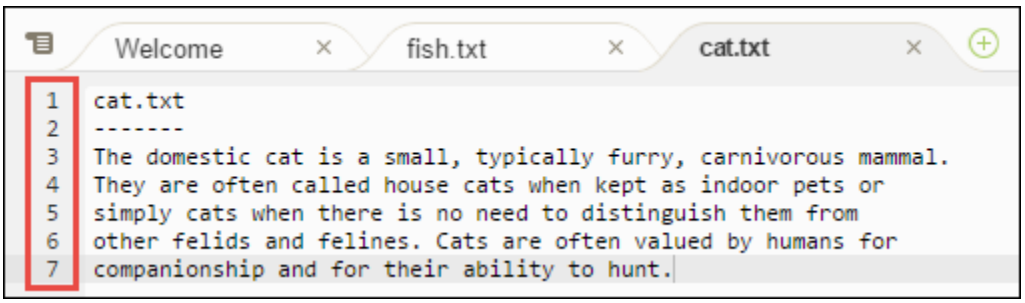


Open Files(열려 있는 파일) 섹션을 표시하거나 숨기려면 메뉴 모음에서 보기, Open Files(열려 있는 파일)를 선택합니다.

열려 있는 다른 파일로 이동하려면 목록에서 원하는 파일을 선택합니다.

7단계: 거터

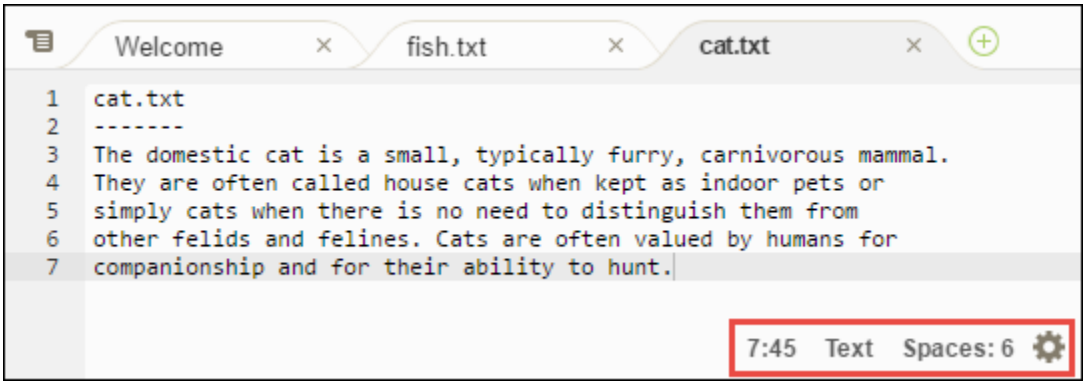
파일 작업 시 편집기에서 각 파일의 가장자리에 있는 거터에는 숫자 및 컨텍스트 기호와 같은 항목이 표시됩니다.



거터를 표시하거나 숨기려면 메뉴 모음에서 보기, Gutter(거터)를 선택합니다.

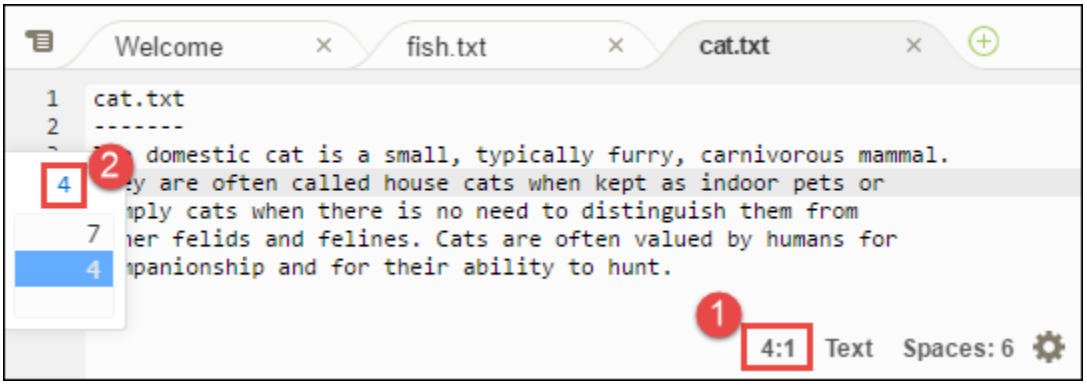
8단계: 상태 표시줄

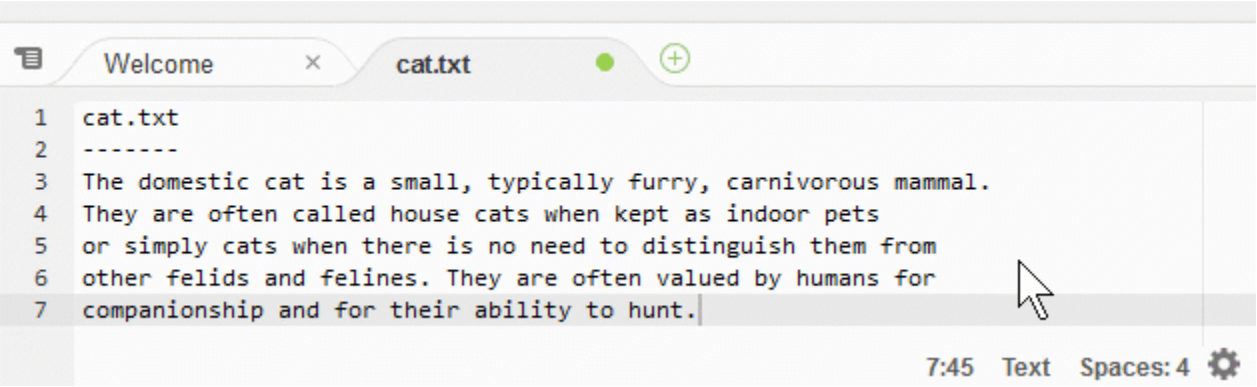
편집기에서 각 파일의 가장자리에 있는 상태 표시줄에는 줄 및 문자 번호, 파일 형식 기본 설정, 공백 및 탭 설정, 관련 편집기 설정과 같은 항목이 표시됩니다.



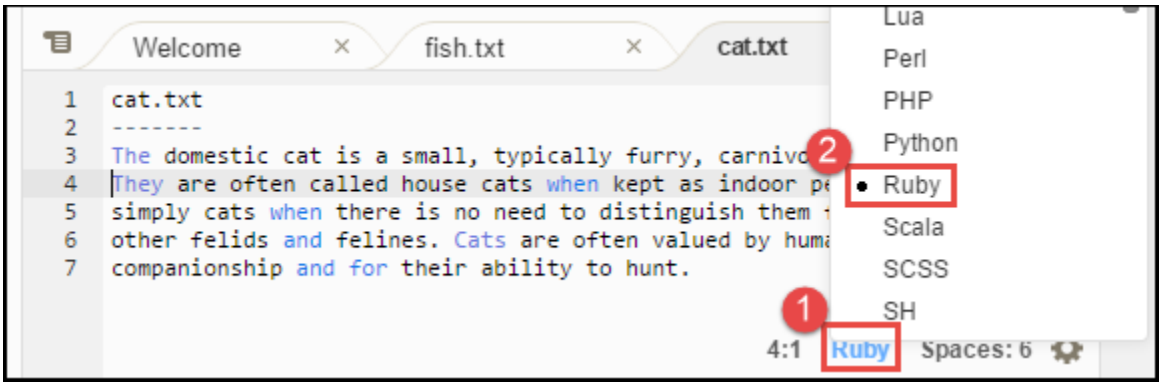
상태 표시줄을 표시하거나 숨기려면 메뉴 모음에서 보기, Status Bar(상태 표시줄)를 선택합니다.

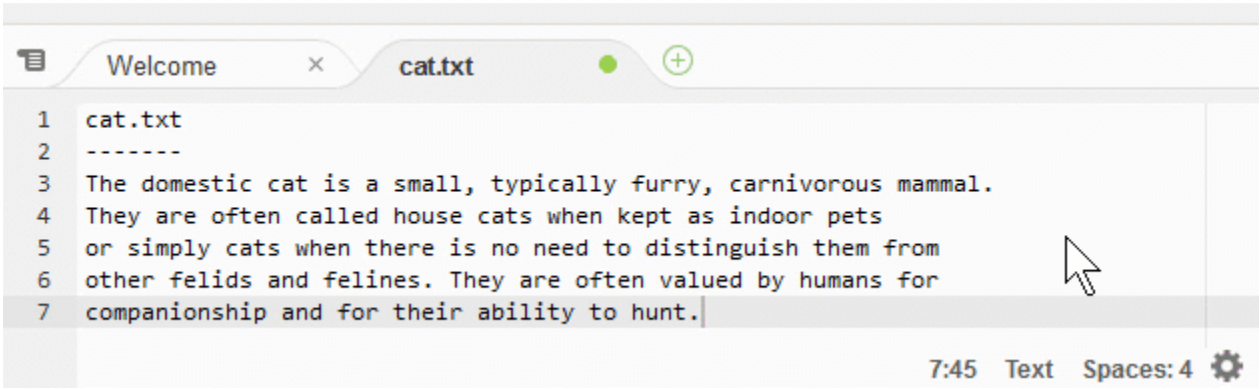
특정 줄 번호로 이동하려면 원하는 파일이 있는 탭을 선택합니다. 그런 다음 상태 표시줄에서 줄 및 문자 번호를 선택합니다(이 번호는 7:45와 같은 형식이어야 합니다). 줄 번호(예: 4)를 입력한 다음 `Enter`를 누릅니다.





파일 유형 기본 설정을 변경하려면 상태 표시줄에서 다른 파일 유형을 선택합니다. 예를 들어, cat.txt의 경우 Ruby를 선택하여 구문 색상 변경을 봅니다. 일반 텍스트로 돌아가려면 다음과 같이 Plain Text(일반 텍스트)를 선택합니다.





9단계: 개요 창

Outline(개요) 창을 사용하여 특정 파일 위치로 빠르게 이동할 수 있습니다.

개요(Outline) 창과 개요(Outline) 버튼을 표시하거나 숨기려면 메뉴 모음에서 창(Window), 개요(Outline)를 선택합니다.

Outline(개요) 창이 작동하는 방식을 보려면 `hello.rb`라는 파일을 생성합니다. 다음 코드를 파일로 복사하고 저장합니다.

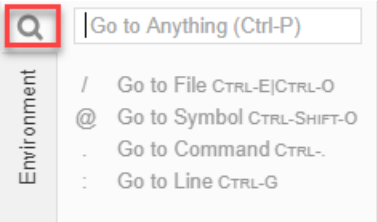
```
def say_hello(i)
  puts "Hello!"
  puts "i is #{i}"
end

def say_goodbye(i)
  puts "i is now #{i}"
  puts "Goodbye!"
end

i = 1
say_hello(i)
i += 1
say_goodbye(i)
```

Outline(개요) 창 내용을 표시하거나 숨기려면 Outline(개요) 버튼을 선택합니다.

아래와 같이 Outline(개요) 창에서 say_hello(i)를 선택한 후 say_goodbye(i)를 선택합니다.

10단계: 이동 창

Go(이동) 창을 사용하여 편집기에서 파일을 열거나, 기호 정의로 이동하거나, 명령을 실행하거나, 편집기에 있는 활성 파일의 한 줄로 이동할 수 있습니다.



이동 창의 내용을 표시하려면 이동 버튼(돋보기 아이콘)을 선택합니다.

이동(Go) 창과 이동(Go) 버튼을 표시하거나 숨기려면 메뉴 모음에서 창(Window), 이동(Go)을 선택합니다.

이동 창이 열린 상태에서 다음 작업을 할 수 있습니다.

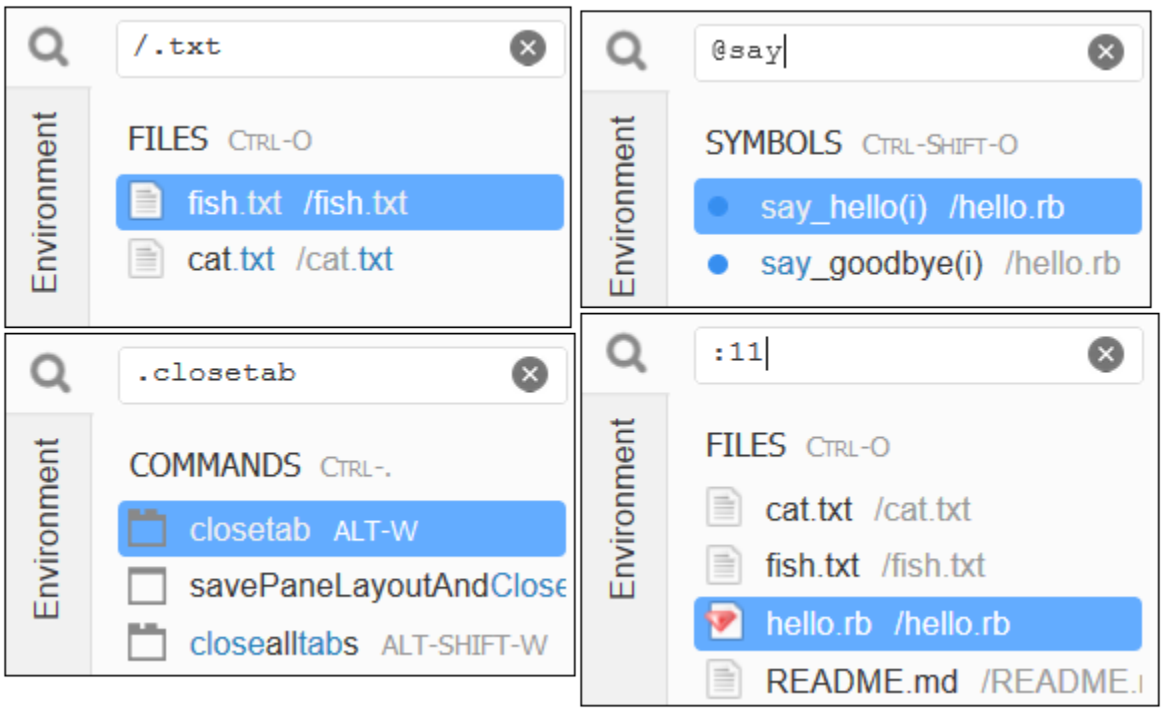
- 슬래시(/)를 입력한 다음 파일 이름의 일부 또는 전부를 입력합니다. 다음에 표시되는 일치하는 파일 목록에서 파일을 선택하여 편집기에서 해당 파일을 엽니다. 예를 들어, `/fish`를 입력하면 `fish.txt`가 나열되는 반면, `/.txt`를 입력하면 `fish.txt`와 `cat.txt`가 모두 나열됩니다.

> **Note**
>
> 파일 검색은 Environment(환경) 창에서 숨겨지지 않은 파일 및 숨겨지지 않은 폴더 범위에서만 수행됩니다.

- 기호(@)를 입력한 다음 기호의 이름을 입력합니다. 다음에 표시되는 일치하는 기호 목록에서 기호를 선택하여 편집기에서 해당 기호로 이동합니다. 예를 들어, 편집기에 `hello.rb` 파일이 열려 있고 활성인 상태에서 `@hello`를 입력하여 `say_hello(i)`를 나열하거나, `@say`를 입력하여 `say_hello(i)`와 `say_goodbye(i)`를 모두 나열합니다.

> **Note**
>
> 편집기에 있는 활성 파일이 지원되는 언어 프로젝트의 일부인 경우 기호 검색은 현재 프로젝트의 범위에서 수행됩니다. 그렇지 않으면 기호 검색은 편집기에 있는 활성 파일의 범위에서만 수행됩니다. 자세한 내용은 향상된 TypeScript 지원 및 기능 섹션을 참조하세요.

- 점(`.`)을 입력한 다음 명령의 이름을 입력합니다. 다음에 표시되는 명령 목록에서 명령을 선택하여 해당 명령을 실행합니다. 예를 들어, `.closetab`를 입력한 다음 `Enter`를 누르면 편집기에서 현재 탭이 닫힙니다. 사용할 수 있는 명령의 목록은 AWS Cloud9 통합 개발 환경(IDE)의 명령 참조 단원을 참조하십시오.
- 콜론(`:`)을 입력한 다음 번호를 입력하여 편집기의 활성 파일에 있는 해당 줄 번호로 이동합니다. 예를 들어, 편집기에 `hello.rb` 파일이 열려 있고 활성인 상태에서 `:11`을 입력하여 해당 파일의 11번 줄로 이동합니다.



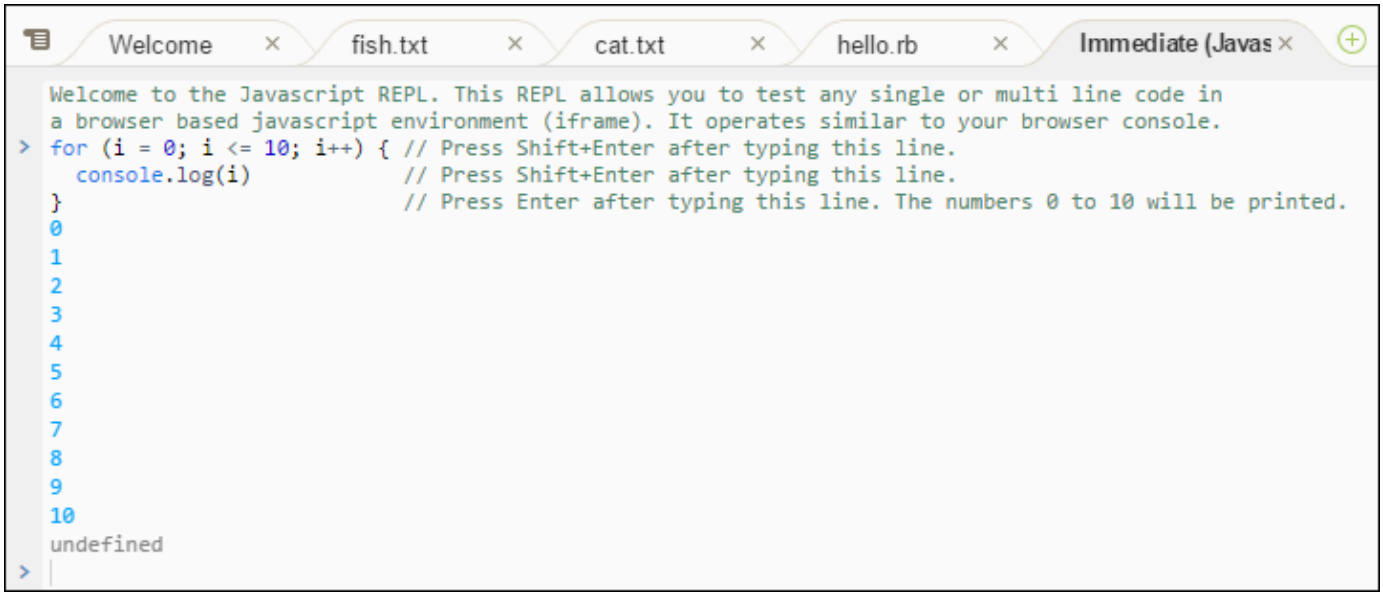
현재 키보드 모드 및 운영 체제에 따른 이러한 각 작업에 대한 키 바인딩을 보려면 메뉴 모음의 [이동(Go)] 메뉴에서 사용 가능한 각 [이동(Go To)] 명령을 참조하세요.

11단계: 직접 실행 탭

Immediate(즉시) 탭을 사용하여 JavaScript 코드의 작은 조각을 테스트할 수 있습니다. Immediate(즉시) 탭이 작동하는 방식을 보려면 다음을 수행합니다.

1. 메뉴 모음에서 [창(Window)], [새 직접 실행 창(ew Immediate Window)]을 선택하여 [직접 실행(Immediate)] 탭을 엽니다.
2. Immediate(즉시) 탭에서 코드를 실행합니다. 이렇게 해 보려면 1번 줄을 입력한 후 `Shift-Enter`를 누르고 2번 줄을 입력한 후 다시 눌러 다음 코드를 창에 입력합니다. 3번 줄을 입력한 후 `Enter`를 누릅니다. (1번 줄 또는 2번 줄을 입력한 후 `Shift-Enter` 대신 `Enter`를 누르면 원하는 것보다 일직 코드가 실행됩니다.)

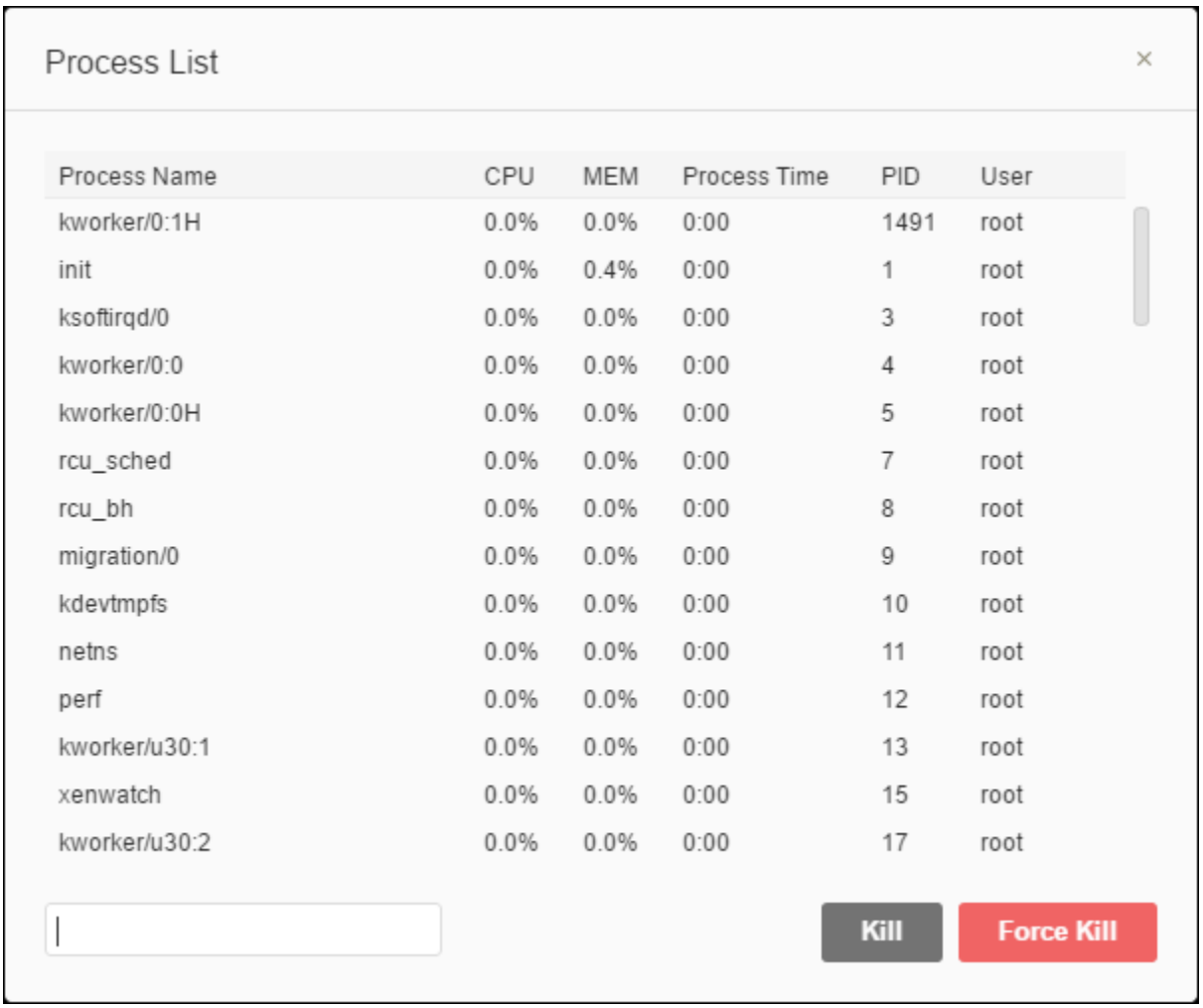
```
for (i = 0; i <= 10; i++) { // Press Shift-Enter after typing this line.
  console.log(i)            // Press Shift-Enter after typing this line.
}                           // Press Enter after typing this line. The numbers 0 to
 10 will be printed.
```



12단계: 프로세스 목록

Process List(프로세스 목록)에는 실행 중인 모든 프로세스가 표시됩니다. 더 이상 실행하지 않으려는 프로세스를 중지하거나 강제로 중지할 수도 있습니다. Process List(프로세스 목록) 창이 작동하는 방식을 보려면 다음을 수행합니다.

1. 메뉴 모음에서 [도구(Tools)], [프로세스 목록(Process List)]을 선택하여 [프로세스 목록(Process List)]을 표시합니다.
2. 프로세스를 찾습니다. Process List(프로세스 목록)에 프로세스의 이름을 입력합니다.
3. 프로세스를 중지하거나 강제로 중지합니다. 프로세스 목록에서 프로세스를 선택한 다음 Kill(종료) 또는 Force Kill(강제 종료)을 선택합니다.

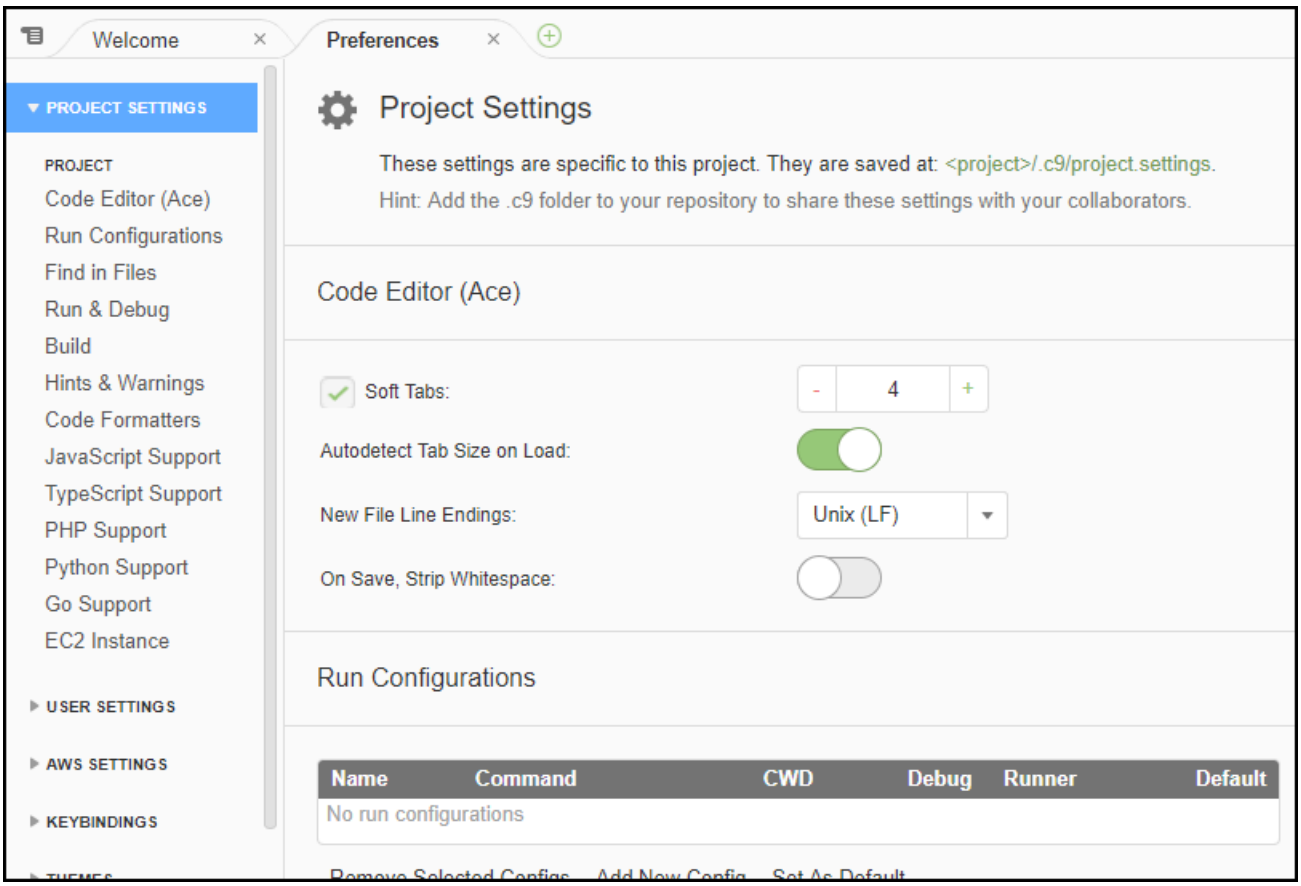


13단계: 기본 설정

기본 설정에는 다음 설정이 포함됩니다.

- 현재 환경에만 적용되는 대한 설정(예: 편집기에서 소프트 탭을 사용할지 여부, 무시할 파일 유형, PHP 및 Python과 같은 언어에 대한 코드 완성 동작).
- 각 환경 전체의 사용자 설정(예: 색상, 글꼴, 편집기 동작).
- 키 결합(파일 및 편집기 작업에 기본적으로 사용할 바로 가기 키 조합).
- IDE의 전체 테마.

기본 설정을 표시하려면 메뉴 모음에서 AWS Cloud9, 기본 설정을 선택합니다. 다음과 같이 표시됩니다.



14단계: 터미널

IDE에서 하나 이상의 터미널 세션을 실행할 수 있습니다. 터미널 세션을 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다. 또는 콘솔 탭 옆의 "더하기" 아이콘을 선택하고 New Terminal(새 터미널)을 선택합니다.

터미널에서 명령을 실행해 볼 수 있습니다. 예를 들어 터미널에서 `echo $PATH`를 입력한 후 `Enter` 키를 눌러 PATH 환경 변수의 값을 인쇄합니다.

추가 명령을 실행해 볼 수도 있습니다. 예를 들어, 다음과 같은 명령을 실행해 볼 수 있습니다.

- **`pwd`** - 현재 디렉터리의 경로를 인쇄합니다.
- **`aws --version`** - AWS CLI에 대한 버전 정보를 인쇄합니다.
- **`ls -l`** - 현재 디렉터리에 대한 정보를 인쇄합니다.

15단계: 디버거 창

Debugger(디버거) 창을 사용하여 코드를 디버깅할 수 있습니다. 예를 들어, 한 번에 한 부분씩 단계적으로 코드를 실행하고, 시간에 따른 변수의 값을 본 다음, 호출 스택을 탐색할 수 있습니다.

> **Note**
>
> 이 절차는 기본 IDE 자습서 중 하나의 2단계: IDE의 기본 사항 둘러보기 섹션과 유사합니다.

디버거(Debugger) 창과 디버거(Debugger) 버튼을 표시하거나 숨기려면 메뉴 모음에서 창(Window), 디버거(Debugger)를 선택합니다.

이 자습서에서는 다음과 같이 Debugger(디버거) 창과 JavaScript 코드 몇 가지를 실험할 수 있습니다.

1. 터미널 세션에서 **`node --version`** 명령을 실행하여 사용자 환경에서 Node.js 설치를 확인합니다. Node.js가 설치된 경우 Node.js 버전 번호가 출력에 표시되고 이 절차의 3단계로 건너뛸 수 있습니다("JavaScript 코드 작성...").

2. Node.js를 설치해야 할 경우 다음을 수행합니다.

   a. 다음 두 개의 명령을 한 번에 하나씩 실행하여 환경에 최신 업데이트가 있는지 확인한 다음 노드 버전 관리자(nvm)를 다운로드합니다. (nvm은 Node.js 버전을 설치하고 관리하는 데 유용한 간단한 Bash 셸 스크립트입니다. 자세한 내용은 GitHub에서 Node Version Manager를 참조하세요.)

      Amazon Linux의 경우:

      ```
      sudo yum -y update
      curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh |
       bash
      ```

      Ubuntu Server:

      ```
      sudo apt update
      curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh |
       bash
      ```

   b. 텍스트 편집기를 사용하여 nvm을 로드할 수 있도록 셸 프로필 파일(예: `~/.bashrc`)을 업데이트합니다. 예를 들어 IDE의 [환경(Environment)] 창에서 기어 모양 아이콘을 선택한 다음 [즐겨찾기에 홈 표시(Show Home in Favorites)]를 선택합니다. 이 단계를 반복하고 Show Hidden Files(숨겨진 파일 표시)도 선택합니다.

   c. `~/.bashrc` 파일을 엽니다.

   d. nvm을 로드할 수 있도록 다음 코드를 파일의 끝 부분에 입력하거나 붙여 넣습니다.

      Amazon Linux의 경우:

      ```
      export NVM_DIR="/home/ec2-user/.nvm"
      [ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh"  # This loads nvm.
      ```

      Ubuntu Server:

      ```
      export NVM_DIR="/home/ubuntu/.nvm"
      [ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh"  # This loads nvm.
      ```

   e. 파일을 저장합니다.

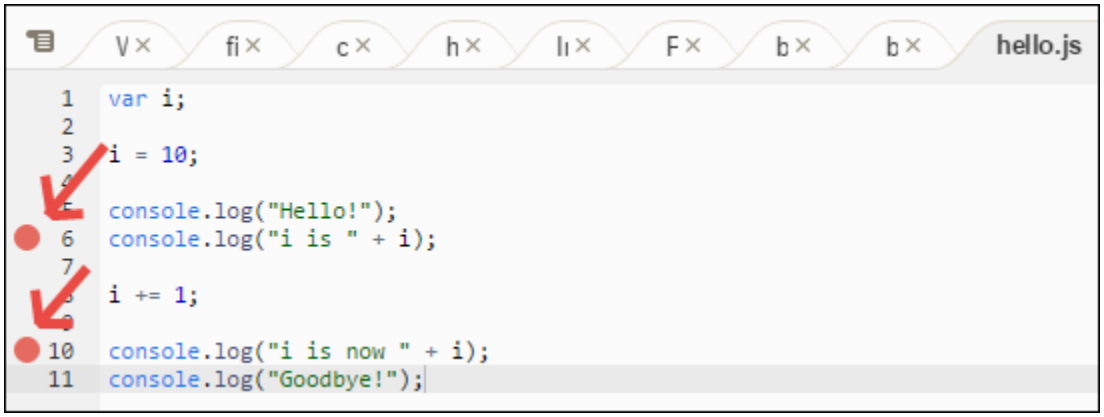
   f. 터미널 세션을 닫고 새로운 세션을 시작합니다. 다음 명령을 실행하여 최신 버전의 Node.js를 설치합니다.

```
nvm install node
```

3. 디버깅할 JavaScript 코드를 작성합니다. 예를 들어, 파일을 생성하고, 다음 코드를 파일에 추가한 다음, 파일을 `hello.js`로 저장합니다.

```
var i;

i = 10;

console.log("Hello!");
console.log("i is " + i);

i += 1;

console.log("i is now " + i);
console.log("Goodbye!");
```

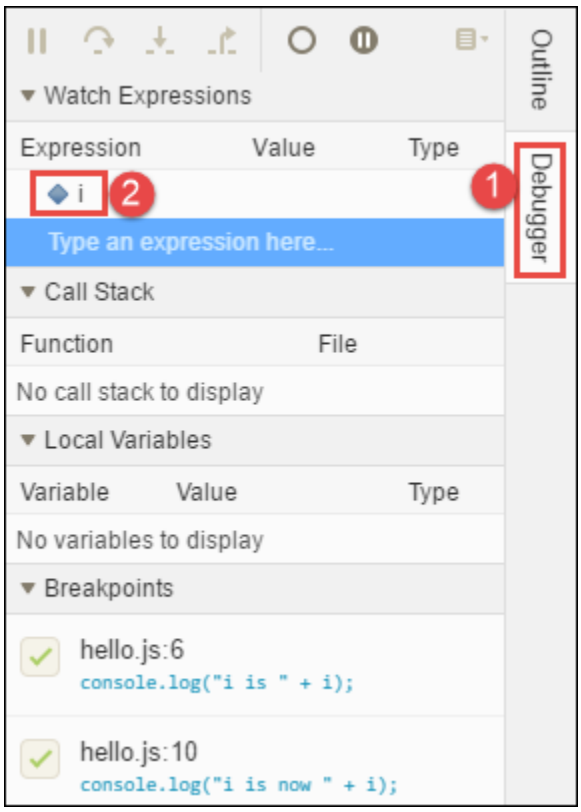
4. 중단점을 코드에 추가합니다. 예를 들어, 거터에서 6번 및 10번 줄 옆의 여백을 선택합니다. 이러한 각 줄 번호 옆에는 다음과 같이 빨간색 원이 표시됩니다.



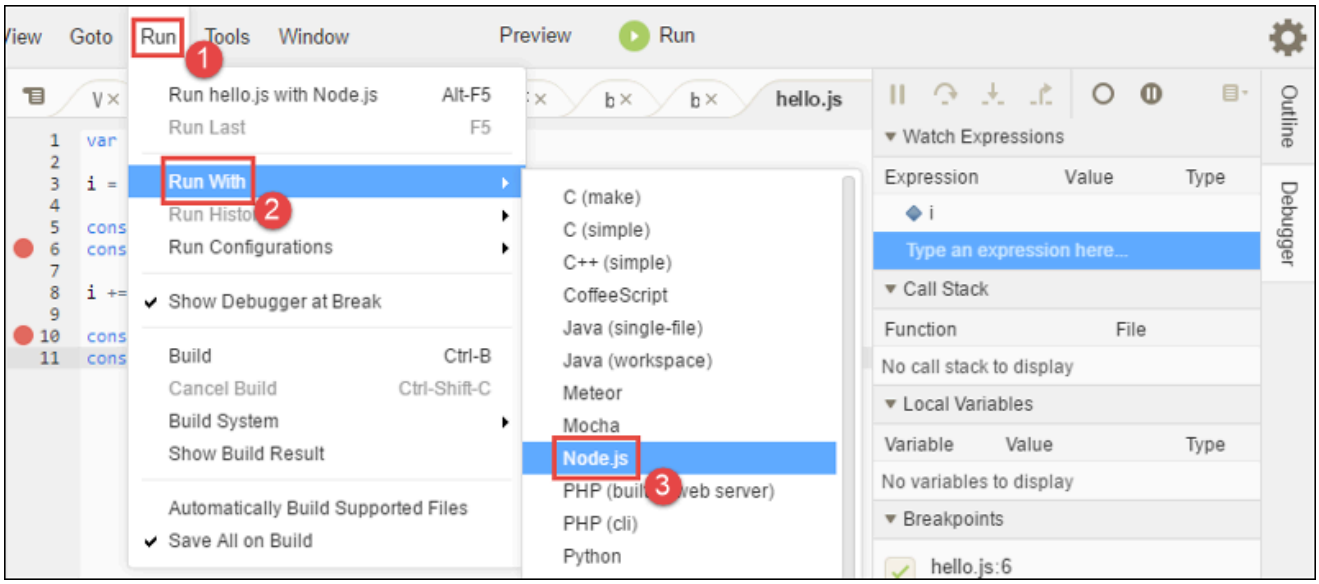
5. 이제 JavaScript 코드를 디버깅할 준비가 되었습니다. 이렇게 해 보려면 다음을 수행합니다.

   a. Debugger(디버거) 창의 내용을 표시하거나 숨기려면 다음 단계에 나온 대로 Debugger(디버거) 버튼을 선택합니다.

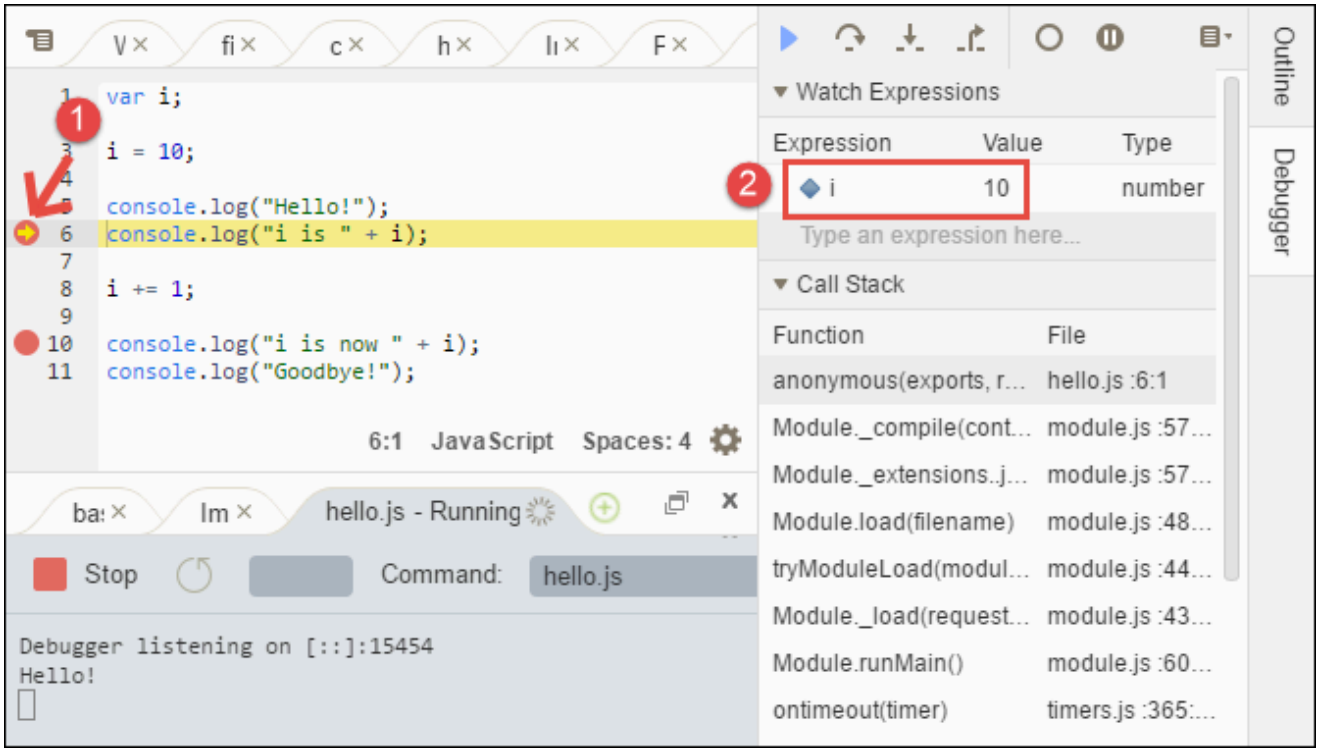
   b. 코드가 실행되는 동안 `i`라는 변수의 값을 봅니다. Debugger(디버거) 창의 Watch Expressions(조사식)에서 Type an expression here(여기에 표현식 입력)를 선택합니다. 다음과 같이 문자 `i`를 입력한 다음 `Enter`를 누릅니다.



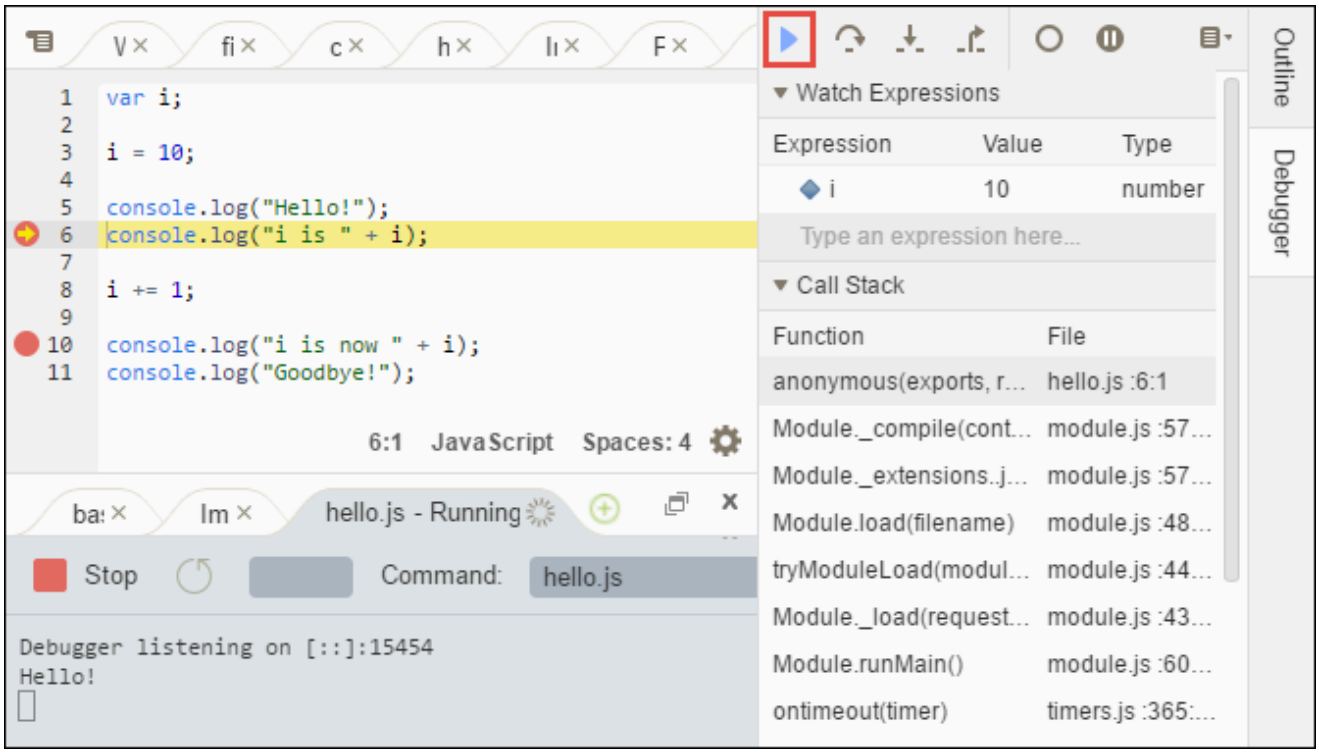
c. 코드 실행을 시작합니다. 다음과 같이 Run(실행), Run With(다음으로 실행), Node.js를 선택합니다.



d. 6번 행에서 코드 실행이 일시 중지됩니다. [디버거(Debugger)] 창에 조사식(Watch Expressions)의 `i` 값(현재 `10`)이 표시됩니다.

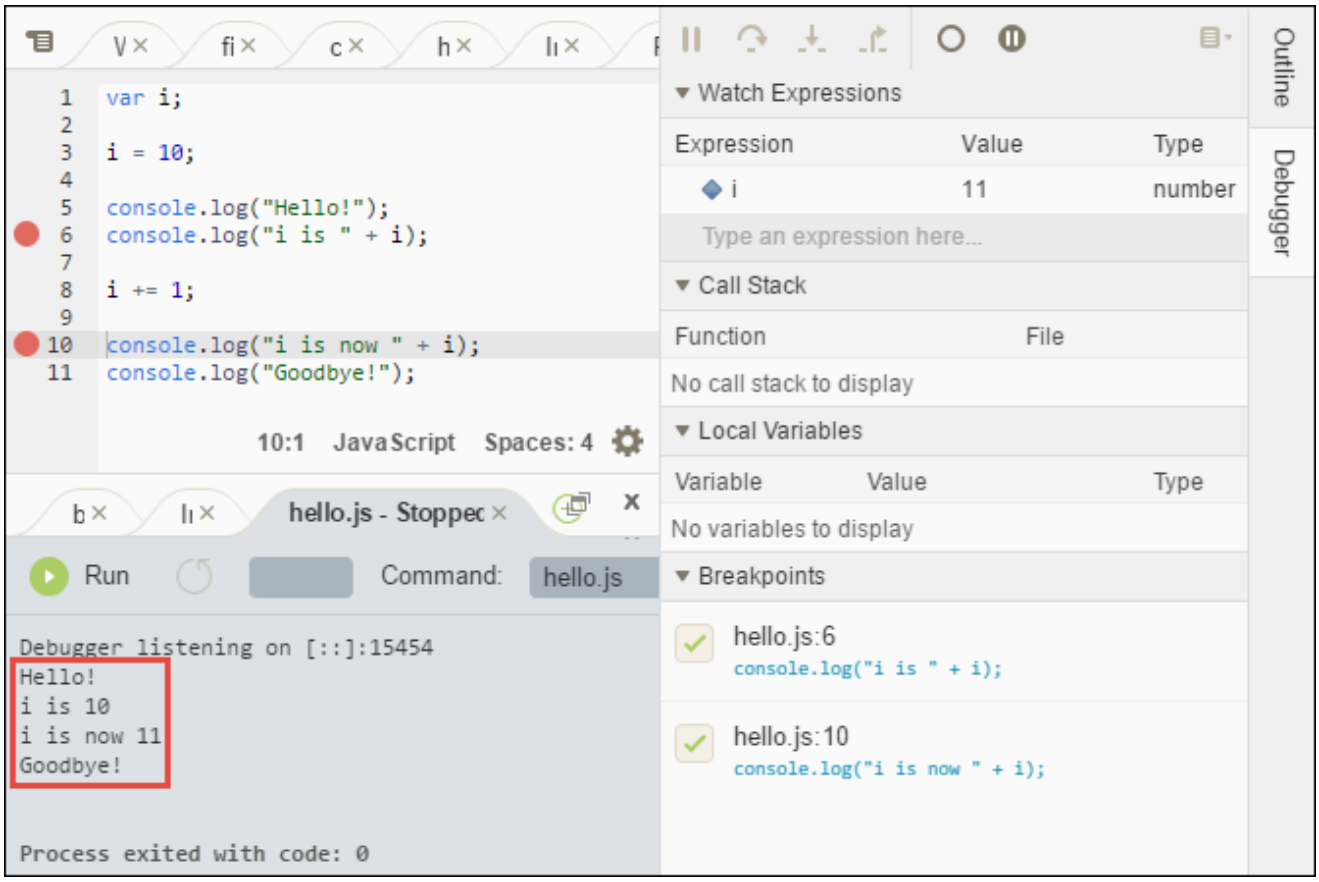


e. 다음과 같이 Debugger(디버거) 창에서 Resume(다시 시작)(파란색 화살표 아이콘)을 선택합니다.

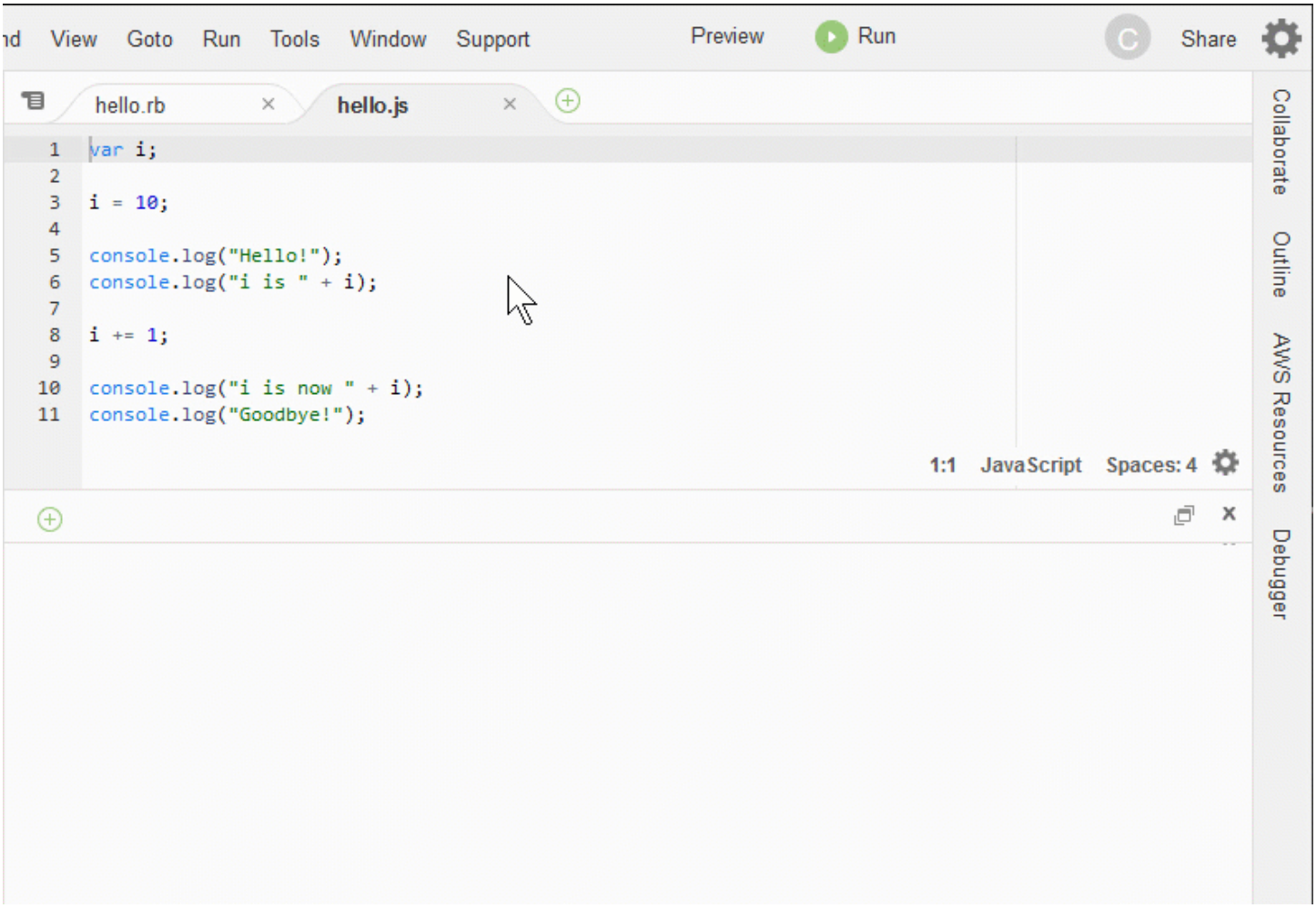


f. 10번 행에서 코드 실행이 일시 중지됩니다. 이제 Debugger(디버거) 창에 `i`의 새 값이 표시됩니다. 이 값은 현재 11입니다.

g. Resume(다시 시작)을 다시 선택합니다. 코드가 끝까지 실행됩니다. 다음과 같이 출력이 콘솔의 hello.js 탭에 인쇄됩니다.



결과를 다음과 비교합니다.



결론

> **Warning**
>
> AWS Cloud9 개발 환경이 있으면 AWS 계정에 요금이 발생할 수 있습니다. EC2 환경을 사용 중인 경우 Amazon EC2 요금이 포함될 수 있습니다. 자세한 내용은 Amazon EC2 요금을 참조하세요.
>
> 상위 섹션(IDE 작업)에 살펴볼 수 있는 추가 항목이 있습니다. 하지만 AWS Cloud9 IDE 둘러보기를 마치고 환경이 더 이상 필요하지 않은 경우 환경 삭제에 설명된 대로 환경 및 연결된 리소스를 삭제해야 합니다.

AWS Cloud9 통합 개발 환경 (IDE) 에서의 언어 지원

AWS Cloud9 IDE는 다양한 프로그래밍 언어를 지원합니다. 다음 표에는 지원되는 언어와 지원 수준이 나열됩니다.

언어	구문 강조 [1]	실행 UI [2]	개요 보기	코드 힌트와 Linting	코드 완성	디버깅 [3]
C++	✓	✓	✓		✓ [5]	✓ [4]
C#	✓		✓		✓ [5]	
CoffeeScript	✓	✓				
CSS	✓				✓	
Dart	✓					
Go	✓	✓	✓	✓	✓ [4]	✓ [4]
Haskell	✓					
HTML	✓	✓	✓		✓	
Java[6]	✓	✓	✓	✓	✓	✓
JavaScript	✓	✓	✓	✓	✓	
Node.js	✓	✓	✓	✓	✓	✓
PHP	✓	✓	✓	✓	✓ [7]	✓
Python	✓	✓	✓	✓	✓ [8]	✓
Ruby	✓	✓	✓	✓	✓ [5]	
셸 스크립트	✓	✓	✓	✓	✓ [5]	

언어	구문 강조 [1]	실행 UI [2]	개요 보기	코드 힌트와 Linting	코드 완성	디버깅 [3]
TypeScript [9]	✓	✓	✓	✓	✓	

참고

[1] AWS Cloud9 IDE는 더 많은 언어에 대한 구문 강조 기능을 제공합니다. 전체 목록을 보려면 IDE의 메뉴 모음에서 [보기, 구문(View, Syntax)]을 선택합니다.

[2] ✓로 표시된 언어의 경우 명령줄을 사용하지 않고 버튼을 클릭하여 프로그램 또는 스크립트를 실행할 수 있습니다. ✓가 표시되어 있지 않거나 실행, 다음으로 실행(Run, Run With) 메뉴 모음에 표시되지 않는 언어의 경우 해당 언어에 대한 러너를 생성할 수 있습니다. 지침은 빌더 또는 실행기 생성을 참조하십시오.

[3] ✓가 표시된 언어의 경우 IDE의 기본 제공 도구를 사용하여 프로그램이나 스크립트를 디버깅할 수 있습니다. 지침은 코드 디버깅을 참조하십시오.

[4] 이 언어에서는 이 기능이 시험 단계에 있습니다. 완전히 구현되지 않았으므로 문서화되거나 지원되지 않습니다.

[5] 이 언어의 경우 이 기능은 로컬 함수만 지원합니다.

[6] Java SE 11 기능에 대한 향상된 지원은 2GiB 이상의 메모리가 있는 AWS Cloud9 EC2 개발 환경에서 활성화할 수 있습니다. 자세한 정보는 Java 개발을 위한 향상된 지원을 참조하세요.

[7] 사용자 지정 PHP 코드를 AWS Cloud9 완성하는 데 사용할 경로를 지정하려면 AWS Cloud9 IDE에서 프로젝트, PHP 지원을 켜고 환경 설정에서 PHP 코드 완성 설정을 활성화한 다음 사용자 지정 코드의 경로를 프로젝트, PHP 지원, PHP 완료 경로 포함 설정에 추가합니다.

[8] 사용자 지정 Python 코드를 AWS Cloud9 완료하는 데 사용할 경로를 지정하려면 AWS Cloud9 IDE에서 프로젝트, Python 지원을 켜고 환경 설정에서 Python 코드 완성 설정을 활성화한 다음 사용자 지정 코드의 경로를 프로젝트, Python 지원, PYTHONPATH 설정에 추가합니다.

[9] AWS Cloud9 IDE는 언어 프로젝트의 컨텍스트 내에서 TypeScript (AWS Cloud9 IDE에서 지원되는 버전 3.7.5) 와 같은 일부 프로그래밍 언어에 대한 추가 지원을 제공합니다. 자세한 내용은 언어 프로젝트 작업을 참조하십시오.

AWS Cloud9 통합 개발 환경 (IDE) 에서 지원되는 프로그래밍 언어 버전

아래 표에는 IDE의 특정 AMI에서 지원되는 프로그래밍 언어 버전이 요약되어 있습니다. AWS Cloud9 Ubuntu 18은 2023년에 EOL이 되었으므로 프로그래밍 언어 버전을 AWS Cloud9에서 업데이트할 수 없습니다.

언어	Amazon Linux 2023	Amazon Linux 2	Ubuntu 18	Ubuntu 22
Python3	3.9	3.8	3.6	3.10
TypeScript	3.7.5	3.7.5	3.7.5	3.7.5
PHP	8.2	8.2	7.2	8.1
Ruby	3.2	3.0	3.0	3.2
Java	11, 17	11	11	11, 17
파이썬 2	N/A	2.7	N/A	N/A
C++*	23	17	17	23
Go	1.20	1.20	1.9	1.21
CoffeeScript	2.7	2.7	2.7	2.7

*다음 명령을 실행하여 사용하려는 프로그래밍 언어 버전으로 C++ 파일을 컴파일할 수 있습니다.

```
g++ -std=c++[version-number] "$file" -o "$file.o"
```

AWS Cloud9 통합 개발 환경(IDE)의 강화된 언어 지원

AWS Cloud9에서 다음 언어로 코딩할 때 개발 경험을 개선하기 위한 향상된 언어 지원을 제공합니다.

- Java: 확장을 통해 코드 완성, 오류에 대한 linting, 컨텍스트별 작업 및 디버깅 옵션과 같은 기능을 제공할 수 있습니다.
- Typescritpt: 언어 프로젝트는 TypeScript의 향상된 생산성 기능에 대한 액세스를 제공합니다.

주제

Java 개발을 위한 향상된 지원

AWS Cloud9에서 Java로 작업할 때 개발 경험을 개선하기 위한 향상된 언어 지원을 제공합니다. 주요 생산성 기능에는 코드 완성, 오류에 대한 linting, 코드 렌즈, 디버깅 옵션(예: 중단점 및 단계별 실행)이 포함됩니다.

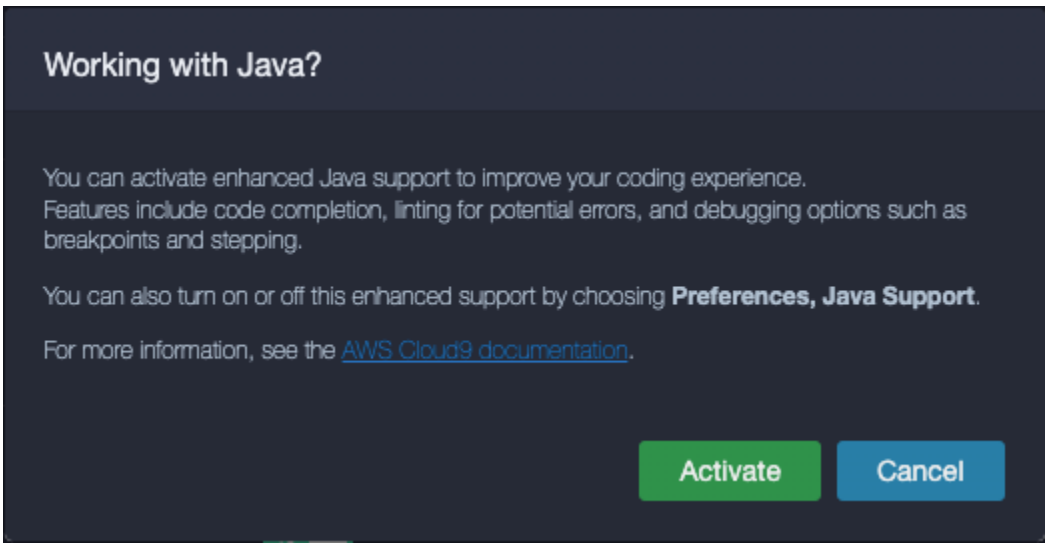
> **Important**
>
> 향상된 생산성 기능은 Amazon EC2 인스턴스에 연결된 AWS Cloud9 개발 환경에만 사용할 수 있습니다.
> 또한 Java에 대한 향상된 언어 지원을 사용할 때 최적의 IDE 환경을 보장하려면 AWS Cloud9 환경을 지원하는 Amazon EC2 컴퓨팅 인스턴스에 2GiB 이상의 메모리가 필요합니다. AWS Cloud9에서 EC2 컴퓨팅 인스턴스에 RAM이 충분하지 않음을 감지하는 경우 Java에 대한 향상된 기능을 활성화하는 옵션이 제공되지 않습니다.

향상된 Java 지원 활성화 및 사용자 지정

다음 조건이 충족되면 Java에 대한 향상된 지원을 활성화하는 옵션이 자동으로 표시됩니다.

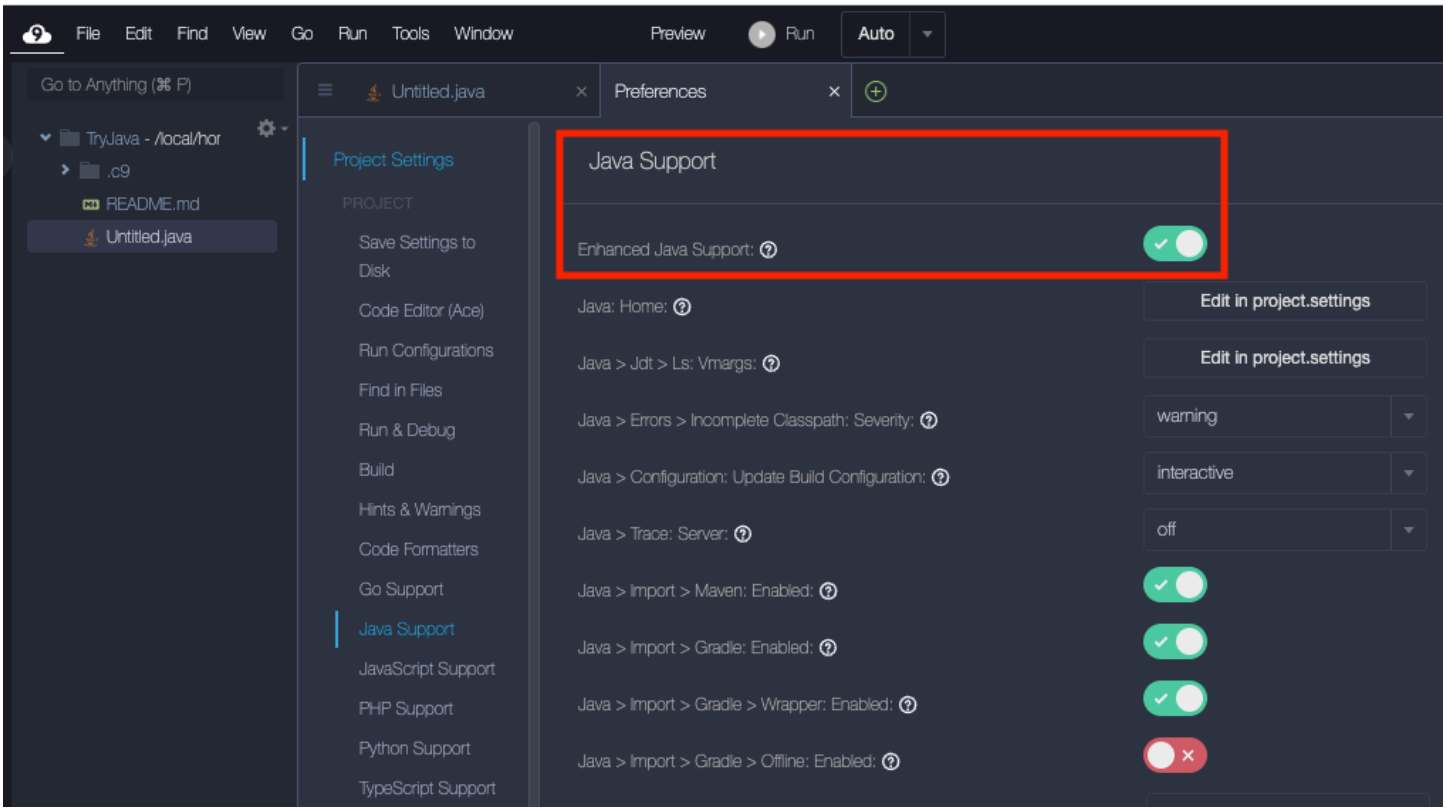
- AWS Cloud9 환경이 2GiB 이상의 메모리가 있는 Amazon EC2 인스턴스에 연결되어 있습니다.
- Java 개발과 연결된 파일로 작업하고 있습니다. AWS Cloud9은 `*.java`, `*.gradle`(Gradle 빌드 도구와 연결됨) 및 `pom.xml`(Apache Maven 빌드 도구와 연결됨) 파일 이름과 확장명을 확인합니다.
- 2020년 12월 11일 이후 생성된 AWS Cloud9 환경에서 작업하고 있습니다. 현재 이 날짜 이전에 생성된 개발 환경에서는 Java 생산성 기능을 사용할 수 없습니다.

이러한 조건이 충족되면 Java 코딩 및 디버깅을 위한 추가 생산성 기능을 활성화할 것인지 묻는 대화 상자가 표시됩니다. 활성화(Activate)를 선택하면 IDE의 기능을 사용할 수 있습니다.



> **Note**
>
> AWS Cloud9 환경을 생성할 때 시작되는 Amazon EC2 인스턴스에는 Amazon Coretto 11이 이미 설치되어 있습니다. Amazon Coretto는 무료로 사용할 수 있는 Open Java Development Kit(OpenJDK)의 프로덕션용 멀티플랫폼 배포판입니다. 즉, AWS Cloud9에서 Java 애플리케이션 개발 및 실행을 바로 시작할 수 있습니다.

AWS Cloud9 인터페이스를 사용하여 향상된 언어 및 디버깅 지원을 수동으로 활성화하고 비활성화할 수도 있습니다. 기본 설정(Preferences), Java 지원(Java Support), 향상된 Java 지원(Enhanced Java Support)을 선택합니다.



AWS Cloud9의 향상된 Java 개발 지원은 IDE에 대한 두 가지 확장을 통해 제공됩니다.

- Red Hat의 Java(TM) 언어 지원
- Java용 디버거

AWS Cloud9 인터페이스를 사용하면 이러한 확장의 성능을 사용자 지정하는 다양한 설정에 액세스할 수 있습니다. 확장 설정을 변경하려면 기본 설정(Preferences), Java 지원(Java Support)을 선택합니다.

이러한 설정에 대한 자세한 내용은 확장의 GitHub 리포지토리에서 설치된 버전의 ReadMe 페이지를 참조하세요.

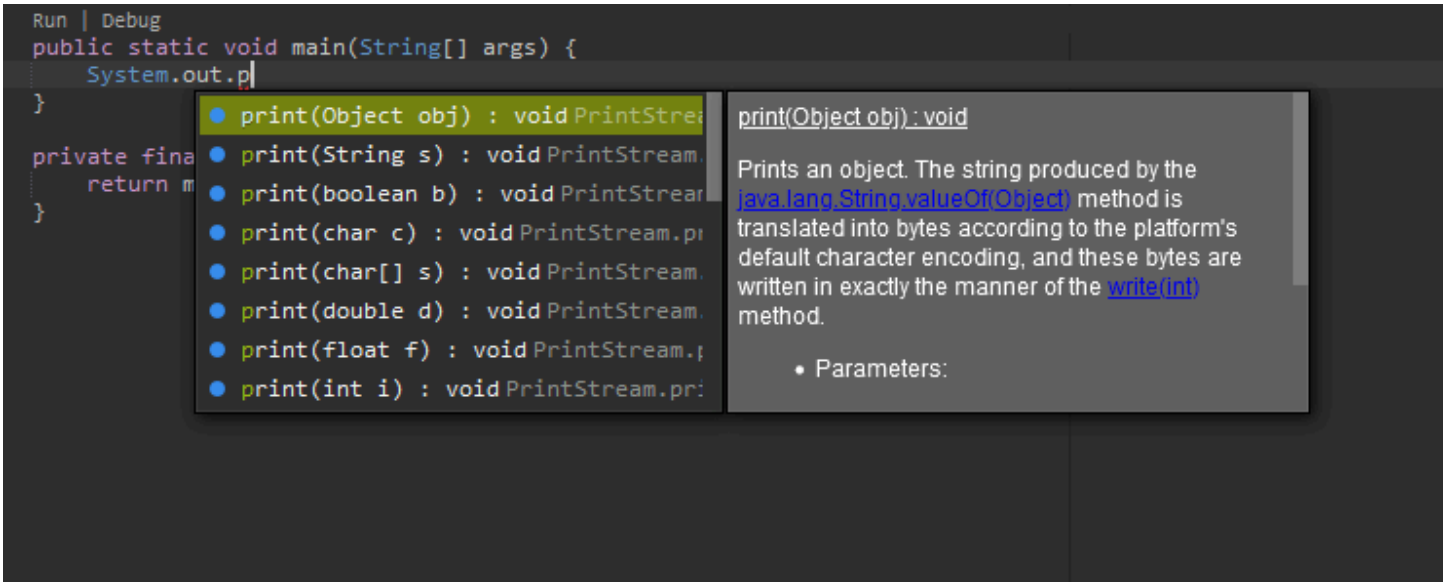
- Red Hat의 Java(TM) 언어 지원
- Java용 디버거

기능 하이라이트

향상된 Java 지원을 활성화한 후 다양한 생산성 향상 기능을 사용할 수 있습니다.

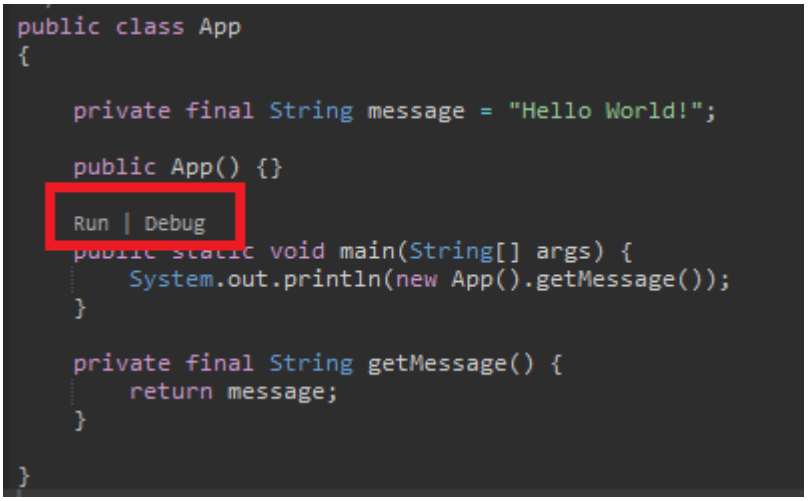
코드 완성

코드 완성 기능을 통해 편집기는 입력하는 코드를 기반으로 컨텍스트 인식 제안을 제공합니다. 예를 들어 객체 이름 뒤에 점(".") 연산자를 입력하면 편집기에 해당 객체에 사용할 수 있는 메서드나 속성이 표시됩니다.



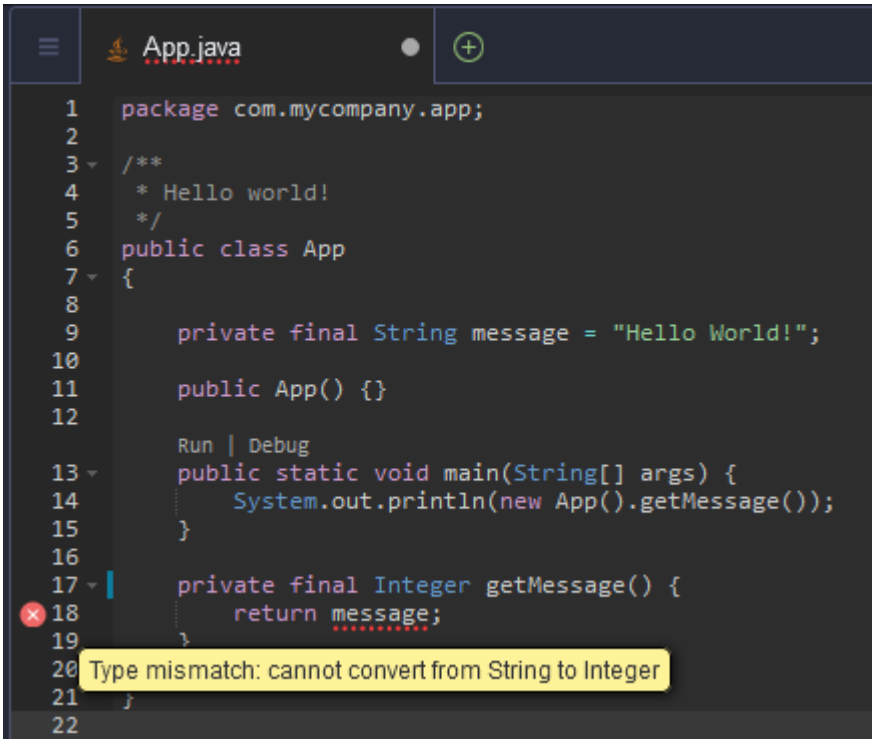
코드 렌즈

코드 렌즈를 사용하면 소스 코드에서 직접 컨텍스트별 작업에 액세스할 수 있습니다. Java 개발의 경우 코드 렌즈는 특정 메서드를 실행하고 디버그할 수 있도록 하여 단위 테스트를 용이하게 합니다.



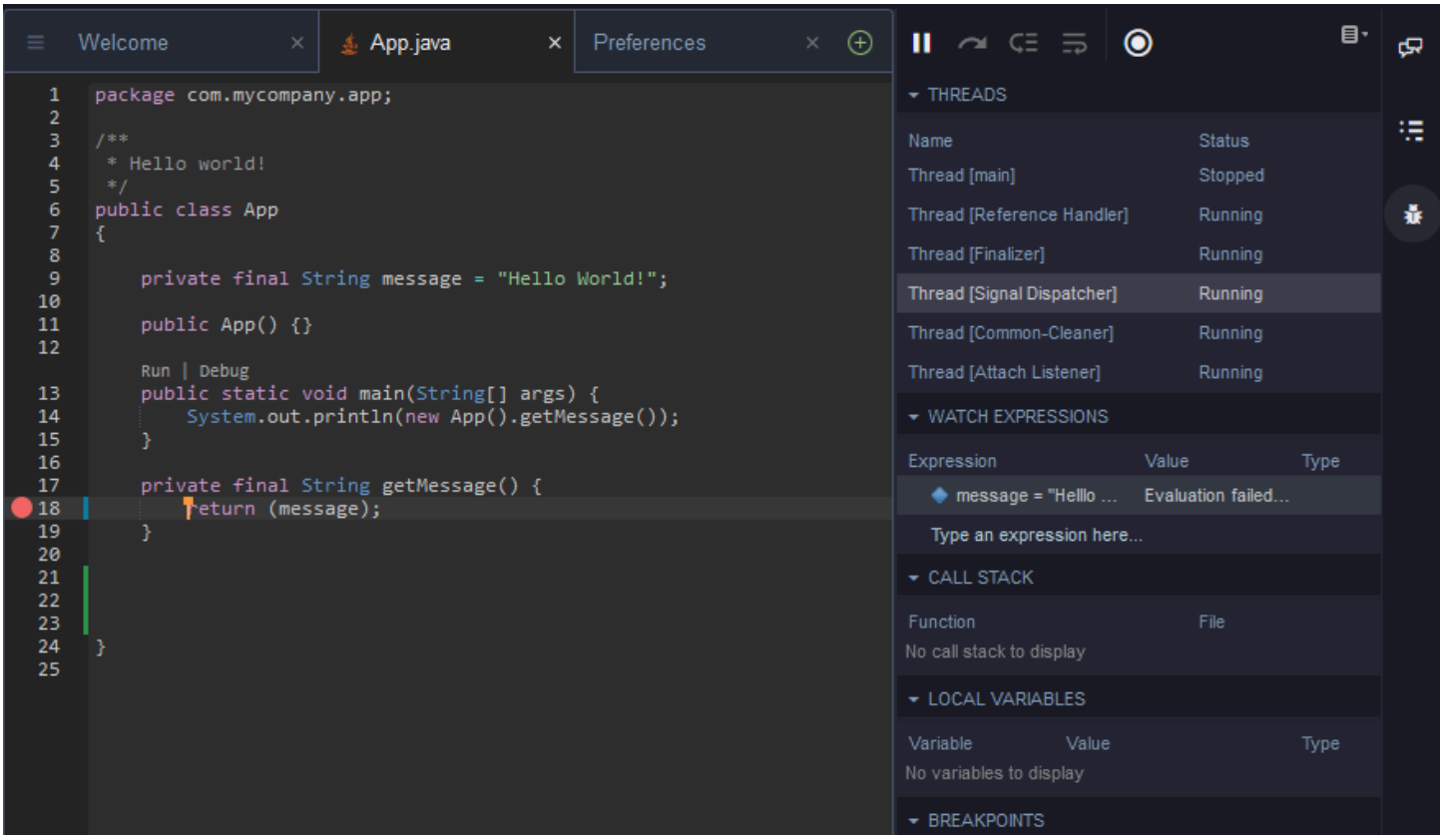
코드 linting

코드 linting은 코드를 빌드하기도 전에 편집기가 코드의 잠재적 오류를 강조 표시하는 방법을 설명합니다. 예를 들어, linting 도구는 초기화되지 않은 변수를 사용하려고 하거나 다른 유형을 예상하는 변수에 값을 할당하려고 하면 호출합니다.



디버깅 옵션

중단점과 조사식을 구현할 수 있습니다. 소스 코드에서 중단점을 설정하고 디버거 창을 표시하여 관련 조건을 정의합니다.

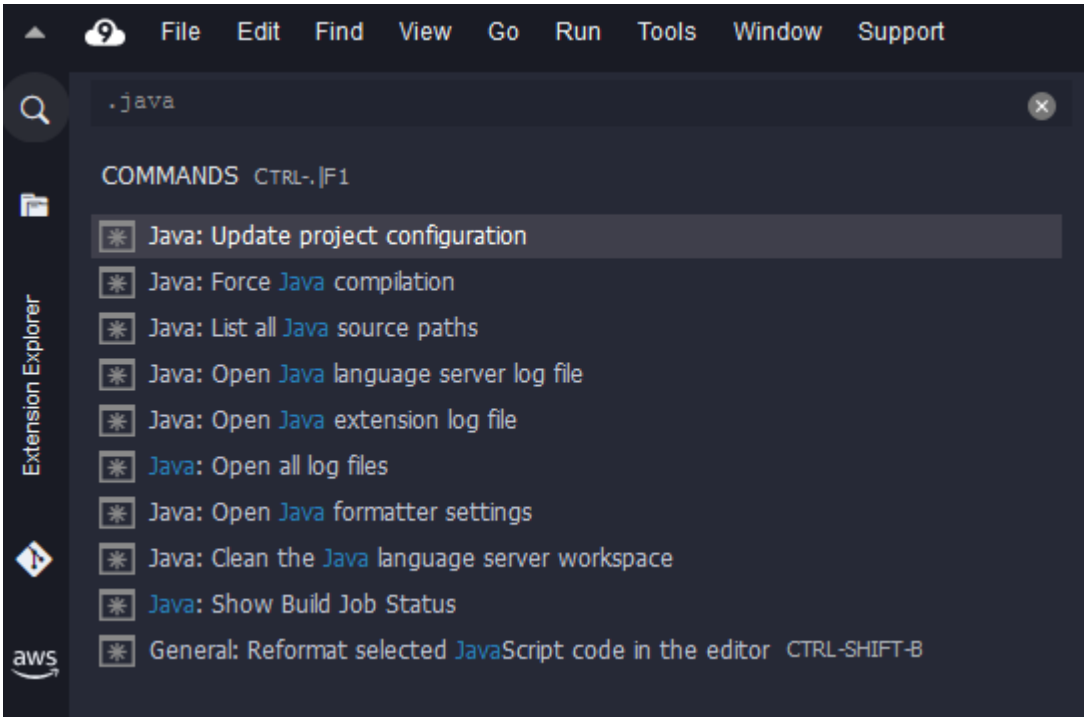


구성 파일을 사용하여 디버깅

또한 AWS Cloud9이 `launch.json` 및 `tasks.json` 구성 파일을 통해 지원하는 시작 구성 및 작업을 사용하여 디버깅 구성을 제어할 수도 있습니다. 시작 구성의 예와 사용 방법은 Java 디버그 구성을 참조하세요.

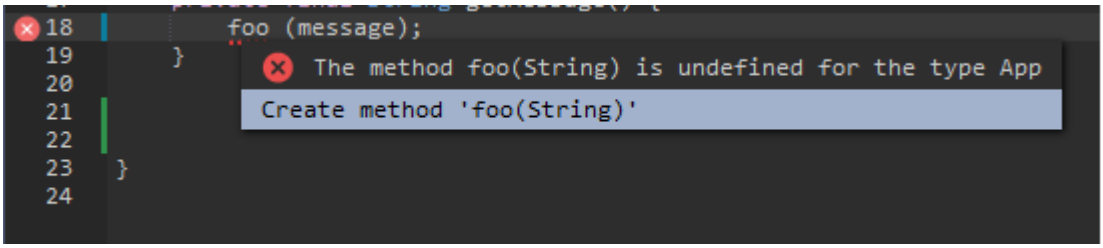
Java 명령

Ctrl+. 또는 F1 키를 눌러 AWS Cloud9 명령 패널에서 명령을 실행할 수 있습니다. 그런 다음 "java"를 입력하여 관련 명령을 필터링합니다.



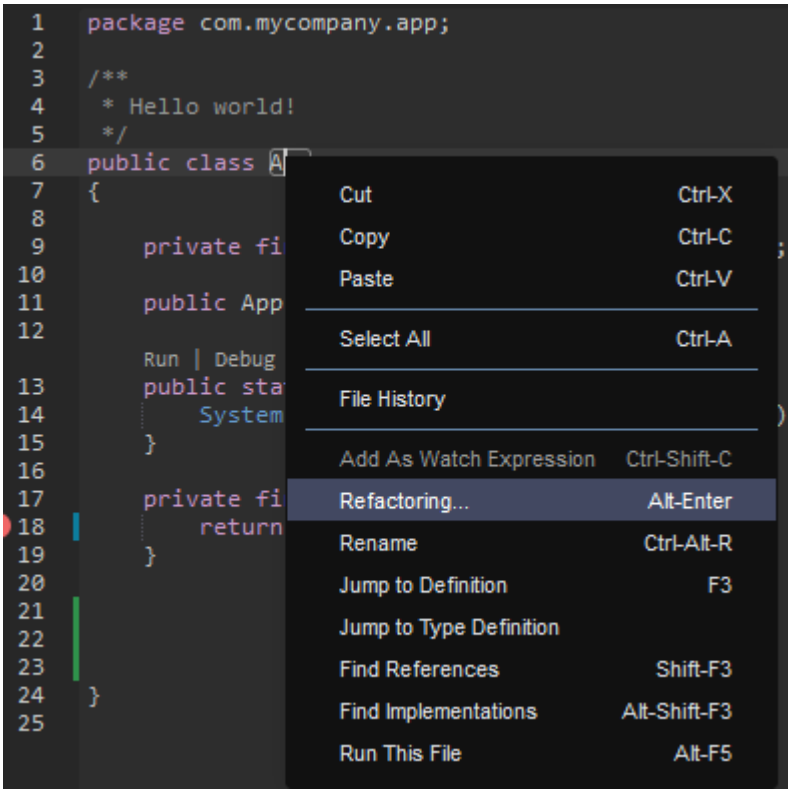
빠른 수정 사항

빠른 수정을 통해 누락된 요소에 대한 스텁을 생성하여 선언되지 않은 변수 또는 정의되지 않은 메서드를 사용하여 발생하는 오류를 해결할 수 있습니다.



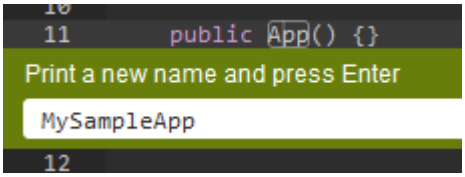
리팩터링

리팩터링을 동작을 변경하지 않고 코드를 재구성할 수 있습니다. 가져오기 구성 또는 생성자 생성과 같은 옵션에 액세스하려면 항목에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 리팩터링(Refactoring)을 선택합니다.



이름 바꾸기

이름 바꾸기는 단일 작업으로 코드에 나타나는 모든 위치에서 선택한 변수, 함수 및 클래스의 이름을 쉽게 수정할 수 있는 리팩터링 기능입니다. 이름을 변경하려면 항목에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 이름 바꾸기(Rename)를 선택합니다. 이름 바꾸기는 코드에서 이름의 모든 인스턴스에 영향을 줍니다.



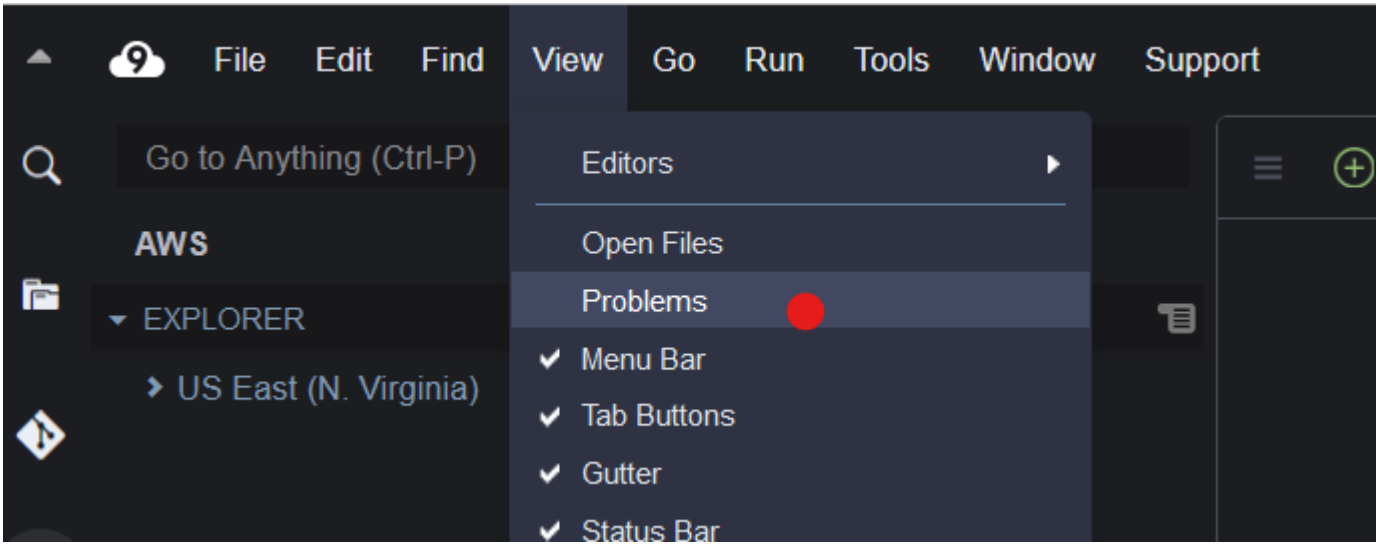
Java 개발을 위한 선택적 도구

향상된 Java 지원을 제공하는 확장에는 Gradle 및 Maven 자동화 도구를 프로젝트 개발에 통합할 수 있는 기능이 포함됩니다. 이러한 도구는 AWS Cloud9 개발 환경에 사전 설치되어 있지 않습니다. 이러한 선택적 빌드 도구를 설치하고 사용하는 방법에 대한 자세한 내용은 다음 리소스를 참조하세요.

- Gradle: 시작 가이드
- Maven: 5분 동안 알아보는 Maven의 모든 것

Java 확장 관련 문제 탭

AWS Cloud9 IDE의 문제 탭에서 AWS Cloud9 환경 내의 Java 프로젝트 관련 문제를 보고 해결할 수 있습니다. AWS Cloud9 IDE에서 문제 탭을 표시하려면 View(보기)를 선택하고 메뉴 모음에서 Problems(문제)를 선택합니다.



콘솔에서 + 아이콘을 선택하고 Open Problems(문제 열기)를 선택하여 문제 탭을 열 수도 있습니다. 탭에서 문제를 선택하면 해당 파일이 열리고 문제 세부 정보가 표시됩니다.

향상된 TypeScript 지원 및 기능

AWS Cloud9 IDE를 사용하면 언어 프로젝트로 TypeScript의 향상된 생산성 기능에 액세스할 수 있습니다. 언어 프로젝트는 AWS Cloud9 개발 환경에 대한 IDE의 관련 파일, 폴더 및 설정으로 이루어진 모음입니다.

IDE를 사용하여 사용자 환경에서 언어 프로젝트를 생성하려면 언어 프로젝트 생성 섹션을 참조하세요.

사용 가능한 프로젝트 생산성 기능

AWS Cloud9 IDE는 TypeScript에 대해 다음과 같은 프로젝트 생산성 기능을 제공합니다.

자동 완성

편집기에서 파일을 입력하면 해당 컨텍스트의 삽입점에 기호 리스트가 표시됩니다(해당 기호를 사용할 수 있는 경우).

삽입 지점의 목록에서 기호를 삽입하려면 기호가 아직 선택되어 있지 않은 경우 위쪽 화살표 또는 아래쪽 화살표 키를 사용하여 기호를 선택한 다음 Tab 키를 누릅니다.

Tab 키를 누르기 전에 선택한 기호에 대한 정보가 포함된 스크린 팁이 표시될 수 있습니다(해당 정보가 제공되는 경우).

기호를 삽입하지 않고 목록을 닫으려면 Esc 키를 누릅니다.

거터 아이콘

활성 파일의 거터에 아이콘이 표시될 수 있습니다. 이러한 아이콘은 코드를 실행하기 전에 코드의 경고 및 오류와 같은 가능한 문제를 강조 표시합니다.

문제에 대한 자세한 내용을 보려면 마우스 포인터로 해당 문제의 아이콘을 잠시 가리킵니다.

빠른 수정 사항

편집기의 활성 파일에서, 코딩 오류 및 경고에 대한 정보를 표시하고 해당 코드에 자동으로 적용할 수 있는 수정 사항을 표시할 수 있습니다. 오류 또는 경고 정보와 가능한 수정 사항을 표시하려면, 빨간색 점선 밑줄(오류의 경우) 또는 회색 점선 밑줄(경고의 경우)이 있는 코드 부분을 선택합니다. 아니면 빨간색 또는 회색 점선 밑줄이 있는 코드에 커서를 놓고 Option-Enter 키(macOS) 또는 Alt-Enter 키(Linux 또는 Windows)를 누릅니다. 제안된 수정 사항을 적용하려면 목록에서 수정 사항을 선택하거나 화살표 키를 사용하여 수정 사항을 선택하고 Enter 키를 누릅니다. 마우스 클릭으로 빠른 수정 사항을 선택하는 기능을 켜거나 끄려면 AWS Cloud9, [기본 설정(Preferences)], [사용자 설정(User Settings)], [언어(Language)], [힌트 및 경고(Hints & Warnings)], [클릭 시 사용 가능한 빠른 수정 사항 표시(Show Available Quick Fixes on Click)]를 선택합니다.

참조 찾기

편집기의 활성 파일에서, 해당 참조에 대한 액세스 권한이 IDE에 있는 경우 기호에 대한 모든 참조를 삽입점에 표시할 수 있습니다.

이렇게 하려면 기호 내의 아무 삽입점에서나 **Find References** 명령을 실행합니다. 예:

- 삽입점에서 마우스 오른쪽 버튼을 클릭한 다음 [참조 찾기(Find References)]를 선택합니다.
- 메뉴 모음에서 [이동, 참조 찾기(Go, Find References)]를 선택합니다.
- macOS, Windows 또는 Linux에 대해 기본적으로 Shift-F3 키를 누릅니다.

참조를 사용할 수 있는 경우 활성 파일의 위에, 해당 기호 옆에 창이 열립니다. 이 창에는 해당 기호가 참조되는 파일의 목록이 있습니다. 이 창에는 목록의 첫 번째 참조가 표시됩니다. 다른 참조를 표시하려면 목록에서 해당 참조를 선택합니다.

창을 닫으려면 닫기(X) 아이콘을 선택하거나 Esc 키를 누릅니다.

다음 조건에서는 **Find References** 명령을 사용할 수 없거나 명령이 정상적으로 작동하지 않을 수 있습니다.

- 활성 파일의 프로젝트에는 해당 기호에 대한 참조가 없습니다.
- IDE가 활성 파일의 프로젝트에서 해당 기호의 참조 중 일부 또는 전부를 찾을 수 없습니다.
- IDE가 활성 파일의 프로젝트에서 해당 기호가 참조되는 하나 이상의 위치에 액세스할 수 없습니다.

정의로 이동

편집기의 활성 파일에서, 해당 참조에 대한 액세스 권한이 IDE에 있는 경우 기호에서 해당 기호가 정의된 위치로 이동할 수 있습니다.

이렇게 하려면 기호 내의 아무 삽입점에서나 **Jump to Definition** 명령을 실행합니다. 예:

- 삽입점에서 마우스 오른쪽 버튼을 클릭한 다음 [정의로 이동(Jump to Definition]을 선택합니다.
- 메뉴 모음에서 [이동, 정의로 이동(Go, Jump to Definition]을 선택합니다.
- macOS, Windows 또는 Linux에 대해 기본적으로 F3 키를 누릅니다.

정의를 사용할 수 있는 경우 해당 정의가 별도의 파일에 있더라도 삽입점이 해당 정의로 전환됩니다.

다음 조건에서는 **Jump to Definition** 명령을 사용할 수 없거나 명령이 정상적으로 작동하지 않을 수 있습니다.

- 이 기호는 해당 언어의 기본 기호입니다.
- IDE가 활성 파일의 프로젝트에서 정의 위치를 찾을 수 없습니다.
- IDE가 활성 파일의 프로젝트에서 정의의 위치에 액세스할 수 없습니다.

기호로 이동

다음과 같이 프로젝트 내의 특정 기호로 이동할 수 있습니다.

1. 편집기에서 파일을 열어 프로젝트의 파일 중 하나를 활성화합니다. 파일이 이미 열려 있으면 편집기에서 탭을 선택하여 해당 파일을 활성 상태로 만듭니다.
2. **Go to Symbol** 명령을 실행합니다. 예:

   - [이동(Go)] 창 버튼(돋보기 아이콘)을 선택합니다. [바로 가기(Go to Anything)] 상자에 @을 입력한 다음 기호 입력을 시작합니다.

- 메뉴 모음에서 [이동, 기호로 이동(Go, Go To Symbol)]을 선택합니다. [이동(Go)] 창에서 @ 뒤에 기호 입력을 시작합니다.
- macOS 경우 기본적으로 `Command-2` 또는 `Command-Shift-0`를 누르고 Windows 또는 Linux의 경우 기본적으로 `Ctrl-Shift-0`를 누릅니다. [이동(Go)] 창에서 @ 뒤에 기호 입력을 시작합니다.

예를 들어 이름이 `toString`인 프로젝트에서 기호를 모두 찾으려면 `@toString` 입력을 시작합니다(또는 @가 이미 표시되어 있는 경우 @ 뒤에 `toString` 입력 시작).

3. 원하는 기호가 [기호(Symbols)] 목록에 표시되면 해당 기호를 클릭하여 선택합니다. 아니면 위쪽 화살표 또는 아래쪽 화살표 키를 사용하여 선택한 다음 `Enter` 키를 누릅니다. 그러면 삽입점이 해당 기호로 전환됩니다.

이동하려는 기호가 활성 파일의 프로젝트에 없는 경우 이 절차가 정상적으로 작동하지 않을 수 있습니다.

언어 프로젝트 생성

AWS Cloud9 IDE에서, 지원되는 프로젝트 생산성 기능과 함께 작동하는 언어 프로젝트를 만들려면 다음 절차를 따릅니다.

> **Note**
>
> 지원되는 프로젝트 생산성 기능은 언어 프로젝트의 일부인 파일에 사용하는 것이 좋습니다. 프로젝트의 일부가 아닌 파일에 일부 지원되는 프로젝트 생산성 기능을 사용할 수 있지만, 이러한 기능은 예기치 않은 결과를 보일 수 있습니다.
> 예를 들어 IDE를 사용하여 프로젝트의 일부가 아닌 환경의 루트 수준에 있는 파일 내에서 참조 및 정의를 검색할 수 있습니다. 그러면 IDE는 동일한 루트 수준의 파일에서만 검색할 수 있습니다. 이 경우 참조 또는 정의가 동일한 환경의 다른 언어 프로젝트에 실제로 존재하더라도 참조나 정의가 검색되지 않을 수 있습니다.

TypeScript 언어 프로젝트 생성

1. TypeScript가 환경에 설치되어 있는지 확인합니다. 자세한 내용은 AWS Cloud9용 TypeScript 자습서의 1단계: 필수 도구 설치 섹션을 참조하세요.
2. 환경의 IDE의 터미널 세션에서 프로젝트를 생성할 디렉터리로 전환합니다. 디렉터리가 없으면 새로 생성한 후 해당 디렉터리로 전환합니다. 예를 들어 다음 명령을 사용하여 환경의 루트(~/

environment)에 my-demo-project라는 디렉터리를 생성한 다음, 해당 디렉터리로 전환합니다.

```
mkdir ~/environment/my-demo-project
cd ~/environment/my-demo-project
```

3. 프로젝트를 생성하려는 디렉터리의 루트에서 **--init** 옵션을 사용하여 TypeScript 컴파일러를 실행합니다.

```
tsc --init
```

이 명령이 성공적으로 실행되면 TypeScript 컴파일러가 프로젝트 디렉터리의 루트에 tsconfig.json 파일을 생성합니다. 이 파일을 사용하여 TypeScript 컴파일러 옵션, 프로젝트에 포함하거나 제외할 특정 파일 등 다양한 프로젝트 설정을 정의할 수 있습니다.

tsconfig.json 파일에 대한 자세한 내용은 다음을 참조하세요.

- TypeScript 웹 사이트에서 tsconfig.json 개요를 참조하세요.
- json.schemastore.org 웹 사이트에서 tsconfig.json 스키마를 참조하세요.

AWS Cloud9 통합 개발 환경(IDE)의 메뉴 모음 명령 참조

다음 목록에는 AWS Cloud9 IDE의 기본 메뉴 모음 명령에 대한 설명이 나와 있습니다. 메뉴 모음이 보이지 않으면 IDE 상단 가장자리를 따라 있는 얇은 막대를 선택하면 표시됩니다.

- 미리 보기(Preview) 메뉴
- 기타 메뉴 모음 명령

AWS Cloud9 메뉴

명령	설명
환경설정	다음 중 하나를 수행하세요. • [기본 설정(Preferences)] 탭이 열려 있지 않은 경우 엽니다. • [기본 설정(Preferences)] 탭이 열려 있지만 활성화되지 않은 경우 활성화합니다. • [기본 설정(Preferences)] 탭을 클릭합니다(활성 상태인 경우). 프로젝트 설정 작업, 사용자 설정 작업, 키 바인딩 작업, 테마 작업 및 초기화 스크립트 작업을 참조하세요.
대시보드로 이동	별도의 웹 브라우저 탭에서 AWS Cloud9 콘솔을 엽니다. 환경 생성, 환경 열기, 환경 설정 변경 및 환경 삭제를 참조하세요.
시작 페이지	[시작(Welcome)] 탭을 클릭합니다.
프로젝트 설정 열기	현재 환경의 `project.settings` 파일을 엽니다. 프로젝트 설정 작업을 참조하세요.
사용자 설정 열기	현재 사용자의 `user.settings` 파일을 엽니다. 사용자 설정 작업을 참조하세요.
키맵 열기	현재 사용자의 `keybindings.settings` 파일을 엽니다. 키 바인딩 작업을 참조하세요.
초기화 스크립트 열기	현재 사용자의 `init.js` 파일을 엽니다. 초기화 스크립트 작업을 참조하세요.

명령	설명
스타일시트 열기	현재 사용자의 `styles.css` 파일을 엽니다. 테마 작업을 참조하세요.

파일(File) 메뉴

명령	설명
새 파일	새 파일을 만듭니다.
템플릿에서 새로 만들기	선택한 파일 템플릿을 기반으로 새 파일을 만듭니다.
Open	[탐색(Navigate)] 창을 표시하고 이 창으로 이동합니다.
Open Recent	선택한 파일을 엽니다.
Save	현재 파일을 저장합니다.
다른 이름으로 저장	현재 파일을 다른 파일 이름이나 위치 또는 두 가지 모두를 사용하여 저장합니다.
Save All(모두 저장)	저장되지 않은 파일을 모두 저장합니다.
저장된 상태로 되돌리기	마지막으로 저장한 이후 적용된 현재 파일의 변경 내용을 취소합니다.
모두 저장된 상태로 되돌리기	마지막으로 저장한 이후 저장되지 않은 모든 파일의 변경 내용을 취소합니다.
파일 수정 이력 표시	편집기에서 현재 파일의 변경 내용을 보고 관리합니다. 파일 수정 작업을 참조하세요.
로컬 파일 업로드	[파일 업로드(Upload Files)] 대화 상자를 표시합니다. 이 대화 상자를 통해 로컬 컴퓨터에서 환경으로 파일을 끌어올 수 있습니다.

명령	설명
프로젝트 다운로드	환경의 파일을 로컬 컴퓨터에 다운로드할수 있는 .zip 파일로 결합합니다.
줄 끝	Windows(캐리지 리턴과 줄 바꿈) 또는 Unix(줄 바꿈만) 줄 끝을 사용합니다.
파일 닫기	현재 파일을 닫습니다.
모든 파일 닫기	열려 있는 파일을 모두 닫습니다.

편집(Edit) 메뉴

명령	설명
Undo	마지막 작업을 실행 취소합니다.
Redo	마지막으로 실행 취소한 작업을 다시 실행합니다.
잘라내기	선택 영역을 클립보드로 이동합니다.
복사	선택 영역을 클립보드로 복사합니다.
붙여넣기	클립보드의 내용을 선택 포인트에 복사합니다.
키보드 모드	`Default`, `Vim`, `Emacs`, `Sublime` 등, 사용할 키 바인딩 집합입니다. 키 바인딩 작업을 참조하세요.
선택, 모두 선택	선택 가능한 내용을 모두 선택합니다.
선택, 줄 분할	현재 줄의 끝에 커서를 추가합니다.
선택, 단일 선택	이전 선택 내용을 모두 지웁니다.

명령	설명
선택, 다중 선택, 위에 커서 추가	활성 커서의 한 줄 위에 커서를 추가합니다. 커서가 이미 추가되어 있으면 그 위에 다른 커서를 추가합니다.
선택, 다중 선택, 아래에 커서 추가	활성 커서의 한 줄 아래에 커서를 추가합니다. 커서가 이미 추가되어 있으면 그 아래에 다른 커서를 추가합니다.
선택, 다중 선택, 활성 커서 위로 이동	활성 커서의 한 줄 위에 두 번째 커서를 추가합니다. 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 위로 이동합니다.
선택, 다중 선택, 활성 커서 아래로 이동	활성 커서의 한 줄 아래에 두 번째 커서를 추가합니다. 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 아래로 이동합니다.
선택, 다중 선택, 다음 일치하는 선택 항목 추가	선택 항목의 뒤에 일치하는 선택 항목을 더 포함합니다.
선택, 다중 선택, 이전 일치하는 선택 항목 추가	선택 항목의 앞에 일치하는 선택 항목을 더 포함합니다.
선택, 다중 선택, 선택 범위 병합	현재 줄의 끝에 커서를 추가합니다.
선택, 오른쪽 단어 선택	선택 영역에 커서 오른쪽의 다음 단어를 포함합니다.
선택, 왼쪽 단어 선택	선택 영역에 커서 왼쪽의 다음 단어를 포함합니다.
선택, 줄 끝까지 선택	커서에서 현재 줄의 끝까지 선택 영역에 포함합니다.
선택, 줄 시작까지 선택	현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.

명령	설명
선택, 문서 끝까지 선택	커서에서 현재 파일의 끝까지 선택 영역에 포함합니다.
선택, 문서 시작까지 선택	커서에서 현재 파일의 처음까지 선택 영역에 포함합니다.
줄, 들여쓰기	선택 영역을 한 탭 들여씁니다.
줄, 내어쓰기	선택 영역을 한 탭 내어씁니다.
줄, 줄 위로 이동	선택 영역을 한 줄 위로 이동합니다.
줄, 줄 아래로 이동	선택 영역을 한 줄 아래로 이동합니다.
줄, 줄 위로 복사	줄의 내용을 복사하고 복사한 내용을 한 줄 위에 붙여 넣습니다.
줄, 줄 아래로 복사	줄의 내용을 복사하고 복사한 내용을 한 줄 아래에 붙여 넣습니다.
줄, 줄 제거	현재 줄의 내용을 삭제합니다.
줄, 줄 끝까지 제거	커서에서 현재 줄의 끝까지 삭제합니다.
줄, 줄 시작까지 제거	커서에서 현재 줄의 시작까지 삭제합니다.
줄, 줄 나누기	커서의 내용을 줄의 끝에 새 줄로 이동합니다.
텍스트, 오른쪽 단어 제거	커서의 오른쪽에 있는 단어를 삭제합니다.
텍스트, 왼쪽 단어 제거	커서의 왼쪽에 있는 단어를 삭제합니다.
텍스트, 정렬	정렬이 잘못된 경우 모든 커서를 각 행의 활성 커서와 동일한 공간으로 이동합니다.
텍스트, 문자 바꾸기	선택 내용을 바꿉니다.
텍스트, 대문자로	선택 내용을 모두 대문자로 변경합니다.

명령	설명
텍스트, 소문자로	선택 내용을 모두 소문자로 변경합니다.
설명, 설명 토글	선택한 각 줄의 처음에 줄 설명 문자를 추가하거나, 줄 설명 문자가 있는 경우 제거합니다.
코드 접기, 접기 전환	코드를 접거나, 코드 접기를 제거합니다(있는 경우).
코드 접기, 펼치기	선택한 코드를 펼칩니다.
코드 접기, 기타 접기	현재 선택 범위를 제외하고 접을 수 있는 모든 요소를 접습니다.
코드 접기, 모두 접기	접을 수 있는 모든 요소를 접습니다.
코드 접기, 모두 펼치기	전체 파일의 코드 접기를 펼칩니다.
코드 서식 지정, 코드 서식 적용	선택한 JavaScript 코드를 다시 포맷합니다.
코드 서식 지정, 언어 및 서식 기본 설정 열기	[기본 설정(Preferences)] 탭의 [프로젝트 설정(Project Settings) 섹션을 열어 프로그래밍 언어 설정으로 이동합니다.

찾기(Find) 메뉴

자세한 내용은 텍스트 찾기 및 바꾸기를 참조하세요.

명령	설명
찾기	현재 문서의 찾기 및 바꾸기 모음을 표시하고 [찾기(Find)] 표현식을 사용합니다.
다음 찾기	마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 다음 항목으로 이동합니다.
이전 찾기	마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 이전 항목으로 이동합니다.

명령	설명
Replace	현재 문서의 찾기 및 바꾸기 모음을 표시하고 [바꾸기(Replace With)] 표현식을 사용합니다.
다음 바꾸기	현재 문서의 찾기 및 바꾸기 모음에서 [찾기(Find)] 및 [바꾸기(Replace With)]와 일치하는 다음 항목을 바꿉니다.
이전 바꾸기	현재 문서의 찾기 및 바꾸기 모음에서 [찾기(Find)] 및 [바꾸기(Replace With)]와 일치하는 이전 항목을 바꿉니다.
모두 바꾸기	현재 문서의 찾기 및 바꾸기 모음에서 [찾기(Find)] 및 [바꾸기(Replace With)]와 일치하는 모든 항목을 바꿉니다.
Find in Files(파일에서 찾기)	여러 파일의 찾기 및 바꾸기 모음을 표시합니다.

보기(View) 메뉴

명령	설명
편집기	선택한 편집기를 표시합니다.
파일 열기	[환경(Environment)] 창에 [파일 열기(Open Files) 목록을 표시하거나 표시된 경우 숨깁니다.
문제	터미널의 Problems(문제) 패널에서 환경 내 Java 프로젝트와 관련된 문제를 표시합니다. 문제를 선택하여 대상 파일을 열 수 있습니다.
메뉴 모음	메뉴 모음을 표시하거나 표시된 경우 숨깁니다.
탭 버튼	탭을 표시하거나 표시된 경우 숨깁니다.
거터	거터를 표시하거나 표시된 경우 숨깁니다.

명령	설명
상태 표시줄	상태 표시줄을 표시하거나 표시된 경우 숨깁니다.
콘솔	[콘솔(Console)] 창을 표시하거나 표시된 경우 숨깁니다.
레이아웃, 단일	단일 창을 표시합니다.
레이아웃, 수직 분할	상단과 하단으로 두 개의 창을 표시합니다.
레이아웃, 수평 분할	두 개의 창을 나란히 표시합니다.
레이아웃, 교차 분할	같은 크기의 네 개의 창을 표시합니다.
레이아웃, 1:2 분할	왼쪽에 하나의 창과 오른쪽에 두 개의 창을 표시합니다.
레이아웃, 2:1 분할	왼쪽에 두 개의 창과 오른쪽에 하나의 창을 표시합니다.
글꼴 크기, 글꼴 크기 늘리기	글꼴 크기를 늘립니다.
글꼴 크기, 글꼴 크기 줄이기	글꼴 크기를 줄입니다.
구문	현재 문서의 구문 유형을 표시합니다.
테마	IDE 테마 유형을 표시합니다.
줄 바꿈	현재 창의 가장자리로 단어를 줄 바꿈하거나 이미 줄 바꿈이 있는 경우 단어 줄 바꿈을 중지합니다.
인쇄 여백으로 줄 바꿈	현재 인쇄 여백의 가장자리로 단어를 줄 바꿈하거나 이미 줄 바꿈이 있는 경우 단어 줄 바꿈을 중지합니다.

이동(Go) 메뉴

명령	설명
바로 가기	[바로 가기(Go to Anything)] 모드로 [이동(Go)] 창을 표시합니다.
기호로 이동	[기호로 이동(Go to Symbol)] 모드로 [이동(Go)] 창을 표시합니다.
파일로 이동	[파일로 이동(Go to File)] 모드로 [이동(Go)] 창을 표시합니다.
명령으로 이동	[명령으로 이동(Go to Command)] 모드로 [이동(Go)] 창을 표시합니다.
줄로 이동	[줄로 이동(Go to Line)] 모드로 [이동(Go)] 창을 표시합니다.
다음 오류	다음 오류로 이동합니다.
이전 오류	이전 오류로 이동합니다.
오른쪽 단어	오른쪽으로 한 단어 이동합니다.
왼쪽 단어	왼쪽으로 한 단어 이동합니다.
줄 끝	현재 줄의 끝으로 이동합니다.
줄 시작	현재 줄의 처음으로 이동합니다.
정의로 이동	커서 위치에 있는 변수 또는 함수의 정의로 이동합니다.
쌍을 이루는 중괄호로 이동	현재 범위에서 쌍을 이루는 기호로 이동합니다.
선택 항목으로 스크롤	선택 항목을 더 잘 볼 수 있도록 스크롤합니다.

실행(Run) 메뉴

명령	설명
실행	현재 애플리케이션을 실행하거나 디버그합니다.
마지막 실행	마지막 실행 파일을 실행하거나 디버그합니다.
다음으로 실행	선택한 러너를 사용하여 실행하거나 디버그합니다. 빌더, 러너 및 디버거 작업을 참조하세요.
실행 내역	실행 내역을 봅니다.
실행 구성	실행 또는 디버그할 실행 구성을 선택하거나, 실행 구성을 생성 또는 관리합니다. 빌더, 러너 및 디버거 작업을 참조하세요.
중단 시 디버거 표시	실행 코드가 중단 점에 도달하면 [디버거(Debugger)] 창을 표시합니다.
빌드	현재 파일을 빌드합니다.
빌드 취소	현재 파일의 빌드를 중지합니다.
빌드 시스템	선택한 빌드 시스템을 사용하여 빌드합니다.
빌드 결과 표시	관련 빌드 결과를 표시합니다.
Automatically Build Supported Files(지원되는 파일 자동 빌드)	지원되는 파일을 자동으로 빌드합니다.
빌드 시 모두 저장	빌드 시에, 저장되지 않은 모든 관련 파일을 저장합니다.

도구(Tools) 메뉴

명령	설명
후행 공백 제거	줄 끝에 있는 공백을 자릅니다.
미리 보기, 파일 미리 보기	미리 보기 탭에서 현재 문서를 미리 봅니다.
미리 보기, 실행 중인 애플리케이션 미리 보기	현재 애플리케이션을 별도의 웹 브라우저 탭에서 미리 봅니다.
미리 보기, 미리 보기 URL 구	[기본 설정] 탭의 [프로젝트 설정(Project Settings)] 섹션에서 [실행 및 디버그, URL 미리 보기(Run & Debug, Preview URL)] 상자를 엽니다.
미리 보기, 활성 서버 표시	[프로세스 목록(Process List)] 대화 상자에 사용 가능한 활성 서버 주소의 목록을 표시합니다.
프로세스 목록	[프로세스 목록(Process List)] 대화 상자를 표시합니다.
자동 완성 표시	코드 완성 컨텍스트 메뉴를 표시합니다.
변수 이름 바꾸기	선택 항목의 이름 바꾸기 리팩터링을 시작합니다.
매크로 기록 토글	키 입력 기록을 시작하거나 이미 기록 중인 경우 중지합니다.
매크로 재생	이전에 기록한 키 입력을 재생합니다.

창(Window) 메뉴

명령	설명
Go	[이동(Go)] 창을 표시하거나 표시된 경우 숨깁니다.

명령	설명
새 터미널	새 [터미널(Terminal)] 탭을 엽니다.
새 직접 실행 창	새 [직접 실행(Immediate)] 탭을 엽니다.
공유	[이 환경 공유(Share this environment)] 대화 상자를 표시합니다.
설치 관리자	AWS Cloud9 설치 관리자 대화 상자를 표시합니다.
협업	[협업(Collaborate)] 창을 표시하거나 표시된 경우 숨깁니다.
개요	[개요(Outline)] 창을 표시하거나 표시된 경우 숨깁니다.
AWS 리소스	AWS 리소스 창을 표시하거나 표시된 경우 숨깁니다.
Environment	[환경(Environment)] 창을 표시하거나 표시된 경우 숨깁니다.
디버거	[디버거(Debugger)] 창을 표시하거나 표시된 경우 숨깁니다.
탐색, 오른쪽으로 탭	오른쪽으로 한 탭 이동합니다.
탐색, 왼쪽으로 탭	왼쪽으로 한 탭 이동합니다.
탐색, 기록의 다음 탭	다음 탭으로 이동합니다.
탐색, 기록의 이전 탭	이전 탭으로 이동합니다.
탐색, 탭을 오른쪽으로 이동	현재 탭을 오른쪽으로 이동합니다. 탭이 이미 맨 오른쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.

명령	설명
탐색, 탭을 왼쪽으로 이동	현재 탭을 왼쪽으로 이동합니다. 탭이 이미 맨 왼쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.
탐색, 탭을 위로 이동	현재 탭을 하나 위의 창으로 이동합니다. 탭이 이미 맨 위에 있는 경우 해당 위치에 분할 탭을 만듭니다.
탐색, 탭을 아래로 이동	현재 탭을 하나 아래의 창으로 이동합니다. 탭이 이미 맨 아래에 있는 경우 해당 위치에 분할 탭을 만듭니다.
탐색, 창을 오른쪽으로 이동	오른쪽으로 창을 하나 이동합니다.
탐색, 창을 왼쪽으로 이동	왼쪽으로 창을 하나 이동합니다.
탐색, 창을 위로 이동	위로 창을 하나 이동합니다.
탐색, 창을 아래로 이동	아래로 창을 하나 이동합니다.
탐색, 편집기와 터미널 간 전환	편집기와 [터미널(Terminal)] 탭 간을 전환합니다.
탐색, 기록의 다음 창	다음 창으로 이동합니다.
탐색, 기록의 이전 창	이전 창으로 이동합니다.
저장된 레이아웃, 저장	현재 레이아웃을 저장합니다. 나중에 이 레이아웃으로 전환하려면 [저장된 레이아웃, 레이아웃-ID(Saved Layouts, LAYOUT-ID)]를 선택합니다.
저장된 레이아웃, 저장 및 모두 닫기	현재 레이아웃을 저장한 다음 모든 탭과 창을 닫습니다.
저장된 레이아웃, 파일 트리에 저장된 레이아웃 표시	저장된 모든 레이아웃을 [환경(Environment)] 창에 표시합니다.
탭, 창 닫기	현재 창을 닫습니다.

명령	설명
탭, 모든 창에서 모든 탭 닫기	모든 창에서 열려 있는 탭을 모두 닫습니다.
탭, 현재 탭을 제외한 모든 탭 닫기	현재 창에서 현재 탭을 제외하고 열려 있는 탭을 모두 닫습니다.
탭, 두 행으로 창 분할	현재 창을 위쪽과 아래쪽의 두 창으로 나눕니다.
탭, 두 열로 창 분할	현재 창을 왼쪽과 오른쪽의 두 창으로 나눕니다.
사전 설정, 전체 IDE	전체 IDE 모드로 전환합니다.
사전 설정, 최소 편집기	최소 편집기 모드로 전환합니다.
사전 설정, Sublime 모드	Sublime 모드로 전환합니다.

지원(Support) 메뉴

명령	설명
시작 페이지	[시작(Welcome)] 탭을 클릭합니다.
지원 받기(커뮤니티)	AWS Cloud9 온라인 커뮤니티 웹 사이트를 별도의 웹 브라우저 탭에서 엽니다.
문서 읽기	AWS Cloud9 사용 설명서를 별도의 웹 브라우저 탭에서 엽니다.

미리 보기(Preview) 메뉴

명령	설명
미리 보기 파일	미리 보기 탭에서 현재 문서를 미리 봅니다.
실행 중인 애플리케이션 미리 보기	현재 애플리케이션을 별도의 웹 브라우저 탭에서 미리 봅니다.

명령	설명
미리 보기 URL 구성	[기본 설정] 탭의 [프로젝트 설정(Project Settings)] 섹션에서 [실행 및 디버그, URL 미리 보기(Run & Debug, Preview URL)] 상자를 엽니다.
활성 서버 표시	[프로세스 목록(Process List)] 대화 상자에 사용 가능한 활성 서버 주소의 목록을 표시합니다.

기타 메뉴 모음 명령

명령	설명
실행	현재 애플리케이션을 실행하거나 디버그합니다.
공유	[이 환경 공유(Share this environment)] 대화 상자를 엽니다.
[기본 설정(Preferences)](기어 아이콘)	[기본 설정(Preferences)] 탭을 엽니다.

AWS Cloud9 통합 개발 환경(IDE)에서 텍스트 찾기 및 바꾸기

AWS Cloud9 IDE에서 찾기 및 바꾸기 모음을 사용하여 한 파일 또는 여러 파일의 텍스트를 찾고 바꿀 수 있습니다.

- 한 파일에서 텍스트 찾기
- 한 파일에서 텍스트 바꾸기
- 여러 파일에서 텍스트 찾기
- 여러 파일에서 텍스트 바꾸기
- 찾기 및 바꾸기 옵션

한 파일에서 텍스트 찾기

1. 텍스트를 찾을 파일을 엽니다. 파일이 이미 열려 있으면 파일의 탭을 선택하여 파일을 활성 상태로 만듭니다.
2. 메뉴 모음에서 Find, Find(찾기, 찾기)를 선택합니다.
3. 찾기 및 바꾸기 모음의 Find(찾기)에 찾을 텍스트를 입력합니다.
4. 찾기 옵션을 추가로 지정하려면 찾기 및 바꾸기 옵션을 참조하십시오.
5. 일치하는 항목이 있으면 [찾기(Find)] 상자의 [0 중 0(0 of 0)]이 0이 아닌 숫자로 바뀝니다. 일치하는 항목이 있으면 편집기가 첫 번째 일치 항목으로 이동합니다. 일치 항목이 두 개 이상 있는 경우 다음 일치 항목으로 이동하려면, Find(찾기) 상자의 오른쪽 화살표를 선택하거나 메뉴 모음에서 Find, Find Next(찾기, 다음 찾기)를 선택합니다. 이전 일치 항목으로 이동하려면 Find(찾기) 상자의 왼쪽 화살표를 선택하거나 메뉴 모음의 Find, Find Previous(찾기, 이전 찾기)를 선택합니다.

한 파일에서 텍스트 바꾸기

1. 텍스트를 바꿀 파일을 엽니다. 파일이 이미 열려 있으면 파일의 탭을 선택하여 파일을 활성 상태로 만듭니다.
2. 메뉴 모음에서 Find, Replace(찾기, 바꾸기)를 선택합니다.
3. 찾기 및 바꾸기 모음의 Find(찾기)에 찾을 텍스트를 입력합니다.
4. Replace With(다음으로 바꾸기)에 Find(찾기)에서 텍스트를 바꿀 텍스트를 입력합니다.
5. 찾기 및 바꾸기 옵션을 추가로 지정하려면 찾기 및 바꾸기 옵션을 참조하십시오.
6. 일치하는 항목이 있으면 [찾기(Find)] 상자의 [0 중 0(0 of 0)]이 0이 아닌 숫자로 바뀝니다. 일치하는 항목이 있으면 편집기가 첫 번째 일치 항목으로 이동합니다. 일치 항목이 두 개 이상 있는 경우 다음 일치 항목으로 이동하려면, Find(찾기) 상자의 오른쪽 화살표를 선택하거나 메뉴 모음에서 Find, Find Next(찾기, 다음 찾기)를 선택합니다. 이전 일치 항목으로 이동하려면 Find(찾기) 상자의 왼쪽 화살표를 선택하거나 메뉴 모음의 Find, Find Previous(찾기, 이전 찾기)를 선택합니다.
7. 현재 일치 항목을 Replace With(다음으로 바꾸기)의 텍스트로 바꾼 후 다음 일치 항목으로 이동하려면 바꾸기를 선택합니다. 모든 일치 항목을 Replace With(다음으로 바꾸기)의 텍스트로 바꾸려면 Replace All(모두 바꾸기)을 선택합니다.

여러 파일에서 텍스트 찾기

1. 메뉴 모음에서 Find, Find in Files(찾기, 파일에서 찾기)를 선택합니다.
2. 찾기 및 바꾸기 모음의 Find(찾기)에 찾을 텍스트를 입력합니다.

3. 찾기 옵션을 추가로 지정하려면 찾기 및 바꾸기 옵션을 참조하십시오.
4. Find(찾기) 버튼의 오른쪽 상자(`*.*, -.*`가 있는 상자)에 찾기에 포함시키거나 제외시킬 파일 집합을 입력합니다. 예:
   - 비워 둠, `*` 또는 `*.*`: 모든 파일을 찾습니다.
   - `my-file.txt`: 이름이 `my-file.txt`인 파일만 찾습니다.
   - `my*`: 파일 이름이 `my`로 시작하는 파일만 찾습니다.
   - `my*.txt`: 파일 이름이 `my`로 시작하고 파일 확장명이 `.txt`인 파일만 찾습니다.
   - `my*.htm*`: 파일 이름이 `my`로 시작하고 파일 확장명이 `.htm`으로 시작하는 모든 파일을 찾습니다.
   - `my*.htm, my*.html`: 파일 이름이 `my`로 시작하고 파일 확장명이 `.htm` 또는 `.html`인 모든 파일을 찾습니다.
   - `-my-file.txt`: 이름이 `my-file.txt`인 파일을 검색하지 않습니다.
   - `-my*`: `my`로 시작하는 파일을 검색하지 않습니다.
   - `-my*.htm*`: 파일 이름이 `my`로 시작하고 파일 확장명이 `.htm`으로 시작하는 파일을 검색하지 않습니다.
   - `my*.htm*, -my*.html`: 파일 이름이 `my`로 시작하고 파일 확장명이 `.htm`으로 시작하는 모든 파일을 검색합니다. 그러나 파일 이름이 `my`로 시작하고 파일 확장명이 `.html`인 파일을 검색하지 않습니다.
5. 위 상자 옆의 드롭다운 목록에서 다음 중 하나를 선택하여 찾기를 특정 위치로만 추가로 제한합니다.
   - 환경: 환경 창의 파일만 찾습니다.
   - 프로젝트(.gitignore'd 제외)(Project (excludes .gitignore'd)): `.gitignore` 파일이 있는 경우 환경에서 `.gitignore` 파일에 나열된 파일 또는 파일 유형을 제외한 환경의 모든 파일을 찾습니다.
   - Selection(선택):: 환경 창에서 현재 선택한 파일만 찾습니다.

     > **Note**
     >
     > 찾기를 폴더 하나로만 추가로 제한하려면 환경 창에서 폴더를 선택한 후 Selection(선택)을 선택합니다. 또는 환경 창에서 해당 폴더를 마우스 오른쪽 버튼으로 클릭한 후 컨텍스트 메뉴에서 Search In This Folder(이 폴더에서 검색)를 선택합니다.

   - 즐겨찾기(Favorites): [환경(Environment)] 창의 [즐겨찾기(Favorites)] 목록에 있는 파일만 찾습니다.
   - Active File(활성 파일): 활성 파일만 찾습니다.

- 파일 열기(Open Files): [환경(Environment)] 창의 [파일 열기(Open Files)] 목록에 있는 파일만 찾습니다.

6. Find(찾기)를 선택합니다.
7. 일치 항목이 포함된 파일로 이동하려면 검색 결과 탭에서 해당 파일 이름을 두 번 클릭합니다. 특정 일치 항목으로 이동하려면 검색 결과 탭에서 해당 일치 항목을 두 번 클릭합니다.

여러 파일에서 텍스트 바꾸기

1. 메뉴 모음에서 Find, Find in Files(찾기, 파일에서 찾기)를 선택합니다.
2. 찾기 및 바꾸기 모음의 Find(찾기)에 찾을 텍스트를 입력합니다.
3. 찾기 옵션을 추가로 지정하려면 찾기 및 바꾸기 옵션을 참조하십시오.
4. Find(찾기) 버튼의 오른쪽 상자(`*.*, -.*`가 있는 상자)에 찾기에 포함시키거나 제외시킬 파일 집합을 입력합니다. 예:
   - 비워 둠, * 또는 `*.*`: 모든 파일
   - `my-file.txt`: 이름이 `my-file.txt`인 파일만
   - `my*`: 파일 이름이 `my`로 시작하는 파일만
   - `my*.txt`: 파일 이름이 `my`로 시작하고 파일 확장명이 `.txt`인 파일만
   - `my*.htm*`: 파일 이름이 `my`로 시작하고 파일 확장명이 `.htm`으로 시작하는 모든 파일
   - `my*.htm, my*.html`: 파일 이름이 `my`로 시작하고 파일 확장명이 `.htm` 또는 `.html`인 모든 파일
   - `-my-file.txt`: 이름이 `my-file.txt`인 파일을 검색하지 않습니다.
   - `-my*`: `my`로 시작하는 파일을 검색하지 않습니다.
   - `-my*.htm*`: 파일 이름이 `my`로 시작하고 파일 확장명이 `.htm`으로 시작하는 파일을 검색하지 않습니다.
   - `my*.htm*, -my*.html`: 파일 이름이 `my`로 시작하고 파일 확장명이 `.htm`으로 시작하는 모든 파일을 검색합니다. 그러나 파일 이름이 `my`로 시작하고 파일 확장명이 `.html`인 파일을 검색하지 않습니다.
5. 위 상자 옆의 드롭다운 목록에서 다음 중 하나를 선택하여 찾기를 특정 위치로만 추가로 제한합니다.
   - 환경: 환경 창의 파일만
   - 프로젝트(.gitignore'd 제외)(Project (excludes .gitignore'd)): `.gitignore` 파일이 있는 경우 환경에서 `.gitignore` 파일에 나열된 파일 또는 파일 유형을 제외한 환경의 모든 파일을 찾습니다.

- Selection: /(선택: /): 현재 선택한 파일만
- 즐겨찾기(Favorites): [환경(Environment)] 창의 [즐겨찾기(Favorites)] 목록에 있는 파일만 찾습니다.
- Active File(활성 파일): 활성 파일만
- 파일 열기(Open Files): [환경(Environment)] 창의 [파일 열기(Open Files)] 목록에 있는 파일만 찾습니다.

6. Replace With(다음으로 바꾸기)에 Find(찾기)를 바꿀 텍스트를 입력합니다.
7. 바꾸기를 선택합니다.

> **Note**
>
> 범위 내의 모든 파일에서 바꾸기 작업이 즉시 수행됩니다. 이 작업은 쉽게 실행 취소할 수 없습니다. 바꾸기 작업을 시작하기 전에 변경될 항목을 보려면 찾기를 선택합니다.

8. 바뀐 항목이 포함된 파일로 이동하려면 검색 결과 탭에서 해당 파일 이름을 두 번 클릭합니다. 특정 바뀐 항목으로 이동하려면 검색 결과 창에서 해당 바뀐 항목을 두 번 클릭합니다.

찾기 및 바꾸기 옵션

찾기 및 바꾸기 모음에서 다음 버튼을 선택하여 찾기 및 바꾸기 작업을 수정합니다.

- 정규식: Find(찾기) 또는 Find in Files(파일에서 찾기)에 지정된 정규식과 일치하는 텍스트를 찾습니다. Mozilla 개발자 네트워크의 JavaScript 정규식 주제에서 정규식 패턴 작성을 참조하세요.

- Match Case(대/소문자 구분): Find(찾기) 또는 Find in Files(파일에서 찾기)에 지정된 대/소문자와 일치하는 텍스트를 찾습니다.
- Whole Words(단어 단위로): 표준 단어 문자 규칙을 사용하여 Find(찾기) 또는 Find in Files(파일에서 찾기)에서 텍스트를 찾습니다.
- Wrap Around(순환): 파일이 하나인 경우, 다음 또는 이전 일치 항목으로 이동할 때 파일의 끝이나 시작에서 중지하지 않습니다.
- Search Selection(선택 영역 검색): 파일이 하나인 경우, 선택 영역에서만 찾습니다.
- 콘솔에 표시(Show in Console): 여러 파일에 대해, 활성 창이 아니라 콘솔의 [검색 결과(Search Results)] 탭을 표시합니다.
- Preserve Case(대/소문자 보존): 파일이 하나인 경우에 한해 텍스트를 바꿀 때 해당하는 경우 대/소문자를 보존합니다.

AWS Cloud9 통합 개발 환경(IDE)에서 파일 미리 보기

AWS Cloud9 IDE를 사용하여 AWS Cloud9 개발 환경의 파일을 IDE 내에서 미리 볼 수 있습니다.

미리 볼 파일 열기

AWS Cloud9 IDE에서 다음 중 하나를 선택하여 환경 내에서 파일 미리 보기 탭을 열 수 있습니다.

- 환경 창에서 미리 보려고 하는 파일을 마우스 오른쪽 버튼으로 클릭한 다음 미리 보기를 선택합니다.

> **Note**
>
> 이 방법을 사용하여 파일을 미리 볼 수 있지만 다음 파일 확장명을 가진 파일에서 미리 보기가 가장 잘 작동합니다.
>
> - `.htm`
> - `.html`

  - `.pdf`
  - `.svg`
  - `.xhtml`
  - 마크다운 형식의 콘텐츠가 포함된 모든 파일.

- 다음 파일 확장명 중 하나를 사용하는 파일을 엽니다.
  - `.pdf`
  - `.svg`
- 미리 보려고 하는 파일이 이미 열려 있고 활성화된 상태로, 메뉴 모음에서 [미리 보기, FILE_NAME 파일 미리 보기(Preview, Preview File FILE_NAME)]를 선택합니다. 또는 [도구, 미리 보기, FILE_NAME 파일 미리 보기(Tools, Preview, Preview File FILE_NAME)]를 선택합니다. 여기서 FILE_NAME은 미리 보려고 하는 파일의 이름입니다.

  > **Note**
  >
  > 이러한 명령은 다음 파일 형식에서만 작동합니다.
  >
  > - `.htm`
  > - `.html`
  > - `.markdown`
  > - `.md`
  > - `.pdf`
  > - `.svg`
  > - `.txt`: 파일의 콘텐츠가 마크다운 형식인 경우 미리 보기가 가장 잘 작동합니다.
  > - `.xhtml`: 파일에 내용 프레젠테이션 정보가 포함되거나 참조되는 경우 미리 보기가 가장 잘 작동합니다.

> **Note**
>
> 파일 미리 보기 탭의 [미리 보기 설정(Preview Settings)] 메뉴는 현재 작동하지 않으며 해당 메뉴 명령을 선택해도 아무런 효과가 없습니다.

파일 미리 보기 다시 로드

파일 미리 보기 탭에서 [새로 고침(Refresh)] 버튼(원형 화살표)을 선택합니다.

파일 미리 보기 유형 변경

파일 미리 보기 탭의 미리 보기 유형 목록에서 다음 중 하나를 선택합니다.

- 브라우저(Browser): 다음 파일 형식에 대해서만 웹 브라우저 형식으로 파일을 미리 봅니다.
  - `.htm`
  - `.html`
  - `.pdf`
  - `.svg`
  - `.xhtml`: 파일에 내용 프레젠테이션 정보가 포함되거나 참조되는 경우 미리 보기가 가장 잘 작동합니다.
- 원시 콘텐츠(UTF-8)(Raw Content (UTF-8)): 파일의 원래 콘텐츠를 유니코드 변환 형식 8비트(UTF-8) 형식으로 미리 봅니다. 일부 파일 형식의 경우 예기치 않은 콘텐츠가 표시될 수 있습니다.
- 마크다운(Markdown): 마크다운 형식이 포함된 파일을 미리 봅니다. 다른 파일 형식을 미리 보려고 시도할 경우 예기치 않은 콘텐츠가 표시될 수 있습니다.

별도의 웹 브라우저 탭에서 파일 미리 보기 열기

파일 미리 보기 탭에서 [새 창으로 팝업(Pop Out Into New Window)]을 선택합니다.

다른 파일 미리 보기로 전환

파일 미리 보기 탭에서 주소 표시줄에 다른 파일 경로에 대한 경로를 입력합니다. 주소 표시줄은 [새로 고침(Refresh)] 버튼과 미리 보기 유형 목록 사이에 있습니다.

AWS Cloud9 통합 개발 환경 (IDE) 에서 실행 중인 응용 프로그램 미리 보기

IDE를 사용하여 AWS Cloud9 IDE 내에서 실행 중인 응용 프로그램을 미리 볼 수 있습니다.

주제

- 애플리케이션 실행

- 실행 중인 애플리케이션 미리 보기
- 애플리케이션 미리 보기 다시 로드
- 애플리케이션 미리 보기 유형 변경
- 별도의 웹 브라우저 탭에서 애플리케이션 미리 보기 열기
- 다른 미리 보기로 전환
- 인터넷을 통해 실행 중인 애플리케이션 공유

애플리케이션 실행

IDE 내에서 응용 프로그램을 미리 보려면 먼저 응용 프로그램이 AWS Cloud9 개발 환경에서 실행되고 있어야 합니다. 다음 포트를 통해 HTTP를 사용해야 합니다.

- `8080`
- `8081`
- `8082`

위의 모든 포트는 `127.0.0.1` `localhost` 또는 `0.0.0.0`의 IP 주소를 사용해야 합니다.

> **Note**
>
> IP 주소가 `127.0.0.1`, `localhost` 또는 `0.0.0.0`인 포트 `8080`, `8081` 또는 `8082`를 통해 HTTP를 사용하여 애플리케이션을 실행할 필요가 없습니다. 단, 이 경우 IDE 내에서 실행 중인 애플리케이션을 미리 볼 수 없습니다.

> **Note**
>
> 미리 보기 애플리케이션은 IDE 내에서 실행되며 iframe 요소 내에 로드됩니다. 일부 애플리케이션 서버는 기본적으로 X-Frame-Options 헤더와 같은 iframe 요소에서 오는 요청을 차단할 수 있습니다. 미리 보기 애플리케이션이 미리 보기 탭에 표시되지 않는 경우 애플리케이션 서버가 iframe에 콘텐츠를 표시하는 것을 금지하지 않는지 확인하세요.

특정 포트와 IP 주소에서 애플리케이션을 실행하는 코드를 작성하려면 애플리케이션 문서를 참조하세요.

애플리케이션을 실행하려면 코드 실행을 참조하세요.

이 동작을 테스트하려면 환경의 `server.js` 루트에 이름이 지정된 파일에 다음 JavaScript 코드를 추가하세요. 이 코드는 Node.js라는 파일을 사용하여 서버를 실행합니다.

> **Note**
>
> 다음 예에서 `text/html`은 반환된 콘텐츠의 `Content-Type`입니다. 콘텐츠를 다른 형식으로 반환하려면 다른 `Content-Type`을 지정합니다. 예를 들어, CSS 파일 형식으로 `text/css`를 지정할 수 있습니다.

```
var http = require('http');
var fs = require('fs');
var url = require('url');

http.createServer( function (request, response) {
  var pathname = url.parse(request.url).pathname;
  console.log("Trying to find '" + pathname.substr(1) + "'...");

  fs.readFile(pathname.substr(1), function (err, data) {
    if (err) {
      response.writeHead(404, {'Content-Type': 'text/html'});
      response.write("ERROR: Cannot find '" + pathname.substr(1) + "'.");
      console.log("ERROR: Cannot find '" + pathname.substr(1) + "'.");
    } else {
      console.log("Found '" + pathname.substr(1) + "'.");
      response.writeHead(200, {'Content-Type': 'text/html'});
      response.write(data.toString());
    }
    response.end();
  });
}).listen(8080, 'localhost'); // Or 8081 or 8082 instead of 8080. Or '127.0.0.1'
 instead of 'localhost'.
```

또는 환경의 루트에 있는 `server.py`라는 이름의 파일에 다음 Python 코드를 추가 할 수 있습니다. 다음 예제에서는 Python을 사용하여 서버를 실행합니다.

```
import os
import http.server
import socketserver
```

```
ip = 'localhost' # Or '127.0.0.1' instead of 'localhost'.
port = '8080' # Or '8081' or '8082' instead of '8080'.
Handler = http.server.SimpleHTTPRequestHandler
httpd = socketserver.TCPServer((ip, int(port)), Handler)
httpd.serve_forever()
```

환경의 루트에서 `index.html`이라는 파일에 다음 HTML 코드를 추가합니다.

```
<html>
  <head>
    <title>Hello Home Page</title>
  </head>
  <body>
    <p style="font-family:Arial;color:blue">Hello, World!</p>
  </body>
</html>
```

애플리케이션 미리 보기 탭에서 이 파일의 HTML 출력을 보려면 Node.js의 경우 `server.js`, Python의 경우 `server.py` 파일을 실행합니다. 그러고 나서 다음 섹션의 단계에 따라 미리 봅니다. 애플리케이션 미리 보기 탭에서 URL의 끝부분에 `/index.html`을 추가한 다음 `Enter` 키를 누릅니다.

실행 중인 애플리케이션 미리 보기

애플리케이션을 미리 보기 전에 다음 사항을 확인하세요.

- 애플리케이션은 포트 `8080`, `8081` 또는 `8082`를 통해 HTTP 프로토콜을 사용하여 실행됩니다.
- 해당 환경의 애플리케이션 IP 주소는 `127.0.0.1`, `localhost` 또는 `0.0.0.0`입니다.
- 애플리케이션 코드 파일은 AWS Cloud9 IDE에서 열려 있고 활성화되어 있습니다.

이러한 세부 정보를 모두 확인한 후 메뉴에서 다음 옵션 중 하나를 선택합니다.

- 미리 보기, 실행 중인 애플리케이션 미리 보기
- 도구, 미리 보기, 실행 중인 애플리케이션 미리 보기

이러한 옵션 중 하나를 선택하면 환경 내에서 애플리케이션 미리 보기 탭이 열리고 애플리케이션의 출력이 탭에 표시됩니다.

> **Note**
>
> 애플리케이션 미리 보기 탭에 오류가 표시되거나 비어 있으면 애플리케이션 미리 보기 탭에 오류가 표시되거나 이 탭이 비어 있음의 문제 해결 단계를 따릅니다. 애플리케이션이나 파일을 미리 볼 때 "브라우저에 타사 쿠키가 비활성화되어 있으므로 미리 보기 기능이 비활성화되었습니다."라는 알림이 표시되면 애플리케이션 미리 보기 또는 파일 미리 보기 알림: '서드 파티 쿠키가 사용 중지됨(Third-party cookies disabled)'의 문제 해결 단계를 따르세요.

> **Note**
>
> 애플리케이션이 아직 실행되고 있지 않으면 애플리케이션 미리 보기 탭에 오류가 표시됩니다. 이 문제를 해결하려면 애플리케이션을 실행하거나 다시 시작한 다음 메뉴 모음 명령을 다시 선택합니다.
>
> 예를 들어, 언급된 어떤 포트나 IP에서도 애플리케이션을 실행할 수 없습니다. 또는 이러한 포트 중 하나 이상에서 애플리케이션을 동시에 실행해야 합니다. 예를 들어, 애플리케이션을 포트 `8080`과 `3000`에서 동시에 실행해야 합니다. 이 경우 애플리케이션 미리 보기 탭에 오류가 표시되거나 비어 있을 수 있습니다. 이는 환경 내의 애플리케이션 미리 보기 탭이 앞의 포트와 IP에서만 작동하고 한 번에 하나의 포트에서만 작동하기 때문입니다.
>
> 애플리케이션 미리 보기 탭의 URL을 다른 사용자와 공유하지 않는 것이 좋습니다. (URL 형식은 다음과 같습니다`https://12a34567b8cd9012345ef67abcd890e1.vfs.cloud9.us-east-2.amazonaws.com/`. 이 형식에서 `12a34567b8cd9012345ef67abcd890e1` 는 환경에 AWS Cloud9 할당되는 ID입니다. `us-east-2`환경의 ID입니다.) AWS 리전 이 URL은 환경의 IDE가 열려 있고 애플리케이션이 동일한 웹 브라우저에서 실행 중인 경우에만 작동합니다. IDE의 `127.0.0.1` IP를 방문하거나 `0.0.0.0` IDE의 응용 프로그램 미리 보기 탭이나 IDE 외부의 별도 웹 브라우저 탭을 사용하여 방문하려고 하면 AWS Cloud9 IDE는 기본적으로 환경에 연결된 인스턴스나 자체 서버 대신 로컬 컴퓨터로 이동하려고 시도합니다. `localhost`

IDE 외부에서 실행 중인 애플리케이션의 미리 보기를 다른 사용자에게 제공하는 방법에 대한 지침은 인터넷을 통해 실행 중인 애플리케이션 공유 섹션을 참조하세요.

애플리케이션 미리 보기 다시 로드

애플리케이션 미리 보기 탭에서 [새로 고침(Refresh)] 버튼(원형 화살표)을 선택합니다.

> **Note**
>
> 이 명령은 서버를 다시 시작하지 않습니다. 단지 애플리케이션 미리 보기 탭의 내용을 새로 고칩니다.

애플리케이션 미리 보기 유형 변경

애플리케이션 미리 보기 탭의 미리 보기 유형 목록에서 다음 중 하나를 선택합니다.

- [브라우저(Browser)]: 출력을 웹 브라우저 형식으로 미리 봅니다.
- [원시 컨텐츠(UTF-8)(Raw Content (UTF-8))]: 가능한 경우 유니코드 변환 형식 8비트(UTF-8) 형식으로 출력을 미리 봅니다.
- Markdown(마크다운): 가능한 경우 마크다운 형식으로 출력을 미리 봅니다.

별도의 웹 브라우저 탭에서 애플리케이션 미리 보기 열기

애플리케이션 미리 보기 탭에서 [새 창으로 팝업(Pop Out Into New Window)]을 선택합니다.

> **Note**
>
> 또한 AWS Cloud9 IDE는 동일한 웹 브라우저의 다른 탭 하나 이상에서 실행되고 있어야 합니다. 그렇지 않으면 애플리케이션 미리 보기가 별도의 웹 브라우저 탭에 표시되지 않습니다.
> 또한 AWS Cloud9 IDE는 동일한 웹 브라우저의 다른 탭 하나 이상에서 실행되고 있어야 합니다. 그렇지 않으면 애플리케이션 미리 보기가 별도의 웹 브라우저 탭에 표시되지 않습니다. 애플리케이션 미리 보기 탭에 오류가 표시되거나 비어 있으면 애플리케이션 미리 보기 알림: '서드 파티 쿠키가 사용 중지됨(Third-party cookies disabled)'의 문제 해결 단계를 따릅니다.

다른 미리 보기로 전환

애플리케이션 미리 보기 탭에서 주소 표시줄에 다른 URL에 대한 경로를 입력합니다. 주소 표시줄은 [새로 고침(Refresh)] 버튼과 미리 보기 유형 목록 사이에 있습니다.

인터넷을 통해 실행 중인 애플리케이션 공유

실행 중인 애플리케이션을 미리 본 후 인터넷을 통해 다른 사용자가 사용할 수 있도록 설정할 수 있습니다.

Amazon EC2 인스턴스가 환경에 연결되어 있는 경우 다음 단계를 따릅니다. 그렇지 않으면 해당 서버의 문서를 참조하세요.

주제

1단계: 인스턴스의 ID 및 IP 주소 가져오기

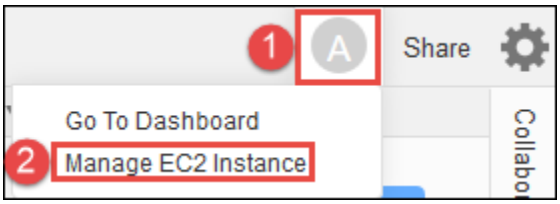
이 단계에서는 환경에 연결된 Amazon EC2 인스턴스의 인스턴스 ID 및 퍼블릭 IP 주소를 기록해 둡니다. 이후 단계에서 들어오는 애플리케이션 요청을 허용하려면 인스턴스 ID가 필요합니다. 그런 다음 다른 사용자와 퍼블릭 IP 주소를 공유하여 실행 중인 애플리케이션에 액세스할 수 있도록 합니다.

1. Amazon EC2 인스턴스의 ID 가져오기 이 정보를 얻으려면 다음 중 하나를 수행합니다.

   - 환경용 AWS Cloud9 IDE의 터미널 세션에서 다음 명령을 실행하여 Amazon EC2 인스턴스의 ID를 가져옵니다.

     ```
     curl http://169.254.169.254/latest/meta-data/instance-id
     ```

     인스턴스 ID의 형식은 `i-12a3b456c789d0123`입니다. 이 인스턴스 ID를 기록해 둡니다.
   - 환경의 IDE에 있는 메뉴 모음에서 사용자 아이콘을 선택한 다음 [EC2 인스턴스 관리(Manage EC2 Instance)]를 선택합니다.

     표시되는 Amazon EC2 콘솔에서 [인스턴스 ID(Instance ID)] 열에 표시되는 인스턴스 ID를 기록해 둡니다. 인스턴스 ID의 형식은 `i-12a3b456c789d0123`입니다.

2. Amazon EC2 인스턴스의 퍼블릭 IP 주소를 가져옵니다. 이 정보를 얻으려면 다음 중 하나를 수행합니다.

   - 환경의 IDE에 있는 메뉴 모음에서 [공유(Share)]를 선택합니다. [이 환경 공유(Share this environment)] 대화 상자에서 [애플리케이션(Application)] 상자의 퍼블릭 IP 주소를 기록해 둡니다. 퍼블릭 IP 주소의 형식은 `192.0.2.0`입니다.
   - 환경의 IDE에 있는 터미널 세션에서 다음 명령을 실행하여 Amazon EC2 인스턴스의 퍼블릭 IP 주소를 가져옵니다.

     ```
     curl http://169.254.169.254/latest/meta-data/public-ipv4
     ```

     퍼블릭 IP 주소의 형식은 `192.0.2.0`입니다. 이 퍼블릭 IP 주소를 기록해 둡니다.
   - 환경의 IDE에 있는 메뉴 모음에서 사용자 아이콘을 선택한 다음 [EC2 인스턴스 관리(Manage EC2 Instance)]를 선택합니다. 표시되는 Amazon EC2 콘솔의 [설명(Description)] 탭에서 [IPv4 퍼블릭 IP(IPv4 Public IP)] 필드의 퍼블릭 IP 주소를 기록해 둡니다. 퍼블릭 IP 주소의 형식은 `192.0.2.0`입니다.

   > **Note**
   >
   > 애플리케이션의 퍼블릭 IP 주소는 애플리케이션 인스턴스가 다시 시작될 때마다 변경될 수 있습니다. IP 주소가 변경되지 않도록 하려면 탄력적 IP 주소를 할당합니다. 그런 다음 실행 중인 인스턴스에 해당 주소를 할당합니다. 지침은 Amazon EC2 사용 설명서의 엘라스틱 IP 주소 할당 및 실행 중인 인스턴스에 엘라스틱 IP 주소 연결을 참조하십시오. 엘라스틱 IP 주소를 할당하면 요금이 부과될 수 있습니다 AWS 계정 . 자세한 내용은 Amazon EC2 요금을 참조하세요.

2단계: 인스턴스의 보안 그룹 설정

이 단계에서는 Amazon EC2 콘솔에서 환경에 연결된 인스턴스에 대한 Amazon EC2 보안 그룹을 설정합니다. 포트 8080, 8081 또는 8082를 통해 들어오는 HTTP 요청을 허용하도록 설정합니다.

> **Note**
>
> 포트 `8080`, `8081` 또는 `8082`를 통해 HTTP를 사용하여 실행할 필요는 없습니다. 이 경우 IDE 내에서 실행 중인 애플리케이션을 미리 볼 수 없습니다. 자세한 정보는 실행 중인 애플리케이

션 미리 보기을 참조하세요. 그렇지 않은 경우 다른 프로토콜 또는 포트에서 실행 중이라면 이 단계에서 해당 값으로 바꿉니다.
추가 보안 계층을 구현하기 위해, 인스턴스가 사용할 수 있는 VPC의 서브넷에 대한 네트워크 액세스 제어 목록(ACL)을 설정할 수도 있습니다. 보안 그룹과 네트워크 ACL에 대한 자세한 내용은 다음을 참조하세요.

- 3단계: 인스턴스의 서브넷 설정
- Amazon VPC 사용 설명서의 보안
- Amazon VPC 사용 설명서의 VPC의 보안 그룹
- Amazon VPC 사용 설명서의 네트워크 ACL

1. 환경의 IDE에 있는 메뉴 모음에서 사용자 아이콘을 선택한 다음 [EC2 인스턴스 관리(Manage EC2 Instance)]를 선택합니다. 그런 다음 이 절차의 3단계로 건너뜁니다.
2. Manage EC2 Instance(EC2 인스턴스 관리)를 선택하거나 이 절차의 다른 단계에서 오류가 반환되는 경우 AWS 계정의 관리자 보안 인증 정보를 사용하여 Amazon EC2 콘솔에 로그인합니다. 다음 지침을 따릅니다. 이렇게 할 수 없으면 AWS 계정 관리자에게 문의하세요.

   a. 아직 로그인하지 않았다면 AWS Management Console https://console.aws.amazon.com/ 에 로그인하세요.

   b. Amazon EC2 콘솔을 엽니다. 이렇게 하려면 탐색 모음에서 Services(서비스)를 선택합니다. 그런 다음 EC2를 선택합니다.

   c. 탐색 표시줄에서 환경이 위치한 AWS 리전 위치를 선택합니다.

   d. [EC2 대시보드(EC2 Dashboard)]가 표시되면 [실행 인스턴스(Running Instances)]를 선택합니다. 그렇지 않으면, 서비스 탐색 모음에서 아직 확장하지 않은 경우 Instances(인스턴스)를 확장한 후 Instances(인스턴스)를 선택합니다.

   e. 인스턴스 목록에서 Instance ID(인스턴스 ID)가 앞서 기록해 둔 인스턴스 ID와 일치하는 인스턴스를 선택합니다.
3. 인스턴스의 Description(설명) 탭에서 Security groups(보안 그룹) 옆에 있는 보안 그룹 링크를 선택합니다.
4. 보안 그룹이 표시된 상태에서 [인바운드(Inbound)] 탭을 봅니다. Type(유형)이 Custom TCP Rule(사용자 지정 TCP 규칙)로 설정되어 있고 Port Range(포트 범위)가 8080, 8081 또는 8082로 설정되어 있는 규칙이 이미 있으면, Cancel(취소)을 선택하고 3단계: 인스턴스의 서브넷 설정 섹션으로 건너뜁니다. 그렇지 않으면 [편집(Edit)]을 선택합니다.
5. [인바운드 규칙 편집(Edit inbound rules)] 대화 상자에서 [규칙 추가(Add Rule)]를 선택합니다.

6. 유형의 경우 사용자 지정 TCP 규칙을 선택합니다.
7. Port Range(포트 범위)에 `8080`, `8081` 또는 `8082`를 입력합니다.
8. Source(소스)에서 Anywhere(위치 무관)를 선택합니다.

> **Note**
>
> Source(소스)에서 Anywhere(위치 무관)을 선택하면 모든 IP 주소에서 들어오는 요청이 허용됩니다. 이를 특정 IP 주소로 제한하려면 Custom(사용자 지정)을 선택한 다음 IP 주소 범위를 입력합니다. 또는 My IP(내 IP)를 선택하여 사용자의 IP 주소에서만 요청을 보내도록 제한합니다.

9. 저장을 선택합니다.

3단계: 인스턴스의 서브넷 설정

Amazon EC2 및 Amazon VPC 콘솔을 사용하여 환경에 연결된 Amazon EC2 인스턴스용 서브넷을 설정합니다. 포트 8080, 8081 또는 8082를 통해 들어오는 HTTP 요청을 허용합니다.

> **Note**
>
> 포트 `8080`, `8081` 또는 `8082`를 통해 HTTP를 사용하여 실행할 필요는 없습니다. 단, 이 경우 IDE 내에서 실행 중인 애플리케이션을 미리 볼 수 없습니다. 자세한 정보는 실행 중인 애플리케이션 미리 보기을 참조하세요. 그렇지 않은 경우 다른 프로토콜 또는 포트에서 실행 중이라면 이 단계에서 해당 값으로 바꿉니다.
>
> 이 단계에서는 인스턴스가 사용할 수 있는 Amazon VPC 서브넷의 네트워크 ACL을 설정하는 방법에 대해 설명합니다. 이 단계는 필수는 아니지만 권장됩니다. 네트워크 ACL을 설정하면 보안 계층이 하나 더 추가됩니다. 네트워크 ACL에 대한 자세한 내용은 다음을 참조하세요.
>
> - Amazon VPC 사용 설명서의 보안
> - Amazon VPC 사용 설명서의 네트워크 ACL

1. Amazon EC2 콘솔의 서비스 탐색 모음에서 아직 확장하지 않은 경우 Instances(인스턴스)를 확장한 후 Instances(인스턴스)를 선택합니다.
2. 인스턴스 목록에서 Instance ID(인스턴스 ID)가 앞서 기록해 둔 인스턴스 ID와 일치하는 인스턴스를 선택합니다.

3. 인스턴스의 [설명(Description)] 탭에서 [서브넷 ID(Subnet ID)]의 값을 기록해 둡니다. 서브넷 ID의 형식은 `subnet-1fab8aEX`입니다.
4. Amazon VPC 콘솔을 엽니다. 이렇게 하려면 AWS 탐색 모음에서 [Services] 를 선택한 다음 VPC 를 선택합니다.

   이 단계에서는 AWS 계정의 관리자 보안 인증 정보를 사용하여 Amazon VPC 콘솔에 로그인하는 것이 좋습니다. 이 작업을 수행할 수 없는 경우 AWS 계정 관리자에게 문의하세요.
5. [VPC 대시보드(VPC Dashboard)]가 표시되면 [서브넷(Subnets)]을 선택합니다. 그렇지 않은 경우 서비스 탐색 창에서 [서브넷(Subnets)]을 선택합니다.
6. 서브넷 목록에서 Subnet ID(서브넷 ID) 값이 앞서 기록해 둔 값과 일치하는 서브넷을 선택합니다.
7. Summary(요약) 탭에서 Network ACL(네트워크 ACL) 옆에 있는 네트워크 ACL 링크를 선택합니다.
8. 네트워크 ACL 목록에서 네트워크 ACL을 선택합니다. (네트워크 ACL이 하나만 있습니다.)
9. [인바운드 규칙(Inbound Rules)] 탭에서 네트워크 ACL을 확인합니다. [유형(Type)]이 [HTTP* (8080)], [HTTP* (8081)] 또는 [HTTP* (8082)]로 설정된 규칙이 이미 있는 경우 4단계: 실행 중인 애플리케이션 URL 공유 섹션으로 건너뜁니다. 그렇지 않으면 [편집(Edit)]을 선택합니다.
10. 다른 규칙 추가(Add another rule)를 선택합니다.
11. Rule #(규칙 번호)에 규칙의 번호를 입력합니다(예: `200`).
12. 유형의 경우 사용자 지정 TCP 규칙을 선택합니다.
13. [포트 범위(Port Range)]에 `8080`, `8081` 또는 `8082`를 입력합니다.
14. [소스(Source)]에 들어오는 요청을 허용할 IP 주소 범위를 입력합니다. 예를 들어 모든 IP 주소에서 들어오는 요청을 허용하려면 `0.0.0.0/0`을 입력합니다.
15. [허용/거부(Allow / Deny)]가 [허용(ALLOW)]으로 설정된 상태에서 [저장(Save)]을 선택합니다.

4단계: 실행 중인 애플리케이션 URL 공유

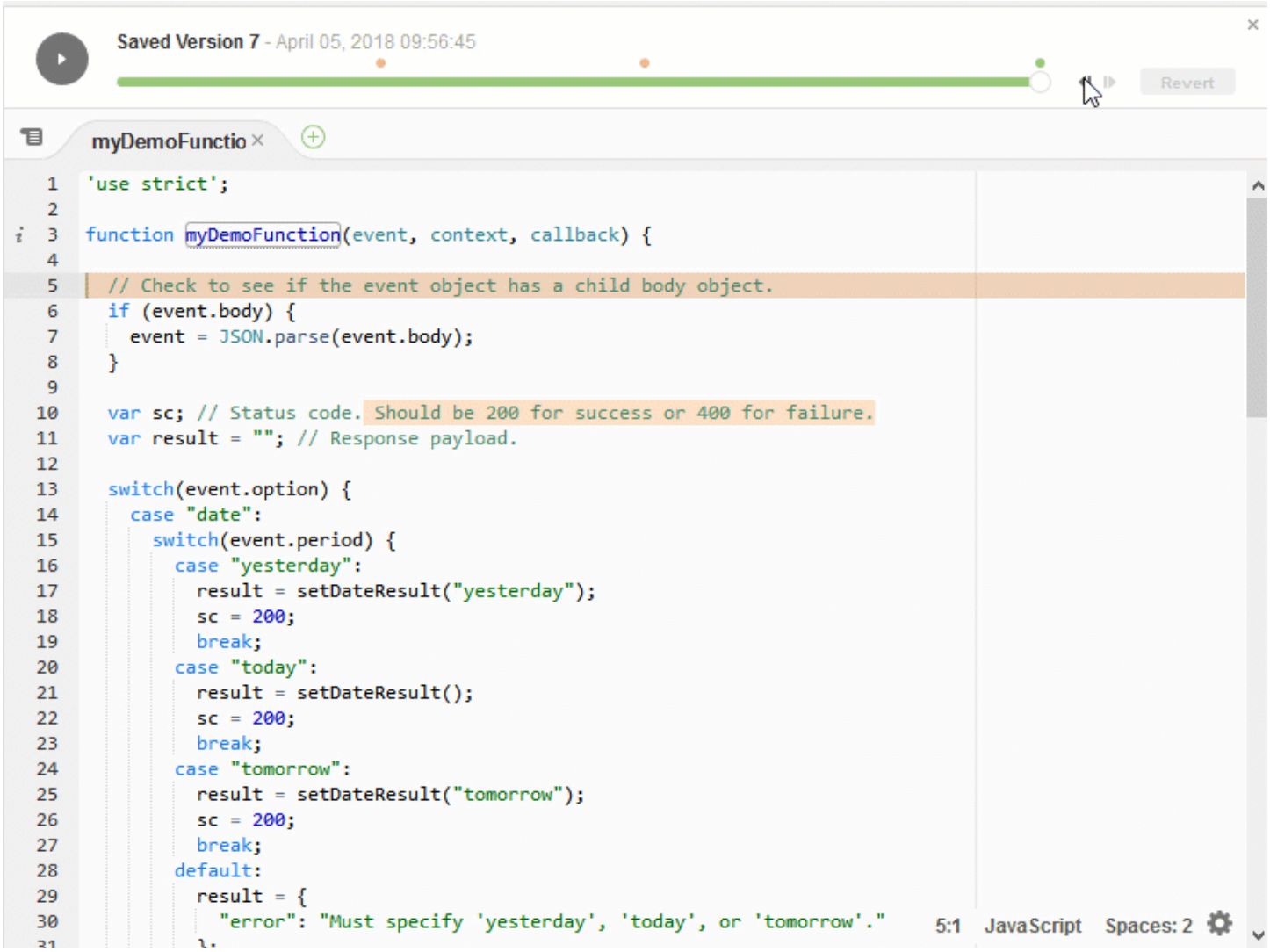
애플리케이션이 실행된 후에는 애플리케이션의 URL을 제공하여 다른 사용자와 애플리케이션을 공유할 수 있습니다. 이를 위해서는 앞서 기록해 둔 퍼블릭 IP 주소가 필요합니다. 애플리케이션의 전체 URL을 작성하려면 애플리케이션의 퍼블릭 IP 주소를 올바른 프로토콜로 시작해야 합니다. 다음으로 애플리케이션 포트가 사용하는 프로토콜의 기본 포트가 아닌 경우 포트 번호 정보를 추가합니다. 다음은 포트 8080을 통해 HTTP를 사용하는 애플리케이션 URL의 예입니다. `http://192.0.2.0:8080/index.html`

결과 웹 브라우저 탭에 오류가 표시되거나 탭이 비어 있으면 IDE 외부에서 실행 중인 애플리케이션을 표시할 수 없음의 문제 해결 단계를 따릅니다.

> **Note**
>
> 애플리케이션의 퍼블릭 IP 주소는 애플리케이션 인스턴스가 다시 시작될 때마다 변경될 수 있습니다. IP 주소가 변경되지 않도록 하려면 탄력적 IP 주소를 할당한 후 해당 주소를 실행 중인 인스턴스에 할당합니다. 지침은 Amazon EC2 사용 설명서의 엘라스틱 IP 주소 할당 및 실행 중인 인스턴스에 엘라스틱 IP 주소 연결을 참조하십시오. 엘라스틱 IP 주소를 할당하면 요금이 부과될 수 있습니다 AWS 계정 . 자세한 내용은 Amazon EC2 요금을 참조하세요.
> 포트 8080, 8081 또는 8082를 통한 HTTP를 사용하여 애플리케이션을 실행할 필요는 없습니다. 단, 이 경우 IDE 내에서 실행 중인 애플리케이션을 미리 볼 수 없습니다. 자세한 정보는 실행 중인 애플리케이션 미리 보기을 참조하세요.
> 예를 들어, 요청이 요청된 프로토콜 또는 포트를 통한 트래픽을 차단하는 VPN에서 시작되는 경우 애플리케이션 URL에 대한 액세스 요청이 실패할 수 있습니다. 요청된 프로토콜 및 포트를 통한 트래픽을 허용하는 다른 네트워크에서 요청을 수행해야 합니다. 자세한 내용은 네트워크 관리자에게 문의하세요.
> IDE의 애플리케이션 미리 보기 탭에 있는 URL을 다른 사용자와 공
> 유하지 않는 것이 좋습니다. (이 URL의 형식은 다음과 같습니다.
> `https://12a34567b8cd9012345ef67abcd890e1.vfs.cloud9.us-east-2.amazonaws.com/` 이 형식에서 `12a34567b8cd9012345ef67abcd890e1` 는 환경에 AWS Cloud9 할당되는 ID입니다. `us-east-2`환경의 AWS 리전 ID입니다.) 이 URL은 환경의 IDE가 열려 있고 애플리케이션이 동일한 웹 브라우저에서 실행 중인 경우에만 작동합니다.

AWS Cloud9 통합 개발 환경(IDE)의 파일 개정 작업

AWS Cloud9 IDE의 [파일 개정 이력(File Revision History)] 창을 사용하여 AWS Cloud9 EC2 개발 환경에 있는 파일의 변경 사항을 보고 관리할 수 있습니다. [파일 개정 이력(File Revision History)] 창은 AWS Cloud9 SSH 개발 환경의 파일에서는 사용할 수 없습니다.

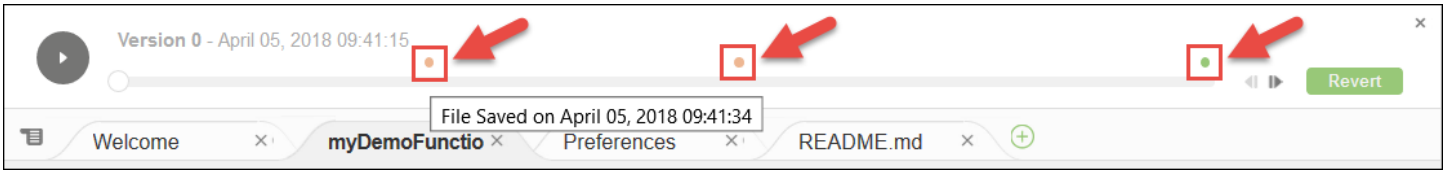


파일의 File Revision History(파일 개정 이력) 창을 표시하려면 편집기에서 해당 파일을 엽니다. 그런 다음 메뉴 모음에서 File, Show File Revision History(파일, 파일 개정 이력 표시)를 선택합니다.

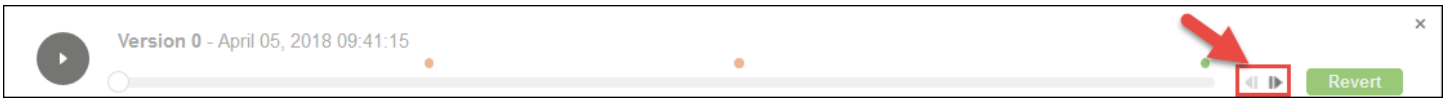
[파일 개정 이력(File Revision History)] 창은 환경에서 편집기에서 파일을 처음 연 후 해당 환경에 대해서만 IDE에서 파일의 개정 기록을 추적하기 시작합니다. File Revision History(파일 개정 이력) 창은 편집기 자체에서만 파일의 개정을 추적합니다. 다른 방법으로 만든 파일의 개정은 추적하지 않습니다(예: 터미널, Git 또는 다른 파일 개정 도구).

File Revision History(파일 개정 이력) 창이 표시된 동안에는 파일을 편집할 수 없습니다. 창을 숨기려면 File, Show Revision History(파일, 개정 이력 표시)를 다시 선택하거나, 창 모서리의 X(Close timeslider(시간 슬라이더 닫기))를 선택합니다.

파일 저장 작업과 관련된 파일 버전으로 이동하려면 개정 슬라이더 위의 File Saved on(파일을 저장한 시점) 점을 선택합니다.



개정 슬라이더에서 현재 선택한 파일 버전을 한 버전 앞으로 또는 뒤로 이동하려면 이동 화살표(Step revision forward(개정 앞으로 이동) 또는 Step revision backward(개정 뒤로 이동)) 중 하나를 선택합니다.



한 파일 버전의 개정 이력을 처음부터 끝까지 한 번에 자동으로 앞으로 이동하려면 재생 버튼(Playback file history(파일 이력 재생))을 선택합니다.

현재 선택한 파일 버전을 개정 이력의 최신 버전으로 만들려면 되돌리기를 선택합니다.

AWS Cloud9 통합 개발 환경(IDE)의 이미지 파일 작업

AWS Cloud9 IDE를 사용하여 이미지 파일을 보고 편집할 수 있습니다.

- 이미지 보기 또는 편집
- 이미지 크기 조정
- 이미지 자르기
- 이미지 회전
- 이미지 반전
- 이미지 확대/축소
- 이미지 부드럽게

이미지 보기 또는 편집

AWS Cloud9 IDE에서 보거나 편집할 이미지의 파일을 엽니다. 지원되는 이미지 파일 형식은 다음과 같습니다.

- `.bmp`
- `.gif`(보기 전용)

- .ico(보기 전용)
- .jpeg
- .jpg
- .png
- .tiff

이미지 크기 조정

1. IDE에서 이미지 파일을 엽니다.
2. 이미지 편집 모음에서 [크기 조정(Resize)]을 선택합니다.
3. 이미지 너비를 변경하려면 새 [너비(Width)]를 픽셀 단위로 입력합니다. 아니면 [너비(Width)] 옆에 있는 '-' 또는 '+'를 사용하여 현재 너비를 한 번에 한 픽셀씩 변경합니다.
4. 이미지 높이를 변경하려면 새 [높이(Height)]를 픽셀 단위로 입력합니다. 아니면 [높이(Height)] 옆에 있는 '-' 또는 '+'를 사용하여 현재 높이를 한 번에 한 픽셀씩 변경합니다.
5. 이미지의 너비와 높이 비율을 유지하려면 [종횡비 유지(Maintain Aspect Ratio)]를 선택합니다.
6. 이미지의 새 크기를 확인하려면 이미지 편집 모음에서 너비(W) 및 높이(H) 측정치를 픽셀 단위로 확인합니다.
7. 크기 조정(Resize)을 선택합니다.
8. 크기 조정을 취소하려면 메뉴 모음에서 [편집(Edit)], [실행 취소(Undo)]를 선택합니다. 새 크기를 유지하려면 [파일(File)], [저장(Save)]을 선택합니다.

이미지 자르기

1. IDE에서 이미지 파일을 엽니다.
2. 이미지의 유지하려는 부분 위로 포인터를 드래그합니다.
3. 선택 영역의 크기를 확인하려면 이미지 편집 모음에서 [선택 영역(Selection)] 치수를 다음과 같이 설정합니다.
   - 원본 이미지의 왼쪽 가장자리에서 선택 영역의 왼쪽 가장자리까지의 픽셀 단위 거리(L)
   - 원본 이미지의 위쪽 가장자리에서 선택 영역의 위쪽 가장자리까지의 픽셀 단위 거리(T)
   - 선택 영역의 픽셀 단위 너비(W)
   - 선택 영역의 픽셀 단위 높이(H)
4. 이미지 편집 모음에서 [자르기(Crop)]를 선택합니다.

5. 자르기를 취소하려면 메뉴 모음에서 [편집(Edit)], [실행 취소(Undo)]를 선택합니다. 자른 새 이미지를 유지하려면 [파일(File)], [저장(Save)]을 선택합니다.

이미지 회전

1. IDE에서 이미지 파일을 엽니다.
2. 이미지를 시계 반대 방향으로 회전하려면 이미지 편집 모음에서 [왼쪽으로 90도 회전(Rotate 90 Degrees Left)]을 선택합니다.
3. 이미지를 시계 방향으로 회전하려면 이미지 편집 모음에서 [오른쪽으로 90도 회전(Rotate 90 Degrees Right)]을 선택합니다.
4. 회전을 취소하려면 메뉴 모음에서 [편집(Edit)], [실행 취소(Undo)]를 선택합니다. 회전한 새 이미지를 유지하려면 [파일(File)], [저장(Save)]을 선택합니다.

이미지 반전

1. IDE에서 이미지 파일을 엽니다.
2. 이미지를 가로로 뒤집으려면 이미지 편집 모음에서 [FlipH]를 선택합니다.
3. 이미지를 세로로 뒤집으려면 이미지 편집 모음에서 [FlipV]를 선택합니다.
4. 뒤집기를 취소하려면 메뉴 모음에서 [편집(Edit)], [실행 취소(Undo)]를 선택합니다. 뒤집힌 새 이미지를 유지하려면 [파일(File)], [저장(Save)]을 선택합니다.

이미지 확대/축소

1. IDE에서 이미지 파일을 엽니다.
2. 이미지 편집 모음에서 사용 가능한 확대/축소 계수(예: 75%, 100% 또는 200%) 중 하나를 선택합니다.

이미지 부드럽게

1. IDE에서 이미지 파일을 엽니다.
2. 이미지 편집 모음에서 [부드럽게(Smooth)]를 선택하여 이미지의 픽셀화 양을 줄입니다. 부드럽게 하는 효과를 취소하려면 [부드럽게(Smooth)]를 선택 취소합니다.
3. 메뉴 모음에서 [파일(File)], [저장(Save)]을 선택합니다.

AWS Cloud9 통합 개발 환경(IDE)의 빌더, 러너 및 디버거 작업

빌더는 AWS Cloud9 IDE에 프로젝트의 파일을 작성하는 방법을 지시합니다. 러너는 AWS Cloud9 IDE에 특정 유형의 파일을 실행하는 방법을 지시합니다. 러너는 디버거를 사용하여 파일의 소스 코드에서 문제를 찾는 데 도움을 줄 수 있습니다.

AWS Cloud9 IDE를 사용하여 다음과 같은 방법으로 코드를 빌드, 실행 및 디버그할 수 있습니다.

- 빌더를 사용하여 프로젝트의 파일을 빌드합니다. 프로젝트의 파일 빌드를 참조하세요.
- 러너를 사용하여 코드를 실행(선택적으로 디버그)합니다. 기본 빌드, 실행 및 디버그 지원 및 코드 실행을 참조하세요.
- 원래 정의된 방식과 다른 방식으로 코드를 실행(선택적으로 디버그)하도록 기본 제공 러너를 변경합니다. 기본 제공 러너 변경을 참조하세요.
- 러너를 사용하여 파일 이름, 명령줄 옵션, 디버그 모드, 현재 작업 디렉터리 및 환경 변수의 사용자 지정 조합으로 코드를 실행(선택적으로 디버그)합니다. 실행 구성 생성을 참조하세요.
- 자체 빌더 또는 러너를 생성합니다. 빌더 또는 러너 생성을 참조하세요.

기본 빌드, 실행 및 디버그 지원

AWS Cloud9 IDE는 여러 언어에 대한 코드 작성, 실행 및 디버깅을 위한 지원을 기본 제공합니다. 전체 목록은 언어 지원을 참조하세요.

기본 빌드 지원은 메뉴 모음에서 [실행(Run)], [빌드 시스템] 및 [실행(Run], [빌드(Build)] 메뉴 명령을 통해 사용할 수 있습니다. 목록에 없는 프로그래밍 언어 또는 도구에 대한 지원을 추가하려면 빌더 또는 러너 생성을 참조하세요.

기본 실행 지원은 [실행(Run)] 버튼과 메뉴 모음에서 [실행(Run)], [다음으로 실행(Run With)] 및 [실행(Run)], [실행 구성(Run Configurations)] 메뉴 명령을 통해 사용할 수 있습니다. 목록에 없는 프로그래밍 언어 또는 도구에 대한 지원을 추가하려면 빌더 또는 러너 생성 및 실행 구성 생성을 참조하세요.

기본 디버그 지원은 [디버거(Debugger)] 창을 통해 사용할 수 있습니다. [디버거(Debugger)] 창을 표시하려면 [디버거(Debugger)] 버튼을 선택합니다. [디버거(Debugger)] 버튼이 표시되지 않으면 메뉴 모음에서 [창(Window)], [디버거(Debugger)]를 선택합니다.

프로젝트의 파일 빌드

1. 빌드하려는 코드에 해당하는 파일을 엽니다.

2. 메뉴 모음에서 [실행, 빌드 시스템(Run, Build System)]을 선택한 다음 사용할 빌더의 이름을 선택합니다(아직 선택되지 않은 경우). 사용하려는 빌더가 목록에 없으면 이 절차를 중지하고 빌더 또는 러너 생성의 단계를 완료한 후 이 절차로 돌아옵니다.
3. [실행, 빌드(Run, Build)]를 선택합니다.

코드 실행

1. 파일이 아직 열려 있지 않고 선택되지 않은 경우 실행할 코드에 해당하는 파일을 엽니다.
2. 메뉴 모음에서 다음 중 하나를 선택합니다.
   - 가장 일치하는 기본 제공 러너로 코드를 실행하려면 [실행, 실행(Run, Run)]을 선택합니다. AWS Cloud9이 그러한 러너를 찾지 못할 경우 이 명령이 사용 중지됩니다.
   - AWS Cloud9이 마지막으로 사용한 실행 구성으로 코드를 실행하려면 [실행, 마지막 실행(Run, Run Last)]을 선택합니다.
   - 특정 러너로 코드를 실행하려면 [실행, 다음으로 실행(Run, Run With)]을 선택하고 러너 이름을 선택합니다. 사용하려는 러너가 목록에 없으면 이 절차를 중지하고 빌더 또는 러너 생성의 단계를 완료한 후 이 절차로 돌아옵니다.
   - 파일 이름, 명령줄 옵션, 디버그 모드, 현재 작업 디렉터리 및 환경 변수의 사용자 지정 조합으로 특정 러너를 사용하여 코드를 실행하려면 [실행, 실행 구성(Run, Run Configurations)]을 선택한 다음 실행 구성의 이름을 선택합니다. 표시되는 실행 구성 탭에서 [러너: 자동(Runner: Auto)]을 선택하고 사용할 러너를 선택한 다음 [실행(Run)]을 선택합니다. 사용하려는 러너가 목록에 없으면 이 절차를 중지하고 빌더 또는 러너 생성의 단계를 완료한 후 이 절차로 돌아옵니다.

코드 디버그

1. 코드에 대한 실행 구성 탭에서 [디버그 모드로 실행(Run in Debug Mode)]을 선택합니다. 버그 아이콘이 흰색 배경의 녹색으로 바뀝니다. 자세한 내용은 코드 실행 및 실행 구성 생성을 참조하세요.
2. 실행 중에 일시 중지할 코드의 중단점을 다음과 같이 설정합니다.
   a. 중단점을 설정할 각 파일을 엽니다.
   b. 파일의 중단점을 설정할 각 지점에서, 줄 번호 왼쪽에 있는 거터의 빈 영역을 선택합니다. 빨간색 원이 나타납니다.

      중단점을 제거하려면 거터에서 기존 중단점을 선택합니다.

중단점을 제거하는 대신 사용 중지하려면 [디버거(Debugger)] 창의 [중단점(Breakpoint)]에서 사용 중지할 중단점에 해당하는 상자를 선택 취소합니다. 중단점을 다시 사용하려면 선택 취소한 상자를 선택합니다.

한 번에 모든 중단점을 사용 중지하려면 [디버거(Debugger)] 창에서 [모든 중단점 비활성화(Deactivate All Breakpoints)]를 선택합니다. 모든 중단점을 다시 사용하려면 [모든 중단점 활성화(Activate All Breakpoints)]를 선택합니다.

[디버거(Debugger)] 창이 표시되지 않는 경우 [디버거(Debugger)] 버튼을 선택합니다. [디버거(Debugger)] 버튼이 표시되지 않으면 메뉴 모음에서 [창(Window)], [디버거(Debugger)]를 선택합니다.

3. 실행이 일시 중지되는 지점에서 값을 가져올 조사식을 다음과 같이 설정합니다.
   a. [디버거(Debugger)] 창의 [조사식(Watch Expressions)]에서 [여기에 표현식 입력(Type an expression here)]를 선택합니다.
   b. 조사하려는 표현식을 입력한 다음 `Enter` 키를 누릅니다.

      기존 조사식을 변경하려면 조사식을 마우스 오른쪽 버튼으로 클릭한 다음 [조사식 편집(Edit Watch Expression)]을 선택합니다. 변경 내용을 입력한 후 `Enter` 키를 누릅니다.

      기존 조사식을 제거하려면 조사식을 마우스 오른쪽 버튼으로 클릭한 다음 [조사식 제거(Remove Watch Expression)]를 선택합니다.
4. 코드 실행에서 설명하는 대로 코드를 실행합니다.

실행이 일시 중지될 때마다 표시된 코드(예: 변수)를 포인터로 가리켜 제공되는 관련 정보를 도구 설명에 표시할 수도 있습니다.

기본 제공 러너 변경

1. 메뉴 모음에서 [실행, 다음으로 실행(Run, Run With)]을 선택한 후 변경할 기본 제공 러너를 선택합니다.
2. 표시되는 실행 구성 탭에서 [중지(Stop)]를 선택하여 러너가 코드를 실행하려고하지 않도록 합니다.
3. [러너: 내 러너(Runner: My Runner)]를 선택합니다. 여기서 [내 러너(My Runner)]는 변경하려는 러너의 이름입니다. 그런 다음 [러너 편집(Edit Runner)]을 선택합니다.
4. 표시되는 [My Runner.run] 탭에서 러너의 현재 정의를 변경합니다. 빌더 또는 러너 정의를 참조하세요.

5. File, Save As(파일, 다른 이름으로 저장)를 선택합니다. 파일을 `my-environment/.c9/runners` 디렉터리에 같은 이름(My Runner.run)으로 저장합니다. 여기서 `my-environment`는 AWS Cloud9 개발 환경의 이름입니다.

> **Note**
>
> 기본 제공 러너에 대한 변경 사항은 해당 변경을 수행한 환경에만 적용됩니다. 변경 사항을 별도의 환경에 적용하려면 다른 환경을 연 다음 위의 단계에 따라 해당 기본 제공 러너를 열고 편집하고 동일한 변경 사항을 저장합니다.

실행 구성 생성

메뉴 모음에서 [실행, 실행 구성, 새 실행 구성(Run, Run Configurations, New Run Configuration)]을 선택합니다. 표시되는 실행 구성 탭에서 다음을 수행합니다.

1. [실행(Run)] 및 [다시 시작(Restart)] 옆의 상자에 이 실행 구성의 [실행, 실행 구성(Run, Run Configurations)] 메뉴에 표시할 이름을 입력합니다.
2. [명령(Command)] 상자에 사용할 사용자 지정 명령줄 옵션을 입력합니다.
3. 이 실행 구성에 러너의 미리 정의된 디버깅 설정을 사용하도록 하려면 [디버그 모드로 실행(Run in Debug Mode)]을 선택합니다. 버그 아이콘이 흰색 배경의 녹색으로 바뀝니다.
4. 이 실행 구성에 특정 작업 디렉터리를 사용하도록 하려면 [CWD]를 선택하고 사용할 디렉터리를 선택한 다음 [선택(Select)]을 선택합니다.
5. 이 실행 구성에 특정 환경 변수를 사용하도록 하려면 [ENV]를 선택한 다음 각 환경 변수의 이름과 값을 입력합니다.

이 실행 구성을 사용하려면 실행할 코드에 해당하는 파일을 엽니다. 메뉴 모음에서 [실행, 실행 구성(Run, Run Configurations)]를 선택한 다음 이 실행 구성의 이름을 선택합니다. 표시되는 실행 구성 탭에서 [러너: 자동(Runner: Auto)]을 선택하고 사용할 러너를 선택한 다음 [실행(Run)]을 선택합니다.

> **Note**
>
> 생성한 모든 실행 구성은 해당 실행 구성을 생성한 환경에만 적용됩니다. 해당 실행 구성을 별도의 환경에 추가하려면 다른 환경을 연 다음 위의 단계에 따라 해당 환경에서 동일한 실행 구성을 생성합니다.

빌더 또는 러너 생성

1. 빌더를 생성하려면 메뉴 모음에서 [실행, 빌드 시스템, 새 빌드 시스템(Run, Build System, New Build System)]을 선택합니다. 러너를 생성하려면 메뉴 모음에서 [실행, 실행 도구, 새 러너(Run, Run With, New Runner)]를 선택합니다.
2. 표시되는 빌더 탭([My Builder.build]로 표시됨) 또는 러너 탭([My Runner.run]으로 표시됨)에서 빌더 또는 러너를 정의합니다. 빌더 또는 러너 정의를 참조하세요.
3. 빌더 또는 러너를 정의한 후 [파일, 다른 이름으로 저장(File, Save As)]을 선택합니다. 빌더의 경우, `my-environment/.c9/builders` 디렉터리에 `.build` 확장명으로 파일을 저장합니다. 여기서 `my-environment`는 환경의 이름입니다. 러너의 경우, `my-environment/.c9/runners` 디렉터리에 `.run` 파일 확장명으로 파일을 저장합니다. 여기서 `my-environment`는 환경의 이름입니다. 지정한 파일 이름은 [실행, 빌드 시스템(Run, Build System)] 메뉴(빌더의 경우) 또는 [실행, 다음으로 실행(Run, Run With)] 메뉴(러너의 경우)에 표시되는 이름이 됩니다. 따라서 다른 파일 이름을 지정하지 않는 한 기본적으로 표시 이름은 My Builder(빌더의 경우) 또는 My Runner(러너의 경우)가 됩니다.

이 빌더 또는 러너를 사용하려면 프로젝트의 파일 빌드 또는 코드 실행을 참조하세요.

> **Note**
>
> 생성하는 모든 빌더 또는 러너는 해당 빌더 또는 러너를 생성한 환경에만 적용됩니다. 해당 실행 빌더 또는 러너를 별도의 환경에 추가하려면 다른 환경을 연 다음 위의 단계에 따라 해당 환경에서 동일한 빌더 또는 러너를 생성합니다.

빌더 또는 러너 정의

이 절차에서는 [실행, 빌드 시스템, 새 빌드 시스템(Run, Build System, New Build System)](빌더의 경우) 또는 [실행, 다음으로 실행, 새 러너(Run, Run With, New Runner)](러너의 경우)를 선택하여 빌더 또는 러너를 이미 생성하기 시작했다고 가정합니다.

표시되는 빌더 또는 러너 탭에서 JSON을 사용하여 러너 또는 빌더를 정의합니다. 다음 코드를 템플릿으로 사용하여 시작합니다.

빌더의 경우 이 코드로 시작합니다.

```
{
  "cmd": [],
```

```
  "info": "",
  "env": {},
  "selector": ""
}
```

러너의 경우 이 코드로 시작합니다.

```
{
  "cmd": [],
  "script": "",
  "working_dir": "",
  "info": "",
  "env": {},
  "selector": "",
  "debugger": "",
  "debugport": ""
}
```

앞의 코드에서:

- `cmd`: AWS Cloud9이 단일 명령으로 실행할 문자열의 쉼표로 구분된 목록을 나타냅니다.

  AWS Cloud9이 이 명령을 실행하면 목록의 각 문자열은 단일 공백으로 구분됩니다. 예를 들어 AWS Cloud9은 `"cmd": [ "ls", "$file", "$args"]`를 `ls $file $args`로 실행합니다. 여기서 AWS Cloud9은 `$file`을 현재 파일의 전체 경로로 바꾸고 `$args`를 파일 이름 뒤에 입력된 인수로 바꿉니다. 자세한 내용은 이 섹션의 뒷부분에 나오는 지원되는 변수 목록을 참조하세요.
- `script`: 러너가 터미널에서 실행하는 bash 스크립트(가독성을 위해 필요에 따라 줄 배열로 지정할 수도 있음)를 나타냅니다.
- `working_dir`: 러너가 실행될 디렉터리를 나타냅니다.
- `info`: 실행 시작 시 사용자에게 표시할 텍스트 문자열을 나타냅니다. 이 문자열은 `Running $project_path$file_name...`과 같은 변수를 포함할 수 있습니다. 여기서 AWS Cloud9은 `$project_path`를 현재 파일의 디렉터리 경로로 바꾸고 `$file_name`을 현재 파일의 이름 부분으로 바꿉니다. 이 섹션의 뒷부분에 나오는 지원되는 변수 목록을 참조하세요.
- `env`: AWS Cloud9이 사용할 명령줄 인수의 배열을 나타냅니다. 예를 들면 다음과 같습니다.

  ```
  "env": {
    "LANG": "en_US.UTF-8",
    "SHLVL": "1"
  }
  ```

- selector: AWS Cloud9이 이 러너에 적용되는 파일 이름을 식별하는 데 사용하도록 할 정규식을 나타냅니다. 예를 들어 Python 파일의 경우 source.py로 지정할 수 있습니다.
- debugger: AWS Cloud9이 사용하도록 할, 이 러너와 호환되는 사용 가능한 디버거의 이름을 나타냅니다. 예를 들어 V8 디버거의 경우 v8으로 지정할 수 있습니다.
- debugport: AWS Cloud9이 디버깅 중에 사용하도록 할 포트 번호를 나타냅니다. 예를 들어 사용할 포트 번호로 15454를 지정할 수 있습니다.

다음 표에는 사용할 수 있는 변수가 나와 있습니다.

변수	설명
$file_path	현재 파일의 디렉터리(예: /home/ec2-user/environment 또는 /home/ubuntu/environment)
$file	현재 파일의 전체 경로(예: /home/ec2-user/environment/hello.py 또는 /home/ubuntu/environment/hello.py)
$args	파일 이름 뒤에 입력된 인수(예: "5" "9")
$file_name	현재 파일의 이름 부분(예: hello.py)
$file_extension	현재 파일의 확장명(예: py)
$file_base_name	파일 확장명을 제외한 현재 파일의 이름(예: hello)
$packages	패키지 폴더의 전체 경로
$project	현재 프로젝트 폴더의 전체 경로
$project_path	현재 프로젝트 파일의 디렉터리(예: /home/ec2-user/environment/ 또는 /home/ubuntu/environment/)

변수	설명
$project_name	파일 확장명을 제외한 현재 프로젝트 파일의 이름(예: my-demo-environment)
$project_extension	현재 프로젝트 파일의 확장명
$project_base_name	파일 확장명을 제외한 현재 프로젝트 파일의 이름
$hostname	환경의 호스트 이름(예: 192.0.2.0)
$hostname_path	프로젝트 파일의 상대 경로를 사용한 환경의 호스트 이름(예: https://192.0.2.0/hello.js)
$url	환경에 액세스하기 위한 전체 URL(예: https://192.0.2.0.)
$port	환경에 할당된 포트(예: 8080)
$ip	환경에 대해 프로세스를 실행할 IP 주소(예: 0.0.0.0)

예를 들어 G++.build라는 다음 빌더 파일은 -o 옵션을 사용하여 **g++** 명령을 실행하여 현재 파일(예:hello.cpp)을 객체 모듈로 컴파일하는 GCC의 빌더를 정의합니다. 그런 다음 객체 모듈을 현재 파일(예: hello)과 같은 이름의 프로그램에 연결합니다. 여기서 이에 해당하는 명령은 g++ -o hello hello.cpp입니다.

```
{
  "cmd": [ "g++", "-o", "$file_base_name", "$file_name" ],
  "info": "Compiling $file_name and linking to $file_base_name...",
  "selector": "source.cpp"
}
```

또 다른 예로서, Python.run이라는 다음 러너 파일은 Python을 사용하여 제공된 인수로 현재 파일을 실행하는 러너를 정의합니다. 예를 들어 현재 파일의 이름이 hello.py이고 인수 5 및 9가 제공된 경우 해당하는 명령은 python hello.py 5 9입니다.

```
{
  "cmd": [ "python", "$file_name", "$args" ],
  "info": "Running $file_name...",
  "selector": "source.py"
}
```

마지막으로, `Print Run Variables.run`이라는 다음 러너 파일은 사용 가능한 각 변수의 값을 출력한 다음 중지하는 러너를 정의합니다.

```
{
  "info": "file_path = $file_path, file = $file, args = $args, file_name = $file_name,
 file_extension = $file_extension, file_base_name = $file_base_name, packages
 = $packages, project = $project, project_path = $project_path, project_name
 = $project_name, project_extension = $project_extension, project_base_name =
 $project_base_name, hostname = $hostname, hostname_path = $hostname_path, url = $url,
 port = $port, ip = $ip"
}
```

AWS Cloud9 통합 개발 환경(IDE)의 사용자 지정 환경 변수 작업

AWS Cloud9 IDE는 사용자 지정 환경 변수 설정을 지원합니다. 다음과 같은 방법으로 AWS Cloud9 IDE에서 사용자 지정 환경 변수를 설정할 수 있습니다.

- 명령 수준 사용자 지정 환경 변수 설정
- ~/.bash_profile에서 사용자 지정 사용자 환경 변수 설정
- 로컬 사용자 지정 환경 변수 설정
- ~/.bashrc에서 사용자 지정 사용자 환경 변수 설정
- ENV 목록에서 사용자 지정 환경 변수 설정

명령 수준 사용자 지정 환경 변수 설정

AWS Cloud9 개발 환경에서 명령을 실행할 때 명령 수준 사용자 지정 환경 변수를 설정할 수 있습니다. 이 동작을 테스트하려면 다음 코드를 사용하여 `script.sh`라는 파일을 만듭니다.

```
#!/bin/bash

echo $MY_ENV_VAR
```

다음 명령을 실행하면 터미널에 `Terminal session`이 표시됩니다.

```
MY_ENV_VAR='Terminal session' sh ./script.sh
```

이 주제에 설명된 여러 접근 방식을 사용하여 사용자 지정 환경 변수를 설정한 경우, 사용자 지정 환경 변수의 값을 가져오려고 할 때 이 설정이 다른 모든 설정보다 우선합니다.

~/.bash_profile에서 사용자 지정 사용자 환경 변수 설정

환경의 `~/.bash_profile` 파일에서 사용자 지정 사용자 환경 변수를 설정할 수 있습니다. 이 동작을 테스트하려면 환경의 `~/.bash_profile` 파일에 다음 코드를 추가합니다.

```
export MY_ENV_VAR='.bash_profile file'
```

그런 다음 명령줄에서 `sh ./script.sh`를 실행하면 터미널에 `.bash_profile file`가 표시됩니다. (앞서 설명한 대로 `script.sh` 파일을 만들었다고 가정합니다.)

로컬 사용자 지정 환경 변수 설정

`export` 명령을 실행하여 터미널 세션에서 로컬 사용자 지정 환경 변수를 설정할 수 있습니다. 이 동작을 테스트하려면 터미널 세션에서 다음 명령을 실행합니다.

```
export MY_ENV_VAR='Command line export'
```

그런 다음 명령줄에서 `sh ./script.sh`를 실행하면 터미널에 `Command line export`가 표시됩니다. (앞서 설명한 대로 `script.sh` 파일을 만들었다고 가정합니다.)

`export` 명령 및 `~/.bash_profile` 파일에서 동일한 사용자 지정 환경 변수를 설정한 경우, 사용자 지정 환경 변수의 값을 가져오려고 할 때 **`export`** 명령 설정이 우선합니다.

~/.bashrc에서 사용자 지정 사용자 환경 변수 설정

환경의 `~/.bashrc` 파일에서 사용자 지정 사용자 환경 변수를 설정할 수 있습니다. 이 동작을 테스트하려면 환경의 `~/.bashrc` 파일에 다음 코드를 추가합니다.

```
export MY_ENV_VAR='.bashrc file'
```

그런 다음 명령줄에서 `sh ./script.sh`를 실행하면 터미널에 `.bashrc file`가 표시됩니다. (앞서 설명한 대로 `script.sh` 파일을 만들었다고 가정합니다.)

export 명령 및 ~/.bashrc 파일에서 동일한 사용자 지정 환경 변수를 설정한 경우, 사용자 지정 환경 변수의 값을 가져오려고 할 때 **export** 명령 설정이 우선합니다.

ENV 목록에서 사용자 지정 환경 변수 설정

[실행(Run)] 탭의 [ENV] 목록에서 사용자 지정 환경 변수를 설정할 수 있습니다.

이 동작을 테스트하려면 다음을 수행합니다.

1. 메뉴 모음에서 Run(실행), Run Configurations(실행 구성), New Run Configuration(새로운 실행 구성)을 선택합니다.
2. [New] - Idle([신규] - 유휴) 탭에서 Runner: Auto(실행기: 자동)를 선택한 다음 셸 스크립트를 선택합니다.
3. [ENV]를 선택한 다음 [이름(Name)]에 MY_ENV_VAR을 입력하고 [값(Value)]에 ENV list를 입력합니다.
4. 명령에 ./script.sh를 입력합니다.
5. 실행 버튼을 선택합니다. 실행기 탭에 ENV list가 표시됩니다. (앞서 설명한 대로 script.sh 파일을 만들었다고 가정합니다.)

~/.bash_profile 파일, **export** 명령, ~/.bashrc 파일, ENV 목록에서 동일한 사용자 지정 환경 변수를 설정한 경우, 사용자 지정 환경 변수의 값을 가져오려고 할 때 ~/.bash_profile 파일 설정이 가장 우선하고, **export** 명령 설정, ~/.bashrc 파일 설정, ENV 목록 설정이 그 뒤를 잇습니다.

> **Note**
>
> ENV 목록은 셸 스크립트와는 별개로 코드를 사용하여 사용자 지정 환경 변수를 가져오고 설정하기 위한 유일한 방법입니다.

AWS Cloud9 통합 개발 환경(IDE)의 프로젝트 설정 작업

현재 AWS Cloud9 개발 환경에 적용되는 프로젝트 설정은 다음과 같은 종류의 설정을 포함합니다.

- 소프트 탭 및 새 파일 줄 끝을 사용할지 여부와 같은 코드 편집기 동작
- 무시할 파일 형식
- 표시하거나 표시하지 않을 힌트 및 경고 유형
- JavaScript, PHP, Python 및 Go 같은 프로그래밍 언어의 코드 및 서식 설정

- 코드를 실행하고 빌드할 때 사용할 구성 유형

프로젝트 설정은 단일 환경에만 적용되지만 특정 환경에 대한 프로젝트 설정을 다른 환경에 적용할 수 있습니다.

- 프로젝트 설정 보기 또는 변경
- 환경의 현재 프로젝트 설정을 다른 환경에 적용
- 사용자가 변경할 수 있는 프로젝트 설정

프로젝트 설정 보기 또는 변경

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. 현재 환경에 대한 프로젝트 설정을 보려면 [기본 설정(Preferences)] 탭의 측면 탐색 창에서 [프로젝트 설정(Project Settings)]을 선택합니다.
3. 환경에 대한 현재 프로젝트 설정을 변경하려면 Project Settings(프로젝트 설정) 창에서 원하는 설정을 변경합니다.

사용자가 변경할 수 있는 프로젝트 설정을 참조하세요.

환경의 현재 프로젝트 설정을 다른 환경에 적용

1. 소스 환경과 대상 환경의 AWS Cloud9 IDE 메뉴 모음에서 AWS Cloud9, Open Your Project Settings(프로젝트 설정 열기)를 선택합니다.
2. 소스 환경에서 화면에 표시된 project.settings 탭의 콘텐츠를 복사합니다.
3. 대상 환경에서 project.settings 탭의 콘텐츠를 소스 환경에서 복사한 콘텐츠로 덮어씁니다.
4. 대상 환경에서 project.settings 탭을 저장합니다.

사용자가 변경할 수 있는 프로젝트 설정

다음 섹션에서는 [기본 설정(Preferences)] 탭의 [프로젝트 설정(Project Settings)] 창에서 변경할 수 있는 프로젝트 설정 종류를 설명합니다.

- EC2 인스턴스
- 코드 편집기(Ace)
- 파일에서 찾기

- 힌트 및 경고
- JavaScript 지원
- 빌드
- 실행 및 디버그
- 실행 구성
- 코드 포맷터
- TypeScript 지원
- PHP 지원
- Python 지원
- Go 지원

EC2 인스턴스

내 환경 중지

해당 환경의 IDE에 연결된 모든 웹 브라우저 인스턴스를 닫은 후 환경의 Amazon EC2 인스턴스를 자동으로 중지할 시기를 선택합니다(사용된 경우). 일주일에서 30분 사이의 기간 범위를 선택할 수 있습니다. 또한 AWS Cloud9 IDE를 종료한 후 자동으로 Amazon EC2 인스턴스를 중지하지 않도록 선택할 수도 있습니다.

IDE를 종료한 후 30분 이내에 인스턴스를 중지하려는 경우 콘솔 인터페이스를 사용하여 수동으로 중지합니다.

코드 편집기(Ace)

소프트 탭

이 옵션을 선택하면 Tab 키를 누를 때마다 탭 문자 대신 지정된 수의 공백을 삽입합니다.

로드 시 탭 크기 자동 감지

이 옵션을 선택하면 AWS Cloud9이 탭 크기를 추측하려고 시도합니다.

새 파일 줄 끝

새 파일에 사용할 줄 끝의 유형입니다.

유효한 옵션에는 다음이 포함됩니다.

- Windows(CRLF)의 경우 캐리지 리턴으로 줄을 끝낸 다음 줄 바꿈으로 끝냅니다.
- Unix(LF)의 경우 줄 바꿈만으로 줄을 끝낼 수 있습니다.

저장 시, 공백 제거

이 옵션을 선택하면 AWS Cloud9이 파일이 저장될 때마다 파일에서 불필요한 공백과 탭을 제거하려고 시도합니다.

파일에서 찾기

이 파일 무시

파일에서 찾을 때 AWS Cloud9가 무시하는 파일 형식입니다.

검색할 최대 파일 수(1,000개 단위)

파일에서 찾을 때 AWS Cloud9는 현재 범위에서 찾을 최대 파일 수를 1,000의 배수로 지정합니다.

힌트 및 경고

경고 레벨

사용하도록 설정할 최소 메시지 수준입니다.

유효한 값은 다음과 같습니다.

- 정보(Info) - 정보, 경고 및 오류 메시지를 사용합니다.
- 경고(Warning) - 경고 및 오류 메시지만 사용합니다.
- 오류(Error - 오류 메시지만 사용합니다.

누락된 선택적 세미콜론 표시

선택할 경우 AWS Cloud9이 코드에 사용할 수 있지만 사용되지 않는 세미콜론을 발견할 때마다 파일에 플래그를 추가합니다.

선언되지 않은 변수 표시

선택할 경우 AWS Cloud9이 코드에서 선언되지 않은 변수를 발견할 때마다 파일에 플래그를 추가합니다.

사용되지 않은 함수 인수 표시

선택할 경우 AWS Cloud9이 함수에서 사용되지 않는 인수를 발견할 때마다 파일에 플래그를 추가합니다.

Ignore Messages Matching Regex(정규식과 일치하는 메시지 무시)

AWS Cloud9이 지정된 정규식과 일치하는 메시지를 표시하지 않습니다. 자세한 내용은 Mozilla 개발자 네트워크의 JavaScript 정규식 주제에서 정규식 패턴 작성을 참조하세요.

JavaScript 지원

.eslintrc를 사용하여 JavaScript 경고를 사용자 정의하세요.

선택할 경우 AWS Cloud9는 `.eslintrc` 파일을 사용하여 사용하거나 사용 중지할 JavaScript 경고를 결정합니다. 자세한 내용은 ESLint 웹 사이트에서 구성 파일 형식을 참조하세요.

JavaScript 라이브러리 코드 완성

이 제안 또는 자동 코드 완성을 시도할 때 AWS Cloud9가 사용하는 JavaScript 라이브러리입니다.

저장 시 코드 포맷

선택할 경우, 해당 파일이 저장될 때마다 AWS Cloud9는 JavaScript 파일의 코드 포맷을 시도합니다.

내장 JSBeautify를 코드 포맷터로 사용

선택할 경우, AWS Cloud9는 JSBeautify의 내부 구현을 사용하여 파일의 코드 가독성을 높이려고 합니다.

사용자 지정 코드 포맷터

이 JavaScript 파일에서 코드를 포맷할 때 AWS Cloud9가 실행을 시도하는 명령입니다.

빌드

환경의 빌더 경로

모든 사용자 지정 빌드 구성에 대한 경로입니다.

실행 및 디버그

환경의 러너 경로

모든 사용자 지정 실행 구성에 대한 경로입니다.

미리 보기 URL

환경의 애플리케이션을 미리 보는 데 사용할 URL입니다.

실행 구성

이 환경에 대한 사용자 지정 실행 구성입니다.

선택한 구성 제거

선택한 실행 구성을 삭제합니다.

새 구성 추가

새 실행 구성을 생성합니다.

기본값으로 설정

선택한 실행 구성을 기본 실행 구성으로 설정합니다.

코드 포맷터

JSBeautify 설정

파일에서 코드의 가독성을 높이기위한 설정입니다.

저장 시 코드 포맷

선택할 경우, AWS Cloud9는 코드 파일이 저장될 때마다 JSBeautify 설정을 적용하려고 시도합니다.

JavaScript에 JSBeautify를 사용

선택할 경우, AWS Cloud9는 JavaScript 파일이 저장될 때마다 JSBeautify 설정을 적용하려고 시도합니다.

빈 줄 유지

선택할 경우 AWS Cloud9는 코드 파일에서 빈 줄을 제거하지 않습니다.

배열 들여쓰기 유지

선택할 경우 AWS Cloud9이 코드 파일의 배열에서 요소 선언의 들여쓰기를 보존합니다.

JSLint 강제 공백

선택할 경우 AWS Cloud9이 JSLint 공백 규칙을 코드 파일에 적용하려고 시도합니다. 자세한 내용은 JSLint 도움말에서 '공백'을 참조하세요.

중괄호

코드에서 중괄호 정렬을 지정합니다.

유효한 값은 다음과 같습니다.

- 제어 문의 중괄호 - 각 시작 및 끝 중괄호를 이동하여 필요에 따라 관련 제어 문과 정렬합니다.

  예를 들어, 이 코드의 형식은 다음과 같습니다.

  ```
  for (var i = 0; i < 10; i++) { if (i == 5) { console.log("Halfway done.") }}
  ```

  파일이 저장될 때 이 코드로 바뀝니다.

  ```
  for (var i = 0; i < 10; i++) {
     if (i == 5) {
        console.log("Halfway done.")
     }
  }
  ```

- 중괄호 새 줄로 이동 - 필요에 따라 각 중괄호를 새 줄로 이동합니다.

  예를 들어, 이 코드의 형식은 다음과 같습니다.

  ```
  for (var i = 0; i < 10; i++) { if (i == 5) { console.log("Halfway done.") }}
  ```

  파일이 저장될 때 이 코드로 바뀝니다.

  ```
  for (var i = 0; i < 10; i++) {if (i == 5)
    {
       console.log("Halfway done.")
    }
    }
  ```

- 끝 중괄호 새 줄로 이동 - 필요에 따라 각 끝 중괄호를 새 줄로 이동합니다.

  예를 들어, 이 코드의 형식은 다음과 같습니다.

```
for (var i = 0; i < 10; i++) {
  if (i == 5) { console.log("Halfway done.") }
}
```

파일이 저장될 때 이 코드로 바뀝니다.

```
for (var i = 0; i < 10; i++) {
   if (i == 5) {
      console.log("Halfway done.")
   }
}
```

인라인 블록 유지

선택할 경우 AWS Cloud9는 중괄호가 같은 줄에 있으면 인라인 블록의 시작 및 끝 중괄호를 별도의 줄로 옮기려고 시도하지 않습니다.

조건 앞에 공백 추가

선택할 경우 AWS Cloud9이 필요에 따라 각 조건부 선언 앞에 공백을 추가합니다.

문자열 이스케이프 해제

선택할 경우 AWS Cloud9이 이스케이프된 문자열을 이스케이프되지 않은 문자열로 변환합니다. 예를 들어 `\n`를 개행 문자로 변환하고 `\r`를 캐리지 리턴 문자로 변환합니다.

들여쓰기 내부 HTML

선택할 경우 AWS Cloud9이 HTML 코드의 `<head>` 및 `<body>` 섹션을 들여씁니다.

TypeScript 지원

저장 시 코드 포맷

선택할 경우, AWS Cloud9는 TypeScript 파일이 저장될 때마다 TypeScript 코드의 포맷을 시도합니다.

사용자 지정 코드 포맷터

TypeScript 코드에 대한 사용자 지정 코드 포맷 구성의 경로입니다.

PHP 지원

PHP 코드 완성 사용

선택할 경우, AWS Cloud9는 PHP 코드를 완성하려고 시도합니다.

PHP 완성 포함 경로

PHP 코드 완성을 돕기 위해 AWS Cloud9가 사용하는 위치입니다. 예를 들어 AWS Cloud9가 코드 완성에 사용하도록 할 사용자 지정 PHP 파일이 `~/environment` 디렉터리에 있는 경우 이 경로에 `~/environment`를 추가합니다.

저장 시 코드 포맷

선택할 경우, AWS Cloud9는 PHP 파일이 저장될 때마다 PHP 코드의 포맷을 시도합니다.

사용자 지정 코드 포맷터

PHP 코드에 대한 사용자 지정 코드 포맷 구성의 경로입니다.

Python 지원

Python 코드 완성 사용

선택할 경우, AWS Cloud9는 Python 코드를 완성하려고 시도합니다. AWS Cloud9가 Python 코드를 완성하는 데 사용할 경로를 설정하려면 PYTHONPATH 설정을 사용합니다.

Python 버전

사용할 Python 버전을 지정합니다.

Pylint 명령줄 옵션

AWS Cloud9가 Python 코드와 함께 Pylint에 사용할 옵션입니다. 자세한 내용은 Pylint 웹 사이트에서 Pylint 사용자 설명서를 참조하세요.

PYTHONPATH

AWS Cloud9가 사용할 Python 라이브러리 및 패키지의 경로입니다. 예를 들어 사용자 지정 Python 라이브러리와 패키지가 `~/environment` 디렉터리에 있는 경우 `~/environment`를 이 경로에 추가합니다.

저장 시 코드 포맷

선택할 경우, AWS Cloud9는 Python 파일이 저장될 때마다 Python 코드의 포맷을 시도합니다.

사용자 지정 코드 포맷터

Python 코드에 대한 사용자 지정 코드 포맷 구성의 경로입니다.

Go 지원

Go 코드 완성 사용

선택할 경우, AWS Cloud9는 Go 코드를 완성하려고 시도합니다.

저장 시 코드 포맷

선택할 경우, AWS Cloud9는 Go 파일이 저장될 때마다 Go 코드의 포맷을 시도합니다.

사용자 지정 코드 포맷터

Go 코드에 대한 사용자 지정 코드 포맷 구성의 경로입니다.

수동으로 환경의 EC2 인스턴스 중지

이 EC2 인스턴스 설정을 사용하면 IDE에 연결된 모든 웹 브라우저 인스턴스를 닫고 나서 30분 후에 환경의 Amazon EC2 인스턴스를 자동으로 중지할 수 있습니다.

콘솔을 사용하여 즉시 인스턴스를 수동으로 중지할 수도 있습니다.

환경의 EC2 인스턴스를 수동으로 중지하려면

1. IDE에 연결된 모든 웹 브라우저 인스턴스를 닫은 후 AWS Cloud9 콘솔에서 Your environments(환경)를 선택합니다.
2. 창의 오른쪽 상단에서, 사용 중인 환경의 세부 정보를 표시하는 단추를 선택하고 [세부 정보 보기(View details)]를 선택합니다.
3. [환경 세부 정보(Environment details)]의 [EC2 인스턴스(EC2 Instance)]에서 [인스턴스로 이동(Go To Instance)]을 선택합니다.
4. Amazon EC2 콘솔의 Instance state(인스턴스 상태)에서 확인란을 선택하여 환경의 인스턴스를 선택합니다. 인스턴스 상태는 인스턴스가 아직 실행 중임을 나타낼 수 있습니다.
5. [인스턴스 상태(Instance state)], [인스턴스 중지(Stop instance)]를 차례로 선택합니다.
6. 확인 메시지가 표시되면 [중지(Stop)]를 선택합니다. 인스턴스가 중지하는 데 몇 분 정도 걸릴 수 있습니다.

AWS Cloud9 IDE의 사용자 설정 작업

사용자 설정은 AWS Identity and Access Management(IAM) 사용자와 연결된 각 AWS Cloud9 개발 환경에서 적용되는 설정입니다. 여기에는 다음 설정이 포함됩니다.

- 일반 사용자 인터페이스 설정(예: 애니메이션 활성화 및 변경된 탭 표시)
- 파일 시스템 탐색 설정
- 파일 찾기 및 검색 설정
- 터미널 세션 및 출력의 색 구성표
- 추가 코드 편집기 설정(예: 글꼴 크기, 코드 접기, 전체 줄 선택, 스크롤 애니메이션 및 글꼴 크기)

사용자 설정을 변경하면 AWS Cloud9가 그러한 변경 사항을 클라우드에 푸시하고 IAM 사용자와 연결합니다. 또한 AWS Cloud9는 IAM 사용자와 연결된 사용자 설정에 변경 사항이 있는지 확인하기 위해 클라우드를 계속 검사하고, 그러한 설정을 현재 환경에 적용합니다. 이를 사용하면 어떤 AWS Cloud9 환경에서 작업 중이든 상관없이 동일한 모양과 느낌을 경험할 수 있습니다.

> **Note**
>
> IDE 설정을 저장하고 검색하기 위해 AWS Cloud9은 내부 API `GetUserSettings` 및 `UpdateUserSettings`를 사용합니다.

다음과 같이 사용자 설정을 다른 사용자와 공유할 수 있습니다.

- 사용자 설정 보기 또는 변경
- 다른 사용자와 사용자 설정 공유
- 사용자가 변경할 수 있는 사용자 설정

사용자 설정 보기 또는 변경

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. 소유한 각 환경의 사용자 설정을 보려면 기본 설정 탭의 측면에 있는 탐색 창에서 User Settings(사용자 설정)를 선택합니다.
3. User Settings(사용자 설정) 창에서 각 환경의 사용자 설정을 변경합니다.

4. 소유한 다른 환경에 변경 사항을 적용하려면 해당 환경을 열면 됩니다. 해당 환경이 이미 열려 있는 경우, 해당 환경의 웹 브라우저 탭을 새로 고칩니다.

자세한 내용은 사용자가 변경할 수 있는 사용자 설정을 참조하십시오.

다른 사용자와 사용자 설정 공유

1. 소스 환경과 대상 환경의 AWS Cloud9 IDE 메뉴 모음에서 AWS Cloud9, Open Your User Settings(사용자 설정 열기)를 선택합니다.
2. 소스 환경에서 화면에 표시된 user.settings 탭의 콘텐츠를 복사합니다.
3. 대상 환경에서 user.settings 탭의 콘텐츠를 소스 환경에서 복사한 콘텐츠로 덮어씁니다.
4. 대상 환경에서 user.settings 탭을 저장합니다.

사용자가 변경할 수 있는 사용자 설정

이 단원에서는 Preferences(기본 설정) 탭의 User Settings(사용자 설정) 창에서 변경할 수 있는 사용자 설정 종류를 설명합니다.

- 일반
- 사용자 인터페이스
- 협업
- Tree 및 Go 패널
- 파일에서 찾기
- 메타데이터
- 감시자
- 터미널
- 출력
- 코드 편집기(Ace)
- 입력
- 힌트 및 경고
- 실행 및 디버그
- 평가판
- 빌드

일반

Reset to Factory Settings(초기 기본 설정으로 재설정)

Reset to Default(기본값으로 재설정) 버튼이 선택되어 있는 경우, AWS Cloud9는 모든 사용자 설정을 AWS Cloud9 기본 사용자 설정으로 재설정합니다. 확인하려면 Reset settings(설정 재설정)를 선택합니다.

> **Warning**
>
> 이 작업을 취소할 수 없습니다.

Warn Before Exiting(종료 전 경고)

IDE를 닫으려고 할 때마다 AWS Cloud9에서 종료할지 확인하는 메시지를 표시합니다.

사용자 인터페이스

Enable UI Animations(UI 애니메이션 활성화)

AWS Cloud9는 IDE에서 애니메이션을 사용합니다.

Use an Asterisk (*) to Mark Changed Tabs(별표(*)를 사용하여 변경된 탭 표시)

AWS Cloud9는 변경 사항이 있지만 콘텐츠가 아직 저장되지 않은 탭에 별표(*)를 추가합니다.

Display Title of Active Tab as Browser Title(활성 탭의 제목을 브라우저 제목으로 표시)

AWS Cloud9는 연결된 웹 브라우저 탭의 제목을 활성 탭의 제목으로 변경합니다(예: Untitled1, hello.js, 터미널, 기본 설정 등).

Automatically Close Empty Panes(비어 있는 창을 자동으로 닫기)

사용자가 환경을 다시 로드할 때마다 AWS Cloud9이 비어 있다고 간주하는 창을 자동으로 닫습니다.

Environment Files Icon and Selection Style(환경 파일 아이콘 및 선택 스타일)

AWS Cloud9이 환경 파일에 사용하는 아이콘과 AWS Cloud9이 사용하는 파일 선택 동작.

유효한 값으로는 다음이 포함됩니다.

- 기본값(Default) - AWS Cloud9이 기본 아이콘과 기본 파일 선택 동작을 사용합니다.
- 대체(Alternative) – AWS Cloud9이 대체 아이콘과 대체 파일 선택 동작을 사용합니다.

협업

Disable collaboration security warning(협업 보안 경고 비활성화)

환경에 읽기/쓰기 멤버가 추가될 때 AWS Cloud9는 보안 경고 대화 상자를 표시하지 않습니다.

Show Authorship Info(저작권 정보 표시)

AWS Cloud9는 다른 환경 멤버가 입력한 텍스트에 밑줄을 치고 거터에 관련 강조 표시를 합니다.

Tree 및 Go 패널

Scope Go to Anything to Favorites(바로 가기 범위를 즐겨찾기로 지정)

[이동(Go)] 창의 [파일로 이동(Go to File)]에 [환경(Environment)] 창의 [즐겨찾기(Favorites)]에 한정된 결과가 표시됩니다.

Enable Preview on Tree Selection(트리 선택에서 미리 보기 활성화)

AWS Cloud9이 두 번 클릭 대신에 한 번 클릭으로 선택한 파일을 표시합니다.

Hidden File Pattern(숨겨진 파일 패턴)

AWS Cloud9가 숨겨짐으로 처리할 파일 유형.

Reveal Active File in Project Tree(프로젝트 트리에서 활성 파일 표시)

AWS Cloud9가 환경 창에서 활성 파일을 강조 표시합니다.

Download Files As(다운로드 파일 다른 이름으로 저장)

AWS Cloud9가 파일을 다운로드할 때 사용할 동작.

유효한 값은 다음과 같습니다.

- 자동(auto) – AWS Cloud9이 수정 없이 파일을 다운로드합니다.
- tar.gz – AWS Cloud9는 파일을 압축된 TAR 파일로 다운로드합니다.
- zip – AWS Cloud9는 파일을 .zip 파일로 다운로드합니다.

Find in Files(파일에서 찾기)

Search In This Path When 'Project' Is Selected('프로젝트'가 선택된 경우 이 경로에서 검색)

Find in Files(파일에서 찾기) 모음에서 프로젝트가 검색 범위로 선택되면 검색에 사용할 경로.

Show Full Path in Results(결과에 전체 경로 표시)

선택한 경우 Search Results(검색 결과) 탭에서 일치하는 각 파일의 전체 경로를 표시합니다.

Clear Results Before Each Search(각 검색 전에 결과 지우기)

현재 검색을 시작하기 전에 이전 검색 결과의 Search Results(검색 결과) 탭을 지웁니다.

Scroll Down as Search Results Come In(검색 결과가 나타나면 아래로 스크롤)

검색 결과가 확인되면 Search Results(검색 결과) 탭을 결과 목록의 맨 아래로 스크롤합니다.

Open Files when Navigating Results with (Up and Down)(위쪽 및 아래쪽으로 결과를 탐색할 때 파일 열기)

결과 목록 내의 Search Results(검색 결과) 탭에서 위쪽 및 아래쪽 화살표 키를 누르면 일치하는 각 파일을 엽니다.

메타데이터

Maximum of Undo Stack Items in Meta Data(메타데이터에서 실행 취소 스택 항목의 최대 수)

AWS Cloud9가 실행 취소할 수 있는 작업 목록에 유지하는 최대 항목 수.

감시자

Auto-Merge Files When a Conflict Occurs(충돌이 발생하면 파일 자동 병합)

병합 충돌이 발생할 때마다 AWS Cloud9가 파일을 자동으로 병합하려고 시도합니다.

터미널

텍스트 색

터미널 탭의 텍스트 색.

배경 색상

터미널 탭의 배경 색상.

Selection Color(선택 색상)

터미널 탭에서 선택한 텍스트 색.

Font Family(글꼴 그룹)

터미널 탭의 텍스트 글꼴 스타일.

글꼴 크기

터미널 탭의 텍스트 크기.

Antialiased Fonts(앤티 앨리어싱된 글꼴)

AWS Cloud9가 터미널 탭의 텍스트 표시를 부드럽게 만들려고 시도합니다.

Blinking Cursor(깜박이는 커서)

AWS Cloud9가 터미널 탭에서 커서를 계속 깜박입니다.

Scrollback(스크롤백)

터미널 탭에서 위로 또는 뒤로 스크롤할 수 있는 행 수.

AWS Cloud9을 기본 편집기로 사용

AWS Cloud9를 기본 문자 텍스트 편집기로 사용합니다.

출력

텍스트 색

출력을 표시하는 탭의 텍스트 색.

배경 색상

출력을 표시하는 탭의 텍스트 배경 색상.

Selection Color(선택 색상)

출력을 표시하는 탭에서 선택한 텍스트 색.

Warn Before Closing Unnamed Configuration(이름 없는 구성을 닫기 전에 경고)

AWS Cloud9가 저장하지 않은 구성 탭을 닫기 전에 저장할 것인지 여부를 묻습니다.

Preserve log between runs(실행 간 로그 보존)

AWS Cloud9가 시도된 모든 실행의 로그를 유지합니다.

코드 편집기(Ace)

Auto-pair Brackets, Quotes, etc.(대괄호, 따옴표 등의 쌍 자동 완성)

AWS Cloud9가 편집기 탭에 입력된 각 관련 시작 문자에 대해 부합하는 닫는 문자를 추가하려고 시도합니다(예: 대괄호, 인용 부호, 중괄호).

Wrap Selection with Brackets, Quote, etc.(대괄호, 따옴표 등으로 선택 영역 묶기)

텍스트가 선택되고 관련 시작 문자가 입력된 후 AWS Cloud9가 편집기 탭의 텍스트 끝에 부합하는 닫는 문자를 삽입하려고 시도합니다(예: 대괄호, 인용 부호, 중괄호).

Code Folding(코드 접기)

AWS Cloud9가 관련 코드 구문 규칙에 따라 편집기 탭에서 코드 섹션을 표시, 확장, 숨기기 또는 축소하려고 시도합니다.

Fade Fold Widgets(접기 위젯 투명 표시)

사용자가 편집기 탭의 코드 접기 컨트롤 위에서 마우스를 멈출 때마다 AWS Cloud9가 거터에 코드 접기 컨트롤을 표시합니다.

선택 영역이 비어 있는 상태로 복사

AWS Cloud9을 사용하면 텍스트 복사 및 잘라내기가 가능하며 이 옵션은 빈 텍스트를 클립보드에 복사할지 여부를 결정합니다.

Full Line Selection(행 전체 선택)

AWS Cloud9가 편집기 탭에서 세 번 클릭되는 행 전체를 선택합니다.

Highlight Active Line(현재 행 강조 표시)

AWS Cloud9가 편집기 탭에서 현재 행 전체를 강조 표시합니다.

Highlight Gutter Line(여백 줄 강조 표시)

AWS Cloud9가 편집기 탭에서 현재 행 옆의 거터 위치를 강조 표시합니다.

Show Invisible Characters(보이지 않는 문자 표시)

AWS Cloud9가 편집기 탭에서 보이지 않는 문자로 간주하는 문자를 표시합니다(예: 캐리지 리턴 및 줄 바꿈, 공백, 탭).

Show Gutter(여백 표시)

AWS Cloud9가 거터를 표시합니다.

Show Line Numbers(행 번호 표시)

여백에 행 번호를 표시하기 위한 동작.

유효한 값은 다음과 같습니다.

- 정상(Normal) – 행 번호를 표시합니다.
- 상대(Relative) – 현재 행을 기준으로 행 번호를 표시합니다.
- 없음(None) – 행 번호를 숨깁니다.

Show Indent Guides(들여쓰기 가이드 표시)

AWS Cloud9가 편집기 탭에서 들여쓴 텍스트를 더 쉽게 시각화할 수 있도록 가이드를 표시합니다.

Highlight Selected Word(선택한 단어 강조 표시)

AWS Cloud9가 편집기 탭에서 두 번 클릭되는 단어 전체를 선택합니다.

Scroll Past the End of the Document(문서 끝을 지나 스크롤)

사용자가 편집기 탭에서 현재 파일의 끝을 지나 스크롤할 수 있게 하는 동작.

유효한 값은 다음과 같습니다.

- 해제(Off) – 현재 파일의 끝을 지나 스크롤하는 것을 허용하지 않습니다.
- 편집기 높이 절반(Half Editor Height) – 현재 파일의 끝을 지나 편집기 화면 높이의 절반까지 스크롤할 수 있습니다.
- 전체 편집기 높이(Full Editor Height) – 현재 파일의 끝을 지나 편집기의 전체 화면 높이까지 스크롤할 수 있습니다.

Animate Scrolling(스크롤할 때 애니메이션 효과 적용)

AWS Cloud9가 편집기 탭에서 작업을 스크롤하는 동안 애니메이션 동작을 적용합니다.

Font Family(글꼴 그룹)

편집기 탭에서 사용할 글꼴 스타일.

글꼴 크기

편집기 탭에서 사용할 글꼴 크기.

Antialiased Fonts(앤티 앨리어싱된 글꼴)

AWS Cloud9가 편집기 탭의 텍스트 표시를 부드럽게 만들려고 시도합니다.

Show Print Margin(인쇄 여백 표시)

편집기 탭에서 지정된 문자 위치 뒤에 세로선을 표시합니다.

Mouse Scroll Speed(마우스 스크롤 속도)

편집기 탭에서 마우스 스크롤의 상대 속도. 값이 클수록 스크롤 속도가 빨라집니다.

Cursor Style(커서 스타일)

편집기 탭에서 포인터의 스타일과 동작.

유효한 값으로는 다음이 포함됩니다.

- Ace – Slim(가늘게)보다 넓은 세로 막대로 포인터를 표시합니다.
- Slim(가늘게) – 상대적으로 가는 세로 막대로 포인터를 표시합니다.
- Smooth(부드럽게) – Slim(가늘게)보다 넓고 Slim(가늘게)보다 더 부드럽게 깜박이는 세로 막대로 포인터를 표시합니다.
- Smooth and Slim(부드럽고 얇게) – Slim(가늘게)보다 부드럽게 깜박이며 상대적으로 가는 세로 막대로 포인터를 표시합니다.
- Wide(넓게) – 상대적으로 넓은 세로 막대로 포인터를 표시합니다.

Merge Undo Deltas(델타 병합 실행 취소)

- 항상(Always) – 병합 충돌을 되돌리도록 허용합니다.
- 안 함(Never) – 병합 충돌을 되돌리는 것을 허용하지 않습니다.
- Timed(일정 시간) – 지정된 시간 후에 병합 충돌을 되돌리도록 허용합니다.

Enable Wrapping For New Documents(새 문서에 줄 바꿈 사용)

AWS Cloud9는 코드를 새 파일에 래핑합니다.

입력

Complete As You Type(입력 시 자동 완성)

입력과 동시에 AWS Cloud9가 가능한 텍스트로 완성시켜 표시하려고 시도합니다.

Complete On Enter(Enter로 자동 완성)

Enter 키를 누른 후 AWS Cloud9이 가능한 텍스트로 완성시켜 표시하려고 시도합니다.

Highlight Variable Under Cursor(커서 아래의 변수 강조 표시)

AWS Cloud9가 선택한 변수에 대한 코드의 모든 참조를 강조 표시합니다.

Use Cmd-Click for Jump to Definition(Cmd 키를 클릭하여 정의로 이동)

AWS Cloud9가 Mac의 Command 또는 Windows의 Ctrl을 계속 누른 상태로 유지하면서 코드의 원래 정의로 이동합니다.

힌트 및 경고

Enable Hints and Warnings(힌트 및 경고 활성화)

AWS Cloud9가 해당하는 힌트 및 경고 메시지를 표시합니다.

클릭 시 사용 가능한 빠른 수정 사항 표시

코드 내에서 키워드를 클릭하면 AWS Cloud9은 리팩터링 제안과 함께 도구 설명을 표시합니다.

Ignore Messages Matching Regex(정규식과 일치하는 메시지 무시)

AWS Cloud9가 지정된 정규식과 일치하는 메시지를 표시하지 않습니다. 자세한 내용은 단원을 참조하십시오.정규식 패턴 작성의JavaScript 정규식모질라 개발자 네트워크에 대한 주제를 참조하세요.

실행 및 디버그

Save All Unsaved Tabs Before Running(실행 전에 저장되지 않은 모든 탭 저장)

연결된 코드를 실행하기 전에 AWS Cloud9가 열린 탭으로 저장되지 않은 모든 파일을 저장하려고 시도합니다.

미리 보기

Preview Running Apps(실행 중인 앱 미리 보기)

[미리 보기(Preview)] 버튼이 선택될 때마다 AWS Cloud9이 활성 탭에 코드 출력의 미리 보기를 표시하려고 시도합니다.

Default Previewer(기본 미리 보기)

AWS Cloud9가 코드 출력을 미리 보는 데 사용하는 형식.

유효한 값으로는 다음이 포함됩니다.

- 원시(Raw) – 코드 출력을 일반 형식으로 표시하려고 시도합니다.
- Browser(브라우저) – 코드 출력을 웹 브라우저의 기본 형식으로 표시하려고 시도합니다.

When Saving Reload Previewer(저장 시 미리 보기 다시 표시)

코드 파일을 저장할 때마다 AWS Cloud9가 코드 출력 미리 보기를 표시하기 위해 사용하는 동작입니다.

유효한 값은 다음과 같습니다.

- Ctrl-Enter를 누를 경우에만(Only on Ctrl-Enter) – 현재 코드 탭에 대해 Ctrl+Enter를 누를 때마다 코드 출력 미리 보기를 표시하려고 시도합니다.
- 항상(Always) – 코드 파일이 저장될 때마다 코드 출력을 미리 보기를 표시하려고 시도합니다.

빌드

Automatically Build Supported Files(지원되는 파일 자동 빌드)

빌드 작업이 시작되고 코드가 지원되는 형식인 경우 AWS Cloud9가 현재 코드를 자동으로 빌드하려고 시도합니다.

AWS Cloud9 통합 개발 환경(IDE)의 AWS 프로젝트 및 사용자 설정 작업

Preferences(기본 설정) 탭의 AWS Settings(설정) 창에 있는 AWS 서비스 설정에는 다음과 같은 종류의 설정이 포함됩니다.

- AWS Resources(리소스) 창에 사용할 AWS 리전
- AWS 관리형 임시 자격 증명 사용 여부
- AWS Serverless Application Model(AWS SAM) 템플릿 편집기를 일반 텍스트로 표시할지, 시각적 모드로 표시할지 여부

이러한 설정을 보거나 변경하려면 환경에 대한 IDE의 메뉴 모음에서 AWS Cloud9, Preferences(AC9, 기본 설정)을 선택합니다.

다음 목록에서 프로젝트 수준 설정은 현재 AWS Cloud9 개발 환경에만 적용됩니다. 반대로 사용자 수준 설정은 IAM 사용자와 연결된 각 환경에 적용됩니다. 자세한 내용은 Apply the Current Project Settings for an Environment to Another Environment 및 Share Your User Settings with Another User 단원을 참조하십시오.

- 프로젝트 수준 설정

- 사용자 수준 설정

프로젝트 수준 설정

AWS 리전

AWS Resources(리소스) 창의 Lambda 섹션에 사용할 AWS 리전.

AWS 관리형 임시 보안 인증 정보

켜져 있는 경우 환경의 AWS CLI, AWS CloudShell 또는 AWS SDK 코드에서 AWS 서비스를 호출할 때 AWS 관리형 임시 보안 인증 정보를 사용합니다. 자세한 내용은 AWS 관리형 임시 자격 증명을 참조하세요.

사용자 수준 설정

AWS SAM 시각적 편집기 사용

켜져 있는 경우 AWS Resources(리소스) 창의 Lambda 섹션을 사용할 때 시각적 모드로 AWS Serverless Application Model(AWS SAM) 템플릿 편집기를 표시합니다. 꺼진 경우 편집기를 텍스트 모드로 표시합니다.

AWS Cloud9 통합 개발 환경(IDE)의 키 바인딩 작업

키 바인딩은 바로 가기 키 조합을 정의합니다. 키 바인딩은 IAM 사용자와 연결된 각 AWS Cloud9 개발 환경에 적용됩니다. 키 바인딩을 변경하면 AWS Cloud9가 그러한 변경 사항을 클라우드에 푸시하고 IAM 사용자와 연결합니다. 또한 AWS Cloud9는 IAM 사용자와 연결된 키 바인딩에 변경 사항이 있는지 클라우드를 지속적으로 검사하고, 그러한 변경 사항을 현재 환경에 적용합니다.

키 바인딩을 다른 사용자와 공유할 수 있습니다.

- 키 바인딩 보기 또는 변경
- 다른 사용자와 키 바인딩 공유
- 키보드 모드 변경
- 운영 체제 키 바인딩 변경
- 특정 키 바인딩 변경
- 모든 사용자 지정 키 바인딩 제거

키 바인딩 보기 또는 변경

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. 소유한 각 환경의 키 바인딩을 보려면 [기본 설정(Preferences)] 탭의 측면에 있는 탐색 창에서 [키 바인딩(Keybindings)]를 선택합니다.
3. 각 환경에서 키 바인딩을 변경하려면 Keybindings(키 바인딩) 창에서 원하는 설정을 변경합니다.
4. 환경에 변경 사항을 적용하려면 해당 환경을 열면 됩니다. 해당 환경이 이미 열려 있는 경우, 해당 환경의 웹 브라우저 탭을 새로 고칩니다.

자세한 내용은 다음을 참조하세요.

- MacOS 기본 키 바인딩 참조
- MacOS Vim 키 바인딩 참조
- MacOS Emacs 키 바인딩 참조
- MacOS Sublime 키 바인딩 참조
- Windows/Linux 기본 키 바인딩 참조
- Windows/Linux Vim 키 바인딩 참조
- Windows/Linux Emacs 키 바인딩 참조
- Windows/Linux Sublime 키 바인딩 참조

다른 사용자와 키 바인딩 공유

1. 소스 환경과 대상 환경의 AWS Cloud9 IDE 메뉴 모음에서 AWS Cloud9, Open Your Keymap(키맵 열기)을 선택합니다.
2. 소스 환경에서 화면에 표시된 keybindings.settings 탭의 콘텐츠를 복사합니다.
3. 대상 환경에서 keybindings.settings 탭의 콘텐츠를 소스 환경에서 복사한 콘텐츠로 덮어씁니다.
4. 대상 환경에서 keybindings.settings 탭을 저장합니다.

키보드 모드 변경

AWS Cloud9 IDE가 IAM 사용자와 연결된 각 환경에서 편집기의 텍스트와 상호 작용하는 데 사용하는 키보드 모드를 변경할 수 있습니다.

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. [기본 설정(Preferences)] 탭 측면의 탐색 창에서 [키 바인딩(Keybindings)]을 선택합니다.
3. [키보드 모드(Keyboard Mode)]에서 다음 키보드 모드 중 하나를 선택합니다.
   - 기본 키 바인딩 세트를 사용하려면 [기본값(Default)]을 선택합니다.
   - Vim 모드를 사용하려면 [Vim]을 선택합니다. 자세한 내용은 Vim 도움말 파일 웹 사이트를 참조하세요.
   - Emacs 모드를 사용하려면 [Emacs]를 선택합니다. 자세한 내용은 GNU 운영 체제 웹 사이트에서 Emacs 편집기를 참조하세요.
   - Sublime 모드를 사용하려면 [Sublime]을 선택합니다. 자세한 내용은 Sublime 텍스트 문서 웹 사이트를 참조하세요.

운영 체제 키 바인딩 변경

AWS Cloud9 IDE가 IAM 사용자와 연결된 각 환경에서 인식하는 운영 체제 키 바인딩 세트를 변경할 수 있습니다.

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. [기본 설정(Preferences)] 탭 측면의 탐색 창에서 [키 바인딩(Keybindings)]을 선택합니다.
3. [운영 체제(Operating System)]에서 다음 운영 체제 중 하나를 선택합니다.
   - [자동(Auto)] - AWS Cloud9 IDE가 사용할 운영 체제 키 바인딩 집합을 감지하려고 시도합니다.
   - MacOS - AWS Cloud9 IDE가 macOS 형식으로 나열된 키 바인딩을 사용합니다.
   - Windows/Linux - AWS Cloud9 IDE가 Windows 및 Linux 형식으로 나열된 키 바인딩을 사용합니다.

특정 키 바인딩 변경

IAM 사용자와 연결된 각 환경에서 개별 키 바인딩을 변경할 수 있습니다.

키 바인딩을 한 번에 하나씩 변경하려면

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. [기본 설정(Preferences)] 탭 측면의 탐색 창에서 [키 바인딩(Keybindings)]을 선택합니다.
3. 키 바인딩 목록의 Keystroke(키 입력) 열에서 변경하려는 키 바인딩을 엽니다(두 번 클릭).
4. 키보드를 사용하여 대체 키 조합을 지정한 다음 `Enter` 키를 누릅니다.

> **Note**
>
> 현재 키 조합을 완전히 제거하려면 Windows 또는 Linux의 경우 `Backspace` 키를 누르고 macOS의 경우 `Delete` 키를 누릅니다.

한 번에 여러 개의 키 바인딩을 변경하려면

1. 메뉴 모음에서 [AWS Cloud9], [키맵 열기(Open Your Keymap)]를 선택합니다.
2. `keybindings.settings` 파일에서 변경할 각 키 바인딩을 정의합니다. 다음은 구문의 예제입니다.

```
[
  {
    "command": "addfavorite",
    "keys": {
      "win": ["Ctrl-Alt-F"],
      "mac": ["Ctrl-Option-F"]
    }
  },
  {
    "command": "copyFilePath",
    "keys": {
      "win": ["Ctrl-Shift-F"],
      "mac": ["Alt-Shift-F"]
    }
  }
]
```

이 예에서 `addFavorite` 및 `copyFilePath`는 [기본 설정(Preferences)] 탭에서 [키 바인딩(Keybindings)] 창의 [키 입력(Keystroke)] 열에 있는 키 바인딩의 이름입니다. Windows 또는 Linux와 macOS의 경우에서 원하는 키 바인딩은 각각 `win`과 `mac`입니다.

변경 내용을 적용하려면 `keybindings.settings` 파일을 저장합니다. 변경 내용이 잠시 후 Keybindings(키 바인딩) 창에 표시됩니다.

모든 사용자 지정 키 바인딩 제거

IAM 사용자와 연결된 각 환경에서 모든 사용자 지정 키 바인딩을 제거하고 모든 키 바인딩을 기본값으로 복원할 수 있습니다.

> ⚠ Warning
> 이 작업은 취소할 수 없습니다.

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. [기본 설정(Preferences)] 탭 측면의 탐색 창에서 [키 바인딩(Keybindings)]을 선택합니다.
3. [기본값으로 재설정(Reset to defaults)]을 선택합니다.

AWS Cloud9 통합 개발 환경(IDE)의 테마 작업

테마는 전체 IDE 색상을 정의합니다. 이는 IAM 사용자와 연결된 각 AWS Cloud9 개발 환경에 적용됩니다. 테마를 변경하면 AWS Cloud9가 그러한 변경 사항을 클라우드에 푸시하고 IAM 사용자와 연결합니다. 또한 AWS Cloud9는 IAM 사용자와 연결된 테마에 변경 사항이 있는지 클라우드를 지속적으로 검사하고, AWS Cloud9는 그러한 변경 사항을 현재 환경에 적용합니다.

- 테마 보기 또는 변경하기
- 변경할 수 있는 전체 테마 설정
- 테마 재정의

테마 보기 또는 변경

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. 소유한 각 환경의 테마를 보려면 [기본 설정(Preferences)] 탭의 측면에 있는 탐색 창에서 [테마(Themes)]를 선택합니다.
3. 각 환경에서 테마를 변경하려면 [테마(Themes)] 창에서 원하는 설정을 변경합니다. 코드를 사용하여 테마의 일부를 변경하려면 스타일시트 링크를 선택합니다.
4. 환경에 변경 사항을 적용하려면 해당 환경을 엽니다. 해당 환경이 이미 열려 있는 경우, 해당 환경의 웹 브라우저 탭을 새로 고칩니다.

변경할 수 있는 전체 테마 설정

[기본 설정] 탭의 테마(Themes)] 창에서 다음과 같은 유형의 전체 테마 설정을 변경할 수 있습니다.

플랫 테마

AWS Cloud9 IDE 전체에 기본 제공 플랫 테마를 적용합니다.

클래식 테마

AWS Cloud9 IDE 전체에 선택한 기본 제공 클래식 테마를 적용합니다.

구문 테마

AWS Cloud9 IDE 전체의 코드 파일에 선택한 테마를 적용합니다.

테마 재정의

Important

AWS Cloud9은 사용자가 `styles.css` 파일을 업데이트하여 IDE 테마를 재정의할 수 있는 기능을 더 이상 지원하지 않습니다.엽니다. 사용자는 계속해서 편집기를 사용하여 `styles.css` 파일을 보고, 편집하고, 저장할 수 있습니다. 하지만 AWS Cloud9 IDE가 로드될 때 테마 오버라이드는 적용되지 않습니다.
AWS Cloud9이 `styles.css` 파일이 수정된 것을 감지하면 IDE에 다음 메시지가 표시됩니다. 테마 재정의에 대한 지원이 중단되었습니다. 이 styles.css 파일의 내용은 더 이상 AWS Cloud9 IDE를 로드할 때 적용되지 않습니다.
스타일시트를 사용하여 IDE의 테마를 정의해야 하는 경우 당사에 직접 문의하세요.

AWS Cloud9 통합 개발 환경(IDE)에서 초기화 스크립트 관리

Important

AWS Cloud9에서는 사용자가 초기화 스크립트를 사용자 지정할 수 있는 실험 기능을 더 이상 지원하지 않습니다. 이 스크립트는 IDE에서 자동으로 실행되었습니다. 사용자는 계속해서 편집기를 사용하여 `init.js` 파일을 보고, 편집하고, 저장할 수 있습니다. 하지만 사용자 지정된 초기화 스크립트는 더 이상 실행할 수 없으며 IDE의 동작을 수정할 수 없습니다.
AWS Cloud9이 `init.js` 파일이 수정된 것을 감지하면 IDE에 다음 메시지가 표시됩니다.

초기화 스크립트에 대한 지원이 중단되었습니다.(Support for initialization scripts has been discontinued.) 이 init.js 파일의 내용은 더 이상 AWS Cloud9 IDE를 로드할 때 실행되지 않습니다.
IDE에 대한 사용자 지정 초기화 스크립트를 실행해야 하는 경우 당사에 문의하세요.

초기화 스크립트는 모든 플러그인이 로드된 후 IDE에서 실행할 초기화 코드를 정의합니다. 이는 IAM 사용자와 연결된 각 AWS Cloud9 개발 환경에 적용됩니다. AWS Cloud9는 초기화 스크립트에 대한 변경 사항을 지속적으로 검색하고 수정이 발생한 경우 사용자에게 알립니다.

초기화 스크립트를 열기

초기화 스크립트를 열려면 메뉴 모음에서 [AWS Cloud9],]초기화 스크립트 열기(Open Your Init Script)]를 선택합니다.

Important

편집기를 사용하여 `init.js` 파일을 편집하고 저장할 수 있지만 사용자 지정 스크립트는 IDE에서 실행되지 않습니다.

AWS Cloud9 통합 개발 환경(IDE)에 대한 macOS 기본 키 바인딩 참조

다음은 AWS Cloud9 IDE의 macOS 운영 체제용 기본 키보드 모드 키 바인딩 목록입니다.

자세한 내용은 AWS Cloud9 IDE에서 다음을 참조하세요.

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. [기본 설정(Preferences)] 탭에서 [키 바인딩(Keybindings)]을 선택합니다.
3. [키보드 모드(Keyboard Mode)]에서 [기본값(Default)]을 선택합니다.
4. [운영 체제(Operating system)]에서 [MacOS]를 선택합니다.

키 바인딩 작업도 참조하세요.

- 일반

- 탭
- 패널
- 코드 편집기
- emmet
- 터미널
- 실행 및 디버그

일반

설명	키 바인딩	명령
선택 영역을 시계 표현식으로 추가합니다.	Command-Shift-C	addwatchfromselection
클립보드에서 잘라낸 선택 영역을 제거합니다.	Esc	clearcut
코드 완성 컨텍스트 메뉴를 표시합니다.	Control-Space | Option-Space	complete
코드를 완료한 다음 덮어씁니다.	Control-Shift-Space | Option-Shift-Space	completeoverwrite
선택 영역을 클립보드로 복사합니다.	Command-C	copy
선택 영역을 클립보드로 잘라냅니다.	Command-X	cut
코드를 확장합니다(해당하는 경우).	Tab	expandSnippet
현재 문서의 찾기 및 바꾸기 막대를 표시합니다.	Command-F	find
현재 문서에서 일치하는 항목을 모두 선택합니다.	Control-Option-G	findAll

설명	키 바인딩	명령
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 다음 항목으로 이동합니다.	Command-G	findnext
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 이전 항목으로 이동합니다.	Command-Shift-G	findprevious
편집기에서 활성 파일의 삽입 점에 기호에 대한 알려진 모든 참조를 표시합니다.	Shift-F3	findReferences
[환경(Environment) 창을 연 다음 파일 목록을 활성 상태로 만듭니다.	Shift-Esc	focusTree
선택한 JavaScript 코드를 다시 포맷합니다.	Command-Shift-B	formatcode
줄로 이동 상자를 표시합니다.	Command-L	gotoline
찾기 및 바꾸기 막대가 표시된 경우 숨깁니다.	Esc	hidesearchreplace
커서 위치에 있는 변수 또는 함수의 정의로 이동합니다.	F3	jumptodef
AWS 리소스 창의 Lambda 섹션에서 로컬 Lambda 함수를 선택한 경우 해당 함수를 원격 함수로 Lambda에 업로드하려고 시도합니다.	Command-Shift-U	lambdaUploadFunction
새 파일을 만듭니다.	Control-N	newfile
[기본 설정(Preferences)] 탭을 표시합니다.	Command-,	openpreferences

설명	키 바인딩	명령
[터미널(Terminal)] 탭을 연 다음 파일 목록에서 선택한 파일의 상위 폴더로 전환합니다.	Command-Option-L	opentermhere
클립보드의 현재 내용을 커서 위치에 붙여 넣습니다.	Command-V	paste
오류 수정에 대한 제안 사항을 표시합니다.	Command-F3	quickfix
마지막 작업을 다시 실행합니다.	Command-Shift-Z | Command-Y	redo
미리 보기 창을 새로 고칩니다.	Command-Enter	reloadpreview
선택 항목의 이름 바꾸기 리팩터링을 시작합니다.	Option-Command-R	renameVar
현재 문서의 찾기 및 바꾸기 모음을 표시하고 [바꾸기(Replace With)] 표현식을 사용합니다.	Option-Command-F	replace
초기화 스크립트를 다시 실행합니다.	Command-Enter	rerunInitScript
환경을 다시 시작합니다.	Command-R	restartc9
현재 파일을 마지막으로 저장한 버전으로 재설정합니다.	Control-Shift-Q	reverttosaved
열려 있는 각 파일을 저장된 버전으로 재설정합니다.	Option-Shift-Q	reverttosavedall
현재 파일을 디스크에 저장합니다.	Command-S	save

설명	키 바인딩	명령
현재 파일을 다른 파일 이름을 사용하여 디스크에 저장합니다.	Command-Shift-S	saveas
여러 파일의 찾기 및 바꾸기 모음을 표시합니다.	Shift-Command-F	searchinfiles
[프로세스 목록(Process List)] 대화 상자를 표시합니다.	Command-Option-P	showprocesslist
마지막 작업을 실행 취소합니다.	Command-Z	undo

탭

설명	키 바인딩	명령
현재 창에서 현재 탭을 제외하고 열려 있는 탭을 모두 닫습니다.	Option-Control-W	closeallbutme
모든 창에서 열려 있는 탭을 모두 닫습니다.	Option-Shift-W	closealltabs
현재 창을 닫습니다.	Command-Control-W	closepane
현재 탭을 닫습니다.	Option-W	closetab
아래로 창을 하나 이동합니다.	Control-Command-Down	gotopanedown
왼쪽으로 창을 하나 이동합니다.	Control-Command-Left	gotopaneleft
오른쪽으로 창을 하나 이동합니다.	Control-Command-Right	gotopaneright

설명	키 바인딩	명령
위로 창을 하나 이동합니다.	Control-Command-Up	gottopaneup
왼쪽으로 한 탭 이동합니다.	Command-[	gototableft
오른쪽으로 한 탭 이동합니다.	Command-]	gototabright
현재 탭을 한 창 아래로 이동하거나 탭이 이미 맨 아래에 있는 경우 분할 탭을 만듭니다.	Command-Option-Shift-Down	movetabdown
현재 탭을 왼쪽으로 이동하거나 탭이 이미 맨 왼쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Command-Option-Shift-Left	movetableft
현재 탭을 오른쪽으로 이동하거나 탭이 이미 맨 오른쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Command-Option-Shift-Right	movetabright
현재 탭을 한 창 위로 이동하거나 탭이 이미 맨 위에 있는 경우 분할 탭을 만듭니다.	Command-Option-Shift-Up	movetabup
다음 창으로 이동합니다.	Option-Esc	nextpane
다음 탭으로 이동합니다.	Option-Tab	nexttab
이전 창으로 이동합니다.	Option-Shift-Esc	previouspane
이전 탭으로 이동합니다.	Option-Shift-Tab	previoustab
마지막 탭으로 돌아갑니다.	Esc	refocusTab
마지막 탭을 다시 엽니다.	Option-Shift-T	reopenLastTab
파일 트리에 현재 탭을 표시합니다.	Command-Shift-L	revealtab

설명	키 바인딩	명령
10번째 탭으로 이동합니다.	`Command-0`	`tab0`
첫 번째 탭으로 이동합니다.	`Command-1`	`tab1`
두 번째 탭으로 이동합니다.	`Command-2`	`tab2`
세 번째 탭으로 이동합니다.	`Command-3`	`tab3`
네 번째 탭으로 이동합니다.	`Command-4`	`tab4`
다섯 번째 탭으로 이동합니다.	`Command-5`	`tab5`
여섯 번째 탭으로 이동합니다.	`Command-6`	`tab6`
일곱 번째 탭으로 이동합니다.	`Command-7`	`tab7`
여덟 번째 탭으로 이동합니다.	`Command-8`	`tab8`
아홉 번째 탭으로 이동합니다.	`Command`	`tab9`

패널

설명	키 바인딩	명령
[바로 가기(Go to Anything)] 모드로 [이동(Go)] 창을 표시합니다.	`Command-E|Command-P`	`gotoanything`
[명령으로 이동(Go to Command)] 모드로 [이동(Go)] 창을 표시합니다.	`Command-.|F1`	`gotocommand`
[파일로 이동(Go to File)] 모드로 [이동(Go)] 창을 표시합니다.	`Command-O`	`gotofile`

설명	키 바인딩	명령
[기호로 이동(Go to Symbol)] 모드로 [이동(Go)] 창을 표시합니다.	`Command-Shift-O`	`gotosymbol`
[개요(Outline)] 창을 표시합니다.	`Command-Shift-E`	`outline`
숨겨져 있는 경우 [콘솔(Console)] 창을 표시하거나 표시된 경우 숨깁니다.	`Control-Esc`	`toggleconsole`
숨겨져 있는 경우 [환경(Environment)] 창을 표시하거나 표시된 경우 숨깁니다.	`Command-U`	`toggletree`

코드 편집기

설명	키 바인딩	명령
활성 커서 한 줄 위에 커서를 추가하거나 커서가 이미 추가된 경우 해당 커서 위에 다른 커서를 추가합니다.	`Control-Option-Up`	`addCursorAbove`
활성 커서 한 줄 위에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 위로 이동합니다.	`Control-Option-Shift-Up`	`addCursorAboveSkipCurrent`
활성 커서 한 줄 아래에 커서를 추가하거나 커서가 이미 추가된 경우 해당 커서 아래에 다른 커서를 추가합니다.	`Control-Option-Down`	`addCursorBelow`

설명	키 바인딩	명령
활성 커서 한 줄 아래에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 아래로 이동합니다.	`Control-Option-Shift-Down`	`addCursorBelowSkipCurrent`
정렬이 잘못된 경우 모든 커서를 각 행의 활성 커서와 동일한 공간으로 이동합니다.	`Control-Option-A`	`alignCursors`
단일 공백을 백스페이스합니다.	`Control-Backspace | Shift-Backspace | Backspace`	`backspace`
선택 영역을 한 탭 들여씁니다.	`Control-]`	`blockindent`
선택 영역을 한 탭 내어씁니다.	`Control-[`	`blockoutdent`
포커스를 편집기에서 IDE의 다른 곳으로 전환할 수 있는지 여부를 제어합니다.	`Command-Z | Command-Shift-Z | Command-Y`	`cancelBrowserUndoInAce`
선택 영역을 가운데로 정렬합니다.	`Control-L`	`centerselection`
줄의 내용을 복사하고 복사한 내용을 한 줄 아래에 붙여 넣습니다.	`Command-Option-Down`	`copylinesdown`
줄의 내용을 복사하고 복사한 내용을 한 줄 위에 붙여 넣습니다.	`Command-Option-Up`	`copylinesup`
공백을 하나 삭제합니다.	`Delete | Control-Delete | Shift-Delete`	`del`

설명	키 바인딩	명령
선택 영역의 내용을 복사하고 복사한 내용을 선택 영역 바로 뒤에 붙여넣습니다.	Command-Shift-D	duplicateSelection
선택 영역에 현재 줄의 내용을 포함합니다.	Command-Shift-L	expandtoline
선택 영역에 일치하는 다음 기호까지의 내용을 포함합니다.	Control-Shift-M	expandToMatching
선택한 코드를 접거나 접힌 단위가 선택된 경우 펼칩니다.	Command-Option-L | Command-F1	fold
접을 수 있는 모든 요소를 접습니다.	Control-Command-Option-0	foldall
현재 선택 범위를 제외하고 접을 수 있는 모든 요소를 접습니다.	Command-Option-0	foldOther
한 줄 아래로 이동합니다.	Down | Control-N	golinedown
한 줄 위로 이동합니다.	Up | Control-P	golineup
로그 파일의 끝 부분으로 이동합니다.	Command-End | Command-Down	gotoend
한 칸씩 왼쪽으로 이동합니다.	Left | Control-B	gotoleft
현재 줄의 끝으로 이동합니다.	Command-Right | End | Control-E	gotolineend
현재 줄의 처음으로 이동합니다.	Command-Left | Home | Control-A	gotolinestart
다음 오류로 이동합니다.	F4	goToNextError
한 페이지 아래로 이동합니다.	Page Down | Control-V	gotopagedown

설명	키 바인딩	명령
한 페이지 위로 이동합니다.	Page Up	gotopageup
이전 오류로 이동합니다.	Shift-F4	goToPreviousError
오른쪽으로 공백 하나 이동합니다.	Right | Control-F	gotoright
로그 파일의 시작 부분으로 이동합니다.	Command-Home | Command-Up	gotostart
왼쪽으로 한 단어 이동합니다.	Option-Left	gotowordleft
오른쪽으로 한 단어 이동합니다.	Option-Right	gotowordright
선택 영역을 한 탭 들여씁니다.	Tab	indent
현재 범위에서 쌍을 이루는 기호로 이동합니다.	Control-P	jumptomatching
글꼴 크기를 늘립니다.	Command-+ | Command-=	largerfont
커서의 왼쪽에 있는 숫자를 1씩 줄입니다(숫자인 경우).	Option-Shift-Down	modifyNumberDown
커서의 왼쪽에 있는 숫자를 1씩 늘립니다(숫자인 경우).	Option-Shift-Up	modifyNumberUp
선택 영역을 한 줄 아래로 이동합니다.	Option-Down	movelinesdown
선택 영역을 한 줄 위로 이동합니다.	Option-Up	movelinesup
선택 영역을 한 탭 내어씁니다.	Shift-Tab	outdent
덮어쓰기 모드를 켜거나 켜진 경우 끕니다.	Insert	overwrite

설명	키 바인딩	명령
한 페이지 아래로 이동합니다.	Option-Page Down	pagedown
한 페이지 위로 이동합니다.	Option-Page Up	pageup
현재 줄을 제거합니다.	Command-D	removeline
커서에서 현재 줄의 끝까지 삭제합니다.	Control-K	removetolineend
커서에서 현재 줄의 시작까지 삭제합니다.	Command-Backspace	removetolinestart
커서의 왼쪽에 있는 단어를 삭제합니다.	Option-Backspace | Control-Option-Backspace	removewordleft
커서의 오른쪽에 있는 단어를 삭제합니다.	Option-Delete	removewordright
이전에 기록한 키 입력을 재생합니다.	Command-Shift-E	replaymacro
선택 가능한 내용을 모두 선택합니다.	Command-A	selectall
선택 영역에 다음 줄 아래의 내용을 포함합니다.	Shift-Down | Control-Shift-N	selectdown
선택 영역에 다음 공백 왼쪽의 내용을 포함합니다.	Shift-Left | Control-Shift-B	selectleft
커서부터 현재 줄의 나머지 부분을 선택 영역에 포함합니다.	Shift-End	selectlineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Shift-Home	selectlinestart

설명	키 바인딩	명령
선택 영역의 뒤에 일치하는 선택 영역을 더 포함합니다.	Control-Option-Right	selectMoreAfter
선택 영역의 앞에 일치하는 선택 영역을 더 포함합니다.	Control-Option-Left	selectMoreBefore
선택 영역의 뒤에 있는 일치하는 다음 선택 영역을 포함합니다.	Control-Option-Shift-Right	selectNextAfter
선택 영역의 앞에 있는 일치하는 다음 선택 영역을 포함합니다.	Control-Option-Shift-Left	selectNextBefore
일치하는 다음 선택 항목을 선택하거나 찾습니다.	Control-G	selectOrFindNext
일치하는 이전 선택 항목 선택하거나 찾습니다.	Control-Shift-G	selectOrFindPrevious
커서에서 현재 페이지의 끝까지 선택 영역에 포함합니다.	Shift-Page Down	selectpagedown
커서에서 현재 페이지의 처음까지 선택 영역에 포함합니다.	Shift-Page Up	selectpageup
선택 영역에 커서 오른쪽의 다음 공백를 포함합니다.	Shift-Right	selectright
커서에서 현재 파일의 끝까지 선택 영역에 포함합니다.	Command-Shift-End | Command-Shift-Down	selecttoend
커서에서 현재 줄의 끝까지 선택 영역에 포함합니다.	Command-Shift-Right | Shift-End | Control-Shift-E	selecttolineend

설명	키 바인딩	명령
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Command-Shift-Left | Control-Shift-A	selecttolinestart
커서에서 현재 범위의 일치하는 다음 기호까지 포함합니다.	Control-Shift-P	selecttomatching
커서에서 현재 파일의 처음까지 선택 영역에 포함합니다.	Command-Shift-Home | Command-Shift-Up	selecttostart
선택 영역에 다음 줄 위의 내용을 포함합니다.	Shift-Up | Control-Shift-Up	selectup
선택 영역에 커서 왼쪽의 다음 단어를 포함합니다.	Option-Shift-Left	selectwordleft
선택 영역에 커서 오른쪽의 다음 단어를 포함합니다.	Option-Shift-Right	selectwordright
[기본 설정(Preferences)] 탭을 표시합니다.	Command-,	showSettingsMenu
이전 선택 영역을 모두 지웁니다.	Esc	singleSelection
글꼴 크기를 줄입니다.	Command--	smallerfont
여러 줄을 선택한 경우 정렬된 순서로 다시 정렬합니다.	Command-Option-S	sortlines
현재 줄의 끝에 커서를 추가합니다.	Control-Option-L	splitIntoLines
커서의 내용을 줄의 끝에 새 줄로 이동합니다.	Control-O	splitline
블록 주석 문자로 선택 영역을 둘러싸거나 해당 문자가 있는 경우 제거합니다.	Command-Shift-/	toggleBlockComment

설명	키 바인딩	명령
선택한 각 줄의 처음에 줄 설명 문자를 추가하거나, 줄 설명 문자가 있는 경우 제거합니다.	Command-/	togglecomment
코드를 접거나, 코드 접기를 제거합니다(있는 경우).	F2	toggleFoldWidget
상위 코드를 접거나, 접기를 제거합니다(있는 경우).	Option-F2	toggleParentFoldWidget
키 입력 기록을 시작하거나 이미 기록 중인 경우 중지합니다.	Command-Option-E	togglerecording
단어를 줄 바꿈하거나 이미 줄 바꿈이 있는 경우 단어 줄 바꿈을 중지합니다.	Control-W	toggleWordWrap
선택 영역을 모두 소문자로 변경합니다.	Control-Shift-U	tolowercase
선택 영역을 모두 대문자로 변경합니다.	Control-U	touppercase
선택 영역의 대소문자를 바꿉니다.	Control-T	transposeletters
선택한 코드를 펼칩니다.	Command-Option-Shift-L | Command-Shift-F1	unfold
전체 파일의 코드 접기를 펼칩니다.	Command-Option-Shift-0	unfoldall

emmet

설명	키 바인딩	명령
간단한 수학 표현식(예: 2*4 또는 10/2)과 그 결과를 출력합니다.	Shift-Command-Y	emmet_evaluate_math_expression
현재 파일의 구문에 따라 CSS와 유사한 약어를 HTML, XML 또는 CSS 코드로 확장합니다.	Control-Option-E	emmet_expand_abbreviation
탭 정지에 의해 확장된 CSS와 유사한 약어를 순회합니다.	Tab	emmet_expand_abbreviation_with_tab
편집 가능한 다음 코드 부분으로 이동합니다.	Shift-Command-.	emmet_select_next_item
이전 편집 가능한 코드 부분으로 이동합니다.	Shift-Command-,	emmet_select_previous_item
약어를 확장한 다음 생성된 코드 조각의 마지막 요소 내에 현재 선택 영역을 배치합니다.	Shift-Control-A	emmet_wrap_with_abbreviation

터미널

설명	키 바인딩	명령
새 [터미널(Terminal)] 탭을 엽니다.	Option-T	openterminal
편집기와 [터미널(Terminal)] 탭 간을 전환합니다.	Option-S	switchterminal

실행 및 디버깅

설명	키 바인딩	명령
현재 파일을 빌드합니다.	Command-B	build
현재 일시 중지된 프로세스를 다시 시작합니다.	F8 | Command-\	resume
현재 애플리케이션을 실행하거나 디버그합니다.	Option-F5	run
마지막 실행 파일을 실행하거나 디버그합니다.	F5	runlast
스택 옆에 있는 함수로 진행합니다.	F11 | Command-;	stepinto
현재 함수 범위를 종료합니다.	Shift-F11 | Command-Shift-'	stepout
스택의 현재 표현식으로 진행합니다.	F10 | Command-'	stepover
현재 애플리케이션의 실행 또는 디버깅을 중지합니다.	Shift-F5	stop
현재 파일의 빌드를 중지합니다.	Control-Shift-C	stopbuild

AWS Cloud9 통합 개발 환경(IDE)에 대한 macOS Vim 키 바인딩 참조

다음은 AWS Cloud9 IDE의 macOS 운영 체제용 Vim 키보드 모드 키 바인딩 목록입니다.

자세한 내용은 AWS Cloud9 IDE에서 다음을 참조하세요.

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.

2. [기본 설정(Preferences)] 탭에서 [키 바인딩(Keybindings)]을 선택합니다.

3. 용키보드 모드를 선택하고이맥스.

4. [운영 체제(Operating system)]에서 [MacOS]를 선택합니다.

키 바인딩 작업도 참조하세요.

- 일반
- 탭
- 패널
- 코드 편집기
- emmet
- 터미널
- 실행 및 디버그

일반

설명	키 바인딩	명령
선택 영역을 시계 표현식으로 추가합니다.	`Command-Shift-C`	`addwatchfromselection`
클립보드에서 잘라낸 선택 영역을 제거합니다.	`Esc`	`clearcut`
코드 완성 컨텍스트 메뉴를 표시합니다.	`Control-Space` | `Option-Space`	`complete`
코드를 완료한 다음 덮어씁니다.	`Control-Shift-Space` | `Option-Shift-Space`	`completeoverwrite`
선택 영역을 클립보드로 복사합니다.	`Command-C`	`copy`
선택 영역을 클립보드로 잘라냅니다.	`Command-X`	`cut`

설명	키 바인딩	명령
코드를 확장합니다(해당하는 경우).	Tab	expandSnippet
현재 문서의 찾기 및 바꾸기 막대를 표시합니다.	Command-F	find
현재 문서에서 일치하는 항목을 모두 선택합니다.	Control-Option-G	findAll
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 다음 항목으로 이동합니다.	Command-G	findnext
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 이전 항목으로 이동합니다.	Command-Shift-G	findprevious
편집기에서 활성 파일의 삽입점에 기호에 대한 알려진 모든 참조를 표시합니다.	Shift-F3	findReferences
[환경(Environment) 창을 연 다음 파일 목록을 활성 상태로 만듭니다.	Shift-Esc	focusTree
선택한 JavaScript 코드를 다시 포맷합니다.	Command-Shift-B	formatcode
줄로 이동 상자를 표시합니다.	Command-L	gotoline
찾기 및 바꾸기 막대가 표시된 경우 숨깁니다.	Esc	hidesearchreplace
커서 위치에 있는 변수 또는 함수의 정의로 이동합니다.	F3	jumptodef

설명	키 바인딩	명령
AWS 리소스 창의 Lambda 섹션에서 로컬 Lambda 함수를 선택한 경우 해당 함수를 원격 함수로 Lambda에 업로드하려고 시도합니다.	`Command-Shift-U`	`lambdaUploadFunction`
새 파일을 만듭니다.	`Control-N`	`newfile`
[기본 설정(Preferences)] 탭을 표시합니다.	`Command-,`	`openpreferences`
[터미널(Terminal)] 탭을 연 다음 파일 목록에서 선택한 파일의 상위 폴더로 전환합니다.	`Command-Option-L`	`opentermhere`
클립보드의 현재 내용을 커서 위치에 붙여 넣습니다.	`Command-V`	`paste`
오류 수정에 대한 제안 사항을 표시합니다.	`Command-F3`	`quickfix`
마지막 작업을 다시 실행합니다.	`Command-Shift-Z` | `Command-Y`	`redo`
미리 보기 창을 새로 고칩니다.	`Command-Enter`	`reloadpreview`
선택 항목의 이름 바꾸기 리팩터링을 시작합니다.	`Option-Command-R`	`renameVar`
현재 문서의 찾기 및 바꾸기 모음을 표시하고 [바꾸기(Replace With)] 표현식을 사용합니다.	`Option-Command-F`	`replace`
초기화 스크립트를 다시 실행합니다.	`Command-Enter`	`rerunInitScript`

설명	키 바인딩	명령
환경을 다시 시작합니다.	Command-R	restartc9
현재 파일을 마지막으로 저장한 버전으로 재설정합니다.	Control-Shift-Q	reverttosaved
열려 있는 각 파일을 저장된 버전으로 재설정합니다.	Option-Shift-Q	reverttosavedall
현재 파일을 디스크에 저장합니다.	Command-S	save
현재 파일을 다른 파일 이름을 사용하여 디스크에 저장합니다.	Command-Shift-S	saveas
여러 파일의 찾기 및 바꾸기 모음을 표시합니다.	Shift-Command-F	searchinfiles
[프로세스 목록(Process List)] 대화 상자를 표시합니다.	Command-Option-P	showprocesslist
마지막 작업을 실행 취소합니다.	Command-Z	undo

탭

설명	키 바인딩	명령
현재 창에서 현재 탭을 제외하고 열려 있는 탭을 모두 닫습니다.	Option-Control-W	closeallbutme
모든 창에서 열려 있는 탭을 모두 닫습니다.	Option-Shift-W	closealltabs
현재 창을 닫습니다.	Command-Control-W	closepane

설명	키 바인딩	명령
현재 탭을 닫습니다.	Option-W	closetab
아래로 창을 하나 이동합니다.	Control-Command-Down	gotopanedown
왼쪽으로 창을 하나 이동합니다.	Control-Command-Left	gotopaneleft
오른쪽으로 창을 하나 이동합니다.	Control-Command-Right	gotopaneright
위로 창을 하나 이동합니다.	Control-Command-Up	gottopaneup
왼쪽으로 한 탭 이동합니다.	Command-[	gototableft
오른쪽으로 한 탭 이동합니다.	Command-]	gototabright
현재 탭을 한 창 아래로 이동하거나 탭이 이미 맨 아래에 있는 경우 분할 탭을 만듭니다.	Command-Option-Shift-Down	movetabdown
현재 탭을 왼쪽으로 이동하거나 탭이 이미 맨 왼쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Command-Option-Shift-Left	movetableft
현재 탭을 오른쪽으로 이동하거나 탭이 이미 맨 오른쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Command-Option-Shift-Right	movetabright
현재 탭을 한 창 위로 이동하거나 탭이 이미 맨 위에 있는 경우 분할 탭을 만듭니다.	Command-Option-Shift-Up	movetabup
다음 창으로 이동합니다.	Option-Esc	nextpane
다음 탭으로 이동합니다.	Option-Tab	nexttab

설명	키 바인딩	명령
이전 창으로 이동합니다.	Option-Shift-Esc	previouspane
이전 탭으로 이동합니다.	Option-Shift-Tab	previoustab
마지막 탭으로 돌아갑니다.	Esc	refocusTab
마지막 탭을 다시 엽니다.	Option-Shift-T	reopenLastTab
파일 트리에 현재 탭을 표시합니다.	Command-Shift-L	revealtab
10번째 탭으로 이동합니다.	Command-0	tab0
첫 번째 탭으로 이동합니다.	Command-1	tab1
두 번째 탭으로 이동합니다.	Command-2	tab2
세 번째 탭으로 이동합니다.	Command-3	tab3
네 번째 탭으로 이동합니다.	Command-4	tab4
다섯 번째 탭으로 이동합니다.	Command-5	tab5
여섯 번째 탭으로 이동합니다.	Command-6	tab6
일곱 번째 탭으로 이동합니다.	Command-7	tab7
여덟 번째 탭으로 이동합니다.	Command-8	tab8
아홉 번째 탭으로 이동합니다.	Command	tab9

패널

설명	키 바인딩	명령
[바로 가기(Go to Anything)] 모드로 [이동(Go)] 창을 표시합니다.	`Command-E|Command-P`	`gotoanything`
[명령으로 이동(Go to Command)] 모드로 [이동(Go)] 창을 표시합니다.	`Command-.|F1`	`gotocommand`
[파일로 이동(Go to File)] 모드로 [이동(Go)] 창을 표시합니다.	`Command-O`	`gotofile`
[기호로 이동(Go to Symbol)] 모드로 [이동(Go)] 창을 표시합니다.	`Command-Shift-O`	`gotosymbol`
[개요(Outline)] 창을 표시합니다.	`Command-Shift-E`	`outline`
숨겨져 있는 경우 [콘솔(Console)] 창을 표시하거나 표시된 경우 숨깁니다.	`Control-Esc`	`toggleconsole`
숨겨져 있는 경우 [환경(Environment)] 창을 표시하거나 표시된 경우 숨깁니다.	`Command-U`	`toggletree`

코드 편집기

설명	키 바인딩	명령
활성 커서 한 줄 위에 커서를 추가하거나 커서가 이미 추가된	`Control-Option-Up`	`addCursorAbove`

설명	키 바인딩	명령
경우 해당 커서 위에 다른 커서를 추가합니다.		
활성 커서 한 줄 위에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 위로 이동합니다.	`Control-Option-Shift-Up`	`addCursorAboveSkipCurrent`
활성 커서 한 줄 아래에 커서를 추가하거나 커서가 이미 추가된 경우 해당 커서 아래에 다른 커서를 추가합니다.	`Control-Option-Down`	`addCursorBelow`
활성 커서 한 줄 아래에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 아래로 이동합니다.	`Control-Option-Shift-Down`	`addCursorBelowSkipCurrent`
정렬이 잘못된 경우 모든 커서를 각 행의 활성 커서와 동일한 공간으로 이동합니다.	`Control-Option-A`	`alignCursors`
단일 공백을 백스페이스합니다.	`Control-Backspace | Shift-Backspace | Backspace`	`backspace`
선택 영역을 한 탭 들여씁니다.	`Control-]`	`blockindent`
선택 영역을 한 탭 내어씁니다.	`Control-[`	`blockoutdent`
포커스를 편집기에서 IDE의 다른 곳으로 전환할 수 있는지 여부를 제어합니다.	`Command-Z | Command-Shift-Z | Command-Y`	`cancelBrowserUndoInAce`
선택 영역을 가운데로 정렬합니다.	`Control-L`	`centerselection`

설명	키 바인딩	명령
줄의 내용을 복사하고 복사한 내용을 한 줄 아래에 붙여 넣습니다.	Command-Option-Down	copylinesdown
줄의 내용을 복사하고 복사한 내용을 한 줄 위에 붙여 넣습니다.	Command-Option-Up	copylinesup
공백을 하나 삭제합니다.	Delete | Control-Delete | Shift-Delete	del
선택 영역의 내용을 복사하고 복사한 내용을 선택 영역 바로 뒤에 붙여넣습니다.	Command-Shift-D	duplicateSelection
선택 영역에 현재 줄의 내용을 포함합니다.	Command-Shift-L	expandtoline
선택 영역에 일치하는 다음 기호까지의 내용을 포함합니다.	Control-Shift-M	expandToMatching
선택한 코드를 접거나 접힌 단위가 선택된 경우 펼칩니다.	Command-Option-L | Command-F1	fold
접을 수 있는 모든 요소를 접습니다.	Control-Command-Option-0	foldall
현재 선택 범위를 제외하고 접을 수 있는 모든 요소를 접습니다.	Command-Option-0	foldOther
한 줄 아래로 이동합니다.	Down | Control-N	golinedown
한 줄 위로 이동합니다.	Up | Control-P	golineup
로그 파일의 끝 부분으로 이동합니다.	Command-End | Command-Down	gotoend

설명	키 바인딩	명령
한 칸씩 왼쪽으로 이동합니다.	`Left | Control-B`	`gotoleft`
현재 줄의 끝으로 이동합니다.	`Command-Right | End | Control-E`	`gotolineend`
현재 줄의 처음으로 이동합니다.	`Command-Left | Home | Control-A`	`gotolinestart`
다음 오류로 이동합니다.	`F4`	`goToNextError`
한 페이지 아래로 이동합니다.	`Page Down | Control-V`	`gotopagedown`
한 페이지 위로 이동합니다.	`Page Up`	`gotopageup`
이전 오류로 이동합니다.	`Shift-F4`	`goToPreviousError`
오른쪽으로 공백 하나 이동합니다.	`Right | Control-F`	`gotoright`
로그 파일의 시작 부분으로 이동합니다.	`Command-Home | Command-Up`	`gotostart`
왼쪽으로 한 단어 이동합니다.	`Option-Left`	`gotowordleft`
오른쪽으로 한 단어 이동합니다.	`Option-Right`	`gotowordright`
선택 영역을 한 탭 들여씁니다.	`Tab`	`indent`
현재 범위에서 쌍을 이루는 기호로 이동합니다.	`Control-P`	`jumptomatching`
글꼴 크기를 늘립니다.	`Command-+ | Command-=`	`largerfont`
커서의 왼쪽에 있는 숫자를 1씩 줄입니다(숫자인 경우).	`Option-Shift-Down`	`modifyNumberDown`
커서의 왼쪽에 있는 숫자를 1씩 늘립니다(숫자인 경우).	`Option-Shift-Up`	`modifyNumberUp`

설명	키 바인딩	명령
선택 영역을 한 줄 아래로 이동합니다.	Option-Down	movelinesdown
선택 영역을 한 줄 위로 이동합니다.	Option-Up	movelinesup
선택 영역을 한 탭 내어씁니다.	Shift-Tab	outdent
덮어쓰기 모드를 켜거나 켜진 경우 끕니다.	Insert	overwrite
한 페이지 아래로 이동합니다.	Option-Page Down	pagedown
한 페이지 위로 이동합니다.	Option-Page Up	pageup
현재 줄을 제거합니다.	Command-D	removeline
커서에서 현재 줄의 끝까지 삭제합니다.	Control-K	removetolineend
커서에서 현재 줄의 시작까지 삭제합니다.	Command-Backspace	removetolinestart
커서의 왼쪽에 있는 단어를 삭제합니다.	Option-Backspace | Control-Option-Backspace	removewordleft
커서의 오른쪽에 있는 단어를 삭제합니다.	Option-Delete	removewordright
이전에 기록한 키 입력을 재생합니다.	Command-Shift-E	replaymacro
선택 가능한 내용을 모두 선택합니다.	Command-A	selectall
선택 영역에 다음 줄 아래의 내용을 포함합니다.	Shift-Down | Control-Shift-N	selectdown

설명	키 바인딩	명령
선택 영역에 다음 공백 왼쪽의 내용을 포함합니다.	Shift-Left | Control-Shift-B	selectleft
커서부터 현재 줄의 나머지 부분을 선택 영역에 포함합니다.	Shift-End	selectlineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Shift-Home	selectlinestart
선택 영역의 뒤에 일치하는 선택 영역을 더 포함합니다.	Control-Option-Right	selectMoreAfter
선택 영역의 앞에 일치하는 선택 영역을 더 포함합니다.	Control-Option-Left	selectMoreBefore
선택 영역의 뒤에 있는 일치하는 다음 선택 영역을 포함합니다.	Control-Option-Shift-Right	selectNextAfter
선택 영역의 앞에 있는 일치하는 다음 선택 영역을 포함합니다.	Control-Option-Shift-Left	selectNextBefore
일치하는 다음 선택 항목을 선택하거나 찾습니다.	Control-G	selectOrFindNext
일치하는 이전 선택 항목 선택하거나 찾습니다.	Control-Shift-G	selectOrFindPrevious
커서에서 현재 페이지의 끝까지 선택 영역에 포함합니다.	Shift-Page Down	selectpagedown
커서에서 현재 페이지의 처음까지 선택 영역에 포함합니다.	Shift-Page Up	selectpageup
선택 영역에 커서 오른쪽의 다음 공백를 포함합니다.	Shift-Right	selectright

설명	키 바인딩	명령
커서에서 현재 파일의 끝까지 선택 영역에 포함합니다.	Command-Shift-End | Command-Shift-Down	selecttoend
커서에서 현재 줄의 끝까지 선택 영역에 포함합니다.	Command-Shift-Right | Shift-End | Control-Shift-E	selecttolineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Command-Shift-Left | Control-Shift-A	selecttolinestart
커서에서 현재 범위의 일치하는 다음 기호까지 포함합니다.	Control-Shift-P	selecttomatching
커서에서 현재 파일의 처음까지 선택 영역에 포함합니다.	Command-Shift-Home | Command-Shift-Up	selecttostart
선택 영역에 다음 줄 위의 내용을 포함합니다.	Shift-Up | Control-Shift-P	selectup
선택 영역에 커서 왼쪽의 다음 단어를 포함합니다.	Option-Shift-Left	selectwordleft
선택 영역에 커서 오른쪽의 다음 단어를 포함합니다.	Option-Shift-Right	selectwordright
[기본 설정(Preferences)] 탭을 표시합니다.	Command-,	showSettingsMenu
이전 선택 영역을 모두 지웁니다.	Esc	singleSelection
글꼴 크기를 줄입니다.	Command--	smallerfont
여러 줄을 선택한 경우 정렬된 순서로 다시 정렬합니다.	Command-Option-S	sortlines
현재 줄의 끝에 커서를 추가합니다.	Control-Option-L	splitIntoLines

설명	키 바인딩	명령
커서의 내용을 줄의 끝에 새 줄로 이동합니다.	Control-O	splitline
블록 주석 문자로 선택 영역을 둘러싸거나 해당 문자가 있는 경우 제거합니다.	Command-Shift-/	toggleBlockComment
선택한 각 줄의 처음에 줄 설명 문자를 추가하거나, 줄 설명 문자가 있는 경우 제거합니다.	Command-/	togglecomment
코드를 접거나, 코드 접기를 제거합니다(있는 경우).	F2	toggleFoldWidget
상위 코드를 접거나, 접기를 제거합니다(있는 경우).	Option-F2	toggleParentFoldWidget
키 입력 기록을 시작하거나 이미 기록 중인 경우 중지합니다.	Command-Option-E	togglerecording
단어를 줄 바꿈하거나 이미 줄 바꿈이 있는 경우 단어 줄 바꿈을 중지합니다.	Control-W	toggleWordWrap
선택 영역을 모두 소문자로 변경합니다.	Control-Shift-U	tolowercase
선택 영역을 모두 대문자로 변경합니다.	Control-U	touppercase
선택 영역의 대소문자를 바꿉니다.	Control-T	transposeletters
선택한 코드를 펼칩니다.	Command-Option-Shift-L | Command-Shift-F1	unfold

설명	키 바인딩	명령
전체 파일의 코드 접기를 펼칩니다.	Command-Option-Shift-0	unfoldall

emmet

설명	키 바인딩	명령
간단한 수학 표현식(예: 2*4 또는 10/2)과 그 결과를 출력합니다.	Shift-Command-Y	emmet_evaluate_math_expression
현재 파일의 구문에 따라 CSS와 유사한 약어를 HTML, XML 또는 CSS 코드로 확장합니다.	Control-Option-E	emmet_expand_abbreviation
탭 정지에 의해 확장된 CSS와 유사한 약어를 순회합니다.	Tab	emmet_expand_abbreviation_with_tab
편집 가능한 다음 코드 부분으로 이동합니다.	Shift-Command-.	emmet_select_next_item
이전 편집 가능한 코드 부분으로 이동합니다.	Shift-Command-,	emmet_select_previous_item
약어를 확장한 다음 생성된 코드 조각의 마지막 요소 내에 현재 선택 영역을 배치합니다.	Shift-Control-A	emmet_wrap_with_abbreviation

터미널

설명	키 바인딩	명령
새 [터미널(Terminal)] 탭을 엽니다.	Option-T	openterminal

설명	키 바인딩	명령
편집기와 [터미널(Terminal)] 탭 간을 전환합니다.	Option-S	switchterminal

실행 및 디버깅

설명	키 바인딩	명령
현재 파일을 빌드합니다.	Command-B	build
현재 일시 중지된 프로세스를 다시 시작합니다.	F8 | Command-\	resume
현재 애플리케이션을 실행하거나 디버그합니다.	Option-F5	run
마지막 실행 파일을 실행하거나 디버그합니다.	F5	runlast
스택 옆에 있는 함수로 진행합니다.	F11 | Command-;	stepinto
현재 함수 범위를 종료합니다.	Shift-F11 | Command-Shift-'	stepout
스택의 현재 표현식으로 진행합니다.	F10 | Command-'	stepover
현재 애플리케이션의 실행 또는 디버깅을 중지합니다.	Shift-F5	stop
현재 파일의 빌드를 중지합니다.	Control-Shift-C	stopbuild

AWS Cloud9 통합 개발 환경(IDE)에 대한 macOS Emacs 키 바인딩 참조

다음은 AWS Cloud9 IDE의 macOS 운영 체제용 Emacs 키보드 모드 키 바인딩 목록입니다.

자세한 내용은 AWS Cloud9 IDE에서 다음을 참조하세요.

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. [기본 설정(Preferences)] 탭에서 [키 바인딩(Keybindings)]을 선택합니다.
3. 용키보드 모드를 선택하고이맥스.
4. [운영 체제(Operating system)]에서 [MacOS]를 선택합니다.

키 바인딩 작업도 참조하세요.

일반

설명	키 바인딩	명령
선택 영역을 시계 표현식으로 추가합니다.	`Command-Shift-C`	`addwatchfromselection`
클립보드에서 잘라낸 선택 영역을 제거합니다.	`Esc`	`clearcut`
코드 완성 컨텍스트 메뉴를 표시합니다.	`Control-Space` | `Option-Space`	`complete`

설명	키 바인딩	명령
코드를 완료한 다음 덮어씁니다.	`Control-Shift-Space` | `Option-Shift-Space`	`completeoverwrite`
선택 영역을 클립보드로 복사합니다.	`Command-C`	`copy`
선택 영역을 클립보드로 잘라냅니다.	`Command-X`	`cut`
코드를 확장합니다(해당하는 경우).	`Tab`	`expandSnippet`
현재 문서의 찾기 및 바꾸기 막대를 표시합니다.	`Command-F`	`find`
현재 문서에서 일치하는 항목을 모두 선택합니다.	`Control-Option-G`	`findAll`
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 다음 항목으로 이동합니다.	`Command-G`	`findnext`
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 이전 항목으로 이동합니다.	`Command-Shift-G`	`findprevious`
편집기에서 활성 파일의 삽입점에 기호에 대한 알려진 모든 참조를 표시합니다.	`Shift-F3`	`findReferences`
[환경(Environment) 창을 연 다음 파일 목록을 활성 상태로 만듭니다.	`Shift-Esc`	`focusTree`
선택한 JavaScript 코드를 다시 포맷합니다.	`Command-Shift-B`	`formatcode`

설명	키 바인딩	명령
줄로 이동 상자를 표시합니다.	Command-L	gotoline
찾기 및 바꾸기 막대가 표시된 경우 숨깁니다.	Esc	hidesearchreplace
커서 위치에 있는 변수 또는 함수의 정의로 이동합니다.	F3	jumptodef
AWS 리소스 창의 Lambda 섹션에서 로컬 Lambda 함수를 선택한 경우 해당 함수를 원격 함수로 Lambda에 업로드하려고 시도합니다.	Command-Shift-U	lambdaUploadFunction
새 파일을 만듭니다.	Control-N	newfile
[기본 설정(Preferences)] 탭을 표시합니다.	Command-,	openpreferences
[터미널(Terminal)] 탭을 연 다음 파일 목록에서 선택한 파일의 상위 폴더로 전환합니다.	Command-Option-L	opentermhere
클립보드의 현재 내용을 커서 위치에 붙여 넣습니다.	Command-V	paste
오류 수정에 대한 제안 사항을 표시합니다.	Command-F3	quickfix
마지막 작업을 다시 실행합니다.	Command-Shift-Z | Command-Y	redo
미리 보기 창을 새로 고칩니다.	Command-Enter	reloadpreview
선택 항목의 이름 바꾸기 리팩터링을 시작합니다.	Option-Command-R	renameVar

설명	키 바인딩	명령
현재 문서의 찾기 및 바꾸기 모음을 표시하고 [바꾸기(Replace With)] 표현식을 사용합니다.	`Option-Command-F`	`replace`
초기화 스크립트를 다시 실행합니다.	`Command-Enter`	`rerunInitScript`
환경을 다시 시작합니다.	`Command-R`	`restartc9`
현재 파일을 마지막으로 저장한 버전으로 재설정합니다.	`Control-Shift-Q`	`reverttosaved`
열려 있는 각 파일을 저장된 버전으로 재설정합니다.	`Option-Shift-Q`	`reverttosavedall`
현재 파일을 디스크에 저장합니다.	`Command-S`	`save`
현재 파일을 다른 파일 이름을 사용하여 디스크에 저장합니다.	`Command-Shift-S`	`saveas`
여러 파일의 찾기 및 바꾸기 모음을 표시합니다.	`Shift-Command-F`	`searchinfiles`
[프로세스 목록(Process List)] 대화 상자를 표시합니다.	`Command-Option-P`	`showprocesslist`
마지막 작업을 실행 취소합니다.	`Command-Z`	`undo`

탭

설명	키 바인딩	명령
현재 창에서 현재 탭을 제외하고 열려 있는 탭을 모두 닫습니다.	Option-Control-W	closeallbutme
모든 창에서 열려 있는 탭을 모두 닫습니다.	Option-Shift-W	closealltabs
현재 창을 닫습니다.	Command-Control-W	closepane
현재 탭을 닫습니다.	Option-W	closetab
아래로 창을 하나 이동합니다.	Control-Command-Down	gotopanedown
왼쪽으로 창을 하나 이동합니다.	Control-Command-Left	gotopaneleft
오른쪽으로 창을 하나 이동합니다.	Control-Command-Right	gotopaneright
위로 창을 하나 이동합니다.	Control-Command-Up	gottopaneup
왼쪽으로 한 탭 이동합니다.	Command-[	gototableft
오른쪽으로 한 탭 이동합니다.	Command-]	gototabright
현재 탭을 한 창 아래로 이동하거나 탭이 이미 맨 아래에 있는 경우 분할 탭을 만듭니다.	Command-Option-Shift-Down	movetabdown
현재 탭을 왼쪽으로 이동하거나 탭이 이미 맨 왼쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Command-Option-Shift-Left	movetableft
현재 탭을 오른쪽으로 이동하거나 탭이 이미 맨 오른쪽에 있	Command-Option-Shift-Right	movetabright

설명	키 바인딩	명령
는 경우 해당 위치에 분할 탭을 만듭니다.		
현재 탭을 한 창 위로 이동하거나 탭이 이미 맨 위에 있는 경우 분할 탭을 만듭니다.	Command-Option-Shift-Up	movetabup
다음 창으로 이동합니다.	Option-Esc	nextpane
다음 탭으로 이동합니다.	Option-Tab	nexttab
이전 창으로 이동합니다.	Option-Shift-Esc	previouspane
이전 탭으로 이동합니다.	Option-Shift-Tab	previoustab
마지막 탭으로 돌아갑니다.	Esc	refocusTab
마지막 탭을 다시 엽니다.	Option-Shift-T	reopenLastTab
파일 트리에 현재 탭을 표시합니다.	Command-Shift-L	revealtab
10번째 탭으로 이동합니다.	Command-0	tab0
첫 번째 탭으로 이동합니다.	Command-1	tab1
두 번째 탭으로 이동합니다.	Command-2	tab2
세 번째 탭으로 이동합니다.	Command-3	tab3
네 번째 탭으로 이동합니다.	Command-4	tab4
다섯 번째 탭으로 이동합니다.	Command-5	tab5
여섯 번째 탭으로 이동합니다.	Command-6	tab6
일곱 번째 탭으로 이동합니다.	Command-7	tab7
여덟 번째 탭으로 이동합니다.	Command-8	tab8

설명	키 바인딩	명령
아홉 번째 탭으로 이동합니다.	Command	tab9

패널

설명	키 바인딩	명령
[바로 가기(Go to Anything)] 모드로 [이동(Go)] 창을 표시합니다.	Command-E|Command-P	gotoanything
[명령으로 이동(Go to Command)] 모드로 [이동(Go)] 창을 표시합니다.	Command-.|F1	gotocommand
[파일로 이동(Go to File)] 모드로 [이동(Go)] 창을 표시합니다.	Command-0	gotofile
[기호로 이동(Go to Symbol)] 모드로 [이동(Go)] 창을 표시합니다.	Command-Shift-0	gotosymbol
[개요(Outline)] 창을 표시합니다.	Command-Shift-E	outline
숨겨져 있는 경우 [콘솔(Console)] 창을 표시하거나 표시된 경우 숨깁니다.	Control-Esc	toggleconsole
숨겨져 있는 경우 [환경(Environment)] 창을 표시하거나 표시된 경우 숨깁니다.	Command-U	toggletree

코드 편집기

설명	키 바인딩	명령
활성 커서 한 줄 위에 커서를 추가하거나 커서가 이미 추가된 경우 해당 커서 위에 다른 커서를 추가합니다.	`Control-Option-Up`	`addCursorAbove`
활성 커서 한 줄 위에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 위로 이동합니다.	`Control-Option-Shift-Up`	`addCursorAboveSkipCurrent`
활성 커서 한 줄 아래에 커서를 추가하거나 커서가 이미 추가된 경우 해당 커서 아래에 다른 커서를 추가합니다.	`Control-Option-Down`	`addCursorBelow`
활성 커서 한 줄 아래에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 아래로 이동합니다.	`Control-Option-Shift-Down`	`addCursorBelowSkipCurrent`
정렬이 잘못된 경우 모든 커서를 각 행의 활성 커서와 동일한 공간으로 이동합니다.	`Control-Option-A`	`alignCursors`
단일 공백을 백스페이스합니다.	`Control-Backspace | Shift-Backspace | Backspace`	`backspace`
선택 영역을 한 탭 들여씁니다.	`Control-]`	`blockindent`
선택 영역을 한 탭 내어씁니다.	`Control-[`	`blockoutdent`

설명	키 바인딩	명령
포커스를 편집기에서 IDE의 다른 곳으로 전환할 수 있는지 여부를 제어합니다.	Command-Z | Command-Shift-Z | Command-Y	cancelBrowserUndoInAce
선택 영역을 가운데로 정렬합니다.	Control-L	centerselection
줄의 내용을 복사하고 복사한 내용을 한 줄 아래에 붙여 넣습니다.	Command-Option-Down	copylinesdown
줄의 내용을 복사하고 복사한 내용을 한 줄 위에 붙여 넣습니다.	Command-Option-Up	copylinesup
공백을 하나 삭제합니다.	Delete | Control-Delete | Shift-Delete	del
선택 영역의 내용을 복사하고 복사한 내용을 선택 영역 바로 뒤에 붙여넣습니다.	Command-Shift-D	duplicateSelection
선택 영역에 현재 행의 내용을 포함합니다.	Command-Shift-L	expandtoline
선택 영역에 일치하는 다음 기호까지 포함합니다.	Control-Shift-M	expandToMatching
선택한 코드를 접습니다. 접힌 단위가 선택된 경우 펼칩니다.	Command-Option-L | Command-F1	fold
접을 수 있는 모든 요소를 접습니다.	Control-Command-Option-0	foldall
현재 선택 범위를 제외하고 접을 수 있는 모든 요소를 접습니다.	Command-Option-0	foldOther

설명	키 바인딩	명령
한 줄 아래로 이동합니다.	Down | Control-N	golinedown
한 줄 위로 이동합니다.	Up | Control-P	golineup
로그 파일의 끝 부분으로 이동합니다.	Command-End | Command-Down	gotoend
한 칸씩 왼쪽으로 이동합니다.	Left | Control-B	gotoleft
현재 줄의 끝으로 이동합니다.	Command-Right | End | Control-E	gotolineend
현재 줄의 처음으로 이동합니다.	Command-Left | Home | Control-A	gotolinestart
다음 오류로 이동합니다.	F4	goToNextError
한 페이지 아래로 이동합니다.	Page Down | Control-V	gotopagedown
한 페이지 위로 이동합니다.	Page Up	gotopageup
이전 오류로 이동합니다.	Shift-F4	goToPreviousError
오른쪽으로 공백 하나 이동합니다.	Right | Control-F	gotoright
로그 파일의 시작 부분으로 이동합니다.	Command-Home | Command-Up	gotostart
왼쪽으로 한 단어 이동합니다.	Option-Left	gotowordleft
오른쪽으로 한 단어 이동합니다.	Option-Right	gotowordright
선택 영역을 한 탭 들여씁니다.	Tab	indent
현재 범위에서 쌍을 이루는 기호로 이동합니다.	Control-P	jumptomatching

설명	키 바인딩	명령
글꼴 크기를 늘립니다.	Command-+ | Command-=	largerfont
커서의 왼쪽에 있는 숫자를 1씩 줄입니다(숫자인 경우).	Option-Shift-Down	modifyNumberDown
커서의 왼쪽에 있는 숫자를 1씩 늘립니다(숫자인 경우).	Option-Shift-Up	modifyNumberUp
선택 영역을 한 줄 아래로 이동합니다.	Option-Down	movelinesdown
선택 영역을 한 줄 위로 이동합니다.	Option-Up	movelinesup
선택 영역을 한 탭 내어씁니다.	Shift-Tab	outdent
덮어쓰기 모드를 켜거나 켜진 경우 끕니다.	Insert	overwrite
한 페이지 아래로 이동합니다.	Option-Page Down	pagedown
한 페이지 위로 이동합니다.	Option-Page Up	pageup
현재 줄을 제거합니다.	Command-D	removeline
커서에서 현재 줄의 끝까지 삭제합니다.	Control-K	removetolineend
커서에서 현재 줄의 시작까지 삭제합니다.	Command-Backspace	removetolinestart
커서의 왼쪽에 있는 단어를 삭제합니다.	Option-Backspace | Control-Option-Backspace	removewordleft
커서의 오른쪽에 있는 단어를 삭제합니다.	Option-Delete	removewordright

설명	키 바인딩	명령
이전에 기록한 키 입력을 재생합니다.	Command-Shift-E	replaymacro
선택 가능한 내용을 모두 선택합니다.	Command-A	selectall
선택 영역에 다음 줄 아래의 내용을 포함합니다.	Shift-Down | Control-Shift-N	selectdown
선택 영역에 다음 공백 왼쪽의 내용을 포함합니다.	Shift-Left | Control-Shift-B	selectleft
커서부터 현재 줄의 나머지 부분을 선택 영역에 포함합니다.	Shift-End	selectlineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Shift-Home	selectlinestart
선택 영역의 뒤에 일치하는 선택 영역을 더 포함합니다.	Control-Option-Right	selectMoreAfter
선택 영역의 앞에 일치하는 선택 영역을 더 포함합니다.	Control-Option-Left	selectMoreBefore
선택 영역의 뒤에 있는 일치하는 다음 선택 영역을 포함합니다.	Control-Option-Shift-Right	selectNextAfter
선택 영역의 앞에 있는 일치하는 다음 선택 영역을 포함합니다.	Control-Option-Shift-Left	selectNextBefore
일치하는 다음 선택 항목을 선택하거나 찾습니다.	Control-G	selectOrFindNext
일치하는 이전 선택 항목 선택하거나 찾습니다.	Control-Shift-G	selectOrFindPrevious

설명	키 바인딩	명령
커서에서 현재 페이지의 끝까지 선택 영역에 포함합니다.	Shift-Page Down	selectpagedown
커서에서 현재 페이지의 처음까지 선택 영역에 포함합니다.	Shift-Page Up	selectpageup
선택 영역에 커서 오른쪽의 다음 공백를 포함합니다.	Shift-Right	selectright
커서에서 현재 파일의 끝까지 선택 영역에 포함합니다.	Command-Shift-End | Command-Shift-Down	selecttoend
커서에서 현재 줄의 끝까지 선택 영역에 포함합니다.	Command-Shift-Right | Shift-End | Control-Shift-E	selecttolineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Command-Shift-Left | Control-Shift-A	selecttolinestart
커서에서 현재 범위의 일치하는 다음 기호까지 포함합니다.	Control-Shift-P	selecttomatching
커서에서 현재 파일의 처음까지 선택 영역에 포함합니다.	Command-Shift-Home | Command-Shift-Up	selecttostart
선택 영역에 다음 줄 위의 내용을 포함합니다.	Shift-Up | Control-Shift-Up	selectup
선택 영역에 커서 왼쪽의 다음 단어를 포함합니다.	Option-Shift-Left	selectwordleft
선택 영역에 커서 오른쪽의 다음 단어를 포함합니다.	Option-Shift-Right	selectwordright
[기본 설정(Preferences)] 탭을 표시합니다.	Command-,	showSettingsMenu

설명	키 바인딩	명령
이전 선택 영역을 모두 지웁니다.	Esc	singleSelection
글꼴 크기를 줄입니다.	Command--	smallerfont
여러 줄을 선택한 경우 정렬된 순서로 다시 정렬합니다.	Command-Option-S	sortlines
현재 줄의 끝에 커서를 추가합니다.	Control-Option-L	splitIntoLines
커서의 내용을 줄의 끝에 새 줄로 이동합니다.	Control-O	splitline
블록 주석 문자로 선택 영역을 둘러싸거나 해당 문자가 있는 경우 제거합니다.	Command-Shift-/	toggleBlockComment
선택한 각 줄의 처음에 줄 설명 문자를 추가하거나, 줄 설명 문자가 있는 경우 제거합니다.	Command-/	togglecomment
코드를 접거나, 코드 접기를 제거합니다(있는 경우).	F2	toggleFoldWidget
상위 코드를 접거나, 접기를 제거합니다(있는 경우).	Option-F2	toggleParentFoldWidget
키 입력 기록을 시작하거나 이미 기록 중인 경우 중지합니다.	Command-Option-E	togglerecording
단어를 줄 바꿈하거나 이미 줄 바꿈이 있는 경우 단어 줄 바꿈을 중지합니다.	Control-W	toggleWordWrap
선택 내용을 모두 소문자로 변경합니다.	Control-Shift-U	tolowercase

설명	키 바인딩	명령
선택 내용을 모두 소문자로 변경합니다.	Control-U	touppercase
선택 영역의 대소문자를 바꿉니다.	Control-T	transposeletters
선택한 코드를 펼칩니다.	Command-Option-Shift-L | Command-Shift-F1	unfold
전체 파일의 코드 접기를 펼칩니다.	Command-Option-Shift-0	unfoldall

emmet

설명	키 바인딩	명령
간단한 수학 표현식(예: 2*4 또는 10/2)과 그 결과를 출력합니다.	Shift-Command-Y	emmet_evaluate_math_expression
현재 파일의 구문에 따라 CSS와 유사한 약어를 HTML, XML 또는 CSS 코드로 확장합니다.	Control-Option-E	emmet_expand_abbreviation
탭 정지에 의해 확장된 CSS와 유사한 약어를 순회합니다.	Tab	emmet_expand_abbreviation_with_tab
편집 가능한 다음 코드 부분으로 이동합니다.	Shift-Command-.	emmet_select_next_item
이전 편집 가능한 코드 부분으로 이동합니다.	Shift-Command-,	emmet_select_previous_item

설명	키 바인딩	명령
약어를 확장한 다음 생성된 코드 조각의 마지막 요소 내에 현재 선택 영역을 배치합니다.	Shift-Control-A	emmet_wrap_with_abbreviation

터미널

설명	키 바인딩	명령
새 [터미널(Terminal)] 탭을 엽니다.	Option-T	openterminal
편집기와 [터미널(Terminal)] 탭 간을 전환합니다.	Option-S	switchterminal

실행 및 디버깅

설명	키 바인딩	명령
현재 파일을 빌드합니다.	Command-B	build
현재 일시 중지된 프로세스를 다시 시작합니다.	F8 | Command-\	resume
현재 애플리케이션을 실행하거나 디버그합니다.	Option-F5	run
마지막 실행 파일을 실행하거나 디버그합니다.	F5	runlast
스택 옆에 있는 함수로 진행합니다.	F11 | Command-;	stepinto
현재 함수 범위를 종료합니다.	Shift-F11 | Command-Shift-'	stepout

설명	키 바인딩	명령
스택의 현재 표현식으로 진행합니다.	`F10 | Command-'`	`stepover`
현재 애플리케이션의 실행 또는 디버깅을 중지합니다.	`Shift-F5`	`stop`
현재 파일의 빌드를 중지합니다.	`Control-Shift-C`	`stopbuild`

AWS Cloud9 통합 개발 환경(IDE)에 대한 macOS Sublime 키 바인딩 참조

다음은 AWS Cloud9 IDE의 macOS 운영 체제용 Sublime 키보드 모드 키 바인딩 목록입니다.

자세한 내용은 AWS Cloud9 IDE에서 다음을 참조하세요.

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. [기본 설정(Preferences)] 탭에서 [키 바인딩(Keybindings)]을 선택합니다.
3. 용키보드 모드를 선택하고숭고한.
4. [운영 체제(Operating system)]에서 [MacOS]를 선택합니다.

키 바인딩 작업도 참조하세요.

- 일반
- 탭
- 패널
- 코드 편집기
- emmet
- 터미널
- 실행 및 디버그

일반

설명	키 바인딩	명령
선택 영역을 시계 표현식으로 추가합니다.	Command-Shift-C	addwatchfromselection
클립보드에서 잘라낸 선택 영역을 제거합니다.	Esc	clearcut
코드 완성 컨텍스트 메뉴를 표시합니다.	Control-Space | Option-Space	complete
코드를 완료한 다음 덮어씁니다.	Control-Shift-Space | Option-Shift-Space	completeoverwrite
선택 영역을 클립보드로 복사합니다.	Command-C	copy
선택 영역을 클립보드로 잘라냅니다.	Command-X	cut
커서에서 줄의 시작까지 삭제합니다.	Command-K Command-Backspace | Command-Backspace	delete_to_hard_bol
커서에서 줄의 끝까지 삭제합니다.	Command-K Command-K | Command-Delete | Control-K	delete_to_hard_eol
코드를 확장합니다(해당하는 경우).	Tab	expandSnippet
현재 문서의 찾기 및 바꾸기 막대를 표시합니다.	Command-F	find
선택 항목에 대한 모든 일치 항목 강조 표시	Control-Command-G	find_all_under

설명	키 바인딩	명령
선택 항목에 대해 다음 일치 항목 강조 표시	Option-Command-G	find_under
커서 주위에 강조 표시하고 강조 표시와 일치하는 모든 항목에 강조 표시합니다.	Command-D	find_under_expand
커서 주위에 강조 표시하고 강조 표시와 일치하는 모든 항목에 윤곽선을 표시합니다.	Command-K Command-D	find_under_expand_skip
선택 영역에 대해 이전 일치 항목에 강조 표시합니다.	Shift-Option-Command-G	find_under_previous
현재 문서에서 일치하는 항목을 모두 선택합니다.	Control-Option-G	findAll
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 다음 항목으로 이동합니다.	Command-G	findnext
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 이전 항목으로 이동합니다.	Shift-Command-G	findprevious
편집기에서 활성 파일의 삽입점에 기호에 대한 알려진 모든 참조를 표시합니다.	Shift-F3	findReferences
[환경(Environment) 창을 연 다음 파일 목록을 활성 상태로 만듭니다.	Shift-Esc	focusTree
선택한 JavaScript 코드를 다시 포맷합니다.	Control-Option-F	formatcode
줄로 이동 상자를 표시합니다.	Control-G	gotoline

설명	키 바인딩	명령
찾기 및 바꾸기 막대가 표시된 경우 숨깁니다.	Esc	hidesearchreplace
커서 위치에 있는 변수 또는 함수의 정의로 이동합니다.	F12 | Command-Option-Down	jumptodef
AWS 리소스 창의 Lambda 섹션에서 로컬 Lambda 함수를 선택한 경우 해당 함수를 원격 함수로 Lambda에 업로드하려고 시도합니다.	Command-Shift-U	lambdaUploadFunction
현재 단어의 끝으로 이동합니다.	Option-Right	moveToWordEndRight
현재 단어의 처음으로 이동합니다.	Option-Left	moveToWordStartLeft
새 파일을 만듭니다.	Control-N	newfile
[기본 설정(Preferences)] 탭을 표시합니다.	Command-,	openpreferences
[터미널(Terminal)] 탭을 연 다음 파일 목록에서 선택한 파일의 상위 폴더로 전환합니다.	Command-Option-L	opentermhere
클립보드의 현재 내용을 커서 위치에 붙여 넣습니다.	Command-V	paste
오류 수정에 대한 제안 사항을 표시합니다.	Command-F3	quickfix
마지막 작업을 다시 실행합니다.	Command-Shift-Z | Command-Y	redo
미리 보기 창을 새로 고칩니다.	Command-Enter	reloadpreview

설명	키 바인딩	명령
선택 항목의 이름 바꾸기 리팩터링을 시작합니다.	Option-Command-R	renameVar
현재 문서의 찾기 및 바꾸기 모음을 표시하고 [바꾸기(Replace With)] 표현식을 사용합니다.	Command-Option-F	replace
찾기 및 바꾸기 막대에서 모든 찾기 표현식 일치 항목을 바꾸기 표현식으로 대체합니다.	Control-Option-Enter	replaceall
찾기 및 바꾸기 막대에서 다음 찾기 표현식 일치 항목을 바꾸기 표현식으로 대체합니다.	Command-Option-E	replacenext
초기화 스크립트를 다시 실행합니다.	Command-Enter	rerunInitScript
환경을 다시 시작합니다.	Command-R	restartc9
현재 파일을 마지막으로 저장한 버전으로 재설정합니다.	Control-Shift-Q	reverttosaved
열려 있는 각 파일을 저장된 버전으로 재설정합니다.	Option-Shift-Q	reverttosavedall
현재 파일을 디스크에 저장합니다.	Command-S	save
현재 파일을 다른 파일 이름을 사용하여 디스크에 저장합니다.	Command-Shift-S	saveas
여러 파일의 찾기 및 바꾸기 모음을 표시합니다.	Command-Shift-F	searchinfiles

설명	키 바인딩	명령
커서에서 단어의 끝까지 선택 영역에 포함합니다.	Option-Shift-Right	selectToWordEndRight
커서에서 단어의 처음까지 선택 영역에 포함합니다.	Option-Shift-Left	selectToWordStartLeft
[프로세스 목록(Process List)] 대화 상자를 표시합니다.	Command-Option-P	showprocesslist
마지막 작업을 실행 취소합니다.	Command-Z	undo

탭

설명	키 바인딩	명령
현재 창에서 현재 탭을 제외하고 열려 있는 탭을 모두 닫습니다.	Option-Control-W	closeallbutme
모든 창에서 열려 있는 탭을 모두 닫습니다.	Option-Shift-W	closealltabs
현재 창을 닫습니다.	Command-Control-W	closepane
현재 탭을 닫습니다.	Option-W	closetab
아래로 창을 하나 이동합니다.	Control-Command-Down	gotopanedown
왼쪽으로 창을 하나 이동합니다.	Control-Command-Left	gotopaneleft
오른쪽으로 창을 하나 이동합니다.	Control-Command-Right	gotopaneright
위로 창을 하나 이동합니다.	Control-Command-Up	gottopaneup

설명	키 바인딩	명령
왼쪽으로 한 탭 이동합니다.	Command-Shift-[| Command-Option-Left	gototableft
오른쪽으로 한 탭 이동합니다.	Command-Shift-] | Command-Option-Right	gototabright
현재 탭을 한 창 아래로 이동하거나 탭이 이미 맨 아래에 있는 경우 분할 탭을 만듭니다.	Command-Option-Shift-Down	movetabdown
현재 탭을 왼쪽으로 이동하거나 탭이 이미 맨 왼쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Command-Option-Shift-Left	movetableft
현재 탭을 오른쪽으로 이동하거나 탭이 이미 맨 오른쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Command-Option-Shift-Right	movetabright
현재 탭을 한 창 위로 이동하거나 탭이 이미 맨 위에 있는 경우 분할 탭을 만듭니다.	Command-Option-Shift-Up	movetabup
다음 탭으로 이동합니다.	Control-Tab	nexttab
이전 창으로 이동합니다.	Option-Shift-Esc	previouspane
이전 탭으로 이동합니다.	Control-Shift-Tab	previoustab
마지막 탭으로 돌아갑니다.	Esc	refocusTab
마지막 탭을 다시 엽니다.	Command-Shift-T	reopenLastTab
파일 트리에 현재 탭을 표시합니다.	Command-E	revealtab
10번째 탭으로 이동합니다.	Command-0	tab0

설명	키 바인딩	명령
첫 번째 탭으로 이동합니다.	`Command-1`	`tab1`
두 번째 탭으로 이동합니다.	`Command-2`	`tab2`
세 번째 탭으로 이동합니다.	`Command-3`	`tab3`
네 번째 탭으로 이동합니다.	`Command-4`	`tab4`
다섯 번째 탭으로 이동합니다.	`Command-5`	`tab5`
여섯 번째 탭으로 이동합니다.	`Command-6`	`tab6`
일곱 번째 탭으로 이동합니다.	`Command-7`	`tab7`
여덟 번째 탭으로 이동합니다.	`Command-8`	`tab8`
아홉 번째 탭으로 이동합니다.	`Command`	`tab9`

패널

설명	키 바인딩	명령
[바로 가기(Go to Anything)] 모드로 [이동(Go)] 창을 표시합니다.	`Command-E|Command-P`	`gotoanything`
[명령으로 이동(Go to Command)] 모드로 [이동(Go)] 창을 표시합니다.	`Command-.|F1`	`gotocommand`
[파일로 이동(Go to File)] 모드로 [이동(Go)] 창을 표시합니다.	`Command-O`	`gotofile`
[기호로 이동(Go to Symbol)] 모드로 [이동(Go)] 창을 표시합니다.	`Command-Shift-O`	`gotosymbol`

설명	키 바인딩	명령
[개요(Outline)] 창을 표시합니다.	`Command-Shift-R`	`outline`
숨겨져 있는 경우 [콘솔(Console)] 창을 표시하거나 표시된 경우 숨깁니다.	`Control-``	`toggleconsole`
숨겨져 있는 경우 [환경(Environment)] 창을 표시하거나 표시된 경우 숨깁니다.	`Command-K Command-B`	`toggletree`

코드 편집기

설명	키 바인딩	명령
활성 커서 한 줄 위에 커서를 추가하거나 커서가 이미 추가된 경우 해당 커서 위에 다른 커서를 추가합니다.	`Control-Shift-Up`	`addCursorAbove`
활성 커서 한 줄 위에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 위로 이동합니다.	`Control-Option-Shift-Up`	`addCursorAboveSkipCurrent`
활성 커서 한 줄 아래에 커서를 추가하거나 커서가 이미 추가된 경우 해당 커서 아래에 다른 커서를 추가합니다.	`Control-Shift-Down`	`addCursorBelow`
활성 커서 한 줄 아래에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 아래로 이동합니다.	`Control-Option-Shift-Down`	`addCursorBelowSkipCurrent`

설명	키 바인딩	명령
정렬이 잘못된 경우 모든 커서를 각 행의 활성 커서와 동일한 공간으로 이동합니다.	Control-Option-A	alignCursors
단일 공백을 백스페이스합니다.	Control-Backspace | Shift-Backspace | Backspace	backspace
선택 영역을 한 탭 들여씁니다.	Control-]	blockindent
선택 영역을 한 탭 내어씁니다.	Control-[	blockoutdent
포커스를 편집기에서 IDE의 다른 곳으로 전환할 수 있는지 여부를 제어합니다.	Command-Z | Command-Shift-Z | Command-Y	cancelBrowserUndoInAce
선택 영역을 가운데로 정렬합니다.	Command-K Command-C | Control-L	centerselection
줄의 내용을 복사하고 복사한 내용을 한 줄 아래에 붙여 넣습니다.	Command-Option-Down	copylinesdown
줄의 내용을 복사하고 복사한 내용을 한 줄 위에 붙여 넣습니다.	Command-Option-Up	copylinesup
공백을 하나 삭제합니다.	Delete | Control-Delete | Shift-Delete	del
선택 영역의 내용을 복사하고 복사한 내용을 선택 영역 바로 뒤에 붙여넣습니다.	Command-Shift-D	duplicateSelection
선택 영역에 현재 행의 내용을 포함합니다.	Command-L	expandtoline

설명	키 바인딩	명령
선택 영역에 일치하는 다음 기호까지 포함합니다.	Control-Shift-M	expandToMatching
선택한 코드를 접습니다. 접힌 단위가 선택된 경우 펼칩니다.	Command-Option-L | Command-F1	fold
접을 수 있는 모든 요소를 접습니다.	Control-Command-Option-0	foldall
현재 선택 범위를 제외하고 접을 수 있는 모든 요소를 접습니다.	Command-K Command-1	foldOther
한 줄 아래로 이동합니다.	Down | Control-N	golinedown
한 줄 위로 이동합니다.	Up | Control-P	golineup
로그 파일의 끝 부분으로 이동합니다.	Command-End | Command-Down	gotoend
한 칸씩 왼쪽으로 이동합니다.	Left | Control-B	gotoleft
현재 줄의 끝으로 이동합니다.	Command-Right | End | Control-E	gotolineend
현재 줄의 처음으로 이동합니다.	Command-Left | Home | Control-A	gotolinestart
다음 오류로 이동합니다.	Control-F6	goToNextError
한 페이지 아래로 이동합니다.	Page Down | Control-V	gotopagedown
한 페이지 위로 이동합니다.	Page Up	gotopageup
이전 오류로 이동합니다.	Control-Shift-F6	goToPreviousError
오른쪽으로 공백 하나 이동합니다.	Right | Control-F	gotoright

설명	키 바인딩	명령
로그 파일의 시작 부분으로 이동합니다.	Command-Home | Command-Up	gotostart
왼쪽으로 한 단어 이동합니다.	Option-Left	gotowordleft
오른쪽으로 한 단어 이동합니다.	Option-Right	gotowordright
선택 영역을 한 탭 들여씁니다.	Tab	indent
선택한 줄을 한 줄로 결합합니다.	Command-J	joinlines
현재 범위에서 쌍을 이루는 기호로 이동합니다.	Control-M	jumptomatching
글꼴 크기를 늘립니다.	Command-= | Command-+	largerfont
커서의 왼쪽에 있는 숫자를 1씩 줄입니다(숫자인 경우).	Option-Down	modifyNumberDown
커서의 왼쪽에 있는 숫자를 1씩 늘립니다(숫자인 경우).	Option-Up	modifyNumberUp
선택 영역을 한 줄 아래로 이동합니다.	Control-Command-Down	movelinesdown
선택 영역을 한 줄 위로 이동합니다.	Control-Command-Up	movelinesup
선택 영역을 한 탭 내어씁니다.	Shift-Tab	outdent
덮어쓰기 모드를 켜거나 켜진 경우 끕니다.	Insert	overwrite
한 페이지 아래로 이동합니다.	Option-Page Down	pagedown
한 페이지 위로 이동합니다.	Option-Page Up	pageup

설명	키 바인딩	명령
현재 줄의 내용을 삭제합니다.	Control-Shift-K	removeline
커서에서 현재 줄의 끝까지 삭제합니다.	Control-K	removetolineend
커서에서 현재 줄의 시작까지 삭제합니다.	Command-Backspace	removetolinestart
커서의 왼쪽에 있는 단어를 삭제합니다.	Option-Backspace | Control-Option-Backspace	removewordleft
커서의 오른쪽에 있는 단어를 삭제합니다.	Option-Delete	removewordright
이전에 기록한 키 입력을 재생합니다.	Control-Shift-Q	replaymacro
선택 가능한 내용을 모두 선택합니다.	Command-A	selectall
선택 영역에 다음 줄 아래의 내용을 포함합니다.	Shift-Down | Control-Shift-N	selectdown
선택 영역에 다음 공백 왼쪽의 내용을 포함합니다.	Shift-Left | Control-Shift-B	selectleft
커서부터 현재 줄의 나머지 부분을 선택 영역에 포함합니다.	Shift-End	selectlineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Shift-Home	selectlinestart
선택 영역의 뒤에 일치하는 선택 영역을 더 포함합니다.	Control-Option-Right	selectMoreAfter
선택 영역의 앞에 일치하는 선택 영역을 더 포함합니다.	Control-Option-Left	selectMoreBefore

설명	키 바인딩	명령
선택 영역의 뒤에 있는 일치하는 다음 선택 영역을 포함합니다.	Control-Option-Shift-Right	selectNextAfter
선택 영역의 앞에 있는 일치하는 다음 선택 영역을 포함합니다.	Control-Option-Shift-Left	selectNextBefore
일치하는 다음 선택 항목을 선택하거나 찾습니다.	Control-G	selectOrFindNext
일치하는 이전 선택 항목 선택하거나 찾습니다.	Control-Shift-G	selectOrFindPrevious
커서에서 현재 페이지의 끝까지 선택 영역에 포함합니다.	Shift-Page Down	selectpagedown
커서에서 현재 페이지의 처음까지 선택 영역에 포함합니다.	Shift-Page Up	selectpageup
선택 영역에 커서 오른쪽의 다음 공백를 포함합니다.	Shift-Right	selectright
커서에서 현재 파일의 끝까지 선택 영역에 포함합니다.	Command-Shift-End | Command-Shift-Down	selecttoend
커서에서 현재 줄의 끝까지 선택 영역에 포함합니다.	Command-Shift-Right | Shift-End | Control-Shift-E	selecttolineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Command-Shift-Left | Control-Shift-A	selecttolinestart
커서에서 현재 범위의 일치하는 다음 기호까지 포함합니다.	Control-Shift-P	selecttomatching

설명	키 바인딩	명령
커서에서 현재 파일의 처음까지 선택 영역에 포함합니다.	Command-Shift-Home | Command-Shift-Up	selecttostart
선택 영역에 다음 줄 위의 내용을 포함합니다.	Shift-Up | Control-Shift-P	selectup
선택 영역에 커서 왼쪽의 다음 단어를 포함합니다.	Option-Shift-Left	selectwordleft
선택 영역에 커서 오른쪽의 다음 단어를 포함합니다.	Option-Shift-Right	selectwordright
[기본 설정(Preferences)] 탭을 표시합니다.	Command-,	showSettingsMenu
이전 선택 영역을 모두 지웁니다.	Esc	singleSelection
글꼴 크기를 줄입니다.	Command--	smallerfont
여러 줄을 선택한 경우 정렬된 순서로 다시 정렬합니다.	F5	sortlines
현재 줄의 끝에 커서를 추가합니다.	Command-Shift-L	splitIntoLines
커서의 내용을 줄의 끝에 새 줄로 이동합니다.	Control-O	splitline
블록 주석 문자로 선택 영역을 둘러싸거나 해당 문자가 있는 경우 제거합니다.	Command-Option-/	toggleBlockComment
선택한 각 줄의 처음에 줄 설명 문자를 추가하거나, 줄 설명 문자가 있는 경우 제거합니다.	Command-/	togglecomment

설명	키 바인딩	명령
코드를 접거나, 코드 접기를 제거합니다(있는 경우).	Command-Option-[	toggleFoldWidget
상위 코드를 접거나, 접기를 제거합니다(있는 경우).	Option-F2	toggleParentFoldWidget
키 입력 기록을 시작하거나 이미 기록 중인 경우 중지합니다.	Control-Q	togglerecording
단어를 줄 바꿈하거나 이미 줄 바꿈이 있는 경우 단어 줄 바꿈을 중지합니다.	Control-W	toggleWordWrap
선택 영역을 모두 소문자로 변경합니다.	Command-K Command-L	tolowercase
선택 영역을 모두 대문자로 변경합니다.	Command-K Command-U	touppercase
선택 영역의 대소문자를 바꿉니다.	Control-T	transposeletters
선택한 코드를 펼칩니다.	Command-Option-]	unfold
전체 파일의 코드 접기를 펼칩니다.	Command-K Command-0 | Command-K Command-J	unfoldall

emmet

설명	키 바인딩	명령
간단한 수학 표현식(예: 2*4 또는 10/2)과 그 결과를 출력합니다.	Shift-Command-Y	emmet_evaluate_math_expression

설명	키 바인딩	명령
현재 파일의 구문에 따라 CSS와 유사한 약어를 HTML, XML 또는 CSS 코드로 확장합니다.	`Control-Option-E`	`emmet_expand_abbreviation`
탭 정지에 의해 확장된 CSS와 유사한 약어를 순회합니다.	`Tab`	`emmet_expand_abbreviation_with_tab`
편집 가능한 다음 코드 부분으로 이동합니다.	`Shift-Command-.`	`emmet_select_next_item`
이전 편집 가능한 코드 부분으로 이동합니다.	`Shift-Command-,`	`emmet_select_previous_item`
약어를 확장한 다음 생성된 코드 조각의 마지막 요소 내에 현재 선택 영역을 배치합니다.	`Shift-Control-A`	`emmet_wrap_with_abbreviation`

터미널

설명	키 바인딩	명령
새 [터미널(Terminal)] 탭을 엽니다.	`Option-T`	`openterminal`
편집기와 [터미널(Terminal)] 탭 간을 전환합니다.	`Option-S`	`switchterminal`

실행 및 디버깅

설명	키 바인딩	명령
현재 파일을 빌드합니다.	`F7 | Command-B`	`build`

설명	키 바인딩	명령
현재 일시 중지된 프로세스를 다시 시작합니다.	F8 | Command-\	resume
현재 애플리케이션을 실행하거나 디버그합니다.	Command-Shift-B	run
마지막 실행 파일을 실행하거나 디버그합니다.	F5	runlast
스택 옆에 있는 함수로 진행합니다.	F11 | Command-;	stepinto
현재 함수 범위를 종료합니다.	Shift-F11 | Command-Shift-'	stepout
스택의 현재 표현식으로 진행합니다.	F10 | Command-'	stepover
현재 애플리케이션의 실행 또는 디버깅을 중지합니다.	Shift-F5	stop
현재 파일의 빌드를 중지합니다.	Control-Break	stopbuild

AWS Cloud9 통합 개발 환경(IDE)에 대한 Windows/Linux 기본 키 바인딩 참조

다음은 AWS Cloud9 IDE의 Windows/Linux 운영 체제용 기본 키보드 모드 키 바인딩 목록입니다.

자세한 내용은 AWS Cloud9 IDE에서 다음을 참조하세요.

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. [기본 설정(Preferences)] 탭에서 [키 바인딩(Keybindings)]을 선택합니다.
3. [키보드 모드(Keyboard Mode)]에서 [기본값(Default)]을 선택합니다.
4. [운영 체제(Operating System)]에서 [Windows / Linux]를 선택합니다.

키 바인딩 작업도 참조하세요.

- 일반
- 탭
- 패널
- 코드 편집기
- emmet
- 터미널
- 실행 및 디버그

일반

설명	키 바인딩	명령
선택 영역을 시계 표현식으로 추가합니다.	`Ctrl-Shift-C`	`addwatchfromselection`
클립보드에서 잘라낸 선택 영역을 제거합니다.	`Esc`	`clearcut`
코드 완성 컨텍스트 메뉴를 표시합니다.	`Ctrl-Space | Alt-Space`	`complete`
코드를 완료한 다음 덮어씁니다.	`Ctrl-Shift-Space | Alt-Shift-Space`	`completeoverwrite`
선택 영역을 클립보드로 복사합니다.	`Ctrl-C`	`copy`
선택 영역을 클립보드로 잘라냅니다.	`Ctrl-X`	`cut`
코드를 확장합니다(해당하는 경우).	`Tab`	`expandSnippet`
현재 문서의 찾기 및 바꾸기 막대를 표시합니다.	`Ctrl-F`	`find`

설명	키 바인딩	명령
현재 문서에서 일치하는 항목을 모두 선택합니다.	`Ctrl-Alt-K`	`findall`
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 다음 항목으로 이동합니다.	`Ctrl-K`	`findnext`
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 이전 항목으로 이동합니다.	`Ctrl-Shift-K`	`findprevious`
편집기에서 활성 파일의 삽입 점에 기호에 대한 알려진 모든 참조를 표시합니다.	`Shift-F3`	`findReferences`
[환경(Environment) 창을 연 다음 파일 목록을 활성 상태로 만듭니다.	`Shift-Esc`	`focusTree`
선택한 JavaScript 코드를 다시 포맷합니다.	`Ctrl-Shift-B`	`formatcode`
줄로 이동 상자를 표시합니다.	`Ctrl-G`	`gotoline`
찾기 및 바꾸기 막대가 표시된 경우 숨깁니다.	`Esc`	`hidesearchreplace`
커서 위치에 있는 변수 또는 함수의 정의로 이동합니다.	`F3`	`jumptodef`
AWS 리소스 창의 Lambda 섹션에서 로컬 Lambda 함수를 선택한 경우 해당 함수를 원격 함수로 Lambda에 업로드하려고 시도합니다.	`Ctrl-Shift-U`	`lambdaUploadFunction`
새 파일을 만듭니다.	`Alt-N`	`newfile`

설명	키 바인딩	명령
[기본 설정(Preferences)] 탭을 표시합니다.	`Ctrl-,`	`openpreferences`
[터미널(Terminal)] 탭을 연 다음 파일 목록에서 선택한 파일의 상위 폴더로 전환합니다.	`Alt-L`	`opentermhere`
클립보드의 현재 내용을 커서 위치에 붙여 넣습니다.	`Ctrl-V`	`paste`
오류 수정에 대한 제안 사항을 표시합니다.	`Ctrl-F3`	`quickfix`
마지막 작업을 다시 실행합니다.	`Ctrl-Shift-Z |Ctrl-Y`	`redo`
미리 보기 창을 새로 고칩니다.	`Ctrl-Enter`	`reloadpreview`
선택 항목의 이름 바꾸기 리팩터링을 시작합니다.	`Ctrl-Alt-R`	`renameVar`
현재 문서의 찾기 및 바꾸기 모음을 표시하고 [바꾸기(Replace With)] 표현식을 사용합니다.	`Alt-Shift-F |Ctrl-H`	`replace`
초기화 스크립트를 다시 실행합니다.	`Ctrl-Enter`	`rerunInitScript`
환경을 다시 시작합니다.	`Ctrl-R`	`restartc9`
현재 파일을 마지막으로 저장한 버전으로 재설정합니다.	`Ctrl-Shift-Q`	`reverttosaved`
열려 있는 각 파일을 저장된 버전으로 재설정합니다.	`Alt-Shift-Q`	`reverttosavedall`

설명	키 바인딩	명령
현재 파일을 디스크에 저장합니다.	Ctrl-S	save
현재 파일을 다른 파일 이름을 사용하여 디스크에 저장합니다.	Ctrl-Shift-S	saveas
여러 파일의 찾기 및 바꾸기 모음을 표시합니다.	Ctrl-Shift-F	searchinfiles
[프로세스 목록(Process List)] 대화 상자를 표시합니다.	Ctrl-Alt-P	showprocesslist
마지막 작업을 실행 취소합니다.	Ctrl-Z	undo

탭

설명	키 바인딩	명령
현재 창에서 현재 탭을 제외하고 열려 있는 탭을 모두 닫습니다.	Ctrl-Alt-W	closeallbutme
모든 창에서 열려 있는 탭을 모두 닫습니다.	Alt-Shift-W	closealltabs
현재 창을 닫습니다.	Ctrl-W	closepane
현재 탭을 닫습니다.	Alt-W	closetab
아래로 창을 하나 이동합니다.	Ctrl-Meta-Down	gotopanedown
왼쪽으로 창을 하나 이동합니다.	Ctrl-Meta-Left	gotopaneleft

설명	키 바인딩	명령
오른쪽으로 창을 하나 이동합니다.	Ctrl-Meta-Right	gotopaneright
위로 창을 하나 이동합니다.	Ctrl-Meta-Up	gottopaneup
왼쪽으로 한 탭 이동합니다.	Ctrl-[	gototableft
오른쪽으로 한 탭 이동합니다.	Ctrl-]	gototabright
현재 탭을 한 창 아래로 이동하거나 탭이 이미 맨 아래에 있는 경우 분할 탭을 만듭니다.	Ctrl-Meta-Down	movetabdown
현재 탭을 왼쪽으로 이동하거나 탭이 이미 맨 왼쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Ctrl-Meta-Left	movetableft
현재 탭을 오른쪽으로 이동하거나 탭이 이미 맨 오른쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Ctrl-Meta-Right	movetabright
현재 탭을 한 창 위로 이동하거나 탭이 이미 맨 위에 있는 경우 분할 탭을 만듭니다.	Ctrl-Meta-Up	movetabup
다음 창으로 이동합니다.	Ctrl-`	nextpane
다음 탭으로 이동합니다.	Ctrl-Tab | Alt-`	nexttab
이전 창으로 이동합니다.	Ctrl-Shift-`	previouspane
이전 탭으로 이동합니다.	Ctrl-Shift-Tab | Alt-Shift-`	previoustab
마지막 탭으로 돌아갑니다.	Esc	refocusTab

설명	키 바인딩	명령
마지막 탭을 다시 엽니다.	`Alt-Shift-T`	`reopenLastTab`
파일 트리에 현재 탭을 표시합니다.	`Ctrl-Shift-L`	`revealtab`
10번째 탭으로 이동합니다.	`Ctrl-0`	`tab0`
첫 번째 탭으로 이동합니다.	`Ctrl-1`	`tab1`
두 번째 탭으로 이동합니다.	`Ctrl-2`	`tab2`
세 번째 탭으로 이동합니다.	`Ctrl-3`	`tab3`
네 번째 탭으로 이동합니다.	`Ctrl-4`	`tab4`
다섯 번째 탭으로 이동합니다.	`Ctrl-5`	`tab5`
여섯 번째 탭으로 이동합니다.	`Ctrl-6`	`tab6`
일곱 번째 탭으로 이동합니다.	`Ctrl-7`	`tab7`
여덟 번째 탭으로 이동합니다.	`Ctrl-8`	`tab8`
아홉 번째 탭으로 이동합니다.	`Ctrl-9`	`tab9`

패널

설명	키 바인딩	명령
[바로 가기(Go to Anything)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-E|Ctrl-P`	`gotoanything`
[명령으로 이동(Go to Command)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-.|F1`	`gotocommand`

설명	키 바인딩	명령
[파일로 이동(Go to File)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-O`	`gotofile`
[기호로 이동(Go to Symbol)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-Shift-O`	`gotosymbol`
[개요(Outline)] 창을 표시합니다.	`Ctrl-Shift-E`	`outline`
숨겨져 있는 경우 [콘솔(Console)] 창을 표시하거나 표시된 경우 숨깁니다.	`F6`	`toggleconsole`
숨겨져 있는 경우 [환경(Environment)] 창을 표시하거나 표시된 경우 숨깁니다.	`Ctrl-I`	`toggletree`

코드 편집기

설명	키 바인딩	명령
활성 커서 한 줄 위에 커서를 추가하거나 커서가 이미 추가된 경우 해당 커서 위에 다른 커서를 추가합니다.	`Ctrl-Alt-Up`	`addCursorAbove`
활성 커서 한 줄 위에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 위로 이동합니다.	`Ctrl-Alt-Shift-Up`	`addCursorAboveSkipCurrent`
활성 커서 한 줄 아래에 커서를 추가하거나 커서가 이미 추가	`Ctrl-Alt-Down`	`addCursorBelow`

설명	키 바인딩	명령
된 경우 해당 커서 아래에 다른 커서를 추가합니다.		
활성 커서 한 줄 아래에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 아래로 이동합니다.	Ctrl-Alt-Shift-Down	addCursorBelowSkipCurrent
정렬이 잘못된 경우 모든 커서를 각 행의 활성 커서와 동일한 공간으로 이동합니다.	Ctrl-Alt-A	alignCursors
단일 공백을 백스페이스합니다.	Shift-Backspace | Backspace	backspace
선택 영역을 한 탭 들여씁니다.	Ctrl-]	blockindent
선택 영역을 한 탭 내어씁니다.	Ctrl-[	blockoutdent
포커스를 편집기에서 IDE의 다른 곳으로 전환할 수 있는지 여부를 제어합니다.	Ctrl-Z | Ctrl-Shift-Z | Ctrl-Y	cancelBrowserUndoInAce
선택 영역을 가운데로 정렬합니다.	Ctrl-L	centerselection
줄의 내용을 복사하고 복사한 내용을 한 줄 아래에 붙여 넣습니다.	Alt-Shift-Down	copylinesdown
줄의 내용을 복사하고 복사한 내용을 한 줄 위에 붙여 넣습니다.	Alt-Shift-Up	copylinesup
선택 영역을 잘라내거나 선택 영역이 없으면 공백 하나 삭제	Shift-Delete	cut_or_delete

설명	키 바인딩	명령
공백을 하나 삭제합니다.	Delete	del
선택 영역의 내용을 복사하고 복사한 내용을 선택 영역 바로 뒤에 붙여넣습니다.	Ctrl-Shift-D	duplicateSelection
선택 영역에 현재 행의 내용을 포함합니다.	Ctrl-Shift-L	expandtoline
선택 영역에 일치하는 다음 기호까지 포함합니다.	Ctrl-Shift-M	expandToMatching
선택한 코드를 접습니다. 접힌 단위가 선택된 경우 펼칩니다.	Alt-L | Ctrl-F1	fold
접을 수 있는 모든 요소를 접습니다.	Ctrl-Command-Option-0	foldall
현재 선택 범위를 제외하고 접을 수 있는 모든 요소를 접습니다.	Alt-0	foldOther
한 줄 아래로 이동합니다.	Down	golinedown
한 줄 위로 이동합니다.	Up	golineup
로그 파일의 끝 부분으로 이동합니다.	Ctrl-End	gotoend
한 칸씩 왼쪽으로 이동합니다.	Left	gotoleft
현재 줄의 끝으로 이동합니다.	Alt-Right | End	gotolineend
현재 줄의 처음으로 이동합니다.	Alt-Left | Home	gotolinestart
다음 오류로 이동합니다.	Alt-E	goToNextError

설명	키 바인딩	명령
한 페이지 아래로 이동합니다.	`Page Down`	`gotopagedown`
한 페이지 위로 이동합니다.	`Page Up`	`gotopageup`
이전 오류로 이동합니다.	`Alt-Shift-E`	`goToPreviousError`
오른쪽으로 공백 하나 이동합니다.	`Right`	`gotoright`
로그 파일의 시작 부분으로 이동합니다.	`Ctrl-Home`	`gotostart`
왼쪽으로 한 단어 이동합니다.	`Ctrl-Left`	`gotowordleft`
오른쪽으로 한 단어 이동합니다.	`Ctrl-Right`	`gotowordright`
선택 영역을 한 탭 들여씁니다.	`Tab`	`indent`
현재 범위에서 쌍을 이루는 기호로 이동합니다.	`Ctrl-P`	`jumptomatching`
글꼴 크기를 늘립니다.	`Ctrl-+ | Ctrl-=`	`largerfont`
커서의 왼쪽에 있는 숫자를 1씩 줄입니다(숫자인 경우).	`Ctrl-Shift-Down`	`modifyNumberDown`
커서의 왼쪽에 있는 숫자를 1씩 늘립니다(숫자인 경우).	`Ctrl-Shift-Up`	`modifyNumberUp`
선택 영역을 한 줄 아래로 이동합니다.	`Alt-Down`	`movelinesdown`
선택 영역을 한 줄 위로 이동합니다.	`Alt-Up`	`movelinesup`
선택 영역을 한 탭 내어씁니다.	`Shift-Tab`	`outdent`

설명	키 바인딩	명령
덮어쓰기 모드를 켜거나 켜진 경우 끕니다.	Insert	overwrite
한 페이지 아래로 이동합니다.	Option-Page Down	pagedown
한 페이지 위로 이동합니다.	Option-Page Up	pageup
현재 줄의 내용을 삭제합니다.	Ctrl-D	removeline
커서에서 현재 줄의 끝까지 삭제합니다.	Alt-Delete	removetolineend
커서에서 현재 줄의 시작까지 삭제합니다.	Alt-Backspace	removetolinestart
커서의 왼쪽에 있는 단어를 삭제합니다.	Ctrl-Backspace	removewordleft
커서의 오른쪽에 있는 단어를 삭제합니다.	Ctrl-Delete	removewordright
이전에 기록한 키 입력을 재생합니다.	Ctrl-Shift-E	replaymacro
현재 파일을 한 줄씩 아래로 스크롤합니다.	Ctrl-Down	scrolldown
현재 파일을 한 줄씩 위로 스크롤합니다.	Ctrl-Up	scrollup
선택 가능한 내용을 모두 선택합니다.	Ctrl-A	selectall
선택 영역에 다음 줄 아래의 내용을 포함합니다.	Shift-Down	selectdown
선택 영역에 다음 공백 왼쪽의 내용을 포함합니다.	Shift-Left	selectleft

설명	키 바인딩	명령
커서부터 현재 줄의 나머지 부분을 선택 영역에 포함합니다.	Shift-End	selectlineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Shift-Home	selectlinestart
선택 영역의 뒤에 일치하는 선택 영역을 더 포함합니다.	Ctrl-Alt-Right	selectMoreAfter
선택 영역의 앞에 일치하는 선택 영역을 더 포함합니다.	Ctrl-Alt-Left	selectMoreBefore
선택 영역의 뒤에 있는 일치하는 다음 선택 영역을 포함합니다.	Ctrl-Alt-Shift-Right	selectNextAfter
선택 영역의 앞에 있는 일치하는 다음 선택 영역을 포함합니다.	Ctrl-Alt-Shift-Left	selectNextBefore
일치하는 다음 선택 항목을 선택하거나 찾습니다.	Alt-K	selectOrFindNext
일치하는 이전 선택 항목 선택하거나 찾습니다.	Alt-Shift-K	selectOrFindPrevious
커서에서 현재 페이지의 끝까지 선택 영역에 포함합니다.	Shift-Page Down	selectpagedown
커서에서 현재 페이지의 처음까지 선택 영역에 포함합니다.	Shift-Page Up	selectpageup
선택 영역에 커서 오른쪽의 다음 공백를 포함합니다.	Shift-Right	selectright
커서에서 현재 파일의 끝까지 선택 영역에 포함합니다.	Ctrl-Shift-End	selecttoend

설명	키 바인딩	명령
커서에서 현재 줄의 끝까지 선택 영역에 포함합니다.	Alt-Shift-Right	selecttolineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Alt-Shift-Left	selecttolinestart
커서에서 현재 범위의 일치하는 다음 기호까지 포함합니다.	Ctrl-Shift-P	selecttomatching
커서에서 현재 파일의 처음까지 선택 영역에 포함합니다.	Ctrl-Shift-Home	selecttostart
선택 영역에 다음 줄 위의 내용을 포함합니다.	Shift-Up	selectup
선택 영역에 커서 왼쪽의 다음 단어를 포함합니다.	Ctrl-Shift-Left	selectwordleft
선택 영역에 커서 오른쪽의 다음 단어를 포함합니다.	Ctrl-Shift-Right	selectwordright
[기본 설정(Preferences)] 탭을 표시합니다.	Ctrl-,	showSettingsMenu
이전 선택 영역을 모두 지웁니다.	Esc	singleSelection
글꼴 크기를 줄입니다.	Ctrl--	smallerfont
여러 줄을 선택한 경우 정렬된 순서로 다시 정렬합니다.	Ctrl-Alt-S	sortlines
현재 줄의 끝에 커서를 추가합니다.	Ctrl-Alt-L	splitIntoLines
커서의 내용을 줄의 끝에 새 줄로 이동합니다.	Ctrl-O	splitline

설명	키 바인딩	명령
블록 주석 문자로 선택 영역을 둘러싸거나 해당 문자가 있는 경우 제거합니다.	Ctrl-Shift-/	toggleBlockComment
선택한 각 줄의 처음에 줄 설명 문자를 추가하거나, 줄 설명 문자가 있는 경우 제거합니다.	Ctrl-/	togglecomment
코드를 접거나, 코드 접기를 제거합니다(있는 경우).	F2	toggleFoldWidget
상위 코드를 접거나, 접기를 제거합니다(있는 경우).	Alt-F2	toggleParentFoldWidget
키 입력 기록을 시작하거나 이미 기록 중인 경우 중지합니다.	Ctrl-Alt-E	togglerecording
단어를 줄 바꿈하거나 이미 줄 바꿈이 있는 경우 단어 줄 바꿈을 중지합니다.	Ctrl-Q	toggleWordWrap
선택 영역을 모두 소문자로 변경합니다.	Ctrl-Shift-U	tolowercase
선택 영역을 모두 대문자로 변경합니다.	Ctrl-U	touppercase
선택 영역의 대소문자를 바꿉니다.	Alt-X	transposeletters
선택한 코드를 펼칩니다.	Alt-Shift-L | Ctrl-Shift-F1	unfold
전체 파일의 코드 접기를 펼칩니다.	Alt-Shift-0	unfoldall

emmet

설명	키 바인딩	명령
간단한 수학 표현식(예: 2*4 또는 10/2)과 그 결과를 출력합니다.	Shift-Ctrl-Y	emmet_evaluate_math_expression
현재 파일의 구문에 따라 CSS와 유사한 약어를 HTML, XML 또는 CSS 코드로 확장합니다.	Ctrl-Alt-E	emmet_expand_abbreviation
탭 정지에 의해 확장된 CSS와 유사한 약어를 순회합니다.	Tab	emmet_expand_abbreviation_with_tab
편집 가능한 다음 코드 부분으로 이동합니다.	Shift-Ctrl-.	emmet_select_next_item
이전 편집 가능한 코드 부분으로 이동합니다.	Shift-Ctrl-,	emmet_select_previous_item
약어를 확장한 다음 생성된 코드 조각의 마지막 요소 내에 현재 선택 영역을 배치합니다.	Shift-Ctrl-A	emmet_wrap_with_abbreviation

터미널

설명	키 바인딩	명령
새 [터미널(Terminal)] 탭을 엽니다.	Alt-T	openterminal
편집기와 [터미널(Terminal)] 탭 간을 전환합니다.	Alt-S	switchterminal

실행 및 디버깅

설명	키 바인딩	명령
현재 파일을 빌드합니다.	Ctrl-B	build
현재 일시 중지된 프로세스를 다시 시작합니다.	F8	resume
현재 애플리케이션을 실행하거나 디버그합니다.	Alt-F5	run
마지막 실행 파일을 실행하거나 디버그합니다.	F5	runlast
스택 옆에 있는 함수로 진행합니다.	F11	stepinto
현재 함수 범위를 종료합니다.	Shift-F11	stepout
스택의 현재 표현식으로 진행합니다.	F10	stepover
현재 애플리케이션의 실행 또는 디버깅을 중지합니다.	Shift-F5	stop
현재 파일의 빌드를 중지합니다.	Ctrl-Shift-C	stopbuild

AWS Cloud9 통합 개발 환경(IDE)에 대한 Windows/Linux Vim 키 바인딩 참조

다음은 AWS Cloud9 IDE의 Windows/Linux 운영 체제용 Vim 키보드 모드 키 바인딩 목록입니다.

자세한 내용은 AWS Cloud9 IDE에서 다음을 참조하세요.

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. [기본 설정(Preferences)] 탭에서 [키 바인딩(Keybindings)]을 선택합니다.

3. 용키보드 모드를 선택하고이맥스.

4. [운영 체제(Operating System)]에서 [Windows / Linux]를 선택합니다.

키 바인딩 작업도 참조하세요.

- 일반
- 탭
- 패널
- 코드 편집기
- emmet
- 터미널
- 실행 및 디버그

일반

설명	키 바인딩	명령
선택 영역을 시계 표현식으로 추가합니다.	`Ctrl-Shift-C`	`addwatchfromselection`
클립보드에서 잘라낸 선택 영역을 제거합니다.	`Esc`	`clearcut`
코드 완성 컨텍스트 메뉴를 표시합니다.	`Ctrl-Space | Alt-Space`	`complete`
코드를 완료한 다음 덮어씁니다.	`Ctrl-Shift-Space | Alt-Shift-Space`	`completeoverwrite`
선택 영역을 클립보드로 복사합니다.	`Ctrl-C`	`copy`
선택 영역을 클립보드로 잘라냅니다.	`Ctrl-X`	`cut`

설명	키 바인딩	명령
코드를 확장합니다(해당하는 경우).	Tab	expandSnippet
현재 문서의 찾기 및 바꾸기 막대를 표시합니다.	Ctrl-F	find
현재 문서에서 일치하는 항목을 모두 선택합니다.	Ctrl-Alt-K	findall
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 다음 항목으로 이동합니다.	Ctrl-K	findnext
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 이전 항목으로 이동합니다.	Ctrl-Shift-K	findprevious
편집기에서 활성 파일의 삽입 점에 기호에 대한 알려진 모든 참조를 표시합니다.	Shift-F3	findReferences
[환경(Environment) 창을 연 다음 파일 목록을 활성 상태로 만듭니다.	Shift-Esc	focusTree
선택한 JavaScript 코드를 다시 포맷합니다.	Ctrl-Shift-B	formatcode
줄로 이동 상자를 표시합니다.	Ctrl-G	gotoline
찾기 및 바꾸기 막대가 표시된 경우 숨깁니다.	Esc	hidesearchreplace
커서 위치에 있는 변수 또는 함수의 정의로 이동합니다.	F3	jumptodef

설명	키 바인딩	명령
AWS 리소스 창의 Lambda 섹션에서 로컬 Lambda 함수를 선택한 경우 해당 함수를 원격 함수로 Lambda에 업로드하려고 시도합니다.	`Ctrl-Shift-U`	`lambdaUploadFunction`
새 파일을 만듭니다.	`Alt-N`	`newfile`
[기본 설정(Preferences)] 탭을 표시합니다.	`Ctrl-,`	`openpreferences`
[터미널(Terminal)] 탭을 연 다음 파일 목록에서 선택한 파일의 상위 폴더로 전환합니다.	`Alt-L`	`opentermhere`
클립보드의 현재 내용을 커서 위치에 붙여 넣습니다.	`Ctrl-V`	`paste`
오류 수정에 대한 제안 사항을 표시합니다.	`Ctrl-F3`	`quickfix`
마지막 작업을 다시 실행합니다.	`Ctrl-Shift-Z | Ctrl-Y`	`redo`
미리 보기 창을 새로 고칩니다.	`Ctrl-Enter`	`reloadpreview`
선택 항목의 이름 바꾸기 리팩터링을 시작합니다.	`Ctrl-Alt-R`	`renameVar`
현재 문서의 찾기 및 바꾸기 모음을 표시하고 [바꾸기(Replace With)] 표현식을 사용합니다.	`Alt-Shift-F | Ctrl-H`	`replace`
초기화 스크립트를 다시 실행합니다.	`Ctrl-Enter`	`rerunInitScript`

설명	키 바인딩	명령
환경을 다시 시작합니다.	Ctrl-R	restartc9
현재 파일을 마지막으로 저장한 버전으로 재설정합니다.	Ctrl-Shift-Q	reverttosaved
열려 있는 각 파일을 저장된 버전으로 재설정합니다.	Alt-Shift-Q	reverttosavedall
현재 파일을 디스크에 저장합니다.	Ctrl-S	save
현재 파일을 다른 파일 이름을 사용하여 디스크에 저장합니다.	Ctrl-Shift-S	saveas
여러 파일의 찾기 및 바꾸기 모음을 표시합니다.	Ctrl-Shift-F	searchinfiles
[프로세스 목록(Process List)] 대화 상자를 표시합니다.	Ctrl-Alt-P	showprocesslist
마지막 작업을 실행 취소합니다.	Ctrl-Z	undo

탭

설명	키 바인딩	명령
현재 창에서 현재 탭을 제외하고 열려 있는 탭을 모두 닫습니다.	Ctrl-Alt-W	closeallbutme
모든 창에서 열려 있는 탭을 모두 닫습니다.	Alt-Shift-W	closealltabs
현재 창을 닫습니다.	Ctrl-W	closepane

설명	키 바인딩	명령
현재 탭을 닫습니다.	Alt-W	closetab
아래로 창을 하나 이동합니다.	Ctrl-Meta-Down	gotopanedown
왼쪽으로 창을 하나 이동합니다.	Ctrl-Meta-Left	gotopaneleft
오른쪽으로 창을 하나 이동합니다.	Ctrl-Meta-Right	gotopaneright
위로 창을 하나 이동합니다.	Ctrl-Meta-Up	gottopaneup
왼쪽으로 한 탭 이동합니다.	Ctrl-[	gototableft
오른쪽으로 한 탭 이동합니다.	Ctrl-]	gototabright
현재 탭을 한 창 아래로 이동하거나 탭이 이미 맨 아래에 있는 경우 분할 탭을 만듭니다.	Ctrl-Meta-Down	movetabdown
현재 탭을 왼쪽으로 이동하거나 탭이 이미 맨 왼쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Ctrl-Meta-Left	movetableft
현재 탭을 오른쪽으로 이동하거나 탭이 이미 맨 오른쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Ctrl-Meta-Right	movetabright
현재 탭을 한 창 위로 이동하거나 탭이 이미 맨 위에 있는 경우 분할 탭을 만듭니다.	Ctrl-Meta-Up	movetabup
다음 창으로 이동합니다.	Ctrl-`	nextpane
다음 탭으로 이동합니다.	Ctrl-Tab | Alt-`	nexttab

설명	키 바인딩	명령
이전 창으로 이동합니다.	Ctrl-Shift-`	previouspane
이전 탭으로 이동합니다.	Ctrl-Shift-Tab | Alt-Shift-`	previoustab
마지막 탭으로 돌아갑니다.	Esc	refocusTab
마지막 탭을 다시 엽니다.	Alt-Shift-T	reopenLastTab
파일 트리에 현재 탭을 표시합니다.	Ctrl-Shift-L	revealtab
10번째 탭으로 이동합니다.	Ctrl-0	tab0
첫 번째 탭으로 이동합니다.	Ctrl-1	tab1
두 번째 탭으로 이동합니다.	Ctrl-2	tab2
세 번째 탭으로 이동합니다.	Ctrl-3	tab3
네 번째 탭으로 이동합니다.	Ctrl-4	tab4
다섯 번째 탭으로 이동합니다.	Ctrl-5	tab5
여섯 번째 탭으로 이동합니다.	Ctrl-6	tab6
일곱 번째 탭으로 이동합니다.	Ctrl-7	tab7
여덟 번째 탭으로 이동합니다.	Ctrl-8	tab8
아홉 번째 탭으로 이동합니다.	Ctrl-9	tab9

패널

설명	키 바인딩	명령
[바로 가기(Go to Anything)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-E|Ctrl-P`	`gotoanything`
[명령으로 이동(Go to Command)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-.|F1`	`gotocommand`
[파일로 이동(Go to File)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-O`	`gotofile`
[기호로 이동(Go to Symbol)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-Shift-O`	`gotosymbol`
[개요(Outline)] 창을 표시합니다.	`Ctrl-Shift-E`	`outline`
숨겨져 있는 경우 [콘솔(Console)] 창을 표시하거나 표시된 경우 숨깁니다.	`F6`	`toggleconsole`
숨겨져 있는 경우 [환경(Environment)] 창을 표시하거나 표시된 경우 숨깁니다.	`Ctrl-I`	`toggletree`

코드 편집기

설명	키 바인딩	명령
활성 커서 한 줄 위에 커서를 추가하거나 커서가 이미 추가된	`Ctrl-Alt-Up`	`addCursorAbove`

설명	키 바인딩	명령
경우 해당 커서 위에 다른 커서를 추가합니다.		
활성 커서 한 줄 위에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 위로 이동합니다.	Ctrl-Alt-Shift-Up	addCursorAboveSkipCurrent
활성 커서 한 줄 아래에 커서를 추가하거나 커서가 이미 추가된 경우 해당 커서 아래에 다른 커서를 추가합니다.	Ctrl-Alt-Down	addCursorBelow
활성 커서 아래에 두 번째 커서를 한 줄 추가합니다. 또는 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 아래로 이동합니다.	Ctrl-Alt-Shift-Down	addCursorBelowSkipCurrent
정렬이 잘못된 경우 모든 커서를 각 행의 활성 커서와 동일한 공간으로 이동합니다.	Ctrl-Alt-A	alignCursors
단일 공백을 백스페이스합니다.	Shift-Backspace | Backspace	backspace
선택 영역을 한 탭 들여씁니다.	Ctrl-]	blockindent
선택 영역을 한 탭 내어씁니다.	Ctrl-[	blockoutdent
포커스를 편집기에서 IDE의 다른 곳으로 전환할 수 있는지 여부를 제어합니다.	Ctrl-Z | Ctrl-Shift-Z | Ctrl-Y	cancelBrowserUndoInAce
줄의 내용을 복사하고 복사한 내용을 한 줄 아래에 붙여 넣습니다.	Alt-Shift-Down	copylinesdown

설명	키 바인딩	명령
줄의 내용을 복사하고 복사한 내용을 한 줄 위에 붙여 넣습니다.	Alt-Shift-Up	copylinesup
선택 영역을 자릅니다. 선택 영역이 없으면 공백을 하나 삭제합니다.	Shift-Delete	cut_or_delete
공백을 하나 삭제합니다.	Delete	del
선택 영역의 내용을 복사하고 복사한 내용을 선택 영역 바로 뒤에 붙여넣습니다.	Ctrl-Shift-D	duplicateSelection
선택 영역에 현재 행의 내용을 포함합니다.	Ctrl-Shift-L	expandtoline
선택 영역에 일치하는 다음 기호까지 포함합니다.	Ctrl-Shift-M	expandToMatching
선택한 코드를 접습니다. 접힌 단위가 선택된 경우 펼칩니다.	Alt-L | Ctrl-F1	fold
현재 선택 범위를 제외하고 접을 수 있는 모든 요소를 접습니다.	Alt-0	foldOther
한 줄 아래로 이동합니다.	Down	golinedown
한 줄 위로 이동합니다.	Up	golineup
로그 파일의 끝 부분으로 이동합니다.	Ctrl-End	gotoend
한 칸씩 왼쪽으로 이동합니다.	Left	gotoleft
현재 줄의 끝으로 이동합니다.	Alt-Right | End	gotolineend

설명	키 바인딩	명령
현재 줄의 처음으로 이동합니다.	Alt-Left | Home	gotolinestart
다음 오류로 이동합니다.	Alt-E	goToNextError
한 페이지 아래로 이동합니다.	Page Down	gotopagedown
한 페이지 위로 이동합니다.	Page Up	gotopageup
이전 오류로 이동합니다.	Alt-Shift-E	goToPreviousError
오른쪽으로 공백 하나 이동합니다.	Right	gotoright
로그 파일의 시작 부분으로 이동합니다.	Ctrl-Home	gotostart
왼쪽으로 한 단어 이동합니다.	Ctrl-Left	gotowordleft
오른쪽으로 한 단어 이동합니다.	Ctrl-Right	gotowordright
선택 영역을 한 탭 들여씁니다.	Tab	indent
현재 범위에서 쌍을 이루는 기호로 이동합니다.	Ctrl-P	jumptomatching
글꼴 크기를 늘립니다.	Ctrl-+ | Ctrl-=	largerfont
커서의 왼쪽에 있는 숫자를 1씩 줄입니다(숫자인 경우).	Ctrl-Shift-Down	modifyNumberDown
커서의 왼쪽에 있는 숫자를 1씩 늘립니다(숫자인 경우).	Ctrl-Shift-Up	modifyNumberUp
선택 영역을 한 줄 아래로 이동합니다.	Alt-Down	movelinesdown

설명	키 바인딩	명령
선택 영역을 한 줄 위로 이동합니다.	Alt-Up	movelinesup
선택 영역을 한 탭 내어씁니다.	Shift-Tab	outdent
덮어쓰기 모드를 켜거나 켜진 경우 끕니다.	Insert	overwrite
현재 줄의 내용을 삭제합니다.	Ctrl-D	removeline
커서에서 현재 줄의 끝까지 삭제합니다.	Alt-Delete	removetolineend
커서에서 현재 줄의 시작까지 삭제합니다.	Alt-Backspace	removetolinestart
커서의 왼쪽에 있는 단어를 삭제합니다.	Ctrl-Backspace	removewordleft
커서의 오른쪽에 있는 단어를 삭제합니다.	Ctrl-Delete	removewordright
이전에 기록한 키 입력을 재생합니다.	Ctrl-Shift-E	replaymacro
현재 파일을 한 줄씩 아래로 스크롤합니다.	Ctrl-Down	scrolldown
현재 파일을 한 줄씩 위로 스크롤합니다.	Ctrl-Up	scrollup
선택 가능한 내용을 모두 선택합니다.	Ctrl-A	selectall
선택 영역에 다음 줄 아래의 내용을 포함합니다.	Shift-Down	selectdown

설명	키 바인딩	명령
선택 영역에 다음 공백 왼쪽의 내용을 포함합니다.	Shift-Left	selectleft
커서부터 현재 줄의 나머지 부분을 선택 영역에 포함합니다.	Shift-End	selectlineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Shift-Home	selectlinestart
선택 영역의 뒤에 일치하는 선택 영역을 더 포함합니다.	Ctrl-Alt-Right	selectMoreAfter
선택 영역의 앞에 일치하는 선택 영역을 더 포함합니다.	Ctrl-Alt-Left	selectMoreBefore
선택 영역의 뒤에 있는 일치하는 다음 선택 영역을 포함합니다.	Ctrl-Alt-Shift-Right	selectNextAfter
선택 영역의 앞에 있는 일치하는 다음 선택 영역을 포함합니다.	Ctrl-Alt-Shift-Left	selectNextBefore
일치하는 다음 선택 항목을 선택하거나 찾습니다.	Alt-K	selectOrFindNext
일치하는 이전 선택 항목 선택하거나 찾습니다.	Alt-Shift-K	selectOrFindPrevious
커서에서 현재 페이지의 끝까지 선택 영역에 포함합니다.	Shift-Page Down	selectpagedown
커서에서 현재 페이지의 처음까지 선택 영역에 포함합니다.	Shift-Page Up	selectpageup
선택 영역에 커서 오른쪽의 다음 공백를 포함합니다.	Shift-Right	selectright

설명	키 바인딩	명령
커서에서 현재 파일의 끝까지 선택 영역에 포함합니다.	`Ctrl-Shift-End`	`selecttoend`
커서에서 현재 줄의 끝까지 선택 영역에 포함합니다.	`Alt-Shift-Right`	`selecttolineend`
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	`Alt-Shift-Left`	`selecttolinestart`
커서에서 현재 범위의 일치하는 다음 기호까지 포함합니다.	`Ctrl-Shift-P`	`selecttomatching`
커서에서 현재 파일의 처음까지 선택 영역에 포함합니다.	`Ctrl-Shift-Home`	`selecttostart`
선택 영역에 다음 줄 위의 내용을 포함합니다.	`Shift-Up`	`selectup`
선택 영역에 커서 왼쪽의 다음 단어를 포함합니다.	`Ctrl-Shift-Left`	`selectwordleft`
선택 영역에 커서 오른쪽의 다음 단어를 포함합니다.	`Ctrl-Shift-Right`	`selectwordright`
[기본 설정(Preferences)] 탭을 표시합니다.	`Ctrl-,`	`showSettingsMenu`
이전 선택 영역을 모두 지웁니다.	`Esc`	`singleSelection`
글꼴 크기를 줄입니다.	`Ctrl--`	`smallerfont`
여러 줄을 선택한 경우 정렬된 순서로 다시 정렬합니다.	`Ctrl-Alt-S`	`sortlines`
현재 줄의 끝에 커서를 추가합니다.	`Ctrl-Alt-L`	`splitIntoLines`

설명	키 바인딩	명령
블록 주석 문자로 선택 영역을 둘러싸거나 해당 문자가 있는 경우 제거합니다.	Ctrl-Shift-/	toggleBlockComment
선택한 각 줄의 처음에 줄 설명 문자를 추가하거나, 줄 설명 문자가 있는 경우 제거합니다.	Ctrl-/	togglecomment
코드를 접거나, 코드 접기를 제거합니다(있는 경우).	F2	toggleFoldWidget
상위 코드를 접거나, 접기를 제거합니다(있는 경우).	Alt-F2	toggleParentFoldWidget
키 입력 기록을 시작하거나 이미 기록 중인 경우 중지합니다.	Ctrl-Alt-E	togglerecording
단어를 줄 바꿈하거나 이미 줄 바꿈이 있는 경우 단어 줄 바꿈을 중지합니다.	Ctrl-Q	toggleWordWrap
선택 영역을 모두 소문자로 변경합니다.	Ctrl-Shift-U	tolowercase
선택 영역을 모두 대문자로 변경합니다.	Ctrl-U	touppercase
선택 영역의 대소문자를 바꿉니다.	Alt-X	transposeletters
선택한 코드를 펼칩니다.	Alt-Shift-L | Ctrl-Shift-F1	unfold
전체 파일의 코드 접기를 펼칩니다.	Alt-Shift-0	unfoldall

emmet

설명	키 바인딩	명령
간단한 수학 표현식(예: 2*4 또는 10/2)과 그 결과를 출력합니다.	Shift-Ctrl-Y	emmet_evaluate_math_expression
현재 파일의 구문에 따라 CSS와 유사한 약어를 HTML, XML 또는 CSS 코드로 확장합니다.	Ctrl-Alt-E	emmet_expand_abbreviation
탭 정지에 의해 확장된 CSS와 유사한 약어를 순회합니다.	Tab	emmet_expand_abbreviation_with_tab
편집 가능한 다음 코드 부분으로 이동합니다.	Shift-Ctrl-.	emmet_select_next_item
이전 편집 가능한 코드 부분으로 이동합니다.	Shift-Ctrl-,	emmet_select_previous_item
약어를 확장한 다음 생성된 코드 조각의 마지막 요소 내에 현재 선택 영역을 배치합니다.	Shift-Ctrl-A	emmet_wrap_with_abbreviation

터미널

설명	키 바인딩	명령
새 [터미널(Terminal)] 탭을 엽니다.	Alt-T	openterminal
편집기와 [터미널(Terminal)] 탭 간을 전환합니다.	Alt-S	switchterminal

실행 및 디버깅

설명	키 바인딩	명령
현재 파일을 빌드합니다.	Ctrl-B	build
현재 일시 중지된 프로세스를 다시 시작합니다.	F8	resume
현재 애플리케이션을 실행하거나 디버그합니다.	Alt-F5	run
마지막 실행 파일을 실행하거나 디버그합니다.	F5	runlast
스택 옆에 있는 함수로 진행합니다.	F11	stepinto
현재 함수 범위를 종료합니다.	Shift-F11	stepout
스택의 현재 표현식으로 진행합니다.	F10	stepover
현재 애플리케이션의 실행 또는 디버깅을 중지합니다.	Shift-F5	stop
현재 파일의 빌드를 중지합니다.	Ctrl-Shift-C	stopbuild

AWS Cloud9 통합 개발 환경(IDE)에 대한 Windows/Linux Emacs 키 바인딩 참조

다음은 AWS Cloud9 IDE의 Windows/Linux 운영 체제용 Emacs 키보드 모드 키 바인딩 목록입니다.

자세한 내용은 AWS Cloud9 IDE에서 다음을 참조하세요.

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. [기본 설정(Preferences)] 탭에서 [키 바인딩(Keybindings)]을 선택합니다.

3. 용키보드 모드를 선택하고이맥스.

4. [운영 체제(Operating System)]에서 [Windows / Linux]를 선택합니다.

키 바인딩 작업도 참조하세요.

- 일반
- 탭
- 패널
- 코드 편집기
- emmet
- 터미널
- 실행 및 디버그

일반

설명	키 바인딩	명령
선택 영역을 시계 표현식으로 추가합니다.	`Ctrl-Shift-C`	`addwatchfromselection`
클립보드에서 잘라낸 선택 영역을 제거합니다.	`Esc`	`clearcut`
코드 완성 컨텍스트 메뉴를 표시합니다.	`Ctrl-Space | Alt-Space`	`complete`
코드를 완료한 다음 덮어씁니다.	`Ctrl-Shift-Space | Alt-Shift-Space`	`completeoverwrite`
선택 영역을 클립보드로 복사합니다.	`Ctrl-C`	`copy`
선택 영역을 클립보드로 잘라냅니다.	`Ctrl-X`	`cut`

설명	키 바인딩	명령
코드를 확장합니다(해당하는 경우).	Tab	expandSnippet
현재 문서의 찾기 및 바꾸기 막대를 표시합니다.	Ctrl-F	find
현재 문서에서 일치하는 항목을 모두 선택합니다.	Ctrl-Alt-K	findall
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 다음 항목으로 이동합니다.	Ctrl-K	findnext
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 이전 항목으로 이동합니다.	Ctrl-Shift-K	findprevious
편집기에서 활성 파일의 삽입점에 기호에 대한 알려진 모든 참조를 표시합니다.	Shift-F3	findReferences
[환경(Environment) 창을 연 다음 파일 목록을 활성 상태로 만듭니다.	Shift-Esc	focusTree
선택한 JavaScript 코드를 다시 포맷합니다.	Ctrl-Shift-B	formatcode
줄로 이동 상자를 표시합니다.	Ctrl-G	gotoline
찾기 및 바꾸기 막대가 표시된 경우 숨깁니다.	Esc	hidesearchreplace
커서 위치에 있는 변수 또는 함수의 정의로 이동합니다.	F3	jumptodef

설명	키 바인딩	명령
AWS 리소스 창의 Lambda 섹션에서 로컬 Lambda 함수를 선택한 경우 해당 함수를 원격 함수로 Lambda에 업로드하려고 시도합니다.	`Ctrl-Shift-U`	`lambdaUploadFunction`
새 파일을 만듭니다.	`Alt-N`	`newfile`
[기본 설정(Preferences)] 탭을 표시합니다.	`Ctrl-,`	`openpreferences`
[터미널(Terminal)] 탭을 연 다음 파일 목록에서 선택한 파일의 상위 폴더로 전환합니다.	`Alt-L`	`opentermhere`
클립보드의 현재 내용을 커서 위치에 붙여 넣습니다.	`Ctrl-V`	`paste`
오류 수정에 대한 제안 사항을 표시합니다.	`Ctrl-F3`	`quickfix`
마지막 작업을 다시 실행합니다.	`Ctrl-Shift-Z | Ctrl-Y`	`redo`
미리 보기 창을 새로 고칩니다.	`Ctrl-Enter`	`reloadpreview`
선택 항목의 이름 바꾸기 리팩터링을 시작합니다.	`Ctrl-Alt-R`	`renameVar`
현재 문서의 찾기 및 바꾸기 모음을 표시하고 [바꾸기(Replace With)] 표현식을 사용합니다.	`Alt-Shift-F | Ctrl-H`	`replace`
초기화 스크립트를 다시 실행합니다.	`Ctrl-Enter`	`rerunInitScript`

설명	키 바인딩	명령
환경을 다시 시작합니다.	Ctrl-R	restartc9
현재 파일을 마지막으로 저장한 버전으로 재설정합니다.	Ctrl-Shift-Q	reverttosaved
열려 있는 각 파일을 저장된 버전으로 재설정합니다.	Alt-Shift-Q	reverttosavedall
현재 파일을 디스크에 저장합니다.	Ctrl-S	save
현재 파일을 다른 파일 이름을 사용하여 디스크에 저장합니다.	Ctrl-Shift-S	saveas
여러 파일의 찾기 및 바꾸기 모음을 표시합니다.	Ctrl-Shift-F	searchinfiles
[프로세스 목록(Process List)] 대화 상자를 표시합니다.	Ctrl-Alt-P	showprocesslist
마지막 작업을 실행 취소합니다.	Ctrl-Z	undo

탭

설명	키 바인딩	명령
현재 창에서 현재 탭을 제외하고 열려 있는 탭을 모두 닫습니다.	Ctrl-Alt-W	closeallbutme
모든 창에서 열려 있는 탭을 모두 닫습니다.	Alt-Shift-W	closealltabs
현재 창을 닫습니다.	Ctrl-W	closepane

설명	키 바인딩	명령
현재 탭을 닫습니다.	Alt-W	closetab
아래로 창을 하나 이동합니다.	Ctrl-Meta-Down	gotopanedown
왼쪽으로 창을 하나 이동합니다.	Ctrl-Meta-Left	gotopaneleft
오른쪽으로 창을 하나 이동합니다.	Ctrl-Meta-Right	gotopaneright
위로 창을 하나 이동합니다.	Ctrl-Meta-Up	gottopaneup
왼쪽으로 한 탭 이동합니다.	Ctrl-[	gototableft
오른쪽으로 한 탭 이동합니다.	Ctrl-]	gototabright
현재 탭을 한 창 아래로 이동하거나 탭이 이미 맨 아래에 있는 경우 분할 탭을 만듭니다.	Ctrl-Meta-Down	movetabdown
현재 탭을 왼쪽으로 이동하거나 탭이 이미 맨 왼쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Ctrl-Meta-Left	movetableft
현재 탭을 오른쪽으로 이동하거나 탭이 이미 맨 오른쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Ctrl-Meta-Right	movetabright
현재 탭을 한 창 위로 이동하거나 탭이 이미 맨 위에 있는 경우 분할 탭을 만듭니다.	Ctrl-Meta-Up	movetabup
다음 창으로 이동합니다.	Ctrl-`	nextpane
다음 탭으로 이동합니다.	Ctrl-Tab | Alt-`	nexttab

설명	키 바인딩	명령
이전 창으로 이동합니다.	Ctrl-Shift-`	previouspane
이전 탭으로 이동합니다.	Ctrl-Shift-Tab | Alt-Shift-`	previoustab
마지막 탭으로 돌아갑니다.	Esc	refocusTab
마지막 탭을 다시 엽니다.	Alt-Shift-T	reopenLastTab
파일 트리에 현재 탭을 표시합니다.	Ctrl-Shift-L	revealtab
10번째 탭으로 이동합니다.	Ctrl-0	tab0
첫 번째 탭으로 이동합니다.	Ctrl-1	tab1
두 번째 탭으로 이동합니다.	Ctrl-2	tab2
세 번째 탭으로 이동합니다.	Ctrl-3	tab3
네 번째 탭으로 이동합니다.	Ctrl-4	tab4
다섯 번째 탭으로 이동합니다.	Ctrl-5	tab5
여섯 번째 탭으로 이동합니다.	Ctrl-6	tab6
일곱 번째 탭으로 이동합니다.	Ctrl-7	tab7
여덟 번째 탭으로 이동합니다.	Ctrl-8	tab8
아홉 번째 탭으로 이동합니다.	Ctrl-9	tab9

패널

설명	키 바인딩	명령
[바로 가기(Go to Anything)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-E|Ctrl-P`	`gotoanything`
[명령으로 이동(Go to Command)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-.|F1`	`gotocommand`
[파일로 이동(Go to File)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-O`	`gotofile`
[기호로 이동(Go to Symbol)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-Shift-O`	`gotosymbol`
[개요(Outline)] 창을 표시합니다.	`Ctrl-Shift-E`	`outline`
숨겨져 있는 경우 [콘솔(Console)] 창을 표시하거나 표시된 경우 숨깁니다.	`F6`	`toggleconsole`
숨겨져 있는 경우 [환경(Environment)] 창을 표시하거나 표시된 경우 숨깁니다.	`Ctrl-I`	`toggletree`

코드 편집기

설명	키 바인딩	명령
활성 커서 한 줄 위에 커서를 추가하거나 커서가 이미 추가된	`Ctrl-Alt-Up`	`addCursorAbove`

설명	키 바인딩	명령
경우 해당 커서 위에 다른 커서를 추가합니다.		
활성 커서 한 줄 위에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 위로 이동합니다.	Ctrl-Alt-Shift-Up	addCursorAboveSkipCurrent
활성 커서 한 줄 아래에 커서를 추가하거나 커서가 이미 추가된 경우 해당 커서 아래에 다른 커서를 추가합니다.	Ctrl-Alt-Down	addCursorBelow
활성 커서 한 줄 아래에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 아래로 이동합니다.	Ctrl-Alt-Shift-Down	addCursorBelowSkipCurrent
정렬이 잘못된 경우 모든 커서를 각 행의 활성 커서와 동일한 공간으로 이동합니다.	Ctrl-Alt-A	alignCursors
단일 공백을 백스페이스합니다.	Shift-Backspace | Backspace	backspace
선택 영역을 한 탭 들여씁니다.	Ctrl-]	blockindent
선택 영역을 한 탭 내어씁니다.	Ctrl-[	blockoutdent
포커스를 편집기에서 IDE의 다른 곳으로 전환할 수 있는지 여부를 제어합니다.	Ctrl-Z | Ctrl-Shift-Z | Ctrl-Y	cancelBrowserUndoInAce
줄의 내용을 복사하고 복사한 내용을 한 줄 아래에 붙여 넣습니다.	Alt-Shift-Down	copylinesdown

설명	키 바인딩	명령
줄의 내용을 복사하고 복사한 내용을 한 줄 위에 붙여 넣습니다.	Alt-Shift-Up	copylinesup
선택 영역을 잘라내거나 선택 영역이 없으면 공백 하나 삭제	Shift-Delete	cut_or_delete
공백을 하나 삭제합니다.	Delete	del
선택 영역의 내용을 복사하고 복사한 내용을 선택 영역 바로 뒤에 붙여넣습니다.	Ctrl-Shift-D	duplicateSelection
선택 영역에 현재 줄의 내용을 포함합니다.	Ctrl-Shift-L	expandtoline
선택 영역에 일치하는 다음 기호까지의 내용을 포함합니다.	Ctrl-Shift-M	expandToMatching
선택한 코드를 접습니다. 접힌 단위가 선택된 경우 펼칩니다.	Alt-L | Ctrl-F1	fold
현재 선택 범위를 제외하고 접을 수 있는 모든 요소를 접습니다.	Alt-0	foldOther
한 줄 아래로 이동합니다.	Down	golinedown
한 줄 위로 이동합니다.	Up	golineup
로그 파일의 끝 부분으로 이동합니다.	Ctrl-End	gotoend
한 칸씩 왼쪽으로 이동합니다.	Left	gotoleft
현재 줄의 끝으로 이동합니다.	Alt-Right | End	gotolineend

설명	키 바인딩	명령
현재 줄의 처음으로 이동합니다.	Alt-Left | Home	gotolinestart
다음 오류로 이동합니다.	Alt-E	goToNextError
한 페이지 아래로 이동합니다.	Page Down	gotopagedown
한 페이지 위로 이동합니다.	Page Up	gotopageup
이전 오류로 이동합니다.	Alt-Shift-E	goToPreviousError
오른쪽으로 공백 하나 이동합니다.	Right	gotoright
로그 파일의 시작 부분으로 이동합니다.	Ctrl-Home	gotostart
왼쪽으로 한 단어 이동합니다.	Ctrl-Left	gotowordleft
오른쪽으로 한 단어 이동합니다.	Ctrl-Right	gotowordright
선택 영역을 한 탭 들여씁니다.	Tab	indent
현재 범위에서 쌍을 이루는 기호로 이동합니다.	Ctrl-P	jumptomatching
글꼴 크기를 늘립니다.	Ctrl-+ | Ctrl-=	largerfont
커서의 왼쪽에 있는 숫자를 1씩 줄입니다(숫자인 경우).	Ctrl-Shift-Down	modifyNumberDown
커서의 왼쪽에 있는 숫자를 1씩 늘립니다(숫자인 경우).	Ctrl-Shift-Up	modifyNumberUp
선택 영역을 한 줄 아래로 이동합니다.	Alt-Down	movelinesdown

설명	키 바인딩	명령
선택 영역을 한 줄 위로 이동합니다.	Alt-Up	movelinesup
선택 영역을 한 탭 내어씁니다.	Shift-Tab	outdent
덮어쓰기 모드를 켜거나 켜진 경우 끕니다.	Insert	overwrite
현재 줄의 내용을 삭제합니다.	Ctrl-D	removeline
커서에서 현재 줄의 끝까지 삭제합니다.	Alt-Delete	removetolineend
커서에서 현재 줄의 시작까지 삭제합니다.	Alt-Backspace	removetolinestart
커서의 왼쪽에 있는 단어를 삭제합니다.	Ctrl-Backspace	removewordleft
커서의 오른쪽에 있는 단어를 삭제합니다.	Ctrl-Delete	removewordright
이전에 기록한 키 입력을 재생합니다.	Ctrl-Shift-E	replaymacro
현재 파일을 한 줄씩 아래로 스크롤합니다.	Ctrl-Down	scrolldown
현재 파일을 한 줄씩 위로 스크롤합니다.	Ctrl-Up	scrollup
선택 가능한 내용을 모두 선택합니다.	Ctrl-A	selectall
선택 영역에 다음 줄 아래의 내용을 포함합니다.	Shift-Down	selectdown

설명	키 바인딩	명령
선택 영역에 왼쪽의 다음 공백을 포함합니다.	Shift-Left	selectleft
커서부터 현재 줄의 나머지 부분을 선택 영역에 포함합니다.	Shift-End	selectlineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Shift-Home	selectlinestart
선택 영역의 뒤에 일치하는 선택 영역을 더 포함합니다.	Ctrl-Alt-Right	selectMoreAfter
선택 영역의 앞에 일치하는 선택 영역을 더 포함합니다.	Ctrl-Alt-Left	selectMoreBefore
선택 영역의 뒤에 있는 일치하는 다음 선택 영역을 포함합니다.	Ctrl-Alt-Shift-Right	selectNextAfter
선택 영역의 앞에 있는 일치하는 다음 선택 영역을 포함합니다.	Ctrl-Alt-Shift-Left	selectNextBefore
일치하는 다음 선택 항목을 선택하거나 찾습니다.	Alt-K	selectOrFindNext
일치하는 이전 선택 항목 선택하거나 찾습니다.	Alt-Shift-K	selectOrFindPrevious
커서에서 현재 페이지의 끝까지 선택 영역에 포함합니다.	Shift-Page Down	selectpagedown
커서에서 현재 페이지의 처음까지 선택 영역에 포함합니다.	Shift-Page Up	selectpageup
선택 영역에 커서 오른쪽의 다음 공백를 포함합니다.	Shift-Right	selectright

설명	키 바인딩	명령
커서에서 현재 파일의 끝까지 선택 영역에 포함합니다.	`Ctrl-Shift-End`	`selecttoend`
커서에서 현재 줄의 끝까지 선택 영역에 포함합니다.	`Alt-Shift-Right`	`selecttolineend`
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	`Alt-Shift-Left`	`selecttolinestart`
커서에서 현재 범위의 일치하는 다음 기호까지 포함합니다.	`Ctrl-Shift-P`	`selecttomatching`
커서에서 현재 파일의 처음까지 선택 영역에 포함합니다.	`Ctrl-Shift-Home`	`selecttostart`
선택 영역에 다음 줄 위의 내용을 포함합니다.	`Shift-Up`	`selectup`
선택 영역에 커서 왼쪽의 다음 단어를 포함합니다.	`Ctrl-Shift-Left`	`selectwordleft`
선택 영역에 커서 오른쪽의 다음 단어를 포함합니다.	`Ctrl-Shift-Right`	`selectwordright`
[기본 설정(Preferences)] 탭을 표시합니다.	`Ctrl-,`	`showSettingsMenu`
이전 선택 영역을 모두 지웁니다.	`Esc`	`singleSelection`
글꼴 크기를 줄입니다.	`Ctrl--`	`smallerfont`
여러 줄을 선택한 경우 정렬된 순서로 다시 정렬합니다.	`Ctrl-Alt-S`	`sortlines`
현재 줄의 끝에 커서를 추가합니다.	`Ctrl-Alt-L`	`splitIntoLines`

설명	키 바인딩	명령
커서의 내용을 줄의 끝에 새 줄로 이동합니다.	Ctrl-O	splitline
블록 주석 문자로 선택 영역을 둘러싸거나 해당 문자가 있는 경우 제거합니다.	Ctrl-Shift-/	toggleBlockComment
선택한 각 줄의 처음에 줄 설명 문자를 추가하거나, 줄 설명 문자가 있는 경우 제거합니다.	Ctrl-/	togglecomment
코드를 접거나, 코드 접기를 제거합니다(있는 경우).	F2	toggleFoldWidget
상위 코드를 접거나, 접기를 제거합니다(있는 경우).	Alt-F2	toggleParentFoldWidget
키 입력 기록을 시작하거나 이미 기록 중인 경우 중지합니다.	Ctrl-Alt-E	togglerecording
단어를 줄 바꿈하거나 이미 줄 바꿈이 있는 경우 단어 줄 바꿈을 중지합니다.	Ctrl-Q	toggleWordWrap
선택 영역을 모두 소문자로 변경합니다.	Ctrl-Shift-U	tolowercase
선택 영역을 모두 대문자로 변경합니다.	Ctrl-U	touppercase
선택 영역의 대소문자를 바꿉니다.	Alt-X	transposeletters
선택한 코드를 펼칩니다.	Alt-Shift-L | Ctrl-Shift-F1	unfold

설명	키 바인딩	명령
전체 파일의 코드 접기를 펼칩니다.	Alt-Shift-0	unfoldall

emmet

설명	키 바인딩	명령
간단한 수학 표현식(예: 2*4 또는 10/2)과 그 결과를 출력합니다.	Shift-Ctrl-Y	emmet_evaluate_math_expression
현재 파일의 구문에 따라 CSS와 유사한 약어를 HTML, XML 또는 CSS 코드로 확장합니다.	Ctrl-Alt-E	emmet_expand_abbreviation
탭 정지에 의해 확장된 CSS와 유사한 약어를 순회합니다.	Tab	emmet_expand_abbreviation_with_tab
편집 가능한 다음 코드 부분으로 이동합니다.	Shift-Ctrl-.	emmet_select_next_item
이전 편집 가능한 코드 부분으로 이동합니다.	Shift-Ctrl-,	emmet_select_previous_item
약어를 확장한 다음 생성된 코드 조각의 마지막 요소 내에 현재 선택 영역을 배치합니다.	Shift-Ctrl-A	emmet_wrap_with_abbreviation

터미널

설명	키 바인딩	명령
새 [터미널(Terminal)] 탭을 엽니다.	Alt-T	openterminal

설명	키 바인딩	명령
편집기와 [터미널(Terminal)] 탭 간을 전환합니다.	Alt-S	switchterminal

실행 및 디버깅

설명	키 바인딩	명령
현재 파일을 빌드합니다.	Ctrl-B	build
현재 일시 중지된 프로세스를 다시 시작합니다.	F8	resume
현재 애플리케이션을 실행하거나 디버그합니다.	Alt-F5	run
마지막 실행 파일을 실행하거나 디버그합니다.	F5	runlast
스택 옆에 있는 함수로 진행합니다.	F11	stepinto
현재 함수 범위를 종료합니다.	Shift-F11	stepout
스택의 현재 표현식으로 진행합니다.	F10	stepover
현재 애플리케이션의 실행 또는 디버깅을 중지합니다.	Shift-F5	stop
현재 파일의 빌드를 중지합니다.	Ctrl-Shift-C	stopbuild

AWS Cloud9 통합 개발 환경(IDE)에 대한 Windows/Linux Sublime 키 바인딩 참조

다음은 AWS Cloud9 IDE의 Windows/Linux 운영 체제용 Sublime 키보드 모드 키 바인딩 목록입니다.

자세한 내용은 AWS Cloud9 IDE에서 다음을 참조하세요.

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. [기본 설정(Preferences)] 탭에서 [키 바인딩(Keybindings)]을 선택합니다.
3. 용키보드 모드를 선택하고숭고한.
4. [운영 체제(Operating System)]에서 [Windows / Linux]를 선택합니다.

키 바인딩 작업도 참조하세요.

일반

설명	키 바인딩	명령
선택 영역을 시계 표현식으로 추가합니다.	`Ctrl-Shift-C`	`addwatchfromselection`
클립보드에서 잘라낸 선택 영역을 제거합니다.	`Esc`	`clearcut`
코드 완성 컨텍스트 메뉴를 표시합니다.	`Ctrl-Space`	`complete`

설명	키 바인딩	명령
코드를 완료한 다음 덮어씁니다.	Ctrl-Shift-Space | Alt-Shift-Space	completeoverwrite
선택 영역을 클립보드로 복사합니다.	Ctrl-C	copy
선택 영역을 클립보드로 잘라냅니다.	Ctrl-X	cut
커서에서 줄의 시작까지 삭제합니다.	Ctrl-Shift-Backspace | Ctrl-K Ctrl-Backspace	delete_to_hard_bol
커서에서 줄의 끝까지 삭제합니다.	Ctrl-Shift-Delete | Ctrl-K Ctrl-K	delete_to_hard_eol
코드를 확장합니다(해당하는 경우).	Tab	expandSnippet
현재 문서의 찾기 및 바꾸기 막대를 표시합니다.	Ctrl-F	find
선택 항목에 대한 모든 일치 항목 강조 표시	Alt-F3	find_all_under
선택 항목에 대해 다음 일치 항목 강조 표시	Ctrl-F3	find_under
커서 주위에 강조 표시하고 강조 표시와 일치하는 모든 항목에 강조 표시합니다.	Ctrl-D	find_under_expand
커서 주위에 강조 표시하고 강조 표시와 일치하는 모든 항목에 윤곽선을 표시합니다.	Ctrl-K Ctrl-D	find_under_expand_skip

설명	키 바인딩	명령
선택 영역에 대해 이전 일치 항목에 강조 표시합니다.	`Ctrl-Shift-F3`	`find_under_prev`
현재 문서에서 일치하는 항목을 모두 선택합니다.	`Ctrl-Alt-K`	`findAll`
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 다음 항목으로 이동합니다.	`F3`	`findnext`
마지막으로 입력한 찾기 쿼리에 대해 현재 문서에서 일치하는 이전 항목으로 이동합니다.	`Shift-F3`	`findprevious`
편집기에서 활성 파일의 삽입 점에 기호에 대한 알려진 모든 참조를 표시합니다.	`Shift-F3`	`findReferences`
[환경(Environment) 창을 연 다음 파일 목록을 활성 상태로 만듭니다.	`Shift-Esc`	`focusTree`
선택한 JavaScript 코드를 다시 포맷합니다.	`Ctrl-Alt-F`	`formatcode`
줄로 이동 상자를 표시합니다.	`Ctrl-G`	`gotoline`
찾기 및 바꾸기 막대가 표시된 경우 숨깁니다.	`Esc`	`hidesearchreplace`
커서 위치에 있는 변수 또는 함수의 정의로 이동합니다.	`F12`	`jumptodef`

설명	키 바인딩	명령
AWS 리소스 창의 Lambda 섹션에서 로컬 Lambda 함수를 선택한 경우 해당 함수를 원격 함수로 Lambda에 업로드하려고 시도합니다.	`Ctrl-Shift-U`	`lambdaUploadFunction`
현재 단어의 끝으로 이동합니다.	`Ctrl-Right`	`moveToWordEndRight`
현재 단어의 처음으로 이동합니다.	`Ctrl-Left`	`moveToWordStartLeft`
새 파일을 만듭니다.	`Alt-N`	`newfile`
[기본 설정(Preferences)] 탭을 표시합니다.	`Ctrl-,`	`openpreferences`
[터미널(Terminal)] 탭을 연 다음 파일 목록에서 선택한 파일의 상위 폴더로 전환합니다.	`Alt-L`	`opentermhere`
클립보드의 현재 내용을 커서 위치에 붙여 넣습니다.	`Ctrl-V`	`paste`
오류 수정에 대한 제안 사항을 표시합니다.	`Ctrl-F3`	`quickfix`
마지막 작업을 다시 실행합니다.	`Ctrl-Shift-Z | Ctrl-Y`	`redo`
미리 보기 창을 새로 고칩니다.	`Ctrl-Enter`	`reloadpreview`
선택 항목의 이름 바꾸기 리팩터링을 시작합니다.	`Ctrl-Alt-R`	`renameVar`

설명	키 바인딩	명령
현재 문서의 찾기 및 바꾸기 모음을 표시하고 [바꾸기(Replace With)] 표현식을 사용합니다.	Ctrl-H	replace
찾기 및 바꾸기 막대에서 모든 찾기 표현식 일치 항목을 바꾸기 표현식으로 대체합니다.	Ctrl-Alt-Enter	replaceall
찾기 및 바꾸기 막대에서 다음 찾기 표현식 일치 항목을 바꾸기 표현식으로 대체합니다.	Ctrl-Shift-H	replacenext
초기화 스크립트를 다시 실행합니다.	Ctrl-Enter	rerunInitScript
환경을 다시 시작합니다.	Ctrl-R	restartc9
현재 파일을 마지막으로 저장한 버전으로 재설정합니다.	Ctrl-Shift-Q	reverttosaved
열려 있는 각 파일을 저장된 버전으로 재설정합니다.	Alt-Shift-Q	reverttosavedall
현재 파일을 디스크에 저장합니다.	Ctrl-S	save
현재 파일을 다른 파일 이름을 사용하여 디스크에 저장합니다.	Ctrl-Shift-S	saveas
여러 파일의 찾기 및 바꾸기 모음을 표시합니다.	Ctrl-Shift-F	searchinfiles
커서에서 단어의 끝까지 선택 영역에 포함합니다.	Ctrl-Shift-Right	selectToWordEndRight

설명	키 바인딩	명령
커서에서 단어의 처음까지 선택 영역에 포함합니다.	`Ctrl-Shift-Left`	`selectToWordStartLeft`
[프로세스 목록(Process List)] 대화 상자를 표시합니다.	`Ctrl-Alt-P`	`showprocesslist`
마지막 작업을 실행 취소합니다.	`Ctrl-Z`	`undo`

탭

설명	키 바인딩	명령
현재 창에서 현재 탭을 제외하고 열려 있는 탭을 모두 닫습니다.	`Ctrl-Alt-W`	`closeallbutme`
모든 창에서 열려 있는 탭을 모두 닫습니다.	`Alt-Shift-W`	`closealltabs`
현재 창을 닫습니다.	`Ctrl-W`	`closepane`
현재 탭을 닫습니다.	`Alt-W`	`closetab`
아래로 창을 하나 이동합니다.	`Ctrl-Meta-Down`	`gotopanedown`
왼쪽으로 창을 하나 이동합니다.	`Ctrl-Meta-Left`	`gotopaneleft`
오른쪽으로 창을 하나 이동합니다.	`Ctrl-Meta-Right`	`gotopaneright`
위로 창을 하나 이동합니다.	`Ctrl-Meta-Up`	`gottopaneup`
왼쪽으로 한 탭 이동합니다.	`Ctrl-Page Up`	`gototableft`
오른쪽으로 한 탭 이동합니다.	`Ctrl-Page Down`	`gototabright`

설명	키 바인딩	명령
현재 탭을 한 창 아래로 이동하거나 탭이 이미 맨 아래에 있는 경우 분할 탭을 만듭니다.	Ctrl-Meta-Down	movetabdown
현재 탭을 왼쪽으로 이동하거나 탭이 이미 맨 왼쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Ctrl-Meta-Left	movetableft
현재 탭을 오른쪽으로 이동하거나 탭이 이미 맨 오른쪽에 있는 경우 해당 위치에 분할 탭을 만듭니다.	Ctrl-Meta-Right	movetabright
현재 탭을 한 창 위로 이동하거나 탭이 이미 맨 위에 있는 경우 분할 탭을 만듭니다.	Ctrl-Meta-Up	movetabup
다음 탭으로 이동합니다.	Ctrl-Tab	nexttab
이전 창으로 이동합니다.	Ctrl-Shift-`	previouspane
이전 탭으로 이동합니다.	Ctrl-Shift-Tab	previoustab
마지막 탭으로 돌아갑니다.	Esc	refocusTab
마지막 탭을 다시 엽니다.	Ctrl-Shift-T	reopenLastTab
파일 트리에 현재 탭을 표시합니다.	Ctrl-E	revealtab
10번째 탭으로 이동합니다.	Ctrl-0	tab0
첫 번째 탭으로 이동합니다.	Ctrl-1	tab1
두 번째 탭으로 이동합니다.	Ctrl-2	tab2
세 번째 탭으로 이동합니다.	Ctrl-3	tab3

설명	키 바인딩	명령
네 번째 탭으로 이동합니다.	`Ctrl-4`	`tab4`
다섯 번째 탭으로 이동합니다.	`Ctrl-5`	`tab5`
여섯 번째 탭으로 이동합니다.	`Ctrl-6`	`tab6`
일곱 번째 탭으로 이동합니다.	`Ctrl-7`	`tab7`
여덟 번째 탭으로 이동합니다.	`Ctrl-8`	`tab8`
아홉 번째 탭으로 이동합니다.	`Ctrl-9`	`tab9`

패널

설명	키 바인딩	명령
[바로 가기(Go to Anything)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-E|Ctrl-P`	`gotoanything`
[명령으로 이동(Go to Command)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-.|F1`	`gotocommand`
[파일로 이동(Go to File)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-O`	`gotofile`
[기호로 이동(Go to Symbol)] 모드로 [이동(Go)] 창을 표시합니다.	`Ctrl-Shift-O`	`gotosymbol`
[개요(Outline)] 창을 표시합니다.	`Ctrl-R|Ctrl-Shift-R`	`outline`

설명	키 바인딩	명령
숨겨져 있는 경우 [콘솔(Console)] 창을 표시하거나 표시된 경우 숨깁니다.	`Ctrl-``	`toggleconsole`
숨겨져 있는 경우 [환경(Environment)] 창을 표시하거나 표시된 경우 숨깁니다.	`Ctrl-K Ctrl-B`	`toggletree`

코드 편집기

설명	키 바인딩	명령
활성 커서 한 줄 위에 커서를 추가하거나 커서가 이미 추가된 경우 해당 커서 위에 다른 커서를 추가합니다.	`Ctrl-Alt-Up`	`addCursorAbove`
활성 커서 한 줄 위에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 위로 이동합니다.	`Ctrl-Alt-Shift-Up`	`addCursorAboveSkipCurrent`
활성 커서 한 줄 아래에 커서를 추가하거나 커서가 이미 추가된 경우 해당 커서 아래에 다른 커서를 추가합니다.	`Ctrl-Alt-Down`	`addCursorBelow`
활성 커서 한 줄 아래에 두 번째 커서를 추가하거나 두 번째 커서가 이미 추가된 경우 두 번째 커서를 한 줄 아래로 이동합니다.	`Ctrl-Alt-Shift-Down`	`addCursorBelowSkipCurrent`

설명	키 바인딩	명령
정렬이 잘못된 경우 모든 커서를 각 행의 활성 커서와 동일한 공간으로 이동합니다.	Ctrl-Alt-A	alignCursors
단일 공백을 백스페이스합니다.	Shift-Backspace | Backspace	backspace
선택 영역을 한 탭 들여씁니다.	Ctrl-]	blockindent
선택 영역을 한 탭 내어씁니다.	Ctrl-[	blockoutdent
포커스를 편집기에서 IDE의 다른 곳으로 전환할 수 있는지 여부를 제어합니다.	Ctrl-Z | Ctrl-Shift-Z | Ctrl-Y	cancelBrowserUndoInAce
선택 영역을 가운데로 정렬합니다.	Ctrl-K Ctrl-C	centerselection
줄의 내용을 복사하고 복사한 내용을 한 줄 아래에 붙여 넣습니다.	Alt-Shift-Down	copylinesdown
줄의 내용을 복사하고 복사한 내용을 한 줄 위에 붙여 넣습니다.	Alt-Shift-Up	copylinesup
선택 영역을 잘라내거나 선택 영역이 없으면 공백 하나 삭제	Shift-Delete	cut_or_delete
공백을 하나 삭제합니다.	Delete	del
선택 영역의 내용을 복사하고 복사한 내용을 선택 영역 바로 뒤에 붙여넣습니다.	Ctrl-Shift-D	duplicateSelection
선택 영역에 현재 행의 내용을 포함합니다.	Ctrl-Shift-L	expandtoline

설명	키 바인딩	명령
선택 영역에 일치하는 다음 기호까지 포함합니다.	Ctrl-Shift-M	expandToMatching
선택한 코드를 접습니다. 접힌 단위가 선택된 경우 펼칩니다.	Alt-L | Ctrl-F1	fold
현재 선택 범위를 제외하고 접을 수 있는 모든 요소를 접습니다.	Ctrl-K Ctrl-1	foldOther
한 줄 아래로 이동합니다.	Down	golinedown
한 줄 위로 이동합니다.	Up	golineup
로그 파일의 끝 부분으로 이동합니다.	Ctrl-End	gotoend
한 칸씩 왼쪽으로 이동합니다.	Left	gotoleft
현재 줄의 끝으로 이동합니다.	Alt-Right | End	gotolineend
현재 줄의 처음으로 이동합니다.	Alt-Left | Home	gotolinestart
다음 오류로 이동합니다.	Ctrl-F6	goToNextError
한 페이지 아래로 이동합니다.	Page Down	gotopagedown
한 페이지 위로 이동합니다.	Page Up	gotopageup
이전 오류로 이동합니다.	Ctrl-Shift-F6	goToPreviousError
오른쪽으로 공백 하나 이동합니다.	Right	gotoright
로그 파일의 시작 부분으로 이동합니다.	Ctrl-Home	gotostart
왼쪽으로 한 단어 이동합니다.	Ctrl-Left	gotowordleft

설명	키 바인딩	명령
오른쪽으로 한 단어 이동합니다.	Ctrl-Right	gotowordright
선택 영역을 한 탭 들여씁니다.	Tab	indent
커서에서 단어의 처음까지 선택 영역에 포함합니다.	Ctrl-J	joinlines
현재 범위에서 쌍을 이루는 기호로 이동합니다.	Ctrl-M	jumptomatching
글꼴 크기를 늘립니다.	Ctrl-- | Ctrl-= | Ctrl-+	largerfont
커서의 왼쪽에 있는 숫자를 1씩 줄입니다(숫자인 경우).	Alt-Down	modifyNumberDown
커서의 왼쪽에 있는 숫자를 1씩 늘립니다(숫자인 경우).	Alt-Up	modifyNumberUp
선택 영역을 한 줄 아래로 이동합니다.	Ctrl-Shift-Down	movelinesdown
선택 영역을 한 줄 위로 이동합니다.	Ctrl-Shift-Up	movelinesup
선택 영역을 한 탭 내어씁니다.	Shift-Tab	outdent
덮어쓰기 모드를 켜거나 켜진 경우 끕니다.	Insert	overwrite
현재 줄의 내용을 삭제합니다.	Ctrl-Shift-K	removeline
커서에서 현재 줄의 끝까지 삭제합니다.	Alt-Delete	removetolineend
커서에서 현재 줄의 시작까지 삭제합니다.	Alt-Backspace	removetolinestart

설명	키 바인딩	명령
커서의 왼쪽에 있는 단어를 삭제합니다.	Ctrl-Backspace	removewordleft
커서의 오른쪽에 있는 단어를 삭제합니다.	Ctrl-Delete	removewordright
이전에 기록한 키 입력을 재생합니다.	Ctrl-Shift-Q	replaymacro
현재 파일을 한 줄씩 아래로 스크롤합니다.	Ctrl-Down	scrolldown
현재 파일을 한 줄씩 위로 스크롤합니다.	Ctrl-Up	scrollup
선택 가능한 내용을 모두 선택합니다.	Ctrl-A	selectall
선택 영역에 다음 줄 아래의 내용을 포함합니다.	Shift-Down	selectdown
선택 영역에 왼쪽의 다음 공백을 포함합니다.	Shift-Left	selectleft
커서부터 현재 줄의 나머지 부분을 선택 영역에 포함합니다.	Shift-End	selectlineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Shift-Home	selectlinestart
선택 영역의 뒤에 일치하는 선택 영역을 더 포함합니다.	Ctrl-Alt-Right	selectMoreAfter
선택 영역의 앞에 일치하는 선택 영역을 더 포함합니다.	Ctrl-Alt-Left	selectMoreBefore

설명	키 바인딩	명령
선택 영역의 뒤에 있는 일치하는 다음 선택 영역을 포함합니다.	Ctrl-Alt-Shift-Right	selectNextAfter
선택 영역의 앞에 있는 일치하는 다음 선택 영역을 포함합니다.	Ctrl-Alt-Shift-Left	selectNextBefore
일치하는 다음 선택 항목을 선택하거나 찾습니다.	Alt-K	selectOrFindNext
일치하는 이전 선택 항목 선택하거나 찾습니다.	Alt-Shift-K	selectOrFindPrevious
커서에서 현재 페이지의 끝까지 선택 영역에 포함합니다.	Shift-Page Down	selectpagedown
커서에서 현재 페이지의 처음까지 선택 영역에 포함합니다.	Shift-Page Up	selectpageup
선택 영역에 커서 오른쪽의 다음 공백를 포함합니다.	Shift-Right	selectright
커서에서 현재 파일의 끝까지 선택 영역에 포함합니다.	Ctrl-Shift-End	selecttoend
커서에서 현재 줄의 끝까지 선택 영역에 포함합니다.	Alt-Shift-Right	selecttolineend
현재 줄의 처음부터 커서까지 선택 영역에 포함합니다.	Alt-Shift-Left	selecttolinestart
커서에서 현재 범위의 일치하는 다음 기호까지 포함합니다.	Ctrl-Shift-P	selecttomatching
커서에서 현재 파일의 처음까지 선택 영역에 포함합니다.	Ctrl-Shift-Home	selecttostart

설명	키 바인딩	명령
선택 영역에 다음 줄 위의 내용을 포함합니다.	Shift-Up	selectup
선택 영역에 커서 왼쪽의 다음 단어를 포함합니다.	Ctrl-Shift-Left	selectwordleft
선택 영역에 커서 오른쪽의 다음 단어를 포함합니다.	Ctrl-Shift-Right	selectwordright
[기본 설정(Preferences)] 탭을 표시합니다.	Ctrl-,	showSettingsMenu
이전 선택 영역을 모두 지웁니다.	Esc	singleSelection
글꼴 크기를 줄입니다.	Ctrl-- | Ctrl-Shift-= | Ctrl-Shift-+	smallerfont
여러 줄을 선택한 경우 정렬된 순서로 다시 정렬합니다.	F9	sortlines
현재 줄의 끝에 커서를 추가합니다.	Ctrl-Shift-L	splitIntoLines
블록 주석 문자로 선택 영역을 둘러싸거나 해당 문자가 있는 경우 제거합니다.	Ctrl-Shift-/	toggleBlockComment
선택한 각 줄의 처음에 줄 설명 문자를 추가하거나, 줄 설명 문자가 있는 경우 제거합니다.	Ctrl-/	togglecomment
코드를 접거나, 코드 접기를 제거합니다(있는 경우).	Ctrl-Shift-[	toggleFoldWidget
상위 코드를 접거나, 접기를 제거합니다(있는 경우).	Alt-F2	toggleParentFoldWidget

설명	키 바인딩	명령
키 입력 기록을 시작하거나 이미 기록 중인 경우 중지합니다.	Ctrl-Q	togglerecording
단어를 줄 바꿈하거나 이미 줄 바꿈이 있는 경우 단어 줄 바꿈을 중지합니다.	Ctrl-Q	toggleWordWrap
선택 영역을 모두 소문자로 변경합니다.	Ctrl-K Ctrl-L	tolowercase
선택 영역을 모두 대문자로 변경합니다.	Ctrl-K Ctrl-U	touppercase
선택 영역의 대소문자를 바꿉니다.	Alt-X	transposeletters
선택한 코드를 펼칩니다.	Ctrl-Shift-]	unfold
전체 파일의 코드 접기를 펼칩니다.	Ctrl-K Ctrl-0 | Ctrl-K Ctrl-J	unfoldall

emmet

설명	키 바인딩	명령
간단한 수학 표현식(예: 2*4 또는 10/2)과 그 결과를 출력합니다.	Shift-Ctrl-Y	emmet_evaluate_math_expression
현재 파일의 구문에 따라 CSS와 유사한 약어를 HTML, XML 또는 CSS 코드로 확장합니다.	Ctrl-Alt-E	emmet_expand_abbreviation
탭 정지에 의해 확장된 CSS와 유사한 약어를 순회합니다.	Tab	emmet_expand_abbreviation_with_tab

설명	키 바인딩	명령
편집 가능한 다음 코드 부분으로 이동합니다.	`Shift-Ctrl-.`	`emmet_select_next_item`
이전 편집 가능한 코드 부분으로 이동합니다.	`Shift-Ctrl-,`	`emmet_select_previous_item`
약어를 확장한 다음 생성된 코드 조각의 마지막 요소 내에 현재 선택 영역을 배치합니다.	`Shift-Ctrl-A`	`emmet_wrap_with_abbreviation`

터미널

설명	키 바인딩	명령
새 [터미널(Terminal)] 탭을 엽니다.	`Alt-T`	`openterminal`
편집기와 [터미널(Terminal)] 탭 간을 전환합니다.	`Alt-S`	`switchterminal`

실행 및 디버깅

설명	키 바인딩	명령
현재 파일을 빌드합니다.	`F7 | Ctrl-B`	`build`
현재 일시 중지된 프로세스를 다시 시작합니다.	`F8`	`resume`
현재 애플리케이션을 실행하거나 디버그합니다.	`Ctrl-Shift-B`	`run`
마지막 실행 파일을 실행하거나 디버그합니다.	`F5`	`runlast`

설명	키 바인딩	명령
스택 옆에 있는 함수로 진행합니다.	F11	stepinto
현재 함수 범위를 종료합니다.	Shift-F11	stepout
스택의 현재 표현식으로 진행합니다.	F10	stepover
현재 애플리케이션의 실행 또는 디버깅을 중지합니다.	Shift-F5	stop
현재 파일의 빌드를 중지합니다.	Ctrl-Break	stopbuild

AWS Cloud9 통합 개발 환경(IDE)의 명령 참조

AWS Cloud9 IDE에서 명령을 실행하려면:

1. [이동(Go)] 버튼(돋보기)을 선택하여 [이동(Go)] 창을 표시합니다. [이동(Go)] 버튼이 표시되지 않으면 메뉴 모음에서 [창, 이동(Window, Go)]을 선택합니다.
2. 바로 가기(Go to Anything) 상자에서 명령 그룹(예: Code Editor)의 이름을 입력하기 시작합니다. 그룹에는 공통 테마 또는 IDE 기능을 중심으로 구성된 여러 명령이 포함되어 있습니다.
3. [명령(Commands)] 머리글 아래의 그룹에서 실행할 특정 명령을 선택합니다.

사용 가능한 명령 그룹

명령 그룹	설명
AWS	AWS 도구 키트에 대한 명령
Clipboard	콘텐츠 복사 및 붙여넣기에 대한 명령
Code Editor	코드 편집기 인터페이스를 탐색하고 편집기의 콘텐츠와 상호 작용하는 명령

명령 그룹	설명
Emmet	HTML 및 CSS 콘텐츠에 사용되는 Emmet 도구 키트로 작업하기 위한 명령
General	IDE의 구성 및 프로젝트 파일을 관리하기 위한 기타 명령
Panels	IDE 인터페이스에서 패널 표시를 관리하기 위한 명령
Run & Debug	AWS Cloud9에서 프로젝트를 실행 및 디버그하기 위한 명령
Tabs	IDE 인터페이스에서 탭 표시 및 탐색을 관리하기 위한 명령
Terminal	명령줄 터미널을 관리하기 위한 명령
Window	IDE 창에서 창 레이아웃을 관리하기 위한 명령

다른 AWS 서비스와 함께 사용

AWS Cloud9을 사용할 때 Amazon Lightsail, AWS CodeStar 및 AWS CodePipeline을 적극 활용하여 작업할 수 있습니다. 이를 수행하는 방법이 이 섹션의 주제입니다.

> **Important**
>
> AWS 도구 키트 기능 AWS Lambda, AWS Serverless Application Model 및 Amazon S3와 같은 주요 AWS 서비스를 사용한 작업을 위한 편리한 시각적 인터페이스를 제공합니다. 자세한 내용은 AWS 도구 키트 섹션을 참조하세요.

주제

AWS Cloud9 통합 개발 환경(IDE)의 Amazon Lightsail 인스턴스스

AWS Cloud9 IDE를 사용하여 널리 사용되는 애플리케이션 및 프레임워크로 사전 구성된 Amazon Lightsail 인스턴스의 코드로 작업할 수 있습니다. 여기에는 WordPress, LAMP(Linux, Apache, MySQL 및 PHP), Node.js, NGINX, Drupal 및 Joomla가 포함됩니다. Amazon Linux, Ubuntu, Debian, FreeBSD 및 openSUSE 등의 Linux 배포판이 포함됩니다.

Lightsail은 편리하고 빠른 설치 가상 사설 서버 솔루션을 제공합니다. Lightsail은 클라우드에서 웹 사이트 및 웹 애플리케이션을 배포하고 관리하기 위한 컴퓨팅, 스토리지, 네트워킹 용량 및 기능을 제공합니다. Lightsail을 사용하면 저렴하고 예측 가능한 월별 요금으로 프로젝트를 빠르게 시작할 수 있습니다. 자세한 내용은 Amazon Lightsail 기능을 참조하세요.

이 주제에서 AWS Cloud9과 호환되는 Linux 기반 Lightsail 인스턴스를 생성하고 설정할 수 있습니다. 그런 다음 AWS Cloud9 SSH 개발 환경을 생성하여 Lightsail 인스턴스에 연결합니다.

> **Note**
>
> 다음 절차를 완료하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Lightsail과 같은 서비스에 대해 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon Lightsail 요금을 참조하세요.
>
> AWS Cloud9 IDE, 소스 제어, 빌드, 배포, 가상 서버 또는 서버리스 리소스 등으로 구성된 도구 체인을 포함하는 고급 솔루션을 생성하고 설정하려면 AWS CodeStar 프로젝트 작업을 참조하세요.
>
> AWS Cloud9 IDE를 사용하여 샘플 코드를 포함하지 않는 Amazon Linux 또는 Ubuntu Server를 실행하는 Amazon EC2 인스턴스로 작업하려면 시작하기: 기본 자습서을 참조하세요.

1단계: Linux 기반 Lightsail 인스턴스 생성

이 단계에서는 Lightsail 콘솔을 사용하여 Linux 기반 배포판에서 앱을 실행하는 Amazon EC2 인스턴스를 생성합니다. 이 인스턴스에는 다음 사항이 자동으로 포함됩니다.

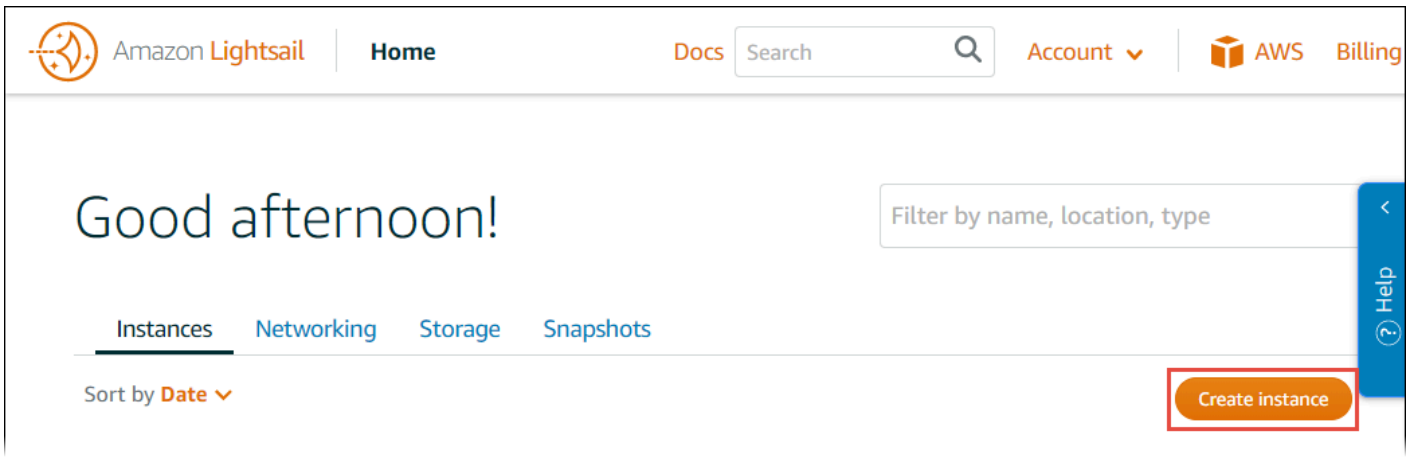
- 퍼블릭 및 프라이빗 IP 주소. (나중에 정적 퍼블릭 IP를 생성할 수 있습니다.)
- SSH(포트 22), HTTP(포트 80) 및 HTTPS(포트 443)를 사용하여 인스턴스에 액세스. (이 설정은 변경할 수 있습니다.)
- 블록 스토리지 디스크 (나중에 추가 디스크를 연결할 수 있습니다.)
- 기본 제공 시스템 보고 기능.

Lightsail 콘솔에서는 인스턴스를 나중에 백업, 재부팅, 중지 또는 삭제할 수 있습니다.

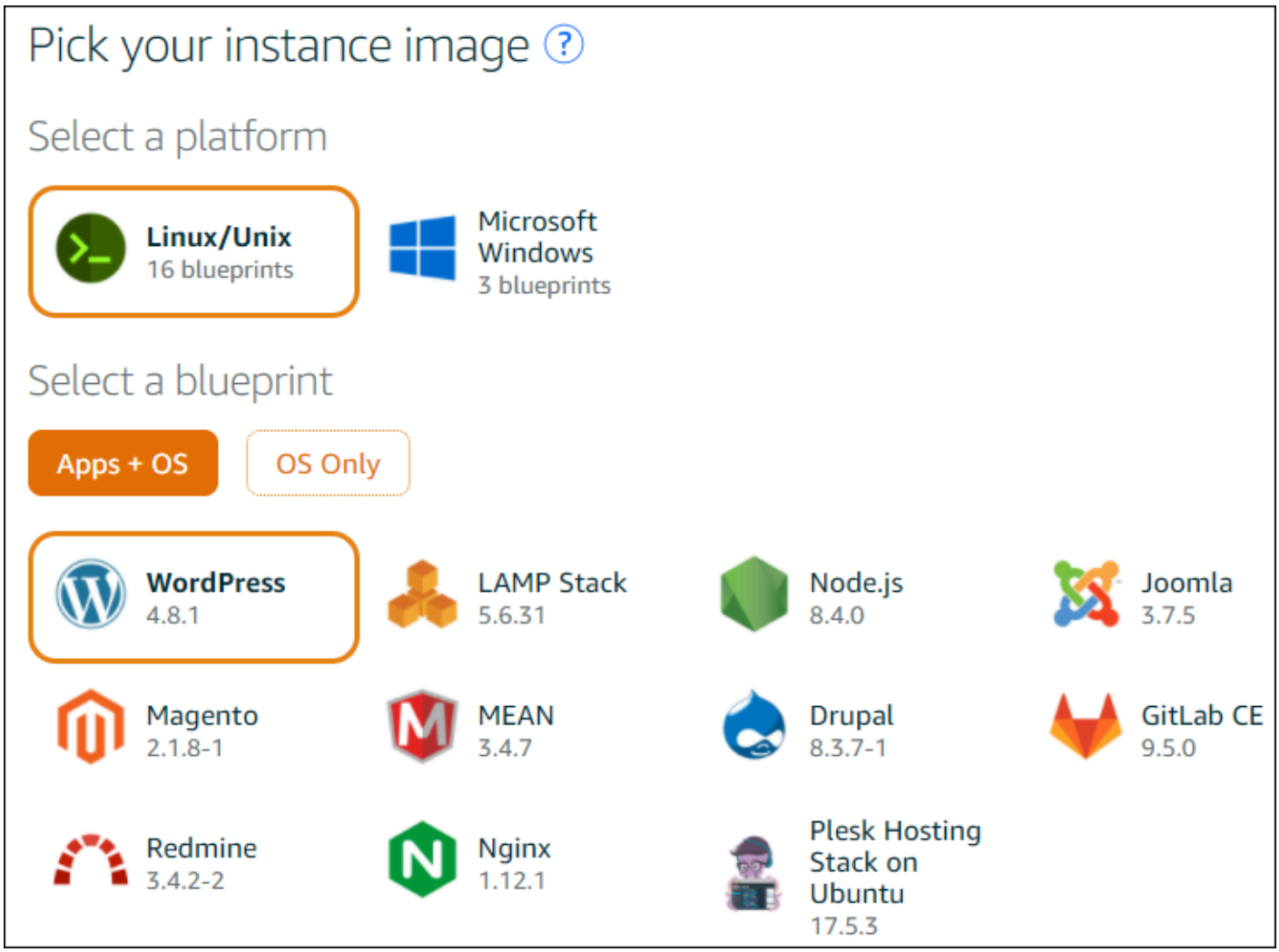
1. Lightsail 콘솔(https://lightsail.aws.amazon.com)을 열어서 로그인합니다.

   AWS 계정의 IAM 관리자 사용자용 보안 인증 정보를 사용하여 로그인하는 것이 좋습니다. IAM 관리자 사용자로 로그인할 수 없으면 AWS 계정 관리자에게 문의하세요.
2. 해당 메시지가 표시되면 콘솔에서 사용할 언어를 선택하고 나서 저장을 선택합니다.

3. 해당 메시지가 표시되면 시작하기를 선택합니다.
4. 홈 페이지에 이미 선택되어 있는 인스턴스 탭에서 인스턴스 생성을 선택합니다.



5. 인스턴스 위치에서 인스턴스를 만들려는 위치는 AWS Cloud9을 사용할 수 있고 AWS 리전에 해당되는지 확인합니다. 자세한 내용은 AWS Cloud9의 Amazon Web Services 일반 참조 섹션을 참조하세요. AWS 리전, 가용 영역 또는 두 가지를 모두 변경하려면 AWS 리전 및 가용 영역 변경을 선택한 후 화면의 지침을 따릅니다.
6. [인스턴스 이미지 선택(Pick your instance image)]에서 [플랫폼 선택(Select a platform)]에 [Linux/Unix]가 이미 선택되어 있고 [블루프린트 선택(Select a blueprint)]에 [Apps + OS]가 이미 선택되어 있는 상태로 블루프린트를 선택합니다.



> **Note**
>
> 앱이 없는 인스턴스를 생성하려면 [Apps + OS] 대신 [OS만(OS Only)]을 선택한 다음 배포판을 선택합니다.
> 사용 가능한 선택 사항에 대한 자세한 내용은 Lightsail 웹사이트에서 Amazon Lightsail 인스턴스 이미지 선택을 참조하세요.

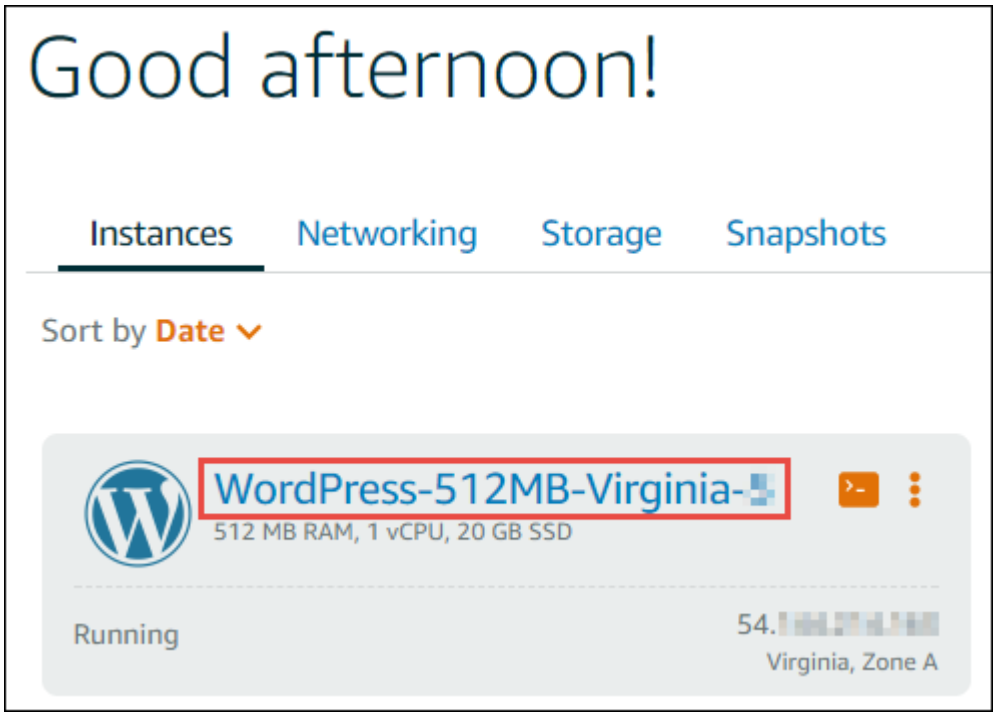
7. 인스턴스 계획 선택에서 계획을 선택하거나 선택된 기본 계획을 그대로 사용합니다.
8. 인스턴스 이름 지정에서 인스턴스의 이름을 입력하거나 제안된 기본 이름을 그대로 사용합니다.
9. 인스턴스 수에는 생성할 인스턴스 수를 입력하거나 기본값인 단일 인스턴스(x 1)를 그대로 유지합니다.
10. 생성을 선택합니다.

2단계: AWS Cloud9에서 사용하도록 인스턴스 설정

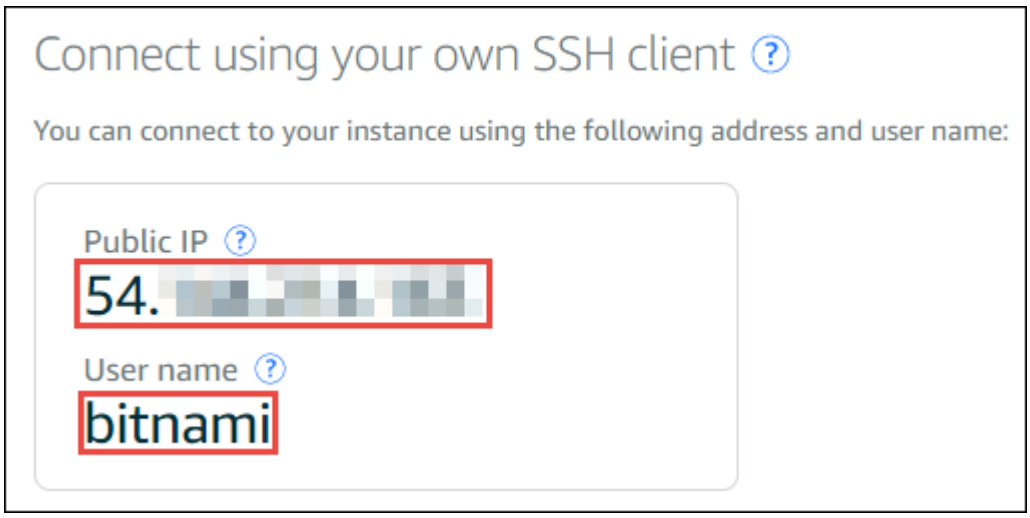
이 단계에서 실행 중인 인스턴스에 연결한 다음 AWS Cloud9에서 나중에 사용할 수 있도록 설정합니다.

> **Note**
>
> 다음 지침에서는 이전 단계에서 앱 + OS를 선택했다고 가정합니다. OS 전용과 배포판(Ubuntu 이외)을 선택한 경우 다음 지침에 따라 조정해야 할 수도 있습니다.

1. 이전 단계의 Lightsail 콘솔이 열려 있는 상태로 인스턴스 탭의 인스턴스 카드에서 인스턴스의 이름을 선택합니다.



2. Connect(연결) 탭의 Connect using your own SSH client(자체 SSH 클라이언트를 사용하여 연결)에 표시된 퍼블릭 IP와 사용자 이름 값을 적어둡니다. 나중에 이들 값이 필요합니다.



3. SSH를 사용하여 연결을 선택합니다.
4. 인스턴스에 최신 시스템 업데이트가 있는지 확인합니다. 이렇게 하려면 표시되는 터미널 세션에서 **`sudo apt update`** 명령을 실행합니다.
5. Python이 설치되었는지 확인하고 설치된 경우 버전이 2.7인지 확인합니다. 버전을 확인하려면 **`python --version`** 명령을 실행하고 표시되는 버전 번호를 적어둡니다. 버전 번호가 표시되지 않거나 버전이 2.7이 아니면 **`sudo apt install -y python-minimal`** 명령을 실행하여 인스턴스에 Python 2.7을 설치합니다.
6. Node.js가 설치되었는지 확인하고 설치된 경우 버전이 0.6.16 또는 그 이후 버전인지 확인합니다. 버전을 확인하려면 **`node --version`** 명령을 실행하고 표시되는 버전 번호를 적어둡니다. 버전 번호가 표시되지 않거나 버전이 0.6.16 또는 그 이후 버전이 아닐 경우, Node Version Manager(nvm)를 사용하여 인스턴스에 Node.js를 설치하는 것이 좋습니다.

   이렇게 하려면 다음 명령을 다음 순서에 따라 한 번에 하나씩 실행하여 인스턴스를 업데이트하고 인스턴스에 Node Version Manager(nvm)를 설치한 다음, 인스턴스에서 nvm을 활성화하고 나서 인스턴스에 Node.js의 최신 버전을 설치합니다.

   ```
   sudo apt update
   curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh | bash
   . ~/.bashrc
   nvm install node
   ```

7. **`which node`** 명령을 실행하고 표시되는 값을 적어둡니다. 잠시 후 필요한 정보입니다.

> **Note**
>
> **`which node`** 명령의 출력이 `/usr/sbin/node`와 비슷한 경우, AWS Cloud9는 해당 경로에서 Node.js를 찾을 수 없습니다. 대신에 이 절차의 이전 단계에 설명된 대로 `nvm`을 사용하여 Node.js를 설치합니다. 그런 다음 `which node` 명령을 다시 실행하고 표시되는 새 값을 적어둡니다.

8. 해당 인스턴스에서 AWS Cloud9 설치 관리자를 다운로드합니다.

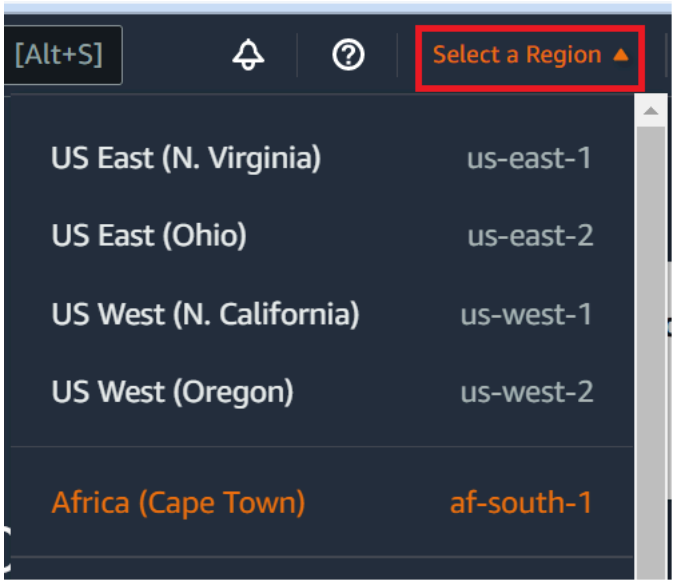
3단계: AWS Cloud9 SSH 개발 환경 생성 및 연결

이 단계에서는 AWS Cloud9 콘솔 및 인스턴스의 터미널을 사용하여 SSH 환경을 생성하고 환경을 실행 중인 인스턴스에 연결합니다.

1. 이전 단계의 터미널 세션이 열려 있는 상태로 다음과 같이 AWS Cloud9 콘솔에 로그인합니다.
   - AWS 계정를 혼자만 사용하는 경우 또는 단일 AWS 계정의 IAM 사용자인 경우 https://console.aws.amazon.com/cloud9/로 이동합니다.
   - 조직에서 AWS IAM Identity Center을 사용하는 경우 로그인 지침은 AWS 계정 관리자를 참조하세요.

> **Note**
>
> 이 단계에서는 서로 다른 두 AWS 서비스를 동시에 사용합니다. Lightsail 콘솔에 IAM 관리자 사용자로 로그인했지만 다른 엔터티에서 새 SSH 환경을 소유하도록 하려는 경우를 가정해 보겠습니다. 이 경우 다른 웹 브라우저를 열고 해당 엔티티로 AWS Cloud9 콘솔에 로그인하는 것이 좋습니다.

2. AWS Cloud9 콘솔에서 인스턴스를 생성한 리전과 일치하는 AWS 리전을 선택합니다.



3. 시작 페이지가 표시되면 새 AWS Cloud9 환경(New AWS Cloud9 environment)에서 환경 생성(Create environment)을 선택합니다. 그렇지 않으면 Create environment(환경 생성)를 선택합니다.

또는 다음과 같습니다.

4. [환경 이름 지정(Name environment)] 페이지의 [이름(Name)]에 환경의 이름을 입력합니다.
5. 환경에 설명을 추가하려면 설명 필드에 내용을 입력합니다.
6. 환경 유형에서 기존 컴퓨팅을 선택합니다. 이는 사용자 및 호스트 옵션을 표시하려면 이 옵션을 선택해야 하므로 중요합니다.
7. User(사용자)에 앞에서 적어둔 User name(사용자 이름) 값을 입력합니다.
8. Host(호스트)에 앞에서 적어둔 Public IP(퍼블릭 IP) 값을 입력합니다.
9. 포트에서 기본값 22를 그대로 둡니다.
10. 추가 세부 정보를 확장합니다.
11. 환경 경로에서 로그인 후 AWS Cloud9가 시작되는 경로(즉, ~/)를 입력합니다. 이것은 사용자의 홈 디렉터리 루트입니다.

12Node.js 바이너리 경로에 앞에서 적어둔 **`which node`** 명령의 값을 입력합니다.

13SSH jump host(SSH 점프 호스트)를 비워 둡니다.

14.시스템 클립보드에 AWS Cloud9에서 이 환경에 대해 생성하는 퍼블릭 SSH 키를 저장합니다. 이렇게 하려면 Copy key to clipboard(클립보드에 키 복사)를 선택합니다.

> **Note**
>
> 복사된 퍼블릭 SSH 키 값을 보려면 View public SSH key(퍼블릭 SSH 키 보기)를 확장합니다.

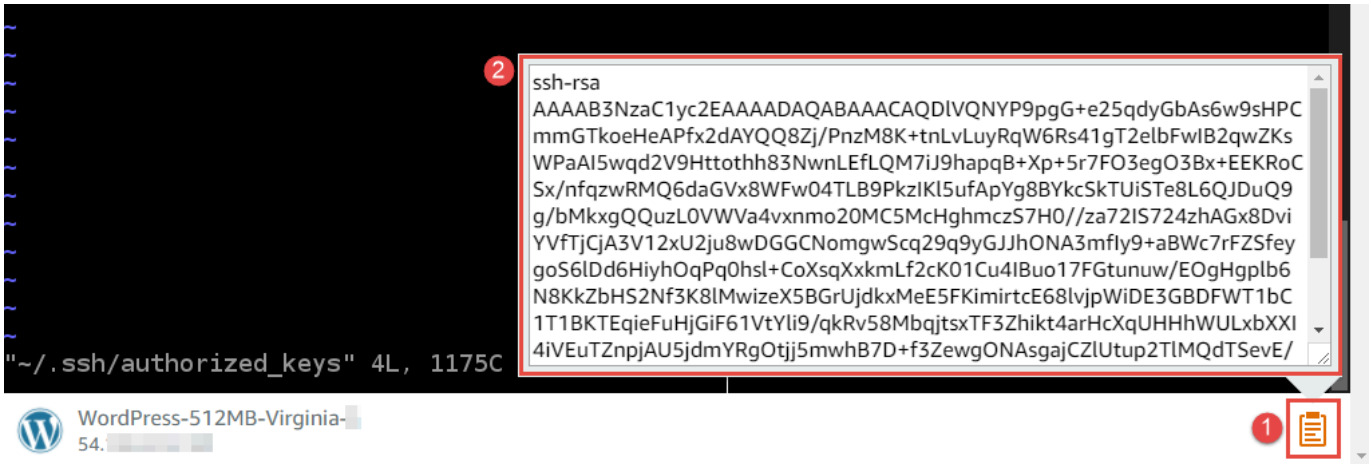
15.방금 복사했던 퍼블릭 SSH 키 값을 인스턴스에 저장합니다. 이렇게 하려면 인스턴스에 이미 설치되어 있는 널리 사용되는 텍스트 편집기인 vi를 사용합니다.

a. 인스턴스의 터미널 세션에서 **`vi ~/.ssh/authorized_keys`** 명령을 실행합니다.

b. 화면에 표시되는 vi 편집기에서 파일의 끝으로 이동하여 삽입 모드로 전환합니다. 이렇게 하려면 `I` 키를 누른 다음 `A` 키를 누릅니다. (-- INSERT --가 vi 편집기의 하단에 표시됩니다.)

c. `Enter`를 두 번 눌러서 파일 끝에 캐리지 리턴 두 개를 추가합니다.

d. 시스템 클립보드의 내용(방금 복사한 퍼블릭 SSH 키 값을 포함함)을 터미널 세션 클립보드에 붙여 넣습니다. 이렇게 하려면 터미널 세션 창의 아래 모서리에서 클립보드 버튼을 선택한 다음 시스템 클립보드의 내용을 상자에 붙여 넣습니다.



e. 터미널 세션 클립보드의 내용을 vi 편집기에 붙여 넣습니다. 이렇게 하려면 vi 편집기의 삽입점에서 `Ctrl + Shift + V`를 누릅니다.

f. 파일을 저장합니다. 이렇게 하려면 `Esc` 키를 눌러 명령 모드로 들어갑니다. (-- INSERT --가 vi 편집기의 하단에서 사라집니다.) `:wq`(파일을 `write`하고 나서 vi 편집기를 `quit`)를 입력한 다음 `Enter` 키를 누릅니다.

16AWS Cloud9 콘솔로 돌아와서 Next step(다음 단계)를 선택합니다.

17 Review choices(선택 항목 검토) 페이지에서 Create environment(환경 생성)를 선택합니다. AWS Cloud9에서 환경을 생성하고 환경용 AWS Cloud9 IDE를 표시할 때까지 기다립니다. 몇 분 정도 걸릴 수 있습니다.

AWS Cloud9에서 환경을 생성하고 나면 환경용 AWS Cloud9 IDE를 표시합니다.

최소 5분 후에도 AWS Cloud9에서 IDE가 표시되지 않으면 웹 브라우저, AWS 액세스 권한, 인스턴스 또는 연결된 가상 사설 클라우드(VPC)에 문제가 있을 수 있습니다. 가능한 해결 방법은 문제 해결에서 환경을 열 수 없음 섹션을 참조하세요.

4단계: AWS Cloud9 IDE를 사용하여 인스턴스에서 코드 변경

IDE가 새 환경에 대해 표시되고, Lightsail 터미널 세션 대신에 IDE의 터미널 세션을 사용할 수 있습니다. IDE는 여러 프로그래밍 언어 및 런타임 디버거를 지원하는 풍부한 코드 편집 환경을 제공합니다. IDE는 색상 테마, 바로 가기 키 결합, 프로그래밍 언어별 구문 색상 지정 및 코드 서식 지정 등도 포함합니다.

IDE 사용 방법을 배우려면 AWS Cloud9 IDE 둘러보기 섹션을 참조하세요.

인스턴스에서 코드를 변경하는 방법에 대해 알아보려면 다음 리소스를 사용하는 것이 좋습니다

- Lightsail 웹 사이트에서 All 'Bitnami 제공' Lightsail 이미지에 대한 애플리케이션 암호 가져오기
- Drupal: BitnamiDrupal - Bitnami 웹 사이트의 AWS 클라우드와 Drupal 웹 사이트의 튜토리얼 및 사이트 레시피
- GitLab CE: BitnamiGitLab CE - Bitnami 웹 사이트의 AWS 클라우드 및 GitLab 웹 사이트의 GitLab 설명서
- Joomla: BitnamiJoomla! - Bitnami 웹 사이트의 AWS 클라우드, Joomla! 웹 사이트에서 Joomla! 시작하기
- LAMP 스택: BitnamiLAMP - Bitnami 웹 사이트의 AWS 클라우드
- Magento: BitnamiMagento - Bitnami 웹 사이트의 AWS 클라우드, Magento 웹 사이트의 Magento User Guide
- MEAN: BitnamiMEAN - Bitnami 웹 사이트의 AWS 클라우드
- NGINX: BitnamiNGINX - Bitnami 웹 사이트의 AWS 클라우드, NGINX 웹 사이트의 NGINX Wiki
- Node.js: BitnamiNode.Js - Bitnami 웹 사이트의 AWS 클라우드, Node.js 웹 사이트의 시작 안내서
- Ubuntu의 Plesk 호스팅 스택: Amazon Lightsail에서 Plesk를 설정하고 구성합니다.

- Redmine: Bitnami Redmine - Bitnami 웹 사이트의 AWS 클라우드, Redmine 웹 사이트에서 시작하기
- WordPress: Lightsail 웹 사이트의 Amazon Lightsail 인스턴스에서 WordPress 사용 시작하기 및 Bitnami 웹 사이트의 AWS 클라우드용 Bitnami WordPress

AWS Cloud9 통합 개발 환경(IDE)의 AWS CodeStar 프로젝트로 작업

AWS Cloud9 IDE를 사용하여 AWS CodeStar 프로젝트의 코드 작업을 수행할 수 있습니다.

AWS CodeStar는 AWS에서 소프트웨어 개발 프로젝트를 생성, 관리, 작업하기 위한 클라우드 기반 서비스입니다. AWS CodeStar 프로젝트를 통해 AWS에서 애플리케이션을 빠르게 개발, 빌드, 배포할 수 있습니다. AWS CodeStar 프로젝트는 프로젝트 개발 도구 체인에 대한 AWS 서비스를 생성 및 통합합니다. 선택한 AWS CodeStar 프로젝트 템플릿에 따라 소스 제어, 빌드, 배포, 가상 서버 또는 서버리스 리소스 등이 해당 도구 체인에 포함될 수 있습니다. 자세한 정보는 AWS CodeStar 사용 설명서를 참조하세요.

> **Note**
>
> 다음 절차를 완료하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2, AWS CodeStar와 같은 서비스 및 AWS CodeStar에서 지원되는 AWS 서비스에 대해 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금, AWS CodeStar 요금 및 클라우드 서비스 요금을 참조하세요.
>
> AWS Cloud9 IDE를 사용하여 WordPress, MySQL, PHP, Node.js, NGINX, Drupal, Joomla 등 인기있는 앱 또는 프레임워크라든가 Ubuntu, Debian, FreeBSD, openSUSE과 같은 Linux 배포를 통해 사전 구성되어 새로 시작된 Amazon EC2 인스턴스로 작업을 수행하기 위해 Amazon Lightsail을 AWS Cloud9와 함께 사용할 수 있습니다. 이렇게 하려면 이 주제의 나머지 부분을 건너뛰고 그 대신 Lightsail 인스턴스 작업을 참조하세요.
>
> AWS Cloud9 IDE를 사용하여 샘플 코드가 포함되지 않은 Amazon Linux를 실행하는 새로 시작된 Amazon EC2 인스턴스로 작업을 수행하려면 이 주제의 나머지 부분을 건너뛰고 그 대신 시작하기: 기본 자습서 섹션을 참조하세요.

- 1단계: AWS CodeStar 프로젝트로 작업하기 위한 준비
- 2단계: AWS CodeStar에서 프로젝트 만들기
- 3단계: AWS Cloud9 개발 환경을 생성하여 프로젝트에 연결

1단계: AWS CodeStar 프로젝트로 작업하기 위한 준비

이 단계에서는 AWS CodeStar 프로젝트 만들기 및 작업을 시작할 수 있도록 AWS CodeStar 서비스 역할과 Amazon EC2 키 페어를 생성합니다.

전에 AWS CodeStar를 사용한 적이 있는 경우 2단계: AWS CodeStar에서 프로젝트 만들기로 진행하세요.

이 단계에서는 AWS CodeStar 사용 설명서의 AWS CodeStar 설정에서 설명하는 지침을 따릅니다. 해당 지침을 따르는 동안 새 AWS 계정, IAM 사용자 또는 IAM 그룹을 생성하지 마세요. AWS Cloud9에 대한 팀 설정에서 생성하거나 식별한 항목을 사용합니다. 해당 지침을 모두 따른 후에는 이 주제로 돌아옵니다.

2단계: AWS CodeStar에서 프로젝트 만들기

이 단계에서는 AWS CodeStar에서 프로젝트를 생성합니다.

사용할 프로젝트가 AWS CodeStar에 이미 있는 경우 3단계: AWS Cloud9 개발 환경을 생성하여 프로젝트에 연결로 진행합니다.

이 단계에서는 AWS CodeStar 사용 설명서의 AWS CodeStar에서 프로젝트 생성에 나와 있는 지침을 따릅니다. AWS CodeStar 프로젝트 만들기 마법사에서 Set up tools(도구 설정) 페이지 또는 Connect to your source repository(소스 리포지토리에 연결) 페이지에 도달하면 Skip(건너뛰기)을 선택한 다음 이 주제로 돌아옵니다.

3단계: AWS Cloud9 개발 환경을 생성하여 프로젝트에 연결

이 단계에서는 AWS CodeStar 또는 AWS Cloud9 콘솔에서 AWS Cloud9 개발 환경을 생성합니다. 그런 다음 새 환경을 AWS CodeStar 프로젝트에 연결합니다.

이 단계에서는 다음과 같은 일련의 지침 중 하나를 따르세요. 이는 사용하려는 AWS Cloud9 개발 환경 유형과 AWS CodeStar 프로젝트에서 코드를 저장하는 리포지토리의 유형에 따라 달라집니다.

환경 유형	리포지토리 유형	지침
EC2 환경	CodeCommit	AWS CodeStar 사용 설명서의 프로젝트에 대한 AWS Cloud9 환경 생성
SSH 환경	CodeCommit	AWS CodeCommit 샘플

환경 유형	리포지토리 유형	지침
EC2 또는 SSH 환경	GitHub	AWS CodeStar 사용 설명서의 GitHub를 AWS Cloud9과 함께 사용

다음을 사용하여 Amazon Q 개발자와 협력하기 AWS Cloud9

Amazon Q란 무엇인가요?

Amazon Q Developer는 애플리케이션을 이해, 구축, 확장 및 운영하는 데 도움이 되는 생성적 인공 지능 (AI) 기반 대화형 도우미입니다. AWS 통합 AWS 코딩 환경에서 Amazon Q는 개발자의 코드뿐만 아니라 자연어로 작성된 주석을 기반으로 코드 권장 사항을 생성할 수 있습니다. Amazon Q는 코드형 인프라 (IaC) 언어인 JSON ()SQL, YAML (), HCL (테라폼AWS CloudFormation) 및 CDK (AWS CloudFormation타이프스크립트, Python) 언어를 가장 많이 지원합니다. Java Python JavaScript TypeScript C# Go PHP Rust Kotlin ,,, 및 에 대한 코드 생성도 지원합니다. Ruby C++ C Shell Scala Amazon Q가 AWS Cloud9 AWS Cloud9 IDE와 통합되고 코드 제안을 표시하는 방법에 대한 예는 Amazon Q 개발자 사용 설명서의 코드 예제를 참조하십시오.

에서 Amazon Q를 사용하는 방법에 대한 자세한 내용은 Amazon Q 개발자 사용 설명서를 참조하십시오. AWS Cloud9

AWS Identity and Access Management 권한은 다음과 같습니다. AWS Cloud9

Amazon Q가 AWS Cloud9 콘솔에서 권장 사항을 제공하려면 IAM 사용자 또는 역할에 대해 올바른 IAM 권한을 활성화해야 합니다. 아래 샘플 IAM 정책에 설명된 대로 `codewhisperer:GenerateRecommendations` 권한을 추가해야 합니다.

> **Note**
>
> `codewhisperer`접두사는 Amazon Q Developer와 병합된 서비스의 기존 이름입니다. 자세한 내용은 Amazon Q 개발자 이름 변경 - 변경 요약을 참조하십시오.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AmazonQDeveloperPermissions",
      "Effect": "Allow",
      "Action": ["codewhisperer:GenerateRecommendations"],
      "Resource": "*"
    }
  ]
}
```

IAM 정책을 사용하여 IAM 보안 주체에 제한적인 권한을 부여하는 것이 가장 좋습니다.

AWS Cloud9 통합 개발 환경(IDE)의 AWS CodePipeline 작업

AWS Cloud9 IDE를 사용하여 AWS CodePipeline과 호환되는 리포지토리의 소스 코드로 작업할 수 있습니다.

CodePipeline은 소프트웨어 릴리스에 필요한 단계를 모델링, 시각화 및 자동화하는 데 사용할 수 있는 지속적 전달 서비스입니다. CodePipeline을 사용하여 소프트웨어 릴리스 프로세스의 다양한 단계를 신속하게 모델링하고 구성할 수 있습니다. 자세한 정보는 AWS CodePipeline 사용 설명서를 참조하세요.

> **Note**
>
> 다음 절차를 완료하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2, CodePipeline, Amazon S3와 같은 서비스 및 CodePipeline에서 지원되는 AWS 서비스에 대해 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금, AWS CodePipeline 요금, Amazon S3 요금 및 클라우드 서비스 요금을 참조하세요.
>
> AWS CodeStar는 파이프라인과 함께 프로젝트 템플릿, 대시보드 및 팀과 같은 추가 기능을 제공합니다. CodePipeline 대신 AWS CodeStar를 사용하려면 이 주제의 나머지 내용을 건너뛰고 그 대신 AWS CodeCommit 프로젝트로 작업을 참조하세요.

- 1단계: 소스 코드 리포지토리 만들기 또는 식별
- 2단계: AWS Cloud9 개발 환경을 생성하고, 이 환경을 코드 리포지토리에 연결한 다음, 코드 업로드
- 3단계: AWS CodePipeline로 작업하기 위한 준비
- 4단계: AWS CodePipeline에서 파이프라인 생성

1단계: 소스 코드 리포지토리 만들기 또는 식별

이 단계에서는 CodePipeline과 호환되는 소스 코드 리포지토리를 생성하거나 식별합니다.

이 주제의 뒷부분에서 소프트웨어의 소스 코드를 해당 리포지토리에 업로드합니다. CodePipeline은 생성한 관련 파이프라인을 사용하여 해당 리포지토리에서 업로드한 소스 코드를 빌드, 테스트 및 배포합니다.

소스 코드 리포지토리는 CodePipeline에서 지원하는 다음 리포지토리 유형 중 하나여야 합니다.

- AWS CodeCommit. 사용할 리포지토리가 CodeCommit에 이미 있는 경우 2단계: AWS Cloud9 개발 환경을 생성하고, 이 환경을 코드 리포지토리에 연결한 다음, 코드 업로드로 진행합니다. 그렇지 않고 CodeCommit을 사용하려면 AWS CodeCommit 샘플의 다음 지침을 다음 순서로 따른 다음 이 주제로 돌아옵니다.
  - 1단계: 필요한 액세스 권한을 사용하여 IAM 그룹 설정
  - 2단계: AWS CodeCommit에서 리포지토리 생성
- Amazon S3. 사용할 버킷이 Amazon S3에 이미 있는 경우 2단계: AWS Cloud9 개발 환경을 생성하고, 이 환경을 코드 리포지토리에 연결한 다음, 코드 업로드로 진행합니다. 그렇지 않은 경우 Amazon S3를 사용하려면 Amazon Simple Storage Service 사용 설명서의 지침을 이 순서대로 따른 후 이 주제로 돌아오십시오.
  - Amazon S3에 가입
  - 버킷 만들기
- GitHub. GitHub에 리포지토리가 이미 있는 경우, Git 패널 인터페이스를 사용하여 이를 복제하고 개발 환경에서 로컬 사본을 만들 수 있습니다. GitHub에 아직 계정 또는 리포지토리가 설정되어 있지 않은 경우 관련 문서에서 자세한 지침을 참조하세요.

2단계: AWS Cloud9 개발 환경을 생성하고, 이 환경을 코드 리포지토리에 연결한 다음, 코드 업로드

이 단계에서는 AWS Cloud9 콘솔에서 AWS Cloud9 개발 환경을 생성합니다. 그런 다음 CodePipeline이 사용할 리포지토리에 환경을 연결합니다. 마지막으로, AWS Cloud9 IDE를 환경에 사용하여 소스 코드를 리포지토리에 업로드합니다.

환경을 생성하려면 환경 생성의 지침을 따른 다음 이 주제로 돌아옵니다. (환경이 이미 있으면 해당 환경을 사용할 수 있으며, 새로 생성할 필요가 없습니다.)

환경을 리포지토리에 연결한 다음, 이 위치에 아직 없는 경우 소스 코드를 리포지토리에 업로드하려면 다음 지침 중 하나를 사용합니다. 선택하는 지침은 소스 코드를 저장하는 리포지토리의 유형에 따라 다릅니다.

리포지토리 유형	지침
CodeCommit	AWS CodeCommit 샘플의 다음 지침을 따릅니다. • 3단계: 원격 리포지토리를 환경에 연결 • 4단계: 원격 리포지토리를 환경에 복제 • 5단계: 파일을 리포지토리에 추가(고유의 소스 코드로 이 단계 대체)
Amazon S3	• AWS CLI 및 AWS CloudShell 샘플 샘플에서 설명하는 것처럼 AWS CLI 또는 AWS CloudShell을 환경에 설치하고 구성합니다. • 소스 코드를 버킷에 업로드하려면 환경에서 AWS CLI 또는 AWS CloudShell을 사용하여 aws s3 cp 명령을 실행합니다. (AWS CloudShell의 경우 명령에서 aws를 제거할 수 있습니다.)
GitHub	Git 패널 인터페이스를 사용하여 GitHub에서 호스트되는 리포지토리를 복제하고 상호 작용할 수 있습니다.

환경을 리포지토리에 연결한 후에는 소스 코드 변경을 AWS Cloud9 IDE에서 리포지토리로 푸시할 때마다 CodePipeline은 빌드, 테스트 및 배포할 관련 파이프라인을 통해 해당 변경 내용을 자동으로 보냅니다. 이 주제의 뒷부분에서 관련 파이프라인을 생성합니다.

3단계: AWS CodePipeline으로 작업하기 위한 준비

이 단계에서는 특정 AWS 관리형 정책을 [팀 설정(Team Setup)]에서 생성하거나 식별한 IAM 그룹에 연결합니다. 이렇게 하면 그룹 사용자가 CodePipeline에서 파이프라인 생성 및 작업을 시작할 수 있습니다.

전에 CodePipeline을 사용한 적이 있는 경우 4단계: AWS CodePipeline에서 파이프라인 생성으로 진행합니다.

이 단계에서는 AWS CodePipeline 사용 설명서의 3단계: IAM 관리형 정책을 사용하여 IAM 사용자에게 AWS CodePipeline 권한 할당에 나와 있는 지침을 따른 다음 이 주제로 돌아옵니다.

4단계: AWS CodePipeline에서 파이프라인 생성

이 단계에서는 이 주제의 앞부분에서 생성하거나 식별한 리포지토리를 사용하는 파이프라인을 CodePipeline에서 생성합니다.

이 단계에서는 AWS CodePipeline 사용 설명서의 AWS CodePipeline에서 파이프라인 생성에 나와 있는 지침을 따릅니다.

파이프라인을 생성한 후 CodePipeline은 빌드, 테스트 및 배포할 파이프라인을 통해 리포지토리에 있는 현재 버전의 소스 코드를 보냅니다. 그런 다음, 소스 코드 변경을 AWS Cloud9 IDE에서 리포지토리로 푸시할 때마다 CodePipeline은 빌드, 테스트 및 배포할 파이프라인을 통해 해당 변경 내용을 자동으로 보냅니다.

파이프라인을 보려면 AWS CodePipeline 사용 설명서의 AWS CodePipeline에서 파이프라인 세부 정보 및 이력 보기에 나와 있는 지침을 따릅니다.

아마존과 협력하기 CodeCatalyst

CodeCatalyst Amazon은 소프트웨어 개발 팀을 위한 클라우드 기반 협업 공간입니다. CodeCatalyst CI/CD (지속적 통합/전달) 도구를 사용하여 작업하고, 코드로 협업하고, 애플리케이션을 구축, 테스트 및 배포할 수 있는 통합된 장소입니다. 공간에 연결하여 AWS 리소스를 프로젝트와 연결할 수 있습니다. AWS 계정 CodeCatalyst 를 CodeCatalyst 사용하여 소프트웨어를 빠르고 확실하게 제공할 수도 있습니다. 에 대한 CodeCatalyst 자세한 내용은 Amazon이란 무엇입니까 CodeCatalyst? 를 참조하십시오. 아마존 CodeCatalyst 가이드에서.

개발 환경은 프로젝트의 소스 리포지토리에 저장된 코드를 작업하는 CodeCatalyst 데 사용할 수 있는 클라우드 기반 개발 환경입니다. 에서 개발 환경을 만들 수 있습니다. CodeCatalyst 그런 다음 지원되는 통합 개발 환경 (IDE) 을 CodeCatalyst 사용하여 프로젝트별 코드 작업을 수행할 수 있습니다. 또는 빈 개발 환경을 만들고 타사 리포지토리의 코드를 복제하여 지원되는 IDE로 작업할 수도 있습니다.

CodeCatalyst 콘솔에서 개발 환경에 액세스하는 데 사용되는 AWS Cloud9 IDE는 에서 실행되는 AWS Cloud9 IDE와 다릅니다. AWS CodeCatalyst AWS Cloud9 IDE에서는 자동으로 CodeCatalyst 로그인되며 IDE 내의 aws-explorer 옵션을 사용하여 서비스에 액세스할 수 있습니다. 툴킷에 대한 자세한 내용은 가이드의 AWS 툴킷 용을 참조하십시오AWS . AWS Cloud9AWS Cloud9

주제

시작하기

이 섹션에서는 사용을 시작하는 방법에 대한 개요를 제공합니다 CodeCatalyst. 이 섹션의 주제에서는 AWS Cloud9 Amazon에서 사용하는 방법과 AWS Cloud9 환경을 CodeCatalyst Amazon에서 복제하는 방법을 다룹니다. CodeCatalyst 이후 항목에서는 개발 환경을 만드는 방법과 IDE를 사용하여 CodeCatalyst 개발 환경에 액세스하는 방법도 자세히 설명합니다. AWS Cloud9

AWS 툴킷은 AWS 클라우드 계정, 서비스, 리소스에 빠르게 액세스할 수 있는 IDE 전용 소프트웨어 개발 키트 (SDK) 입니다. AWS 툴킷의 사용자 CodeCatalyst 계정에서 편리한 인터페이스로 CodeCatalyst 개발 환경, 스페이스 및 프로젝트를 보고, 편집하고, 관리할 수 있습니다. AWS 툴킷을 통해 사용할 수 있는 AWS 클라우드 서비스 및 기능에 대한 자세한 내용은 무엇입니까? 를 참조하십시오. AWS Toolkit for Visual Studio Code, AWS 툴킷 용도 및 AWS Cloud9용도는 무엇입니까? AWS Toolkit for JetBrains . AWS Toolkit for JetBrains가이드는 무엇인가요?

CodeCatalyst AWS Cloud9 IDE와 함께 사용하려면 CodeCatalyst 콘솔에서 만든 기존 스페이스, 프로젝트 및 개발 환경이 있어야 합니다.

> **Note**
>
> AWS Cloud9 IDE의 파일 시스템 내에서 이름이 같은 폴더 내에 projects라는 하위 폴더를 만들지 마십시오. CodeCatalyst 이렇게 하면 이 디렉터리에 있는 어떤 파일에도 액세스할 수 없습니다. 이 문제는 파일 경로 /projects/projects에 영향을 줍니다. /test/projects 및 /projects/test/projts와 같은 파일 경로는 이 문제의 영향을 받지 않습니다. 이는 알려진 문제이며 AWS Cloud9 IDE 파일 탐색기에만 영향을 미칩니다.

> **Note**
>
> 현재는 AWS Cloud9 IDE의 파일 시스템을 사용하여 같은 이름의 폴더 내에 projts라는 하위 폴더를 만들 수 없습니다. CodeCatalyst AWS Cloud9 IDE 파일 탐색기에서는 이 디렉터리 내의

파일에 액세스할 수 없지만 명령줄을 사용하여 액세스할 수 있습니다. 다른 폴더 이름을 사용하십시오. 이 문제는 /projects/projects 파일 경로에만 영향을 미치며, /test/projects와 /projects/test/projects 같은 파일 경로가 적합합니다. 이는 알려진 문제이며 AWS Cloud9 IDE 파일 탐색기에만 영향을 미칩니다.

Amazon의 AWS Cloud9 코드 리소스 복제 CodeCatalyst

AWS Cloud9 CodeCatalyst in은 상호 작용할 수 있는 완전 관리형 환경을 제공합니다. AWS Cloud9 CodeCatalystAmazon에서 현재 AWS Cloud9 코드 리소스를 수동으로 복제할 수 있습니다. 다음 섹션에서는 이 프로세스를 상세히 설명합니다. 코드 리소스를 이동하고 복제하려면 내부에 스페이스를 만드십시오. CodeCatalyst 스페이스는 회사, 부서 또는 그룹을 나타냅니다. 프로젝트, 구성원, 생성한 관련 클라우드 리소스를 추가할 스페이스를 만들어야 합니다. CodeCatalyst 사용자가 프로젝트 초대를 수락하면 CodeCatalyst 자동으로 스페이스에 추가됩니다. 스페이스 관리자 역할을 가진 사용자는 스페이스를 관리할 수 있습니다.

이 스페이스 내에서 프로젝트를 만들고 소스 리포지토리를 추가합니다. 프로젝트는 개발 팀과 작업을 CodeCatalyst 지원하는 협업 공간입니다. 프로젝트를 생성한 후 리소스를 추가, 업데이트 또는 제거할 수 있습니다. 또한 프로젝트 대시보드를 사용자 지정하고 팀 작업의 진행 상황을 모니터링할 수 있습니다. 스페이스 내에 여러 프로젝트를 포함할 수 있습니다. 추가하는 소스 리포지토리 수는 AWS Cloud9 환경에서 이미 사용 중인 리포지토리 수에 따라 달라집니다. 이 프로젝트를 만들고 해당하는 소스 리포지토리를 추가한 후에는 환경으로 돌아가 AWS Cloud9 환경 데이터를 의 새 리포지토리에 복제해야 할 수 있습니다. CodeCatalyst 수행하는 작업은 AWS Cloud9에서 사용 중인 소스 리포지토리의 유형에 따라 달라집니다.

스페이스, 프로젝트 및 소스 리포지토리를 만든 후 Dev Environment와 함께 사용하여 환경을 시작할 수 있습니다. CodeCatalyst AWS Cloud9 개발 환경은 클라우드 기반 개발 환경입니다. 에서 CodeCatalyst 개발 환경을 사용하여 프로젝트의 소스 리포지토리에 저장된 코드를 작업할 수 있습니다. 지원되는 통합 개발 환경 (IDE) 이 있는 프로젝트별 개발 환경에서 CodeCatalyst 코드 작업을 위한 개발 환경을 만들 수도 있습니다.

복제 도구를 사용하여 현재 AWS Cloud9 코드 리소스를 복제할 수도 있습니다. CodeCatalyst 이 도구는 사용자 AWS Cloud9 환경에서 다운로드하여 실행할 수 있습니다. 이미 스페이스에 가입하여 CodeCatalyst 생성한 경우 도구가 자동으로 이 스페이스 내에 프로젝트를 만들고 코드 리소스를 새 리포지토리에 복제합니다. CodeCatalyst 이는 수동 복제 프로세스와 비슷합니다. 이 작업은 AWS Cloud9에서 사용 중인 소스 리포지토리의 유형에 따라 달라집니다. 예를 들어 GitHub 리포지토리가 있는 경우에도 콘솔의 확장 프로그램을 사용하여 리포지토리를 복제해야 합니다. GitHub CodeCatalyst

- 단계 1. Amazon에 CodeCatalyst 가입하고 스페이스 만들기
- 단계 2. 스페이스에서 프로젝트 생성
- 단계 3. 프로젝트에 소스 리포지토리 만들기
- 단계 4. AWS Cloud9 코드 리소스를 다음 소스 리포지토리에 복제 CodeCatalyst
- 단계 5. 사용 시 개발 환경 만들기 CodeCatalyst AWS Cloud9

단계 1. Amazon에 CodeCatalyst 가입하고 스페이스 만들기

기존 스페이스나 프로젝트에 CodeCatalyst 초대하지 않고도 Amazon에 가입할 수 있습니다. 가입하면 스페이스와 프로젝트가 생성됩니다. 사용한 기존 AWS 계정 ID를 입력할 수 있습니다 AWS Cloud9. 청구 목적으로도 동일하게 사용할 AWS 계정 수 있습니다. ID를 찾는 방법에 대한 자세한 내용은 AWS 계정 AWS 계정 ID 및 별칭을 참조하십시오. 다음 절차에 따라 Amazon CodeCatalyst 프로필에 가입하고, 공간을 만들고, 공간에 대한 계정을 추가하십시오.

새 사용자 등록

1. CodeCatalyst 콘솔을 엽니다.
2. 시작 페이지에서 가입을 선택합니다.

   AWS 빌더 ID 만들기 페이지가 표시됩니다. AWS BuilderID는 로그인하기 위해 생성하는 자격 증명입니다. 이 ID는 AWS 계정 ID와 다릅니다. AWS 빌더 ID에 대해 자세히 알아보려면 AWS 로그인 사용 안내서의AWS 빌더 ID 및 기타 AWS 자격 증명을 참조하십시오.
3. 이메일 주소에 연결하려는 이메일 주소를 입력합니다. CodeCatalyst 그리고 다음을 선택합니다.
4. AWS 빌더 ID를 사용하는 애플리케이션에 표시할 이름과 성을 사용자 이름에 입력합니다.

   이 이름은 AWS 빌더 ID 프로필 이름입니다. 원하는 경우 나중에 이름을 변경할 수 있습니다.

   다음을 선택합니다. 이메일 확인 페이지가 나타납니다. 지정한 이메일로 확인 코드가 전송됩니다.
5. 받은 코드를 확인 코드에 입력한 다음 확인을 선택합니다.

   5분 후에도 코드를 받지 못하고 스팸 폴더나 정크 폴더에서 코드를 찾을 수 없다면 코드 재전송을 선택하세요.
6. 코드가 확인되면 암호를 입력하고 암호 확인을 선택합니다.

   AWS 고객 계약 및 AWS 서비스 약관을 읽고 승인했음을 확인하는 확인란을 선택한 다음 Create my profile (Create my profile) 을 선택합니다.

7. 별칭 만들기 페이지에서 사용할 별칭을 입력합니다. CodeCatalyst 다른 CodeCatalyst 사용자들은 댓글과 풀 리퀘스트에서 이 별칭을 사용하여 여러분의 @mention 의견을 남길 수도 있습니다. CodeCatalyst 프로필에는 AWS 빌더 ID의 전체 이름과 CodeCatalyst 별칭이 모두 포함됩니다. CodeCatalyst 별칭은 변경할 수 없습니다.

   전체 이름과 별칭은 의 여러 영역에 표시됩니다. CodeCatalyst 예를 들어 프로필 이름은 활동 피드에 표시되지만 프로젝트 구성원은 사용자의 별칭을 사용자를 @mention하는 데 사용합니다.

   별칭 생성을 선택합니다. 페이지가 업데이트되어 스페이스 생성 섹션이 표시됩니다.
8. 스페이스 이름에 스페이스 이름을 입력하고 다음을 선택합니다.

   이 이름은 변경할 수 없습니다.
9. AWS 계정 ID의 경우 스페이스에 연결하려는 계정의 12자리 ID를 연결합니다.

   AWS 계정 확인 토큰에서 생성된 토큰 ID를 복사합니다. 토큰은 자동으로 복사됩니다. 하지만 AWS 연결 요청을 승인하는 동안 이 정보를 저장하는 것이 좋습니다.
10. 인증을 선택합니다. AWS
11. 에서 Amazon CodeCatalyst 스페이스 확인 페이지가 열립니다 AWS Management Console.

    아마존 CodeCatalyst 스페이스 페이지입니다. 페이지에 액세스하려면 로그인해야 할 수 있습니다.

    페이지에 액세스하려면 에서 Amazon CodeCatalyst Spaces에 AWS Management Console로그인하십시오.

    의 확인 토큰 AWS Management Console 필드는 에서 CodeCatalyst 생성된 토큰으로 자동으로 채워집니다.
12. 스페이스 확인을 선택합니다.

    계정이 스페이스에 추가되었음을 보여주는 계정 확인 성공 메시지가 표시됩니다.

    기본적으로 CodeCatalyst 프리 티어를 사용하게 됩니다. 변경하려면 변경하려면 표준 계층을 활성화하거나 이 공간에 대한 IAM 역할을 추가하려면 공간 세부 정보 보기를 선택합니다.

    CodeCatalyst 요금 계층에 대한 자세한 내용은 Amazon CodeCatalyst - 요금을 참조하십시오.

    CodeCatalyst 공간 세부 정보 페이지가 에서 AWS Management Console열립니다. 아마존 CodeCatalyst 스페이스 페이지입니다. 페이지에 액세스하려면 로그인해야 할 수 있습니다.
13. Amazon으로 이동을 선택합니다 CodeCatalyst.

14. 의 생성 페이지에서 스페이스 생성을 선택합니다. CodeCatalyst

    스페이스가 생성되는 동안 상태 메시지가 표시됩니다. 스페이스가 생성되면 스페이스 페이지가 CodeCatalyst 열립니다. 보기의 기본값은 프로젝트 탭입니다.

    > **Note**
    >
    > 권한 오류 또는 배너가 표시되면 페이지를 새로 고치고 페이지를 다시 확인해 봅니다.

스페이스에 CodeCatalyst 가입하여 스페이스를 만든 후 복제 프로세스의 다음 단계는 이 스페이스 내에 프로젝트를 만드는 것입니다.

단계 2. 스페이스에서 프로젝트 생성

다음 단계에서는 이전 단계에서 생성한 스페이스 내에 빈 프로젝트를 생성하는 방법을 간략히 살펴봅니다. 이 프로젝트를 사용하면 나중에 원하는 리소스를 수동으로 추가할 수 있습니다. 프로젝트를 만들기 전에 스페이스 관리자 역할이 있어야 하며 프로젝트를 만들려는 스페이스에 가입해야 합니다. 스페이스를 만들면 스페이스 관리자 역할이 CodeCatalyst 자동으로 할당됩니다. 스페이스 관리자 역할은 에서 CodeCatalyst 가장 강력한 역할입니다. 이 역할 및 권한에 대한 자세한 내용은 스페이스 관리자 역할을 참조하세요.

빈 프로젝트를 만들려면

1. 프로젝트를 생성하려는 스페이스로 이동합니다.
2. 스페이스 대시보드에서 프로젝트 생성을 선택합니다.
3. 처음부터 시작을 선택합니다.
4. 프로젝트에 이름 부여에서 프로젝트에 할당할 이름을 입력합니다. 이름은 스페이스 내에서 고유해야 합니다.
5. 프로젝트 만들기를 선택합니다.

프로젝트를 만든 후 복제 프로세스의 다음 단계는 소스 리포지토리를 하나 이상 만드는 것입니다.

단계 3. 프로젝트에 소스 리포지토리 만들기

방금 만든 프로젝트 내에서 소스 리포지토리를 만들어야 합니다. 이 리포지토리에는 언제든지 편집하거나 삭제할 수 있는 단일 파일인 README.md 파일이 있습니다. 소스 리포지토리를 만들 때 선택한 사항에 따라 소스 리포지토리에 `.gitignore` 파일이 포함될 수도 있습니다.

소스 리포지토리를 생성하려면

1. CodeCatalyst 콘솔을 엽니다.
2. 프로젝트로 이동합니다.
3. 탐색 창에서 코드를 선택한 다음 소스 리포지토리를 선택합니다.
4. 리포지토리 추가를 선택하고 리포지토리 생성을 선택합니다.
5. 리포지토리 이름에 리포지토리 이름을 제공합니다.

   리포지토리 이름은 프로젝트에서 고유해야 합니다. 리포지토리 이름 요구 사항에 대한 자세한 내용은 의 소스 리포지토리 할당량을 참조하십시오. CodeCatalyst
6. (선택 사항) 설명에 리포지토리에 대한 설명을 추가하여 프로젝트의 다른 사용자가 리포지토리의 용도를 이해하는 데 도움이 되도록 합니다.
7. (선택 사항) 푸시하려는 코드 유형에 맞는 `.gitignore` 파일을 추가합니다.
8. 생성을 선택하세요.

   > **Note**
   >
   > CodeCatalyst `README.md`파일을 만들 때 저장소에 파일을 추가합니다. CodeCatalyst 또한 main이라는 기본 브랜치에 리포지토리에 대한 초기 커밋을 생성합니다. README.md 파일을 편집하거나 삭제할 수 있지만 기본 브랜치를 변경하거나 삭제할 수는 없습니다.
9. 소스 리포지토리 복제 URL 및 PAT를 가져오려면 리포지토리 복제를 선택합니다.
10. HTTPS 복제 URL과 PAT를 각각 복사하려면 복사를 선택합니다. 그런 다음 복제 URL과 PAT를 검색할 수 있는 위치에 저장합니다.

    복제 URL과 PAT는 4단계에서 사용되며, `CODECATALYST_SOURCE_REPO_CLONE_URL` 및 `CODECATALYST_PAT`로 참조됩니다.

프로젝트 내에 소스 리포지토리를 만든 후 AWS Cloud9 데이터를 이 소스 리포지토리에 복제합니다.

4단계. AWS Cloud9 코드 리소스를 의 소스 리포지토리에 복제하기 CodeCatalyst

AWS Cloud9 환경에 있는 소스 리포지토리의 유형에 따라 코드 리소스를 생성한 CodeCatalyst 소스 리포지토리로 가져오기 위해 따르는 복제 방법이 결정됩니다. 옵션은 다음과 같습니다.

- 에서 GitHub 리포지토리 사용 AWS Cloud9
- 예를 GitHub 들어 Bitbucket이 아닌 GitLab 리포지토리 사용 AWS Cloud9

- AWS Cloud9에서 빈 리포지토리 사용 이 옵션은 AWS Cloud9에서 소스 리포지토리를 사용하지 않을 것임을 의미합니다.

의 리포지토리 사용 GitHub CodeCatalyst

GitHub리포지토리 확장을 사용하면 Amazon 프로젝트의 연결된 GitHub 리포지토리를 사용할 수 있습니다. AWS Cloud9 CodeCatalyst 다음 단계는 카탈로그에서 GitHub 확장 프로그램을 설치하는 방법을 설명합니다. CodeCatalyst 이 단계에서는 기존 GitHub 계정을 CodeCatalyst 스페이스에 연결하고 GitHub 리포지토리를 CodeCatalyst 프로젝트에 연결하는 방법도 보여줍니다.

이 방법의 첫 번째 단계는 카탈로그에서 GitHub리포지토리 확장을 설치하는 것입니다 CodeCatalyst . 확장을 설치하려면 다음 단계를 수행합니다.

> **Important**
>
> Github 리포지토리 확장 설치 및 구성의 일환으로 GitHub 계정에 확장을 설치해야 합니다. 이렇게 하려면 GitHub 계정 관리자와 CodeCatalyst 스페이스 관리자여야 합니다.

단계 1. CodeCatalyst 카탈로그에서 확장 프로그램을 설치하려면

1. CodeCatalyst 콘솔을 엽니다.
2. 스페이스로 이동합니다.

   > **Tip**
   >
   > 둘 이상의 스페이스에 속해 있는 경우 상단 탐색 표시줄에서 보려는 스페이스를 선택할 수 있습니다.

3. 검색 창 옆의 CodeCatalyst 상단 메뉴 표시줄에서 카탈로그 아이콘을 선택하여 카탈로그로 이동합니다. GitHub 리포지토리를 검색하거나 범주를 기준으로 확장을 필터링할 수 있습니다.
4. (선택 사항) 확장과 관련된 권한 등 확장에 대한 자세한 내용을 보려면 GitHub 리포지토리 확장 이름을 선택합니다.
5. 설치를 선택합니다. 확장에 필요한 권한을 검토하고 계속하려면 설치를 다시 선택합니다.

GitHub 리포지토리 확장을 설치하면 연결된 GitHub 계정과 연결된 GitHub 리포지토리를 보고 관리할 수 있는 GitHub 리포지토리 확장 세부 정보 페이지로 이동합니다.

GitHub리포지토리 확장 프로그램을 설치한 후 GitHub 계정을 스페이스에 연결합니다. CodeCatalyst GitHub 계정을 연결하려면 다음 단계를 수행합니다.

단계 2. 계정을 GitHub 연결하려면 CodeCatalyst

1. 연결된 Github 계정 탭에서 GitHub 계정 연결을 선택하여 GitHub의 외부 사이트로 이동합니다.
2. GitHub자격 증명을 사용하여 GitHub 계정에 로그인한 다음 Amazon을 설치할 계정을 선택합니다 CodeCatalyst.
3. 현재 및 미래의 모든 리포지토리에 대한 액세스를 CodeCatalyst 허용할지 여부를 선택합니다. 또는 사용하려는 특정 GitHub 저장소를 선택할 수도 있습니다. CodeCatalyst 기본 옵션은 GitHub 스페이스의 모든 GitHub 리포지토리입니다.
4. 부여된 권한을 검토한 다음 설치를 선택합니다. CodeCatalyst

GitHub계정을 CodeCatalyst 연결한 후 GitHub리포지토리 확장 세부 정보 페이지의 GitHub계정 탭에서 연결된 계정을 볼 수 있습니다.

에서 CodeCatalyst 리포지토리를 사용하기 위한 마지막 단계는 GitHub 리포지토리를 사용하려는 CodeCatalyst 프로젝트에 연결하는 것입니다. GitHub리포지토리를 CodeCatalyst 프로젝트에 연결하려면 전체 프로세스의 3단계에 설명된 다음 단계를 수행하십시오.

단계 3. GitHub리포지토리 확장 세부정보 페이지에서 GitHub 리포지토리를 CodeCatalyst 프로젝트에 연결하려면

1. 연결된 GitHub 리포지토리 탭에서 GitHub 리포지토리 연결을 선택합니다.
2. GitHub 계정의 경우, 연결하려는 리포지토리가 포함된 GitHub 계정을 선택합니다.
3. GitHub리포지토리의 경우 프로젝트에 연결하려는 리포지토리를 선택합니다. CodeCatalyst
4. CodeCatalyst 프로젝트의 경우 GitHub 리포지토리를 연결하려는 CodeCatalyst 프로젝트를 선택합니다.
5. 연결을 선택합니다.

이제 방금 푸시한 업데이트된 파일과 커밋이 CodeCatalyst 저장소에 있을 것입니다. 이제 이 브랜치에서 개발 환경을 만들고 AWS Cloud9을 사용하여 열 수 있습니다. 개발 환경에 대한 자세한 내용은 의 개발 환경을 참조하십시오. CodeCatalyst

이제 이 브랜치에서 개발 환경을 만들고 AWS Cloud9을 사용하여 열 수 있습니다. 이 작업을 수행하는 단계는 5단계: in을 사용하여 개발 환경 만들기에 요약되어 있습니다. AWS Cloud9 CodeCatalyst

리포지토리가 아닌 곳 GitHub 사용 CodeCatalyst

리포지토리가 아닌 곳에서 환경을 복제하려면 CodeCatalyst 먼저 Amazon에서 개인 액세스 토큰(PAT) AWS Cloud9 을 생성해야 합니다. GitHub 다음 섹션에서는 이 토큰을 생성하는 방법을 간략하게 설명합니다.

Amazon에서 개인용 액세스 토큰 생성 CodeCatalyst

프로젝트에서 만든 소스 리포지토리는 Git 클라이언트가 있는 로컬 컴퓨터 또는 통합 개발 환경(IDE)에서 액세스할 수 있습니다. 이렇게 하려면 애플리케이션별 암호를 입력해야 합니다. 이 용도로 사용할 개인용 액세스 토큰(PAT)을 만들 수 있습니다. 생성한 개인용 액세스 토큰 (PAT) 은 내 모든 공간 및 프로젝트에서 CodeCatalyst 사용자 ID와 연결됩니다. 생성한 PAT의 이름과 만료 날짜를 볼 수 있으며 더 이상 필요하지 않은 PAT는 삭제할 수 있습니다. PAT 비밀은 생성할 때만 복사할 수 있습니다.

개인용 액세스 토큰(PAT)을 만들려면

1. https://codecatalyst.aws/ 에서 CodeCatalyst 콘솔을 엽니다.
2. 상단 메뉴 바에서 프로필 배지를 선택한 다음 내 설정을 선택합니다.

   > **Tip**
   >
   > 사용자 프로필도 찾을 수 있습니다. 이렇게 하려면 프로젝트 또는 스페이스의 구성원 페이지에 있는 구성원 목록에서 이름을 선택합니다.

3. 개인용 액세스 토큰에서 생성을 선택합니다.
4. PAT 이름에 개인용 액세스 토큰(PAT)을 설명하는 이름을 입력합니다.
5. 만료 날짜에 기본 날짜를 유지하거나 달력 아이콘을 선택하여 사용자 지정 날짜를 선택합니다. 만료 날짜는 기본적으로 현재 날짜로부터 1년 후입니다.
6. 생성을 선택하세요.

   > **Tip**
   >
   > 소스 리포지토리의 복제 리포지토리를 선택할 때도 이 토큰을 생성할 수 있습니다.

7. PAT 비밀을 복사하려면 복사를 선택합니다. PAT 비밀을 검색할 수 있는 곳에 저장합니다.

> **Important**
>
> PAT 비밀은 한 번만 표시됩니다. 창을 닫은 후에는 검색할 수 없습니다. PAT 비밀을 안전한 위치에 저장하지 않은 경우 다른 비밀을 만들 수 있습니다.

소스 리포지토리에 대한 PAT를 생성한 후에는 아래 섹션에 설명된 대로 AWS Cloud9 환경에 원격 리포지토리를 추가하고 데이터를 이 리포지토리로 푸시하여 AWS Cloud9 환경의 데이터를 복제하십시오. CodeCatalyst

사용자 환경에 원격 리포지토리 추가 AWS Cloud9

GitHub 리포지토리가 아닌 리포지토리를 실행하고 있다고 가정해 보겠습니다. AWS Cloud9 환경에 원격 리포지토리를 추가하고 에서 소스 리포지토리로 데이터를 푸시할 수 있습니다. CodeCatalyst 이 프로세스를 완료하려면 다음 명령을 실행합니다.

AWS Cloud9 IDE 내에서 복제 프로세스의 3단계에서 만든 원본 리포지토리를 가리키는 원격 리포지토리를 추가합니다 CodeCatalyst. 명령의 CODECATALYST_SOURCE_REPO_CLONE_URL을 3단계: 프로젝트에 소스 리포지토리 생성의 10단계에서 저장한 복제 URL로 바꿉니다.

```
git remote add codecatalyst CODECATALYST_SOURCE_REPO_CLONE_URL
```

다음 명령을 사용하여 새 브랜치를 소스 리포지토리로 푸시합니다. 암호를 입력하라는 메시지가 표시되면 3단계: 프로젝트에 소스 리포지토리 생성의 10단계에서 저장한 CODECATALYST_PAT를 사용합니다.

```
git checkout -b replication && git push codecatalyst replication
```

다음은 예상되는 명령 실행 출력의 예입니다.

```
Switched to a new branch 'replication'
Password for 'https://[aws-account-id]@[aws-region].codecatalyst.aws/v1/
MySpace222581768915/Replication/Repository':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 982 bytes | 122.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
remote: Validating objects: 100%
```

```
To https://[aws-account-id].codecatalyst.aws/v1/MySpace222581768915/Replication/
Repository
* [new branch] replication # replication
```

이 브랜치는 에서 만든 소스 리포지토리에서 사용할 수 CodeCatalyst 있습니다. 이 브랜치에서 개발 환경을 만들고 AWS Cloud9을 사용하여 열 수 있습니다. 개발 환경에 대한 자세한 내용은 의 개발 환경을 참조하십시오. CodeCatalyst

이제 이 브랜치에서 개발 환경을 만들고 AWS Cloud9을 사용하여 열 수 있습니다. 이 작업을 수행하는 단계는 5단계: in을 사용하여 개발 환경 만들기에 요약되어 있습니다. AWS Cloud9 CodeCatalyst

에서 빈 리포지토리 사용 AWS Cloud9

빈 리포지토리를 AWS Cloud9 사용하여 환경을 복제하려면 CodeCatalyst 먼저 Amazon에서 개인 액세스 토큰 (PAT) 을 생성하십시오. 다음 섹션에서는 이 토큰을 생성하는 방법을 간략하게 설명합니다.

Amazon에서 개인용 액세스 토큰 생성 CodeCatalyst

프로젝트에서 만든 소스 리포지토리는 Git 클라이언트가 있는 로컬 컴퓨터 또는 통합 개발 환경(IDE)에서 액세스할 수 있습니다. 이렇게 하려면 애플리케이션별 암호를 입력해야 합니다. 이 용도로 사용할 개인용 액세스 토큰(PAT)을 만들 수 있습니다. 생성한 개인용 액세스 토큰 (PAT) 은 내 모든 공간 및 프로젝트에서 CodeCatalyst 사용자 ID와 연결됩니다. 생성한 PAT의 이름과 만료 날짜를 볼 수 있으며 더 이상 필요하지 않은 PAT는 삭제할 수 있습니다. PAT 비밀은 생성할 때만 복사할 수 있습니다.

개인용 액세스 토큰(PAT)을 만들려면

1. https://codecatalyst.aws/ 에서 CodeCatalyst 콘솔을 엽니다.
2. 상단 메뉴 바에서 프로필 배지를 선택한 다음 내 설정을 선택합니다.

   > **Tip**
   >
   > 사용자 프로필도 찾을 수 있습니다. 이렇게 하려면 프로젝트 또는 스페이스의 구성원 페이지에 있는 구성원 목록에서 이름을 선택합니다.

3. 개인용 액세스 토큰에서 생성을 선택합니다.
4. PAT 이름에 개인용 액세스 토큰(PAT)을 설명하는 이름을 입력합니다.
5. 만료 날짜에 기본 날짜를 유지하거나 달력 아이콘을 선택하여 사용자 지정 날짜를 선택합니다. 만료 날짜는 기본적으로 현재 날짜로부터 1년 후입니다.

6. 생성을 선택하세요.

> **Tip**
>
> 소스 리포지토리의 복제 리포지토리를 선택할 때도 이 토큰을 생성할 수 있습니다.

7. PAT 비밀을 복사하려면 복사를 선택합니다. PAT 비밀을 검색할 수 있는 곳에 저장합니다.

> **Important**
>
> PAT 비밀은 한 번만 표시됩니다. 창을 닫은 후에는 검색할 수 없습니다. PAT 비밀을 안전한 위치에 저장하지 않은 경우 다른 비밀을 만들 수 있습니다.

소스 리포지토리에 대한 PAT를 생성한 후에는 아래 섹션에 설명된 대로 AWS Cloud9 환경에서 빈 리포지토리를 시작하고 CodeCatalyst 생성한 소스 리포지토리를 가리켜 AWS Cloud9 환경의 데이터를 다른 곳으로 복제하십시오. CodeCatalyst

에서 빈 리포지토리 시작 AWS Cloud9

에 설정된 소스 리포지토리가 없는 AWS Cloud9경우 에서 빈 리포지토리를 시작하십시오. AWS Cloud9또한 에서 CodeCatalyst 만든 소스 리포지토리를 가리키고 복제하려는 파일을 추가하고 푸시하세요. Git 다음 단계를 수행하고 다음 명령을 실행하여 AWS Cloud9 파일을 복제합니다. CodeCatalyst

1. 사용자 AWS Cloud9 환경에서 다음 명령을 실행하여 빈 리포지토리를 시작합니다.

```
git init -b main
```

   그러면 아래와 같은 유사한 출력이 표시됩니다.

```
Initialized empty Git repository in /home/ec2-user/environment/.git/
```

2. 에서 CodeCatalyst 소스 리포지토리 URL을 복제합니다. CodeCatalyst 콘솔에서 만든 CodeCatalyst 프로젝트로 이동한 다음 탐색 창에서 코드를 선택한 다음 소스 리포지토리를 선택합니다.
3. 원하는 소스 리포지토리 목록에서 리포지토리를 선택하고 복제 리포지토리를 선택하여 복제 URL을 복사합니다.
4. 복제한 URL을 사용하여 CodeCatalyst 리포지토리를 추가하고 빈 리포지토리에 이미 있는 콘텐츠를 다음 위치로 푸시합니다. CodeCatalyst

```
git remote add origin [...]
git push origin --force
```

5. 복제할 파일을 추가합니다. 환경 디렉터리의 모든 파일을 복제하려면 `git add -A`를 실행합니다.

```
git add -A .
git commit -m "replicate"
```

6. 관련 없는 두 기록을 병합합니다. 병합 충돌이 발생하는 경우 다음과 같이 해결합니다.

```
git merge origin/main --allow-unrelated-histories
```

7. 다음 명령을 CodeCatalyst 실행하여 변경 내용을 소스 리포지토리로 다시 푸시합니다. 암호를 입력하라는 메시지가 표시되면 3단계: 프로젝트에 소스 리포지토리 생성의 10단계에서 생성한 개인 액세스 토큰(CODECATALYST_PAT)을 입력합니다.

```
Admin:~/environment (main) $ git push origin main
Password for 'https://222581768915@git.us-west-2.codecatalyst.aws/v1/
MySpace222581768915/Replication/Replication':
```

이 절차를 완료하면 방금 푸시한 업데이트된 파일과 커밋이 CodeCatalyst 저장소에 저장됩니다. 이제 이 브랜치에서 개발 환경을 만들고 AWS Cloud9을 사용하여 열 수 있습니다. 이 작업을 수행하는 단계는 아래 섹션에 요약되어 있습니다.

5단계: in을 사용하여 개발 환경 만들기 AWS Cloud9 CodeCatalyst

다음 절차는 방금 복제한 AWS Cloud9 데이터와 CodeCatalyst 사용 중인 개발 환경을 만드는 방법을 설명합니다.

를 사용하여 개발 환경을 만들려면 AWS Cloud9

1. https://codecatalyst.aws/ 에서 CodeCatalyst 콘솔을 엽니다.
2. 개발 환경을 생성하려는 스페이스로 이동합니다.
3. 탐색 창에서 개요를 선택한 다음 내 개발 환경 섹션으로 이동합니다.
4. 개발 환경 생성을 선택합니다.
5. 드롭다운 AWS Cloud9 메뉴에서 선택합니다.
6. 리포지토리 복제를 선택합니다.

> **Note**
>
> 현재는 타사 리포지토리 복제를 CodeCatalyst 지원하지 않지만, 개발 환경을 만들고 선택한 IDE에서 타사 리포지토리를 복제할 수 있습니다.

7. 다음 중 하나를 수행하십시오.

   a. 복제할 리포지토리를 선택하고, 기존 브랜치에서 작업을 선택한 다음 기존 브랜치 드롭다운 메뉴에서 브랜치를 선택합니다.

   b. 복제할 리포지토리를 선택하고, 새 브랜치에서 작업을 선택하고, 브랜치 이름 필드에 브랜치 이름을 입력하고, 다음에서 브랜치 생성 드롭다운 메뉴에서 새 브랜치를 만들 브랜치를 선택합니다.

8. 원하는 경우 개발 환경의 별칭을 추가할 수 있습니다.
9. 선택적으로 개발 환경 구성 편집 버튼을 선택하여 개발 환경의 컴퓨팅, 스토리지 또는 제한 시간 구성을 편집합니다.
10. 생성을 선택하세요. 개발 환경이 생성되는 동안 개발 환경 상태 열에 시작 중이 표시되고, 개발 환경이 생성되면 상태 열에 실행 중이 표시됩니다.

복제 도구 사용

AWS Cloud9 CodeCatalyst iN은 상호 작용을 위한 완전 관리형 환경을 제공합니다. AWS Cloud9고객이 in을 사용해 AWS Cloud9 볼 수 있도록 복제 도구를 만들었습니다. CodeCatalyst AWS Cloud9 환경에서 스크립트를 복사하여 실행한 후에는 프롬프트에 따라 스크립트를 실행하고 에서 AWS Cloud9 로 코드 리소스를 복제하십시오. CodeCatalyst 복제 도구 및 프로세스에 대한 자세한 내용은 아래에 설명된 복제 프로세스에 대한 FAQ를 참조하세요.

> **Note**
>
> 이 복제 프로세스는 기존 AWS Cloud9 환경에 영향을 주지 않습니다. 복제 프로세스가 완료된 후 개발 환경, 소스 리포지토리, 프로젝트 및 공간을 삭제할 수 있으며 이 경우 환경에 영향을 주지 않습니다. AWS Cloud9 이 도구는 코드 리소스만 AWS Cloud9 in에 복사할 뿐 CodeCatalyst 기존 환경을 삭제하거나 구성하지는 않습니다. AWS Cloud9 이 복제 도구는 초기 엄선된 AWS 계정 그룹에 출시되었습니다. 따라서 특정 AWS 계정에는 표시되지 않을 수 있습니다.

> **Note**
>
> 도구를 다운로드하기 전에 Amazon에 CodeCatalyst 가입하고 공간을 생성하는 것이 좋습니다. 가입에 CodeCatalyst 대한 자세한 내용은 Amazon에 가입 CodeCatalyst 및 스페이스 생성을 참조하십시오.

AWS Cloud9 Amazon에서 사용할 때의 이점 CodeCatalyst

다음 섹션에서는 에서 사용할 때 경험할 수 있는 몇 가지 성능상의 이점과 향상된 기능에 AWS Cloud9 대해 CodeCatalyst 간략히 설명합니다.

- CodeCatalyst 완전 관리형 개발 환경을 사용하여 단일 위치에서 전체 소프트웨어 개발 수명 주기를 관리할 수 있는 통합 환경을 제공합니다.
- 출시 시 Amazon EBS 볼륨 크기 옵션이 향상되었습니다.
- 임시 환경을 지원하고 필요에 따라 개발 환경의 컴퓨팅을 확장할 수 있습니다.
- 사용자 지정 이미지 사양을 통해 사용할 수 있는 사용자 지정 AMI 지원.
- 구성을 코드로 설명할 수 있는 Devfile 지원

복제 도구를 CodeCatalyst 사용하여 AWS Cloud9 코드 리소스를 복제하기

다음 절차에서는 복제 도구를 복사하고 실행하여 복제 프로세스를 완료하는 방법을 자세히 설명합니다.

1. 아래 스크립트를 복사하고 AWS Cloud9 환경 내에서 실행해야 합니다.

```
curl https://dx5z5embsyrja.cloudfront.net -o /tmp/replicate-tool.tar.gz && tar
 --no-same-owner --no-same-permissions -xvf /tmp/replicate-tool.tar.gz -C /tmp &&
 node /tmp/cloud9-replication-tools
```

2. [선택 사항] 복제 도구는 원격 측정에 사용자의 AWS 계정 ID를 사용합니다. 이 도구의 목적은 도구를 사용하는 동안 발생할 수 있는 문제를 더 잘 식별할 수 있도록 하는 것입니다. `tool starts`, `tool fails`, `tool is cancelled by user`, `tool completes successfully` 및 `tool creates a Dev Environment for the user`에 대한 원격 측정 이벤트가 발생합니다. 복제 도구를 사용하여 원격 분석을 사용하지 않도록 설정하려면 아래 복제 도구의 원격 측정 비활성화를 참조하세요.

3. AWS Cloud9 환경에서 복제 도구를 복사하고 실행한 후에는 브라우저에서 액세스 AWS 계정 URL로 이동한 다음 10분 이내에 허용을 클릭하여 AWS 빌더 ID와 연결해야 합니다. 링크는 한 번만 열어주세요. 링크를 여러 번 열면 오류가 발생하고 다시 시작해야 합니다. AWS Builder ID에 대한 자세한 내용은 로그인 사용 안내서의 AWS Builder ID로AWS 로그인을 참조하십시오. 이렇게 하면 복제 도구에 코드 리소스를 복제할 목적으로 코드 리소스에 대한 액세스 권한이 부여됩니다. CodeCatalyst
4. 사용하려는 스페이스를 선택합니다. 스페이스가 하나뿐인 경우 해당 스페이스가 선택됩니다. 스페이스에 대한 자세한 내용은 Amazon CodeCatalyst 사용 설명서의 스페이스를 참조하십시오. CodeCatalyst
5. 코드를 복제할지 CodeCatalyst 아니면 새 개발 환경에서 시도할지 선택하십시오. 에서 직접 코드를 복제하는 것이 좋습니다. CodeCatalyst 개발 환경에 대한 자세한 내용은 Amazon CodeCatalyst 사용 설명서의 개발 환경을 참조하십시오. CodeCatalyst
6. 프로젝트 이름을 입력하거나 Enter 키를 눌러 제공된 기본 이름을 사용합니다.
7. 메시지가 표시되면 에서 새 소스 리포지토리에 파일을 복사할 방법을 선택합니다. CodeCatalyst 루트 폴더를 단일 CodeCatalyst 리포지토리로 푸시하거나 하위 폴더를 별도의 CodeCatalyst 리포지토리로 푸시할 수 있습니다.
8. 도구가 완성되면 터미널 메시지에 제공된 URL을 통해 CodeCatalyst 콘솔 내에서 프로젝트로 이동하여 코드 리소스에 액세스할 수 있습니다. CodeCatalyst

이 절차를 완료하면 방금 푸시한 업데이트된 파일과 커밋이 CodeCatalyst 저장소에 저장됩니다. 이제 이 브랜치에서 개발 환경을 만들고 AWS Cloud9을 사용하여 열 수 있습니다.

복제 도구에 대한 원격 측정 비활성화

다음 단계에서는 복제 도구에 대한 원격 측정을 비활성화하도록 환경 변수를 설정하는 방법을 간략하게 설명합니다.

1. 사용자 AWS Cloud9 환경에서 터미널을 엽니다.
2. 다음과 같은 명령 중 하나를 실행합니다.

```
export CLOUD9_REPLICATION_TOOL_TELEMETRY=off
```

또는

```
export CLOUD9_REPLICATION_TOOL_TELEMETRY=0
```

3. 위 명령 중 하나를 실행하면 환경 변수가 설정되고 복제 도구에 대한 원격 측정이 비활성화됩니다. 원격 측정을 비활성화하도록 설정한 후에는 복제 도구 스크립트를 복사하고 다시 실행하여 프로세스를 시작해야 합니다.

복제 도구 피드백

문제가 발생하거나 복제 도구 사용 경험에 대한 피드백을 제공하려면 지원 사례를 작성하여 제출하세요. 지원 사례 생성에 대한 자세한 내용은 지원 사례 및 사례 관리 생성을 참조하세요.

아마존과의 AWS Cloud9 차이점 CodeCatalyst

다음 표에는 두 제품 간의 몇 가지 차이점이 AWS Cloud9 요약되어 AWS Cloud9 있습니다. CodeCatalyst

AWS Cloud9	AWS Cloud9 아마존에서 CodeCatalyst
프라이빗 VPC는 다음과 매우 잘 작동합니다. AWS Cloud9	프라이빗 VPC의 사용은 현재 on에서 지원되지 않습니다. AWS Cloud9 CodeCatalyst
AWS Cloud9 사전 구성된 AWS 관리형 자격 증명을 지원합니다.	AWS Cloud9 CodeCatalyst켜려면 자격 증명을 수동으로 구성해야 합니다.
30분에서 7일 사이의 간격을 두고 를 사용하여 시스템 종료를 비활성화할 수 있습니다. AWS Cloud9	15분 ~ 20시간 간격으로 켤 수 CodeCatalyst 있으며 시스템 종료를 AWS Cloud9 비활성화할 수는 없습니다.
AWS Cloud9 우분투 및 AL2 OS 플랫폼을 지원합니다.	AWS Cloud9 CodeCatalyst on은 MDE 유니버설 이미지와 우분투와 AL2를 포함할 수 있는 사용자 지정 이미지를 지원합니다. 이에 대한 자세한 내용은 Amazon CodeCatalyst 사용 설명서의 범용 devfile 이미지를 참조하십시오.
업로드 및 다운로드는 에서 지원됩니다. AWS Cloud9	현재 on에서는 업로드 및 다운로드가 지원되지 않습니다. AWS Cloud9 CodeCatalyst 사용자는 Amazon S3 버킷을 사용하여 업로드하고 다운로드해야 합니다.
에서 공동 작업을 수행할 수 있습니다. AWS Cloud9	현재 AWS Cloud9 온에서는 공동 작업을 사용할 수 없습니다 CodeCatalyst.

복제 프로세스에 대한 FAQ

다음 섹션은 복제 도구 및 복제 프로세스와 관련된 몇 가지 FAQ에 대한 답변을 제공하는 것을 목표로 합니다.

질문: 환경을 에 복제하면 내 AWS Cloud9 환경이 영향을 CodeCatalyst 받나요? AWS Cloud9

답변: 아니요. 환경 복제는 코드 리소스만 복사하여 작업을 계속할 AWS Cloud9 CodeCatalyst 수 있도록 합니다. 코드 리소스와 AWS Cloud9 환경은 어떤 식으로든 영향을 받지 않습니다.

질문: 롤백하려는 경우 내 AWS Cloud9 환경이 영향을 받나요?

답변: 아니요. CodeCatalyst 개발 환경, 소스 리포지토리, 프로젝트, 스페이스는 삭제할 수 있으며 환경에 영향을 주지 않습니다. AWS Cloud9

질문: 새 위치는 HIPAA, SOC 등과 같은 표준을 준수하나요?

답변: 의 개발 CodeCatalyst 환경은 현재 이러한 표준을 준수하지 않습니다. 이러한 표준의 준수는 로드맵에 포함되어 있습니다.

질문: 코드 리소스는 어디로 이동하나요?

답변: 코드 리소스가 프로젝트 내의 소스 리포지토리에 복사됩니다. CodeCatalyst

질문: 사용량이 제한되나요?

답변: 복제 프로세스의 일환으로 프리 티어 내에 16GB의 개발 환경을 만들게 됩니다. 즉, 최대 4개의 개발 환경을 사용할 수 있습니다. 가격, 보관 및 사용 가능한 등급에 대한 자세한 내용은 Amazon CodeCatalyst - 요금을 참조하십시오.

질문: 내 컴퓨팅은 어디로 이동하나요?

답변: 기존 컴퓨팅은 변경되지 않습니다. 그대로 유지됩니다.

질문: 기존 AWS 계정 자격 증명을 에서 CodeCatalyst 사용할 수 있나요? 그러면 자동으로 이전되나요?

답변: 에서 AWS 계정 자격 증명을 수동으로 구성할 수 CodeCatalyst 있습니다. 자동으로 전송되지는 않습니다.

질문: 비용이 얼마나 드나요?

답변: CodeCatalyst 무료로 사용을 시작할 수 있습니다. 가격 및 사용 가능한 등급에 대한 자세한 내용은 Amazon CodeCatalyst - 요금을 참조하십시오.

질문: 데이터 복제 프로세스와 데이터 스토리지는 CodeCatalyst 안전한가요?

답변: 예, git push와 https를 사용하여 코드 리소스를 복사하고 서비스 내에 데이터를 CodeCatalyst 안전하게 저장할 것입니다. 모든 데이터는 전송 및 저장 시 암호화됩니다. 의 데이터 보호에 대한 자세한 내용은 Amazon CodeCatalyst 사용 설명서의 CodeCatalyst Amazon에서의 데이터 보호를 참조하십시오. CodeCatalyst

질문: 어떤 복제 접근 방식을 선택해야 하나요?

답변: 복제 도구는 두 가지 접근 방식을 제공합니다. 코드 리소스를 단일 CodeCatalyst 소스 리포지토리로 CodeCatalyst 푸시하여 코드 리소스를 AWS Cloud9 ~로 복사하거나 각 하위 폴더를 별개의 CodeCatalyst 소스 리포지토리로 변환할 수 있습니다. 첫 번째 접근 방식은 소스 리포지토리와 같은 CodeCatalyst 개념에 대한 사전 지식이 필요하지 않으므로 사용하는 것이 좋습니다. 이 접근 방식은 기존에 사용하던 것과 비슷한 설정으로 작업하면서 에서의 AWS Cloud9 CodeCatalyst 경험을 탐색하기에 좋은 출발점입니다. AWS Cloud9

두 번째 옵션은 루트 AWS Cloud9 환경 폴더 아래에 있는 하위 폴더를 독립적으로 사용하는 경우에 가장 적합합니다. 이 방법을 사용하면 루트 폴더 아래의 모든 파일이 복제되지 않습니다. 의 소스 리포지토리에 대한 자세한 내용은 Amazon 사용 설명서의 소스 리포지토리를 참조하십시오. CodeCatalyst CodeCatalyst CodeCatalyst

질문: 복제 프로세스에서 생성되는 개인용 액세스 토큰은 무엇이며 왜 필요한가요? 분실한 경우 다시 생성할 수 있나요?

답변: 개인 액세스 토큰은 의 사용자 ID와 연결되어 있습니다. CodeCatalyst git을 사용하여 로컬 변경 내용을 CodeCatalyst 소스 리포지토리로 푸시할 때 암호로 필요합니다. 토큰 및 토큰 생성 방법에 대한 자세한 내용은 Amazon CodeCatalyst 사용 설명서의 CodeCatalyst Amazon에서 개인 액세스 토큰 관리를 참조하십시오.

질문: 복제 프로세스 중에 오류가 발생하면 어떻게 되나요?

답변: 복제 도구를 사용할 때 오류가 발생하면 먼저 도구를 다시 시도해야 합니다. 소스 리포지토리와 관련된 오류인 경우, 코드 리소스를 복제한 후 CodeCatalyst 소스 리포지토리에 수동으로 푸시할 수 있습니다. 로컬 리포지토리가 이미 업스트림과 함께 작동하도록 구성되어 있으므로 제대로 작동할 것입니다. CodeCatalyst 문제가 지속되면 지원 사례를 생성하여 제출하세요. 지원 사례 생성에 대한 자세한 내용은 지원 사례 및 사례 관리 생성을 참조하세요.

질문: BuilderID를 사용하여 복제 도구를 인증하고 권한을 부여해야 하는 이유는 무엇입니까? AWS

답변: 복제 프로세스 중에 복제 도구는 사용자를 대신하여 여러 리소스 (프로젝트, 개발 환경, 소스 리포지토리) 를 읽고 쓰고 로컬 콘텐츠를 복사해야 하므로 이 작업을 수행하려면 사용자의 승인이 필요합니다. CodeCatalyst

질문: 다음으로 옮기면 지연 시간이 달라지나요? CodeCatalyst

답변: 수행 중인 작업에 따라 지연 시간이 줄어들 수 있습니다. 이는 CodeCatalyst 서버가 PDX 지역에서 호스팅되기 때문입니다.

질문: 설치된 소프트웨어가 모두 이전되나요?

답변: 아니요. 코드 리소스만 전송됩니다. 바이너리, 구성 및 설치된 소프트웨어는 이전되지 않습니다.

아마존의 개발 환경 CodeCatalyst

다음 섹션에서는 IDE를 CodeCatalyst 사용하여 개발 환경을 만들고 관리하는 방법을 간략하게 설명합니다. AWS Cloud9

- Dev Environment 생성
- Dev Environment 설정 열기
- Dev Environment 재개
- Dev Environment 삭제
- Dev Environment용 리포지토리 devfile 편집
- 리포지토리 복제
- Dev Environment 문제 해결

Dev Environment 생성

Dev Environment는 다양한 방법으로 만들 수 있습니다.

- 요약, 개발 환경 또는 CodeCatalyst 소스 리포지토리 페이지의 소스 리포지토리를 CodeCatalyst 사용하여 개발 환경을 만드세요.
- Dev Environments의 소스 리포지토리에 연결되지 않은 빈 개발 환경을 만드세요. CodeCatalyst
- 선택한 IDE에 개발 환경을 만들고 CodeCatalyst 소스 리포지토리를 개발 환경에 복제하세요.

각 브랜치 및 리포지토리별로 Dev Environment 하나를 만들 수 있습니다. 프로젝트에는 여러 리포지토리가 존재할 수 있습니다. 개발 환경은 계정에만 연결되며 CodeCatalyst 계정으로만 관리할 수 있습니다. CodeCatalyst Dev Environment를 열고 지원되는 모든 IDE를 이용해 작업할 수 있습니다. 특정 IDE를 선택한 후에는 선택한 IDE로만 해당 개발 환경을 열 수 있습니다. 다른 IDE를 사용하려면 탐색 표시줄에서 개발 환경을 선택하고 편집을 선택하거나 새 개발 환경을 만들어 IDE를 변경할 수 있습니다. 기본적으로 개발 환경은 2코어 프로세서, 4GB RAM 및 16GB 영구 스토리지로 생성합니다.

에서 CodeCatalyst 개발 환경을 만드는 방법에 대한 자세한 내용은 Amazon CodeCatalyst 가이드의 개발 환경 생성을 참조하십시오.

에서 CodeCatalyst 개발 환경을 생성하는 방법에 대한 자세한 내용 및 단계는 Amazon CodeCatalyst 사용 설명서의 개발 환경 생성을 참조하십시오.

> **Note**
>
> 이제 타사 소스 리포지토리를 사용하여 개발 환경을 만들 수 있습니다. 타사 소스 리포지토리를 프로젝트에 연결하는 방법에 대한 자세한 내용은 Amazon CodeCatalyst User Guide의 소스 리포지토리 연결을 참조하십시오. CodeCatalyst

Dev Environment 설정 열기

CodeCatalyst 콘솔에서 개발 환경을 생성한 후 특정 개발 환경 설정을 볼 수 있습니다.

1. CodeCatalyst 콘솔에서 IDE를 통해 개발 환경으로 이동합니다. AWS Cloud9
2. AWS Cloud9 사이드바에서 aws-explorer를 선택합니다.
3. 개발자 도구 탐색 창에서 [설정 열기] 를 CodeCatalyst펼치고 [설정 열기] 를 선택하여 [개발 환경 설정] 보기를 엽니다.
4. Dev Environment Settings(개발 환경 설정) 보기의 다음 섹션에는 개발 환경 관련 옵션이 포함되어 있습니다.

   - 별칭: 개발 환경에 할당된 별칭을 보고 변경합니다.
   - 상태: 현재 개발 환경 상태와 개발 환경에 할당된 프로젝트를 확인하고 개발 환경을 중지합니다.
   - Devfile: 개발 환경용 Devfile의 이름과 위치를 봅니다. 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 할당한 스토리지 양을 변경할 수 없습니다.

> **Note**
>
> CodeCatalyst AWS CLI 터미널에서 Amazon을 사용하는 경우 명령을 실행하기 전에 AWS_Profile=CodeCatalyst를 설정했는지 확인해야 합니다. CodeCatalyst

Dev Environment 재개

Dev Environment $HOME 디렉터리에 있는 모든 항목은 영구적으로 저장됩니다. 필요하다면 개발 환경에서 작업을 중단한 다음 나중에 작업을 재개할 수 있습니다. 개발 환경이 생성될 때 제한 시간 필드에서 선택한 시간보다 오랫동안 개발 환경이 유휴 상태로 남아 있다고 가정합니다. 이 경우 세션이 자동으로 중지됩니다.

에서만 개발 환경을 재개할 수 있습니다. CodeCatalyst 개발 환경을 재개하는 방법에 대한 자세한 내용은 Amazon 가이드의 개발 환경 재개를 참조하십시오. CodeCatalyst

> **Note**
>
> 개발 환경을 재개하는 데 몇 분 정도 걸릴 수 있습니다.

Dev Environment 삭제

개발 환경에 저장된 콘텐츠 작업을 마치면 해당 콘텐츠를 삭제할 수 있습니다. 개발 환경을 삭제하기 전에 코드 변경 사항을 커밋하고 원본 소스 리포지토리에 푸시해야 합니다. 개발 환경을 삭제하면 개발 환경에 대한 컴퓨팅 및 스토리지 청구가 종료됩니다.

의 개발 환경 페이지에서만 개발 환경을 삭제할 수 있습니다. CodeCatalyst 개발 환경을 삭제하는 방법에 대한 자세한 내용은 Amazon CodeCatalyst 가이드의 개발 환경 삭제를 참조하십시오.

개발 환경용 리포지토리 devfile 편집

개발 환경의 구성을 변경하려면 devfile을 편집해야 합니다. devfiles을 사용하면 팀 전체의 개발 환경을 표준화할 수 있습니다. 에서 소스 리포지토리의 devfile 루트에서 편집할 수 있습니다. CodeCatalyst 지

원되는 IDE에서 devfile을 편집하는 방법도 있습니다. 지원되는 IDE에서 devfile을 편집하는 경우, 변경 사항을 커밋하고 소스 리포지토리에 푸시하거나 풀 요청을 생성합니다. 이렇게 하면 팀원이 devfile 편집을 검토하고 승인할 수 있습니다.

> **Note**
>
> devfile에는 퍼블릭 컨테이너 이미지만 포함할 수 있습니다.

> **Note**
>
> 종속성이 없는 경우 일부 AWS Cloud9 IDE 기능이 사용자 devfile 지정에서 작동하지 않을 수 있습니다. Linux x64가 아닌 일부 플랫폼에서 작동하게 하려면 추가 작업이 필요할 수 있습니다.

에서 개발 환경의 리포지토리를 devfile 편집하려면 AWS Cloud9

1. CodeCatalyst 콘솔에서 IDE를 통해 개발 환경으로 이동합니다. AWS Cloud9
2. AWS Cloud9 사이드바에서 aws-explorer를 선택합니다.
3. 개발자 도구 탐색 창에서 툴킷 메뉴를 선택합니다. CodeCatalyst
4. Open Devfile(Devfile 열기)을 선택합니다.
5. devfile을 편집하고 파일을 저장합니다.
6. 메뉴 사이드바에서 Git 확장자인 소스 제어를 선택합니다.
7. Message(메시지) 텍스트 필드에 스테이징 변경 전 메시지를 입력합니다.
8. 커밋을 준비하려면 Stage All Changes (+)(모든 변경 사항 준비 (+)) 아이콘을 선택합니다.
9. Git 명령을 보려면 리포지토리 이름 옆에 있는 메뉴 아이콘을 선택합니다.
10. Commit(커밋)과 Push(푸시)를 선택합니다.
11. AWS Toolkit 메뉴에서 개발 환경 업데이트를 선택합니다.

    Commit(커밋)과 Push(푸시)를 선택합니다. 업데이트된 devfile이 저장되었고 변경 사항이 커밋 및 푸시되었습니다.

> **Note**
>
> 사용자 지정 devfile을 사용하여 시작하려는 개발 환경이 작동하지 않는다고 가정해 보겠습니다. 이는 devfile이 AWS Cloud9과 호환되지 않기 때문일 수 있습니다. 문제를 해결하려면 devfile을 검토하세요. 문제가 해결되지 않는다면 삭제한 다음 새로 만들어 보세요.

를 통해 개발 devfile 환경용으로 편집할 수도 있습니다. CodeCatalyst 자세한 내용은 Amazon CodeCatalyst 가이드의 개발 환경 구성을 참조하십시오.

리포지토리 복제

소스 리포지토리의 여러 파일, 브랜치 및 커밋으로 효과적으로 작업하려면 소스 리포지토리를 로컬 컴퓨터에 복제하면 됩니다. 그런 다음 Git 클라이언트나 IDE를 사용하여 변경합니다. CodeCatalyst부터는 다른 Git 호스트 공급자와 동일한 방식으로 그리고 명령줄을 사용하여 AWS Cloud9 IDE Git 확장을 사용할 수 있습니다. 서드 파티 리포지토리를 복제하는 방법은 Git 리포지토리 초기화 또는 복제를 참조하세요.

소스 리포지토리에서 개발 환경을 생성하고 이를 사용하여 CodeCatalyst 복제하는 방법에 대한 자세한 내용은 Amazon CodeCatalyst 가이드의 소스 리포지토리 개념을 참조하십시오.

Dev Environment 문제 해결

개발 환경에 문제가 발생하는 경우 Amazon CodeCatalyst 가이드의 개발 환경 문제 해결을 참조하십시오.

> **Note**
>
> CodeCatalyst AWS CLI 터미널에서 Amazon을 사용하는 경우 명령을 실행하기 전에 AWS_Profile=CodeCatalyst를 설정했는지 확인해야 합니다. CodeCatalyst

개발 환경에 문제가 발생하는 경우 Amazon CodeCatalyst 가이드의 개발 환경 문제 해결을 참조하십시오.

AWS Cloud9 통합 개발 환경(IDE)의 AWS CDK 작업

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 안내서에서 확인할 수 있습니다.

AWS CDK 앱은 구성체라는 구성 요소로 이루어집니다. 이러한 구성 요소에는 AWS CloudFormation 스택과 스택 내 AWS 리소스에 대한 정의가 포함됩니다. AWS CDK Explorer를 사용하면 AWS CDK 트리 뷰에서 정의된 스택 및 리소스를 볼 수 있습니다. AWS Cloud9 편집기 내의 Developer Tools(개발자 도구) 창에서 이 뷰에 액세스할 수 있습니다.

이 섹션에서는 AWS Cloud9 편집기에서 AWS CDK를 액세스하고 사용하는 방법에 대한 정보를 제공합니다.

AWS CDK 애플리케이션 작업

AWS Cloud9 통합 개발 환경(IDE)에서 AWS CDK Explorer를 사용하여 AWS CDK 애플리케이션을 시각화하고 관련 작업을 수행합니다.

필수 조건

AWS CDK 명령줄 인터페이스를 설치합니다. 지침을 보려면 AWS Cloud Development Kit (AWS CDK) 개발자 안내서의 AWS CDK 시작하기를 참조하세요.

> **Important**
>
> 설치된 AWS CDK 버전은 1.17.0 이상이어야 합니다. **`cdk --version`** 명령을 사용하여 실행 중인 버전을 확인할 수 있습니다.

AWS CDK 애플리케이션 시각화

AWS Cloud9 IDE AWS CDK Explorer를 사용하면 앱의 CDK 구조에 저장된 스택 및 리소스를 관리할 수 있습니다. AWS CDK Explorer는 `tree.json` 파일에 정의된 정보를 사용하여 리소스를 트리 뷰로 표시합니다. **`cdk synth`** 명령을 실행하면 이 파일이 생성됩니다. 기본적으로 앱의 `cdk.out` 디렉터리에 `tree.json` 파일이 위치합니다.

Toolkit AWS CDK Explorer 사용을 시작하려면 CDK 애플리케이션을 생성합니다.

1. AWS CDK 개발자 안내서에 있는 Hello World 자습서의 처음 몇 단계를 완료합니다.

   > **Important**
   >
   > 스택 배포 단계에 도달하면 중지하고 이 가이드로 돌아옵니다.

   > **Note**
   >
   > 자습서에 제공된 명령(예: **`mkdir`** 및 **`cdk init`**)을 운영 체제 명령줄 인터페이스나 VS 코드 편집기 내의 Terminal(터미널) 창에서 실행할 수 있습니다.

2. CDK 자습서의 필수 단계를 완료한 후 AWS Cloud9 IDE 편집기에서 생성한 CDK 콘텐츠를 엽니다.

3. AWS 탐색 창에서 CDK 제목을 확장합니다. 이제 CDK 애플리케이션 및 관련 리소스가 CDK Explorer 트리 뷰에 표시됩니다. AWS Cloud9 내의 터미널에서 다음 명령을 실행하여 CDK 기능이 작동하는지 확인할 수도 있습니다.

   ```
   mkdir mycdkapp
   cd mycdkapp
   cdk init app --language=typescript
   cdk synth
   cdk bootstrap
   ```

중요 정보

- CDK 앱을 AWS Cloud9 편집기로 로드하면 한 번에 여러 폴더를 로드할 수 있습니다. 앞의 이미지와 같이 각 폴더에는 여러 개의 CDK 앱이 포함될 수 있습니다. AWS CDK Explorer는 프로젝트 루트 디렉터리와 이 디렉터리의 직접적인 하위 디렉터리에서 앱을 찾습니다.

- 이 자습서의 처음 몇 단계를 수행할 때 마지막으로 실행한 명령이 **`cdk synth`**이라는 것을 알 수 있습니다. 이 명령은 AWS CDK 앱을 CFN으로 변환하여 CloudFormation 템플릿을 합성합니다. 부산물로 `tree.json` 파일도 생성합니다. CDK 앱을 변경한 경우 **`cdk synth`** 명령을 다시 실행하여 트리 뷰에 반영된 변경 사항을 확인합니다. 변경 사항의 한 가지 예는 앱에 더 많은 리소스를 추가하는 것입니다.

AWS CDK 앱에서 기타 작업 수행

명령줄 인터페이스를 사용하는 것과 동일한 방식으로 AWS Cloud9 편집기를 사용하여 CDK 앱에서 다른 작업을 수행할 수 있습니다. 예를 들어, 편집기에서 코드 파일을 업데이트하고 AWS Cloud9 Terminal(터미널) 창을 사용하여 앱을 배포할 수 있습니다.

이러한 유형의 작업을 시험해 보려면 AWS Cloud9 편집기를 사용하여 AWS CDK개발자 안내서의 Hello World 자습서를 계속 진행합니다. 마지막 단계인 앱 리소스 삭제를 수행해야 합니다. 그렇지 않으면 AWS 계정에 예상치 못한 비용이 발생할 수 있습니다.

Git 패널로 시각적 소스 제어

AWS Cloud9용 Git 패널은 필수적인 Git 기능을 사용하기 위한 편리한 시각적 인터페이스를 제공합니다.

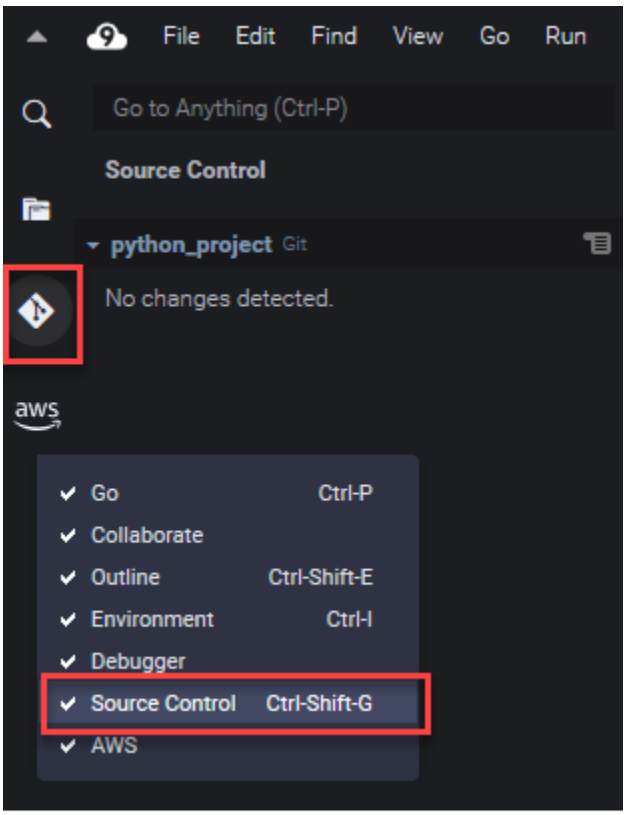
Git 패널 인터페이스의 옵션을 사용하여 리포지토리 초기화하거나 원격 리포지토리를 복제하고, 준비 영역에 파일을 추가하고, 스테이징된 파일을 작업 디렉터리에 커밋한 다음 변경 사항을 업스트림 리포지토리로 푸시하는 등, 전체 소스 제어 수명 주기를 관리할 수 있습니다.

Git 패널 인터페이스에서 몇 번의 클릭만으로 브랜치 생성 및 병합과 같은 Git의 핵심 협업 및 프로젝트 관리 기능을 신속하게 구현할 수 있습니다. 또한 IDE의 편집기 창을 사용하여 병합 충돌을 식별하고 해결할 수 있습니다.

> **Important**
>
> Git 패널은 Amazon EC2 인스턴스로 생성된 AWS Cloud9 환경에서만 사용할 수 있습니다. EC2 환경 대신 SSH 개발 환경을 사용하는 경우 이 기능에 액세스할 수 없습니다.
> 또한 Git 패널은 기본적으로 2020년 12월 11일 이후에 생성된 새로운 AWS Cloud9 환경에서 사용할 수 있습니다. 현재, 이 날짜 이전에 생성된 개발 환경에서 Git 패널을 지원하기 위해 노력하고 있습니다.

인터페이스에 액세스하고 상호 작용하려면 [창(Window)], [소스 제어(Source Control)]를 선택합니다. 또는 IDE의 측면 패널 아무 곳이나 마우스 오른쪽 버튼을 클릭하고 Source Control(소스 제어)을 선택하여 소스 제어로 이동할 수도 있습니다. 그런 다음 IDE 인터페이스에 표시되는 Git 아이콘을 선택합니다.

Ctrl-Shift-G 키 조합을 사용하여 Git 패널의 디스플레이를 토글할 수도 있습니다.



> **Note**
>
> 제트 다크(Jett Dark) 테마가 적용된 AWS Cloud9 IDE를 보여 주는 Git 패널 문서의 스크린샷 다른 테마로 IDE를 사용하는 경우 일부 인터페이스 요소가 다르게 표시됩니다. Git 아이콘 대신 [소스 제어(Source Control)]라는 레이블이 있는 링크를 선택하여 Git 패널을 열 수 있습니다.

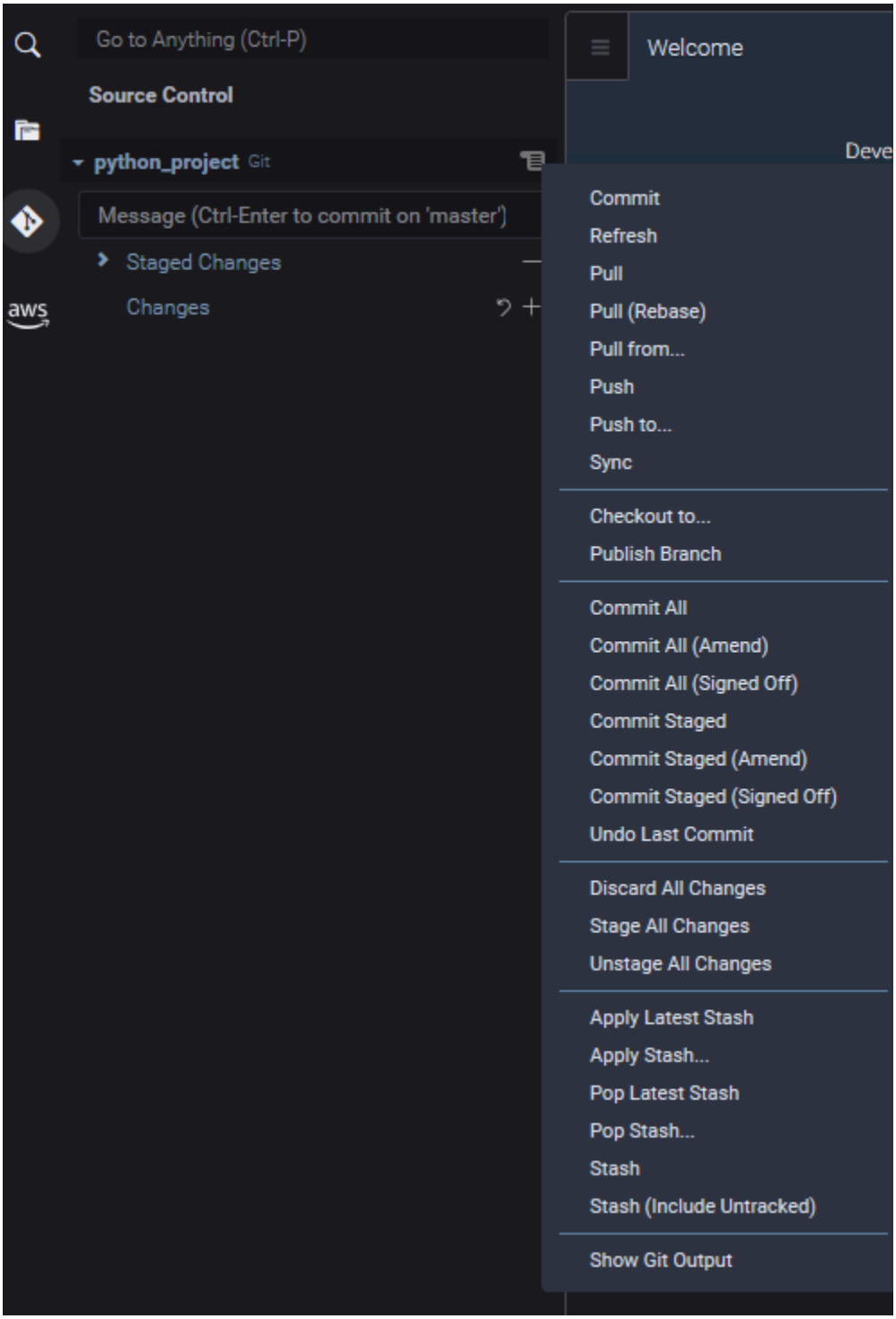
주제

- Git 패널로 소스 제어 관리
- 참조: Git 패널에서 사용할 수 있는 Git 명령

Git 패널로 소스 제어 관리

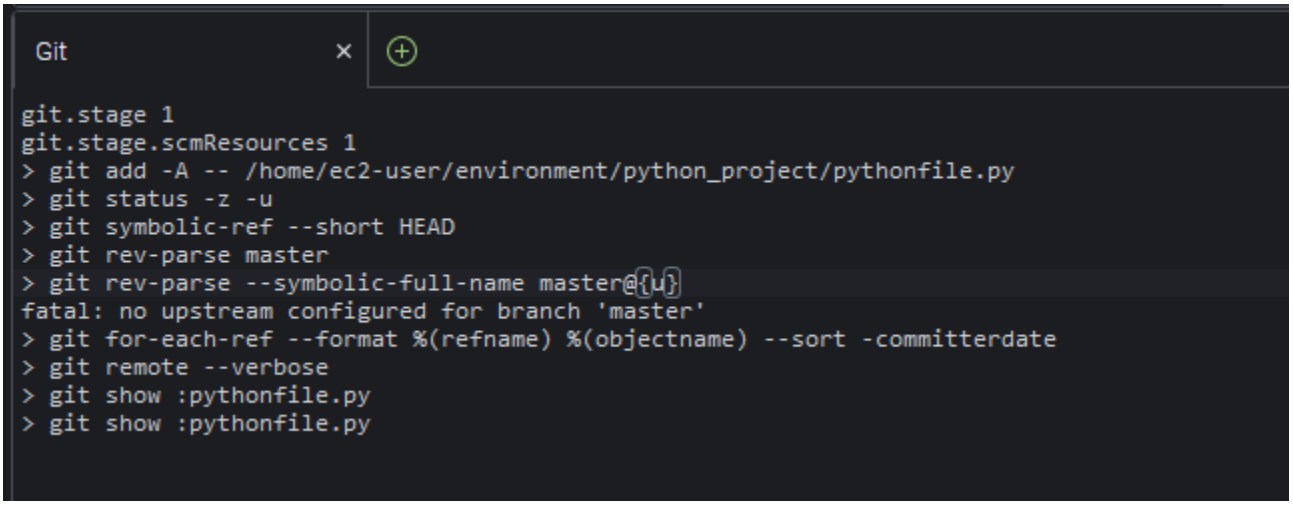
AWS Cloud9의 Git 패널 확장은 코어 및 고급 Git 명령에 대한 편리한 사용자 인터페이스 액세스를 제공합니다.

이 섹션에서는 소스 제어를 관리하기 위한 주요 Git 기능에 액세스하는 방법을 보여 줍니다. 이 절차에서는 Git 패널 메뉴를 사용하여 리포지토리와 그 콘텐츠에 대해 Git 명령을 실행하는 데 중점을 둡니다.



Git 패널 검색 상자에 이름을 입력하기 시작하여 지원되는 모든 Git 명령에 액세스 할 수도 있습니다.

또한 Git 패널 인터페이스와 상호 작용할 때 실행되는 실제 Git 명령을 볼 수 있습니다. 명령줄 작업을 보려면 Git 패널 메뉴로 이동하여 [Git 출력 표시(Show Git Output)]를 선택합니다.

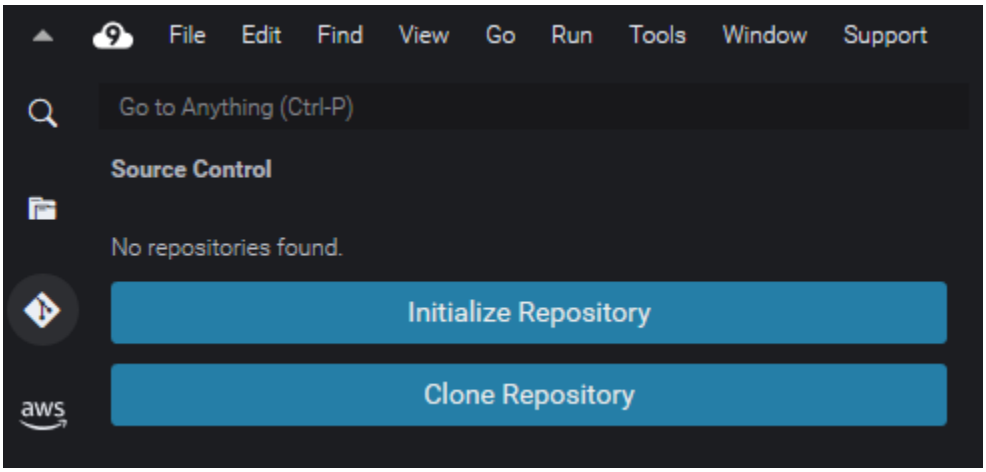


Git 리포지토리 초기화 또는 복제

Git 리포지토리('repo')는 처음부터 프로젝트의 전체 기록을 포함합니다. 리포지토리는 스테이징된 파일을 repo로 커밋할 때마다 캡처된 프로젝트 콘텐츠의 모든 스냅샷으로 구성됩니다.

Git 패널은 Git 리포지토리를 얻는 두 가지 방법을 지원합니다.

- 기존 디렉터리를 Git 리포지토리로 초기화합니다.
- 기존 리포지토리를 복제하고 로컬 디렉터리에 복사합니다.



> **Note**
>
> 리포지토리를 초기화하거나 복제하기 위한 인터페이스 옵션은 환경의 WorkSpace 폴더에 Git 리포지토리가 아직 추가되지 않은 경우에만 사용할 수 있습니다. 리포지토리에 대한 작업 디렉터리가 이미 있는 경우 Git 패널 창에 작업 디렉터리 및 준비 영역의 상태가 표시됩니다. Git 패널 메뉴를 사용하여 리포지토리에 대해 실행할 수 있는 Git 명령에 액세스할 수도 있습니다.

리포지토리를 초기화거나 복제하려면

1. Git 패널을 아직 사용할 수 없는 경우 [Window], [소스 제어(Source Control)]를 선택한 다음 Git 아이콘을 선택하여 액세스합니다.

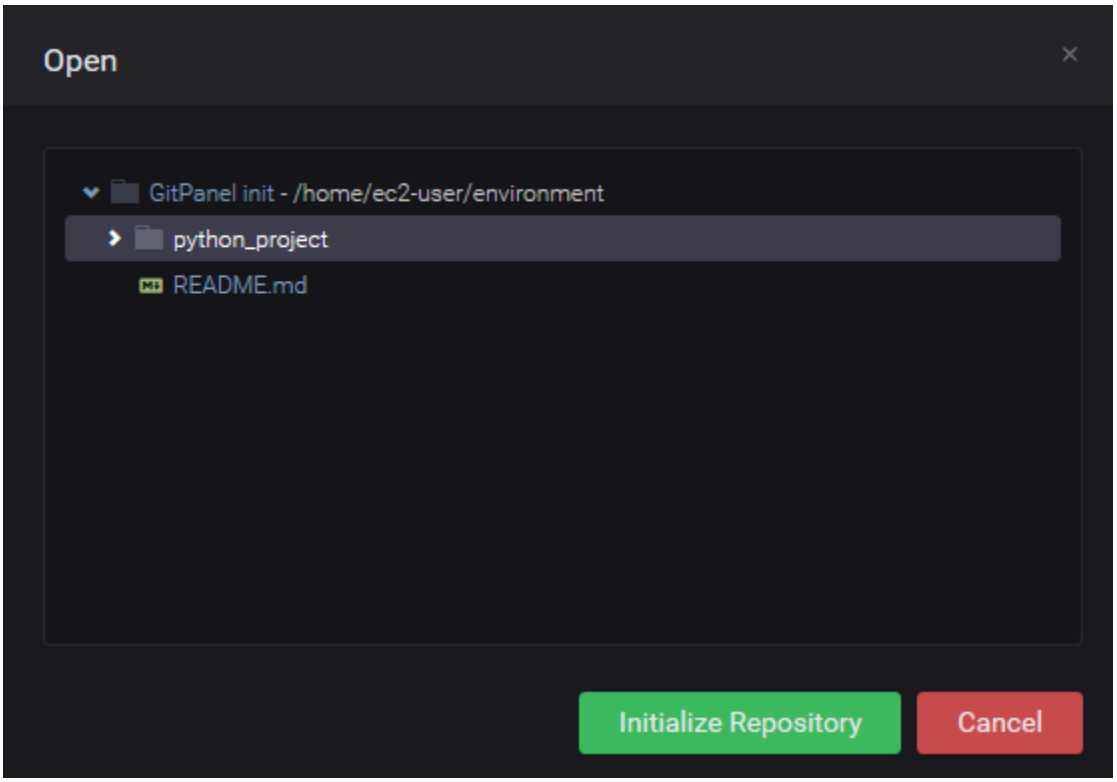
   > **Note**
   >
   > Ctrl+Shift+G라는 키보드 바로 가기를 사용하여 Git 패널을 열 수도 있습니다.

2. 새 리포지토리를 초기화할지 아니면 기존 리포지토리를 복제할지를 선택합니다.

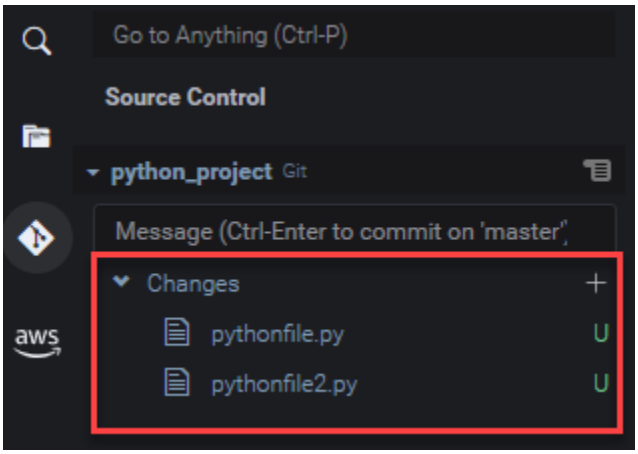
   Initialize a repository

   - Git 패널에서 [리포지토리 초기화(Initialize Repository)]를 선택합니다.

- 그런 다음 Git 리포지토리가 초기화될 WorkSpace 폴더를 선택합니다. 폴더 경로를 입력하거나, 경로를 선택하거나, 대화 상자에서 폴더를 선택할 수 있습니다.
- 대화 상자를 사용하는 경우 대상 폴더를 선택하고 [리포지토리 초기화(Initialize Repository)]를 선택합니다.



선택한 폴더에서 Git 리포지토리를 초기화하고 나면, Git 패널에 해당 폴더에 있는 모든 파일이 추적되지 않는 상태이며 Git 준비 영역에 추가할 준비가 된 상태로 표시됩니다.



Clone a repository

- Git 패널 창에서 [리포지토리 복제(Clone Repository)]를 선택합니다.
- 다음으로 복제할 원격 리포지토리의 URL을 입력하고(예: GitHub에서 호스팅되는 리포지토리를 복제하려면 `https://github.com/my-own-repo/my-repo-project-name.git` 입력) Return 키를 누릅니다.
- 표시되는 대화 상자에서, 복제된 리포지토리의 WorkSpace 폴더를 선택하고 [리포지토리 위치 선택(Select Repository Location)]을 선택합니다.

> **Note**
>
> 외부 사이트(예: GitHub)에서 호스팅되는 리포지토리에 액세스하는 경우 프로세스를 완료하려면 해당 사이트의 로그인 보안 인증 정보도 입력해야 합니다.

선택한 폴더에서 원격 리포지토리를 복제한 후에는 `git pull` 명령을 실행하여 로컬 리포지토리를 원격 리포지토리의 최신 변경 사항과 동기화할 수 있습니다. 자세한 내용은 원격 리포지토리 작업 섹션을 참조하세요.

파일 스테이징 및 커밋

Git 리포지토리를 얻은 후에는 다음 두 단계 프로세스를 사용하여 콘텐츠로 채울 수 있습니다.

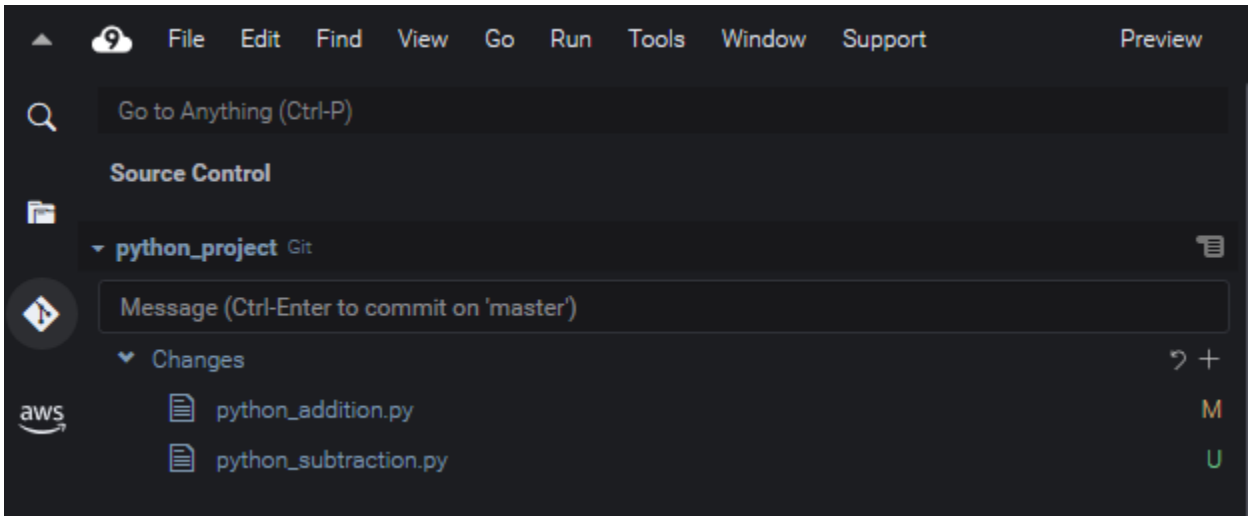
1. 추적되지 않거나 최근에 수정한 콘텐츠를 준비 영역에 추가합니다.
2. 준비 영역의 파일을 작업 디렉터리에 커밋합니다.

> **Important**
>
> 작업 디렉터리의 모든 파일을 리포지토리에 커밋하지는 않으려는 경우도 있습니다. 예를 들어 런타임 중에 생성된 파일을 프로젝트의 리포지토리에 추가해서는 안 될 것입니다. Git 패널을 사용하면 파일을 `.gitignore` 파일의 목록에 추가하여 무시하도록 수 표시할 수 있습니다. `.gitignore`의 목록을 업데이트하려면 준비 영역에 추가되지 않은 파일을 마우스 오른쪽 버튼으로 클릭하고 [.gitignore에 파일 추가(Add File to .gitignore)]를 선택합니다. IDE가 `.gitignore` 파일을 열고 선택한 파일의 이름이 무시된 파일 목록에 추가됩니다.

`.gitignore`에서 패턴 매칭을 사용하여 파일 형식을 제외하는 방법에 대한 자세한 내용은 관련 git-scm.com 사이트를 참조하세요.

Stage files

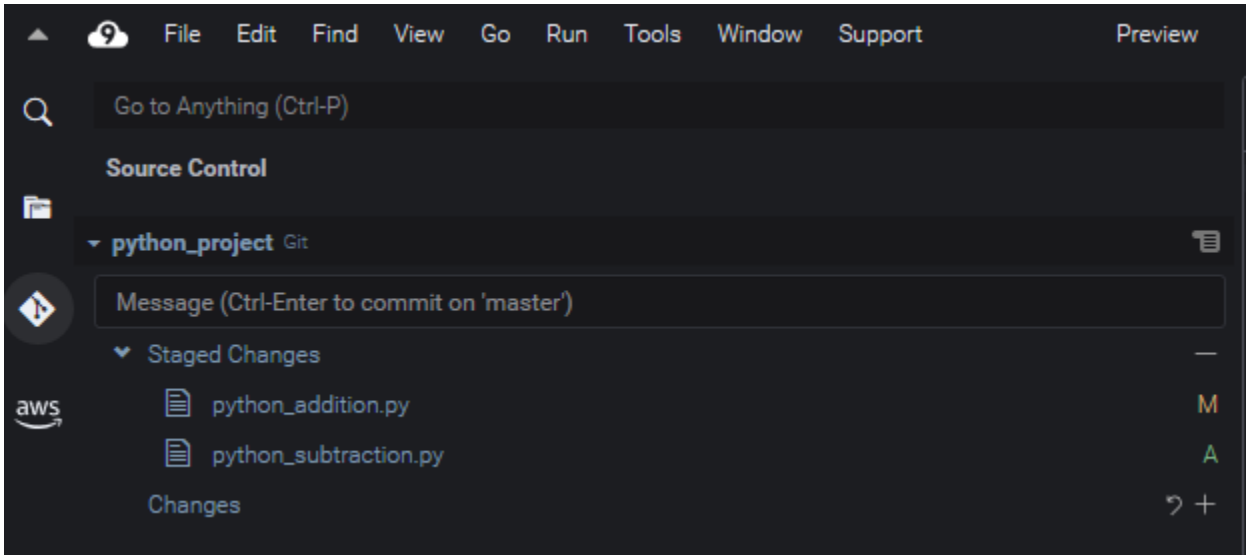
준비 영역에 추가되지 않는 파일(레이블이 'U'인 파일)과 수정된 파일(레이블이 'M'인 파일)은 Git 패널 창의 Changes(변경 사항) 아래에 나열됩니다.



Git 패널 인터페이스를 사용하여 특정 파일 또는 추적되지 않고 수정된 모든 파일을 준비 영역에 추가할 수 있습니다.

- 특정 파일: 파일을 일시 중지한 다음 +를 선택하여 준비 영역에 추가합니다. 또는 파일을 마우스 오른쪽 버튼으로 클릭하고 [변경 사항 스테이징(Stage Changes)]을 선택합니다.
- 모든 파일: Git 패널 메뉴를 클릭하고 [모든 변경 사항 스테이징(Stage All Changes)]을 선택합니다.

리포지토리의 인덱스에 추가된 파일은 [스테이징된 변경 사항(Staged Changes)]에 나열됩니다. 이전에 추적되지 않은 파일에는 'A'라는 레이블이 표시되어 스테이징되었음을 나타냅니다.



> **Note**
>
> 특정 변경 사항 또는 모든 변경 사항을 언스테이징할 수도 있습니다. 단일 파일의 경우 파일을 일시 중지한 다음 -를 선택합니다. 또는 파일을 마우스 오른쪽 버튼으로 클릭하고 [변경 사항 언스테이징(Unstage Changes)]을 선택합니다. 모든 변경 내용을 언스테이징하려면 Git 패널 메뉴로 이동하여 [모든 변경 사항 언스테이징(Unstage All Changes)]을 선택합니다.

Commit files

Git의 `commit` 명령을 사용하여 스테이징된 파일을 리포지토리의 영구 스냅샷으로 캡처할 수 있습니다. Git 패널 인터페이스를 사용하여 커밋할 파일을 선택할 수 있습니다.

- 준비 영역의 파일 커밋: Git 패널 메뉴로 이동하여 [커밋(Commit)] 또는 [스테이징된 항목 커밋(Commit Staged)]을 선택합니다.
- 작업 디렉터리의 파일 모두 커밋: Git 패널 메뉴로 이동하여 [모두 커밋(Commit All)]을 선택합니다. (이 옵션은 `git commit`을 호출하기 전에 `git add`를 사용하여 준비 영역에 파일을 추가합니다.)

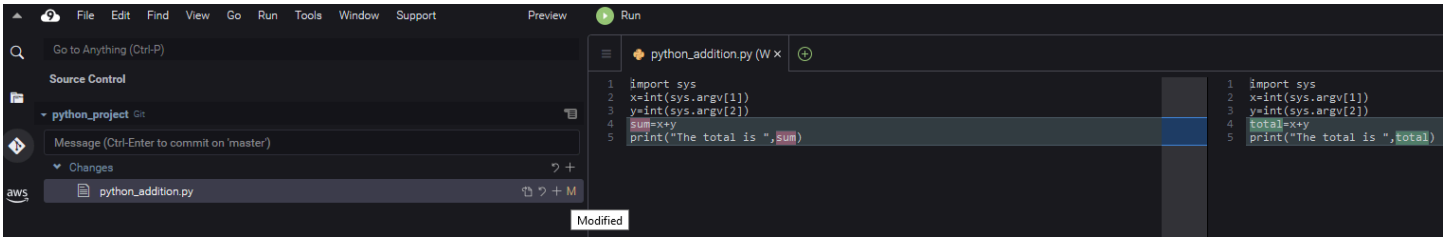
> **Note**
>
> 또한 Git 패널을 사용하여 파일을 커밋할 때 amend 및 signed-off 옵션을 사용할 수 있습니다. amend 옵션은 가장 최근 커밋의 커밋 메시지를 수정합니다. sign-off 옵션은 Git 로그에서 커밋을 수행한 사용자를 식별할 수 있습니다.
> Git 패널 메뉴로 이동하고 [마지막 커밋 실행 취소(Undo Last Commit)]를 선택하여 커밋을 되돌릴 수도 있습니다.

다른 파일 버전 보기

스테이징되거나 커밋된 후에 수정된 파일의 버전을 비교할 수 있습니다.

- [변경 사항(Changes)]에 나열된 파일: 작업 디렉터리에 있는 버전과 마지막으로 스테이징되었거나 리포지토리에 커밋된 버전 간의 차이점을 보려면 'M'을 선택합니다.
- [스테이징된 변경 사항(Staged Changes)]에 나열된 파일: 준비 영역에 있는 버전과 마지막으로 리포지토리에 커밋된 버전 간의 차이점을 보려면 'M'을 선택합니다.

'M'을 선택하면 IDE 창에 두 버전의 파일 간의 차이점이 표시됩니다. 한쪽은 리포지토리에서 현재 버전으로 추적되는 버전을 보여줍니다. 다른 쪽은 아직 커밋되지 않은 수정된 버전을 보여줍니다.



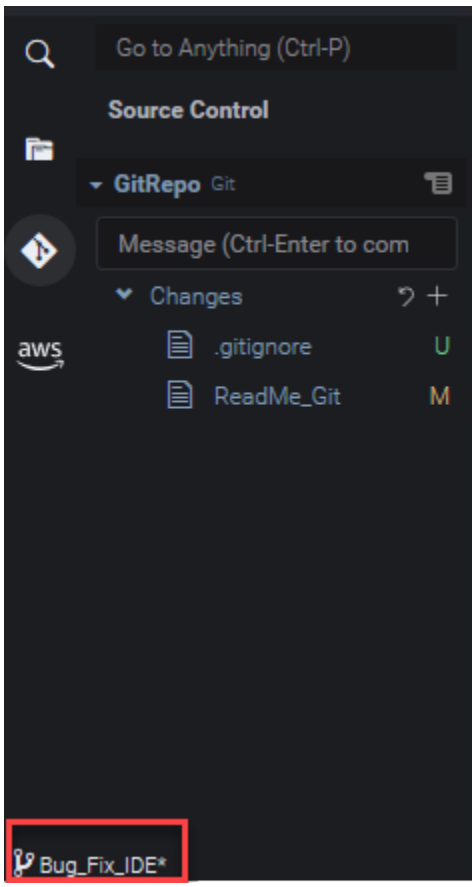
브랜치 작업

Git는 리포지토리의 주요 브랜치와 독립적인 브랜치의 새로운 기능을 사용할 수 있도록 함으로써 워크플로 관리를 훨씬 용이하게 합니다. 여러 브랜치 간에 원활하게 전환하면서 기본 브랜치에 언제든 빌드할 소스 코드가 준비되어 있도록 합니다.

브랜치 생성

브랜치를 생성하려면 브랜치의 이름을 지정하고 시작점을 선택해야 합니다.

1. Git 패널 메뉴에서 [다음으로 체크아웃(Checkout to)]을 선택합니다. 또는 Git 패널의 하단에 표시되는 현재 브랜치의 이름을 선택할 수 있습니다.



2. 새 브랜치를 생성하는 옵션을 선택합니다.
   - 새 브랜치 생성: 현재 브랜치의 마지막 커밋에서 새 브랜치가 시작됩니다.
   - 다음에서 새 브랜치 생성: 다음 화면에서 선택하는 브랜치의 마지막 커밋에서 새 브랜치가 시작됩니다.
3. 새 브랜치의 이름을 입력합니다.
4. 특정 브랜치를 브랜치의 시작점으로 지정하는 경우 목록에서 하나를 선택합니다.

새 브랜치로 전환한 후 Git 패널의 하단에서 현재 브랜치의 이름을 확인할 수 있습니다.

> **Note**
>
> 원격 리포지토리로 작업하는 경우 업스트림 원격 리포지토리에 새 브랜치를 게시하여 다른 사용자가 콘텐츠에 액세스할 수 있도록 합니다.

브랜치 전환

Git을 사용하여 소스 제어를 관리하는 데 따른 주요 이점 중 하나는 브랜치를 전환하여 여러 프로젝트 간을 간단히 이동할 수 있다는 것입니다.

> **Important**
>
> 현재 브랜치에 리포지토리에 커밋되지 않은 파일이 있는 경우 브랜치를 전환할 수 없습니다. 먼저 작업 내용을 커밋 또는 stash하여 작업 디렉터리를 정리해야합니다.

1. Git 패널 하단에서 현재 브랜치의 이름을 선택합니다. 또는 Git 패널로 이동하여 [다음으로 체크아웃(Checkout to)]을 선택합니다.
2. 표시된 목록에서 브랜치를 선택합니다.

전환하고 나면, 리포지토리의 작업 디렉터리가 선택한 브랜치에 가장 최근에 커밋된 파일 버전으로 업데이트됩니다.

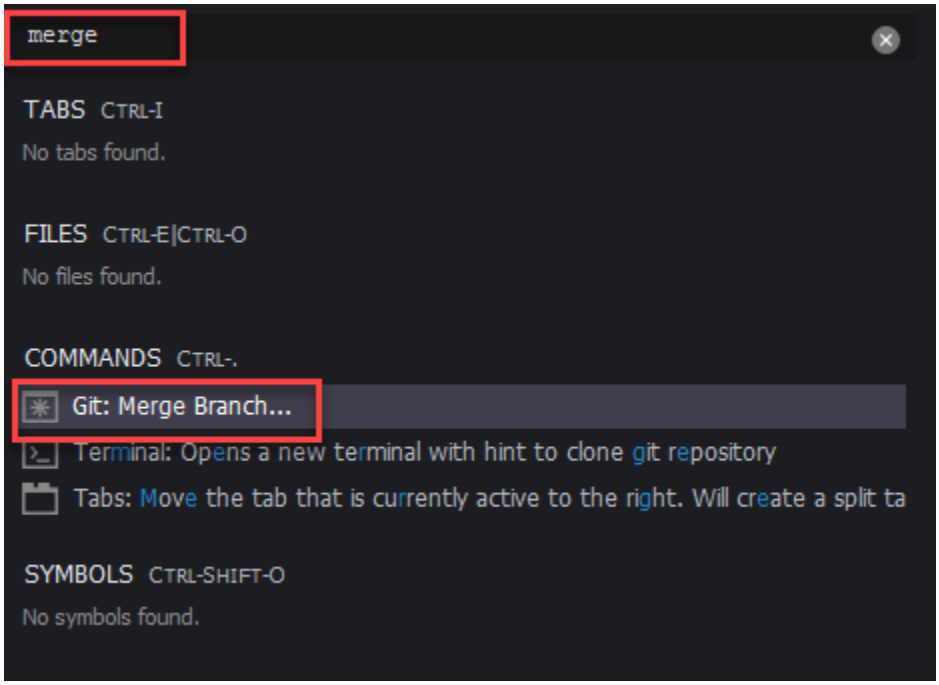
브랜치 병합

개별 브랜치의 기능 작업을 마친 후에는 일반적으로 변경 내용을 주 프로젝트에 통합해야 합니다. Git를 사용하면 특정 브랜치(예: 기능 브랜치)를 다른 브랜치(일반적으로 리포지토리의 주 또는 기본 브랜치)로 병합하여 이러한 종류의 통합을 쉽게 수행할 수 있습니다.

1. 다른 브랜치를 병합할 브랜치를 선택하려면 Git 패널 메뉴로 이동한 후 [다음으로 체크아웃(Checkout to)]을 선택합니다.

   또는 Git 패널의 하단ㄴ에서 현재 브랜치의 이름을 선택하합니다.
2. 표시된 목록에서 전환할 브랜치를 선택합니다.
3. [검색(Search)] 상자에 ‘merge’라는 단어를 입력하기 시작합니다.

   [명령(Commands)] 목록에 Git: Merge Branch가 표시되면 해당 명령을 선택합니다.

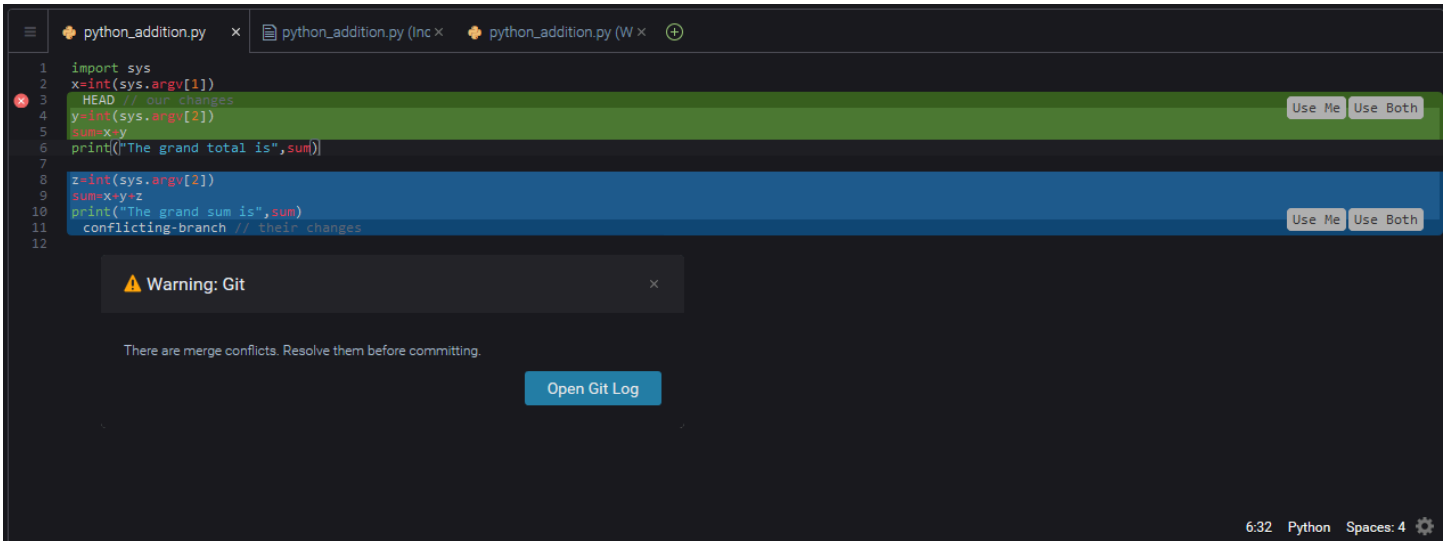


4. 표시된 목록에서 대상 브랜치로 병합할 브랜치를 선택합니다.

   병합이 충돌 없이 완료되면 Git 패널 인터페이스가 새로 고쳐져 병합된 변경 사항이 포함된 대상 브랜치가 표시됩니다.

브랜치를 병합하는 경우 동일한 콘텐츠에 대한 서로 호환되지 않는 변경 사항으로 인해 병합 충돌이 발생할 수 있습니다. 이 경우 병합을 커밋하기 전에 충돌을 해결해야 한다는 경고가 표시됩니다.

IDE의 코드 편집기 창을 사용하여 두 브랜치에서 충돌하는 콘텐츠를 식별한 다음 변경하여 차이를 해결할 수 있습니다.



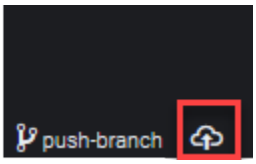
원격 리포지토리 작업

인터넷이나 네트워크에서 호스팅되는 원격 리포지토리는 팀원이 로컬 리포지토리에 커밋한 변경 사항을 공유할 수 있도록 허용하여 협업을 용이하게 합니다. 데이터를 업로드하고 다운로드하는 Git 명령을 사용하면 '다운스트림'(로컬) 리포지토리의 콘텐츠가 '업스트림'(원격) 리포지토리의 콘텐츠와 동기화되도록 할 수 있습니다.

원격 리포지토리에 브랜치 게시

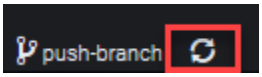
로컬 리포지토리의 브랜치를 생성하고 나면, 해당 브랜치는 사용자에게 프라이빗이며 원격 리포지토리에 '업스트림'으로 푸시할 때까지 공동 작업자가 사용할 수 없습니다.

1. 현재 브랜치를 게시하려면 Git 패널 메뉴로 이동하여 [브랜치 게시(Publish Branch)]를 선택합니다. 또는 Git 패널 하단의 브랜치 이름 옆에 있는 클라우드 기호를 클릭합니다.

2. 필요하다면 원격 리포지토리에 액세스할 수 있는 로그인 보안 인증 정보를 입력합니다.

브랜치가 원격 리포지토리에 성공적으로 게시되면 Git 패널 하단의 브랜치 이름 옆에 동기화 기호가 표시됩니다. 로컬 및 원격 리포지토리의 컨텐츠를 동기화하려면 이 기호를 선택합니다.



로컬 리포지토리와 원격 리포지토리 간에 컨텐츠 푸시 및 풀

Git을 사용하여 공유 프로젝트에서 협업하는 경우 일반적으로 다른 팀원의 최근 변경 사항을 원격 리포지토리에서 로컬 리포지토리로 풀하는 방식으로 작업을 시작합니다. 그리고 로컬 리포지토리의 변경 사항을 커밋한 후에는 나머지 팀이 액세스할 수 있도록 원격 리포지토리로 푸시합니다. 이러한 작업은 `git pull` 및 `git push` 명령을 사용하여 수행합니다.

> **Note**
>
> 대부분의 호스트된 리포지토리(예: GitHub의 리포지토리)에서 변경 사항을 푸시하고 가져올 때 로그인 보안 인증 정보를 입력해야 합니다.

Pull changes from remote

Git 패널 인터페이스를 통해 `git pull` 명령을 사용하면 원격 리포지토리의 브랜치에 커밋된 최신 변경 사항으로 로컬 리포지토리를 업데이트할 수 있습니다.

1. Git 패널 메뉴에서 [다음으로 체크아웃(Checkout to)]을 선택합니다.
2. 브랜치 목록에서 변경 사항을 가져올 로컬 브랜치를 선택합니다.
3. 그런 다음 Git 패널 메뉴로 이동하여 [다음에서 풀(Pull from)]을 선택합니다.
4. 원격 리포지토리를 선택한 다음 해당 리포지토리의 브랜치를 선택하여 변경 사항을 가져옵니다.

풀 작업을 수행한 후에는 리포지토리 작업 디렉터리의 원격 리포지토리에서 검색한 파일에 액세스할 수 있습니다. 파일을 수정한 후 변경 사항을 원격 브랜치로 푸시할 수 있습니다.

Push changes to remote

Git 패널 인터페이스를 통해 `git push` 명령을 사용하면 로컬 리포지토리의 지정된 브랜치에 적용된 최신 변경 사항으로 원격 리포지토리를 업데이트할 수 있습니다.

1. Git 패널 메뉴에서 [다음으로 체크아웃(Checkout to)]을 선택합니다.
2. 브랜치 목록에서 변경 사항을 적용할 로컬 브랜치를 선택합니다.
3. 그런 다음 Git 패널 메뉴로 이동하여 [다음으로 푸시(Push to)]를 선택합니다.
4. 원격 리포지토리를 선택한 다음 해당 리포지토리의 브랜치를 선택하여 변경 사항을 푸시합니다.

푸시 작업을 수행하고 나면 다른 팀원이 해당 변경 사항을 리포지토리의 로컬 복사본으로 풀하여 액세스할 수 있습니다.

파일 stash 및 검색

Git의 stash 기능을 사용하면 먼저 스테이징 또는 수정된 파일을 커밋하지 않고도 브랜치를 전환할 수 있습니다. stash 기능은 작업 디렉터리 및 준비 영역의 현재 상태를 캡처하고 나중에 사용할 수 있도록 저장합니다. 이 기능은 아직 완성되지 않은 콘텐츠로 작업하고 지체없이 브랜치를 전환해야 할 때 유용합니다.

작업 stash

1. 작업 디렉터리의 현재 상태를 숨기려면 Git 패널 메뉴로 이동하여 다음 옵션 중 하나를 선택합니다.

   - [Stash]: 작업 디렉터리의 수정되거나 스테이징된 파일이 모두 stash에 추가됩니다. 추적되지 않는 파일은 추가되지 않습니다.
   - [Stash(추적되지 않는 항목 포함)(Stash (include Untracked))]: 아직 추적되지 않는 파일을 포함하여 작업 디렉터리의 모든 파일이 stash에 추가됩니다.
2. 나중에 검색할 stash를 식별하는 데 도움이 될 선택적 메시지를 입력합니다.

stash하고 나면 Git 패널 인터페이스가 새로 고쳐져 정리된 작업 디렉터리가 표시됩니다.

stash 검색

1. stash를 검색하여 작업 디렉터리에 적용하려면 Git 패널 메뉴로 이동하여 다음 옵션 중 하나를 선택합니다.

   - [stash 적용(Apply Stash)]: 선택한 stash를 작업 디렉터리에 적용하고 나중에 사용할 수 있도록 stash를 유지합니다.
   - [Stage 팝(Pop Stash)]: 선택한 stash를 작업 디렉터리에 적용하고 해당 stash를 stash 스택에서 삭제합니다.

   > **Note**
   >
   > stash 스택에 추가된 마지막 stash를 적용하거나 팝하도록 선택할 수도 있습니다.
2. 작업 디렉터리에 적용할 stash를 선택합니다.

Git 패널 인터페이스가 새로 고쳐져 stash가 적용된 작업 디렉터리가 표시됩니다.

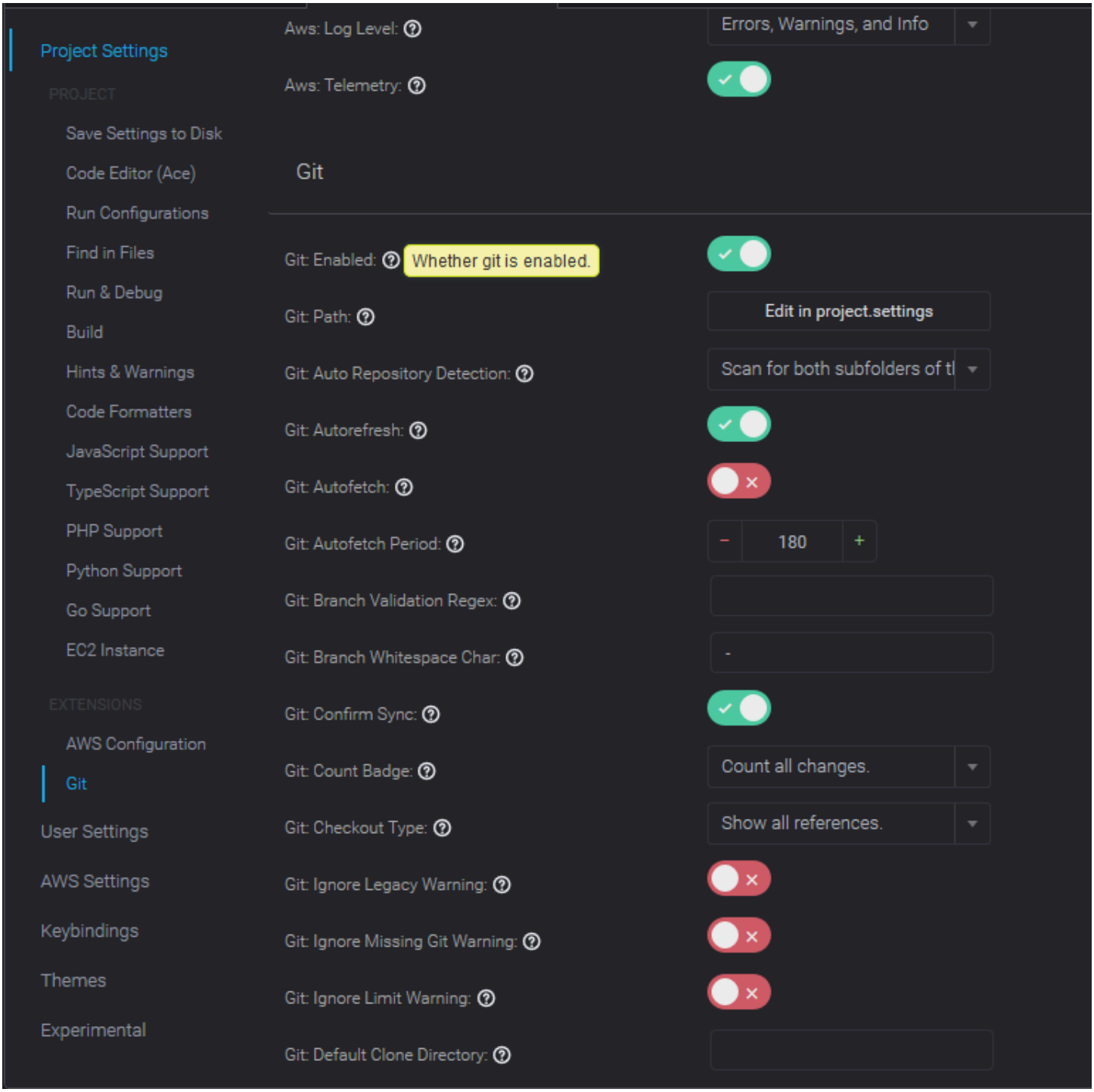
참조: Git 패널에서 사용할 수 있는 Git 명령

AWS Cloud9의 Git 패널 메뉴는 코어 및 고급 Git 명령에 대한 편리한 사용자 인터페이스 액세스를 제공합니다.

브랜치를 병합하고 삭제하는 등의 특정 git 명령은 Git 패널 검색 필드를 통해서만 사용할 수 있습니다.

Git 패널이 명령을 실행하고 리포지토리와 상호 작용하는 방법을 사용자 지정할 수도 있습니다. 기본 설정을 수정하려면 먼저 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다. 다음으로, [기본 설정(Preferences)] 창의 [프로젝트 설정(Project Settings)]에서 [Git]를 선택합니다.

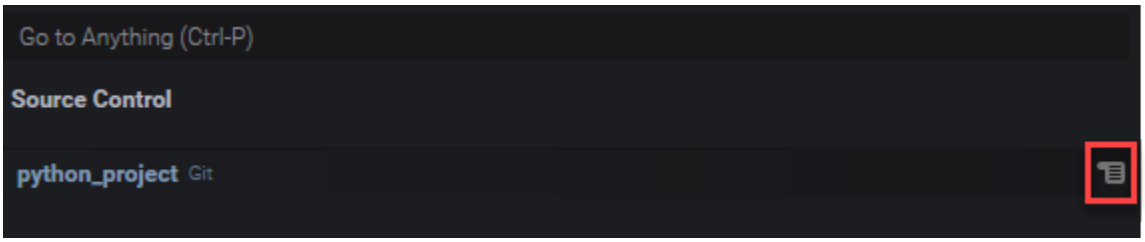
설정에 대한 간략한 설명을 보려면 정보 아이콘을 잠시 가리킵니다.



> **Note**
>
> 공식 Git 사이트(https://git-scm.com/doc)에서 나열된 Git 명령에 대한 자세한 문서서에 액세스할 수 있습니다.

Git 패널 메뉴에서 사용할 수 있는 Git 명령에 대한 참조

리포지토리의 이름 반대편에 있는 기호를 선택하여 [Git 패널(Git panel)] 메뉴의 옵션에 액세스합니다.



Git 패널 메뉴

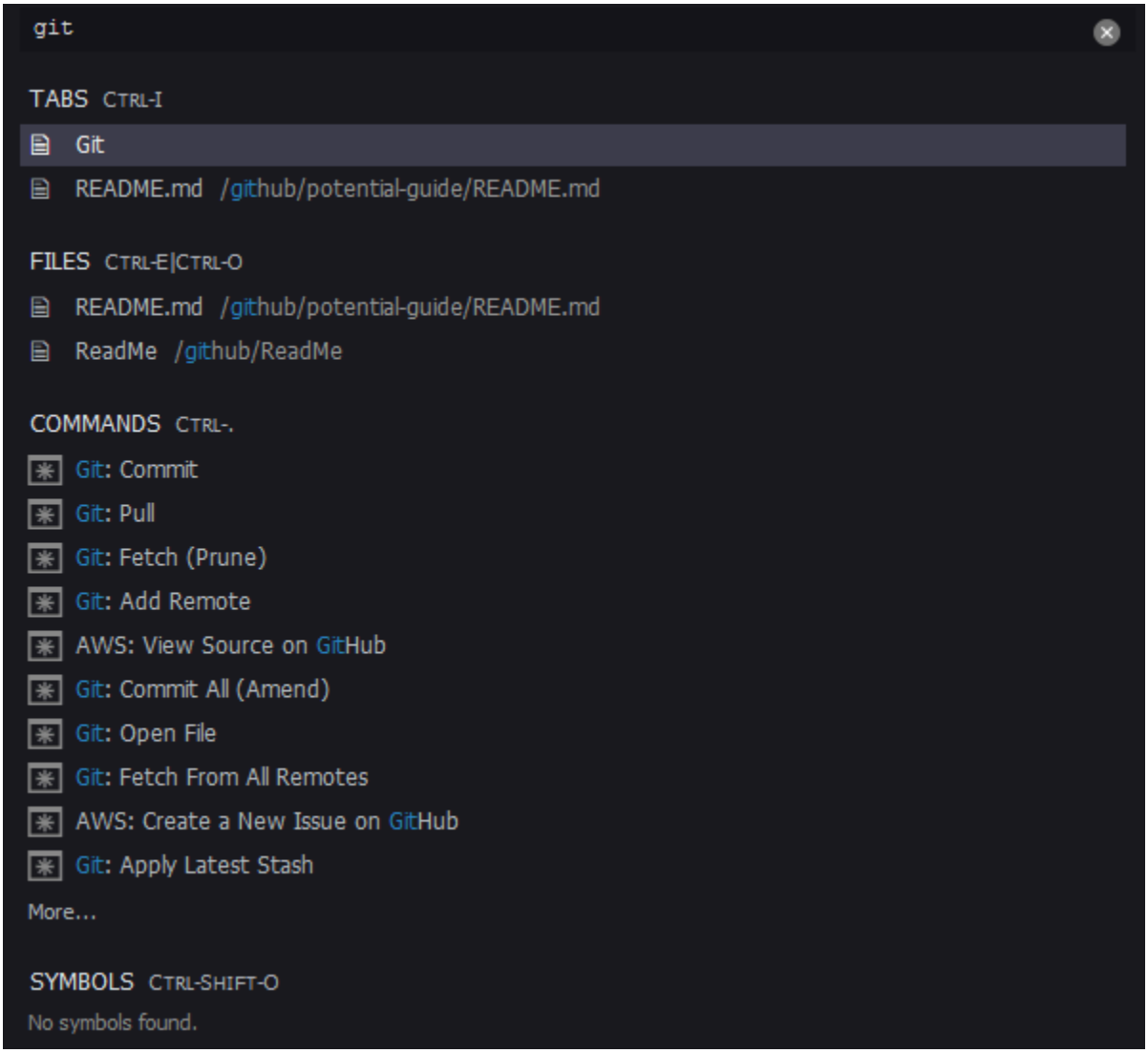
메뉴 옵션	설명
커밋	준비 영역에 추가된 콘텐츠를 리포지토리의 작업 디렉터리에 커밋합니다. 커밋 메시지를 추가합니다.
새로 고침	작업 디렉터리와 준비 영역의 상태를 표시하기 위해 GitPanel 인터페이스를 새로 고칩니다.
풀	원격 리포지토리에서 로컬 리포지토리로 최신 변경 사항을 가져옵니다.
풀(리베이스)	원격 브랜치에서 가져온 원격 변경 사항에 로컬 변경 사항을 다시 적용합니다.
다음에서 푸시...	로컬 리포지토리의 브랜치에 커밋된 변경 사항을 원격 리포지토리의 브랜치로 푸시합니다.
푸시	로컬 리포지토리에 커밋된 변경 사항을 원격 리포지토리로 푸시합니다.
다음에 푸시...	로컬 리포지토리의 브랜치에 커밋된 변경 사항을 원격 리포지토리의 브랜치로 푸시합니다.

메뉴 옵션	설명
동기화	`git pull` 명령과 `git push`명령을 차례로 실행하여 로컬 및 원격 리포지토리의 콘텐츠를 동기화합니다.
다음으로 체크아웃...	기존 브랜치로 전환하거나, 브랜치를 만들어 해당 브랜치로 전환합니다.
브랜치 게시	로컬 리포지토리에 생성된 프라이빗 브랜치를 게시하고 원격 리포지토리에서 사용할 수 있도록 합니다.
모두 커밋	스테이징된 파일과 언스테이징된 파일을 모두 리포지토리에 커밋합니다. (`git commit` 명령을 실행하기 전에 `git add -A` 명령을 실행하여 준비 영역에 파일을 추가합니다.)
모두 커밋(수정)	마지막 커밋의 메시지를 수정합니다. (`git commit` 명령을 실행할 때 `-amend` 옵션을 추가합니다.)
모두 커밋(사인오프)	Git 로그에서 커밋을 수행한 사람을 식별합니다. (`git commit` 명령을 실행할 때 `-signed-off` 옵션을 추가합니다.)
스테이징된 항목 커밋	스테이징된 파일만 리포지토리에 커밋합니다.
스테이징된 항목 커밋(수정)	마지막 커밋의 메시지를 수정합니다. (`git commit` 명령을 실행할 때 `-amend` 옵션을 추가합니다.)
스테이징된 항목 커밋(사인오프)	Git 로그에서 커밋을 수행한 사람을 식별합니다. (`git commit` 명령을 실행할 때 `-signed-off` 옵션을 추가합니다.)
마지막 커밋 실행 취소(Undo Last Commit)	이전 커밋을 실행 취소합니다. 파일이 준비 영역으로 다시 이동됩니다.
모든 변경 사항 취소	리포지토리의 준비 영역에서 모든 파일과 폴더를 삭제합니다.
모든 변경 사항 스테이징	추적되지 않거나 수정된 콘텐츠를 준비 영역에 추가합니다.

메뉴 옵션	설명
모든 변경 사항 언스테이징	모든 파일을 스테이징 영역 외부로 이동합니다. 언스테이징된 파일은 리포지토리에 커밋할 수 없습니다.
최근 Stash 적용	스택 stash에 추가된 마지막 stash를 작업 디렉터리에 적용합니다. stash는 스택에 남아 있습니다.
Stash 적용...	stash 스택에서 선택한 stash를 작업 디렉터리에 적용합니다. stash는 스택에 남아 있습니다.
최신 Stash 팝	스택 stash에 추가된 마지막 stash를 작업 디렉터리에 적용합니다. 그러면 stash가 스택에서 삭제됩니다.
Stash 팝...	선택한 stash를 작업 디렉터리에 적용합니다. 그러면 stash가 스택에서 삭제됩니다.
Stash	작업 디렉터리의 수정된 파일과 스테이징된 파일을 명명된 stash에 추가합니다.
Stash(추적되지 않은 항목 포함)	추적되지 않은 파일을 포함하여 작업 디렉터리의 모든 파일을 명명된 stash에 추가합니다.
Git 출력 표시	Git 패널 인터페이스와 상호 작용할 때 실행되는 Git 명령을 보여 주는 창을 표시합니다.

Git 패널 검색 필드에서 사용 가능한 Git 명령

검색 상자에 ‘git’를 입력하여 Git 패널 메뉴에서 사용할 수 없는 지원되는 일부 Git 명령에 액세스할 수도 있습니다.



다음 표에서는 이 방법으로 액세스할 수 있는 선택한 Git 명령에 대한 설명을 제공합니다.

선택한 Git 명령

메뉴 옵션	설명
Git: 원격 추가	Git 구성 파일에 원격 리포지토리에 대한 연결을 추가합니다.
Git: 브랜치 삭제	지정한 브랜치를 삭제합니다.
Git: 가져오기	원격 리포지토리의 브랜치에서 콘텐츠를 다운로드합니다. `git pull`과 대조적으로, 원격 변경 사항이 로컬 리포지토리에 병합되지 않습니다.

메뉴 옵션	설명
Git: 브랜치 병합	한 브랜치의 변경 사항을 다른 브랜치에 통합합니다. 자세한 내용은 브랜치 병합 절차를 참조하세요.

AWS 도구 키트

AWS 도구 키트를 사용하는 이유는 무엇입니까?

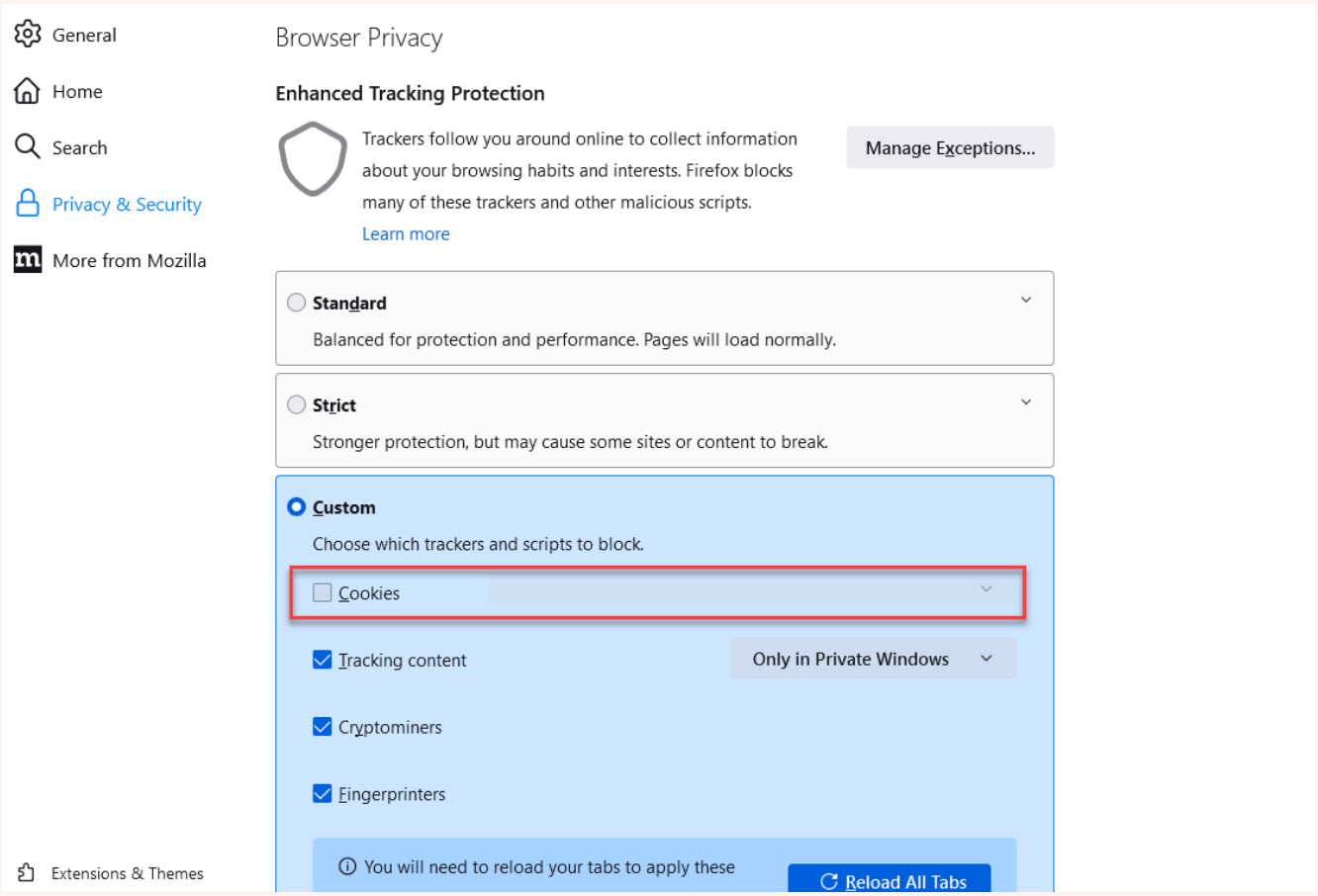
AWS 도구 키트는 AWS Cloud9 통합 개발 환경(IDE)의 확장 기능입니다. 이 확장을 사용하면 다양한 AWS 서비스에 액세스하여 작업할 수 있습니다. AWS 도구 키트는 AWS Cloud9의 Lambda 플러그인에서 제공하는 기능을 대체합니다. 자세한 내용은 AWS 도구 키트 사용 중지 섹션을 참조하세요.

Important

AWS 도구 키트 지원은 AWS Cloud9의 통합형 기능입니다. 현재는 서드 파티 확장을 사용하여 AWS Cloud9 IDE를 사용자 지정할 수 없습니다.

Warning

Mozilla Firefox를 AWS Cloud9 IDE가 설치된 기본 브라우저로 사용하는 경우 브라우저에서 AWS Cloud9 웹뷰 및 AWS Toolkit이 제대로 작동하지 않도록 하는 타사 쿠키 설정이 있습니다. 이 문제를 해결하려면 아래 이미지에 표시된 대로 브라우저 설정의 개인 정보 보호 및 보안 섹션에서 쿠키를 차단하지 않았는지 확인해야 합니다.



현재 AWS 도구 키트 확장을 통해 다음 AWS 서비스 및 리소스에 액세스할 수 있습니다.

- AWS App Runner
- API Gateway
- AWS CloudFormation 스택
- CloudWatch Logs
- AWS Lambda
- 리소스
- Amazon S3 버킷 및 객체
- AWS Serverless Application Model 애플리케이션
- Step Functions 및 상태 시스템
- Systems Manager 자동화 문서
- AWS Cloud9 IDE에서 Amazon ECR 작업

- AWS IoT
- ???
- Amazon EventBridge
- Amazon CodeWhisperer 작업
- AWS Cloud Development Kit (AWS CDK) 작업

AWS 도구 키트 사용

환경에서 AWS 도구 키트를 사용할 수 없는 경우 [기본 설정(Preferences)] 탭에서 사용하도록 설정할 수 있습니다.

AWS 도구 키트를 사용하려면

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. 기본 설정(Preferences) 탭 측면의 탐색 창에서 AWS 설정(AWS Settings)을 선택합니다.
3. AWS Resources(리소스) 창에서 AWS Toolkit(도구 키트)를 활성화하여 녹색 배경에 확인 표시가 표시되도록 합니다.

   AWS 도구 키트를 활성화하면 통합 개발 환경(IDE) 이 새로고침되어 업데이트된 Enable AWS Toolkit(도구 키트 활성화) 설정이 표시됩니다. 또한 Environment(환경) 옵션 아래 IDE 측면에 AWS Toolkit(도구 키트) 옵션이 표시됩니다.

> **Important**
>
> AWS Cloud9 환경의 EC2 인스턴스가 인터넷에 액세스할 수 없는 경우(즉 아웃바운드 트래픽이 허용되지 않음) AWS 도구 키트를 켜고 IDE를 다시 활성화하면 메시지가 표시될 수 있습니다. 이 메시지는 AWS 도구 키트에 필요한 종속 구성 요소를 다운로드할 수 없다는 내용입니다. 이 경우 AWS 도구 키트도 사용할 수 없습니다.
> 이 문제를 해결하려면 Amazon S3용 VPC 엔드포인트를 생성합니다. 이렇게 하면 IDE를 최신 상태로 유지하는 데 필요한 종속 구성 요소가 포함된 AWS 리전에 있는 Amazon S3 버킷에 액세스할 수 있습니다.
> 자세한 내용은 종속 구성 요소를 다운로드하도록 Amazon S3의 VPC 엔드포인트 구성 섹션을 참조하세요.

AWS 도구 키트에 대한 액세스 자격 증명 관리

AWS 도구 키트는 다양한 AWS 서비스와 상호 작용합니다. 액세스 제어를 관리하려면 AWS 도구 키트 서비스의 IAM 엔터티에 이렇게 다양한 서비스 범위에 필요한 권한이 있는지 확인하십시오. 먼저 AWS 관리형 임시 보안 인증 정보를 사용하여 필요한 권한을 얻으십시오. 이러한 관리형 보안 인증 정보는 IAM 사용자 같은 AWS 엔터티 대신 AWS 서비스에 EC2 환경 액세스 권한을 부여하는 방식으로 작동합니다.

하지만 개발 환경의 EC2 인스턴스를 프라이빗 서브넷에서 시작하면 AWS 관리형 임시 자격 증명을 사용할 수 없습니다. 대신 자체 보안 인증 정보 세트를 수동으로 만들어 AWS 도구 키트가 AWS 서비스에 액세스하도록 허용할 수 있습니다. 이 세트를 프로파일이라고 합니다. 프로파일에는 액세스 키라는 장기 보안 인증 정보가 있습니다. IAM 콘솔에서 이러한 액세스 키를 얻을 수 있습니다.

AWS 도구 키트의 액세스 자격 증명을 제공하는 프로파일 만들기

1. 액세스 키(액세스 키 ID 및 비밀 액세스 키로 이루어짐)를 얻으려면 IAM 콘솔(https://console.aws.amazon.com/iam)로 이동합니다.
2. 탐색 모음에서 [사용자(Users)]를 선택한 다음 AWS 사용자 이름(확인란 아님)을 선택합니다.
3. [보안 자격 증명(Security credentials)] 탭을 선택한 후 [액세스 키 생성(Create access key)]을 선택합니다.

   > **Note**
   >
   > 이미 액세스 키가 있지만 비밀 키에 액세스할 수 없는 경우 이전 키를 비활성화하고 새 키를 만듭니다.

4. 액세스 키 ID와 비밀 액세스 키가 표시된 대화 상자에서 [.csv 파일 다운로드(Download .csv file)]를 선택하여 이 정보를 안전한 위치에 저장합니다.
5. 액세스 키를 다운로드한 후 Windo)(창), New Terminal(새 터미널)을 선택하여 AWS Cloud9 환경을 시작하고 터미널 세션을 시작합니다.
6. 터미널 창에서 다음 명령을 실행합니다.

   ```
   aws configure --profile toolkituser
   ```

   이 예에서는 `toolkituser`라는 프로파일 이름을 사용 중이지만 직접 선택할 수 있습니다.
7. 명령줄에서 IAM 콘솔에서 앞서 다운로드한 `AWS Access Key ID` 및 `AWS Secret Access Key`를 입력합니다.

- `Default region name`에 AWS 리전을 지정합니다(예: `us-east-1`).
- `Default output format`에 파일 형식을 지정합니다(예: `json`).

> **Note**
>
> 프로파일 구성 옵션에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서에서 구성 기본 사항을 참조하세요.

8. 프로파일을 만든 후 AWS 도구 키트를 시작하고 AWS Toolkit menu(도구 키트 메뉴)로 이동한 다음 Connect to AWS(AWS에 연결)을 선택합니다.
9. Select an AWS credential profile(보안 인증 정보 프로파일 선택) 필드에서 방금 터미널에서 만든 프로필을 선택합니다(예: `profile:toolkituser`).

선택한 프로파일에 유효한 액세스 자격 증명이 포함된 경우 AWS 탐색기 창이 새로 고쳐져 액세스할 수 있는 AWS 서비스가 표시됩니다.

IAM 역할을 사용하여 EC2 인스턴스의 애플리케이션에 권한 부여

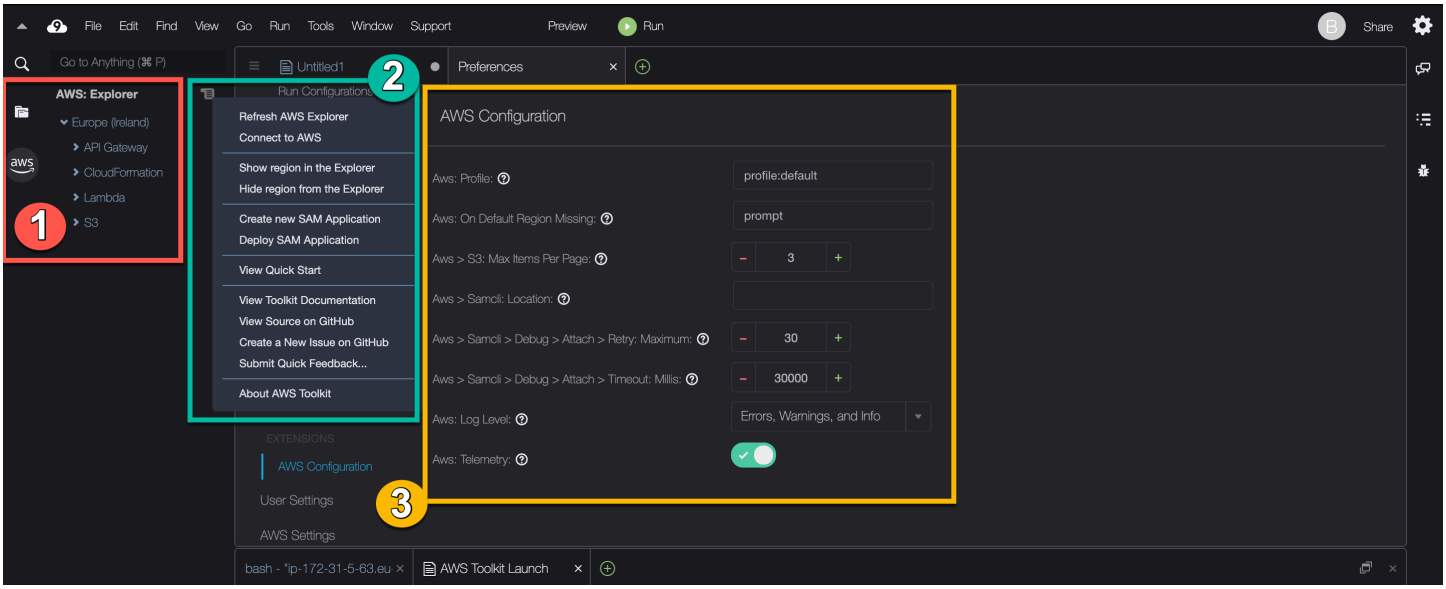
IAM 역할을 사용하여 EC2 인스턴스에서 실행되는 애플리케이션의 임시 자격 증명을 관리할 수도 있습니다. 애플리케이션에서 다른 AWS 리소스를 호출할 때 사용할 수 있는 임시 권한을 역할이 제공합니다. EC2 인스턴스를 시작할 때 IAM 역할을 지정해 인스턴스에 연결합니다. 그러면 AWS 서비스에 대해 API 요청을 실행할 때 이 인스턴스에서 실행되는 애플리케이션은 역할 제공 임시 자격 증명을 사용할 수 있습니다.

역할을 만든 후에는 인스턴스 프로파일을 만들어 이 역할 및 관련 권한을 인스턴스에 할당합니다. 그러면 인스턴스 프로파일이 인스턴스에 연결되고 인스턴스에서 실행되는 애플리케이션에 이 역할의 임시 자격 증명을 제공할 수 있습니다.

자세한 내용은 IAM 사용 설명서의 IAM 역할을 사용하여 Amazon EC2 인스턴스에서 실행되는 애플리케이션에 권한 부여를 참조하세요.

AWS 도구 키트 구성 요소 식별

아래 스크린샷은 AWS 도구 키트의 세 가지 주요 UI 구성 요소를 보여줍니다.



1. AWS 탐색기 창: 도구 키트를 통해 액세스 가능한 AWS 서비스와 상호 작용하는 데 사용합니다. 통합 개발 환경(IDE) 왼쪽에 있는 AWS 옵션을 사용하여 AWS 탐색기를 표시하거나 숨길 수 있습니다. 이 인터페이스 구성 요소와 다른 AWS 리전의 AWS 서비스 액세스에 대한 자세한 내용은 AWS 탐색기를 통해 여러 리전의 서비스 및 리소스 사용 섹션을 참조하세요.
2. 도구 키트 메뉴: AWS에 대한 연결을 관리하고 AWS 탐색기 창의 디스플레이를 사용자 지정하고, 서버리스 애플리케이션을 생성 및 배포하고, GitHub 리포지토리 작업을 수행하고, 문서에 액세스하는 데 사용합니다. 자세한 내용은 AWS 도구 키트 메뉴 액세스 및 사용 섹션을 참조하세요.
3. AWS 구성 창: 도구 키트를 사용하여 상호 작용하는 AWS 서비스의 동작을 사용자 지정하는 데 사용합니다. 자세한 내용은 AWS 구성 창을 사용하여 AWS 도구 키트 설정 수정 섹션을 참조하세요.

AWS 도구 키트 사용 중지

[기본 설정(Preferences)] 탭에서 AWS 도구 키트를 사용 중지할 수 있습니다.

AWS 도구 키트를 사용 중지하려면

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. 기본 설정(Preferences) 탭 측면의 탐색 창에서 AWS 설정(AWS Settings)을 선택합니다.
3. AWS 리소스(AWS Resources) 창에서 AWS AWS 도구 키트를 해제합니다.

   AWS 도구 키트를 사용 중지하면 통합 개발 환경(IDE)이 새로 고쳐져 IDE의 측면에 있는 Environment(환경) 옵션 아래에서 AWS 도구 키트가 제거됩니다.

AWS 도구 키트 주제

AWS 도구 키트 탐색 및 구성

다음 AWS 도구 키트 인터페이스 요소를 통해 리소스에 액세스하고 설정을 수정할 수 있습니다.

- AWS 탐색기 창: 다른 AWS 리전에서 AWS 서비스에 액세스합니다.
- AWS 도구 키트 메뉴: 서버리스 애플리케이션을 만들어 배포하고, AWS 리전을 표시하거나 숨기고, 사용자 지원에 액세스하며, Git 리포지토리와 상호 작용합니다.
- AWS 구성 창: AWS 도구 키트에서 AWS 서비스와 상호 작용할 수 있는 방법에 영향을 주는 설정을 수정합니다.

AWS 탐색기를 통해 여러 리전의 서비스 및 리소스 사용

AWS 탐색기 창을 사용하면 AWS 서비스를 선택하고 해당 서비스와 관련한 특정 리소스를 사용할 수 있습니다. AWS 탐색기에서 서비스 이름 노드(예: API Gateway 또는 Lambda)를 선택합니다. 그런 다음 해당 서비스와 관련된 특정 리소스(예: REST API 또는 Lambda 함수)를 선택합니다. 특정 리소스를 선택하면 메뉴에 업로드 또는 다운로드, 호출 또는 복사와 같은 가능한 상호 작용 옵션이 표시됩니다.

다음 예제를 살펴보세요. AWS 계정 보안 인증 정보가 Lambda 함수에 액세스할 수 있다면, AWS 리전에 대해 나열된 Lambda 노드를 확장한 다음 AWS Cloud9 IDE에 코드로 호출하거나 업로드할 특정 Lambda 함수를 선택합니다. 또한 노드 제목의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열어 AWS Serverless Application Model을 사용하는 애플리케이션 생성을 시작할 수도 있습니다.

> **Note**
>
> 통합 개발 환경(IDE)에서 AWS 탐색기 창을 보는 옵션이 표시되지 않을 경우 AWS 도구 키트를 활성화했는지 확인하세요. 활성화되었는지 확인한 후 다시 시도하세요. 자세한 내용은 AWS 도구 키트 사용 섹션을 참조하세요.

AWS 탐색기 창에는 여러 AWS 리전에서 호스트되는 서비스가 표시될 수도 있습니다.

선택한 리전에서 AWS 서비스에 액세스하려면

1. AWS 탐색기 창에서 [도구 키트(Toolkit)] 메뉴, [탐색기에 리전 표시(Show region in the Explorer)]를 선택합니다.
2. Select a region to show in the AWS Explorer)(탐색기에 표시할 리전 선택) 목록에서 AWS 리전을 선택합니다.

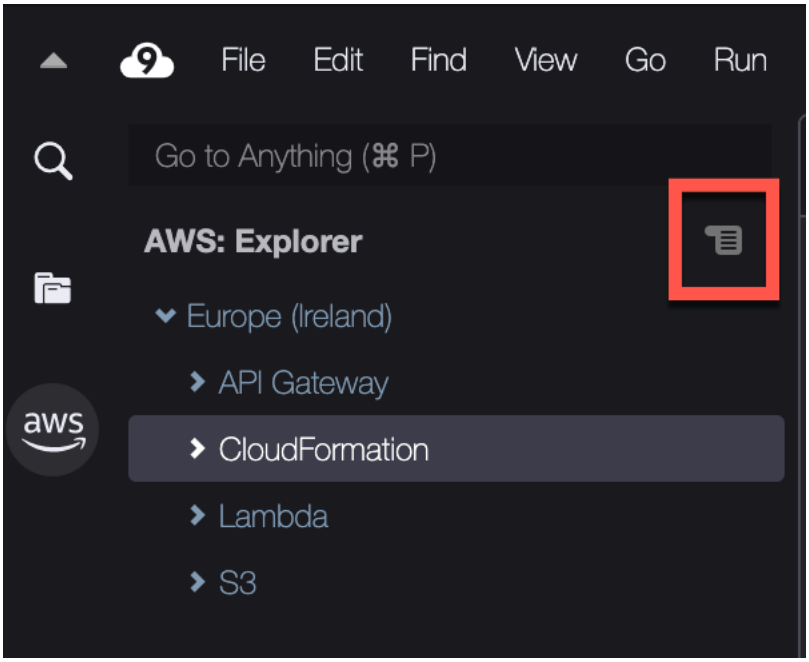
   선택한 리전이 AWS 탐색기 창에 표시됩니다. 사용 가능한 서비스 및 리소스에 액세스하려면 리전의 이름 앞에 있는 화살표(>)를 선택합니다.

> **Note**
>
> 다음 옵션을 사용하여 선택한 AWS 리전을 AWS 탐색기 창에서 숨길 수도 있습니다.
>
> - 리전의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Hide region from the Explorer(탐색기에서 리전 숨기기)를 선택합니다.
> - AWS 도구 키트 메뉴에서 Hide region from the Explorer(탐색기에서 리전 숨기기)를 선택하고 숨길 리전을 선택합니다.

AWS 도구 키트 메뉴 액세스 및 사용

AWS 도구 키트는 서버리스 애플리케이션을 만들고 배포하는 옵션에 대한 액세스를 제공합니다. 이 메뉴를 사용하면 연결을 관리하고 AWS: 탐색기 창을 업데이트하고 문서에 액세스하고 GitHub 리포지토리와 상호 작용할 수 있습니다.

도구 키트 메뉴에 액세스하려면 AWS 탐색기 창에서 AWS: 탐색기 제목의 반대편에 있는 스크롤 아이콘을 선택합니다.



다음 표에서는 Toolkit(도구 키트) 메뉴에서 사용할 수 있는 옵션 개요를 확인할 수 있습니다.

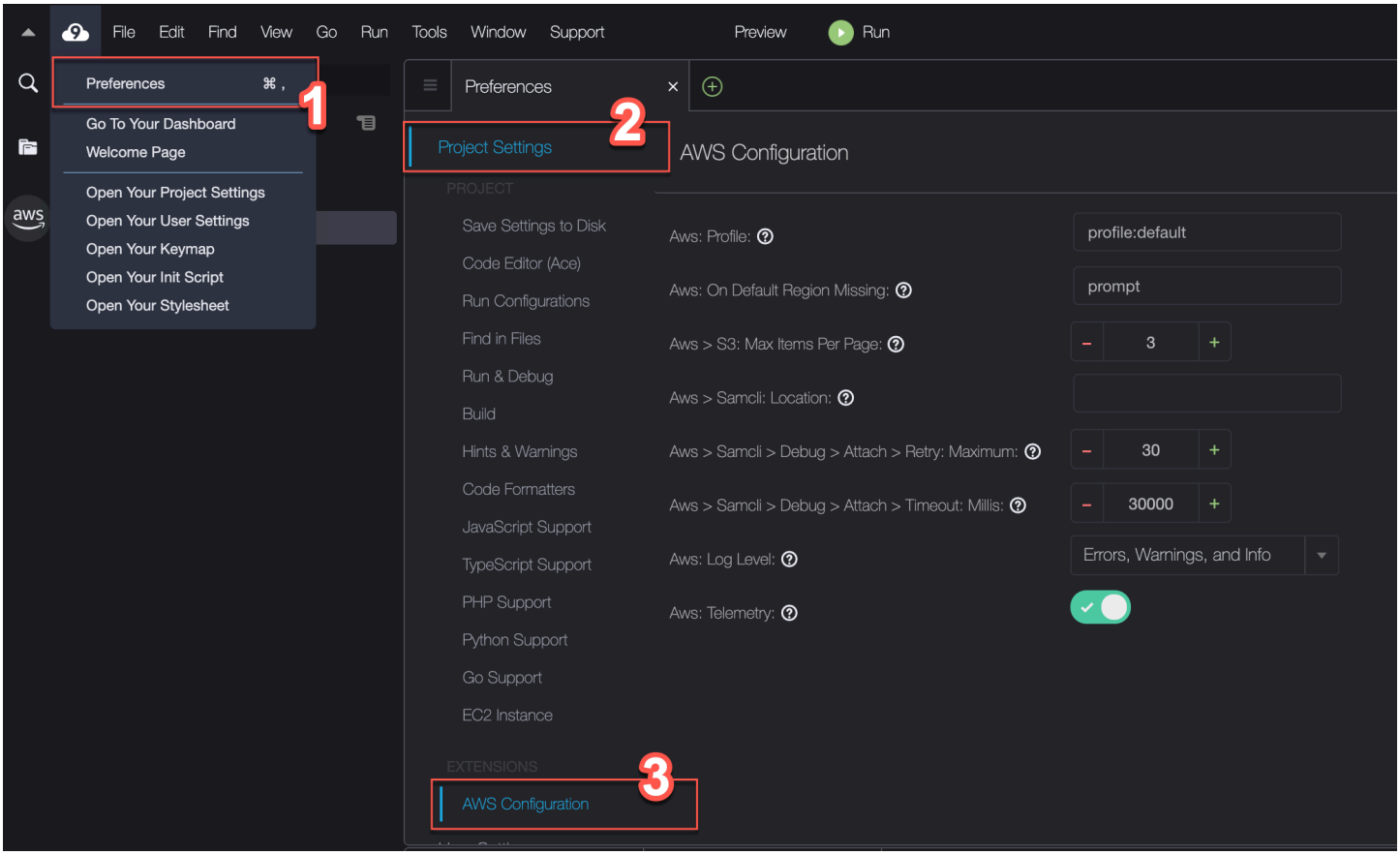
도구 키트(Toolkit) 메뉴 옵션

메뉴 옵션	설명
AWS Explorer 새로 고침	AWS 탐색기를 새로 고쳐 창을 마지막으로 연 이후에 수정된 AWS 서비스를 표시하려면 이 옵션을 선택합니다.
AWS에 연결	profile(프로파일)에 저장된 보안 인증 정보를 사용하여 AWS 도구 키트를 AWS 계정에 연결합니다. 자세한 내용은 AWS 도구 키트에 대한 액세스 자격 증명 관리 섹션을 참조하세요.
탐색기에 리전 표시	AWS 탐색기 창에 AWS 리전을 표시합니다. 자세한 내용은 AWS 탐색기를 통해 여러 리전의 서비스 및 리소스 사용 섹션을 참조하세요.
탐색기에서 리전 숨기기	AWS 탐색기 창에서 AWS 리전을 숨깁니다. 자세한 내용은 AWS 탐색기를 통해 여러 리전의 서비스 및 리소스 사용 섹션을 참조하세요.

메뉴 옵션	설명
새 SAM 애플리케이션 생성	새 AWS 서버리스 애플리케이션의 코드 파일 세트를 생성합니다. SAM 애플리케이션을 생성하고 배포하는 방법에 대한 자세한 정보는 AWS 도구 키트를 사용한 AWS 서버리스 애플리케이션 작업 단원을 참조하십시오.
SAM 애플리케이션 배포	서버리스 애플리케이션을 AWS에 배포합니다. SAM 애플리케이션을 생성하고 배포하는 방법에 대한 자세한 정보는 AWS 도구 키트를 사용한 AWS 서버리스 애플리케이션 작업 단원을 참조하십시오.
빠른 시작 보기	빠른 시작 가이드를 엽니다.
도구 키트 문서 보기	AWS 도구 키트의 사용 설명서를 엽니다.
GitHub에서 소스 보기	AWS 도구 키트에 대한 GitHub 리포지토리를 엽니다.
GitHub에서 새로운 사례 만들기	Github에서 AWS 도구 키트의 새로운 사례 페이지를 엽니다.
빠른 피드백 제출	비공개적인 단방향 피드백을 AWS 도구 키트 개발 팀에 제출합니다. 대화 또는 버그 수정이 필요한 문제가 있다면 Create a New Issue on Github(Github에서 새로운 사례 만들기) 메뉴 옵션을 선택하여 Github에서 사례를 제출하세요.
AWS도구 키트 정보	실행 중인 도구 키트 버전 및 해당 도구 키트가 구성된 Amazon 운영 체제에 대한 정보를 표시합니다.

AWS 구성 창을 사용하여 AWS 도구 키트 설정 수정

AWS 구성 창에 액세스하려면 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다. 다음으로 Preferences(기본 설정) 창의 Project Settings(프로젝트 설정)에서 AWS Configuration(구성)을 선택합니다.



다음 표에서는 AWS Configuration(구성) 페이지에서 사용할 수 있는 옵션 개요를 확인할 수 있습니다.

메뉴 옵션	설명
AWS: 프로파일	자격 증명을 가져올 자격 증명 프로파일의 이름을 설정합니다.
AWS: 기본 리전 누락 시	선택한 자격 증명 프로파일의 기본 AWS 리전을 AWS 탐색기 창에서 사용할 수 없는 경우 취할 조치를 나타냅니다. 다음 세 가지 옵션 중에서 선택할 수 있습니다.

메뉴 옵션	설명
	• prompt(프롬프트)(기본값): 사용자에게 수행할 작업을 묻는 메시지가 표시됩니다. • add(추가): 리전이 AWS 탐색기 창에 표시됩니다. • 무시(ignore): 아무 작업도 수행하지 않습니다.
AWS > S3: 페이지당 최대 항목 수	AWS 탐색기창에 한 번에 표시되는 Amazon S3 객체 또는 폴더의 수를 지정합니다. 최대 개수가 표시되면더 불러오기을 클릭하여 다음 배치를 표시합니다. 이 필드에 허용되는 값의 범위는 3에서 1,000 사이입니다. 이 설정은 한 번에 표시되는 객체 또는 폴더 수에만 적용됩니다. 생성한 모든 버킷이 한 번에 표시됩니다. 기본적으로 AWS 계정 각각에 대해 최대 100개의 버킷을 만들 수 있습니다.
AWS > Samcli: 위치	서버리스 애플리케이션을 생성, 빌드, 패키지 및 배포하는 데 사용되는 SAM CLI의 위치를 나타냅니다.
AWS > Samcli > 디버그 > 연결 > 재시도: 최대:	도구 키트가 포기하기 전에 SAM CLI 디버거를 연결하려고 시도하는 횟수를 지정합니다. 기본 할당량은 30회입니다. AWS SAMCLI 내에서 디버그 모드로 Lambda 함수를 로컬로 호출하면 디버거를 연결할 수 있습니다.

메뉴 옵션	설명
AWS > Samcli > 디버그 > 연결 > 시간 초과: 밀리초:	도구 키트가 포기하기 전에 SAM CLI 디버거를 연결하려고 시도하는 시간을 지정합니다. 기본값은 30,000밀리초(30초)입니다. AWS SAMCLI 내에서 디버그 모드로 Lambda 함수를 로컬로 호출하면 디버거를 연결할 수 있습니다.
AWS : 로그 수준:	로깅되는 워크플로 이벤트의 범주를 설정합니다. 다음 수준을 사용할 수 있습니다. • 오류만 • 오류 및 경고 • 오류, 경고 및 정보(기본 옵션) • 오류, 경고 및 정보, 자세한 정보 및 디버그
AWS : 원격 측정	AWS에 사용량 데이터를 전송하는 기능을 사용하거나 사용 중지합니다. 기본적으로 사용됩니다.

AWS 도구 키트를 사용한 API Gateway 작업

API Gateway를 사용하여 실시간 양방향 통신 애플리케이션을 지원하는 RESTful API 및 WebSocket API를 생성할 수 있습니다. API Gateway를 사용하여 API를 생성하고 관리하는 방법에 대한 자세한 내용은 API Gateway 개발자 가이드를 참조하세요.

AWS 도구 키를 사용하면 REST 리소스, 메서드 유형 및 입력으로 전달되는 데이터를 지정하여 REST API에 대한 호출을 구성할 수 있습니다.

API Gateway에서 REST API 호출

> **Important**
>
> AWS 도구 키트를 사용하여 API 메서드를 호출하면 리소스가 변경될 수 있으며 이 변경은 취소할 수 없습니다. 예를 들어 POST 메서드를 호출하면 호출이 성공할 경우 API의 리소스가 업데이트됩니다.

AWS 도구 키트에서 AWS의 API Gateway를 호출할 수 있습니다.

REST API를 호출하려면

1. AWS 탐색기 창에서 API Gateway 노드를 선택하여 현재 AWS 리전에서 사용 가능한 REST API의 목록을 봅니다.
2. REST API를 마우스 오른쪽 버튼으로 클릭한 다음 Invoke on AWS(AWS에서 호출)를 선택합니다.

   > **Note**
   >
   > 컨텍스트 메뉴를 사용하여 REST API의 URL, 이름 및 Amazon 리소스 이름(ARN)을 복사할 수 있습니다.

   Invoke methods(메서드 호출) 창이 표시됩니다. API에 대한 호출을 구성할 수 있습니다.
3. Select a resource(리소스 선택)에서 상호 작용할 REST 리소스를 선택합니다.
4. [메서드 선택(Select a method)]에서 다음 메서드 유형 중 하나를 선택합니다.
   - [GET]: API를 통해 액세스되는 백엔드 서비스에서 리소스를 가져옵니다.
   - [OPTIONS]: API Gateway가 지원하는 메서드 및 작업에 대한 정보를 요청합니다.
   - [POST]: API를 통해 액세스되는 백엔드 서비스에 새 리소스를 만듭니다.
5. API 메서드 호출에 입력을 제공하려면 쿼리 문자열 또는 JSON 형식의 페이로드를 사용할 수 있습니다.
   - Query string(쿼리 문자열): `parameter1=value1&parameter2=value2` 형식을 사용하여 쿼리 문자열을 입력합니다. (쿼리 문자열을 사용하려면 먼저 매핑 템플릿을 생성하여 들어오는 웹 요청이 통합 백엔드로 전송되기 전에 변환합니다.)

- JSON 형식: [메서드 호출(Invoke methods)] 창의 라지 텍스트 필드에 JSON 형식의 페이로드를 정의할 수 있습니다.

  예를 들어 다음 페이로드를 포함하는 POST 메서드를 사용하여 새 리소스를 추가할 수 있습니다.

  ```
  {"type": "soda", "price" : 3.99}
  ```

6. [호출(Invoke)] 버튼을 선택하여 REST API 리소스를 호출합니다.

   REST API 응답이 AWS Remote Invocations(원격 호출) 탭에 표시됩니다. 응답 본문에는 JSON 형식의 리소스 데이터가 포함됩니다.

AWS 도구 키트와 AWS App Runner 사용

AWS App Runner는 소스 코드 또는 컨테이너 이미지에서 AWS 클라우드의 확장 가능하고 안전한 웹 애플리케이션으로 직접 배포하는 빠르고 비용 효율적인 방법을 제공합니다. 이를 사용하면 새로운 기술을 배우거나 사용할 컴퓨팅 서비스를 결정하거나 AWS 리소스를 프로비저닝 및 구성하는 방법을 알 필요가 없습니다.

소스 이미지 또는 소스 코드를 기반으로 서비스를 생성하고 관리하는 데 AWS App Runner를 사용할 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 Container Registry(Amazon ECR): AWS 계정에 프라이빗 이미지를 저장합니다.
- 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 제공업체로 GitHub를 지원합니다.

사전 조건

AWS 도구 키트를 사용하여 App Runner와 상호 작용하려면 다음이 필요합니다.

- AWS 계정
- AWS App Runner를 제공하는 AWS 도구 키트 버전

이러한 핵심 요구 사항 외에도 모든 관련 IAM 사용자에게 App Runner 서비스와 상호 작용할 수 있는 권한이 있는지 확인하십시오. 또한 컨테이너 이미지 URI 및 GitHub 리포지토리에 대한 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 빠르게 부여하려면, 기존 AWS 관리형 정책을 관련 AWS Identity and Access Management(IAM) 엔터티를 연결합니다. 특히 사용자 또는 그룹에 정책을 연결할 수 있습니다. 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.

AWS 도구 키트의 명령 창을 사용하여 서비스 인스턴스를 구성할 때 이 역할을 자동으로 생성할 수 있습니다.

> **Note**
>
> AWSServiceRoleForAppRunner 서비스 연동 역할을 통해 AWS App Runner가 다음 작업을 완료할 수 있습니다.
>
> - Amazon CloudWatch Logs에 로그 그룹에 로그를 푸시합니다.
> - Amazon Elastic Container Registry(Amazon ECR) 이미지 푸시를 구독하는 Amazon CloudWatch Events 규칙을 생성합니다.
>
> 서비스 연동 역할을 수동으로 생성하지 않아도 됩니다. AWS 도구 키트에서 호출하는 API 작업을 사용하여 또는 AWS Management Console에서 AWS App Runner를 생성할 때 AWS App Runner에서 이 서비스 연결 역할을 생성합니다.

자세한 내용은 AWS App Runner 개발자 안내서에서 App Runner의 자격 증명 및 액세스 관리를 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 리포지토리에 대한 링크를 얻을 수 있습니다.

- Amazon ECR 프라이빗 레지스트리: https://console.aws.amazon.com/ecr/repositories에서 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**
>
> AWS 도구 키트의 AWS Explorer에서 직접 프라이빗 Amazon ECR 리포지토리에 대한 URI를 얻을 수도 있습니다.
>
> - AWS Explorer를 열고 ECR 노드를 확장하여 AWS 리전에 대한 리포지토리 목록을 봅니다.
> - 리포지토리의 컨텍스트(마우스 오른쪽 단추 클릭) 메뉴를 열고 Copy Repository URI(리포지토리 URI 복사)를 선택하여 링크를 클립보드에 복사합니다.

AWS 도구 키트 명령 창으로 서비스 인스턴스를 구성할 때 이미지 리포지토리에 대한 URI를 지정합니다.

자세한 내용은 AWS App Runner 개발자 안내서의 소스 이미지 기반의 App Runner 서비스를 참조하세요.

Source code

소스 코드를 AWS App Runner 서비스에 배포하려면 Git 리포지토리에 해당 코드를 저장해야 합니다. 이 Git 리포지토리는 지원되는 리포지토리 공급자가 유지 관리해야 합니다. App Runner는 하나의 소스 코드 리포지토리 공급자인 GitHub를 지원합니다.

GitHub 리포지토리를 설정하는 방법에 대한 자세한 내용은 GitHub의 시작하기 설명서를 참조하세요.

GitHub 리포지토리에서 App Runner 서비스에 소스 코드를 배포하기 위해 App Runner는 GitHub에 대한 연결을 설정합니다. 리포지토리가 프라이빗인 경우(즉, GitHub에서 공개적으로 액세스할 수 없는 경우) App Runner에 연결 세부 정보를 제공해야 합니다.

> **Important**
>
> GitHub 연결을 생성하려면 App Runner 콘솔(https://console.aws.amazon.com/apprunner)을 사용하여 GitHub를 AWS에 연결하는 연결을 생성해야 합니다. AWS 도구 키트의 명령 창으로 서비스 인스턴스를 구성할 때 GitHub 연결(GitHub connections) 페이지에서 사용 가능한 연결을 선택할 수 있습니다.
> 자세한 내용은 AWS App Runner 개발자 안내서의 App Runner 연결 관리를 참조하세요.

App Runner 서비스 인스턴스는 코드를 빌드하고 실행할 수 있는 관리형 런타임을 제공합니다. AWS App Runner에서는 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 파일을 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 지불합니다.

자세한 내용은 AWS App Runner 요금을 참조하십시오.

주제

App Runner 서비스 생성

AWS Explorer를 사용하여 AWS 도구 키트에서 App Runner 서비스를 생성할 수 있습니다. 특정 AWS 리전에서 서비스를 생성하기로 선택하면, AWS 도구 키트의 명령 창에서 애플리케이션이 실행되는 서비스 인스턴스를 구성하는 방법을 확인할 수 있습니다.

App Runner 서비스를 생성하기 전에 전제 조건을 완료했는지 확인합니다. 여기에는 관련 IAM 권한을 제공하고 배포하려는 특정 소스 리포지토리를 확인하는 작업이 포함됩니다.

App Runner 서비스 생성

1. AWS Explorer가 열려 있지 않은 경우 이를 엽니다.
2. App Runner 노드를 마우스 오른쪽 버튼으로 클릭하고 서비스 생성(Create Service)을 선택합니다.

   AWS 도구 키트 명령 창이 표시됩니다.
3. 소스 코드 위치 유형 선택(Select a source code location type)에서 ECR 또는 리포지토리(Repository)를 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 GitHub를 소스 코드 리포지토리 제공자로 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 자동 배포가 꺼져 있는지 확인합니다. App Runner는 ECR 퍼블릭 리포지토리에 있는 이미지의 자동 배포를 지원하지 않습니다.
> 자동 배포는 기본적으로 꺼짐 상태입니다. 명령 창 머리글의 아이콘에 대각선이 있다면 꺼짐 상태라는 뜻입니다. 자동 배포를 사용하기로 한 경우 추가 비용이 발생할 수 있다는 메시지가 표시됩니다.

2. 명령 창의 단계에서 No tags found(태그를 찾을 수 없음)을 보고하는 경우 태그가 지정된 컨테이너 이미지가 포함된 리포지토리를 선택하는 단계로 돌아갑니다.
3. Port(포트)에서 서비스에서 사용하는 IP 포트(예를 들어 포트 8000)를 입력합니다.
4. (선택 사항) Configure environment variables(환경 변수 구성)에서 서비스 인스턴스의 동작을 사용자 지정하는 데 사용하는 환경 변수가 포함된 파일을 지정합니다.
5. Amazon ECR 프라이빗 레지스트리를 사용하는 경우 AppRrunneRecrAccessRole ECR 액세스 역할이 필요합니다. 이 역할은 App Runner가 계정에서 Amazon Elastic Container Registry(Amazon ECR) 이미지에 액세스할 수 있도록 허용합니다. 명령 창 헤더에서 “+” 아이콘을 선택하여 이 역할을 만듭니다. 이미지가 공개적으로 제공되는 Amazon ECR 퍼블릭에 이미지가 저장되어 있다면 액세스 역할은 필요 없습니다.
6. Name your service(서비스 이름 지정)에서 고유한 이름을 입력하고 Enter를 누릅니다. 이름에는 공백이 있어서는 안 됩니다.
7. Select instance configuration(인스턴스 구성 선택)에서 서비스 인스턴스의 CPU 유닛과 메모리(두 항목 모두 GB 단위) 조합을 선택합니다.

   서비스가 생성되면 상태가 생성(Creating)에서 실행(Running)으로 변경됩니다.
8. 서비스 실행을 시작한 후 서비스의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Copy Service URL(서비스 URL 복사)을 선택합니다.
9. 배포된 애플리케이션에 액세스하려면 복사한 URL을 웹 브라우저의 주소 표시줄에 붙여넣습니다.

원격 리포지토리에서 배포

1. 연결 선택(Select a connection)에서 GitHub를 AWS에 연결하는 연결을 선택합니다. 선택할 수 있는 연결은 App Runner 콘솔의 GitHub 연결(GitHub connections) 페이지에 나열됩니다.
2. 원격 GitHub 리포지토리 선택(Select a remote GitHub repository)에서 원격 리포지토리의 URL을 선택하거나 입력합니다.

AWS Cloud9 소스 제어 관리로 이미 구성된 원격 리포지토리를 선택할 수 있습니다. 리포지토리가 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 명령 창을 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. Port(포트)에서 서비스에서 사용하는 IP 포트(예를 들어 포트 `8000`)를 입력합니다.
6. (선택 사항) Configure environment variables(환경 변수 구성)에서 서비스 인스턴스의 동작을 사용자 지정하는 환경 변수가 포함된 파일을 지정합니다.
7. Name your service(서비스 이름 지정)에서 고유한 이름을 입력하고 Enter를 누릅니다. 이름에는 공백이 있어서는 안 됩니다.
8. 인스턴스 구성 선택(Select instance configuration)에서 서비스 인스턴스의 CPU 유닛과 메모리(GB) 조합을 선택합니다.

   서비스가 생성 중일 때는 상태가 Creating(생성)에서 Running(실행)으로 변경됩니다.
9. 서비스 실행을 시작한 후 서비스의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Copy Service URL(서비스 URL 복사)을 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 정보는 App Runner 개발자 안내서에서 서비스 생성이 실패하는 경우를 참조하세요.

App Runner 서비스 관리

App Runner 서비스를 생성한 후 AWS Explorer 창을 사용하여 다음 활동을 수행하여 이를 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 비활성화하고 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 Explorer가 열려 있지 않은 경우 이를 엽니다.
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 Explorer가 열려 있지 않은 경우 이를 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 Explorer가 열려 있지 않은 경우 이를 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 Logs Logs를 사용하여 App Runner와 같은 서비스에 대한 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 단추로 클릭하고 로그 스트림 보기(View Log Streams)를 선택합니다.
4. 명령 창에서 그룹의 로그 스트림을 선택합니다.

   AWS Cloud9 IDE는 스트림을 구성하는 로그 이벤트 목록을 표시합니다. 이전 이벤트 또는 최신 이벤트를 편집기에 로드하도록 선택할 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 열려 있지 않은 경우 이를 엽니다.
2. App Runner를 확장하여 서비스 목록을 봅니다.
3. 서비스를 마우스 오른쪽 단추로 클릭하고 서비스 삭제(Delete Service)를 선택합니다.
4. AWS 도구 키트 명령 창에서 delete를 입력한 다음 Enter 키를 눌러 확인합니다.

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

AWS 도구 키트를 사용한 AWS CloudFormation 스택 작업

AWS 도구 키트는 AWS CloudFormation 스택을 지원합니다. AWS Toolkit을 사용하여 AWS CloudFormation 스택을 삭제할 수 있습니다.

AWS CloudFormation 스택 삭제

AWS 도구 키트를 사용하여 AWS CloudFormation 스택을 보고 삭제할 수 있습니다.

필수 조건

- AWS Cloud9 환경에서 사용하는 자격 증명에 AWS CloudFormation 서비스에 대한 적절한 읽기/쓰기 액세스 권한이 포함되어 있는지 확인합니다. AWS 탐색기의 CloudFormation에서 “Error loading CloudFormation resources(Cloud Formation 리소스를 로드하는 동안 오류가 발생했습니다)”와 유사

한 메시지가 표시되면 해당 자격 증명에 연결된 권한을 확인합니다. 권한을 변경한 경우 AWS 탐색기에 적용되는 데 몇 분 정도 걸립니다.

AWS CloudFormation 스택을 삭제하려면

1. AWS 탐색기에서 삭제할 AWS CloudFormation 스택의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 엽니다.
2. Delete CloudFormation Stack(CloudFormation 스택 삭제)을 선택합니다.
3. 나타나는 메시지에서 Yes(예)를 선택하여 삭제를 확인합니다.

스택이 삭제되면 더 이상 AWS 탐색기에 나열되지 않습니다.

AWS 도구 키트를 사용한 CloudWatch Logs 작업

Amazon CloudWatch Logs를 사용하여 확장성이 뛰어난 단일 서비스에서 사용하는 모든 시스템, 애플리케이션 및 AWS 서비스 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 도구 키트를 사용하여 AWS 계정의 CloudWatch Logs를 사용하는 방법에 대해 설명합니다.

주제

- AWS 도구 키트를 사용하여 CloudWatch 로그 그룹 및 로그 스트림 보기
- AWS 도구 키트를 사용한 로그 스트림의 CloudWatch 로그 이벤트 작업

AWS 도구 키트를 사용하여 CloudWatch 로그 그룹 및 로그 스트림 보기

로그 스트림은 동일한 소스를 공유하는 로그 이벤트 시퀀스입니다. CloudWatch Logs에서 각 별도의 로그 소스가 별도의 로그 스트림을 구성합니다.

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

자세한 내용은 Amazon CloudWatch 사용 설명서에서 로그 그룹 및 로그 스트림 작업을 참조하세요.

주제

CloudWatch Logs 노드를 사용하여 로그 그룹 및 로그 스트림 보기

1. AWS Explorer가 열려 있지 않은 경우 이를 엽니다.
2. [CloudWatch Logs] 노드를 클릭하여 로그 그룹 목록을 확장합니다.

   현재 AWS 리전의 로그 그룹이 CloudWatch Logs 노드 아래에 표시됩니다.
3. 특정 로그 그룹의 로그 스트림을 보려면 로그 그룹 이름의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 View Log Streams(로그 스트림 보기)를 선택합니다.
4. 로그 그룹의 콘텐츠가 [로그 스트림 선택(Select a log stream)] 머리글 아래에 표시됩니다.

   목록에서 특정 스트림을 선택하거나 필드에 텍스트를 입력하여 스트림을 필터링할 수 있습니다.

   스트림을 선택하면 해당 스트림의 이벤트가 IDE의 Log Streams(로그 스트림) 창에 표시됩니다. 각 스트림의 로그 이벤트와 상호 작용하는 방법에 대한 자세한 내용은 CloudWatch 로그 이벤트 작업 섹션을 참조하세요.

AWS 도구 키트를 사용한 로그 스트림의 CloudWatch 로그 이벤트 작업

로그 스트림 창을 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 스트림을 열 때 [로그 스트림(Log Stream)] 창에 해당 스트림의 로그 이벤트 시퀀스가 표시됩니다.

1. 보려는 로그 스트림을 찾으려면 Log Stream(로그 스트림) 창을 엽니다. 자세한 내용은 CloudWatch 로그 그룹 및 로그 스트림 보기 섹션을 참조하세요.

   이벤트가 나열된 각 줄에는 기록된 시점을 표시하는 타임스탬프가 지정됩니다.
2. 다음 옵션을 사용하여 스트림의 이벤트에 대한 정보를 보고 복사할 수 있습니다.

- View events by time(시간별 이벤트 보기): Load newer events(더 새로운 이벤트 로드) 또는 Load older events(더 이전 이벤트 로드)를 선택하여 최신 및 이전 로그 이벤트를 표시합니다.

  > **Note**
  >
  > Log Stream(로그 스트림) 편집기는 처음에 가장 최근의 10,000개 줄의 로그 이벤트 또는 1MB의 로그 데이터 중 더 작은 항목의 배치를 로드합니다. [더 새로운 이벤트 로드(Load newer events)]를 선택하면 편집기는 마지막 배치가 로깅된 후 기록된 이벤트를 표시합니다. [더 이전 이벤트 로드(Load older events)]를 선택하면 편집기는 현재 표시된 이벤트보다 먼저 발생한 이벤트의 배치를 표시합니다.

- Copy log events(로그 이벤트 복사): 복사할 이벤트를 선택한 다음 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 메뉴에서 Copy(복사)를 선택합니다.
- Copy the log stream's name(로그 스트림의 이름 복사): Log Stream(로그 스트림) 창의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Copy Log Stream Name(로그 스트림 이름 복사)을 선택합니다.

로그 스트림 편집기의 콘텐츠를 로컬 파일에 저장

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

> **Note**
>
> 이 옵션을 사용하면 로그 스트림 편집기에 현재 표시된 로그 이벤트만 파일에 저장할 수 있습니다. 예를 들어 로그 스트림의 총 크기가 5MB이고 편집기에 2MB만 로드되었다고 가정해보겠습니다. 저장된 파일은 2MB의 로그 데이터만 포함합니다. 저장할 데이터를 더 표시하려면 편집기에서 [더 새로운 이벤트 로드(Load newer events)] 또는 [더 이전 이벤트 로드(Load older events)]를 선택합니다.

1. 복사할 로그 스트림을 찾으려면 [로그 스트림(Log Streams)] 창을 엽니다(CloudWatch 로그 그룹 및 로그 스트림 보기 참조).
2. Log Stream(로그 스트림) 창 탭의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Save Current Log Content to File(현재 로그 콘텐츠를 파일에 저장)을 선택합니다.
3. 이 대화 상자를 사용하여 로그 파일의 다운로드 폴더를 선택하거나 생성한 다음 Save(저장)를 선택합니다.

AWS 도구 키트를 사용한 AWS Lambda 함수 작업

AWS 도구 키트는 AWS Lambda 함수를 지원합니다. AWS 도구 키트는 AWS Cloud9의 Lambda 플러그인에서 제공하는 기능을 대체합니다. AWS 도구 키트를 사용하여 서버리스 애플리케이션 일부인 Lambda 함수의 코드를 작성할 수 있습니다. 또한 Lambda 함수를 로컬로 호출하거나 AWS에서 호출할 수 있습니다.

Lambda는 사용자 지정 코드로 생성하거나 다양한 AWS 서비스에서 생성된 이벤트에 대한 응답으로 코드를 실행하는 완벽하게 관리되는 컴퓨팅 서비스입니다. 대표적인 예는 Amazon Simple Storage Service(S3), Amazon DynamoDB, Amazon Kinesis, Amazon Simple Notification Service(SNS)와 Amazon Cognito입니다.

> **Important**
>
> Serverless Application Model(SAM)에서 제공하는 리소스를 사용하는 Lambda 애플리케이션을 빌드하려면 AWS 도구 키트를 사용한 AWS 서버리스 애플리케이션 작업 섹션을 참조하세요.

주제

원격 Lambda 함수 호출

AWS 도구 키트를 사용하면 다양한 방법으로 AWS Lambda 함수와 상호 작용할 수 있습니다.

Lambda에 대한 자세한 내용은 AWS Lambda 개발자 가이드를 참조하세요.

> **Note**
>
> AWS Management Console을 사용하는 등의 방식으로 이미 Lambda 함수를 만들었다고 가정하겠습니다. AWS 도구 키트에서 이러한 함수를 호출할 수 있습니다. AWS 도구 키트를 사용하여 AWS Lambda에 배포할 수 있는 새 함수를 생성하려면 먼저 서버리스 애플리케이션을 생성해야 합니다.

필수 조건

- 구성한 자격 증명에 AWS Lambda 서비스에 대한 적절한 읽기/쓰기 액세스 권한이 포함되어 있는지 확인합니다. AWS 탐색기의 [Lambda] 아래에 “Error loading Lambda resources(Lambda 리소스를 로드하는 동안 오류가 발생했습니다)”와 유사한 메시지가 표시되면 해당 자격 증명에 연결된 권한을 확인합니다. 권한을 변경한 경우 AWS 도구 키트에서 AWS 탐색기에 적용되는 데 몇 분 정도 걸립니다.

Lambda 함수 호출

> **Important**
>
> AWS 도구 키트를 사용하여 API 메서드를 호출하면 리소스가 변경될 수 있으며 이 변경은 취소할 수 없습니다. 예를 들어 POST 메서드를 호출하면 호출이 성공할 경우 API의 리소스가 업데이트됩니다.

AWS도구 키트를 사용하여 AWS에서 Lambda 함수를 호출할 수 있습니다.

1. AWS 탐색기에서 호출하려는 Lambda 함수의 이름을 선택한 다음 컨텍스트 메뉴를 엽니다.
2. AWS에서 호출(Invoke on AWS)을 선택합니다.
3. [함수 호출(Invoke function)] 창이 열리면 Lambda 함수에 필요한 페이로드의 옵션을 선택합니다. (페이로드는 Lambda 함수에 입력으로 제공하려는 JSON입니다.) Browse(찾아보기)를 선택하여 페이로드로 사용할 파일을 선택하거나 드롭다운 필드를 사용하여 페이로드에 대한 템플릿을 선택합니다. 이 경우 Lambda 함수는 텍스트 상자에 표시된 것처럼 문자열인 입력으로 표시될 있습니다.

Invoke)(호출)을 선택하여 Lambda를 호출하고 페이로드를 전달합니다.

Lambda 함수의 출력은 AWS Lambda 탭에 표시됩니다.

Lambda 함수 다운로드, 업로드 및 삭제

AWS Toolkit은 AWS Cloud9 IDE에서 Lambda 함수를 가져오고 업로드하기 위한 옵션을 제공합니다.

Lambda 함수 다운로드

Lambda 함수를 다운로드하면 AWS 클라우드에서 함수를 설명하는 프로젝트를 다운로드하고 AWS Cloud9 IDE에서 사용할 수도 있습니다.

Lambda 함수를 다운로드하려면

1. AWS 탐색기의 Lambda 노드에서 함수의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Download(다운로드)를 선택합니다.
2. 새 프로젝트의 WorkSpace 폴더를 선택하라는 메시지가 나타나면 다음 중 하나를 수행할 수 있습니다.

   - 제안된 폴더를 선택하여 Lambda 프로젝트와 같은 이름의 하위 폴더를 만듭니다.
   - [다른 폴더 선택(Select a different folder)]을 선택하여 프로젝트 하위 폴더의 다른 상위 폴더를 찾아 선택하는 대화 상자를 엽니다.

   IDE가 새 편집기 창을 엽니다.

실행 및 디버깅을 위해 다운로드한 Lambda 함수 구성

다운로드한 Lambda 함수를 서버리스 애플리케이션으로 실행하고 디버그하려면 시작 구성을 `launch.json` 파일에 정의해야 합니다. AWS Management Console에서 생성된 Lambda 함수는 시작 구성에 포함되지 않을 수 있습니다. 따라서 수동으로 추가해야 할 수 있습니다.

시작 구성에 Lambda 함수를 추가하려면

1. Lambda 함수를 다운로드한 후 Environment(환경) 창을 열어 해당 폴더와 파일을 봅니다.
2. 그런 다음 Lambda 함수가 `/home/ec2-user/.c9/launch.json` 파일에 포함되어 있는지 확인합니다. 없는 경우 다음을 수행하여 함수 코드에 CodeLens 링크를 추가합니다.

   1. Lambda 함수를 정의하는 소스 코드 파일(예:`.js` 또는 `.py` 파일)을 엽니다. 그런 다음 Lambda 함수를 `launch.json` 파일에 추가하는 데 사용할 수 있는 CodeLens 링크가 있는지 확인합니다. CodeLens는 함수 위에 나타나며 `Add Debug Config` 링크를 포함합니다.
   2. IDE 왼쪽에서 Go(이동)(돋보기 아이콘)을 선택하고 'sam hint'를 입력하여 `AWS: Toggle SAM hints in source files` 명령을 표시합니다. 명령을 선택하여 실행합니다.
   3. Lambda 소스 코드 파일을 닫은 다음 다시 엽니다.

4. 파일을 다시 연 후 소스 코드에서 CodeLens를 사용할 수 있는 경우 `Add Debug Config`를 선택하여 시작 구성을 추가합니다.

3. SAM 힌트 옵션을 전환한 후에도 CodeLens를 추가할 수 없는 경우 다음을 수행하여 시작 구성을 추가합니다.

   1. IDE 왼쪽에서 이동(Go)(돋보기 아이콘)을 선택하고 'config'를 입력하여 `AWS: SAM Debug Configuration Editor` 명령을 표시합니다. 명령을 선택하여 실행합니다.
   2. SAM Debug Configuration Editor(SAM 디버그 구성 편집기)가 표시됩니다. 이 편집기를 사용하여 시작 구성 속성을 정의할 수 있습니다. 자세한 내용은 SAM 템플릿을 사용하여 서버리스 애플리케이션 실행 및 디버깅의 configuring launch properties 단계를 참조하세요.

      > **Note**
      >
      > Lambda 함수에 SAM 애플리케이션용 `template.yaml`이 없다면 이를 추가해야 합니다. 자세한 내용은 AWS SAM 템플릿 생성을 참조하세요.

   3. 편집기에서 필수 구성 정보의 입력을 마치면 시작 구성이 launch.json 파일에 추가됩니다.

Lambda 함수에 대한 시작 구성을 정의한 후 다음을 수행하여 해당 시작 구성을 실행할 수 있습니다.

1. IDE 상단에서 자동(Auto) 옆에 있는 화살표를 선택하고 해당 시작 구성을 선택합니다.
2. 그런 다음 실행(Run)을 선택합니다

Lambda 함수 업로드

로컬 코드로 기존 Lambda 함수를 업데이트할 수 있습니다. 이 방법으로 코드를 업데이트하면 AWS Serverless Application Model CLI가 배포에 사용되지 않으며 AWS CloudFormation 스택이 생성되지 않습니다. 이렇게 하면 Lambda에서 지원하는 모든 런타임을 사용하여 Lambda 함수를 업로드할 수 있습니다.

AWS 도구 키트를 사용하여 Lambda 함수를 업로드하기 위한 몇 가지 인터페이스 옵션이 있습니다.

환경(Environment) 창 또는 명령(Command) 창에서 업로드

1. 프로젝트 파일의 Environment(환경) 창에서 업로드하려는 Lambda 애플리케이션용 `template.yaml`의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Upload Lambda(Lambda 업로드)를 선택합니다.

또는 Ctrl+P를 눌러 바로 가기(Go to Anything) 창을 열고 'lambda'를 입력하여 AWS Upload Lambda 명령에 액세스합니다. 그런 다음 이를 선택하여 업로드 프로세스를 시작합니다.

2. 다음으로 업로드할 AWS 리전을 선택합니다.
3. 이제 Lambda 함수를 업로드하기 위한 옵션을 선택합니다.

   .zip 아카이브 업로드

   1. 메뉴에서 [ZIP 아카이브(ZIP Archive)]를 선택합니다.
   2. AWS Cloud9 파일 시스템에서 .zip 파일을 선택하고 [열기(Open)]를 선택합니다.

   디렉터리를 있는 그대로 업로드

   1. 메뉴에서 [디렉터리(Directory)]를 선택합니다.
   2. AWS Cloud9 파일 시스템에서 디렉터리를 선택하고 [열기(Open)]를 선택합니다.
4. 이벤트를 처리하는 Lambda 함수 핸들러를 지정합니다. 함수가 호출되면 Lambda가 이 핸들러 메서드를 실행합니다.

   > **Note**
   >
   > Lambda 함수를 선택할 때는 표시된 목록에서 선택할 수 있습니다. 어떤 함수를 선택해야 할지 모르는 경우 도구 키트에서 사용할 수 있는 Lambda 함수의 Amazon 리소스 번호(ARN)를 입력할 수 있습니다.

   이 코드를 Lambda 함수의 최신 버전으로 게시할지 여부를 묻는 대화 상자가 표시됩니다. 예(Yes)를 선택하여 게시를 확인합니다.

   > **Note**
   >
   > 상위 폴더의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Upload Lambda(Lambda 업로드)를 선택하여 Lambda 애플리케이션을 업로드할 수도 있습니다. 업로드 시 상위 폴더는 자동으로 선택됩니다.

AWS 탐색기에서 업로드

1. AWS 탐색기에서 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고(마우스 오른쪽 버튼 클릭), 가져올 Lambda 함수의 이름을 선택합니다.
2. Lambda 업로드(Upload Lambda)를 선택합니다.
3. Lambda 함수를 업로드하는 세 가지 옵션 중에서 선택합니다.

   미리 만든 .zip 아카이브 업로드

   1. 메뉴에서 [ZIP 아카이브(ZIP Archive)]를 선택합니다.
   2. AWS Cloud9 파일 시스템에서 .zip 파일을 선택하고 [열기(Open)]를 선택합니다.
   3. 모달 대화 상자로 업로드를 확인합니다. 이렇게 하면 .zip 파일이 업로드되고 배포 후 Lambda가 즉시 업데이트됩니다.

   디렉터리를 있는 그대로 업로드

   1. 메뉴에서 [디렉터리(Directory)]를 선택합니다.
   2. AWS Cloud9 파일 시스템에서 디렉터리를 선택하고 [열기(Open)]를 선택합니다.
   3. 디렉터리를 빌드할지 묻는 메시지가 나타나면 [아니요(No)]를 선택합니다.
   4. 모달 대화 상자로 업로드를 확인합니다. 이렇게 하면 디렉터리가 있는 그대로 업로드되고 배포 후 Lambda가 즉시 업데이트됩니다.

   디렉터리 구축 및 업로드

   1. 메뉴에서 [디렉터리(Directory)]를 선택합니다.
   2. AWS Cloud9 파일 시스템에서 디렉터리를 선택하고 [열기(Open)]를 선택합니다.
   3. 디렉터리를 빌드할지 묻는 메시지가 나타나면 [예(Yes)]를 선택합니다.
   4. 모달 대화 상자로 업로드를 확인합니다. 이렇게 하면 AWS SAM CLI `sam build` 명령을 사용하여 디렉터리에 코드가 빌드되고 배포 후 Lambda가 즉시 업데이트됩니다.

원격 액세스를 위한 Lambda 함수 배포

로컬 함수를 서버리스 SAM 애플리케이션으로 배포하여 원격으로 사용 가능하게 할 수 있습니다.

Lambda 함수를 SAM 애플리케이션으로 배포하려면

1. AWS 탐색기 창에서 Lambda 노드의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Deploy SAM Application(SAM 애플리케이션 배포)를 선택합니다.
2. 명령 창에서 함수를 서버리스 애플리케이션으로 정의하는 YAML 템플릿을 선택합니다.
3. 그런 다음 Lambda 배포에 사용할 Amazon S3 버킷을 선택합니다. 배포용 버킷을 생성하도록 선택할 수도 있습니다.
4. 이제 배포할 AWS CloudFormation 스택의 이름을 입력합니다. 기존 스택을 지정하면 명령이 스택을 업데이트합니다. 새 스택을 지정하면 명령이 스택을 생성합니다.

   스택 이름을 입력하면 Lambda 함수가 SAM 애플리케이션으로 배포되기 시작합니다. 배포에 성공하면 SAM Lambda 애플리케이션을 원격으로 사용할 수 있습니다. 이렇게 하면 SAM Lambda 애플리케이션을 다른 AWS Cloud9 개발 환경에서 다운로드하거나 호출할 수 있습니다.

처음부터 Lambda 함수를 생성하려면 AWS를 사용하여 서버리스 애플리케이션 만들기 단계를 따르는 것이 좋습니다.

Lambda 함수 삭제

동일한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 사용하여 Lambda 함수를 삭제할 수도 있습니다.

> **Warning**
>
> 이 절차를 사용하여 AWS CloudFormation과 연결된 Lambda 함수를 삭제해선 안 됩니다. 예를 들어 이 설명서 앞부분에서 서버리스 애플리케이션을 만들 때 생성한 Lambda 함수를 삭제해선 안 됩니다. 이러한 함수는 AWS CloudFormation 스택을 통해 삭제해야 합니다.

1. AWS 탐색기에서 삭제할 Lambda 함수의 이름을 선택한 다음 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 엽니다.
2. 삭제를 선택합니다.
3. 나타나는 메시지에서 Yes(예)를 선택하여 삭제를 확인합니다.

함수가 삭제되면 더 이상 AWS 탐색기 보기에 나열되지 않습니다.

리소스 작업

AWS Explorer에 기본적으로 나열되는 AWS 서비스에 액세스하는 것 외에도 리소스(Resources)로 이동하여 수백 개의 리소스 중에서 선택하여 인터페이스에 추가할 수도 있습니다. AWS에서 리소스란 작업할 수 있는 엔터티입니다. 추가된 리소스 중 일부는 Amazon AppFlow, Amazon Kinesis Data Streams,AWS IAM 역할, Amazon VPC 및 Amazon CloudFront 배포입니다.

사용 가능한 리소스를 보려면 리소스(Resources)로 이동하고 리소스 유형을 확장하여 해당 유형에 사용 가능한 리소스를 나열합니다. 예를 들어 `AWS::Lambda::Function` 리소스 유형을 선택하면 다양한 기능, 해당 속성 및 특성을 정의하는 리소스에 액세스할 수 있습니다.

리소스 유형을 리소스(Resources)에 추가한 후 다음과 같은 방법으로 해당 리소스와 상호 작용할 수 있습니다.

- 이 리소스 유형에 대해 현재 AWS 리전에서 사용할 수 있는 기존 리소스 목록을 봅니다.
- 리소스를 설명하는 JSON 파일의 읽기 전용 버전을 봅니다.
- 리소스의 리소스 식별자를 복사합니다.
- 리소스 모델링을 위한 리소스 유형 및 스키마(JSON 및 YAML 형식)의 목적을 설명하는 AWS 설명서를 봅니다.

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 `AWS::Lambda::Function` 리소스에 액세스하기 위해 Lambda 권한이 필요합니다.

서비스 리소스에 대한 권한 외에도 IAM 엔터티에는 AWS 도구 키트가 AWS Cloud Control API 작업을 호출하도록 허용하는 권한이 필요합니다. Cloud Control API 작업을 통해 IAM 사용자 또는 역할이 원격 리소스에 액세스하고 업데이트할 수 있습니다.

AWS 관리형 정책인 PowerUserAccess를 도구 키트 인터페이스를 사용하여 이러한 API 작업을 호출하는 IAM 엔터티에 연결하면 권한을 빠르게 부여할 수 있습니다. 이 관리형 정책은 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 리소스와 상호 작용

1. AWS Explorer에서 리소스(Resources)를 선택합니다.

   리소스 유형 목록이 리소스(Resources) 노드에 표시됩니다.
2. 리소스 유형에 대한 템플릿을 정의하는 구문을 설명하는 설명서가 있습니다. 이 설명서에 액세스하려면 해당 리소스 유형의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 설명서 보기를 선택합니다.

   > **Note**
   >
   > 설명서 페이지에 액세스할 수 있도록 브라우저의 팝업 차단기를 해제하는 메시지가 표시될 수 있습니다.
3. 리소스 유형에 대해 이미 존재하는 리소스를 보려면 해당 유형의 항목을 확장합니다.

   사용 가능한 리소스 목록이 해당 리소스 유형 아래에 표시됩니다.
4. 특정 리소스와 상호 작용하려면 리소스 이름의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 다음 옵션 중 하나를 선택합니다.

   - 식별자 복사: 특정 리소스의 식별자를 클립보드에 복사합니다. 예를 들어 `TableName` 속성을 사용하여 `AWS::DynamoDB::Table` 리소스를 식별할 수 있습니다.
   - 미리 보기: 리소스를 설명하는 JSON 형식 템플릿의 읽기 전용 버전을 봅니다.

AWS 도구 키트를 사용한 Amazon S3 작업

다음 주제에서는 AWS 도구 키트를 사용하여 AWS 계정의 Amazon S3 버킷 및 객체를 사용하는 방법에 대해 설명합니다.

주제

- Amazon S3 버킷 작업
- Amazon S3 객체 작업

Amazon S3 버킷 작업

Amazon S3에 저장한 모든 객체는 버킷에 존재합니다. 디렉토리로 파일 시스템 내 파일을 그룹화하듯 버킷으로 관련 객체를 그룹화할 수 있습니다.

주제

Amazon S3 버킷 생성

1. AWS 탐색기에서 S3 노드의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 연 다음 Create Bucket(버킷 생성)을 선택합니다.
2. [버킷 이름(Bucket Name)] 필드에 버킷의 유효한 이름을 입력합니다. Enter 키를 눌러 확인합니다.

   새 버킷이 S3 노드 아래에 표시됩니다.

   > **참고**
   >
   > S3 버킷은 공개적으로 액세스할 수 URL로 사용할 수 있으므로, 선택하는 버킷 이름은 전 세계적으로 고유해야 합니다. 다른 계정이 이미 선택한 이름을 가진 버킷을 생성한 경우 다른 이름을 사용해야 합니다.
   > 버킷을 만들 수 없다면 Output(출력) 탭에서 AWS Toolkit Logs(도구 키트 로그)를 확인하세요. 예를 들어 이미 사용 중인 버킷 이름을 사용하면 `BucketAlreadyExists` 오류가 발생합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서.의 버킷 규제 및 제한을 참조하세요.

   버킷이 생성된 후에는 버킷의 이름과 Amazon 리소스 이름(ARN)을 클립보드로 복사할 수 있습니다. 버킷 항목에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 관련 옵션을 선택합니다.

Amazon S3 버킷에 폴더 추가

객체를 폴더에 그룹화하여 버킷의 콘텐츠를 구성합니다. 다른 폴더 내에 폴더를 만들 수도 있습니다.

1. AWS 탐색기에서 [S3] 노드를 선택하여 버킷 목록을 봅니다.
2. 버킷 또는 폴더에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 연 다음 Create Folder(폴더 생성)를 선택합니다.
3. [폴더 이름(Folder Name)]을 입력한 다음 Enter 키를 누릅니다.

이제 새 폴더가 AWS 탐색기 창의 선택한 버킷 및 폴더 아래에 표시됩니다.

Amazon S3 버킷 삭제

버킷을 삭제하면 버킷에 포함된 폴더와 객체도 삭제됩니다. 버킷을 삭제하기 전에 이 작업을 정말로 수행할지 묻는 메시지가 나타납니다.

> **Note**
>
> 전체 버킷이 아니라 폴더만 삭제하려면 AWS Management Console을 사용합니다.

1. AWS 탐색기에서 [S3] 노드를 선택하여 버킷 목록을 확장합니다.
2. 삭제할 버킷의 컨텍스트 메뉴를 연 다음 [삭제(Delete)]를 선택합니다.
3. 버킷의 이름을 입력하여 삭제 의사를 확인한 다음 Enter 키를 누릅니다.

> **Note**
>
> 버킷에 객체가 포함되어 있으면 삭제하기 전에 버킷이 비워집니다. 수천 개 객체의 모든 버전을 삭제해야 하는 경우 다소 시간이 걸릴 수 있습니다. 삭제 프로세스가 완료되면 알림이 표시됩니다.

Amazon S3 항목의 표시 구성

많은 수의 Amazon S3 객체 또는 폴더로 작업하는 경우 한 번에 표시되는 수를 지정하는 것이 도움이 됩니다. 최대 개수의 객체 또는 폴더가 표시되어 있는 경우 [추가로 로드(Load More)]를 선택하여 다음 배치를 표시할 수 있습니다.

1. 메뉴 모음에서 [AWS Cloud9], [기본 설정(Preferences)]을 선택합니다.
2. 기본 설정(Preferences) 창에서 프로젝트 설정(Project Settings)을 확장하고 확장(EXTENSIONS) 섹션으로 이동하여 AWS 구성(AWS Configuration)을 선택합니다.
3. AWS 구성(AWS Configuration) 창에서 AWS > S3: 페이지당 최대 항목 수(AWS > S3: Max Items Per Page) 설정으로 이동합니다.
4. 추가 항목을 로드하도록 선택하기 전에, 표시하려는 S3 항목 수로 기본값을 변경합니다.

> **Note**
>
> 허용되는 값의 범위는 3에서 1,000 사이입니다. 이 설정은 한 번에 표시되는 객체 또는 폴더 수에만 적용됩니다. 생성한 모든 버킷이 한 번에 표시됩니다. 기본적으로 AWS 계정 각각에 대해 최대 100개의 버킷을 만들 수 있습니다.

Amazon S3 객체 작업

객체는 Amazon S3에 저장되는 기본 개체입니다. 객체는 객체 데이터와 메타데이터로 구성됩니다.

주제

- Amazon S3 버킷에 파일 업로드
- Amazon S3 객체 다운로드
- Amazon S3 객체 삭제
- Amazon S3 객체에 대해 미리 서명된 URL 생성

Amazon S3 버킷에 파일 업로드

도구 키트 인터페이스 또는 명령을 사용하여 버킷에 파일을 업로드할 수 있습니다.

두 방법 중 무엇을 사용하든, 사용자의 환경에서 파일을 업로드하고 AWS 클라우드에 S3 객체로 저장할 수 있습니다. 버킷이나, 버킷의 콘텐츠를 구성하는 폴더에 파일을 업로드할 수 있습니다.

인터페이스를 사용하여 S3 버킷에 파일 업로드

1. AWS 탐색기에서 [S3] 노드를 선택하여 버킷 목록을 봅니다.
2. 버킷 또는 해당 버킷의 폴더에 대한 컨텍스트 메뉴를 연(마우스 오른쪽 버튼 클릭) 다음 [파일 업로드(Upload File)]를 선택합니다.

   > **참고**
   >
   > S3 객체의 컨텍스트 메뉴를 연(마우스 오른쪽 버튼 클릭) 경우 [상위에 업로드(Upload to Parent)]를 선택할 수 있습니다. 이렇게 하면 선택한 파일이 들어 있는 폴더나 버킷에 파일을 추가할 수 있습니다.

3. 환경의 파일 관리자를 사용하여 파일을 선택한 다음 [업로드(Upload)]를 선택합니다.

선택한 파일이 S3 객체로 버킷 또는 폴더에 업로드됩니다. 각 객체의 항목은 저장된 객체의 크기와 업로드 기간을 설명합니다. 객체의 목록을 일시 중지하여 마지막으로 수정한 경로, 크기 및 시간을 볼 수 있습니다.

명령을 사용하여 S3 버킷에 현재 파일 업로드

1. 업로드할 파일을 선택하려면 해당 파일의 탭을 선택합니다.
2. Ctrl+P를 눌러 [명령(Commands)] 창을 표시합니다.
3. [바로 가기(Go To Anything)] `upload file`이라는 문구를 입력하기 시작하여 `AWS: Upload File` 명령을 표시합니다. 명령이 표시되면 선택합니다.
4. [1단계: 업로드할 파일 선택(Step 1: Select a file to upload)]에서 선택한 파일을 선택하거나 다른 파일을 찾아볼 수 있습니다.
5. [2단계: 업로드할 S3 버킷 선택(Step 2: Select an S3 bucket to upload to)의 목록에서 버킷을 선택합니다.

   선택한 파일이 S3 객체로 버킷 또는 폴더에 업로드됩니다. 각 객체의 항목은 저장된 객체의 크기와 업로드 기간을 설명합니다. 객체의 목록을 일시 중지하여 마지막으로 수정한 경로, 크기 및 시간을 볼 수 있습니다.

Amazon S3 객체 다운로드

AWS 클라우드에서 AWS Cloud9 환경의 폴더로 Amazon S3 버킷의 객체를 다운로드할 수 있습니다.

1. AWS 탐색기에서 [S3] 노드를 선택하여 버킷 목록을 봅니다.
2. 버킷이나 버킷의 폴더에서 객체의 컨텍스트 메뉴를 연(마우스 오른쪽 버튼 클릭) 다음 [다른 이름으로 다운로드(Download As)]를 선택합니다.
3. 환경의 파일 관리자를 사용하여 대상 폴더를 선택하고 파일 이름을 입력한 다음 [다운로드(Download)]를 선택합니다.

파일을 다운로드한 후에는 AWS Cloud9에서 열 수 있습니다.

Amazon S3 객체 삭제

버전이 지정되지 않은 버킷에 있는 객체를 영구적으로 삭제할 수 있습니다. 하지만 버전 관리를 사용하는 버킷의 경우 삭제 요청이 해당 객체를 영구적으로 삭제하지 않습니다. 대신 Amazon S3가 버킷에

삭제 마커를 삽입합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서에서 객체 버전 삭제를 참조하세요.

1. AWS 탐색기에서 [S3] 노드를 선택하여 버킷 목록을 봅니다.
2. 버킷이나 버킷의 폴더에서 객체의 컨텍스트 메뉴를 연(마우스 오른쪽 버튼 클릭) 다음 [삭제(Delete)]를 선택합니다.
3. [삭제(Delete)]를 선택하여 삭제를 확인합니다.

Amazon S3 객체에 대해 미리 서명된 URL 생성

미리 서명된 URL을 통해 객체 소유자는 객체를 다운로드할 수 있는 제한된 시간 권한을 부여하여 프라이빗 Amazon S3 객체를 다른 사용자와 공유할 수 있습니다. 자세한 내용은 Amazon S3 사용 설명서의 미리 서명된 URL로 객체 공유를 참조하세요.

1. AWS 탐색기에서 [S3] 노드를 선택하여 버킷 목록을 봅니다.
2. 버킷이나 버킷의 폴더에서 객체를 마우스 오른쪽 버튼으로 클릭한 다음 미리 서명된 URL 생성을 선택합니다.
3. AWS 도구 키트 명령 창에서 URL을 사용하여 객체에 액세스할 수 있는 시간(분)을 입력합니다. Enter 키를 눌러 확인합니다.

   IDE 하단의 상태는 객체에 대해 미리 서명된 URL이 클립보드에 복사되었음을 확인합니다.

AWS 도구 키트를 사용한 AWS 서버리스 애플리케이션 작업

AWS 도구 키트는 서버리스 애플리케이션에 대한 지원을 제공합니다. AWS 도구 키트를 사용하면 AWS Lambda 함수를 포함하는 서버리스 애플리케이션을 만든 다음 해당 애플리케이션을 AWS CloudFormation 스택에 배포할 수 있습니다.

주제

- 서버리스 애플리케이션 만들기
- 서버리스 애플리케이션 실행 및 디버깅
- 서버리스 애플리케이션 동기화
- AWS Toolkit 코드 렌즈 활성화
- AWS 클라우드에서 서버리스 애플리케이션 삭제
- 서버리스 애플리케이션 디버깅을 위한 구성 옵션

서버리스 애플리케이션 만들기

이 예에서는 AWS 도구 키트를 사용하여 서버리스 애플리케이션을 만드는 방법을 보여 줍니다. 서버리스 애플리케이션을 실행하고 디버깅하는 방법에 대한 자세한 내용은 서버리스 애플리케이션 실행 및 디버깅 섹션을 참조하세요.

서버리스 애플리케이션을 만드는 데 필요한 사전 조건에는 AWS SAM CLI 및 AWS CLI가 포함됩니다. 이들은 AWS Cloud9에 포함되어 있습니다. AWS SAM CLI가 설치되지 않았거나 오래된 경우 설치 또는 업그레이드를 실행해야 할 수 있습니다. AWS SAM CLI 설치 방법에 대한 지침은 AWS SAM CLI 설치 섹션을 참조하고, AWS SAM CLI 업그레이드 방법에 대한 지침은 AWS SAM CLI 업그레이드 섹션을 참조하십시오.

AWS를 사용하여 서버리스 애플리케이션 만들기

이 예에서는 AWS Serverless Application Model(AWS SAM)을 사용하여 AWS 도구 키트로 서버리스 애플리케이션을 만드는 방법을 보여 줍니다.

1. AWS 탐색기에서 Lambda 노드의 컨텍스트(마우스 오른쪽 버튼 클릭) 노드를 열고 Create Lambda SAM Application(Lambda SAM 애플리케이션 생성)을 선택합니다.

   > **Note**
   >
   > 또는 AWS: 탐색기 머리글의 맞은편에 있는 메뉴 아이콘을 선택하고 Create Lambda SAM Application(Lambda SAM 애플리케이션 생성)을 선택합니다.

2. SAM 애플리케이션의 런타임을 선택합니다. 이 예에서는 nodejs12.x를 선택합니다.

   > **Note**
   >
   > “(Image)”라는 표시가 있는 런타임 중 하나를 선택하면 애플리케이션의 패키지 유형이 `Image`가 됩니다. “(Image)”라는 표시가 없는 런타임 중 하나를 선택하면 애플리케이션의 유형이 `Zip`이 됩니다. `Image` 패키지 유형과 `Zip` 패키지 유형의 차이점에 대한 자세한 내용은 AWS Lambda 개발자 가이드에서 Lambda 배포 패키지를 참조하세요.

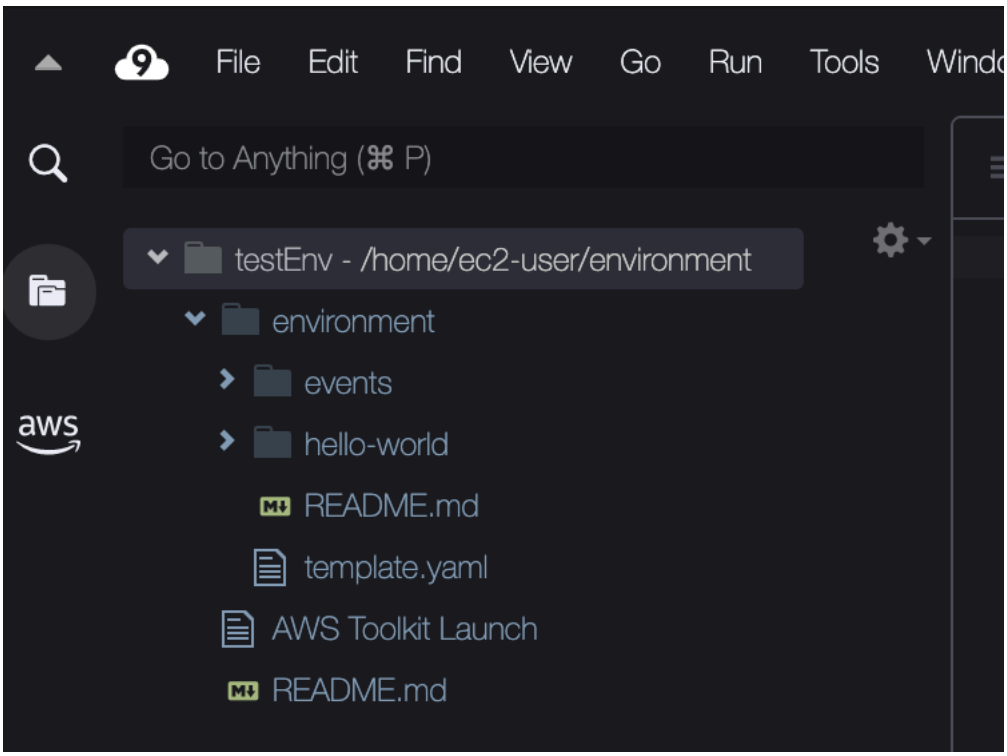
3. 서버리스 앱에 대해 다음 템플릿 중 하나를 선택합니다.

   - AWS SAM Hello World: 고전적인 ‘Hello World’ 메시지를 반환하는 Lambda 함수가 있는 기본 템플릿입니다.

- AWS Step Functions 샘플 앱: 주식 거래 워크플로를 실행하는 샘플 애플리케이션입니다. Step Functions는 관련된 Lambda 함수의 상호 작용을 오케스트레이션합니다.

4. 새 프로젝트의 위치를 선택합니다. 사용 가능한 기존 workspace 폴더가 있다면 폴더를 선택합니다. 그렇지 않다면 다른 폴더를 찾아봅니다. Select a different folder(다른 폴더 선택)를 선택하면 폴더 위치를 선택할 수 있는 대화 상자가 표시됩니다.
5. 새 애플리케이션의 이름을 입력합니다. `my-sam-app-nodejs`를 입력하세요. Enter 키를 누르면 AWS 도구 키트에서 프로젝트를 생성하는 데 몇 분 정도 걸립니다.

프로젝트가 생성되면 [환경(Environment)] 창에서 애플리케이션의 파일을 볼 수 있습니다. Explorer(탐색기) 창에서 파일을 찾습니다.



서버리스 애플리케이션 실행 및 디버깅

AWS 도구 키트를 사용하여 서버리스 애플리케이션을 디버그하고 개발 환경에서 로컬로 실행하는 방법을 구성할 수 있습니다. AWS Serverless Application Model(AWS SAM) 템플릿에 의해 정의된 서버리스 애플리케이션을 디버그할 수 있습니다. 이 템플릿은 간단한 YAML 구문을 사용하여 함수, API, 데이터베이스, 서버리스 애플리케이션을 구성하는 이벤트 소스 매핑 등의 리소스를 설명합니다.

AWS SAM 템플릿에 대한 자세한 내용은 AWS Serverless Application Model 개발자 가이드에서 AWS SAM 템플릿 구조를 참조하세요.

또는 SAM 템플릿에 커밋되지 않은 서버리스 애플리케이션을 빠르게 디버그할 수 있습니다.

인라인 작업을 사용하여 적격 AWS Lambda 함수를 식별함으로써 디버그 동작 구성을 시작입니다. SAM 템플릿에 의해 정의된 인프라를 사용하려면, 관련 YAML 형식 파일에서 인라인 작업을 사용합니다. 템플릿 없이 함수를 직접 테스트하려면 애플리케이션 파일의 Lambda 핸들러에 대한 컨텍스트 인식 링크를 사용합니다.

> **Note**
>
> 이 예에서는 JavaScript를 사용하는 애플리케이션을 디버그합니다. 하지만 AWS 도구 키트에서 제공되는 디버깅 기능을 다음 언어 및 런타임에 사용할 수 있습니다.
>
> - JavaScript – Node.js 10.x, 12.x, 14.x
> - Python – 3.7, 3.8, 3.9, 3.10(Python 2.7 및 3.6 서버리스 애플리케이션을 실행할 수 있지만 AWS Toolkit으로 디버깅되지 않음)
>
> 선택하는 언어는 컨텍스트 인식 링크에서 적합한 Lambda 핸들러를 나타내는 방법에도 영향을 미칩니다. 자세한 내용은 코드에서 직접 서버리스 함수 실행 및 디버깅 섹션을 참조하세요.

SAM 템플릿을 사용하여 서버리스 애플리케이션 실행 및 디버깅

SAM 템플릿을 사용하여 실행 및 디버그되는 애플리케이션의 경우, YAML 형식의 파일은 애플리케이션의 동작 및 애플리케이션에서 사용하는 리소스를 설명합니다. AWS 도구 키트를 사용하여 서버리스 애플리케이션을 만드는 경우 `template.yaml`이 프로젝트에 대해 자동으로 생성됩니다.

이 절차에서는 서버리스 애플리케이션 만들기에서 생성된 예제 애플리케이션을 사용합니다.

SAM 템플릿을 사용하여 서버리스 애플리케이션을 실행하고 디버그하려면

1. 서버리스 애플리케이션을 구성하는 애플리케이션 파일을 보려면 [환경(Environment)] 창으로 이동합니다.
2. 애플리케이션 폴더(예: my-sample-app)에서 `template.yaml` 파일을 엽니다.
3. `template.yaml`의 경우 Edit Launch Configuration(실행 구성 편집)을 선택합니다.

   새 편집기에 기본 속성의 디버깅 구성을 제공하는 `launch.json` 파일이 표시됩니다.
4. 다음 구성 속성의 값을 편집하거나 확인합니다.

- "name" – Run(실행) 보기의 Configuration(구성) 드롭다운 필드에 표시할 알아보기 쉬운 이름을 입력합니다.
- "target" – 값이 "template"인지 확인합니다. 이렇게 하면 SAM 템플릿이 디버그 세션의 진입점이 됩니다.
- "templatePath" - template.yaml 파일의 상대 경로 또는 절대 경로를 입력합니다.
- "logicalId" - 이 이름이 SAM 템플릿의 Resources(리소스) 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.

5. 디버그 구성이 만족스럽다면 launch.json을 저장합니다. 그런 다음 RUN(실행) 옆에 있는 녹색 '재생' 버튼을 선택하여 디버깅을 시작합니다.

> **Note**
>
> SAM 애플리케이션이 실행되지 않는 경우 [출력(Output)] 창에서 빌드되지 않은 Docker 이미지로 인해 오류가 발생했는지 확인합니다. 환경에서 디스크 공간을 확보해야 할 수 있습니다.
> 자세한 내용은 AWS Cloud9 환경에 디스크 공간이 충분하지 않아 AWS 툴킷에서 SAM 응용 프로그램을 로컬로 실행하는 중 오류가 발생했습니다. 섹션을 참조하세요.

디버깅 세션이 시작되면 DEBUG CONSOLE(콘솔 디버그) 패널에 디버깅 출력이 표시되고 Lambda 함수에서 반환된 모든 값이 표시됩니다. SAM 애플리케이션을 디버그하는 경우 Output(출력) 패널의 Output(출력) 채널로 AWS Toolkit(도구 키트)이 선택됩니다.

> **Note**
>
> Windows 사용자의 경우 이 프로세스 중에 Docker 탑재 오류가 발생하면 Docker Settings(Docker 설정)에서 공유 드라이브의 자격 증명을 새로 고쳐야 할 수 있습니다. Docker 탑재 오류는 다음과 유사합니다.
>
> ```
> Fetching lambci/lambda:nodejs10.x Docker container image......
> 2019-07-12 13:36:58 Mounting C:\Users\<username>\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 ...
```

코드에서 직접 서버리스 함수 실행 및 디버깅

AWS SAM 애플리케이션을 테스트할 때 Lambda 함수만 실행 및 디버그하도록 선택할 수 있습니다. SAM 템플릿으로 정의된 다른 리소스는 제외합니다. 이 방법에서는 인라인 작업을 사용하여 직접 호출할 수 있는 소스 코드에서 Lambda 함수 핸들러를 식별해야 합니다.

컨텍스트 인식 링크로 감지되는 Lambda 핸들러는 애플리케이션에 사용 중인 언어와 런타임에 따라 다릅니다.

언어/런타임	컨텍스트 인식 링크로 식별되는 Lambda 함수에 대한 조건
JavaScript(Node.js 10.x, 12.x 및 14.x)	이 함수에는 다음 기능도 포함됩니다. • 최대 세 개의 파라미터가 있는 내보낸 함수입니다. • WorkSpace 폴더 내의 상위 폴더에 `package.json` 파일이 있습니다.
Python (3.7, 3.8, 3.9, 3.10)	이 함수에는 다음 기능도 포함합니다. • 최상위 함수입니다. • WorkSpace 폴더 내의 상위 폴더에 `requirements.txt` 파일이 있습니다.

애플리케이션 코드에서 직접 서버리스 애플리케이션을 실행하고 디버그하려면

1. 서버리스 애플리케이션 파일을 보려면 편집기 옆의 폴더 아이콘을 선택하여 애플리케이션 폴더로 이동합니다.
2. 애플리케이션 폴더(예: my-sample-app)에서 함수 폴더(이 예의 경우 hello-world)를 확장하고 `app.js` 파일을 엽니다.

3. 적합한 Lambda 핸들러 함수를 식별하는 인라인 작업에서 `Add Debug Configuration`을 선택합니다. 디버그 구성 추가 옵션이 나타나지 않으면 코드 렌즈를 활성화해야 합니다. 코드 렌즈를 활성화하려면 the section called “AWS Toolkit 코드 렌즈 활성화” 섹션을 참조하세요.
4. SAM 애플리케이션을 실행하는 런타임을 선택합니다.
5. `launch.json` 파일의 편집기에서 다음 구성 속성의 값을 편집하거나 확인합니다.
   - `"name"` - 알아보기 쉬운 이름을 입력합니다.
   - `"target"` - Lambda 함수 핸들러가 직접 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.
6. 디버그 구성에 만족하면 [실행(RUN)] 옆에 있는 녹색 재생 화살표를 선택하여 디버깅을 시작합니다.

   디버깅 세션이 시작되면 DEBUG CONSOLE(콘솔 디버그) 패널에 디버깅 출력이 표시되고 Lambda 함수에서 반환된 모든 값이 표시됩니다. SAM 애플리케이션을 디버그할 때는 Output(출력) 패널의 Output(출력) 채널로 AWS Toolkit(도구 키트)이 선택됩니다.

   > **Note**
   >
   > 오류 메시지에 Docker가 언급된 경우 이 참고를 참조하세요.

로컬 Amazon API Gateway 리소스 실행 및 디버깅

`template.yaml`에 지정된 AWS SAM API Gateway 로컬 리소스를 실행하거나 디버그할 수 있습니다. 이렇게 하려면 `invokeTarget.target=api`를 사용하여 `type=aws-sam`의 AWS Cloud9 시작 구성을 실행해야 합니다.

> **Note**
>
> API Gateway는 두 가지 유형의 API를 지원합니다. 바로 REST와 HTTP API입니다. 그러나 AWS 도구 키트를 사용한 API Gateway 기능은 REST API만 지원합니다. HTTP API를 'API Gateway V2 API'라고 하기도 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(특히 .js, .cs 또는 .py 파일)로 이동하여 Lambda 핸들러를 마우스 포인터로 가리킨 다음 Add Debug Configuration(디버그 구성 추가)를 선택합니다. 디버그 구성 추가 옵션이 나타나지 않는다면 코드 렌즈를 활성화합니다. 코드 렌즈를 활성화하려면 the section called "AWS Toolkit 코드 렌즈 활성화" 섹션을 참조하세요. 그런 다음 메뉴에서 API 이벤트로 표시된 항목을 선택합니다.
   - 옵션 2: `launch.json`을 편집하고 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. Run(실행) 버튼 옆에 있는 드롭다운 메뉴에서 시작 구성(위의 예에서는 `myConfig`)을 선택합니다.

3. (선택 사항) Lambda 프로젝트 코드에 중단점을 추가합니다.
4. [실행(Run)] 버튼 옆에 있는 녹색 '재생' 버튼을 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 경로입니다(예: `"path": "/hello"`).

`invokeTarget.templatePath`에 의해 지정된 `template.yaml`에서 확인된 유효한 API 경로여야 합니다.

httpMethod

"delete," "get," "head," "options," "patch," "post 및 "put" 동사 중 하나를 사용합니다.

payload

요청에서 보낼 JSON 페이로드(HTTP 본문)로, lambda.payload 필드와 구조와 규칙이 같습니다.

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

`payload.json`은 JSON 페이로드를 인라인으로 지정합니다.

헤더

이름-값 페어의 선택적 맵입니다. 요청에 포함할 HTTP 헤더를 지정하는 데 사용합니다.

```
"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

(선택 사항) 이 문자열을 사용하여 요청의 `querystring`을 설정합니다(예: `"querystring": "abc=def&ghi=jkl"`).

aws

AWS 연결 정보를 제공하는 방법입니다. 자세한 내용은 서버리스 애플리케이션 디버깅을 위한 구성 옵션에서 AWS 연결(**aws**) 속성 표를 참조하세요.

sam

AWS SAM CLI가 애플리케이션을 빌드하는 방식입니다. 자세한 내용은 서버리스 애플리케이션 디버깅을 위한 구성 옵션에서 AWS SAM CLI(‘**sam**‘) 속성을 참조하세요.

서버리스 애플리케이션 동기화

이 예에서는 이전 주제(서버리스 애플리케이션 만들기)에서 생성한 서버리스 애플리케이션을 AWS Toolkit for Visual Studio Code를 사용하여 AWS에 동기화하는 방법을 보여 줍니다.

사전 조건

- 전역적으로 고유한 Amazon S3 버킷 이름을 선택했는지 확인합니다.
- 구성한 자격 증명에 Amazon S3, AWS CloudFormation, AWS Lambda 및 Amazon API Gateway 서비스에 대한 적절한 읽기/쓰기 액세스 권한이 포함되어 있는지 확인합니다.
- 배포 유형이 `Image`인 애플리케이션의 경우 배포에 사용할 전역적으로 고유한 Amazon S3 버킷 이름과 Amazon ECR 리포지토리 URI가 모두 있는지 확인합니다.

서버리스 애플리케이션 동기화

1. AWS Explorer 창에서 Lambda 노드의 컨텍스트 메뉴(마우스 오른쪽 버튼 클릭)를 열고 SAM 애플리케이션 동기화를 선택합니다.
2. 이제 배포할 AWS 리전을 선택합니다.
3. 배포에 사용할 `template.yaml` 파일을 선택합니다.
4. 이 배포에서 사용할 수 있는 Amazon S3 버킷의 이름을 입력합니다. 버킷은 배포하려는 리전에 있어야 합니다.

   > **Warning**
   >
   > Amazon S3 버킷 이름은 Amazon S3의 모든 기존 버킷 이름에서 전역적으로 고유해야 합니다. 다음 예에 제공된 이름에 고유 식별자를 추가하거나 다른 이름을 선택해야 합니다.

5. 서버리스 애플리케이션에 패키지 유형이 `Image`인 함수가 포함되어 있는 경우, 이 배포에서 사용할 수 있는 Amazon ECR 리포지토리 이름을 입력합니다. 이 리포지토리는 배포하려는 리전에 있어야 합니다.
6. 배포된 스택의 이름(새 스택 이름 또는 기존 스택 이름)을 입력합니다.
7. 콘솔의 AWS 도구 키트(Toolkit) 탭에서 배포의 성공을 확인합니다.

   오류가 발생하면 메시지가 오른쪽 하단에 팝업됩니다.

   이러한 경우 자세한 내용은 AWS 도구 키트(AWS Toolkit) 탭의 텍스트를 확인하세요. 다음은 오류 세부 정보의 예입니다.

```
Error with child process: Unable to upload artifact HelloWorldFunction referenced
 by CodeUri parameter of HelloWorldFunction resource.
S3 Bucket does not exist. Execute the command to create a new bucket
aws s3 mb s3://pbart-my-sam-app-bucket
An error occurred while deploying a SAM Application. Check the logs for more
 information by running the "View AWS Toolkit Logs" command from the Command
 Palette.
```

이 예에서는 Amazon S3 버킷이 없기 때문에 오류가 발생했습니다.

배포가 완료되면 AWS 탐색기에 해당 애플리케이션이 나열됩니다. 애플리케이션의 일부로 생성된 Lambda 함수를 호출하는 방법을 알아보려면 원격 Lambda 함수 호출 섹션을 참조하세요.

AWS Toolkit 코드 렌즈 활성화

1. 메뉴 모음에서 AWS Cloud9, 기본 설정(Preferences)을 선택합니다.
2. 기본 설정(Preferences) 탭에서 사이드바에 있는 AWS Toolkit을 선택합니다.
3. 코드 렌즈를 활성화하려면 코드 렌즈 활성화(Enable Code Lenses)를 선택합니다.

AWS 클라우드에서 서버리스 애플리케이션 삭제

서버리스 애플리케이션을 삭제하려면 AWS 클라우드에 이전에 배포한 AWS CloudFormation 스택을 삭제해야 합니다. 이 절차는 로컬 호스트에서 애플리케이션 디렉터리를 삭제하지 않습니다.

1. AWS 탐색기를 엽니다.
2. AWS 탐색기 창에서 삭제하려는 배포된 애플리케이션이 포함된 리전을 확장한 다음 AWS CloudFormation을 확장합니다.
3. 삭제할 서버리스 애플리케이션에 해당하는 AWS CloudFormation 스택의 이름에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 엽니다. 그런 다음 Delete CloudFormation Stack(CloudFormation 스택 삭제)을 선택합니다.
4. 선택한 스택을 삭제하겠다는 의사를 확인하고 [삭제(Delete)]를 선택합니다.

스택 삭제가 성공하면 AWS 도구 키트는 해당 스택 이름을 AWS 탐색기의 AWS CloudFormation 목록에서 제거합니다.

서버리스 애플리케이션 디버깅을 위한 구성 옵션

인라인 작업을 사용하면 직접 또는 SAM 템플릿을 사용하여 Lambda 함수를 호출하기 위한 속성을 손쉽게 찾고 정의할 수 있습니다. "lambda"(함수 실행 방법), "sam"(AWS SAM CLI가 애플리케이션을 빌드하는 방법) 및 "aws"(AWS 연결 정보가 제공되는 방법)에 대한 속성을 정의할 수도 있습니다.

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

속성	설명
type	시작 구성을 관리하는 확장을 지정합니다. AWS SAM CLI를 사용하여 로컬로 빌드하고 디버그하도록 항상 aws-sam으로 설정합니다.
name	[시작 구성 디버그(Debug launch configuration)] 목록에 표시할 알아보기 쉬운 이름을 지정합니다.
request	지정된 확장(aws-sam)에 의해 수행될 구성의 유형을 지정합니다 Lambda 함수를 시작하도록 항상 direct-invoke 로 설정합니다.
invokeTarget	리소스를 호출하기 위한 진입점을 지정합니다. Lambda 함수를 직접 호출하려면 다음 invokeTarget 필드의 값을 설정합니다. • target – code(으)로 설정합니다. • lambdaHandler - 호출할 Lambda 함수 핸들러의 이름입니다. • projectRoot - Lambda 핸들러가 포함된 애플리케이션 파일의 경로입니다. SAM 템플릿을 사용하여 Lambda 리소스를 호출하려면 다음 invokeTarget 필드의 값을 설정합니다. • target – template(으)로 설정합니다. • templatePath - SAM 템플릿 파일의 경로입니다.

속성	설명
	• `logicalId` - 호출할 `AWS::Lambda::Function` 또는 `AWS::Serverless::Function` 의 리소스 이름입니다. 리소스 이름은 YAML 형식의 SAM 템플릿에서 확인할 수 있습니다.

Lambda(**`"lambda"`**) 속성

속성	설명
`environmentVariables`	작업 파라미터를 함수에 전달합니다. 예를 들어 Amazon S3 버킷에 기록하는 경우 버킷 이름을 환경 변수로 구성합니다. 작성하는 버킷 이름을 하드코딩하지 마십시오.
`payload`	Lambda 함수에 입력으로 제공할 이벤트 페이로드에 대한 두 가지 옵션을 제공합니다. • `"json"`: 이벤트 페이로드를 정의하는 JSON 형식의 키 값 페어를 정의합니다. • `"path"`: 이벤트 페이로드로 사용되는 파일의 경로입니다.
`memoryMB`	호출된 Lambda 함수의 실행을 위해 제공되는 메모리의 용량(메가바이트)를 지정합니다.
`runtime`	Lambda 함수에서 사용하는 런타임을 지정합니다. 자세한 내용은 AWS Lambda 런타임을 참조하세요.
`timeoutSec`	디버그 세션이 시간 초과될 때까지 허용되는 시간(초)을 설정합니다.

AWS 도구 키트 확장은 AWS SAM CLI를 사용하여 서버리스 애플리케이션을 로컬로 빌드하고 디버그합니다. `launch.json` 파일에 있는 `"sam"` 구성의 속성을 사용하여 AWS SAM CLI 명령의 동작을 구성 할 수 있습니다.

AWS SAM CLI(**"sam"**) 속성

속성	설명	기본값
`buildArguments`	`sam build` 명령이 Lambda 소스 코드를 빌드하는 방법을 구성합니다. 빌드 옵션을 보려면 AWS Serverless Application Model 개발자 가이드에서 sam 빌드를 참조하세요.	비어 있는 문자열
`containerBuild`	AWS Lambda 스타일의 Docker 컨테이너 내부에 함수를 빌드할지 여부를 나타냅니다.	`false`
`dockerNetwork`	Lambda Docker 컨테이너가 연결해야 하는 기존 Docker 네트워크의 이름 또는 ID와 기본 브리지 네트워크를 지정합니다. 지정하지 않으면 Lambda 컨테이너는 기본 브리지 Docker 네트워크에만 연결됩니다.	비어 있는 문자열
`localArguments`	추가 로컬 호출 인수.	비어 있는 문자열
`skipNewImageCheck`	명령이 Lambda 런타임에 대한 최신 Docker 이미지를 가져오는 단계를 건너뛸지 여부를 지정합니다.	`false`
`template`	파라미터를 사용하여 고객 값을 입력함으로써 SAM 템플릿을 사용자 정의합니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 파라미터를 참조하세요.	`"parameters":{}`

AWS 연결(**"aws"**) 속성

속성	설명	기본값
`credentials`	자격 증명 파일에서 특정 프로파일(예:`profile:default`)을 선택하여 AWS 자격 증명을 가져옵니다.	기존 공유 AWS 구성 파일 또는 공유 AWS 자격 증명 파일에서 제공하는 AWS 자격 증명.
`Region`	서비스의 AWS 리전(예: us-east-1)을 설정합니다.	활성 자격 증명 프로파일과 연결된 기본 AWS 리전.

AWS Toolkit를 사용한 AWS Step Functions 작업

AWS Toolkit는 AWS Step Functions를 지원합니다. Step Functions를 사용하면 비즈니스 크리티컬 애플리케이션을 지원하는 AWS Lambda 함수와 기타 AWS 서비스에 대한 워크플로를 정의하는 상태 머신을 생성할 수 있습니다.

AWS 도구 키트를 사용하여 Step Functions로 다음을 수행할 수 있습니다.

- 개별 단계로 구성된 워크플로인 상태 머신 생성 및 게시
- 상태 머신 워크플로를 정의하는 파일 다운로드
- 입력하거나 선택한 입력으로 상태 머신 워크플로 실행

주제

- 필수 조건
- 상태 머신 생성 및 게시
- AWS 도구 키트에서 상태 머신 실행
- 상태 머신 정의 파일 다운로드 및 워크플로 시각화

필수 조건

Step Functions는 코드를 실행하고 AWS 리소스에 액세스할 수 있습니다(예: Lambda 함수 호출). 보안 유지를 위해 사용자는 IAM 역할을 사용하여 이러한 리소스에 대한 Step Functions 액세스 권한을 부여해야 합니다.

AWS Toolkit를 사용하면 상태 시스템을 생성하는 AWS 리전에 유효한 자동으로 생성된 IAM 역할을 사용할 수 있습니다. 상태 머신에 대한 고유한 IAM 역할을 생성하려면 AWS Step Functions 개발자 가이드의 AWS Step Functions의 IAM과 작용 원리를 참조하세요.

상태 머신 생성 및 게시

AWS 도구 키트로 상태 머신을 생성할 때 비즈니스 사례에 대한 워크플로를 정의하는 스타터 템플릿을 선택합니다. 그런 다음 특정 요구 사항에 맞게 해당 템플릿을 편집하거나 바꿀 수 있습니다. 구조를 나타내는 파일에서 상태 머신을 정의하는 방법에 대한 자세한 내용은 AWS Step Functions 개발자 가이드의 Amazon States Language를 참조하세요.

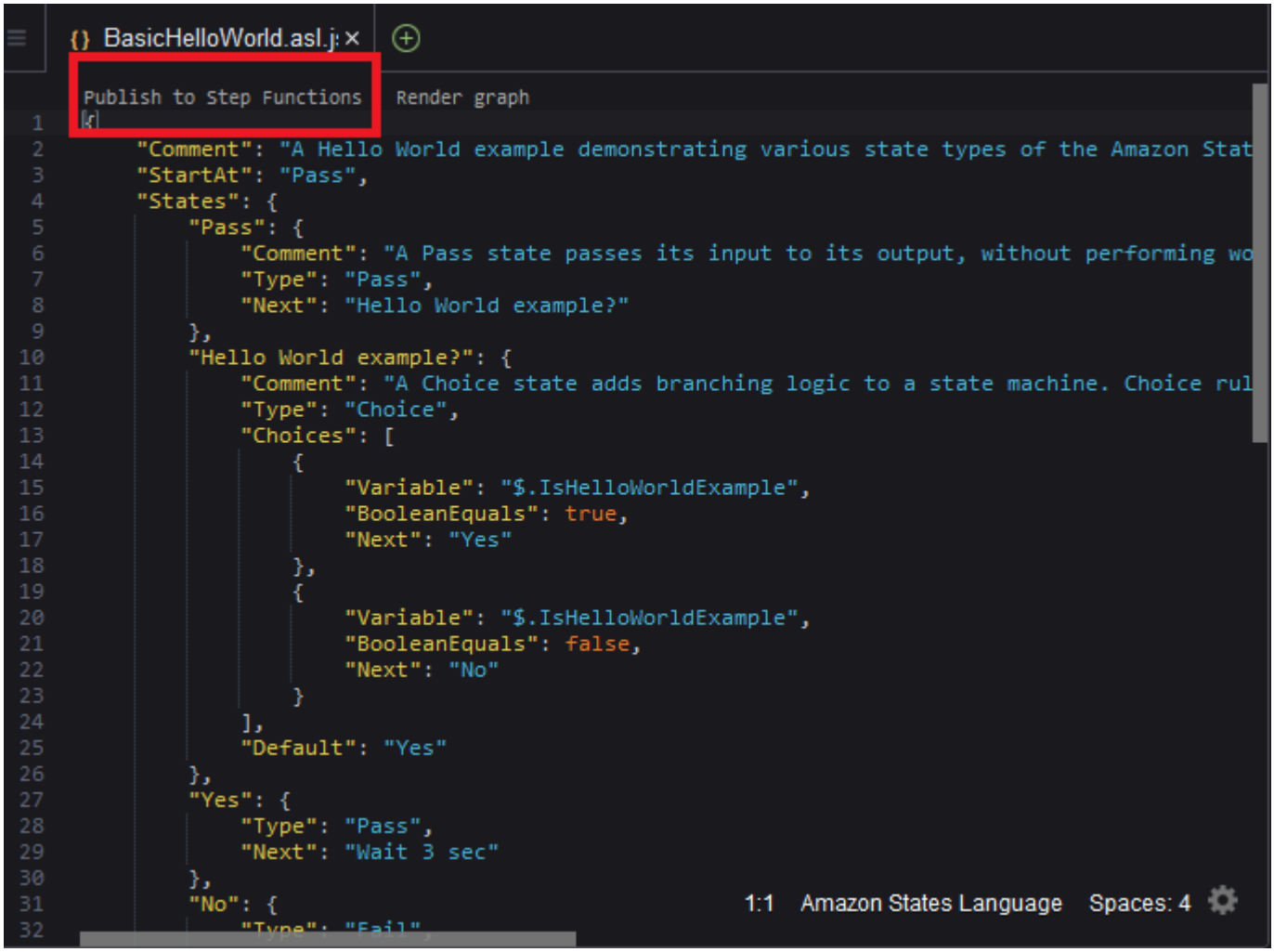
1. AWS Explorer 창에서 Step Functions에 대한 컨텍스트 메뉴(마우스 오른쪽 버튼 클릭)를 연 다음 Step Function 상태 머신 새로 생성을 선택합니다.
2. 명령 패널에서 상태 머신의 워크플로에 대한 스타터 템플릿을 선택합니다.
3. 그런 다음 상태 머신을 정의하는 ASL(Amazon States Language) 파일의 형식을 선택합니다.

   편집기가 열리고 상태 머신의 워크플로를 정의하는 ASL 파일이 표시됩니다.

   > **Note**
   >
   > ASL 파일을 편집하여 워크플로를 사용자 지정하는 방법에 대한 자세한 내용은 상태 머신 구조를 참조하세요.

4. ASL 파일에서 Step Functions에 게시(Publish to Step Functions)를 선택하여 AWS 클라우드에 상태 머신을 추가합니다.

   > **Note**
   >
   > ASL 파일에서 그래프 렌더링(Render graph)을 선택하여 상태 머신의 워크플로를 시각적으로 표시할 수도 있습니다.



5. 명령 패널에서 Step Function을 호스팅할 AWS 리전을 선택합니다.
6. 다음으로, 새 Step Function을 생성하거나 기존 Step Function을 업데이트하도록 선택할 수 있습니다.

   Quick Create

   이 옵션을 사용하면 step-functions/latest/dg/concepts-standard-vs-express.html로 ASL 파일에서 새 Step Function을 생성할 수 있습니다. 다음을 지정하라는 메시지가 나타납니다.

   - Step Function이 코드를 실행하고 AWS 리소스에 액세스할 수 있도록 하는 IAM 역할. (상태 머신을 생성하는 AWS 리전에 유효한 자동 생성 IAM 역할을 선택할 수 있습니다.)
   - 새 함수의 이름.

상태 머신이 성공적으로 생성되었는지 확인하고 AWS 도구 키트 출력 탭에서 해당 ARN을 확인할 수 있습니다.

Quick Update

상태 머신이 이미 AWS 리전에 있는 경우 현재 ASL 파일로 업데이트할 상태 머신을 선택할 수 있습니다.

상태 시스템이 성공적으로 업데이트되었는지 확인하고 AWS Toolkit 출력 탭에서 해당 ARN을 확인할 수 있습니다.

상태 머신을 생성하면 AWS Explorer 창의 Step Functions 아래에 해당 상태 머신이 나타납니다. 즉시 나타나지 않으면 도구 키트(Toolkit) 메뉴, 탐색기 새로 고침(Refresh Explorer)을 선택합니다.

AWS 도구 키트에서 상태 머신 실행

AWS 도구 키트를 사용하여 원격 상태 머신을 실행할 수 있습니다. 실행 중인 상태 시스템은 JSON 텍스트를 입력으로 수신하고 해당 입력을 워크플로의 첫 번째 상태에 전달합니다. 개별 상태는 JSON을 입력으로 수신하고 일반적으로 다음 상태에서 JSON을 출력으로 넘겨줍니다. 자세한 내용은 Step Functions 입/출력 처리를 참조하세요.

1. AWS Explorer 창에서 Step Functions를 선택합니다. 그런 다음 특정 상태 머신에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 실행 시작(Start Execution)을 선택합니다.
2. 실행 시작(Start Execution) 창에서 아래 필드에 직접 텍스트를 입력하거나 로컬 디바이스에서 파일을 업로드하여 상태 머신의 워크플로에 대한 JSON 형식 입력을 추가합니다.
3. 실행(Execute)을 선택합니다.

   AWS 도구 키트 출력 탭에 워크플로가 시작되었다는 확인 메시지와 프로세스 ID의 ARN이 표시됩니다. 해당 프로세스 ID를 사용하여 워크플로가 성공적으로 실행되었는지 여부를 AWS Step Functions 콘솔에서 확인할 수 있습니다. 워크플로가 시작되고 종료된 시간에 대한 타임스탬프도 볼 수 있습니다.

상태 머신 정의 파일 다운로드 및 워크플로 시각화

상태 머신을 다운로드한다는 것은 해당 상태 머신의 구조를 나타내는 JSON 텍스트가 포함된 파일을 다운로드한다는 의미입니다. 그런 다음 이 파일을 편집하여 새 상태 머신을 생성하거나 기존 상태 머신

을 업데이트할 수 있습니다. 자세한 내용은 AWS Step Functions Developer Guide의 Amazon States Language를 참조하세요.

1. AWS Explorer 창에서 Step Functions를 선택합니다. 그런 다음 특정 상태 시스템에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 정의 다운로드(Download Definition)를 선택합니다.

   > **Note**
   >
   > 컨텍스트 메뉴는 이름 복사(Copy Name) 및 ARN 복사(Copy ARN) 옵션도 제공합니다.

2. 저장(Save) 대화 상자에서 다운로드한 상태 머신 파일을 저장할 환경의 폴더를 선택한 다음 저장(Save)을 선택합니다.

   상태 머신의 워크플로를 정의하는 JSON 형식의 파일이 편집기에 표시됩니다.

3. 워크플로의 시각적 표현을 표시하려면 그래프 렌더링(Render graph)을 선택합니다.

   상태 머신 워크플로의 상태 시퀀스를 보여주는 순서도가 창에 표시됩니다.

Systems Manager 자동화 문서로 작업

AWS Systems Manager를 사용하면 AWS의 인프라에 대한 가시성과 제어를 확보할 수 있습니다. Systems Manager는 통합된 사용자 인터페이스를 제공하므로 여러 AWS 서비스의 운영 데이터를 보고 AWS 리소스 전체에서 운영 작업을 자동화할 수 있습니다.

Systems Manager 문서는 Systems Manager가 관리형 인스턴스에서 실행하는 작업을 정의합니다. 자동화 문서는 일반적인 유지 관리 및 배포 작업을 수행하는 데 사용하는 Systems Manager 문서 유형입니다. Amazon Machine Image(AMI) 생성이나 업데이트도 여기에 포함됩니다. 이 주제에서는 AWS 도구 키트를 사용하여 자동화 문서를 생성, 편집, 게시 및 삭제하는 방법을 간략하게 설명합니다.

주제

- 가정 및 사전 조건
- Systems Manager Automation 문서에 대한 IAM 권한
- 새 Systems Manager 자동화 문서 생성
- Systems Manager 자동화 문서 게시
- 기존 Systems Manager 자동화 문서 편집
- 버전 작업

- Systems Manager 자동화 문서 삭제
- Systems Manager 자동화 문서 실행
- AWS 도구 키트에서 Systems Manager 자동화 문서 문제 해결

가정 및 사전 조건

시작하기 전에 다음 조건을 충족하는지 확인하세요.

- Systems Manager에 대해 잘 알고 있습니다. 자세한 정보는 AWS Systems Manager 사용 설명서를 참조하세요.
- Systems Manager 자동화 사용 사례에 대해 잘 알고 있습니다. 자세한 내용은 AWS Systems Manager 사용 설명서의 AWS Systems Manager Automation을 참조하세요.

Systems Manager Automation 문서에 대한 IAM 권한

Systems Manager 자동화 문서를 생성, 편집, 게시 및 삭제하려면, 필요한 AWS Identity and Access Management(IAM) 권한이 포함된 보안 인증 정보 프로파일이 있어야 합니다. 다음 정책 문서는 보안 주체 정책에서 사용할 수 있는 필요한 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 생성을 참조하세요.

새 Systems Manager 자동화 문서 생성

AWS 도구 키트를 사용하여 JSON 또는 YAML 형식의 자동화 문서를 생성할 수 있습니다. 자동화 문서를 생성하면 제목 없는 파일로 표시됩니다. 파일 이름을 지정하고 저장할 수 있습니다. 그러나 파일은 게시되기 전에는 AWS에 업로드되지 않습니다.

자동화 문서를 새로 생성하려면

1. 왼쪽 탐색 창에서 검색 아이콘을 선택하거나 Ctrl+P를 눌러 검색 창을 엽니다.
2. 검색 창에서 "systems manager"라는 용어를 입력하기 시작하고 AWS: Create a new Systems Manager Document Locally(AWS: 로컬로 새 Systems Manager 문서 생성) 명령이 표시되면 해당 명령을 선택합니다.
3. “Hello World” 예제의 스타터 템플릿 중 하나를 선택합니다.
4. 문서 형식으로 JSON 또는 YAML을 선택합니다.

   편집기에 새 자동화 문서가 표시됩니다.

> **Note**
>
> 로컬 자동화 문서를 처음 생성하면 AWS에 자동으로 표시되지 않습니다. 자동화 문서를 실행하려면 먼저 AWS에 게시해야 합니다.

Systems Manager 자동화 문서 게시

AWS 도구 키트에서 자동화 문서를 생성하거나 편집한 후 AWS에 게시할 수 있습니다.

자동화 문서를 게시하려면

1. 기존 Systems Manager 자동화 문서 편집에 약술된 절차를 사용하여 게시하려는 자동화 문서를 엽니다.
2. 왼쪽 탐색 창에서 검색 아이콘을 선택하거나 Ctrl+P를 눌러 검색(Search) 창을 엽니다.
3. 검색 창에서 "systems manager"라는 용어를 입력하기 시작하고 AWS: Publish a new Systems Manager Document(AWS: 새 Systems Manager 문서 게시) 명령이 표시되면 해당 명령을 선택합니다.

4. 1/3단계(Step 1 of 3)에서 문서를 게시할 AWS 리전을 선택합니다.
5. 2/3단계(Step 2 of 3)에서 빠른 생성(Quick Create)을 선택하여 자동화 문서를 생성합니다. 또는 Quick Update(빠른 업데이트)를 선택하여 해당 리전의 기존 자동화 문서를 업데이트합니다.

> **Note**
>
> 소유한 자동화 문서만 업데이트할 수 있습니다. 빠른 업데이트(Quick Update)를 선택하고 해당 리전에 문서를 소유하고 있지 않은 경우 문서를 업데이트하기 전에 게시하라는 메시지가 나타납니다.

6. 3/3단계(Step 3 of 3)에서 이전 단계에서 선택한 항목에 따라 새 자동화 문서의 이름을 입력하거나 업데이트할 기존 문서를 선택합니다.

> **Note**
>
> AWS의 기존 자동화 문서에 대한 업데이트를 게시하면 새 버전이 문서에 추가됩니다. 문서에 여러 버전이 있는 경우 기본 버전을 설정할 수 있습니다.

기존 Systems Manager 자동화 문서 편집

AWS 탐색기를 사용하여 기존 Systems Manager 자동화 문서를 찾습니다. 기존 문서를 열면 AWS Cloud9 편집기에서 제목 없는 파일로 나타납니다. 다운로드하는 자동화 문서에는 세 가지 유형이 있습니다.

- Amazon 소유(Owned by Amazon): 런타임에 파라미터를 지정하여 사용할 수 있는 미리 구성된 SSM 문서입니다.
- 내 소유(Owned by me): 내가 생성하고 AWS에 게시한 문서입니다.
- 나와 공유(Shared with me): 내 AWS 계정 ID를 기반으로 소유자가 나와 공유한 문서입니다.

AWS에서 업데이트할 수 있는 문서 유형은 내 소유(Owned by me) 문서뿐입니다. Amazon에서 공유하거나 소유한 자동화 문서를 다운로드하고 AWS Cloud9에서 편집할 수도 있습니다. 그러나 AWS에 게시할 때는 새 문서를 생성하거나 소유하고 있는 기존 문서를 업데이트해야 합니다. 다른 소유자가 있거나 Amazon에서 소유한 문서의 새 버전을 생성할 수 없습니다.

자세한 내용은 AWS Systems Manager 사용 설명서의 AWS Systems Manager 문서를 참조하세요.

1. AWS Explorer의 Systems Manager에서 다운로드하려는 SSM 문서의 범주(Amazon 소유(Owned by Amazon), 내 소유(Owned by me) 또는 나와 공유(Shared with me))를 선택합니다.
2. 특정 문서에 대해 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 YAML로 다운로드(Download as YAML) 또는 JSON으로 다운로드(Download as JSON)를 선택합니다.

   형식이 지정된 SSM 문서가 새 편집기 탭에 표시됩니다.

편집을 마친 후 AWS: Publish a new Systems Manager Document(AWS: 새 Systems Manager 문서 게시) 명령을 사용하여 AWS 클라우드에서 새 문서를 생성하거나 소유하고 있는 기존 문서를 업데이트할 수 있습니다.

버전 작업

Systems Manager 자동화 문서는 변경 관리를 위해 버전을 사용합니다. AWS 도구 키트를 사용하면 문서를 실행할 때 사용되는 버전인 문서의 기본 버전을 설정할 수 있습니다.

기본 버전을 설정하려면

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

  > **Note**
  >
  > 선택한 문서에 버전이 하나만 있는 경우 기본값을 변경할 수 없습니다.

Systems Manager 자동화 문서 삭제

AWS 도구 키트에서 소유한 자동화 문서를 삭제할 수 있습니다. Automation 문서를 삭제하면 문서와 문서의 모든 버전이 삭제됩니다.

> **Important**
>
> - 삭제는 실행 취소할 수 없습니다.
> - 이미 시작된 자동화 문서를 삭제해도 실행 시 생성되거나 수정된 AWS 리소스는 삭제되지 않습니다.
> - 문서를 소유하는 경우에만 삭제가 허용됩니다.

자동화 문서를 삭제하려면

1. AWS 탐색기 창의 Systems Manager(시스템 관리자)에서 Owned by Me(내 소유)를 확장하여 문서를 나열합니다.
2. 삭제할 문서의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 문서 삭제(Delete document)를 선택합니다.
3. 나타나는 경고 대화 상자에서 삭제(Delete)를 선택하여 확인합니다.

Systems Manager 자동화 문서 실행

자동화 문서가 AWS에 게시된 후 이를 실행하여 AWS 계정에서 사용자를 대신하여 태스크를 수행할 수 있습니다. Automation 문서를 실행하려면 AWS Management Console, Systems Manager API, AWS CLI 또는 AWS Tools for PowerShell을 사용합니다. 자동화 문서를 실행하는 방법에 대한 지침은 AWS Systems Manager 사용 설명서의 단순한 자동화 실행을 참조하세요.

또는 AWS SDK 중 하나를 Systems Manager API와 함께 사용하여 Automation 문서를 실행하려는 경우 AWS SDK 참조 자료를 참조하세요.

> **Important**
>
> 자동화 문서를 실행하면 AWS에서 새 리소스를 생성할 수 있으며 청구 비용이 발생할 수 있습니다. 자동화 문서를 실행하기 전에 계정에서 생성할 자동화 문서를 이해하는 것이 좋습니다.

AWS 도구 키트에서 Systems Manager 자동화 문서 문제 해결

자동화 문서를 AWS 도구 키트에 저장했지만 AWS Management Console에 표시되지 않음

AWS 도구 키트에 자동화 문서를 저장해도 자동화 문서가 AWS에 게시되지 않습니다. Automation 문서를 게시하는 방법에 대한 자세한 내용은 Systems Manager 자동화 문서 게시 섹션을 참조하세요.

권한 오류로 인해 자동화 문서를 게시하지 못함

AWS 자격 증명 프로파일에 Automation 문서를 게시하는 데 필요한 권한이 있는지 확인하세요. 권한 정책 예제는 Systems Manager Automation 문서에 대한 IAM 권한 단원을 참조하십시오.

자동화 문서를 AWS에 게시했지만 AWS 탐색기 창에 해당 문서가 표시되지 않음

AWS 탐색기 창에서 탐색 중인 동일한 AWS 리전에 문서를 게시했는지 확인하세요.

자동화 문서를 삭제했지만 생성된 리소스에 대한 요금이 계속 청구됨

자동화 문서를 삭제해도 생성되거나 수정된 리소스는 삭제되지 않습니다. AWS Billing Management Console(결제 관리 콘솔)에서 생성한 AWS 리소스를 식별하고 요금을 살펴보고 삭제할 리소스를 선택할 수 있습니다.

AWS Cloud9 IDE에서 Amazon ECR 작업

Amazon Elastic Container Registry(Amazon ECR)는 안전하고 확장 가능한 AWS 관리형 컨테이너 레지스트리 서비스입니다. 여러 Amazon ECR 서비스 기능을 AWS Toolkit Explorer에서 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 CLI와 다른 플랫폼을 설치하여 AWS Cloud9 콘솔을 통해 모든 Amazon ECR 기능에 액세스할 수 있습니다.

Amazon ECR에 대한 자세한 내용은 Amazon Elastic Container Registry 사용 설명서의 Amazon ECR이란 무엇입니까?를 참조하세요.

필수 조건

다음은 AWS Cloud9 Amazon EC2 환경용 AWS Cloud9 IDE에 사전 설치되어 있습니다. 이는 AWS Cloud9 IDE에서 Amazon ECR 서비스에 액세스하는 데 필요합니다.

IAM 자격 증명

AWS 콘솔에서 생성하여 인증에 사용한 IAM 역할입니다. IAM에 대한 자세한 내용은 AWS Identity and Access Management 사용 설명서를 참조하십시오.

도커 구성

Docker는 AWS Cloud9 Amazon EC2 환경용 AWS Cloud9 IDE에 사전 설치되어 있습니다. Docker에 대한 자세한 내용은 Docker Engine 설치를 참조하세요.

AWS CLI 버전 2 구성

AWS CLI 버전 2는 AWS Cloud9 Amazon EC2 환경용 AWS Cloud9 IDE에 사전 설치되어 있습니다. AWS CLI 버전 2에 대한 자세한 내용은 AWS CLI 버전 2 설치, 업데이트 및 제거를 참조하세요.

주제

- 에서 Amazon Elastic 컨테이너 레지스트리 서비스와 함께 작업하기 AWS Cloud9

에서 Amazon Elastic 컨테이너 레지스트리 서비스와 함께 작업하기 AWS Cloud9

IDE의 탐색기에서 AWS Amazon Elastic Container 레지스트리 (Amazon ECR) 서비스에 직접 액세스할 수 있습니다. AWS Cloud9 Amazon ECR을 사용하여 Amazon ECR 리포지토리에 프로그램 이미지를 푸시할 수 있습니다. 시작하려면 다음 단계를 따릅니다.

1. 이미지를 빌드하는 데 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.

Sections

- 필수 조건
- 1. Dockerfile 생성
- 2. Dockerfile에서 이미지 빌드
- 3. 새 리포지토리 생성
- 4. 이미지 푸시, 가져오기, 삭제

필수 조건

AWS 툴킷의 Amazon ECR 기능을 사용하려면 먼저 이러한 사전 요구 사항을 충족해야 합니다. AWS Cloud9이러한 사전 요구 사항은 Amazon EC2용 AWS Cloud9 IDE 환경에 사전 설치되어 있으며 AWS Cloud9 Amazon ECR에 액세스하는 데 필요합니다.

1. Dockerfile 생성

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

Dockerfile 생성

1. Dockerfile을 저장할 디렉터리로 이동하려면 AWS Cloud9 IDE의 왼쪽 탐색 표시줄에서 Toggle Tree(트리 전환) 옵션을 선택합니다.
2. Dockerfile이라는 새 파일을 생성합니다.

> **Note**
>
> AWS Cloud9 IDE에서 파일 유형이나 파일 확장자를 선택하라는 메시지가 표시될 수 있습니다. 이런 경우에는 일반 텍스트를 선택하십시오. AWS Cloud9 IDE의 확장자는 “도커파일”입니다. 그러나 사용하지 않는 것이 좋습니다. 이 확장자가 특정 버전의 Docker 또는 기타 관련 애플리케이션과 충돌을 일으킬 수 있기 때문입니다.

IDE를 사용하여 도커파일 편집하기 AWS Cloud9

Dockerfile에 파일 확장자가 있는 경우 파일에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 해당 파일 확장자를 제거합니다. 확장자가 있는 Dockerfile은 특정 버전의 Docker 또는 기타 관련 애플리케이션과 충돌을 일으킬 수 있기 때문입니다.

Dockerfile에서 파일 확장자를 제거한 후 다음을 수행합니다.

1. IDE에서 직접 빈 도커파일을 엽니다. AWS Cloud9
2. 다음 예제의 내용을 Dockerfile에 복사합니다.

   Example Dockerfile 이미지 템플릿

```
FROM ubuntu:22.04

# 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 22.04 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.

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 Dockerfile과 동일한 디렉터리에서 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

출력값은 다음과 같습니다.

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

4. Ubuntu 22.04를 기반으로 새로 빌드된 이미지를 실행하려면 echo 명령을 사용합니다.

> **Note**
>
> 이 단계는 이미지를 생성하거나 푸시하는 데 필요하지 않습니다. 그러나 프로그램 이미지가 실행되면 어떻게 작동하는지 확인할 수 있습니다.

```
FROM ubuntu:22.04
CMD ["echo", "Hello from Docker in Cloud9"]
```

그런 다음 Dockerfile을 실행하고 빌드합니다. Dockerfile과 동일한 디렉터리에서 이 명령을 실행해야 합니다.

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

Example

출력값은 다음과 같습니다.

```
Hello from Docker in Cloud9
```

Docker run 명령에 대한 자세한 내용은 Docker 웹 사이트에서 Docker run reference(Docker 실행 참조)를 참조하세요.

3. 새 리포지토리 생성

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

새 Amazon ECT 리포지토리 생성

1. AWS Cloud9 IDE 내비게이션 바에서 툴킷 아이콘을 선택합니다.AWS
2. AWS Explorer 메뉴를 확장합니다.
3. AWS 리전 해당 버전과 연결된 기본값을 찾으십시오. AWS 계정그런 다음 선택하면 AWS Cloud9 IDE를 통해 제공되는 서비스 목록이 표시됩니다.
4. ECR 옵션의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Create new repository(새 리포지토리 생성) 프로세스를 시작합니다. 그런 다음 Create Repository(리포지토리 생성)를 선택합니다.
5. 프롬프트의 메시지를 따라 프로세스를 완료합니다.
6. 프로세스가 완료되면 AWS Explorer 메뉴의 ECR 섹션에서 새 리포지토리에 액세스할 수 있습니다.

4. 이미지 푸시, 가져오기, 삭제

Dockerfile에서 이미지를 빌드하고 리포지토리를 생성한 후에는 Amazon ECR 리포지토리로 이미지를 푸시할 수 있습니다. 또한 AWS 탐색기와 Docker 및 AWS CLI를 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. IDE 내에서 터미널을 엽니다. AWS Cloud9
2. get-login-password방법을 사용하여 사설 ECR 레지스트리에 인증하고 지역 및 AWS 계정 ID를 입력합니다.

```
aws ecr get-login-password \
```

```
    --region <region> \
| docker login \
    --username AWS \
    --password-stdin <aws_account_id>.dkr.ecr.<region>.amazonaws.com
```

> **Important**
>
> 앞의 명령에서 **region** 및 **AWS_account_id**를 AWS 계정과 관련된 정보로 바꿉니다. 유효한 **region** 값은 us-east-1입니다.

이미지 태그 지정 및 리포지토리에 푸시

의 인스턴스로 Docker를 인증한 후 이미지를 AWS저장소로 푸시합니다.

1. docker images 명령을 사용하여 로컬에 저장한 이미지를 보고 태그를 지정할 이미지를 식별합니다.

   ```
   docker images
   ```

   Example

   출력값은 다음과 같습니다.

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

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.

> **Important**
>
> 로컬 리포지토리의 이름이 AWS Amazon EC2 리포지토리의 이름과 동일한지 확인하십시오. 이 예에서는 두 리포지토리의 이름이 모두 `hello-world`여야 합니다. Docker를 사용하여 이미지를 푸시하는 방법에 대한 자세한 내용은 도커 이미지 푸시를 참조하세요.

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

Example

출력값은 다음과 같습니다.

```
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 성공적으로 업로드한 후 탐색기 탭에서 Refresh Explorer (탐색기 새로 고침) 를 선택하여 AWS 툴킷을 새로 고칩니다. AWS 그러면 IDE의 AWS 탐색기 메뉴에서 해당 이미지를 볼 수 AWS Cloud9 있습니다.

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

출력값은 다음과 같습니다.

```
azonaws.com/hello-world:latest
latest: Pulling from hello-world
Digest: sha256:e02c521fd65eae4ef1acb746883df48de85d55fc85a4172a09a124b11b339f5e
Status: Image is up to date for 922327013870.dkr.ecr.us-west-2.amazonaws.com/hello-
world.latest
```

Amazon ECR 리포지토리에서 이미지 삭제

AWS Cloud9 IDE에서 이미지를 삭제하는 방법에는 두 가지가 있습니다. 첫 번째 방법은 AWS 탐색기를 사용하는 것입니다.

1. AWS 탐색기에서 ECR 메뉴를 확장합니다.
2. 이미지를 삭제할 리포지토리를 확장합니다.
3. 삭제할 이미지와 연결된 이미지 태그의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 엽니다.
4. 해당 태그와 연결된 저장된 이미지를 모두 삭제하려면 Delete Tag...(태그 삭제)를 선택합니다.

AWS CLI를 사용하여 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

출력값은 다음과 같습니다.

```
{
    "failures": [],
    "imageIds": [
        {
            "imageTag": "latest",
```

```
            "imageDigest":
 "sha256:215d7e4121b30157d8839e81c4e0912606fca105775bb0636b95aed25f52c89b"
        }
    ]
}
```

Amazon ECR 인스턴스에서 리포지토리 삭제

AWS Cloud9 IDE에서 리포지토리를 삭제하는 방법은 두 가지가 있습니다. 첫 번째 방법은 AWS 탐색기를 사용하는 것입니다.

1. AWS 탐색기에서 ECR 메뉴를 확장합니다.
2. 삭제할 리포지토리에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 엽니다.
3. Delete Repository...(리포지토리 삭제...)를 선택합니다.

CLI에서 Amazon ECR 리포지토리 삭제 AWS

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

> **Note**
>
> 일반적으로 리포지토리에 포함된 이미지를 먼저 삭제하지 않으면 리포지토리를 삭제할 수 없습니다. 하지만 --force 플래그를 추가하면 리포지토리와 모든 해당 이미지를 한 번에 삭제할 수 있습니다.

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

Example

출력값은 다음과 같습니다.

```
--repository-name hello-world --force
```

```
{
    "repository": {
        "repositoryUri": "922327013870.dkr.ecr.us-west-2.amazonaws.com/hello-
world",
        "registryId": "922327013870",
        "imageTagMutability": "MUTABLE",
        "repositoryArn": "arn:aws:ecr:us-west-2:922327013870:repository/hello-
world",
        "repositoryName": "hello-world",
        "createdAt": 1664469874.0
    }
}
```

AWS Cloud9 IDE에서의 AWS IoT 작업

AWS Cloud9 IDE에서 AWS IoT를 사용하면, AWS Cloud9에서의 작업 흐름 방해를 최소화하면서 AWS IoT 서비스와 상호 작용할 수 있습니다. 이 설명서에서는 AWS Cloud9 IDE에서 제공하는 AWS IoT 서비스 기능을 사용하는 방법을 설명합니다. 자세한 정보는 AWS IoT 개발자 설명서.의 AWS IoT란 무엇입니까?를 참조하세요.

AWS IoT 사전 조건

AWS Cloud9 IDE에서 AWS IoT를 사용하려면 AWS 계정 및 AWS Cloud9 설정이 모든 요구 사항을 충족하는지 확인해야 합니다. AWS IoT 서비스에 관한 AWS 계정 요구 사항과 AWS 사용자 권한에 대한 자세한 내용은 AWS IoT 개발자 설명서의 AWS IoT Core 시작하기를 참조하세요.

AWS IoT 사물

AWS IoT는 장치를 AWS 서비스 및 AWS 리소스에 연결합니다. 사물이라는 객체를 사용하여 장치를 AWS IoT에 연결할 수 있습니다. 사물이란 특정 디바이스 또는 논리적 엔터티의 표현입니다. 사물은 물리적 디바이스 또는 센서일 수 있습니다(예: 전구 또는 벽면 스위치). AWS IoT 사물에 대한 자세한 내용은 AWS IoT 개발자 설명서의 AWS IoT로 장치 관리를 참조하세요.

AWS IoT 사물 관리

AWS Cloud9 IDE는 사물을 효율적으로 관리할 수 있는 몇 가지 기능을 제공합니다. AWS IoT 사물을 관리하려면 다음 단계를 따르세요.

- Create a thing

- Attach a certificate to a thing
- Detach a certificate from a thing
- Delete a thing

사물을 생성하려면

1. AWS 탐색기에서 IoT 서비스 섹션을 확장합니다.
2. thing(사물)의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Create Thing(사물 생성)을 선택합니다.
3. Thing Name(사물 이름) 필드에 사물의 이름을 입력하고 지시에 따릅니다.
4. 이 단계가 완료되면 사물 아이콘과 사용자가 지정한 이름이 Thing(사물) 섹션에 표시됩니다.

사물에 인증서를 첨부하려면

1. AWS 탐색기에서 IoT 서비스 섹션을 확장합니다.
2. Things(사물) 하위 섹션에서 인증서를 첨부할 사물을 찾습니다.
3. 사물에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 컨텍스트 메뉴에서 Attach Certificate(인증서 첨부)를 선택하여 인증서 목록이 포함된 입력 선택기를 엽니다.
4. 목록에서 사물에 첨부할 인증서에 해당하는 certificate ID(인증서 ID)를 선택합니다.
5. 이 단계가 완료되면 AWS 탐색기에서 인증서를 (사용자가 사물에 첨부한 항목으로) 액세스할 수 있습니다.

사물에서 인증서를 분리하려면

1. AWS 탐색기에서 IoT 서비스 섹션을 확장합니다.
2. Things(사물) 하위 섹션에서 인증서를 분리할 사물을 찾습니다.
3. thing(사물)의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Attach Certificate(인증서 첨부)를 선택합니다.
4. 이 단계가 완료되면 분리된 인증서는 더 이상 AWS 탐색기의 사물 항목에 표시되지 않습니다. 하지만 Certificates(인증서) 하위 섹션에서는 여전히 액세스할 수 있습니다.

사물을 삭제하려면

1. AWS 탐색기에서 IoT 서비스 섹션을 확장합니다.
2. Things(사물) 하위 섹션에서 삭제할 사물을 찾습니다.
3. thing(사물)의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Delete Thing(사물 삭제)을 선택합니다.
4. 이 단계가 완료되면 삭제된 사물은 Things(사물) 하위 섹션에서 더 이상 사용할 수 없습니다.

> **Note**
>
> 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 자격 증명과 연결하는 디지털 인증서입니다. AWS IoT 인증에 대한 자세한 내용은 AWS IoT 개발자 설명서의 인증(IoT)를 참조하세요.

인증서 관리

AWS 툴킷은 AWS 탐색기에서 바로 AWS IoT 인증서를 관리하는 다양한 방법을 제공합니다. 이러한 내용은 다음 단계에 나와 있습니다.

- Create a certificate
- Change a certificate status
- Attach a policy to a certificate
- Delete a certificate

AWS IoT 인증서를 생성하려면

X.509 인증서는 AWS IoT 인스턴스에 연결하는 데 사용합니다.

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

> **Note**
>
> - 기본 파일 이름에는 인증서 ID가 접두사로 포함됩니다.
> - X.509 인증서만 AWS IoT 서비스를 통해 AWS 계정와 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 Toolkit은 AWS 계정에서 인증서를 삭제합니다.

인증서 상태를 수정하려면

개별 인증서의 상태는 AWS 탐색기의 인증서 ID 옆에 표시되며 active(활성), inactive(비활성) 또는 revoked(해지됨)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(해지됨) 상태인 인증서는 활성화할 수 없습니다.

1. AWS 탐색기에서 IoT 서비스 섹션을 확장합니다.
2. Certificates(인증서) 하위 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 상태를 activate(활성)으로 변경합니다.
- 인증서의 상태가 inactive(비활성)인 경우 activate(활성)를 선택하여 상태를 activate(활성)로 변경합니다.

- 인증서의 상태가 active(활성) 또는 inactive(비활성)인 경우 revoke(해지)를 선택하여 상태를 revoked(취소됨)로 변경합니다.

> **Note**
>
> 이러한 상태 변경 작업은 Things(사물) 하위 섹션에 표시되는 사물에 첨부된 인증서를 선택하면 수행할 수 있습니다.

인증서에 IoT 정책을 연결하려면

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

인증서에서 IoT 정책을 분리하려면

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

인증서를 삭제하려면

1. AWS 탐색기에서 IoT 서비스 헤딩을 확장합니다.
2. Certificates(인증서) 하위 섹션에서 삭제할 인증서를 찾습니다.

3. 인증서의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 컨텍스트 메뉴에서 Delete Certificate(인증서 삭제)를 선택합니다.

> **Note**
>
> 사물에 첨부되거나 활성 상태인 인증서는 삭제할 수 없습니다. 정책이 연결된 인증서는 삭제할 수 있습니다.

AWS IoT 정책

AWS IoT Core 정책은 JSON 문서를 통해 정의됩니다. 각 정책은 하나 이상의 정책 문을 포함해야 합니다. 정책은 AWS IoT, AWS, 및 장치가 서로 상호 작용하는 방법을 정의합니다. 정책 문서를 만드는 방법에 대한 자세한 내용은 AWS IoT 개발자 안내서의 IoT 정책을 참조하세요.

> **Note**
>
> 이름이 지정된 정책은 롤백할 수 있도록 버전이 지정됩니다. AWS 탐색기에서 IoT 정책은 AWS IoT 서비스의 Policies(정책) 하위 섹션에 나열됩니다. 정책을 확장하면 정책 버전을 볼 수 있습니다. 기본 버전은 별표(*)로 표시됩니다.

정책 관리

AWS Cloud9 IDE는 AWS IoT 서비스 정책을 관리하는 다양한 방법을 제공합니다. 다음은 VS Code의 AWS 탐색기에서 바로 정책을 관리하거나 수정하는 방법입니다.

- Create a policy
- Upload a new policy version
- Edit a policy version
- Change the policy version defualt
- Change the policy version defualt

AWS IoT 정책을 생성하려면

> **Note**
>
> AWS 탐색기에서 새 정책을 생성할 수 있습니다. 하지만 정책을 정의하는 JSON 문서가 파일 시스템에 있어야 합니다.

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

새 AWS IoT 정책 버전을 업로드하려면

JSON 문서를 정책에 업로드하면 새 버전의 정책을 만들 수 있습니다.

> **Note**
>
> AWS 탐색기를 사용하여 새 버전을 만들려면 파일 시스템에 새 JSON 문서가 있어야 합니다.

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

   새 버전은 AWS 탐색기의 정책에서 액세스할 수 있습니다.

AWS IoT 정책 버전을 편집하려면

AWS Cloud9을 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 탐색기에서 IoT 서비스 섹션을 확장합니다.
2. Policies(정책) 하위 섹션을 펼치고 업데이트할 정책을 찾습니다.
3. Policy Name(정책 이름)을 열려면 Document(문서)에서 Create Policy(정책 생성)를 선택합니다.
4. 업데이트할 정책을 확장한 후 편집할 정책 버전의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 엽니다.
5. AWS Cloud9에서 정책 버전을 열려면 컨텍스트 메뉴에서 View(보기)를 선택하여 정책 버전을 엽니다.
6. 정책 문서가 열리면 변경 사항을 편집하고 저장합니다.

> **Note**
>
> 이 시점에서 정책에 적용한 변경 사항은 로컬 파일 시스템에만 저장됩니다. 버전을 업데이트하고 AWS 탐색기를 사용하여 추적하려면 Upload a new policy version에 나오는 단계를 반복합니다.

새 정책 버전 기본값을 선택하려면

1. AWS 탐색기에서 IoT 서비스 섹션을 확장합니다.
2. Policies(정책) 하위 섹션을 펼치고 업데이트할 정책을 찾습니다.
3. 업데이트할 정책을 확장한 후 설정할 정책 버전의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 연 다음 Set as Default(기본값으로 설정)를 선택합니다.

   이 작업이 완료되면 선택한 새 기본 버전 옆에 별표가 표시됩니다.

정책을 삭제하려면

> **Note**
>
> 정책 또는 정책 버전을 삭제하기 전에 다음 조건이 충족되는지 확인하세요.
>
> - 정책이 인증서에 연결된 경우 정책을 삭제할 수 없습니다.
> - 기본 버전이 아닌 버전이 있는 정책은 삭제할 수 없습니다.
> - 새 기본 버전을 선택하거나 전체 정책을 삭제했다면 정책의 기본 버전만 삭제할 수 있습니다.

- 전체 정책을 삭제하기 전에 동일한 정책의 기본이 아닌 버전을 모두 삭제해야 합니다.

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

Amazon Elastic Container Service 작업

AWS Cloud9 IDE는 Amazon Elastic Container Service(Amazon ECS) 에 대한 몇 가지 지원을 제공합니다. AWS Cloud9 IDE를 사용하여 Amazon ECS 리소스를 관리할 수 있습니다. 예를 들어, 태스크 정의를 만들 수 있습니다.

주제

- AWS Toolkit for AWS Cloud9의 Amazon Elastic Container Service Exec

AWS Toolkit for AWS Cloud9의 Amazon Elastic Container Service Exec

AWS Toolkit for AWS Cloud9을 사용하여 Amazon Elastic Container Service(Amazon ECS) 컨테이너에서 단일 명령을 실행할 수 있습니다. Amazon ECS Exec 기능을 사용하여 이 작업을 수행할 수 있습니다.

Important

Amazon ECS Exec을 활성화 및 비활성화하면 AWS 계정에서 ECS 리소스의 상태가 변경됩니다. 변경 사항에는 서비스 중지 및 재시작이 포함됩니다. 또한 Amazon ECS Exec이 활성화된 상태에서 리소스 상태를 변경하면 예상치 못한 결과가 발생할 수 있습니다. Amazon ECS에 대한 자세한 내용은 Amazon ECS 개발자 안내서의 디버깅에 Amazon ECS Exec 사용을 참조하세요.

Amazon ECS Exec 사전 조건

Amazon ECS Exec 기능을 사용하려면 먼저 충족해야 하는 몇 가지 사전 조건이 있습니다.

Amazon ECS 사전 조건

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

- Amazon EC2를 사용하는 경우 2021년 1월 20일 이후에 출시된 Amazon ECS 최적화 AMI를 에이전트 버전 1.50.2 이상으로 사용해야 합니다. 자세한 내용은 Amazon ECS 개발자 안내서의 Amazon ECS 최적화 AMI를 참조하세요.
- AWS Fargate를 사용하는 경우 플랫폼 버전 1.4.0 이상을 사용해야 합니다. 자세한 내용은 Amazon ECS 개발자 안내서의 AWS Fargate 플랫폼 버전을 참조하세요.

AWS 계정 구성 및 IAM 권한

Amazon ECS Exec 기능을 사용하려면 AWS 계정과 연결된 기존 Amazon ECS 클러스터가 있어야 합니다. Amazon ECS Exec은 Systems Manager를 사용하여 클러스터의 컨테이너와 연결을 설정합니다. Amazon ECS에서는 SSM 서비스와 통신하기 위해 특정 작업 IAM 역할 권한이 필요합니다.

Amazon ECS Exec와 관련된 IAM 역할 및 정책에 대한 자세한 내용은 Amazon ECS 개발자 안내서의 ECS Exec에 필요한 IAM 권한을 참조하세요.

Amazon ECS Exec 작업

AWS Toolkit for AWS Cloud9의 AWS Explorer에서 바로 Amazon ECS Exec을 활성화하거나 비활성화할 수 있습니다. Amazon ECS Exec을 활성화한 후에는 Amazon ECS 메뉴에서 컨테이너를 선택하고 해당 컨테이너에 대해 명령을 실행합니다.

Amazon ECS Exec 활성화

1. AWS Explorer에서 Amazon ECS 메뉴를 찾아 확장합니다.
2. 수정할 서비스가 포함된 클러스터를 확장합니다.
3. 서비스의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Enable Command Execution(명령 실행 활성화)을 선택합니다.

> **Important**
>
> 이 단계는 서비스의 새 배포를 시작하며 몇 분 정도 걸릴 수 있습니다. 자세한 내용은 이 섹션의 시작 부분에 나오는 참고를 참조하세요.

Amazon ECS Exec 비활성화

1. AWS Explorer에서 Amazon ECS 메뉴를 찾아 확장합니다.
2. 원하는 서비스가 포함된 클러스터를 확장합니다.
3. 서비스의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Disable Command Execution(명령 실행 비활성화)을 선택합니다.

> **Important**
>
> 이 단계는 서비스의 새 배포를 시작하며 몇 분 정도 걸릴 수 있습니다. 자세한 내용은 이 섹션의 시작 부분에 나오는 참고를 참조하세요.

컨테이너에 대한 명령 실행

AWS Explorer를 사용하여 컨테이너에 대해 명령을 실행하려면 Amazon ECS Exec을 활성화해야 합니다. 활성화되지 않은 경우 이 섹션의 Amazon ECS Exec 활성화 절차를 참조하세요.

1. AWS Explorer에서 Amazon ECS 메뉴를 찾아 확장합니다.
2. 원하는 서비스가 포함된 클러스터를 확장합니다.
3. 서비스를 확장하여 연결된 컨테이너를 나열합니다.
4. 컨테이너에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Run Command in Container(컨테이너에서 명령 실행)를 선택합니다.
5. 실행 중인 태스크 목록과 함께 프롬프트가 열립니다. 원하는 태스크 ARN을 선택합니다.

> **Note**
>
> 실행 중인 작업이 하나뿐이면 프롬프트가 열리지 않습니다. 대신 태스크가 자동으로 선택됩니다.

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

Amazon EventBridge 작업

AWS Cloud9용 AWS 도구 키트는 Amazon EventBridge를 지원합니다. AWS Cloud9용 AWS 도구 키트를 사용하여 스키마를 비롯한 EventBridg의 특정 측면을 작업할 수 있습니다.

주제

- Amazon EventBridge 스키마 작업

Amazon EventBridge 스키마 작업

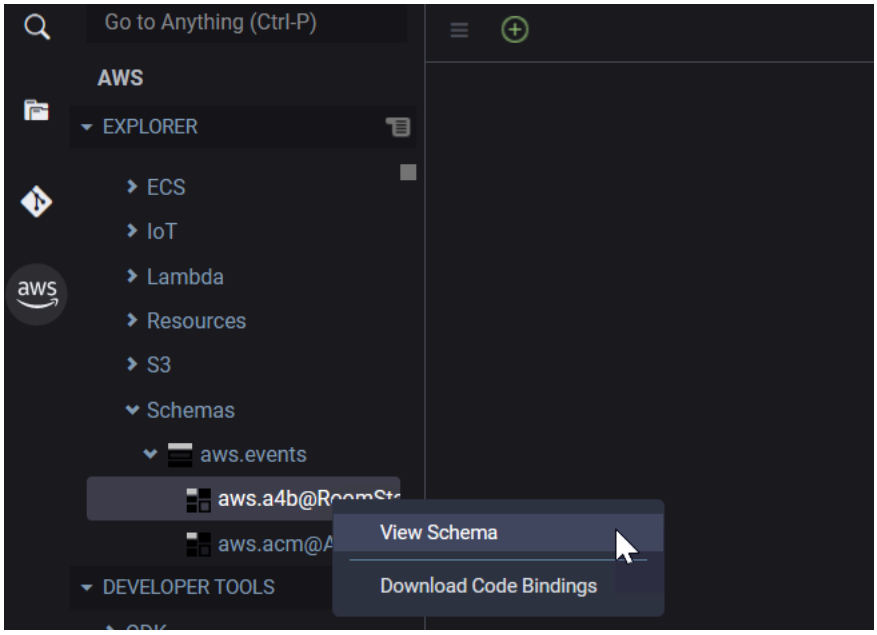
AWS Cloud9용 AWS Toolkit을 사용하여 Amazon EventBridge 스키마에서 다양한 작업을 수행할 수 있습니다.

필수 조건

작업할 EventBridge 스키마를 AWS 계정에서 사용할 수 있어야 합니다. 사용할 수 없다면 스키마를 만들거나 업로드해야 합니다. 자세한 정보는 Amazon EventBridge 사용 설명서의 Amazon EventBridge 스키마를 참조하세요.

사용 가능한 스키마 보기

1. AWS 탐색기에서 스키마를 확장합니다.
2. 확인할 스키마가 포함된 레지스트리의 이름을 확장합니다. 예를 들어, AWS에서 제공하는 대부분의 스키마는 aws.events 레지스트리에 있습니다.
3. 편집기에서 스키마를 보려면 스키마의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 연 다음 View Schema(스키마 보기)를 선택합니다.



사용 가능한 스키마 찾기

AWS 탐색기에서 다음 중 하나 이상을 수행합니다.

- 찾으려는 스키마의 제목을 입력합니다. AWS 탐색기는 일치 항목을 포함하는 스키마 제목을 강조 표시합니다. 강조 표시된 제목을 보려면 레지스트리를 확장해야 합니다.

- Schemas(스키마)에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 연 다음 Search Schemas(스키마 검색)를 선택합니다. 또는 Schemas(스키마)를 확장하고 찾을 스키마가 포함된 레지스트리의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 연 다음 Search Schemas in Registry(레지스트리에서 스키마 검색)를 선택합니다. EventBridge Schemas Search(EventBridge 스키마 검색) 대화 상자에서 찾으려는 스키마의 제목 입력을 시작합니다. 대화 상자에 일치 항목이 포함된 스키마 제목이 표시됩니다.

  대화 상자의 스키마를 표시하려면 스키마의 제목을 선택합니다.

사용 가능한 스키마에 대한 코드 생성

1. AWS 탐색기에서 스키마를 확장합니다.
2. 코드를 생성할 스키마가 포함된 레지스트리의 이름을 확장합니다.
3. 스키마의 제목에 대한 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고 Download code bindings(코드 바인딩 다운로드)를 선택합니다.

4. 결과 마법사 페이지에서 다음을 선택합니다.

   - 스키마의 버전
   - 코드 바인딩 언어
   - 로컬 개발 시스템에서 생성된 코드를 저장하려는 업무 공간 폴더

AWS Cloud9에 대한 자습서

AWS Cloud9를 처음 사용하시나요? 시작하기: 기본 자습서에서 IDE를 둘러보십시오.

이 자습서와 샘플 코드를 사용해 보면서 다양한 프로그래밍 언어 및 AWS 서비스에서 AWS Cloud9을 사용하는 것에 대한 지식과 자신감을 높일 수 있습니다.

주제

- AWS Cloud9용 AWS Command Line Interface 및 aws-shell 자습서
- AWS Cloud9용 AWS CodeCommit 자습서
- AWS Cloud9용 Amazon DynamoDB 자습서
- AWS Cloud9용 AWS CDK 자습서
- AWS Cloud9용 LAMP 자습서
- AWS Cloud9용 WordPress 자습서
- AWS Cloud9용 Java 자습서
- AWS Cloud9용 C++ 자습서
- AWS Cloud9용 Python 자습서
- AWS Cloud9용 .NET 자습서
- Node.js 튜토리얼 AWS Cloud9
- 에 대한 PHP 튜토리얼 AWS Cloud9
- AWS Cloud9의 Ruby
- AWS Cloud9용 Go 자습서
- AWS Cloud9용 TypeScript 자습서
- 에 대한 도커 튜토리얼 AWS Cloud9
- 관련 자습서

AWS Cloud9용 AWS Command Line Interface 및 aws-shell 자습서

다음 자습서를 통해 AWS Command Line Interface(AWS CLI), aws-shell 또는 두 가지 모두를 AWS Cloud9 개발 환경에서 설정할 수 있습니다. AWS CLI 및 aws-shell은 AWS에 포함된 모든 부분과 상호 작용할 수 있는 일관된 인터페이스를 제공하는 통합 도구입니다. AWS Management Console 대신에 AWS CLI을 사용하여 AWS와 상호 작용하는 명령을 신속히 실행할 수 있으며, 이들 명령 중 일부는 AWS CLI 또는 AWS CloudShell로만 실행할 수 있습니다.

AWS CLI에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요. aws-shell의 경우 다음 리소스를 참조하세요.

- GitHub 웹 사이트의 aws-shell
- pip 웹 사이트의 aws-shell

AWS CLI에서 AWS와 상호 작용하기 위해 실행할 수 있는 명령의 목록은 AWS CLI 명령 참조를 참조하세요. AWS CloudShell과 동일한 명령을 사용할 수 있습니다. 단, aws 접두사 없이 명령을 시작해야 합니다.

이 샘플을 생성하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2 및 Amazon S3 같은 서비스에 대한 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금 및 Amazon S3 요금을 참조하세요.

주제

사전 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는 운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 내용은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.
- 기존 환경에 대한 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열 때 AWS Cloud9은 웹 브라우저에서 환경을 위한 IDE를 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기 섹션을 참조하세요.

1단계: 환경에 AWS CLI, aws-shell 또는 두 가지 모두 설치

이 단계에서는 AWS와 상호 작용하는 명령을 실행할 수 있도록 AWS Cloud9 IDE를 사용하여 환경에 AWS CLI, aws-shell 또는 두 가지 모두를 설치합니다.

AWS Cloud9 EC2 개발 환경을 사용하고 AWS CLI만 사용하려는 경우 3단계: 환경에서 AWS CLI 또는 aws-shell을 사용하여 기본 명령 실행 섹션으로 이동할 수 있습니다. 이 단계로 이동할 수 있는 것은 AWS CLI가 EC2 환경에 이미 설치되었고 AWS 액세스 자격 증명 세트가 이미 환경에 설정되어 있기 때문입니다. 자세한 내용은 AWS 관리형 임시 자격 증명 섹션을 참조하세요.

EC2 환경을 사용하지 않는 경우 다음을 수행하여 AWS CLI를 설치합니다.

1. 환경이 열린 상태로, IDE에서 AWS CLI가 이미 설치되었는지 확인합니다. 이 터미널에서 **`aws --version`** 명령을 실행합니다. (터미널 세션을 새로 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다.) AWS CLI가 설치된 경우 Python 버전 번호 및 Amazon EC2 인스턴스 또는 자체 서버의 운영 체제 버전 번호와 같은 정보와 함께 버전 번호가 표시됩니다. AWS CLI가 설치된 경우 2단계: 환경에서 자격 증명 관리 설정로 이동합니다.
2. AWS CLI를 설치하려면 AWS Command Line Interface 사용 설명서의 AWS Command Line Interface 설치를 참조하세요. 예를 들어 Amazon Linux를 실행하는 EC2 환경의 경우, 터미널에서 한 번에 하나씩 이들 세 명령을 실행하여 AWS CLI를 설치합니다.

   ```
   sudo yum -y update          # Install the latest system updates.
   sudo yum -y install aws-cli # Install the AWS CLI.
   aws --version               # Confirm the AWS CLI was installed.
   ```

   예를 들어 Ubuntu Server를 실행하는 EC2 환경의 경우, 터미널에서 한 번에 하나씩 이들 세 명령을 실행하여 AWS CLI를 설치합니다.

   ```
   sudo apt update             # Install the latest system updates.
   sudo apt install -y awscli  # Install the AWS CLI.
   aws --version               # Confirm the AWS CLI was installed.
   ```

aws-shell을 설치하려는 경우 다음을 수행합니다.

1. 환경이 열린 상태로, IDE에서 aws-shell이 이미 설치되었는지 확인합니다. 이 터미널에서 **`aws-shell`** 명령을 실행합니다. (터미널 세션을 새로 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다.) aws-shell이 설치된 경우 `aws>` 프롬프트가 표시됩니다. aws-shell이 설치된 경우 2단계: 환경에서 자격 증명 관리 설정 단원으로 이동합니다.
2. aws-shell을 설치하려면 pip를 사용합니다. pip를 사용하려면 Python이 설치되어 있어야 합니다.

   Python이 이미 설치되어 있는지 확인하려면(또한 필요한 경우 설치하려면) Python 샘플의 1단계: Python 설치에서 설명하는 지침을 따른 다음 이 주제로 돌아옵니다.

pip가 이미 설치되었는지 확인하려면 터미널에서 **pip --version** 명령을 실행합니다. pip가 설치되었으면 버전 번호가 표시됩니다. pip가 설치되어 있지 않은 경우 터미널에서 한 번에 하나씩 이들 세 명령을 실행하여 설치합니다.

```
wget https://bootstrap.pypa.io/get-pip.py # Get the pip install file.
sudo python get-pip.py                    # Install pip. (You might need to run
 'sudo python2 get-pip.py' or 'sudo python3 get-pip.py' instead, depending on how
 Python is installed.)
rm get-pip.py                             # Delete the pip install file, as it is
 no longer needed.
```

3. pip를 사용하여 aws-shell을 설치하려면 다음 명령을 실행합니다.

```
sudo pip install aws-shell
```

2단계: 환경에서 자격 증명 관리 설정

AWS CLI 또는 aws-shell을 사용하여 AWS 서비스를 호출할 때마다 호출에 따른 자격 증명을 제공해야 합니다. 이러한 자격 증명은 AWS CLI 또는 aws-shell이 해당 호출을 할 수 있는 적절한 권한이 있는지 결정합니다. 자격 증명으로 적절한 권한이 확인되지 않는 경우 호출이 실패합니다.

AWS Cloud9 EC2 개발 환경을 사용하는 경우 3단계: 환경에서 AWS CLI 또는 aws-shell을 사용하여 기본 명령 실행 섹션으로 건너뛸 수 있습니다. 이렇게 하는 이유는 EC2 환경에서 이미 자격 증명이 설정되었기 때문입니다. 자세한 내용은 AWS 관리형 임시 자격 증명 섹션을 참조하세요.

EC2 환경을 사용하고 있지 않은 경우 환경에서 자격 증명을 수동으로 저장해야 합니다. 이렇게 하려면 AWS Cloud9의 환경에서 AWS 서비스 호출 섹션의 지침을 수행한 다음 이 주제로 돌아옵니다.

3단계: 환경에서 AWS CLI 또는 aws-shell을 사용하여 기본 명령 실행

이 단계에서는 환경의 AWS CLI 또는 aws-shell을 사용하여 Amazon S3에서 버킷을 생성하고, 사용 가능한 버킷을 나열한 다음 버킷을 삭제합니다.

1. aws-shell을 사용하려고 하지만 아직 시작하지 않은 경우 `aws-shell` 명령을 실행하여 aws-shell을 시작합니다. `aws>` 프롬프트가 표시됩니다.
2. 버킷을 만듭니다. AWS CLI 또는 **s3 mb** 명령을 aws-shell과 함께 사용하여 **aws s3 mb** 명령을 실행함으로써 생성할 버킷의 이름을 제공합니다. 이 예제에서는 `cloud9-123456789012-`

bucket이라는 버킷을 사용합니다. 여기서 123456789012는 AWS 계정 ID입니다. 다른 이름을 사용하는 경우 이 단계 전체에서 해당 이름으로 바꿉니다.

```
aws s3 mb s3://cloud9-123456789012-bucket # For the AWS CLI.
s3 mb s3://cloud9-123456789012-bucket     # For the aws-shell.
```

> **Note**
>
> 버킷 이름은 해당 AWS 계정뿐만 아니라 모든 AWS에서 고유해야 합니다. 앞에서 제안된 버킷 이름은 고유한 버킷 이름을 제공하는 데 도움이 됩니다. `BucketAlreadyExists` 오류를 포함하는 메시지가 표시되면 다른 버킷 이름과 함께 다시 명령을 실행해야 합니다.

3. 사용 가능한 버킷을 나열합니다. AWS CLI 또는 **`s3 ls`** 명령을 aws-shell과 함께 사용하여 **`aws s3 ls`** 명령을 실행합니다. 사용 가능한 버킷의 목록이 표시됩니다.
4. 버킷을 삭제합니다. AWS CLI 또는 **`s3 rb`** 명령을 aws-shell과 함께 사용하여 **`aws s3 rb`** 명령을 실행함으로써 삭제할 버킷의 이름을 제공합니다.

```
aws s3 rb s3://cloud9-123456789012-bucket # For the AWS CLI.
s3 rb s3://cloud9-123456789012-bucket     # For the aws-shell.
```

버킷이 삭제되었는지 확인하려면 다시 AWS CLI 또는 **`s3 ls`** 명령을 aws-shell과 함께 사용하여 **`aws s3 ls`** 명령을 실행합니다. 삭제된 버킷의 이름이 더 이상 목록에 표시되지 않아야 합니다.

> **Note**
>
> 버킷을 계속 사용하려는 경우 해당 버킷을 삭제할 필요가 없습니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷에 객체 추가를 참조하세요. AWS CLI 명령 참조에서 s3 명령도 참조하세요. (버킷을 삭제하지 않는 경우 AWS 계정에 요금이 계속 부과될 수 있습니다.)

AWS CLI로 실험을 계속하려면 AWS Command Line Interface 사용 설명서의 Amazon Web Services 작업과 AWS CLI 명령 참조를 참조하세요. aws-shell 시험을 계속하려면 AWS CLI 명령 참조를 참조하세요. `aws` 접두사를 사용하지 않고 명령을 시작해야 한다는 점에 유의하세요.

4단계: 정리

aws-shell을 사용 중인 경우 **`.exit`** 또는 **`.quit`** 명령을 실행하여 사용을 중지할 수 있습니다.

이 샘플 사용 후 AWS 계정에 계속하여 요금이 부과되지 않도록 환경을 삭제해야 합니다. 지침은 AWS Cloud9에서 환경 삭제 단원을 참조하세요.

AWS Cloud9용 AWS CodeCommit 자습서

AWS CodeCommit 자습서를 사용하면 AWS Cloud9 개발 환경을 설치하여 CodeCommit의 원격 코드 리포지토리와 상호 작용할 수 있습니다. CodeCommit은 AWS 클라우드에 Git 리포지토리를 비공개로 저장하고 관리하는 데 사용할 수 있는 소스 코드 제어 서비스입니다. CodeCommit에 대한 자세한 내용은 AWS CodeCommit 사용 설명서를 참조하세요.

이 자습서를 따르고 이 샘플을 생성하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2 및 CodeCommit 같은 서비스에 대한 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금 및 AWS CodeCommit 요금을 참조하세요.

필수 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는 운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 내용은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.
- 기존 환경에 대한 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열 때 AWS Cloud9은 웹 브라우저에서 환경을 위한 IDE를 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기 섹션을 참조하세요.

1단계: 필요한 액세스 권한을 사용하여 IAM 그룹 설정

AWS 보안 인증 정보가 AWS 계정의 관리자 사용자와 연결되어 있고, 해당 사용자를 이용해 CodeCommit으로 작업하려는 경우 2단계: AWS CodeCommit에서 리포지토리 만들기 섹션으로 진행하세요.

이 단계는 AWS Management Console 또는 AWS 명령줄 인터페이스(AWS CLI)를 사용하여 완료할 수 있습니다.

콘솔을 사용하여 필요한 액세스 권한으로 IAM 그룹 설정

1. 아직 로그인하지 않았다면 AWS Management Console에 로그인합니다.

   이 단계에서는 AWS 계정의 관리자 사용자에 대한 보안 인증 정보를 사용하여 로그인하는 것이 좋습니다. 이렇게 할 수 없으면 AWS 계정 관리자에게 문의하세요.
2. IAM 콘솔(IAM console)을 엽니다. 이렇게 하려면 콘솔의 탐색 모음에서 서비스를 선택합니다. 그런 다음, IAM을 선택합니다.
3. 그룹을 선택합니다.
4. 그룹의 이름을 선택합니다.
5. 권한 탭의 관리형 정책에서 정책 연결을 선택합니다.
6. 정책 이름 목록에서 다음 상자 중 하나를 선택합니다.
   - CodeCommit 및 리포지토리 관련 리소스의 모든 기능에 액세스하려면 AWS CodeCommitPowerUser를 선택하세요. 하지만 이렇게 하면 CodeCommit 리포지토리를 삭제하거나 다른 AWS 서비스에서 Amazon CloudWatch Events와 같은 리포지토리 관련 리소스를 생성 또는 삭제할 수 없습니다.
   - AWS 계정에서 CodeCommit 리포지토리 및 관련 리소스를 완벽하게 제어하려면 AWSCodeCommitFullAccess를 선택합니다. 여기에는 리포지토리 삭제 기능이 포함됩니다.

   목록에서 이러한 정책 이름 중 하나가 보이지 않으면 표시할 정책 이름을 Filter(필터) 상자에 입력합니다.
7. 정책 연결(Attach Policy)을 선택합니다.

이러한 AWS 관리형 정책에서 그룹에 부여하는 액세스 권한 목록을 보려면 AWS User Guide의 AWS CodeCommit Managed (Predefined) Policies for AWS CodeCommit을 참조하세요.

2단계: AWS CodeCommit에서 리포지토리 만들기 섹션으로 진행합니다.

AWS CLI를 사용하여 필요한 액세스 권한으로 IAM 그룹 설정

필요한 액세스 권한을 설명하는 AWS 관리형 정책의 Amazon 리소스 이름(ARN)과 그룹의 이름을 지정하여 IAM `attach-group-policy` 명령을 실행합니다. 구문은 다음과 같습니다.

```
aws iam attach-group-policy --group-name MyGroup --policy-arn POLICY_ARN
```

앞의 명령에서 `MyGroup`을 그룹의 이름으로 바꿉니다. 다음과 같이 `POLICY_ARN`을 AWS 관리형 정책의 ARN으로 바꿉니다.

- CodeCommit 및 리포지토리 관련 리소스의 모든 기능에 액세스하려면 `arn:aws:iam::aws:policy/AWSCodeCommitPowerUser`를 선택하세요. 하지만 이렇게 하면 CodeCommit 리포지토리를 삭제하거나 다른 AWS 서비스에서 Amazon CloudWatch Events와 같은 리포지토리 관련 리소스를 생성 또는 삭제할 수 없습니다.
- AWS 계정에서 CodeCommit 리포지토리 및 관련 리소스를 완벽하게 제어하려면 `arn:aws:iam::aws:policy/AWSCodeCommitFullAccess`를 선택합니다. 여기에는 리포지토리 삭제 기능이 포함됩니다.

이러한 AWS 관리형 정책에서 그룹에 부여하는 액세스 권한 목록을 보려면 AWS User Guide의 AWS CodeCommit Managed (Predefined) Policies for AWS CodeCommit을 참조하세요.

2단계: CodeCommit에서 리포지토리 생성

이 단계에서는 CodeCommit 콘솔을 사용하여 CodeCommit에서 원격 코드 리포지토리를 생성합니다.

이미 CodeCommit 리포지토리가 있으면 3단계: 원격 리포지토리를 환경에 연결 섹션으로 건너뜁니다.

이 단계는 AWS Management Console 또는 AWS 명령줄 인터페이스(AWS CLI)를 사용하여 완료할 수 있습니다.

콘솔을 사용하여 CodeCommit에 리포지토리 생성

1. 이전 단계에서 관리자 사용자로 AWS Management Console에 로그인한 경우 관리자 사용자를 사용하여 리포지토리를 생성하지 않으려는 경우를 가정해보겠습니다. 그런 다음, AWS Management Console에서 로그아웃합니다.
2. https://console.aws.amazon.com/codecommit에서 CodeCommit 콘솔을 엽니다.
3. 콘솔의 탐색 모음에서 리전 선택기를 사용하여 리포지토리를 생성할 AWS 리전(예: 미국 동부(오하이오))을 선택합니다.

4. 시작 페이지가 표시되면 [시작하기(Get started)]를 선택합니다. 그렇지 않은 경우 [리포지토리 생성(Create repository)]을 선택합니다.
5. Create repository(리포지토리 생성) 페이지에서 Repository name(리포지토리 이름)에 새 리포지토리의 이름(예: `MyDemoCloud9Repo`)을 입력합니다. 다른 이름을 선택하는 경우 이 샘플 전체에서 해당 이름으로 바꿉니다.
6. (선택 사항) Description(설명)에 리포지토리에 관한 설명을 입력합니다. 예를 들면 다음과 같이 입력할 수 있습니다. `This is a demonstration repository for the AWS Cloud9 sample.`
7. 리포지토리 생성(Create repository)을 선택합니다. [리포지토리에 연결(Connect to your repository)] 창이 표시됩니다. 이 주제의 뒷부분에서 다른 방법으로 리포지토리에 연결하게 되므로 [닫기(Close)]를 선택합니다.

3단계: 원격 리포지토리를 환경에 연결 섹션으로 진행합니다.

AWS CLI를 사용하여 CodeCommit에 리포지토리 생성

AWS CodeCommit `create-repository` 명령을 실행합니다. 리포지토리 이름, 선택적 설명 및 리포지토리를 생성할 리전인 AWS 리전을 지정합니다.

```
aws codecommit create-repository --repository-name MyDemoCloud9Repo --repository-
description "This is a demonstration repository for the AWS Cloud9 sample." --region
 us-east-2
```

앞의 명령에서 `us-east-2`를 리포지토리를 생성할 AWS 리전의 ID로 바꿉니다. 지원되는 리전 목록은 Amazon Web Services 일반 참조의 AWS CodeCommit 섹션을 참조하세요.

다른 이름을 사용하도록 선택하는 경우 이 샘플 전체에서 해당 이름으로 바꿉니다.

3단계: 원격 리포지토리에 환경을 연결

이 단계에서는 AWS Cloud9 IDE를 사용하여 이전 단계에서 생성하거나 식별한 CodeCommit 리포지토리에 연결합니다.

> **Note**
>
> 시각적 인터페이스를 통해 Git로 작업하는 것을 선호하는 경우 원격 리포지토리를 복제할 수 있습니다. 그런 다음, IDE에서 사용할 수 있는 Git 패널 기능을 사용하여 파일을 추가할 수 있습니다.

사용 중인 AWS Cloud9 개발 환경의 유형에 따라 다음 절차 중 하나를 완료합니다.

환경 유형	따라야 할 절차
EC2 환경	1. IDE의 터미널 세션에서 다음 두 명령을 실행합니다. `git config --global credentia l.helper '!aws codecommit credentia l-helper $@' git config --global credentia l.UseHttpPath true` 자세한 내용은 AWS CodeCommit 사용 설명서의 AWS Cloud9과 AWS CodeCommit 통합에서 2단계: AWS Cloud9 EC2 개발 환경에서 AWS CLI 자격 증명 헬퍼 구성을 참조하세요. 2. 이 주제의 후반부에 나오는 4단계: 원격 리포지토리를 환경에 복제로 건너뜁니다.
SSH 환경	1. Git가 아직 환경에 설치되어 있지 않은 경우, IDE에서 터미널 세션을 사용하여 설치합니다. 자세한 내용은 AWS CodeCommit 사용 설명서의 Linux, macOS 또는 Unix에서 AWS CodeCommit 리포지토리에 대한 SSH 연결을 위한 설정 단계에서 2단계: Git 설치를 참조하세요. 2. AWS CodeCommit 사용 설명서의 Linux, macOS 또는 Unix에서 AWS CodeCommit 리포지토리에 대한 SSH 연결을 위한 설정 단계에서 3단계: Linux, macOS 또는 Unix에 대한 자격 증명 구성을 완료합니다. AWS Management Console에 로그인하고 IAM 콘솔을 열라는 지침이 있다면 AWS 계정에서 관리자 사용자의 보안 인증 정보를 사용

환경 유형	따라야 할 절차
	하여 로그인하는 것이 좋습니다. 이렇게 할 수 없으면 AWS 계정 관리자에게 문의하세요. 3. 이 주제의 후반부에 나오는 4단계: 원격 리포지토리를 환경에 복제로 건너뜁니다.

4단계: 원격 리포지토리를 환경에 복제

이 단계에서는 AWS Cloud9 IDE를 사용하여 환경에 CodeCommit을 설치합니다.

리포지토리를 복제하려면 **`git clone`** 명령을 실행합니다. *`CLONE_URL`*을 리포지토리의 클론 URL로 바꿉니다.

```
git clone CLONE_URL
```

EC2 환경의 경우 `https://`로 시작하는 HTTPS 복제 URL을 제공합니다. SSH 환경의 경우 `ssh://`로 시작하는 SSH 복제 URL을 제공합니다.

리포지토리의 전체 복제 URL을 가져오려면 AWS CodeCommit 사용 설명서에서 AWS CodeCommit 콘솔을 사용하여 리포지토리 세부 정보 보기를 참조하세요.

리포지토리에 파일이 없으면 `You appear to have cloned an empty repository.`와 같은 경고 메시지가 표시됩니다. 이는 예상할 수 있는 경우입니다. 나중에 설명하겠습니다.

5단계: 리포지토리에 파일 추가

이 단계에서는 AWS Cloud9 환경에 복제된 리포지토리로 세 가지 간단한 파일을 생성합니다. 다음으로 복제된 저장소의 Git 스테이징 영역에 파일을 추가합니다. 마지막으로 준비된 파일을 커밋하고 커밋을 CodeCommit의 원격 리포지토리로 푸시합니다.

복제된 리포지토리의 이 위치에 파일이 이미 있는 경우 작업이 완료되었으며 이 샘플의 나머지 부분을 건너뛸 수 있습니다.

리포지토리에 파일을 추가하려면

1. 새 파일을 만듭니다. 메뉴 모음에서 File(파일)과 New File(새 파일)을 선택합니다.
2. 다음 내용을 파일에 입력하고 나서 File(파일), Save(저장)를 선택하여 파일을 AWS Cloud9 환경의 `MyDemoCloud9Repo` 디렉터리에 `bird.txt`로 저장합니다.

```
bird.txt
--------
Birds are a group of endothermic vertebrates, characterized by feathers,
toothless beaked jaws, the laying of hard-shelled eggs, a high metabolic
rate, a four-chambered heart, and a lightweight but strong skeleton.
```

> **Note**
>
> 이 파일을 올바른 디렉터리로 저장하는지 확인하려면 Save As(다른 이름으로 저장) 대화 상자에서 MyDemoCloud9Repo 폴더를 선택하세요. 그런 다음, Folder(폴더)에서 /MyDemoCloud9Repo이 표시되는지 확인합니다.

3. 이름이 `insect.txt` 및 `reptile.txt`인 파일을 다음과 같은 내용으로 두 개 더 만듭니다. MyDemoCloud9Repo 디렉터리에 파일을 저장합니다.

```
insect.txt
----------
Insects are a class of invertebrates within the arthropod phylum that
have a chitinous exoskeleton, a three-part body (head, thorax, and abdomen),
three pairs of jointed legs, compound eyes, and one pair of antennae.
```

```
reptile.txt
-----------
Reptiles are tetrapod (four-limbed vertebrate) animals in the class
Reptilia, comprising today's turtles, crocodilians, snakes,
amphisbaenians, lizards, tuatara, and their extinct relatives.
```

4. 이 터미널에서는 **cd** 명령을 실행하여 MyDemoCloud9Repo 디렉터리로 바꿉니다.

```
cd MyDemoCloud9Repo
```

5. **git status** 명령을 실행하여 파일이 MyDemoCloud9Repo 디렉터리에 성공적으로 저장되었는지 확인합니다. 두 개의 파일 모두 추적되지 않는 파일로 나열됩니다.

```
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        bird.txt
        insect.txt
```

```
        reptile.txt
```

6. **`git add`** 명령을 실행하여 파일을 Git 스테이징 지역에 추가합니다.

```
git add --all
```

7. **`git status`** 명령을 다시 실행하여 파일이 Git 스테이징 지역에 성공적으로 추가되었는지 확인합니다. 변경 사항이 커밋되면 세 개 파일 모두가 나열됩니다.

```
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   bird.txt
        new file:   insect.txt
        new file:   reptile.txt
```

8. **`git commit`** 명령을 실행하여 스테이징된 파일을 커밋합니다.

```
git commit -m "Added information about birds, insects, and reptiles."
```

9. **`git push`** 명령을 실행하여 CodeCommit에서 원격 리포지토리로 커밋을 푸시합니다.

```
git push -u origin master
```

10. 파일이 성공적으로 푸시되었는지 확인합니다. 열려 있지 않은 경우 https://console.aws.amazon.com/codecommit에서 CodeCommit 콘솔을 엽니다.
11. 상단 탐색 모음의 오른쪽 가장자리 근처에서 리포지토리를 생성한 AWS 리전(예: 미국 동부(오하이오))을 선택합니다.
12. [대시보드(Dashboard)] 페이지에서 MyDemoCloud9Repo를 선택합니다. 세 개의 파일이 표시됩니다.

CodeCommit 리포지토리를 계속 실험하려면 AWS CodeCommit 사용 설명서에서 리포지토리 콘텐츠 검색을 참조하세요.

Git에 익숙하지 않고 CodeCommit 리포지토리에 서툴다면 Git 시도하기의 샘플 Git 리포지토리를 먼저 경험해보세요.

6단계: 정리

이 샘플 사용 후 AWS 계정에 계속하여 요금이 부과되지 않도록 CodeCommit 리포지토리를 삭제합니다. 지침은 AWS CodeCommit 사용 설명서에서 AWS CodeCommit 리포지토리 삭제를 참조하세요.

또한 환경도 삭제해야 합니다. 지침에 대한 내용은 환경 삭제를 참조하십시오.

AWS Cloud9용 Amazon DynamoDB 자습서

이 자습서를 사용하면 Amazon DynamoDB와 함께 작동하도록 AWS Cloud9 개발 환경을 설정할 수 있습니다.

DynamoDB는 완전관리형 NoSQL 데이터베이스 서비스입니다. DynamoDB를 사용하여 데이터 규모에 관계없이 데이터를 저장 및 검색하고, 어떤 수준의 요청 트래픽이라도 처리할 수 있는 데이터베이스 테이블을 생성할 수 있습니다. DynamoDB는 테이블의 데이터와 트래픽을 충분한 수의 서버로 자동 분산하여 지정한 요청 용량과 저장된 데이터 규모를 처리하면서도 일관되고 빠른 성능을 발휘합니다. 자세한 내용은 AWS 웹 사이트에서 Amazon DynamoDB를 참조하세요.

이 샘플을 생성하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2 및 DynamoDB 같은 서비스에 대한 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금 및 Amazon DynamoDB 요금을 참조하세요.

추가 AWS데이터베이스 제품 및 서비스에 대한 자세한 내용은 AWS 웹 사이트에서 Amazon Relational Database Service(RDS), Amazon ElastiCache 및 Amazon Redshift를 참조하세요. AWS 웹 사이트에서 AWS Database Migration Service 섹션도 참조하세요.

필수 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는 운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 내용은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.
- 기존 환경에 대한 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열 때 AWS Cloud9은 웹 브라우저에서 환경을 위한 IDE를 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기 섹션을 참조하세요.

1단계: 환경에 AWS CLI, AWS CloudShell 또는 두 가지 모두 설치 및 구성

이 단계에서는 DynamoDB와 상호 작용하는 명령을 실행할 수 있도록 AWS Cloud9 IDE를 사용하여 환경에 AWS CLI, AWS CloudShell 또는 두 가지 모두를 설치하고 구성합니다. 그런 다음 AWS CLI를 사용하여 기본 DynamoDB 명령을 실행함으로써 설치 및 구성을 테스트합니다.

1. AWS CLI 또는 AWS CloudShell에 대한 자격 증명 관리를 설정하려면 AWS CLI, AWS CloudShell 또는 두 가지 모두를 환경에 설치하고 AWS CLI 및 AWS CloudShell 샘플의 1단계와 2단계를 따른 다음 이 주제로 돌아옵니다. AWS CLI, AWS CloudShell 또는 두 가지 모두를 환경에 이미 설치 및 구성한 경우, 다시 설치할 필요가 없습니다.
2. 환경의 터미널 세션에서 DynamoDB **`list-tables`** 명령을 실행하여 기존 DynamoDB 테이블을 나열(있는 경우)하는 방법으로 AWS CLI, aws-shell 또는 두 가지 모두의 설치 및 구성을 테스트 합니다. 터미널 세션을 새로 시작하려면 메뉴 모음에서 [창(Window)], [새 터미널(New Terminal)]을 선택합니다.

```
aws dynamodb list-tables # For the AWS CLI.
dynamodb list-tables     # For the aws-shell.
```

> **Note**
>
> 이 샘플 전체에서 aws-shell을 사용하는 경우 aws로 시작하는 각 명령에서 aws를 생략합니다. aws-shell을 시작하려면 **`aws-shell`** 명령을 실행합니다. aws-shell의 사용을 중지하려면 **`.exit`** 또는 **`.quit`** 명령을 실행합니다.

이 명령이 성공하면 이미 있는 기존 DynamoDB 테이블 목록이 포함된 `TableNames` 배열이 출력됩니다. 아직 DynamoDB 테이블이 없는 경우 `TableNames` 배열은 비어 있습니다.

```
{
  "TableNames": []
}
```

DynamoDB 테이블이 있는 경우 `TableNames` 배열에는 테이블 이름 목록이 들어 있습니다.

2단계: 테이블 생성

이 단계에서는 DynamoDB에서 테이블을 생성하고 테이블의 이름, 레이아웃, 단순 기본 키 및 데이터 처리량 설정을 지정합니다.

`Weather`라는 이 샘플 테이블에는 미국의 일부 도시에 대한 일기 예보에 대한 정보가 포함되어 있습니다. 이 테이블에는 다음과 같은 유형의 정보가 들어 있습니다(DynamoDB에서는 각 정보를 속성이라고 함).

- 필요한 고유 도시 ID(`CityID`)
- 필수 예보 일자(`Date`)
- 도시 이름(`City`)
- 주 이름(`State`)
- 예보 기상 조건(`Conditions`)
- 예보 기온(`Temperatures`)
  - 예보 최고 기온, 화씨 단위(`HighF`)
  - 예보 최저 기온, 화씨 단위(`LowF`)

테이블을 만들려면 AWS Cloud9 IDE의 터미널 세션에서 DynamoDB **`create-table`** 명령을 실행합니다.

```
aws dynamodb create-table \
--table-name Weather \
--attribute-definitions \
  AttributeName=CityID,AttributeType=N AttributeName=Date,AttributeType=S \
--key-schema \
```

```
  AttributeName=CityID,KeyType=HASH AttributeName=Date,KeyType=RANGE \
--provisioned-throughput ReadCapacityUnits=5,WriteCapacityUnits=5
```

이 명령에서:

- `--table-name`은 테이블 이름을 나타냅니다(이 샘플의 경우 `Weather`). 테이블 이름은 AWS 계정의 각 AWS 리전 내에서 고유해야 합니다.
- `--attribute-definitions`는 테이블 항목을 고유하게 식별하는 데 사용되는 속성을 나타냅니다. 이 테이블의 각 항목은 숫자 `ID` 속성과 ISO-8601 형식의 문자열로 표현되는 `Date` 속성의 조합으로 고유하게 식별됩니다.
- `--key-schema`는 테이블의 키 스키마를 나타냅니다. 이 테이블은 `CityID` 및 `Date`의 복합 기본 키를 갖고 있습니다. 즉, 각 테이블 항목은 반드시 `CityID` 속성 값과 `Date` 속성 값을 가져야 하지만, 테이블의 두 항목이 모두 동일한 `CityID` 속성 값과 `Date` 속성 값을 가질 수 있습니다.
- `--provisioned-throughput`은 테이블의 읽기/쓰기 용량을 나타냅니다. DynamoDB는 최대 4KB 크기의 항목에 대해 초당 강력히 일관된 읽기 5회 또는 최대 4KB 크기의 항목에 대해 초당 최종적으로 일관된 읽기 5회를 허용합니다. 또한 DynamoDB는 크기가 최대 1KB인 항목에 대해 초당 최대 5회의 쓰기를 허용합니다.

> **Note**
>
> 프로비저닝된 처리량을 높게 설정하면 AWS 계정에 추가 요금이 발생합니다.
> 이 명령과 다른 DynamoDB 명령에 대한 자세한 내용은 AWS CLI 명령 참조에서 dynamodb를 참조하세요.

이 명령이 성공하면 생성 중인 새 테이블에 대한 요약 정보가 표시됩니다. 테이블이 성공적으로 생성되었는지 확인하려면 테이블 이름(`--table-name`)을 지정하여 DynamoDB **`describe-table`** 명령을 실행합니다.

```
aws dynamodb describe-table --table-name Weather
```

테이블이 성공적으로 생성되면 `TableStatus` 값이 `CREATING`에서 `ACTIVE`로 바뀝니다. 테이블이 성공적으로 생성될 때까지 이 단계를 넘어가지 마세요.

3단계: 테이블에 항목 추가

이 단계에서는 방금 생성한 테이블에 항목을 추가합니다.

1. 다음 콘텐츠가 포함된 `weather-item.json`이라는 파일을 생성합니다. 새 파일을 생성하려면 메뉴 모음에서 [파일(File)], [새 파일(New File)]을 선택합니다. 파일을 저장하려면 File(파일), Save(저장)를 선택합니다.

```
{
  "CityID": { "N": "1" },
  "Date": { "S": "2017-04-12" },
  "City": { "S": "Seattle" },
  "State": { "S": "WA" },
  "Conditions": { "S": "Rain" },
  "Temperatures": { "M": {
      "HighF": { "N": "59" },
      "LowF": { "N": "46" }
    }
  }
}
```

   이 코드에서 `N`은 숫자인 속성 값을 나타냅니다. `S`는 문자열 속성 값입니다. `M`은 속성-값 페어의 세트인 맵 속성입니다. 항목 작업을 할 때마다 속성의 데이터 유형을 지정해야 합니다. 사용 가능한 추가 속성 데이터 형식은 Amazon DynamoDB 개발자 가이드에서 데이터 형식을 참조하세요.

2. 테이블 이름(`--table-name`)과 JSON 형식의 항목(`--item`)에 대한 경로를 지정하여 DynamoDB **`put-item`** 명령을 실행합니다.

```
aws dynamodb put-item \
--table-name Weather \
--item file://weather-item.json
```

   명령이 성공하면 오류 없이 실행되며 확인 메시지가 표시되지 않습니다.

3. 테이블의 현재 콘텐츠를 확인하려면 테이블 이름(`--table-name`)을 지정하여 DynamoDB **`scan`** 명령을 실행합니다.

```
aws dynamodb scan --table-name Weather
```

   명령이 성공하면 테이블 및 방금 추가한 항목에 대한 요약 정보가 표시됩니다.

4단계: 테이블에 여러 항목 추가

이 단계에서는 `Weather` 테이블에 항목을 몇 개 더 추가합니다.

1. 다음 콘텐츠가 포함된 more-weather-items.json이라는 파일을 생성합니다.

```
{
  "Weather": [
    {
      "PutRequest": {
        "Item": {
          "CityID": { "N": "1" },
          "Date": { "S": "2017-04-13" },
          "City": { "S": "Seattle" },
          "State": { "S": "WA" },
          "Conditions": { "S": "Rain" },
          "Temperatures": { "M": {
              "HighF": { "N": "52" },
              "LowF": { "N": "43" }
            }
          }
        }
      }
    },
    {
      "PutRequest": {
        "Item": {
          "CityID": { "N": "1" },
          "Date": { "S": "2017-04-14" },
          "City": { "S": "Seattle" },
          "State": { "S": "WA" },
          "Conditions": { "S": "Rain" },
          "Temperatures": { "M": {
              "HighF": { "N": "49" },
              "LowF": { "N": "43" }
            }
          }
        }
      }
    },
    {
      "PutRequest": {
        "Item": {
          "CityID": { "N": "2" },
          "Date": { "S": "2017-04-12" },
          "City": { "S": "Portland" },
          "State": { "S": "OR" },
```

```
          "Conditions": { "S": "Thunderstorms" },
          "Temperatures": { "M": {
              "HighF": { "N": "59" },
              "LowF": { "N": "43" }
            }
          }
        }
      }
    },
    {
      "PutRequest": {
        "Item": {
          "CityID": { "N": "2" },
          "Date": { "S": "2017-04-13" },
          "City": { "S": "Portland" },
          "State": { "S": "OR" },
          "Conditions": { "S": "Rain" },
          "Temperatures": { "M": {
              "HighF": { "N": "51" },
              "LowF": { "N": "41" }
            }
          }
        }
      }
    },
    {
      "PutRequest": {
        "Item": {
          "CityID": { "N": "2" },
          "Date": { "S": "2017-04-14" },
          "City": { "S": "Portland" },
          "State": { "S": "OR" },
          "Conditions": { "S": "Rain Showers" },
          "Temperatures": { "M": {
              "HighF": { "N": "49" },
              "LowF": { "N": "39" }
            }
          }
        }
      }
    },
    {
      "PutRequest": {
        "Item": {
```

```
          "CityID": { "N": "3" },
          "Date": { "S": "2017-04-12" },
          "City": { "S": "Portland" },
          "State": { "S": "ME" },
          "Conditions": { "S": "Rain" },
          "Temperatures": { "M": {
              "HighF": { "N": "59" },
              "LowF": { "N": "40" }
            }
          }
        }
      },
      {
        "PutRequest": {
          "Item": {
            "CityID": { "N": "3" },
            "Date": { "S": "2017-04-13" },
            "City": { "S": "Portland" },
            "State": { "S": "ME" },
            "Conditions": { "S": "Partly Sunny" },
            "Temperatures": { "M": {
                "HighF": { "N": "54" },
                "LowF": { "N": "37" }
              }
            }
          }
        }
      },
      {
        "PutRequest": {
          "Item": {
            "CityID": { "N": "3" },
            "Date": { "S": "2017-04-14" },
            "City": { "S": "Portland" },
            "State": { "S": "ME" },
            "Conditions": { "S": "Mostly Sunny" },
            "Temperatures": { "M": {
                "HighF": { "N": "53" },
                "LowF": { "N": "37" }
              }
            }
          }
        }
```

```
    }
  ]
}
```

이 코드에서는 8개의 `Item` 객체는 이전 단계에서 정의한 단일 항목과 유사하게 테이블에 추가할 8개 항목을 정의합니다. 그러나 다음 단계에서 DynamoDB **batch-write-item** 명령을 실행할 때 포함하는 `PutRequest` 객체에 각 `Item` 객체를 포함하는 JSON 형식 객체를 제공해야 합니다. 그런 다음 해당 `PutRequest` 객체를 테이블과 같은 이름의 상위 배열에 포함해야 합니다.

2. 추가할 JSON 형식의 항목(`--request-items`)에 대한 경로를 지정하여 DynamoDB **batch-write-item** 명령을 실행합니다.

```
aws dynamodb batch-write-item \
--request-items file://more-weather-items.json
```

명령이 성공하면 항목이 성공적으로 추가되었음을 확인하는 다음 메시지가 표시됩니다.

```
{
  "UnprocessedItems": {}
}
```

3. 테이블의 현재 콘텐츠를 확인하려면 DynamoDB **scan** 명령을 다시 실행합니다.

```
aws dynamodb scan --table-name Weather
```

이 명령이 성공하면 9개의 항목이 표시됩니다.

5단계: 글로벌 보조 인덱스 생성

DynamoDB **scan** 명령을 실행하여 항목에 대한 정보를 얻으려면 속도가 느릴 수 있습니다. 테이블의 크기가 커지거나 얻으려는 정보 유형이 복잡할 때 특히 그렇습니다. 하나 이상의 보조 인덱스를 생성하여 보다 쉽게 정보를 얻을 수 있습니다. 이 단계에서는 DynamoDB 가 지원하는 두 가지 유형의 보조 인덱스에 대해 알아봅니다. 이를 로컬 보조 인덱스 및 글로벌 보조 인덱스라고 합니다. 다음으로 글로벌 보조 인덱스를 생성합니다.

이러한 보조 인덱스 유형을 이해하려면 먼저 테이블의 항목을 고유하게 식별하는 기본 키에 대해 알아야합니다. DynamoDB는 단순 기본 키 또는 복합 기본 키를 지원합니다. 단순 기본 키에는 단일 속성이 있으며 해당 속성 값은 테이블의 각 항목에 대해 고유해야 합니다. 이 속성을 파티션 키(또는해시 속성)이라고도 하며 DynamoDB가 빠른 액세스를 위해 항목을 분할하는 데 이 속성을 사용할 수 있습니다.

테이블은 두 개의 속성을 포함하는 복합 기본 키를 가질 수도 있습니다. 첫 번째 속성은 파티션 키이고, 두 번째 속성은 정렬 키(범위 속성이라고도 함)입니다. 복합 기본 키가 있는 테이블에서 두 항목은 동일한 파티션 키 값을 가질 수 있지만 동일한 정렬 키 값까지 가질 수는 없습니다. `Weather` 테이블은 복합 기본 키를 갖고 있습니다.

로컬 보조 인덱스의 파티션 키는 테이블 자체와 동일하지만, 이 인덱스 유형의 정렬 키는 다를 수 있습니다. 글로벌 보조 인덱스는 파티션 키 및 정렬 키가 모두 테이블 차제와 다를 수 있습니다.

예를 들어 기본 키를 사용하여 `CityID`를 기준으로 `Weather` 항목에 액세스할 수 있습니다. `State`를 기준으로 `Weather` 항목에 액세스하려면 파티션 키가 `CityID`(테이블 자체와 동일해야 함)이고 정렬 키가 `State`인 로컬 보조 인덱스를 생성할 수 있습니다. `City`를 기준으로 `Weather` 항목에 액세스하려면 파티션 키가 `City`이고 정렬 키가 `Date`인 글로벌 보조 인덱스를 생성할 수 있습니다.

테이블을 생성하는 동안에만 로컬 보조 인덱스를 생성할 수 있습니다. `Weather` 테이블이 이미 있기 때문에 로컬 보조 인덱스를 추가할 수 없습니다. 하지만 보조 인덱스는 추가할 수 있습니다. 지금 추가해 보세요.

> **Note**
>
> 보조 인덱스를 생성하면 AWS 계정에 추가 요금이 발생할 수 있습니다.

1. 다음 콘텐츠가 포함된 `weather-global-index.json`이라는 파일을 생성합니다.

```
[
  {
    "Create": {
      "IndexName": "weather-global-index",
      "KeySchema": [
        {
          "AttributeName": "City",
          "KeyType": "HASH"
        },
        {
          "AttributeName": "Date",
          "KeyType": "RANGE"
        }
      ],
      "Projection": {
        "ProjectionType": "INCLUDE",
        "NonKeyAttributes": [
```

```
          "State",
          "Conditions",
          "Temperatures"
        ]
      },
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 5
      }
    }
  }
]
```

이 코드에는 다음이 포함됩니다.

- 이 글로벌 보조 인덱스의 이름은 `weather-global-index`입니다.
- `City` 속성은 파티션 키(해시 속성)이며 `Date` 속성은 정렬 키(범위 속성)입니다.
- `Projection`은 이 인덱스를 사용하는 테이블 검색과 일치하는 모든 항목에 대해 기본적으로 검색할 속성(해시 속성 및 범위 속성 제외)을 정의합니다. 이 샘플에서 `State`, `Conditions`, `HighF`(`Temperatures`의 일부) 및 `LowF`(역시 `Temperatures`의 일부) 속성(`City` 및 `Date`속성 포함)은 일치하는 모든 항목에 대해 검색됩니다.
- 테이블과 마찬가지로 글로벌 보조 인덱스는 프로비저닝된 처리량 설정을 정의해야 합니다.
- `IndexName`, `KeySchema`, `Projection` 및 `ProvisionedThroughput` 설정이 `Create` 객체에 포함되어야 합니다. 이 객체는 다음 단계에서 DynamoDB **`update-table`** 명령을 실행할 때 생성할 글로벌 보조 인덱스를 정의합니다.

2. DynamoDB **`update-table`** 명령을 실행합니다.

```
aws dynamodb update-table \
--table-name Weather \
--attribute-definitions \
  AttributeName=City,AttributeType=S AttributeName=Date,AttributeType=S \
--global-secondary-index-updates file://weather-global-index.json
```

이 명령에서:

- `--table-name`은 업데이트할 테이블의 이름입니다.
- `--attribute-definitions`는 인덱스에 포함할 속성입니다. 파티션 키가 항상 먼저 나열되고 정렬 키는 항상 두 번째로 나열됩니다.

- `--global-secondary-index-updates`는 글로벌 보조 인덱스를 정의하는 파일의 경로입니다.

이 명령이 성공하면 생성 중인 새 글로벌 보조 인덱스에 대한 요약 정보가 표시됩니다. 글로벌 보조 인덱스가 성공적으로 생성되었는지 확인하려면 테이블 이름(`--table-name`)을 지정하여 DynamoDB **`describe-table`** 명령을 실행합니다.

```
aws dynamodb describe-table --table-name Weather
```

글로벌 보조 인덱스가 성공적으로 생성되면 `TableStatus` 값이 `UPDATING`에서 `ACTIVE`로 바뀌고 `IndexStatus`값이 `CREATING`에서 `ACTIVE`로 바뀝니다. 글로벌 보조 인덱스가 성공적으로 생성될 때까지 이 단계를 넘어가지 마세요. 몇 분 정도 걸릴 수 있습니다.

6단계: 테이블에서 항목 가져오기

테이블에서 항목을 가져오는 방법에는 여러 가지가 있습니다. 이 단계에서는 테이블의 기본 키를 사용하고, 테이블의 다른 속성을 사용하고, 글로벌 보조 인덱스를 사용하여 항목을 가져옵니다.

항목의 기본 키 값을 기준으로 테이블에서 단일 항목을 가져오려면

항목의 기본 키 값을 알고 있는 경우 DynamoDB 명령 **`get-item`** , **`scan`** 또는 **`query`**를 실행하여 일치하는 항목을 가져올 수 있습니다. 이들 명령의 주요 차이점은 다음과 같습니다.

- **`get-item`**은 지정된 기본 키와 함께 항목에 대한 속성 세트를 반환합니다.
- **`scan`**은 테이블 또는 보조 인덱스의 모든 항목에 액세스하여 하나 이상의 항목 및 항목 속성을 반환합니다.
- **`query`**는 기본 키 값을 기반으로 항목을 찾습니다. 복합 기본 키(파티션 키 및 정렬 키)가 있는 테이블 또는 보조 인덱스를 쿼리할 수 있습니다.

이 샘플에서는 이러한 각 명령을 사용하여 `CityID` 속성 값 `1` 및 `Date` 속성 값 `2017-04-12`를 포함하는 항목을 가져오는 방법을 보여줍니다.

1. DynamoDB **`get-item`** 명령을 실행하려면 테이블의 이름(`--table-name`), 기본 키 값(`--key`) 및 표시할 항목의 속성 값(`--projection-expression`)을 지정합니다. `Date`는 DynamoDB에서 예약된 키워드이므로 `Date` 속성 값(`--expression-attribute-names`)에 대한 별칭도 제공해야 합니다(`State`도 예약된 키워드이므로 이후 단계에서 별칭이 제공되는 것을 확인할 수 있음).

```
aws dynamodb get-item \
--table-name Weather \
--key '{ "CityID": { "N": "1" }, "Date": { "S": "2017-04-12" } }' \
--projection-expression \
  "City, #D, Conditions, Temperatures.HighF, Temperatures.LowF" \
--expression-attribute-names '{ "#D": "Date" }'
```

이 명령과 다른 명령에서 항목의 모든 속성을 표시하려면 `--projection-expression`을 포함하지 마세요. 이 예에서는 `--projection-expression`을 포함하지 않으므로 `--expression-attribute-names`를 포함할 필요도 없습니다.

```
aws dynamodb get-item \
--table-name Weather \
--key '{ "CityID": { "N": "1" }, "Date": { "S": "2017-04-12" } }'
```

2. DynamoDB **scan** 명령을 실행하려면 다음을 지정합니다.
   - 테이블의 이름(`--table-name`).
   - 실행할 검색(`--filter-expression`).
   - 사용할 검색 조건(`--expression-attribute-values`).
   - 일치하는 항목에 대해 표시할 속성 종류(`--select`).
   - 표시할 항목에 대한 속성 값(`--projection-expression`).
   - DynamoDB에서 예약된 키워드를 사용하는 속성이 있는 경우 해당 속성의 별칭(`--expression-attribute-names`).

```
aws dynamodb scan \
--table-name Weather \
--filter-expression "(CityID = :cityID) and (#D = :date)" \
--expression-attribute-values \
  '{ ":cityID": { "N": "1" }, ":date": { "S": "2017-04-12" } }' \
--select SPECIFIC_ATTRIBUTES \
--projection-expression \
  "City, #D, Conditions, Temperatures.HighF, Temperatures.LowF" \
--expression-attribute-names '{ "#D": "Date" }'
```

3. DynamoDB **query** 명령을 실행하려면 다음을 지정합니다.
   - 테이블의 이름(`--table-name`).
   - 실행할 검색(`--key-condition-expression`).

- 검색에 사용할 속성 값(--expression-attribute-values).
- 일치하는 항목에 대해 표시할 속성 종류(--select).
- 표시할 항목에 대한 속성 값(--projection-expression).
- DynamoDB에서 예약된 키워드를 사용하는 속성이 있는 경우 해당 속성의 별칭(--expression-attribute-names).

```
aws dynamodb query \
--table-name Weather \
--key-condition-expression "(CityID = :cityID) and (#D = :date)" \
--expression-attribute-values \
  '{ ":cityID": { "N": "1" }, ":date": { "S": "2017-04-12" } }' \
--select SPECIFIC_ATTRIBUTES \
--projection-expression \
  "City, #D, Conditions, Temperatures.HighF, Temperatures.LowF" \
--expression-attribute-names '{ "#D": "Date" }'
```

scan 명령은 결과를 얻기 위해 9개의 항목을 모두 스캔하는 데 필요한 반면, **query** 명령은 1개의 항목만 검색하는 데 필요합니다.

항목의 기본 키 값을 기준으로 테이블에서 여러 항목을 가져오려면

항목의 기본 키 값을 알고 있는 경우 DynamoDB **batch-get-item** 명령을 실행하여 일치하는 항목을 가져올 수 있습니다. 이 샘플에서는 이러한 각 명령을 사용하여 CityID 속성 값 3 및 Date 속성 값 2017-04-13 또는 2017-04-14를 포함하는 항목을 가져오는 방법을 보여줍니다.

가져올 항목을 설명하는 파일의 경로(--request-items)를 지정하여 DynamoDB **batch-get-item** 명령을 실행합니다.

```
aws dynamodb batch-get-item --request-items file://batch-get-item.json
```

이 샘플의 경우 batch-get-item.json 파일의 코드는 Weather 테이블에서 CityID가 3이고 Date가 2017-04-13 또는 2017-04-14인 항목을 검색하도록 지정합니다. 검색된 각 항목에 대해 City, State, Date 및 HighF(Temperatures의 일부)의 속성 값이 표시됩니다(있는 경우).

```
{
  "Weather" : {
    "Keys": [
      {
```

```
        "CityID": { "N": "3" },
        "Date": { "S": "2017-04-13" }
      },
      {
        "CityID": { "N": "3" },
        "Date": { "S": "2017-04-14" }
      }
    ],
    "ProjectionExpression": "City, #S, #D, Temperatures.HighF",
    "ExpressionAttributeNames": { "#S": "State", "#D": "Date" }
  }
}
```

테이블에서 일치하는 모든 항목을 가져오려면

테이블의 속성 값에 대해 알고 있는 경우 DynamoDB **scan** 명령을 실행하여 일치하는 항목을 가져올 수 있습니다. 이 샘플에서는 `Conditions` 속성 값에 `Sunny`가 포함되어 있고 `HighF` 속성 값(`Temperatures`의 일부)이 `53`보다 클 경우 날짜를 가져오는 방법을 보여줍니다.

다음을 지정하여 DynamoDB **scan** 명령을 실행합니다.

- 테이블의 이름(`--table-name`).
- 실행할 검색(`--filter-expression`).
- 사용할 검색 조건(`--expression-attribute-values`).
- 일치하는 항목에 대해 표시할 속성 종류(`--select`).
- 표시할 항목에 대한 속성 값(`--projection-expression`).
- DynamoDB에서 예약된 키워드를 사용하는 속성이 있는 경우 해당 속성의 별칭(`--expression-attribute-names`).

```
aws dynamodb scan \
--table-name Weather \
--filter-expression \
  "(contains (Conditions, :sun)) and (Temperatures.HighF > :h)" \
--expression-attribute-values \
  '{ ":sun": { "S" : "Sunny" }, ":h": { "N" : "53" } }' \
--select SPECIFIC_ATTRIBUTES \
--projection-expression "City, #S, #D, Conditions, Temperatures.HighF" \
--expression-attribute-names '{ "#S": "State", "#D": "Date" }'
```

글로벌 보조 인덱스에서 일치하는 모든 항목을 가져오려면

글로벌 보조 인덱스를 사용하여 검색하려면 DynamoDB **query** 명령을 사용합니다. 이 샘플에서는 `weather-global-index` 보조 인덱스를 사용하여 `Portland`라는 도시의 `2017-04-13` 및 `2017-04-14` 날짜 예보 조건을 가져오는 방법을 보여줍니다.

다음을 지정하여 DynamoDB **query** 명령을 실행합니다.

- 테이블의 이름(`--table-name`).
- 글로벌 보조 인덱스의 이름(`--index-name`).
- 실행할 검색(`--key-condition-expression`).
- 검색에 사용할 속성 값(`--expression-attribute-values`).
- 일치하는 항목에 대해 표시할 속성 종류(`--select`).
- DynamoDB에서 예약된 키워드를 사용하는 속성이 있는 경우 해당 속성의 별칭(`--expression-attribute-names`).

```
aws dynamodb query \
--table-name Weather \
--index-name weather-global-index \
--key-condition-expression "(City = :city) and (#D between :date1 and :date2)" \
--expression-attribute-values \
  '{ ":city": { "S" : "Portland" }, ":date1": { "S": "2017-04-13" }, ":date2": { "S":
 "2017-04-14" } }' \
--select SPECIFIC_ATTRIBUTES \
--projection-expression "City, #S, #D, Conditions, Temperatures.HighF" \
--expression-attribute-names '{ "#S": "State", "#D": "Date" }'
```

7단계: 정리

이 샘플 사용 후 AWS 계정에 계속하여 요금이 부과되지 않도록 테이블을 삭제해야 합니다. 테이블을 삭제하면 글로벌 보조 인덱스도 삭제됩니다. 또한 환경도 삭제해야 합니다.

테이블을 삭제하려면 테이블 이름(`--table-name`)을 지정하여 DynamoDB **delete-table** 명령을 실행합니다.

```
aws dynamodb delete-table --table-name Weather
```

명령이 성공하면 `TableStatus` 값 `DELETING`을 포함하여 테이블에 대한 정보가 표시됩니다.

테이블이 성공적으로 삭제되었는지 확인하려면 테이블 이름(`--table-name`)을 지정하여 DynamoDB **`describe-table`** 명령을 실행합니다.

```
aws dynamodb describe-table --table-name Weather
```

테이블이 성공적으로 삭제되면 `Requested resource not found`라는 구문이 포함된 메시지가 표시됩니다.

환경을 삭제하려면 환경 삭제를 참조하세요.

AWS Cloud9용 AWS CDK 자습서

이 자습서는 AWS Cloud9 개발 환경에서 AWS Cloud Development Kit (AWS CDK)를 사용하는 방법을 보여줍니다. AWS CDK는 개발자가 AWS 인프라 구성 요소를 코드로 모델링하는 데 사용할 수 있는 소프트웨어 도구 및 라이브러리로 구성된 세트입니다.

AWS CDK에는 AWS에서 여러 작업을 신속히 확인하는 데 사용할 수 있는 AWS Construct Library가 포함되어 있습니다. 예를 들면 `Fleet` 생성문을 사용하여 코드를 호스트 집합에 코드를 완전하고 안전하게 배포할 수 있습니다. 고유의 생성문을 작성하여 아키텍처의 다양한 요소를 모델링하거나 다른 사람과 공유하거나 커뮤니티에 게시할 수 있습니다. 자세한 내용은 AWS Cloud Development Kit 개발자 안내서를 참조하십시오.

이 자습서를 따르고 이 샘플을 생성하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2, Amazon SNS 및 Amazon SQS 같은 서비스에 대한 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금, Amazon SNS 요금 및 Amazon SQS 요금을 참조하세요.

주제

필수 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는 운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 내용은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.
- 기존 환경에 대한 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열 때 AWS Cloud9은 웹 브라우저에서 환경을 위한 IDE를 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기 섹션을 참조하세요.

1단계: 필수 도구 설치

이 단계에서는 환경에서 TypeScript 프로그래밍 언어로 작성된 샘플을 실행하는 데 필요한 모든 도구를 AWS CDK에 설치합니다.

1. 노드 버전 관리자 또는 **nvm** - 나중에 Node.js를 설치하는 데 사용합니다.
2. Node.js - 샘플에 필요하고 노드 패키지 관리자 또는 **npm** 을 포함하며, 나중에 TypeScript 및 AWS CDK를 설치하는 데 사용합니다.
3. TypeScript - 이 샘플에 필요합니다. AWS CDK는 여러 다른 프로그래밍 언어도 지원합니다.
4. AWS CDK.

1.1단계: 노드 버전 관리자(nvm) 설치

1. AWS Cloud9 IDE의 터미널 세션에서 최근 보안 업데이트와 버그 수정 사항이 설치되었는지 확인합니다. 이를 수행하려면 **yum update** (Amazon Linux) 또는 **apt update** 명령(Ubuntu Server)을 실행합니다. (터미널 세션을 새로 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다.)

   Amazon Linux의 경우:

   ```
   sudo yum -y update
   ```

   Ubuntu Server:

   ```
   sudo apt update
   ```

2. **nvm**이 이미 설치되었는지 확인합니다. 이렇게 하려면 **--version** 옵션을 사용하여 **nvm** 명령을 실행합니다.

```
nvm --version
```

명령이 성공적으로 실행되면 출력에 **nvm** 버전 번호가 포함되며, 1.2단계: Node.js 설치로 이동할 수 있습니다.

3. **nvm**을 다운로드하여 설치합니다. 이렇게 하려면 설치 스크립트를 실행하십시오. 이 예에서는 v0.33.0이 설치되어 있지만 최신 버전의 **nvm**을 여기에서 확인할 수 있습니다.

```
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh | bash
```

4. **nvm** 사용을 시작합니다. 터미널 세션을 닫고 나서 다시 시작하거나 명령을 포함하는 `~/.bashrc` 파일을 소싱하여 **nvm**을 로드합니다.

```
. ~/.bashrc
```

1.2단계: Node.js 설치

1. Node.js 설치 여부를 확인하고, 설치한 경우 설치된 버전이 16.17.0 이상인지 확인합니다. 이 샘플은 Node.js 16.17.0에서 테스트되었습니다. 확인하려면 IDE에 여전히 열려 있는 터미널 세션에서 **`--version`** 옵션을 사용하여 **node** 명령을 실행합니다.

```
node --version
```

Node.js를 설치했으면 출력에 버전 번호가 포함됩니다. 버전 번호가 v16.17.0이면 1.3단계: TypeScript 설치 섹션으로 이동합니다.

2. **nvm** 명령을 **`install`** 작업과 함께 실행하여 Node.js 16을 설치합니다.

> **Note**
>
> **`nvm install node`**을 실행하여 Node.js의 LTS(장기 지원) 버전을 설치할 수도 있습니다. AWS Cloud9는 Node.js의 LTS 버전 추적을 지원합니다.

```
nvm install v16
```

3. Node.js 16 사용을 시작합니다. 이렇게 하려면 다음과 같이 **alias** 작업, 별칭을 지정할 버전 번호 및 해당 별칭에 사용할 버전을 사용하여 **nvm** 명령을 실행합니다.

```
nvm alias default 16
```

> **Note**
>
> 이전 명령은 Node.js 16을 기본 Node.js 버전으로 설정합니다. 또는 **alias** 작업 대신 **use** 작업과 함께 **nvm** 명령을 실행할 수 있습니다(예: **nvm use 16.17.0**). 하지만 **use** 작업을 지정하면 현재 터미널 세션이 실행 중인 동안에만 Node.js 버전이 실행됩니다.

4. Node.js 16을 사용 중인지 확인하려면 **node --version** 명령을 다시 실행합니다. 올바른 버전이 설치된 경우 출력에 버전 v16이 포함됩니다.

1.3단계: TypeScript 설치

1. 이미 TypeScript를 설치했는지 여부를 확인합니다. 이렇게 하려면 IDE에 여전히 열려 있는 터미널 세션에서 **--version** 옵션을 사용하여 명령줄 TypeScript 컴파일러를 실행합니다.

```
tsc --version
```

TypeScript를 설치한 경우 출력에 TypeScript 버전 번호가 포함됩니다. TypeScript가 설치된 경우 1.4단계: AWS CDK 설치 단원으로 이동합니다.

2. TypeScript를 설치합니다. 이렇게 하려면 **install** 작업, **-g** 옵션, TypeScript 패키지의 이름을 사용하여 **npm** 명령을 실행합니다. 그러면 환경에 TypeScript가 글로벌 패키지로 설치됩니다.

```
npm install -g typescript
```

3. TypeScript가 설치되었는지 확인합니다. 이렇게 하려면 **--version** 옵션을 지정하여 명령줄 TypeScript 컴파일러를 실행합니다.

```
tsc --version
```

TypeScript를 설치한 경우 출력에 TypeScript 버전 번호가 포함됩니다.

1.4단계: AWS CDK 설치

1. 이미 AWS CDK를 설치했는지 여부를 확인합니다. 이렇게 하려면 IDE에 여전히 열려 있는 터미널 세션에서 **`--version`** 옵션을 사용하여 **`cdk`** 명령을 실행합니다.

   ```
   cdk --version
   ```

   AWS CDK를 설치한 경우 출력에 AWS CDK 버전 및 빌드 번호가 포함됩니다. 2단계: 코드 추가 단원으로 이동합니다.
2. 패키지를 환경에 전역적으로 설치하도록 `install` 작업, 설치할 AWS CDK 패키지의 이름, `-g` 옵션을 사용하여 **`npm`** 명령을 실행하여 AWS CDK를 설치합니다.

   ```
   npm install -g aws-cdk
   ```
3. AWS CDK가 설치되었고 올바르게 참조되는지 확인합니다. 이렇게 하려면 **`--version`** 옵션을 사용하여 **`cdk`** 명령을 실행합니다.

   ```
   cdk --version
   ```

   성공하면 AWS CDK 버전 및 빌드 번호가 표시됩니다.

2단계: 코드 추가

이 단계에서는 AWS CDK에서 AWS CloudFormation 스택을 프로그래밍 방식으로 배포하는 데 필요한 모든 소스 코드를 포함하는 샘플 TypeScript 프로젝트를 생성합니다. 이 스택은 AWS 계정의 Amazon SNS 주제 및 Amazon SQS 대기열을 생성한 다음 대기열에서 주제를 구독하도록 지정합니다.

1. IDE에 여전히 열려 있는 터미널 세션을 사용하여 환경에서 프로젝트의 소스 코드를 저장할 디렉터리(예: `~/environment/hello-cdk` 디렉터리)를 생성합니다. 그런 다음 해당 디렉터리로 전환합니다.

   ```
   rm -rf ~/environment/hello-cdk # Remove this directory if it already exists.
   mkdir ~/environment/hello-cdk  # Create the directory.
   cd ~/environment/hello-cdk     # Switch to the directory.
   ```
2. 해당 디렉터리를 AWS CDK에 대한 TypeScript 언어 프로젝트로 설정합니다. 이렇게 하려면 **`init`** 작업, **`sample-app`** 템플릿, **`--language`** 옵션을 프로그래밍 언어의 이름과 함께 사용하여 **`cdk`** 명령을 실행합니다.

```
cdk init sample-app --language typescript
```

그러면 다음 파일 및 하위 디렉터리가 해당 디렉터리에 생성됩니다.

- 숨은 `.git` 하위 디렉터리와 숨은 `.gitignore` 파일 - Git와 같은 소스 제어 도구와 호환되는 프로젝트를 만듭니다.
- `lib` 하위 디렉터리 - `hello-cdk-stack.ts` 파일을 포함합니다. 이 파일에는 AWS CDK 스택의 코드가 들어 있습니다. 이 코드는 이 절차의 다음 단계에 설명되어 있습니다.
- `bin` 하위 디렉터리 - `hello-cdk.ts` 파일을 포함합니다. 이 파일에는 AWS CDK 앱의 진입점이 포함되어 있습니다.
- `node_modules` 하위 디렉터리 - 필요 시 앱과 스택에서 사용할 수 있는 지원 코드 패키지가 들어 있습니다.
- 숨은 `.npmignore` 파일 - 코드 빌드 시 **npm**에 필요하지 않은 하위 디렉터리 및 파일 유형을 나열합니다.
- `cdk.json` 파일 - **cdk** 명령을 보다 쉽게 실행할 수 있도록 하는 정보가 들어 있습니다.
- `package-lock.json` 파일 - **npm**에서 발생 가능한 빌드 및 실행 오류를 줄이는 데 사용할 수 있는 정보가 들어 있습니다.
- `package.json` 파일 - **npm** 명령을 좀 더 쉽게 실행하고 빌드 및 실행 오류가 거의 발생하지 않도록 하는 정보가 들어 있습니다.
- `README.md` 파일 - **npm** 및 AWS CDK와 함께 실행할 수 있는 유용한 명령이 나열되어 있습니다.
- `tsconfig.json` 파일 - **tsc** 명령을 좀 더 쉽게 실행하고 빌드 및 실행 오류가 거의 발생하지 않도록 하는 정보가 들어 있습니다.

3. Environment(환경) 창에서 `lib/hello-cdk-stack.ts` 파일을 열고 해당 파일에서 다음 코드를 찾아봅니다.

```
import sns = require('@aws-cdk/aws-sns');
import sqs = require('@aws-cdk/aws-sqs');
import cdk = require('@aws-cdk/cdk');

export class HelloCdkStack extends cdk.Stack {
  constructor(parent: cdk.App, name: string, props?: cdk.StackProps) {
    super(parent, name, props);

    const queue = new sqs.Queue(this, 'HelloCdkQueue', {
```

```
      visibilityTimeoutSec: 300
    });

    const topic = new sns.Topic(this, 'HelloCdkTopic');

    topic.subscribeQueue(queue);
  }
}
```

- `Stack`, `App`, `StackProps`, `Queue` 및 `Topic` 클래스는 각각 AWS CloudFormation 스택 및 속성, 실행 프로그램, Amazon SQS 대기열 및 Amazon SNS 주제를 나타냅니다.
- `HelloCdkStack` 클래스는 이 애플리케이션에 대한 AWS CloudFormation 스택을 나타냅니다. 이 스택에는 이 애플리케이션에 대한 새 Amazon SQS 대기열 및 Amazon SNS 주제가 들어 있습니다.

4. Environment(환경) 창에서 `bin/hello-cdk.ts` 파일을 열고 해당 파일에서 다음 코드를 찾아봅니다.

```
#!/usr/bin/env node
import cdk = require('@aws-cdk/cdk');
import { HelloCdkStack } from '../lib/hello-cdk-stack';

const app = new cdk.App();
new HelloCdkStack(app, 'HelloCdkStack');
app.run();
```

이 코드는 `lib/hello-cdk-stack.ts` 파일에서 `HelloCdkStack` 클래스를 로드하고 인스턴스화한 후 실행합니다.

5. **npm**을 통해 TypeScript 컴파일러를 실행하여 코딩 오류 여부를 확인하고 나서 AWS CDK에서 프로젝트의 `bin/hello-cdk.js` 파일을 실행할 수 있도록 설정합니다. 이렇게 하려면 프로젝트의 루트 디렉터리에서 다음과 같이 `package.json` 파일에 **`build`** 명령 값을 지정하고 **`run`** 작업을 사용하여 **`npm`** 명령을 실행합니다.

```
npm run build
```

이전 명령을 실행하면 TypeScript 컴파일러가 실행되어 지원되는 `bin/hello-cdk.d.ts` 및 `lib/hello-cdk-stack.d.ts` 파일이 추가됩니다. 이 컴파일러는 `hello-cdk.ts` 및 `hello-cdk-stack.ts` 파일을 `hello-cdk.js` 및 `hello-cdk-stack.js` 파일로 트랜스컴파일합니다.

3단계: 코드 실행

이 단계에서는 AWS CDK에 `bin/hello-cdk.js` 파일의 코드에 따라 AWS CloudFormation 스택 템플릿을 생성하도록 지시합니다. 그런 다음 AWS CDK에 스택을 배포하도록 지시합니다. 그러면 Amazon SNS 주제 및 Amazon SQS 대기열이 생성되고 나서 대기열이 주제를 구독하도록 지정됩니다. 그런 다음 주제의 메시지를 대기열로 전송하여 주제 및 대기열이 성공적으로 배포되었는지 확인할 수 있습니다.

1. AWS CDK에서 AWS CloudFormation 스택 템플릿을 생성하도록 합니다. 이렇게 하려면 터미널 세션이 아직 열려 있는 상태로, **`synth`** 작업 및 스택의 이름을 사용하여 프로젝트의 루트 디렉터리에서 **`cdk`** 명령을 실행합니다.

   ```
   cdk synth HelloCdkStack
   ```

   성공하면 출력에 AWS CloudFormation 스택 템플릿의 `Resources` 섹션이 표시됩니다.

2. AWS CDK 앱을 특정 AWS 계정 및 AWS 리전 조합을 위한 환경에 처음 배포하는 경우 부트스트랩 스택을 설치해야 합니다. 이 스택에는 AWS CDK에서 다양한 작업을 완료하는 데 필요한 여러 가지 리소스가 포함되어 있습니다. 예를 들면 이 스택에는 배포 프로세스 동안 AWS CDK가 템플릿 및 자산을 저장하는 데 사용하는 Amazon S3 버킷이 포함되어 있습니다. 부트스트랩 스택을 설치하려면 **`bootstrap`** 작업을 사용하여 **`cdk`** 명령을 실행합니다.

   ```
   cdk bootstrap
   ```

   > **Note**
   >
   > 옵션을 지정하지 않고 `cdk bootstrap`을 실행할 경우 기본 AWS 계정 및 AWS 리전이 사용됩니다. 프로필과 계정/리전 조합을 지정하여 특정 환경을 부트스트랩할 수도 있습니다. 예:
   >
   > ```
   > cdk bootstrap --profile test 123456789012/us-east-1
   > ```

3. AWS CDK에서 AWS CloudFormation 스택 템플릿을 실행하여 스택을 배포하도록 합니다. 이렇게 하려면 **`deploy`** 작업 및 스택의 이름을 사용하여 프로젝트의 루트 디렉터리에서 **`cdk`** 명령을 실행합니다.

   ```
   cdk deploy HelloCdkStack
   ```

성공하면 출력에 `HelloCdkStack` 스택이 오류 없이 배포되었다는 메시지가 표시됩니다.

> **Note**
>
> 스택에 환경이 정의되지 않았고 AWS 자격 증명을 표준 위치에서 가져올 수 없거나 리전이 구성되지 않았다는 메시지가 출력에 표시되면 IDE에 AWS 자격 증명이 올바르게 설정되었는지 확인하고 나서 **`cdk deploy`** 명령을 다시 실행합니다. 자세한 내용은 AWS Cloud9의 환경에서 AWS 서비스 호출 섹션을 참조하세요.

4. Amazon SNS 주제 및 Amazon SQS 대기열이 성공적으로 배포되었는지 확인하려면 메시지를 주제로 전송하고 나서 대기열에 수신된 메시지가 있는지 확인합니다. 이렇게 하려면 AWS Command Line Interface(AWS CLI) 또는 AWS CloudShell과 같은 도구를 사용합니다. 이러한 도구에 대한 자세한 내용은 AWS Cloud9용 AWS Command Line Interface 및 aws-shell 자습서 단원을 참조하십시오.

   예를 들어 주제로 메시지를 전송하려면 IDE에 터미널 세션이 아직 열려 있는 상태로 AWS CLI를 사용하여 Amazon SNS **`publish`** 명령을 실행합니다. 이때 메시지의 제목과 본문, 주제의 AWS 리전 및 주제의 Amazon 리소스 이름(ARN)을 지정합니다.

   ```
   aws sns publish --subject "Hello from the AWS CDK" --message "This is a message
    from the AWS CDK." --topic-arn arn:aws:sns:us-east-2:123456789012:HelloCdkStack-
   HelloCdkTopic1A234567-8BCD9EFGHIJ0K
   ```

   이전 명령에서 `arn:aws:sns:us-east-2:123456789012:HelloCdkStack-HelloCdkTopic1A234567-8BCD9EFGHIJ0K`을 AWS CloudFormation에서 주제에 할당하는 ARN으로 바꿉니다. ID를 가져오려면 Amazon SNS **`list-topics`** 명령을 실행합니다.

   ```
   aws sns list-topics --output table --query 'Topics[*].TopicArn'
   ```

   성공하면 **`publish`** 명령의 출력에 게시된 메시지에 대한 `MessageId` 값이 표시됩니다.

   대기열에서 수신된 메시지가 있는지 확인하려면 대기열의 URL을 지정하여 Amazon SQS **`receive-message`** 명령을 실행합니다.

   ```
   aws sqs receive-message --queue-url https://queue.amazonaws.com/123456789012/
   HelloCdkStack-HelloCdkQueue1A234567-8BCD9EFGHIJ0K
   ```

이전 명령에서 `https://queue.amazonaws.com/123456789012/HelloCdkStack-HelloCdkQueue1A234567-8BCD9EFGHIJ0K`을 AWS CloudFormation에서 대기열에 할당하는 ARN으로 바꿉니다. URL을 가져오려면 Amazon SQS **`list-queues`** 명령을 실행합니다.

```
aws sqs list-queues --output table --query 'QueueUrls[*]'
```

성공하면 **`receive-message`** 명령의 출력에 수신된 메시지에 대한 정보가 표시됩니다.

4단계: 정리

이 샘플 사용 후 AWS 계정에 계속하여 요금이 부과되지 않도록 AWS CloudFormation 스택을 삭제해야 합니다. 이렇게 하면 Amazon SNS 주제 및 Amazon SQS 대기열이 삭제됩니다. 또한 환경도 삭제해야 합니다.

4.1단계: 스택 삭제

터미널 세션이 아직 열려 있는 상태로, **`destroy`** 작업 및 스택의 이름을 사용하여 프로젝트의 루트 디렉터리에서 **`cdk`** 명령을 실행합니다.

```
cdk destroy HelloCdkStack
```

스택을 삭제할지 묻는 메시지가 표시되면 `y`를 입력하고 나서 `Enter`를 누릅니다.

성공하면 출력에 `HelloCdkStack` 스택이 오류 없이 삭제되었다는 메시지가 표시됩니다.

4.2단계: 환경 삭제

환경을 삭제하려면 AWS Cloud9에서 환경 삭제 섹션을 참조하세요.

AWS Cloud9용 LAMP 자습서

이 자습서를 통해 AWS Cloud9 개발 환경 내에서 LAMP(Linux, Apache HTTP 서버, MySQL 및 PHP)를 설정하고 실행할 수 있습니다.

이 자습서를 따르고 이 샘플을 생성하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon Elastic Compute Cloud(Amazon EC2)와 같은 AWS 서비스에 대해 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금을 참조하세요.

주제

필수 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는 운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 내용은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.
- 기존 환경에 대한 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열 때 AWS Cloud9은 웹 브라우저에서 환경을 위한 IDE를 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기 섹션을 참조하세요.

1단계: 도구 설치

이 단계에서는 다음 도구를 설치합니다.

- Apache HTTP 서버 - 웹 서버 호스트.
- PHP - 특히 웹 개발에 적합하고 HTML에 포함할 수 있는 스크립팅 언어.
- MySQL - 데이터베이스 관리 시스템.

그런 다음 Apache HTTP Server와 MySQL을 차례로 시작하여 이 단계를 완료합니다.

1. 최신 보안 업데이트와 버그 수정이 인스턴스에 설치되어 있는지 확인합니다. 이를 수행하려면 AWS Cloud9 IDE의 터미널 세션에서 **`yum update`**(Amazon Linux) 또는 **`apt update`**(Ubuntu Server) 명령을 실행합니다. (터미널 세션을 새로 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다.)

   Amazon Linux의 경우:

   ```
   sudo yum -y update
   ```

Ubuntu 서버용:

```
sudo apt -y update
```

2. Apache HTTP Server가 이미 설치되어 있는지 확인합니다. 이를 수행하려면 **httpd -v**(Amazon Linux) 또는 **apache2 -v**(Ubuntu Server) 명령을 실행합니다.

   성공할 경우, 출력에 Apache HTTP Server 버전 번호가 포함됩니다.

   오류가 표시되면 **install** 명령을 실행하여 Apache HTTP Server를 설치합니다.

   Amazon Linux의 경우:

   ```
   sudo yum install -y httpd24
   ```

   Ubuntu 서버용:

   ```
   sudo apt install -y apache2
   ```

3. **php -v** 명령을 실행하여 PHP가 이미 설치되어 있는지 확인합니다.

   성공할 경우, PHP 버전 숫자를 포함해 출력값이 생성됩니다.

   오류가 표시되면 **install** 명령을 실행하여 PHP를 설치합니다.

   Amazon Linux의 경우:

   ```
   sudo yum install -y php56
   ```

   Ubuntu 서버용:

   ```
   sudo apt install -y php libapache2-mod-php php-xml
   ```

4. **mysql --version** 명령을 실행하여 MySQL가 이미 설치되어 있는지 확인합니다.

   성공할 경우, 출력에 MySQL 버전 번호가 포함됩니다.

   오류가 표시되면 **install** 명령을 실행하여 MySQL을 설치합니다.

   Amazon Linux의 경우:

```
sudo yum install -y mysql-server
```

Ubuntu 서버용:

```
sudo apt install -y mysql-server
```

5. Apache HTTP Server, PHP, MySQL을 설치한 후 Apache HTTP Server를 시작하고 다음 명령을 실행하여 시작되었는지 확인합니다.

   Amazon Linux의 경우(이 명령을 두 번 실행해야 할 수 있음):

   ```
   sudo service httpd start && sudo service httpd status
   ```

   Ubuntu Server의 경우(명령 프롬프트로 돌아가려면 q를 누릅니다):

   ```
   sudo service apache2 start && sudo service apache2 status
   ```

6. MySQL을 시작한 후 다음 명령을 실행하여 시작되었는지 확인합니다.

   Amazon Linux의 경우:

   ```
   sudo service mysqld start && sudo service mysqld status
   ```

   Ubuntu Server의 경우(명령 프롬프트로 돌아가려면 q를 누릅니다):

   ```
   sudo service mysql start && sudo service mysql status
   ```

2단계: MySQL 설정

이 단계에서는 MySQL 보안 모범 사례를 따르도록 MySQL을 설정합니다. 이러한 보안 모범 사례에는 루트 계정의 암호 설정과 로컬 호스트 외부에서 액세스할 수 있는 루트 계정 제거가 포함됩니다. 염두에 두어야 할 다른 모범 사례로는 익명 사용자 제거, 테스트 데이터베이스 제거, 누구든지 이름이 `test_`로 시작하는 데이터베이스에 액세스할 수 있는 권한 제거 등이 있습니다.

그런 다음 MySQL 명령줄 클라이언트의 시작 및 종료를 연습하여 이 단계를 완료합니다.

1. AWS Cloud9 IDE의 터미널 세션에서 다음 명령을 실행하여 MySQL 설치에 대한 MySQL 보안 모범 사례를 구현합니다.

```
sudo mysql_secure_installation
```

2. 메시지가 표시되면 지정된 대로 다음 질문에 답합니다.

   Amazon Linux의 경우:

   1. 루트의 현재 암호 입력(입력하지 않으려면 Enter) - `Enter` 키를 누릅니다(암호가 없는 경우).
   2. 루트 암호 설정 - `Y`를 입력한 후 `Enter` 키를 누릅니다.
   3. 새 암호 - 암호를 입력한 후 `Enter` 키를 누릅니다.
   4. 새 암호 다시 입력 - 암호를 다시 입력한 후 `Enter` 키를 누릅니다. (나중에 사용할 수 있도록 암호를 안전한 위치에 저장해야 합니다.)
   5. 익명 사용자 제거 - `Y`를 입력한 후 `Enter` 키를 누릅니다.
   6. 원격으로 루트 로그인 허용 안 함 - `Y`를 입력한 후 `Enter` 키를 누릅니다.
   7. 테스트 데이터베이스를 제거하고 액세스 - `Y`를 입력한 후 `Enter` 키를 누릅니다.
   8. 지금 권한 테이블 다시 로드 - `Y`를 입력한 후 `Enter` 키를 누릅니다.

   Ubuntu 서버용:

   1. VALIDATE PASSWORD 플러그인을 설정하시겠습니까 - `y`를 입력한 후 `Enter` 키를 누릅니다.
   2. 세 가지 수준의 암호 확인 정책이 있습니다 - `0`, `1` 또는 `2`를 입력한 후 `Enter` 키를 누릅니다.
   3. 새 암호 - 암호를 입력한 후 `Enter` 키를 누릅니다.
   4. 새 암호 다시 입력 - 암호를 다시 입력한 후 `Enter` 키를 누릅니다. 나중에 사용할 수 있도록 암호를 안전한 위치에 저장해야 합니다.
   5. 제공된 암호를 계속 사용하시겠습니까 - `y`를 입력한 후 `Enter` 키를 누릅니다.
   6. 익명 사용자 제거 - `y`를 입력한 후 `Enter` 키를 누릅니다.
   7. 원격으로 루트 로그인 허용 안 함 - `y`를 입력한 후 `Enter` 키를 누릅니다.
   8. 테스트 데이터베이스를 제거하고 액세스 - `y`를 입력한 후 `Enter` 키를 누릅니다.
   9. 지금 권한 테이블 다시 로드 - `y`를 입력한 후 `Enter` 키를 누릅니다.

3. MySQL과 직접 상호 작용하려면 다음 명령을 실행하여 MySQL 명령줄 클라이언트를 루트 사용자로 시작합니다. 메시지가 표시되면 앞서 설정한 루트 사용자의 암호를 입력한 후 `Enter`를 누릅니다. MySQL 명령줄 클라이언트에 있는 동안 프롬프트가 `mysql>`로 바뀝니다.

```
sudo mysql -uroot -p
```

4. MySQL 명령줄 클라이언트를 종료하려면 다음 명령을 실행합니다. 프롬프트가 $로 다시 바뀝니다.

```
exit;
```

3단계: 웹 사이트 설정

이 단계에서는 권장되는 소유자 및 액세스 권한으로 Apache HTTP Server의 기본 웹 사이트 루트를 설정합니다. 그런 다음, 해당 기본 웹 사이트 루트 내에서 PHP 기반 웹 페이지를 만듭니다.

그런 다음 이 EC2 환경과 연결된 Amazon EC2의 보안 그룹과 Amazon Virtual Private Cloud(Amazon VPC)의 네트워크 액세스 제어 목록(네트워크 ACL)을 설정하여 해당 웹 페이지를 보도록 수신 웹 트래픽을 활성화합니다. 각 EC2 환경은 Amazon EC2의 보안 그룹 및 Amazon VPC의 네트워크 ACL과 연결되어 있어야 합니다. 그러나 AWS 계정의 기본 네트워크 ACL이 환경의 모든 수신 및 발신 트래픽을 허용하는 반면, 기본 보안 그룹은 포트 22를 통해 SSH를 사용하는 수신 트래픽만 허용합니다. 자세한 내용은 the section called “Amazon VPC 설정” 섹션을 참조하세요.

그런 다음 AWS Cloud9 IDE 외부에서 웹 페이지를 봄으로써 이 단계를 완료합니다.

1. 권장되는 소유자 및 액세스 권한으로 Apache HTTP Server의 기본 웹 사이트 루트(`/var/www/html`)를 설정합니다. 이를 수행하려면 AWS Cloud9 IDE의 터미널 세션에서 다음과 같은 여섯 가지 명령을 다음 순서로 한 번에 하나씩 실행합니다. 각 명령이 수행하는 작업을 이해하려면 각 명령 뒤의 # 문자 뒤의 정보를 읽습니다.

   Amazon Linux의 경우:

```
sudo groupadd web-content # Create a group named web-content.

sudo usermod -G web-content -a ec2-user # Add the user ec2-user (your default user
 for this environment) to the group web-content.

sudo usermod -G web-content -a apache # Add the user apache (Apache HTTP Server) to
 the group web-content.

sudo chown -R ec2-user:web-content /var/www/html # Change the owner of /var/www/
html and its files to user ec2-user and group web-content.
```

```
sudo find /var/www/html -type f -exec chmod u=rw,g=rx,o=rx {} \; # Change all file
 permissions within /var/www/html to user read/write, group read-only, and others
 read/execute.

sudo find /var/www/html -type d -exec chmod u=rwx,g=rx,o=rx {} \; # Change /var/
www/html directory permissions to user read/write/execute, group read/execute, and
 others read/execute.
```

Ubuntu 서버용:

```
sudo groupadd web-content # Create a group named web-content.

sudo usermod -G web-content -a ubuntu # Add the user ubuntu (your default user for
 this environment) to the group web-content.

sudo usermod -G web-content -a www-data # Add the user www-data (Apache HTTP
 Server) to the group web-content.

sudo chown -R ubuntu:web-content /var/www/html # Change the owner of /var/www/html
 and its files to user ubuntu and group web-content.

sudo find /var/www/html -type f -exec chmod u=rw,g=rx,o=rx {} \; # Change all file
 permissions within /var/www/html to user read/write, group read-only, and others
 read/execute.

sudo find /var/www/html -type d -exec chmod u=rwx,g=rx,o=rx {} \; # Change /var/
www/html directory permissions to user read/write/execute, group read/execute, and
 others read/execute.
```

2. 다음 명령을 실행하여 Apache HTTP Server의 기본 웹 사이트 루트 폴더(`/var/www/html`)에 `index.php`라는 PHP 기반 웹 페이지를 만듭니다.

   Amazon Linux의 경우:

   ```
   sudo touch /var/www/html/index.php && sudo chown -R ec2-user:web-content /var/www/
   html/index.php && sudo chmod u=rw,g=rx,o=rx /var/www/html/index.php && sudo printf
    '%s\n%s\n%s' '<?php' '  phpinfo();' '?>' >> /var/www/html/index.php
   ```

   앞의 Amazon Linux용 명령은 파일의 소유자를 `ec2-user`로 변경하고, 파일의 그룹을 `web-content`로 변경하며, 파일의 권한을 사용자의 경우 읽기/쓰기, 그룹 및 기타의 경우 읽기/실행으로 변경합니다.

Ubuntu 서버용:

```
sudo touch /var/www/html/index.php && sudo chown -R ubuntu:web-content /var/www/
html/index.php && sudo chmod u=rw,g=rx,o=rx /var/www/html/index.php && sudo printf
 '%s\n%s\n%s' '<?php' '  phpinfo();' '?>' >> /var/www/html/index.php
```

앞의 Ubuntu 서버용 명령은 파일의 소유자를 `ubuntu`로 변경하고, 파일의 그룹을 `web-content`로 변경하며, 파일의 권한을 사용자의 경우 읽기/쓰기, 그룹 및 기타의 경우 읽기/실행으로 변경합니다.

성공하면 앞의 명령이 다음 내용이 포함된 `index.php` 파일을 만듭니다.

```
<?php
  phpinfo();
?>
```

3. Amazon VPC의 네트워크 ACL 및 이 EC2 환경과 연결된 Amazon EC2의 보안 그룹을 설정하여 새 웹 페이지를 보도록 포트 80을 통한 수신 웹 트래픽을 활성화합니다. 이를 수행하려면 다음과 같은 여덟 가지 명령을 다음 순서로 한 번에 하나씩 실행합니다. 각 명령이 수행하는 작업을 이해하려면 각 명령의 # 문자 뒤의 정보를 읽습니다.

> **Important**
>
> 다음 명령을 실행하면 이 환경의 보안 그룹 및 네트워크 ACL과 연결된 모든 EC2 환경 및 Amazon EC2 인스턴스에 대해 포트 80을 통해 들어오는 웹 트래픽이 허용됩니다. 그러면 이것 외에도 EC2 환경 및 Amazon EC2 인스턴스에 대해 포트 80을 통한 수신 웹 트래픽이 예기치 않게 허용될 수 있습니다.

> **Note**
>
> 다음 두 번째부터 네 번째 명령은 보안 그룹이 포트 80을 통한 수신 웹 트래픽을 허용할 수 있게 합니다. 포트 22를 통한 수신 SSH 트래픽만 허용하는 기본 보안 그룹이 있는 경우, 첫 번째 명령을 실행한 후 이러한 두 번째부터 네 번째 명령을 실행해야 합니다. 그러나 사용자 지정 보안 그룹이 이미 포트 80을 통한 수신 웹 트래픽을 허용하는 경우, 그러한 명령 실행을 건너뛸 수 있습니다.
>
> 다음 다섯 번째부터 여덟 번째 명령은 네트워크 ACL이 포트 80을 통한 수신 웹 트래픽을 허용할 수 있게 합니다. 이미 모든 포트를 통한 모든 수신 트래픽을 허용하는 기본 네트워

크 ACL이 있는 경우 그러한 명령 실행을 안전하게 건너뛸 수 있습니다. 하지만 포트 80을 통한 수신 웹 트래픽을 허용하지 않는 사용자 지정 네트워크 ACL이 있다고 가정해 보겠습니다. 그런 다음, 첫 번째 명령을 실행한 후 다섯 번째 명령부터 여덟 번째 명령까지 실행합니다.

```
MY_INSTANCE_ID=$(curl http://169.254.169.254/latest/meta-data/instance-id) # Get
 the ID of the instance for the environment, and store it temporarily.

MY_SECURITY_GROUP_ID=$(aws ec2 describe-instances --instance-id $MY_INSTANCE_ID
 --query 'Reservations[].Instances[0].SecurityGroups[0].GroupId' --output text)
 # Get the ID of the security group associated with the instance, and store it
 temporarily.

aws ec2 authorize-security-group-ingress --group-id $MY_SECURITY_GROUP_ID --
protocol tcp --cidr 0.0.0.0/0 --port 80 # Add an inbound rule to the security group
 to allow all incoming IPv4-based traffic over port 80.

aws ec2 authorize-security-group-ingress --group-id $MY_SECURITY_GROUP_ID --ip-
permissions IpProtocol=tcp,Ipv6Ranges='[{CidrIpv6=::/0}]',FromPort=80,ToPort=80 #
 Add an inbound rule to the security group to allow all incoming IPv6-based traffic
 over port 80.

MY_SUBNET_ID=$(aws ec2 describe-instances --instance-id $MY_INSTANCE_ID --query
 'Reservations[].Instances[0].SubnetId' --output text) # Get the ID of the subnet
 associated with the instance, and store it temporarily.

MY_NETWORK_ACL_ID=$(aws ec2 describe-network-acls --filters
 Name=association.subnet-id,Values=$MY_SUBNET_ID --query
 'NetworkAcls[].Associations[0].NetworkAclId' --output text) # Get the ID of the
 network ACL associated with the subnet, and store it temporarily.

aws ec2 create-network-acl-entry --network-acl-id $MY_NETWORK_ACL_ID --ingress --
protocol tcp --rule-action allow --rule-number 10000 --cidr-block 0.0.0.0/0 --port-
range From=80,To=80 # Add an inbound rule to the network ACL to allow all IPv4-
based traffic over port 80. Advanced users: change this suggested rule number as
 desired.

aws ec2 create-network-acl-entry --network-acl-id $MY_NETWORK_ACL_ID --ingress --
protocol tcp --rule-action allow --rule-number 10100 --ipv6-cidr-block ::/0 --port-
range From=80,To=80 # Add an inbound rule to the network ACL to allow all IPv6-
```

```
based traffic over port 80. Advanced users: change this suggested rule number as
 desired.
```

4. 웹 서버 루트 내에 `index.php` 파일의 URL을 가져옵니다. 이를 수행하려면 다음 명령을 실행하고, 새 웹 브라우저 탭 또는 AWS Cloud9 IDE와 다른 웹 브라우저를 사용하여 표시되는 URL로 이동합니다. 성공하면 웹 페이지에 Apache HTTP Server, MySQL, PHP 및 기타 관련 설정에 관한 정보가 표시됩니다.

```
MY_PUBLIC_IP=$(curl http://169.254.169.254/latest/meta-data/public-ipv4) && echo
 http://$MY_PUBLIC_IP/index.php # Get the URL to the index.php file within the web
 server root.
```

4단계: 정리

이 환경을 계속 사용하고 싶지만 포트 80을 통해 들어오는 웹 트래픽을 비활성화하려는 상황을 가정해 보겠습니다. 그렇다면 다음 여덟 가지 명령을 다음 순서로 한 번에 하나씩 실행하여 환경과 연결된 보안 그룹 및 네트워크 ACL에서 앞서 설정한 해당 수신 트래픽 규칙을 삭제합니다. 각 명령이 수행하는 작업을 이해하려면 각 명령의 # 문자 뒤의 정보를 읽습니다.

Important

다음 명령을 실행하면 이 환경의 보안 그룹 및 네트워크 ACL과 연결된 모든 EC2 환경 및 Amazon EC2 인스턴스에 대해 포트 80을 통해 들어오는 웹 트래픽이 허용되지 않습니다. 그러면 이것 외에도 EC2 환경 및 Amazon EC2 인스턴스에 대해 포트 80을 통한 수신 웹 트래픽이 예기치 않게 허용되지 않을 수 있습니다.

Note

다음 다섯 번째 명령부터 여덟 번째 명령은 네트워크 ACL이 포트 80을 통한 수신 웹 트래픽을 허용하지 못하도록 기존 규칙을 제거합니다. 이미 모든 포트를 통한 모든 수신 트래픽을 허용하는 기본 네트워크 ACL이 있는 경우 그러한 명령 실행을 건너뛸 수 있습니다. 하지만 포트 80을 통한 수신 웹 트래픽을 허용하는 기존 규칙이 포함된 사용자 지정 네트워크 ACL이 있고 해당 규칙을 삭제하려는 경우를 가정해 보겠습니다. 그런 다음, 첫 번째 명령을 실행한 후 다섯 번째 명령부터 여덟 번째 명령까지 실행해야 합니다.

```
MY_INSTANCE_ID=$(curl http://169.254.169.254/latest/meta-data/instance-id) # Get the ID
 of the instance for the environment, and store it temporarily.

MY_SECURITY_GROUP_ID=$(aws ec2 describe-instances --instance-id $MY_INSTANCE_ID --query
 'Reservations[].Instances[0].SecurityGroups[0].GroupId' --output text) # Get the ID of
 the security group associated with the instance, and store it temporarily.

aws ec2 revoke-security-group-ingress --group-id $MY_SECURITY_GROUP_ID --protocol tcp
 --cidr 0.0.0.0/0 --port 80 # Delete the existing inbound rule from the security group
 to block all incoming IPv4-based traffic over port 80.

aws ec2 revoke-security-group-ingress --group-id $MY_SECURITY_GROUP_ID --ip-permissions
 IpProtocol=tcp,Ipv6Ranges='[{CidrIpv6=::/0}]',FromPort=80,ToPort=80 # Delete the
 existing inbound rule from the security group to block all incoming IPv6-based traffic
 over port 80.

MY_SUBNET_ID=$(aws ec2 describe-instances --instance-id $MY_INSTANCE_ID --query
 'Reservations[].Instances[0].SubnetId' --output text) # Get the ID of the subnet
 associated with the instance, and store it temporarily.

MY_NETWORK_ACL_ID=$(aws ec2 describe-network-acls --filters Name=association.subnet-
id,Values=$MY_SUBNET_ID --query 'NetworkAcls[].Associations[0].NetworkAclId' --output
 text) # Get the ID of the network ACL associated with the subnet, and store it
 temporarily.

aws ec2 delete-network-acl-entry --network-acl-id $MY_NETWORK_ACL_ID --ingress --rule-
number 10000 # Delete the existing inbound rule from the network ACL to block all IPv4-
based traffic over port 80. Advanced users: if you originally created this rule with a
 different number, change this suggested rule number to match.

aws ec2 delete-network-acl-entry --network-acl-id $MY_NETWORK_ACL_ID --ingress --rule-
number 10100 # Delete the existing inbound rule from the network ACL to block all IPv6-
based traffic over port 80. Advanced users: if you originally created this rule with a
 different number, change this suggested rule number to match.
```

이 환경의 사용을 완료하면 AWS 계정에 계속 요금이 부과되지 않도록 환경을 삭제하세요. 지침은 AWS Cloud9에서 환경 삭제 단원을 참조하세요.

AWS Cloud9용 WordPress 자습서

이 자습서를 사용하면 AWS Cloud9 개발 환경 내에서 WordPress를 설치하고 실행할 수 있습니다. WordPress는 전송 웹 콘텐츠에 널리 사용되는 오픈 소스 콘텐츠 관리 시스템(CMS)입니다.

> **Note**
>
> 이 자습서를 따르고 이 샘플을 생성하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon Elastic Compute Cloud(Amazon EC2)와 같은 서비스에 대해 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금을 참조하세요.

필수 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는 운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 내용은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.
- 기존 환경에 대한 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열 때 AWS Cloud9은 웹 브라우저에서 환경을 위한 IDE를 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기 섹션을 참조하세요.
- 최신 소프트웨어 패키지가 모두 포함된 최신 EC2 인스턴스가 있습니다. AWS Cloud9 IDE 터미널 창에서 `-y` 옵션으로 `yum update`를 실행하여 확인 의사를 묻지 않고 업데이트를 설치할 수 있습니다. 설치 전에 업데이트 정보를 확인하려면 이 옵션을 생략합니다.

```
sudo yum update -y
```

설치 개요

환경의 EC2 인스턴스에 WordPress를 설치하는 단계는 다음과 같습니다.

1. WordPress 설치에 대한 정보를 저장하는 오픈 소스 관계형 데이터베이스인 MariaDB 서버 설치 및 구성
2. `wordpress.conf` 구성 파일이 포함된 WordPress 설치 및 구성
3. WordPress 사이트를 호스트하는 Apache 서버 구성
4. Apache 서버에서 호스트하는 WordPress 웹 콘텐츠 미리 보기

1단계: MariaDB 서버 설치 및 구성

1. AWS Cloud9 IDE에서 [창(Window)], [새 터미널(New Terminal)]을 선택하고 다음 명령을 입력하여 MariaDB 서버 설치를 시작합니다.

   ```
   sudo yum install -y mariadb-server
   sudo systemctl start mariadb
   ```

2. 다음으로, `mysql_secure_installation` 스크립트를 실행하여 MariaDB 서버 설치의 보안을 강화합니다.

   이 스크립트에 대한 응답을 제공할 때 첫 번째 질문에 대해 Enter 키를 입력하여 루트 암호를 비워 둡니다. `Set root password?`에 대해 n 키를 누른 다음 나머지 각 보안 옵션에 대해 y 키를 누릅니다.

   ```
   mysql_secure_installation
   ```

3. 이제 MariaDB 클라이언트를 사용하여 WordPress 정보를 저장할 데이터베이스 테이블을 만듭니다.

   (암호를 묻는 메시지가 나타나면 Enter 키를 누릅니다.)

   ```
   sudo mysql -u root -p
   MariaDB [(none)]> create database wp_test;
   MariaDB [(none)]> grant all privileges on wp_test.* to root@localhost identified by
    ';'
   ```

4. MariaDB 클라이언트에서 로그아웃하려면 `exit` 명령을 실행합니다.

2단계: WordPress 설치 및 구성

1. IDE 터미널 창에서 `environment` 디렉터리로 이동한 다음 `config` 및 `wordpress` 디렉터리를 만듭니다. 그런 다음 `touch` 명령을 실행하여 `config` 디렉터리에 `wordpress.conf`라는 파일을 생성합니다.

   ```
   cd /home/ec2-user/environment
   mkdir config wordpress
   touch config/wordpress.conf
   ```

2. IDE 편집기 또는 vim을 사용하여, Apache 서버가 WordPress 콘텐츠를 제공할 수 있도록 허용하는 호스트 구성 정보로 `wordpress.conf`를 업데이트합니다.

```
# Ensure that Apache listens on port 80
Listen 8080
<VirtualHost *:8080>
    DocumentRoot "/var/www/wordpress"
    ServerName www.example.org
    # Other directives here
</VirtualHost>
```

3. 이제 다음 명령을 실행하여 필요한 아카이브 파일을 검색하고 WordPress를 설치합니다.

```
cd /home/ec2-user/environment
wget https://wordpress.org/latest.tar.gz
tar xvf latest.tar.gz
```

4. `touch` 명령을 실행하여 `environment/wordpress` 디렉터리에 `wp-config.php`라는 파일을 생성합니다.

```
touch wordpress/wp-config.php
```

5. IDE 편집기 또는 vim을 사용하여 `wp-config.php`로 업데이트한 다음 샘플 데이터를 설정으로 바꿉니다.

```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', 'wp_test' );

/** MySQL database username */
define( 'DB_USER', 'wp_user' );

/** MySQL database password */
define( 'DB_PASSWORD', 'YourSecurePassword' );

/** MySQL hostname */
define( 'DB_HOST', 'localhost' );

/** Database Charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8' );

/** The Database Collate type. Don't change this if in doubt. */
```

```
define( 'DB_COLLATE', '' );

define('FORCE_SSL', true);

if ($_SERVER['HTTP_X_FORWARDED_PROTO'] == 'https') $_SERVER['HTTPS'] = 'on';
```

3 단계: Apache HTTP 서버 구성

1. AWS Cloud9 IDE 터미널 창에서 Apache가 설치되어 있는지 확인합니다.

   ```
   httpd -v
   ```

   Apache 서버가 설치되어 있지 않으면 다음 명령을 실행합니다.

   ```
   sudo yum install -y httpd
   ```

2. `/etc/httpd/conf.d` 디렉터리로 이동합니다. 이 디렉터리는 Apache의 가상 호스트 구성 파일의 위치입니다. 그런 다음 `ln` 명령을 사용하여 앞서 생성한 `wordpress.conf`를 현재 작업 디렉터리(`/etc/httpd/conf.d`)에 연결합니다.

   ```
   cd /etc/httpd/conf.d
   sudo ln -s /home/ec2-user/environment/config/wordpress.conf
   ```

3. 이제 `/var/www` 디렉터리로 이동합니다. 이 디렉터리는 Apache 서버의 기본 루트 폴더입니다. 그리고 `ln` 명령을 사용하여 앞서 생성한 `wordpress` 디렉터리를 현재 작업 디렉터리(`/var/www`)에 연결합니다.

   ```
   cd /var/www
   sudo ln -s /home/ec2-user/environment/wordpress
   ```

4. `chmod` 명령을 실행하여 Apache 서버가 `wordpress` 하위 디렉터리의 톤텐츠를 실행하도록 허용합니다.

   ```
   sudo chmod +x /home/ec2-user/
   ```

5. 이제 Apache 서버를 다시 시작하면 새 구성이 감지됩니다.

   ```
   sudo service httpd restart
   ```

4 단계: WordPress 웹 콘텐츠 미리 보기

1. AWS Cloud9 IDE를 사용하여 `index.html`이라는 새 파일을 `environment/wordpress` 디렉터리에 생성합니다.
2. HTML 형식의 텍스트를 `index.html`에 추가합니다. 예:

   ```
   <h1>Hello World!</h1>
   ```

3. 환경 창에서 `index.html` 파일을 선택한 다음 미리 보기, 실행 중인 애플리케이션 미리 보기를 선택합니다.

   Hello World! 메시지가 표시된 웹 페이지가 애플리케이션 미리 보기 탭에 나타납니다. 선호하는 브라우저에서 웹 콘텐츠를 보려면 [새 창으로 팝업(Pop Out Into a New Window)]을 선택합니다.

   `index.html` 파일을 삭제하고 애플리케이션 미리 보기 탭을 새로 고치면 WordPress 구성 페이지가 표시됩니다.

혼합 콘텐츠 오류 관리

웹 브라우저는 HTTPS 및 HTTP 스크립트 또는 콘텐츠를 동시에 로드하는 경우 WordPress 사이트에 대해 혼합 콘텐츠 오류를 표시합니다. 오류 메시지의 표현은 사용 중인 웹 브라우저에 따라 다르지만 사이트에 대한 연결이 안전하지 않거나 완전하게 보안되지 않는다는 메시지가 표시됩니다. 또한 웹 브라우저가 혼합 콘텐츠에 대한 액세스를 차단합니다.

> **Important**
>
> 기본적으로 AWS Cloud9 IDE의 애플리케이션 미리 보기 탭에서 액세스하는 모든 웹 페이지는 HTTPS 프로토콜을 자동으로 사용합니다. 페이지의 URI에 안전하지 않은 `http` 프로토콜이 있으면 자동으로 `https`로 대체됩니다. 그리고 수동으로 `https`를 다시 `http`로 변경하더라도 안전하지 않은 콘텐츠에 액세스할 수 없습니다.
> 웹 사이트에 대한 HTTPS를 구현하는 방법에 대한 지침은 WordPress 문서를 참조하세요.

AWS Cloud9용 Java 자습서

> **Important**
>
> 2GiB 이상의 메모리가 있는 EC2 인스턴스에서 지원하는 AWS Cloud9 개발 환경을 사용하는 경우 향상된 Java 지원을 활성화하는 것이 좋습니다. 이를 통해 코드 완성, 오류에 대한 linting, 컨텍스트별 작업, 디버깅 옵션(예: 중단점 및 단계별 실행) 등의 생산성 기능에 액세스할 수 있습니다.
> 자세한 내용은 Java 개발을 위한 향상된 지원 섹션을 참조하세요.

이 자습서를 사용하면 AWS Cloud9 개발 환경에서 Java 코드를 실행할 수 있습니다.

이 자습서를 따르고 이 샘플을 생성하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2 및 Amazon S3 같은 서비스에 대한 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금 및 Amazon S3 요금을 참조하세요.

주제

필수 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는

운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 내용은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.

- 기존 환경에 대한 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열 때 AWS Cloud9은 웹 브라우저에서 환경을 위한 IDE를 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기 섹션을 참조하세요.

1단계: 필수 도구 설치

이 단계에서는 AWS Cloud9 개발 환경에 Java 개발 도구 세트를 설치합니다. Oracle JDK 또는 OpenJDK와 같은 Java 개발 도구 세트가 환경에 이미 설치되어 있는 경우 2단계: 코드 추가 섹션으로 진행합니다. 이 샘플은 OpenJDK 8에서 개발되었으며 다음 절차를 완료하여 사용자 환경에 설치할 수 있습니다.

1. OpenJDK 8이 이미 설치되었는지 확인합니다. 이렇게 하려면 AWS Cloud9 IDE의 터미널 세션에서 **-version** 옵션을 사용하여 Java 러너의 명령줄 버전을 실행합니다. (터미널 세션을 새로 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다.)

   ```
   java -version
   ```

   이전 명령의 출력을 기준으로 다음 중 하나를 수행합니다.

   - 출력에 `java` 명령을 찾을 수 없다고 표시될 경우 이 절차의 2단계를 진행하여 OpenJDK 8을 설치합니다.
   - 출력에 `Java(TM)`, `Java Runtime Environment`, `Java SE`, `J2SE` 또는 `Java2`로 시작하는 값이 포함되어 있는 경우, OpenJDK가 설치되지 않았거나 기본 Java 개발 도구 세트로 설정되지 않은 것입니다. 이 절차의 2단계를 계속 진행하여 OpenJDK 8을 설치한 다음 OpenJDK 8을 사용하도록 전환합니다.
   - 출력에 `java version 1.8` 및 `OpenJDK`로 시작하는 값이 포함되어 있는 경우 2단계: 코드 추가 섹션으로 건너뜁니다. 이 샘플에 대해 OpenJDK 8이 올바르게 설치되어 있습니다.
   - 출력에 `1.8`보다 이전 `java version`과 `OpenJDK`로 시작하는 값이 포함되어 있는 경우 이 절차의 2단계를 계속 진행하여 설치된 OpenJDK 버전을 OpenJDK 8로 업그레이드합니다.
2. 최신 보안 업데이트 및 버그 수정 사항이 설치되었는지 확인합니다. 이렇게 하려면 **update** 명령을 사용하여 yum 도구(Amazon Linux) 또는 apt 도구(Ubuntu Server)를 실행합니다.

   Amazon Linux의 경우:

   ```
   sudo yum -y update
   ```

Ubuntu Server:

```
sudo apt update
```

3. OpenJDK 8을 설치합니다. 이렇게 하려면 OpenJDK 8 패키지를 지정하고 **install** 명령을 사용하여 yum 도구(Amazon Linux) 또는 apt 도구(Ubuntu Server)를 실행합니다.

   Amazon Linux의 경우:

   ```
   sudo yum -y install java-1.8.0-openjdk-devel
   ```

   Ubuntu Server:

   ```
   sudo apt install -y openjdk-8-jdk
   ```

   자세한 내용은 OpenJDK 웹 사이트에서 OpenJDK 패키지 다운로드 및 설치 방법을 참조하세요.

4. 기본 Java 개발 도구 세트를 OpenJDK 8로 전환하거나 업그레이드합니다. 이렇게 하려면 **--config** 옵션을 사용하여 **update-alternatives** 명령을 실행합니다. Java 러너 및 컴파일러의 명령줄 버전을 전환하거나 업그레이드하려면 이 명령을 두 번 실행합니다.

   ```
   sudo update-alternatives --config java
   sudo update-alternatives --config javac
   ```

   각 프롬프트에서 OpenJDK 8의 선택 번호(`java-1.8`이 포함된 번호)를 입력합니다.

5. Java 러너 및 컴파일러의 명령줄 버전이 OpenJDK 8을 사용하고 있는지 확인합니다. 이렇게 하려면 `-version` 옵션을 사용하여 Java 러너 및 컴파일러의 명령줄 버전을 실행합니다.

   ```
   java -version
   javac -version
   ```

   OpenJDK 8이 올바르게 설치되고 설정된 경우, Java 러너 버전 출력에 `openjdk version 1.8`로 시작하는 값이 포함되며, Java 컴파일러 버전 출력이 `javac 1.8`이라는 값으로 시작됩니다.

2단계: 코드 추가

AWS Cloud9 IDE에서 다음 코드를 포함하는 파일을 생성하고 이 파일을 hello.java라는 이름으로 저장합니다. (파일을 생성하려면 메뉴 모음에서 [파일(File)], [새 파일(New File)]을 선택합니다. 파일을 저장하려면 [파일(File)], [저장(Save)]을 선택합니다.)

```
public class hello {

  public static void main(String []args) {
    System.out.println("Hello, World!");

    System.out.println("The sum of 2 and 3 is 5.");

    int sum = Integer.parseInt(args[0]) + Integer.parseInt(args[1]);

    System.out.format("The sum of %s and %s is %s.\n",
      args[0], args[1], Integer.toString(sum));
  }
}
```

3단계: 코드 빌드 및 실행

1. Java 컴파일러의 명령줄 버전을 사용하여 hello.java 파일을 hello.class 파일로 컴파일합니다. 이렇게 하려면 AWS Cloud9 IDE의 터미널을 사용하여 hello.java 파일과 동일한 디렉터리에서 hello.java 파일을 지정하고 Java 컴파일러를 실행합니다.

   ```
   javac hello.java
   ```

2. Java 러너의 명령줄 버전을 사용하여 hello.class 파일을 실행합니다. 이렇게 하려면 hello.class 파일과 동일한 디렉터리에서 hello.java 파일에 선언된 hello 클래스의 이름과 추가할 두 개의 정수(예: 5 및 9)를 지정하여 Java 러너를 실행합니다.

   ```
   java hello 5 9
   ```

3. 출력을 비교합니다.

   ```
   Hello, World!
   The sum of 2 and 3 is 5.
   The sum of 5 and 9 is 14.
   ```

4단계: AWS SDK for Java를 사용하도록 설정

이 샘플을 응용해 AWS SDK for Java을 사용하여 Amazon S3 버킷을 생성하고 사용 가능한 버킷을 나열한 다음 방금 생성한 버킷을 삭제합니다.

이 단계에서는 환경에 Apache Maven 또는 Gradle을 설치합니다. Maven과 Gradle은 Java 프로젝트에 사용할 수 있는 일반적인 빌드 자동화 시스템입니다. Maven 또는 Gradle을 설치한 후, 새로운 Java 프로젝트를 생성하는 데 사용합니다. 이 새 프로젝트에서는 AWS SDK for Java에 대한 참조를 추가합니다. 이 AWS SDK for Java는 Java 코드에서 Amazon S3과 같은 AWS 서비스와 상호 작용할 수 있는 편리한 방법을 제공합니다.

주제

- Maven 설정
- Gradle을 사용하여 설정

Maven 설정

1. 환경에 Maven을 설치합니다. Maven이 이미 설치되어 있는지 확인하려면 AWS Cloud9 IDE의 터미널을 사용하여 **`-version`** 옵션을 지정하고 Maven을 실행합니다.

   ```
   mvn -version
   ```

   성공할 경우, 출력에 Maven 버전 번호가 포함됩니다. Maven이 이미 설치되어 있는 경우 이 절차의 4단계로 건너뛰어 Maven을 사용하여 환경에서 새 Java 프로젝트를 생성합니다.
2. 다음 명령을 실행하는 터미널을 사용하여 Maven을 설치합니다.

   Amazon Linux의 경우, 다음 명령은 Maven이 저장되어 있는 패키지 리포지토리에 대한 정보를 가져오고 이 정보를 사용하여 Maven을 설치합니다.

   ```
   sudo wget http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-
   maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo
   sudo sed -i s/\$releasever/6/g /etc/yum.repos.d/epel-apache-maven.repo
   sudo yum install -y apache-maven
   ```

   앞의 명령에 대한 자세한 내용은 Fedora Project Wiki 웹 사이트에서 Extra Packages for Enterprise Linux(EPEL)를 참조하세요.

   Ubuntu Server의 경우 다음 명령을 대신 실행합니다.

```
sudo apt install -y maven
```

3. **`-version`** 옵션을 지정하고 Maven을 실행하여 설치를 확인합니다.

```
mvn -version
```

4. Maven을 사용하여 새 Java 프로젝트를 생성합니다. 이렇게 하려면 터미널을 사용하여, Maven이 프로젝트를 생성하도록 하려는 디렉터리(예: 환경의 루트 디렉터리)에서 다음 명령을 실행합니다.

```
mvn archetype:generate -DgroupId=com.mycompany.app -DartifactId=my-app -
DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

앞의 명령은 사용자 환경에서 프로젝트에 대해 다음과 같은 디렉터리 구조를 생성합니다.

```
my-app
  |- src
  |    `- main
  |         `- java
  |              `- com
  |                   `- mycompany
  |                        `- app
  |                             `-App.java
  |- test
  |    `- java
  |         `- com
  |              `- mycompany
  |                   `- app
  |                        `- AppTest.java
  `- pom.xml
```

앞의 디렉터리 구조에 대한 자세한 내용은 Apache Maven Project 웹 사이트에서 Maven Quickstart Archetype 및 표준 디렉터리 레이아웃 소개를 참조하세요.

5. 프로젝트의 Project Object Model(POM) 파일을 수정합니다. (POM 파일은 Maven 프로젝트의 설정을 정의합니다.) 이렇게 하려면 [환경(Environment)] 창에서 `my-app/pom.xml` 파일을 엽니다. 편집기에서 이 파일의 현재 내용을 다음 코드로 바꾼 다음 `pom.xml` 파일을 저장합니다.

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"
```

```
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/
maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.mycompany.app</groupId>
  <artifactId>my-app</artifactId>
  <packaging>jar</packaging>
  <version>1.0-SNAPSHOT</version>
  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-assembly-plugin</artifactId>
        <version>3.6.0</version>
        <configuration>
          <descriptorRefs>
            <descriptorRef>jar-with-dependencies</descriptorRef>
          </descriptorRefs>
          <archive>
            <manifest>
              <mainClass>com.mycompany.app.App</mainClass>
            </manifest>
          </archive>
        </configuration>
        <executions>
          <execution>
            <phase>package</phase>
              <goals>
                <goal>single</goal>
              </goals>
          </execution>
        </executions>
      </plugin>
    </plugins>
  </build>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>com.amazonaws</groupId>
      <artifactId>aws-java-sdk</artifactId>
```

```
      <version>1.11.330</version>
    </dependency>
  </dependencies>
</project>
```

앞의 POM 파일에는 다음과 같은 선언을 지정하는 프로젝트 설정이 포함되어 있습니다.

- `my-app`의 `artifactid` 설정은 프로젝트의 루트 디렉터리 이름을 설정하고, `com.mycompany.app`의 `group-id` 설정은 `com/mycompany/app` 하위 디렉터리 구조와 `App.Java` 및 `AppTest.java` 파일의 `package`선언을 설정합니다.
- `my-app`의 `artifactId` 설정은 `jar`의 `packaging` 설정, `1.0-SNAPSHOT`의 `version` 설정, 그리고 `jar-with-dependencies`의 `descriptorRef` 설정과 함께 출력 JAR 파일의 이름(`my-app-1.0-SNAPSHOT-jar-with-dependencies.jar`)을 설정합니다.
- `plugin` 섹션은 모든 종속성을 포함하는 단일 JAR이 빌드된다는 것을 선언합니다.
- `groupId` 설정 `com.amazon.aws` 및 `artifactId` 설정 `aws-java-sdk`가 있는 `dependency` 섹션에는 AWS SDK for Java 라이브러리 파일이 포함되어 있습니다. 사용할 AWS SDK for Java 버전은 `version` 설정에 의해 선언됩니다. 다른 버전을 사용하려면 이 버전 번호를 바꿉니다.

5단계: 환경에서 AWS 자격 증명 관리 설정 단원으로 이동합니다.

Gradle을 사용하여 설정

1. 환경에 Gradle을 설치합니다. Gradle이 이미 설치되어 있는지 확인하려면 AWS Cloud9 IDE의 터미널을 사용하여 **`-version`** 옵션을 지정하고 Gradle을 실행합니다.

   ```
   gradle -version
   ```

   성공할 경우, 출력에 Gradle 버전 번호가 포함됩니다. Gradle이 이미 설치되어 있는 경우 이 절차의 4단계로 건너뛰어 Gradle을 사용하여 환경에서 새 Java 프로젝트를 생성합니다.

2. 다음 명령을 실행하는 터미널을 사용하여 Gradle을 설치합니다. 이 명령은 SDKMAN! 도구를 설치하고 실행한 다음, SDKMAN!을 사용하여 Gradle 최신 버전을 설치합니다.

   ```
   curl -s "https://get.sdkman.io" | bash
   source "$HOME/.sdkman/bin/sdkman-init.sh"
   sdk install gradle
   ```

앞의 명령에 대한 자세한 내용은 SDKMAN! 웹 사이트의 설치와 Gradle 웹 사이트의 패키지 관리자를 사용하여 설치를 참조하세요.

3. **-version** 옵션을 지정하고 Gradle을 실행하여 설치를 확인합니다.

```
gradle -version
```

4. Gradle을 사용하여 환경에서 새 Java 프로젝트를 생성합니다. 이렇게 하려면 터미널에서 다음 명령을 실행하여 프로젝트의 디렉터리를 생성한 다음 디렉터리로 전환합니다.

```
mkdir my-app
cd my-app
```

5. 다음 명령을 실행하여 Gradle이 환경의 `my-app` 디렉터리에 새 Java 애플리케이션 프로젝트를 생성하도록 합니다.

```
gradle init --type java-application
```

앞의 명령은 사용자 환경에서 프로젝트에 대해 다음과 같은 디렉터리 구조를 생성합니다.

```
my-app
  |- .gradle
  |   `- (various supporting project folders and files)
  |- gradle
  |   `- (various supporting project folders and files)
  |- src
  |   |- main
  |   |    `- java
  |   |         `- App.java
  |   `- test
  |        `- java
  |             `- AppTest.java
  |- build.gradle
  |- gradlew
  |- gradlew.bat
  `- settings.gradle
```

6. 프로젝트의 `AppTest.java`를 수정합니다. (이렇게 하지 않으면 프로젝트가 정상적으로 빌드되거나 실행되지 않을 수 있습니다.) 이렇게 하려면 [환경(Environment)] 창에서 `my-app/src/test/java/AppTest.java` 파일을 엽니다. 편집기에서 이 파일의 현재 내용을 다음 코드로 바꾼 다음 `AppTest.java` 파일을 저장합니다.

```
import org.junit.Test;
import static org.junit.Assert.*;

public class AppTest {
  @Test public void testAppExists () {
    try {
      Class.forName("com.mycompany.app.App");
    } catch (ClassNotFoundException e) {
      fail("Should have a class named App.");
    }
  }
}
```

7. 프로젝트의 build.gradle 파일을 수정합니다. (build.gradle 파일은 Gradle 프로젝트의 설정을 정의합니다.) 이렇게 하려면 [환경(Environment)] 창에서 my-app/build.gradle 파일을 엽니다. 편집기에서 이 파일의 현재 내용을 다음 코드로 바꾼 다음 build.gradle 파일을 저장합니다.

```
apply plugin: 'java'
apply plugin: 'application'

repositories {
  jcenter()
  mavenCentral()
}

buildscript {
  repositories {
    mavenCentral()
  }
  dependencies {
    classpath "io.spring.gradle:dependency-management-plugin:1.0.3.RELEASE"
  }
}

apply plugin: "io.spring.dependency-management"

dependencyManagement {
  imports {
    mavenBom 'com.amazonaws:aws-java-sdk-bom:1.11.330'
  }
}
```

```
dependencies {
  compile 'com.amazonaws:aws-java-sdk-s3'
  testCompile group: 'junit', name: 'junit', version: '4.12'
}

run {
  if (project.hasProperty("appArgs")) {
    args Eval.me(appArgs)
  }
}

mainClassName = 'App'
```

앞의 `build.gradle` 파일에는 다음과 같은 선언을 지정하는 프로젝트 설정이 포함되어 있습니다.

- `io.spring.dependency-management` 플러그인은 프로젝트에 대한 AWS SDK for Java 종속 구성 요소를 관리할 AWS SDK for Java Maven Bill of Materials(BOM)를 가져오는 데 사용됩니다. `classpath`는 사용할 버전을 선언합니다. 다른 버전을 사용하려면 이 버전 번호를 바꿉니다.
- `com.amazonaws:aws-java-sdk-s3`는 AWS SDK for Java 라이브러리 파일의 Amazon S3 부분을 포함합니다. `mavenBom`은 사용할 버전을 선언합니다. 다른 버전을 사용하려면 이 버전 번호를 바꿉니다.

5단계: 환경에서 AWS 자격 증명 관리 설정

AWS SDK for Java를 사용하여 AWS 서비스를 호출할 때마다 호출과 함께 AWS 자격 증명 세트를 제공합니다. 이러한 자격 증명은 AWS SDK for Java가 해당 호출을 할 수 있는 적절한 권한이 있는지 여부를 결정합니다. 자격 증명으로 적절한 권한이 확인되지 않는 경우 호출이 실패합니다.

이 단계에서는 환경 내에서 자격 증명을 저장합니다. 이렇게 하려면 AWS Cloud9의 환경에서 AWS 서비스 호출 섹션의 지침을 수행한 다음 이 주제로 돌아옵니다.

자세한 내용은 AWS SDK for Java 개발자 안내서에서 개발을 위한 AWS 자격 증명 및 리전 설정을 참조하세요.

6단계: AWS SDK 코드 추가

이 단계에서는 코드를 추가하여 Amazon S3와 상호 작용하고, 버킷을 생성하고, 사용 가능한 버킷을 나열한 다음 막 생성한 버킷을 삭제합니다.

[환경(Environment)] 창에서 Maven의 경우 `my-app/src/main/java/com/mycompany/app/App.java` 파일 또는 Gradle의 경우 `my-app/src/main/java/App.java` 파일을 엽니다. 편집기에서 이 파일의 현재 내용을 다음 코드로 바꾼 다음 `App.java` 파일을 저장합니다.

```
package com.mycompany.app;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.AmazonS3Exception;
import com.amazonaws.services.s3.model.Bucket;
import com.amazonaws.services.s3.model.CreateBucketRequest;

import java.util.List;

public class App {

    private static AmazonS3 s3;

    public static void main(String[] args) {
        if (args.length < 2) {
            System.out.format("Usage: <the bucket name> <the AWS Region to use>\n" +
                    "Example: my-test-bucket us-east-2\n");
            return;
        }

        String bucket_name = args[0];
        String region = args[1];

        s3 = AmazonS3ClientBuilder.standard()
                .withCredentials(new ProfileCredentialsProvider())
                .withRegion(region)
                .build();

        // List current buckets.
        ListMyBuckets();

        // Create the bucket.
```

```
        if (s3.doesBucketExistV2(bucket_name)) {
            System.out.format("\nCannot create the bucket. \n" +
                    "A bucket named '%s' already exists.", bucket_name);
            return;
        } else {
            try {
                System.out.format("\nCreating a new bucket named '%s'...\n\n",
 bucket_name);
                s3.createBucket(new CreateBucketRequest(bucket_name, region));
            } catch (AmazonS3Exception e) {
                System.err.println(e.getErrorMessage());
            }
        }

        // Confirm that the bucket was created.
        ListMyBuckets();

        // Delete the bucket.
        try {
            System.out.format("\nDeleting the bucket named '%s'...\n\n", bucket_name);
            s3.deleteBucket(bucket_name);
        } catch (AmazonS3Exception e) {
            System.err.println(e.getErrorMessage());
        }

        // Confirm that the bucket was deleted.
        ListMyBuckets();

    }

    private static void ListMyBuckets() {
        List<Bucket> buckets = s3.listBuckets();
        System.out.println("My buckets now are:");

        for (Bucket b : buckets) {
            System.out.println(b.getName());
        }
    }

}
```

7단계: AWS SDK 코드 빌드 및 실행

이전 단계의 코드를 실행하려면 터미널에서 다음 명령을 실행합니다. 이 명령은 Maven 또는 Gradle을 사용하여 프로젝트의 JAR 실행 파일을 만든 다음 Java 러너를 사용하여JAR을 실행합니다. JAR은 Amazon S3에 생성할 버킷 이름(예: `my-test-bucket`)과 버킷을 생성할 AWS 리전의 ID(예: `us-east-2`)를 입력으로 사용하여 실행됩니다.

Maven의 경우 다음 명령을 실행합니다.

```
cd my-app
mvn package
java -cp target/my-app-1.0-SNAPSHOT-jar-with-dependencies.jar com.mycompany.app.App my-test-bucket us-east-2
```

Gradle의 경우 다음 명령을 실행합니다.

```
gradle build
gradle run -PappArgs="['my-test-bucket', 'us-east-2']"
```

결과를 다음 출력과 비교합니다.

```
My buckets now are:

Creating a new bucket named 'my-test-bucket'...

My buckets now are:

my-test-bucket

Deleting the bucket named 'my-test-bucket'...

My buckets now are:
```

8단계: 정리

이 샘플 사용 후 AWS 계정에 계속하여 요금이 부과되지 않도록 환경을 삭제해야 합니다. 지침은 AWS Cloud9에서 환경 삭제 단원을 참조하세요.

AWS Cloud9용 C++ 자습서

이 자습서를 사용하면 AWS Cloud9 개발 환경에서 C++ 코드를 실행할 수 있습니다. 이 코드는 Amazon Web Services에 연결하는 데 사용할 수 있는 모듈화된 플랫폼 간 오픈 소스 라이브러리인 AWS SDK for C++에서 제공되는 리소스도 사용합니다.

이 자습서를 따르고 이 샘플을 생성하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2 및 Amazon S3 같은 서비스에 대한 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금 및 Amazon S3 요금을 참조하세요.

주제

필수 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는 운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 내용은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.
- 기존 환경에 대한 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열 때 AWS Cloud9은 웹 브라우저에서 환경을 위한 IDE를 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기 섹션을 참조하세요.

1단계: g++ 및 필요한 개발 패키지 설치

C++ 애플리케이션을 빌드하고 실행하려면 GNU Complier Collection(GCC)에서 제공하는 C++ 컴파일러인 g++ 같은 유틸리티가 필요합니다.

또한 libcurl, libopenssl, libuuid, zlib용 헤더 파일(-dev 패키지)과 선택적으로 Amazon Polly 지원을 위한 libpulse도 필요합니다.

개발 도구를 설치하는 프로세스는 Amazon Linux/Amazon Linux 2 인스턴스를 사용하는지 아니면 Ubuntu 인스턴스를 사용하는지에 따라 약간 다릅니다.

Amazon Linux-based systems

AWS Cloud9 터미널에서 다음 명령을 실행하여 gcc가 이미 설치되어 있는지 확인할 수 있습니다.

```
g++ --version
```

g++가 설치되어 있지 않은 경우 'Development Tools'라는 패키지 그룹의 일부로 손쉽게 설치할 수 있습니다. 이러한 도구는 yum groupinstall 명령을 사용하여 인스턴스에 추가됩니다.

```
sudo yum groupinstall "Development Tools"
```

g++ --version을 다시 실행하여 컴파일러가 설치되었는지 확인합니다.

이제 시스템의 패키지 관리자를 사용하여 필요한 라이브러리의 패키지를 설치합니다.

```
sudo yum install libcurl-devel openssl-devel libuuid-devel pulseaudio-libs-devel
```

Ubuntu-based systems

AWS Cloud9 터미널에서 다음 명령을 실행하여 gcc가 이미 설치되어 있는지 확인할 수 있습니다.

```
g++ --version
```

gcc가 설치되어 있지 않은 경우, 다음 명령을 실행하여 Ubuntu 기반 시스템에 설치할 수 있습니다.

```
sudo apt update
sudo apt install build-essential
sudo apt-get install manpages-dev
```

g++ --version을 다시 실행하여 컴파일러가 설치되었는지 확인합니다.

이제 시스템의 패키지 관리자를 사용하여 필요한 라이브러리의 패키지를 설치합니다.

```
sudo apt-get install libcurl4-openssl-dev libssl-dev uuid-dev zlib1g-dev libpulse-
dev
```

2단계: CMake 설치

소스 코드에서 실행 파일을 작성하는 프로세스를 자동화하는 cmake 도구를 설치해야 합니다.

1. IDE 터미널 창에서 다음 명령을 실행하여 필요한 아카이브를 가져옵니다.

```
wget https://cmake.org/files/v3.18/cmake-3.18.0.tar.gz
```

2. 아카이브에서 파일을 추출하고 압축을 푼 파일이 포함된 디렉터리로 이동합니다.

```
tar xzf cmake-3.18.0.tar.gz
cd cmake-3.18.0
```

3. 그런 다음 부트 스트랩 스크립트를 실행하고 다음 명령을 실행하여 cmake를 설치합니다.

```
./bootstrap
make
sudo make install
```

4. 다음 명령을 실행하여 도구를 설치했는지 확인합니다.

```
cmake --version
```

3단계: SDK for C++ 가져오기 및 빌드

AWS SDK for C++를 설정하려면 소스에서 직접 SDK를 빌드하거나 패키지 관리자를 사용하여 라이브러리를 다운로드 할 수 있습니다. 사용 가능한 옵션에 대한 자세한 내용은 AWS SDK for C++ 개발자 가이드에서 AWS SDK for C++를 사용하여 시작하기를 참조하세요.

이 샘플에서는 git를 사용하여 SDK 소스 코드를 복제하고 cmake SDK for C++를 빌드하는 방법을 보여줍니다.

1. 터미널에서 다음 명령을 실행하여 원격 리포지토리를 복제하고 AWS Cloud9 환경의 모든 git 하위 모듈을 재귀적으로 가져옵니다.

```
git clone --recurse-submodules https://github.com/aws/aws-sdk-cpp
```

2. 새aws-sdk-cpp 디렉터리로 이동하여 AWS SDK for C++를 빌드할 하위 디렉터리를 생성하고 다음으로 이동합니다.

```
cd aws-sdk-cpp
mkdir sdk_build
cd sdk_build
```

3.

> **Note**
>
> 시간을 절약하기 위해 이 단계에서는 AWS SDK for C++의 Amazon S3 부분만 빌드합니다. 전체 SDK를 빌드하려면 cmake 명령에서 -DBUILD_ONLY=s3를 생략하세요.
> 전체 SDK for C++를 구축하는 데에는 Amazon EC2 인스턴스 또는 자체 서버에서 사용할 수 있는 컴퓨팅 리소스에 따라 1시간 이상 걸릴 수 있습니다.

다음 명령을 실행함으로써 cmake를 사용하여 SDK for C++의 Amazon S3 부분을 sdk_build 디렉터리에 빌드합니다.

```
cmake .. -DBUILD_ONLY=s3
```

4. 이제 빌드된 SDK에 액세스할 수 있도록 make install 명령을 실행합니다.

```
sudo make install
cd ..
```

4단계: C++ 및 CMakeLists 파일 생성

이 단계에서는 프로젝트 사용자가 Amazon S3 버킷과 상호 작용할 수 있도록 하는 C++ 파일을 생성합니다.

또한 cmake가 C++ 라이브러리를 빌드하는 데 사용하는 지침을 제공하는 CMakeLists.txt 파일을 생성합니다.

1. AWS Cloud9 IDE에서 이 콘텐츠를 포함하는 파일을 생성하고, 이 파일을 환경의 루트(/)에 s3-demo.cpp라는 이름으로 저장합니다.

```
#include <iostream>
#include <aws/core/Aws.h>
#include <aws/s3/S3Client.h>
#include <aws/s3/model/Bucket.h>
#include <aws/s3/model/CreateBucketConfiguration.h>
```

```
#include <aws/s3/model/CreateBucketRequest.h>
#include <aws/s3/model/DeleteBucketRequest.h>

// Look for a bucket among all currently available Amazon S3 buckets.
bool FindTheBucket(const Aws::S3::S3Client &s3Client,
                   const Aws::String &bucketName) {

    Aws::S3::Model::ListBucketsOutcome outcome = s3Client.ListBuckets();

    if (outcome.IsSuccess()) {

        std::cout << "Looking for a bucket named '" << bucketName << "'..."
                  << std::endl << std::endl;

        Aws::Vector<Aws::S3::Model::Bucket> bucket_list =
                outcome.GetResult().GetBuckets();

        for (Aws::S3::Model::Bucket const &bucket: bucket_list) {
            if (bucket.GetName() == bucketName) {
                std::cout << "Found the bucket." << std::endl << std::endl;

                return true;
            }
        }

        std::cout << "Could not find the bucket." << std::endl << std::endl;
    }
    else {
        std::cerr << "ListBuckets error: "
                  << outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}

// Create an Amazon S3 bucket.
bool CreateTheBucket(const Aws::S3::S3Client &s3Client,
                     const Aws::String &bucketName,
                     const Aws::String& region) {

    std::cout << "Creating a bucket named '"
              << bucketName << "'..." << std::endl << std::endl;

    Aws::S3::Model::CreateBucketRequest request;
```

```
    request.SetBucket(bucketName);

     if (region != "us-east-1") {
        Aws::S3::Model::CreateBucketConfiguration createBucketConfig;
        createBucketConfig.SetLocationConstraint(

 Aws::S3::Model::BucketLocationConstraintMapper::GetBucketLocationConstraintForName(
                        region));
        request.SetCreateBucketConfiguration(createBucketConfig);
    }

    Aws::S3::Model::CreateBucketOutcome outcome =
            s3Client.CreateBucket(request);

    if (outcome.IsSuccess()) {
        std::cout << "Bucket created." << std::endl << std::endl;
    }
    else {
        std::cerr << "CreateBucket error: "
                  << outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}

// Delete an existing Amazon S3 bucket.
bool DeleteTheBucket(const Aws::S3::S3Client &s3Client,
                     const Aws::String &bucketName) {

    std::cout << "Deleting the bucket named '"
              << bucketName << "'..." << std::endl << std::endl;

    Aws::S3::Model::DeleteBucketRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::DeleteBucketOutcome outcome =
            s3Client.DeleteBucket(request);

    if (outcome.IsSuccess()) {
        std::cout << "Bucket deleted." << std::endl << std::endl;
    }
    else {
        std::cerr << "DeleteBucket error: "
                  << outcome.GetError().GetMessage() << std::endl;
```

```
    }

    return outcome.IsSuccess();
}

#ifndef TESTING_BUILD
// Create an S3 bucket and then delete it.
// Before and after creating the bucket, and again after deleting the bucket,
// try to determine whether that bucket still exists.
int main(int argc, char *argv[]) {

    if (argc < 3) {
        std::cout << "Usage: s3-demo <bucket name> <AWS Region>" << std::endl
                  << "Example: s3-demo my-bucket us-east-1" << std::endl;
        return 1;
    }

    Aws::SDKOptions options;
    Aws::InitAPI(options);
    {
        Aws::String bucket_name = argv[1];
        Aws::String region = argv[2];

        Aws::Client::ClientConfiguration config;

        config.region = region;

        Aws::S3::S3Client s3_client(config);

        if (!FindTheBucket(s3_client, bucket_name)) {
            return 1;
        }

        if (!CreateTheBucket(s3_client, bucket_name, region)) {
            return 1;
        }

        if (!FindTheBucket(s3_client, bucket_name)) {
            return 1;
        }

        if (!DeleteTheBucket(s3_client, bucket_name)) {
            return 1;
        }
```

```
        if (!FindTheBucket(s3_client, bucket_name)) {
            return 1;
        }
    }
    Aws::ShutdownAPI(options);

    return 0;
}
#endif  // TESTING_BUILD
```

2. 이 콘텐츠를 포함하는 두 번째 파일을 생성하고, 이 파일을 환경의 루트(/)에 CMakeLists.txt라는 이름으로 저장합니다. 이 파일을 사용하면 코드를 실행 파일로 빌드할 수 있습니다.

```
# A minimal CMakeLists.txt file for the AWS SDK for C++.

# The minimum version of CMake that will work.
cmake_minimum_required(VERSION 2.8)

# The project name.
project(s3-demo)

# Locate the AWS SDK for C++ package.
set(AWSSDK_ROOT_DIR, "/usr/local/")
set(BUILD_SHARED_LIBS ON)
find_package(AWSSDK REQUIRED COMPONENTS s3)

# The executable name and its source files.
add_executable(s3-demo s3-demo.cpp)

# The libraries used by your executable.
target_link_libraries(s3-demo ${AWSSDK_LINK_LIBRARIES})
```

5단계: C++ 코드 빌드 및 실행

1. s3-demo.cpp 및 CMakeLists.txt를 저장한 환경의 루트 디렉터리에서 cmake를 실행하여 프로젝트를 빌드합니다.

```
cmake .
make
```

2. 이제 명령줄에서 프로그램을 실행할 수 있습니다. 다음 명령에서 `my-unique-bucket-name`을 Amazon S3 버킷의 고유 이름으로 바꾸고, 필요한 경우 `us-east-1`을 버킷을 생성하려는 다른 AWS 리전의 식별자로 바꿉니다.

```
./s3-demo my-unique-bucket-name us-east-1
```

프로그램이 제대로 실행되면 다음과 비슷한 출력이 반환됩니다.

```
Looking for a bucket named 'my-unique-bucket-name'...

Could not find the bucket.

Creating a bucket named 'my-unique-bucket-name'...

Bucket created.

Looking for a bucket named 'my-unique-bucket-name'...

Found the bucket.

Deleting the bucket named 'my-unique-bucket-name'...

Bucket deleted.

Looking for a bucket named 'my-unique-bucket-name'...

Could not find the bucket.
```

6단계: 정리

이 샘플의 사용을 끝낸 후에는 AWS 계정에 계속하여 요금이 부과되지 않도록 환경을 삭제합니다. 지침은 AWS Cloud9에서 환경 삭제 단원을 참조하세요.

AWS Cloud9용 Python 자습서

이 자습서에서는 AWS Cloud9 개발 환경에서 Python 코드를 실행하는 방법을 보여줍니다.

이 자습서를 이용하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon Elastic Compute Cloud(Amazon EC2) 및 Amazon Simple Storage Service(Amazon S3)와 같은 서비스에 대해 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금 및 Amazon S3 요금을 참조하세요.

주제

필수 조건

이 자습서를 사용하기 전에 다음 요구 사항을 충족하는지 확인하십시오.

- 기존 AWS Cloud9 EC2 개발 환경이어야 함

  이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 세부 정보는 EC2 환경 생성을 참조하십시오.

  다른 유형의 환경 또는 운영 체제가 있다면 이 자습서의 지침을 적용해야 합니다.
- 이 환경에 AWS Cloud9 IDE 개방

  환경을 열 때 AWS Cloud9은 웹 브라우저에서 환경을 위한 IDE를 엽니다. 세부 정보는 AWS Cloud9에서 환경 열기을 참조하십시오.

1단계: Python 설치

1. AWS Cloud9 IDE의 터미널 세션에서 **`python --version`** 명령을 실행하여 Python이 이미 설치되었는지 여부를 확인합니다. (새 터미널 세션을 시작하려면 메뉴 모음에서 창, New Terminal(새 터미널)을 선택합니다.) Python이 설치된 경우 2단계: 코드 추가 섹션으로 건너뜁니다.
2. **`yum update`**(Amazon Linux) 또는 **`apt update`**(Ubuntu Server) 명령을 실행하여 최신 보안 업데이트와 버그 수정이 설치되어 있는지 확인합니다.

   Amazon Linux의 경우:

```
sudo yum -y update
```

Ubuntu Server:

```
sudo apt update
```

3. **`install`** 명령을 실행하여 Python을 설치합니다.

   Amazon Linux의 경우:

   ```
   sudo yum -y install python3
   ```

   Ubuntu Server:

   ```
   sudo apt-get install python3
   ```

2단계: 코드 추가

AWS Cloud9 IDE에서 다음 콘텐츠를 포함하는 파일을 생성하고 이 파일을 `hello.py`라는 이름으로 저장합니다. (파일을 생성하려면 메뉴 모음에서 [파일(File), [새 파일(New File)]을 선택합니다. 파일을 저장하려면 [파일(File)], [저장(Save)]을 선택합니다.)

```
import sys

print('Hello, World!')

print('The sum of 2 and 3 is 5.')

sum = int(sys.argv[1]) + int(sys.argv[2])

print('The sum of {0} and {1} is {2}.'.format(sys.argv[1], sys.argv[2], sum))
```

3단계: 코드 실행

1. AWS Cloud9 IDE의 메뉴 모음에서 [실행(Run)], [실행 구성(Run Configurations)], [새 실행 구성(New Run Configuration)]을 선택합니다.

2. [[새로 만들기] - 중지됨([New] - Stopped)] 탭에서 [명령(Command]에 `hello.py 5 9`를 입력합니다. 이 코드에서 5는 `sys.argv[1]`을 나타내고 9는 `sys.argv[2]`를 나타냅니다.
3. 실행을 선택하여 출력을 비교합니다.

```
Hello, World!
The sum of 2 and 3 is 5.
The sum of 5 and 9 is 14.
```

4. 기본적으로 AWS Cloud9에서는 코드의 실행기를 자동으로 선택합니다. 실행기를 변경하려면 Runner(실행기)를 선택한 다음 Python 2 또는 Python 3을 선택합니다.

> **Note**
>
> 특정 버전의 Python에 대한 사용자 지정 실행기를 만들 수 있습니다. 자세한 내용은 빌더 또는 러너 생성 섹션을 참조하세요.

4단계: AWS SDK for Python (Boto3) 설치 및 구성

AWS SDK for Python (Boto3)에서는 Python 코드를 사용하여 Amazon S3 같은 AWS 서비스와 상호 작용할 수 있습니다. 예를 들어 SDK를 사용하여 Amazon S3 버킷을 생성하고 사용 가능한 버킷을 나열한 다음 방금 생성한 버킷을 삭제할 수 있습니다.

PIP 설치

AWS Cloud9 IDE에서 **`python -m pip --version`** 명령을 실행하여 `pip`가 Python의 활성 버전에 대한 이미 설치되어 있는지 확인합니다. `pip`가 설치된 경우 다음 섹션으로 건너뜁니다.

`pip`를 설치하려면 다음 명령을 실행합니다. sudo는 사용자와 다른 환경에 있기 때문에 현재 별칭 버전과 다른 경우 사용할 Python 버전을 지정해야 합니다.

```
curl -O https://bootstrap.pypa.io/get-pip.py # Get the install script.
sudo python3 get-pip.py                      # Install pip for Python 3.
python -m pip --version                       # Verify pip is installed.
rm get-pip.py                                 # Delete the install script.
```

자세한 내용은 `pip` 웹 사이트에서 설치를 참조하세요.

AWS SDK for Python (Boto3) 설치

pip 설치 후 **pip install** 명령을 실행하여 AWS SDK for Python (Boto3)을 설치합니다.

```
sudo python3 -m pip install boto3  # Install boto3 for Python 3.
python -m pip show boto3            # Verify boto3 is installed for the current version
 of Python.
```

자세한 내용은 AWS SDK for Python (Boto3)의 Quickstart(빠른 시작)에서 "설치" 섹션을 참조하세요.

환경에서 자격 증명 설정

AWS SDK for Python (Boto3)을 사용하여 AWS 서비스를 호출할 때마다 호출과 함께 자격 증명 세트를 제공해야 합니다. 이러한 자격 증명은 SDK에 호출을 수행하는 데 필요한 권한이 있는지 여부를 결정합니다. 자격 증명이 필요한 권한을 포함하지 않으면 호출이 실패합니다.

환경에 자격 증명을 저장하려면 AWS Cloud9의 환경에서 AWS 서비스 호출의 지침을 따른 다음 이 주제로 돌아옵니다.

자세한 내용은 AWS SDK for Python (Boto3)에서 자격 증명을 참조하세요.

5단계: AWS SDK 코드 추가

Amazon S3를 사용하는 코드를 추가하여 버킷을 생성하고, 사용 가능한 버킷을 나열하고, 방금 생성한 버킷을 선택적으로 삭제합니다.

AWS Cloud9 IDE에서 다음 콘텐츠를 포함하는 파일을 생성하고 이 파일을 s3.py라는 이름으로 저장합니다.

```
import sys
import boto3
from botocore.exceptions import ClientError


def list_my_buckets(s3_resource):
    print("Buckets:\n\t", *[b.name for b in s3_resource.buckets.all()], sep="\n\t")


def create_and_delete_my_bucket(s3_resource, bucket_name, keep_bucket):
    list_my_buckets(s3_resource)

    try:
        print("\nCreating new bucket:", bucket_name)
```

```
        bucket = s3_resource.create_bucket(
            Bucket=bucket_name,
            CreateBucketConfiguration={
                "LocationConstraint": s3_resource.meta.client.meta.region_name
            },
        )
    except ClientError as e:
        print(
            f"Couldn't create a bucket for the demo. Here's why: "
            f"{e.response['Error']['Message']}"
        )
        raise

    bucket.wait_until_exists()
    list_my_buckets(s3_resource)

    if not keep_bucket:
        print("\nDeleting bucket:", bucket.name)
        bucket.delete()

        bucket.wait_until_not_exists()
        list_my_buckets(s3_resource)
    else:
        print("\nKeeping bucket:", bucket.name)


def main():
    import argparse

    parser = argparse.ArgumentParser()
    parser.add_argument("bucket_name", help="The name of the bucket to create.")
    parser.add_argument("region", help="The region in which to create your bucket.")
    parser.add_argument(
        "--keep_bucket",
        help="Keeps the created bucket. When not "
        "specified, the bucket is deleted "
        "at the end of the demo.",
        action="store_true",
    )

    args = parser.parse_args()
    s3_resource = (
        boto3.resource("s3", region_name=args.region)
        if args.region
```

```
        else boto3.resource("s3")
    )
    try:
        create_and_delete_my_bucket(s3_resource, args.bucket_name, args.keep_bucket)
    except ClientError:
        print("Exiting the demo.")


if __name__ == "__main__":
    main()
```

6단계: AWS SDK 코드 실행

1. 메뉴 모음에서 Run(실행), Run Configurations(실행 구성), New Run Configuration(새로운 실행 구성)을 선택합니다.
2. 명령에 `s3.py my-test-bucket us-west-2`를 입력합니다. 여기서 `my-test-bucket`은 생성할 버킷의 이름이고 `us-west-2`는 버킷이 생성되는 AWS 리전의 ID입니다. 기본적으로 버킷은 스크립트가 종료되기 전에 삭제됩니다. 버킷을 유지하려면 명령에 `--keep_bucket`을 추가합니다. AWS 리전 ID의 목록은 AWS 일반 참조의 Amazon Simple Storage Service 엔드포인트 및 할당량을 참조하세요.

   > **Note**
   >
   > Amazon S3 버킷 이름은 해당 AWS 계정뿐만 아니라 모든 AWS에서 고유해야 합니다.

3. Run(실행)을 선택하여 출력을 비교합니다.

```
Buckets:

        a-pre-existing-bucket

Creating new bucket: my-test-bucket
Buckets:

        a-pre-existing-bucket
        my-test-bucket

Deleting bucket: my-test-bucket
Buckets:
```

```
        a-pre-existing-bucket
```

7단계: 정리

이 자습서를 마친 후 AWS 계정에 계속하여 요금이 부과되지 않도록 AWS Cloud9 환경을 삭제해야 합니다. 지침은 AWS Cloud9에서 환경 삭제 단원을 참조하세요.

AWS Cloud9용 .NET 자습서

이 자습서를 사용하면 AWS Cloud9 개발 환경에서 .NET 코드를 실행할 수 있습니다.

이 자습서를 따르고 이 샘플을 생성하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2 및 Amazon S3 같은 서비스에 대한 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금 및 Amazon S3 요금을 참조하세요.

주제

필수 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는

운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 내용은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.

- 기존 환경에 대한 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열 때 AWS Cloud9은 웹 브라우저에서 환경을 위한 IDE를 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기 섹션을 참조하세요.

1단계: 필수 도구 설치

이 단계에서는 이 샘플을 실행하는 데 필요한 .NET SDK를 환경에 설치합니다.

1. .NET SDK의 최신 버전이 환경에 이미 설치되었는지 여부를 확인합니다. 이렇게 하려면 AWS Cloud9 IDE의 터미널 세션에서 **`--version`** 옵션을 지정하여 .NET Core 명령줄 인터페이스(CLI)를 실행합니다.

   ```
   dotnet --version
   ```

   .NET 명령줄 도구 버전이 표시되고 버전이 2.0 이상이면 3단계: .NET 콘솔 애플리케이션 프로젝트 만들기 단원으로 이동합니다. 버전이 2.0 미만이거나 `bash: dotnet: command not found`와 같은 오류가 표시되면 계속해서 .NET SDK를 설치합니다.

2. Amazon Linux의 경우 AWS Cloud9 IDE의 터미널 세션에서 다음 명령을 실행하여 최신 보안 업데이트 및 버그 수정 사항이 설치되었는지 확인하고 .NET SDK에 필요한 `libunwind` 패키지를 설치합니다. (터미널 세션을 새로 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다.)

   ```
   sudo yum -y update
   sudo yum -y install libunwind
   ```

   Ubuntu Server의 경우 AWS Cloud9 IDE의 터미널 세션에서 다음 명령을 실행하면 최신 보안 업데이트 및 버그 수정 사항이 설치되어 있는지 확인할 수 있습니다. (터미널 세션을 새로 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다.)

   ```
   sudo apt -y update
   ```

3. 다음 명령을 실행하여 환경으로 .NET SDK 설치 프로그램을 다운로드합니다.

   ```
   wget https://dot.net/v1/dotnet-install.sh
   ```

4. 다음 명령을 실행하여 현재 사용자가 설치 프로그램 스크립트 실행 파일을 생성합니다.

```
sudo chmod u=rx dotnet-install.sh
```

5. 다음 명령을 실행하여 .NET SDK를 다운로드해 설치하는 설치 프로그램 스크립트를 실행합니다.

```
./dotnet-install.sh -c Current
```

6. PATH에 .NET SDK를 추가합니다. 이렇게 하려면 환경에 대한 셸 프로필(예: `.bashrc` 파일)에서 다음과 같이 환경에 대한 `PATH` 변수에 `$HOME/.dotnet` 하위 디렉터리를 추가합니다.

   a. **`vi`** 명령을 사용하여 편집을 위해 `.bashrc` 파일을 엽니다.

   ```
   vi ~/.bashrc
   ```

   b. Amazon Linux의 경우 아래쪽 화살표 또는 `j` 키를 사용하여 `export PATH`로 시작하는 행으로 이동합니다.

   Ubuntu Server의 경우 `G`를 입력하여 파일의 마지막 행으로 이동합니다.

   c. 오른쪽 화살표 또는 `$` 키를 사용하여 해당 행의 끝으로 이동합니다.

   d. `i` 키를 눌러서 삽입 모드로 전환합니다.`-- INSERT ---`가 표시 화면의 끝에 나타납니다.

   e. Amazon Linux의 경우 `:$HOME/.dotnet`을 입력하여 `$HOME/.dotnet` 하위 디렉터리를 **`PATH`** 변수에 추가합니다. 반드시 콜론 문자(`:`)를 포함해야 합니다. 이제 이 행은 다음과 같습니다.

   ```
   export PATH=$PATH:$HOME/.local/bin:$HOME/bin:$HOME/.dotnet
   ```

   Ubuntu Server의 경우 오른쪽 화살표 키를 누르고 `Enter` 키를 두 번 누른 다음 파일 끝에 직접 다음 행을 입력합니다.

   ```
   export PATH=$HOME/.dotnet:$PATH
   ```

   f. 파일을 저장합니다. 이렇게 하려면 `Esc` 키를 누릅니다. 표시 화면 끝에서 `-- INSERT ---`가 사라집니다. (파일에 쓰고 나서 파일을 종료하기 위해 `:wq`를 입력하고 나서 `Enter`를 누릅니다.

7. `.bashrc` 파일을 소싱하여 .NET SDK를 로드합니다.

```
. ~/.bashrc
```

8. **`--help`** 옵션을 지정한 상태로 .NET CLI를 실행하여 .NET SDK가 로드되었는지 확인합니다.

```
dotnet --help
```

성공하면 추가 사용법 정보와 함께 .NET SDK 버전 번호가 표시됩니다.

9. 환경에서 .NET SDK 설치 프로그램 스크립트를 더 이상 유지하지 않으려면 다음과 같이 해당 파일을 삭제할 수 있습니다.

```
rm dotnet-install.sh
```

2단계(선택 사항): Lambda 함수에 대한 .NET CLI 확장 설치

이 자습서에는 필요하지 않지만, `Amazon.Lambda.Tools` 패키지도 설치하는 경우 .NET CLI를 사용하여 AWS Lambda 함수 및 AWS Serverless Application Model 애플리케이션을 배포합니다.

1. 이 패키지를 설치하려면 다음 명령을 실행합니다.

```
dotnet tool install -g Amazon.Lambda.Tools
```

2. 이제 설치된 Lambda 도구를 가리키도록 `PATH` 및 `DOTNET_ROOT` 환경 변수를 설정합니다. `.bashrc` 파일에서 `export PATH` 섹션을 찾아 다음과 같이 표시되도록 편집합니다(이 파일 편집에 대한 자세한 내용은 1단계 참조).

```
export PATH=$PATH:$HOME/.local/bin:$HOME/bin:$HOME/.dotnet:$HOME/.dotnet/tools
export DOTNET_ROOT=$HOME/.dotnet
```

3단계: .NET 콘솔 애플리케이션 프로젝트 만들기

이 단계에서는 .NET을 사용하여 `hello` 프로젝트를 생성합니다. 이 프로젝트에는 IDE의 터미널에서 단순 애플리케이션을 실행하기 위해 .NET에 필요한 모든 파일이 포함되어 있습니다. 애플리케이션의 코드는 C#으로 작성되었습니다.

.NET 콘솔 애플리케이션 프로젝트를 생성합니다. 이렇게 하려면 사용할 콘솔 애플리케이션 프로젝트 템플릿 유형 및 프로그래밍 언어(이 샘플에서는 C#)를 지정하여 **new** 명령과 함께 .NET CLI를 실행합니다.

`-n` 옵션은 프로젝트가 `hello`라는 새 디렉터리로 출력됨을 나타냅니다. 그런 다음 해당 디렉터리로 이동합니다.

```
dotnet new console -lang C# -n hello
cd hello
```

이전 명령은 여러 파일을 포함하는 obj 하위 디렉터리와 추가 독립 실행형 파일을 hello 디렉터리에 추가합니다. 다음 키 파일 두 개에 유의해야 합니다.

- hello/hello.csproj 파일에는 콘솔 애플리케이션 프로젝트에 대한 정보가 들어 있습니다.
- hello/Program.cs 파일에는 실행할 애플리케이션의 코드가 들어 있습니다.

4단계: 코드 추가

이 단계에서는 애플리케이션에 코드를 추가합니다.

AWS Cloud9 IDE의 [환경(Environment)] 창에서 hello/Program.cs 파일을 엽니다.

편집기에서 이 파일의 현재 내용을 다음 코드로 바꾼 다음 Program.cs 파일을 저장합니다.

```
using System;

namespace hello
{
  class Program
  {
    static void Main(string[] args)
    {
     if (args.Length < 2) {
       Console.WriteLine("Please provide 2 numbers");
       return;
     }

     Console.WriteLine("Hello, World!");

     Console.WriteLine("The sum of 2 and 3 is 5.");

     int sum = Int32.Parse(args[0]) + Int32.Parse(args[1]);

     Console.WriteLine("The sum of {0} and {1} is {2}.",
     args[0], args[1], sum);

    }
  }
```

```
}
```

5단계: 코드 빌드 및 실행

이 단계에서는 프로젝트 및 관련 종속 항목을 바이너리 파일 세트(실행 가능한 애플리케이션 파일 포함)로 빌드합니다. 그런 다음 애플리케이션을 실행합니다.

1. IDE에서 다음과 같이 .NET용 빌더를 생성합니다.

   a. 메뉴 모음에서 Run, Build System, New Build System(실행, 빌드 시스템, 새 빌드 시스템)을 선택합니다.

   b. My Builder.build 탭에서 탭의 내용을 다음 코드로 바꿉니다.

   ```
   {
     "cmd" : ["dotnet", "build"],
     "info" : "Building..."
   }
   ```

   c. File, Save As(파일, 다른 이름으로 저장)를 선택합니다.

   d. Filename(파일 이름)에 `.NET.build`를 입력합니다.

   e. Folder(폴더)에 `/.c9/builders`를 입력합니다.

   f. Save를 선택합니다.

2. 편집기에 `Program.cs` 파일의 내용이 표시된 상태로 메뉴 모음에서 실행, 빌드 시스템, .NET을 선택합니다. 그런 다음 Run, Build(실행, 빌드)를 선택합니다.

   이 빌더는 `bin` 하위 디렉터리를 추가하고 `Debug` 하위 디렉터리를 `hello/obj` 하위 디렉터리에 추가합니다. 다음 키 파일 세 개에 유의하십시오.

   - `hello/bin/Debug/netcoreapp3.1/hello.dll` 파일은 실행 가능한 애플리케이션 파일입니다.
   - `hello/bin/Debug/netcoreapp3.1/hello.deps.json` 파일은 애플리케이션의 종속 항목을 나열합니다.
   - `hello/bin/Debug/netcoreapp3.1/hello.runtimeconfig.json` 파일은 애플리케이션의 버전과 공유 실행 시간을 지정합니다.

> **Note**
>
> 폴더 이름 `netcoreapp3.1`은 이 예에 사용된 .NET SDK의 버전을 나타냅니다. 설치한 버전에 따라 폴더 이름에 다른 번호가 표시될 수 있습니다.

3. 다음과 같이 .NET용 실행기를 생성합니다.

   a. 메뉴 모음에서 Run, Run With, New Runner(실행, 실행 도구, 새 실행기)를 선택합니다.

   b. My Runner.run 탭에서 탭의 내용을 다음 코드로 바꿉니다.

   ```
   {
     "cmd" : ["dotnet", "run", "$args"],
     "working_dir": "$file_path",
     "info" : "Running..."
   }
   ```

   c. File, Save As(파일, 다른 이름으로 저장)를 선택합니다.

   d. Filename(파일 이름)에 `.NET.run`를 입력합니다.

   e. Folder(폴더)에 `/.c9/runners`를 입력합니다.

   f. Save를 선택합니다.

4. 다음과 같이 추가할 두 정수(예: 5와 9)를 지정하여 애플리케이션을 실행합니다.

   a. 편집기에 `Program.cs` 파일의 내용이 표시된 상태로, Run(실행), Run Configurations(실행 구성), New Run Configuration(새로운 실행 구성)을 선택합니다.

   b. [새로 만들기] - 유휴 탭에서 실행기: 자동을 선택한 다음 .NET을 선택합니다.

   c. Command(명령) 상자에 `hello 5 9`를 입력합니다.

   d. 실행을 선택합니다.

   기본적으로 이 실행기는 .NET에 `hello/bin/Debug/netcoreapp3.1` 디렉터리의 `hello.dll` 파일을 실행하도록 지시합니다.

   출력을 다음과 비교합니다.

   ```
   Hello, World!
   The sum of 2 and 3 is 5.
   The sum of 5 and 9 is 14.
   ```

6단계: AWS SDK for .NET을 사용하는 .NET 콘솔 애플리케이션 프로젝트 생성 및 설정

이 샘플을 응용해 AWS SDK for .NET을 사용하여 Amazon S3 버킷을 생성하고 사용 가능한 버킷을 나열한 다음 방금 생성한 버킷을 삭제합니다.

이 새 프로젝트에서는 AWS SDK for .NET에 대한 참조를 추가합니다. AWS SDK for .NET은 .NET 코드에서 Amazon S3와 같은 AWS 서비스와 상호 작용할 수 있는 편리한 방법을 제공합니다. 그런 다음 환경에서 AWS 자격 증명 관리를 설정할 수 있습니다. AWS SDK for .NET는 AWS 서비스와 상호 작용하기 위해서 이런 자격 증명이 필요합니다.

프로젝트를 만들려면

1. .NET 콘솔 애플리케이션 프로젝트를 생성합니다. 이렇게 하려면 사용할 콘솔 애플리케이션 프로젝트 템플릿 유형 및 프로그래밍 언어를 지정하여 **new** 명령과 함께 .NET CLI를 실행합니다.

   `-n` 옵션은 프로젝트가 `s3`라는 새 디렉터리로 출력됨을 나타냅니다. 그런 다음 해당 디렉터리로 이동합니다.

   ```
   dotnet new console -lang C# -n s3
   cd s3
   ```

2. AWS SDK for .NET에 Amazon S3 패키지에 대한 프로젝트 참조를 추가합니다. 이렇게 하려면, NuGet의 Amazon S3 패키지 이름을 지정하여 **add package** 명령으로 .NET CLI를 실행합니다. (NuGet은 .NET용 패키지의 생성, 호스팅 및 사용 방식을 정의하고 이러한 각 역할에 사용할 도구를 제공합니다.)

   ```
   dotnet add package AWSSDK.S3
   ```

   Amazon S3 패키지에 대한 프로젝트 참조를 추가하면 NuGet에서 AWS SDK for .NET의 나머지에 대한 프로젝트 참조도 추가합니다.

   > **Note**
   >
   > NuGet의 다른 AWS 관련 패키지의 이름 및 버전의 경우, NuGet 웹 사이트의 NuGet packages tagged with aws-sdk를 참조하십시오.

AWS 자격 증명 관리를 설정하는 방법

AWS SDK for .NET를 사용하여 AWS 서비스를 호출할 때마다 호출과 함께 AWS 자격 증명 세트를 제공합니다. 이러한 자격 증명은 AWS SDK for .NET가 해당 호출을 할 수 있는 적절한 권한이 있는지 여부를 결정합니다. 자격 증명으로 적절한 권한이 확인되지 않는 경우 호출이 실패합니다.

환경에 자격 증명을 저장하려면 AWS Cloud9의 환경에서 AWS 서비스 호출의 지침을 따른 다음 이 주제로 돌아옵니다.

자세한 내용은 AWS SDK for .NET 개발자 가이드에서 AWS 자격 증명 구성을 참조하세요.

7단계: AWS SDK 코드 추가

이 단계에서는 Amazon S3와 상호 작용하는 코드를 추가하여 버킷을 생성하고 생성된 버킷을 삭제한 다음 사용 가능한 버킷의 목록을 나열합니다.

AWS Cloud9 IDE의 [환경(Environment)] 창에서 `s3/Program.cs` 파일을 엽니다. 편집기에서 이 파일의 현재 내용을 다음 코드로 바꾼 다음 `Program.cs` 파일을 저장합니다.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.S3.Util;
using System;
using System.Threading.Tasks;

namespace s3
{
  class Program
  {
   async static Task Main(string[] args)
   {
    if (args.Length < 2) {
      Console.WriteLine("Usage: <the bucket name> <the AWS Region to use>");
      Console.WriteLine("Example: my-test-bucket us-east-2");
      return;
    }

    if (args[1] != "us-east-2") {
      Console.WriteLine("Cannot continue. The only supported AWS Region ID is " +
      "'us-east-2'.");
       return;
     }
```

```
var bucketRegion = RegionEndpoint.USEast2;
// Note: You could add more valid AWS Regions above as needed.

using (var s3Client = new AmazonS3Client(bucketRegion)) {
var bucketName = args[0];

// Create the bucket.
try
{
 if (await AmazonS3Util.DoesS3BucketExistV2Async(s3Client, bucketName))
 {
   Console.WriteLine("Cannot continue. Cannot create bucket. \n" +
   "A bucket named '{0}' already exists.", bucketName);
   return;
 } else {
   Console.WriteLine("\nCreating the bucket named '{0}'...", bucketName);
   await s3Client.PutBucketAsync(bucketName);
   }
 }
 catch (AmazonS3Exception e)
 {
  Console.WriteLine("Cannot continue. {0}", e.Message);
 }
 catch (Exception e)
 {
  Console.WriteLine("Cannot continue. {0}", e.Message);
 }

 // Confirm that the bucket was created.
 if (await AmazonS3Util.DoesS3BucketExistV2Async(s3Client, bucketName))
 {
    Console.WriteLine("Created the bucket named '{0}'.", bucketName);
 } else {
   Console.WriteLine("Did not create the bucket named '{0}'.", bucketName);
 }

 // Delete the bucket.
 Console.WriteLine("\nDeleting the bucket named '{0}'...", bucketName);
 await s3Client.DeleteBucketAsync(bucketName);

 // Confirm that the bucket was deleted.
 if (await AmazonS3Util.DoesS3BucketExistV2Async(s3Client, bucketName))
 {
```

```
           Console.WriteLine("Did not delete the bucket named '{0}'.", bucketName);
        } else {
          Console.WriteLine("Deleted the bucket named '{0}'.", bucketName);
        };

         // List current buckets.
        Console.WriteLine("\nMy buckets now are:");
        var response = await s3Client.ListBucketsAsync();

        foreach (var bucket in response.Buckets)
        {
        Console.WriteLine(bucket.BucketName);
        }
       }
     }
   }
}
```

8단계: AWS SDK 코드 빌드 및 실행

이 단계에서는 프로젝트 및 관련 종속 항목을 바이너리 파일 세트(실행 가능한 애플리케이션 파일 포함)로 빌드합니다. 그런 다음 애플리케이션을 실행합니다.

1. 프로젝트를 빌드합니다. 이렇게 하려면 편집기에 `s3/Program.cs`파일의 내용이 표시된 상태로 메뉴 모음에서 Run, Build(실행, 빌드)를 선택합니다.
2. 다음과 같이 생성할 Amazon S3 버킷 이름과 및 버킷을 생성할 AWS 리전의 ID를 지정하여 애플리케이션(예: `my-test-bucket` 및 `us-east-2`)을 실행합니다.

   a. 편집기에 `s3/Program.cs` 파일의 내용이 여전히 표시된 상태로, Run(실행), Run Configurations(실행 구성), New Run Configuration(새로운 실행 구성)을 선택합니다.

   b. [새로 만들기] - 유휴 탭에서 실행기: 자동을 선택한 다음 .NET을 선택합니다.

   c. [명령(Command)] 상자에서 애플리케이션의 이름, 생성할 Amazon S3 버킷의 이름, 그리고 버킷을 생성할 AWS 리전의 ID를 입력합니다(예: `s3 my-test-bucket us-east-2`).

   d. 실행을 선택합니다.

      기본적으로 이 실행기는 .NET에 `s3/bin/Debug/netcoreapp3.1` 디렉터리의 `s3.dll` 파일을 실행하도록 지시합니다.

      결과를 다음 출력과 비교합니다.

```
Creating a new bucket named 'my-test-bucket'...
Created the bucket named 'my-test-bucket'.

Deleting the bucket named 'my-test-bucket'...
Deleted the bucket named 'my-test-bucket'.

My buckets now are:
```

9단계: 정리

이 샘플 사용 후 AWS 계정에 계속하여 요금이 부과되지 않도록 환경을 삭제해야 합니다. 지침은 AWS Cloud9에서 환경 삭제 단원을 참조하세요.

Node.js 튜토리얼 AWS Cloud9

이 자습서에서는 AWS Cloud9 개발 환경에서 일부 Node.js 스크립트를 실행할 수 있습니다.

이 자습서를 따라 이 샘플을 만들면 AWS 계정에 요금이 청구될 수 있습니다. 여기에는 Amazon EC2 및 Amazon S3 같은 서비스에 대한 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금 및 Amazon S3 요금을 참조하세요.

주제

필수 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는 운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 설명은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.
- 기존 환경용 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열면 웹 브라우저에서 해당 환경의 IDE가 AWS Cloud9 열립니다. 자세한 설명은 AWS Cloud9에서 환경 열기 섹션을 참조하세요.

1단계: 필수 도구 설치

이 단계에서는 이 샘플을 실행할 때 필요한 Node.js를 설치합니다.

1. AWS Cloud9 IDE의 터미널 세션에서 **`node --version`**명령을 실행하여 Node.js 가 이미 설치되어 있는지 확인합니다. (터미널 세션을 새로 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다.) 성공할 경우, 출력에 Node.js 버전 번호가 포함됩니다. Node.js가 설치된 경우 2단계: 코드 추가 섹션으로 이동합니다.
2. **`yum update`** 명령(Amazon Linux) 또는 **`apt update`** 명령(Ubuntu Server)을 실행하여 최신 보안 업데이트와 버그 수정이 설치되어 있는지 확인합니다.

   Amazon Linux의 경우:

   ```
   sudo yum -y update
   ```

   Ubuntu Server:

   ```
   sudo apt update
   ```

3. Node.js 설치하려면 먼저 이 명령을 실행하여 노드 버전 관리자 (nvm) 를 다운로드합니다. (nvm은 Node.js 버전을 설치하고 관리하는 데 유용한 간단한 Bash 셸 스크립트입니다. 자세한 내용은 GitHub 웹 사이트의 노드 버전 관리자를 참조하십시오.

   ```
   curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.5/install.sh | bash
   ```

4. nvm 사용을 시작하려면 터미널 세션을 닫고 나서 다시 시작하거나, nvm을 로드하는 명령을 포함하는 `~/.bashrc` 파일을 소싱합니다.

   ```
   . ~/.bashrc
   ```

5. 이 명령을 실행하여 Amazon Linux 2, Amazon Linux 1, Ubuntu 18.04.에 Node.js 16을 설치합니다. Amazon Linux 1 및 Ubuntu 18.04 인스턴스는 Node.js v16까지만 지원합니다.

```
nvm install 16
```

이 명령을 실행하여 Amazon Linux 2023 및 Ubuntu 22.04에 Node.js의 최신 버전을 설치합니다.

```
nvm install --lts && nvm alias default lts/*
```

> **Note**
>
> 최신 AL2023 AWS Cloud9 이미지에는 Node.js 20이 설치되어 있고 최신 아마존 리눅스 2 AWS Cloud9 이미지에는 Node.js 18이 설치되어 있습니다. Amazon Linux 2에 Node.js 18을 AWS Cloud9 수동으로 설치하려면 AWS Cloud9 IDE 터미널에서 다음 명령을 실행합니다.
>
> ```
> C9_NODE_INSTALL_DIR=~/.nvm/versions/node/v18.17.1
> C9_NODE_URL=https://d3kgj69l4ph6w4.cloudfront.net/static/node-amazon/node-
> v18.17.1-linux-x64.tar.gz
> mkdir -p $C9_NODE_INSTALL_DIR
> curl -fSsl $C9_NODE_URL  | tar xz --strip-components=1 -C
>  "$C9_NODE_INSTALL_DIR"
> nvm alias default v18.17.1
> nvm use default
> echo -e 'nvm use default' >> ~/.bash_profile
> ```

2단계: 코드 추가

AWS Cloud9 IDE에서 이 내용으로 파일을 만들고 해당 이름을 사용하여 파일을 저장합니다`hello.js`. (파일을 생성하려면 메뉴 모음에서 [파일(File)], [새 파일(New File)]을 선택합니다. 파일을 저장하려면 [파일(File)], [저장(Save)]을 선택합니다.)

```
console.log('Hello, World!');

console.log('The sum of 2 and 3 is 5.');

var sum = parseInt(process.argv[2], 10) + parseInt(process.argv[3], 10);
```

```
console.log('The sum of ' + process.argv[2] + ' and ' +
  process.argv[3] + ' is ' + sum + '.');
```

3단계: 코드 실행

1. AWS Cloud9 IDE의 메뉴 막대에서 [실행], [구성 실행], [새 실행 구성] 을 선택합니다.
2. [[새로 만들기] - 유휴([New] - Idle)] 탭에서 [러너: 자동(Runner: Auto)]을 선택한 다음 [Node.js]를 선택합니다.
3. 명령에 `hello.js 5 9`를 입력합니다. 코드에서 5는 `process.argv[2]`를 나타내고 9는 `process.argv[3]`를 나타냅니다. (`process.argv[0]`는 런타임의 이름(node)을 나타내고 `process.argv[1]`는 파일의 이름(`hello.js`)을 나타냅니다.)
4. Run(실행) 버튼을 선택하여 출력값을 비교합니다.

```
Hello, World!
The sum of 2 and 3 is 5.
The sum of 5 and 9 is 14.
```

4단계: Node.js 버전용 AWS JavaScript SDK 설치 및 구성

에서 AWS Cloud9 Node.js 스크립트를 실행할 때 버전 3 (V3) 용 AWS SDK와 JavaScript 버전 2 (V2) 용 JavaScript 이전 AWS SDK 중에서 선택할 수 있습니다. V2와 마찬가지로 V3도 Amazon Web Services를 사용하여 쉽게 작업할 수 있지만 모듈화된 TypeScript 패키지와 같이 자주 요청되는 몇 가지 기능이 추가되어 있습니다.

AWS SDK for JavaScript (V3)

이 샘플을 개선하여 Node.js 용 AWS JavaScript SDK를 사용하여 Amazon S3 버킷을 생성하고, 사용 가능한 버킷을 나열한 다음, 방금 생성한 버킷을 삭제할 수 있습니다.

이 단계에서는 JavaScript 코드에서 Amazon S3 서비스와 상호 작용하는 편리한 방법을 제공하는 Node.js 용 AWS JavaScript SDK의 Amazon S3 AWS 서비스 클라이언트 모듈을 설치하고 구성합니다.

다른 AWS 서비스를 사용하려면 해당 서비스를 별도로 설치해야 합니다. AWS 모듈 설치에 대한 자세한 내용은 AWS 개발자 안내서 (V3) 를 참조하십시오. (V3) 용 Node.js 및 AWS SDK를 시작하는 방법에 대한 자세한 내용은 JavaScript개발자용 SDK 가이드 JavaScript (V3) 의 AWS Node.js 시작하기를 참조하십시오.

Node.js JavaScript 에서 AWS SDK를 설치한 후에는 해당 환경에서 자격 증명 관리를 설정해야 합니다. Node.js 용 AWS SDK는 서비스와 JavaScript 상호 AWS 작용하기 위해 이러한 자격 증명이 필요합니다.

Node.js 형식으로 AWS SDK를 JavaScript 설치하려면

npm을 사용하여 **`install`** 명령을 실행합니다.

```
npm install @aws-sdk/client-s3
```

자세한 내용은 JavaScript AWS SDK for JavaScript 개발자 안내서의 SDK 설치를 참조하십시오.

환경에서 보안 인증 관리를 설정하려면

Node.js 용 AWS JavaScript SDK를 사용하여 AWS 서비스를 호출할 때마다 호출과 함께 자격 증명 세트를 제공해야 합니다. 이러한 자격 증명은 Node.js 용 AWS JavaScript SDK에 해당 호출을 수행할 수 있는 적절한 권한이 있는지 여부를 결정합니다. 자격 증명으로 적절한 권한이 확인되지 않는 경우 호출이 실패합니다.

이 단계에서는 환경 내에서 자격 증명을 저장합니다. 이렇게 하려면 AWS Cloud9의 환경에서 AWS 서비스 호출 섹션의 지침을 수행한 다음 이 주제로 돌아옵니다.

자세한 내용은 AWS SDK for JavaScript 개발자 가이드의 Node.js에서 자격 증명 설정을 참조하세요.

AWS SDK for JavaScript (V2)

이 샘플을 개선하여 Node.js 용 AWS JavaScript SDK를 사용하여 Amazon S3 버킷을 생성하고, 사용 가능한 버킷을 나열한 다음, 방금 생성한 버킷을 삭제할 수 있습니다.

이 단계에서는 JavaScript 코드에서 Amazon S3와 같은 AWS 서비스와 상호 작용할 수 있는 편리한 방법을 제공하는 Node.js 용 AWS JavaScript SDK를 설치하고 구성합니다. Node.js for에 AWS SDK를 JavaScript 설치한 후에는 해당 환경에서 자격 증명 관리를 설정해야 합니다. Node.js 용 AWS SDK는 서비스와 JavaScript 상호 AWS 작용하기 위해 이러한 자격 증명이 필요합니다.

Node.js 형식으로 AWS SDK를 JavaScript 설치하려면

npm을 사용하여 **`install`** 명령을 실행합니다.

```
npm install aws-sdk
```

자세한 내용은 JavaScript AWS SDK for JavaScript 개발자 안내서의 SDK 설치를 참조하십시오.

환경에서 보안 인증 관리를 설정하려면

Node.js 용 AWS JavaScript SDK를 사용하여 AWS 서비스를 호출할 때마다 호출과 함께 자격 증명 세트를 제공해야 합니다. 이러한 자격 증명은 Node.js 용 AWS JavaScript SDK에 해당 호출을 수행할 수 있는 적절한 권한이 있는지 여부를 결정합니다. 자격 증명으로 적절한 권한이 확인되지 않는 경우 호출이 실패합니다.

이 단계에서는 환경 내에서 자격 증명을 저장합니다. 이렇게 하려면 AWS Cloud9의 환경에서 AWS 서비스 호출 섹션의 지침을 수행한 다음 이 주제로 돌아옵니다.

자세한 내용은 AWS SDK for JavaScript 개발자 가이드의 Node.js에서 자격 증명 설정을 참조하세요.

5단계: AWS SDK 코드 추가

AWS SDK for JavaScript (V3)

이 단계에서는 몇 가지 코드를 더 추가합니다. 이번 경우에는 Amazon S3와 상호 작용하기 위한 것이며 버킷을 생성하고 사용 가능한 버킷을 나열한 다음 막 생성한 버킷을 삭제합니다. 이 코드를 나중에 실행합니다.

AWS Cloud9 IDE에서 이 내용으로 파일을 만들고 이름을 `s3.js` 사용하여 파일을 저장합니다.

```
import {
  CreateBucketCommand,
  DeleteBucketCommand,
  ListBucketsCommand,
  S3Client,
} from "@aws-sdk/client-s3";

const wait = async (milliseconds) => {
  return new Promise((resolve) => setTimeout(resolve, milliseconds));
};

export const main = async () => {
  const client = new S3Client({});
  const now = Date.now();
  const BUCKET_NAME = `easy-bucket-${now.toString()}`;

  const createBucketCommand = new CreateBucketCommand({ Bucket: BUCKET_NAME });
  const listBucketsCommand = new ListBucketsCommand({});
  const deleteBucketCommand = new DeleteBucketCommand({ Bucket: BUCKET_NAME });

  try {
    console.log(`Creating bucket ${BUCKET_NAME}.`);
    await client.send(createBucketCommand);
    console.log(`${BUCKET_NAME} created`);

    await wait(2000);

    console.log(`Here are your buckets:`);
    const { Buckets } = await client.send(listBucketsCommand);
    Buckets.forEach((bucket) => {
      console.log(` • ${bucket.Name}`);
    });

    await wait(2000);

    console.log(`Deleting bucket ${BUCKET_NAME}.`);
    await client.send(deleteBucketCommand);
    console.log(`${BUCKET_NAME} deleted`);
  } catch (err) {
    console.error(err);
  }
};
```

```
main();
```

AWS SDK for JavaScript (V2)

이 단계에서는 몇 가지 코드를 더 추가합니다. 이번 경우에는 Amazon S3와 상호 작용하기 위한 것이며 버킷을 생성하고 사용 가능한 버킷을 나열한 다음 막 생성한 버킷을 삭제합니다. 이 코드를 나중에 실행합니다.

AWS Cloud9 IDE에서 이 내용으로 파일을 만들고 이름을 사용하여 파일을 저장합니다`s3.js`.

```
if (process.argv.length < 4) {
  console.log(
    "Usage: node s3.js <the bucket name> <the AWS Region to use>\n" +
      "Example: node s3.js my-test-bucket us-east-2"
  );
  process.exit(1);
}

var AWS = require("aws-sdk"); // To set the AWS credentials and region.
var async = require("async"); // To call AWS operations asynchronously.

AWS.config.update({
  region: region,
});

var s3 = new AWS.S3({ apiVersion: "2006-03-01" });
var bucket_name = process.argv[2];
var region = process.argv[3];

var create_bucket_params = {
  Bucket: bucket_name,
  CreateBucketConfiguration: {
    LocationConstraint: region,
  },
};

var delete_bucket_params = { Bucket: bucket_name };

// List all of your available buckets in this AWS Region.
function listMyBuckets(callback) {
```

```
  s3.listBuckets(function (err, data) {
    if (err) {
    } else {
      console.log("My buckets now are:\n");

      for (var i = 0; i < data.Buckets.length; i++) {
        console.log(data.Buckets[i].Name);
      }
    }

    callback(err);
  });
}

// Create a bucket in this AWS Region.
function createMyBucket(callback) {
  console.log("\nCreating a bucket named " + bucket_name + "...\n");

  s3.createBucket(create_bucket_params, function (err, data) {
    if (err) {
      console.log(err.code + ": " + err.message);
    }

    callback(err);
  });
}

// Delete the bucket you just created.
function deleteMyBucket(callback) {
  console.log("\nDeleting the bucket named " + bucket_name + "...\n");

  s3.deleteBucket(delete_bucket_params, function (err, data) {
    if (err) {
      console.log(err.code + ": " + err.message);
    }

    callback(err);
  });
}

// Call the AWS operations in the following order.
async.series([
  listMyBuckets,
  createMyBucket,
```

```
  listMyBuckets,
  deleteMyBucket,
  listMyBuckets,
]);
```

6단계: AWS SDK 코드 실행

1. 코드에서 npm을 사용하여 **`install`** 명령을 실행함으로써 Amazon S3 작업을 비동기로 호출할 수 있게 합니다.

   ```
   npm install async
   ```

2. AWS Cloud9 IDE의 메뉴 막대에서 실행, 구성 실행, 새 실행 구성을 선택합니다.
3. [[새로 만들기] - 유휴([New] - Idle)] 탭에서 [러너: 자동(Runner: Auto)]을 선택한 다음 [Node.js]를 선택합니다.
4. JavaScript (V3) 용 AWS SDK를 사용하는 경우 명령 유형을 선택합니다. `s3.js` Javascript용 AWS SDK (v2) 를 사용하는 경우`s3.js my-test-bucket us-east-2`, 명령 유형의 경우 여기서 `my-test-bucket` 는 만들고 삭제하려는 버킷의 이름이고, `us-east-2` 는 버킷을 만들려는 AWS 지역의 ID입니다. 더 많은 ID는 Amazon Web Services 일반 참조의 Amazon Simple Storage Service(Amazon S3)를 참조하세요.

   > **Note**
   >
   > Amazon S3 버킷 이름은 계정뿐만 AWS아니라 전체 AWS 계정에서 고유해야 합니다.

5. Run(실행) 버튼을 선택하여 출력값을 비교합니다.

   ```
   My buckets now are:

   Creating a new bucket named 'my-test-bucket'...

   My buckets now are:

   my-test-bucket

   Deleting the bucket named 'my-test-bucket'...

   My buckets now are:
   ```

7단계: 정리

이 샘플 사용을 완료한 후 AWS 계정에 계속 요금이 청구되는 것을 방지하려면 환경을 삭제해야 합니다. 지침은 AWS Cloud9에서 환경 삭제 단원을 참조하세요.

에 대한 PHP 튜토리얼 AWS Cloud9

이 자습서에서는 AWS Cloud9 개발 환경에서 일부 PHP 스크립트를 실행할 수 있습니다.

이 자습서를 따르고 이 샘플을 생성하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2 및 Amazon S3 같은 서비스에 대한 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금 및 Amazon S3 요금을 참조하세요.

주제

필수 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는 운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 정보는 에서 환경 만들기 AWS Cloud9을 참조하세요.
- 기존 환경용 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열면 웹 브라우저에서 해당 환경의 IDE가 AWS Cloud9 열립니다. 자세한 정보는 AWS Cloud9에서 환경 열기을 참조하세요.

1단계: 필수 도구 설치

이 단계에서는 이 샘플을 실행할 때 필요한 PHP를 설치합니다.

> **Note**
>
> 다음 절차에서는 PHP만 설치합니다. Apache 웹 서버 및 MySQL 데이터베이스와 같은 관련 도구를 설치하려면 Amazon EC2 사용 설명서의 자습서: Amazon Linux에 LAMP 웹 서버 설치를 참조하십시오.

1. AWS Cloud9 IDE의 터미널 세션에서 명령을 실행하여 PHP가 이미 설치되어 있는지 확인합니다. **`php --version`** (터미널 세션을 새로 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다.) 성공할 경우, PHP 버전 숫자를 포함해 출력값이 생성됩니다. PHP가 설치된 경우 2단계: 코드 추가로 이동합니다.
2. **`yum update`** 명령(Amazon Linux) 또는 **`apt update`** 명령(Ubuntu Server)을 실행하여 최신 보안 업데이트와 버그 수정이 설치되어 있는지 확인합니다.

   Amazon Linux 2 및 Amazon Linux의 경우:

   ```
   sudo yum -y update
   ```

   Ubuntu Server:

   ```
   sudo apt update
   ```
3. **`install`** 명령을 실행하여 PHP를 설치합니다.

   대상 Amazon Linux 2:

   ```
   sudo amazon-linux-extras install -y php7.2
   ```

   Amazon Linux의 경우:

   ```
   sudo yum -y install php72
   ```

> **Note**
>
> 다음 명령을 사용하여 Amazon Linux 버전을 볼 수 있습니다.
>
> ```
> cat /etc/system-release
> ```

Ubuntu Server:

```
sudo apt install -y php php-xml
```

자세한 내용은 PHP 웹 사이트에서 설치 및 구성을 참조하세요.

2단계: 코드 추가

AWS Cloud9 IDE에서 이 내용으로 파일을 만들고 이름을 `hello.php` 사용하여 파일을 저장합니다. (파일을 생성하려면 메뉴 모음에서 [파일(File)], [새 파일]을 선택합니다. 파일을 저장하려면 [파일(File)], [저장(Save)]을 선택하고 [파일 이름(Filename)]에 `hello.php`을 입력한 다음 [저장(Save)]을 선택합니다.)

```
<?php
  print('Hello, World!');

  print("\nThe sum of 2 and 3 is 5.");

  $sum = (int)$argv[1] + (int)$argv[2];

  print("\nThe sum of $argv[1] and $argv[2] is $sum.");
?>
```

> **Note**
>
> 위의 코드는 외부 파일에 종속되지 않습니다. 그러나 파일에 다른 PHP 파일을 포함하거나 필요로 AWS Cloud9 하는 경우 입력하는 동안 해당 파일을 사용하여 코드를 완성하려면 환경설정에서 프로젝트, PHP 지원, PHP 코드 완성 활성화 설정을 켠 다음 해당 파일의 경로를 프로

젝트, PHP 지원, PHP 완료 경로 포함 설정에 추가하십시오. (기본 설정을 보고 변경하려면 메뉴 모음에서 AWS Cloud9, Preferences(기본 설정)를 선택합니다.)

3단계: 코드 실행

1. AWS Cloud9 IDE의 메뉴 표시줄에서 [실행], [구성 실행], [구성 새로 실행] 을 선택합니다.
2. [[새로 만들기] - 유휴([New] - Idle)] 탭에서 [러너: 자동(Runner: Auto)]을 선택한 다음 [PHP(cli)]를 선택합니다.
3. 명령에 `hello.php 5 9`를 입력합니다. 코드에서 5는 `$argv[1]`를 나타내고 9는 `$argv[2]`를 나타냅니다. (`$argv[0]`는 파일의 이름(`hello.php`)을 나타냅니다.)
4. Run(실행) 버튼을 선택하여 출력값을 비교합니다.

```
Hello, World!
The sum of 2 and 3 is 5.
The sum of 5 and 9 is 14.
```

4단계: 설치 및 구성 AWS SDK for PHP

이 샘플을 개선하여 를 사용하여 Amazon S3 버킷을 생성하고, 사용 가능한 버킷을 나열한 다음, 방금 생성한 버킷을 삭제할 수 있습니다. AWS SDK for PHP

이 단계에서는 PHP 코드에서 Amazon S3와 같은 AWS 서비스와 상호 작용하는 편리한 방법을 제공하는 를 설치하고 구성합니다. AWS SDK for PHP를 설치하려면 먼저 Composer를 설치해야 합니다.

AWS SDK for PHP를 설치한 후에는 사용자 환경에서 자격 증명 관리를 설정해야 합니다. AWS SDK for PHP AWS 서비스와 상호 작용하려면 이러한 자격 증명이 AWS SDK for PHP 필요합니다.

Composer를 설치하려면

자동(`-s`) 및 오류 표시(`-S`) 옵션을 사용하여 **`curl`** 명령을 실행하고, Composer 설치 관리자를 `composer.phar`(명명 규칙에 따른 이름)이라는 PHP 아카이브(PHAR) 파일로 파이핑합니다.

```
curl -sS https://getcomposer.org/installer | php
```

설치하려면 AWS SDK for PHP

Ubuntu Server의 경우, Composer가 AWS SDK for PHP를 설치하는 데 필요한 추가 패키지를 설치합니다.

```
sudo apt install -y php-xml php-curl
```

Amazon Linux 또는 Ubuntu Server의 경우 php 명령을 사용하여 Composer 설치 관리자를 실행하여 AWS SDK for PHP를 설치합니다.

```
php composer.phar require aws/aws-sdk-php
```

이 명령은 사용자 환경에 여러 폴더와 파일을 만듭니다. 사용할 주 파일은 `autoload.php`이고 환경의 `vendor` 폴더에 있습니다.

> **Note**
>
> 설치 후 Composer는 추가 종속 구성 요소를 설치하도록 제안할 수 있습니다. 설치할 종속 구성 요소 목록을 지정하고 다음과 같은 명령을 사용하여 아룰 수행할 수 있습니다. 예를 들어 다음 명령은 종속 구성 요소의 다음 목록을 설치하도록 Composer에 지시합니다.
>
> ```
> php composer.phar require psr/log ext-curl doctrine/cache aws/aws-php-sns-message-validator
> ```

자세한 내용은 AWS SDK for PHP 개발자 가이드에서 설치를 참조하세요.

환경에서 보안 인증 관리를 설정하려면

를 사용하여 AWS 서비스를 AWS SDK for PHP 호출할 때마다 호출과 함께 자격 증명 세트를 제공해야 합니다. 이러한 자격 증명은 해당 AWS SDK for PHP 호출을 수행할 수 있는 적절한 권한이 있는지 여부를 결정합니다. 자격 증명으로 적절한 권한이 확인되지 않는 경우 호출이 실패합니다.

이 단계에서는 환경 내에서 자격 증명을 저장합니다. 이렇게 하려면 AWS Cloud9의 환경에서 AWS 서비스 호출 섹션의 지침을 수행한 다음 이 주제로 돌아옵니다.

자세한 내용은 AWS SDK for PHP 개발자 가이드에서 기본 사용법의 '클라이언트 생성' 섹션을 참조하세요.

5단계: AWS SDK 코드 추가

이 단계에서는 몇 가지 코드를 더 추가합니다. 이번 경우에는 Amazon S3와 상호 작용하기 위한 것이며 버킷을 생성하고 사용 가능한 버킷을 나열한 다음 막 생성한 버킷을 삭제합니다. 이 코드를 나중에 실행합니다.

AWS Cloud9 IDE에서 이 내용으로 파일을 만들고 이름을 `s3.php` 사용하여 파일을 저장합니다.

```
<?php
require './vendor/autoload.php';

if ($argc < 4) {
    exit("Usage: php s3.php <the time zone> <the bucket name> <the AWS Region to use>
\n" .
        "Example: php s3.php America/Los_Angeles my-test-bucket us-east-2");
}

$timeZone = $argv[1];
$bucketName = $argv[2];
$region = $argv[3];

date_default_timezone_set($timeZone);

$s3 = new Aws\S3\S3Client([
    'region' => $region,
    'version' => '2006-03-01'
]);

# Lists all of your available buckets in this AWS Region.
function listMyBuckets($s3)
{
```

```
    print("\nMy buckets now are:\n");

    $promise = $s3->listBucketsAsync();

    $result = $promise->wait();

    foreach ($result['Buckets'] as $bucket) {
        print("\n");
        print($bucket['Name']);
    }
}

listMyBuckets($s3);

# Create a new bucket.
print("\n\nCreating a new bucket named '$bucketName'...\n");

try {
    $promise = $s3->createBucketAsync([
        'Bucket' => $bucketName,
        'CreateBucketConfiguration' => [
            'LocationConstraint' => $region
        ]
    ]);

    $promise->wait();
} catch (Exception $e) {
    if ($e->getCode() == 'BucketAlreadyExists') {
        exit("\nCannot create the bucket. " .
            "A bucket with the name '$bucketName' already exists. Exiting.");
    }
}

listMyBuckets($s3);

# Delete the bucket you just created.
print("\n\nDeleting the bucket named '$bucketName'...\n");

$promise = $s3->deleteBucketAsync([
    'Bucket' => $bucketName
]);

$promise->wait();
```

```
listMyBuckets($s3);

?>
```

6단계: AWS SDK 코드 실행

1. AWS Cloud9 IDE의 메뉴 막대에서 실행, 구성 실행, 새 실행 구성을 선택합니다.
2. [[새로 만들기] - 유휴([New] - Idle)] 탭에서 [러너: 자동(Runner: Auto)]을 선택한 다음 [PHP(cli)]를 선택합니다.
3. 명령에 `s3.php America/Los_Angeles my-test-bucket us-east-2`을 입력합니다.

   - `America/Los_Angeles`는 기본 시간대 ID입니다. 더 많은 ID를 보려면 PHP 웹 사이트에서 지원되는 시간대 목록을 참조하세요.
   - `my-test-bucket`은 생성한 다음 삭제하려는 버킷 이름입니다.

     > **Note**
     >
     > Amazon S3 버킷 이름은 AWS 계정뿐만 AWS아니라 전체에서 고유해야 합니다.

   - `us-east-2`버킷을 생성하려는 AWS 지역의 ID입니다. 더 많은 ID는 Amazon Web Services 일반 참조의 Amazon Simple Storage Service(Amazon S3)를 참조하세요.
4. Run(실행) 버튼을 선택하여 출력값을 비교합니다.

```
My buckets now are:

Creating a new bucket named 'my-test-bucket'...

My buckets now are:

my-test-bucket

Deleting the bucket named 'my-test-bucket'...

My buckets now are:
```

7단계: 정리

이 샘플 사용을 완료한 후 AWS 계정에 계속 요금이 청구되는 것을 방지하려면 환경을 삭제해야 합니다. 지침은 AWS Cloud9에서 환경 삭제을 참조하세요.

AWS Cloud9용 PHP 러너 관련 문제 해결 방법

PHP CLI 러너에서 문제가 발생하는 경우, 러너가 PHP로 설정되어 있고 디버거 모드가 활성화되어 있는지 확인해야 합니다.

AWS Cloud9의 Ruby

Ruby용 AWS SDK와 함께 AWS Cloud9을 사용하는 방법에 대한 자세한 내용은 Ruby용 AWSSDK 개발자 안내서의 Ruby용 AWS SDK와 함께 AWS Cloud9 사용을 참조하세요.

> **Note**
>
> 이 자습서를 이용하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2 및 Amazon S3 같은 서비스에 대한 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금 및 Amazon S3 요금을 참조하세요.

AWS Cloud9용 Go 자습서

이 자습서를 사용하면 AWS Cloud9 개발 환경에서 Go 코드를 실행할 수 있습니다.

이 자습서를 따르고 이 샘플을 생성하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2 및 Amazon S3 같은 서비스에 대한 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금 및 Amazon S3 요금을 참조하세요.

주제

사전 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는 운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 내용은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.
- 기존 환경에 대한 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열 때 AWS Cloud9은 웹 브라우저에서 환경을 위한 IDE를 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기 섹션을 참조하세요.

1단계: 필수 도구 설치

이 단계에서는 이 샘플을 실행할 때 필요한 Go를 설치하고 구성합니다.

1. AWS Cloud9 IDE의 터미널 세션에서 **`go version`** 명령을 실행하여 Go가 이미 설치되었는지 여부를 확인합니다. (터미널 세션을 새로 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다.) 성공할 경우, 출력에 Go 버전 번호가 포함됩니다. 그렇지 않으면 오류 메시지가 출력됩니다. Go가 설치된 경우 2단계: 코드 추가 섹션으로 건너뜁니다.
2. **`yum update`** 명령(Amazon Linux) 또는 **`apt update`** 명령(Ubuntu Server)을 실행하여 최신 보안 업데이트와 버그 수정이 설치되어 있는지 확인합니다.

   Amazon Linux의 경우:

   ```
   sudo yum -y update
   ```

   Ubuntu Server:

   ```
   sudo apt update
   ```
3. Go를 설치하려면 한 번에 하나씩 다음 명령을 실행합니다.

   ```
   wget https://storage.googleapis.com/golang/go1.9.3.linux-amd64.tar.gz # Download
    the Go installer.
   ```

```
sudo tar -C /usr/local -xzf ./go1.9.3.linux-amd64.tar.gz                # Install Go.
rm ./go1.9.3.linux-amd64.tar.gz                                         # Delete the
 installer.
```

앞의 명령은 이 주제가 작성된 당시의 안정된 최신 버전의 Go를 기준으로 합니다. 자세한 내용은 Go 프로그래밍 언어 웹 사이트에서 다운로드를 참조하세요.

4. 다음과 같이 Go 바이너리에 대한 경로를 PATH 환경 변수에 추가합니다.

   a. 편집을 위해 셸 프로파일 파일(예: ~/.bashrc)을 엽니다.

   b. 이 코드 줄 끝에 다음을 입력하여 코드가 다음과 같이 표시되도록 합니다.

   ```
   PATH=$PATH:/usr/local/go/bin
   ```

   c. 파일을 저장합니다.

5. 터미널이 방금 참조한 Go 바이너리를 찾을 수 있도록 ~/.bashrc 파일을 소스로 지정합니다.

   ```
   . ~/.bashrc
   ```

6. **go version** 명령을 실행하여 Go가 성공적으로 설치 및 구성되었는지 확인합니다. 성공할 경우, 출력에 Go 버전 번호가 포함됩니다.

2단계: 코드 추가

AWS Cloud9 IDE에서 이 콘텐츠를 사용해 파일을 생성하고 hello.go라는 이름으로 파일을 저장합니다. (파일을 생성하려면 메뉴 모음에서 [파일(File)], [새 파일(New File)]을 선택합니다. 파일을 저장하려면 [파일(File)], [저장(Save)]을 선택합니다.)

```
package main

import (
  "fmt"
  "os"
  "strconv"
)

func main() {
  fmt.Printf("Hello, World!\n")

  fmt.Printf("The sum of 2 and 3 is 5.\n")
```

```
  first, _ := strconv.Atoi(os.Args[1])
  second, _ := strconv.Atoi(os.Args[2])
  sum := first + second

  fmt.Printf("The sum of %s and %s is %s.",
    os.Args[1], os.Args[2], strconv.Itoa(sum))
}
```

3단계: 코드 실행

1. AWS Cloud9 IDE의 메뉴 모음에서 [실행(Run)], [실행 구성(Run Configurations)] 및 [새 실행 구성(New Run Configuration)]을 선택합니다.
2. [[새로 만들기] - 유휴([New] - Idle)] 탭에서 [러너: 자동(Runner: Auto)]을 선택한 다음 [Go]를 선택합니다.

   > **Note**
   >
   > Go를 사용할 수 없는 경우 Go용 사용자 지정 러너를 생성할 수 있습니다.
   >
   > 1. [New] - Idle([새로 만들기] = 유휴) 탭에서 Runner: Auto(실행기: 자동)를 선택하고 나서 New Runner(새 실행기)를 선택합니다.
   > 2. My Runner.run(내 실행기.run) 탭에서 탭의 내용을 이 코드로 바꿉니다.
   >
   >    ```
   >    {
   >      "cmd" : ["go", "run", "$file", "$args"],
   >      "info" : "Running $project_path$file_name...",
   >      "selector" : "source.go"
   >    }
   >    ```
   >
   > 3. 메뉴 모음에서 File(파일), Save As(다른 이름으로 저장)를 선택한 후 파일을 `/.c9/runners` 폴더의 `Go.run`으로 저장합니다.
   > 4. [[새로 만들기] - 유휴([New] - Idle)] 탭에서 [러너: 자동(Runner: Auto)]을 선택한 다음 [Go]를 선택합니다.
   > 5. hello.go 탭을 선택하여 활성화합니다.

3. 명령에 `hello.go 5 9`를 입력합니다. 이 코드에서 `5`는 `os.Args[1]`을 나타내고 `9`는 `os.Args[2]`를 나타냅니다.

4. Run(실행) 버튼을 선택하여 출력값을 비교합니다.

```
Hello, World!
The sum of 2 and 3 is 5.
The sum of 5 and 9 is 14.
```

4단계: AWS SDK for Go 설치 및 구성

이 샘플을 응용해 AWS SDK for Go을 사용하여 Amazon S3 버킷을 생성하고 사용 가능한 버킷을 나열한 다음 방금 생성한 버킷을 삭제합니다.

이 단계에서는 Go 코드에서 Amazon S3와 같은 AWS 서비스와 상호 작용할 수 있는 편리한 방법을 제공하는 AWS SDK for Go를 설치 및 구성합니다. AWS SDK for Go를 설치하기 전에 GOPATH 환경 변수를 설정해야 합니다. AWS SDK for Go를 설치하고 GOPATH 환경 변수를 설정한 후에는 환경에서 자격 증명 관리를 설정해야 합니다. AWS SDK for Go는 AWS 서비스와 상호 작용하기 위해서 이런 자격 증명이 필요합니다.

GOPATH 환경 변수를 설정하려면

1. 편집을 위해 ~/.bashrc 파일을 엽니다.
2. 파일의 마지막 줄 뒤에 이 코드를 입력합니다.

```
GOPATH=~/environment/go

export GOPATH
```

3. 파일을 저장합니다.
4. 터미널이 방금 참조한 GOPATH 환경 변수를 찾을 수 있도록 ~/.bashrc 파일을 소스로 지정합니다.

```
. ~/.bashrc
```

5. **echo $GOPATH** 명령을 실행하여 GOPATH 환경 변수가 성공적으로 설정되었는지 확인합니다. 성공적으로 설정된 경우 /home/ec2-user/environment/go 또는 /home/ubuntu/environment/go가 출력되어야 합니다.

AWS SDK for Go를 설치하려면

AWS SDK for Go소스의 위치를 지정하여 **go get** 명령을 실행합니다.

```
go get -u github.com/aws/aws-sdk-go/...
```

Go는 GOPATH 환경 변수에 의해 지정된 위치, 즉 환경의 go폴더에 AWS SDK for Go 소스를 설치합니다.

환경에서 자격 증명 관리를 설정하려면

AWS SDK for Go을 사용하여 AWS 서비스를 호출할 때마다 호출과 함께 자격 증명 세트를 제공해야 합니다. 이러한 자격 증명은 AWS SDK for Go가 해당 호출을 할 수 있는 적절한 권한이 있는지 여부를 결정합니다. 자격 증명으로 적절한 권한이 확인되지 않는 경우 호출이 실패합니다.

이 단계에서는 환경 내에서 자격 증명을 저장합니다. 이렇게 하려면 AWS Cloud9의 환경에서 AWS 서비스 호출 섹션의 지침을 수행한 다음 이 주제로 돌아옵니다.

자세한 내용은 AWS SDK for Go 개발자 가이드에서 자격 증명 지정을 참조하세요.

5단계: AWS SDK 코드 추가

이 단계에서는 몇 가지 코드를 더 추가합니다. 이번 경우에는 Amazon S3와 상호 작용하기 위한 것이며 버킷을 생성하고 사용 가능한 버킷을 나열한 다음 막 생성한 버킷을 삭제합니다. 이 코드를 나중에 실행합니다.

AWS Cloud9 IDE에서 이 콘텐츠를 사용해 파일을 생성하고 `s3.go`라는 이름으로 파일을 저장합니다.

```
package main

import (
 "fmt"
 "os"

 "github.com/aws/aws-sdk-go/aws"
 "github.com/aws/aws-sdk-go/aws/session"
 "github.com/aws/aws-sdk-go/service/s3"
)

func main() {

 if len(os.Args) < 3 {
  fmt.Printf("Usage: go run s3.go <the bucket name> <the AWS Region to use>\n" +
   "Example: go run s3.go my-test-bucket us-east-2\n")
  os.Exit(1)
 }

 sess := session.Must(session.NewSessionWithOptions(session.Options{
  SharedConfigState: session.SharedConfigEnable,
 }))
 svc := s3.New(sess, &aws.Config{
  Region: aws.String(os.Args[2]),
 })

 listMyBuckets(svc)
 createMyBucket(svc, os.Args[1], os.Args[2])
 listMyBuckets(svc)
 deleteMyBucket(svc, os.Args[1])
 listMyBuckets(svc)
}

// List all of your available buckets in this AWS Region.
func listMyBuckets(svc *s3.S3) {
```

```
 result, err := svc.ListBuckets(nil)

 if err != nil {
  exitErrorf("Unable to list buckets, %v", err)
 }

 fmt.Println("My buckets now are:\n")

 for _, b := range result.Buckets {
  fmt.Printf(aws.StringValue(b.Name) + "\n")
 }

 fmt.Printf("\n")
}

// Create a bucket in this AWS Region.
func createMyBucket(svc *s3.S3, bucketName string, region string) {
 fmt.Printf("\nCreating a new bucket named '" + bucketName + "'...\n\n")

 _, err := svc.CreateBucket(&s3.CreateBucketInput{
  Bucket: aws.String(bucketName),
  CreateBucketConfiguration: &s3.CreateBucketConfiguration{
   LocationConstraint: aws.String(region),
  },
 })

 if err != nil {
  exitErrorf("Unable to create bucket, %v", err)
 }

 // Wait until bucket is created before finishing
 fmt.Printf("Waiting for bucket %q to be created...\n", bucketName)

 err = svc.WaitUntilBucketExists(&s3.HeadBucketInput{
  Bucket: aws.String(bucketName),
 })
}

// Delete the bucket you just created.
func deleteMyBucket(svc *s3.S3, bucketName string) {
 fmt.Printf("\nDeleting the bucket named '" + bucketName + "'...\n\n")

 _, err := svc.DeleteBucket(&s3.DeleteBucketInput{
  Bucket: aws.String(bucketName),
```

```
 })

 if err != nil {
  exitErrorf("Unable to delete bucket, %v", err)
 }

 // Wait until bucket is deleted before finishing
 fmt.Printf("Waiting for bucket %q to be deleted...\n", bucketName)

 err = svc.WaitUntilBucketNotExists(&s3.HeadBucketInput{
  Bucket: aws.String(bucketName),
 })
}

// If there's an error, display it.
func exitErrorf(msg string, args ...interface{}) {
 fmt.Fprintf(os.Stderr, msg+"\n", args...)
 os.Exit(1)
}
```

6단계: AWS SDK 코드 실행

1. AWS Cloud9 IDE의 메뉴 모음에서 [실행(Run)], [실행 구성(Run Configurations)] 및 [새 실행 구성(New Run Configuration)]을 선택합니다.
2. [[새로 만들기] - 유휴([New] - Idle)] 탭에서 [러너: 자동(Runner: Auto)]을 선택한 다음 [Go]를 선택합니다.
3. [명령(Command)]에 `s3.go` *`YOUR_BUCKET_NAME`* *`THE_AWS_REGION`* 을 입력합니다. 여기서 *`YOUR_BUCKET_NAME`* 은 생성했다고 삭제할 버킷의 이름이며, *`THE_AWS_REGION`* 는 버킷을 생성할 AWS 리전의 ID입니다. 예를 들어, 미국 동부(오하이오) 리전의 경우 `us-east-2`를 사용합니다. 더 많은 ID는 Amazon Web Services 일반 참조의 Amazon Simple Storage Service(Amazon S3)를 참조하세요.

   > **Note**
   >
   > Amazon S3 버킷 이름은 해당 AWS 계정뿐만 아니라 모든 AWS에서 고유해야 합니다.

4. Run(실행) 버튼을 선택하여 출력값을 비교합니다.

```
My buckets now are:
```

```
Creating a new bucket named 'my-test-bucket'...

My buckets now are:

my-test-bucket

Deleting the bucket named 'my-test-bucket'...

My buckets now are:
```

7단계: 정리

이 샘플 사용 후 AWS 계정에 계속하여 요금이 부과되지 않도록 환경을 삭제해야 합니다. 지침은 AWS Cloud9에서 환경 삭제 단원을 참조하세요.

AWS Cloud9용 TypeScript 자습서

이 자습서는 AWS Cloud9 개발 환경에서 TypeScript를 사용하는 방법을 보여줍니다.

이 자습서를 따르고 이 샘플을 생성하면 AWS 계정에 요금이 발생할 수 있습니다. 여기에는 Amazon EC2 및 Amazon S3 같은 서비스에 대한 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금 및 Amazon S3 요금을 참조하세요.

주제

필수 조건

이 샘플을 사용하기 전에 설정이 다음 요구 사항을 충족하는지 확인하세요.

- 기존 AWS Cloud9 EC2 개발 환경이 있어야 합니다. 이 샘플에서는 Amazon Linux 또는 Ubuntu 서버를 실행 중인 Amazon EC2 인스턴스에 연결된 EC2 환경이 이미 있다고 가정합니다. 다른 환경 또는 운영 시스템이라면 이 샘플 지침을 관련 도구를 설치하는 데에 적용해야 합니다. 자세한 내용은 에서 환경 만들기 AWS Cloud9 섹션을 참조하세요.
- 기존 환경에 대한 AWS Cloud9 IDE가 이미 열려 있습니다. 환경을 열 때 AWS Cloud9은 웹 브라우저에서 환경을 위한 IDE를 엽니다. 자세한 내용은 AWS Cloud9에서 환경 열기 섹션을 참조하세요.

1단계: 필수 도구 설치

이 단계에서는 Node Package Manager(**npm**)를 사용하여 TypeScript를 설치합니다. **npm**을 설치하려면 Node Version Manager(**nvm**)를 사용합니다. **nvm**이 없는 경우 이 단계에서 먼저 설치합니다.

1. AWS Cloud9 IDE의 터미널 세션에서 **--version** 옵션을 사용하여 명령줄 TypeScript 컴파일러를 실행함으로써 TypeScript가 이미 설치되었는지 여부를 확인합니다. (터미널 세션을 새로 시작하려면 메뉴 모음에서 Window(창), New Terminal(새 터미널)을 선택합니다.) 성공할 경우, 출력에 TypeScript 버전 번호가 포함됩니다. TypeScript가 설치된 경우 2단계: 코드 추가 단원으로 이동합니다.

   ```
   tsc --version
   ```

2. **--version** 옵션을 지정하고 **npm** 을 실행하여 **npm** 이 이미 설치되어 있는지 확인합니다. 성공할 경우, 출력에 **npm** 버전 번호가 포함됩니다. **npm**이 설치된 경우 이 절차의 10단계로 건너뛰고 **npm**을 사용하여 TypeScript t를 설치합니다.

   ```
   npm --version
   ```

3. **yum update** 명령(Amazon Linux) 또는 **apt update** 명령(Ubuntu Server)을 실행하여 최신 보안 업데이트와 버그 수정이 설치되어 있는지 확인합니다.

   Amazon Linux의 경우:

   ```
   sudo yum -y update
   ```

   Ubuntu Server:

   ```
   sudo apt update
   ```

4. **npm** 을 설치하려면, 먼저 다음 명령을 실행하여 Node Version Manager(**nvm**)를 다운로드합니다. (**nvm** 은 Node.js 버전을 설치하고 관리하는 데 유용한 간단한 Bash 셸 스크립트입니다. 자세한 내용은 GitHub 웹 사이트의 Node Version Manager를 참조하세요.)

```
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh | bash
```

5. **nvm** 사용을 시작하려면 터미널 세션을 닫고 나서 다시 시작하거나, **nvm**을 로드하는 명령을 포함하는 `~/.bashrc` 파일을 소싱합니다.

```
. ~/.bashrc
```

6. **nvm** 옵션을 지정하고 **nvm** 을 실행하여 **--version** 이 설치되어 있는지 확인합니다.

```
nvm --version
```

7. **nvm**을 실행하여 최신 버전인 Node.js 16을 설치합니다(**npm**은 Node.js 에 포함되어 있음).

```
nvm install v16
```

8. **--version** 옵션으로 명령줄 버전의 Node.js를 실행하여 Node.js가 설치되어 있는지 확인합니다.

```
node --version
```

9. **npm** 옵션을 지정하고 **npm** 을 실행하여 **--version** 이 설치되어 있는지 확인합니다.

```
npm --version
```

10. **-g** 옵션으로 **npm**을 실행하여 TypeScript 설치합니다. 그러면 환경에 TypeScript가 글로벌 패키지로 설치됩니다.

```
npm install -g typescript
```

11. **--version** 옵션으로 명령줄 TypeScript 컴파일러를 실행하여 TypeScript 가 설치되어 있는지 확인합니다.

```
tsc --version
```

2단계: 코드 추가

1. AWS Cloud9 IDE에 이름이 `hello.ts`인 파일을 생성합니다. (파일을 생성하려면 메뉴 모음에서 [파일(File)], [새 파일(New File)]을 선택합니다. 파일을 저장하려면 [파일(File)], [저장(Save)]을 선택합니다.)
2. IDE의 터미널에서, `hello.ts` 파일과 동일한 디렉터리에서 **npm**을 실행하여 `@types/node` 라이브러리를 설치합니다.

   ```
   npm install @types/node
   ```

   그러면 `node_modules/@types/node` 파일과 동일한 디렉터리에 `hello.ts` 폴더가 추가됩니다. 이 새 폴더에는 이 절차의 뒷부분에서 `hello.ts` 파일에 추가할 `console.log` 및 `process.argv` 속성에 대해 TypeScript가 필요로 하는 Node.js 형식 정의가 포함되어 있습니다.
3. 다음 코드를 `hello.ts` 파일에 추가합니다:

   ```
   console.log('Hello, World!');

   console.log('The sum of 2 and 3 is 5.');

   const sum: number = parseInt(process.argv[2], 10) + parseInt(process.argv[3], 10);

   console.log('The sum of ' + process.argv[2] + ' and ' +
     process.argv[3] + ' is ' + sum + '.');
   ```

3단계: 코드 실행

1. 터미널에서, `hello.ts` 파일과 동일한 디렉터리에서 TypeScript 컴파일러를 실행합니다. 포함할 `hello.ts` 파일 및 추가 라이브러리를 지정합니다.

   ```
   tsc hello.ts --lib es6
   ```

   TypeScript는 `hello.ts` 파일 및 ECMAScript 6(ES6) 라이브러리 파일 세트를 사용하여 `hello.ts` 파일의 TypeScript 코드를 `hello.js`라는 파일의 해당하는 JavaScript 코드로 변환합니다.
2. [환경(Environment)] 창에서 `hello.js` 파일을 엽니다.

3. 메뉴 모음에서 Run(실행), Run Configurations(실행 구성), New Run Configuration(새로운 실행 구성)을 선택합니다.
4. [[새로 만들기] - 유휴([New] - Idle)] 탭에서 [러너: 자동(Runner: Auto)]을 선택한 다음 [Node.js]를 선택합니다.
5. 명령에 `hello.js 5 9`를 입력합니다. 코드에서 5는 `process.argv[2]`를 나타내고 9는 `process.argv[3]`를 나타냅니다. (`process.argv[0]`는 런타임의 이름(node)을 나타내고 `process.argv[1]`는 파일의 이름(`hello.js`)을 나타냅니다.)
6. Run(실행)을 선택하여 출력을 비교합니다. 완료했으면 [중지(Stop)]를 선택합니다.

```
Hello, World!
The sum of 2 and 3 is 5.
The sum of 5 and 9 is 14.
```

> **Note**
>
> IDE에서 새 실행 구성을 생성하는 대신 터미널에서 **`node hello.js 5 9`** 명령을 실행하여 이 코드를 실행할 수도 있습니다.

4단계: AWS SDK for JavaScript in Node.js 설치 및 구성

이 샘플을 응용해 AWS SDK for JavaScript in Node.js를 사용하여 Amazon S3 버킷을 생성하고 사용 가능한 버킷을 나열한 다음 방금 생성한 버킷을 삭제합니다.

이 단계에서는 AWS SDK for JavaScript in Node.js를 설치하고 구성합니다. 이 SDK는 JavaScript 코드에서 Amazon S3과 같은 AWS 서비스와 상호 작용할 수 있는 편리한 방법을 제공합니다. AWS SDK for JavaScript in Node.js를 설치한 후 환경에서 자격 증명 관리를 설정해야 합니다. SDK는 AWS 서비스와 상호 작용하기 위해서 이런 자격 증명이 필요합니다.

AWS SDK for JavaScript in Node.js를 설치하려면

AWS Cloud9 IDE의 터미널 세션에서 3단계: 코드 실행의 `hello.js` 파일과 동일한 디렉터리에서 **`npm`** 을 실행하여 AWS Node.js의 JavaScript용 SDK를 설치합니다.

```
npm install aws-sdk
```

이 명령은 3단계: 코드 실행의 `node_modules` 폴더에 여러 폴더를 추가합니다. 이러한 폴더에는 AWS SDK for JavaScript in Node.js의 소스 코드 및 종속 구성 요소가 들어 있습니다. 자세한 내용은 AWS SDK for JavaScript 개발자 가이드에서 SDK for JavaScript 설치를 참조하세요.

환경에서 보안 인증 관리를 설정하려면

AWS SDK for JavaScript in Node.js를 사용하여 AWS 서비스를 호출할 때마다 호출과 함께 자격 증명 세트를 제공해야 합니다. 이러한 자격 증명은 AWS SDK for JavaScript in Node.js가 해당 호출을 할 수 있는 적절한 권한이 있는지 여부를 결정합니다. 자격 증명으로 적절한 권한이 확인되지 않는 경우 호출이 실패합니다.

이 단계에서는 환경 내에서 자격 증명을 저장합니다. 이렇게 하려면 AWS Cloud9의 환경에서 AWS 서비스 호출 섹션의 지침을 수행한 다음 이 주제로 돌아옵니다.

자세한 내용은 AWS SDK for JavaScript 개발자 가이드의 Node.js에서 자격 증명 설정을 참조하세요.

5단계: AWS SDK 코드 추가

이 단계에서는 몇 가지 코드를 더 추가합니다. 이번 경우에는 Amazon S3와 상호 작용하기 위한 것이며 버킷을 생성하고 사용 가능한 버킷을 나열한 다음 막 생성한 버킷을 삭제합니다. 이 코드를 나중에 실행합니다.

1. AWS Cloud9 IDE에서, 이전 단계의 `hello.js` 파일과 동일한 디렉터리에 `s3.ts`라는 파일을 생성합니다.

2. AWS Cloud9 IDE의 터미널에서, `s3.ts` 파일과 동일한 디렉터리에서 **`npm`**를 두 번 실행하여 TypeScript용과 JavaScript용의 비동기 라이브러리를 설치함으로써 코드가 Amazon S3 작업을 비동기로 호출할 수 있도록 합니다.

```
npm install @types/async # For TypeScript.
npm install async        # For JavaScript.
```

3. 다음 코드를 `s3.ts` 파일에 추가합니다:

```
import * as async from 'async';
import * as AWS from 'aws-sdk';

if (process.argv.length < 4) {
  console.log('Usage: node s3.js <the bucket name> <the AWS Region to use>\n' +
    'Example: node s3.js my-test-bucket us-east-2');
  process.exit(1);
}

const AWS = require('aws-sdk'); // To set the AWS credentials and AWS Region.
const async = require('async'); // To call AWS operations asynchronously.

const s3: AWS.S3 = new AWS.S3({apiVersion: '2006-03-01'});
const bucket_name: string = process.argv[2];
const region: string = process.argv[3];

AWS.config.update({
  region: region
});

const create_bucket_params: any = {
  Bucket: bucket_name,
  CreateBucketConfiguration: {
    LocationConstraint: region
  }
};

const delete_bucket_params: any = {
  Bucket: bucket_name
};

// List all of your available buckets in this AWS Region.
function listMyBuckets(callback): void {
  s3.listBuckets(function(err, data) {
```

```
    if (err) {

    } else {
      console.log("My buckets now are:\n");

      for (let i: number = 0; i < data.Buckets.length; i++) {
        console.log(data.Buckets[i].Name);
      }
    }

    callback(err);
  });
}

// Create a bucket in this AWS Region.
function createMyBucket(callback): void {
  console.log("\nCreating a bucket named '" + bucket_name + "'...\n");

  s3.createBucket(create_bucket_params, function(err, data) {
    if (err) {
      console.log(err.code + ": " + err.message);
    }

    callback(err);
  });
}

// Delete the bucket you just created.
function deleteMyBucket(callback): void {
  console.log("\nDeleting the bucket named '" + bucket_name + "'...\n");

  s3.deleteBucket(delete_bucket_params, function(err, data) {
    if (err) {
      console.log(err.code + ": " + err.message);
    }

    callback(err);
  });
}

// Call the AWS operations in the following order.
async.series([
  listMyBuckets,
  createMyBucket,
```

```
  listMyBuckets,
  deleteMyBucket,
  listMyBuckets
]);
```

6단계: AWS SDK 코드 실행

1. 터미널에서, `s3.ts` 파일과 동일한 디렉터리에서 TypeScript 컴파일러를 실행합니다. 포함할 `s3.ts` 파일 및 추가 라이브러리를 지정합니다.

   ```
   tsc s3.ts --lib es6
   ```

   TypeScript는 `s3.ts` 파일, AWS SDK for JavaScript in Node.js, 비동기 라이브러리 및 ECMAScript 6(ES6) 라이브러리 파일 세트를 사용하여 `s3.ts` 파일의 TypeScript 코드를 `s3.js`라는 파일의 해당하는 JavaScript 코드로 변환합니다.
2. [환경(Environment)] 창에서 `s3.js` 파일을 엽니다.
3. 메뉴 모음에서 Run(실행), Run Configurations(실행 구성), New Run Configuration(새로운 실행 구성)을 선택합니다.
4. [[새로 만들기] - 유휴([New] - Idle)] 탭에서 [러너: 자동(Runner: Auto)]을 선택한 다음 [Node.js]를 선택합니다.
5. [명령(Command)]에 `s3.js` *`YOUR_BUCKET_NAME`* *`THE_AWS_REGION`* 을 입력합니다. 여기서 *`YOUR_BUCKET_NAME`* 은 생성했다고 삭제할 버킷의 이름이며, *`THE_AWS_REGION`* 는 버킷을 생성할 AWS 리전의 ID입니다. 예를 들어 미국 동부(오하이오) 리전의 경우 `us-east-2`를 사용합니다. 더 많은 ID는 Amazon Web Services 일반 참조의 Amazon Simple Storage Service(Amazon S3)를 참조하세요.

   > **Note**
   >
   > Amazon S3 버킷 이름은 해당 AWS 계정뿐만 아니라 모든 AWS에서 고유해야 합니다.
6. Run(실행)을 선택하여 출력을 비교합니다. 완료했으면 [중지(Stop)]를 선택합니다.

   ```
   My buckets now are:

   Creating a new bucket named 'my-test-bucket'...

   My buckets now are:
   ```

```
my-test-bucket

Deleting the bucket named 'my-test-bucket'...

My buckets now are:
```

7단계: 정리

이 샘플 사용 후 AWS 계정에 계속하여 요금이 부과되지 않도록 환경을 삭제해야 합니다. 지침은 AWS Cloud9에서 환경 삭제 단원을 참조하세요.

에 대한 도커 튜토리얼 AWS Cloud9

이 자습서에서는 Amazon EC2의 Amazon Linux 인스턴스 내에서 실행 중인 Docker 컨테이너에 AWS Cloud9 SSH 개발 환경을 연결하는 방법을 보여줍니다. 이를 통해 AWS Cloud9 IDE를 사용하여 Docker 컨테이너 내의 코드 및 파일로 작업하고 해당 컨테이너에서 명령을 실행할 수 있습니다. 도커에 대한 자세한 내용은 도커 웹 사이트의 도커란 무엇인가?를 참조하십시오.

이 자습서를 따라 이 샘플을 만들면 AWS 계정에 요금이 부과될 수 있습니다. 여기에는 Amazon EC2와 같은 서비스에 대해 발생할 수 있는 요금이 포함됩니다. 자세한 내용은 Amazon EC2 요금을 참조하세요.

주제

필수 조건

- Amazon Linux 또는 Ubuntu Server를 실행 중인 Amazon EC2 인스턴스가 있어야 합니다. 이 샘플은 계정에 아마존 리눅스 또는 우분투 서버를 실행하는 Amazon EC2 인스턴스가 이미 있다고 가정

합니다. AWS Amazon EC2 인스턴스를 시작하려면 Linux 가상 머신 시작을 참조하세요. 마법사의 [Amazon 머신 이미지(AMI) 선택(Choose an Amazon Machine Image (AMI))] 페이지에서 표시 이름이 Amazon Linux AMI 또는 Ubuntu Server로 시작하는 AMI를 선택합니다.

- Amazon EC2 인스턴스가 Amazon VPC에서 실행되는 경우 추가 요구 사항이 수반됩니다. 개발 환경을 위한 AWS Cloud9 VPC 설정를 참조하세요.
- Amazon EC2 인스턴스에 사용 가능한 디스크 공간이 8~16GB 이상 있어야 합니다. 이 샘플은 크기가 3GB를 초과하는 도커 이미지를 사용하며 추가 증분량이 3GB 이상인 디스크 공간을 사용하여 이미지를 빌드할 수 있습니다. 사용 가능한 공간이 8GB 미만인 디스크에서 이 샘플을 실행하려 할 경우 도커 이미지를 빌드할 수 없거나 도커 컨테이너가 실행되지 않을 것입니다. 인스턴스의 사용 가능한 디스크 공간을 확인하려면 인스턴스에서 **`df -h`**("인간 판독 가능한 형식의 디스크 파일 시스템 정보")와 같은 명령을 실행할 수 있습니다. 기존 인스턴스의 디스크 크기를 늘리려면 Amazon EC2 사용 설명서의 볼륨 수정을 참조하십시오.

1단계: Docker 설치 및 실행

이 단계에서는 Docker가 Amazon EC2 인스턴스에 설치되어 있는지 여부를 확인하고 아직 설치되지 않은 경우 Docker를 설치합니다. 도커를 설치한 후에는 인스턴스에서 도커를 실행합니다.

1. **`ssh`** 유틸리티 또는 PuTTY와 같은 SSH 클라이언트를 사용하여 실행 중인 Amazon EC2 인스턴스에 연결합니다. 이렇게 하려면 Linux 가상 머신 시작에서 "3단계: 인스턴스에 연결"을 참조하십시오.
2. 도커가 인스턴스에 설치되었는지 확인합니다. 이렇게 하려면 **`--version`** 옵션을 사용하여 인스턴스에 대해 **`docker`** 명령을 실행합니다.

   ```
   docker --version
   ```

   도커가 설치되면 도커 버전과 빌드 번호가 표시됩니다. 이 경우, 이 절차의 뒷부분인 5단계로 이동합니다.
3. 도커를 설치합니다. 이렇게 하려면 설치할 **`docker`** 또는 **`docker.io`** 패키지를 지정하고 **`install`** 작업을 사용하여 **`yum`** 또는 **`apt`** 명령을 실행합니다.

   Amazon Linux의 경우:

   ```
   sudo yum install -y docker
   ```

   Ubuntu Server:

```
sudo apt install -y docker.io
```

4. 도커가 설치되었는지 확인합니다. 이렇게 하려면 **docker --version** 명령을 다시 실행합니다. 도커 버전 및 빌드 번호가 표시됩니다.
5. 도커를 실행합니다. 이렇게 하려면 **docker** 서비스와 **start** 작업을 사용하여 **service** 명령을 실행합니다.

```
sudo service docker start
```

6. 도커가 실행 중인지 확인합니다. 이렇게 하려면 **info** 작업을 사용하여 **docker** 명령을 실행합니다.

```
sudo docker info
```

도커를 실행 중인 경우 도커에 대한 정보가 표시됩니다.

2단계: 이미지 빌드

이 단계에서 Dockerfile을 사용하여 도커 이미지를 인스턴스에 빌드합니다. 이 샘플에서는 Node.js와 샘플 채팅 서버 애플리케이션을 포함하는 이미지를 사용합니다.

1. 인스턴스에서 Dockerfile을 생성합니다. 이렇게 하려면 인스턴스의 `/tmp` 디렉터리에서 인스턴스에 여전히 연결되어 있는 SSH 클라이언트를 사용하여 `Dockerfile` 이름의 파일을 생성합니다. 예를 들면 다음과 같이 **touch** 명령을 실행합니다.

```
sudo touch /tmp/Dockerfile
```

2. 다음 내용을 `Dockerfile` 파일에 추가합니다.

```
# Build a Docker image based on the Amazon Linux 2 Docker image.
FROM amazonlinux:2

# install common tools
RUN yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-
latest-7.noarch.rpm
RUN yum update -y
RUN yum install -y sudo bash curl wget git man-db nano vim bash-completion tmux
 gcc gcc-c++ make tar
```

```
# Enable the Docker container to communicate with AWS Cloud9 by
# installing SSH.
RUN yum install -y openssh-server

# Ensure that Node.js is installed.
RUN yum install -y nodejs

# Create user and enable root access
RUN useradd --uid 1000 --shell /bin/bash -m --home-dir /home/ubuntu ubuntu && \
    sed -i 's/%wheel\s.*/%wheel ALL=NOPASSWD:ALL/' /etc/sudoers && \
    usermod -a -G wheel ubuntu

# Add the AWS Cloud9 SSH public key to the Docker container.
# This assumes a file named authorized_keys containing the
# AWS Cloud9 SSH public key already exists in the same
# directory as the Dockerfile.
RUN mkdir -p /home/ubuntu/.ssh
ADD ./authorized_keys /home/ubuntu/.ssh/authorized_keys
RUN chown -R ubuntu /home/ubuntu/.ssh /home/ubuntu/.ssh/authorized_keys && \
chmod 700 /home/ubuntu/.ssh && \
chmod 600 /home/ubuntu/.ssh/authorized_keys

# Update the password to a random one for the user ubuntu.
RUN echo "ubuntu:$(cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 32 | head -n 1)"
 | chpasswd

# pre-install Cloud9 dependencies
USER ubuntu
RUN curl https://d2j6vhu5uywtq3.cloudfront.net/static/c9-install.sh | bash

USER root
# Start SSH in the Docker container.
CMD ssh-keygen -A && /usr/sbin/sshd -D
```

앞의 내용을 `Dockerfile` 파일에 추가하려면 다음과 같이 인스턴스에서 **`vi`** 유틸리티를 사용할 수 있습니다.

a. 를 AWS Cloud9 사용하여 파일을 열고 `/tmp/Dockerfile` 편집할 수 있습니다.

```
sudo vi /tmp/Dockerfile
```

b. 앞의 내용을 `Dockerfile` 파일에 붙여 넣습니다. 이 동작을 수행하는 방법을 잘 모르는 경우 SSH 클라이언트 설명서를 참조하십시오.

c. 명령 모드로 전환합니다. 이렇게 하려면 `Esc` 키를 누릅니다. 그러면 `-- INSERT --`가 창의 하단에서 사라집니다.

d. `:wq`를 입력하여 `/tmp/Dockerfile` 파일에 쓰고 나서 파일을 저장한 다음 **`vi`**를 종료한 후, `Enter`를 누릅니다.

> **Note**
>
> 에서 AWS CodeBuild자주 업데이트되는 Docker 이미지 목록에 액세스할 수 있습니다. 자세한 내용은 AWS CodeBuild 사용 CodeBuild 설명서에서 제공하는 Docker 이미지를 참조하십시오.

3. 인스턴스에서 Docker 컨테이너가 사용할 AWS Cloud9 SSH 공개 키가 포함된 파일을 생성합니다. 이렇게 하려면 예를 들어 **`touch`** 명령을 실행하여 `Dockerfile` 파일과 같은 디렉터리에 이름이 `authorized_keys`인 파일을 생성합니다.

```
sudo touch /tmp/authorized_keys
```

4. AWS Cloud9 SSH 퍼블릭 키를 파일에 추가합니다. `authorized_keys` AWS Cloud9 SSH 공개 키를 가져오려면 다음과 같이 하십시오.

a. https://console.aws.amazon.com/cloud9/ 에서 AWS Cloud9 콘솔을 엽니다.

b. AWS 탐색 표시줄의 AWS 지역 선택기에서 이 항목의 뒷부분에서 AWS Cloud9 개발 환경을 만들려는 AWS 지역을 선택합니다.

c. 시작 페이지가 표시되면 새 AWS Cloud9 환경에서 환경 만들기를 선택합니다. 그렇지 않으면 Create environment(환경 생성)를 선택합니다.

d. [환경 이름 지정(Name environment)] 페이지의 [이름(Name)]에 환경의 이름을 입력합니다. (여기서는 이름이 중요하지 않습니다. 나중에 다른 이름을 선택할 수 있습니다.)

e. 다음 단계를 선택합니다.

f. Environment type(환경 유형)에서 Connect and run in remote server(SSH)(원격 서버에서 연결 및 실행(SSH))를 선택합니다.

g. View public SSH key(퍼블릭 SSH 키 보기)를 확장합니다.

h. Copy key to clipboard(클립보드에 키 복사)를 선택합니다. (이 항목은 View public SSH key(퍼블릭 SSH 키 보기) 및 Advanced settings(고급 설정) 사이에 있습니다.)

i. 취소를 선택합니다.

j. 클립보드의 내용을 `authorized_keys` 파일에 붙여 넣고 나서 파일을 저장합니다. 예를 들어 이 단계의 앞부분에서 설명한 대로 **`vi`** 유틸리티를 사용할 수 있습니다.

5. **`build`** 작업을 사용하고 `cloud9-image:latest` 태그를 이미지에 추가하고 사용할 `Dockerfile` 파일의 경로를 지정하여 **`docker`** 명령을 실행함으로써 이미지를 빌드합니다.

```
sudo docker build -t cloud9-image:latest /tmp
```

명령이 성공적으로 실행되면 빌드 출력의 마지막 두 줄에 `Successfully built` 및 `Successfully tagged`가 표시됩니다.

Docker가 이미지를 성공적으로 빌드했는지 확인하려면 `image ls` 작업을 사용하여 **`docker`** 명령을 실행합니다.

```
sudo docker image ls
```

명령이 성공적으로 실행되면 출력 디스플레이에 항목 하나가 표시됩니다. 이 항목에서 `REPOSITORY` 필드는 `cloud9-image`로 설정되었으며 `TAG` 필드는 `latest`로 설정되었습니다.

6. Amazon EC2 인스턴스의 퍼블릭 IP 주소를 적어둡니다. 4단계: 환경 생성에 필요한 정보입니다. 인스턴스의 퍼블릭 IP 주소를 잘 모르는 경우 인스턴스에서 다음 명령을 실행하여 해당 주소를 가져올 수 있습니다.

```
curl http://169.254.169.254/latest/meta-data/public-ipv4
```

3단계: 컨테이너 실행

이 단계에서는 인스턴스에서 도커 컨테이너를 실행합니다. 이 컨테이너는 이전 단계에서 빌드한 이미지에 기초합니다.

1. Docker 컨테이너를 실행하려면 **`run`** 작업과 다음 옵션을 사용하여 인스턴스에 대해 **`docker`** 명령을 실행합니다.

```
sudo docker run -d -it --expose 9090 -p 0.0.0.0:9090:22 --name cloud9 cloud9-
image:latest
```

- `-d`는 컨테이너를 분리 모드로 실행하고 컨테이너(이 샘플에서는 SSH 클라이언트)를 실행하는 데 사용되는 루트 프로세스가 종료될 때마다 종료합니다.
- `-it`는 의사 TTY가 할당된 컨테이너를 실행하고 컨테이너가 연결되지 않았어도 STDIN을 열린 상태로 유지합니다.
- `--expose`는 지정한 포트(이 샘플에서는 `9090`)를 컨테이너에서 사용할 수 있도록 설정합니다.
- `-p`는 지정한 IP 주소 및 포트를 통해 지정한 포트를 Amazon EC2 인스턴스 내부에서 사용할 수 있도록 설정합니다. 이 샘플에서 컨테이너의 포트 `9090`은 Amazon EC2 인스턴스의 포트 `22`를 통해 내부에서 액세스할 수 있습니다.
- `--name`은 컨테이너용 인간이 읽을 수 있는 이름(이 샘플에서는 `cloud9`)입니다.
- `cloud9-image:latest`는 컨테이너를 실행하는 데 사용할 빌드된 이미지의 인간이 읽을 수 있는 이름입니다.

Docker가 컨테이너를 성공적으로 실행하고 있는지 확인하려면 `container ls` 작업을 사용하여 **`docker`** 명령을 실행합니다.

```
sudo docker container ls
```

명령이 성공적으로 실행되면 출력 디스플레이에 항목 하나가 표시됩니다. 이 항목에서 `IMAGE` 필드는 `cloud9-image:latest`로 설정되었으며 `NAMES` 필드는 `cloud9`로 설정되었습니다.

2. 실행 중인 컨테이너에 로그인합니다. 이렇게 하려면 **`exec`** 작업과 다음 옵션을 사용하여 **`docker`** 명령을 실행합니다.

```
sudo docker exec -it cloud9 bash
```

- `-it`는 의사 TTY가 할당된 컨테이너를 실행하고 컨테이너가 연결되지 않았어도 STDIN을 열린 상태로 유지합니다.
- `cloud9`는 실행 중인 컨테이너의 인간이 읽을 수 있는 이름입니다.
- `bash`는 실행 중인 컨테이너에서 표준 셸을 시작합니다.

성공할 경우 터미널 프롬프트가 변경되어 컨테이너의 로그인된 사용자 이름과 컨테이너의 ID를 표시합니다.

> **Note**
>
> 실행 중인 컨테이너에서 로그아웃하려면 **exit** 명령을 실행합니다. 터미널 프롬프트가 다시 변경되어 인스턴스의 로그인된 사용자 이름과 인스턴스의 프라이빗 DNS를 표시합니다. 컨테이너가 여전히 실행 중이어야 합니다.

3. 실행 중인 컨테이너에서 로그인한 후 AWS Cloud9 시작하려는 디렉터리의 액세스 권한을 로 설정합니다 **rwxr-xr-x**. 이는 소유자에 대한 read-write-execute 권한, 그룹에 대한 읽기 실행 권한, 다른 사람에 대한 읽기 실행 권한을 의미합니다. 예를 들어 디렉터리의 경로가 ~인 경우 다음과 같이 실행 중인 컨테이너에서 **chmod** 명령을 실행하여 디렉터리에서 이러한 권한을 설정할 수 있습니다.

```
sudo chmod u=rwx,g=rx,o=rx ~
```

4. Node.js 바이너리를 포함하는 실행 중인 컨테이너의 디렉터리에 대한 경로를 적어둡니다. 이 경로는 4단계: 환경 생성에 필요합니다. 이 경로를 잘 모르면 실행 중인 컨테이너에서 다음 명령을 실행하여 가져옵니다.

```
which node
```

4단계: 환경 생성

이 단계에서는 AWS Cloud9 SSH 개발 환경을 만들고 AWS Cloud9 이를 실행 중인 Docker 컨테이너에 연결하는 데 사용합니다. 환경을 AWS Cloud9 만든 후에는 AWS Cloud9 IDE를 표시하므로 컨테이너의 파일 및 코드 작업을 시작할 수 있습니다.

AWS Cloud9 콘솔을 사용하여 AWS Cloud9 SSH 개발 환경을 생성합니다. CLI를 사용하여 SSH 환경을 만들 수는 없습니다.

필수 조건

- 먼저 AWS Cloud9 설정의 단계를 완료했는지 확인합니다. 이렇게 해야 AWS Cloud9 콘솔에 로그인하여 환경을 생성할 수 있습니다.
- 환경에 AWS Cloud9 연결하려는 기존 클라우드 컴퓨팅 인스턴스 (예: Amazon EC2 인스턴스 AWS 계정) 또는 자체 서버를 식별합니다.

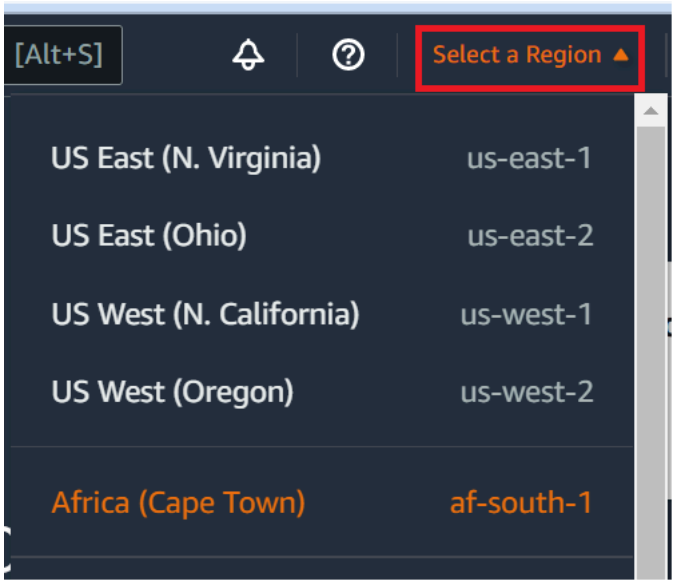
- 기존 인스턴스 또는 자체 서버가 모든 SSH 호스트 요구 사항을 충족해야 합니다. 이러한 요구 사항에는 특정 버전의 Python, Node.js 및 기타 구성 요소 설치, 로그인 후 AWS Cloud9 이 시작하도록 할 디렉터리에 대한 특정 권한 설정, 연결된 Amazon Virtual Private Cloud 설정이 포함됩니다.

SSH 환경 생성

1. 위의 사전 조건을 완료해야 합니다.
2. 아직 연결되지 않은 경우 SSH 클라이언트를 사용하여 기존 인스턴스 또는 자체 서버에 연결합니다. 이렇게 하면 필요한 공개 SSH 키 값을 인스턴스나 서버에 추가할 수 있습니다. 이 절차의 뒷부분에서 자세하게 설명합니다.

   > **Note**
   >
   > 기존 AWS 클라우드 컴퓨팅 인스턴스에 연결하려면 다음 리소스 중 하나 이상을 참조하십시오.
   >
   > - Amazon EC2의 경우 Amazon EC2 사용 설명서의 Linux 인스턴스에 연결을 참조하십시오.
   > - Amazon Lightsail의 경우 Amazon Lightsail 문서에서 Linux/Unix 기반 Lightsail 인스턴스에 연결을 참조하세요.
   > - 에 대해서는 AWS Elastic BeanstalkAWS Elastic Beanstalk 개발자 안내서의 서버 인스턴스 나열 및 연결을 참조하십시오.
   > - 에 대해서는 AWS OpsWorks사용 설명서의 SSH를 사용하여 Linux 인스턴스에 로그인하는 방법을 참조하십시오.AWS OpsWorks
   > - 기타 AWS 서비스정보는 해당 서비스의 설명서를 참조하십시오.
   >
   > 자체 서버에 연결하려면 SSH를 사용하세요. SSH는 macOS 및 Linux 운영 체제에서는 이미 설치되어 있습니다. 윈도우에서 SSH를 사용하여 서버에 연결하려면 PuTTY를 설치해야 합니다.

3. AWS Cloud9 콘솔의 https://console.aws.amazon.com/cloud9/ 에 로그인합니다.
4. AWS Cloud9 콘솔에 로그인한 후 상단 탐색 표시줄에서 환경을 만들 항목을 AWS 리전 선택합니다. 사용 가능한 목록은 AWS 리전AWS Cloud9을 참조하십시오 AWS 일반 참조.



5. 개발 환경을 처음 생성하는 경우 시작 페이지가 표시됩니다. 새 AWS Cloud9 환경 패널에서 환경 만들기를 선택합니다.

   이전에 개발 환경을 생성한 경우 화면 왼쪽의 창을 확장합니다. Your environments(사용자 환경)를 선택하고 Create environment(환경 생성)를 선택합니다.

   시작 페이지에서:

   또는 Your environments(환경) 페이지에서:

6. Create environment(환경 생성) 페이지에 환경의 이름을 입력합니다.
7. Description(설명)에 환경에 대한 설명을 입력합니다. 본 자습서에서는 `This environment is for the AWS Cloud9 tutorial.`을 사용합니다.
8. Environment type(환경 유형)에서는 다음 옵션 중에서 Existing Compute(기존 컴퓨팅)를 선택합니다.

- 새 EC2 인스턴스 — SSH를 통해 직접 연결할 수 AWS Cloud9 있는 Amazon EC2 인스턴스를 시작합니다.

- 기존 컴퓨팅 — 개방형 인바운드 포트가 필요하지 않은 Amazon EC2 인스턴스를 시작합니다. AWS Cloud9 를 통해 인스턴스에 연결합니다. AWS Systems Manager
  - Existing compute(기존 컴퓨팅) 옵션을 선택하면 Systems Manager가 사용자를 대신하여 EC2 인스턴스와 상호 작용할 수 있도록 서비스 역할과 IAM 인스턴스 프로파일이 생성됩니다. 인터페이스의 더 아래에 있는 Service role and instance profile for Systems Manager access(Systems Manager 액세스를 위한 서비스 역할 및 인스턴스 프로파일) 섹션에서 서비스 역할과 인스턴스 프로파일의 이름을 모두 확인할 수 있습니다. 자세한 정보는 AWS Systems Manager를 사용하여 수신하지 않는 EC2 인스턴스에 액세스을 참조하세요.

> **Warning**
>
> 사용자 환경에 맞게 EC2 인스턴스를 생성하면 Amazon EC2에 AWS 계정 대한 요금이 부과될 수 있습니다. Systems Manager를 사용하여 EC2 인스턴스에 대한 연결을 관리하는 데 따른 추가 비용은 없습니다.

> **Warning**
>
> AWS Cloud9 SSH 퍼블릭 키를 사용하여 서버에 안전하게 연결합니다. 보안 연결을 설정하려면 다음 단계에서 공개 키를 `~/.ssh/authorized_keys` 파일에 추가하고 로그인 보안 인증 정보를 제공하십시오. Copy key to clipboard(클립보드에 키 복사)를 선택하여 SSH 키를 복사하거나, View public SSH key(공개 SSH 키 보기)를 선택하여 키를 표시합니다.

9. Existing compute(기존 컴퓨팅) 패널의 User(사용자)에 이 절차의 앞부분에서 인스턴스 또는 서버에 연결하는 데 사용한 로그인 이름을 입력합니다. 예를 들어 AWS 클라우드 컴퓨팅 인스턴스의 경우 이 값은 `ec2-user`, `ubuntu` 또는 `root`일 수 있습니다.

> **Note**
>
> 로그인 이름이 인스턴스 또는 서버의 관리 권한 또는 관리자 사용자와 연결되는 것이 좋습니다. 더 자세히 말하자면, 이 로그인 이름은 인스턴스 또는 서버에서 Node.js 설치를

소유하는 것이 좋습니다. 이를 확인하려면 인스턴스 또는 서버의 터미널에서 **`ls -l $(which node)`** (또는 nvm을 사용하는 경우 **`ls -l $(nvm which node)`**) 명령을 실행합니다. 이 명령은 Node.js 설치의 소유자 이름을 표시합니다. 설치의 권한, 그룹 이름과 위치도 표시합니다.

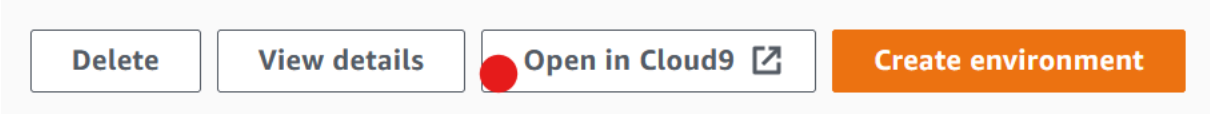
10. Host(호스트)에 인스턴스 또는 서버의 퍼블릭 IP 주소(기본) 또는 호스트 이름을 입력합니다.
11. Port에는 인스턴스 또는 서버에 연결하는 AWS Cloud9 데 사용할 포트를 입력합니다. 또는 기본 포트를 그대로 유지합니다.
12. Additional details - optional(추가 상세 정보 - 선택 사항)을 선택하여 환경 경로, node.js 바이너리 경로 및 SSH 점프 호스트 정보를 표시합니다.
13. 환경 경로에는 AWS Cloud9 시작하려는 인스턴스 또는 서버의 디렉터리 경로를 입력합니다. 이 절차의 사전 요구 사항에서 이를 이미 확인했습니다. 이 항목을 비워 두면 AWS Cloud9 은 로그인 후 인스턴스 또는 서버가 일반적으로 시작되는 디렉터리를 사용합니다. 이 디렉터리는 일반적으로 홈 또는 기본 디렉터리입니다.
14. Path to Node.js binary path(Node.js 바이너리 경로로 가는 경로)에 경로 정보를 입력하여, 인스턴스 또는 서버의 Node.js 바이너리로 가는 경로를 지정합니다. 경로를 가져오려면 인스턴스 또는 서버에서 **`which node`**(또는 nvm을 사용하는 경우 **`nvm which node`**) 명령을 실행할 수 있습니다. 예를 들어 이 경로는 `/usr/bin/node`일 수 있습니다. 이 항목을 비워 두면 AWS Cloud9 은 연결을 시도할 때 Node.js 바이너리의 위치를 추측합니다.
15. SSH jump host(SSH 점프 호스트)에 인스턴스 또는 서버가 사용하는 점프 호스트에 대한 정보를 입력합니다. 형식 `USER_NAME@HOSTNAME:PORT_NUMBER`(예:`ec2-user@:ip-192-0-2-0:22`)를 사용합니다.

    점프 호스트는 다음 요구 사항을 충족해야 합니다.

    - SSH를 사용하여 퍼블릭 인터넷을 통해 접근 가능해야 합니다.
    - 지정된 포트를 통해 어떤 IP 주소에서든 인바운드 액세스를 허용해야 합니다.
    - 기존 인스턴스 또는 서버의 `~/.ssh/authorized_keys` 파일에 복사된 퍼블릭 SSH 키 값을 점프 호스트의 `~/.ssh/authorized_keys` 파일에도 복사해야 합니다.
    - Netcat이 설치되어 있어야 합니다.
16. 각 태그에 키와 값을 지정하여 최대 50개의 태그를 추가합니다. Add new tag(새 태그 추가)를 선택하면 됩니다. 태그는 AWS Cloud9 환경에 리소스 태그로 연결되며 AWS CloudFormation 스택, Amazon EC2 인스턴스, Amazon EC2 보안 그룹과 같은 기본 리소스에 전파됩니다. 태그에 대한 자세한 내용은 IAM 사용 설명서의 AWS 리소스 태그를 사용한 액세스 제어 및 이 안내서의 태그에 대한 고급 정보를 참조하십시오.

> **Warning**
>
> 태그를 생성한 후 이러한 태그를 업데이트하면 변경 사항이 기본 리소스에 전파되지 않습니다. 자세한 내용은 태그에 대한 고급 정보에서 기본 리소스에 태그 업데이트 전파 섹션을 참조하세요.

17. Create(생성)를 선택하여 환경을 만들면 홈 페이지로 리디렉션됩니다. 계정이 성공적으로 생성되면 AWS Cloud9 콘솔 상단에 녹색 플래쉬 바가 나타납니다. 새 환경을 선택하고 Open in Cloud9(Cloud9에서 열기)을 선택하여 IDE를 시작할 수 있습니다.



계정이 생성되지 못하면 AWS Cloud9 콘솔 상단에 적색 플래시바가 나타납니다. 웹 브라우저, AWS 액세스 권한, 인스턴스 또는 관련 네트워크에 문제가 있어 계정을 만들지 못할 수 있습니다. 계정 장애를 일으킬 수 있는 문제를 해결하는 방법 관련 정보는 AWS Cloud9 문제 해결 섹션에서 확인할 수 있습니다.

> **Note**
>
> 프록시를 사용하여 인터넷에 액세스하는 환경인 경우 종속 항목을 설치할 수 AWS Cloud9 있도록 프록시 세부 정보를 제공해야 합니다. 자세한 정보는 종속성을 설치하지 못함을 참조하세요.

5단계: 코드 실행

이 단계에서는 AWS Cloud9 IDE를 사용하여 실행 중인 Docker 컨테이너 내에서 샘플 애플리케이션을 실행합니다.

1. 실행 중인 컨테이너의 AWS Cloud9 IDE가 표시된 상태에서 샘플 채팅 서버를 시작합니다. 이렇게 하려면 Environment(환경) 창에서 샘플 `workspace/server.js` 파일을 마우스 오른쪽 버튼으로 클릭하고 나서 Run(실행)을 선택합니다.
2. 샘플 애플리케이션을 미리 봅니다. 이렇게 하려면 Environment(환경) 창에서 `workspace/client/index.html` 파일을 엽니다. 그런 다음, 메뉴 모음에서 Tools, Preview, Preview Running Application(도구, 미리 보기, 실행 중인 애플리케이션 미리 보기)을 선택합니다.

3. 애플리케이션의 미리 보기 탭에서 Your Name(이름)에 자신의 이름을 입력합니다. [Message]에 메시지를 입력합니다. 그런 다음 Send(전송)를 선택합니다. 채팅 서버에서 이름과 메시지를 목록에 추가합니다.

6단계: 정리

이 단계에서는 환경을 삭제하고 Amazon EC2 AWS Cloud9 인스턴스에서 Docker 지원 파일을 제거합니다. 또한 이 샘플 사용을 완료한 후 AWS 계정에 계속 요금이 청구되는 것을 방지하려면 Docker를 실행 중인 Amazon EC2 인스턴스를 종료해야 합니다.

6.1단계: 환경 삭제

환경을 삭제하려면 AWS Cloud9에서 환경 삭제 섹션을 참조하세요.

6.2단계: 컨테이너에서 AWS Cloud9 지원 파일 제거

환경을 삭제한 후에도 일부 AWS Cloud9 지원 파일은 여전히 컨테이너에 남아 있습니다. 컨테이너를 계속 사용하지만 이러한 지원 파일이 더 이상 필요하지 않은 경우, 로그인한 후 AWS Cloud9 시작하도록 지정한 컨테이너의 디렉터리에서 `.c9` 폴더를 삭제하십시오. 예를 들어 디렉터리가 ~인 경우 다음과 같이 **`-r`** 옵션을 사용하여 **`rm`** 명령을 실행합니다.

```
sudo rm -r ~/.c9
```

6.3단계: 인스턴스에서 Docker 지원 파일 제거

Amazon EC2 인스턴스에서 Docker 컨테이너, Docker 이미지 및 Docker를 더 이상 유지하지 않고 인스턴스만 유지하려는 경우 다음과 같이 이러한 Docker 지원 파일을 제거할 수 있습니다.

1. 인스턴스에서 도커 컨테이너를 제거합니다. 이렇게 하려면 **`stop`** 및 **`rm`** 중지 작업과 사용자가 알아볼 수 있는 컨테이너 이름을 사용하여 인스턴스에 대해 **`docker`** 명령을 실행합니다.

   ```
   sudo docker stop cloud9
   sudo docker rm cloud9
   ```

2. 인스턴스에서 도커 이미지를 제거합니다. 이렇게 하려면 **`image rm`** 작업과 이미지의 태그를 사용하여 인스턴스에 대해 **`docker`** 명령을 실행합니다.

   ```
   sudo docker image rm cloud9-image:latest
   ```

3. 여전히 존재할 수 있는 추가 도커 지원 파일을 제거합니다. 이렇게 하려면 **system prune** 작업을 사용하여 인스턴스에 대해 **docker** 명령을 실행합니다.

```
sudo docker system prune -a
```

4. 도커를 제거합니다. 이렇게 하려면 제거할 **docker** 패키지를 지정하고 **remove** 작업을 사용하여 인스턴스에 대해 **yum** 명령을 실행합니다.

   Amazon Linux의 경우:

```
sudo yum -y remove docker
```

   Ubuntu Server:

```
sudo apt -y remove docker
```

   또한 이전에 생성한 `Dockerfile` 및 `authorized_keys` 파일도 제거할 수 있습니다. 예를 들면 인스턴스에서 **rm** 명령을 실행합니다.

```
sudo rm /tmp/Dockerfile
sudo rm /tmp/authorized_keys
```

6.4단계: 인스턴스 종료

Amazon EC2 인스턴스를 종료하려면 Amazon EC2 사용 설명서의 인스턴스 종료를 참조하십시오.

관련 자습서

- AWS RoboMaker 개발자 가이드의 AWS RoboMaker 시작하기. 이 자습서는 AWS Cloud9을 사용하여 샘플 로봇 애플리케이션을 수정, 빌드 및 번들화합니다.

AWS Cloud9에 대한 고급 주제

이 주제에 포함되는 정보는 다음과 같습니다.

- 고급 구성 및 의사 결정에 사용되는 정보
- 특정 태스크와 관련이 있고 AWS Cloud9을 자세히 이해하는 데 도움이 되지만 태스크 완료에 그리 중요하지 않은 정보

주제

AWS Cloud9에서 EC2 환경과 SSH 환경 비교

환경 및 컴퓨팅 리소스의 소개와 환경 작업에서 설명하는 대로 AWS Cloud9 환경을 EC2 환경 또는 SSH 환경으로 설정할 수 있습니다.

다음 표는 AWS Cloud9에서 EC2 환경을 사용할 때와 SSH 환경을 사용할 때의 유사점과 차이점을 모두 보여 줍니다.

EC2 환경	SSH 환경
AWS Cloud9연결된 Amazon EC2 인스턴스를 생성하고 인스턴스의 수명 주기를 관리합니다. 여기에는 시작, 중지 및 종료 작업이 포함됩니다.	기존 클라우드 컴퓨팅 인스턴스 또는 자체 서버를 사용합니다. 수명 주기를 관리할 책임은 사용자에게 있습니다.

EC2 환경	SSH 환경
인스턴스는 Amazon Linux 또는 Ubuntu Server에서 실행됩니다.	Linux를 실행하는 클라우드 컴퓨팅 인스턴스를 사용하거나 Linux를 실행하는 자체 서버를 사용할 수 있습니다.
AWS Cloud9가 AWS Cloud9 사용을 시작하도록 인스턴스를 자동으로 설정합니다.	AWS Cloud9를 사용하도록 인스턴스 또는 자체 서버를 수동으로 구성해야 합니다.
AWS Cloud9이 인스턴스에서 AWS Command Line Interface(AWS CLI)를 자동으로 설정합니다.	인스턴스 또는 자체 서버에서 AWS CLI를 사용하려면 직접 설정해야 합니다.
인스턴스가 이미 설치 및 구성된 일반적인 패키지를 포함하여 수백 개의 유용한 패키지에 액세스할 수 있습니다. 예를 들면 Git, Docker, Node.js 및 Python이 있습니다.	일반 작업을 완료하려면 추가 패키지를 다운로드, 설치 및 구성해야 할 수 있습니다.
예를 들어 시스템 업데이트를 주기적으로 적용하여 인스턴스를 유지 관리합니다.	인스턴스 또는 자체 서버를 유지 관리합니다.
환경을 삭제하면 AWS Cloud9이 연결된 인스턴스를 자동으로 종료합니다.	환경을 삭제해도 인스턴스 또는 자체 서버는 유지됩니다.
AWS 관리형 임시 보안 인증 정보는 EC2 환경에서 사용할 수 없습니다. 이러한 보안 인증 정보를 사용하면 호출자의 AWS 계정에서 모든 AWS 리소스에 대한 모든 AWS 작업을 일부 제한 사항 적용에 따라 사용 또는 사용 중지할 수 있습니다. 환경의 Amazon EC2 인스턴스에 대한 인스턴스 프로파일을 구성하거나 AWS 엔터티(예: IAM 사용자)의 영구 AWS 액세스 보안 인증 정보를 저장할 필요가 없습니다. 해당 환경의 EC2 인스턴스가 프라이빗 서브넷에서 시작된 경우 AWS 관리형 임시 보안 인증을 사용하여 EC2 환경이 AWS 엔터티(예: IAM 사용자)를 대신하여 AWS 서비스에 액세스하도록 허용할 수 없습니다.	AWS 관리형 임시 자격 증명은 SSH 환경에서 사용할 수 없습니다. AWS Identity and Access Management를 사용하여 AWS Cloud9과 기타 AWS 서비스 및 리소스를 모두 사용할 수 있는 권한을 관리해야 합니다.

EC2 환경	SSH 환경
AWS 도구 키트, Git 패널 및 향상된 Java 지원을 사용할 수 있습니다.	AWS 도구 키트, Git 패널 및 향상된 Java 지원을 사용할 수 없습니다.

개발 환경을 위한 AWS Cloud9 VPC 설정

Amazon VPC (가상 사설 클라우드) 와 관련된 모든 AWS Cloud9 개발 환경은 특정 VPC 요구 사항을 충족해야 합니다. 이러한 환경에는 EC2 환경 및 VPC 내에서 실행되는 AWS 클라우드 컴퓨팅 인스턴스와 연결된 SSH 환경이 포함됩니다. 예를 들면 Amazon EC2 및 Amazon Lightsail 인스턴스가 있습니다.

다음에 대한 Amazon VPC 요구 사항 AWS Cloud9

를 AWS Cloud9 사용하는 Amazon VPC에는 다음 설정이 필요합니다. 이미 이러한 요구 사항을 잘 알고 있고 단지 호환되는 VPC를 생성하고 싶은 경우, VPC 및 기타 VPC 리소스 생성 섹션으로 이동합니다.

다음 체크리스트를 사용하여 VPC가 다음 요구 사항을 모두 충족하는지 확인합니다.

- VPC는 AWS Cloud9 개발 환경과 같을 수도 AWS 계정 있고 AWS 리전 , VPC는 환경과 다른 공유 VPC일 수도 있습니다. AWS 계정 하지만 VPC는 환경과 AWS 리전 동일해야 합니다. Amazon VPC에 대한 자세한 내용은 AWS 리전을 참조하십시오AWS 리전의 VPC 목록 보기. Amazon VPC 생성에 대한 AWS Cloud9자세한 지침은 을 참조하십시오. VPC 및 기타 VPC 리소스 생성 공유 Amazon VPC로 작업에 대한 자세한 내용은 Amazon VPC 사용 설명서의 공유 VPC 작업을 참조하세요.
- VPC에는 퍼블릭 서브넷이 있어야 합니다. 서브넷의 트래픽이 인터넷 게이트웨이로 라우팅되는 경우 서브넷이 퍼블릭입니다. Amazon VPC의 서브넷 목록은 VPC의 서브넷 목록 보기 단원을 참조하세요.
- 환경이 SSH를 통해 EC2 인스턴스에 직접 액세스하는 경우 퍼블릭 서브넷에서만 인스턴스를 시작할 수 있습니다. 서브넷이 퍼블릭인지 확인하는 방법에 대한 자세한 내용은 서브넷이 퍼블릭인지 여부 확인 단원을 참조하세요.
- 수신하지 않는 Amazon EC2 인스턴스에 Systems Manager를 사용하여 액세스하는 경우 인스턴스를 퍼블릭 또는 프라이빗 서브넷으로 시작할 수 있습니다.
- 퍼블릭 서브넷을 사용하는 경우 VPC와 인터넷 게이트웨이를 연결합니다. 이렇게 하면 인스턴스의 AWS Systems Manager Agent (SSM Agent) 를 Systems Manager에 연결할 수 있습니다.

- 프라이빗 서브넷을 사용하는 경우 퍼블릭 서브넷에서 NAT 게이트웨이를 호스트하여 서브넷의 인스턴스가 인터넷과 통신할 수 있도록 합니다. 인터넷 게이트웨이의 설정 보기 또는 변경에 대한 자세한 내용은 인터넷 게이트웨이의 설정 보기 또는 변경 단원을 참조하세요.
- 퍼블릭 서브넷에는 최소 경로 집합이 포함된 라우팅 테이블이 있어야 합니다. 서브넷에 라우팅 테이블이 있는지 확인하는 방법은 서브넷에 라우팅 테이블이 있는지 확인 단원을 참조하세요. 라우팅 테이블 생성 방법에 대한 자세한 내용은 라우팅 테이블 생성 단원을 참조하세요.
- VPC (또는 아키텍처에 따라 AWS 클라우드 컴퓨팅 인스턴스) 의 관련 보안 그룹은 최소한의 인바운드 및 아웃바운드 트래픽 세트를 허용해야 합니다. Amazon VPC의 보안 그룹 목록은 VPC의 보안 그룹 목록 보기 단원을 참조하세요. Amazon VPC에서 보안 그룹을 생성하는 방법에 대한 자세한 내용은 VPC에 보안 그룹 생성 단원을 참조하세요.
- 추가 보안 계층을 위해 VPC에 네트워크 ACL이 있는 경우, 네트워크 ACL이 최소 집합의 인바운드 및 아웃바운드 트래픽을 허용해야 합니다. Amazon VPC에 네트워크 ACL이 하나 이상 있는지 확인하려면 VPC에 네트워크 ACL이 하나 이상 있는지 확인 단원을 참조하세요. 네트워크 ACL 생성에 대한 자세한 내용은 네트워크 ACL을 생성 단원을 참조하세요.
- 개발 환경에서 SSM을 사용하여 EC2 인스턴스에 액세스하는 경우 인스턴스가 시작된 퍼블릭 서브넷에 의해 인스턴스에 퍼블릭 IP 주소가 할당되었는지 확인합니다. 이렇게 하려면 퍼블릭 서브넷에 대한 퍼블릭 IP 주소 자동 할당 옵션을 활성화하고 이를 Yes로 설정해야 합니다. 서브넷 설정 페이지에서 AWS Cloud9 환경을 만들기 전에 퍼블릭 서브넷에서 이 기능을 활성화할 수 있습니다. 퍼블릭 서브넷에서 자동 할당 IP 설정을 수정하는 단계는 Amazon VPC 사용 설명서의 서브넷에 대한 퍼블릭 IPv4 주소 지정 속성 수정을 참조하세요. 퍼블릭 서브넷과 프라이빗 서브넷에 대한 자세한 내용은 서브넷을 퍼블릭 또는 프라이빗으로 구성 섹션을 참조하세요.

> **Note**
>
> 다음 절차를 수행하려면 에 AWS Management Console 로그인하고 관리자 자격 증명을 사용하여 Amazon VPC 콘솔 (https://console.aws.amazon.com/vpc) 또는 Amazon EC2 콘솔 (https://console.aws.amazon.com/ec2) 을 여십시오.
>
> AWS CLI 또는 를 사용하는 경우 관리자 자격 증명을 AWS CloudShell 사용하여 AWS CLI 또는 를 구성하는 것이 좋습니다. AWS CloudShell AWS 계정이 작업을 수행할 수 없는 경우 AWS 계정 관리자에게 문의하세요.

AWS 리전의 VPC 목록 보기

Amazon VPC 콘솔을 사용하려면 AWS 탐색 표시줄에서 환경을 AWS 리전 AWS Cloud9 생성하는 콘솔을 선택합니다. 그런 다음 탐색 창에서 VPC를 선택합니다.

AWS CLI 또는 를 사용하려면 Amazon EC2 **describe-vpcs**명령을 실행합니다 (예: 다음과 같이). AWS CloudShell

```
aws ec2 describe-vpcs --output table --query 'Vpcs[*].VpcId' --region us-east-2
```

이전 명령에서 환경을 AWS 리전 AWS Cloud9 생성하는 us-east-2 명령으로 바꾸십시오. Windows에서 앞의 명령을 실행하려면 작은따옴표(' ')를 큰따옴표(" ")로 바꿉니다. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

출력에 VPC ID 목록이 포함되어 있습니다.

VPC의 서브넷 목록 보기

Amazon VPC 콘솔을 사용하려면 탐색 창에서 VPC를 선택합니다. [VPC ID] 열의 VPC ID를 적어 둡니다. 그런 다음 탐색 창에서 서브넷을 선택하고, VPC 열에서 해당 ID가 포함된 서브넷을 찾습니다.

AWS CLI 또는 를 사용하려면 Amazon EC2 **describe-subnets**명령을 실행합니다 (예: 다음과 같이). aws-shell

```
aws ec2 describe-subnets --output table --query 'Subnets[*].[SubnetId,VpcId]' --region
 us-east-2
```

이전 명령에서 서브넷이 AWS 리전 포함된 us-east-2 명령으로 대체하십시오. Windows에서 앞의 명령을 실행하려면 작은따옴표(' ')를 큰따옴표(" ")로 바꿉니다. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

출력에서 VPC ID와 일치하는 서브넷을 찾습니다.

서브넷이 퍼블릭인지 여부 확인

Important

환경의 EC2 인스턴스를 프라이빗 서브넷으로 시작한다고 가정해 보겠습니다. SSM 서비스에 연결할 수 있도록 해당 인스턴스에 대해 아웃바운드 트래픽이 허용되는지 확인합니다. 프라이빗 서브넷의 경우 일반적으로 아웃바운드 트래픽은 네트워크 주소 변환(NAT) 게이트웨이 또는 VPC 엔드포인트를 통해 구성됩니다. (NAT 게이트웨이에는 퍼블릭 서브넷이 필요합니다.) SSM에 액세스하기 위해 NAT 게이트웨이 대신 VPC 엔드포인트를 선택하는 경우를 가정해 보겠습니다. 인터넷 액세스가 필요한 인스턴스에 대한 자동 업데이트 및 보안 패치는 작동하지 않을 수 있습니다. AWS Systems Manager Patch Manager와 같은 다른 응용 프로그램을 사용

하여 사용자 환경에 필요할 수 있는 소프트웨어 업데이트를 관리할 수 있습니다. AWS Cloud9 소프트웨어는 정상적으로 업데이트됩니다.

Amazon VPC 콘솔을 사용하려면 탐색 창에서 [서브넷(Subnets)]을 선택합니다. AWS Cloud9 사용하려는 서브넷 옆의 상자를 선택합니다. [라우팅 테이블(Route Table)] 탭의 [대상(Target)] 열에 igw-로 시작하는 항목이 있는 경우 해당 서브넷은 퍼블릭입니다.

AWS CLI 또는 를 aws-shell 사용하려면 Amazon EC2 **describe-route-tables**명령을 실행합니다.

```
aws ec2 describe-route-tables --output table --query 'RouteTables[*].Routes[*].
{GatewayIds:GatewayId}' --region us-east-2 --filters Name=association.subnet-
id,Values=subnet-12a3456b
```

이전 명령에서 서브넷이 AWS 리전 포함된 us-east-2 명령으로 바꾸고 서브넷 subnet-12a3456b ID로 대체하십시오. Windows에서 앞의 명령을 실행하려면 작은따옴표(' ')를 큰따옴표(" ")로 바꿉니다. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

출력에서 igw-로 시작하는 결과가 하나 이상 있는 경우, 서브넷이 퍼블릭입니다.

출력에서 결과가 없는 경우 라우팅 테이블이 서브넷 대신에 VPC와 연결된 것일 수 있습니다. 이를 확인하려면 예를 들면 다음과 같이 서브넷 자체 대신에 서브넷의 관련 VPC에 대해 Amazon EC2 **describe-route-tables** 명령을 실행합니다.

```
aws ec2 describe-route-tables --output table --query 'RouteTables[*].Routes[*].
{GatewayIds:GatewayId}' --region us-east-1 --filters Name=vpc-id,Values=vpc-1234ab56
```

이전 명령에서 AWS 리전 VPC가 포함된 us-east-2 명령으로 바꾸고 VPC ID로 vpc-1234ab56 바꾸십시오. Windows에서 앞의 명령을 실행하려면 작은따옴표(' ')를 큰따옴표(" ")로 바꿉니다. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

출력에서 igw-로 시작하는 결과가 하나 이상 있는 경우, VPC에 인터넷 게이트웨이가 포함된 것입니다.

인터넷 게이트웨이의 설정 보기 또는 변경

Amazon VPC 콘솔을 사용하려면 탐색 창에서 인터넷 게이트웨이를 선택합니다. 해당 인터넷 게이트웨이 옆의 상자를 선택합니다. 설정을 확인하려면 각 탭을 봅니다. 탭에서 설정을 변경하려면 해당하는 경우 편집을 선택한 후, 화면 지침을 따릅니다.

AWS CLI 또는 를 사용하여 설정을 aws-shell 보려면 Amazon EC2 **describe-internet-gateways**명령을 실행합니다.

```
aws ec2 describe-internet-gateways --output table --region us-east-2 --internet-
gateway-id igw-1234ab5c
```

이전 명령에서 인터넷 게이트웨이가 포함된 AWS 리전 것으로 바꾸고 us-east-2 인터넷 게이트웨이 igw-1234ab5c ID로 바꾸십시오. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

인터넷 게이트웨이 생성

Amazon VPC 콘솔을 사용하려면 탐색 창에서 인터넷 게이트웨이를 선택합니다. Create internet gateway(인터넷 게이트웨이 생성)를 선택한 후, 화면 지침을 따릅니다.

AWS CLI 또는 를 aws-shell 사용하려면 Amazon EC2 **create-internet-gateway**명령을 실행합니다.

```
aws ec2 create-internet-gateway --output text --query
 'InternetGateway.InternetGatewayId' --region us-east-2
```

이전 명령에서 새 인터넷 게이트웨이가 AWS 리전 포함된 us-east-2 명령으로 대체하십시오. Windows에서 앞의 명령을 실행하려면 작은따옴표(' ')를 큰따옴표(" ")로 바꿉니다. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

출력에 새 인터넷 게이트웨이의 ID가 포함되어 있습니다.

VPC에 인터넷 게이트웨이 연결

Amazon VPC 콘솔을 사용하려면 탐색 창에서 인터넷 게이트웨이를 선택합니다. 해당 인터넷 게이트웨이 옆의 상자를 선택합니다. Actions, Attach to VPC(작업, VPC에 연결)을 선택한 후(사용 가능한 경우), 화면 지침을 따릅니다.

AWS CLI 또는 를 사용하려면 Amazon EC2 **attach-internet-gateway**명령을 실행합니다 (예: 다음과 같이). aws-shell

```
aws ec2 attach-internet-gateway --region us-east-2 --internet-gateway-id igw-a1b2cdef
 --vpc-id vpc-1234ab56
```

이전 명령에서 인터넷 게이트웨이가 AWS 리전 포함된 us-east-2 명령으로 바꾸십시오. igw-a1b2cdef를 인터넷 게이트웨이 ID로 바꿉니다. 그리고 vpc-1234ab56을 VPC ID로 바꾸세요. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

서브넷에 라우팅 테이블이 있는지 확인

Amazon VPC 콘솔을 사용하려면 탐색 창에서 [서브넷(Subnets)]을 선택합니다. AWS Cloud9 사용하려는 VPC의 퍼블릭 서브넷 옆에 있는 상자를 선택합니다. 라우팅 테이블 탭에서 라우팅 테이블에 값이 있는 경우, 퍼블릭 서브넷에 라우팅 테이블이 있는 것입니다.

AWS CLI 또는 를 aws-shell 사용하려면 Amazon EC2 **describe-route-tables**명령을 실행합니다.

```
aws ec2 describe-route-tables --output table --query 'RouteTables[*].Associations[*].
{RouteTableIds:RouteTableId}' --region us-east-2 --filters Name=association.subnet-
id,Values=subnet-12a3456b
```

이전 명령에서 퍼블릭 서브넷이 포함된 AWS 리전 것으로 바꾸고 us-east-2 퍼블릭 서브넷 subnet-12a3456b ID로 대체하십시오. Windows에서 앞의 명령을 실행하려면 작은따옴표(' ')를 큰따옴표(" ")로 바꿉니다. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

출력에 값이 있는 경우, 퍼블릭 서브넷에 라우팅 테이블이 하나 이상 있는 것입니다.

출력에서 결과가 없는 경우 라우팅 테이블이 서브넷 대신에 VPC와 연결된 것일 수 있습니다. 이를 확인하려면 예를 들면 다음과 같이 서브넷 자체 대신에 서브넷의 관련 VPC에 대해 Amazon EC2 **describe-route-tables** 명령을 실행합니다.

```
aws ec2 describe-route-tables --output table --query 'RouteTables[*].Associations[*].
{RouteTableIds:RouteTableId}' --region us-east-2 --filters Name=vpc-
id,Values=vpc-1234ab56
```

이전 명령에서 AWS 리전 VPC가 포함된 us-east-2 명령으로 바꾸고 VPC ID로 vpc-1234ab56 바꾸십시오. Windows에서 앞의 명령을 실행하려면 작은따옴표(' ')를 큰따옴표(" ")로 바꿉니다. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

출력에서 결과가 하나 이상 있는 경우, VPC에 라우팅 테이블이 하나 이상 있는 것입니다.

서브넷에 라우팅 테이블 연결

Amazon VPC 콘솔을 사용하려면 탐색 창에서 [라우팅 테이블(Route Tables)]을 선택합니다. 연결하려는 라우팅 테이블 옆의 상자를 선택합니다. 서브넷 연결 탭에서 편집을 선택하고, 연결하려는 서브넷 옆의 상자를 선택한 후, 저장을 선택합니다.

AWS CLI 또는 를 사용하려면 Amazon EC2 **associate-route-table**명령을 실행합니다 (예: 다음과 같이). aws-shell

```
aws ec2 associate-route-table --region us-east-2 --subnet-id subnet-12a3456b --route-
table-id rtb-ab12cde3
```

이전 명령에서 라우팅 테이블이 AWS 리전 포함된 us-east-2 명령으로 대체하십시오. subnet-12a3456b를 서브넷 ID로 바꿉니다. 그리고 rtb-ab12cde3을 라우팅 테이블 ID로 바꾸세요. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

라우팅 테이블 생성

Amazon VPC 콘솔을 사용하려면 탐색 창에서 [라우팅 테이블(Route Tables)]을 선택합니다. Create Route Table(라우팅 테이블 생성)을 선택한 후, 화면 지침을 따릅니다.

AWS CLI 또는 를 사용하려면 Amazon EC2 **create-route-table**명령을 실행합니다 (예: 다음과 같이). aws-shell

```
aws ec2 create-route-table --output text --query 'RouteTable.RouteTableId' --region us-
east-2 --vpc-id vpc-1234ab56
```

이전 명령에서 새 라우팅 테이블이 포함된 AWS 리전 것으로 바꾸고 VPC vpc-1234ab56 ID로 대체합니다. us-east-2 Windows에서 앞의 명령을 실행하려면 작은따옴표(' ')를 큰따옴표(" ")로 바꿉니다. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

출력에 새 라우팅 테이블의 ID가 포함되어 있습니다.

라우팅 테이블의 설정 보기 또는 변경

Amazon VPC 콘솔을 사용하려면 탐색 창에서 [라우팅 테이블(Route Tables)]을 선택합니다. 해당 라우팅 테이블 옆의 상자를 선택합니다. 설정을 확인하려면 각 탭을 봅니다. 탭에서 설정을 변경하려면 편집을 선택한 후, 화면 지침을 따릅니다.

AWS CLI 또는 를 사용하여 설정을 aws-shell 보려면 Amazon EC2 **describe-route-tables**명령을 실행합니다 (예: 다음과 같이).

```
aws ec2 describe-route-tables --output table --region us-east-2 --route-table-ids rtb-
ab12cde3
```

이전 명령에서 라우팅 테이블이 포함된 AWS 리전 것으로 바꾸고 us-east-2 라우팅 테이블 rtb-ab12cde3 ID로 바꾸십시오. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

에 대한 최소 권장 라우팅 테이블 설정 AWS Cloud9

대상	대상	상태	전파 완료
CIDR-BLOCK	로컬	활성	아니요
0.0.0.0/0	igw-INTERNET-GATEWAY-ID	활성	아니요

이러한 설정에서 *CIDR-BLOCK*은 서브넷의 CIDR 블록이고, igw-*INTERNET-GATEWAY-ID* 는 호환되는 인터넷 게이트웨이의 ID입니다.

VPC의 보안 그룹 목록 보기

Amazon VPC 콘솔을 사용하려면 탐색 창에서 [보안 그룹(Security Groups)]을 선택합니다. [보안 그룹 검색(Search Security Groups)] 상자에 VPC의 ID 또는 이름을 입력한 후, Enter 키를 누릅니다. 해당 VPC의 보안 그룹이 검색 결과 목록에 표시됩니다.

AWS CLI 또는 를 aws-shell 사용하려면 Amazon EC2 **describe-security-groups**명령을 실행합니다.

```
aws ec2 describe-security-groups --output table --query 'SecurityGroups[*].GroupId' --
region us-east-2 --filters Name=vpc-id,Values=vpc-1234ab56
```

이전 명령에서 AWS 리전 VPC가 포함된 us-east-2 명령으로 바꾸고 VPC ID로 vpc-1234ab56 바꾸십시오. Windows에서 앞의 명령을 실행하려면 작은따옴표(' ')를 큰따옴표(" ")로 바꿉니다. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

출력에 해당 VPC의 보안 그룹 ID 목록이 포함되어 있습니다.

컴퓨팅 인스턴스의 보안 그룹 목록 보기 AWS 클라우드

Amazon EC2 콘솔을 사용하려면 탐색 창에서 [인스턴스(Instances)]를 확장한 후, [인스턴스(Instances)]를 선택합니다. 인스턴스 목록에서 해당 인스턴스 옆의 상자를 선택합니다. 해당 인스턴스의 보안 그룹이 [보안 그룹(Security groups)] 옆에 있는 [설명(Description)] 탭에 표시됩니다.

AWS CLI 또는 를 사용하려면 Amazon EC2 **describe-security-groups**명령을 실행합니다 (예: 다음과 같이). aws-shell

```
aws ec2 describe-instances --output table --query
 'Reservations[*].Instances[*].NetworkInterfaces[*].Groups[*].GroupId' --region us-
east-2 --instance-ids i-12a3c456d789e0123
```

이전 명령에서는 인스턴스가 AWS 리전 포함된 `us-east-2` 명령으로 바꾸고 인스턴스 `i-12a3c456d789e0123` ID로 대체합니다. Windows에서 앞의 명령을 실행하려면 작은따옴표(' ')를 큰따옴표(" ")로 바꿉니다. `aws-shell`을 사용하여 앞의 명령을 실행하려면 `aws`를 뺍니다.

출력에 해당 인스턴스의 보안 그룹 ID 목록이 포함되어 있습니다.

VPC에 있는 보안 그룹의 설정 보기 또는 변경

Amazon VPC 콘솔을 사용하려면 탐색 창에서 [보안 그룹(Security Groups)]을 선택합니다. 해당 보안 그룹 옆의 상자를 선택합니다. 설정을 확인하려면 각 탭을 봅니다. 탭에서 설정을 변경하려면 해당하는 경우 편집을 선택한 후, 화면 지침을 따릅니다.

AWS CLI 또는 를 사용하여 설정을 `aws-shell` 보려면 Amazon EC2 **`describe-security-groups`**명령을 실행합니다 (예: 다음과 같이).

```
aws ec2 describe-security-groups --output table --region us-east-2 --group-ids
 sg-12a3b456
```

이전 명령에서는 인스턴스가 AWS 리전 포함된 `us-east-2` 명령으로 바꾸고 보안 그룹 `sg-12a3b456` ID로 대체합니다. `aws-shell`을 사용하여 앞의 명령을 실행하려면 `aws`를 뺍니다.

AWS 클라우드 컴퓨팅 인스턴스 보안 그룹의 설정 보기 또는 변경

Amazon EC2 콘솔을 사용하려면 탐색 창에서 [인스턴스(Instances)]를 확장한 후, [인스턴스(Instances)]를 선택합니다. 인스턴스 목록에서 해당 인스턴스 옆의 상자를 선택합니다. 설명 탭의 보안 그룹에서 보안 그룹을 선택합니다. 각 탭을 봅니다. 탭에서 설정을 변경하려면 해당하는 경우 편집을 선택한 후, 화면 지침을 따릅니다.

AWS CLI 또는 를 사용하여 설정을 `aws-shell` 보려면 Amazon EC2 **`describe-security-groups`**명령을 실행합니다 (예: 다음과 같이).

```
aws ec2 describe-security-groups --output table --region us-east-2 --group-ids
 sg-12a3b456
```

이전 명령에서는 인스턴스가 AWS 리전 포함된 `us-east-2` 명령으로 바꾸고 보안 그룹 `sg-12a3b456` ID로 대체합니다. `aws-shell`을 사용하여 앞의 명령을 실행하려면 `aws`를 뺍니다.

에 대한 최소 인바운드 및 아웃바운드 트래픽 설정 AWS Cloud9

> **Important**
>
> 인스턴스의 IA 보안 그룹에는 인바운드 규칙이 없을 수 있습니다. 이러한 경우에는 다른 호스트에서 시작하여 인스턴스로 전송되는 수신 트래픽이 허용되지 않습니다. 수신하지 않는 EC2 인스턴스 사용에 대한 자세한 내용은 AWS Systems Manager를 사용하여 수신하지 않는 EC2 인스턴스에 액세스 섹션을 참조하세요.

- 인바운드: 포트 22를 통해 SSH를 사용하는 모든 IP 주소. 하지만 이러한 IP 주소를 사용하는 IP 주소로만 제한할 수 있습니다. AWS Cloud9 자세한 정보는 AWS Cloud9의 인바운드 SSH IP 주소 범위을 참조하세요.

  > **Note**
  >
  > 2018년 7월 31일 또는 그 이후에 생성된 EC2 환경의 경우, 보안 그룹을 AWS Cloud9 사용하여 포트 22를 통한 SSH를 사용하여 인바운드 IP 주소를 제한합니다. 이러한 인바운드 IP 주소는 특별히 사용하는 주소일 뿐입니다. AWS Cloud9 자세한 정보는 AWS Cloud9의 인바운드 SSH IP 주소 범위을 참조하세요.

- 인바운드(네트워크 ACL만 해당): Amazon Linux 또는 Ubuntu Server를 실행하는 Amazon EC2 인스턴스와 연결된 EC2 환경과 SSH 환경의 경우, 모든 IP 주소는 포트 32768-61000을 통해 TCP를 사용합니다. 자세한 내용과 다른 Amazon EC2 인스턴스 유형의 포트 범위는 Amazon VPC 사용 설명서에서 휘발성 포트를 참조하세요.
- 아웃바운드: 모든 프로토콜과 포트를 사용하는 모든 트래픽 소스.

보안 그룹 수준에서 이 동작을 설정할 수 있습니다. 추가 보안 수준을 위해 네트워크 ACL을 사용할 수도 있습니다. 자세한 내용은 Amazon VPC 사용 설명서에서 보안 그룹 및 네트워크 ACL 비교를 참조하세요.

예를 들어 보안 그룹에 인바운드 및 아웃바운드 규칙을 추가하려면 그러한 규칙을 다음과 같이 설정할 수 있습니다.

인바운드 규칙

Type	프로토콜	포트 범위	소스
SSH(22)	TCP(6)	22	0.0.0.0(다음 메모 및 AWS Cloud9의 인바운드 SSH IP 주소 범위 단원 참조)

> **Note**
>
> 2018년 7월 31일 또는 그 이후에 생성된 EC2 환경의 경우, 포트 22를 통한 SSH를 사용하여 인바운드 IP 주소를 제한하는 인바운드 규칙을 AWS Cloud9 추가합니다. 이는 사용하는 주소로만 특별히 제한됩니다. AWS Cloud9 자세한 정보는 AWS Cloud9의 인바운드 SSH IP 주소 범위을 참조하세요.

아웃바운드 규칙

Type	프로토콜	포트 범위	소스
모든 트래픽	ALL	ALL	0.0.0.0/0

또한 네트워크 ACL에 인바운드 및 아웃바운드 규칙을 추가하기로 선택한 경우, 그러한 규칙을 다음과 같이 설정할 수 있습니다.

인바운드 규칙

규칙 #	Type	프로토콜	포트 범위	소스	허용/거부
100	SSH(22)	TCP(6)	22	0.0.0.0(AWS Cloud9의 인바운드 SSH IP 주소 범위 단원 참조)	ALLOW

규칙 #	Type	프로토콜	포트 범위	소스	허용/거부
200	사용자 지정 TCP 규칙	TCP(6)	32768-61000(Amazon Linux 및 Ubuntu Server 인스턴스의 경우. 다른 인스턴스 유형의 경우 휘발성 포트 참조)	0.0.0.0/0	ALLOW
*	모든 트래픽	ALL	ALL	0.0.0.0/0	DENY

아웃바운드 규칙

규칙 #	Type	프로토콜	포트 범위	소스	허용/거부
100	모든 트래픽	ALL	ALL	0.0.0.0/0	ALLOW
*	모든 트래픽	ALL	ALL	0.0.0.0/0	DENY

보안 그룹 및 네트워크 ACL에 대한 자세한 내용은 Amazon VPC 사용 설명서에서 다음을 참조하세요.

- 보안
- VPC의 보안 그룹
- 네트워크 ACL

VPC에 보안 그룹 생성

Amazon VPC 또는 Amazon EC2 콘솔을 사용하려면 다음 작업 중 하나를 수행합니다.

- Amazon VPC 콘솔의 탐색 창에서 [보안 그룹(Security Groups)]을 선택합니다. Create Security Group(보안 그룹 생성)을 선택한 후, 화면 지침을 따릅니다.

- Amazon EC2 콘솔의 탐색 창에서 [네트워크 및 보안(Network & Security)]을 확장한 후, [보안 그룹(Security Groups)]을 선택합니다. Create Security Group(보안 그룹 생성)을 선택한 후, 화면 지침을 따릅니다.

AWS CLI 또는 를 사용하려면 Amazon EC2 **create-security-group**명령을 실행합니다 (예: 다음과 같이). aws-shell

```
aws ec2 create-security-group --region us-east-2 --vpc-id vpc-1234ab56
```

이전 명령에서 AWS 리전 VPC가 포함된 us-east-2 명령으로 바꾸고 VPC ID로 vpc-1234ab56 바꾸십시오. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

VPC에 네트워크 ACL이 하나 이상 있는지 확인

Amazon VPC 콘솔을 사용하려면 탐색 창에서 VPC를 선택합니다. AWS Cloud9 사용하려는 VPC 옆의 상자를 선택합니다. [요약(Summary)] 탭에서 [네트워크 ACL(Network ACL)]에 값이 있는 경우 VPC에 네트워크 ACL이 하나 이상 있는 것입니다.

AWS CLI 또는 를 aws-shell 사용하려면 Amazon EC2 **describe-network-acls**명령을 실행합니다.

```
aws ec2 describe-network-acls --output table --query
 'NetworkAcls[*].Associations[*].NetworkAclId' --region us-east-2 --filters Name=vpc-
id,Values=vpc-1234ab56
```

이전 명령에서 AWS 리전 VPC가 포함된 us-east-2 명령으로 바꾸고 VPC ID로 vpc-1234ab56 바꾸십시오. Windows에서 앞의 명령을 실행하려면 작은따옴표(' ')를 큰따옴표(" ")로 바꿉니다. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

출력의 목록에 항목이 하나 이상 포함된 경우, VPC에 네트워크 ACL이 하나 이상 있는 것입니다.

VPC의 네트워크 ACL 목록 보기

Amazon VPC 콘솔을 사용하려면 탐색 창에서 [네트워크 ACL(Network ACLs)]을 선택합니다. [네트워크 ACL 검색(Search Network ACLs)] 상자에 VPC ID 또는 이름을 입력한 후, Enter 키를 누릅니다. 해당 VPC의 네트워크 ACL이 검색 결과 목록에 표시됩니다.

AWS CLI 또는 를 aws-shell 사용하려면 Amazon EC2 **describe-network-acls**명령을 실행합니다.

```
aws ec2 describe-network-acls --output table --query
 'NetworkAcls[*].Associations[*].NetworkAclId' --region us-east-2 --filters Name=vpc-
id,Values=vpc-1234ab56
```

이전 명령에서 AWS 리전 VPC가 포함된 us-east-2 명령으로 바꾸고 VPC ID로 vpc-1234ab56 바꾸십시오. Windows에서 앞의 명령을 실행하려면 작은따옴표(' ')를 큰따옴표(" ")로 바꿉니다. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

출력에 해당 VPC의 네트워크 ACL 목록이 포함되어 있습니다.

네트워크 ACL의 설정 보기 또는 변경

Amazon VPC 콘솔을 사용하려면 탐색 창에서 [네트워크 ACL(Network ACLs)]을 선택합니다. 해당 네트워크 ACL 옆의 상자를 선택합니다. 설정을 확인하려면 각 탭을 봅니다. 탭에서 설정을 변경하려면 해당하는 경우 [편집(Edit)]을 선택한 후, 화면 지침을 따릅니다.

AWS CLI 또는 를 사용하여 설정을 aws-shell 보려면 Amazon EC2 **describe-network-acls**명령을 실행합니다.

```
aws ec2 describe-network-acls --output table --region us-east-2 --network-acl-ids
 acl-1234ab56
```

이전 명령에서 네트워크 ACL이 포함된 AWS 리전 것으로 바꾸고 us-east-2 네트워크 ACL acl-1234ab56 ID로 바꾸십시오. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

네트워크 ACL을 생성

Amazon VPC 콘솔을 사용하려면 탐색 창에서 [네트워크 ACL(Network ACLs)]을 선택합니다. 네트워크 ACL 생성을 선택한 후, 화면 지침을 따릅니다.

AWS CLI 또는 를 aws-shell 사용하려면 Amazon EC2 **create-network-acl**명령을 실행합니다.

```
aws ec2 create-network-acl --region us-east-2 --vpc-id vpc-1234ab56
```

이전 명령에서 새 네트워크 ACL을 연결할 VPC가 AWS 리전 포함된 us-east-2 명령으로 바꾸십시오. 또한 vpc-1234ab56을 VPC ID로 바꿉니다. aws-shell을 사용하여 앞의 명령을 실행하려면 aws를 뺍니다.

VPC 및 기타 VPC 리소스 생성

다음 절차에 따라 애플리케이션을 실행하는 데 필요한 VPC와 추가 VPC 리소스를 생성합니다. VPC 리소스에는 서브넷, 라우팅 테이블, 인터넷 게이트웨이, NAT 게이트웨이가 포함됩니다.

콘솔을 사용하여 VPC, 서브넷 및 기타 VPC 리소스를 생성하려면

1. https://console.aws.amazon.com/vpc/에서 Amazon VPC 콘솔을 여세요.
2. VPC 대시보드에서 VPC 생성을 선택합니다.
3. 생성할 리소스에서 VPC 등을 선택합니다.
4. VPC 리소스의 이름 태그를 생성하려면 이름 태그 자동 생성을 선택한 상태로 유지합니다. VPC 리소스에 고유한 이름 태그를 제공하려면 이름을 지웁니다.
5. IPv4 CIDR 블록에 VPC의 IPv4 주소 범위를 입력해야 합니다. 의 권장 IPv4 범위는 입니다. AWS Cloud9 `10.0.0.0/16`
6. (선택 사항) IPv6 트래픽을 지원하려면 IPv6 CIDR 블록, Amazon에서 제공한 IPv6 CIDR 블록을 선택합니다.
7. 테넌시 옵션을 선택합니다. 이 옵션은 VPC로 시작하는 EC2 인스턴스가 다른 AWS 계정 과 공유되는 하드웨어에서 실행되는지 아니면 사용자 전용 하드웨어에서 실행되는지를 정의합니다. VPC의 테넌시를 `Default`로 선택하면 이 VPC로 시작된 EC2 인스턴스는 인스턴스를 시작할 때 지정된 테넌시 속성을 사용합니다. 자세한 내용은 Amazon EC2 사용 설명서의 정의된 파라미터를 사용하여 인스턴스 시작을 참조하십시오.

   VPC의 테넌시를 `Dedicated`로 선택하면 인스턴스는 항상 전용 하드웨어에서 전용 인스턴스로 실행됩니다. AWS Outposts를 사용하는 경우 Outpost에 프라이빗 연결이 필요하며, `Default` 테넌시를 사용해야 합니다.
8. 가용 영역(AZ) 수에서 프로덕션 환경의 경우 2개 이상의 Availability Zones에 서브넷을 프로비저닝하는 것이 좋습니다. 서브넷의 AZ를 선택하려면 AZ 사용자 지정을 확장합니다. 그렇지 않으면 AZ를 직접 AWS 선택해도 됩니다.
9. 서브넷을 구성하려면 퍼블릭 서브넷 수 및 프라이빗 서브넷 수의 값을 선택합니다. 서브넷의 IP 주소 범위를 선택하려면 서브넷 CIDR 블록 사용자 지정을 확장합니다. 그렇지 않으면, AWS 대신 선택해 드리겠습니다.
10. (선택 사항) 프라이빗 서브넷의 리소스가 IPv4를 통해 퍼블릭 인터넷에 액세스해야 하는 경우: NAT 게이트웨이에서 NAT 게이트웨이를 생성할 AZ 수를 선택합니다. 프로덕션 환경에서는 퍼블릭 인터넷에 액세스해야 하는 리소스가 있는 각 AZ에 NAT 게이트웨이를 배포하는 것이 좋습니다.
11. (선택 사항) 프라이빗 서브넷의 리소스가 IPv6을 통해 퍼블릭 인터넷에 액세스해야 하는 경우: 송신 전용 인터넷 게이트웨이에서 예를 선택합니다.

12. (선택 사항) VPC에서 직접 Amazon S3에 액세스하려면 VPC 엔드포인트, S3 게이트웨이를 선택합니다. 이는 Amazon S3 게이트웨이 VPC 엔드포인트를 생성합니다. 자세한 내용은 AWS PrivateLink 사용 설명서의 게이트웨이 VPC 엔드포인트를 참조하세요.
13. (선택 사항) DNS 옵션의 경우 도메인 이름 확인을 위한 두 가지 옵션이 기본적으로 모두 활성화됩니다. 기본값이 요구 사항을 충족하지 않는 경우 해당 옵션을 비활성화할 수 있습니다.
14. (선택 사항) VPC에 태그를 추가하려면 추가 태그를 확장하고 새 태그 추가를 선택하여 태그 키와 태그 값을 입력합니다.
15. 미리 보기 창에서 구성한 VPC 리소스 간의 관계를 시각화할 수 있습니다. 실선은 리소스 간의 관계를 나타냅니다. 점선은 NAT 게이트웨이, 인터넷 게이트웨이 및 게이트웨이 엔드포인트에 대한 네트워크 트래픽을 나타냅니다. VPC를 생성한 후에는 리소스 맵 탭을 사용하여 언제든지 VPC의 리소스를 이 형식으로 시각화할 수 있습니다.
16. VPC 구성을 마친 후 VPC 생성을 선택합니다.

VPC만 생성

다음 절차에 따라 Amazon VPC 콘솔을 사용하여 추가 VPC 리소스 없이 VPC를 생성합니다.

콘솔을 사용하여 추가 VPC 리소스 없이 VPC를 생성하려면

1. https://console.aws.amazon.com/vpc/에서 Amazon VPC 콘솔을 여세요.
2. VPC 대시보드에서 VPC 생성을 선택합니다.
3. 생성할 리소스에서 VPC 전용을 선택합니다.
4. (선택 사항) 이름 태그에 VPC의 이름을 입력합니다. 이렇게 하면 Name 키와 지정한 값으로 태그가 생성됩니다.
5. IPv4 CIDR 블록의 경우 다음 중 하나를 수행합니다.
   - IPv4 CIDR 수동 입력 을 선택하고 VPC에 대한 IPv4 주소 범위를 입력합니다. 권장 IPv4 범위는 입니다. AWS Cloud9 `10.0.0.0/16`
   - IPAM에서 할당된 IPv4 CIDR 블록을 선택하고, Amazon VPC IP 주소 관리자(IPAM) IPv4 주소 풀과 넷마스크를 선택합니다. CIDR 블록의 크기는 IPAM 풀의 할당 규칙에 의해 제한됩니다. IPAM은 워크로드의 IP 주소를 계획, 추적 및 모니터링하는 데 도움이 되는 VPC 기능입니다. AWS 자세한 내용은 Amazon Virtual Private Cloud 관리자 안내서의 IPAM이란 무엇인가요?를 참조하세요.

     IPAM을 사용하여 IP 주소를 관리하는 경우 이 옵션을 선택하는 것이 좋습니다. 그렇지 않으면 VPC에 지정한 CIDR 블록이 IPAM CIDR 할당과 겹칠 수 있습니다.

6. (선택 사항) 듀얼 스택 VPC를 생성하려면 VPC에 IPv6 주소 범위를 지정합니다. IPv6 CIDR 블록의 경우 다음 중 하나를 수행합니다.
   - IPAM에서 할당된 IPv6 CIDR 블록을 선택하고 IPAM IPv6 주소 풀을 선택합니다. CIDR 블록의 크기는 IPAM 풀의 할당 규칙에 의해 제한됩니다.
   - Amazon IPv6 주소 풀에서 IPv6 CIDR 블록을 요청하려면 Amazon에서 제공한 IPv6 CIDR 블록을 선택합니다. 네트워크 경계 그룹의 경우 IP 주소를 광고하는 그룹을 선택합니다. AWS Amazon은 /56의 고정 IPv6 CIDR 블록 크기를 제공합니다.
   - 내가 소유한 IPv6 CIDR을 선택하면 기존 IP 주소 가져오기 (BYOIP) 를 AWS 사용하여 가져온 IPv6 CIDR 블록을 사용할 수 있습니다. Pool(풀)에서 IPv6 CIDR 블록을 할당할 IPv6 주소 풀을 선택합니다.
7. (선택 사항) 테넌시 옵션을 선택합니다. 이 옵션은 VPC로 시작하는 EC2 인스턴스를 AWS 계정 다른 사람과 공유하는 하드웨어에서 실행할지, 아니면 사용자 전용 하드웨어에서 실행할지를 정의합니다. VPC의 테넌시를 `Default`로 선택하면 이 VPC로 시작된 EC2 인스턴스는 인스턴스를 시작할 때 지정된 테넌시 속성을 사용합니다. 자세한 내용은 Amazon EC2 사용 설명서의 정의된 파라미터를 사용하여 인스턴스 시작을 참조하십시오.

   VPC의 테넌시를 `Dedicated`로 선택하면 인스턴스는 항상 전용 하드웨어에서 전용 인스턴스로 실행됩니다. AWS Outposts를 사용하는 경우 Outpost에 프라이빗 연결이 필요하며, `Default` 테넌시를 사용해야 합니다.
8. (선택 사항) VPC에 태그를 추가하려면 새 태그 추가를 선택하고 태그 키와 태그 값을 입력합니다.
9. VPC 생성을 선택합니다.
10. VPC를 생성한 후 서브넷을 추가할 수 있습니다.

에 대한 서브넷을 생성하십시오. AWS Cloud9

Amazon VPC 콘솔을 사용하여 호환되는 VPC용 서브넷을 생성할 수 있습니다. AWS Cloud9 EC2 인스턴스의 프라이빗 서브넷이나 퍼블릭 서브넷을 생성할 수 있는지 여부는 환경이 인스턴스에 연결하는 방식에 따라 다릅니다.

- SSH를 통한 직접 액세스: 퍼블릭 서브넷 전용
- Systems Manager를 통한 액세스: 퍼블릭 또는 프라이빗 서브넷

환경의 EC2를 프라이빗 서브넷으로 시작하는 옵션은 콘솔, 명령줄 또는 AWS CloudFormation을 사용하여 '수신하지 않는' EC2 환경을 생성하는 경우에만 사용할 수 있습니다.

퍼블릭 또는 프라이빗으로 설정할 수 있는 서브넷을 생성하는 것과 동일한 단계를 따릅니다. 그리고 서브넷이 인터넷 게이트웨이로 가는 경로가 있는 라우팅 테이블과 연결되는 경우, 퍼블릭 서브넷이 됩니다. 반면, 서브넷이 인터넷 게이트웨이로 향하는 라우팅이 없는 라우팅 테이블과 연결되는 경우 프라이빗 서브넷이 됩니다. 자세한 내용은 서브넷을 퍼블릭 또는 프라이빗으로 구성 섹션을 참조하세요.

이전 절차에 따라 VPC를 만들었다면 이 절차도 따를 필요가 없습니다. AWS Cloud9새 VPC 생성 마법사가 서브넷을 자동으로 생성해 주기 때문입니다.

⚠ Important

- 환경과 동일한 AWS 리전 VPC가 이미 호환되는 VPC가 AWS 계정 있어야 합니다. 자세한 내용은 다음에 대한 Amazon VPC 요구 사항 AWS Cloud9 단원의 VPC 요구 사항을 참조하십시오.
- 이 절차에서는 IAM 관리자의 자격 증명을 사용하여 Amazon VPC 콘솔에 AWS Management Console 로그인하고 여는 것이 좋습니다. AWS 계정이 작업을 수행할 수 없는 경우 관리자에게 문의하십시오. AWS 계정
- 일부 조직에서는 서브넷을 직접 생성하는 것을 허용하지 않을 수 있습니다. 서브넷을 생성할 수 없는 경우 AWS 계정 관리자 또는 네트워크 관리자에게 문의하십시오.

서브넷을 생성하는 방법

1. Amazon VPC 콘솔이 아직 열려 있지 않은 경우 https://console.aws.amazon.com/vpc 에서 AWS Management Console 로그인하고 Amazon VPC 콘솔을 여십시오.
2. 탐색 표시줄에서 해당 환경의 지역과 동일하지 AWS 리전 않은 경우 올바른 지역을 선택합니다.
3. [서브넷(Subnets)] 페이지가 표시되어 있지 않으면 탐색 창에서 [서브넷(Subnets)]을 선택합니다.
4. 서브넷 생성을 선택합니다.
5. [서브넷 생성(Create Subnet)] 대화 상자의 [이름 태그(Name tag)]에 서브넷의 이름을 입력합니다.
6. VPC에서 서브넷을 연결할 VPC를 선택합니다.
7. 가용 영역의 경우 서브넷의 가용 영역 내에서 사용할 AWS 리전 가용 영역을 선택하거나 No Preference (기본 설정 없음) 를 선택하여 가용 영역을 AWS 선택할 수 있습니다.
8. [IPv4 CIDR 블록(IPv4 CIDR block)]에 사용할 서브넷의 IP 주소 범위를 CIDR 형식으로 입력합니다. 이 IP 주소 범위는 VPC에 있는 IP 주소의 하위 집합이어야 합니다.

CIDR 블록에 대한 자세한 내용은 Amazon VPC 사용 설명서에서 VPC 및 서브넷 크기 조정을 참조하세요. 또한 3.1을 참조하세요. RFC 4632의 Basic Concept and Prefix Notation 또는 Wikipedia의 IPv4 CIDR blocks를 참조하세요.

서브넷을 생성한 후 퍼블릭 또는 프라이빗 서브넷으로 구성합니다.

서브넷을 퍼블릭 또는 프라이빗으로 구성

서브넷을 생성한 후 인터넷과 통신하는 방법을 지정하여 서브넷을 퍼블릭 또는 프라이빗으로 만들 수 있습니다.

퍼블릭 서브넷에는 퍼블릭 IP 주소가 있으며, 인터넷 게이트웨이(IGW)가 연결되어 서브넷과 인터넷 및 기타 AWS 서비스의 인스턴스 간 통신을 지원합니다.

프라이빗 서브넷의 인스턴스에는 프라이빗 IP 주소가 있으며, 서브넷과 인터넷의 인스턴스와 기타 AWS 서비스간에 트래픽을 주고받는 데 Network Address Translation(NAT) 게이트웨이가 사용됩니다. NAT 게이트웨이는 퍼블릭 서브넷에서 호스트되어야 합니다.

Public subnets

> **Note**
>
> 환경의 인스턴스가 프라이빗 서브넷에서 시작되더라도 VPC PC에는 하나 이상의 퍼블릭 서브넷이 있어야 합니다. 이는 인스턴스에서 주고받는 트래픽을 전달하는 NAT 게이트웨이가 퍼블릭 서브넷에서 호스트되어야 하기 때문입니다.

서브넷을 퍼블릭으로 구성하려면 인터넷 게이트웨이(IGW)를 연결하고, 해당 IGW에 대한 경로를 지정하도록 라우팅 테이블을 구성하고, 보안 그룹의 설정을 정의하여 인바운드 및 아웃바운드 트래픽을 제어합니다.

이러한 태스크를 수행하는 방법에 대한 지침은 VPC 및 기타 VPC 리소스 생성에서 참조할 수 있습니다.

> **Important**
>
> 개발 환경에서 SSM을 사용하여 EC2 인스턴스에 액세스하는 경우 인스턴스가 시작된 퍼블릭 서브넷에 의해 인스턴스에 퍼블릭 IP 주소가 할당되었는지 확인합니다. 이렇게 하려면 퍼블릭 서브넷에 대한 퍼블릭 IP 주소 자동 할당 옵션을 활성화하고 이를 Yes로 설정해

야 합니다. 서브넷 설정 페이지에서 AWS Cloud9 환경을 만들기 전에 퍼블릭 서브넷에서 이 기능을 활성화할 수 있습니다. 퍼블릭 서브넷에서 자동 할당 IP 설정을 수정하는 단계는 Amazon VPC 사용 설명서의 서브넷에 대한 퍼블릭 IPv4 주소 지정 속성 수정을 참조하세요. 퍼블릭 서브넷과 프라이빗 서브넷에 대한 자세한 내용은 서브넷을 퍼블릭 또는 프라이빗으로 구성 섹션을 참조하세요.

Private subnets

Systems Manager를 통해 액세스하는 수신하지 않는 인스턴스를 생성하는 경우 프라이빗 서브넷으로 시작할 수 있습니다. 프라이빗 서브넷에는 퍼블릭 IP 주소가 없습니다. 따라서 프라이빗 IP 주소를 요청의 퍼블릭 주소에 매핑하려면 NAT 게이트웨이가 필요하며 퍼블릭 IP 주소를 응답의 프라이빗 주소로 다시 매핑해야 합니다.

⚠ Warning

계정에서 NAT 게이트웨이 생성 및 사용에 대한 요금이 청구됩니다. NAT 게이트웨이 시간당 사용 요금 및 데이터 처리 요금이 적용됩니다. Amazon EC2 데이터 전송 요금도 적용됩니다. 자세한 내용은 Amazon VPC 요금을 참조하십시오.

NAT 게이트웨이를 생성하고 구성하기 전에 다음을 수행해야 합니다.

- 퍼블릭 VPC 서브넷을 생성하여 NAT 게이트웨이를 호스트합니다.
- NAT 게이트웨이에 할당할 수 있는 탄력적 IP 주소를 프로비저닝합니다.
- 프라이빗 서브넷의 경우 [퍼블릭 IPv4 주소 자동 할당 사용(Enable auto-assign public IPv4 address)] 확인란을 선택 취소하여 해당 인스턴스에 프라이빗 IP 주소가 할당되도록 합니다. 자세한 내용은 Amazon VPC 사용 설명서의 VPC에서 IP 주소 지정 을 참조하세요.

이 태스크의 단계에 대한 자세한 내용은 Amazon VPC 사용 설명서에서 NAT 게이트웨이 작업을 참조하세요.

⚠ Important

현재 환경의 EC2 인스턴스를 프라이빗 서브넷으로 시작하는 경우 AWS 관리형 임시 자격 증명을 사용하여 IAM 사용자와 같은 AWS 엔티티를 AWS 서비스 대신하여 EC2 환경에 액세스하도록 허용할 수 없습니다.

SSH 환경 호스트 요구 사항

환경을 기존 클라우드 컴퓨팅 인스턴스 또는 자체 서버에 AWS Cloud9 연결하도록 지시하려면 AWS Cloud9 SSH 개발 환경을 생성해야 합니다. 그러나 SSH 환경을 만들기 전에 EC2 환경을 생성하는 데 따른 이점을 고려하세요.

EC2 환경을 생성할 때는 AWS Cloud9 이 새 환경을 생성하고 새 인스턴스를 시작하도록 Amazon EC2에 요청한 후, 새로 시작된 인스턴스를 새 환경에 연결합니다. EC2 환경을 생성하면 다음과 같은 장점이 있습니다.

- 자동 인스턴스 시작. EC2 환경을 생성할 때 Amazon EC2에 동시에 새 인스턴스를 생성하도록 AWS Cloud9 요청합니다. SSH 환경에서 기존 클라우드 컴퓨팅 인스턴스(예: Amazon EC2 인스턴스) 또는 자체 서버를 직접 제공해야 합니다.
- 자동 인스턴스 종료. 기본적으로 AWS Cloud9 은 EC2 환경의 IDE에 연결된 모든 웹 브라우저 인스턴스가 닫히고 30분 후에 EC2 환경을 자동으로 종료합니다. (이 동작을 언제든 변경할 수 있습니다.) 이렇게 하면 AWS 계정 에 Amazon EC2 사용에 따른 추가 요금이 부과될 가능성을 줄일 수 있습니다.
- 자동 인스턴스 정리. EC2 환경을 삭제하면 연결된 Amazon EC2 인스턴스가 자동으로 삭제됩니다. 또한 Amazon EC2 사용에 AWS 계정 대한 추가 요금이 부과될 가능성을 줄이는 데도 도움이 됩니다. 클라우드 컴퓨팅 인스턴스에 연결된 SSH 환경에서는 인스턴스를 직접 삭제해야 합니다.
- AWS 관리형 임시 자격 증명. EC2 환경에서는 호출자의 모든 AWS 리소스에 대한 모든 AWS 작업을 쉽게 켜거나 끌 수 있습니다 AWS 계정 (일부 제한 있음). 환경의 Amazon EC2 인스턴스에 대한 인스턴스 프로필을 구성하거나 AWS 개체 (예: IAM 사용자) 의 영구 AWS 액세스 자격 증명을 저장할 필요가 없습니다.

  자세한 정보는 AWS 관리형 임시 자격 증명을 참조하세요.
- AWS 툴킷 및 Git 패널 시각적 소스 제어와 상호 AWS 서비스 작용하고 사용하기 위한 이러한 도구는 Amazon EC2 인스턴스로 생성된 AWS Cloud9 환경에서만 사용할 수 있습니다.

대신 EC2 환경을 생성하려면 EC2 환경 생성 섹션을 참조하세요. 그렇지 않으면 SSH 환경 생성에 대한 정보를 계속 읽으세요.

SSH 환경을 생성하는 시나리오와 방법

다음과 같은 요구 사항이 있을 때마다 EC2 환경 대신 SSH 환경을 만들어야 합니다.

요구 사항	지침
컴퓨팅 인스턴스 사용에 AWS 계정 AWS 클라우드 대해 추가 요금이 부과되는 것은 바람직하지 않습니다. 따라서 대신 자체 서버 AWS 또는 외부 기존 클라우드 컴퓨팅 인스턴스에 AWS Cloud9 연결하기로 결정합니다.	1. 인스턴스 또는 서버가 이 주제의 뒷부분에서 설명하는 요구 사항을 충족하는지 확인합니다. 2. 인스턴스 또는 서버를 연결할 AWS Cloud9 용 SSH 환경을 생성합니다.
환경 생성과 동시에 새 인스턴스를 시작하는 AWS 계정 대신 기존 AWS 클라우드 컴퓨팅 인스턴스 (예: Amazon EC2 인스턴스) AWS Cloud9 를 사용하고 싶습니다.	1. 인스턴스가 이 주제의 뒷부분에서 설명하는 요구 사항을 충족하는지 확인합니다. 2. SSH 환경용 AWS Cloud9 를 클릭하여 인스턴스를 에 연결합니다.
AWS Cloud9 현재 EC2 환경을 지원하지 않는 Amazon EC2 인스턴스 유형 (예:) 을 사용하려고 합니다. R4	1. 원하는 인스턴스 유형에 따라 Amazon EC2 인스턴스를 시작합니다. 또는 원하는 인스턴스 유형을 AWS 계정 실행하는 기존 인스턴스를 식별하십시오. 2. 인스턴스가 이 주제의 뒷부분에서 설명하는 요구 사항을 충족하는지 확인합니다. 3. SSH 환경용 AWS Cloud9 를 클릭하여 인스턴스를 에 연결합니다.
Amazon Linux 또는 Ubuntu Server가 아닌 Amazon Machine Image(AMI)를 기반으로 하는 Amazon EC2 인스턴스를 사용하고자 합니다.	1. 원하는 AMI에 따라 Amazon EC2 인스턴스를 시작합니다. 또는 원하는 AMI를 기반으로 AWS 계정 하는 기존 인스턴스를 식별하십시오. 2. 인스턴스가 이 주제의 뒷부분에서 설명하는 요구 사항을 충족하는지 확인합니다. 3. SSH 환경용 AWS Cloud9 를 클릭하여 인스턴스를 에 연결합니다.
여러 환경을 기존의 단일 클라우드 컴퓨팅 인스턴스 또는 자체 서버에 연결하고자 합니다.	1. 인스턴스 또는 서버가 이 주제의 뒷부분에서 설명하는 요구 사항을 충족하는지 확인합니다.

요구 사항	지침
	2. 인스턴스 또는 서버를 AWS Cloud9 연결하려는 각 환경에 대한 SSH 환경을 생성합니다.

> **Note**
>
> Amazon EC2 인스턴스를 시작하면 AWS 계정 에 Amazon EC2에 대한 요금이 부과될 수 있습니다. 자세한 내용은 Amazon EC2 요금을 참조하세요.

SSH 호스트 요구 사항

기존 클라우드 컴퓨팅 인스턴스 또는 자체 서버가 SSH 환경에 AWS Cloud9 연결하려면 다음 요구 사항을 충족해야 합니다.

- Linux를 실행해야 합니다. (윈도우는AWS Cloud9 지원하지 않습니다.)
- Arm 기반 아키텍처를 사용해서는 안 됩니다. (Arm 프로세서를 기반으로 구축된 시스템에 대한 지원은 검토 중입니다.)
- SSH를 사용하여 퍼블릭 인터넷을 통해 접근 가능해야 합니다. 가상 프라이빗 클라우드(VPC) 또는 가상 프라이빗 네트워크(VPN)를 통해서만 접근 가능한 경우, 해당 VPC 또는 VPN이 퍼블릭 인터넷에 액세스할 수 있어야 합니다.
- 호스트가 Amazon VPC (가상 사설 AWS 클라우드) 의 일부인 기존 클라우드 컴퓨팅 인스턴스인 경우 추가 요구 사항이 있습니다. 자세한 내용은 Amazon VPC 설정을 참조하십시오.
- Python3설치 시점과 기본 Python 버전으로 설치 AWS Cloud9및 pip3 설정되어 있어야 합니다. 버전을 확인하려면 기존 인스턴스 또는 서버의 터미널에서 **`python --version`** 명령을 실행합니다. 인스턴스 또는 서버에 Python을 설치하려면 다음 리소스 중 하나를 참조하세요.
  - Python 샘플에서 1단계: 필수 도구를 설치합니다.
  - Python 웹사이트에서 Python을 다운로드하십시오.

> **Note**
>
> 기존 AWS 클라우드 컴퓨팅 인스턴스에 연결하여 요구 사항을 확인하고 충족하려면 다음 리소스 중 하나 이상을 참조하십시오.

- Amazon EC2의 경우 Amazon EC2 사용 설명서의 Linux 인스턴스에 연결을 참조하십시오.
- Amazon Lightsail의 경우 Amazon Lightsail 문서에서 Linux/Unix 기반 Lightsail 인스턴스에 연결을 참조하세요.
- 에 대해서는 AWS Elastic BeanstalkAWS Elastic Beanstalk 개발자 안내서의 서버 인스턴스 나열 및 연결을 참조하십시오.
- 에 대해서는 AWS OpsWorks사용 설명서의 SSH를 사용하여 Linux 인스턴스에 로그인하는 방법을 참조하십시오.AWS OpsWorks
- 기타 AWS 서비스정보는 서비스 설명서를 참조하십시오.

자체 서버에 연결하여 요구 사항을 확인 및 충족하려면 'SSH 명령을 사용하여 서버에 연결'(macOS 또는 Linux) 또는 'PuTTY를 사용하여 서버에 연결'(Windows)과 같은 단계를 사용하여 인터넷을 검색하세요.

- 모든 필수 패키지를 설치하려면 다음 명령을 실행합니다.

  Amazon Linux의 경우:

  ```
  sudo yum install -y make glibc-devel gcc gcc-c++
  ```

  Ubuntu Server:

  ```
  sudo apt install build-essential
  ```

- 이 경우 Node.js가 설치되어 있어야 합니다. 호스트 운영 체제에서 지원하는 최신 Node.js 버전을 설치하는 것이 좋습니다.

  > **Warning**
  >
  > AWS Cloud9 에서 지원하지 않는 Node.js 버전을 사용하는 경우 SSH 환경을 만들 때 설치 문제가 발생할 수 있습니다. AWS Cloud9

  버전을 확인하려면 기존 인스턴스 또는 서버의 터미널에서 **`node --version`** 명령을 실행합니다. 인스턴스 또는 서버에 Node.js를 설치하려면 다음 리소스 중 하나를 참조하세요.

  - Node.js 샘플의 1단계: 필수 도구 설치
  - Node.js 웹 사이트의 패키지 관리자를 통해 Node.js 설치

  - 노드 버전 관리자가 켜져 있습니다. GitHub
- 로그인 후부터 AWS Cloud9 가 시작할 기존 인스턴스 또는 서버의 디렉터리 경로는 액세스 권한이 `rwxr-xr-x`로 설정되어 있어야 합니다. 즉, 환경 만들기 마법사의 설정 구성 페이지에서 사용자를 위한 로그인 이름에 해당하는 소유자 read-write-run 권한, 이 소유자가 속한 그룹에 대한 읽기 실행 권한 및 다른 사용자의 읽기 실행 권한을 의미합니다.

  예를 들어 디렉터리의 경로가 ~인 경우(여기서 ~는 Configure settings(구성 설정) 페이지에서 User(사용자)에 지정한 로그인 이름의 홈 디렉터리를 나타냄) 다음 명령 및 이어지는 지시문을 사용하여 인스턴스 또는 서버에 대해 **chmod** 명령을 실행함으로써 디렉터리에 이러한 권한을 설정할 수 있습니다.

```
sudo chmod u=rwx,g=rx,o=rx ~
```

- 기존 인스턴스 또는 서버에 AWS Cloud9 설치 관리자를 다운로드하고 실행합니다.
- 선택적으로 SSH를 통한 인바운드 트래픽을 사용하는 IP 주소로만 제한할 수 있습니다. AWS Cloud9 이를 수행하려면 AWS Cloud9의 인바운드 SSH IP 주소 범위에 설명된 대로 인바운드 SSH 트래픽을 해당 IP 범위로 설정합니다.

인스턴스 또는 서버가 위의 요구 사항을 충족하는지 확인한 후 AWS Cloud9 연결할 SSH 환경을 만드십시오.

AWS Cloud9 SSH 환경용 AWS Cloud9 설치 프로그램 사용

AWS Cloud9 SSH 개발 환경을 만들기 전에 환경에 연결할 클라우드 컴퓨팅 인스턴스(예: Amazon EC2 인스턴스) 또는 자체 서버가 SSH 호스트 요구 사항을 충족해야 합니다. 이러한 요구 사항 중 하나는 인스턴스 또는 서버에 AWS Cloud9 설치 관리자를 다운로드하고 실행해야 한다는 것입니다. AWS Cloud9 설치 프로그램은 인스턴스 또는 서버가 AWS Cloud9가 지원하는 운영 체제 플랫폼 및 아키텍처에서 실행되는지 여부를 확인하는 Linux 셸 스크립트입니다. 이 확인이 성공하면 이 스크립트는 AWS Cloud9를 사용하려면 인스턴스 또는 서버에 있어야 하는 구성 요소 및 종속 항목을 설치하려고 시도합니다.

이 주제에서는 대상 인스턴스 또는 서버에 설치 프로그램 스크립트를 다운로드하고 실행하는 방법을 설명합니다.

- AWS Cloud9 설치 프로그램 다운로드 및 실행
- AWS Cloud9 설치 프로그램 문제 해결

AWS Cloud9 설치 프로그램 다운로드 및 실행

1. 환경에 연결할 클라우드 컴퓨팅 인스턴스 또는 자체 서버가 SSH 호스트 요구 사항을 충족하는지 확인합니다. 이러한 요구 사항에는 특정 버전의 Python 및 Node.js 설치, 로그인 후 AWS Cloud9이 시작하도록 할 디렉터리에 대한 특정 권한 설정, 연결된 Amazon Virtual Private Cloud 설정 등이 포함됩니다.
2. 인스턴스 또는 서버에 연결되어 있는 동안 해당 인스턴스 또는 서버에서 다음 명령 중 하나를 실행합니다. 명령 중 하나를 실행하기 전에 `gcc`를 설치해야 합니다.

   ```
   curl -L https://d3kgj69l4ph6w4.cloudfront.net/static/c9-install-2.0.0.sh | bash
   wget -O - https://d3kgj69l4ph6w4.cloudfront.net/static/c9-install-2.0.0.sh | bash
   ```

3. 오류 없이 완료 메시지가 표시되면 SSH 환경을 생성할 수 있습니다.

   오류 메시지가 표시될 경우, 다음 단원에서 문제 해결 정보를 참조하십시오.

AWS Cloud9 설치 프로그램 문제 해결

이 단원에서는 일반적인 문제, 가능한 원인, AWS Cloud9 설치 프로그램 오류를 해결하기 위한 권장 솔루션을 설명합니다.

해당 문제가 나와 있지 않거나 추가 도움이 필요한 경우, AWS Cloud9 토론 포럼을 참조하세요. (이 포럼에 들어갈 때 AWS에서 로그인을 요청할 수 있습니다.) 또한 직접 당사에 문의할 수도 있습니다.

- -bash: wget: command not found
- Error: please install make to proceed
- Error: please install gcc to proceed
- configure: error: curses not found

-bash: wget: command not found

문제: 설치 프로그램 스크립트를 실행하면 `-bash: wget: command not found` 메시지가 표시됩니다.

가능한 원인: 인스턴스 또는 서버에 **`wget`** 유틸리티가 설치되지 않았습니다.

권장 솔루션: 대신에 **`curl`** 유틸리티를 사용하여 인스턴스 또는 서버에서 설치 프로그램 스크립트를 실행하십시오.

Error: please install make to proceed

문제: 설치 프로그램 스크립트를 실행하면 `Error: please install make to proceed` 메시지가 표시됩니다.

가능한 원인: 인스턴스 또는 서버에 **make** 유틸리티가 설치되지 않았습니다.

권장 솔루션: **make** 유틸리티를 설치한 후, 인스턴스 또는 서버에서 설치 프로그램 스크립트를 다시 실행해 보십시오.

make 유틸리티를 설치하려면 인스턴스 또는 서버에서 다음 명령 중 하나를 실행합니다.

- Amazon EC2에서 실행 중인 Amazon Linux, Amazon Linux 2 및 Red Hat Enterprise Linux(RHEL)의 경우: **sudo yum -y groupinstall "Development Tools"**
- Amazon EC2에서 실행 중인 Ubuntu Server의 경우: **sudo apt install -y build-essential**
- SUSE의 경우: **sudo zypper install -y make**

Error: please install gcc to proceed

문제: 설치 프로그램 스크립트를 실행하면 `Error: please install gcc to proceed` 메시지가 표시됩니다.

가능한 원인: 인스턴스 또는 서버에 **gcc** 유틸리티가 설치되지 않았습니다.

권장 솔루션: **gcc** 유틸리티를 설치한 후, 인스턴스 또는 서버에서 설치 프로그램 스크립트를 다시 실행해 보십시오.

gcc 유틸리티를 설치하려면 인스턴스 또는 서버에서 다음 명령 중 하나를 실행합니다.

- Amazon EC2에서 실행 중인 Amazon Linux, Amazon Linux 2 및 Red Hat Enterprise Linux(RHEL)의 경우: **sudo yum -y groupinstall "Development Tools"**
- Amazon EC2에서 실행 중인 Ubuntu Server의 경우: **sudo apt install -y build-essential**
- SUSE의 경우: **sudo zypper install -y gcc**
- 다른 운영 체제의 경우 Installing GCC 참조

configure: error: curses not found

문제: 설치 프로그램 스크립트를 실행하면 `configure: error: curses not found` 메시지가 표시됩니다.

가능한 원인: 인스턴스 또는 서버에 **ncurses** 터미널 관리 라이브러리가 설치되지 않았습니다.

권장 솔루션: **ncurses** 터미널 관리 라이브러리(일부 운영 체제의 경우, **glibc-static** 라이브러리)를 설치한 후, 인스턴스 또는 서버에서 설치 프로그램 스크립트를 다시 실행해 보십시오.

ncurses 터미널 관리 라이브러리(일부 운영 체제의 경우, **glibc-static** 라이브러리)를 설치하려면 인스턴스 또는 서버에서 다음 명령 중 하나를 실행합니다.

- Amazon EC2에서 실행 중인 Amazon Linux, Amazon Linux 2 및 Red Hat Enterprise Linux(RHEL)의 경우: **sudo yum -y install ncurses-devel**
- SUSE의 경우: **sudo zypper install -y ncurses-devel** 및 **sudo zypper install -y glibc-static**

AWS Cloud9의 인바운드 SSH IP 주소 범위

AWS Cloud9이 SSH를 통해 Amazon VPC의 AWS 클라우드 컴퓨팅 인스턴스(예: Amazon EC2 인스턴스) 또는 네트워크의 자체 서버에 연결하기 위해 사용하는 IP 주소 범위로만 수신 트래픽을 제한할 수 있습니다.

> **Note**
>
> 수신 트래픽을 AWS Cloud9에서 SSH를 통해 연결하는 데 사용하는 IP 주소 범위로만 제한할 수 있습니다. 2018년 7월 31일 이후에 생성된 EC2 환경의 경우, 이 주제를 건너뛸 수 있습니다. AWS Cloud9이 해당 환경의 인바운드 SSH 트래픽을 이 주제의 뒷부분에서 설명하는 IP 주소로만 자동으로 제한하기 때문입니다. AWS Cloud9은 환경의 Amazon EC2 인스턴스와 연결된 보안 그룹에 규칙을 자동으로 추가하여 이를 수행합니다. 이 규칙은 포트 22를 통한 인바운드 SSH 트래픽을 연결된 AWS 리전의 IP 주소로만 제한합니다. 네트워크에 있는 자체 서버도 이 주제의 뒷부분에서 설명하는 단계를 따라야 합니다.

대부분 AWS 리전의 IP 주소 범위는 AWS 일반 참조의 AWS IP 주소 범위에서 설명하는 것처럼 `ip-ranges.json` 파일에 있습니다.

> **Note**
>
> 현재 `ip-ranges.json` 파일에 포함되어 있지 않은 아시아 태평양(홍콩), EU(밀라노) 및 중동(바레인) 리전의 IP 주소 범위는 아래를 참조하세요.

ip-ranges.json 파일에서 IP 범위를 찾으려면:

- Windows의 경우 AWS Tools for Windows PowerShell를 사용하여 다음 명령을 실행합니다.

```
Get-AWSPublicIpAddressRange -ServiceKey CLOUD9
```

- Linux의 경우 ip-ranges.json 파일을 다운로드합니다. 그리고 다음 명령을 실행하여 **jq** 등의 도구로 쿼리할 수 있습니다.

```
jq '.prefixes[] | select(.service=="CLOUD9")' < ip-ranges.json
```

이러한 IP 주소는 경우에 따라 변경할 수 있습니다. 변경될 때마다 AmazonIpSpaceChanged 주제의 구독자에게 알림을 전송합니다. 이러한 알림을 받으려면 AWS 일반 참조의 AWS IP 주소 범위 알림을 참조하세요.

AWS 클라우드 컴퓨팅 인스턴스를 사용하는 환경을 구성할 때 이러한 IP 주소 범위를 사용하려면 개발 환경을 위한 AWS Cloud9 VPC 설정 섹션을 참조하세요. 또한 Amazon Linux 또는 Ubuntu Server를 실행하는 Amazon EC2 인스턴스와 연결된 EC2 환경 또는 SSH 환경의 수신 트래픽을 제한하려면, 최소한 포트 32768-61000을 통한 TCP를 사용하는 모든 IP 주소를 허용해야 합니다. 자세한 내용과 다른 AWS 클라우드 컴퓨팅 인스턴스 유형의 포트 범위는 Amazon VPC 사용 설명서에서 휘발성 포트를 참조하세요.

자체 네트워크를 사용하는 SSH 환경을 구성할 때 이러한 IP 주소 범위를 사용하려면 해당 네트워크의 문서를 참조하거나 네트워크 관리자에게 문의하세요.

ip-ranges.json에 없는 IP 주소

아시아 태평양(홍콩), EU(밀라노) 및 중동(바레인) AWS 리전의 AWS Cloud9 IP 주소 범위는 현재 ip-ranges.json 파일에서 제공되지 않습니다. 다음 표에 해당 리전의 IP 범위가 나와 있습니다.

> **Note**
>
> 각 리전에는 AWS Cloud9 제어 플레인(정보 라우팅)과 데이터 영역(정보 처리) 서비스를 지원하는 두 개의 IP 주소 범위가 있습니다.

AWS 리전	코드	IP 범위(CIDR 표기법)
아시아 태평양(홍콩)	ap-east1	18.163.201.96/27 18.163.139.32/27
유럽(밀라노)	eu-south-1	15.161.135.64/27 15.161.135.96/27
중동(바레인)	me-south-1	15.185.141.160/27 15.185.91.32/27

AWS Cloud9 EC2 개발 환경의 Amazon Machine Image(AMI) 콘텐츠

다음 정보를 사용하여 AWS Cloud9 에서 EC2 환경에 사용하는 Amazon 머신 이미지(AMI)에 대한 세부 정보를 가져옵니다.

> **Important**
>
> 해당 환경의 Amazon EC2 인스턴스가 Amazon Linux 2023 AMI 또는 Amazon Linux 2 AMI 템플릿을 기반으로 하는 경우 보안 업데이트가 시작된 직후 인스턴스에 설치됩니다. 그리고 보안 패치는 매시간 인스턴스에 자동으로 적용됩니다. 이러한 업데이트는 백그라운드 프로세스에 의해 적용되며 인스턴스 사용에 영향을 주지 않습니다.
> Ubuntu EC2 환경의 경우 보안 업데이트가 시작된 직후 인스턴스에 보안 업데이트도 설치됩니다. 그런 다음 `unattended-upgrades` 패키지는 매일 사용 가능한 업데이트를 자동으로 설치합니다.

주제

- Amazon Linux 2023 및 Amazon Linux 2
- Ubuntu 서버

Amazon Linux 2023 및 Amazon Linux 2

> **Important**
>
> 콘솔을 사용하여 Amazon EC2 환경 생성할 때는 Amazon Linux 2023 옵션을 선택하는 것이 좋습니다. Amazon Linux 2023 AMI는 안전하고 안정적인 고성능 런타임 환경을 제공할 뿐만 아니라 2024년까지 장기적인 지원을 제공합니다.

Amazon Linux 인스턴스의 버전을 표시하려면 연결된 환경용 AWS Cloud9 IDE에서 또는 명령 또는 PuTTY와 같은 SSH 유틸리티에서 다음 ssh 명령을 실행합니다.

```
cat /etc/system-release
```

Amazon Linux 인스턴스에 설치된 패키지 목록을 표시하려면 다음 명령 중 하나 이상을 실행합니다.

설치된 모든 패키지를 목록 하나로 표시하려면 다음 명령을 실행합니다.

```
sudo yum list installed
```

패키지 이름에 지정된 텍스트가 포함되어 있는 설치된 패키지 목록을 표시하려면 다음 명령을 실행합니다.

```
sudo yum list installed | grep YOUR_SEARCH_TERM
```

앞의 명령에서 `YOUR_SEARCH_TERM`을 패키지 이름의 일부분으로 바꿉니다. 예를 들어, 이름에 `sql`이 포함되어 있는 설치된 모든 패키지 목록을 표시하려면 다음 명령을 실행합니다.

```
sudo yum list installed | grep sql
```

설치된 모든 패키지의 목록을 한 번에 한 페이지씩 표시하려면 다음 명령을 실행합니다.

```
sudo yum list installed | less
```

표시된 페이지를 스크롤하려면 다음 명령을 실행합니다.

- 줄 하나 아래로 이동하려면 **j**를 누릅니다.
- 줄 하나 위로 이동하려면 **k**를 누릅니다.

- 페이지 하나 아래로 이동하려면 **Ctrl-F**를 누릅니다.
- 페이지 하나 위로 이동하려면 **Ctrl-B**를 누릅니다.
- 종료하려면 **q**를 누릅니다.

> **Note**
>
> Amazon Linux 2를 사용하면 Extras Library를 사용하여 인스턴스에 애플리케이션 및 소프트웨어 업데이트를 설치할 수 있습니다. 이러한 소프트웨어 업데이트를 주제라고 합니다. 자세한 내용은 Amazon EC2 사용 설명서의 엑스트라 라이브러리 (Amazon Linux 2) 를 참조하십시오.

추가 옵션을 보려면 man yum 명령을 실행합니다. 다음 리소스도 참조하세요.

- Amazon Linux 2023: AMI 페이지입니다.
- Amazon Linux: Amazon Linux AMI 2018.03 패키지.

Ubuntu 서버

Ubuntu Server 인스턴스의 버전을 표시하려면 AWS Cloud9 IDE에서 연결된 환경에 대해 다음 명령을 실행하거나 ssh 명령 또는 PuTTY 등과 같은 SSH 유틸리티에서 다음 명령을 실행합니다.

```
lsb_release -a
```

버전이 설명 필드 옆에 표시됩니다.

Ubuntu Server에 설치된 패키지 목록을 표시하려면 다음 명령 중 하나 이상을 실행합니다.

설치된 모든 패키지를 목록 하나로 표시하려면 다음 명령을 실행합니다.

```
sudo apt list --installed
```

패키지 이름에 지정된 텍스트가 포함되어 있는 설치된 패키지 목록을 표시하려면 다음 명령을 실행합니다.

```
sudo apt list --installed | grep YOUR_SEARCH_TERM
```

앞의 명령에서 YOUR_SEARCH_TERM을 패키지 이름의 일부분으로 바꿉니다. 예를 들어, 이름에 sql이 포함되어 있는 설치된 모든 패키지 목록을 표시하려면 다음 명령을 실행합니다.

```
sudo apt list --installed grep sql
```

설치된 모든 패키지의 목록을 한 번에 한 페이지씩 표시하려면 다음 명령을 실행합니다.

```
sudo apt list --installed | less
```

표시된 페이지를 스크롤하려면 다음 명령을 실행합니다.

- 줄 하나 아래로 이동하려면 **j**를 누릅니다.
- 줄 하나 위로 이동하려면 **k**를 누릅니다.
- 페이지 하나 아래로 이동하려면 **Ctrl-F**를 누릅니다.
- 페이지 하나 위로 이동하려면 **Ctrl-B**를 누릅니다.
- 종료하려면 **q**를 누릅니다.

추가 옵션을 보려면 man apt 명령을 실행합니다. Ubuntu 웹 사이트에서 Ubuntu Packages Search를 참조하십시오.

AWS Cloud9에 서비스 연결 역할 사용

AWS Cloud9은 AWS Identity and Access Management(IAM) 서비스 연결 역할을 사용합니다. 서비스 연결 역할은 AWS Cloud9에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 AWS Cloud9에서 사전 정의하며 서비스에서 다른 AWS 서비스를 자동으로 호출하기 위해 필요한 모든 권한을 포함합니다.

서비스 연결 역할을 사용하면 AWS Cloud9 설정이 쉬워집니다. 필요한 권한을 추가할 필요가 없기 때문입니다. AWS Cloud9에서 서비스 연결 역할 권한을 정의하므로, AWS Cloud9에서만 해당 역할을 맡을 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며, 이 권한 정책은 다른 IAM 엔터티에 연결할 수 없습니다.

먼저 역할의 관련 리소스를 삭제해야만 역할을 삭제할 수 있습니다. 이렇게 하면 리소스에 대한 액세스 권한을 부주의로 삭제할 수 없기 때문에 AWS Cloud9 리소스가 보호됩니다.

서비스 연결 역할을 지원하는 기타 서비스에 대한 자세한 내용은 IAM으로 작업하는 AWS 서비스를 참조하고 서비스 연결 역할(Service-Linked Role) 열에 예(Yes)가 있는 서비스를 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예(Yes) 링크를 선택합니다.

- AWS Cloud9에 대한 서비스 연결 역할 권한

- AWS Cloud9에 대한 서비스 연결 역할 생성
- AWS Cloud9에 대한 서비스 연결 역할 편집
- AWS Cloud9에 대한 서비스 연결 역할 삭제
- AWS Cloud9 서비스 연결 역할에 대해 지원되는 리전

AWS Cloud9에 대한 서비스 연결 역할 권한

AWS Cloud9은 AWSServiceRoleForAWSCloud9이라는 서비스 연결 역할을 사용합니다. . 이 서비스 연결 역할은 `cloud9.amazonaws.com` 서비스에 해당 역할을 맡깁니다.

이 서비스 연결 역할의 권한 정책은 AWSCloud9ServiceRolePolicy이며 AWS Cloud9이 지정된 리소스에 대한 정책에 나열된 작업을 수행하도록 허용합니다.

> **Important**
>
> License Manager를 사용 중이고 `unable to access your environment` 오류가 발생하면 이전 서비스 연결 역할을 License Manager를 지원하는 버전으로 바꿔야 합니다. 이전 역할을 삭제하여 간단히 바꿀 수 있습니다. 그러면 업데이트된 역할이 자동으로 생성됩니다.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Action": [
    "ec2:RunInstances",
    "ec2:CreateSecurityGroup",
    "ec2:DescribeVpcs",
    "ec2:DescribeSubnets",
    "ec2:DescribeSecurityGroups",
    "ec2:DescribeInstances",
    "ec2:DescribeInstanceStatus",
    "cloudformation:CreateStack",
    "cloudformation:DescribeStacks",
    "cloudformation:DescribeStackEvents",
    "cloudformation:DescribeStackResources"
   ],
   "Resource": "*"
  },
```

```
  {
   "Effect": "Allow",
   "Action": [
    "ec2:TerminateInstances",
    "ec2:DeleteSecurityGroup",
    "ec2:AuthorizeSecurityGroupIngress"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "cloudformation:DeleteStack"
   ],
   "Resource": "arn:aws:cloudformation:*:*:stack/aws-cloud9-*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:CreateTags"
   ],
   "Resource": [
    "arn:aws:ec2:*:*:instance/*",
    "arn:aws:ec2:*:*:security-group/*"
   ],
   "Condition": {
    "StringLike": {
     "aws:RequestTag/Name": "aws-cloud9-*"
    }
   }
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:StartInstances",
    "ec2:StopInstances"
   ],
   "Resource": "*",
   "Condition": {
    "StringLike": {
     "ec2:ResourceTag/aws:cloudformation:stack-name": "aws-cloud9-*"
    }
   }
  },
```

```
  {
   "Effect": "Allow",
   "Action": [
    "ec2:StartInstances",
    "ec2:StopInstances"
   ],
   "Resource": [
    "arn:aws:license-manager:*:*:license-configuration:*"
   ]
  },
  {
   "Effect": "Allow",
   "Action": [
    "iam:ListInstanceProfiles",
    "iam:GetInstanceProfile"
   ],
   "Resource": [
    "arn:aws:iam::*:instance-profile/cloud9/*"
   ]
  },
  {
   "Effect": "Allow",
   "Action": [
    "iam:PassRole"
   ],
   "Resource": [
    "arn:aws:iam::*:role/service-role/AWSCloud9SSMAccessRole"
   ],
   "Condition": {
    "StringLike": {
     "iam:PassedToService": "ec2.amazonaws.com"
    }
   }
  }
 ]
}
```

IAM 엔터티(사용자, 그룹, 역할 등)를 대신해 AWS Cloud9이 서비스 연결 역할을 생성하도록 권한을 구성해야 합니다.

AWS Cloud9이 AWSServiceRoleForAWSCloud9 서비스 연결 역할을 생성하도록 허용하려면 AWS Cloud9이 대신하여 서비스 연결 역할을 생성해야 하는 IAM 엔터티의 권한 정책에 다음 문을 추가합니다.

```
{
  "Effect": "Allow",
  "Action": [
    "iam:CreateServiceLinkedRole"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "iam:AWSServiceName": "cloud9.amazonaws.com"
    }
  }
}
```

또는 IAM 엔터티에 `AWSCloud9User` 또는 `AWSCloud9Administrator` AWS 관리형 정책을 추가할 수도 있습니다.

IAM 엔터티가 AWSServiceRoleForAWSCloud9 서비스 연결 역할을 삭제하도록 허용하려면 서비스 연결 역할을 삭제해야 하는 IAM 엔터티의 권한 정책에 다음 문을 추가합니다.

```
{
  "Effect": "Allow",
  "Action": [
    "iam:DeleteServiceLinkedRole",
    "iam:GetServiceLinkedRoleDeletionStatus"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "iam:AWSServiceName": "cloud9.amazonaws.com"
    }
  }
}
```

AWS Cloud9에 대한 서비스 연결 역할 생성

서비스 연결 역할은 생성할 필요가 없습니다. AWS Cloud9 개발 환경을 생성할 때 AWS Cloud9이 서비스 연결 역할을 자동으로 생성합니다.

AWS Cloud9에 대한 서비스 연결 역할 편집

AWS Cloud9의 AWSServiceRoleForAWSCloud9 서비스 연결 역할은 편집할 수 없습니다. 예를 들어 서비스 연결 역할을 생성한 후에는 다양한 개체가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

AWS Cloud9에 대한 서비스 연결 역할 삭제

서비스 연결 역할이 필요한 기능 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제할 것을 권합니다. 이렇게 하면 적극적으로 모니터링하거나 유지 관리하지 않는 미사용 개체가 없게 됩니다.

IAM에서 서비스 연결 역할 삭제

IAM을 사용하여 서비스 연결 역할을 삭제하기 전에 역할에서 사용되는 AWS Cloud9 리소스를 제거해야 합니다. AWS Cloud9 리소스를 제거하려면 환경 삭제를 참조하세요.

IAM 콘솔을 사용하여 AWSServiceRoleForAWSCloud9 서비스 연결 역할을 삭제할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 삭제 섹션을 참조하세요.

AWS Cloud9 서비스 연결 역할이 지원되는 리전

AWS Cloud9에서는 서비스를 사용할 수 있는 모든 리전에서 서비스 연결 역할 사용을 지원합니다. 자세한 내용은 AWS Cloud9의 Amazon Web Services 일반 참조 섹션을 참조하세요.

AWS CloudTrail을 사용하여 AWS Cloud9 API 호출 로깅

AWS Cloud9는 AWS Cloud9에서 사용자, 역할, 또는 AWS 서비스가 수행한 작업에 대한 레코드를 제공하는 서비스인 CloudTrail과 통합됩니다. CloudTrail은 AWS Cloud9에 대한 모든 API 호출을 이벤트로 캡처합니다. 캡처되는 호출에는 AWS Cloud9 콘솔의 호출과 AWS Cloud9 API에 대한 코드의 호출이 포함됩니다. 추적을 생성하면 AWS Cloud9에 대한 이벤트를 포함한 CloudTrail 이벤트를 Amazon Simple Storage Service(Amazon S3) 버킷에 지속적으로 전송할 수 있습니다. 추적을 구성하지 않은 경우에도 CloudTrail 콘솔의 Event history(이벤트 기록)에서 최신 이벤트를 볼 수 있습니다. CloudTrail에서 수집한 정보를 사용하여 AWS Cloud9에 수행된 요청, 요청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

CloudTrail에 대한 자세한 내용은 AWS CloudTrail 사용 설명서를 참조하세요.

CloudTrail의 AWS Cloud9 정보

CloudTrail은 계정 생성 시 AWS 계정에서 사용되도록 설정됩니다. AWS Cloud9에서 활동이 발생하면 해당 활동이 이벤트 기록의 다른 AWS 서비스 이벤트와 함께 CloudTrail 이벤트에 기록됩니다. AWS 계정에서 최신 이벤트를 확인, 검색 및 다운로드할 수 있습니다. 자세한 내용은 CloudTrail 이벤트 기록을 사용하여 이벤트 보기를 참조하세요.

AWS Cloud9에 대한 이벤트를 포함하여 AWS 계정에 이벤트를 지속적으로 기록하려면 추적을 생성합니다. CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 콘솔에서 추적을 생성하면 기본적으로 모든 AWS 리전에 추적이 적용됩니다. 추적은 AWS 파티션에 있는 모든 리전의 이벤트를 로깅하고 지정된 S3 버킷으로 로그 파일을 전송합니다. 또는 CloudTrail 로그에서 수집된 이벤트 데이터를 추가 분석 및 처리하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 내용은 다음 자료를 참조하세요.

- 추적 생성 개요
- CloudTrail 지원 서비스 및 통합
- CloudTrail에 대한 Amazon SNS 알림 구성
- 여러 리전에서 CloudTrail 로그 파일 받기 및 여러 계정에서 CloudTrail 로그 파일 받기

AWS Cloud9는 CloudTrail 로그 파일에 다음 작업을 이벤트로 로깅합니다.

- `CreateEnvironmentEC2`
- `CreateEnvironmentSSH`
- `CreateEnvironmentMembership`
- `DeleteEnvironment`
- `DeleteEnvironmentMembership`
- `DescribeEnvironmentMemberships`
- `DescribeEnvironments`
- `DescribeEnvironmentStatus`
- `ListEnvironments`
- `ListTagsForResource`
- `TagResource`
- `UntagResource`
- `UpdateEnvironment`

- `UpdateEnvironmentMembership`

> **Note**
>
> AWS Cloud9에 대한 일부 CloudTrail 이벤트는 퍼블릭 API 작업에 의해 발생하지 않습니다. 대신 사용자 인증 및 관리형 임시 자격 증명에 영향을 주는 내부 업데이트에 의해 다음 이벤트가 시작됩니다.
>
> - `DisableManagedCredentialsByCollaborator`
> - `EnvironmentTokenSuccessfullyCreated`
> - `ManagedCredentialsUpdatedOnEnvironment`

모든 이벤트 및 로그 항목에는 요청을 생성한 사용자에 대한 정보가 들어 있습니다. 자격 증명 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트로 했는지 아니면 AWS Identity and Access Management IAM 사용자 보안 인증 정보로 했는지 여부.
- 역할 또는 연합된 사용자에 대한 임시 보안 자격 증명을 사용하여 요청이 생성되었는지 여부.
- 다른 AWS 서비스에서 요청했는지 여부.

자세한 내용은 CloudTrail userIdentity 요소를 참조하세요.

AWS Cloud9 로그 파일 항목 이해

추적이란 지정한 Amazon S3 버킷에 이벤트를 로그 파일로 입력할 수 있게 하는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함될 수 있습니다. 이벤트는 모든 소스로부터의 단일 요청을 나타내며 요청 작업, 작업 날짜와 시간, 요청 파라미터에 관한 정보가 들어 있습니다. CloudTrail 로그 파일은 퍼블릭 API 호출의 주문 스택 트레이스가 아니므로 특정 순서로 표시되지 않습니다.

- CreateEnvironmentEC2
- CreateEnvironmentSSH
- CreateEnvironmentMembership
- DeleteEnvironment
- DeleteEnvironmentMembership

- DescribeEnvironmentMemberships
- DescribeEnvironments
- DescribeEnvironmentStatus
- ListEnvironments
- ListTagsForResource
- TagResource
- UntagResource
- UpdateEnvironment
- UpdateEnvironmentMembership

CreateEnvironmentEC2

다음은 `CreateEnvironmentEC2` 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.


```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "CreateEnvironmentEC2",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
```

```
      "requestParameters": {
        "instanceType": "t2.small",
        "subnetId": "subnet-1d4a9eEX",
        "description": "HIDDEN_DUE_TO_SECURITY_REASONS",
        "dryRun": true,
        "automaticStopTimeMinutes": 30,
        "name": "my-test-environment",
        "clientRequestToken": "cloud9-console-f8e37272-e541-435d-a567-5c684EXAMPLE"
      },
      "responseElements": null,
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```

CreateEnvironmentSSH

다음은 `CreateEnvironmentSSH` 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
```

```
      "eventName": "CreateEnvironmentSSH",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "host": "198.51.100.0",
        "port": 22,
        "name": "my-ssh-environment",
        "description": "HIDDEN_DUE_TO_SECURITY_REASONS",
        "clientRequestToken": "cloud9-console-b015a0e9-469e-43e3-be90-6f432EXAMPLE",
        "loginName": "ec2-user"
      },
      "responseElements": {
        "environmentId": "5c39cc4a85d74a8bbb6e23ed6EXAMPLE"
      },
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```

CreateEnvironmentMembership

다음은 `CreateEnvironmentMembership` 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
```

```
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "CreateEnvironmentMembership",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "environmentId": "2f5ff70a640f49398f67e3bdeEXAMPLE",
        "userArn": "arn:aws:iam::111122223333:user/MyUser",
        "permissions": "read-write"
      },
      "responseElements": {
        "membership": {
          "environmentId": "2f5ff70a640f49398f67e3bdeEXAMPLE",
          "permissions": "read-write",
          "userId": "AIDACKCEVSQ6C2EXAMPLE",
          "userArn": "arn:aws:iam::111122223333:user/MyUser"
        }
      },
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```

DeleteEnvironment

다음은 `DeleteEnvironment` 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
```

```
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "DeleteEnvironment",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "environmentId": "2f5ff70a640f49398f67e3bdeEXAMPLE"
      },
      "responseElements": null,
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```

DeleteEnvironmentMembership

다음은 `DeleteEnvironmentMembership` 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
```

```
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "DeleteEnvironmentMembership",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "environmentId": "2f5ff70a640f49398f67e3bdeEXAMPLE",
        "userArn": "arn:aws:iam::111122223333:user/MyUser",
      },
      "responseElements": null,
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```

DescribeEnvironmentMemberships

다음은 `DescribeEnvironmentMemberships` 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
```

```
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "DescribeEnvironmentMemberships",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "nextToken": "NEXT_TOKEN_EXAMPLE",
        "permissions": [ "owner" ],
        "maxResults": 15
      },
      "responseElements": null,
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "readOnly": true,
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```

DescribeEnvironments

다음은 `DescribeEnvironments` 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
   "Records": [
     {
       "eventVersion": "1.05",
       "userIdentity": {
         "type": "IAMUser",
         "principalId": "AIDACKCEVSQ6C2EXAMPLE",
         "arn": "arn:aws:iam::111122223333:user/MyUser",
```

```
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "DescribeEnvironments",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "environmentIds": [
          "2f5ff70a640f49398f67e3bdeb811ab2"
        ]
      },
      "responseElements": null,
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "readOnly": true,
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```

DescribeEnvironmentStatus

다음은 `DescribeEnvironmentStatus` 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:sts::123456789012:myuser_role",
```

```
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:sts::123456789012:myuser_role",
                "accountId": "123456789012",
                "userName": "barshane_role"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-03-12T15:10:54Z"
            }
        }
    },
    "eventTime": "2021-03-12T15:13:31Z",
    "eventSource": "cloud9.amazonaws.com",
    "eventName": "DescribeEnvironmentStatus",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "XX.XX.XXX.XX",
    "userAgent": "aws-internal/3 aws-sdk-java/1.11.951
 Linux/4.9.230-0.1.ac.223.84.332.metal1.x86_64 OpenJDK_64-Bit_Server_VM/25.282-b08
 java/1.8.0_282 vendor/Oracle_Corporation",
    "requestParameters": {
        "environmentId": "31ea8a12746a4221b7d8e07d9ef6ee21"
    },
    "responseElements": null,
    "requestID": "68b163fb-aa88-4f40-bafd-4a18bf24cbd5",
    "eventID": "c0fc52a9-7331-4ad0-a8ee-157995dfb5e6",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123456789012"
}
```


ListEnvironments

다음은 `ListEnvironments` 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.


```
{
```

```
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "ListEnvironments",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "nextToken": "NEXT_TOKEN_EXAMPLE",
        "maxResults": 15
      },
      "responseElements": null,
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "readOnly": true,
      "eventType": "AwsApiCall",
      "recipientAccountId": "123456789012"
    }
  ]
}
```

ListTagsForResource

다음은 `ListTagsForResource` 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
```

```
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:sts::123456789012:myuser_role",
        "accountId": "123456789012",
        "accessKeyId": "AIDACKCEVSQ6C2EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "123456789012:myuser_role",
                "accountId": "123456789012",
                "userName": "barshane_role"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-03-23T16:41:51Z"
            }
        }
    },
    "eventTime": "2021-03-23T16:42:58Z",
    "eventSource": "cloud9.amazonaws.com",
    "eventName": "ListTagsForResource",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "XX.XX.XXX.XX",
    "userAgent": "aws-internal/3 aws-sdk-java/1.11.976
 Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 OpenJDK_64-Bit_Server_VM/25.282-b08
 java/1.8.0_282 vendor/Oracle_Corporation cfg/retry-mode/legacy",
    "requestParameters": {
        "resourceARN": "arn:aws:cloud9:us-
east-1:123456789012:environment:3XXXXXXXXX6a4221b7d8e07d9ef6ee21"
    },
    "responseElements": {
        "tags": "HIDDEN_DUE_TO_SECURITY_REASONS"
    },
    "requestID": "5750a344-8462-4020-82f9-f1d500a75162",
    "eventID": "188d572d-9a14-4082-b98b-0389964c7c30",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123456789012"
```

```
}
```

TagResource

다음은 TagResource 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:sts:: 123456789012:myuser_role",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/myuser_role",
                "accountId": "123456789012",
                "userName": "MyUser"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-03-23T15:03:57Z"
            }
        }
    },
    "eventTime": "2021-03-23T15:08:16Z",
    "eventSource": "cloud9.amazonaws.com",
    "eventName": "TagResource",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "54.XXX.XXX.XXX",
    "userAgent": "aws-internal/3 aws-sdk-java/1.11.976
 Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 OpenJDK_64-Bit_Server_VM/25.282-b08
 java/1.8.0_282 vendor/Oracle_Corporation cfg/retry-mode/legacy",
    "requestParameters": {
        "resourceARN": "arn:aws:cloud9:us-
east-1:123456789012:environment:3XXXXXXXXX6a4221b7d8e07d9ef6ee21",
        "tags": "HIDDEN_DUE_TO_SECURITY_REASONS"
    },
    "responseElements": null,
```

```
    "requestID": "658e9d70-91c2-41b8-9a69-c6b4cc6a9456",
    "eventID": "022b2893-73d1-44cb-be6f-d3faa68e83b1",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123456789012"
}
```

UntagResource

다음은 UntagResource 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:sts::123456789012/MyUser",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::123456789012:MyUser",
                "accountId": "123456789012",
                "userName": "MyUser"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-03-23T15:58:36Z"
            }
        }
    },
    "eventTime": "2021-03-23T16:05:08Z",
    "eventSource": "cloud9.amazonaws.com",
    "eventName": "UntagResource",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "3.XX.XX.XXX",
```

```
    "userAgent": "aws-internal/3 aws-sdk-java/1.11.976
 Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 OpenJDK_64-Bit_Server_VM/25.282-b08
 java/1.8.0_282 vendor/Oracle_Corporation cfg/retry-mode/legacy",
    "requestParameters": {
        "resourceARN": "arn:aws:cloud9:us-
east-1:123456789012:environment:3XXXXXXXXX6a4221b7d8e07d9ef6ee21",
        "tagKeys": "HIDDEN_DUE_TO_SECURITY_REASONS"
    },
    "responseElements": null,
    "requestID": "0eadaef3-dc0a-4cd7-85f6-135b8529f75f",
    "eventID": "41f2f2e2-4b17-43d4-96fc-9857981ca1de",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123456789012"
}
```

UpdateEnvironment

다음은 UpdateEnvironment 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
```

```
      "eventName": "UpdateEnvironment",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "environmentId": "2f5ff70a640f49398f67e3bdeEXAMPLE",
        "description": "HIDDEN_DUE_TO_SECURITY_REASONS",
        "name": "my-test-environment-renamed"
      },
      "responseElements": null,
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```

UpdateEnvironmentMembership

다음은 `UpdateEnvironmentMembership` 작업을 보여주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
```

```
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "UpdateEnvironmentMembership",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "environmentId": "2f5ff70a640f49398f67e3bdeEXAMPLE",
        "userArn": "arn:aws:iam::111122223333:user/MyUser",
        "permissions": "read-only"
      },
      "responseElements": {
        "membership": {
          "environmentId": "2f5ff70a640f49398f67e3bdeEXAMPLE",
          "permissions": "read-only",
          "userId": "AIDACKCEVSQ6C2EXAMPLE",
          "userArn": "arn:aws:iam::111122223333:user/MyUser"
           }
      },
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]}
```

Tags

태그는 사용자 또는 AWS가 AWS 리소스에 연결하는 레이블 또는 속성입니다. 각 태그는 키와 값 페어로 구성됩니다. IAM 사용 설명서의 AWS 리소스 태그를 사용하여 액세스 제어에서 설명하는 바와 같이, 태그를 사용하여 AWS Cloud9 리소스에 대한 액세스를 제어할 수 있습니다. 사용자 정의 비용 할당 태그에 설명된 대로 태그를 사용하여 결제 정보도 관리할 수 있습니다.

AWS Cloud9 EC2 개발 환경을 생성하면 AWS Cloud9이 환경을 관리하는 데 필요한 특정 시스템 태그를 포함합니다. 시스템 태그는 "aws:“로 시작됩니다. 환경을 생성하는 동안 사용자 고유의 리소스 태그를 추가할 수도 있습니다.

환경을 생성한 후 환경에 연결된 태그를 보거나, 환경에 새 리소스 태그를 추가하거나, 이전에 추가한 태그를 수정 또는 제거할 수 있습니다. 최대 50개의 사용자 정의 태그를 AWS Cloud9 환경에 연결할 수 있습니다.

태그를 보거나 업데이트하려면 다음 방법 중 하나 이상을 사용합니다.

- AWS Cloud9 콘솔에서 해당 환경을 선택한 다음 View Details(세부 정보 보기)를 선택합니다.

- AWS Cloud9 CLI 명령 `list-tags-for-resource`, `tag-resource` 및 `untag-resource`를 사용합니다.
- AWS Cloud9 API 작업 ListTagsForResource, TagResource 및 UntagResource를 사용합니다.

> **Warning**
>
> 앞의 방법을 사용하여 AWS Cloud9에 대해 생성하거나 업데이트하는 태그는 기본 리소스에 자동으로 전파되지 않습니다. 이 작업을 수행하는 방법에 대한 자세한 내용은 다음 단원(기본 리소스에 태그 업데이트 전파)을 참조하십시오.

기본 리소스에 태그 업데이트 전파

AWS Cloud9 CLI 명령 또는 API 작업을 사용하여 AWS Cloud9 환경에 연결된 태그를 추가, 수정 또는 제거하는 경우 이러한 변경 사항이 AWS CloudFormation 스택, Amazon EC2 인스턴스 및 Amazon EC2 보안 그룹과 같은 기본 리소스에 자동으로 전파되지 않습니다. 이러한 변경 사항을 수동으로 전파해야 합니다.

해당 환경의 환경 ID를 확인해 두면 아래 절차를 더 쉽게 수행할 수 있습니다. 이렇게 하지 않으려면 다음 절차를 따르세요.

1. AWS Cloud9 콘솔에서 해당 환경을 선택한 다음 [세부 정보 보기(View Details)]를 선택합니다.
2. Environment ARN(환경 ARN) 속성을 찾아 환경 ID를 기록해 둡니다. 환경 ARN에서 "environment:" 뒤에 나오는 부분이 환경 ID입니다.

태그를 사용할 대상에 따라 다음 위치 중 하나 이상에 태그 업데이트를 전파해야 합니다.

AWS CloudFormation 스택에 태그 업데이트 전파

> **Note**
>
> AWS CloudFormation 스택에 대한 태그를 업데이트하면 해당 업데이트가 스택과 연결된 Amazon EC2 인스턴스 및 Amazon EC2 보안 그룹에 자동으로 전파됩니다.

1. AWS CloudFormation 콘솔로 이동합니다.
2. 해당 AWS Cloud9 환경의 스택을 찾아 선택합니다. 환경 ID를 기록해 둔 경우 이 ID를 사용하여 환경을 필터링할 수 있습니다.
3. Stack info(스택 정보) 탭의 Tags(태그) 섹션에서 태그 목록을 검토합니다.
4. 태그를 업데이트해야 하는 경우 페이지 상단에 있는 Update(업데이트)를 선택하고 지침을 따릅니다. 자세한 내용은 AWS CloudFormation 사용 설명서에서 직접 스택 업데이트를 참조하세요.

`describe-stacks` 및 `update-stack` CLI 명령을 사용하여 태그를 업데이트할 수도 있습니다.

Amazon EC2 인스턴스에 태그 업데이트 전파

1. Amazon EC2 인스턴스 콘솔로 이동합니다.
2. 해당 AWS Cloud9 환경의 Amazon EC2 인스턴스를 찾아 선택합니다. 앞에서 환경 ID를 기록해 둔 경우 이 ID를 사용하여 환경을 필터링할 수 있습니다.
3. 태그 탭에서 필요에 따라 태그를 보고 업데이트합니다.

describe-tags, create-tags 및 delete-tags CLI 명령을 사용하여 태그를 업데이트할 수도 있습니다.

Amazon EC2 보안 그룹에 태그 업데이트 전파

1. Amazon EC2 보안 그룹 콘솔로 이동합니다.
2. 해당 AWS Cloud9 환경의 보안 그룹을 찾아 선택합니다. 앞에서 환경 ID를 기록해 둔 경우 이 ID를 사용하여 환경을 필터링할 수 있습니다.
3. Tags(태그) 탭에서 필요에 따라 태그를 보고 업데이트합니다.

describe-tags, create-tags 및 delete-tags CLI 명령을 사용하여 태그를 업데이트할 수도 있습니다.

보안 대상 AWS Cloud9

Amazon Web Services(AWS)에서 가장 우선순위가 높은 것이 클라우드 보안입니다. AWS 고객은 가장 보안에 민감한 조직의 요구 사항을 충족하도록 구축된 데이터 센터 및 네트워크 아키텍처를 활용할 수 있습니다. 보안은 기업과 기업 간의 AWS 공동 책임입니다. 공동 책임 모델은 이 사항을 클라우드 내 보안 및 클라우드의 보안으로 설명합니다.

클라우드 보안 — AWS 클라우드에서 제공되는 모든 서비스를 실행하는 인프라를 보호하고 안전하게 사용할 수 있는 서비스를 제공하는 역할을 합니다. AWS 당사의 보안 책임은 최우선 과제이며 AWS, AWS 규정 준수 프로그램의 일환으로 타사 감사자가 보안 효과를 정기적으로 테스트하고 검증합니다.

클라우드에서의 보안 — 사용자의 책임은 사용 중인 AWS 서비스와 데이터의 민감도, 조직의 요구 사항, 관련 법률 및 규정을 비롯한 기타 요인에 따라 결정됩니다.

AWS Cloud9 지원하는 특정 AWS 서비스를 통해 공동 책임 모델을 따릅니다. AWS 서비스 보안 정보는 AWS 서비스 보안 설명서 페이지 및 AWS 규정 준수 프로그램의 규정 준수 노력 범위에 속하는AWS 서비스를 참조하십시오.

다음 항목에서는 보안 및 규정 준수 목표를 AWS Cloud9 충족하도록 구성하는 방법을 보여줍니다.

주제

- 데이터 보호: AWS Cloud9
- Identity 및 Access Management에 대한 AWS Cloud9
- 로그인 및 모니터링 AWS Cloud9
- 규정 준수 검증: AWS Cloud9
- 의 레질리언스 AWS Cloud9
- 의 인프라 보안 AWS Cloud9
- 소프트웨어 업데이트 및 패치 적용
- 에 대한 보안 모범 사례 AWS Cloud9

데이터 보호: AWS Cloud9

AWS 공동 책임 모델 의 데이터 보호에 적용됩니다 AWS Cloud9. 이 모델에 설명된 대로 AWS 는 모든 데이터를 실행하는 글로벌 인프라를 보호하는 역할을 AWS 클라우드합니다. 사용자는 인프라에서 호스팅되는 콘텐츠를 관리해야 합니다. 사용하는 AWS 서비스 의 보안 구성과 관리 작업에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 대한 자세한 내용은 데이터 프라이버시 FAQ를 참조하세

요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS 보안 블로그의 AWS 공동 책임 모델 및 GDPR 블로그 게시물을 참조하세요.

데이터 보호를 위해 AWS 계정 자격 증명을 보호하고 AWS IAM Identity Center OR AWS Identity and Access Management (IAM) 을 사용하여 개별 사용자를 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 멀티 팩터 인증 설정(MFA)을 사용하세요.
- SSL/TLS를 사용하여 리소스와 통신하세요. AWS TLS 1.2는 필수이며 TLS 1.3를 권장합니다.
- 를 사용하여 API 및 사용자 활동 로깅을 설정합니다. AWS CloudTrail
- 포함된 모든 기본 보안 제어와 함께 AWS 암호화 솔루션을 사용하십시오 AWS 서비스.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용하세요.
- 명령줄 인터페이스 또는 API를 AWS 통해 액세스할 때 FIPS 140-2로 검증된 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용하십시오. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 FIPS(Federal Information Processing Standard) 140-2를 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 양식 필드에 입력하지 않는 것이 좋습니다. 여기에는 콘솔, API AWS Cloud9 또는 AWS 서비스 SDK를 사용하거나 다른 방법으로 작업하는 경우가 포함됩니다. AWS CLI AWS 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 보안 인증 정보를 URL에 포함해서는 안 됩니다.

데이터 암호화

데이터 암호화란 전송 중 (AWS Cloud9 계정과 사용자 AWS 계정 간에 이동하는 데이터) 및 저장 중 (AWS Cloud9 구성 저장소 및 AWS 클라우드 컴퓨팅 인스턴스에 저장되는 동안) 을 보호하는 것을 말합니다.

와 관련하여 다음과 같은 유형의 데이터에는 암호화를 통한 보호가 필요할 수 있습니다. AWS Cloud9

사용자의 콘텐츠 및 데이터

사용자가 조작, 수집 및 저장하는 정보. 다음은 이러한 유형의 데이터의 예입니다.

- 사용자의 코드 파일
- 연결된 EC2 환경 또는 SSH 환경에 대한 구성, 애플리케이션 및 데이터

AWS Cloud9 메타데이터

AWS Cloud9 조작, 수집 및 저장하는 데이터. 다음은 이러한 유형의 데이터의 예입니다.

- 탭 상태, 열린 파일, IDE 기본 설정과 같은 IDE 설정
- AWS Cloud9 개발 환경 메타데이터 (예: 환경 이름 및 설명)
- AWS Cloud9 서비스 API 및 콘솔 로그
- HTTP 요청과 같은 서비스 로그

AWS Cloud9 또한 데이터 플레인 서비스를 통해 일부 콘텐츠와 데이터를 전송합니다. 여기에는 파일, 터미널 입력, 출력 텍스트 및 일부 IDE 명령(예: 파일 저장)이 포함됩니다.

저장 중 암호화

유휴 데이터 암호화는 저장된 데이터를 암호화하여 무단 액세스로부터 데이터를 보호하는 것을 의미합니다. 코드 파일, 패키지 또는 종속성과 같은 AWS Cloud9 환경에 저장된 모든 고객 데이터는 항상 고객 리소스에 저장됩니다. 고객이 Amazon EC2 환경을 사용하는 경우 데이터는 계정에 있는 관련 Amazon EBS (엘라스틱 블록 스토어) 볼륨에 저장됩니다. AWS 고객이 SSH 환경을 사용하는 경우 데이터는 Linux 서버의 로컬 스토리지에 저장됩니다.

Amazon EC2 인스턴스가 AWS Cloud9 개발 환경용으로 생성되면 암호화되지 않은 Amazon EBS 볼륨이 생성되어 해당 인스턴스에 연결됩니다. 데이터를 암호화하려는 고객은 암호화된 EBS 볼륨을 생성하여 EC2 인스턴스에 연결해야 합니다. AWS Cloud9 및 연결된 Amazon EBS 볼륨은 기본적으로 지역별 설정인 Amazon EBS 기본 암호화를 지원합니다. 자세한 내용을 알아보려면 AWS Elastic Compute Cloud 사용 설명서의 암호화 기본 제공을 참조하세요.

환경 이름, 환경 구성원, IDE 설정과 같은 AWS Cloud9 개발 환경에 대한 메타데이터는 고객 리소스가 아닌 사용자가 AWS저장합니다. 환경 설명 및 IDE 설정과 같은 고객별 정보는 암호화됩니다.

전송 중 암호화

전송 중 데이터 암호화는 데이터가 통신 엔드포인트 간을 이동하는 동안 데이터를 가로채기에서 보호하는 것을 의미합니다. 고객의 클라이언트와 AWS Cloud9 서비스 간에 전송되는 모든 데이터는 HTTPS, WSS 및 암호화된 SSH를 통해 암호화됩니다.

- HTTPS — 고객의 웹 브라우저와 서비스 간의 보안 요청을 보장합니다. AWS Cloud9 AWS Cloud9 또한 고객 브라우저에서 HTTPS를 통해 CloudFront 전송된 Amazon의 자산을 로드합니다.

- WSS (WebSocket 보안) — 고객의 웹 브라우저와 서비스 WebSockets 간에 안전한 양방향 통신을 가능하게 합니다. AWS Cloud9
- 암호화된 SSH (Secure Shell): 클라이언트의 웹 브라우저와 서비스 간에 데이터를 안전하게 전송할 수 있습니다. AWS Cloud9

에서 지원하는 브라우저를 사용하느냐에 따라 HTTPS, WSS 및 SSH 프로토콜 사용이 달라집니다. AWS Cloud9AWS Cloud9에 지원되는 브라우저를 참조하세요.

> **Note**
>
> 암호화 프로토콜은 AWS Cloud9에서 기본적으로 구현됩니다. 고객은 설정을 변경할 수 없습니다. encryption-in-transit

키 관리

AWS Key Management Service (AWS KMS) 는 고객 데이터를 암호화하는 데 사용되는 암호화 키를 생성하고 제어하는 AWS KMS keys관리형 서비스입니다. AWS Cloud9 고객을 대신하여 데이터 암호화를 위한 암호화 키를 생성하고 관리합니다.

인터네트워크 트래픽 개인 정보

SSH 환경은 고객 소유의 온프레미스 컴퓨팅 및 스토리지에 연결합니다. 암호화된 SSH, HTTPS 및 WSS 연결은 서비스와 SSH 환경 간의 데이터 전송을 지원합니다.

특정 VPC 및 서브넷 내에서 시작되도록 AWS Cloud9 EC2 개발 환경 (Amazon EC2 인스턴스 기반) 을 구성할 수 있습니다. Amazon Virtual Private Cloud 설정에 대한 자세한 내용은 개발 환경을 위한 AWS Cloud9 VPC 설정 섹션을 참조하세요.

Identity 및 Access Management에 대한 AWS Cloud9

AWS Identity and Access Management (IAM) 은 관리자가 리소스에 대한 액세스를 안전하게 제어할 수 AWS 서비스 있도록 AWS 도와줍니다. IAM 관리자는 리소스를 사용할 수 있는 인증 (로그인) 및 권한 부여 (권한 보유) 를 받을 수 있는 사용자를 제어합니다. AWS Cloud9 IAM은 추가 AWS 서비스 비용 없이 사용할 수 있습니다.

주제

고객

사용하는 방식 AWS Identity and Access Management (IAM) 은 수행하는 작업에 따라 다릅니다. AWS Cloud9

서비스 사용자 - AWS Cloud9 서비스를 사용하여 작업을 수행하는 경우 관리자가 필요한 자격 증명과 권한을 제공합니다. 더 많은 AWS Cloud9 기능을 사용하여 작업을 수행함에 따라 추가 권한이 필요할 수 있습니다. 액세스 권한 관리 방식을 이해하면 적절한 권한을 관리자에게 요청할 수 있습니다. AWS Cloud9의 기능에 액세스할 수 없는 경우 AWS Cloud9 ID 및 액세스 문제 해결을 참조하세요.

서비스 관리자 — 회사에서 AWS Cloud9 리소스를 담당하는 경우 전체 액세스 권한이 있을 수 AWS Cloud9있습니다. 서비스 사용자가 액세스해야 하는 AWS Cloud9 기능과 리소스를 결정하는 것은 여러분의 몫입니다. 그런 다음, IAM 관리자에게 요청을 제출하여 서비스 사용자의 권한을 변경해야 합니다. 이 페이지의 정보를 검토하여 IAM의 기본 개념을 이해하십시오. 회사에서 IAM을 어떻게 사용할 수 있는지 자세히 AWS Cloud9알아보려면 을 참조하십시오IAM의 AWS Cloud9 작동 방식.

IAM 관리자 - IAM 관리자라면 AWS Cloud9에 대한 액세스 권한 관리 정책 작성 방법을 자세히 알고 싶을 것입니다. IAM에서 사용할 수 있는 AWS Cloud9 ID 기반 정책의 예를 보려면 을 참조하십시오. AWS Cloud9에 대한 자격 증명 기반 정책 예시

ID를 통한 인증

인증은 ID 자격 증명을 AWS 사용하여 로그인하는 방법입니다. IAM 사용자로 인증 (로그인 AWS) 하거나 IAM 역할을 맡아 인증 (로그인) 해야 합니다. AWS 계정 루트 사용자

ID 소스를 통해 제공된 자격 증명을 사용하여 페더레이션 ID로 로그인할 수 있습니다. AWS AWS IAM Identity Center (IAM ID 센터) 사용자, 회사의 싱글 사인온 인증, Google 또는 Facebook 자격 증명이 페더레이션 ID의 예입니다. 연동 자격 증명으로 로그인할 때 관리자가 이전에 IAM 역할을 사용하여 ID 페더레이션을 설정했습니다. 페더레이션을 사용하여 액세스하는 경우 AWS 간접적으로 역할을 맡게 됩니다.

사용자 유형에 따라 AWS Management Console 또는 AWS 액세스 포털에 로그인할 수 있습니다. 로그인에 대한 자세한 내용은 AWS 로그인 사용 설명서의 내 로그인 방법을 참조하십시오. AWS AWS 계정

AWS 프로그래밍 방식으로 액세스하는 경우 자격 증명을 사용하여 요청에 암호화 방식으로 서명할 수 있는 소프트웨어 개발 키트 (SDK) 와 명령줄 인터페이스 (CLI) 를 AWS 제공합니다. AWS 도구를 사용하지 않는 경우 요청에 직접 서명해야 합니다. 권장 방법을 사용하여 직접 요청에 서명하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 AWS API 요청 서명을 참조하십시오.

사용하는 인증 방법에 상관없이 추가 보안 정보를 제공해야 할 수도 있습니다. 예를 들어, AWS 계정의 보안을 강화하기 위해 다단계 인증 (MFA) 을 사용할 것을 권장합니다. 자세한 내용은 AWS IAM Identity Center 사용 설명서의 다중 인증 및 IAM 사용 설명서의 AWS에서 다중 인증(MFA) 사용을 참조하세요.

AWS 계정 루트 사용자

계정을 AWS 계정만들 때는 먼저 계정의 모든 AWS 서비스 리소스에 대한 완전한 액세스 권한을 가진 하나의 로그인 ID로 시작합니다. 이 ID를 AWS 계정 루트 사용자라고 하며, 계정을 만들 때 사용한 이메일 주소와 비밀번호로 로그인하여 액세스할 수 있습니다. 일상적인 태스크에 루트 사용자를 사용하지 않을 것을 강력히 권장합니다. 루트 사용자 보안 인증 정보를 보호하고 루트 사용자만 수행할 수 있는 태스크를 수행하는 데 사용하세요. 루트 사용자로 로그인해야 하는 태스크의 전체 목록은 IAM 사용자 안내서의 루트 사용자 보안 인증이 필요한 태스크를 참조하세요.

연동 자격 증명

가장 좋은 방법은 관리자 액세스가 필요한 사용자를 비롯한 수동 AWS 서비스 사용자가 ID 공급자와의 페더레이션을 사용하여 임시 자격 증명을 사용하여 액세스하도록 하는 것입니다.

페더레이션 ID는 기업 사용자 디렉토리, 웹 ID 공급자, Identity Center 디렉터리의 사용자 또는 ID 소스를 통해 제공된 자격 증명을 사용하여 액세스하는 AWS 서비스 모든 사용자를 말합니다. AWS Directory Service페더레이션 ID에 AWS 계정액세스하면 이들이 역할을 맡고 역할은 임시 자격 증명을 제공합니다.

중앙 집중식 액세스 관리를 위해 AWS IAM Identity Center(을)를 사용하는 것이 좋습니다. IAM Identity Center에서 사용자 및 그룹을 생성하거나 자체 ID 소스의 사용자 및 그룹 집합에 연결하고 동기화하여

모든 사용자 및 애플리케이션에서 사용할 수 있습니다. AWS 계정 IAM Identity Center에 대한 자세한 내용은AWS IAM Identity Center 사용 설명서에서 IAM Identity Center란 무엇입니까?를 참조하세요.

IAM 사용자 및 그룹

IAM 사용자는 단일 사용자 또는 애플리케이션에 대한 특정 권한을 AWS 계정 가진 사용자 내 자격 증명입니다. 가능하면 암호 및 액세스 키와 같은 장기 자격 증명이 있는 IAM 사용자를 생성하는 대신 임시 자격 증명을 사용하는 것이 좋습니다. 하지만 IAM 사용자의 장기 자격 증명이 필요한 특정 사용 사례가 있는 경우 액세스 키를 교체하는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 장기 보안 인증이 필요한 사용 사례의 경우 정기적으로 액세스 키 교체를 참조하세요.

IAM 그룹은 IAM 사용자 컬렉션을 지정하는 자격 증명입니다. 사용자는 그룹으로 로그인할 수 없습니다. 그룹을 사용하여 여러 사용자의 권한을 한 번에 지정할 수 있습니다. 그룹을 사용하면 대규모 사용자 집합의 권한을 더 쉽게 관리할 수 있습니다. 예를 들어, IAMAdmins라는 그룹이 있고 이 그룹에 IAM 리소스를 관리할 권한을 부여할 수 있습니다.

사용자는 역할과 다릅니다. 사용자는 한 사람 또는 애플리케이션과 고유하게 연결되지만, 역할은 해당 역할이 필요한 사람이라면 누구나 수임할 수 있습니다. 사용자는 영구적인 장기 보안 인증을 가지고 있지만, 역할은 임시 보안 인증만 제공합니다. 자세한 정보는 IAM 사용 설명서의 IAM 사용자를 만들어야 하는 경우(역할이 아님)를 참조하세요.

IAM 역할

IAM 역할은 특정 권한을 가진 사용자 AWS 계정 내의 자격 증명입니다. IAM 사용자와 유사하지만, 특정 개인과 연결되지 않습니다. 역할을 AWS Management Console 전환하여 에서 일시적으로 IAM 역할을 맡을 수 있습니다. AWS CLI 또는 AWS API 작업을 호출하거나 사용자 지정 URL을 사용하여 역할을 수임할 수 있습니다. 역할 사용 방법에 대한 자세한 정보는 IAM 사용 설명서의 IAM 역할 사용을 참조하세요.

임시 보안 인증이 있는 IAM 역할은 다음과 같은 상황에서 유용합니다.

- 페더레이션 사용자 액세스 - 연동 자격 증명에 권한을 부여하려면 역할을 생성하고 해당 역할의 권한을 정의합니다. 연동 자격 증명이 인증되면 역할이 연결되고 역할에 정의된 권한이 부여됩니다. 페더레이션 역할에 대한 자세한 내용은 IAM 사용 설명서의 타사 자격 증명 공급자의 역할 만들기를 참조하세요. IAM Identity Center를 사용하는 경우 권한 세트를 구성합니다. 인증 후 아이덴티티가 액세스할 수 있는 항목을 제어하기 위해 IAM Identity Center는 권한 세트를 IAM의 역할과 연관 짓습니다. 권한 세트에 대한 자세한 내용은 AWS IAM Identity Center 사용 설명서의 권한 세트를 참조하세요.
- 임시 IAM 사용자 권한 - IAM 사용자 또는 역할은 IAM 역할을 수임하여 특정 태스크에 대한 다양한 권한을 임시로 받을 수 있습니다.

- 크로스 계정 액세스 - IAM 역할을 사용하여 다른 계정의 사용자(신뢰할 수 있는 보안 주체)가 내 계정의 리소스에 액세스하도록 허용할 수 있습니다. 역할은 계정 간 액세스를 부여하는 기본적인 방법입니다. 그러나 일부 AWS 서비스경우에는 역할을 프록시로 사용하는 대신 정책을 리소스에 직접 연결할 수 있습니다. 크로스 계정 액세스를 위한 역할과 리소스 기반 정책의 차이점을 알아보려면 IAM 사용 설명서의 IAM 역할과 리소스 기반 정책의 차이를 참조하세요.
- 서비스 간 액세스 — 일부는 다른 AWS 서비스서비스의 기능을 AWS 서비스 사용합니다. 예컨대, 어떤 서비스에서 호출을 수행하면 일반적으로 해당 서비스는 Amazon EC2에서 애플리케이션을 실행하거나 Amazon S3에 객체를 저장합니다. 서비스는 호출하는 보안 주체의 권한을 사용하거나, 서비스 역할을 사용하거나, 또는 서비스 연결 역할을 사용하여 이 작업을 수행할 수 있습니다.
  - 순방향 액세스 세션 (FAS) — IAM 사용자 또는 역할을 사용하여 작업을 수행하는 경우 보안 AWS 주체로 간주됩니다. 일부 서비스를 사용하는 경우 다른 서비스에서 다른 작업을 시작하는 작업을 수행할 수 있습니다. FAS는 전화를 거는 주체의 권한을 다운스트림 AWS 서비스서비스에 AWS 서비스 요청하기 위한 요청과 결합하여 사용합니다. FAS 요청은 다른 서비스 AWS 서비스 또는 리소스와의 상호 작용이 필요한 요청을 서비스가 수신한 경우에만 이루어집니다. 이 경우 두 작업을 모두 수행할 수 있는 권한이 있어야 합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.
  - 서비스 역할 - 서비스 역할은 서비스가 사용자를 대신하여 태스크를 수행하기 위해 맡는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 정보는 IAM 사용 설명서의 AWS 서비스에 대한 권한을 위임할 역할 생성을 참조하세요.
  - 서비스 연결 역할 — 서비스 연결 역할은 에 연결된 서비스 역할의 한 유형입니다. AWS 서비스서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은 사용자에게 AWS 계정 표시되며 해당 서비스가 소유합니다. IAM 관리자는 서비스 링크 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.
- Amazon EC2에서 실행되는 애플리케이션 — IAM 역할을 사용하여 EC2 인스턴스에서 실행되고 API 요청을 AWS CLI 하는 애플리케이션의 임시 자격 증명을 관리할 수 있습니다. AWS 이는 EC2 인스턴스 내에 액세스 키를 저장할 때 권장되는 방법입니다. EC2 인스턴스에 AWS 역할을 할당하고 모든 애플리케이션에서 사용할 수 있게 하려면 인스턴스에 연결된 인스턴스 프로필을 생성합니다. 인스턴스 프로파일에는 역할이 포함되어 있으며 EC2 인스턴스에서 실행되는 프로그램이 임시 보안 인증을 얻을 수 있습니다. 자세한 정보는 IAM 사용 설명서의 IAM 역할을 사용하여 Amazon EC2 인스턴스에서 실행되는 애플리케이션에 권한 부여를 참조하세요.

IAM 역할을 사용할지 또는 IAM 사용자를 사용할지를 알아보려면 IAM 사용 설명서의 IAM 역할(사용자 대신)을 생성하는 경우를 참조하세요.

정책을 사용한 액세스 관리

정책을 생성하고 이를 AWS ID 또는 리소스에 AWS 연결하여 액세스를 제어할 수 있습니다. 정책은 ID 또는 리소스와 연결될 때 AWS 해당 권한을 정의하는 객체입니다. AWS 주도자 (사용자, 루트 사용자 또는 역할 세션) 가 요청할 때 이러한 정책을 평가합니다. 정책에서 권한은 요청이 허용되거나 거부되는 지를 결정합니다. 대부분의 정책은 JSON 문서로 AWS 저장됩니다. JSON 정책 문서의 구조와 콘텐츠에 대한 자세한 정보는 IAM 사용 설명서의 JSON 정책 개요를 참조하세요.

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

기본적으로, 사용자와 역할에는 어떠한 권한도 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다. 그런 다음 관리자가 IAM 정책을 역할에 추가하고, 사용자가 역할을 수임할 수 있습니다.

IAM 정책은 작업을 수행하기 위해 사용하는 방법과 상관없이 작업에 대한 권한을 정의합니다. 예를 들어, `iam:GetRole`태스크를 허용하는 정책이 있다고 가정합니다. 해당 정책을 사용하는 사용자는 AWS Management Console, AWS CLI, 또는 AWS API에서 역할 정보를 가져올 수 있습니다.

ID 기반 정책

ID 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 자격 증명에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자와 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성을 참조하세요.

자격 증명 기반 정책은 인라인 정책 또는 관리형 정책으로 한층 더 분류할 수 있습니다. 인라인 정책은 단일 사용자, 그룹 또는 역할에 직접 포함됩니다. 관리형 정책은 내 여러 사용자, 그룹 및 역할에 연결할 수 있는 독립형 정책입니다. AWS 계정관리형 정책에는 AWS 관리형 정책과 고객 관리형 정책이 포함됩니다. 관리형 정책 또는 인라인 정책을 선택하는 방법을 알아보려면 IAM 사용 설명서의 관리형 정책과 인라인 정책의 선택을 참조하세요.

리소스 기반 정책

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 리소스 기반 정책의 예는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 보안 주체에는 계정, 사용자, 역할, 연동 사용자 등이 포함될 수 있습니다. AWS 서비스

리소스 기반 정책은 해당 서비스에 있는 인라인 정책입니다. IAM의 AWS 관리형 정책은 리소스 기반 정책에 사용할 수 없습니다.

액세스 제어 목록(ACLs)

액세스 제어 목록(ACL)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는지를 제어합니다. ACLs는 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

ACL을 지원하는 서비스의 예로는 아마존 S3와 아마존 VPC가 있습니다. AWS WAF ACL에 대해 자세히 알아보려면 Amazon Simple Storage Service 개발자 안내서의 액세스 제어 목록(ACL) 개요를 참조하세요.

기타 정책 타입

AWS 일반적이지 않은 추가 정책 유형을 지원합니다. 이러한 정책 타입은 더 일반적인 정책 타입에 따라 사용자에게 부여되는 최대 권한을 설정할 수 있습니다.

- 권한 경계 – 권한 경계는 보안 인증 기반 정책에 따라 IAM 엔터티(IAM 사용자 또는 역할)에 부여할 수 있는 최대 권한을 설정하는 고급 기능입니다. 개체에 대한 권한 경계를 설정할 수 있습니다. 그 결과로 얻는 권한은 엔터티의 자격 증명 기반 정책과 그 권한 경계의 교집합입니다. `Principal` 필드에서 사용자나 역할을 보안 주체로 지정하는 리소스 기반 정책은 권한 경계를 통해 제한되지 않습니다. 이러한 정책 중 하나에 포함된 명시적 거부는 허용을 재정의합니다. 권한 경계에 대한 자세한 정보는 IAM 사용 설명서의 IAM 엔터티에 대한 권한 경계를 참조하세요.
- 서비스 제어 정책 (SCP) - SCP는 조직 또는 조직 단위 (OU) 에 대한 최대 권한을 지정하는 JSON 정책입니다. AWS Organizations AWS Organizations 사업체가 소유한 여러 AWS 계정 개를 그룹화하고 중앙에서 관리하는 서비스입니다. 조직에서 모든 기능을 활성화할 경우 서비스 제어 정책(SCP)을 임의의 또는 모든 계정에 적용할 수 있습니다. SCP는 구성원 계정의 엔티티 (각 엔티티 포함) 에 대한 권한을 제한합니다. AWS 계정 루트 사용자조직 및 SCP에 대한 자세한 정보는 AWS Organizations 사용 설명서의 SCP 작동 방식을 참조하세요.
- 세션 정책 – 세션 정책은 역할 또는 페더레이션 사용자에 대해 임시 세션을 프로그래밍 방식으로 생성할 때 파라미터로 전달하는 고급 정책입니다. 결과적으로 얻는 세션의 권한은 사용자 또는 역할 자격 증명 기반 정책의 교차 및 세션 정책입니다. 또한 권한을 리소스 기반 정책에서 가져올 수도 있습니다. 이러한 정책 중 하나에 포함된 명시적 거부는 허용을 재정의합니다. 자세한 정보는 IAM 사용 설명서의 세션 정책을 참조하세요.

여러 정책 타입

여러 정책 타입이 요청에 적용되는 경우 결과 권한은 이해하기가 더 복잡합니다. 여러 정책 유형이 관련된 경우 요청을 허용할지 여부를 AWS 결정하는 방법을 알아보려면 IAM 사용 설명서의 정책 평가 로직을 참조하십시오.

IAM의 AWS Cloud9 작동 방식

IAM을 사용하여 액세스를 AWS Cloud9관리하기 전에 어떤 IAM 기능과 함께 사용할 수 있는지 알아보세요. AWS Cloud9

함께 사용할 수 있는 IAM 기능 AWS Cloud9

IAM 특성	AWS Cloud9 지원
ID 기반 정책	예
리소스 기반 정책	아니요
정책 작업	예
정책 리소스	예
정책 조건 키(서비스별)	예
ACLs	아니요
ABAC(정책의 태그)	예
임시 보안 인증	예
전달 액세스 세션(FAS)	예
서비스 역할	예
서비스 연결 역할	예

AWS Cloud9 및 기타 AWS 서비스가 대부분의 IAM 기능과 어떻게 작동하는지 자세히 알아보려면 IAM 사용 설명서의 IAM과 함께 작동하는AWS 서비스를 참조하십시오.

ID 기반 정책은 다음과 같습니다. AWS Cloud9

ID 기반 정책 지원	예

자격 증명 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 자격 증명에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자와 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성을 참조하세요.

IAM 자격 증명 기반 정책을 사용하면 허용되거나 거부되는 작업와 리소스뿐 아니라 작업이 허용되거나 거부되는 조건을 지정할 수 있습니다. 자격 증명 기반 정책에서는 보안 주체가 연결된 사용자 또는 역할에 적용되므로 보안 주체를 지정할 수 없습니다. JSON 정책에서 사용하는 모든 요소에 대해 알아보려면 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하세요.

다음에 대한 ID 기반 정책 예제 AWS Cloud9

AWS Cloud9 ID 기반 정책의 예를 보려면 을 참조하십시오. AWS Cloud9에 대한 자격 증명 기반 정책 예시

내 리소스 기반 정책 AWS Cloud9

리소스 기반 정책 지원	아니요

리소스 기반 정책은 리소스에 연결하는 JSON 정책 문서입니다. 리소스 기반 정책의 예는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 보안 주체에는 계정, 사용자, 역할, 연동 사용자 등이 포함될 수 있습니다. AWS 서비스

계정 간 액세스를 활성화하려는 경우 전체 계정이나 다른 계정의 IAM 엔터티를 리소스 기반 정책의 보안 주체로 지정할 수 있습니다. 리소스 기반 정책에 크로스 계정 보안 주체를 추가하는 것은 트러스트 관계 설정의 절반밖에 되지 않는다는 것을 유념하세요. 보안 주체와 리소스가 다른 AWS 계정경우 신뢰할 수 있는 계정의 IAM 관리자는 보안 주체 개체 (사용자 또는 역할) 에게 리소스에 액세스할 수 있는 권한도 부여해야 합니다. 개체에 자격 증명 기반 정책을 연결하여 권한을 부여합니다. 하지만 리소

스 기반 정책이 동일 계정의 보안 주체에 액세스를 부여하는 경우 추가 자격 증명 기반 정책이 필요하지 않습니다. 자세한 정보는 IAM 사용 설명서의 IAM 역할과 리소스 기반 정책의 차이를 참조하세요.

AWS Cloud9 리소스 기반 정책을 지원하지 않지만 AWS Cloud9 API 및 IDE를 통해 AWS Cloud9 환경 구성원의 AWS Cloud9 환경 리소스 권한을 제어할 수는 있습니다. AWS Cloud9

에 대한 정책 조치 AWS Cloud9

정책 작업 지원	예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 `Action`요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 태스크를 설명합니다. 정책 작업은 일반적으로 관련 AWS API 작업과 이름이 같습니다. 일치하는 API 작업이 없는 권한 전용 작업 같은 몇 가지 예외도 있습니다. 정책에서 여러 작업이 필요한 몇 가지 작업도 있습니다. 이러한 추가 작업을 일컬어 종속 작업이라고 합니다.

연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하십시오.

AWS Cloud9 작업 목록을 보려면 서비스 권한 부여 AWS Cloud9참조에 정의된 작업을 참조하십시오.

정책 조치는 조치 앞에 다음 접두사를 AWS Cloud9 사용합니다.

```
account
```

단일 문에서 여러 작업을 지정하려면 다음과 같이 쉼표로 구분합니다.

```
"Action": [
      "account:action1",
      "account:action2"
         ]
```

AWS Cloud9 ID 기반 정책의 예를 보려면 을 참조하십시오. AWS Cloud9에 대한 자격 증명 기반 정책 예시

에 대한 정책 리소스 AWS Cloud9

정책 리소스 지원	예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지 지정할 수 있습니다.

`Resource` JSON 정책 요소는 작업이 적용되는 하나 이상의 개체를 지정합니다. 문장에는 `Resource`또는 `NotResource`요소가 반드시 추가되어야 합니다. 모범 사례에 따라 Amazon 리소스 이름(ARN)을 사용하여 리소스를 지정합니다. 리소스 수준 권한이라고 하는 특정 리소스 타입을 지원하는 작업에 대해 이 작업을 수행할 수 있습니다.

작업 나열과 같이 리소스 수준 권한을 지원하지 않는 작업의 경우, 와일드카드(*)를 사용하여 해당 문이 모든 리소스에 적용됨을 나타냅니다.

```
"Resource": "*"
```

AWS Cloud9 리소스 유형 및 해당 ARN 목록을 보려면 서비스 권한 부여 AWS Cloud9 참조에 정의된 리소스를 참조하십시오. 각 리소스의 ARN을 지정할 수 있는 작업을 알아보려면 AWS Cloud9가 정의한 작업을 참조하십시오.

AWS Cloud9 ID 기반 정책의 예를 보려면 을 참조하십시오. AWS Cloud9에 대한 자격 증명 기반 정책 예시

에 대한 정책 조건 키 AWS Cloud9

서비스별 정책 조건 키 지원	예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지 지정할 수 있습니다.

`Condition` 요소(또는 `Condition` 블록)를 사용하면 정책이 발효되는 조건을 지정할 수 있습니다. `Condition` 요소는 옵션입니다. 같거나 작음과 같은 조건 연산자를 사용하여 정책의 조건을 요청의 값과 일치시키는 조건식을 생성할 수 있습니다.

한 문에서 여러 Condition요소를 지정하거나 단일 Condition요소에서 여러 키를 지정하는 경우 AWS 는 논리적 AND태스크를 사용하여 평가합니다. 단일 조건 키에 여러 값을 지정하는 경우 는 논리적 OR 연산을 사용하여 조건을 AWS 평가합니다. 명문의 권한을 부여하기 전에 모든 조건을 충족해야 합니다.

조건을 지정할 때 자리 표시자 변수를 사용할 수도 있습니다. 예를 들어, IAM 사용자에게 IAM 사용자 이름으로 태그가 지정된 경우에만 리소스에 액세스할 수 있는 권한을 부여할 수 있습니다. 자세한 정보는 IAM 사용 설명서의 IAM 정책 요소: 변수 및 태그를 참조하세요.

AWS 글로벌 조건 키 및 서비스별 조건 키를 지원합니다. 모든 AWS 글로벌 조건 키를 보려면 IAM 사용 AWS 설명서의 글로벌 조건 컨텍스트 키를 참조하십시오.

AWS Cloud9 조건 키 목록을 보려면 서비스 권한 부여 참조의 조건 키를 참조하십시오 AWS Cloud9. 조건 키를 사용할 수 있는 작업 및 리소스에 대해 알아보려면 작업 정의 기준을 참조하십시오 AWS Cloud9.

AWS Cloud9 ID 기반 정책의 예를 보려면 을 참조하십시오. AWS Cloud9에 대한 자격 증명 기반 정책 예시

내 ACL AWS Cloud9

ACL 지원	아니요

액세스 제어 목록(ACLs)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는 지를 제어합니다. ACLs는 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

ABAC 포함 AWS Cloud9

ABAC 지원(정책의 태그)	예

ABAC(속성 기반 액세스 제어)는 속성을 기반으로 권한을 정의하는 권한 부여 전략입니다. AWS에서는 이러한 속성을 태그라고 합니다. IAM 개체 (사용자 또는 역할) 및 여러 AWS 리소스에 태그를 첨부할 수 있습니다. ABAC의 첫 번째 단계로 개체 및 리소스에 태그를 지정합니다. 그런 다음 보안 주체의 태그가 액세스하려는 리소스의 태그와 일치할 때 작업을 허용하도록 ABAC 정책을 설계합니다.

ABAC는 빠르게 성장하는 환경에서 유용하며 정책 관리가 번거로운 상황에 도움이 됩니다.

태그를 기반으로 액세스를 제어하려면 `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` 또는 `aws:TagKeys` 조건 키를 사용하여 정책의 조건 요소에 태그 정보를 제공합니다.

서비스가 모든 리소스 타입에 대해 세 가지 조건 키를 모두 지원하는 경우, 값은 서비스에 대해 예입니다. 서비스가 일부 리소스 타입에 대해서만 세 가지 조건 키를 모두 지원하는 경우, 값은 부분적입니다.

ABAC에 대한 자세한 정보는 IAM 사용 설명서의 ABAC란 무엇인가요?를 참조하세요. ABAC 설정 단계가 포함된 자습서를 보려면 IAM 사용 설명서의 속성 기반 액세스 제어(ABAC) 사용을 참조하세요.

임시 자격 증명 사용: AWS Cloud9

임시 보안 인증 지원	예

임시 자격 증명을 사용하여 로그인하면 작동하지 AWS 서비스 않는 것도 있습니다. 임시 자격 증명을 사용하는 방법을 AWS 서비스 비롯한 추가 정보는 IAM 사용 설명서의 IAM과AWS 서비스 연동되는 내용을 참조하십시오.

사용자 이름과 암호를 제외한 다른 방법을 AWS Management Console 사용하여 로그인하면 임시 자격 증명을 사용하는 것입니다. 예를 들어 회사의 SSO (Single Sign-On) 링크를 AWS 사용하여 액세스하는 경우 이 프로세스에서 자동으로 임시 자격 증명을 생성합니다. 또한 콘솔에 사용자로 로그인한 다음 역할을 전환할 때 임시 보안 인증을 자동으로 생성합니다. 역할 전환에 대한 자세한 정보는 IAM 사용 설명서의 역할로 전환(콘솔)을 참조하세요.

또는 API를 사용하여 임시 자격 증명을 수동으로 생성할 수 있습니다 AWS CLI . AWS 그런 다음 해당 임시 자격 증명을 사용하여 액세스할 수 AWS있습니다. AWS 장기 액세스 키를 사용하는 대신 임시 자격 증명을 동적으로 생성할 것을 권장합니다. 자세한 정보는 IAM의 임시 보안 인증 섹션을 참조하세요.

전달 액세스 세션 대상: AWS Cloud9

전달 액세스 세션(FAS) 지원	예

IAM 사용자 또는 역할을 사용하여 작업을 수행하는 AWS경우 사용자는 보안 주체로 간주됩니다. 일부 서비스를 사용하는 경우 다른 서비스에서 다른 작업을 시작하는 작업을 수행할 수 있습니다. FAS는 전화를 거는 주체의 권한을 다운스트림 서비스에 AWS 서비스 요청하라는 요청과 결합하여 사용합니다. AWS 서비스 FAS 요청은 다른 서비스 AWS 서비스 또는 리소스와의 상호 작용이 필요한 요청을 서비스가 수신한 경우에만 이루어집니다. 이 경우 두 작업을 모두 수행할 수 있는 권한이 있어야 합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.

AWS Cloud9의 서비스 역할

서비스 역할 지원	예

서비스 역할은 서비스가 사용자를 대신하여 작업을 수행하는 것으로 가정하는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 정보는 IAM 사용 설명서의 AWS 서비스에 대한 권한을 위임할 역할 생성을 참조하세요.

> **Warning**
>
> 서비스 역할의 권한을 변경하면 AWS Cloud9 기능이 중단될 수 있습니다. 서비스 역할을 편집하기 위한 지침이 AWS Cloud9 제공되는 경우에만 서비스 역할을 편집하십시오.

서비스 연결 역할은 다음과 같습니다. AWS Cloud9

서비스 링크 역할 지원	예

서비스 연결 역할은 에 연결된 서비스 역할 유형입니다. AWS 서비스서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은 사용자에게 AWS 계정 표시되며 해당 서비스가 소유합니다. IAM 관리자는 서비스 링크 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.

서비스 연결 역할 생성 또는 관리에 대한 자세한 내용은 IAM으로 작업하는AWS 서비스 섹션을 참조하세요. 서비스 연결 역할 열에서 Yes(이)가 포함된 서비스를 테이블에서 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 Yes(네) 링크를 선택합니다.

AWS Cloud9에 대한 자격 증명 기반 정책 예시

기본적으로 사용자 및 역할에는 AWS Cloud9 리소스를 생성하거나 수정할 수 있는 권한이 없습니다. 또한 AWS Management Console, AWS Command Line Interface (AWS CLI) 또는 AWS API를 사용하여 작업을 수행할 수 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다. 그런 다음 관리자가 IAM 정책을 역할에 추가하고, 사용자가 역할을 맡을 수 있습니다.

이러한 예제 JSON 정책 문서를 사용하여 IAM ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성을 참조하세요.

각 리소스 유형의 ARN 형식을 비롯하여 에서 정의한 AWS Cloud9작업 및 리소스 유형에 대한 자세한 내용은 서비스 권한 부여 참조의 작업, 리소스 및 조건 키를 참조하십시오. AWS Cloud9

주제

- 정책 모범 사례
- AWS Cloud9 콘솔 사용
- 사용자가 자신의 고유한 권한을 볼 수 있도록 허용

정책 모범 사례

ID 기반 정책은 누군가가 계정에서 AWS Cloud9 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부를 결정합니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. 자격 증명 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따르십시오.

- AWS 관리형 정책으로 시작하여 최소 권한 권한으로 이동 — 사용자와 워크로드에 권한을 부여하려면 여러 일반적인 사용 사례에 권한을 부여하는 AWS 관리형 정책을 사용하세요. 해당 내용은 에서 사용할 수 있습니다. AWS 계정사용 사례에 맞는 AWS 고객 관리형 정책을 정의하여 권한을 더 줄이는 것이 좋습니다. 자세한 정보는 IAM 사용 설명서의 AWS 관리형 정책 또는 AWS 직무에 대한 관리형 정책을 참조하세요.
- 최소 권한 적용 – IAM 정책을 사용하여 권한을 설정하는 경우 태스크를 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 최소 권한으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용 설명서에 있는 IAM의 정책 및 권한을 참조하세요.
- IAM 정책의 조건을 사용하여 액세스 추가 제한 – 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어 SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정책 조건을 작성할 수 있습니다. 예를 AWS 서비스들어 특정 작업을 통해 서비스 작업을 사용하는 경우 조건을 사용하여 서비스 작업에 대한 액세스 권한을 부여할 수도 AWS CloudFormation있습니다. 자세한 정보는 IAM 사용 설명서의 IAM JSON 정책 요소: 조건을 참조하세요.
- IAM Access Analyzer를 통해 IAM 정책을 검증하여 안전하고 기능적인 권한 보장 – IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 모범 사례가 정책에서 준수되도록 신규 및 기존 정책을 검증합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 추천을 제공하여 안전하고 기능적인 정책을 작성하도록 돕습니다. 자세한 정보는 IAM 사용 설명서의 IAM Access Analyzer 정책 검증을 참조하세요.
- 멀티 팩터 인증 (MFA) 필요 - IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우 추가 보안을 위해 AWS 계정 MFA를 활성화하십시오. API 작업을 직접 호출할 때 MFA가 필요하면 정책에

MFA 조건을 추가합니다. 자세한 정보는 IAM 사용 설명서의 MFA 보호 API 액세스 구성을 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 IAM의 보안 모범 사례를 참조하세요.

AWS Cloud9 콘솔 사용

AWS Cloud9 콘솔에 액세스하려면 최소한의 권한이 있어야 합니다. 이러한 권한을 통해 내 AWS Cloud9 리소스의 세부 정보를 나열하고 볼 수 있어야 AWS 계정합니다. 최소 필수 권한보다 더 제한적인 자격 증명 기반 정책을 만들면 콘솔이 해당 정책에 연결된 엔터티(사용자 또는 역할)에 대해 의도대로 작동하지 않습니다.

AWS CLI 또는 AWS API만 호출하는 사용자에게 최소 콘솔 권한을 허용할 필요는 없습니다. 그 대신, 수행하려는 API 작업과 일치하는 작업에만 액세스할 수 있도록 합니다.

사용자와 역할이 AWS Cloud9 콘솔을 계속 사용할 수 있도록 하려면 엔티티에 AWS Cloud9 *ConsoleAccess* 또는 *ReadOnly* AWS 관리형 정책도 연결하세요. 자세한 내용은 IAM 사용 설명서의 사용자에게 권한 추가를 참조하십시오.

사용자가 자신의 고유한 권한을 볼 수 있도록 허용

이 예시는 IAM 사용자가 자신의 사용자 자격 증명에 연결된 인라인 및 관리형 정책을 볼 수 있도록 허용하는 정책을 생성하는 방법을 보여줍니다. 이 정책에는 콘솔에서 AWS CLI 또는 AWS API를 사용하여 프로그래밍 방식으로 이 작업을 완료할 수 있는 권한이 포함됩니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS Cloud9 ID 및 액세스 문제 해결

다음 정보를 사용하면 및 IAM을 사용할 때 발생할 수 있는 일반적인 문제를 AWS Cloud9 진단하고 해결하는 데 도움이 됩니다.

주제

- 저는 다음과 같은 작업을 수행할 권한이 없습니다. AWS Cloud9
- 저는 IAM을 수행할 권한이 없습니다. PassRole
- 외부 사용자가 내 AWS Cloud9 리소스에 액세스할 수 있도록 AWS 계정 허용하고 싶습니다.

저는 다음과 같은 작업을 수행할 권한이 없습니다. AWS Cloud9

작업을 수행할 권한이 없다는 오류가 수신되면, 작업을 수행할 수 있도록 정책을 업데이트해야 합니다.

다음 예제 오류는 mateojacksonIAM 사용자가 콘솔을 사용하여 가상 *my-example-widget*리소스에 대한 세부 정보를 보려고 하지만 가상 awes:*GetWidget*권한이 없을 때 발생합니다.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 awes:GetWidget on resource: my-example-widget
```

이 경우 awes:*GetWidget*작업을 사용하여 *my-example-widget*리소스에 액세스할 수 있도록 mateojackson사용자 정책을 업데이트해야 합니다.

도움이 필요하면 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

저는 IAM을 수행할 권한이 없습니다. PassRole

`iam:PassRole` 작업을 수행할 수 있는 권한이 없다는 오류가 수신되면 AWS Cloud9에 역할을 전달할 수 있도록 정책을 업데이트해야 합니다.

새 서비스 역할 또는 서비스 연결 역할을 만드는 대신 기존 역할을 해당 서비스에 전달할 AWS 서비스 수 있는 기능도 있습니다. 이렇게 하려면 사용자가 서비스에 역할을 전달할 수 있는 권한을 가지고 있어야 합니다.

다음 예 오류는 `marymajor`라는 IAM 사용자가 콘솔을 사용하여 AWS Cloud9에서 작업을 수행하려고 하는 경우에 발생합니다. 하지만 작업을 수행하려면 서비스 역할이 부여한 권한이 서비스에 있어야 합니다. Mary는 서비스에 역할을 전달할 수 있는 권한을 가지고 있지 않습니다.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

이 경우 Mary가 `iam:PassRole`작업을 수행할 수 있도록 Mary의 정책을 업데이트해야 합니다.

도움이 필요하면 관리자에게 문의하세요. AWS 관리자는 로그인 자격 증명을 제공한 사람입니다.

외부 사용자가 내 AWS Cloud9 리소스에 액세스할 수 있도록 AWS 계정 허용하고 싶습니다.

다른 계정의 사용자 또는 조직 외부의 사람이 리소스에 액세스할 때 사용할 수 있는 역할을 생성할 수 있습니다. 역할을 수임할 신뢰할 수 있는 사람을 지정할 수 있습니다. 리소스 기반 정책 또는 액세스 제어 목록(ACL)을 지원하는 서비스의 경우 이러한 정책을 사용하여 다른 사람에게 리소스에 대한 액세스 권한을 부여할 수 있습니다.

자세히 알아보려면 다음을 참조하세요.

- 이러한 기능의 AWS Cloud9 지원 여부를 알아보려면 을 참조하십시오IAM의 AWS Cloud9 작동 방식.
- 소유한 리소스에 대한 액세스 권한을 AWS 계정 부여하는 방법을 알아보려면 IAM 사용 설명서에서 자신이 소유한 다른 AWS 계정 IAM 사용자에게 액세스 권한 제공을 참조하십시오.
- 제3자에게 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 타사 AWS 계정AWS 계정 소유에 대한 액세스 제공을 참조하십시오.

- ID 페더레이션을 통해 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 외부에서 인증된 사용자에게 액세스 권한 제공(ID 페더레이션)을 참조하세요.
- 크로스 계정 액세스를 위한 역할과 리소스 기반 정책 사용의 차이점을 알아보려면 IAM 사용 설명서의 IAM 역할과 리소스 기반 정책의 차이를 참조하세요.

IAM 리소스 및 운영 활용 방법 AWS Cloud9

AWS Identity and Access Management AWS Cloud9 개발 환경 AWS 서비스 및 기타 리소스 모두를 사용할 수 있는 권한을 관리하는 데 사용됩니다.

AWS Cloud9 리소스 및 운영

AWS Cloud9에서는 기본 리소스가 AWS Cloud9 개발 환경입니다. 정책에서 Amazon 리소스 이름(ARN)을 사용하여 정책이 적용되는 리소스를 식별합니다. 다음 표에는 환경 ARN이 나열되어 있습니다. 자세한 내용은 Amazon Web Services 일반 참조의 Amazon 리소스 이름(ARN) 및 AWS 서비스 네임스페이스를 참조하세요.

리소스 유형	ARN 형식
환경	`arn:aws:cloud9:` *`REGION_ID`* `:`*`ACCOUNT_ID`* `:environment:` *`ENVIRONMENT_ID`*
지정한 AWS 리전에서 지정한 계정이 소유한 모든 환경	`arn:aws:cloud9:` *`REGION_ID`* `:`*`ACCOUNT_ID`* `:environment:*`
지정한 리전 내에 지정된 계정이 소유한 모든 환경	`arn:aws:cloud9:` *`REGION_ID`* `:`*`ACCOUNT_ID`* `:*`
계정과 지역에 상관없이 모든 AWS Cloud9 리소스	`arn:aws:cloud9:*`

예를 들어, 명령문에서 다음과 같이 Amazon 리소스 이름(ARN)을 사용하여 특정 환경을 나타낼 수 있습니다.

```
"Resource": "arn:aws:cloud9:us-
east-2:123456789012:environment:70d899206236474f9590d93b7c41dfEX"
```

모든 리소스를 지정하려면 Resource 요소에 와일드카드 문자(*)를 사용합니다.

```
"Resource": "*"
```

단일 명령문에서 여러 리소스를 지정하려면 Amazon 리소스 이름(ARN)을 쉼표로 구분합니다.

```
"Resource": [
  "arn:aws:cloud9:us-east-2:123456789012:environment:70d899206236474f9590d93b7c41dfEX",
  "arn:aws:cloud9:us-east-2:123456789012:environment:81e900317347585a0601e04c8d52eaEX"
]
```

AWS Cloud9 AWS Cloud9 리소스 작업을 위한 일련의 작업을 제공합니다. 목록은 단원을 참조하세요AWS Cloud9 권한 참조

리소스 소유권 이해

누가 리소스를 만들었든 상관없이 계정에서 생성된 리소스는 계정을 소유합니다. AWS 계정

다음 사용 사례 및 시나리오를 고려합니다.

- 의 루트 계정 자격 증명을 사용하여 AWS Cloud9 개발 환경을 만든다고 AWS 계정 가정해 보겠습니다. 이것은 가능하긴 하지만 권장되지는 않습니다. 이 경우 환경의 소유자는 사용자 AWS 계정 소유자입니다.
- 에서 IAM 사용자를 생성하고 해당 AWS 계정 사용자에게 환경을 생성할 권한을 부여한다고 가정해 보겠습니다. 그러면 사용자가 환경을 생성할 수 있습니다. 하지만 사용자가 속해 있는 YY는 여전히 환경을 소유하고 있습니다. AWS 계정
- 환경을 만들 수 있는 권한을 AWS 계정 가진 IAM 역할을 생성한다고 가정해 보겠습니다. 그러면 이 역할을 수임할 수 있는 자는 누구나 환경을 생성할 수 있습니다. 이 역할이 속한 AWS 계정가 환경을 소유합니다.

> **Note**
>
> 하나 이상의 AWS Cloud9 환경에서 ARN 소유자인 사용자 계정을 삭제하면 이러한 환경은 소유자가 없게 됩니다. 이 시나리오의 해결 방법은 AWS Cloud9 SDK를 사용하여 CreateEnvironmentMembership 작업과 데이터 유형을 사용하여 읽기 및 쓰기 권한이 있는 다른 IAM 사용자를 추가하는 것입니다. EnvironmentMember 이 IAM 사용자를 추가한 후에는 환경 파일을 새 AWS Cloud9 환경에 복사하고 이 소유자를 ARN 소유자로 설정할 수 있습니다. 이 작업에 대한 자세한 내용은 을 참조하십시오 CreateEnvironmentMembership.

이 데이터 유형에 대한 자세한 내용은 AWS Cloud9 API 참조 안내서를 참조하십시오 EnvironmentMember.

리소스 액세스 관리

권한 정책은 누가 어떤 리소스에 액세스 할 수 있는지를 나타냅니다.

> **Note**
>
> 이 섹션에서는 AWS Cloud9에서 IAM을 사용하는 방법에 대해 설명하며, IAM 서비스에 대한 자세한 정보는 다루지 않습니다. IAM 설명서 전체 내용은 IAM 사용 설명서의 IAM이란 무엇입니까? 섹션을 참조하세요. IAM 정책 구문과 설명에 대한 자세한 내용은 IAM 사용 설명서의 IAM JSON 정책 참조를 참조하세요.

IAM 자격 증명에 연결된 정책을 자격 증명 기반 정책(또는 IAM 정책)이라고 합니다. 리소스에 연결된 정책을 리소스 기반 정책이라고 합니다. AWS Cloud9 ID 기반 정책과 리소스 기반 정책을 모두 지원합니다.

다음 각 API 작업을 수행하려면 이러한 API 작업을 호출하려는 IAM 자격 증명에 IAM 정책을 연결하면 됩니다.

- `CreateEnvironmentEC2`
- `DescribeEnvironments`

다음 API 작업을 수행하려면 리소스 기반 정책이 필요합니다. IAM 정책은 필수는 아니지만 이러한 API 작업을 호출하려는 IAM ID에 연결된 경우 IAM 정책을 AWS Cloud9 사용합니다. 리소스 기반 정책을 원하는 리소스에 적용해야 합니다. AWS Cloud9

- `CreateEnvironmentMembership`
- `DeleteEnvironment`
- `DeleteEnvironmentMembership`
- `DescribeEnvironmentMemberships`
- `DescribeEnvironmentStatus`
- `UpdateEnvironment`
- `UpdateEnvironmentMembership`

이러한 각 API 작업의 기능에 관한 자세한 정보는 AWS Cloud9 API 참조를 참조하세요.

리소스 기반 정책을 리소스에 직접 연결할 수는 없습니다. AWS Cloud9 대신 환경 AWS Cloud9 구성원을 추가, 수정, 업데이트 또는 삭제할 때 적절한 리소스 기반 정책을 AWS Cloud9 리소스에 연결합니다.

사용자에게 AWS Cloud9 리소스에서 작업을 수행할 수 있는 권한을 부여하려면 사용자가 속한 IAM 그룹에 권한 정책을 연결합니다. AWS Cloud9 가능하면 AWS 관리형 (사전 정의된) 정책을 연결하는 것이 좋습니다. AWS 관리형 정책에는 일반적인 사용 시나리오 및 사용자 유형 (예: 전체 환경 관리, 환경 사용자, 환경에 대한 읽기 전용 액세스 권한만 있는 사용자) 에 대해 미리 정의된 액세스 권한 집합이 포함되어 있습니다. 에 대한 AWS 관리형 정책 목록은 을 AWS Cloud9참조하십시오. AWS 관리형 정책은 다음과 같습니다. AWS Cloud9

세부적인 사용 시나리오와 고유의 사용자 유형이 필요한 경우 고유의 고객 관리형 정책을 생성하여 연결할 수 있습니다. AWS Cloud9에 대한 추가 설정 옵션(팀 및 엔터프라이즈) 및 에 대한 고객 관리형 정책 생성 AWS Cloud9 단원을 참조하세요.

IAM 정책 (AWS 관리형 또는 고객 관리형) 을 IAM ID에 연결하려면 IAM 사용 설명서의 IAM 정책 연결 (콘솔) 을 참조하십시오.

API 작업에 대한 세션 권한

AWS CLI 또는 AWS API를 사용하여 역할 또는 연동 사용자를 위한 임시 세션을 프로그래밍 방식으로 생성하는 경우 세션 정책을 파라미터로 전달하여 역할 세션의 범위를 확장할 수 있습니다. 즉, 세션의 실질적인 권한은 사용자 또는 역할의 자격 증명 기반 정책의 교집합과 세션 정책입니다.

세션 중에 리소스에 액세스하라는 요청이 전달될 때 세션 정책에 적용 가능한 `Deny` 문이 없는데 적용 가능한 `Allow` 문도 없는 경우 정책 평가의 결과는 묵시적 거부입니다. (자세한 내용은 IAM 사용 설명서에서 계정 내에서 요청 허용 여부 결정을 참조하세요.)

하지만 리소스 기반 정책 (위 참조) 이 필요한 AWS Cloud9 API 작업의 경우, 리소스 정책에 지정된 경우 호출하는 IAM 엔티티에 권한이 부여됩니다. `Principal` 이 명시적 권한은 세션 정책의 암시적 거부보다 우선하므로 세션이 API 작업을 성공적으로 호출할 수 있습니다. AWS Cloud9

AWS 관리형 정책은 다음과 같습니다. AWS Cloud9

AWS 관리형 정책은 에서 생성하고 관리하는 독립형 정책입니다. AWS AWS 관리형 정책은 많은 일반 사용 사례에 대한 권한을 제공하도록 설계되었으므로 사용자, 그룹 및 역할에 권한을 할당하기 시작할 수 있습니다.

AWS 관리형 정책은 모든 AWS 고객이 사용할 수 있으므로 특정 사용 사례에 대해 최소 권한 권한을 부여하지 않을 수도 있다는 점에 유의하세요. 사용 사례에 고유한 고객 관리형 정책을 정의하여 권한을 줄이는 것이 좋습니다.

관리형 정책에 정의된 권한은 변경할 수 없습니다. AWS AWS 관리형 정책에 정의된 권한을 업데이트하는 경우 AWS 해당 업데이트는 정책이 연결된 모든 주체 ID (사용자, 그룹, 역할) 에 영향을 미칩니다. AWS 새 API 작업이 시작되거나 기존 서비스에 새 AWS 서비스 API 작업을 사용할 수 있게 되면 AWS 관리형 정책을 업데이트할 가능성이 가장 높습니다.

자세한 내용은 IAM 사용자 설명서의 AWS 관리형 정책을 참조하세요.

AWS 관리형 정책: AWSCloud9Administrator

`AWSCloud9Administrator` 정책을 IAM 보안 인증에 연결할 수 있습니다.

 # ### ##### ### ### #### ### ### AWS Cloud9#####.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- AWS Cloud9 — 해당 AWS Cloud9 정책의 모든 작업 AWS 계정.
- Amazon EC2 — 여러 Amazon VPC 및 해당 리소스의 서브넷 리소스에 대한 정보를 가져옵니다. AWS 계정
- IAM — IAM 사용자의 IAM 사용자에 대한 정보를 얻고 AWS 계정필요에 따라 AWS Cloud9 서비스 연결 역할을 생성합니다. AWS 계정
- Systems Manager - 사용자가 전화를 걸어 StartSession 세션 관리자 세션의 인스턴스에 대한 연결을 시작할 수 있도록 허용합니다. 이 권한은 Systems Manager를 통해 EC2 인스턴스와 통신하는 환경을 여는 사용자에게 필요합니다. 자세한 내용은 AWS Systems Manager를 사용하여 수신하지 않는 EC2 인스턴스에 액세스 섹션을 참조하세요.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloud9:*",
                "iam:GetUser",
                "iam:ListUsers",
```

```
                "ec2:DescribeVpcs",
                "ec2:DescribeSubnets",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeRouteTables"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "cloud9.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession",
                "ssm:GetConnectionStatus"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:cloud9:environment": "*"
                },
                "StringEquals": {
                    "aws:CalledViaFirst": "cloud9.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:document/*"
            ]
        }
```

```
    ]
}
```

AWS 관리형 정책: AWSCloud9User

`AWSCloud9User` 정책을 IAM 보안 인증에 연결할 수 있습니다.

이 정책은 AWS Cloud9 개발 환경을 생성하고 소유한 환경을 관리할 ### 권한을 부여합니다.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- AWS Cloud9 — 환경에 대한 정보를 생성 및 가져오고 해당 환경에 대한 사용자 설정을 가져오고 변경합니다.
- Amazon EC2 — 여러 Amazon VPC 및 해당 리소스의 서브넷 리소스에 대한 정보를 가져옵니다. AWS 계정
- IAM — IAM 사용자의 IAM 사용자에 대한 정보를 얻고 AWS 계정필요에 따라 AWS Cloud9 서비스 연결 역할을 생성합니다. AWS 계정
- Systems Manager - 사용자가 전화를 걸어 StartSession 세션 관리자 세션의 인스턴스에 대한 연결을 시작할 수 있도록 허용합니다. 이 권한은 Systems Manager를 통해 EC2 인스턴스와 통신하는 환경을 여는 사용자에게 필요합니다. 자세한 내용은 AWS Systems Manager를 사용하여 수신하지 않는 EC2 인스턴스에 액세스 섹션을 참조하세요.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloud9:UpdateUserSettings",
                "cloud9:GetUserSettings",
                "iam:GetUser",
                "iam:ListUsers",
                "ec2:DescribeVpcs",
                "ec2:DescribeSubnets",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeRouteTables"
            ],
            "Resource": "*"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloud9:CreateEnvironmentEC2",
                "cloud9:CreateEnvironmentSSH"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "cloud9:OwnerArn": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloud9:GetUserPublicKey"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "cloud9:UserArn": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloud9:DescribeEnvironmentMemberships"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "Null": {
                    "cloud9:UserArn": "true",
                    "cloud9:EnvironmentId": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "cloud9.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession",
                "ssm:GetConnectionStatus"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:cloud9:environment": "*"
                },
                "StringEquals": {
                    "aws:CalledViaFirst": "cloud9.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:document/*"
            ]
        }
    ]
}
```

AWS 관리형 정책: AWSCloud9EnvironmentMember

`AWSCloud9EnvironmentMember` 정책을 IAM 보안 인증에 연결할 수 있습니다.

이 정책은 AWS Cloud9 공유 환경에 참여할 수 있는 권한을 제공하는 ### 권한을 부여합니다.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- AWS Cloud9 — 해당 환경에 대한 정보를 얻고 해당 환경에 대한 사용자 설정을 가져오고 변경합니다.
- IAM — IAM 사용자에 대한 정보를 얻고 필요에 따라 해당 AWS 계정사용자에게 AWS Cloud9 서비스 연결 역할을 생성합니다. AWS 계정
- Systems Manager - 사용자가 전화를 걸어 StartSession 세션 관리자 세션의 인스턴스에 대한 연결을 시작할 수 있도록 허용합니다. 이 권한은 Systems Manager를 통해 EC2 인스턴스와 통신하는 환경을 여는 사용자에게 필요합니다. 자세한 내용은 AWS Systems Manager를 사용하여 수신하지 않는 EC2 인스턴스에 액세스 섹션을 참조하세요.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloud9:GetUserSettings",
                "cloud9:UpdateUserSettings",
                "iam:GetUser",
                "iam:ListUsers"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloud9:DescribeEnvironmentMemberships"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "Null": {
                    "cloud9:UserArn": "true",
                    "cloud9:EnvironmentId": "true"
                }
            }
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "ssm:StartSession",
                "ssm:GetConnectionStatus"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:cloud9:environment": "*"
                },
                "StringEquals": {
                    "aws:CalledViaFirst": "cloud9.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:document/*"
            ]
        }
    ]
}
```

AWS 관리형 정책: **AWSCloud9ServiceRolePolicy**

서비스 연결 역할 AWSServiceRoleForAWSCloud9는 이 정책을 사용하여 AWS Cloud9 환경이 Amazon AWS CloudFormation EC2 및 리소스와 상호 작용하도록 허용합니다.

권한 세부 정보

는 AWSServiceRoleForAWSCloud 9명에게 개발 환경을 만들고 실행하는 AWS Cloud9 데 필요한 AWS 서비스 (Amazon EC2 및 AWS CloudFormation) 와 상호 작용하는 데 필요한 권한을 AWSCloud9ServiceRolePolicy부여합니다.

AWS Cloud9 서비스 연결 역할의 권한을 정의하며 해당 역할만 AWS Cloud9 수임할 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며 이 권한 정책은 다른 IAM 엔터티에 연결할 수 없습니다.

서비스 연결 역할을 AWS Cloud9 사용하는 방법에 대한 자세한 내용은 을 참조하십시오. AWS Cloud9에 서비스 연결 역할 사용

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Action": [
    "ec2:RunInstances",
    "ec2:CreateSecurityGroup",
    "ec2:DescribeVpcs",
    "ec2:DescribeSubnets",
    "ec2:DescribeSecurityGroups",
    "ec2:DescribeInstances",
    "ec2:DescribeInstanceStatus",
    "cloudformation:CreateStack",
    "cloudformation:DescribeStacks",
    "cloudformation:DescribeStackEvents",
    "cloudformation:DescribeStackResources"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:TerminateInstances",
    "ec2:DeleteSecurityGroup",
    "ec2:AuthorizeSecurityGroupIngress"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "cloudformation:DeleteStack"
   ],
   "Resource": "arn:aws:cloudformation:*:*:stack/aws-cloud9-*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:CreateTags"
   ],
   "Resource": [
    "arn:aws:ec2:*:*:instance/*",
```

```
    "arn:aws:ec2:*:*:security-group/*"
   ],
   "Condition": {
    "StringLike": {
     "aws:RequestTag/Name": "aws-cloud9-*"
    }
   }
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:StartInstances",
    "ec2:StopInstances"
   ],
   "Resource": "*",
   "Condition": {
    "StringLike": {
     "ec2:ResourceTag/aws:cloudformation:stack-name": "aws-cloud9-*"
    }
   }
  },
  {
   "Effect": "Allow",
   "Action": [
    "ec2:StartInstances",
    "ec2:StopInstances"
   ],
   "Resource": [
    "arn:aws:license-manager:*:*:license-configuration:*"
   ]
  },
  {
   "Effect": "Allow",
   "Action": [
    "iam:ListInstanceProfiles",
    "iam:GetInstanceProfile"
   ],
   "Resource": [
    "arn:aws:iam::*:instance-profile/cloud9/*"
   ]
  },
  {
   "Effect": "Allow",
   "Action": [
```

```
    "iam:PassRole"
   ],
   "Resource": [
    "arn:aws:iam::*:role/service-role/AWSCloud9SSMAccessRole"
   ],
   "Condition": {
    "StringLike": {
     "iam:PassedToService": "ec2.amazonaws.com"
    }
   }
  }
 ]
}
```

AWS Cloud9 관리형 정책 업데이트 AWS

이 서비스가 이러한 변경 사항을 추적하기 시작한 AWS Cloud9 이후의 AWS 관리형 정책 업데이트에 대한 세부 정보를 볼 수 있습니다. 이 페이지의 변경 사항에 대한 자동 알림을 받으려면 AWS Cloud9 문서 기록 페이지에서 RSS 피드를 구독하십시오.

변경 사항	설명	날짜
AWSCloud9Administrator및 AWSCloud9EnvironmentMember정책에 새 작업이 추가되었습니다 AWSCloud9User.	`ssm:GetConnectionStatus` 작업이 AWSCloud9UserAWSCloud9Administrator및 AWSCloud9EnvironmentMember정책에 추가되었습니다. 이 작업을 수행하면 사용자에게 SSM 연결 상태를 확인할 수 있는 권한이 부여됩니다. `cloud9:ValidateEnvironmentName` API는 더 이상 사용되지 않으므로 AWSCloud9User정책에서 제거되었습니다.	2023년 10월 12일
API가 AWSCloud9User및 AWSCloud9Administrator정책에 추가되었습니다.	AWSCloud9User및 AWSCloud9Administrator정책에 두 개의 새 API가 추가되었	2023년 8월 2일

변경 사항	설명	날짜
	습니다. 이 API는 `ec2:DescribeInstanceTypeOfferings` 및 `ec2:DescribeRouteTables` 입니다. 이러한 API의 목적은 기본 서브넷이 AWS Cloud9 환경을 만들 때 고객이 선택한 인스턴스 유형을 지원하는지 검증할 수 있도록 AWS Cloud9 하는 것입니다.	
로 업데이트 AWSCloud9ServiceRolePolicy	AWSCloud9ServiceRolePolicyLicense Manager 라이선스 AWS Cloud9 구성으로 관리되는 Amazon EC2 인스턴스를 시작하고 중지할 수 있도록 업데이트되었습니다.	2022년 1월 12일
AWS Cloud9 변경 내용 추적 시작	AWS Cloud9 AWS 관리형 정책의 변경 사항 추적을 시작했습니다.	2021년 3월 15일

에 대한 고객 관리형 정책 생성 AWS Cloud9

액세스 제어 요구 사항을 충족하는 AWS 관리형 정책이 없는 경우 자체 고객 관리형 정책을 생성하여 첨부할 수 있습니다.

고객 관리형 정책을 생성하려면 IAM 사용 설명서에서 IAM 정책 생성(콘솔)을 참조하세요.

주제

- 정책 요소 지정: 효과, 보안 주체, 작업 및 리소스
- 고객 관리형 정책 예

정책 요소 지정: 효과, 보안 주체, 작업 및 리소스

서비스는 각 AWS Cloud9 리소스에 대해 일련의 API 작업을 정의합니다. 이러한 API 작업에 대한 권한을 부여하려면 정책에서 지정할 수 있는 작업 세트를 AWS Cloud9 정의하십시오.

다음은 기본 정책 요소입니다.

- `Effect` - 사용자가 작업을 요청하는 경우 효과(허용 또는 거부)를 지정합니다. 명시적으로 리소스에 대한 액세스 권한을 부여(허용)하지 않는 경우, 액세스는 묵시적으로 거부됩니다. 리소스에 대한 액세스를 명시적으로 거부할 수도 있습니다. 다른 정책에서 액세스 권한을 부여하더라도 사용자가 해당 리소스에 액세스할 수 없도록 하려고 할 때 이러한 작업을 수행할 수 있습니다.
- `Principal` - 자격 증명 기반 정책(IAM 정책)에서 정책이 연결되는 사용자는 암시적인 보안 주체입니다. 리소스 기반 정책의 경우 사용자, 계정, 서비스 또는 권한의 수신자인 기타 엔터티를 지정합니다.
- `Resource` – Amazon 리소스 이름(ARN)을 사용하여 정책을 적용할 리소스를 식별합니다.
- `Action` - 작업 키워드를 사용하여 허용 또는 거부할 리소스 작업을 식별합니다. 예를 들어, `cloud9:CreateEnvironmentEC2` 권한은 사용자에게 `CreateEnvironmentEC2` 작업 수행 권한을 제공합니다.

IAM 정책 구문과 설명에 대한 자세한 내용은 IAM 사용 설명서에서 IAM JSON 정책 참조를 참조하세요.

모든 AWS Cloud9 API 작업과 해당 작업이 적용되는 리소스를 보여주는 표는 를 참조하십시오AWS Cloud9 권한 참조.

고객 관리형 정책 예

이 단원에서는 AWS Cloud9 작업에 대한 권한을 부여하는 정책의 예를 볼 수 있습니다. IAM 자격 증명에 대한 AWS Cloud9 액세스를 허용하거나 명시적으로 거부하도록 다음 IAM 정책 예를 조정할 수 있습니다.

고객 관리형 정책을 만들거나 IAM 자격 증명에 연결하려면 IAM 사용 설명서에서 IAM 정책 생성(콘솔) 및 IAM 정책 연결(콘솔)을 참조하세요.

> **Note**
>
> 다음 예에서는 미국 동부 (오하이오) 지역 (us-east-2), 가상 AWS 계정 ID (123456789012) 및 가상 AWS Cloud9 개발 환경 ID () 를 사용합니다. 81e900317347585a0601e04c8d52eaEX

주제

환경에 대한 정보 가져오기

IAM 엔터티에 연결된 다음 예제 IAM 정책 문은 해당 엔터티가 계정의 환경에 대한 정보를 가져오도록 허용합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:DescribeEnvironments",
```

```
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> 이전 액세스 권한은 이미 관리형 정책 및 에 포함되어 있습니다. AWS `AWSCloud9Administrator` `AWSCloud9User`

EC2 환경 생성

IAM 엔티티에 첨부된 다음 예시 IAM 정책 설명을 사용하면 해당 엔티티가 자신의 계정에 AWS Cloud9 EC2 개발 환경을 생성할 수 있습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> 이전 액세스 권한은 관리형 정책 및 에 이미 포함되어 있습니다. AWS `AWSCloud9Administrator` `AWSCloud9User`

특정 Amazon EC2 인스턴스 유형으로 EC2 환경 생성

IAM 엔티티에 첨부된 다음 예시 IAM 정책 설명을 사용하면 해당 엔티티가 자신의 계정에 AWS Cloud9 EC2 개발 환경을 생성할 수 있습니다. 하지만 EC2 환경은 Amazon EC2 인스턴스 유형의 지정된 클래스만 사용할 수 있습니다.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "cloud9:InstanceType": "t3.*"
        }
      }
    }
  ]
}
```

> **Note**
>
> AWS 관리형 정책 `AWSCloud9Administrator` 또는 `AWSCloud9User` 이 IAM 엔티티에 이미 연결되어 있는 경우 해당 AWS 관리형 정책은 이전 IAM 정책 설명의 동작보다 우선합니다. 이는 이러한 AWS 관리형 정책이 더 관대하기 때문입니다.

특정 Amazon VPC 서브넷에서 EC2 환경 생성

IAM 엔티티에 첨부된 다음 예시 IAM 정책 설명을 사용하면 해당 엔티티가 자신의 계정에 AWS Cloud9 EC2 개발 환경을 만들 수 있습니다. 하지만 EC2 환경은 지정된 Amazon VPC 서브넷만 사용할 수 있습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "cloud9:SubnetId": [
            "subnet-12345678",
            "subnet-23456789"
          ]
        }
```

```
      }
    }
  ]
}
```

> **Note**
>
> AWS 관리형 정책 `AWSCloud9Administrator` 또는 `AWSCloud9User` 이 IAM 엔티티에 이미 연결되어 있는 경우 해당 AWS 관리형 정책은 이전 IAM 정책 설명의 동작보다 우선합니다. 이는 이러한 AWS 관리형 정책이 더 관대하기 때문입니다.

특정 환경 이름으로 EC2 환경 생성

IAM 엔티티에 첨부된 다음 예시 IAM 정책 설명은 해당 엔티티가 자신의 계정에 AWS Cloud9 EC2 개발 환경을 만들 수 있도록 허용합니다. 하지만 EC2 환경에는 지정된 이름만 사용할 수 있습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "cloud9:EnvironmentName": "my-demo-environment"
        }
      }
    }
  ]
}
```

> **Note**
>
> AWS 관리형 정책 `AWSCloud9Administrator` 또는 `AWSCloud9User` 이 IAM 엔티티에 이미 연결되어 있는 경우 해당 AWS 관리형 정책은 이전 IAM 정책 설명의 동작보다 우선합니다. 이는 이러한 AWS 관리형 정책이 더 관대하기 때문입니다.

SSH 환경만 생성

IAM 개체에 첨부된 다음 예시 IAM 정책 설명을 사용하면 해당 개체가 자신의 계정에 AWS Cloud9 SSH 개발 환경을 만들 수 있습니다. 하지만 엔티티는 AWS Cloud9 EC2 개발 환경을 만들 수 없습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentSSH",
      "Resource": "*"
    },
    {
      "Effect": "Deny",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*"
    }
  ]
}
```

환경 업데이트 또는 환경 업데이트 금지

IAM 엔티티에 첨부된 다음 예시 IAM 정책 설명을 사용하면 해당 엔티티가 계정의 모든 AWS Cloud9 개발 환경에 대한 정보를 변경할 수 있습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:UpdateEnvironment",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> 이전 액세스 권한은 이미 관리형 정책에 포함되어 있습니다. AWS `AWSCloud9Administrator`

IAM 엔터티에 연결된 다음 예제 IAM 정책 문은 해당 엔터티가 Amazon 리소스 이름(ARN)이 지정된 환경에 관한 정보를 변경하는 것을 명시적으로 금지합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "cloud9:UpdateEnvironment",
      "Resource": "arn:aws:cloud9:us-
east-2:123456789012:environment:81e900317347585a0601e04c8d52eaEX"
    }
  ]
}
```

환경 멤버 목록 가져오기

IAM 엔터티에 연결된 다음 예제 IAM 정책 문은 해당 엔터티가 계정의 환경에 대한 멤버의 목록을 가져오도록 허용합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:DescribeEnvironmentMemberships",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> 이전 액세스 권한은 이미 AWS 관리형 정책에 포함되어 있습니다. `AWSCloud9Administrator` 또한 이전 액세스 권한은 관리형 정책의 해당 액세스 권한보다 더 관대합니다. AWS `AWSCloud9User`

특정 사용자만 환경 공유

IAM 엔터티에 연결된 다음 예제 IAM 정책 문은 해당 엔터티가 계정의 환경을 지정된 사용자만 공유하도록 허용합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloud9:CreateEnvironmentMembership"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "cloud9:UserArn": "arn:aws:iam::123456789012:user/MyDemoUser"
        }
      }
    }
  ]
}
```

> **Note**
>
> AWS 관리형 정책 `AWSCloud9Administrator` 또는 `AWSCloud9User` 이 IAM 엔티티에 이미 연결되어 있는 경우 해당 AWS 관리형 정책은 이전 IAM 정책 설명의 동작보다 우선합니다. 이는 이러한 AWS 관리형 정책이 더 관대하기 때문입니다.

환경 공유 금지

IAM 엔터티에 연결된 다음 예제 IAM 정책 문은 해당 엔터티가 계정에서 환경을 공유하는 것을 금지합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "cloud9:CreateEnvironmentMembership",
        "cloud9:UpdateEnvironmentMembership"
      ],
      "Resource": "*"
    }
  ]
}
```

환경 멤버의 설정 변경 또는 변경 금지

IAM 엔터티에 연결된 다음 예제 IAM 정책 문은 해당 엔터티가 계정의 환경에 있는 멤버의 설정을 변경하도록 허용합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:UpdateEnvironmentMembership",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> 이전 액세스 권한은 이미 관리형 정책에 포함되어 있습니다. AWS `AWSCloud9Administrator`

IAM 엔터티에 연결된 다음 예제 IAM 정책 문은 해당 엔터티가 Amazon 리소스 이름(ARN)이 지정된 환경에 있는 멤버의 설정을 변경하는 것을 명시적으로 금지합니다.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "cloud9:UpdateEnvironmentMembership",
      "Resource": "arn:aws:cloud9:us-
east-2:123456789012:environment:81e900317347585a0601e04c8d52eaEX"
    }
  ]
}
```

환경 멤버 제거 또는 제거 금지

IAM 엔터티에 연결된 다음 예제 IAM 정책 문은 해당 엔터티가 계정의 환경에서 멤버를 제거하도록 허용합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:DeleteEnvironmentMembership",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> 이전 액세스 권한은 이미 AWS 관리형 정책에 포함되어 있습니다. `AWSCloud9Administrator`

IAM 엔터티에 연결된 다음 예제 IAM 정책 문은 해당 엔터티가 Amazon 리소스 이름(ARN)이 지정된 환경에서 멤버를 제거하는 것을 명시적으로 금지합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "cloud9:DeleteEnvironmentMembership",
```

```
      "Resource": "arn:aws:cloud9:us-
east-2:123456789012:environment:81e900317347585a0601e04c8d52eaEX"
    }
  ]
}
```

환경 삭제 또는 삭제 금지

IAM 엔터티에 연결된 다음 예제 IAM 정책 문은 해당 엔터티가 계정에서 환경을 삭제하도록 허용합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:DeleteEnvironment",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> 이전 액세스 권한은 이미 AWS 관리형 정책에 포함되어 있습니다. `AWSCloud9Administrator`

IAM 엔터티에 연결된 다음 예제 IAM 정책 문은 해당 엔터티가 Amazon 리소스 이름(ARN)이 지정된 환경을 삭제하는 것을 명시적으로 금지합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "cloud9:DeleteEnvironment",
      "Resource": "arn:aws:cloud9:us-
east-2:123456789012:environment:81e900317347585a0601e04c8d52eaEX"
    }
  ]
```

```
}
```

SSM 환경 생성을 위한 사용자 지정 IAM 정책

현재 AWSCloud9Administrator 또는 AWSCloud9User 정책이 연결된 SSM 환경을 생성할 때 발생하는 권한 문제가 있습니다. 다음 예시 IAM 정책 설명을 IAM 개체에 연결하면 사용자가 AWS 관리형 정책 또는 중 하나를 연결하여 사용할 수 있습니다. AWSCloud9Administrator AWSCloud9User

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloud9:UpdateUserSettings",
                "cloud9:GetUserSettings",
                "iam:GetUser",
                "iam:ListUsers",
                "iam:ListRoles",
                "ec2:DescribeVpcs",
                "ec2:DescribeSubnets",
                "ec2:DescribeRouteTables"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloud9:CreateEnvironmentEC2",
                "cloud9:CreateEnvironmentSSH"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "cloud9:OwnerArn": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloud9:GetUserPublicKey"
            ],
```

```
            "Resource": "*",
            "Condition": {
                "Null": {
                    "cloud9:UserArn": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloud9:DescribeEnvironmentMemberships"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "Null": {
                    "cloud9:UserArn": "true",
                    "cloud9:EnvironmentId": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "cloud9.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ssm:StartSession",
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:cloud9:environment": "*"
                },
                "StringEquals": {
                    "aws:CalledViaFirst": "cloud9.amazonaws.com"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:document/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": ["iam:ListInstanceProfilesForRole", "iam:CreateRole"],
            "Resource": ["arn:aws:iam::*:role/service-role/AWSCloud9SSMAccessRole"]
        },
        {
            "Effect": "Allow",
            "Action": ["iam:AttachRolePolicy"],
            "Resource": ["arn:aws:iam::*:role/service-role/AWSCloud9SSMAccessRole"],
            "Condition": {
                "StringEquals": {
                    "iam:PolicyARN": "arn:aws:iam::aws:policy/
AWSCloud9SSMInstanceProfile"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/service-role/AWSCloud9SSMAccessRole",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateInstanceProfile",
                "iam:AddRoleToInstanceProfile"
            ],
```

```
            "Resource": [
                "arn:aws:iam::*:instance-profile/cloud9/AWSCloud9SSMInstanceProfile"
            ]
        }
    ]
}
```

AWS Cloud9 권한 참조

AWS Cloud9 정책에서 AWS 광범위한 조건 키를 사용하여 조건을 표현할 수 있습니다. 목록은 IAM 사용 설명서에서 IAM JSON 정책 요소: 조건을 참조하세요.

정책의 `Action` 필드에 작업을 지정합니다. 작업을 지정하려면 `cloud9:` 접두사 다음에 API 작업 명칭을 사용합니다(예: `"Action": "cloud9:DescribeEnvironments"`). 문장 하나에 여러 작업을 지정하려면 쉼표로 구분합니다(예: `"Action": [ "cloud9:UpdateEnvironment", "cloud9:DeleteEnvironment" ]`).

와일드카드 문자 사용

정책의 `Resource` 필드에 리소스 값으로 와일드카드 문자(*)를 사용하거나 사용하지 않고 ARN을 지정합니다. 와일드카드를 사용하여 여러 작업 또는 리소스를 지정할 수 있습니다. 예를 들어, `cloud9:*`는 모든 AWS Cloud9 작업을 지정하고 로 시작하는 모든 AWS Cloud9 작업을 `cloud9:Describe*` 지정합니다`Describe`.

다음 예제는 IAM 엔터티가 계정에 있는 환경 및 환경 멤버십에 대한 정보를 가져오도록 허용합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloud9:Describe*"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> 이전 액세스 권한은 이미 AWS 관리형 정책에 `AWSCloud9Administrator` 포함되어 있습니다. 또한 이전 액세스 권한은 관리형 정책의 동등한 액세스 권한보다 더 관대합니다. AWS `AWSCloud9User`

AWS Cloud9 API 작업 및 작업에 필요한 권한

> **Note**
>
> IAM 자격 증명에 연결할 권한 정책(자격 증명 기반 정책)을 작성하고 액세스 제어를 설정할 때 아래의 표를 참조로 사용할 수 있습니다.
> Public API operations 표에는 SDK와 AWS Command Line Interface를 사용하는 고객이 호출할 수 있는 API 작업이 나열되어 있습니다.
> Permission-only API operations에는 고객 코드 또는 AWS Command Line Interface에서 직접 호출 할 수 없는 API 작업을 나열되어 있습니다. 그러나 IAM 사용자는 콘솔을 사용하여 AWS Cloud9 작업을 수행하는 경우에 호출되는 이러한 작업에 대한 권한이 필요합니다.

퍼블릭 API 작업

AWS Cloud9 오퍼레이션	필요한 권한(API 작업)	Resource
`CreateEnvironmentEC2`	`cloud9:CreateEnvironmentEC2` AWS Cloud9 EC2 개발 환경을 만드는 데 필요합니다.	*
`CreateEnvironmentMembership`	`cloud9:CreateEnvironmentMembership` 환경에 멤버를 추가하는 데 필요합니다.	`arn:aws:cloud9:` *REGION_ID* `:` *ACCOUNT_ID* `:environment:` *ENVIRONMENT_ID*
`DeleteEnvironment`	`cloud9:DeleteEnvironment`	`arn:aws:cloud9:` *REGION_ID*

AWS Cloud9 오퍼레이션	필요한 권한(API 작업)	Resource
	환경을 삭제하는 데 필요합니다.	:*ACCOUNT_ID* :environment: *ENVIRONMENT_ID*
DeleteEnvironmentMembership	cloud9:DeleteEnvironmentMembership 환경에서 멤버를 제거하는 데 필요합니다.	arn:aws:cloud9: *REGION_ID* :*ACCOUNT_ID* :environment: *ENVIRONMENT_ID*
DescribeEnvironmentMemberships	cloud9:DescribeEnvironmentMemberships 환경에서 멤버 목록을 가져오는 데 필요합니다.	*
DescribeEnvironments	cloud9:DescribeEnvironments 환경에 대한 정보를 가져오는 데 필요합니다.	arn:aws:cloud9: *REGION_ID* :*ACCOUNT_ID* :environment: *ENVIRONMENT_ID*
DescribeEnvironmentStatus	cloud9:DescribeEnvironmentStatus 환경의 상태에 대한 정보를 가져오는 데 필요합니다.	arn:aws:cloud9: *REGION_ID* :*ACCOUNT_ID* :environment: *ENVIRONMENT_ID*
UpdateEnvironment	cloud9:UpdateEnvironment 환경에 대한 설정을 업데이트하는 데 필요합니다.	arn:aws:cloud9: *REGION_ID* :*ACCOUNT_ID* :environment: *ENVIRONMENT_ID*
UpdateEnvironmentMembership	cloud9:UpdateEnvironmentMembership 환경의 멤버에 대한 설정을 업데이트하는 데 필요합니다.	arn:aws:cloud9: *REGION_ID* :*ACCOUNT_ID* :environment: *ENVIRONMENT_ID*

권한 전용 API 작업

AWS Cloud9 운영	설명	콘솔 문서
`ActivateEC2Remote`	`cloud9:ActivateEC2Remote` AWS Cloud9 IDE가 연결되는 Amazon EC2 인스턴스를 시작합니다.	AWS Cloud9에서 환경 열기
`CreateEnvironmentSSH`	`cloud9:CreateEnvironmentSSH` AWS Cloud9 SSH 개발 환경을 만듭니다.	SSH 환경 생성
`CreateEnvironmentToken`	`cloud9:CreateEnvironmentToken` AWS Cloud9 IDE 및 사용자 환경 사이의 연결을 허용하는 인증 토큰을 생성합니다.	EC2 환경 생성
`DescribeEC2Remote`	`cloud9:DescribeEC2Remote` 호스트, 사용자 및 포트를 포함하여 EC2 개발 환경 연결에 대한 세부 정보를 가져옵니다.	EC2 환경 생성
`DescribeSSHRemote`	`cloud9:DescribeSSHRemote` 호스트, 사용자 및 포트를 포함하여 SSH 개발 환경 연결에 대한 세부 정보를 가져옵니다.	SSH 환경 생성
`GetEnvironmentConfig`	`cloud9:GetEnvironmentConfig`	AWS Cloud9 통합 개발 환경(IDE) 작업

AWS Cloud9 운영	설명	콘솔 문서
	AWS Cloud9 IDE를 초기화하는 데 사용되는 구성 정보를 가져옵니다.	
`GetEnvironmentSettings`	`cloud9:GetEnvironmentSettings` 지정된 개발 환경의 AWS Cloud9 IDE 설정을 가져옵니다.	AWS Cloud9 통합 개발 환경 (IDE) 작업
`GetMembershipSettings`	`cloud9:GetMembershipSettings` 지정된 환경 구성원의 AWS Cloud9 IDE 설정을 가져옵니다.	AWS Cloud9의 공유 환경 작업
`GetUserPublicKey`	`cloud9:GetUserPublicKey` SSH 개발 환경에 연결하는 AWS Cloud9 데 사용되는 사용자의 공개 SSH 키를 가져옵니다.	SSH 환경 생성
`GetUserSettings`	`cloud9:GetUserSettings` 지정된 사용자의 AWS Cloud9 IDE 설정을 가져옵니다.	AWS Cloud9 통합 개발 환경 (IDE) 작업

AWS Cloud9 운영	설명	콘솔 문서
`ModifyTemporaryCredentialsOnEnvironmentEC2`	`cloud9:ModifyTemporaryCredentialsOnEnvironmentEC2` AWS Cloud9 통합 개발 환경(IDE)에서 사용하는 Amazon EC2 인스턴스에 AWS 관리형 임시 자격 증명을 설정합니다.	AWS 관리형 임시 자격 증명
`UpdateEnvironmentSettings`	`cloud9:UpdateEnvironmentSettings` 지정된 개발 환경을 위한 AWS Cloud9 IDE 설정을 업데이트합니다.	AWS Cloud9 통합 개발 환경(IDE) 작업
`UpdateMembershipSettings`	`cloud9:UpdateMembershipSettings` 지정된 환경 구성원의 AWS Cloud9 IDE 설정을 업데이트합니다.	AWS Cloud9의 공유 환경 작업
`UpdateSSHRemote`	`cloud9:UpdateSSHRemote` 호스트, 사용자 및 포트를 포함하여 SSH 개발 환경 연결에 대한 세부 정보를 업데이트합니다.	SSH 환경 생성
`UpdateUserSettings`	`cloud9:UpdateUserSettings` 지정된 사용자의 AWS Cloud9 IDE 설정을 업데이트합니다.	AWS Cloud9 통합 개발 환경(IDE) 작업

AWS Cloud9 운영	설명	콘솔 문서
`GetMigrationExperiences`	`cloud9:GetMigrationExperiences` AWS Cloud9 사용자에게 에서 AWS Cloud9 로 마이그레이션 환경을 가져올 수 있는 권한을 CodeCatalyst 부여합니다.	

AWS 관리형 임시 자격 증명

AWS 관리형 임시 자격 증명이 지원하는 작업 목록을 찾고 있다면 다음으로 건너뛰세요. AWS 관리형 임시 자격 증명이 지원하는 작업

AWS Cloud9 EC2 개발 환경의 경우 환경에서 임시 AWS Cloud9 AWS 액세스 자격 증명을 사용할 수 있도록 합니다. 이러한 자격 증명을 AWS 관리형 임시 자격 증명이라고 합니다. 이 기능에는 다음과 같은 이점이 있습니다.

- AWS 엔티티 (예: IAM 사용자) 의 영구 AWS 액세스 자격 증명을 환경 어디에도 저장할 필요가 없습니다. 따라서 사용자가 모르게 승인 없이 환경 멤버가 해당 자격 증명에 액세스할 수 없습니다.
- 인스턴스 프로파일을 수동으로 설정하거나, 관리하거나, 환경에 연결되는 Amazon EC2 인스턴스에 연결할 필요가 없습니다. 인스턴스 프로필은 임시 AWS 액세스 자격 증명을 관리하는 또 다른 방법입니다.
- AWS Cloud9 임시 자격 증명을 지속적으로 갱신하므로 단일 자격 증명 세트를 제한된 기간 동안만 사용할 수 있습니다. 이는 AWS 보안 모범 사례입니다. 자세한 정보는 AWS 관리형 임시 자격 증명 생성 및 업데이트을 참조하세요.
- AWS Cloud9 임시 자격 증명을 사용하여 환경의 AWS 작업 및 리소스에 액세스하는 방법을 추가로 제한합니다. 이는 AWS 보안 모범 사례이기도 합니다.

Important

현재 환경의 EC2 인스턴스를 프라이빗 서브넷에서 시작하는 경우 AWS 관리형 임시 자격 증명을 사용하여 EC2 환경이 AWS 엔티티 (예: IAM 사용자) 를 대신하여 AWS 서비스에 액세스하도록 허용할 수 없습니다.
프라이빗 서브넷에서 EC2 인스턴스를 시작할 수 있는 시나리오에 대한 자세한 내용은 에 대한 서브넷을 생성하십시오. AWS Cloud9 섹션을 참조하세요.

Note

AWS 관리형 임시 자격 증명을 사용할 때는 인라인 정책 대신 관리형 정책을 사용하는 것이 좋습니다. AWS

EC2 환경이 AWS 엔티티 (예: IAM 사용자) 를 AWS 서비스 대신하여 액세스를 시도할 때마다 AWS 관리형 임시 자격 증명이 작동하는 방식은 다음과 같습니다.

1. AWS Cloud9 호출 AWS 주체 (예: IAM 사용자) 가 요청된 리소스에 대해 요청된 작업을 수행할 권한이 있는지 확인합니다. AWS권한이 없거나 명시적으로 거부되면 요청이 실패합니다.
2. AWS Cloud9 AWS 관리되는 임시 자격 증명을 검사하여 해당 권한이 요청된 리소스에 AWS대해 요청된 작업을 허용하는지 확인합니다. 권한이 없거나 명시적으로 거부되면 요청이 실패합니다. 임시 자격 증명 지원을 AWS 관리하는 권한 목록은 을 참조하십시오AWS 관리형 임시 자격 증명이 지원하는 작업.

- AWS 엔티티와 AWS 관리형 임시 자격 증명이 모두 요청된 리소스에 대해 요청된 작업을 허용하면 요청이 성공합니다.
- AWS 엔티티 또는 AWS 관리형 임시 자격 증명이 요청된 리소스에 대해 요청된 작업을 명시적으로 거부하거나 명시적으로 허용하지 않는 경우 요청은 실패합니다. 즉, 호출 AWS 엔티티에 올바른 권한이 있더라도 명시적으로 AWS Cloud9 허용하지 않으면 요청이 실패합니다. 마찬가지로 특정 리소스에 대해 특정 작업을 수행하도록 AWS Cloud9 허용하는 경우 AWS 엔티티에서 명시적으로 허용하지 않으면 요청이 실패합니다.

EC2 환경 소유자는 다음과 같이 언제든지 해당 환경에 대한 AWS 관리형 임시 자격 증명을 켜거나 끌 수 있습니다.

1. 환경이 열린 상태에서 AWS Cloud9 IDE의 메뉴 표시줄에서 기본 설정을 선택합니다AWS Cloud9.
2. 기본 설정 탭의 탐색 창에서 AWS 설정, 자격 증명을 선택합니다.
3. AWS 관리형 임시 자격 증명를 사용하여 AWS 관리형 임시 자격 증명을 켜거나 끕니다.

> **Note**
>
> AWS Cloud9 API 작업을 `UpdateEnvironment`호출하고 `managedCredentialsAction` 파라미터에 값을 할당하여 AWS 관리형 임시 자격 증명을 켜거나 끌 수도 있습니다. AWS SDK 및 와 같은 표준 AWS 도구를 사용하여 이 API 작업을 요청할 수 있습니다. AWS CLI

AWS 관리형 임시 자격 증명을 끄면 요청한 AWS 엔티티에 관계없이 환경에서 아무 자격 증명에도 AWS 서비스액세스할 수 없습니다. 하지만 환경에 대한 AWS 관리형 임시 자격 증명을 활성화할 수 없거나 활성화하고 싶지 않은데 AWS 서비스액세스하려면 환경이 여전히 필요하다고 가정해 보겠습니다. 다음과 같은 대안을 고려하세요.

- 환경에 연결되는 Amazon EC2 인스턴스에 인스턴스 프로파일을 연결합니다. 지침은 인스턴스 프로파일을 생성하고 사용하여 임시 자격 증명 관리를 참조하세요.
- 예를 들어 특수 환경 변수를 설정하거나 `aws configure` 명령을 실행하여 영구 AWS 액세스 자격 증명을 환경에 저장하십시오. 지침은 영구 액세스 자격 증명을 생성하여 환경에 저장을 참조하세요.

위의 대안은 EC2 환경에서 AWS 관리형 임시 자격 증명이 허용(또는 거부)하는 모든 권한을 재정의합니다.

AWS 관리형 임시 자격 증명이 지원하는 작업

AWS Cloud9 EC2 개발 환경의 경우 AWS 관리형 임시 자격 증명은 호출자의 AWS 계정모든 AWS 리소스에 대한 모든 AWS 작업을 허용하지만 다음과 같은 제한이 있습니다.

- 의 AWS Cloud9경우 다음 작업만 허용됩니다.
  - `cloud9:CreateEnvironmentEC2`
  - `cloud9:CreateEnvironmentSSH`
  - `cloud9:DescribeEnvironmentMemberships`
  - `cloud9:DescribeEnvironments`
  - `cloud9:DescribeEnvironmentStatus`
  - `cloud9:UpdateEnvironment`

- IAM의 경우 다음 작업만 허용됩니다.
  - iam:AttachRolePolicy
  - iam:ChangePassword
  - iam:CreatePolicy
  - iam:CreatePolicyVersion
  - iam:CreateRole
  - iam:CreateServiceLinkedRole
  - iam:DeletePolicy
  - iam:DeletePolicyVersion
  - iam:DeleteRole
  - iam:DeleteRolePolicy
  - iam:DeleteSSHPublicKey
  - iam:DetachRolePolicy
  - iam:GetInstanceProfile
  - iam:GetPolicy
  - iam:GetPolicyVersion
  - iam:GetRole
  - iam:GetRolePolicy
  - iam:GetSSHPublicKey
  - iam:GetUser
  - iam:List*
  - iam:PassRole
  - iam:PutRolePolicy
  - iam:SetDefaultPolicyVersion
  - iam:UpdateAssumeRolePolicy
  - iam:UpdateRoleDescription
  - iam:UpdateSSHPublicKey
  - iam:UploadSSHPublicKey
- 역할과 상호 작용하는 모든 IAM 작업은 Cloud9-으로 시작하는 역할 이름에만 허용됩니다. 하지만 iam:PassRole은 모든 역할 이름에 작동합니다.

- AWS Security Token Service (AWS STS) 의 경우 다음 작업만 허용됩니다.
  - `sts:GetCallerIdentity`
  - `sts:DecodeAuthorizationMessage`
- 지원되는 모든 AWS 작업은 환경의 IP 주소로 제한됩니다. 이는 AWS 보안 모범 사례입니다.

액세스하기 위해 EC2 환경이 필요한 작업 또는 리소스를 AWS Cloud9 지원하지 않는 경우 또는 EC2 환경에서 AWS 관리형 임시 자격 증명이 꺼져 있는데 다시 켤 수 없는 경우 다음 대안을 고려해 보십시오.

- EC2 환경에 연결되는 Amazon EC2 인스턴스에 인스턴스 프로파일을 연결합니다. 지침은 인스턴스 프로파일을 생성하고 사용하여 임시 자격 증명 관리을 참조하세요.
- 예를 들어 특수 환경 변수를 설정하거나 명령을 실행하여 영구 AWS 액세스 자격 증명을 EC2 환경에 저장합니다. `aws configure` 지침은 영구 액세스 자격 증명을 생성하여 환경에 저장을 참조하세요.

위의 대안은 EC2 환경에서 AWS 관리형 임시 자격 증명이 허용(또는 거부)하는 모든 권한을 재정의합니다.

AWS 관리형 임시 자격 증명 생성 및 업데이트

AWS Cloud9 EC2 개발 환경의 경우 환경을 처음 열 때 AWS 관리형 임시 자격 증명이 생성됩니다.

AWS 관리형 임시 자격 증명은 다음 조건 중 하나에서 업데이트됩니다.

- 특정 기간이 경과할 때마다. 현재, 이 값은 5분입니다.
- 환경에 대한 IDE가 표시된 웹 브라우저 탭을 다시 로드할 때마다.
- 환경에 대한 `~/.aws/credentials` 파일에 나열된 타임스탬프에 도달할 때.
- AWS managed temporary credentials(관리형 임시 자격 증명) 설정이 꺼져 있는 경우 다시 켤 때마다. (이 설정을 보거나 변경하려면 IDE의 메뉴 모음에서 AWS Cloud9, Preferences(기본 설정)을 선택합니다. 탐색 창의 Preferences(기본 설정) 탭에서 AWS Settings, Credentials(설정, 자격 증명)을 선택합니다.)
- 보안을 위해 AWS 관리형 임시 자격 증명은 15분 후에 자동으로 만료됩니다. 자격 증명을 새로 고치려면 환경 소유자가 IDE를 통해 AWS Cloud9 환경에 연결해야 합니다. 환경 소유자의 역할에 대한 자세한 내용은 AWS 관리형 임시 자격 증명에 대한 액세스 제어 섹션을 참조하세요.

AWS 관리형 임시 자격 증명에 대한 액세스 제어

AWS 관리형 임시 자격 증명을 보유한 공동 작업자는 다른 사람과 상호 작용하는 AWS Cloud9 데 사용할 수 있습니다. AWS 서비스신뢰할 수 있는 공동 작업자만 AWS 관리형 임시 자격 증명을 사용하도록 하기 위해, 환경 소유자가 아닌 다른 사람이 새 멤버를 추가한 경우 이러한 자격 증명을 사용할 수 없습니다. 자격 증명은 `~/.aws/credentials` 파일을 삭제하면 사용 중지됩니다.

> **Important**
>
> AWS 또한 관리형 임시 자격 증명은 15분마다 자동으로 만료됩니다. 공동 작업자가 자격 증명을 계속 사용할 수 있도록 자격 증명을 새로 고치려면 환경 소유자가 IDE를 통해 AWS Cloud9 환경에 연결되어 있어야 합니다.

환경 소유자만 AWS 관리형 임시 자격 증명을 다시 활성화하여 다른 구성원과 공유할 수 있습니다. 환경 소유자가 IDE를 열면 AWS 관리형 임시 자격 증명이 비활성화되었는지 확인하는 대화 상자가 나타납니다. 환경 소유자는 모든 멤버에 대해 자격 증명을 다시 사용하도록 설정하거나 모든 멤버에 대해 자격 증명을 사용 중지할 수 있습니다.

> **Warning**
>
> 보안 모범 사례를 준수하려면 환경에 마지막으로 추가된 사용자의 ID가 확실하지 않은 경우 관리형 임시 자격 증명을 사용하지 않도록 설정합니다. 읽기/쓰기 권한이 있는 멤버 목록은 [협업(Collaborate)] 창에서 확인할 수 있습니다.

로그인 및 모니터링 AWS Cloud9

다음을 통한 활동 모니터링 CloudTrail

AWS Cloud9 에서 사용자 AWS CloudTrail, 역할 또는 서비스가 수행한 작업의 기록을 제공하는 AWS 서비스와 통합됩니다 AWS Cloud9. CloudTrail 모든 API 호출을 AWS Cloud9 이벤트로 캡처합니다. 캡처된 호출에는 AWS Cloud9 콘솔에서의 호출 및 AWS Cloud9 API에 대한 코드 호출이 포함됩니다.

트레일을 생성하면 Amazon Simple Storage Service (Amazon S3) 버킷으로 CloudTrail 이벤트를 지속적으로 전송할 수 있습니다. 여기에는 에 대한 이벤트가 포함됩니다. AWS Cloud9

트레일을 구성하지 않아도 CloudTrail 콘솔의 이벤트 기록에서 가장 최근 이벤트를 계속 볼 수 있습니다. 에서 수집한 CloudTrail 정보를 사용하여 요청을 받은 사람 AWS Cloud9, 요청한 IP 주소, 요청한 사람, 요청 시기 및 추가 세부 정보를 확인할 수 있습니다.

자세한 정보는 AWS CloudTrail을 사용하여 AWS Cloud9 API 호출 로깅을 참조하세요.

EC2 환경 성능 모니터링

AWS Cloud9 EC2 개발 환경을 사용하는 경우 관련 Amazon EC2 인스턴스의 안정성, 가용성 및 성능을 모니터링할 수 있습니다. 예를 들어 인스턴스 상태 모니터링 작업은 Amazon EC2가 인스턴스의 애플리케이션 실행에 지장을 줄 수 있는 문제를 발견했을 때 빠르게 확인할 수 있는 방법입니다.

자세한 내용은 Amazon EC2 사용 설명서의 Amazon EC2 모니터링을 참조하십시오.

규정 준수 검증: AWS Cloud9

제3자 감사자는 여러 규정 AWS 준수 프로그램의 일환으로 AWS 서비스의 보안 및 규정 준수를 평가합니다.

AWS Cloud9 다음 규정 준수 프로그램의 적용 범위입니다.

SOC

AWS 시스템 및 조직 규제 (SOC) 보고서는 주요 규정 준수 규제 항목 및 AWS 목표를 달성하는 방법을 보여주는 독립적인 제3자 검토 보고서입니다.

Service	SDK	SOC 1,2,3
AWS Cloud9	cloud9	✓

PCI

결제 카드 산업 데이터 보안 표준 (PCI DSS) 은 아메리칸 익스프레스, 디스커버 파이낸셜 서비스, JCB 인터내셔널, 월드와이드 및 비자 주식회사가 설립한 PCI 보안 표준 위원회에서 관리하는 독점적인 정보 보안 표준입니다. MasterCard

Service	SDK	PCI
AWS Cloud9	cloud9	✓

FedRAMP

연방정부 위험 및 권한 부여 관리 프로그램(FedRAMP)은 클라우드 제품 및 서비스의 보안 평가, 권한 부여 및 지속적인 모니터링에 대한 표준 접근 방식을 제공하는 정부 차원의 프로그램입니다.

FedRAMP 평가 및 권한 부여를 거치는 서비스는 다음과 같은 상태를 갖습니다.

- 서드 파티 평가 조직(3PAO) 평가: 이 서비스는 현재 서드 파티 평가자에 의해 평가를 받는 중입니다.
- 공동 권한 부여 위원회(JAB) 검토: 이 서비스는 현재 JAB의 검토를 받는 중입니다.

Service	SDK	FedRAMP Moderate(동부/서부)	페드램프 하이 () GovCloud
AWS Cloud9	cloud9	JAB 검토	N/A

DoD CC SRG

국방부(DoD) 클라우드 컴퓨팅 보안 요구 사항 안내서(SRG)에서는 DoD 고객에게 서비스를 제공하기 위해 클라우드 서비스 제공업체(CSP)가 DoD 임시 인증을 획득할 수 있도록 표준화된 평가 및 인증 프로세스를 제공합니다.

DoD CC SRG 평가 및 권한 부여를 거치는 서비스는 다음과 같은 상태를 갖습니다.

- 서드 파티 평가 조직(3PAO) 평가: 이 서비스는 현재 서드 파티 평가자에 의해 평가를 받는 중입니다.
- 공동 권한 부여 위원회(JAB) 검토: 이 서비스는 현재 JAB의 검토를 받는 중입니다.
- 국방 정보 시스템 기관(DISA) 검토: 이 서비스는 현재 DISA의 검토를 받는 중입니다.

Service	SDK	DoD CC SRG IL2(동부/서부)	DoD CC SRG IL2 () GovCloud	DoD CC SRG IP4 () GovCloud	DoD CC SRG IP5 () GovCloud	DoD CC SRG IL6 (시크릿 리전)AWS
AWS Cloud9	cloud9	JAB 검토	N/A	해당 사항 없음	해당 사항 없음	N/A

HIPAA BAA

1996년에 제정된 미국 건강 보험 양도 및 책임에 관한 법(HIPAA)은 환자의 동의나 지식 없이 민감한 환자 건강 정보가 공개되지 않도록 방지하기 위해 국가 표준의 수립을 요구한 연방법입니다.

AWS HIPAA의 적용을 받는 피보험 대상 단체 및 비즈니스 동료는 보호 대상 의료 정보 (PHI) 를 안전하게 처리, 저장 및 전송할 수 있습니다. 또한 2013년 7월부터 이러한 고객을 위해 표준화된 비즈니스 제휴 부록 (BAA) 을 AWS 제공합니다.

Service	SDK	HIPAA BAA
AWS Cloud9	cloud9	✓

IRAP

IRAP(Information Security Registered Assessors Program)를 통해 호주 정부 고객은 적절한 제어가 이루어지는지 검증하고 호주 사이버 보안 센터(ACSC)에서 제작한 호주 정부 ISM(Information Security Manual)의 요구 사항을 해결하기 위한 적절한 책임 모델을 결정할 수 있습니다.

Service	네임스페이스*	IRAP 보호
AWS Cloud9	cloud9	✓

*네임스페이스는 환경 전반의 서비스를 식별하는 데 도움이 됩니다. AWS 예를 들어, IAM 정책을 생성하고 Amazon 리소스 이름 (ARN) 으로 작업하고 로그를 읽을 AWS CloudTrail 때.

C5

C5(Cloud Computing Compliance Controls Catalogue)는 연방 정보 보안 사무소(BSI)가 독일에서 도입한 독일 정부 지원 증명 체계로서, 독일 정부의 '클라우드 공급자를 위한 보안 권장 사항'에 따라 클라우드 서비스를 사용할 때 일반적인 사이버 공격에 대한 운영 보안을 입증할 수 있도록 돕습니다.

Service	SDK	C5
AWS Cloud9	cloud9	✓

FINMA

FINMA는 스위스의 독립적인 금융 시장 규제 기관입니다. Amazon Web Services(AWS)는 FINMA ISAE 3000 Type 2 Report를 완료했습니다.

Service	SDK	FINMA
AWS Cloud9	cloud9	✓

GSMA

GSM 협회는 전 세계 모바일 네트워크 사업자의 이익을 대표하는 산업 조직입니다. Amazon Web Services(AWS) 유럽(파리) 및 미국 동부(오하이오) 리전은 현재 데이터 센터 운영 및 관리(DCOM) 범위에서 보안 인증 체계 구독 관리(SAS-SM)에 따라 GSM 협회(GSMA)의 인증을 받았습니다. 이러한 GSMA 요구 사항 준수는 클라우드 서비스 공급자에 대한 높아진 기대치에 부응하려는 지속적인 노력을 보여줍니다.

Service	미국 동부(오하이오)	유럽(파리)
AWS Cloud9	✓	✓

PiTuKri

AWS PiTuKri 요구 사항을 준수한다는 것은 핀란드 교통통신국인 Traficom이 설정한 클라우드 서비스 공급자에 대한 높아진 기대치를 충족하기 위한 당사의 지속적인 노력을 보여줍니다.

Service	SDK	PiTuKri
AWS Cloud9	cloud9	✓

AWS 서비스 a가 특정 규정 준수 프로그램의 범위 내에 있는지 알아보려면AWS 서비스 규정 준수 프로그램의 범위별 범위 내 규정 준수 프로그램의AWS 서비스 관심 있는 규정 준수 프로그램을 선택하십시오. 일반 정보는 AWS 규정 준수 프로그램AWS 보증 프로그램 규정AWS 참조하십시오.

를 사용하여 AWS Artifact타사 감사 보고서를 다운로드할 수 있습니다. 자세한 내용은 의 보고서 https://docs.aws.amazon.com/artifact/latest/ug/downloading-documents.html 참조하십시오 AWS Artifact.

사용 시 규정 준수 AWS 서비스 책임은 데이터의 민감도, 회사의 규정 준수 목표, 관련 법률 및 규정에 따라 결정됩니다. AWS 규정 준수에 도움이 되는 다음 리소스를 제공합니다.

- 보안 및 규정 준수 퀵 스타트 가이드 - 이 배포 가이드에서는 아키텍처 고려 사항을 설명하고 보안 및 규정 준수에 AWS 중점을 둔 기본 환경을 배포하기 위한 단계를 제공합니다.
- Amazon Web Services의 HIPAA 보안 및 규정 준수를 위한 설계 — 이 백서에서는 기업이 HIPAA 적격 애플리케이션을 만드는 AWS 데 사용할 수 있는 방법을 설명합니다.

> **Note**
>
> 모든 AWS 서비스 사람이 HIPAA 자격을 갖춘 것은 아닙니다. 자세한 내용은 HIPAA 적격 서비스 참조를 참조하십시오.

- AWS 규정 준수 리소스AWS — 이 워크북 및 가이드 모음은 해당 산업 및 지역에 적용될 수 있습니다.
- AWS 고객 규정 준수 가이드 — 규정 준수의 관점에서 공동 책임 모델을 이해하십시오. 이 가이드에서는 보안을 유지하기 위한 모범 사례를 AWS 서비스 요약하고 여러 프레임워크 (미국 표준 기술 연구소 (NIST), 결제 카드 산업 보안 표준 위원회 (PCI), 국제 표준화기구 (ISO) 등) 에서 보안 제어에 대한 지침을 매핑합니다.
- AWS Config 개발자 안내서의 규칙을 사용하여 리소스 평가 — 이 AWS Config 서비스는 리소스 구성이 내부 관행, 업계 지침 및 규정을 얼마나 잘 준수하는지 평가합니다.
- AWS Security Hub— 이를 AWS 서비스 통해 내부 AWS보안 상태를 포괄적으로 파악할 수 있습니다. Security Hub는 보안 제어를 사용하여 AWS 리소스를 평가하고 보안 업계 표준 및 모범 사례에 대한 규정 준수를 확인합니다. 지원되는 서비스 및 제어 목록은 Security Hub 제어 참조를 참조하십시오.
- Amazon GuardDuty — 환경에 의심스럽고 악의적인 활동이 있는지 AWS 계정모니터링하여 워크로드, 컨테이너 및 데이터에 대한 잠재적 위협을 AWS 서비스 탐지합니다. GuardDuty 특정 규정 준수 프레임워크에서 요구하는 침입 탐지 요구 사항을 충족하여 PCI DSS와 같은 다양한 규정 준수 요구 사항을 해결하는 데 도움이 될 수 있습니다.
- AWS Audit Manager— 이를 AWS 서비스 통해 AWS 사용량을 지속적으로 감사하여 위험을 관리하고 규정 및 업계 표준을 준수하는 방법을 단순화할 수 있습니다.

의 레질리언스 AWS Cloud9

AWS 글로벌 인프라는 AWS 지역 및 가용 영역을 중심으로 구축됩니다. AWS 지역은 물리적으로 분리되고 격리된 여러 가용 영역을 제공하며, 이러한 가용 영역은 지연 시간이 짧고 처리량이 높으며 중복성이 높은 네트워킹으로 연결됩니다. 가용 영역을 사용하면 중단 없이 영역 간에 자동으로 장애 극복 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 지역 및 가용 영역에 대한 자세한 내용은 글로벌 인프라를 참조하십시오AWS .

AWS 글로벌 인프라 외에도 데이터 복원력 및 백업 요구 사항을 AWS Cloud9 지원하는 특정 기능을 지원합니다.

- Amazon Web Services에서 호스팅하는 버전 제어 서비스와 AWS Cloud9 통합하여 클라우드의 자산 (예: 문서, 소스 코드, 바이너리 파일) 을 비공개로 저장하고 관리하는 데 사용할 수 있습니다. AWS CodeCommit자세한 내용은 AWS CodeCommit 사용 AWS CodeCommit설명서의 AWS Cloud9 통합을 참조하십시오.
- AWS Cloud9 개발 환경에서 Git 버전 제어 시스템을 사용하여 원격 GitHub 리포지토리의 파일 및 데이터를 백업할 수 있습니다. 자세한 정보는 Git 패널로 시각적 소스 제어을 참조하세요.

의 인프라 보안 AWS Cloud9

관리형 서비스로서 AWS 글로벌 네트워크 보안으로 AWS Cloud9 보호됩니다. AWS 보안 서비스 및 인프라 AWS 보호 방법에 대한 자세한 내용은 AWS 클라우드 보안을 참조하십시오. 인프라 보안 모범 사례를 사용하여 AWS 환경을 설계하려면 Security Pillar AWS Well-Architected Framework의 인프라 보호를 참조하십시오.

AWS 게시된 API 호출을 사용하여 네트워크를 통해 액세스할 AWS Cloud9 수 있습니다. 고객은 다음을 지원해야 합니다.

- 전송 계층 보안(TLS) TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군 Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

또한 요청은 액세스 키 ID 및 IAM 주체와 관련된 비밀 액세스 키를 사용하여 서명해야 합니다. 또는 AWS Security Token Service(AWS STS)를 사용하여 임시 보안 인증을 생성하여 요청에 서명할 수 있습니다.

> **Note**
>
> 기본적으로 AWS Cloud9 EC2 개발 환경은 인스턴스의 시스템 패키지에 대한 보안 패치를 자동으로 설치합니다.

소프트웨어 업데이트 및 패치 적용

AWS Cloud9 개발 환경은 클라우드 컴퓨팅 리소스를 기반으로 실행됩니다. 클라우드 컴퓨팅 리소스는 EC2 환경의 Amazon EC2 인스턴스일 수 있으며, SSH 환경의 자체 클라우드 컴퓨팅 리소스일 수 있습니다. 두 옵션에 대한 자세한 내용은 환경 및 컴퓨팅 리소스 섹션을 참조하십시오.

AWS Cloud9 EC2 환경은 환경이 시작된 후 운영 체제 보안 패치와 업데이트를 자동으로 설치합니다. AWS Cloud9 또한 환경에는 IDE 기능이 작동하고 지원하는 AWS Cloud9 데 필요한 소프트웨어 패키지도 포함되어 있습니다. 이러한 패키지는 환경이 로드될 때 자동으로 패치됩니다. 특정 개발 도구가 AWS Cloud9 환경에 사전 설치되어 있습니다. AWS Cloud9 AMI에서 이러한 도구를 업데이트하지만 사용자 환경에서 자동으로 업데이트하지는 않습니다. 이러한 도구를 업데이트하는 방법에 대한 자세한 내용은 아래에 설명된 섹션을 참조하십시오.

- AWS Command Line Interface 사용 설명서의 AWS CLI의 최신 버전 설치 또는 업데이트.
- AWS Serverless Application Model 개발자 AWS SAM 안내서의 CLI 버전 관리
- AWS Cloud Development Kit (AWS CDK) 개발자 안내서의 AWS CDK설치

기본 클라우드 컴퓨팅 리소스나 자동 업데이트 빈도에 관계없이 클라우드 컴퓨팅 리소스에 패치를 적용하고 최신 상태로 유지하는 것은 여전히 AWS Cloud9 사용자 또는 AWS Cloud9 관리자의 책임입니다.

고객의 책임에 대한 자세한 내용은 공동 책임 모델에서 데이터 보호: AWS Cloud9 섹션을 참조하세요.

에 대한 보안 모범 사례 AWS Cloud9

다음 모범 사례는 일반적인 지침이며 완벽한 보안 솔루션을 나타내지는 않습니다. 이러한 모범 사례는 환경에 적절하지 않거나 충분하지 않을 수 있으므로 참고용으로만 사용해 주십시오.

에 대한 몇 가지 보안 모범 사례 AWS Cloud9

- 버전 제어 시스템(예: AWS CodeCommit)에 안전하게 코드를 저장합니다.
- AWS Cloud9 EC2 개발 환경에서는 Amazon Elastic Block Store의 암호화된 볼륨을 구성하고 사용하십시오.
- EC2 환경의 경우 태그를 사용하여 AWS Cloud9 리소스에 대한 액세스를 제어합니다.
- 공유 AWS Cloud9 개발 환경의 경우 해당 공유 개발 환경의 모범 사례를 따르세요.

문제 해결 AWS Cloud9

다음 정보를 사용하여 문제를 식별하고 해결하십시오 AWS Cloud9.

해당 문제가 나와 있지 않거나 추가 도움이 필요한 경우, AWS Cloud9 토론 포럼을 참조하세요. 이 포럼에 들어갈 때 로그인해야 할 수 있습니다. 또한 직접 당사에 문의할 수도 있습니다.

주제

설치 관리자

다음 섹션에서는 AWS Cloud9 설치 프로그램과 관련된 문제 해결 문제를 간략하게 설명합니다.

AWS Cloud9 설치 프로그램이 중단되거나 실패합니다.

문제: AWS Cloud9 설치 프로그램을 다운로드하여 실행하면 오류가 하나 이상 발생하고 설치 스크립트가 표시되지 않습니다. `Done`

원인: AWS Cloud9 설치 프로그램에서 복구할 수 없는 오류가 하나 이상 발생하여 오류가 발생했습니다.

해결 방법: AWS Cloud9 설치 프로그램 문제 해결 섹션에서 자세한 정보를 확인합니다. 제공된 일반적인 문제, 가능한 원인 및 권장 해결 방법을 참조하세요.

AWS Cloud9 “패키지 Cloud9 IDE 1"이 표시된 후 설치 프로그램이 종료되지 않음

문제: AWS Cloud9 SSH 개발 환경 생성 프로세스의 일부로 기존 Amazon EC2 인스턴스 또는 자체 서버에 설치되었습니다. AWS Cloud9 설치 관리자 대화 상자에 "Package Cloud9 IDE 1" 메시지가 표시

된 경우에는 설치가 중지됩니다. 취소를 선택하면 "Installation Failed(설치 실패)" 메시지가 표시됩니다. 이 오류는 고객의 SSH 호스트에 AWS Cloud9 패키지를 설치할 수 없을 때 발생합니다.

원인: SSH 호스트에는 Node.js가 설치되어 있어야 합니다. 호스트 운영 체제에서 지원하는 최신 Node.js 버전을 설치하는 것이 좋습니다. Node.js호스트에 지원되지 AWS Cloud9 않는 버전이 있는 경우 설치 오류가 발생할 수 있습니다.

권장 해결 방법: SSH 호스트에서 AWS Cloud9 지원하는 Node.js 버전을 설치하십시오.

종속성을 설치하지 못함

문제: 종속 항목을 다운로드하려면 인터넷 액세스가 AWS Cloud9 필요합니다.

가능한 원인:

- 프록시를 사용하여 인터넷에 액세스하는 AWS Cloud9 환경에서 사용하는 경우 종속성을 설치하려면 프록시 세부 정보가 AWS Cloud9 필요합니다. 에 프록시 세부 정보를 제공하지 않은 AWS Cloud9경우 이 오류가 나타납니다.
- 이 문제의 또 다른 원인은 환경에서 아웃바운드 트래픽을 허용하지 않는 경우일 수 있습니다.

권장 솔루션:

- 프록시 세부 정보를 AWS Cloud9제공하려면 환경 `~/.bashrc` 파일에 다음 코드를 추가하십시오.

```
export http_proxy=[proxy url for http]
export https_proxy=[proxy url for https]
#Certificate Authority used by your proxy
export NODE_EXTRA_CA_CERTS=[path_to_pem_certificate]
```

예를 들어 HTTP 프록시 URL이 `https://172.31.26.80:3128`이고 HTTP 프록시 URL이 `https://172.31.26.80:3129`인 경우 `~/.bashrc` 파일에 다음 줄을 추가하고 `NODE_EXTRA_CA_CERTS`를 PEM 형식의 인증 기관 파일 경로로 설정합니다. 이 변수에 대한 자세한 내용은 https://nodejs.org/api/cli.html#node_extra_ca_certsfile 섹션을 참조하세요.

```
export http_proxy=http://172.31.26.80:3128
export https_proxy=https://172.31.26.80:3129
export NODE_EXTRA_CA_CERTS=[path_to_pem_certificate]
```

- 무수신 Amazon EC2 인스턴스를 사용하는 경우 Amazon S3용 Amazon VPC 엔드포인트가 구성되어 있는지 확인해야 합니다. 자세한 내용은 종속 구성 요소를 다운로드하도록 Amazon S3의 VPC 엔드포인트 구성을 참조하세요.

SSH 환경 오류: 'pty.js를 설치하려면 Python 버전 3이 필요함'

문제: AWS Cloud9 SSH 개발 환경을 연 후 AWS Cloud9 IDE의 터미널에 "pty.js 설치하려면 Python 버전 3이 필요합니다."로 시작하는 메시지가 표시됩니다.

원인: 예상대로 작동하도록 하려면 SSH 환경에 Python 버전 3이 설치되어 있어야 합니다.

솔루션: 환경에 Python 버전 3을 설치합니다. 버전을 확인하려면 서버의 터미널에서 **`python --version`** 명령을 실행하십시오. 서버에 Python 3을 설치하려면 다음 중 하나를 참조하십시오.

- Python 샘플의 1단계: Python 설치
- Python 웹 사이트에서 Python을 다운로드하십시오.

AWS Cloud9 환경

다음 섹션에서는 AWS Cloud9 환경과 관련된 문제 해결 문제를 간략하게 설명합니다.

환경 생성 오류: ‘EC2 인스턴스를 생성할 수 없습니다.(We are unable to create EC2 instances ...)’

문제: AWS Cloud9 개발 환경을 만들려고 하면 “계정 확인 및 활성화 중에 계정에 EC2 인스턴스를 생성할 수 없습니다.” 라는 메시지가 표시됩니다.

원인: 현재 확인 및 활성화 AWS 중입니다. AWS 계정활성화가 완료될 때까지 최대 24시간이 소요될 수 있으며, 그 전에는 이 환경이나 다른 환경을 생성할 수 없습니다.

솔루션: 환경을 나중에 다시 생성해 보세요. 24시간이 지난 후에도 이 메시지가 계속 표시되면 지원에 문의하세요. 또한 환경을 생성하려는 시도가 실패할 경우에도 AWS CloudFormation 에서 계정에 관련 스택을 생성한다는 점에 유의해야 합니다. 이러한 스택은 계정의 스택 생성 할당량에 포함됩니다. 스택 생성 할당량을 소모하지 않도록 이 같은 실패한 스택을 삭제할 수 있습니다. 자세한 정보는 AWS CloudFormation 사용 설명서의 AWS CloudFormation 콘솔에서 스택 삭제를 참조하세요.

환경 생성 오류: “sts를 수행할 권한이 없음:” AssumeRole

문제: 새 환경을 만들려고 하면 “sts를 수행할 권한이 없음:AssumeRole”이라는 오류가 표시되고 환경이 생성되지 않습니다.

가능한 원인: AWS Cloud9 서비스 연결 역할이 사용자 계정에 존재하지 않습니다. AWS 계정

권장 해결 방법: 에서 AWS Cloud9 서비스 연결 역할을 생성하십시오. AWS 계정 AWS Command Line Interface (AWS CLI)또는 AWS CloudShell에서 다음 명령을 실행하면 이 작업을 수행할 수 있습니다.

```
aws iam create-service-linked-role --aws-service-name cloud9.amazonaws.com # For the
 AWS CLI.
iam create-service-linked-role --aws-service-name cloud9.amazonaws.com     # For the
 aws-shell.
```

이렇게 할 수 없는 경우 관리자에게 문의하세요. AWS 계정

이 명령을 실행한 후 환경을 다시 생성해 보세요.

페더레이션 ID로 환경을 만들 수 없음

문제: AWS 페더레이션 ID를 사용하여 AWS Cloud9 개발 환경을 만들려고 하면 액세스 오류 메시지가 표시되고 환경이 생성되지 않습니다.

원인:: 서비스 AWS Cloud9 연결 역할을 사용합니다. 서비스 연결 역할은 `iam:CreateServiceLinkedRole` 호출을 사용하여 계정에서 환경이 처음 생성될 때 생성됩니다. 그러나 페더레이션 사용자는 IAM API를 호출할 수 없습니다. 자세한 내용은 AWS Security Token Service API GetFederationToken참조를 참조하십시오.

해결 방법: AWS 계정 관리자에게 IAM 콘솔에서 AWS Cloud9 또는 () 와 함께 이 명령을 실행하여 서비스 연결 역할을 생성하도록 AWS Command Line Interface 요청하십시오.AWS CLI

```
aws iam create-service-linked-role --aws-service-name cloud9.amazonaws.com
```

또는 -shell을 사용하여 다음 명령을 실행하십시오. AWS

```
iam create-service-linked-role --aws-service-name cloud9.amazonaws.com
```

자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 사용을 참조하세요.

콘솔 오류: '사용자가 리소스에 대한 작업을 수행하도록 승인되지 않음(User is not authorized to perform action on resource)'

문제: AWS Cloud9 콘솔을 사용하여 AWS Cloud9 개발 환경을 만들거나 관리하려고 하면 "사용자가 `arn:aws:iam::123456789012:user/MyUser` `cloud9:action` `arn:aws:cloud9:us-east-2:123456789012:environment:12a34567b8cd9012345ef67abcd890e1` 리소스에 대한 작업을 수행할 권한이 없습니다"와 비슷한 문구가 포함된 오류가 표시됩니다. 여기서 다음과 같습니다.

- `arn:aws:iam::123456789012:user/MyUser`는 요청 사용자의 Amazon 리소스 이름(ARN)입니다.
- `action`은 사용자가 요청한 작업 이름입니다.
- `arn:aws:cloud9:us-east-2:123456789012:environment:12a34567b8cd9012345ef67abcd890e1`은 사용자가 작업 실행을 위해 요청한 환경의 ARN입니다.

원인: AWS Cloud9 콘솔에 로그인한 사용자에게 작업을 수행할 수 있는 올바른 AWS 액세스 권한이 없습니다.

해결 방법: 사용자에게 올바른 AWS 액세스 권한이 있는지 확인한 다음 작업을 다시 수행해 봅니다. 자세한 내용은 다음 자료를 참조하세요.

- 팀 설정의 3단계: 그룹에 AWS Cloud9 액세스 권한 추가
- 엔터프라이즈 설정의 6단계. 그룹 및 사용자가 조직 내에서 AWS Cloud9을 사용하도록 지정
- 공유 환경 사용의 환경 멤버 액세스 역할 정보

환경에 연결할 수 없음

문제: 사용자가 환경에 연결할 수 없고 연결 단계에서 멈춥니다.

원인: 파일의 권한을 변경하거나, 해당 `~/ .ssh/authorized_keys` 파일에서 AWS Cloud9 키를 제거하거나, 파일을 완전히 제거하면 이 문제가 발생할 수 있습니다.

해결 방법: 이 파일은 삭제하지 마세요. 삭제하는 경우 환경을 다시 생성해야 하며 기존 환경의 EBS 볼륨을 새 EC2 환경에 연결해야 할 수 있습니다. 이것은 손실된 데이터를 복구하기 위한 것입니다. 누락된 권한이 있는 경우 파일에 `Read-Write` 권한이 있는지 확인합니다. 이는 SSH 데몬이 읽을 수 있도록 하기 위한 것입니다.

환경을 열 수 없음

문제: 환경을 열려고 하는데 IDE가 5분 이상 표시되지 않습니다.

가능한 원인:

- AWS Cloud9 콘솔에 로그인한 IAM 사용자에게는 환경을 여는 데 필요한 AWS 액세스 권한이 없습니다.
- 환경이 AWS 클라우드 컴퓨팅 인스턴스 (예: Amazon EC2 인스턴스) 와 연결되어 있는 경우 다음과 같은 상황이 발생할 수 있습니다.
  - 인스턴스와 연결된 VPC가 올바른 설정으로 설정되어 있지 않습니다. AWS Cloud9
  - 인스턴스가 상태 간에 전환 중이거나 인스턴스에 연결하려고 할 때 AWS Cloud9 자동 상태 확인에 실패합니다.
- 환경이 SSH 환경인 경우 연결된 클라우드 컴퓨팅 인스턴스 또는 자체 서버가 액세스를 AWS Cloud9 허용하도록 올바르게 설정되지 않은 것입니다.

권장 솔루션:

- AWS Cloud9 콘솔에 로그인한 IAM 사용자에게 환경을 여는 데 필요한 AWS 액세스 권한이 있는지 확인하십시오. 그런 다음 환경을 다시 열어 보세요. 자세한 내용은 다음 항목을 읽어보거나 AWS 계정 관리자에게 문의하세요.
  - 팀 설정의 3단계: 그룹에 AWS Cloud9 액세스 권한 추가
  - 인증 및 액세스 제어의 AWS 관리형 정책은 다음과 같습니다. AWS Cloud9
  - 고급 팀 설정의 AWS Cloud9를 사용한 팀용 고객 관리형 정책 예
  - 인증 및 액세스 제어의 고객 관리형 정책 예
  - IAM 사용 설명서의 IAM 사용자의 권한 변경
  - IAM 사용 설명서의 IAM 정책 문제 해결

  로그인한 IAM 사용자가 여전히 환경을 열 수 없는 경우 로그아웃한 다음 계정의 AWS 계정 루트 사용자 또는 관리자 사용자로 다시 로그인해 보십시오. 그런 다음 환경을 다시 열어 보세요. 이러한 방식으로 환경을 열 수 없는 경우 IAM 사용자의 액세스 권한 관련 문제가 있었을 가능성이 가장 큽니다.
- 환경이 AWS 클라우드 컴퓨팅 인스턴스 (예: Amazon EC2 인스턴스) 와 연결된 경우 다음을 수행하십시오.

- 인스턴스와 연결된 VPC가 올바른 설정으로 설정되어 있는지 확인한 다음 환경을 다시 열어보십시오. AWS Cloud9자세한 정보는 다음에 대한 Amazon VPC 요구 사항 AWS Cloud9을 참조하세요.

  AWS 클라우드 컴퓨팅 인스턴스와 연결된 VPC가 올바른 설정으로 설정되어 있는데도 환경을 열 수 없는 경우, 인스턴스의 보안 그룹이 액세스를 AWS Cloud9 차단하고 있을 수 있습니다. AWS Cloud9문제 해결 기법으로, 보안 그룹에서 최소한 모든 IP 주소(Anywhere 또는 `0.0.0.0/0`)에 대해 인바운드 SSH 트래픽이 포트 22를 통해 허용되는지 확인하십시오. 지침은 Amazon EC2 사용 설명서의 보안 그룹 설명 및 보안 그룹 규칙 업데이트를 참조하십시오.

  추가 VPC 문제 해결 단계는 관련 동영상 (5분) 을 참조하십시오. AWS 지식 센터 동영상: VPC의 인스턴스에 연결할 수 없는 경우 어떻게 확인할 수 있습니까? on. YouTube

  > **Warning**
  >
  > 문제 해결을 완료했으면 인바운드 규칙을 적절한 주소 범위로 설정해야 합니다. 자세한 정보는 the section called “인바운드 SSH IP 주소 범위”을 참조하세요.

- 인스턴스를 다시 시작하고, 인스턴스가 실행 중이며 모든 시스템 검사를 통과했는지 확인한 후 환경을 다시 열어 보세요. 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 재부팅 및 상태 확인 보기를 참조하십시오.

- 환경이 SSH 환경인 경우 해당 환경과 연결된 클라우드 컴퓨팅 인스턴스 또는 자체 서버가 액세스를 AWS Cloud9 허용하도록 올바르게 설정되어 있는지 확인하십시오. 그런 다음 환경을 다시 열어 보세요. 자세한 정보는 SSH 환경 호스트 요구 사항을 참조하세요.

AWS Cloud9 환경을 열 수 없음: “이 환경은 현재 공동 작업자가 액세스할 수 없습니다. Please wait until the removal of managed temporary credentials is complete, or contact the owner of this environment'(현재 공동 작업자가 이 환경에 액세스할 수 없습니다. 관리형 임시 보안 인증 정보 제거가 완료될 때까지 기다리거나 이 환경의 소유자에게 문의하세요.)

문제: 환경 소유자가 아닌 사람이 새 공동 작업자를 환경에 추가하면 AWS 관리되는 임시 자격 증명이 비활성화됩니다. 보안 인증 정보는 `~/.aws/credentials` 파일을 삭제하면 사용 중지됩니다. `~/.aws/credentials`파일이 삭제되는 동안에는 새 공동 작업자가 환경에 액세스할 수 없습니다. AWS Cloud9

원인: AWS 관리형 임시 보안 인증 정보를 삭제하는 동안 환경에 액세스하지 못하도록 하는 것은 보안을 위한 조치입니다. 이를 통해 환경 소유자는 신뢰할 수 있는 공동 작업자만 관리형 보안 인증 정보에 액세스하도록 할 수 있습니다. 공동 작업자 목록이 유효하다고 생각되면 환경 소유자는 관리형 자격 증명을 다시 사용하도록 설정하여 공유할 수 있습니다. 자세한 정보는 AWS 관리형 임시 자격 증명에 대한 액세스 제어을 참조하세요.

권장 해결 방법: `~/.aws/credentials` 파일이 완전히 삭제될 때까지 기다린 후 AWS Cloud9 환경을 다시 열어보세요. 자격 증명 만료의 최대 대기 시간은 15분입니다. 또는 환경 소유자에게 관리형 임시 자격 증명을 다시 사용하도록 설정하거나 사용 중지하도록 요청합니다. 자격 증명을 다시 사용하도록 설정하거나 사용 중지하고 나면 공동 작업자가 즉시 환경에 액세스할 수 있습니다. 환경 소유자는 관리형 보안 인증 정보의 상태를 ENABLED 또는 DISABLED로 전환하여 자격 증명이 중간 상태로 남아 있지 않도록 합니다. 중간 통계 때문에 공동 작업자가 환경에 액세스하지 못할 수 있습니다.

> **Note**
>
> 환경 소유자와 공동 작업자가 동일한 AWS 계정에 속하는 경우, 공동 작업자는 콘솔의 Your environments(환경) 페이지에서 환경의 카드를 검토하여 환경 소유자를 식별하고 연락할 수 있습니다. 환경 소유자는 [환경 세부 정보(Environment details)] 페이지에도 나열되어 있습니다.

환경 삭제 오류: '하나 이상의 환경을 삭제하지 못함(One or more environments failed to delete)'

문제: AWS Cloud9 콘솔에서 하나 이상의 환경을 삭제하려고 하면 "하나 이상의 환경을 삭제하지 못했습니다."라는 메시지가 표시되고 환경 중 하나 이상이 삭제되지 않습니다.

가능한 원인: 하나 이상의 환경을 삭제하는 중에 문제가 AWS CloudFormation 있을 수 있습니다. AWS Cloud9 환경을 만들고 삭제하는 AWS CloudFormation 데 사용됩니다.

권장 해결 방법: 삭제되지 않은 환경을 각각 삭제하는 AWS CloudFormation 데 사용해 보십시오.

1. https://console.aws.amazon.com/cloudformation 에서 AWS CloudFormation 콘솔을 엽니다.
2. AWS 탐색 표시줄에서 해당 AWS 리전 환경을 선택합니다.
3. 스택 목록에서 스택 이름에 삭제되지 않은 환경 이름이 포함되고 상태가 DELETE_FAILED인 항목을 선택합니다. AWS CloudFormation 예를 들어, 환경 이름이 인 경우 **`my-demo-environment`** aws-cloud9-라는 이름으로 시작하는 스택을 선택하십시오. my-demo-environment (환경 이름 자체가 아니라 환경 이름 옆에 있는 상자 또는 옵션을 선택합니다.)
4. 작업, 스택 삭제을 선택합니다.

5. 메시지가 나타나면 예, 삭제를 선택합니다.

스택 삭제 프로세스는 몇 분 정도 걸릴 수 있습니다.

목록에서 스택이 사라지면 이제 환경이 삭제됩니다.

몇 분 후 스택에 DELETE_FAILED 상태가 표시되면 해당 환경이 아직 삭제되지 않은 것입니다. 실패한 스택의 각 리소스를 수동으로 삭제해 볼 수 있습니다.

> **Note**
>
> 장애가 발생한 스택의 리소스를 수동으로 삭제해도 스택 자체가 제거되지는 않습니다. AWS 계정

이러한 리소스를 수동으로 삭제하려면 다음과 같이 합니다. AWS CloudFormation 콘솔에서 장애가 발생한 스택을 선택한 다음 리소스 섹션을 선택합니다. 이 목록에 있는 각 리소스의 AWS 콘솔로 이동한 다음 해당 콘솔을 사용하여 리소스를 삭제하십시오.

IDE 환경의 타임아웃 시간 변경 AWS Cloud9

문제: 사용자가 Amazon EC2 환경의 제한 시간을 업데이트하려고 합니다.

원인: 기본 제한 시간은 30분입니다. 일부 사용자에게는 이 시간이 너무 짧을 수 있습니다.

권장 솔루션

1. 구성하려는 환경을 엽니다.
2. AWS Cloud9 IDE의 메뉴 모음에서 AWS Cloud9 기본 설정을 선택합니다.
3. 기본 설정 창에서 Amazon EC2 인스턴스 섹션으로 스크롤합니다.
4. 사용 가능한 목록에서 제한 시간 값을 선택하고 업데이트합니다.

AWS Cloud9 환경에 디스크 공간이 충분하지 않아 AWS 툴킷에서 SAM 응용 프로그램을 로컬로 실행하는 중 오류가 발생했습니다.

문제: AWS 툴킷을 사용하여 SAM 템플릿으로 정의된 애플리케이션에 대해 AWS SAM CLI 명령을 실행할 때 오류가 발생합니다.

가능한 원인: AWS 툴킷을 사용하여 로컬에서 서버리스 애플리케이션을 실행하고 디버그하면 이미지가 사용됩니다. AWS SAM Docker 이러한 이미지는 런타임 환경을 제공하고 배포하려는 Lambda 환경을 에뮬레이션하는 도구를 빌드합니다.

하지만 환경의 디스크 공간이 부족하면 이러한 기능을 제공하는 Docker 이미지를 빌드할 수 없으며 로컬 SAM 애플리케이션이 실행되지 않습니다. 이 경우 Output(출력) 탭에 다음과 유사한 오류가 나타날 수 있습니다.

```
Error: Could not find amazon/aws-sam-cli-emulation-image-python3.7:rapid-1.18.1 image
 locally and failed to pull it from docker.
```

이 오류는 Python 런타임을 사용하여 빌드된 SAM 애플리케이션과 관련이 있습니다. 애플리케이션용으로 선택한 런타임에 따라 약간 다른 메시지가 나타날 수 있습니다.

권장 솔루션: Docker 이미지를 빌드할 수 있도록 환경에서 디스크 공간을 확보합니다. IDE 터미널에서 다음 명령을 실행하여 사용하지 않는 Docker 이미지를 제거합니다.

```
docker image prune -a
```

디스크 공간의 제약으로 인해 SAM CLI 명령에 반복적으로 문제가 발생하는 경우 다른 인스턴스 유형을 사용하는 개발 환경으로 전환해 보세요.

(맨 위로 이동)

이전 버전의 Microsoft Edge 브라우저를 사용하여 IDE를 로드할 수 없음

문제: 웹 `HTTP403: FORBIDDEN` 브라우저를 사용하여 AWS Cloud9 IDE를 로드하려고 하면 오류가 반환됩니다. Microsoft Edge

가능한 원인: AWS Cloud9 IDE가 일부 이전 버전을 지원하지 않습니다Microsoft Edge.

권장 솔루션: 브라우저를 업데이트하려면 Microsoft Edge 도구 모음에서 줄임표(...) 버튼을 선택합니다. 메뉴에서 Settings(설정)를 선택한 다음 About Microsoft Edge(Microsoft Edge 정보)를 선택합니다. 업데이트가 필요한 경우 자동으로 다운로드되어 설치됩니다.

(맨 위로 이동)

AWS Cloud9 IDE 파일 탐색기에서 하위 폴더 구조인 /home/ec2-user/environment/home/ec2-user/environment를 만들 수 없습니다.

문제: AWS Cloud9 IDE 파일 탐색기에서 하위 폴더 구조 /home/ec2-user/environment/home/ec2-user/environment 를 만들 때 이 디렉토리를 열 수 없다는 오류 메시지가 나타납니다.

가능한 원인: 현재 IDE의 파일 시스템을 사용하여 같은 이름의 폴더 내에 하위 폴더 구조 /home/ec2-user/environment 를 만들 수 없습니다. AWS Cloud9 AWS Cloud9 IDE 파일 탐색기에서는 이 디렉터리 내의 파일에 액세스할 수 없지만 명령줄을 사용하여 액세스할 수 있습니다. 이 문제는 파일 경로 /home/ec2-user/environment/home/ec2-user/environment에만 영향을 미치며 /test/home/ec2-user/environment 및 /home/ec2-user/environment/test와 같은 파일 경로가 적합합니다. 이는 알려진 문제이며 AWS Cloud9 IDE 파일 탐색기에만 영향을 미칩니다.

권장 해결 방법: 다른 파일 이름 및 구조를 사용하세요.

(맨 위로 이동)

IDE의 파일 탐색기에서 하위 폴더 구조 /projects/projects를 만들 수 없습니다. AWS Cloud9 CodeCatalyst

문제: AWS Cloud9 IDE 파일 탐색기에서 하위 폴더 구조 /projects/projects를 만들 때 이 디렉토리를 열 수 없다는 오류 메시지가 나타납니다. CodeCatalyst

가능한 원인: 현재 IDE의 파일 탐색기를 사용하여 같은 이름의 폴더 내에 하위 폴더 구조 /projects를 만들 수 없습니다. AWS Cloud9 CodeCatalyst AWS Cloud9 IDE 파일 탐색기에서는 이 디렉터리 내의 파일에 액세스할 수 없지만 명령줄을 사용하여 액세스할 수 있습니다. 이 문제는 /projects/projects 파일 경로에만 영향을 미치며, /test/projects와 /projects/test/projects 같은 파일 경로가 적합합니다. 이는 알려진 문제이며 해당 AWS Cloud9 IDE 파일 탐색기에만 영향을 줍니다 CodeCatalyst.

권장 해결 방법: 다른 파일 이름 및 구조를 사용하세요.

(맨 위로 이동)

tmux 세션 오류 때문에 AWS Cloud9 에서 터미널 창과 상호 작용할 수 없습니다.

문제: 에서 AWS Cloud9새 터미널 창을 시작하려고 하면 예상한 명령줄 인터페이스를 사용할 수 없습니다. 명령 프롬프트가 없으며 텍스트를 입력할 수 없습니다. `tmux: need UTF-8 locale (LC_CTYPE)` 및 `invalid LC_ALL, LC_CTYPE or LANG` 같은 오류 메시지가 반환됩니다.

가능한 원인: tmux 오류로 인해 터미널이 응답하지 않을 수 있습니다. AWS Cloud9 tmux 유틸리티를 사용합니다. 이렇게 하면 페이지가 다시 로드되거나 개발 환경에 다시 연결할 때도 터미널에 표시되는 정보를 유지합니다.

tmux 세션에서 터미널 창에 표시되는 항목은 클라이언트에 의해 처리됩니다. 클라이언트는 여러 세션을 관리할 수 있는 서버와 통신합니다. 서버와 클라이언트는 tmp 폴더에 있는 소켓을 통해 통신합니다. 만약 tmp 폴더가 개발 환경에서 누락되었거나 지나치게 제한적인 권한이 적용되면 tmux 세션을 실행할 수 없습니다. 이 경우 IDE의 터미널 창이 응답하지 않습니다.

권장 해결 방법: tmux 오류로 인해 터미널 창과 상호 작용할 수 없는 경우 올바른 권한이 있는 tmp 폴더를 생성하는 다른 방법을 사용해야 합니다. 그렇게 하면 tmux 세션을 실행할 수 있습니다. 한 가지 해결책은 LC_CTYPE을 .bash_profile 또는 .bashrc 파일에 내보내는 것입니다. 또 다른 권장 해결 방법은 호스트 관리 AWS Systems Manager 구성을 설정하는 데 사용하는 것입니다. 이렇게 하면 Amazon EC2 콘솔을 통해 관련 인스턴스에 액세스할 수 있습니다.

호스트 관리 설정

1. 먼저 AWS Cloud9 콘솔에서 환경 인스턴스의 이름을 찾으십시오. Your environments(환경) 페이지에서 관련 패널을 선택하고 View details(세부 정보 보기)를 선택하면 됩니다. 환경 세부 정보 페이지에서 인스턴스로 이동(Go to Instance)을 선택합니다. Amazon EC2 콘솔에서 액세스해야 하는 인스턴스의 이름을 확인합니다.
2. 이제 AWS Systems Manager 콘솔로 이동하여 탐색 창에서 Quick Setup을 선택합니다.
3. 빠른 설정 페이지에서 생성(Create)을 선택합니다.
4. 구성 유형(Configuration types)에서 호스트 관리(Host Management)로 이동하고 생성(Create)을 선택합니다.
5. 호스트 관리 구성 옵션 사용자 지정(Customize Host Management configuration options)의 대상(Targets) 섹션에서 수동(Manual)을 선택합니다.
6. 액세스하려는 EC2 인스턴스를 선택한 다음 생성(Create)을 선택합니다.

인스턴스에 연결 및 명령 실행

> **Note**
>
> 다음 단계는 새로운 EC2 콘솔용입니다.

1. Amazon EC2 콘솔의 탐색 창에서 인스턴스(Instances)를 선택한 다음 연결하려는 인스턴스를 선택합니다.
2. 연결을 선택합니다.

   Connect(연결)이 활성화되지 않은 경우 먼저 인스턴스를 시작해야 할 수 있습니다.
3. Connect to your instance(인스턴스에 연결) 창에서 Connection method(연결 방법)로 Session Manager를 선택한 다음 Connect(연결)를 선택합니다.
4. 표시되는 터미널 세션 창에서 다음 명령을 입력합니다. 이 명령은 tmux 소켓을 사용할 수 있도록 올바른 권한을 가진 `tmp` 폴더를 만듭니다.

```
sudo mkdir /tmp
sudo chmod 777 /tmp
sudo rmdir /tmp/tmux-*
```

(맨 위로 이동)

Amazon EC2

다음 섹션에서는 Amazon EC2와 관련된 문제 해결을 간략하게 설명합니다.

Amazon EC2 인스턴스가 자동으로 업데이트되지 않음

문제: 최신 시스템 업데이트가 AWS Cloud9 개발 환경에 연결된 Amazon EC2 인스턴스에 자동으로 적용되지 않습니다.

원인: 최신 시스템 업데이트를 자동으로 적용하면 코드 또는 Amazon EC2 인스턴스가 사전 알림 또는 승인 없이 예기치 않은 방식으로 동작할 수 있습니다.

권장 솔루션:

Amazon EC2 사용 설명서의 인스턴스 소프트웨어 업데이트에 있는 지침에 따라 Amazon EC2 인스턴스에 정기적으로 시스템 업데이트를 적용하십시오.

인스턴스에서 명령을 실행하려면 인스턴스에 연결된 환경에서 AWS Cloud9 IDE의 터미널 세션을 사용하면 됩니다.

또는 ssh 또는 PuTTY 등과 같은 SSH 원격 액세스 유틸리티를 사용하여 인스턴스에 연결할 수도 있습니다. 이렇게 하려면 로컬 컴퓨터에서 ssh-keygen 또는 PuTTYgen 등과 같은 SSH 키 페어 생성 유틸리티를 사용합니다. 인스턴스에 연결된 환경의 AWS Cloud9 IDE를 사용하여 생성된 공개 키를 인스턴

스에 저장합니다. 그런 다음 생성한 프라이빗 키와 함께 SSH 원격 액세스 유틸리티를 사용하여 인스턴스에 액세스합니다. 자세한 내용은 해당 유틸리티 설명서를 참조하세요.

AWS CLI 또는 AWS-shell 오류: EC2 환경에서 “요청에 포함된 보안 토큰이 유효하지 않습니다.”

문제: EC2 환경용 AWS Cloud9 IDE에서 AWS Command Line Interface (AWS CLI) 또는 AWS-shell을 사용하여 명령을 실행하려고 하면 “요청에 포함된 보안 토큰이 유효하지 않습니다.” 라는 오류 메시지가 표시됩니다.

원인: AWS 관리형 임시 자격 증명을 사용하도록 설정하고 다음 중 하나가 발생할 경우에 보안 토큰이 잘못될 수 있습니다.

- AWS 관리형 임시 자격 증명으로 허용되지 않는 명령을 실행하려고 했습니다. 허용되는 명령 목록은 단원을 참조하세요AWS 관리형 임시 자격 증명이 지원하는 작업
- AWS 관리형 임시 자격 증명은 15분 후에 자동으로 만료됩니다.
- 환경 소유자가 아닌 다른 사람이 새 구성원을 추가했기 때문에 공유 환경의 AWS 관리형 임시 자격 증명이 비활성화되었습니다.

권장 솔루션:

- AWS 관리형 임시 자격 증명에서 허용하는 명령만 실행하십시오. AWS 관리형 임시 자격 증명으로 허용되지 않는 명령을 실행해야 하는 경우 영구 자격 증명 세트를 사용하여 환경에서 AWS CLI 또는 AWS-shell을 구성하십시오. 이렇게 하면 이러한 제한이 해소됩니다. 지침은 영구 액세스 자격 증명을 생성하여 환경에 저장을 참조하세요.
- 비활성화되거나 만료된 자격 증명의 경우 환경 소유자가 환경을 열어 해당 환경의 임시 자격 증명을 새로 고칠 AWS Cloud9 수 있도록 해야 합니다. 자세한 정보는 AWS 관리형 임시 자격 증명에 대한 액세스 제어을 참조하세요.

Docker에서 VPC의 IP 주소를 사용하므로 EC2 환경에 연결할 수 없음

문제: EC2 환경의 경우, IPv4 Classless Inter-Domain Routing(CIDR) 블록 `172.17.0.0/16`을 사용하는 Amazon VPC로 EC2 인스턴스를 시작하면 해당 환경을 열려고 할 때 연결이 멈출 수 있습니다.

원인: Docker는 동일한 브리지 네트워크에 연결된 컨테이너가 통신할 수 있도록 브리지 네트워크라는 링크 계층 장치를 사용합니다. AWS Cloud9 컨테이너 통신에 기본 브리지를 사용하는 컨테이너를 생성합니다. 기본 브리지는 일반적으로 `172.17.0.0/16` 서브넷을 컨테이너 네트워킹에 사용합니다.

환경 인스턴스의 VPC 서브넷이 Docker가 이미 사용하는 것과 동일한 주소 범위를 사용하는 경우, IP 주소 충돌이 발생할 수 있습니다. 따라서 AWS Cloud9 가 인스턴스에 연결하려고 시도하면 해당 연결은 게이트웨이 라우팅 테이블에 의해 Docker 브리지로 라우팅됩니다. 이렇게 하면 AWS Cloud9 개발 환경을 지원하는 EC2 인스턴스에 연결할 수 없습니다.

권장 해결 방법: 동일한 IPv4 CIDR 주소 블록을 사용하는 Amazon VPC와 Docker로 인한 IP 주소 충돌을 해결하려면 EC2 환경을 지원하는 인스턴스에 대해 새 VPC를 구성합니다. 이 새 VPC의 CIDR 블록을 `172.17.0.0/16`과 다른 블록으로 구성합니다. (기존 VPC 또는 서브넷의 IP 주소 범위는 변경할 수 없습니다.)

구성 정보는 Amazon VPC 사용 설명서에서 VPC 및 서브넷 크기 조정을 참조하세요.

AWS Cloud9 IDE 파일 탐색기에서 하위 폴더 구조인 /home/ec2-user/environment/home/ec2-user/environment를 만들 수 없습니다.

문제: AWS Cloud9 IDE 파일 탐색기에서 하위 폴더 구조 /home/ec2-user/environment/home/ec2-user/environment를 생성할 때 이 디렉토리를 열 수 없다는 오류 메시지가 나타납니다.

가능한 원인: 현재 IDE의 파일 시스템을 사용하여 같은 이름의 폴더 내에 하위 폴더 구조 /home/ec2-user/environment 를 만들 수 없습니다. AWS Cloud9 AWS Cloud9 IDE 파일 탐색기에서는 이 디렉터리 내의 파일에 액세스할 수 없지만 명령줄을 사용하여 액세스할 수 있습니다. 이 문제는 파일 경로 /home/ec2-user/environment/home/ec2-user/environment에만 영향을 미치며 /test/home/ec2-user/environment 및 /home/ec2-user/environment/test와 같은 파일 경로가 적합합니다. 이는 알려진 문제이며 AWS Cloud9 IDE 파일 탐색기에만 영향을 미칩니다.

권장 해결 방법: 다른 파일 이름 및 구조를 사용하세요.

AWS License Manager 라이선스 구성이 Amazon EC2 인스턴스와 연결된 경우 AWS Cloud9 콘솔에서 시작할 수 없음

문제: 콘솔에서 AWS Cloud9 EC2 환경을 시작하려고 하면 오류 메시지가 `unable to access your environment` 반환됩니다.

가능한 원인: 전체 소프트웨어 공급업체 라이선스의 관리를 AWS License Manager 간소화합니다. AWS 클라우드 License Manager를 설정할 때 기업 계약 조건을 기반으로 하는 라이선스 규칙 집합인 라이선스 구성을 생성합니다. 이러한 라이선스 구성은 Amazon 머신 이미지 (AMI) 또는 같은 메커니즘에 연결할 수 AWS CloudFormation있습니다. 이러한 메커니즘 중 하나를 사용하여 EC2 인스턴스를 시작할 수 있습니다.

AWSCloud9ServiceRolePolicyfor AWSServiceRoleForAWSCloud 9 서비스 연결 역할 (SLR) 의 이전 버전에는 현재 리소스 조건이 포함되어 있지 않습니다. `license-configuration` 이로 인해 인스턴스를 시작하고 중지할 수 AWS Cloud9 없습니다. 따라서 Amazon EC2 인스턴스에 대한 AWS Cloud9 액세스가 거부되고 오류가 반환됩니다.

권장 해결 방법: 기존 AWS Cloud9 환경에 액세스하여 License Manager를 사용할 수 없는 경우, 이전 AWSCloud9ServiceRolePolicy서비스 연결 역할을 인스턴스에 적용할 때 EC2 작업을 명시적으로 허용하는 SLR 버전으로 교체하십시오. `license-configuration` 이전 역할을 삭제하여 간단히 바꿀 수 있습니다. 그러면 업데이트된 역할이 자동으로 생성됩니다.

EC2 환경에서 일부 명령 또는 스크립트를 실행할 수 없음

문제: AWS Cloud9 EC2 개발 환경을 연 후에는 일부 유형의 패키지를 `yum` 설치하거나 또는 `apt` 같은 명령을 실행할 수 없으며 일반적으로 다른 Linux 운영 체제에서 작동하는 명령이 포함된 스크립트를 실행할 수 없습니다.

원인: EC2 환경에 AWS Cloud9 사용하는 Amazon EC2 인스턴스는 아마존 리눅스 (레드햇 엔터프라이즈 리눅스 (RHEL) 기반) 또는 우분투 서버를 기반으로 합니다.

솔루션: 패키지를 설치 또는 관리하거나 EC2 환경에 대한 IDE에서 명령 또는 스크립트를 실행하는 경우 해당 환경의 인스턴스에 따라 RHEL(Amazon Linux의 경우) 또는 Ubuntu Server와 호환되는지 확인하세요.

를 사용하여 EC2 환경을 생성할 때 “계정에 인스턴스 AWSCloud9SSMInstanceProfile 프로필이 없습니다”라는 오류 메시지가 보고됩니다. AWS CloudFormation

문제: AWS::Cloud9::Environment EC2 AWS CloudFormation 리소스를 사용하여 EC2 환경을 생성할 때 사용자에게 인스턴스 프로필이 계정에 존재하지 AWSCloud9SSMInstanceProfile 않는다는 오류 메시지가 표시됩니다.

원인: 수신하지 않는 EC2 환경을 생성할 때 서비스 역할 `AWSCloud9SSMAccessRole`과 인스턴스 프로파일 `AWSCloud9SSMInstanceProfile`을 생성해야 합니다. 이러한 IAM 리소스를 통해 Systems Manager는 개발 환경을 백업하는 EC2 인스턴스를 관리할 수 있습니다.

콘솔을 사용하여 수신하지 않는 환경을 만드는 경우 `AWSCloud9SSMAccessRole` 및 `AWSCloud9SSMInstanceProfile`이 자동으로 생성됩니다. 하지만 AWS CloudFormation 또는 AWS CLI 를 사용하여 처음으로 무수신 환경을 만들 때는 이러한 IAM 리소스를 수동으로 생성해야 합니다.

권장 솔루션: AWS CloudFormation 템플릿 편집 및 IAM 권한 업데이트에 대한 자세한 내용은 을 참조하십시오. AWS CloudFormation을 사용하여 수신하지 않는 EC2 환경 생성

AWS CloudFormation을 사용하여 EC2 환경을 만들 때 '리소스에서 **perform: ssm:StartSession**에 대한 권한 없음'이라는 오류 메시지가 표시됨

문제: AWS::Cloud9::Environment EC2 AWS CloudFormation 리소스를 사용하여 EC2 환경을 만들 때 사용자는 "리소스에서 작업을 수행할 권한이 없음"이라는 알림을 `AccessDeniedException` 받고 알림을 받습니다. `ssm:StartSession`

원인: Systems Manager가 수신하지 않는 인스턴스에 사용하는 EC2 환경에 대한 구성의 일부로서 필요한 `StartSession` API를 호출할 권한이 사용자에게 없습니다.

권장 솔루션: AWS CloudFormation 템플릿 편집 및 IAM 권한 업데이트에 대한 자세한 내용은 을 참조하십시오. AWS CloudFormation을 사용하여 수신하지 않는 EC2 환경 생성

AWS CLI를 사용하여 EC2 환경을 생성할 때 "리소스에 **iam:GetInstanceProfile**: 인스턴스 프로파일 **AWSCloud9SSMInstanceProfile**를 수행할" 권한이 없음을 보고하는 오류 메시지

문제: 를 사용하여 EC2 환경을 만들 때 사용자는 해당 환경이 "리소스: 인스턴스 GetInstanceProfile 프로필에서 iam을 수행할 수 있는" 권한이 없다는 알림을 받고 알림을 받습니다. AWS CLI`AccessDeniedException` AWS Cloud9 `AWSCloud9SSMInstanceProfile`

원인: AWS Cloud9 무수신 인스턴스에 대해 Systems Manager를 사용하는 EC2 환경 구성의 일부로 필요한 `StartSession` API를 호출할 권한이 없습니다.

권장 솔루션: 환경에 필요한 `AWSCloud9SSMAccessRole` 서비스 역할을 추가하는 방법에 대한 자세한 내용은 `AWSCloud9SSMInstanceProfile` 을 참조하십시오. AWS Cloud9 AWS CLI를 사용하여 Systems Manager의 인스턴스 프로파일 관리

Amazon EBS 볼륨에 기본 암호화가 적용될 때 환경을 생성하지 못함

문제: Amazon EC2 환경을 생성하려고 할 때 `Failed to create environments. The development environment '[environment-ID]' failed to create` 오류가 반환됩니다.

가능한 원인: AWS Cloud9 IDE에서 기본적으로 암호화된 Amazon EBS 볼륨을 사용하는 경우 AWS Identity and Access Management 서비스 연결 역할에 대한 EBS 볼륨에 대한 액세스 권한이 AWS Cloud9 필요합니다. AWS KMS keys 액세스가 제공되지 않으면 AWS Cloud9 IDE가 시작되지 않을 수 있으며 문제를 디버깅하기 어려울 수 있습니다.

권장 솔루션: 액세스를 제공하려면 Amazon EBS 볼륨에서 사용하는 고객 관리 키에 AWS Cloud9`AWSServiceRoleForAWSCloud9`, 의 서비스 연결 역할을 추가하십시오.

이 작업에 대한 자세한 내용은 AWS 규범적 지침 패턴의 AWS Cloud9 기본 암호화가 적용된 Amazon EBS 볼륨을 사용하는 Amazon EBS 볼륨 생성을 참조하십시오.

EC2-Classic 계정에 대한 VPC 오류: "Unable to access your environment(사용자 환경에 액세스할 수 없음)"

문제: EC2-Classic이 Amazon EC2의 원래 릴리스에 도입되었습니다. 2013년 12월 4일 이전에 설정된 AWS 계정 것을 사용하는 경우, AWS Cloud9 EC2 개발 환경을 생성할 때 Amazon VPC와 서브넷을 구성하지 않으면 이 오류가 발생할 수 있습니다.

기본 VPC 설정을 수락하면 EC2-Classic 네트워크에서 Amazon EC2 인스턴스가 시작됩니다. 인스턴스는 기본 VPC의 서브넷에서 시작되지 않습니다. 환경 생성이 실패하면 다음과 같은 메시지가 표시됩니다.

Environment Error

Unable to access your environment

The environment creation failed with the error: The following resource(s) failed to create: [Instance]. . Rollback requested by user..

EC2 인스턴스가 기본 VPC에 있지 않아 오류가 발생했음을 확인할 수 있습니다. 개발 환경의 스택 이벤트 기록을 보는 AWS CloudFormation 데 사용합니다.

1. AWS CloudFormation 콘솔을 엽니다. 자세한 내용은 AWS CloudFormation 콘솔에 로그인을 참조하세요.
2. AWS CloudFormation 콘솔에서 스택을 선택합니다.
3. Stacks(스택) 페이지에서 생성하지 못한 개발 환경의 이름을 선택합니다.
4. Stack details(스택 세부 정보) 페이지에서 Events(이벤트) 탭을 선택하고 다음 항목을 확인합니다.

   Status: CREATE_FAILED

상태 이유: AssociatePublicIpAddress 매개변수는 VPC 시작 시에만 지원됩니다. [...]

원인: AWS Cloud9 개발 환경은 특정 VPC 요구 사항을 충족하는 Amazon VPC와 연결되어야 합니다. EC2-Classic이 활성화된 계정의 경우, EC2 환경 생성 시 기본 네트워크 설정을 수락하면 VPC에서 필수 EC2 인스턴스가 시작되지 않습니다. 대신, EC2-Classic 네트워크에서 인스턴스가 시작됩니다.

권장 솔루션: EC2-Classic 계정의 경우, EC2 환경 생성 시 VPC 및 서브넷을 선택해야 합니다. Configure settings(설정 구성) 페이지의 Network settings (advanced)(네트워크 설정(고급)) 섹션에서 EC2 인스턴스를 시작할 수 있는 VPC 및 서브넷을 선택합니다.

기타 서비스 AWS

다음 섹션에서는 기타 AWS 서비스와 관련된 문제 해결 문제를 간략하게 설명합니다.

IDE의 파일 탐색기 내에서 하위 폴더 구조 /projects/projects를 만들 수 없습니다. AWS Cloud9 CodeCatalyst

문제: AWS Cloud9 IDE 파일 탐색기에서 하위 폴더 구조 /projects/projects를 만들 때 이 디렉토리를 열 수 없다는 오류 메시지가 나타납니다. CodeCatalyst

가능한 원인: 현재 IDE의 파일 탐색기를 사용하여 같은 이름의 폴더 내에 하위 폴더 구조 /projects를 만들 수 없습니다. AWS Cloud9 CodeCatalyst AWS Cloud9 IDE 파일 탐색기에서는 이 디렉터리 내의 파일에 액세스할 수 없지만 명령줄을 사용하여 액세스할 수 있습니다. 이 문제는 /projects/projects 파일 경로에만 영향을 미치며, /test/projects와 /projects/test/projects 같은 파일 경로가 적합합니다. 이는 알려진 문제이며 해당 AWS Cloud9 IDE 파일 탐색기에만 영향을 줍니다 CodeCatalyst.

권장 해결 방법: 다른 파일 이름 및 구조를 사용하세요.

IDE 외부에서 실행 중인 애플리케이션을 표시할 수 없음

문제: IDE 외부에서 웹 브라우저 탭에서 실행 중인 애플리케이션을 보려고 하면 웹 브라우저 탭에 오류가 표시되거나 탭이 비어 있습니다.

가능한 원인:

- IDE에서 애플리케이션이 실행되고 있지 않습니다.
- 애플리케이션이 IP `127.0.0.1` 또는 `localhost`를 사용해 실행 중입니다.

- 애플리케이션이 AWS Cloud9 EC2 개발 환경에서 실행 중입니다. 해당하는 Amazon EC2 인스턴스와 연결된 하나 이상의 보안 그룹이 애플리케이션에 필요한 프로토콜, 포트 또는 IP 주소를 통한 인바운드 트래픽을 허용하지 않습니다.
- 애플리케이션이 AWS 클라우드 컴퓨팅 인스턴스 (예: Amazon EC2 인스턴스) 를 위한 AWS Cloud9 SSH 개발 환경에서 실행되고 있습니다. 또한 해당 인스턴스와 연결된 Virtual Private Cloud(VPC)의 서브넷에 대한 네트워크 ACL이 애플리케이션에 필요한 프로토콜 포트 또는 IP 주소를 통한 인바운드 트래픽을 허용하지 않습니다.
- URL이 올바르지 않습니다.
- 인스턴스의 퍼블릭 IP 주소 대신 애플리케이션 미리 보기 탭의 URL이 요청됩니다.
- `127.0.0.1` 또는 `localhost` IP가 포함된 주소로 이동하려고 했습니다. 이러한 IP는 환경의 리소스가 아니라 로컬 컴퓨터의 리소스에 액세스하려고 합니다.
- 인스턴스의 퍼블릭 IP 주소가 변경되었습니다.
- 웹 요청이 애플리케이션에 필요한 프로토콜, 포트 또는 IP 주소를 통한 트래픽을 차단하는 가상 프라이빗 네트워크(VPN)에서 시작되었습니다.
- 애플리케이션이 SSH 환경에서 실행 중입니다. 하지만 서버 또는 연결된 네트워크가 애플리케이션에 필요한 프로토콜, 포트 또는 IP 주소를 통한 트래픽을 허용하지 않습니다.

권장 솔루션:

- 애플리케이션이 IDE에서 실행 중인지 확인하세요.
- 애플리케이션이 IP `127.0.0.1` 또는 `localhost`를 사용해 실행 중이지 않은지 확인하세요. Node.js 및 Python으로 작성된 예제는 단원을 참조하세요애플리케이션 실행
- 애플리케이션이 AWS 클라우드 컴퓨팅 인스턴스 (예: Amazon EC2 인스턴스) 에서 실행되고 있다고 가정해 보겠습니다. 해당하는 인스턴스와 연결된 모든 보안 그룹이 애플리케이션에 필요한 프로토콜, 포트 또는 IP 주소를 통한 인바운드 트래픽을 허용하지 않는지 확인하세요. 지침은 인터넷을 통해 실행 중인 애플리케이션 공유에서 2단계: 인스턴스의 보안 그룹 설정 섹션을 참조하세요. 또한 Amazon VPC 사용 설명서에서 VPC의 보안 그룹도 참조하세요.
- 애플리케이션이 AWS 클라우드 컴퓨팅 인스턴스에서 실행되고 있다고 가정해 보겠습니다. 해당 인스턴스와 연결된 VPC의 서브넷에 대한 네트워크 ACL이 있는 경우 네트워크 ACL이 애플리케이션에 필요한 프로토콜, 포트 또는 IP 주소를 통한 인바운드 트래픽을 허용하는지 확인하세요. 지침은 인터넷을 통해 실행 중인 애플리케이션 공유에서 3단계: 인스턴스의 서브넷 설정 섹션을 참조하세요. Amazon VPC 사용 설명서에서 네트워크 ACL도 참조하세요.

- 프로토콜(및 포트, 지정되어 있어야 함)을 포함한 요청 URL이 올바른지 확인하십시오. 자세한 내용은 인터넷을 통해 실행 중인 애플리케이션 공유에서 4단계: 실행 중인 애플리케이션 URL 공유 섹션을 참조하세요.
- 다음과 같은 형식의 URL을 요청하지 않는 것이 좋습니다 `https://12a34567b8cd9012345ef67abcd890e1.vfs.cloud9.us-east-2.amazonaws.com/` (여기서 `12a34567b8cd9012345ef67abcd890e1` 는 환경에 AWS Cloud9 할당되는 ID이고 다른 `us-east-2` 하나는 환경에 대한 AWS 지역 ID). 이 URL은 환경의 IDE가 열려 있고 애플리케이션이 동일한 웹 브라우저에서 실행 중인 경우에만 작동합니다.
- `127.0.0.1` 또는 `localhost` IP가 포함된 주소로 이동하려고 했다면 대신 실행 중인 애플리케이션의 로컬 주소가 아닌 올바른 주소로 이동해 보세요. 자세한 정보는 인터넷을 통해 실행 중인 애플리케이션 공유을 참조하세요.
- 애플리케이션이 AWS 클라우드 컴퓨팅 인스턴스에서 실행되고 있다고 가정해 보겠습니다. 인스턴스의 퍼블릭 IP 주소가 변경되었는지 확인하세요. 인스턴스의 퍼블릭 IP 주소는 인스턴스가 다시 시작되면 언제든 변경될 수 있습니다. 이 IP 주소가 변경되지 않도록 하려면 탄력적 IP 주소를 할당한 후 해당 주소를 실행 중인 인스턴스에 할당합니다. 자세한 내용은 인터넷을 통해 실행 중인 애플리케이션 공유에서 4단계: 실행 중인 애플리케이션 URL 공유 섹션을 참조하세요.
- 웹 요청이 VPN에서 시작된 경우 VPN이 애플리케이션에 필요한 프로토콜, 포트 또는 IP 주소를 통한 트래픽을 허용하는지 확인하십시오. VPN을 변경할 수 없는 경우 네트워크 관리자에게 문의하세요. 또는 가능한 경우 다른 네트워크에서 웹 요청을 수행하세요.
- 애플리케이션이 자체 서버의 SSH 환경에서 실행 중인 경우 서버 및 연결된 네트워크가 애플리케이션에 필요한 프로토콜, 포트 및 IP 주소를 통한 트래픽을 허용하는지 확인하세요. 서버 또는 연결된 네트워크를 변경할 수 없는 경우 서버 또는 네트워크 관리자에게 문의하세요.
- URL 앞에서 `curl` 명령을 실행하여 환경의 터미널에서 애플리케이션을 실행해 보세요. 이 명령어에 오류 메시지가 표시되는 경우 관련 없는 다른 문제가 있을 수 있습니다. AWS Cloud9

AWS 툴킷 실행 중 오류: “환경에 inode가 부족합니다. 'fs.inotify.max_user_watchs' 제한을 늘리십시오.”

문제: AWS 툴킷에서 사용하는 파일 감시 유틸리티가 감시할 수 있는 파일의 현재 한도 또는 할당량에 근접하고 있습니다.

원인: AWS 툴킷은 파일 및 디렉토리의 변경 사항을 모니터링하는 파일 감시 유틸리티를 사용합니다. 유틸리티가 감시할 수 있는 파일 수의 현재 할당량에 가까울 때 경고 메시지가 나타납니다.

권장 솔루션: 파일 감시자가 처리할 수 있는 최대 파일 수를 늘리려면 다음을 수행합니다.

1. 메뉴 모음에서 [창(Window)], [새 터미널(New Terminal)]을 선택하여 터미널 세션을 시작합니다.
2. 다음 명령을 입력합니다.

```
sudo bash -c 'echo "fs.inotify.max_user_watches=524288" >> /etc/sysctl.conf' &&
 sudo sysctl -p
```

Lambda 로컬 함수 실행 오류: SAM Local을 설치할 수 없음

문제: AWS Cloud9 IDE에서 AWS Lambda 함수의 로컬 버전을 실행하려고 하면 대화 상자가 표시됩니다. 대화 상자에 SAM Local을 설치하는 데 문제가 있다고 표시됩니다. AWS Cloud9 AWS Cloud9 IDE에서 로컬 버전의 AWS Lambda 함수를 실행하려면 SAM Local이 필요합니다. SAM Local이 설치될 때까지 IDE에서 Lambda 함수의 로컬 버전을 실행할 수 없습니다.

원인: 환경의 예상 경로에서 SAM Local을 찾을 AWS Cloud9 수 없습니다`~/.c9/bin/sam`. 이는 SAM Local이 아직 설치되지 않았거나, 설치된 경우에는 AWS Cloud9 에서 설치 위치를 찾을 수 없기 때문입니다.

권장 해결 방법: SAM Local 설치가 완료될 때까지 AWS Cloud9 기다리거나 직접 설치할 수 있습니다.

SAM Local 설치 시도가 어떻게 AWS Cloud9 진행되는지 보려면 메뉴 표시줄에서 [창], [설치 프로그램]을 선택하십시오.

SAM Local을 직접 설치하려면 AWS Serverless Application Model 개발자 안내서의 Linux에 AWS SAM CLI 설치에 나와 있는 지침을 따르십시오.

AWS Control Tower 다음을 사용하여 AWS Cloud9 Amazon EC2 환경을 생성하려고 할 때 오류가 발생했습니다. “오류가 발생하여 환경 생성에 실패했습니다. [: :GuardControlTower: :Hook].”

문제: 사전 예방적 제어 AWS Cloud9 AWS Control Tower CT.EC2.PR.8에 호환성 문제가 있습니다. 이 컨트롤이 활성화되면 AWS Cloud9에서 EC2 환경을 만들 수 없습니다.

원인: 매개 변수가 AWS Control Tower 템플릿에 있을 것으로 예상하고 있습니다. AssociatePublicIpAddress AWS CloudFormation 지금은 이 파라미터를 추가할 수 없습니다.

권장 해결 방법: 콘솔에서 컨트롤 CT.EC2.PR.8을 비활성화하고 에서 환경을 다시 생성하십시오 AWS Control Tower . AWS Cloud9

Amazon EBS 볼륨에 기본 암호화가 적용될 때 환경을 생성하지 못함

문제: Amazon EC2 환경을 생성하려고 할 때 `Failed to create environments. The development environment '[environment-ID]' failed to create` 오류가 반환됩니다.

가능한 원인: AWS Cloud9 IDE에서 기본적으로 암호화된 Amazon EBS 볼륨을 사용하는 경우 AWS Identity and Access Management 서비스 연결 역할에 대한 EBS 볼륨에 대한 액세스 권한이 AWS Cloud9 필요합니다. AWS KMS keys 액세스가 제공되지 않으면 AWS Cloud9 IDE가 시작되지 않을 수 있으며 문제를 디버깅하기 어려울 수 있습니다.

권장 솔루션: 액세스를 제공하려면 Amazon EBS 볼륨에서 사용하는 고객 관리 키에 AWS Cloud9`AWSServiceRoleForAWSCloud9`, 의 서비스 연결 역할을 추가하십시오.

이 작업에 대한 자세한 내용은 AWS 규범적 지침 패턴의 AWS Cloud9 기본 암호화가 적용된 Amazon EBS 볼륨을 사용하는 Amazon EBS 볼륨 생성을 참조하십시오.

(맨 위로 이동)

AWS License Manager 라이선스 구성이 Amazon EC2 인스턴스와 연결된 경우 AWS Cloud9 콘솔에서 시작할 수 없음

문제: 콘솔에서 AWS Cloud9 EC2 환경을 시작하려고 하면 오류 메시지가 `unable to access your environment` 반환됩니다.

가능한 원인: 전체 소프트웨어 공급업체 라이선스의 관리를 AWS License Manager 간소화합니다. AWS 클라우드 License Manager를 설정할 때 기업 계약 조건을 기반으로 하는 라이선스 규칙 집합인 라이선스 구성을 생성합니다. 이러한 라이선스 구성은 Amazon 머신 이미지 (AMI) 또는 같은 메커니즘에 연결할 수 AWS CloudFormation있습니다. 이러한 메커니즘 중 하나를 사용하여 EC2 인스턴스를 시작할 수 있습니다.

AWSCloud9ServiceRolePolicyfor AWSServiceRoleForAWSCloud 9 서비스 연결 역할 (SLR) 의 이전 버전에는 현재 리소스 조건이 포함되어 있지 않습니다. `license-configuration` 이로 인해 인스턴스를 시작하고 중지할 수 AWS Cloud9 없습니다. 따라서 Amazon EC2 인스턴스에 대한 AWS Cloud9 액세스가 거부되고 오류가 반환됩니다.

권장 해결 방법: 기존 AWS Cloud9 환경에 액세스하여 License Manager를 사용할 수 없는 경우, 이전 AWSCloud9ServiceRolePolicy서비스 연결 역할을 인스턴스에 적용할 때 EC2 작업을 명시적으로 허용하는 SLR 버전으로 교체하십시오. `license-configuration` 이전 역할을 삭제하여 간단히 바꿀 수 있습니다. 그러면 업데이트된 역할이 자동으로 생성됩니다.

(맨 위로 이동)

애플리케이션 미리 보기

다음 섹션에서는 애플리케이션 미리 보기와 관련된 문제 해결 문제를 간략하게 설명합니다.

환경을 다시 로드한 후 애플리케이션 미리 보기를 새로 고쳐야 함

문제: 애플리케이션 미리 보기 탭을 표시하는 환경을 다시 로드한 후 해당 탭에 애플리케이션 미리 보기가 표시되지 않습니다.

원인: 때때로 사용자가 무한 루프를 실행할 수 있는 코드를 작성합니다. 또는 코드에 너무 많은 메모리를 사용해서 애플리케이션 미리보기가 실행 중일 때 AWS Cloud9 IDE가 일시 중지되거나 중지될 수 있습니다. 이 문제가 발생하지 않도록 하려면 환경을 다시 로드할 때마다 응용 프로그램 미리 보기 탭을 다시 로드하지 AWS Cloud9 않습니다.

솔루션: 애플리케이션 미리 보기 탭을 표시하는 환경을 다시 로드한 후 애플리케이션 미리 보기를 표시하려면 탭에서 [페이지를 로드하려면 클릭(Click to load the page)] 버튼을 선택합니다.

애플리케이션 미리 보기 또는 파일 미리 보기 알림: '서드 파티 쿠키가 사용 중지됨(Third-party cookies disabled)'

문제: 응용 프로그램 또는 파일을 미리 보려고 하면 알림에 다음과 같은 메시지가 표시됩니다. "브라우저에 타사 쿠키가 비활성화되어 있으므로 미리 보기 기능이 비활성화됩니다."

원인: IDE를 여는 데 타사 쿠키가 필요하지 않습니다. AWS Cloud9 그러나 애플리케이션 미리 보기 또는 파일 미리 보기 기능을 사용하려면 서드 파티 쿠키를 활성화해야 합니다.

해결 방법: 웹 브라우저에서 타사 쿠키를 활성화하고, IDE를 다시 로드한 다음, 미리 보기를 다시 열어 보십시오.

- Apple Safari: Apple Support 웹 사이트에서 Safari에서 쿠키 및 웹 사이트 데이터 관리를 참조하세요.
- Google Chrome: Chrome 도움말 웹사이트에서 Chrome에서 쿠키 지우기, 사용 및 관리의 쿠키 설정 변경을 참조하세요.
- Internet Explorer: Microsoft 지원 웹 사이트에서 쿠키 삭제 및 관리의 쿠키 차단 또는 허용을 참조하세요.
- Microsoft Edge: Microsoft 지원 웹 사이트에서 서드 파티 쿠키 차단을 참조하세요.

- Mozilla Firefox: Mozilla 지원 웹 사이트에서 웹 사이트에서 사용자의 기본 설정을 추적하는 데 사용하는 쿠키 사용 및 사용 중지의 서드 파티 쿠키 허용 설정을 참조하세요.
- 기타 웹 브라우저: 해당 웹 브라우저의 설명서를 참조하세요.

웹 브라우저에서 이러한 세분성을 허용하는 경우 AWS Cloud9에 대해서만 서드 파티 쿠키를 활성화할 수 있습니다. 이렇게 하려면 AWS Cloud9을 사용하려는 지원 AWS 리전 에 따라 다음 도메인을 지정합니다.

AWS 리전	도메인

미국 동부(버지니아 북부)	`*.vfs.cloud9.us-east-1.amazonaws.com`
	
	`vfs.cloud9.us-east-1.amazonaws.com`

미국 동부(오하이오)	`*.vfs.cloud9.us-east-2.amazonaws.com`
	
	`vfs.cloud9.us-east-2.amazonaws.com`

미국 서부(캘리포니아 북부)	`*.vfs.cloud9.us-west-1.amazonaws.com`
	
	`vfs.cloud9.us-west-1.amazonaws.com`

미국 서부(오레곤)	`*.vfs.cloud9.us-west-2.amazonaws.com`
	
	`vfs.cloud9.us-west-2.amazonaws.com`

아프리카(케이프타운)	`*.vfs.cloud9.af-south-1.amazonaws.com`

AWS 리전	도메인
	vfs.cloud9.af-south-1.amazonaws.com
아시아 태평양(홍콩)	*.vfs.cloud9.ap-east-1.amazonaws.com vfs.cloud9.ap-east-1.amazonaws.com
아시아 태평양(뭄바이)	*.vfs.cloud9.ap-south-1.amazonaws.com vfs.cloud9.ap-south-1.amazonaws.com
아시아 태평양(오사카)	*.vfs.cloud9.ap-northeast-3.amazonaws.com vfs.cloud9.ap-northeast-3.amazonaws.com
아시아 태평양(서울)	*.vfs.cloud9.ap-northeast-2.amazonaws.com vfs.cloud9.ap-northeast-2.amazonaws.com
아시아 태평양(싱가포르)	*.vfs.cloud9.ap-southeast-1.amazonaws.com vfs.cloud9.ap-southeast-1.amazonaws.com
아시아 태평양(시드니)	*.vfs.cloud9.ap-southeast-2.amazonaws.com vfs.cloud9.ap-southeast-2.amazonaws.com

AWS 리전	도메인
아시아 태평양(도쿄)	*.vfs.cloud9.ap-northeast-1.amazonaws.com vfs.cloud9.ap-northeast-1.amazonaws.com
캐나다(중부)	*.vfs.cloud9.ca-central-1.amazonaws.com vfs.cloud9.ca-central-1.amazonaws.com
유럽(프랑크푸르트)	*.vfs.cloud9.eu-central-1.amazonaws.com vfs.cloud9.eu-central-1.amazonaws.com
유럽(아일랜드)	*.vfs.cloud9.eu-west-1.amazonaws.com vfs.cloud9.eu-west-1.amazonaws.com
유럽(런던)	*.vfs.cloud9.eu-west-2.amazonaws.com vfs.cloud9.eu-west-2.amazonaws.com
유럽(밀라노)	*.vfs.cloud9.eu-south-1.amazonaws.com vfs.cloud9.eu-south-1.amazonaws.com

AWS 리전	도메인
유럽(파리)	*.vfs.cloud9.eu-west-3.amazonaws.com vfs.cloud9.eu-west-3.amazonaws.com
유럽(스톡홀름)	*.vfs.cloud9.eu-north-1.amazonaws.com vfs.cloud9.eu-north-1.amazonaws.com
중동(바레인)	*.vfs.cloud9.me-south-1.amazonaws.com vfs.cloud9.me-south-1.amazonaws.com
남아메리카(상파울루)	*.vfs.cloud9.sa-east-1.amazonaws.com vfs.cloud9.sa-east-1.amazonaws.com

애플리케이션 미리 보기 탭에 오류가 표시되거나 이 탭이 비어 있음

문제: IDE의 메뉴 모음에서 [미리 보기, 실행 중인 애플리케이션 미리 보기(Preview, Preview Running Application)] 또는 [도구, 미리 보기, 실행 중인 애플리케이션 미리 보기(Tools, Preview, Preview Running Application)]를 선택하여 IDE의 미리 보기 탭에 애플리케이션을 표시하려고 하면 해당 탭에 오류가 표시되거나 탭이 비어 있습니다.

가능한 원인:

- IDE에서 애플리케이션이 실행되지 않습니다.
- 애플리케이션이 HTTP를 사용하여 실행 중이지 않습니다.
- 애플리케이션이 두 개 이상의 포트를 통해 실행 중입니다.

- 애플리케이션이 8080, 8081 또는 8082 이외의 포트를 통해 실행 중입니다.
- 애플리케이션이 127.0.0.1, localhost 또는 0.0.0.0 이외의 IP를 통해 실행 중입니다.
- 포트(8080, 8081 또는 8082)가 미리 보기 탭의 URL에 지정되어 있지 않습니다.
- 네트워크가 포트 8080, 8081 또는 8082에 대한 인바운드 트래픽을 차단합니다.
- 127.0.0.1, localhost 또는 0.0.0.0 IP가 포함된 주소로 이동하려고 했습니다. 기본적으로 AWS Cloud9 IDE는 로컬 컴퓨터로 이동하려고 시도합니다. 환경에 연결된 인스턴스나 자체 서버로 이동하려고 시도하지 않습니다.

권장 솔루션:

- 애플리케이션이 IDE에서 실행 중인지 확인하세요.
- 애플리케이션이 HTTP를 사용하여 실행 중인지 확인하십시오. Node.js 및 Python으로 작성된 예제는 단원을 참조하세요애플리케이션 실행
- 애플리케이션이 하나의 포트만 통해 실행 중인지 확인하십시오. Node.js 및 Python으로 작성된 예제는 단원을 참조하세요애플리케이션 실행
- 애플리케이션이 포트 8080, 8081 또는 8082를 통해 실행 중인지 확인하십시오. Node.js 및 Python으로 작성된 예제는 단원을 참조하세요애플리케이션 실행
- 애플리케이션이 IP 127.0.0.1, localhost 또는 0.0.0.0을 사용해 실행 중인지 확인하십시오. Node.js 및 Python으로 작성된 예제는 단원을 참조하세요애플리케이션 실행
- 미리 보기 탭의 URL에 :8080, :8081 또는 :8082를 추가합니다.
- 네트워크가 포트 8080, 8081 또는 8082를 통한 인바운드 트래픽을 허용하는지 확인하십시오. 네트워크를 변경할 수 없는 경우 네트워크 관리자에게 문의하세요.
- IP가 127.0.0.1, localhost 또는 0.0.0.0인 주소로 이동하려는 경우 대신 https://12a34567b8cd9012345ef67abcd890e1.vfs.cloud9.us-east-2.amazonaws.com/ 주소로 이동합니다. 이 주소에서 12a34567b8cd9012345ef67abcd890e1은 AWS Cloud9 이 환경에 할당하는 ID입니다. us-east-2는 환경이 위치한 AWS 리전 의 ID입니다. IDE 외부에서 이 주소로 이동해 볼 수도 있습니다. 하지만 이 주소는 환경의 IDE가 열려 있고 애플리케이션이 동일한 웹 브라우저에서 실행 중인 경우에만 작동합니다.
- 앞서 설명한 모든 조건이 충족되는지 확인한 후 애플리케이션을 중지한 다음 다시 시작해 보세요.
- 애플리케이션을 중지한 다음 다시 시작하면 메뉴 모음에서 다시 Preview, Preview Running Application(미리 보기, 실행 중인 애플리케이션 미리 보기) 또는 Tools, Preview, Preview Running Application(도구, 미리 보기, 실행 중인 애플리케이션 미리 보기)을 선택해 봅니다. 아니면 탭이 이미

표시되어 있는 경우에는 해당하는 애플리케이션 미리 보기 탭에서 Refresh(새로 고침) 버튼(원형 화살표)을 선택해 봅니다.

사이트에 대한 연결이 안전하지 않아 IDE에서 웹 콘텐츠를 미리 볼 수 없음

문제: AWS Cloud9 EC2 환경에서 호스팅되는 WordPress 사이트와 같은 웹 콘텐츠에 액세스하려고 하면 IDE 미리 보기 창에 해당 콘텐츠가 표시되지 않습니다.

가능한 원인: 기본적으로 AWS Cloud9 IDE의 애플리케이션 미리 보기 탭에서 액세스하는 모든 웹 페이지는 자동으로 HTTPS 프로토콜을 사용합니다. 페이지의 URI에 안전하지 않은 `http` 프로토콜이 있으면 자동으로 `https`로 대체됩니다. 그리고 수동으로 `https`를 다시 `http`로 변경하더라도 안전하지 않은 콘텐츠에 액세스할 수 없습니다.

권장 솔루션: IDE에서 미리 보려는 웹 사이트에서 안전하지 않은 HTTP 스크립트 또는 콘텐츠를 제거합니다. 웹 서버 또는 콘텐츠 관리 시스템에 대한 지침에서 HTTPS 구현에 대한 지침을 따릅니다.

파일을 미리 보려고 하는데 499 오류가 반환됨

문제: 특성이 포함되고 `src` 속성이 로 설정된 `<script>` 요소가 포함된 파일을 AWS Cloud9 IDE를 사용하여 미리 보려고 하면 499 오류가 발생하고 스크립트가 예상대로 실행되지 않습니다. `type module`

원인: AWS Cloud9 IDE의 파일 미리 보기 가져오기 요청은 인증을 위해 웹 브라우저에서 쿠키를 보내야 합니다. 기본적으로 웹 브라우저는 일반 스크립트 요청에 대해 쿠키를 전송하지만 `crossorigin` 속성을 추가하지 않는 한 모듈 스크립트 요청에 대해서는 쿠키를 전송하지 않습니다.

솔루션: `<script>` 요소에 `crossorigin` 속성을 추가하십시오. 예를 들어 `<script type="module" src="index.js" crossorigin></script>`입니다. 그런 다음 변경된 파일을 저장하고 미리 보기를 다시 시도해 보세요.

성능

다음 섹션에서는 성능과 관련된 문제 해결을 간략하게 설명합니다.

AWS Cloud9 IDE가 상당한 시간 동안 멈췄습니다.

문제: 시작 중 및 새로 고침을 수행할 때 AWS Cloud9 IDE 터미널이 상당한 시간 동안 정지되어 사용할 수 없게 됩니다.

원인: AWS Cloud9의 파일 감시 모듈에서 반복적으로 감시하는 많은 양의 파일이 사용자 환경에 있을 수 있습니다.

권장 솔루션: 파일 감시 깊이(최소값은 1)를 줄이고 소스 코드와 관련되지 않은 큰 폴더 또는 폴더(빌드 출력/아티팩트, 서드 파티 패키지)를 무시된 패턴에 추가하는 것을 고려할 수 있습니다. 이렇게 하려면 기본 설정 > 사용자 설정 > 파일 감시로 이동합니다. 이로 인해 CodeLenses In AWS Toolkit이 제대로 작동하지 않을 수 있다는 점에 유의하십시오.

또 다른 가능한 해결 방법은 검색할 최대 파일 수를 줄여 소스 코드와 관련이 없는 대용량 파일 및 폴더를 무시하는 것입니다. 이렇게 하려면 기본 설정 > 프로젝트 설정 > 파일에서 찾기로 이동합니다. 이렇게 하면 무시된 폴더가 파일 검색에 표시되지 않으니 주의하세요.

콘솔 경고: '최소 코드 완성 엔진으로 전환하는 중...(Switching to the minimal code completion engine...)'

문제: AWS Cloud9 콘솔에서 작업할 때 (예: IDE를 열거나 IDE의 웹 페이지를 새로 고칠 때) 다음 메시지가 표시됩니다. "하나 이상의 세션 또는 공동 작업자가 이 환경에서 활동 중입니다. 메모리를 절약하기 위해 최소 코드 완성 엔진으로 전환하는 중입니다."라는 메시지가 표시됩니다. 이 메시지와 관련하여, 코드 완성 동작이 느려지거나 간헐적일 수 있습니다.

원인: 코드 완성 엔진을 실행하느라 환경의 메모리 및 CPU 주기가 소모됩니다. 또한 협력자와 추가 세션마다 별도의 코드 완성 엔진이 필요합니다. 특히 t2.nano 및 같은 작은 인스턴스 크기에서 리소스를 너무 많이 사용하지 않으려면 최소 코드 t2.micro 완성 엔진으로 AWS Cloud9 전환하십시오.

권장 해결 방법: 장기간에 걸쳐 자주 협업을 하는 경우 EC2 환경을 만들 때 더 큰 Amazon EC2 인스턴스를 선택하거나 용량이 더 큰 인스턴스에 SSH 환경을 연결합니다.

> **Note**
>
> 더 큰 Amazon EC2 인스턴스를 선택하면 추가 요금이 발생할 AWS 계정 수 있습니다. 자세한 내용은 Amazon EC2 요금을 참조하세요.

IDE 경고: '이 환경의 메모리가 부족함(This environment is running low on memory)' 또는 '이 환경의 CPU 부하 상태가 높음(This environment has high CPU load)'

문제: IDE가 실행 중인데 '이 환경의 메모리가 부족함(this environment is running low on memory)' 또는 '이 환경의 CPU 부하 상태가 높음(this environment has high CPU load)'이라는 문구가 포함된 메시지가 표시됩니다.

원인: IDE에 지연 또는 중단 없이 계속해서 실행하는 데 사용할 수 있는 충분한 컴퓨터 리소스가 없을 수 있습니다.

권장 솔루션:

- 실행 중인 프로세스를 하나 이상 중지해 사용 가능한 메모리를 확보하십시오. 이렇게 하려면 환경용 IDE의 메뉴 모음에서 [도구, 프로세스 목록(Tools, Process List)]을 선택합니다. 중지하려는 각 프로세스에 대해 프로세스를 선택한 다음 Force Kill(강제 종료)을 선택합니다.
- 환경에서 스왑 파일을 생성합니다. 스왑 파일은 운영 체제가 가상 메모리로 사용할 수 있는 환경의 파일입니다.

  환경에서 현재 스왑 메모리를 사용하고 있는지 확인하려면 환경의 터미널 세션에서 **`top`** 명령을 실행합니다. 스왑 메모리를 사용 중인 경우 출력에 0이 아닌 `Swap` 메모리 통계가 표시됩니다(예: `Swap: 499996k total, 1280k used, 498716 free, 110672k cached`). 실제 메모리 정보 표시를 중지하려면 `Ctrl + C`를 누릅니다.

  스왑 파일을 생성하려면 환경에서 다음과 같은 명령을 실행합니다.

  ```
  sudo fallocate --length 512MB /var/swapfile && sudo chmod 600 /var/swapfile && sudo
   mkswap /var/swapfile && echo '/var/swapfile swap swap defaults 0 0' | sudo tee -a /
  etc/fstab > /dev/null
  ```

  위의 명령은 다음 작업을 수행합니다.

  1. `/var` 디렉터리에 `swapfile`이라는 512MB 파일을 생성합니다.
  2. 소유자만 읽기-쓰기가 가능하도록 `swapfile` 파일에 대한 액세스 권한을 변경합니다.
  3. `swapfile` 파일을 스왑 파일로 설정합니다.
  4. `/etc/fstab file`에 정보를 씁니다. 그러면 시스템을 재부팅할 때마다 이 스왑 파일을 사용할 수 있습니다.

위의 명령을 실행한 후 이 스왑 파일을 즉시 사용하도록 설정하려면 다음 명령을 실행합니다.

```
sudo swapon /var/swapfile
```

- 추가 컴퓨팅 리소스를 사용하여 인스턴스 또는 서버로 환경을 이동하거나 크기를 조정합니다. Amazon EC2 인스턴스를 이동하거나 크기를 조정하려면 환경 이동 및 Amazon EBS 볼륨 크기 조정 또는 암호화 섹션을 참조하세요. 다른 인스턴스 또는 서버 유형의 경우 인스턴스 또는 서버의 설명서를 참조하세요.

IDE에서 파일을 업로드할 수 없습니다. AWS Cloud9

문제: 사용자가 AWS Cloud9 IDE에서 대용량 파일을 업로드할 수 없습니다. 이러한 업로드는 실패합니다.

원인: AWS Cloud9 IDE로의 업로드 속도를 AWS Cloud9 제한하여 파일 업로드 요청 시간이 초과됩니다.

권장 해결 방법: Amazon S3에 파일을 업로드한 다음 Amazon S3를 사용하여 IDE의 CLI가 있는 환경에 파일을 다운로드하는 것이 좋습니다. AWS Cloud9 Amazon S3에 객체를 업로드하는 방법에 대한 자세한 내용은 Amazon S3 사용 설명서의 객체 업로드를 참조하세요.

IDE의 다운로드 속도가 느립니다. AWS Cloud9

문제: 사용자가 AWS Cloud9 IDE에서 파일을 다운로드하려고 할 때 다운로드 속도가 느려지는 문제를 겪고 있습니다.

원인: IDE에서 로컬 파일 시스템으로 파일을 다운로드할 때 전송 속도는 0.1MB/초의 속도로 제한됩니다.

권장 솔루션: 파일 전송 속도를 높이려면 AWS Cloud9 IDE의 CLI를 사용하여 Amazon S3에 파일을 업로드한 다음 Amazon S3를 사용하여 파일을 다운로드합니다.

사이트에 대한 연결이 안전하지 않아 IDE에서 웹 콘텐츠를 미리 볼 수 없음

문제: AWS Cloud9 EC2 환경에서 호스팅되는 WordPress 사이트와 같은 웹 콘텐츠에 액세스하려고 하면 IDE 미리 보기 창에 해당 콘텐츠가 표시되지 않습니다.

가능한 원인: 기본적으로 AWS Cloud9 IDE의 애플리케이션 미리 보기 탭에서 액세스하는 모든 웹 페이지는 자동으로 HTTPS 프로토콜을 사용합니다. 페이지의 URI에 안전하지 않은 `http` 프로토콜이 있으

면 자동으로 https로 대체됩니다. 그리고 수동으로 https를 다시 http로 변경하더라도 안전하지 않은 콘텐츠에 액세스할 수 없습니다.

권장 솔루션: IDE에서 미리 보려는 웹 사이트에서 안전하지 않은 HTTP 스크립트 또는 콘텐츠를 제거합니다. 웹 서버 또는 콘텐츠 관리 시스템에 대한 지침에서 HTTPS 구현에 대한 지침을 따릅니다.

(맨 위로 이동)

서드 파티 애플리케이션 및 서비스

다음 섹션에서는 서드 파티 애플리케이션 및 서비스와 관련된 문제 해결 문제를 간략하게 설명합니다.

tmux 세션 오류 때문에 AWS Cloud9 에서 터미널 창과 상호 작용할 수 없습니다.

문제: 에서 AWS Cloud9새 터미널 창을 시작하려고 하면 예상한 명령줄 인터페이스를 사용할 수 없습니다. 명령 프롬프트가 없으며 텍스트를 입력할 수 없습니다. `tmux: need UTF-8 locale (LC_CTYPE)` 및 `invalid LC_ALL, LC_CTYPE or LANG` 같은 오류 메시지가 반환됩니다.

가능한 원인: tmux 오류로 인해 터미널이 응답하지 않을 수 있습니다. AWS Cloud9 tmux 유틸리티를 사용합니다. 이렇게 하면 페이지가 다시 로드되거나 개발 환경에 다시 연결할 때도 터미널에 표시되는 정보를 유지합니다.

tmux 세션에서 터미널 창에 표시되는 항목은 클라이언트에 의해 처리됩니다. 클라이언트는 여러 세션을 관리할 수 있는 서버와 통신합니다. 서버와 클라이언트는 tmp 폴더에 있는 소켓을 통해 통신합니다. 만약 tmp 폴더가 개발 환경에서 누락되었거나 지나치게 제한적인 권한이 적용되면 tmux 세션을 실행할 수 없습니다. 이 경우 IDE의 터미널 창이 응답하지 않습니다.

권장 해결 방법: tmux 오류로 인해 터미널 창과 상호 작용할 수 없는 경우 올바른 권한이 있는 tmp 폴더를 생성하는 다른 방법을 사용해야 합니다. 그렇게 하면 tmux 세션을 실행할 수 있습니다. 한 가지 해결책은 LC_CTYPE을 `.bash_profile` 또는 `.bashrc` 파일에 내보내는 것입니다. 또 다른 권장 해결 방법은 호스트 관리 AWS Systems Manager 구성을 설정하는 데 사용하는 것입니다. 이렇게 하면 Amazon EC2 콘솔을 통해 관련 인스턴스에 액세스할 수 있습니다.

호스트 관리 설정

1. 먼저 AWS Cloud9 콘솔에서 환경 인스턴스의 이름을 찾으십시오. Your environments(환경) 페이지에서 관련 패널을 선택하고 View details(세부 정보 보기)를 선택하면 됩니다. 환경 세부 정보 페

이지에서 인스턴스로 이동(Go to Instance)을 선택합니다. Amazon EC2 콘솔에서 액세스해야 하는 인스턴스의 이름을 확인합니다.

2. 이제 AWS Systems Manager 콘솔로 이동하여 탐색 창에서 Quick Setup을 선택합니다.
3. 빠른 설정 페이지에서 생성(Create)을 선택합니다.
4. 구성 유형(Configuration types)에서 호스트 관리(Host Management)로 이동하고 생성(Create)을 선택합니다.
5. 호스트 관리 구성 옵션 사용자 지정(Customize Host Management configuration options)의 대상(Targets) 섹션에서 수동(Manual)을 선택합니다.
6. 액세스하려는 EC2 인스턴스를 선택한 다음 생성(Create)을 선택합니다.

인스턴스에 연결 및 명령 실행

> **Note**
>
> 다음 단계는 새로운 EC2 콘솔용입니다.

1. Amazon EC2 콘솔의 탐색 창에서 인스턴스(Instances)를 선택한 다음 연결하려는 인스턴스를 선택합니다.
2. 연결을 선택합니다.

   Connect(연결)이 활성화되지 않은 경우 먼저 인스턴스를 시작해야 할 수 있습니다.
3. Connect to your instance(인스턴스에 연결) 창에서 Connection method(연결 방법)로 Session Manager를 선택한 다음 Connect(연결)를 선택합니다.
4. 표시되는 터미널 세션 창에서 다음 명령을 입력합니다. 이 명령은 tmux 소켓을 사용할 수 있도록 올바른 권한을 가진 `tmp` 폴더를 만듭니다.

```
sudo mkdir /tmp
sudo chmod 777 /tmp
sudo rmdir /tmp/tmux-*
```

이전 버전의 Microsoft Edge 브라우저를 사용하여 IDE를 로드할 수 없음

문제: Microsoft Edge 웹 브라우저를 사용하여 AWS Cloud9 IDE를 로드하려고 하면 `HTTP403: FORBIDDEN` 오류가 반환됩니다.

가능한 원인: AWS Cloud9 IDE가 일부 이전 버전을 지원하지 않습니다Microsoft Edge.

권장 솔루션: 브라우저를 업데이트하려면 Microsoft Edge 도구 모음에서 줄임표(...) 버튼을 선택합니다. 메뉴에서 Settings(설정)를 선택한 다음 About Microsoft Edge(Microsoft Edge 정보)를 선택합니다. 업데이트가 필요한 경우 자동으로 다운로드되어 설치됩니다.

C++ 프로젝트를 디버깅할 때 **gdb**에서 오류 발생

문제: gdb IDE에서 C++ 프로젝트를 디버그하려고 하면 디버거에서 오류가 발생합니다.

가능한 원인: AWS Cloud9 환경에서 특정 EC2 인스턴스 유형 (예: `t3.small` 또는`m5.large`) 을 사용한다고 가정해 보겠습니다. IDE의 기본 제공 러너를 사용하여 C++ 프로젝트를 실행하고 디버그하려고 할 때 디버그 오류가 발생할 수 있습니다. 이 오류는 환경용으로 사전 설치된 gdb(GNU 프로젝트 디버거)의 버전이 특정 프로세서 플랫폼에서 작동하지 않아서 발생할 수 있습니다. 다음과 같은 오류 코드가 나타날 수 있습니다.

```
GDB server terminated with code 1
```

권장 솔루션: gdb가 특정 프로세서 플랫폼을 지원하지 않는 문제는 3.0 버전부터 수정되었습니다. 따라서 이전 버전의 디버거를 제거하고 최신 버전의 gdb로 업그레이드하세요.

1. 터미널에서 다음 명령을 실행하여 기존 버전의 디버거를 제거합니다. AWS Cloud9

   ```
   sudo yum -y remove gdb
   ```

2. 아카이브에서 gdb를 검색하고, 압축을 푼 후, 다음 명령을 실행하여 압축을 푼 파일이 있는 디렉터리로 이동합니다.

   ```
   wget "http://ftp.gnu.org/gnu/gdb/gdb-8.3.tar.gz"
   tar xzf gdb-8.3.tar.gz
   cd gdb-8.3
   ```

3. 다음 명령을 실행하여 디버거를 빌드합니다. 이렇게 하려면 다음 텍스트를 단일 블록으로 복사하여 붙여 넣은 다음 Return을 눌러 `make`를 실행합니다.

   ```
   ./configure --prefix=/usr \
               --with-system-readline \
               --with-python=/usr/bin/python3 &&
   make
   ```

4. 디버거를 설치합니다.

```
sudo make -C gdb install
```

5. 업데이트된 버전의 디버거가 설치되었는지 확인합니다.

```
 gdb --version
```

의 PHP 러너 관련 문제 AWS Cloud9

문제: 사용자가 PHP CLI 러너 터미널에서 출력을 볼 수 없습니다.

원인: CLI 러너를 PHP로 설정하고 디버거 모드를 활성화해야 합니다.

권장 솔루션: CLI 러너를 PHP로 설정하고 디버거 모드가 활성화되었는지 확인합니다.

Node.js 관련 GLIBC 오류

문제: 사용자가 Node.js 명령을 실행할 수 없고 GLIBC 오류가 발생합니다. 이러한 오류 메시지의 예는 다음과 같습니다.

```
node: /lib64/libm.so.6: version `GLIBC_2.27' not found (required by node)
node: /lib64/libc.so.6: version `GLIBC_2.28' not found (required by node)
```

원인: 사용 중인 인스턴스와 관련된 Node.js 버전 문제일 수 있습니다.

권장 솔루션: AWS Cloud9용 Node.js 설치 방법에 대한 자세한 내용은 1단계: 필수 도구 설치 섹션을 참조하세요.

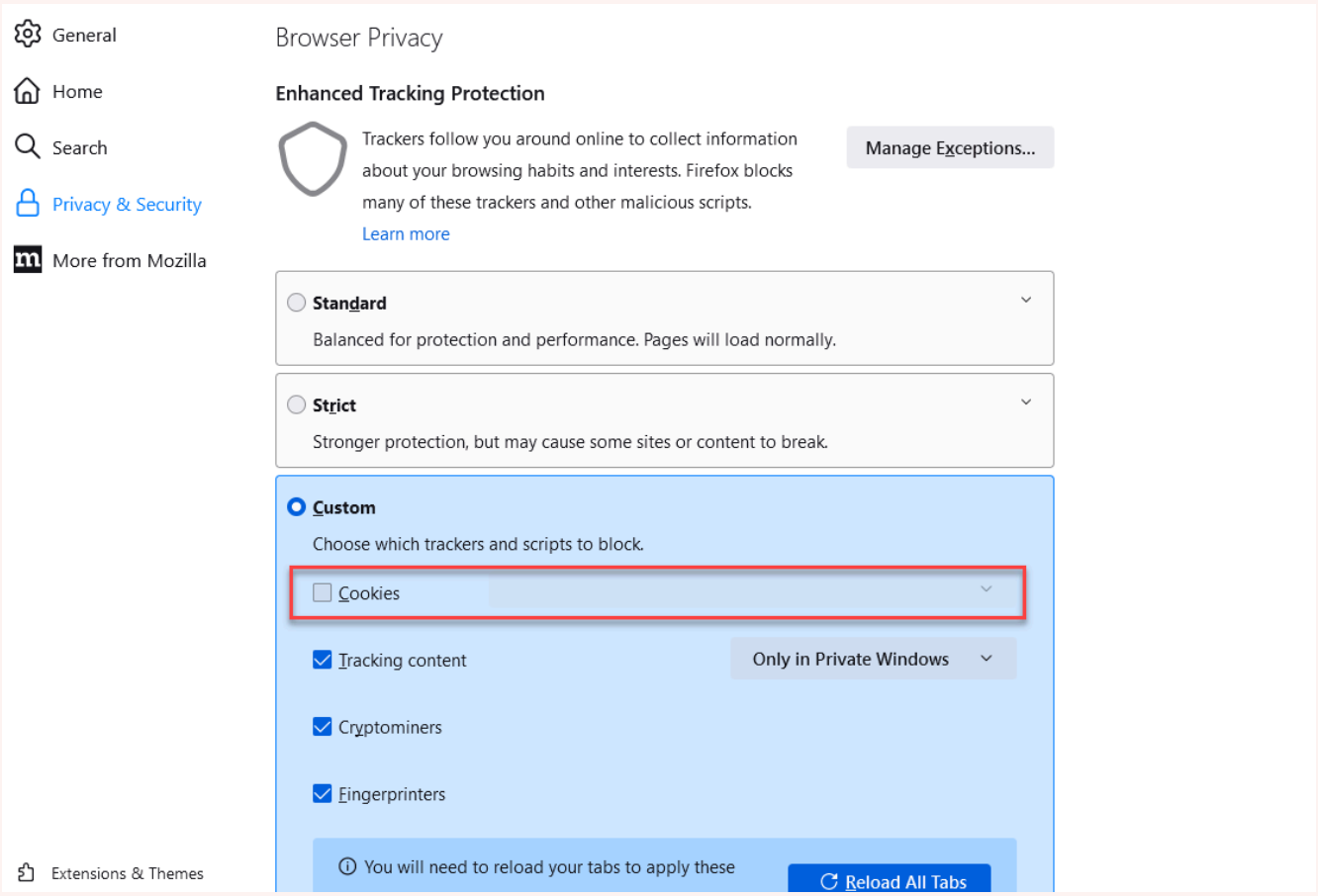
AWS Cloud9에 지원되는 브라우저

다음 표에는 AWS Cloud9에 지원되는 브라우저가 나와 있습니다.

브라우저	버전
Google Chrome	최신 3개 버전
Mozilla Firefox	최신 3개 버전
Microsoft Edge	최신 3개 버전
macOS용 Apple Safari	최근 두 버전

Warning

Mozilla Firefox를 AWS Cloud9 IDE가 설치된 기본 브라우저로 사용하는 경우 브라우저에서 AWS Cloud9 웹뷰 및 AWS Toolkit이 제대로 작동하지 않도록 하는 타사 쿠키 설정이 있습니다. 이 문제를 해결하려면 아래 이미지에 표시된 대로 브라우저 설정의 개인 정보 보호 및 보안 섹션에서 쿠키를 차단하지 않았는지 확인해야 합니다.


General
Home
Search
Privacy & Security
More from Mozilla
Extensions & Themes
Browser Privacy
Enhanced Tracking Protection
Trackers follow you around online to collect information about your browsing habits and interests. Firefox blocks many of these trackers and other malicious scripts.
Learn more
Manage Exceptions...
Standard
Balanced for protection and performance. Pages will load normally.
Strict
Stronger protection, but may cause some sites or content to break.
Custom
Choose which trackers and scripts to block.
Cookies
Tracking content
Only in Private Windows
Cryptominers
Fingerprinters
You will need to reload your tabs to apply these
Reload All Tabs

AWS Cloud9의 한도

다음 표에는 AWS Cloud9 및 관련 AWS 서비스의 한도가 나와 있습니다.

- AWS Cloud9 제한
- 관련 AWS 서비스 한도

AWS Cloud9 한도

다음 표에서는 AWS 계정의 AWS Cloud9 서비스에 대한 기본 한도에 대해 설명합니다. 별도의 언급이 없는 한, 각각의 한도는 리전별로 적용됩니다. AWS 관리 콘솔 또는 AWS CLI를 사용하여 증가를 요청할 수 있습니다. 할당량 증가를 요청하려면 Service Quotas 사용 설명서의 할당량 증가 요청 섹션을 참조하십시오.

이 증가는 즉시 허용되지 않으므로 증가가 적용되려면 이틀 정도 걸릴 수 있습니다.

Resource	기본 제한	조정 가능
최대 AWS Cloud9 EC2 개발 환경 수	• 사용자당 100개 • 계정당 200개	예
최대 SSH 환경 수	• 사용자당 100개 • 계정당 200개	예
환경 내 최대 구성원 수	기본 최대 구성원 수는 해당 환경에 대한 인스턴스의 메모리를 60MB로 나눈 값과 동일하며, 결과 값은 내림됩니다. 예를 들어, 메모리가 1GiB인 인스턴스의 경우 최대 멤버 수는 17명입니다(1GiB를 60MB로 나눈 후 반내림한 값). AWS Cloud9에서 인스턴스의 메모리를 확인할 수 없는 경우 해당 인스턴스와 연결된 각 환	아니요[1]

Resource	기본 제한	조정 가능
	경의 최대 사용자 수는 기본적으로 8명입니다. 단일 환경의 절대 최대 구성원 수는 25명입니다.	
최대 편집 가능 파일 크기	8MB	아니요

[1] 기본 최대 멤버 수를 늘리기 위한 시도로 환경을 이동할 수 있습니다. 하지만 여전히 단일 환경의 절대 최대 구성원 수는 25명입니다.

AWS Cloud9 IDE 다운로드 제한

AWS Cloud9 IDE에서 로컬 파일 시스템으로 파일을 다운로드할 때 전송 속도는 0.1MB/초의 속도로 제한됩니다. 파일 전송 속도를 높이려면 AWS Cloud9 IDE의 CLI를 사용하여 Amazon S3에 파일을 업로드한 다음 Amazon S3를 사용하여 Amazon S3에서 파일을 다운로드하세요.

관련 AWS 서비스 한도

Amazon Elastic Block Store(Amazon EBS) 볼륨의 최대 수	5,000 자세한 내용은 Amazon Web Services 일반 참조의 Amazon Elastic Block Store(Amazon EBS) 제한을 참조하세요.
최대 AWS CloudFormation 스택 수	200 자세한 내용은 AWS CloudFormation 사용 설명서에서 AWS CloudFormation 한도를 참조하세요.
Amazon EC2 한도	Amazon Web Services 일반 참조의 Amazon Elastic Compute Cloud(Amazon EC2) 제한을 참조하세요.

AWS Cloud9 사용 설명서 기록

이 주제에는 AWS Cloud9 사용 설명서에 대한 중요한 변경 사항 목록이 포함되어 있습니다. 이 설명서의 업데이트에 대한 알림을 받으려면 RSS 피드를 구독할 수 있습니다.

최신 업데이트

다음 표는 2019년 3월 이후 AWS Cloud9 사용 설명서에서 변경된 중요 사항을 기술한 것입니다.

변경 사항	설명	날짜
AWS Cloud9용 Amazon Linux 2023에 대한 지원이 추가되었습니다.	이제 AWS Cloud9에서 Amazon Linux 2023을 지원합니다.	2023년 12월 15일
Node.js 자습서가 업데이트되었습니다.	Amazon Linux 2 및 Node.js 18에 대한 지원과 관련된 Node.js 자습서가 업데이트되었습니다.	2023년 10월 23일
Amazon VPC 대시보드를 사용하여 Amazon VPC를 생성하는 방법에 대한 섹션을 업데이트했습니다.	Amazon VPC 대시보드를 사용하여 Amazon VPC를 생성하는 방법에 대한 섹션을 업데이트했습니다.	2023년 7월 27일
Amazon EventBridge 스키마 작업 관련 섹션	AWS Cloud9용 AWS Toolkit을 사용한 Amazon EventBridge 스키마 작업 관련 섹션이 추가되었습니다.	2022년 12월 15일
CodeCatalyst 섹션 추가됨	새로운 Amazon CodeCatalyst 서비스 관련 섹션이 추가되었습니다.	2022년 12월 2일
AWS IoT 콘텐츠 추가됨	AWS IoT 사용에 대한 섹션이 추가되었습니다.	2022년 11월 1일
AWS Cloud9 IDE용 Amazon ECR 서비스 개요	AWS Cloud9 IDE에서 액세스할 수 있는 Amazon ECS 서비	2022년 10월 20일

	스의 특징과 기능에 대한 개요 및 안내가 추가되었습니다.	
AWS Cloud9 통합 개발 환경(IDE)의 AWS CDK 작업	AWS Cloud9 통합 개발 환경(IDE)의 AWS CDK 작업에 대한 섹션이 추가되었습니다.	2022년 10월 5일
Amazon ECR 콘텐츠 추가	AWS Amazon ECR 사용에 대한 섹션이 추가되었습니다.	2022년 10월 4일
규정 준수 검증	AWS Cloud9이 범위에 포함된 규정 준수 프로그램의 목록이 업데이트되었습니다.	2022년 3월 4일
향상된 Java 지원	Java로 작업할 때 개발 경험을 개선하기 위한 추가 언어 지원. 주요 생산성 기능에는 코드 완성, 오류에 대한 linting, 컨텍스트별 작업, 디버깅 옵션(예: 중단점 및 단계별 실행)이 포함됩니다.	2022년 1월 18일
업데이트된 AWSServiceRoleForAWSCloud9	License Manager를 사용하여 EC2 인스턴스를 지원하도록 서비스 연결 역할이 업데이트되었습니다.	2022년 1월 12일
Step Functions 문서 지원	Step Functions를 사용하여 상태 머신을 생성, 편집 및 실행하는 방법을 설명하는 내용이 추가되었습니다.	2021년 12월 20일
AWS Systems Manager 문서 지원	Systems Manager 자동화 문서를 설명하는 내용이 추가되었습니다.	2021년 12월 20일
Amazon Elastic Container Service Exec 사용 설명서 작성	다음은 Amazon ECS Exec의 개요입니다.	2021년 12월 13일

AWS IoTAWS Cloud9 IDE 서비스 사용 설명서 작성	이 사용 설명서에서는 AWS Cloud9 IDE용 AWS IoT 서비스 사용을 시작하는 방법을 설명합니다.	2021년 11월 22일
AWS 리소스에 대한 지원	리소스 및 관련 문서를 보기 위한 인터페이스 옵션과 함께 리소스 유형에 대한 액세스 지원이 추가되었습니다.	2021년 11월 5일
AWS Cloud9 IDE용 Amazon ECR 서비스 개요	AWS Cloud9 IDE에서 액세스할 수 있는 Amazon ECR 서비스의 특징과 기능에 대한 개요 및 안내가 추가되었습니다.	2021년 10월 14일
App Runner 지원	AWS App Runner에서 AWS 도구 키트에 대한 지원이 추가되었습니다.	2021년 9월 30일
아프리카(케이프타운) 및 아시아 태평양(오사카) 리전에서도 AWS Cloud9 사용 가능	이제 아프리카(케이프타운) 및 아시아 태평양(오사카) 리전에서도 AWS Cloud9을 사용할 수 있습니다. 이 리전 및 다른 AWS 리전과 관련한 서비스 엔드포인트 및 서비스 할당량에 대한 자세한 내용은 Amazon Web Services 일반 참조의 AWS Cloud9 섹션을 참조하세요.	2021년 9월 1일
AWS 도구 키트의 CloudWatch Logs 및 Amazon S3	AWS Cloud9에 대한 CloudWatch Logs 지원이 AWS 도구 키트에 추가되었습니다. Amazon S3 버킷에 현재 파일을 업로드할 수 있는 새로운 기능입니다.	2021년 7월 16일

Amazon S3용 VPC 엔드포인트	Amazon S3 VPC 엔드포인트 구성에 대한 지원이 추가되었습니다.	2021년 4월 22일
Git 패널을 통한 시각적 소스 제어	개발자는 Git 패널을 사용하여 사용자 인터페이스에서 Git 명령을 실행할 수 있습니다.	2021년 2월 1일
프라이빗 서브넷으로 환경 인스턴스 시작	Systems Manager를 통해 액세스되는 EC2 인스턴스를 프라이빗 서브넷으로 시작할 수 있도록 하는 지원 기능이 추가되었습니다.	2021년 1월 21일
AWS 도구 키트의 통합	이제 AWS 탐색기 창을 통해 AWS Toolkit을 사용하여 AWS 서비스를 탐색하고 상호 작용할 수 있습니다.	2020년 12월 11일
AWS CloudFormation 및 수신하지 않는 EC2 환경	AWS CloudFormation 템플릿을 사용하여 수신하지 않는 EC2 환경을 생성하는 방법에 대한 확장 문서.	2020년 10월 29일
Amazon Linux 2 기반 EC2 환경	콘솔에서 EC2 환경을 생성할 때 EC2 인스턴스에 대해 Amazon Linux 2 AMI를 선택할 수 있습니다.	2020년 10월 7일
Systems Manager를 사용하는 수신하지 않는 EC2 인스턴스	AWS Systems Manager로 프라이빗 EC2 인스턴스에 액세스하기 위한 지원이 추가되었습니다.	2020년 8월 12일
향상된 AWS 서버리스 애플리케이션 로컬 디버깅 기능	AWS 서버리스 애플리케이션 로컬 디버깅 기능에 대한 지원이 새로 추가되었습니다.	2020년 7월 30일

EU(밀라노) 리전에서도 AWS Cloud9 사용 가능	이제 EU(밀라노) 리전에서도 AWS Cloud9을 사용할 수 있습니다. 이 리전 및 다른 AWS 리전과 관련한 서비스 엔드포인트 및 서비스 할당량에 대한 자세한 내용은 Amazon Web Services 일반 참조의 AWS Cloud9 섹션을 참조하세요.	2020년 7월 29일
Amazon EBS 암호화	AWS Cloud9 개발 환경에 사용되는 EC2 인스턴스용 Amazon EBS 볼륨을 암호화하는 방법을 설명하는 섹션.	2020년 7월 3일
AWS Cloud9에 대한 리전 지원 추가	이제 미국 서부(캘리포니아 북부), 아시아 태평양(홍콩), EU(파리), 중동(바레인), 남아메리카(상파울루) 리전에서도 AWS Cloud9을 사용할 수 있습니다. 이 리전 및 다른 AWS 리전과 관련한 서비스 엔드포인트 및 서비스 할당량에 대한 자세한 내용은 Amazon Web Services 일반 참조의 AWS Cloud9 섹션을 참조하세요.	2020년 5월 7일
보안	보안 장이 AWS Cloud9 사용 설명서에 추가되었습니다.	2020년 4월 30일
태그	태그를 사용하여 AWS Cloud9 리소스에 대한 액세스를 제어하고 결제 정보를 관리할 수 있습니다.	2020년 1월 22일

AWS Cloud9에 대한 리전 지원 추가	이제 아시아 태평양(뭄바이), 아시아 태평양(서울), 아시아 태평양(시드니), 캐나다(중부), EU(런던) 및 EU(스톡홀름) 리전에서도 AWS Cloud9을 사용할 수 있습니다. 이 리전 및 다른 AWS 리전과 관련한 서비스 엔드포인트 및 서비스 할당량에 대한 자세한 내용은 Amazon Web Services 일반 참조의 AWS Cloud9 섹션을 참조하세요.	2019년 12월 18일
업데이트: 문제 해결, 환경을 열 수 없음	IDE를 여는 데 타사 쿠키가 더 이상 필요하지 않습니다.	2019년 11월 6일
추가: 문제 해결, 서드 파티 쿠키 비활성화	IDE를 여는 데 서드 파티 쿠키가 더 이상 필요하지 않습니다. 하지만 애플리케이션 미리 보기 또는 파일 미리 보기 기능에는 필요합니다. 문제 해결 항목에서 이에 대한 정보를 찾을 수 있습니다.	2019년 11월 6일
문서 조직	특히 처음 사용자의 탐색을 돕기 위해 조직 변경 사항이 사용 설명서에 적용되었습니다.	2019년 8월 15일
EU(프랑크푸르트) 리전에서도 AWS Cloud9 사용 가능	이제 EU(프랑크푸르트) 리전에서도 AWS Cloud9을 사용할 수 있습니다. 이 리전 및 다른 AWS 리전과 관련한 서비스 엔드포인트 및 서비스 할당량에 대한 자세한 내용은 Amazon Web Services 일반 참조의 AWS Cloud9 섹션을 참조하세요.	2019년 5월 15일

LAMP 샘플이 추가됨	AWS Cloud9를 LAMP(Linux, Apache HTTP Server, MySQL, PHP)와 사용하는 방법을 보여 주는 샘플이 새로 추가되었습니다. 자세한 내용은 AWS Cloud9의 LAMP 샘플을 참조하세요.	2019년 5월 10일
WordPress 샘플이 추가됨	WordPress와 함께 AWS Cloud9를 사용하는 방법을 보여 주는 새로운 샘플이 추가되었습니다. 자세한 내용은 AWS Cloud9의 WordPress 샘플을 참조하세요.	2019년 4월 19일
아시아 태평양(도쿄) 리전에서도 AWS Cloud9 사용 가능	이제 아시아 태평양(도쿄) 리전에서도 AWS Cloud9을 사용할 수 있습니다. 이 리전 및 다른 AWS 리전과 관련한 서비스 엔드포인트 및 서비스 할당량에 대한 자세한 내용은 Amazon Web Services 일반 참조의 AWS Cloud9 섹션을 참조하세요.	2019년 4월 4일

EC2 환경의 Ubuntu Server 지원에 대한 정보 추가	AWS Cloud9 콘솔을 사용하여 Ubuntu Server에 연결하는 AWS Cloud9 EC2 개발 환경을 생성하는 지침이 추가되었습니다. 자세한 내용은 EC2 환경 생성 섹션을 참조하세요. 현재는 코드를 사용하여 AWS CLI, AWS CloudFormation, AWS SDK, Tools for Windows PowerShell, AWS Cloud9 API 등을 통해 Ubuntu Server에 연결하는 AWS Cloud9 EC2 개발 환경을 생성할 수 없습니다. 이러한 방법은 향후 지원될 것입니다.	2019년 4월 2일

이전 업데이트

아래 표에서는 2019년 4월 이전의 AWS Cloud9 사용 설명서에 적용되는 주요 변경 사항을 설명합니다.

변경 사항	설명	변경 날짜
학생, 교육자 및 기업을 위한 시작하기 지침 추가	학생, 교육자 및 기업을 위한 단계를 포함하도록 AWS Cloud9 시작하기에 대한 지침이 확장되었습니다. 자세한 내용은 AWS Cloud9 설정 섹션을 참조하세요.	2019년 2월 7일
AWS CloudTrail 지원 추가	AWS CloudTrail가 이제 AWS Cloud9를 지원합니다. 자세한 내용은 AWS CloudTrail을 사용하여 AWS Cloud9 API 호출 로깅 섹션을 참조하세요.	2019년 1월 21일

변경 사항	설명	변경 날짜
공유 VPC 지원 추가	AWS Cloud9이 이제 Amazon VPC에서 공유 VPC를 지원합니다. 자세한 내용은 다음에 대한 Amazon VPC 요구 사항 AWS Cloud9 섹션을 참조하세요.	2018년 12월 7일
AWS RoboMaker 통합 추가	AWS Cloud9가 이제 지능형 로보틱스 애플리케이션을 대규모로 쉽게 개발, 테스트 및 배포할 수 있는 서비스인 AWS RoboMaker를 지원합니다. 자세한 내용은 AWS RoboMaker 개발자 가이드에서 AWS RoboMaker 시작하기 및 AWS Cloud9으로 개발을 참조하세요.	2018년 11월 26일
언어 프로젝트를 위한 추가 생산성 기능에 대한 정보가 추가됨	AWS Cloud9 IDE가 이제 언어 프로젝트의 맥락에서 일부 언어에 대해 추가 생산성 기능을 제공합니다. 자세한 내용은 향상된 TypeScript 지원 및 기능 섹션을 참조하세요.	2018년 10월 2일

변경 사항	설명	변경 날짜
Go(이동) 창이 추가되고 Navigate(탐색) 및 Commands(명령) 창은 제거됨	[이동(Go)] 창이 2018년 10월 2일 또는 그 이후에 생성된 환경의 AWS Cloud9 IDE에 추가되었습니다. 이 새 창은 [탐색(Navigate)] 및 [명령(Commands)] 창을 대체하며, 이 두 창은 2018년 10월 2일 또는 그 이후에 생성된 환경의 IDE에서 모두 제거되었습니다. 자세한 내용은 IDE 둘러보기에서 10단계: 이동 창 섹션을 참조하세요.	2018년 10월 2일
AWS CDK 샘플이 추가됨	AWS Cloud Development Kit(AWS CDK)와 함께 AWS Cloud9을 사용하는 방법을 보여 주는 새로운 샘플을 추가했습니다. 자세한 내용은 AWS Cloud9용 AWS CDK 자습서 섹션을 참조하세요.	2018년 8월 30일
EC2 환경에 자동으로 추가되는 SSH IP 주소 제한 사항에 대한 정보 추가	2018년 7월 31일 이후에 생성된 AWS Cloud9 EC2 개발 환경의 경우, AWS Cloud9이 이제 AWS Cloud9이 SSH를 통해 연결하기 위해 사용하는 IP 주소 범위로만 수신 SSH 트래픽을 자동으로 제한합니다. 자세한 내용은 AWS Cloud9의 인바운드 SSH IP 주소 범위 섹션을 참조하세요.	2018년 7월 31일

변경 사항	설명	변경 날짜
도커 샘플이 추가됨	도커와 함께 AWS Cloud9를 사용하는 방법을 보여 주는 새로운 샘플을 추가했습니다. 자세한 내용은 에 대한 도커 튜토리얼 AWS Cloud9 섹션을 참조하세요.	2018년 6월 19일
Java, .NET Core 및 TypeScript에 대한 샘플이 추가됨	Java, .NET Core, TypeScript와 함께 AWS Cloud9를 사용하는 방법을 보여 주는 새로운 샘플을 추가했습니다. 자세한 내용은 AWS Cloud9용 Java 자습서, AWS Cloud9용 .NET 자습서 및 AWS Cloud9용 TypeScript 자습서 섹션을 참조하세요.	2018년 5월 29일
지원되는 브라우저 목록이 추가됨	AWS Cloud9에 지원되는 브라우저에 대한 정보를 추가했습니다. 자세한 내용은 AWS Cloud9에 지원되는 브라우저 섹션을 참조하세요.	2018년 5월 23일
SSH IP 트래픽 제한 정보가 추가됨	AWS Cloud9가 SSH를 통해 호스트에 연결하기 위해 사용하는 IP 주소 범위로만 수신 트래픽을 제한하는 방법에 대한 정보를 추가했습니다. 자세한 내용은 AWS Cloud9의 인바운드 SSH IP 주소 범위 섹션을 참조하세요.	2018년 4월 19일

변경 사항	설명	변경 날짜
애플리케이션을 미리 보고 실행 중인 애플리케이션을 공유하는 것과 관련된 문제 해결사를 추가했습니다.	애플리케이션을 미리 보고 실행 중인 애플리케이션을 공유하는 것과 관련된 새로운 문제 해결사를 추가했습니다. 자세한 내용은 애플리케이션 미리 보기 탭에 오류가 표시되거나 이 탭이 비어 있음 및 IDE 외부에서 실행 중인 애플리케이션을 표시할 수 없음 단원을 참조하세요.	2018년 4월 19일
File Revision History(파일 개정 이력) 정보가 추가됨	IDE에서 [파일 개정 이력(File Revision History)] 창을 사용하는 방법에 대한 정보가 추가되었습니다. 자세한 내용은 AWS Cloud9 통합 개발 환경(IDE)의 파일 개정 작업 섹션을 참조하세요.	2018년 4월 19일
환경 열기와 관련된 문제 해결사 추가	AWS Cloud9 개발 환경 열기와 관련된 새로운 문제 해결사를 추가했습니다. 자세한 내용은 환경을 열 수 없음 섹션을 참조하세요.	2018년 3월 19일
AWS Cloud9 설치 프로그램과 관련된 문제 해결사가 추가됨	AWS Cloud9 설치 프로그램과 관련된 새로운 문제 해결사를 추가했습니다. 자세한 내용은 AWS Cloud9 설치 프로그램이 중단되거나 실패합니다. 섹션을 참조하세요.	2018년 3월 19일

변경 사항	설명	변경 날짜
AWS CodePipeline 정보 추가됨	AWS CodePipeline과 함께 AWS Cloud9를 사용하는 방법에 대한 정보를 추가했습니다. 자세한 내용은 AWS Cloud9 통합 개발 환경(IDE)의 AWS CodePipeline 작업 섹션을 참조하세요.	2018년 2월 13일
AWS CloudShell 정보 추가됨	AWS CloudShell과 함께 AWS Cloud9을 사용하는 방법에 대한 정보를 추가했습니다. 자세한 내용은 AWS Cloud9용 AWS Command Line Interface 및 aws-shell 자습서 섹션을 참조하세요.	2018년 1월 19일
GitHub에서 설명서를 볼 수 있음	이제 GitHub에서도 이 설명서를 사용할 수 있습니다. 또한 GitHub를 이용해 이 설명서의 콘텐츠에 대한 피드백과 변경 요구사항을 제출할 수도 있습니다. 자세한 내용을 확인하려면 설명서의 탐색 모음에서 GitHub에서 편집 아이콘을 선택하거나, GitHub 웹 사이트의 awsdocs/aws-cloud9-user-guide 리포지토리를 참조하세요.	2018년 1월 10일
Kindle 형식 가용성	이 설명서가 이제 Amazon Kindle 형식으로 제공됩니다. 자세한 내용은 설명서의 탐색 모음에서 Kindle 열기 섹션을 참조하십시오.	2018년 1월 2일

변경 사항	설명	변경 날짜
Amazon Lightsail 정보 추가됨	Amazon Lightsail과 함께 AWS Cloud9를 사용하는 방법에 대한 정보를 추가했습니다. 자세한 내용은 AWS Cloud9 통합 개발 환경(IDE)의 Amazon Lightsail 인스턴스스 섹션을 참조하세요.	2017년 12월 19일
AWS의 환경 설정 설명 추가	AWS Cloud9 개발 환경의 특정 AWS 설정에 대한 설명을 추가했습니다. 자세한 내용은 AWS Cloud9 통합 개발 환경(IDE)의 AWS 프로젝트 및 사용자 설정 작업 섹션을 참조하세요.	2017년 12월 7일
AWS 계정 루트 사용자를 위한 시작하기 지침 및 팀을 위한 고급 설정 단계가 추가됨	AWS 계정 루트 사용자로 AWS Cloud9를 사용하기 위한 설정 단계를 추가했습니다. 팀에서 AWS Cloud9를 사용하기 위한 고급 설정 단계를 추가했습니다. 자세한 내용은 AWS Cloud9 설정 섹션을 참조하세요.	2017년 12월 5일
환경 요구 사항에 맞게 범위 확장	AWS Cloud9 SSH 개발 환경에 연결하기 위한 Amazon EC2 인스턴스 또는 자체 서버에 대한 요구 사항 범위를 확장했습니다. 자세한 내용은 SSH 환경 호스트 요구 사항 섹션을 참조하세요.	2017년 12월 4일
최초 문서 릴리스	이는 AWS Cloud9 사용 설명서의 최초 릴리스입니다.	2017년 11월 30일

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.