Guida per gli sviluppatori

AWS Serverless Application Model

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Serverless Application Model: Guida per gli sviluppatori

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

I marchi e l’immagine commerciale di Amazon non possono essere utilizzati in relazione a prodotti o servizi che non siano di Amazon, in una qualsiasi modalità che possa causare confusione tra i clienti o in una qualsiasi modalità che denigri o discrediti Amazon. Tutti gli altri marchi non di proprietà di Amazon sono di proprietà dei rispettivi proprietari, che possono o meno essere affiliati, collegati o sponsorizzati da Amazon.

Table of Contents

Che cos'è AWS SAM? ... 1
Quando usare AWS SAM ... 1
Funzionalità principali ... 2
Informazioni correlate ... 3
Come funziona ... 3
Quali sono le specifiche del AWS SAM modello? ... 4
Cosa sono il AWS SAM progetto e il modello? AWS SAM ... 4
Che cos'è il? AWS SAMCLI ... 11
Ulteriori informazioni ... 19
Fasi successive ... 19
Concetti serverless ... 19
Concetti serverless ... 20
Infrastruttura come codice ... 21
Strumenti IaC per applicazioni serverless AWS ... 21
Confronto degli strumenti IaC per applicazioni serverless ... 22
Ulteriori informazioni ... 23
Nozioni di base ... 24
Prerequisiti ... 24
Passaggio 1: crea un account AWS ... 25
Fase 2: Creare un account utente IAM ... 25
Passaggio 3: Crea un ID della chiave di accesso e una chiave di accesso segreta ... 26
Fase 4: Installare il AWS CLI ... 29
Passaggio 5: Utilizzare AWS CLI per configurare le AWS credenziali ... 29
Passaggio 6: (Opzionale) Installazione di AWS Toolkit for VS Code ... 30
Fasi successive ... 31
Installa il AWS SAMCLI ... 31
Installazione di AWS SAMCLI ... 32
Risoluzione degli errori di installazione ... 42
Fasi successive ... 43
Facoltativo: verifica il programma di installazione AWS SAMCLI ... 44
Tutorial Hello World ... 57
Prerequisiti ... 58
Passaggio 1: inizializza l'applicazione Hello World di esempio ... 58
Fase 2: Crea la tua applicazione ... 63

Passo 3: Distribuisci la tua applicazione su Cloud AWS ... 64
Passaggio 4: Esegui l'applicazione ... 69
Fase 5: Interagite con la vostra funzione nel Cloud AWS ... 71
Fase 6: Modificare e sincronizzare l'applicazione con Cloud AWS ... 71
Passaggio 7: (Facoltativo) Testa l'applicazione localmente ... 75
Fase 8: Eliminare l'applicazione dal Cloud AWS ... 77
Risoluzione dei problemi ... 78
Ulteriori informazioni ... 78
Conversione da Lambda ... 78
Ulteriori informazioni ... 79
Come usare AWS SAM ... 80
AWS SAM CLI ... 81
Come vengono AWS SAMCLI documentati i comandi ... 81
Configurazione del AWS SAMCLI ... 83
Come creare il file di configurazione (il `samconfig` file) ... 84
Configura le impostazioni del progetto ... 84
Configurare le credenziali e le impostazioni di base ... 89
Comandi principali ... 89
Test locali con AWS SAMCLI ... 90
Testa le AWS SAM applicazioni localmente ... 91
Prova i modelli localmente CloudFormation ... 91
Testa le applicazioni AWS CDK localmente ... 91
Testa le applicazioni Terraform localmente ... 91
AWS SAM modello ... 93
Anatomia dei modelli ... 94
YAML ... 94
Sezioni del modello ... 95
Fasi successive ... 97
Elementi globali ... 98
Risorse e proprietà ... 103
AWS::Serverless::Api ... 104
AWS::Serverless::Application ... 163
AWS::Serverless::CapacityProvider ... 168
AWS::Serverless::Connector ... 178
AWS::Serverless::Function ... 192
AWS::Serverless::GraphQLApi ... 350

AWS::Serverless::HttpApi ... 385
AWS::Serverless::LayerVersion ... 417
AWS::Serverless::SimpleTable ... 423
AWS::Serverless::StateMachine ... 429
Risorse generate ... 471
Riferimento alle risorse generate CloudFormation ... 471
Scenari di CloudFormation risorse generate ... 472
AWS::Serverless::Api ... 475
AWS::Serverless::Application ... 477
AWS::Serverless::CapacityProvider ... 477
AWS::Serverless::Connector ... 478
AWS::Serverless::Function ... 479
AWS::Serverless::GraphQLApi ... 485
AWS::Serverless::HttpApi ... 487
AWS::Serverless::LayerVersion ... 488
AWS::Serverless::SimpleTable ... 489
AWS::Serverless::StateMachine ... 489
Attributi delle risorse supportati ... 491
Eccezioni ... 492
estensioni Gateway API ... 492
Funzioni intrinseche ... 494
Applicazioni di esempio ... 495
Elabora eventi DynamoDB ... 495
Prima di iniziare ... 495
Passaggio 1: inizializza l'applicazione ... 495
Fase 2: Testare l'applicazione localmente ... 496
Fase 3: Package dell'applicazione ... 496
Fase 4: Distribuire l'applicazione ... 497
Fasi successive ... 498
Elabora gli eventi di Amazon S3 ... 498
Prima di iniziare ... 498
Passaggio 1: inizializza l'applicazione ... 499
Fase 2: Package dell'applicazione ... 499
Fase 3: Distribuire l'applicazione ... 500
Fase 4: Prova l'applicazione localmente ... 501
Fasi successive ... 501

Sviluppa la tua applicazione ... 502
Crea la tua applicazione ... 502
Inizializza una nuova applicazione serverless ... 503
Opzioni per sam init ... 509
Risoluzione dei problemi ... 509
Esempi ... 509
Ulteriori informazioni ... 510
Fasi successive ... 510
Definisci la tua infrastruttura ... 511
Definisci le risorse dell'applicazione ... 511
Configurare l’accesso ... 513
Controlla l'accesso alle API ... 592
Aumenta l'efficienza con i livelli ... 605
Riutilizza il codice ... 608
Gestisci eventi basati sul tempo ... 612
Orchestrazione delle applicazioni ... 615
Configura la firma del codice ... 616
Convalida dei file AWS SAM modello ... 620
Crea la tua applicazione ... 620
Introduzione a sam build ... 621
Compilazione predefinita ... 635
Personalizza la tua build ... 644
Testa la tua applicazione ... 674
Introduzione a sam local ... 674
Utilizzo del sam local comando ... 675
Introduzione a sam local generate-event ... 675
Introduzione a sam local invoke ... 682
Introduzione a sam local start-api ... 688
Introduzione a sam local start-lambda ... 694
Invoca localmente le funzioni ... 697
File di variabili di ambiente ... 698
Livelli ... 700
Ulteriori informazioni ... 700
API Gateway eseguito localmente ... 700
File di variabili di ambiente ... 701
Livelli ... 703

Esegui il test con sam remote test-event ... 703
Configura AWS SAMCLI da usare sam remote test-event ... 704
Utilizzo del comando sam remote test-event ... 704
Utilizzo di eventi di test condivisibili ... 707
Gestione di eventi di test condivisibili ... 707
Esegui il test con sam remote invoke ... 709
Utilizzo del comando sam remote invoke ... 710
Utilizzo delle opzioni del comando sam remote invoke ... 714
Configura il file di configurazione del progetto ... 719
Esempi ... 720
Link correlati ... 735
Automatizza i test di integrazione ... 736
Genera payload di esempio ... 737
Testa ed esegui il debug di funzioni durevoli ... 738
Flusso di lavoro di test locale ... 738
Scenari di test comuni ... 739
Risoluzione dei problemi ... 739
Suggerimenti per il debug ... 740
Ulteriori informazioni ... 741
Esegui il debug della tua applicazione ... 742
Funzioni di debug locali ... 742
Utilizzo dei AWS Toolkit ... 743
Esecuzione AWS SAM locale in modalità debug ... 745
Passa più argomenti di runtime ... 746
Convalida con cfn-lint ... 746
Esempi ... 747
Ulteriori informazioni ... 747
Implementa la tua applicazione e le tue risorse ... 748
Introduzione a sam deploy ... 748
Prerequisiti ... 749
Distribuzione di applicazioni utilizzando sam deploy ... 749
Best practice ... 760
Opzioni per sam deploy ... 760
Risoluzione dei problemi ... 760
Esempi ... 760
Ulteriori informazioni ... 769

Opzioni di implementazione ... 769
Come utilizzare il comando AWS SAMCLI per la distribuzione manuale ... 769
Implementa con CI/CD sistemi e pipeline ... 770
Implementazioni graduali ... 770
Risoluzione dei problemi di distribuzione utilizzando il AWS SAMCLI ... 770
Ulteriori informazioni ... 700
Implementazione con CI/CD sistemi e pipeline ... 771
Cos'è una pipeline? ... 772
Come AWS SAM carica i file locali ... 773
Genera una pipeline iniziale ... 781
Personalizza le pipeline iniziali ... 787
Automatizza le tue implementazioni ... 789
Usa l'autenticazione OIDC ... 793
Introduzione a sam sync ... 796
Rileva e sincronizza automaticamente le modifiche locali al Cloud AWS ... 797
Personalizza le modifiche locali da sincronizzare con Cloud AWS ... 798
Prepara la tua applicazione nel cloud per il test e la convalida ... 799
Opzioni per il comando sam sync ... 799
Risoluzione dei problemi ... 802
Esempi ... 802
Ulteriori informazioni ... 809
Monitora la tua applicazione ... 810
Application Insights ... 810
Configurazione di CloudWatch Application Insights con AWS SAM ... 810
Fasi successive ... 814
Lavorare con i log ... 814
Recupero dei log per pila CloudFormation ... 814
Recupero dei log in base al nome della funzione Lambda ... 815
Tronchi di coda ... 815
Visualizzazione dei log per un intervallo di tempo specifico ... 815
Filtraggio dei log ... 815
Evidenziazione degli errori ... 815
Bella stampa in JSON ... 815
AWS SAM riferimento ... 816
AWS SAM le specifiche e il modello AWS SAM ... 816
AWS SAMCLIriferimento ai comandi ... 816

AWS SAM modelli di policy ... 817
Argomenti ... 817
AWS SAMCLIcomandi ... 817
sam build ... 819
sam delete ... 824
sam deploy ... 826
sam init ... 832
sam list ... 836
sam local callback ... 844
sam local execution ... 849
sam local generate-event ... 856
sam local invoke ... 857
sam local start-api ... 863
sam local start-lambda ... 868
sam logs ... 873
sam package ... 877
sam pipeline bootstrap ... 880
sam pipeline init ... 885
sam publish ... 886
sam remote callback ... 889
sam remote execution ... 895
sam remote invoke ... 902
sam remote test-event ... 907
sam sync ... 914
sam traces ... 921
sam validate ... 923
AWS SAMCLIgestione ... 924
AWS SAMCLIfile di configurazione ... 925
Gestione delle AWS SAMCLI versioni ... 932
Impostazione delle AWS credenziali ... 941
AWS SAMCLItelemetria ... 943
Risoluzione dei problemi ... 945
riferimento del connettore ... 951
Tipi di risorse di connettore supportati ... 951
Politiche IAM create dai connettori ... 962
Installazione di Docker ... 985

Installazione di Docker ... 985
Fasi successive ... 988
Installazione di Finch ... 988
Installazione di Finch ... 989
Configurazione Finch con CLI AWS SAM ... 991
Verifica dell'installazione ... 992
Interrompere Finch ... 993
Risoluzione dei problemi ... 994
Fasi successive ... 995
Archivi di immagini ... 995
Archivio di immagini URIs ... 996
Esempi ... 998
Implementazione graduale ... 998
Implementazione graduale di una funzione Lambda per la prima volta ... 1001
Ulteriori informazioni ... 1002
Note importanti ... 1002
2023 ... 1003
Supporto di Terraform ... 1004
Nozioni di base ... 1004
Prerequisiti ... 1004
Utilizzo dei AWS SAMCLI comandi con Terraform ... 1005
Configurazione per Terraform progetti ... 1005
Configurazione di Terraform Cloud ... 1011
AWS SAMCLIUsando con Terraform ... 1013
Test locali con sam local invoke ... 1013
Test locali con sam local start-api ... 1013
Test locali con sam local start-lambda ... 1015
Limitazioni di Terraform ... 1016
Utilizzo AWS SAMCLI con Serverless.TF ... 1016
Riferimento Terraform ... 1017
AWS SAM riferimento alle funzionalità supportate ... 1017
Terraformriferimento specifico ... 1017
metadati sam ... 1017
AWS SAMCLITerraformsupporto ... 1020
Che cos'è il? AWS SAMCLI ... 1021
Come si usa il AWS SAMCLI conTerraform? ... 1021

Fasi successive 1022
Pubblicazione per l'utilizzo da parte di altri 1023
Prerequisiti 1023
Pubblicazione di una nuova applicazione 1025
Passaggio 1: aggiungere una `Metadata` sezione al AWS SAM modello 1025
Fase 2: Package dell'applicazione 1025
Fase 3: Pubblicare l'applicazione 1026
Passaggio 4: Condividi l'applicazione (opzionale) 1026
Pubblicazione di una nuova versione di un'applicazione esistente 1027
Argomenti aggiuntivi 1027
Proprietà della sezione Metadati 1027
Properties 1027
Casi d’uso 1030
Esempio 1031
Cronologia dei documenti 1033
..... mlvii

Cos'è il AWS Serverless Application Model (AWS SAM)?

AWS Serverless Application Model (AWS SAM) è un framework open source per la creazione di applicazioni serverless utilizzando l'infrastruttura come codice (IaC). Con AWS SAM la sua sintassi abbreviata, gli sviluppatori dichiarano CloudFormationrisorse e risorse serverless specializzate che vengono trasformate in infrastruttura durante l'implementazione. Quando lavori con AWS SAM, interagirai con:

1. AWS SAM CLI- Uno strumento da riga di comando che consente di sviluppare, testare localmente e distribuire applicazioni serverless.
2. AWS SAM Modello: un'estensione CloudFormation che fornisce una sintassi semplificata per la definizione di risorse serverless.

Quando si utilizza il sam init comando, viene creata una directory di progetto, denominata AWS SAM progetto, che in genere include il AWS SAM modello, il codice dell'applicazione e altri file di configurazione.

Quando usare AWS SAM

AWS SAM è una soluzione IaC ideale per scenari in cui si desidera uno sviluppo serverless semplificato con tutta la potenza di. CloudFormation Ad esempio, è possibile utilizzare SAM per:

- Applicazioni serverless: puoi utilizzare SAM per definire rapidamente AWS Lambda funzioni, funzioni durevoli Lambda, Amazon API APIs Gateway, tabelle Amazon DynamoDB e altre risorse serverless con codice minimo.
- CloudFormation miglioramento: puoi combinare SAM con CloudFormation modelli esistenti per aggiungere componenti serverless all'infrastruttura tradizionale. Le risorse SAM funzionano insieme alle CloudFormation risorse standard nello stesso modello.
- Sviluppo e test locali: puoi utilizzare SAM CLI per testare le funzioni Lambda localmente, simulare gli endpoint API Gateway ed eseguire il debug di applicazioni serverless sulla tua macchina di sviluppo prima della distribuzione su. AWS
- CI/CD per sistemi serverless: è possibile creare pipeline di distribuzione utilizzando modelli SAM che generano automaticamente l'infrastruttura necessaria per gli ambienti di staging e produzione. CloudFormation

- Migrazione da risorse create dalla console: è possibile convertire le funzioni Lambda e le risorse API Gateway create nella console di AWS gestione in infrastruttura come codice utilizzando modelli SAM.

Confronto AWS SAM con altri strumenti IAc

- Utilizza SAM invece di CloudFormation semplificare le definizioni delle risorse serverless mantenendo la compatibilità dei modelli.
- Utilizza SAM anziché AWS CDK se preferisci un approccio dichiarativo alla descrizione dell'infrastruttura piuttosto che uno programmatico.
- Combina SAM con le funzionalità AWS CDK di test locali di SAM CLI per migliorare le tue applicazioni CDK.

Funzionalità principali

AWS SAM offre una serie di vantaggi che migliorano l'esperienza degli sviluppatori consentendoti di:

Definite rapidamente il codice dell'infrastruttura applicativa, utilizzando meno codice

Crea AWS SAM modelli per definire il codice dell'infrastruttura applicativa serverless. Implementa i tuoi modelli direttamente per CloudFormation fornire le tue risorse.

Gestisci le tue applicazioni serverless durante l'intero ciclo di sviluppo

Utilizzalo AWS SAMCLI per gestire la tua applicazione serverless durante le fasi di creazione, creazione, distribuzione, test e monitoraggio del ciclo di vita dello sviluppo. Per ulteriori informazioni, consulta AWS SAM CLI.

Fornisci rapidamente le autorizzazioni tra le risorse con i connettori AWS SAM

Usa i AWS SAM connettori nei tuoi AWS SAM modelli per definire le autorizzazioni tra le tue AWS risorse. AWS SAM trasforma il tuo codice nelle autorizzazioni IAM necessarie per facilitare il tuo intento. Per ulteriori informazioni, consulta Gestione delle autorizzazioni delle risorse con i connettori AWS SAM.

Sincronizza continuamente le modifiche locali nel cloud durante lo sviluppo

Usa il AWS SAMCLI sam sync comando per sincronizzare automaticamente le modifiche locali nel cloud, velocizzando i flussi di lavoro di sviluppo e test sul cloud. Per ulteriori informazioni, consulta Introduzione all'uso sam sync della sincronizzazione con Cloud AWS.

Gestisci le tue applicazioni serverless Terraform

Usa il AWS SAMCLI per eseguire il debug e il test locali delle funzioni e dei livelli Lambda. Per ulteriori informazioni, consulta AWS SAMCLITerraformsupporto.

Informazioni correlate

- Per informazioni su come funziona, consulta AWS SAM . Come funziona AWS SAM
- Per iniziare a usare AWS SAM, consultaGuida introduttiva con AWS SAM.
- Per una panoramica su come AWS SAM creare un'applicazione serverless, consultaCome usare AWS SAM.

Come funziona AWS SAM

Quando lavori con AWS SAM to per creare un'applicazione serverless, interagirai con i seguenti componenti:

1. AWS SAM modello— Un file importante che definisce le AWS risorse. Questo modello include la specifica del AWS SAM modello, il framework open source dotato di una sintassi abbreviata semplificata che consente di definire le funzioni, gli eventi APIs, le configurazioni e le autorizzazioni dell'applicazione serverless. Questo file si trova nel AWS SAM progetto, che è la cartella dell'applicazione che viene creata quando si esegue il comando. sam init
2. AWS SAM CLI— Uno strumento da riga di comando che puoi utilizzare con il tuo AWS SAM progetto e che supporta integrazioni di terze parti per creare ed eseguire applicazioni serverless. AWS SAMCLIÈ lo strumento che usi per eseguire comandi sul tuo AWS SAM progetto e alla fine trasformarlo in un'applicazione serverless.

Per esprimere risorse, mappature dei sorgenti degli eventi e altre proprietà che definiscono l'applicazione serverless, è necessario definire le risorse e sviluppare l'applicazione nel AWS SAM modello e in altri file del progetto. AWS SAM È possibile utilizzare i comandi AWS SAMCLI per eseguire i comandi sul AWS SAM progetto, ovvero inizializzare, creare, testare e distribuire l'applicazione serverless.

Sei nuovo nel mondo serverless?

Ti consigliamo di recensireConcetti serverless per AWS Serverless Application Model.

Quali sono le specifiche del AWS SAM modello?

La specifica del AWS SAM modello è un framework open source che è possibile utilizzare per definire e gestire il codice dell'infrastruttura applicativa serverless. La specifica del AWS SAM modello è:

- Integrato AWS CloudFormation: puoi utilizzare la CloudFormation sintassi direttamente nel tuo AWS SAM modello, sfruttando il suo ampio supporto per le configurazioni di risorse e proprietà. Se lo conoscete già CloudFormation, non è necessario imparare a usare un nuovo servizio per gestire il codice dell'infrastruttura delle applicazioni.
- Un'estensione di CloudFormation: AWS SAM offre una sintassi unica che si concentra specificamente sull'accelerazione dello sviluppo serverless. È possibile utilizzare sia la CloudFormation AWS SAM sintassi che all'interno dello stesso modello.
- Una sintassi astratta e abbreviata: utilizzando la AWS SAM sintassi, è possibile definire l'infrastruttura rapidamente, con un minor numero di righe di codice e con una minore possibilità di errori. La sua sintassi è particolarmente curata per eliminare la complessità della definizione dell'infrastruttura applicativa serverless.
- Trasformazionale: AWS SAM esegue il complesso lavoro di trasformazione del modello nel codice necessario per il provisioning dell'infrastruttura. CloudFormation

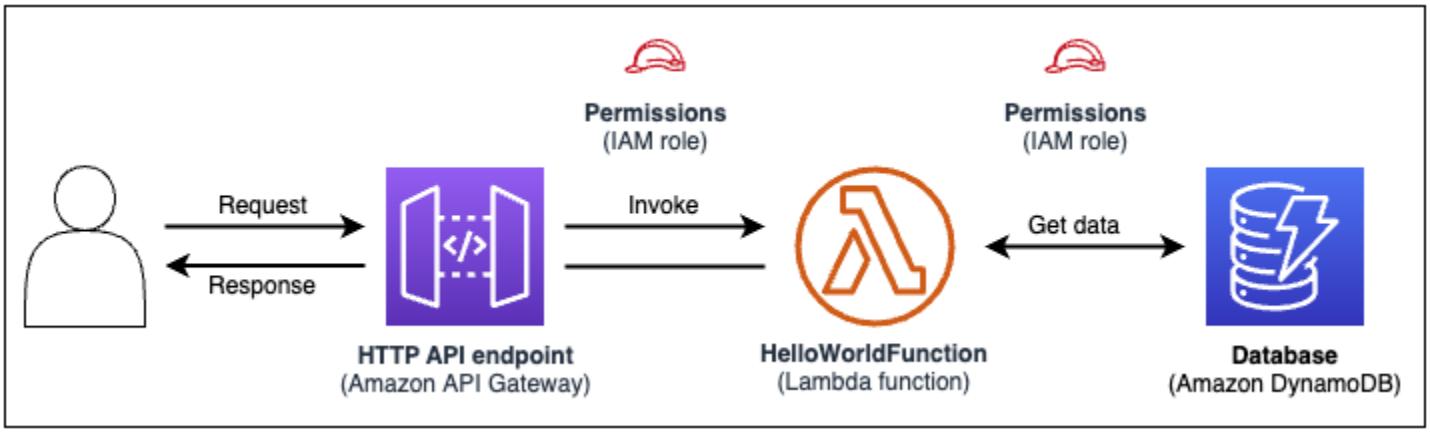
Cosa sono il AWS SAM progetto e il modello? AWS SAM

Il AWS SAM progetto include il AWS SAM modello che contiene le specifiche del AWS SAM modello. Questa specifica è il framework open source utilizzato per definire l'infrastruttura applicativa serverless AWS, con alcuni componenti aggiuntivi che ne semplificano l'utilizzo. In questo senso, i AWS SAM modelli sono un'estensione dei CloudFormation modelli.

Ecco un esempio di applicazione serverless di base. Questa applicazione elabora le richieste per ottenere tutti gli elementi da un database tramite una richiesta HTTP. Consiste delle seguenti parti:

1. Una funzione che contiene la logica per elaborare la richiesta.
2. Un'API HTTP che funge da comunicazione tra il client (richiedente) e l'applicazione.

3. Un database per memorizzare gli articoli.
4. Autorizzazioni per l'esecuzione sicura dell'applicazione.



Il codice di infrastruttura di questa applicazione può essere definito nel seguente AWS SAM modello:

```
AWSTemplateFormatVersion: 2010-09-09
Transform: AWS::Serverless-2016-10-31
Resources:
  getAllItemsFunction:
    Type: AWS::Serverless::Function
    Properties:
      Handler: src/get-all-items.getAllItemsHandler
      Runtime: nodejs20.x
      Events:
        Api:
          Type: HttpApi
          Properties:
            Path: /
            Method: GET
    Connectors:
      MyConn:
        Properties:
          Destination:
            Id: SampleTable
          Permissions:
            - Read
  SampleTable:
    Type: AWS::Serverless::SimpleTable
```

In 23 righe di codice, viene definita la seguente infrastruttura:

- Una funzione che utilizza il AWS Lambda servizio.
- Un'API HTTP che utilizza il servizio Amazon API Gateway.
- Un database che utilizza il servizio Amazon DynamoDB.
- Le autorizzazioni AWS Identity and Access Management (IAM) necessarie per consentire a questi servizi di interagire tra loro.

Per effettuare il provisioning di questa infrastruttura, il modello viene distribuito su. CloudFormation Durante la distribuzione, AWS SAM trasforma le 23 righe di codice nella CloudFormation sintassi richiesta per generare queste risorse. AWS Il CloudFormation modello trasformato contiene oltre 200 righe di codice!

CloudFormation Modello trasformato

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "getAllItemsFunction": {
      "Type": "AWS::Lambda::Function",
      "Metadata": {
        "SamResourceId": "getAllItemsFunction"
      },
      "Properties": {
        "Code": {
          "S3Bucket": "amzn-s3-demo-source-bucket-1a4x26zbcdkqr",
          "S3Key": "what-is-app/a6f856abf1b2c4f7488c09b367540b5b"
        },
        "Handler": "src/get-all-items.getAllItemsHandler",
        "Role": {
          "Fn::GetAtt": [
            "getAllItemsFunctionRole",
            "Arn"
          ]
        },
        "Runtime": "nodejs12.x",
        "Tags": [
          {
            "Key": "lambda:createdBy",
            "Value": "SAM"
          }
        ]
      }
```

```
    },
    "getAllItemsFunctionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Action": [
                "sts:AssumeRole"
              ],
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  "lambda.amazonaws.com"
                ]
              }
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole"
        ],
        "Tags": [
          {
            "Key": "lambda:createdBy",
            "Value": "SAM"
          }
        ]
      }
    },
    "getAllItemsFunctionApiPermission": {
      "Type": "AWS::Lambda::Permission",
      "Properties": {
        "Action": "lambda:InvokeFunction",
        "FunctionName": {
          "Ref": "getAllItemsFunction"
        },
        "Principal": "apigateway.amazonaws.com",
        "SourceArn": {
          "Fn::Sub": [
            "arn:${AWS::Partition}:execute-api:${AWS::Region}:${AWS::AccountId}:
${__ApiId__}/${__Stage__}/GET/",
            {
```

```
              "__ApiId__": {
                "Ref": "ServerlessHttpApi"
              },
              "__Stage__": "*"
            }
          ]
        }
      }
    },
    "ServerlessHttpApi": {
      "Type": "AWS::ApiGatewayV2::Api",
      "Properties": {
        "Body": {
          "info": {
            "version": "1.0",
            "title": {
              "Ref": "AWS::StackName"
            }
          },
          "paths": {
            "/": {
              "get": {
                "x-amazon-apigateway-integration": {
                  "httpMethod": "POST",
                  "type": "aws_proxy",
                  "uri": {
                    "Fn::Sub": "arn:${AWS::Partition}:apigateway:
${AWS::Region}:lambda:path/2015-03-31/functions/${getAllItemsFunction.Arn}/invocations"
                  },
                  "payloadFormatVersion": "2.0"
                },
                "responses": {}
              }
            }
          },
          "openapi": "3.0.1",
          "tags": [
            {
              "name": "httpapi:createdBy",
              "x-amazon-apigateway-tag-value": "SAM"
            }
          ]
        }
      }
```

```
    },
    "ServerlessHttpApiApiGatewayDefaultStage": {
      "Type": "AWS::ApiGatewayV2::Stage",
      "Properties": {
        "ApiId": {
          "Ref": "ServerlessHttpApi"
        },
        "StageName": "$default",
        "Tags": {
          "httpapi:createdBy": "SAM"
        },
        "AutoDeploy": true
      }
    },
    "SampleTable": {
      "Type": "AWS::DynamoDB::Table",
      "Metadata": {
        "SamResourceId": "SampleTable"
      },
      "Properties": {
        "AttributeDefinitions": [
          {
            "AttributeName": "id",
            "AttributeType": "S"
          }
        ],
        "KeySchema": [
          {
            "AttributeName": "id",
            "KeyType": "HASH"
          }
        ],
        "BillingMode": "PAY_PER_REQUEST"
      }
    },
    "getAllItemsFunctionMyConnPolicy": {
      "Type": "AWS::IAM::ManagedPolicy",
      "Metadata": {
        "aws:sam:connectors": {
          "getAllItemsFunctionMyConn": {
            "Source": {
              "Type": "AWS::Serverless::Function"
            },
            "Destination": {
```

```
              "Type": "AWS::Serverless::SimpleTable"
            }
          }
        }
      },
      "Properties": {
        "PolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Action": [
                "dynamodb:GetItem",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchGetItem",
                "dynamodb:ConditionCheckItem",
                "dynamodb:PartiQLSelect"
              ],
              "Resource": [
                {
                  "Fn::GetAtt": [
                    "SampleTable",
                    "Arn"
                  ]
                },
                {
                  "Fn::Sub": [
                    "${DestinationArn}/index/*",
                    {
                      "DestinationArn": {
                        "Fn::GetAtt": [
                          "SampleTable",
                          "Arn"
                        ]
                      }
                    }
                  ]
                }
              ]
            }
          ]
        },
        "Roles": [
```

```
          {
            "Ref": "getAllItemsFunctionRole"
          }
        ]
      }
    }
  }
}
```

Utilizzando AWS SAM, si definiscono 23 righe di codice dell'infrastruttura. AWS SAM trasforma il codice nelle oltre 200 righe di CloudFormation codice necessarie per il provisioning dell'applicazione.

Che cos'è il? AWS SAMCLI

AWS SAMCLIÈ uno strumento a riga di comando che puoi utilizzare con AWS SAM modelli e integrazioni di terze parti supportate per creare ed eseguire applicazioni serverless. Usa il per: AWS SAMCLI

- Inizializza rapidamente un nuovo progetto applicativo.
- Crea la tua applicazione per la distribuzione.
- Esegui il debug e il test locali.
- Distribuisci l'applicazione.
- Configura le pipeline CI/CD di distribuzione.
- Monitora e risolvi i problemi della tua applicazione nel cloud.
- Sincronizza le modifiche locali sul cloud durante lo sviluppo.
- E molto altro ancora!

AWS SAMCLIViene utilizzato al meglio se utilizzato con CloudFormation modelli AWS SAM e. Funziona anche con prodotti di terze parti comeTerraform.

Inizializza un nuovo progetto

Seleziona tra i modelli iniziali o scegli una posizione personalizzata per i modelli per iniziare un nuovo progetto.

Qui, usiamo il sam init comando per inizializzare un nuovo progetto applicativo. Selezioniamo il progetto Hello World Example per iniziare. AWS SAMCLIScarica un modello iniziale e crea la struttura di directory delle cartelle del nostro progetto.

```
→  what-is sam init

You can preselect a particular runtime or package type when using the `sam init` experience.
Call `sam init --help` to learn more.

Which template source would you like to use?
        1 - AWS Quick Start Templates
        2 - Custom Template Location
Choice: 1

Choose an AWS Quick Start application template
        1 - Hello World Example
        2 - Multi-step workflow
        3 - Serverless API
        4 - Scheduled task
        5 - Standalone function
        6 - Data processing
        7 - Infrastructure event management
        8 - Serverless Connector Hello World Example
        9 - Multi-step workflow with Connectors
        10 - Lambda EFS example
        11 - Machine Learning
Template: 1

Use the most popular runtime and package type? (Python and zip) [y/N]:
```

Per ulteriori dettagli, consultare Crea la tua candidatura in AWS SAM.

Crea la tua applicazione per la distribuzione

Package delle dipendenze delle funzioni e organizza il codice del progetto e la struttura delle cartelle per prepararti alla distribuzione.

Qui, utilizziamo il sam build comando per preparare la nostra applicazione per la distribuzione. AWS SAMCLICrea una `.aws-sam` directory e organizza le dipendenze delle nostre applicazioni e i file per la distribuzione.

```
➜  sam-app sam build
Building codeuri: /Users/evzz/Demo/what-is/sam-app/hello_world runtime: python3.9 metadata: {} archit
ecture: x86_64 functions: HelloWorldFunction
Running PythonPipBuilder:ResolveDependencies
Running PythonPipBuilder:CopySource

Build Succeeded

Built Artifacts  : .aws-sam/build
Built Template   : .aws-sam/build/template.yaml

Commands you can use next
=========================
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name {{stack-name}} --watch
[*] Deploy: sam deploy --guided
➜  sam-app cd .aws-sam
➜  .aws-sam ls
build      build.toml
➜  .aws-sam
```

Per ulteriori dettagli, consultare Crea la tua applicazione.

Esegui il debug e il test locali

Sul computer locale, simula eventi, test APIs, richiama funzioni e altro ancora per eseguire il debug e testare l'applicazione.

Qui, usiamo il sam local invoke comando per richiamare il nostro localmente. `HelloWorldFunction` Per fare ciò, AWS SAMCLI crea un contenitore locale, crea la nostra funzione, la richiama e restituisce i risultati. Puoi usare un'applicazione come Docker per eseguire contenitori sulla tua macchina.

```
➜  sam-app sam local invoke HelloWorldFunction
Invoking app.lambda_handler (python3.9)
Local image was not found.
Removing rapid images for repo public.ecr.aws/sam/emulation-python3.9
Building image.....................................................................................
.....................................................................................................
.............................................................
Using local image: public.ecr.aws/lambda/python:3.9-rapid-x86_64.

Mounting /Users/evzz/Demo/what-is/sam-app/.aws-sam/build/HelloWorldFunction as /var/task:ro,delegated
 inside runtime container
START RequestId: 6f8347ce-6b04-4246-a0de-6dc37f0eef51 Version: $LATEST
END RequestId: 6f8347ce-6b04-4246-a0de-6dc37f0eef51
REPORT RequestId: 6f8347ce-6b04-4246-a0de-6dc37f0eef51  Init Duration: 1.23 ms  Duration: 639.26 ms B
illed Duration: 640 ms  Memory Size: 128 MB     Max Memory Used: 128 MB
{"statusCode": 200, "body": "{\"message\": \"hello world\"}"}
```

Per ulteriori dettagli, consulta Testa la tua applicazione e Esegui il debug della tua applicazione.

Distribuzione dell'applicazione

Configura le impostazioni di distribuzione dell'applicazione e distribuiscila nel AWS cloud per fornire le tue risorse.

Qui, utilizziamo il sam deploy --guided comando per distribuire la nostra applicazione tramite un flusso interattivo. Ci AWS SAMCLI guida nella configurazione delle impostazioni di distribuzione della nostra applicazione, trasforma il nostro modello in CloudFormation e lo CloudFormation implementa per creare le nostre risorse.

```
➜  sam-app sam deploy --guided

Configuring SAM deploy
======================

        Looking for config file [samconfig.toml] :  Not found

        Setting default arguments for 'sam deploy'
        =========================================
        Stack Name [sam-app]:
        AWS Region [us-west-2]:
        #Shows you resources changes to be deployed and require a 'Y' to initiate deploy
        Confirm changes before deploy [y/N]:
        #SAM needs permission to be able to create roles to connect to the resources in your template
        Allow SAM CLI IAM role creation [Y/n]:
        #Preserves the state of previously provisioned resources when an operation fails
        Disable rollback [y/N]:
        HelloWorldFunction may not have authorization defined, Is this okay? [y/N]: y
        Save arguments to configuration file [Y/n]:
        SAM configuration file [samconfig.toml]:
        SAM configuration environment [default]:

        Looking for resources needed for deployment:
         Managed S3 bucket: aws-sam-cli-managed-default-samclisourcebucket-1a4x26zbcdkqr
         A different default S3 bucket can be set in samconfig.toml
```

Per ulteriori dettagli, consultare Implementa la tua applicazione e le tue risorse.

Configura le pipeline CI/CD di distribuzione

Crea pipeline sicure di integrazione e distribuzione continue (CI/CD) utilizzando un sistema supportato. CI/CD

Qui, utilizziamo il sam pipeline init --bootstrap comando per configurare una pipeline di distribuzione CI/CD per la nostra applicazione. Ci AWS SAMCLI guida attraverso le nostre opzioni e genera le AWS risorse e il file di configurazione da utilizzare con il nostro sistema CI/CD.

```
[3] Reference application build resources
Enter the pipeline execution role ARN if you have previously created one, or we will create one for y
ou []:
Enter the CloudFormation execution role ARN if you have previously created one, or we will create one
 for you []:
Please enter the artifact bucket ARN for your Lambda function. If you do not have a bucket, we will c
reate one for you []:
Does your application contain any IMAGE type Lambda functions? [y/N]: n

[4] Summary
Below is the summary of the answers:
        1 - Account: 513423067560
        2 - Stage configuration name: dev
        3 - Region: us-west-2
        4 - Pipeline user: [to be created]
        5 - Pipeline execution role: [to be created]
        6 - CloudFormation execution role: [to be created]
        7 - Artifacts bucket: [to be created]
        8 - ECR image repository: [skipped]
Press enter to confirm the values above, or select an item to edit the value:

This will create the following required resources for the 'dev' configuration:
        - Pipeline IAM user
        - Pipeline execution role
        - CloudFormation execution role
        - Artifact bucket
Should we proceed with the creation? [y/N]:
```

Per ulteriori dettagli, consultare Implementa con CI/CD sistemi e pipeline.

Monitora e risolvi i problemi della tua applicazione nel cloud

Visualizza informazioni importanti sulle risorse distribuite, raccogli i log e utilizza strumenti di monitoraggio integrati come. AWS X-Ray

Qui, utilizziamo il sam list comando per visualizzare le nostre risorse distribuite. Otteniamo il nostro endpoint API e lo invochiamo, il che attiva la nostra funzione. Quindi, utilizziamo sam logs per visualizzare i log della nostra funzione.

```
➜  sam-app sam logs --stack-name sam-app
2023/03/13/[$LATEST]0a433e844dd445bd82d0d78cd55e0cdc 2023-03-13T21:06:42.075000 INIT_START Runtime Ve
rsion: python:3.9.v16   Runtime Version ARN: arn:aws:lambda:us-west-2::runtime:07a48df201798d627f2b95
0f03bb227aab4a655a1d019c3296406f95937e2525
2023/03/13/[$LATEST]0a433e844dd445bd82d0d78cd55e0cdc 2023-03-13T21:06:42.180000 START RequestId: 778e
4226-0a80-435f-929b-5b19292ed9a7 Version: $LATEST
2023/03/13/[$LATEST]0a433e844dd445bd82d0d78cd55e0cdc 2023-03-13T21:06:42.181000 END RequestId: 778e42
26-0a80-435f-929b-5b19292ed9a7
2023/03/13/[$LATEST]0a433e844dd445bd82d0d78cd55e0cdc 2023-03-13T21:06:42.182000 REPORT RequestId: 778
e4226-0a80-435f-929b-5b19292ed9a7       Duration: 1.69 ms       Billed Duration: 2 ms   Memory Size:
128 MB  Max Memory Used: 36 MB  Init Duration: 104.13 ms
```

Per ulteriori dettagli, consultare Monitora la tua applicazione.

Sincronizza le modifiche locali sul cloud durante lo sviluppo

Mentre sviluppi sul tuo computer locale, sincronizza automaticamente le modifiche nel cloud. Visualizza rapidamente le modifiche ed esegui test e convalide nel cloud.

Qui, usiamo il sam sync --watch comando per AWS SAMCLI controllare le modifiche locali. Modifichiamo il nostro `HelloWorldFunction` codice e rileva AWS SAMCLI automaticamente la modifica e distribuisce i nostri aggiornamenti nel cloud.

```
-----------------------------------------------------------------------------------------------
Key                 HelloWorldFunctionIamRole
Description         Implicit IAM Role created for Hello World function
Value               arn:aws:iam::513423067560:role/sam-app-HelloWorldFunctionRole-15GLOUR9LMT1W

Key                 HelloWorldApi
Description         API Gateway endpoint URL for Prod stage for Hello World function
Value               https://ets1gv8lxi.execute-api.us-west-2.amazonaws.com/Prod/hello/

Key                 HelloWorldFunction
Description         Hello World Lambda Function ARN
Value               arn:aws:lambda:us-west-2:513423067560:function:sam-app-HelloWorldFunction-
yQDNe17r9maD
-----------------------------------------------------------------------------------------------

Stack update succeeded. Sync infra completed.

Infra sync completed.
CodeTrigger not created as CodeUri or DefinitionUri is missing for ServerlessRestApi.
Syncing Lambda Function HelloWorldFunction...
Manifest is not changed for (HelloWorldFunction), running incremental build
Building codeuri: /Users/evzz/Demo/what-is/sam-app/hello_world runtime: python3.9 metadata: {} archit
ecture: x86_64 functions: HelloWorldFunction
Running PythonPipBuilder:CopySource
Finished syncing Lambda Function HelloWorldFunction.
```

Prova le risorse supportate nel cloud

Richiama e trasferisci eventi alle risorse supportate nel cloud.

Qui, utilizziamo il sam remote invoke comando per testare una funzione Lambda distribuita nel cloud. Invochiamo la nostra funzione Lambda e ne riceviamo i log e la risposta. Con la nostra funzione Lambda configurata per lo streaming delle risposte, quest' AWS SAMCLIultima invia la risposta in tempo reale.

```
→ lambda-streaming-app sam remote invoke StreamingFunction --stack-name lambda-streaming-app
```

Ulteriori informazioni

Per continuare a saperne di più AWS SAM, consulta le seguenti risorse:

- Il AWS SAM workshop completo: un workshop progettato per insegnarti molte delle principali funzionalità che AWS SAM offre.
- Sessioni con SAM — Serie di video creata dal nostro team di AWS Serverless Developer Advocate sull'utilizzo. AWS SAM
- Serverless Land: sito che riunisce le informazioni, i blog, i video, il codice e le risorse di apprendimento più recenti per la tecnologia serverless. AWS

Fasi successive

Se è la prima volta che lo usi AWS SAM, vedi. Guida introduttiva con AWS SAM

Concetti serverless per AWS Serverless Application Model

Scopri i concetti base del serverless prima di usare (). AWS Serverless Application Model AWS SAM

Concetti serverless

Architettura basata su eventi

Un'applicazione serverless è costituita da singoli AWS servizi, ad esempio AWS Lambda per l'elaborazione e Amazon DynamoDB per la gestione dei database, che svolgono ciascuno un ruolo specializzato. Questi servizi sono quindi liberamente integrati tra loro tramite un'architettura basata sugli eventi. Per ulteriori informazioni sull'architettura basata sugli eventi, vedi Cos'è un'architettura basata sugli eventi? .

Infrastruttura come codice (IaC)

Infrastructure as Code (IaC) è un modo di trattare l'infrastruttura nello stesso modo in cui gli sviluppatori trattano il codice, applicando lo stesso rigore dello sviluppo del codice applicativo al provisioning dell'infrastruttura. Definisci la tua infrastruttura in un file modello, la distribuisci e AWS crea le AWS risorse per te. Con IAc, definisci in codice ciò che desideri AWS fornire. Per un confronto degli strumenti IaC disponibili per AWS, vedere. Infrastruttura come codice (IaC)

Tecnologie serverless

Con le tecnologie AWS serverless, puoi creare ed eseguire applicazioni senza dover gestire i tuoi server. Tutta la gestione dei server viene effettuata da AWS noi e offre numerosi vantaggi, come la scalabilità automatica e l'elevata disponibilità integrata, che consentono di portare rapidamente la propria idea alla produzione. Utilizzando le tecnologie serverless, puoi concentrarti sul core del tuo prodotto senza doverti preoccupare della gestione e del funzionamento dei server. Per ulteriori informazioni sulla tecnologia serverless, consulta quanto segue:

- Serverless attivo AWS
- Guida per sviluppatori serverless: fornisce una panoramica concettuale dello sviluppo serverless nel cloud. AWS

Per un'introduzione di base ai principali servizi AWS serverless, consulta Serverless 101: Understanding the serverless services at Serverless Land.

Applicazione serverless

Quando si utilizza AWS SAM, si gestiscono le risorse correlate in un'applicazione, che consiste nel AWS SAM progetto e nel modello. Tutte le risorse dell'applicazione sono definite o a cui si fa riferimento nel AWS SAM modello. Quando AWS SAM elabora il modello, crea CloudFormation risorse. In CloudFormation, le risorse vengono gestite in una singola unità chiamata stack e tutte le risorse in uno stack sono definite dal modello dello stack. CloudFormation

Infrastruttura come codice (IaC)

Con Infrastructure as Code (IaC), puoi automatizzare l'implementazione e la gestione delle tue AWS risorse, comprese le applicazioni serverless. IaC consente di definire l'infrastruttura utilizzando il codice, semplificando la versione, la condivisione e la replica delle distribuzioni. Questo approccio ti aiuta a:

- Accelera il ciclo di sviluppo
- Semplifica la gestione della configurazione
- Migliora l'affidabilità e la coerenza delle tue implementazioni

Strumenti IaC per applicazioni serverless AWS

AWS offre diversi strumenti IaC per aiutarti a creare, implementare e gestire le tue risorse cloud. Questa sezione spiega come AWS SAM si inserisce in questo ecosistema e come funziona con altri strumenti AWS IaC.

AWS CloudFormation

Utilizzando CloudFormation, puoi modellare e fornire l'intera AWS infrastruttura con modelli YAML o JSON. CloudFormation gestisce automaticamente la creazione, gli aggiornamenti e l'eliminazione delle risorse. Quando distribuisci AWS SAM le applicazioni, CloudFormation elabora i modelli trasformati per creare e gestire le tue risorse.

AWS Serverless Application Model (AWS SAM)

AWS SAM consente di creare applicazioni serverless con una sintassi semplificata per la definizione di risorse serverless. È possibile utilizzare AWS SAM modelli per eseguire il provisioning di funzioni Lambda APIs, database e sorgenti di eventi utilizzando una sintassi YAML concisa. AWS SAM trasforma questi modelli in modelli durante la distribuzione. CloudFormation

> **Note**
>
> Sebbene sia AWS SAM specializzato in applicazioni serverless, puoi utilizzare qualsiasi tipo di CloudFormation risorsa nei tuoi modelli. AWS SAM Ciò offre la flessibilità necessaria per includere risorse non serverless quando necessario.

AWS Cloud Development Kit (AWS CDK)

Con AWS CDK, puoi definire la tua infrastruttura serverless utilizzando linguaggi di programmazione familiari come Python TypeScript, Java, C#/.Net o Go. È possibile utilizzare costrutti di programmazione come loop e condizioni per definire l'infrastruttura e generare modelli per l'implementazione. AWS CDK CloudFormation È possibile utilizzare il AWS SAMCLI per testare ed eseguire il debug localmente delle applicazioni create con. AWS CDK Per ulteriori informazioni, consulta Testare localmente le applicazioni CDK.

Confronto degli strumenti IaC per applicazioni serverless

Quando scegli uno strumento IaC per le tue applicazioni serverless, considera le preferenze del tuo team, i requisiti del progetto e i flussi di lavoro esistenti. La tabella seguente confronta le caratteristiche chiave degli strumenti AWS IaC per lo sviluppo senza server:

Strumento	Uso primario	Ideale per	Funziona con AWS SAM	Quando scegliere
CloudFormation	Gestire AWS un'infrastruttura complessa	Applicazioni che richiedono un controllo dettagliato delle AWS risorse	AWS SAM i modelli si trasformano in CloudFormation modelli durante la distribuzione	Per un controllo granulare delle risorse non serverless
AWS SAM	Sviluppo di applicazioni serverless	Team che creano applicazioni serverless utilizzando Lambda	Funzionalità nativa	Quando ci si concentra principalmente su architetture serverless con funzioni Lambda, API Gateway e altre risorse serverless APIs
AWS CDK	Definizione dell'infrastruttura utilizzando	Team che preferiscono i linguaggi tipizzati	Genera AWS SAM modelli e	Quando è necessaria la definizione di

Strumento	Uso primario	Ideale per	Funziona con AWS SAM	Quando scegliere
	linguaggi di programmazione	e un approccio basato sul codice	usali per i test AWS SAMCLI	un'infrastruttura programmatica o una logica di configurazione delle risorse complessa

> **Note**
>
> Sebbene questa guida si concentri sugli strumenti IaC AWS nativi, Terraform è un'altra popolare soluzione IaC che può essere utilizzata per definire applicazioni serverless. AWS SAMCLISupporta il test locale delle funzioni Lambda definite in Terraform. Per ulteriori informazioni, consulta AWS SAMCLITerraformsupporto.

Ulteriori informazioni

- Per saperne di più sulle DevOps pratiche su AWS, vedi Introduzione a on DevOps AWS
- Per informazioni sull'utilizzo di Lambda con diversi strumenti IAc, consulta Using Lambda with infrastructure as code (IaC)

Guida introduttiva con AWS SAM

Inizia AWS SAM esaminando e completando gli argomenti di questa sezione. AWS SAM prerequisitifornisce istruzioni dettagliate sulla configurazione di un AWS account, la creazione di utenti IAM, la creazione dell'accesso tramite chiave e l'installazione e la AWS SAMCLI configurazione di. Dopo aver completato i prerequisiti, puoi scegliere il tuo approccio di sviluppo:

- Installa il AWS SAMCLI, che puoi eseguire su sistemi operativi Linux, Windows e macOS.
- Guida introduttiva a AWS SAM utilizzando VS Code Toolkit per un'esperienza di sviluppo visivo con funzionalità di debug e test integrate.

Una volta completata l'installazione, puoi facoltativamente seguire il tutorial Hello World. AWS SAM Questo tutorial ti guiderà attraverso il processo di creazione di un'applicazione serverless di base con. AWS SAM Dopo aver completato il tutorial, sarai pronto per rivedere i concetti descritti in dettaglio inCome usare AWS Serverless Application Model (AWS SAM).

Argomenti

- AWS SAM prerequisiti
- Installa il AWS SAMCLI
- Tutorial: Implementa un'applicazione Hello World con AWS SAM
- Conversione delle funzioni Lambda in applicazioni AWS SAM

AWS SAM prerequisiti

Completate i seguenti prerequisiti prima di installare e utilizzare l'interfaccia a riga di AWS Serverless Application Model comando ()AWS SAMCLI.

Per utilizzare AWS SAMCLI, è necessario quanto segue:

- Un AWS account, credenziali AWS Identity and Access Management (IAM) e una coppia di key di accesso IAM.
- Il AWS Command Line Interface (AWS CLI) per configurare le AWS credenziali.

Argomenti

- Passaggio 1: crea un account AWS
- Fase 2: Creare un account utente IAM
- Passaggio 3: Crea un ID della chiave di accesso e una chiave di accesso segreta
- Fase 4: Installare il AWS CLI
- Passaggio 5: Utilizzare AWS CLI per configurare le AWS credenziali
- Passaggio 6: (Opzionale) Installazione di AWS Toolkit for VS Code
- Fasi successive

Passaggio 1: crea un account AWS

Se non ne hai uno Account AWS, completa i seguenti passaggi per crearne uno.

Per iscriverti a un Account AWS

1. Apri la https://portal.aws.amazon.com/billing/registrazione.
2. Segui le istruzioni online.

   Nel corso della procedura di registrazione riceverai una telefonata o un messaggio di testo e ti verrà chiesto di inserire un codice di verifica attraverso la tastiera del telefono.

   Quando ti iscrivi a un Account AWS, Utente root dell'account AWSviene creato un. L'utente root dispone dell'accesso a tutte le risorse e tutti i Servizi AWS nell'account. Come best practice di sicurezza, assegna l'accesso amministrativo a un utente e utilizza solo l'utente root per eseguire attività che richiedono l'accesso di un utente root.

Fase 2: Creare un account utente IAM

Per creare un utente amministratore, scegli una delle seguenti opzioni.

Scelta di un modo per gestire il tuo amministratore	Per	Come	Puoi anche
In IAM Identity Center (Consigliato)	Usa credenziali a breve termine per accedere a AWS. Ciò è in linea con le best practice per la sicurezza. Per informazioni sulle best practice, consulta Best practice per la sicurezza in IAM nella Guida per l'utente di IAM.	Segui le istruzioni riportate in Nozioni di base nella Guida per l'utente di AWS IAM Identity Center .	Configura l'accesso programmatico configurando l'uso AWS IAM Identity Center nella Guida AWS CLI per l'AWS Command Line Interface utente.
In IAM (Non consigliato)	Usa credenziali a lungo termine per accedere a AWS.	Segui le istruzioni in Creare un utente IAM per l'accesso di emergenza nella Guida per l'utente di IAM.	Configura l'accesso programmatico seguendo quanto riportato in Gestione delle chiavi di accesso per gli utenti IAM nella Guida per l'utente di IAM.

Passaggio 3: Crea un ID della chiave di accesso e una chiave di accesso segreta

Per l'accesso alla CLI, sono necessari un ID chiave di accesso e una chiave di accesso segreta. Utilizza credenziali temporanee al posto delle chiavi di accesso a lungo termine quando possibile. Le credenziali temporanee includono un ID della chiave di accesso, una chiave di accesso segreta e

un token di sicurezza che ne indica la scadenza. Per ulteriori informazioni, consulta Using temporary credenziali with AWS resources nella IAM User Guide.

Gli utenti necessitano dell'accesso programmatico se desiderano interagire con l' AWS esterno di. Console di gestione AWS Il modo per concedere l'accesso programmatico dipende dal tipo di utente che accede. AWS

Per fornire agli utenti l'accesso programmatico, scegli una delle seguenti opzioni.

Quale utente necessita dell'accesso programmatico?	Per	Come
IAM	(Consigliato) Utilizza le credenziali della console come credenziali temporanee per firmare le richieste programmatiche a,, o. AWS CLI AWS SDKs AWS APIs	Segui le istruzioni per l'interfaccia che desideri utilizzare. • Per la AWS CLI, consulta Login for AWS local development nella Guida per l'AWS Command Line Interface utente. • Per AWS SDKs, consulta Login for AWS local development nella AWS SDKs and Tools Reference Guide.
Identità della forza lavoro (Utenti gestiti nel centro identità IAM)	Utilizza credenziali temporanee per firmare le richieste programmatiche a AWS CLI, AWS SDKs, o. AWS APIs	Segui le istruzioni per l'interfaccia che desideri utilizzare. • Per la AWS CLI, vedere Configurazione dell'uso AWS IAM Identity Center nella AWS CLI Guida per l'utente.AWS Command Line Interface • Per AWS SDKs gli strumenti e AWS APIs, consulta l'autenticazione di IAM

Quale utente necessita dell'accesso programmatico?	Per	Come
		Identity Center nella Guida di riferimento AWS SDKs and Tools.
IAM	Utilizza credenziali temporane e per firmare le richieste programmatiche a AWS CLI, AWS SDKs, o. AWS APIs	Seguendo le istruzioni riportate in Utilizzo delle credenziali temporanee con le AWS risorse nella Guida per l'utente IAM.
IAM	(Non consigliato) Utilizza credenziali a lungo termine per firmare richieste programmatiche a AWS CLI,, AWS SDKs o. AWS APIs	Segui le istruzioni per l'interfa ccia che desideri utilizzare. • Per la AWS CLI, consulta Autenticazione tramite credenziali utente IAM nella Guida per l'utente.AWS Command Line Interface • Per gli strumenti AWS SDKs e gli strumenti, consulta Autenticazione tramite credenziali a lungo termine nella Guida di riferimento agli strumenti e agli AWS SDKs strumenti. • Per AWS APIs, consulta la sezione Gestione delle chiavi di accesso per gli utenti IAM nella Guida per l'utente IAM.

Fase 4: Installare il AWS CLI

AWS CLI è uno strumento open source che consente di interagire Servizi AWS utilizzando i comandi nella shell della riga di comando. Lo AWS SAMCLI richiede AWS CLI per attività come la configurazione delle credenziali. Per ulteriori informazioni su AWS CLI, consulta What is the? AWS Command Line Interface nella Guida AWS Command Line Interface per l'utente.

Per installare il AWS CLI, vedere Installazione o aggiornamento della versione più recente di AWS CLI nella Guida per l'AWS Command Line Interface utente.

Passaggio 5: Utilizzare AWS CLI per configurare le AWS credenziali

Per configurare le credenziali con IAM Identity Center

- Per configurare le credenziali con IAM Identity Center, consulta Configurare il profilo con la procedura guidata di AWS configurazione sso.

Per configurare le credenziali con AWS CLI

1. Esegui il `aws configure` comando dalla riga di comando.
2. Configura quanto segue. Seleziona ogni link per saperne di più:

   a. ID della chiave di accesso

   b. Chiave di accesso segreta

   c. Regione AWS

   d. Formato di output

   L'esempio seguente illustra i valori di esempio.

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

AWS CLI Memorizza queste informazioni in un profilo (una raccolta di impostazioni) denominato `default` nei `config` file `credentials` and. Questi file si trovano nel `.aws` file della tua home

directory. Per impostazione predefinita, le informazioni in questo profilo vengono utilizzate quando si esegue un AWS CLI comando che non specifica esplicitamente un profilo da utilizzare. Per ulteriori informazioni sul `credentials` file, consulta Impostazioni dei file di configurazione e credenziali nella Guida per l'AWS Command Line Interface utente.

Per ulteriori informazioni sulla configurazione delle credenziali, ad esempio sull'utilizzo di un file di configurazione e credenziali esistente, vedere Configurazione rapida nella Guida per l'utente.AWS Command Line Interface

Passaggio 6: (Opzionale) Installazione di AWS Toolkit for VS Code

Per gli sviluppatori che preferiscono un ambiente di sviluppo integrato, il AWS Toolkit for VS Code offre potenti funzionalità per lo sviluppo senza server, tra cui debug visivoCodeLens , integrazione e flussi di lavoro di distribuzione semplificati.

Prerequisiti per lo sviluppo di VS Code

- Visual Studio Code (versione 1.73.0 o versione successiva) installato sul sistema
- Estensione di supporto del linguaggio YAML per VS Code

Per installare il AWS Toolkit for VS Code

1. Apri Visual Studio Code
2. Apri la vista Estensioni (Ctrl+Shift+X o Cmd+Shift+X)
3. Cerca AWS "Toolkit»
4. Installa l'estensione "AWS Toolkit» di Amazon Web Services
5. Installa l'estensione «YAML» di Red Hat (richiesta per le funzionalità del modello SAM) CodeLens

Vantaggi dell'utilizzo di VS Code con SAM AWS

- Debug visivo: imposta i punti di interruzione e gestisci le funzioni Lambda a livello locale
- CodeLens integrazione: crea, distribuisci e richiama funzioni direttamente dal tuo modello SAM
- Terminale integrato: accedi ai AWS CLI comandi AWS SAM senza uscire dall'editor
- Convalida dei modelli: convalida in tempo reale e IntelliSense per modelli SAM

Per informazioni sulla configurazione delle AWS credenziali in VS Code, consulta Configurazione delle credenziali nella AWS Toolkit for VS Code User Guide.

Fasi successive

- Ora sei pronto per installare e iniziare a utilizzare. AWS SAMCLI AWS SAM Per installare il AWS SAMCLI, vedereInstalla il AWS SAMCLI.
- Per configurare Visual Studio Code per lo sviluppo senza server, vedi Configurazione del AWS Toolkit for VS Code.

Installa il AWS SAMCLI

Installa l'ultima versione di AWS Serverless Application Model Command Line Interface (AWS SAMCLI) sui sistemi operativi supportati seguendo le istruzioni inFase 4: Installare il AWS CLI.

Per informazioni sulla gestione di una versione attualmente installata di AWS SAMCLI, incluso l'aggiornamento, la disinstallazione o la gestione delle nightly build, consulta. Gestione delle AWS SAMCLI versioni

> È la prima volta che installi il? AWS SAM CLI
>
> Completa tutti i prerequisiti della sezione precedente prima di procedere. Questo include:
>
> 1. Registrazione di un AWS account.
> 2. Configurazione dell'accesso sicuro a AWS.
> 3. Creazione di un ID della chiave di accesso e di una chiave di accesso segreta.
> 4. Installazione di AWS CLI.
> 5. Configurazione delle AWS credenziali.

Argomenti

- Installazione di AWS SAMCLI
- Risoluzione degli errori di installazione
- Fasi successive
- Facoltativo: verifica l'integrità del AWS SAMCLI programma di installazione

Installazione di AWS SAMCLI

> **Note**
>
> A partire da settembre 2023, AWS non manterrà più il programma di Homebrew installazione AWS gestito per AWS SAMCLI ()`aws/tap/aws-sam-cli`. Se utilizzi Homebrew per installare e gestire il AWS SAMCLI, consulta le seguenti opzioni:
>
> - Per continuare a utilizzareHomebrew, puoi utilizzare il programma di installazione gestito dalla community. Per ulteriori informazioni, consulta Gestire il AWS SAMCLI con Homebrew.
> - Si consiglia di utilizzare uno dei metodi di installazione proprietari documentati in questa pagina. Prima di utilizzare uno di questi metodi, consultaPassa da Homebrew.
> - Per ulteriori dettagli, fare riferimento alla versione di rilascio: 1.121.0.

Per installare AWS SAMCLI, segui le istruzioni relative al tuo sistema operativo.

Linux

x86_64 - command line installer

1. Scarica il AWS SAMCLIfile.zip in una directory a tua scelta.
2. (Facoltativo) È possibile verificare l'integrità del programma di installazione prima dell'installazione. Per istruzioni, consulta Facoltativo: verifica l'integrità del AWS SAMCLI programma di installazione.
3. Decomprimi i file di installazione in una directory a tua scelta. Di seguito è riportato un esempio di utilizzo della `sam-installation` sottodirectory.

   > **Note**
   >
   > Se il sistema operativo non dispone del comando unzip integrato, utilizza un comando equivalente.

   ```
   $ unzip aws-sam-cli-linux-x86_64.zip -d sam-installation
   ```

4. Installa AWS SAMCLI eseguendo l'`install`eseguibile. Questo eseguibile si trova nella directory utilizzata nel passaggio precedente. Di seguito è riportato un esempio di utilizzo della `sam-installation` sottodirectory:

```
$ sudo ./sam-installation/install
```

5. Verifica l'installazione.

```
$ sam --version
```

Per confermare l'avvenuta installazione, dovrebbe apparire un output che sostituisce il seguente testo tra parentesi con l'ultima versione disponibile:

```
SAM CLI, <latest version>
```

arm64 - command line installer

1. Scarica il AWS SAMCLIfile.zip in una directory a tua scelta.
2. (Facoltativo) È possibile verificare l'integrità del programma di installazione prima dell'installazione. Per istruzioni, consulta Facoltativo: verifica l'integrità del AWS SAMCLI programma di installazione.
3. Decomprimi i file di installazione in una directory a tua scelta. Di seguito è riportato un esempio di utilizzo della `sam-installation` sottodirectory.

> **Note**
>
> Se il sistema operativo non dispone del comando unzip integrato, utilizza un comando equivalente.

```
$ unzip aws-sam-cli-linux-arm64.zip -d sam-installation
```

4. Installa AWS SAMCLI eseguendo l'`install`eseguibile. Questo eseguibile si trova nella directory utilizzata nel passaggio precedente. Di seguito è riportato un esempio di utilizzo della `sam-installation` sottodirectory:

```
$ sudo ./sam-installation/install
```

5. Verifica l'installazione.

```
$ sam --version
```

Per confermare l'avvenuta installazione, dovresti vedere un output come il seguente, ma che sostituisce il testo tra parentesi con l'ultima versione SAM CLI:

```
SAM CLI, <latest version>
```

macOS

Fasi di installazione

Usa il programma di installazione del pacchetto per installare. AWS SAMCLI Inoltre, il programma di installazione del pacchetto offre due metodi di installazione tra cui scegliere: GUI e riga di comando. È possibile eseguire l'installazione per tutti gli utenti o solo per l'utente corrente. Per eseguire l'installazione per tutti gli utenti, è richiesta l'autorizzazione di superutente.

> **Note**
> AWS SAM La CLI non supporta le versioni di macOS precedenti a macOS 13.x. Esegui l'upgrade a una versione supportata di macOS o installa CLI AWS SAM con. Homebrew

GUI - All users

Per scaricare il pacchetto, il programma di installazione e installare il AWS SAMCLI

> **Note**
> Se hai già installato il AWS SAMCLI Through Homebrew orpip, devi prima disinstallarlo. Per istruzioni, consulta Disinstallazione di AWS SAMCLI.

1. Scarica macOS in una directory `pkg` a tua scelta:

- Per i Mac con processori Intel, scegli x86_64 — -x86_64.pkg aws-sam-cli-macos
- Per i Mac che eseguono Apple Silicon, scegli arm64 — -arm64.pkg aws-sam-cli-macos

> **Note**
>
> È possibile verificare l'integrità del programma di installazione prima dell'installazione. Per istruzioni, consulta Facoltativo: verifica l'integrità del AWS SAMCLI programma di installazione.

2. Esegui il file scaricato e segui le istruzioni sullo schermo per continuare con i passaggi di Introduzione, Read Me e Licenza.
3. Per Destination Select, seleziona Installa per tutti gli utenti di questo computer.
4. Per Tipo di installazione, scegli dove AWS SAMCLI verrà installato e premi Installa. La posizione predefinita consigliata è`/usr/local/aws-sam-cli`.

> **Note**
>
> Per richiamarla AWS SAMCLI con il sam comando, il programma di installazione crea automaticamente un collegamento simbolico tra `/usr/local/bin/sam` e una delle due cartelle `/usr/local/aws-sam-cli/sam` di installazione scelte.

5. AWS SAMCLIVerrà visualizzato il messaggio «L'installazione è avvenuta con successo». Premere Chiudi.

Per verificare la corretta installazione

- Verifica che sia AWS SAMCLI installato correttamente e che il tuo link simbolico sia configurato eseguendo:

```
$ which sam
/usr/local/bin/sam
$ sam --version
SAM CLI, <latest version>
```

GUI - Current user

Per scaricare e installare AWS SAMCLI

> **Note**
>
> Se hai già installato il AWS SAMCLI Through Homebrew orpip, devi prima disinstallarlo. Per istruzioni, consulta Disinstallazione di AWS SAMCLI.

1. Scarica macOS in una directory `pkg` a tua scelta:
   - Per i Mac con processori Intel, scegli x86_64 — -x86_64.pkg aws-sam-cli-macos
   - Per i Mac che eseguono Apple Silicon, scegli arm64 — -arm64.pkg aws-sam-cli-macos

   > **Note**
   >
   > È possibile verificare l'integrità del programma di installazione prima dell'installazione. Per istruzioni, consulta Facoltativo: verifica l'integrità del AWS SAMCLI programma di installazione.

2. Esegui il file scaricato e segui le istruzioni sullo schermo per continuare con i passaggi di Introduzione, Read Me e Licenza.
3. Per Destination Select, seleziona Installa solo per me. Se non vedi questa opzione, vai al passaggio successivo.
4. Per Tipo di installazione, procedi come segue:
   1. Scegli dove AWS SAMCLI verrà installato. Il percorso predefinito è `/usr/local/aws-sam-cli`. Seleziona una posizione per la quale disponi delle autorizzazioni di scrittura. Per modificare la posizione di installazione, seleziona locale e scegli la tua posizione. Al termine, premi Continua.
   2. Se non hai avuto la possibilità di scegliere Installa solo per me nel passaggio precedente, seleziona Cambia posizione di installazione > Installa solo per me e premi Continua.
   3. Premi Installa.
5. AWS SAMCLIVerrà installato e verrà visualizzato il messaggio «L'installazione è avvenuta con successo». Premere Chiudi.

Per creare un collegamento simbolico

- Per richiamarlo AWS SAMCLI con il sam comando, è necessario creare manualmente un collegamento simbolico tra il AWS SAMCLI programma e il proprio. $PATH Crea il tuo link simbolico modificando ed eseguendo il seguente comando:

```
$ sudo ln -s /path-to/aws-sam-cli/sam /path-to-symlink-directory/sam
```

  - *sudo*— Se l'utente dispone dei permessi di scrittura per$PATH, non sudo è necessario. In caso contrario, sudo è richiesto.
  - *path-to*— Percorso in cui è stato installato il AWS SAMCLI programma. Ad esempio, `/Users/myUser/Desktop`.
  - *path-to-symlink-directory*— La tua variabile di $PATH ambiente. Il percorso predefinito è `/usr/local/bin`.

Per verificare la corretta installazione

- Verifica che sia AWS SAMCLI installato correttamente e che il tuo link simbolico sia configurato eseguendo:

```
$ which sam
/usr/local/bin/sam
$ sam --version
SAM CLI, <latest version>
```

Command line - All users

Per scaricare e installare AWS SAMCLI

> **Note**
>
> Se hai già installato il AWS SAMCLI Through Homebrew orpip, devi prima disinstallarlo. Per istruzioni, consulta Disinstallazione di AWS SAMCLI.

1. Scarica macOS in una directory pkg a tua scelta:

- Per i Mac con processori Intel, scegli x86_64 — -x86_64.pkg aws-sam-cli-macos
- Per i Mac che eseguono Apple Silicon, scegli arm64 — -arm64.pkg aws-sam-cli-macos

> **Note**
>
> È possibile verificare l'integrità del programma di installazione prima dell'installazione. Per istruzioni, consulta Facoltativo: verifica l'integrità del AWS SAMCLI programma di installazione.

2. Modifica ed esegui lo script di installazione:

```
$ sudo installer -pkg path-to-pkg-installer/name-of-pkg-installer -target /
installer: Package name is AWS SAM CLI
installer: Upgrading at base path /
installer: The upgrade was successful.
```

> **Note**
>
> Per richiamarlo AWS SAMCLI con il sam comando, l'installatore crea automaticamente un collegamento simbolico tra e. `/usr/local/bin/sam` `/usr/local/aws-sam-cli/sam`

Per verificare l'avvenuta installazione

- Verifica che sia AWS SAMCLI installato correttamente e che il tuo link simbolico sia configurato eseguendo:

```
$ which sam
/usr/local/bin/sam
$ sam --version
SAM CLI, <latest version>
```

Command line - Current user

Per scaricare e installare AWS SAMCLI

> **Note**
>
> Se hai già installato il AWS SAMCLI Through Homebrew orpip, devi prima disinstallarlo. Per istruzioni, consulta Disinstallazione di AWS SAMCLI.

1. Scarica macOS in una directory `pkg` a tua scelta:

   - Per i Mac con processori Intel, scegli x86_64 — -x86_64.pkg aws-sam-cli-macos
   - Per i Mac che eseguono Apple Silicon, scegli arm64 — -arm64.pkg aws-sam-cli-macos

   > **Note**
   >
   > È possibile verificare l'integrità del programma di installazione prima dell'installazione. Per istruzioni, consulta Facoltativo: verifica l'integrità del AWS SAMCLI programma di installazione.

2. Determinate una directory di installazione per la quale disponete dei permessi di scrittura. Quindi, create un `xml` file utilizzando il modello e modificatelo in modo che rifletta la directory di installazione. La directory deve già esistere.

   Ad esempio, se si sostituisce *`path-to-my-directory`* con`/Users/myUser/Desktop`, la cartella del `aws-sam-cli` programma verrà installata lì.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
  <array>
    <dict>
      <key>choiceAttribute</key>
      <string>customLocation</string>
      <key>attributeSetting</key>
      <string>path-to-my-directory</string>
      <key>choiceIdentifier</key>
      <string>default</string>
```

```
    </dict>
  </array>
</plist>
```

3. Salva il `xml` file e verifica che sia valido eseguendo quanto segue:

```
$ installer -pkg path-to-pkg-installer \
-target CurrentUserHomeDirectory \
-showChoicesAfterApplyingChangesXML path-to-your-xml-file
```

   L'output dovrebbe mostrare le preferenze che verranno applicate al AWS SAMCLI programma.

4. Esegui quanto segue per installare AWS SAMCLI:

```
$ installer -pkg path-to-pkg-installer \
-target CurrentUserHomeDirectory \
-applyChoiceChangesXML path-to-your-xml-file

# Example output
installer: Package name is AWS SAM CLI
installer: choices changes file 'path-to-your-xml-file' applied
installer: Upgrading at base path base-path-of-xml-file
installer: The upgrade was successful.
```

Per creare un collegamento simbolico

- Per richiamarlo AWS SAMCLI con il sam comando, è necessario creare manualmente un collegamento simbolico tra il AWS SAMCLI programma e il proprio. `$PATH` Crea il tuo link simbolico modificando ed eseguendo il seguente comando:

```
$ sudo ln -s /path-to/aws-sam-cli/sam /path-to-symlink-directory/sam
```

  - *`sudo`*— Se l'utente dispone dei permessi di scrittura per`$PATH`, non sudo è necessario. In caso contrario, sudo è richiesto.
  - *`path-to`*— Percorso in cui è stato installato il AWS SAMCLI programma. Ad esempio, `/Users/myUser/Desktop`.
  - *`path-to-symlink-directory`*— La tua variabile di `$PATH` ambiente. Il percorso predefinito è `/usr/local/bin`.

Per verificare la corretta installazione

- Verifica che sia AWS SAMCLI installato correttamente e che il tuo link simbolico sia configurato eseguendo:

```
$ which sam
/usr/local/bin/sam
$ sam --version
SAM CLI, <latest version>
```

Windows

I file Windows Installer (MSI) sono i file di installazione dei pacchetti per il sistema operativo Windows.

Segui questi passaggi per installarli AWS SAMCLI utilizzando il file MSI.

1. Scarica la AWS SAMCLI versione a 64 bit.
2. (Facoltativo) È possibile verificare l'integrità del programma di installazione prima dell'installazione. Per istruzioni, consulta Facoltativo: verifica l'integrità del AWS SAMCLI programma di installazione.
3. Verifica l'installazione.

   Dopo aver completato l'installazione, verificatela aprendo un nuovo prompt o PowerShell prompt dei comandi. Dovresti essere in grado di richiamare `sam` dalla riga di comando.

   ```
   sam --version
   ```

   Dopo la corretta installazione di AWS SAMCLI, si dovrebbe vedere un risultato simile al seguente:

   ```
   SAM CLI, <latest version>
   ```

4. Abilita percorsi lunghi (solo Windows 10 e versioni successive).

   > **Important**
   >
   > AWS SAMCLIPotrebbero interagire con percorsi di file che superano il limite massimo di percorso di Windows. Ciò può causare errori durante l'esecuzione `sam init` a causa

delle MAX_PATH limitazioni di Windows 10. Per risolvere questo problema, è necessario configurare il nuovo comportamento dei percorsi lunghi.

Per abilitare percorsi lunghi, vedi Abilitare percorsi lunghi in Windows 10, versione 1607 e successive nella documentazione sullo sviluppo di app per Microsoft Windows.

5. Installa Git.

   Per scaricare applicazioni di esempio utilizzando il `sam init` comando, è necessario installare anche Git. Per istruzioni, consulta Installazione di Git.

Risoluzione degli errori di installazione

Linux

Errore Docker: «Impossibile connettersi al demone Docker. Il demone docker è in esecuzione su questo host?»

In alcuni casi, per fornire le autorizzazioni per accedere `ec2-user` al demone Docker, potrebbe essere necessario riavviare l'istanza. Se ricevi questo errore, prova a riavviare l'istanza.

Errore di shell: «comando non trovato»

Se ricevi questo errore, la tua shell non è in grado di localizzare l' AWS SAMCLIeseguibile nel percorso. Verifica la posizione della directory in cui hai installato l' AWS SAMCLIeseguibile, quindi verifica che la directory si trovi sul tuo percorso.

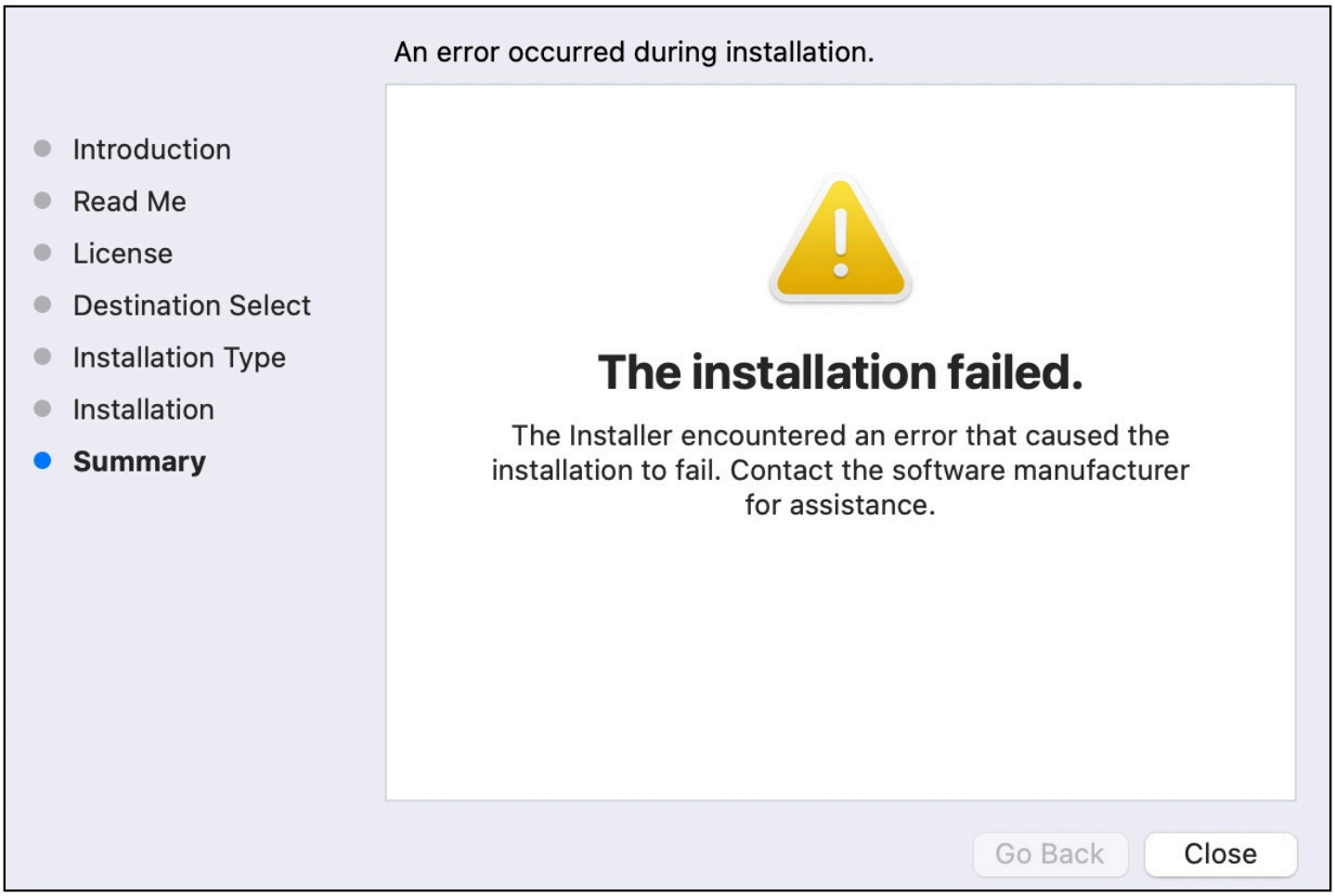
AWS SAMCLIerrore: «/lib64/libc.so.6: versione `GLIBC_2.14' non trovata (richiesta da/.so.1)» usr/local/aws-sam-cli/dist/libz

Se ricevi questo errore, stai usando una versione non supportata di Linux e la versione glibc integrata non è aggiornata. Prova una delle seguenti soluzioni:

- Aggiorna il tuo host Linux alla versione a 64 bit di una distribuzione recente di CentOS, Fedora, Ubuntu o Amazon Linux 2.
- Segui le istruzioni per. Installa il AWS SAMCLI

macOS

L'installazione non è riuscita



Se stai installando il file AWS SAMCLI per il tuo utente e hai selezionato una directory di installazione per la quale non disponi delle autorizzazioni di scrittura, potrebbe verificarsi questo errore. Prova una delle seguenti soluzioni:

1. Seleziona una directory di installazione diversa per la quale disponi dei permessi di scrittura.
2. Eliminare il programma di installazione. Quindi, scaricalo ed eseguilo di nuovo.

Fasi successive

Per ulteriori informazioni AWS SAMCLI e per iniziare a creare applicazioni serverless personalizzate, consulta quanto segue:

- Tutorial: Implementa un'applicazione Hello World con AWS SAM— Step-by-step istruzioni per scaricare, creare e distribuire un'applicazione serverless di base.
- Il AWS SAM workshop completo: un workshop progettato per insegnarti molte delle principali funzionalità che AWS SAM offre.
- AWS SAM esempi di applicazioni e modelli: esempi di applicazioni e modelli di autori della community con cui è possibile sperimentare ulteriormente.

Facoltativo: verifica l'integrità del AWS SAMCLI programma di installazione

Quando si installa l'interfaccia a riga di AWS Serverless Application Model comando (AWS SAMCLI) utilizzando un programma di installazione di pacchetti, è possibile verificarne l'integrità prima dell'installazione. Si tratta di un passaggio facoltativo, ma altamente consigliato.

Le due opzioni di verifica disponibili sono:

- Verifica il file di firma del programma di installazione del pacchetto.
- Verifica il valore hash del programma di installazione del pacchetto.

Se disponibile per la tua piattaforma, ti consigliamo di verificare l'opzione del file di firma. Questa opzione offre un ulteriore livello di sicurezza poiché i valori chiave sono pubblicati qui e gestiti separatamente dal nostro GitHub repository.

Argomenti

- Verifica il file di firma dell'installatore
- Verifica il valore hash

Verifica il file di firma dell'installatore

Linux

arm64 - programma di installazione a riga di comando

AWS SAM usa GnuPG per firmare AWS SAMCLI il programma di installazione.zip. La verifica viene eseguita nei seguenti passaggi:

1. Utilizza la chiave pubblica primaria per verificare la chiave pubblica del firmatario.

2. Usa la chiave pubblica del firmatario per verificare il programma di installazione del AWS SAMCLI pacchetto.

Per verificare l'integrità della chiave pubblica del firmatario

1. Copia la chiave pubblica principale e salvala sul computer locale come `.txt` file. Ad esempio, *`primary-public-key.txt`*.


```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2.0.22 (GNU/Linux)

mQINBGRuSzMBEADsqiwOy78w7F4+sshaMFRIwRGNRm94p5Qey2KMZBxekFtoryVD
D9jEOnvupx4tvhfBHz5EcUHCEOdl4MTqdBy6vVAshozgxVb9RE8JpECn5lw7XC69
4Y7Gy1TKKQMEWtDXElkGxIFdUWvWjSnPlzfnoXwQYGeE93CUS3h5dImP22Yk1Ct6
eGGhlcbg1X4L8EpFMj7GvcsU8f7ziVI/PyC1Xwy39Q8/I67ip5eU5ddxO/xHqrbL
YC7+8pJPbRMej2twT2LrcpWWYAbprMtRoa6WfE0/thoo3xhHpIMHdPfAA86ZNGIN
kRLjGUg7jnPTRW4Oin3pCc8nT4Tfc1QERkHm641gTC/jUvpmQsM6h/FUVP2i5iE/
JHpJcMuL2Mg6zDo3x+3gTCf+Wqz3rZzxB+wQT3yryZs6efcQy7nROiRxYBxCSXX0
2cNYzsYLb/bYaW8yqWIHD5IqKhw269gp2E5Khs60zgS3CorMb5/xHgXjUCVgcu8a
a8ncdf9fjl3WS5p0ohetPbO2ZjWv+MaqrZOmUIgKbA4RpWZ/fU97P5BW9ylwmIDB
sWy0cMxg8MlvSdLytPieogaM0qMg3u5qXRGBr6Wmevkty0qgnmpGGc5zPiUbtOE8
CnFFqyxBpj5IOnG0KZGVihvn+iRxrv6GO7WWO92+Dc6m94U0EEiBR7QiOwARAQAB
tDRBV1MgU0FNIENMSSBQcmltYXJ5IDxhd3Mtc2FtLWNsaS1wcmltYXJ5QGFtYXpv
bi5jb20+iQI/BBMBCQApBQJkbkszAhsvBQkHhM4ABwsJCAcDAgEGFQgCCQoLBBYC
AwECHgECF4AACgkQQv1fenOtiFqTuhAAzi5+ju5UVOWqHKevOJSO08T4QB8HcqAE
SVO3mY6/j29knkcL8ubZP/DbpV7QpHPI2PB5qSXsiDTP3IYPbeY78zHSDjljaIK3
njJLMScFeGPyfPpwMsuY4nzrRIgAtXShPA8N/k4ZJcafnpNqKj7QnPxiC1KaIQWm
pOtvb8msUF3/s0UTa5Ys/lNRhVC0eGg32ogXGdojZA2kHZWdm9udLo4CDrDcrQT7
NtDcJASapXSQL63XfAS3snEc4e1941YxcjfYZ33rel8K9juyDZfi1slWR/L3AviI
QFIaqSHzyOtP1oinUkoVwL8ThevKD3Ag9CZflZLzNCV7yqlF8RlhEZ4zcE/3s9El
WzCFsozb5HfE1AZonmrDh3SyOEIBMcS6vG5dWnvJrAuSYv2rX38++K5Pr/MIAfOX
DOI1rtA+XDsHNv9lSwSy0lt+iClawZANO9IXCiN1rOYcVQlwzDFwCNWDgkwdOqS0
gOA2f8NF9lE5nBbeEuYquoOl1Vy8+ICbgOFs9LoWZlnVh7/RyY6ssowiU9vGUnHI
L8f9jqRspIz/Fm3JD86ntZxLVGkeZUz62FqErdohYfkFIVcv7GONTEyrz5HLlnpv
FJ0MR0HjrMrZrnOVZnwBKhpbLocTsH+3t5It4ReYEX0f1DIOL/KRwPvjMvBVkXY5
hblRVDQoOWc=
=d9oG
-----END PGP PUBLIC KEY BLOCK-----
```


2. Importa la chiave pubblica principale nel tuo portachiavi.

```
$ gpg --import primary-public-key.txt
```

```
gpg: directory `/home/.../.gnupg' created
gpg: new configuration file `/home/.../.gnupg/gpg.conf' created
gpg: WARNING: options in `/home/.../.gnupg/gpg.conf' are not yet active during this
 run
gpg: keyring `/home/.../.gnupg/secring.gpg' created
gpg: keyring `/home/.../.gnupg/pubring.gpg' created
gpg: /home/.../.gnupg/trustdb.gpg: trustdb created
gpg: key 73AD885A: public key "AWS SAM CLI Primary <aws-sam-cli-
primary@amazon.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
```

3. Copia la chiave pubblica del firmatario e salvala sul computer locale come `.txt` file. Ad esempio, *`signer-public-key.txt`*.


```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2.0.22 (GNU/Linux)

mQINBGgrxIgBEADGCTudveeeVbWpZDGX9Ni57mBRMVSJwQJ6F/PC34jw0DozxTtd
H+ZPsXLvLwerN/DVXbK8E1qNZ5RGptak8j7MPz+MC3n4txibEJpB61vpjJJM+9cC
7whaMLDT/SbykHYXdrnHqa8KsUJl7rPLJcaRN722NSxvYVMIOA9ffVXV7cfEyZi5
MbYF2Gc9LNbKaknImIva7EKeeh2/wI6YCqC5yytyfWU5dL6oHXsgTnFL9mhziMxv
WhyzawyJG6EJZsJ3WLlbIKApN6XZSXyCxOvlBrebYZjD5v0nA+TJaQ7is8atjtOI
DGe0AViw7kO8ChTpjA7YG/Uu7n/Fy7qLF/3Nz0b6cBNjemjBazQ3A3KNCpi5hqFM
Uo1WpoVLr5CXQnc0B3fBUnTIoxi0Sk5MKjH9AbYxfgqEX0ZJB9hAlc6LIEy0Yru6
MMBrIHE86IMl1NfE/DeLnCdPG23+1PttwyOt3+9z5QwmPe3VPpEfCySPcdxHKZSP
rLile8qDznEvlPDvQ0qkBxdMtVa2yct5VJkdqy6UrN2xa0dpspHjRUjHh/EY/xMt
fwMUjOKohaZ/1pjotCcksAsZWUxCNcFvLYxuxeytVk4F09Es1hj4ihhLUI+43/ic
3DHSEiext7Q8/UccNArkhSCT7UOvvL7QTuP+pjYTyiC8Vx6g/Y5Ht5+qywARAQAB
tDBBV1MgU0FNIENMSSBUZWFtIDxhd3Mtc2FtLWNsaS1zaWduZXJAYW1hem9uLmNv
bT6JAj8EEwEJACkFAmgrxIgCGy8FCQPCZwAHCwkIBwMCAQYVCAIJCgsEFgIDAQIe
AQIXgAAKCRBAlKuxvt/atJo6EAC/5C8uJs76W5f5V5XNAMzwBFiZuYpop3DRReCo
P68ZZylokAC9ShRZnIOujpDJtlNS7T/G00BzmcpspkYYE531ALaXcHWmb9XV0Ajg
J8iboAVBLY0C7mhL/cbJ3v9QlpXXjyTuhexkJCV8rdHVX/0H8WqTZplEaRuZ7p8q
PMxddg4ClwstYuH3O/dmNdlGqfb4Fqy8MnV1yGSXRs5Jf+sDlN2UO4mbpyk/mr1c
f/jFxmx86IkCWJVvdXWCVTe2AFy3NHCdLtdnEvFhokCOQd9wibUWX0j9vq4cVRZT
qamnpAQaOlH3lXOwrjqo8b1AIPoRWSfMtCYvh6kA8MAJv4cAznzXILSLtOE0mzaU
qp5qoy37wNIjeztX6c/q4wss05qTlJhnNu4s3nh5VHultooaYpmDxp+ala5TWeuM
KZDI4KdAGF4z0Raif+N53ndOYIiXkY0goUbsPCnVrCwoK9PjjyoJncq7c14wNl5O
IQUZEjyYAQDGZqs5XSfY4zW2cCXatrfozKF7R1kSU14DfJwPUyksoNAQEQezfXyq
kr0gfIWK1r2nMdqS7WgSx/ypS5kdyrHuPZdaYfEVtuezpoT2lQQxOSZqqlp5hI4R
nqmPte53WXJhbC0tgTIJWn+Uy/d5Q/aSIfD6o8gNLS1BDs1j1ku0XKu1sFCHUcZG
```

```
aerdsIkCHAQQAQkABgUCaCvFeAAKCRBC/V96c62IWt3/D/9gOLzWtz62lqJRCsri
wcA/yz88ayKb/GUv3FCT5Nd9JZt8y1tW+AE3SPTdcpfZmt5UN2sRzljO61mpKJzp
eBvYQ9og/34ZrRQqeg8bz02u34LKYl1gD0xY0bWtB7TGIxIZZYqZECoPR0Dp6ZzB
abzkRSsJkEk0vbZzJhfWFYs98qfp/G0suFSBE79O8Am33DB2jQ/Sollh1VmNE6Sv
EOgR6+2yEkS2D0+msJMa/V82v9gBTPnxSlNV1d8Dduvt9rbM3LoxiNXUgx/s52yY
U6H3bwUcQ3UY6uRe1UWo5QnMFcDwfg43+q5rmjB4xQyX/BaQyF5K0hZyG+42/pH1
EMwl8qN617FTxo3hvQUi/cBahlhQ8EVYsGnHDVxLCisbq5iZvp7+XtmMy1Q417gT
EQRo8feJh31elGWlccVR2pZgIm1PQ69dzzseHnnKkGhifik0bDGo5/IH2EgI1KFn
SG399RMU/qRzOPLVP3i+zSJmhMqG8cnZaUwE5V4P21vQSclhhd2Hv/C4SVKNqA2i
+oZbHj2vAkuzTTL075AoANebEjPGqwsKZi5mWUE5Pa931JeiXxWZlEB7rkgQ1PAB
fsDBhYLt4MxCWAhifLMA6uQ4BhXu2RuXOqNfSbqa8jVF6DB6cD8eAHGpPKfJOl30
LtZnq+n4SfeNbZjD2FQWZR4CrA==
=lHfs
-----END PGP PUBLIC KEY BLOCK-----
```

4. Importa la chiave pubblica del firmatario nel tuo portachiavi.

```
$ gpg --import signer-public-key.txt

gpg: key 4094ABB1BEDFDAB4: public key "AWS SAM CLI Team <aws-sam-cli-
signer@amazon.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
gpg: no ultimately trusted keys found
```

Prendi nota del valore della chiave dall'output. Ad esempio, *4094ABB1BEDFDAB4*.

5. Usa il valore della chiave per ottenere e verificare l'impronta digitale della chiave pubblica del firmatario.

```
$ gpg --fingerprint 4094ABB1BEDFDAB4

pub   rsa4096 2025-05-19 [SCEA] [expires: 2027-05-19]
      EF46 3E86 CA31 933B B688  CC1A 4094 ABB1 BEDF DAB4
uid           [ unknown] AWS SAM CLI Team <aws-sam-cli-signer@amazon.com>
```

L'impronta digitale deve corrispondere alla seguente:

```
EF46 3E86 CA31 933B B688  CC1A 4094 ABB1 BEDF DAB4
```

Se la stringa dell'impronta digitale non corrisponde, non utilizzate il AWS SAMCLI programma di installazione. Rivolgiti al AWS SAM team creando un problema nell'archivio. aws-sam-cli GitHub

6. Verifica le firme della chiave pubblica del firmatario:

```
$ gpg --check-sigs 4094ABB1BEDFDAB4

pub   rsa4096 2025-05-19 [SCEA] [expires: 2027-05-19]
      EF463E86CA31933BB688CC1A4094ABB1BEDFDAB4
uid           [ unknown] AWS SAM CLI Team <aws-sam-cli-signer@amazon.com>
sig!3        4094ABB1BEDFDAB4 2025-05-19  [self-signature]
sig!         42FD5F7A73AD885A 2025-05-19 AWS SAM CLI Primary <aws-sam-cli-
primary@amazon.com>
```

   Se vedi `1 signature not checked due to a missing key`, ripeti i passaggi precedenti per importare la chiave pubblica principale e quella del firmatario nel tuo portachiavi.

   Dovresti vedere elencati i valori chiave sia per la chiave pubblica principale che per la chiave pubblica del firmatario.

Ora che hai verificato l'integrità della chiave pubblica del firmatario, puoi utilizzare la chiave pubblica del firmatario per verificare il programma di installazione del AWS SAMCLI pacchetto.

Per verificare l'integrità del programma di installazione del pacchetto AWS SAMCLI

1. Ottieni il file della firma del AWS SAMCLI pacchetto: scarica il file della firma per il programma di installazione del AWS SAMCLI pacchetto utilizzando il seguente comando:

```
$ wget https://github.com/aws/aws-sam-cli/releases/latest/download/aws-sam-cli-
linux-arm64.zip.sig
```

2. Verifica il file della firma: passa sia il file scaricato che `.sig` i `.zip` file come parametri al `gpg` comando. Di seguito è riportato un esempio:

```
$ gpg --verify aws-sam-cli-linux-arm64.zip.sig aws-sam-cli-linux-arm64.zip
```

   L'output visualizzato dovrebbe essere simile al seguente:

```
gpg: Signature made Mon 19 May 2025 01:21:57 AM UTC using RSA key ID
 4094ABB1BEDFDAB4
gpg: Good signature from "AWS SAM CLI Team <aws-sam-cli-signer@amazon.com>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
```

```
Primary key fingerprint: EF46 3E86 CA31 933B B688  CC1A 4094 ABB1 BEDF DAB4
```

- Il `WARNING: This key is not certified with a trusted signature!` messaggio può essere ignorato. Si verifica perché non esiste una catena di fiducia tra la tua chiave PGP personale (se ne hai una) e la chiave PGP AWS SAM CLI. Per ulteriori informazioni, consulta Web of trust.
- Se l'output contiene la fraseBAD `signature`, verificate di aver eseguito correttamente la procedura. Se continui a ricevere questa risposta, rivolgiti al AWS SAM team creando un problema nell'aws-sam-cli GitHub archivio ed evita di usare il file scaricato.

Il `Good signature from "AWS SAM CLI Team <aws-sam-cli-signer@amazon.com>"` messaggio indica che la firma è stata verificata e che puoi procedere con l'installazione.

x86_64 - programma di installazione da riga di comando

AWS SAM usa GnuPG per firmare AWS SAMCLI il programma di installazione.zip. La verifica viene eseguita nei seguenti passaggi:

1. Utilizza la chiave pubblica primaria per verificare la chiave pubblica del firmatario.
2. Usa la chiave pubblica del firmatario per verificare il programma di installazione del AWS SAMCLI pacchetto.

Per verificare l'integrità della chiave pubblica del firmatario

1. Copia la chiave pubblica principale e salvala sul computer locale come `.txt` file. Ad esempio, *`primary-public-key.txt`*.


```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2.0.22 (GNU/Linux)

mQINBGRuSzMBEADsqiwOy78w7F4+sshaMFRIwRGNRm94p5Qey2KMZBxekFtoryVD
D9jEOnvupx4tvhfBHz5EcUHCEOdl4MTqdBy6vVAshozgxVb9RE8JpECn5lw7XC69
4Y7Gy1TKKQMEWtDXElkGxIFdUWvWjSnPlzfnoXwQYGeE93CUS3h5dImP22Yk1Ct6
eGGhlcbg1X4L8EpFMj7GvcsU8f7ziVI/PyC1Xwy39Q8/I67ip5eU5ddxO/xHqrbL
YC7+8pJPbRMej2twT2LrcpWWYAbprMtRoa6WfE0/thoo3xhHpIMHdPfAA86ZNGIN
kRLjGUg7jnPTRW4Oin3pCc8nT4Tfc1QERkHm641gTC/jUvpmQsM6h/FUVP2i5iE/
JHpJcMuL2Mg6zDo3x+3gTCf+Wqz3rZzxB+wQT3yryZs6efcQy7nROiRxYBxCSXX0
2cNYzsYLb/bYaW8yqWIHD5IqKhw269gp2E5Khs60zgS3CorMb5/xHgXjUCVgcu8a
```

```
a8ncdf9fjl3WS5p0ohetPbO2ZjWv+MaqrZOmUIgKbA4RpWZ/fU97P5BW9ylwmIDB
sWy0cMxg8MlvSdLytPieogaM0qMg3u5qXRGBr6Wmevkty0qgnmpGGc5zPiUbtOE8
CnFFqyxBpj5IOnG0KZGVihvn+iRxrv6GO7WWO92+Dc6m94U0EEiBR7QiOwARAQAB
tDRBV1MgU0FNIENMSSBQcmltYXJ5IDxhd3Mtc2FtLWNsaS1wcmltYXJ5QGFtYXpv
bi5jb20+iQI/BBMBCQApBQJkbkszAhsvBQkHhM4ABwsJCAcDAgEGFQgCCQoLBBYC
AwECHgECF4AACgkQQv1fenOtiFqTuhAAzi5+ju5UVOWqHKevOJSO08T4QB8HcqAE
SVO3mY6/j29knkcL8ubZP/DbpV7QpHPI2PB5qSXsiDTP3IYPbeY78zHSDjljaIK3
njJLMScFeGPyfPpwMsuY4nzrRIgAtXShPA8N/k4ZJcafnpNqKj7QnPxiC1KaIQWm
pOtvb8msUF3/s0UTa5Ys/lNRhVC0eGg32ogXGdojZA2kHZWdm9udLo4CDrDcrQT7
NtDcJASapXSQL63XfAS3snEc4e1941YxcjfYZ33rel8K9juyDZfi1slWR/L3AviI
QFIaqSHzyOtP1oinUkoVwL8ThevKD3Ag9CZflZLzNCV7yqlF8RlhEZ4zcE/3s9El
WzCFsozb5HfE1AZonmrDh3SyOEIBMcS6vG5dWnvJrAuSYv2rX38++K5Pr/MIAfOX
DOI1rtA+XDsHNv9lSwSy0lt+iClawZANO9IXCiN1rOYcVQlwzDFwCNWDgkwdOqS0
gOA2f8NF9lE5nBbeEuYquoOl1Vy8+ICbgOFs9LoWZlnVh7/RyY6ssowiU9vGUnHI
L8f9jqRspIz/Fm3JD86ntZxLVGkeZUz62FqErdohYfkFIVcv7GONTEyrz5HLlnpv
FJ0MR0HjrMrZrnOVZnwBKhpbLocTsH+3t5It4ReYEX0f1DIOL/KRwPvjMvBVkXY5
hblRVDQoOWc=
=d9oG
-----END PGP PUBLIC KEY BLOCK-----
```

2. Importa la chiave pubblica principale nel tuo portachiavi.

```
$ gpg --import primary-public-key.txt

gpg: directory `/home/.../.gnupg' created
gpg: new configuration file `/home/.../.gnupg/gpg.conf' created
gpg: WARNING: options in `/home/.../.gnupg/gpg.conf' are not yet active during this
 run
gpg: keyring `/home/.../.gnupg/secring.gpg' created
gpg: keyring `/home/.../.gnupg/pubring.gpg' created
gpg: /home/.../.gnupg/trustdb.gpg: trustdb created
gpg: key 73AD885A: public key "AWS SAM CLI Primary <aws-sam-cli-
primary@amazon.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
```

3. Copia la chiave pubblica del firmatario e salvala sul computer locale come `.txt` file. Ad esempio, *`signer-public-key.txt`*.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2.0.22 (GNU/Linux)

mQINBGgrxIgBEADGCTudveeeVbWpZDGX9Ni57mBRMVSJwQJ6F/PC34jw0DozxTtd
```

```
H+ZPsXLvLwerN/DVXbK8E1qNZ5RGptak8j7MPz+MC3n4txibEJpB61vpjJJM+9cC
7whaMLDT/SbykHYXdrnHqa8KsUJl7rPLJcaRN722NSxvYVMIOA9ffVXV7cfEyZi5
MbYF2Gc9LNbKaknImIva7EKeeh2/wI6YCqC5yytyfWU5dL6oHXsgTnFL9mhziMxv
WhyzawyJG6EJZsJ3WLlbIKApN6XZSXyCxOvlBrebYZjD5v0nA+TJaQ7is8atjtOI
DGe0AViw7kO8ChTpjA7YG/Uu7n/Fy7qLF/3Nz0b6cBNjemjBazQ3A3KNCpi5hqFM
Uo1WpoVLr5CXQnc0B3fBUnTIoxi0Sk5MKjH9AbYxfgqEX0ZJB9hAlc6LIEy0Yru6
MMBrIHE86IMl1NfE/DeLnCdPG23+1PttwyOt3+9z5QwmPe3VPpEfCySPcdxHKZSP
rLile8qDznEvlPDvQ0qkBxdMtVa2yct5VJkdqy6UrN2xa0dpspHjRUjHh/EY/xMt
fwMUjOKohaZ/1pjotCcksAsZWUxCNcFvLYxuxeytVk4F09Es1hj4ihhLUI+43/ic
3DHSEiext7Q8/UccNArkhSCT7UOvvL7QTuP+pjYTyiC8Vx6g/Y5Ht5+qywARAQAB
tDBBV1MgU0FNIENMSSBUZWFtIDxhd3Mtc2FtLWNsaS1zaWduZXJAYW1hem9uLmNv
bT6JAj8EEwEJACkFAmgrxIgCGy8FCQPCZwAHCwkIBwMCAQYVCAIJCgsEFgIDAQIe
AQIXgAAKCRBAlKuxvt/atJo6EAC/5C8uJs76W5f5V5XNAMzwBFiZuYpop3DRReCo
P68ZZylokAC9ShRZnIOujpDJtlNS7T/G00BzmcpspkYYE531ALaXcHWmb9XV0Ajg
J8iboAVBLY0C7mhL/cbJ3v9QlpXXjyTuhexkJCV8rdHVX/0H8WqTZplEaRuZ7p8q
PMxddg4ClwstYuH3O/dmNdlGqfb4Fqy8MnV1yGSXRs5Jf+sDlN2UO4mbpyk/mr1c
f/jFxmx86IkCWJVvdXWCVTe2AFy3NHCdLtdnEvFhokCOQd9wibUWX0j9vq4cVRZT
qamnpAQaOlH3lXOwrjqo8b1AIPoRWSfMtCYvh6kA8MAJv4cAznzXILSLtOE0mzaU
qp5qoy37wNIjeztX6c/q4wss05qTlJhnNu4s3nh5VHultooaYpmDxp+ala5TWeuM
KZDI4KdAGF4z0Raif+N53ndOYIiXkY0goUbsPCnVrCwoK9PjjyoJncq7c14wNl5O
IQUZEjyYAQDGZqs5XSfY4zW2cCXatrfozKF7R1kSU14DfJwPUyksoNAQEQezfXyq
kr0gfIWK1r2nMdqS7WgSx/ypS5kdyrHuPZdaYfEVtuezpoT2lQQxOSZqqlp5hI4R
nqmPte53WXJhbC0tgTIJWn+Uy/d5Q/aSIfD6o8gNLS1BDs1j1ku0XKu1sFCHUcZG
aerdsIkCHAQQAQkABgUCaCvFeAAKCRBC/V96c62IWt3/D/9gOLzWtz62lqJRCsri
wcA/yz88ayKb/GUv3FCT5Nd9JZt8y1tW+AE3SPTdcpfZmt5UN2sRzljO61mpKJzp
eBvYQ9og/34ZrRQqeg8bz02u34LKYl1gD0xY0bWtB7TGIxIZZYqZECoPR0Dp6ZzB
abzkRSsJkEk0vbZzJhfWFYs98qfp/G0suFSBE79O8Am33DB2jQ/Sollh1VmNE6Sv
EOgR6+2yEkS2D0+msJMa/V82v9gBTPnxSlNV1d8Dduvt9rbM3LoxiNXUgx/s52yY
U6H3bwUcQ3UY6uRe1UWo5QnMFcDwfg43+q5rmjB4xQyX/BaQyF5K0hZyG+42/pH1
EMwl8qN617FTxo3hvQUi/cBahlhQ8EVYsGnHDVxLCisbq5iZvp7+XtmMy1Q417gT
EQRo8feJh31elGWlccVR2pZgIm1PQ69dzzseHnnKkGhifik0bDGo5/IH2EgI1KFn
SG399RMU/qRzOPLVP3i+zSJmhMqG8cnZaUwE5V4P21vQSclhhd2Hv/C4SVKNqA2i
+oZbHj2vAkuzTTL075AoANebEjPGqwsKZi5mWUE5Pa931JeiXxWZlEB7rkgQ1PAB
fsDBhYLt4MxCWAhifLMA6uQ4BhXu2RuXOqNfSbqa8jVF6DB6cD8eAHGpPKfJOl30
LtZnq+n4SfeNbZjD2FQWZR4CrA==
=lHfs
-----END PGP PUBLIC KEY BLOCK-----
```


4. Importa la chiave pubblica del firmatario nel tuo portachiavi.

```
$ gpg --import signer-public-key.txt

gpg: key 4094ABB1BEDFDAB4: public key "AWS SAM CLI Team <aws-sam-cli-
signer@amazon.com>" imported
```

```
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
gpg: no ultimately trusted keys found
```

Prendi nota del valore della chiave dall'output. Ad esempio, *4094ABB1BEDFDAB4*.

5. Usa il valore della chiave per ottenere e verificare l'impronta digitale della chiave pubblica del firmatario.

```
$ gpg --fingerprint 4094ABB1BEDFDAB4

pub   rsa4096 2025-05-19 [SCEA] [expires: 2027-05-19]
      EF46 3E86 CA31 933B B688  CC1A 4094 ABB1 BEDF DAB4
uid           [ unknown] AWS SAM CLI Team <aws-sam-cli-signer@amazon.com>
```

L'impronta digitale deve corrispondere alla seguente:

```
EF46 3E86 CA31 933B B688  CC1A 4094 ABB1 BEDF DAB4
```

Se la stringa dell'impronta digitale non corrisponde, non utilizzate il AWS SAMCLI programma di installazione. Rivolgiti al AWS SAM team creando un problema nell'archivio. aws-sam-cli GitHub

6. Verifica le firme della chiave pubblica del firmatario:

```
$ gpg --check-sigs 4094ABB1BEDFDAB4

pub   rsa4096 2025-05-19 [SCEA] [expires: 2027-05-19]
      EF463E86CA31933BB688CC1A4094ABB1BEDFDAB4
uid           [ unknown] AWS SAM CLI Team <aws-sam-cli-signer@amazon.com>
sig!3        4094ABB1BEDFDAB4 2025-05-19  [self-signature]
sig!         42FD5F7A73AD885A 2025-05-19 AWS SAM CLI Primary <aws-sam-cli-
primary@amazon.com>
```

Se vedi`1 signature not checked due to a missing key`, ripeti i passaggi precedenti per importare la chiave pubblica principale e quella del firmatario nel tuo portachiavi.

Dovresti vedere elencati i valori chiave sia per la chiave pubblica principale che per la chiave pubblica del firmatario.

Ora che hai verificato l'integrità della chiave pubblica del firmatario, puoi utilizzare la chiave pubblica del firmatario per verificare il programma di installazione del AWS SAMCLI pacchetto.

Per verificare l'integrità del programma di installazione del pacchetto AWS SAMCLI

1. Ottieni il file della firma del AWS SAMCLI pacchetto: scarica il file della firma per il programma di installazione del AWS SAMCLI pacchetto utilizzando il seguente comando:

```
$ wget https://github.com/aws/aws-sam-cli/releases/latest/download/aws-sam-cli-linux-x86_64.zip.sig
```

2. Verifica il file della firma: passa sia il file scaricato che `.sig` i `.zip` file come parametri al `gpg` comando. Di seguito è riportato un esempio:

```
$ gpg --verify aws-sam-cli-linux-x86_64.zip.sig aws-sam-cli-linux-x86_64.zip
```

L'output visualizzato dovrebbe essere simile al seguente:

```
gpg: Signature made Mon 19 May 2025 01:21:57 AM UTC using RSA key ID
 4094ABB1BEDFDAB4
gpg: Good signature from "AWS SAM CLI Team <aws-sam-cli-signer@amazon.com>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: EF46 3E86 CA31 933B B688  CC1A 4094 ABB1 BEDF DAB4
```

- Il `WARNING: This key is not certified with a trusted signature!` messaggio può essere ignorato. Si verifica perché non esiste una catena di fiducia tra la tua chiave PGP personale (se ne hai una) e la chiave PGP AWS SAM CLI. Per ulteriori informazioni, consulta Web of trust.
- Se l'output contiene la fraseBAD `signature`, verificate di aver eseguito correttamente la procedura. Se continui a ricevere questa risposta, rivolgiti al AWS SAM team creando un problema nell'aws-sam-cli GitHub archivio ed evita di usare il file scaricato.

Il `Good signature from "AWS SAM CLI Team <aws-sam-cli-signer@amazon.com>"` messaggio indica che la firma è stata verificata e che puoi procedere con l'installazione.

macOS

GUI e programma di installazione a riga di comando

È possibile verificare l'integrità del file di firma del programma di installazione del AWS SAMCLI pacchetto utilizzando lo `pkgutil` strumento o manualmente.

Per verificare usando pkgutil

1. Esegui il comando seguente, fornendo il percorso del programma di installazione scaricato sul tuo computer locale:

```
$ pkgutil --check-signature /path/to/aws-sam-cli-installer.pkg
```

   Di seguito è riportato un esempio:

```
$ pkgutil --check-signature /Users/user/Downloads/aws-sam-cli-macos-arm64.pkg
```

2. Dall'output, individua il file SHA256 fingerprint perDeveloper ID Installer: AMZN Mobile LLC. Di seguito è riportato un esempio:

```
Package "aws-sam-cli-macos-arm64.pkg":
   Status: signed by a developer certificate issued by Apple for distribution
   Notarization: trusted by the Apple notary service
   Signed with a trusted timestamp on: 2026-01-28 07:39:16 +0000
   Certificate Chain:
    1. Developer ID Installer: AMZN Mobile LLC (94KV3E626L)
       Expires: 2030-09-26 00:18:06 +0000
       SHA256 Fingerprint:
           5C 45 BE 63 FD 52 10 07 2D 66 56 77 5C A9 FF 25 91 6D 3F 01 F7 0E
           9A 8A 05 F6 2D 62 B2 88 8D A9
       ------------------------------------------------------------------------
    2. Developer ID Certification Authority
       Expires: 2031-09-17 00:00:00 +0000
       SHA256 Fingerprint:
           F1 6C D3 C5 4C 7F 83 CE A4 BF 1A 3E 6A 08 19 C8 AA A8 E4 A1 52 8F
           D1 44 71 5F 35 06 43 D2 DF 3A
       ------------------------------------------------------------------------
    3. Apple Root CA
       Expires: 2035-02-09 21:40:36 +0000
       SHA256 Fingerprint:
           B0 B1 73 0E CB C7 FF 45 05 14 2C 49 F1 29 5E 6E DA 6B CA ED 7E 2C
```

```
            68 C5 BE 91 B5 A1 10 01 F0 24
```

3. Developer ID Installer: AMZN Mobile LLC SHA256 fingerprintDovrebbe corrispondere al seguente valore:

```
5C 45 BE 63 FD 52 10 07 2D 66 56 77 5C A9 FF 25 91 6D 3F 01 F7 0E 9A 8A 05 F6 2D 62
 B2 88 8D A9
```

   Se la stringa dell'impronta digitale non corrisponde, non utilizzate il AWS SAMCLI programma di installazione. Rivolgiti al AWS SAM team creando un problema nell'archivio. aws-sam-cli GitHub Se la stringa dell'impronta digitale corrisponde, puoi procedere con l'utilizzo del programma di installazione del pacchetto.

Per verificare manualmente il programma di installazione del pacchetto

- Vedi Come verificare l'autenticità degli aggiornamenti software Apple scaricati manualmente sul sito web del supporto Apple.

Windows

Il AWS SAMCLI programma di installazione è confezionato come MSI file per il Windows sistema operativo.

Per verificare l'integrità del programma di installazione

1. Fare clic con il pulsante destro del mouse sul programma di installazione e aprire la finestra Proprietà.
2. Scegli la scheda Firme digitali.
3. Dall'elenco delle firme, scegli Amazon Web Services, Inc., quindi scegli Dettagli.
4. Scegli la scheda Generale, se non è già selezionata, quindi scegli Visualizza certificato.
5. Scegli la scheda Dettagli, quindi seleziona Tutto nell'elenco a discesa Mostra, se non è già selezionato.
6. Scorri fino a visualizzare il campo Identificazione personale, quindi scegli Identificazione personale. In questo modo viene visualizzato l'intero valore dell'identificazione personale nella finestra inferiore.

7. Abbina il valore dell'impronta digitale al valore seguente. Se il valore corrisponde, procedi con l'installazione. In caso contrario, rivolgiti al AWS SAM team creando un problema nell'aws-sam-cli GitHub archivio.

```
cd62479397f09d72a04c7399a254b0a91da53d6c
```

Verifica il valore hash

Linux

x86_64 - programma di installazione da riga di comando

Verifica l'integrità e l'autenticità dei file di installazione scaricati generando un valore hash utilizzando il seguente comando:

```
$ sha256sum aws-sam-cli-linux-x86_64.zip
```

L'output dovrebbe essere simile al seguente esempio:

```
<64-character SHA256 hash value> aws-sam-cli-linux-x86_64.zip
```

Confrontate il valore hash SHA-256 a 64 caratteri con quello della AWS SAMCLI versione desiderata nelle note di rilascio su.AWS SAMCLI GitHub

macOS

GUI e programma di installazione a riga di comando

Verifica l'integrità e l'autenticità del programma di installazione scaricato generando un valore hash utilizzando il seguente comando:

```
$ shasum -a 256 path-to-pkg-installer/name-of-pkg-installer

# Examples
$ shasum -a 256 ~/Downloads/aws-sam-cli-macos-arm64.pkg
$ shasum -a 256 ~/Downloads/aws-sam-cli-macos-x86_64.pkg
```

Confrontate il vostro valore hash SHA-256 a 64 caratteri con il valore corrispondente nell'archivio delle note di rilascio.AWS SAMCLI GitHub

Tutorial: Implementa un'applicazione Hello World con AWS SAM

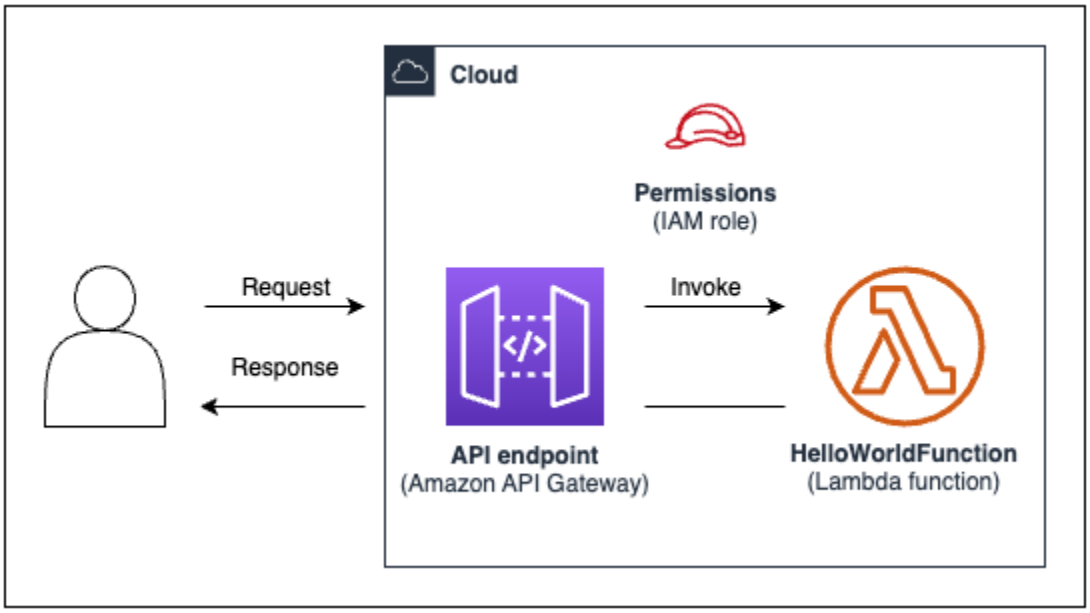
In questo tutorial, si utilizza l'interfaccia a riga di AWS Serverless Application Model comando (AWS SAMCLI) per completare quanto segue:

- Inizializza, crea e distribuisci un'applicazione Hello World di esempio.
- Apporta modifiche locali e sincronizza con. AWS CloudFormation
- Esegui test locali sul tuo host di sviluppo.
- Eliminare l'applicazione di esempio da Cloud AWS.

L'applicazione Hello World di esempio implementa un backend API di base. Consiste delle seguenti risorse:

- Amazon API Gateway: endpoint API che utilizzerai per richiamare la tua funzione.
- AWS Lambda— Funzione che elabora la richiesta HTTP API GET e restituisce un `hello world` messaggio.
- AWS Identity and Access Management Ruolo (IAM): fornisce le autorizzazioni per consentire ai servizi di interagire in modo sicuro.

Il diagramma seguente mostra i componenti di questa applicazione:



Argomenti

- Prerequisiti
- Passaggio 1: inizializza l'applicazione Hello World di esempio
- Fase 2: Crea la tua applicazione
- Passo 3: Distribuisci la tua applicazione su Cloud AWS
- Passaggio 4: Esegui l'applicazione
- Fase 5: Interagite con la vostra funzione nel Cloud AWS
- Fase 6: Modificare e sincronizzare l'applicazione con Cloud AWS
- Passaggio 7: (Facoltativo) Testa l'applicazione localmente
- Fase 8: Eliminare l'applicazione dal Cloud AWS
- Risoluzione dei problemi
- Ulteriori informazioni

Prerequisiti

Verifica di aver completato quanto segue:

- AWS SAM prerequisiti
- Installa il AWS SAMCLI

> **Note**
>
> Se preferisci usare Visual Studio Code per lo sviluppo serverless, consulta Creazione di applicazioni serverless nella AWS Toolkit for VS Code User Guide step-by-step per istruzioni sull'uso dell'interfaccia VS Code.

Passaggio 1: inizializza l'applicazione Hello World di esempio

In questo passaggio, utilizzerai il AWS SAMCLI per creare un esempio di progetto applicativo Hello World sul tuo computer locale.

Per inizializzare l'applicazione Hello World di esempio

1. Nella riga di comando, esegui quanto segue da una directory iniziale a tua scelta:

```
$ sam init
```

> **Note**
>
> Questo comando inizializza l'applicazione serverless e crea una directory di progetto. La directory contiene diversi file e cartelle, incluso il `template.yaml` file, che è il modello. AWS SAM
>
> La versione di Python installata deve corrispondere alla `Runtime` proprietà specificata in. `template.yaml` Per i carichi di lavoro di produzione, consigliamo di utilizzare Python 3.12 o versioni precedenti completamente supportate negli ambienti. AWS Se le versioni di Python non corrispondono, si verificheranno errori di compilazione.
>
> Per risolvere i problemi di compatibilità delle versioni:
>
> - Specificate un runtime compatibile durante l'inizializzazione: `sam init --runtime python3.9`
> - Modifica la `Runtime` proprietà `template.yaml` dopo l'inizializzazione

2. Ti AWS SAMCLI guiderà attraverso l'inizializzazione di una nuova applicazione. Configura quanto segue:

   1. Seleziona Modelli di avvio AWS rapido per scegliere un modello iniziale.
   2. Scegli il modello Hello World Example e scaricalo.
   3. Usa il Python runtime e il tipo di `zip` pacchetto.
   4. Per questo tutorial, disattiva il AWS X-Ray tracciamento. Per saperne di più, vedi Cos'è AWS X-Ray? nella Guida per gli AWS X-Ray sviluppatori.
   5. Per questo tutorial, disattiva il monitoraggio con Amazon CloudWatch Application Insights. Per ulteriori informazioni, consulta Amazon CloudWatch Application Insights nella Amazon CloudWatch User Guide.
   6. Per questo tutorial, disattiva l'impostazione della registrazione strutturata in formato JSON sulle tue funzioni Lambda.
   7. Assegna un nome alla tua applicazione come sam-app.

   Per utilizzare il flusso AWS SAMCLI interattivo:

- Le parentesi (`[ ]`) indicano i valori predefiniti. Lascia vuota la risposta per selezionare il valore predefinito.
- Inserisci **y** per sì e **n** per no.

Di seguito è riportato un esempio di flusso `sam init` interattivo:

```
$ sam init
...
Which template source would you like to use?
    1 - AWS Quick Start Templates
    2 - Custom Template Location
Choice: 1

Choose an AWS Quick Start application template
    1 - Hello World Example
    2 - Data processing
    3 - Hello World Example with Powertools for AWS Lambda
    4 - Multi-step workflow
    5 - Scheduled task
    6 - Standalone function
    7 - Serverless API
    8 - Infrastructure event management
    9 - Lambda Response Streaming
   10 - GraphQLApi Hello World Example
   11 - Full Stack
   12 - Lambda EFS example
   13 - Serverless Connector Hello World Example
   14 - Multi-step workflow with Connectors
   15 - DynamoDB Example
   16 - Machine Learning
Template: 1

Use the most popular runtime and package type? (Python and zip) [y/N]: y

Would you like to enable X-Ray tracing on the function(s) in your application?  [y/
N]: ENTER

Would you like to enable monitoring using CloudWatch Application Insights?
For more info, please view https://docs.aws.amazon.com/AmazonCloudWatch/latest/
monitoring/cloudwatch-application-insights.html [y/N]: ENTER
```

```
Would you like to set Structured Logging in JSON format on your Lambda functions?
 [y/N]: ENTER

Project name [sam-app]: ENTER
```

3. AWS SAMCLIScarica il modello iniziale e crea la struttura di directory del progetto applicativo sul computer locale. Di seguito è riportato un esempio dell' AWS SAMCLIoutput:

```
Cloning from https://github.com/aws/aws-sam-cli-app-templates (process may take a
 moment)

    -----------------------
    Generating application:
    -----------------------
    Name: sam-app
    Runtime: python3.9
    Architectures: x86_64
    Dependency Manager: pip
    Application Template: hello-world
    Output Directory: .
    Configuration file: sam-app/samconfig.toml

    Next steps can be found in the README file at sam-app/README.md


Commands you can use next
=========================
[*] Create pipeline: cd sam-app && sam pipeline init --bootstrap
[*] Validate SAM template: cd sam-app && sam validate
[*] Test Function in the Cloud: cd sam-app && sam sync --stack-name {stack-name} --
watch
```

4. Dalla riga di comando, spostatevi nella `sam-app` directory appena creata. Di seguito è riportato un esempio di ciò che AWS SAMCLI ha creato:

> **Note**
>
> Se `tree` il comando non è installato automaticamente, esegui questo comando: `brew install tree`

```
$ cd sam-app
```

```
$ tree

### README.md
### __init__.py
### events
#   ### event.json
### hello_world
#   ### __init__.py
#   ### app.py
#   ### requirements.txt
### samconfig.toml
### template.yaml
### tests
    ### __init__.py
    ### integration
    #   ### __init__.py
    #   ### test_api_gateway.py
    ### requirements.txt
    ### unit
        ### __init__.py
        ### test_handler.py

6 directories, 14 files
```

Alcuni file importanti da evidenziare:

- `hello_world/app.py`— Contiene il codice della funzione Lambda.
- `hello_world/requirements.txt`— Contiene tutte Python le dipendenze richieste dalla funzione Lambda.
- `samconfig.toml`— File di configurazione per l'applicazione che memorizza i parametri predefiniti utilizzati da. AWS SAMCLI
- `template.yaml`— Il AWS SAM modello che contiene il codice dell'infrastruttura dell'applicazione.

Ora hai un'applicazione serverless completamente creata sul tuo computer locale!

Fase 2: Crea la tua applicazione

In questo passaggio, si utilizza il AWS SAMCLI per creare l'applicazione e prepararla per la distribuzione. Quando compili, AWS SAMCLI crea una `.aws-sam` directory e vi organizza le dipendenze delle funzioni, il codice del progetto e i file di progetto.

Per creare la tua applicazione

- Nella riga di comando, dalla directory del `sam-app` progetto, esegui quanto segue:

```
$ sam build
```

> **Note**
>
> Se non lo hai Python sul tuo computer locale, usa invece il sam build --use-container comando. AWS SAMCLICreerà un Docker contenitore che include il runtime e le dipendenze della funzione. Questo comando richiede la Docker presenza sul computer locale. Per l'installazioneDocker, vedereInstallazione di Docker.

Di seguito è riportato un esempio dell' AWS SAMCLIoutput:

```
$ sam build
Starting Build use cache
Manifest file is changed (new hash: 3298f1304...d4d421) or dependency folder (.aws-
sam/deps/4d3dfad6-a267-47a6-a6cd-e07d6fae318c) is missing for (HelloWorldFunction),
 downloading dependencies and copying/building source
Building codeuri: /Users/.../Demo/sam-tutorial1/sam-app/hello_world runtime:
 python3.9 metadata: {} architecture: x86_64 functions: HelloWorldFunction
Running PythonPipBuilder:CleanUp
Running PythonPipBuilder:ResolveDependencies
Running PythonPipBuilder:CopySource
Running PythonPipBuilder:CopySource

Build Succeeded

Built Artifacts  : .aws-sam/build
Built Template   : .aws-sam/build/template.yaml

Commands you can use next
```

```
=========================
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name {{stack-name}} --watch
[*] Deploy: sam deploy --guided
```

Di seguito è riportato un esempio abbreviato della `.aws-sam` directory creata dalla AWS SAM CLI:

```
.aws-sam
### build
#   ### HelloWorldFunction
#   #   ### __init__.py
#   #   ### app.py
#   #   ### requirements.txt
#   ### template.yaml
### build.toml
```

Alcuni file importanti da evidenziare:

- `build/HelloWorldFunction`— Contiene il codice della funzione Lambda e le dipendenze. AWS SAMCLICrea una directory per ogni funzione dell'applicazione.
- `build/template.yaml`— Contiene una copia del AWS SAM modello a cui fa riferimento CloudFormation durante la distribuzione.
- `build.toml`— File di configurazione che memorizza i valori dei parametri predefiniti a cui fa riferimento AWS SAMCLI durante la creazione e la distribuzione dell'applicazione.

Ora sei pronto per distribuire la tua applicazione su. Cloud AWS

Passo 3: Distribuisci la tua applicazione su Cloud AWS

> **Note**
>
> Questo passaggio richiede la configurazione AWS delle credenziali. Per ulteriori informazioni, consulta Passaggio 5: Utilizzare AWS CLI per configurare le AWS credenziali in AWS SAM prerequisiti.

In questo passaggio, si utilizza AWS SAMCLI per distribuire l'applicazione su. Cloud AWS AWS SAMCLIFaranno quanto segue:

- Guida l'utente nella configurazione delle impostazioni dell'applicazione per la distribuzione.
- Carica i file dell'applicazione su Amazon Simple Storage Service (Amazon S3).
- Trasforma il tuo AWS SAM modello in un CloudFormation modello. Quindi carica il modello sul CloudFormation servizio per fornire AWS le tue risorse.

Per distribuire l'applicazione su

1. Nella riga di comando, dalla directory del `sam-app` progetto, esegui quanto segue:

```
$ sam deploy --guided
```

2. Segui il flusso AWS SAMCLI interattivo per configurare le impostazioni dell'applicazione. Configura quanto segue:

    1. Il nome CloudFormation dello stack: uno stack è una raccolta di AWS risorse che puoi gestire come singola unità. Per ulteriori informazioni, consulta Lavorare con gli stack nella Guida per l'AWS CloudFormation utente.
    2. Il Regione AWSsu cui distribuire lo CloudFormation stack. Per ulteriori informazioni, consulta Endpoint CloudFormation nella Guida per l'utente di AWS CloudFormation .
    3. Per questo tutorial, disattiva la conferma delle modifiche prima della distribuzione.
    4. Consenti la creazione di ruoli IAM: consente di AWS SAM creare il ruolo IAM necessario per l'interazione tra la risorsa API Gateway e la risorsa della funzione Lambda.
    5. Per questo tutorial, disattiva la disabilitazione del rollback.
    6. Consenti HelloWorldFunction senza autorizzazione definita: questo messaggio viene visualizzato perché l'endpoint API Gateway è configurato per essere accessibile pubblicamente, senza autorizzazione. Poiché questa è la configurazione prevista per l'applicazione Hello World, consenti AWS SAMCLI di continuare. Per ulteriori informazioni sulla configurazione dell'autorizzazione, consultaControlla l'accesso alle API con il tuo AWS SAM modello.
    7. Salva gli argomenti nel file di configurazione: questo aggiornerà il `samconfig.toml` file dell'applicazione con le preferenze di distribuzione.
    8. Seleziona il nome del file di configurazione predefinito.
    9. Seleziona l'ambiente di configurazione predefinito.

Di seguito è riportato un esempio di output del flusso `sam deploy --guided` interattivo:

```
$ sam deploy --guided

Configuring SAM deploy
======================

    Looking for config file [samconfig.toml] :  Found
    Reading default arguments  :  Success

    Setting default arguments for 'sam deploy'
    =========================================
    Stack Name [sam-app]: ENTER
    AWS Region [us-west-2]: ENTER
    #Shows you resources changes to be deployed and require a 'Y' to initiate
 deploy
    Confirm changes before deploy [Y/n]: n
    #SAM needs permission to be able to create roles to connect to the resources in
 your template
    Allow SAM CLI IAM role creation [Y/n]: ENTER
    #Preserves the state of previously provisioned resources when an operation
 fails
    Disable rollback [y/N]: ENTER
    HelloWorldFunction may not have authorization defined, Is this okay? [y/N]: y
    Save arguments to configuration file [Y/n]: ENTER
    SAM configuration file [samconfig.toml]: ENTER
    SAM configuration environment [default]: ENTER
```

3. AWS SAMCLIImplementa l'applicazione effettuando le seguenti operazioni:

   - Quindi AWS SAMCLI crea un bucket Amazon S3 e carica la tua directory. `.aws-sam`
   - AWS SAMCLITrasforma il AWS SAM modello in CloudFormation e lo carica sul servizio. CloudFormation
   - CloudFormation fornisce le tue risorse.

   Durante la distribuzione, AWS SAMCLI mostra i tuoi progressi. Di seguito è riportato un esempio di output:

```
Looking for resources needed for deployment:
```

```
    Managed S3 bucket: aws-sam-cli-managed-default-samclisam-s3-demo-
bucket-1a4x26zbcdkqr
    A different default S3 bucket can be set in samconfig.toml

    Parameter "stack_name=sam-app" in [default.deploy.parameters] is defined as a
 global parameter [default.global.parameters].
    This parameter will be only saved under [default.global.parameters] in /
Users/.../Demo/sam-tutorial1/sam-app/samconfig.toml.

    Saved arguments to config file
    Running 'sam deploy' for future deployments will use the parameters saved
 above.
    The above parameters can be changed by modifying samconfig.toml
    Learn more about samconfig.toml syntax at
    https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/
serverless-sam-cli-config.html

File with same data already exists at sam-app/da3c598813f1c2151579b73ad788cac8,
 skipping upload

    Deploying with following values
    ===============================
    Stack name                   : sam-app
    Region                       : us-west-2
    Confirm changeset            : False
    Disable rollback             : False
    Deployment s3 bucket         : aws-sam-cli-managed-default-samclisam-s3-demo-
bucket-1a4x26zbcdkqr
    Capabilities                 : ["CAPABILITY_IAM"]
    Parameter overrides          : {}
    Signing Profiles             : {}

Initiating deployment
=====================

File with same data already exists at sam-
app/2bebf67c79f6a743cc5312f6dfc1efee.template, skipping upload


Waiting for changeset to be created..

CloudFormation stack changeset
-----------------------------------------------------------------------------------------------
```

```
Operation                           LogicalResourceId
 ResourceType                        Replacement
-------------------------------------------------------------------------------------------
* Modify                             HelloWorldFunction
 AWS::Lambda::Function               False
* Modify                             ServerlessRestApi
 AWS::ApiGateway::RestApi            False
- Delete                             AwsSamAutoDependencyLayerNestedSt
 AWS::CloudFormation::Stack          N/A
                                     ack
-------------------------------------------------------------------------------------------



Changeset created successfully. arn:aws:cloudformation:us-
west-2:012345678910:changeSet/samcli-deploy1678917603/22e05525-08f9-4c52-a2c4-
f7f1fd055072


2023-03-15 12:00:16 - Waiting for stack create/update to complete

CloudFormation events from stack operations (refresh every 0.5 seconds)
-------------------------------------------------------------------------------------------
ResourceStatus                      ResourceType
 LogicalResourceId                   ResourceStatusReason
-------------------------------------------------------------------------------------------
UPDATE_IN_PROGRESS                  AWS::Lambda::Function
 HelloWorldFunction                  -
UPDATE_COMPLETE                     AWS::Lambda::Function
 HelloWorldFunction                  -
UPDATE_COMPLETE_CLEANUP_IN_PROGRE   AWS::CloudFormation::Stack          sam-app
                          -
SS
DELETE_IN_PROGRESS                  AWS::CloudFormation::Stack
 AwsSamAutoDependencyLayerNestedSt   -
                                                                        ack
DELETE_COMPLETE                     AWS::CloudFormation::Stack
 AwsSamAutoDependencyLayerNestedSt   -
                                                                        ack
UPDATE_COMPLETE                     AWS::CloudFormation::Stack          sam-app
                          -
-------------------------------------------------------------------------------------------

CloudFormation outputs from deployed stack
-------------------------------------------------------------------------------------------
```

```
Outputs
-----------------------------------------------------------------------------------------------
Key                 HelloWorldFunctionIamRole
Description         Implicit IAM Role created for Hello World function
Value               arn:aws:iam::012345678910:role/sam-app-
HelloWorldFunctionRole-15GLOUR9LMT1W

Key                 HelloWorldApi
Description         API Gateway endpoint URL for Prod stage for Hello World
 function
Value               https://<restapiid>.execute-api.us-west-2.amazonaws.com/Prod/
hello/

Key                 HelloWorldFunction
Description         Hello World Lambda Function ARN
Value               arn:aws:lambda:us-west-2:012345678910:function:sam-app-
HelloWorldFunction-yQDNe17r9maD
-----------------------------------------------------------------------------------------------



Successfully created/updated stack - sam-app in us-west-2
```

L'applicazione è ora distribuita e in esecuzione in! Cloud AWS

Passaggio 4: Esegui l'applicazione

In questo passaggio, invierai una richiesta GET all'endpoint API e vedrai l'output della funzione Lambda.

Per ottenere il valore dell'endpoint dell'API

1. Dalle informazioni visualizzate AWS SAMCLI nel passaggio precedente, individua la `Outputs` sezione. In questa sezione, individua la `HelloWorldApi` risorsa per trovare il valore dell'endpoint HTTP. Di seguito è riportato un esempio di output:

```
-----------------------------------------------------------------------------------------------
Outputs
-----------------------------------------------------------------------------------------------
...
Key                 HelloWorldApi
Description         API Gateway endpoint URL for Prod stage for Hello World
 function
```

```
Value               https://ets1gv8lxi.execute-api.us-west-2.amazonaws.com/Prod/
hello/
...
-----------------------------------------------------------------------------------------------
```

2. In alternativa, è possibile utilizzare il sam list endpoints --output json comando per ottenere queste informazioni. Di seguito è riportato un esempio di output:

```
$ sam list endpoints --output json
2023-03-15 12:39:19 Loading policies from IAM...
2023-03-15 12:39:25 Finished loading policies from IAM.
[
  {
    "LogicalResourceId": "HelloWorldFunction",
    "PhysicalResourceId": "sam-app-HelloWorldFunction-yQDNe17r9maD",
    "CloudEndpoint": "-",
    "Methods": "-"
  },
  {
    "LogicalResourceId": "ServerlessRestApi",
    "PhysicalResourceId": "ets1gv8lxi",
    "CloudEndpoint": [
      "https://ets1gv8lxi.execute-api.us-west-2.amazonaws.com/Prod",
      "https://ets1gv8lxi.execute-api.us-west-2.amazonaws.com/Stage"
    ],
    "Methods": [
      "/hello['get']"
    ]
  }
]
```

Per richiamare la tua funzione

- Utilizzando il browser o la riga di comando, invia una richiesta GET all'endpoint API. Di seguito è riportato un esempio di utilizzo del comando curl:

```
$ curl https://ets1gv8lxi.execute-api.us-west-2.amazonaws.com/Prod/hello/
{"message": "hello world"}
```

Fase 5: Interagite con la vostra funzione nel Cloud AWS

In questo passaggio, si utilizza AWS SAMCLI per richiamare la funzione Lambda in. Cloud AWS

Per richiamare la funzione Lambda nel cloud

1. Prendi nota delle tue funzioni del `LogicalResourceId` passaggio precedente. Dovrebbe essere `HelloWorldFunction`.
2. Nella riga di comando, dalla directory del `sam-app` progetto, esegui quanto segue:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app
```

3. AWS SAMCLIrichiama la tua funzione nel cloud e restituisce una risposta. Di seguito è riportato un esempio di output:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app

Invoking Lambda Function HelloWorldFunction
START RequestId: d5ef494b-5f45-4086-86fd-d7322fa1a1f9 Version: $LATEST
END RequestId: d5ef494b-5f45-4086-86fd-d7322fa1a1f9
REPORT RequestId: d5ef494b-5f45-4086-86fd-d7322fa1a1f9  Duration: 6.62 ms
 Billed Duration: 7 ms     Memory Size: 128 MB     Max Memory Used: 67 MB  Init
 Duration: 164.06 ms
{"statusCode":200,"body":"{\"message\":\"hello world\"}"}%
```

Fase 6: Modificare e sincronizzare l'applicazione con Cloud AWS

In questo passaggio, si utilizza il AWS SAMCLI sam sync --watch comando per sincronizzare le modifiche locali a Cloud AWS.

Per usare sam sync

1. Nella riga di comando, dalla directory del `sam-app` progetto, esegui quanto segue:

```
$ sam sync --watch
```

2. Ti AWS SAMCLI chiede di confermare che stai sincronizzando uno stack di sviluppo. Poiché il sam sync --watch comando distribuisce automaticamente le modifiche locali Cloud AWS in tempo reale, lo consigliamo solo per gli ambienti di sviluppo.

AWS SAMCLIEsegue una distribuzione iniziale prima di iniziare il monitoraggio delle modifiche locali. Di seguito è riportato un esempio di output:

```
$ sam sync --watch
The SAM CLI will use the AWS Lambda, Amazon API Gateway, and AWS StepFunctions APIs
 to upload your code without
performing a CloudFormation deployment. This will cause drift in your
 CloudFormation stack.
**The sync command should only be used against a development stack**.

Confirm that you are synchronizing a development stack.

Enter Y to proceed with the command, or enter N to cancel:
 [Y/n]: y
Queued infra sync. Waiting for in progress code syncs to complete...
Starting infra sync.
Manifest is not changed for (HelloWorldFunction), running incremental build
Building codeuri: /Users/.../Demo/sam-tutorial1/sam-app/hello_world runtime:
 python3.9 metadata: {} architecture: x86_64 functions: HelloWorldFunction
Running PythonPipBuilder:CopySource

Build Succeeded

Successfully packaged artifacts and wrote output template to file /var/
folders/45/5ct135bx3fn2551_ptl5g6_80000gr/T/tmpq3x9vh63.
Execute the following command to deploy the packaged template
sam deploy --template-file /var/folders/45/5ct135bx3fn2551_ptl5g6_80000gr/T/
tmpq3x9vh63 --stack-name <YOUR STACK NAME>


    Deploying with following values
    ===============================
    Stack name                   : sam-app
    Region                       : us-west-2
    Disable rollback             : False
    Deployment s3 bucket         : aws-sam-cli-managed-default-samclisam-s3-demo-
bucket-1a4x26zbcdkqr
    Capabilities                 : ["CAPABILITY_NAMED_IAM",
 "CAPABILITY_AUTO_EXPAND"]
    Parameter overrides          : {}
    Signing Profiles             : null
```

```
Initiating deployment
=====================


2023-03-15 13:10:05 - Waiting for stack create/update to complete

CloudFormation events from stack operations (refresh every 0.5 seconds)
---------------------------------------------------------------------------------------------
ResourceStatus                      ResourceType
 LogicalResourceId                   ResourceStatusReason
---------------------------------------------------------------------------------------------
UPDATE_IN_PROGRESS                  AWS::CloudFormation::Stack          sam-app
                         Transformation succeeded
CREATE_IN_PROGRESS                  AWS::CloudFormation::Stack
 AwsSamAutoDependencyLayerNestedSt   -
                                                                         ack
CREATE_IN_PROGRESS                  AWS::CloudFormation::Stack
 AwsSamAutoDependencyLayerNestedSt   Resource creation Initiated
                                                                         ack
CREATE_COMPLETE                     AWS::CloudFormation::Stack
 AwsSamAutoDependencyLayerNestedSt   -
                                                                         ack
UPDATE_IN_PROGRESS                  AWS::Lambda::Function
 HelloWorldFunction                   -
UPDATE_COMPLETE                     AWS::Lambda::Function
 HelloWorldFunction                   -
UPDATE_COMPLETE_CLEANUP_IN_PROGRE   AWS::CloudFormation::Stack          sam-app
                         -
SS
UPDATE_COMPLETE                     AWS::CloudFormation::Stack          sam-app
                         -
---------------------------------------------------------------------------------------------

CloudFormation outputs from deployed stack
---------------------------------------------------------------------------------------------
Outputs
---------------------------------------------------------------------------------------------
Key                 HelloWorldFunctionIamRole
Description         Implicit IAM Role created for Hello World function
Value               arn:aws:iam::012345678910:role/sam-app-
HelloWorldFunctionRole-15GLOUR9LMT1W

Key                 HelloWorldApi
```

```
Description         API Gateway endpoint URL for Prod stage for Hello World
 function
Value               https://ets1gv8lxi.execute-api.us-west-2.amazonaws.com/Prod/
hello/

Key                 HelloWorldFunction
Description         Hello World Lambda Function ARN
Value               arn:aws:lambda:us-west-2:012345678910:function:sam-app-
HelloWorldFunction-yQDNe17r9maD
-----------------------------------------------------------------------------------------



Stack update succeeded. Sync infra completed.

Infra sync completed.
CodeTrigger not created as CodeUri or DefinitionUri is missing for
 ServerlessRestApi.
```

Successivamente, modificherai il codice della funzione Lambda. AWS SAMCLIRileverà automaticamente questa modifica e sincronizzerà l'applicazione con. Cloud AWS

Per modificare e sincronizzare l'applicazione

1. Apri il `sam-app/hello_world/app.py` file nel tuo IDE preferito.
2. Modifica `message` e salva il file. Di seguito è riportato un esempio:

```
import json
...
def lambda_handler(event, context):
    ...
    return {
        "statusCode": 200,
        "body": json.dumps({
            "message": "hello everyone!",
            ...
        }),
    }
```

3. AWS SAMCLIRileva la modifica e sincronizza l'applicazione con. Cloud AWS Di seguito è riportato un esempio di output:

```
Syncing Lambda Function HelloWorldFunction...
Manifest is not changed for (HelloWorldFunction), running incremental build
Building codeuri: /Users/.../Demo/sam-tutorial1/sam-app/hello_world runtime:
 python3.9 metadata: {} architecture: x86_64 functions: HelloWorldFunction
Running PythonPipBuilder:CopySource
Finished syncing Lambda Function HelloWorldFunction.
```

4. Per verificare la modifica, invia nuovamente una richiesta GET all'endpoint API.

```
$ curl https://ets1gv8lxi.execute-api.us-west-2.amazonaws.com/Prod/hello/
{"message": "hello everyone!"}
```

Passaggio 7: (Facoltativo) Testa l'applicazione localmente

> **Note**
>
> Questa fase è facoltativa.

> **Important**
>
> Questo passaggio richiede un Docker computer locale. È necessario averlo Docker installato e configurato per utilizzarlo AWS SAMCLI per i test locali. Per ulteriori informazioni, consulta Installazione di Docker.

In questo passaggio, si utilizza il AWS SAMCLI sam local comando per testare l'applicazione localmente. A tale scopo, AWS SAMCLI crea un ambiente locale utilizzandoDocker. Questo ambiente locale emula l'ambiente di esecuzione basato sul cloud della funzione Lambda.

Si eseguiranno le operazioni indicate di seguito.

1. Crea un ambiente locale per la tua funzione Lambda e invocala.
2. Ospita l'endpoint dell'API HTTP localmente e usalo per richiamare la funzione Lambda.

Per richiamare la funzione Lambda localmente

1. Nella riga di comando, dalla directory del `sam-app` progetto, esegui quanto segue:

```
$ sam local invoke
```

2. AWS SAMCLICrea un Docker contenitore locale e richiama la tua funzione. Di seguito è riportato un esempio di output:

```
$ sam local invoke
Invoking app.lambda_handler (python3.9)
Local image was not found.
Removing rapid images for repo public.ecr.aws/sam/emulation-python3.9
Building image.....................
Using local image: public.ecr.aws/lambda/python:3.9-rapid-x86_64.

Mounting /Users/.../Demo/sam-tutorial1/sam-app/.aws-sam/build/HelloWorldFunction
 as /var/task:ro,delegated inside runtime container
START RequestId: b046db01-2a00-415d-af97-35f3a02e9eb6 Version: $LATEST
END RequestId: b046db01-2a00-415d-af97-35f3a02e9eb6
REPORT RequestId: b046db01-2a00-415d-af97-35f3a02e9eb6    Init Duration: 1.01 ms
    Duration: 633.45 ms    Billed Duration: 634 ms    Memory Size: 128 MB    Max
 Memory Used: 128 MB
{"statusCode": 200, "body": "{\"message\": \"hello world\"}"}
```

Per ospitare la tua API localmente

1. Nella riga di comando, dalla directory del `sam-app` progetto, esegui quanto segue:

```
$ sam local start-api
```

2. AWS SAMCLICrea un Docker contenitore locale per la tua funzione Lambda e crea un server HTTP locale per simulare l'endpoint dell'API. Di seguito è riportato un esempio di output:

```
$ sam local start-api
Initializing the lambda functions containers.
Local image is up-to-date
Using local image: public.ecr.aws/lambda/python:3.9-rapid-x86_64.

Mounting /Users/.../Demo/sam-tutorial1/sam-app/.aws-sam/build/HelloWorldFunction
 as /var/task:ro,delegated inside runtime container
```

```
Containers Initialization is done.
Mounting HelloWorldFunction at http://127.0.0.1:3000/hello [GET]
You can now browse to the above endpoints to invoke your functions. You do not
 need to restart/reload SAM CLI while working on your functions, changes will be
 reflected instantly/automatically. If you used sam build before running local
 commands, you will need to re-run sam build for the changes to be picked up. You
 only need to restart AWS SAM CLI if you update your AWS SAM template
2023-03-15 14:25:21 WARNING: This is a development server. Do not use it in a
 production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:3000
2023-03-15 14:25:21 Press CTRL+C to quit
```

3. Utilizzando il browser o la riga di comando, inviate una richiesta GET all'endpoint API locale. Di seguito è riportato un esempio di utilizzo del comando curl:

```
$ curl http://127.0.0.1:3000/hello
{"message": "hello world"}
```

Fase 8: Eliminare l'applicazione dal Cloud AWS

In questo passaggio, si utilizza il AWS SAMCLI sam delete comando per eliminare l'applicazione da Cloud AWS.

Per eliminare l'applicazione da Cloud AWS

1. Nella riga di comando, dalla directory del `sam-app` progetto, esegui quanto segue:

```
$ sam delete
```

2. Ti AWS SAMCLI chiederà di confermare. Quindi, eliminerà il bucket e lo stack Amazon S3 dell'applicazione. CloudFormation Di seguito è riportato un esempio di output:

```
$ sam delete
    Are you sure you want to delete the stack sam-app in the region us-west-2 ? [y/
N]: y
    Are you sure you want to delete the folder sam-app in S3 which contains the
 artifacts? [y/N]: y
    - Deleting S3 object with key c6ce8fa8b5a97dd022ecd006536eb5a4
    - Deleting S3 object with key 5d513a459d062d644f3b7dd0c8b56a2a.template
    - Deleting S3 object with key sam-app/2bebf67c79f6a743cc5312f6dfc1efee.template
    - Deleting S3 object with key sam-app/6b208d0e42ad15d1cee77d967834784b.template
```

```
    - Deleting S3 object with key sam-app/da3c598813f1c2151579b73ad788cac8
    - Deleting S3 object with key sam-app/f798cdd93cee188a71d120f14a035b11
    - Deleting Cloudformation stack sam-app

Deleted successfully
```

Risoluzione dei problemi

Per risolvere il problema AWS SAMCLI, vedere. AWS SAMCLIrisoluzione dei problemi

Ulteriori informazioni

Per continuare a saperne di più AWS SAM, consulta le seguenti risorse:

- Il AWS SAM workshop completo: un workshop progettato per insegnarti molte delle principali funzionalità che AWS SAM offre.
- Sessioni con SAM — Serie di video creata dal nostro team di AWS Serverless Developer Advocate sull'utilizzo. AWS SAM
- Serverless Land: sito che riunisce le informazioni, i blog, i video, il codice e le risorse di apprendimento più recenti per la tecnologia serverless. AWS

Conversione delle funzioni Lambda in applicazioni AWS SAM

Se hai una funzione nella console Lambda che desideri gestire in VS Code come infrastructure as code (IaC), puoi trasferirla su VS Code e convertirla in un modello. AWS SAM Con la funzione convertita in un AWS SAM modello, puoi controllare il controllo delle versioni dell'infrastruttura, automatizzare le distribuzioni, eseguire il debug delle funzioni in remoto e mantenere ambienti coerenti per tutto il ciclo di vita dello sviluppo. Con le funzionalità di debug locale e remoto, puoi anche risolvere i problemi in modo più efficace esaminando il codice, ispezionando le variabili e impostando i punti di interruzione sia localmente che nel cloud. AWS

Per istruzioni su come spostare le funzioni dalla console e configurare lo sviluppo locale, consulta Sviluppare le funzioni Lambda localmente con VS Code nella AWS Lambda Developer Guide. Segui queste istruzioni per spostare una funzione Lambda dalla console a VS Code e convertirla in un AWS SAM modello. Dopo averlo fatto, puoi:

- Esegui il debug remoto della tua funzione nel AWS cloud. Per i dettagli, consulta Eseguire il debug remoto delle funzioni Lambda con VS Code nella Guida per gli sviluppatori.AWS Lambda

- Gestisci le risorse e le autorizzazioni associate come infrastruttura come codice. Per i dettagli, consulta Using Lambda with infrastructure as code (IaC) nella Developer Guide AWS Lambda
- Distribuisci e aggiorna la tua applicazione utilizzando. AWS SAMCLI Per informazioni dettagliate, vedi AWS SAM CLI.
- Progetta visivamente la tua architettura serverless utilizzando Infrastructure Composer. Per i dettagli, consulta la Infrastructure Composer Developer Guide.

Ulteriori informazioni

Per continuare a saperne di più AWS SAM, consulta le seguenti risorse:

- Il AWS SAM workshop completo: un workshop progettato per insegnarti molte delle principali funzionalità che AWS SAM offre.
- Sessioni con SAM — Serie di video creata dal nostro team di AWS Serverless Developer Advocate sull'utilizzo. AWS SAM
- Serverless Land: sito che riunisce le informazioni, i blog, i video, il codice e le risorse di apprendimento più recenti per la tecnologia serverless. AWS

Come usare AWS Serverless Application Model (AWS SAM)

Gli strumenti principali utilizzati per sviluppare l'applicazione sono il AWS SAM modello AWS SAMCLIe il AWS SAM progetto (che è la directory dei progetti dell'applicazione). Utilizzate questi strumenti per:

1. Sviluppa la tua applicazione(ciò include l'inizializzazione dell'applicazione, la definizione delle risorse e la creazione dell'applicazione).
2. Testa la tua applicazione.
3. Esegui il debug della tua applicazione.
4. Implementa la tua applicazione e le tue risorse.
5. Monitora la tua applicazione.

AWS SAM crea il AWS SAM progetto dopo aver eseguito il sam init comando e completato il flusso di lavoro successivo. Definisci la tua applicazione serverless aggiungendo codice al tuo AWS SAM progetto. Sebbene il AWS SAM progetto sia costituito da un insieme di file e cartelle, il file più importante al suo interno è il AWS SAM modello (denominato`template.yaml`). In questo modello, scrivi il codice per esprimere risorse, mappature delle sorgenti degli eventi e altre proprietà che definiscono la tua applicazione serverless.

AWS SAMCLIContiene un archivio di comandi che usi nel tuo progetto. AWS SAM Più specificamente, AWS SAMCLI è ciò che usi per creare, trasformare, distribuire, eseguire il debug, impacchettare, inizializzare e sincronizzare il progetto. AWS SAM In altre parole, è ciò che usi per trasformare il tuo AWS SAM progetto in un'applicazione serverless.

Per maggiori dettagli su questi strumenti, consulta i seguenti capitoli:

AWS SAMCLI- Scopri di più su AWS SAMCLI e come utilizzarlo per creare ed eseguire applicazioni serverless.

AWS SAM progetto e AWS SAM modello: scopri la struttura del AWS SAM progetto e come utilizzare il AWS SAM modello per definire un'applicazione serverless.

AWS SAM CLI

AWS Serverless Application Model Command Line Interface (AWS SAMCLI) è uno strumento a riga di comando per lo sviluppo e il test locali di applicazioni serverless. Ti AWS SAMCLI consente di creare, trasformare, distribuire, eseguire il debug, impacchettare, inizializzare e sincronizzare le tue applicazioni serverless localmente prima di distribuirle nel cloud.

AWS SAMCLIfunziona con applicazioni serverless definite utilizzando diversi framework e strumenti Infrastructure as Code (IaC), con diversi livelli di supporto:

- AWS SAM modelli: fornisce supporto nativo con il set completo di funzionalità, tra cui funzionalità locali di test, debug, creazione di pacchetti e implementazione.
- AWS Applicazioni CDK: supporta il test locale delle funzioni Lambda dopo aver sintetizzato l'applicazione CloudFormation CDK in modelli AWS utilizzando il comando cdk synth.
- CloudFormation modelli — Offre una compatibilità diretta perché AWS SAM estende CloudFormation, supporta risorse serverless definite nei modelli standard. CloudFormation
- Applicazioni Terraform: forniscono un supporto limitato per la creazione e il test locale delle funzioni Lambda. Richiede la generazione di artefatti AWS SAM modello che rappresentino le funzioni Lambda definite da Terraform.

Per il supporto più completo delle funzionalità e un'esperienza di sviluppo semplificata, consigliamo di utilizzare modelli nativi. AWS SAM

Argomenti

- Come vengono AWS SAMCLI documentati i comandi
- Configurazione del AWS SAMCLI
- AWS SAMCLIcomandi principali
- Test locali con AWS SAMCLI

Come vengono AWS SAMCLI documentati i comandi

AWS SAMCLIi comandi sono documentati utilizzando il seguente formato:

- Richiesta: per impostazione predefinita, la Linux richiesta è documentata e viene visualizzata come (). `$` Per i comandi Windows specifici, (`>`) viene utilizzato come prompt. Non includere il prompt quando digiti i comandi.
- Directory: quando i comandi devono essere eseguiti da una directory specifica, il nome della directory viene visualizzato prima del simbolo del prompt.
- Input utente: il testo del comando da immettere nella riga di comando viene formattato come **`user input`**.
- Testo sostituibile: il testo variabile, ad esempio i nomi di file e i parametri, viene formattato come. *`replaceable text`* Nei comandi a più righe o nei comandi in cui è richiesto un input specifico da tastiera, l'input da tastiera può essere visualizzato anche come testo sostituibile. Ad esempio, *`ENTER`*.
- Output: l'output restituito come risposta al comando viene formattato come. `computer output`

Il `sam deploy` comando e l'output seguenti sono un esempio:

```
$ sam deploy --guided --template template.yaml

Configuring SAM deploy
======================

    Looking for config file [samconfig.toml] :  Found
    Reading default arguments  :  Success

    Setting default arguments for 'sam deploy'
    =========================================
    Stack Name [sam-app]: ENTER
    AWS Region [us-west-2]: ENTER
    #Shows you resources changes to be deployed and require a 'Y' to initiate deploy
    Confirm changes before deploy [y/N]: ENTER
    #SAM needs permission to be able to create roles to connect to the resources in
 your template
    Allow SAM CLI IAM role creation [Y/n]: ENTER
    #Preserves the state of previously provisioned resources when an operation fails
    Disable rollback [y/N]: ENTER
    HelloWorldFunction may not have authorization defined, Is this okay? [y/N]: y
    Save arguments to configuration file [Y/n]: ENTER
    SAM configuration file [samconfig.toml]: ENTER
    SAM configuration environment [default]: ENTER
```

1. `sam deploy --guided --template template.yaml`è il comando immesso nella riga di comando.
2. **`sam deploy --guided --template`**deve essere fornito così com'è.
3. *`template.yaml`*può essere sostituito con il nome di file specifico.
4. L'output inizia da`Configuring SAM deploy`.
5. Nell'output, *`ENTER`* e *`y`* indica i valori sostituibili che fornisci.

Configurazione del AWS SAMCLI

Uno dei vantaggi AWS SAM è che ottimizza il tempo dello sviluppatore eliminando le attività ripetitive. AWS SAMCLIinclude un file di configurazione denominato a questo `samconfig` scopo. Per impostazione predefinita, non AWS SAMCLI è necessaria alcuna configurazione, ma è possibile aggiornare il file di configurazione per consentire di eseguire comandi con un minor numero di parametri, consentendo invece di fare riferimento AWS SAM ai parametri personalizzati nel file di configurazione. Gli esempi nella tabella seguente mostrano come ottimizzare i comandi:

Originale	Ottimizzato con **`samconfig`**
sam build --cached --parallel --use-containers	sam build
sam local invoke --env-vars locals.json	sam local invoke
sam local start-api --env-vars locals.json --warm-containers EAGER	sam local start-api

AWS SAMCLIFornisce una serie di comandi per aiutare gli sviluppatori a creare, sviluppare e distribuire applicazioni serverless. Ciascuno di questi comandi è configurabile con flag opzionali in base alle preferenze dell'applicazione e dello sviluppatore. Per ulteriori informazioni, consulta il contenuto in AWS SAMCLI GitHub

Gli argomenti di questa sezione mostrano come creare AWS SAMCLIfile di configurazione e personalizzare le impostazioni predefinite per ottimizzare i tempi di sviluppo dell'applicazione serverless.

Argomenti

- Come creare il file di configurazione (il samconfig file)

- Configura le impostazioni del progetto
- Configurare le credenziali e le impostazioni di base

Come creare il file di configurazione (il **`samconfig`** file)

Il file di AWS SAMCLI configurazione (filename`samconfig`) è un file di testo che in genere utilizza la struttura TOML, ma può anche essere in YAML. Quando si utilizza un modello AWS Quick Start, questo file viene creato quando si esegue il comando. sam init È possibile aggiornare questo file quando si distribuisce un'applicazione utilizzando il sam deploy -\-guided comando.

Una volta completata la distribuzione, il `samconfig` file contiene un profilo denominato `default` se sono stati utilizzati i valori predefiniti. Quando si esegue nuovamente il deploy comando, AWS SAM applica le impostazioni di configurazione memorizzate da questo profilo.

Il vantaggio del `samconfig` file è che AWS SAM memorizza le impostazioni di configurazione per tutti gli altri comandi disponibili oltre al comando deploy. Oltre a questi valori creati durante una nuova distribuzione, nel `samconfig` file è possibile impostare una serie di attributi che possono semplificare altri aspetti del flusso di lavoro degli sviluppatori. AWS SAMCLI

Configura le impostazioni del progetto

È possibile specificare impostazioni specifiche del progetto, ad esempio i valori dei parametri dei AWS SAMCLI comandi, in un file di configurazione da utilizzare con. AWS SAMCLI Per ulteriori informazioni su questo file di configurazione, vedere. AWS SAMCLIfile di configurazione

Utilizzo dei file di configurazione

I file di configurazione sono strutturati per ambiente, comando e valore dei parametri. Per ulteriori informazioni, consulta Nozioni di base sui file di configurazione.

Per configurare un nuovo ambiente

1. Specificate il nuovo ambiente nel file di configurazione.

   Di seguito è riportato un esempio di specificazione di un nuovo `prod` ambiente:

   TOML

   ```
   [prod.global.parameters]
   ```

YAML

```
prod:
  global:
    parameters:
```

2. Specificate i valori dei parametri come coppie chiave-valore nella sezione parametri del file di configurazione.

   Di seguito è riportato un esempio di specificazione del nome dello stack dell'applicazione per l'ambiente. `prod`

   TOML

   ```
   [prod.global.parameters]
   stack_name = "prod-app"
   ```

   YAML

   ```
   prod:
     global:
       parameters:
         stack_name: prod-app
   ```

3. Usa l'`--config-env`opzione per specificare l'ambiente da usare.

   Di seguito è riportato un esempio:

   ```
   $ sam deploy --config-env "prod"
   ```

Per configurare i valori dei parametri

1. Specificate il AWS SAMCLI comando per cui desiderate configurare i valori dei parametri. Per configurare i valori dei parametri per tutti AWS SAMCLI i comandi, usa l'`global`identificatore.

   Di seguito è riportato un esempio di specificazione dei valori dei parametri per il comando `default` dell'ambiente: `sam deploy`

TOML

```
[default.deploy.parameters]
confirm_changeset = true
```

YAML

```
default:
  deploy:
    parameters:
      confirm_changeset: true
```

Di seguito è riportato un esempio di specificazione dei valori dei parametri per tutti i AWS SAMCLI comandi nell'`default`ambiente:

TOML

```
[default.global.parameters]
stack_name = "sam-app"
```

YAML

```
default:
  global:
    parameters:
      stack_name: sam-app
```

2. È inoltre possibile specificare i valori dei parametri e modificare il file di configurazione tramite il flusso AWS SAMCLI interattivo.

   Di seguito è riportato un esempio del flusso `sam deploy --guided` interattivo:

```
$ sam deploy --guided

Configuring SAM deploy
======================

    Looking for config file [samconfig.toml] :  Found
    Reading default arguments  :  Success
```

```
    Setting default arguments for 'sam deploy'
    =========================================
    Stack Name [sam-app]: ENTER
    AWS Region [us-west-2]: ENTER
    #Shows you resources changes to be deployed and require a 'Y' to initiate
deploy
    Confirm changes before deploy [Y/n]: n
    #SAM needs permission to be able to create roles to connect to the resources in
your template
    Allow SAM CLI IAM role creation [Y/n]: ENTER
    #Preserves the state of previously provisioned resources when an operation
fails
    Disable rollback [y/N]: ENTER
    HelloWorldFunction may not have authorization defined, Is this okay? [y/N]: y
    Save arguments to configuration file [Y/n]: ENTER
    SAM configuration file [samconfig.toml]: ENTER
    SAM configuration environment [default]: ENTER
```

Per ulteriori informazioni, consulta Creazione e modifica dei file di configurazione.

Esempi

TOMLEsempio di base

Di seguito è riportato un esempio di file di `samconfig.toml` configurazione:

```
...
version = 0.1

[default]
[default.global]
[default.global.parameters]
stack_name = "sam-app"

[default.build.parameters]
cached = true
parallel = true

[default.deploy.parameters]
capabilities = "CAPABILITY_IAM"
confirm_changeset = true
```

```
resolve_s3 = true

[default.sync.parameters]
watch = true

[default.local_start_api.parameters]
warm_containers = "EAGER"

[prod]
[prod.sync]
[prod.sync.parameters]
watch = false
```

YAMLEsempio di base

Di seguito è riportato un esempio di file di `samconfig.yaml` configurazione:

```
version 0.1
default:
  global:
    parameters:
      stack_name: sam-app
  build:
    parameters:
      cached: true
      parallel: true
  deploy:
    parameters:
      capabilities: CAPABILITY_IAM
      confirm_changeset: true
      resolve_s3: true
  sync:
    parameters:
      watch: true
  local_start_api:
    parameters:
      warm_containers: EAGER
prod:
  sync:
    parameters:
      watch: false
```

Configurare le credenziali e le impostazioni di base

Usa AWS Command Line Interface (AWS CLI) per configurare le impostazioni di base come AWS credenziali, nome dell'area predefinita e formato di output predefinito. Una volta configurate, è possibile utilizzare queste impostazioni con. AWS SAMCLI Per ulteriori informazioni, consulta quanto segue nella Guida AWS Command Line Interface per l'utente:

- Nozioni di base sulla configurazione
- Impostazioni dei file di configurazione e credenziali
- Profili denominati per AWS CLI
- Utilizzo di un profilo denominato abilitato per IAM Identity Center

Per istruzioni di configurazione rapide, consultaPassaggio 5: Utilizzare AWS CLI per configurare le AWS credenziali.

AWS SAMCLIcomandi principali

AWS SAMCLIcontiene alcuni comandi di base che puoi utilizzare per creare, compilare, testare, distribuire e sincronizzare le tue applicazioni serverless. La tabella seguente elenca questi comandi e fornisce collegamenti con ulteriori informazioni per ciascuno di essi.

Per un elenco completo dei AWS SAMCLI comandi, vedereAWS SAMCLIriferimento al comando.

Comando	Cosa fa	Argomenti correlati
sam build	Prepara un'applicazione per i passaggi successivi del flusso di lavoro degli sviluppatori, come i test locali o la AWS distribuzione nel cloud.	• Introduzione alla costruzione con AWS SAM • sam build
sam deploy	Distribuisce un'applicazione nel AWS cloud utilizzando. CloudFormation	• Introduzione alla distribuzione con AWS SAM • sam deploy

Comando	Cosa fa	Argomenti correlati
sam init	Fornisce opzioni per inizializzare e creare una nuova applicazione serverless.	• Crea la tua candidatura in AWS SAM • sam init
sam local	Fornisce sottocomandi per testare localmente le applicazioni serverless.	• Introduzione ai test con il sam local comando • sam local generate-event • sam local invoke • sam local start-api • sam local start-lambda
sam remote invoke	Fornisce un modo per interagire con AWS le risorse supportate nel AWS cloud.	• Introduzione ai test nel cloud con sam remote invoke • sam remote invoke
sam remote test-event	Fornisce un modo per accedere e gestire eventi di test condivisibili per le funzioni AWS Lambda.	• Introduzione al cloud testing con sam remote test-event • sam remote test-event
sam sync	Fornisce opzioni per sincronizzare rapidamente le modifiche locali delle applicazioni nel AWS cloud.	• Introduzione all'uso sam sync della sincronizzazione con Cloud AWS • sam sync

Test locali con AWS SAMCLI

AWS SAMCLIConsente di testare le applicazioni serverless a livello locale su diversi strumenti Infrastructure as Code (IaC). Questa guida spiega come utilizzarlo AWS SAMCLI per i test locali con vari framework IaC.

I vantaggi principali dell'utilizzo AWS SAMCLI per i test locali includono:

- Sviluppo rapido: verifica le modifiche al codice senza distribuirle su AWS
- Efficienza in termini di costi: sviluppo e test senza incorrere in AWS costi
- Funzionalità offline: lavora sulle tue applicazioni senza una connessione Internet
- Debug semplificato: analizza il codice della funzione Lambda localmente utilizzando un debugger
- Test realistici: testa le tue applicazioni utilizzando l'emulazione locale dei servizi AWS

Testa le AWS SAM applicazioni localmente

Per informazioni sul test delle applicazioni definite utilizzando AWS SAM modelli, consulta la sezione Test e debug delle applicazioni serverless in questa guida.

Prova i modelli localmente CloudFormation

Per utilizzare le funzionalità AWS SAMCLI di test locali con CloudFormation, aggiungi la AWS SAM trasformazione al tuo CloudFormation modello. Per ulteriori informazioni, consulta l'anatomia AWS SAM del modello per:

- Aggiungere una AWS SAM trasformazione ai modelli CloudFormation
- Comprendere la compatibilità dei modelli
- Esplorazione della sintassi delle risorse serverless

Testa le applicazioni AWS CDK localmente

È possibile utilizzarlo AWS SAMCLI per testare localmente le applicazioni AWS CDK dopo aver sintetizzato l'applicazione CDK in CloudFormation modelli utilizzando il AWS comando cdk synth.

Per ulteriori informazioni, consulta la sezione Test e creazione di applicazioni CDK a livello locale utilizzando la guida per AWS sviluppatori CDK. AWS SAMCLI AWS Questo include:

- Guida introduttiva ai test a livello locale
- Test locali delle applicazioni AWS CDK con AWS SAM

Testa le applicazioni Terraform localmente

AWS SAMCLISupporta i progetti Terraform e Terraform Cloud. Puoi usarlo per eseguire il debug e il test locali di: funzioni e livelli Lambda e Amazon API Gateway HTTP e REST. APIs

Per configurare il tuo ambiente e conoscere tutte le funzionalità disponibili, consulta Terraform Support in questa guida. Questo include:

- Guida introduttiva al supporto Terraform per AWS SAMCLI
- Utilizzo di AWS SAMCLI with Terraform per il debug e il test locali

AWS SAM modello

Dopo aver eseguito il sam init comando e completato il flusso di lavoro successivo, AWS SAM crea la directory del progetto dell'applicazione, che è il AWS SAM progetto. Definite la vostra applicazione serverless aggiungendo codice al AWS SAM progetto. Sebbene il AWS SAM progetto sia costituito da un insieme di file e cartelle, il file su cui lavori principalmente è il AWS SAM modello (denominato`template.yaml`). In questo modello, scrivi il codice per esprimere risorse, mappature delle sorgenti degli eventi e altre proprietà che definiscono la tua applicazione serverless.

> **Note**
>
> Un elemento chiave del AWS SAM modello è la specifica del AWS SAM modello. Questa specifica fornisce la sintassi abbreviata che, rispetto a CloudFormation, consente di utilizzare un minor numero di righe di codice per definire le risorse, i mapping delle sorgenti degli eventi, le autorizzazioni e altre proprietà dell'applicazione serverless. APIs

Questa sezione fornisce dettagli su come utilizzare le sezioni del AWS SAM modello per definire i tipi di risorse, le proprietà delle risorse, i tipi di dati, gli attributi delle risorse, le funzioni intrinseche e le estensioni API Gateway.

AWS SAM i modelli sono un'estensione dei CloudFormation modelli, con tipi di sintassi unici che utilizzano una sintassi abbreviata con un minor numero di righe di codice rispetto a. CloudFormation Ciò velocizza lo sviluppo durante la creazione di un'applicazione serverless. Per ulteriori informazioni, vedi AWS SAM risorse e proprietà. Per il riferimento completo ai CloudFormation modelli, consulta CloudFormation Template Reference nella Guida per l'AWS CloudFormation utente.

Durante lo sviluppo, troverete spesso utile suddividere il codice dell'applicazione in file separati per organizzare e gestire meglio l'applicazione. Un esempio di base è l'utilizzo di un file separato per il codice della AWS Lambda funzione anziché inserire questo codice nel AWS SAM modello. Fatelo organizzando il codice della funzione Lambda in una sottodirectory del progetto e facendo riferimento al suo percorso locale all'interno del modello (). AWS Serverless Application Model AWS SAM

Argomenti

- AWS SAM anatomia del modello
- AWS SAM risorse e proprietà

- CloudFormation Risorse generate per AWS SAM
- Attributi delle risorse supportati da AWS SAM
- Estensioni API Gateway per AWS SAM
- Funzioni intrinseche per AWS SAM

AWS SAM anatomia del modello

Un file AWS SAM modello segue da vicino il formato di un file CloudFormation modello, descritto in Anatomia del modello nella Guida per l'AWS CloudFormation utente. Le differenze principali tra i file AWS SAM modello e i file CloudFormation modello sono le seguenti:

- Dichiarazione Transform. La dichiarazione `Transform: AWS::Serverless-2016-10-31` è obbligatoria per i file AWS SAM modello. Questa dichiarazione identifica un file CloudFormation modello come file AWS SAM modello. Per ulteriori informazioni sulle trasformazioni, consulta Transform nella Guida per l'AWS CloudFormation utente.
- Sezione Globals. La `Globals` sezione è unica per. AWS SAM Definisce proprietà comuni a tutte le funzioni serverless e APIs. Tutte le`AWS::Serverless::Function`,`AWS::Serverless::Api`,`AWS::Serverless::CapacityProvid` `AWS::Serverless::HttpApiAWS::Serverless::SimpleTable`, e `AWS::Serverless::StateMachine` le risorse ereditano le proprietà definite nella `Globals` sezione. Per ulteriori informazioni su questa sezione, vedereSezione Globals del modello AWS SAM.
- Sezione Risorse. Nei AWS SAM modelli la `Resources` sezione può contenere una combinazione di CloudFormation risorse e AWS SAM risorse. Per ulteriori informazioni sulle CloudFormation risorse, consultate il riferimento ai tipi di AWS risorse e proprietà nella Guida AWS CloudFormation per l'utente. Per ulteriori informazioni sulle AWS SAM risorse, vedereAWS SAM risorse e proprietà.

Tutte le altre sezioni di un file AWS SAM modello corrispondono alla sezione del file CloudFormation modello con lo stesso nome.

YAML

Il seguente esempio mostra un frammento di modello in formato YAML.

```
Transform: AWS::Serverless-2016-10-31
```

```
Globals:
  set of globals

Description:
  String

Metadata:
  template metadata

Parameters:
  set of parameters

Mappings:
  set of mappings

Conditions:
  set of conditions

Resources:
  set of resources

Outputs:
  set of outputs
```

Sezioni del modello

AWS SAM i modelli possono includere diverse sezioni principali. Sono necessarie solo `Resources` le sezioni `Transform` e.

Puoi includere le sezioni del modello in qualsiasi ordine. Tuttavia, se si utilizzano estensioni linguistiche, è necessario aggiungerle `AWS::LanguageExtensions` prima della trasformazione serverless (ovvero prima`AWS::Serverless-2016-10-31`), come mostrato nell'esempio seguente:

```
Transform:
  - AWS::LanguageExtensions
  - AWS::Serverless-2016-10-31
```

Durante la creazione del modello, può essere utile utilizzare l'ordine logico mostrato nell'elenco seguente. Questo perché i valori di una sezione potrebbero fare riferimento a valori di una sezione precedente.

Trasforma (obbligatorio)

Per AWS SAM i modelli, è necessario includere questa sezione con un valore di`AWS::Serverless-2016-10-31`.

Le trasformazioni aggiuntive sono facoltative. Per ulteriori informazioni sulle trasformazioni, consulta Transform nella Guida per l'AWS CloudFormation utente.

Globali (opzionale)

Proprietà comuni a tutte le funzioni serverless e APIs tabelle semplici. Tutte le`AWS::Serverless::Function`,`AWS::Serverless::Api`,`AWS::Serverless::CapacityProvi` `AWS::Serverless::HttpApi``AWS::Serverless::SimpleTable`, e `AWS::Serverless::StateMachine` le risorse ereditano le proprietà definite nella `Globals` sezione.

Questa sezione è esclusiva di. AWS SAM Non esiste una sezione corrispondente nei CloudFormation modelli.

Descrizione (facoltativa)

Stringa di testo che descrive il modello.

Questa sezione corrisponde direttamente alla `Description` sezione dei CloudFormation modelli.

Metadata (Metadati) (facoltativa)

Oggetti che forniscono informazioni aggiuntive sul modello.

Questa sezione corrisponde direttamente alla `Metadata` sezione dei CloudFormation modelli.

Parameters (Parametri) (facoltativa)

Valori da passare al modello in fase di runtime (quando crei o aggiorni uno stack). Puoi fare riferimento ai parametri dalle sezioni `Resources` e `Outputs` del modello. Gli oggetti dichiarati nella `Parameters` sezione fanno sì che il sam deploy --guided comando presenti all'utente istruzioni aggiuntive.

I valori passati utilizzando il `--parameter-overrides` parametro del `sam deploy` comando e le voci nel file di configurazione hanno la precedenza sulle voci del file modello. AWS SAM Per ulteriori informazioni sul comando, vedere nel riferimento ai `sam deploy` comandi. sam deploy AWS SAMCLI Per ulteriori informazioni sul file di configurazione, consulta AWS SAMCLIfile di configurazione.

Mappings (Mappature) (facoltativa)

Mappatura di chiavi e valori associati che puoi utilizzare per specificare i valori dei parametri condizionali, in modo analogo a una tabella di ricerca. È possibile abbinare una chiave a un valore corrispondente utilizzando la funzione `Fn::FindInMap`intrinseca nelle sezioni `Resources` and`Outputs`.

Questa sezione corrisponde direttamente alla `Mappings` sezione dei CloudFormation modelli.

Conditions (Condizioni) (facoltativa)

Condizioni che consentono di controllare se alcune risorse sono state creati o se a determinate proprietà di risorsa è stato assegnato un valore durante la creazione o l'aggiornamento dello stack. Ad esempio, puoi creare in modo condizionale una risorsa a seconda che lo stack sia destinato a un ambiente di test o di produzione.

Questa sezione corrisponde direttamente alla `Conditions` sezione dei CloudFormation modelli.

Resources (Risorse) (obbligatoria)

Le risorse dello stack e le relative proprietà, ad esempio un'istanza Amazon Elastic Compute Cloud (Amazon EC2) o un bucket Amazon Simple Storage Service (Amazon S3). Puoi fare riferimento alle risorse nelle sezioni `Resources` e `Outputs` del modello.

Questa sezione è simile alla sezione dei `Resources` modelli. CloudFormation Nei AWS SAM modelli, questa sezione può contenere AWS SAM risorse oltre alle CloudFormation risorse.

Outputs (Output) (facoltativa)

I valori che vengono restituiti ogni volta che si visualizzano le proprietà dello stack. Ad esempio, puoi dichiarare un output per il nome di un bucket S3 e quindi chiamare il comando `aws cloudformation describe-stacks` AWS Command Line Interface (AWS CLI) per visualizzare il nome.

Questa sezione corrisponde direttamente alla `Outputs` sezione dei modelli. CloudFormation

Fasi successive

Per scaricare e distribuire un'applicazione serverless di esempio che contiene un file AWS SAM modello, consulta Guida introduttiva con AWS SAM e segui le istruzioni riportate in. Tutorial: Implementa un'applicazione Hello World con AWS SAM

Sezione Globals del modello AWS SAM

A volte le risorse dichiarate in un AWS SAM modello hanno configurazioni comuni. Ad esempio, potresti avere un'applicazione con più `AWS::Serverless::Function` risorse con configurazioni`Runtime`,, `Memory` `VPCConfig``Environment`, e `Cors` identiche. Invece di duplicare queste informazioni in ogni risorsa, potete dichiararle una volta nella `Globals` sezione e lasciare che le risorse le ereditino.

La `Globals` sezione supporta i seguenti tipi di risorse: AWS SAM

- `AWS::Serverless::Api`
- `AWS::Serverless::CapacityProvider`
- `AWS::Serverless::Function`
- `AWS::Serverless::HttpApi`
- `AWS::Serverless::SimpleTable`
- `AWS::Serverless::StateMachine`

Esempio:

```
Globals:
  Function:
    Runtime: nodejs12.x
    Timeout: 180
    Handler: index.handler
    Environment:
      Variables:
        TABLE_NAME: data-table

Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Properties:
      Environment:
        Variables:
          MESSAGE: "Hello From SAM"

  ThumbnailFunction:
    Type: AWS::Serverless::Function
    Properties:
      Events:
```

```
        Thumbnail:
          Type: Api
          Properties:
            Path: /thumbnail
            Method: POST
```

In questo esempio, entrambi `ThumbnailFunction` utilizzano «nodejs12.x» per`Runtime`, «180" secondi per `HelloWorldFunction` e «index.handler» per`Timeout`. `Handler` `HelloWorldFunction`aggiunge la variabile di ambiente MESSAGE, oltre alla variabile TABLE_NAME ereditata. `ThumbnailFunction`eredita tutte le `Globals` proprietà e aggiunge una fonte di eventi API.

Risorse e proprietà supportate

AWS SAM supporta le seguenti risorse e proprietà.

```
Globals:
  Api:
    AccessLogSetting:
    Auth:
    BinaryMediaTypes:
    CacheClusterEnabled:
    CacheClusterSize:
    CanarySetting:
    Cors:
    DefinitionUri:
    Domain:
    EndpointConfiguration:
    GatewayResponses:
    MethodSettings:
    MinimumCompressionSize:
    Name:
    OpenApiVersion:
    PropagateTags:
    TracingEnabled:
    Variables:

  CapacityProvider:
    InstanceRequirements:
    KmsKeyArn:
    OperatorRole:
    PropagateTags:
    ScalingConfig:
```

```
    Tags:
    VpcConfig:

  Function:
    Architectures:
    AssumeRolePolicyDocument:
    AutoPublishAlias:
    AutoPublishAliasAllProperties:
    CapacityProviderConfig:
    CodeSigningConfigArn:
    CodeUri:
    DeadLetterQueue:
    DeploymentPreference:
    Description:
    DurableConfig:
    Environment:
    EphemeralStorage:
    EventInvokeConfig:
    FileSystemConfigs:
    FunctionScalingConfig:
    FunctionUrlConfig:
    Handler:
    KmsKeyArn:
    Layers:
    LoggingConfig:
    MemorySize:
    PermissionsBoundary:
    PropagateTags:
    ProvisionedConcurrencyConfig:
    PublishToLatestPublished:
    RecursiveLoop:
    ReservedConcurrentExecutions:
    RolePath:
    Runtime:
    RuntimeManagementConfig:
    SnapStart:
    SourceKMSKeyArn:
    Tags:
    TenancyConfig:
    Timeout:
    Tracing:
    VersionDeletionPolicy:
    VpcConfig:
```

```
  HttpApi:
    AccessLogSettings:
    Auth:
    PropagateTags:
    StageVariables:
    Tags:

  SimpleTable:
    SSESpecification:

  StateMachine:
    PropagateTags:
```

> **Note**
>
> Tutte le risorse e le proprietà che non sono incluse nell'elenco precedente non sono supportate. Alcuni motivi per non supportarle includono: 1) aprono potenziali problemi di sicurezza o 2) rendono il modello difficile da capire.

Implicito APIs

AWS SAM viene creato in modo implicito APIs quando si dichiara un'API nella sezione. `Events` È possibile utilizzare `Globals` per sovrascrivere tutte le proprietà implicite. APIs

Proprietà sovrascrivibili

Le risorse possono sovrascrivere le proprietà dichiarate nella sezione. `Globals` Ad esempio, è possibile aggiungere nuove variabili a una mappa di variabili di ambiente oppure sovrascrivere le variabili dichiarate globalmente. Ma la risorsa non può rimuovere una proprietà specificata nella `Globals` sezione.

Più in generale, la `Globals` sezione dichiara le proprietà condivise da tutte le risorse. Alcune risorse possono fornire nuovi valori per le proprietà dichiarate a livello globale, ma non possono rimuoverle. Se alcune risorse utilizzano una proprietà ma altre no, non è necessario dichiararle nella `Globals` sezione.

Le sezioni seguenti descrivono come funziona l'override per diversi tipi di dati.

I tipi di dati primitivi vengono sostituiti

I tipi di dati primitivi includono stringhe, numeri, valori booleani e così via.

Il valore specificato nella `Resources` sezione sostituisce il valore nella sezione. `Globals`

Esempio:

```
Globals:
  Function:
    Runtime: nodejs12.x

Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      Runtime: python3.9
```

La forma `Runtime` `MyFunction` è impostata su. `python3.9`

Le mappe vengono unite

Le mappe sono note anche come dizionari o raccolte di coppie chiave-valore.

Le voci della mappa nella `Resources` sezione vengono unite alle voci della mappa globale. Se sono presenti duplicati, la voce della `Resource` sezione sostituisce la voce della sezione. `Globals`

Esempio:

```
Globals:
  Function:
    Environment:
      Variables:
        STAGE: Production
        TABLE_NAME: global-table

Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      Environment:
        Variables:
          TABLE_NAME: resource-table
```

```
          NEW_VAR: hello
```

Le variabili di ambiente di `MyFunction` sono impostate come segue:

```
{
  "STAGE": "Production",
  "TABLE_NAME": "resource-table",
  "NEW_VAR": "hello"
}
```

Gli elenchi sono additivi

Gli elenchi sono noti anche come matrici.

Le voci dell'elenco nella `Globals` sezione vengono anteposte all'elenco nella sezione. `Resources`

Esempio:

```
Globals:
  Function:
    VpcConfig:
      SecurityGroupIds:
        - sg-123
        - sg-456

Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      VpcConfig:
        SecurityGroupIds:
          - sg-first
```

I `SecurityGroupIds` for `MyFunction` `VpcConfig` sono impostati come segue:

```
[ "sg-123", "sg-456", "sg-first" ]
```

AWS SAM risorse e proprietà

Questa sezione descrive i tipi di risorse e proprietà specifici per. AWS SAM Queste risorse e proprietà vengono definite utilizzando la sintassi AWS SAM abbreviata. AWS SAM supporta anche

tipi di CloudFormation risorse e proprietà. Per informazioni di riferimento per tutti i tipi di AWS risorse e proprietà CloudFormation e il AWS SAM supporto, consulta il riferimento ai tipi di AWS risorse e proprietà nella Guida per l'AWS CloudFormation utente.

Argomenti

- AWS::Serverless::Api
- AWS::Serverless::Application
- AWS::Serverless::CapacityProvider
- AWS::Serverless::Connector
- AWS::Serverless::Function
- AWS::Serverless::GraphQLApi
- AWS::Serverless::HttpApi
- AWS::Serverless::LayerVersion
- AWS::Serverless::SimpleTable
- AWS::Serverless::StateMachine

AWS::Serverless::Api

Crea una raccolta di risorse e metodi Amazon API Gateway che possono essere richiamati tramite endpoint HTTPS.

Non è necessario aggiungere esplicitamente una AWS::Serverless::Api risorsa a un modello AWS Serverless Application Definition. Una risorsa di questo tipo viene creata implicitamente dall'unione di eventi Api definiti su AWS::Serverless::Function risorse definite nel modello che non fanno riferimento a una risorsa. AWS::Serverless::Api

È necessario utilizzare una AWS::Serverless::Api risorsa per definire e documentare l'utilizzo dell'API OpenApi, che offre una maggiore capacità di configurazione delle risorse Amazon API Gateway sottostanti.

Ti consigliamo di utilizzare CloudFormation hook o policy IAM per verificare che alle risorse API Gateway siano associate autorizzazioni per controllarne l'accesso.

Per ulteriori informazioni sull'uso degli CloudFormation hook, consulta Registrazione degli hook nella guida per l'utente della CloudFormation CLI e nel repository. apigw-enforce-authorizer GitHub

Per ulteriori informazioni sull'utilizzo delle policy IAM, consulta Require che le route API abbiano l'autorizzazione nella API Gateway Developer Guide.

> **Note**
>
> Quando esegui la distribuzione a AWS CloudFormation, AWS SAM trasforma le tue AWS SAM risorse in CloudFormation risorse. Per ulteriori informazioni, consulta CloudFormation Risorse generate per AWS SAM.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Type: AWS::Serverless::Api
Properties:
  AccessLogSetting: AccessLogSetting
  AlwaysDeploy: Boolean
  ApiKeySourceType: String
  Auth: ApiAuth
  BinaryMediaTypes: List
  CacheClusterEnabled: Boolean
  CacheClusterSize: String
  CanarySetting: CanarySetting
  Cors: String | CorsConfiguration
  DefinitionBody: JSON
  DefinitionUri: String | ApiDefinition
  Description: String
  DisableExecuteApiEndpoint: Boolean
  Domain: DomainConfiguration
  EndpointConfiguration: EndpointConfiguration
  FailOnWarnings: Boolean
  GatewayResponses: Map
  MergeDefinitions: Boolean
  MethodSettings: MethodSettings
  MinimumCompressionSize: Integer
  Mode: String
  Models: Map
  Name: String
```

```
  OpenApiVersion: String
  PropagateTags: Boolean
  Policy: JSON
  StageName: String
  Tags: Map
  TracingEnabled: Boolean
  Variables: Map
```

Properties

`AccessLogSetting`

Configura l'impostazione del registro di accesso per uno stage.

Tipo: AccessLogSetting

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `AccessLogSetting` proprietà di una `AWS::ApiGateway::Stage` risorsa.

`AlwaysDeploy`

Implementa sempre l'API, anche quando non sono state rilevate modifiche all'API.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ApiKeySourceType`

L'origine della chiave API per le richieste di misurazione in base a un piano di utilizzo. I valori validi sono `HEADER` e `AUTHORIZER`.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ApiKeySourceType` proprietà di una `AWS::ApiGateway::RestApi` risorsa.

`Auth`

Configura l'autorizzazione per controllare l'accesso alla tua API API Gateway.

Per ulteriori informazioni sulla configurazione dell'accesso tramite, AWS SAM vedereControlla l'accesso alle API con il tuo AWS SAM modello. Per un esempio che mostra come sovrascrivere un autorizzatore globale, vedi. the section called “Sostituisci un autorizzatore globale per la tua API REST di Amazon API Gateway”

Tipo: ApiAuth

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`BinaryMediaTypes`

Elenco dei tipi MIME che l'API potrebbe restituire. Usalo per abilitare il supporto binario perAPIs.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è simile alla `BinaryMediaTypes` proprietà di una `AWS::ApiGateway::RestApi` risorsa. L'elenco di BinaryMediaTypes viene aggiunto sia alla CloudFormation risorsa che al documento OpenAPI.

`CacheClusterEnabled`

Indica se la memorizzazione nella cache è abilitata per lo stage. Per memorizzare nella cache le risposte, è inoltre necessario `CachingEnabled` impostare su `true` under`MethodSettings`.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CacheClusterEnabled` proprietà di una `AWS::ApiGateway::Stage` risorsa.

`CacheClusterSize`

Le dimensioni del cluster di cache della fase.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CacheClusterSize` proprietà di una `AWS::ApiGateway::Stage` risorsa.

`CanarySetting`

Configura un'impostazione Canary in una fase di distribuzione regolare.

Tipo: CanarySetting

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CanarySetting` proprietà di una `AWS::ApiGateway::Stage` risorsa.

`Cors`

Gestisci la condivisione delle risorse tra le origini (CORS) per tutti i tuoi API Gateway. APIs Specificate il dominio da consentire come stringa o specificate un dizionario con una configurazione Cors aggiuntiva.

> **Note**
>
> CORS richiede AWS SAM di modificare la definizione OpenAPI. Crea una definizione OpenAPI in linea per attivare `DefinitionBody` CORS.

Per ulteriori informazioni su CORS, consulta Enable CORS for an API Gateway REST API Resource nella API Gateway Developer Guide.

Tipo: Stringa | CorsConfiguration

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`DefinitionBody`

Specificazione OpenAPI che descrive la tua API. Se `DefinitionUri` nessuna delle due `DefinitionBody` viene specificata, SAM genererà una `DefinitionBody` per te in base alla configurazione del modello.

Per fare riferimento a un OpenAPI file locale che definisce la tua API, usa la `AWS::Include` trasformazione. Per ulteriori informazioni, consulta Come AWS SAM carica i file locali.

Type: JSON

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `Body` proprietà di una `AWS::ApiGateway::RestApi` risorsa. Se vengono fornite determinate proprietà, il contenuto può essere inserito o modificato nella DefinitionBody prima di essere passato a CloudFormation. Le proprietà includono `AuthBinaryMediaTypes`,`Cors`,`GatewayResponses`,`Models`, e una `EventSource` di tipo Api per una corrispondente`AWS::Serverless::Function`.

`DefinitionUri`

Uri Amazon S3, percorso del file locale o oggetto posizione del documento OpenAPI che definisce l'API. L'oggetto Amazon S3 a cui fa riferimento questa proprietà deve essere un file OpenAPI valido. Se non viene specificato `DefinitionUri` nessuno `DefinitionBody` dei due, SAM genererà un file `DefinitionBody` per te in base alla configurazione del modello.

Se viene fornito un percorso di file locale, il modello deve passare attraverso il flusso di lavoro che include il `sam package` comando `sam deploy` o, affinché la definizione venga trasformata correttamente.

Le funzioni intrinseche non sono supportate nei OpenApi file esterni a cui fa riferimento. `DefinitionUri` Utilizzate invece la `DefinitionBody` proprietà con Include Transform per importare una OpenApi definizione nel modello.

Tipo: String | ApiDefinition

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è simile alla `BodyS3Location` proprietà di una `AWS::ApiGateway::RestApi` risorsa. Le proprietà annidate di Amazon S3 sono denominate in modo diverso.

`Description`

Una descrizione della risorsa Api.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una `AWS::ApiGateway::RestApi` risorsa.

`DisableExecuteApiEndpoint`

Specifica se i client possono richiamare l'API utilizzando l'endpoint `execute-api` predefinito. Per impostazione predefinita, i client possono richiamare la tua API con l'impostazione predefinita`https://{api_id}.execute-api.{region}.amazonaws.com`. Per richiedere che i client utilizzino un nome di dominio personalizzato per richiamare l'API, specifica `True`.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `DisableExecuteApiEndpoint` proprietà di una `AWS::ApiGateway::RestApi` risorsa. Viene passato direttamente alla `disableExecuteApiEndpoint` proprietà di un' `x-amazon-apigateway-endpoint-configuration`estensione, che viene aggiunta alla `Body` proprietà di una `AWS::ApiGateway::RestApi` risorsa.

`Domain`

Configura un dominio personalizzato per questa API API Gateway.

Tipo: DomainConfiguration

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`EndpointConfiguration`

Il tipo di endpoint di un'API REST.

Tipo: EndpointConfiguration

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `EndpointConfiguration` proprietà di una `AWS::ApiGateway::RestApi` risorsa. Le proprietà di configurazione annidate sono denominate in modo diverso.

`FailOnWarnings`

Speciifica se ripristinare la creazione dell'API (`true`) o meno (`false`) quando viene rilevato un avviso. Il valore predefinito è `false`.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FailOnWarnings` proprietà di una `AWS::ApiGateway::RestApi` risorsa.

`GatewayResponses`

Configura Gateway Responses per un'API. Le risposte Gateway sono risposte restituite da API Gateway, direttamente o tramite l'uso di Lambda Authorizers. Per ulteriori informazioni, consulta la documentazione dell' OpenApi estensione Api Gateway per Gateway Responses.

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`MergeDefinitions`

AWS SAM genera una OpenAPI specifica dalla fonte dell'evento API. `true`Specificate di AWS SAM unirlo alla OpenAPI specifica in linea definita nella vostra `AWS::Serverless::Api` risorsa. `false`Specificare di non unire.

`MergeDefinitions`richiede la definizione della `DefinitionBody` proprietà `AWS::Serverless::Api` per. `MergeDefinitions`non è compatibile con la `DefinitionUri` proprietà di`AWS::Serverless::Api`.

Valore predefinito: `false`

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`MethodSettings`

Configura tutte le impostazioni per API Stage, tra cui Logging, Metrics, CacheTTL, Throttling.

Tipo: elenco di MethodSetting

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di una risorsa. `MethodSettings` `AWS::ApiGateway::Stage`

`MinimumCompressionSize`

Consenti la compressione dei corpi di risposta in base all'intestazione Accept-Encoding del client. La compressione viene attivata quando la dimensione del corpo di risposta è maggiore o uguale alla soglia configurata. La soglia di dimensione corporea massima è 10 MB (10.485.760 byte). - Sono supportati i seguenti tipi di compressione: gzip, deflate e identity.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MinimumCompressionSize` proprietà di una risorsa. `AWS::ApiGateway::RestApi`

`Mode`

Questa proprietà si applica solo quando si utilizza OpenAPI per definire l'API REST. `Mode` determina il modo in cui Gateway API gestisce gli aggiornamenti delle risorse. Per ulteriori informazioni, vedere la proprietà Mode del tipo di AWS::ApiGateway::RestApirisorsa.

Valori validi: `overwrite` o `merge`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Mode` proprietà di una `AWS::ApiGateway::RestApi` risorsa.

`Models`

Gli schemi che devono essere utilizzati dai metodi API. Questi schemi possono essere descritti utilizzando JSON o YAML. Vedi la sezione Esempi in fondo a questa pagina per esempi di modelli.

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Name`

Un nome per la RestApi risorsa API Gateway

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::ApiGateway::RestApi` risorsa.

`OpenApiVersion`

Versione di OpenApi da usare. Questo può riguardare `2.0` la specifica Swagger o una delle versioni OpenApi 3.0, ad esempio. `3.0.1` Per ulteriori informazioni su OpenAPI, vedere la specifica OpenAPI.

> **Note**
>
> AWS SAM crea una fase chiamata di `Stage` default. L'impostazione di questa proprietà su qualsiasi valore valido impedirà la creazione dello stage`Stage`.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`PropagateTags`

Indica se passare o meno i tag dalla `Tags` proprietà alle risorse AWS::Serverless::Api generate. `True`Specificate di propagare i tag nelle risorse generate.

Tipo: Booleano

Obbligatorio: no

Default: `False`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Policy`

Un documento di policy che contiene le autorizzazioni per l'API. Per impostare l'ARN per la policy, utilizza la funzione intrinseca `!Join` con `""` come delimitatore e valori di `"execute-api:/"` e `"*"`.

Type: JSON

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà Policy di una `AWS::ApiGateway::RestApi` risorsa.

`StageName`

Il nome della fase, che API Gateway utilizza come primo segmento di percorso nell'URI (Uniform Resource Identifier) di invoke.

Per fare riferimento alla risorsa dello stage, usa. `<api-logical-id>.Stage` Per ulteriori informazioni sul riferimento alle risorse generate quando viene specificata una AWS::Serverless::Api risorsa, vedereCloudFormation risorse generate quando AWS::Serverless::Api è specificato. Per informazioni generali sulle CloudFormation risorse generate, vedereCloudFormation Risorse generate per AWS SAM.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è simile alla `StageName` proprietà di una `AWS::ApiGateway::Stage` risorsa. È richiesto in SAM, ma non richiesto in API Gateway

Note aggiuntive: L'API Implicit ha il nome d'arte «Prod».

`Tags`

Una mappa (da stringa a stringa) che specifica i tag da aggiungere a questa fase di API Gateway. Per informazioni dettagliate sulle chiavi e i valori validi per i tag, consulta Resource tag nella Guida per l'AWS CloudFormation utente.

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `Tags` proprietà di una `AWS::ApiGateway::Stage` risorsa. La proprietà Tags in SAM è costituita da coppie Key:Value; in CloudFormation essa è costituita da un elenco di oggetti Tag.

`TracingEnabled`

Indica se il tracciamento attivo con X-Ray è abilitato per lo stage. Per ulteriori informazioni su X-Ray, consulta Tracciare le richieste degli utenti su REST usando APIs X-Ray nella API Gateway Developer Guide.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `TracingEnabled` proprietà di una risorsa. `AWS::ApiGateway::Stage`

`Variables`

Una mappa (da stringa a stringa) che definisce le variabili dello stage, dove il nome della variabile è la chiave e il valore della variabile è il valore. I nomi della variabile sono limitati ai caratteri alfanumerici. I valori devono corrispondere alla seguente espressione regolare: `[A-Za-z0-9._~:/?#&=,-]+`.

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Variables` proprietà di una `AWS::ApiGateway::Stage` risorsa.

Valori restituiti

Ref

Quando l'ID logico di questa risorsa viene fornito alla funzione `Ref` intrinseca, restituisce l'ID dell'API Gateway API sottostante.

Per ulteriori informazioni sull'utilizzo della `Ref` funzione, consulta `Ref`la Guida per l'AWS CloudFormation utente.

Fn: GetAtt

`Fn::GetAtt` restituisce un valore per un attributo specificato di questo tipo. Di seguito sono riportati gli attributi disponibili e i valori restituiti di esempio.

Per ulteriori informazioni sull'utilizzo`Fn::GetAtt`, consulta `Fn::GetAtt`la Guida per l'AWS CloudFormation utente.

`RootResourceId`

L'ID della risorsa root per una risorsa `RestApi`, ad esempio `a0bc123d4e`.

Esempi

SimpleApiExample

Un file AWS SAM modello Hello World che contiene una funzione Lambda con un endpoint API. Si tratta di un file AWS SAM modello completo per un'applicazione serverless funzionante.

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: AWS SAM template with a simple API definition
Resources:
  ApiGatewayApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: prod
  ApiFunction: # Adds a GET method at the root resource via an Api event
    Type: AWS::Serverless::Function
    Properties:
      Events:
        ApiEvent:
          Type: Api
          Properties:
            Path: /
            Method: get
            RestApiId:
              Ref: ApiGatewayApi
```

```
      Runtime: python3.10
      Handler: index.handler
      InlineCode: |
        def handler(event, context):
            return {'body': 'Hello World!', 'statusCode': 200}
```

ApiCorsExample

Un frammento di AWS SAM modello con un'API definita in un file Swagger esterno insieme a integrazioni Lambda e configurazioni CORS. Questa è solo una parte di un file modello che mostra una definizione. AWS SAM AWS::Serverless::Api

YAML

```
Resources:
  ApiGatewayApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: Prod
      # Allows www.example.com to call these APIs
      # SAM will automatically add AllowMethods with a list of methods for this API
      Cors: "'www.example.com'"
      DefinitionBody: # Pull in an OpenApi definition from S3
        'Fn::Transform':
          Name: 'AWS::Include'
          # Replace "bucket" with your bucket name
          Parameters:
            Location: s3://bucket/swagger.yaml
```

ApiCognitoAuthExample

Un frammento di AWS SAM modello con un'API che utilizza Amazon Cognito per autorizzare le richieste contro l'API. Questa è solo una parte di un file AWS SAM modello che mostra una definizione. AWS::Serverless::Api

YAML

```
Resources:
  ApiGatewayApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: Prod
```

```
      Cors: "'*'"
      Auth:
        DefaultAuthorizer: MyCognitoAuthorizer
        Authorizers:
          MyCognitoAuthorizer:
            UserPoolArn:
              Fn::GetAtt: [MyCognitoUserPool, Arn]
```

ApiModelsExample

Un frammento di AWS SAM modello con un'API che include uno schema Models. Questa è solo una parte di un file AWS SAM modello, che mostra una AWS::Serverless::Api definizione con due schemi di modello.

YAML

```
Resources:
  ApiGatewayApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: Prod
      Models:
        User:
          type: object
          required:
            - username
            - employee_id
          properties:
            username:
              type: string
            employee_id:
              type: integer
            department:
              type: string
        Item:
          type: object
          properties:
            count:
              type: integer
            category:
              type: string
            price:
              type: integer
```

Esempio di caching

Un file AWS SAM modello Hello World che contiene una funzione Lambda con un endpoint API. L'API ha la memorizzazione nella cache abilitata per una risorsa e un metodo. Per ulteriori informazioni sulla memorizzazione nella cache, consulta Enabling API caching to enhance responsiveness nella API Gateway Developer Guide.

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: AWS SAM template with a simple API definition with caching turned on
Resources:
  ApiGatewayApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: prod
      CacheClusterEnabled: true
      CacheClusterSize: '0.5'
      MethodSettings:
        - ResourcePath: /
          HttpMethod: GET
          CachingEnabled: true
          CacheTtlInSeconds: 300
      Tags:
        CacheMethods: All

  ApiFunction: # Adds a GET method at the root resource via an Api event
    Type: AWS::Serverless::Function
    Properties:
      Events:
        ApiEvent:
          Type: Api
          Properties:
            Path: /
            Method: get
            RestApiId:
              Ref: ApiGatewayApi
      Runtime: python3.10
      Handler: index.handler
      InlineCode: |
        def handler(event, context):
            return {'body': 'Hello World!', 'statusCode': 200}
```

Dominio personalizzato con un esempio di API privata

Un file AWS SAM modello Hello World che contiene una funzione Lambda con un endpoint API mappato su un dominio privato. Il modello crea un'associazione di accesso al dominio tra un endpoint VPC e il dominio privato. Per ulteriori informazioni, consulta Nomi di dominio personalizzati per uso privato APIs in API Gateway.

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: AWS SAM template configured with a custom domain using a private API

Parameters:
    DomainName:
      Type: String
      Default: mydomain.example.com
    CertificateArn:
      Type: String
    HostedZoneId:
      Type: String
    VpcEndpointId:
      Type: String
    VpcEndpointDomainName:
      Type: String
    VpcEndpointHostedZoneId:
      Type: String

Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      InlineCode: |
        def handler(event, context):
            return {'body': 'Hello World!', 'statusCode': 200}
      Handler: index.handler
      Runtime: python3.13
      Events:
        Fetch:
          Type: Api
          Properties:
            RestApiId:
              Ref: MyApi
```

```
          Method: Get
          Path: /get
  MyApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: Prod
      EndpointConfiguration:
        Type: PRIVATE
        VPCEndpointIds:
        - !Ref VpcEndpointId
      Policy:
        Version: '2012-10-17       '
        Statement:
        - Effect: Allow
          Principal: '*'
          Action: execute-api:Invoke
          Resource: execute-api:/*
        - Effect: Deny
          Principal: '*'
          Action: execute-api:Invoke
          Resource: execute-api:/*
          Condition:
            StringNotEquals:
              aws:SourceVpce: !Ref VpcEndpointId
      Domain:
        DomainName: !Ref DomainName
        CertificateArn: !Ref CertificateArn
        EndpointConfiguration: PRIVATE
        BasePath:
        - /
        Route53:
          HostedZoneId: !Ref HostedZoneId
          VpcEndpointDomainName: !Ref VpcEndpointDomainName
          VpcEndpointHostedZoneId: !Ref VpcEndpointHostedZoneId
        AccessAssociation:
          VpcEndpointId: !Ref VpcEndpointId
        Policy:
          Version: '2012-10-17       '
          Statement:
          - Effect: Allow
            Principal: '*'
            Action: execute-api:Invoke
            Resource: execute-api:/*
          - Effect: Deny
```

```
            Principal: '*'
            Action: execute-api:Invoke
            Resource: execute-api:/*
            Condition:
              StringNotEquals:
                aws:SourceVpce: !Ref VpcEndpointId
```

ApiAuth

Configura l'autorizzazione per controllare l'accesso alla tua API API Gateway.

Per ulteriori informazioni ed esempi sulla configurazione dell'accesso utilizzando, AWS SAM vedereControlla l'accesso alle API con il tuo AWS SAM modello.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  AddApiKeyRequiredToCorsPreflight: Boolean
  AddDefaultAuthorizerToCorsPreflight: Boolean
  ApiKeyRequired: Boolean
  Authorizers: CognitoAuthorizer | LambdaTokenAuthorizer | LambdaRequestAuthorizer |
AWS_IAM
  DefaultAuthorizer: String
  InvokeRole: String
  ResourcePolicy: ResourcePolicyStatement
  UsagePlan: ApiUsagePlan
```

> **Note**
>
> La `Authorizers` proprietà includeAWS_IAM, ma non è necessaria alcuna configurazione aggiuntiva per. `AWS_IAM` Per vedere un esempio, consulta AWS IAM.

Properties

`AddApiKeyRequiredToCorsPreflight`

Se le `Cors` proprietà `ApiKeyRequired` and sono impostate, l'impostazione `AddApiKeyRequiredToCorsPreflight` causerà l'aggiunta della chiave API alla `Options` proprietà.

Tipo: Booleano

Obbligatorio: no

Default: `True`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AddDefaultAuthorizerToCorsPreflight`

Se le `Cors` proprietà `DefaultAuthorizer` and sono impostate, l'impostazione `AddDefaultAuthorizerToCorsPreflight` farà sì che l'autorizzatore predefinito venga aggiunto alla `Options` proprietà nella sezione OpenAPI.

Tipo: Booleano

Obbligatorio: no

Predefinito: True

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ApiKeyRequired`

Se impostato su true, è necessaria una chiave API per tutti gli eventi API. Per ulteriori informazioni sulle chiavi API, consulta Creare e utilizzare piani di utilizzo con chiavi API nella Guida per sviluppatori di API Gateway.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Authorizers`

L'autorizzatore utilizzato per controllare l'accesso all'API API Gateway.

Per ulteriori informazioni, consulta Controlla l'accesso alle API con il tuo AWS SAM modello.

Tipo: CognitoAuthorizer| | LambdaTokenAuthorizer| LambdaRequestAuthorizer AWS_IAM

Obbligatorio: no

Default: nessuna

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Note aggiuntive: SAM aggiunge gli Authorizers alla OpenApi definizione di un'Api.

`DefaultAuthorizer`

Specificare un autorizzatore predefinito per un'API API Gateway, che verrà utilizzato per autorizzare le chiamate API per impostazione predefinita.

> **Note**
>
> Se l'Api EventSource per la funzione associata a questa API è configurata per utilizzare le autorizzazioni IAM, questa proprietà deve essere impostata su`AWS_IAM`, altrimenti si verificherà un errore.

- Tipo: stringa

Obbligatorio: no

Default: nessuna

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`InvokeRole`

Imposta le credenziali di integrazione per tutte le risorse e i metodi su questo valore.

`CALLER_CREDENTIALS`maps to`arn:aws:iam::*:<user>/*`, che utilizza le credenziali del chiamante per richiamare l'endpoint.

Valori validi:,, `CALLER_CREDENTIALS` `NONE` `IAMRoleArn`

- Tipo: stringa

Obbligatorio: no

Default: `CALLER_CREDENTIALS`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ResourcePolicy`

Configura la politica delle risorse per tutti i metodi e i percorsi su un'API.

Tipo: ResourcePolicyStatement

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Note aggiuntive: Questa impostazione può essere definita anche individualmente `AWS::Serverless::Function` utilizzandoApiFunctionAuth. È necessario per APIs con`EndpointConfiguration: PRIVATE`.

`UsagePlan`

Configura un piano di utilizzo associato a questa API. Per ulteriori informazioni sui piani di utilizzo, consulta Create and Use Plans with API Keys nella API Gateway Developer Guide.

Questa AWS SAM proprietà genera tre CloudFormation risorse aggiuntive quando è impostata: an AWS::ApiGateway::UsagePlanAWS::ApiGateway::UsagePlanKey, an e an AWS::ApiGateway::ApiKey. Per informazioni su questo scenario, vedereUsagePlan la proprietà è specificata. Per informazioni generali sulle CloudFormation risorse generate, vedereCloudFormation Risorse generate per AWS SAM.

Tipo: ApiUsagePlan

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

CognitoAuth

Esempio di autenticazione Cognito

YAML

```
Auth:
  Authorizers:
    MyCognitoAuth:
     UserPoolArn:
       Fn::GetAtt:
         - MyUserPool
         - Arn
     AuthType: "COGNITO_USER_POOLS"
  DefaultAuthorizer: MyCognitoAuth
  InvokeRole: CALLER_CREDENTIALS
  AddDefaultAuthorizerToCorsPreflight: false
  ApiKeyRequired: false
  ResourcePolicy:
    CustomStatements: [{
      "Effect": "Allow",
      "Principal": "*",
      "Action": "execute-api:Invoke",
      "Resource": "execute-api:/Prod/GET/pets",
      "Condition": {
          "IpAddress": {
              "aws:SourceIp": "1.2.3.4"
          }
        }
    }]
    IpRangeDenylist:
      - "10.20.30.40"
```

AWS IAM

AWS Esempio IAM

YAML

```
Auth:
  Authorizers: AWS_IAM
```

ApiUsagePlan

Configura un piano di utilizzo per un'API API Gateway. Per ulteriori informazioni sui piani di utilizzo, consulta Creare e utilizzare piani di utilizzo con chiavi API nella Guida per sviluppatori di API Gateway.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  CreateUsagePlan: String
  Description: String
  Quota: QuotaSettings
  Tags: List
  Throttle: ThrottleSettings
  UsagePlanName: String
```

Properties

`CreateUsagePlan`

Determina come è configurato questo piano di utilizzo. I valori validi sono `PER_API`, `SHARED` e `NONE`.

`PER_API`crea AWS::ApiGateway::UsagePlanAWS::ApiGateway::ApiKeye AWS::ApiGateway::UsagePlanKeyrisorse specifiche per questa API. Queste risorse hanno la logica IDs di `<api-logical-id>``UsagePlan``<api-logical-id>``ApiKey`, e`<api-logical-id>``UsagePlanKey`, rispettivamente.

`SHARED`crea AWS::ApiGateway::UsagePlane AWS::ApiGateway::UsagePlanKeyrisorse condivise tra qualsiasi API che include `CreateUsagePlan: SHARED` anche lo stesso AWS SAM modello. AWS::ApiGateway::ApiKey Queste risorse hanno la logica IDs di

`ServerlessUsagePlanServerlessApiKey`, `eServerlessUsagePlanKey`, rispettivamente. Se si utilizza questa opzione, si consiglia di aggiungere una configurazione aggiuntiva per questo piano di utilizzo su una sola risorsa API per evitare definizioni contrastanti e uno stato incerto.

`NONE`disabilita la creazione o l'associazione di un piano di utilizzo con questa API. Ciò è necessario solo se `SHARED` o `PER_API` è specificato in. Sezione Globals del modello AWS SAM

Valori validi: `PER_API`, `SHARED` e `NONE`

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Description`

Una descrizione del piano di utilizzo.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una `AWS::ApiGateway::UsagePlan` risorsa.

`Quota`

Consente di configurare il numero di richieste che possono essere eseguite dagli utenti in un determinato intervallo.

Tipo: QuotaSettings

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Quota` proprietà di una `AWS::ApiGateway::UsagePlan` risorsa.

`Tags`

Un array di tag arbitrari (coppie chiave-valore) da associare al piano di utilizzo.

Questa proprietà utilizza il tipo di CloudFormation tag.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Tags` proprietà di una `AWS::ApiGateway::UsagePlan` risorsa.

`Throttle`

Consente di configurare il tasso di richiesta complessivo (richieste medie al secondo) e la capacità di ottimizzazione.

Tipo: ThrottleSettings

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Throttle` proprietà di una `AWS::ApiGateway::UsagePlan` risorsa.

`UsagePlanName`

Un nome per il piano di utilizzo.

Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `UsagePlanName` proprietà di una `AWS::ApiGateway::UsagePlan` risorsa.

Esempi

UsagePlan

Di seguito è riportato un esempio di piano di utilizzo.

YAML

```
Auth:
  UsagePlan:
```

```
    CreateUsagePlan: PER_API
    Description: Usage plan for this API
    Quota:
      Limit: 500
      Period: MONTH
    Throttle:
      BurstLimit: 100
      RateLimit: 50
    Tags:
      - Key: TagName
        Value: TagValue
```

CognitoAuthorizer

Definisci un autorizzatore del pool di utenti Amazon Cognito.

Per maggiori informazioni ed esempi, consulta Controlla l'accesso alle API con il tuo AWS SAM modello.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
AuthorizationScopes: List
Identity: CognitoAuthorizationIdentity
UserPoolArn: String
```

Properties

`AuthorizationScopes`

Elenco degli ambiti di autorizzazione per questo autorizzatore.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Identity`

Questa proprietà può essere utilizzata per specificare una richiesta `IdentitySource` in entrata per un autorizzatore.

Tipo: CognitoAuthorizationIdentity

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`UserPoolArn`

Può riferirsi a un utente: pool/specify un userpool (arn) a cui si desidera aggiungere questo autorizzatore cognito

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

Esempi

CognitoAuth

Esempio di autenticazione con Cognito

YAML

```
Auth:
  Authorizers:
    MyCognitoAuth:
      AuthorizationScopes:
        - scope1
        - scope2
      UserPoolArn:
        Fn::GetAtt:
          - MyCognitoUserPool
```

```
          - Arn
      Identity:
        Header: MyAuthorizationHeader
        ValidationExpression: myauthvalidationexpression
```

CognitoAuthorizationIdentity

Questa proprietà può essere utilizzata per specificare una richiesta IdentitySource in entrata per un autorizzatore. Per ulteriori informazioni su, IdentitySource consulta l'estensione ApiGateway OpenApi Authorizer.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Header: String
ReauthorizeEvery: Integer
ValidationExpression: String
```

Properties

`Header`

Specificate il nome dell'intestazione per Authorization nella definizione. OpenApi

- Tipo: stringa

Obbligatorio: no

Impostazione predefinita: autorizzazione

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ReauthorizeEvery`

Il periodo time-to-live (TTL), in secondi, che specifica per quanto tempo API Gateway memorizza nella cache i risultati dell'autorizzazione. Se il valore specificato è maggiore di 0, API Gateway

memorizza nella cache le risposte di autorizzazione. Per impostazione predefinita, API Gateway imposta questa proprietà su 300. Il valore massimo è 3600, ovvero un'ora.

Tipo: integer

Obbligatorio: no

Valore predefinito: 300

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ValidationExpression`

Specificare un'espressione di convalida per convalidare l'identità in ingresso

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

Esempi

CognitoAuthIdentity

YAML

```
Identity:
  Header: MyCustomAuthHeader
  ValidationExpression: Bearer.*
  ReauthorizeEvery: 30
```

LambdaRequestAuthorizer

Configura un Lambda Authorizer per controllare l'accesso alla tua API con una funzione Lambda.

Per maggiori informazioni ed esempi, consulta Controlla l'accesso alle API con il tuo AWS SAM modello.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
DisableFunctionDefaultPermissions: Boolean
FunctionArn: String
FunctionInvokeRole: String
FunctionPayloadType: String
Identity: LambdaRequestAuthorizationIdentity
```

Properties

`DisableFunctionDefaultPermissions`

`true`Specificare per AWS SAM impedire la creazione automatica di una `AWS::Lambda::Permissions` risorsa per fornire le autorizzazioni tra la `AWS::Serverless::Api` risorsa e la funzione Lambda dell'autorizzatore.

Valore predefinito: `false`

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`FunctionArn`

Specificare la funzione ARN della funzione Lambda che fornisce l'autorizzazione per l'API.

> **Note**
>
> AWS SAM creerà automaticamente una `AWS::Lambda::Permissions` risorsa quando `FunctionArn` viene specificato per. `AWS::Serverless::Api` La `AWS::Lambda::Permissions` risorsa fornisce le autorizzazioni tra l'API e la funzione Lambda dell'autorizzazione.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`FunctionInvokeRole`

Aggiunge le credenziali dell'autorizzatore alla OpenApi definizione dell'autorizzatore Lambda.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`FunctionPayloadType`

Questa proprietà può essere utilizzata per definire il tipo di Lambda Authorizer per un'API.

Valori validi: `TOKEN` o `REQUEST`

- Tipo: stringa

Obbligatorio: no

Default: `TOKEN`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Identity`

Questa proprietà può essere utilizzata per specificare una richiesta `IdentitySource` in entrata per un autorizzatore. Questa proprietà è richiesta solo se è impostata su. `FunctionPayloadType` `REQUEST`

Tipo: LambdaRequestAuthorizationIdentity

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

LambdaRequestAuth

YAML

```
Authorizers:
  MyLambdaRequestAuth:
    FunctionPayloadType: REQUEST
    FunctionArn:
      Fn::GetAtt:
        - MyAuthFunction
        - Arn
    FunctionInvokeRole:
      Fn::GetAtt:
        - LambdaAuthInvokeRole
        - Arn
    Identity:
      Headers:
        - Authorization1
```

LambdaRequestAuthorizationIdentity

Questa proprietà può essere utilizzata per specificare una richiesta IdentitySource in entrata per un autorizzatore. Per ulteriori informazioni su, IdentitySource consulta l'estensione ApiGateway OpenApi Authorizer.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  Context: List
  Headers: List
  QueryStrings: List
  ReauthorizeEvery: Integer
  StageVariables: List
```

Properties

`Context`

Converte le stringhe di contesto fornite nelle espressioni di mappatura del formato. `context.contextString`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`Headers`

Converte le intestazioni in una stringa separata da virgole di mappatura delle espressioni di formato. `method.request.header.name`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`QueryStrings`

Converte le stringhe di query fornite in stringhe separate da virgole di mappatura delle espressioni di formato. `method.request.querystring.queryString`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`ReauthorizeEvery`

Il periodo time-to-live (TTL), in secondi, che specifica per quanto tempo API Gateway memorizza nella cache i risultati dell'autorizzazione. Se il valore specificato è maggiore di 0, API Gateway memorizza nella cache le risposte di autorizzazione. Per impostazione predefinita, API Gateway imposta questa proprietà su 300. Il valore massimo è 3600, ovvero un'ora.

Tipo: integer

Obbligatorio: no

Valore predefinito: 300

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`StageVariables`

Converte le variabili di fase specificate in una stringa separata da virgole di mappatura delle espressioni di formato. `stageVariables.stageVariable`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

Esempi

LambdaRequestIdentity

YAML

```
Identity:
  QueryStrings:
    - auth
  Headers:
    - Authorization
  StageVariables:
    - VARIABLE
  Context:
    - authcontext
  ReauthorizeEvery: 100
```

LambdaTokenAuthorizer

Configura un Lambda Authorizer per controllare l'accesso alla tua API con una funzione Lambda.

Per maggiori informazioni ed esempi, consulta Controlla l'accesso alle API con il tuo AWS SAM modello.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
DisableFunctionDefaultPermissions: Boolean
FunctionArn: String
FunctionInvokeRole: String
FunctionPayloadType: String
Identity: LambdaTokenAuthorizationIdentity
```

Properties

`DisableFunctionDefaultPermissions`

`true`Specificare per AWS SAM impedire la creazione automatica di una `AWS::Lambda::Permissions` risorsa per fornire le autorizzazioni tra la `AWS::Serverless::Api` risorsa e la funzione Lambda dell'autorizzatore.

Valore predefinito: `false`

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`FunctionArn`

Specificare la funzione ARN della funzione Lambda che fornisce l'autorizzazione per l'API.

> **Note**
>
> AWS SAM creerà automaticamente una `AWS::Lambda::Permissions` risorsa quando `FunctionArn` viene specificato per. `AWS::Serverless::Api` La `AWS::Lambda::Permissions` risorsa fornisce le autorizzazioni tra l'API e la funzione Lambda dell'autorizzazione.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`FunctionInvokeRole`

Aggiunge le credenziali dell'autorizzatore alla OpenApi definizione dell'autorizzatore Lambda.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`FunctionPayloadType`

Questa proprietà può essere utilizzata per definire il tipo di Lambda Authorizer per un'Api.

Valori validi: `TOKEN` o `REQUEST`

- Tipo: stringa

Obbligatorio: no

Default: `TOKEN`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Identity`

Questa proprietà può essere utilizzata per specificare una richiesta `IdentitySource` in entrata per un autorizzatore. Questa proprietà è richiesta solo se è impostata su. `FunctionPayloadType` `REQUEST`

Tipo: LambdaTokenAuthorizationIdentity

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

LambdaTokenAuth

YAML

```
Authorizers:
  MyLambdaTokenAuth:
    FunctionArn:
      Fn::GetAtt:
        - MyAuthFunction
        - Arn
    Identity:
      Header: MyCustomAuthHeader # OPTIONAL; Default: 'Authorization'
      ValidationExpression: mycustomauthexpression # OPTIONAL
      ReauthorizeEvery: 20 # OPTIONAL; Service Default: 300
```

BasicLambdaTokenAuth

YAML

```
Authorizers:
  MyLambdaTokenAuth:
    FunctionArn:
      Fn::GetAtt:
        - MyAuthFunction
        - Arn
```

LambdaTokenAuthorizationIdentity

Questa proprietà può essere utilizzata per specificare una richiesta IdentitySource in entrata per un autorizzatore. Per ulteriori informazioni su, IdentitySource consulta l'estensione ApiGateway OpenApi Authorizer.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  Header: String
```

```
  ReauthorizeEvery: Integer
  ValidationExpression: String
```

Properties

`Header`

Specificate il nome dell'intestazione per Authorization nella definizione. OpenApi

- Tipo: stringa

Obbligatorio: no

Impostazione predefinita: autorizzazione

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ReauthorizeEvery`

Il periodo time-to-live (TTL), in secondi, che specifica per quanto tempo API Gateway memorizza nella cache i risultati dell'autorizzazione. Se il valore specificato è maggiore di 0, API Gateway memorizza nella cache le risposte di autorizzazione. Per impostazione predefinita, API Gateway imposta questa proprietà su 300. Il valore massimo è 3600, ovvero un'ora.

Tipo: integer

Obbligatorio: no

Valore predefinito: 300

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ValidationExpression`

Specificare un'espressione di convalida per convalidare l'identità in ingresso.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

Esempi

LambdaTokenIdentity

YAML

```
Identity:
  Header: MyCustomAuthHeader
  ValidationExpression: Bearer.*
  ReauthorizeEvery: 30
```

ResourcePolicyStatement

Configura una politica delle risorse per tutti i metodi e i percorsi di un'API. Per ulteriori informazioni sulle politiche delle risorse, consulta Controllare l'accesso a un'API con le politiche delle risorse di API Gateway nella Guida per sviluppatori di API Gateway.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  AwsAccountBlacklist: List
  AwsAccountWhitelist: List
  CustomStatements: List
  IntrinsicVpcBlacklist: List
  IntrinsicVpcWhitelist: List
  IntrinsicVpceBlacklist: List
  IntrinsicVpceWhitelist: List
  IpRangeBlacklist: List
  IpRangeWhitelist: List
  SourceVpcBlacklist: List
  SourceVpcWhitelist: List
```

Properties

`AwsAccountBlacklist`

Gli AWS account da bloccare.

Tipo: elenco di String

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AwsAccountWhitelist`

Gli AWS account da consentire. Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi in fondo a questa pagina.

Tipo: elenco di String

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`CustomStatements`

Un elenco di istruzioni personalizzate sulla politica delle risorse da applicare a questa API. Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi in fondo a questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IntrinsicVpcBlacklist`

L'elenco dei cloud privati virtuali (VPCs) da bloccare, in cui ogni VPC è specificato come riferimento, ad esempio un riferimento dinamico o una funzione `Ref` intrinseca. Per un esempio di utilizzo di questa proprietà, consultate la sezione Esempi in fondo a questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IntrinsicVpcWhitelist`

L'elenco di VPCs to allow, in cui ogni VPC è specificato come riferimento, ad esempio un riferimento dinamico o una funzione `Ref` intrinseca.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`IntrinsicVpceBlacklist`

L'elenco degli endpoint VPC da bloccare, in cui ogni endpoint VPC è specificato come riferimento, ad esempio un riferimento dinamico o la funzione intrinseca. `Ref`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`IntrinsicVpceWhitelist`

L'elenco degli endpoint VPC da consentire, in cui ogni endpoint VPC è specificato come riferimento, ad esempio un riferimento dinamico o la funzione intrinseca. `Ref` Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi in fondo a questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IpRangeBlacklist`

Gli indirizzi IP o gli intervalli di indirizzi da bloccare. Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi nella parte inferiore di questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IpRangeWhitelist`

Gli indirizzi IP o gli intervalli di indirizzi da consentire.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`SourceVpcBlacklist`

Gli endpoint VPC o VPC di origine da bloccare. I nomi dei VPC di origine devono iniziare con `"vpc-"` e i nomi degli endpoint VPC di origine devono iniziare con. `"vpce-"` Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi nella parte inferiore di questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`SourceVpcWhitelist`

Gli endpoint VPC o VPC di origine da consentire. I nomi dei VPC di origine devono iniziare con `"vpc-"` e i nomi degli endpoint VPC di origine devono iniziare con. `"vpce-"`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

Esempi

Esempio di politica delle risorse

L'esempio seguente blocca due indirizzi IP e un VPC di origine e consente un AWS account.

YAML

```
Auth:
  ResourcePolicy:
    CustomStatements: [{
                          "Effect": "Allow",
                          "Principal": "*",
                          "Action": "execute-api:Invoke",
                          "Resource": "execute-api:/Prod/GET/pets",
                          "Condition": {
                            "IpAddress": {
                              "aws:SourceIp": "1.2.3.4"
                            }
                          }
                        }]
    IpRangeBlacklist:
      - "10.20.30.40"
      - "1.2.3.4"
    SourceVpcBlacklist:
      - "vpce-1a2b3c4d"
    AwsAccountWhitelist:
      - "111122223333"
    IntrinsicVpcBlacklist:
      - "{{resolve:ssm:SomeVPCReference:1}}"
      - !Ref MyVPC
    IntrinsicVpceWhitelist:
      - "{{resolve:ssm:SomeVPCEReference:1}}"
      - !Ref MyVPCE
```

ApiDefinition

Un documento OpenAPI che definisce l'API.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  Bucket: String
  Key: String
  Version: String
```

Properties

`Bucket`

Il nome del bucket Amazon S3 in cui è archiviato il file OpenAPI.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Bucket` proprietà del tipo di dati. `AWS::ApiGateway::RestApi` `S3Location`

`Key`

La chiave Amazon S3 del file OpenAPI.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Key` proprietà del tipo di `AWS::ApiGateway::RestApi` `S3Location` dati.

`Version`

Per gli oggetti con versione, la versione del file OpenAPI.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Version` proprietà del tipo di `AWS::ApiGateway::RestApi` `S3Location` dati.

Esempi

Definizione: esempio Uri

Esempio di definizione dell'API

YAML

```
DefinitionUri:
```

```
  Bucket: amzn-s3-demo-bucket-name
  Key: mykey-name
  Version: 121212
```

CorsConfiguration

Gestisci la condivisione delle risorse tra le origini (CORS) per il tuo API Gateway. APIs Specificate il dominio da consentire come stringa o specificate un dizionario con una configurazione Cors aggiuntiva.

> **Note**
>
> CORS richiede AWS SAM di modificare la definizione OpenAPI. Crea una definizione OpenAPI in linea per attivare `DefinitionBody` CORS. Se `CorsConfiguration` è impostato nella definizione di OpenAPI e anche a livello di proprietà, li AWS SAM unisce. Il livello di proprietà ha la precedenza sulla definizione OpenAPI.

Per ulteriori informazioni su CORS, consulta Enable CORS for an API Gateway REST API Resource nella API Gateway Developer Guide.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
AllowCredentials: Boolean
AllowHeaders: String
AllowMethods: String
AllowOrigin: String
MaxAge: String
```

Properties

`AllowCredentials`

Booleano che indica se la richiesta può contenere credenziali.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`AllowHeaders`

Stringa di intestazioni da consentire.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AllowMethods`

Stringa contenente i metodi HTTP da consentire.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AllowOrigin`

Stringa di origine da consentire. Può essere un elenco separato da virgole in formato stringa.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`MaxAge`

Stringa contenente il numero di secondi di memorizzazione nella cache della richiesta CORS Preflight.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

CorsConfiguration

Esempio di configurazione CORS. Questa è solo una parte di un file AWS SAM modello che mostra una AWS::Serverless::Api definizione con CORS configurato e un. AWS::Serverless::Function Se utilizzi un'integrazione proxy Lambda o un'integrazione con proxy HTTP, il backend deve restituire le intestazioni `Access-Control-Allow-OriginAccess-Control-Allow-Methods`, e. `Access-Control-Allow-Headers`

YAML

```
Resources:
  ApiGatewayApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: Prod
      Cors:
        AllowMethods: "'POST, GET'"
        AllowHeaders: "'X-Forwarded-For'"
        AllowOrigin: "'https://example.com'"
        MaxAge: "'600'"
        AllowCredentials: true
  ApiFunction: # Adds a GET method at the root resource via an Api event
    Type: AWS::Serverless::Function
    Properties:
      Events:
        ApiEvent:
          Type: Api
          Properties:
            Path: /
            Method: get
            RestApiId:
              Ref: ApiGatewayApi
      Runtime: python3.10
```

```
      Handler: index.handler
      InlineCode: |
        import json
        def handler(event, context):
          return {
          'statusCode': 200,
          'headers': {
            'Access-Control-Allow-Headers': 'Content-Type',
            'Access-Control-Allow-Origin': 'https://example.com',
            'Access-Control-Allow-Methods': 'POST, GET'
            },
          'body': json.dumps('Hello from Lambda!')
          }
```

DomainConfiguration

Configura un dominio personalizzato per un'API.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
AccessAssociation: DomainAccessAssociation
BasePath: List
CertificateArn: String
DomainName: String
EndpointConfiguration: String
MutualTlsAuthentication: MutualTlsAuthentication
NormalizeBasePath: Boolean
OwnershipVerificationCertificateArn: String
Policy: Json
Route53: Route53Configuration
SecurityPolicy: String
```

Properties

`AccessAssociation`

La configurazione richiesta per generare `AWS::ApiGateway::DomainNameAccessAssociation` una risorsa.

AWS SAM genera una AWS::ApiGateway::DomainNameAccessAssociationrisorsa quando questa proprietà è impostata. Per informazioni sulle CloudFormation risorse generate, vedereCloudFormation Risorse generate per AWS SAM.

Tipo: DomainAccessAssociation

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`BasePath`

Un elenco dei percorsi di base da configurare con il nome di dominio Amazon API Gateway.

Tipo: elenco

Obbligatorio: no

Predefinito:/

CloudFormation compatibilità: questa proprietà è simile alla `BasePath` proprietà di una `AWS::ApiGateway::BasePathMapping` risorsa. AWS SAM crea più `AWS::ApiGateway::BasePathMapping` risorse, una per quelle `BasePath` specificate in questa proprietà.

`CertificateArn`

L'Amazon Resource Name (ARN) di un certificato AWS gestito, endpoint di questo nome di dominio. AWS Certificate Manager è l'unica fonte supportata.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è simile alla `CertificateArn` proprietà di una `AWS::ApiGateway::DomainName` risorsa. Se `EndpointConfiguration` è impostato su `REGIONAL` (il valore predefinito), viene `CertificateArn` mappato a RegionalCertificateArninAWS::ApiGateway::DomainName. Se `EndpointConfiguration` è impostato su`EDGE`, viene `CertificateArn` mappato a CertificateArndentroAWS::ApiGateway::DomainName. Se `EndpointConfiguration` è impostato su`PRIVATE`, questa proprietà viene passata alla risorsa AWS::ApiGateway::DomainNameV2.

Note aggiuntive: per un `EDGE` endpoint, è necessario creare il certificato nella `us-east-1` AWS regione.

`DomainName`

Il nome di dominio personalizzato per l'API API Gateway. Le lettere maiuscole non sono supportate.

AWS SAM genera una AWS::ApiGateway::DomainNamerisorsa quando questa proprietà è impostata. Per informazioni su questo scenario, vedereDomainName la proprietà è specificata. Per informazioni sulle CloudFormation risorse generate, vedereCloudFormation Risorse generate per AWS SAM.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `DomainName` proprietà di una `AWS::ApiGateway::DomainName` risorsa o a `AWS::ApiGateway::DomainNameV2`quando EndpointConfiguration è impostata su`PRIVATE`.

`EndpointConfiguration`

Definisce il tipo di endpoint API Gateway da mappare al dominio personalizzato. Il valore di questa proprietà determina il modo in cui la `CertificateArn` proprietà viene mappata. CloudFormation

Valori validi: `EDGE`, `REGIONAL` o `PRIVATE`

- Tipo: stringa

Obbligatorio: no

Default: `REGIONAL`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`MutualTlsAuthentication`

La configurazione di autenticazione TLS (Mutual Transport Layer Security) per un nome di dominio personalizzato.

Tipo: MutualTlsAuthentication

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MutualTlsAuthentication` proprietà di una `AWS::ApiGateway::DomainName` risorsa.

`NormalizeBasePath`

Indica se i caratteri non alfanumerici sono consentiti nei percorsi di base definiti dalla proprietà. `BasePath` Se impostato su`True`, i caratteri non alfanumerici vengono rimossi dai percorsi di base.

Utilizzare con la proprietà. `NormalizeBasePath` `BasePath`

Tipo: Booleano

Obbligatorio: no

Impostazione predefinita: True

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`OwnershipVerificationCertificateArn`

L'ARN del certificato pubblico emesso da ACM per convalidare la proprietà del dominio personalizzato. Richiesto solo quando si configura il TLS reciproco e si specifica un ARN di certificato CA privato o importato da ACM per. `CertificateArn`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `OwnershipVerificationCertificateArn` proprietà di una risorsa. `AWS::ApiGateway::DomainName`

`Policy`

La policy IAM da collegare al nome di dominio API Gateway. Applicabile solo quando `EndpointConfiguration` è impostato su`PRIVATE`.

Type: Json

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Policy` proprietà di una `AWS::ApiGateway::DomainNameV2` risorsa quando `EndpointConfiguration` è impostata su`PRIVATE`. Per esempi di documenti politici validi, vedere AWS::ApiGateway::DomainNameV2.

`Route53`

Definisce una configurazione Amazon Route 53.

Tipo: configurazione Route53

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`SecurityPolicy`

La versione TLS più la suite di crittografia per questo nome di dominio.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SecurityPolicy` proprietà di una `AWS::ApiGateway::DomainName` risorsa o a `AWS::ApiGateway::DomainNameV2`when `EndpointConfiguration` è impostata su. `PRIVATE` Per gli `PRIVATE` endpoint, è supportato solo TLS_1_2.

Esempi

DomainName

DomainName esempio

YAML

```
Domain:
  DomainName: www.example.com
  CertificateArn: arn-example
```

```
  EndpointConfiguration: EDGE
  Route53:
    HostedZoneId: Z1PA6795UKMFR9
  BasePath:
    - foo
    - bar
```

DomainAccessAssociation

Configura un'associazione di accesso al dominio tra un'origine di associazione di accesso e un nome di dominio personalizzato privato. L'unica fonte di associazione di accesso supportata è un ID endpoint VPC. Per ulteriori informazioni, consulta Nomi di dominio personalizzati per uso privato APIs in API Gateway.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
VpcEndpointId: String
```

Properties

`VpcEndpointId`

L'ID endpoint dell'endpoint dell'interfaccia VPC associato al servizio VPC API Gateway.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `AccessAssociationSource` proprietà di una risorsa. `AWS::ApiGateway::DomainNameAccessAssociation`

Route53Configuration

Configura i set di record Route53 per un'API.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
DistributionDomainName: String
EvaluateTargetHealth: Boolean
HostedZoneId: String
HostedZoneName: String
IpV6: Boolean
Region: String
SetIdentifier: String
VpcEndpointDomainName: String
VpcEndpointHostedZoneId: String
```

Properties

`DistributionDomainName`

Configura una distribuzione personalizzata del nome di dominio personalizzato dell'API.

- Tipo: stringa

Obbligatorio: no

Impostazione predefinita: utilizza la distribuzione API Gateway.

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `DNSName` proprietà di una `AWS::Route53::RecordSetGroup AliasTarget` risorsa.

Note aggiuntive: il nome di dominio di una CloudFrontdistribuzione.

`EvaluateTargetHealth`

Quando EvaluateTargetHealth è vero, un record alias eredita lo stato della AWS risorsa di riferimento, ad esempio un sistema di bilanciamento del carico Elastic Load Balancing o un altro record nella zona ospitata.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di una risorsa. `EvaluateTargetHealth` `AWS::Route53::RecordSetGroup AliasTarget`

Note aggiuntive: non è possibile impostare su EvaluateTargetHealth true quando l'alias target è una CloudFront distribuzione.

`HostedZoneId`

L'ID della zona ospitata in cui creare i record.

Specifica `HostedZoneName` o `HostedZoneId`, ma non entrambi. Se disponi di più zone ospitate con lo stesso nome di dominio, devi specificare la zona ospitata utilizzando `HostedZoneId`.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `HostedZoneId` proprietà di una `AWS::Route53::RecordSetGroup RecordSet` risorsa.

`HostedZoneName`

Il nome della zona ospitata in cui desideri creare record.

Specifica `HostedZoneName` o `HostedZoneId`, ma non entrambi. Se disponi di più zone ospitate con lo stesso nome di dominio, devi specificare la zona ospitata utilizzando `HostedZoneId`.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `HostedZoneName` proprietà di una `AWS::Route53::RecordSetGroup RecordSet` risorsa.

`IpV6`

Quando questa proprietà è impostata, AWS SAM crea una `AWS::Route53::RecordSet` risorsa e imposta Type su `AAAA` quella fornita HostedZone.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Region`

Solo set di record di risorse basati sulla latenza: la EC2 regione Amazon in cui hai creato la risorsa a cui si riferisce questo set di record di risorse. La risorsa è in genere una AWS risorsa, EC2 ad esempio un'istanza o un sistema di bilanciamento del carico ELB, a cui si fa riferimento tramite un indirizzo IP o un nome di dominio DNS, a seconda del tipo di record.

Quando Amazon Route 53 riceve una query DNS per un nome e un tipo di dominio per i quali sono stati creati set di record di risorse di latenza, Route 53 seleziona il set di record di risorse di latenza con la latenza più bassa tra l'utente finale e la regione Amazon associata. EC2 Route 53 restituisce il valore associato al set di record della risorsa selezionato.

Tenere presente quanto segue:

- È possibile specificare un `ResourceRecord` per set di record di risorse di latenza.
- Puoi creare un solo set di record di risorse di latenza per ogni EC2 regione Amazon.
- Non è necessario creare set di record di risorse di latenza per tutte le EC2 regioni Amazon. Route 53 sceglie la Regione con la migliore latenza tra quelle per cui hai creato i set di record della risorsa di latenza.
- Non è possibile creare set di record di risorse non di latenza le cui proprietà `Name` e `Type` abbiano lo stesso valore dei set di record di risorse di latenza.

▬Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Region` proprietà di un tipo di `AWS::Route53::RecordSetGroup` `RecordSet` dati.

`SetIdentifier`

Set di record di risorse con un criterio di routing non semplice: un identificatore che differenzia tra più set di record di risorse che hanno la stessa combinazione di nome e tipo, ad esempio più set di record di risorse ponderati denominati acme.example.com di tipo A. In un gruppo di set di record di risorse con lo steso nome e lo stesso tipo, il valore di `SetIdentifier` deve essere univoco per ogni set di record di risorse.

Per informazioni sulle politiche di routing, consulta la sezione Choose a routing policy nella Amazon Route 53 Developer Guide.

▬Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SetIdentifier` proprietà di un tipo di `AWS::Route53::RecordSetGroup` `RecordSet` dati.

`VpcEndpointDomainName`

Un nome DNS dell'endpoint dell'interfaccia VPC associato al servizio VPC API Gateway. Questa proprietà è richiesta solo per i domini privati.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `DNSName` proprietà di un `AWS::Route53::RecordSet` `AliasTarget` campo.

`VpcEndpointHostedZoneId`

L'ID della zona ospitata dell'endpoint dell'interfaccia VPC associato al servizio VPC API Gateway. Questa proprietà è richiesta solo per i domini privati.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `HostedZoneId` proprietà di un `AWS::Route53::RecordSet` `AliasTarget` campo.

Esempi

Esempio di base

In questo esempio, configuriamo un dominio personalizzato e i set di record Route 53 per la nostra API.

YAML

```
Resources:
  MyApi:
    Type: AWS::Serverless::Api
    Properties:
```

```
      StageName: Prod
      Domain:
        DomainName: www.example.com
        CertificateArn: arn:aws:acm:us-east-1:123456789012:certificate/
abcdef12-3456-7890-abcd-ef1234567890
        EndpointConfiguration: REGIONAL
        Route53:
          HostedZoneId: ABCDEFGHIJKLMNOP
```

EndpointConfiguration

Il tipo di endpoint di un'API REST.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  IpAddressType: String
  Type: String
  VPCEndpointIds: List
```

Properties

`IpAddressType`

I tipi di indirizzi IP che possono richiamare un'API (). RestApi

Valori validi: `ipv4` o `dualstack`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `IpAddressType` proprietà del tipo di `AWS::ApiGateway::RestApi` `EndpointConfiguration` dati.

`Type`

Il tipo di endpoint di un'API REST.

Valori validi: `EDGE` o o `REGIONAL` `PRIVATE`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Types` proprietà del tipo di `AWS::ApiGateway::RestApi` `EndpointConfiguration` dati.

`VPCEndpointIds`

Un elenco di endpoint VPC IDs di un'API REST su cui creare alias Route53.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `VpcEndpointIds` proprietà del tipo di dati. `AWS::ApiGateway::RestApi` `EndpointConfiguration`

Esempi

EndpointConfiguration

Esempio di configurazione degli endpoint

YAML

```
EndpointConfiguration:
  Type: PRIVATE
  VPCEndpointIds:
    - vpce-123a123a
    - vpce-321a321a
```

AWS::Serverless::Application

Incorpora un'applicazione serverless da AWS Serverless Application Repositoryo da un bucket Amazon S3 come applicazione annidata. Le applicazioni annidate vengono distribuite come risorse annidate, che possono contenere molte altre AWS::CloudFormation::Stackrisorse, incluse altre risorse. AWS::Serverless::Application

> **Note**
>
> Quando esegui la distribuzione su AWS CloudFormation, AWS SAM trasforma le tue risorse in risorse. AWS SAM CloudFormation Per ulteriori informazioni, consulta CloudFormation Risorse generate per AWS SAM.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Type: AWS::Serverless::Application
Properties:
  Location: String | ApplicationLocationObject
  NotificationARNs: List
  Parameters: Map
  Tags: Map
  TimeoutInMinutes: Integer
```

Properties

`Location`

URL del modello, percorso del file o oggetto di posizione di un'applicazione annidata.

Se viene fornito un URL del modello, deve seguire il formato specificato nella CloudFormation TemplateUrl documentazione e contenere un modello SAM CloudFormation o valido. An ApplicationLocationObject può essere utilizzato per specificare un'applicazione che è stata pubblicata su AWS Serverless Application Repository.

Se viene fornito un percorso di file locale, il modello deve passare attraverso il flusso di lavoro che include il `sam package` comando `sam deploy` or, affinché l'applicazione venga trasformata correttamente.

Tipo: String | ApplicationLocationObject

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà è simile alla `TemplateURL` proprietà di una `AWS::CloudFormation::Stack` risorsa. La CloudFormation versione non richiede un ApplicationLocationObject per recuperare un'applicazione da. AWS Serverless Application Repository

`NotificationARNs`

Un elenco di argomenti Amazon SNS esistenti a cui vengono inviate le notifiche sugli eventi dello stack.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `NotificationARNs` proprietà di una `AWS::CloudFormation::Stack` risorsa.

`Parameters`

Valori dei parametri dell'applicazione.

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Parameters` proprietà di una `AWS::CloudFormation::Stack` risorsa.

`Tags`

Una mappa (da stringa a stringa) che specifica i tag da aggiungere a questa applicazione. Le chiavi e i valori sono limitati ai caratteri alfanumerici. Le chiavi possono contenere da 1 a 127 caratteri Unicode e non possono avere come prefisso aws:. I valori possono avere una lunghezza compresa tra 1 e 255 caratteri Unicode.

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `Tags` proprietà di una `AWS::CloudFormation::Stack` risorsa. La proprietà Tags in SAM è costituita da coppie Key:Value; in CloudFormation essa è costituita da un elenco di oggetti Tag. Quando lo stack viene creato, SAM aggiungerà automaticamente un `lambda:createdBy:SAM` tag a questa applicazione. Inoltre, se questa applicazione

proviene da AWS Serverless Application Repository, SAM invierà automaticamente anche i due tag aggiuntivi `serverlessrepo:applicationId:`*`ApplicationId`* `eserverlessrepo:semanticVersion:`*`SemanticVersion`*.

`TimeoutInMinutes`

Periodo di tempo, in minuti, che CloudFormation attende che lo stack annidato raggiunga lo stato. `CREATE_COMPLETE` Il valore predefinito non prevede alcun timeout. Quando CloudFormation rileva che lo stack nidificato ha raggiunto `CREATE_COMPLETE` lo stato, contrassegna la risorsa dello stack nidificato come nello stack principale e riprende `CREATE_COMPLETE` a creare lo stack principale. Se il periodo di timeout scade prima del raggiungimento dello stack nidificato`CREATE_COMPLETE`, CloudFormation contrassegna lo stack nidificato come fallito e ripristina sia lo stack nidificato che lo stack principale.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di una risorsa. `TimeoutInMinutes` `AWS::CloudFormation::Stack`

Valori restituiti

Ref

Quando l'ID logico di questa risorsa viene fornito alla funzione `Ref` intrinseca, restituisce il nome della risorsa sottostante`AWS::CloudFormation::Stack`.

Per ulteriori informazioni sull'uso della `Ref` funzione, consulta `Ref`la Guida per l'AWS CloudFormation utente.

Fn: GetAtt

`Fn::GetAtt` restituisce un valore per un attributo specificato di questo tipo. Di seguito sono riportati gli attributi disponibili e i valori restituiti di esempio.

Per ulteriori informazioni sull'utilizzo`Fn::GetAtt`, consulta `Fn::GetAtt`la Guida per l'AWS CloudFormation utente.

`Outputs.ApplicationOutputName`

Il valore dello stack in uscita con il nome*`ApplicationOutputName`*.

Esempi

Applicazione SAR

Applicazione che utilizza un modello dal Serverless Application Repository

YAML

```
Type: AWS::Serverless::Application
Properties:
  Location:
    ApplicationId: 'arn:aws:serverlessrepo:us-east-1:012345678901:applications/my-
application'
    SemanticVersion: 1.0.0
  Parameters:
    StringParameter: parameter-value
    IntegerParameter: 2
```

Applicazione normale

Applicazione da un URL S3

YAML

```
Type: AWS::Serverless::Application
Properties:
  Location: https://s3.amazonaws.com/sam-s3-demo-bucket/template.yaml
```

ApplicationLocationObject

Un'applicazione che è stata pubblicata su AWS Serverless Application Repository.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  ApplicationId: String
  SemanticVersion: String
```

Properties

`ApplicationId`

Il nome della risorsa Amazon (ARN) dell'applicazione.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`SemanticVersion`

La versione semantica dell'applicazione.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

la mia applicazione

Esempio di oggetto di localizzazione dell'applicazione

YAML

```
Location:
  ApplicationId: 'arn:aws:serverlessrepo:us-east-1:012345678901:applications/my-
application'
  SemanticVersion: 1.0.0
```

AWS::Serverless::CapacityProvider

Crea un provider di capacità per AWS Lambda funzioni che consente l'esecuzione di istanze gestite Lambda su istanze Amazon Elastic Compute Cloud di proprietà del cliente. Questa risorsa fa parte

della funzionalità Lambda Managed Instances, che fornisce l'ottimizzazione dei costi per carichi di lavoro Lambda su larga scala utilizzando i modelli di prezzo di Amazon. EC2

Il provider di capacità gestisce il ciclo di vita delle EC2 istanze Amazon e fornisce l'infrastruttura necessaria per l'esecuzione delle funzioni Lambda su risorse di elaborazione di proprietà del cliente, mantenendo al contempo il modello di programmazione serverless.

> **Note**
>
> Quando esegui la distribuzione su, trasforma le tue risorse in risorse. AWS CloudFormation AWS SAM AWS SAM CloudFormation Per ulteriori informazioni, consulta CloudFormation Risorse generate per AWS SAM.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Type: AWS::Serverless::CapacityProvider
Properties:
  CapacityProviderName: String
  VpcConfig: VpcConfig
  OperatorRole: String
  Tags: Map
  PropagateTags: Boolean
  InstanceRequirements: InstanceRequirements
  ScalingConfig: ScalingConfig
  KmsKeyArn: String
```

Properties

`CapacityProviderName`

Nome del provider di capacità. Questo nome deve essere unico all'interno del tuo AWS account e della tua regione.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CapacityProviderName` proprietà di una `AWS::Lambda::CapacityProvider` risorsa.

`VpcConfig`

La configurazione VPC per il provider di capacità. Speciifica le sottoreti VPC e i gruppi di sicurezza in cui verranno lanciate le istanze EC2 Amazon.

Tipo: VpcConfig

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di una risorsa. `VpcConfig` `AWS::Lambda::CapacityProvider`

`OperatorRole`

L'ARN del ruolo operatore per Lambda con le autorizzazioni per creare e gestire EC2 istanze Amazon e risorse correlate nell'account cliente. Se non viene fornito, genera AWS SAM automaticamente un ruolo con le autorizzazioni necessarie.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CapacityProviderOperatorRoleArn` proprietà `PermissionsConfig` di una `AWS::Lambda::CapacityProvider` risorsa.

`Tags`

Una mappa di coppie chiave-valore da applicare al provider di capacità e alle risorse associate.

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Tags` proprietà di una `AWS::Lambda::CapacityProvider` risorsa. La `Tags` proprietà in AWS SAM è composta da coppie chiave-valore (mentre in CloudFormation questa proprietà è costituita da un elenco di oggetti Tag). Inoltre, aggiunge AWS SAM automaticamente un `lambda:createdBy:SAM` tag a questa funzione Lambda e ai ruoli predefiniti generati per questa funzione.

`PropagateTags`

Indica se passare o meno i tag dalla proprietà Tags alle risorse `AWS::Serverless::CapacityProvider` generate. Imposta questa opzione `True` per propagare i tag nelle risorse generate.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`InstanceRequirements`

Specifiche per i tipi di istanze di calcolo che il provider di capacità può utilizzare. Ciò include i requisiti di architettura `allowed` e/o i tipi di `excluded` istanza.

Tipo: InstanceRequirements

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `InstanceRequirements` proprietà di una `AWS::Lambda::CapacityProvider` risorsa.

`ScalingConfig`

La configurazione di scalabilità per il provider di capacità. Definisce in che modo il provider di capacità ridimensiona le EC2 istanze Amazon in base alla domanda.

Tipo: ScalingConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CapacityProviderScalingConfig` proprietà di una `AWS::Lambda::CapacityProvider` risorsa.

`KmsKeyArn`

L'ARN della AWS KMS chiave utilizzata per crittografare i dati inattivi e in transito per il provider di capacità.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `KmsKeyArn` proprietà di una `AWS::Lambda::CapacityProvider` risorsa.

Valori restituiti

Ref

Quando l'ID logico di questa risorsa viene fornito alla funzione `Ref` intrinseca, restituisce il nome del fornitore di capacità.

Per ulteriori informazioni sull'uso della `Ref` funzione, `Ref`consultate la Guida per l'AWS CloudFormation utente.

Fn:: GetAtt

`Fn::GetAtt` restituisce un valore per un attributo specificato di questo tipo. Di seguito sono riportati gli attributi disponibili e i valori restituiti di esempio.

Per ulteriori informazioni sull'utilizzo`Fn::GetAtt`, consulta `Fn::GetAtt`la Guida per l'AWS CloudFormation utente.

`Arn`

L'ARN del fornitore di capacità.

Esempi

Fornitore di capacità di base

L'esempio seguente crea un provider di capacità di base con configurazione VPC.

```
MyCapacityProvider:
  Type: AWS::Serverless::CapacityProvider
  Properties:
    CapacityProviderName: my-capacity-provider
    VpcConfig:
      SubnetIds:
        - subnet-12345678
        - subnet-87654321
```

```
      SecurityGroupIds:
        - sg-12345678
    Tags:
      Environment: Production
      Team: ServerlessTeam
```

Provider di capacità avanzata con scalabilità

L'esempio seguente crea un provider di capacità con requisiti di istanza personalizzati e configurazione di scalabilità.

```
AdvancedCapacityProvider:
  Type: AWS::Serverless::CapacityProvider
  Properties:
    CapacityProviderName: advanced-capacity-provider
    VpcConfig:
      SubnetIds:
        - subnet-12345678
        - subnet-87654321
      SecurityGroupIds:
        - sg-12345678
    OperatorRole: arn:aws:iam::123456789012:role/MyCapacityProviderRole
    PropagateTags: true
    InstanceRequirements:
      Architectures:
        - x86_64
      ExcludedTypes:
        - t2.micro
    ScalingConfig:
      MaxInstanceCount: 10
      ManualScalingPolicies:
        AverageCPUUtilization: 70.0
    KmsKeyArn: arn:aws:kms:us-
east-1:123456789012:key/12345678-1234-1234-1234-123456789012
    Tags:
      Environment: Production
      CostCenter: Engineering
```

VpcConfig

Configura le impostazioni VPC per un provider di capacità, incluse le sottoreti e i gruppi di sicurezza in EC2 cui verranno avviate le istanze.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
SubnetIds: List
SecurityGroupIds: List
```

Properties

`SubnetIds`

Un elenco di sottoreti IDs in cui verranno EC2 lanciate le istanze. È necessario specificare almeno una sottorete.

Tipo: elenco

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SubnetIds` proprietà `VpcConfig` di una `AWS::Lambda::CapacityProvider` risorsa.

`SecurityGroupIds`

Un elenco di gruppi di sicurezza IDs da associare alle EC2 istanze. Se non specificato, verrà utilizzato il gruppo di sicurezza predefinito per il VPC.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SecurityGroupIds` proprietà della `VpcConfig` proprietà di una `AWS::Lambda::CapacityProvider` risorsa.

Esempi

Configurazione VPC

L'esempio seguente mostra una configurazione VPC con più sottoreti e gruppi di sicurezza.

```
VpcConfig:
```

```
  SubnetIds:
    - subnet-12345678
    - subnet-87654321
  SecurityGroupIds:
    - sg-12345678
    - sg-87654321
```

InstanceRequirements

Speciifica i requisiti per le EC2 istanze che verranno lanciate dal provider di capacità, incluse le architetture e i vincoli relativi al tipo di istanza.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Architectures: List
AllowedTypes: List
ExcludedTypes: List
```

> **Note**
>
> È possibile scegliere di specificare una delle due opzioni `AllowedTypes` oppure `ExcludedTypes` quando si definiscono i requisiti di istanza per il provider di capacità, ma non entrambi.

Properties

`Architectures`

Le architetture del set di istruzioni per le istanze del provider di capacità.

Valori validi: `x86_64` o `arm64`

Tipo: elenco

Obbligatorio: no

Default: `x86_64`

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Architectures` proprietà `InstanceRequirements` di una risorsa. `AWS::Lambda::CapacityProvider`

`AllowedTypes`

Un elenco di tipi di EC2 istanza consentiti per l'istanza del provider di capacità.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `AllowedInstanceTypes` proprietà `InstanceRequirements` di una `AWS::Lambda::CapacityProvider` risorsa.

`ExcludedTypes`

Un elenco di tipi di EC2 istanze da escludere dal fornitore di capacità.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ExcludedInstanceTypes` proprietà `InstanceRequirements` di una `AWS::Lambda::CapacityProvider` risorsa.

Esempi

Configurazione dei requisiti dell'istanza

L'esempio seguente mostra i requisiti delle istanze con vincoli specifici di architettura e tipo di istanza.

```
InstanceRequirements:
  Architectures:
    - x86_64
  ExcludedTypes:
    - t2.micro
```

ScalingConfig

Configura il modo in cui il provider di capacità ridimensiona le EC2 istanze in base alla domanda, inclusi i limiti massimi di istanze e le politiche di scalabilità.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
MaxVCpuCount: Integer
AverageCPUUtilization: Double
```

Properties

`MaxVCpuCount`

Il numero massimo di v CPUs che il provider di capacità può fornire su tutte le istanze di calcolo.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaxVCpuCount` proprietà `CapacityProviderScalingConfig` di una `AWS::Lambda::CapacityProvider` risorsa.

`AverageCPUUtilization`

La percentuale media di utilizzo della CPU prevista (0-100) per le decisioni di scalabilità. Quando l'utilizzo medio della CPU supera questa soglia, il provider di capacità scalerà le istanze Amazon EC2 . Quando specificato, AWS SAM costruisce `CapacityProviderScalingConfig` una `AWS::Lambda::CapacityProvider` risorsa con set to e `ScalingMode` set to`'Manual'`. `ScalingPolicies` `[{PredefinedMetricType: 'LambdaCapacityProviderAverageCPUUtilization', TargetValue: <this value>}]`

Tipo: numero a doppia precisione

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

Scaling configuration (Configurazione dimensionamento)

L'esempio seguente mostra una configurazione di scalabilità con VCpu conteggio massimo e utilizzo medio della CPU.

```
ScalingConfig:
  MaxVCpuCount: 10
  AverageCPUUtilization: 70.0
```

AWS::Serverless::Connector

Configura le autorizzazioni tra due risorse. Per un'introduzione ai connettori, vedere. Gestione delle autorizzazioni delle risorse con i connettori AWS SAM

Per ulteriori informazioni sulle AWS CloudFormation risorse generate, vedereCloudFormation risorse generate quando si specifica AWS::Serverless::Connector.

Per fornire feedback sui connettori, invia un nuovo problema al serverless-application-model AWS GitHub repository.

> **Note**
>
> Quando lo distribuisci a AWS CloudFormation, AWS SAM trasforma le tue AWS SAM risorse in risorse. CloudFormation Per ulteriori informazioni, consulta CloudFormation Risorse generate per AWS SAM.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa una delle seguenti sintassi.

> **Note**
>
> Consigliamo di utilizzare la sintassi dei connettori incorporati per la maggior parte dei casi d'uso. Essendo incorporato nella risorsa di origine ne semplifica la lettura e la manutenzione nel tempo. Quando devi fare riferimento a una risorsa di origine che non si trova all'interno

dello stesso AWS SAM modello, ad esempio una risorsa in uno stack annidato o una risorsa condivisa, usa la `AWS::Serverless::Connector` sintassi.

Connettori incorporati

```
<source-resource-logical-id>:
  Connectors:
    <connector-logical-id:
      Properties:
        Destination: ResourceReference | List of ResourceReference
        Permissions: List
        SourceReference: SourceReference
```

AWS::Serverless::Connector

```
Type: AWS::Serverless::Connector
Properties:
  Destination: ResourceReference | List of ResourceReference
  Permissions: List
  Source: ResourceReference
```

Properties

`Destination`

La risorsa di destinazione.

Tipo: ResourceReference| Elenco di ResourceReference

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Permissions`

Il tipo di autorizzazione che la risorsa di origine può eseguire sulla risorsa di destinazione.

`Read`include azioni AWS Identity and Access Management (IAM) che consentono la lettura dei dati dalla risorsa.

`Write`include azioni IAM che consentono l'avvio e la scrittura di dati su una risorsa.

Valori validi: `Read` o `Write`

Tipo: elenco

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Source`

La risorsa di origine. Richiesto quando si utilizza la `AWS::Serverless::Connector` sintassi.

Tipo: ResourceReference

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`SourceReference`

La risorsa di origine.

> **Note**
> Da utilizzare con la sintassi dei connettori incorporati per definire proprietà aggiuntive per la risorsa di origine.

Tipo: SourceReference

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

Connettori incorporati

L'esempio seguente utilizza connettori integrati per definire una connessione `Write` dati tra una AWS Lambda funzione e una tabella Amazon DynamoDB:

```
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyTable:
    Type: AWS::Serverless::SimpleTable
  MyFunction:
    Type: AWS::Serverless::Function
    Connectors:
      MyConn:
        Properties:
          Destination:
            Id: MyTable
          Permissions:
            - Write
    ...
```

L'esempio seguente utilizza connettori incorporati per definire `Read` e `Write` autorizzazioni:

```
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Connectors:
      MyConn:
        Properties:
          Destination:
            Id: MyTable
          Permissions:
            - Read
            - Write
  MyTable:
    Type: AWS::DynamoDB::Table
    ...
```

L'esempio seguente utilizza connettori incorporati per definire una risorsa di origine con una proprietà diversa `Id` da:

```
Transform: AWS::Serverless-2016-10-31
Transform: AWS::Serverless-2016-10-31
...
Resources:
```

```
  MyApi:
    Type: AWS::Serverless::Api
    Connectors:
      ApitoLambdaConn:
        Properties:
          SourceReference:
            Qualifier: Prod/GET/foobar
          Destination:
            Id: MyTable
          Permissions:
            - Read
            - Write
  MyTable:
    Type: AWS::DynamoDB::Table
    ...
```

AWS::Serverless::Connector

L'esempio seguente utilizza la AWS::Serverless::Connector risorsa per leggere e scrivere una AWS Lambda funzione su una tabella Amazon DynamoDB:

```
MyConnector:
  Type: AWS::Serverless::Connector
  Properties:
    Source:
      Id: MyFunction
    Destination:
      Id: MyTable
    Permissions:
      - Read
      - Write
```

L'esempio seguente utilizza la AWS::Serverless::Connector risorsa per scrivere una funzione Lambda su un argomento Amazon SNS, con entrambe le risorse nello stesso modello:

```
MyConnector:
  Type: AWS::Serverless::Connector
  Properties:
    Source:
      Id: MyLambda
    Destination:
      Id: MySNSTopic
    Permissions:
```

```
      - Write
```

L'esempio seguente utilizza la AWS::Serverless::Connector risorsa per scrivere un argomento Amazon SNS su una funzione Lambda, che quindi scrive su una tabella Amazon DynamoDB, con tutte le risorse nello stesso modello:

```
Transform: AWS::Serverless-2016-10-31
Resources:
  Topic:
    Type: AWS::SNS::Topic
    Properties:
      Subscription:
        - Endpoint: !GetAtt Function.Arn
          Protocol: lambda

  Function:
    Type: AWS::Serverless::Function
    Properties:
      Runtime: nodejs16.x
      Handler: index.handler
      InlineCode: |
        const AWS = require('aws-sdk');
        exports.handler = async (event, context) => {
          const docClient = new AWS.DynamoDB.DocumentClient();
          await docClient.put({
            TableName: process.env.TABLE_NAME,
            Item: {
              id: context.awsRequestId,
              event: JSON.stringify(event)
            }
          }).promise();
        };
      Environment:
        Variables:
          TABLE_NAME: !Ref Table

  Table:
    Type: AWS::Serverless::SimpleTable

  TopicToFunctionConnector:
    Type: AWS::Serverless::Connector
    Properties:
      Source:
```

```
        Id: Topic
      Destination:
        Id: Function
      Permissions:
        - Write

  FunctionToTableConnector:
    Type: AWS::Serverless::Connector
    Properties:
      Source:
        Id: Function
      Destination:
        Id: Table
      Permissions:
        - Write
```

Di seguito è riportato il AWS CloudFormation modello trasformato dell'esempio precedente:

```
"FunctionToTableConnectorPolicy": {
  "Type": "AWS::IAM::ManagedPolicy",
  "Metadata": {
    "aws:sam:connectors": {
      "FunctionToTableConnector": {
        "Source": {
          "Type": "AWS::Lambda::Function"
        },
        "Destination": {
          "Type": "AWS::DynamoDB::Table"
        }
      }
    }
  },
  "Properties": {
    "PolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": [
            "dynamodb:PutItem",
            "dynamodb:UpdateItem",
            "dynamodb:DeleteItem",
            "dynamodb:BatchWriteItem",
```

```
            "dynamodb:PartiQLDelete",
            "dynamodb:PartiQLInsert",
            "dynamodb:PartiQLUpdate"
          ],
          "Resource": [
            {
              "Fn::GetAtt": [
                "MyTable",
                "Arn"
              ]
            },
            {
              "Fn::Sub": [
                "${DestinationArn}/index/*",
                {
                  "DestinationArn": {
                    "Fn::GetAtt": [
                      "MyTable",
                      "Arn"
                    ]
                  }
                }
              ]
            }
          ]
        }
      ]
    },
    "Roles": [
      {
        "Ref": "MyFunctionRole"
      }
    ]
  }
}
```

ResourceReference

Un riferimento a una risorsa utilizzata dal tipo di AWS::Serverless::Connector risorsa.

> **Note**
>
> Per le risorse nello stesso modello, fornisci`Id`. Per le risorse non incluse nello stesso modello, utilizzate una combinazione di altre proprietà. Per ulteriori informazioni, consulta AWS SAM riferimento del connettore.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
Arn: String
Id: String
Name: String
Qualifier: String
QueueUrl: String
ResourceId: String
RoleName: String
Type: String
```

Properties

`Arn`

L'ARN di una risorsa.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Id`

L'ID logico di una risorsa nello stesso modello.

> **Note**
>
> Quando `Id` viene specificato, se il connettore genera politiche AWS Identity and Access Management (IAM), il ruolo IAM associato a tali politiche verrà dedotto dalla risorsa`Id`. Quando non `Id` è specificato, fornisci `RoleName` la risorsa per i connettori per collegare le policy IAM generate a un ruolo IAM.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Name`

Il nome di una risorsa.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Qualifier`

Un qualificatore per una risorsa che ne restringe l'ambito. `Qualifier`sostituisce il `*` valore alla fine di un vincolo di risorse ARN. Per vedere un esempio, consulta API Gateway che richiama una funzione Lambda.

> **Note**
>
> La definizione del qualificatore varia in base al tipo di risorsa. Per un elenco dei tipi di risorse di origine e destinazione supportati, vedere. AWS SAM riferimento del connettore

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`QueueUrl`

L'URL della coda Amazon SQS. Questa proprietà si applica solo alle risorse Amazon SQS.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ResourceId`

L'ID di una risorsa. Ad esempio, l'ID API Gateway API.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`RoleName`

Il nome del ruolo associato a una risorsa.

> **Note**
>
> Quando `Id` viene specificato, se il connettore genera politiche IAM, il ruolo IAM associato a tali politiche verrà dedotto dalla risorsa`Id`. Quando non `Id` è specificato, fornisci `RoleName` la risorsa per i connettori per collegare le policy IAM generate a un ruolo IAM.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Type

Il CloudFormation tipo di risorsa. Per ulteriori informazioni, consulta il riferimento ai tipi di AWS risorse e proprietà.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

API Gateway che richiama una funzione Lambda

L'esempio seguente utilizza la AWS::Serverless::Connector risorsa per consentire ad Amazon API Gateway di richiamare una AWS Lambda funzione.

YAML

```
Transform: AWS::Serverless-2016-10-31
Resources:
  MyRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Action: sts:AssumeRole
            Principal:
              Service: lambda.amazonaws.com
      ManagedPolicyArns:
        - !Sub arn:${AWS::Partition}:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole

  MyFunction:
    Type: AWS::Lambda::Function
    Properties:
      Role: !GetAtt MyRole.Arn
      Runtime: nodejs16.x
      Handler: index.handler
      Code:
```

```
        ZipFile: |
          exports.handler = async (event) => {
            return {
              statusCode: 200,
              body: JSON.stringify({
                "message": "It works!"
              }),
            };
          };

  MyApi:
    Type: AWS::ApiGatewayV2::Api
    Properties:
      Name: MyApi
      ProtocolType: HTTP

  MyStage:
    Type: AWS::ApiGatewayV2::Stage
    Properties:
      ApiId: !Ref MyApi
      StageName: prod
      AutoDeploy: True

  MyIntegration:
    Type: AWS::ApiGatewayV2::Integration
    Properties:
      ApiId: !Ref MyApi
      IntegrationType: AWS_PROXY
      IntegrationUri: !Sub arn:aws:apigateway:${AWS::Region}:lambda:path/2015-03-31/
functions/${MyFunction.Arn}/invocations
      IntegrationMethod: POST
      PayloadFormatVersion: "2.0"

  MyRoute:
    Type: AWS::ApiGatewayV2::Route
    Properties:
      ApiId: !Ref MyApi
      RouteKey: GET /hello
      Target: !Sub integrations/${MyIntegration}

  MyConnector:
    Type: AWS::Serverless::Connector
    Properties:
      Source: # Use 'Id' when resource is in the same template
```

```
        Type: AWS::ApiGatewayV2::Api
        ResourceId: !Ref MyApi
        Qualifier: prod/GET/hello # Or "*" to allow all routes
      Destination: # Use 'Id' when resource is in the same template
        Type: AWS::Lambda::Function
        Arn: !GetAtt MyFunction.Arn
      Permissions:
        - Write

Outputs:
  Endpoint:
    Value: !Sub https://${MyApi}.execute-api.${AWS::Region}.${AWS::URLSuffix}/prod/
hello
```

SourceReference

Un riferimento a una risorsa di origine utilizzata dal tipo di AWS::Serverless::Connector risorsa.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Qualifier: String
```

Properties

`Qualifier`

Un qualificatore per una risorsa che ne restringe l'ambito. `Qualifier`sostituisce il `*` valore alla fine di un vincolo di risorse ARN.

> **Note**
>
> La definizione del qualificatore varia in base al tipo di risorsa. Per un elenco dei tipi di risorse di origine e destinazione supportati, vedere. AWS SAM riferimento del connettore

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

L'esempio seguente utilizza connettori incorporati per definire una risorsa di origine con una proprietà diversa da**`Id`**:

```
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyApi:
    Type: AWS::Serverless::Api
    Connectors:
      ApitoLambdaConn:
        Properties:
          SourceReference:
            Qualifier: Prod/GET/foobar
          Destination:
            Id: MyTable
          Permissions:
            - Read
            - Write
  MyTable:
    Type: AWS::DynamoDB::Table
    ...
```

AWS::Serverless::Function

Crea una AWS Lambda funzione, un ruolo di esecuzione AWS Identity and Access Management (IAM) e mappature delle sorgenti degli eventi che attivano la funzione.

La AWS::Serverless::Function risorsa supporta anche l'attributo `Metadata` resource, quindi puoi indicare di AWS SAM creare runtime personalizzati richiesti dall'applicazione. Per ulteriori informazioni sulla creazione di runtime personalizzati, consulta. Creazione di funzioni Lambda con runtime personalizzati in AWS SAM

> **Note**
>
> Quando esegui la distribuzione su AWS CloudFormation, AWS SAM trasforma le tue AWS SAM risorse in risorse. CloudFormation Per ulteriori informazioni, consulta CloudFormation Risorse generate per AWS SAM.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Type: AWS::Serverless::Function
Properties:
  Architectures: List
  AssumeRolePolicyDocument: JSON
  AutoPublishAlias: String
  AutoPublishAliasAllProperties: Boolean
  AutoPublishCodeSha256: String
  CapacityProviderConfig: CapacityProviderConfig
  CodeSigningConfigArn: String
  CodeUri: String | FunctionCode
  DeadLetterQueue: Map | DeadLetterQueue
  DeploymentPreference: DeploymentPreference
  Description: String
  DurableConfig: DurableConfig
  Environment: Environment
  EphemeralStorage: EphemeralStorage
  EventInvokeConfig: EventInvokeConfiguration
  Events: EventSource
  FileSystemConfigs: List
  FunctionName: String
  FunctionScalingConfig: FunctionScalingConfig
  FunctionUrlConfig: FunctionUrlConfig
  Handler: String
  ImageConfig: ImageConfig
  ImageUri: String
  InlineCode: String
  KmsKeyArn: String
  Layers: List
```

```
  LoggingConfig: LoggingConfig
  MemorySize: Integer
  PackageType: String
  PermissionsBoundary: String
  Policies: String | List | Map
  PublishToLatestPublished: Boolean
  PropagateTags: Boolean
  ProvisionedConcurrencyConfig: ProvisionedConcurrencyConfig
  RecursiveLoop: String
  ReservedConcurrentExecutions: Integer
  Role: String
  RolePath: String
  Runtime: String
  RuntimeManagementConfig: RuntimeManagementConfig
  SnapStart: SnapStart
  SourceKMSKeyArn: String
  Tags: Map
  TenancyConfig: TenancyConfig
  Timeout: Integer
  Tracing: String
  VersionDescription: String
  VersionDeletionPolicy: String
  VpcConfig: VpcConfig
```

Properties

`Architectures`

L'architettura del set di istruzioni per la funzione.

Per ulteriori informazioni su questa proprietà, consulta Architetture dei set di istruzioni Lambda nella Guida per gli sviluppatori.AWS Lambda

Valori validi: uno dei o `x86_64` `arm64`

Tipo: elenco

Obbligatorio: no

Default: `x86_64`

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Architectures` proprietà di una `AWS::Lambda::Function` risorsa.

`AssumeRolePolicyDocument`

Aggiunge un AssumeRolePolicyDocument valore predefinito creato `Role` per questa funzione. Se questa proprietà non è specificata, AWS SAM aggiunge un ruolo di assunzione predefinito per questa funzione.

Type: JSON

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `AssumeRolePolicyDocument` proprietà di una `AWS::IAM::Role` risorsa. AWS SAM aggiunge questa proprietà al ruolo IAM generato per questa funzione. Se per questa funzione viene fornito l'Amazon Resource Name (ARN) di un ruolo, questa proprietà non fa nulla.

`AutoPublishAlias`

Il nome dell'alias Lambda. Per ulteriori informazioni sugli alias Lambda, consulta Alias delle funzioni Lambda nella Developer Guide.AWS Lambda Per esempi che utilizzano questa proprietà, consulta. Implementazione graduale di applicazioni serverless con AWS SAM

AWS SAM genera AWS::Lambda::Versione AWS::Lambda::Aliasrisorse quando questa proprietà è impostata. Per informazioni su questo scenario, vedereAutoPublishAlias la proprietà è specificata. Per informazioni generali sulle CloudFormation risorse generate, vedereCloudFormation Risorse generate per AWS SAM.

Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AutoPublishAliasAllProperties`

Specifica quando ne `AWS::Lambda::Version`viene creato uno nuovo. Quando`true`, viene creata una nuova versione Lambda quando viene modificata una proprietà della funzione Lambda. Quando`false`, viene creata una nuova versione Lambda solo quando viene modificata una delle seguenti proprietà:

- `Environment`, `MemorySize`, oppure `SnapStart`.
- Qualsiasi modifica che comporti un aggiornamento della `Code` proprietà, ad esempio `CodeDictImageUri`, `oInlineCode`.

Questa proprietà `AutoPublishAlias` deve essere definita.

Se `AutoPublishCodeSha256` viene specificato anche, il suo comportamento ha la precedenza su`AutoPublishAliasAllProperties: true`.

Tipo: Booleano

Obbligatorio: no

Valore predefinito: `false`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AutoPublishCodeSha256`

Quando viene utilizzata, questa stringa funziona con il `CodeUri` valore per determinare se è necessario pubblicare una nuova versione di Lambda. Questa proprietà viene spesso utilizzata per risolvere il seguente problema di distribuzione: un pacchetto di distribuzione viene archiviato in una posizione Amazon S3 e sostituito da un nuovo pacchetto di distribuzione con codice funzione Lambda aggiornato, ma la `CodeUri` proprietà rimane invariata (al contrario del nuovo pacchetto di distribuzione caricato in una nuova posizione Amazon S3 e `CodeUri` trasferito nella nuova posizione).

Questo problema è contrassegnato da un AWS SAM modello con le seguenti caratteristiche:

- L'`DeploymentPreference`oggetto è configurato per distribuzioni graduali (come descritto in) Implementazione graduale di applicazioni serverless con AWS SAM
- La `AutoPublishAlias` proprietà è impostata e non cambia tra le distribuzioni
- La `CodeUri` proprietà è impostata e non cambia tra le distribuzioni.

In questo scenario, l'aggiornamento `AutoPublishCodeSha256` comporta la creazione corretta di una nuova versione Lambda. Tuttavia, il nuovo codice funzionale distribuito su Amazon S3 non verrà riconosciuto. Per riconoscere il nuovo codice funzionale, prendi in considerazione l'utilizzo del controllo delle versioni nel tuo bucket Amazon S3. Specificate la `Version` proprietà per la funzione Lambda e configurate il bucket per utilizzare sempre il pacchetto di distribuzione più recente.

In questo scenario, per attivare correttamente la distribuzione graduale, è necessario fornire un valore univoco per. `AutoPublishCodeSha256`

➖Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`CapacityProviderConfig`

Configura il provider di capacità a cui verranno allegate le versioni pubblicate della funzione. Ciò consente l'esecuzione della funzione su istanze EC2 di proprietà del cliente gestite da Lambda Managed Instances.

Tipo: CapacityProviderConfig

Obbligatorio: no

CloudFormation compatibilità: SAM appiattisce la proprietà passata alla proprietà di una risorsa e ricostruisce la `CapacityProviderConfig` struttura annidata. `AWS::Lambda::Function`

`CodeSigningConfigArn`

L'ARN della AWS::Lambda::CodeSigningConfigrisorsa, utilizzato per abilitare la firma del codice per questa funzione. Per ulteriori informazioni sulla firma del codice, vedereConfigura la firma del codice per la tua AWS SAM applicazione.

-Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CodeSigningConfigArn` proprietà di una `AWS::Lambda::Function` risorsa.

`CodeUri`

Il codice della funzione. I valori accettati includono:

- L'URI della funzione è Amazon S3. Ad esempio, `s3://bucket-123456789/sam-app/1234567890abcdefg`.
- Il percorso locale della funzione. Ad esempio, `hello_world/`.
- Oggetto FunctionCode.

> **Note**
>
> Se fornisci l'URI o l'FunctionCodeoggetto Amazon S3 di una funzione, devi fare riferimento a un pacchetto di distribuzione Lambda valido.

Se fornisci un percorso di file locale, usa il AWS SAMCLI per caricare il file locale al momento della distribuzione. Per ulteriori informazioni, consulta Come AWS SAM vengono caricati i file locali al momento della distribuzione.
Se si utilizzano funzioni intrinseche nella `CodeUri` proprietà, non AWS SAM sarà possibile analizzare correttamente i valori. Prendi invece in considerazione l'utilizzo di AWS::LanguageExtensions transform.

Tipo: [String |FunctionCode]

Obbligatorio: condizionale. Quando `PackageType` è impostato su`Zip`, `InlineCode` è richiesto uno dei `CodeUri` o.

CloudFormation compatibilità: questa proprietà è simile alla `Code` proprietà di una `AWS::Lambda::Function` risorsa. Le proprietà annidate di Amazon S3 sono denominate in modo diverso.

`DeadLetterQueue`

Configura un argomento Amazon Simple Notification Service (Amazon SNS) o una coda Amazon Simple Queue Service (Amazon SQS) in cui Lambda invia eventi che non può elaborare. Per ulteriori informazioni sulla funzionalità Dead-letter queue, consulta Dead-letter queues nella Developer Guide.AWS Lambda

> **Note**
>
> Se l'origine degli eventi della tua funzione Lambda è una coda Amazon SQS, configura una coda di lettere non scritte per la coda di origine, non per la funzione Lambda. La coda di lettere morte configurata per una funzione viene utilizzata per la coda di invocazione asincrona della funzione, non per le code di origine degli eventi.

Tipo: Mappa | DeadLetterQueue

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è simile alla `DeadLetterConfig` proprietà di una `AWS::Lambda::Function` risorsa. CloudFormation Nel tipo è derivato da`TargetArn`, mentre in è AWS SAM necessario passare il tipo insieme a`TargetArn`.

`DeploymentPreference`

Le impostazioni per abilitare le distribuzioni Lambda graduali.

Se viene specificato un `DeploymentPreference` oggetto, AWS SAM crea un AWS::CodeDeploy::Applicationchiamato `ServerlessDeploymentApplication` (uno per stack), un AWS::CodeDeploy::DeploymentGroupchiamato e un chiamato*<function-logical-id>*`DeploymentGroup`. AWS::IAM::Role`CodeDeployServiceRole`

Tipo: DeploymentPreference

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Vedi anche: Per ulteriori informazioni su questa proprietà, vediImplementazione graduale di applicazioni serverless con AWS SAM.

`Description`

Una descrizione della funzione.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una `AWS::Lambda::Function` risorsa.

`DurableConfig`

Configurazione per funzioni durevoli. Abilita l'esecuzione con stato con funzionalità di checkpoint e replay automatici.

Tipo: DurableConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`Environment`

La configurazione per l'ambiente di runtime.

Type: Environment

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Environment` proprietà di una `AWS::Lambda::Function` risorsa.

`EphemeralStorage`

Un oggetto che specifica lo spazio su disco, in MB, disponibile per la funzione Lambda in. `/tmp`

Per ulteriori informazioni su questa proprietà, consulta l'ambiente di esecuzione Lambda nella AWS Lambda Developer Guide.

Tipo: EphemeralStorage

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EphemeralStorage` proprietà di una `AWS::Lambda::Function` risorsa.

`EventInvokeConfig`

L'oggetto che descrive la configurazione Event Invoke su una funzione Lambda.

Tipo: EventInvokeConfiguration

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Events`

Speciifica gli eventi che attivano questa funzione. Gli eventi sono costituiti da un tipo e da un insieme di proprietà che dipendono dal tipo.

Tipo: EventSource

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`FileSystemConfigs`

Elenco di FileSystemConfigoggetti che specificano le impostazioni di connessione per un file system Amazon Elastic File System (Amazon EFS).

Se il modello contiene una AWS::EFS::MountTargetrisorsa, è necessario specificare anche un attributo di `DependsOn` risorsa per garantire che la destinazione di montaggio venga creata o aggiornata prima della funzione.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FileSystemConfigs` proprietà di una `AWS::Lambda::Function` risorsa.

`FunctionName`

Nome per la funzione. Se non specifichi un nome, viene generato un nome univoco.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FunctionName` proprietà di una `AWS::Lambda::Function` risorsa.

`FunctionScalingConfig`

Configura il comportamento di scalabilità per le funzioni Lambda in esecuzione sui provider di capacità. Definisce il numero minimo e massimo di ambienti di esecuzione.

Tipo: FunctionScalingConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FunctionScalingConfig` proprietà di una `AWS::Lambda::Function` risorsa.

`FunctionUrlConfig`

L'oggetto che descrive l'URL di una funzione. L'URL di una funzione è un endpoint HTTPS che puoi utilizzare per richiamare la tua funzione.

Per ulteriori informazioni, consulta Function URLs nella AWS Lambda Developer Guide.

Tipo: FunctionUrlConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Handler`

La funzione all'interno del codice che viene chiamata per iniziare l'esecuzione. Questa proprietà è richiesta solo se è impostata su`Zip`. `PackageType`

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Handler` proprietà di una `AWS::Lambda::Function` risorsa.

`ImageConfig`

L'oggetto utilizzato per configurare le impostazioni dell'immagine del contenitore Lambda. Per ulteriori informazioni, consulta Using container images with Lambda nella AWS Lambda Developer Guide.

Tipo: ImageConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ImageConfig` proprietà di una `AWS::Lambda::Function` risorsa.

`ImageUri`

L'URI del repository Amazon Elastic Container Registry (Amazon ECR) per l'immagine del contenitore della funzione Lambda. Questa proprietà si applica solo se è impostata su`Image`, altrimenti viene ignorata. `PackageType` Per ulteriori informazioni, consulta Using container images with Lambda nella AWS Lambda Developer Guide.

> **Note**
>
> Se la `PackageType` proprietà è impostata su`Image`, allora una delle due `ImageUri` è obbligatoria oppure è necessario creare l'applicazione con `Metadata` le voci necessarie

nel file AWS SAM modello. Per ulteriori informazioni, consulta Compilazione predefinita con AWS SAM.

La creazione dell'applicazione con `Metadata` le voci necessarie ha la precedenza sulla creazione dell'applicazione`ImageUri`, quindi se si specificano entrambe le voci `ImageUri` viene ignorata.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ImageUri` proprietà del tipo di `AWS::Lambda::Function` `Code` dati.

`InlineCode`

Il codice della funzione Lambda scritto direttamente nel modello. Questa proprietà si applica solo se è impostata su`Zip`, altrimenti viene ignorata. `PackageType`

> **Note**
>
> Se la `PackageType` proprietà è impostata su `Zip` (impostazione predefinita), `InlineCode` è richiesta una delle `CodeUri` o.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ZipFile` proprietà del tipo di `AWS::Lambda::Function` `Code` dati.

`KmsKeyArn`

L'ARN di una chiave AWS Key Management Service (AWS KMS) utilizzata da Lambda per crittografare e decrittografare le variabili di ambiente della funzione.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di una risorsa. `KmsKeyArn` `AWS::Lambda::Function`

`Layers`

L'elenco di ciò `LayerVersion` ARNs che questa funzione dovrebbe usare. L'ordine specificato qui è l'ordine in cui verranno importati durante l'esecuzione della funzione Lambda. La versione è un ARN completo che include la versione o un riferimento a una LayerVersion risorsa. Ad esempio, un riferimento a `LayerVersion` sarà `!Ref MyLayer` mentre lo sarà un ARN completo, inclusa la versione. `arn:aws:lambda:`*`region`*`:`*`account-id`*`:layer:`*`layer-name`*`:`*`version`*

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Layers` proprietà di una `AWS::Lambda::Function` risorsa.

`LoggingConfig`

Le impostazioni di configurazione di Amazon CloudWatch Logs della funzione.

Tipo: LoggingConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `LoggingConfig`proprietà di una `AWS::Lambda::Function` risorsa.

`MemorySize`

La dimensione della memoria in MB allocata per invocazione della funzione.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MemorySize` proprietà di una risorsa. `AWS::Lambda::Function`

`PackageType`

Il tipo di pacchetto di distribuzione della funzione Lambda. Per ulteriori informazioni, consulta i pacchetti di distribuzione Lambda nella AWS Lambda Developer Guide.

Note:

1. Se questa proprietà è impostata su `Zip` (impostazione predefinita), allora `InlineCode` si applica `CodeUri` o `ImageUri` viene ignorata.

2. Se questa proprietà è impostata su`Image`, viene `ImageUri` applicata solo `CodeUri` ed entrambe le proprietà `InlineCode` vengono ignorate. Il repository Amazon ECR necessario per archiviare l'immagine del contenitore della funzione può essere creato automaticamente da. AWS SAMCLI Per ulteriori informazioni, consulta sam deploy.

Valori validi: `Zip` o `Image`

- Tipo: stringa

Obbligatorio: no

Default: `Zip`

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `PackageType` proprietà di una `AWS::Lambda::Function` risorsa.

`PermissionsBoundary`

L'ARN di un limite di autorizzazioni da utilizzare per il ruolo di esecuzione di questa funzione. Questa proprietà funziona solo se il ruolo viene generato automaticamente.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `PermissionsBoundary` proprietà di una `AWS::IAM::Role` risorsa.

`Policies`

Politiche di autorizzazione per questa funzione. Le politiche verranno aggiunte al ruolo di esecuzione predefinito AWS Identity and Access Management (IAM) della funzione.

Questa proprietà accetta un singolo valore o un elenco di valori. I valori autorizzati includono:

- AWS SAM modelli di policy.
- ARNDi una politica AWS gestita o di una politica gestita dal cliente.
- Il nome di una politica AWS gestita dall'elenco seguente.

- Una politica IAM in linea formattata YAML come mappa.

> **Note**
>
> Se si imposta la `Role` proprietà, questa proprietà viene ignorata.

Tipo: Stringa | Elenco | Mappa

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è simile alla `Policies` proprietà di una `AWS::IAM::Role` risorsa.

`PublishToLatestPublished`

Speciifica se pubblicare la versione più recente della funzione quando la funzione viene aggiornata.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `PublishToLatestPublished` proprietà di una `AWS::Lambda::Function` risorsa.

`PropagateTags`

Indica se passare o meno i tag dalla `Tags` proprietà alle risorse AWS::Serverless::Function generate. `True`Specificate di propagare i tag nelle risorse generate.

Tipo: Booleano

Obbligatorio: no

Default: `False`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ProvisionedConcurrencyConfig`

La configurazione simultanea fornita dell'alias di una funzione.

> **Note**
>
> `ProvisionedConcurrencyConfig`può essere specificato solo se è impostato. `AutoPublishAlias` In caso contrario, si verifica un errore.

Tipo: ProvisionedConcurrencyConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ProvisionedConcurrencyConfig` proprietà di una `AWS::Lambda::Alias` risorsa.

`RecursiveLoop`

Lo stato della configurazione di rilevamento ricorsivo del loop della funzione.

Quando questo valore è impostato su `Allow` e Lambda rileva che la funzione viene richiamata come parte di un ciclo ricorsivo, non esegue alcuna azione.

Quando questo valore è impostato su `Terminate` e Lambda rileva che la funzione viene richiamata come parte di un ciclo ricorsivo, interrompe la chiamata della funzione e invia una notifica all'utente.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà della risorsa. `RecursiveLoop` `AWS::Lambda::Function`

`ReservedConcurrentExecutions`

Il numero massimo di esecuzioni simultanee che si desidera riservare per la funzione.

Per ulteriori informazioni su questa proprietà, consulta Lambda Function Scaling nella Developer Guide.AWS Lambda

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ReservedConcurrentExecutions` proprietà di una `AWS::Lambda::Function` risorsa.

`Role`

L'ARN di un ruolo IAM da utilizzare come ruolo di esecuzione di questa funzione.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `Role` proprietà di una `AWS::Lambda::Function` risorsa. È obbligatorio in CloudFormation ma non in AWS SAM. Se non viene specificato un ruolo, ne viene creato uno con un ID logico di`<function-logical-id>``Role`.

`RolePath`

Il percorso del ruolo di esecuzione IAM della funzione.

Usa questa proprietà quando il ruolo viene generato per te. Non utilizzare quando il ruolo è specificato con la `Role` proprietà.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Path` proprietà di una `AWS::IAM::Role` risorsa.

`Runtime`

L'identificatore del runtime della funzione. Questa proprietà è richiesta solo se la `PackageType` proprietà è impostata su`Zip`.

> **Note**
>
> Se si specifica l'`provide`identificatore per questa proprietà, è possibile utilizzare l'attributo `Metadata` resource per indicare di AWS SAM creare il runtime personalizzato richiesto da questa funzione. Per ulteriori informazioni sulla creazione di runtime personalizzati, vedere. Creazione di funzioni Lambda con runtime personalizzati in AWS SAM

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Runtime` proprietà di una `AWS::Lambda::Function` risorsa.

`RuntimeManagementConfig`

Configura le opzioni di gestione del runtime per le funzioni Lambda, come gli aggiornamenti dell'ambiente di runtime, il comportamento di rollback e la selezione di una versione di runtime specifica. Per saperne di più, consulta gli aggiornamenti del runtime di Lambda nella AWS Lambda Developer Guide.

Tipo: RuntimeManagementConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RuntimeManagementConfig` proprietà di una `AWS::Lambda::Function` risorsa.

`SnapStart`

Crea un'istantanea di ogni nuova versione della funzione Lambda. Un'istantanea è uno stato memorizzato nella cache della funzione inizializzata, incluse tutte le sue dipendenze. La funzione viene inizializzata una sola volta e lo stato memorizzato nella cache viene riutilizzato per tutte le future chiamate, migliorando le prestazioni dell'applicazione riducendo il numero di volte in cui la funzione deve essere inizializzata. Per ulteriori informazioni, consulta Migliorare le prestazioni di avvio con Lambda SnapStart nella Guida per gli AWS Lambda sviluppatori.

Tipo: SnapStart

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SnapStart` proprietà di una `AWS::Lambda::Function` risorsa.

`SourceKMSKeyArn`

Rappresenta l'ARN di una chiave KMS utilizzata per crittografare il codice della funzione ZIP del cliente.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SourceKMSKeyArn` proprietà di un `AWS::Lambda::Function` `Code` tipo di dati.

`Tags`

Una mappa (da stringa a stringa) che specifica i tag aggiunti a questa funzione. Per informazioni dettagliate sulle chiavi e i valori validi per i tag, consulta Tag Key and Value Requirements nella AWS Lambda Developer Guide.

Quando lo stack viene creato, aggiunge AWS SAM automaticamente un `lambda:createdBy:SAM` tag a questa funzione Lambda e ai ruoli predefiniti generati per questa funzione.

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `Tags` proprietà di una `AWS::Lambda::Function` risorsa. La `Tags` proprietà in AWS SAM è costituita da coppie chiave-valore (mentre in CloudFormation questa proprietà è costituita da un elenco di `Tag` oggetti). Inoltre, aggiunge AWS SAM automaticamente un `lambda:createdBy:SAM` tag a questa funzione Lambda e ai ruoli predefiniti generati per questa funzione.

`TenancyConfig`

Configurazione per la modalità di isolamento dei tenant Lambda. Assicura che gli ambienti di esecuzione non vengano mai condivisi tra tenant diversi IDs, fornendo un isolamento a livello di calcolo per le applicazioni multi-tenant.

Tipo: TenancyConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di una risorsa. `TenancyConfig` `AWS::Lambda::Function`

`Timeout`

Il tempo massimo in secondi in cui la funzione può essere eseguita prima che venga interrotta.

Tipo: integer

Obbligatorio: no

Di default: 3

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Timeout` proprietà di una `AWS::Lambda::Function` risorsa.

`Tracing`

La stringa che specifica la modalità di tracciamento X-Ray della funzione.

- `Active`— Attiva il tracciamento a raggi X per la funzione.
- `Disabled`— Disattiva X-Ray per la funzione.
- `PassThrough`— Attiva il tracciamento a raggi X per la funzione. La decisione sul campionamento è delegata ai servizi a valle.

Se specificato come `Active` o `PassThrough` e la `Role` proprietà non è impostata, AWS SAM aggiunge la `arn:aws:iam::aws:policy/AWSXrayWriteOnlyAccess` politica al ruolo di esecuzione Lambda che crea per te.

Per ulteriori informazioni su X-Ray, consulta Using AWS Lambda with AWS X-Ray nella AWS Lambda Developer Guide.

Valori validi: [`Active`| `Disabled` |`PassThrough`]

➖Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `TracingConfig` proprietà di una `AWS::Lambda::Function` risorsa.

`VersionDescription`

Speciifica il `Description` campo che viene aggiunto alla nuova risorsa della versione Lambda.

➖Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una `AWS::Lambda::Version` risorsa.

`VersionDeletionPolicy`

Speciifica la politica di eliminazione per la risorsa della versione Lambda che viene creata `AutoPublishAlias` quando viene impostata. Questo controlla se la risorsa della versione viene mantenuta o eliminata quando lo stack viene eliminato.

Valori validi: `Delete`, `Retain` o `Snapshot`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente. Imposta l'`DeletionPolicy`attributo sulla `AWS::Lambda::Version` risorsa generata.

`VpcConfig`

La configurazione che consente a questa funzione di accedere alle risorse private all'interno del tuo cloud privato virtuale (VPC).

Tipo: VpcConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `VpcConfig` proprietà di una `AWS::Lambda::Function` risorsa.

Valori restituiti

Ref

Quando l'ID logico di questa risorsa viene fornito alla funzione `Ref` intrinseca, restituisce il nome della risorsa della funzione Lambda sottostante.

Per ulteriori informazioni sull'uso della *`Ref`* funzione, consulta *`Ref`*la Guida per l'AWS CloudFormation utente.

Fn: GetAtt

`Fn::GetAtt` restituisce un valore per un attributo specificato di questo tipo. Di seguito sono riportati gli attributi disponibili e i valori restituiti di esempio.

Per ulteriori informazioni sull'utilizzo`Fn::GetAtt`, consulta `Fn::GetAtt`la Guida per l'AWS CloudFormation utente.

`Arn`

L'ARN della funzione Lambda sottostante.

Esempi

Funzione semplice

Di seguito è riportato un esempio di base di una AWS::Serverless::Function risorsa di tipo di pacchetto `Zip` (impostazione predefinita) e codice funzione in un bucket Amazon S3.

YAML

```
Type: AWS::Serverless::Function
Properties:
  Handler: index.handler
  Runtime: python3.9
  CodeUri: s3://bucket-name/key-name
```

Esempio di proprietà delle funzioni

Di seguito è riportato un esempio di un tipo AWS::Serverless::Function di pacchetto `Zip` (impostazione predefinita) che utilizza `InlineCodeLayers`,`Tracing`,`Policies`,`Amazon EFS`, e un'origine di `Api` eventi.

YAML

```
Type: AWS::Serverless::Function
DependsOn: MyMountTarget        # This is needed if an AWS::EFS::MountTarget resource
 is declared for EFS
Properties:
  Handler: index.handler
  Runtime: python3.9
  InlineCode: |
    def handler(event, context):
      print("Hello, world!")
  ReservedConcurrentExecutions: 30
  Layers:
    - Ref: MyLayer
  Tracing: Active
  Timeout: 120
  FileSystemConfigs:
    - Arn: !Ref MyEfsFileSystem
      LocalMountPath: /mnt/EFS
  Policies:
    - AWSLambdaExecute
    - Version: '2012-10-17       '
```

```
      Statement:
        - Effect: Allow
          Action:
            - s3:GetObject
            - s3:GetObjectACL
          Resource: 'arn:aws:s3:::sam-s3-demo-bucket/*'
  Events:
    ApiEvent:
      Type: Api
      Properties:
        Path: /path
        Method: get
```

ImageConfig esempio

Di seguito è riportato un esempio di una funzione `ImageConfig` for a Lambda di tipo pacchetto. `Image`

YAML

```
HelloWorldFunction:
  Type: AWS::Serverless::Function
  Properties:
    PackageType: Image
    ImageUri: account-id.dkr.ecr.region.amazonaws.com/ecr-repo-name:image-name
    ImageConfig:
      Command:
        - "app.lambda_handler"
      EntryPoint:
        - "entrypoint1"
      WorkingDirectory: "workDir"
```

RuntimeManagementConfig esempi

Una funzione Lambda configurata per aggiornare il suo ambiente di runtime in base al comportamento corrente:

```
TestFunction
  Type: AWS::Serverless::Function
  Properties:
    ...
    Runtime: python3.9
    RuntimeManagementConfig:
```

```
      UpdateRuntimeOn: Auto
```

Una funzione Lambda configurata per aggiornare il suo ambiente di runtime quando la funzione viene aggiornata:

```
TestFunction
  Type: AWS::Serverless::Function
  Properties:
    ...
    Runtime: python3.9
    RuntimeManagementConfig:
      UpdateRuntimeOn: FunctionUpdate
```

Una funzione Lambda configurata per aggiornare manualmente il suo ambiente di runtime:

```
TestFunction
  Type: AWS::Serverless::Function
  Properties:
    ...
    Runtime: python3.9
    RuntimeManagementConfig:
      RuntimeVersionArn: arn:aws:lambda:us-
east-1::runtime:4c459dd0104ee29ec65dcad056c0b3ddbe20d6db76b265ade7eda9a066859b1e
      UpdateRuntimeOn: Manual
```

SnapStart esempi

Esempio di una funzione Lambda SnapStart attivata per le versioni future:

```
TestFunc
  Type: AWS::Serverless::Function
  Properties:
    ...
    SnapStart:
      ApplyOn: PublishedVersions
```

TenancyConfig esempi

Esempio di una funzione Lambda con la modalità di isolamento del tenant attivata:

```
TestFunction
  Type: AWS::Serverless::Function
```

```
  Properties:
    ...
    TenancyConfig:
      TenantIsolationMode: PER_TENANT
```

DeadLetterQueue

Specifica una coda SQS o un argomento SNS a cui (AWS Lambda Lambda) invia eventi quando non è in grado di elaborarli. Per ulteriori informazioni sulla funzionalità Dead Letter Queue, consulta Dead-letter queues nella Developer Guide.AWS Lambda

SAM aggiungerà automaticamente l'autorizzazione appropriata al ruolo di esecuzione della funzione Lambda per consentire al servizio Lambda di accedere alla risorsa. sqs: SendMessage verrà aggiunto per le code SQS e SNS:Publish per gli argomenti SNS.

Sintassi

Per dichiarare questa entità nel tuo modello (), usa la seguente sintassi. AWS Serverless Application Model AWS SAM

YAML

```
  TargetArn: String
  Type: String
```

Properties

`TargetArn`

L'Amazon Resource Name (ARN) di una coda di Amazon SQS o Amazon SNS.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `TargetArn` proprietà del tipo di `AWS::Lambda::Function` `DeadLetterConfig` dati.

`Type`

Il tipo di coda di lettere morte.

Valori validi: `SNS`, `SQS`

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

DeadLetterQueue

Esempio di Dead Letter Queue per un argomento SNS.

YAML

```
DeadLetterQueue:
  Type: SNS
  TargetArn: arn:aws:sns:us-east-2:123456789012:my-topic
```

DeploymentPreference

Specifica le configurazioni per abilitare le distribuzioni Lambda graduali. Per ulteriori informazioni sulla configurazione di distribuzioni Lambda graduali, consulta. Implementazione graduale di applicazioni serverless con AWS SAM

> **Note**
> È necessario specificare un `AutoPublishAlias` nel proprio AWS::Serverless::Function per utilizzare un `DeploymentPreference` oggetto, altrimenti si verificherà un errore.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  Alarms: List
  Enabled: Boolean
  Hooks: Hooks
```

```
  PassthroughCondition: Boolean
  Role: String
  TriggerConfigurations: List
  Type: String
```

Properties

`Alarms`

Un elenco di CloudWatch allarmi che desideri vengano attivati da eventuali errori generati dalla distribuzione.

Questa proprietà accetta la funzione `Fn::If` intrinseca. Vedi la sezione Esempi alla fine di questo argomento per un modello di esempio che utilizza. `Fn::If`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Enabled`

Se questa preferenza di distribuzione è abilitata.

Tipo: Booleano

Obbligatorio: no

Impostazione predefinita: True

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Hooks`

Funzioni Lambda di convalida eseguite prima e dopo lo spostamento del traffico.

Tipo: ganci

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`PassthroughCondition`

Se True e se questa preferenza di distribuzione è abilitata, la Condition della funzione verrà trasmessa alla CodeDeploy risorsa generata. In genere, è necessario impostarlo su True. Altrimenti, la CodeDeploy risorsa verrebbe creata anche se la condizione della funzione viene risolta su False.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Role`

Un ARN di ruolo IAM che CodeDeploy verrà utilizzato per lo spostamento del traffico. Se viene fornito, non verrà creato un ruolo IAM.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`TriggerConfigurations`

Un elenco di configurazioni di trigger che si desidera associare al gruppo di distribuzione. Utilizzato per notificare un argomento SNS sugli eventi del ciclo di vita.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `TriggerConfigurations` proprietà di una risorsa. `AWS::CodeDeploy::DeploymentGroup`

`Type`

Al momento esistono due categorie di tipi di distribuzione: Linear e Canary. Per ulteriori informazioni sui tipi di distribuzione disponibili, vedereImplementazione graduale di applicazioni serverless con AWS SAM.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

DeploymentPreference con agganci pre e post-traffico.

Esempio di preferenza di implementazione che contiene gli hook precedenti e successivi al traffico.

YAML

```
DeploymentPreference:
  Enabled: true
  Type: Canary10Percent10Minutes
  Alarms:
    - !Ref: AliasErrorMetricGreaterThanZeroAlarm
    - !Ref: LatestVersionErrorMetricGreaterThanZeroAlarm
  Hooks:
    PreTraffic:
      !Ref: PreTrafficLambdaFunction
    PostTraffic:
      !Ref: PostTrafficLambdaFunction
```

DeploymentPreference con la funzione intrinseca Fn: :If

Esempio di preferenza di distribuzione utilizzata `Fn::If` per la configurazione degli allarmi. In questo esempio, `Alarm1` verrà configurato se lo `MyCondition` è `true` `Alarm2` e `Alarm5` verrà configurato se `MyCondition` lo è. `false`

YAML

```
DeploymentPreference:
  Enabled: true
  Type: Canary10Percent10Minutes
  Alarms:
    Fn::If:
      - MyCondition
      - - Alarm1
      - - Alarm2
        - Alarm5
```

Hooks

Funzioni Lambda di convalida eseguite prima e dopo lo spostamento del traffico.

> **Note**
>
> Le funzioni Lambda a cui si fa riferimento in questa proprietà configurano l'`CodeDeployLambdaAliasUpdate`oggetto della risorsa risultante. AWS::Lambda::Alias Per ulteriori informazioni, consulta CodeDeployLambdaAliasUpdate Policy nella Guida per l'AWS CloudFormation utente.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
PostTraffic: String
PreTraffic: String
```

Properties

`PostTraffic`

Funzione Lambda che viene eseguita dopo lo spostamento del traffico.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`PreTraffic`

Funzione Lambda che viene eseguita prima dello spostamento del traffico.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

Hook

Esempi di funzioni hook

YAML

```
Hooks:
  PreTraffic:
    Ref: PreTrafficLambdaFunction
  PostTraffic:
    Ref: PostTrafficLambdaFunction
```

DurableConfig

Configura impostazioni di esecuzione durevoli per le AWS Lambda funzioni. Le funzioni durevoli possono funzionare fino a un anno e controllano automaticamente i progressi, consentendo flussi di lavoro a lunga durata e applicazioni con tolleranza ai guasti. Per ulteriori informazioni sulle funzioni durevoli, consulta Funzioni durevoli Lambda nella Guida per gli AWS Lambda sviluppatori.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  ExecutionTimeout: Integer
  RetentionPeriodInDays: Integer
```

Properties

`ExecutionTimeout`

La quantità di tempo (in secondi) in cui Lambda consente a una funzione durevole di funzionare prima di interromperla. Il massimo è un anno di 366 giorni o 31.622.400 secondi.

Tipo: integer

Obbligatorio: sì

Minimo: 1

Massimo: 3162.400

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ExecutionTimeout` proprietà del tipo di `AWS::Lambda::Function` `DurableConfig` dati.

`RetentionPeriodInDays`

Il numero di giorni dopo la chiusura di un'esecuzione durevole in cui Lambda conserva la cronologia, da uno a 90 giorni. L'impostazione di default è 14 giorni.

Tipo: integer

Obbligatorio: no

Impostazione predefinita: 14

Minimo: 1

Massimo: 90

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RetentionPeriodInDays` proprietà del tipo di `AWS::Lambda::Function` `DurableConfig` dati.

Esempi

DurableConfig

Esempio di configurazione durevole per una funzione con un timeout di esecuzione di 1 ora e un periodo di conservazione di 7 giorni.

YAML

```
DurableConfig:
  ExecutionTimeout: 3600
  RetentionPeriodInDays: 7
```

EventInvokeConfiguration

Opzioni di configurazione per le chiamate asincrone di Lambda Alias o Version.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model ()AWS SAM, usa la seguente sintassi.

YAML

```
DestinationConfig: EventInvokeDestinationConfiguration
MaximumEventAgeInSeconds: Integer
MaximumRetryAttempts: Integer
```

Properties

`DestinationConfig`

Oggetto di configurazione che specifica la destinazione di un evento dopo che Lambda lo elabora.

Tipo: EventInvokeDestinationConfiguration

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è simile alla `DestinationConfig` proprietà di una `AWS::Lambda::EventInvokeConfig` risorsa. SAM richiede un parametro aggiuntivo, «Type», che non esiste in CloudFormation.

`MaximumEventAgeInSeconds`

L'età massima di una richiesta inviata da Lambda a una funzione per l'elaborazione.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumEventAgeInSeconds` proprietà di una `AWS::Lambda::EventInvokeConfig` risorsa.

`MaximumRetryAttempts`

Il numero massimo di tentativi da ripetere prima che la funzione restituisca un errore.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumRetryAttempts` proprietà di una `AWS::Lambda::EventInvokeConfig` risorsa.

Esempi

MaximumEventAgeInSeconds

MaximumEventAgeInSeconds esempio

YAML

```
EventInvokeConfig:
  MaximumEventAgeInSeconds: 60
  MaximumRetryAttempts: 2
  DestinationConfig:
    OnSuccess:
      Type: SQS
      Destination: arn:aws:sqs:us-west-2:012345678901:my-queue
    OnFailure:
      Type: Lambda
      Destination: !GetAtt DestinationLambda.Arn
```

EventInvokeDestinationConfiguration

Oggetto di configurazione che specifica la destinazione di un evento dopo che Lambda lo elabora.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  OnFailure: OnFailure
```

```
  OnSuccess: OnSuccess
```

Properties

`OnFailure`

Destinazione per gli eventi la cui elaborazione non è riuscita.

Tipo: OnFailure

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è simile alla `OnFailure` proprietà di una `AWS::Lambda::EventInvokeConfig` risorsa. Richiede `Type` una proprietà aggiuntiva solo per SAM.

`OnSuccess`

Destinazione per gli eventi che sono stati elaborati correttamente.

Tipo: OnSuccess

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è simile alla `OnSuccess` proprietà di una `AWS::Lambda::EventInvokeConfig` risorsa. Richiede `Type` una proprietà aggiuntiva solo per SAM.

Esempi

OnSuccess

OnSuccess esempio

YAML

```
EventInvokeConfig:
  DestinationConfig:
    OnSuccess:
      Type: SQS
      Destination: arn:aws:sqs:us-west-2:012345678901:my-queue
    OnFailure:
      Type: Lambda
```

```
      Destination: !GetAtt DestinationLambda.Arn
```

OnFailure

Destinazione per gli eventi la cui elaborazione non è riuscita.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Destination: String
Type: String
```

Properties

`Destination`

L'Amazon Resource Name (ARN) della risorsa di destinazione.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: Questa proprietà è simile alla `OnFailure` proprietà di una `AWS::Lambda::EventInvokeConfig` risorsa. SAM aggiungerà tutte le autorizzazioni necessarie al ruolo IAM generato automaticamente associato a questa funzione per accedere alla risorsa a cui si fa riferimento in questa proprietà.

Note aggiuntive: se il tipo è Lambda/EventBridge, è richiesta la destinazione.

`Type`

Tipo di risorsa a cui si fa riferimento nella destinazione. I tipi supportati sono`SQS`,`SNS`, `S3Lambda`, e`EventBridge`.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Note aggiuntive: se il tipo è SQS/SNS e la `Destination` proprietà viene lasciata vuota, la SQS/SNS risorsa viene generata automaticamente da SAM. Per fare riferimento alla risorsa, usa *`<function-logical-id>`*`.DestinationQueue` per SQS o *`<function-logical-id>`*`.DestinationTopic` per SNS. Se il tipo è Lambda/EventBridge, `Destination` è obbligatorio.

Esempi

EventInvoke Esempio di configurazione con destinazioni SQS e Lambda

In questo esempio non viene fornita alcuna destinazione per la OnSuccess configurazione SQS, quindi SAM crea implicitamente una coda SQS e aggiunge le autorizzazioni necessarie. Inoltre, in questo esempio, nella OnFailure configurazione viene specificata una destinazione per una risorsa Lambda dichiarata nel file modello, quindi SAM aggiunge le autorizzazioni necessarie a questa funzione Lambda per chiamare la funzione Lambda di destinazione.

YAML

```
EventInvokeConfig:
  DestinationConfig:
    OnSuccess:
      Type: SQS
    OnFailure:
      Type: Lambda
      Destination: !GetAtt DestinationLambda.Arn  # Arn of a Lambda function declared
 in the template file.
```

EventInvoke Esempio di configurazione con destinazione SNS

In questo esempio viene fornita una destinazione per un argomento SNS dichiarato nel file modello per la OnSuccess configurazione.

YAML

```
EventInvokeConfig:
  DestinationConfig:
    OnSuccess:
      Type: SNS
      Destination:
        Ref: DestinationSNS       # Arn of an SNS topic declared in the tempate file
```

OnSuccess

Destinazione per gli eventi che sono stati elaborati correttamente.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Destination: String
Type: String
```

Properties

`Destination`

L'Amazon Resource Name (ARN) della risorsa di destinazione.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: Questa proprietà è simile alla `OnSuccess` proprietà di una `AWS::Lambda::EventInvokeConfig` risorsa. SAM aggiungerà tutte le autorizzazioni necessarie al ruolo IAM generato automaticamente associato a questa funzione per accedere alla risorsa a cui si fa riferimento in questa proprietà.

Note aggiuntive: se il tipo è Lambda/EventBridge, è richiesta la destinazione.

`Type`

Tipo di risorsa a cui si fa riferimento nella destinazione. I tipi supportati sono`SQS`,`SNS`, `S3Lambda`, `eEventBridge`.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Note aggiuntive: se il tipo è SQS/SNS e la `Destination` proprietà viene lasciata vuota, la SQS/SNS risorsa viene generata automaticamente da SAM. Per fare riferimento alla risorsa, usa *`<function-logical-id>`*`.DestinationQueue` per SQS o *`<function-logical-id>`*`.DestinationTopic` per SNS. Se il tipo è Lambda/EventBridge, `Destination` è obbligatorio.

Esempi

EventInvoke Esempio di configurazione con destinazioni SQS e Lambda

In questo esempio non viene fornita alcuna destinazione per la OnSuccess configurazione SQS, quindi SAM crea implicitamente una coda SQS e aggiunge le autorizzazioni necessarie. Inoltre, in questo esempio, nella OnFailure configurazione viene specificata una destinazione per una risorsa Lambda dichiarata nel file modello, quindi SAM aggiunge le autorizzazioni necessarie a questa funzione Lambda per chiamare la funzione Lambda di destinazione.

YAML

```
EventInvokeConfig:
  DestinationConfig:
    OnSuccess:
      Type: SQS
    OnFailure:
      Type: Lambda
      Destination: !GetAtt DestinationLambda.Arn  # Arn of a Lambda function declared
 in the template file.
```

EventInvoke Esempio di configurazione con destinazione SNS

In questo esempio viene fornita una destinazione per un argomento SNS dichiarato nel file modello per la OnSuccess configurazione.

YAML

```
EventInvokeConfig:
  DestinationConfig:
    OnSuccess:
      Type: SNS
      Destination:
        Ref: DestinationSNS       # Arn of an SNS topic declared in the tempate file
```

EventSource

L'oggetto che descrive l'origine degli eventi che attivano la funzione. Ogni evento è costituito da un tipo e da un insieme di proprietà che dipendono da quel tipo. Per ulteriori informazioni sulle proprietà di ciascuna fonte di eventi, consultate l'argomento corrispondente a quel tipo.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Properties: AlexaSkill | Api | CloudWatchEvent | CloudWatchLogs | Cognito
| DocumentDB | DynamoDB | EventBridgeRule | HttpApi | IoTRule | Kinesis | MQ | MSK
| S3 | Schedule | ScheduleV2 | SelfManagedKafka | SNS | SQS
Type: String
```

Properties

`Properties`

Oggetto che descrive le proprietà di questa mappatura degli eventi. L'insieme di proprietà deve essere conforme al tipo definito.

Tipo : AlexaSkill| Api | | CloudWatchEvent| Cognito CloudWatchLogs| DocumentDB | DynamoDB | | Io | Kinesis | MQ EventBridgeRule??? | MSK HttpApi??? | S3 TRule| Pianificazione| ScheduleV2 | | SNS | SQS SelfManagedKafka

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà è unica e non ha un equivalente. AWS SAM CloudFormation

`Type`

Il tipo di evento,

Valori validi: `AlexaSkill` `ApiCloudWatchEvent`,`CloudWatchLogs`,`Cognito`,`DocumentDB`,`DynamoDB`,`EventBridgeRule`,`Htt` `MQMSK`,`S3`,`Schedule`,`ScheduleV2`,`SelfManagedKafka`,`SNS`, `SQS`

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

APIEvent

Esempio di utilizzo di un evento API

YAML

```
ApiEvent:
  Type: Api
  Properties:
    Method: get
    Path: /group/{user}
    RestApiId:
      Ref: MyApi
```

AlexaSkill

L'oggetto che descrive un tipo di origine `AlexaSkill` dell'evento.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  SkillId: String
```

Properties

`SkillId`

L'Alexa Skill ID per la tua Alexa Skill. Per ulteriori informazioni su Skill ID, consulta Configurare il trigger per una funzione Lambda nella documentazione di Alexa Skills Kit.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

AlexaSkillTrigger

Esempio di evento Alexa Skill

YAML

```
AlexaSkillEvent:
  Type: AlexaSkill
```

Api

L'oggetto che descrive un tipo di origine `Api` dell'evento. Se viene definita una AWS::Serverless::Api risorsa, i valori del percorso e del metodo devono corrispondere a un'operazione nella definizione OpenAPI dell'API.

Se AWS::Serverless::Api è definito no, la funzione input e output sono una rappresentazione della richiesta HTTP e della risposta HTTP.

Ad esempio, utilizzando l' JavaScript API, il codice di stato e il corpo della risposta possono essere controllati restituendo un oggetto con le chiavi statusCode e body.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  Auth: ApiFunctionAuth
  Method: String
  Path: String
```

```
  RequestModel: RequestModel
  RequestParameters: List of [ String | RequestParameter ]
  RestApiId: String
  ResponseTransferMode: String
  TimeoutInMillis: Integer
```

Properties

`Auth`

Configurazione di autenticazione per questo metodo Api+Path+specifico.

Utile per sovrascrivere l'`DefaultAuthorizer`impostazione auth config dell'API su un singolo percorso quando non è specificato no `DefaultAuthorizer` o per sovrascrivere l'impostazione predefinita. `ApiKeyRequired`

Tipo: ApiFunctionAuth

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`Method`

Metodo HTTP per il quale viene richiamata questa funzione. Le opzioni includono `DELETEGET`,`HEAD`,`OPTIONS`,`PATCH`, `POSTPUT`, e`ANY`. Per ulteriori informazioni, consulta Configurare un metodo HTTP nella Guida per gli sviluppatori di API Gateway.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Path`

Percorso Uri per il quale viene richiamata questa funzione. Deve iniziare con`/`.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`RequestModel`

Richiedi il modello da utilizzare per questo specifico metodo Api+Path+. Questo dovrebbe fare riferimento al nome di un modello specificato nella `Models` sezione di una risorsa. AWS::Serverless::Api

Tipo: RequestModel

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`RequestParameters`

Richiedi la configurazione dei parametri per questo metodo Api+Path+ specifico. Tutti i nomi dei parametri devono iniziare con `method.request` e devono essere limitati a`method.request.header`, o. `method.request.querystring` `method.request.path`

Un elenco può contenere sia stringhe di nomi di parametri che RequestParameteroggetti. Per le stringhe, l'impostazione predefinita delle `Caching` proprietà `Required` and è. `false`

Tipo: Elenco di [String | RequestParameter]

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`RestApiId`

Identificatore di una RestApi risorsa, che deve contenere un'operazione con il percorso e il metodo indicati. In genere, è impostato per fare riferimento a una AWS::Serverless::Api risorsa definita in questo modello.

Se non definisci questa proprietà, AWS SAM crea una AWS::Serverless::Api risorsa predefinita utilizzando un `OpenApi` documento generato. Tale risorsa contiene un'unione di tutti i percorsi e i metodi definiti dagli `Api` eventi nello stesso modello che non specificano a`RestApiId`.

Questo non può fare riferimento a una AWS::Serverless::Api risorsa definita in un altro modello.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ResponseTransferMode`

La modalità di trasferimento della risposta per l'integrazione della funzione Lambda. Impostato per `RESPONSE_STREAM` abilitare lo streaming di risposte Lambda tramite API Gateway, permettendo alla funzione di trasmettere le risposte ai client. Se impostato su`RESPONSE_STREAM`, API Gateway utilizza l'API Lambda InvokeWithResponseStreaming .

- Tipo: stringa

Obbligatorio: no

Valori validi: `BUFFERED` | `RESPONSE_STREAM`

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ResponseTransferMode`proprietà di un`AWS::ApiGateway::Method Integration`.

`TimeoutInMillis`

Timeout personalizzato tra 50 e 29.000 millisecondi.

> **Note**
>
> Quando si specifica questa proprietà, AWS SAM modifica la definizione OpenAPI. La definizione OpenAPI deve essere specificata in linea utilizzando la proprietà. `DefinitionBody`

Tipo: integer

Obbligatorio: no

Impostazione predefinita: 29.000 millisecondi o 29 secondi

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

Esempi

Esempio di base

YAML

```
Events:
  ApiEvent:
    Type: Api
    Properties:
      Path: /path
      Method: get
      RequestParameters:
        - method.request.header.Authorization
        - method.request.querystring.keyword:
            Required: true
            Caching: false
```

ApiFunctionAuth

Configura l'autorizzazione a livello di evento, per un'API, un percorso e un metodo specifici.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  ApiKeyRequired: Boolean
  AuthorizationScopes: List
  Authorizer: String
  InvokeRole: String
  OverrideApiAuth: Boolean
  ResourcePolicy: ResourcePolicyStatement
```

Properties

`ApiKeyRequired`

Richiede una chiave API per questa API, percorso e metodo.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AuthorizationScopes`

Gli ambiti di autorizzazione da applicare a questa API, percorso e metodo.

Gli ambiti specificati sostituiranno tutti gli ambiti applicati dalla `DefaultAuthorizer` proprietà, se l'hai specificata.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`Authorizer`

Il `Authorizer` per una funzione specifica.

Se hai specificato un autorizzatore globale per la tua `AWS::Serverless::Api` risorsa, puoi sovrascrivere l'autorizzatore impostando su. `Authorizer` `NONE` Per vedere un esempio, consulta Sostituisci un autorizzatore globale per la tua API REST di Amazon API Gateway.

> **Note**
>
> Se utilizzi la `DefinitionBody` proprietà di una `AWS::Serverless::Api` risorsa per descrivere la tua API, devi usare `OverrideApiAuth` with `Authorizer` per sovrascrivere l'autorizzazione globale. Per ulteriori informazioni, consulta `OverrideApiAuth`.

Valori validi: `AWS_IAM``NONE`, o l'ID logico per qualsiasi autorizzatore definito nel modello. AWS SAM

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`InvokeRole`

Specifica l'uso `InvokeRole` per l'`AWS_IAM`autorizzazione.

- Tipo: stringa

Obbligatorio: no

Default: `CALLER_CREDENTIALS`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Note aggiuntive: `CALLER_CREDENTIALS` maps to`arn:aws:iam::`*`:<user>/`*, che utilizza le credenziali del chiamante per richiamare l'endpoint.

`OverrideApiAuth`

Specificate come `true` sovrascrivere la configurazione di autorizzazione globale della risorsa. `AWS::Serverless::Api` Questa proprietà è richiesta solo se si specifica un autorizzatore globale e si utilizza la `DefinitionBody` proprietà di una `AWS::Serverless::Api` risorsa per descrivere l'API.

> **Note**
>
> Quando specifichi `OverrideApiAuth` as`true`, AWS SAM sostituirà l'autorizzazione globale con tutti i valori forniti per`ApiKeyRequired`, `Authorizer` o. `ResourcePolicy` Pertanto, è necessario specificare almeno una di queste proprietà quando si utilizza. `OverrideApiAuth` Per vedere un esempio, consulta Sostituisci un autorizzatore globale quando viene specificato for DefinitionBody AWS::Serverless::Api.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ResourcePolicy`

Configura la politica delle risorse per questo percorso su un'API.

Tipo: ResourcePolicyStatement

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

Autenticazione della funzione

L'esempio seguente specifica l'autorizzazione a livello di funzione.

YAML

```
Auth:
  ApiKeyRequired: true
  Authorizer: NONE
```

Sostituisci un autorizzatore globale per la tua API REST di Amazon API Gateway

Puoi specificare un autorizzatore globale per la tua risorsa. `AWS::Serverless::Api` Di seguito è riportato un esempio che configura un autorizzatore globale predefinito:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyApiWithLambdaRequestAuth:
    Type: AWS::Serverless::Api
    Properties:
      ...
      Auth:
        Authorizers:
          MyLambdaRequestAuth:
            FunctionArn: !GetAtt MyAuthFn.Arn
        DefaultAuthorizer: MyLambdaRequestAuth
```

Per sovrascrivere l'autorizzatore predefinito per la tua AWS Lambda funzione, puoi specificare as. `Authorizer` `NONE` Di seguito è riportato un esempio:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  ...
  MyFn:
    Type: AWS::Serverless::Function
    Properties:
      ...
      Events:
        LambdaRequest:
          Type: Api
          Properties:
            RestApiId: !Ref MyApiWithLambdaRequestAuth
            Method: GET
            Auth:
              Authorizer: NONE
```

Sostituisci un autorizzatore globale quando viene specificato for DefinitionBody AWS::Serverless::Api

Quando si utilizza la `DefinitionBody` proprietà per descrivere la `AWS::Serverless::Api` risorsa, il metodo override precedente non funziona. Di seguito è riportato un esempio di utilizzo della `DefinitionBody` proprietà per una `AWS::Serverless::Api` risorsa:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyApiWithLambdaRequestAuth:
    Type: AWS::Serverless::Api
    Properties:
      ...
      DefinitionBody:
        swagger: 2.0
        ...
        paths:
          /lambda-request:
            ...
      Auth:
        Authorizers:
```

```
          MyLambdaRequestAuth:
            FunctionArn: !GetAtt MyAuthFn.Arn
        DefaultAuthorizer: MyLambdaRequestAuth
```

Per sovrascrivere l'autorizzatore globale, utilizzate la `OverrideApiAuth` proprietà. Di seguito è riportato un esempio che utilizza `OverrideApiAuth` per sostituire l'autorizzatore globale con il valore fornito per: `Authorizer`

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyApiWithLambdaRequestAuth:
    Type: AWS::Serverless::Api
    Properties:
      ...
      DefinitionBody:
        swagger: 2-0
        ...
        paths:
          /lambda-request:
            ...
      Auth:
        Authorizers:
          MyLambdaRequestAuth:
            FunctionArn: !GetAtt MyAuthFn.Arn
        DefaultAuthorizer: MyLambdaRequestAuth

    MyAuthFn:
      Type: AWS::Serverless::Function
      ...

    MyFn:
      Type: AWS::Serverless::Function
        Properties:
          ...
          Events:
            LambdaRequest:
              Type: Api
              Properties:
                RestApiId: !Ref MyApiWithLambdaRequestAuth
                Method: GET
                Auth:
```

```
                    Authorizer: NONE
                    OverrideApiAuth: true
                  Path: /lambda-token
```

ResourcePolicyStatement

Configura una politica delle risorse per tutti i metodi e i percorsi di un'API. Per ulteriori informazioni sulle politiche delle risorse, consulta Controllare l'accesso a un'API con le politiche delle risorse di API Gateway nella Guida per sviluppatori di API Gateway.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
AwsAccountBlacklist: List
AwsAccountWhitelist: List
CustomStatements: List
IntrinsicVpcBlacklist: List
IntrinsicVpcWhitelist: List
IntrinsicVpceBlacklist: List
IntrinsicVpceWhitelist: List
IpRangeBlacklist: List
IpRangeWhitelist: List
SourceVpcBlacklist: List
SourceVpcWhitelist: List
```

Properties

`AwsAccountBlacklist`

Gli AWS account da bloccare.

Tipo: elenco di String

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AwsAccountWhitelist`

Gli AWS account da consentire. Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi in fondo a questa pagina.

Tipo: elenco di String

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`CustomStatements`

Un elenco di istruzioni personalizzate sulla politica delle risorse da applicare a questa API. Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi in fondo a questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IntrinsicVpcBlacklist`

L'elenco dei cloud privati virtuali (VPCs) da bloccare, in cui ogni VPC è specificato come riferimento, ad esempio un riferimento dinamico o una funzione `Ref` intrinseca. Per un esempio di utilizzo di questa proprietà, consultate la sezione Esempi in fondo a questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IntrinsicVpcWhitelist`

L'elenco di VPCs to allow, in cui ogni VPC è specificato come riferimento, ad esempio un riferimento dinamico o una funzione `Ref` intrinseca.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`IntrinsicVpceBlacklist`

L'elenco degli endpoint VPC da bloccare, in cui ogni endpoint VPC è specificato come riferimento, ad esempio un riferimento dinamico o la funzione intrinseca. `Ref`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`IntrinsicVpceWhitelist`

L'elenco degli endpoint VPC da consentire, in cui ogni endpoint VPC è specificato come riferimento, ad esempio un riferimento dinamico o la funzione intrinseca. `Ref` Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi in fondo a questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IpRangeBlacklist`

Gli indirizzi IP o gli intervalli di indirizzi da bloccare. Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi nella parte inferiore di questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IpRangeWhitelist`

Gli indirizzi IP o gli intervalli di indirizzi da consentire.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`SourceVpcBlacklist`

Gli endpoint VPC o VPC di origine da bloccare. I nomi dei VPC di origine devono iniziare con `"vpc-"` e i nomi degli endpoint VPC di origine devono iniziare con. `"vpce-"` Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi nella parte inferiore di questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`SourceVpcWhitelist`

Gli endpoint VPC o VPC di origine da consentire. I nomi dei VPC di origine devono iniziare con `"vpc-"` e i nomi degli endpoint VPC di origine devono iniziare con. `"vpce-"`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

Esempi

Esempio di politica delle risorse

L'esempio seguente blocca due indirizzi IP e un VPC di origine e consente un AWS account.

YAML

```
Auth:
  ResourcePolicy:
    CustomStatements: [{
                        "Effect": "Allow",
                        "Principal": "*",
```

```
                        "Action": "execute-api:Invoke",
                        "Resource": "execute-api:/Prod/GET/pets",
                        "Condition": {
                          "IpAddress": {
                            "aws:SourceIp": "1.2.3.4"
                          }
                        }
                      }]
    IpRangeBlacklist:
      - "10.20.30.40"
      - "1.2.3.4"
    SourceVpcBlacklist:
      - "vpce-1a2b3c4d"
    AwsAccountWhitelist:
      - "111122223333"
    IntrinsicVpcBlacklist:
      - "{{resolve:ssm:SomeVPCReference:1}}"
      - !Ref MyVPC
    IntrinsicVpceWhitelist:
      - "{{resolve:ssm:SomeVPCEReference:1}}"
      - !Ref MyVPCE
```

RequestModel

Configura un modello di richiesta per uno specifico metodo Api+Path+.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Model: String
Required: Boolean
ValidateBody: Boolean
ValidateParameters: Boolean
```

Properties

`Model`

Nome di un modello definito nella proprietà Models di. AWS::Serverless::Api

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Required`

Aggiunge una `required` proprietà nella sezione dei parametri della OpenApi definizione per l'endpoint API specificato.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ValidateBody`

Specifica se API Gateway utilizza `Model` per convalidare il corpo della richiesta. Per ulteriori informazioni, consulta Abilitare la convalida delle richieste in API Gateway nella Guida per sviluppatori di API Gateway.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ValidateParameters`

Specifica se API Gateway utilizza `Model` per convalidare i parametri del percorso della richiesta, le stringhe di query e le intestazioni. Per ulteriori informazioni, consulta Abilitare la convalida delle richieste in API Gateway nella Guida per sviluppatori di API Gateway.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

Richiedi modello

Esempio di modello di richiesta

YAML

```
RequestModel:
  Model: User
  Required: true
  ValidateBody: true
  ValidateParameters: true
```

RequestParameter

Configura il parametro di richiesta per un metodo Api+Path+ specifico.

È necessario `Required` specificare una delle due `Caching` proprietà per il parametro di richiesta

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  Caching: Boolean
  Required: Boolean
```

Properties

`Caching`

Aggiunge una `cacheKeyParameters` sezione alla OpenApi definizione di API Gateway

Tipo: Booleano

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Required`

Questo campo specifica se un parametro è obbligatorio

Tipo: Booleano

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

Parametro della richiesta

Esempio di impostazione dei parametri di richiesta

YAML

```
RequestParameters:
  - method.request.header.Authorization:
      Required: true
      Caching: true
```

CloudWatchEvent

L'oggetto che descrive un tipo di origine `CloudWatchEvent` dell'evento.

AWS Serverless Application Model (AWS SAM) genera una AWS::Events::Rulerisorsa quando viene impostato questo tipo di evento.

Nota importante: EventBridgeRule è il tipo di sorgente di eventi preferito da utilizzare, anzichéCloudWatchEvent. EventBridgeRulee CloudWatchEvent utilizza lo stesso servizio, API e CloudFormation risorse sottostanti. Tuttavia, AWS SAM aggiungerà il supporto per le nuove funzionalità solo aEventBridgeRule.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
Enabled: Boolean
EventBusName: String
Input: String
InputPath: String
Pattern: EventPattern
State: String
```

Properties

`Enabled`

Indica se la regola è abilitata.

Per disabilitare la regola, imposta questa proprietà su. `false`

> **Note**
>
> Specificate la `State` proprietà `Enabled` o, ma non entrambe.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `State` proprietà di una `AWS::Events::Rule` risorsa. Se questa proprietà è impostata su `true` allora AWS SAM passa`ENABLED`, altrimenti passa`DISABLED`.

`EventBusName`

Il bus evento da associare a questa regola. Se si omette questa proprietà, AWS SAM utilizza il bus di eventi predefinito.

-Tipo: stringa

Obbligatorio: no

Predefinito: bus eventi predefinito

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventBusName` proprietà di una `AWS::Events::Rule` risorsa.

`Input`

Testo JSON valido passato al target. Se utilizzi questa proprietà, al target non viene passata alcuna informazione sul testo dell'evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Input` proprietà di una `AWS::Events::Rule Target` risorsa.

`InputPath`

Quando non vuoi passare l'intero evento corrispondente alla destinazione, usa la `InputPath` proprietà per descrivere quale parte dell'evento trasmettere.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `InputPath` proprietà di una `AWS::Events::Rule Target` risorsa.

`Pattern`

Descrive quali eventi vengono instradati nella destinazione specificata. Per ulteriori informazioni, consulta Events and Event Patterns EventBridge nella Amazon EventBridge User Guide.

Tipo: EventPattern

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventPattern` proprietà di una `AWS::Events::Rule` risorsa.

`State`

Lo stato della regola.

Valori accettati: `DISABLED | ENABLED`

> **Note**
>
> Specificate la `State` proprietà `Enabled` o, ma non entrambe.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `State` proprietà di una `AWS::Events::Rule` risorsa.

Esempi

CloudWatchEvent

Di seguito è riportato un esempio di tipo di sorgente di `CloudWatchEvent` eventi.

YAML

```
CWEvent:
  Type: CloudWatchEvent
  Properties:
    Enabled: false
    Input: '{"Key": "Value"}'
    Pattern:
      detail:
        state:
          - running
```

CloudWatchLogs

L'oggetto che descrive un tipo di origine `CloudWatchLogs` dell'evento.

Questo evento genera una AWS::Logs::SubscriptionFilterrisorsa, specifica un filtro di sottoscrizione e lo associa al gruppo di log specificato.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  FilterPattern: String
  LogGroupName: String
```

Properties

`FilterPattern`

Le espressioni di filtraggio che limitano ciò che viene consegnato alla risorsa di destinazione. AWS Per ulteriori informazioni sulla sintassi dei modelli di filtro, consulta Sintassi di filtri e modelli.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FilterPattern` proprietà di una `AWS::Logs::SubscriptionFilter` risorsa.

`LogGroupName`

Il gruppo di log a cui associare il filtro sottoscrizioni. Tutti gli eventi di registro caricati in questo gruppo di log vengono filtrati e consegnati alla AWS risorsa specificata se il modello di filtro corrisponde agli eventi di registro.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `LogGroupName` proprietà di una `AWS::Logs::SubscriptionFilter` risorsa.

Esempi

Filtro di abbonamento Cloudwatchlogs

Esempio di filtro di abbonamento Cloudwatchlogs

YAML

```
CWLog:
  Type: CloudWatchLogs
  Properties:
    LogGroupName:
      Ref: CloudWatchLambdaLogsGroup
    FilterPattern: My pattern
```

Cognito

L'oggetto che descrive un tipo di origine `Cognito` dell'evento.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Trigger: List
UserPool: String
```

Properties

`Trigger`

Le informazioni di configurazione del trigger Lambda per il nuovo pool di utenti.

Tipo: elenco

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `LambdaConfig` proprietà di una `AWS::Cognito::UserPool` risorsa.

`UserPool`

Riferimento a UserPool definito nello stesso modello

▬Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

Evento Cognito

Esempio di evento Cognito

YAML

```
CognitoUserPoolPreSignup:
  Type: Cognito
  Properties:
    UserPool:
      Ref: MyCognitoUserPool
    Trigger: PreSignUp
```

DocumentDB

L'oggetto che descrive un tipo di origine `DocumentDB` dell'evento. Per ulteriori informazioni, consulta Using AWS Lambda with Amazon DocumentDB nella AWS Lambda Developer Guide.

Sintassi

Per dichiarare questa entità nel tuo AWS SAM modello, usa la seguente sintassi.

YAML

```
BatchSize: Integer
Cluster: String
CollectionName: String
DatabaseName: String
Enabled: Boolean
FilterCriteria: FilterCriteria
FullDocument: String
KmsKeyArn: String
MaximumBatchingWindowInSeconds: Integer
SecretsManagerKmsKeyId: String
SourceAccessConfigurations: List
StartingPosition: String
StartingPositionTimestamp: Double
```

Properties

`BatchSize`

Il numero massimo di elementi da recuperare in un singolo batch.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `BatchSize` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`Cluster`

L'Amazon Resource Name (ARN) del cluster Amazon DocumentDB.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventSourceArn` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`CollectionName`

Il nome della raccolta da utilizzare all'interno del database. Se non specifichi una raccolta, Lambda utilizza tutte le raccolte.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CollectionName` proprietà di un tipo di `AWS::Lambda::EventSourceMapping` `DocumentDBEventSourceConfig` dati.

`DatabaseName`

Il nome del database da utilizzare all'interno del cluster Amazon DocumentDB.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `DatabaseName` proprietà di un tipo di `AWS::Lambda::EventSourceMapping` `DocumentDBEventSourceConfig` dati.

`Enabled`

Se`true`, la mappatura della fonte dell'evento è attiva. Per mettere in pausa il polling e la chiamata, imposta su. `false`

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Enabled` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`FilterCriteria`

Un oggetto che definisce i criteri che determinano se Lambda deve elaborare un evento. Per ulteriori informazioni, consulta Filtraggio degli eventi Lambda nella Developer Guide.AWS Lambda

Tipo: FilterCriteria

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FilterCriteria` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`FullDocument`

Determina cosa invia Amazon DocumentDB al tuo flusso di eventi durante le operazioni di aggiornamento dei documenti. Se impostato su`UpdateLookup`, Amazon DocumentDB invia un delta che descrive le modifiche, insieme a una copia dell'intero documento. Altrimenti, Amazon DocumentDB invia solo un documento parziale contenente le modifiche.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FullDocument` proprietà di un tipo di `AWS::Lambda::EventSourceMapping` `DocumentDBEventSourceConfig` dati.

`KmsKeyArn`

L'Amazon Resource Name (ARN) della chiave per crittografare le informazioni relative a questo evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `KmsKeyArn` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumBatchingWindowInSeconds`

Il tempo massimo in secondi per la raccolta dei record prima di richiamare la funzione.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumBatchingWindowInSeconds` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`SecretsManagerKmsKeyId`

L'ID chiave AWS Key Management Service (AWS KMS) di una chiave gestita dal cliente di AWS Secrets Manager. Richiesto quando si utilizza una chiave gestita dal cliente di Secrets Manager con un ruolo di esecuzione Lambda che non include l'`kms:Decrypt`autorizzazione.

Il valore di questa proprietà è un UUID. Ad esempio: `1abc23d4-567f-8ab9-cde0-1fab234c5d67`.

Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`SourceAccessConfigurations`

Una matrice del protocollo di autenticazione o dell'host virtuale. Specificalo utilizzando il tipo di SourceAccessConfigurationsdati.

Per il tipo di origine dell'`DocumentDB`evento, l'unico tipo di configurazione valido è`BASIC_AUTH`.

- `BASIC_AUTH`— Il segreto di Secrets Manager che memorizza le credenziali del broker. Per questo tipo, la credenziale deve essere nel seguente formato:. `{"username": "your-username", "password": "your-password"}` È consentito un solo oggetto di tipo`BASIC_AUTH`.

Tipo: elenco

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla SourceAccessConfigurations proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`StartingPosition`

La posizione in un flusso da cui iniziare la lettura.

- `AT_TIMESTAMP`— Specificare l'ora a partire dalla quale iniziare a leggere i record.
- `LATEST`— Leggi solo i nuovi record.
- `TRIM_HORIZON`— Elabora tutti i record disponibili.

▬Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla StartingPosition proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`StartingPositionTimestamp`

L'ora da cui iniziare a leggere, in secondi di tempo Unix. Definisce `StartingPositionTimestamp` quando `StartingPosition` è specificato come`AT_TIMESTAMP`.

Tipo: numero a doppia precisione

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla StartingPositionTimestamp proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

Esempi

Fonte di eventi Amazon DocumentDB

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyFunction:
```

```
    Type: AWS::Serverless::Function
    Properties:
    ...
      Events:
        MyDDBEvent:
          Type: DocumentDB
          Properties:
            Cluster: "arn:aws:rds:us-west-2:123456789012:cluster:docdb-2023-01-01"
            BatchSize: 10
            MaximumBatchingWindowInSeconds: 5
            DatabaseName: "db1"
            CollectionName: "collection1"
            FullDocument: "UpdateLookup"
            SourceAccessConfigurations:
              - Type: BASIC_AUTH
                URI: "arn:aws:secretsmanager:us-west-2:123456789012:secret:doc-db"
```

DynamoDB

L'oggetto che descrive un tipo di origine `DynamoDB` dell'evento. Per ulteriori informazioni, consulta Using AWS Lambda with Amazon DynamoDB nella AWS Lambda Developer Guide.

AWS SAM genera una AWS::Lambda::EventSourceMappingrisorsa quando viene impostato questo tipo di evento.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  BatchSize: Integer
  BisectBatchOnFunctionError: Boolean
  DestinationConfig: DestinationConfig
  Enabled: Boolean
  FilterCriteria: FilterCriteria
  FunctionResponseTypes: List
  KmsKeyArn: String
  MaximumBatchingWindowInSeconds: Integer
  MaximumRecordAgeInSeconds: Integer
  MaximumRetryAttempts: Integer
  MetricsConfig: MetricsConfig
```

```
  ParallelizationFactor: Integer
  StartingPosition: String
  StartingPositionTimestamp: Double
  Stream: String
  TumblingWindowInSeconds: Integer
```

Properties

`BatchSize`

Il numero massimo di elementi da recuperare in un singolo batch.

Tipo: integer

Obbligatorio: no

Impostazione predefinita: 100

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `BatchSize` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

Minimum: 1

Maximum: `1000`

`BisectBatchOnFunctionError`

Se la funzione restituisce un errore, dividi il batch in due e riprova.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `BisectBatchOnFunctionError` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`DestinationConfig`

Una coda Amazon Simple Queue Service (Amazon SQS) o una destinazione tematica di Amazon Simple Notification Service (Amazon SNS) per i record scartati.

Tipo: DestinationConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di una risorsa. `DestinationConfig` `AWS::Lambda::EventSourceMapping`

`Enabled`

Disabilita la mappatura dell'origine eventi per sospendere il polling e la chiamata.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Enabled` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`FilterCriteria`

Un oggetto che definisce i criteri per determinare se Lambda deve elaborare un evento. Per ulteriori informazioni, consulta il filtro AWS Lambda degli eventi nella Guida per gli AWS Lambda sviluppatori.

Tipo: FilterCriteria

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FilterCriteria` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`FunctionResponseTypes`

Un elenco dei tipi di risposta attualmente applicati alla mappatura delle sorgenti degli eventi. Per ulteriori informazioni, consulta la pagina Segnalazione di errori degli elementi batch nella Guida per gli sviluppatori di AWS Lambda .

Valori validi: `ReportBatchItemFailures`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FunctionResponseTypes` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`KmsKeyArn`

L'Amazon Resource Name (ARN) della chiave per crittografare le informazioni relative a questo evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `KmsKeyArn` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumBatchingWindowInSeconds`

Il tempo massimo in secondi per la raccolta dei record prima di richiamare la funzione.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumBatchingWindowInSeconds` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumRecordAgeInSeconds`

L'età massima di un record che Lambda invia a una funzione per l'elaborazione.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumRecordAgeInSeconds` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumRetryAttempts`

Il numero massimo di ripetizioni consentite quando la funzione restituisce un errore.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumRetryAttempts` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MetricsConfig`

Una configurazione opzionale per ottenere metriche avanzate per le mappature delle sorgenti degli eventi che catturano ogni fase dell'elaborazione. Per vedere un esempio, consulta MetricsConfig evento.

Tipo: MetricsConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MetricsConfig` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`ParallelizationFactor`

Il numero di batch da elaborare contemporaneamente da ogni shard.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ParallelizationFactor` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`StartingPosition`

La posizione in un flusso da cui iniziare la lettura.

- `AT_TIMESTAMP`— Specificare l'ora a partire dalla quale iniziare a leggere i record.
- `LATEST`— Leggi solo i nuovi record.
- `TRIM_HORIZON`— Elabora tutti i record disponibili.

Valori validi: `AT_TIMESTAMP` | `LATEST` | `TRIM_HORIZON`

➖Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StartingPosition` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`StartingPositionTimestamp`

L'ora da cui iniziare a leggere, in secondi di tempo Unix. Definisce `StartingPositionTimestamp` quando `StartingPosition` è specificato come`AT_TIMESTAMP`.

Tipo: numero a doppia precisione

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StartingPositionTimestamp` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`Stream`

L'Amazon Resource Name (ARN) del flusso DynamoDB.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventSourceArn` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`TumblingWindowInSeconds`

La durata, in secondi, di una finestra di elaborazione. L'intervallo valido è compreso tra 1 e 900 (15 minuti).

Per ulteriori informazioni, consulta Tumbling windows nella AWS Lambda Developer Guide.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `TumblingWindowInSeconds` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

Esempi

MetricsConfig evento

Di seguito è riportato un esempio di risorsa che utilizza la `MetricsConfig` proprietà per acquisire ogni fase dell'elaborazione per le mappature delle sorgenti degli eventi.

```
Resources:
  FilteredEventsFunction:
    Type: AWS::Serverless::Function
    Properties:
```

```
      CodeUri: s3://sam-demo-bucket/metricsConfig.zip
      Handler: index.handler
      Runtime: nodejs16.x
      Events:
        KinesisStream:
          Type: Kinesis
          Properties:
            Stream: !GetAtt KinesisStream.Arn
            StartingPosition: LATEST
            MetricsConfig:
              Metrics:
              - EventCount
```

Fonte di eventi DynamoDB per la tabella DynamoDB esistente

Fonte di eventi DynamoDB per una tabella DynamoDB già esistente in un account. AWS

YAML

```
Events:
  DDBEvent:
    Type: DynamoDB
    Properties:
      Stream: arn:aws:dynamodb:us-east-1:123456789012:table/TestTable/
stream/2016-08-11T21:21:33.291
      StartingPosition: TRIM_HORIZON
      BatchSize: 10
      Enabled: false
```

Evento DynamoDB per la tabella DynamoDB dichiarato nel modello

Evento DynamoDB per una tabella DynamoDB dichiarata nello stesso file modello.

YAML

```
Events:
  DDBEvent:
    Type: DynamoDB
    Properties:
      Stream:
        !GetAtt MyDynamoDBTable.StreamArn   # This must be the name of a DynamoDB table
 declared in the same template file
```

```
      StartingPosition: TRIM_HORIZON
      BatchSize: 10
      Enabled: false
```

EventBridgeRule

L'oggetto che descrive un tipo di origine di `EventBridgeRule` eventi, che imposta la tua funzione serverless come destinazione di una regola Amazon EventBridge . Per ulteriori informazioni, consulta What Is Amazon EventBridge? nella Amazon EventBridge User Guide.

AWS SAM genera una AWS::Events::Rulerisorsa quando viene impostato questo tipo di evento. AWS SAM crea anche una `AWS::Lambda::Permission` risorsa, necessaria per `EventBridgeRule` poter chiamare Lambda.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
DeadLetterConfig: DeadLetterConfig
EventBusName: String
Input: String
InputPath: String
InputTransformer: InputTransformer
Pattern: EventPattern
RetryPolicy: RetryPolicy
RuleName: String
State: String
Target: Target
```

Properties

`DeadLetterConfig`

Configura la coda Amazon Simple Queue Service (Amazon SQS) a EventBridge cui invia gli eventi dopo una chiamata di destinazione non riuscita. L'invocazione può fallire, ad esempio, quando si invia un evento a una funzione Lambda che non esiste o quando EventBridge le autorizzazioni sono insufficienti per richiamare la funzione Lambda. Per ulteriori informazioni,

consulta la politica di riprova degli eventi e l'utilizzo delle code di lettere morte nella Amazon User Guide. EventBridge

> **Note**
>
> Il tipo di AWS::Serverless::Function risorsa ha un tipo di dati simile`DeadLetterQueue`, che gestisce gli errori che si verificano dopo una chiamata riuscita della funzione Lambda di destinazione. Esempi di questi tipi di errori includono la limitazione Lambda o gli errori restituiti dalla funzione di destinazione Lambda. Per ulteriori informazioni sulla *`DeadLetterQueue`* proprietà della funzione, consulta Dead-letter queues nella Developer Guide.AWS Lambda

Tipo: DeadLetterConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `DeadLetterConfig` proprietà del tipo di dati. `AWS::Events::Rule Target` La AWS SAM versione di questa proprietà include sottoproprietà aggiuntive, nel caso in cui si desideri AWS SAM creare automaticamente la coda di lettere non scritte.

`EventBusName`

Il bus evento da associare a questa regola. Se si omette questa proprietà, AWS SAM utilizza il bus eventi predefinito.

- Tipo: stringa

Obbligatorio: no

Predefinito: bus eventi predefinito

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventBusName` proprietà di una `AWS::Events::Rule` risorsa.

`Input`

Testo JSON valido passato al target. Se utilizzi questa proprietà, al target non viene passata alcuna informazione sul testo dell'evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Input` proprietà di una `AWS::Events::Rule Target` risorsa.

`InputPath`

Quando non vuoi passare l'intero evento corrispondente alla destinazione, usa la `InputPath` proprietà per descrivere quale parte dell'evento trasmettere.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `InputPath` proprietà di una `AWS::Events::Rule Target` risorsa.

`InputTransformer`

Impostazioni che consentono di fornire input personalizzato a un target in base a determinati dati di eventi. Puoi estrarre una o più coppie chiave-valore dall'evento e quindi utilizzare tali dati per l'invio di input personalizzato al target. Per ulteriori informazioni, consulta Amazon EventBridge Input Transformation nella Amazon EventBridge User Guide.

Tipo: InputTransformer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `InputTransformer` proprietà di un tipo di `AWS::Events::Rule Target` dati.

`Pattern`

Descrive quali eventi vengono instradati nella destinazione specificata. Per ulteriori informazioni, consulta EventBridgegli eventi e i modelli di EventBridge eventi di Amazon nella Amazon EventBridge User Guide.

Tipo: EventPattern

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventPattern` proprietà di una `AWS::Events::Rule` risorsa.

`RetryPolicy`

Un oggetto `RetryPolicy` che include informazioni sulle impostazioni dei criteri di nuovi tentativi. Per ulteriori informazioni, consulta la politica di riprova degli eventi e l'utilizzo delle code di lettere morte nella Amazon User Guide. EventBridge

Tipo: RetryPolicy

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RetryPolicy` proprietà del tipo di dati. `AWS::Events::Rule Target`

`RuleName`

Nome della regola .

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::Events::Rule` risorsa.

`State`

Lo stato della regola.

Valori accettati: `DISABLED` | `ENABLED` | `ENABLED_WITH_ALL_CLOUDTRAIL_MANAGEMENT_EVENTS`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `State` proprietà di una `AWS::Events::Rule` risorsa.

`Target`

La AWS risorsa che EventBridge richiama quando viene attivata una regola. È possibile utilizzare questa proprietà per specificare l'ID logico della destinazione. Se questa proprietà non è specificata, AWS SAM genera l'ID logico della destinazione.

Tipo: Target

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `Targets` proprietà di una `AWS::Events::Rule` risorsa. `Amazon EC2 RebootInstances API call`è un esempio di proprietà di destinazione. La AWS SAM versione di questa proprietà consente solo di specificare l'ID logico di una singola destinazione.

Esempi

EventBridgeRule

Di seguito è riportato un esempio di tipo di origine di `EventBridgeRule` evento.

YAML

```
EBRule:
  Type: EventBridgeRule
  Properties:
    Input: '{"Key": "Value"}'
    Pattern:
      detail:
        state:
          - terminated
    RetryPolicy:
      MaximumRetryAttempts: 5
      MaximumEventAgeInSeconds: 900
    DeadLetterConfig:
      Type: SQS
      QueueLogicalId: EBRuleDLQ
    Target:
      Id: MyTarget
```

DeadLetterConfig

L'oggetto utilizzato per specificare la coda Amazon Simple Queue Service (Amazon SQS) in EventBridge cui invia gli eventi dopo una chiamata di destinazione non riuscita. L'invocazione può fallire, ad esempio, quando si invia un evento a una funzione Lambda che non esiste o le autorizzazioni sono insufficienti per richiamare la funzione Lambda. Per ulteriori informazioni, consulta la politica di riprova degli eventi e l'utilizzo delle code di lettere morte nella Amazon User Guide. EventBridge

> **Note**
>
> Il tipo di AWS::Serverless::Function risorsa ha un tipo di dati simile, `DeadLetterQueue` che gestisce gli errori che si verificano dopo una chiamata riuscita della funzione Lambda di destinazione. Esempi di questo tipo di errore includono la limitazione Lambda o gli errori restituiti dalla funzione di destinazione Lambda. Per ulteriori informazioni sulla *`DeadLetterQueue`* proprietà della funzione, consulta Dead-letter queues nella Developer Guide.AWS Lambda

Sintassi

Per dichiarare questa entità nel modello AWS Serverless Application Model (AWS SAM), utilizzate la seguente sintassi.

YAML

```yaml
Arn: String
QueueLogicalId: String
Type: String
```

Properties

`Arn`

L'Amazon Resource Name (ARN) della coda Amazon SQS specificato come destinazione per la coda di lettere non scritte.

> **Note**
>
> Specificare la proprietà o la `Type` proprietà, ma non entrambe. `Arn`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Arn` proprietà del tipo di `AWS::Events::Rule` `DeadLetterConfig` dati.

`QueueLogicalId`

`Type`Viene specificato il nome personalizzato della coda di lettere morte che AWS SAM crea if.

> **Note**
>
> Se la `Type` proprietà non è impostata, viene ignorata.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Type`

Il tipo di coda. Quando questa proprietà è impostata, crea AWS SAM automaticamente una coda di lettere morte e allega la politica basata sulle risorse necessaria per concedere l'autorizzazione alla risorsa delle regole di inviare eventi alla coda.

> **Note**
>
> Specificate la proprietà o la proprietà, ma non entrambe. `Type` `Arn`

Valori validi: `SQS`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

DeadLetterConfig

DeadLetterConfig

YAML

```
DeadLetterConfig:
  Type: SQS
  QueueLogicalId: MyDLQ
```

Target

Configura la AWS risorsa che EventBridge richiama quando viene attivata una regola.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  Id: String
```

Properties

`Id`

L'ID logico della destinazione.

Il valore di `Id` può includere caratteri alfanumerici, punti (`.`), trattini () e caratteri di - sottolineatura (). _

-Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Id` proprietà del tipo di dati. `AWS::Events::Rule Target`

Esempi

Target

YAML

```
EBRule:
```

```
  Type: EventBridgeRule
  Properties:
    Target:
      Id: MyTarget
```

HttpApi

L'oggetto che descrive un'origine di eventi con tipo HttpApi.

Se nell'API esiste una OpenApi definizione per il percorso e il metodo specificati, SAM aggiungerà automaticamente la sezione sull'integrazione e la sicurezza Lambda (se applicabile).

Se nell'API non esiste alcuna OpenApi definizione per il percorso e il metodo specificati, SAM la creerà automaticamente.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
ApiId: String
Auth: HttpApiFunctionAuth
Method: String
Path: String
PayloadFormatVersion: String
RouteSettings: RouteSettings
TimeoutInMillis: Integer
```

Properties

`ApiId`

Identificatore di una AWS::Serverless::HttpApi risorsa definita in questo modello.

Se non è definita, viene creata una AWS::Serverless::HttpApi risorsa predefinita chiamata `ServerlessHttpApi` utilizzando un OpenApi documento generato contenente un'unione di tutti i percorsi e i metodi definiti dagli eventi Api definiti in questo modello che non specificano un`ApiId`.

Questo non può fare riferimento a una AWS::Serverless::HttpApi risorsa definita in un altro modello.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Auth`

Configurazione di autenticazione per questo specifico metodo Api+Path+.

Utile per sovrascrivere le API `DefaultAuthorizer` o impostare la configurazione di autenticazione su un percorso individuale quando non è specificato. `DefaultAuthorizer`

Tipo: HttpApiFunctionAuth

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`Method`

Metodo HTTP per il quale viene richiamata questa funzione.

Se non `Method` viene specificato alcun `Path` e, SAM creerà un percorso API predefinito che indirizza qualsiasi richiesta che non sia mappata a un endpoint diverso per questa funzione Lambda. Può esistere solo uno di questi percorsi predefiniti per API.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Path`

Percorso Uri per il quale viene richiamata questa funzione. Deve iniziare con`/`.

Se non `Method` viene specificato alcun `Path` e, SAM creerà un percorso API predefinito che indirizza qualsiasi richiesta che non sia mappata a un endpoint diverso per questa funzione Lambda. Può esistere solo uno di questi percorsi predefiniti per API.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`PayloadFormatVersion`

Specifica il formato del payload inviato a un'integrazione.

NOTA: PayloadFormatVersion richiede che SAM modifichi la definizione di OpenAPI, quindi funziona solo con inline OpenApi definito nella proprietà. `DefinitionBody`

- Tipo: stringa

Obbligatorio: no

Predefinito: 2.0

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`RouteSettings`

Le impostazioni del percorso per percorso per questa API HTTP. Per ulteriori informazioni sulle impostazioni del percorso, consulta AWS::ApiGatewayV2::Stage RouteSettingsla Guida per gli sviluppatori di API Gateway.

Nota: se RouteSettings sono specificate sia nella HttpApi risorsa che nell'origine dell'evento, le AWS SAM unisce alle proprietà dell'origine dell'evento che hanno la precedenza.

Tipo: RouteSettings

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RouteSettings` proprietà di una `AWS::ApiGatewayV2::Stage` risorsa.

`TimeoutInMillis`

Timeout personalizzato tra 50 e 29.000 millisecondi.

NOTA: TimeoutInMillis richiede che SAM modifichi la definizione di OpenAPI, quindi funziona solo con inline OpenApi definito nella proprietà. `DefinitionBody`

Tipo: integer

Obbligatorio: no

Valore predefinito: 5000

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

HttpApi Evento predefinito

HttpApi Evento che utilizza il percorso predefinito. Tutti i percorsi e i metodi non mappati su questa API verranno indirizzati a questo endpoint.

YAML

```
Events:
  HttpApiEvent:
    Type: HttpApi
```

HttpApi

HttpApi Evento che utilizza un percorso e un metodo specifici.

YAML

```
Events:
  HttpApiEvent:
    Type: HttpApi
    Properties:
      Path: /
      Method: GET
```

HttpApi Autorizzazione

HttpApi Evento che utilizza un Authorizer.

YAML

```
Events:
```

```
  HttpApiEvent:
    Type: HttpApi
    Properties:
      Path: /authenticated
      Method: GET
      Auth:
        Authorizer: OpenIdAuth
        AuthorizationScopes:
          - scope1
          - scope2
```

HttpApiFunctionAuth

Configura l'autorizzazione a livello di evento.

Configura l'autenticazione per un metodo API + Path + specifico

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
AuthorizationScopes: List
Authorizer: String
```

Properties

`AuthorizationScopes`

Gli ambiti di autorizzazione da applicare a questa API, percorso e metodo.

Gli ambiti elencati qui sostituiranno tutti gli ambiti applicati da, se ne esiste uno. `DefaultAuthorizer`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`Authorizer`

Il `Authorizer` per una funzione specifica. Per utilizzare l'autorizzazione IAM, specifica `AWS_IAM` e specifica `true` for `EnableIamAuthorizer` nella `Globals` sezione del modello.

Se hai specificato un Global Authorizer sull'API e desideri rendere pubblica una funzione specifica, esegui l'override `Authorizer` impostando su. `NONE`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

Autenticazione della funzione

Specificazione dell'autorizzazione a livello di funzione

YAML

```
Auth:
  Authorizer: OpenIdAuth
  AuthorizationScopes:
    - scope1
    - scope2
```

Autorizzazione IAM

Specifica l'autorizzazione IAM a livello di evento. Per utilizzare `AWS_IAM` l'autorizzazione a livello di evento, è inoltre necessario specificare `true` for `EnableIamAuthorizer` nella `Globals` sezione del modello. Per ulteriori informazioni, consulta Sezione Globals del modello AWS SAM.

YAML

```
Globals:
  HttpApi:
    Auth:
      EnableIamAuthorizer: true
```

```
Resources:
  HttpApiFunctionWithIamAuth:
    Type: AWS::Serverless::Function
    Properties:
      Events:
        ApiEvent:
          Type: HttpApi
          Properties:
            Path: /iam-auth
            Method: GET
            Auth:
              Authorizer: AWS_IAM
      Handler: index.handler
      InlineCode: |
        def handler(event, context):
          return {'body': 'HttpApiFunctionWithIamAuth', 'statusCode': 200}
      Runtime: python3.9
```

IoTRule

L'oggetto che descrive un tipo di sorgente di `IoTRule` eventi.

Crea una AWS::IoT::TopicRulerisorsa per dichiarare una AWS IoT regola. Per ulteriori informazioni, consulta la documentazione CloudFormation

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  AwsIotSqlVersion: String
  Sql: String
```

Properties

`AwsIotSqlVersion`

Versione del motore di regole SQL da usare durante la valutazione della regola.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `AwsIotSqlVersion` proprietà di una `AWS::IoT::TopicRule TopicRulePayload` risorsa.

`Sql`

Istruzione SQL usata per eseguire query sull'argomento. Per ulteriori informazioni, vedere AWS IoT SQL Reference nella AWS IoT Developer Guide.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Sql` proprietà di una `AWS::IoT::TopicRule TopicRulePayload` risorsa.

Esempi

Regola IOT

Esempio di regola IOT

YAML

```
IoTRule:
  Type: IoTRule
  Properties:
    Sql: SELECT * FROM 'topic/test'
```

Kinesis

L'oggetto che descrive un tipo di origine `Kinesis` dell'evento. Per ulteriori informazioni, consulta Using AWS Lambda with Amazon Kinesis nella AWS Lambda Developer Guide.

AWS SAM genera una AWS::Lambda::EventSourceMappingrisorsa quando questo tipo di evento è impostato.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
BatchSize: Integer
BisectBatchOnFunctionError: Boolean
DestinationConfig: DestinationConfig
Enabled: Boolean
FilterCriteria: FilterCriteria
FunctionResponseTypes: List
KmsKeyArn: String
MaximumBatchingWindowInSeconds: Integer
MaximumRecordAgeInSeconds: Integer
MaximumRetryAttempts: Integer
MetricsConfig: MetricsConfig
ParallelizationFactor: Integer
StartingPosition: String
StartingPositionTimestamp: Double
Stream: String
TumblingWindowInSeconds: Integer
```

Properties

`BatchSize`

Il numero massimo di elementi da recuperare in un singolo batch.

Tipo: integer

Obbligatorio: no

Impostazione predefinita: 100

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `BatchSize` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

Minimum: 1

Maximum: `10000`

`BisectBatchOnFunctionError`

Se la funzione restituisce un errore, dividi il batch in due e riprova.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `BisectBatchOnFunctionError` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`DestinationConfig`

Una coda Amazon Simple Queue Service (Amazon SQS) o una destinazione tematica di Amazon Simple Notification Service (Amazon SNS) per i record scartati.

Tipo: DestinationConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di una risorsa. `DestinationConfig` `AWS::Lambda::EventSourceMapping`

`Enabled`

Disabilita la mappatura dell'origine eventi per sospendere il polling e la chiamata.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Enabled` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`FilterCriteria`

Un oggetto che definisce i criteri per determinare se Lambda deve elaborare un evento. Per ulteriori informazioni, consulta il filtro AWS Lambda degli eventi nella Guida per gli AWS Lambda sviluppatori.

Tipo: FilterCriteria

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FilterCriteria` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`FunctionResponseTypes`

Un elenco dei tipi di risposta attualmente applicati alla mappatura delle sorgenti degli eventi. Per ulteriori informazioni, consulta la pagina Segnalazione di errori degli elementi batch nella Guida per gli sviluppatori di AWS Lambda .

Valori validi: `ReportBatchItemFailures`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FunctionResponseTypes` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`KmsKeyArn`

L'Amazon Resource Name (ARN) della chiave per crittografare le informazioni relative a questo evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `KmsKeyArn` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumBatchingWindowInSeconds`

Il tempo massimo in secondi per la raccolta dei record prima di richiamare la funzione.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumBatchingWindowInSeconds` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumRecordAgeInSeconds`

L'età massima di un record che Lambda invia a una funzione per l'elaborazione.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumRecordAgeInSeconds` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumRetryAttempts`

Il numero massimo di ripetizioni consentite quando la funzione restituisce un errore.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumRetryAttempts` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MetricsConfig`

Una configurazione opzionale per ottenere metriche avanzate per le mappature delle sorgenti degli eventi che catturano ogni fase dell'elaborazione. Per vedere un esempio, consulta MetricsConfig evento.

Tipo: MetricsConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MetricsConfig` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`ParallelizationFactor`

Il numero di batch da elaborare contemporaneamente da ogni shard.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ParallelizationFactor` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`StartingPosition`

La posizione in un flusso da cui iniziare la lettura.

- `AT_TIMESTAMP`— Specificare l'ora a partire dalla quale iniziare a leggere i record.
- `LATEST`— Leggi solo i nuovi record.
- `TRIM_HORIZON`— Elabora tutti i record disponibili.

Valori validi: `AT_TIMESTAMP` | `LATEST` | `TRIM_HORIZON`

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StartingPosition` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`StartingPositionTimestamp`

L'ora da cui iniziare a leggere, in secondi di tempo Unix. Definisce `StartingPositionTimestamp` quando `StartingPosition` è specificato come`AT_TIMESTAMP`.

Tipo: numero a doppia precisione

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StartingPositionTimestamp` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`Stream`

L'Amazon Resource Name (ARN) del flusso di dati o di un consumatore di streaming.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventSourceArn` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`TumblingWindowInSeconds`

La durata, in secondi, di una finestra di elaborazione. L'intervallo valido è compreso tra 1 e 900 (15 minuti).

Per ulteriori informazioni, consulta Tumbling windows nella AWS Lambda Developer Guide.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `TumblingWindowInSeconds` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

Esempi

MetricsConfig evento

Di seguito è riportato un esempio di risorsa che utilizza la `MetricsConfig` proprietà per acquisire ogni fase dell'elaborazione per le mappature delle sorgenti degli eventi.

```
Resources:
  FilteredEventsFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: s3://sam-demo-bucket/metricsConfig.zip
      Handler: index.handler
      Runtime: nodejs16.x
      Events:
        KinesisStream:
          Type: Kinesis
          Properties:
            Stream: !GetAtt KinesisStream.Arn
            StartingPosition: LATEST
            MetricsConfig:
              Metrics:
              - EventCount
```

Fonte di eventi Kinesis

Di seguito è riportato un esempio di sorgente di eventi Kinesis.

YAML

```
Events:
  KinesisEvent:
    Type: Kinesis
    Properties:
      Stream: arn:aws:kinesis:us-east-1:123456789012:stream/my-stream
      StartingPosition: TRIM_HORIZON
      BatchSize: 10
      Enabled: false
      FilterCriteria:
```

```
        Filters:
          - Pattern: '{"key": ["val1", "val2"]}'
```

MQ

L'oggetto che descrive un tipo di origine `MQ` dell'evento. Per ulteriori informazioni, consulta Using Lambda with Amazon MQ nella AWS Lambda Developer Guide.

AWS Serverless Application Model (AWS SAM) genera una AWS::Lambda::EventSourceMappingrisorsa quando viene impostato questo tipo di evento.

> **Note**
>
> Per avere una coda Amazon MQ in un cloud privato virtuale (VPC) che si connette a una funzione Lambda in una rete pubblica, il ruolo di esecuzione della funzione deve includere le seguenti autorizzazioni:
>
> - `ec2:CreateNetworkInterface`
> - `ec2:DeleteNetworkInterface`
> - `ec2:DescribeNetworkInterfaces`
> - `ec2:DescribeSecurityGroups`
> - `ec2:DescribeSubnets`
> - `ec2:DescribeVpcs`
>
> Per ulteriori informazioni, consulta Autorizzazioni per i ruoli di esecuzione nella Guida per gli sviluppatori.AWS Lambda

Sintassi

Per dichiarare questa entità nel tuo AWS SAM modello, usa la seguente sintassi.

YAML

```yaml
BatchSize: Integer
Broker: String
DynamicPolicyName: Boolean
Enabled: Boolean
```

```
  FilterCriteria: FilterCriteria
  KmsKeyArn: String
  MaximumBatchingWindowInSeconds: Integer
  Queues: List
  SecretsManagerKmsKeyId: String
  SourceAccessConfigurations: List
```

Properties

`BatchSize`

Il numero massimo di elementi da recuperare in un singolo batch.

Tipo: integer

Obbligatorio: no

Impostazione predefinita: 100

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `BatchSize` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

Minimum: 1

Maximum: `10000`

`Broker`

L'Amazon Resource Name (ARN) del gestore di identità di Amazon MQ.

Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventSourceArn` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`DynamicPolicyName`

Per impostazione predefinita, il nome della policy AWS Identity and Access Management (IAM) serve `SamAutoGeneratedAMQPolicy` per la compatibilità con le versioni precedenti. Specificate `true` di utilizzare un nome generato automaticamente per la vostra policy IAM. Questo nome includerà l'ID logico dell'origine dell'evento Amazon MQ.

> **Note**
>
> Quando utilizzi più di una fonte di eventi Amazon MQ, specifica `true` per evitare nomi di policy IAM duplicati.

Tipo: Booleano

Obbligatorio: no

Default: `false`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Enabled`

Se`true`, la mappatura della fonte dell'evento è attiva. Per mettere in pausa il polling e la chiamata, imposta su. `false`

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Enabled` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`FilterCriteria`

Un oggetto che definisce i criteri che determinano se Lambda deve elaborare un evento. Per ulteriori informazioni, consulta il filtro AWS Lambda degli eventi nella Guida per gli AWS Lambda sviluppatori.

Tipo: FilterCriteria

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FilterCriteria` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`KmsKeyArn`

L'Amazon Resource Name (ARN) della chiave per crittografare le informazioni relative a questo evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `KmsKeyArn` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumBatchingWindowInSeconds`

Il tempo massimo in secondi per la raccolta dei record prima di richiamare la funzione.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumBatchingWindowInSeconds` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`Queues`

Il nome della coda di destinazione del broker Amazon MQ da utilizzare.

Tipo: elenco

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Queues` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`SecretsManagerKmsKeyId`

L'ID della chiave AWS Key Management Service (AWS KMS) di una chiave gestita dal cliente da Gestione dei segreti AWS. Richiesto quando si utilizza una chiave gestita dal cliente di Secrets Manager con un ruolo di esecuzione Lambda che non include l'`kms:Decrypt`autorizzazione.

Il valore di questa proprietà è un UUID. Ad esempio: `1abc23d4-567f-8ab9-cde0-1fab234c5d67`.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`SourceAccessConfigurations`

Una matrice del protocollo di autenticazione o dell'host virtuale. Specificalo utilizzando il tipo di SourceAccessConfigurationsdati.

Per il tipo di origine `MQ` dell'evento, gli unici tipi di configurazione validi sono `BASIC_AUTH` e`VIRTUAL_HOST`.

- **`BASIC_AUTH`**— Il segreto di Secrets Manager che memorizza le credenziali del broker. Per questo tipo, la credenziale deve essere nel seguente formato:. `{"username": "your-username", "password": "your-password"}` È consentito un solo oggetto di tipo`BASIC_AUTH`.
- **`VIRTUAL_HOST`**— Il nome dell'host virtuale nel tuo broker RabbitMQ. Lambda utilizzerà l'host di questo Rabbit MQ come origine dell'evento. È consentito un solo oggetto di tipo`VIRTUAL_HOST`.

Tipo: elenco

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SourceAccessConfigurations` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

Esempi

Fonte di eventi Amazon MQ

Di seguito è riportato un esempio di tipo di origine di `MQ` eventi per un broker Amazon MQ.

YAML

```
Events:
  MQEvent:
    Type: MQ
    Properties:
      Broker: arn:aws:mq:us-
east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9
      Queues: List of queues
      SourceAccessConfigurations:
```

```
        - Type: BASIC_AUTH
          URI: arn:aws:secretsmanager:us-east-1:01234567890:secret:MyBrokerSecretName
      BatchSize: 200
      Enabled: true
```

MSK

L'oggetto che descrive un tipo di sorgente di `MSK` eventi. Per ulteriori informazioni, consulta Using AWS Lambda with Amazon MSK nella AWS Lambda Developer Guide.

AWS Serverless Application Model (AWS SAM) genera una AWS::Lambda::EventSourceMappingrisorsa quando viene impostato questo tipo di evento.

Per utilizzare Schema Registry, devi definire autorizzazioni specifiche per i ruoli IAM per la tua funzione. Vedi Configurazione completa con ruoli IAM per un esempio della configurazione richiesta.

Sintassi

Per dichiarare questa entità nel tuo AWS SAM modello, usa la seguente sintassi.

YAML

```yaml
BatchSize: Integer
BisectBatchOnFunctionError: Boolean
ConsumerGroupId: String
DestinationConfig: DestinationConfig
Enabled: Boolean
FilterCriteria: FilterCriteria
FunctionResponseTypes: List
KmsKeyArn: String
MaximumBatchingWindowInSeconds: Integer
MaximumRecordAgeInSeconds: Integer
MaximumRetryAttempts: Integer
LoggingConfig: LoggingConfig
MetricsConfig: MetricsConfig
ProvisionedPollerConfig: ProvisionedPollerConfig
SchemaRegistryConfig: SchemaRegistryConfig
SourceAccessConfigurations: SourceAccessConfigurations
StartingPosition: String
StartingPositionTimestamp: Double
Stream: String
Topics: List
```

Properties

`BatchSize`

Il numero massimo di registri in ogni batch che Lambda estrae dal flusso o dalla coda e invia alla funzione. Lambda passa tutti i registri del batch alla funzione in una singola chiamata, fino al limite di payload per la chiamata sincrona (6 MB).

Impostazione predefinita: 100

Intervallo valido: valore minimo di 1. Valore massimo di 10.000.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `BatchSize` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`BisectBatchOnFunctionError`

Se la funzione restituisce un errore, dividi il batch in due e riprova.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `BisectBatchOnFunctionError` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`ConsumerGroupId`

Una stringa che configura il modo in cui gli eventi verranno letti dagli argomenti di Kafka.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `AmazonManagedKafkaConfiguration` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`DestinationConfig`

Oggetto di configurazione che specifica la destinazione di un evento dopo che Lambda lo elabora.

Utilizza questa proprietà per specificare la destinazione delle chiamate non riuscite dall'origine dell'evento Amazon MSK.

Tipo: DestinationConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `DestinationConfig` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`Enabled`

Disabilita la mappatura dell'origine eventi per sospendere il polling e la chiamata.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Enabled` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`FilterCriteria`

Un oggetto che definisce i criteri che determinano se Lambda deve elaborare un evento. Per ulteriori informazioni, consulta il filtro AWS Lambda degli eventi nella Guida per gli AWS Lambda sviluppatori.

Tipo: FilterCriteria

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FilterCriteria` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`FunctionResponseTypes`

Un elenco dei tipi di risposta attualmente applicati alla mappatura delle sorgenti degli eventi. Per ulteriori informazioni, consulta la pagina Segnalazione di errori degli elementi batch nella Guida per gli sviluppatori di AWS Lambda .

Valori validi: `ReportBatchItemFailures`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FunctionResponseTypes` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`KmsKeyArn`

L'Amazon Resource Name (ARN) della chiave per crittografare le informazioni relative a questo evento.

Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `KmsKeyArn` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumBatchingWindowInSeconds`

Il tempo massimo in secondi per la raccolta dei record prima di richiamare la funzione.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumBatchingWindowInSeconds` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumRecordAgeInSeconds`

L'età massima di un record che Lambda invia a una funzione per l'elaborazione.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumRecordAgeInSeconds` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumRetryAttempts`

Il numero massimo di ripetizioni consentite quando la funzione restituisce un errore.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumRetryAttempts` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`LoggingConfig`

Un oggetto di configurazione che specifica la configurazione di registrazione per la mappatura dell'origine degli eventi.

Tipo: LoggingConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `LoggingConfig` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`MetricsConfig`

Un oggetto di configurazione che specifica la configurazione delle metriche per la mappatura dell'origine degli eventi.

Tipo: MetricsConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MetricsConfig` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`ProvisionedPollerConfig`

Configurazione per aumentare la quantità di poller utilizzati per calcolare le mappature delle sorgenti degli eventi. Questa configurazione consente un minimo di 1 poller e un massimo di 2000 poller. Per un esempio, fare riferimento a. ProvisionedPollerConfig esempio

Tipo: ProvisionedPollerConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ProvisionedPollerConfig` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`SchemaRegistryConfig`

Configurazione per l'utilizzo di un registro degli schemi con l'origine degli eventi Kafka.

> **Note**
>
> Questa funzionalità deve essere `ProvisionedPollerConfig` configurata.

Tipo: SchemaRegistryConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `AmazonManagedKafkaEventSourceConfig` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`SourceAccessConfigurations`

Un array del protocollo di autenticazione, dei componenti VPC o dell'host virtuale per proteggere e definire l'origine eventi.

Valori validi: `CLIENT_CERTIFICATE_TLS_AUTH`

Tipo: elenco di SourceAccessConfiguration

Obbligatorio: no

CloudFormation compatibilità: questa proprietà fa parte della AmazonManagedKafkaEventSourceConfigproprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`StartingPosition`

La posizione in un flusso da cui iniziare la lettura.

- `AT_TIMESTAMP`— Specificare l'ora a partire dalla quale iniziare a leggere i record.
- `LATEST`— Leggi solo i nuovi record.
- `TRIM_HORIZON`— Elabora tutti i record disponibili.

Valori validi: `AT_TIMESTAMP` | `LATEST` | `TRIM_HORIZON`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StartingPosition` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`StartingPositionTimestamp`

L'ora da cui iniziare a leggere, in secondi di tempo Unix. Definisce `StartingPositionTimestamp` quando `StartingPosition` è specificato come`AT_TIMESTAMP`.

Tipo: numero a doppia precisione

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StartingPositionTimestamp` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`Stream`

L'Amazon Resource Name (ARN) del flusso di dati o di un consumatore di streaming.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventSourceArn` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`Topics`

Il nome dell'argomento Kafka.

Tipo: elenco

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Topics` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

Esempi

Configurazione completa con ruoli IAM

L'esempio seguente mostra una configurazione completa che include la configurazione del ruolo IAM richiesta per l'utilizzo di Schema Registry:

```
Parameters:
  PreCreatedSubnetOne:
    Type: String
  PreCreatedSubnetTwo:
    Type: String
  MskClusterName4:
    Type: String

Resources:
  MyLambdaExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
        - Action: [sts:AssumeRole]
          Effect: Allow
          Principal:
            Service: [lambda.amazonaws.com]
      Policies:
      - PolicyName: KafkaClusterPermissions
        PolicyDocument:
          Statement:
          - Action: [kafka:DescribeClusterV2, kafka:GetBootstrapBrokers]
            Effect: Allow
            Resource: 'arn:aws:kafka:us-east-1:123456789012:cluster/*'
      - PolicyName: KafkaAuthPolicy
        PolicyDocument:
          Statement:
          - Action: [secretsmanager:GetSecretValue, kms:Decrypt]
            Effect: "Allow"
            Resource: ['arn:aws:secretsmanager:us-
west-2:123456789012:secret:kafkaSecret-******',
                        'arn:aws:kms:us-west-2:123456789012:key/keyId']
      - PolicyName: ENIPolicy
        PolicyDocument:
          Statement:
```

```
          - Action: [ec2:CreateNetworkInterface,
              ec2:DescribeNetworkInterfaces, ec2:DescribeVpcs,
 ec2:DeleteNetworkInterface,
              ec2:DescribeSubnets, ec2:DescribeSecurityGroups]
            Effect: Allow
            Resource: '*'
      - PolicyName: SchemaRegistryPolicy
        PolicyDocument:
          Statement:
          - Action: [glue:GetRegistry]
            Effect: Allow
            Resource: 'arn:aws:glue:{region}:{account-id}:registry/{registry-name}'
      - PolicyName: SchemaVersionsPolicy
        PolicyDocument:
          Statement:
          - Action: [glue:GetSchemaVersions]
            Effect: Allow
            Resource: '*'
      ManagedPolicyArns:
      - !Sub arn:${AWS::Partition}:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole
      Tags:
      - {Value: SAM, Key: lambda:createdBy}

  MyMskCluster:
    Type: AWS::MSK::Cluster
    Properties:
      BrokerNodeGroupInfo:
        ClientSubnets:
        - Ref: PreCreatedSubnetOne
        - Ref: PreCreatedSubnetTwo
        InstanceType: kafka.t3.small
        StorageInfo:
          EBSStorageInfo:
            VolumeSize: 1
      ClusterName:
        Ref: MskClusterName4
      KafkaVersion: 3.8.x
      NumberOfBrokerNodes: 2

  MyMskStreamProcessor:
    Type: AWS::Serverless::Function
    Properties:
      Runtime: nodejs18.x
```

```
      Handler: index.handler
      CodeUri: ${codeuri}
      Role:
        Fn::GetAtt: [MyLambdaExecutionRole, Arn]
      Events:
        MyMskEvent:
          Type: MSK
          Properties:
            StartingPosition: LATEST
            Stream:
              Ref: MyMskCluster
            SourceAccessConfigurations:
            - Type: SASL_SCRAM_512_AUTH
              URI: !Sub arn:${AWS::Partition}:secretsmanager:us-
west-2:123456789012:secret:my-path/my-secret-name-1a2b3c
            Topics:
            - SchemaRegistryTestTopic
            ProvisionedPollerConfig:
              MinimumPollers: 1
            SchemaRegistryConfig:
              AccessConfigs:
              - Type: BASIC_AUTH
                URI: !Sub arn:${AWS::Partition}:secretsmanager:us-
west-2:123456789012:secret:my-path/my-secret-name-1a2b3c
              SchemaValidationConfigs:
              - Attribute: KEY
              EventRecordFormat: JSON
              SchemaRegistryURI: !Sub arn:${AWS::Partition}:glue:us-
west-2:123456789012:registry/myregistry
```

ProvisionedPollerConfig esempio

```
ProvisionedPollerConfig:
  MinimumPollers: 1
  MaximumPollers: 200
```

Esempio di Amazon MSK per un cluster esistente

Di seguito è riportato un esempio di un tipo di origine di `MSK` eventi per un cluster Amazon MSK che esiste già in un Account AWS.

YAML

```
Events:
  MSKEvent:
    Type: MSK
    Properties:
      StartingPosition: LATEST
      Stream: arn:aws:kafka:us-east-1:012345678012:cluster/exampleClusterName/
abcdefab-1234-abcd-5678-cdef0123ab01-2
      Topics:
        - MyTopic
```

Esempio di Amazon MSK per un cluster dichiarato nello stesso modello

Di seguito è riportato un esempio di un tipo di origine di `MSK` eventi per un cluster Amazon MSK dichiarato nello stesso file modello.

YAML

```
Events:
  MSKEvent:
    Type: MSK
    Properties:
      StartingPosition: LATEST
      Stream:
        Ref: MyMskCluster   # This must be the name of an MSK cluster declared in the
 same template file
      Topics:
        - MyTopic
```

Fonte di eventi MSK con registro dello schema

Di seguito è riportato un esempio di un tipo di origine di `MSK` eventi configurato con un registro dello schema.

```
Events:
  MSKEvent:
    Type: MSK
    Properties:
      StartingPosition: LATEST
      Stream:
```

```
        Ref: MyMskCluster
      Topics:
        - SchemaRegistryTestTopic
      ProvisionedPollerConfig:
        MinimumPollers: 1
      SchemaRegistryConfig:
        SchemaRegistryURI: !Sub arn:${AWS::Partition}:glue:us-
west-2:123456789012:registry/myregistry
        EventRecordFormat: JSON
        SchemaValidationConfigs:
          - Attribute: KEY
          - Attribute: VALUE
```

Fonte di eventi MSK con registro degli schemi Confluent

Di seguito è riportato un esempio di un tipo di origine di `MSK` eventi configurato con un Confluent Schema Registry.

```
Events:
  MSKEvent:
    Type: MSK
    Properties:
      StartingPosition: LATEST
      Stream:
        Ref: MyMskCluster
      Topics:
        - SchemaRegistryTestTopic
      ProvisionedPollerConfig:
        MinimumPollers: 1
      SchemaRegistryConfig:
        SchemaRegistryURI: https://my-schema-registry.confluent.cloud
        AccessConfigs:
          - Type: BASIC_AUTH
            URI: !Sub arn:${AWS::Partition}:secretsmanager:us-
west-2:123456789012:secret:my-secret
        EventRecordFormat: JSON
        SchemaValidationConfigs:
          - Attribute: KEY
          - Attribute: VALUE
```

S3

L'oggetto che descrive un tipo di origine `S3` dell'evento.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Bucket: String
Events: String | List
Filter: NotificationFilter
```

Properties

`Bucket`

Nome bucket S3. Questo bucket deve esistere nello stesso modello.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è simile alla `BucketName` proprietà di una `AWS::S3::Bucket` risorsa. Questo è un campo obbligatorio in SAM. Questo campo accetta solo un riferimento al bucket S3 creato in questo modello

`Events`

L'evento bucket Amazon S3 per il quale richiamare la funzione Lambda. Consulta i tipi di eventi supportati da Amazon S3 per un elenco di valori validi.

Tipo: Stringa | Elenco

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà viene passata direttamente alla `Event` proprietà del tipo di `AWS::S3::Bucket` `LambdaConfiguration` dati.

`Filter`

Le regole di filtro che determinano quali oggetti Amazon S3 richiamano la funzione Lambda. Per informazioni sul filtraggio dei nomi chiave di Amazon S3, consulta Configuring Amazon S3 Event Notifications nella Amazon Simple Storage Service User Guide.

Tipo: NotificationFilter

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Filter` proprietà del tipo di dati. `AWS::S3::Bucket LambdaConfiguration`

Esempi

Evento S3

Esempio di evento S3.

YAML

```
Events:
  S3Event:
    Type: S3
    Properties:
      Bucket:
        Ref: ImagesBucket     # This must be the name of an S3 bucket declared in the
 same template file
      Events: s3:ObjectCreated:*
      Filter:
        S3Key:
          Rules:
          - Name: prefix      # or "suffix"
            Value: value      # The value to search for in the S3 object key names
```

Schedule

L'oggetto che descrive un tipo di origine `Schedule` degli eventi, che imposta la funzione serverless come destinazione di una EventBridge regola Amazon che si attiva in base a una pianificazione. Per ulteriori informazioni, consulta What Is Amazon EventBridge? nella Amazon EventBridge User Guide.

AWS Serverless Application Model (AWS SAM) genera una AWS::Events::Rulerisorsa quando viene impostato questo tipo di evento.

> **Note**
>
> EventBridge ora offre una nuova funzionalità di pianificazione, Amazon EventBridge Scheduler. Amazon EventBridge Scheduler è uno strumento di pianificazione senza server che ti consente di creare, eseguire e gestire attività da un unico servizio gestito centralizzato.

EventBridge Schedulerè altamente personalizzabile e offre una migliore scalabilità rispetto alle regole EventBridge pianificate, con un set più ampio di operazioni API mirate e. Servizi AWS
Ti consigliamo di utilizzare EventBridge Scheduler per invocare le destinazioni in base a una pianificazione. Per definire questo tipo di origine degli eventi nei tuoi AWS SAM modelli, consultaScheduleV2.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
DeadLetterConfig: DeadLetterConfig
Description: String
Enabled: Boolean
Input: String
Name: String
RetryPolicy: RetryPolicy
Schedule: String
State: String
```

Properties

`DeadLetterConfig`

Configura la coda Amazon Simple Queue Service (Amazon SQS) a EventBridge cui invia gli eventi dopo una chiamata di destinazione non riuscita. L'invocazione può fallire, ad esempio, quando si invia un evento a una funzione Lambda che non esiste o quando EventBridge le autorizzazioni sono insufficienti per richiamare la funzione Lambda. Per ulteriori informazioni, consulta la politica di riprova degli eventi e l'utilizzo delle code di lettere morte nella Amazon User Guide. EventBridge

> **Note**
>
> Il tipo di AWS::Serverless::Function risorsa ha un tipo di dati simile`DeadLetterQueue`, che gestisce gli errori che si verificano dopo una chiamata riuscita della funzione Lambda di destinazione. Esempi di questi tipi di errori includono la limitazione Lambda o gli

errori restituiti dalla funzione di destinazione Lambda. Per ulteriori informazioni sulla *`DeadLetterQueue`* proprietà della funzione, consulta Dead-letter queues nella Developer Guide.AWS Lambda

Tipo: DeadLetterConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `DeadLetterConfig` proprietà del tipo di dati. `AWS::Events::Rule Target` La AWS SAM versione di questa proprietà include sottoproprietà aggiuntive, nel caso in cui si desideri AWS SAM creare automaticamente la coda di lettere non scritte.

`Description`

Una descrizione della regola.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una risorsa. `AWS::Events::Rule`

`Enabled`

Indica se la regola è abilitata.

Per disabilitare la regola, imposta questa proprietà su`false`.

> **Note**
>
> Specificate la `State` proprietà `Enabled` o, ma non entrambe.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `State` proprietà di una `AWS::Events::Rule` risorsa. Se questa proprietà è impostata su `true` allora AWS SAM passa`ENABLED`, altrimenti passa`DISABLED`.

`Input`

Testo JSON valido passato al target. Se utilizzi questa proprietà, al target non viene passata alcuna informazione sul testo dell'evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Input` proprietà di una `AWS::Events::Rule Target` risorsa.

`Name`

Nome della regola . Se non specifichi un nome, CloudFormation genera un ID fisico univoco e utilizza tale ID per il nome della regola.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::Events::Rule` risorsa.

`RetryPolicy`

Un oggetto `RetryPolicy` che include informazioni sulle impostazioni dei criteri di nuovi tentativi. Per ulteriori informazioni, consulta la politica di riprova degli eventi e l'utilizzo delle code di lettere morte nella Amazon User Guide. EventBridge

Tipo: RetryPolicy

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RetryPolicy` proprietà del tipo di dati. `AWS::Events::Rule Target`

`Schedule`

L'espressione di programmazione che determina il momento e la frequenza di esecuzione della regola. Per ulteriori informazioni, consulta Pianificazione di espressioni per regole.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ScheduleExpression` proprietà di una `AWS::Events::Rule` risorsa.

`State`

Lo stato della regola.

Valori accettati: `DISABLED | ENABLED`

> **Note**
>
> Specificate la `State` proprietà `Enabled` o, ma non entrambe.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `State` proprietà di una `AWS::Events::Rule` risorsa.

Esempi

CloudWatch Pianifica evento

CloudWatch Esempio di evento di pianificazione

YAML

```
CWSchedule:
  Type: Schedule
  Properties:
    Schedule: 'rate(1 minute)'
    Name: TestSchedule
    Description: test schedule
    Enabled: false
```

DeadLetterConfig

L'oggetto utilizzato per specificare la coda Amazon Simple Queue Service (Amazon SQS) in EventBridge cui invia gli eventi dopo una chiamata di destinazione non riuscita. L'invocazione

può fallire, ad esempio, quando si invia un evento a una funzione Lambda che non esiste o le autorizzazioni sono insufficienti per richiamare la funzione Lambda. Per ulteriori informazioni, consulta la politica di riprova degli eventi e l'utilizzo delle code di lettere morte nella Amazon User Guide. EventBridge

> **Note**
>
> Il tipo di AWS::Serverless::Function risorsa ha un tipo di dati simile, `DeadLetterQueue` che gestisce gli errori che si verificano dopo una chiamata riuscita della funzione Lambda di destinazione. Esempi di questo tipo di errore includono la limitazione Lambda o gli errori restituiti dalla funzione di destinazione Lambda. Per ulteriori informazioni sulla *`DeadLetterQueue`* proprietà della funzione, consulta Dead-letter queues nella Developer Guide.AWS Lambda

Sintassi

Per dichiarare questa entità nel modello AWS Serverless Application Model (AWS SAM), utilizzate la seguente sintassi.

YAML

```yaml
Arn: String
QueueLogicalId: String
Type: String
```

Properties

`Arn`

L'Amazon Resource Name (ARN) della coda Amazon SQS specificato come destinazione per la coda di lettere non scritte.

> **Note**
>
> Specificare la proprietà o la `Type` proprietà, ma non entrambe. `Arn`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Arn` proprietà del tipo di `AWS::Events::Rule` `DeadLetterConfig` dati.

`QueueLogicalId`

`Type`Viene specificato il nome personalizzato della coda di lettere morte che AWS SAM crea if.

> **Note**
>
> Se la `Type` proprietà non è impostata, viene ignorata.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Type`

Il tipo di coda. Quando questa proprietà è impostata, crea AWS SAM automaticamente una coda di lettere morte e allega la politica basata sulle risorse necessaria per concedere l'autorizzazione alla risorsa delle regole di inviare eventi alla coda.

> **Note**
>
> Specificate la proprietà o la proprietà, ma non entrambe. `Type` `Arn`

Valori validi: `SQS`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

DeadLetterConfig

DeadLetterConfig

YAML

```
DeadLetterConfig:
  Type: SQS
  QueueLogicalId: MyDLQ
```

ScheduleV2

L'oggetto che descrive un tipo di origine `ScheduleV2` dell'evento, che imposta la funzione serverless come destinazione di un evento Amazon EventBridge Scheduler che si attiva in base a una pianificazione. Per ulteriori informazioni, consulta Cos'è Amazon EventBridge Scheduler? nella Guida per l'utente di EventBridge Scheduler.

AWS Serverless Application Model (AWS SAM) genera una AWS::Scheduler::Schedulerisorsa quando viene impostato questo tipo di evento.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
DeadLetterConfig: DeadLetterConfig
Description: String
EndDate: String
FlexibleTimeWindow: FlexibleTimeWindow
GroupName: String
Input: String
KmsKeyArn: String
Name: String
OmitName: Boolean
PermissionsBoundary: String
RetryPolicy: RetryPolicy
RoleArn: String
ScheduleExpression: String
```

```
ScheduleExpressionTimezone: String
StartDate: String
State: String
```

Properties

`DeadLetterConfig`

Configura la coda Amazon Simple Queue Service (Amazon SQS) a EventBridge cui invia gli eventi dopo una chiamata di destinazione non riuscita. L'invocazione può fallire, ad esempio, quando si invia un evento a una funzione Lambda che non esiste o quando EventBridge le autorizzazioni sono insufficienti per richiamare la funzione Lambda. Per ulteriori informazioni, consulta Configurazione di una coda di lettere non scritte per Scheduler nella Scheduler User Guide. EventBridge EventBridge

> **Note**
>
> Il tipo di AWS::Serverless::Function risorsa ha un tipo di dati simile`DeadLetterQueue`, che gestisce gli errori che si verificano dopo una chiamata riuscita della funzione Lambda di destinazione. Esempi di questi tipi di errori includono la limitazione Lambda o gli errori restituiti dalla funzione di destinazione Lambda. Per ulteriori informazioni sulla *`DeadLetterQueue`* proprietà della funzione, consulta Dead-letter queues nella Developer Guide.AWS Lambda

Tipo: DeadLetterConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `DeadLetterConfig` proprietà del tipo di dati. `AWS::Scheduler::Schedule Target` La AWS SAM versione di questa proprietà include sottoproprietà aggiuntive, nel caso in cui si desideri AWS SAM creare automaticamente la coda di lettere non scritte.

`Description`

Una descrizione della pianificazione.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`EndDate`

La data, in UTC, prima della quale la pianificazione può richiamare il suo obiettivo. A seconda dell'espressione di ricorrenza della pianificazione, le chiamate potrebbero interrompersi nel momento in cui specifichi EndDate o prima.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EndDate` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`FlexibleTimeWindow`

Consente la configurazione di una finestra all'interno della quale è possibile richiamare una pianificazione.

Tipo: FlexibleTimeWindow

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FlexibleTimeWindow` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`GroupName`

Il nome del gruppo di pianificazioni da associare a questa pianificazione. Se non è definito, viene utilizzato il gruppo predefinito.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `GroupName` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`Input`

Testo JSON valido passato al target. Se utilizzi questa proprietà, al target non viene passata alcuna informazione sul testo dell'evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Input` proprietà di una `AWS::Scheduler::Schedule Target` risorsa.

`KmsKeyArn`

L'ARN per una chiave KMS che verrà utilizzata per crittografare i dati dei clienti.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `KmsKeyArn` proprietà di una risorsa. `AWS::Scheduler::Schedule`

`Name`

Il nome della pianificazione. Se non specifichi un nome, AWS SAM genera un nome nel formato *`Function-Logical-IDEvent-Source-Name`* e utilizza quell'ID per il nome della pianificazione.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`OmitName`

Per impostazione predefinita, AWS SAM genera e utilizza un nome di pianificazione nel formato di*`<Function-logical-ID><event-source-name>`*. Imposta questa proprietà in `true` modo da CloudFormation generare un ID fisico univoco e utilizzalo invece per il nome della pianificazione.

Tipo: Booleano

Obbligatorio: no

Default: `false`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`PermissionsBoundary`

L'ARN della policy utilizzata per impostare il limite delle autorizzazioni per il ruolo.

> **Note**
>
> Se `PermissionsBoundary` definito, AWS SAM applicherà gli stessi limiti al ruolo IAM di destinazione della pianificazione dello scheduler.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `PermissionsBoundary` proprietà di una `AWS::IAM::Role` risorsa.

`RetryPolicy`

Un oggetto RetryPolicy che include informazioni sulle impostazioni dei criteri di nuovi tentativi.

Tipo: RetryPolicy

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RetryPolicy` proprietà del tipo di `AWS::Scheduler::Schedule Target` dati.

`RoleArn`

L'ARN del ruolo IAM che EventBridge Scheduler utilizzerà per la destinazione quando viene richiamata la pianificazione.

Tipo: RoleArn

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RoleArn` proprietà del tipo di dati. `AWS::Scheduler::Schedule Target`

`ScheduleExpression`

L'espressione di pianificazione che determina quando e con quale frequenza viene eseguito l'evento di pianificazione dello scheduler.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ScheduleExpression` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`ScheduleExpressionTimezone`

Il fuso orario in cui viene valutata l'espressione di pianificazione.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ScheduleExpressionTimezone` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`StartDate`

La data, in UTC, dopo la quale la pianificazione può iniziare a richiamare un obiettivo. A seconda dell'espressione di ricorrenza della pianificazione, le chiamate potrebbero interrompersi nel momento in cui specifichi StartDate o dopo.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StartDate` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`State`

Lo stato della pianificazione dello Scheduler.

Valori accettati: `DISABLED | ENABLED`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `State` proprietà di una `AWS::Scheduler::Schedule` risorsa.

Esempi

Esempio di base di definizione di una risorsa ScheduleV2

```
Resources:
  Function:
    Properties:
      ...
      Events:
        ScheduleEvent:
          Type: ScheduleV2
          Properties:
            ScheduleExpression: "rate(1 minute)"
        ComplexScheduleEvent:
          Type: ScheduleV2
          Properties:
            ScheduleExpression: rate(1 minute)
            FlexibleTimeWindow:
              Mode: FLEXIBLE
              MaximumWindowInMinutes: 5
            StartDate: '2022-12-28T12:00:00.000Z'
            EndDate: '2023-01-28T12:00:00.000Z'
            ScheduleExpressionTimezone: UTC
            RetryPolicy:
              MaximumRetryAttempts: 5
              MaximumEventAgeInSeconds: 300
            DeadLetterConfig:
              Type: SQS
```

> **Note**
>
> L'ID fisico generato di ScheduleV2 non include il nome dello stack.

SelfManagedKafka

L'oggetto che descrive un tipo di origine `SelfManagedKafka` dell'evento. Per ulteriori informazioni, consulta Utilizzo AWS Lambda con Apache Kafka autogestito nella Guida per gli sviluppatori.AWS Lambda

AWS Serverless Application Model (AWS SAM) genera una AWS::Lambda::EventSourceMappingrisorsa quando viene impostato questo tipo di evento.

Per utilizzare Schema Registry, devi definire autorizzazioni di ruolo IAM specifiche per la tua funzione. Vedi Configurazione completa con ruoli IAM per un esempio della configurazione richiesta.

Sintassi

Per dichiarare questa entità nel AWS SAM modello, utilizza la seguente sintassi.

YAML

```
BatchSize: Integer
BisectBatchOnFunctionError: Boolean
ConsumerGroupId: String
DestinationConfig: DestinationConfig
Enabled: Boolean
FilterCriteria: FilterCriteria
KafkaBootstrapServers: List
FunctionResponseTypes: List
KmsKeyArn: String
LoggingConfig: LoggingConfig
MaximumRecordAgeInSeconds: Integer
MaximumRetryAttempts: Integer
MetricsConfig: MetricsConfig
ProvisionedPollerConfig: ProvisionedPollerConfig
SchemaRegistryConfig: SchemaRegistryConfig
SourceAccessConfigurations: SourceAccessConfigurations
StartingPosition: String
StartingPositionTimestamp: Double
Topics: List
```

Properties

`BatchSize`

Il numero massimo di record in ogni batch che Lambda estrae dal tuo stream e invia alla tua funzione.

Tipo: integer

Obbligatorio: no

Impostazione predefinita: 100

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `BatchSize` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

Minimum: 1

Maximum: 10000

`BisectBatchOnFunctionError`

Se la funzione restituisce un errore, dividi il batch in due e riprova.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `BisectBatchOnFunctionError` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`ConsumerGroupId`

Una stringa che configura il modo in cui gli eventi verranno letti dagli argomenti di Kafka.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SelfManagedKafkaEventSourceConfig` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`DestinationConfig`

Oggetto di configurazione che specifica la destinazione di un evento dopo che Lambda lo elabora.

Utilizzate questa proprietà per specificare la destinazione delle chiamate fallite dalla fonte di eventi Kafka autogestita.

Tipo: DestinationConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di una risorsa. `DestinationConfig` `AWS::Lambda::EventSourceMapping`

`Enabled`

Disabilita la mappatura dell'origine eventi per sospendere il polling e la chiamata.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Enabled` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`FilterCriteria`

Un oggetto che definisce i criteri per determinare se Lambda deve elaborare un evento. Per ulteriori informazioni, consulta il filtro AWS Lambda degli eventi nella Guida per gli AWS Lambda sviluppatori.

Tipo: FilterCriteria

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FilterCriteria` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`KafkaBootstrapServers`

L'elenco dei server bootstrap per i tuoi broker Kafka. Includi la porta, ad esempio `broker.example.com:`*`xxxx`*

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`FunctionResponseTypes`

Un elenco dei tipi di risposta attualmente applicati alla mappatura delle sorgenti degli eventi. Per ulteriori informazioni, consulta la pagina Segnalazione di errori degli elementi batch nella Guida per gli sviluppatori di AWS Lambda .

Valori validi: `ReportBatchItemFailures`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FunctionResponseTypes` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`KmsKeyArn`

L'Amazon Resource Name (ARN) della chiave per crittografare le informazioni relative a questo evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `KmsKeyArn` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`LoggingConfig`

La configurazione di registrazione per l'origine dell'evento.

Tipo: LoggingConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `LoggingConfig` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumRecordAgeInSeconds`

L'età massima di un record che Lambda invia a una funzione per l'elaborazione.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumRecordAgeInSeconds` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MetricsConfig`

La configurazione delle metriche per l'origine del tuo evento.

Tipo: MetricsConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MetricsConfig` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumRetryAttempts`

Il numero massimo di ripetizioni consentite quando la funzione restituisce un errore.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumRetryAttempts` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`ProvisionedPollerConfig`

Configurazione per aumentare la quantità di poller utilizzati per calcolare le mappature delle sorgenti degli eventi. Questa configurazione consente un minimo di 1 poller e un massimo di 2000 poller. Per un esempio, fare riferimento a ProvisionedPollerConfig esempio

Tipo: ProvisionedPollerConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ProvisionedPollerConfig` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`SchemaRegistryConfig`

Configurazione per l'utilizzo di un registro degli schemi con la fonte di eventi Kafka autogestita.

> **Note**
>
> Questa funzionalità deve essere configurata`ProvisionedPollerConfig`.

Tipo: SchemaRegistryConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SelfManagedKafkaEventSourceConfig` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`SourceAccessConfigurations`

Un array del protocollo di autenticazione, dei componenti VPC o dell'host virtuale per proteggere e definire l'origine eventi.

Valori validi: `BASIC_AUTH | CLIENT_CERTIFICATE_TLS_AUTH | SASL_SCRAM_256_AUTH | SASL_SCRAM_512_AUTH | SERVER_ROOT_CA_CERTIFICATE`

Tipo: elenco di SourceAccessConfiguration

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà fa parte della `SelfManagedKafkaEventSourceConfig` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`StartingPosition`

La posizione in un flusso da cui iniziare la lettura.

- `AT_TIMESTAMP`— Specificare l'ora a partire dalla quale iniziare a leggere i record.
- `LATEST`— Leggi solo i nuovi record.
- `TRIM_HORIZON`— Elabora tutti i record disponibili.

Valori validi: `AT_TIMESTAMP` | `LATEST` | `TRIM_HORIZON`

Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StartingPosition` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`StartingPositionTimestamp`

L'ora da cui iniziare a leggere, in secondi di tempo Unix. Definisce `StartingPositionTimestamp` quando `StartingPosition` è specificato come`AT_TIMESTAMP`.

Tipo: numero a doppia precisione

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StartingPositionTimestamp` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`Topics`

Il nome dell'argomento Kafka.

Tipo: elenco

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Topics` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

Esempi

Configurazione completa con ruoli IAM

L'esempio seguente mostra una configurazione completa che include la configurazione del ruolo IAM richiesta per l'utilizzo di Schema Registry:

```
Parameters:
  PreCreatedSubnetOne:
    Type: String
  PreCreatedSubnetTwo:
    Type: String

Resources:
  MyLambdaExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
        - Action: [sts:AssumeRole]
          Effect: Allow
          Principal:
            Service: [lambda.amazonaws.com]
      Policies:
      - PolicyName: KafkaAuthPolicy
        PolicyDocument:
          Statement:
          - Action: [secretsmanager:GetSecretValue, kms:Decrypt]
            Effect: "Allow"
            Resource: ['arn:aws:secretsmanager:us-
west-2:123456789012:secret:kafkaSecret-******',
                        'arn:aws:kms:us-west-2:123456789012:key/keyId']
      - PolicyName: ENIPolicy
        PolicyDocument:
          Statement:
```

```
          - Action: [ec2:CreateNetworkInterface,
              ec2:DescribeNetworkInterfaces, ec2:DescribeVpcs,
 ec2:DeleteNetworkInterface,
              ec2:DescribeSubnets, ec2:DescribeSecurityGroups]
            Effect: Allow
            Resource: '*'
      - PolicyName: SchemaRegistryPolicy
        PolicyDocument:
          Statement:
          - Action: [glue:GetRegistry]
            Effect: Allow
            Resource: 'arn:aws:glue:{region}:{account-id}:registry/{registry-name}'
      - PolicyName: SchemaVersionsPolicy
        PolicyDocument:
          Statement:
          - Action: [glue:GetSchemaVersions]
            Effect: Allow
            Resource: '*'
      ManagedPolicyArns:
      - !Sub arn:${AWS::Partition}:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole
      Tags:
      - {Value: SAM, Key: lambda:createdBy}

  MyKafkaProcessor:
    Type: AWS::Serverless::Function
    Properties:
      Runtime: nodejs18.x
      Handler: index.handler
      CodeUri: ${codeuri}
      Role:
        Fn::GetAtt: [MyLambdaExecutionRole, Arn]
      Events:
        SelfManagedKafkaEvent:
          Type: SelfManagedKafka
          Properties:
            KafkaBootstrapServers:
              - my-kafka-broker-1:9092
              - my-kafka-broker-2:9092
            Topics:
              - SchemaRegistryTestTopic
            StartingPosition: LATEST
            SourceAccessConfigurations:
              - Type: VPC_SUBNET
```

```
                URI: subnet:subnet-12345678
              - Type: VPC_SECURITY_GROUP
                URI: security_group:sg-12345678
              - Type: BASIC_AUTH
                URI: !Sub arn:${AWS::Partition}:secretsmanager:us-
west-2:123456789012:secret:my-path/my-secret-name-1a2b3c
            ProvisionedPollerConfig:
              MinimumPollers: 1
            SchemaRegistryConfig:
              AccessConfigs:
              - Type: BASIC_AUTH
                URI: !Sub arn:${AWS::Partition}:secretsmanager:us-
west-2:123456789012:secret:my-path/my-secret-name-1a2b3c
              SchemaValidationConfigs:
              - Attribute: KEY
              EventRecordFormat: JSON
              SchemaRegistryURI: !Sub arn:${AWS::Partition}:glue:us-
west-2:123456789012:registry/myregistry
```

ProvisionedPollerConfig esempio

```
ProvisionedPollerConfig:
  MinimumPollers: 1
  MaximumPollers: 200
```

Fonte di eventi Kafka autogestita

Di seguito è riportato un esempio di tipo di sorgente di `SelfManagedKafka` eventi.

YAML

```
Events:
  SelfManagedKafkaEvent:
    Type: SelfManagedKafka
    Properties:
      BatchSize: 1000
      Enabled: true
      KafkaBootstrapServers:
        - abc.xyz.com:xxxx
      SourceAccessConfigurations:
        -  Type: BASIC_AUTH
           URI: arn:aws:secretsmanager:us-west-2:123456789012:secret:my-path/my-secret-
name-1a2b3c
```

```
      Topics:
        - MyKafkaTopic
```

Fonte di eventi Kafka gestita autonomamente con Schema Registry AWS Glue

Di seguito è riportato un esempio di un tipo di sorgente di `SelfManagedKafka` eventi configurato con AWS Glue Schema Registry.

```
Events:
  SelfManagedKafkaEvent:
    Type: SelfManagedKafka
    Properties:
      KafkaBootstrapServers:
        - abc.xyz.com:9092
      Topics:
        - SchemaRegistryTestTopic
      StartingPosition: LATEST
      ProvisionedPollerConfig:
        MinimumPollers: 1
      SchemaRegistryConfig:
        SchemaRegistryURI: !Sub arn:${AWS::Partition}:glue:us-
west-2:123456789012:registry/myregistry
        EventRecordFormat: JSON
        SchemaValidationConfigs:
          - Attribute: KEY
          - Attribute: VALUE
      SourceAccessConfigurations:
        - Type: VPC_SUBNET
          URI: subnet:subnet-12345678
        - Type: VPC_SECURITY_GROUP
          URI: security_group:sg-12345678
```

Fonte di eventi Kafka gestita autonomamente con Confluent Schema Registry

Di seguito è riportato un esempio di un tipo di sorgente di `SelfManagedKafka` eventi configurato con Confluent Schema Registry.

```
Events:
  SelfManagedKafkaEvent:
    Type: SelfManagedKafka
    Properties:
      KafkaBootstrapServers:
```

```
        - abc.xyz.com:9092
      Topics:
        - SchemaRegistryTestTopic
      StartingPosition: LATEST
      ProvisionedPollerConfig:
        MinimumPollers: 1
      SchemaRegistryConfig:
        SchemaRegistryURI: https://my-schema-registry.confluent.cloud
        AccessConfigs:
          - Type: BASIC_AUTH
            URI: !Sub arn:${AWS::Partition}:secretsmanager:us-
west-2:123456789012:secret:my-secret
        EventRecordFormat: JSON
        SchemaValidationConfigs:
          - Attribute: KEY
          - Attribute: VALUE
      SourceAccessConfigurations:
        - Type: VPC_SUBNET
          URI: subnet:subnet-12345678
        - Type: VPC_SECURITY_GROUP
          URI: security_group:sg-12345678
        - Type: BASIC_AUTH
          URI: !Sub arn:${AWS::Partition}:secretsmanager:us-
west-2:123456789012:secret:kafka-secret
```

SNS

L'oggetto che descrive un tipo di origine SNS dell'evento.

SAM genera AWS::SNS::Subscriptionuna risorsa quando viene impostato questo tipo di evento

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  FilterPolicy: SnsFilterPolicy
  FilterPolicyScope: String
  RedrivePolicy: Json
  Region: String
  SqsSubscription: Boolean | SqsSubscriptionObject
```

```
  Topic: String
```

Properties

`FilterPolicy`

Il codice JSON della policy di filtro assegnato alla sottoscrizione. Per ulteriori informazioni, consulta GetSubscriptionAttributesAmazon Simple Notification Service API Reference.

Tipo: SnsFilterPolicy

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FilterPolicy` proprietà di una `AWS::SNS::Subscription` risorsa.

`FilterPolicyScope`

Questo attributo consente di scegliere l'ambito di filtraggio utilizzando uno dei seguenti tipi di valori di stringa:

- `MessageAttributes`— Il filtro viene applicato agli attributi del messaggio.
- `MessageBody`— Il filtro viene applicato al corpo del messaggio.

Tipo: stringa

Obbligatorio: no

Default: `MessageAttributes`

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FilterPolicyScope` proprietà di una `AWS::SNS::Subscription` risorsa.

`RedrivePolicy`

Quando è specificato, invia i messaggi non recapitabili alla coda DLQ di Amazon SQS specificata. I messaggi che non possono essere recapitati a causa di errori client (ad esempio, quando l'endpoint sottoscritto non è raggiungibile) o errori del server (ad esempio, quando il servizio che alimenta l'endpoint sottoscritto diventa non disponibile) vengono mantenuti nella coda DLQ per ulteriori analisi o elaborazione.

Per ulteriori informazioni sulla politica di redrive e sulle code di lettera morta di Amazon SQS, consulta la pagina Amazon SQS Dead-letter queues nella Amazon Simple Queue Service Developer Guide.

Type: Json

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di una risorsa. `RedrivePolicy` `AWS::SNS::Subscription`

`Region`

Per le sottoscrizioni in più Regioni, la Regione in cui si trova l'argomento.

Se non viene specificata alcuna regione, CloudFormation utilizza la regione del chiamante come impostazione predefinita.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Region` proprietà di una `AWS::SNS::Subscription` risorsa.

`SqsSubscription`

Imposta questa proprietà su true o specifica di `SqsSubscriptionObject` abilitare l'invio in batch delle notifiche degli argomenti SNS in una coda SQS. L'impostazione di questa proprietà per `true` creare una nuova coda SQS, mentre se si specifica a si utilizza una coda SQS esistente. `SqsSubscriptionObject`

Tipo: Boolean | SqsSubscriptionObject

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Topic`

L'ARN dell'argomento a cui effettuare la sottoscrizione

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `TopicArn` proprietà di una `AWS::SNS::Subscription` risorsa.

Esempi

Esempio di origine degli eventi SNS

Esempio di origine di eventi SNS

YAML

```
Events:
  SNSEvent:
    Type: SNS
    Properties:
      Topic: arn:aws:sns:us-east-1:123456789012:my_topic
      SqsSubscription: true
      FilterPolicy:
        store:
          - example_corp
        price_usd:
          - numeric:
              - ">="
              - 100
```

SqsSubscriptionObject

Specificate un'opzione di coda SQS esistente per l'evento SNS

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  BatchSize: String
  Enabled: Boolean
  QueueArn: String
  QueuePolicyLogicalId: String
  QueueUrl: String
```

Properties

`BatchSize`

Il numero massimo di elementi da recuperare in un singolo batch per la coda SQS.

- Tipo: stringa

Obbligatorio: no

Impostazione predefinita: 10

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`Enabled`

Disattiva la mappatura delle sorgenti degli eventi SQS per sospendere il polling e la chiamata.

Tipo: Booleano

Obbligatorio: no

Impostazione predefinita: True

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`QueueArn`

Specificare un arn di coda SQS esistente.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`QueuePolicyLogicalId`

Assegna un nome LogicalID personalizzato per la AWS::SQS::QueuePolicyrisorsa.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`QueueUrl`

Specificare l'URL della coda associato alla `QueueArn` proprietà.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

Evento SQS for SNS esistente

Esempio di aggiunta di una coda SQS esistente per l'iscrizione a un argomento SNS.

YAML

```
QueuePolicyLogicalId: CustomQueuePolicyLogicalId
QueueArn:
  Fn::GetAtt: MyCustomQueue.Arn
QueueUrl:
  Ref: MyCustomQueue
BatchSize: 5
```

SQS

L'oggetto che descrive un tipo di origine `SQS` dell'evento. Per ulteriori informazioni, consulta Using AWS Lambda with Amazon SQS nella AWS Lambda Developer Guide.

SAM genera AWS::Lambda::EventSourceMappingrisorse quando viene impostato questo tipo di evento

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
  BatchSize: Integer
  Enabled: Boolean
  FilterCriteria: FilterCriteria
  FunctionResponseTypes: List
  KmsKeyArn: String
  MaximumBatchingWindowInSeconds: Integer
  MetricsConfig: MetricsConfig
  ProvisionedPollerConfig: ProvisionedPollerConfig
  Queue: String
  ScalingConfig: ScalingConfig
```

Properties

`BatchSize`

Il numero massimo di elementi da recuperare in un singolo batch.

Tipo: integer

Obbligatorio: no

Impostazione predefinita: 10

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `BatchSize` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

Minimum: 1

Maximum: `10000`

`Enabled`

Disabilita la mappatura dell'origine eventi per sospendere il polling e la chiamata.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Enabled` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`FilterCriteria`

Un oggetto che definisce i criteri per determinare se Lambda deve elaborare un evento. Per ulteriori informazioni, consulta il filtro AWS Lambda degli eventi nella Guida per gli AWS Lambda sviluppatori.

Tipo: FilterCriteria

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FilterCriteria` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`FunctionResponseTypes`

Un elenco dei tipi di risposta attualmente applicati alla mappatura delle sorgenti degli eventi. Per ulteriori informazioni, consulta Segnalazione degli errori relativi agli elementi in batch nella Guida per gli AWS Lambda sviluppatori.

Valori validi: `ReportBatchItemFailures`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FunctionResponseTypes` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`KmsKeyArn`

L'Amazon Resource Name (ARN) della chiave per crittografare le informazioni relative a questo evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `KmsKeyArn` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MaximumBatchingWindowInSeconds`

Il tempo massimo, in secondi, per raccogliere i record prima di richiamare la funzione.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaximumBatchingWindowInSeconds` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`MetricsConfig`

Una configurazione opzionale per ottenere metriche avanzate per le mappature delle sorgenti degli eventi che catturano ogni fase dell'elaborazione. Per vedere un esempio, consulta MetricsConfig evento.

Tipo: MetricsConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MetricsConfig` proprietà di una risorsa. `AWS::Lambda::EventSourceMapping`

`ProvisionedPollerConfig`

Configurazione per aumentare la quantità di poller utilizzati per calcolare le mappature delle sorgenti degli eventi. Questa configurazione consente un minimo di 2 poller e un massimo di 2000 poller. Per un esempio, fare riferimento a. ProvisionedPollerConfig esempio

Tipo: ProvisionedPollerConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ProvisionedPollerConfig` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`Queue`

L'ARN della coda.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventSourceArn` proprietà di una `AWS::Lambda::EventSourceMapping` risorsa.

`ScalingConfig`

Configurazione scalabile dei poller SQS per controllare la frequenza di richiamo e impostare il numero massimo di richiami simultanei.

Tipo: `ScalingConfig`

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di una risorsa. `ScalingConfig` `AWS::Lambda::EventSourceMapping`

Esempi

MetricsConfig evento

Di seguito è riportato un esempio di risorsa che utilizza la `MetricsConfig` proprietà per acquisire ogni fase dell'elaborazione per le mappature delle sorgenti degli eventi.

```
Resources:
  FilteredEventsFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: s3://sam-demo-bucket/metricsConfig.zip
      Handler: index.handler
      Runtime: nodejs16.x
      Events:
        KinesisStream:
          Type: Kinesis
          Properties:
            Stream: !GetAtt KinesisStream.Arn
            StartingPosition: LATEST
            MetricsConfig:
              Metrics:
              - EventCount
```

Evento SQS di base

```
Events:
  SQSEvent:
    Type: SQS
```

```
    Properties:
      Queue: arn:aws:sqs:us-west-2:012345678901:my-queue
      BatchSize: 10
      Enabled: false
      FilterCriteria:
        Filters:
          - Pattern: '{"key": ["val1", "val2"]}'
```

Configura il reporting parziale in batch per la tua coda SQS

```
Events:
  SQSEvent:
    Type: SQS
    Properties:
      Enabled: true
      FunctionResponseTypes:
        - ReportBatchItemFailures
      Queue: !GetAtt MySqsQueue.Arn
      BatchSize: 10
```

Funzione Lambda con un evento SQS con scalabilità configurata

```
MyFunction:
  Type: AWS::Serverless::Function
  Properties:
    ...
    Events:
      MySQSEvent:
        Type: SQS
        Properties:
          ...
          ScalingConfig:
            MaximumConcurrency: 10
```

ProvisionedPollerConfig esempio

```
MyFunction:
  Type: AWS::Serverless::Function
  Properties:
    Handler: index.handler
    Runtime: nodejs18.x
    Timeout: 30
```

```
    Events:
      SQSEvent:
        Type: SQS
        Properties:
          Queue: !GetAtt MyQueue.Arn
          BatchSize: 10
          Enabled: True
          ProvisionedPollerConfig:
            MaximumPollers: 300
            MinimumPollers: 10
```

FunctionCode

Il pacchetto di distribuzione per una funzione Lambda.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Bucket: String
Key: String
Version: String
```

Properties

`Bucket`

Un bucket Amazon S3 nella stessa AWS regione della tua funzione.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `S3Bucket` proprietà del tipo di `AWS::Lambda::Function Code` dati.

`Key`

La chiave Amazon S3 del pacchetto di implementazione.

- Tipo: stringa

  Obbligatorio: sì

  CloudFormation compatibilità: questa proprietà viene passata direttamente alla `S3Key` proprietà del tipo di `AWS::Lambda::Function` `Code` dati.

`Version`

Per gli oggetti con versione, la versione dell'oggetto del pacchetto di implementazione da utilizzare.

- Tipo: stringa

  Obbligatorio: no

  CloudFormation compatibilità: questa proprietà viene passata direttamente alla `S3ObjectVersion` proprietà del tipo di `AWS::Lambda::Function` `Code` dati.

Esempi

FunctionCode

`CodeUri`: Esempio di codice funzionale

YAML

```
CodeUri:
  Bucket: sam-s3-demo-bucket-name
  Key: mykey-name
  Version: 121212
```

FunctionUrlConfig

Crea un URL di AWS Lambda funzione con i parametri di configurazione specificati. L'URL di una funzione Lambda è un endpoint HTTPS che puoi usare per richiamare la tua funzione.

Per impostazione predefinita, l'URL della funzione che crei utilizza la `$LATEST` versione della tua funzione Lambda. Se si specifica un `AutoPublishAlias` per la funzione Lambda, l'endpoint si connette all'alias della funzione specificato.

Per ulteriori informazioni, consulta la funzione Lambda URLs nella AWS Lambda Developer Guide.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
AuthType: String
Cors: Cors
InvokeMode: String
```

Properties

`AuthType`

Il tipo di autorizzazione per l'URL della funzione. Per utilizzare AWS Identity and Access Management (IAM) per autorizzare le richieste, imposta su. `AWS_IAM` Per l'accesso aperto, imposta su. `NONE`

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `AuthType` proprietà di una `AWS::Lambda::Url` risorsa.

`Cors`

Le impostazioni CORS (cross-origin resource sharing) per l'URL della funzione.

Type: Cors

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Cors` proprietà di una `AWS::Lambda::Url` risorsa.

`InvokeMode`

La modalità in cui verrà richiamato l'URL della funzione. Per fare in modo che la funzione restituisca la risposta al termine dell'invocazione, imposta su. `BUFFERED` Per fare in modo che la

funzione trasmetta in streaming la risposta, imposta su. `RESPONSE_STREAM` Il valore predefinito è `BUFFERED`.

Valori validi: `BUFFERED` o `RESPONSE_STREAM`

- Tipo: stringa

Obbligatorio: no

AWS CloudFormation compatibilità: questa proprietà viene passata direttamente alla `InvokeMode`proprietà di una `AWS::Lambda::Url` risorsa.

Esempi

URL della funzione

L'esempio seguente crea una funzione Lambda con un URL di funzione. L'URL della funzione utilizza l'autorizzazione IAM.

YAML

```
HelloWorldFunction:
  Type: AWS::Serverless::Function
  Properties:
    CodeUri: hello_world/
    Handler: index.handler
    Runtime: nodejs20.x
    FunctionUrlConfig:
      AuthType: AWS_IAM
      InvokeMode: RESPONSE_STREAM

Outputs:
  MyFunctionUrlEndpoint:
      Description: "My Lambda Function URL Endpoint"
      Value:
        Fn::GetAtt: HelloWorldFunctionUrl.FunctionUrl
```

CapacityProviderConfig

Configura il provider di capacità a cui verranno allegate le versioni pubblicate della funzione. Ciò consente l'esecuzione della funzione su EC2 istanze di proprietà del cliente gestite da Lambda.

> **Note**
>
> Questa configurazione determina il tipo di calcolo di una funzione e deve essere specificata durante la prima implementazione della funzione. Non può essere aggiunta o rimossa dopo la creazione della risorsa funzionale.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
Arn: String
ExecutionEnvironmentMemoryGiBPerVCpu: Float
PerExecutionEnvironmentMaxConcurrency: Integer
```

Properties

`Arn`

L'ARN del fornitore di capacità da utilizzare per questa funzione.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica per SAM.

`ExecutionEnvironmentMemoryGiBPerVCpu`

Il rapporto tra memoria (in GiB) e vCPU per ogni ambiente di esecuzione.

> **Note**
>
> Il rapporto di memoria per CPU non può superare la memoria totale della funzione di 2048 MB. I memory-to-CPU rapporti supportati sono 2 GB, 4 GB o 8 GB per CPU.

Tipo: Float

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ExecutionEnvironmentMemoryGiBPerVCpu` proprietà di una `AWS::Lambda::Function` risorsa.

`PerExecutionEnvironmentMaxConcurrency`

Il numero massimo di esecuzioni simultanee per ambiente di esecuzione (sandbox).

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `PerExecutionEnvironmentMaxConcurrency` proprietà di una risorsa. `AWS::Lambda::Function`

Esempi

configurazione del provider di capacità

L'esempio seguente mostra una configurazione di un provider di capacità che fa riferimento a una risorsa del provider di capacità.

```
CapacityProviderConfig:
  Arn: !GetAtt MyCapacityProvider.Arn
  ExecutionEnvironmentMemoryGiBPerVCpu: 4.0
  PerExecutionEnvironmentMaxConcurrency: 100
```

FunctionScalingConfig

Configura il comportamento di scalabilità per le versioni della funzione Lambda, controllando il numero di ambienti di esecuzione (sandbox) che possono essere creati. Questa configurazione si applica sia alla versione $LATEST.PUBLISHED che alla versione della funzione numerica.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
MinExecutionEnvironments: Integer
MaxExecutionEnvironments: Integer
```

Properties

`MinExecutionEnvironments`

Il numero minimo di ambienti di esecuzione da mantenere per la versione della funzione.

Tipo: integer

Obbligatorio: no

Default: `3`

Minimum: `0`

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MinExecutionEnvironments` proprietà di una `AWS::Lambda::Function` risorsa.

`MaxExecutionEnvironments`

Il numero massimo di ambienti di esecuzione che possono essere creati per la versione della funzione.

Tipo: integer

Obbligatorio: no

Default: `3`

Minimum: `0`

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaxExecutionEnvironments` proprietà di una `AWS::Lambda::Function` risorsa.

Esempi

Configurazione del ridimensionamento delle funzioni

L'esempio seguente mostra una configurazione di ridimensionamento delle funzioni con ambienti di esecuzione minimi e massimi.

```
FunctionScalingConfig:
  MinExecutionEnvironments: 5
  MaxExecutionEnvironments: 100
```

AWS::Serverless::GraphQLApi

Utilizzate il tipo di `AWS::Serverless::GraphQLApi` risorsa AWS Serverless Application Model (AWS SAM) per creare e configurare un' AWS AppSync GraphQLAPI per la vostra applicazione serverless.

Per ulteriori informazioni AWS AppSync, consulta What is AWS AppSync? nella Guida per gli AWS AppSync sviluppatori.

Sintassi

YAML

```
LogicalId:
  Type: AWS::Serverless::GraphQLApi
  Properties:
    ApiKeys: ApiKeys
    Auth: Auth
    Cache: AWS::AppSync::ApiCache
    DataSources: DataSource
    DomainName: AWS::AppSync::DomainName
    Functions: Function
    Logging: LogConfig
    Name: String
    Resolvers: Resolver
    SchemaInline: String
    SchemaUri: String
    Tags:
    - Tag
    XrayEnabled: Boolean
```

Properties

`ApiKeys`

Crea una chiave univoca che può essere utilizzata per eseguire GraphQL operazioni che richiedono una chiave API.

Tipo: ApiKeys

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Auth`

Configura l'autenticazione per la tua GraphQL API.

Tipo: autenticazione

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Cache`

L'input di un'`CreateApiCache`operazione.

Tipo: AWS::AppSync::ApiCache

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla AWS::AppSync::ApiCacherisorsa.

`DataSources`

Crea sorgenti di dati per le funzioni AWS AppSync a cui connettersi. AWS SAM supporta Amazon DynamoDB e fonti di dati. AWS Lambda

Tipo: DataSource

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`DomainName`

Nome di dominio personalizzato per la tua GraphQL API.

Tipo: AWS::AppSync::DomainName

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla AWS::AppSync::DomainNamerisorsa. AWS SAM genera automaticamente la AWS::AppSync::DomainNameApiAssociationrisorsa.

`Functions`

Configura le funzioni in GraphQL APIs per eseguire determinate operazioni.

Tipo: funzione

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Logging`

Configura Amazon CloudWatch logging per la tua GraphQL API.

Se non specifichi questa proprietà, AWS SAM genererà `CloudWatchLogsRoleArn` e imposterà i seguenti valori:

- `ExcludeVerboseContent: true`
- `FieldLogLevel: ALL`

Per disattivare la registrazione, specificate quanto segue:

```
Logging: false
```

Tipo: LogConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `LogConfig` proprietà di una `AWS::AppSync::GraphQLApi` risorsa.

`LogicalId`

Il nome univoco della tua GraphQL API.

Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::AppSync::GraphQLApi` risorsa.

`Name`

Il nome della tua GraphQL API. Specificate questa proprietà per sovrascrivere il `LogicalId` valore.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::AppSync::GraphQLApi` risorsa.

`Resolvers`

Configura i resolver per i campi della tua API. GraphQL AWS SAM supporta i resolver JavaScript per pipeline.

Tipo: Resolver

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`SchemaInline`

La rappresentazione testuale di uno GraphQL schema in SDL formato.

- Tipo: stringa

Obbligatorio: condizionale. È necessario specificare `SchemaInline` o`SchemaUri`.

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Definition` proprietà di una `AWS::AppSync::GraphQLSchema` risorsa.

`SchemaUri`

L'URI del bucket Amazon Simple Storage Service (Amazon S3) S3 dello schema o il percorso di una cartella locale.

Se specifichi un percorso verso una cartella locale, è CloudFormation necessario che il file venga prima caricato su Amazon S3 prima della distribuzione. Puoi usare il AWS SAMCLI per facilitare

questo processo. Per ulteriori informazioni, consulta Come AWS SAM vengono caricati i file locali al momento della distribuzione.

- Tipo: stringa

Obbligatorio: condizionale. È necessario specificare `SchemaInline` o `SchemaUri`.

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `DefinitionS3Location` proprietà di una `AWS::AppSync::GraphQLSchema` risorsa.

`Tags`

Tag (coppie chiave-valore) per questa GraphQL API. Usa i tag per identificare e classificare le risorse.

Tipo: elenco di Tag

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Tag` proprietà di una `AWS::AppSync::GraphQLApi` risorsa.

`XrayEnabled`

Indicare se utilizzare il tracciamento AWS X-Ray per questa risorsa.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `XrayEnabled` proprietà di una `AWS::AppSync::GraphQLApi` risorsa.

Valori restituiti

Per un elenco dei valori restituiti, consulta la Guida per AWS::Serverless::GraphQLApil'CloudFormation utente.

Esempi

GraphQL APIcon origine dati DynamoDB

In questo esempio, creiamo un'GraphQLAPI che utilizza una tabella DynamoDB come origine dati.

schema.graphql

```
schema {
  query: Query
  mutation: Mutation
}

type Query {
  getPost(id: String!): Post
}

type Mutation {
  addPost(author: String!, title: String!, content: String!): Post!
}

type Post {
  id: String!
  author: String
  title: String
  content: String
  ups: Int!
  downs: Int!
  version: Int!
}
```

modello.yaml

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  DynamoDBPostsTable:
    Type: AWS::Serverless::SimpleTable

  MyGraphQLAPI:
    Type: AWS::Serverless::GraphQLApi
    Properties:
      SchemaUri: ./sam_graphql_api/schema.graphql
      Auth:
        Type: AWS_IAM
      DataSources:
        DynamoDb:
          PostsDataSource:
            TableName: !Ref DynamoDBPostsTable
            TableArn: !GetAtt DynamoDBPostsTable.Arn
```

```
      Functions:
        preprocessPostItem:
          Runtime:
            Name: APPSYNC_JS
            Version: 1.0.0
          DataSource: NONE
          CodeUri: ./sam_graphql_api/preprocessPostItem.js
        createPostItem:
          Runtime:
            Name: APPSYNC_JS
            Version: "1.0.0"
          DataSource: PostsDataSource
          CodeUri: ./sam_graphql_api/createPostItem.js
        getPostFromTable:
          Runtime:
            Name: APPSYNC_JS
            Version: "1.0.0"
          DataSource: PostsDataSource
          CodeUri: ./sam_graphql_api/getPostFromTable.js
      Resolvers:
        Mutation:
          addPost:
            Runtime:
              Name: APPSYNC_JS
              Version: "1.0.0"
            Pipeline:
            - preprocessPostItem
            - createPostItem
        Query:
          getPost:
            CodeUri: ./sam_graphql_api/getPost.js
            Runtime:
              Name: APPSYNC_JS
              Version: "1.0.0"
            Pipeline:
            - getPostFromTable
```

createPostItem.js

```
import { util } from "@aws-appsync/utils";

export function request(ctx) {
  const { key, values } = ctx.prev.result;
```

```
  return {
    operation: "PutItem",
    key: util.dynamodb.toMapValues(key),
    attributeValues: util.dynamodb.toMapValues(values),
  };
}

export function response(ctx) {
  return ctx.result;
}
```

getPostFromTable.js

```
import { util } from "@aws-appsync/utils";

export function request(ctx) {
  return dynamoDBGetItemRequest({ id: ctx.args.id });
}

export function response(ctx) {
  return ctx.result;
}

/**
 * A helper function to get a DynamoDB item
 */
function dynamoDBGetItemRequest(key) {
  return {
    operation: "GetItem",
    key: util.dynamodb.toMapValues(key),
  };
}
```

preprocessPostItem.js

```
import { util } from "@aws-appsync/utils";

export function request(ctx) {
  const id = util.autoId();
  const { ...values } = ctx.args;
  values.ups = 1;
  values.downs = 0;
  values.version = 1;
```

```
  return { payload: { key: { id }, values: values } };
}

export function response(ctx) {
  return ctx.result;
}
```

Ecco il nostro codice resolver:

getPost.js

```
export function request(ctx) {
  return {};
}

export function response(ctx) {
  return ctx.prev.result;
}
```

GraphQLAPI con una funzione Lambda come origine dati

In questo esempio, creiamo un'GraphQLAPI che utilizza una funzione Lambda come origine dati.

template.yaml

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyLambdaFunction:
    Type: AWS::Serverless::Function
    Properties:
      Handler: index.handler
      Runtime: nodejs20.x
      CodeUri: ./lambda

  MyGraphQLAPI:
    Type: AWS::Serverless::GraphQLApi
    Properties:
      Name: MyApi
      SchemaUri: ./gql/schema.gql
      Auth:
        Type: API_KEY
```

```
      ApiKeys:
        MyApiKey:
          Description: my api key
      DataSources:
        Lambda:
          MyLambdaDataSource:
            FunctionArn: !GetAtt MyLambdaFunction.Arn
      Functions:
        lambdaInvoker:
          Runtime:
            Name: APPSYNC_JS
            Version: 1.0.0
          DataSource: MyLambdaDataSource
          CodeUri: ./gql/invoker.js
      Resolvers:
        Mutation:
          addPost:
            Runtime:
              Name: APPSYNC_JS
              Version: 1.0.0
            Pipeline:
            - lambdaInvoker
        Query:
          getPost:
            Runtime:
              Name: APPSYNC_JS
              Version: 1.0.0
            Pipeline:
            - lambdaInvoker

Outputs:
  MyGraphQLAPI:
    Description: AppSync API
    Value: !GetAtt MyGraphQLAPI.GraphQLUrl
  MyGraphQLAPIMyApiKey:
    Description: API Key for authentication
    Value: !GetAtt MyGraphQLAPIMyApiKey.ApiKey
```

schema.graphql

```
schema {
  query: Query
  mutation: Mutation
```

```
}
type Query {
  getPost(id: ID!): Post
}
type Mutation {
  addPost(id: ID!, author: String!, title: String, content: String): Post!
}
type Post {
  id: ID!
  author: String!
  title: String
  content: String
  ups: Int
  downs: Int
}
```

Ecco le nostre funzioni:

lambda/index.js

```
exports.handler = async (event) => {
  console.log("Received event {}", JSON.stringify(event, 3));

  const posts = {
    1: {
      id: "1",
      title: "First book",
      author: "Author1",
      content: "Book 1 has this content",
      ups: "100",
      downs: "10",
    },
  };

  console.log("Got an Invoke Request.");
  let result;
  switch (event.field) {
    case "getPost":
      return posts[event.arguments.id];
    case "addPost":
      // return the arguments back
      return event.arguments;
    default:
      throw new Error("Unknown field, unable to resolve " + event.field);
```

```
  }
};
```

invoker.js

```
import { util } from "@aws-appsync/utils";

export function request(ctx) {
  const { source, args } = ctx;
  return {
    operation: "Invoke",
    payload: { field: ctx.info.fieldName, arguments: args, source },
  };
}

export function response(ctx) {
  return ctx.result;
}
```

ApiKeys

Crea una chiave univoca che può essere utilizzata per eseguire GraphQL operazioni che richiedono una chiave API.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
LogicalId:
  ApiKeyId: String
  Description: String
  ExpiresOn: Double
```

Properties

`ApiKeyId`

Il nome univoco della tua chiave API. Specificare di sovrascrivere il `LogicalId` valore.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ApiKeyId` proprietà di una `AWS::AppSync::ApiKey` risorsa.

`Description`

Descrizione della tua chiave API.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una `AWS::AppSync::ApiKey` risorsa.

`ExpiresOn`

Tempo dopo il quale scade la chiave API. La data è rappresentata come secondi dall'epoca, arrotondati per difetto all'ora più vicina.

Tipo: numero a doppia precisione

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Expires` proprietà di una `AWS::AppSync::ApiKey` risorsa.

`LogicalId`

Il nome univoco della tua chiave API.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ApiKeyId` proprietà di una `AWS::AppSync::ApiKey` risorsa.

Auth

Configura l'autorizzazione per la tua GraphQL API.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Additional:
- AuthProvider
LambdaAuthorizer: LambdaAuthorizerConfig
OpenIDConnect: OpenIDConnectConfig
Type: String
UserPool: UserPoolConfig
```

Properties

`Additional`

Un elenco di tipi di autorizzazione aggiuntivi per la tua GraphQL API.

Tipo: Elenco di AuthProvider

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`LambdaAuthorizer`

Specificate la configurazione di autorizzazione opzionale per l'autorizzatore della funzione Lambda. È possibile configurare questa proprietà opzionale quando `Type` è specificata come. `AWS_LAMBDA`

Tipo: LambdaAuthorizerConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `LambdaAuthorizerConfig` proprietà di una `AWS::AppSync::GraphQLApi` risorsa.

`OpenIDConnect`

Specificate la configurazione di autorizzazione opzionale per il OpenID Connect servizio conforme. È possibile configurare questa proprietà opzionale quando `Type` è specificata come. `OPENID_CONNECT`

Tipo: Open IDConnect Config

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `OpenIDConnectConfig` proprietà di una `AWS::AppSync::GraphQLApi` risorsa.

`Type`

Il tipo di autorizzazione predefinito tra le applicazioni e l' AWS AppSync GraphQLAPI.

Per un elenco e una descrizione dei valori consentiti, consulta Autorizzazione e autenticazione nella Guida per gli AWS AppSync sviluppatori.

Quando specifichi un Lambda authorizer (`AWS_LAMBDA`), AWS SAM crea una policy AWS Identity and Access Management (IAM) per fornire le autorizzazioni tra l'API GraphQL e la funzione Lambda.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `AuthenticationType` proprietà di una risorsa. `AWS::AppSync::GraphQLApi`

`UserPool`

Specificare la configurazione di autorizzazione opzionale per l'utilizzo dei pool di utenti di Amazon Cognito. Puoi configurare questa proprietà opzionale quando `Type` è specificata come`AMAZON_COGNITO_USER_POOLS`.

Tipo: UserPoolConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `UserPoolConfig` proprietà di una `AWS::AppSync::GraphQLApi` risorsa.

Esempi

Configura un tipo di autorizzazione predefinito e aggiuntivo

In questo esempio, iniziamo configurando un autorizzatore Lambda come tipo di autorizzazione predefinito per la nostra API. GraphQL

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyGraphQLAPI:
    Type: AWS::Serverless::GraphQLApi
    Properties:
      Auth:
        Type: AWS_LAMBDA
        LambdaAuthorizer:
          AuthorizerUri: !GetAtt Authorizer1.Arn
          AuthorizerResultTtlInSeconds: 10
          IdentityValidationExpression: hello
```

Successivamente, configuriamo tipi di autorizzazione aggiuntivi per la nostra GraphQL API aggiungendo quanto segue al nostro modello: AWS SAM

```
        Additional:
        - Type: AWS_IAM
        - Type: API_KEY
        - Type: OPENID_CONNECT
          OpenIDConnect:
            AuthTTL: 10
            ClientId: myId
            IatTTL: 10
            Issuer: prod
```

Ciò si traduce nel seguente AWS SAM modello:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyGraphQLAPI:
    Type: AWS::Serverless::GraphQLApi
    Properties:
      Auth:
        Type: AWS_LAMBDA
        LambdaAuthorizer:
          AuthorizerUri: !GetAtt Authorizer1.Arn
          AuthorizerResultTtlInSeconds: 10
          IdentityValidationExpression: hello
```

```
        Additional:
        - Type: AWS_IAM
        - Type: API_KEY
        - Type: OPENID_CONNECT
          OpenIDConnect:
            AuthTTL: 10
            ClientId: myId
            IatTTL: 10
            Issuer: prod
```

AuthProvider

Configurazione di autorizzazione opzionale per i tipi di autorizzazione GraphQL API aggiuntivi.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
LambdaAuthorizer: LambdaAuthorizerConfig
OpenIDConnect: OpenIDConnectConfig
Type: String
UserPool: UserPoolConfig
```

Properties

`LambdaAuthorizer`

Specificate la configurazione di autorizzazione opzionale per l'autorizzatore della AWS Lambda funzione. È possibile configurare questa proprietà opzionale quando `Type` è specificata come`AWS_LAMBDA`.

Tipo: LambdaAuthorizerConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `LambdaAuthorizerConfig` proprietà di un `AWS::AppSync::GraphQLApi` `AdditionalAuthenticationProvider` oggetto.

`OpenIDConnect`

Specificate la configurazione di autorizzazione opzionale per il OpenID Connect servizio conforme. È possibile configurare questa proprietà opzionale quando `Type` è specificata come. `OPENID_CONNECT`

Tipo: Open IDConnect Config

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `OpenIDConnectConfig` proprietà di un `AWS::AppSync::GraphQLApi` `AdditionalAuthenticationProvider` oggetto.

`Type`

Il tipo di autorizzazione predefinito tra le applicazioni e l' AWS AppSync GraphQLAPI.

Per un elenco e una descrizione dei valori consentiti, consulta Autorizzazione e autenticazione nella Guida per gli AWS AppSync sviluppatori.

Quando specifichi un Lambda authorizer (`AWS_LAMBDA`), AWS SAM crea una policy AWS Identity and Access Management (IAM) per fornire le autorizzazioni tra l'API GraphQL e la funzione Lambda.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di un `AuthenticationType` oggetto. `AWS::AppSync::GraphQLApi` `AdditionalAuthenticationProvider`

`UserPool`

Specificare la configurazione di autorizzazione opzionale per l'utilizzo dei pool di utenti di Amazon Cognito. Puoi configurare questa proprietà opzionale quando `Type` è specificata come`AMAZON_COGNITO_USER_POOLS`.

Tipo: UserPoolConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `UserPoolConfig` proprietà di un `AWS::AppSync::GraphQLApi` `AdditionalAuthenticationProvider` oggetto.

DataSource

Configura una fonte di dati a cui il tuo resolver GraphQL API può connettersi. Puoi utilizzare i modelli AWS Serverless Application Model (AWS SAM) per configurare le connessioni alle seguenti fonti di dati:

- Amazon DynamoDB
- AWS Lambda

Per ulteriori informazioni sulle fonti di dati, consulta Allegare una fonte di dati nella Guida per gli AWS AppSync sviluppatori.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
DynamoDb: DynamoDb
Lambda: Lambda
```

Properties

`DynamoDb`

Configura una tabella DynamoDB come origine dati per GraphQL il tuo resolver API.

Tipo: DynamoDb

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`Lambda`

Configura una funzione Lambda come origine dati per il tuo resolver GraphQL API.

Type: Lambda

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

DynamoDb

Configura una tabella Amazon DynamoDB come origine dati per GraphQL il tuo resolver API.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
LogicalId:
  DeltaSync: DeltaSyncConfig
  Description: String
  Name: String
  Permissions: List
  Region: String
  ServiceRoleArn: String
  TableArn: String
  TableName: String
  UseCallerCredentials: Boolean
  Versioned: Boolean
```

Properties

`DeltaSync`

Descrive una configurazione Delta Sync.

Tipo: DeltaSyncConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `DeltaSyncConfig` proprietà di un `AWS::AppSync::DataSource DynamoDBConfig` oggetto.

`Description`

La descrizione della tua fonte di dati.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una `AWS::AppSync::DataSource` risorsa.

`LogicalId`

Il nome univoco della tua fonte di dati.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::AppSync::DataSource` risorsa.

`Name`

Il nome della tua fonte di dati. Specificate questa proprietà per sovrascrivere il `LogicalId` valore.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::AppSync::DataSource` risorsa.

`Permissions`

Fornisci le autorizzazioni alla tua fonte di dati utilizzandoAWS SAM connettori. È possibile fornire uno dei seguenti valori in un elenco:

- `Read`— Consenti al resolver di leggere la fonte dei dati.
- `Write`— Consenti al resolver di scrivere sulla tua fonte di dati.

AWS SAM utilizza una `AWS::Serverless::Connector` risorsa che viene trasformata al momento dell'implementazione per fornire le autorizzazioni. Per maggiori informazioni sulle risorse generate, consultaCloudFormation risorse generate quando si specifica AWS::Serverless::Connector.

> **Note**
>
> Puoi specificare `Permissions` o `ServiceRoleArn`, ma non entrambi. Se non viene specificato nessuno dei due, AWS SAM genererà i valori predefiniti di `Read` and`Write`. Per revocare l'accesso all'origine dati, rimuovi l'oggetto DynamoDB dal modello. AWS SAM

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation È simile alla `Permissions` proprietà di una `AWS::Serverless::Connector` risorsa.

`Region`

La tua Regione AWS tabella DynamoDB. Se non lo specifichi, AWS SAM utilizza. `AWS::Region`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `AwsRegion` proprietà di un `AWS::AppSync::DataSource DynamoDBConfig` oggetto.

`ServiceRoleArn`

L'ARN del ruolo di servizio AWS Identity and Access Management (IAM) per l'origine dati. Il sistema assume questo ruolo durante l'accesso all'origine dati.

Puoi specificare `Permissions` o `ServiceRoleArn`, ma non entrambi.

- Tipo: stringa

Obbligatorio: No. Se non specificato, AWS SAM applica il valore predefinito per`Permissions`.

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ServiceRoleArn` proprietà di una `AWS::AppSync::DataSource` risorsa.

`TableArn`

L'ARN per la tabella DynamoDB.

- Tipo: stringa

Obbligatorio: condizionale. Se non lo specifichi`ServiceRoleArn`, `TableArn` è obbligatorio.

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`TableName`

Il nome della tabella.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `TableName` proprietà di un `AWS::AppSync::DataSource DynamoDBConfig` oggetto.

`UseCallerCredentials`

Imposta `true` per utilizzare IAM con questa fonte di dati.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `UseCallerCredentials` proprietà di un `AWS::AppSync::DataSource DynamoDBConfig` oggetto.

`Versioned`

Imposta per `true` utilizzare Conflict Detection, Conflict Resolution e Sync con questa fonte di dati.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Versioned` proprietà di un `AWS::AppSync::DataSource DynamoDBConfig` oggetto.

Lambda

Configura una AWS Lambda funzione come fonte di dati per il tuo resolver GraphQL API.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
LogicalId:
  Description: String
  FunctionArn: String
  Name: String
  ServiceRoleArn: String
```

Properties

`Description`

La descrizione della tua fonte di dati.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una `AWS::AppSync::DataSource` risorsa.

`FunctionArn`

L'ARN per la funzione Lambda.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `LambdaFunctionArn` proprietà di un `AWS::AppSync::DataSource LambdaConfig` oggetto.

`LogicalId`

Il nome univoco della tua fonte di dati.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::AppSync::DataSource` risorsa.

`Name`

Il nome della tua fonte di dati. Specificate questa proprietà per sovrascrivere il `LogicalId` valore.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::AppSync::DataSource` risorsa.

`ServiceRoleArn`

L'ARN del ruolo di servizio AWS Identity and Access Management (IAM) per l'origine dati. Il sistema assume questo ruolo durante l'accesso all'origine dati.

> **Note**
>
> Per revocare l'accesso all'origine dati, rimuovi l'oggetto Lambda dal modello. AWS SAM

- Tipo: stringa

Obbligatorio: No. Se non specificato, AWS SAM fornirà `Write` le autorizzazioni utilizzandoAWS SAM connettori.

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ServiceRoleArn` proprietà di una `AWS::AppSync::DataSource` risorsa.

Funzione

Configura le funzioni in GraphQL APIs per eseguire determinate operazioni.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
LogicalId:
  CodeUri: String
  DataSource: String
  Description: String
  Id: String
  InlineCode: String
  MaxBatchSize: Integer
  Name: String
  Runtime: Runtime
  Sync: SyncConfig
```

Properties

`CodeUri`

L'URI o il percorso della cartella locale di Amazon Simple Storage Service (Amazon S3) del codice funzione.

Se specifichi un percorso verso una cartella locale, è CloudFormation necessario che il file venga prima caricato su Amazon S3 prima della distribuzione. Puoi usare il AWS SAMCLI per facilitare questo processo. Per ulteriori informazioni, consulta Come AWS SAM vengono caricati i file locali al momento della distribuzione.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CodeS3Location` proprietà di una `AWS::AppSync::FunctionConfiguration` risorsa.

`DataSource`

Il nome della fonte di dati a cui verrà collegata questa funzione.

- Per fare riferimento a un'origine dati all'interno della `AWS::Serverless::GraphQLApi` risorsa, specificane l'ID logico.
- Per fare riferimento a una fonte di dati esterna alla `AWS::Serverless::GraphQLApi` risorsa, fornisci il relativo `Name` attributo utilizzando la funzione `Fn::GetAtt` intrinseca. Ad esempio, `! GetAtt MyLambdaDataSource.Name`.
- Per fare riferimento a una fonte di dati da uno stack diverso, usa. `Fn::ImportValue`

Se `[NONE | None | none]` viene specificata una variazione di, AWS SAM genererà un `None` valore per l'`AWS::AppSync::DataSourceType`oggetto.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `DataSourceName` proprietà di una `AWS::AppSync::FunctionConfiguration` risorsa.

`Description`

La descrizione della tua funzione.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una `AWS::AppSync::FunctionConfiguration` risorsa.

`Id`

L'ID della funzione per una funzione situata all'esterno della `AWS::Serverless::GraphQLApi` risorsa.

- Per fare riferimento a una funzione all'interno dello stesso AWS SAM modello, utilizzate la funzione `Fn::GetAtt` intrinseca. Ad esempio, `Id: !GetAtt createPostItemFunc.FunctionId`.
- Per fare riferimento a una funzione da uno stack diverso, usa. `Fn::ImportValue`

Durante l'utilizzo`Id`, tutte le altre proprietà non sono consentite. AWS SAM passerà automaticamente l'ID funzione della funzione di riferimento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`InlineCode`

Il codice della funzione che contiene le funzioni di richiesta e risposta.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Code` proprietà di una `AWS::AppSync::FunctionConfiguration` risorsa.

`LogicalId`

Il nome univoco della tua funzione.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::AppSync::FunctionConfiguration` risorsa.

`MaxBatchSize`

Il numero massimo di input di richiesta del resolver che verranno inviati a una singola funzione AWS Lambda in un'operazione `BatchInvoke`.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla MaxBatchSizeproprietà di una `AWS::AppSync::FunctionConfiguration` risorsa.

`Name`

Il nome della funzione. Specificare di sovrascrivere il `LogicalId` valore.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::AppSync::FunctionConfiguration` risorsa.

`Runtime`

Descrive un runtime utilizzato da un resolver o da una funzione di AWS AppSync pipeline. AWS AppSync Specifica il nome e la versione del runtime da utilizzare.

Tipo: Runtime

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente. È simile alla `Runtime` proprietà di una `AWS::AppSync::FunctionConfiguration` risorsa.

`Sync`

Descrive una configurazione di sincronizzazione per una funzione.

Speciifica la strategia di rilevamento dei conflitti e la strategia di risoluzione da utilizzare quando la funzione viene richiamata.

Tipo: SyncConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SyncConfig` proprietà di una `AWS::AppSync::FunctionConfiguration` risorsa.

Runtime

Il runtime del resolver o della funzione della pipeline. Specificate il nome e la versione da utilizzare.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Name: String
Version: String
```

Properties

`Name`

Il nome del runtime da usare. Attualmente, l'unico valore consentito è `APPSYNC_JS`.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di un `AWS::AppSync::FunctionConfiguration` `AppSyncRuntime` oggetto.

`Version`

La versione del runtime da utilizzare. Attualmente, l'unica versione consentita è `1.0.0`.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RuntimeVersion` proprietà di un `AWS::AppSync::FunctionConfiguration` `AppSyncRuntime` oggetto.

Risolutore

Configura i resolver per i campi della tua API. GraphQL AWS Serverless Application Model (AWS SAM) supporta JavaScript i resolver per pipeline.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
OperationType:
  LogicalId:
    Caching: CachingConfig
    CodeUri: String
    FieldName: String
    InlineCode: String
    MaxBatchSize: Integer
    Pipeline: List
    Runtime: Runtime
    Sync: SyncConfig
```

Properties

`Caching`

La configurazione di memorizzazione nella cache per il resolver che ha attivato la memorizzazione nella cache.

Tipo: CachingConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CachingConfig` proprietà di una risorsa. `AWS::AppSync::Resolver`

`CodeUri`

L'URI o il percorso di una cartella locale del codice della funzione resolver di Amazon Simple Storage Service (Amazon S3).

Se specifichi un percorso verso una cartella locale, è CloudFormation necessario che il file venga prima caricato su Amazon S3 prima della distribuzione. Puoi usare il AWS SAMCLI per facilitare questo processo. Per ulteriori informazioni, consulta Come AWS SAM vengono caricati i file locali al momento della distribuzione.

Se nessuno dei due `CodeUri` `InlineCode` viene fornito, AWS SAM genererà `InlineCode` che reindirizza la richiesta alla prima funzione di pipeline e riceve la risposta dall'ultima funzione di pipeline.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CodeS3Location` proprietà di una risorsa. `AWS::AppSync::Resolver`

`FieldName`

Il nome del tuo resolver. Specificate questa proprietà per sovrascrivere il valore. `LogicalId`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FieldName` proprietà di una `AWS::AppSync::Resolver` risorsa.

`InlineCode`

Il codice del resolver che contiene le funzioni di richiesta e risposta.

Se nessuna delle due `CodeUri` o `InlineCode` viene fornita, AWS SAM genererà un `InlineCode` messaggio che reindirizza la richiesta alla prima funzione di pipeline e riceve la risposta dall'ultima funzione di pipeline.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Code` proprietà di una risorsa. `AWS::AppSync::Resolver`

`LogicalId`

Il nome univoco del tuo resolver. In uno GraphQL schema, il nome del resolver deve corrispondere al nome del campo per cui viene utilizzato. Usa lo stesso nome di campo per. `LogicalId`

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`MaxBatchSize`

Il numero massimo di input di richiesta del resolver che verranno inviati a una singola funzione AWS Lambda in un'operazione `BatchInvoke`.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MaxBatchSize` proprietà di una `AWS::AppSync::Resolver` risorsa.

`OperationType`

Il tipo di GraphQL operazione associato al resolver. Ad esempio, `Query`, `Mutation` o `Subscription`. È possibile annidare più resolver all'interno di un unico resolver. `LogicalId` `OperationType`

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `TypeName` proprietà di una risorsa. `AWS::AppSync::Resolver`

`Pipeline`

Funzioni collegate al risolver della pipeline. Specificate le funzioni in base all'ID logico in un elenco.

Tipo: elenco

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente. È simile alla `PipelineConfig` proprietà di una `AWS::AppSync::Resolver` risorsa.

`Runtime`

Il runtime del resolver o della funzione della pipeline. Specificate il nome e la versione da utilizzare.

Tipo: Runtime

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente. È simile alla `Runtime` proprietà di una `AWS::AppSync::Resolver` risorsa.

`Sync`

Descrive una configurazione di sincronizzazione per un resolver.

Specifica quale strategia di rilevamento dei conflitti e strategia di risoluzione utilizzare quando viene richiamato il resolver.

Tipo: SyncConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SyncConfig` proprietà di una `AWS::AppSync::Resolver` risorsa.

Esempi

Usa il codice della funzione resolver AWS SAM generato e salva i campi come variabili

Ecco GraphQL lo schema per il nostro esempio:

```
schema {
  query: Query
  mutation: Mutation
}

type Query {
  getPost(id: ID!): Post
}

type Mutation {
  addPost(author: String!, title: String!, content: String!): Post!
}

type Post {
  id: ID!
  author: String
  title: String
  content: String
}
```

Ecco un frammento del nostro AWS SAM modello:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyGraphQLApi:
    Type: AWS::Serverless::GraphQLApi
    Properties:
      ...
      Functions:
        preprocessPostItem:
          ...
        createPostItem:
          ...
      Resolvers:
        Mutation:
```

```
          addPost:
            Runtime:
              Name: APPSYNC_JS
              Version: 1.0.0
            Pipeline:
            - preprocessPostItem
            - createPostItem
```

Nel nostro AWS SAM modello, non `CodeUri` specifichiamo o. `InlineCode` Al momento dell'implementazione, genera AWS SAM automaticamente il seguente codice in linea per il nostro resolver:

```
export function request(ctx) {
  return {};
}

export function response(ctx) {
  return ctx.prev.result;
}
```

Questo codice resolver predefinito reindirizza la richiesta alla prima funzione di pipeline e riceve la risposta dall'ultima funzione di pipeline.

Nella nostra prima funzione pipeline, possiamo usare il `args` campo fornito per analizzare l'oggetto della richiesta e creare le nostre variabili. Possiamo quindi utilizzare queste variabili all'interno della nostra funzione. Ecco un esempio della nostra `preprocessPostItem` funzione:

```
import { util } from "@aws-appsync/utils";

export function request(ctx) {
  const author = ctx.args.author;
  const title = ctx.args.title;
  const content = ctx.args.content;

  // Use variables to process data

}

export function response(ctx) {
  return ctx.result;
}
```

Runtime

Il runtime del resolver o della funzione della pipeline. Specificate il nome e la versione da utilizzare.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Name: String
Version: String
```

Properties

`Name`

Il nome del runtime da usare. Attualmente, l'unico valore consentito è `APPSYNC_JS`.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di un `AWS::AppSync::Resolver AppSyncRuntime` oggetto.

`Version`

La versione del runtime da utilizzare. Attualmente, l'unica versione consentita è `1.0.0`.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RuntimeVersion` proprietà di un `AWS::AppSync::Resolver AppSyncRuntime` oggetto.

AWS::Serverless::HttpApi

Crea un'API HTTP Amazon API Gateway, che ti consente di creare RESTful APIs con una latenza inferiore e costi inferiori rispetto a REST APIs. Per ulteriori informazioni, consulta Working with HTTP APIs nella API Gateway Developer Guide.

Ti consigliamo di utilizzare CloudFormation hook o policy IAM per verificare che alle risorse API Gateway siano associate autorizzazioni per controllarne l'accesso.

Per ulteriori informazioni sull'uso degli CloudFormation hook, consulta Registrazione degli hook nella guida per l'utente della CloudFormation CLI e nel repository. apigw-enforce-authorizer GitHub

Per ulteriori informazioni sull'utilizzo delle policy IAM, consulta Require che le route API abbiano l'autorizzazione nella API Gateway Developer Guide.

> **Note**
>
> Quando esegui la distribuzione a AWS CloudFormation, AWS SAM trasforma le tue AWS SAM risorse in CloudFormation risorse. Per ulteriori informazioni, consulta CloudFormation Risorse generate per AWS SAM.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Type: AWS::Serverless::HttpApi
Properties:
  AccessLogSettings: AccessLogSettings
  Auth: HttpApiAuth
  CorsConfiguration: String | HttpApiCorsConfiguration
  DefaultRouteSettings: RouteSettings
  DefinitionBody: JSON
  DefinitionUri: String | HttpApiDefinition
  Description: String
  DisableExecuteApiEndpoint: Boolean
  Domain: HttpApiDomainConfiguration
  FailOnWarnings: Boolean
  Name: String
  PropagateTags: Boolean
  RouteSettings: RouteSettings
  StageName: String
  StageVariables: Json
  Tags: Map
```

Properties

`AccessLogSettings`

Le impostazioni per la registrazione degli accessi in una fase.

Tipo: AccessLogSettings

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `AccessLogSettings` proprietà di una `AWS::ApiGatewayV2::Stage` risorsa.

`Auth`

Configura l'autorizzazione per il controllo dell'accesso all'API HTTP API Gateway.

Per ulteriori informazioni, consulta Controllare l'accesso a HTTP APIs con gli autorizzatori JWT nella API Gateway Developer Guide.

Tipo: HttpApiAuth

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`CorsConfiguration`

Gestisce la condivisione delle risorse tra le origini (CORS) per tutti i tuoi API Gateway HTTP. APIs Specificate il dominio da consentire come stringa o specificate un `HttpApiCorsConfiguration` oggetto. Nota che CORS richiede AWS SAM di modificare la definizione OpenAPI, quindi CORS funziona solo se `DefinitionBody` la proprietà è specificata.

Per ulteriori informazioni, consulta Configurazione di CORS per un'API HTTP nella Guida per sviluppatori di API Gateway.

> **Note**
>
> Se `CorsConfiguration` è impostato sia in una definizione OpenAPI che a livello di proprietà, AWS SAM unisce entrambe le fonti di configurazione con le proprietà che hanno la precedenza. Se questa proprietà è impostata `sutrue`, tutte le origini sono consentite.

Tipo: String | HttpApiCorsConfiguration

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`DefaultRouteSettings`

Le impostazioni di routing predefinite per questa API HTTP. Queste impostazioni si applicano a tutti i percorsi a meno che non vengano sovrascritte dalla `RouteSettings` proprietà per determinati percorsi.

Tipo: RouteSettings

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RouteSettings` proprietà di una `AWS::ApiGatewayV2::Stage` risorsa.

`DefinitionBody`

La definizione OpenAPI che descrive la tua API HTTP. Se non specifichi a `DefinitionUri` o a`DefinitionBody`, AWS SAM genera una `DefinitionBody` per te in base alla configurazione del modello.

Type: JSON

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `Body` proprietà di una `AWS::ApiGatewayV2::Api` risorsa. Se vengono fornite determinate proprietà, AWS SAM può inserire o modificare il contenuto `DefinitionBody` prima che venga passato a CloudFormation. Le proprietà includono `Auth` e un `EventSource` tipo HttpApi per una `AWS::Serverless::Function` risorsa corrispondente.

`DefinitionUri`

L'URI, il percorso del file locale o l'oggetto location di Amazon Simple Storage Service (Amazon S3) Simple Storage Service (Amazon S3) Simple Storage Service (Amazon S3) che definisce l'API HTTP. L'oggetto Amazon S3 a cui fa riferimento questa proprietà deve essere un file di definizione OpenAPI valido. Se non specifichi a `DefinitionUri` o a are `DefinitionBody` sono specificati, AWS SAM genera un messaggio `DefinitionBody` in base alla configurazione del modello.

Se fornite un percorso di file locale, il modello deve passare attraverso il flusso di lavoro che include il `sam package` comando `sam deploy` o affinché la definizione venga trasformata correttamente.

Le funzioni intrinseche non sono supportate nei file di OpenApi definizione esterni a cui si fa riferimento. `DefinitionUri` Per importare una OpenApi definizione nel modello, utilizzate la `DefinitionBody` proprietà con la trasformazione Include.

Tipo: String | HttpApiDefinition

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è simile alla `BodyS3Location` proprietà di una `AWS::ApiGatewayV2::Api` risorsa. Le proprietà annidate di Amazon S3 sono denominate in modo diverso.

`Description`

La descrizione della risorsa API HTTP.

Quando si specifica`Description`, AWS SAM modificherà la OpenApi definizione della risorsa API HTTP impostando il `description` campo. I seguenti scenari genereranno un errore:

- La `DefinitionBody` proprietà viene specificata con il `description` campo impostato nella definizione di Open API: ciò genera un conflitto del `description` campo che AWS SAM non verrà risolto.
- La `DefinitionUri` proprietà è specificata: AWS SAM non modificherà una definizione di Open API recuperata da Amazon S3.

Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`DisableExecuteApiEndpoint`

Speciifica se i client possono richiamare l'API HTTP utilizzando l'endpoint predefinito`execute-api`. `https://{api_id}.execute-api.{region}.amazonaws.com` Per impostazione predefinita, i client possono richiamare l'API con l'endpoint predefinito. Per richiedere che i client utilizzino solo un nome di dominio personalizzato per richiamare l'API, disabilita l'endpoint predefinito.

Per utilizzare questa proprietà, è necessario specificare la `DefinitionBody` proprietà anziché la `DefinitionUri` proprietà o definire `x-amazon-apigateway-endpoint-configuration` con `disableExecuteApiEndpoint` nella definizione OpenAPI.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è simile alla `DisableExecuteApiEndpoint` proprietà di una `AWS::ApiGatewayV2::Api` risorsa. Viene passato direttamente alla `disableExecuteApiEndpoint` proprietà di un' `x-amazon-apigateway-endpoint-configuration`estensione, che viene aggiunta alla `Body` proprietà di una `AWS::ApiGatewayV2::Api` risorsa.

`Domain`

Configura un dominio personalizzato per questa API HTTP API Gateway.

Tipo: HttpApiDomainConfiguration

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`FailOnWarnings`

Speciifica se ripristinare la creazione dell'API HTTP (`true`) o meno (`false`) quando viene rilevato un avviso. Il valore predefinito è `false`.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FailOnWarnings` proprietà di una `AWS::ApiGatewayV2::Api` risorsa.

`Name`

Il nome della risorsa API HTTP.

Quando si specifica`Name`, AWS SAM modificherà la definizione OpenAPI della risorsa API HTTP impostando il `title` campo. I seguenti scenari genereranno un errore:

- La `DefinitionBody` proprietà viene specificata con il `title` campo impostato nella definizione di Open API: ciò genera un conflitto del `title` campo che AWS SAM non verrà risolto.
- La `DefinitionUri` proprietà è specificata: AWS SAM non modificherà una definizione di Open API recuperata da Amazon S3.

Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`PropagateTags`

Indica se passare o meno i tag dalla `Tags` proprietà alle risorse AWS::Serverless::HttpApi generate. `True`Specificate di propagare i tag nelle risorse generate.

Tipo: Booleano

Obbligatorio: no

Default: `False`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`RouteSettings`

Le impostazioni del percorso, per percorso, per questa API HTTP. Per ulteriori informazioni, consulta Working with routes for HTTP APIs nella API Gateway Developer Guide.

Tipo: RouteSettings

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RouteSettings` proprietà di una `AWS::ApiGatewayV2::Stage` risorsa.

`StageName`

Il nome della fase API. Se non viene specificato alcun nome, AWS SAM utilizza lo `$default` stage di API Gateway.

- Tipo: stringa

Obbligatorio: no

Predefinito: $default

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StageName` proprietà di una `AWS::ApiGatewayV2::Stage` risorsa.

`StageVariables`

Mappa che definisce le variabili delle fasi. I nomi delle variabili possono avere caratteri alfanumerici e caratteri di sottolineatura. I valori devono corrispondere a [a-zA-Z0-9-._~:/? #&=,] +.

Type: Json

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StageVariables` proprietà di una `AWS::ApiGatewayV2::Stage` risorsa.

`Tags`

Una mappa (da stringa a stringa) che specifica i tag da aggiungere a questa fase di API Gateway. Le chiavi possono contenere da 1 a 128 caratteri Unicode e non possono includere il prefisso. `aws:` Puoi utilizzare uno qualsiasi dei seguenti caratteri: l'insieme delle lettere Unicode, cifre, spazi, _, `.`, `/`, =, + e `-`. I valori possono avere una lunghezza compresa tra 1 e 256 caratteri Unicode.

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Note aggiuntive: La `Tags` proprietà richiede di AWS SAM modificare la definizione di OpenAPI, quindi i tag vengono aggiunti solo se la `DefinitionBody` proprietà è specificata, non viene aggiunto alcun tag se la proprietà è specificata. `DefinitionUri` AWS SAM aggiunge automaticamente un tag. `httpapi:createdBy:SAM` I tag vengono aggiunti anche alla `AWS::ApiGatewayV2::Stage` risorsa e alla `AWS::ApiGatewayV2::DomainName` risorsa (se `DomainName` specificato).

Valori restituiti

Ref

Quando si passa l'ID logico di questa risorsa alla `Ref` funzione intrinseca, `Ref` restituisce l'ID API della `AWS::ApiGatewayV2::Api` risorsa sottostante, ad esempio. `a1bcdef2gh`

Per ulteriori informazioni sull'uso della `Ref` funzione, consulta `Ref`la Guida per l'AWS CloudFormation utente.

Esempi

Semplice HttpApi

L'esempio seguente mostra il minimo necessario per configurare un endpoint API HTTP supportato da una funzione Lambda. Questo esempio utilizza l'API HTTP predefinita che AWS SAM crea.

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: AWS SAM template with a simple API definition
Resources:
  ApiFunction:
    Type: AWS::Serverless::Function
    Properties:
      Events:
        ApiEvent:
          Type: HttpApi
      Handler: index.handler
      InlineCode: |
        def handler(event, context):
            return {'body': 'Hello World!', 'statusCode': 200}
      Runtime: python3.7
Transform: AWS::Serverless-2016-10-31
```

HttpApi con Auth

L'esempio seguente mostra come impostare l'autorizzazione sugli endpoint dell'API HTTP.

YAML

```
Properties:
  FailOnWarnings: true
  Auth:
```

```
    DefaultAuthorizer: OAuth2
    Authorizers:
      OAuth2:
        AuthorizationScopes:
          - scope4
        JwtConfiguration:
          issuer: "https://www.example.com/v1/connect/oauth2"
          audience:
            - MyApi
        IdentitySource: "$request.querystring.param"
```

HttpApi con definizione OpenAPI

L'esempio seguente mostra come aggiungere una definizione OpenAPI al modello.

Tieni presente che AWS SAM compila tutte le integrazioni Lambda mancanti HttpApi per gli eventi che fanno riferimento a questa API HTTP. AWS SAM aggiunge anche eventuali percorsi mancanti a cui fanno riferimento gli HttpApi eventi.

YAML

```
Properties:
  FailOnWarnings: true
  DefinitionBody:
    info:
      version: '1.0'
      title:
        Ref: AWS::StackName
    paths:
      "/":
        get:
          security:
          - OpenIdAuth:
            - scope1
            - scope2
          responses: {}
    openapi: 3.0.1
    securitySchemes:
      OpenIdAuth:
        type: openIdConnect
        x-amazon-apigateway-authorizer:
          identitySource: "$request.querystring.param"
          type: jwt
```

```
          jwtConfiguration:
            audience:
            - MyApi
            issuer: https://www.example.com/v1/connect/oidc
          openIdConnectUrl: https://www.example.com/v1/connect/oidc/.well-known/openid-
configuration
```

HttpApi con impostazioni di configurazione

L'esempio seguente mostra come aggiungere API HTTP e configurazioni di stage al modello.

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Parameters:
  StageName:
    Type: String
    Default: Prod

Resources:
  HttpApiFunction:
    Type: AWS::Serverless::Function
    Properties:
      InlineCode: |
          def handler(event, context):
              import json
              return {
                  "statusCode": 200,
                  "body": json.dumps(event),
              }
      Handler: index.handler
      Runtime: python3.7
      Events:
        ExplicitApi: # warning: creates a public endpoint
          Type: HttpApi
          Properties:
            ApiId: !Ref HttpApi
            Method: GET
            Path: /path
            TimeoutInMillis: 15000
            PayloadFormatVersion: "2.0"
            RouteSettings:
              ThrottlingBurstLimit: 600
```

```
  HttpApi:
    Type: AWS::Serverless::HttpApi
    Properties:
      StageName: !Ref StageName
      Tags:
        Tag: Value
      AccessLogSettings:
        DestinationArn: !GetAtt AccessLogs.Arn
        Format: $context.requestId
      DefaultRouteSettings:
        ThrottlingBurstLimit: 200
      RouteSettings:
        "GET /path":
          ThrottlingBurstLimit: 500 # overridden in HttpApi Event
      StageVariables:
        StageVar: Value
      FailOnWarnings: true

  AccessLogs:
    Type: AWS::Logs::LogGroup

Outputs:
  HttpApiUrl:
    Description: URL of your API endpoint
    Value:
      Fn::Sub: 'https://${HttpApi}.execute-api.${AWS::Region}.${AWS::URLSuffix}/
${StageName}/'
  HttpApiId:
    Description: Api id of HttpApi
    Value:
      Ref: HttpApi
```

HttpApiAuth

Configura l'autorizzazione per controllare l'accesso alla tua API HTTP Amazon API Gateway.

Per ulteriori informazioni sulla configurazione dell'accesso a HTTP APIs, consulta Controllare e gestire l'accesso a un'API HTTP in API Gateway nella API Gateway Developer Guide.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
Authorizers: OAuth2Authorizer | LambdaAuthorizer
DefaultAuthorizer: String
EnableIamAuthorizer: Boolean
```

Properties

`Authorizers`

L'autorizzatore utilizzato per controllare l'accesso all'API API Gateway.

Tipo: OAuth2Autorizzatore | LambdaAuthorizer

Obbligatorio: no

Default: nessuna

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Note aggiuntive: AWS SAM aggiunge gli autorizzatori alla definizione OpenAPI.

`DefaultAuthorizer`

Specificate l'autorizzatore predefinito da utilizzare per autorizzare le chiamate API alla vostra API API Gateway. È possibile specificare `AWS_IAM` come autorizzatore predefinito se `EnableIamAuthorizer` è impostato su. `true` Altrimenti, specifica un autorizzatore che hai definito in. `Authorizers`

➖Tipo: stringa

Obbligatorio: no

Default: nessuna

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`EnableIamAuthorizer`

Specificare se utilizzare l'autorizzazione IAM per il percorso dell'API.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

OAuth Autorizzatore 2.0

OAuth Esempio di autorizzatore 2.0

YAML

```
Auth:
  Authorizers:
    OAuth2Authorizer:
      AuthorizationScopes:
        - scope1
        - scope2
      JwtConfiguration:
        issuer: "https://www.example.com/v1/connect/oauth2"
        audience:
          - MyApi
      IdentitySource: "$request.querystring.param"
  DefaultAuthorizer: OAuth2Authorizer
```

Autorizzatore IAM

Esempio di autorizzatore IAM

YAML

```
Auth:
  EnableIamAuthorizer: true
  DefaultAuthorizer: AWS_IAM
```

LambdaAuthorizer

Configura un autorizzatore Lambda per controllare l'accesso alla tua API HTTP Amazon API Gateway con una funzione. AWS Lambda

Per ulteriori informazioni ed esempi, consulta Working with AWS Lambda authorizers for HTTP APIs nella API Gateway Developer Guide.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
AuthorizerPayloadFormatVersion: String
EnableFunctionDefaultPermissions: Boolean
EnableSimpleResponses: Boolean
FunctionArn: String
FunctionInvokeRole: String
Identity: LambdaAuthorizationIdentity
```

Properties

`AuthorizerPayloadFormatVersion`

Specifica il formato del payload inviato a un autorizzatore Lambda API HTTP. Richiesto per gli autorizzatori Lambda API HTTP.

Questo viene passato alla `authorizerPayloadFormatVersion` sezione di an `x-amazon-apigateway-authorizer` nella `securitySchemes` sezione di una definizione OpenAPI.

Valori validi: `1.0` o `2.0`

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`EnableFunctionDefaultPermissions`

Per impostazione predefinita, alla risorsa API HTTP non viene concessa l'autorizzazione per richiamare l'autorizzatore Lambda. Specificate questa proprietà `true` per creare automaticamente le autorizzazioni tra la risorsa API HTTP e l'autorizzatore Lambda.

Tipo: Booleano

Obbligatorio: no

Valore predefinito: `false`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`EnableSimpleResponses`

Specifica se un autorizzatore Lambda restituisce una risposta in un formato semplice. Per impostazione predefinita, un autorizzatore Lambda deve restituire una policy AWS Identity and Access Management (IAM). Se abilitato, l'autorizzatore Lambda può restituire un valore booleano invece di una policy IAM.

Questo viene passato alla `enableSimpleResponses` sezione di an `x-amazon-apigateway-authorizer` nella `securitySchemes` sezione di una definizione OpenAPI.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`FunctionArn`

L'Amazon Resource Name (ARN) della funzione Lambda che fornisce l'autorizzazione per l'API.

Questo viene passato alla `authorizerUri` sezione di an `x-amazon-apigateway-authorizer` nella `securitySchemes` sezione di una definizione OpenAPI.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`FunctionInvokeRole`

L'ARN del ruolo IAM che dispone delle credenziali necessarie affinché API Gateway richiami la funzione di autorizzazione. Specificate questo parametro se la policy basata sulle risorse della funzione non concede l'autorizzazione API Gateway. `lambda:InvokeFunction`

Questo viene passato alla `authorizerCredentials` sezione di an `x-amazon-apigateway-authorizer` nella `securitySchemes` sezione di una definizione OpenAPI.

Per ulteriori informazioni, consulta Create a Lambda authorizer nella API Gateway Developer Guide.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Identity`

Speciifica un elemento `IdentitySource` in una richiesta in entrata per un autorizzatore.

Questo viene passato alla `identitySource` sezione di an `x-amazon-apigateway-authorizer` nella `securitySchemes` sezione di una definizione OpenAPI.

Tipo: LambdaAuthorizationIdentity

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

LambdaAuthorizer

LambdaAuthorizer esempio

YAML

```
Auth:
  Authorizers:
    MyLambdaAuthorizer:
      AuthorizerPayloadFormatVersion: 2.0
      FunctionArn:
        Fn::GetAtt:
          - MyAuthFunction
```

```
          - Arn
      FunctionInvokeRole:
        Fn::GetAtt:
          - LambdaAuthInvokeRole
          - Arn
      Identity:
        Headers:
          - Authorization
```

LambdaAuthorizationIdentity

La proprietà use può essere utilizzata per specificare un IdentitySource in una richiesta in entrata per un autorizzatore Lambda. Per ulteriori informazioni sulle fonti di identità, consulta Identity sources nella API Gateway Developer Guide.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Context: List
Headers: List
QueryStrings: List
ReauthorizeEvery: Integer
StageVariables: List
```

Properties

`Context`

Converte le stringhe di contesto fornite in un elenco di espressioni di mappatura nel formato. `$context.contextString`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`Headers`

Converte le intestazioni in un elenco di espressioni di mappatura nel formato. `$request.header.name`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`QueryStrings`

Converte le stringhe di query fornite in un elenco di espressioni di mappatura nel formato. `$request.querystring.queryString`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`ReauthorizeEvery`

Il periodo time-to-live (TTL), in secondi, che specifica per quanto tempo API Gateway memorizza nella cache i risultati dell'autorizzazione. Se il valore specificato è maggiore di 0, API Gateway memorizza nella cache le risposte di autorizzazione. Il valore massimo è 3600, ovvero un'ora.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`StageVariables`

Converte le variabili di fase specificate in un elenco di espressioni di mappatura nel formato. `$stageVariables.stageVariable`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

LambdaRequestIdentity

Esempio di identità della richiesta Lambda

YAML

```
Identity:
  QueryStrings:
    - auth
  Headers:
    - Authorization
  StageVariables:
    - VARIABLE
  Context:
    - authcontext
  ReauthorizeEvery: 100
```

OAuth2Authorizer

Definizione di un autorizzatore OAuth 2.0, noto anche come autorizzatore JSON Web Token (JWT).

Per ulteriori informazioni, consulta Controllare l'accesso a HTTP APIs con gli autorizzatori JWT nella API Gateway Developer Guide.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  AuthorizationScopes: List
  IdentitySource: String
  JwtConfiguration: Map
```

Properties

`AuthorizationScopes`

Elenco degli ambiti di autorizzazione per questo autorizzatore.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IdentitySource`

Espressione di origine dell'identità per questo autorizzatore.

Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`JwtConfiguration`

Configurazione JWT per questo autorizzatore.

Questo viene passato alla `jwtConfiguration` sezione di an `x-amazon-apigateway-authorizer` nella `securitySchemes` sezione di una definizione OpenAPI.

> **Note**
>
> Le proprietà `issuer` e `audience` non fanno distinzione tra maiuscole e minuscole e possono essere utilizzate sia in minuscolo come in OpenAPI che in maiuscolo e come in. `Issuer` `Audience` AWS::ApiGatewayV2::Authorizer

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

OAuth autorizzatore 2.0

OAuth Esempio di autorizzatore 2.0

YAML

```
Auth:
  Authorizers:
    OAuth2Authorizer:
      AuthorizationScopes:
        - scope1
      JwtConfiguration:
        issuer: "https://www.example.com/v1/connect/oauth2"
        audience:
          - MyApi
      IdentitySource: "$request.querystring.param"
  DefaultAuthorizer: OAuth2Authorizer
```

HttpApiCorsConfiguration

Gestisci la condivisione delle risorse tra le origini (CORS) per il tuo HTTP. APIs Specificate il dominio da consentire come stringa o specificate un dizionario con una configurazione Cors aggiuntiva. NOTA: Cors richiede SAM per modificare la definizione di OpenAPI, quindi funziona solo con OpenApi inline definito nella proprietà. `DefinitionBody`

Per ulteriori informazioni su CORS, vedere Configuring CORS for an HTTP API nella API Gateway Developer Guide.

Nota: se HttpApiCorsConfiguration è impostato sia in OpenAPI che a livello di proprietà, li AWS SAM unisce alle proprietà che hanno la precedenza.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  AllowCredentials: Boolean
```

```
  AllowHeaders: List
  AllowMethods: List
  AllowOrigins: List
  ExposeHeaders: List
  MaxAge: Integer
```

Properties

`AllowCredentials`

Specifica se le credenziali sono incluse nella richiesta CORS.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AllowHeaders`

Rappresenta una raccolta di intestazioni consentite.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AllowMethods`

Rappresenta una raccolta di metodi HTTP consentiti.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AllowOrigins`

Rappresenta una raccolta di origini consentite.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ExposeHeaders`

Rappresenta una raccolta di intestazioni esposte.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`MaxAge`

Specifica il numero di secondi in cui il browser deve memorizzare nella cache i risultati delle richieste preliminari.

Tipo: integer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

HttpApiCorsConfiguration

Esempio di configurazione HTTP API Cors.

YAML

```
CorsConfiguration:
  AllowOrigins:
    - "https://example.com"
  AllowHeaders:
    - x-apigateway-header
```

```
  AllowMethods:
    - GET
  MaxAge: 600
  AllowCredentials: true
```

HttpApiDefinition

Un documento OpenAPI che definisce l'API.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Bucket: String
Key: String
Version: String
```

Properties

`Bucket`

Il nome del bucket Amazon S3 in cui è archiviato il file OpenAPI.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Bucket` proprietà del tipo di dati. `AWS::ApiGatewayV2::Api BodyS3Location`

`Key`

La chiave Amazon S3 del file OpenAPI.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Key` proprietà del tipo di `AWS::ApiGatewayV2::Api BodyS3Location` dati.

`Version`

Per gli oggetti con versione, la versione del file OpenAPI.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Version` proprietà del tipo di `AWS::ApiGatewayV2::Api` `BodyS3Location` dati.

Esempi

Definizione: esempio Uri

Esempio di definizione dell'API

YAML

```
DefinitionUri:
  Bucket: sam-s3-demo-bucket-name
  Key: mykey-name
  Version: 121212
```

HttpApiDomainConfiguration

Configura un dominio personalizzato per un'API.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  BasePath: List
  CertificateArn: String
  DomainName: String
  EndpointConfiguration: String
  MutualTlsAuthentication: MutualTlsAuthentication
  OwnershipVerificationCertificateArn: String
```

```
  Route53: Route53Configuration
  SecurityPolicy: String
```

Properties

`BasePath`

Un elenco dei percorsi di base da configurare con il nome di dominio Amazon API Gateway.

Tipo: elenco

Obbligatorio: no

Predefinito:/

CloudFormation compatibilità: questa proprietà è simile alla `ApiMappingKey` proprietà di una `AWS::ApiGatewayV2::ApiMapping` risorsa. AWS SAM crea più `AWS::ApiGatewayV2::ApiMapping` risorse, una per valore specificato in questa proprietà.

`CertificateArn`

L'Amazon Resource Name (ARN) di un certificato AWS gestito per l'endpoint di questo nome di dominio. AWS Certificate Manager è l'unica fonte supportata.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CertificateArn` proprietà di una `AWS::ApiGateway2::DomainName` `DomainNameConfiguration` risorsa.

`DomainName`

Il nome di dominio personalizzato per l'API API Gateway. Le lettere maiuscole non sono supportate.

AWS SAM genera una `AWS::ApiGatewayV2::DomainName` risorsa quando questa proprietà è impostata. Per informazioni su questo scenario, vedereDomainName la proprietà è specificata. Per informazioni sulle CloudFormation risorse generate, vedereCloudFormation Risorse generate per AWS SAM.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `DomainName` proprietà di una `AWS::ApiGateway2::DomainName` risorsa.

`EndpointConfiguration`

Definisce il tipo di endpoint API Gateway da mappare al dominio personalizzato. Il valore di questa proprietà determina il modo in cui la `CertificateArn` proprietà viene mappata. CloudFormation

L'unico valore valido per HTTP APIs è`REGIONAL`.

- Tipo: stringa

Obbligatorio: no

Default: `REGIONAL`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`MutualTlsAuthentication`

La configurazione di autenticazione TLS (Mutual Transport Layer Security) per un nome di dominio personalizzato.

Tipo: MutualTlsAuthentication

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `MutualTlsAuthentication` proprietà di una `AWS::ApiGatewayV2::DomainName` risorsa.

`OwnershipVerificationCertificateArn`

L'ARN del certificato pubblico emesso da ACM per convalidare la proprietà del dominio personalizzato. Richiesto solo quando si configura il TLS reciproco e si specifica un ARN di certificato CA privato o importato da ACM per. `CertificateArn`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `OwnershipVerificationCertificateArn` proprietà del tipo di dati. `AWS::ApiGatewayV2::DomainName` `DomainNameConfiguration`

`Route53`

Definisce una configurazione Amazon Route 53.

Tipo: configurazione Route53

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`SecurityPolicy`

La versione TLS della politica di sicurezza per questo nome di dominio.

L'unico valore valido per HTTP APIs è`TLS_1_2`.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SecurityPolicy` proprietà del tipo di `AWS::ApiGatewayV2::DomainName` `DomainNameConfiguration` dati.

Esempi

DomainName

DomainName esempio

YAML

```
Domain:
  DomainName: www.example.com
  CertificateArn: arn-example
  EndpointConfiguration: REGIONAL
  Route53:
    HostedZoneId: Z1PA6795UKMFR9
  BasePath:
```

```
    - foo
    - bar
```

Route53Configuration

Configura i set di record Route53 per un'API.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
  DistributionDomainName: String
  EvaluateTargetHealth: Boolean
  HostedZoneId: String
  HostedZoneName: String
  IpV6: Boolean
  Region: String
  SetIdentifier: String
```

Properties

`DistributionDomainName`

Configura una distribuzione personalizzata del nome di dominio personalizzato dell'API.

▬Tipo: stringa

Obbligatorio: no

Impostazione predefinita: utilizza la distribuzione API Gateway.

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `DNSName` proprietà di una `AWS::Route53::RecordSetGroup AliasTarget` risorsa.

Note aggiuntive: il nome di dominio di una CloudFrontdistribuzione.

`EvaluateTargetHealth`

Quando EvaluateTargetHealth è vero, un record alias eredita lo stato della AWS risorsa di riferimento, ad esempio un sistema di bilanciamento del carico Elastic Load Balancing o un altro record nella zona ospitata.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla proprietà di una risorsa. `EvaluateTargetHealth` `AWS::Route53::RecordSetGroup AliasTarget`

Note aggiuntive: non è possibile impostare su EvaluateTargetHealth true quando l'alias target è una CloudFront distribuzione.

`HostedZoneId`

L'ID della zona ospitata in cui creare i record.

Specifica `HostedZoneName` o `HostedZoneId`, ma non entrambi. Se disponi di più zone ospitate con lo stesso nome di dominio, devi specificare la zona ospitata utilizzando `HostedZoneId`.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `HostedZoneId` proprietà di una `AWS::Route53::RecordSetGroup RecordSet` risorsa.

`HostedZoneName`

Il nome della zona ospitata in cui desideri creare record. È necessario includere un punto finale (ad esempio, `www.example.com.`) come parte di `HostedZoneName`.

Specifica `HostedZoneName` o `HostedZoneId`, ma non entrambi. Se disponi di più zone ospitate con lo stesso nome di dominio, devi specificare la zona ospitata utilizzando `HostedZoneId`.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `HostedZoneName` proprietà di una `AWS::Route53::RecordSetGroup RecordSet` risorsa.

`IpV6`

Quando questa proprietà è impostata, AWS SAM crea una `AWS::Route53::RecordSet` risorsa e imposta `Type` su `AAAA` quella fornita HostedZone.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Region`

Solo set di record di risorse basati sulla latenza: la EC2 regione Amazon in cui hai creato la risorsa a cui si riferisce questo set di record di risorse. La risorsa è in genere una AWS risorsa, EC2 ad esempio un'istanza o un sistema di bilanciamento del carico ELB, a cui si fa riferimento tramite un indirizzo IP o un nome di dominio DNS, a seconda del tipo di record.

Quando Amazon Route 53 riceve una query DNS per un nome e un tipo di dominio per i quali sono stati creati set di record di risorse di latenza, Route 53 seleziona il set di record di risorse di latenza con la latenza più bassa tra l'utente finale e la regione Amazon associata. EC2 Route 53 restituisce il valore associato al set di record della risorsa selezionato.

Tenere presente quanto segue:

- È possibile specificare un `ResourceRecord` per set di record di risorse di latenza.
- Puoi creare un solo set di record di risorse di latenza per ogni EC2 regione Amazon.
- Non è necessario creare set di record di risorse di latenza per tutte le EC2 regioni Amazon. Route 53 sceglie la Regione con la migliore latenza tra quelle per cui hai creato i set di record della risorsa di latenza.
- Non è possibile creare set di record di risorse non di latenza le cui proprietà `Name` e `Type` abbiano lo stesso valore dei set di record di risorse di latenza.

➖Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Region` proprietà di un tipo di `AWS::Route53::RecordSetGroup` `RecordSet` dati.

`SetIdentifier`

Set di record di risorse con un criterio di routing non semplice: un identificatore che differenzia tra più set di record di risorse che hanno la stessa combinazione di nome e tipo, ad esempio più set di record di risorse ponderati denominati acme.example.com di tipo A. In un gruppo di set di record di risorse con lo steso nome e lo stesso tipo, il valore di `SetIdentifier` deve essere univoco per ogni set di record di risorse.

Per informazioni sulle politiche di routing, consulta la sezione Choose a routing policy nella Amazon Route 53 Developer Guide.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SetIdentifier` proprietà di un tipo di `AWS::Route53::RecordSetGroup` `RecordSet` dati.

Esempi

Esempio di configurazione della Route 53

Questo esempio mostra come configurare Route 53.

YAML

```
Domain:
  DomainName: www.example.com
  CertificateArn: arn-example
  EndpointConfiguration: EDGE
  Route53:
    HostedZoneId: Z1PA6795UKMFR9
    EvaluateTargetHealth: true
    DistributionDomainName: xyz
```

AWS::Serverless::LayerVersion

Crea una Lambda LayerVersion che contiene il codice di libreria o di runtime necessario per una funzione Lambda.

La AWS::Serverless::LayerVersion risorsa supporta anche l'attributo `Metadata` resource, quindi puoi indicare di AWS SAM creare livelli inclusi nell'applicazione. Per ulteriori informazioni sulla creazione di livelli, consultaCreazione di livelli Lambda in AWS SAM.

Nota importante: dal rilascio dell'attributo UpdateReplacePolicyresource in CloudFormation, AWS::Lambda::LayerVersion(consigliato) offre gli stessi vantaggi diAWS::Serverless::LayerVersion.

Quando un Serverless LayerVersion viene trasformato, SAM trasforma anche l'ID logico della risorsa in modo che LayerVersions le versioni precedenti non vengano eliminate automaticamente CloudFormation quando la risorsa viene aggiornata.

> **Note**
>
> Quando esegui la distribuzione in AWS CloudFormation, AWS SAM trasforma le tue AWS SAM risorse in risorse. CloudFormation Per ulteriori informazioni, consulta CloudFormation Risorse generate per AWS SAM.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Type: AWS::Serverless::LayerVersion
Properties:
  CompatibleArchitectures: List
  CompatibleRuntimes: List
  ContentUri: String | LayerContent
  Description: String
  LayerName: String
  LicenseInfo: String
  PublishLambdaVersion: Boolean
  RetentionPolicy: String
```

Properties

`CompatibleArchitectures`

Specifica le architetture dei set di istruzioni supportate per la versione di livello.

Per ulteriori informazioni su questa proprietà, consulta Architetture dei set di istruzioni Lambda nella Guida per gli sviluppatori.AWS Lambda

Valori validi: `x86_64`, `arm64`

Tipo: elenco

Obbligatorio: no

Default: `x86_64`

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CompatibleArchitectures` proprietà di una risorsa. `AWS::Lambda::LayerVersion`

`CompatibleRuntimes`

Elenco di runtime compatibili con questo LayerVersion.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `CompatibleRuntimes` proprietà di una `AWS::Lambda::LayerVersion` risorsa.

`ContentUri`

Amazon S3 Uri, percorso della cartella locale o LayerContent oggetto del codice del livello.

Se viene fornito un Uri o un oggetto Amazon S3, l' LayerContent oggetto Amazon S3 a cui si fa riferimento deve essere un archivio ZIP valido che contiene il contenuto di un livello Lambda.

Se viene fornito un percorso a una cartella locale, affinché il contenuto venga trasformato correttamente, il modello deve passare attraverso il flusso di lavoro che include, sam build seguito da o. sam deploy sam package Per impostazione predefinita, i percorsi relativi vengono risolti rispetto alla posizione del AWS SAM modello.

Tipo: String | LayerContent

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà è simile alla `Content` proprietà di una `AWS::Lambda::LayerVersion` risorsa. Le proprietà annidate di Amazon S3 sono denominate in modo diverso.

`Description`

Descrizione di questo livello.

-Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una `AWS::Lambda::LayerVersion` risorsa.

`LayerName`

Il nome o l'Amazon Resource Name (ARN) del livello.

- Tipo: stringa

Obbligatorio: no

Predefinito: ID logico della risorsa

CloudFormation compatibilità: questa proprietà è simile alla `LayerName` proprietà di una `AWS::Lambda::LayerVersion` risorsa. Se non si specifica un nome, l'id logico della risorsa verrà utilizzato come nome.

`LicenseInfo`

Informazioni sulla licenza a tale scopo LayerVersion.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà viene passata direttamente alla `LicenseInfo` proprietà di una `AWS::Lambda::LayerVersion` risorsa.

`PublishLambdaVersion`

Una proprietà opt-in che crea una nuova versione Lambda ogni volta che si verifica una modifica nella `LayerVersion` risorsa di riferimento. Se abilitata con `AutoPublishAlias` e `AutoPublishAliasAllProperties` nella funzione Lambda connessa, verrà creata una nuova versione Lambda per ogni modifica apportata alla risorsa. `LayerVersion`

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`RetentionPolicy`

Questa proprietà specifica se le versioni precedenti del file `LayerVersion` vengono conservate o eliminate quando si elimina una risorsa. Se è necessario conservare le vecchie versioni della risorsa `LayerVersion` quando si aggiorna o si sostituisce una risorsa, è necessario che

l'`UpdateReplacePolicy`attributo sia abilitato. Per informazioni su questa operazione, consulta l'`UpdateReplacePolicy`attributo nella Guida per l'AWS CloudFormation utente.

Valori validi: `Retain` o `Delete`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Note aggiuntive: quando si specifica`Retain`, AWS SAM aggiunge un Attributi delle risorse supportati da AWS SAM di `DeletionPolicy: Retain` alla `AWS::Lambda::LayerVersion` risorsa trasformata.

Valori restituiti

Ref

Quando l'ID logico di questa risorsa viene fornito alla funzione `Ref` intrinseca, restituisce l'ARN della risorsa Lambda sottostante. LayerVersion

Per ulteriori informazioni sull'uso della *`Ref`* funzione, consulta *`Ref`*la Guida per l'utente.AWS CloudFormation

Esempi

LayerVersionExample

Esempio di LayerVersion

YAML

```
Properties:
  LayerName: MyLayer
  Description: Layer description
  ContentUri: 's3://sam-s3-demo-bucket/my-layer.zip'
  CompatibleRuntimes:
    - nodejs10.x
    - nodejs12.x
  LicenseInfo: 'Available under the MIT-0 license.'
```

```
  RetentionPolicy: Retain
```

LayerContent

Un archivio ZIP che racchiude i contenuti di un livello Lambda.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  Bucket: String
  Key: String
  Version: String
```

Properties

`Bucket`

Il bucket Amazon S3 dell'archivio del livello.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `S3Bucket` proprietà del tipo di `AWS::Lambda::LayerVersion` `Content` dati.

`Key`

La chiave Amazon S3 dell'archivio del livello.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `S3Key` proprietà del tipo di `AWS::Lambda::LayerVersion` `Content` dati.

`Version`

Per gli oggetti con versione, la versione dell'oggetto dell'archivio del livello da utilizzare.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `S3ObjectVersion` proprietà del tipo di `AWS::Lambda::LayerVersion` `Content` dati.

Esempi

LayerContent

Esempio di contenuto del livello

YAML

```
LayerContent:
  Bucket: amzn-s3-demo-bucket-name
  Key: mykey-name
  Version: 121212
```

AWS::Serverless::SimpleTable

Crea una tabella DynamoDB con una chiave primaria a singolo attributo. È utile quando è necessario accedere ai dati solo tramite una chiave primaria.

Per funzionalità più avanzate, utilizza una AWS::DynamoDB::Tablerisorsa in CloudFormation. Queste risorse possono essere utilizzate in. AWS SAM Sono completi e offrono ulteriori personalizzazioni, inclusa key schemala personalizzazione. resource policy

> **Note**
>
> Quando lo distribuisci AWS CloudFormation, AWS SAM trasforma le tue AWS SAM risorse in risorse. CloudFormation Per ulteriori informazioni, consulta CloudFormation Risorse generate per AWS SAM.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Type: AWS::Serverless::SimpleTable
Properties:
  PointInTimeRecoverySpecification: PointInTimeRecoverySpecification
  PrimaryKey: PrimaryKeyObject
  ProvisionedThroughput: ProvisionedThroughputObject
  SSESpecification: SSESpecification
  TableName: String
  Tags: Map
```

Properties

`PointInTimeRecoverySpecification`

Le impostazioni utilizzate per consentire il ripristino point-in-time.

Tipo: PointInTimeRecoverySpecification

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `PointInTimeRecoverySpecification` proprietà di una `AWS::DynamoDB::Table` risorsa.

`PrimaryKey`

Nome e tipo di attributo da utilizzare come chiave primaria della tabella. Se non viene fornita, la chiave primaria sarà un `String` con un valore di`id`.

> **Note**
>
> Il valore di questa proprietà non può essere modificato dopo la creazione di questa risorsa.

Tipo: PrimaryKeyObject

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`ProvisionedThroughput`

Leggi e scrivi le informazioni sul throughput provisioning.

Se non `ProvisionedThroughput` è specificato `BillingMode` verrà specificato come. `PAY_PER_REQUEST`

Tipo: ProvisionedThroughputObject

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ProvisionedThroughput` proprietà di una `AWS::DynamoDB::Table` risorsa.

`SSESpecification`

Specifica le impostazioni per abilitare la crittografia lato server.

Tipo: SSESpecification

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `SSESpecification` proprietà di una `AWS::DynamoDB::Table` risorsa.

`TableName`

Nome per la tabella DynamoDB.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `TableName` proprietà di una `AWS::DynamoDB::Table` risorsa.

`Tags`

Una mappa (da stringa a stringa) che specifica i tag da aggiungere a questa SimpleTable. Per i dettagli sulle chiavi e i valori validi per i tag, consulta Resource tag nella Guida per l'AWS CloudFormation utente.

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `Tags` proprietà di una `AWS::DynamoDB::Table` risorsa. La proprietà Tags in SAM è costituita da coppie Key:Value; in CloudFormation essa è costituita da un elenco di oggetti Tag.

Valori restituiti

Ref

Quando l'ID logico di questa risorsa viene fornito alla funzione intrinseca Ref, restituisce il nome della risorsa della tabella DynamoDB sottostante.

Per ulteriori informazioni sull'uso della *`Ref`* funzione, consulta *`Ref`*la Guida per l'utente.AWS CloudFormation

Esempi

SimpleTableExample

Esempio di SimpleTable

YAML

```
Properties:
  TableName: my-table
  PrimaryKey:
    Name: MyPrimaryKey
    Type: String
  ProvisionedThroughput:
    ReadCapacityUnits: 5
    WriteCapacityUnits: 5
  Tags:
    Department: Engineering
    AppType: Serverless
```

PrimaryKeyObject

L'oggetto che descrive le proprietà di una chiave primaria.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Name: String
Type: String
```

Properties

`Name`

Nome dell'attributo della chiave primaria.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `AttributeName` proprietà del tipo di `AWS::DynamoDB::Table` `AttributeDefinition` dati.

Note aggiuntive: questa proprietà viene passata anche alla AttributeNameproprietà di un tipo di `AWS::DynamoDB::Table` `KeySchema` dati.

`Type`

Il tipo di dati per la chiave primaria.

Valori validi:`String`,`Number`, `Binary`

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `AttributeType` proprietà del tipo di `AWS::DynamoDB::Table` `AttributeDefinition` dati.

Esempi

PrimaryKey

Esempio di chiave primaria.

YAML

```
Properties:
  PrimaryKey:
```

```
    Name: MyPrimaryKey
    Type: String
```

ProvisionedThroughputObject

L'oggetto che descrive le proprietà di un throughput assegnato.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  ReadCapacityUnits: Integer
  WriteCapacityUnits: Integer
```

Properties

`ReadCapacityUnits`

Il numero massimo di letture fortemente consistenti consumate al secondo prima che DynamoDB restituisca un `ThrottlingException`.

Tipo: integer

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ReadCapacityUnits` proprietà del tipo di `AWS::DynamoDB::Table` `ProvisionedThroughput` dati.

`WriteCapacityUnits`

Il numero massimo di scritture consumate al secondo prima che DynamoDB restituisca un `ThrottlingException`.

Tipo: integer

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `WriteCapacityUnits` proprietà del tipo di `AWS::DynamoDB::Table` `ProvisionedThroughput` dati.

Esempi

ProvisionedThroughput

Esempio di throughput assegnato.

YAML

```
Properties:
   ProvisionedThroughput:
     ReadCapacityUnits: 5
     WriteCapacityUnits: 5
```

AWS::Serverless::StateMachine

Crea una macchina a AWS Step Functions stati, che puoi utilizzare per orchestrare AWS Lambda funzioni e altre AWS risorse per formare flussi di lavoro complessi e robusti.

Per ulteriori informazioni su Step Functions, consulta la AWS Step Functions Guida per sviluppatori di .

> **Note**
>
> Quando lo distribuisci su AWS CloudFormation, AWS SAM trasforma le tue risorse in risorse. AWS SAM CloudFormation Per ulteriori informazioni, consulta CloudFormation Risorse generate per AWS SAM.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Type: AWS::Serverless::StateMachine
Properties:
  AutoPublishAlias: String
  UseAliasAsEventTarget: Boolean
  Definition: Map
```

```
  DefinitionSubstitutions: Map
  DefinitionUri: String | S3Location
  DeploymentPreference: DeploymentPreference
  Events: EventSource
  Logging: LoggingConfiguration
  Name: String
  PermissionsBoundary: String
  Policies: String | List | Map
  PropagateTags: Boolean
  RolePath: String
  Role: String
  Tags: Map
  Tracing: TracingConfiguration
  Type: String
```

Properties

`AutoPublishAlias`

Il nome dell'alias della macchina a stati. Per ulteriori informazioni sull'utilizzo degli alias delle macchine a stati Step Functions, consulta Gestire le distribuzioni continue con versioni e alias nella Developer Guide.AWS Step Functions

Utilizzalo `DeploymentPreference` per configurare le preferenze di distribuzione per il tuo alias. Se non lo specifichi`DeploymentPreference`, AWS SAM configurerà il traffico in modo che passi alla versione più recente della macchina a stati contemporaneamente.

AWS SAM imposta la versione `DeletionPolicy` e la versione `UpdateReplacePolicy` di `Retain` default. Le versioni precedenti non verranno eliminate automaticamente.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::StepFunctions::StateMachineAlias` risorsa.

`UseAliasAsEventTarget`

Indica se passare o meno l'alias, creato utilizzando la `AutoPublishAlias` proprietà, alla destinazione dell'origine degli eventi definita con Events.

`True`Specificare di utilizzare l'alias come destinazione degli eventi.

Tipo: Booleano

Obbligatorio: no

Default: `False`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Definition`

La definizione della macchina a stati è un oggetto, in cui il formato dell'oggetto corrisponde al formato del file AWS SAM modello, ad esempio JSON o YAML. Le definizioni delle macchine a stati aderiscono ad Amazon States Language.

Per un esempio di definizione di macchina a stati in linea, vedere. Esempi

È necessario fornire un `Definition` o un`DefinitionUri`.

Tipo: Mappa

Required: Conditional

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`DefinitionSubstitutions`

Una string-to-string mappa che specifica le mappature per le variabili segnaposto nella definizione della macchina a stati. Ciò consente di inserire i valori ottenuti in fase di esecuzione (ad esempio, da funzioni intrinseche) nella definizione della macchina a stati.

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `DefinitionSubstitutions` proprietà di una `AWS::StepFunctions::StateMachine` risorsa. Se vengono specificate funzioni intrinseche in una definizione di macchina a stati in linea, AWS SAM aggiunge voci a questa proprietà per inserirle nella definizione della macchina a stati.

`DefinitionUri`

L'URI di Amazon Simple Storage Service (Amazon S3) o il percorso del file locale della definizione della macchina a stati scritta in Amazon States Language.

Se fornisci un percorso di file locale, il modello deve passare attraverso il flusso di lavoro che include il `sam package` comando `sam deploy` o per trasformare correttamente la definizione. A tale scopo, è necessario utilizzare la versione 0.52.0 o successiva della CLI AWS SAM .

È necessario fornire un o un`Definition`. `DefinitionUri`

Tipo: String | S3Location

Required: Conditional

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `DefinitionS3Location` proprietà di una `AWS::StepFunctions::StateMachine` risorsa.

`DeploymentPreference`

Le impostazioni che abilitano e configurano l'implementazione graduale di macchine a stati. Per ulteriori informazioni sulle distribuzioni graduali di Step Functions, consulta Manage continuous deployments with versions and alias nella Developer Guide.AWS Step Functions

Specificare prima di configurare questa proprietà. `AutoPublishAlias` `DeploymentPreference`Le impostazioni verranno applicate all'alias specificato con. `AutoPublishAlias`

Quando si specifica`DeploymentPreference`, AWS SAM genera automaticamente il valore della `StateMachineVersionArn` sottoproprietà.

Tipo: DeploymentPreference

Obbligatorio: no

CloudFormation compatibilità: AWS SAM genera e attribuisce il valore della `StateMachineVersionArn` proprietà `DeploymentPreference` e lo passa `DeploymentPreference` alla `DeploymentPreference` proprietà di una `AWS::StepFunctions::StateMachineAlias` risorsa.

`Events`

Speciifica gli eventi che attivano questa macchina a stati. Gli eventi sono costituiti da un tipo e da un insieme di proprietà che dipendono dal tipo.

Tipo: EventSource

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Logging`

Definisce quali eventi della cronologia di esecuzione vengono registrati e dove vengono registrati.

Tipo: LoggingConfiguration

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `LoggingConfiguration` proprietà di una `AWS::StepFunctions::StateMachine` risorsa.

`Name`

Il nome della macchina a stati.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StateMachineName` proprietà di una `AWS::StepFunctions::StateMachine` risorsa.

`PermissionsBoundary`

L'ARN di un limite di autorizzazioni da utilizzare per il ruolo di esecuzione di questa macchina a stati. Questa proprietà funziona solo se il ruolo viene generato automaticamente.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `PermissionsBoundary` proprietà di una `AWS::IAM::Role` risorsa.

`Policies`

Politiche di autorizzazione per questa macchina a stati. Le politiche verranno aggiunte al ruolo di esecuzione predefinito AWS Identity and Access Management (IAM) della macchina a stati.

Questa proprietà accetta un singolo valore o un elenco di valori. I valori autorizzati includono:

- AWS SAM modelli di policy.
- ARNDi una politica AWS gestita o di una politica gestita dal cliente.

- Il nome di una politica AWS gestita dall'elenco seguente.
- Una politica IAM in linea formattata YAML come mappa.

> **Note**
>
> Se si imposta la `Role` proprietà, questa proprietà viene ignorata.

Tipo: Stringa | Elenco | Mappa

Obbligatorio: no

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`PropagateTags`

Indica se passare o meno i tag dalla `Tags` proprietà alle risorse AWS::Serverless::StateMachine generate. `True`Specificate di propagare i tag nelle risorse generate.

Tipo: Booleano

Obbligatorio: no

Default: `False`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Role`

L'ARN di un ruolo IAM da utilizzare come ruolo di esecuzione di questa macchina a stati.

Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RoleArn` proprietà di una `AWS::StepFunctions::StateMachine` risorsa.

`RolePath`

Il percorso del ruolo di esecuzione IAM della macchina a stati.

Usa questa proprietà quando il ruolo viene generato per te. Non utilizzare quando il ruolo è specificato con la `Role` proprietà.

- Tipo: stringa

Required: Conditional

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Path` proprietà di una `AWS::IAM::Role` risorsa.

`Tags`

Una string-to-string mappa che specifica i tag aggiunti alla macchina a stati e il ruolo di esecuzione corrispondente. Per informazioni sulle chiavi e i valori validi per i tag, vedete la proprietà Tags di una AWS::StepFunctions::StateMachinerisorsa.

Tipo: Mappa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `Tags` proprietà di una `AWS::StepFunctions::StateMachine` risorsa. AWS SAM aggiunge automaticamente un `stateMachine:createdBy:SAM` tag a questa risorsa e al ruolo predefinito generato per essa.

`Tracing`

Seleziona se AWS X-Ray è abilitato o meno per la macchina a stati. Per ulteriori informazioni sull'utilizzo di X-Ray con Step Functions, vedere AWS X-Ray Step Functions nella AWS Step Functions Developer Guide.

Tipo: TracingConfiguration

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `TracingConfiguration` proprietà di una `AWS::StepFunctions::StateMachine` risorsa.

`Type`

Il tipo di macchina a stati.

Valori validi: `STANDARD` o `EXPRESS`

- Tipo: stringa

Obbligatorio: no

Default: `STANDARD`

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StateMachineType` proprietà di una `AWS::StepFunctions::StateMachine` risorsa.

Valori restituiti

Ref

Quando fornisci l'ID logico di questa risorsa alla funzione intrinseca Ref, Ref restituisce l'Amazon Resource Name (ARN) della risorsa sottostante. `AWS::StepFunctions::StateMachine`

Per ulteriori informazioni sull'uso della *`Ref`* funzione, consulta *`Ref`*la Guida per l'AWS CloudFormation utente.

Fn: GetAtt

`Fn::GetAtt` restituisce un valore per un attributo specificato di questo tipo. Di seguito sono riportati gli attributi disponibili e i valori restituiti di esempio.

Per ulteriori informazioni sull'utilizzo`Fn::GetAtt`, consulta `Fn::GetAtt`la Guida per l'AWS CloudFormation utente.

`Name`

Restituisce il nome della macchina a stati, ad esempio`HelloWorld-StateMachine`.

Esempi

File di definizione della macchina a stati

Di seguito è riportato un esempio di definizione di macchina a stati in linea che consente a una funzione lambda di richiamare una macchina a stati. Si noti che questo esempio prevede che la `Role` proprietà configuri una politica appropriata per consentire l'invocazione. Il `my_state_machine.asl.json` file deve essere scritto in Amazon States Language.

In questo esempio, le `DefinitionSubstitution` voci consentono alla macchina a stati di includere risorse dichiarate nel file AWS SAM modello.

YAML

```
MySampleStateMachine:
  Type: AWS::Serverless::StateMachine
  Properties:
    DefinitionUri: statemachine/my_state_machine.asl.json
    Role: arn:aws:iam::123456123456:role/service-role/my-sample-role
    Tracing:
      Enabled: true
    DefinitionSubstitutions:
      MyFunctionArn: !GetAtt MyFunction.Arn
      MyDDBTable: !Ref TransactionTable
```

Definizione della macchina a stati in linea

Di seguito è riportato un esempio di definizione di macchina a stati in linea.

In questo esempio, il file AWS SAM modello è scritto in YAML, quindi anche la definizione della macchina a stati è in YAML. Per dichiarare una definizione di macchina a stati in linea in JSON, scrivi il file modello in JSON. AWS SAM

YAML

```
MySampleStateMachine:
  Type: AWS::Serverless::StateMachine
  Properties:
    Definition:
      StartAt: MyLambdaState
      States:
        MyLambdaState:
          Type: Task
          Resource: arn:aws:lambda:us-east-1:123456123456:function:my-sample-lambda-app
          End: true
    Role: arn:aws:iam::123456123456:role/service-role/my-sample-role
    Tracing:
      Enabled: true
```

EventSource

L'oggetto che descrive l'origine degli eventi che attivano la macchina a stati. Ogni evento è costituito da un tipo e da un insieme di proprietà che dipendono da quel tipo. Per ulteriori informazioni sulle proprietà di ciascuna fonte di eventi, vedete il sottoargomento corrispondente a quel tipo.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Properties: Schedule | ScheduleV2 | CloudWatchEvent | EventBridgeRule | Api
Type: String
```

Properties

`Properties`

Un oggetto che descrive le proprietà di questa mappatura degli eventi. L'insieme di proprietà deve essere conforme a quanto definito. `Type`

Tipo: Schedule | ScheduleV2 | | | Api CloudWatchEventEventBridgeRule

Obbligatorio: sì

CloudFormation compatibilità: Questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Type`

Il tipo di evento,

Valori validi:`Api`,`Schedule`,`ScheduleV2`,`CloudWatchEvent`, `EventBridgeRule`

Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

"Hello, World!"

Di seguito è riportato un esempio di evento del `API` tipo.

YAML

```
ApiEvent:
  Type: Api
  Properties:
    Method: get
    Path: /group/{user}
    RestApiId:
      Ref: MyApi
```

Api

L'oggetto che descrive un tipo di origine `Api` dell'evento. Se viene definita una AWS::Serverless::Api risorsa, i valori del percorso e del metodo devono corrispondere a un'operazione nella definizione OpenAPI dell'API.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  Auth: ApiStateMachineAuth
  Method: String
  Path: String
  RestApiId: String
  UnescapeMappingTemplate: Boolean
```

Properties

`Auth`

La configurazione di autorizzazione per questa API, percorso e metodo.

Utilizzate questa proprietà per sovrascrivere l'`DefaultAuthorizer`impostazione dell'API per un singolo percorso, quando non `DefaultAuthorizer` è specificato nessuno, o per sovrascrivere l'impostazione predefinita`ApiKeyRequired`.

Tipo: ApiStateMachineAuth

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Method`

Il metodo HTTP per cui viene richiamata questa funzione.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Path`

Il percorso URI per il quale viene richiamata questa funzione. Il valore deve iniziare con`/`.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`RestApiId`

L'identificatore di una `RestApi` risorsa, che deve contenere un'operazione con il percorso e il metodo indicati. In genere, è impostato per fare riferimento a una AWS::Serverless::Api risorsa definita in questo modello.

Se non definisci questa proprietà, AWS SAM crea una AWS::Serverless::Api risorsa predefinita utilizzando un `OpenApi` documento generato. Tale risorsa contiene un'unione di tutti i percorsi e i metodi definiti dagli `Api` eventi nello stesso modello che non specificano a`RestApiId`.

Questa proprietà non può fare riferimento a una AWS::Serverless::Api risorsa definita in un altro modello.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`UnescapeMappingTemplate`

Rimuove le virgolette singole, sostituendole `\'` con `'`, sull'input passato alla macchina a stati. Da utilizzare quando l'input contiene virgolette singole.

> **Note**
>
> Se è impostato su `False` e l'input contiene virgolette singole, si verificherà un errore.

Tipo: Booleano

Obbligatorio: no

Impostazione predefinita: False

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

ApiEvent

Di seguito è riportato un esempio di evento di `Api` questo tipo.

YAML

```
Events:
  ApiEvent:
    Type: Api
    Properties:
      Path: /path
      Method: get
```

ApiStateMachineAuth

Configura l'autorizzazione a livello di evento, per un'API, un percorso e un metodo specifici.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
ApiKeyRequired: Boolean
AuthorizationScopes: List
Authorizer: String
ResourcePolicy: ResourcePolicyStatement
```

Properties

`ApiKeyRequired`

Richiede una chiave API per questa API, percorso e metodo.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AuthorizationScopes`

Gli ambiti di autorizzazione da applicare a questa API, percorso e metodo.

Gli ambiti specificati sostituiranno tutti gli ambiti applicati dalla `DefaultAuthorizer` proprietà, se l'hai specificata.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`Authorizer`

`Authorizer`Per una macchina a stati specifica.

Se hai specificato un autorizzatore globale per l'API e desideri rendere pubblica questa macchina a stati, sostituisci l'autorizzatore globale impostando su. `Authorizer` `NONE`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`ResourcePolicy`

Configura la politica delle risorse per questa API e questo percorso.

Tipo: ResourcePolicyStatement

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

StateMachine-Autenticazione

L'esempio seguente specifica l'autorizzazione a livello di macchina a stati.

YAML

```
Auth:
  ApiKeyRequired: true
  Authorizer: NONE
```

ResourcePolicyStatement

Configura una politica delle risorse per tutti i metodi e i percorsi di un'API. Per ulteriori informazioni sulle politiche delle risorse, consulta Controllare l'accesso a un'API con le politiche delle risorse di API Gateway nella Guida per sviluppatori di API Gateway.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
AwsAccountBlacklist: List
AwsAccountWhitelist: List
CustomStatements: List
IntrinsicVpcBlacklist: List
IntrinsicVpcWhitelist: List
IntrinsicVpceBlacklist: List
IntrinsicVpceWhitelist: List
IpRangeBlacklist: List
IpRangeWhitelist: List
SourceVpcBlacklist: List
SourceVpcWhitelist: List
```

Properties

`AwsAccountBlacklist`

Gli AWS account da bloccare.

Tipo: elenco di String

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`AwsAccountWhitelist`

Gli AWS account da consentire. Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi nella parte inferiore di questa pagina.

Tipo: elenco di String

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`CustomStatements`

Un elenco di istruzioni personalizzate sulle politiche relative alle risorse da applicare a questa API. Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi nella parte inferiore di questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IntrinsicVpcBlacklist`

L'elenco dei cloud privati virtuali (VPCs) da bloccare, in cui ogni VPC è specificato come riferimento, ad esempio un riferimento dinamico o una funzione `Ref` intrinseca. Per un esempio di utilizzo di questa proprietà, consultate la sezione Esempi in fondo a questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IntrinsicVpcWhitelist`

L'elenco di VPCs to allow, in cui ogni VPC è specificato come riferimento, ad esempio un riferimento dinamico o una funzione `Ref` intrinseca.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`IntrinsicVpceBlacklist`

L'elenco degli endpoint VPC da bloccare, in cui ogni endpoint VPC è specificato come riferimento, ad esempio un riferimento dinamico o la funzione intrinseca. `Ref`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

`IntrinsicVpceWhitelist`

L'elenco degli endpoint VPC da consentire, in cui ogni endpoint VPC è specificato come riferimento, ad esempio un riferimento dinamico o la funzione intrinseca. `Ref` Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi in fondo a questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IpRangeBlacklist`

Gli indirizzi IP o gli intervalli di indirizzi da bloccare. Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi nella parte inferiore di questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`IpRangeWhitelist`

Gli indirizzi IP o gli intervalli di indirizzi da consentire.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`SourceVpcBlacklist`

Gli endpoint VPC o VPC di origine da bloccare. I nomi degli endpoint VPC di origine devono iniziare con `"vpc-"` e i nomi degli endpoint VPC di origine devono iniziare con. `"vpce-"` Per un esempio di utilizzo di questa proprietà, consulta la sezione Esempi nella parte inferiore di questa pagina.

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`SourceVpcWhitelist`

Gli endpoint VPC o VPC di origine da consentire. I nomi degli endpoint VPC di origine devono iniziare con `"vpc-"` e i nomi degli endpoint VPC di origine devono iniziare con. `"vpce-"`

Tipo: elenco

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un equivalente. CloudFormation

Esempi

Esempio di politica delle risorse

L'esempio seguente blocca due indirizzi IP e un VPC di origine e consente un AWS account.

YAML

```
Auth:
  ResourcePolicy:
    CustomStatements: [{
                        "Effect": "Allow",
                        "Principal": "*",
                        "Action": "execute-api:Invoke",
                        "Resource": "execute-api:/Prod/GET/pets",
                        "Condition": {
                          "IpAddress": {
                            "aws:SourceIp": "1.2.3.4"
                          }
                        }
                      }]
    IpRangeBlacklist:
      - "10.20.30.40"
      - "1.2.3.4"
```

```
    SourceVpcBlacklist:
      - "vpce-1a2b3c4d"
    AwsAccountWhitelist:
      - "111122223333"
    IntrinsicVpcBlacklist:
      - "{{resolve:ssm:SomeVPCReference:1}}"
      - !Ref MyVPC
    IntrinsicVpceWhitelist:
      - "{{resolve:ssm:SomeVPCEReference:1}}"
      - !Ref MyVPCE
```

CloudWatchEvent

L'oggetto che descrive un tipo di origine `CloudWatchEvent` dell'evento.

AWS Serverless Application Model (AWS SAM) genera una AWS::Events::Rulerisorsa quando viene impostato questo tipo di evento.

Nota importante: EventBridgeRule è il tipo di sorgente di eventi preferito da utilizzare, anzichéCloudWatchEvent. EventBridgeRulee CloudWatchEvent utilizza lo stesso servizio, API e CloudFormation risorse sottostanti. Tuttavia, AWS SAM aggiungerà il supporto per le nuove funzionalità solo aEventBridgeRule.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
EventBusName: String
Input: String
InputPath: String
Pattern: EventPattern
```

Properties

`EventBusName`

Il bus evento da associare a questa regola. Se omettete questa proprietà, AWS SAM utilizza il bus di eventi predefinito.

- Tipo: stringa

Obbligatorio: no

Predefinito: bus eventi predefinito

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventBusName` proprietà di una `AWS::Events::Rule` risorsa.

`Input`

Testo JSON valido passato al target. Se utilizzi questa proprietà, al target non viene passata alcuna informazione sul testo dell'evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Input` proprietà di una `AWS::Events::Rule Target` risorsa.

`InputPath`

Quando non vuoi passare l'intero evento corrispondente alla destinazione, usa la `InputPath` proprietà per descrivere quale parte dell'evento trasmettere.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `InputPath` proprietà di una `AWS::Events::Rule Target` risorsa.

`Pattern`

Descrive quali eventi vengono instradati nella destinazione specificata. Per ulteriori informazioni, consulta Events and Event Patterns EventBridge nella Amazon EventBridge User Guide.

Tipo: EventPattern

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventPattern` proprietà di una `AWS::Events::Rule` risorsa.

Esempi

CloudWatchEvent

Di seguito è riportato un esempio di tipo di sorgente di `CloudWatchEvent` eventi.

YAML

```
CWEvent:
  Type: CloudWatchEvent
  Properties:
    Input: '{"Key": "Value"}'
    Pattern:
      detail:
        state:
          - running
```

EventBridgeRule

L'oggetto che descrive un tipo di origine di `EventBridgeRule` eventi, che imposta la tua macchina a stati come destinazione per una EventBridge regola Amazon. Per ulteriori informazioni, consulta What Is Amazon EventBridge? nella Amazon EventBridge User Guide.

AWS SAM genera una AWS::Events::Rulerisorsa quando viene impostato questo tipo di evento.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  DeadLetterConfig: DeadLetterConfig
  EventBusName: String
  Input: String
  InputPath: String
  InputTransformer: InputTransformer
  Pattern: EventPattern
  RetryPolicy: RetryPolicy
  RuleName: String
  State: String
  Target: Target
```

Properties

`DeadLetterConfig`

Configura la coda Amazon Simple Queue Service (Amazon SQS) a EventBridge cui invia gli eventi dopo una chiamata di destinazione non riuscita. L'invocazione può fallire, ad esempio, quando si invia un evento a una funzione Lambda che non esiste o quando EventBridge le autorizzazioni sono insufficienti per richiamare la funzione Lambda. Per ulteriori informazioni, consulta la politica di riprova degli eventi e l'utilizzo delle code di lettere morte nella Amazon User Guide. EventBridge

Tipo: DeadLetterConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `DeadLetterConfig` proprietà del tipo di dati. `AWS::Events::Rule Target` La AWS SAM versione di questa proprietà include sottoproprietà aggiuntive, nel caso in cui si desideri AWS SAM creare automaticamente la coda di lettere non scritte.

`EventBusName`

Il bus evento da associare a questa regola. Se si omette questa proprietà, AWS SAM utilizza il bus eventi predefinito.

- Tipo: stringa

Obbligatorio: no

Predefinito: bus eventi predefinito

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventBusName` proprietà di una `AWS::Events::Rule` risorsa.

`Input`

Testo JSON valido passato al target. Se utilizzi questa proprietà, al target non viene passata alcuna informazione sul testo dell'evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Input` proprietà di una `AWS::Events::Rule Target` risorsa.

`InputPath`

Quando non vuoi passare l'intero evento corrispondente alla destinazione, usa la `InputPath` proprietà per descrivere quale parte dell'evento trasmettere.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `InputPath` proprietà di una `AWS::Events::Rule Target` risorsa.

`InputTransformer`

Impostazioni che consentono di fornire input personalizzato a un target in base a determinati dati di eventi. Puoi estrarre una o più coppie chiave-valore dall'evento e quindi utilizzare tali dati per l'invio di input personalizzato al target. Per ulteriori informazioni, consulta Amazon EventBridge Input Transformation nella Amazon EventBridge User Guide.

Tipo: InputTransformer

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `InputTransformer` proprietà di un tipo di `AWS::Events::Rule Target` dati.

`Pattern`

Descrive quali eventi vengono instradati nella destinazione specificata. Per ulteriori informazioni, consulta Events and Event Patterns EventBridge nella Amazon EventBridge User Guide.

Tipo: EventPattern

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EventPattern` proprietà di una `AWS::Events::Rule` risorsa.

`RetryPolicy`

Un oggetto `RetryPolicy` che include informazioni sulle impostazioni dei criteri di nuovi tentativi. Per ulteriori informazioni, consulta la politica di riprova degli eventi e l'utilizzo delle code di lettere morte nella Amazon User Guide. EventBridge

Tipo: RetryPolicy

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RetryPolicy` proprietà del tipo di dati. `AWS::Events::Rule Target`

`RuleName`

Nome della regola .

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::Events::Rule` risorsa.

`State`

Lo stato della regola.

Valori validi: `[ DISABLED | ENABLED ]`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `State` proprietà di una `AWS::Events::Rule` risorsa.

`Target`

La AWS risorsa che EventBridge richiama quando viene attivata una regola. È possibile utilizzare questa proprietà per specificare l'ID logico della destinazione. Se questa proprietà non è specificata, AWS SAM genera l'ID logico della destinazione.

Tipo: Target

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `Targets` proprietà di una `AWS::Events::Rule` risorsa. La AWS SAM versione di questa proprietà consente solo di specificare l'ID logico di una singola destinazione.

Esempi

EventBridgeRule

Di seguito è riportato un esempio di tipo di origine di `EventBridgeRule` evento.

YAML

```
EBRule:
  Type: EventBridgeRule
  Properties:
    Input: '{"Key": "Value"}'
    Pattern:
      detail:
        state:
          - terminated
```

DeadLetterConfig

L'oggetto utilizzato per specificare la coda Amazon Simple Queue Service (Amazon SQS) in EventBridge cui invia gli eventi dopo una chiamata di destinazione non riuscita. L'invocazione può fallire, ad esempio, quando si invia un evento a una macchina a stati che non esiste o le autorizzazioni sono insufficienti per richiamare la macchina a stati. Per ulteriori informazioni, consulta la politica di riprova degli eventi e l'utilizzo delle code di lettere morte nella Amazon User Guide. EventBridge

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  Arn: String
  QueueLogicalId: String
  Type: String
```

Properties

`Arn`

L'Amazon Resource Name (ARN) della coda Amazon SQS specificato come destinazione per la coda di lettere non scritte.

> **Note**
>
> Specificare la proprietà o la `Type` proprietà, ma non entrambe. `Arn`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Arn` proprietà del tipo di `AWS::Events::Rule` `DeadLetterConfig` dati.

`QueueLogicalId`

`Type`Viene specificato il nome personalizzato della coda di lettere morte che AWS SAM crea if.

> **Note**
>
> Se la `Type` proprietà non è impostata, viene ignorata.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Type`

Il tipo di coda. Quando questa proprietà è impostata, crea AWS SAM automaticamente una coda di lettere morte e allega la politica basata sulle risorse necessaria per concedere l'autorizzazione alla risorsa delle regole di inviare eventi alla coda.

> **Note**
>
> Specificate la proprietà o la proprietà, ma non entrambe. `Type` `Arn`

Valori validi: `SQS`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

DeadLetterConfig

DeadLetterConfig

YAML

```
DeadLetterConfig:
  Type: SQS
  QueueLogicalId: MyDLQ
```

Target

Configura la AWS risorsa che EventBridge richiama quando viene attivata una regola.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  Id: String
```

Properties

`Id`

L'ID logico della destinazione.

Il valore di `Id` può includere caratteri alfanumerici, punti (`.`), trattini () e caratteri di - sottolineatura (). _

- Tipo: stringa

  Obbligatorio: sì

  CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Id` proprietà del tipo di dati. `AWS::Events::Rule Target`

Esempi

Target

YAML

```
EBRule:
  Type: EventBridgeRule
  Properties:
    Target:
      Id: MyTarget
```

Schedule

L'oggetto che descrive un tipo di origine `Schedule` dell'evento, che imposta la macchina a stati come destinazione di una EventBridge regola che si attiva in base a una pianificazione. Per ulteriori informazioni, consulta What Is Amazon EventBridge? nella Amazon EventBridge User Guide.

AWS Serverless Application Model (AWS SAM) genera una AWS::Events::Rulerisorsa quando viene impostato questo tipo di evento.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
  DeadLetterConfig: DeadLetterConfig
  Description: String
  Enabled: Boolean
  Input: String
  Name: String
```

```
  RetryPolicy: RetryPolicy
  RoleArn: String
  Schedule: String
  State: String
  Target: Target
```

Properties

`DeadLetterConfig`

Configura la coda Amazon Simple Queue Service (Amazon SQS) a EventBridge cui invia gli eventi dopo una chiamata di destinazione non riuscita. L'invocazione può fallire, ad esempio, quando si invia un evento a una funzione Lambda che non esiste o quando EventBridge le autorizzazioni sono insufficienti per richiamare la funzione Lambda. Per ulteriori informazioni, consulta la politica di riprova degli eventi e l'utilizzo delle code di lettere morte nella Amazon User Guide. EventBridge

Tipo: DeadLetterConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `DeadLetterConfig` proprietà del tipo di dati. `AWS::Events::Rule Target` La AWS SAM versione di questa proprietà include sottoproprietà aggiuntive, nel caso in cui si desideri AWS SAM creare automaticamente la coda di lettere morte.

`Description`

Una descrizione della regola.

-Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una risorsa. `AWS::Events::Rule`

`Enabled`

Indica se la regola è abilitata.

Per disabilitare la regola, imposta questa proprietà su`false`.

> **Note**
>
> Specificate la `State` proprietà `Enabled` o, ma non entrambe.

Tipo: Booleano

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `State` proprietà di una `AWS::Events::Rule` risorsa. Se questa proprietà è impostata su `true` allora AWS SAM passa`ENABLED`, altrimenti passa`DISABLED`.

`Input`

Testo JSON valido passato al target. Se utilizzi questa proprietà, al target non viene passata alcuna informazione sul testo dell'evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Input` proprietà di una `AWS::Events::Rule Target` risorsa.

`Name`

Nome della regola . Se non specifichi un nome, CloudFormation genera un ID fisico univoco e utilizza tale ID per il nome della regola.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::Events::Rule` risorsa.

`RetryPolicy`

Un oggetto `RetryPolicy` che include informazioni sulle impostazioni dei criteri di nuovi tentativi. Per ulteriori informazioni, consulta la politica di riprova degli eventi e l'utilizzo delle code di lettere morte nella Amazon User Guide. EventBridge

Tipo: RetryPolicy

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RetryPolicy` proprietà del tipo di dati. `AWS::Events::Rule Target`

`RoleArn`

L'ARN del ruolo IAM che EventBridge Scheduler utilizzerà per la destinazione quando viene richiamata la pianificazione.

Tipo: RoleArn

Obbligatorio: No. Se non viene fornito, verrà creato e utilizzato un nuovo ruolo.

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RoleArn` proprietà del tipo di `AWS::Scheduler::Schedule Target` dati.

`Schedule`

L'espressione di programmazione che determina il momento e la frequenza di esecuzione della regola. Per ulteriori informazioni, consulta Pianificazione di espressioni per regole.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ScheduleExpression` proprietà di una `AWS::Events::Rule` risorsa.

`State`

Lo stato della regola.

Valori accettati: `DISABLED | ENABLED`

> **Note**
>
> Specificate la `State` proprietà `Enabled` o, ma non entrambe.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `State` proprietà di una `AWS::Events::Rule` risorsa.

`Target`

La AWS risorsa che EventBridge richiama quando viene attivata una regola. È possibile utilizzare questa proprietà per specificare l'ID logico della destinazione. Se questa proprietà non è specificata, AWS SAM genera l'ID logico della destinazione.

Tipo: Target

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla `Targets` proprietà di una `AWS::Events::Rule` risorsa. La AWS SAM versione di questa proprietà consente solo di specificare l'ID logico di una singola destinazione.

Esempi

CloudWatch Pianifica evento

CloudWatch Esempio di evento di pianificazione

YAML

```
CWSchedule:
  Type: Schedule
  Properties:
    Schedule: 'rate(1 minute)'
    Name: TestSchedule
    Description: test schedule
    Enabled: false
```

DeadLetterConfig

L'oggetto utilizzato per specificare la coda Amazon Simple Queue Service (Amazon SQS) in EventBridge cui invia gli eventi dopo una chiamata di destinazione non riuscita. L'invocazione può fallire, ad esempio, quando si invia un evento a una macchina a stati che non esiste o le autorizzazioni sono insufficienti per richiamare la macchina a stati. Per ulteriori informazioni, consulta la politica di riprova degli eventi e l'utilizzo delle code di lettere morte nella Amazon User Guide. EventBridge

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Arn: String
QueueLogicalId: String
Type: String
```

Properties

`Arn`

L'Amazon Resource Name (ARN) della coda Amazon SQS specificato come destinazione per la coda di lettere non scritte.

> **Note**
>
> Specificare la proprietà o la `Type` proprietà, ma non entrambe. `Arn`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Arn` proprietà del tipo di `AWS::Events::Rule` `DeadLetterConfig` dati.

`QueueLogicalId`

`Type`Viene specificato il nome personalizzato della coda di lettere morte che AWS SAM crea if.

> **Note**
>
> Se la `Type` proprietà non è impostata, viene ignorata.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`Type`

Il tipo di coda. Quando questa proprietà è impostata, crea AWS SAM automaticamente una coda di lettere morte e allega la politica basata sulle risorse necessaria per concedere l'autorizzazione alla risorsa delle regole di inviare eventi alla coda.

> **Note**
>
> Specificate la proprietà o la proprietà, ma non entrambe. `Type` `Arn`

Valori validi: `SQS`

Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

Esempi

DeadLetterConfig

DeadLetterConfig

YAML

```
DeadLetterConfig:
  Type: SQS
  QueueLogicalId: MyDLQ
```

Target

Configura la AWS risorsa che EventBridge richiama quando viene attivata una regola.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```
Id: String
```

Properties

`Id`

L'ID logico della destinazione.

Il valore di `Id` può includere caratteri alfanumerici, punti (`.`), trattini () e caratteri di - sottolineatura (). _

Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Id` proprietà del tipo di dati. `AWS::Events::Rule Target`

Esempi

Target

YAML

```
EBRule:
  Type: Schedule
  Properties:
    Target:
      Id: MyTarget
```

ScheduleV2

L'oggetto che descrive un tipo di origine `ScheduleV2` dell'evento, che imposta la tua macchina a stati come destinazione di un evento Amazon EventBridge Scheduler che si attiva in base a una pianificazione. Per ulteriori informazioni, consulta Cos'è Amazon EventBridge Scheduler? nella Guida per l'utente di EventBridge Scheduler.

AWS Serverless Application Model (AWS SAM) genera una AWS::Scheduler::Schedulerisorsa quando viene impostato questo tipo di evento.

Sintassi

Per dichiarare questa entità nel tuo modello AWS Serverless Application Model (AWS SAM), usa la seguente sintassi.

YAML

```yaml
DeadLetterConfig: DeadLetterConfig
Description: String
EndDate: String
FlexibleTimeWindow: FlexibleTimeWindow
GroupName: String
Input: String
KmsKeyArn: String
Name: String
OmitName: Boolean
PermissionsBoundary: String
RetryPolicy: RetryPolicy
RoleArn: String
ScheduleExpression: String
ScheduleExpressionTimezone: String
StartDate: String
State: String
```

Properties

`DeadLetterConfig`

Configura la coda Amazon Simple Queue Service (Amazon SQS) a EventBridge cui invia gli eventi dopo una chiamata di destinazione non riuscita. L'invocazione può fallire, ad esempio, quando si invia un evento a una funzione Lambda che non esiste o quando EventBridge le autorizzazioni sono insufficienti per richiamare la funzione Lambda. Per ulteriori informazioni, consulta Configurazione di una coda di lettere non scritte per Scheduler nella Scheduler User Guide. EventBridge EventBridge

Tipo: DeadLetterConfig

Obbligatorio: no

CloudFormation compatibilità: questa proprietà è simile alla proprietà del `DeadLetterConfig` tipo di dati. `AWS::Scheduler::Schedule Target` La AWS SAM versione di questa proprietà

include sottoproprietà aggiuntive, nel caso in cui si desideri AWS SAM creare automaticamente la coda di lettere non scritte.

`Description`

Una descrizione della pianificazione.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Description` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`EndDate`

La data, in UTC, prima della quale la pianificazione può richiamare il suo obiettivo. A seconda dell'espressione di ricorrenza della pianificazione, le chiamate potrebbero interrompersi nel momento in cui specifichi EndDate o prima.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `EndDate` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`FlexibleTimeWindow`

Consente la configurazione di una finestra all'interno della quale è possibile richiamare una pianificazione.

Tipo: FlexibleTimeWindow

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `FlexibleTimeWindow` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`GroupName`

Il nome del gruppo di pianificazioni da associare a questa pianificazione. Se non è definito, viene utilizzato il gruppo predefinito.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `GroupName` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`Input`

Testo JSON valido passato al target. Se utilizzi questa proprietà, al target non viene passata alcuna informazione sul testo dell'evento.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Input` proprietà di una `AWS::Scheduler::Schedule Target` risorsa.

`KmsKeyArn`

L'ARN per una chiave KMS che verrà utilizzata per crittografare i dati dei clienti.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `KmsKeyArn` proprietà di una risorsa. `AWS::Scheduler::Schedule`

`Name`

Il nome della pianificazione. Se non specifichi un nome, AWS SAM genera un nome nel formato *`StateMachine-Logical-IDEvent-Source-Name`* e utilizza quell'ID per il nome della pianificazione.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `Name` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`OmitName`

Per impostazione predefinita, AWS SAM genera e utilizza un nome di pianificazione nel formato di*`<State-machine-logical-ID><event-source-name>`*. Imposta questa proprietà in `true` modo da CloudFormation generare un ID fisico univoco e utilizzalo invece per il nome della pianificazione.

Tipo: Booleano

Obbligatorio: no

Default: `false`

CloudFormation compatibilità: questa proprietà è unica AWS SAM e non ha un CloudFormation equivalente.

`PermissionsBoundary`

L'ARN della policy utilizzata per impostare il limite delle autorizzazioni per il ruolo.

> **Note**
>
> Se `PermissionsBoundary` definito, AWS SAM applicherà gli stessi limiti al ruolo IAM di destinazione della pianificazione dello scheduler.

Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `PermissionsBoundary` proprietà di una `AWS::IAM::Role` risorsa.

`RetryPolicy`

Un oggetto `RetryPolicy` che include informazioni sulle impostazioni dei criteri di nuovi tentativi.

Tipo: RetryPolicy

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RetryPolicy` proprietà del tipo di `AWS::Scheduler::Schedule Target` dati.

`RoleArn`

L'ARN del ruolo IAM che EventBridge Scheduler utilizzerà per la destinazione quando viene richiamata la pianificazione.

Tipo: RoleArn

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `RoleArn` proprietà del tipo di dati. `AWS::Scheduler::Schedule Target`

`ScheduleExpression`

L'espressione di pianificazione che determina quando e con che frequenza viene eseguita la pianificazione.

- Tipo: stringa

Obbligatorio: sì

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ScheduleExpression` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`ScheduleExpressionTimezone`

Il fuso orario in cui viene valutata l'espressione di pianificazione.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `ScheduleExpressionTimezone` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`StartDate`

La data, in UTC, dopo la quale la pianificazione può iniziare a richiamare un obiettivo. A seconda dell'espressione di ricorrenza della pianificazione, le chiamate potrebbero interrompersi nel momento in cui specifichi StartDate o dopo.

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `StartDate` proprietà di una `AWS::Scheduler::Schedule` risorsa.

`State`

Lo stato della pianificazione.

Valori accettati: `DISABLED | ENABLED`

- Tipo: stringa

Obbligatorio: no

CloudFormation compatibilità: questa proprietà viene passata direttamente alla `State` proprietà di una `AWS::Scheduler::Schedule` risorsa.

Esempi

Esempio di base di definizione di una risorsa ScheduleV2

```
StateMachine:
  Type: AWS::Serverless::StateMachine
  Properties:
    Name: MyStateMachine
    Events:
      ScheduleEvent:
        Type: ScheduleV2
        Properties:
          ScheduleExpression: "rate(1 minute)"
      ComplexScheduleEvent:
        Type: ScheduleV2
        Properties:
          ScheduleExpression: rate(1 minute)
          FlexibleTimeWindow:
            Mode: FLEXIBLE
            MaximumWindowInMinutes: 5
          StartDate: '2022-12-28T12:00:00.000Z'
          EndDate: '2023-01-28T12:00:00.000Z'
          ScheduleExpressionTimezone: UTC
          RetryPolicy:
            MaximumRetryAttempts: 5
            MaximumEventAgeInSeconds: 300
          DeadLetterConfig:
            Type: SQS
    DefinitionUri:
      Bucket: sam-sam-s3-demo-bucket
      Key: my-state-machine.asl.json
      Version: 3
    Policies:
      - LambdaInvokePolicy:
          FunctionName: !Ref MyFunction
```

CloudFormation Risorse generate per AWS SAM

Questa sezione fornisce dettagli sulle CloudFormation risorse che vengono create durante l' AWS SAM elaborazione del AWS modello. L'insieme di CloudFormation risorse che AWS SAM genera varia a seconda degli scenari specificati. Uno scenario è la combinazione di AWS SAM risorse e proprietà specificata nel file modello. È possibile fare riferimento alle CloudFormation risorse generate altrove all'interno del file modello, in modo analogo a come si fa riferimento alle risorse dichiarate esplicitamente nel file modello.

Ad esempio, se specifichi una `AWS::Serverless::Function` risorsa nel file AWS SAM modello, genera AWS SAM sempre una risorsa di `AWS::Lambda::Function` base. Se specificate anche la `AutoPublishAlias` proprietà opzionale, genera `AWS::Lambda::Alias` e `AWS::Lambda::Version` risorse AWS SAM aggiuntive.

Questa sezione elenca gli scenari e le CloudFormation risorse che generano e mostra come fare riferimento alle CloudFormation risorse generate nel file AWS SAM modello.

Riferimento alle risorse generate CloudFormation

Sono disponibili due opzioni per fare riferimento CloudFormation alle risorse generate all'interno del file AWS SAM modello, in base `LogicalId` o tramite proprietà referenziabili.

Riferimento alle risorse generate da CloudFormation LogicalId

Ciascuna delle CloudFormation risorse AWS SAM generate ha un `LogicalId` identificatore alfanumerico (A-Z, a-z, 0-9) univoco all'interno di un file modello. AWS SAM utilizza le AWS SAM risorse `LogicalIds` del file modello per costruire le risorse che genera. `LogicalIds` CloudFormation Puoi utilizzare le informazioni `LogicalId` di una CloudFormation risorsa generata per accedere alle proprietà di quella risorsa all'interno del tuo file modello, proprio come faresti per una CloudFormation risorsa che hai dichiarato esplicitamente. Per ulteriori informazioni su `LogicalIds` in CloudFormation e AWS SAM modelli, consulta Resources in the AWS CloudFormation User Guide.

> **Note**
>
> Alcune risorse generate includono un valore hash univoco per evitare conflitti tra namespace. `LogicalIds` Alcune `LogicalIds` di queste risorse vengono derivate al momento della creazione dello stack. È possibile recuperarle solo dopo aver creato lo stack utilizzando

Console di gestione AWS AWS CLI, o uno dei. AWS SDKs Non è consigliabile fare riferimento a queste risorse `LogicalId` perché i valori hash potrebbero cambiare.

Riferimento alle CloudFormation risorse generate mediante proprietà referenziabili

Per alcune risorse generate, AWS SAM fornisce una proprietà referenziabile della risorsa. AWS SAM È possibile utilizzare questa proprietà per fare riferimento a una CloudFormation risorsa generata e alle relative proprietà all'interno del file AWS SAM modello.

> **Note**
>
> Non tutte le CloudFormation risorse generate hanno proprietà referenziabili. Per tali risorse, è necessario utilizzare. `LogicalId`

Scenari di CloudFormation risorse generate

La tabella seguente riepiloga le AWS SAM risorse e le proprietà che costituiscono gli scenari che generano CloudFormation risorse. Gli argomenti della colonna Scenari forniscono dettagli sulle CloudFormation risorse aggiuntive AWS SAM generate per quello scenario.

AWS SAM risorsa	CloudFormation Risorsa di base	Scenari
`AWS::Serverless::Api`	`AWS::ApiGateway::RestApi`	• DomainName la proprietà è specificata • UsagePlan la proprietà è specificata
`AWS::Serverless::Application`	`AWS::CloudFormation::Stack`	• Oltre alla generazione della CloudFormation risorsa di base, non esistono scenari aggiuntivi per questa risorsa serverless.
`AWS::Serverless::C`	`AWS::Lambda::Capac`	• OperatorRole la proprietà non è specificata

AWS SAM risorsa	CloudFormation Risorsa di base	Scenari
apacityProvider	ityProvider	
AWS::Serverless::Function	AWS::Lambda::Function	• AutoPublishAlias la proprietà è specificata • La proprietà del ruolo non è specificata • DeploymentPreference la proprietà è specificata • Viene specificata un'origine di eventi Api • Viene specificata un'origine dell'evento HttpApi • Viene specificata una fonte di eventi di streaming • Viene specificata una fonte di eventi Event Bridge (o Event Bus) • Viene specificata un'origine dell'evento IotRule • OnSuccess (o OnFailure) la proprietà è specificata per gli eventi Amazon SNS • OnSuccess (o OnFailure) la proprietà è specificata per gli eventi Amazon SQS
AWS::Serverless::HttpApi	AWS::ApiGatewayV2::Api	• StageName la proprietà è specificata • StageName la proprietà non è specificata • DomainName la proprietà è specificata

AWS SAM risorsa	CloudFormation Risorsa di base	Scenari
AWS::Serverless::LayerVersion	AWS::Lambda::LayerVersion	• Oltre alla generazione della CloudFormation risorsa di base, non esistono scenari aggiuntivi per questa risorsa serverless.
AWS::Serverless::SimpleTable	AWS::DynamoDB::Table	• Oltre alla generazione della CloudFormation risorsa di base, non esistono scenari aggiuntivi per questa risorsa serverless.
AWS::Serverless::StateMachine	AWS::StepFunctions::StateMachine	• La proprietà del ruolo non è specificata • Viene specificata un'origine di eventi API • Viene specificata una fonte di eventi Event Bridge (o Event Bus)

Argomenti

- CloudFormation risorse generate quando AWS::Serverless::Api è specificato
- CloudFormation risorse generate quando AWS::Serverless::Application è specificato
- CloudFormation risorse generate quando AWS::Serverless::CapacityProvider è specificato
- CloudFormation risorse generate quando si specifica AWS::Serverless::Connector
- CloudFormation risorse generate quando AWS::Serverless::Function è specificato
- CloudFormation risorse generate quando AWS::Serverless::GraphQLApi è specificato
- CloudFormation risorse generate quando AWS::Serverless::HttpApi è specificato
- CloudFormation risorse generate quando AWS::Serverless::LayerVersion è specificato
- CloudFormation risorse generate quando AWS::Serverless::SimpleTable è specificato
- CloudFormation risorse generate quando AWS::Serverless::StateMachine è specificato

CloudFormation risorse generate quando AWS::Serverless::Api è specificato

Quando `AWS::Serverless::Api` viene specificato un, AWS Serverless Application Model (AWS SAM) genera sempre una CloudFormation risorsa di `AWS::ApiGateway::RestApi` base. Inoltre, genera sempre una `AWS::ApiGateway::Deployment` risorsa `AWS::ApiGateway::Stage` e una.

`AWS::ApiGateway::RestApi`

`LogicalId`: *`<api-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

`AWS::ApiGateway::Stage`

`LogicalId`: *`<api-LogicalId><stage-name>`*`Stage`

*`<stage-name>`*è la stringa su cui è impostata la `StageName` proprietà. Ad esempio, se si imposta `StageName` su`Gamma`, `LogicalId` è*`MyRestApiGamma`*`Stage`.

Proprietà referenziabile: *`<api-LogicalId>`*`.Stage`

`AWS::ApiGateway::Deployment`

`LogicalId`: *`<api-LogicalId>`*`Deployment`*`<sha>`*

*`<sha>`*è un valore hash unico che viene generato quando viene creato lo stack. Ad esempio, *`MyRestApi`*`Deployment`*`926eeb5ff1`*.

Proprietà referenziabile: *`<api-LogicalId>`*`.Deployment`

Oltre a queste CloudFormation risorse, quando `AWS::Serverless::Api` viene specificato, AWS SAM genera CloudFormation risorse aggiuntive per i seguenti scenari.

Scenari

- DomainName la proprietà è specificata
- UsagePlan la proprietà è specificata

DomainName la proprietà è specificata

Quando `AWS::Serverless::Api` viene specificata la `Domain` proprietà della proprietà di an, AWS SAM genera la `AWS::ApiGateway::DomainName` CloudFormation risorsa. `DomainName`

`AWS::ApiGateway::DomainName`

`LogicalId`: `ApiGatewayDomainName`*`<sha>`*

*`<sha>`*è un valore hash univoco che viene generato quando viene creato lo stack. Ad esempio: `ApiGatewayDomainName`*`926eeb5ff1`*.

Proprietà referenziabile: *`<api-LogicalId>`*`.DomainName`

UsagePlan la proprietà è specificata

Quando `AWS::Serverless::Api` viene specificata la `Auth` proprietà della proprietà di an, AWS SAM genera le seguenti CloudFormation risorse: `AWS::ApiGateway::UsagePlan``AWS::ApiGateway::UsagePlanKey`, e`AWS::ApiGateway::ApiKey`. `UsagePlan`

`AWS::ApiGateway::UsagePlan`

`LogicalId`: *`<api-LogicalId>`*`UsagePlan`

Proprietà referenziabile: *`<api-LogicalId>`*`.UsagePlan`

`AWS::ApiGateway::UsagePlanKey`

`LogicalId`: *`<api-LogicalId>`*`UsagePlanKey`

Proprietà referenziabile: *`<api-LogicalId>`*`.UsagePlanKey`

`AWS::ApiGateway::ApiKey`

`LogicalId`: *`<api-LogicalId>`*`ApiKey`

Proprietà referenziabile: *`<api-LogicalId>`*`.ApiKey`

CloudFormation risorse generate quando AWS::Serverless::Application è specificato

Quando `AWS::Serverless::Application` viene specificato un, AWS Serverless Application Model (AWS SAM) genera una CloudFormation risorsa di `AWS::CloudFormation::Stack` base.

`AWS::CloudFormation::Stack`

`LogicalId`: *`<application-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

CloudFormation risorse generate quando AWS::Serverless::CapacityProvider è specificato

Quando `AWS::Serverless::CapacityProvider` viene specificato un, AWS Serverless Application Model (AWS SAM) genera una CloudFormation risorsa di `AWS::Lambda::CapacityProvider` base.

`AWS::Lambda::CapacityProvider`

`LogicalId`: *`<capacityprovider-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

Oltre a questa CloudFormation risorsa, quando `AWS::Serverless::CapacityProvider` viene specificata, genera AWS SAM anche CloudFormation risorse per i seguenti scenari:

Scenari

- OperatorRole la proprietà non è specificata

OperatorRole la proprietà non è specificata

Quando la `OperatorRole` proprietà di an non *`AWS::Serverless::CapacityProvider`* è specificata, AWS SAM genera una `AWS::IAM::Role` CloudFormation risorsa con la politica `AWSLambdaManagedEC2ResourceOperator` gestita allegata.

`AWS::IAM::Role`

`LogicalId`: *`<capacityprovider-LogicalId>`*`OperatorRole`

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

CloudFormation risorse generate quando si specifica AWS::Serverless::Connector

> **Note**
>
> Quando definisci i connettori tramite la `Connectors` proprietà incorporata, questi vengono prima trasformati in una `AWS::Serverless::Connector` risorsa prima di generare tali risorse.

Quando si specifica una `AWS::Serverless::Connector` risorsa in un AWS SAM modello, AWS SAM genera le seguenti AWS CloudFormation risorse in base alle esigenze.

`AWS::IAM::ManagedPolicy`

`LogicalId`:*`<connector-LogicalId>`*`Policy`

Proprietà referenziabile: N/A (Per fare riferimento a questa CloudFormation risorsa, è necessario utilizzare il`LogicalId`.)

`AWS::SNS::TopicPolicy`

`LogicalId`:*`<connector-LogicalId>`*`TopicPolicy`

Proprietà referenziabile: N/A (Per fare riferimento a questa CloudFormation risorsa, è necessario utilizzare il.) `LogicalId`

`AWS::SQS::QueuePolicy`

`LogicalId`:*`<connector-LogicalId>`*`QueuePolicy`

Proprietà referenziabile: N/A (Per fare riferimento a questa CloudFormation risorsa, è necessario utilizzare il.) `LogicalId`

`AWS::Lambda::Permission`

`LogicalId`:*`<connector-LogicalId><permission>`*`LambdaPermission`

*`<permission>`*è un'autorizzazione specificata dalla proprietà. `Permissions` Ad esempio, `Write`.

Proprietà referenziabile: N/A (Per fare riferimento a questa CloudFormation risorsa, è necessario utilizzare il`LogicalId`.)

CloudFormation risorse generate quando AWS::Serverless::Function è specificato

Quando `AWS::Serverless::Function` viene specificato un, AWS Serverless Application Model (AWS SAM) crea sempre una CloudFormation risorsa di `AWS::Lambda::Function` base.

`AWS::Lambda::Function`

`LogicalId`: *`<function-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

Oltre a questa CloudFormation risorsa, quando `AWS::Serverless::Function` viene specificata, genera AWS SAM anche CloudFormation risorse per i seguenti scenari.

Scenari

- Proprietà delle funzioni principali
- Origini eventi
- Configurazione dell'evento

Proprietà delle funzioni principali

I seguenti scenari generano CloudFormation risorse in base alle proprietà delle funzioni principali:

La proprietà del ruolo non è specificata

Quando la `Role` proprietà di an non *`AWS::Serverless::Function`* è specificata, AWS SAM genera una `AWS::IAM::Role` CloudFormation risorsa.

`AWS::IAM::Role`

`LogicalId`: *`<function-LogicalId>`*`Role`

Proprietà referenziabile: N/A (è necessario utilizzare per fare riferimento `LogicalId` a questa risorsa) CloudFormation

AutoPublishAlias la proprietà è specificata

Quando `AWS::Serverless::Function` viene specificata la `AutoPublishAlias` proprietà di an, AWS SAM genera le seguenti CloudFormation risorse: `AWS::Lambda::Alias` and`AWS::Lambda::Version`.

`AWS::Lambda::Alias`

`LogicalId`: *`<function-LogicalId>`*`Alias`*`<alias-name>`*

*`<alias-name>`*è la stringa `AutoPublishAlias` impostata su. Ad esempio, se si imposta `AutoPublishAlias` su`live`, `LogicalId` è: *`MyFunction`* Alias*`live`*.

Proprietà referenziabile: *`<function-LogicalId>`*`.Alias`

`AWS::Lambda::Version`

`LogicalId`: *`<function-LogicalId>`*`Version`*`<sha>`*

*`<sha>`*è un valore hash unico che viene generato quando viene creato lo stack. Ad esempio, Version*`MyFunction`*. *`926eeb5ff1`*

Proprietà referenziabile: *`<function-LogicalId>`*`.Version`

Per ulteriori informazioni sulla `AutoPublishAlias` proprietà, vedere la sezione Proprietà di. AWS::Serverless::Function

DeploymentPreference la proprietà è specificata

Quando `AWS::Serverless::Function` viene specificata la `DeploymentPreference` proprietà di an, AWS SAM genera le seguenti CloudFormation risorse: `AWS::CodeDeploy::Application` and`AWS::CodeDeploy::DeploymentGroup`. Inoltre, se la `Role` proprietà dell'`DeploymentPreference`oggetto non è specificata, genera AWS SAM anche una `AWS::IAM::Role` CloudFormation risorsa.

`AWS::CodeDeploy::Application`

`LogicalId`: `ServerlessDeploymentApplication`

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

`AWS::CodeDeploy::DeploymentGroup`

`LogicalId`: *`<function-LogicalId>`*`DeploymentGroup`

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

`AWS::IAM::Role`

`LogicalId`: `CodeDeployServiceRole`

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

FunctionUrlConfig la proprietà è specificata

Quando la `FunctionUrlConfig` proprietà viene specificata, AWS SAM genera CloudFormation risorse diverse in base a`AuthType`.

Quando `AuthType: NONE` viene specificato, AWS SAM genera le seguenti CloudFormation risorse:

`AWS::Lambda::Permission`(Invoke Access)

`LogicalId`: *`<function-LogicalId>`*`URLInvokeAllowPublicAccess`

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

`AWS::Lambda::Permission`(Accesso pubblico)

`LogicalId`: *`<function-LogicalId>`*`UrlPublicPermissions`

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

`AWS::Lambda::Url`

`LogicalId`: `<function-LogicalId>Url`

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

Quando `AuthType: AWS_IAM` è specificato, genera solo: AWS SAM

`AWS::Lambda::Url`

`LogicalId`: `<function-LogicalId>Url`

Proprietà referenziabile: N/A (è necessario utilizzare per fare riferimento `LogicalId` a questa risorsa) CloudFormation

Per ulteriori informazioni sulla proprietà, vedere. `FunctionUrlConfig` FunctionUrlConfig

Origini eventi

I seguenti scenari generano CloudFormation risorse basate sulle fonti degli eventi:

Viene specificata un'origine di eventi Api

Quando la `Event` proprietà di an `AWS::Serverless::Function` è impostata su`Api`, ma la `RestApiId` proprietà non è specificata, AWS SAM genera la `AWS::ApiGateway::RestApi` CloudFormation risorsa.

`AWS::ApiGateway::RestApi`

`LogicalId`: `ServerlessRestApi`

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

Viene specificata un'origine dell'evento HttpApi

Quando la `Event` proprietà di an `AWS::Serverless::Function` è impostata su`HttpApi`, ma la `ApiId` proprietà non è specificata, AWS SAM genera la `AWS::ApiGatewayV2::Api` CloudFormation risorsa.

`AWS::ApiGatewayV2::Api`

`LogicalId`: `ServerlessHttpApi`

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

Viene specificata una fonte di eventi di streaming

Quando la `Event` proprietà di an `AWS::Serverless::Function` è impostata su uno dei tipi di streaming, AWS SAM genera la `AWS::Lambda::EventSourceMapping` CloudFormation risorsa. Questo vale per i seguenti tipi: `DynamoDB``Kinesis`,`MQ`,`MSK`, e`SQS`.

`AWS::Lambda::EventSourceMapping`

`LogicalId`: *`<function-LogicalId><event-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

Viene specificata una fonte di eventi Event Bridge (o Event Bus)

Quando la `Event` proprietà di an `AWS::Serverless::Function` è impostata su uno dei tipi di event bridge (o event bus), AWS SAM genera la `AWS::Events::Rule` CloudFormation risorsa. Questo vale per i seguenti tipi: `EventBridgeRuleSchedule`, e`CloudWatchEvents`.

`AWS::Events::Rule`

`LogicalId`: *`<function-LogicalId><event-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare per fare riferimento `LogicalId` a questa risorsa) CloudFormation

Viene specificata un'origine dell'evento IotRule

Quando la `Event` proprietà di an `AWS::Serverless::Function` è impostata su IoTRule, AWS SAM genera la `AWS::IoT::TopicRule` CloudFormation risorsa.

`AWS::IoT::TopicRule`

`LogicalId`: *`<function-LogicalId><event-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

Configurazione dell'evento

I seguenti scenari generano CloudFormation risorse in base alla configurazione degli eventi:

OnSuccess (o OnFailure) la proprietà è specificata per gli eventi Amazon SNS

Quando viene specificata la proprietà `OnSuccess` (or`OnFailure`) della `DestinationConfig` proprietà della `EventInvokeConfig` proprietà di an `AWS::Serverless::Function` e il tipo di destinazione è uguale `SNS` ma l'ARN di destinazione non è specificato, AWS SAM genera le seguenti CloudFormation risorse: `AWS::Lambda::EventInvokeConfig` and. `AWS::SNS::Topic`

`AWS::Lambda::EventInvokeConfig`

`LogicalId`: *`<function-LogicalId>`*`EventInvokeConfig`

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

`AWS::SNS::Topic`

`LogicalId`: (o) *`<function-LogicalId>`*`OnSuccessTopic` *`<function-LogicalId>`*`OnFailureTopic`

Proprietà referenziabile: *`<function-LogicalId>`*`.DestinationTopic`

Se entrambi `OnSuccess` `OnFailure` sono specificati per un evento Amazon SNS, per distinguere tra le risorse generate, devi utilizzare il. `LogicalId`

OnSuccess (o OnFailure) la proprietà è specificata per gli eventi Amazon SQS

Quando viene specificata la proprietà `OnSuccess` (or`OnFailure`) della `DestinationConfig` proprietà della `EventInvokeConfig` proprietà di an `AWS::Serverless::Function` e il tipo di destinazione è uguale `SQS` ma l'ARN di destinazione non è specificato, AWS SAM genera le seguenti CloudFormation risorse: `AWS::Lambda::EventInvokeConfig` and. `AWS::SQS::Queue`

`AWS::Lambda::EventInvokeConfig`

`LogicalId`: *`<function-LogicalId>`*`EventInvokeConfig`

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

`AWS::SQS::Queue`

`LogicalId`: (o) *`<function-LogicalId>`*`OnSuccessQueue` *`<function-LogicalId>`*`OnFailureQueue`

Proprietà referenziabile: *`<function-LogicalId>`*`.DestinationQueue`

Se entrambi `OnSuccess` `OnFailure` sono specificati per un evento Amazon SQS, per distinguere tra le risorse generate, è necessario utilizzare il. `LogicalId`

CloudFormation risorse generate quando AWS::Serverless::GraphQLApi è specificato

Quando specificate una `AWS::Serverless::GraphQLApi` risorsa in un modello AWS Serverless Application Model (AWS SAM), crea AWS SAM sempre le seguenti AWS CloudFormation risorse di base.

`AWS::AppSync::DataSource`

`LogicalId`: *`<graphqlapi-LogicalId><datasource-RelativeId><datasource-Type>`*`DataSource`

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

`AWS::AppSync::FunctionConfiguration`

`LogicalId`: *`<graphqlapi-LogicalId><function-RelativeId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

`AWS::AppSync::GraphQLApi`

`LogicalId`: *`<graphqlapi-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

`AWS::AppSync::GraphQLSchema`

`LogicalId`: *`<graphqlapi-LogicalId>`*`Schema`

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

`AWS::AppSync::Resolver`

`LogicalId`: *`<graphqlapi-LogicalId><OperationType><resolver-RelativeId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

Oltre a queste CloudFormation risorse, quando `AWS::Serverless::GraphQLApi` specificato, AWS SAM può generare anche le seguenti risorse. CloudFormation

`AWS::AppSync::ApiCache`

`LogicalId`: *`<graphqlapi-LogicalId>`*`ApiCache`

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

`AWS::AppSync::ApiKey`

`LogicalId`: *`<graphqlapi-LogicalId><apikey-RelativeId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

`AWS::AppSync::DomainName`

`LogicalId`: *`<graphqlapi-LogicalId>`*`DomainName`

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

`AWS::AppSync::DomainNameApiAssociation`

`LogicalId`: *`<graphqlapi-LogicalId>`*`DomainNameApiAssociation`

Proprietà referenziabile: N/A (è necessario utilizzare il per fare riferimento a questa risorsa) `LogicalId` CloudFormation

AWS SAM può anche utilizzare la `AWS::Serverless::Connector` risorsa per fornire le autorizzazioni. Per ulteriori informazioni, consulta CloudFormation risorse generate quando si specifica AWS::Serverless::Connector.

CloudFormation risorse generate quando AWS::Serverless::HttpApi è specificato

Quando `AWS::Serverless::HttpApi` viene specificato un, AWS Serverless Application Model (AWS SAM) genera una CloudFormation risorsa di `AWS::ApiGatewayV2::Api` base.

`AWS::ApiGatewayV2::Api`

`LogicalId`: *`<httpapi-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

Oltre a questa CloudFormation risorsa, quando `AWS::Serverless::HttpApi` viene specificata, genera AWS SAM anche CloudFormation risorse per i seguenti scenari:

Scenari

- StageName la proprietà è specificata
- StageName la proprietà non è specificata
- DomainName la proprietà è specificata

StageName la proprietà è specificata

Quando `AWS::Serverless::HttpApi` viene specificata la `StageName` proprietà di an, AWS SAM genera la `AWS::ApiGatewayV2::Stage` CloudFormation risorsa.

`AWS::ApiGatewayV2::Stage`

`LogicalId`: *`<httpapi-LogicalId><stage-name>`*`Stage`

*`<stage-name>`*è la stringa su cui è impostata la `StageName` proprietà. Ad esempio, se è `StageName` impostato su`Gamma`, `LogicalId` è: *`MyHttpApiGamma`* Stage.

Proprietà referenziabile: *`<httpapi-LogicalId>`*`.Stage`

StageName la proprietà non è specificata

Quando la `StageName` proprietà di an non *`AWS::Serverless::HttpApi`* è specificata, AWS SAM genera la `AWS::ApiGatewayV2::Stage` CloudFormation risorsa.

`AWS::ApiGatewayV2::Stage`

`LogicalId`: *`<httpapi-LogicalId>`*`ApiGatewayDefaultStage`

Proprietà referenziabile: *`<httpapi-LogicalId>`*`.Stage`

DomainName la proprietà è specificata

Quando `AWS::Serverless::HttpApi` viene specificata la `Domain` proprietà della proprietà di an, AWS SAM genera la `AWS::ApiGatewayV2::DomainName` CloudFormation risorsa. `DomainName`

`AWS::ApiGatewayV2::DomainName`

`LogicalId`: `ApiGatewayDomainNameV2`*`<sha>`*

*`<sha>`*è un valore hash univoco che viene generato quando viene creato lo stack. Ad esempio, `ApiGatewayDomainNameV2`*`926eeb5ff1`*.

Proprietà referenziabile: *`<httpapi-LogicalId>`*`.DomainName`

CloudFormation risorse generate quando AWS::Serverless::LayerVersion è specificato

Quando `AWS::Serverless::LayerVersion` viene specificato un, AWS Serverless Application Model (AWS SAM) genera una CloudFormation risorsa di `AWS::Lambda::LayerVersion` base.

`AWS::Lambda::LayerVersion`

`LogicalId`: *`<layerversion-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

CloudFormation risorse generate quando AWS::Serverless::SimpleTable è specificato

Quando `AWS::Serverless::SimpleTable` viene specificato un, AWS Serverless Application Model (AWS SAM) genera una CloudFormation risorsa di `AWS::DynamoDB::Table` base.

`AWS::DynamoDB::Table`

`LogicalId`: *`<simpletable-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

CloudFormation risorse generate quando AWS::Serverless::StateMachine è specificato

Quando `AWS::Serverless::StateMachine` viene specificato un, AWS Serverless Application Model (AWS SAM) genera una CloudFormation risorsa di `AWS::StepFunctions::StateMachine` base.

`AWS::StepFunctions::StateMachine`

`LogicalId`: *`<statemachine-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare il `LogicalId` per fare riferimento a questa risorsa) CloudFormation

Oltre a questa CloudFormation risorsa, quando `AWS::Serverless::StateMachine` viene specificata, genera AWS SAM anche CloudFormation risorse per i seguenti scenari:

Scenari

- La proprietà del ruolo non è specificata
- Viene specificata un'origine di eventi API
- Viene specificata una fonte di eventi Event Bridge (o Event Bus)

La proprietà del ruolo non è specificata

Quando la `Role` proprietà di un non *`AWS::Serverless::StateMachine`* è specificata, AWS SAM genera una `AWS::IAM::Role` CloudFormation risorsa.

`AWS::IAM::Role`

`LogicalId`: *`<statemachine-LogicalId>`*`Role`

Proprietà referenziabile: N/A (è necessario utilizzare per fare riferimento `LogicalId` a questa risorsa) CloudFormation

Viene specificata un'origine di eventi API

Quando la `Event` proprietà di an `AWS::Serverless::StateMachine` è impostata su`Api`, ma la `RestApiId` proprietà non è specificata, AWS SAM genera la `AWS::ApiGateway::RestApi` CloudFormation risorsa.

`AWS::ApiGateway::RestApi`

`LogicalId`: `ServerlessRestApi`

Proprietà referenziabile: N/A (è necessario utilizzare per fare riferimento `LogicalId` a questa risorsa) CloudFormation

Viene specificata una fonte di eventi Event Bridge (o Event Bus)

Quando la `Event` proprietà di an `AWS::Serverless::StateMachine` è impostata su uno dei tipi di event bridge (o event bus), AWS SAM genera la `AWS::Events::Rule` CloudFormation risorsa. Questo vale per i seguenti tipi: `EventBridgeRuleSchedule`, `eCloudWatchEvents`.

`AWS::Events::Rule`

`LogicalId`: *`<statemachine-LogicalId><event-LogicalId>`*

Proprietà referenziabile: N/A (è necessario utilizzare per fare riferimento `LogicalId` a questa risorsa) CloudFormation

Attributi delle risorse supportati da AWS SAM

Gli attributi delle risorse sono attributi che è possibile aggiungere AWS SAM e CloudFormation risorse per controllare comportamenti e relazioni aggiuntivi. Per ulteriori informazioni sugli attributi delle risorse, consulta Resource Attribute Reference nella Guida per l'AWS CloudFormation utente.

AWS SAM supporta un sottoinsieme di attributi delle risorse definiti da CloudFormation. Degli attributi di risorsa supportati, alcuni vengono copiati solo nella CloudFormation risorsa generata di base della AWS SAM risorsa corrispondente, mentre altri vengono copiati in tutte le CloudFormation risorse generate derivanti dalla risorsa corrispondente. AWS SAM Per ulteriori informazioni sulle CloudFormation risorse generate dalle AWS SAM risorse corrispondenti, vedere. CloudFormation Risorse generate per AWS SAM

La tabella seguente riassume il supporto degli attributi di risorsa per AWS SAM, in base a quanto Eccezioni elencato di seguito.

Attributi delle risorse	Risorsa/e generata/e dalla destinazione
DependsOn Metadati [1, 2]	Solo risorsa CloudFormation generata dalla base. Per informazioni sulla mappatura tra AWS SAM risorse e risorse di base CloudFormation , vedereScenari di CloudFormation risorse generate.
Condizione DeletionPolicy UpdateReplacePolicy	Tutte le CloudFormation risorse generate dalla AWS SAM risorsa corrispondente. Per informazioni sugli scenari per CloudFormation le risorse generate, vedereScenari di CloudFormation risorse generate.

Note:

1. Per ulteriori informazioni sull'utilizzo dell'attributo `Metadata` resource con il tipo di `AWS::Serverless::Function` risorsa, vedereCreazione di funzioni Lambda con runtime personalizzati in AWS SAM.
2. Per ulteriori informazioni sull'utilizzo dell'attributo `Metadata` resource con il tipo di `AWS::Serverless::LayerVersion` risorsa, vedereCreazione di livelli Lambda in AWS SAM.

Eccezioni

Esistono numerose eccezioni alle regole degli attributi delle risorse descritte in precedenza:

- PerchéAWS::Lambda::LayerVersion, il campo personalizzato AWS SAM-only RetentionPolicy imposta DeletionPolicy le risorse generate CloudFormation . Questo ha una precedenza maggiore DeletionPolicy di se stesso. Se nessuna delle due è impostata, per impostazione predefinita DeletionPolicy è impostata su. Retain
- PerchéAWS::Lambda::Version, se non DeletionPolicy è specificato, l'impostazione predefinita èRetain.
- Nello scenario in cui DeploymentPreferences è specificato per una funzione serverless, gli attributi delle risorse non vengono copiati nelle seguenti risorse generate CloudFormation :
  - AWS::CodeDeploy::Application
  - AWS::CodeDeploy::DeploymentGroup
  - Il AWS::IAM::Role nome CodeDeployServiceRole creato per questo scenario
- Se il AWS SAM modello contiene più funzioni con sorgenti di eventi API create in modo implicito, le funzioni condivideranno la AWS::ApiGateway::RestApi risorsa generata. In questo scenario, se le funzioni hanno attributi di risorsa diversi, per la AWS::ApiGateway::RestApi risorsa generata, AWS SAM copia gli attributi della risorsa in base ai seguenti elenchi con priorità:
  - UpdateReplacePolicy:
    1. Retain
    2. Snapshot
    3. Delete
  - DeletionPolicy:
    1. Retain
    2. Delete

Estensioni API Gateway per AWS SAM

Le estensioni API Gateway, progettate specificamente per AWS, forniscono personalizzazioni e funzionalità aggiuntive per la progettazione e la gestione APIs. Le estensioni API Gateway sono estensioni della specifica OpenAPI che supportano l'autorizzazione specifica e le integrazioni API AWS specifiche per API Gateway. Per ulteriori informazioni sulle estensioni API Gateway, consulta API Gateway Extensions to OpenAPI.

AWS SAM supporta un sottoinsieme di estensioni API Gateway. Per vedere da quali estensioni API Gateway sono supportate AWS SAM, consulta la tabella seguente.

Estensione API Gateway	Supportato da AWS SAM
x-amazon-apigateway-anyOggetto -method	Sì
x-amazon-apigateway-apiProprietà -key-source	No
x-amazon-apigateway-auth Oggetto	Sì
x-amazon-apigateway-authorizer Oggetto	Sì
x-amazon-apigateway-authtype Proprietà	Sì
x-amazon-apigateway-binaryProprietà -media-types	Sì
x-amazon-apigateway-documentation Oggetto	No
x-amazon-apigateway-endpoint-oggetto di configurazione	No
x-amazon-apigateway-gatewayOggetto -response	Sì
x-amazon-apigateway-gateway-Oggetto Response.GatewayResponse	Sì
x-amazon-apigateway-gateway-Oggetto Response.ResponseParameters	Sì
x-amazon-apigateway-gateway-Oggetto Response.responseTemplates	Sì
x-amazon-apigateway-integration Oggetto	Sì
x-amazon-apigateway-integration.RequestTemplates Oggetto	Sì
x-amazon-apigateway-integration.RequestParameters Oggetto	No
x-amazon-apigateway-integrationOggetto .response	Sì
x-amazon-apigateway-integrationOggetto .response	Sì
x-amazon-apigateway-integration.ResponseTemplates Oggetto	Sì

x-amazon-apigateway-integration.ResponseParameters	Sì
x-amazon-apigateway-requestProprietà -validator	No
x-amazon-apigateway-requestOggetto -validators	No
x-amazon-apigateway-requestOggetto -Validators.RequestValidator	No

Funzioni intrinseche per AWS SAM

Le funzioni intrinseche sono funzioni integrate che consentono di assegnare valori a proprietà disponibili solo in fase di esecuzione. AWS SAM ha un supporto limitato per alcune proprietà delle funzioni intrinseche, quindi non è in grado di risolvere alcune funzioni intrinseche. Di conseguenza, consigliamo di aggiungere la `AWS::LanguageExtensions` trasformazione per risolvere il problema. `AWS::LanguageExtensions`È una macro ospitata da CloudFormation che consente di utilizzare funzioni intrinseche e altre funzionalità che per impostazione predefinita non sono incluse. CloudFormation

```
Transform:
  - AWS::LanguageExtensions
  - AWS::Serverless-2016-10-31
```

> **Note**
>
> Se si utilizzano funzioni intrinseche nella CodeUri proprietà, non AWS SAM sarà in grado di analizzare correttamente i valori. Considera invece l'utilizzo di transform`AWS::LanguageExtensions`.
> Per ulteriori informazioni, fare riferimento alla sezione Proprietà di AWS::Serverless::Function.

Per ulteriori informazioni sulle funzioni intrinseche, vedere Intrinsic Function Reference nella Guida per l'utente.AWS CloudFormation

Esempi di applicazioni serverless per AWS SAM

Questa sezione include due applicazioni di esempio: una che elabora gli eventi DynamoDB e l'altra che elabora gli eventi Amazon S3. Ogni esempio illustra il step-by-step processo di creazione di un'applicazione. Inoltre, entrambi includono dettagli sulla configurazione delle fonti e AWS delle risorse degli eventi. Entrambi iniziano identificando ciò che deve essere fatto prima di iniziare e seguono i passaggi per l'inizializzazione, il test, la creazione di pacchetti e la distribuzione dell'applicazione.

Argomenti

- Elabora gli eventi DynamoDB con AWS SAM
- Elabora gli eventi di Amazon S3 con AWS SAM

Elabora gli eventi DynamoDB con AWS SAM

Con questa applicazione di esempio, si basa su ciò che è stato appreso nella panoramica e nella guida rapida e si installa un'altra applicazione di esempio. Questa applicazione è costituita da una funzione Lambda richiamata da una sorgente di eventi di tabella DynamoDB. La funzione Lambda è molto semplice: registra i dati trasmessi tramite il messaggio di origine dell'evento.

Questo esercizio mostra come imitare i messaggi di origine degli eventi che vengono passati alle funzioni Lambda quando vengono richiamate.

Prima di iniziare

Assicurati di aver completato la configurazione richiesta in. Installa il AWS SAMCLI

Passaggio 1: inizializza l'applicazione

In questa sezione, si scarica il pacchetto dell'applicazione, che consiste in un AWS SAM modello e un codice dell'applicazione.

Per inizializzare l'applicazione

1. Eseguite il comando seguente al AWS SAMCLI prompt dei comandi.

```
sam init \
```

```
--location gh:aws-samples/cookiecutter-aws-sam-dynamodb-python \
--no-input
```

Nota che `gh:` nel comando precedente viene espanso all' GitHub URL`https://github.com/`.

2. Esaminare il contenuto della directory creata dal comando (`dynamodb_event_reader/`):

   - `template.yaml`— Definisce due AWS risorse necessarie all'applicazione Read DynamoDB: una funzione Lambda e una tabella DynamoDB. Il modello definisce anche la mappatura tra le due risorse.
   - `read_dynamodb_event/`directory — Contiene il codice dell'applicazione DynamoDB.

Fase 2: Testare l'applicazione localmente

Per i test locali, usa per AWS SAMCLI generare un evento DynamoDB di esempio e richiamare la funzione Lambda:

```
sam local generate-event dynamodb update | sam local invoke --event - ReadDynamoDBEvent
```

Il `generate-event` comando crea un messaggio sorgente dell'evento di test come i messaggi che vengono creati quando tutti i componenti vengono distribuiti nel Cloud. AWS Questo messaggio di origine dell'evento viene reindirizzato alla funzione ReadDynamo DBEvent Lambda.

Verifica che i messaggi previsti vengano stampati sulla console, in base al codice sorgente in. `app.py`

Fase 3: Package dell'applicazione

Dopo aver testato l'applicazione localmente, si utilizza AWS SAMCLI per creare un pacchetto di distribuzione, da utilizzare per distribuire l'applicazione nel AWS cloud.

Per creare un pacchetto di distribuzione Lambda

1. Creare un bucket S3 nella posizione in cui si desidera salvare il codice nel pacchetto. Se si desidera utilizzare un bucket S3 esistente, ignorare questo passaggio.

   ```
   aws s3 mb s3://bucketname
   ```

2. Crea il pacchetto di distribuzione eseguendo il seguente comando `package` CLI al prompt dei comandi.

```
sam package \
    --template-file template.yaml \
    --output-template-file packaged.yaml \
    --s3-bucket bucketname
```

Il nuovo file modello viene specificato quando `packaged.yaml` si distribuisce l'applicazione nel passaggio successivo.

Fase 4: Distribuire l'applicazione

Ora che hai creato il pacchetto di distribuzione, lo usi per distribuire l'applicazione nel cloud. AWS Quindi testate l'applicazione.

Per distribuire l'applicazione serverless nel cloud AWS

- In AWS SAMCLI, utilizza il comando `deploy` CLI per distribuire tutte le risorse definite nel modello.

```
sam deploy \
    --template-file packaged.yaml \
    --stack-name sam-app \
    --capabilities CAPABILITY_IAM \
    --region us-east-1
```

Nel comando, il `--capabilities` parametro consente di AWS CloudFormation creare un ruolo IAM.

CloudFormation crea le AWS risorse definite nel modello. È possibile accedere ai nomi di queste risorse nella CloudFormation console.

Per testare l'applicazione serverless nel Cloud AWS

1. Aprire la console DynamoDB.
2. Inserisci un record nella tabella che hai appena creato.
3. Vai alla scheda Metriche della tabella e scegli Visualizza tutte le CloudWatch metriche. Nella CloudWatch console, scegli Registri per poter visualizzare l'output del registro.

Fasi successive

Per continuare a saperne di più AWS SAM, consulta le seguenti risorse:

- AWS SAM applicazioni di esempio: il AWS SAM GitHub repository contiene ulteriori applicazioni di esempio da scaricare e sperimentare.
- Serverless Land: sito che riunisce modelli serverless, esempi di codice e risorse di apprendimento specifiche per AWS le applicazioni SAM.

Elabora gli eventi di Amazon S3 con AWS SAM

Con questa applicazione di esempio, ti basi su ciò che hai appreso negli esempi precedenti e installi un'applicazione più complessa. Questa applicazione è costituita da una funzione Lambda richiamata da un'origine di eventi di caricamento di oggetti Amazon S3. Questo esercizio mostra come accedere alle AWS risorse ed effettuare chiamate di AWS servizio tramite una funzione Lambda.

Questa applicazione serverless di esempio elabora gli eventi di creazione di oggetti in Amazon S3. Per ogni immagine caricata in un bucket, Amazon S3 rileva l'evento creato dall'oggetto e richiama una funzione Lambda. La funzione Lambda richiama Amazon Rekognition per rilevare il testo contenuto nell'immagine. Quindi memorizza i risultati restituiti da Amazon Rekognition in una tabella DynamoDB.

> **Note**
>
> Con questa applicazione di esempio, esegui i passaggi in un ordine leggermente diverso rispetto agli esempi precedenti. La ragione di ciò è che questo esempio richiede la creazione di AWS risorse e la configurazione delle autorizzazioni IAM prima di poter testare la funzione Lambda localmente. Lo sfrutteremo per CloudFormation creare le risorse e configurare le autorizzazioni per te. In caso contrario, è necessario eseguire questa operazione manualmente prima di poter testare la funzione Lambda a livello locale.
> Poiché questo esempio è più complicato, assicuratevi di avere dimestichezza con l'installazione delle applicazioni di esempio precedenti prima di eseguire questo.

Prima di iniziare

Assicurati di aver completato la configurazione richiesta inInstalla il AWS SAMCLI.

Passaggio 1: inizializza l'applicazione

In questa sezione, si scarica l'applicazione di esempio, che consiste in un AWS SAM modello e un codice dell'applicazione.

Per inizializzare l'applicazione

1. Eseguite il comando seguente al AWS SAMCLI prompt dei comandi.

```
sam init \
--location https://github.com/aws-samples/cookiecutter-aws-sam-s3-rekognition-
dynamodb-python \
--no-input
```

2. Esaminare il contenuto della directory creata dal comando (`aws_sam_ocr/`):

   - `template.yaml`— Definisce tre AWS risorse di cui l'applicazione Amazon S3 ha bisogno: una funzione Lambda, un bucket Amazon S3 e una tabella DynamoDB. Il modello definisce anche le mappature e le autorizzazioni tra queste risorse.
   - `src/`directory: contiene il codice dell'applicazione Amazon S3.
   - `SampleEvent.json`— La fonte di eventi di esempio, utilizzata per i test locali.

Fase 2: Package dell'applicazione

Prima di poter testare questa applicazione localmente, è necessario utilizzare il AWS SAMCLI per creare un pacchetto di distribuzione, da utilizzare per distribuire l'applicazione nel AWS cloud. Questa distribuzione crea le AWS risorse e le autorizzazioni necessarie per testare l'applicazione localmente.

Per creare un pacchetto di distribuzione Lambda

1. Creare un bucket S3 nella posizione in cui si desidera salvare il codice nel pacchetto. Se si desidera utilizzare un bucket S3 esistente, ignorare questo passaggio.

```
aws s3 mb s3://bucketname
```

2. Crea il pacchetto di distribuzione eseguendo il seguente comando `package` CLI al prompt dei comandi.

```
sam package \
```

```
    --template-file template.yaml \
    --output-template-file packaged.yaml \
    --s3-bucket bucketname
```

Il nuovo file modello viene specificato quando `packaged.yaml` si distribuisce l'applicazione nel passaggio successivo.

Fase 3: Distribuire l'applicazione

Ora che hai creato il pacchetto di distribuzione, lo usi per distribuire l'applicazione nel cloud. AWS Quindi testate l'applicazione richiamandola nel AWS Cloud.

Per distribuire l'applicazione serverless nel cloud AWS

- In AWS SAMCLI, usa il `deploy` comando per distribuire tutte le risorse che hai definito nel modello.

```
sam deploy \
    --template-file packaged.yaml \
    --stack-name aws-sam-ocr \
    --capabilities CAPABILITY_IAM \
    --region us-east-1
```

Nel comando, il `--capabilities` parametro consente di CloudFormation creare un ruolo IAM.

CloudFormation crea le AWS risorse definite nel modello. È possibile accedere ai nomi di queste risorse nella CloudFormation console.

Per testare l'applicazione serverless nel Cloud AWS

1. Carica un'immagine nel bucket Amazon S3 che hai creato per questa applicazione di esempio.
2. Apri la console DynamoDB e trova la tabella che è stata creata. Consulta la tabella per i risultati restituiti da Amazon Rekognition.
3. Verifica che la tabella DynamoDB contenga nuovi record contenenti testo che Amazon Rekognition ha trovato nell'immagine caricata.

Fase 4: Prova l'applicazione localmente

Prima di poter testare l'applicazione localmente, è necessario recuperare i nomi delle AWS risorse create da CloudFormation.

- Recupera il nome della chiave Amazon S3 e il nome del bucket da. CloudFormation Modifica il `SampleEvent.json` file sostituendo i valori per la chiave dell'oggetto, il nome del bucket e l'ARN del bucket.
- Recupera il nome della tabella DynamoDB. Questo nome viene utilizzato per il seguente comando. `sam local invoke`

Usa AWS SAMCLI per generare un evento Amazon S3 di esempio e richiamare la funzione Lambda:

```
TABLE_NAME=Table name obtained from CloudFormation console sam local invoke --event
 SampleEvent.json
```

La `TABLE_NAME=` parte imposta il nome della tabella DynamoDB. Il `--event` parametro specifica il file che contiene il messaggio dell'evento di test da passare alla funzione Lambda.

Ora puoi verificare che i record DynamoDB previsti siano stati creati, in base ai risultati restituiti da Amazon Rekognition.

Fasi successive

Per continuare a saperne di più, consulta le seguenti AWS SAM risorse:

- AWS SAM applicazioni di esempio: il AWS SAM GitHub repository contiene ulteriori applicazioni di esempio da scaricare e sperimentare.
- Serverless Land: sito che riunisce modelli serverless, esempi di codice e risorse di apprendimento specifiche per AWS le applicazioni SAM.

Sviluppa la tua applicazione serverless con AWS SAM

Questa sezione contiene argomenti sulla convalida del AWS SAM modello e sulla creazione di un'applicazione con dipendenze. Contiene anche argomenti sull'utilizzo AWS SAM per determinati casi d'uso, come l'utilizzo dei livelli Lambda, l'uso di applicazioni annidate, il controllo dell'accesso ad API Gateway, l'orchestrazione delle APIs risorse con AWS Step Functions e la firma del codice delle applicazioni. Di seguito sono elencate le tre tappe principali da completare per sviluppare l'applicazione.

Argomenti

- Crea la tua candidatura in AWS SAM
- Definisci la tua infrastruttura con AWS SAM
- Crea la tua applicazione con AWS SAM

Crea la tua candidatura in AWS SAM

Dopo aver completato la Guida introduttiva e la letturaCome usare AWS Serverless Application Model (AWS SAM), sarai pronto per creare un AWS SAM progetto nel tuo ambiente di sviluppo. Il tuo AWS SAM progetto servirà come punto di partenza per scrivere la tua applicazione serverless. Per un elenco delle opzioni di AWS SAMCLI `sam init` comando, vederesam init.

Il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam init` fornisce opzioni per inizializzare una nuova applicazione serverless composta da:

- Un AWS SAM modello per definire il codice dell'infrastruttura.
- Una struttura di cartelle che organizza l'applicazione.
- Configurazione per le tue AWS Lambda funzioni.

Per creare un AWS SAM progetto, consulta gli argomenti di queste sezioni.

> **Note**
>
> `sam init`include modelli di progetto per funzioni durevoli nei runtime supportati (TypeScript, Python, Java). Questi modelli forniscono codice di avvio e configurazione per la creazione di applicazioni serverless con stato.

Argomenti

- Inizializza una nuova applicazione serverless
- Opzioni per sam init
- Risoluzione dei problemi
- Esempi
- Ulteriori informazioni
- Fasi successive

Inizializza una nuova applicazione serverless

Per inizializzare una nuova applicazione serverless utilizzando il AWS SAMCLI

1. `cd`in una directory iniziale.
2. Esegui quanto segue nella riga di comando:

```
$ sam init
```

3. Ti AWS SAMCLI guiderà attraverso un flusso interattivo per creare una nuova applicazione serverless.

> **Note**
>
> Come descritto in dettaglioTutorial: Implementa un'applicazione Hello World con AWS SAM, questo comando inizializza l'applicazione serverless, creando la directory del progetto. Questa directory conterrà diversi file e cartelle. Il file più importante è`template.yaml`. Questo è il tuo AWS SAM modello. La tua versione di python deve corrispondere alla versione di python elencata nel `template.yaml` file creato dal sam init comando.

Scegli un modello di partenza

Un modello è composto da quanto segue:

1. Un AWS SAM modello per il codice dell'infrastruttura.
2. Una directory di progetto iniziale che organizza i file di progetto. Ad esempio, ciò può includere:

a. Una struttura per il codice della funzione Lambda e le relative dipendenze.

b. Una `events` cartella che contiene eventi di test per i test locali.

c. Una `tests` cartella per supportare i test unitari.

d. Un `samconfig.toml` file per configurare le impostazioni del progetto.

e. Un `ReadMe` file e altri file di base del progetto iniziale.

Di seguito è riportato un esempio di cartella iniziale del progetto:

```
sam-app
### README.md
### __init__.py
### events
#   ### event.json
### hello_world
#   ### __init__.py
#   ### app.py
#   ### requirements.txt
### samconfig.toml
### template.yaml
### tests
    ### __init__.py
    ### integration
    #   ### __init__.py
    #   ### test_api_gateway.py
    ### requirements.txt
    ### unit
        ### __init__.py
        ### test_handler.py
```

È possibile selezionare da un elenco di modelli di avvio AWS rapido disponibili o fornire una posizione personalizzata per i modelli.

Per scegliere un modello di avvio AWS rapido

1. Quando richiesto, seleziona Modelli di avvio AWS rapido.
2. Seleziona un modello AWS Quick Start con cui iniziare. Di seguito è riportato un esempio:

```
Which template source would you like to use?
    1 - AWS Quick Start Templates
```

```
    2 - Custom Template Location
Choice: 1

Choose an AWS Quick Start application template
    1 - Hello World Example
    2 - Multi-step workflow
    3 - Serverless API
    4 - Scheduled task
    5 - Standalone function
    6 - Data processing
    7 - Hello World Example With Powertools
    8 - Infrastructure event management
    9 - Serverless Connector Hello World Example
    10 - Multi-step workflow with Connectors
    11 - Lambda EFS example
    12 - DynamoDB Example
    13 - Machine Learning
Template: 4
```

Per scegliere la posizione del modello personalizzata

1. Quando richiesto, seleziona la posizione del modello personalizzato.

```
Which template source would you like to use?
    1 - AWS Quick Start Templates
    2 - Custom Template Location
Choice: 2
```

2. Ti AWS SAMCLI verrà chiesto di fornire una posizione per il modello.

```
Template location (git, mercurial, http(s), zip, path):
```

   Fornisci una delle seguenti posizioni all'archivio del file.zip del modello:

   - GitHubrepository: il percorso del file.zip nel tuo repository. GitHub Il file deve trovarsi nella cartella principale del repository.
   - Mercurialrepository: il percorso del file.zip nel tuo repository. Mercurial Il file deve trovarsi nella cartella principale del repository.
   - Percorso.zip: un percorso HTTPS o locale del file.zip.

3. AWS SAMCLIInizializzerà l'applicazione serverless utilizzando il modello personalizzato.

Scegli un runtime

Quando scegli un modello AWS Quick Start, AWS SAMCLI ti viene richiesto di selezionare un runtime per le tue funzioni Lambda. L'elenco di opzioni visualizzato da AWS SAMCLI sono i runtime supportati nativamente da Lambda.

- Il runtime fornisce un ambiente specifico del linguaggio di programmazione che viene eseguito in un ambiente di esecuzione.
- Quando viene distribuito su Cloud AWS, il servizio Lambda richiama la funzione in un ambiente di esecuzione.

È possibile utilizzare qualsiasi altro linguaggio di programmazione con un runtime personalizzato. A tale scopo, è necessario creare manualmente la struttura dell'applicazione iniziale. È quindi possibile utilizzarla `sam init` per inizializzare rapidamente l'applicazione configurando una posizione personalizzata del modello.

Dalla selezione, AWS SAMCLI crea la directory iniziale per il codice della funzione Lambda e le dipendenze.

Se Lambda supporta più gestori delle dipendenze per il tuo runtime, ti verrà richiesto di scegliere il tuo gestore delle dipendenze preferito.

Scegli un tipo di pacchetto

Quando si sceglie un modello di avvio AWS rapido e un runtime, viene AWS SAMCLI richiesto di selezionare un tipo di pacchetto. Il tipo di pacchetto determina il modo in cui le funzioni Lambda vengono distribuite per l'uso con il servizio Lambda. I due tipi di pacchetto supportati sono:

1. Immagine del contenitore: contiene il sistema operativo di base, il runtime, le estensioni Lambda, il codice dell'applicazione e le relative dipendenze.
2. Archivio di file.zip: contiene il codice dell'applicazione e le relative dipendenze.

Per ulteriori informazioni sui tipi di pacchetti di distribuzione, consulta i pacchetti di distribuzione Lambda nella AWS Lambda Developer Guide.

Di seguito è riportato un esempio di struttura di directory di un'applicazione con una funzione Lambda impacchettata come immagine contenitore. AWS SAMCLIScarica l'immagine e ne crea una `Dockerfile` nella directory della funzione per specificare l'immagine.

```
sam-app
### README.md
### __init__.py
### events
#   ### event.json
### hello_world
#   ### Dockerfile
#   ### __init__.py
#   ### app.py
#   ### requirements.txt
### samconfig.toml
### template.yaml
### tests
    ### __init__.py
    ### unit
        ### __init__.py
        ### test_handler.py
```

Di seguito è riportato un esempio di struttura di directory di un'applicazione con una funzione impacchettata come archivio di file.zip.

```
sam-app
### README.md
### __init__.py
### events
#   ### event.json
### hello_world
#   ### __init__.py
#   ### app.py
#   ### requirements.txt
### samconfig.toml
### template.yaml
### tests
    ### __init__.py
    ### integration
    #   ### __init__.py
    #   ### test_api_gateway.py
    ### requirements.txt
    ### unit
        ### __init__.py
        ### test_handler.py
```

Configura il tracciamento AWS X-Ray

È possibile scegliere di attivare il AWS X-Ray tracciamento. Per ulteriori informazioni, vedi Cos'è AWS X-Ray? nella Guida per gli AWS X-Ray sviluppatori.

Se si attiva, AWS SAMCLI configura il AWS SAM modello. Di seguito è riportato un esempio:

```
Globals:
  Function:
    ...
    Tracing: Active
  Api:
    TracingEnabled: True
```

Configura il monitoraggio con Amazon CloudWatch Application Insights

Puoi scegliere di attivare il monitoraggio utilizzando Amazon CloudWatch Application Insights. Per ulteriori informazioni, consulta Amazon CloudWatch Application Insights nella Amazon CloudWatch User Guide.

Se lo attivi, AWS SAMCLI configura il tuo AWS SAM modello. Di seguito è riportato un esempio:

```
Resources:
  ApplicationResourceGroup:
    Type: AWS::ResourceGroups::Group
    Properties:
      Name:
        Fn::Join:
        - ''
        - - ApplicationInsights-SAM-
          - Ref: AWS::StackName
      ResourceQuery:
        Type: CLOUDFORMATION_STACK_1_0
  ApplicationInsightsMonitoring:
    Type: AWS::ApplicationInsights::Application
    Properties:
      ResourceGroupName:
        Fn::Join:
        - ''
        - - ApplicationInsights-SAM-
          - Ref: AWS::StackName
      AutoConfigurationEnabled: 'true'
```

```
    DependsOn: ApplicationResourceGroup
```

Assegna un nome all'applicazione

Fornisci un nome per la tua candidatura. AWS SAMCLICrea una cartella di primo livello per l'applicazione utilizzando questo nome.

Opzioni per sam init

Di seguito sono riportate alcune delle opzioni principali che è possibile utilizzare con il `sam init` comando. Per un elenco di tutte le opzioni, vedere sam init.

Inizializza un'applicazione utilizzando una posizione personalizzata del modello

Utilizza l'`--location`opzione e fornisci una posizione del modello personalizzata supportata. Di seguito è riportato un esempio:

```
$ sam init --location https://github.com/aws-samples/sessions-with-aws-sam/raw/master/
starter-templates/web-app.zip
```

Inizializza un'applicazione senza il flusso interattivo

Utilizza l'`--no-interactive`opzione e fornisci le tue scelte di configurazione nella riga di comando per saltare il flusso interattivo. Di seguito è riportato un esempio:

```
$ sam init --no-interactive --runtime go1.x --name go-demo --dependency-manager mod --
app-template hello-world
```

Risoluzione dei problemi

Per risolvere il problema, consulta. AWS SAMCLI AWS SAMCLIrisoluzione dei problemi

Esempi

Inizializza una nuova applicazione serverless utilizzando il modello Hello World Starter AWS

Per questo esempio, vedi Passaggio 1: inizializza l'applicazione Hello World di esempio in Tutorial: Deploying a Hello World application.

Inizializza una nuova applicazione serverless con una posizione del modello personalizzata

Di seguito sono riportati alcuni esempi di come fornire una GitHub posizione al modello personalizzato:

```
$ sam init --location gh:aws-samples/cookiecutter-aws-sam-python
$ sam init --location git+sh://git@github.com/aws-samples/cookiecutter-aws-sam-
python.git
$ sam init --location hg+ssh://hg@bitbucket.org/repo/template-name
```

Di seguito è riportato un esempio di percorso di file locale:

```
$ sam init --location /path/to/template.zip
```

Di seguito è riportato un esempio di percorso raggiungibile tramite HTTPS:

```
$ sam init --location https://github.com/aws-samples/sessions-with-aws-sam/raw/master/
starter-templates/web-app.zip
```

Ulteriori informazioni

Per ulteriori informazioni sull'utilizzo del `sam init` comando, consulta quanto segue:

- Apprendimento AWS SAM: sam init — Serie Serverless Land «Learning AWS SAM» suYouTube.
- Strutturazione di applicazioni serverless da utilizzare con AWS SAMCLI (Sessions with SAM S2E7) — Sessioni con serie attivate. AWS SAM YouTube

Fasi successive

Ora che hai creato il tuo AWS SAM progetto, sei pronto per iniziare a creare la tua applicazione. Definisci la tua infrastruttura con AWS SAMPer eseguire questa operazione, consultate le istruzioni dettagliate sulle attività da completare.

Definisci la tua infrastruttura con AWS SAM

Ora che hai creato il tuo progetto, sei pronto per definire la tua infrastruttura applicativa con AWS SAM. A tale scopo, configurate il AWS SAM modello per definire le risorse e le proprietà dell'applicazione, ovvero il `template.yaml` file del AWS SAM progetto.

Gli argomenti di questa sezione forniscono contenuti sulla definizione dell'infrastruttura nel AWS SAM modello (il `template.yaml` file). Contiene anche argomenti sulla definizione delle risorse per casi d'uso specifici, come l'utilizzo dei livelli Lambda, l'utilizzo di applicazioni annidate, il controllo dell'accesso ad API Gateway, l'orchestrazione delle risorse con AWS Step Functions APIs, la firma del codice delle applicazioni e la convalida del modello. AWS SAM

Argomenti

- Definisci le risorse dell'applicazione nel tuo AWS SAM modello
- Configura e gestisci l'accesso alle risorse nel tuo AWS SAM modello
- Controlla l'accesso alle API con il tuo AWS SAM modello
- Aumenta l'efficienza utilizzando i livelli Lambda con AWS SAM
- Riutilizza codice e risorse utilizzando applicazioni annidate in AWS SAM
- Gestisci eventi basati sul tempo con EventBridge Scheduler in AWS SAM
- Orchestrazione delle risorse con AWS SAM AWS Step Functions
- Configura la firma del codice per la tua AWS SAM applicazione
- Convalida dei file AWS SAM modello

Definisci le risorse dell'applicazione nel tuo AWS SAM modello

Definisci le AWS risorse utilizzate dall'applicazione serverless nella `Resources` sezione del AWS SAM modello. Quando si definisce una risorsa, si identifica cos'è la risorsa, come interagisce con altre risorse e in che modo è possibile accedervi (ovvero le autorizzazioni della risorsa).

La `Resources` sezione del AWS SAM modello può contenere una combinazione di CloudFormation risorse e AWS SAM risorse. Inoltre, puoi utilizzare AWS SAM la sintassi abbreviata per le seguenti risorse:

AWS SAM sintassi abbreviata	Cosa fa con una risorsa correlata AWS
AWS::Serverless::Api	Crea una raccolta di risorse e metodi API Gateway che possono essere richiamati tramite endpoint HTTPS.
AWS::Serverless::Application	Incorpora un'applicazione serverless da AWS Serverless Application Repositoryo da un bucket Amazon S3 come applicazione annidata.
AWS::Serverless::Connector	Configura le autorizzazioni tra due risorse. Per un'introduzione ai connettori, vedere. Gestione delle autorizzazioni delle risorse con i connettori AWS SAM
AWS::Serverless::Function	Crea una AWS Lambda funzione, un ruolo di esecuzione AWS Identity and Access Management (IAM) e mappature delle sorgenti degli eventi che attivano la funzione.
AWS::Serverless::GraphQLApi	crea e configura un' AWS AppSync GraphQLAPI per la tua applicazione serverless.
AWS::Serverless::HttpApi	Crea un'API HTTP Amazon API Gateway, che ti consente di creare RESTful APIs con una latenza inferiore e costi inferiori rispetto a REST APIs.
AWS::Serverless::LayerVersion	Crea una Lambda LayerVersion che contiene il codice di libreria o di runtime necessario per una funzione Lambda.
AWS::Serverless::SimpleTable	Crea una tabella DynamoDB con una chiave primaria a singolo attributo.
AWS::Serverless::StateMachine	Crea una macchina a AWS Step Functions stati, che puoi utilizzare per orchestrare AWS

AWS SAM sintassi abbreviata	Cosa fa con una risorsa correlata AWS
	Lambda funzioni e altre AWS risorse per formare flussi di lavoro complessi e robusti.

Le risorse di cui sopra sono elencate anche in. AWS SAM risorse e proprietà

Per informazioni di riferimento per tutti i tipi di AWS risorse e proprietà CloudFormation e il AWS SAM supporto, vedere il riferimento ai tipi di AWS risorse e proprietà nella Guida per l'AWS CloudFormation utente.

Configura e gestisci l'accesso alle risorse nel tuo AWS SAM modello

Affinché le AWS risorse interagiscano tra loro, è necessario configurare l'accesso e le autorizzazioni corretti tra le risorse. A tale scopo è necessaria la configurazione di utenti, ruoli e policy AWS Identity and Access Management (IAM) per realizzare l'interazione in modo sicuro.

Gli argomenti di questa sezione sono tutti relativi alla configurazione dell'accesso alle risorse definite nel modello. Questa sezione inizia con le migliori pratiche generali. Nei due argomenti successivi vengono esaminate due opzioni disponibili per configurare l'accesso e le autorizzazioni tra le risorse a cui fa riferimento l'applicazione serverless: AWS SAM connettori e AWS SAM modelli di policy. L'ultimo argomento fornisce dettagli per la gestione dell'accesso degli utenti utilizzando gli stessi meccanismi CloudFormation utilizzati per la gestione degli utenti.

Per ulteriori informazioni, consulta Controllare l'accesso con AWS Identity and Access Management nella Guida per l'AWS CloudFormation utente.

La AWS Serverless Application Model (AWS SAM) offre due opzioni che semplificano la gestione degli accessi e delle autorizzazioni per le applicazioni serverless.

1. AWS SAM connettori
2. AWS SAM modelli di policy

AWS SAM connettori

I connettori sono un modo per fornire autorizzazioni tra due risorse. A tale scopo, descrivi come devono interagire tra loro nel modello. AWS SAM Possono essere definiti utilizzando l'attributo di `Connectors` risorsa o il tipo di `AWS::Serverless::Connector` risorsa. I connettori supportano

la fornitura `Read` e `Write` l'accesso a dati ed eventi tra una combinazione di AWS risorse. Per ulteriori informazioni sui AWS SAM connettori, consultaGestione delle autorizzazioni delle risorse con i connettori AWS SAM.

AWS SAM modelli di policy

AWS SAM i modelli di policy sono set di autorizzazioni predefiniti che è possibile aggiungere ai AWS SAM modelli per gestire l'accesso e le autorizzazioni tra le AWS Lambda funzioni, le macchine a AWS Step Functions stati e le risorse con cui interagiscono. Per ulteriori informazioni sui modelli di AWS SAM policy, consulta. AWS SAM modelli di policy

AWS CloudFormation meccanismi

CloudFormation i meccanismi includono la configurazione di utenti, ruoli e policy IAM per gestire le autorizzazioni tra le AWS risorse. Per ulteriori informazioni, consulta Gestione delle AWS SAM autorizzazioni con meccanismi CloudFormation.

Best practice

Nelle tue applicazioni serverless, puoi utilizzare diversi metodi per configurare le autorizzazioni tra le tue risorse. Pertanto, è possibile selezionare l'opzione migliore per ogni scenario e utilizzare più opzioni contemporaneamente in tutte le applicazioni. Ecco alcuni aspetti da considerare quando si sceglie l'opzione migliore per voi:

- AWS SAM i connettori e i modelli di policy riducono entrambi le competenze IAM necessarie per facilitare le interazioni sicure tra le AWS risorse. Utilizza connettori e modelli di policy se supportati.
- AWS SAM i connettori forniscono una sintassi abbreviata semplice e intuitiva per definire le autorizzazioni nei AWS SAM modelli e richiedono la minima esperienza in materia di IAM. Quando sono supportati sia i AWS SAM connettori che i modelli di policy, utilizza i connettori. AWS SAM
- AWS SAM i connettori possono fornire `Read` e `Write` accedere a dati ed eventi tra le risorse di AWS SAM origine e di destinazione supportate. Per un elenco delle risorse supportate, vedereAWS SAM riferimento del connettore. Se supportato, usa i AWS SAM connettori.
- Sebbene i modelli di AWS SAM policy siano limitati alle autorizzazioni tra le funzioni Lambda, le macchine a stati di Step Functions e AWS le risorse con cui interagiscono, i modelli di policy supportano tutte le operazioni CRUD. Se supportati e quando è disponibile un modello di AWS SAM policy per il tuo scenario, utilizza AWS SAM i modelli di policy. Per un elenco dei modelli di policy disponibili, consultaAWS SAM modelli di policy.

- Per tutti gli altri scenari o quando è richiesta la granularità, utilizzate CloudFormation i meccanismi.

Gestione delle autorizzazioni delle risorse con i connettori AWS SAM

I connettori sono un tipo di risorsa astratto AWS Serverless Application Model (AWS SAM), identificato come`AWS::Serverless::Connector`, che fornisce autorizzazioni semplici e ben definite tra le risorse delle applicazioni serverless.

Vantaggi dei connettori AWS SAM

Componendo automaticamente le politiche di accesso appropriate tra le risorse, Connectors vi offre la possibilità di creare applicazioni serverless e concentrarvi sull'architettura dell'applicazione senza bisogno di competenze in termini di funzionalità di AWS autorizzazione, linguaggio delle policy e impostazioni di sicurezza specifiche del servizio. Pertanto, i connettori rappresentano un grande vantaggio per gli sviluppatori che potrebbero essere alle prime armi nello sviluppo serverless o per gli sviluppatori esperti che desiderano aumentare la velocità di sviluppo.

Utilizzo dei connettori AWS SAM

Usa l'attributo `Connectors` resource incorporandolo in una risorsa di origine. Quindi, definisci la risorsa di destinazione e descrivi come i dati o gli eventi devono fluire tra tali risorse. AWS SAM compone quindi le politiche di accesso necessarie per facilitare le interazioni richieste.

Di seguito viene descritto come viene scritto questo attributo di risorsa:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  <source-resource-logical-id>:
    Type: <resource-type>
    ...
    Connectors:
      <connector-name>:
        Properties:
          Destination:
            <properties-that-identify-destination-resource>
          Permissions:
            <permission-types-to-provision>
  ...
```

Come funzionano i connettori

> **Note**
>
> Questa sezione spiega come i connettori forniscono le risorse necessarie dietro le quinte. Ciò avviene automaticamente quando si utilizzano i connettori.

Innanzitutto, l'attributo di `Connectors` risorsa incorporato viene trasformato in un tipo di `AWS::Serverless::Connector` risorsa. Il relativo ID logico viene creato automaticamente come*`<source-resource-logical-id><embedded-connector-logical-id>`*.

Ad esempio, ecco un connettore incorporato:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyFunction:
    Type: AWS::Lambda::Function
    Connectors:
      MyConn:
        Properties:
          Destination:
            Id: MyTable
          Permissions:
            - Read
            - Write
  MyTable:
    Type: AWS::DynamoDB::Table
```

Questo genererà la seguente `AWS::Serverless::Connector` risorsa:

```
Transform: AWS::Serverless-2016-10-31
Resources:
  ...
  MyFunctionMyConn:
    Type: AWS::Serverless::Connector
    Properties:
      Source:
        Id: MyFunction
      Destination:
```

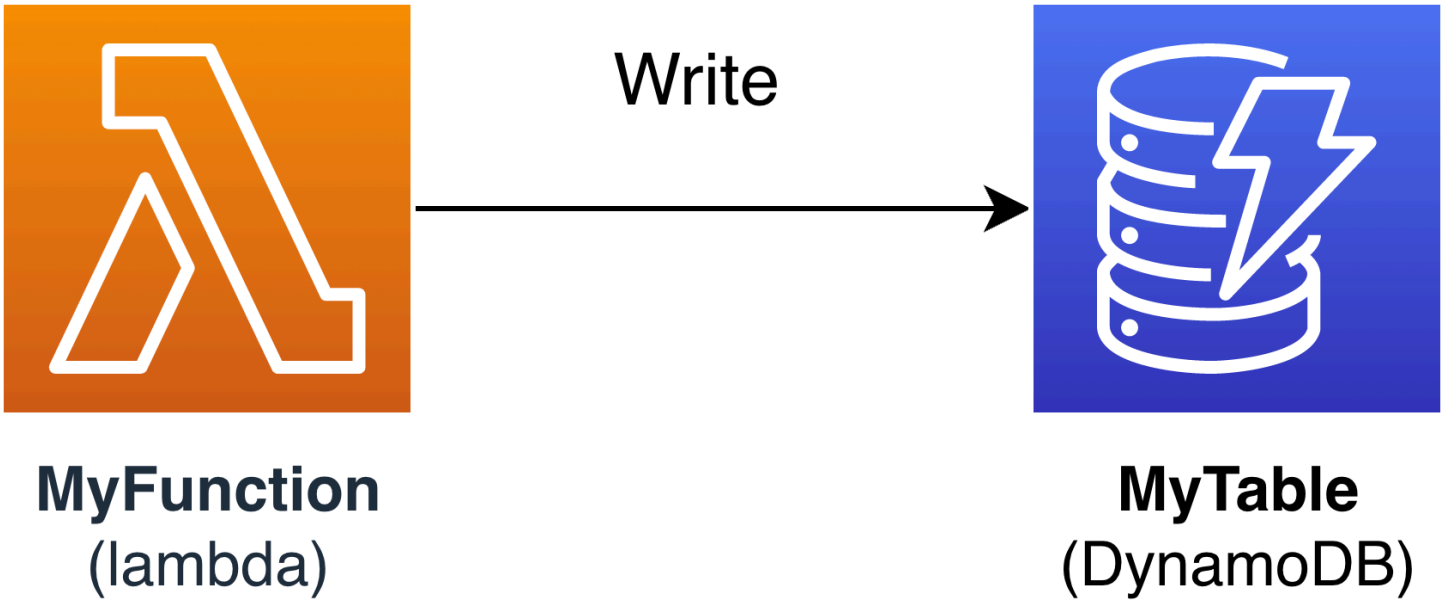
```
        Id: MyTable
      Permissions:
        - Read
        - Write
```

> **Note**
>
> È inoltre possibile definire i connettori nel AWS SAM modello utilizzando questa sintassi. Questa operazione è consigliata quando la risorsa di origine è definita su un modello separato dal connettore.

Successivamente, le politiche di accesso necessarie per questa connessione vengono composte automaticamente. Per ulteriori informazioni sulle risorse generate dai connettori, vedereCloudFormation risorse generate quando si specifica AWS::Serverless::Connector.

Esempio di connettori

L'esempio seguente mostra come utilizzare i connettori per scrivere dati da una AWS Lambda funzione a una tabella Amazon DynamoDB.




```
Transform: AWS::Serverless-2016-10-31
Resources:
  MyTable:
    Type: AWS::Serverless::SimpleTable
  MyFunction:
```

```
    Type: AWS::Serverless::Function
    Connectors:
      MyConn:
        Properties:
          Destination:
            Id: MyTable
          Permissions:
            - Write
    Properties:
      Runtime: nodejs16.x
      Handler: index.handler
      InlineCode: |
        const AWS = require("aws-sdk");
        const docClient = new AWS.DynamoDB.DocumentClient();
        exports.handler = async (event, context) => {
          await docClient.put({
            TableName: process.env.TABLE_NAME,
            Item: {
              id: context.awsRequestId,
              event: JSON.stringify(event)
            }
          }).promise();
        }
      Environment:
        Variables:
          TABLE_NAME: !Ref MyTable
```

L'attributo `Connectors` resource è incorporato nella risorsa sorgente della funzione Lambda. La tabella DynamoDB è definita come risorsa di destinazione utilizzando la proprietà. `Id` I connettori forniranno le `Write` autorizzazioni tra queste due risorse.

Quando distribuisci il AWS SAM modello su CloudFormation, AWS SAM comporrà automaticamente le politiche di accesso necessarie per il funzionamento di questa connessione.

Connessioni supportate tra risorse di origine e destinazione

Supportano i connettori `Read` e `Write` i tipi di autorizzazione per dati ed eventi tra una combinazione selezionata di connessioni alle risorse di origine e destinazione. Ad esempio, i connettori supportano una `Write` connessione tra una `AWS::ApiGateway::RestApi` risorsa di origine e una risorsa di `AWS::Lambda::Function` destinazione.

Le risorse di origine e di destinazione possono essere definite utilizzando una combinazione di proprietà supportate. I requisiti delle proprietà dipenderanno dalla connessione che si sta effettuando e da dove vengono definite le risorse.

> **Note**
>
> I connettori possono fornire le autorizzazioni tra i tipi di risorse serverless e non serverless supportati.

Per un elenco delle connessioni di risorse supportate e dei relativi requisiti di proprietà, vedere. Tipi di risorse di origine e destinazione supportati per i connettori

Definire le autorizzazioni di lettura e scrittura in AWS SAM

In AWS SAM, `Read` e `Write` le autorizzazioni possono essere fornite all'interno di un singolo connettore:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyFunction:
    Type: AWS::Lambda::Function
    Connectors:
      MyTableConn:
        Properties:
          Destination:
            Id: MyTable
          Permissions:
            - Read
            - Write
  MyTable:
    Type: AWS::DynamoDB::Table
```

Per ulteriori informazioni sull'uso dei connettori, fare riferimento a. AWS SAM riferimento del connettore

Definire le risorse utilizzando altre proprietà supportate in AWS SAM

Per le risorse di origine e di destinazione, se definite all'interno dello stesso modello, utilizzate la `Id` proprietà. Facoltativamente, `Qualifier` è possibile aggiungere un per restringere l'ambito della

risorsa definita. Se la risorsa non si trova all'interno dello stesso modello, utilizzate una combinazione di proprietà supportate.

- Per un elenco delle combinazioni di proprietà supportate per le risorse di origine e di destinazione, vedereTipi di risorse di origine e destinazione supportati per i connettori.
- Per una descrizione delle proprietà che è possibile utilizzare con i connettori, vedereAWS::Serverless::Connector.

Quando definite una risorsa di origine con una proprietà diversa da`Id`, utilizzate la `SourceReference` proprietà.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  <source-resource-logical-id>:
    Type: <resource-type>
    ...
    Connectors:
      <connector-name>:
        Properties:
          SourceReference:
            Qualifier: <optional-qualifier>
            <other-supported-properties>
          Destination:
            <properties-that-identify-destination-resource>
          Permissions:
            <permission-types-to-provision>
```

Ecco un esempio di utilizzo di `Qualifier` a per restringere l'ambito di una risorsa Amazon API Gateway:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyApi:
    Type: AWS::Serverless::Api
    Connectors:
      ApiToLambdaConn:
        Properties:
```

```
          SourceReference:
            Qualifier: Prod/GET/foobar
          Destination:
            Id: MyFunction
          Permissions:
            - Write
  ...
```

Ecco un esempio di utilizzo di una combinazione supportata di `Arn` e `Type` per definire una risorsa di destinazione da un altro modello:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Connectors:
      TableConn:
        Properties:
          Destination:
            Type: AWS::DynamoDB::Table
            Arn: !GetAtt MyTable.Arn
  ...
```

Per ulteriori informazioni sull'uso dei connettori, fare riferimento aAWS SAM riferimento del connettore.

Crea più connettori da un'unica fonte in AWS SAM

All'interno di una risorsa di origine, è possibile definire più connettori, ciascuno con una risorsa di destinazione diversa.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Connectors:
      BucketConn:
        Properties:
```

```
          Destination:
            Id: amzn-s3-demo-bucket
          Permissions:
            - Read
            - Write
      SQSConn:
        Properties:
          Destination:
            Id: MyQueue
          Permissions:
            - Read
            - Write
      TableConn:
        Properties:
          Destination:
            Id: MyTable
          Permissions:
            - Read
            - Write
      TableConnWithTableArn:
        Properties:
          Destination:
            Type: AWS::DynamoDB::Table
            Arn: !GetAtt MyTable.Arn
          Permissions:
            - Read
            - Write
...
```

Per ulteriori informazioni sull'uso dei connettori, fare riferimento aAWS SAM riferimento del connettore.

Creare connettori a destinazione multipla in AWS SAM

All'interno di una risorsa di origine, è possibile definire un singolo connettore con più risorse di destinazione. Ecco un esempio di risorsa sorgente di una funzione Lambda che si connette a un bucket Amazon Simple Storage Service (Amazon S3) e a una tabella DynamoDB:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyFunction:
```

```
    Type: AWS::Serverless::Function
    Connectors:
      WriteAccessConn:
        Properties:
          Destination:
            - Id: OutputBucket
            - Id: CredentialTable
          Permissions:
            - Write
  ...
  OutputBucket:
    Type: AWS::S3::Bucket
  CredentialTable:
    Type: AWS::DynamoDB::Table
```

Per ulteriori informazioni sull'uso dei connettori, consulta. AWS SAM riferimento del connettore

Definire gli attributi delle risorse con i connettori in AWS SAM

È possibile definire gli attributi delle risorse per specificare comportamenti e relazioni aggiuntivi. Per ulteriori informazioni sugli attributi delle risorse, consulta il riferimento agli attributi delle risorse nella Guida per l'AWS CloudFormation utente.

È possibile aggiungere attributi di risorsa al connettore incorporato definendoli allo stesso livello delle proprietà del connettore. Quando il AWS SAM modello viene trasformato durante la distribuzione, gli attributi passeranno alle risorse generate.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Connectors:
      MyConn:
        DeletionPolicy: Retain
        DependsOn: AnotherFunction
        Properties:
          ...
```

Per ulteriori informazioni sull'uso dei connettori, fare riferimento aAWS SAM riferimento del connettore.

Ulteriori informazioni

Per ulteriori informazioni sull'uso dei AWS SAM connettori, consulta i seguenti argomenti:

- AWS::Serverless::Connector
- Definire le autorizzazioni di lettura e scrittura in AWS SAM
- Definire le risorse utilizzando altre proprietà supportate in AWS SAM
- Crea più connettori da un'unica fonte in AWS SAM
- Creare connettori a destinazione multipla in AWS SAM
- Definire le autorizzazioni di lettura e scrittura in AWS SAM
- Definire gli attributi delle risorse con i connettori in AWS SAM

Rilascio di feedback

Per fornire feedback sui connettori, invia un nuovo problema al serverless-application-model AWS GitHubrepository.

AWS SAM modelli di policy

Il AWS Serverless Application Model (AWS SAM) consente di scegliere da un elenco di modelli di policy per definire l'ambito delle autorizzazioni delle funzioni Lambda AWS Step Functions e delle macchine a stati per le risorse utilizzate dall'applicazione.

AWS SAM le applicazioni AWS Serverless Application Repository che utilizzano modelli di policy non richiedono alcun riconoscimento speciale da parte dei clienti per distribuire l'applicazione da. AWS Serverless Application Repository

Se desideri richiedere l'aggiunta di un nuovo modello di policy, procedi come segue:

1. Invia una pull request per il file sorgente policy_templates.json nel ramo del progetto. `develop` AWS SAM GitHub Puoi trovare il file sorgente in policy_templates.json sul sito web. GitHub
2. Invia un problema nel AWS SAM GitHub progetto che includa i motivi della pull request e un link alla richiesta. Usa questo link per inviare un nuovo numero: AWS Serverless Application Model: Problemi.

Sintassi

Per ogni modello di policy specificato nel file di AWS SAM modello, è necessario specificare sempre un oggetto contenente i valori segnaposto del modello di policy. Se un modello di policy non richiede alcun valore segnaposto, è necessario specificare un oggetto vuoto.

YAML

```
MyFunction:
  Type: AWS::Serverless::Function
  Properties:
    Policies:
      - PolicyTemplateName1:        # Policy template with placeholder value
          Key1: Value1
      - PolicyTemplateName2: {}     # Policy template with no placeholder value
```

> **Note**
>
> Se hai impostato una normale policy IAM o hai gestito policy tramite Lambda, il modello di policy può essere impostato senza utilizzare un oggetto vuoto.

Esempi

Esempio 1: modello di policy con valori placeholder

L'esempio seguente mostra che il modello di policy SQSPollerPolicy si aspetta `QueueName` come risorsa. Il AWS SAM modello recupera il nome della coda `MyQueue` "" Amazon SQS, che puoi creare nella stessa applicazione o richiesta come parametro all'applicazione.

```
MyFunction:
  Type: 'AWS::Serverless::Function'
  Properties:
    CodeUri: ${codeuri}
    Handler: hello.handler
    Runtime: python2.7
    Policies:
      - SQSPollerPolicy:
          QueueName:
            !GetAtt MyQueue.QueueName
```

Esempio 2: modello di policy senza valori placeholder

L'esempio seguente contiene il modello di policy CloudWatchPutMetricPolicy senza valori placeholder.

> **Note**
>
> Anche se non sono presenti valori segnaposto, è necessario specificare un oggetto vuoto, altrimenti si verificherà un errore.

```
MyFunction:
  Type: 'AWS::Serverless::Function'
  Properties:
    CodeUri: ${codeuri}
    Handler: hello.handler
    Runtime: python2.7
    Policies:
      - CloudWatchPutMetricPolicy: {}
```

Esempio 3: modello di policy con valori segnaposto e una normale policy IAM

L'esempio seguente contiene la policy e DynamoDBCrudPolicy il modello di policy di Amazon SQSFull Access. La policy di Amazon SQSFull Access è una policy IAM e non una AWS SAM policy, quindi non è necessario specificare un oggetto vuoto poiché la policy verrebbe passata direttamente a CloudFormation.

```
MyFunction:
  Type: 'AWS::Serverless::Function'
  Properties:
    CodeUri: ${codeuri}
    Handler: hello.handler
    Runtime: python2.7
    Policies:
      - AmazonSQSFullAccess // IAM policy could be set without passing an empty object
      - DynamoDBCrudPolicy: // SAM specific policy, has a defined structure
          TableName:
            !Ref SampleTable
```

Tabella dei modelli di policy

Di seguito è riportata una tabella dei modelli di policy disponibili.

Modello di policy	Description		
AcmGetCertificatePolicy	Fornisce il permesso di leggere un certificato da AWS Certificate Manager.		
AMIDescribePolicy	Autorizza a descrivere Amazon Machine Images (AMIs).		
AthenaQueryPolicy	Fornisce le autorizzazioni per eseguire le query Athena.		
AWSSecretsManagerGetSecretValuePolicy	Autorizza a ottenere il valore segreto per il segreto specificato. Gestione dei segreti AWS		
AWSSecretsManagerRotationPolicy	Dà il permesso di far ruotare un segreto. Gestione dei segreti AWS		
CloudFormationDescribeStacksPolicy	Dà il permesso di descrivere le CloudFormation pile.		
CloudWatchDashboardPolicy	Fornisce le autorizzazioni per inserire le metriche in modo che funzionino nei dashboard. CloudWatch		
CloudWatchDescribeAlarmHistoryPolicy	Autorizza a descrivere CloudWatch la cronologia degli allarmi.		

Modello di policy	Description		
CloudWatchPutMetricPolicy	Autorizza a inviare metriche a. CloudWatch		
CodeCommitCrudPolicy	Fornisce le autorizzazioni agli create/read/update/delete oggetti all'interno di un repository specifico CodeCommit.		
CodeCommitReadPolicy	Fornisce le autorizzazioni per leggere oggetti all'interno di un repository specifico . CodeCommit		
CodePipelineLambdaExecutionPolicy	Fornisce l'autorizzazione a una funzione Lambda richiamata da CodePipeline per segnalare lo stato del lavoro.		
CodePipelineReadOnlyPolicy	Fornisce il permesso di lettura per ottenere dettagli su una CodePipeline pipeline.		
ComprehendBasicAccessPolicy	Autorizza il rilevamento di entità, frasi chiave, lingue e sentimenti.		
CostExplorerReadOnlyPolicy	Fornisce l'autorizzazione di sola lettura al Cost APIs Explorer di sola lettura per la cronologia di fatturazione.		
DynamoDBBackupFullAccessPolicy	Fornisce l'autorizzazione di lettura e scrittura ai backup su richiesta di DynamoDB per una tabella.		
DynamoDBCrudPolicy	Fornisce autorizzazioni di creazione, lettura, aggiornamento ed eliminazione per una tabella Amazon DynamoDB.		

Modello di policy	Description		
DynamoDBReadPolicy	Fornisce l'autorizzazione di sola lettura a una tabella DynamoDB.		
DynamoDBReconfigurePolicy	Fornisce il permesso di riconfigurare una tabella DynamoDB.		
DynamoDBRestoreFromBackupPolicy	Autorizza il ripristino di una tabella DynamoDB dal backup.		
DynamoDBStreamReadPolicy	Autorizza a descrivere e leggere i flussi e i record di DynamoDB.		
DynamoDBWritePolicy	Fornisce l'autorizzazione di sola scrittura a una tabella DynamoDB.		
EC2CopyImagePolicy	Autorizza a copiare le immagini di Amazon EC2.		
EC2DescribePolicy	Autorizza a descrivere le istanze Amazon Elastic Compute Cloud (Amazon EC2).		
EcsRunTaskPolicy	Fornisce il permesso di iniziare una nuova attività per la definizione di un'attività.		
EFSWriteAccessPolicy	Autorizza il montaggio di un file system Amazon EFS con accesso in scrittura.		
EKSDescribePolicy	Autorizza a descrivere o elencare i cluster Amazon EKS.		

Modello di policy	Description		
ElasticMapReduceAddJobFlowStepsPolicy	Autorizza ad aggiungere nuovi passaggi a un cluster in esecuzione.		
ElasticMapReduceCancelStepsPolicy	Autorizza ad annullare uno o più passaggi in sospeso in un cluster in esecuzione.		
ElasticMapReduceModifyInstanceFleetPolicy	Autorizza a elencare i dettagli e modificare le capacità, ad esempio le flotte all'interno di un cluster.		
ElasticMapReduceModifyInstanceGroupsPolicy	Autorizza a elencare i dettagli e modificare le impostazioni per i gruppi di istanze all'interno di un cluster.		
ElasticMapReduceSetTerminationProtectionPolicy	Autorizza a impostare la protezione dalla terminazione per un cluster.		
ElasticMapReduceTerminateJobFlowsPolicy	Autorizza la chiusura di un cluster.		
ElasticsearchHttpPostPolicy	Fornisce l'autorizzazione POST ad Amazon OpenSearch Service.		

Modello di policy	Description		
EventBridgePutEventsPolicy	Fornisce le autorizzazioni a cui inviare eventi. EventBridge		
FilterLogEventsPolicy	Autorizza a filtrare gli eventi di CloudWatch log da un gruppo di log specificato.		
FirehoseCrudPolicy	Autorizza a creare, scrivere, aggiornare ed eliminare un flusso di distribuzione Firehose.		
FirehoseWritePolicy	Autorizza la scrittura su un flusso di distribuzione Firehose.		
KinesisCrudPolicy	Autorizza a creare, pubblicare ed eliminare uno stream Amazon Kinesis.		
KinesisStreamReadPolicy	Autorizza a elencare e leggere uno stream Amazon Kinesis.		
KMSDecryptPolicy	Autorizza la decrittografia con una chiave AWS Key Management Service ()AWS KMS.		
KMSEncryptPolicy	Dà il permesso di cifrare con una chiave AWS Key Management Service ()AWS KMS.		
LambdaInvokePolicy	Fornisce il permesso di richiamare una AWS Lambda funzione, un alias o una versione.		

Modello di policy	Description		
MobileAnalyticsWriteOnlyAccessPolicy	Fornisce l'autorizzazione di sola scrittura per inserire i dati degli eventi per tutte le risorse dell'applicazione.		
OrganizationsListAccountsPolicy	Fornisce l'autorizzazione di sola lettura per elencare i nomi degli account secondari e. IDs		
PinpointEndpointAccessPolicy	Autorizza a ottenere e aggiornare gli endpoint per un'applicazione Amazon Pinpoint.		
PollyFullAccessPolicy	Fornisce l'autorizzazione di accesso completo alle risorse lessicali di Amazon Polly.		
RekognitionDetectOnlyPolicy	Autorizza il rilevamento di volti, etichette e testo.		
RekognitionFacesManagementPolicy	Autorizza ad aggiungere, eliminare e cercare volti in una raccolta Amazon Rekognition.		
RekognitionFacesPolicy	Autorizza a confrontare e rilevare volti ed etichette.		
RekognitionLabelsPolicy	Autorizza il rilevamento degli oggetti e delle etichette di moderazione.		
RekognitionNoDataAccessPolicy	Autorizza a confrontare e rilevare volti ed etichette.		

Modello di policy	Description		
RekognitionReadPolicy	Autorizza a elencare e cercare volti.		
RekognitionWriteOnlyAccessPolicy	Autorizza a creare volti da collezione e da indicizzare.		
Route53ChangeResourceRecordSetsPolicy	Autorizza a modificare i set di record di risorse in Route 53.		
S3CrudPolicy	Fornisce l'autorizzazione di creazione, lettura, aggiornamento ed eliminazione per agire sugli oggetti in un bucket Amazon S3.		
S3FullAccessPolicy	Fornisce l'autorizzazione di accesso completa per agire sugli oggetti in un bucket Amazon S3.		
S3ReadPolicy	Fornisce l'autorizzazione di sola lettura per leggere oggetti in un bucket Amazon Simple Storage Service (Amazon S3).		
S3WritePolicy	Fornisce il permesso di scrittura per scrivere oggetti in un bucket Amazon S3.		
SageMakerCreateEndpointConfigPolicy	Fornisce il permesso di creare una configurazione di endpoint in AI. SageMaker		
SageMakerCreateEndpointPolicy	Fornisce il permesso di creare un endpoint in SageMaker AI.		

Modello di policy	Description		
ServerlessRepoReadWriteAccessPolicy	Autorizza a creare ed elencare applicazioni nel AWS Serverless Application Repository servizio.		
SESBulkTemplatedCrudPolicy	Autorizza l'invio di e-mail, e-mail basate su modelli, e-mail collettive basate su modelli e verifica l'identità.		
SESBulkTemplatedCrudPolicy_v2	Autorizza a inviare e-mail Amazon SES, e-mail basate su modelli e e-mail collettive basate su modelli e a verificare l'identità.		
SESCrudPolicy	Autorizza l'invio di e-mail e la verifica dell'identità.		
SESEmailTemplateCrudPolicy	Autorizza a creare, ottenere, elencare, aggiornare ed eliminare modelli di e-mail Amazon SES.		
SESSendBouncePolicy	Fornisce SendBounce l'autorizzazione a un'identità Amazon Simple Email Service (Amazon SES).		
SNSCrudPolicy	Autorizza a creare, pubblicare e sottoscrivere argomenti di Amazon SNS.		
SNSPublishMessagePolicy	Autorizza a pubblicare un messaggio su un argomento di Amazon Simple Notification Service (Amazon SNS).		
SQSPollerPolicy	Autorizza il polling di una coda Amazon Simple Queue Service (Amazon SQS).		
SQSSendMessagePolicy	Autorizza l'invio di messaggi a una coda Amazon SQS.		

Modello di policy	Description		
SSMParameterReadPolicy	Fornisce l'autorizzazione ad accedere a un parametro da un archivio di parametri Amazon EC2 Systems Manager (SSM) per caricare i segreti in questo account. Da utilizzare quando il nome del parametro non ha il prefisso slash.		
SSMParameterWithSlashPrefixReadPolicy	Fornisce l'autorizzazione ad accedere a un parametro da un archivio di parametri Amazon EC2 Systems Manager (SSM) per caricare i segreti in questo account. Da utilizzare quando il nome del parametro ha un prefisso slash.		
StepFunctionsExecutionPolicy	Fornisce il permesso di avviare l'esecuzione di una macchina a stati Step Functions .		
TextractDetectAnalyzePolicy	Fornisce l'accesso per rilevare e analizzare documenti con Amazon Textract.		
TextractGetResultPolicy	Fornisce l'accesso per ottenere documenti rilevati e analizzati da Amazon Textract.		
TextractPolicy	Fornisce accesso completo ad Amazon Textract.		
VPCAccessPolicy	Fornisce l'accesso per creare, eliminare, descrivere e scollegare interfacce di rete elastiche.		

Risoluzione dei problemi

Errore CLI SAM: «È necessario specificare valori di parametro validi per il modello di policy policy-template-name '< >'»

Durante l'esecuzione di `sam build`, viene visualizzato il seguente errore:

```
"Must specify valid parameter values for policy template '<policy-template-name>'"
```

Ciò significa che non è stato passato un oggetto vuoto durante la dichiarazione di un modello di policy privo di valori segnaposto.

Per risolvere questo problema, dichiarate la politica come nell'esempio seguente per. CloudWatchPutMetricPolicy

```
MyFunction:
  Policies:
    - CloudWatchPutMetricPolicy: {}
```

AWS SAM elenco di modelli di policy

Di seguito sono riportati i modelli di policy disponibili, insieme alle autorizzazioni applicate a ciascuno di essi. AWS Serverless Application Model (AWS SAM) compila automaticamente gli elementi segnaposto (come AWS Regione e ID account) con le informazioni appropriate.

Argomenti

- AcmGetCertificatePolicy
- AMIDescribePolicy
- AthenaQueryPolicy
- AWSSecretsManagerGetSecretValuePolicy
- AWSSecretsManagerRotationPolicy
- CloudFormationDescribeStacksPolicy
- CloudWatchDashboardPolicy
- CloudWatchDescribeAlarmHistoryPolicy
- CloudWatchPutMetricPolicy
- CodePipelineLambdaExecutionPolicy

- CodePipelineReadOnlyPolicy
- CodeCommitCrudPolicy
- CodeCommitReadPolicy
- ComprehendBasicAccessPolicy
- CostExplorerReadOnlyPolicy
- DynamoDBBackupFullAccessPolicy
- DynamoDBCrudPolicy
- DynamoDBReadPolicy
- DynamoDBReconfigurePolicy
- DynamoDBRestoreFromBackupPolicy
- DynamoDBStreamReadPolicy
- DynamoDBWritePolicy
- EC2CopyImagePolicy
- EC2DescribePolicy
- EcsRunTaskPolicy
- EFSWriteAccessPolicy
- EKSDescribePolicy
- ElasticMapReduceAddJobFlowStepsPolicy
- ElasticMapReduceCancelStepsPolicy
- ElasticMapReduceModifyInstanceFleetPolicy
- ElasticMapReduceModifyInstanceGroupsPolicy
- ElasticMapReduceSetTerminationProtectionPolicy
- ElasticMapReduceTerminateJobFlowsPolicy
- ElasticsearchHttpPostPolicy
- EventBridgePutEventsPolicy
- FilterLogEventsPolicy
- FirehoseCrudPolicy
- FirehoseWritePolicy
- KinesisCrudPolicy
- KinesisStreamReadPolicy

- KMSDecryptPolicy
- KMSEncryptPolicy
- LambdaInvokePolicy
- MobileAnalyticsWriteOnlyAccessPolicy
- OrganizationsListAccountsPolicy
- PinpointEndpointAccessPolicy
- PollyFullAccessPolicy
- RekognitionDetectOnlyPolicy
- RekognitionFacesManagementPolicy
- RekognitionFacesPolicy
- RekognitionLabelsPolicy
- RekognitionNoDataAccessPolicy
- RekognitionReadPolicy
- RekognitionWriteOnlyAccessPolicy
- Route53ChangeResourceRecordSetsPolicy
- S3CrudPolicy
- S3FullAccessPolicy
- S3ReadPolicy
- S3WritePolicy
- SageMakerCreateEndpointConfigPolicy
- SageMakerCreateEndpointPolicy
- ServerlessRepoReadWriteAccessPolicy
- SESBulkTemplatedCrudPolicy
- SESBulkTemplatedCrudPolicy_v2
- SESCrudPolicy
- SESEmailTemplateCrudPolicy
- SESSendBouncePolicy
- SNSCrudPolicy
- SNSPublishMessagePolicy
- SQSPollerPolicy

- SQSSendMessagePolicy
- SSMParameterReadPolicy
- SSMParameterWithSlashPrefixReadPolicy
- StepFunctionsExecutionPolicy
- TextractDetectAnalyzePolicy
- TextractGetResultPolicy
- TextractPolicy
- VPCAccessPolicy

AcmGetCertificatePolicy

Fornisce il permesso di leggere un certificato da. AWS Certificate Manager

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "acm:GetCertificate"
    ],
    "Resource": {
      "Fn::Sub": [
        "${certificateArn}",
        {
          "certificateArn": {
            "Ref": "CertificateArn"
          }
        }
      ]
    }
  }
]
```

AMIDescribePolicy

Autorizza a descrivere Amazon Machine Images (AMIs).

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
```

```
      "ec2:DescribeImages"
    ],
    "Resource": "*"
  }
]
```

AthenaQueryPolicy

Fornisce le autorizzazioni per eseguire le query Athena.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "athena:ListWorkGroups",
      "athena:GetExecutionEngine",
      "athena:GetExecutionEngines",
      "athena:GetNamespace",
      "athena:GetCatalogs",
      "athena:GetNamespaces",
      "athena:GetTables",
      "athena:GetTable"
    ],
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": [
      "athena:StartQueryExecution",
      "athena:GetQueryResults",
      "athena:DeleteNamedQuery",
      "athena:GetNamedQuery",
      "athena:ListQueryExecutions",
      "athena:StopQueryExecution",
      "athena:GetQueryResultsStream",
      "athena:ListNamedQueries",
      "athena:CreateNamedQuery",
      "athena:GetQueryExecution",
      "athena:BatchGetNamedQuery",
      "athena:BatchGetQueryExecution",
      "athena:GetWorkGroup"
    ],
    "Resource": {
      "Fn::Sub": [
```

```
        "arn:${AWS::Partition}:athena:${AWS::Region}:${AWS::AccountId}:workgroup/
${workgroupName}",
        {
          "workgroupName": {
            "Ref": "WorkGroupName"
          }
        }
      ]
    }
  }
]
```

AWSSecretsManagerGetSecretValuePolicy

Autorizza a ottenere il valore segreto per il segreto specificato. Gestione dei segreti AWS

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "secretsmanager:GetSecretValue"
    ],
    "Resource": {
      "Fn::Sub": [
        "${secretArn}",
        {
          "secretArn": {
            "Ref": "SecretArn"
          }
        }
      ]
    }
  }
]
```

AWSSecretsManagerRotationPolicy

Dà il permesso di far ruotare un segreto. Gestione dei segreti AWS

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
```

```
      "secretsmanager:DescribeSecret",
      "secretsmanager:GetSecretValue",
      "secretsmanager:PutSecretValue",
      "secretsmanager:UpdateSecretVersionStage"
    ],
    "Resource": {
      "Fn::Sub": "arn:${AWS::Partition}:secretsmanager:${AWS::Region}:
${AWS::AccountId}:secret:*"
    },
    "Condition": {
      "StringEquals": {
        "secretsmanager:resource/AllowRotationLambdaArn": {
          "Fn::Sub": [
            "arn:${AWS::Partition}:lambda:${AWS::Region}:${AWS::AccountId}:function:
${functionName}",
            {
              "functionName": {
                "Ref": "FunctionName"
              }
            }
          ]
        }
      }
    }
  },
  {
    "Effect": "Allow",
    "Action": [
      "secretsmanager:GetRandomPassword"
    ],
    "Resource": "*"
  }
]
```

CloudFormationDescribeStacksPolicy

Dà il permesso di descrivere le CloudFormation pile.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "cloudformation:DescribeStacks"
    ],
```

```
    "Resource": {
      "Fn::Sub": "arn:${AWS::Partition}:cloudformation:${AWS::Region}:
${AWS::AccountId}:stack/*"
    }
  }
]
```

CloudWatchDashboardPolicy

Fornisce le autorizzazioni per inserire le metriche in modo che funzionino nei dashboard. CloudWatch

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "cloudwatch:GetDashboard",
      "cloudwatch:ListDashboards",
      "cloudwatch:PutDashboard",
      "cloudwatch:ListMetrics"
    ],
    "Resource": "*"
  }
]
```

CloudWatchDescribeAlarmHistoryPolicy

Autorizza a descrivere la cronologia degli CloudWatch allarmi di Amazon.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "cloudwatch:DescribeAlarmHistory"
    ],
    "Resource": "*"
  }
]
```

CloudWatchPutMetricPolicy

Autorizza a inviare metriche a. CloudWatch

```
"Statement": [
```

```
  {
    "Effect": "Allow",
    "Action": [
      "cloudwatch:PutMetricData"
    ],
    "Resource": "*"
  }
]
```

CodePipelineLambdaExecutionPolicy

Fornisce l'autorizzazione a una funzione Lambda richiamata da AWS CodePipeline per segnalare lo stato del lavoro.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "codepipeline:PutJobSuccessResult",
      "codepipeline:PutJobFailureResult"
    ],
    "Resource": "*"
  }
]
```

CodePipelineReadOnlyPolicy

Fornisce il permesso di lettura per ottenere dettagli su una CodePipeline pipeline.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "codepipeline:ListPipelineExecutions"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:codepipeline:${AWS::Region}:${AWS::AccountId}:
${pipelinename}",
        {
          "pipelinename": {
            "Ref": "PipelineName"
          }
```

```
        }
      ]
    }
  }
]
```

CodeCommitCrudPolicy

Fornisce le autorizzazioni per creare, leggere, aggiornare ed eliminare oggetti all'interno di un CodeCommit repository specifico.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "codecommit:GitPull",
      "codecommit:GitPush",
      "codecommit:CreateBranch",
      "codecommit:DeleteBranch",
      "codecommit:GetBranch",
      "codecommit:ListBranches",
      "codecommit:MergeBranchesByFastForward",
      "codecommit:MergeBranchesBySquash",
      "codecommit:MergeBranchesByThreeWay",
      "codecommit:UpdateDefaultBranch",
      "codecommit:BatchDescribeMergeConflicts",
      "codecommit:CreateUnreferencedMergeCommit",
      "codecommit:DescribeMergeConflicts",
      "codecommit:GetMergeCommit",
      "codecommit:GetMergeOptions",
      "codecommit:BatchGetPullRequests",
      "codecommit:CreatePullRequest",
      "codecommit:DescribePullRequestEvents",
      "codecommit:GetCommentsForPullRequest",
      "codecommit:GetCommitsFromMergeBase",
      "codecommit:GetMergeConflicts",
      "codecommit:GetPullRequest",
      "codecommit:ListPullRequests",
      "codecommit:MergePullRequestByFastForward",
      "codecommit:MergePullRequestBySquash",
      "codecommit:MergePullRequestByThreeWay",
      "codecommit:PostCommentForPullRequest",
      "codecommit:UpdatePullRequestDescription",
```

```
      "codecommit:UpdatePullRequestStatus",
      "codecommit:UpdatePullRequestTitle",
      "codecommit:DeleteFile",
      "codecommit:GetBlob",
      "codecommit:GetFile",
      "codecommit:GetFolder",
      "codecommit:PutFile",
      "codecommit:DeleteCommentContent",
      "codecommit:GetComment",
      "codecommit:GetCommentsForComparedCommit",
      "codecommit:PostCommentForComparedCommit",
      "codecommit:PostCommentReply",
      "codecommit:UpdateComment",
      "codecommit:BatchGetCommits",
      "codecommit:CreateCommit",
      "codecommit:GetCommit",
      "codecommit:GetCommitHistory",
      "codecommit:GetDifferences",
      "codecommit:GetObjectIdentifier",
      "codecommit:GetReferences",
      "codecommit:GetTree",
      "codecommit:GetRepository",
      "codecommit:UpdateRepositoryDescription",
      "codecommit:ListTagsForResource",
      "codecommit:TagResource",
      "codecommit:UntagResource",
      "codecommit:GetRepositoryTriggers",
      "codecommit:PutRepositoryTriggers",
      "codecommit:TestRepositoryTriggers",
      "codecommit:GetBranch",
      "codecommit:GetCommit",
      "codecommit:UploadArchive",
      "codecommit:GetUploadArchiveStatus",
      "codecommit:CancelUploadArchive"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:codecommit:${AWS::Region}:${AWS::AccountId}:
${repositoryName}",
        {
          "repositoryName": {
            "Ref": "RepositoryName"
          }
        }
```

```
      ]
    }
  }
]
```

CodeCommitReadPolicy

Fornisce le autorizzazioni per leggere oggetti all'interno di un CodeCommit repository specifico.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "codecommit:GitPull",
      "codecommit:GetBranch",
      "codecommit:ListBranches",
      "codecommit:BatchDescribeMergeConflicts",
      "codecommit:DescribeMergeConflicts",
      "codecommit:GetMergeCommit",
      "codecommit:GetMergeOptions",
      "codecommit:BatchGetPullRequests",
      "codecommit:DescribePullRequestEvents",
      "codecommit:GetCommentsForPullRequest",
      "codecommit:GetCommitsFromMergeBase",
      "codecommit:GetMergeConflicts",
      "codecommit:GetPullRequest",
      "codecommit:ListPullRequests",
      "codecommit:GetBlob",
      "codecommit:GetFile",
      "codecommit:GetFolder",
      "codecommit:GetComment",
      "codecommit:GetCommentsForComparedCommit",
      "codecommit:BatchGetCommits",
      "codecommit:GetCommit",
      "codecommit:GetCommitHistory",
      "codecommit:GetDifferences",
      "codecommit:GetObjectIdentifier",
      "codecommit:GetReferences",
      "codecommit:GetTree",
      "codecommit:GetRepository",
      "codecommit:ListTagsForResource",
      "codecommit:GetRepositoryTriggers",
      "codecommit:TestRepositoryTriggers",
      "codecommit:GetBranch",
```

```
      "codecommit:GetCommit",
      "codecommit:GetUploadArchiveStatus"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:codecommit:${AWS::Region}:${AWS::AccountId}:
${repositoryName}",
        {
          "repositoryName": {
            "Ref": "RepositoryName"
          }
        }
      ]
    }
  }
]
```

ComprehendBasicAccessPolicy

Autorizza il rilevamento di entità, frasi chiave, lingue e sentimenti.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "comprehend:BatchDetectKeyPhrases",
      "comprehend:DetectDominantLanguage",
      "comprehend:DetectEntities",
      "comprehend:BatchDetectEntities",
      "comprehend:DetectKeyPhrases",
      "comprehend:DetectSentiment",
      "comprehend:BatchDetectDominantLanguage",
      "comprehend:BatchDetectSentiment"
    ],
    "Resource": "*"
  }
]
```

CostExplorerReadOnlyPolicy

Fornisce l'autorizzazione di sola lettura alla modalità di sola lettura (Cost AWS Cost Explorer Explorer) APIs per la cronologia di fatturazione.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "ce:GetCostAndUsage",
      "ce:GetDimensionValues",
      "ce:GetReservationCoverage",
      "ce:GetReservationPurchaseRecommendation",
      "ce:GetReservationUtilization",
      "ce:GetTags"
    ],
    "Resource": "*"
  }
]
```

DynamoDBBackupFullAccessPolicy

Fornisce l'autorizzazione di lettura e scrittura ai backup su richiesta di DynamoDB per una tabella.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "dynamodb:CreateBackup",
      "dynamodb:DescribeContinuousBackups"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}",
        {
          "tableName": {
            "Ref": "TableName"
          }
        }
      ]
    }
  },
  {
    "Effect": "Allow",
    "Action": [
      "dynamodb:DeleteBackup",
      "dynamodb:DescribeBackup",
```

```
      "dynamodb:ListBackups"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}/backup/*",
        {
          "tableName": {
            "Ref": "TableName"
          }
        }
      ]
    }
  }
]
```

DynamoDBCrudPolicy

Fornisce autorizzazioni di creazione, lettura, aggiornamento ed eliminazione per una tabella Amazon DynamoDB.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "dynamodb:GetItem",
      "dynamodb:DeleteItem",
      "dynamodb:PutItem",
      "dynamodb:Scan",
      "dynamodb:Query",
      "dynamodb:UpdateItem",
      "dynamodb:BatchWriteItem",
      "dynamodb:BatchGetItem",
      "dynamodb:DescribeTable",
      "dynamodb:ConditionCheckItem"
    ],
    "Resource": [
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}",
          {
            "tableName": {
```

```
              "Ref": "TableName"
            }
          }
        ]
      },
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}/index/*",
          {
            "tableName": {
              "Ref": "TableName"
            }
          }
        ]
      }
    ]
  }
]
```

DynamoDBReadPolicy

Fornisce l'autorizzazione di sola lettura a una tabella DynamoDB.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "dynamodb:GetItem",
      "dynamodb:Scan",
      "dynamodb:Query",
      "dynamodb:BatchGetItem",
      "dynamodb:DescribeTable"
    ],
    "Resource": [
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}",
          {
            "tableName": {
              "Ref": "TableName"
            }
          }
```

```
        ]
      },
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}/index/*",
          {
            "tableName": {
              "Ref": "TableName"
            }
          }
        ]
      }
    ]
  }
]
```

DynamoDBReconfigurePolicy

Fornisce il permesso di riconfigurare una tabella DynamoDB.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "dynamodb:UpdateTable"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}",
        {
          "tableName": {
            "Ref": "TableName"
          }
        }
      ]
    }
  }
]
```

DynamoDBRestoreFromBackupPolicy

Autorizza il ripristino di una tabella DynamoDB dal backup.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "dynamodb:RestoreTableFromBackup"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}/backup/*",
        {
          "tableName": {
            "Ref": "TableName"
          }
        }
      ]
    }
  },
  {
    "Effect": "Allow",
    "Action": [
      "dynamodb:PutItem",
      "dynamodb:UpdateItem",
      "dynamodb:DeleteItem",
      "dynamodb:GetItem",
      "dynamodb:Query",
      "dynamodb:Scan",
      "dynamodb:BatchWriteItem"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}",
        {
          "tableName": {
            "Ref": "TableName"
          }
        }
      ]
    }
```

```
  }
]
```

DynamoDBStreamReadPolicy

Autorizza a descrivere e leggere i flussi e i record di DynamoDB.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "dynamodb:DescribeStream",
      "dynamodb:GetRecords",
      "dynamodb:GetShardIterator"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}/stream/${streamName}",
        {
          "tableName": {
            "Ref": "TableName"
          },
          "streamName": {
            "Ref": "StreamName"
          }
        }
      ]
    }
  },
  {
    "Effect": "Allow",
    "Action": [
      "dynamodb:ListStreams"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}/stream/*",
        {
          "tableName": {
            "Ref": "TableName"
          }
        }
```

```
      ]
    }
  }
]
```

DynamoDBWritePolicy

Fornisce l'autorizzazione di sola scrittura a una tabella DynamoDB.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "dynamodb:PutItem",
      "dynamodb:UpdateItem",
      "dynamodb:BatchWriteItem"
    ],
    "Resource": [
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}",
          {
            "tableName": {
              "Ref": "TableName"
            }
          }
        ]
      },
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}/index/*",
          {
            "tableName": {
              "Ref": "TableName"
            }
          }
        ]
      }
    ]
  }
]
```

EC2CopyImagePolicy

Autorizza a copiare EC2 le immagini di Amazon.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "ec2:CopyImage"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:ec2:${AWS::Region}:${AWS::AccountId}:image/${imageId}",
        {
          "imageId": {
            "Ref": "ImageId"
          }
        }
      ]
    }
  }
]
```

EC2DescribePolicy

Autorizza a descrivere le istanze di Amazon Elastic Compute Cloud (Amazon EC2).

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "ec2:DescribeRegions",
      "ec2:DescribeInstances"
    ],
    "Resource": "*"
  }
]
```

EcsRunTaskPolicy

Dà il permesso di iniziare una nuova attività per la definizione di un'attività.

```
"Statement": [
```

```
  {
    "Action": [
      "ecs:RunTask"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:ecs:${AWS::Region}:${AWS::AccountId}:task-definition/
${taskDefinition}",
        {
          "taskDefinition": {
            "Ref": "TaskDefinition"
          }
        }
      ]
    },
    "Effect": "Allow"
  }
]
```

EFSWriteAccessPolicy

Autorizza il montaggio di un file system Amazon EFS con accesso in scrittura.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "elasticfilesystem:ClientMount",
      "elasticfilesystem:ClientWrite"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:elasticfilesystem:${AWS::Region}:${AWS::AccountId}:file-
system/${FileSystem}",
        {
          "FileSystem": {
            "Ref": "FileSystem"
          }
        }
      ]
    },
    "Condition": {
      "StringEquals": {
        "elasticfilesystem:AccessPointArn": {
```

```
          "Fn::Sub": [
            "arn:${AWS::Partition}:elasticfilesystem:${AWS::Region}:
${AWS::AccountId}:access-point/${AccessPoint}",
            {
              "AccessPoint": {
                "Ref": "AccessPoint"
              }
            }
          ]
        }
      }
    }
  }
]
```

EKSDescribePolicy

Autorizza a descrivere o elencare i cluster Amazon Elastic Kubernetes Service (Amazon EKS).

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "eks:DescribeCluster",
      "eks:ListClusters"
    ],
    "Resource": "*"
  }
]
```

ElasticMapReduceAddJobFlowStepsPolicy

Autorizza ad aggiungere nuovi passaggi a un cluster in esecuzione.

```
"Statement": [
  {
    "Action": "elasticmapreduce:AddJobFlowSteps",
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:elasticmapreduce:${AWS::Region}:
${AWS::AccountId}:cluster/${clusterId}",
        {
          "clusterId": {
```

```
            "Ref": "ClusterId"
          }
        }
      ]
    },
    "Effect": "Allow"
  }
]
```

ElasticMapReduceCancelStepsPolicy

Autorizza l'annullamento di uno o più passaggi in sospeso in un cluster in esecuzione.

```
"Statement": [
  {
    "Action": "elasticmapreduce:CancelSteps",
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:elasticmapreduce:${AWS::Region}:
${AWS::AccountId}:cluster/${clusterId}",
        {
          "clusterId": {
            "Ref": "ClusterId"
          }
        }
      ]
    },
    "Effect": "Allow"
  }
]
```

ElasticMapReduceModifyInstanceFleetPolicy

Autorizza a elencare i dettagli e modificare le capacità, ad esempio le flotte all'interno di un cluster.

```
"Statement": [
  {
    "Action": [
      "elasticmapreduce:ModifyInstanceFleet",
      "elasticmapreduce:ListInstanceFleets"
    ],
    "Resource": {
```

```
      "Fn::Sub": [
        "arn:${AWS::Partition}:elasticmapreduce:${AWS::Region}:
${AWS::AccountId}:cluster/${clusterId}",
        {
          "clusterId": {
            "Ref": "ClusterId"
          }
        }
      ]
    },
    "Effect": "Allow"
  }
]
```

ElasticMapReduceModifyInstanceGroupsPolicy

Autorizza a elencare i dettagli e modificare le impostazioni per i gruppi di istanze all'interno di un cluster.

```
"Statement": [
  {
    "Action": [
      "elasticmapreduce:ModifyInstanceGroups",
      "elasticmapreduce:ListInstanceGroups"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:elasticmapreduce:${AWS::Region}:
${AWS::AccountId}:cluster/${clusterId}",
        {
          "clusterId": {
            "Ref": "ClusterId"
          }
        }
      ]
    },
    "Effect": "Allow"
  }
]
```

ElasticMapReduceSetTerminationProtectionPolicy

Fornisce il permesso di impostare la protezione dalla terminazione per un cluster.

```
"Statement": [
  {
    "Action": "elasticmapreduce:SetTerminationProtection",
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:elasticmapreduce:${AWS::Region}:
${AWS::AccountId}:cluster/${clusterId}",
        {
          "clusterId": {
            "Ref": "ClusterId"
          }
        }
      ]
    },
    "Effect": "Allow"
  }
]
```

ElasticMapReduceTerminateJobFlowsPolicy

Autorizza la chiusura di un cluster.

```
"Statement": [
  {
    "Action": "elasticmapreduce:TerminateJobFlows",
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:elasticmapreduce:${AWS::Region}:
${AWS::AccountId}:cluster/${clusterId}",
        {
          "clusterId": {
            "Ref": "ClusterId"
          }
        }
      ]
    },
    "Effect": "Allow"
  }
]
```

ElasticsearchHttpPostPolicy

Fornisce le autorizzazioni POST e PUT ad Amazon OpenSearch Service.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "es:ESHttpPost",
      "es:ESHttpPut"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:es:${AWS::Region}:${AWS::AccountId}:domain/
${domainName}/*",
        {
          "domainName": {
            "Ref": "DomainName"
          }
        }
      ]
    }
  }
]
```

EventBridgePutEventsPolicy

Fornisce le autorizzazioni per inviare eventi ad Amazon EventBridge.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": "events:PutEvents",
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:events:${AWS::Region}:${AWS::AccountId}:event-bus/
${eventBusName}",
        {
          "eventBusName": {
            "Ref": "EventBusName"
          }
        }
      ]
    }
  }
]
```

FilterLogEventsPolicy

Autorizza a filtrare gli eventi di CloudWatch Logs da un gruppo di log specificato.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "logs:FilterLogEvents"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:logs:${AWS::Region}:${AWS::AccountId}:log-group:
${logGroupName}:log-stream:*",
        {
          "logGroupName": {
            "Ref": "LogGroupName"
          }
        }
      ]
    }
  }
]
```

FirehoseCrudPolicy

Autorizza a creare, scrivere, aggiornare ed eliminare un flusso di distribuzione Firehose.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "firehose:CreateDeliveryStream",
      "firehose:DeleteDeliveryStream",
      "firehose:DescribeDeliveryStream",
      "firehose:PutRecord",
      "firehose:PutRecordBatch",
      "firehose:UpdateDestination"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:firehose:${AWS::Region}:
${AWS::AccountId}:deliverystream/${deliveryStreamName}",
```

```
        {
          "deliveryStreamName": {
            "Ref": "DeliveryStreamName"
          }
        }
      ]
    }
  }
]
```

FirehoseWritePolicy

Autorizza la scrittura su un flusso di distribuzione Firehose.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "firehose:PutRecord",
      "firehose:PutRecordBatch"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:firehose:${AWS::Region}:
${AWS::AccountId}:deliverystream/${deliveryStreamName}",
        {
          "deliveryStreamName": {
            "Ref": "DeliveryStreamName"
          }
        }
      ]
    }
  }
]
```

KinesisCrudPolicy

Autorizza a creare, pubblicare ed eliminare uno stream Amazon Kinesis.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
```

```
      "kinesis:AddTagsToStream",
      "kinesis:CreateStream",
      "kinesis:DecreaseStreamRetentionPeriod",
      "kinesis:DeleteStream",
      "kinesis:DescribeStream",
      "kinesis:DescribeStreamSummary",
      "kinesis:GetShardIterator",
      "kinesis:IncreaseStreamRetentionPeriod",
      "kinesis:ListTagsForStream",
      "kinesis:MergeShards",
      "kinesis:PutRecord",
      "kinesis:PutRecords",
      "kinesis:SplitShard",
      "kinesis:RemoveTagsFromStream"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:kinesis:${AWS::Region}:${AWS::AccountId}:stream/
${streamName}",
        {
          "streamName": {
            "Ref": "StreamName"
          }
        }
      ]
    }
  }
]
```

KinesisStreamReadPolicy

Autorizza a elencare e leggere uno stream Amazon Kinesis.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "kinesis:ListStreams",
      "kinesis:DescribeLimits"
    ],
    "Resource": {
      "Fn::Sub": "arn:${AWS::Partition}:kinesis:${AWS::Region}:
${AWS::AccountId}:stream/*"
    }
```

```
  },
  {
    "Effect": "Allow",
    "Action": [
      "kinesis:DescribeStream",
      "kinesis:DescribeStreamSummary",
      "kinesis:GetRecords",
      "kinesis:GetShardIterator"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:kinesis:${AWS::Region}:${AWS::AccountId}:stream/
${streamName}",
        {
          "streamName": {
            "Ref": "StreamName"
          }
        }
      ]
    }
  }
]
```

KMSDecryptPolicy

Dà il permesso di decrittografare con una AWS Key Management Service chiave ().AWS KMS Nota che `keyId` deve essere un ID di AWS KMS chiave e non un alias di chiave.

```
"Statement": [
  {
    "Action": "kms:Decrypt",
    "Effect": "Allow",
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:kms:${AWS::Region}:${AWS::AccountId}:key/${keyId}",
        {
          "keyId": {
            "Ref": "KeyId"
          }
        }
      ]
    }
  }
```

```
]
```

KMSEncryptPolicy

Autorizza la crittografia con una AWS KMS chiave. Nota che keyID deve essere un ID AWS KMS chiave e non un alias di chiave.

```
"Statement": [
  {
    "Action": "kms:Encrypt",
    "Effect": "Allow",
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:kms:${AWS::Region}:${AWS::AccountId}:key/${keyId}",
        {
          "keyId": {
            "Ref": "KeyId"
          }
        }
      ]
    }
  }
]
```

LambdaInvokePolicy

Fornisce il permesso di richiamare una AWS Lambda funzione, un alias o una versione.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "lambda:InvokeFunction"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:lambda:${AWS::Region}:${AWS::AccountId}:function:
${functionName}*",
        {
          "functionName": {
            "Ref": "FunctionName"
          }
        }
```

```
      ]
    }
  }
]
```

MobileAnalyticsWriteOnlyAccessPolicy

Fornisce l'autorizzazione di sola scrittura per inserire i dati degli eventi per tutte le risorse dell'applicazione.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "mobileanalytics:PutEvents"
    ],
    "Resource": "*"
  }
]
```

OrganizationsListAccountsPolicy

Fornisce l'autorizzazione di sola lettura per elencare i nomi degli account secondari e. IDs

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "organizations:ListAccounts"
    ],
    "Resource": "*"
  }
]
```

PinpointEndpointAccessPolicy

Autorizza a ottenere e aggiornare gli endpoint per un'applicazione Amazon Pinpoint.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "mobiletargeting:GetEndpoint",
```

```
      "mobiletargeting:UpdateEndpoint",
      "mobiletargeting:UpdateEndpointsBatch"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:mobiletargeting:${AWS::Region}:${AWS::AccountId}:apps/
${pinpointApplicationId}/endpoints/*",
        {
          "pinpointApplicationId": {
            "Ref": "PinpointApplicationId"
          }
        }
      ]
    }
  }
]
```

PollyFullAccessPolicy

Fornisce l'autorizzazione di accesso completo alle risorse lessicali di Amazon Polly.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "polly:GetLexicon",
      "polly:DeleteLexicon"
    ],
    "Resource": [
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:polly:${AWS::Region}:${AWS::AccountId}:lexicon/
${lexiconName}",
          {
            "lexiconName": {
              "Ref": "LexiconName"
            }
          }
        ]
      }
    ]
  },
  {
    "Effect": "Allow",
```

```
    "Action": [
      "polly:DescribeVoices",
      "polly:ListLexicons",
      "polly:PutLexicon",
      "polly:SynthesizeSpeech"
    ],
    "Resource": [
      {
        "Fn::Sub": "arn:${AWS::Partition}:polly:${AWS::Region}:
${AWS::AccountId}:lexicon/*"
      }
    ]
  }
]
```

RekognitionDetectOnlyPolicy

Autorizza a rilevare volti, etichette e testo.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "rekognition:DetectFaces",
      "rekognition:DetectLabels",
      "rekognition:DetectModerationLabels",
      "rekognition:DetectText"
    ],
    "Resource": "*"
  }
]
```

RekognitionFacesManagementPolicy

Autorizza ad aggiungere, eliminare e cercare volti in una raccolta Amazon Rekognition.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "rekognition:IndexFaces",
      "rekognition:DeleteFaces",
      "rekognition:SearchFaces",
```

```
      "rekognition:SearchFacesByImage",
      "rekognition:ListFaces"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:rekognition:${AWS::Region}:${AWS::AccountId}:collection/
${collectionId}",
        {
          "collectionId": {
            "Ref": "CollectionId"
          }
        }
      ]
    }
  }
]
```

RekognitionFacesPolicy

Autorizza il confronto e il rilevamento di volti ed etichette.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "rekognition:CompareFaces",
      "rekognition:DetectFaces"
    ],
    "Resource": "*"
  }
]
```

RekognitionLabelsPolicy

Autorizza il rilevamento degli oggetti e delle etichette di moderazione.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "rekognition:DetectLabels",
      "rekognition:DetectModerationLabels"
    ],
```

```
    "Resource": "*"
  }
]
```

RekognitionNoDataAccessPolicy

Autorizza a confrontare e rilevare volti ed etichette.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "rekognition:CompareFaces",
      "rekognition:DetectFaces",
      "rekognition:DetectLabels",
      "rekognition:DetectModerationLabels"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:rekognition:${AWS::Region}:${AWS::AccountId}:collection/
${collectionId}",
        {
          "collectionId": {
            "Ref": "CollectionId"
          }
        }
      ]
    }
  }
]
```

RekognitionReadPolicy

Autorizza a elencare e cercare volti.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "rekognition:ListCollections",
      "rekognition:ListFaces",
      "rekognition:SearchFaces",
      "rekognition:SearchFacesByImage"
```

```
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:rekognition:${AWS::Region}:${AWS::AccountId}:collection/
${collectionId}",
        {
          "collectionId": {
            "Ref": "CollectionId"
          }
        }
      ]
    }
  }
]
```

RekognitionWriteOnlyAccessPolicy

Autorizza a creare volti da collezione e da indicizzare.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "rekognition:CreateCollection",
      "rekognition:IndexFaces"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:rekognition:${AWS::Region}:${AWS::AccountId}:collection/
${collectionId}",
        {
          "collectionId": {
            "Ref": "CollectionId"
          }
        }
      ]
    }
  }
]
```

Route53ChangeResourceRecordSetsPolicy

Autorizza a modificare i set di record di risorse in Route 53.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "route53:ChangeResourceRecordSets"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:route53:::hostedzone/${HostedZoneId}",
        {
          "HostedZoneId": {
            "Ref": "HostedZoneId"
          }
        }
      ]
    }
  }
]
```

S3CrudPolicy

Fornisce l'autorizzazione di creazione, lettura, aggiornamento ed eliminazione per agire sugli oggetti in un bucket Amazon S3.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:ListBucket",
      "s3:GetBucketLocation",
      "s3:GetObjectVersion",
      "s3:PutObject",
      "s3:PutObjectAcl",
      "s3:GetLifecycleConfiguration",
      "s3:PutLifecycleConfiguration",
      "s3:DeleteObject"
    ],
    "Resource": [
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:s3:::${bucketName}",
          {
```

```
            "bucketName": {
              "Ref": "BucketName"
            }
          }
        ]
      },
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:s3:::${bucketName}/*",
          {
            "bucketName": {
              "Ref": "BucketName"
            }
          }
        ]
      }
    ]
  }
]
```

S3FullAccessPolicy

Fornisce l'autorizzazione di accesso completa per agire sugli oggetti in un bucket Amazon S3.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:GetObjectAcl",
      "s3:GetObjectVersion",
      "s3:PutObject",
      "s3:PutObjectAcl",
      "s3:DeleteObject",
      "s3:DeleteObjectTagging",
      "s3:DeleteObjectVersionTagging",
      "s3:GetObjectTagging",
      "s3:GetObjectVersionTagging",
      "s3:PutObjectTagging",
      "s3:PutObjectVersionTagging"
    ],
    "Resource": [
      {
        "Fn::Sub": [
```

```
          "arn:${AWS::Partition}:s3:::${bucketName}/*",
          {
            "bucketName": {
              "Ref": "BucketName"
            }
          }
        ]
      }
    ]
  },
  {
    "Effect": "Allow",
    "Action": [
      "s3:ListBucket",
      "s3:GetBucketLocation",
      "s3:GetLifecycleConfiguration",
      "s3:PutLifecycleConfiguration"
    ],
    "Resource": [
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:s3:::${bucketName}",
          {
            "bucketName": {
              "Ref": "BucketName"
            }
          }
        ]
      }
    ]
  }
]
```

S3ReadPolicy

Fornisce l'autorizzazione di sola lettura per leggere oggetti in un bucket Amazon Simple Storage Service (Amazon S3).

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
```

```
      "s3:ListBucket",
      "s3:GetBucketLocation",
      "s3:GetObjectVersion",
      "s3:GetLifecycleConfiguration"
    ],
    "Resource": [
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:s3:::${bucketName}",
          {
            "bucketName": {
              "Ref": "BucketName"
            }
          }
        ]
      },
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:s3:::${bucketName}/*",
          {
            "bucketName": {
              "Ref": "BucketName"
            }
          }
        ]
      }
    ]
  }
]
```

S3WritePolicy

Fornisce il permesso di scrittura per scrivere oggetti in un bucket Amazon S3.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "s3:PutObject",
      "s3:PutObjectAcl",
      "s3:PutLifecycleConfiguration"
    ],
    "Resource": [
      {
```

```
        "Fn::Sub": [
          "arn:${AWS::Partition}:s3:::${bucketName}",
          {
            "bucketName": {
              "Ref": "BucketName"
            }
          }
        ]
      },
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:s3:::${bucketName}/*",
          {
            "bucketName": {
              "Ref": "BucketName"
            }
          }
        ]
      }
    ]
  }
]
```

SageMakerCreateEndpointConfigPolicy

Fornisce il permesso di creare una configurazione degli endpoint in SageMaker AI.

```
"Statement": [
  {
    "Action": [
      "sagemaker:CreateEndpointConfig"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:sagemaker:${AWS::Region}:${AWS::AccountId}:endpoint-
config/${endpointConfigName}",
        {
          "endpointConfigName": {
            "Ref": "EndpointConfigName"
          }
        }
      ]
    },
    "Effect": "Allow"
```

```
  }
]
```

SageMakerCreateEndpointPolicy

Dà il permesso di creare un endpoint in SageMaker AI.

```
"Statement": [
  {
    "Action": [
      "sagemaker:CreateEndpoint"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:sagemaker:${AWS::Region}:${AWS::AccountId}:endpoint/
${endpointName}",
        {
          "endpointName": {
            "Ref": "EndpointName"
          }
        }
      ]
    },
    "Effect": "Allow"
  }
]
```

ServerlessRepoReadWriteAccessPolicy

Autorizza a creare ed elencare applicazioni nel servizio AWS Serverless Application Repository (AWS SAM).

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "serverlessrepo:CreateApplication",
      "serverlessrepo:CreateApplicationVersion",
      "serverlessrepo:GetApplication",
      "serverlessrepo:ListApplications",
      "serverlessrepo:ListApplicationVersions"
    ],
```

```
    "Resource": [
      {
        "Fn::Sub": "arn:${AWS::Partition}:serverlessrepo:${AWS::Region}:
${AWS::AccountId}:applications/*"
      }
    ]
  }
]
```

SESBulkTemplatedCrudPolicy

Autorizza a inviare e-mail Amazon SES, e-mail basate su modelli e e-mail collettive basate su modelli e a verificare l'identità.

> **Note**
>
> L'`ses:SendTemplatedEmail`azione richiede un modello ARN. Usare invece `SESBulkTemplatedCrudPolicy_v2`.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "ses:GetIdentityVerificationAttributes",
      "ses:SendEmail",
      "ses:SendRawEmail",
      "ses:SendTemplatedEmail",
      "ses:SendBulkTemplatedEmail",
      "ses:VerifyEmailIdentity"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:ses:${AWS::Region}:${AWS::AccountId}:identity/
${identityName}",
        {
          "identityName": {
            "Ref": "IdentityName"
          }
        }
      ]
    }
```

```
  }
]
```

SESBulkTemplatedCrudPolicy_v2

Autorizza a inviare e-mail Amazon SES, e-mail basate su modelli e e-mail collettive basate su modelli e a verificare l'identità.

```
"Statement": [
  {
    "Action": [
      "ses:SendEmail",
      "ses:SendRawEmail",
      "ses:SendTemplatedEmail",
      "ses:SendBulkTemplatedEmail"
    ],
    "Effect": "Allow",
    "Resource": [
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:ses:${AWS::Region}:${AWS::AccountId}:identity/
${identityName}",
          {
            "identityName": {
              "Ref": "IdentityName"
            }
          }
        ]
      },
      {
        "Fn::Sub": [
          "arn:${AWS::Partition}:ses:${AWS::Region}:${AWS::AccountId}:template/
${templateName}",
          {
            "templateName": {
              "Ref": "TemplateName"
            }
          }
        ]
      }
    ]
  },
  {
```

```
    "Action": [
      "ses:GetIdentityVerificationAttributes",
      "ses:VerifyEmailIdentity"
    ],
    "Effect": "Allow",
    "Resource": "*"
  }
]
```

SESCrudPolicy

Autorizza l'invio di e-mail e la verifica dell'identità.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "ses:GetIdentityVerificationAttributes",
      "ses:SendEmail",
      "ses:SendRawEmail",
      "ses:VerifyEmailIdentity"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:ses:${AWS::Region}:${AWS::AccountId}:identity/
${identityName}",
        {
          "identityName": {
            "Ref": "IdentityName"
          }
        }
      ]
    }
  }
]
```

SESEmailTemplateCrudPolicy

Autorizza a creare, ottenere, elencare, aggiornare ed eliminare modelli di e-mail Amazon SES.

```
"Statement": [
  {
    "Effect": "Allow",
```

```
    "Action": [
      "ses:CreateTemplate",
      "ses:GetTemplate",
      "ses:ListTemplates",
      "ses:UpdateTemplate",
      "ses:DeleteTemplate",
      "ses:TestRenderTemplate"
    ],
    "Resource": "*"
  }
]
```

SESSendBouncePolicy

Fornisce SendBounce l'autorizzazione a un'identità Amazon Simple Email Service (Amazon SES).

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "ses:SendBounce"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:ses:${AWS::Region}:${AWS::AccountId}:identity/
${identityName}",
        {
          "identityName": {
            "Ref": "IdentityName"
          }
        }
      ]
    }
  }
]
```

SNSCrudPolicy

Autorizza a creare, pubblicare e sottoscrivere argomenti di Amazon SNS.

```
"Statement": [
  {
    "Effect": "Allow",
```

```
    "Action": [
      "sns:ListSubscriptionsByTopic",
      "sns:CreateTopic",
      "sns:SetTopicAttributes",
      "sns:Subscribe",
      "sns:Publish"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:sns:${AWS::Region}:${AWS::AccountId}:${topicName}*",
        {
          "topicName": {
            "Ref": "TopicName"
          }
        }
      ]
    }
  }
]
```

SNSPublishMessagePolicy

Autorizza a pubblicare un messaggio su un argomento di Amazon Simple Notification Service (Amazon SNS).

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "sns:Publish"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:sns:${AWS::Region}:${AWS::AccountId}:${topicName}",
        {
          "topicName": {
            "Ref": "TopicName"
          }
        }
      ]
    }
  }
]
```

SQSPollerPolicy

Autorizza il polling di una coda Amazon Simple Queue Service (Amazon SQS).

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "sqs:ChangeMessageVisibility",
      "sqs:ChangeMessageVisibilityBatch",
      "sqs:DeleteMessage",
      "sqs:DeleteMessageBatch",
      "sqs:GetQueueAttributes",
      "sqs:ReceiveMessage"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:sqs:${AWS::Region}:${AWS::AccountId}:${queueName}",
        {
          "queueName": {
            "Ref": "QueueName"
          }
        }
      ]
    }
  }
]
```

SQSSendMessagePolicy

Autorizza l'invio di messaggi a una coda Amazon SQS.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "sqs:SendMessage*"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:sqs:${AWS::Region}:${AWS::AccountId}:${queueName}",
        {
          "queueName": {
```

```
            "Ref": "QueueName"
          }
        }
      ]
    }
  }
]
```

SSMParameterReadPolicy

Fornisce l'autorizzazione ad accedere a un parametro da un archivio di parametri Amazon EC2 Systems Manager (SSM) per caricare i segreti in questo account. Da utilizzare quando il nome del parametro non ha il prefisso slash.

> **Note**
>
> Se non si utilizza la chiave predefinita, sarà necessaria anche la `KMSDecryptPolicy` politica.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "ssm:DescribeParameters"
    ],
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": [
      "ssm:GetParameters",
      "ssm:GetParameter",
      "ssm:GetParametersByPath"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:ssm:${AWS::Region}:${AWS::AccountId}:parameter/
${parameterName}",
        {
          "parameterName": {
            "Ref": "ParameterName"
```

```
          }
        }
      ]
    }
  }
]
```

SSMParameterWithSlashPrefixReadPolicy

Fornisce l'autorizzazione ad accedere a un parametro da un archivio di parametri Amazon EC2 Systems Manager (SSM) per caricare i segreti in questo account. Da utilizzare quando il nome del parametro ha un prefisso slash.

> **Note**
>
> Se non si utilizza la chiave predefinita, sarà necessaria anche la `KMSDecryptPolicy` politica.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "ssm:DescribeParameters"
    ],
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": [
      "ssm:GetParameters",
      "ssm:GetParameter",
      "ssm:GetParametersByPath"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:ssm:${AWS::Region}:${AWS::AccountId}:parameter
${parameterName}",
        {
          "parameterName": {
            "Ref": "ParameterName"
          }
```

```
        }
      ]
    }
  }
]
```

StepFunctionsExecutionPolicy

Fornisce il permesso di avviare l'esecuzione di una macchina a stati Step Functions.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "states:StartExecution"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:states:${AWS::Region}:${AWS::AccountId}:stateMachine:
${stateMachineName}",
        {
          "stateMachineName": {
            "Ref": "StateMachineName"
          }
        }
      ]
    }
  }
]
```

TextractDetectAnalyzePolicy

Fornisce l'accesso per rilevare e analizzare documenti con Amazon Textract.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "textract:DetectDocumentText",
      "textract:StartDocumentTextDetection",
      "textract:StartDocumentAnalysis",
      "textract:AnalyzeDocument"
    ],
```

```
    "Resource": "*"
  }
]
```

TextractGetResultPolicy

Fornisce l'accesso per ottenere documenti rilevati e analizzati da Amazon Textract.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "textract:GetDocumentTextDetection",
      "textract:GetDocumentAnalysis"
    ],
    "Resource": "*"
  }
]
```

TextractPolicy

Fornisce accesso completo ad Amazon Textract.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "textract:*"
    ],
    "Resource": "*"
  }
]
```

VPCAccessPolicy

Fornisce l'accesso per creare, eliminare, descrivere e scollegare interfacce di rete elastiche.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "ec2:CreateNetworkInterface",
      "ec2:DeleteNetworkInterface",
```

```
      "ec2:DescribeNetworkInterfaces",
      "ec2:DetachNetworkInterface"
    ],
    "Resource": "*"
  }
]
```

Gestione delle AWS SAM autorizzazioni con meccanismi CloudFormation

Per controllare l'accesso alle AWS risorse, il AWS Serverless Application Model (AWS SAM) può utilizzare gli stessi CloudFormation meccanismi di. Per ulteriori informazioni, consulta Controllo dell'accesso con AWS Identity and Access Management nella Guida per l'utente di AWS CloudFormation .

Esistono tre opzioni principali per concedere all'utente l'autorizzazione a gestire applicazioni serverless. Ogni opzione offre agli utenti diversi livelli di controllo degli accessi.

- Concedi le autorizzazioni di amministratore.
- Allega le politiche AWS gestite necessarie.
- Concedi autorizzazioni specifiche AWS Identity and Access Management (IAM).

A seconda dell'opzione scelta, gli utenti possono gestire solo applicazioni serverless contenenti AWS risorse a cui hanno l'autorizzazione di accesso.

Le sezioni seguenti descrivono ciascuna opzione in modo più dettagliato.

Concedi le autorizzazioni di amministratore

Se concedi le autorizzazioni di amministratore a un utente, quest'ultimo può gestire applicazioni serverless che contengono qualsiasi combinazione di risorse. AWS Questa è l'opzione più semplice, ma concede anche agli utenti il set di autorizzazioni più ampio, che quindi consente loro di eseguire azioni con il massimo impatto.

Per ulteriori informazioni sulla concessione delle autorizzazioni di amministratore a un utente, consulta Creazione del primo utente e gruppo di amministratori IAM nella Guida per l'utente IAM.

Allega le politiche gestite necessarie AWS

È possibile concedere agli utenti un sottoinsieme di autorizzazioni utilizzando policy AWS gestite, anziché concedere autorizzazioni amministrative complete. Se utilizzi questa opzione, assicurati che

il set di policy AWS gestite copra tutte le azioni e le risorse necessarie per le applicazioni serverless gestite dagli utenti.

Ad esempio, le seguenti politiche AWS gestite sono sufficienti per distribuire l'applicazione Hello World di esempio:

- AWSCloudFormationFullAccess
- IAMFullAccesso
- AWSLambda_FullAccess
- APIGatewayAmministratore Amazon
- Amazon S3 FullAccess
- Amazon EC2 ContainerRegistryFullAccess

Per informazioni sull'associazione delle policy a un utente IAM, consulta Modifica delle autorizzazioni per un utente IAM nella Guida per l'utente IAM.

Concedi autorizzazioni IAM specifiche

Per il livello più granulare di controllo degli accessi, puoi concedere autorizzazioni IAM specifiche agli utenti utilizzando dichiarazioni politiche. Se utilizzi questa opzione, assicurati che l'informativa sulla politica includa tutte le azioni e le risorse necessarie per le applicazioni serverless gestite dagli utenti.

La migliore pratica con questa opzione consiste nel negare agli utenti l'autorizzazione a creare ruoli, inclusi i ruoli di esecuzione Lambda, in modo che non possano concedersi autorizzazioni avanzate. Pertanto, l'amministratore deve prima creare un ruolo di esecuzione Lambda che verrà specificato nelle applicazioni serverless gestite dagli utenti. Per informazioni sulla creazione di ruoli di esecuzione Lambda, consulta Creazione di un ruolo di esecuzione nella console IAM.

Per l'applicazione Hello World di esempio AWSLambdaBasicExecutionRoleè sufficiente eseguire l'applicazione. Dopo aver creato un ruolo di esecuzione Lambda, modifica il file AWS SAM modello dell'applicazione Hello World di esempio per aggiungere la seguente proprietà alla `AWS::Serverless::Function` risorsa:

```
Role: lambda-execution-role-arn
```

Con l'applicazione Hello World modificata, la seguente dichiarazione politica concede agli utenti autorizzazioni sufficienti per distribuire, aggiornare ed eliminare l'applicazione:

> **Note**
>
> L'informativa di esempio contenuta in questa sezione concede autorizzazioni sufficienti per distribuire, aggiornare ed eliminare l'applicazione Hello World di esempio. Se aggiungi altri tipi di risorse all'applicazione, devi aggiornare l'informativa sulla politica per includere quanto segue:
>
> 1. Autorizzazione per l'applicazione a richiamare le azioni del servizio.
> 2. Il responsabile del servizio, se necessario per le azioni del servizio.
>
> Ad esempio, se si aggiunge un flusso di lavoro Step Functions, potrebbe essere necessario aggiungere le autorizzazioni per le azioni elencate qui e il responsabile del `states.amazonaws.com` servizio.

Per ulteriori informazioni sulle policy IAM, consulta Managing IAM policies nella IAM User Guide.

Controlla l'accesso alle API con il tuo AWS SAM modello

Il controllo dell'accesso al tuo API Gateway APIs aiuta a garantire che l'applicazione serverless sia sicura e accessibile solo tramite l'autorizzazione abilitata. Puoi abilitare l'autorizzazione nel tuo AWS SAM modello per controllare chi può accedere al tuo API Gateway APIs.

AWS SAM supporta diversi meccanismi per controllare l'accesso al tuo API Gateway APIs. L'insieme di meccanismi supportati varia a seconda dei tipi `AWS::Serverless::HttpApi` di `AWS::Serverless::Api` risorse.

La tabella seguente riassume i meccanismi supportati da ciascun tipo di risorsa.

Meccanismi per il controllo dell'accesso	AWS::Serverless::HttpApi	AWS::Serverless::Api
Autorizzatori Lambda	✓	✓
Autorizzazioni IAM		✓
Pool di utenti Amazon Cognito	✓ *	✓
Chiavi API		✓

Meccanismi per il controllo dell'accesso	AWS::Serverless::HttpApi	AWS::Serverless::Api
Policy delle risorse		✓
OAuth Autorizzatori 2.0/JWT	✓	

* Puoi usare Amazon Cognito come emittente di JSON Web Token (JWT) con il tipo di risorsa. `AWS::Serverless::HttpApi`

- Autorizzatori Lambda: un autorizzatore Lambda (precedentemente noto come autorizzatore personalizzato) è una funzione Lambda fornita per controllare l'accesso all'API. Quando viene chiamata l'API, questa funzione Lambda viene richiamata con un contesto di richiesta o un token di autorizzazione fornito dall'applicazione client. La funzione Lambda risponde se il chiamante è autorizzato a eseguire l'operazione richiesta.

  Entrambi i tipi `AWS::Serverless::HttpApi` di `AWS::Serverless::Api` risorse supportano gli autorizzatori Lambda.

  Per ulteriori informazioni sugli autorizzatori Lambda con`AWS::Serverless::HttpApi`, consulta Working with AWS Lambda authorizers for HTTP nella APIs API Gateway Developer Guide. Per ulteriori informazioni sugli autorizzatori Lambda con*`AWS::Serverless::Api`*, consulta Use API Gateway Lambda authorizers nella API Gateway Developer Guide.

  Per esempi di autorizzatori Lambda per entrambi i tipi di risorse, vedere. Esempi di autorizzazioni Lambda per AWS SAM

- Autorizzazioni IAM: puoi controllare chi può richiamare la tua API utilizzando le autorizzazioni AWS Identity and Access Management (IAM). Gli utenti che chiamano la tua API devono essere autenticati con credenziali IAM. Le chiamate alla tua API hanno esito positivo solo se all'utente IAM è associata una policy IAM che rappresenta il chiamante dell'API, un gruppo IAM che contiene l'utente o un ruolo IAM assunto dall'utente.

  Solo il tipo di `AWS::Serverless::Api` risorsa supporta le autorizzazioni IAM.

  Per ulteriori informazioni, consulta Controllare l'accesso a un'API con autorizzazioni IAM nella Guida per sviluppatori di API Gateway. Per vedere un esempio, consulta Esempio di autorizzazione IAM per AWS SAM.

- Pool di utenti Amazon Cognito: i pool di utenti di Amazon Cognito sono directory di utenti in Amazon Cognito. Un client della tua API deve prima accedere a un utente al pool di utenti e ottenere un'identità o un token di accesso per l'utente. Quindi il client chiama l'API con uno dei token restituiti. La chiamata API ha esito positivo solo se il token richiesto è valido.

  Il tipo di `AWS::Serverless::Api` risorsa supporta i pool di utenti di Amazon Cognito. Il tipo di `AWS::Serverless::HttpApi` risorsa supporta l'uso di Amazon Cognito come emittente JWT.

  Per ulteriori informazioni, consulta la sezione Control access to a REST API using Amazon Cognito user pools as authorizer (Controllo degli accessi a un'API REST utilizzando pool di utenti di Amazon Cognito come autorizzazione) nella Guida per sviluppatori di API Gateway. Per vedere un esempio, consulta Esempio di pool di utenti Amazon Cognito per AWS SAM.
- Chiavi API: le chiavi API sono valori di stringhe alfanumeriche che distribuisci ai clienti sviluppatori di applicazioni per concedere l'accesso alla tua API.

  Solo il tipo di `AWS::Serverless::Api` risorsa supporta le chiavi API.

  Per ulteriori informazioni sulle chiavi API, consulta Creazione e utilizzo di piani di utilizzo con chiavi API nella Guida per sviluppatori di API Gateway. Per un esempio di chiavi API, consultaEsempio di chiave API per AWS SAM.
- Politiche delle risorse: le politiche delle risorse sono documenti di policy JSON che è possibile allegare a un'API API Gateway. Utilizza le policy relative alle risorse per controllare se un principale specificato (in genere un utente o un ruolo IAM) può richiamare l'API.

  Solo il tipo di `AWS::Serverless::Api` risorsa supporta le policy delle risorse come meccanismo per controllare l'accesso all'API Gateway APIs.

  Per ulteriori informazioni sulle politiche delle risorse, consulta Controllare l'accesso a un'API con le politiche delle risorse di API Gateway nella Guida per sviluppatori di API Gateway. Per un esempio di politiche relative alle risorse, consultaEsempio di policy in materia di risorse per AWS SAM.
- OAuth Autorizzatori 2.0/JWT: puoi utilizzare JWTs come parte dei framework OpenID Connect (OIDC) e OAuth 2.0 per controllare l'accesso al tuo. APIs API Gateway convalida JWTs le richieste API inviate dai client e consente o rifiuta le richieste in base alla convalida del token e, facoltativamente, agli ambiti nel token.

  Solo il tipo di `AWS::Serverless::HttpApi` risorsa supporta gli autorizzatori 2.0/JWT. OAuth

Per ulteriori informazioni, consulta Controllare l'accesso a HTTP APIs con gli autorizzatori JWT nella API Gateway Developer Guide. Per vedere un esempio, consulta OAuth Esempio di autorizzazione 2.0/JWT per AWS SAM.

Scelta di un meccanismo per controllare l'accesso

Il meccanismo che scegli di utilizzare per controllare l'accesso al tuo API Gateway APIs dipende da alcuni fattori. Ad esempio, se hai un progetto greenfield senza autorizzazione o configurazione del controllo degli accessi, i pool di utenti di Amazon Cognito potrebbero essere l'opzione migliore. Questo perché quando configuri i pool di utenti, configuri automaticamente anche l'autenticazione e il controllo degli accessi.

Tuttavia, se l'applicazione ha già configurato l'autenticazione, l'utilizzo degli autorizzatori Lambda potrebbe essere l'opzione migliore. Questo perché puoi chiamare il servizio di autenticazione esistente e restituire un documento di policy basato sulla risposta. Inoltre, se la tua applicazione richiede un'autenticazione personalizzata o una logica di controllo degli accessi che i pool di utenti non supportano, gli autorizzatori Lambda potrebbero essere l'opzione migliore.

Dopo aver scelto il meccanismo da utilizzare, consulta la sezione corrispondente Esempi per scoprire come configurare l'applicazione AWS SAM per utilizzare quel meccanismo.

Personalizzazione delle risposte agli errori

È possibile utilizzarlo AWS SAM per personalizzare il contenuto di alcune risposte di errore di API Gateway. Solo il tipo di `AWS::Serverless::Api` risorsa supporta risposte API Gateway personalizzate.

Per ulteriori informazioni sulle risposte di API Gateway, consulta le risposte del gateway in API Gateway nella API Gateway Developer Guide. Per un esempio di risposte personalizzate, consultaEsempio di risposta personalizzata per AWS SAM.

Esempi

- Esempi di autorizzazioni Lambda per AWS SAM
- Esempio di autorizzazione IAM per AWS SAM
- Esempio di pool di utenti Amazon Cognito per AWS SAM
- Esempio di chiave API per AWS SAM

- Esempio di policy in materia di risorse per AWS SAM
- OAuth Esempio di autorizzazione 2.0/JWT per AWS SAM
- Esempio di risposta personalizzata per AWS SAM

Esempi di autorizzazioni Lambda per AWS SAM

Il tipo di `AWS::Serverless::Api` risorsa supporta due tipi di autorizzatori Lambda: autorizzatori e `TOKEN` autorizzatori. `REQUEST` Il tipo di `AWS::Serverless::HttpApi` risorsa supporta solo gli autorizzatori. `REQUEST` Di seguito sono riportati alcuni esempi di ogni tipo.

Esempio di **`TOKEN`** autorizzazione Lambda () AWS::Serverless::Api

Puoi controllare l'accesso al tuo APIs definendo un `TOKEN` autorizzatore Lambda all'interno del tuo modello. AWS SAM A tale scopo, si utilizza il tipo di ApiAuth dati.

Di seguito è riportato un esempio di sezione AWS SAM modello per un autorizzatore Lambda: `TOKEN`

> **Note**
>
> Nell'esempio seguente, il SAM `FunctionRole` viene generato implicitamente.

```
Resources:
  MyApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: Prod
      Auth:
        DefaultAuthorizer: MyLambdaTokenAuthorizer
        Authorizers:
          MyLambdaTokenAuthorizer:
            FunctionArn: !GetAtt MyAuthFunction.Arn

  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: ./src
      Handler: index.handler
      Runtime: nodejs12.x
      Events:
```

```
        GetRoot:
          Type: Api
          Properties:
            RestApiId: !Ref MyApi
            Path: /
            Method: get

  MyAuthFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: ./src
      Handler: authorizer.handler
      Runtime: nodejs12.x
```

Per ulteriori informazioni sugli autorizzatori Lambda, consulta Use API Gateway Lambda authorizers nella API Gateway Developer Guide.

Esempio di **`REQUEST`** autorizzazione Lambda () AWS::Serverless::Api

Puoi controllare l'accesso al tuo APIs definendo un `REQUEST` autorizzatore Lambda all'interno del tuo modello. AWS SAM A tale scopo, si utilizza il tipo di ApiAuth dati.

Di seguito è riportato un esempio di sezione AWS SAM modello per un autorizzatore Lambda: `REQUEST`

```
Resources:
  MyApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: Prod
      Auth:
        DefaultAuthorizer: MyLambdaRequestAuthorizer
        Authorizers:
          MyLambdaRequestAuthorizer:
            FunctionPayloadType: REQUEST
            FunctionArn: !GetAtt MyAuthFunction.Arn
            Identity:
              QueryStrings:
                - auth

  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
```

```
      CodeUri: ./src
      Handler: index.handler
      Runtime: nodejs12.x
      Events:
        GetRoot:
          Type: Api
          Properties:
            RestApiId: !Ref MyApi
            Path: /
            Method: get

  MyAuthFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: ./src
      Handler: authorizer.handler
      Runtime: nodejs12.x
```

Per ulteriori informazioni sugli autorizzatori Lambda, consulta Use API Gateway Lambda authorizers nella API Gateway Developer Guide.

Esempio di autorizzazione Lambda () AWS::Serverless::HttpApi

Puoi controllare l'accesso al tuo HTTP APIs definendo un autorizzatore Lambda all'interno del tuo modello. AWS SAM A tale scopo, si utilizza il tipo di HttpApiAuth dati.

Di seguito è riportato un esempio di sezione AWS SAM modello per un autorizzatore Lambda:

```
Resources:
  MyApi:
    Type: AWS::Serverless::HttpApi
    Properties:
      StageName: Prod
      Auth:
        DefaultAuthorizer: MyLambdaRequestAuthorizer
        Authorizers:
          MyLambdaRequestAuthorizer:
            FunctionArn: !GetAtt MyAuthFunction.Arn
            FunctionInvokeRole: !GetAtt MyAuthFunctionRole.Arn
            Identity:
              Headers:
                - Authorization
            AuthorizerPayloadFormatVersion: 2.0
```

```
            EnableSimpleResponses: true

  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: ./src
      Handler: index.handler
      Runtime: nodejs12.x
      Events:
        GetRoot:
          Type: HttpApi
          Properties:
            ApiId: !Ref MyApi
            Path: /
            Method: get
            PayloadFormatVersion: "2.0"

  MyAuthFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: ./src
      Handler: authorizer.handler
      Runtime: nodejs12.x
```

Esempio di autorizzazione IAM per AWS SAM

Puoi controllare l'accesso al tuo APIs definendo le autorizzazioni IAM all'interno del tuo AWS SAM modello. A tale scopo, si utilizza il tipo di ApiAuth dati.

Di seguito è riportato un AWS SAM modello di esempio che utilizza per le autorizzazioni IAM:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Resources:
  MyApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: Prod
      Description: 'API with IAM authorization'
      Auth:
        DefaultAuthorizer: AWS_IAM #sets AWS_IAM auth for all methods in this API
  MyFunction:
    Type: AWS::Serverless::Function
```

```
    Properties:
      Handler: index.handler
      Runtime: python3.10
      Events:
        GetRoot:
          Type: Api
          Properties:
            RestApiId: !Ref MyApi
            Path: /
            Method: get
      InlineCode: |
        def handler(event, context):
          return {'body': 'Hello World!', 'statusCode': 200}
```

Per ulteriori informazioni sulle autorizzazioni IAM, consulta Control access for invoking an API nella API Gateway Developer Guide.

Esempio di pool di utenti Amazon Cognito per AWS SAM

Puoi controllare l'accesso ai tuoi utenti APIs definendo i pool di utenti di Amazon Cognito all'interno del AWS SAM modello. Per fare ciò, usi il tipo di ApiAuth dati.

Di seguito è riportato un esempio AWS SAM di sezione modello per un pool di utenti:

```
Resources:
  MyApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: Prod
      Cors: "'*'"
      Auth:
        DefaultAuthorizer: MyCognitoAuthorizer
        Authorizers:
          MyCognitoAuthorizer:
            UserPoolArn: !GetAtt MyCognitoUserPool.Arn

  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: ./src
      Handler: lambda.handler
      Runtime: nodejs12.x
      Events:
```

```
        Root:
          Type: Api
          Properties:
            RestApiId: !Ref MyApi
            Path: /
            Method: GET

  MyCognitoUserPool:
    Type: AWS::Cognito::UserPool
    Properties:
      UserPoolName: !Ref CognitoUserPoolName
      Policies:
        PasswordPolicy:
          MinimumLength: 8
      UsernameAttributes:
        - email
      Schema:
        - AttributeDataType: String
          Name: email
          Required: false

  MyCognitoUserPoolClient:
    Type: AWS::Cognito::UserPoolClient
    Properties:
      UserPoolId: !Ref MyCognitoUserPool
      ClientName: !Ref CognitoUserPoolClientName
      GenerateSecret: false
```

Per ulteriori informazioni sui pool di utenti di Amazon Cognito, consulta Controllare l'accesso a un'API REST utilizzando i pool di utenti di Amazon Cognito come autorizzatore nella Guida per sviluppatori di API Gateway.

Esempio di chiave API per AWS SAM

Puoi controllare l'accesso al tuo APIs richiedendo le chiavi API all'interno del tuo AWS SAM modello. A tale scopo, si utilizza il tipo di ApiAuth dati.

Di seguito è riportato un esempio di sezione AWS SAM modello per le chiavi API:

```
Resources:
  MyApi:
    Type: AWS::Serverless::Api
    Properties:
```

```
      StageName: Prod
      Auth:
        ApiKeyRequired: true # sets for all methods

  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: .
      Handler: index.handler
      Runtime: nodejs12.x
      Events:
        ApiKey:
          Type: Api
          Properties:
            RestApiId: !Ref MyApi
            Path: /
            Method: get
            Auth:
              ApiKeyRequired: true
```

Per ulteriori informazioni sulle chiavi API, consulta Creazione e utilizzo di piani di utilizzo con chiavi API nella Guida per sviluppatori di API Gateway.

Esempio di policy in materia di risorse per AWS SAM

Puoi controllare l'accesso ai tuoi APIs allegando una politica delle risorse all'interno del tuo AWS SAM modello. A tale scopo, si utilizza il tipo di ApiAuth dati.

Di seguito è riportato un AWS SAM modello di esempio per un'API privata. Un'API privata deve disporre di una politica delle risorse da distribuire.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Resources:
  MyPrivateApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: Prod
      EndpointConfiguration: PRIVATE  # Creates a private API. Resource policies are
 required for all private APIs.
      Auth:
        ResourcePolicy:
          CustomStatements:
```

```
            - Effect: 'Allow'
              Action: 'execute-api:Invoke'
              Resource: ['execute-api:/*/*/*']
              Principal: '*'
            - Effect: 'Deny'
              Action: 'execute-api:Invoke'
              Resource: ['execute-api:/*/*/*']
              Principal: '*'
  MyFunction:
    Type: 'AWS::Serverless::Function'
    Properties:
      InlineCode: |
        def handler(event, context):
          return {'body': 'Hello World!', 'statusCode': 200}
      Handler: index.handler
      Runtime: python3.10
      Events:
        AddItem:
          Type: Api
          Properties:
            RestApiId:
              Ref: MyPrivateApi
            Path: /
            Method: get
```

Per ulteriori informazioni sulle politiche delle risorse, consulta Controllare l'accesso a un'API con le politiche delle risorse di API Gateway nella Guida per sviluppatori di API Gateway. Per ulteriori informazioni sulla privacy APIs, consulta la sezione Creazione di un'API privata in Amazon API Gateway nella API Gateway Developer Guide.

OAuth Esempio di autorizzazione 2.0/JWT per AWS SAM

Puoi controllare l'accesso al tuo APIs utilizzo JWTs come parte dei framework OpenID Connect (OIDC) e 2.0. OAuth A tale scopo, si utilizza il tipo di dati. HttpApiAuth

Di seguito è riportato un esempio di sezione AWS SAM modello per un autorizzatore OAuth 2.0/JWT:

```
Resources:
  MyApi:
    Type: AWS::Serverless::HttpApi
    Properties:
      Auth:
```

```
        Authorizers:
          MyOauth2Authorizer:
            AuthorizationScopes:
              - scope
            IdentitySource: $request.header.Authorization
            JwtConfiguration:
              audience:
                - audience1
                - audience2
              issuer: "https://www.example.com/v1/connect/oidc"
        DefaultAuthorizer: MyOauth2Authorizer
      StageName: Prod
  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: ./src
      Events:
        GetRoot:
          Properties:
            ApiId: MyApi
            Method: get
            Path: /
            PayloadFormatVersion: "2.0"
          Type: HttpApi
      Handler: index.handler
      Runtime: nodejs12.x
```

Per ulteriori informazioni sugli autorizzatori OAuth 2.0/JWT, consulta Controlling access to HTTP APIs with JWT authorizers nella API Gateway Developer Guide.

Esempio di risposta personalizzata per AWS SAM

Puoi personalizzare alcune risposte di errore di API Gateway definendo le intestazioni di risposta all'interno del tuo AWS SAM modello. A tale scopo, si utilizza il tipo di dati Gateway Response Object.

Di seguito è riportato un AWS SAM modello di esempio che crea una risposta personalizzata per l'`DEFAULT_5XX`errore.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Resources:
  MyApi:
    Type: AWS::Serverless::Api
```

```
    Properties:
      StageName: Prod
      GatewayResponses:
        DEFAULT_5XX:
          ResponseParameters:
            Headers:
              Access-Control-Expose-Headers: "'WWW-Authenticate'"
              Access-Control-Allow-Origin: "'*'"
              ErrorHeader: "'MyCustomErrorHeader'"
          ResponseTemplates:
            application/json: "{\"message\": \"Error on the $context.resourcePath
resource\" }"

  GetFunction:
    Type: AWS::Serverless::Function
    Properties:
      Runtime: python3.10
      Handler: index.handler
      InlineCode: |
        def handler(event, context):
          raise Exception('Check out the new response!')
      Events:
        GetResource:
          Type: Api
          Properties:
            Path: /error
            Method: get
            RestApiId: !Ref MyApi
```

Per ulteriori informazioni sulle risposte di API Gateway, consulta le risposte del gateway in API Gateway nella API Gateway Developer Guide.

Aumenta l'efficienza utilizzando i livelli Lambda con AWS SAM

Utilizzando AWS SAM, è possibile includere livelli nelle applicazioni serverless. AWS Lambda i livelli consentono di estrarre il codice da una funzione Lambda in un livello Lambda che può quindi essere utilizzato in diverse funzioni Lambda. In questo modo è possibile ridurre le dimensioni dei pacchetti di distribuzione, separare la logica delle funzioni principali dalle dipendenze e condividere le dipendenze tra più funzioni. Per ulteriori informazioni sui layer, consulta i layer Lambda nella AWS Lambda Developer Guide.

Questo argomento fornisce informazioni su quanto segue:

- Inclusione di livelli nell'applicazione
- In che modo i livelli vengono memorizzati nella cache locale

Per informazioni sulla creazione di livelli personalizzati, consultaCreazione di livelli Lambda in AWS SAM.

Inclusione di livelli nell'applicazione

Per includere livelli nell'applicazione, utilizzate la `Layers` proprietà del tipo di AWS::Serverless::Function risorsa.

Di seguito è riportato un AWS SAM modello di esempio con una funzione Lambda che include un livello:

```
ServerlessFunction:
  Type: AWS::Serverless::Function
  Properties:
    CodeUri: .
    Handler: my_handler
    Runtime: Python3.7
    Layers:
        - <LayerVersion ARN>
```

Come i livelli vengono memorizzati nella cache locale

Quando si richiama la funzione utilizzando uno dei `sam local` comandi, il pacchetto layers della funzione viene scaricato e memorizzato nella cache sull'host locale.

La tabella seguente mostra le posizioni delle directory di cache predefinite per diversi sistemi operativi.

Sistema operativo	Location (Ubicazione)
Windows 7	`C:\Users\<user>\AppData\Roaming\AWS SAM`
Windows 8	`C:\Users\<user>\AppData\Roaming\AWS SAM`
Windows 10	`C:\Users\<user>\AppData\Roaming\AWS SAM`
macOS	`~/.aws-sam/layers-pkg`

Sistema operativo	Location (Ubicazione)
Unix	`~/.aws-sam/layers-pkg`

Dopo che il pacchetto è stato memorizzato nella cache, AWS SAMCLI sovrappone i livelli su un'immagine Docker utilizzata per richiamare la funzione. AWS SAMCLIGenera i nomi delle immagini che crea, oltre a LayerVersions quelle conservate nella cache. Puoi trovare maggiori dettagli sullo schema nelle seguenti sezioni.

Per ispezionare i livelli sovrapposti, esegui il seguente comando per avviare una sessione bash nell'immagine che vuoi ispezionare:

```
docker run -it --entrypoint=/bin/bash samcli/lambda:<Tag following the schema outlined
 in Docker Image Tag Schema> -i
```

Schema dei nomi della Layer Caching Directory

LayerVersionArn Dato un valore definito nel modello, AWS SAMCLI estrae la versione LayerName and dall'ARN. Crea una directory in cui inserire il contenuto del livello in named. `LayerName-Version-<first 10 characters of sha256 of ARN>`

Esempio:

```
ARN = arn:aws:lambda:us-west-2:111111111111:layer:myLayer:1
Directory name = myLayer-1-926eeb5ff1
```

Schema di tag Docker Images

Per calcolare l'hash univoco dei livelli, combina tutti i nomi univoci dei livelli con un delimitatore «-», prendi l' SHA256 hash e poi prendi i primi 10 caratteri.

Esempio:

```
ServerlessFunction:
  Type: AWS::Serverless::Function
  Properties:
    CodeUri: .
    Handler: my_handler
```

```
    Runtime: Python3.7
    Layers:
        - arn:aws:lambda:us-west-2:111111111111:layer:myLayer:1
        - arn:aws:lambda:us-west-2:111111111111:layer:mySecondLayer:1
```

I nomi univoci vengono calcolati nello stesso modo dello schema dei nomi di Layer Caching Directory:

```
arn:aws:lambda:us-west-2:111111111111:layer:myLayer:1 = myLayer-1-926eeb5ff1
arn:aws:lambda:us-west-2:111111111111:layer:mySecondLayer:1 =
 mySecondLayer-1-6bc1022bdf
```

Per calcolare l'hash univoco dei livelli, combina tutti i nomi univoci dei livelli con un delimitatore '-', prendi l'hash sha256 e poi prendi i primi 25 caratteri:

```
myLayer-1-926eeb5ff1-mySecondLayer-1-6bc1022bdf = 2dd7ac5ffb30d515926aef
```

Quindi combina questo valore con il runtime e l'architettura della funzione, con un delimitatore di '-':

```
python3.7-x86_64-2dd7ac5ffb30d515926aefffd
```

Riutilizza codice e risorse utilizzando applicazioni annidate in AWS SAM

Un'applicazione serverless può includere una o più applicazioni annidate. Un'applicazione nidificata fa parte di un'applicazione più ampia e può essere impacchettata e distribuita come elemento autonomo o come componente di un'applicazione più grande. Le applicazioni annidate consentono di trasformare il codice utilizzato di frequente in un'applicazione propria, che può quindi essere riutilizzata su un'applicazione serverless più grande o su più applicazioni serverless.

Man mano che le architetture serverless crescono, in genere emergono modelli comuni in cui gli stessi componenti sono definiti in più modelli di applicazione. Le applicazioni annidate consentono di riutilizzare codice, funzionalità, risorse e configurazioni comuni in AWS SAM modelli separati, consentendovi di gestire solo il codice proveniente da un'unica fonte. Ciò riduce il codice e le configurazioni duplicati. Inoltre, questo approccio modulare semplifica lo sviluppo, migliora l'organizzazione del codice e facilita la coerenza tra le applicazioni serverless. Con le applicazioni annidate, puoi concentrarti maggiormente sulla logica di business che è unica per la tua applicazione.

Per definire un'applicazione nidificata nella tua applicazione serverless, utilizza il AWS::Serverless::Application tipo di risorsa.

È possibile definire applicazioni nidificate dalle seguenti due fonti:

- Un'AWS Serverless Application Repository applicazione: è possibile definire applicazioni nidificate utilizzando le applicazioni disponibili per il proprio account in. AWS Serverless Application Repository Queste possono essere applicazioni private nel tuo account, applicazioni condivise privatamente con il tuo account o applicazioni condivise pubblicamente in. AWS Serverless Application RepositoryPer ulteriori informazioni sui diversi livelli di autorizzazioni di distribuzione, consulta Autorizzazioni di distribuzione delle applicazioni e applicazioni di pubblicazione nella Guida per gli AWS Serverless Application Repository sviluppatori.
- Un'applicazione locale: è possibile definire applicazioni annidate utilizzando applicazioni archiviate nel file system locale.

Consultate le sezioni seguenti per i dettagli su come AWS SAM definire entrambi questi tipi di applicazioni annidate nell'applicazione serverless.

> **Note**
>
> Il numero massimo di applicazioni che possono essere annidate in un'applicazione serverless è 200.
> Il numero massimo di parametri che un'applicazione annidata può avere è 60.

Definizione di un'applicazione nidificata da AWS Serverless Application Repository

È possibile definire applicazioni nidificate utilizzando applicazioni disponibili in. AWS Serverless Application RepositoryÈ inoltre possibile archiviare e distribuire applicazioni che contengono applicazioni nidificate utilizzando. AWS Serverless Application Repository Per esaminare i dettagli di un'applicazione annidata in AWS Serverless Application Repository, puoi utilizzare l' AWS SDK, la o la console AWS CLI Lambda.

Per definire un'applicazione ospitata nel AWS SAM modello dell'applicazione serverless, utilizza il pulsante Copia come risorsa SAM nella pagina dei dettagli di ogni applicazione. AWS Serverless Application Repository AWS Serverless Application Repository A tale scopo, seguire queste fasi:

1. Assicurati di aver effettuato l'accesso a. Console di gestione AWS
2. Trova l'applicazione in cui desideri annidare AWS Serverless Application Repository utilizzando i passaggi indicati nella sezione Navigazione, ricerca e distribuzione delle applicazioni della Guida per gli AWS Serverless Application Repository sviluppatori.

3. Scegli il pulsante Copia come risorsa SAM. La sezione dei modelli SAM per l'applicazione che stai visualizzando è ora negli appunti.
4. Incolla la sezione del modello SAM nella `Resources:` sezione del file modello SAM per l'applicazione che desideri annidare in questa applicazione.

Di seguito è riportato un esempio di sezione modello SAM per un'applicazione nidificata ospitata in: AWS Serverless Application Repository

```
Transform: AWS::Serverless-2016-10-31

Resources:
  applicationaliasname:
    Type: AWS::Serverless::Application
    Properties:
      Location:
        ApplicationId: arn:aws:serverlessrepo:us-
east-1:123456789012:applications/application-alias-name
        SemanticVersion: 1.0.0
      Parameters:
        # Optional parameter that can have default value overridden
        # ParameterName1: 15 # Uncomment to override default value
        # Required parameter that needs value to be provided
        ParameterName2: YOUR_VALUE
```

Se non sono presenti impostazioni dei parametri obbligatorie, è possibile omettere la `Parameters:` sezione del modello.

> **Important**
>
> Le applicazioni che contengono applicazioni nidificate ospitate in AWS Serverless Application Repository ereditano le restrizioni di condivisione delle applicazioni nidificate.
> Ad esempio, supponiamo che un'applicazione sia condivisa pubblicamente, ma contenga un'applicazione nidificata condivisa privatamente solo con l'account che ha creato l'applicazione principale. AWS In questo caso, se il tuo AWS account non dispone dell'autorizzazione per distribuire l'applicazione nidificata, non puoi distribuire l'applicazione principale. Per ulteriori informazioni sulle autorizzazioni per la distribuzione delle applicazioni, consulta Application Deployment Permissions and Publishing Applications nella Developer Guide.AWS Serverless Application Repository

Definizione di un'applicazione annidata dal file system locale

È possibile definire applicazioni annidate utilizzando applicazioni archiviate nel file system locale. A tale scopo, è necessario specificare il percorso del file AWS SAM modello archiviato nel file system locale.

Di seguito è riportato un esempio di sezione modello SAM per un'applicazione locale annidata:

```
Transform: AWS::Serverless-2016-10-31

Resources:
  applicationaliasname:
    Type: AWS::Serverless::Application
    Properties:
      Location: ../my-other-app/template.yaml
      Parameters:
        # Optional parameter that can have default value overridden
        # ParameterName1: 15 # Uncomment to override default value
        # Required parameter that needs value to be provided
        ParameterName2: YOUR_VALUE
```

Se non sono presenti impostazioni dei parametri, è possibile omettere la `Parameters:` sezione del modello.

Distribuzione di applicazioni annidate

È possibile distribuire l'applicazione annidata utilizzando il comando. AWS SAMCLI `sam deploy` Per ulteriori dettagli, consultare Implementa la tua applicazione e le tue risorse con AWS SAM.

> **Note**
>
> Quando si distribuisce un'applicazione che contiene applicazioni nidificate, è necessario confermare che contiene applicazioni nidificate. Puoi farlo passando `CAPABILITY_AUTO_EXPAND` all'CreateCloudFormationChangeSet API o utilizzando il comando. `aws serverlessrepo create-cloud-formation-change-set` AWS CLI Per ulteriori informazioni sul riconoscimento delle applicazioni annidate, consulta la sezione Riconoscimento dei ruoli IAM, delle politiche delle risorse e delle applicazioni annidate durante la distribuzione delle applicazioni nella Guida per gli sviluppatori.AWS Serverless Application Repository

Gestisci eventi basati sul tempo con EventBridge Scheduler in AWS SAM

Il contenuto di questo argomento fornisce dettagli su cos'è Amazon EventBridge Scheduler, quali AWS SAM offerte di supporto, come creare eventi Scheduler ed esempi a cui puoi fare riferimento quando crei eventi Scheduler.

Cos'è Amazon EventBridge Scheduler?

Usa EventBridge Scheduler per pianificare eventi nei tuoi AWS SAM modelli. Amazon EventBridge Scheduler è un servizio di pianificazione che consente di creare, avviare e gestire decine di milioni di eventi e attività in tutti i servizi. AWS Questo servizio è particolarmente utile per gli eventi legati al tempo. È possibile utilizzarlo per pianificare eventi e invocazioni ricorrenti basate sul tempo. Supporta anche eventi occasionali, nonché espressioni rate e chron con un'ora di inizio e di fine.

Per ulteriori informazioni su Amazon EventBridge Scheduler, consulta What is Amazon EventBridge Scheduler? nella Guida per l'utente di EventBridge Scheduler.

Argomenti

- EventBridge Supporto Scheduler in AWS SAM
- Creazione di eventi Scheduler EventBridge in AWS SAM
- Esempi
- Ulteriori informazioni

EventBridge Supporto Scheduler in AWS SAM

La specifica del modello AWS Serverless Application Model (AWS SAM) fornisce una sintassi semplice e abbreviata che è possibile utilizzare per pianificare eventi con EventBridge Scheduler for and. AWS Lambda AWS Step Functions

Creazione di eventi Scheduler EventBridge in AWS SAM

Imposta la `ScheduleV2` proprietà come tipo di evento nel AWS SAM modello per definire l'evento EventBridge Scheduler. Questa proprietà supporta i tipi di `AWS::Serverless::StateMachine` risorse `AWS::Serverless::Function` and.

```
MyFunction:
  Type: AWS::Serverless::Function
  Properties:
    Events:
```

```
      CWSchedule:
        Type: ScheduleV2
        Properties:
          ScheduleExpression: 'rate(1 minute)'
          Name: TestScheduleV2Function
          Description: Test schedule event

MyStateMachine:
  Type: AWS::Serverless::StateMachine
  Properties:
    Events:
      CWSchedule:
        Type: ScheduleV2
        Properties:
          ScheduleExpression: 'rate(1 minute)'
          Name: TestScheduleV2StateMachine
          Description: Test schedule event
```

EventBridge La pianificazione degli eventi di Scheduler supporta anche le code di lettere morte (DLQ) per gli eventi non elaborati. Per ulteriori informazioni sulle code di lettere non scritte, vedere Configurazione di una coda di lettere non scritte per Scheduler nella Guida per l'utente di Scheduler. EventBridge EventBridge

Quando viene specificato un ARN DLQ AWS SAM , configura le autorizzazioni per la pianificazione dello Scheduler per inviare messaggi al DLQ. Quando non viene specificato un ARN DLQ AWS SAM , creerà la risorsa DLQ.

Esempi

Esempio di base di definizione di un EventBridge evento Scheduler con AWS SAM

```
Transform: AWS::Serverless-2016-10-31
Resources:
  MyLambdaFunction:
    Type: AWS::Serverless::Function
    Properties:
      Handler: index.handler
      Runtime: python3.8
      InlineCode: |
        def handler(event, context):
            print(event)
            return {'body': 'Hello World!', 'statusCode': 200}
      MemorySize: 128
```

```
      Events:
        Schedule:
          Type: ScheduleV2
          Properties:
            ScheduleExpression: rate(1 minute)
            Input: '{"hello": "simple"}'

  MySFNFunction:
    Type: AWS::Serverless::Function
    Properties:
      Handler: index.handler
      Runtime: python3.8
      InlineCode: |
        def handler(event, context):
            print(event)
            return {'body': 'Hello World!', 'statusCode': 200}
      MemorySize: 128

  StateMachine:
    Type: AWS::Serverless::StateMachine
    Properties:
      Type: STANDARD
      Definition:
        StartAt: MyLambdaState
        States:
          MyLambdaState:
            Type: Task
            Resource: !GetAtt MySFNFunction.Arn
            End: true
      Policies:
        - LambdaInvokePolicy:
            FunctionName: !Ref MySFNFunction
      Events:
        Schedule:
          Type: ScheduleV2
          Properties:
            ScheduleExpression: rate(1 minute)
            Input: '{"hello": "simple"}'
```

Ulteriori informazioni

Per ulteriori informazioni sulla definizione della proprietà `ScheduleV2` EventBridge Scheduler, consulta:

- ScheduleV2per`AWS::Serverless::Function`.
- ScheduleV2per`AWS::Serverless::StateMachine`.

Orchestrazione delle risorse con AWS SAM AWS Step Functions

È possibile utilizzarlo AWS Step Functionsper orchestrare AWS Lambda funzioni e altre AWS risorse per formare flussi di lavoro complessi e robusti. Step Functions per indicare all'applicazione quando e in quali condizioni vengono utilizzate le AWS risorse, ad esempio AWS Lambda le funzioni. Ciò semplifica il processo di creazione di flussi di lavoro complessi e robusti. UtilizzandoAWS::Serverless::StateMachine, si definiscono le singole fasi del flusso di lavoro, si associano le risorse in ciascuna fase e quindi si mettono in sequenza questi passaggi. Puoi anche aggiungere transizioni e condizioni dove sono necessarie. Ciò semplifica il processo di creazione di un flusso di lavoro complesso e robusto.

> **Note**
>
> Per gestire i AWS SAM modelli che contengono le macchine a stati Step Functions, è necessario utilizzare la versione 0.52.0 o successiva di. AWS SAMCLI Per verificare la versione in uso, esegui il comando. `sam --version`

Step Functions si basa sui concetti di task e macchine a stati. Definisci le macchine a stati utilizzando Amazon States Language basato su JSON. La console Step Functions mostra una visualizzazione grafica della struttura della macchina a stati in modo da poter controllare visivamente la logica della macchina a stati e monitorare le esecuzioni.

Con il supporto Step Functions in AWS Serverless Application Model (AWS SAM), puoi fare quanto segue:

- Definite le macchine a stati, direttamente all'interno di un AWS SAM modello o in un file separato
- Crea ruoli di esecuzione delle macchine a stati tramite modelli di AWS SAM policy, policy in linea o policy gestite
- Attiva esecuzioni di macchine a stati con API Gateway o EventBridge eventi Amazon, in base a una pianificazione all'interno di un AWS SAM modello o chiamando direttamente APIs
- Utilizza i modelli di AWS SAM policy disponibili per i modelli di sviluppo Step Functions comuni.

Esempio

Il seguente frammento di esempio tratto da un file AWS SAM modello definisce una macchina a stati Step Functions in un file di definizione. Tieni presente che il `my_state_machine.asl.json` file deve essere scritto in Amazon States Language.

```
AWSTemplateFormatVersion: "2010-09-09"
Transform: AWS::Serverless-2016-10-31
Description: Sample SAM template with Step Functions State Machine

Resources:
  MyStateMachine:
    Type: AWS::Serverless::StateMachine
    Properties:
      DefinitionUri: statemachine/my_state_machine.asl.json
      ...
```

Per scaricare un' AWS SAM applicazione di esempio che include una macchina a stati Step Functions, consulta Creare una macchina a stati Step Functions Using AWS SAM nella AWS Step Functions Developer Guide.

Ulteriori informazioni

Per saperne di più su Step Functions e sul suo utilizzo con AWS SAM, consulta quanto segue:

- Funzionamento di AWS Step Functions
- AWS Step Functions e AWS Serverless Application Model
- Tutorial: Creare una macchina a stati Step Functions utilizzando AWS SAM
- AWS SAM Specifiche: AWS::Serverless::StateMachine

Configura la firma del codice per la tua AWS SAM applicazione

Per garantire che venga distribuito solo codice affidabile, puoi utilizzare AWS SAM per abilitare la firma del codice con le tue applicazioni serverless. La firma del codice aiuta a garantire che il codice non sia stato alterato dopo la firma e che nelle funzioni Lambda vengano eseguiti solo pacchetti di codice firmati da editori affidabili. Questo aiuta a liberare le organizzazioni dall'onere di creare componenti gatekeeper nelle loro pipeline di implementazione.

Per ulteriori informazioni sulla firma del codice, consulta Configurazione della firma del codice per le funzioni Lambda nella Guida per AWS Lambda gli sviluppatori.

Prima di poter configurare la firma del codice per la tua applicazione serverless, devi creare un profilo di firma utilizzando Signer. AWS Utilizzi questo profilo di firma per le seguenti attività:

1. Creazione di una configurazione di firma del codice: dichiara una `AWS::Lambda::CodeSigningConfig`risorsa per specificare i profili di firma degli editori attendibili e impostare l'azione politica per i controlli di convalida. È possibile dichiarare questo oggetto nello stesso AWS SAM modello della funzione serverless, in un modello diverso o in un AWS SAM modello. CloudFormation Quindi abiliti la firma del codice per una funzione serverless specificando la `CodeSigningConfigArn`proprietà della funzione con l'Amazon Resource Name (ARN) di `AWS::Lambda::CodeSigningConfig`una risorsa.
2. Firma del codice: utilizza il `sam deploy`comando `sam package`or con l'`--signing-profile`sopzione.

> **Note**
>
> Per firmare correttamente il codice con i `sam deploy` comandi `sam package` o, è necessario abilitare il controllo delle versioni per il bucket Amazon S3 che usi con questi comandi. Se utilizzi il bucket Amazon S3 AWS SAM creato per te, il controllo delle versioni viene abilitato automaticamente. Per ulteriori informazioni sul controllo delle versioni dei bucket Amazon S3 e istruzioni per abilitare il controllo delle versioni su un bucket Amazon S3 da te fornito, consulta Using versioning in Amazon S3 bucket nella Amazon Simple Storage Service User Guide.

Quando distribuisci un'applicazione serverless, Lambda esegue controlli di convalida su tutte le funzioni per le quali hai abilitato la firma del codice. Lambda esegue anche controlli di convalida su tutti i livelli da cui dipendono tali funzioni. Per ulteriori informazioni sui controlli di convalida di Lambda, consulta la convalida della firma nella Guida per gli sviluppatori.AWS Lambda

Esempio

Creazione di un profilo di firma

Per creare un profilo di firma, esegui il seguente comando:

```
aws signer put-signing-profile --platform-id "AWSLambda-SHA384-ECDSA" --profile-
name MySigningProfile
```

Se il comando precedente ha esito positivo, viene restituito l'ARN del profilo di firma. Esempio:

```
{
    "arn": "arn:aws:signer:us-east-1:111122223333:/signing-profiles/MySigningProfile",
    "profileVersion": "SAMPLEverx",
    "profileVersionArn": "arn:aws:signer:us-east-1:111122223333:/signing-
profiles/MySigningProfile/SAMPLEverx"
}
```

Il `profileVersionArn` campo contiene l'ARN da utilizzare quando si crea la configurazione di firma del codice.

Creazione di una configurazione di firma del codice e attivazione della firma del codice per una funzione

Il seguente AWS SAM modello di esempio dichiara una `AWS::Lambda::CodeSigningConfig`risorsa e abilita la firma del codice per una funzione Lambda. In questo esempio, esiste un profilo affidabile e le distribuzioni vengono rifiutate se i controlli della firma hanno esito negativo.

```
Resources:
  HelloWorld:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: hello_world/
      Handler: app.lambda_handler
      Runtime: python3.7
      CodeSigningConfigArn: !Ref MySignedFunctionCodeSigningConfig

  MySignedFunctionCodeSigningConfig:
    Type: AWS::Lambda::CodeSigningConfig
    Properties:
      Description: "Code Signing for MySignedLambdaFunction"
      AllowedPublishers:
        SigningProfileVersionArns:
          - MySigningProfile-profileVersionArn
      CodeSigningPolicies:
        UntrustedArtifactOnDeployment: "Enforce"
```

Firma del codice

Puoi firmare il codice durante il pacchetto o la distribuzione dell'applicazione. Specificate l'`--signing-profiles`opzione con il `sam deploy` comando `sam package` o, come illustrato nei seguenti comandi di esempio.

Firma del codice della funzione al momento del pacchetto dell'applicazione:

```
sam package --signing-profiles HelloWorld=MySigningProfile --s3-bucket amzn-s3-demo-bucket --output-template-file packaged.yaml
```

Firma sia il codice della funzione che il livello da cui dipende la funzione, durante la creazione del pacchetto dell'applicazione:

```
sam package --signing-profiles HelloWorld=MySigningProfile MyLayer=MySigningProfile --s3-bucket amzn-s3-demo-bucket --output-template-file packaged.yaml
```

Firma del codice funzionale e di un livello, quindi esecuzione di una distribuzione:

```
sam deploy --signing-profiles HelloWorld=MySigningProfile MyLayer=MySigningProfile --s3-bucket amzn-s3-demo-bucket --template-file packaged.yaml --stack-name --region us-east-1 --capabilities CAPABILITY_IAM
```

> **Note**
>
> Per firmare correttamente il codice con i `sam deploy` comandi `sam package` o, è necessario abilitare il controllo delle versioni per il bucket Amazon S3 che usi con questi comandi. Se utilizzi il bucket Amazon S3 AWS SAM creato per te, il controllo delle versioni viene abilitato automaticamente. Per ulteriori informazioni sul controllo delle versioni dei bucket Amazon S3 e istruzioni per abilitare il controllo delle versioni su un bucket Amazon S3 da te fornito, consulta Using versioning in Amazon S3 bucket nella Amazon Simple Storage Service User Guide.

Fornire profili di firma con **`sam deploy --guided`**

Quando si esegue il `sam deploy --guided` comando con un'applicazione serverless configurata con la firma del codice, viene AWS SAM richiesto di fornire il profilo di firma da utilizzare per la firma

del codice. Per ulteriori informazioni sui `sam deploy --guided` prompt, sam deploy consulta il riferimento ai comandi. AWS SAMCLI

Convalida dei file AWS SAM modello

Convalida i tuoi modelli con. `sam validate` Attualmente, questo comando verifica che il modello fornito sia JSON/YAML valido. Come la maggior parte dei AWS SAMCLI comandi, per impostazione predefinita cerca un `template.[yaml|yml]` file nella directory di lavoro corrente. È possibile specificare un modello diverso file/location con l'`--template`opzione `-t` o.

Esempio:

```
$ sam validate
<path-to-template>/template.yaml is a valid SAM Template
```

> **Note**
>
> Il `sam validate` comando richiede la configurazione AWS delle credenziali. Per ulteriori informazioni, consulta Configurazione del AWS SAMCLI.

Crea la tua applicazione con AWS SAM

Dopo aver aggiunto l'infrastruttura come codice (IaC) al AWS SAM modello, sarete pronti per iniziare a creare l'applicazione utilizzando il sam build comando. Questo comando crea artefatti di compilazione dai file nella directory del progetto dell'applicazione (ovvero il file AWS SAM modello, il codice dell'applicazione e qualsiasi file e dipendenza specifici del linguaggio applicabile). Questi elementi di build preparano l'applicazione serverless per le fasi successive dello sviluppo dell'applicazione, come il test locale e la distribuzione nel cloud. AWS Sia il test che la distribuzione utilizzano gli artefatti di build come input.

Puoi utilizzarlo sam build per creare la tua intera applicazione serverless. Inoltre, puoi creare build personalizzate, come quelle con funzioni, livelli o runtime personalizzati specifici. Per saperne di più su come e perché lo usisam build, consulta gli argomenti di questa sezione. Per un'introduzione all'uso del `sam build` comando, vedereIntroduzione alla costruzione con AWS SAM.

Argomenti

- Introduzione alla costruzione con AWS SAM

- Compilazione predefinita con AWS SAM
- Personalizza le build con AWS SAM

Introduzione alla costruzione con AWS SAM

Utilizzate il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam build` per preparare l'applicazione serverless per le fasi successive del flusso di lavoro di sviluppo, come il test locale o la distribuzione su. Cloud AWS Questo comando crea una `.aws-sam` directory che struttura l'applicazione in un formato e in una posizione richiesti`sam local`. `sam deploy`

- Per un'introduzione a AWS SAMCLI, vedereChe cos'è il? AWS SAMCLI.
- Per un elenco delle opzioni di `sam build` comando, vederesam build.
- Per un esempio di utilizzo `sam build` durante un tipico flusso di lavoro di sviluppo, vediFase 2: Crea la tua applicazione.

> **Note**
>
> L'utilizzo `sam build` richiede di iniziare con i componenti di base di un'applicazione serverless sulla macchina di sviluppo. Ciò include un AWS SAM modello, un codice di AWS Lambda funzione e qualsiasi file e dipendenza specifici del linguaggio. Per ulteriori informazioni, consulta Crea la tua candidatura in AWS SAM.

Argomenti

- Creazione di applicazioni con sam build
- Test e distribuzione locali
- Best practice
- Opzioni per sam build
- Risoluzione dei problemi
- Esempi
- Ulteriori informazioni

Creazione di applicazioni con sam build

Prima dell'uso`sam build`, valuta la possibilità di configurare quanto segue:

1. Funzioni e livelli Lambda: il `sam build` comando può creare funzioni e livelli Lambda. Per ulteriori informazioni sui layer Lambda, consulta. Creazione di livelli Lambda in AWS SAM
2. Lambda runtime: il runtime fornisce un ambiente specifico del linguaggio che esegue la funzione in un ambiente di esecuzione quando viene richiamata. È possibile configurare runtime nativi e personalizzati.
   a. Runtime nativo: crea le tue funzioni Lambda in un runtime Lambda supportato e crea funzioni per utilizzare un runtime Lambda nativo in. Cloud AWS
   b. Runtime personalizzato: crea le tue funzioni Lambda utilizzando qualsiasi linguaggio di programmazione e crea il tuo runtime utilizzando un processo personalizzato definito in un builder makefile o di terze parti come. esbuild Per ulteriori informazioni, consulta Creazione di funzioni Lambda con runtime personalizzati in AWS SAM.
3. Tipo di pacchetto Lambda: le funzioni Lambda possono essere incluse nei seguenti tipi di pacchetti di distribuzione Lambda:
   a. Archivio di file.zip: contiene il codice dell'applicazione e le sue dipendenze.
   b. Immagine del contenitore: contiene il sistema operativo di base, il runtime, le estensioni Lambda, il codice dell'applicazione e le relative dipendenze.

Queste impostazioni dell'applicazione possono essere configurate durante l'inizializzazione di un'applicazione utilizzando. `sam init`

- Per ulteriori informazioni sull'utilizzo`sam init`, vedere. Crea la tua candidatura in AWS SAM
- Per ulteriori informazioni sulla configurazione di queste impostazioni nell'applicazione, consultaCompilazione predefinita con AWS SAM.

Per creare un'applicazione

1. `cd`alla radice del tuo progetto. Questa è la stessa posizione del AWS SAM modello.

```
$ cd sam-app
```

2. Esegui il seguente codice:

```
sam-app $ sam build <arguments> <options>
```

> **Note**
>
> Un'opzione comunemente usata è`--use-container`. Per ulteriori informazioni, consulta Creazione di una funzione Lambda all'interno di un contenitore fornito.

Di seguito è riportato un esempio dell' AWS SAMCLIoutput:

```
sam-app $ sam build
Starting Build use cache
Manifest file is changed (new hash: 3298f1304...d4d421) or dependency folder (.aws-
sam/deps/4d3dfad6-a267-47a6-a6cd-e07d6fae318c) is missing for (HelloWorldFunction),
 downloading dependencies and copying/building source
Building codeuri: /Users/.../sam-app/hello_world runtime: python3.12 metadata: {}
 architecture: x86_64 functions: HelloWorldFunction
Running PythonPipBuilder:CleanUp
Running PythonPipBuilder:ResolveDependencies
Running PythonPipBuilder:CopySource
Running PythonPipBuilder:CopySource

Build Succeeded

Built Artifacts  : .aws-sam/build
Built Template   : .aws-sam/build/template.yaml

Commands you can use next
=========================
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name {{stack-name}} --watch
[*] Deploy: sam deploy --guided
```

3. Quindi AWS SAMCLI crea una cartella di `.aws-sam` compilazione. Di seguito è riportato un esempio:

```
.aws-sam
### build
#   ### HelloWorldFunction
```

```
#   #   ### __init__.py
#   #   ### app.py
#   #   ### requirements.txt
#   ### template.yaml
### build.toml
```

A seconda di come è configurata l'applicazione, AWS SAMCLI esegue le seguenti operazioni:

1. Scarica, installa e organizza le dipendenze nella directory. `.aws-sam/build`
2. Prepara il codice Lambda. Ciò può includere la compilazione del codice, la creazione di file binari eseguibili e la creazione di immagini di container.
3. Copia gli artefatti della build nella directory. `.aws-sam` Il formato varierà in base al tipo di pacchetto dell'applicazione.
   a. Per i tipi di pacchetti.zip, gli artefatti non sono ancora compressi, quindi possono essere utilizzati per i test locali. Comprime l' AWS SAMCLIapplicazione durante l'utilizzo. `sam deploy`
   b. Per i tipi di pacchetto di immagini del contenitore, un'immagine del contenitore viene creata localmente e referenziata nel `.aws-sam/build.toml` file.
4. Copia il AWS SAM modello nella `.aws-sam` directory e lo modifica con nuovi percorsi di file quando necessario.

Di seguito sono riportati i componenti principali che costituiscono gli artefatti di compilazione presenti nella directory: `.aws-sam`

- La directory di compilazione: contiene le funzioni e i livelli Lambda strutturati indipendentemente l'uno dall'altro. Ciò si traduce in una struttura unica per ogni funzione o livello nella `.aws-sam/build` directory.
- Il AWS SAM modello: modificato con valori aggiornati in base alle modifiche durante il processo di creazione.
- Il file build.toml: un file di configurazione che contiene le impostazioni di build utilizzate da. AWS SAMCLI

Test e distribuzione locali

Quando si eseguono test locali con `sam local` o si distribuiscono con`sam deploy`, AWS SAMCLI esegue le seguenti operazioni:

1. Innanzitutto verifica se esiste una `.aws-sam` directory e se un AWS SAM modello si trova all'interno di quella directory. Se queste condizioni sono soddisfatte, la AWS SAMCLI considera la directory principale dell'applicazione.
2. Se queste condizioni non sono soddisfatte, AWS SAMCLI considera la posizione originale del AWS SAM modello come cartella principale dell'applicazione.

Durante lo sviluppo, se vengono apportate modifiche ai file dell'applicazione originale, esegui `sam build` per aggiornare la `.aws-sam` directory prima di eseguire il test a livello locale.

Best practice

- Non modificare alcun codice contenuto nella `.aws-sam/build` directory. Aggiorna invece il codice sorgente originale nella cartella del progetto ed `sam build` esegui per aggiornare la `.aws-sam/build` directory.
- Quando modifichi i file originali, esegui `sam build` per aggiornare la `.aws-sam/build` directory.
- Potresti voler fare riferimento AWS SAMCLI alla directory principale originale del tuo progetto anziché alla `.aws-sam` directory, ad esempio durante lo sviluppo e il test con`sam local`. Eliminate la `.aws-sam` directory o il AWS SAM modello nella `.aws-sam` directory per fare in modo che AWS SAMCLI riconoscano la directory del progetto originale come cartella principale del progetto. Quando sei pronto, esegui di `sam build` nuovo per creare la `.aws-sam` directory.
- Quando eseguisam `build`, la `.aws-sam/build` directory viene sovrascritta ogni volta. La `.aws-sam` directory no. Se desideri archiviare file, come i registri, memorizzali in `.aws-sam` per evitare che vengano sovrascritti.

Opzioni per sam build

Costruire un'unica risorsa

Fornisci l'ID logico della risorsa per creare solo quella risorsa. Di seguito è riportato un esempio:

```
$ sam build HelloWorldFunction
```

Per creare una risorsa di un'applicazione o di uno stack annidato, fornisci l'ID logico dell'applicazione o dello stack insieme all'ID logico della risorsa utilizzando il formato: *<stack-logical-id>*/*<resource-logical-id>*

```
$ sam build MyNestedStack/MyFunction
```

Creazione di una funzione Lambda all'interno di un contenitore fornito

L'`--use-container`opzione scarica un'immagine del contenitore e la utilizza per creare le funzioni Lambda. Il contenitore locale viene quindi referenziato nel file`.aws-sam/build.toml`.

Questa opzione deve Docker essere installata. Per istruzioni, consulta Installazione di Docker.

Di seguito è riportato un esempio di questo comando:

```
$ sam build --use-container
```

È possibile specificare l'immagine del contenitore da utilizzare con l'`--build-image`opzione. Di seguito è riportato un esempio:

```
$ sam build --use-container --build-image amazon/aws-sam-cli-build-image-nodejs20.x
```

Per specificare l'immagine del contenitore da utilizzare per una singola funzione, fornite l'ID logico della funzione. Di seguito è riportato un esempio:

```
$ sam build --use-container --build-image Function1=amazon/aws-sam-cli-build-image-
python3.12
```

Passa le variabili di ambiente al contenitore di compilazione

Usa `--container-env-var` per passare le variabili di ambiente al contenitore di compilazione. Di seguito è riportato un esempio:

```
$ sam build --use-container --container-env-var Function1.GITHUB_TOKEN=<token1> --
container-env-var GLOBAL_ENV_VAR=<global-token>
```

Per passare le variabili di ambiente da un file, usa l'`--container-env-var-file`opzione. Di seguito è riportato un esempio:

```
$ sam build --use-container --container-env-var-file <env.json>
```

Esempio del `env.json` file:

```
{
  "MyFunction1": {
    "GITHUB_TOKEN": "TOKEN1"
  },
  "MyFunction2": {
    "GITHUB_TOKEN": "TOKEN2"
  }
}
```

Accelera la creazione di applicazioni che contengono più funzioni

Quando viene eseguita `sam build` su un'applicazione con più funzioni, AWS SAMCLI crea ogni funzione una alla volta. Per velocizzare il processo di compilazione, utilizzate l'`--parallel`opzione. Questo crea tutte le funzioni e i livelli contemporaneamente.

Di seguito è riportato un esempio di questo comando:

```
$ sam build —-parallel
```

Accelera i tempi di compilazione creando il progetto nella cartella dei sorgenti

Per i runtime e i metodi di compilazione supportati, puoi utilizzare l'`--build-in-source`opzione per creare il tuo progetto direttamente nella cartella dei sorgenti. Per impostazione predefinita, AWS SAM CLI le build si trovano in una directory temporanea, che prevede la copia del codice sorgente e dei file di progetto. Con`--build-in-source`, AWS SAM CLI le build vengono create direttamente nella cartella di origine, il che accelera il processo di compilazione eliminando la necessità di copiare i file in una directory temporanea.

Per un elenco dei runtime e dei metodi di compilazione supportati, consulta. `--build-in-source`

Risoluzione dei problemi

Per risolvere il problema AWS SAMCLI, vedi. AWS SAMCLIrisoluzione dei problemi

Esempi

Creazione di un'applicazione che utilizza un runtime nativo e un tipo di pacchetto.zip

Per questo esempio, vediTutorial: Implementa un'applicazione Hello World con AWS SAM.

Creazione di un'applicazione che utilizza un tipo di pacchetto di immagini e runtime nativi

Per prima cosa, eseguiamo `sam init` l'inizializzazione di una nuova applicazione. Durante il flusso interattivo, selezioniamo il tipo di `Image` pacchetto. Di seguito è riportato un esempio:

```
$ sam init
...
Which template source would you like to use?
        1 - AWS Quick Start Templates
        2 - Custom Template Location
Choice: 1

Choose an AWS Quick Start application template
        1 - Hello World Example
        2 - Multi-step workflow
        3 - Serverless API
        4 - Scheduled task
        5 - Standalone function
        6 - Data processing
        7 - Hello World Example With Powertools
        8 - Infrastructure event management
        9 - Serverless Connector Hello World Example
        10 - Multi-step workflow with Connectors
        11 - Lambda EFS example
        12 - DynamoDB Example
        13 - Machine Learning
Template: 1

Use the most popular runtime and package type? (Python and zip) [y/N]: ENTER

Which runtime would you like to use?
        ...
        10 - java8
        11 - nodejs20.x
        12 - nodejs18.x
        13 - nodejs16.x
        ...
Runtime: 12

What package type would you like to use?
        1 - Zip
        2 - Image
Package type: 2
```

```
Based on your selections, the only dependency manager available is npm.
We will proceed copying the template using npm.

Would you like to enable X-Ray tracing on the function(s) in your application?  [y/
N]: ENTER

Would you like to enable monitoring using CloudWatch Application Insights?
For more info, please view https://docs.aws.amazon.com/AmazonCloudWatch/latest/
monitoring/cloudwatch-application-insights.html [y/N]: ENTER

Project name [sam-app]: ENTER

Cloning from https://github.com/aws/aws-sam-cli-app-templates (process may take a
 moment)

    -----------------------
    Generating application:
    -----------------------
    Name: sam-app
    Base Image: amazon/nodejs18.x-base
    Architectures: x86_64
    Dependency Manager: npm
    Output Directory: .
    Configuration file: sam-app/samconfig.toml

    Next steps can be found in the README file at sam-app/README.md

...
```

AWS SAMCLinizializza un'applicazione e crea la seguente directory di progetto:

```
sam-app
### README.md
### events
#   ### event.json
### hello-world
#   ### Dockerfile
#   ### app.mjs
#   ### package.json
#   ### tests
#       ### unit
#           ### test-handler.mjs
### samconfig.toml
```

```
### template.yaml
```

Successivamente, corriamo `sam build` a creare la nostra applicazione:

```
sam-app $ sam build
Building codeuri: /Users/.../build-demo/sam-app runtime: None metadata: {'DockerTag':
 'nodejs18.x-v1', 'DockerContext': '/Users/.../build-demo/sam-app/hello-world',
 'Dockerfile': 'Dockerfile'} architecture: arm64 functions: HelloWorldFunction
Building image for HelloWorldFunction function
Setting DockerBuildArgs: {} for HelloWorldFunction function
Step 1/4 : FROM public.ecr.aws/lambda/nodejs:18
 ---> f5b68038c080
Step 2/4 : COPY app.mjs package*.json ./
 ---> Using cache
 ---> 834e565aae80
Step 3/4 : RUN npm install
 ---> Using cache
 ---> 31c2209dd7b5
Step 4/4 : CMD ["app.lambdaHandler"]
 ---> Using cache
 ---> 2ce2a438e89d
Successfully built 2ce2a438e89d
Successfully tagged helloworldfunction:nodejs18.x-v1

Build Succeeded

Built Artifacts  : .aws-sam/build
Built Template   : .aws-sam/build/template.yaml

Commands you can use next
=========================
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name {{stack-name}} --watch
[*] Deploy: sam deploy --guided
```

Creazione di un'applicazione che includa un linguaggio di programmazione compilato

In questo esempio, creiamo un'applicazione che contiene una funzione Lambda utilizzando il Go runtime.

Innanzitutto, inizializziamo una nuova applicazione utilizzando `sam init` e configuriamo la nostra applicazione per utilizzare: Go

```
$ sam init

...

Which template source would you like to use?
        1 - AWS Quick Start Templates
        2 - Custom Template Location
Choice: 1

Choose an AWS Quick Start application template
        1 - Hello World Example
        2 - Multi-step workflow
        3 - Serverless API
        ...
Template: 1

Use the most popular runtime and package type? (Python and zip) [y/N]: ENTER

Which runtime would you like to use?
        ...
        4 - dotnetcore3.1
        5 - go1.x
        6 - go (provided.al2)
        ...
Runtime: 5

What package type would you like to use?
        1 - Zip
        2 - Image
Package type: 1

Based on your selections, the only dependency manager available is mod.
We will proceed copying the template using mod.

Would you like to enable X-Ray tracing on the function(s) in your application?  [y/
N]: ENTER

Would you like to enable monitoring using CloudWatch Application Insights?
For more info, please view https://docs.aws.amazon.com/AmazonCloudWatch/latest/
monitoring/cloudwatch-application-insights.html [y/N]: ENTER

Project name [sam-app]: ENTER
```

```
Cloning from https://github.com/aws/aws-sam-cli-app-templates (process may take a
 moment)

    -----------------------
    Generating application:
    -----------------------
    Name: sam-app
    Runtime: go1.x
    Architectures: x86_64
    Dependency Manager: mod
    Application Template: hello-world
    Output Directory: .
    Configuration file: sam-app/samconfig.toml

    Next steps can be found in the README file at sam-app-go/README.md

...
```

Quindi AWS SAMCLI inizializza l'applicazione. Di seguito è riportato un esempio della struttura delle cartelle dell'applicazione:

```
sam-app
### Makefile
### README.md
### events
#   ### event.json
### hello-world
#   ### go.mod
#   ### go.sum
#   ### main.go
#   ### main_test.go
### samconfig.toml
### template.yaml
```

Facciamo riferimento al `README.md` file per i requisiti di questa applicazione.

```
...
## Requirements
* AWS CLI already configured with Administrator permission
* [Docker installed](https://www.docker.com/community-edition)
* [Golang](https://golang.org)
* SAM CLI - [Install the SAM CLI](https://docs.aws.amazon.com/serverless-application-
model/latest/developerguide/serverless-sam-cli-install.html)
```

```
...
```

Successivamente, corriamo `sam local invoke` per testare la nostra funzione. Questo comando ha un errore poiché non Go è installato sul nostro computer locale:

```
sam-app $ sam local invoke
Invoking hello-world (go1.x)
Local image was not found.
Removing rapid images for repo public.ecr.aws/sam/emulation-go1.x
Building
 image.....................................................................................
Using local image: public.ecr.aws/lambda/go:1-rapid-x86_64.

Mounting /Users/.../Playground/build/sam-app/hello-world as /var/task:ro,delegated
 inside runtime container
START RequestId: c6c5eddf-042b-4e1e-ba66-745f7c86dd31 Version: $LATEST
fork/exec /var/task/hello-world: no such file or directory: PathError
null
END RequestId: c6c5eddf-042b-4e1e-ba66-745f7c86dd31
REPORT RequestId: c6c5eddf-042b-4e1e-ba66-745f7c86dd31  Init Duration: 0.88 ms
 Duration: 175.75 ms Billed Duration: 176 ms Memory Size: 128 MB     Max Memory Used:
 128 MB
{"errorMessage":"fork/exec /var/task/hello-world: no such file or
 directory","errorType":"PathError"}%
```

Successivamente, corriamo `sam build` a creare la nostra applicazione. Riscontriamo un errore poiché non Go è installato sul nostro computer locale:

```
sam-app $ sam build
Starting Build use cache
Cache is invalid, running build and copying resources for following functions
 (HelloWorldFunction)
Building codeuri: /Users/.../Playground/build/sam-app/hello-world runtime: go1.x
 metadata: {} architecture: x86_64 functions: HelloWorldFunction

Build Failed
Error: GoModulesBuilder:Resolver - Path resolution for runtime: go1.x of binary: go was
 not successful
```

Sebbene sia possibile configurare la nostra macchina locale per creare correttamente la nostra funzione, utilizziamo invece l'`--use-container`opzione con`sam build`. AWS SAMCLIScarica

l'immagine di un contenitore, crea la nostra funzione usando quella nativa GoModulesBuilder e copia il file binario risultante `.aws-sam/build/HelloWorldFunction` nella nostra directory.

```
sam-app $ sam build --use-container
Starting Build use cache
Starting Build inside a container
Cache is invalid, running build and copying resources for following functions
 (HelloWorldFunction)
Building codeuri: /Users/.../build/sam-app/hello-world runtime: go1.x metadata: {}
 architecture: x86_64 functions: HelloWorldFunction

Fetching public.ecr.aws/sam/build-go1.x:latest-x86_64 Docker container
 image.....................................................................................
Mounting /Users/.../build/sam-app/hello-world as /tmp/samcli/source:ro,delegated inside
 runtime container
Running GoModulesBuilder:Build

Build Succeeded

Built Artifacts  : .aws-sam/build
Built Template   : .aws-sam/build/template.yaml

Commands you can use next
=========================
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name {{stack-name}} --watch
[*] Deploy: sam deploy --guided
```

Di seguito è riportato un esempio della `.aws-sam` directory:

```
.aws-sam
### build
#   ### HelloWorldFunction
#   #   ### hello-world
#   ### template.yaml
### build.toml
### cache
#   ### c860d011-4147-4010-addb-2eaa289f4d95
#       ### hello-world
### deps
```

Successivamente, corriamo`sam local invoke`. La nostra funzione è stata richiamata con successo:

```
sam-app $ sam local invoke
Invoking hello-world (go1.x)
Local image is up-to-date
Using local image: public.ecr.aws/lambda/go:1-rapid-x86_64.

Mounting /Users/.../Playground/build/sam-app/.aws-sam/build/HelloWorldFunction as /var/
task:ro,delegated inside runtime container
START RequestId: cfc8ffa8-29f2-49d4-b461-45e8c7c80479 Version: $LATEST
END RequestId: cfc8ffa8-29f2-49d4-b461-45e8c7c80479
REPORT RequestId: cfc8ffa8-29f2-49d4-b461-45e8c7c80479  Init Duration: 1.20 ms
 Duration: 1782.46 ms        Billed Duration: 1783 ms        Memory Size: 128 MB
 Max Memory Used: 128 MB
{"statusCode":200,"headers":null,"multiValueHeaders":null,"body":"Hello,
 72.21.198.67\n"}%
```

Ulteriori informazioni

Per ulteriori informazioni sull'utilizzo del `sam build` comando, consulta quanto segue:

- Apprendimento AWS SAM: sam build — Serie Serverless Land «Learning AWS SAM» suYouTube.
- Apprendimento AWS SAM | sam build | E3 — Continuata la serie Serverless Land «Learning AWS SAM». YouTube
- AWS SAM build: come fornisce gli artefatti per l'implementazione (Sessions With SAM S2E8) — Sessioni con serie attivate. AWS SAM YouTube
- AWS SAM build personalizzate: come usare Makefile per personalizzare le build in SAM (S2E9) — Sessioni con serie accese. AWS SAM YouTube

Compilazione predefinita con AWS SAM

Per creare la tua applicazione serverless, usa il `sam build` comando. Questo comando raccoglie anche gli elementi di compilazione delle dipendenze dell'applicazione e li colloca nel formato e nella posizione corretti per le fasi successive, come il test, la creazione di pacchetti e la distribuzione a livello locale.

È possibile specificare le dipendenze dell'applicazione in un file manifest, ad esempio `requirements.txt` (Python) `package.json` o (Node.js), oppure utilizzando `Layers` la proprietà

di una risorsa funzione. La `Layers` proprietà contiene un elenco di risorse di AWS Lambda livello da cui dipende la funzione Lambda.

Il formato degli elementi di compilazione dell'applicazione dipende dalla proprietà di ciascuna funzione. `PackageType` Le opzioni per questa proprietà sono:

- **`Zip`**— Un archivio di file.zip, che contiene il codice dell'applicazione e le sue dipendenze. Se impacchettate il codice come archivio di file.zip, dovete specificare un runtime Lambda per la funzione.
- **`Image`**— Un'immagine del contenitore, che include il sistema operativo di base, il runtime e le estensioni, oltre al codice dell'applicazione e alle sue dipendenze.

Per ulteriori informazioni sui tipi di pacchetti Lambda, consulta Pacchetti di distribuzione Lambda nella Developer Guide.AWS Lambda

Argomenti

- Creazione di un archivio di file.zip
- Creazione di un'immagine di contenitore
- File variabile di ambiente del contenitore
- Accelera i tempi di compilazione creando il progetto nella cartella dei sorgenti
- Esempi
- Costruire funzioni al di fuori di AWS SAM

Creazione di un archivio di file.zip

Per creare un'applicazione serverless come archivio di file.zip, dichiarate `PackageType: Zip` di utilizzare la funzione serverless.

AWS SAM crea l'applicazione per l'architettura specificata. Se non si specifica un'architettura, AWS SAM utilizza `x86_64` per impostazione predefinita.

Se la tua funzione Lambda dipende da pacchetti con programmi compilati nativamente, usa il flag. `--use-container` Questo flag compila localmente le tue funzioni in un contenitore che si comporta come un ambiente Lambda, quindi sono nel formato giusto quando le distribuisci sul Cloud. AWS

Quando utilizzi l'`--use-container`opzione, per impostazione predefinita AWS SAM estrae l'immagine del contenitore da Amazon ECR Public. Se desideri estrarre un'immagine del contenitore

da un altro repository o per una versione specifica della AWS SAM CLI, puoi utilizzare l'`--build-image`opzione e fornire l'URI di un'immagine contenitore alternativa. Di seguito sono riportati due comandi di esempio per la creazione di applicazioni che utilizzano immagini di contenitori da una versione specifica della AWS SAM CLI:

```
# Build a Node.js 20 application using a container image for a specific version of AWS
 SAM CLI (1.136.0)
sam build --use-container --build-image public.ecr.aws/sam/build-nodejs22.x:1.136.0

# Build a function resource using the Python 3.13 container image from a specific
 version of AWS SAM CLI (1.136.0)(
sam build --use-container --build-image Function1=public.ecr.aws/sam/build-
python3.13:1.136.0
```

Per ulteriori esempi di creazione di un'applicazione di archiviazione di file con estensione zip, consultate la sezione Esempi più avanti in questo argomento.

Creazione di un'immagine di contenitore

Per creare un'applicazione serverless come immagine contenitore, dichiarate di utilizzare `PackageType: Image` la funzione serverless. È inoltre necessario dichiarare l'attributo `Metadata` resource con le seguenti voci:

`Dockerfile`

Il nome del Dockerfile associato alla funzione Lambda.

`DockerContext`

La posizione del Dockerfile.

`DockerTag`

(Facoltativo) Un tag da applicare all'immagine costruita.

`DockerBuildArgs`

Crea argomenti per la compilazione.

> **Important**
>
> AWS SAMCLINon oscura né offusca le informazioni che includi negli argomenti. `DockerBuildArgs` Si consiglia vivamente di non utilizzare questa sezione per archiviare informazioni riservate, ad esempio password o segreti.

Di seguito è riportato un esempio `Metadata` di sezione relativa agli attributi delle risorse:

```
Metadata:
  Dockerfile: Dockerfile
  DockerContext: ./hello_world
  DockerTag: v1
```

Per scaricare un'applicazione di esempio configurata con il tipo di `Image` pacchetto, consultaTutorial: Implementa un'applicazione Hello World con AWS SAM. Quando viene richiesto il tipo di pacchetto da installare, sceglieteImage.

> **Note**
>
> Se specifichi un'immagine di base multiarchitettura nel tuo Dockerfile, AWS SAM crea l'immagine del contenitore per l'architettura della tua macchina host. Per creare per un'architettura diversa, specifica un'immagine di base che utilizzi l'architettura di destinazione specifica.

File variabile di ambiente del contenitore

Per fornire un file JSON che contenga variabili di ambiente per il contenitore di compilazione, usa l'`--container-env-var-file`argomento con il `sam build` comando. Puoi fornire una singola variabile di ambiente che si applica a tutte le risorse serverless o variabili di ambiente diverse per ogni risorsa.

Formato

Il formato per il passaggio delle variabili di ambiente a un contenitore di build dipende dal numero di variabili di ambiente fornite per le risorse.

Per fornire un'unica variabile di ambiente per tutte le risorse, specifica un `Parameters` oggetto come il seguente:

```
{
  "Parameters": {
    "GITHUB_TOKEN": "TOKEN_GLOBAL"
  }
}
```

Per fornire variabili di ambiente diverse per ogni risorsa, specificate gli oggetti per ogni risorsa come segue:

```
{
  "MyFunction1": {
    "GITHUB_TOKEN": "TOKEN1"
  },
  "MyFunction2": {
    "GITHUB_TOKEN": "TOKEN2"
  }
}
```

Salvate le variabili di ambiente come file, ad esempio denominato`env.json`. Il comando seguente utilizza questo file per passare le variabili di ambiente al contenitore di compilazione:

```
sam build --use-container --container-env-var-file env.json
```

Priorità

- Le variabili di ambiente fornite per risorse specifiche hanno la precedenza sulla singola variabile di ambiente per tutte le risorse.
- Le variabili di ambiente fornite nella riga di comando hanno la precedenza sulle variabili di ambiente in un file.

Accelera i tempi di compilazione creando il progetto nella cartella dei sorgenti

Per i runtime e i metodi di compilazione supportati, puoi utilizzare l'`--build-in-source`opzione per creare il tuo progetto direttamente nella cartella di origine. Per impostazione predefinita, AWS SAM CLI le build si trovano in una directory temporanea, che prevede la copia del codice sorgente e dei

file di progetto. Con`--build-in-source`, AWS SAM CLI le build vengono create direttamente nella cartella di origine, il che accelera il processo di compilazione eliminando la necessità di copiare i file in una directory temporanea.

Per un elenco dei runtime e dei metodi di compilazione supportati, consulta. `--build-in-source`

Esempi

Esempio 1: archivio di file.zip

I seguenti `sam build` comandi creano un archivio di file.zip:

```
# Build all functions and layers, and their dependencies
sam build

# Run the build process inside a Docker container that functions like a Lambda
 environment
sam build --use-container

# Build a Node.js 20 application using a container image for a specific version of AWS
 SAM CLI (1.136.0)
sam build --use-container --build-image public.ecr.aws/sam/build-nodejs22.x:1.136.0

# Build a function resource using the Python 3.13 container image from a specific
 version of AWS SAM CLI (1.136.0)(
sam build --use-container --build-image Function1=public.ecr.aws/sam/build-
python3.13:1.136.0

# Build and run your functions locally
sam build && sam local invoke

# For more options
sam build --help
```

Esempio 2: immagine del contenitore

Il seguente AWS SAM modello viene creato come immagine del contenitore:

```
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Properties:
```

```
      PackageType: Image
      ImageConfig:
        Command: ["app.lambda_handler"]
    Metadata:
      Dockerfile: Dockerfile
      DockerContext: ./hello_world
      DockerTag: v1
```

Di seguito è riportato un esempio di Dockerfile:

```
FROM public.ecr.aws/lambda/python:3.12

COPY app.py requirements.txt ./

RUN python3.12 -m pip install -r requirements.txt

# Overwrite the command by providing a different command directly in the template.
CMD ["app.lambda_handler"]
```

Esempio 3: npm ci

Per le applicazioni Node.js, è possibile utilizzare `npm ci` invece di `npm install` installare le dipendenze. Per utilizzarlo`npm ci`, specifica `UseNpmCi: True` sotto `BuildProperties` nell'attributo `Metadata` resource della funzione Lambda. Per essere utilizzata`npm ci`, l'applicazione deve avere un `npm-shrinkwrap.json` file `package-lock.json` or presente nella funzione `CodeUri` for your Lambda.

L'esempio seguente utilizza `npm ci` per installare le dipendenze durante l'esecuzione: `sam build`

```
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: hello-world/
      Handler: app.handler
      Runtime: nodejs20.x
      Architectures:
        - x86_64
      Events:
        HelloWorld:
          Type: Api
          Properties:
```

```
            Path: /hello
            Method: get
    Metadata:
      BuildProperties:
        UseNpmCi: True
```

Pacchetti principali in Python

Per le applicazioni Python, puoi preservare la struttura del pacchetto durante il processo di compilazione per abilitare le importazioni assolute. Per preservare la struttura del pacchetto, specifica `ParentPackageMode` sotto `BuildProperties` nell'attributo `Metadata` resource della funzione Lambda.

L'esempio seguente conserva la struttura del `app` pacchetto durante l'esecuzione: `sam build`

```
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: hello-world/
      Handler: app.main.handler
      Runtime: python3.12
      Architectures:
        - x86_64
    Metadata:
      BuildProperties:
        ParentPackageMode: explicit
        ParentPackages: app
```

Con questa configurazione, il codice può utilizzare importazioni assolute come `from app.utils import logger` invece di importazioni relative come`from .utils import logger`.

Costruire funzioni al di fuori di AWS SAM

Per impostazione predefinita, quando si eseguesam build, AWS SAM crea tutte le risorse funzionali. Altre opzioni includono:

- Costruisci tutte le risorse funzionali all'esterno di AWS SAM: se crei tutte le tue risorse funzionali manualmente o tramite un altro strumento, non sam build è necessario. Puoi saltare sam build e passare alla fase successiva del processo, ad esempio l'esecuzione di test locali o la distribuzione dell'applicazione.

- Crea alcune risorse funzionali all'esterno AWS SAM: se desideri AWS SAM creare alcune delle tue risorse funzionali con altre risorse funzionali integrate all'esterno AWS SAM, puoi specificarlo nel tuo AWS SAM modello.

Crea alcune risorse funzionali al di fuori di AWS SAM

Per fare in modo che una funzione venga AWS SAM ignorata durante l'utilizzosam build, configura quanto segue nel AWS SAM modello:

1. Aggiungi la proprietà `SkipBuild: True` dei metadati alla tua funzione.
2. Specificate il percorso delle risorse funzionali integrate.

Ecco un esempio, con `TestFunction` configurato per essere ignorato. Le sue risorse integrate si trovano in`built-resources/TestFunction.zip`.

```
TestFunction:
  Type: AWS::Serverless::Function
  Properties:
    CodeUri: built-resources/TestFunction.zip
    Handler: TimeHandler::handleRequest
    Runtime: java11
  Metadata:
    SkipBuild: True
```

Ora, quando corrisam build, AWS SAM farà quanto segue:

1. AWS SAM salterà le funzioni configurate con`SkipBuild: True`.
2. AWS SAM creerà tutte le altre risorse funzionali e le memorizzerà nella directory di `.aws-sam` compilazione.
3. Per le funzioni ignorate, il relativo modello nella directory di `.aws-sam` compilazione verrà automaticamente aggiornato per fare riferimento al percorso specificato delle risorse delle funzioni create.

   Ecco un esempio del modello memorizzato nella cache per `TestFunction` nella directory di `.aws-sam` compilazione:

   ```
   TestFunction:
     Type: AWS::Serverless::Function
   ```

```
  Properties:
    CodeUri: ../../built-resources/TestFunction.zip
    Handler: TimeHandler::handleRequest
    Runtime: java11
  Metadata:
    SkipBuild: True
```

Personalizza le build con AWS SAM

Puoi personalizzare la build per includere funzioni Lambda o layer Lambda specifici. Una funzione è una risorsa che è possibile invocare per eseguire il codice in Lambda. Un layer Lambda consente di estrarre codice da una funzione Lambda che può quindi essere riutilizzato in diverse funzioni Lambda. Puoi scegliere di personalizzare la tua build con funzioni Lambda specifiche se desideri concentrarti sullo sviluppo e la distribuzione di singole funzioni serverless senza la complessità della gestione di dipendenze o risorse condivise. Inoltre, puoi scegliere di creare un layer Lambda per aiutarti a ridurre le dimensioni dei pacchetti di distribuzione, separare la logica delle funzioni principali dalle dipendenze e consentire di condividere le dipendenze tra più funzioni.

Gli argomenti di questa sezione esplorano alcuni dei diversi modi con cui è possibile creare funzioni Lambda. AWS SAM Ciò include la creazione di funzioni Lambda con i runtime dei clienti e la creazione di layer Lambda. I runtime personalizzati consentono di installare e utilizzare un linguaggio non elencato nei runtime Lambda nella Developer Guide. AWS Lambda Ciò consente di creare un ambiente di esecuzione specializzato per l'esecuzione di funzioni e applicazioni serverless. La creazione di soli livelli Lambda (anziché la creazione dell'intera applicazione) può apportare vantaggi in alcuni modi. Può aiutarvi a ridurre le dimensioni dei pacchetti di distribuzione, separare la logica delle funzioni principali dalle dipendenze e consentire di condividere le dipendenze tra più funzioni.

Per ulteriori informazioni sulle funzioni, consulta i concetti di Lambda nella AWS Lambda Developer Guide.

Argomenti

- Creazione di funzioni Lambda Node.js con esbuild in AWS SAM
- Creazione di funzioni.NET Lambda con la compilazione AOT nativa in AWS SAM
- Creazione di funzioni Rust Lambda con in Cargo Lambda AWS SAM
- Creazione di funzioni Lambda in Python con in uv AWS SAM
- Creazione di funzioni Lambda con runtime personalizzati in AWS SAM
- Creazione di livelli Lambda in AWS SAM

Creazione di funzioni Lambda Node.js con esbuild in AWS SAM

Per creare e impacchettare AWS Lambda le funzioni Node.js, puoi usarle AWS SAMCLI con il bundler esbuild JavaScript . Il bundler esbuild supporta le funzioni Lambda in cui scrivi. TypeScript

Per creare una funzione Lambda Node.js con esbuild, aggiungi un `Metadata` oggetto alla tua `AWS:Serverless::Function` risorsa e specifica `esbuild` per. `BuildMethod` Quando esegui il sam build comando, AWS SAM usa esbuild per raggruppare il codice della funzione Lambda.

Proprietà dei metadati

L'`Metadata`oggetto supporta le seguenti proprietà per esbuild.

BuildMethod

Specifica il bundler per l'applicazione. L'unico valore supportato è `esbuild`.

BuildProperties

Specifica le proprietà di compilazione per il codice della funzione Lambda.

L'`BuildPropertiesoggetto` supporta le seguenti proprietà per esbuild. Tutte le proprietà sono opzionali. Per impostazione predefinita, AWS SAM utilizza il gestore di funzioni Lambda per il punto di ingresso.

EntryPoints

Specifica i punti di ingresso per l'applicazione.

Esterno

Specifica l'elenco dei pacchetti da omettere dalla compilazione. Per ulteriori informazioni, consulta External nel sito Web. esbuild

Formato

Specificate il formato di output dei JavaScript file generati nell'applicazione. Per ulteriori informazioni, consulta Format nel sito web esbuild.

Loader

Specifica l'elenco delle configurazioni per il caricamento dei dati per un determinato tipo di file.

MainFields

Speciifica quali `package.json` campi provare a importare durante la risoluzione di un pacchetto. Il valore predefinito è `main,module`.

Minimizza

Specifica se minimizzare il codice di output in bundle. Il valore predefinito è `true`.

OutExtension

Personalizza l'estensione dei file generati da esbuild. Per ulteriori informazioni, consulta l'estensione Out nel sito Web di esbuild.

Mappa dei sorgenti

Speciifica se il bundler produce un file di mappa sorgente. Il valore predefinito è `false`.

Se impostato su`true`, `NODE_OPTIONS: --enable-source-maps` viene aggiunto alle variabili di ambiente della funzione Lambda e viene generata e inclusa nella funzione una mappa di origine.

In alternativa, quando `NODE_OPTIONS: --enable-source-maps` è incluso nelle variabili di ambiente della funzione, `Sourcemap` viene impostato automaticamente su. `true`

In caso di conflitto, ha `Sourcemap: false` la precedenza su. `NODE_OPTIONS: --enable-source-maps`

> **Note**
>
> Per impostazione predefinita, Lambda crittografa tutte le variabili di ambiente inattive con AWS Key Management Service ().AWS KMS Quando si utilizzano mappe di origine, affinché la distribuzione abbia esito positivo, il ruolo di esecuzione della funzione deve disporre dell'autorizzazione per eseguire l'`kms:Encrypt`azione.

SourcesContent

Speciifica se includere il codice sorgente nel file della mappa di origine. Configura questa proprietà quando `Sourcemap` è impostata su. `'true'`

- `SourcesContent: 'true'`Specificare di includere tutto il codice sorgente.

- `SourcesContent: 'false'`Specificare di escludere tutto il codice sorgente. Ciò si traduce in una riduzione delle dimensioni dei file delle mappe di origine, utili nella produzione in quanto riducono i tempi di avvio. Tuttavia, il codice sorgente non sarà disponibile nel debugger.

Il valore predefinito è `SourcesContent: true`.

Per ulteriori informazioni, consulta il contenuto di Sources nel sito Web di esbuild.

Target

Specifica la versione di destinazione ECMAScript . Il valore predefinito è `es2020`.

TypeScript Esempio di funzione Lambda

Il seguente frammento di AWS SAM modello di esempio utilizza esbuild per creare una funzione Lambda Node.js dal codice in. TypeScript `hello-world/app.ts`

```
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: hello-world/
      Handler: app.handler
      Runtime: nodejs20.x
      Architectures:
        - x86_64
      Events:
        HelloWorld:
          Type: Api
          Properties:
            Path: /hello
            Method: get
      Environment:
        Variables:
          NODE_OPTIONS: --enable-source-maps
    Metadata:
      BuildMethod: esbuild
      BuildProperties:
        Format: esm
        Minify: false
        OutExtension:
          - .js=.mjs
        Target: "es2020"
```

```
        Sourcemap: true
        EntryPoints:
          - app.ts
        External:
          - "<package-to-exclude>"
```

Creazione di funzioni.NET Lambda con la compilazione AOT nativa in AWS SAM

Crea e impacchetta le tue AWS Lambda funzioni.NET 8 con AWS Serverless Application Model (AWS SAM), utilizzando la compilazione Native Ahead-of-Time (AOT) per migliorare i tempi di avvio a freddo. AWS Lambda

Argomenti

- Panoramica su AOT nativo.NET 8
- Utilizzo AWS SAM con le funzioni Lambda di.NET 8
- Prerequisiti di installazione
- Definisci le funzioni.NET 8 Lambda nel tuo modello AWS SAM
- Crea la tua applicazione con AWS SAMCLI
- Ulteriori informazioni

Panoramica su AOT nativo.NET 8

Storicamente, le funzioni.NET Lambda hanno tempi di avvio a freddo che influiscono sull'esperienza utente, sulla latenza del sistema e sui costi di utilizzo delle applicazioni serverless. Con la compilazione AOT nativa.NET, puoi migliorare i tempi di avvio a freddo delle tue funzioni Lambda. Per ulteriori informazioni su Native AOT for .NET 8, consulta Using Native AOT nel repository Dotnet. GitHub

Utilizzo AWS SAM con le funzioni Lambda di.NET 8

Effettua le seguenti operazioni per configurare le funzioni Lambda di.NET 8 con AWS Serverless Application Model ()AWS SAM:

- Installa i prerequisiti sulla tua macchina di sviluppo.
- Definisci le funzioni.NET 8 Lambda nel tuo AWS SAM modello.
- Crea la tua applicazione con. AWS SAMCLI

Prerequisiti di installazione

I seguenti sono i prerequisiti obbligatori:

- Il AWS SAMCLI
- La CLI.NET Core
- Lo strumento globale.NET Core di Amazon.Lambda.Tools
- Docker

Installa il AWS SAMCLI

1. Per verificare se è già AWS SAMCLI installato, esegui quanto segue:

   ```
   sam --version
   ```

2. Per installare AWS SAMCLI, vedereInstalla il AWS SAMCLI.
3. Per aggiornare una versione installata di AWS SAMCLI, vedereAggiornamento del AWS SAMCLI.

Installazione di.NET Core CLI

1. Per scaricare e installare il.NET Core CLI, vedi Scaricare.NET dal sito Web di Microsoft.
2. Per ulteriori informazioni su .NET Core CLI, consulta.NET Core CLI nella Developer Guide.AWS Lambda

Installa lo strumento globale Amazon.Lambda.Tools .NET Core

1. Esegui il comando seguente:

   ```
   dotnet tool install -g Amazon.Lambda.Tools
   ```

2. Se lo strumento è già stato installato, è possibile verificare di avere in uso la versione più recente con il comando seguente:

   ```
   dotnet tool update -g Amazon.Lambda.Tools
   ```

3. Per ulteriori informazioni sullo strumento globale Amazon.Lambda.Tools .NET Core, consulta l'archivio Extensions AWS for .NET CLI su. GitHub

Installazione di Docker

- La creazione con Native AOT richiede l'installazione. Docker Per le istruzioni di installazione, consulta Installazione di Docker da utilizzare con AWS SAMCLI.

Definisci le funzioni.NET 8 Lambda nel tuo modello AWS SAM

Per definire un. NET8 Funzione Lambda nel AWS SAM modello, procedi come segue:

1. Esegui il comando seguente da una directory iniziale a tua scelta:

```
sam init
```

2. Seleziona `AWS Quick Start Templates` per scegliere un modello di partenza.
3. Scegli il `Hello World Example` modello.
4. Scegli di non utilizzare il runtime e il tipo di pacchetto più diffusi inserendon.
5. Per il runtime, sceglietedotnet8.
6. Per il tipo di pacchetto, scegliZip.
7. Per il tuo modello iniziale, scegliHello World Example using native AOT.

Installazione di Docker

- La creazione con Native AOT richiede Docker l'installazione. Per le istruzioni di installazione, consulta Installazione di Docker da utilizzare con AWS SAMCLI.

```
Resources:
HelloWorldFunction:
  Type: AWS::Serverless::Function
  Properties:
    CodeUri: ./src/HelloWorldAot/
    Handler: bootstrap
    Runtime: dotnet8
    Architectures:
      - x86_64
    Events:
      HelloWorldAot:
        Type: Api
        Properties:
```

```
          Path: /hello
          Method: get
```

> **Note**
>
> Quando la `Event` proprietà di an `AWS::Serverless::Function` è impostata su`Api`, ma la `RestApiId` proprietà non è specificata, AWS SAM genera la `AWS::ApiGateway::RestApi` CloudFormation risorsa.

Crea la tua applicazione con AWS SAMCLI

Dalla directory principale del progetto, esegui il `sam build` comando per iniziare a creare l'applicazione. Se la `PublishAot` proprietà è stata definita nel file di progetto.NET 8, AWS SAMCLI verrà creata con la compilazione AOT nativa. Per ulteriori informazioni sulla *`PublishAot`* proprietà, consulta Native AOT Deployment nella documentazione .NET di Microsoft.

Per creare la tua funzione, AWS SAMCLI richiama la CLI.NET Core che utilizza lo strumento globale Amazon.Lambda.Tools .NET Core.

> **Note**
>
> Durante la creazione, se un `.sln` file esiste nella stessa directory o nella directory principale del progetto, la directory contenente il file verrà montata nel contenitore. `.sln` Se non viene trovato un `.sln` file, viene montata solo la cartella del progetto. Pertanto, se state creando un'applicazione multiprogetto, assicuratevi che il `.sln` file sia posizionato correttamente.

Ulteriori informazioni

Per ulteriori informazioni sulla creazione di funzioni.NET 8 Lambda, vedere Introduzione al runtime di.NET 8 per. AWS Lambda

Per un riferimento al sam build comando, vedere sam build.

Creazione di funzioni Rust Lambda con in Cargo Lambda AWS SAM

Questa funzionalità è disponibile in anteprima AWS SAM ed è soggetta a modifiche.

Usa l'interfaccia a riga di AWS Serverless Application Model comando (AWS SAMCLI) con le tue AWS Lambda funzioni Rust.

Argomenti

- Prerequisiti
- Configurazione AWS SAM per l'uso con le funzioni Rust Lambda
- Esempi

Prerequisiti

Rustlingua

Per l'installazioneRust, consulta il sito Web Installa Rust nella Rust lingua.

Cargo Lambda

AWS SAMCLIRichiede l'installazione di Cargo Lambda, un sottocomando perCargo. Per le istruzioni di installazione, vedere Installazione nella Cargo Lambdadocumentazione.

Docker

La creazione e il test delle funzioni Rust Lambda richiedono. Docker Per le istruzioni di installazione, consulta Installazione di Docker.

Attiva la funzionalità AWS SAMCLI beta

Poiché questa funzionalità è disponibile in anteprima, devi attivarla utilizzando uno dei seguenti metodi:

1. Usa la variabile di ambiente:`SAM_CLI_BETA_RUST_CARGO_LAMBDA=1`.
2. Aggiungi il codice seguente al file `samconfig.toml`:

```
[default.build.parameters]
beta_features = true
[default.sync.parameters]
beta_features = true
```

3. Usa l'`--beta-features`opzione quando usi un AWS SAMCLI comando supportato. Esempio:

```
$ sam build --beta-features
```

4. Scegli l'opzione y quando AWS SAMCLI ti viene richiesto di aderire. Di seguito è riportato un esempio:

```
$ sam build
Starting Build use cache
Build method "rust-cargolambda" is a beta feature.
Please confirm if you would like to proceed
You can also enable this beta feature with "sam build --beta-features". [y/N]: y
```

Configurazione AWS SAM per l'uso con le funzioni Rust Lambda

Fase 1: Configura il modello AWS SAM

Configura il tuo AWS SAM modello con quanto segue:

- Binario: facoltativo. Specificate quando il modello contiene più funzioni Rust Lambda.
- BuildMethod – `rust-cargolambda`.
- CodeUri— percorso del `Cargo.toml` file.
- Gestore —`bootstrap`.
- Durata —`provided.al2`.

Per ulteriori informazioni sui runtime personalizzati, consulta la sezione AWS Lambda Runtime personalizzati nella AWS Lambda Developer Guide.

Ecco un esempio di modello AWS SAM configurato:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Metadata:
      BuildMethod: rust-cargolambda
      BuildProperties: function_a
    Properties:
      CodeUri: ./rust_app
      Handler: bootstrap
      Runtime: provided.al2
```

```
...
```

Passaggio 2: utilizzare la funzione Lambda AWS SAMCLI con Rust

Usa qualsiasi AWS SAMCLI comando con il tuo AWS SAM modello. Per ulteriori informazioni, consulta AWS SAM CLI.

Esempi

Esempio di Hello World

In questo esempio, creiamo l'applicazione Hello World di esempio utilizzando Rust come runtime.

Innanzitutto, inizializziamo una nuova applicazione serverless utilizzando. `sam init` Durante il flusso interattivo, selezioniamo l'applicazione Hello World e scegliamo il runtime Rust.

```
$ sam init
...
Which template source would you like to use?
        1 - AWS Quick Start Templates
        2 - Custom Template Location
Choice: 1

Choose an AWS Quick Start application template
        1 - Hello World Example
        2 - Multi-step workflow
        3 - Serverless API
        ...
Template: 1

Use the most popular runtime and package type? (Python and zip) [y/N]: ENTER

Which runtime would you like to use?
        1 - dotnet8
        2 - dotnet6
        3 - go (provided.al2)
        ...
        18 - python3.11
        19 - python3.10
        20 - ruby3.3
        21 - ruby3.2
        22 - rust (provided.al2)
        23 - rust (provided.al2023)
```

```
Runtime: 22

Based on your selections, the only Package type available is Zip.
We will proceed to selecting the Package type as Zip.

Based on your selections, the only dependency manager available is cargo.
We will proceed copying the template using cargo.

Would you like to enable X-Ray tracing on the function(s) in your application?  [y/
N]: ENTER

Would you like to enable monitoring using CloudWatch Application Insights?
For more info, please view https://docs.aws.amazon.com/AmazonCloudWatch/latest/
monitoring/cloudwatch-application-insights.html [y/N]: ENTER

Project name [sam-app]: hello-rust

    -----------------------
    Generating application:
    -----------------------
    Name: hello-rust
    Runtime: rust (provided.al2)
    Architectures: x86_64
    Dependency Manager: cargo
    Application Template: hello-world
    Output Directory: .
    Configuration file: hello-rust/samconfig.toml

    Next steps can be found in the README file at hello-rust/README.md


Commands you can use next
=========================
[*] Create pipeline: cd hello-rust && sam pipeline init --bootstrap
[*] Validate SAM template: cd hello-rust && sam validate
[*] Test Function in the Cloud: cd hello-rust && sam sync --stack-name {stack-name} --
watch
```

Di seguito è riportata la struttura della nostra applicazione Hello World:

```
hello-rust
### README.md
### events
```

```
#   ### event.json
### rust_app
#   ### Cargo.toml
#   ### src
#       ### main.rs
### samconfig.toml
### template.yaml
```

Nel nostro AWS SAM modello, la nostra Rust funzione è definita come segue:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Metadata:
      BuildMethod: rust-cargolambda
    Properties:
      CodeUri: ./rust_app
      Handler: bootstrap
      Runtime: provided.al2
      Architectures:
        - x86_64
      Events:
        HelloWorld:
          Type: Api
            Path: /hello
            Method: get
```

Successivamente, `sam build` eseguiamo la creazione della nostra applicazione e ci prepariamo per la distribuzione. AWS SAMCLICrea una `.aws-sam` directory e vi organizza i nostri artefatti di compilazione. La nostra funzione è creata utilizzando Cargo Lambda e archiviata come binario eseguibile in. `.aws-sam/build/HelloWorldFunction/bootstrap`

> **Note**
>
> Se prevedi di eseguire il sam local invoke comando in macOS, devi creare funzioni diverse prima di richiamarlo. Per fare ciò, usa il seguente comando:
>
> - SAM_BUILD_MODE=debug sam build

Questo comando è necessario solo se verranno eseguiti test locali. Questo non è consigliato quando si crea per la distribuzione.

```
hello-rust$ sam build
Starting Build use cache
Build method "rust-cargolambda" is a beta feature.
Please confirm if you would like to proceed
You can also enable this beta feature with "sam build --beta-features". [y/N]: y

Experimental features are enabled for this session.
Visit the docs page to learn more about the AWS Beta terms https://aws.amazon.com/
service-terms/.

Cache is invalid, running build and copying resources for following functions
 (HelloWorldFunction)
Building codeuri: /Users/.../hello-rust/rust_app runtime: provided.al2 metadata:
 {'BuildMethod': 'rust-cargolambda'} architecture: x86_64 functions: HelloWorldFunction
Running RustCargoLambdaBuilder:CargoLambdaBuild
Running RustCargoLambdaBuilder:RustCopyAndRename

Build Succeeded

Built Artifacts  : .aws-sam/build
Built Template   : .aws-sam/build/template.yaml

Commands you can use next
=========================
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name {{stack-name}} --watch
[*] Deploy: sam deploy --guided
```

Successivamente, distribuiamo la nostra applicazione utilizzando`sam deploy --guided`.

```
hello-rust$ sam deploy --guided

Configuring SAM deploy
======================

        Looking for config file [samconfig.toml] :  Found
```

```
        Reading default arguments  :  Success

        Setting default arguments for 'sam deploy'
        =========================================
        Stack Name [hello-rust]: ENTER
        AWS Region [us-west-2]: ENTER
        #Shows you resources changes to be deployed and require a 'Y' to initiate
 deploy
        Confirm changes before deploy [Y/n]: ENTER
        #SAM needs permission to be able to create roles to connect to the resources in
 your template
        Allow SAM CLI IAM role creation [Y/n]: ENTER
        #Preserves the state of previously provisioned resources when an operation
 fails
        Disable rollback [y/N]: ENTER
        HelloWorldFunction may not have authorization defined, Is this okay? [y/N]: y
        Save arguments to configuration file [Y/n]: ENTER
        SAM configuration file [samconfig.toml]: ENTER
        SAM configuration environment [default]: ENTER

        Looking for resources needed for deployment:

        ...

        Uploading to hello-rust/56ba6585d80577dd82a7eaaee5945c0b  817973 / 817973
 (100.00%)

        Deploying with following values
        ===============================
        Stack name                   : hello-rust
        Region                       : us-west-2
        Confirm changeset            : True
        Disable rollback             : False
        Deployment s3 bucket         : aws-sam-cli-managed-default-samclisam-s3-demo-
bucket-1a4x26zbcdkqr
        Capabilities                 : ["CAPABILITY_IAM"]
        Parameter overrides          : {}
        Signing Profiles             : {}

Initiating deployment
=====================

        Uploading to hello-rust/a4fc54cb6ab75dd0129e4cdb564b5e89.template  1239 / 1239
 (100.00%)
```

```
Waiting for changeset to be created..

CloudFormation stack changeset
-----------------------------------------------------------------------------------------------
Operation                   LogicalResourceId          ResourceType
 Replacement
-----------------------------------------------------------------------------------------------
+ Add                       HelloWorldFunctionHelloW   AWS::Lambda::Permission    N/A

                            orldPermissionProd

...
-----------------------------------------------------------------------------------------------

Changeset created successfully. arn:aws:cloudformation:us-
west-2:012345678910:changeSet/samcli-deploy1681427201/
f0ef1563-5ab6-4b07-9361-864ca3de6ad6


Previewing CloudFormation changeset before deployment
=====================================================
Deploy this changeset? [y/N]: y

2023-04-13 13:07:17 - Waiting for stack create/update to complete

CloudFormation events from stack operations (refresh every 5.0 seconds)
-----------------------------------------------------------------------------------------------
ResourceStatus             ResourceType               LogicalResourceId
 ResourceStatusReason
-----------------------------------------------------------------------------------------------
CREATE_IN_PROGRESS         AWS::IAM::Role             HelloWorldFunctionRole     -

CREATE_IN_PROGRESS         AWS::IAM::Role             HelloWorldFunctionRole
 Resource creation
...
-----------------------------------------------------------------------------------------------

CloudFormation outputs from deployed stack
-----------------------------------------------------------------------------------------------
Outputs

-----------------------------------------------------------------------------------------------
```

```
Key                 HelloWorldFunctionIamRole

Description         Implicit IAM Role created for Hello World function

Value               arn:aws:iam::012345678910:role/hello-rust-
HelloWorldFunctionRole-10II2P13AUDUY

Key                 HelloWorldApi

Description         API Gateway endpoint URL for Prod stage for Hello World function

Value               https://ggdxec9le9.execute-api.us-west-2.amazonaws.com/Prod/hello/


Key                 HelloWorldFunction

Description         Hello World Lambda Function ARN

Value               arn:aws:lambda:us-west-2:012345678910:function:hello-rust-
HelloWorldFunction-
yk4HzGzYeZBj

-----------------------
```

```
      BuildMethod: rust-cargolambda # More info about Cargo Lambda: https://github.com/
cargo-lambda/cargo-lambda
    Properties:
      CodeUri: ./rust_app   # Points to dir of Cargo.toml
      Handler: bootstrap    # Do not change, as this is the default executable name
 produced by Cargo Lambda
      Runtime: provided.al2
      Architectures:
        - arm64
...
```

Poiché `arm64` in questo esempio abbiamo modificato la nostra architettura dalla `x86_64` a alla, `sam build` eseguiamo l'aggiornamento dei nostri artefatti di build. Quindi eseguiamo per `sam local invoke` richiamare localmente la nostra funzione.

```
hello-rust$ sam local invoke
Invoking bootstrap (provided.al2)
Local image was not found.
Removing rapid images for repo public.ecr.aws/sam/emulation-provided.al2
Building
 image.....................................................................................
Using local image: public.ecr.aws/lambda/provided:al2-rapid-arm64.

Mounting /Users/.../hello-rust/.aws-sam/build/HelloWorldFunction as /var/
task:ro,delegated, inside runtime container
START RequestId: fbc55e6e-0068-45f9-9f01-8e2276597fc6 Version: $LATEST
{"statusCode":200,"body":"Hello World!"}END RequestId:
 fbc55e6e-0068-45f9-9f01-8e2276597fc6
REPORT RequestId: fbc55e6e-0068-45f9-9f01-8e2276597fc6  Init Duration: 0.68 ms
 Duration: 130.63 ms     Billed Duration: 131 ms     Memory Size: 128 MB     Max Memory
 Used: 128 MB
```

Progetto a singola funzione Lambda

Ecco un esempio di applicazione serverless contenente una funzione Rust Lambda.

Struttura delle cartelle del progetto:

```
.
### Cargo.lock
### Cargo.toml
### src
#   ### main.rs
```

```
### template.yaml
```

AWS SAM modello:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Metadata:
      BuildMethod: rust-cargolambda
    Properties:
      CodeUri: ./
      Handler: bootstrap
      Runtime: provided.al2
...
```

Progetto con più funzioni Lambda

Ecco un esempio di applicazione serverless contenente più funzioni Rust Lambda.

Struttura delle cartelle del progetto:

```
.
### Cargo.lock
### Cargo.toml
### src
#   ### function_a.rs
#   ### function_b.rs
### template.yaml
```

AWS SAM modello:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  FunctionA:
    Type: AWS::Serverless::Function
    Metadata:
      BuildMethod: rust-cargolambda
```

```
      BuildProperties:
        Binary: function_a
    Properties:
      CodeUri: ./
      Handler: bootstrap
      Runtime: provided.al2
  FunctionB:
    Type: AWS::Serverless::Function
    Metadata:
      BuildMethod: rust-cargolambda
      BuildProperties:
        Binary: function_b
    Properties:
      CodeUri: ./
      Handler: bootstrap
      Runtime: provided.al2
```

`Cargo.toml` file

```
[package]
name = "test-handler"
version = "0.1.0"
edition = "2021"

[dependencies]
lambda_runtime = "0.6.0"
serde = "1.0.136"
tokio = { version = "1", features = ["macros"] }
tracing = { version = "0.1", features = ["log"] }
tracing-subscriber = { version = "0.3", default-features = false, features = ["fmt"] }

[[bin]]
name = "function_a"
path = "src/function_a.rs"

[[bin]]
name = "function_b"
path = "src/function_b.rs"
```

Creazione di funzioni Lambda in Python con in uv AWS SAM

Questa funzionalità è disponibile in anteprima AWS SAM ed è soggetta a modifiche.

Usa la AWS Serverless Application Model Command Line Interface (AWS SAMCLI) conuv, un programma di installazione e risoluzione di pacchetti Python veloce, per creare le tue funzioni Python. AWS Lambda

Argomenti

- Prerequisiti
- Configurazione AWS SAM per l'uso con le funzioni Python Lambda e uv
- Esempi

Prerequisiti

Python

Per installare Python, consulta Download Python nel sito Web Python.

uv

AWS SAMCLIRichiede l'installazione di uv, un programma di installazione e risoluzione di pacchetti Python estremamente veloce. Per le istruzioni di installazione, vedete Installazione nella documentazione. uv

Attiva la funzionalità AWS SAMCLI beta

Poiché questa funzionalità è disponibile in anteprima, devi attivarla utilizzando uno dei seguenti metodi:

1. Usa la variabile di ambiente:`SAM_CLI_BETA_PYTHON_UV=1`.
2. Aggiungi il codice seguente al file `samconfig.toml`:

```
[default.build.parameters]
beta_features = true
[default.sync.parameters]
beta_features = true
```

3. Usa l'`--beta-features`opzione quando usi un AWS SAMCLI comando supportato. Esempio:

```
$ sam build --beta-features
```

4. Scegli l'opzione y quando AWS SAMCLI ti viene richiesto di aderire. Di seguito è riportato un esempio:

```
$ sam build
Starting Build use cache
Build method "python-uv" is a beta feature.
Please confirm if you would like to proceed
You can also enable this beta feature with "sam build --beta-features". [y/N]: y
```

Configurazione AWS SAM per l'uso con le funzioni Python Lambda e uv

Passaggio 1: configura il tuo AWS SAM modello

Configura il tuo AWS SAM modello con quanto segue:

- BuildMethod – `python-uv`.
- CodeUri— percorso della directory del codice della funzione contenente `pyproject.toml` o `requirements.txt`.
- Handler: il tuo gestore di funzioni (ad es.). `app.lambda_handler`
- Runtime — Versione runtime di Python (ad es.). `python3.12`

Ecco un esempio di AWS SAM modello configurato:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: ./my_function
      Handler: app.lambda_handler
      Runtime: python3.12
    Metadata:
      BuildMethod: python-uv
...
```

Esempi

Esempio di Hello World

In questo esempio, creiamo un'applicazione Hello World di esempio usando Python uv come gestore di pacchetti.

uvpuò usare uno o l'altro `pyproject.toml` o `requirements.txt` leggere le dipendenze. Se vengono forniti entrambi, `sam build` leggerà da `requirements.txt` per verificare le dipendenze.

Di seguito è riportata la struttura della nostra applicazione Hello World:

```
hello-python-uv
### README.md
### events
#   ### event.json
### hello_world
#   ### __init__.py
#   ### app.py
#   ### pyproject.toml
### samconfig.toml
### template.yaml
```

`pyproject.toml` file

```
[project]
name = "my-function"
version = "0.1.0"
requires-python = ">=3.12"
dependencies = [
    "requests>=2.31.0",
    "boto3>=1.28.0",
]
```

Nel nostro AWS SAM modello, la nostra funzione Python è definita come segue:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
```

```
    Properties:
      CodeUri: hello_world/
      Handler: app.lambda_handler
      Runtime: python3.12
      Architectures:
        - x86_64
    Metadata:
      BuildMethod: python-uv
```

Successivamente, `sam build` eseguiamo la creazione della nostra applicazione e la preparazione per la distribuzione. AWS SAMCLICrea una `.aws-sam` directory e vi organizza i nostri artefatti di compilazione. Le nostre dipendenze dalle funzioni vengono installate utilizzando uv e archiviate in. `.aws-sam/build/HelloWorldFunction/`

```
hello-python-uv$ sam build
Starting Build use cache
Build method "python-uv" is a beta feature.
Please confirm if you would like to proceed
You can also enable this beta feature with "sam build --beta-features". [y/N]: y

Experimental features are enabled for this session.
Visit the docs page to learn more about the AWS Beta terms https://aws.amazon.com/
service-terms/.

Cache is invalid, running build and copying resources for following functions
 (HelloWorldFunction)
Building codeuri: /Users/.../hello-python-uv/hello_world runtime: python3.12 metadata:
 {'BuildMethod': 'python-uv'} architecture: x86_64 functions: HelloWorldFunction
Running PythonUvBuilder:UvBuild
Running PythonUvBuilder:CopySource

Build Succeeded

Built Artifacts  : .aws-sam/build
Built Template   : .aws-sam/build/template.yaml

Commands you can use next
=========================
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name {{stack-name}} --watch
[*] Deploy: sam deploy --guided
```

> **Note**
>
> Il metodo di `python-uv` compilazione è configurato per funzione nella `Metadata` sezione. Ogni funzione del modello può utilizzare un metodo di compilazione diverso, che consente di combinare funzioni uv `pip` basate con funzioni basate nello stesso AWS SAM modello. Se non viene specificato alcun metodo di compilazione, `pip` viene utilizzato per impostazione predefinita.

Creazione di funzioni Lambda con runtime personalizzati in AWS SAM

Puoi usare il `sam build` comando per creare i runtime personalizzati richiesti per la tua funzione Lambda. Dichiari che la tua funzione Lambda utilizzi un runtime personalizzato `Runtime: provided` specificando la funzione.

Per creare un runtime personalizzato, dichiarate l'attributo `Metadata` resource con una voce. `BuildMethod: makefile` Fornisci un makefile personalizzato, in cui dichiari un obiettivo di compilazione del modulo `build-`*`function-logical-id`* che contiene i comandi di build per il tuo runtime. Il makefile è responsabile della compilazione del runtime personalizzato, se necessario, e della copia degli elementi di compilazione nella posizione corretta richiesta per le fasi successive del flusso di lavoro. La posizione del makefile è specificata dalla `CodeUri` proprietà della risorsa della funzione e deve essere denominata. `Makefile`

Esempi

Esempio 1: runtime personalizzato per una funzione scritta in Rust

> **Note**
>
> Consigliamo di creare funzioni Lambda con. Cargo Lambda Per ulteriori informazioni, consulta Creazione di funzioni Rust Lambda con in Cargo Lambda AWS SAM.

Il AWS SAM modello seguente dichiara una funzione che utilizza un runtime personalizzato per una funzione Lambda scritta in Rust e `sam build` indica di eseguire i comandi per `build-HelloRustFunction` l'obiettivo di compilazione.

```
Resources:
```

```
  HelloRustFunction:
    Type: AWS::Serverless::Function
    Properties:
      FunctionName: HelloRust
      Handler: bootstrap.is.real.handler
      Runtime: provided
      MemorySize: 512
      CodeUri: .
    Metadata:
      BuildMethod: makefile
```

Il seguente makefile contiene l'obiettivo di compilazione e i comandi che verranno eseguiti. Nota che la `CodeUri` proprietà è impostata su`.`, quindi il makefile deve trovarsi nella directory principale del progetto (ovvero, la stessa directory del file AWS SAM modello dell'applicazione). Il nome del file deve essere. `Makefile`

```
build-HelloRustFunction:
 cargo build --release --target x86_64-unknown-linux-musl
 cp ./target/x86_64-unknown-linux-musl/release/bootstrap $(ARTIFACTS_DIR)
```

Per ulteriori informazioni sulla configurazione dell'ambiente di sviluppo per l'esecuzione del `cargo build` comando precedente`makefile`, consultate il AWS Lambda post sul blog di Rust Runtime for.

Esempio 2: Makefile builder per Python3.12 (alternativa all'utilizzo del builder in bundle)

Potresti voler usare una libreria o un modulo che non è incluso in un builder in bundle. Questo esempio mostra un AWS SAM modello per un runtime Python3.12 con un generatore di makefile.

```
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: hello_world/
      Handler: app.lambda_handler
      Runtime: python3.12
    Metadata:
      BuildMethod: makefile
```

Il seguente makefile contiene l'obiettivo di compilazione e i comandi che verranno eseguiti. Nota che la `CodeUri` proprietà è impostata su`hello_world`, quindi il makefile deve trovarsi nella radice della `hello_world` sottodirectory e il nome del file deve essere. `Makefile`

```
build-HelloWorldFunction:
 cp *.py $(ARTIFACTS_DIR)
 cp requirements.txt $(ARTIFACTS_DIR)
 python -m pip install -r requirements.txt -t $(ARTIFACTS_DIR)
 rm -rf $(ARTIFACTS_DIR)/bin
```

Creazione di livelli Lambda in AWS SAM

Puoi usarlo AWS SAM per creare layer Lambda personalizzati. I layer Lambda consentono di estrarre codice da una funzione Lambda che può quindi essere riutilizzato in diverse funzioni Lambda. La creazione di soli livelli Lambda (anziché la creazione dell'intera applicazione) può apportare vantaggi in alcuni modi. Può aiutarvi a ridurre le dimensioni dei pacchetti di distribuzione, separare la logica delle funzioni principali dalle dipendenze e consentire di condividere le dipendenze tra più funzioni. Per informazioni sui livelli, consulta i layer AWS Lambda nella AWS Lambda Developer Guide.

Come creare un layer Lambda in AWS SAM

> **Note**
>
> Prima di poter creare un layer Lambda, devi prima scrivere un layer Lambda nel tuo modello. AWS SAM Per informazioni ed esempi su come eseguire questa operazione, consulta. Aumenta l'efficienza utilizzando i livelli Lambda con AWS SAM

Per creare un layer personalizzato, dichiaratelo nel vostro file modello AWS Serverless Application Model (AWS SAM) e includete una sezione dedicata agli attributi `Metadata` delle risorse con una `BuildMethod` voce. I valori validi per `BuildMethod` sono identificatori per un AWS Lambda runtime, o. `makefile` Includi una `BuildArchitecture` voce per specificare le architetture del set di istruzioni supportate dal livello. I valori validi per `BuildArchitecture` le architetture dei set di istruzioni Lambda.

Se lo specificate`makefile`, fornite il makefile personalizzato, in cui dichiarate un obiettivo di compilazione del modulo `build-`*`layer-logical-id`* che contiene i comandi di compilazione per il livello. Il makefile è responsabile della compilazione del layer, se necessario, e della copia degli artefatti di compilazione nella posizione corretta richiesta per le fasi successive del flusso di lavoro. La posizione del makefile è specificata dalla `ContentUri` proprietà della risorsa del livello e deve essere denominata. `Makefile`

> **Note**
>
> Quando create un layer personalizzato, la ricerca del codice del layer AWS Lambda dipende dalle variabili di ambiente. I runtime Lambda includono i percorsi nella `/opt` directory in cui viene copiato il codice del layer. La struttura delle cartelle degli artefatti di compilazione del progetto deve corrispondere alla struttura delle cartelle prevista per il runtime in modo da poter trovare il codice del layer personalizzato.
> Ad esempio, per Python puoi inserire il codice nella `python/` sottodirectory. Per NodeJS, puoi inserire il codice nella sottodirectory. `nodejs/node_modules/`
> Per ulteriori informazioni, consulta Inclusione delle dipendenze delle librerie in un livello nella Guida per gli sviluppatori.AWS Lambda

Di seguito è riportato un esempio di sezione relativa agli attributi `Metadata` delle risorse.

```
Metadata:
  BuildMethod: python3.12
  BuildArchitecture: arm64
```

> **Note**
>
> Se non includi la sezione degli attributi `Metadata` delle risorse, AWS SAM non crea il layer. Copia invece gli artefatti di compilazione dalla posizione specificata nella `CodeUri` proprietà della risorsa del livello. Per ulteriori informazioni, vedete la ContentUriproprietà del tipo di `AWS::Serverless::LayerVersion` risorsa.

Quando includete la sezione degli attributi `Metadata` delle risorse, potete usare il `sam build` comando per creare il livello, sia come oggetto indipendente che come dipendenza di una AWS Lambda funzione.

- Come oggetto indipendente. Potresti voler creare solo l'oggetto layer, ad esempio quando stai testando localmente una modifica al codice del layer e non hai bisogno di creare l'intera applicazione. Per creare il livello in modo indipendente, specificate la risorsa del livello con il `sam build` *`layer-logical-id`* comando.
- Come dipendenza di una funzione Lambda. Quando includi l'ID logico di un livello nella `Layers` proprietà di una funzione Lambda nello stesso file AWS SAM modello, il livello è una dipendenza

di quella funzione Lambda. Quando quel livello include anche una sezione di attributi di `Metadata` risorsa con una `BuildMethod` voce, si crea il livello creando l'intera applicazione con il `sam build` comando o specificando la risorsa della funzione con il comando. `sam build` *`function-logical-id`*

Esempi

Esempio di modello 1: crea un livello rispetto all'ambiente di runtime Python 3.12

Il seguente AWS SAM modello di esempio crea un livello rispetto all'ambiente di runtime Python 3.12.

```
Resources:
  MyLayer:
    Type: AWS::Serverless::LayerVersion
    Properties:
      ContentUri: my_layer
      CompatibleRuntimes:
        - python3.12
    Metadata:
      BuildMethod: python3.12   # Required to have AWS SAM build this layer
```

Esempio di modello 2: crea un livello utilizzando un makefile personalizzato

Il seguente AWS SAM modello di esempio utilizza un modello personalizzato `makefile` per creare il layer.

```
Resources:
  MyLayer:
    Type: AWS::Serverless::LayerVersion
    Properties:
      ContentUri: my_layer
      CompatibleRuntimes:
        - python3.12
    Metadata:
      BuildMethod: makefile
```

Quanto segue `makefile` contiene l'obiettivo di compilazione e i comandi che verranno eseguiti. Nota che la `ContentUri` proprietà è impostata sumy`_layer`, quindi il makefile deve trovarsi nella radice della `my_layer` sottodirectory e il nome del file deve essere. `Makefile` Nota anche che gli artefatti di compilazione vengono copiati nella `python/` sottodirectory in modo che sia AWS Lambda possibile trovare il codice del livello.

```
build-MyLayer:
  mkdir -p "$(ARTIFACTS_DIR)/python"
  cp *.py "$(ARTIFACTS_DIR)/python"
  python -m pip install -r requirements.txt -t "$(ARTIFACTS_DIR)/python"
```

> **Note**
>
> Quando `makefile` viene chiamato, viene attivato il target appropriato e gli artefatti devono essere copiati nella variabile ambientale esposta. `$ARTIFACTS_DIR` Per ulteriori informazioni, fare riferimento a in. aws-lambda-builders GitHub

Esempi di comandi sam build

I seguenti `sam build` comandi creano livelli che includono le sezioni degli attributi `Metadata` delle risorse.

```
# Build the 'layer-logical-id' resource independently
$ sam build layer-logical-id

# Build the 'function-logical-id' resource and layers that this function depends on
$ sam build function-logical-id

# Build the entire application, including the layers that any function depends on
$ sam build
```

Testa la tua applicazione serverless con AWS SAM

Dopo aver scritto e creato l'applicazione, sarete pronti a testarla per verificarne il corretto funzionamento. Con l'interfaccia a riga di AWS SAM comando (CLI), puoi testare localmente la tua applicazione serverless prima di caricarla sul Cloud. AWS Il test dell'applicazione consente di confermare la funzionalità, l'affidabilità e le prestazioni dell'applicazione, identificando al contempo i problemi (bug) che dovranno essere risolti.

Questa sezione fornisce indicazioni sulle pratiche comuni da seguire per testare l'applicazione. Gli argomenti di questa sezione si concentrano principalmente sui test locali che è possibile eseguire prima della distribuzione nel AWS cloud. Il test prima della distribuzione consente di identificare i problemi in modo proattivo, riducendo i costi non necessari associati ai problemi di implementazione. Ogni argomento di questa sezione descrive un test che è possibile eseguire, dice a yoxi include href? u i vantaggi del suo utilizzo e include esempi che mostrano come eseguire il test. Dopo aver testato l'applicazione, sarete pronti a eseguire il debug di eventuali problemi riscontrati.

Argomenti

- Introduzione ai test con il sam local comando
- Richiama localmente le funzioni Lambda con AWS SAM
- API Gateway eseguito localmente con AWS SAM
- Introduzione al cloud testing con sam remote test-event
- Introduzione ai test nel cloud con sam remote invoke
- Automatizza i test di integrazione locali con AWS SAM
- Genera payload di eventi di esempio con AWS SAM
- Test e debug di funzioni durevoli

Introduzione ai test con il sam local comando

Utilizzate il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam local` per testare localmente le vostre applicazioni serverless.

Per un'introduzione a AWS SAMCLI, vedereChe cos'è il? AWS SAMCLI.

Prerequisiti

Per utilizzarlo`sam local`, AWS SAMCLI installate il compilando quanto segue:

- AWS SAM prerequisiti.
- Installa il AWS SAMCLI.

Prima dell'uso`sam local`, si consiglia una conoscenza di base di quanto segue:

- Configurazione del AWS SAMCLI.
- Crea la tua candidatura in AWS SAM.
- Introduzione alla costruzione con AWS SAM.
- Introduzione alla distribuzione con AWS SAM.

Utilizzo del sam local comando

Utilizzate il `sam local` comando con uno qualsiasi dei suoi sottocomandi per eseguire diversi tipi di test locali per l'applicazione.

```
$ sam local <subcommand>
```

Per ulteriori informazioni su ciascun sottocomando, vedere quanto segue:

- Introduzione a sam local generate-event— Genera Servizio AWS eventi per test locali.
- Introduzione a sam local invoke— Avvia una chiamata una tantum di una funzione localmente. AWS Lambda
- Introduzione a sam local start-api— Esegui le tue funzioni Lambda utilizzando un server HTTP locale.
- Introduzione a sam local start-lambda— Esegui le tue funzioni Lambda utilizzando un server HTTP locale da utilizzare con o. AWS CLI SDKs

Introduzione ai test con sam local generate-event

Utilizzate il AWS Serverless Application Model `sam local generate-event` sottocomando Command Line Interface (AWS SAMCLI) per generare esempi di payload di eventi supportati. Servizi AWS Potete quindi modificare e passare questi eventi alle risorse locali per i test.

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

- Per un elenco delle opzioni di `sam local generate-event` comando, vedere sam local generate-event.

Un evento è un oggetto JSON che viene generato quando un Servizio AWS esegue un'azione o un'attività. Questi eventi contengono informazioni specifiche, come i dati elaborati o il timestamp dell'evento. La maggior parte Servizi AWS genera eventi e gli eventi di ogni servizio sono formattati in modo univoco per il relativo servizio.

Gli eventi generati da un servizio vengono passati ad altri servizi come fonte di eventi. Ad esempio, un articolo inserito in un bucket Amazon Simple Storage Service (Amazon S3) può generare un evento. Questo evento può quindi essere utilizzato come origine dell'evento per una AWS Lambda funzione che elabora ulteriormente i dati.

Gli eventi generati con `sam local generate-event` sono formattati nella stessa struttura degli eventi effettivi creati dal AWS servizio. È possibile modificare il contenuto di questi eventi e utilizzarli per testare le risorse dell'applicazione.

Prerequisiti

Per utilizzarlo `sam local generate-event`, AWS SAMCLI installate il completando quanto segue:

- AWS SAM prerequisiti.
- Installa il AWS SAMCLI.

Prima dell'uso `sam local generate-event`, si consiglia una conoscenza di base di quanto segue:

- Configurazione del AWS SAMCLI.
- Crea la tua candidatura in AWS SAM.
- Introduzione alla costruzione con AWS SAM.
- Introduzione alla distribuzione con AWS SAM.

Genera eventi di esempio

Usa il AWS SAMCLI `sam local generate-event` sottocomando per generare eventi per i supporti Servizi AWS.

Per visualizzare un elenco di quelli supportati Servizi AWS

1. Esegui il seguente codice:

```
$ sam local generate-event
```

2. Servizi AWS Verrà visualizzato l'elenco dei supportati. Di seguito è riportato un esempio:

```
$ sam local generate-event
...
Commands:
  alb
  alexa-skills-kit
  alexa-smart-home
  apigateway
  appsync
  batch
  cloudformation
  ...
```

Per generare un evento locale

1. Esegui `sam local generate-event` e fornisci il nome del servizio supportato. Verrà visualizzato un elenco di tipi di eventi che è possibile generare. Di seguito è riportato un esempio:

```
$ sam local generate-event s3

Usage: sam local generate-event s3 [OPTIONS] COMMAND [ARGS]...

Options:
  -h, --help  Show this message and exit.

Commands:
  batch-invocation  Generates an Amazon S3 Batch Operations Invocation Event
  delete            Generates an Amazon S3 Delete Event
  put               Generates an Amazon S3 Put Event
```

2. Per generare l'evento di esempio, eseguisam `local generate-event`, fornendo il servizio e il tipo di evento.

```
$ sam local generate-event <service> <event>
```

Di seguito è riportato un esempio:

```
$ sam local generate-event s3 put
{
  "Records": [
    {
      "eventVersion": "2.0",
      "eventSource": "aws:s3",
      "awsRegion": "us-east-1",
      "eventTime": "1970-01-01T00:00:00.000Z",
      "eventName": "ObjectCreated:Put",
      "userIdentity": {
        "principalId": "EXAMPLE"
      },
      "requestParameters": {
        "sourceIPAddress": "127.0.0.1"
      },
      "responseElements": {
        "x-amz-request-id": "EXAMPLE123456789",
        "x-amz-id-2": "EXAMPLE123/5678abcdefghijklambdaisawesome/
mnopqrstuvwxyzABCDEFGH"
      },
      "s3": {
        "s3SchemaVersion": "1.0",
        "configurationId": "testConfigRule",
        "bucket": {
          "name": "sam-s3-demo-bucket",
          "ownerIdentity": {
            "principalId": "EXAMPLE"
          },
          "arn": "arn:aws:s3:::sam-s3-demo-bucket"
        },
        "object": {
          "key": "test/key",
          "size": 1024,
          "eTag": "0123456789abcdef0123456789abcdef",
          "sequencer": "0A1B2C3D4E5F678901"
        }
      }
    }
```

```
  ]
}
```

Questi eventi di esempio contengono valori segnaposto. È possibile modificare questi valori in modo che facciano riferimento alle risorse effettive dell'applicazione o ai valori per facilitare i test locali.

Per modificare un evento di esempio

1. È possibile modificare gli eventi di esempio al prompt dei comandi. Per visualizzare le opzioni disponibili, esegui quanto segue:

```
$ sam local generate-event <service> <event> --help
```

   Di seguito è riportato un esempio:

```
$ sam local generate-event s3 put --help

Usage: sam local generate-event s3 put [OPTIONS]

Options:
  --region TEXT       Specify the region name you'd like, otherwise the
                      default = us-east-1
  --partition TEXT    Specify the partition name you'd like, otherwise the
                      default = aws
  --bucket TEXT       Specify the bucket name you'd like, otherwise the
                      default = example-bucket
  --key TEXT          Specify the key name you'd like, otherwise the default =
                      test/key
  --debug             Turn on debug logging to print debug message generated
                      by AWS SAM CLI and display timestamps.
  --config-file TEXT  Configuration file containing default parameter values.
                      [default: samconfig.toml]
  --config-env TEXT   Environment name specifying default parameter values in
                      the configuration file.  [default: default]
  -h, --help          Show this message and exit.
```

2. Utilizzate una di queste opzioni al prompt dei comandi per modificare il payload dell'evento di esempio. Di seguito è riportato un esempio:

```
$ sam local generate-event s3 put--bucket sam-s3-demo-bucket
```

```
{
  "Records": [
    {
      "eventVersion": "2.0",
      "eventSource": "aws:s3",
      "awsRegion": "us-east-1",
      "eventTime": "1970-01-01T00:00:00.000Z",
      "eventName": "ObjectCreated:Put",
      "userIdentity": {
        "principalId": "EXAMPLE"
      },
      "requestParameters": {
        "sourceIPAddress": "127.0.0.1"
      },
      "responseElements": {
        "x-amz-request-id": "EXAMPLE123456789",
        "x-amz-id-2": "EXAMPLE123/5678abcdefghijklambdaisawesome/
mnopqrstuvwxyzABCDEFGH"
      },
      "s3": {
        "s3SchemaVersion": "1.0",
        "configurationId": "testConfigRule",
        "bucket": {
          "name": "sam-s3-demo-bucket",
          "ownerIdentity": {
            "principalId": "EXAMPLE"
          },
          "arn": "arn:aws:s3:::sam-s3-demo-bucket"
        },
        "object": {
          "key": "test/key",
          "size": 1024,
          "eTag": "0123456789abcdef0123456789abcdef",
          "sequencer": "0A1B2C3D4E5F678901"
        }
      }
    }
  ]
}
```

Usa gli eventi generati per i test locali

Salva gli eventi generati localmente e usa altri `sam local` sottocomandi per eseguire i test.

Per salvare gli eventi generati localmente

- Esegui il seguente codice:

```
$ sam local generate-event <service> <event> <event-option> > <filename.json>
```

  Di seguito è riportato un esempio di evento salvato come `s3.json` file nella `events` cartella del nostro progetto.

```
sam-app$ sam local generate-event s3 put --bucket amzn-s3-demo-bucket > events/
s3.json
```

Per utilizzare un evento generato per i test locali

- Passa l'evento con altri `sam local` sottocomandi utilizzando l'`--event`opzione.

  Di seguito è riportato un esempio di utilizzo dell'`s3.json`evento per richiamare la nostra funzione Lambda localmente:

```
sam-app$ sam local invoke --event events/s3.json S3JsonLoggerFunction

Invoking src/handlers/s3-json-logger.s3JsonLoggerHandler (nodejs18.x)
Local image is up-to-date
Using local image: public.ecr.aws/lambda/nodejs:18-rapid-x86_64.

Mounting /Users/.../sam-app/.aws-sam/build/S3JsonLoggerFunction as /var/
task:ro,delegated, inside runtime container
START RequestId: f4f45b6d-2ec6-4235-bc7b-495ec2ae0128 Version: $LATEST
END RequestId: f4f45b6d-2ec6-4235-bc7b-495ec2ae0128
REPORT RequestId: f4f45b6d-2ec6-4235-bc7b-495ec2ae0128  Init Duration: 1.23 ms
 Duration: 9371.93 ms      Billed Duration: 9372 ms        Memory Size: 128 MB
 Max Memory Used: 128 MB
```

Ulteriori informazioni

Per un elenco di tutte le `sam local generate-event` opzioni, consultasam local generate-event.

Per una dimostrazione dell'utilizzo`sam local`, vedi AWS SAM per lo sviluppo locale. Test Cloud AWS delle risorse provenienti da ambienti di sviluppo locali nella serie Serverless Land Sessions with SAM on YouTube.

Introduzione ai test con sam local invoke

Utilizzate il AWS Serverless Application Model `sam local invoke` sottocomando Command Line Interface (AWS SAMCLI) per avviare una chiamata una tantum di una funzione localmente. AWS Lambda

- Per un'introduzione a, vedere AWS SAMCLI Che cos'è il? AWS SAMCLI
- Per un elenco delle opzioni di `sam local invoke` comando, vederesam local invoke.
- Per un esempio di utilizzo `sam local invoke` durante un tipico flusso di lavoro di sviluppo, vediPassaggio 7: (Facoltativo) Testa l'applicazione localmente.

> **Note**
>
> `sam local invoke`supporta funzioni durevoli con funzionalità di checkpoint e replay automatici. Quando si richiamano funzioni durevoli localmente, lo stato di esecuzione viene gestito automaticamente.

Prerequisiti

Per utilizzarlo`sam local invoke`, installa il AWS SAMCLI completando quanto segue:

- AWS SAM prerequisiti.
- Installa il AWS SAMCLI.

Prima dell'uso`sam local invoke`, si consiglia una conoscenza di base di quanto segue:

- Configurazione del AWS SAMCLI.
- Crea la tua candidatura in AWS SAM.

- Introduzione alla costruzione con AWS SAM.
- Introduzione alla distribuzione con AWS SAM.

Richiama una funzione Lambda localmente

Quando si eseguesam `local invoke`, si AWS SAMCLI presuppone che la directory di lavoro corrente sia la directory principale del progetto. AWS SAMCLICercheranno innanzitutto un `template.[yaml|yml]` file all'interno di una `.aws-sam` sottocartella. Se non viene trovato, AWS SAMCLI cercherà un `template.[yaml|yml]` file nella directory di lavoro corrente.

Per richiamare una funzione Lambda localmente

1. Dalla directory principale del progetto, esegui quanto segue:

```
$ sam local invoke <options>
```

2. Se l'applicazione contiene più di una funzione, fornite l'ID logico della funzione. Di seguito è riportato un esempio:

```
$ sam local invoke HelloWorldFunction
```

3. AWS SAMCLICostruisce la tua funzione in un contenitore locale utilizzandoDocker. Quindi richiama la tua funzione e restituisce la risposta della tua funzione.

   Di seguito è riportato un esempio:

```
$ sam local invoke
Invoking app.lambda_handler (python3.9)
Local image is out of date and will be updated to the latest runtime. To skip this,
 pass in the parameter --skip-pull-image
Building
 image.....................................................................................
Using local image: public.ecr.aws/lambda/python:3.9-rapid-x86_64.

Mounting /Users/.../sam-app/.aws-sam/build/HelloWorldFunction as /var/
task:ro,delegated, inside runtime container
START RequestId: 64bf7e54-5509-4762-a97c-3d740498d3df Version: $LATEST
END RequestId: 64bf7e54-5509-4762-a97c-3d740498d3df
REPORT RequestId: 64bf7e54-5509-4762-a97c-3d740498d3df  Init Duration: 1.09 ms
  Duration: 608.42 ms       Billed Duration: 609 ms Memory Size: 128 MB     Max
 Memory Used: 128 MB
```

```
{"statusCode": 200, "body": "{\"message\": \"hello world\"}"}%
```

Gestione dei log

Quando si utilizza`sam local invoke`, l'output del runtime della funzione Lambda (ad esempio, i log) viene emesso su e il risultato della funzione Lambda viene emesso su. `stderr` `stdout`

Di seguito è riportato un esempio di una funzione Lambda di base:

```
def handler(event, context):
    print("some log") # this goes to stderr
    return "hello world" # this goes to stdout
```

È possibile salvare queste uscite standard. Di seguito è riportato un esempio:

```
$ sam local invoke 1> stdout.log
...

$ cat stdout.log
"hello world"

$ sam local invoke 2> stderr.log
...

$ cat stderr.log
Invoking app.lambda_handler (python3.9)
Local image is up-to-date
Using local image: public.ecr.aws/lambda/python:3.9-rapid-x86_64.
Mounting /Users/.../sam-app/.aws-sam/build/HelloWorldFunction as /var/
task:ro,delegated, inside runtime container
START RequestId: 0b46e646-3bdf-4b58-8beb-242d00912c46 Version: $LATEST
some log
END RequestId: 0b46e646-3bdf-4b58-8beb-242d00912c46
REPORT RequestId: 0b46e646-3bdf-4b58-8beb-242d00912c46  Init Duration: 0.91 ms
 Duration: 589.19 ms Billed Duration: 590 ms Memory Size: 128 MB Max Memory Used: 128
 MB
```

È possibile utilizzare questi output standard per automatizzare ulteriormente i processi di sviluppo locale.

Opzioni

Passa eventi personalizzati per richiamare la funzione Lambda

Per passare un evento alla funzione Lambda, utilizzate l'`--event`opzione. Di seguito è riportato un esempio:

```
$ sam local invoke --event events/s3.json S3JsonLoggerFunction
```

È possibile creare eventi con il `sam local generate-event` sottocomando. Per ulteriori informazioni, consulta Introduzione ai test con sam local generate-event.

Passa le variabili di ambiente quando richiami la funzione Lambda

Se la tua funzione Lambda utilizza variabili di ambiente, puoi passarle durante i test locali con l'`--env-vars`opzione. Questo è un ottimo modo per testare una funzione Lambda localmente con i servizi dell'applicazione che sono già distribuiti nel cloud. Di seguito è riportato un esempio:

```
$ sam local invoke --env-vars locals.json
```

Specificate un modello o una funzione

Per specificare un modello AWS SAMCLI a cui fare riferimento, utilizzate l'`--template`opzione. AWS SAMCLICaricherà solo quel AWS SAM modello e le risorse a cui punta.

Per richiamare una funzione di un'applicazione o di uno stack annidato, fornisci l'ID logico dell'applicazione o dello stack insieme all'ID logico della funzione. Di seguito è riportato un esempio:

```
$ sam local invoke StackLogicalId/FunctionLogicalId
```

Prova una funzione Lambda dal tuo progetto Terraform

Usa l'`--hook-name`opzione per testare localmente le funzioni Lambda dei tuoi Terraform progetti. Per ulteriori informazioni, consulta Utilizzo di AWS SAMCLI with Terraform per il debug e il test locali.

Di seguito è riportato un esempio:

```
$ sam local invoke --hook-name terraform --beta-features
```

Best practice

Se la tua applicazione ha una `.aws-sam` directory in esecuzione`sam build`, assicurati di eseguirla `sam build` ogni volta che aggiorni il codice della funzione. Quindi, `sam local invoke` esegui per testare localmente il codice funzionale aggiornato.

I test locali sono un'ottima soluzione per lo sviluppo e il test rapidi prima della distribuzione nel cloud. Tuttavia, i test locali non convalidano tutto, come le autorizzazioni tra le risorse nel cloud. Per quanto possibile, testa le tue applicazioni nel cloud. Ti consigliamo di `sam sync`utilizzarlo per velocizzare i flussi di lavoro di test sul cloud.

Esempi

Genera un evento di esempio Amazon API Gateway e utilizzalo per richiamare una funzione Lambda localmente

Innanzitutto, generiamo un payload di eventi API Gateway HTTP API Gateway e lo salviamo `events` nella nostra cartella.

```
$ sam local generate-event apigateway http-api-proxy > events/apigateway_event.json
```

Successivamente, modifichiamo la nostra funzione Lambda per restituire un valore di parametro dall'evento.

```
def lambda_handler(event, context):
    print("HelloWorldFunction invoked")
    return {
        "statusCode": 200,
        "body": json.dumps({
            "message": event['queryStringParameters']['parameter2'],
        }),
    }
```

Successivamente, invochiamo localmente la nostra funzione Lambda e forniamo il nostro evento personalizzato.

```
$ sam local invoke --event events/apigateway_event.json

Invoking app.lambda_handler (python3.9)
```

```
Local image is up-to-date
Using local image: public.ecr.aws/lambda/python:3.9-rapid-x86_64.

Mounting /Users/...sam-app/.aws-sam/build/HelloWorldFunction as /var/task:ro,delegated,
 inside runtime container
START RequestId: 59535d0d-3d9e-493d-8c98-6264e8e961b8 Version: $LATEST
some log
END RequestId: 59535d0d-3d9e-493d-8c98-6264e8e961b8
REPORT RequestId: 59535d0d-3d9e-493d-8c98-6264e8e961b8  Init Duration: 1.63 ms
 Duration: 564.07 ms       Billed Duration: 565 ms Memory Size: 128 MB     Max Memory
 Used: 128 MB
{"statusCode": 200, "body": "{\"message\": \"value\"}"}%
```

Passa le variabili di ambiente quando richiami una funzione Lambda localmente

Questa applicazione dispone di una funzione Lambda che utilizza una variabile di ambiente per il nome di una tabella Amazon DynamoDB. Di seguito è riportato un esempio della funzione definita nel modello: AWS SAM

```
AWSTemplateFormatVersion: 2010-09-09
Transform: AWS::Serverless-2016-10-31
...
Resources:
  getAllItemsFunction:
    Type: AWS::Serverless::Function
    Properties:
      Handler: src/get-all-items.getAllItemsHandler
      Description: get all items
      Policies:
        - DynamoDBReadPolicy:
            TableName: !Ref SampleTable
      Environment:
        Variables:
          SAMPLE_TABLE: !Ref SampleTable
...
```

Vogliamo testare localmente la nostra funzione Lambda facendola interagire con la nostra tabella DynamoDB nel cloud. Per fare ciò, creiamo il nostro file delle variabili di ambiente e lo salviamo nella directory principale del nostro progetto come. `locals.json` Il valore fornito qui per `SAMPLE_TABLE` fa riferimento alla nostra tabella DynamoDB nel cloud.

```
{
```

```
    "getAllItemsFunction": {
        "SAMPLE_TABLE": "dev-demo-SampleTable-1U991234LD5UM98"
    }
}
```

Successivamente, eseguiamo `sam local invoke` e passiamo le nostre variabili di ambiente con l'`--env-vars`opzione.

```
$ sam local invoke getAllItemsFunction --env-vars locals.json

Mounting /Users/...sam-app/.aws-sam/build/HelloWorldFunction as /var/task:ro,delegated,
 inside runtime container
START RequestId: 59535d0d-3d9e-493d-8c98-6264e8e961b8 Version: $LATEST
some log
END RequestId: 59535d0d-3d9e-493d-8c98-6264e8e961b8
REPORT RequestId: 59535d0d-3d9e-493d-8c98-6264e8e961b8  Init Duration: 1.63 ms
 Duration: 564.07 ms       Billed Duration: 565 ms Memory Size: 128 MB     Max Memory
 Used: 128 MB
{"statusCode":200,"body":"{}"}
```

Ulteriori informazioni

Per un elenco di tutte le `sam local invoke` opzioni, vedisam local invoke.

Per una dimostrazione dell'utilizzo`sam local`, vedi AWS SAM per lo sviluppo locale. Test Cloud AWS delle risorse provenienti da ambienti di sviluppo locali nella serie Serverless Land Sessions with SAM on YouTube.

Introduzione ai test con sam local start-api

Utilizzate il AWS Serverless Application Model `sam local start-api` sottocomando Command Line Interface (AWS SAMCLI) per eseguire le AWS Lambda funzioni localmente e testarle tramite un host di server HTTP locale. Questo tipo di test è utile per le funzioni Lambda richiamate da un endpoint Amazon API Gateway.

- Per un'introduzione a, consulta AWS SAMCLI Che cos'è il? AWS SAMCLI
- Per un elenco delle opzioni di `sam local start-api` comando, vederesam local start-api.
- Per un esempio di utilizzo `sam local start-api` durante un tipico flusso di lavoro di sviluppo, vediPassaggio 7: (Facoltativo) Testa l'applicazione localmente.

> **Note**
>
> `sam local start-api`supporta funzioni durevoli con funzionalità di checkpoint e replay automatici. Le funzioni durevoli funzionano perfettamente con l'endpoint API Gateway locale.

Prerequisiti

Per utilizzarlo`sam local start-api`, installa il AWS SAMCLI completando quanto segue:

- AWS SAM prerequisiti.
- Installa il AWS SAMCLI.

Prima dell'uso`sam local start-api`, si consiglia una conoscenza di base di quanto segue:

- Configurazione del AWS SAMCLI.
- Crea la tua candidatura in AWS SAM.
- Introduzione alla costruzione con AWS SAM.
- Introduzione alla distribuzione con AWS SAM.

Usare sam local start-api

Quando si esegue`sam local start-api`, si AWS SAMCLI presuppone che la directory di lavoro corrente sia la directory principale del progetto. AWS SAMCLICercheranno innanzitutto un `template.[yaml|yml]` file all'interno di una `.aws-sam` sottocartella. Se non viene trovato, AWS SAMCLI cercherà un `template.[yaml|yml]` file all'interno della directory di lavoro corrente.

Per avviare un server HTTP locale

1. Dalla directory principale del progetto, esegui quanto segue:

```
$ sam local start-api <options>
```

2. AWS SAMCLICrea le tue funzioni Lambda in un Docker contenitore locale. Quindi restituisce l'indirizzo locale dell'endpoint del server HTTP. Di seguito è riportato un esempio:

```
$ sam local start-api
```

```
Initializing the lambda functions containers.
Local image is up-to-date
Using local image: public.ecr.aws/lambda/python:3.9-rapid-x86_64.

Mounting /Users/.../sam-app/.aws-sam/build/HelloWorldFunction as /var/
task:ro,delegated, inside runtime container
Containers Initialization is done.
Mounting HelloWorldFunction at http://127.0.0.1:3000/hello [GET]
You can now browse to the above endpoints to invoke your functions. You do not
 need to restart/reload SAM CLI while working on your functions, changes will be
 reflected instantly/automatically. If you used sam build before running local
 commands, you will need to re-run sam build for the changes to be picked up. You
 only need to restart SAM CLI if you update your AWS SAM template
2023-04-12 14:41:05 WARNING: This is a development server. Do not use it in a
 production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:3000
```

3. È possibile richiamare la funzione Lambda tramite il browser o il prompt dei comandi. Di seguito è riportato un esempio:

```
sam-app$ curl http://127.0.0.1:3000/hello
{"message": "Hello world!"}%
```

4. Quando apporti modifiche al codice della funzione Lambda, considera quanto segue per aggiornare il server HTTP locale:

   - Se l'applicazione non dispone di una `.aws-sam` directory e la funzione utilizza un linguaggio interpretato, AWS SAMCLI aggiornerà automaticamente la funzione creando un nuovo contenitore e ospitandolo.
   - Se l'applicazione dispone di una `.aws-sam` directory, è necessario eseguirla `sam build` per aggiornare la funzione. Quindi esegui `sam local start-api` nuovamente per ospitare la funzione.
   - Se la tua funzione utilizza un linguaggio compilato o se il tuo progetto richiede un supporto complesso per la creazione di pacchetti, esegui la tua soluzione di compilazione per aggiornare la funzione. Quindi esegui `sam local start-api` nuovamente per ospitare la funzione.

Funzioni Lambda che utilizzano gli autorizzatori Lambda

> **Note**
>
> Questa funzionalità è nuova nella AWS SAMCLI versione 1.80.0. Per eseguire l'aggiornamento, consulta Aggiornamento del AWS SAMCLI.

Per le funzioni Lambda che utilizzano gli autorizzatori Lambda, richiameranno AWS SAMCLI automaticamente l'autorizzatore Lambda prima di richiamare l'endpoint della funzione Lambda.

Di seguito è riportato un esempio di avvio di un server HTTP locale per una funzione che utilizza un autorizzatore Lambda:

```
$ sam local start-api
2023-04-17 15:02:13 Attaching import module proxy for analyzing dynamic imports

AWS SAM CLI does not guarantee 100% fidelity between authorizers locally
and authorizers deployed on AWS. Any application critical behavior should
be validated thoroughly before deploying to production.

Testing application behaviour against authorizers deployed on AWS can be done using the
 sam sync command.

Mounting HelloWorldFunction at http://127.0.0.1:3000/authorized-request [GET]
You can now browse to the above endpoints to invoke your functions. You do not need
 to restart/reload SAM CLI while working on your functions, changes will be reflected
 instantly/automatically. If you used sam build before running local commands, you will
 need to re-run sam build for the changes to be picked up. You only need to restart SAM
 CLI if you update your AWS SAM template
2023-04-17 15:02:13 WARNING: This is a development server. Do not use it in a
 production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:3000
2023-04-17 15:02:13 Press CTRL+C to quit
```

Quando richiami l'endpoint della funzione Lambda tramite il server HTTP locale, il primo richiama AWS SAMCLI l'autorizzazione Lambda. Se l'autorizzazione ha esito positivo, AWS SAMCLI invocherà l'endpoint della funzione Lambda. Di seguito è riportato un esempio:

```
$ curl http://127.0.0.1:3000/authorized-request --header "header:my_token"
{"message": "from authorizer"}%
```

```
Invoking app.authorizer_handler (python3.8)
Local image is up-to-date
Using local image: public.ecr.aws/lambda/python:3.8-rapid-x86_64.

Mounting /Users/.../sam-app/... as /var/task:ro,delegated, inside runtime container
START RequestId: 38d3b472-a2c8-4ea6-9a77-9b386989bef0 Version: $LATEST
END RequestId: 38d3b472-a2c8-4ea6-9a77-9b386989bef0
REPORT RequestId: 38d3b472-a2c8-4ea6-9a77-9b386989bef0    Init Duration: 1.08 ms
 Duration: 628.26 msBilled Duration: 629 ms    Memory Size: 128 MB    Max Memory Used:
 128 MB
Invoking app.request_handler (python3.8)
Using local image: public.ecr.aws/lambda/python:3.8-rapid-x86_64.

Mounting /Users/.../sam-app/... as /var/task:ro,delegated, inside runtime container
START RequestId: fdc12255-79a3-4365-97e9-9459d06446ff Version: $LATEST
END RequestId: fdc12255-79a3-4365-97e9-9459d06446ff
REPORT RequestId: fdc12255-79a3-4365-97e9-9459d06446ff    Init Duration: 0.95 ms
 Duration: 659.13 msBilled Duration: 660 ms    Memory Size: 128 MB    Max Memory Used:
 128 MB
No Content-Type given. Defaulting to 'application/json'.
2023-04-17 15:03:03 127.0.0.1 - - [17/Apr/2023 15:03:03] "GET /authorized-request
 HTTP/1.1" 200 -
```

Opzioni

Riutilizza continuamente i contenitori per velocizzare le chiamate alle funzioni locali

Per impostazione predefinita, AWS SAMCLI crea un nuovo contenitore ogni volta che la funzione viene richiamata tramite il server HTTP locale. Utilizzate l'`--warm-containers`opzione per riutilizzare automaticamente il contenitore per le chiamate di funzioni. Ciò velocizza il tempo necessario per AWS SAMCLI preparare la funzione Lambda per la chiamata locale. È possibile personalizzare ulteriormente questa opzione fornendo l'`eager`argomento or. `lazy`

- `eager`— I contenitori per tutte le funzioni vengono caricati all'avvio e persistono tra una chiamata e l'altra.
- `lazy`— I contenitori vengono caricati solo quando ogni funzione viene richiamata per la prima volta. Quindi persistono per ulteriori invocazioni.

Di seguito è riportato un esempio:

```
$ sam local start-api --warm-containers eager
```

Quando usi `--warm-containers` e modifichi il codice della funzione Lambda:

- Se l'applicazione ha una `.aws-sam` directory, eseguila `sam build` per aggiornare il codice della funzione negli artefatti di build dell'applicazione.
- Quando viene rilevata una modifica al codice, spegne AWS SAMCLI automaticamente il contenitore delle funzioni Lambda.
- Quando richiami nuovamente la funzione, crea AWS SAMCLI automaticamente un nuovo contenitore.

Specificate un'immagine del contenitore da usare per le funzioni Lambda

Per impostazione predefinita, AWS SAMCLI utilizza immagini di base Lambda da Amazon Elastic Container Registry (Amazon ECR) per richiamare le funzioni localmente. Usa l'`--invoke-image`opzione per fare riferimento a un'immagine del contenitore personalizzata. Di seguito è riportato un esempio:

```
$ sam local start-api --invoke-image public.ecr.aws/sam/emu-python3.8
```

È possibile specificare la funzione da utilizzare con l'immagine del contenitore personalizzata. Di seguito è riportato un esempio:

```
$ sam local start-api --invoke-image Function1=amazon/aws/sam-cli-emulation-image-
python3.8
```

Specificate un modello da testare localmente

Per specificare un modello AWS SAMCLI a cui fare riferimento, utilizzate l'`--template`opzione. AWS SAMCLICaricherà solo quel AWS SAM modello e le risorse a cui punta. Di seguito è riportato un esempio:

```
$ sam local start-api --template myTemplate.yaml
```

Specificate l'ambiente di sviluppo host della vostra funzione Lambda

Per impostazione predefinita, il `sam local start-api` sottocomando crea un server HTTP utilizzando `localhost` un indirizzo IP. `127.0.0.1` È possibile personalizzare questi valori se l'ambiente di sviluppo locale è isolato dal computer locale.

Utilizzate l'`--container-host`opzione per specificare un host. Di seguito è riportato un esempio:

```
$ sam local start-api --container-host host.docker.internal
```

Utilizzate l'`--container-host-interface`opzione per specificare l'indirizzo IP della rete host a cui devono collegarsi le porte del contenitore. Di seguito è riportato un esempio:

```
$ sam local start-api --container-host-interface 0.0.0.0
```

Best practice

Se l'applicazione ha una `.aws-sam` directory in esecuzione`sam build`, assicuratevi di eseguirla `sam build` ogni volta che aggiornate il codice della funzione. Quindi, `sam local start-api` esegui per testare localmente il codice funzionale aggiornato.

I test locali sono un'ottima soluzione per lo sviluppo e il test rapidi prima della distribuzione nel cloud. Tuttavia, i test locali non convalidano tutto, come le autorizzazioni tra le risorse nel cloud. Per quanto possibile, testa le tue applicazioni nel cloud. Ti consigliamo di `sam sync`utilizzarlo per velocizzare i flussi di lavoro di test sul cloud.

Ulteriori informazioni

Per un elenco di tutte le `sam local start-api` opzioni, consultasam local start-api.

Introduzione ai test con sam local start-lambda

Usa il AWS SAMCLI sottocomando `sam local start-lambda` per richiamare la funzione Lambda tramite and. AWS CLI SDKs Questo comando avvia un endpoint locale che emula Lambda.

- Per un'introduzione a, vedere AWS SAMCLI Che cos'è il? AWS SAMCLI
- Per un elenco delle opzioni di `sam local start-lambda` comando, vederesam local start-lambda.

> **Note**
>
> `sam local start-lambda`supporta funzioni durevoli con funzionalità di checkpoint e replay automatici. Le funzioni durevoli funzionano perfettamente con l'endpoint Lambda locale.

Prerequisiti

Per utilizzarlo`sam local start-lambda`, installa il AWS SAMCLI completando quanto segue:

- AWS SAM prerequisiti.
- Installa il AWS SAMCLI.

Prima dell'uso`sam local start-lambda`, si consiglia una conoscenza di base di quanto segue:

- Configurazione del AWS SAMCLI.
- Crea la tua candidatura in AWS SAM.
- Introduzione alla costruzione con AWS SAM.
- Introduzione alla distribuzione con AWS SAM.

Usare sam local start-lambda

Quando si esegue`sam local start-lambda`, si AWS SAMCLI presuppone che la directory di lavoro corrente sia la directory principale del progetto. AWS SAMCLICercheranno innanzitutto un `template.[yaml|yml]` file all'interno di una `.aws-sam` sottocartella. Se non viene trovato, AWS SAMCLI cercherà un `template.[yaml|yml]` file all'interno della directory di lavoro corrente.

Per usare sam local start-lambda

1. Dalla directory principale del progetto, esegui quanto segue:

```
$ sam local start-lambda <options>
```

2. AWS SAMCLICrea le tue funzioni Lambda in un Docker contenitore locale. Quindi invia l'indirizzo locale all'endpoint del server HTTP. Di seguito è riportato un esempio:

```
$ sam local start-lambda
```

```
Initializing the lambda functions containers.
Local image is up-to-date
Using local image: public.ecr.aws/lambda/python:3.9-rapid-x86_64.

Mounting /Users/.../sam-app/hello_world as /var/task:ro,delegated, inside runtime
 container
Containers Initialization is done.
Starting the Local Lambda Service. You can now invoke your Lambda Functions defined
 in your template through the endpoint.
2023-04-13 07:25:43 WARNING: This is a development server. Do not use it in a
 production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:3001
2023-04-13 07:25:43 Press CTRL+C to quit
```

3. Usa AWS CLI o SDKs per richiamare la funzione Lambda localmente.

   Di seguito è riportato un esempio di utilizzo di: AWS CLI

```
$ aws lambda invoke --function-name "HelloWorldFunction" --endpoint-
url "http://127.0.0.1:3001" --no-verify-ssl out.txt

StatusCode: 200
(END)
```

   Di seguito è riportato un esempio di utilizzo del AWS SDK comando forPython:

```
import boto3
from botocore.config import Config
from botocore import UNSIGNED

lambda_client = boto3.client('lambda',
                             endpoint_url="http://127.0.0.1:3001",
                             use_ssl=False,
                             verify=False,
                             config=Config(signature_version=UNSIGNED,
                                           read_timeout=1,
                                           retries={'max_attempts': 0}
                                           )
                            )
lambda_client.invoke(FunctionName="HelloWorldFunction")
```

Opzioni

Specificare un modello

Per specificare un modello AWS SAMCLI a cui fare riferimento, utilizzate l'`--template`opzione. AWS SAMCLICaricherà solo quel AWS SAM modello e le risorse a cui punta. Di seguito è riportato un esempio:

```
$ sam local start-lambda --template myTemplate.yaml
```

Per ulteriori informazioni sui AWS SAM modelli, vedereAWS SAM anatomia del modello.

Best practice

Se l'applicazione ha una `.aws-sam` directory in esecuzione`sam build`, assicurati di eseguirla `sam build` ogni volta che aggiorni il codice della funzione. Quindi, `sam local start-lambda` esegui per testare localmente il codice funzionale aggiornato.

I test locali sono un'ottima soluzione per lo sviluppo e il test rapidi prima della distribuzione nel cloud. Tuttavia, i test locali non convalidano tutto, come le autorizzazioni tra le risorse nel cloud. Per quanto possibile, testa le tue applicazioni nel cloud. Ti consigliamo di `sam sync`utilizzarlo per velocizzare i flussi di lavoro di test sul cloud.

Ulteriori informazioni

Per un elenco di tutte le `sam local start-lambda` opzioni, consultasam local start-lambda.

Richiama localmente le funzioni Lambda con AWS SAM

Richiamare localmente una funzione Lambda prima del test o dell'implementazione nel cloud può avere una serie di vantaggi. Ti consente di testare più rapidamente la logica della tua funzione. I test innanzitutto a livello locale riducono la probabilità di identificare problemi durante i test nel cloud o durante l'implementazione, il che può aiutarvi a evitare costi inutili. Inoltre, i test locali semplificano l'esecuzione del debug.

È possibile richiamare la funzione Lambda localmente utilizzando sam local invoke il comando e fornendo l'ID logico della funzione e un file di eventi. sam local invokeaccetta anche `stdin` come evento. Per ulteriori informazioni sugli eventi, consulta Event nella AWS Lambda Developer Guide.

Per informazioni sui formati dei messaggi relativi agli eventi di diversi AWS servizi, consulta Utilizzo AWS Lambda con altri servizi nella Guida per gli AWS Lambda sviluppatori.

> **Note**
>
> Il sam local invoke comando corrisponde al comando AWS Command Line Interface (AWS CLI) `aws lambda invoke`. È possibile utilizzare entrambi i comandi per richiamare una funzione Lambda.

È necessario eseguire il sam local invoke comando nella directory del progetto che contiene la funzione che si desidera richiamare.

Esempi:

```
# Invoking function with event file
$ sam local invoke "Ratings" -e event.json

# Invoking function with event via stdin
$ echo '{"message": "Hey, are you there?" }' | sam local invoke --event - "Ratings"

# For more options
$ sam local invoke --help
```

File di variabili di ambiente

Per dichiarare localmente le variabili di ambiente che sovrascrivono i valori definiti nei modelli, procedi come segue:

1. Crea un file JSON che contenga le variabili di ambiente da sovrascrivere.
2. Usa l'`--env-vars`argomento per sovrascrivere i valori definiti nei tuoi modelli.

Dichiarazione delle variabili di ambiente

Per dichiarare le variabili di ambiente che si applicano globalmente a tutte le risorse, specificate un `Parameters` oggetto come il seguente:

```
{
```

```
    "Parameters": {
        "TABLE_NAME": "localtable",
        "BUCKET_NAME": "amzn-s3-demo-bucket",
        "STAGE": "dev"
    }
}
```

Per dichiarare variabili di ambiente diverse per ogni risorsa, specificate gli oggetti per ogni risorsa come segue:

```
{
    "MyFunction1": {
        "TABLE_NAME": "localtable",
        "BUCKET_NAME": "amzn-s3-demo-bucket",
    },
    "MyFunction2": {
        "TABLE_NAME": "localtable",
        "STAGE": "dev"
    }
}
```

Quando si specificano oggetti per ogni risorsa, è possibile utilizzare i seguenti identificatori, elencati in ordine di precedenza dalla più alta alla più bassa:

1. `logical_id`
2. `function_id`
3. `function_name`
4. Identificatore di percorso completo

È possibile utilizzare entrambi i metodi precedenti per dichiarare le variabili di ambiente insieme in un unico file. In tal caso, le variabili di ambiente fornite per risorse specifiche hanno la precedenza sulle variabili di ambiente globali.

Salva le variabili di ambiente in un file JSON, ad esempio. `env.json`

Ignorare i valori delle variabili di ambiente

Per sovrascrivere le variabili di ambiente con quelle definite nel tuo file JSON, usa l'`--env-vars`argomento con i invoke comandi or. start-api Esempio:

```
sam local invoke --env-vars env.json
```

Livelli

Se l'applicazione include livelli, per informazioni su come eseguire il debug dei problemi con i layer sull'host locale, consultaAumenta l'efficienza utilizzando i livelli Lambda con AWS SAM.

Ulteriori informazioni

Per un esempio pratico di invocazione di funzioni a livello locale, vedi Modulo 2 - Esegui localmente in The Complete Workshop. AWS SAM

API Gateway eseguito localmente con AWS SAM

L'esecuzione locale di Amazon API Gateway può offrire una serie di vantaggi. Ad esempio, l'esecuzione locale di API Gateway consente di testare gli endpoint API localmente prima della distribuzione nel AWS cloud. Se esegui prima il test localmente, spesso puoi ridurre i test e lo sviluppo nel cloud, il che può aiutare a ridurre i costi. Inoltre, l'esecuzione locale semplifica il debug.

Per avviare un'istanza locale di API Gateway da utilizzare per testare la funzionalità di richiesta/risposta HTTP, utilizzate il `sam local start-api` AWS SAMCLI comando. Questa funzionalità include il ricaricamento a caldo in modo da poter sviluppare e iterare rapidamente le funzioni.

> **Note**
>
> Il ricaricamento a caldo avviene quando vengono aggiornati solo i file modificati e lo stato dell'applicazione rimane lo stesso. Al contrario, il ricaricamento in tempo reale avviene quando l'intera applicazione viene aggiornata e lo stato dell'applicazione viene perso.

Per istruzioni sull'uso del `sam local start-api` comando, vedere. Introduzione ai test con sam local start-api

Per impostazione predefinita, AWS SAM utilizza integrazioni AWS Lambda proxy e supporta sia i tipi di risorse che `HttpApi` i tipi di `Api` risorse. Per ulteriori informazioni sulle integrazioni proxy per i tipi di `HttpApi` risorse, consulta Working with AWS Lambda proxy integrations for HTTP APIs nella API Gateway Developer Guide. Per ulteriori informazioni sulle integrazioni proxy con i tipi di `Api` risorse, consulta Understand API Gateway Lambda Proxy Integration nella API Gateway Developer Guide.

Esempio:

```
$ sam local start-api
```

AWS SAM trova automaticamente tutte le funzioni all'interno del AWS SAM modello che hanno `HttpApi` o sono state definite fonti di `Api` eventi. Quindi, monta la funzione nei percorsi HTTP definiti.

Nell'`Api`esempio seguente, la `Ratings` funzione viene montata `ratings.py:handler()` su `/ratings` per `GET` le richieste:

```
Ratings:
  Type: AWS::Serverless::Function
  Properties:
    Handler: ratings.handler
    Runtime: python3.9
    Events:
      Api:
        Type: Api
        Properties:
          Path: /ratings
          Method: get
```

Ecco un esempio di `Api` risposta:

```
// Example of a Proxy Integration response
exports.handler = (event, context, callback) => {
    callback(null, {
        statusCode: 200,
        headers: { "x-custom-header" : "my custom header value" },
        body: "hello world"
    });
}
```

Se modifichi il codice della funzione, esegui il `sam build` comando for `sam local start-api` per rilevare le modifiche.

File di variabili di ambiente

Per dichiarare localmente le variabili di ambiente che sovrascrivono i valori definiti nei modelli, procedi come segue:

1. Crea un file JSON che contenga le variabili di ambiente da sovrascrivere.
2. Usa l'`--env-vars`argomento per sovrascrivere i valori definiti nei tuoi modelli.

Dichiarazione delle variabili di ambiente

Per dichiarare variabili di ambiente che si applicano globalmente a tutte le risorse, specificate un `Parameters` oggetto come il seguente:

```
{
    "Parameters": {
        "TABLE_NAME": "localtable",
        "BUCKET_NAME": "amzn-s3-demo-bucket",
        "STAGE": "dev"
    }
}
```

Per dichiarare variabili di ambiente diverse per ogni risorsa, specificate gli oggetti per ogni risorsa come segue:

```
{
    "MyFunction1": {
        "TABLE_NAME": "localtable",
        "BUCKET_NAME": "amzn-s3-demo-bucket",
    },
    "MyFunction2": {
        "TABLE_NAME": "localtable",
        "STAGE": "dev"
    }
}
```

Quando si specificano oggetti per ogni risorsa, è possibile utilizzare i seguenti identificatori, elencati in ordine di precedenza dalla più alta alla più bassa:

1. `logical_id`
2. `function_id`
3. `function_name`
4. Identificatore di percorso completo

È possibile utilizzare entrambi i metodi precedenti per dichiarare le variabili di ambiente insieme in un unico file. In tal caso, le variabili di ambiente fornite per risorse specifiche hanno la precedenza sulle variabili di ambiente globali.

Salva le variabili di ambiente in un file JSON, ad esempio. `env.json`

Ignorare i valori delle variabili di ambiente

Per sovrascrivere le variabili di ambiente con quelle definite nel tuo file JSON, usa l'`--env-vars`argomento con i invoke comandi or. start-api Esempio:

```
$ sam local start-api --env-vars env.json
```

Livelli

Se l'applicazione include livelli, per informazioni su come eseguire il debug dei problemi con i layer sull'host locale, consultaAumenta l'efficienza utilizzando i livelli Lambda con AWS SAM.

Introduzione al cloud testing con sam remote test-event

Utilizzate il AWS Serverless Application Model comando Command Line Interface (AWS SAM CLI) `sam remote test-event` per accedere e gestire eventi di test condivisibili per le vostre AWS Lambda funzioni.

Per ulteriori informazioni sugli eventi di test condivisibili, consulta gli eventi di test condivisibili nella Guida per gli sviluppatori.AWS Lambda

Argomenti

- Configura AWS SAMCLI da usare sam remote test-event
- Utilizzo del comando sam remote test-event
- Utilizzo di eventi di test condivisibili
- Gestione di eventi di test condivisibili

Prerequisiti

Per utilizzarlo`sam remote test-event`, installa AWS SAMCLI completando quanto segue:

- AWS SAM prerequisiti.
- Installa il AWS SAMCLI.

Se l'hai già AWS SAM CLI installato, ti consigliamo di eseguire l'aggiornamento alla versione più recente della AWS SAMCLI versione. Per ulteriori informazioni, consulta Aggiornamento del AWS SAMCLI.

Prima dell'uso`sam remote test-event`, si consiglia una conoscenza di base di quanto segue:

- Configurazione del AWS SAMCLI.
- Crea la tua candidatura in AWS SAM.
- Introduzione alla costruzione con AWS SAM.
- Introduzione alla distribuzione con AWS SAM.
- Introduzione all'uso sam sync della sincronizzazione con Cloud AWS.

Configura AWS SAMCLI da usare sam remote test-event

Completa i seguenti passaggi di configurazione per utilizzare il AWS SAM CLI `sam remote test-event` comando:

1. Configuralo AWS SAM CLI per utilizzare il tuo Account AWS: gli eventi di test condivisibili per Lambda sono accessibili e gestiti dagli utenti all'interno della stessa. Account AWS Per configurare l' AWS SAM CLIutilizzo del tuo Account AWS, vedi. Configurazione del AWS SAMCLI
2. Configura le autorizzazioni per gli eventi di test condivisibili: per accedere e gestire gli eventi di test condivisibili, devi disporre delle autorizzazioni appropriate. Per ulteriori informazioni, consulta Shareable test events nella Developer Guide.AWS Lambda

Utilizzo del comando sam remote test-event

Il AWS SAM CLI `sam remote test-event` comando fornisce i seguenti sottocomandi che potete utilizzare per accedere e gestire gli eventi di test condivisibili:

- `delete`— Eliminare un evento di test condivisibile dal registro degli EventBridge schemi di Amazon.
- `get`— Ottieni un evento di test condivisibile dal registro dello EventBridge schema.

- `list`— Elenca gli eventi di test condivisibili esistenti per una funzione dal registro dello EventBridge schema.
- `put`— Salva un evento da un file locale nel registro dello EventBridge schema.

Per elencare questi sottocomandi utilizzando il AWS SAM CLI, esegui quanto segue:

```
$ sam remote test-event --help
```

Eliminazione di eventi di test condivisibili

È possibile eliminare un evento di test condivisibile utilizzando il `delete` sottocomando insieme a quanto segue:

- Fornire il nome dell'evento di test condivisibile da eliminare.
- Fornisci un ID accettabile della funzione Lambda associata all'evento.
- Se si fornisce l'ID logico della funzione Lambda, è necessario fornire anche il nome dello AWS CloudFormation stack associato alla funzione Lambda.

Di seguito è riportato un esempio:

```
$ sam remote test-event delete HelloWorldFunction --stack-name sam-app --name demo-
event
```

Per un elenco delle opzioni da utilizzare con il `delete` sottocomando, vedere. sam remote test-event delete È inoltre possibile eseguire quanto segue da: AWS SAM CLI

```
$ sam remote test-event delete --help
```

Ottenere eventi di test condivisibili

È possibile ottenere un evento di test condivisibile dal registro dello EventBridge schema utilizzando il `get` sottocomando insieme a quanto segue:

- Fornisci il nome dell'evento di test condivisibile da ottenere.
- Fornisci un ID accettabile della funzione Lambda associata all'evento.
- Se si fornisce l'ID logico della funzione Lambda, è necessario fornire anche il nome dello AWS CloudFormation stack associato alla funzione Lambda.

Di seguito è riportato un esempio che ottiene un evento di test condivisibile denominato `demo-event` associato alla funzione `HelloWorldFunction` Lambda dello `sam-app` stack. Questo comando stamperà l'evento sulla tua console.

```
$ sam remote test-event get HelloWorldFunction --stack-name sam-app --name demo-event
```

Per ottenere un evento di test condivisibile e salvarlo sul computer locale, usa l'`--output-file`opzione e fornisci un percorso e un nome del file. Di seguito è riportato un esempio che salva `demo-event` come `demo-event.json` nella directory di lavoro corrente:

```
$ sam remote test-event get HelloWorldFunction --stack-name sam-app --name demo-event
 --output-file demo-event.json
```

Per un elenco delle opzioni da utilizzare con il `get` sottocomando, vedetesam remote test-event get. È inoltre possibile eseguire quanto segue da: AWS SAM CLI

```
$ sam remote test-event get --help
```

Elenco degli eventi di test condivisibili

Puoi elencare tutti gli eventi di test condivisibili per una particolare funzione Lambda dal registro dello schema. Utilizzate il `list` sottocomando insieme a quanto segue:

- Fornisci un ID accettabile della funzione Lambda associata agli eventi.
- Se si fornisce l'ID logico della funzione Lambda, è necessario fornire anche il nome dello CloudFormation stack associato alla funzione Lambda.

Di seguito è riportato un esempio che ottiene un elenco di tutti gli eventi di test condivisibili associati alla funzione `HelloWorldFunction` Lambda dello stack: `sam-app`

```
$ sam remote test-event list HelloWorldFunction --stack-name sam-app
```

Per un elenco delle opzioni da utilizzare con il sottocomando, vedere. `list` sam remote test-event list È inoltre possibile eseguire quanto segue da: AWS SAM CLI

```
$ sam remote test-event list --help
```

Salvataggio di eventi di test condivisibili

È possibile salvare eventi di test condivisibili nel registro dello EventBridge schema. Utilizzate il `put` sottocomando insieme a quanto segue:

- Fornisci un ID accettabile della funzione Lambda associata all'evento di test condivisibile.
- Fornisci un nome per l'evento di test condivisibile.
- Fornisci il percorso e il nome del file all'evento locale da caricare.

Di seguito è riportato un esempio che salva l'`demo-event.json`evento locale come `demo-event` e lo associa alla funzione `HelloWorldFunction` Lambda dello `sam-app` stack:

```
$ sam remote test-event put HelloWorldFunction --stack-name sam-app --name demo-event
 --file demo-event.json
```

Se nel registro dello EventBridge schema esiste un evento di test condivisibile con lo stesso nome, non lo AWS SAM CLI sovrascriverà. Per sovrascrivere, aggiungi l'`--force`opzione al tuo comando.

Per un elenco delle opzioni da utilizzare con il `put` sottocomando, vedere. sam remote test-event put È inoltre possibile eseguire quanto segue da: AWS SAM CLI

```
$ sam remote test-event put --help
```

Utilizzo di eventi di test condivisibili

Usa eventi di test condivisibili per testare le tue funzioni Lambda con Cloud AWS `sam remote invoke` il comando. Per ulteriori informazioni, consulta Passa eventi di test condivisibili a una funzione Lambda nel cloud.

Gestione di eventi di test condivisibili

Questo argomento contiene esempi su come gestire e utilizzare eventi di test condivisibili.

Ottieni un evento di test condivisibile, modificalo e utilizzalo

È possibile ottenere un evento di test condivisibile dal registro degli EventBridge schemi, modificarlo localmente e utilizzare l'evento di test locale con la funzione Lambda in. Cloud AWS Di seguito è riportato un esempio:

1. Recupera l'evento di test condivisibile: utilizza il `sam remote test-event get` sottocomando per recuperare un evento di test condivisibile per una funzione Lambda specifica e salvarlo localmente:

```
$ sam remote test-event get HelloWorldFunction --stack-name sam-app --name demo-event
 --output-file demo-event.json
```

2. Modifica l'evento di test condivisibile: utilizza un editor di testo a tua scelta per modificare l'evento di test condivisibile.
3. Usa l'evento di test condivisibile: utilizza il `sam remote invoke` comando e fornisci il percorso del file e il nome dell'evento con: `--event-file`

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --event-file demo-
event.json
```

Ottieni un evento di test condivisibile, modificalo, caricalo e usalo

È possibile ottenere un evento di test condivisibile dal registro dello EventBridge schema, modificarlo localmente e caricarlo. Quindi, puoi passare l'evento di test condivisibile direttamente alla tua funzione Lambda in. Cloud AWS Di seguito è riportato un esempio:

1. Recupera l'evento di test condivisibile: utilizza il `sam remote test-event get` sottocomando per recuperare un evento di test condivisibile per una funzione Lambda specifica e salvarlo localmente:

```
$ sam remote test-event get HelloWorldFunction --stack-name sam-app --name demo-event
 --output-file demo-event.json
```

2. Modifica l'evento di test condivisibile: utilizza un editor di testo a tua scelta per modificare l'evento di test condivisibile.
3. Carica l'evento di test condivisibile: utilizza il `sam remote test-event put` sottocomando per caricare e salvare l'evento di test condivisibile nel registro dello schema. EventBridge In questo esempio, utilizziamo l'`--force`opzione per sovrascrivere una versione precedente del nostro test condivisibile:

```
$ sam remote test-event put HelloWorldFunction --stack-name sam-app --name demo-event
 --file demo-event.json --force
```

4. Passa l'evento di test condivisibile alla tua funzione Lambda: usa `sam remote invoke` il comando per passare l'evento di test condivisibile direttamente alla tua funzione Lambda in: Cloud AWS

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --test-event-name demo-
event
```

Introduzione ai test nel cloud con sam remote invoke

Utilizzate il AWS Serverless Application Model comando Command Line Interface (AWS SAM CLI) `sam remote invoke` per interagire con AWS le risorse supportate in Cloud AWS. È possibile utilizzare `sam remote invoke` per richiamare le seguenti risorse:

- Amazon Kinesis Data Streams: invia record di dati alle applicazioni Kinesis Data Streams.
- AWS Lambda— Richiama e passa eventi alle funzioni Lambda.
- Amazon Simple Queue Service (Amazon SQS): invia messaggi alle code di Amazon SQS.
- AWS Step Functions— Richiama le macchine a stati Step Functions per avviare l'esecuzione.

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Per un esempio di utilizzo `sam remote invoke` durante un tipico flusso di lavoro di sviluppo, vediFase 5: Interagite con la vostra funzione nel Cloud AWS.

Argomenti

- Utilizzo del comando sam remote invoke
- Utilizzo delle opzioni del comando sam remote invoke
- Configura il file di configurazione del progetto
- Esempi
- Link correlati

Prerequisiti

Per utilizzarlo`sam remote invoke`, installa il AWS SAMCLI completando quanto segue:

- AWS SAM prerequisiti.

- Installa il AWS SAMCLI.

Ti consigliamo inoltre di eseguire l'aggiornamento alla versione più recente di. AWS SAMCLI Per ulteriori informazioni, consulta Aggiornamento del AWS SAMCLI.

Prima dell'uso`sam remote invoke`, si consiglia una conoscenza di base di quanto segue:

- Configurazione del AWS SAMCLI.
- Crea la tua candidatura in AWS SAM.
- Introduzione alla costruzione con AWS SAM.
- Introduzione alla distribuzione con AWS SAM.
- Introduzione all'uso sam sync della sincronizzazione con Cloud AWS.

Utilizzo del comando sam remote invoke

Prima di utilizzare questo comando, la risorsa deve essere distribuita su. Cloud AWS

Utilizza la seguente struttura di comandi ed esegui dalla directory principale del progetto:

```
$ sam remote invoke <arguments> <options>
```

> **Note**
>
> Questa pagina mostrerà le opzioni fornite al prompt dei comandi. È inoltre possibile configurare le opzioni nel file di configurazione del progetto anziché passarle al prompt dei comandi. Per ulteriori informazioni, vedi Configura le impostazioni del progetto.

Per una descrizione degli `sam remote invoke` argomenti e delle opzioni, vedere sam remote invoke.

Utilizzo con Kinesis Data Streams

È possibile inviare record di dati a un'applicazione Kinesis Data Streams. AWS SAM CLIInvierà il record di dati e restituirà uno shard ID e un numero di sequenza. Di seguito è riportato un esempio:

```
$ sam remote invoke KinesisStream --stack-name kinesis-example --event hello-world
```

```
 Putting record to Kinesis data stream KinesisStream

 Auto converting value 'hello-world' into JSON '"hello-world"'. If you don't want auto-
conversion, please provide
 a JSON string as event

 {
   "ShardId": "shardId-000000000000",
   "SequenceNumber": "49646251411914806775980850790050483811301135051202232322"
 }%
```

Per inviare un record di dati

1. Fornisci un valore ID risorsa come argomento per l'applicazione Kinesis Data Streams. Per informazioni sulla risorsa valida IDs, consulta Resource ID.
2. Fornisci il record di dati come evento da inviare all'applicazione Kinesis Data Streams. È possibile fornire l'evento dalla riga di comando utilizzando l'`--event`opzione o da un file utilizzando. `--event-file` Se non fornisci un evento, AWS SAM CLI invia un evento vuoto.

Utilizzo con le funzioni Lambda

Puoi richiamare una funzione Lambda nel cloud e passare un evento vuoto o fornire un evento dalla riga di comando o da un file. AWS SAM CLIInvocherà la funzione Lambda e restituirà la sua risposta. Di seguito è riportato un esempio:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app

Invoking Lambda Function HelloWorldFunction
START RequestId: d5ef494b-5f45-4086-86fd-d7322fa1a1f9 Version: $LATEST
END RequestId: d5ef494b-5f45-4086-86fd-d7322fa1a1f9
REPORT RequestId: d5ef494b-5f45-4086-86fd-d7322fa1a1f9  Duration: 6.62 ms       Billed
 Duration: 7 ms     Memory Size: 128 MB     Max Memory Used: 67 MB  Init Duration:
 164.06 ms
{"statusCode":200,"body":"{\"message\":\"hello world\"}"}%
```

Per richiamare una funzione Lambda

1. Fornisci un valore ID risorsa come argomento per la tua funzione Lambda. Per informazioni sulla risorsa valida IDs, consulta Resource ID.

2. Fornisci un evento da inviare alla tua funzione Lambda. Puoi fornire l'evento dalla riga di comando utilizzando l'`--event`opzione o da un file utilizzando`--event-file`. Se non fornisci un evento, AWS SAM CLI invia un evento vuoto.

Funzioni Lambda configurate con lo streaming di risposte

Il `sam remote invoke` comando supporta le funzioni Lambda configurate per lo streaming di risposte. Puoi configurare una funzione Lambda per lo streaming delle risposte utilizzando la `FunctionUrlConfig` proprietà nei tuoi AWS SAM modelli. Quando si utilizza`sam remote invoke`, rileva AWS SAMCLI automaticamente la configurazione Lambda e la richiama con lo streaming di risposta.

Per vedere un esempio, consulta Richiama una funzione Lambda configurata per lo streaming di risposte.

Passa eventi di test condivisibili a una funzione Lambda nel cloud

Gli eventi di test condivisibili sono eventi di test che puoi condividere con altri nello stesso ambiente. Account AWS Per saperne di più, consulta Shareable test events nella AWS Lambda Developer Guide.

Accesso e gestione degli eventi di test condivisibili

È possibile utilizzare il AWS SAM CLI `sam remote test-event` comando per accedere e gestire eventi di test condivisibili. Ad esempio, puoi usare `sam remote test-event` per fare quanto segue:

- Recupera eventi di test condivisibili dal registro degli EventBridge schemi di Amazon.
- Modifica gli eventi di test condivisibili localmente e caricali nel registro dello EventBridge schema.
- Elimina gli eventi di test condivisibili dal registro dello EventBridge schema.

Per ulteriori informazioni, consulta Introduzione al cloud testing con sam remote test-event.

Passa un evento di test condivisibile a una funzione Lambda nel cloud

Per passare un evento di test condivisibile dal registro degli EventBridge schemi alla tua funzione Lambda nel cloud, usa l'opzione e fornisci `--test-event-name` il nome dell'evento di test condivisibile. Di seguito è riportato un esempio:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --test-event-name demo-
event
```

Se salvi l'evento di test condivisibile localmente, puoi utilizzare l'`--event-file`opzione e fornire il percorso del file e il nome dell'evento di test locale. Di seguito è riportato un esempio:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --event-file demo-
event.json
```

Using with Amazon SQS

Puoi inviare messaggi alle code di Amazon SQS. AWS SAM CLIRestituisce quanto segue:

- ID messaggio
- MD5 del corpo del messaggio
- Metadati di risposta

Di seguito è riportato un esempio:

```
$ sam remote invoke MySqsQueue --stack-name sqs-example -event hello

Sending message to SQS queue MySqsQueue

{
  "MD5OfMessageBody": "5d41402abc4b2a76b9719d911017c592",
  "MessageId": "05c7af65-9ae8-4014-ae28-809d6d8ec652"
}%
```

Come inviare un messaggio

1. Fornisci un valore ID risorsa come argomento per la coda Amazon SQS. Per informazioni sulla risorsa valida IDs, consulta Resource ID.
2. Fornisci un evento da inviare alla tua coda Amazon SQS. Puoi fornire l'evento dalla riga di comando utilizzando l'`--event`opzione o da un file utilizzando. `--event-file` Se non fornisci un evento, AWS SAM CLI invia un evento vuoto.

Utilizzo con Step Functions

È possibile richiamare una macchina a stati Step Functions per avviare l'esecuzione. AWS SAM CLIAttenderà il completamento del flusso di lavoro della macchina a stati e restituirà un output dell'ultimo passaggio dell'esecuzione. Di seguito è riportato un esempio:

```
$ sam remote invoke HelloWorldStateMachine --stack-name state-machine-example --
event '{"is_developer": true}'

Invoking Step Function HelloWorldStateMachine

"Hello Developer World"%
```

Per richiamare una macchina a stati

1. Fornisci un valore ID risorsa come argomento per la macchina a stati Step Functions. Per informazioni sulla risorsa valida IDs, vedere Resource ID.
2. Fornisci un evento da inviare alla tua macchina a stati. È possibile fornire l'evento dalla riga di comando utilizzando l'`--event`opzione o da un file utilizzando`--event-file`. Se non fornisci un evento, AWS SAM CLI invia un evento vuoto.

Utilizzo delle opzioni del comando sam remote invoke

Questa sezione descrive alcune delle opzioni principali che è possibile utilizzare con il `sam remote invoke` comando. Per un elenco completo delle opzioni, consultasam remote invoke.

Passa un evento alla tua risorsa

Usa le seguenti opzioni per passare eventi alle tue risorse nel cloud:

- `--event`— Passa un evento dalla riga di comando.
- `--event-file`— Passa un evento da un file.

Esempi Lambda

Si usa **`--event`** per passare un evento alla riga di comando come valore di stringa:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --event '{"message":
 "hello!"}'
```

```
Invoking Lambda Function HelloWorldFunction


START RequestId: b992292d-1fac-4aa2-922a-c9dc5c6fceab Version: $LATEST
END RequestId: b992292d-1fac-4aa2-922a-c9dc5c6fceab
REPORT RequestId: b992292d-1fac-4aa2-922a-c9dc5c6fceab  Duration: 16.41 ms      Billed
 Duration: 17 ms  Memory Size: 128 MB     Max Memory Used: 67 MB  Init Duration: 185.96
 ms
{"statusCode":200,"body":"{\"message\":\"hello!\"}"}%
```

`--event-file`Utilizzate per passare un evento da un file e fornire il percorso del file:

```
$ cat event.json

{"message": "hello from file"}%

$ sam remote invoke HelloWorldFunction --stack-name sam-app --event-file event.json


Invoking Lambda Function HelloWorldFunction


START RequestId: 3bc71f7d-153a-4b1e-8c9a-901d91b1bec9 Version: $LATEST
END RequestId: 3bc71f7d-153a-4b1e-8c9a-901d91b1bec9
REPORT RequestId: 3bc71f7d-153a-4b1e-8c9a-901d91b1bec9  Duration: 21.15 ms      Billed
 Duration: 22 ms  Memory Size: 128 MB     Max Memory Used: 67 MB
{"statusCode":200,"body":"{\"message\":\"hello from file\"}"}%
```

Passa un evento usando**`stdin`**:

```
$ cat event.json

{"message": "hello from file"}%

$ cat event.json | sam remote invoke HelloWorldFunction --stack-name sam-app --event-
file -

Reading event from stdin (you can also pass it from file with --event-file)

Invoking Lambda Function HelloWorldFunction

START RequestId: 85ecc902-8ad0-4a2b-a8c8-9bb4f65f5a7a Version: $LATEST
```

```
END RequestId: 85ecc902-8ad0-4a2b-a8c8-9bb4f65f5a7a
REPORT RequestId: 85ecc902-8ad0-4a2b-a8c8-9bb4f65f5a7a  Duration: 1.36 ms       Billed
 Duration: 2 ms   Memory Size: 128 MB       Max Memory Used: 67 MB
{"statusCode":200,"body":"{\"message\":\"hello from file\"}"}%
```

Configura l'output di AWS SAMCLI risposta

Quando si richiama una risorsa supportata con`sam remote invoke`, AWS SAMCLI restituisce una risposta che contiene quanto segue:

- Metadati della richiesta: metadati associati alla richiesta. Ciò include l'ID della richiesta e l'ora di inizio della richiesta.
- Risposta della risorsa: la risposta della risorsa dopo essere stata richiamata nel cloud.

È possibile utilizzare l'`--output`opzione per configurare la risposta AWS SAM CLI in uscita. Sono disponibili i seguenti valori di opzione:

- `json`— I metadati e la risposta delle risorse vengono restituiti in una JSON struttura. La risposta contiene l'SDKoutput completo.
- `text`— I metadati vengono restituiti nella struttura del testo. La risposta della risorsa viene restituita nel formato di output della risorsa.

Di seguito è riportato un esempio di `json` output:

```
$ sam remote invoke --stack-name sam-app --output json

Invoking Lambda Function HelloWorldFunction



{
  "ResponseMetadata": {
    "RequestId": "3bdf9a30-776d-4a90-94a6-4cccc0fc7b41",
    "HTTPStatusCode": 200,
    "HTTPHeaders": {
      "date": "Mon, 19 Jun 2023 17:15:46 GMT",
      "content-type": "application/json",
      "content-length": "57",
      "connection": "keep-alive",
      "x-amzn-requestid": "3bdf9a30-776d-4a90-94a6-4cccc0fc7b41",
      "x-amzn-remapped-content-length": "0",
```

```
      "x-amz-executed-version": "$LATEST",
      "x-amz-log-result":
 "U1RBUlQgUmVxdWVzdElkOiAzYmRmOWEzMC03NzZkLTRhOTAtOTRhNi00Y2NjYzBmYzdiNDEgVmVyc2lvbjogJExBVEVTV
      "x-amzn-trace-id":
 "root=1-64908d42-17dab270273fcc6b527dd6b8;sampled=0;lineage=2301f8dc:0"
    },
    "RetryAttempts": 0
  },
  "StatusCode": 200,
  "LogResult":
 "U1RBUlQgUmVxdWVzdElkOiAzYmRmOWEzMC03NzZkLTRhOTAtOTRhNi00Y2NjYzBmYzdiNDEgVmVyc2lvbjogJExBVEVTV
  "ExecutedVersion": "$LATEST",
  "Payload": "{\"statusCode\":200,\"body\":\"{\\\"message\\\":\\\"hello world\\\"}\"}"
}%
```

Quando si specifica un `json` output, viene restituita l'intera risposta `stdout`. Di seguito è riportato un esempio:

```
$ sam remote invoke --stack-name sam-app --output json 1> stdout.log

Invoking Lambda Function HelloWorldFunction



$ cat stdout.log

{
  "ResponseMetadata": {
    "RequestId": "d30d280f-8188-4372-bc94-ce0f1603b6bb",
    "HTTPStatusCode": 200,
    "HTTPHeaders": {
      "date": "Mon, 19 Jun 2023 17:35:56 GMT",
      "content-type": "application/json",
      "content-length": "57",
      "connection": "keep-alive",
      "x-amzn-requestid": "d30d280f-8188-4372-bc94-ce0f1603b6bb",
      "x-amzn-remapped-content-length": "0",
      "x-amz-executed-version": "$LATEST",
      "x-amz-log-result":
 "U1RBUlQgUmVxdWVzdElkOiBkMzBkMjgwZi04MTg4LTQzNzItYmM5NC1jZTBmMTYwM2I2YmIgVmVyc2lvbjogJExBVEVTV
      "x-amzn-trace-id":
 "root=1-649091fc-771473c7778689627a6122b7;sampled=0;lineage=2301f8dc:0"
    },
```

```
    "RetryAttempts": 0
  },
  "StatusCode": 200,
  "LogResult":
 "U1RBUlQgUmVxdWVzdElkOiBkMzBkMjgwZi04MTg4LTQzNzItYmM5NC1jZTBmMTYwM2I2YmIgVmVyc2lvbjogJExBVEVTV
  "ExecutedVersion": "$LATEST",
  "Payload": "{\"statusCode\":200,\"body\":\"{\\\"message\\\":\\\"hello world\\\"}\"}"
}%
```

Di seguito è riportato un esempio di `text` output:

```
$ sam remote invoke --stack-name sam-app --output text

Invoking Lambda Function HelloWorldFunction



START RequestId: 4dbacc43-1ec6-47c2-982b-9dc4620144d6 Version: $LATEST
END RequestId: 4dbacc43-1ec6-47c2-982b-9dc4620144d6
REPORT RequestId: 4dbacc43-1ec6-47c2-982b-9dc4620144d6  Duration: 9.13 ms       Billed
 Duration: 10 ms  Memory Size: 128 MB     Max Memory Used: 67 MB  Init Duration: 165.50
 ms
{"statusCode":200,"body":"{\"message\":\"hello world\"}"}%
```

Quando si specifica un `text` output, viene restituito l'output del runtime della funzione Lambda (ad esempio, i log). `stderr` Il payload della funzione Lambda viene restituito a. `stdout` Di seguito è riportato un esempio:

```
$ sam remote invoke --stack-name sam-app --output text 2> stderr.log

{"statusCode":200,"body":"{\"message\":\"hello world\"}"}%



$ cat stderr.log

Invoking Lambda Function HelloWorldFunction
START RequestId: 82273c3b-aa3a-4d16-8f1c-1d2ad3ace891 Version: $LATEST
END RequestId: 82273c3b-aa3a-4d16-8f1c-1d2ad3ace891
REPORT RequestId: 82273c3b-aa3a-4d16-8f1c-1d2ad3ace891  Duration: 40.62 ms      Billed
 Duration: 41 ms  Memory Size: 128 MB     Max Memory Used: 68 MB

$ sam remote invoke --stack-name sam-app --output text 1> stdout.log
```

```
Invoking Lambda Function HelloWorldFunction


START RequestId: 74acaa9f-5b80-4a5c-b3b8-ffaccb84cbbd Version: $LATEST
END RequestId: 74acaa9f-5b80-4a5c-b3b8-ffaccb84cbbd
REPORT RequestId: 74acaa9f-5b80-4a5c-b3b8-ffaccb84cbbd  Duration: 2.31 ms       Billed
 Duration: 3 ms   Memory Size: 128 MB     Max Memory Used: 67 MB

$ cat stdout.log

{"statusCode":200,"body":"{\"message\":\"hello world\"}"}%
```

Personalizza i parametri Boto3

Infatti`sam remote invoke`, AWS SAM CLI utilizza l' AWS SDK for Python (Boto3) per interagire con le tue risorse nel cloud. È possibile utilizzare l'opzione per personalizzare i parametri. `--parameter` Boto3 Per un elenco dei parametri supportati che è possibile personalizzare, vedere`--parameter`.

Esempi

Invoca una funzione Lambda per convalidare i valori dei parametri e verificare le autorizzazioni:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --
parameter InvocationType="DryRun"
```

Usa l'**`--parameter`**opzione più volte in un unico comando per fornire più parametri:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --
parameter InvocationType="Event" --parameter LogType="None"
```

Altre opzioni

Per un elenco completo delle `sam remote invoke` opzioni, consultasam remote invoke.

Configura il file di configurazione del progetto

Per configurare `sam remote invoke` nel tuo file di configurazione, `remote_invoke` usalo nella tua tabella. Di seguito è riportato un esempio di `samconfig.toml` file che configura i valori predefiniti per il `sam remote invoke` comando.

```
...
version =0.1

[default]
...
[default.remote_invoke.parameters]
stack_name = "cloud-app"
event = '{"message": "Hello!"}'
```

Esempi

Per un esempio di base di utilizzo`sam remote invoke`, consulta [Testing AWS Lambda functions with AWS SAM remote](#) nel blog di AWS Compute.

Esempi di Kinesis Data Streams

Esempi di base

Invia un record di dati a un'applicazione Kinesis Data Streams da un file. L'applicazione Kinesis Data Streams viene identificata fornendo un ARN per l'ID della risorsa:

```
$ sam remote invoke arn:aws:kinesis:us-west-2:01234567890:stream/kinesis-example-KinesisStream-BgnLcAey4xUQ --event-file event.json
```

Invia un evento fornito dalla riga di comando a un'applicazione Kinesis Data Streams:

```
$ sam remote invoke KinesisStream --stack-name kinesis-example --event hello-world

Putting record to Kinesis data stream KinesisStream

Auto converting value 'hello-world' into JSON '"hello-world"'. If you don't want auto-conversion, please provide
a JSON string as event

{
  "ShardId": "shardId-000000000000",
  "SequenceNumber": "49646251411914806775980903986194508740483329854174920706"
}%
```

Ottieni l'ID fisico dell'applicazione Kinesis Data Streams. Quindi, fornisci un evento nella riga di comando:

```
$ sam list resources --stack-name kinesis-example --output json

[
  {
    "LogicalResourceId": "KinesisStream",
    "PhysicalResourceId": "kinesis-example-KinesisStream-ZgnLcQey4xUQ"
  }
]

$ sam remote invoke kinesis-example-KinesisStream-ZgnLcQey4xUQ --event hello

Putting record to Kinesis data stream KinesisStream

Auto converting value 'hello' into JSON '"hello"'. If you don't want auto-conversion,
 please provide a JSON
string as event

{
  "ShardId": "shardId-000000000000",
  "SequenceNumber": "49646251411914806775980904340716841045751814812900261890"
}%
```

Fornisci una stringa JSON nella riga di comando come evento:

```
$ sam remote invoke KinesisStream --stack-name kinesis-example --event '{"method":
 "GET", "body": ""}'

Putting record to Kinesis data stream KinesisStream

{
  "ShardId": "shardId-000000000000",
  "SequenceNumber": "49646251411914806775980904492868617924990209230536441858"
}%
```

Invia un evento vuoto all'applicazione Kinesis Data Streams:

```
$ sam remote invoke KinesisStream --stack-name kinesis-example

Putting record to Kinesis data stream KinesisStream

{
  "ShardId": "shardId-000000000000",
```

```
  "SequenceNumber": "49646251411914806775980904866469008589597168190416224258"
}%
```

Restituisci la AWS SAM CLI risposta in formato JSON:

```
$ sam remote invoke KinesisStream --stack-name kinesis-example --event '{"hello":
 "world"}' --output json

Putting record to Kinesis data stream KinesisStream

{
  "ShardId": "shardId-000000000000",
  "SequenceNumber": "49646251411914806775980905078409420803696667195489648642",
  "ResponseMetadata": {
    "RequestId": "ebbbd307-3e9f-4431-b67c-f0715e9e353e",
    "HTTPStatusCode": 200,
    "HTTPHeaders": {
      "x-amzn-requestid": "ebbbd307-3e9f-4431-b67c-f0715e9e353e",
      "x-amz-id-2": "Q3yBcgTwtPaQTV26IKclbECmZikUYOzKY+CzcxA84ZHgCkc5T2N/
ITWg6RPOQcWw8Gn0tNPcEJBEHyVVqboJAPgCritqsvCu",
      "date": "Thu, 09 Nov 2023 18:13:10 GMT",
      "content-type": "application/x-amz-json-1.1",
      "content-length": "110"
    },
    "RetryAttempts": 0
  }
}%
```

Restituisce l'output JSON su stdout:

```
$ sam remote invoke KinesisStream --stack-name kinesis-example --event '{"hello":
 "world"}' --output json 1> stdout.log

Putting record to Kinesis data stream KinesisStream


$ cat stdout.log
{
  "ShardId": "shardId-000000000000",
  "SequenceNumber": "49646251411914806775980906397777867595039988349006774274",
  "ResponseMetadata": {
    "RequestId": "f4290006-d84b-b1cd-a9ee-28306eeb2939",
    "HTTPStatusCode": 200,
```

```
    "HTTPHeaders": {
      "x-amzn-requestid": "f4290006-d84b-b1cd-a9ee-28306eeb2939",
      "x-amz-id-2": "npCqz
+IBKpoL4sQ1ClbUmxuJlbeA24Fx1UgpIrS6mm2NoIeV2qdZSN5AhNurdssykXajBrXaC9anMhj2eG/h7Hnbf
+bPuotU",
      "date": "Thu, 09 Nov 2023 18:33:26 GMT",
      "content-type": "application/x-amz-json-1.1",
      "content-length": "110"
    },
    "RetryAttempts": 0
  }
}%
```

Esempi Lambda

Esempi di base

Invoca una funzione Lambda fornendo l'ARN come ID di risorsa:

```
$ sam remote invoke arn:aws:lambda:us-west-2:012345678910:function:sam-app-
HelloWorldFunction-ohRFEn2RuAvp
```

Invoca una funzione Lambda fornendo l'ID logico come ID di risorsa:

È inoltre necessario fornire il nome dello CloudFormation stack utilizzando l'opzione. `--stack-name` Di seguito è riportato un esempio:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app
```

Se l'applicazione contiene una sola funzione Lambda, non è necessario specificarne l'ID logico. Puoi fornire solo l'`--stack-name`opzione. Di seguito è riportato un esempio:

```
$ sam remote invoke --stack-name sam-app
```

Invoca una funzione Lambda fornendo l'ID fisico come ID della risorsa:

L'ID fisico viene creato quando si distribuisce utilizzando. CloudFormation

```
$ sam remote invoke sam-app-HelloWorldFunction-TZvxQRFNv0k4
```

Invoca una funzione Lambda di uno stack secondario:

Per questo esempio, la nostra applicazione contiene la seguente struttura di directory:

```
lambda-example
### childstack
#   ### function
#   #   ### __init__.py
#   #   ### app.py
#   #   ### requirements.txt
#   ### template.yaml
### events
#   ### event.json
### samconfig.toml
### template.yaml
```

Per richiamare la `childstack` nostra funzione Lambda, eseguiamo quanto segue:

```
$ sam remote invoke ChildStack/HelloWorldFunction --stack-name lambda-example

Invoking Lambda Function HelloWorldFunction

START RequestId: 207a864b-e67c-4307-8478-365b004d4bcd Version: $LATEST
END RequestId: 207a864b-e67c-4307-8478-365b004d4bcd
REPORT RequestId: 207a864b-e67c-4307-8478-365b004d4bcd  Duration: 1.27 ms       Billed
 Duration: 2 ms   Memory Size: 128 MB     Max Memory Used: 36 MB  Init Duration: 111.07
 ms
{"statusCode": 200, "body": "{\"message\": \"Hello\", \"received_event\": {}}"}%
```

Richiama una funzione Lambda configurata per lo streaming di risposte

In questo esempio, utilizziamo il AWS SAMCLI per inizializzare una nuova applicazione serverless che contiene una funzione Lambda configurata per lo streaming della risposta. Implementiamo la nostra applicazione Cloud AWS e la utilizziamo `sam remote invoke` per interagire con la nostra funzione nel cloud.

Iniziamo eseguendo il `sam init` comando per creare una nuova applicazione serverless. Selezioniamo il modello di avvio rapido Lambda Response Streaming e assegniamo un nome alla nostra applicazione. lambda-streaming-nodejs-app

```
$ sam init

 You can preselect a particular runtime or package type when using the `sam init`
 experience.
```

```
 Call `sam init --help` to learn more.

 Which template source would you like to use?
        1 - AWS Quick Start Templates
        2 - Custom Template Location
 Choice: 1

 Choose an AWS Quick Start application template
        1 - Hello World Example
        ...
        9 - Lambda Response Streaming
        ...
        15 - Machine Learning
 Template: 9

 Which runtime would you like to use?
        1 - go (provided.al2)
        2 - nodejs18.x
        3 - nodejs16.x
 Runtime: 2

 Based on your selections, the only Package type available is Zip.
 We will proceed to selecting the Package type as Zip.

 Based on your selections, the only dependency manager available is npm.
 We will proceed copying the template using npm.

 Would you like to enable X-Ray tracing on the function(s) in your application?  [y/
N]: ENTER

 Would you like to enable monitoring using CloudWatch Application Insights?
 For more info, please view https://docs.aws.amazon.com/AmazonCloudWatch/latest/
monitoring/cloudwatch-application-insights.html [y/N]: ENTER

 Project name [sam-app]: lambda-streaming-nodejs-app

     -----------------------
     Generating application:
     -----------------------
     Name: lambda-streaming-nodejs-app
     Runtime: nodejs18.x
     Architectures: x86_64
     Dependency Manager: npm
     Application Template: response-streaming
```

```
     Output Directory: .
     Configuration file: lambda-streaming-nodejs-app/samconfig.toml

     Next steps can be found in the README file at lambda-streaming-nodejs-app/
README.md


 Commands you can use next
 =========================
 [*] Create pipeline: cd lambda-streaming-nodejs-app && sam pipeline init --bootstrap
 [*] Validate SAM template: cd lambda-streaming-nodejs-app && sam validate
 [*] Test Function in the Cloud: cd lambda-streaming-nodejs-app && sam sync --stack-
name {stack-name} --watch
```

AWS SAMCLICrea il nostro progetto con la seguente struttura:

```
lambda-streaming-nodejs-app
 ### README.md
 ### __tests__
 #   ### unit
 #       ### index.test.js
 ### package.json
 ### samconfig.toml
 ### src
 #   ### index.js
 ### template.yaml
```

Di seguito è riportato un esempio del nostro codice di funzione Lambda:

```
exports.handler = awslambda.streamifyResponse(
   async (event, responseStream, context) => {
     const httpResponseMetadata = {
       statusCode: 200,
       headers: {
         "Content-Type": "text/html",
         "X-Custom-Header": "Example-Custom-Header"
       }
     };

     responseStream = awslambda.HttpResponseStream.from(responseStream,
 httpResponseMetadata);
     // It's recommended to use a `pipeline` over the `write` method for more complex
 use cases.
```

```
    // Learn more: https://docs.aws.amazon.com/lambda/latest/dg/configuration-
response-streaming.html
    responseStream.write("<html>");
    responseStream.write("<p>First write!</p>");

    responseStream.write("<h1>Streaming h1</h1>");
    await new Promise(r => setTimeout(r, 1000));
    responseStream.write("<h2>Streaming h2</h2>");
    await new Promise(r => setTimeout(r, 1000));
    responseStream.write("<h3>Streaming h3</h3>");
    await new Promise(r => setTimeout(r, 1000));

    // Long strings will be streamed
    const loremIpsum1 = "Lorem ipsum dolor sit amet, consectetur adipiscing elit.
 Quisque vitae mi tincidunt tellus ultricies dignissim id et diam. Morbi pharetra eu
 nisi et finibus. Vivamus diam nulla, vulputate et nisl cursus, pellentesque vehicula
 libero. Cras imperdiet lorem ante, non posuere dolor sollicitudin a. Vestibulum ipsum
 lacus, blandit nec augue id, lobortis dictum urna. Vestibulum ante ipsum primis in
 faucibus orci luctus et ultrices posuere cubilia curae; Morbi auctor orci eget tellus
 aliquam, non maximus massa porta. In diam ante, pulvinar aliquam nisl non, elementum
 hendrerit sapien. Vestibulum massa nunc, mattis non congue vitae, placerat in quam.
 Nam vulputate lectus metus, et dignissim erat varius a.";
    responseStream.write(`<p>${loremIpsum1}</p>`);
    await new Promise(r => setTimeout(r, 1000));

    responseStream.write("<p>DONE!</p>");
    responseStream.write("</html>");
    responseStream.end();
  }
);
```

Di seguito è riportato un esempio del nostro `template.yaml` file. Lo streaming di risposta per la nostra funzione Lambda è configurato utilizzando la `FunctionUrlConfig` proprietà.

```
AWSTemplateFormatVersion: '2010-09-09'
 Transform: AWS::Serverless-2016-10-31

 Description: >
   Sample SAM Template for lambda-streaming-nodejs-app

 Resources:
   StreamingFunction:
     Type: AWS::Serverless::Function
```

```
    Properties:
      CodeUri: src/
      Handler: index.handler
      Runtime: nodejs18.x
      Architectures:
        - x86_64
      Timeout: 10
      FunctionUrlConfig:
        AuthType: AWS_IAM
        InvokeMode: RESPONSE_STREAM

Outputs:
  StreamingFunction:
    Description: "Streaming Lambda Function ARN"
    Value: !GetAtt StreamingFunction.Arn
  StreamingFunctionURL:
    Description: "Streaming Lambda Function URL"
    Value: !GetAtt StreamingFunctionUrl.FunctionUrl
```

In genere, è possibile utilizzare `sam build` e `sam deploy --guided` creare e distribuire un'applicazione di produzione. In questo esempio, assumeremo un ambiente di sviluppo e utilizzeremo il `sam sync` comando per creare e distribuire la nostra applicazione.

> **Note**
>
> Il `sam sync` comando è consigliato per ambienti di sviluppo. Per ulteriori informazioni, consulta Introduzione all'uso sam sync della sincronizzazione con Cloud AWS.

Prima dell'esecuzione`sam sync`, verifichiamo che il nostro progetto sia configurato correttamente nel nostro `samconfig.toml` file. Soprattutto, verifichiamo i valori di `stack_name` e`watch`. Con questi valori specificati nel nostro file di configurazione, non è necessario fornirli nella riga di comando.

```
version = 0.1

 [default]
 [default.global.parameters]
 stack_name = "lambda-streaming-nodejs-app"

 [default.build.parameters]
 cached = true
```

```
parallel = true

[default.validate.parameters]
lint = true

[default.deploy.parameters]
capabilities = "CAPABILITY_IAM"
confirm_changeset = true
resolve_s3 = true
s3_prefix = "lambda-streaming-nodejs-app"
region = "us-west-2"
image_repositories = []

[default.package.parameters]
resolve_s3 = true

[default.sync.parameters]
watch = true

[default.local_start_api.parameters]
warm_containers = "EAGER"

[default.local_start_lambda.parameters]
warm_containers = "EAGER"
```

Successivamente, eseguiamo `sam sync` la creazione e la distribuzione della nostra applicazione. Poiché l'`--watch`opzione è configurata nel nostro file di configurazione, AWS SAMCLI creerà la nostra applicazione, distribuirà la nostra applicazione e controllerà le modifiche.

```
$ sam sync

The SAM CLI will use the AWS Lambda, Amazon API Gateway, and AWS StepFunctions APIs to
upload your code
without

performing a CloudFormation deployment. This will cause drift in your CloudFormation
stack.
**The sync command should only be used against a development stack**.



Queued infra sync. Waiting for in progress code syncs to complete...
```

```
 Starting infra sync.

 Building codeuri:

 /Users/.../lambda-streaming-nodejs-app/src runtime: nodejs18.x metadata: {}
 architecture: x86_64 functions: StreamingFunction
 package.json file not found. Continuing the build without dependencies.

 Running NodejsNpmBuilder:CopySource


 Build Succeeded

 Successfully packaged artifacts and wrote output template to file /var/
folders/45/5ct135bx3fn2551_ptl5g6_80000gr/T/tmpavrzdhgp.
 Execute the following command to deploy the packaged template
 sam deploy --template-file /var/folders/45/5ct135bx3fn2551_ptl5g6_80000gr/T/
tmpavrzdhgp --stack-name <YOUR STACK NAME>


         Deploying with following values
         ===============================
         Stack name                   : lambda-streaming-nodejs-app
         Region                       : us-west-2
         Disable rollback             : False
         Deployment s3 bucket         : aws-sam-cli-managed-default-samclisam-s3-demo-
bucket-1a4x26zbcdkqr
         Capabilities                 : ["CAPABILITY_NAMED_IAM",
 "CAPABILITY_AUTO_EXPAND"]
         Parameter overrides          : {}
         Signing Profiles             : null

 Initiating deployment
 =====================


 2023-06-20 12:11:16 - Waiting for stack create/update to complete

 CloudFormation events from stack operations (refresh every 0.5 seconds)
 ---------------------------------------------------------------------------------------------------
 ResourceStatus            ResourceType              LogicalResourceId
 ResourceStatusReason
 ---------------------------------------------------------------------------------------------------
```

```
CREATE_IN_PROGRESS        AWS::CloudFormation::St   lambda-streaming-
Transformation
                          ack                       nodejs-app
succeeded
CREATE_IN_PROGRESS        AWS::IAM::Role            StreamingFunctionRole     -

CREATE_IN_PROGRESS        AWS::CloudFormation::St   AwsSamAutoDependencyLay   -

                          ack                       erNestedStack

CREATE_IN_PROGRESS        AWS::IAM::Role            StreamingFunctionRole     Resource
creation

Initiated
CREATE_IN_PROGRESS        AWS::CloudFormation::St   AwsSamAutoDependencyLay   Resource
creation
                          ack                       erNestedStack
Initiated
CREATE_COMPLETE           AWS::IAM::Role            StreamingFunctionRole     -

CREATE_COMPLETE           AWS::CloudFormation::St   AwsSamAutoDependencyLay   -

                          ack                       erNestedStack

CREATE_IN_PROGRESS        AWS::Lambda::Function     StreamingFunction         -

CREATE_IN_PROGRESS        AWS::Lambda::Function     StreamingFunction         Resource
creation

Initiated
CREATE_COMPLETE           AWS::Lambda::Function     StreamingFunction         -

CREATE_IN_PROGRESS        AWS::Lambda::Url          StreamingFunctionUrl      -

CREATE_IN_PROGRESS        AWS::Lambda::Url          StreamingFunctionUrl      Resource
creation

Initiated
CREATE_COMPLETE           AWS::Lambda::Url          StreamingFunctionUrl      -

CREATE_COMPLETE           AWS::CloudFormation::St   lambda-streaming-         -

                          ack                       nodejs-app
```

```
 -----------------------------------------------------------------------------------------------------

 CloudFormation outputs from deployed stack
 -----------------------------------------------------------------------------------------------------
 Outputs

 -----------------------------------------------------------------------------------------------------
 Key                 StreamingFunction

 Description         Streaming Lambda Function ARN

 Value               arn:aws:lambda:us-west-2:012345678910:function:lambda-streaming-
nodejs-app-
 StreamingFunction-gUmhO833A0vZ


 Key                 StreamingFunctionURL

 Description         Streaming Lambda Function URL

 Value               https://wxgkcc2dyntgtrwhf2dgdcvylu0rnnof.lambda-url.us-
west-2.on.aws/
 -----------------------------------------------------------------------------------------------------



 Stack creation succeeded. Sync infra completed.



 Infra sync completed.
```

Ora che la nostra funzione è implementata nel cloud, possiamo usarla `sam remote invoke` per interagire con la nostra funzione. Rileva AWS SAMCLI automaticamente che la nostra funzione è configurata per lo streaming di risposte e inizia immediatamente a emettere una risposta in streaming della nostra funzione in tempo reale.

```
$ sam remote invoke StreamingFunction

 Invoking Lambda Function StreamingFunction

 {"statusCode":200,"headers":{"Content-Type":"text/html","X-Custom-Header":"Example-
Custom-Header"}}<html><p>First write!</p><h1>Streaming h1</h1><h2>Streaming h2</
```

```
h2><h3>Streaming h3</h3><p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.
 Quisque vitae mi tincidunt tellus ultricies dignissim id et diam. Morbi pharetra eu
 nisi et finibus. Vivamus diam nulla, vulputate et nisl cursus, pellentesque vehicula
 libero. Cras imperdiet lorem ante, non posuere dolor sollicitudin a. Vestibulum ipsum
 lacus, blandit nec augue id, lobortis dictum urna. Vestibulum ante ipsum primis in
 faucibus orci luctus et ultrices posuere cubilia curae; Morbi auctor orci eget tellus
 aliquam, non maximus massa porta. In diam ante, pulvinar aliquam nisl non, elementum
 hendrerit sapien. Vestibulum massa nunc, mattis non congue vitae, placerat in quam.
 Nam vulputate lectus metus, et dignissim erat varius a.</p><p>DONE!</p></html>START
 RequestId: 1e4cdf04-60de-4769-b3a2-c1481982deb4 Version: $LATEST
 END RequestId: 1e4cdf04-60de-4769-b3a2-c1481982deb4
 REPORT RequestId: 1e4cdf04-60de-4769-b3a2-c1481982deb4  Duration: 4088.66 ms
 Billed Duration: 4089 ms        Memory Size: 128 MB     Max Memory Used: 68 MB  Init
 Duration: 168.45 ms
```

Quando modifichiamo il codice della nostra funzione, rileva e AWS SAMCLI implementa immediatamente le nostre modifiche. Ecco un esempio dell' AWS SAMCLIoutput dopo aver apportato modifiche al nostro codice di funzione:

```
Syncing Lambda Function StreamingFunction...

 Building codeuri:

 /Users/.../lambda-streaming-nodejs-app/src runtime: nodejs18.x metadata: {}
 architecture:
 x86_64 functions: StreamingFunction

 package.json file not found. Continuing the build without dependencies.

 Running NodejsNpmBuilder:CopySource

 Finished syncing Lambda Function StreamingFunction.

 Syncing Layer StreamingFunctione9cfe924DepLayer...

 SyncFlow [Layer StreamingFunctione9cfe924DepLayer]: Skipping resource update as the

 content didn't change

 Finished syncing Layer StreamingFunctione9cfe924DepLayer.
```

Ora possiamo `sam remote invoke` riutilizzarlo per interagire con la nostra funzione nel cloud e testare le nostre modifiche.

Esempi SQS

Esempi di base

Richiama una coda Amazon SQS fornendo l'ARN come ID di risorsa:

```
$ sam remote invoke arn:aws:sqs:us-west-2:01234567890:sqs-example-4DonhBsjsW1b --
event '{"hello": "world"}' --output json

Sending message to SQS queue MySqsQueue

{
  "MD5OfMessageBody": "49dfdd54b01cbcd2d2ab5e9e5ee6b9b9",
  "MessageId": "4f464cdd-15ef-4b57-bd72-3ad225d80adc",
  "ResponseMetadata": {
    "RequestId": "95d39377-8323-5ef0-9223-ceb198bd09bd",
    "HTTPStatusCode": 200,
    "HTTPHeaders": {
      "x-amzn-requestid": "95d39377-8323-5ef0-9223-ceb198bd09bd",
      "date": "Wed, 08 Nov 2023 23:27:26 GMT",
      "content-type": "application/x-amz-json-1.0",
      "content-length": "106",
      "connection": "keep-alive"
    },
    "RetryAttempts": 0
  }
}%
```

Esempi di Step Functions

Esempi di base

Richiama una macchina a stati fornendo il suo ID fisico come ID di risorsa:

Innanzitutto, `sam list resources` per ottenere il nostro ID fisico, utilizziamo:

```
$ sam list resources --stack-name state-machine-example --output json

[
```

```
  {
    "LogicalResourceId": "HelloWorldStateMachine",
    "PhysicalResourceId": "arn:aws:states:us-
west-2:513423067560:stateMachine:HelloWorldStateMachine-z69tFEUx0F66"
  },
  {
    "LogicalResourceId": "HelloWorldStateMachineRole",
    "PhysicalResourceId": "simple-state-machine-HelloWorldStateMachineRole-
PduA0BDGuFXw"
  }
]
```

Successivamente, invochiamo la nostra macchina a stati utilizzando l'ID fisico come ID di risorsa. Passiamo un evento alla riga di comando con l'`--event`opzione:

```
$ sam remote invoke arn:aws:states:us-
west-2:01234567890:stateMachine:HelloWorldStateMachine-z69tFEUx0F66 --
event '{"is_developer": true}'

Invoking Step Function arn:aws:states:us-
west-2:01234567890:stateMachine:HelloWorldStateMachine-z69tFEUx0F66
"Hello Developer World"%
```

Invoca una macchina a stati passando un evento vuoto:

```
$ sam remote invoke HelloWorldStateMachine --stack-name state-machine-example

Invoking Step Function HelloWorldStateMachine

"Hello World"%
```

Link correlati

Per la documentazione relativa `sam remote invoke` e all'utilizzo di AWS SAMCLI, vedere quanto segue:

- sam remote invoke
- AWS SAMCLIrisoluzione dei problemi

Automatizza i test di integrazione locali con AWS SAM

Sebbene sia possibile utilizzarlo Introduzione ai test con sam local invoke per testare manualmente il codice, consente AWS SAM anche di testarlo utilizzando test di integrazione automatizzati. I test di integrazione consentono di rilevare i problemi nelle prime fasi del ciclo di sviluppo, migliorare la qualità del codice e risparmiare tempo riducendo al contempo i costi.

Per creare test di integrazione automatizzati AWS SAM, devi prima eseguire test sulle funzioni Lambda locali prima di distribuirli nel Cloud. AWS Il Introduzione ai test con sam local start-lambda comando avvia un endpoint locale che emula l'endpoint Invoke Lambda. Puoi richiamarlo dai tuoi test automatici. Poiché questo endpoint emula l'endpoint Lambda invoke, puoi scrivere i test una volta e poi eseguirli (senza alcuna modifica) sulla funzione Lambda locale o su una funzione Lambda distribuita. Puoi anche eseguire gli stessi test su uno stack distribuito nella tua pipeline. AWS SAM CI/CD

Ecco come funziona il processo:

1. Avvia l'endpoint Lambda locale.

   Avvia l'endpoint Lambda locale eseguendo il seguente comando nella directory che contiene il modello: AWS SAM

   ```
   sam local start-lambda
   ```

   Questo comando avvia un endpoint locale in cui viene emulato`http://127.0.0.1:3001`. AWS Lambda Puoi eseguire test automatici su questo endpoint Lambda locale. Quando richiami questo endpoint utilizzando AWS CLI o SDK, esegue localmente la funzione Lambda specificata nella richiesta e restituisce una risposta.
2. Esegui un test di integrazione sull'endpoint Lambda locale.

   Nel test di integrazione, puoi utilizzare l' AWS SDK per richiamare la funzione Lambda con i dati di test, attendere la risposta e verificare che la risposta sia quella prevista. Per eseguire il test di integrazione localmente, devi configurare l' AWS SDK per inviare una chiamata API Lambda Invoke per richiamare l'endpoint Lambda locale avviato nel passaggio precedente.

   Quello che segue è un esempio di Python (i quattro AWS SDKs linguaggi hanno configurazioni simili):

   ```
   import boto3
   ```

```
import botocore

# Set "running_locally" flag if you are running the integration test locally
running_locally = True

if running_locally:

    # Create Lambda SDK client to connect to appropriate Lambda endpoint
    lambda_client = boto3.client('lambda',
        region_name="us-west-2",
        endpoint_url="http://127.0.0.1:3001",
        use_ssl=False,
        verify=False,
        config=botocore.client.Config(
            signature_version=botocore.UNSIGNED,
            read_timeout=15,
            retries={'max_attempts': 0},
        )
    )
else:
    lambda_client = boto3.client('lambda')


# Invoke your Lambda function as you normally usually do. The function will run
# locally if it is configured to do so
response = lambda_client.invoke(FunctionName="HelloWorldFunction")

# Verify the response
assert response == "Hello World"
```

È possibile utilizzare questo codice per testare le funzioni Lambda distribuite impostando su`running_locally`. `False` Questo configura l' AWS SDK a cui connettersi AWS Lambda nel cloud. AWS

Genera payload di eventi di esempio con AWS SAM

Per testare le tue funzioni Lambda, puoi generare e personalizzare payload di eventi di esempio che imitano i dati che le tue funzioni Lambda riceveranno quando vengono attivate da altri servizi. AWS Ciò include servizi come API Gateway CloudFormation, Amazon S3 e altro ancora.

La generazione di payload di eventi di esempio consente di testare il comportamento della funzione Lambda con una varietà di input diversi senza dover lavorare in un ambiente live. Questo approccio consente inoltre di risparmiare tempo rispetto alla creazione manuale di esempi di eventi di AWS servizio per testare le funzioni.

Per l'elenco completo dei servizi per i quali è possibile generare payload di eventi di esempio, utilizzate questo comando:

```
sam local generate-event --help
```

Per l'elenco delle opzioni che puoi usare per un particolare servizio, usa questo comando:

```
sam local generate-event [SERVICE] --help
```

Esempi:

```
#Generates the event from S3 when a new object is created
sam local generate-event s3 put

# Generates the event from S3 when an object is deleted
sam local generate-event s3 delete
```

Test e debug di funzioni durevoli

Il test e il debug delle funzioni durevoli a livello locale funzionano in modo simile alle normali funzioni Lambda, con supporto automatico per checkpoint e replay. Questa guida illustra gli scenari di test e le tecniche di risoluzione dei problemi più comuni.

Flusso di lavoro di test locale

Quando si testano funzioni durevoli a livello locale, il flusso di lavoro è diverso dalle normali funzioni Lambda:

Flusso di lavoro durevole per la verifica

1. Crea la tua applicazione:

   ```
   $ sam build
   ```

2. Invoca la tua funzione duratura:

```
$ sam local invoke MyDurableFunction --durable-execution-name test
```

3. Controlla la cronologia delle esecuzioni, se necessario:

```
$ sam local execution history execution-id
```

Scenari di test comuni

Test del comportamento dei checkpoint

Per verificare che la tua funzione funzioni correttamente, i checkpoint indicano:

```
# Example Python durable function
def handler(event, context):
    # This will create a checkpoint
    context.wait(10)  # Wait 10 seconds

    # Function resumes here after wait
    return {"message": "Function resumed after wait"}
```

Quando si richiama questa funzione localmente, il periodo di attesa viene gestito automaticamente.

Scenari di callback di test

Per le funzioni che attendono callback esterni:

1. Avvia la tua funzione duratura che attende una richiamata
2. In un altro terminale, risolvi il callback:

```
$ sam local callback succeed callback-id
```

3. Osserva che la funzione riprende l'esecuzione

Risoluzione dei problemi

Funzione durevole che non viene eseguita correttamente

Problema: la funzione non si comporta come una funzione durevole.

Soluzioni:

- Verifica che `DurableConfig` sia impostato nel tuo modello SAM
- Assicurati che il codice della funzione utilizzi metodi SDK per funzioni durevoli (ad esempio,`context.wait()`)
- Verifica di utilizzare un runtime supportato (TypeScript, JavaScript, Python)

Impossibile recuperare la cronologia delle esecuzioni

Problema: il `local execution history` comando non restituisce risultati.

Soluzioni:

- Verifica che l'ID di esecuzione sia corretto
- Verifica che la funzione sia stata richiamata almeno una volta

I comandi di callback non funzionano

Problema: i comandi di callback non risolvono le operazioni in sospeso.

Soluzioni:

- Verifica che l'ID di callback sia corretto
- Assicurati che la funzione sia effettivamente in attesa di una richiamata
- Verifica di utilizzare la sintassi corretta del comando di callback

Suggerimenti per il debug

- Usa la cronologia di esecuzione: esamina la cronologia delle esecuzioni per comprendere il flusso della tua funzione duratura
- Esegui test in modo incrementale: inizia con semplici operazioni di attesa prima di aggiungere una logica complessa
- Usa la registrazione dettagliata: abilita la registrazione dettagliata per tracciare il flusso di esecuzione

Ulteriori informazioni

Per ulteriori informazioni su test e debug, consulta:

- Introduzione ai test con sam local invoke- Documentazione locale sulle invoke
- sam local execution history- Cronologia delle esecuzioni

Esegui il debug della tua applicazione serverless con AWS SAM

Dopo aver testato l'applicazione, sarete pronti a eseguire il debug di eventuali problemi riscontrati. Con l'interfaccia a riga di AWS SAM comando (CLI), puoi testare ed eseguire il debug localmente della tua applicazione serverless prima di caricarla sul Cloud. AWS Il debug dell'applicazione identifica e corregge problemi o errori nell'applicazione.

È possibile utilizzarlo AWS SAM per eseguire il debug dettagliato, ovvero un metodo per eseguire il codice una riga o un'istruzione alla volta. Quando richiami localmente una funzione Lambda in modalità debug all'interno AWS SAMCLI di, puoi quindi collegarle un debugger. Con il debugger, puoi scorrere il codice riga per riga, vedere i valori di diverse variabili e risolvere i problemi nello stesso modo in cui faresti per qualsiasi altra applicazione. Puoi verificare se l'applicazione si comporta come previsto, eseguire il debug di eventuali problemi e risolvere eventuali problemi, prima di completare le fasi di imballaggio e distribuzione dell'applicazione.

> **Note**
>
> Se l'applicazione include uno o più livelli, quando si esegue e si esegue il debug dell'applicazione localmente, il pacchetto layers viene scaricato e memorizzato nella cache sull'host locale. Per ulteriori informazioni, consulta Come i livelli vengono memorizzati nella cache locale.

Argomenti

- Funzioni di debug locali con AWS SAM
- Passa più argomenti di runtime durante il debug con AWS SAM
- Convalida le tue AWS SAM applicazioni con CloudFormation Linter

Funzioni di debug locali con AWS SAM

È possibile utilizzare AWS SAM una varietà di AWS toolkit e debugger per testare ed eseguire il debug delle applicazioni serverless a livello locale. Il debug graduale delle funzioni Lambda consente di identificare e risolvere i problemi dell'applicazione una riga o un'istruzione alla volta nell'ambiente locale.

Alcuni dei modi in cui è possibile eseguire il debug locale graduale includono l'impostazione dei punti di interruzione, l'ispezione delle variabili e l'esecuzione del codice di funzione una riga alla volta. Il debug locale step-through restringe il ciclo di feedback, consentendoti di individuare e risolvere i problemi che potresti riscontrare nel cloud.

Puoi usare AWS Toolkit per eseguire il debug ed eseguirlo anche in modalità debug. AWS SAM Per informazioni dettagliate, consulta gli argomenti di questa sezione.

Utilizzo dei AWS Toolkit

AWS I toolkit sono plugin IDE (Integrated Development Environment) che offrono la possibilità di eseguire molte attività di debug comuni, come l'impostazione dei punti di interruzione, l'ispezione delle variabili e l'esecuzione del codice di funzione una riga alla volta. AWS I toolkit semplificano lo sviluppo, il debug e la distribuzione di applicazioni serverless create utilizzando. AWS SAM Offrono un'esperienza per la creazione, il test, il debug, la distribuzione e l'invocazione di funzioni Lambda integrate nel tuo IDE.

Per ulteriori informazioni sui AWS Toolkit che puoi utilizzare, consulta quanto segue: AWS SAM

- AWS Toolkit for Visual Studio Code
- AWS Cloud9
- AWS Toolkit for JetBrains

Esistono diversi AWS toolkit che funzionano con diverse combinazioni e tempi di esecuzione. IDEs La tabella seguente elenca le IDE/runtime combinazioni comuni che supportano il debug dettagliato delle applicazioni: AWS SAM

IDE	Runtime	AWS Toolkit	Istruzioni per il debug dettagliato
Visual Studio Code	• Node.js • Python • .NET • Java • Go	AWS Toolkit for Visual Studio Code	Utilizzo della Guida per l'utente Applicazione Serverless AWSAWS Toolkit for Visual Studio Code

IDE	Runtime	AWS Toolkit	Istruzioni per il debug dettagliato
AWS Cloud9	• Node.js • Python	AWS Cloud9[1], con AWS Toolkit abilitato	Utilizzo di applicazioni AWS serverless utilizzando il AWS Toolkit nella Guida per l'AWS Cloud9 utente.
WebStorm	Node.js	AWS Toolkit for JetBrains[2]	Esecuzione (richiamo) o debug di una funzione locale in AWS Toolkit for JetBrains
PyCharm	Python	AWS Toolkit for JetBrains[2]	Esecuzione (richiamo) o debug di una funzione locale in AWS Toolkit for JetBrains
Rider	.NET	AWS Toolkit for JetBrains[2]	Esecuzione (richiamo) o debug di una funzione locale in AWS Toolkit for JetBrains
IntelliJ	Java	AWS Toolkit for JetBrains[2]	Esecuzione (richiamo) o debug di una funzione locale in AWS Toolkit for JetBrains

IDE	Runtime	AWS Toolkit	Istruzioni per il debug dettagliato
GoLand	Go	AWS Toolkit for JetBrains[2]	Esecuzione (richiamo) o debug di una funzione locale in AWS Toolkit for JetBrains

Note:

1. Per poter eseguire il debug delle AWS SAM applicazioni, il Toolkit AWS Cloud9 deve essere abilitato. AWS Per ulteriori informazioni, vedere Enabling the AWS Toolkit nella Guida per l'utente.AWS Cloud9
2. Per utilizzare le AWS SAM applicazioni AWS Toolkit for JetBrains di debug dettagliate, è necessario innanzitutto installarle e configurarle seguendo le istruzioni disponibili in Installazione di. AWS Toolkit for JetBrainsAWS Toolkit for JetBrains

Esecuzione AWS SAM locale in modalità debug

Oltre all'integrazione con AWS Toolkits, puoi anche eseguirlo AWS SAM in modalità debug per collegarlo a debugger di terze parti come ptvsd o delve.

Per eseguire in modalità debug, usa i comandi o AWS SAM con l'opzione or. sam local invoke sam local start-api `--debug-port -d`

Esempio:

```
# Invoke a function locally in debug mode on port 5858
sam local invoke -d 5858 <function logical id>

# Start local API Gateway in debug mode on port 5858
sam local start-api -d 5858
```

> **Note**
>
> Se la utilizzi`sam local start-api`, l'istanza API Gateway locale espone tutte le tue funzioni Lambda. Tuttavia, poiché è possibile specificare una singola porta di debug, è

possibile eseguire il debug di una sola funzione alla volta. È necessario chiamare l'API prima del collegamento alla porta AWS SAMCLI, il che consente al debugger di connettersi.

Passa più argomenti di runtime durante il debug con AWS SAM

Puoi scegliere di passare argomenti di runtime aggiuntivi AWS SAM per esaminare i problemi e risolvere le variabili in modo più efficace. In questo modo si ottengono maggiore controllo e flessibilità al processo di debug, il che può aiutarvi con configurazioni e ambienti di runtime personalizzati.

Per passare argomenti di runtime aggiuntivi durante il debug della funzione, utilizzate la variabile di ambiente. `DEBUGGER_ARGS` Questa operazione passa una stringa di argomenti direttamente al comando run che AWS SAMCLI utilizza per avviare la funzione.

Ad esempio, se vuoi caricare un debugger come i KPdb durante l'esecuzione della tua funzione Python, puoi passare quanto segue come. `DEBUGGER_ARGS: -m ikpdb --ikpdb-port=5858 --ikpdb-working-directory=/var/task/ --ikpdb-client-working-directory=/myApp --ikpdb-address=0.0.0.0` Questo caricherebbe i KPdb in fase di esecuzione con gli altri argomenti che hai specificato.

In questo caso, il tuo AWS SAMCLI comando completo sarebbe:

```
DEBUGGER_ARGS="-m ikpdb --ikpdb-port=5858 --ikpdb-working-directory=/var/task/ --ikpdb-
client-working-directory=/myApp --ikpdb-address=0.0.0.0" echo {} | sam local invoke -d
 5858 myFunction
```

È possibile passare argomenti del debugger alle funzioni di tutti i runtime.

Convalida le tue AWS SAM applicazioni con CloudFormation Linter

AWS CloudFormation Linter (cfn-lint) è uno strumento open source che puoi utilizzare per eseguire una convalida dettagliata dei tuoi modelli. CloudFormation CFN-lint contiene regole guidate dalle specifiche delle risorse. CloudFormation Usa cfn-lint per confrontare le tue risorse con quelle regole per ricevere messaggi dettagliati su errori, avvisi o suggerimenti informativi. In alternativa, crea le tue regole personalizzate in base alle quali convalidare. Per saperne di più su cfn-lint, consulta cfn-lint nel repository.CloudFormation GitHub

Puoi usare cfn-lint per convalidare i tuoi modelli () tramite l'interfaccia a riga di comando AWS Serverless Application Model (AWS SAM) eseguendoli con l' AWS SAM opzione.AWS SAMCLI sam validate --lint

```
sam validate --lint
```

Per personalizzare il comportamento di cfn-lint, come creare regole personalizzate o specificare opzioni di convalida, puoi definire un file di configurazione. Per saperne di più, consulta Config File nel repository CloudFormation GitHub cfn-lint. Quando eseguisam validate --lint, verrà applicato il comportamento cfn-lint definito nel tuo file di configurazione.

Esempi

Esegui la convalida cfn-lint su un modello AWS SAM

```
sam validate --lint --template myTemplate.yaml
```

Ulteriori informazioni

Per ulteriori informazioni sul comando sam validate, consulta sam validate.

Implementa la tua applicazione e le tue risorse con AWS SAM

La distribuzione dell'applicazione fornisce e configura le AWS risorse nel AWS cloud, facendo funzionare l'applicazione nel cloud. AWS SAM utilizza CloudFormationcome meccanismo di implementazione sottostante. AWS SAM utilizza gli elementi di compilazione creati durante l'esecuzione del sam build comando come input standard per la distribuzione dell'applicazione serverless.

Con AWS SAM, puoi distribuire l'applicazione serverless manualmente oppure puoi automatizzare le distribuzioni. Per automatizzare le implementazioni, si utilizzano AWS SAM pipeline con un sistema di integrazione e distribuzione continua (CI/CD) di propria scelta. La pipeline di distribuzione è una sequenza automatizzata di passaggi eseguiti per rilasciare una nuova versione dell'applicazione serverless.

Gli argomenti di questa sezione forniscono indicazioni sulle distribuzioni automatiche e manuali. Per distribuire l'applicazione manualmente, si utilizzano i comandi. AWS SAMCLI Per automatizzare le distribuzioni, consulta gli argomenti di questa sezione. Forniscono in particolare contenuti approfonditi sull'automazione delle implementazioni utilizzando pipeline e un sistema. CI/CD Ciò include la generazione di una pipeline di avvio, la configurazione dell'automazione, la risoluzione dei problemi di implementazione, l'utilizzo dell'autenticazione utente OpenID Connect (OIDC) e il caricamento di file locali al momento dell'implementazione.

Argomenti

- Introduzione alla distribuzione con AWS SAM
- Opzioni per la distribuzione dell'applicazione con AWS SAM
- Utilizzo CI/CD di sistemi e pipeline con cui implementare AWS SAM
- Introduzione all'uso sam sync della sincronizzazione con Cloud AWS

Introduzione alla distribuzione con AWS SAM

Utilizzate il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam deploy` per distribuire l'applicazione serverless su. Cloud AWS

- Per un'introduzione a AWS SAMCLI, vedere. Che cos'è il? AWS SAMCLI

- Per un elenco delle opzioni di `sam deploy` comando, vedere sam deploy.
- Per un esempio di utilizzo `sam deploy` durante un tipico flusso di lavoro di sviluppo, vedi Passo 3: Distribuisci la tua applicazione su Cloud AWS.

Argomenti

- Prerequisiti
- Distribuzione di applicazioni utilizzando sam deploy
- Best practice
- Opzioni per sam deploy
- Risoluzione dei problemi
- Esempi
- Ulteriori informazioni

Prerequisiti

Per utilizzarlo `sam deploy`, AWS SAMCLI installate il completando quanto segue:

- AWS SAM prerequisiti.
- Installa il AWS SAMCLI.

Prima dell'uso `sam deploy`, si consiglia una conoscenza di base di quanto segue:

- Configurazione del AWS SAMCLI.
- Crea la tua candidatura in AWS SAM.
- Introduzione alla costruzione con AWS SAM.

Distribuzione di applicazioni utilizzando sam deploy

Quando distribuisci un'applicazione serverless per la prima volta, utilizza l'opzione. `--guided` Ti AWS SAMCLI guiderà attraverso un flusso interattivo per configurare le impostazioni di distribuzione dell'applicazione.

Per distribuire un'applicazione utilizzando il flusso interattivo

1. Vai alla directory principale del tuo progetto. Questa è la stessa posizione del AWS SAM modello.

```
$ cd sam-app
```

2. Esegui il comando seguente:

```
$ sam deploy --guided
```

3. Durante il flusso interattivo, AWS SAMCLI ti vengono richieste le opzioni per configurare le impostazioni di distribuzione dell'applicazione.

   Le parentesi (`[ ]`) indicano i valori predefiniti. Lascia vuota la risposta per selezionare il valore predefinito. I valori predefiniti sono ottenuti dai seguenti file di configurazione:

   - `~/.aws/config`— Le impostazioni generali AWS dell'account.
   - `~/.aws/credentials`— Le credenziali AWS del tuo account.
   - `<project>/samconfig.toml`— Il file di configurazione del tuo progetto.

   Fornisci valori rispondendo alle istruzioni AWS SAMCLI. Ad esempio, è possibile immettere valori **y** per sì, **n** per no o come stringa.

   AWS SAMCLIScrive le risposte nel `samconfig.toml` file del progetto. Per le distribuzioni successive, è possibile utilizzare `sam deploy` to deploy utilizzando questi valori configurati. Per riconfigurare questi valori, `sam deploy --guided` riutilizza o modifica direttamente i file di configurazione.

   Di seguito è riportato un esempio di output:

```
sam-app $ sam deploy --guided

Configuring SAM deploy
======================

        Looking for config file [samconfig.toml] :  Found
        Reading default arguments  :  Success

        Setting default arguments for 'sam deploy'
```

```
        ========================================
        Stack Name [sam-app]: ENTER
        AWS Region [us-west-2]: ENTER
        #Shows you resources changes to be deployed and require a 'Y' to initiate
 deploy
        Confirm changes before deploy [Y/n]: ENTER
        #SAM needs permission to be able to create roles to connect to the
 resources in your template
        Allow SAM CLI IAM role creation [Y/n]: ENTER
        #Preserves the state of previously provisioned resources when an operation
 fails
        Disable rollback [y/N]: ENTER
        HelloWorldFunction may not have authorization defined, Is this okay? [y/
N]: y
        Save arguments to configuration file [Y/n]: ENTER
        SAM configuration file [samconfig.toml]: ENTER
        SAM configuration environment [default]: ENTER
```

4. Successivamente, AWS SAMCLI distribuisce l'applicazione su. Cloud AWS Durante la distribuzione, lo stato di avanzamento viene visualizzato nel prompt dei comandi. Di seguito sono riportate le fasi principali della distribuzione:

   - Per le applicazioni con AWS Lambda funzioni impacchettate come archivio di file.zip, il AWS SAMCLI pacchetto viene compresso e caricato in un bucket Amazon Simple Storage Service (Amazon S3). Se necessario, creerà un nuovo bucket AWS SAMCLI.
   - Per le applicazioni con funzioni Lambda, il pacchetto viene caricato come immagine contenitore, AWS SAMCLI carica l'immagine su Amazon Elastic Container Registry (Amazon ECR) Elastic Container Registry (Amazon ECR). Se necessario, AWS SAMCLI creeranno un nuovo repository.
   - AWS SAMCLICrea un set di AWS CloudFormation modifiche e distribuisce l'applicazione CloudFormation come stack.
   - AWS SAMCLImodifica il AWS SAM modello distribuito con il nuovo `CodeUri` valore per le funzioni Lambda.

   Di seguito è riportato un esempio dell'output di distribuzione: AWS SAMCLI

```
        Looking for resources needed for deployment:

        Managed S3 bucket: aws-sam-cli-managed-default-samclisam-s3-demo-
bucket-1a4x26zbcdkqr
```

```
        A different default S3 bucket can be set in samconfig.toml and auto
 resolution of buckets turned off by setting resolve_s3=False

        Parameter "stack_name=sam-app" in [default.deploy.parameters] is defined as
 a global parameter [default.global.parameters].
        This parameter will be only saved under [default.global.parameters] in /
Users/.../sam-app/samconfig.toml.

        Saved arguments to config file
        Running 'sam deploy' for future deployments will use the parameters saved
 above.
        The above parameters can be changed by modifying samconfig.toml
        Learn more about samconfig.toml syntax at
        https://docs.aws.amazon.com/serverless-application-model/latest/
developerguide/serverless-sam-cli-config.html

        Uploading to sam-app-zip/da3c598813f1c2151579b73ad788cac8  262144 / 619839
 (42.29%)Uploading to sam-app-zip/da3c598813f1c2151579b73ad788cac8  524288 / 619839
  (84.58%)Uploading to sam-app-zip/da3c598813f1c2151579b73ad788cac8  619839 /
 619839  (100.00%)

        Deploying with following values
        ===============================
        Stack name                   : sam-app
        Region                       : us-west-2
        Confirm changeset            : True
        Disable rollback             : False
        Deployment s3 bucket         : aws-sam-cli-managed-default-samclisam-s3-
demo-bucket-1a4x26zbcdkqr
        Capabilities                 : ["CAPABILITY_IAM"]
        Parameter overrides          : {}
        Signing Profiles             : {}

Initiating deployment
=====================

        Uploading to sam-app-zip/be84c20f868068e4dc4a2c11966edf2d.template  1212 /
 1212  (100.00%)


Waiting for changeset to be created..

CloudFormation stack changeset
-----------------------------------------------------------------------------------------------
```

```
Operation                 LogicalResourceId        ResourceType
 Replacement
-----------------------------------------------------------------------------------------------
+ Add                     HelloWorldFunctionHell   AWS::Lambda::Permissio   N/A

                          oWorldPermissionProd     n

+ Add                     HelloWorldFunctionRole   AWS::IAM::Role           N/A

+ Add                     HelloWorldFunction       AWS::Lambda::Function    N/A

+ Add                     ServerlessRestApiDeplo   AWS::ApiGateway::Deplo   N/A

                          yment47fc2d5f9d          yment

+ Add                     ServerlessRestApiProdS   AWS::ApiGateway::Stage   N/A

                          tage

+ Add                     ServerlessRestApi        AWS::ApiGateway::RestA   N/A

                                                   pi

-----------------------------------------------------------------------------------------------


Changeset created successfully. arn:aws:cloudformation:us-
west-2:012345678910:changeSet/samcli-deploy1680559234/d9f58a77-98bc-41cd-
b9f4-433a5a450d7a


Previewing CloudFormation changeset before deployment
=====================================================
Deploy this changeset? [y/N]: y

2023-04-03 12:00:50 - Waiting for stack create/update to complete

CloudFormation events from stack operations (refresh every 5.0 seconds)
-----------------------------------------------------------------------------------------------
ResourceStatus            ResourceType             LogicalResourceId
 ResourceStatusReason
-----------------------------------------------------------------------------------------------
CREATE_IN_PROGRESS        AWS::IAM::Role           HelloWorldFunctionRole   -
```

```
CREATE_IN_PROGRESS       AWS::IAM::Role           HelloWorldFunctionRole   Resource
 creation

 Initiated
CREATE_COMPLETE          AWS::IAM::Role           HelloWorldFunctionRole   -

CREATE_IN_PROGRESS       AWS::Lambda::Function    HelloWorldFunction       -

CREATE_IN_PROGRESS       AWS::Lambda::Function    HelloWorldFunction       Resource
 creation

 Initiated
CREATE_COMPLETE          AWS::Lambda::Function    HelloWorldFunction       -

CREATE_IN_PROGRESS       AWS::ApiGateway::RestA   ServerlessRestApi        -

                         pi

CREATE_IN_PROGRESS       AWS::ApiGateway::RestA   ServerlessRestApi        Resource
 creation
                         pi
 Initiated
CREATE_COMPLETE          AWS::ApiGateway::RestA   ServerlessRestApi        -

                         pi

CREATE_IN_PROGRESS       AWS::Lambda::Permissio   HelloWorldFunctionHell   -

                         n                        oWorldPermissionProd

CREATE_IN_PROGRESS       AWS::ApiGateway::Deplo   ServerlessRestApiDeplo   -

                         yment                    yment47fc2d5f9d

CREATE_IN_PROGRESS       AWS::Lambda::Permissio   HelloWorldFunctionHell   Resource
 creation
                         n                        oWorldPermissionProd
 Initiated
CREATE_IN_PROGRESS       AWS::ApiGateway::Deplo   ServerlessRestApiDeplo   Resource
 creation
                         yment                    yment47fc2d5f9d
 Initiated
CREATE_COMPLETE          AWS::ApiGateway::Deplo   ServerlessRestApiDeplo   -
```

```
                         yment                     yment47fc2d5f9d

CREATE_IN_PROGRESS       AWS::ApiGateway::Stage    ServerlessRestApiProdS   -

                                                   tage

CREATE_IN_PROGRESS       AWS::ApiGateway::Stage    ServerlessRestApiProdS   Resource
 creation
                                                   tage
 Initiated
CREATE_COMPLETE          AWS::ApiGateway::Stage    ServerlessRestApiProdS   -

                                                   tage

CREATE_COMPLETE          AWS::Lambda::Permissio    HelloWorldFunctionHell   -

                         n                         oWorldPermissionProd

CREATE_COMPLETE          AWS::CloudFormation::S    sam-app-zip              -

                         tack

-----------------------------------------------------------------------------------------------

CloudFormation outputs from deployed stack
-----------------------------------------------------------------------------------------------
Outputs

-----------------------------------------------------------------------------------------------
Key                 HelloWorldFunctionIamRole

Description         Implicit IAM Role created for Hello World function

Value               arn:aws:iam::012345678910:role/sam-app-zip-

HelloWorldFunctionRole-11ZOGSCG28H0M


Key                 HelloWorldApi

Description         API Gateway endpoint URL for Prod stage for Hello World
 function
Value               https://njzfhdmls0.execute-api.us-west-2.amazonaws.com/Prod/
hello/
```

```
Key                 HelloWorldFunction

Description         Hello World Lambda Function ARN

Value               arn:aws:lambda:us-west-2:012345678910:function:sam-app-

HelloWorldFunction-XPqNX4TBu7qn

-----------------------
```

2. Durante la distribuzione, ti AWS SAMCLI chiederà di confermare le modifiche prima della distribuzione. Di seguito è riportato un esempio:

```
Waiting for changeset to be created..

CloudFormation stack changeset
-----------------------------------------------------------------------------------------------
Operation                LogicalResourceId        ResourceType
 Replacement
-----------------------------------------------------------------------------------------------
+ Add                    HelloWorldFunctionHell   AWS::Lambda::Permissio   N/A

                         oWorldPermissionProd     n

+ Add                    HelloWorldFunctionRole   AWS::IAM::Role           N/A

+ Add                    HelloWorldFunction       AWS::Lambda::Function    N/A

+ Add                    ServerlessRestApiDeplo   AWS::ApiGateway::Deplo   N/A

                         yment47fc2d5f9d          yment

+ Add                    ServerlessRestApiProdS   AWS::ApiGateway::Stage   N/A

                         tage

+ Add                    ServerlessRestApi        AWS::ApiGateway::RestA   N/A

                                                  pi

-----------------------------------------------------------------------------------------------

Changeset created successfully. arn:aws:cloudformation:us-
west-2:012345678910:changeSet/samcli-deploy1680559234/d9f58a77-98bc-41cd-
b9f4-433a5a450d7a
Previewing CloudFormation changeset before deployment
=====================================================
Deploy this changeset? [y/N]: y
```

Specificare parametri aggiuntivi durante la distribuzione

È possibile specificare valori di parametro aggiuntivi da configurare durante la distribuzione. È possibile farlo modificando il AWS SAM modello e configurando il valore del parametro durante la distribuzione.

Per specificare parametri aggiuntivi

1. Modifica la `Parameters` sezione del AWS SAM modello. Di seguito è riportato un esempio:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Globals:
...
Parameters:
  DomainName:
    Type: String
    Default: example
    Description: Domain name
```

2. Esegui `sam deploy --guided`. Di seguito è riportato un esempio di output:

```
sam-app $ sam deploy --guided

Configuring SAM deploy
======================

        Looking for config file [samconfig.toml] :  Found
        Reading default arguments  :  Success

        Setting default arguments for 'sam deploy'
        =========================================
        Stack Name [sam-app-zip]: ENTER
        AWS Region [us-west-2]: ENTER
        Parameter DomainName [example]: ENTER
```

Configura la firma del codice per le tue funzioni Lambda

Puoi configurare la firma del codice per le tue funzioni Lambda al momento dell'implementazione. Puoi farlo modificando il AWS SAM modello e configurando la firma del codice durante la distribuzione.

Per configurare la firma del codice

1. `CodeSigningConfigArn`Specificalo nel tuo AWS SAM modello. Di seguito è riportato un esempio:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: hello_world/
      Handler: app.lambda_handler
      Runtime: python3.7
      CodeSigningConfigArn: arn:aws:lambda:us-east-1:111122223333:code-signing-
config:csc-12e12345db1234567
```

2. Esegui `sam deploy --guided`. Ti AWS SAMCLI chiederà di configurare la firma del codice. Di seguito è riportato un esempio di output:

```
#Found code signing configurations in your function definitions
Do you want to sign your code? [Y/n]: ENTER
#Please provide signing profile details for the following functions & layers
#Signing profile details for function 'HelloWorld'
Signing Profile Name:
Signing Profile Owner Account ID (optional):
#Signing profile details for layer 'MyLayer', which is used by functions
 {'HelloWorld'}
Signing Profile Name:
Signing Profile Owner Account ID (optional):
```

Best practice

- Quando si utilizza`sam deploy`, AWS SAMCLI distribuisce gli artefatti di compilazione dell'applicazione che si trovano nella directory. `.aws-sam` Quando apporti modifiche ai file originali dell'applicazione, esegui `sam build` per aggiornare la `.aws-sam` directory prima della distribuzione.
- Quando distribuisci un'applicazione per la prima volta, utilizzala per configurare `sam deploy --guided` le impostazioni di distribuzione. Per le distribuzioni successive, puoi utilizzarlo per eseguire la distribuzione con `sam deploy` le impostazioni configurate.

Opzioni per sam deploy

Di seguito sono riportate le opzioni comunemente utilizzate per`sam deploy`. Per un elenco di tutte le opzioni, vedere[sam deploy](#).

Usa il flusso interattivo guidato per distribuire la tua applicazione

Utilizza l'`--guided`opzione per configurare le impostazioni di distribuzione dell'applicazione tramite un flusso interattivo. Di seguito è riportato un esempio:

```
$ sam deploy --guided
```

Le impostazioni di distribuzione dell'applicazione vengono salvate nel `samconfig.toml` file del progetto. Per ulteriori informazioni, consulta [Configura le impostazioni del progetto](#).

Risoluzione dei problemi

Per risolvere il problema AWS SAMCLI, consulta. [AWS SAMCLIrisoluzione dei problemi](#)

Esempi

Implementa un'applicazione Hello World che contiene una funzione Lambda impacchettata come archivio di file.zip

Per un esempio, vedi [Passo 3: Distribuisci la tua applicazione su Cloud AWS](#) il tutorial dell'applicazione Hello World.

Implementa un'applicazione Hello World che contiene una funzione Lambda impacchettata come immagine contenitore

Innanzitutto, creiamo la nostra applicazione `sam init` Hello World. Durante il flusso interattivo, scegliamo il `Python3.9` runtime e il tipo di `Image` pacchetto.

```
$ sam init
...
Which template source would you like to use?
        1 - AWS Quick Start Templates
        2 - Custom Template Location
Choice: 1

Choose an AWS Quick Start application template
        1 - Hello World Example
        2 - Multi-step workflow
        ...
Template: 1

Use the most popular runtime and package type? (Python and zip) [y/N]:  ENTER

Which runtime would you like to use?
        1 - aot.dotnet7 (provided.al2)
        ...
        15 - nodejs12.x
        16 - python3.9
        17 - python3.8
        ...
Runtime: 16

What package type would you like to use?
        1 - Zip
        2 - Image
Package type: 2

Based on your selections, the only dependency manager available is pip.
We will proceed copying the template using pip.
...
Project name [sam-app]: ENTER

    -----------------------
    Generating application:
    -----------------------
```

```
    Name: sam-app
    Base Image: amazon/python3.9-base
    Architectures: x86_64
    Dependency Manager: pip
    Output Directory: .
    Configuration file: sam-app/samconfig.toml

    Next steps can be found in the README file at sam-app/README.md
...
```

Successivamente, andiamo `cd` alla directory principale del nostro progetto ed eseguiamo`sam build`. AWS SAMCLICostruisce la nostra funzione Lambda localmente utilizzando. Docker

```
sam-app $ sam build
Building codeuri: /Users/.../sam-app runtime: None metadata: {'Dockerfile':
 'Dockerfile', 'DockerContext': '/Users/.../sam-app/hello_world', 'DockerTag':
 'python3.9-v1'} architecture: x86_64 functions: HelloWorldFunction
Building image for HelloWorldFunction function
Setting DockerBuildArgs: {} for HelloWorldFunction function
Step 1/5 : FROM public.ecr.aws/lambda/python:3.9
 ---> 0a5e3da309aa
Step 2/5 : COPY requirements.txt ./
 ---> abc4e82e85f9
Step 3/5 : RUN python3.9 -m pip install -r requirements.txt -t .
 ---> [Warning] The requested image's platform (linux/amd64) does not match the
 detected host platform (linux/arm64/v8) and no specific platform was requested
 ---> Running in 43845e7aa22d
Collecting requests
  Downloading requests-2.28.2-py3-none-any.whl (62 kB)
     ######################################## 62.8/62.8 KB 829.5 kB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.4-py3-none-any.whl (61 kB)
     ######################################## 61.5/61.5 KB 2.4 MB/s eta 0:00:00
Collecting charset-normalizer<4,>=2
  Downloading charset_normalizer-3.1.0-cp39-cp39-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (199 kB)
     ######################################## 199.2/199.2 KB 2.1 MB/s eta 0:00:00
Collecting certifi>=2017.4.17
  Downloading certifi-2022.12.7-py3-none-any.whl (155 kB)
     ######################################## 155.3/155.3 KB 10.2 MB/s eta 0:00:00
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.15-py2.py3-none-any.whl (140 kB)
     ######################################## 140.9/140.9 KB 9.1 MB/s eta 0:00:00
```

```
Installing collected packages: urllib3, idna, charset-normalizer, certifi, requests
Successfully installed certifi-2022.12.7 charset-normalizer-3.1.0 idna-3.4
 requests-2.28.2 urllib3-1.26.15
Removing intermediate container 43845e7aa22d
 ---> cab8ace899ce
Step 4/5 : COPY app.py ./
 ---> 4146f3cd69f2
Step 5/5 : CMD ["app.lambda_handler"]
 ---> [Warning] The requested image's platform (linux/amd64) does not match the
 detected host platform (linux/arm64/v8) and no specific platform was requested
 ---> Running in f4131ddffb31
Removing intermediate container f4131ddffb31
 ---> d2f5180b2154
Successfully built d2f5180b2154
Successfully tagged helloworldfunction:python3.9-v1


Build Succeeded

Built Artifacts  : .aws-sam/build
Built Template   : .aws-sam/build/template.yaml

Commands you can use next
=========================
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name {{stack-name}} --watch
[*] Deploy: sam deploy --guided
```

Successivamente, `sam deploy --guided` eseguiamo la distribuzione della nostra applicazione. Ci AWS SAMCLI guida nella configurazione delle nostre impostazioni di distribuzione. Quindi, AWS SAMCLI distribuisce la nostra applicazione su. Cloud AWS

```
sam-app $ sam deploy --guided

Configuring SAM deploy
======================

        Looking for config file [samconfig.toml] :  Found
        Reading default arguments  :  Success

        Setting default arguments for 'sam deploy'
        ==========================================
```

```
        Stack Name [sam-app]: ENTER
        AWS Region [us-west-2]: ENTER
        #Shows you resources changes to be deployed and require a 'Y' to initiate
 deploy
        Confirm changes before deploy [Y/n]: ENTER
        #SAM needs permission to be able to create roles to connect to the resources in
 your template
        Allow SAM CLI IAM role creation [Y/n]: ENTER
        #Preserves the state of previously provisioned resources when an operation
 fails
        Disable rollback [y/N]: ENTER
        HelloWorldFunction may not have authorization defined, Is this okay? [y/N]: y
        Save arguments to configuration file [Y/n]: ENTER
        SAM configuration file [samconfig.toml]: ENTER
        SAM configuration environment [default]: ENTER

        Looking for resources needed for deployment:

        Managed S3 bucket: aws-sam-cli-managed-default-samclisam-s3-demo-
bucket-1a4x26zbcdkqr
        A different default S3 bucket can be set in samconfig.toml and auto resolution
 of buckets turned off by setting resolve_s3=False

        Parameter "stack_name=sam-app" in [default.deploy.parameters] is defined as a
 global parameter [default.global.parameters].
        This parameter will be only saved under [default.global.parameters] in /
Users/.../sam-app/samconfig.toml.

        Saved arguments to config file
        Running 'sam deploy' for future deployments will use the parameters saved
 above.
        The above parameters can be changed by modifying samconfig.toml
        Learn more about samconfig.toml syntax at
        https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/
serverless-sam-cli-config.html

e95fc5e75742: Pushed
d8df51e7bdd7: Pushed
b1d0d7e0b34a: Pushed
0071317b94d8: Pushed
d98f98baf147: Pushed
2d244e0816c6: Pushed
eb2eeb1ebe42: Pushed
a5ca065a3279: Pushed
```

```
fe9e144829c9: Pushed
helloworldfunction-d2f5180b2154-python3.9-v1: digest:
 sha256:cceb71401b47dc3007a7a1e1f2e0baf162999e0e6841d15954745ecc0c447533 size: 2206

        Deploying with following values
        ===============================
        Stack name                   : sam-app
        Region                       : us-west-2
        Confirm changeset            : True
        Disable rollback             : False
        Deployment image repository  :
                                       {
                                           "HelloWorldFunction":
 "012345678910.dkr.ecr.us-west-2.amazonaws.com/samapp7427b055/
helloworldfunction19d43fc4repo"
                                       }
        Deployment s3 bucket         : aws-sam-cli-managed-default-samclisam-s3-demo-
bucket-1a4x26zbcdkqr
        Capabilities                 : ["CAPABILITY_IAM"]
        Parameter overrides          : {}
        Signing Profiles             : {}

Initiating deployment
=====================

HelloWorldFunction may not have authorization defined.
        Uploading to sam-app/682ad27c7cf7a17c7f77a1688b0844f2.template  1328 / 1328
 (100.00%)


Waiting for changeset to be created..

CloudFormation stack changeset
-----------------------------------------------------------------------------------------------------
Operation                LogicalResourceId        ResourceType             Replacement

-----------------------------------------------------------------------------------------------------
+ Add                    HelloWorldFunctionHell   AWS::Lambda::Permissio   N/A

                         oWorldPermissionProd     n

+ Add                    HelloWorldFunctionRole   AWS::IAM::Role           N/A
```

```
+ Add                     HelloWorldFunction       AWS::Lambda::Function    N/A

+ Add                     ServerlessRestApiDeplo   AWS::ApiGateway::Deplo   N/A

                          yment47fc2d5f9d          yment

+ Add                     ServerlessRestApiProdS   AWS::ApiGateway::Stage   N/A

                          tage

+ Add                     ServerlessRestApi        AWS::ApiGateway::RestA   N/A

                                                   pi

-----------------------------------------------------------------------------------------------------------------------------------


Changeset created successfully. arn:aws:cloudformation:us-
west-2:012345678910:changeSet/samcli-deploy1680634124/0ffd4faf-2e2b-487e-
b9e0-9116e8299ac4


Previewing CloudFormation changeset before deployment
=====================================================
Deploy this changeset? [y/N]: y

2023-04-04 08:49:15 - Waiting for stack create/update to complete

CloudFormation events from stack operations (refresh every 5.0 seconds)
-----------------------------------------------------------------------------------------------------------------------------------
ResourceStatus           ResourceType             LogicalResourceId
 ResourceStatusReason
-----------------------------------------------------------------------------------------------------------------------------------
CREATE_IN_PROGRESS       AWS::CloudFormation::S   sam-app                  User
 Initiated
                          tack

CREATE_IN_PROGRESS       AWS::IAM::Role           HelloWorldFunctionRole   -

CREATE_IN_PROGRESS       AWS::IAM::Role           HelloWorldFunctionRole   Resource
 creation
                                                                           Initiated
```

```
CREATE_COMPLETE          AWS::IAM::Role           HelloWorldFunctionRole   -

CREATE_IN_PROGRESS       AWS::Lambda::Function    HelloWorldFunction       -

CREATE_IN_PROGRESS       AWS::Lambda::Function    HelloWorldFunction       Resource
 creation
                                                                           Initiated

CREATE_COMPLETE          AWS::Lambda::Function    HelloWorldFunction       -

CREATE_IN_PROGRESS       AWS::ApiGateway::RestA   ServerlessRestApi        -

                         pi

CREATE_IN_PROGRESS       AWS::ApiGateway::RestA   ServerlessRestApi        Resource
 creation
                         pi                                                Initiated

CREATE_COMPLETE          AWS::ApiGateway::RestA   ServerlessRestApi        -

                         pi

CREATE_IN_PROGRESS       AWS::Lambda::Permissio   HelloWorldFunctionHell   -

                         n                        oWorldPermissionProd

CREATE_IN_PROGRESS       AWS::ApiGateway::Deplo   ServerlessRestApiDeplo   -

                         yment                    yment47fc2d5f9d

CREATE_IN_PROGRESS       AWS::Lambda::Permissio   HelloWorldFunctionHell   Resource
 creation
                         n                        oWorldPermissionProd     Initiated

CREATE_IN_PROGRESS       AWS::ApiGateway::Deplo   ServerlessRestApiDeplo   Resource
 creation
                         yment                    yment47fc2d5f9d          Initiated

CREATE_COMPLETE          AWS::ApiGateway::Deplo   ServerlessRestApiDeplo   -

                         yment                    yment47fc2d5f9d

CREATE_IN_PROGRESS       AWS::ApiGateway::Stage   ServerlessRestApiProdS   -
```

```
                                                    tage

CREATE_IN_PROGRESS       AWS::ApiGateway::Stage   ServerlessRestApiProdS   Resource
 creation
                                                    tage                     Initiated

CREATE_COMPLETE          AWS::ApiGateway::Stage   ServerlessRestApiProdS   -

                                                    tage

CREATE_COMPLETE          AWS::Lambda::Permissio   HelloWorldFunctionHell   -

                         n                        oWorldPermissionProd

CREATE_COMPLETE          AWS::CloudFormation::S   sam-app                  -

                         tack

-------------------------------------------------------------------------------------------------------

CloudFormation outputs from deployed stack
-------------------------------------------------------------------------------------------------------
Outputs

-------------------------------------------------------------------------------------------------------
Key                 HelloWorldFunctionIamRole

Description         Implicit IAM Role created for Hello World function

Value               arn:aws:iam::012345678910:role/sam-app-HelloWorldFunctionRole-
JFML1JOKHJ71

Key                 HelloWorldApi

Description         API Gateway endpoint URL for Prod stage for Hello World function

Value               https://endlwiqqod.execute-api.us-west-2.amazonaws.com/Prod/hello/


Key                 HelloWorldFunction

Description         Hello World Lambda Function ARN
```

```
Value                arn:aws:lambda:us-west-2:012345678910:function:sam-app-
HelloWorldFunction-
kyg6Y2iNRUPg

-----------------------
```

su Amazon Simple Storage Service (Amazon S3) (per gli archivi di file.zip) o su Amazon Elastic Container Registry (Amazon ECR) (per le immagini dei contenitori). Il comando distribuisce quindi l'applicazione nel cloud. AWS

Esempio:

```
# Deploy an application using prompts:
sam deploy --guided
```

Implementa con CI/CD sistemi e pipeline

AWS SAM ti aiuta ad automatizzare l'implementazione utilizzando pipeline e un sistema di integrazione e distribuzione continua (CI/CD). AWS SAM può essere utilizzato per creare pipeline e CI/CD semplificare le attività per le applicazioni serverless. Più CI/CD sistemi supportano la AWS SAM creazione di immagini di container e forniscono AWS SAM anche una serie di modelli di pipeline predefiniti per più CI/CD sistemi che racchiudono AWS le migliori pratiche di implementazione.

Per ulteriori informazioni, consulta Utilizzo CI/CD di sistemi e pipeline con cui implementare AWS SAM.

Implementazioni graduali

Se si desidera distribuire l' AWS SAM applicazione gradualmente anziché tutto in una volta, è possibile specificare le configurazioni di distribuzione appropriate. AWS CodeDeploy Per ulteriori informazioni, consulta Lavorare con le configurazioni di distribuzione CodeDeploy nella Guida per l'AWS CodeDeploy utente.

Per informazioni sulla configurazione AWS SAM dell'applicazione per una distribuzione graduale, consulta. Implementazione graduale di applicazioni serverless con AWS SAM

Risoluzione dei problemi di distribuzione utilizzando il AWS SAMCLI

AWS SAMCLIerrore: «Vincoli di sicurezza non soddisfatti»

Durante l'esecuzionesam deploy --guided, ti viene posta la domanda. `HelloWorldFunction may not have authorization defined, Is this okay? [y/N]` Se rispondi a questa richiesta con **`N`** (la risposta predefinita), viene visualizzato il seguente errore:

```
Error: Security Constraints Not Satisfied
```

Il messaggio ti informa che l'applicazione che stai per distribuire potrebbe avere un'API Amazon API Gateway configurata senza autorizzazione. Rispondendo **N** a questa richiesta, stai dicendo che non va bene.

Per risolvere questo problema, sono disponibili le seguenti opzioni:

- Configura la tua applicazione con autorizzazione. Per informazioni sulla configurazione dell'autorizzazione, vedereControlla l'accesso alle API con il tuo AWS SAM modello.
- Rispondi a questa domanda indicando che sei d'accordo con la distribuzione di un'applicazione con un'API API Gateway configurata senza autorizzazione. **Y**

Ulteriori informazioni

Per esempi pratici di implementazione di applicazioni serverless, consulta quanto segue tratto da The Complete Workshop: AWS SAM

- Modulo 3 - Distribuzione manuale: scopri come creare, impacchettare e distribuire un'applicazione serverless utilizzando il. AWS SAMCLI
- Modulo 4 - CI/CD: scopri come automatizzare le fasi di creazione, pacchetto e distribuzione creando una pipeline di integrazione e distribuzione continue (CI/CD).

Utilizzo CI/CD di sistemi e pipeline con cui implementare AWS SAM

AWS SAM aiuta le organizzazioni a creare pipeline per i propri CI/CD sistemi preferiti, in modo che possano trarne i vantaggi CI/CD con il minimo sforzo, ad esempio accelerando la frequenza di implementazione, accorciando i tempi di esecuzione delle modifiche e riducendo gli errori di implementazione.

AWS SAM semplifica le CI/CD attività per le applicazioni serverless con l'aiuto della creazione di immagini di container. Le immagini AWS SAM fornite includono gli AWS SAMCLI strumenti di compilazione per una serie di runtime supportati AWS Lambda . Ciò semplifica la creazione e il pacchetto di applicazioni serverless utilizzando. AWS SAMCLI Queste immagini riducono inoltre la necessità per i team di creare e gestire le proprie immagini per CI/CD i sistemi. Per ulteriori informazioni sulla AWS SAM creazione di immagini dei container, consultaArchivi di immagini per AWS SAM.

Più CI/CD sistemi supportano la AWS SAM creazione di immagini dei container. CI/CD Il sistema da utilizzare dipende da diversi fattori. Questi includono se l'applicazione utilizza un singolo runtime o più runtime o se si desidera creare l'applicazione all'interno di un'immagine del contenitore o direttamente su una macchina host, una macchina virtuale (VM) o un host bare metal.

AWS SAM fornisce inoltre una serie di modelli di pipeline predefiniti per più CI/CD sistemi che AWS incapsulano le migliori pratiche di implementazione. Questi modelli di pipeline predefiniti utilizzano formati di configurazione delle JSON/YAML pipeline standard e le best practice integrate aiutano a eseguire distribuzioni con più account e più regioni e a verificare che le pipeline non possano apportare modifiche involontarie all'infrastruttura.

Sono disponibili due opzioni principali AWS SAM da utilizzare per distribuire le applicazioni serverless: 1) modificare la configurazione della pipeline esistente per utilizzare AWS SAMCLI i comandi o 2) generare una configurazione di pipeline di esempio CI/CD da utilizzare come punto di partenza per la propria applicazione.

Argomenti

- Cos'è una pipeline?
- Come AWS SAM vengono caricati i file locali al momento della distribuzione
- Genera una CI/CD pipeline iniziale con AWS SAM
- Come personalizzare le pipeline di avviamento con AWS SAM
- Automatizza l'implementazione della tua AWS SAM applicazione
- Come utilizzare l'autenticazione OIDC con le pipeline AWS SAM

Cos'è una pipeline?

Una pipeline è una sequenza automatizzata di passaggi eseguiti per rilasciare una nuova versione di un'applicazione. Con AWS SAM, puoi utilizzare molti CI/CD sistemi comuni per distribuire le tue applicazioni, tra cui Jenkins AWS CodePipeline, GitLab CI/CD e Actions. GitHub

I modelli di pipeline includono le migliori pratiche AWS di implementazione per facilitare le distribuzioni con più account e più regioni. AWS ambienti come sviluppo e produzione in genere esistono in account diversi. AWS Ciò consente ai team di sviluppo di configurare pipeline di implementazione sicure, senza apportare modifiche involontarie all'infrastruttura.

Puoi anche fornire modelli di pipeline personalizzati per aiutare a standardizzare le pipeline tra i team di sviluppo.

Come AWS SAM vengono caricati i file locali al momento della distribuzione

Quando si distribuisce l'applicazione su Cloud AWS, è AWS CloudFormation necessario che i file locali vengano prima caricati su un AWS servizio accessibile, come Amazon Simple Storage Service (Amazon S3) Simple Storage Service (Amazon S3). Ciò include i file locali a cui fa riferimento il AWS SAM modello. Per soddisfare questo requisito, AWS SAMCLI esegue le seguenti operazioni quando si utilizza il `sam package` comando `sam deploy` or:

1. Carica automaticamente i file locali su un AWS servizio accessibile.
2. Aggiorna automaticamente il modello dell'applicazione in modo che faccia riferimento al nuovo percorso del file.

Argomenti

- Demo: usa il codice della funzione AWS SAMCLI Lambda per caricare il codice della funzione Lambda
- Casi di utilizzo supportati
- Ulteriori informazioni

Demo: usa il codice della funzione AWS SAMCLI Lambda per caricare il codice della funzione Lambda

In questa demo, inizializziamo l'applicazione Hello World di esempio utilizzando un tipo di pacchetto.zip per la nostra funzione Lambda. Utilizziamo il AWS SAMCLI per caricare automaticamente il nostro codice della funzione Lambda su Amazon S3 e fare riferimento al suo nuovo percorso nel nostro modello di applicazione.

Innanzitutto, `sam init` eseguiamo l'inizializzazione della nostra applicazione Hello World.

```
$ sam init
...
Which template source would you like to use?
        1 - AWS Quick Start Templates
        2 - Custom Template Location
Choice: 1

Choose an AWS Quick Start application template
        1 - Hello World Example
```

```
        2 - Multi-step workflow
        ...
Template: 1

Use the most popular runtime and package type? (Python and zip) [y/N]: y

Would you like to enable X-Ray tracing on the function(s) in your application?  [y/
N]: ENTER

Would you like to enable monitoring using CloudWatch Application Insights?
For more info, please view https://docs.aws.amazon.com/AmazonCloudWatch/latest/
monitoring/cloudwatch-application-insights.html [y/N]: ENTER

Project name [sam-app]: demo

    -----------------------
    Generating application:
    -----------------------
    Name: demo
    Runtime: python3.9
    Architectures: x86_64
    Dependency Manager: pip
    Application Template: hello-world
    Output Directory: .
    Configuration file: demo/samconfig.toml

...
```

Il nostro codice di funzione Lambda è organizzato nella `hello_world` sottodirectory del nostro progetto.

```
demo
### README.md
### hello_world
#   ### __init__.py
#   ### app.py
#   ### requirements.txt
### template.yaml
### tests
```

All'interno AWS SAM del nostro modello, facciamo riferimento al percorso locale del nostro codice di funzione Lambda utilizzando la `CodeUri` proprietà.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function # More info about Function Resource:
 https://github.com/awslabs/serverless-application-model/blob/master/
versions/2016-10-31.md#awsserverlessfunction
    Properties:
      CodeUri: hello_world/
      Handler: app.lambda_handler
      Runtime: python3.9
      ...
```

Successivamente, `sam build` eseguiamo la creazione della nostra applicazione e ci prepariamo per la distribuzione.

```
$ sam build
Starting Build use cache
Manifest file is changed (new hash: 3298f13049d19cffaa37ca931dd4d421) or dependency
 folder (.aws-sam/deps/7896875f-9bcc-4350-8adb-2c1d543627a1) is missing for
 (HelloWorldFunction), downloading dependencies and copying/building source
Building codeuri: /Users/.../demo/hello_world runtime: python3.9 metadata: {}
 architecture: x86_64 functions: HelloWorldFunction
Running PythonPipBuilder:CleanUp
Running PythonPipBuilder:ResolveDependencies
Running PythonPipBuilder:CopySource
Running PythonPipBuilder:CopySource

Build Succeeded

Built Artifacts  : .aws-sam/build
Built Template   : .aws-sam/build/template.yaml
...
```

Successivamente, corriamo `sam deploy --guided` per distribuire la nostra applicazione.

```
$ sam deploy --guided

Configuring SAM deploy
======================
```

```
        Looking for config file [samconfig.toml] :  Found
        Reading default arguments  :  Success

        Setting default arguments for 'sam deploy'
        =========================================
        Stack Name [demo]: ENTER
        AWS Region [us-west-2]: ENTER
        #Shows you resources changes to be deployed and require a 'Y' to initiate
 deploy
        Confirm changes before deploy [Y/n]: n
        #SAM needs permission to be able to create roles to connect to the resources in
 your template
        Allow SAM CLI IAM role creation [Y/n]: ENTER
        #Preserves the state of previously provisioned resources when an operation
 fails
        Disable rollback [y/N]: ENTER
        HelloWorldFunction may not have authorization defined, Is this okay? [y/N]: y
        Save arguments to configuration file [Y/n]: ENTER
        SAM configuration file [samconfig.toml]: ENTER
        SAM configuration environment [default]: ENTER

        Looking for resources needed for deployment:
        ...
        Saved arguments to config file
        Running 'sam deploy' for future deployments will use the parameters saved
 above.
        The above parameters can be changed by modifying samconfig.toml
        Learn more about samconfig.toml syntax at
        https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/
serverless-sam-cli-config.html

File with same data already exists at demo/da3c598813f1c2151579b73ad788cac8, skipping
 upload

        Deploying with following values
        ===============================
        Stack name                   : demo
        Region                       : us-west-2
        Confirm changeset            : False
        Disable rollback             : False
        Deployment s3 bucket         : aws-sam-cli-managed-default-samclisam-s3-demo-
bucket-1a4x26zbcdkqr
        Capabilities                 : ["CAPABILITY_IAM"]
        Parameter overrides          : {}
```

```
        Signing Profiles             : {}

Initiating deployment
=====================
...
Waiting for changeset to be created..
CloudFormation stack changeset
-----------------------------------------------------------------------------------------------------
Operation                LogicalResourceId        ResourceType             Replacement

-----------------------------------------------------------------------------------------------------
+ Add                    HelloWorldFunctionHell   AWS::Lambda::Permissio   N/A

                         oWorldPermissionProd     n

+ Add                    HelloWorldFunctionRole   AWS::IAM::Role           N/A

...
-----------------------------------------------------------------------------------------------------

Changeset created successfully. arn:aws:cloudformation:us-
west-2:012345678910:changeSet/samcli-deploy1680906292/1164338d-72e7-4593-a372-
f2b3e67f542f

2023-04-07 12:24:58 - Waiting for stack create/update to complete

CloudFormation events from stack operations (refresh every 5.0 seconds)
-----------------------------------------------------------------------------------------------------
ResourceStatus           ResourceType             LogicalResourceId
 ResourceStatusReason
-----------------------------------------------------------------------------------------------------
CREATE_IN_PROGRESS       AWS::IAM::Role           HelloWorldFunctionRole   -

CREATE_IN_PROGRESS       AWS::IAM::Role           HelloWorldFunctionRole   Resource
 creation
                                                                           Initiated

...
-----------------------------------------------------------------------------------------------------
CloudFormation outputs from deployed stack
-----------------------------------------------------------------------------------------------------
Outputs

-----------------------------------------------------------------------------------------------------
```

```
Key                 HelloWorldFunctionIamRole

Description         Implicit IAM Role created for Hello World function

Value               arn:aws:iam::012345678910:role/demo-HelloWorldFunctionRole-
VQ4CU7UY7S2K

Key                 HelloWorldApi

Description         API Gateway endpoint URL for Prod stage for Hello World function

Value               https://satnon55e9.execute-api.us-west-2.amazonaws.com/Prod/hello/


Key                 HelloWorldFunction

Description         Hello World Lambda Function ARN

Value               arn:aws:lambda:us-west-2:012345678910:function:demo-

HelloWorldFunction-G14inKTmSQvK

-----------------------------------------------------------------------------------------------------------------------
Successfully created/updated stack - demo in us-west-2
```

Durante la distribuzione, carica AWS SAMCLI automaticamente il nostro codice della funzione Lambda su Amazon S3 e aggiorna il nostro modello. Il nostro modello modificato nella CloudFormation console riflette il percorso del bucket Amazon S3.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: s3://aws-sam-cli-managed-default-samclisam-s3-demo-bucket-1a4x26zbcdkqr/
demo/da3c598813f1c2151579b73ad788cac8
      Handler: app.lambda_handler
      ...
```

Casi di utilizzo supportati

AWS SAMCLIPossono facilitare automaticamente questo processo per diversi tipi di file, tipi di CloudFormation risorse e CloudFormation macro.

Tipi di file

I file e Docker le immagini delle applicazioni sono supportati.

CloudFormation tipi di risorse

Di seguito è riportato un elenco dei tipi di risorse supportati e delle relative proprietà:

Risorsa	Properties
`AWS::ApiGateway::RestApi`	`BodyS3Location`
`AWS::ApiGatewayV2::Api`	`BodyS3Location`

`AWS::AppSync::FunctionConfiguration`	`CodeS3Location`
	`RequestMappingTemplateS3Location`
	`ResponseMappingTemplateS3Location`

`AWS::AppSync::GraphQLSchema`	`DefinitionS3Location`

`AWS::AppSync::Resolver`	`CodeS3Location`
	`RequestMappingTemplateS3Location`
	`ResponseMappingTemplateS3Location`

`AWS::CloudFormation::ModuleVersion`	`ModulePackage`
`AWS::CloudFormation::ResourceVersion`	`SchemaHandlerPackage`

Risorsa	Properties
`AWS::ECR::Repository`	`RepositoryName`
`AWS::ElasticBeanstalk::ApplicationVersion`	`SourceBundle`
`AWS::Glue::Job`	`Command.ScriptLocation`
`AWS::Lambda::Function`	`Code` `Code.ImageUri`
`AWS::Lambda::LayerVersion`	`Content`
`AWS::Serverless::Api`	`DefinitionUri`
`AWS::Serverless::Function`	`CodeUri` `ImageUri`

`AWS::Serverless::GraphQLApi`	`SchemaUri`
	`Function.CodeUri`
	`Resolver.CodeUri`

`AWS::Serverless::HttpApi`	`DefinitionUri`
`AWS::Serverless::LayerVersion`	`ContentUri`
`AWS::Serverless::StateMachine`	`DefinitionUri`
`AWS::StepFunctions::StateMachine`	`DefinitionS3Location`

CloudFormation macro

I file a cui si fa riferimento utilizzando la macro di `AWS::Include` trasformazione sono supportati.

Ulteriori informazioni

Per ulteriori informazioni sulla `AWS::Include` trasformazione, consulta la sezione AWS::Include Transform nella Guida AWS CloudFormation per l'utente.

Per vedere un esempio di utilizzo della `AWS::Include` trasformazione in un AWS SAM modello, consulta il pattern API Gateway HTTP API to SQS su Serverless Land.

Genera una CI/CD pipeline iniziale con AWS SAM

Quando sei pronto per automatizzare l'implementazione, puoi utilizzare uno dei modelli di pipeline AWS SAM di avvio disponibili per generare una pipeline di distribuzione per il CI/CD sistema che scegli di utilizzare. La pipeline di distribuzione è ciò che configuri e utilizzi per automatizzare la distribuzione della tua applicazione serverless. Un modello di pipeline iniziale è preconfigurato per aiutarti a configurare rapidamente la pipeline di distribuzione per la tua applicazione serverless.

Con un modello di pipeline iniziale, puoi generare pipeline in pochi minuti utilizzando il comando. sam pipeline init

I modelli di pipeline starter utilizzano la JSON/YAML sintassi familiare del CI/CD sistema e incorporano best practice come la gestione degli artefatti su più account e aree e l'utilizzo della quantità minima di autorizzazioni richieste per distribuire l'applicazione. Attualmente, la AWS SAM CLI supporta la generazione di configurazioni di CI/CD pipeline di avvio per Jenkins, GitLab CI/CD AWS CodePipeline, Actions e Bitbucket Pipelines. GitHub

Ecco le attività di alto livello da eseguire per generare una configurazione di pipeline iniziale:

1. Crea risorse infrastrutturali: la tua pipeline richiede determinate AWS risorse, ad esempio l'utente e i ruoli IAM con le autorizzazioni necessarie, un bucket Amazon S3 e, facoltativamente, un repository Amazon ECR.
2. Connetti il tuo repository Git al tuo CI/CD sistema: il tuo CI/CD sistema deve sapere quale repository Git attiverà l'esecuzione della pipeline. Nota che questo passaggio potrebbe non essere necessario, a seconda della combinazione di repository Git e CI/CD sistema che stai utilizzando.
3. Genera la configurazione della pipeline: questo passaggio genera una configurazione iniziale della pipeline che include due fasi di distribuzione.
4. Esegui il commit della configurazione della pipeline nel tuo repository Git: questo passaggio è necessario per garantire che il CI/CD sistema sia a conoscenza della configurazione della pipeline e verrà eseguito quando verranno eseguite le modifiche.

Dopo aver generato la configurazione della pipeline di partenza e averla salvata nel tuo repository Git, ogni volta che qualcuno esegue una modifica al codice in quel repository, la pipeline verrà attivata per essere eseguita automaticamente.

L'ordine di questi passaggi e i dettagli di ogni passaggio variano in base al sistema CI/CD in uso:

- Se si utilizza AWS CodePipeline, vedere. Generazione di una pipeline di avviamento per AWS CodePipeline AWS SAM
- Se utilizzi Jenkins, GitLab CI/CD, GitHub Actions o Bitbucket Pipelines, vedi. AWS SAM Da utilizzare per generare pipeline di avvio per Jenkins, GitLab CI/CD, Actions, Bitbucket Pipelines GitHub

Generazione di una pipeline di avviamento per AWS CodePipeline AWS SAM

Per generare una configurazione di pipeline di avviamento per AWS CodePipeline, eseguite le seguenti attività in questo ordine:

1. Creare risorse infrastrutturali
2. Genera la configurazione della pipeline
3. Esegui il commit della configurazione della pipeline su Git
4. Connect il tuo repository Git con il tuo sistema CI/CD

> **Note**
>
> La procedura seguente utilizza due AWS SAMCLI comandi, `sam pipeline bootstrap` e. `sam pipeline init` Il motivo per cui esistono due comandi è quello di gestire il caso d'uso in cui gli amministratori (ovvero gli utenti che necessitano dell'autorizzazione per configurare le AWS risorse dell'infrastruttura come gli utenti e i ruoli IAM) dispongono di maggiori autorizzazioni rispetto agli sviluppatori (ovvero gli utenti che necessitano solo dell'autorizzazione per configurare le singole pipeline, ma non le risorse di infrastruttura AWS richieste).

Fase 1: Creare risorse infrastrutturali

Le pipeline che utilizzano AWS SAM richiedono determinate AWS risorse, come un utente IAM e ruoli con le autorizzazioni necessarie, un bucket Amazon S3 e, facoltativamente, un repository Amazon

ECR. È necessario disporre di un set di risorse infrastrutturali per ogni fase di implementazione della pipeline.

È possibile eseguire il comando seguente per facilitare questa configurazione:

```
sam pipeline bootstrap
```

> **Note**
>
> Esegui il comando precedente per ogni fase di distribuzione della pipeline.

Fase 2: Generare la configurazione della pipeline

Per generare la configurazione della pipeline, esegui il comando seguente:

```
sam pipeline init
```

Passaggio 3: Esegui il commit della configurazione della pipeline nell'archivio Git

Questo passaggio è necessario per garantire che il CI/CD sistema sia a conoscenza della configurazione della pipeline e venga eseguito quando vengono apportate le modifiche.

Passaggio 4: Connect il repository Git al sistema CI/CD

Perché ora AWS CodePipeline puoi creare la connessione eseguendo il seguente comando:

```
sam deploy -t codepipeline.yaml --stack-name <pipeline-stack-name> --
capabilities=CAPABILITY_IAM --region <region-X>
```

Se utilizzi GitHub o Bitbucket, dopo aver eseguito il sam deploy comando in precedenza, completa la connessione seguendo la procedura riportata in Per completare una connessione disponibile nell'argomento Aggiornare una connessione in sospeso nella guida per l'utente della console Developer Tools. Inoltre, memorizza una copia del file `CodeStarConnectionArn` dall'output del sam deploy comando, poiché ti servirà se desideri utilizzarlo AWS CodePipeline con un altro ramo di. `main`

Configurazione di altre filiali

Per impostazione predefinita, AWS CodePipeline utilizza il `main` ramo con AWS SAM. Se si desidera utilizzare un ramo diverso da`main`, è necessario eseguire nuovamente il sam deploy

comando. Nota che, a seconda del repository Git che stai utilizzando, potresti dover fornire anche: `CodeStarConnectionArn`

```
# For GitHub and Bitbucket
sam deploy -t codepipeline.yaml --stack-name <feature-pipeline-stack-name> --
capabilities=CAPABILITY_IAM --parameter-overrides="FeatureGitBranch=<branch-name>
 CodeStarConnectionArn=<codestar-connection-arn>"

# For AWS CodeCommit
sam deploy -t codepipeline.yaml --stack-name <feature-pipeline-stack-name> --
capabilities=CAPABILITY_IAM --parameter-overrides="FeatureGitBranch=<branch-name>"
```

Ulteriori informazioni

Per un esempio pratico di configurazione di una CI/CD pipeline, vedi CI/CD with in The Complete Workshop. AWS CodePipeline AWS SAM

AWS SAM Da utilizzare per generare pipeline di avvio per Jenkins, GitLab CI/CD, Actions, Bitbucket Pipelines GitHub

Per generare una configurazione di pipeline iniziale per Jenkins, GitLab CI/CD, Actions o Bitbucket Pipelines, esegui le seguenti attività in questo ordine GitHub :

1. Crea risorse infrastrutturali
2. Connect il tuo repository Git con il tuo sistema CI/CD
3. Crea oggetti credenziali
4. Genera la configurazione della pipeline
5. Esegui il commit della configurazione della pipeline nel repository Git

> **Note**
>
> La procedura seguente utilizza due AWS SAMCLI comandi, e. `sam pipeline bootstrap` `sam pipeline init` Il motivo per cui esistono due comandi è quello di gestire il caso d'uso in cui gli amministratori (ovvero gli utenti che necessitano dell'autorizzazione per configurare le AWS risorse dell'infrastruttura come gli utenti e i ruoli IAM) dispongono di maggiori autorizzazioni rispetto agli sviluppatori (ovvero gli utenti che necessitano solo

dell'autorizzazione per configurare le singole pipeline, ma non le risorse di infrastruttura AWS richieste).

Fase 1: Creare risorse infrastrutturali

Le pipeline che utilizzano AWS SAM richiedono determinate AWS risorse, come un utente IAM e ruoli con le autorizzazioni necessarie, un bucket Amazon S3 e, facoltativamente, un repository Amazon ECR. È necessario disporre di un set di risorse infrastrutturali per ogni fase di implementazione della pipeline.

È possibile eseguire il comando seguente per facilitare questa configurazione:

```
sam pipeline bootstrap
```

> **Note**
>
> Esegui il comando precedente per ogni fase di distribuzione della pipeline.

È necessario acquisire le AWS credenziali (ID chiave e chiave segreta) per gli utenti della pipeline per ogni fase di distribuzione della pipeline, poiché sono necessarie per i passaggi successivi.

Passaggio 2: Connect il repository Git al sistema CI/CD

La connessione del repository Git al CI/CD sistema è necessaria in modo che il CI/CD sistema possa accedere al codice sorgente dell'applicazione per le build e le distribuzioni.

> **Note**
>
> Puoi saltare questo passaggio se utilizzi una delle seguenti combinazioni, poiché la connessione viene eseguita automaticamente:
>
> 1. GitHub Azioni con repository GitHub
> 2. GitLab CI/CD con repository GitLab
> 3. Bitbucket Pipelines con un repository Bitbucket

Per connettere il tuo repository Git al tuo CI/CD sistema, esegui una delle seguenti operazioni:

- Se usi Jenkins, consulta la documentazione di Jenkins per «Aggiungere una sorgente filiale».
- Se utilizzi GitLab CI/CD e un repository Git diverso da quello GitLab, consulta la GitLabdocumentazione per «connettere un repository esterno».

Fase 3: Creare oggetti di credenziali

Ogni CI/CD sistema ha il proprio modo di gestire le credenziali necessarie al CI/CD sistema per accedere al tuo repository Git.

Per creare gli oggetti di credenziali necessari, esegui una delle seguenti operazioni:

- Se utilizzi Jenkins, crea un'unica «credenziale» che memorizzi sia l'ID della chiave che la chiave segreta. Segui le istruzioni nel AWS SAM blog Building a Jenkins Pipeline with, nella sezione Configure Jenkins. Avrai bisogno del «Credential ID» per il passaggio successivo.
- Se utilizzi GitLab CI/CD, crea due «variabili protette», una per ogni ID chiave e chiave segreta. Segui le istruzioni nella GitLab documentazione: per il passaggio successivo avrai bisogno di due «chiavi variabili».
- Se utilizzi GitHub Actions, crea due «segreti crittografati», uno per ciascuna chiave e chiave segreta. Segui le istruzioni nella GitHubdocumentazione: per il passaggio successivo ti serviranno due «nomi segreti».
- Se utilizzi Bitbucket Pipelines, crea due «variabili sicure», una per ogni ID chiave e chiave segreta. Segui le istruzioni in Variabili e segreti: per il passaggio successivo ti serviranno due «nomi segreti».

Fase 4: Generazione della configurazione della pipeline

Per generare la configurazione della pipeline, esegui il comando seguente. Dovrai inserire l'oggetto credenziale che hai creato nel passaggio precedente:

```
sam pipeline init
```

Passaggio 5: Esegui il commit della configurazione della pipeline nell'archivio Git

Questo passaggio è necessario per garantire che il CI/CD sistema sia a conoscenza della configurazione della pipeline e venga eseguito una volta apportate le modifiche.

Ulteriori informazioni

Per un esempio pratico di configurazione di una CI/CD pipeline utilizzandoGitHub Actions, vedi CI/CD with GitHub in The Complete Workshop. AWS SAM

Come personalizzare le pipeline di avviamento con AWS SAM

In qualità di CI/CD amministratore, potresti voler personalizzare un modello di pipeline iniziale e i relativi prompt guidati, che gli sviluppatori della tua organizzazione possono utilizzare per creare configurazioni di pipeline.

AWS SAMCLIUtilizza i modelli Cookiecutter per la creazione di modelli iniziali. Per dettagli sui modelli di stampini per biscotti, Cookiecutter.

È inoltre possibile personalizzare i prompt che AWS SAMCLI vengono visualizzati agli utenti durante la creazione di configurazioni di pipeline utilizzando il comando. `sam pipeline init` Per personalizzare i prompt degli utenti, effettuate le seguenti operazioni:

1. Crea un **`questions.json`** file: il `questions.json` file deve trovarsi nella radice del repository del progetto. Questa è la stessa directory del file. `cookiecutter.json` Per visualizzare lo schema del `questions.json` file, consulta questions.json.schema. Per visualizzare un file di esempio`questions.json`, vedi questions.json.
2. Mappa le chiavi delle domande con i nomi cookiecutter: ogni oggetto nel `questions.json` file necessita di una chiave che corrisponda a un nome nel modello cookiecutter. Questa corrispondenza di tasti consente di mappare le risposte richieste dall'utente AWS SAMCLI al modello di cookie cutter. Per vedere esempi di questa corrispondenza di tasti, consultate la File di esempio sezione più avanti in questo argomento.
3. Crea un **`metadata.json`** file: dichiara il numero di fasi che la pipeline avrà nel `metadata.json` file. Il numero di fasi indica al `sam pipeline init` comando per quante fasi richiedere informazioni o, nel caso dell'`--bootstrap`opzione, per quante fasi creare risorse di infrastruttura. Per visualizzare un `metadata.json` file di esempio che dichiara una pipeline con due fasi, vedete metadata.json.

Progetti di esempio

Ecco alcuni progetti di esempio, ciascuno dei quali include un modello Cookiecutter, un `questions.json` file e un file: `metadata.json`

- Esempio Jenkins: modello di pipeline Jenkins a due fasi
- CodePipeline esempio: modello di pipeline a due fasi CodePipeline

File di esempio

Il seguente set di file mostra come le domande nel `questions.json` file sono associate alle voci nel file modello Cookiecutter. Nota che questi esempi sono frammenti di file, non file completi. Per vedere esempi di file completi, consultate la Progetti di esempio sezione precedente di questo argomento.

Esempio **`questions.json`**:

```
{
  "questions": [{
    "key": "intro",
    "question": "\nThis template configures a pipeline that deploys a serverless
 application to a testing and a production stage.\n",
    "kind": "info"
  }, {
    "key": "pipeline_user_jenkins_credential_id",
    "question": "What is the Jenkins credential ID (via Jenkins plugin \"aws-
credentials\") for pipeline user access key?",
    "isRequired": true
  }, {
    "key": "sam_template",
    "question": "What is the template file path?",
    "default": "template.yaml"
  }, {
    ...
```

Esempio **`cookiecutter.json`**:

```
{
  "outputDir": "aws-sam-pipeline",
  "pipeline_user_jenkins_credential_id": "",
  "sam_template": "",
    ...
```

Esempio **`Jenkinsfile`**:

```
pipeline {
```

```
  agent any
  environment {
    PIPELINE_USER_CREDENTIAL_ID =
'{{cookiecutter.pipeline_user_jenkins_credential_id}}'
    SAM_TEMPLATE = '{{cookiecutter.sam_template}}'
    ...
```

Automatizza l'implementazione della tua AWS SAM applicazione

Inoltre AWS SAM, il modo in cui automatizzi la distribuzione dell' AWS SAM applicazione varia a seconda del CI/CD sistema che stai utilizzando. Per questo motivo, gli esempi in questa sezione mostrano come configurare vari CI/CD sistemi per automatizzare la creazione di applicazioni serverless in un'immagine del contenitore di AWS SAM compilazione. Queste immagini dei container di compilazione semplificano la creazione e il pacchetto di applicazioni serverless utilizzando. AWS SAMCLI

Le procedure utilizzate dalla CI/CD pipeline esistente per la distribuzione di applicazioni serverless AWS SAM sono leggermente diverse a seconda del CI/CD sistema in uso.

I seguenti argomenti forniscono esempi per configurare il CI/CD sistema per creare applicazioni serverless all'interno di un'immagine del contenitore di compilazione: AWS SAM

Argomenti

- Utilizzo AWS CodePipeline per la distribuzione con AWS SAM
- Utilizzo di Bitbucket Pipelines per la distribuzione con AWS SAM
- Utilizzo di Jenkins per la distribuzione con AWS SAM
- Utilizzo di GitLab CI/CD per la distribuzione con AWS SAM
- Utilizzo GitHub delle azioni con cui eseguire la distribuzione AWS SAM

Utilizzo AWS CodePipeline per la distribuzione con AWS SAM

Per configurare la AWS CodePipelinepipeline in modo da automatizzare la creazione e la distribuzione dell' AWS SAM applicazione, il CloudFormation modello e il `buildspec.yml` file devono contenere righe che eseguano le seguenti operazioni:

1. Fai riferimento a un'immagine del contenitore di compilazione con il runtime necessario tra le immagini disponibili. L'esempio seguente utilizza l'immagine del contenitore di `public.ecr.aws/sam/build-nodejs20.x` compilazione.

2. Configura le fasi della pipeline per eseguire i comandi CLI (AWS SAM Command Line Interface) necessari. L'esempio seguente esegue due AWS SAMCLI comandi: sam build e sam deploy (con le opzioni necessarie).

L'esempio presuppone che abbiate dichiarato tutte le funzioni e i livelli nel file AWS SAM modello con`runtime: nodejs20.x`.

CloudFormation frammento di modello:

```
  CodeBuildProject:
    Type: AWS::CodeBuild::Project
    Properties:
      Environment:
        ComputeType: BUILD_GENERAL1_SMALL
        Image: public.ecr.aws/sam/build-nodejs20.x
        Type: LINUX_CONTAINER
      ...
```

`buildspec.yml`frammento:

```
version: 0.2
phases:
  build:
    commands:
      - sam build
      - sam deploy --no-confirm-changeset --no-fail-on-empty-changeset
```

Per un elenco delle immagini dei container di build Amazon Elastic Container Registry (Amazon ECR) disponibili per diversi runtime, consulta. Archivi di immagini per AWS SAM

Utilizzo di Bitbucket Pipelines per la distribuzione con AWS SAM

Per configurare Bitbucket Pipeline per automatizzare la creazione e la distribuzione dell' AWS SAM applicazione, il `bitbucket-pipelines.yml` file deve contenere righe che eseguano le seguenti operazioni:

1. Fai riferimento a un'immagine del contenitore di compilazione con il runtime necessario tra le immagini disponibili. L'esempio seguente utilizza l'immagine del contenitore di `public.ecr.aws/sam/build-nodejs20.x` compilazione.

2. Configura le fasi della pipeline per eseguire i comandi CLI (AWS SAM Command Line Interface) necessari. L'esempio seguente esegue due AWS SAMCLI comandi: sam build e sam deploy (con le opzioni necessarie).

L'esempio presuppone che abbiate dichiarato tutte le funzioni e i livelli nel file AWS SAM modello con`runtime: nodejs20.x`.

```
image: public.ecr.aws/sam/build-nodejs20.x

pipelines:
  branches:
    main: # branch name
      - step:
          name: Build and Package
          script:
            - sam build
            - sam deploy --no-confirm-changeset --no-fail-on-empty-changeset
```

Per un elenco delle immagini dei container di build Amazon Elastic Container Registry (Amazon ECR) disponibili per diversi runtime, consulta. Archivi di immagini per AWS SAM

Utilizzo di Jenkins per la distribuzione con AWS SAM

Per configurare la pipeline Jenkins per automatizzare la compilazione e la distribuzione dell' AWS SAM applicazione, è `Jenkinsfile` necessario contenere righe che eseguano le seguenti operazioni:

1. Fai riferimento a un'immagine del contenitore di compilazione con il runtime necessario tra le immagini disponibili. L'esempio seguente utilizza l'immagine del contenitore di `public.ecr.aws/sam/build-nodejs20.x` compilazione.
2. Configura le fasi della pipeline per eseguire i comandi CLI (AWS SAM Command Line Interface) necessari. L'esempio seguente esegue due AWS SAMCLI comandi: sam build e sam deploy (con le opzioni necessarie).

L'esempio presuppone che abbiate dichiarato tutte le funzioni e i livelli nel file AWS SAM modello con`runtime: nodejs20.x`.

```
pipeline {
```

```
    agent { docker { image 'public.ecr.aws/sam/build-nodejs20.x' } }
    stages {
        stage('build') {
            steps {
                sh 'sam build'
                sh 'sam deploy --no-confirm-changeset --no-fail-on-empty-changeset'
            }
        }
    }
}
```

Per un elenco delle immagini dei container di build Amazon Elastic Container Registry (Amazon ECR) disponibili per diversi runtime, consulta. Archivi di immagini per AWS SAM

Utilizzo di GitLab CI/CD per la distribuzione con AWS SAM

Per configurare la GitLabpipeline in modo da automatizzare la creazione e la distribuzione dell' AWS SAM applicazione, il `gitlab-ci.yml` file deve contenere righe che eseguano le seguenti operazioni:

1. Fai riferimento a un'immagine del contenitore di compilazione con il runtime necessario tra le immagini disponibili. L'esempio seguente utilizza l'immagine del contenitore di `public.ecr.aws/sam/build-nodejs20.x` compilazione.
2. Configura le fasi della pipeline per eseguire i comandi CLI (AWS SAM Command Line Interface) necessari. L'esempio seguente esegue due AWS SAMCLI comandi: sam build e sam deploy (con le opzioni necessarie).

L'esempio presuppone che abbiate dichiarato tutte le funzioni e i livelli nel file AWS SAM modello con`runtime: nodejs20.x`.

```
image: public.ecr.aws/sam/build-nodejs20.x
deploy:
  script:
    - sam build
    - sam deploy --no-confirm-changeset --no-fail-on-empty-changeset
```

Per un elenco delle immagini dei container di build Amazon Elastic Container Registry (Amazon ECR) disponibili per diversi runtime, consulta. Archivi di immagini per AWS SAM

Utilizzo GitHub delle azioni con cui eseguire la distribuzione AWS SAM

Per configurare la GitHubpipeline in modo da automatizzare la creazione e la distribuzione dell' AWS SAM applicazione, è necessario innanzitutto installare l'interfaccia a riga di AWS SAM comando (CLI) sull'host. Puoi utilizzare GitHub Actions nel tuo GitHub flusso di lavoro per facilitare questa configurazione.

Il seguente GitHub flusso di lavoro di esempio configura un host Ubuntu utilizzando una serie di GitHub azioni, quindi esegue AWS SAMCLI i comandi per creare e distribuire un' AWS SAM applicazione:

```
on:
  push:
    branches:
      - main
jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
      - uses: actions/setup-python@v3
      - uses: aws-actions/setup-sam@v2
      - uses: aws-actions/configure-aws-credentials@v1
        with:
          aws-access-key-id: ${{ secrets.AWS_ACCESS_KEY_ID }}
          aws-secret-access-key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
          aws-region: us-east-2
      - run: sam build --use-container
      - run: sam deploy --no-confirm-changeset --no-fail-on-empty-changeset
```

Per un elenco delle immagini dei container di build Amazon Elastic Container Registry (Amazon ECR) disponibili per diversi runtime, consulta. Archivi di immagini per AWS SAM

Come utilizzare l'autenticazione OIDC con le pipeline AWS SAM

AWS Serverless Application Model (AWS SAM) supporta l'autenticazione utente OpenID Connect (OIDC) per Bitbucket, GitHub Actions e l'integrazione GitLab continua e la distribuzione continua (account CI/CD) platforms. With this support, you can use authorized CI/CD utente) da una di queste piattaforme per gestire le pipeline di applicazioni serverless. Altrimenti, dovresti creare e gestire più utenti AWS Identity and Access Management (IAM) per controllare l'accesso alle pipeline. AWS SAM

Configura OIDC con pipeline AWS SAM

Durante il processo `sam pipeline bootstrap` di configurazione, procedi come segue per configurare OIDC con la tua pipeline. AWS SAM

1. Quando viene richiesto di scegliere un provider di identità, seleziona OIDC.
2. Quindi, seleziona un provider OIDC supportato.
3. Inserisci l'URL del provider OIDC, iniziando con. **`https://`**

> **Note**
>
> AWS SAM fa riferimento a questo URL quando genera il tipo di `AWS::IAM::OIDCProvider` risorsa.

4. Quindi, segui le istruzioni e inserisci le informazioni sulla CI/CD piattaforma necessarie per accedere alla piattaforma selezionata. Questi dettagli variano in base alla piattaforma e possono includere:
   - ID cliente OIDC.
   - Nome del repository di codici o identificatore univoco universale (UUID).
   - Nome del gruppo o dell'organizzazione associato al repository.
   - GitHub organizzazione a cui appartiene il repository di codice.
   - GitHub nome del repository.
   - Filiale da cui verranno effettuate le distribuzioni.
5. AWS SAM visualizza un riepilogo della configurazione OIDC inserita. Immettete il numero di un'impostazione per modificarla oppure premete Enter per continuare.
6. Quando viene richiesto di confermare la creazione delle risorse necessarie per supportare la connessione OIDC inserita, premi Y per continuare.

AWS SAM genera una `AWS::IAM::OIDCProvider` AWS CloudFormation risorsa con la configurazione fornita che assume il ruolo di esecuzione della pipeline. Per ulteriori informazioni su questo tipo di CloudFormation risorse, consulta AWS: :IAM:: OIDCProvider nella Guida per l'AWS CloudFormation utente.

> **Note**
>
> Se la risorsa del provider di identità (IdP) esiste già nel tuo Account AWS, vi AWS SAM fa riferimento invece di creare una nuova risorsa.

Esempio

Di seguito è riportato un esempio di configurazione di OIDC con pipeline. AWS SAM

```
Select a permissions provider:
    1 - IAM (default)
    2 - OpenID Connect (OIDC)
Choice (1, 2): 2
Select an OIDC provider:
    1 - GitHub Actions
    2 - GitLab
    3 - Bitbucket
Choice (1, 2, 3): 1
Enter the URL of the OIDC provider [https://token.actions.githubusercontent.com]:
Enter the OIDC client ID (sometimes called audience) [sts.amazonaws.com]:
Enter the GitHub organization that the code repository belongs to. If there is no
 organization enter your username instead: my-org
Enter GitHub repository name: testing
Enter the name of the branch that deployments will occur from [main]:

[3] Reference application build resources
Enter the pipeline execution role ARN if you have previously created one, or we will
 create one for you []:
Enter the CloudFormation execution role ARN if you have previously created one, or we
 will create one for you []:
Please enter the artifact bucket ARN for your Lambda function. If you do not have a
 bucket, we will create one for you []:
Does your application contain any IMAGE type Lambda functions? [y/N]:

[4] Summary
Below is the summary of the answers:
    1 - Account: 123456
    2 - Stage configuration name: dev
    3 - Region: us-east-1
    4 - OIDC identity provider URL: https://token.actions.githubusercontent.com
```

```
    5 - OIDC client ID: sts.amazonaws.com
    6 - GitHub organization: my-org
    7 - GitHub repository: testing
    8 - Deployment branch: main
    9 - Pipeline execution role: [to be created]
    10 - CloudFormation execution role: [to be created]
    11 - Artifacts bucket: [to be created]
    12 - ECR image repository: [skipped]
Press enter to confirm the values above, or select an item to edit the value:

This will create the following required resources for the 'dev' configuration:
    - IAM OIDC Identity Provider
    - Pipeline execution role
    - CloudFormation execution role
    - Artifact bucket
Should we proceed with the creation? [y/N]:
```

Ulteriori informazioni

Per ulteriori informazioni sull'utilizzo di OIDC con AWS SAM pipeline, vedere. sam pipeline bootstrap

Introduzione all'uso sam sync della sincronizzazione con Cloud AWS

Il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam sync` fornisce opzioni per sincronizzare rapidamente le modifiche dell'applicazione locale con. Cloud AWS`sam sync`Utilizzatelo quando sviluppate le vostre applicazioni per:

1. Rileva e sincronizza automaticamente le modifiche locali su Cloud AWS.
2. Personalizza le modifiche locali sincronizzate con. Cloud AWS
3. Prepara la tua applicazione nel cloud per il test e la convalida.

Con`sam sync`, puoi creare un flusso di lavoro di sviluppo rapido che riduce il tempo necessario per sincronizzare le modifiche locali nel cloud per test e convalida.

> **Note**
>
> Il `sam sync` comando è consigliato per gli ambienti di sviluppo. Per gli ambienti di produzione, si consiglia di utilizzare `sam deploy` o configurare una pipeline di integrazione

e distribuzione continue (CI/CD). Per ulteriori informazioni, consulta Implementa la tua applicazione e le tue risorse con AWS SAM.

Il `sam sync` comando fa parte di. AWS SAM Accelerate AWS SAM Accelerateforniscе strumenti che è possibile utilizzare per velocizzare l'esperienza di sviluppo e test di applicazioni serverless in. Cloud AWS

Argomenti

- Rileva e sincronizza automaticamente le modifiche locali al Cloud AWS
- Personalizza le modifiche locali da sincronizzare con Cloud AWS
- Prepara la tua applicazione nel cloud per il test e la convalida
- Opzioni per il comando sam sync
- Risoluzione dei problemi
- Esempi
- Ulteriori informazioni

Rileva e sincronizza automaticamente le modifiche locali al Cloud AWS

Esegui `sam sync` con l'`--watch`opzione per iniziare a sincronizzare l' Cloud AWS applicazione con. Questa operazione esegue le seguenti operazioni:

1. Crea la tua applicazione: questo processo è simile all'utilizzo del `sam build` comando.
2. Distribuisci l'applicazione: AWS SAMCLI distribuisce l'applicazione AWS CloudFormation utilizzando le impostazioni predefinite. Vengono utilizzati i seguenti valori predefiniti:
   a. AWS credenziali e impostazioni di configurazione generali presenti nella cartella `.aws` utente.
   b. impostazioni di distribuzione dell'applicazione disponibili nel `samconfig.toml` file dell'applicazione.

   Se non riesci a trovare i valori predefiniti, ti AWS SAMCLI informerà e chiuderà il processo di sincronizzazione.
3. Controlla le modifiche locali: AWS SAMCLI rimane in esecuzione e controlla le modifiche locali all'applicazione. Questo è ciò che offre l'`--watch`opzione.

Questa opzione può essere attivata per impostazione predefinita. Per i valori predefiniti, consultate il `samconfig.toml` file dell'applicazione. Di seguito è riportato un esempio del file .

```
...
[default.sync]
[default.sync.parameters]
watch = true
...
```

4. Sincronizza le modifiche locali con Cloud AWS: quando apporti modifiche locali, AWS SAMCLI rileva e sincronizza tali modifiche Cloud AWS tramite il metodo più rapido disponibile. A seconda del tipo di modifica, può verificarsi quanto segue:
   a. Se la risorsa aggiornata supporta il AWS servizio APIs, la AWS SAMCLI utilizzeranno per distribuire le modifiche. Ciò si traduce in una sincronizzazione rapida per aggiornare la risorsa in. Cloud AWS
   b. Se la risorsa aggiornata non supporta il AWS servizio APIs, AWS SAMCLI eseguirà una CloudFormation distribuzione. Questo aggiorna l'intera applicazione in Cloud AWS. Sebbene non sia altrettanto rapido, evita di dover avviare manualmente una distribuzione.

Poiché il `sam sync` comando aggiorna automaticamente l'applicazione in Cloud AWS, è consigliato solo per gli ambienti di sviluppo. Quando eseguisam `sync`, ti verrà chiesto di confermare:

```
**The sync command should only be used against a development stack**.

Confirm that you are synchronizing a development stack.

Enter Y to proceed with the command, or enter N to cancel:
 [Y/n]: ENTER
```

Personalizza le modifiche locali da sincronizzare con Cloud AWS

Fornisci opzioni per personalizzare le modifiche locali sincronizzate con. Cloud AWS Ciò può velocizzare il tempo necessario per visualizzare le modifiche locali nel cloud per i test e la convalida.

Ad esempio, offri la `--code` possibilità di sincronizzare solo le modifiche al codice, come il codice di AWS Lambda funzione. Durante lo sviluppo, se ti concentri specificamente sul codice Lambda, le modifiche verranno trasferite rapidamente nel cloud per il test e la convalida. Di seguito è riportato un esempio:

```
$ sam sync --code --watch
```

Per sincronizzare solo le modifiche al codice per una funzione o un layer Lambda specifico, utilizzate l'`--resource-id`opzione. Di seguito è riportato un esempio:

```
$ sam sync --code --resource-id HelloWorldFunction --resource-id HelloWorldLayer
```

Prepara la tua applicazione nel cloud per il test e la convalida

Il `sam sync` comando trova automaticamente il metodo più rapido disponibile per aggiornare l'applicazione in. Cloud AWS Ciò può velocizzare i flussi di lavoro di sviluppo e test sul cloud. Utilizzando il AWS servizio APIs, puoi sviluppare, sincronizzare e testare rapidamente le risorse supportate. Per un esempio pratico, consulta il Modulo 6 - AWS SAM Accelerare in The Complete Workshop. AWS SAM

Opzioni per il comando sam sync

Di seguito sono riportate alcune delle opzioni principali che è possibile utilizzare per modificare il `sam sync` comando. Per un elenco di tutte le opzioni, vederesam sync.

Eseguire una distribuzione una tantum CloudFormation

Usa l'`--no-watch`opzione per disattivare la sincronizzazione automatica. Di seguito è riportato un esempio:

```
$ sam sync --no-watch
```

AWS SAMCLIEseguirà una distribuzione una tantum. CloudFormation Questo comando raggruppa le azioni eseguite dai `sam deploy` comandi `sam build` and.

Salta la distribuzione iniziale CloudFormation

È possibile personalizzare se è necessaria una CloudFormation distribuzione ogni volta che `sam sync` viene eseguita.

- Provvedere `--no-skip-deploy-sync` a richiedere una CloudFormation distribuzione ogni volta `sam sync` che viene eseguita. Ciò garantisce la sincronizzazione dell'infrastruttura locale con CloudFormation, prevenendo eventuali deviazioni. L'utilizzo di questa opzione aggiunge ulteriore tempo al flusso di lavoro di sviluppo e test.

- Fornisci `--skip-deploy-sync` per rendere facoltativa CloudFormation la distribuzione. AWS SAMCLIConfronterà il AWS SAM modello locale con il CloudFormation modello distribuito e salterà la CloudFormation distribuzione iniziale se non viene rilevata una modifica. Saltare la CloudFormation distribuzione può farti risparmiare tempo durante la sincronizzazione delle modifiche locali con. Cloud AWS

  Se non viene rilevata alcuna modifica, AWS SAMCLI eseguirà comunque una CloudFormation distribuzione nei seguenti scenari:

  - Se sono trascorsi almeno 7 giorni dall'ultima CloudFormation distribuzione.
  - Se viene rilevato un gran numero di modifiche al codice della funzione Lambda, la CloudFormation distribuzione è il metodo più rapido per aggiornare l'applicazione.

Di seguito è riportato un esempio:

```
$ sam sync --skip-deploy-sync
```

Sincronizza una risorsa da uno stack annidato

Per sincronizzare una risorsa da uno stack annidato

1. Fornisci lo stack principale utilizzando. `--stack-name`
2. Identifica la risorsa nello stack annidato utilizzando il seguente formato:. *`nestedStackId/resourceId`*
3. Fornisci la risorsa nello stack nidificato utilizzando. `--resource-id`

   Di seguito è riportato un esempio:

   ```
   $ sam sync --code --stack-name sam-app --resource-id myNestedStack/
   HelloWorldFunction
   ```

Per ulteriori informazioni sulla creazione di applicazioni annidate, vedere. Riutilizza codice e risorse utilizzando applicazioni annidate in AWS SAM

Specificare uno CloudFormation stack specifico da aggiornare

Per specificare uno CloudFormation stack specifico da aggiornare, fornisci l'`--stack-name`opzione. Di seguito è riportato un esempio:

```
$ sam sync --stack-name dev-sam-app
```

Accelera i tempi di compilazione creando il tuo progetto nella cartella dei sorgenti

Per i runtime e i metodi di compilazione supportati, puoi utilizzare l'`--build-in-source`opzione per creare il tuo progetto direttamente nella cartella di origine. Per impostazione predefinita, AWS SAM CLI le build si trovano in una directory temporanea, che prevede la copia del codice sorgente e dei file di progetto. Con`--build-in-source`, AWS SAM CLI le build vengono create direttamente nella cartella di origine, il che accelera il processo di compilazione eliminando la necessità di copiare i file in una directory temporanea.

Per un elenco dei runtime e dei metodi di compilazione supportati, consulta. `--build-in-source`

Specificate i file e le cartelle che non avvieranno una sincronizzazione

Utilizzate l'`--watch-exclude`opzione per specificare qualsiasi file o cartella che non avvierà una sincronizzazione una volta aggiornato. Per ulteriori informazioni su questa opzione, consulta `--watch-exclude`.

Di seguito è riportato un esempio che esclude il `package-lock.json` file associato alla nostra `HelloWorldFunction` funzione:

```
$ sam sync --watch --watch-exclude HelloWorldFunction=package-lock.json
```

Quando viene eseguito questo comando, AWS SAM CLI avvierà il processo di sincronizzazione. Questo include gli output seguenti:

- Esegui `sam build` per creare le tue funzioni e preparare l'applicazione per la distribuzione.
- Esegui `sam deploy` per distribuire la tua applicazione.
- Controlla le modifiche alla tua applicazione.

Quando modifichiamo il `package-lock.json` file, AWS SAM CLI non avvieranno una sincronizzazione. Quando un altro file viene aggiornato, AWS SAM CLI avvierà una sincronizzazione, che includerà il `package-lock.json` file.

Di seguito è riportato un esempio di specificazione di una funzione Lambda di uno stack secondario:

```
$ sam sync --watch --watch-exclude ChildStackA/MyFunction=database.sqlite3
```

Risoluzione dei problemi

Per risolvere il problema, vedere. AWS SAMCLI AWS SAMCLIrisoluzione dei problemi

Esempi

Utilizzo di sam sync per aggiornare l'applicazione Hello World

In questo esempio, iniziamo inizializzando l'applicazione Hello World di esempio. Per ulteriori informazioni su questa applicazione, consulta. Tutorial: Implementa un'applicazione Hello World con AWS SAM

L'esecuzione `sam sync` avvia il processo di creazione e distribuzione.

```
$ sam sync

The SAM CLI will use the AWS Lambda, Amazon API Gateway, and AWS StepFunctions APIs to
 upload your code without
performing a CloudFormation deployment. This will cause drift in your CloudFormation
 stack.
**The sync command should only be used against a development stack**.

Confirm that you are synchronizing a development stack.

Enter Y to proceed with the command, or enter N to cancel:
 [Y/n]:
Queued infra sync. Waiting for in progress code syncs to complete...
Starting infra sync.
Manifest file is changed (new hash: 3298f13049d19cffaa37ca931dd4d421) or dependency
 folder (.aws-sam/deps/0663e6fe-a888-4efb-b908-e2344261e9c7) is missing for
 (HelloWorldFunction), downloading dependencies and copying/building source
Building codeuri: /Users/.../Demo/sync/sam-app/hello_world runtime: python3.9 metadata:
 {} architecture: x86_64 functions: HelloWorldFunction
Running PythonPipBuilder:CleanUp
Running PythonPipBuilder:ResolveDependencies
Running PythonPipBuilder:CopySource

Build Succeeded

Successfully packaged artifacts and wrote output template to file /var/
folders/45/5ct135bx3fn2551_ptl5g6_80000gr/T/tmpx_5t4u3f.
Execute the following command to deploy the packaged template
```

```
sam deploy --template-file /var/folders/45/5ct135bx3fn2551_ptl5g6_80000gr/T/tmpx_5t4u3f
 --stack-name <YOUR STACK NAME>


    Deploying with following values
    ===============================
    Stack name                   : sam-app
    Region                       : us-west-2
    Disable rollback             : False
    Deployment s3 bucket         : aws-sam-cli-managed-default-samclisam-s3-demo-
bucket-1a4x26zbcdkqr
    Capabilities                 : ["CAPABILITY_NAMED_IAM", "CAPABILITY_AUTO_EXPAND"]
    Parameter overrides          : {}
    Signing Profiles             : null

Initiating deployment
=====================



2023-03-17 11:17:19 - Waiting for stack create/update to complete

CloudFormation events from stack operations (refresh every 0.5 seconds)
-----------------------------------------------------------------------------------------------
ResourceStatus                      ResourceType
 LogicalResourceId                   ResourceStatusReason
-----------------------------------------------------------------------------------------------
CREATE_IN_PROGRESS                  AWS::CloudFormation::Stack          sam-app
                     Transformation succeeded
CREATE_IN_PROGRESS                  AWS::CloudFormation::Stack
 AwsSamAutoDependencyLayerNestedSt   -
                                                                         ack
CREATE_IN_PROGRESS                  AWS::IAM::Role
 HelloWorldFunctionRole              -
CREATE_IN_PROGRESS                  AWS::IAM::Role
 HelloWorldFunctionRole              Resource creation Initiated
CREATE_IN_PROGRESS                  AWS::CloudFormation::Stack
 AwsSamAutoDependencyLayerNestedSt   Resource creation Initiated
                                                                         ack
CREATE_COMPLETE                     AWS::IAM::Role
 HelloWorldFunctionRole              -
CREATE_COMPLETE                     AWS::CloudFormation::Stack
 AwsSamAutoDependencyLayerNestedSt   -
                                                                         ack
```

```
CREATE_IN_PROGRESS                  AWS::Lambda::Function
 HelloWorldFunction                  -
CREATE_IN_PROGRESS                  AWS::Lambda::Function
 HelloWorldFunction                  Resource creation Initiated
CREATE_COMPLETE                     AWS::Lambda::Function
 HelloWorldFunction                  -
CREATE_IN_PROGRESS                  AWS::ApiGateway::RestApi
 ServerlessRestApi                   -
CREATE_IN_PROGRESS                  AWS::ApiGateway::RestApi
 ServerlessRestApi                   Resource creation Initiated
CREATE_COMPLETE                     AWS::ApiGateway::RestApi
 ServerlessRestApi                   -
CREATE_IN_PROGRESS                  AWS::ApiGateway::Deployment
 ServerlessRestApiDeployment47fc2d   -
                                                                        5f9d
CREATE_IN_PROGRESS                  AWS::Lambda::Permission
 HelloWorldFunctionHelloWorldPermi   -
                                                                        ssionProd
CREATE_IN_PROGRESS                  AWS::Lambda::Permission
 HelloWorldFunctionHelloWorldPermi   Resource creation Initiated
                                                                        ssionProd
CREATE_IN_PROGRESS                  AWS::ApiGateway::Deployment
 ServerlessRestApiDeployment47fc2d   Resource creation Initiated
                                                                        5f9d
CREATE_COMPLETE                     AWS::ApiGateway::Deployment
 ServerlessRestApiDeployment47fc2d   -
                                                                        5f9d
CREATE_IN_PROGRESS                  AWS::ApiGateway::Stage
 ServerlessRestApiProdStage          -
CREATE_IN_PROGRESS                  AWS::ApiGateway::Stage
 ServerlessRestApiProdStage          Resource creation Initiated
CREATE_COMPLETE                     AWS::ApiGateway::Stage
 ServerlessRestApiProdStage          -
CREATE_COMPLETE                     AWS::Lambda::Permission
 HelloWorldFunctionHelloWorldPermi   -
                                                                        ssionProd
CREATE_COMPLETE                     AWS::CloudFormation::Stack          sam-app
                      -
-----------------------------------------------------------------------------------------------------------------------

CloudFormation outputs from deployed stack
-----------------------------------------------------------------------------------------------------------------------
Outputs
-----------------------------------------------------------------------------------------------------------------------
```

```
Key                 HelloWorldFunctionIamRole
Description         Implicit IAM Role created for Hello World function
Value               arn:aws:iam::012345678910:role/sam-app-HelloWorldFunctionRole-
BUFVMO2PJIYF

Key                 HelloWorldApi
Description         API Gateway endpoint URL for Prod stage for Hello World function
Value               https://pcrx5gdaof.execute-api.us-west-2.amazonaws.com/Prod/hello/

Key                 HelloWorldFunction
Description         Hello World Lambda Function ARN
Value               arn:aws:lambda:us-west-2:012345678910:function:sam-app-
HelloWorldFunction-2PlN6TPTQoco
-----------------------------------------------------------------------------------------------------
Stack creation succeeded. Sync infra completed.

Infra sync completed.
CodeTrigger not created as CodeUri or DefinitionUri is missing for ServerlessRestApi.
```

Una volta completata la distribuzione, modifichiamo il `HelloWorldFunction` codice. AWS SAMCLIRileva questa modifica e sincronizza la nostra applicazione con. Cloud AWS Poiché AWS Lambda supporta il AWS servizio APIs, viene eseguita una sincronizzazione rapida.

```
Syncing Lambda Function HelloWorldFunction...
Manifest is not changed for (HelloWorldFunction), running incremental build
Building codeuri: /Users/.../Demo/sync/sam-app/hello_world runtime: python3.9 metadata:
 {} architecture: x86_64 functions: HelloWorldFunction
Running PythonPipBuilder:CopySource
Finished syncing Lambda Function HelloWorldFunction.
```

Successivamente, modifichiamo il nostro endpoint API nel AWS SAM modello dell'applicazione. Passiamo `/hello` a. `/helloworld`

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  HelloWorldFunction:
    ...
    Properties:
      ...
      Events:
```

```
        HelloWorld:
          Type: Api
          Properties:
            Path: /helloworld
            Method: get
```

Poiché la risorsa Amazon API Gateway non supporta l'API del AWS servizio, esegue AWS SAMCLI automaticamente una CloudFormation distribuzione. Di seguito è riportato un esempio di output:

```
Queued infra sync. Waiting for in progress code syncs to complete...
Starting infra sync.
Manifest is not changed for (HelloWorldFunction), running incremental build
Building codeuri: /Users/.../Demo/sync/sam-app/hello_world runtime: python3.9 metadata:
 {} architecture: x86_64 functions: HelloWorldFunction
Running PythonPipBuilder:CopySource

Build Succeeded

Successfully packaged artifacts and wrote output template to file /var/
folders/45/5ct135bx3fn2551_ptl5g6_80000gr/T/tmpuabo0jb9.
Execute the following command to deploy the packaged template
sam deploy --template-file /var/folders/45/5ct135bx3fn2551_ptl5g6_80000gr/T/tmpuabo0jb9
 --stack-name <YOUR STACK NAME>


    Deploying with following values
    ===============================
    Stack name                   : sam-app
    Region                       : us-west-2
    Disable rollback             : False
    Deployment s3 bucket         : aws-sam-cli-managed-default-samclisam-s3-demo-
bucket-1a4x26zbcdkqr
    Capabilities                 : ["CAPABILITY_NAMED_IAM", "CAPABILITY_AUTO_EXPAND"]
    Parameter overrides          : {}
    Signing Profiles             : null

Initiating deployment
=====================



2023-03-17 14:41:18 - Waiting for stack create/update to complete

CloudFormation events from stack operations (refresh every 0.5 seconds)
```

```
------------------------------------------------------------------------------------------------------
ResourceStatus                      ResourceType
 LogicalResourceId                   ResourceStatusReason
------------------------------------------------------------------------------------------------------
UPDATE_IN_PROGRESS                  AWS::CloudFormation::Stack          sam-app
                     Transformation succeeded
UPDATE_IN_PROGRESS                  AWS::CloudFormation::Stack
 AwsSamAutoDependencyLayerNestedSt   -
                                                                         ack
UPDATE_COMPLETE                     AWS::CloudFormation::Stack
 AwsSamAutoDependencyLayerNestedSt   -
                                                                         ack
UPDATE_IN_PROGRESS                  AWS::ApiGateway::RestApi
 ServerlessRestApi                   -
UPDATE_COMPLETE                     AWS::ApiGateway::RestApi
 ServerlessRestApi                   -
CREATE_IN_PROGRESS                  AWS::ApiGateway::Deployment
 ServerlessRestApiDeployment8cf30e   -
                                                                         d3cd
UPDATE_IN_PROGRESS                  AWS::Lambda::Permission
 HelloWorldFunctionHelloWorldPermi   Requested update requires the
                                                                         ssionProd
                     creation of a new physical

                     resource; hence creating one.
UPDATE_IN_PROGRESS                  AWS::Lambda::Permission
 HelloWorldFunctionHelloWorldPermi   Resource creation Initiated
                                                                         ssionProd
CREATE_IN_PROGRESS                  AWS::ApiGateway::Deployment
 ServerlessRestApiDeployment8cf30e   Resource creation Initiated
                                                                         d3cd
CREATE_COMPLETE                     AWS::ApiGateway::Deployment
 ServerlessRestApiDeployment8cf30e   -
                                                                         d3cd
UPDATE_IN_PROGRESS                  AWS::ApiGateway::Stage
 ServerlessRestApiProdStage          -
UPDATE_COMPLETE                     AWS::ApiGateway::Stage
 ServerlessRestApiProdStage          -
UPDATE_COMPLETE                     AWS::Lambda::Permission
 HelloWorldFunctionHelloWorldPermi   -
                                                                         ssionProd
UPDATE_COMPLETE_CLEANUP_IN_PROGRE   AWS::CloudFormation::Stack          sam-app
                     -
SS
```

```
DELETE_IN_PROGRESS                  AWS::Lambda::Permission
 HelloWorldFunctionHelloWorldPermi   -
                                                                        ssionProd
DELETE_IN_PROGRESS                  AWS::ApiGateway::Deployment
 ServerlessRestApiDeployment47fc2d   -
                                                                        5f9d
DELETE_COMPLETE                     AWS::ApiGateway::Deployment
 ServerlessRestApiDeployment47fc2d   -
                                                                        5f9d
UPDATE_COMPLETE                     AWS::CloudFormation::Stack
 AwsSamAutoDependencyLayerNestedSt   -
                                                                        ack
DELETE_COMPLETE                     AWS::Lambda::Permission
 HelloWorldFunctionHelloWorldPermi   -
                                                                        ssionProd
UPDATE_COMPLETE                     AWS::CloudFormation::Stack          sam-app
                      -
-----------------------------------------------------------------------------------------------------------

CloudFormation outputs from deployed stack
-----------------------------------------------------------------------------------------------------------
Outputs
-----------------------------------------------------------------------------------------------------------
Key                 HelloWorldFunctionIamRole
Description         Implicit IAM Role created for Hello World function
Value               arn:aws:iam::012345678910:role/sam-app-HelloWorldFunctionRole-
BUFVMO2PJIYF

Key                 HelloWorldApi
Description         API Gateway endpoint URL for Prod stage for Hello World function
Value               https://pcrx5gdaof.execute-api.us-west-2.amazonaws.com/Prod/hello/

Key                 HelloWorldFunction
Description         Hello World Lambda Function ARN
Value               arn:aws:lambda:us-west-2:012345678910:function:sam-app-
HelloWorldFunction-2PlN6TPTQoco
-----------------------------------------------------------------------------------------------------------



Stack update succeeded. Sync infra completed.

Infra sync completed.
```

Ulteriori informazioni

Per una descrizione di tutte le `sam sync` opzioni, consultasam sync.

Monitora la tua applicazione serverless con AWS SAM

Dopo aver distribuito l'applicazione serverless, è possibile monitorarla per fornire informazioni dettagliate sulle sue operazioni e rilevare anomalie, il che può facilitare la risoluzione dei problemi. Questa sezione fornisce dettagli sul monitoraggio dell'applicazione serverless. Ciò include informazioni su come configurare Amazon CloudWatch per avvisarti quando rileva anomalie. Fornisce inoltre informazioni sull'utilizzo dei log, tra cui l'evidenziazione degli errori e suggerimenti per la visualizzazione, il filtraggio, il recupero e l'archiviazione dei log.

Argomenti

- Utilizzo di CloudWatch Application Insights per monitorare le applicazioni AWS SAM serverless
- Lavorare con i log in AWS SAM

Utilizzo di CloudWatch Application Insights per monitorare le applicazioni AWS SAM serverless

Amazon CloudWatch Application Insights ti aiuta a monitorare le AWS risorse delle tue applicazioni per identificare potenziali problemi. Può analizzare i dati relativi alle AWS risorse per individuare eventuali segni di problemi e creare dashboard automatizzati per visualizzarli. Puoi configurare CloudWatch Application Insights per l'uso con le tue AWS Serverless Application Model (AWS SAM) applicazioni. Per ulteriori informazioni su CloudWatch Application Insights, consulta Amazon CloudWatch Application Insights nella Amazon CloudWatch User Guide.

Argomenti

- Configurazione di CloudWatch Application Insights con AWS SAM
- Fasi successive

Configurazione di CloudWatch Application Insights con AWS SAM

Configura CloudWatch Application Insights per AWS SAM le tue applicazioni tramite l'interfaccia a riga di AWS SAM comando (AWS SAMCLI) o tramite i tuoi AWS SAM modelli.

Configura tramite AWS SAMCLI

Quando inizializzi l'applicazione consam init, attiva CloudWatch Application Insights tramite il flusso interattivo o utilizzando l'opzione. --application-insights

Per attivare CloudWatch Application Insights tramite il flusso AWS SAMCLI interattivo, inserisci **y** quando richiesto.

```
Would you like to enable monitoring using CloudWatch Application Insights?
For more info, please view https://docs.aws.amazon.com/AmazonCloudWatch/latest/
monitoring/cloudwatch-application-insights.html [y/N]:
```

Per attivare CloudWatch Application Insights con l'--application-insightsopzione, procedi come segue.

```
sam init --application-insights
```

Per ulteriori informazioni sull'utilizzo del sam init comando, consultasam init.

Configurazione tramite AWS SAM modelli

Attiva CloudWatch Application Insights definendo `AWS::ApplicationInsights::Application` le risorse `AWS::ResourceGroups::Group` e i AWS SAM modelli.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  ApplicationResourceGroup:
    Type: AWS::ResourceGroups::Group
    Properties:
      Name:
        Fn::Join:
        - ''
        - - ApplicationInsights-SAM-
        - Ref: AWS::StackName
      ResourceQuery:
        Type: CLOUDFORMATION_STACK_1_0
  ApplicationInsightsMonitoring:
    Type: AWS::ApplicationInsights::Application
    Properties:
      ResourceGroupName:
```

```
        Fn::Join:
          - ''
          - - ApplicationInsights-SAM-
          - Ref: AWS::StackName
        AutoConfigurationEnabled: 'true'
    DependsOn: ApplicationResourceGroup
```

- `AWS::ResourceGroups::Group`— Crea un gruppo per organizzare le AWS risorse al fine di gestire e automatizzare le attività su un gran numero di risorse contemporaneamente. Qui puoi creare un gruppo di risorse da utilizzare con CloudWatch Application Insights. Per ulteriori informazioni su questo tipo di risorsa, `AWS::ResourceGroups::Group`consulta la Guida AWS CloudFormation per l'utente.
- `AWS::ApplicationInsights::Application`— Configura CloudWatch Application Insights per il gruppo di risorse. Per ulteriori informazioni su questo tipo di risorsa, consulta `AWS::ApplicationInsights::Application`la Guida per l'AWS CloudFormation utente.

Entrambe le risorse vengono trasferite automaticamente al CloudFormation momento della distribuzione dell'applicazione. È possibile utilizzare la CloudFormation sintassi del AWS SAM modello per configurare ulteriormente CloudWatch Application Insights. Per ulteriori informazioni, consulta Usare CloudFormation i modelli nella Amazon CloudWatch User Guide.

Quando usi il sam init --application-insights comando, entrambe queste risorse vengono generate automaticamente nel tuo AWS SAM modello. Ecco un esempio di modello generato.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: >
  sam-app-test

  Sample SAM Template for sam-app-test

# More info about Globals: https://github.com/awslabs/serverless-application-model/
blob/master/docs/globals.rst
Globals:
  Function:
    Timeout: 3
    MemorySize: 128

Resources:
  HelloWorldFunction:
```

```
    Type: AWS::Serverless::Function # More info about Function Resource:
 https://github.com/awslabs/serverless-application-model/blob/master/
versions/2016-10-31.md#awsserverlessfunction
    Properties:
      CodeUri: hello_world/
      Handler: app.lambda_handler
      Runtime: python3.9
      Architectures:
      - x86_64
      Events:
        HelloWorld:
          Type: Api # More info about API Event Source: https://github.com/awslabs/
serverless-application-model/blob/master/versions/2016-10-31.md#api
          Properties:
            Path: /hello
            Method: get

  ApplicationResourceGroup:
    Type: AWS::ResourceGroups::Group
    Properties:
      Name:
        Fn::Join:
        - ''
        - - ApplicationInsights-SAM-
        - Ref: AWS::StackName
      ResourceQuery:
       Type: CLOUDFORMATION_STACK_1_0
  ApplicationInsightsMonitoring:
    Type: AWS::ApplicationInsights::Application
    Properties:
      ResourceGroupName:
        Fn::Join:
        - ''
        - - ApplicationInsights-SAM-
        - Ref: AWS::StackName
      AutoConfigurationEnabled: 'true'
    DependsOn: ApplicationResourceGroup

Outputs:
  # ServerlessRestApi is an implicit API created out of Events key under
 Serverless::Function
  # Find out more about other implicit resources you can reference within SAM
  # https://github.com/awslabs/serverless-application-model/blob/master/docs/internals/
generated_resources.rst#api
```

```
  HelloWorldApi:
    Description: API Gateway endpoint URL for Prod stage for Hello World function
    Value: !Sub "https://${ServerlessRestApi}.execute-api.${AWS::Region}.amazonaws.com/
Prod/hello/"
  HelloWorldFunction:
    Description: Hello World Lambda Function ARN
    Value: !GetAtt HelloWorldFunction.Arn
  HelloWorldFunctionIamRole:
    Description: Implicit IAM Role created for Hello World function
    Value: !GetAtt HelloWorldFunctionRole.Arn
```

Fasi successive

Dopo aver configurato CloudWatch Application Insights, utilizzalo sam build per creare l'applicazione e sam deploy distribuirla. Tutte le risorse supportate da CloudWatch Application Insights verranno configurate per il monitoraggio.

- Per un elenco delle risorse supportate, consulta Supported logs and metrics nella Amazon CloudWatch User Guide.
- Per informazioni su come accedere ad CloudWatch Application Insights, consulta Access CloudWatch Application Insights nella Amazon CloudWatch User Guide.

Lavorare con i log in AWS SAM

Per semplificare la risoluzione dei problemi, AWS SAMCLI ha un comando chiamato`sam logs`. Questo comando consente di recuperare i log generati dalla funzione Lambda dalla riga di comando.

> **Note**
>
> Il `sam logs` comando funziona per tutte le AWS Lambda funzioni, non solo per quelle che utilizzi per la distribuzione. AWS SAM

Recupero dei log per pila CloudFormation

Quando la tua funzione fa parte di uno CloudFormation stack, puoi recuperare i log utilizzando l'ID logico della funzione:

```
sam logs -n HelloWorldFunction --stack-name mystack
```

Recupero dei log in base al nome della funzione Lambda

In alternativa, puoi recuperare i log usando il nome della funzione:

```
sam logs -n mystack-HelloWorldFunction-1FJ8PD
```

Tronchi di coda

Aggiungi l'`--tail`opzione per attendere nuovi registri e visualizzarli man mano che arrivano. Ciò è utile durante l'implementazione o per la risoluzione di un problema di produzione.

```
sam logs -n HelloWorldFunction --stack-name mystack --tail
```

Visualizzazione dei log per un intervallo di tempo specifico

È possibile visualizzare i registri per un intervallo di tempo specifico utilizzando le opzioni `-s` and`-e`:

```
sam logs -n HelloWorldFunction --stack-name mystack -s '10min ago' -e '2min ago'
```

Filtraggio dei log

Utilizza l'`--filter`opzione per trovare rapidamente i log che corrispondono a termini, frasi o valori nei tuoi eventi di registro:

```
sam logs -n HelloWorldFunction --stack-name mystack --filter "error"
```

Nell'output, AWS SAMCLI sottolinea tutte le occorrenze della parola «errore» in modo da poter individuare facilmente la parola chiave del filtro all'interno dell'output del registro.

Evidenziazione degli errori

Quando la funzione Lambda si blocca o scade, AWS SAMCLI evidenzia il messaggio di timeout in rosso. Questo ti aiuta a localizzare facilmente esecuzioni specifiche che stanno scadendo all'interno di un enorme flusso di log in uscita.

Bella stampa in JSON

Se i tuoi messaggi di registro stampano stringhe JSON, AWS SAMCLI automaticamente Pretty stampa il JSON per aiutarti ad analizzare e comprendere visivamente il JSON.

AWS SAM riferimento

Questa sezione contiene materiale AWS SAM di riferimento. Ciò include materiale AWS SAMCLI di riferimento, come informazioni di riferimento sui AWS SAMCLI comandi e AWS SAMCLI informazioni aggiuntive, come la configurazione, il controllo della versione e la risoluzione dei problemi. Inoltre, questa sezione include informazioni di riferimento sulla AWS SAM specifica e sul AWS SAM modello, come informazioni di riferimento su connettori, archivi di immagini e distribuzioni.

AWS SAM le specifiche e il modello AWS SAM

La AWS SAM specifica è una specifica open source con licenza Apache 2.0. La versione corrente della AWS SAM specifica è disponibile in. AWS SAM modello AWS SAM la specifica viene fornita con una sintassi abbreviata semplificata utilizzata per definire le funzioni, gli eventi APIs, le configurazioni e le autorizzazioni dell'applicazione serverless.

L'interazione con le AWS SAM specifiche avviene tramite la directory del progetto AWS SAM dell'applicazione, che è costituita dalle cartelle e dai file creati quando si esegue il comando. sam init Questa directory include il AWS SAM modello, un file importante che definisce le AWS risorse. Il AWS SAM modello è un'estensione di un AWS CloudFormation modello. Per il riferimento completo ai CloudFormation modelli, consulta la sezione Riferimento ai modelli nella Guida per l'AWS CloudFormation utente.

AWS SAMCLIriferimento ai comandi

La AWS Serverless Application Model Command Line Interface (AWS SAMCLI) è uno strumento da riga di comando che puoi utilizzare con AWS SAM modelli e integrazioni di terze parti supportate per creare ed eseguire applicazioni serverless.

È possibile utilizzare i AWS SAMCLI comandi per sviluppare, testare e distribuire applicazioni serverless su. Cloud AWS Di seguito sono riportati alcuni esempi di AWS SAMCLI comandi:

- `sam init`— Se sei un AWS SAMCLI utente alle prime armi, puoi eseguire il `sam init` comando senza parametri per creare un'applicazione Hello World. Il comando genera un AWS SAM modello preconfigurato e un codice applicativo di esempio nella lingua scelta.
- `sam local invoke`e `sam local start-api` — Utilizzate questi comandi per testare il codice dell'applicazione localmente, prima di distribuirlo su. Cloud AWS

- `sam logs`— Utilizzate questo comando per recuperare i log generati dalla funzione Lambda. Questo può aiutarti a testare ed eseguire il debug dell'applicazione dopo averla distribuita su. Cloud AWS
- `sam package`— Utilizzate questo comando per raggruppare il codice dell'applicazione e le dipendenze in un pacchetto di distribuzione. È necessario il pacchetto di distribuzione per caricare l'applicazione su. Cloud AWS
- `sam deploy`— Utilizzate questo comando per distribuire l'applicazione serverless su. Cloud AWS Crea le AWS risorse e imposta le autorizzazioni e altre configurazioni definite nel modello. AWS SAM

Per istruzioni sull'installazione di AWS SAMCLI, vedere. Installa il AWS SAMCLI

AWS SAM modelli di policy

Con AWS SAM, puoi scegliere da un elenco di modelli di policy per definire le autorizzazioni della tua AWS Lambda funzione per le risorse utilizzate dall'applicazione. Per un elenco dei modelli di policy disponibili, fare riferimento aTabella dei modelli di policy. Per informazioni generali sui modelli di policy e AWS SAM, fare riferimento aAWS SAM modelli di policy.

Argomenti

- AWS SAM modello
- AWS SAMCLIriferimento al comando
- AWS SAMCLIfile di configurazione
- AWS SAM riferimento del connettore
- AWS SAM modelli di policy
- Archivi di immagini per AWS SAM
- Telemetria nel AWS SAMCLI
- Configura e gestisci l'accesso alle risorse nel tuo AWS SAM modello

AWS SAMCLIriferimento al comando

Questa sezione include informazioni di riferimento sui AWS SAMCLI comandi. Ciò include dettagli sull'utilizzo, un elenco completo delle diverse opzioni disponibili per ogni comando e informazioni

aggiuntive. Se applicabile, le informazioni aggiuntive includono dettagli come argomenti, variabili di ambiente ed eventi. Vedi ogni comando per i dettagli. Per istruzioni sull'installazione di AWS SAMCLI, vedereInstalla il AWS SAMCLI.

Argomenti

- sam build
- sam delete
- sam deploy
- sam init
- sam list
- sam local callback
- sam local execution
- sam local generate-event
- sam local invoke
- sam local start-api
- sam local start-lambda
- sam logs
- sam package
- sam pipeline bootstrap
- sam pipeline init
- sam publish
- sam remote callback
- sam remote execution
- sam remote invoke
- sam remote test-event
- sam sync
- sam traces
- sam validate

sam build

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam build`.

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'uso del AWS SAMCLI `sam build` comando, vedereIntroduzione alla costruzione con AWS SAM.

Il `sam build` comando prepara un'applicazione per i passaggi successivi del flusso di lavoro degli sviluppatori, ad esempio il test locale o la distribuzione su. Cloud AWS

Utilizzo

```
$ sam build <arguments> <options>
```

Arguments (Argomenti)

Resource ID (ID risorsa)

Opzionale. Indica AWS SAM a creare una singola risorsa dichiarata in un AWS SAM modello. Gli artefatti di compilazione per la risorsa specificata saranno gli unici disponibili per i comandi successivi nel flusso di lavoro, ad esempio e. `sam package` `sam deploy`

Opzioni

`--base-dir, -s` *`DIRECTORY`*

Risolve i percorsi relativi al codice sorgente della funzione o del livello rispetto a questa directory. Utilizzate questa opzione se desiderate modificare il modo in cui vengono risolti i percorsi relativi alle cartelle del codice sorgente. Per impostazione predefinita, i percorsi relativi vengono risolti rispetto alla posizione del AWS SAM modello.

Oltre alle risorse nell'applicazione root o nello stack che state creando, questa opzione si applica anche alle applicazioni o agli stack annidati.

Questa opzione si applica ai seguenti tipi e proprietà di risorse:

- Tipo di risorsa: `AWS::Serverless::Function` Proprietà: `CodeUri`

- Tipo di risorsa: Attributo della `AWS::Serverless::Function` risorsa: `Metadata` Voce: `DockerContext`
- Tipo di risorsa: `AWS::Serverless::LayerVersion` Proprietà: `ContentUri`
- Tipo di risorsa: `AWS::Lambda::Function` Proprietà: `Code`
- Tipo di risorsa: `AWS::Lambda::LayerVersion` Proprietà: `Content`

`--beta-features | --no-beta-features`

Consenti o nega le funzionalità beta.

`--build-dir, -b` *`DIRECTORY`*

Il percorso di una directory in cui sono archiviati gli artefatti creati. Questa directory e tutto il suo contenuto vengono rimossi con questa opzione.

`--build-image` *`TEXT`*

L'URI dell'immagine del contenitore che desideri estrarre per la build. Per impostazione predefinita, AWS SAM estrae l'immagine del contenitore da Amazon ECR Public. Usa questa opzione per estrarre l'immagine da un'altra posizione.

Puoi specificare questa opzione più volte. Ogni istanza di questa opzione può accettare una stringa o una coppia chiave-valore. Se si specifica una stringa, si tratta dell'URI dell'immagine del contenitore da utilizzare per tutte le risorse dell'applicazione. Ad esempio, `sam build --use-container --build-image amazon/aws-sam-cli-build-image-python3.8`. Se si specifica una coppia chiave-valore, la chiave è il nome della risorsa e il valore è l'URI dell'immagine del contenitore da utilizzare per quella risorsa. Ad esempio, `sam build --use-container --build-image Function1=amazon/aws-sam-cli-build-image-python3.8`. Con le coppie chiave-valore, è possibile specificare immagini di contenitori diverse per risorse diverse.

Questa opzione si applica solo se l'`--use-container`opzione è specificata, altrimenti si verificherà un errore.

`--build-in-source | --no-build-in-source`

`--build-in-source`Provvedi a creare il tuo progetto direttamente nella cartella dei sorgenti.

L'`--build-in-source`opzione supporta i seguenti runtime e metodi di compilazione:

- Runtime: qualsiasi Node.js runtime supportato dall'`sam init --runtime`opzione.

- Metodi di compilazione:`Makefile`,`esbuild`.

L'`--build-in-source`opzione non è compatibile con le seguenti opzioni:

- `--hook-name`
- `--use-container`

Default: `--no-build-in-source`

`--cached | --no-cached`

Abilita o disabilita le build memorizzate nella cache. Usa questa opzione per riutilizzare gli artefatti della build che non sono stati modificati rispetto alle build precedenti. AWS SAM valuta se sono stati modificati i file nella directory del progetto. Per impostazione predefinita, le build non vengono memorizzate nella cache. Se l'`--no-cached`opzione viene richiamata, sostituisce l'impostazione in samconfig.toml. `cached = true`

> **Note**
>
> AWS SAM non valuta se hai modificato i moduli di terze parti da cui dipende il tuo progetto, laddove non hai fornito una versione specifica. Ad esempio, se la tua funzione Python include un `requirements.txt` file con la voce `requests=1.x` e l'ultima versione del modulo di richiesta cambia da `1.1` a`1.2`, non estrae la versione più recente finché AWS SAM non esegui una build non memorizzata nella cache.

`--cache-dir`

La directory in cui vengono memorizzati gli elementi della cache quando viene specificata. `--cached` La directory cache predefinita è. `.aws-sam/cache`

`--config-env` *`TEXT`*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *`PATH`*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è "`samconfig.toml`" nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--container-env-var, -e` *TEXT*

Variabili di ambiente da passare al contenitore di compilazione. Puoi specificare questa opzione più volte. Ogni istanza di questa opzione accetta una coppia chiave-valore, dove la chiave è la variabile risorsa e ambiente e il valore è il valore della variabile di ambiente. Ad esempio: `--container-env-var Function1.GITHUB_TOKEN=TOKEN1 --container-env-var Function2.GITHUB_TOKEN=TOKEN2`.

Questa opzione si applica solo se l'`--use-container`opzione è specificata, altrimenti si verificherà un errore.

`--container-env-var-file, -ef` *PATH*

Il percorso e il nome di un file JSON che contiene valori per le variabili di ambiente del contenitore. Per ulteriori informazioni sui file delle variabili di ambiente del contenitore, vedereFile variabile di ambiente del contenitore.

Questa opzione si applica solo se l'`--use-container`opzione è specificata, altrimenti si verificherà un errore.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug che AWS SAMCLI genera e per visualizzare i timestamp.

`--docker-network` *TEXT*

Speciifica il nome o l'ID di una Docker rete esistente a cui i contenitori Docker Lambda devono connettersi, insieme alla rete bridge predefinita. Se non specificato, i contenitori Lambda si connettono solo alla rete bridge Docker predefinita.

`--exclude, -x`

Il nome delle risorse da escludere da. `sam build` Ad esempio, se il modello contiene `Function1Function2`, e `Function3` e lo eseguisam `build --exclude Function2`, solo `Function1` e `Function3` verrà creato.

`--help`

Mostra questo messaggio ed esce.

`--hook-name` *TEXT*

Il nome dell'hook utilizzato per estendere le AWS SAMCLI funzionalità.

Valori accettati:`terraform`.

`--manifest , -m` *PATH*

Il percorso di un file manifesto delle dipendenze personalizzato (ad esempio, package.json) da utilizzare al posto di quello predefinito.

`--mount-symlinks`

Assicura che monti AWS SAMCLI sempre i collegamenti simbolici presenti nei file da creare o richiamare. Questo vale solo per i collegamenti simbolici nella directory di primo livello (ovvero i collegamenti simbolici che si trovano direttamente nella radice della funzione). Per impostazione predefinita, i collegamenti simbolici non vengono montati ad eccezione di quelli necessari per essere utilizzati `build-in-source` `node_modules` in NodeJS.

`--no-use-container`

Un'opzione che consente di utilizzare il toolkit IDE per impostare il comportamento predefinito. Puoi anche usarlo `sam build --no-use-container` per eseguire una build nella tua macchina locale anziché in un contenitore docker.

`--parallel`

Abilita le build parallele. Usa questa opzione per creare le funzioni e i livelli del tuo AWS SAM modello in parallelo. Per impostazione predefinita, le funzioni e i livelli vengono creati in sequenza.

`--parameter-overrides`

(Facoltativo) Una stringa che contiene le sostituzioni CloudFormation dei parametri codificate come coppie chiave-valore. Utilizza lo stesso formato di (). AWS Command Line Interface AWS CLI Ad esempio: '`ParameterKey=KeyPairName`, `ParameterValue` `MyKey` `ParameterKey` `=InstanceType`, `ParameterValue` `=t1.micro`'. Questa opzione non è compatibile con`--hook-name`.

`--profile` *TEXT*

Il profilo specifico del file delle credenziali che ottiene le AWS credenziali.

`--region` *TEXT*

Il Regione AWS verso cui eseguire la distribuzione. Ad esempio, us-east-1.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file di AWS SAM configurazione.

`--skip-prepare-infra`

Salta la fase di preparazione se non sono state apportate modifiche all'infrastruttura. Da utilizzare con l'`--hook-name`opzione.

`--skip-pull-image`

Specifica se il comando deve ignorare l'estrazione dell'ultima immagine Docker per il runtime Lambda.

`--template-file, --template, -t` *PATH*

Il percorso e il nome del file modello. AWS SAM `[default: template.[yaml|yml]]` Questa opzione non è compatibile con`--hook-name`.

`--terraform-project-root-path`

Il percorso relativo o assoluto della directory di primo livello contenente i file di Terraform configurazione o il codice sorgente della funzione. Se questi file si trovano all'esterno della directory contenente il modulo Terraform root, utilizzate questa opzione per specificarne il percorso assoluto o relativo. Questa opzione richiede che `--hook-name` sia impostata `suterraform`.

`--use-container`, `-u`

Se le tue funzioni dipendono da pacchetti che hanno dipendenze compilate in modo nativo, usa questa opzione per creare la tua funzione all'interno di un contenitore Docker simile a Lambda.

Esempio

Per un esempio dettagliato e una procedura dettagliata sull'utilizzo del `sam build` sottocomando, fare riferimento a. Introduzione alla costruzione con AWS SAM

sam delete

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam delete`.

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam delete` comando elimina un' AWS SAM applicazione eliminando lo CloudFormation stack, gli artefatti che sono stati impacchettati e distribuiti su Amazon S3 e Amazon ECR e il file modello. AWS SAM

Questo comando verifica anche se è stato distribuito uno stack complementare Amazon ECR e, in tal caso, richiede all'utente di eliminare quello stack e i repository Amazon ECR. Se `--no-prompts` specificato, gli stack complementari e i repository Amazon ECR vengono eliminati per impostazione predefinita.

Utilizzo

```
$ sam delete <options>
```

Opzioni

`--config-env` *TEXT*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è `default`. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *PATH*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito si trova `samconfig.toml` nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--debug`

Attiva la registrazione di debug per stampare il messaggio di debug AWS SAMCLI generato e per visualizzare i timestamp.

`--help`

Mostra questo messaggio ed esce.

`--no-prompts`

Specificate questa opzione per farla AWS SAM funzionare in modalità non interattiva. Il nome dello stack deve essere fornito, con l'`--stack-name`opzione o nel file di configurazione`toml`.

`--profile` *TEXT*

Il profilo specifico del file di credenziali che ottiene AWS le credenziali.

`--region` *TEXT*

La AWS regione in cui effettuare la distribuzione. Ad esempio, us-east-1.

`--s3-bucket`

Il percorso del bucket Amazon S3 che desideri eliminare.

`--s3-prefix`

Il prefisso del bucket Amazon S3 che desideri eliminare.

`--save-params`

Salva i parametri che fornisci nella riga di comando nel file di configurazione. AWS SAM

`--stack-name` *`TEXT`*

Il nome dello CloudFormation stack che desideri eliminare.

Esempi

Il comando seguente elimina lo stack. `MY-STACK`

```
$ sam delete --stack-name MY-STACK
```

Il comando seguente elimina lo stack `MY-STACK` e il bucket S3: `sam-s3-demo-bucket`

```
$ sam delete \
    --stack-name MyStack \
    --s3-bucket MySAMBucket
```

sam deploy

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam deploy`.

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'uso del AWS SAMCLI `sam deploy` comando, vedereIntroduzione alla distribuzione con AWS SAM.

Il `sam deploy` comando distribuisce un'applicazione all'utente Cloud AWS . AWS CloudFormation

Utilizzo

```
$ <environment variables> sam deploy <options>
```

Variabili di ambiente

`SAM_CLI_POLL_DELAY`

Imposta la variabile di `SAM_CLI_POLL_DELAY` ambiente con un valore di secondi nella shell per configurare la frequenza con cui la CLI AWS SAM controlla lo stato dello CloudFormation stack, il che è utile quando si verifica la limitazione da. CloudFormation Questa variabile env viene utilizzata per il polling delle chiamate `describe_stack` API, che vengono effettuate durante l'esecuzione. `sam deploy`

Di seguito è riportato un esempio di questa variabile:

```
$ SAM_CLI_POLL_DELAY=5 sam deploy
```

Opzioni

`--capabilities` *`LIST`*

Un elenco di funzionalità che è necessario specificare per consentire la creazione CloudFormation di determinati stack. Alcuni modelli di stack potrebbero includere risorse che influiscono sulle autorizzazioni dell'utente Account AWS, ad esempio creando nuovi utenti AWS Identity and Access Management (IAM). Per questi stack, è necessario riconoscerne esplicitamente le funzionalità specificando questa opzione. Gli unici valori validi sono `CAPABILITY_IAM` e `CAPABILITY_NAMED_IAM`. Se disponi di risorse IAM, puoi specificare entrambe le funzionalità. Se disponi di risorse IAM con nomi personalizzati, devi specificare`CAPABILITY_NAMED_IAM`. Se non specifichi questa opzione, l'operazione restituisce un `InsufficientCapabilities` errore.

Quando si distribuisce un'applicazione che contiene applicazioni nidificate, è necessario utilizzare `CAPABILITY_AUTO_EXPAND` per confermare che l'applicazione contiene applicazioni nidificate. Per ulteriori informazioni, consulta Distribuzione di applicazioni annidate.

`--config-env` *`TEXT`*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è `default`. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *`PATH`*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito si trova `samconfig.toml` nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--confirm-changeset | --no-confirm-changeset`

Richiede di confermare se AWS SAMCLI distribuisce il changeset calcolato.

`--debug`

Attiva la registrazione di debug per stampare il messaggio di debug generato e per visualizzare i timestamp. AWS SAMCLI

`--disable-rollback | --no-disable-rollback`

Specificate se ripristinare lo CloudFormation stack se si verifica un errore durante una distribuzione. Per impostazione predefinita, se si verifica un errore durante una distribuzione, lo CloudFormation stack torna all'ultimo stato stabile. Se si specifica `--disable-rollback` e si verifica un errore durante una distribuzione, non viene eseguito il rollback delle risorse create o aggiornate prima che si verificasse l'errore.

`--fail-on-empty-changeset | --no-fail-on-empty-changeset`

Specificate se restituire un codice di uscita diverso da zero se non ci sono modifiche da apportare allo stack. Il comportamento predefinito consiste nel restituire un codice di uscita diverso da zero.

`--force-upload`

Specificate questa opzione per caricare gli artefatti anche se corrispondono agli artefatti esistenti nel bucket Amazon S3. Gli artefatti corrispondenti vengono sovrascritti.

`--guided, -g`

Specificate questa opzione per far sì che i prompt di AWS SAMCLI utilizzo vi guidino nella distribuzione.

`--help`

Mostra questo messaggio ed esci.

`--image-repositories` *`TEXT`*

Una mappatura delle funzioni sull'URI del loro repository Amazon ECR. Funzioni di riferimento in base al relativo ID logico. Di seguito è riportato un esempio:

```
$ sam deploy --image-repositories Function1=123456789012.dkr.ecr.us-
east-1.amazonaws.com/my-repo
```

È possibile specificare questa opzione più volte in un unico comando.

`--image-repository` *TEXT*

Il nome dell'archivio Amazon ECR in cui questo comando carica l'immagine della tua funzione. Questa opzione è necessaria per le funzioni dichiarate con il `Image` tipo di pacchetto.

`--kms-key-id` *TEXT*

L'ID di una chiave AWS Key Management Service (AWS KMS) utilizzata per crittografare gli artefatti che sono a riposo nel bucket Amazon S3. Se non specifichi questa opzione, AWS SAM utilizza le chiavi di crittografia gestite da Amazon S3.

`--metadata`

Una mappa di metadati da allegare a tutti gli artefatti a cui si fa riferimento nel modello.

`--no-execute-changeset`

Indica se applicare il changeset. Specificate questa opzione se desiderate visualizzare le modifiche apportate allo stack prima di applicare il changeset. Questo comando crea un CloudFormation changeset e quindi esce senza applicare il changeset. Per applicare il changeset, esegui lo stesso comando senza questa opzione.

`--no-progressbar`

Non visualizzare una barra di avanzamento durante il caricamento di artefatti su Amazon S3.

`--notification-arns` *LIST*

Un elenco di ARNs argomenti di Amazon Simple Notification Service (Amazon SNS) associati allo CloudFormation stack.

`--on-failure [ROLLBACK | DELETE | DO_NOTHING]`

Specificate l'azione da intraprendere quando la creazione di uno stack non riesce.

Sono disponibili le seguenti opzioni:

- `ROLLBACK`— Riporta lo stack a uno stato precedente noto come buono.
- `DELETE`— Riporta la pila a uno stato precedente noto come buono, se ne esiste uno. Altrimenti, elimina lo stack.

- `DO_NOTHING`— Non ripristina né elimina lo stack. L'effetto è lo stesso di. `--disable-rollback`

Il comportamento predefinito è `ROLLBACK`.

> **Note**
>
> È possibile specificare l'`--disable-rollback`opzione o l'`--on-failure`opzione, ma non entrambe.

`--parameter-overrides` *`LIST`*

Una stringa che contiene le sostituzioni CloudFormation dei parametri codificate come coppie chiave-valore. Ogni override utilizza il formato. `ParameterKey=name,ParameterValue=value` Le sostituzioni multiple sono separate da spazi. Di seguito, sono riportati due esempi:

```
$ sam deploy --parameter-overrides ParameterKey=value1,ParameterValue=value2
```

```
$ sam deploy --parameter-overrides ParameterKey=value1,ParameterValue=value2
 ParameterKey=hello,ParameterValue=world ParameterKey=apple,ParameterValue=banana
```

`--profile` *`TEXT`*

Il profilo specifico del file delle credenziali che ottiene le credenziali. AWS

`--region` *`TEXT`*

Il Regione AWS verso cui eseguire la distribuzione. Ad esempio, us-east-1.

`--resolve-image-repos`

Crea automaticamente repository Amazon ECR da utilizzare per il packaging e la distribuzione per distribuzioni non guidate. Questa opzione si applica solo alle funzioni e ai livelli specificati. `PackageType: Image` Se si specifica l'`--guided`opzione, la AWS SAMCLI ignora. `--resolve-image-repos`

> **Note**
>
> Se crea AWS SAM automaticamente un archivio Amazon ECR per funzioni o livelli con questa opzione e successivamente elimini tali funzioni o livelli dal AWS SAM modello, i repository Amazon ECR corrispondenti vengono eliminati automaticamente.

`--resolve-s3`

Crea automaticamente un bucket Amazon S3 da utilizzare per il packaging e la distribuzione per distribuzioni non guidate. Se si specifica l'`--guided`opzione, la AWS SAM CLI la ignora. `--resolve-s3` Se si specificano entrambe le `--resolve-s3` opzioni `--s3-bucket` e, si verifica un errore.

`--role-arn` *`TEXT`*

L'Amazon Resource Name (ARN) di un ruolo IAM che CloudFormation assume quando si applica il changeset.

`--s3-bucket` *`TEXT`*

Il nome del bucket Amazon S3 in cui questo comando carica il modello. CloudFormation Se il modello è più grande di 51.200 byte, è necessaria l'opzione o l'`--s3-bucket`opzione. `--resolve-s3` Se si specificano entrambe le `--resolve-s3` opzioni `--s3-bucket` e, si verifica un errore.

`--s3-prefix` *`TEXT`*

Il prefisso aggiunto ai nomi degli artefatti caricati nel bucket Amazon S3. Il nome del prefisso è un nome di percorso (nome della cartella) per il bucket Amazon S3.

`--save-params`

Salva i parametri che fornisci nella riga di comando nel file di configurazione. AWS SAM

`--signing-profiles` *`LIST`*

L'elenco dei profili di firma con cui firmare i pacchetti di distribuzione. Questa opzione richiede un elenco di coppie chiave-valore, in cui la chiave è il nome della funzione o del livello da firmare e il valore è il profilo di firma, con un proprietario del profilo opzionale delimitato da. `:` Ad esempio, `FunctionNameToSign=SigningProfileName1 LayerNameToSign=SigningProfileName2:SigningProfileOwner`.

`--stack-name TEXT`

(Obbligatorio) Il nome dello CloudFormation stack in cui stai eseguendo la distribuzione. Se specificate uno stack esistente, il comando aggiorna lo stack. Se specificate un nuovo stack, il comando lo crea.

`--tags LIST`

Un elenco di tag da associare allo stack creato o aggiornato. CloudFormation propaga inoltre questi tag alle risorse dello stack che li supportano.

`--template-file, --template, -t PATH`

Il percorso e il nome del file in cui si trova il AWS SAM modello.

> **Note**
>
> Se si specifica questa opzione, AWS SAM distribuisce solo il modello e le risorse locali a cui punta.

`--use-json`

Esporta JSON per il modello. CloudFormation L'output predefinito è YAML.

Esempio

Per un esempio dettagliato e una procedura dettagliata sull'utilizzo del `sam deploy` sottocomando, fare riferimento a. Introduzione alla distribuzione con AWS SAM

sam init

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI). `sam init`

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'uso del AWS SAMCLI `sam init` comando, vedereCrea la tua candidatura in AWS SAM.

Il `sam init` comando fornisce opzioni per inizializzare una nuova applicazione serverless.

Utilizzo

```
$ sam init <options>
```

Opzioni

`--app-template` *TEXT*

L'identificatore del modello di applicazione gestita che si desidera utilizzare. Se non sei sicuro, chiama `sam init` senza opzioni per un flusso di lavoro interattivo.

Questo parametro è obbligatorio se `--no-interactive` è specificato e non `--location` viene fornito.

Questo parametro è disponibile solo nella AWS SAMCLI versione 0.30.0 e successive. La specificazione di questo parametro con una versione precedente genera un errore.

`--application-insights | --no-application-insights`

Attiva il monitoraggio di Amazon CloudWatch Application Insights per la tua applicazione. Per ulteriori informazioni, consulta Utilizzo di CloudWatch Application Insights per monitorare le applicazioni AWS SAM serverless.

L'opzione predefinita è `--no-application-insights`.

`--architecture, -a` *[x86_64 | arm64]*

L'architettura del set di istruzioni per le funzioni Lambda dell'applicazione. Specificate uno dei `x86_64` o`arm64`.

`--base-image` *[amazon/dotnet8-base | amazon/dotnet6-base | amazon/java25-base | amazon/java21-base | amazon/java17-base | amazon/java11-base | amazon/nodejs24.x-base | amazon/nodejs22.x-base | amazon/nodejs20.x-base | amazon/nodejs18.x-base | amazon/nodejs16.x-base | amazon/python3.14-base | amazon/python3.13-base | amazon/python3.12-base | amazon/python3.11-base | amazon/python3.10-base | amazon/python3.9-base | amazon/python3.8-base | amazon/ruby3.4-base | amazon/ruby3.3-base | amazon/ruby3.2-base]*

L'immagine di base dell'applicazione. Questa opzione si applica solo quando il tipo di pacchetto è`Image`.

Questo parametro è obbligatorio se `--no-interactive` è specificato, `--package-type` è specificato come `Image` e non `--location` è specificato.

`--config-env` *TEXT*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *PATH*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è «samconfig.toml» nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati e per visualizzare i AWS SAMCLI timestamp.

`--dependency-manager, -d` *[gradle | mod | maven | bundler | npm | cli-package | pip]*

Il gestore delle dipendenze del tuo runtime Lambda.

`--extra-content`

Sovrascrivi qualsiasi parametro personalizzato nella `cookiecutter.json` configurazione del modello, ad esempio. `{"customParam1": "customValue1", "customParam2":"customValue2"}`

`--help, -h`

Mostra questo messaggio ed esce.

`--location, -l` *TEXT*

La posizione del modello o dell'applicazione (Git, Mercurial, HTTP/HTTPS, file.zip, percorso).

Questo parametro è obbligatorio se `--no-interactive` è specificato e `--runtime--name`, e `--app-template` non vengono forniti.

Per i repository Git, è necessario utilizzare la posizione della radice del repository.

Per i percorsi locali, il modello deve essere in formato.zip o Cookiecutter.

`--name, -n` *TEXT*

Il nome del progetto da generare come directory.

Questo parametro è obbligatorio se `--no-interactive` è specificato e non `--location` viene fornito.

`--no-input`

Disattiva la richiesta di Cookiecutter e accetta i valori vcfdefault definiti nella configurazione del modello.

`--no-interactive`

Disabilita la richiesta interattiva per i parametri init e fallisce se mancano i valori richiesti.

`--output-dir, -o` *`PATH`*

La posizione in cui viene emessa l'applicazione inizializzata.

`--package-type` *`[ Zip | Image ]`*

Il tipo di pacchetto dell'applicazione di esempio. `Zip`crea un archivio di file.zip e `Image` crea un'immagine del contenitore.

`--runtime, -r` *`[ dotnet8 | dotnet6 | java25 | java21 | java17 | java11 | nodejs24.x | nodejs22.x | nodejs20.x | nodejs18.x | nodejs16.x | python3.14 | python3.13 | python3.12 | python3.11 | python3.10 | python3.9 | python3.8 | ruby3.4 | ruby3.3 | ruby3.2 ]`*

Il runtime Lambda della tua applicazione. Questa opzione si applica solo quando il tipo di pacchetto è`Zip`.

Questo parametro è obbligatorio se `--no-interactive` è specificato, `--package-type` è specificato come `Zip` e non `--location` è specificato.

`--save-params`

Salva i parametri forniti nella riga di comando nel file AWS SAM di configurazione.

`--tracing | --no-tracing`

Attiva il AWS X-Ray tracciamento per le tue funzioni Lambda.

Esempio

Per un esempio dettagliato e una procedura dettagliata sull'utilizzo del `sam init` sottocomando, fare riferimento a. Crea la tua candidatura in AWS SAM

sam list

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam list`.

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam list` comando fornisce informazioni importanti sulle risorse dell'applicazione serverless e sullo stato dell'applicazione serverless. Utilizzalo sam list prima e dopo l'implementazione per fornire assistenza durante lo sviluppo locale e nel cloud.

Utilizzo

```
$ sam list <options> <subcommand>
```

Opzioni

`--help, -h`

Mostra questo messaggio ed esci.

Sottocomandi

`endpoints`

Visualizza un elenco di endpoint cloud e locali dal tuo CloudFormation stack. Per ulteriori informazioni, consulta sam list endpoints.

`resources`

Visualizza le risorse del modello AWS Serverless Application Model (AWS SAM) create AWS CloudFormation al momento della distribuzione. Per ulteriori informazioni, consulta sam list resources.

`stack-outputs`

Visualizza gli output dello CloudFormation stack da un modello AWS SAM or CloudFormation . Per ulteriori informazioni, consulta sam list stack-outputs.

sam list endpoints

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model `sam list endpoints` sottocomando Command Line Interface (AWS SAMCLI).

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam list endpoints` sottocomando visualizza un elenco di endpoint cloud e locali dallo stack. CloudFormation È possibile interagire con queste risorse tramite i sam local comandi and. sam sync

AWS Lambda e i tipi di risorse Amazon API Gateway sono supportati con questo comando.

> **Note**
>
> I domini personalizzati sono supportati se configurati per le tue risorse Amazon API Gateway. Questo comando genererà il dominio personalizzato anziché l'endpoint predefinito.

Utilizzo

```
$ sam list endpoints <options>
```

Opzioni

`--config-env` *`TEXT`*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare.

Valore predefinito: `default`

Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *`TEXT`*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare.

Valore predefinito: `samconfig.toml` nella directory di lavoro corrente.

Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dai timestamp with. AWS SAMCLI

`--help, -h`

Mostra questo messaggio ed esci.

`--output [json|table]`

Specificate il formato per l'output dei risultati.

Valore predefinito: `table`

`--profile` *TEXT*

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--region` *TEXT*

Imposta la AWS regione del servizio. Ad esempio, `us-east-1`.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file AWS SAM di configurazione.

`--stack-name` *TEXT*

Nome dello CloudFormation stack distribuito. Il nome dello stack si trova nel `samconfig.toml` file dell'applicazione o nel file di configurazione designato.

Quando questa opzione non è specificata, verranno visualizzate le risorse locali definite nel modello.

`--template-file, --template, -t` *PATH*

AWS SAM file modello.

Valore predefinito: `template.[yaml|yml|json]`

Esempi

Visualizza un output, in formato json, degli endpoint di risorse distribuiti dallo stack denominato. CloudFormation `test-stack`

```
$ sam list endpoints --stack-name test-stack --output json
```

```
[
  {
    "LogicalResourceId": "HelloWorldFunction",
    "PhysicalResourceId": "sam-app-test-list-HelloWorldFunction-H85Y7yIV7ZLq",
    "CloudEndpoint": "https://zt55oi7kbljxjmcoahsj3cknwu0rposq.lambda-url.us-
east-1.on.aws/",
    "Methods": "-"
  },
  {
    "LogicalResourceId": "ServerlessRestApi",
    "PhysicalResourceId": "uj80uoe2o2",
    "CloudEndpoint": [
      "https://uj80uoe2o2.execute-api.us-east-1.amazonaws.com/Prod",
      "https://uj80uoe2o2.execute-api.us-east-1.amazonaws.com/Stage"
    ],
    "Methods": [
      "/hello['get']"
    ]
  }
]
```

sam list resources

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model `sam list resources` sottocomando Command Line Interface (AWS SAMCLI).

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam list resources` sottocomando visualizza le risorse del modello AWS Serverless Application Model (AWS SAM) create AWS CloudFormation dalla AWS SAM trasformazione al momento della distribuzione.

sam list resourcesUtilizzalo con un AWS SAM modello prima della distribuzione per vedere le risorse che verranno create. Fornisci un nome di CloudFormation stack per visualizzare un elenco consolidato che include le risorse distribuite.

> **Note**
>
> Per generare un elenco di risorse dal AWS SAM modello, viene eseguita una trasformazione locale del modello. Le risorse che verranno distribuite con determinate condizioni, ad esempio all'interno di una regione specifica, sono incluse in questo elenco.

Utilizzo

```
$ sam list resources <options>
```

Opzioni

`--config-env` *TEXT*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare.

Valore predefinito: `default`

Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *TEXT*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare.

Valore predefinito: `samconfig.toml` nella directory di lavoro corrente.

Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dai timestamp with. AWS SAMCLI

`--help, -h`

Mostra questo messaggio ed esci.

`--output [json|table]`

Specificate il formato per l'output dei risultati.

Valore predefinito: `table`

`--profile` *TEXT*

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--region` *TEXT*

Imposta la AWS regione del servizio. Ad esempio, `us-east-1`.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file AWS SAM di configurazione.

`--stack-name` *TEXT*

Nome dello CloudFormation stack distribuito. Il nome dello stack si trova nel `samconfig.toml` file dell'applicazione o nel file di configurazione designato.

Una volta fornite, le risorse logiche IDs del modello verranno mappate all'ingresso fisico IDs corrispondente. CloudFormation Per ulteriori informazioni sulle risorse fisiche IDs, consulta i campi delle risorse nella Guida per l'AWS CloudFormation utente.

Quando questa opzione non è specificata, verranno visualizzate le risorse locali definite nel modello.

`--template-file, --template, -t` *PATH*

AWS SAM file modello.

Valore predefinito: `template.[yaml|yml|json]`

Esempi

Visualizza un output, in formato tabella, delle risorse locali del tuo AWS SAM modello e delle risorse distribuite dallo CloudFormation stack denominato. `test-stack` Esegui dalla stessa directory del modello locale.

```
$ sam list resources --stack-name test-stack --output table

-------------------------------------------------------------------------------------------------------
Logical ID                                                      Physical ID
-------------------------------------------------------------------------------------------------------
HelloWorldFunction                                              sam-app-test-list-
HelloWorldFunction-H85Y7yIV7ZLq
HelloWorldFunctionHelloWorldPermissionProd                      sam-app-test-list-

 HelloWorldFunctionHelloWorldPermissionProd-1QH7CPOCBL2IK
HelloWorldFunctionRole                                          sam-app-test-list-
HelloWorldFunctionRole-SRJDMJ6F7F41
ServerlessRestApi                                               uj80uoe2o2
ServerlessRestApiDeployment47fc2d5f9d                           pncw5f
ServerlessRestApiProdStage                                      Prod
ServerlessRestApiDeploymentf5716dc08b                           -
```

```
----------------------------------------------------------------------------------------------------
```

sam list stack-outputs

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model `sam list stack-outputs` sottocomando Command Line Interface (AWS SAMCLI).

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam list stack-outputs` sottocomando visualizza gli output dello AWS CloudFormation stack da un AWS Serverless Application Model (AWS SAM) o da un modello. CloudFormation Per ulteriori informazioni*`Outputs`*, consultate Outputs nella User Guide.AWS CloudFormation

Utilizzo

```
$ sam list stack-outputs <options>
```

Opzioni

`--config-env` *`TEXT`*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare.

Valore predefinito: `default`

Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *`TEXT`*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare.

Valore predefinito: `samconfig.toml` nella directory di lavoro corrente.

Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dai timestamp with. AWS SAMCLI

`--help, -h`

Mostra questo messaggio ed esci.

`--output [json|table]`

Specificate il formato per l'output dei risultati.

Valore predefinito: `table`

`--profile` *TEXT*

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--region` *TEXT*

Imposta la AWS regione del servizio. Ad esempio, `us-east-1`.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file AWS SAM di configurazione.

`--stack-name` *TEXT*

Nome dello CloudFormation stack distribuito. Il nome dello stack si trova nel `samconfig.toml` file dell'applicazione o nel file di configurazione designato.

Questa opzione è obbligatoria.

Esempi

Visualizza gli output, in formato tabellare, delle risorse presenti nello CloudFormation stack denominato. `test-stack`

```
$ sam list stack-outputs --stack-name test-stack --output table

-----------------------------------------------------------------------------------------------------
OutputKey                                 OutputValue
 Description
-----------------------------------------------------------------------------------------------------
HelloWorldFunctionIamRole                 arn:aws:iam::account-number:role/sam-
 Implicit IAM Role created for Hello
                                          app-test-list-HelloWorldFunctionRole-    World
 function
                                          SRJDMJ6F7F41
HelloWorldApi                             https://uj80uoe2o2.execute-api.us-       API
 Gateway endpoint URL for Prod
                                          east-1.amazonaws.com/Prod/hello/         stage
 for Hello World function
```

```
HelloWorldFunction                      arn:aws:lambda:us-                      Hello
 World Lambda Function ARN
                                        east-1:account-number:function:sam-app-
                                        test-list-
                                        HelloWorldFunction-H85Y7yIV7ZLq
-----------------------------------------------------------------------------------------------
```

sam local callback

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam local callback`.

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam local callback` comando consente di inviare callback a esecuzioni di funzioni durevoli locali durante lo sviluppo e il test.

Utilizzo

```
$ sam local callback <subcommand> <options>
```

Opzioni

`--help, -h`

Mostra questo messaggio ed esci.

Sottocomandi

`succeed`

Invia una callback riuscita a un'esecuzione duratura della funzione. Per ulteriori informazioni, consulta sam local callback succeed.

`fail`

Invia un callback di errore a un'esecuzione duratura della funzione. Per ulteriori informazioni, consulta sam local callback fail.

`heartbeat`

Invia un callback del battito cardiaco a un'esecuzione duratura della funzione. Per ulteriori informazioni, consulta sam local callback heartbeat.

sam local callback succeed

Invia una callback riuscita a un'esecuzione duratura della funzione.

Utilizzo

```
$ sam local callback succeed CALLBACK_ID [OPTIONS]
```

Argomenti richiesti

`CALLBACK_ID`

L'identificatore univoco del callback a cui inviare una risposta riuscita.

Opzioni

`-r, --result TEXT`

Risultato di successo: payload come stringa.

`--region TEXT`

Imposta la AWS regione del servizio (ad esempio,`us-east-1`).

`--profile TEXT`

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--config-env TEXT`

Nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione. Impostazione predefinita: `default`

`--config-file TEXT`

File di configurazione contenente i valori dei parametri predefiniti. Impostazione predefinita: `samconfig.toml`

`--save-params`

Salva i parametri forniti tramite la riga di comando nel file di configurazione.

`--beta-features / --no-beta-features`

Abilita/disabilita le funzionalità beta.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dalla CLI e visualizzare i timestamp AWS SAM .

`-h, --help`

Mostra questo messaggio ed esce.

Esempi

Invia una richiamata riuscita senza risultati:

```
$ sam local callback succeed my-callback-id
```

Invia una richiamata riuscita con risultato:

```
$ sam local callback succeed my-callback-id --result 'Task completed successfully'
```

Invia una richiamata riuscita con l'opzione breve:

```
$ sam local callback succeed my-callback-id -r 'Success result'
```

sam local callback fail

Invia un callback di errore a un'esecuzione duratura della funzione.

Utilizzo

```
$ sam local callback fail CALLBACK_ID [OPTIONS]
```

Argomenti richiesti

`CALLBACK_ID`

L'identificatore univoco del callback a cui inviare una risposta di errore.

Opzioni

`--error-data TEXT`

Dati di errore aggiuntivi.

`--stack-trace TEXT`

Immetti le voci di tracciamento dello stack. Questa opzione può essere specificata più volte.

`--error-type TEXT`

Tipo di errore.

`--error-message TEXT`

Messaggio di errore dettagliato.

`--region TEXT`

Imposta la AWS regione del servizio (ad esempio,`us-east-1`).

`--profile TEXT`

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--config-env TEXT`

Nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione. Impostazione predefinita: `default`

`--config-file TEXT`

File di configurazione contenente i valori dei parametri predefiniti. Impostazione predefinita: `samconfig.toml`

`--save-params`

Salva i parametri forniti tramite la riga di comando nel file di configurazione.

`--beta-features / --no-beta-features`

Abilita/disabilita le funzionalità beta.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dalla CLI e visualizzare i timestamp AWS SAM .

`-h, --help`

Mostra questo messaggio ed esce.

Esempi

Invia un callback di errore senza parametri:

```
$ sam local callback fail my-callback-id
```

Invia una richiamata di errore con messaggio di errore:

```
$ sam local callback fail my-callback-id --error-message 'Task failed'
```

Invia una richiamata di errore con tutti i parametri:

```
$ sam local callback fail my-callback-id --error-message 'Task failed' --error-type
 'ValidationError' --stack-trace 'at line 42' --error-data '{"code": 500}'
```

sam local callback heartbeat

Invia una richiamata del battito cardiaco a un'esecuzione duratura della funzione.

Utilizzo

```
$ sam local callback heartbeat CALLBACK_ID [OPTIONS]
```

Argomenti richiesti

`CALLBACK_ID`

L'identificatore univoco del callback a cui inviare un heartbeat.

Opzioni

`--region TEXT`

Imposta la AWS regione del servizio (ad esempio,`us-east-1`).

`--profile TEXT`

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--config-env TEXT`

Nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione. Impostazione predefinita: `default`

`--config-file TEXT`

File di configurazione contenente i valori dei parametri predefiniti. Impostazione predefinita: `samconfig.toml`

`--save-params`

Salva i parametri forniti tramite la riga di comando nel file di configurazione.

`--beta-features / --no-beta-features`

Abilita/disabilita le funzionalità beta.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dalla CLI e visualizzare i timestamp AWS SAM .

`-h, --help`

Mostra questo messaggio ed esce.

Esempi

Invia una richiamata del battito cardiaco:

```
$ sam local callback heartbeat my-callback-id
```

sam local execution

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam local execution`.

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam local execution` comando consente di gestire e controllare le esecuzioni locali di funzioni durevoli durante lo sviluppo e il test.

> **Note**
>
> Questi comandi potrebbero non richiedere l'accesso alle credenziali. AWS

Utilizzo

```
$ sam local execution <subcommand> <options>
```

Opzioni

`--help, -h`

Mostra questo messaggio ed esci.

Sottocomandi

`get`

Ottieni i dettagli di un'esecuzione duratura della funzione. Per ulteriori informazioni, consulta sam local execution get.

`history`

Ottieni la cronologia di esecuzione di un'esecuzione duratura di una funzione. Per ulteriori informazioni, consulta sam local execution history.

`stop`

Interrompe l'esecuzione duratura di una funzione. Per ulteriori informazioni, consulta sam local execution stop.

sam local execution get

Recupera i dettagli dell'esecuzione durevole di una funzione specifica.

> **Note**
>
> Questo comando potrebbe non richiedere l'accesso alle AWS credenziali.

Utilizzo

```
$ sam local execution get DURABLE_EXECUTION_ARN [OPTIONS]
```

Argomenti richiesti

`DURABLE_EXECUTION_ARN`

L'Amazon Resource Name (ARN) dell'esecuzione durevole della funzione di cui recuperare i dettagli.

Opzioni

`--format [summary|json]`

Formato di output. Impostazione predefinita: `summary`

`--region TEXT`

Imposta la AWS regione del servizio (ad esempio,`us-east-1`).

`--profile TEXT`

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--config-env TEXT`

Nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione. Impostazione predefinita: `default`

`--config-file TEXT`

File di configurazione contenente i valori dei parametri predefiniti. Impostazione predefinita: `samconfig.toml`

`--save-params`

Salva i parametri forniti tramite la riga di comando nel file di configurazione.

`--beta-features / --no-beta-features`

Abilita/disabilita le funzionalità beta.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dalla CLI e visualizzare i timestamp AWS SAM .

`-h, --help`

Mostra questo messaggio ed esce.

Esempi

Ottieni i dettagli dell'esecuzione con il formato di riepilogo predefinito:

```
$ sam local execution get arn:aws:lambda:us-east-1:123456789012:function:MyFunction:
$LATEST/durable-execution/my-execution-name/my-execution-id
```

Ottieni i dettagli di esecuzione in formato JSON:

```
$ sam local execution get arn:aws:lambda:us-east-1:123456789012:function:MyFunction:
$LATEST/durable-execution/my-execution-name/my-execution-id --format json
```

sam local execution history

Ottieni la cronologia di esecuzione di un'esecuzione duratura di una funzione.

Utilizzo

```
$ sam local execution history EXECUTION_ID [OPTIONS]
```

Argomenti richiesti

`EXECUTION_ID`

L'identificatore univoco dell'esecuzione durevole della funzione di cui recuperare la cronologia.

Opzioni

`--format [table|json]`

Formato di output. Impostazione predefinita: `table`

`--region TEXT`

Imposta la AWS regione del servizio (ad esempio,`us-east-1`).

`--profile TEXT`

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--config-env TEXT`

Nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione. Impostazione predefinita: `default`

`--config-file TEXT`

File di configurazione contenente i valori dei parametri predefiniti. Impostazione predefinita: `samconfig.toml`

`--save-params`

Salva i parametri forniti tramite la riga di comando nel file di configurazione.

`--beta-features / --no-beta-features`

Abilita/disabilita le funzionalità beta.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dalla CLI e visualizzare i timestamp AWS SAM .

`-h, --help`

Mostra questo messaggio ed esce.

Esempi

Ottieni la cronologia di esecuzione in formato tabella:

```
$ sam local execution history my-execution-id
```

Ottieni la cronologia di esecuzione in formato JSON:

```
$ sam local execution history my-execution-id --format json
```

sam local execution stop

Interrompe l'esecuzione durevole di una funzione in esecuzione.

> **Note**
>
> Questo comando potrebbe non richiedere l'accesso alle AWS credenziali.

Utilizzo

```
$ sam local execution stop DURABLE_EXECUTION_ARN [OPTIONS]
```

Argomenti richiesti

`DURABLE_EXECUTION_ARN`

L'Amazon Resource Name (ARN) dell'esecuzione durevole della funzione da interrompere.

Opzioni

`--error-message TEXT`

Messaggio di errore da associare all'esecuzione interrotta.

`--error-type TEXT`

Tipo di errore da associare all'esecuzione interrotta.

`--error-data TEXT`

Dati di errore da associare all'esecuzione interrotta.

`--stack-trace TEXT`

Immissioni di stack trace. Questa opzione può essere specificata più volte.

`--region TEXT`

Imposta la AWS regione del servizio (ad esempio,`us-east-1`).

`--profile TEXT`

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--config-env TEXT`

Nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione. Impostazione predefinita: `default`

`--config-file TEXT`

File di configurazione contenente i valori dei parametri predefiniti. Impostazione predefinita: `samconfig.toml`

`--save-params`

Salva i parametri forniti tramite la riga di comando nel file di configurazione.

`--beta-features / --no-beta-features`

Abilita/disabilita le funzionalità beta.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dalla CLI e visualizzare i timestamp AWS SAM .

`-h, --help`

Mostra questo messaggio ed esce.

Esempi

Interrompi l'esecuzione senza dettagli sull'errore:

```
$ sam local execution stop arn:aws:lambda:us-east-1:123456789012:function:MyFunction:
$LATEST/durable-execution/my-execution-name/my-execution-id
```

Interrompi l'esecuzione con un messaggio di errore e digita:

```
$ sam local execution stop arn:aws:lambda:us-east-1:123456789012:function:MyFunction:
$LATEST/durable-execution/my-execution-name/my-execution-id --error-message "Execution
 cancelled" --error-type "UserCancellation"
```

Interrompi l'esecuzione con i dettagli completi dell'errore e la traccia dello stack:

```
$ sam local execution stop arn:aws:lambda:us-east-1:123456789012:function:MyFunction:
$LATEST/durable-execution/my-execution-name/my-execution-id --error-message "Task
 failed" --error-type "TaskFailure" --error-data '{"reason":"timeout"}' --stack-trace
 "at function1()" --stack-trace "at function2()"
```

sam local generate-event

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model sottocomando Command Line Interface ()AWS SAMCLI. `sam local generate-event`

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'uso del AWS SAMCLI `sam local generate-event` comando, vedereIntroduzione ai test con sam local generate-event.

Il `sam local generate-event` sottocomando genera esempi di payload di eventi per i supporti. Servizi AWS

Utilizzo

```
$ sam local generate-event <options> <service> <event> <event-options>
```

Opzioni

`--config-env TEXT`

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file PATH`

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito si trova `samconfig.toml` nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--help`

Mostra questo messaggio ed esce.

Servizio

Per visualizzare un elenco dei servizi supportati, esegui quanto segue:

```
$ sam local generate-event
```

Event

Per visualizzare un elenco di eventi supportati che possono essere generati per ogni servizio, esegui quanto segue:

```
$ sam local generate-event <service>
```

Opzioni relative agli eventi

Per visualizzare un elenco delle opzioni di evento supportate che puoi modificare, esegui quanto segue:

```
$ sam local generate-event <service> <event> --help
```

Esempi

Per esempi sull'utilizzo del `sam local generate-event` sottocomando, fare riferimento a. Genera eventi di esempio

sam local invoke

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model `sam local invoke` sottocomando Command Line Interface (AWS SAMCLI).

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'uso del AWS SAMCLI `sam local invoke` sottocomando, vedereIntroduzione ai test con sam local invoke.

Il `sam local invoke` sottocomando avvia una chiamata una tantum di una funzione localmente. AWS Lambda

> **Note**
>
> Per funzioni durevoli, `sam local invoke` supporta l'esecuzione con stato con checkpoint e replay automatici. Il contenitore rimane in funzione durante l'esecuzione durevole della funzione per gestire la persistenza e la ripresa dello stato.

Utilizzo

```
$ sam local invoke <arguments> <options>
```

> **Note**
>
> Se hai più di una funzione definita nel tuo AWS SAM modello, fornisci l'ID logico della funzione che desideri richiamare.

Arguments (Argomenti)

Resource ID (ID risorsa)

L'ID della funzione Lambda da richiamare.

Questo argomento è facoltativo. Se l'applicazione contiene una singola funzione Lambda, la AWS SAM CLI la richiamerà. Se l'applicazione contiene più funzioni, fornisci l'ID della funzione da richiamare.

Valori validi: l'ID logico o l'ARN della risorsa.

Opzioni

`--add-host LIST`

Passa un nome host alla mappatura dell'indirizzo IP al file host del contenitore Docker. Questo parametro può essere passato più volte.

Example

Ad esempio: `--add-host example.com:127.0.0.1`

`--beta-features | --no-beta-features`

Consenti o nega le funzionalità beta.

`--config-env TEXT`

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *PATH*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è "`samconfig.toml`" nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--container-env-vars`

(Facoltativo) Passa le variabili di ambiente al contenitore di immagini della funzione Lambda durante il debug locale.

`--container-host` *TEXT*

Host del contenitore Lambda emulato localmente. Il valore predefinito è `localhost`.
Se vuoi eseguirlo AWS SAMCLI in un contenitore Docker su macOS, puoi specificare. `host.docker.internal` Se desideri eseguire il contenitore su un host diverso da quello AWS SAMCLI, puoi specificare l'indirizzo IP dell'host remoto.

`--container-host-interface` *TEXT*

L'indirizzo IP dell'interfaccia di rete host a cui devono collegarsi le porte del contenitore. Il valore predefinito è `127.0.0.1`. Si usa `0.0.0.0` per collegarsi a tutte le interfacce.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug AWS SAMCLI generati e per visualizzare i timestamp.

`--debug-args` *TEXT*

Argomenti aggiuntivi da passare al debugger.

`--debug-port, -d` *TEXT*

Quando specificato, avvia il contenitore di funzioni Lambda in modalità debug ed espone questa porta sull'host locale.

`--debugger-path` *TEXT*

Il percorso dell'host di un debugger montato nel contenitore Lambda.

`--docker-network` *TEXT*

Il nome o l'ID di una rete Docker esistente a cui i contenitori Lambda Docker devono connettersi, insieme alla rete bridge predefinita. Se questo non è specificato, i contenitori Lambda si connettono solo alla rete bridge Docker predefinita.

`--docker-volume-basedir, -v` *TEXT*

La posizione della directory di base in cui si trova il AWS SAM file. Se Docker è in esecuzione su un computer remoto, è necessario montare il percorso in cui esiste il AWS SAM file sulla macchina Docker e modificare questo valore in modo che corrisponda al computer remoto.

`--durable-execution-name` *TEXT*

Nome per l'esecuzione durevole (solo per funzioni durevoli)

`--env-vars, -n` *PATH*

Il file JSON che contiene i valori per le variabili di ambiente della funzione Lambda. Per ulteriori informazioni sui file delle variabili di ambiente, consulta. File di variabili di ambiente

`--event, -e` *PATH*

Il file JSON che contiene i dati degli eventi che vengono passati alla funzione Lambda quando viene richiamata. Se non si specifica questa opzione, non si presume alcun evento. Per inserire JSON `dastdin`, devi inserire il valore '-'. Per informazioni dettagliate sui formati dei messaggi relativi agli eventi di diversi AWS servizi, consulta Lavorare con altri servizi nella Guida per gli AWS Lambda sviluppatori.

`--force-image-build`

Specifica se AWS SAMCLI deve ricostruire l'immagine usata per richiamare le funzioni Lambda con livelli.

`--help`

Mostra questo messaggio ed esce.

`--hook-name TEXT`

Il nome dell'hook utilizzato per estendere le AWS SAMCLI funzionalità.

Valori accettati:`terraform`.

`--invoke-image` *TEXT*

L'URI dell'immagine del contenitore che si desidera utilizzare per l'invocazione della funzione locale. Per impostazione predefinita, AWS SAM estrae l'immagine del contenitore da Amazon ECR Public (che è elencato inArchivi di immagini per AWS SAM). Usa questa opzione per estrarre l'immagine da un'altra posizione.

Ad esempio, `sam local invoke MyFunction --invoke-image amazon/aws-sam-cli-emulation-image-python3.8`.

`--layer-cache-basedir` *DIRECTORY*

Specifica la posizione della directory di base in cui vengono scaricati i layer utilizzati dal modello.

`--log-file, -l` *TEXT*

Il file di registro a cui inviare i log di runtime.

`--mount-symlinks`

Assicura che monti AWS SAMCLI sempre i collegamenti simbolici presenti nei file da compilare o richiamare. Questo vale solo per i collegamenti simbolici nella directory di primo livello (ovvero i collegamenti simbolici che si trovano direttamente nella radice della funzione). Per impostazione predefinita, i collegamenti simbolici non vengono montati ad eccezione di quelli necessari per essere utilizzati `build-in-source` `node_modules` in NodeJS.

`--no-event`

Richiama la funzione con un evento vuoto.

`--no-memory-limit`

Rimuove la limitazione di memoria nel contenitore durante la chiamata locale, anche quando la memoria è configurata nel modello. AWS SAM

`--parameter-overrides`

Una stringa che contiene le sostituzioni CloudFormation dei parametri codificate come coppie chiave-valore. Utilizzate lo stesso formato di (). AWS Command Line Interface AWS CLI Il AWS SAMCLI formato è costituito da parole chiave e valori esplicite, ogni override è separata da uno spazio. Di seguito, sono riportati due esempi:

- `--parameter-overrides ParameterKey=hello,ParameterValue=world`
- `--parameter-overrides ParameterKey=hello,ParameterValue=world ParameterKey=example1,ParameterValue=example2 ParameterKey=apple,ParameterValue=banana`

`--profile` *TEXT*

Il profilo specifico del file delle credenziali che ottiene le credenziali. AWS

`--region` *TEXT*

La AWS regione in cui effettuare la distribuzione. Ad esempio, us-east-1.

`--runtime` *TEXT*

Utilizza il runtime specificato per richiamare una funzione Lambda localmente. Ciò sovrascrive il runtime definito nel file. `template.yml` Ciò consente inoltre di testare le funzioni Lambda con tempi di esecuzione diversi senza modificare la configurazione della funzione originale.

`--save-params`

Salva i parametri forniti nella riga di comando nel file di configurazione. AWS SAM

`--shutdown`

Emula un evento di spegnimento dopo il completamento dell'invoca, per testare la gestione delle estensioni del comportamento di spegnimento.

`--skip-prepare-infra`

Salta la fase di preparazione se non sono state apportate modifiche all'infrastruttura. Da utilizzare con l'`--hook-name`opzione.

`--skip-pull-image`

Per impostazione predefinita, AWS SAMCLI controlla l'ultimo ambiente di runtime remoto di Lambda e aggiorna automaticamente l'immagine locale per mantenerla sincronizzata.

Specificate questa opzione per saltare l'estrazione dell'Dockerimmagine più recente per il vostro ambiente di runtime Lambda.

`--template, -t` *PATH*

Il AWS SAM file modello.

Questa opzione non è compatibile con`--hook-name`.

> **Note**
>
> Se si specifica questa opzione, AWS SAM carica solo il modello e le risorse locali a cui punta.

`--tenant-id` *TEXT*

L'ID tenant per le funzioni Lambda multi-tenant. Utilizzato per garantire l'isolamento del calcolo tra diversi tenant. Richiesto quando si richiamano funzioni configurate con la modalità di isolamento dei tenant.

`--terraform-plan-file`

Il percorso relativo o assoluto del file del Terraform piano locale quando si utilizza with. AWS SAMCLI Terraform Cloud Questa opzione richiede che `--hook-name` sia impostata `suterraform`.

Esempi

L'esempio seguente utilizza un evento generato per i test locali utilizzando un `s3.json` evento per richiamare localmente una funzione Lambda.

```
$ sam local invoke --event events/s3.json S3JsonLoggerFunction
```

L'esempio seguente verifica la funzione `HelloWorldFunction` utilizzando il runtime Python 3.11

```
$ sam local invoke --runtime python3.11 HelloWorldFunction
```

L'esempio seguente verifica la funzione `HelloWorldFunction` con un nome di esecuzione durevole

```
$ sam local invoke HelloWorldFunction --durable-execution-name my-execution
```

sam local start-api

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model sottocomando Command Line Interface (AWS SAMCLI). `sam local start-api`

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'uso del AWS SAMCLI `sam local start-api` sottocomando, vedereIntroduzione ai test con sam local start-api.

Il `sam local start-api` sottocomando esegue le AWS Lambda funzioni localmente per eseguirle tramite un host di server HTTP locale.

Utilizzo

```
$ sam local start-api <options>
```

Opzioni

`--add-host LIST`

Passa un nome host alla mappatura dell'indirizzo IP al file host del contenitore Docker. Questo parametro può essere passato più volte.

Example

Ad esempio: `--add-host example.com:127.0.0.1`

`--beta-features | --no-beta-features`

Consenti o nega le funzionalità beta.

`--config-env TEXT`

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file PATH`

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è «samconfig.toml» nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--container-env-vars`

Opzionale. Passa le variabili di ambiente al contenitore di immagini durante il debug locale.

`--container-host TEXT`

Host del contenitore Lambda emulato localmente. Il valore predefinito è `localhost`.
Se vuoi eseguirlo AWS SAMCLI in un contenitore Docker su macOS, puoi specificare. `host.docker.internal` Se desideri eseguire il contenitore su un host diverso da quello AWS SAMCLI, puoi specificare l'indirizzo IP dell'host remoto.

`--container-host-interface TEXT`

L'indirizzo IP dell'interfaccia di rete host a cui devono collegarsi le porte del contenitore. Il valore predefinito è `127.0.0.1`. Si usa `0.0.0.0` per collegarsi a tutte le interfacce.

`--debug`

Attiva la registrazione di debug per stampare il messaggio di debug generato dai timestamp e visualizzarli. AWS SAMCLI

`--debug-args` *TEXT*

Argomenti aggiuntivi da passare al debugger.

`--debug-function`

Opzionale. Specifica la funzione Lambda a cui applicare le opzioni di debug quando viene specificato. `--warm-containers` Questo parametro si applica a`--debug-port`, `--debugger-path` e. `--debug-args`

`--debug-port, -d` *TEXT*

Quando specificato, avvia il contenitore di funzioni Lambda in modalità debug ed espone questa porta sull'host locale.

`--debugger-path` *TEXT*

Il percorso host di un debugger che verrà montato nel contenitore Lambda.

`--docker-network` *TEXT*

Il nome o l'ID di una rete Docker esistente a cui i contenitori Lambda Docker devono connettersi, insieme alla rete bridge predefinita. Se questo non è specificato, i contenitori Lambda si connettono solo alla rete bridge Docker predefinita.

`--docker-volume-basedir, -v` *TEXT*

La posizione della directory di base in cui si trova il AWS SAM file. Se Docker è in esecuzione su un computer remoto, è necessario montare il percorso in cui esiste il AWS SAM file sulla macchina Docker e modificare questo valore in modo che corrisponda al computer remoto.

`--env-vars, -n` *PATH*

Il file JSON che contiene i valori per le variabili di ambiente della funzione Lambda.

`--force-image-build`

Speciifica se AWS SAM CLI deve ricostruire l'immagine usata per richiamare funzioni con livelli.

`--help`

Mostra questo messaggio ed esce.

`--hook-name` *TEXT*

Il nome dell'hook utilizzato per estendere le AWS SAMCLI funzionalità.

Valori accettati:`terraform`.

`--host` *TEXT*

Il nome host o l'indirizzo IP locale a cui collegarsi (impostazione predefinita: '127.0.0.1').

`--invoke-image` *TEXT*

L'URI dell'immagine del contenitore che desideri utilizzare per le funzioni Lambda. Per impostazione predefinita, AWS SAM estrae l'immagine del contenitore da Amazon ECR Public. Usa questa opzione per estrarre l'immagine da un'altra posizione.

Puoi specificare questa opzione più volte. Ogni istanza di questa opzione può accettare una stringa o una coppia chiave-valore. Se specificate una stringa, è l'URI dell'immagine del contenitore da utilizzare per tutte le funzioni dell'applicazione. Ad esempio, `sam local start-api --invoke-image public.ecr.aws/sam/emu-python3.8`. Se specificate una coppia chiave-valore, la chiave è il nome della risorsa e il valore è l'URI dell'immagine del contenitore da utilizzare per quella risorsa. Ad esempio, `sam local start-api --invoke-image public.ecr.aws/sam/emu-python3.8 --invoke-image Function1=amazon/aws-sam-cli-emulation-image-python3.8` . Con le coppie chiave-valore, è possibile specificare immagini di contenitori diverse per risorse diverse.

`--layer-cache-basedir` *DIRECTORY*

Specifica la posizione basedir in cui vengono scaricati i layer utilizzati dal modello.

`--log-file, -l` *TEXT*

Il file di registro a cui inviare i log di runtime.

`--no-memory-limit`

Rimuove la limitazione della memoria nel contenitore durante la chiamata locale, anche quando la memoria è configurata nel AWS SAM modello.

`--parameter-overrides`

Una stringa che contiene le sostituzioni CloudFormation dei parametri codificate come coppie chiave-valore. Utilizzate lo stesso formato di (). AWS Command Line Interface AWS CLI Il AWS SAMCLI formato è costituito da parole chiave e valori esplicite, ogni override è separata da uno spazio. Di seguito, sono riportati due esempi:

- `--parameter-overrides ParameterKey=hello,ParameterValue=world`
- `--parameter-overrides ParameterKey=hello,ParameterValue=world ParameterKey=example1,ParameterValue=example2 ParameterKey=apple,ParameterValue=banana`

`--port, -p` *`INTEGER`*

Il numero di porta locale su cui ascoltare (impostazione predefinita: '3000').

`--profile` *`TEXT`*

Il profilo specifico del file delle credenziali che ottiene le AWS credenziali.

`--region` *`TEXT`*

La AWS regione in cui effettuare la distribuzione. Ad esempio, us-east-1.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file di AWS SAM configurazione.

`--shutdown`

Emula un evento di spegnimento dopo il completamento dell'invocazione, per testare la gestione delle estensioni del comportamento di spegnimento.

`--skip-prepare-infra`

Salta la fase di preparazione se non sono state apportate modifiche all'infrastruttura. Da utilizzare con l'`--hook-name`opzione.

`--skip-pull-image`

Specifica se la CLI deve ignorare l'estrazione dell'ultima immagine Docker per il runtime Lambda.

`--ssl-cert-file` *`PATH`*

Percorso del file del certificato SSL (impostazione predefinita: Nessuno). Quando si utilizza questa opzione, è necessario utilizzare anche l'`--ssl-key-file`opzione.

`--ssl-key-file` *`PATH`*

Percorso del file della chiave SSL (impostazione predefinita: Nessuno). Quando si utilizza questa opzione, è necessario utilizzare anche l'`--ssl-cert-file`opzione.

`--static-dir, -s` *`TEXT`*

Tutti i file di risorse statici (ad esempioCSS/JavaScript/HTML) che si trovano in questa directory vengono presentati in`/`.

`--template, -t PATH`

Il file AWS SAM modello.

> **Note**
>
> Se si specifica questa opzione, AWS SAM carica solo il modello e le risorse locali a cui punta.

`--terraform-plan-file`

Il percorso relativo o assoluto del file del Terraform piano locale quando si utilizza AWS SAMCLI withTerraform Cloud. Questa opzione richiede che `--hook-name` sia impostata `suterraform`.

`--warm-containers [EAGER | LAZY]`

Opzionale. Specifica come AWS SAMCLI gestisce i contenitori per ogni funzione.

Sono disponibili due opzioni:

`EAGER`: I contenitori per tutte le funzioni vengono caricati all'avvio e persistono tra le chiamate.

`LAZY`: I contenitori vengono caricati solo quando ogni funzione viene richiamata per la prima volta. Questi contenitori persistono per chiamate aggiuntive.

Esempio

L'esempio seguente avvia un server locale, che consente di testare l'applicazione tramite API. Affinché questo comando funzioni, l'applicazione deve essere installata e Docker deve essere in esecuzione.

```
$ sam local start-api --port 3000
```

sam local start-lambda

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model sottocomando Command Line Interface ()AWS SAMCLI. `sam local start-lambda`

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

- Per la documentazione sull'uso del AWS SAMCLI `sam local start-lambda` sottocomando, vedereIntroduzione ai test con sam local start-lambda.

Il `sam local start-lambda` sottocomando avvia un endpoint locale da emulare. AWS Lambda

Utilizzo

```
$ sam local start-lambda <options>
```

Opzioni

`--add-host` *`LIST`*

Passa un nome host alla mappatura dell'indirizzo IP al file host del contenitore Docker. Questo parametro può essere passato più volte.

Example

Ad esempio: `--add-host` *`example.com:127.0.0.1`*

`--beta-features | --no-beta-features`

Consenti o nega le funzionalità beta.

`--config-env` *`TEXT`*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *`PATH`*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è «samconfig.toml» nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--container-env-vars`

Opzionale. Passa le variabili di ambiente al contenitore di immagini durante il debug locale.

`--container-host` *`TEXT`*

Host del contenitore Lambda emulato localmente. Il valore predefinito è `localhost`.
Se vuoi eseguirlo AWS SAMCLI in un contenitore Docker su macOS, puoi specificare.

`host.docker.internal` Se desideri eseguire il contenitore su un host diverso da quello AWS SAMCLI, puoi specificare l'indirizzo IP dell'host remoto.

`--container-host-interface` *TEXT*

L'indirizzo IP dell'interfaccia di rete host a cui devono collegarsi le porte del contenitore. Il valore predefinito è `127.0.0.1`. Si usa `0.0.0.0` per collegarsi a tutte le interfacce.

`--debug`

Attiva la registrazione di debug per stampare il messaggio di debug generato dai timestamp e visualizzarli. AWS SAMCLI

`--debug-args` *TEXT*

Argomenti aggiuntivi da passare al debugger.

`--debug-function`

Opzionale. Specifica la funzione Lambda a cui applicare le opzioni di debug quando viene specificato. `--warm-containers` Questo parametro si applica a`--debug-port`, `--debugger-path` e. `--debug-args`

`--debug-port, -d` *TEXT*

Quando specificato, avvia il contenitore di funzioni Lambda in modalità debug ed espone questa porta sull'host locale.

`--debugger-path` *TEXT*

Il percorso dell'host verso un debugger da montare nel contenitore Lambda.

`--docker-network` *TEXT*

Il nome o l'ID di una rete Docker esistente a cui i contenitori Lambda Docker devono connettersi, insieme alla rete bridge predefinita. Se viene specificato, i contenitori Lambda si connettono solo alla rete bridge Docker predefinita.

`--docker-volume-basedir, -v` *TEXT*

La posizione della directory di base in cui si trova il AWS SAM file. Se Docker è in esecuzione su un computer remoto, è necessario montare il percorso in cui esiste il AWS SAM file sulla macchina Docker e modificare questo valore in modo che corrisponda al computer remoto.

`--env-vars, -n` *PATH*

Il file JSON che contiene i valori per le variabili di ambiente della funzione Lambda.

`--force-image-build`

Specificate se CLI deve ricostruire l'immagine usata per richiamare funzioni con livelli.

`--help`

Mostra questo messaggio ed esce.

`--hook-name` *TEXT*

Il nome dell'hook utilizzato per estendere le AWS SAMCLI funzionalità.

Valori accettati:`terraform`.

`--host` *TEXT*

Il nome host o l'indirizzo IP locale a cui collegarsi (impostazione predefinita: '127.0.0.1').

`--invoke-image` *TEXT*

L'URI dell'immagine del contenitore che si desidera utilizzare per l'invocazione della funzione locale. Per impostazione predefinita, AWS SAM estrae l'immagine del contenitore da Amazon ECR Public. Usa questa opzione per estrarre l'immagine da un'altra posizione.

Ad esempio, `sam local start-lambda MyFunction --invoke-image amazon/aws-sam-cli-emulation-image-python3.8`.

`--layer-cache-basedir` *DIRECTORY*

Specifica la posizione basedir in cui vengono scaricati i layer utilizzati dal modello.

`--log-file, -l` *TEXT*

Il file di registro a cui inviare i log di runtime.

`--no-memory-limit`

Rimuove la limitazione della memoria nel contenitore durante la chiamata locale, anche quando la memoria è configurata nel AWS SAM modello.

`--parameter-overrides`

Una stringa che contiene le sostituzioni CloudFormation dei parametri codificate come coppie chiave-valore. Utilizzate lo stesso formato di (). AWS Command Line Interface AWS CLI Il AWS SAMCLI formato è costituito da parole chiave e valori esplicite, ogni override è separata da uno spazio. Di seguito, sono riportati due esempi:

- `--parameter-overrides ParameterKey=hello,ParameterValue=world`
- `--parameter-overrides ParameterKey=hello,ParameterValue=world ParameterKey=example1,ParameterValue=example2 ParameterKey=apple,ParameterValue=banana`

`--port, -p` *`INTEGER`*

Il numero di porta locale su cui ascoltare (impostazione predefinita: '3001').

`--profile` *`TEXT`*

Il profilo specifico del file delle credenziali che ottiene AWS le credenziali.

`--region` *`TEXT`*

La AWS regione in cui effettuare la distribuzione. Ad esempio, us-east-1.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file di AWS SAM configurazione.

`--shutdown`

Emula un evento di spegnimento dopo il completamento dell'invoca, per testare la gestione delle estensioni del comportamento di spegnimento.

`--skip-prepare-infra`

Salta la fase di preparazione se non sono state apportate modifiche all'infrastruttura. Da utilizzare con l'`--hook-name`opzione.

`--skip-pull-image`

Specifica se CLI non si deve estrarre l'ultima immagine Docker per il runtime Lambda.

`--template, -t` *`PATH`*

AWS SAM Il file modello.

> **Note**
>
> Se si specifica questa opzione, AWS SAM carica solo il modello e le risorse locali a cui punta. Questa opzione non è compatibile con`--hook-name`.

`--terraform-plan-file`

Il percorso relativo o assoluto del file del Terraform piano locale quando si utilizza AWS SAMCLI withTerraform Cloud. Questa opzione richiede che `--hook-name` sia impostata `suterraform`.

`--warm-containers` *`[EAGER | LAZY]`*

Opzionale. Specifica come AWS SAMCLI gestisce i contenitori per ogni funzione.

Sono disponibili due opzioni:

- `EAGER`: I contenitori per tutte le funzioni vengono caricati all'avvio e persistono tra le chiamate.
- `LAZY`: I contenitori vengono caricati solo quando ogni funzione viene richiamata per la prima volta. Questi contenitori persistono per chiamate aggiuntive.

Esempio

Per un esempio dettagliato e una procedura dettagliata sull'utilizzo del `sam local start-lambda` sottocomando, fare riferimento a. Introduzione ai test con sam local start-lambda

sam logs

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam logs`.

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam logs` comando recupera i log generati dalle tue AWS Lambda funzioni.

Utilizzo

```
$ sam logs <options>
```

Opzioni

`--config-env` *`TEXT`*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file PATH`

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è «samconfig.toml» nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--cw-log-group LIST`

Include i log dei gruppi di CloudWatch log Logs specificati. Se si specifica questa opzione insieme aname, AWS SAM include i registri dei gruppi di log specificati oltre ai registri delle risorse denominate.

`--debug`

Attiva la registrazione di debug per stampare il messaggio di debug generato dai timestamp e visualizzarli. AWS SAMCLI

`---end-time, e TEXT`

Recupera i log aggiornati a quest'ora. L'ora può essere costituita da valori relativi come «5 minuti fa», «domani» o un timestamp formattato come «2018-01-01 10:10:10».

`--filter TEXT`

Consente di specificare un'espressione per trovare rapidamente i log che corrispondono a termini, frasi o valori negli eventi di registro. Può trattarsi di una semplice parola chiave (ad esempio «errore») o di un pattern supportato da Amazon CloudWatch Logs. Per la sintassi, consulta la documentazione di Amazon CloudWatch Logs.

`--help`

Mostra questo messaggio ed esce.

`--include-traces`

Include tracce X-Ray nell'output del registro.

`--name, -n TEXT`

Il nome della risorsa per la quale recuperare i log. Se questa risorsa fa parte di uno CloudFormation stack, può essere l'ID logico della risorsa funzionale nel CloudFormation modello /.AWS SAM È possibile fornire più nomi ripetendo nuovamente il parametro. Se la risorsa si trova in uno stack nidificato, al nome può essere aggiunto il nome dello stack nidificato per estrarre i log da quella risorsa (/). NestedStackLogicalId ResourceLogicalId Se il nome della risorsa non viene fornito, lo stack specificato verrà scansionato e verranno recuperate le

informazioni di registro per tutte le risorse supportate. Se non specificate questa opzione, AWS SAM recupera i log per tutte le risorse nello stack specificato. Sono supportati i seguenti tipi di risorse:

- `AWS::Serverless::Function`
- `AWS::Lambda::Function`
- `AWS::Serverless::Api`
- `AWS::ApiGateway::RestApi`
- `AWS::Serverless::HttpApi`
- `AWS::ApiGatewayV2::Api`
- `AWS::Serverless::StateMachine`
- `AWS::StepFunctions::StateMachine`

`--output` *TEXT*

Speciifica il formato di output per i registri. Per stampare registri formattati, specificare. `text` Per stampare i log come JSON, specificare. `json`

`--profile` *TEXT*

Il profilo specifico del file delle credenziali che ottiene le credenziali. AWS

`--region` *TEXT*

La AWS regione in cui effettuare la distribuzione. Ad esempio, us-east-1.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file di AWS SAM configurazione.

`--stack-name` *TEXT*

Il nome dello CloudFormation stack di cui fa parte la risorsa.

`--start-time, -s` *TEXT*

Recupera i log a partire da questo momento. L'ora può essere costituita da valori relativi come «5 minuti fa», «ieri» o un timestamp formattato come «2018-01-01 10:10:10». Il valore predefinito è «10 minuti fa».

`--tail, -t`

Coordina l'output del registro. Questo ignora l'argomento dell'ora di fine e continua a recuperare i log non appena diventano disponibili.

Esempi

Quando le funzioni fanno parte di uno CloudFormation stack, puoi recuperare i log utilizzando l'ID logico della funzione quando specifichi il nome dello stack.

```
$ sam logs -n HelloWorldFunction --stack-name myStack
```

Visualizza i log per un intervallo di tempo specifico utilizzando le opzioni -s (--start-time) e -e (--end-time).

```
$ sam logs -n HelloWorldFunction --stack-name myStack -s '10min ago' -e '2min ago'
```

Puoi anche aggiungere l'`--tail`opzione per attendere nuovi log e vederli man mano che arrivano.

```
$ sam logs -n HelloWorldFunction --stack-name myStack --tail
```

Utilizza l'`--filter`opzione per trovare rapidamente i log che corrispondono a termini, frasi o valori nei tuoi eventi di registro.

```
$ sam logs -n HelloWorldFunction --stack-name myStack --filter "error"
```

Visualizza i log di una risorsa in uno stack secondario.

```
$ sam logs --stack-name myStack -n childStack/HelloWorldFunction
```

Registri di coda per tutte le risorse supportate nell'applicazione.

```
$ sam logs --stack-name sam-app --tail
```

Recupera i log per una funzione Lambda specifica e l'API API Gateway nella tua applicazione.

```
$ sam logs --stack-name sam-app --name HelloWorldFunction --name HelloWorldRestApi
```

Recupera i log per tutte le risorse supportate nell'applicazione e, in aggiunta, dai gruppi di log specificati.

```
$ sam logs --cw-log-group /aws/lambda/myfunction-123 --cw-log-group /aws/lambda/
myfunction-456
```

sam package

La AWS Serverless Application Model Command Line Interface (AWS SAM CLI) impacchetta un' AWS SAM applicazione.

Questo comando crea un `.zip` file con il codice e le dipendenze e carica il file su Amazon Simple Storage Service (Amazon S3). AWS SAM abilita la crittografia per tutti i file archiviati in Amazon S3. Quindi restituisce una copia del AWS SAM modello, sostituendo i riferimenti agli artefatti locali con la posizione Amazon S3 in cui il comando ha caricato gli artefatti.

Per impostazione predefinita, quando si utilizza questo comando, si AWS SAMCLI presuppone che la directory di lavoro corrente sia la directory principale del progetto. Il AWS SAMCLI primo tenta di individuare un file modello creato utilizzando il sam build comando, situato nella `.aws-sam` sottocartella e denominato. `template.yaml` Successivamente, AWS SAMCLI tenta di individuare un file modello denominato `template.yaml` o `template.yml` nella directory di lavoro corrente. Se si specifica l'`--template`opzione, il comportamento predefinito AWS SAMCLI dell'opzione viene sovrascritto e includerà solo il AWS SAM modello e le risorse locali a cui punta.

> **Note**
>
> sam deployora esegue implicitamente la funzionalità di. `sam package` È possibile utilizzare il sam deploy comando direttamente per impacchettare e distribuire l'applicazione.

Utilizzo

```
$ sam package <arguments> <options>
```

Arguments (Argomenti)

Resource ID (ID risorsa)

L'ID della funzione Lambda da impacchettare.

Questo argomento è facoltativo. Se l'applicazione contiene una singola funzione Lambda, la AWS SAM CLI la impacchetterà. Se l'applicazione contiene più funzioni, fornisci l'ID della funzione per impacchettare una singola funzione.

Valori validi: l'ID logico o l'ARN della risorsa.

Opzioni

`--config-env` *TEXT*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *PATH*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è «samconfig.toml» nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--debug`

Attiva la registrazione di debug per stampare il messaggio di debug generato dai timestamp e visualizzarli. AWS SAMCLI

`--force-upload`

Sostituisci i file esistenti nel bucket Amazon S3. Specificate questo flag per caricare gli artefatti anche se corrispondono agli artefatti esistenti nel bucket Amazon S3.

`--help`

Mostra questo messaggio ed esce.

`--image-repository` *TEXT*

L'URI del repository Amazon Elastic Container Registry (Amazon ECR) in cui questo comando carica l'immagine della funzione. Obbligatorio per le funzioni dichiarate con il `Image` tipo di pacchetto.

`--kms-key-id` *TEXT*

L'ID di una chiave AWS Key Management Service (AWS KMS) utilizzata per crittografare gli artefatti che sono a riposo nel bucket Amazon S3. Se questa opzione non è specificata, AWS SAM utilizza le chiavi di crittografia gestite da Amazon S3.

`--metadata`

(Facoltativo) Una mappa di metadati da allegare a tutti gli artefatti a cui si fa riferimento nel modello.

`--no-progressbar`

Non visualizzare una barra di avanzamento durante il caricamento di artefatti su Amazon S3.

`--output-template-file` *PATH*

Il percorso del file in cui il comando scrive il modello confezionato. Se non specificate un percorso, il comando scrive il modello nello standard output.

`--profile` *TEXT*

Il profilo specifico del file delle credenziali che ottiene le AWS credenziali.

`--region` *TEXT*

La AWS regione in cui effettuare la distribuzione. Ad esempio, us-east-1.

`--resolve-s3`

Crea automaticamente un bucket Amazon S3 da utilizzare per il packaging. Se specifichi entrambe le `--resolve-s3` opzioni `--s3-bucket` e, si verificherà un errore.

`--s3-bucket` *TEXT*

Il nome del bucket Amazon S3 in cui questo comando carica il tuo artefatto. Se il tuo artefatto è più grande di 51.200 byte, è richiesta l'opzione o l'opzione. `--s3-bucket` `--resolve-s3` Se si specificano entrambe le `--resolve-s3` opzioni `--s3-bucket` e, si verificherà un errore.

`--s3-prefix` *TEXT*

Prefisso aggiunto al nome degli artefatti che vengono caricati nel bucket Amazon S3. Il nome del prefisso è un nome di percorso (nome della cartella) per il bucket Amazon S3. Questo vale solo per le funzioni dichiarate con `Zip` il tipo di pacchetto.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file AWS SAM di configurazione.

`--signing-profiles` *LIST*

(Facoltativo) L'elenco dei profili di firma con cui firmare i pacchetti di distribuzione. Questo parametro accetta un elenco di coppie chiave-valore, in cui la chiave è il nome della funzione o del livello da firmare e il valore è il profilo di firma, con un proprietario del profilo opzionale delimitato da. `:` Ad esempio, `FunctionNameToSign=SigningProfileName1 LayerNameToSign=SigningProfileName2:SigningProfileOwner`.

`--template-file, --template, -t PATH`

Il percorso e il nome del file in cui si trova il AWS SAM modello.

> **Note**
>
> Se si specifica questa opzione, AWS SAM impacchetta solo il modello e le risorse locali a cui punta.

`--use-json`

Esporta JSON per il CloudFormation modello. Per impostazione predefinita, viene utilizzato YAML.

Esempio

L'esempio seguente crea e impacchetta artefatti per una funzione e applicazioni Lambda. CodeDeploy Gli artefatti vengono caricati in un bucket Amazon S3. L'output del comando è un nuovo file denominato `package.yml`.

```
$ sam package \
  --template-file template.yml \
  --output-template-file package.yml \
  --s3-bucket amzn-s3-demo-bucket
```

sam pipeline bootstrap

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model `sam local pipeline bootstrap` sottocomando Command Line Interface (AWS SAMCLI).

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam pipeline bootstrap` sottocomando genera le risorse di AWS infrastruttura necessarie per connettersi al sistema CI/CD. Questo passaggio deve essere eseguito per ogni fase di distribuzione nella pipeline prima di eseguire il comando. sam pipeline init

Questo sottocomando imposta le seguenti risorse di AWS infrastruttura:

- Possibilità di configurare le autorizzazioni della pipeline tramite:

- Un utente IAM della pipeline con ID della chiave di accesso e credenziali di accesso alla chiave segreta da condividere con il sistema. CI/CD

> **Note**
>
> Consigliamo di ruotare regolarmente le chiavi di accesso. Per ulteriori informazioni, consulta Ruotare regolarmente le chiavi di accesso per i casi d'uso che richiedono credenziali a lungo termine nella Guida per l'utente IAM.

- CI/CD Piattaforme supportate tramite OIDC. Per un'introduzione all'uso di OIDC con AWS SAM pipeline, vai a. Come utilizzare l'autenticazione OIDC con le pipeline AWS SAM

- Un ruolo IAM di CloudFormation esecuzione assunto da CloudFormation per distribuire l'applicazione. AWS SAM
- Un bucket Amazon S3 per contenere gli artefatti. AWS SAM
- Facoltativamente, un repository di immagini Amazon ECR per contenere pacchetti di distribuzione Lambda con immagini container (se disponi di una risorsa di tipo pacchetto). `Image`

Utilizzo

```
$ sam pipeline bootstrap <options>
```

Opzioni

`--bitbucket-repo-uuid` *`TEXT`*

L'UUID del repository Bitbucket. Questa opzione è specifica per l'utilizzo di Bitbucket OIDC per le autorizzazioni.

> **Note**
>
> Questo valore è disponibile all'indirizzo https://bitbucket.org///-connect *`workspace repository`* admin/addon/admin/pipelines/openid

`--bucket` *`TEXT`*

L'ARN del bucket Amazon S3 che contiene gli artefatti. AWS SAM

`--cicd-provider` *TEXT*

La CI/CD piattaforma per la pipeline. AWS SAM

`--cloudformation-execution-role` *TEXT*

L'ARN del ruolo IAM da assumere CloudFormation durante la distribuzione dello stack dell'applicazione. Forniscilo solo se desideri utilizzare il tuo ruolo. Altrimenti, il comando creerà un nuovo ruolo.

`--config-env` *TEXT*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è **`default`**. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *PATH*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito si trova `samconfig.toml` nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--confirm-changeset | --no-confirm-changeset`

Richiedi di confermare la distribuzione delle tue risorse.

`--create-image-repository | --no-create-image-repository`

Specificare se creare un repository di immagini Amazon ECR se non ne viene fornito nessuno. Il repository Amazon ECR contiene le immagini del contenitore delle funzioni Lambda o dei livelli con un tipo di pacchetto di. `Image` Il valore predefinito è `--no-create-image-repository`.

`--debug`

Attiva la registrazione di debug e stampa i messaggi di debug che AWS SAMCLI genera e visualizza i timestamp.

`--deployment-branch` *TEXT*

Nome del ramo da cui verranno effettuate le distribuzioni. Questa opzione è specifica per l'utilizzo di GitHub Actions OIDC per le autorizzazioni.

`--github-org` *TEXT*

L' GitHub organizzazione a cui appartiene il repository. Se non esiste alcuna organizzazione, inserisci il nome utente del proprietario del repository. Questa opzione è specifica per l'utilizzo di GitHub Actions OIDC per le autorizzazioni.

`--github-repo` *TEXT*

Nome del GitHub repository da cui verranno eseguite le distribuzioni. Questa opzione è specifica per l'utilizzo di GitHub Actions OIDC per le autorizzazioni.

`--gitlab-group` *TEXT*

Il GitLab gruppo a cui appartiene il repository. Questa opzione è specifica per l'utilizzo di GitLab OIDC per le autorizzazioni.

`--gitlab-project` *TEXT*

Il GitLab nome del progetto. Questa opzione è specifica per l'utilizzo di GitLab OIDC per le autorizzazioni.

`--help, -h`

Mostra questo messaggio ed esce.

`--image-repository` *TEXT*

L'ARN di un repository di immagini Amazon ECR che contiene le immagini del contenitore di funzioni Lambda o livelli con un tipo di pacchetto di. `Image` Se fornite, le opzioni vengono ignorate. `--create-image-repository` Se non viene fornito e `--create-image-repository` viene specificato, il comando ne crea una.

`--interactive | --no-interactive`

Disabilita la richiesta interattiva per i parametri di bootstrap e fallisce se mancano i parametri richiesti. Il valore predefinito è `--interactive`. Per questo comando, `--stage` è l'unico parametro richiesto.

> **Note**
>
> Se `--no-interactive` specificato insieme a`--use-oidc-provider`, devono essere inclusi tutti i parametri richiesti per il provider OIDC.

`--oidc-client-id` *TEXT*

L'ID client configurato per l'uso con il provider OIDC.

`--oidc-provider` *[github-actions | gitlab | bitbucket-pipelines]*

Nome del CI/CD provider che verrà utilizzato per le autorizzazioni OIDC. GitLab GitHub, e Bitbucket sono supportati.

`--oidc-provider-url` *TEXT*

L'URL del provider OIDC. Il valore deve iniziare con. **`https://`**

`--permissions-provider` *[oidc | iam]*

Scegliete un fornitore di autorizzazioni per assumere il ruolo di esecuzione della pipeline. Il valore predefinito è **`iam`**.

`--pipeline-execution-role` *TEXT*

L'ARN del ruolo IAM che l'utente della pipeline deve assumere per operare in questa fase. Forniscilo solo se desideri utilizzare il tuo ruolo. Se non fornito, questo comando creerà un nuovo ruolo.

`--pipeline-user` *TEXT*

L'Amazon Resource Name (ARN) dell'utente IAM con l'ID della chiave di accesso e la chiave di accesso segreta condivisi con il CI/CD sistema. Viene utilizzato per concedere a questo utente IAM l'autorizzazione ad accedere all' AWS account corrispondente. Se non viene fornito, il comando creerà un utente IAM insieme all'ID della chiave di accesso e alle credenziali della chiave di accesso segreta.

`--profile` *TEXT*

Il profilo specifico del file di credenziali che ottiene AWS le credenziali.

`--region` *TEXT*

La AWS regione in cui effettuare la distribuzione. Ad esempio, `us-east-1`.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file di AWS SAM configurazione.

`--stage` *TEXT*

Il nome della fase di distribuzione corrispondente. Viene utilizzato come suffisso per le risorse AWS infrastrutturali create.

Risoluzione dei problemi

Errore: parametro obbligatorio mancante

Se `--no-interactive` viene specificato insieme a uno qualsiasi dei parametri richiesti `--use-oidc-provider` e non viene fornito, questo messaggio di errore verrà visualizzato insieme a una descrizione dei parametri mancanti.

Esempio

L'esempio seguente crea le AWS risorse necessarie per creare il CI/CD sistema, attiva la registrazione di debug e stampa i messaggi di debug generati da AWS SAMCLI: utilizza un evento generato per i test locali utilizzando un evento per richiamare una funzione `s3.json` Lambda localmente

```
$ sam pipeline bootstrap --debug
```

sam pipeline init

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model `sam pipeline init` sottocomando Command Line Interface (AWS SAMCLI).

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam pipeline init` sottocomando genera un file di configurazione della pipeline che il CI/CD sistema può utilizzare per distribuire applicazioni serverless utilizzando. AWS SAM

Prima dell'usosam pipeline init, è necessario avviare le risorse necessarie per ogni fase della pipeline. Puoi farlo eseguendo sam pipeline init --bootstrap per essere guidato nel processo di generazione del file di installazione e configurazione oppure facendo riferimento alle risorse che hai creato in precedenza con il sam pipeline bootstrap comando.

Utilizzo

```
$ sam pipeline init <options>
```

Opzioni

`--bootstrap`

Abilita la modalità interattiva che guida l'utente nella creazione delle risorse AWS infrastrutturali necessarie.

`--config-env` *TEXT*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è `default`. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *TEXT*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito si trova `samconfig.toml` nella directory principale del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug AWS SAMCLI generati e per visualizzare i timestamp.

`--help, -h`

Mostra questo messaggio ed esce.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file di AWS SAM configurazione.

Esempio

L'esempio seguente mostra come utilizzare l'`--bootstrap`opzione che consente di utilizzare una modalità interattiva che guida l'utente nella creazione delle risorse infrastrutturali necessarie AWS :

```
$ sam pipeline init --bootstrap
```

sam publish

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam publish`.

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam publish` comando pubblica un' AWS SAM applicazione in. AWS Serverless Application Repository Questo comando accetta un AWS SAM modello impacchettato e pubblica l'applicazione nella regione specificata. AWS

Il `sam publish` comando prevede che il AWS SAM modello includa una `Metadata` sezione contenente i metadati dell'applicazione necessari per la pubblicazione. Nella `Metadata` sezione, le `ReadmeUrl` proprietà `LicenseUrl` and devono fare riferimento ai bucket Amazon Simple Storage Service (Amazon S3), non ai file locali. Per ulteriori informazioni sulla `Metadata` sezione del AWS SAM modello, consulta. Pubblicazione dell'applicazione con AWS SAMCLI

Per impostazione predefinita, `sam publish` crea l'applicazione come privata. Prima di consentire ad altri AWS account di visualizzare e distribuire l'applicazione, è necessario condividerla. Per informazioni sulla condivisione delle applicazioni, consulta Esempi di policy AWS Serverless Application Repository basate sulle risorse nella Guida per gli sviluppatori.AWS Serverless Application Repository

> **Note**
>
> Attualmente `sam publish` non supporta la pubblicazione di applicazioni annidate specificate localmente. Se l'applicazione contiene applicazioni nidificate, è necessario pubblicarle separatamente nell'applicazione principale AWS Serverless Application Repository prima di pubblicare l'applicazione principale.

Utilizzo

```
$ sam publish <options>
```

Opzioni

`--config-env` *`TEXT`*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *`PATH`*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è "`samconfig.toml`" nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug AWS SAMCLI generati e per visualizzare i timestamp.

`--help`

Mostra questo messaggio ed esce.

`--profile` *`TEXT`*

Il profilo specifico contenuto nel file delle credenziali che ottiene AWS le credenziali.

`--region` *`TEXT`*

La AWS regione in cui effettuare la distribuzione. Ad esempio, us-east-1.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file di AWS SAM configurazione.

`--semantic-version` *`TEXT`*

(Facoltativo) Utilizzate questa opzione per fornire una versione semantica dell'applicazione che sostituisca la `SemanticVersion` proprietà nella `Metadata` sezione del file modello. Per ulteriori informazioni sul controllo delle versioni semantiche, vedete la specifica Semantic Versioning.

`--template, -t` *`PATH`*

Il percorso del file modello. AWS SAM `[default: template.[yaml|yml]]`

Esempi

Per pubblicare un'applicazione:

```
$ sam publish --template packaged.yaml --region us-east-1
```

sam remote callback

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam remote callback`.

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam remote callback` comando consente di inviare callback a esecuzioni di funzioni durature remote in esecuzione in AWS Lambda.

> **Note**
>
> Questi comandi richiedono l'accesso alle credenziali. AWS

Utilizzo

```
$ sam remote callback <subcommand> <options>
```

Opzioni

`--help, -h`

Mostra questo messaggio ed esci.

Sottocomandi

`succeed`

Invia una callback riuscita a un'esecuzione remota di una funzione duratura. Per ulteriori informazioni, consulta sam remote callback succeed.

`fail`

Invia un errore di callback a un'esecuzione remota di una funzione duratura. Per ulteriori informazioni, consulta sam remote callback fail.

`heartbeat`

Invia un battito cardiaco di callback a un'esecuzione remota di una funzione duratura. Per ulteriori informazioni, consulta sam remote callback heartbeat.

sam remote callback succeed

Invia una callback riuscita a un'esecuzione remota di una funzione duratura.

> **Note**
>
> Questo comando richiede l'accesso alle AWS credenziali.

Utilizzo

```
$ sam remote callback succeed CALLBACK_ID [OPTIONS]
```

Argomenti richiesti

`CALLBACK_ID`

L'identificatore univoco del callback a cui inviare una risposta riuscita.

Opzioni

`-r, --result TEXT`

Risultato di successo: payload come stringa.

`--region TEXT`

Imposta la AWS regione del servizio (ad esempio,`us-east-1`).

`--profile TEXT`

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--config-env TEXT`

Nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione. Impostazione predefinita: `default`

`--config-file TEXT`

File di configurazione contenente i valori dei parametri predefiniti. Impostazione predefinita: `samconfig.toml`

`--save-params`

Salva i parametri forniti tramite la riga di comando nel file di configurazione.

`--beta-features / --no-beta-features`

Abilita/disabilita le funzionalità beta.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dalla CLI e visualizzare i timestamp AWS SAM .

`-h, --help`

Mostra questo messaggio ed esce.

Esempi

Invia una richiamata riuscita senza risultati:

```
$ sam remote callback succeed my-callback-id
```

Invia una richiamata riuscita con risultato:

```
$ sam remote callback succeed my-callback-id --result 'Task completed successfully'
```

sam remote callback fail

Invia un errore di callback a un'esecuzione remota di una funzione duratura.

> **Note**
> Questo comando richiede l'accesso alle AWS credenziali.

Utilizzo

```
$ sam remote callback fail CALLBACK_ID [OPTIONS]
```

Argomenti richiesti

`CALLBACK_ID`

L'identificatore univoco del callback a cui inviare una risposta di errore.

Opzioni

`--error-data TEXT`

Dati di errore aggiuntivi.

`--stack-trace TEXT`

Immetti le voci di tracciamento dello stack. Questa opzione può essere specificata più volte.

`--error-type TEXT`

Tipo di errore.

`--error-message TEXT`

Messaggio di errore dettagliato.

`--region TEXT`

Imposta la AWS regione del servizio (ad esempio,`us-east-1`).

`--profile TEXT`

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--config-env TEXT`

Nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione. Impostazione predefinita: `default`

`--config-file TEXT`

File di configurazione contenente i valori dei parametri predefiniti. Impostazione predefinita: `samconfig.toml`

`--save-params`

Salva i parametri forniti tramite la riga di comando nel file di configurazione.

`--beta-features / --no-beta-features`

Abilita/disabilita le funzionalità beta.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dalla CLI e visualizzare i timestamp AWS SAM .

`-h, --help`

Mostra questo messaggio ed esce.

Esempi

Invia una richiamata di errore senza parametri:

```
$ sam remote callback fail my-callback-id
```

Invia una richiamata di errore con messaggio di errore:

```
$ sam remote callback fail my-callback-id --error-message 'Task failed'
```

Invia una richiamata di errore con tutti i parametri:

```
$ sam remote callback fail my-callback-id --error-message 'Task failed' --error-type
 'ValidationError' --stack-trace 'at line 42' --error-data '{"code": 500}'
```

sam remote callback heartbeat

Invia un battito cardiaco di callback a un'esecuzione remota di una funzione duratura.

> **Note**
>
> Questo comando richiede l'accesso alle credenziali. AWS

Utilizzo

```
$ sam remote callback heartbeat CALLBACK_ID [OPTIONS]
```

Argomenti richiesti

`CALLBACK_ID`

L'identificatore univoco del callback a cui inviare un heartbeat.

Opzioni

`--region TEXT`

Imposta la AWS regione del servizio (ad esempio,`us-east-1`).

`--profile TEXT`

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--config-env TEXT`

Nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione. Impostazione predefinita: `default`

`--config-file TEXT`

File di configurazione contenente i valori dei parametri predefiniti. Impostazione predefinita: `samconfig.toml`

`--save-params`

Salva i parametri forniti tramite la riga di comando nel file di configurazione.

`--beta-features / --no-beta-features`

Abilita/disabilita le funzionalità beta.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dalla CLI e visualizzare i timestamp AWS SAM .

`-h, --help`

Mostra questo messaggio ed esce.

Esempi

Invia una richiamata del battito cardiaco:

```
$ sam remote callback heartbeat my-callback-id
```

sam remote execution

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam remote execution`.

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam remote execution` comando consente di gestire e ispezionare le esecuzioni di funzioni durevoli remote in esecuzione in AWS Lambda.

> **Note**
>
> Questi comandi richiedono l'accesso alle credenziali. AWS

Utilizzo

```
$ sam remote execution <subcommand> <options>
```

Opzioni

`--help, -h`

Mostra questo messaggio ed esci.

Sottocomandi

`get`

Ottieni i dettagli di un'esecuzione duratura. Per ulteriori informazioni, consulta sam remote execution get.

`history`

Ottieni la cronologia di esecuzione di un'esecuzione duratura di una funzione. Per ulteriori informazioni, consulta sam remote execution history.

`stop`

Interrompe l'esecuzione duratura di una funzione. Per ulteriori informazioni, consulta sam remote execution stop.

sam remote execution get

Recupera i dettagli dell'esecuzione durevole di una funzione specifica da AWS Lambda.

L'ARN di esecuzione deve essere nel formato restituito da Invoke o trovato nei log di esecuzione.

> **Note**
> Questo comando richiede l'accesso alle credenziali. AWS

Utilizzo

```
$ sam remote execution get DURABLE_EXECUTION_ARN [OPTIONS]
```

Argomenti richiesti

`DURABLE_EXECUTION_ARN`

L'Amazon Resource Name (ARN) dell'esecuzione durevole della funzione di cui recuperare i dettagli.

Opzioni

`--format [summary|json]`

Formato di output. Impostazione predefinita: `summary`

`--region TEXT`

Imposta la AWS regione del servizio (ad esempio,`us-east-1`).

`--profile TEXT`

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--config-env TEXT`

Nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione. Impostazione predefinita: `default`

`--config-file TEXT`

File di configurazione contenente i valori dei parametri predefiniti. Impostazione predefinita: `samconfig.toml`

`--save-params`

Salva i parametri forniti tramite la riga di comando nel file di configurazione.

`--beta-features / --no-beta-features`

Abilita/disabilita le funzionalità beta.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dalla CLI e visualizzare i timestamp AWS SAM .

`-h, --help`

Mostra questo messaggio ed esce.

Esempi

Ottieni i dettagli dell'esecuzione con il formato di riepilogo predefinito:

```
$ sam remote execution get 'arn:aws:lambda:us-east-1:123456789012:function:my-function:
$LATEST/durable-execution/c63eec67-3415-4eb4-a495-116aa3a86278/1d454231-a3ad-3694-aa03-
c917c175db55'
```

Ottieni i dettagli di esecuzione in formato JSON:

```
$ sam remote execution get 'arn:aws:lambda:us-east-1:123456789012:function:my-function:
$LATEST/durable-execution/c63eec67-3415-4eb4-a495-116aa3a86278/1d454231-a3ad-3694-aa03-
c917c175db55' --format json
```

> **Note**
> È necessario assicurarsi che i caratteri di controllo simili all'ARN di esecuzione vengano espulsi correttamente quando si utilizzano $ i comandi della shell.

sam remote execution history

Recupera la cronologia di esecuzione di una funzione duratura specifica da AWS Lambda.

L'ARN di esecuzione deve essere nel formato restituito StartDurableExecution o trovato nei log di esecuzione.

> **Note**
> Questo comando richiede l'accesso alle credenziali. AWS

Utilizzo

```
$ sam remote execution history DURABLE_EXECUTION_ARN [OPTIONS]
```

Argomenti richiesti

`DURABLE_EXECUTION_ARN`

L'Amazon Resource Name (ARN) dell'esecuzione durevole della funzione di cui recuperare la cronologia.

Opzioni

`--format [table|json]`

Formato di output. Impostazione predefinita: `table`

`--region TEXT`

Imposta la AWS regione del servizio (ad esempio,`us-east-1`).

`--profile TEXT`

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--config-env TEXT`

Nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione. Impostazione predefinita: `default`

`--config-file TEXT`

File di configurazione contenente i valori dei parametri predefiniti. Impostazione predefinita: `samconfig.toml`

`--save-params`

Salva i parametri forniti tramite la riga di comando nel file di configurazione.

`--beta-features / --no-beta-features`

Abilita/disabilita le funzionalità beta.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dalla CLI e visualizzare i timestamp AWS SAM .

`-h, --help`

Mostra questo messaggio ed esce.

Esempi

Ottieni la cronologia di esecuzione con il formato di tabella predefinito:

```
$ sam remote execution history arn:aws:lambda:us-
east-1:123456789012:function:MyFunction:$LATEST/durable-execution/my-execution-name/my-
execution-id
```

Ottieni la cronologia di esecuzione in formato JSON:

```
$ sam remote execution history arn:aws:lambda:us-
east-1:123456789012:function:MyFunction:$LATEST/durable-execution/my-execution-name/my-
execution-id --format json
```

sam remote execution stop

Interrompi l'esecuzione di una funzione durevole in esecuzione in AWS Lambda.

L'ARN di esecuzione deve essere nel formato restituito StartDurableExecution o trovato nei log di esecuzione.

> **Note**
>
> Questo comando richiede l'accesso alle credenziali. AWS

Utilizzo

```
$ sam remote execution stop DURABLE_EXECUTION_ARN [OPTIONS]
```

Argomenti richiesti

`DURABLE_EXECUTION_ARN`

L'Amazon Resource Name (ARN) dell'esecuzione durevole della funzione da interrompere.

Opzioni

`--error-message TEXT`

Messaggio di errore da associare all'esecuzione interrotta.

`--error-type TEXT`

Tipo di errore da associare all'esecuzione interrotta.

`--error-data TEXT`

Dati di errore da associare all'esecuzione interrotta.

`--stack-trace TEXT`

Immissioni di stack trace. Questa opzione può essere specificata più volte.

`--region TEXT`

Imposta la AWS regione del servizio (ad esempio,`us-east-1`).

`--profile TEXT`

Seleziona un profilo specifico dal tuo file di credenziali per ottenere le AWS credenziali.

`--config-env TEXT`

Nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione. Impostazione predefinita: `default`

`--config-file TEXT`

File di configurazione contenente i valori dei parametri predefiniti. Impostazione predefinita: `samconfig.toml`

`--save-params`

Salva i parametri forniti tramite la riga di comando nel file di configurazione.

`--beta-features / --no-beta-features`

Abilita/disabilita le funzionalità beta.

`--debug`

Attiva la registrazione di debug per stampare i messaggi di debug generati dalla CLI e visualizzare i timestamp AWS SAM .

`-h, --help`

Mostra questo messaggio ed esce.

Esempi

Interrompi l'esecuzione senza dettagli sull'errore:

```
$ sam remote execution stop arn:aws:lambda:us-east-1:123456789012:function:MyFunction:
$LATEST/durable-execution/my-execution-name/my-execution-id
```

Interrompi l'esecuzione con un messaggio di errore e digita:

```
$ sam remote execution stop arn:aws:lambda:us-east-1:123456789012:function:MyFunction:
$LATEST/durable-execution/my-execution-name/my-execution-id --error-message "Execution
 cancelled" --error-type "UserCancellation"
```

Interrompi l'esecuzione con i dettagli completi dell'errore e la traccia dello stack:

```
$ sam remote execution stop arn:aws:lambda:us-east-1:123456789012:function:MyFunction:
$LATEST/durable-execution/my-execution-name/my-execution-id --error-message "Task
```

```
failed" --error-type "TaskFailure" --error-data '{"reason":"timeout"}' --stack-trace
"at function1()" --stack-trace "at function2()"
```

sam remote invoke

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam remote invoke`.

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'uso del AWS SAMCLI `sam remote invoke` comando, vedereIntroduzione ai test nel cloud con sam remote invoke.

Il `sam remote invoke` comando richiama le risorse supportate in. Cloud AWS

Utilizzo

```
$ sam remote invoke <arguments> <options>
```

Arguments (Argomenti)

Resource ID (ID risorsa)

L'ID della risorsa supportata da richiamare.

Questo argomento accetta i seguenti valori:

- Amazon Resource Name (ARN): l'ARN della risorsa.

> **Tip**
>
> `sam list stack-outputs --stack-name <stack-name>`Utilizzalo per ottenere l'ARN delle tue risorse.

- ID logico: l'ID logico della risorsa. È inoltre necessario fornire il nome AWS CloudFormation dello stack utilizzando l'`--stack-name`opzione.
- ID fisico: l'ID fisico della risorsa. Questo ID viene creato quando si distribuisce una risorsa utilizzando CloudFormation.

> **Tip**
>
> `sam list resources --stack-name <stack-name>`Utilizzalo per ottenere l'ID fisico delle tue risorse.

Quando fornisci un ARN o un ID fisico:

Se fornisci un ARN o un ID fisico, non fornire un nome dello stack. Quando il nome dello stack viene fornito utilizzando l'`--stack-name`opzione o quando il nome dello stack è definito nel file di configurazione, AWS SAM CLI elaborerà automaticamente l'ID della risorsa come valore ID logico dello stack. CloudFormation

Quando non fornisci un ID di risorsa:

Se non fornisci un ID di risorsa, ma fornisci un nome di stack con `--stack-name` l'opzione, la AWS SAM CLI tenterà di richiamare automaticamente una risorsa nello stack utilizzando CloudFormation la seguente logica:

1. AWS SAM CLIIdentificherà i tipi di risorse nell'ordine seguente e passerà alla fase successiva una volta trovato il tipo di risorsa nello stack:
   a. Lambda
   b. Step Functions
   c. Amazon SQS
   d. Flussi di dati Kinesis
2. Se il tipo di risorsa ha una sola risorsa nello stack, la AWS SAM CLI invocherà. Se nello stack sono presenti più risorse dello stesso tipo di risorsa, AWS SAM CLI verrà restituito un errore.

Di seguito sono riportati alcuni esempi di ciò che AWS SAM CLI farà:

- Stack che contiene due funzioni Lambda e una coda Amazon SQS: AWS SAM CLI individuerà il tipo di risorsa Lambda e restituirà un errore poiché lo stack contiene più di una funzione Lambda.
- Stack che contiene una funzione Lambda e due applicazioni Amazon Kinesis Data Streams AWS SAM CLI: individuerà la funzione Lambda e la richiamerà poiché lo stack contiene una singola risorsa Lambda.

- Stack che contiene una singola coda Amazon SQS e due applicazioni Kinesis Data Streams: AWS SAM CLI individuerà la coda Amazon SQS e la richiamerà poiché lo stack contiene un'unica coda Amazon SQS.

Opzioni

`--beta-features | --no-beta-features`

Consenti o nega le funzionalità beta.

`--config-env` *`TEXT`*

Specificate l'ambiente da utilizzare dal file AWS SAMCLI di configurazione.

Default: `default`

`--config-file` *`FILENAME`*

Specificate il percorso e il nome del file di configurazione.

Per ulteriori informazioni sui file di configurazione, consulta Configurazione del AWS SAMCLI.

Impostazione predefinita: `samconfig.toml` nella radice della directory del progetto.

`--debug`

Attiva la registrazione di debug. Questo stampa i messaggi di debug e i timestamp generati da. AWS SAMCLI

`--event, -e` *`TEXT`*

L'evento da inviare alla risorsa di destinazione.

`--event-file` *`FILENAME`*

Il percorso di un file che contiene l'evento da inviare alla risorsa di destinazione.

`--help, -h`

Mostra il messaggio di aiuto ed esci.

`--output` *`[ text | json ]`*

Visualizza i risultati della chiamata in un formato di output specifico.

`json`— I metadati della richiesta e la risposta delle risorse vengono restituiti nella struttura JSON. La risposta contiene l'output SDK completo.

`text`— I metadati della richiesta vengono restituiti nella struttura del testo. La risposta della risorsa viene restituita nel formato di output della risorsa richiamata.

`--parameter`

Boto3Parametri aggiuntivi che è possibile passare alla risorsa richiamata.

Flusso di dati Amazon Kinesis

I seguenti parametri aggiuntivi possono essere utilizzati per inserire un record nel flusso di dati Kinesis:

- `ExplicitHashKey='string'`
- `PartitionKey='string'`
- `SequenceNumberForOrdering='string'`
- `StreamARN='string'`

Per una descrizione di ogni parametro, vedi Kinesis.client.put_RECORD.

AWS Lambda

I seguenti parametri aggiuntivi possono essere utilizzati per richiamare una risorsa Lambda e ricevere una risposta bufferizzata:

- `ClientContext='base64-encoded string'`
- `InvocationType='[ DryRun | Event | RequestResponse ]'`
- `LogType='[ None | Tail ]'`
- `Qualifier='string'`

I seguenti parametri aggiuntivi possono essere utilizzati per richiamare una risorsa Lambda con streaming di risposte:

- `ClientContext='base64-encoded string'`
- `InvocationType='[ DryRun | RequestResponse ]'`
- `LogType='[ None | Tail ]'`
- `Qualifier='string'`

Per una descrizione di ogni parametro, consulta quanto segue:

- Lambda con risposta bufferizzata — lambda.client.Invoke
- Lambda con streaming di risposte — lambda.client.invoke_with_response_stream

Amazon Simple Queue Service (Amazon SQS)

I seguenti parametri aggiuntivi possono essere utilizzati per inviare un messaggio a una coda Amazon SQS:

- `DelaySeconds=integer`
- `MessageAttributes='json string'`
- `MessageDeduplicationId='string'`
- `MessageGroupId='string'`
- `MessageSystemAttributes='json string'`

Per una descrizione di ogni parametro, consulta sqs.client.send_message.

AWS Step Functions

I seguenti parametri aggiuntivi possono essere utilizzati per avviare l'esecuzione di una macchina a stati:

- `name='string'`
- `traceHeader='string'`

Per una descrizione di ogni parametro, vedere sfn.client.start_execution.

`--profile TEXT`

Il profilo specifico del file di credenziali per ottenere le credenziali. AWS

`--region TEXT`

La Regione AWS risorsa. Ad esempio, `us-east-1`.

`--stack-name TEXT`

Il nome dello CloudFormation stack a cui appartiene la risorsa.

`--tenant-id TEXT`

L'ID tenant per le funzioni Lambda multi-tenant. Utilizzato per garantire l'isolamento del calcolo tra diversi tenant. Richiesto quando si richiamano funzioni configurate con la modalità di isolamento dei tenant.

`--test-event-name` *NAME*

Il nome dell'evento di test condivisibile da passare alla funzione Lambda.

> **Note**
>
> Questa opzione supporta solo le funzioni Lambda.

Esempio

L'esempio seguente richiama le risorse supportate nel AWS Cloud e attiva la registrazione di debug, che stampa i messaggi di debug e i timestamp generati da: AWS SAMCLI

```
$ sam remote invoke--debug
```

sam remote test-event

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam remote test-event`.

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'uso del AWS SAMCLI `sam remote test-event` comando, vedereIntroduzione al cloud testing con sam remote test-event.

Il `sam remote test-event` comando interagisce con eventi di test condivisibili nel registro degli EventBridge schemi di Amazon.

Utilizzo

```
$ sam remote test-event <options> <subcommand>
```

Opzioni

`--help, -h`

Mostra il messaggio di aiuto ed esci.

Sottocomandi

`delete`

Elimina un evento di test condivisibile dal registro dello EventBridge schema. Per ulteriori informazioni di riferimento, vedere sam remote test-event delete.

`get`

Ottieni un evento di test condivisibile dal registro degli EventBridge schemi. Per ulteriori informazioni di riferimento, vedere sam remote test-event get.

`list`

Elenca gli eventi di test condivisibili esistenti per una AWS Lambda funzione. Per ulteriori informazioni di riferimento, vedere sam remote test-event list.

`put`

Salvare un evento da un file locale nel registro dello EventBridge schema. Per ulteriori informazioni di riferimento, vedere sam remote test-event put.

sam remote test-event delete

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model `sam remote test-event delete` sottocomando Command Line Interface (AWS SAMCLI).

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'uso del AWS SAMCLI `sam remote test-event` comando, vedere Introduzione al cloud testing con sam remote test-event.

Il `sam remote test-event delete` sottocomando elimina un evento di test condivisibile dal registro degli schemi di Amazon EventBridge .

Utilizzo

```
$ sam remote test-event delete <arguments> <options>
```

Arguments (Argomenti)

Resource ID (ID risorsa)

L'ID della AWS Lambda funzione associata all'evento di test condivisibile.

Se si fornisce un ID logico, è necessario fornire anche un valore per lo AWS CloudFormation stack associato alla funzione Lambda utilizzando `--stack-name` l'opzione.

Valori validi: l'ID o la risorsa logico della risorsa. ARN

Opzioni

`--config-env` *TEXT*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *PATH*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è "`samconfig.toml`" nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--help, -h`

Mostra il messaggio di aiuto ed esci.

`--name` *TEXT*

Il nome dell'evento di test condivisibile da eliminare.

`--stack-name` *TEXT*

Il nome dello CloudFormation stack associato alla funzione Lambda.

Questa opzione è obbligatoria se si fornisce l'ID logico della funzione Lambda come argomento.

sam remote test-event get

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model `sam remote test-event get` sottocomando Command Line Interface (AWS SAMCLI).

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'uso del AWS SAMCLI `sam remote test-event` comando, vedereIntroduzione al cloud testing con sam remote test-event.

Il `sam remote test-event get` sottocomando ottiene un evento di test condivisibile dal registro degli EventBridge schemi di Amazon.

Utilizzo

```
$ sam remote test-event get <arguments> <options>
```

Arguments (Argomenti)

Resource ID (ID risorsa)

L'ID della AWS Lambda funzione associata all'evento di test condivisibile da ottenere.

Se si fornisce un ID logico, è necessario fornire anche un valore per lo AWS CloudFormation stack associato alla funzione Lambda utilizzando `--stack-name` l'opzione.

Valori validi: l'ID o la risorsa logico della risorsa. ARN

Opzioni

`--config-env` *TEXT*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *PATH*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è "`samconfig.toml`" nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--help, -h`

Mostra il messaggio di aiuto ed esci.

`--name` *TEXT*

Il nome dell'evento di test condivisibile da ottenere.

`--output-file` *`FILENAME`*

Il percorso e il nome del file in cui salvare l'evento sul computer locale.

Se non fornisci questa opzione, AWS SAM CLI invierà il contenuto dell'evento di test condivisibile sulla tua console.

`--stack-name` *`TEXT`*

Il nome dello CloudFormation stack associato alla funzione Lambda.

Questa opzione è obbligatoria se si fornisce l'ID logico della funzione Lambda come argomento.

sam remote test-event list

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model `sam remote test-event list` sottocomando Command Line Interface (AWS SAMCLI).

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'uso del AWS SAMCLI `sam remote test-event` comando, vedereIntroduzione al cloud testing con sam remote test-event.

Il `sam remote test-event list` sottocomando elenca gli eventi di test condivisibili esistenti per una AWS Lambda funzione specifica dal registro degli EventBridge schemi di Amazon.

Utilizzo

```
$ sam remote test-event list <arguments> <options>
```

Arguments (Argomenti)

Resource ID (ID risorsa)

L'ID della funzione Lambda associata agli eventi di test condivisibili.

Se si fornisce un ID logico, è necessario fornire anche un valore per lo AWS CloudFormation stack associato alla funzione Lambda utilizzando `--stack-name` l'opzione.

Valori validi: l'ID o la risorsa logico della risorsa. ARN

Opzioni

`--config-env` *TEXT*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *PATH*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è "`samconfig.toml`" nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--help, -h`

Mostra il messaggio di aiuto ed esci.

`--stack-name` *TEXT*

Il nome dello CloudFormation stack associato alla funzione Lambda.

Questa opzione è obbligatoria se si fornisce l'ID logico della funzione Lambda come argomento.

Esempi

Per esempi sull'utilizzo di questo comando, fare riferimento aElenco degli eventi di test condivisibili.

sam remote test-event put

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model `sam remote test-event put` sottocomando Command Line Interface (AWS SAMCLI).

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'uso del AWS SAMCLI `sam remote test-event` comando, vedereIntroduzione al cloud testing con sam remote test-event.

Il `sam remote test-event put` sottocomando salva un evento di test condivisibile dal computer locale nel registro degli EventBridge schemi Amazon.

Utilizzo

```
$ sam remote test-event put <arguments> <options>
```

Arguments (Argomenti)

Resource ID (ID risorsa)

L'ID della AWS Lambda funzione associata all'evento di test condivisibile.

Se si fornisce un ID logico, è necessario fornire anche un valore per lo AWS CloudFormation stack associato alla funzione Lambda utilizzando `--stack-name` l'opzione.

Valori validi: l'ID o la risorsa logico della risorsa. ARN

Opzioni

`--config-env` *TEXT*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *PATH*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è "`samconfig.toml`" nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--file` *FILENAME*

Il percorso del file e il nome dell'evento sul computer locale.

Fornisci `-` come nome del file il valore da cui leggere`stdin`.

Questa opzione è obbligatoria.

`--force`, `-f`

Sovrascrivi un evento di test condivisibile con lo stesso nome.

`--help, -h`

Mostra il messaggio di aiuto ed esci.

`--name TEXT`

Il nome con cui salvare l'evento di test condivisibile.

Se nel registro dello EventBridge schema esiste un evento di test condivisibile con lo stesso nome, non lo AWS SAM CLI sovrascriverà. Per sovrascrivere, aggiungi l'opzione. `--force`

`--output-file FILENAME`

Il percorso e il nome del file in cui salvare l'evento sul computer locale.

Se non fornisci questa opzione, AWS SAM CLI invierà il contenuto dell'evento di test condivisibile sulla tua console.

`--stack-name TEXT`

Il nome dello CloudFormation stack associato alla funzione Lambda.

Questa opzione è obbligatoria se si fornisce l'ID logico della funzione Lambda come argomento.

Esempio

Per un esempio sull'utilizzo di questo comando, fare riferimento aSalvataggio di eventi di test condivisibili.

sam sync

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam sync`.

- Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI
- Per la documentazione sull'utilizzo di AWS SAMCLI, vedereAWS SAM CLI.

Il `sam sync` comando sincronizza le modifiche dell'applicazione locale con. Cloud AWS

Utilizzo

```
$ sam sync <options>
```

Opzioni

`--base-dir, -s DIRECTORY`

Risolve i percorsi relativi al codice sorgente della funzione o del livello rispetto a questa directory. Utilizzate questa opzione per modificare il modo in cui vengono risolti i percorsi relativi alle cartelle del codice sorgente. Per impostazione predefinita, i percorsi relativi vengono risolti rispetto alla posizione del AWS SAM modello.

Oltre alle risorse nell'applicazione root o nello stack che stai creando, questa opzione si applica anche alle applicazioni o agli stack annidati. Inoltre, questa opzione si applica ai seguenti tipi e proprietà di risorse:

- Tipo di risorsa: `AWS::Serverless::Function` Proprietà: `CodeUri`
- Tipo di risorsa: Attributo della `AWS::Serverless::Function` risorsa: `Metadata` Voce: `DockerContext`
- Tipo di risorsa: `AWS::Serverless::LayerVersion` Proprietà: `ContentUri`
- Tipo di risorsa: `AWS::Lambda::Function` Proprietà: `Code`
- Tipo di risorsa: `AWS::Lambda::LayerVersion` Proprietà: `Content`

`--build-image TEXT`

L'URI per l'immagine del contenitore che desideri utilizzare durante la creazione dell'applicazione. Per impostazione predefinita, AWS SAM utilizza l'URI del repository di immagini del contenitore di Amazon Elastic Container Registry (Amazon ECR) Public. Specificate questa opzione per utilizzare un'immagine diversa.

È possibile utilizzare questa opzione più volte in un unico comando. Ogni opzione accetta una stringa o una coppia chiave-valore.

- Stringa: specifica l'URI dell'immagine del contenitore che verranno utilizzate da tutte le risorse dell'applicazione. Di seguito è riportato un esempio:

```
$ sam sync --build-image amazon/aws-sam-cli-build-image-python3.8
```

- Coppia chiave-valore: specifica il nome della risorsa come chiave e l'URI dell'immagine del contenitore da utilizzare con quella risorsa come valore. Utilizzate questo formato per specificare un URI di immagine del contenitore diverso per ogni risorsa dell'applicazione. Di seguito è riportato un esempio:

```
$ sam sync --build-image Function1=amazon/aws-sam-cli-build-image-python3.8
```

Questa opzione si applica solo se l'`--use-container`opzione è specificata, altrimenti si verificherà un errore.

`--build-in-source | --no-build-in-source`

Fornisce `--build-in-source` la possibilità di creare il progetto direttamente nella cartella dei sorgenti.

L'`--build-in-source`opzione supporta i seguenti runtime e metodi di compilazione:

- Runtime: qualsiasi Node.js runtime supportato dall'`sam init --runtime`opzione.
- Metodi di compilazione:`Makefile`,`esbuild`.

L'`--build-in-source`opzione non è compatibile con le seguenti opzioni:

- `--use-container`

Default: `--no-build-in-source`

`--capabilities LIST`

Un elenco di funzionalità specificate per consentire la creazione CloudFormation di determinati stack. Alcuni modelli di stack potrebbero includere risorse che possono influire sulle autorizzazioni del tuo. Account AWS Ad esempio, creando nuovi utenti AWS Identity and Access Management (IAM). Specificate questa opzione per sovrascrivere i valori predefiniti. I valori validi includono quanto segue:

- CAPABILITY_IAM
- CAPABILITY_NAMED_IAM
- POLITICA DELLE RISORSE IN MATERIA DI CAPACITÀ
- CAPABILITY_AUTO_EXPAND

Predefinito: e `CAPABILITY_NAMED_IAM` `CAPABILITY_AUTO_EXPAND`

`--code`

Per impostazione predefinita, AWS SAM sincronizza tutte le risorse dell'applicazione. Specificate questa opzione per sincronizzare solo le risorse di codice, che includono quanto segue:

- `AWS::Serverless::Function`
- `AWS::Lambda::Function`
- `AWS::Serverless::LayerVersion`
- `AWS::Lambda::LayerVersion`
- `AWS::Serverless::Api`
- `AWS::ApiGateway::RestApi`
- `AWS::Serverless::HttpApi`
- `AWS::ApiGatewayV2::Api`
- `AWS::Serverless::StateMachine`
- `AWS::StepFunctions::StateMachine`

Per sincronizzare le risorse di codice, AWS SAM utilizza APIs direttamente il AWS servizio, anziché distribuirlo tramite AWS CloudFormation. Per aggiornare il tuo CloudFormation stack, sam sync --watch esegui o. sam deploy

`--config-env` *`TEXT`*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *`PATH`*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è "`samconfig.toml`" nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--dependency-layer | --no-dependency-layer`

Speciifica se separare le dipendenze delle singole funzioni in un altro livello per accelerare il processo di sincronizzazione.

Default: `--dependency-layer`

`--image-repository` *`TEXT`*

Il nome del repository Amazon Elastic Container Registry (Amazon ECR) in cui questo comando carica l'immagine della funzione. Obbligatorio per le funzioni dichiarate con il `Image` tipo di pacchetto.

`--image-repositories` *TEXT*

Una mappatura delle funzioni sull'URI del loro repository Amazon ECR. Funzioni di riferimento in base al relativo ID logico. Di seguito è riportato un esempio:

```
$ sam sync --image-repositories Function1=123456789012.dkr.ecr.us-
east-1.amazonaws.com/my-repo
```

È possibile specificare questa opzione più volte in un unico comando.

`--kms-key-id` *TEXT*

L'ID di una chiave AWS Key Management Service (AWS KMS) utilizzata per crittografare gli artefatti che sono a riposo nel bucket Amazon S3. Se non specifichi questa opzione, AWS SAM utilizza le chiavi di crittografia gestite da Amazon S3.

`--metadata`

Una mappa di metadati da allegare a tutti gli artefatti a cui fai riferimento nel modello.

`--notification-arns` *LIST*

Un elenco di ARNs argomenti relativi a Amazon Simple Notification Service (Amazon SNS) associati allo CloudFormation stack.

`--no-use-container`

Un'opzione che consente di utilizzare il toolkit IDE per impostare il comportamento predefinito.

`--parameter-overrides`

Una stringa che contiene le sostituzioni CloudFormation dei parametri codificate come coppie chiave-valore. Utilizzate lo stesso formato di (). AWS Command Line Interface AWS CLI Il AWS SAMCLI formato è costituito da parole chiave e valori esplicite, ogni override è separata da uno spazio. Di seguito, sono riportati due esempi:

- `--parameter-overrides ParameterKey=hello,ParameterValue=world`
- `--parameter-overrides ParameterKey=hello,ParameterValue=world ParameterKey=example1,ParameterValue=example2 ParameterKey=apple,ParameterValue=banana`

`--resource` *TEXT*

Speciifica il tipo di risorsa da sincronizzare. Per sincronizzare più risorse, è possibile specificare questa opzione più volte. Questa opzione è supportata dall'`--code`opzione.

Il valore deve essere una delle risorse elencate sotto`--code`. Ad esempio, `--resource AWS::Serverless::Function --resource AWS::Serverless::LayerVersion`.

`--resource-id` *TEXT*

Speciifica l'ID della risorsa da sincronizzare. Per sincronizzare più risorse, è possibile specificare questa opzione più volte. Questa opzione è supportata dall'`--code`opzione. Ad esempio, `--resource-id Function1 --resource-id Function2`.

`--role-arn` *TEXT*

L'Amazon Resource Name (ARN) di un ruolo IAM che CloudFormation assume quando si applica il changeset.

`--s3-bucket` *TEXT*

Il nome del bucket Amazon Simple Storage Service (Amazon S3) in cui questo comando carica il modello. CloudFormation Se il modello è più grande di 51.200 byte, è richiesta l'opzione o l'`--s3-bucket`opzione. `--resolve-s3` Se si specificano entrambe le `--resolve-s3` opzioni `--s3-bucket` e, si verifica un errore.

`--s3-prefix` *TEXT*

Il prefisso aggiunto ai nomi degli artefatti caricati nel bucket Amazon S3. Il nome del prefisso è un nome di percorso (nome della cartella) per il bucket Amazon S3. Questo vale solo per le funzioni dichiarate con il `Zip` tipo di pacchetto.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file AWS SAM di configurazione.

`--skip-deploy-sync | --no-skip-deploy-sync`

Specifica `--skip-deploy-sync` di saltare la sincronizzazione iniziale dell'infrastruttura se non è richiesta. AWS SAMCLIConfronterà il AWS SAM modello locale con il CloudFormation modello distribuito ed eseguirà una distribuzione solo se viene rilevata una modifica.

Specifica `--no-skip-deploy-sync` di eseguire una CloudFormation distribuzione ogni volta che `sam sync` viene eseguita.

Per ulteriori informazioni, consulta Salta la distribuzione iniziale CloudFormation.

Default: `--skip-deploy-sync`

`--stack-name TEXT`

Il nome dello CloudFormation stack per l'applicazione.

Questa opzione è obbligatoria.

`--tags LIST`

Un elenco di tag da associare allo stack creato o aggiornato. CloudFormation propaga inoltre questi tag alle risorse dello stack che li supportano.

`--template-file, --template, -t PATH`

Il percorso e il nome del file in cui si trova il AWS SAM modello.

> **Note**
>
> Se si specifica questa opzione, AWS SAM distribuisce solo il modello e le risorse locali a cui punta.

`--use-container, -u`

Se le tue funzioni dipendono da pacchetti che hanno dipendenze compilate in modo nativo, usa questa opzione per creare la tua funzione all'interno di un AWS Lambda contenitore simile a -like. Docker

> **Note**
>
> Attualmente, questa opzione non è compatibile con. `--dependency-layer` Se usi `--use-container` con`--dependency-layer`, ti AWS SAMCLI informa e continua con`--no-dependency-layer`.

`--watch`

Avvia un processo che controlla le modifiche apportate all'applicazione locale e le sincronizza automaticamente con. Cloud AWS Per impostazione predefinita, quando si specifica questa opzione, AWS SAM sincronizza tutte le risorse dell'applicazione durante l'aggiornamento. Con questa opzione, AWS SAM esegue una CloudFormation distribuzione iniziale. Quindi, AWS SAM utilizza il AWS servizio APIs per aggiornare le risorse del codice. AWS SAM utilizza

CloudFormation per aggiornare le risorse dell'infrastruttura quando si aggiorna il AWS SAM modello.

`--watch-exclude` *`TEXT`*

Esclude un file o una cartella dall'osservazione delle modifiche apportate ai file. Per utilizzare questa opzione, è `--watch` necessario fornire anche questa opzione.

Questa opzione riceve una coppia chiave-valore:

- Chiave: l'ID logico di una funzione Lambda nell'applicazione.
- Valore: il nome o la cartella del file associato da escludere.

Quando si aggiornano i file o le cartelle specificati con l'`--watch-exclude`opzione, AWS SAM CLI la sincronizzazione non viene avviata. Tuttavia, quando un aggiornamento ad altri file o cartelle avvia una sincronizzazione, tali file o cartelle verranno inclusi in tale sincronizzazione.

È possibile fornire questa opzione più volte in un unico comando.

Esempi

Per esempi sull'utilizzo di questo comando, fare riferimento aOpzioni per il comando sam sync.

sam traces

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam traces`.

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam traces` comando recupera AWS X-Ray le tracce presenti Account AWS nel Regione AWS file.

Utilizzo

```
$ sam traces <options>
```

Opzioni

`--config-env` *TEXT*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *PATH*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è "`samconfig.toml`" nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--end-time` *TEXT*

Recupera le tracce fino a questo momento. L'ora può essere costituita da valori relativi come «5 minuti fa», «domani» o un timestamp formattato come «2018-01-01 10:10:10».

`--output` *TEXT*

Specificate il formato di output per i log. Per stampare registri formattati, specificare. `text` Per stampare i log come JSON, specificare. `json`

`--save-params`

Salva i parametri forniti nella riga di comando nel AWS SAM file di configurazione.

`--start-time` *TEXT*

Recupera le tracce a partire da questo momento. L'ora può essere costituita da valori relativi come «5 minuti fa», «ieri» o un timestamp formattato come «2018-01-01 10:10:10». Il valore predefinito è «10 minuti fa».

`--tail`

Copia l'output della traccia. Questo ignora l'argomento dell'ora di fine e continua a visualizzare le tracce non appena diventano disponibili.

`--trace-id` *TEXT*

L'identificatore univoco per una traccia a raggi X.

Esempi

Esegui il seguente comando per recuperare le tracce X-Ray per ID.

```
$ sam traces --trace-id tracing-id-1 --trace-id tracing-id-2
```

Eseguite il seguente comando per seguire le tracce X-Ray non appena diventano disponibili.

```
$ sam traces --tail
```

sam validate

Questa pagina fornisce informazioni di riferimento per il AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI)`sam validate`.

Per un'introduzione a AWS SAMCLI, vedere Che cos'è il? AWS SAMCLI

Il `sam validate` comando verifica se un file AWS SAM modello è valido.

Utilizzo

```
$ sam validate <options>
```

Opzioni

`--config-env` *`TEXT`*

Il nome dell'ambiente che specifica i valori dei parametri predefiniti nel file di configurazione da utilizzare. Il valore predefinito è «default». Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--config-file` *`PATH`*

Il percorso e il nome del file di configurazione contenente i valori dei parametri predefiniti da utilizzare. Il valore predefinito è «samconfig.toml» nella radice della directory del progetto. Per ulteriori informazioni sui file di configurazione, consulta AWS SAMCLIfile di configurazione.

`--debug`

Attiva la registrazione di debug per stampare il messaggio di debug generato dai timestamp e visualizzarli. AWS SAMCLI

`--lint`

Esegui la convalida del linting sul modello tramite. cfn-lint Crea un file di `cfnlintrc` configurazione per specificare parametri aggiuntivi. Per ulteriori informazioni, vedere cfn-lint nel repository.CloudFormation GitHub

`--profile` *TEXT*

Il profilo specifico del file delle credenziali che ottiene le credenziali. AWS

`--region` *TEXT*

La AWS regione in cui effettuare la distribuzione. Ad esempio, us-east-1.

`--save-params`

Salva i parametri forniti dalla riga di comando nel file di AWS SAM configurazione.

`--template-file, --template, -t` *PATH*

Il file AWS SAM modello. Il valore predefinito è `template.[yaml|yml]`.

Se il modello si trova nella directory di lavoro corrente ed è denominato`template.[yaml|yml|json]`, questa opzione non è richiesta.

Se l'hai appena eseguitosam build, questa opzione non è richiesta.

Esempio

Per un esempio sull'utilizzo di questo comando per convalidare un modello, fare riferimento a. Convalida dei file AWS SAM modello

Per un esempio sull'utilizzo di questo comando con cfn-lint, fare riferimento a. Convalida le tue AWS SAM applicazioni con CloudFormation Linter

AWS SAMCLIgestione

Questa sezione contiene informazioni sui modi in cui è possibile gestire e personalizzare la versione di. AWS SAMCLI Ciò include informazioni su come configurare i valori dei parametri dei AWS SAMCLI comandi utilizzando un file di configurazione a livello di progetto. Include anche informazioni sulla gestione delle diverse versioni del file AWS SAMCLI, sull'impostazione delle AWS credenziali in modo da AWS SAM poter effettuare chiamate ai AWS servizi per conto dell'utente e su diversi

modi di personalizzazione. AWS SAM Questa sezione termina con una sezione sulla AWS SAM risoluzione dei problemi generali.

Argomenti

- AWS SAMCLIfile di configurazione
- Gestione delle AWS SAMCLI versioni
- Impostazione delle AWS credenziali
- Telemetria nel AWS SAMCLI
- AWS SAMCLIrisoluzione dei problemi

AWS SAMCLIfile di configurazione

La AWS Serverless Application Model Command Line Interface (AWS SAMCLI) supporta un file di configurazione a livello di progetto che è possibile utilizzare per configurare i valori dei parametri di AWS SAMCLI comando.

Per la documentazione sulla creazione e l'utilizzo dei file di configurazione, vedere. Configurazione del AWS SAMCLI

Argomenti

- Impostazioni predefinite del file di configurazione
- Formati di file di configurazione supportati
- Specificare un file di configurazione
- Nozioni di base sui file di configurazione
- Regole relative al valore dei parametri
- Precedenza di configurazione
- Creazione e modifica dei file di configurazione

Impostazioni predefinite del file di configurazione

AWS SAM utilizza le seguenti impostazioni predefinite del file di configurazione:

- Nome: `samconfig`.
- Posizione: alla base del progetto. Questa è la stessa posizione `template.yaml` del file.

- Formato:`TOML`. Per saperne di più, consulta TOML nella TOMLdocumentazione.

Di seguito è riportato un esempio di struttura di progetto che include il nome e la posizione del file di configurazione predefiniti:

```
sam-app
### README.md
### __init__.py
### events
### hello_world
### samconfig.toml
### template.yaml
### tests
```

Di seguito è riportato un esempio del file `samconfig.toml`.

```
...
version = 0.1

[default]
[default.global]
[default.global.parameters]
stack_name = "sam-app"

[default.build.parameters]
cached = true
parallel = true

[default.deploy.parameters]
capabilities = "CAPABILITY_IAM"
confirm_changeset = true
resolve_s3 = true

[default.sync.parameters]
watch = true

[default.local_start_api.parameters]
warm_containers = "EAGER"

[prod]
[prod.sync]
[prod.sync.parameters]
```

```
watch = false
```

Formati di file di configurazione supportati

`TOML`e i `[YAML|YML]` formati sono supportati. Vedi la seguente sintassi di base:

TOML

```
version = 0.1
[environment]
[environment.command]
[environment.command.parameters]
option = parameter value
```

YAML

```
version: 0.1
environment:
  command:
    parameters:
      option: parameter value
```

Specificare un file di configurazione

Per impostazione predefinita, AWS SAMCLI cerca un file di configurazione nell'ordine seguente:

1. File di configurazione personalizzato: se si utilizza l'`--config-file`opzione per specificare il nome e la posizione di un file, AWS SAMCLI cerca prima questo file.
2. **`samconfig.toml`**File predefinito: si tratta del nome e del formato del file di configurazione predefiniti, che si trova alla radice del progetto. Se non specifichi un file di configurazione personalizzato, AWS SAMCLI cerca successivamente questo file.
3. **`samconfig.[yaml|yml]`**file — Se `samconfig.toml` non esiste nella radice del progetto, AWS SAMCLI cerca questo file.

Di seguito è riportato un esempio di specificazione di un file di configurazione personalizzato utilizzando l'`--config-file`opzione:

```
$ sam deploy --config-file myconfig.yaml
```

> **Note**
>
> Il `--config-file` parametro deve essere relativo alla posizione del file AWS SAM modello perché AWS SAMCLI deve determinare il contesto in cui viene applicata la configurazione. Il `samconfig.toml` file gestisce le impostazioni di configurazione per la versione in AWS SAMCLI uso di e la CLI cerca il `samconfig.toml` file (o il parametro del file di configurazione sovrascritto) nella cartella relativa del file. `template.yaml`

Nozioni di base sui file di configurazione

Ambiente

Un ambiente è un identificatore denominato che contiene un set unico di impostazioni di configurazione. È possibile avere più ambienti in una singola AWS SAM applicazione.

Il nome dell'ambiente predefinito è`default`.

Utilizzate l' AWS SAMCLI`--config-env`opzione per specificare l'ambiente da utilizzare.

Comando

Il comando è il AWS SAMCLI comando per cui specificare i valori dei parametri.

Per specificare i valori dei parametri per tutti i comandi, utilizzate l'`global`identificatore.

Quando fate riferimento a un AWS SAMCLI comando, sostituite gli spazi ( ) e i trattini () con caratteri di sottolineatura (–). _ Fare riferimento agli esempi riportati di seguito:

- `build`
- `local_invoke`
- `local_start_api`

Parameters

I parametri sono specificati come coppie chiave-valore.

- La chiave è il nome dell'opzione AWS SAMCLI di comando.
- Il valore è il valore da specificare.

Quando specificate le chiavi, utilizzate l'opzione di comando in formato lungo name e sostituite i trattini (-) con caratteri di sottolineatura (). _ Di seguito vengono mostrati gli esempi:

- `region`
- `stack_name`
- `template_file`

Regole relative al valore dei parametri

TOML

- I valori booleani possono essere `true` o. `false` Ad esempio, `confirm_changeset = true`.
- Per i valori di stringa, utilizzare le virgolette (). `""` Ad esempio, `region = "us-west-2"`.
- Per i valori degli elenchi, utilizzate le virgolette (`""`) e separate ogni valore utilizzando uno spazio ( ). Ad esempio: `capabilities = "CAPABILITY_IAM CAPABILITY_NAMED_IAM"`.
- Per i valori che contengono un elenco di coppie chiave-valore, le coppie sono delimitate da spazi ( ) e il valore di ciascuna coppia è racchiuso tra virgolette codificate (). `\" \"` Ad esempio, `tags = "project=\"my-application\" stage=\"production\""`.
- Per i valori dei parametri che possono essere specificati più volte, il valore è una matrice di argomenti. Ad esempio: `image_repositories = ["my-function-1=image-repo-1", "my-function-2=image-repo-2"]`.

YAML

- I valori booleani possono essere `true` o. `false` Ad esempio, `confirm_changeset: true`.
- Per le voci che contengono un solo valore di stringa, le virgolette (`""`) sono facoltative. Ad esempio, `region: us-west-2`. Sono incluse le voci che contengono più coppie chiave-valore fornite come singola stringa. Di seguito è riportato un esempio:

```
$ sam deploy --tags "foo=bar hello=world"
```

```
default:
  deploy:
    parameters:
      tags: foo=bar hello=world
```

- Per le voci che contengono un elenco di valori o le voci che possono essere utilizzate più volte in un singolo comando, specificale come elenco di stringhe.

  Di seguito è riportato un esempio:

```
$ sam remote invoke --parameter "InvocationType=Event" --parameter "LogType=None"
```

```
default:
  remote_invoke:
    parameters:
        parameter:
        - InvocationType=Event
        - LogType=None
```

Precedenza di configurazione

Durante la configurazione dei valori, ha la seguente precedenza:

- I valori dei parametri forniti nella riga di comando hanno la precedenza sui valori corrispondenti nel file di configurazione e nella `Parameters` sezione del file modello.
- Se l'`--parameter-overrides`opzione viene utilizzata nella riga di comando o nel file di configurazione con la `parameter_overrides` chiave, i suoi valori hanno la precedenza sui valori nella `Parameters` sezione del file modello.
- Nel file di configurazione, le voci fornite per un comando specifico hanno la precedenza sulle voci globali. Nell'esempio seguente, il `sam deploy` comando utilizzerà il nome dello stack. `my-app-stack`

  TOML

```
[default.global.parameters]
stack_name = "common-stack"

[default.deploy.parameters]
stack_name = "my-app-stack"
```

  YAML

```
default:
  global:
```

```
    parameters:
      stack_name: common-stack
  deploy:
    parameters:
      stack_name: my-app-stack
```

Creazione e modifica dei file di configurazione

Creazione di file di configurazione

Quando si crea un'applicazione utilizzando`sam init`, viene creato un `samconfig.toml` file predefinito. È inoltre possibile creare manualmente il file di configurazione.

Modifica dei file di configurazione

È possibile modificare manualmente i file di configurazione. Inoltre, durante qualsiasi flusso AWS SAMCLI interattivo, i valori configurati verranno visualizzati tra parentesi (`[ ]`). Se modifichi questi valori, AWS SAMCLI aggiornerà il tuo file di configurazione.

Di seguito è riportato un esempio di flusso interattivo che utilizza il `sam deploy --guided` comando:

```
$ sam deploy --guided

Configuring SAM deploy
======================

    Looking for config file [samconfig.toml] :  Found
    Reading default arguments  :  Success

    Setting default arguments for 'sam deploy'
    =========================================
    Stack Name [sam-app]: ENTER
    AWS Region [us-west-2]: ENTER
    #Shows you resources changes to be deployed and require a 'Y' to initiate deploy
    Confirm changes before deploy [Y/n]: n
    #SAM needs permission to be able to create roles to connect to the resources in
 your template
    Allow SAM CLI IAM role creation [Y/n]: ENTER
    #Preserves the state of previously provisioned resources when an operation fails
    Disable rollback [y/N]: ENTER
```

```
    HelloWorldFunction may not have authorization defined, Is this okay? [y/N]: y
    Save arguments to configuration file [Y/n]: ENTER
    SAM configuration file [samconfig.toml]: ENTER
    SAM configuration environment [default]: ENTER
```

Quando si modifica il file di configurazione, AWS SAMCLI gestisce i valori globali come segue:

- Se il valore del parametro esiste nella `global` sezione del file di configurazione, AWS SAMCLI non scrive il valore nella sezione di comando specifica.
- Se il valore del parametro esiste `global` sia nella sezione di comando che in quella specifica, AWS SAMCLI elimina la voce specifica a favore del valore globale.

Gestione delle AWS SAMCLI versioni

Gestisci le versioni dell'interfaccia a riga di AWS Serverless Application Model comando (AWS SAMCLI) tramite l'aggiornamento, il downgrade e la disinstallazione. Facoltativamente, puoi scaricare e installare la nightly build. AWS SAMCLI

Argomenti

- Aggiornamento del AWS SAMCLI
- Disinstallazione di AWS SAMCLI
- Passa dall'uso Homebrew alla gestione di AWS SAMCLI
- Gestire la build AWS SAMCLI notturna
- Installare il file AWS SAMCLI in un ambiente virtuale utilizzando pip
- Gestire il AWS SAMCLI con Homebrew
- Risoluzione dei problemi

Aggiornamento del AWS SAMCLI

Linux

Per eseguire l'aggiornamento AWS SAMCLI su Linux, segui le istruzioni di installazione riportate inInstallazione di AWS SAMCLI, ma aggiungi l'`--update`opzione al comando install, come segue:

```
sudo ./sam-installation/install --update
```

macOS

AWS SAMCLIDeve essere aggiornato con lo stesso metodo utilizzato per installarlo. Si consiglia di utilizzare il programma di installazione del pacchetto per installare e aggiornare il. AWS SAMCLI

Per aggiornarlo AWS SAMCLI utilizzando il programma di installazione del pacchetto, installa la versione più recente del pacchetto. Per istruzioni, consulta Installazione di AWS SAMCLI.

Windows

Per aggiornare il AWS SAMCLI, ripeti Installa il AWS SAMCLI nuovamente la procedura di installazione di Windows.

Disinstallazione di AWS SAMCLI

Linux

Per disinstallare AWS SAMCLI su Linux, è necessario eliminare il collegamento simbolico e la directory di installazione eseguendo i seguenti comandi:

1. Individuare il collegamento simbolico e i percorsi di installazione.

   - Trovate il collegamento simbolico usando il which comando:

     ```
     which sam
     ```

     L'output mostra il percorso in cui si trovano i AWS SAM file binari, ad esempio:

     ```
      /usr/local/bin/sam
     ```

   - Trova la directory a cui punta il collegamento simbolico usando il ls comando:

     ```
     ls -l /usr/local/bin/sam
     ```

     Nell'esempio seguente, la directory di installazione è`/usr/local/aws-sam-cli`.

     ```
      lrwxrwxrwx 1 ec2-user ec2-user 49 Oct 22 09:49 /usr/local/bin/sam -> /usr/local/
     aws-sam-cli/current/bin/sam
     ```

2. Eliminare il collegamento simbolico.

```
sudo rm /usr/local/bin/sam
```

3. Eliminare la directory di installazione.

```
sudo rm -rf /usr/local/aws-sam-cli
```

macOS

Disinstallalo AWS SAMCLI con lo stesso metodo utilizzato per installarlo. Si consiglia di utilizzare il programma di installazione del pacchetto per installare il AWS SAMCLI.

Se hai installato il pacchetto AWS SAMCLI utilizzando il programma di installazione del pacchetto, segui questi passaggi per disinstallarlo.

Per disinstallare il AWS SAMCLI

1. Rimuovere il AWS SAMCLI programma modificando ed eseguendo quanto segue:

```
$ sudo rm -rf /path-to/aws-sam-cli
```

   a. *sudo*— Se l'utente dispone dei permessi di scrittura sulla posizione in cui è installato il AWS SAMCLI programma, non sudo è necessario. In caso contrario, sudo è richiesto.
   b. *`/path-to`*— Percorso in cui è stato installato il AWS SAMCLI programma. Il percorso predefinito è `/usr/local`.
2. Rimuovere il file AWS SAMCLI `$PATH` modificando ed eseguendo quanto segue:

```
$ sudo rm -rf /path-to-symlink-directory/sam
```

   a. *sudo*— Se l'utente dispone dei permessi di scrittura`$PATH`, non sudo è necessario. In caso contrario, sudo è richiesto.
   b. *`path-to-symlink-directory`*— La tua variabile di `$PATH` ambiente. Il percorso predefinito è `/usr/local/bin`.
3. Verificate che AWS SAMCLI sia disinstallato eseguendo quanto segue:

```
$ sam --version
command not found: sam
```

Windows

Per disinstallarlo AWS SAMCLI utilizzando le impostazioni di Windows, procedi nel seguente modo:

1. Dal menu Start, cerca «Aggiungi o rimuovi programmi».
2. Scegli il risultato denominato AWS SAM Command Line Interface, quindi scegli Disinstalla per avviare il programma di disinstallazione.
3. Conferma di voler disinstallare il. AWS SAMCLI

Passa dall'uso Homebrew alla gestione di AWS SAMCLI

Se si utilizza Homebrew per installare e aggiornare il AWS SAMCLI, si consiglia di utilizzare un metodo AWS supportato. Segui queste istruzioni per passare a un metodo supportato.

Per passare dall'uso Homebrew

1. Segui le istruzioni riportate in Disinstallazione di una CLI Homebrew installata AWS SAM per disinstallare la versione Homebrew gestita.
2. Segui le istruzioni su Installa il AWS SAMCLI per installare la AWS SAM CLI utilizzando un metodo supportato.

Gestire la build AWS SAMCLI notturna

Puoi scaricare e installare la AWS SAMCLI build notturna. Contiene una versione preliminare del AWS SAMCLI codice che potrebbe essere meno stabile della versione di produzione. Una volta installato, è possibile utilizzare il nightly build con il comando. `sam-nightly` È possibile installare e utilizzare contemporaneamente sia la versione di produzione che quella nightly build AWS SAMCLI di.

> **Note**
>
> La nightly build non contiene una versione preliminare dell'immagine di build. Per questo motivo, la creazione di un'applicazione serverless con l'--use-containeropzione utilizza l'ultima versione di produzione dell'immagine di build.

Installazione della AWS SAMCLI nightly build

Per installare la AWS SAMCLI versione notturna, segui queste istruzioni.

Linux

È possibile installare la versione nightly build di AWS SAMCLI sulla piattaforma Linux x86_64 utilizzando il programma di installazione del pacchetto.

Per installare AWS SAMCLI la nightly build

1. Scarica il programma di installazione del pacchetto dal repository sam-cli-nightly. aws-sam-cli GitHub
2. Segui i passaggi per AWS SAMCLIinstallare il pacchetto nightly build.

macOS

È possibile installare la versione nightly build di AWS SAMCLI onmacOS, utilizzando il programma di installazione del pacchetto nightly build.

Per installare la versione nightly build AWS SAMCLI

1. Scarica il programma di installazione del pacchetto per la tua piattaforma dal repository sam-cli-nightly. aws-sam-cli GitHub
2. Segui i passaggi per AWS SAMCLIinstallare il pacchetto nightly build.

Windows

La versione nightly build di AWS SAMCLI è disponibile con questo link per il download: AWS SAMCLInightly build. Per installare la versione nightly build su Windows, esegui gli stessi passaggi diInstalla il AWS SAMCLI, ma utilizza invece il link per il download di nightly build.

Per verificare di aver installato la versione nightly build, esegui il comando. sam-nightly --version L'output di questo comando è nel formato`1.X.Y.dev<YYYYMMDDHHmm>`, ad esempio:

```
SAM CLI, version 1.20.0.dev202103151200
```

Passa da Homebrew al programma di installazione del pacchetto

Se stai utilizzando Homebrew per installare e aggiornare la AWS SAMCLI nightly build e desideri passare all'utilizzo del programma di installazione del pacchetto, segui questi passaggi.

Per passare dal programma di installazione del Homebrew pacchetto

1. Disinstalla la build AWS SAMCLI nightly Homebrew installata.

```
$ brew uninstall aws-sam-cli-nightly
```

2. Verifica che la AWS SAMCLI nightly build sia disinstallata eseguendo quanto segue:

```
$ sam-nightly --version
zsh: command not found: sam-nightly
```

3. Segui i passaggi della sezione precedente per installare la nightly build. AWS SAMCLI

Installare il file AWS SAMCLI in un ambiente virtuale utilizzando pip

Si consiglia di utilizzare il programma di installazione nativo del pacchetto per installare. AWS SAMCLI Se è necessario utilizzarlopip, si consiglia di installarlo AWS SAMCLI in un ambiente virtuale. Ciò garantisce un ambiente di installazione pulito e un ambiente isolato in caso di errori.

> **Note**
>
> A partire dal 24 ottobre 2023, AWS SAM CLI il supporto perPython 3.7. Per ulteriori informazioni, consulta AWS SAMCLIinterruzione del supporto per Python 3.7.

Per installarlo AWS SAMCLI in un ambiente virtuale

1. Da una directory iniziale di tua scelta, crea un ambiente virtuale e assegnagli un nome.

   Linux / macOS

```
$ mkdir project
$ cd project
$ python3 -m venv venv
```

Windows

```
> mkdir project
> cd project
> py -3 -m venv venv
```

2. Attiva l'ambiente virtuale

   Linux / macOS

   ```
   $ . venv/bin/activate
   ```

   La richiesta cambia per mostrare che l'ambiente virtuale è attivo.

   ```
   (venv) $
   ```

   Windows

   ```
   > venv\Scripts\activate
   ```

   Il prompt cambia per mostrare che l'ambiente virtuale è attivo.

   ```
   (venv) >
   ```

3. Installalo AWS SAMCLI nel tuo ambiente virtuale.

   ```
   (venv) $ pip install --upgrade aws-sam-cli
   ```

4. Verifica che AWS SAMCLI sia installato correttamente.

   ```
   (venv) $ sam --version
   SAM CLI, version 1.94.0
   ```

5. Puoi utilizzare il comando `deactivate` per chiudere l'ambiente virtuale. Quando avvii una nuova sessione, devi attivare nuovamente l'ambiente.

Gestire il AWS SAMCLI con Homebrew

> **Note**
>
> A partire da settembre 2023, AWS non manterrà più il programma di Homebrew installazione AWS gestito per AWS SAMCLI ()`aws/tap/aws-sam-cli`. Per continuare a utilizzareHomebrew, puoi utilizzare il programma di installazione gestito dalla community ()`aws-sam-cli`. A partire da settembre 2023, qualsiasi Homebrew comando a cui fa riferimento `aws/tap/aws-sam-cli` verrà reindirizzato a. `aws-sam-cli`
> Ti consigliamo di utilizzare i nostri metodi di installazione e aggiornamento supportati.

Installazione dell' AWS SAMCLIutilizzo Homebrew

> **Note**
>
> Queste istruzioni utilizzano il programma di AWS SAMCLI Homebrew installazione gestito dalla comunità. Per ulteriore assistenza, consulta il repository homebrew-core.

Per installare il AWS SAMCLI

1. Esegui il seguente codice:

```
$ brew install aws-sam-cli
```

2. Verifica l’installazione:

```
$ sam --version
```

   Dopo la corretta installazione di AWS SAMCLI, si dovrebbe vedere un risultato simile al seguente:

```
SAM CLI, version 1.94.0
```

Aggiornamento dell'utilizzo AWS SAMCLI Homebrew

Per aggiornare l' AWS SAMCLIutilizzoHomebrew, esegui il seguente comando:

```
$ brew upgrade aws-sam-cli
```

Disinstallazione di una CLI Homebrew installata AWS SAM

Se AWS SAMCLI è stato installato utilizzandoHomebrew, segui questi passaggi per disinstallarlo.

Per disinstallare il AWS SAMCLI

1. Esegui il seguente codice:

   ```
   $ brew uninstall aws-sam-cli
   ```

2. Verificare che AWS SAMCLI sia disinstallato eseguendo quanto segue:

   ```
   $ sam --version
   command not found: sam
   ```

Passaggio al programma di installazione gestito Homebrew dalla community

Se stai usando il programma di Homebrew installazione AWS gestito (`aws/tap/aws-sam-cli`) e preferisci continuare a usarloHomebrew, ti consigliamo di passare al programma di Homebrew installazione gestito dalla community ()`aws-sam-cli`.

Per passare a un solo comando, esegui quanto segue:

```
$ brew uninstall aws-sam-cli && brew untap aws/tap && brew cleanup aws/tap && brew
 update && brew install aws-sam-cli
```

Segui queste istruzioni per eseguire ogni comando singolarmente.

Per passare al programma di Homebrew installazione gestito dalla community

1. Disinstalla la Homebrew versione AWS gestita di AWS SAMCLI:

   ```
   $ brew uninstall aws-sam-cli
   ```

2. Verifica che sia AWS SAMCLI stato disinstallato:

   ```
   $ which sam
   ```

```
sam not found
```

3. Rimuovi il AWS SAMCLI rubinetto AWS gestito:

```
$ brew untap aws/tap
```

Se ricevi un errore come il seguente, aggiungi l'`--force`opzione e riprova.

```
Error: Refusing to untap aws/tap because it contains the following installed
 formulae or casks:
aws-sam-cli-nightly
```

4. Rimuovi i file memorizzati nella cache per il programma di installazione AWS gestito:

```
$ brew cleanup aws/tap
```

5. Aggiornamento Homebrew e tutte le formule:

```
$ brew update
```

6. Installa la versione gestita dalla community di AWS SAMCLI:

```
$ brew install aws-sam-cli
```

7. Verifica che AWS SAMCLI sia installato correttamente:

```
$ sam --version
SAM CLI, version 1.94.0
```

Risoluzione dei problemi

Se riscontri errori durante l'installazione o l'utilizzo di AWS SAMCLI, consultaAWS SAMCLIrisoluzione dei problemi.

Impostazione delle AWS credenziali

L'interfaccia a riga di AWS SAM comando (CLI) richiede l'impostazione di AWS credenziali in modo da poter effettuare chiamate ai AWS servizi per conto dell'utente. Ad esempio, AWS SAMCLI effettua chiamate verso Amazon S3 e. CloudFormation

Potresti aver già impostato AWS le credenziali per lavorare con AWS strumenti, come uno dei AWS SDKs o il. AWS CLI In caso contrario, in questo argomento vengono illustrati gli approcci consigliati per l'impostazione delle AWS credenziali.

Per impostare AWS le credenziali, devi disporre dell'ID della chiave di accesso e della chiave di accesso segreta per l'utente IAM che desideri configurare. Per informazioni sulla chiave di accesso IDs e sulle chiavi di accesso segrete, consulta Managing Access Keys for IAM Users nella IAM User Guide.

Quindi, stabilisci se hai AWS CLI installato il. Quindi segui le istruzioni in una delle seguenti sezioni:

Usando il AWS CLI

Se l'hai AWS CLI installato, usa il `aws configure` comando e segui le istruzioni:

```
$ aws configure
AWS Access Key ID [None]: your_access_key_id
AWS Secret Access Key [None]: your_secret_access_key
Default region name [None]:
Default output format [None]:
```

Per informazioni sul *`aws configure`* comando, consulta la sezione Configurazione rapida di AWS CLI nella Guida per l'utente.AWS Command Line Interface

Non si utilizza il AWS CLI

Se non l'hai AWS CLI installato, puoi creare un file di credenziali o impostare variabili di ambiente:

- File di credenziali: è possibile impostare le credenziali nel file delle AWS credenziali sul sistema locale. Questo file deve trovarsi in una delle seguenti posizioni:
  - `~/.aws/credentials`su Linux o macOS
  - `C:\Users\`*`USERNAME`*`\.aws\credentials` in Windows

  Questo file deve contenere righe nel seguente formato:

  ```
  [default]
  aws_access_key_id = your_access_key_id
  aws_secret_access_key = your_secret_access_key
  ```

- Variabili di ambiente: puoi impostare le variabili di `AWS_SECRET_ACCESS_KEY` ambiente `AWS_ACCESS_KEY_ID` e.

  Per impostare queste variabili su Linux o macOS, usa il comando export:

  ```
  export AWS_ACCESS_KEY_ID=your_access_key_id
  export AWS_SECRET_ACCESS_KEY=your_secret_access_key
  ```

  Per impostare queste variabili su Windows, usa il comando set:

  ```
  set AWS_ACCESS_KEY_ID=your_access_key_id
  set AWS_SECRET_ACCESS_KEY=your_secret_access_key
  ```

Telemetria nel AWS SAMCLI

In AWS, sviluppiamo e lanciamo servizi basati su ciò che apprendiamo dalle interazioni con i clienti. Utilizziamo il feedback dei clienti per iterare sul nostro prodotto. La telemetria è un'informazione aggiuntiva che ci aiuta a comprendere meglio le esigenze dei nostri clienti, diagnosticare i problemi e fornire funzionalità che migliorano l'esperienza del cliente.

L'interfaccia a riga di AWS SAM comando (CLI) raccoglie dati di telemetria, ad esempio metriche di utilizzo generiche, informazioni sul sistema e sull'ambiente ed errori. Per informazioni dettagliate sui tipi di telemetria raccolti, consulta. Tipo di informazioni da raccogliere

Non AWS SAMCLI raccoglie informazioni personali, come nomi utente o indirizzi e-mail. Inoltre, non estrae informazioni sensibili a livello di progetto.

I clienti controllano se la telemetria è attivata e possono modificare le impostazioni in qualsiasi momento. Se la telemetria rimane attiva, AWS SAMCLI invia i dati di telemetria in background senza richiedere alcuna interazione aggiuntiva con il cliente.

Disattiva la telemetria per una sessione

Nei sistemi operativi macOS e Linux, puoi disattivare la telemetria per una singola sessione. Per disattivare la telemetria per la sessione corrente, esegui il comando seguente per impostare la variabile di ambiente `SAM_CLI_TELEMETRY` su `false`. Ripeti il comando per ogni nuovo terminale o sessione.

```
export SAM_CLI_TELEMETRY=0
```

Disattivazione della telemetria per il tuo profilo in tutte le sessioni

Esegui i seguenti comandi per disattivare la telemetria per tutte le sessioni quando la esegui sul tuo sistema operativo. AWS SAMCLI

Per disattivare la telemetria in Linux

1. Esegui:

```
echo "export SAM_CLI_TELEMETRY=0" >>~/.profile
```

2. Esegui:

```
source ~/.profile
```

Per disattivare la telemetria in Linux

1. Esegui:

```
echo "export SAM_CLI_TELEMETRY=0" >>~/.profile
```

2. Esegui:

```
source ~/.profile
```

Per disattivare la telemetria in Linux

Puoi impostare temporaneamente la variabile di ambiente per tutta la durata della finestra del terminale con il seguente comando:

Se si utilizza il prompt dei comandi:

```
set SAM_CLI_TELEMETRY=0
```

Se si utilizza PowerShell:

```
$env:SAM_CLI_TELEMETRY=0
```

Per impostare la variabile di ambiente in modo permanente nel prompt dei comandi oppure PowerShell, utilizzate il seguente comando:

```
setx SAM_CLI_TELEMETRY 0
```

> **Note**
>
> Le modifiche entreranno in vigore solo dopo la chiusura e la riapertura del terminale.

Tipo di informazioni da raccogliere

- Informazioni sull'utilizzo: i comandi e i sottocomandi generici eseguiti dai clienti.
- Errori e informazioni diagnostiche: lo stato e la durata dei comandi eseguiti dai clienti, inclusi i codici di uscita, i nomi delle eccezioni interne e gli errori durante la connessione a Docker.
- Informazioni sul sistema e sull'ambiente: la versione di Python, il sistema operativo (Windows, Linux o macOS), l'ambiente in cui AWS SAMCLI viene eseguito (ad esempio AWS CodeBuild, un toolkit AWS IDE o un terminale) e i valori hash degli attributi di utilizzo.

Ulteriori informazioni

I dati di telemetria AWS SAMCLI raccolti aderiscono alle politiche sulla privacy dei dati. AWS Per ulteriori informazioni, consulta gli argomenti seguenti:

- AWS Termini del servizio
- Domande frequenti sulla privacy dei dati

AWS SAMCLIrisoluzione dei problemi

Questa sezione fornisce dettagli su come risolvere i messaggi di errore durante l'utilizzo, l'installazione e la gestione dell'interfaccia a riga di AWS Serverless Application Model comando ().AWS SAMCLI

Argomenti

- Risoluzione dei problemi
- Messaggi di errore

- Messaggi di avviso

Risoluzione dei problemi

Per indicazioni sulla risoluzione dei problemi relative a AWS SAMCLI, vedereRisoluzione degli errori di installazione.

Messaggi di errore

Errore curl: «curl: (6) Impossibile risolvere:...»

Quando si tenta di richiamare l'endpoint API Gateway, viene visualizzato il seguente errore:

```
curl: (6) Could not resolve: endpointdomain (Domain name not found)
```

Ciò significa che hai tentato di inviare una richiesta a un dominio non valido. Ciò può accadere se l'applicazione serverless non è stata distribuita correttamente o se hai un errore di battitura nel comando. curl Verifica che l'applicazione sia stata distribuita correttamente utilizzando la CloudFormation console o il AWS CLI, e verifica che il curl comando sia corretto.

Errore: impossibile trovare informazioni esatte sulle risorse con il nome dello stack specificato

Quando si esegue il `sam remote invoke` comando su un'applicazione che contiene una singola risorsa della funzione Lambda, viene visualizzato il seguente errore:

```
Error: Can't find exact resource information with given <stack-name>. Please provide
 full resource ARN or --stack-name to resolve the ambiguity.
```

Causa possibile: non hai fornito l'**`--stack-name`**opzione.

> Se una funzione ARN non viene fornita come argomento, il `sam remote invoke` comando richiede che venga fornita l'`--stack-name`opzione.

Soluzione: fornire l'**`--stack-name`**opzione.

> Di seguito è riportato un esempio:
>
> ```
> $ sam remote invoke --stack-name sam-app
> ```

```
Invoking Lambda Function HelloWorldFunction



START RequestId: 40593abb-e1ad-4d99-87bd-ac032e364e82 Version: $LATEST
END RequestId: 40593abb-e1ad-4d99-87bd-ac032e364e82
REPORT RequestId: 40593abb-e1ad-4d99-87bd-ac032e364e82  Duration: 11.31 ms
  Billed Duration: 12 ms  Memory Size: 128 MB     Max Memory Used: 67 MB  Init
 Duration: 171.71 ms
{"statusCode":200,"body":"{\"message\":\"hello world\"}"}%
```

Errore: impossibile trovare le informazioni sulle risorse dal nome dello stack

Quando si esegue il `sam remote invoke` comando e si passa una funzione Lambda ARN come argomento, viene visualizzato il seguente errore:

```
Error: Can't find resource information from stack name (<stack-name>) and resource id
 (<function-id>)
```

Possibile causa: il valore del nome dello stack è definito nel file. **`samconfig.toml`**

> Il AWS SAMCLI primo verifica la presenza di un `samconfig.toml` nome per lo stack nel file. Se specificato, l'argomento viene passato come valore ID logico.

Soluzione: passate invece l'ID logico della funzione.

> È possibile passare l'ID logico della funzione come argomento anziché l'ARN della funzione.

Soluzione: rimuovete il valore del nome dello stack dal file di configurazione.

> È possibile rimuovere il valore del nome dello stack dal file di configurazione. Ciò impedisce di passare l' AWS SAMCLIARN della funzione come valore ID logico.
>
> Esegui `sam build` dopo aver modificato il file di configurazione.

Errore: creazione di risorse gestite non riuscita: impossibile individuare le credenziali

Quando si esegue il sam deploy comando, viene visualizzato il seguente errore:

```
Error: Failed to create managed resources: Unable to locate credentials
```

Ciò significa che non sono state impostate AWS le credenziali per consentire l'esecuzione AWS SAMCLI di chiamate AWS di servizio. Per risolvere questo problema, è necessario impostare AWS le credenziali. Per ulteriori informazioni, consulta Impostazione delle AWS credenziali.

Errore: FileNotFoundError in Windows

Quando si eseguono AWS SAMCLI comandi in Windows, è possibile che venga visualizzato il seguente errore:

```
Error: FileNotFoundError
```

Possibile causa: AWS SAMCLI potrebbe interagire con percorsi di file che superano il limite massimo di percorso di Windows.

Soluzione: per risolvere questo problema, è necessario abilitare il nuovo comportamento dei percorsi lunghi. A tale scopo, consulta Abilitare percorsi lunghi in Windows 10, versione 1607 e successive nella documentazione sullo sviluppo di app per Microsoft Windows.

Errore: il risolutore di dipendenze di pip...

Esempio di testo dell'errore:

```
ERROR: pip's dependency resolver does not currently take into account all the packages
 that are installed. This behaviour is the source of the following dependency
 conflicts.
aws-sam-cli 1.58.0 requires aws-sam-translator==1.51.0, but you have aws-sam-translator
 1.58.0 which is incompatible.
aws-sam-cli 1.58.0 requires typing-extensions==3.10.0.0, but you have typing-extensions
 4.4.0 which is incompatible.
```

Possibile causa: se si utilizza pip per installare pacchetti, le dipendenze tra i pacchetti potrebbero essere in conflitto.

Ogni versione del `aws-sam-cli` pacchetto dipende da una versione del `aws-sam-translator` pacchetto. Ad esempio, la `aws-sam-cli` v1.58.0 può dipendere dalla v1.51.0. `aws-sam-translator`

Se si installa l' AWS SAMCLIusingpip, quindi si installa un altro pacchetto che dipende da una versione più recente di, si verificherà quanto segue: `aws-sam-translator`

- La versione più recente di `aws-sam-translator` will install.

- La versione corrente `aws-sam-cli` e la versione più recente di `aws-sam-translator` potrebbero non essere compatibili.
- Quando si utilizza il AWS SAMCLI, si verificherà l'errore del risolutore delle dipendenze.

Soluzioni:

1. Usa il programma di installazione del pacchetto AWS SAMCLI nativo.
   a. Disinstalla AWS SAMCLI usando pip. Per istruzioni, consulta Disinstallazione di AWS SAMCLI.
   b. Installa AWS SAMCLI utilizzando il programma di installazione del pacchetto nativo. Per istruzioni, consulta Installa il AWS SAMCLI.
   c. Se necessario, esegui l'aggiornamento AWS SAMCLI utilizzando il programma di installazione del pacchetto nativo. Per istruzioni, consulta Aggiornamento del AWS SAMCLI.
2. Se è necessario utilizzarepip, si consiglia di installare la AWS SAM CLI in un ambiente virtuale. Ciò garantisce un ambiente di installazione pulito e un ambiente isolato in caso di errori. Per istruzioni, consulta Installare il file AWS SAMCLI in un ambiente virtuale utilizzando pip.

Errore: nessun comando di questo tipo «remoto»

Quando si esegue il `sam remote invoke` comando, viene visualizzato il seguente errore:

```
$ sam remote invoke ...
2023-06-20 08:15:07 Command remote not available
Usage: sam [OPTIONS] COMMAND [ARGS]...
Try 'sam -h' for help.

Error: No such command 'remote'.
```

Causa possibile: la tua versione di non AWS SAMCLI è aggiornata.

Il AWS SAMCLI `sam remote invoke` comando è stato rilasciato con la AWS SAMCLI versione 1.88.0. Puoi controllare la tua versione eseguendo il `sam --version` comando.

Soluzione: aggiorna il tuo AWS SAMCLI alla versione più recente.

Per istruzioni, consulta Aggiornamento del AWS SAMCLI.

Errore: l'esecuzione locale di progetti AWS SAM richiedeDocker. L'hai installato?

Quando si esegue il sam local start-api comando, viene visualizzato il seguente errore:

```
Error: Running AWS SAM projects locally requires Docker. Have you got it installed?
```

Ciò significa che l'installazione non è stata Docker eseguita correttamente. Dockerè necessario per testare l'applicazione localmente. Per risolvere questo problema, segui le istruzioni per l'installazione di Docker per il tuo host di sviluppo. Per ulteriori informazioni, consulta Installazione di Docker.

Errore: vincoli di sicurezza non soddisfatti

Durante l'esecuzionesam deploy --guided, ti viene posta la domanda. `Function may not have authorization defined, Is this okay? [y/N]` Se rispondi a questa richiesta con **N** (la risposta predefinita), viene visualizzato il seguente errore:

```
Error: Security Constraints Not Satisfied
```

Il messaggio ti informa che l'applicazione che stai per distribuire potrebbe avere un'API Amazon API Gateway accessibile pubblicamente configurata senza autorizzazione. Rispondendo **N** a questa richiesta, stai dicendo che non va bene.

Per risolvere questo problema, sono disponibili le seguenti opzioni:

- Configura la tua applicazione con autorizzazione. Per informazioni sulla configurazione dell'autorizzazione, vedereControlla l'accesso alle API con il tuo AWS SAM modello.
- Se intendi avere un endpoint API accessibile pubblicamente senza autorizzazione, riavvia la distribuzione e rispondi a questa domanda indicando che sei d'accordo con la distribuzione. **Y**

messaggio: Token di autenticazione mancante

Quando si tenta di richiamare l'endpoint API Gateway, viene visualizzato il seguente errore:

```
{"message":"Missing Authentication Token"}
```

Ciò significa che hai tentato di inviare una richiesta al dominio corretto, ma l'URI non è riconoscibile. Per risolvere questo problema, verifica l'URL completo e aggiorna il curl comando con l'URL corretto.

Messaggi di avviso

Avviso:... AWS non manterrà più il Homebrew programma di installazione per AWS SAM ...

Durante l'installazione di AWS SAMCLI UsingHomebrew, viene visualizzato il seguente messaggio di avviso:

```
Warning: ... AWS will no longer maintain the Homebrew installer for AWS SAM (aws/tap/
aws-sam-cli).
    For AWS supported installations, use the first party installers ...
```

Causa potenziale: AWS non è più possibile mantenere Homebrew il supporto.

A partire da settembre 2023, AWS non manterrà più il Homebrew programma di installazione per. AWS SAMCLI

Soluzione: utilizzare un metodo di installazione AWS supportato.

- I metodi di installazione AWS supportati sono disponibili all'indirizzoInstalla il AWS SAMCLI.

Soluzione: per continuare a utilizzareHomebrew, utilizzate il programma di installazione gestito dalla community.

- È possibile utilizzare il programma di Homebrew installazione gestito dalla community a propria discrezione. Per istruzioni, consulta Gestire il AWS SAMCLI con Homebrew.

AWS SAM riferimento del connettore

Questa sezione contiene informazioni di riferimento per il tipo di risorsa del connettore AWS Serverless Application Model (AWS SAM). Per un'introduzione ai connettori, vedereGestione delle autorizzazioni delle risorse con i connettori AWS SAM.

Tipi di risorse di origine e destinazione supportati per i connettori

Il tipo di `AWS::Serverless::Connector` risorsa supporta un numero selezionato di connessioni tra le risorse di origine e di destinazione. Quando configurate i connettori nel AWS SAM modello, utilizzate la tabella seguente per fare riferimento alle connessioni supportate e alle proprietà che devono essere definite per ogni tipo di risorsa di origine e destinazione. Per ulteriori informazioni sulla configurazione dei connettori nel modello, vedere. AWS::Serverless::Connector

Sia per le risorse di origine che per quelle di destinazione, se definite all'interno dello stesso modello, utilizzate la `Id` proprietà. Facoltativamente, `Qualifier` è possibile aggiungere un per restringere l'ambito della risorsa definita. Se la risorsa non si trova all'interno dello stesso modello, utilizzate una combinazione di proprietà supportate.

Per richiedere nuove connessioni, invia un nuovo problema al serverless-application-model AWS GitHubrepository.

Source type (Tipo di origine)	Tipo di destinazione	Permissions	Proprietà della fonte	Proprietà di destinazione
`AWS::ApiGateway::RestApi`	`AWS::Lambda::Function`	`Write`	`Id` oppure `Qualifier` `ResourceId`, e `Type`	`Id` `Arn` e `Type`
`AWS::ApiGateway::RestApi`	`AWS::Serverless::Function`	`Write`	`Id` oppure `Qualifier` `ResourceId`, e `Type`	`Id` `Arn` e `Type`
`AWS::ApiGatewayV2::Api`	`AWS::Lambda::Function`	`Write`	`Id` oppure `Qualifier` `ResourceId`, e `Type`	`Id` `Arn` e `Type`
`AWS::ApiGatewayV2::Api`	`AWS::Serverless::Function`	`Write`	`Id` oppure `Qualifier` `ResourceId`, e `Type`	`Id` `Arn` e `Type`
`AWS::AppSync::DataSource`	`AWS::DynamoDB::Table`	`Read`	`Id` `RoleName` e `Type`	`Id` `Arn` e `Type`
`AWS::AppSync::DataSource`	`AWS::DynamoDB::Table`	`Write`	`Id` `RoleName` e `Type`	`Id` `Arn` e `Type`

Source type (Tipo di origine)	Tipo di destinazione	Permissions	Proprietà della fonte	Proprietà di destinazione
AWS::AppSync::DataSource	AWS::Events::EventBus	Write	Ido RoleName e Type	Ido Arn e Type
AWS::AppSync::DataSource	AWS::Lambda::Function	Write	Ido RoleName e Type	Ido Arn e Type
AWS::AppSync::DataSource	AWS::Serverless::Function	Write	Ido RoleName e Type	Ido Arn e Type
AWS::AppSync::DataSource	AWS::Serverless::SimpleTable	Read	Ido RoleName e Type	Ido Arn e Type
AWS::AppSync::DataSource	AWS::Serverless::SimpleTable	Write	Ido RoleName e Type	Ido Arn e Type
AWS::AppSync::GraphQLApi	AWS::Lambda::Function	Write	Ido ResourceId e Type	Ido Arn e Type
AWS::AppSync::GraphQLApi	AWS::Serverless::Function	Write	Ido ResourceId e Type	Ido Arn e Type
AWS::DynamoDB::Table	AWS::Lambda::Function	Read	Ido Arn e Type	Ido RoleName e Type
AWS::DynamoDB::Table	AWS::Serverless::Function	Read	Ido Arn e Type	Ido RoleName e Type

Source type (Tipo di origine)	Tipo di destinazione	Permissions	Proprietà della fonte	Proprietà di destinazione
`AWS::Events::Rule`	`AWS::Events::EventBus`	`Write`	`Ido RoleName` e `Type`	`Ido Arn` e `Type`
`AWS::Events::Rule`	`AWS::Lambda::Function`	`Write`	`Ido Arn` e `Type`	`Ido Arn` e `Type`
`AWS::Events::Rule`	`AWS::Serverless::Function`	`Write`	`Ido Arn` e `Type`	`Ido Arn` e `Type`
`AWS::Events::Rule`	`AWS::Serverless::StateMachine`	`Write`	`Ido RoleName` e `Type`	`Ido Arn` e `Type`
`AWS::Events::Rule`	`AWS::SNS::Topic`	`Write`	`Ido Arn` e `Type`	`Ido Arn` e `Type`
`AWS::Events::Rule`	`AWS::SQS::Queue`	`Write`	`Ido Arn` e `Type`	`Id`oppure `ArnQueueUrl`, e `Type`
`AWS::Events::Rule`	`AWS::StepFunctions::StateMachine`	`Write`	`Ido RoleName` e `Type`	`Ido Arn` e `Type`
`AWS::Lambda::Function`	`AWS::DynamoDB::Table`	`Read`, `Write`	`Ido RoleName` e `Type`	`Ido Arn` e `Type`

Source type (Tipo di origine)	Tipo di destinazione	Permissions	Proprietà della fonte	Proprietà di destinazione
AWS::Lambda::Function	AWS::Events::EventBus	Write	Ido RoleName e Type	Ido Arn e Type
AWS::Lambda::Function	AWS::Lambda::Function	Write	Ido RoleName e Type	Ido Arn e Type
AWS::Lambda::Function	AWS::Location::PlaceIndex	Read	Ido RoleName e Type	Ido Arn e Type
AWS::Lambda::Function	AWS::S3::Bucket	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::Lambda::Function	AWS::Serverless::Function	Write	Ido RoleName e Type	Ido Arn e Type
AWS::Lambda::Function	AWS::Serverless::SimpleTable	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::Lambda::Function	AWS::Serverless::StateMachine	Read, Write	Ido RoleName e Type	Idoppure ArnName, e Type
AWS::Lambda::Function	AWS::SNS::Topic	Write	Ido RoleName e Type	Ido Arn e Type

Source type (Tipo di origine)	Tipo di destinazione	Permissions	Proprietà della fonte	Proprietà di destinazione
AWS::Lambda::Function	AWS::SQS::Queue	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::Lambda::Function	AWS::StepFunctions::StateMachine	Read, Write	Ido RoleName e Type	Idoppure ArnName, e Type
AWS::S3::Bucket	AWS::Lambda::Function	Write	Ido Arn e Type	Ido Arn e Type
AWS::S3::Bucket	AWS::Serverless::Function	Write	Ido Arn e Type	Ido Arn e Type
AWS::Serverless::Api	AWS::Lambda::Function	Write	Idoppure Qualifier ResourceId , e Type	Ido Arn e Type
AWS::Serverless::Api	AWS::Serverless::Function	Write	Idoppure Qualifier ResourceId , e Type	Ido Arn e Type
AWS::Serverless::Function	AWS::DynamoDB::Table	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::Function	AWS::Events::EventBus	Write	Ido RoleName e Type	Ido Arn e Type

Source type (Tipo di origine)	Tipo di destinazione	Permissions	Proprietà della fonte	Proprietà di destinazione
AWS::Serverless::Function	AWS::Lambda::Function	Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::Function	AWS::S3::Bucket	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::Function	AWS::Serverless::Function	Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::Function	AWS::Serverless::SimpleTable	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::Function	AWS::Serverless::StateMachine	Read, Write	Ido RoleName e Type	Idoppure ArnName, e Type
AWS::Serverless::Function	AWS::SNS::Topic	Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::Function	AWS::SQS::Queue	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::Function	AWS::StepFunctions::StateMachine	Read, Write	Ido RoleName e Type	Idoppure ArnName, e Type

Source type (Tipo di origine)	Tipo di destinazione	Permissions	Proprietà della fonte	Proprietà di destinazione
AWS::Serverless::HttpApi	AWS::Lambda::Function	Write	Idoppure Qualifier ResourceId , e Type	Ido Arn e Type
AWS::Serverless::HttpApi	AWS::Serverless::Function	Write	Idoppure Qualifier ResourceId , e Type	Ido Arn e Type
AWS::Serverless::SimpleTable	AWS::Lambda::Function	Read	Ido Arn e Type	Ido RoleName e Type
AWS::Serverless::SimpleTable	AWS::Serverless::Function	Read	Ido Arn e Type	Ido RoleName e Type
AWS::Serverless::StateMachine	AWS::DynamoDB::Table	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::StateMachine	AWS::Events::EventBus	Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::StateMachine	AWS::Lambda::Function	Write	Ido RoleName e Type	Ido Arn e Type

Source type (Tipo di origine)	Tipo di destinazione	Permissions	Proprietà della fonte	Proprietà di destinazione
AWS::Serverless::StateMachine	AWS::S3::Bucket	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::StateMachine	AWS::Serverless::Function	Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::StateMachine	AWS::Serverless::SimpleTable	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::StateMachine	AWS::Serverless::StateMachine	Read, Write	Ido RoleName e Type	Idoppure ArnName, e Type
AWS::Serverless::StateMachine	AWS::SNS::Topic	Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::StateMachine	AWS::SQS::Queue	Write	Ido RoleName e Type	Ido Arn e Type
AWS::Serverless::StateMachine	AWS::StepFunctions::StateMachine	Read, Write	Ido RoleName e Type	Idoppure ArnName, e Type

Source type (Tipo di origine)	Tipo di destinazione	Permissions	Proprietà della fonte	Proprietà di destinazione
AWS::SNS::Topic	AWS::Lambda::Function	Write	Ido Arn e Type	Ido Arn e Type
AWS::SNS::Topic	AWS::Serverless::Function	Write	Ido Arn e Type	Ido Arn e Type
AWS::SNS::Topic	AWS::SQS::Queue	Write	Ido Arn e Type	Idoppure ArnQueueUrl, e Type
AWS::SQS::Queue	AWS::Lambda::Function	Read, Write	Ido Arn e Type	Ido RoleName e Type
AWS::SQS::Queue	AWS::Serverless::Function	Read, Write	Ido Arn e Type	Ido RoleName e Type
AWS::StepFunctions::StateMachine	AWS::DynamoDB::Table	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::StepFunctions::StateMachine	AWS::Events::EventBus	Write	Ido RoleName e Type	Ido Arn e Type
AWS::StepFunctions::StateMachine	AWS::Lambda::Function	Write	Ido RoleName e Type	Ido Arn e Type

Source type (Tipo di origine)	Tipo di destinazione	Permissions	Proprietà della fonte	Proprietà di destinazione
AWS::StepFunctions::StateMachine	AWS::S3::Bucket	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::StepFunctions::StateMachine	AWS::Serverless::Function	Write	Ido RoleName e Type	Ido Arn e Type
AWS::StepFunctions::StateMachine	AWS::Serverless::SimpleTable	Read, Write	Ido RoleName e Type	Ido Arn e Type
AWS::StepFunctions::StateMachine	AWS::Serverless::StateMachine	Read, Write	Ido RoleName e Type	Idoppure ArnName, e Type
AWS::StepFunctions::StateMachine	AWS::SNS::Topic	Write	Ido RoleName e Type	Ido Arn e Type
AWS::StepFunctions::StateMachine	AWS::SQS::Queue	Write	Ido RoleName e Type	Ido Arn e Type
AWS::StepFunctions::StateMachine	AWS::StepFunctions::StateMachine	Read, Write	Ido RoleName e Type	Idoppure ArnName, e Type

Politiche IAM create dai connettori

Questa sezione documenta le politiche AWS Identity and Access Management (IAM) create da AWS SAM quando si utilizzano i connettori.

`AWS::DynamoDB::Table` Da a `AWS::Lambda::Function`

Tipo di policy

Politica gestita dal cliente allegata al `AWS::Lambda::Function` ruolo.

Categorie di accesso

`Read`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:DescribeStream",
        "dynamodb:GetRecords",
        "dynamodb:GetShardIterator",
        "dynamodb:ListStreams"
      ],
      "Resource": [
        "%{Source.Arn}/stream/*"
      ]
    }
  ]
}
```

`AWS::Events::Rule` Da a `AWS::SNS::Topic`

Tipo di politica

`AWS::SNS::TopicPolicy`allegato al`AWS::SNS::Topic`.

Categorie di accesso

`Write`

```
{
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Resource": "%{Destination.Arn}",
      "Action": "sns:Publish",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": "%{Source.Arn}"
        }
      }
    }
  ]
}
```

`AWS::Events::Rule` Da a `AWS::Events::EventBus`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::Events::Rule` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutEvents"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

`AWS::Events::Rule` Da a `AWS::StepFunctions::StateMachine`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::Events::Rule` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "states:StartExecution"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

`AWS::Events::Rule` Da a `AWS::Lambda::Function`

Tipo di politica

`AWS::Lambda::Permission`allegato al`AWS::Lambda::Function`.

Categorie di accesso

`Write`

```
{
  "Action": "lambda:InvokeFunction",
  "Principal": "events.amazonaws.com",
  "SourceArn": "%{Source.Arn}"
}
```

`AWS::Events::Rule` Da a `AWS::SQS::Queue`

Tipo di politica

`AWS::SQS::QueuePolicy`allegato al`AWS::SQS::Queue`.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Resource": "%{Destination.Arn}",
      "Action": "sqs:SendMessage",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": "%{Source.Arn}"
        }
      }
    }
  ]
}
```

`AWS::Lambda::Function` Da a `AWS::Lambda::Function`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::Lambda::Function` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeAsync",
        "lambda:InvokeFunction"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
```

```
}
```

`AWS::Lambda::Function` Da a `AWS::S3::Bucket`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::Lambda::Function` ruolo.

Categorie di accesso

`Read`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectAcl",
        "s3:GetObjectLegalHold",
        "s3:GetObjectRetention",
        "s3:GetObjectTorrent",
        "s3:GetObjectVersion",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionForReplication",
        "s3:GetObjectVersionTorrent",
        "s3:ListBucket",
        "s3:ListBucketMultipartUploads",
        "s3:ListBucketVersions",
        "s3:ListMultipartUploadParts"
      ],
      "Resource": [
        "%{Destination.Arn}",
        "%{Destination.Arn}/*"
      ]
    }
  ]
}
```

`Write`

```
{
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": [
        "s3:AbortMultipartUpload",
        "s3:DeleteObject",
        "s3:DeleteObjectVersion",
        "s3:PutObject",
        "s3:PutObjectLegalHold",
        "s3:PutObjectRetention",
        "s3:RestoreObject"
      ],
      "Resource": [
        "%{Destination.Arn}",
        "%{Destination.Arn}/*"
      ]
    }
  ]
}
```

`AWS::Lambda::Function` Da a `AWS::DynamoDB::Table`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::Lambda::Function` ruolo.

Categorie di accesso

`Read`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:Query",
        "dynamodb:Scan",
        "dynamodb:BatchGetItem",
        "dynamodb:ConditionCheckItem",
        "dynamodb:PartiQLSelect"
      ],
      "Resource": [
        "%{Destination.Arn}",
        "%{Destination.Arn}/index/*"
```

```
      ]
    }
  ]
}
```

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:UpdateItem",
        "dynamodb:DeleteItem",
        "dynamodb:BatchWriteItem",
        "dynamodb:PartiQLDelete",
        "dynamodb:PartiQLInsert",
        "dynamodb:PartiQLUpdate"
      ],
      "Resource": [
        "%{Destination.Arn}",
        "%{Destination.Arn}/index/*"
      ]
    }
  ]
}
```

`AWS::Lambda::Function` Da a `AWS::SQS::Queue`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::Lambda::Function` ruolo.

Categorie di accesso

`Read`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "sqs:ReceiveMessage",
        "sqs:GetQueueAttributes"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sqs:DeleteMessage",
        "sqs:SendMessage",
        "sqs:ChangeMessageVisibility",
        "sqs:PurgeQueue"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

`AWS::Lambda::Function` Da a `AWS::SNS::Topic`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::Lambda::Function` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "sns:Publish"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

`AWS::Lambda::Function` Da a `AWS::StepFunctions::StateMachine`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::Lambda::Function` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "states:StartExecution",
        "states:StartSyncExecution"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "states:StopExecution"
      ],
      "Resource": [
        "arn:${AWS::Partition}:states:${AWS::Region}:${AWS::AccountId}:execution:
%{Destination.Name}:*"
      ]
    }
  ]
}
```

`Read`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "states:DescribeStateMachine",
        "states:ListExecutions"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "states:DescribeExecution",
        "states:DescribeStateMachineForExecution",
        "states:GetExecutionHistory"
      ],
      "Resource": [
        "arn:${AWS::Partition}:states:${AWS::Region}:${AWS::AccountId}:execution:
%{Destination.Name}:*"
      ]
    }
  ]
}
```

`AWS::Lambda::Function` Da a `AWS::Events::EventBus`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::Lambda::Function` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "events:PutEvents"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

`AWS::Lambda::Function` Da a `AWS::Location::PlaceIndex`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::Lambda::Function` ruolo.

Categorie di accesso

`Read`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "geo:DescribePlaceIndex",
        "geo:GetPlace",
        "geo:SearchPlaceIndexForPosition",
        "geo:SearchPlaceIndexForSuggestions",
        "geo:SearchPlaceIndexForText"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

`AWS::ApiGatewayV2::Api` Da a `AWS::Lambda::Function`

Tipo di politica

`AWS::Lambda::Permission`allegato al`AWS::Lambda::Function`.

Categorie di accesso

`Write`

```
{
  "Action": "lambda:InvokeFunction",
  "Principal": "apigateway.amazonaws.com",
  "SourceArn": "arn:${AWS::Partition}:execute-api:${AWS::Region}:${AWS::AccountId}:
%{Source.ResourceId}/%{Source.Qualifier}"
}
```

`AWS::ApiGateway::RestApi` Da a `AWS::Lambda::Function`

Tipo di politica

`AWS::Lambda::Permission`allegato al`AWS::Lambda::Function`.

Categorie di accesso

`Write`

```
{
  "Action": "lambda:InvokeFunction",
  "Principal": "apigateway.amazonaws.com",
  "SourceArn": "arn:${AWS::Partition}:execute-api:${AWS::Region}:${AWS::AccountId}:
%{Source.ResourceId}/%{Source.Qualifier}"
}
```

`AWS::SNS::Topic` Da a `AWS::SQS::Queue`

Tipo di politica

`AWS::SQS::QueuePolicy`allegato al`AWS::SQS::Queue`.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "sns.amazonaws.com"
      },
      "Resource": "%{Destination.Arn}",
```

```
      "Action": "sqs:SendMessage",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": "%{Source.Arn}"
        }
      }
    }
  ]
}
```

`AWS::SNS::Topic` Da a `AWS::Lambda::Function`

Tipo di politica

`AWS::Lambda::Permission`allegato al`AWS::Lambda::Function`.

Categorie di accesso

`Write`

```
{
  "Action": "lambda:InvokeFunction",
  "Principal": "sns.amazonaws.com",
  "SourceArn": "%{Source.Arn}"
}
```

`AWS::SQS::Queue` Da a `AWS::Lambda::Function`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::Lambda::Function` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sqs:DeleteMessage"
      ],
      "Resource": [
```

```
        "%{Source.Arn}"
      ]
    }
  ]
}
```

`Read`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sqs:ReceiveMessage",
        "sqs:GetQueueAttributes"
      ],
      "Resource": [
        "%{Source.Arn}"
      ]
    }
  ]
}
```

`AWS::S3::Bucket` Da a `AWS::Lambda::Function`

Tipo di politica

`AWS::Lambda::Permission`allegato al`AWS::Lambda::Function`.

Categorie di accesso

`Write`

```
{
  "Action": "lambda:InvokeFunction",
  "Principal": "s3.amazonaws.com",
  "SourceArn": "%{Source.Arn}",
  "SourceAccount": "${AWS::AccountId}"
}
```

`AWS::StepFunctions::StateMachine` Da a `AWS::Lambda::Function`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::StepFunctions::StateMachine` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeAsync",
        "lambda:InvokeFunction"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

`AWS::StepFunctions::StateMachine` Da a `AWS::SNS::Topic`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::StepFunctions::StateMachine` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
```

```
}
```

`AWS::StepFunctions::StateMachine` Da a `AWS::SQS::Queue`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::StepFunctions::StateMachine` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sqs:SendMessage"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

`AWS::StepFunctions::StateMachine` Da a `AWS::S3::Bucket`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::StepFunctions::StateMachine` ruolo.

Categorie di accesso

`Read`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
```

```
        "s3:GetObjectAcl",
        "s3:GetObjectLegalHold",
        "s3:GetObjectRetention",
        "s3:GetObjectTorrent",
        "s3:GetObjectVersion",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionForReplication",
        "s3:GetObjectVersionTorrent",
        "s3:ListBucket",
        "s3:ListBucketMultipartUploads",
        "s3:ListBucketVersions",
        "s3:ListMultipartUploadParts"
      ],
      "Resource": [
        "%{Destination.Arn}",
        "%{Destination.Arn}/*"
      ]
    }
  ]
}
```

Write

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:AbortMultipartUpload",
        "s3:DeleteObject",
        "s3:DeleteObjectVersion",
        "s3:PutObject",
        "s3:PutObjectLegalHold",
        "s3:PutObjectRetention",
        "s3:RestoreObject"
      ],
      "Resource": [
        "%{Destination.Arn}",
        "%{Destination.Arn}/*"
      ]
    }
  ]
}
```

`AWS::StepFunctions::StateMachine` Da a `AWS::DynamoDB::Table`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::StepFunctions::StateMachine` ruolo.

Categorie di accesso

`Read`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:Query",
        "dynamodb:Scan",
        "dynamodb:BatchGetItem",
        "dynamodb:ConditionCheckItem",
        "dynamodb:PartiQLSelect"
      ],
      "Resource": [
        "%{Destination.Arn}",
        "%{Destination.Arn}/index/*"
      ]
    }
  ]
}
```

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:UpdateItem",
        "dynamodb:DeleteItem",
        "dynamodb:BatchWriteItem",
        "dynamodb:PartiQLDelete",
        "dynamodb:PartiQLInsert",
        "dynamodb:PartiQLUpdate"
```

```
      ],
      "Resource": [
        "%{Destination.Arn}",
        "%{Destination.Arn}/index/*"
      ]
    }
  ]
}
```

`AWS::StepFunctions::StateMachine` Da a `AWS::StepFunctions::StateMachine`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::StepFunctions::StateMachine` ruolo.

Categorie di accesso

`Read`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "states:DescribeExecution"
      ],
      "Resource": [
        "arn:${AWS::Partition}:states:${AWS::Region}:${AWS::AccountId}:execution:
%{Destination.Name}:*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "events:DescribeRule"
      ],
      "Resource": [
        "arn:${AWS::Partition}:events:${AWS::Region}:${AWS::AccountId}:rule/
StepFunctionsGetEventsForStepFunctionsExecutionRule"
      ]
    }
  ]
}
```

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "states:StartExecution"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "states:StopExecution"
      ],
      "Resource": [
        "arn:${AWS::Partition}:states:${AWS::Region}:${AWS::AccountId}:execution:
%{Destination.Name}:*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "events:PutTargets",
        "events:PutRule"
      ],
      "Resource": [
        "arn:${AWS::Partition}:events:${AWS::Region}:${AWS::AccountId}:rule/
StepFunctionsGetEventsForStepFunctionsExecutionRule"
      ]
    }
  ]
}
```

`AWS::StepFunctions::StateMachine` Da a `AWS::Events::EventBus`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::StepFunctions::StateMachine` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutEvents"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

`AWS::AppSync::DataSource` Da a `AWS::DynamoDB::Table`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::AppSync::DataSource` ruolo.

Categorie di accesso

`Read`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:Query",
        "dynamodb:Scan",
        "dynamodb:BatchGetItem",
        "dynamodb:ConditionCheckItem",
        "dynamodb:PartiQLSelect"
      ],
      "Resource": [
        "%{Destination.Arn}",
        "%{Destination.Arn}/index/*"
      ]
    }
  ]
```

```
}
```

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:UpdateItem",
        "dynamodb:DeleteItem",
        "dynamodb:BatchWriteItem",
        "dynamodb:PartiQLDelete",
        "dynamodb:PartiQLInsert",
        "dynamodb:PartiQLUpdate"
      ],
      "Resource": [
        "%{Destination.Arn}",
        "%{Destination.Arn}/index/*"
      ]
    }
  ]
}
```

`AWS::AppSync::DataSource` Da a `AWS::Lambda::Function`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::AppSync::DataSource` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeAsync",
        "lambda:InvokeFunction"
      ],
```

```
      "Resource": [
        "%{Destination.Arn}",
        "%{Destination.Arn}:*"
      ]
    }
  ]
}
```

`AWS::AppSync::DataSource` Da a `AWS::Events::EventBus`

Tipo di politica

Politica gestita dal cliente allegata al `AWS::AppSync::DataSource` ruolo.

Categorie di accesso

`Write`

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutEvents"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

`AWS::AppSync::GraphQLApi` Da a `AWS::Lambda::Function`

Tipo di politica

`AWS::Lambda::Permission`allegato al`AWS::Lambda::Function`.

Categorie di accesso

`Write`

```
{
  "Action": "lambda:InvokeFunction",
```

```
  "Principal": "appsync.amazonaws.com",
  "SourceArn": "arn:${AWS::Partition}:appsync:${AWS::Region}:${AWS::AccountId}:apis/
%{Source.ResourceId}"
}
```

Installazione di Docker da utilizzare con AWS SAMCLI

Dockerè un'applicazione che esegue contenitori sulla macchina. WithDocker, AWS SAM può fornire un ambiente locale simile a AWS Lambda un contenitore per creare, testare ed eseguire il debug delle applicazioni serverless.

> **Note**
>
> Dockerè necessario solo per testare le applicazioni localmente e per creare pacchetti di distribuzione utilizzando l'`--use-container`opzione.

Argomenti

- Installazione di Docker
- Fasi successive

Installazione di Docker

Segui queste istruzioni per l'installazione Docker sul tuo sistema operativo.

Linux

Docker è disponibile su molti sistemi operativi diversi, incluse le più moderne distribuzioni Linux, come CentOSDebian, e. Ubuntu Per informazioni sull'installazione Docker su un particolare sistema operativo, consulta Get Docker sul sito Web di Docker Docs.

Per l'installazione Docker su Amazon Linux 2 o Amazon Linux 2023

1. Aggiorna i pacchetti installati e la cache dei pacchetti sulla tua istanza.

   ```
   $ sudo yum update -y
   ```

2. Installa il pacchetto Docker Community Edition più recente.

- Per Amazon Linux 2, esegui quanto segue:

```
$ sudo amazon-linux-extras install docker
```

- Per Amazon Linux 2023, esegui quanto segue:

```
$ sudo yum install -y docker
```

3. Avviare il servizio Docker.

```
$ sudo service docker start
```

4. Aggiungi il `ec2-user` file al `docker` gruppo in modo da poter eseguire Docker i comandi senza utilizzarlisudo.

```
$ sudo usermod -a -G docker ec2-user
```

5. Ottieni le nuove autorizzazioni di `docker` gruppo disconnettendoti e riconnetterti. Per fare ciò, chiudi la finestra corrente del terminale SSH e riconnettiti all'istanza in una nuova. La tua nuova sessione SSH dovrebbe avere le autorizzazioni di gruppo appropriate`docker`.
6. Verifica che `ec2-user` possa eseguire i comandi Docker senza utilizzarli. sudo

```
$ docker ps
```

Dovresti vedere il seguente output, che conferma che Docker è installato e in esecuzione:

```
CONTAINER ID        IMAGE               COMMAND             CREATED
STATUS              PORTS               NAMES
```

> **Note**
>
> Su Linux, per creare ed eseguire funzioni Lambda con un'architettura di set di istruzioni diversa rispetto alla macchina host, è necessario configurare ulteriori passaggi. Docker Ad esempio, per eseguire `arm64` funzioni su una `x86_64` macchina, puoi eseguire il seguente comando per configurare il Docker demone:. `docker run --rm --privileged multiarch/qemu-user-static --reset -p yes`

Se riscontri problemi di installazioneDocker, vedi. Risoluzione degli errori di installazione In alternativa, consulta la sezione Risoluzione dei problemi della procedura successiva all'installazione per Linux sul sito Web Docker Docs.

macOS

> **Note**
>
> DockerIl desktop è ufficialmente supportato, ma a partire dalla AWS SAMCLI versione 1.47.0, puoi utilizzare alternative purché utilizzino il runtime. Docker

1. Installazione di Docker

   AWS SAMCLISupporta l'Dockeresecuzione su macOS Sierra 10.12 o versioni successive. Per informazioni sull'installazioneDocker, consulta Install Docker Desktop for Mac sul sito Web di Docker Docs.
2. Configura le unità condivise

   AWS SAMCLIRichiede che la directory del progetto, o qualsiasi directory principale, sia elencata in un'unità condivisa. Ciò potrebbe richiedere la condivisione di file. Per ulteriori informazioni, consulta l'argomento sulla risoluzione dei problemi relativi al montaggio di volumi che richiede la condivisione dei file nella Dockerdocumentazione.
3. Verifica dell'installazione

   Dopo Docker l'installazione, verifica che funzioni. Verifica inoltre che sia possibile eseguire Docker comandi dalla riga di comando (ad esempio,`docker ps`). Non è necessario installare, recuperare o scaricare alcun contenitore: lo AWS SAMCLI fa automaticamente come richiesto.

In caso di problemi durante l'installazioneDocker, per ulteriori suggerimenti sulla risoluzione dei problemi, consulta la sezione Risoluzione dei problemi e diagnosi del sito Web di Docs. Docker

Windows

> **Note**
>
> AWS SAM supporta ufficialmente Desktop. Docker Tuttavia, a partire dalla AWS SAMCLI versione 1.47.0, è possibile utilizzare alternative purché utilizzino il Docker runtime.

1. Installare Docker.

   DockerDesktop supporta il sistema operativo Windows più recente. Per le versioni precedenti di Windows, è disponibile il Docker Toolbox. Scegli la tua versione di Windows per i passaggi di Docker installazione corretti:

   - Per l'installazione Docker per Windows 10, vedi Installare Docker Desktop per Windows sul sito Web di Docker Docs.
   - Per l'installazione Docker per le versioni precedenti di Windows, consulta The Docker Toolbox nell'archivio Docker GitHub Toolbox.

2. Configura le tue unità condivise.

   AWS SAMCLIRichiede che la directory del progetto, o qualsiasi directory principale, sia elencata in un'unità condivisa. In alcuni casi, è necessario condividere l'unità Docker per funzionare correttamente.

3. Verifica l'installazione.

   Dopo Docker l'installazione, verifica che funzioni. Verifica inoltre che sia possibile eseguire Docker comandi dalla riga di comando (ad esempio,docker ps). Non è necessario installare, recuperare o scaricare alcun contenitore: lo AWS SAMCLI fa automaticamente come richiesto.

In caso di problemi durante l'installazioneDocker, per ulteriori suggerimenti sulla risoluzione dei problemi, consulta la sezione Risoluzione dei problemi e diagnosi del sito Web di Docs. Docker

Fasi successive

Per informazioni su come installare, consulta. AWS SAMCLI Installa il AWS SAMCLI

Installazione Finch per l'uso con la AWS SAM CLI

Finch AWSè lo strumento di sviluppo di container che fornisce un'alternativaDocker, particolarmente utile, sui sistemi macOS e Linux. Per ulteriori informazioni, vedere Finch on. GitHub ConFinch, AWS SAM può fornire un ambiente locale simile a AWS Lambda un contenitore per creare, testare ed eseguire il debug delle applicazioni serverless.

> **Note**
>
> È necessario testare Finch le applicazioni localmente, creare pacchetti di distribuzione utilizzando l'`--use-container`opzione e creare funzioni di immagine OCI. A partire dalla versione AWS SAM CLI 1.145.0, AWS SAM CLI Finch supporta come alternativa il runtime del contenitore a. Docker

Argomenti

- Installazione di Finch
- Configurazione Finch con CLI AWS SAM
- Verifica dell'installazione
- Interrompere Finch
- Risoluzione dei problemi
- Fasi successive

Installazione di Finch

Segui queste istruzioni per l'installazione Finch sul tuo sistema operativo.

Prerequisiti

> **Note**
>
> Finchl'integrazione nella AWS SAM CLI è disponibile su sistemi macOS e Linux. Per gli utenti Windows, valuta la possibilità di utilizzare una macchina virtuale o un ambiente Linux per Finch motivi di funzionalità.

Versioni macOS:

- macOS 15 Sequoia
- macOS 14 Sonoma

> **Note**
>
> Finchsupporta le ultime due versioni di macOS.

Linux:

- Linux Kernel v4.x+

Installazione di Finch

macOS

Per installare Finch su macOS

1. Installa Finch utilizzando la `brew` CLI. Se richiesto, inserisci la password di macOS.

```
$ brew install finch
```

2. Controlla la versione installata. Finch

```
$ finch --version
```

3. Inizializza la macchina Finch virtuale (prima configurazione).

```
$ finch vm init
```

4. Avvia la macchina Finch virtuale.

```
$ finch vm start
```

5. Controlla lo stato della macchina Finch virtuale.

```
$ finch vm status
```

6. Verifica che Finch funzioni correttamente.

```
$ finch --info
$ finch ps
```

Per ulteriori informazioni, consulta Installazione di Finch su macOS.

Linux

Puoi eseguire l'installazione Finch su Amazon Linux, Ubuntu o utilizzando un metodo di installazione generico. Per ulteriori informazioni, consulta Installazione di Finch su Linux.

Configurazione Finch con CLI AWS SAM

Dopo l'installazioneFinch, la AWS SAM CLI rileva e utilizza automaticamente Finch come runtime del contenitore quando Docker non viene eseguita.

> **Note**
>
> Finchfunge da riserva a. Docker Se le hai entrambe Finch installate Docker e funzionanti, la AWS SAM CLI dà la priorità Docker per impostazione predefinita.

Configurazione delle preferenze dell'amministratore (macOS)

È possibile configurare la AWS SAM CLI per utilizzare un runtime di contenitore specifico come predefinito creando un file delle preferenze dell'amministratore. Questa preferenza a livello di sistema sostituisce il comportamento di fallback predefinito.

Opzioni di configurazione disponibili

- Imposta Finch come runtime preferito del contenitore:

```
$ sudo /usr/libexec/PlistBuddy -c "Add :DefaultContainerRuntime string finch" /
Library/Preferences/com.amazon.samcli.plist
```

- Imposta Docker come runtime preferito del contenitore:

```
$ sudo /usr/libexec/PlistBuddy -c "Add :DefaultContainerRuntime string docker" /
Library/Preferences/com.amazon.samcli.plist
```

- Rimuovi preferenza (ritorno al comportamento predefinito):

```
$ sudo /usr/libexec/PlistBuddy -c "Remove :DefaultContainerRuntime" /Library/
Preferences/com.amazon.samcli.plist
```

> **Note**
>
> Quando non è impostata alcuna preferenza dell'amministratore, la AWS SAM CLI utilizza Docker per impostazione predefinita quando disponibile, Finch mentre come riserva quando non Docker è in esecuzione.

Differenze di montaggio delle directory (macOS)

Su macOS, i comandi AWS SAM locali falliscono se localizzi il progetto fuori dalla tua home directory (~) o. `/Volumes` Questo design limita il montaggio automatico delle directory per motivi di sicurezza a tua insaputa.

Opzioni di mitigazione

- Opzione 1: apri la configurazione di Finch in un editor di testo `~/.finch/finch.yaml` e aggiungi il percorso del progetto nella `additional_directories` sezione, quindi riavvia la Finch macchina virtuale. Per ulteriori informazioni, consultate la documentazione di Finch.

```
$ finch vm stop
$ finch vm start
```

- Opzione 2: spostate il progetto nella vostra home directory.

Verifica dell'installazione

Dopo l'installazione e l'avvioFinch, verifica che funzioni con la AWS SAM CLI.

Verifica di base

Esegui il comando seguente per verificare che la AWS SAM CLI possa utilizzare: Finch

```
$ sam build --use-container
```

Dovresti vedere l'uso della AWS SAM CLI Finch per creare la tua applicazione in un ambiente container.

> **Note**
>
> Finchfunge da riserva a. Docker Se Docker è in esecuzione, la AWS SAM CLI assegnerà la priorità Docker per impostazione predefinita.

Test Finch come fallback

Per eseguire Finch il test come fallback quando Docker è in esecuzione, devi prima Docker fermarti.

Per testare la funzionalità di Finch fallback

1. Smetti Docker di usare i comandi appropriati per il tuo sistema operativo.

   macOS:

   ```
   $ docker stop $(docker ps -q)
   $ pkill -f "Docker Desktop"
   ```

   DockerLa verifica è stata interrotta.

   ```
   $ docker ps
   ```

   Linux:

   ```
   $ docker stop $(docker ps -q)
   $ sudo systemctl stop docker
   ```

   DockerLa verifica è stata interrotta.

   ```
   $ sudo systemctl status docker
   ```

2. Esegui il `sam build --use-container` comando per verificare che la AWS SAM CLI venga utilizzata automaticamente Finch come fallback.

Interrompere Finch

Se devi fermartiFinch, usa i seguenti comandi:

macOS:

```
$ finch vm stop
$ sudo pkill -f finch
```

FinchLa verifica è stata interrotta.

```
$ finch vm status
```

Linux:

```
$ sudo systemctl stop finch
$ sudo pkill -f finch
```

FinchLa verifica è stata interrotta.

```
$ sudo systemctl status finch
```

Risoluzione dei problemi

Se riscontri problemi conFinch:

- Verifica che il tuo sistema operativo soddisfi i prerequisiti.
- Verifica di aver abilitato la virtualizzazione sul tuo sistema.
- Per macOS, verifica di disporre di spazio su disco sufficiente per la macchina virtuale.
- Per Linux, verifica di disporre delle autorizzazioni appropriate per eseguire `systemctl` i comandi.
- Se riscontri errori di CloudFormation distribuzione con errori «tipo di supporto non supportato» quando invii immagini multiarchitettura ad Amazon Elastic Container Registry (Amazon ECR), crea un singolo elemento di immagine anziché utilizzare immagini multiarchitettura. Ciò si verifica perché Finch invia sia l'indice dell'immagine che l'immagine ad Amazon ECR, mentre Docker invia solo l'immagine.
- Se la creazione multipiattaforma fallisce quando l'architettura della funzione di destinazione non corrisponde all'architettura della macchina host su Linux, abilita l'emulazione multipiattaforma eseguendo. `sudo finch run --privileged --rm tonistiigi/binfmt:master --install all`

Per ulteriori informazioni sulla risoluzione dei problemi, consulta la documentazione di Finch o il. AWS SAMCLIrisoluzione dei problemi

Fasi successive

Ora che hai Finch installato, puoi usarlo con la AWS SAM CLI per:

- Crea applicazioni serverless utilizzando contenitori con. `sam build --use-container` Per ulteriori informazioni, consulta the section called “sam build”.
- Testa le funzioni Lambda localmente con. `sam local invoke` Per ulteriori informazioni, consulta the section called “sam local invoke”.
- Avvia un API Gateway locale con`sam local start-api`. Per ulteriori informazioni, consulta the section called “sam local start-api”.
- Avvia un endpoint Lambda locale con. `sam local start-lambda` Per ulteriori informazioni, consulta the section called “sam local start-lambda”.

Per ulteriori informazioni sull'utilizzo della AWS SAM CLI con i contenitori, consulta. Compilazione predefinita con AWS SAM

Archivi di immagini per AWS SAM

AWS SAM semplifica le attività di integrazione continua e distribuzione continua (CI/CD) per applicazioni serverless con l'aiuto della creazione di immagini di container. Le immagini AWS SAM fornite includono l'interfaccia a riga di AWS SAM comando (CLI) e gli strumenti di compilazione per una serie di runtime supportati AWS Lambda . Ciò semplifica la creazione e il confezionamento di applicazioni serverless utilizzando. AWS SAMCLI È possibile utilizzare queste immagini con CI/CD i sistemi per automatizzare la creazione e la distribuzione di AWS SAM applicazioni. Per alcuni esempi, consulta Implementa con CI/CD sistemi e pipeline.

AWS SAM URIs le immagini del contenitore di build sono contrassegnate con la versione AWS SAMCLI inclusa in quell'immagine. Se si specifica l'URI senza tag, viene utilizzata la versione più recente. Ad esempio, `public.ecr.aws/sam/build-nodejs20.x` utilizza l'immagine più recente. Tuttavia, `public.ecr.aws/sam/build-nodejs20.x:1.24.1` utilizza l'immagine contenente la versione AWS SAM CLI 1.24.1.

A partire dalla versione 1.33.0 di AWS SAMCLI, entrambe le immagini `x86_64` e quelle del `arm64` contenitore sono disponibili per i runtime supportati. Per ulteriori informazioni, consulta Lambda runtimes nella Developer Guide.AWS Lambda

> **Note**
>
> Prima della versione 1.22.0 di AWS SAMCLI, DockerHub era il repository predefinito da cui AWS SAMCLI estraevano l'immagine del contenitore. A partire dalla versione 1.22.0, l'archivio predefinito è cambiato in Amazon Elastic Container Registry Public (Amazon ECR Public). Per estrarre l'immagine di un contenitore da un repository diverso da quello predefinito corrente, puoi utilizzare il comando con l'sam buildopzione. --build-image Gli esempi alla fine di questo argomento mostrano come creare applicazioni utilizzando immagini di DockerHub repository.

Archivio di immagini URIs

La tabella seguente elenca le immagini URIs dei container di build Amazon ECR Public che puoi usare per creare e impacchettare applicazioni serverless. AWS SAM

> **Note**
>
> Amazon ECR Public è stato sostituito DockerHub a partire dalla AWS SAMCLI versione 1.22.0. Se utilizzi una versione precedente di AWS SAMCLI, ti consigliamo di effettuare l'aggiornamento.

Runtime	Amazon ECR Public
Runtime personalizzato (AL2023)	public.ecr. aws/sam/build- fornito dal 2023
Runtime personalizzato (AL2)	public.ecr. aws/sam/build- fornito.al2
Runtime personalizzato	pubblico.ecr. aws/sam/build-fornito
Java 25	pubblico.ecr. aws/sam/build-java25
Java 21	public.ecr. aws/sam/build-java21
Java 17	public.ecr. aws/sam/build-java17
Java 11	public.ecr. aws/sam/build-java11

Runtime	Amazon ECR Public
Java 8	public.ecr. aws/sam/build-java 8
.NET 9	pubblico.ecr. aws/sam/build-dotnet9
.NET 8	public.ecr. aws/sam/build-dotnet8
.NET 7	public.ecr. aws/sam/build-dotnet7
.NET 6	public.ecr. aws/sam/build-dotnet6
Node.js 24	pubblico.ecr. aws/sam/build-nodejs24.x
Node.js 22	public.ecr. aws/sam/build-nodejs 22.x
Node.js 20	public.ecr. aws/sam/build-nodejs 20.x
Node.js 18	public.ecr. aws/sam/build-nodejs 18.x
Node.js 16	public.ecr. aws/sam/build-nodejs 16.x
Python 3.14	public.ecr. aws/sam/build-python3.14
Python 3.13	public.ecr. aws/sam/build-python3.13
Python 3.12	public.ecr. aws/sam/build-python3.12
Python 3.11	pubblico.ecr. aws/sam/build-python3.11
Python 3.10	pubblico.ecr. aws/sam/build-python 3.10
Python 3.9	pubblico.ecr. aws/sam/build-python3.9
Python 3.8	pubblico.ecr. aws/sam/build-python3.8
Rubino 3.4	public.ecr. aws/sam/build- rubino 3.4
Ruby 3.3	public.ecr. aws/sam/build- ruby 3.3
Ruby 3.2	public.ecr. aws/sam/build- rubino 3.2

Esempi

I due comandi di esempio seguenti creano applicazioni utilizzando immagini di contenitori dal repository di immagini:

Crea un'Node.js 24applicazione utilizzando un'immagine del contenitore estratta da Amazon ECR:

```
$ sam build --use-container --build-image public.ecr.aws/sam/build-nodejs24.x
```

Crea una risorsa funzionale utilizzando l'immagine del Python 3.14 contenitore estratta da Amazon ECR:

```
$ sam build --use-container --build-image Function1=public.ecr.aws/sam/build-python3.14
```

Implementazione graduale di applicazioni serverless con AWS SAM

AWS Serverless Application Model (AWS SAM) è integrato CodeDeployper fornire implementazioni graduali AWS Lambda . Con solo poche righe di configurazione, AWS SAM esegue le seguenti operazioni per te:

- Implementa nuove versioni della funzione Lambda e crea automaticamente alias che rimandano alla nuova versione.
- Sposta gradualmente il traffico dei clienti verso la nuova versione fino a quando non sei sicuro che funzioni come previsto. Se un aggiornamento non funziona correttamente, puoi ripristinare le modifiche.
- Definisce le funzioni di test precedenti e successivi al traffico per verificare che il codice appena distribuito sia configurato correttamente e che l'applicazione funzioni come previsto.
- Ripristina automaticamente la distribuzione se vengono attivati degli CloudWatch allarmi.

> **Note**
>
> Se abiliti le distribuzioni graduali tramite il tuo AWS SAM modello, viene creata automaticamente una CodeDeploy risorsa per te. È possibile visualizzare la CodeDeploy risorsa direttamente tramite. Console di gestione AWS

Esempio

L'esempio seguente dimostra l'utilizzo della funzione CodeDeploy Lambda per spostare gradualmente i clienti verso la versione appena distribuita della funzione Lambda:

```
Resources:
MyLambdaFunction:
  Type: AWS::Serverless::Function
  Properties:
    Handler: index.handler
    Runtime: nodejs20.x
    CodeUri: s3://bucket/code.zip

    AutoPublishAlias: live

    DeploymentPreference:
      Type: Canary10Percent10Minutes
      Alarms:
        # A list of alarms that you want to monitor
        - !Ref AliasErrorMetricGreaterThanZeroAlarm
        - !Ref LatestVersionErrorMetricGreaterThanZeroAlarm
      Hooks:
        # Validation Lambda functions that are run before & after traffic shifting
        PreTraffic: !Ref PreTrafficLambdaFunction
        PostTraffic: !Ref PostTrafficLambdaFunction
```

Queste revisioni del AWS SAM modello eseguono le seguenti operazioni:

- `AutoPublishAlias`: Aggiungendo questa proprietà e specificando un nome alias,: AWS SAM
  - Rileva quando viene distribuito un nuovo codice, in base alle modifiche all'URI Amazon S3 della funzione Lambda.
  - Crea e pubblica una versione aggiornata di quella funzione con il codice più recente.
  - Crea un alias con un nome fornito dall'utente (a meno che non esista già un alias) e punta alla versione aggiornata della funzione Lambda. Per sfruttare questa funzionalità, le invocazioni della funzione devono utilizzare il qualificatore dell'alias. Se non conosci il controllo delle versioni e gli alias delle funzioni Lambda, consulta Versionamento e alias delle AWS Lambda funzioni.
- `Deployment Preference Type`: Nell'esempio precedente, il 10% del traffico dei clienti viene immediatamente spostato sulla nuova versione. Dopo 10 minuti, tutto il traffico viene spostato sulla nuova versione. Tuttavia, se i test pre-traffico o post-traffico falliscono o se viene attivato un

CloudWatch allarme, ripristina la distribuzione CodeDeploy . Puoi specificare in che modo il traffico deve essere spostato tra le versioni nei seguenti modi:

- `Canary`: il traffico viene trasferito in due incrementi. Puoi scegliere tra opzioni canarie predefinite. Le opzioni specificano la percentuale di traffico che viene trasferita alla versione aggiornata della funzione Lambda nel primo incremento e l'intervallo, in minuti, prima che il traffico rimanente venga spostato nel secondo incremento.
- `Linear`: il traffico viene trasferito in incrementi uguali con lo stesso intervallo di tempo, in minuti, tra ciascun incremento. Puoi scegliere tra opzioni lineari predefinite che specificano la percentuale di traffico che viene spostata in ogni incremento e il numero di minuti tra ogni incremento.
- `AllAtOnce`: tutto il traffico viene spostato contemporaneamente dalla funzione Lambda originale alla versione aggiornata della funzione Lambda.

La tabella seguente descrive altre opzioni di spostamento del traffico disponibili oltre a quella utilizzata nell'esempio.

Tipo di distribuzione di preferenza
Canary10Percent30Minutes
Canary10Percent5Minutes
Canary10Percent10Minutes
Canary10Percent15Minutes
PercentEveryLineare 10 10 minuti
Lineare PercentEvery 10 1 minuto
Lineare 10 PercentEvery 2 minuti
Lineare 10 PercentEvery 3 minuti
AllAtOnce

- `Alarms`: Si tratta di CloudWatch allarmi che vengono attivati da eventuali errori generati dalla distribuzione. Quando vengono rilevati, ripristinano automaticamente la distribuzione. Ad esempio, se il codice aggiornato che stai distribuendo causa errori all'interno dell'applicazione. Un altro

esempio è se alcune AWS Lambda CloudWatch metriche personalizzate che hai specificato hanno superato la soglia di allarme.

- `Hooks`: Si tratta di funzioni di test prima e dopo il traffico che eseguono controlli prima che il cambio di traffico inizi a passare alla nuova versione e dopo il completamento del cambio di traffico.
  - `PreTraffic`: prima che inizi lo spostamento del traffico, CodeDeploy richiama la funzione Lambda pre-traffic hook. Questa funzione Lambda deve richiamare CodeDeploy e indicare l'esito positivo o negativo. Se la funzione fallisce, si interrompe e riporta un errore a. CloudFormation Se la funzione ha esito positivo, si CodeDeploy procede allo spostamento del traffico.
  - `PostTraffic`: una volta completato lo spostamento del traffico, CodeDeploy richiama la funzione Lambda post-traffic hook. È simile al pre-traffic hook, in cui la funzione deve richiamare a per segnalare CodeDeploy un esito positivo o negativo. Utilizzare gli hook post-traffico per eseguire i test di integrazione o altre operazioni di convalida.

Per ulteriori informazioni, consultare la documentazione di riferimento delle SAM per distribuzioni sicure.

Implementazione graduale di una funzione Lambda per la prima volta

Quando si implementa gradualmente una funzione Lambda CodeDeploy , è necessaria una versione della funzione precedentemente distribuita da cui spostare il traffico. Pertanto, la prima implementazione deve essere eseguita in due fasi:

- Passaggio 1: implementa la funzione Lambda e crea automaticamente alias con. `AutoPublishAlias`
- Fase 2: Esegui la distribuzione graduale con. `DeploymentPreference`

Eseguendo la prima implementazione graduale in due fasi si ottiene CodeDeploy una versione precedente della funzione Lambda da cui spostare il traffico.

Fase 1: Implementa la funzione Lambda

```
Resources:
MyLambdaFunction:
  Type: AWS::Serverless::Function
  Properties:
    Handler: index.handler
    Runtime: nodejs20.x
```

```
    CodeUri: s3://bucket/code.zip

    AutoPublishAlias: live
```

Fase 2: Esegui la distribuzione graduale

```
Resources:
MyLambdaFunction:
  Type: AWS::Serverless::Function
  Properties:
    Handler: index.handler
    Runtime: nodejs20.x
    CodeUri: s3://bucket/code.zip

    AutoPublishAlias: live

    DeploymentPreference:
      Type: Canary10Percent10Minutes
      Alarms:
        # A list of alarms that you want to monitor
        - !Ref AliasErrorMetricGreaterThanZeroAlarm
        - !Ref LatestVersionErrorMetricGreaterThanZeroAlarm
      Hooks:
        # Validation Lambda functions that are run before and after traffic shifting
        PreTraffic: !Ref PreTrafficLambdaFunction
        PostTraffic: !Ref PostTrafficLambdaFunction
```

Ulteriori informazioni

Per un esempio pratico di configurazione di una distribuzione graduale, consulta il Modulo 5 - Canary Deployments in The Complete Workshop. AWS SAM

Note di riferimento importanti per AWS SAM

Questa sezione contiene note e annunci importanti per AWS Serverless Application Model ()AWS SAM.

Argomenti

- Note importanti per il 2023

Note importanti per il 2023

Ottobre 2023

AWS SAMCLIinterruzione del supporto per Python 3.7

Pubblicato il 2023-10-20

Python 3.7ha ricevuto end-of-life lo status nel giugno del 2023. Il supporto AWS SAM CLI interromperà Python 3.7 il 24 ottobre 2023. Per ulteriori informazioni, consulta l'annuncio pubblicato nel aws-sam-cli GitHubrepository.

Questa modifica riguarda i seguenti utenti:

- Se si utilizza Python 3.7 e si installa il AWS SAM CLI throughpip.
- Se lo usi `aws-sam-cli` come libreria e crei la tua applicazione conPython 3.7.

Se la installate e la AWS SAM CLI gestite con un altro metodo, non ne risentite.

Agli utenti interessati, consigliamo di aggiornare l'ambiente di sviluppo a Python 3.8 o a una versione più recente.

Questa modifica non influisce sul supporto per l'ambiente Python 3.7 AWS Lambda di runtime. Per ulteriori informazioni, consulta la politica di deprecazione del Runtime nella Developer Guide.AWS Lambda

AWS SAMCLITerraformsupporto

Questa sezione descrive l'utilizzo dell'interfaccia a riga di AWS Serverless Application Model comando (AWS SAMCLI) con i tuoi Terraform progetti e Terraform Cloud.

Per fornire feedback e inviare richieste di funzionalità, crea un GitHubproblema.

Argomenti

- Guida introduttiva Terraform al supporto per AWS SAMCLI
- Utilizzo di AWS SAMCLI with Terraform per il debug e il test locali
- Utilizzo di AWS SAMCLI with Serverless.tf per il debug e il test locali
- AWS SAMCLIcon riferimento Terraform
- A cosa serve AWS SAMCLI il supportoTerraform?

Guida introduttiva Terraform al supporto per AWS SAMCLI

Questo argomento spiega come iniziare a utilizzare l'interfaccia a riga di AWS Serverless Application Model comando (AWS SAMCLI) conTerraform.

Per fornire feedback e inviare richieste di funzionalità, crea un GitHubproblema.

Argomenti

- AWS SAMCLITerraformprerequisiti
- Utilizzo dei AWS SAMCLI comandi con Terraform
- Configurazione per Terraform progetti
- Configurazione di Terraform Cloud

AWS SAMCLITerraformprerequisiti

Completa tutti i prerequisiti per iniziare a utilizzarlo AWS SAMCLI con i tuoi Terraform progetti.

1. Installa o aggiorna il AWS SAMCLI

   Per verificare se è AWS SAMCLI installato, esegui quanto segue:

```
$ sam --version
```

Se AWS SAMCLI è già installato, l'output mostrerà una versione. Per eseguire l'aggiornamento alla versione più recente, vedereAggiornamento del AWS SAMCLI.

Per istruzioni sull'installazione di AWS SAMCLI e tutti i relativi prerequisiti, vedere. Installa il AWS SAMCLI

2. Installazione di Terraform

   Per verificare se l'Terraforminstallazione è stata eseguita, esegui quanto segue:

```
$ terraform -version
```

   Per l'installazioneTerraform, consulta Installa Terraform nel Terraformregistro.

3. Installa Docker per il test locale

   I AWS SAMCLI requisiti Docker per i test locali. Per l'installazioneDocker, vedereInstallazione di Docker da utilizzare con AWS SAMCLI.

Utilizzo dei AWS SAMCLI comandi con Terraform

Quando esegui un AWS SAMCLI comando supportato, usa l'`--hook-name`opzione e fornisci il `terraform` valore. Di seguito è riportato un esempio:

```
$ sam local invoke --hook-name terraform
```

Puoi configurare questa opzione nel tuo file AWS SAMCLI di configurazione con quanto segue:

```
hook_name = "terraform"
```

Configurazione per Terraform progetti

Completa i passaggi descritti in questo argomento per utilizzarlo AWS SAMCLI con i Terraform progetti.

Non è richiesta alcuna configurazione aggiuntiva se costruite gli AWS Lambda artefatti all'esterno del Terraform progetto. Vedi Utilizzo di AWS SAMCLI with Terraform per il debug e il test locali per iniziare a usare. AWS SAMCLI

Se costruisci i tuoi artefatti Lambda all'interno Terraform dei tuoi progetti, devi fare quanto segue:

1. Installa 3.8 Python o versione successiva
2. Installa lo strumento. Make
3. Definisci la logica di costruzione degli artefatti Lambda all'interno del tuo progetto. Terraform
4. Definisci una `sam metadata` risorsa per informarli sulla tua logica AWS SAMCLI di compilazione.
5. Usa il AWS SAMCLI `sam build` comando per creare i tuoi artefatti Lambda.

Installa 3.8 Python o versione successiva

Python3.8 o versione successiva è richiesta per l'utilizzo con. AWS SAMCLI Quando eseguisam `build`, le AWS SAMCLI creazioni `makefiles` che contengono Python comandi per creare i tuoi artefatti Lambda.

Per istruzioni di installazione, consulta Downloading Python in Python's Beginners Guide.

Verifica che Python 3.8 o versione successiva sia aggiunto al percorso del tuo computer eseguendo:

```
$ python --version
```

L'output dovrebbe mostrare una versione di Python 3.8 o successiva.

Installa lo strumento Make

GNUMakeè uno strumento che controlla la generazione di eseguibili e altri file non sorgente per il progetto. Le AWS SAMCLI creazioni `makefiles` che si affidano a questo strumento per creare i tuoi artefatti Lambda.

Se non lo avete Make installato sul computer locale, installatelo prima di procedere.

Per Windows, puoi installarlo usando Chocolatey. Per istruzioni, vedi Uso di Chocolatey in Come installare e usare «Make» in Windows

Definisci la logica di compilazione degli artefatti Lambda

Usa il tipo di `null_resource` Terraform risorsa per definire la logica di compilazione Lambda. Di seguito è riportato un esempio che utilizza uno script di compilazione personalizzato per creare una funzione Lambda.

```
resource "null_resource" "build_lambda_function" {
    triggers = {
        build_number = "${timestamp()}"
    }

    provisioner "local-exec" {
        command =  substr(pathexpand("~"), 0, 1) == "/"? "./
py_build.sh \"${local.lambda_src_path}\" \"${local.building_path}\"
 \"${local.lambda_code_filename}\" Function" : "powershell.exe -File .\\PyBuild.ps1
 ${local.lambda_src_path} ${local.building_path} ${local.lambda_code_filename}
 Function"
    }
}
```

Definire una risorsa sam metadata

La `sam metadata` risorsa è un tipo di `null_resource` Terraform risorsa che fornisce AWS SAMCLI le informazioni necessarie per localizzare gli artefatti Lambda. È necessaria una `sam metadata` risorsa unica per ogni funzione o layer Lambda del progetto. Per ulteriori informazioni su questo tipo di risorsa, consulta null_resource nel registro. Terraform

Per definire una risorsa sam metadata

1. Assegna un nome `sam_metadata_` alla risorsa iniziando con per identificare la risorsa come sam metadata risorsa.
2. Definisci le proprietà degli artefatti Lambda all'interno del `triggers` blocco della risorsa.
3. Specificate il `null_resource` che contiene la logica di compilazione Lambda con l'`depends_on`argomento.

   Di seguito è riportato un modello di esempio:

   ```
   resource "null_resource" "sam_metadata_..." {
     triggers = {
       resource_name = resource_name
       resource_type = resource_type
   ```

```
    original_source_code = original_source_code
    built_output_path = built_output_path
  }
  depends_on = [
    null_resource.build_lambda_function # ref to your build logic
  ]
}
```

Di seguito è riportato un esempio di `sam metadata` risorsa:

```
resource "null_resource" "sam_metadata_aws_lambda_function_publish_book_review" {
    triggers = {
        resource_name = "aws_lambda_function.publish_book_review"
        resource_type = "ZIP_LAMBDA_FUNCTION"
        original_source_code = "${local.lambda_src_path}"
        built_output_path = "${local.building_path}/${local.lambda_code_filename}"
    }
    depends_on = [
        null_resource.build_lambda_function
    ]
}
```

Il contenuto della `sam metadata` risorsa varierà in base al tipo di risorsa Lambda (funzione o layer) e al tipo di pacchetto (ZIP o immagine). Per ulteriori informazioni, oltre ad esempi, consultarisorsa di metadati sam.

Quando configuri una `sam metadata` risorsa e utilizzi un AWS SAMCLI comando supportato, AWS SAMCLI genererà il file di metadati prima di eseguire il AWS SAMCLI comando. Dopo aver generato questo file, puoi utilizzare l'`--skip-prepare-infra`opzione con AWS SAMCLI i comandi futuri per saltare il processo di generazione dei metadati e risparmiare tempo. Questa opzione deve essere utilizzata solo se non hai apportato modifiche all'infrastruttura, come la creazione di nuove funzioni Lambda o nuovi endpoint API.

Usa il AWS SAMCLI per costruire i tuoi artefatti Lambda

Usa il AWS SAMCLI `sam build` comando per creare i tuoi artefatti Lambda. Quando eseguisam `build`, AWS SAMCLI esegue le seguenti operazioni:

1. Cerca `sam metadata` risorse nel tuo Terraform progetto per conoscere e localizzare le tue risorse Lambda.

2. Avvia la logica di compilazione Lambda per creare gli artefatti Lambda.
3. Crea una `.aws-sam` directory che organizza il Terraform progetto da utilizzare con i comandi. AWS SAMCLI `sam local`

Per creare con sam build

1. Dalla directory contenente il modulo Terraform root, esegui quanto segue:

```
$ sam build --hook-name terraform
```

2. Per creare una funzione o un layer Lambda specifico, esegui quanto segue

```
$ sam build --hook-name terraform lambda-resource-id
```

   L'ID della risorsa Lambda può essere il nome della funzione Lambda o l'indirizzo completo della Terraform risorsa, ad esempio o. `aws_lambda_function.list_books` `module.list_book_function.aws_lambda_function.this[0]`

Se il codice sorgente della funzione o altri file di Terraform configurazione si trovano all'esterno della directory contenente il modulo Terraform root, è necessario specificare la posizione. Utilizzate l'`--terraform-project-root-path`opzione per specificare il percorso assoluto o relativo della directory di primo livello contenente questi file. Di seguito è riportato un esempio:

```
$ sam build --hook-name terraform --terraform-project-root-path ~/projects/terraform/
demo
```

Crea usando un contenitore

Quando si esegue il AWS SAMCLI `sam build` comando, è possibile configurarlo AWS SAMCLI per creare l'applicazione utilizzando un Docker contenitore locale.

> **Note**
>
> È necessario averlo Docker installato e configurato. Per istruzioni, consulta Installazione di Docker da utilizzare con AWS SAMCLI.

Per creare utilizzando un contenitore

1. Crea un `Dockerfile` file che contenga gli Make strumenti TerraformPython, e. Dovresti includere anche il runtime della funzione Lambda.

   Di seguito è riportato un esempio`Dockerfile`:

```
FROM public.ecr.aws/amazonlinux/amazonlinux:2

RUN yum -y update \
    && yum install -y unzip tar gzip bzip2-devel ed gcc gcc-c++ gcc-gfortran \
    less libcurl-devel openssl openssl-devel readline-devel xz-devel \
    zlib-devel glibc-static libcxx libcxx-devel llvm-toolset-7 zlib-static \
    && rm -rf /var/cache/yum

RUN yum -y install make \
    && yum -y install zip

RUN yum install -y yum-utils \
    && yum-config-manager --add-repo https://rpm.releases.hashicorp.com/
AmazonLinux/hashicorp.repo \
    && yum -y install terraform \
    && terraform --version

# AWS Lambda Builders
RUN amazon-linux-extras enable python3.8
RUN yum clean metadata && yum -y install python3.8
RUN curl -L get-pip.io | python3.8
RUN pip3 install aws-lambda-builders
RUN ln -s /usr/bin/python3.8 /usr/bin/python3
RUN python3 --version

VOLUME /project
WORKDIR /project

ENTRYPOINT ["sh"]
```

2. docker buildUsalo per creare la tua Docker immagine.

   Di seguito è riportato un esempio:

```
$ docker build --tag terraform-build:v1 <path-to-directory-containing-Dockerfile>
```

3. Esegui il AWS SAMCLI `sam build` comando con le `--build-image` opzioni `--use-container` and.

   Di seguito è riportato un esempio:

```
$ sam build --use-container --build-image terraform-build:v1
```

Fasi successive

Per iniziare a utilizzarlo AWS SAMCLI con i tuoi Terraform progetti, consultaUtilizzo di AWS SAMCLI with Terraform per il debug e il test locali.

Configurazione di Terraform Cloud

Ti consigliamo di utilizzare Terraform v1.6.0 una versione più recente. Se si utilizza una versione precedente, è necessario generare un file del Terraform piano localmente. Il file del piano locale fornisce AWS SAM CLI le informazioni necessarie per eseguire test e debug locali.

Per generare un file di piano locale

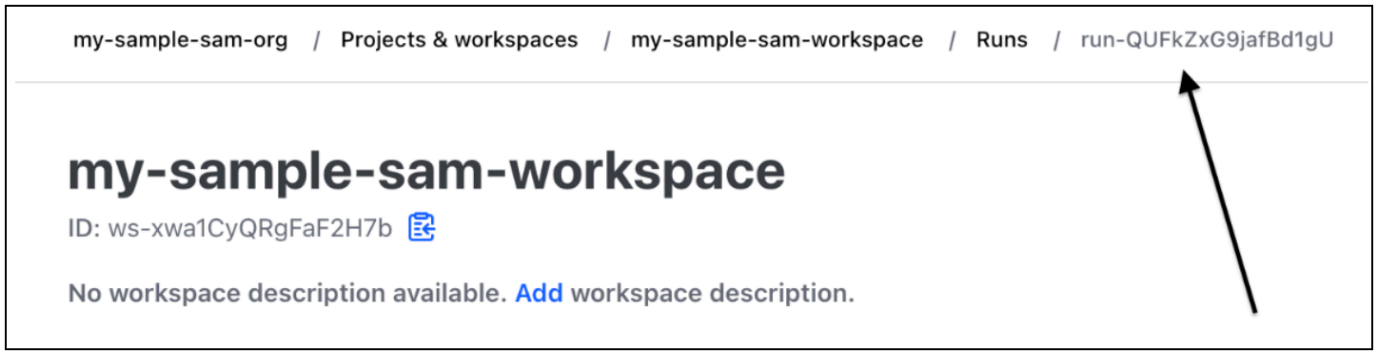
> **Note**
>
> Questi passaggi non sono necessari Terraform v1.6.0 o più recenti. Per iniziare a usare AWS SAM CLI withTerraform Cloud, consulta AWS SAMCLIUsando con Terraform.

1. Configura un token API: il tipo di token dipenderà dal tuo livello di accesso. Per ulteriori informazioni, consulta la sezione relativa ai token API nella Terraform Clouddocumentazione.
2. Imposta la variabile di ambiente del token API: di seguito è riportato un esempio tratto dalla riga di comando:

```
$ export TOKEN="<api-token-value>"
```

3. Ottieni il tuo Run ID: dalla Terraform Cloud console, individua l'ID di esecuzione per la Terraform corsa che desideri utilizzare con AWS SAMCLI.

   L'ID di corsa si trova nel percorso breadcrumb della corsa.



4. Recupera il file del piano: utilizzando il token API, ottieni il file del piano locale. Di seguito è riportato un esempio tratto dalla riga di comando:

```
curl \
   --header "Authorization: Bearer $TOKEN" \
   --header "Content-Type: application/vnd.api+json" \
   --location \
   https://app.terraform.io/api/v2/runs/<run ID>/plan/json-output \
   > custom_plan.json
```

Ora sei pronto per usare AWS SAMCLI conTerraform Cloud. Quando utilizzate un AWS SAMCLI comando supportato, utilizzate l'`--terraform-plan-file`opzione per specificare il nome e il percorso del file del piano locale. Di seguito è riportato un esempio:

```
$ sam local invoke --hook-name terraform --terraform-plan-file custom-plan.json
```

Di seguito è riportato un esempio di utilizzo del `sam local start-api` comando:

```
$ sam local start-api --hook-name terraform --terraform-plan-file custom-plan.json
```

Per un'applicazione di esempio che puoi usare con questi esempi, consulta api_gateway_v2_tf_cloud nel repository aws-samples. GitHub

Fasi successive

Per iniziare a usare AWS SAMCLI withTerraform Cloud, consultaUtilizzo di AWS SAMCLI with Terraform per il debug e il test locali.

Utilizzo di AWS SAMCLI with Terraform per il debug e il test locali

Questo argomento spiega come utilizzare i AWS Serverless Application Model comandi Command Line Interface (AWS SAMCLI) supportati con i Terraform progetti eTerraform Cloud.

Per fornire feedback e inviare richieste di funzionalità, crea un GitHubproblema.

Argomenti

- Test locali con sam local invoke
- Test locali con sam local start-api
- Test locali con sam local start-lambda
- Limitazioni di Terraform

Test locali con sam local invoke

> **Note**
>
> Per utilizzare il AWS SAMCLI test localmente, è necessario che Docker sia installato e configurato. Per istruzioni, consulta Installazione di Docker da utilizzare con AWS SAMCLI.

Di seguito è riportato un esempio di test della funzione Lambda a livello locale trasmettendo un evento:

```
$ sam local invoke --hook-name terraform hello_world_function -e events/event.json -
```

Per ulteriori informazioni sull'utilizzo di questo comando, consultaIntroduzione ai test con sam local invoke.

Test locali con sam local start-api

Per utilizzarlo `sam local start-api` conTerraform, esegui quanto segue:

```
$ sam local start-api --hook-name terraform
```

Di seguito è riportato un esempio:

```
$ sam local start-api --hook-name terraform





Running Prepare Hook to prepare the current application


Executing prepare hook of hook "terraform"


Initializing Terraform application


...
Creating terraform plan and getting JSON output


....
Generating metadata file





Unresolvable attributes discovered in project, run terraform apply to resolve them.




Finished generating metadata file. Storing in...
Prepare hook completed and metadata file generated at: ...
Mounting HelloWorldFunction at http://127.0.0.1:3000/hello [GET]


Mounting None at http://127.0.0.1:3000/hello [POST]
```

```
You can now browse to the above endpoints to invoke your functions. You do not need
 to restart/reload SAM CLI while working on your functions, changes will be reflected
 instantly/automatically. If you
used sam build before running local commands, you will need to re-run sam build for the
 changes to be picked up. You only need to restart SAM CLI if you update your AWS SAM
 template
2023-06-26 13:21:20  * Running on http://127.0.0.1:3000/ (Press CTRL+C to quit)
```

Per ulteriori informazioni su questo comando, vedereIntroduzione ai test con sam local start-api.

Funzioni Lambda che utilizzano gli autorizzatori Lambda

Per le funzioni Lambda configurate per utilizzare gli autorizzatori Lambda, richiameranno AWS SAMCLI automaticamente l'autorizzatore Lambda prima di richiamare l'endpoint della funzione Lambda.

- Per ulteriori informazioni su questa funzionalità, consulta. AWS SAMCLI Funzioni Lambda che utilizzano gli autorizzatori Lambda
- Per ulteriori informazioni sull'utilizzo degli autorizzatori Lambda inTerraform, consulta Resource: aws_api_gateway_authorizernel registro. Terraform

Test locali con sam local start-lambda

Di seguito è riportato un esempio di test della funzione Lambda localmente con AWS Command Line Interface ()AWS CLI:

1. Utilizzate il AWS SAMCLI per creare un ambiente di test locale:

```
$ sam local start-lambda --hook-name terraform hello_world_function
```

2. Usa AWS CLI per richiamare la tua funzione localmente:

```
$ aws lambda invoke --function-name hello_world_function --endpoint-
url http://127.0.0.1:3001/ response.json --cli-binary-format raw-in-base64-out --
payload file://events/event.json
```

Per ulteriori informazioni su questo comando, vedereIntroduzione ai test con sam local start-lambda.

Limitazioni di Terraform

Di seguito sono riportate le limitazioni relative all'utilizzo di AWS SAMCLI conTerraform:

- Funzioni Lambda collegate a più livelli.
- Terraformvariabili locali che definiscono i collegamenti tra le risorse.
- Riferimento a una funzione Lambda che non è stata ancora creata. Ciò include le funzioni definite nell'attributo body della risorsa API REST.

Per evitare queste limitazioni, puoi eseguire l'esecuzione `terraform apply` quando viene aggiunta una nuova risorsa.

Utilizzo di AWS SAMCLI with Serverless.tf per il debug e il test locali

La AWS Serverless Application Model Command Line Interface (AWS SAMCLI) può essere utilizzata con i moduli Serverless.tf per il debug e il test locali di funzioni e livelli. AWS Lambda Sono supportati i seguenti comandi AWS SAMCLI:

- `sam build`
- `sam local invoke`
- `sam local start-api`
- `sam local start-lambda`

> **Note**
>
> La versione 4.6.0 e successive di Serverless.TF supporta l'integrazione. AWS SAMCLI

Per iniziare a utilizzarlo AWS SAMCLI con i tuoi moduli Serverless.TF, esegui l'aggiornamento alla versione più recente di Serverless.TF e il. AWS SAMCLI

A partire dalla versione 6.0.0 di serverless.tf, è necessario impostare il parametro come. `create_sam_metadata true` Ciò genera le risorse di metadati richieste dal comando. AWS SAMCLI `sam build`

Per ulteriori informazioni su Serverless.tf, consulta terraform-aws-lambda-module.

AWS SAMCLIcon riferimento Terraform

Questa sezione è il riferimento per l'utilizzo dell'interfaccia a riga di AWS Serverless Application Model comando (AWS SAMCLI) Terraform per il debug e il test locali.

Per fornire feedback e inviare richieste di funzionalità, crea un problema. GitHub

AWS SAM riferimento alle funzionalità supportate

La documentazione di riferimento per AWS SAMCLI le funzionalità che sono supportate per l'uso Terraform è disponibile qui:

- sam build
- sam local invoke
- sam local start-api
- sam local start-lambda

Terraformriferimento specifico

La documentazione di riferimento specifica per l'utilizzo AWS SAMCLI con Terraform è disponibile qui:

- risorsa di metadati sam

risorsa di metadati sam

Questa pagina contiene informazioni di riferimento per il tipo di sam metadata resource risorsa utilizzato con Terraform i progetti.

- Per un'introduzione all'uso dell'interfaccia a riga di AWS Serverless Application Model comando (AWS SAMCLI) conTerraform, vedereA cosa serve AWS SAMCLI il supportoTerraform?.
- Per usare AWS SAMCLI withTerraform, vediUtilizzo di AWS SAMCLI with Terraform per il debug e il test locali.

Argomenti

- Arguments (Argomenti)
- Esempi

Arguments (Argomenti)

Argomento	Description
`built_output_path`	Il percorso verso gli artefatti incorporati della tua AWS Lambda funzione.
`docker_build_args`	Stringa decodificata dell'oggetto JSON degli argomenti di compilazione di Docker. Questo argomento è facoltativo.
`docker_context`	Il percorso della directory contenente il contesto di creazione dell'immagine Docker.
`docker_file`	Il percorso del file Docker. Questo percorso è relativo al `docker_context` percorso. Questo argomento è facoltativo. Il valore predefinito è `Dockerfile` .
`docker_tag`	Il valore del tag di immagine Docker creato. Questo valore è facoltativo.
`depends_on`	Il percorso verso la risorsa di costruzione per la funzione o il layer Lambda. Per ulteriori informazioni, consulta L'depends_onargomento nel Terraformregistro.
`original_source_code`	Il percorso in cui viene definita la funzione Lambda. Questo valore può essere una stringa, una matrice di stringhe o un oggetto JSON decodificato come stringa. • Per gli array di stringhe, viene utilizzato solo il primo valore poiché non sono supportati più percorsi di codice. • Per gli oggetti JSON, `source_code_property` deve essere definito anche.
`resource_name`	Il nome della funzione Lambda.

Argomento	Description
`resource_type`	Il formato del tipo di pacchetto di funzioni Lambda. I valori accettati sono: • `IMAGE_LAMBDA_FUNCTION` • `LAMBDA_LAYER` • `ZIP_LAMBDA_FUNCTION`
`source_code_property`	Il percorso del codice risorsa Lambda nell'oggetto JSON. Definisci questa proprietà quando `original_source_code` è un oggetto JSON.

Esempi

risorsa di metadati sam che fa riferimento a una funzione Lambda utilizzando il tipo di pacchetto ZIP

```
# Lambda function resource
resource "aws_lambda_function" "tf_lambda_func" {
  filename = "${path.module}/python/hello-world.zip"
  handler = "index.lambda_handler"
  runtime = "python3.8"
  function_name = "function_example"
  role = aws_iam_role.iam_for_lambda.arn
  depends_on = [
    null_resource.build_lambda_function # function build logic
  ]
}

# sam metadata resource
resource "null_resource" "sam_metadata_function_example" {
  triggers = {
    resource_name = "aws_lambda_function.function_example"
    resource_type = "ZIP_LAMBDA_FUNCTION"
    original_source_code = "${path.module}/python"
    built_output_path = "${path.module}/building/function_example"
  }
  depends_on = [
    null_resource.build_lambda_function # function build logic
  ]
```

```
}
```

risorsa di metadati sam che fa riferimento a una funzione Lambda utilizzando il tipo di pacchetto image

```
resource "null_resource" "sam_metadata_function {
  triggers = {
    resource_name = "aws_lambda_function.image_function"
    resource_type = "IMAGE_LAMBDA_FUNCTION"
    docker_context = local.lambda_src_path
    docker_file = "Dockerfile"
    docker_build_args = jsonencode(var.build_args)
    docker_tag = "latest"
  }
}
```

risorsa di metadati sam che fa riferimento a un livello Lambda

```
resource "null_resource" "sam_metadata_layer1" {
  triggers = {
    resource_name = "aws_lambda_layer_version.layer"
    resource_type = "LAMBDA_LAYER"
    original_source_code = local.layer_src
    built_output_path = "${path.module}/${layer_build_path}"
  }
  depends_on = [null_resource.layer_build]
}
```

A cosa serve AWS SAMCLI il supportoTerraform?

Usa la AWS Serverless Application Model Command Line Interface (AWS SAMCLI) con i tuoi Terraform progetti o Terraform Cloud per eseguire il debug e il test locali di:

- AWS Lambda funzioni e livelli.
- Amazon API Gateway APIs.

Per un'introduzione aTerraform, consulta What isTerraform? sul sito HashiCorpTerraformWeb.

Per fornire feedback e inviare richieste di funzionalità, crea un GitHubproblema.

> **Note**
>
> Come parte della fase di analisi AWS SAMCLI dell'integrazione, AWS SAMCLI i comandi utente dei processi generano file e dati di progetto. L'output del comando dovrebbe rimanere invariato, ma in alcuni ambienti, l'ambiente o il runner possono inserire registri o informazioni aggiuntivi nell'output.

Argomenti

- Che cos'è il? AWS SAMCLI
- Come si usa il AWS SAMCLI conTerraform?
- Fasi successive

Che cos'è il? AWS SAMCLI

AWS SAMCLIÈ uno strumento a riga di comando che puoi utilizzare con AWS SAM modelli e integrazioni di terze parti supportateTerraform, ad esempio per creare ed eseguire applicazioni serverless. Per un'introduzione a AWS SAMCLI, vedere. Che cos'è il? AWS SAMCLI

AWS SAMCLISupporta i seguenti comandi perTerraform:

- `sam local invoke`— Avvia localmente una chiamata una tantum di una AWS Lambda risorsa funzionale. Per ulteriori informazioni su questo comando, vedere. Introduzione ai test con sam local invoke
- `sam local start-api`— Esegui le tue risorse Lambda localmente ed esegui il test tramite un host server HTTP locale. Questo tipo di test è utile per le funzioni Lambda richiamate da un endpoint API Gateway. Per ulteriori informazioni su questo comando, consulta. Introduzione ai test con sam local start-api
- `sam local start-lambda`— Avvia un endpoint locale per la tua funzione Lambda per richiamarla localmente AWS Command Line Interface usando AWS CLI() o. SDKs Per ulteriori informazioni su questo comando, consulta. Introduzione ai test con sam local start-lambda

Come si usa il AWS SAMCLI conTerraform?

Il Terraformflusso di lavoro principale è costituito da tre fasi: scrittura, pianificazione e applicazione. Con AWS SAMCLI il supporto perTerraform, puoi sfruttare il AWS SAMCLI `sam local` set di

comandi continuando a utilizzare i Terraform flussi di lavoro per gestire le tue applicazioni su AWS. In genere, ciò significa quanto segue:

- Scrittura: crea la tua infrastruttura come codice utilizzandoTerraform.
- Test ed esegui il debug: utilizzalo per testare ed AWS SAMCLI eseguire il debug delle tue applicazioni a livello locale.
- Pianifica: visualizza in anteprima le modifiche prima di applicarle.
- Applica: effettua il provisioning della tua infrastruttura.

Per un esempio di utilizzo di AWS SAMCLI withTerraform, consulta Better together: AWS SAMCLI and HashiCorp Terraform at the AWS Compute Blog.

Fasi successive

Per completare tutti i prerequisiti e la configurazioneTerraform, vedere. Guida introduttiva Terraform al supporto per AWS SAMCLI

Pubblicazione dell'applicazione con AWS SAMCLI

Per rendere la tua AWS SAM applicazione disponibile ad altri utenti affinché la trovino e la distribuiscano, puoi utilizzare il AWS SAMCLI per pubblicarla su. AWS Serverless Application Repository Per pubblicare l'applicazione utilizzando il AWS SAMCLI, è necessario definirla utilizzando un AWS SAM modello. È inoltre necessario averlo testato localmente o nel AWS cloud.

Segui le istruzioni in questo argomento per creare una nuova applicazione, creare una nuova versione di un'applicazione esistente o aggiornare i metadati di un'applicazione esistente. (Ciò che fate dipende dal fatto che l'applicazione esista già nell'applicazione e che AWS Serverless Application Repository i metadati dell'applicazione stiano cambiando). Per ulteriori informazioni sui metadati dell'applicazione, vedere. AWS SAM proprietà della sezione Metadati del modello

Prerequisiti

Prima di pubblicare un'applicazione AWS Serverless Application Repository utilizzando il AWS SAMCLI, è necessario disporre di quanto segue:

- L' AWS SAMCLIinstallato. Per ulteriori informazioni, consulta Installa il AWS SAMCLI. Per determinare se AWS SAMCLI è installato, esegui il comando seguente:

  ```
  sam --version
  ```

- Un AWS SAM modello valido.
- Il codice dell'applicazione e le dipendenze a cui fa riferimento il AWS SAM modello.
- Una versione semantica, necessaria solo per condividere pubblicamente l'applicazione. Questo valore può essere semplice come 1.0.
- Un URL che rimanda al codice sorgente dell'applicazione.
- Un file `README.md`. Questo file dovrebbe descrivere come i clienti possono utilizzare l'applicazione e come configurarla prima di distribuirla nei propri AWS account.
- Un `LICENSE.txt` file, necessario solo per condividere pubblicamente l'applicazione.
- Se l'applicazione contiene applicazioni annidate, è necessario averle già pubblicate AWS Serverless Application Repository su.
- Una policy bucket valida di Amazon Simple Storage Service (Amazon S3) Simple Storage Service (Amazon S3) che concede al servizio le autorizzazioni di lettura per gli artefatti che carichi su

Amazon S3 quando crei il pacchetto dell'applicazione. Per configurare questa politica, procedi come segue:

1. Apri la console Amazon S3 all'indirizzo. https://console.aws.amazon.com/s3/
2. Scegli il nome del bucket Amazon S3 che hai usato per impacchettare la tua applicazione.
3. Seleziona Autorizzazioni.
4. Nella sezione Autorizzazioni, alla voce Policy del bucket, scegliere Modifica.
5. Nella pagina Modifica policy del bucket, incolla la seguente dichiarazione di policy nell'editor Policy. Nella dichiarazione sulla politica, assicurati di utilizzare il nome del bucket nell'`Resource`elemento e l'ID AWS dell'account nell'`Condition`elemento. L'espressione nell'`Condition`elemento garantisce che AWS Serverless Application Repository disponga del permesso di accedere solo alle applicazioni dell' AWS account specificato. Per ulteriori informazioni sulle dichiarazioni politiche, consulta il riferimento agli elementi delle policy IAM JSON nella IAM User Guide.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service":  "serverlessrepo.amazonaws.com"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::<your-bucket-name>/*",
            "Condition" : {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012"
                }
            }
        }
    ]
}
```

6. Scegli Save changes (Salva modifiche).

Pubblicazione di una nuova applicazione

Passaggio 1: aggiungere una **`Metadata`** sezione al AWS SAM modello

Per prima cosa, aggiungi una `Metadata` sezione al tuo AWS SAM modello. Fornisci le informazioni sull'applicazione da pubblicare su AWS Serverless Application Repository.

Di seguito è riportata una `Metadata` sezione di esempio:

```
Metadata:
  AWS::ServerlessRepo::Application:
    Name: my-app
    Description: hello world
    Author: user1
    SpdxLicenseId: Apache-2.0
    LicenseUrl: LICENSE.txt
    ReadmeUrl: README.md
    Labels: ['tests']
    HomePageUrl: https://github.com/user1/my-app-project
    SemanticVersion: 0.0.1
    SourceCodeUrl: https://github.com/user1/my-app-project

Resources:
  HelloWorldFunction:
    Type: AWS::Lambda::Function
      Properties:
        ...
        CodeUri: source-code1
        ...
```

Per ulteriori informazioni sulla `Metadata` sezione del AWS SAM modello, vedereAWS SAM proprietà della sezione Metadati del modello.

Fase 2: Package dell'applicazione

Esegui il AWS SAMCLI comando seguente, che carica gli artefatti dell'applicazione su Amazon S3 e genera un nuovo file modello chiamato: `packaged.yaml`

```
sam package --output-template-file packaged.yaml --s3-bucket <your-bucket-name>
```

Utilizzerai il file `packaged.yaml` modello nel passaggio successivo per pubblicare l'applicazione su. AWS Serverless Application Repository Questo file è simile al file modello originale (`template.yaml`), ma presenta una differenza fondamentale: le `ReadmeUrl` proprietà `CodeUriLicenseUrl`, e puntano al bucket Amazon S3 e agli oggetti che contengono i rispettivi artefatti.

Il frammento riportato di seguito tratto dal file del modello di esempio `packaged.yaml` mostra la proprietà `CodeUri`:

```
MySampleFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: s3://bucketname/fbd77a3647a4f47a352fc0 objectGUID

...
```

Fase 3: Pubblicare l'applicazione

Per pubblicare una versione privata dell' AWS SAM applicazione su AWS Serverless Application Repository, esegui il seguente AWS SAMCLI comando:

```
sam publish --template packaged.yaml --region us-east-1
```

L'output del `sam publish` comando include un collegamento all'applicazione su AWS Serverless Application Repository. Puoi anche andare direttamente alla pagina di AWS Serverless Application Repository destinazione e cercare la tua applicazione.

Passaggio 4: Condividi l'applicazione (opzionale)

Per impostazione predefinita, l'applicazione è impostata come privata, quindi non è visibile agli altri AWS account. Per condividere l'applicazione con altri, è necessario renderla pubblica o concedere l'autorizzazione a un elenco specifico di AWS account.

Per informazioni sulla condivisione dell'applicazione utilizzando il AWS CLI, consulta Esempi di policy AWS Serverless Application Repository basate sulle risorse nella Guida per gli AWS Serverless Application Repository sviluppatori. Per informazioni sulla condivisione dell'applicazione tramite Console di gestione AWS, consulta Sharing an Application nella Developer Guide.AWS Serverless Application Repository

Pubblicazione di una nuova versione di un'applicazione esistente

Dopo aver pubblicato un'applicazione su AWS Serverless Application Repository, potresti volerne pubblicare una nuova versione. Ad esempio, potresti aver modificato il codice della funzione Lambda o aggiunto un nuovo componente all'architettura dell'applicazione.

Per aggiornare un'applicazione che hai pubblicato in precedenza, pubblica nuovamente l'applicazione utilizzando la stessa procedura descritta in precedenza. Nella `Metadata` sezione del file AWS SAM modello, fornite lo stesso nome di applicazione con cui l'avete pubblicata originariamente, ma includete un nuovo `SemanticVersion` valore.

Ad esempio, considerate un'applicazione pubblicata con il nome `SampleApp` e un `SemanticVersion` di`1.0.0`. Per aggiornare tale applicazione, il AWS SAM modello deve avere il nome dell'applicazione `SampleApp` e un `SemanticVersion` di `1.0.1` (o qualcosa di diverso`1.0.0`).

Argomenti aggiuntivi

- AWS SAM proprietà della sezione Metadati del modello

AWS SAM proprietà della sezione Metadati del modello

`AWS::ServerlessRepo::Application`è una chiave di metadati che potete utilizzare per specificare le informazioni sull'applicazione che desiderate pubblicare su. AWS Serverless Application Repository

> **Note**
>
> CloudFormation le funzioni intrinseche non sono supportate dalla chiave dei `AWS::ServerlessRepo::Application` metadati.

Properties

Questa tabella fornisce informazioni sulle proprietà della `Metadata` sezione del modello. AWS SAM Questa sezione è necessaria per pubblicare applicazioni AWS Serverless Application Repository che utilizzano AWS SAMCLI.

Proprietà	Tipo	Campo obbligatorio	Descrizione
`Name`	Stringa	TRUE	Il nome dell'applicazione. Lunghezza minima = 1. Lunghezza massima = 140. Modello: `"[a-zA-Z0-9\\-]+";`
`Description`	Stringa	TRUE	Descrizione dell'applicazione. Lunghezza minima = 1. Lunghezza massima = 256.
`Author`	Stringa	TRUE	Il nome dell'autore che pubblica l'applicazione. Lunghezza minima = 1. Lunghezza massima = 127. Modello: `"^[a-z0-9](([a-z0-9]|-(?!-))*[a-z0-9])?$";`
`SpdxLicenseId`	Stringa	FALSE	Un identificatore di licenza valido. Per visualizzare l'elenco degli identificatori di licenza validi, vedere l'elenco delle licenze SPDX sul sito Web Software Package Data Exchange (SPDX).
`LicenseUrl`	Stringa	FALSE	Il riferimento a un file di licenza locale o a un collegamento Amazon S3 a un file di licenza che corrisponde al valore SPDXLicenseID dell'applicazione. Un file AWS SAM modello che non è stato impacchettato utilizzando il `sam package` comando può avere un riferimento a un file locale per questa proprietà. Tuttavia, affinché un'applicazione possa essere pubblicata utilizzando il `sam publish` comando, questa proprietà deve essere un riferimento a un bucket Amazon S3. Dimensione massima: 5 MB.

Proprietà	Tipo	Campo obbligatorio	Descrizione
			È necessario fornire un valore per questa proprietà per rendere pubblica l'applicazione. Tieni presente che non puoi aggiornare questa proprietà dopo la pubblicazione dell'applicazione. Pertanto, per aggiungere una licenza a un'applicazione, è necessario prima eliminarla o pubblicare una nuova applicazione con un nome diverso.
`ReadmeUrl`	Stringa	FALSE	Il riferimento a un file readme locale o un collegamento Amazon S3 al file readme che contiene una descrizione più dettagliata dell'applicazione e del suo funzionamento. Un file AWS SAM modello che non è stato impacchettato utilizzando il `sam package` comando può avere un riferimento a un file locale per questa proprietà. Tuttavia, per essere pubblicata utilizzando il `sam publish` comando, questa proprietà deve essere un riferimento a un bucket Amazon S3. Dimensione massima: 5 MB.
`Labels`	Stringa	FALSE	Le etichette che migliorano la scoperta delle applicazioni nei risultati di ricerca. Lunghezza minima = 1. Lunghezza massima = 127. Numero massimo di etichette: 10. Modello: `"^[a-zA-Z0-9+\\-_:\\/@]+$";`
`HomePageUrl`	Stringa	FALSE	Un URL con ulteriori informazioni sull'applicazione, ad esempio la posizione del GitHub repository dell'applicazione.

Proprietà	Tipo	Campo obbligatorio	Descrizione
`SemanticVersion`	Stringa	FALSE	La versione semantica dell'applicazione. Per le specifiche di Semantic Versioning, consultate il sito Web Semantic Versioning. È necessario fornire un valore per questa proprietà per rendere pubblica l'applicazione.
`SourceCodeUrl`	Stringa	FALSE	Un collegamento a un repository pubblico per il codice sorgente dell'applicazione.

Casi d'uso

Questa sezione elenca i casi d'uso per le applicazioni di pubblicazione, insieme alle `Metadata` proprietà elaborate per tale caso d'uso. Le proprietà che non sono elencate per un determinato caso d'uso vengono ignorate.

- Creazione di una nuova applicazione: viene creata una nuova applicazione se non è presente alcuna applicazione AWS Serverless Application Repository con un nome corrispondente per un account.
  - `Name`
  - `SpdxLicenseId`
  - `LicenseUrl`
  - `Description`
  - `Author`
  - `ReadmeUrl`
  - `Labels`
  - `HomePageUrl`
  - `SourceCodeUrl`
  - `SemanticVersion`
  - Il contenuto del AWS SAM modello (ad esempio, eventuali fonti di eventi, risorse e codice funzione Lambda)

- Creazione di una versione dell'applicazione: viene creata una versione dell'applicazione se esiste già un'applicazione AWS Serverless Application Repository con un nome corrispondente per un account e questa SemanticVersion è in corso di modifica.
  - `Description`
  - `Author`
  - `ReadmeUrl`
  - `Labels`
  - `HomePageUrl`
  - `SourceCodeUrl`
  - `SemanticVersion`
  - Il contenuto del AWS SAM modello (ad esempio, eventuali fonti di eventi, risorse e codice funzione Lambda)

- Aggiornamento di un'applicazione: un'applicazione viene aggiornata se esiste già un'applicazione AWS Serverless Application Repository con un nome corrispondente per un account e non SemanticVersion viene modificata.
  - `Description`
  - `Author`
  - `ReadmeUrl`
  - `Labels`
  - `HomePageUrl`

Esempio

Di seguito è riportata una `Metadata` sezione di esempio:

```
Metadata:
  AWS::ServerlessRepo::Application:
    Name: my-app
    Description: hello world
    Author: user1
    SpdxLicenseId: Apache-2.0
```

```
    LicenseUrl: LICENSE.txt
    ReadmeUrl: README.md
    Labels: ['tests']
    HomePageUrl: https://github.com/user1/my-app-project
    SemanticVersion: 0.0.1
    SourceCodeUrl: https://github.com/user1/my-app-project
```

Cronologia dei documenti per AWS SAM

La tabella seguente descrive le modifiche importanti in ogni versione della Guida per gli AWS Serverless Application Model sviluppatori. Per ricevere notifiche sugli aggiornamenti di questa documentazione, puoi abbonarti a un feed RSS.

- Ultimo aggiornamento della documentazione: 20 giugno 2024

Modifica	Descrizione	Data
Contenuti ristrutturati e aggiornati in tutta la guida per gli sviluppatori	La guida è stata riorganizzata e ristrutturata per migliorarne la reperibilità e l'usabilità. Titoli aggiornati e migliorati. Sono stati forniti dettagli aggiuntiv i durante l'introduzione di argomenti e concetti.	20 giugno 2024
Aggiunto AWS SAMCLI il supporto per Ruby 3.3	Ruby 3.3 è ora disponibile come runtime e repositor y di immagini. Vedi Image repositories e sam init per i dettagli.	4 aprile 2024
Opzioni di comando aggiunte AWS SAMCLI	Sono disponibili nuove opzioni per il comando sam local start-api:`--ssl-cert-file` `PATH`,. `--ssl-key-file` `PATH` Inoltre, la nuova opzione della riga di comando `--add-host LIST` è disponibile per sam local invoke, sam localstart-api e sam local start-lambda	20 marzo 2024
È stato aggiunto il supporto per.NET 8 AWS SAMCLI	.NET 8 è ora disponibile come archivio di runtime e immagini.	22 febbraio 2024

	I tempi di esecuzione e gli archivi di immagini per .NET Core 3.1, Node.js 14, Node.js 12, Python 3.7, Ruby 2.7 non sono più supportati. Vedi Archivi di immagini e sam init.	
Aggiunto il programma di installazione del AWS SAMCLI pacchetto arm64 per Linux	Per istruzioni, vedere Installazione di. AWS SAMCLI	6 dicembre 2023
È stata aggiunta l'opzione --watch-exclude per il comando sam sync AWS SAMCLI	Esclude file e cartelle dall'avvio di una sincronizzazione. Per ulteriori informazioni, consulta Specificare file e cartelle che non avvieranno una sincronizzazione.	6 dicembre 2023
Aggiunta: build-in-source opzione per il comando AWS SAMCLI sam sync	Crea il tuo progetto nella cartella dei sorgenti per velocizzare il processo di compilazione. Per ulteriori informazioni, consulta Accelerare i tempi di compilazione creando il progetto nella cartella dei sorgenti.	6 dicembre 2023
Aggiunta: build-in-source opzione per il comando AWS SAMCLI sam build	Crea il tuo progetto nella cartella dei sorgenti per velocizzare il processo di compilazione. Per ulteriori informazioni, consulta Accelerare i tempi di compilazione creando il progetto nella cartella dei sorgenti.	6 dicembre 2023

È stato aggiunto un nuovo supporto alle risorse per il AWS SAMCLI comando di invocazione remota	Utilizzalo `sam remote invoke` con le applicazioni Kinesis Data Streams, le code Amazon SQS e le macchine a stati Step Functions. Per ulteriori informazioni, consulta Using sam remote invoke.	15 novembre 2023
Aggiunto un nuovo AWS SAMCLI comando test-event remoto per eventi di test condivisibili	Usa il AWS SAM CLI per accedere e gestire gli eventi di test condivisibili dal registro degli EventBridge schemi per testare le tue funzioni Lambda in. Cloud AWS Per ulteriori informazioni, consulta Using sam remote test-event.	3 ottobre 2023
AWS SAMCLIil supporto per Terraform è ora disponibile a livello generale	Per ulteriori informazioni sull' AWS SAMCLIassistenza perTerraform, consulta la sezione AWS SAMCLITeraformassistenza.	05 settembre 2023
È stato aggiunto AWS SAMCLI il supporto per Terraform Cloud	AWS SAMCLIOra supporta i test locali perTerraform Cloud. Per ulteriori informazioni, consulta Configurazione per Terraform Cloud.	05 settembre 2023
È stato aggiunto YAML il supporto per il formato di file per il file AWS SAMCLI di configurazione	AWS SAMCLIOra supporta il formato di file `[.yaml|.yml]`. Le pagine di AWS SAMCLI configurazione AWS SAMCLI e dei file di configurazione sono state aggiornate.	18 luglio 2023

È stato aggiunto AWS SAMCLIsam local start-api il supporto ai comandi per Terraform	A cosa serve AWS SAMCLI il supportoTerraform? la sezione è stata aggiornata per includere AWS SAMCLI `sam local start-api` il supporto dei comandi perTerraform.	06 luglio 2023
Aggiunto un nuovo AWS SAMCLI comando di richiamo remoto	Per iniziare a usare`sam remote invoke`, vedi Using sam remote invoke.	22 giugno 2023
È stato aggiunto un tipo di AWS AppSyncGraphQL API risorsa serverless	Crea una nuova `AWS::Serverless::GraphQLApi` sezione che descrive come definire una GraphQL API risorsa con AWS SAM.	22 giugno 2023
Aggiunto AWS SAMCLI il supporto per Ruby 3.2	Aggiorna sam init page per includere nuove immagini di base e valori di runtime. Aggiorna la pagina degli archivi di immagini con Ruby 3.2 URI Amazon ECR.	6 giugno 2023
Sono stati aggiunti passaggi opzionali per la verifica dell'integrità del programma di installazione del pacchetto AWS SAMCLI	Aggiorna Installazione della AWS SAMCLI pagina in modo che rifletta il passaggio facoltativo. Crea Verifica l'integrità della pagina di AWS SAMCLI installazione per documentare i passaggi.	31 maggio 2023

Aggiunta l'opzione sam sync per saltare la sincronizzazione dell'infrastruttura	Personalizza se è necessaria una CloudFormation distribuzione ogni volta che `sam sync` viene eseguita. Per ulteriori informazioni, consulta Ignorare la CloudFormation distribuzione iniziale.	23 marzo 2023
Aggiunto il supporto per il tipo di origine degli eventi DocumentDB	La specifica del AWS SAM modello ora supporta il tipo di origine `DocumentDB` degli eventi per la `AWS::Serverless::Function` risorsa. Per ulteriori informazioni, consulta DocumentDB.	10 marzo 2023
Crea funzioni Rust Lambda con Cargo Lambda	Usa il AWS SAMCLI per creare le tue funzioni Rust Lambda usando. Cargo Lambda Per ulteriori informazioni, consulta Creazione di funzioni Lambda di Rust con. Cargo Lambda	23 febbraio 2023
Crea risorse funzionali esterne a AWS SAM	È stata aggiunta una guida su come saltare le funzioni quando si utilizza il sam build comando. Per ulteriori informazioni, consulta Creazione di funzioni al di fuori di AWS SAM.	14 febbraio 2023

Nuova sintassi dei connettori incorporati	Usa la nuova sintassi dei connettori incorporati per definire le tue `AWS::Serverless::Connector` risorse. Per ulteriori informazioni, consulta Gestione delle autorizzazioni delle risorse con AWS SAM i connettori.	8 febbraio 2023
Aggiunto un nuovo comando sam list per AWS SAMCLI	sam listUtilizzatelo per visualizzare informazioni importanti sulle risorse dell'applicazione serverless. Per ulteriori informazioni, consulta sam list.	2 febbraio 2023
Aggiunte proprietà di OutExtension formattazione e compilazione per esbuild	La creazione di funzioni Lambda Node.js con esbuild ora supporta `Format` e `OutExtension` crea proprietà. Per ulteriori informazioni, consulta Creazione di funzioni Lambda di Node.js con esbuild.	2 febbraio 2023
Sono state aggiunte opzioni di gestione del runtime alla specifica del modello AWS SAM	Configura le opzioni di gestione del runtime per le tue funzioni Lambda. Per ulteriori informazioni, consulta `RuntimeManagementConfig`.	24 gennaio 2023

Proprietà Target aggiunta alla EventSource AWS::Serverless::StateMachine risorsa.	`AWS::Serverless::StateMachine` il tipo di risorsa supporta la `Target` proprietà `EventBridgeRule` e le fonti di `Schedule` eventi.	13 gennaio 2023
Configura il ridimensionamento dei poller SQS per le funzioni Lambda	Configura il ridimensionamento dei poller SQS con la proprietà for. `ScalingConfig` `AWS::Serverless::Function` Per ulteriori informazioni, consulta `ScalingConfig` .	12 gennaio 2023
Convalida AWS SAM le applicazioni con cfn-lint	Puoi usare cfn-lint per convalidare i tuoi modelli tramite. AWS SAM AWS SAMCLI Per saperne di più, consulta Convalida con cfn-lint.	11 gennaio 2023
Monitora le tue applicazioni serverless con Application Insights CloudWatch	Configura Amazon CloudWatch Application Insights per monitorare AWS SAM le tue applicazioni. Per saperne di più, consulta Monitora le tue applicazioni serverless con CloudWatch Application Insights.	19 dicembre 2022
Aggiunto il programma di installazione dei AWS SAMCLI pacchetti per macOS	Installa AWS SAMCLI utilizzando il nuovo programma di installazione del pacchetto macOS. Per ulteriori informazioni, consulta Installazione di. AWS SAMCLI	6 dicembre 2022

Aggiunto il supporto per Lambda SnapStart	Configura le funzioni Lambda SnapStart per creare istantanee, che sono stati memorizzati nella cache delle tue funzioni inizializzate. Per ulteriori informazioni, consulta `AWS::Serverless::Function`.	28 novembre 2022
È stato aggiunto il supporto per nodejs18.x AWS SAMCLI	AWS SAMCLIora supporta il runtime nodejs18.x. Per saperne di più, vedi sam init.	17 novembre 2022
È stata aggiunta una guida sulla configurazione dell'accesso e delle autorizzazioni	AWS SAM offre due opzioni che semplificano la gestione degli accessi e delle autorizzazioni per le applicazioni serverless. Per ulteriori informazioni, consulta Gestione dell'accesso e delle autorizzazioni alle risorse.	17 novembre 2022
Aggiunto supporto per la creazione di funzioni.NET 7 Lambda con compilazione AOT nativa	Crea e impacchetta le tue funzioni.NET 7 Lambda utilizzando la compilazione nativa Ahead-of-Time (AOT) per migliorare i tempi di avvio a freddo di Lambda. AWS SAM Per ulteriori informazioni, consulta Creazione di funzioni.NET 7 Lambda con la compilazione AOT nativa.	15 novembre 2022

È stato aggiunto AWS SAMCLITerraform il supporto per il debug e il test locali	Utilizzali all' AWS SAMCLIinterno Terraform dei tuoi progetti per eseguire il debug e il test locali delle funzioni e dei livelli Lambda. Per ulteriori informazioni, consulta AWS SAM Supporto CLI Terraform.	14 novembre 2022
È stato aggiunto AWS SAM il supporto per Scheduler EventBridge	La specifica del modello AWS Serverless Application Model (AWS SAM) fornisce una sintassi semplice e abbreviat a che è possibile utilizzar e per pianificare eventi con EventBridge Scheduler for and. AWS Lambda AWS Step Functions Per ulteriori informazioni, consulta Pianificazione degli eventi con Scheduler. EventBridge	10 novembre 2022
Istruzioni di installazione semplificate AWS SAMCLI	AWS SAMCLIi prerequisiti e i passaggi facoltativi sono stati spostati in pagine separate. La procedura di installazione per i sistemi operativi supportati è disponibile in Installazione di AWS SAMCLI.	4 novembre 2022

È stata aggiunta una correzione per consentire percorsi lunghi per gli utenti di Windows 10	L'archivio dei modelli di AWS SAMCLI app contiene alcuni percorsi di file lunghi che possono causare errori durante l'esecuzione a `sam init` causa delle MAX_PATH limitazioni di Windows 10. Per ulteriori informazioni, consulta Installazione di AWS SAMCLI	4 novembre 2022
Processo di distribuzione graduale aggiornato per le prime distribuzioni	L'implementazione graduale di una funzione Lambda richiede AWS CodeDeploy due passaggi. Per ulteriori informazioni, consulta Distribuzione graduale di una funzione Lambda per la prima volta.	13 ottobre 2022
Supporto aggiuntivo per il filtraggio degli eventi Lambda per più tipi di eventi	`FilterCritera` proprietà aggiunta a `MSKMQ`, e tipi di origine `SelfManagedKafka` degli eventi.	13 ottobre 2022
Aggiunto il supporto OpenID Connect (OIDC) per la pipeline AWS SAM	AWS SAM supporta l'autenticazione utente OpenID Connect (OIDC) per Bitbucket, GitHub Actions e piattaforme di integrazione GitLab continua e distribuzione continua (CI/CD). Per ulteriori informazioni, consulta Utilizzo degli account utente OIDC con pipeline. AWS SAM	13 ottobre 2022

Nota sulle proprietà JwtConfiguration	È stata aggiunta una nota sulla definizione `issuer` e sulle `audience` proprietà in `JwtConfiguration` for `OAuth2Authorizer`.	7 ottobre 2022
Nuove proprietà per Function e StateMachine EventSource	`Enable`de `State` proprietà aggiunte alla fonte di `CloudWatchEvent` eventi per `AWS::Serverless::Function`. `State`proprietà aggiunta alla fonte di `Schedule` eventi per `AWS::Serverless::Function` and `AWS::Serverless::StateMachine`.	6 ottobre 2022
AWS SAM connettori ora disponibili a livello generale	I connettori sono un tipo di risorsa AWS SAM astratto, identificato come`AWS::Serverless::Connector`, che fornisce un metodo semplice e sicuro per assegnare le autorizzazioni tra le risorse delle applicazioni serverless. Per ulteriori informazioni, consulta Gestione delle autorizzazioni delle risorse con i connettori. AWS Serverless Application Model	6 ottobre 2022
Sono state aggiunte nuove opzioni di sincronizzazione sam al AWS SAMCLI	`--dependency-layer` e `--use-container` opzioni aggiunte a sam sync.	20 settembre 2022

Sono state aggiunte nuove opzioni sam deploy al AWS SAMCLI	`--on-failure` opzione aggiunta a. sam deploy	9 settembre 2022
il supporto per esbuild ora è generalmente disponibile	Per creare e impacchettare le funzioni Lambda di Node.js, puoi usarle AWS SAMCLI con il bundler JavaScript esbuild.	1 settembre 2022
Telemetria aggiornata AWS SAMCLI	La descrizione delle informazioni raccolte sul sistema e sull'ambiente è stata aggiornata per includere i valori hash degli attributi di utilizzo.	1 settembre 2022
È stato aggiunto il supporto per variabili di ambiente locali a AWS SAMCLI	Usa le variabili di ambiente con AWS SAMCLI quando richiami le funzioni Lambda localmente e quando esegui API Gateway localmente.	1 settembre 2022
Support per architetture di set di istruzioni Lambda	Utilizzalo AWS SAMCLI per creare funzioni Lambda e layer Lambda per `x86_64` le nostre architetture di set di istruzioni. `arm64` Per ulteriori informazioni, vedete la proprietà Architectures del tipo di risorsa e la proprietà del tipo di `AWS::Serverless::Function` risorsa. CompatibleArchitectures`AWS::Serverless::LayerVersion`	1° ottobre 2021

Generazione di esempi di configurazioni di pipeline	Utilizzate il AWS SAMCLI per generare pipeline di esempio per più CI/CD sistemi, utilizzando i comandi new `sam pipeline bootstrap` and. `sam pipeline init` Per ulteriori informazioni, vedere Generazione di CI/CD pipeline di esempio.	21 luglio 2021
AWS SAMCLIAWS CDK integrazione (anteprima, fase 2)	Con la fase 2 della versione di anteprima pubblica, ora puoi utilizzare la AWS SAMCLI per impacchettare e distribuire AWS CDK applicazioni. È inoltre possibile scaricare un' AWS CDK applicazione di esempio direttamente utilizzando. AWS SAMCLI Per ulteriori informazioni, vedere AWS Cloud Development Kit (AWS CDK) (Anteprima).	13 luglio 2021
Support per RabbitMQ come fonte di eventi per le funzioni	È stato aggiunto il supporto per RabbitMQ come fonte di eventi per funzioni serverless. Per ulteriori informazioni, vedete la `SourceAccessConfigurations` proprietà dell'origine dell'`MQ`evento del tipo di risorsa. `AWS::Serverless::Function`	7 luglio 2021

Distribuzione di applicazioni serverless utilizzando Amazon ECR, crea immagini di container	Usa Amazon ECR build container images per distribuire applicazioni serverless con sistemi CI/CD comuni come Jenkins AWS CodePipeline, CI/CD e Actions. GitLab GitHub Per ulteriori informazioni, consulta Distribuzione di applicazioni serverless.	24 giugno 2021
Eseguire il debug di applicazioni AWS SAM con Toolkit AWS	AWS Toolkits ora supporta il debug step-through con più combinazioni di ambienti di sviluppo integrati () e runtime. IDEs Per ulteriori informazioni, vedete Utilizzo dei toolkit. AWS	20 maggio 2021
AWS SAMCLIAWS CDK integrazione (anteprima)	Ora puoi utilizzarlo AWS SAMCLI per testare e creare AWS CDK applicazioni localmente. Questa è una versione di anteprima pubblica. Per ulteriori informazioni, vedere AWS Cloud Development Kit (AWS CDK) (Anteprima).	29 aprile 2021
L'archivio di immagini del contenitore predefinito è stato modificato in Amazon ECR Public	L'archivio di immagini del contenitore predefinito è passato DockerHub da Amazon ECR Public. Per ulteriori informazioni, consulta Repository di immagini.	06 aprile 2021

Compilazioni AWS SAMCLI notturne	È ora possibile installare una versione non definitiva di AWS SAMCLI, che viene compilata ogni notte. Per ulteriori informazioni, consulta la sezione Nightly build del sottoargomento del sistema operativo a tua scelta in Installazione di. AWS SAMCLI	25 marzo 2021
Supporta le variabili di ambiente del contenitore Build	Ora puoi passare variabili di ambiente per creare contenitori. Per ulteriori informazioni, consulta la sezione `--container-env-var` e `--container-env-var-file` le opzioni in `sam build`.	04 marzo 2021
Nuovo processo di installazione di Linux	Ora puoi installarlo AWS SAMCLI utilizzando un programma di installazione Linux nativo. Per ulteriori informazioni, vedere Installazione di AWS SAMCLI su Linux.	10 febbraio 2021

Support per le code di lettere non scritte per EventBridge	È stato aggiunto il supporto per le code di lettere non scritte e le fonti di `Schedule` eventi per funzioni EventBridge serverless e macchine a stati. Per ulteriori informazioni, consultate la `DeadLetterConfig` proprietà `EventBridgeRule` e le fonti degli `Schedule` eventi, sia per i tipi di risorse che per quelli di risorse. `AWS::Serverless::Function` `AWS::Serverless::StateMachine`	29 gennaio 2021
Support per checkpoint personalizzati	È stato aggiunto il supporto per checkpoint personalizzati per le sorgenti di eventi DynamoDB e Kinesis per le funzioni serverless. Per ulteriori informazioni, consulta la `FunctionResponseTypes` proprietà `Kinesis`e i tipi di `DynamoDB`dati del tipo di risorsa. `AWS::Serverless::Function`	29 gennaio 2021

Support per finestre ribaltabili	È stato aggiunto il supporto per le finestre tumbling per le sorgenti di eventi DynamoDB e Kinesis per le funzioni serverless. Per ulteriori informazioni, consulta la `TumblingWindowInSeconds` proprietà `Kinesis` e i tipi di `DynamoDB` dati del tipo di risorsa. `AWS::Serverless::Function`	17 dicembre 2020
Support per contenitori caldi	È stato aggiunto il supporto per i contenitori caldi durante i test locali utilizzando i AWS SAMCLI comandi `sam local start-api` e `sam local start-lambda`. Per ulteriori informazioni, consulta l'`--warm-containers` opzione relativa a questi comandi.	16 dicembre 2020
Support per immagini di container Lambda	È stato aggiunto il supporto per le immagini dei container Lambda. Per ulteriori informazioni, consulta Creazione di applicazioni.	1 dicembre 2020
Support per la firma del codice	È stato aggiunto il supporto per la firma del codice e le implementazioni affidabili di codice applicativo serverless. Per ulteriori informazioni, vedere Configurazione della firma del codice per le applicazioni. AWS SAM	23 novembre 2020

Support per build parallele e memorizzate nella cache	Prestazioni migliorate delle build di applicazioni serverless aggiungendo due opzioni al `sam build`comando:`--parallel` , che crea funzioni e livelli in parallelo anziché in sequenza e`--cached`, che utilizza gli artefatti di compilazione delle build precedenti quando non sono state apportate modifiche che richiedono una ricostruzione.	10 novembre 2020
Support per Amazon MQ e autenticazione TLS reciproca	È stato aggiunto il supporto per Amazon MQ come fonte di eventi per funzioni serverless. Per ulteriori informazioni, consulta i tipi di `MQ`dati `EventSource` e i tipi di dati del tipo di `AWS::Serverless::Function` risorsa. È stato inoltre aggiunto il supporto per l'autenticazione reciproca TLS (Transport Layer Security) per API Gateway APIs e HTTP APIs. Per ulteriori informazioni, consulta il tipo di `DomainConfiguration` dati del tipo di `AWS::Serverless::Api` risorsa o il tipo di `HttpApiDomainConfiguration` dati del tipo di `AWS::Serverless::HttpApi` risorsa.	05 novembre 2020

Support per gli autorizzatori Lambda per HTTP APIs	È stato aggiunto il supporto per gli autorizzatori Lambda per il `AWS::Serverless::HttpApi` tipo di risorsa. Per ulteriori informazioni, consulta Esempio di autorizzazione Lambda (). `AWS::Serverless::HttpApi`	27 ottobre 2020
Support per più file e ambienti di configurazione	È stato aggiunto il supporto per più file e ambienti di configurazione per memorizzare i valori dei parametri predefiniti per AWS SAMCLI i comandi. Per ulteriori informazioni, consulta il file AWS SAMCLI di configurazione.	24 settembre 2020
Support per X-Ray with Step Functions e riferimenti per il controllo dell'accesso a APIs	È stato aggiunto il supporto per X-Ray come fonte di eventi per macchine a stati serverless. Per ulteriori informazioni, vedere la `Tracing`proprietà del tipo di `AWS::Serverless::StateMachine` risorsa. È stato inoltre aggiunto il supporto per i riferimenti durante il controllo dell'accesso a APIs. Per ulteriori informazioni, consulta il tipo di `ResourcePolicyStatement` dati.	17 settembre 2020

Support per Amazon MSK	È stato aggiunto il supporto per Amazon MSK come fonte di eventi per funzioni serverless. Ciò consente ai record in un argomento di Amazon MSK di attivare la funzione Lambda. Per ulteriori informazioni, consulta i tipi di `MSK`dati `EventSource` e i tipi di dati del tipo di `AWS::Serverless::Function` risorsa.	13 agosto 2020
Support per Amazon EFS	È stato aggiunto il supporto per il montaggio di file system Amazon EFS nelle directory locali. Ciò consente al codice della funzione Lambda di accedere e modificare le risorse condivise . Per ulteriori informazioni, consulta la `FileSystemConfigs` proprietà del tipo di `AWS::Serverless::Function` risorsa.	16 giugno 2020

Orchestrazione di applicazioni serverless	È stato aggiunto il supporto per l'orchestrazione delle applicazioni mediante la creazione di macchine a stati Step Functions utilizzando. AWS SAM Per ulteriori informazioni, vedere Orchestrazione AWS delle risorse con AWS Step Functions e tipo di risorsa. `AWS::Serverless::StateMachine`	27 maggio 2020
Creazione di runtime personalizzati	È stata aggiunta la possibilità di creare runtime personalizzati. Per ulteriori informazioni, consulta Creazione di runtime personalizzati.	21 maggio 2020
Costruire livelli	È stata aggiunta la possibilità di creare `LayerVersion` risorse individuali. Per ulteriori informazioni, consulta Creazione di livelli.	19 maggio 2020
CloudFormation Risorse generate	Sono stati forniti dettagli sulle CloudFormation risorse AWS SAM generate e su come farvi riferimento. Per ulteriori informazioni, consulta CloudFormation Risorse generate.	8 aprile 2020

Impostazione delle AWS credenziali	Sono state aggiunte istruzioni per configurare AWS le credenziali nel caso in cui non le abbiate già impostate per l'uso con altri AWS strumenti, come uno dei AWS SDKs o il. AWS CLI Per ulteriori informazioni, consulta Configurazione delle AWS credenziali.	17 gennaio 2020
AWS SAM specifiche e aggiornamenti AWS SAMCLI	È stata migrata la AWS SAM specifica da GitHub. Per ulteriori informazioni, vedere le AWS SAM specifiche. Inoltre, è stato aggiornato il flusso di lavoro di distribuzione con modifiche al `sam deploy`comando.	25 novembre 2019
Nuove opzioni per controllare l'accesso all'API Gateway APIs e agli aggiornamenti dei modelli di policy	Sono state aggiunte nuove opzioni per controllare l'accesso ad API Gateway APIs: autorizzazioni IAM, chiavi API e politiche delle risorse. Per ulteriori informazioni, consulta Controllo dell'accesso all'API Gateway APIs. Sono stati inoltre aggiornati due modelli di policy: RekognitionFacesPolicy e ElasticsearchHttpPostPolicy. Per ulteriori informazioni, vedere modelli di AWS SAM policy.	29 agosto 2019

Guida introduttiva agli aggiornamenti	È stato aggiornato il capitolo introduttivo con istruzioni di installazione migliorate per il AWS SAMCLI tutorial Hello World. Per ulteriori informazioni, consulta Guida introduttiva AWS SAM.	25 luglio 2019
Controllo dell'accesso all'API Gateway APIs	È stato aggiunto il supporto per il controllo dell'accesso all'API Gateway APIs. Per ulteriori informazioni, consulta Controllo dell'accesso all'API Gateway APIs.	21 marzo 2019
`sam publish`Aggiunto al AWS SAMCLI	Il nuovo `sam publish`comando di AWS SAMCLI semplifica il processo di pubblicazione di applicazioni serverless in. AWS Serverless Application Repository Per ulteriori informazioni, vedere Pubblicazione di applicazioni serverless utilizzando. AWS SAMCLI	21 dicembre 2018
Supporto per applicazioni e livelli annidati	È stato aggiunto il supporto per applicazioni e livelli annidati. Per ulteriori informazioni, consultate Utilizzo di applicazioni nidificate e Utilizzo dei livelli.	29 novembre 2018

Aggiunto `sam build` al AWS SAMCLI	Il nuovo `sam build`comando di AWS SAMCLI semplific a il processo di compilazi one di applicazioni serverles s con dipendenze in modo da poter testare e distribui re localmente tali applicazi oni. Per ulteriori informazioni, vedere Creazione di applicazi oni.	19 novembre 2018
Sono state aggiunte nuove opzioni di installazione per AWS SAMCLI	Sono state aggiunte le opzioni di installazione di Linuxbrew (Linux), MSI (Windows) e Homebrew (macOS) per. AWS SAMCLI Per ulteriori informazi oni, vedere Installazione di. AWS SAMCLI	7 novembre 2018
Nuova guida	Questa è la prima versione della Guida per sviluppatori di AWS Serverless Application Model .	17 ottobre 2018

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.