



Guia do Desenvolvedor

AWS Serverless Application Model



AWS Serverless Application Model: Guia do Desenvolvedor

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

As marcas comerciais e imagens de marcas da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestige a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, patrocinados pela Amazon ou ter conexão com ela.

Table of Contents

O que AWS SAM é	1
Quando usar AWS SAM	1
Recursos principais	2
Informações relacionadas	3
Como funciona	3
O que é o AWS SAM especificação do modelo?	4
O que é o AWS SAM projeto e AWS SAM modelo?	4
O que é o AWS SAM CLI?	11
Saiba mais	19
Próximas etapas	19
Conceitos de tecnologia sem servidor	19
Conceitos de tecnologia sem servidor	20
Infraestrutura como código	21
Ferramentas IaC para AWS aplicativos sem servidor	21
Comparar ferramentas de IaC com aplicações com tecnologia sem servidor	22
Saiba mais	23
Introdução	24
Pré-requisitos	24
Inscreva-se para um Conta da AWS	25
Etapa 1: instalar o AWS CLI	25
Etapa 2: use o AWS CLI para configurar AWS credenciais	25
Etapa 3: Instalação (opcional) AWS Toolkit for VS Code	26
Próximas etapas	27
Instale o AWS SAM CLI	27
Instalar o AWS SAM CLI	28
Solução de problemas de erros de instalação do	38
Próximas etapas	39
Opcional: verificar o instalador da CLI do AWS SAM	40
Tutorial Hello World	52
Pré-requisitos	53
Etapa 1: Inicialize o aplicativo de amostra Hello World	54
Etapa 2: crie seu aplicativo	58
Etapa 3: implantar seu aplicativo no Nuvem AWS	60
Etapa 4: Executar seu aplicativo	65

Etapa 5: Interaja com sua função no Nuvem AWS	66
Etapa 6: Modifique e sincronize seu aplicativo com o Nuvem AWS	67
Etapa 7: (opcional) Teste seu aplicativo localmente	70
Etapa 8: Exclua seu aplicativo do Nuvem AWS	72
Solução de problemas	73
Saiba mais	73
Conversão a partir do Lambda	73
Saiba mais	74
Como usar AWS SAM	75
AWS SAM CLI	76
Como AWS SAM CLI os comandos são documentados	76
Configurando o AWS SAM CLI	78
Como criar o arquivo de configuração (o arquivo samconfig)	79
Definir configurações do projeto	79
Configurar credenciais e configurações básicas	84
Comandos principais	84
Teste local com a do AWS SAM CLI	86
Testar AWS SAM aplicativos localmente	86
Testar CloudFormation modelos localmente	86
Testar AWS Aplicativos CDK localmente	86
Teste os aplicativos do Terraform localmente	87
AWS SAM modelo	88
Anatomia do modelo	89
YAML	90
Seções do modelo	90
Próximas etapas	93
Variáveis globais	93
Recursos e propriedades	99
AWS::Serverless: :API	100
AWS::Serverless::Application	161
AWS::Serverless::CapacityProvider	166
AWS::Serverless: :Conector	177
AWS::Serverless: :Função	192
AWS::Serverless: :API GraphQL	358
AWS::Serverless::HttpApi	394
AWS::Serverless::WebSocketApi	426

AWS::Serverless::LayerVersion	446
AWS::Serverless::MicrovmImage	453
AWS::Serverless::NetworkConnector	460
AWS::Serverless::SimpleTable	467
AWS::Serverless::StateMachine	473
AWS::Serverless::WebFunction	515
Recursos gerados	533
Referência gerada CloudFormation recursos	534
Gerado CloudFormation cenários de recursos	535
AWS::Serverless::Api	538
AWS::Serverless::Application	540
AWS::Serverless::CapacityProvider	540
AWS::Serverless::Connector	541
AWS::Serverless::Function	542
AWS::Serverless::GraphQLApi	548
AWS::Serverless::HttpApi	550
AWS::Serverless::MicrovmImage	552
AWS::Serverless::NetworkConnector	553
AWS::Serverless::WebFunction	554
AWS::Serverless::WebSocketApi	555
AWS::Serverless::LayerVersion	558
AWS::Serverless::SimpleTable	558
AWS::Serverless::StateMachine	559
Atributos de recursos compatíveis	560
Exceções	561
Extensões do API Gateway	562
Funções intrínsecas	564
Suporte a extensões de idioma	565
Ativando extensões de idioma	566
Fã:: ForEach	567
Funções intrínsecas suportadas	574
Erros de validação	575
Limitações	576
Telemetria	577
Aplicações de exemplo	578
Processar eventos do DynamoDB	578

Antes de começar	578
Etapa 1: Inicializar o aplicativo	578
Etapa 2: Testar o aplicativo localmente	579
Etapa 3: Empacotar o aplicativo	579
Etapa 4: Implantar um aplicativo	580
Próximas etapas	581
Processar eventos do Amazon S3	581
Antes de começar	581
Etapa 1: Inicializar o aplicativo	582
Etapa 2: Empacotar o aplicativo	582
Etapa 3: Implantar um aplicativo	583
Etapa 4: Testar o aplicativo localmente	584
Próximas etapas	584
Desenvolver a aplicação de	585
Criar aplicação	585
Inicialize um novo aplicativo com tecnologia sem servidor.	586
Opções para o sam init	592
Solução de problemas	592
Exemplos	593
Saiba mais	593
Próximas etapas	594
Definir a infraestrutura	594
Definir os recursos da aplicação	594
Acesso de configuração	596
Controlar o acesso à API	675
Aumentar a eficiência com camadas	689
Reutilizar código	692
Gerenciar eventos baseados em tempo	695
Orquestração de aplicativos	698
Configurar a assinatura de código	700
Validar arquivos AWS SAM de modelo	703
Criar aplicações	704
Introdução a sam build	704
Compilação padrão	720
Personalizar criação	728
Testar a aplicação	762

Introdução a sam local	762
Usar o Local do SAM command	763
Introdução a sam local generate-event	763
Introdução a sam local invoke	770
Introdução a sam local start-api	776
Introdução a sam local start-lambda	783
Invocar funções localmente	786
Arquivo de variável de ambiente	787
Layers	789
Saiba mais	789
Executar o API Gateway localmente	789
Arquivo de variável de ambiente	791
Layers	792
Testar com sam remote test-event	792
Configure o AWS SAM CLI para usar evento de teste remoto do sam	793
Usar o evento de teste remoto do sam command	794
Usando eventos de teste compartilháveis	797
Usando eventos de teste compartilháveis	797
Testar com sam remote invoke	798
Usando o comando de invocação remota do sam	799
Usando o comando de invocação remota do sam	804
Configure o arquivo de configuração do seu projeto	809
Exemplos	809
Links relacionados	825
Automatizar testes de integração	825
Gerar exemplos de cargas úteis	827
Teste e depure funções duráveis	828
Fluxo de trabalho de teste local	828
Cenários de teste comuns	828
Solução de problemas	829
Dicas de depuração	830
Saiba mais	830
Depurar a aplicação	831
Depurar funções localmente	831
Utilizar AWS Kits de ferramentas	832
Em execução AWS SAM localmente no modo de depuração	834

Passar vários argumentos de runtime	835
Valide com cfn-lint	835
Exemplos	836
Saiba mais	836
Implantar a aplicação e os recursos	837
Introdução a sam deploy	837
Pré-requisitos	838
Implantação de aplicativos usando o sam deploy	838
Práticas recomendadas	848
Opções para sam deploy	848
Solução de problemas	849
Exemplos	849
Saiba mais	857
Opções de implantação	858
Como usar a CLI do AWS SAM para implantar manualmente	858
Implemente com CI/CD sistemas e pipelines	858
Implantações graduais	859
Solução de problemas de implantações usando o AWS SAM CLI	859
Saiba mais	789
Implemente com CI/CD sistemas e pipelines	860
O que é um pipeline?	861
Como AWS SAM carrega arquivos locais	861
Gerar um pipeline inicial	869
Personalizar pipelines iniciais	875
Automatizar as implantações	877
Usar autenticação OIDC	882
Introdução a sam sync	885
Detecte e sincronize automaticamente as alterações locais no Nuvem AWS	886
Personalize quais alterações locais são sincronizadas com o Nuvem AWS	887
Prepare seu aplicativo na nuvem para testes e validação	887
Opções para o comando sam sync	888
Solução de problemas	890
Exemplos	890
Saiba mais	897
Monitorar a aplicação	898
Insights da aplicação	898

Configurando o CloudWatch Application Insights com AWS SAM	898
Próximas etapas	902
Como trabalhar com logs	902
Buscando registros por CloudFormation pilha	902
Buscando registros pelo nome da função de eventos do Lambda	903
Registros de rejeitos	903
Visualizando registros para um intervalo de tempo específico	903
Filtragem de logs	903
Destaques de erros	903
Impressão bonita em JSON	903
AWS SAM referência	905
AWS SAM especificação e o AWS SAM modelo	905
AWS SAM CLI Referência de comando	905
AWS SAM Modelos de políticas	906
Tópicos	906
Comandos da CLI do AWS SAM	907
sam build	908
sam delete	914
sam deploy	916
sam init	923
sam list	927
sam local callback	935
sam local execution	941
sam local generate-event	947
SAM invocação local	948
SAM API de inicialização local	955
start-lambda do local do sam	961
logs de spam	967
sam package	970
bootstrap do mesmo pipeline	974
sam pipeline init	979
sam publish	980
sam remote callback	983
sam remote execution	988
invocação remota do sam	995
evento de teste remoto do sam	1001

mesma sincronização	1008
sam traces	1015
sam validate	1017
gerenciamento da CLI do AWS SAM	1018
Arquivo de configuração do AWS SAM CLI	1019
Gerenciar versões do AWS SAM CLI	1026
Configurando AWS credenciais	1035
Telemetria AWS SAMCLI	1037
Solução de problemas	1039
Referência do conector	1045
Tipos de recursos de conector compatíveis	1045
Políticas do IAM criadas por conectores	1055
Instalação do Docker	1079
Instalar Docker	1079
Próximas etapas	1082
Instalação do Finch	1082
Instalar Finch	1083
Configurar Finch por AWS SAM CLI	1085
Verificar a instalação	1086
Parando Finch	1087
Solução de problemas	1088
Próximas etapas	1089
Repositórios de imagens	1089
URIs de repositório de imagem	1090
Exemplos	1092
Implantação gradual	1092
Implantando gradualmente uma função de evento Lambda pela primeira vez	1095
Saiba mais	1096
Observações importantes	1097
2023	1097
Suporte a Terraform	1098
Introdução	1098
Pré-requisitos	1098
Utilizar AWS SAM CLI comandos com Terraform	1099
Configurado para Terraform projetos	1099
Configurado para Terraform Cloud	1105

Uso do AWS SAM CLI com a Terraform	1106
Teste local com a do SAM invocação local	1107
Teste local com a do SAM API de inicialização local	1107
Teste local com a do start-lambda do local do sam	1109
Terraform limitações	1109
Uso do AWS SAM CLI com a Serverless.tf	1110
Referência do Terraform	1110
AWS SAM referência de recursos suportados	1111
Terraform referência específica	1111
metadados do sam	1111
Suporte AWS SAM CLI Terraform	1114
O que é o AWS SAM CLI?	1115
Como faço para usar o AWS SAM CLI por Terraform?	1116
Próximas etapas	1116
Como publicar para que outras pessoas usem	1117
Pré-requisitos	1117
Publicar um novo aplicativo	1119
Etapa 1: Adicionar uma seção de metadados ao AWS SAM modelo	1119
Etapa 2: Empacotar o aplicativo	1119
Etapa 3: publicar o aplicativo	1120
Etapa 4: compartilhar o aplicativo (opcional)	1120
Publicar uma nova versão de um aplicativo existente	1121
Tópicos adicionais	1121
Propriedades da seção de metadados	1121
Propriedades	1121
Casos de uso	1124
Exemplo	1125
Histórico do documento	1127
.....	mclxvii

O que é o AWS Serverless Application Model (AWS SAM)?

AWS Serverless Application Model (AWS SAM) é uma estrutura de código aberto para criar aplicativos sem servidor usando infraestrutura como código (IaC). Com AWS SAM a sintaxe abreviada, os desenvolvedores declaram [CloudFormation](#) recursos e recursos especializados sem servidor que são transformados em infraestrutura durante a implantação. Ao trabalhar com AWS SAM, você interagirá com:

1. AWS SAM CLI- Uma ferramenta de linha de comando que ajuda você a desenvolver, testar e implantar localmente seus aplicativos sem servidor.
2. AWS SAM Modelo - Uma extensão CloudFormation que fornece uma sintaxe simplificada para definir recursos sem servidor.

Quando você usa o `sam init` comando, ele cria um diretório de projeto, que chamaremos de AWS SAM projeto, que normalmente inclui seu AWS SAM modelo, código do aplicativo e outros arquivos de configuração.

Quando usar AWS SAM

AWS SAM é uma solução IaC ideal para cenários em que você deseja um desenvolvimento simplificado sem servidor com todo o poder do CloudFormation. Por exemplo, é possível usar o SAM para:

- Aplicativos sem servidor: você pode usar o SAM para definir rapidamente funções de AWS Lambda evento (anteriormente conhecidas como funções Lambda), funções duráveis do Lambda, APIs do Amazon API Gateway, tabelas do Amazon DynamoDB e outros recursos sem servidor com o mínimo de código.
- CloudFormation aprimoramento: você pode combinar o SAM com CloudFormation modelos existentes para adicionar componentes sem servidor à infraestrutura tradicional. Os recursos do SAM funcionam junto com CloudFormation os recursos padrão no mesmo modelo.
- Desenvolvimento e testes locais: você pode usar a CLI do SAM para testar as funções do Lambda localmente, simular endpoints do API Gateway e depurar aplicações com tecnologia sem servidor em sua máquina de desenvolvimento antes da implantação da AWS.
- CI/CD para ambientes sem servidor: você pode criar pipelines de implantação usando modelos de SAM que geram automaticamente a CloudFormation infraestrutura necessária para ambientes de preparação e produção.

- Migração de recursos criados pelo console: você pode converter funções do Lambda e recursos do API Gateway criados no Console de Gerenciamento da AWS na infraestrutura como código usando modelos do SAM.

Comparando AWS SAM com outras ferramentas de IaC

- Use o SAM em vez de CloudFormation simplificar as definições de recursos sem servidor e, ao mesmo tempo, manter a compatibilidade do modelo.
- Use o SAM em vez de AWS CDK se você preferir uma abordagem declarativa para descrever sua infraestrutura em vez de uma abordagem programática.
- Combine o SAM AWS CDK usando os recursos de teste locais do SAM CLI para aprimorar seus aplicativos de CDK.

Recursos principais

AWS SAM oferece uma variedade de benefícios que melhoram a experiência do desenvolvedor, permitindo que você:

Defina seu código de infraestrutura de aplicativos rapidamente, usando menos código

Crie AWS SAM modelos para definir seu código de infraestrutura de aplicativos sem servidor. Implante seus modelos diretamente CloudFormation para provisionar seus recursos.

Gerencie seus aplicativos sem servidor durante todo o ciclo de vida de desenvolvimento

Use o AWS SAMCLI para gerenciar seu aplicativo sem servidor por meio das fases de criação, compilação, implantação, teste e monitoramento do seu ciclo de vida de desenvolvimento. Para obter mais informações, consulte [AWS SAM CLI](#).

Provisione rapidamente permissões entre recursos com AWS SAM conectores

Use AWS SAM conectores em seus AWS SAM modelos para definir permissões entre seus AWS recursos. AWS SAM transforma seu código nas permissões do IAM necessárias para facilitar sua intenção. Para obter mais informações, consulte [Gerenciando permissões de recursos com AWS SAM conectores](#).

Sincronize as alterações locais com a nuvem à medida que você se desenvolve

Use o AWS SAMCLI `sam sync` comando para sincronizar automaticamente as alterações locais na nuvem, acelerando seus fluxos de trabalho de desenvolvimento e teste na nuvem. Para obter

mais informações, consulte [Introdução ao uso mesma sincronização para sincronizar com Nuvem AWS](#).

Gerencie seus aplicativos Terraform sem servidor

Use o AWS SAMCLI para realizar a depuração e o teste locais de suas camadas e funções do Lambda. Para obter mais informações, consulte [AWS SAM CLI Terraform Suporte](#).

Informações relacionadas

- Para obter informações sobre como AWS SAM funciona, consulte [Como AWS SAM funciona](#).
- Para começar a usar AWS SAM, consulte [Começando com AWS SAM](#).
- Para obter uma visão geral sobre como você pode usar AWS SAM para criar um aplicativo sem servidor, consulte [Como usar AWS SAM](#).

Como AWS SAM funciona

Ao trabalhar com AWS SAM a criação de seu aplicativo sem servidor, você interagirá com os seguintes componentes:

1. [AWS SAM modelo](#)— Um arquivo importante que define seus AWS recursos. Esse modelo inclui a especificação do modelo do AWS SAM : a estrutura de código aberto que vem com uma sintaxe abreviada simplificada que você usa para definir as funções, os eventos, as APIs, as configurações e as permissões da aplicação com tecnologia sem servidor. Esse arquivo está localizado no AWS SAM projeto, que é a pasta do aplicativo criada quando você executa o `sam init` comando.
2. [AWS SAM CLI](#)— Uma ferramenta de linha de comando que você pode usar com seu AWS SAM projeto e com integrações de terceiros compatíveis para criar e executar seus aplicativos sem servidor. AWS SAMCLI é a ferramenta que você usa para executar comandos em seu AWS SAM projeto e, eventualmente, transformá-lo em seu aplicativo sem servidor.

Para expressar recursos, mapeamentos de origem de eventos e outras propriedades que definem seu aplicativo sem servidor, você define recursos e desenvolve seu aplicativo no AWS SAM modelo e em outros arquivos do seu projeto. AWS SAM Você usa o AWS SAMCLI para executar comandos em seu AWS SAM projeto, que é como você inicializa, cria, testa e implanta seu aplicativo sem servidor.

Está começando na tecnologia sem servidor?

Recomendamos que você analise [Conceitos sem servidor para AWS Serverless Application Model](#).

O que é o AWS SAM especificação do modelo?

A especificação do AWS SAM modelo é uma estrutura de código aberto que você pode usar para definir e gerenciar seu código de infraestrutura de aplicativos sem servidor. A especificação do AWS SAM modelo é:

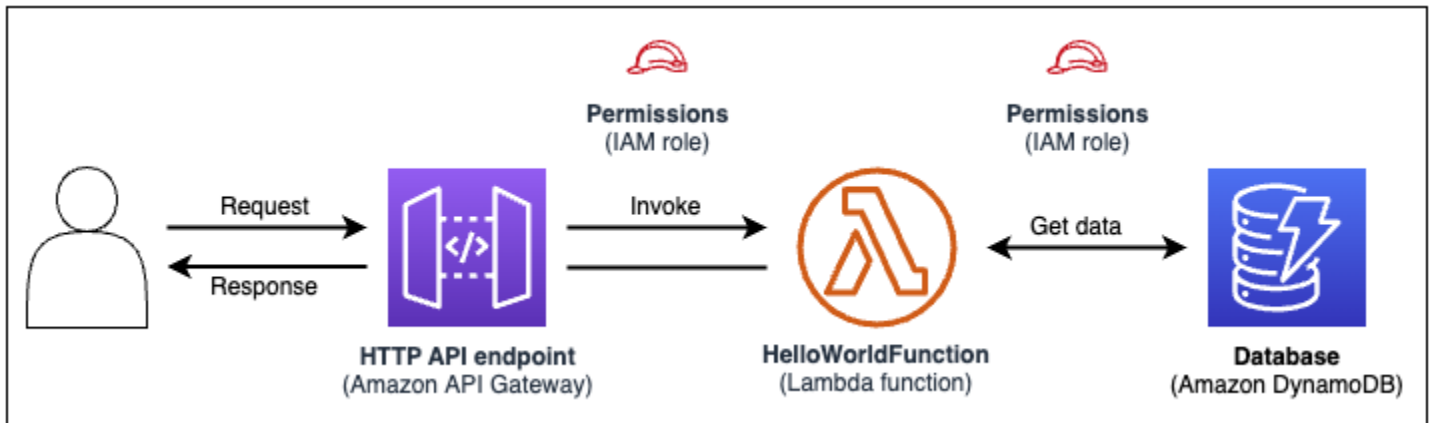
- Baseado em AWS CloudFormation — Você usa a CloudFormation sintaxe diretamente em seu AWS SAM modelo, aproveitando seu amplo suporte para configurações de recursos e propriedades. Se você já está familiarizado CloudFormation, não precisa aprender um novo serviço para gerenciar seu código de infraestrutura de aplicativos.
- Uma extensão de CloudFormation — AWS SAM oferece sua própria sintaxe exclusiva que se concentra especificamente em acelerar o desenvolvimento sem servidor. Você pode usar a CloudFormation AWS SAM sintaxe e no mesmo modelo.
- Uma sintaxe abstrata e abreviada – Usando a AWS SAM sintaxe, você pode definir sua infraestrutura rapidamente, em menos linhas de código e com menor chance de erros. Sua sintaxe é especialmente selecionada para abstrair a complexidade na definição de sua infraestrutura de aplicativos sem servidor.
- Transformacional — AWS SAM faz o trabalho complexo de transformar seu modelo no código necessário para provisionar sua infraestrutura. CloudFormation

O que é o AWS SAM projeto e AWS SAM modelo?

O AWS SAM projeto inclui o AWS SAM modelo que contém a especificação do AWS SAM modelo. Essa especificação é a estrutura de código aberto que você usa para definir sua infraestrutura de aplicativos sem servidor AWS, com alguns componentes adicionais que facilitam o trabalho com eles. Nesse sentido, os AWS SAM modelos são uma extensão dos CloudFormation modelos.

Veja um exemplo de aplicação com tecnologia sem servidor básica. Esse aplicativo processa solicitações para obter todos os itens de um banco de dados por meio de uma solicitação HTTP. Isso consiste nas seguintes partes:

1. Uma função que contém a lógica para processar a solicitação.
2. Uma API HTTP para servir como comunicação entre o cliente (solicitante) e o aplicativo.
3. Um banco de dados para armazenar itens.
4. Permissões para que o aplicativo seja executado com segurança.



O código de infraestrutura desse aplicativo pode ser definido no seguinte modelo 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
      Runtime: nodejs20.x
      Events:
        Api:
          Type: HttpApi
          Properties:
            Path: /
            Method: GET
    Connectors:
      MyConn:
        Properties:
          Destination:
            Id: SampleTable
          Permissions:
            - Read
  SampleTable:
  
```

```
Type: AWS::Serverless::SimpleTable
```

Em 23 linhas de código, a seguinte infraestrutura é definida:

- Uma função usando o AWS Lambda serviço.
- Uma API HTTP usando o serviço Amazon API Gateway.
- Um banco de dados usando o serviço Amazon DynamoDB.
- As permissões AWS Identity and Access Management (IAM) necessárias para que esses serviços interajam entre si.

Para provisionar essa infraestrutura, o modelo é implantado em CloudFormation. Durante a implantação, AWS SAM transforma as 23 linhas de código na CloudFormation sintaxe necessária para gerar esses recursos. AWS O CloudFormation modelo transformado contém mais de 200 linhas de código!

Transformado CloudFormation modelo

```
{
  "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"
  }
]
}
}
```

Ao usar AWS SAM, você define 23 linhas de código de infraestrutura. AWS SAM transforma seu código nas mais de 200 linhas de CloudFormation código necessárias para provisionar seu aplicativo.

O que é o AWS SAM CLI?

AWS SAMCLI é uma ferramenta de linha de comando que você pode usar com AWS SAM modelos e integrações de terceiros compatíveis para criar e executar seus aplicativos sem servidor. Use o AWS SAMCLI para:

- Inicialize rapidamente um projeto de aplicação.
- Crie seu aplicativo para implantação.
- Execute depuração e testes locais.
- Implante o aplicativo.
- Configure os pipelines de CI/CD implantação.
- Monitore e solucione problemas de seu aplicativo na nuvem.
- Sincronize as alterações locais com a nuvem à medida que você se desenvolve.
- E muito mais!

O AWS SAMCLI é melhor utilizado quando usado com AWS SAM CloudFormation modelos. Também funciona com produtos de terceiros, como Terraform.

Inicialize um projeto

Selecione um dos modelos iniciais ou escolha um local de modelo personalizado para começar um novo projeto.

Aqui, usamos o comando `sam init` para inicializar um projeto de aplicação. Seleccionamos o projeto Hello World Example para começar. O AWS SAMCLI baixa um modelo inicial e cria nossa estrutura de diretórios de pastas do projeto.

```
→ 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]:
```

Consulte mais detalhes em [Crie seu aplicativo em AWS SAM](#).

Crie seu aplicativo para implantação

Empacote suas dependências de função e organize o código do projeto e a estrutura de pastas para se preparar para a implantação.

Aqui, usamos o comando `sam build` para preparar nosso aplicativo para implantação. O AWS SAMCLI cria um diretório `.aws-sam` e organiza as dependências e os arquivos do nosso aplicativo para implantação.

```
→ sam-app sam build
Building codeuri: /Users/evzz/Demo/what-is/sam-app/hello_world runtime: python3.9 metadata: {} architecture: 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
```

Consulte mais detalhes em [Criar aplicações](#).

Execute depuração e testes locais

Em sua máquina local, simule eventos, teste APIs, invoque funções e muito mais para depurar e testar seu aplicativo.

Aqui, usamos o comando `sam local invoke` para invocar nosso `HelloWorldFunction` localmente. Para fazer isso, o AWS SAMCLI cria um contêiner local, constrói nossa função, a invoca e gera os resultados. Você pode usar uma aplicação, como o Docker, para executar contêineres no seu computador.

```
→ 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 Billed Duration: 640 ms Memory Size: 128 MB Max Memory Used: 128 MB
{"statusCode": 200, "body": "{\"message\": \"hello world\"}"}
```

Para obter mais detalhes, consulte [Testar a aplicação](#) e [Depurar a aplicação](#).

Implantar o aplicativo

Defina as configurações de implantação do seu aplicativo e implante na AWS nuvem para provisionar seus recursos.

Aqui, usamos o comando `sam deploy --guided` para implantar nosso aplicativo por meio de um fluxo interativo. Ele nos AWS SAMCLI orienta na configuração das configurações de implantação de nosso aplicativo, transforma nosso modelo em CloudFormation e implanta CloudFormation para criar nossos recursos.

```
→ 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
```

Consulte mais detalhes em [Implantar a aplicação e os recursos](#).

Configurar pipelines CI/CD de implantação

Crie pipelines seguros de integração e entrega contínuas (CI/CD) usando um CI/CD sistema compatível.

Aqui, usamos o `sam pipeline init --bootstrap` comando para configurar um pipeline CI/CD de implantação para nosso aplicativo. Ele nos AWS SAMCLI guia por nossas opções e gera os AWS recursos e o arquivo de configuração para usar com nosso CI/CD sistema.

[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]: 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]: ☐

Consulte mais detalhes em [Implemente com CI/CD sistemas e pipelines](#).

Monitore e solucione problemas de seu aplicativo na nuvem

Visualize informações importantes sobre seus recursos implantados, colete registros e utilize ferramentas de monitoramento integradas, como AWS X-Ray.

Aqui, usamos o comando `sam list` para visualizar nossos recursos implantados. Pegamos nosso endpoint de API e o invocamos, o que aciona nossa função. Em seguida, usamos `sam logs` para visualizar os registros da nossa função.

```
→ 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
```

Consulte mais detalhes em [Monitorar a aplicação](#).

Sincronize as alterações locais com a nuvem à medida que você se desenvolve

Conforme você desenvolve em sua máquina local, sincronize automaticamente as alterações na nuvem. Veja rapidamente suas alterações e realize testes e validação na nuvem.

Aqui, usamos o comando `sam sync --watch` para AWS SAMCLI observar as mudanças locais. Modificamos nosso código `HelloWorldFunction`, detectamos AWS SAMCLI automaticamente a alteração e implantamos nossas atualizações na nuvem.


```
-----  
Key           HelloWorldFunctionIamRole  
Description   Implicit IAM Role created for Hello World function  
Value         arn:aws:iam::513423067560:role/sam-app-HelloWorldFunctionRole-15GL0UR9LMT1W  
  
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: {} archi-  
tecture: x86_64 functions: HelloWorldFunction  
Running PythonPipBuilder:CopySource  
Finished syncing Lambda Function HelloWorldFunction.  
█
```

Testes os recursos com suporte na nuvem

Invoke e transmita eventos para recursos compatíveis na nuvem.

Aqui, usamos o comando `sam remote invoke` para testar uma função do Lambda implantada na nuvem. Invocamos nossa função do Lambda e recebemos seus registros e respostas. Com nossa função do Lambda configurada para transmitir respostas, AWS SAMCLI retransmite sua resposta em tempo real.

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

Saiba mais

Para continuar aprendendo sobre isso AWS SAM, consulte os seguintes recursos:

- [O AWS SAM Workshop Completo](#) — Um workshop projetado para ensinar a você muitos dos principais recursos que AWS SAM oferece.
- [Sessões com SAM](#) — Série de vídeos criada por nossa equipe AWS Serverless Developer Advocate sobre o uso. AWS SAM
- [Serverless Land](#) — Site que reúne as informações mais recentes, blogs, vídeos, código e recursos de aprendizado para AWS a tecnologia sem servidor.

Próximas etapas

Se esta é a primeira vez que você usa AWS SAM, consulte [Começando com AWS SAM](#).

Conceitos sem servidor para AWS Serverless Application Model

Saiba mais sobre os conceitos básicos sem servidor antes de usar o AWS Serverless Application Model (AWS SAM).

Conceitos de tecnologia sem servidor

Arquitetura orientada por eventos

Um aplicativo sem servidor consiste em AWS serviços individuais, como computação e Amazon DynamoDB AWS Lambda para gerenciamento de banco de dados, cada um desempenhando uma função especializada. Esses serviços são então vagamente integrados entre si por meio de uma arquitetura orientada por eventos. Para saber mais sobre a arquitetura orientada por eventos, consulte [O que é uma arquitetura orientada por eventos?](#).

infraestrutura como código (IaC)

A infraestrutura como código (IaC) é uma forma de tratar a infraestrutura da mesma forma que os desenvolvedores tratam o código, aplicando o mesmo rigor do desenvolvimento do código do aplicativo ao provisionamento da infraestrutura. Você define sua infraestrutura em um arquivo de modelo AWS, a implanta e AWS cria os recursos para você. Com o IaC, você define no código o que deseja AWS provisionar. Para uma comparação das ferramentas de IaC disponíveis para AWS, consulte [infraestrutura como código \(IaC\)](#).

Tecnologias sem servidor

Com tecnologias AWS sem servidor, você pode criar e executar aplicativos sem precisar gerenciar seus próprios servidores. Todo o gerenciamento do servidor é feito por meio de vários benefícios AWS, como escalabilidade automática e alta disponibilidade incorporada, permitindo que você leve sua ideia à produção rapidamente. Usando tecnologias sem servidor, você pode se concentrar no núcleo do seu produto sem precisar se preocupar com o gerenciamento e a operação de servidores. Para saber mais sobre tecnologia sem servidor, consulte o seguinte:

- [Sem servidor ativado AWS](#)
- [Guia do desenvolvedor de tecnologia sem servidor](#): fornece uma visão geral conceitual do desenvolvimento de tecnologia sem servidor na Nuvem AWS .

Para uma introdução básica aos principais serviços sem servidor, consulte AWS Serverless [101: Understanding the serverless services at Serverless](#) Land.

Aplicação sem servidor

Ao usar AWS SAM, você gerencia recursos relacionados em um aplicativo, que consiste em seu AWS SAM projeto e modelo. Todos os recursos em seu aplicativo são definidos ou mencionados em seu AWS SAM modelo. Quando AWS SAM processa seu modelo, ele cria CloudFormation recursos. Em CloudFormation, os recursos são gerenciados em uma única unidade chamada pilha, e todos os recursos em uma pilha são definidos pelo modelo da pilha. CloudFormation

infraestrutura como código (IaC)

Com a Infraestrutura como Código (IaC), você pode automatizar a implantação e o gerenciamento de seus AWS recursos, incluindo aplicativos sem servidor. A IaC permite que você defina sua infraestrutura usando código, facilitando a criação de versões, o compartilhamento e a replicação de suas implantações. Essa abordagem ajuda você a:

- Acelerar seu ciclo de desenvolvimento
- Simplificar o gerenciamento de configuração
- Melhorar a confiabilidade e a consistência de suas implantações

Ferramentas IaC para AWS aplicativos sem servidor

AWS oferece várias ferramentas de IaC para ajudá-lo a criar, implantar e gerenciar seus recursos de nuvem. Esta seção explica como AWS SAM se encaixa nesse ecossistema e funciona com outras ferramentas de AWS IaC.

AWS CloudFormation

Usando [CloudFormation](#), você pode modelar e provisionar toda a sua AWS infraestrutura com modelos YAML ou JSON. CloudFormation processa automaticamente a criação, as atualizações e a exclusão de recursos. Quando você implanta AWS SAM aplicativos, CloudFormation processa os modelos transformados para criar e gerenciar seus recursos.

AWS Serverless Application Model (AWS SAM)

AWS SAM ajuda você a criar aplicativos sem servidor com sintaxe simplificada para definir recursos sem servidor. Você pode usar AWS SAM modelos para provisionar funções de eventos Lambda, APIs, bancos de dados e fontes de eventos usando a sintaxe YAML concisa. AWS SAM transforma esses modelos em CloudFormation modelos durante a implantação.

Note

Embora seja AWS SAM especializado em aplicativos sem servidor, você pode usar qualquer tipo de CloudFormation recurso em seus modelos. AWS SAM Isso permite a flexibilidade de incluir recursos com tecnologia sem servidor quando necessário.

AWS Kit de desenvolvimento em nuvem (AWS CDK)

Com [AWS CDK](#), você pode definir sua infraestrutura sem servidor usando linguagens de programação conhecidas, como TypeScript Python, Java, C#/.Net ou Go. Você pode usar construções de programação, como loops e condições, para definir sua infraestrutura e AWS CDK gerar CloudFormation modelos para implantação. Você pode usar o AWS SAMCLI para testar e depurar localmente aplicativos criados com AWS CDK. Para saber mais, consulte [Testing CDK applications locally](#).

Comparar ferramentas de IaC com aplicações com tecnologia sem servidor

Ao escolher uma ferramenta de IaC para suas aplicativos com tecnologia sem servidor, considere as preferências da sua equipe, os requisitos do projeto e os fluxos de trabalho existentes. A tabela a seguir compara as principais características das ferramentas de AWS IaC para desenvolvimento sem servidor:

Ferramenta	Uso primário	Melhor para	Funciona com AWS SAM	Quando escolher
CloudFormation	Gerenciando uma AWS infraestrutura complexa	Aplicativos que exigem controle detalhado dos AWS recursos	AWS SAM modelos se transformam em CloudFormation modelos durante a implantação	Para controle preciso sobre recursos que não são com tecnologia sem servidor
AWS SAM	Desenvolvimento de aplicações com tecnologia sem servidor	Equipes criando aplicações com tecnologia sem servidor que usam o Lambda	Funcionalidade nativa	Ao focar principalmente em arquiteturas com tecnologia sem servidor com as funções do Lambda, APIs do API Gateway e outros recursos com tecnologia sem servidor

Ferramenta	Uso primário	Melhor para	Funciona com AWS SAM	Quando escolher
AWS CDK	Definição de infraestrutura usando linguagens de programação	Equipes que preferem linguagens digitadas e abordagem que prioriza o código	Gere AWS SAM modelos e use AWS SAMCLI para testes	Quando você precisa de uma definição de infraestrutura programática ou de uma lógica complexa de configuração de recursos

Note

Embora este guia se concentre em ferramentas AWS de IaC nativas, o Terraform é outra solução popular de IaC que pode ser usada para definir aplicativos sem servidor. A CLI do AWS SAM oferece suporte para testes locais de funções do Lambda definidas no Terraform. Para obter mais informações, consulte [AWS SAM CLI Terraform Suporte](#).

Saiba mais

- Para saber mais sobre DevOps as práticas em AWS, consulte [Introdução DevOps à AWS](#)
- Para obter informações sobre como usar o Lambda com diferentes ferramentas de IaC, consulte [Usar o Lambda com a infraestrutura como código \(IaC\)](#)

Começando com AWS SAM

Comece AWS SAM revisando e concluindo os tópicos desta seção. [AWS SAM Pré-requisitos](#) fornece instruções detalhadas sobre como configurar uma AWS conta, criar usuários do IAM, criar acesso à chave e instalar e configurar o. AWS SAMCLI Após concluir os pré-requisitos, você poderá escolher sua abordagem de desenvolvimento:

- [Instale o AWS SAM CLI](#), o que pode ser feito nos sistemas operacionais Linux, Windows e macOS.
- [Introdução ao AWS SAM usando o VS Code Toolkit](#) para uma experiência de desenvolvimento visual com recursos integrados de depuração e teste.

Após a conclusão da instalação, você pode, opcionalmente, percorrer o tutorial AWS SAM Hello World. Subsequentemente a esse tutorial, você examinará o processo de criação de uma aplicação básica sem servidor com o AWS SAM. Depois de concluir o tutorial, tudo estará pronto para a revisão dos conceitos detalhados em [Como usar AWS Serverless Application Model \(AWS SAM\)](#).

Tópicos

- [AWS SAM Pré-requisitos](#)
- [Instale o AWS SAM CLI](#)
- [Tutorial: implante um aplicativo Hello World com AWS SAM](#)
- [Convertendo funções de eventos Lambda em AWS SAM Aplicativos](#)

AWS SAM Pré-requisitos

Preencha os pré-requisitos a seguir antes de instalar e usar a interface de linha de AWS Serverless Application Model comando ().AWS SAMCLI

Para usar o AWS SAMCLI, você precisa do seguinte:

- Uma AWS conta, credenciais AWS Identity and Access Management (IAM) e um par de chaves de acesso do IAM.
- O AWS Command Line Interface (AWS CLI) para configurar as AWS credenciais.

Tópicos

- [Inscreva-se para um Conta da AWS](#)
- [Etapa 1: instalar o AWS CLI](#)
- [Etapa 2: use o AWS CLI para configurar AWS credenciais](#)
- [Etapa 3: Instalação \(opcional\) AWS Toolkit for VS Code](#)
- [Próximas etapas](#)

Inscreva-se para um Conta da AWS

Para começar AWS, você precisa de um Conta da AWS. Para obter informações sobre como criar um Conta da AWS, consulte [Introdução a um Conta da AWS](#) no Guia de AWS Gerenciamento de contas referência.

Etapa 1: instalar o AWS CLI

AWS CLI É uma ferramenta de código aberto que permite que você interaja com o Serviços da AWS uso de comandos em seu shell de linha de comando. AWS SAMCLIsso requer o AWS CLI para atividades como a configuração de credenciais. Para saber mais sobre o AWS CLI, consulte [O que é o AWS Command Line Interface?](#) no Guia do AWS Command Line Interface usuário.

Para instalar o AWS CLI, consulte [Instalando ou atualizando a versão mais recente do AWS CLI](#) no Guia AWS Command Line Interface do Usuário.

Etapa 2: use o AWS CLI para configurar AWS credenciais

Para configurar as credenciais com o Centro de Identidade do IAM

- Para configurar credenciais com o IAM Identity Center, consulte [Configurar seu perfil com o assistente de AWS configuração de sso](#).

Para configurar credenciais com o AWS CLI

1. Execute o comando `aws configure` da linha de comando.
2. Configure as definições a seguir. Selecione cada um dos links para saber mais:
 - a. [ID da chave de acesso](#)
 - b. [Chave de acesso secreta](#)
 - c. [Região da AWS](#)

d. [Formato da saída](#)

O exemplo a seguir mostra valores de exemplo.

```
$ 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
```

O AWS CLI armazena essas informações em um perfil (uma coleção de configurações) nomeado `default` nos arquivos de configuração `credentials` e `config`. Esses arquivos estão localizados no arquivo `.aws` em seu diretório inicial. Por padrão, as informações desse perfil são usadas quando você executa um comando AWS CLI que não especifica explicitamente um perfil a ser usado. Para obter mais informações sobre o arquivo `credentials`, consulte [Configurações de arquivos de configuração e credenciais](#) no Guia do usuário do AWS Command Line Interface .

Para obter mais informações sobre como configurar credenciais, como usar uma configuração existente e um arquivo de credenciais, consulte [Configuração rápida](#) no Guia do usuário do AWS Command Line Interface .

Etapa 3: Instalação (opcional) AWS Toolkit for VS Code

Para desenvolvedores que preferem um ambiente de desenvolvimento integrado, o AWS Toolkit for VS Code fornece recursos poderosos para o desenvolvimento sem servidor, incluindo depuração `visualCodeLens` , integração e fluxos de trabalho de implantação simplificados.

Pré-requisitos para o desenvolvimento do VS Code

- Visual Studio Code (versão 1.73.0 ou posterior) instalado em seu sistema
- Extensão de suporte à linguagem YAML para VS Code

Para instalar o AWS Toolkit for VS Code

1. Abra o Visual Studio Code
2. Abra a visualização `Extensions` (`Ctrl+Shift+X` ou `Cmd+Shift+X`)
3. Pesquise por "AWS Kit de ferramentas"

4. Instale a extensão “AWS Toolkit” da Amazon Web Services
5. Instale a extensão “YAML” da Red Hat (necessária para os CodeLens recursos do modelo SAM)

Benefícios de usar o VS Code com AWS SAM

- Depuração visual: defina pontos de interrupção e execute suas funções de eventos do Lambda localmente
- CodeLens integração: crie, implante e invoque funções diretamente do seu modelo SAM
- Terminal integrado: acesse AWS CLI os comandos do AWS SAM sem sair do editor
- Validação de modelo: Real-time validação e IntelliSense para modelos SAM

Para obter informações sobre como configurar suas AWS credenciais no VS Code, consulte [Configuração de credenciais](#) no Guia do usuário do AWS Toolkit for VS Code.

Próximas etapas

- Agora você está pronto para instalar o AWS SAMCLI e começar a usar AWS SAM. Para instalar o AWS SAMCLI, consulte [Instale o AWS SAM CLI](#).
- Para configurar o Visual Studio Code para desenvolvimento sem servidor, consulte [Configurando o AWS Toolkit for VS Code](#).

Instale o AWS SAM CLI

Instale a versão mais recente da interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) em sistemas operacionais compatíveis seguindo as instruções em [Etapa 1: instalar o AWS CLI](#).

Para obter informações sobre como gerenciar uma versão atualmente instalada do AWS SAMCLI, incluindo como atualizar, desinstalar ou gerenciar compilações noturnas, consulte. [Gerenciar versões do AWS SAM CLI](#)

 É a primeira vez que você instala o AWS SAM CLI?

Conclua todos os [pré-requisitos](#) na seção anterior antes de prosseguir. Isso inclui:

1. Inscrevendo-se em uma AWS conta.

2. Configurando o acesso seguro AWS a.
3. Como criar um ID de chave de acesso e a chave de acesso secreta.
4. Instalando AWS CLI o.
5. Configurando AWS credenciais.

Tópicos

- [Instalar o AWS SAM CLI](#)
- [Solução de problemas de erros de instalação do](#)
- [Próximas etapas](#)
- [Opcional: verifique a integridade do AWS SAM CLI instalador do](#)

Instalar o AWS SAM CLI

Note

A partir de setembro de 2023, não AWS manterá mais o Homebrew instalador AWS gerenciado do AWS SAMCLI (`aws/tap/aws-sam-cli`). Se você usa Homebrew para instalar e gerenciar o AWS SAMCLI, consulte as seguintes opções:

- Para continuar usando o Homebrew, use o instalador gerenciado pela comunidade. Para obter mais informações, consulte [Gerenciamento o AWS SAM CLI com o Homebrew](#).
- Recomendamos o uso de um dos métodos de instalação primários documentados nesta página. Antes de usar um desses métodos, consulte [Mudar de Homebrew](#).
- Para obter detalhes adicionais, consulte a [Versão de lançamento: 1.121.0](#).

Para instalar o AWS SAMCLI, siga as instruções do seu sistema operacional.

Linux

x86_64 - command line installer

1. Baixe o arquivo [AWS SAM CLI .zip](#) para um diretório de sua escolha.
2. (Opcional) Você pode verificar a integridade do instalador antes da instalação. Para obter instruções, consulte [Opcional: verifique a integridade do AWS SAM CLI instalador do](#).

3. Descompacte os arquivos de instalação em um diretório de sua escolha. Veja a seguir um exemplo, usando o subdiretório `sam-installation`.

 Note

Se o sistema operacional não tiver o comando `unzip` integrado, use um equivalente.

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

4. Instale o AWS SAMCLI executando o `install` executável. Esse executável está localizado no diretório usado na etapa anterior. Veja a seguir um exemplo, usando o subdiretório `sam-installation`:

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

5. Verifique a instalação.

```
$ sam --version
```

Para confirmar uma instalação com êxito, você deve ver uma saída que substitua o seguinte texto entre colchetes pela versão mais recente disponível:

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

arm64 - command line installer

1. Baixe o arquivo [AWS SAM CLI .zip](#) para um diretório de sua escolha.
2. (Opcional) Você pode verificar a integridade do instalador antes da instalação. Para obter instruções, consulte [Opcional: verifique a integridade do AWS SAM CLI instalador do](#).
3. Descompacte os arquivos de instalação em um diretório de sua escolha. Veja a seguir um exemplo, usando o subdiretório `sam-installation`.

 Note

Se o sistema operacional não tiver o comando `unzip` integrado, use um equivalente.

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

4. Instale o AWS SAMCLI executando o `install` executável. Esse executável está localizado no diretório usado na etapa anterior. Veja a seguir um exemplo, usando o subdiretório `sam-installation`:

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

5. Verifique a instalação.

```
$ sam --version
```

Para confirmar uma instalação com êxito, você deve ver uma saída semelhante à seguinte, mas que substitua o texto entre colchetes pela versão mais recente da CLI do SAM:

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

macOS

Etapas de instalação

Use o instalador do pacote para instalar AWS SAMCLI o. Além disso, o instalador do pacote tem dois métodos de instalação que você pode escolher: GUI e Linha de comando. Você pode instalar para todos os usuários ou apenas para o usuário atual. Para instalar para todos os usuários, é necessária autorização de superusuário.

Note

AWS SAM A CLI não é compatível com versões do macOS anteriores ao macOS 13.x. Atualize para uma versão compatível do macOS ou instale a CLI AWS SAM com o Homebrew

GUI - All users

Para baixar o instalador do pacote e instalar a CLI do AWS SAM

Note

Se você instalou anteriormente o AWS SAMCLI por meio do Homebrew ou do pip, você precisa desinstalá-lo primeiro. Para instruções, consulte [Desinstalando o AWS SAMCLI](#).

1. Baixe o pkg do macOS para um diretório de sua escolha:

- Para Macs com processadores Intel, escolha x86_64 — [-x86_64.pkg aws-sam-cli-macos](#)
- Para Macs que executam Apple Silicon, escolha arm64 — [-arm64.pkg aws-sam-cli-macos](#)

Note

Você tem a opção de verificar a integridade do instalador antes da instalação. Para instruções, consulte [Opcional: verifique a integridade do AWS SAM CLI instalador do](#).

2. Execute o arquivo baixado e siga as instruções na tela para continuar com as etapas de Introdução, Leia-me e Licença.
3. Em Seleção de destino, selecione Instalar para todos os usuários deste computador.
4. Em Tipo de instalação, escolha onde AWS SAMCLI será instalado e pressione Instalar. A localização padrão recomendada é `/usr/local/aws-sam-cli`.

Note

Para invocar o AWS SAMCLI com o comando, `sam` o instalador cria automaticamente um symlink entre `/usr/local/bin/sam` e `/usr/local/aws-sam-cli/sam` ou a pasta de instalação que você escolheu.

5. A mensagem AWS SAMCLI será instalada e A instalação foi bem-sucedida será exibida. Pressione Fechar.

Para verificar uma instalação bem-sucedida

- Verifique se o AWS SAMCLI foi instalado corretamente e se seu symlink está configurado executando:

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

GUI - Current user

Para baixar e instalar a CLI do AWS SAM

Note

Se você instalou anteriormente o AWS SAMCLI por meio do Homebrew ou do pip, você precisa desinstalá-lo primeiro. Para instruções, consulte [Desinstalando o AWS SAMCLI](#).

1. Baixe o pkg do macOS para um diretório de sua escolha:
 - Para Macs com processadores Intel, escolha x86_64 — [-x86_64.pkg aws-sam-cli-macos](#)
 - Para Macs que executam Apple Silicon, escolha arm64 — [-arm64.pkg aws-sam-cli-macos](#)

Note

Você tem a opção de verificar a integridade do instalador antes da instalação. Para instruções, consulte [Opcional: verifique a integridade do AWS SAM CLI instalador do](#).

2. Execute o arquivo baixado e siga as instruções na tela para continuar com as etapas de Introdução, Leia-me e Licença.
3. Em Seleção de destino, selecione Instalar somente para mim. Caso não visualize essa opção, vá para a próxima etapa.
4. Para Tipo de instalação, faça o seguinte:

1. Escolha onde o AWS SAMCLI será instalado. O local padrão é `/usr/local/aws-sam-cli`. Selecione um local para o qual você tenha permissões de escrita. Para alterar o local da instalação, selecione local e escolha seu local. Pressione Continuar quando terminar.
2. Se você não teve a opção de escolher Instalar somente para mim na etapa anterior, selecione Alterar local de instalação > Instalar somente para mim e pressione Continuar.
3. Pressione Instalar.
5. A mensagem AWS SAMCLI será instalada e A instalação foi bem-sucedida será exibida. Pressione Fechar.

Como criar um symlink

- Para invocar o AWS SAMCLI com o comando `sam`, você deve criar manualmente um symlink entre o programa AWS SAMCLI e o seu `$PATH`. Crie seu symlink modificando e executando o seguinte comando:

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

- **sudo**— Se o seu usuário tiver permissões de gravação para `$PATH`, não `sudo` é necessário. Caso contrário, o `sudo` será obrigatório.
- **path-to**— Caminho até onde você instalou o AWS SAMCLI programa. Por exemplo, `./Users/myUser/Desktop`
- **path-to-symlink-directory**— Sua variável de `$PATH` ambiente. O local padrão é `/usr/local/bin`.

Para verificar uma instalação bem-sucedida

- Verifique se o AWS SAMCLI foi instalado corretamente e se seu symlink está configurado executando:

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


Command line - All users

Para baixar e instalar a CLI do AWS SAM

Note

Se você instalou anteriormente o AWS SAMCLI por meio do Homebrew ou do pip, você precisa desinstalá-lo primeiro. Para instruções, consulte [Desinstalando o AWS SAMCLI](#).

1. Baixe o pkg do macOS para um diretório de sua escolha:

- Para Macs com processadores Intel, escolha x86_64 — [-x86_64.pkg aws-sam-cli-macos](#)
- Para Macs que executam Apple Silicon, escolha arm64 — [-arm64.pkg aws-sam-cli-macos](#)

Note

Você tem a opção de verificar a integridade do instalador antes da instalação. Para instruções, consulte [Opcional: verifique a integridade do AWS SAM CLI instalador do](#).

2. Modifique e execute o script de instalação:

```
$ 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

Para invocar o AWS SAMCLI com o comando `sam`, o instalador cria automaticamente um symlink entre `/usr/local/bin/sam` e `/usr/local/aws-sam-cli/sam`.

Para verificar uma instalação bem-sucedida

- Verifique se o AWS SAMCLI foi instalado corretamente e se seu symlink está configurado executando:

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

Command line - Current user

Para baixar e instalar a CLI do AWS SAM

Note

Se você instalou anteriormente o AWS SAMCLI por meio do Homebrew ou do pip, você precisa desinstalá-lo primeiro. Para instruções, consulte [Desinstalando o AWS SAMCLI](#).

1. Baixe o pkg do macOS para um diretório de sua escolha:
 - Para Macs com processadores Intel, escolha x86_64 — [-x86_64.pkg aws-sam-cli-macos](#)
 - Para Macs que executam Apple Silicon, escolha arm64 — [-arm64.pkg aws-sam-cli-macos](#)

Note

Você tem a opção de verificar a integridade do instalador antes da instalação. Para instruções, consulte [Opcional: verifique a integridade do AWS SAM CLI instalador do](#).

2. Determine um diretório de instalação no qual você tenha permissões de escrita. Em seguida, crie um arquivo xml usando o modelo e modifique-o para refletir seu diretório de instalação. O diretório já deve existir.

Por exemplo, se você *path-to-my-directory* substituir por `/Users/myUser/Desktop`, a pasta do `aws-sam-cli` programa será instalada lá.

```
<?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. Salve o arquivo xml e verifique se ele é válido executando o seguinte:

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

A saída deve exibir as preferências que serão aplicadas ao programa AWS SAMCLI.

4. Execute o seguinte para instalar o 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.
```

Como criar um symlink

- Para invocar o AWS SAMCLI com o comando `sam`, você deve criar manualmente um symlink entre o programa AWS SAMCLI e o seu `$PATH`. Crie seu symlink modificando e executando o seguinte comando:

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

- **sudo**— Se o seu usuário tiver permissões de gravação para `$PATH`, não `sudo` é necessário. Caso contrário, o `sudo` será obrigatório.

- ***path-to***— Caminho até onde você instalou o AWS SAMCLI programa. Por exemplo, `./Users/myUser/Desktop`
- ***path-to-symlink-directory***— Sua variável de `$PATH` ambiente. O local padrão é `/usr/local/bin`.

Para verificar uma instalação bem-sucedida

- Verifique se o AWS SAMCLI foi instalado corretamente e se seu symlink está configurado executando:

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

Windows

Os arquivos do Windows Installer (MSI) são os arquivos do instalador de pacotes para o sistema operacional Windows.

Siga estas etapas para instalar o AWS SAMCLI usando o arquivo MSI.

1. Baixe a versão AWS SAMCLI [de 64 bits](#).
2. (Opcional) Você pode verificar a integridade do instalador antes da instalação. Para obter instruções, consulte [Opcional: verifique a integridade do AWS SAM CLI instalador do](#).
3. Verifique a instalação.

Depois de concluir a instalação, verifique-a abrindo um novo prompt de comando ou PowerShell prompt. O usuário deverá ser capaz de invocar a `sam` partir da linha de comando.

```
sam --version
```

Após a instalação bem-sucedida do AWS SAMCLI, você deverá ver uma saída como a seguinte:

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

4. Ative caminhos longos (somente Windows 10 e versões mais recentes).

⚠ Important

Eles AWS SAMCLI podem interagir com caminhos de arquivo que excedam a limitação máxima de caminhos do Windows. Isso pode causar erros durante a execução de `sam init` devido às limitações de `MAX_PATH` do Windows 10. Para solucionar esse problema, o novo comportamento de caminhos longos deverá ser configurado.

Para habilitar caminhos longos, consulte [Habilitar caminhos longos no Windows 10, versão 1607 e posteriores](#) na documentação de desenvolvimento de aplicativos do Microsoft Windows.

5. Instale o Git.

Para baixar aplicativos de amostra usando o comando `sam init`, você também deve instalar o Git. Para obter instruções, consulte [Instalando o Git](#).

Solução de problemas de erros de instalação do

Linux

Erro do Docker: “Não é possível conectar-se ao daemon do Docker. O daemon do docker está sendo executado neste host?”

Em alguns casos, para obter permissões para o `ec2-user` acessar o daemon do Docker, pode ser necessário reinicializar sua instância. Se você receber esse erro, tente reinicializar sua instância.

Erro de shell: “comando não encontrado”

Se você receber esse erro, seu shell não conseguirá localizar o AWS SAMCLI executável no caminho. Verifique a localização do diretório em que você instalou o AWS SAMCLI executável e, em seguida, verifique se o diretório está no seu caminho.

AWS SAMCLierro: “/lib64/libc.so.6: versão `GLIBC_2.14' não encontrada (exigida por /usr/local/aws-sam-cli/dist/libz

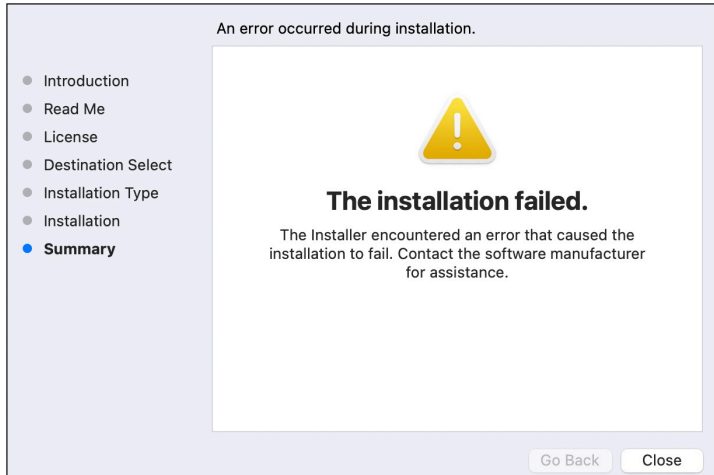
Se você receber esse erro, você está usando uma versão incompatível do Linux e a versão glibc integrada está desatualizada. Tente um dos seguintes:

- Atualize seu host Linux para a versão de 64 bits de uma distribuição recente do CentOS, Fedora, Ubuntu ou Amazon Linux 2.

- Siga as instruções para [Instale o AWS SAM CLI](#).

macOS

A instalação falhou



Se você estiver instalando o AWS SAMCLI para seu usuário e selecionou um diretório de instalação para o qual não tem permissões de escrita, esse erro pode ocorrer. Tente um dos seguintes:

1. Selecione um diretório de instalação diferente para o qual você tenha permissões de escrita.
2. Exclua o instalador. Em seguida, faça o download e execute-o novamente.

Próximas etapas

Para saber mais sobre o AWS SAMCLI e começar a criar os aplicativos com tecnologia sem servidor, consulte o seguinte:

- [Tutorial: implante um aplicativo Hello World com AWS SAM](#) — Step-by-step instruções para baixar, criar e implantar um aplicativo básico sem servidor.
- [O AWS SAM Workshop Completo](#) — Um workshop projetado para ensinar a você muitos dos principais recursos que AWS SAM oferece.
- [AWS SAM exemplos de aplicativos e padrões](#) — Exemplos de aplicativos e padrões de autores da comunidade com os quais você pode experimentar ainda mais.

Opcional: verifique a integridade do AWS SAM CLI instalador do

Ao instalar a interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) usando um instalador de pacotes, você pode verificar sua integridade antes da instalação. Essa etapa é opcional, mas altamente recomendada.

As duas opções de verificação disponíveis para você são:

- Verifique o arquivo de assinatura do instalador do pacote.
- Verifique o valor do hash do instalador do pacote.

Quando disponível para sua plataforma, recomendamos verificar a opção de arquivo de assinatura. Essa opção oferece uma camada extra de segurança, pois os valores-chave são publicados aqui e gerenciados separadamente do nosso repositório GitHub.

Tópicos

- [Verifique o arquivo de assinatura do instalador](#)
- [Verifique o valor do hash](#)

Verifique o arquivo de assinatura do instalador

Linux

arm64 - instalador de linha de comando

AWS SAM usa o [GnuPG](#) para assinar AWS SAMCLI o instalador.zip. A verificação é executada nas seguintes etapas:

1. Use a chave pública primária para verificar a chave pública do signatário.
2. Use a chave pública do signatário para verificar o instalador do pacote AWS SAM CLI.

Para verificar a integridade da chave pública do signatário

1. Copie a chave pública primária e salve-a em sua máquina local como um arquivo .txt. Por exemplo, *.primary-public-key.txt*

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
```

Version: GnuPG v2.0.22 (GNU/Linux)

```
mQINBGRuSzMBEADsqiw0y78w7F4+sshaMFRIwRGNRM94p5Qey2KMZBxekFtoryVD
D9jE0nvupx4tvhfBHz5EcUHCE0dl4MTqdBy6vVAshozgxVb9RE8JpECn5lw7XC69
4Y7Gy1TKKQMEwtDXElkGxIFdUWvWjSnPlzfnoXwQYGeE93CUS3h5dImp22Yk1Ct6
eGGhlcbg1X4L8EpFMj7GvcsU8f7ziVI/PyC1Xwy39Q8/I67ip5eU5ddx0/xHqrbL
YC7+8pJPbRMej2twT2LrcpWWYAbprMtRoa6WfE0/thoo3xhHpIMHdPfAA86ZNGIN
kRLjGUg7jnPTRW40in3pCc8nT4Tfc1QERkHm641gTC/jUvpmQsM6h/FUVP2i5iE/
JHpJcMuL2Mg6zDo3x+3gTCf+Wqz3rZzxB+wQT3yryZs6efcQy7nR0iRxYBxCSXX0
2cNYzsYlB/bYaW8yqWIHD5IqKhW269gp2E5Khs60zgS3CorMb5/xHgXjUCVgcu8a
a8ncdf9fjl3WS5p0ohetPb02ZjWv+MaqrZ0mUIgKbA4RpWZ/fU97P5BW9ylwmIDB
sWy0cMxg8MlvSdLytPieogaM0qMg3u5qXRGBr6Wmevcky0qgnmpGGc5zPiUbtOE8
CnFFqyxBpj5IOng0KZGVihvn+iRrxrv6G07WW092+Dc6m94U0EEiBR7QiOwARAQAB
tDRBV1MgU0FNIENMSSBQcm1tYXJ5IDxhd3Mtc2FtLWNsaS1wcmltYXJ5QGFTYXpv
bi5jb20+iQI/BBMBCQABQJkbksZAhsvBQkHhM4ABwsJCACDAgEGFQgCCQoLBBYC
AwECHgECF4AACGkQQv1fen0tiFqTuhAAzi5+ju5UV0WqHkev0JS008T4QB8HcqAE
SV03mY6/j29knkcL8ubZP/DbpV7QpHPi2PB5qSXsiDTP3IYPbeY78zHSDjljaIK3
njJLMScFeGPfPpwMsuY4nziRIgAtXShPA8N/k4ZJcafnpNqKj7QnPxiC1KaIQWm
p0tvb8msUF3/s0UTa5Ys/1NRhVC0eGg32ogXGdojZA2kHZWdm9udLo4CDrDcrQT7
NtDcJASapXSQL63XfAS3snEc4e1941YxcjFYZ33rel8K9juyDZfi1slWR/L3AviI
QFIaqSHzy0tP1oinUkoVwL8ThevKD3Ag9CZf1ZLzNCV7yqlF8R1hEZ4zcE/3s9E1
WzCFsozb5HfE1AZonmrDh3Sy0EIBMCS6vG5dWnvJrAuSYv2rX38++K5Pr/MIAf0X
DO11rtA+XDshNv91SwSy01t+iClawZAN09IXCiN1r0YcVQ1wzDFwCNWDgkwd0qS0
g0A2f8NF91E5nBbeEuYquo011Vy8+ICbg0Fs9LoWZlnVh7/RyY6ssowiU9vGUNHI
L8f9jqRspIz/Fm3JD86ntZxLVGkeZUz62FqErdohYfkFIVcv7GONTEyrz5HL1npv
FJ0MR0HjrMrZrn0VZnwBKhpblLocTsH+3t5It4ReYEX0f1DIOL/KRwPvjMvBVkXY5
hb1RVDQo0Wc=
=d9oG
-----END PGP PUBLIC KEY BLOCK-----
```

2. Importe a chave pública primária para o seu chaveiro.

```
$ 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. Copie a chave pública do signatário e salve-a em sua máquina local como um arquivo `.txt`. Por exemplo, `.signer-public-key.txt`

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

mQINBGgrxIgBEADGCTudveeeVbWpZDGX9Ni57mBRMVSJwQJ6F/PC34jw0DozxTtd
H+ZPsXLvLwerN/DVXBk8E1qNZ5RGptak8j7MPz+MC3n4txibEJpB61vpjJJM+9cC
7whaMLDT/SbykHYXdrnHqa8KsUJl7rPLJcaRN722NSxvYVMI0A9ffVXV7cfEyZi5
MbYF2Gc9LNBKaknImIva7EKe2/wI6YCqC5yytyfWU5dL6oHXsgTnFL9mhziMxv
WhyzawyJG6EJZsJ3Wl1bIKApN6XZSXYCx0v1BrebYZjD5v0nA+TJaQ7is8atjt0I
DGe0AViw7k08ChTpjA7YG/Uu7n/Fy7qLF/3Nz0b6cBNjemjBazQ3A3KNCpi5hqFM
Uo1WpovLr5CXQnc0B3fBUntIoxi0Sk5MKjH9AbYxfqgEX0ZJB9hAlc6LIEy0Yru6
MMBrIHE86IM11nFE/DeLnCdPG23+1Pttwy0t3+9z5QwmPe3VPpEfCySPcdxHKZSP
rLile8qDznEv1PDvQ0qkBXdmTva2yct5VJkdqy6UrN2xa0dpspHjRUjHh/EY/xMt
fwMUj0KohaZ/1pjotCcksAsZWUxCNcFvLYxuxeytVk4F09Es1hj4ihhLUI+43/ic
3DHSEiext7Q8/UccNArkhSCT7U0vvL7QTuP+pjYTyiC8Vx6g/Y5Ht5+qywARAQAB
tDBBV1MgU0FNIENMSSBUZWFtIDxhd3Mtc2FtLWNsaS1zaWduZXJAYW1hem9uLmNv
bT6JAj8EEwEJACKfAMgrxIgCGy8FCQPCZwAHCwkIBwMCAQYVCAIJCgsEFgIDAQIE
AQIXgAAKCRBA1Kuxvt/atJo6EAC/5C8uJs76W5f5V5XNAMzwBFiZuYpop3DRReCo
P68ZZylokAC9ShRZnIOujpDJt1NS7T/G00BzmcpSPkYEE531ALaXcHWmb9XV0Ajg
J8iboAVBLY0C7mhL/cbJ3v9Q1pXXjyTuhexkJCV8rdHVX/0H8WqTZp1EaRuZ7p8q
PMxddg4ClwstYuh30/dmNd1Gqfb4Fqy8MnV1yGSXR55Jf+sD1N2U04mbpyk/mr1c
f/jFmx86IkCWJVvdXWCVTe2AFy3NHCdLtdnEvFhokC0Qd9wibUWX0j9vq4cVRZT
qamnpAQa0lH3lX0wrjqo8b1AIPoRWSfMtCYvh6kA8MAJv4cAznzXILSLt0E0mzaU
qp5qoy37wNIjeztX6c/q4wss05qTlJhnNu4s3nh5VHu1tooaYpmDxp+ala5TweuM
KZDI4KdAGF4z0Raif+N53nd0YIiXkY0goUbsPCnVrCwoK9PjjyoJncq7c14wN150
IQUZEjyYAQDQZqs5XSfY4zW2cCXatrfzKF7R1kSU14DfJwPUyksoNAQEQuezfXyq
kr0gfIWK1r2nMdqS7WgSx/ypS5kdyrHuPZdaYfEVtuezpoT2lQQx0SZqq1p5hI4R
nqmPte53WXJhbC0tgTIJWn+Uy/d5Q/aSiFD6o8gNLS1BDs1j1ku0XKu1sFCHUcZG
aerdsIkCHAQQAQkABgUCaCvFeAAKCRBC/V96c62IWt3/D/9g0LzWtz62lqJRCsri
wcA/yz88ayKb/GUv3FCT5Nd9JZt8y1tW+AE3SPTdcpfZmt5UN2sRzlj061mpKJzp
eBvYQ9og/34ZrRQqeg8bz02u34LKY11gD0xY0bWtB7TGIXIZZYqZECOPR0Dp6ZzB
abzkRSsJkEk0vbZzJhfWFYs98qfp/G0suFSBE7908Am33DB2jQ/Sol1h1VmNE6Sv
EOgR6+2yEkS2D0+msJMa/V82v9gBTPnxS1NV1d8Dduvt9rbM3LoxiNXUgx/s52yY
U6H3bwUcQ3UY6uRe1UWo5QnMFCdWfg43+q5rmjB4xQyX/BaQyF5K0hZyG+42/pH1
EMw18qN617FTxo3hvQui/cBah1hQ8EVYsGnHDVxLCisbq5iZvp7+XtmMy1Q417gT
EQRo8feJh31e1GWlccVR2pZgIm1PQ69dzzseHnnKkGhifik0bDGo5/IH2EgI1KFn
SG399RMU/qRz0PLVP3i+zSJmhMqG8cnZaUwE5V4P21vQSc1hhd2Hv/C4SVKNqA2i
+oZbHj2vAkuzTTL075AoANebEjPGqwsKZi5mWUE5Pa931JeiXxWZ1EB7rkgQ1PAB
fsDBhYLt4MxCWAhifLMA6uQ4BhXu2RuX0qNfSbqa8jVF6DB6cD8eAHGpPKfJ0130
```

```

LtZnq+n4SfeNbZjD2FQWZR4CrA==
=1Hfs
-----END PGP PUBLIC KEY BLOCK-----

```

4. Importe a chave pública do assinante para o seu chaveiro.

```
$ 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

```

Anote o valor chave da saída. Por exemplo, **.4094ABB1BEDFDAB4**

5. Use o valor da chave para obter e verificar a impressão digital da chave pública do signatário.

```
$ 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>

```

A impressão digital deve corresponder ao mostrado a seguir:

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

Se a string de impressão digital não coincidir, não instale o instalador do AWS SAM CLI. Escale para a AWS SAM equipe [criando um problema](#) no repositório GitHub `aws-sam-cli`.

6. Verifique as assinaturas da chave pública do signatário:

```
$ 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 você vir `1 signature not checked due to a missing key`, repita as etapas anteriores para importar as chaves públicas primária e do signatário para o seu chaveiro.

Você deve ver os valores da chave pública primária e da chave pública do signatário listados.

Agora que você verificou a integridade da chave pública do signatário, você pode usar a chave pública do signatário para verificar o instalador do pacote AWS SAM CLI.

Para verificar a integridade do AWS SAM CLI instalador de pacotes

1. Obtenha o arquivo de assinatura do pacote AWS SAM CLI — Faça o download do arquivo de assinatura para o instalador do pacote AWS SAM CLI usando o seguinte comando:

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

2. Verifique o arquivo de assinatura Passe os nomes dos arquivos `.sig` e `.zip` baixados como parâmetros para o comando `gpg`. Este é um exemplo:

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

A saída deve ser semelhante à seguinte:

```
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
```

- A mensagem **WARNING: This key is not certified with a trusted signature!** pode ser ignorada. Isso ocorre porque não há uma cadeia de confiança entre a chave PGP pessoal (se você tiver uma) e a chave CLI PGP do AWS SAM . Para obter mais informações, consulte [Web of trust](#).
- Se a saída inclui a frase **BAD signature**, verifique se você executou o procedimento corretamente. Se você continuar recebendo essa resposta, encaminhe para a AWS SAM equipe [criando um problema](#) no GitHub repositório `aws-sam-cli` e evite usar o arquivo baixado.

A mensagem `Good signature from "AWS SAM CLI Team <aws-sam-cli-signer@amazon.com>"` significa que a assinatura foi verificada e você pode prosseguir com a instalação.

x86_64 - instalador de linha de comando

AWS SAM usa o [GnuPG](#) para assinar AWS SAMCLI o instalador.zip. A verificação é executada nas seguintes etapas:

1. Use a chave pública primária para verificar a chave pública do signatário.
2. Use a chave pública do signatário para verificar o instalador do pacote AWS SAM CLI.

Para verificar a integridade da chave pública do signatário

1. Copie a chave pública primária e salve-a em sua máquina local como um arquivo `.txt`. Por exemplo, `primary-public-key.txt`

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
```

```
Version: GnuPG v2.0.22 (GNU/Linux)
```

```
mQINBGRuSzMBEADsqiw0y78w7F4+sshaMFRIwRGNRM94p5Qey2KMZBxekFtoryVD
D9jE0nvupx4tvvhfBH5EcUHCE0d14MTqdBy6vVAshozgxVb9RE8JpECn5lw7XC69
4Y7Gy1TKKQMEWtDXElkGxIFdUWwWjSnPlzfnoXwQYGeE93CUS3h5dImp22Yk1Ct6
eGGhlcbg1X4L8EpFMj7GvcsU8f7ziVI/PyC1Xwy39Q8/I67ip5eU5ddx0/xHqrbL
YC7+8pJPbRMej2twT2LrcpWWYAbprMtRoa6WfE0/thoo3xhHpIMHdPfAA86ZNGIN
kRLjGUg7jnPTRW40in3pCc8nT4Tfc1QERkHm641gTC/jUvpmQsM6h/FUVP2i5iE/
JHpJcMuL2Mg6zDo3x+3gTCf+Wqz3rZzxB+wQT3yryZs6efcQy7nR0iRxYBxCSXX0
2cNYzsYLB/bYaW8yqWIHD5IqKhW269gp2E5Khs60zgS3CorMb5/xHgXjUCVgcu8a
a8ncdf9fj13WS5p0ohetPb02ZjWv+MaqrZ0mUIgKbA4RpWZ/fU97P5BW9ylwmIDB
sWy0cMxg8M1vSdLytPieogaM0qMg3u5qXRGBr6WmevktY0qgnmpGGc5zPiUbtOE8
CnFFqyxBpj5I0nG0KZGVihvn+iRxxrv6G07WW092+Dc6m94U0EEiBR7Qi0wARAQAB
tDRBV1MgU0FNIENMSSBQcm1tYXJ5IDxhd3Mt2FtLWNsaS1wcm1tYXJ5QGFtYXpv
bi5jb20+iQI/BBMBCQABQJkbksZAhsvBQkHhM4ABwsJCAcDAgEGFQgCCQoLBBCY
AwECHgECF4AACgkQQQv1fen0tiFqTuhAAzi5+ju5UV0WqHKEv0JS008T4QB8HcqAE
SV03mY6/j29knkcL8ubZP/DbpV7QpHPI2PB5qSXsiDTP3IYPbeY78zHSDjljaIK3
njJLMScFeGPYfPpwMsuY4nzrRIgAtXShPA8N/k4ZJcafnpNqKj7QnPxiC1KaIQWm
p0tvb8msUF3/s0UTa5Ys/1NRhVC0eGg32ogXGdojZA2kHZWdm9udLo4CDrDcrQT7
NtDcJASapXSQL63XfAS3snEc4e1941YxcjFYZ33rel8K9juyDZfi1slWR/L3AviI
QFIaqSHzy0tP1oinUkoVwL8ThevKD3Ag9CZf1ZLzNCV7yq1F8RlhEZ4zcE/3s9E1
```

```
WzCFsozb5HfE1AZonmrDh3Sy0EIBMCS6vG5dWnvJrAuSYv2rX38++K5Pr/MIAfOX
DOI1rtA+XDShNv9lSwSy0lt+iClawZAN09IXCiN1r0YcVQlwzDFwCNWDgkwd0qS0
g0A2f8NF9lE5nBbeEuYquo0l1Vy8+ICbg0Fs9LoWZlnVh7/RyY6ssowiU9vGUnHI
L8f9jqRspIz/Fm3JD86ntZxLVGkeZUz62FqErdohYfkFIVcv7G0NTEyrz5HLlnpv
FJ0MR0HjrMrZrn0VZnwBKhpBLocTsH+3t5It4ReYEX0f1DIOL/KRwPvjMvBVkXY5
hblRVDQo0Wc=
=d9oG
-----END PGP PUBLIC KEY BLOCK-----
```

2. Importe a chave pública primária para o seu chaveiro.

```
$ 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. Copie a chave pública do signatário e salve-a em sua máquina local como um arquivo .txt. Por exemplo, *.signer-public-key.txt*

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

```
mQINBGgrxIgBEADGCTudveeeVbWpZDGX9Ni57mBRMVSJwQJ6F/PC34jw0DozxTtd
H+ZPsXLvLwerN/DVXbK8E1qNZ5RGptak8j7MPz+MC3n4txibEJpB61vpjJJM+9cC
7whaMLDT/SbykHYXdrnHqa8KsUJl7rPLJcaRN722NSxvYVMI0A9ffVXV7cfEyZi5
MbYF2Gc9LnbKaknImIva7EKeeh2/wI6YCqC5yytyfWU5dL6oHXsgTnFL9mhziMxv
WhyzawyJG6EJZsJ3WLlBIKApN6XZSXYCx0vLBrebYZjD5v0nA+TJaQ7is8atjt0I
DGe0AViw7k08ChTpjA7YG/Uu7n/Fy7qLF/3Nz0b6cBNjemjBazQ3A3KNCpi5hqFM
Uo1WpovLr5CXQnc0B3fBUntIOxi0Sk5MKjH9AbYxfqgEX0ZJB9hAlc6LIEy0Yru6
MMBrIHE86IMl1NfE/DeLnCdPG23+1Pttyw0t3+9z5QwmPe3VPPEfCySPcdxHKZSP
rLile8qDznEvLPDvQ0qkBxdMtVa2yct5VJkdqy6UrN2xa0dpspHjRUjHh/EY/xMt
fwMUj0KohaZ/1pjotCcksAsZWUxCNCFvLYxuxeytVk4F09Es1hj4ihhLUI+43/ic
3DHSEiext7Q8/UccNArkhSCT7U0vvL7QTuP+pjYTyiC8Vx6g/Y5Ht5+qywARAQAB
tDBBV1MgU0FNIENMSSBUZWFtIDxhd3Mtc2FtLWNsaS1zaWduZXJAYW1hem9uLmNv
```

```

bT6JAj8EEwEJACKfAMgrxIgCGy8FCQPCZwAHCwkIBwMCAQYVCAIJCgsEFgIDAQIe
AQIXgAAKCRBA1Kuxvt/atJo6EAC/5C8uJs76W5f5V5XNAMzwBFiZuYpop3DRReCo
P68ZZyl0kAC9ShRZnIOujpDJt1NS7T/G00BzmcpSPkYYE531ALaXcHWmb9XV0Ajg
J8iboAVBLY0C7mhL/cbJ3v9Q1pXXjyTuhexkJCV8rdHVX/0H8WqTZplEaRuZ7p8q
PMxddg4ClwstYuH30/dmNd1Gqfb4Fqy8MnV1yGSXR55Jf+sD1N2U04mbpyk/mr1c
f/jFmx86IkCWJVvdXWCVTe2AFy3NHCdLtdnEvFhokC0Qd9wibUWX0j9vq4cVRZT
qamnpAQa0lH3lX0wrjq08b1AIPoRWSfMtCYvh6kA8MAJv4cAznzXILSLt0E0mzaU
qp5qoy37wNIjeztX6c/q4wss05qTlJhnNu4s3nh5VHultooaYpmDxp+ala5TWeuM
KZDI4KdAGF4z0Raif+N53nd0YIiXkY0goUbsPCnVrCwoK9PjjyoJncq7c14wN150
IQUZEjyYAQDQZqs5XSfY4zW2cCXatrf0zKF7R1kSU14DfJwPUyksoNAQEQuezfXyq
kr0gfIWK1r2nMdqS7WgSx/ypS5kdyrHuPZdaYfEVtuezpoT2lQQx0SZqq1p5hI4R
nqmPte53WXJhbC0tgTIJWn+Uy/d5Q/aSIFD6o8gNLS1BDs1j1ku0XKu1sFCHUcZG
aerdsIkCHAQQAQkABgUCaCvFeAAKCRBC/V96c62Iwt3/D/9g0LzWtz62lqJRCsri
wcA/yz88ayKb/GUv3FCT5Nd9JZt8y1tW+AE3SPTdcpfZmt5UN2sRzlj061mpKJzp
eBvYQ9og/34ZrRQqeg8bz02u34LKY11gD0xY0bWtB7TGIxIZZYqZEC0PR0Dp6ZzB
abzKRssJkEk0vbZzJhfWFYs98qfp/G0suFSBE7908Am33DB2jQ/Sollh1VmNE6Sv
EOgR6+2yEkS2D0+msJMa/V82v9gBTPnxS1NV1d8Dduvt9rbM3LoxiNXUgx/s52yY
U6H3bwUcQ3UY6uRe1UWo5QnMFcDwfg43+q5rmjB4xQyX/BaQyF5K0hZyG+42/pH1
EMw18qN617FTxo3hVQUi/cBah1hQ8EVYsGnHDVxLCisbq5iZvp7+XtmMy1Q417gT
EQRo8feJh31e1GLWlccVR2pZgIm1PQ69dzzseHnnKkGhifik0bDGo5/IH2EgI1KFn
SG399RMU/qRz0PLVP3i+zSJmhMqG8cnZaUwE5V4P21vQSc1hhd2Hv/C4SVKNqA2i
+oZbHj2vAkuzTTL075AoANebEjPGqwsKZi5mWUE5Pa931JeiXxWZ1EB7rkgQ1PAB
fsDBhYLt4MxCWAhifLMA6uQ4BhXu2RuX0qNfSbqa8jVF6DB6cD8eAHGpPKfJ0130
LtZnq+n4SfeNbZjD2FQWZR4CrA==
=1Hfs
-----END PGP PUBLIC KEY BLOCK-----

```

4. Importe a chave pública do assinante para o seu chaveiro.

```
$ 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

```

Anote o valor chave da saída. Por exemplo, **.4094ABB1BEDFDAB4**

5. Use o valor da chave para obter e verificar a impressão digital da chave pública do signatário.

```
$ 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>
```

A impressão digital deve corresponder ao mostrado a seguir:

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

Se a string de impressão digital não coincidir, não instale o instalador do AWS SAM CLI. Escale para a AWS SAM equipe [criando um problema](#) no repositório GitHub aws-sam-cli.

6. Verifique as assinaturas da chave pública do signatário:

```
$ 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 você vir 1 signature not checked due to a missing key, repita as etapas anteriores para importar as chaves públicas primária e do signatário para o seu chaveiro.

Você deve ver os valores da chave pública primária e da chave pública do signatário listados.

Agora que você verificou a integridade da chave pública do signatário, você pode usar a chave pública do signatário para verificar o instalador do pacote AWS SAM CLI.

Para verificar a integridade do AWS SAM CLI instalador de pacotes

1. Obtenha o arquivo de assinatura do pacote AWS SAM CLI — Faça o download do arquivo de assinatura para o instalador do pacote AWS SAM CLI usando o seguinte comando:

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

2. Verifique o arquivo de assinatura Passe os nomes dos arquivos .sig e .zip baixados como parâmetros para o comando gpg. Este é um exemplo:

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

A saída deve ser semelhante à seguinte:

```
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
```

- A mensagem **WARNING: This key is not certified with a trusted signature!** pode ser ignorada. Isso ocorre porque não há uma cadeia de confiança entre a chave PGP pessoal (se você tiver uma) e a chave CLI PGP do AWS SAM. Para obter mais informações, consulte [Web of trust](#).
- Se a saída inclui a frase **BAD signature**, verifique se você executou o procedimento corretamente. Se você continuar recebendo essa resposta, encaminhe para a AWS SAM equipe [criando um problema](#) no GitHub repositório `aws-sam-cli` e evite usar o arquivo baixado.

A mensagem **Good signature from "AWS SAM CLI Team <aws-sam-cli-signer@amazon.com>"** significa que a assinatura foi verificada e você pode prosseguir com a instalação.

macOS

Instalador de GUI e linha de comando

Você pode verificar a integridade do arquivo de assinatura do instalador do pacote AWS SAM CLI usando a ferramenta `pkgutil` ou manualmente.

Para verificar usando `pkgutil`

1. Execute o comando a seguir, fornecendo o caminho para o instalador baixado na sua máquina local:

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


Veja um exemplo a seguir:

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

2. Na saída, localize o SHA256 fingerprint para Developer ID Installer: AMZN Mobile LLC. Veja um exemplo a seguir:

```
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. O Developer ID Installer: AMZN Mobile LLC SHA256 fingerprint deve corresponder ao seguinte valor:

```
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 a string de impressão digital não coincidir, não instale o instalador do AWS SAM CLI. Escale para a AWS SAM equipe [criando um problema](#) no repositório GitHub aws-sam-cli. Se a sequência de caracteres da impressão digital corresponder, você poderá continuar usando o instalador de pacotes.

Para verificar o instalador do pacote manualmente

- Consulte [Como verificar a autenticidade das atualizações de software da Apple baixadas manualmente](#) no site de suporte da Apple.

Windows

O AWS SAMCLI instalador é empacotado como MSI arquivos para o sistema Windows operacional.

Para verificar o instalador da integridade do

1. Right-click no instalador e abra a janela Propriedades.
2. Escolha a guia Assinaturas digitais.
3. Em Lista de assinaturas, escolha Amazon Web Services, Inc. e, em seguida, escolha Detalhes.
4. Escolha a guia Geral, se ainda não estiver selecionada, e escolha Visualizar certificado.
5. Selecione a guia Detalhes e Todos na lista suspensa Exibir, se ela ainda não estiver selecionada.
6. Role para baixo até ver o campo Impressão digital e, em seguida, escolha Impressão digital. Isso exibe todo o valor da impressão digital na janela inferior.
7. Combine o valor da impressão digital com o valor a seguir. Se o valor corresponder, prossiga com a instalação. Caso contrário, encaminhe para a AWS SAM equipe [criando um problema](#) no repositório GitHub aws-sam-cli.

```
6c 4e f0 59 67 db 0a 2c 69 87 17 31 ec 9e 51 5d 7c d6 e9 52
```

Verifique o valor do hash

Linux

x86_64 - instalador de linha de comando

Verifique a integridade e a autenticidade dos arquivos do instalador baixados gerando um valor de hash usando o seguinte comando:

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

A saída deve ser como o exemplo a seguir:

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

Compare o valor de SHA-256 hash de 64 caracteres com o da AWS SAMCLI versão desejada nas [notas de AWS SAMCLI lançamento](#) em. GitHub

macOS

Instalador de GUI e linha de comando

Verifique a integridade e a autenticidade do instalador baixado gerando um valor de hash usando o seguinte 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
```

Compare seu valor de SHA-256 hash de 64 caracteres com o valor correspondente no repositório de [notas AWS SAMCLIGitHub de versão](#).

Tutorial: implante um aplicativo Hello World com AWS SAM

Neste tutorial, você usa a interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) para concluir o seguinte:

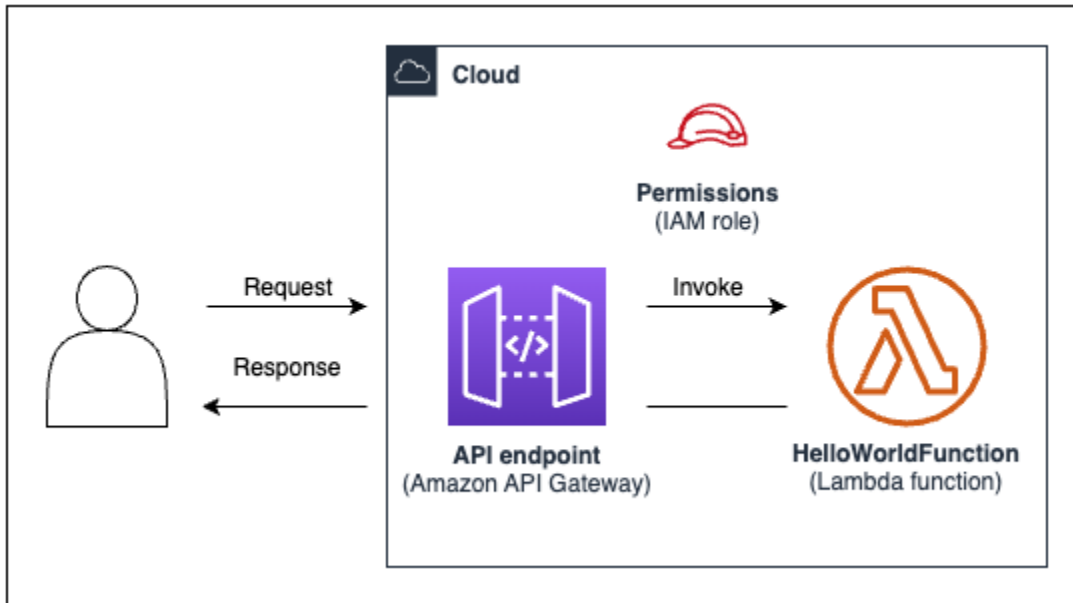
- Inicialize, crie e implante uma amostra do aplicativo Hello World.
- Faça alterações locais e sincronize com AWS CloudFormation.
- Execute testes locais no host de desenvolvimento.
- Exclua a amostra do aplicativo do Nuvem AWS.

A amostra do aplicativo Hello World implementa um backend básico da API. É composto pelos seguintes recursos:

- Amazon API Gateway — Endpoint de API que você usará para invocar sua função.
- AWS Lambda— Função que processa a solicitação GET da API HTTP e retorna uma mensagem `hello world`.

- AWS Identity and Access Management Função (IAM) — Provisiona permissões para que os serviços interajam com segurança.

O diagrama a seguir mostra os componentes deste aplicativo:



Tópicos

- [Pré-requisitos](#)
- [Etapa 1: Inicialize o aplicativo de amostra Hello World](#)
- [Etapa 2: crie seu aplicativo](#)
- [Etapa 3: implantar seu aplicativo no Nuvem AWS](#)
- [Etapa 4: Executar seu aplicativo](#)
- [Etapa 5: Interaja com sua função no Nuvem AWS](#)
- [Etapa 6: Modifique e sincronize seu aplicativo com o Nuvem AWS](#)
- [Etapa 7: \(opcional\) Teste seu aplicativo localmente](#)
- [Etapa 8: Exclua seu aplicativo do Nuvem AWS](#)
- [Solução de problemas](#)
- [Saiba mais](#)

Pré-requisitos

Verifique se você concluiu o seguinte:

- [AWS SAM Pré-requisitos](#)
- [Instale o AWS SAM CLI](#)

Note

Se você preferir usar o Visual Studio Code para desenvolvimento sem servidor, consulte [Criação de aplicativos sem servidor](#) no Guia do usuário do Toolkit for VS AWS Code para obter instruções passo a passo sobre o uso da interface do VS Code.

Etapa 1: Inicialize o aplicativo de amostra Hello World

Nesta etapa, você usará o AWS SAMCLI para criar uma amostra de projeto de aplicativo Hello World em sua máquina local.

Para inicializar o aplicativo Hello World de amostra

1. Na sua linha de comando, execute o seguinte a partir de um diretório inicial de sua escolha:

```
$ sam init
```

Note

Esse comando inicializa a aplicação com tecnologia sem servidor, criando o diretório do projeto. O diretório contém vários arquivos e pastas, incluindo o `template.yaml` arquivo, que é seu AWS SAM modelo.

A versão instalada do Python deve corresponder à propriedade `Runtime` especificada em `template.yaml`. Para cargas de trabalho de produção, recomendamos o uso do Python 3.12 ou versões anteriores que sejam totalmente compatíveis com ambientes. AWS Se as versões do Python não corresponderem, você encontrará erros de compilação.

Para resolver problemas de compatibilidade de versões:

- Especifique um [runtime](#) compatível durante a inicialização: `sam init --runtime python3.9`
- Modifique a propriedade `Runtime` no `template.yaml` após a inicialização

2. O AWS SAMCLI guiará na inicialização de um novo aplicativo. Configure o seguinte:
 1. Selecione AWS Modelos de início rápido para escolher um modelo inicial.
 2. Escolha o modelo Hello World Example e baixe-o.
 3. Use o runtimePython e o tipo de pacote zip.
 4. Para este tutorial, desative o AWS X-Ray rastreamento. Para saber mais, consulte [O que é AWS X-Ray?](#) no Guia do AWS X-Ray desenvolvedor.
 5. Para este tutorial, desative o monitoramento com o Amazon CloudWatch Application Insights. Para saber mais, consulte [Amazon CloudWatch Application Insights](#) no Guia CloudWatch do usuário da Amazon.
 6. Para esse tutorial, opte por não configurar o registro em log estruturado no formato JSON nas funções do Lambda.
 7. Nomeie seu aplicativo como sam-app.

Para usar o AWS SAM CLI fluxo interativo:

- Os colchetes ([]) indicam valores padrão. Deixe sua resposta em branco para selecionar o valor padrão.
- Digite **y** para sim e **n** para não.

A seguir está um exemplo do `sam init` fluxo interativo:

```
$ 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. O AWS SAMCLI baixa seu modelo inicial e cria a estrutura de diretórios do projeto do aplicativo em sua máquina local. A seguir, veja um exemplo da saída AWS SAM CLI:

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. Na sua linha de comando, vá para o diretório sam-app recém-criado. Veja a seguir um exemplo do que o AWS SAMCLI criou:

 Note

Se o comando `tree` não for instalado automaticamente, execute este 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
```


Alguns arquivos importantes a serem destacados:

- `hello_world/app.py`— Contém o código da função do Lambda.
- `hello_world/requirements.txt`— Contém todas as dependências Python que sua função do Lambda exige.
- `samconfig.toml`— Arquivo de configuração do seu aplicativo que armazena os parâmetros padrão usados pelo AWS SAMCLI.
- `template.yaml`— O AWS SAM modelo que contém o código da infraestrutura do seu aplicativo.

Agora você tem um aplicativo com tecnologia sem servidor totalmente criado em sua máquina local!

Etapa 2: crie seu aplicativo

Nesta etapa, você usa o AWS SAMCLI para criar seu aplicativo e se preparar para a implantação. Quando você cria, o AWS SAMCLI cria um diretório `.aws-sam` e organiza as dependências da função, o código do projeto e os arquivos do projeto nele.

Para construir seu aplicativo

- Na sua linha de comando, no diretório do projeto `sam-app`, execute o seguinte:

```
$ sam build
```

Note

Se você não tiver o Python na sua máquina local, use o comando `sam build --use-container` em vez disso. O AWS SAMCLI criará um contêiner Docker que inclui o runtime as dependências da sua função. Esse comando exige o Docker na sua máquina local. Para instalar Docker, consulte [Instalação do Docker](#).

A seguir, veja um exemplo da saída AWS SAM CLI:

```
$ 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

```

Veja a seguir um exemplo abreviado do diretório `.aws-sam` criado pela CLI AWS SAM :

```

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

```

Alguns arquivos importantes a serem destacados:

- `build/HelloWorldFunction`— Contém o código da função do Lambda e as dependências. O AWS SAMCLI cria um diretório para cada função em seu aplicativo.
- `build/template.yaml`— Contém uma cópia do seu AWS SAM modelo que é referenciada CloudFormation na implantação.

- `build.toml`— Arquivo de configuração que armazena valores de parâmetros padrão referenciados pelo AWS SAMCLI ao criar e implantar seu aplicativo.

Agora você está pronto para implantar seu aplicativo no Nuvem AWS.

Etapa 3: implantar seu aplicativo no Nuvem AWS

Note

Essa etapa requer configuração de AWS credenciais. Para obter mais informações, consulte [Etapa 2: use o AWS CLI para configurar AWS credenciais](#) em [AWS SAM Pré-requisitos](#).

Nesta etapa, você usa o AWS SAMCLI para implantar seu aplicativo no Nuvem AWS. Eles AWS SAMCLI farão o seguinte:

- Orientá-lo na definição das configurações do seu aplicativo para implantação.
- Faça o upload dos arquivos do seu aplicativo no Amazon Simple Storage Service (Amazon S3).
- Transforme seu AWS SAM modelo em um CloudFormation modelo. Em seguida, ele carrega seu modelo no CloudFormation serviço para provisionar seus AWS recursos.

Para implantar seu aplicativo

1. Na sua linha de comando, no diretório do projeto `sam-app`, execute o seguinte:

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

2. Siga o fluxo AWS SAMCLI interativo para definir as configurações do aplicativo. Configure o seguinte:

1. O nome da CloudFormation pilha — Uma pilha é uma coleção de AWS recursos que você pode gerenciar como uma única unidade. Para saber mais, consulte [Como trabalhar com pilhas](#) no Guia do AWS CloudFormation usuário.
2. O Região da AWS para onde implantar sua CloudFormation pilha. Para obter mais informações, consulte [CloudFormation Endpoints](#) no AWS CloudFormation Guia do usuário.
3. Neste tutorial, desative confirmar as alterações antes da implantação.

4. Permitir a criação de funções do IAM — Isso permite AWS SAM criar a função do IAM necessária para que o recurso do API Gateway e o recurso da função Lambda interajam.
5. Neste tutorial, desative a desativação da reversão.
6. Permitir HelloWorldFunction sem autorização definida — Essa mensagem é exibida porque seu endpoint do API Gateway está configurado para ser acessível ao público, sem autorização. Como essa é a configuração pretendida para seu aplicativo Hello World, permita que o AWS SAMCLI continue. Para obter mais informações sobre a autorização, consulte [Controle o acesso à API com seu AWS SAM modelo](#).
7. Salvar argumentos no arquivo de configuração — Isso atualizará o arquivo `samconfig.toml` do seu aplicativo com suas preferências de implantação.
8. Selecione o nome do arquivo de configuração padrão.
9. Selecione o ambiente de configuração padrão.

Veja a seguir um exemplo de saída do `sam deploy --guided` fluxo interativo:

```
$ 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. O AWS SAMCLI implanta seu aplicativo fazendo o seguinte:

- O AWS SAMCLI cria um bucket do Amazon S3 e carrega seu diretório `.aws-sam`.
- O AWS SAMCLI transforma seu AWS SAM modelo em CloudFormation e o carrega no CloudFormation serviço.
- CloudFormation provisiona seus recursos.

Durante a implantação, o AWS SAMCLI exibe seu progresso. Veja a seguir um exemplo de saída:

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 ResourceType	LogicalResourceId Replacement
* Modify AWS::Lambda::Function	HelloWorldFunction False
* Modify AWS::ApiGateway::RestApi	ServerlessRestApi False
- Delete AWS::CloudFormation::Stack	AwsSamAutoDependencyLayerNestedSt 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 LogicalResourceId	ResourceType ResourceStatusReason
UPDATE_IN_PROGRESS HelloWorldFunction	AWS::Lambda::Function -
UPDATE_COMPLETE HelloWorldFunction	AWS::Lambda::Function -

```

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-15GL0UR9LMT1W

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

Seu aplicativo agora está implantado e executado no Nuvem AWS!

Etapa 4: Executar seu aplicativo

Nesta etapa, você enviará uma solicitação GET para o endpoint da API e verá a saída da função do Lambda.

Para obter o valor do endpoint da sua API

1. A partir das informações exibidas pelo AWS SAMCLI na etapa anterior, localize a seção **Outputs**. Nesta seção, localize seu recurso `HelloWorldApi` para encontrar o valor do endpoint HTTP. Veja a seguir um exemplo de saída:

```
-----  
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. É possível também usar o comando `sam list endpoints --output json` para obter essas informações. Veja a seguir um exemplo de saída:

```
$ 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']"  
    ]  
  }  
]
```

Para invocar sua função

- Usando seu navegador ou a linha de comando, envie uma solicitação GET para o endpoint da API. Veja a seguir um exemplo usando o comando curl:

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

Etapa 5: Interaja com sua função no Nuvem AWS

Nesta etapa, você usa o AWS SAMCLI para invocar sua função do Lambda no Nuvem AWS.

Para invocar sua função de evento Lambda na nuvem

1. Anote o `LogicalResourceId` das suas funções na etapa anterior. Deveria ser `HelloWorldFunction`.
2. Na sua linha de comando, no diretório do projeto `sam-app`, execute o seguinte:

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

3. O AWS SAMCLI invoca sua função na nuvem e retorna uma resposta. Veja a seguir um exemplo de saída:

```
$ 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\"}}%
```

Etapa 6: Modifique e sincronize seu aplicativo com o Nuvem AWS

Nesta etapa, você usa o AWS SAMCLI `sam sync --watch` comando para sincronizar as alterações locais com Nuvem AWS o.

Para usar o `sam sync`

1. Na sua linha de comando, no diretório do projeto `sam-app`, execute o seguinte:

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

2. O AWS SAMCLI solicita que você confirme que está sincronizando uma pilha de desenvolvimento. Como o `sam sync --watch` comando implanta automaticamente as alterações locais no Nuvem AWS em tempo real, nós o recomendamos somente para ambientes de desenvolvimento.

O AWS SAMCLI executa uma implantação inicial antes de começar a monitorar as alterações locais. Veja a seguir um exemplo de saída:

```
$ 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_pt15g6_80000gr/T/tmpq3x9vh63.
Execute the following command to deploy the packaged template
```

```
sam deploy --template-file /var/folders/45/5ct135bx3fn2551_pt15g6_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		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		
			ack

```

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-15GL0UR9LMT1W

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.

```

Depois, você modificará o código da função do Lambda. O AWS SAMCLI detectará automaticamente essa alteração e sincronizará seu aplicativo com Nuvem AWS o.

Para modificar e sincronizar seu aplicativo

1. No IDE de sua preferência, abra o arquivo `sam-app/hello_world/app.py`.
2. Altere o message e salve o arquivo. Este é um exemplo:

```
import json
```

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

3. O AWS SAMCLI detecta sua alteração e sincroniza seu aplicativo com o. Nuvem AWS Veja a seguir um exemplo de saída:

```
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. Para verificar sua alteração, envie novamente uma solicitação GET para o endpoint da API.

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

Etapa 7: (opcional) Teste seu aplicativo localmente

Note

Esta etapa é opcional.

Important

Esse comando requer o Docker na sua máquina local. Para usar o AWS SAM CLI para testes locais, você deve ter o Docker instalado e configurado. Para obter mais informações, consulte [Instalação do Docker](#).

Nesta etapa, você usa o AWS SAMCLI `local` comando para testar seu aplicativo localmente. Para fazer isso, o AWS SAMCLI cria um ambiente local usando o Docker. Esse ambiente local emula o ambiente de execução baseado em nuvem da sua função do Lambda.

Você fará o seguinte:

1. Crie um ambiente local para sua função do Lambda e invoque-o.
2. Hospede o endpoint da sua API HTTP localmente e use-o para invocar sua função do Lambda.

Para invocar sua função de evento Lambda localmente

1. Na sua linha de comando, no diretório do projeto `sam-app`, execute o seguinte:

```
$ sam local invoke
```

2. AWS SAMCLI Isso cria um contêiner Docker local e invoca sua função. Veja a seguir um exemplo de saída:

```
$ 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\"}"}
```

Para hospedar sua API localmente

1. Na sua linha de comando, no diretório do projeto `sam-app`, execute o seguinte:

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

2. O AWS SAMCLI cria um contêiner Docker local para sua função do Lambda e cria um servidor HTTP local para simular o endpoint da sua API. Veja a seguir um exemplo de saída:

```
$ 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. Usando seu navegador ou a linha de comando, envie uma solicitação GET para o endpoint da API local. Veja a seguir um exemplo usando o comando curl:

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

Etapa 8: Exclua seu aplicativo do Nuvem AWS

Nesta etapa, você usa o AWS SAMCLI `sam delete` comando para excluir seu aplicativo do Nuvem AWS.

Para excluir seu aplicativo do Nuvem AWS

1. Na sua linha de comando, no diretório do projeto `sam-app`, execute o seguinte:

```
$ sam delete
```

2. O AWS SAMCLI pedirá que você confirme. Em seguida, ele excluirá o bucket e a pilha do Amazon S3 do seu aplicativo. CloudFormation Veja a seguir um exemplo de saída:

```
$ 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
```

Solução de problemas

Para solucionar o problema do AWS SAMCLI, consulte [AWS SAM CLI solução de problemas](#).

Saiba mais

Para continuar aprendendo sobre isso AWS SAM, consulte os seguintes recursos:

- [O Workshop AWS SAM Completo](#) – Um workshop projetado para ensinar a você muitos dos principais atributos que AWS SAM oferece.
- [Sessões com o SAM](#) — Série de vídeos criada por nossa equipe AWS Serverless Developer Advocate sobre o uso. AWS SAM
- [Serverless Land](#) — Site que reúne as informações mais recentes, blogs, vídeos, código e recursos de aprendizado para a tecnologia sem servidor AWS .

Convertendo funções de eventos Lambda em AWS SAM Aplicativos

Se você tiver uma função no console Lambda que deseja gerenciar no VS Code como infraestrutura como código (IaC), você pode transferi-la para o VS Code e depois convertê-la em um modelo. AWS SAM Com sua função convertida em um AWS SAM modelo, você pode controlar o controle de versão de sua infraestrutura, automatizar implantações, depurar funções remotamente e manter

ambientes consistentes em todo o ciclo de vida de desenvolvimento. Com recursos de depuração local e remota, você também pode solucionar problemas com mais eficiência examinando seu código, inspecionando variáveis e definindo pontos de interrupção localmente e na nuvem. AWS

Para obter instruções sobre como mover suas funções do console e configurar o desenvolvimento local, consulte [Desenvolver funções do Lambda localmente com o VS Code](#) no Guia do Desenvolvedor do AWS Lambda . Siga estas instruções para mover uma função de evento Lambda do console para o VS Code e depois convertê-la em um AWS SAM modelo. Após fazer isso, é possível:

- Depure remotamente sua função na AWS nuvem. Para obter detalhes, consulte [Depure remotamente as funções do Lambda com o Visual Studio Code](#) no Guia do desenvolvedor do AWS Lambda
- Gerenciar os recursos e permissões associados como infraestrutura como código. Para obter mais detalhes, consulte [Usar o Lambda com a infraestrutura como código \(IaC\)](#) no Guia do desenvolvedor do AWS Lambda
- Implante e atualize seu aplicativo usando AWS SAMCLI o. Para obter detalhes, consulte [AWS SAM CLI](#).
- Projetar visualmente sua arquitetura com tecnologia sem servidor usando o Compositor de Infraestrutura. Para obter mais detalhes, consulte o [Guia do desenvolvedor do Compositor de Infraestrutura](#).

Saiba mais

Para continuar aprendendo sobre isso AWS SAM, consulte os seguintes recursos:

- [O Workshop AWS SAM Completo](#) – Um workshop projetado para ensinar a você muitos dos principais atributos que AWS SAM oferece.
- [Sessões com o SAM](#) — Série de vídeos criada por nossa equipe AWS Serverless Developer Advocate sobre o uso. AWS SAM
- [Serverless Land](#) — Site que reúne as informações mais recentes, blogs, vídeos, código e recursos de aprendizado para a tecnologia sem servidor AWS .

Como usar AWS Serverless Application Model (AWS SAM)

As principais ferramentas que você usa para desenvolver seu aplicativo são o AWS SAM modelo AWS SAMCLI e o AWS SAM projeto (que é o diretório do projeto do aplicativo). Você usa essas ferramentas para:

1. [Desenvolver a aplicação de](#) (isso inclui inicializar a aplicação, definir recursos e criar a aplicação).
2. [Testar a aplicação.](#)
3. [Depurar a aplicação.](#)
4. [Implantar a aplicação e os recursos.](#)
5. [Monitorar a aplicação.](#)

AWS SAM cria seu AWS SAM projeto depois de executar o `sam init` comando e concluir o fluxo de trabalho subsequente. Você define seu aplicativo sem servidor adicionando código ao seu AWS SAM projeto. Embora seu AWS SAM projeto consista em um conjunto de arquivos e pastas, o arquivo mais importante nele é seu AWS SAM modelo (`template.yaml`). Nesse modelo, você escreve o código para expressar os recursos, os mapeamentos da origem do evento e outras propriedades que definem a aplicação com tecnologia sem servidor.

O AWS SAMCLI contém um repositório de comandos que você usa em seu AWS SAM projeto. Mais especificamente, AWS SAMCLI é o que você usa para criar, transformar, implantar, depurar, empacotar, inicializar e sincronizar seu AWS SAM projeto. Em outras palavras, é o que você usa para transformar seu AWS SAM projeto em seu aplicativo sem servidor.

Para obter mais informações sobre como acessar essas ferramentas, consulte as seguintes seções:

[AWS SAM CLI](#): saiba mais sobre o AWS SAM CLI e como usá-lo para criar e executar suas aplicações com tecnologia sem servidor.

[AWS SAM projeto e AWS SAM modelo](#) - Saiba mais sobre a estrutura do AWS SAM projeto e como usar o AWS SAM modelo para definir seu aplicativo sem servidor.

AWS SAM CLI

AWS Serverless Application Model A interface de linha de comando (AWS SAMCLI) é uma ferramenta de linha de comando para desenvolvimento local e teste de aplicativos sem servidor. AWS SAMCLI permite que você crie, transforme, implante, depure, empacote, inicialize e sincronize seus aplicativos sem servidor localmente antes de implantá-los na nuvem.

A CLI do AWS SAM funciona com aplicações com tecnologia sem servidor que são definidos usando diferentes estruturas e ferramentas de infraestrutura como código (IaC), com vários níveis de suporte:

- AWS SAM modelos — fornece suporte nativo com o conjunto completo de recursos, incluindo recursos locais de teste, depuração, empacotamento e implantação.
- AWS Aplicativos CDK — Suporta testes locais de Lambda Event Functions depois de sintetizar AWS o aplicativo CDK em modelos usando o comando `cdk CloudFormation synth`.
- CloudFormation modelos — Oferece compatibilidade direta porque AWS SAM se estende CloudFormation, oferecendo suporte a recursos sem servidor definidos em modelos padrão CloudFormation .
- Aplicações do Terraform: fornece suporte limitado para criação e testes locais de funções do Lambda. Requer que você gere artefatos AWS SAM de modelo que representem suas funções do Terraform-defined Lambda.

Para obter o suporte de recursos mais abrangente e a experiência simplificada do desenvolvedor, recomendamos o uso de AWS SAM modelos nativos.

Tópicos

- [Como AWS SAM CLI os comandos são documentados](#)
- [Configurando o AWS SAM CLI](#)
- [AWS SAM CLI Comandos principais da do](#)
- [Teste local com a do AWS SAM CLI](#)

Como AWS SAM CLI os comandos são documentados

Os comandos AWS SAMCLI são documentados usando o seguinte formato:

- Prompt — O prompt Linux é documentado por padrão e exibida como (\$). Para comandos específicos do Windows, o (>) é usado como prompt. Não inclua prompt quando você digitar comandos.
- Diretório: quando comandos devem ser executados de um diretório específico, o nome do diretório é mostrado antes do símbolo do prompt.
- Entrada do usuário o texto do comando inserido na linha de comando é formatado como **user input**.
- Texto substituível — O texto variável, como nomes de arquivos e parâmetros, é formatado como. *replaceable text* Em comandos ou comandos de várias linhas, em que é necessária uma entrada específica do teclado, a entrada do teclado também pode ser exibida como texto substituível. Por exemplo, *.ENTER*
- Saída — A saída retornada como resposta ao comando é formatada como `computer output`.

O comando `sam deploy` e a saída a seguir são um exemplo:

```
$ 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` é o comando que você insere na linha de comando.
2. **`sam deploy --guided --template`** deve ser fornecido como está.
3. `template.yaml` pode ser substituído pelo nome do arquivo específico.
4. A saída começa em `Configuring SAM deploy`.
5. Na saída, `y` indique **ENTER** os valores substituíveis que você fornece.

Configurando o AWS SAM CLI

Um dos benefícios AWS SAM é que ele otimiza o tempo do desenvolvedor removendo tarefas repetitivas. AWS SAMCLI inclui um arquivo de configuração nomeado `samconfig` para essa finalidade. Por padrão, nenhuma configuração para o AWS SAMCLI é necessária, mas você pode atualizar seu arquivo de configuração para permitir que você execute comandos com menos parâmetros, permitindo, em vez disso, AWS SAM referenciar seus parâmetros personalizados em seu arquivo de configuração. Os exemplos na tabela a seguir mostram como você pode otimizar os comandos:

Original	Otimizado com samconfig
<code>sam build --cached --parallel --use-containers</code>	<code>sam build</code>
<code>sam local invoke --env-vars locals.json</code>	<code>sam local invoke</code>
<code>sam local start-api --env-vars locals.json --warm-containers EAGER</code>	<code>sam local start-api</code>

A CLI do AWS SAM fornece um conjunto de comandos para ajudar os desenvolvedores a criar, desenvolver e implantar aplicações sem servidor. Cada um desses comandos é configurável com sinalizadores opcionais com base nas preferências da aplicação e do desenvolvedor. Para obter mais informações, consulte o [AWS SAMCLI conteúdo em GitHub](#)

Os tópicos desta seção mostram como criar [Arquivo de configuração do AWS SAM CLI](#) e personalizar suas configurações padrão para otimizar o tempo de desenvolvimento da aplicação sem servidor.

Tópicos

- [Como criar o arquivo de configuração \(o arquivo `samconfig`\)](#)
- [Definir configurações do projeto](#)
- [Configurar credenciais e configurações básicas](#)

Como criar o arquivo de configuração (o arquivo `samconfig`)

O arquivo AWS SAMCLI de configuração (nome do arquivo `samconfig`) é um arquivo de texto que normalmente usa a estrutura TOML, mas também pode estar em YAML. Ao usar um modelo de início AWS rápido, esse arquivo é criado quando você executa o `sam init` comando. Você pode atualizar esse arquivo ao implantar uma aplicação usando o comando `sam deploy --guided`.

Depois que a implantação estiver concluída, o arquivo `samconfig` conterá um perfil denominado `default` se você tiver usado os valores padrão. Quando você executa novamente o `deploy` comando, AWS SAM aplica as configurações armazenadas desse perfil.

A vantagem do `samconfig` arquivo é que AWS SAM armazena as configurações de qualquer outro comando disponível além do comando `deploy`. Além desses valores criados em uma nova implantação, há vários atributos que você pode definir no `samconfig` arquivo que podem simplificar outros aspectos do fluxo de trabalho do desenvolvedor AWS SAMCLI.

Definir configurações do projeto

Você pode especificar configurações específicas do projeto, como valores de parâmetros de AWS SAMCLI comando, em um arquivo de configuração para usar com o. AWS SAMCLI Para obter mais informações sobre este arquivo de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

Usando arquivos de configuração

Os arquivos de configuração são estruturados por ambiente, comando e valor de parâmetro. Para obter mais informações, consulte [Noções básicas de arquivos de configuração](#).

Para configurar um novo ambiente

1. Especifique seu novo ambiente em seu arquivo de configuração.

Veja a seguir um exemplo de especificação de um novo ambiente `prod`:

TOML

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

YAML

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

2. Especifique os valores dos parâmetros como pares de valores-chave na seção de parâmetros do arquivo de configuração.

Veja a seguir um exemplo de como especificar o nome da pilha do seu aplicativo para o ambiente prod.

TOML

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

YAML

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

3. Use a opção `--config-env` para especificar o ambiente a ser usado.

Veja um exemplo a seguir:

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

Para configurar valores de parâmetros

1. Especifique o comando AWS SAMCLI para o qual você gostaria de configurar os valores dos parâmetros. Para configurar valores de parâmetros para todos os comandos AWS SAMCLI, use o identificador `global`.

Veja a seguir um exemplo de especificação de valores de parâmetros para o `default` comando do ambiente `sam deploy`:

TOML

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

YAML

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

Veja a seguir um exemplo de especificação de valores de parâmetros para todos os AWS SAMCLI comandos no `default` ambiente:

TOML

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

YAML

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

2. Você também pode especificar valores de parâmetros e modificar seu arquivo de configuração por meio do fluxo interativo AWS SAMCLI.

A seguir está um exemplo do fluxo interativo `sam deploy --guided`:

```
$ 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

```

Para obter mais informações, consulte [Criando e modificando arquivos de configuração](#).

Exemplos

Exemplo básico de TOML

Este é um exemplo de um arquivo de configuração da `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
```

Exemplo básico de YAML

Este é um exemplo de um arquivo de configuração da `samconfig.yaml`:

```
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
```

Configurar credenciais e configurações básicas

Use o AWS Command Line Interface (AWS CLI) para definir configurações básicas, como AWS credenciais, nome da região padrão e formato de saída padrão. Depois de configurado, você pode usar essas configurações com o AWS SAMCLI. Para saber mais, consulte o seguinte no AWS Command Line Interface Guia de usuário:

- [Noções básicas de configuração](#)
- [Configurações do arquivo de configuração e credenciais](#)
- [Perfis nomeados para o AWS CLI](#)
- [Usando um perfil nomeado habilitado para o IAM Identity Center](#)

Para obter instruções de configuração rápida, consulte [Etapa 2: use o AWS CLI para configurar AWS credenciais](#).

AWS SAM CLI Comandos principais da do

A CLI do AWS SAM tem alguns comandos básicos que você usa para criar, compilar, testar, implantar e sincronizar aplicações sem servidor. A tabela abaixo lista esses comandos e fornece links com mais informações para cada um.

Para obter uma lista completa de comandos da CLI do AWS SAM, consulte [AWS SAMCLIReferência de comando](#).

Command	O que ela faz	Tópicos relacionados
sam build	Prepara um aplicativo para as etapas subsequentes do fluxo de trabalho do desenvolvedor, como testes locais ou implantação na AWS nuvem.	<ul style="list-style-type: none">• Introdução à construção com AWS SAM• sam build
sam deploy	Implanta um aplicativo na AWS nuvem usando CloudFormation.	<ul style="list-style-type: none">• Introdução à implantação com AWS SAM• sam deploy

Command	O que ela faz	Tópicos relacionados
sam init	Fornecer opções para inicializar e criar uma aplicação sem servidor.	<ul style="list-style-type: none"> • Crie seu aplicativo em AWS SAM • sam init
sam local	Fornecer subcomandos para testar as aplicações sem servidor localmente.	<ul style="list-style-type: none"> • Introdução aos testes com o Local do SAM command • sam local generate-event • SAM invocação local • SAM API de inicialização local • start-lambda do local do sam
sam remote invoke	Fornecer uma forma de interagir com AWS os recursos suportados na AWS nuvem.	<ul style="list-style-type: none"> • Introdução aos testes na nuvem com invocação remota do sam • invocação remota do sam
sam remote test-event	Fornecer uma maneira de acessar e gerenciar eventos de teste compartilháveis para suas funções de eventos AWS Lambda.	<ul style="list-style-type: none"> • Introdução aos testes em nuvem com evento de teste remoto do sam • evento de teste remoto do sam
sam sync	Fornecer opções para sincronizar rapidamente as alterações do aplicativo local com a AWS nuvem.	<ul style="list-style-type: none"> • Introdução ao uso mesma sincronização para sincronizar com Nuvem AWS • mesma sincronização

Teste local com a do AWS SAM CLI

A CLI do AWS SAM permite que você teste aplicações com tecnologia sem servidor localmente em diferentes ferramentas de infraestrutura como código (IaC). Este guia explica como usar a CLI do AWS SAM para testes locais com várias estruturas de IaC.

Os principais benefícios do uso da CLI do AWS SAM para testes locais incluem:

- Desenvolvimento rápido - Teste as alterações no código sem precisar implantá-las AWS
- Eficiência de custos - Desenvolva e teste sem incorrer AWS em custos
- Capacidade off-line: trabalhe em suas aplicações sem uma conexão com a Internet
- Depuração simplificada - Percorra o código da Lambda Event Function localmente usando um depurador
- Teste realista - Teste seus aplicativos usando emulação local de serviços AWS

Testar AWS SAM aplicativos localmente

Para obter informações sobre como testar aplicativos definidos usando AWS SAM modelos, consulte [Teste e depuração de aplicativos sem servidor neste guia](#).

Testar CloudFormation modelos localmente

Para usar os recursos AWS SAMCLI de teste locais do CloudFormation, adicione a AWS SAM transformação ao seu CloudFormation modelo. Para obter mais informações, consulte [Anatomia do modelo AWS SAM](#) para:

- Adicionando AWS SAM transformação aos CloudFormation modelos
- Noções básicas sobre compatibilidade de modelos
- Explorar a sintaxe de recursos com tecnologia sem servidor

Testar AWS Aplicativos CDK localmente

Você pode usar o AWS SAMCLI para testar aplicativos AWS CDK localmente depois de sintetizar o aplicativo AWS CDK em CloudFormation modelos usando o comando `cdk synth`.

Para obter mais informações, consulte [Teste e crie aplicativos AWS CDK localmente com o guia do AWS SAMCLI](#) desenvolvedor do AWS CDK. Isso inclui:

- [Conceitos básicos sobre testes locais](#)
- [Teste local de aplicativos AWS CDK com AWS SAM](#)

Teste os aplicativos do Terraform localmente

A CLI do AWS SAM oferece suporte a projetos do Terraform e do Terraform Cloud. Você pode usá-lo para realizar a depuração e o teste locais de: funções do Lambda e camadas e as APIs HTTP e REST do Amazon API Gateway.

Para configurar seu ambiente e conhecer todos os recursos disponíveis, consulte o [Suporte do Terraform](#) neste guia. Isso inclui:

- [Conceitos básicos do suporte do Terraform para a CLI do AWS SAM](#)
- [Usar a CLI do AWS SAM with Terraform para depuração e teste locais](#)

AWS SAM modelo

Depois de executar o `sam init` comando e concluir o fluxo de trabalho subsequente, AWS SAM cria o diretório do projeto do aplicativo, que é o seu AWS SAM projeto. Você define seu aplicativo sem servidor adicionando código ao seu AWS SAM projeto. Embora seu AWS SAM projeto consista em um conjunto de arquivos e pastas, o arquivo com o qual você trabalha principalmente é o seu AWS SAM modelo (`template.yaml`). Nesse modelo, você escreve o código para expressar os recursos, os mapeamentos da origem do evento e outras propriedades que definem a aplicação com tecnologia sem servidor.

Note

Um elemento-chave do AWS SAM modelo é a especificação do AWS SAM modelo. Essa especificação fornece a sintaxe abreviada que, quando comparada a CloudFormation, permite que você use menos linhas de código para definir os recursos, mapeamentos de origem de eventos, permissões, APIs e outras propriedades do seu aplicativo sem servidor.

Esta seção fornece detalhes sobre como você usa as seções no AWS SAM modelo para definir tipos de recursos, propriedades de recursos, tipos de dados, atributos de recursos, funções intrínsecas e extensões do API Gateway.

AWS SAM os modelos são uma extensão dos CloudFormation modelos, com tipos de sintaxe exclusivos que usam sintaxe abreviada com menos linhas de código do que CloudFormation. Isso acelera o desenvolvimento na criação de uma aplicação sem servidor. Para obter mais informações, consulte [AWS SAM recursos e propriedades](#). Para obter a referência completa dos CloudFormation modelos, consulte [Referência do CloudFormation modelo](#) no Guia AWS CloudFormation do usuário.

Ao desenvolver, muitas vezes você achará vantajoso dividir o código do aplicativo em arquivos separados para melhor organizar e gerenciar seu aplicativo. Um exemplo básico disso é usar um arquivo separado para seu código de função de AWS Lambda evento em vez de ter esse código em seu AWS SAM modelo. Faça isso organizando o código da função Lambda em um subdiretório do seu projeto e referenciando seu caminho local em seu modelo (`()`). AWS Serverless Application Model

Tópicos

- [AWS SAM anatomia do modelo](#)

- [AWS SAM recursos e propriedades](#)
- [Gerado CloudFormation recursos para AWS SAM](#)
- [Atributos de recursos suportados por AWS SAM](#)
- [Extensões do API Gateway para AWS SAM](#)
- [Funções intrínsecas para AWS SAM](#)
- [CloudFormation suporte a extensões de linguagem](#)

AWS SAM anatomia do modelo

Um arquivo AWS SAM de modelo segue rigorosamente o formato de um arquivo de CloudFormation modelo, que é descrito em [Anatomia do modelo](#) no Guia do AWS CloudFormation usuário. As principais diferenças entre arquivos AWS SAM de modelo e arquivos CloudFormation de modelo são as seguintes:

- Declaração de transformação. A declaração Transform: `AWS::Serverless-2016-10-31` é necessária para arquivos AWS SAM de modelo. Essa declaração identifica um arquivo CloudFormation de modelo como um arquivo AWS SAM de modelo. Para obter mais informações sobre transformações, consulte [Transformações](#) no Guia do usuário do AWS CloudFormation .
- Seção global. A Globals seção é exclusiva de AWS SAM. Ele define propriedades que são comuns a todas as suas funções sem servidor e APIs. Todos os `AWS::Serverless::StateMachine` recursos `AWS::Serverless::Function` `AWS::Serverless::Api` `AWS::Serverless::CapacityProvider`, `AWS::Serverless::HttpApi`, e herdam as propriedades definidas na Globals seção. Para obter mais informações sobre essa seção, consulte [Seção Global do AWS SAM modelo](#).
- Seção de recursos. Nos AWS SAM modelos, a Resources seção pode conter uma combinação de CloudFormation recursos e AWS SAM recursos. Para obter mais informações sobre CloudFormation recursos, consulte a [referência de tipos de AWS recursos e propriedades](#) no Guia AWS CloudFormation do usuário. Para obter mais informações sobre AWS SAM recursos, consulte [AWS SAM recursos e propriedades](#).

Todas as outras seções de um arquivo de AWS SAM modelo correspondem à seção CloudFormation de arquivo de modelo com o mesmo nome.

YAML

O exemplo a seguir mostra um modelo de estilhaço formatado em 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
```

Seções do modelo

AWS SAM os modelos podem incluir várias seções principais. Somente as seções `Transform` e `Resources` são obrigatórias.

Você pode incluir seções de modelo em qualquer ordem. No entanto, se você estiver usando extensões de linguagem, deverá adicionar `AWS::LanguageExtensions` antes da transformação sem servidor (ou seja, antes de `AWS::Serverless-2016-10-31`), conforme mostrado no seguinte exemplo:

```
Transform:
```

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

À medida que você cria o modelo, pode ser útil usar a ordem lógica mostrada na lista a seguir. Isso ocorre porque os valores em uma seção podem se referir aos valores de uma seção anterior.

[Transformar \(obrigatório\)](#)

Para AWS SAM modelos, você deve incluir esta seção com um valor de `AWS::Serverless-2016-10-31`.

Transformações adicionais são opcionais. Para obter mais informações sobre transformações, consulte [Transformações](#) no Guia do usuário do AWS CloudFormation .

[Globais \(opcional\)](#)

Propriedades que são comuns a todas as suas funções sem servidor e APIs tabelas simples. Todos os `AWS::Serverless::StateMachine` recursos `AWS::Serverless::Function` `AWS::Serverless::Api` `AWS::Serverless::CapacityProvider`, `AWS::Serverless::HttpApi` e herdam as propriedades definidas na `Globals` seção.

Esta seção é exclusiva de AWS SAM. Não há uma seção correspondente nos modelo CloudFormation s.

[Description \(opcional\)](#)

Uma sequência de texto que descreve o modelo.

Esta seção corresponde diretamente à `Description` seção de CloudFormation modelos.

[Metadata \(opcional\)](#)

Os objetos que fornecem informações adicionais sobre o modelo.

Esta seção corresponde diretamente à `Metadata` seção de CloudFormation modelos.

[Parameters \(opcional\)](#)

Os valores a serem passados para seu modelo no runtime (ao criar ou atualizar uma pilha). Você pode fazer referência a parâmetros nas seções `Resources` e `Outputs` do modelo. Os objetos declarados na `Parameters` seção fazem com que o `sam deploy --guided` comando apresente solicitações adicionais ao usuário.

Os valores transmitidos usando o parâmetro `--parameter-overrides` do comando `sam deploy` e as entradas no arquivo de configuração têm precedência sobre as entradas no arquivo de modelo do AWS SAM. Para obter mais informações sobre o comando `sam deploy`, consulte [sam deploy](#) na AWS SAM Referência de comandos da CLI. Para obter mais informações sobre o arquivo de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

Mappings (opcional)

Um mapeamento de chaves e valores associados que você pode usar para especificar valores de parâmetros condicionais, semelhante a uma tabela de pesquisa. Você pode vincular uma chave a um valor correspondente usando a função intrínseca `Fn::FindInMap` nas seções `Resources` e `Outputs`.

Esta seção corresponde diretamente à `Mappings` seção de CloudFormation modelos.

Condições (opcional)

As condições que controlam se determinados recursos são criados ou se determinadas propriedades de recursos são atribuídas a um valor durante a criação ou a atualização da pilha. Por exemplo, condicionalmente, você pode criar um recurso que depende de se a pilha é de um ambiente de teste ou de produção.

Esta seção corresponde diretamente à `Conditions` seção de CloudFormation modelos.

Resources (obrigatório)

Os recursos da pilha e suas propriedades, como uma instância do Amazon Elastic Compute Cloud (Amazon EC2) ou um bucket do Amazon Simple Storage Service (Amazon S3). Você pode fazer referência a recursos nas seções `Resources` e `Outputs` do modelo.

Essa seção é semelhante à seção `Resources` de modelos CloudFormation. Nos AWS SAM modelos, essa seção pode conter AWS SAM recursos além CloudFormation dos recursos.

Outputs (opcional)

Os valores que são retornados sempre que você visualiza as propriedades da pilha. Por exemplo, você pode declarar uma saída para o nome de um bucket do S3 e, em seguida, chamar o comando `aws cloudformation describe-stacks` AWS Command Line Interface (AWS CLI) para ver o nome.

Esta seção corresponde diretamente à seção `Outputs` de modelos CloudFormation.

Próximas etapas

Para baixar e implantar um aplicativo sem servidor de amostra que contém um arquivo AWS SAM de modelo, consulte [Começando com AWS SAM](#) e siga as instruções em. [Tutorial: implante um aplicativo Hello World com AWS SAM](#)

Seção Global do AWS SAM modelo

Às vezes, os recursos que você declara em um AWS SAM modelo têm configurações comuns. Por exemplo, você pode ter um aplicativo com vários recursos `AWS::Serverless::Function` que têm configurações `Runtime`, `Memory`, `VPCConfig`, `Environment` e `Cors` idênticas. Em vez de duplicar essas informações em todos os recursos, você pode declará-las uma vez na seção `Globals` e permitir que seus recursos as herdem.

A `Globals` seção oferece suporte aos seguintes tipos de AWS SAM recursos:

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

Exemplo:

```
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
```

Neste exemplo, HelloWorldFunction e ThumbnailFunction usam “nodejs12.x” para Runtime, “180” segundos para Timeout e “index.handler” para Handler. HelloWorldFunction adiciona a variável de ambiente MESSAGE, além do TABLE_NAME herdado. ThumbnailFunction herda todas as propriedades Globals e adiciona uma fonte de eventos de API.

Recursos e propriedades compatíveis

AWS SAM oferece suporte aos seguintes recursos e propriedades.

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

PropagateTags:
SecurityPolicy:
TracingEnabled:
Variables:

CapacityProvider:
InstanceRequirements:
KmsKeyArn:
LoggingConfig:
ManagedResourceTags:
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:

MicrovmImage:

BaseImageArn:

BaseImageVersion:

BuildRoleArn:

Hooks:

EnvironmentVariables:

EgressNetworkConnectors:

CpuConfigurations:

Resources:

AdditionalOsCapabilities:

Logging:

Tags:

PropagateTags:

NetworkConnector:

OperatorRole:

Tags:

PropagateTags:

Note

Os recursos e propriedades que não fazem parte da lista anterior não são aceitas. Alguns motivos para não apoiá-los incluem: 1) Eles abrem possíveis problemas de segurança ou 2) Eles tornam o modelo difícil de entender.

APIs implícitas

AWS SAM cria APIs implícitas quando você declara uma API na `Events` seção. Você pode usar `Globals` para substituir todas as propriedades das APIs implícitas.

Propriedades substituíveis

Os recursos podem substituir as propriedades que você declara na seção `Globals`. Por exemplo, você pode adicionar novas variáveis a um mapa de variáveis de ambiente ou substituir variáveis declaradas globalmente. Mas o recurso não pode remover uma propriedade especificada na seção `Globals`.

De forma mais geral, a seção `Globals` declara propriedades que todos os seus recursos compartilham. Alguns recursos podem fornecer novos valores para propriedades declaradas globalmente, mas não podem removê-las. Se alguns recursos usam uma propriedade, mas outros não, você não deve declará-los na seção `Globals`.

As seções a seguir descrevem como a substituição funciona para diferentes tipos de dados.

Os tipos de dados primitivos são substituídos

Os tipos de dados primitivos incluem cadeias de caracteres, números, booleanos, e assim por diante.

O valor especificado na seção `Resources` substitui o valor na seção `Globals`.

Exemplo:

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



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

O Runtime para MyFunction foi definido como python3.9.

Os mapas são mesclados

Os mapas também são conhecidos como dicionários ou coleções de pares de chave-valor.

As entradas do mapa na seção Resources são mescladas com as entradas do mapa global. Se houver duplicatas, a entrada da seção Resource substituirá a entrada da seção Globals.

Exemplo:

```
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
```

As variáveis de ambiente de MyFunction são definidas da seguinte forma:

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

As listas são aditivas

As listas também são conhecidas como matrizes.

As entradas da lista na seção `Globals` são anexadas à lista na seção `Resources`.

Exemplo:

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

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

Os `SecurityGroupIds` para `MyFunction VpcConfig` estão definidos da seguinte forma:

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

AWS SAM recursos e propriedades

Esta seção descreve os tipos de recursos e propriedades que são específicos de AWS SAM. Você define esses recursos e propriedades usando a AWS SAM sintaxe abreviada. AWS SAM também suporta tipos CloudFormation de recursos e propriedades. Para obter informações de referência sobre todos os tipos de AWS recursos e propriedades CloudFormation e AWS SAM suporte, consulte a [referência de tipos de AWS recursos e propriedades](#) no Guia AWS CloudFormation do usuário.

Tópicos

- [AWS::Serverless::API](#)
- [AWS::Serverless::Application](#)

- [AWS::Serverless::CapacityProvider](#)
- [AWS::Serverless::Connector](#)
- [AWS::Serverless::Function](#)
- [AWS::Serverless::API GraphQL](#)
- [AWS::Serverless::HttpApi](#)
- [AWS::Serverless::WebSocketApi](#)
- [AWS::Serverless::LayerVersion](#)
- [AWS::Serverless::MicrovmImage](#)
- [AWS::Serverless::NetworkConnector](#)
- [AWS::Serverless::SimpleTable](#)
- [AWS::Serverless::StateMachine](#)
- [AWS::Serverless::WebFunction](#)

AWS::Serverless::API

Cria uma coleção de recursos e métodos do Amazon API Gateway que podem ser invocados por meio de endpoints HTTPS.

Um [AWS::Serverless::API](#) recurso não precisa ser adicionado explicitamente a um modelo de definição de aplicativo AWS sem servidor. Um recurso desse tipo é criado implicitamente a partir da união de eventos de API definidos nos recursos [AWS::Serverless::Function](#) definidos no modelo que não se referem a um recurso [AWS::Serverless::API](#).

Um [AWS::Serverless::API](#) recurso deve ser usado para definir e documentar o uso da API OpenApi, o que fornece mais capacidade de configurar os recursos subjacentes do Amazon API Gateway.

Recomendamos que você use CloudFormation ganchos ou políticas do IAM para verificar se os recursos do API Gateway têm autorizadores vinculados a eles para controlar o acesso a eles.

[Para obter mais informações sobre o uso de CloudFormation ganchos, consulte Registrando ganchos no guia do usuário da CloudFormation CLI e no repositório apigw-enforce-authorizer.](#)
GitHub

Para obter mais informações sobre o uso de políticas do IAM, consulte [Exigir que as rotas de API tenham autorização](#) no Guia do desenvolvedor do API Gateway.

 Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

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
  EndpointAccessMode: String
  EndpointConfiguration: EndpointConfiguration
  FailOnWarnings: Boolean
  GatewayResponses: Map
  MergeDefinitions: Boolean
  MethodSettings: MethodSettings
  MinimumCompressionSize: Integer
  Mode: String
  Models: Map
  Name: String
  OpenApiVersion: String
  PropagateTags: Boolean
```

```
Policy: JSON  
SecurityPolicy: String  
StageName: String  
Tags: Map  
TracingEnabled: Boolean  
Variables: Map
```

Propriedades

AccessLogSetting

Configura a configuração do log de acesso para um estágio.

Digite: [AccessLogSetting](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AccessLogSetting](#) propriedade de um `AWS::ApiGateway::Stage` recurso.

AlwaysDeploy

Sempre implanta a API, mesmo quando nenhuma alteração na API foi detectada.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ApiKeySourceType

A origem da chave de API para as solicitações de medição de acordo com um plano de uso. Os valores válidos são `HEADER` e `AUTHORIZER`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ApiKeySourceType](#) propriedade de um `AWS::ApiGateway::RestApi` recurso.

Auth

Configure a autorização para controlar o acesso à sua API API Gateway.

Para obter mais informações sobre como configurar o acesso usando, AWS SAM consulte [Controle o acesso à API com seu AWS SAM modelo](#). Para ver um exemplo de como substituir um autorizador global, consulte [the section called “Substitua um autorizador global para sua API REST do Amazon API Gateway”](#).

Digite: [ApiAuth](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

BinaryMediaTypes

Lista de tipos de MIME que sua API pode retornar. Use isso para ativar o suporte binário para APIs.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [BinaryMediaTypes](#) propriedade de um `AWS::ApiGateway::RestApi` recurso. A lista de `BinaryMediaTypes` é adicionada ao CloudFormation recurso e ao documento da OpenAPI.

CacheClusterEnabled

Indica se o cache está habilitado para o estágio. Para armazenar respostas em cache, você também deve definir `CachingEnabled` como `true` abaixo `MethodSettings`.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CacheClusterEnabled](#) propriedade de um `AWS::ApiGateway::Stage` recurso.

CacheClusterSize

O tamanho do cluster de cache da etapa.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CacheClusterSize](#) propriedade de um `AWS::ApiGateway::Stage` recurso.

CanarySetting

Configure uma configuração de canário para um estágio de uma implantação regular.

Digite: [CanarySetting](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CanarySetting](#) propriedade de um `AWS::ApiGateway::Stage` recurso.

Cors

Gerencie o compartilhamento de Cross-origin recursos (CORS) para todas as suas APIs do API Gateway. Especifique o domínio a ser permitido como uma string ou especifique um dicionário com configuração adicional do Cors.

Note

O CORS exige que AWS SAM você modifique sua definição de OpenAPI. Crie uma definição de OpenAPI em linha no `DefinitionBody` para ativar o CORS.

Para obter mais informações sobre o CORS, consulte [Habilitar o CORS para um recurso API REST para a API Gateway](#) no Guia do desenvolvedor do API Gateway.

Tipo: String | [CorsConfiguration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

DefinitionBody

Especificação OpenAPI que descreve sua API. Se o `DefinitionUri` nem o `DefinitionBody` for especificado, o SAM gerará um `DefinitionBody` para você com base na configuração do seu modelo.

Para fazer referência a um arquivo OpenAPI local que define sua API, use a transformação `AWS::Include`. Para saber mais, consulte [Como AWS SAM carrega arquivos locais](#).

Type: JSON

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Body](#) propriedade de um `AWS::ApiGateway::RestApi` recurso. Se determinadas propriedades forem fornecidas, o conteúdo poderá ser inserido ou modificado no `DefinitionBody` antes de ser passado para CloudFormation. As propriedades incluem `Auth`, `BinaryMediaTypes`, `Cors`, `GatewayResponses`, `Models`, e um `EventSource` tipo API para um arquivo `AWS::Serverless::Function`.

DefinitionUri

Uri do Amazon S3, caminho de arquivo local ou objeto de localização do documento OpenAPI que define a API. O objeto do Amazon S3 a que essa propriedade faz referência deve ser um arquivo OpenAPI válido. Se o `DefinitionUri` nem o `DefinitionBody` for especificado, o SAM gerará um `DefinitionBody` para você com base na configuração do seu modelo.

Se um caminho de arquivo local for fornecido, o modelo deverá passar pelo fluxo de trabalho que inclui o comando `sam deploy` ou `sam package` para que a definição seja transformada adequadamente.

As funções intrínsecas não são suportadas em OpenApi arquivos externos referenciados por `DefinitionUri`. Em vez disso, use a `DefinitionBody` propriedade com a [Transformação Include](#) para importar uma OpenApi definição para o modelo.

Tipo: String | [ApiDefinition](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [BodyS3Location](#) propriedade de um `AWS::ApiGateway::RestApi` recurso. As propriedades aninhadas do Amazon S3 têm nomes diferentes.

Description

Uma descrição do recurso da API.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::ApiGateway::RestApi` recurso.

DisableExecuteApiEndpoint

Especifica se os clientes podem invocar sua API usando o endpoint `execute-api` padrão. Por padrão, os clientes podem invocar sua API com o padrão `https://{api_id}.execute-api.{region}.amazonaws.com`. Para exigir que os clientes usem um nome de domínio personalizado para invocar sua API, especifique `True`.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [DisableExecuteApiEndpoint](#) propriedade de um `AWS::ApiGateway::RestApi` recurso. Ele é passado diretamente para a propriedade `disableExecuteApiEndpoint` de uma extensão [x-amazon-apigateway-endpoint-configuration](#), que é adicionada à propriedade [Body](#) de um recurso `AWS::ApiGateway::RestApi`.

Domain

Configura um domínio personalizado para essa API do API Gateway.

Digite: [DomainConfiguration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

EndpointConfiguration

O tipo de endpoint de uma API REST.

Digite: [EndpointConfiguration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [EndpointConfiguration](#) propriedade de um `AWS::ApiGateway::RestApi` recurso. As propriedades aninhadas de configuração têm nomes diferentes.

EndpointAccessMode

O modo de acesso para o endpoint da API REST. Obrigatório ao usar políticas de segurança aprimoradas (aquelas prefixadas com `SecurityPolicy_`).

Valores válidos: STRICT ou BASIC

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EndpointAccessMode](#) propriedade de um `AWS::ApiGateway::RestApi` recurso.

FailOnWarnings

Especifica se deve reverter a criação da API (`true`) ou não (`false`) quando um aviso é encontrado. O valor padrão é `false`.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FailOnWarnings](#) propriedade de um `AWS::ApiGateway::RestApi` recurso.

GatewayResponses

Configura as respostas do Gateway para uma API. As respostas do Gateway são respostas retornadas pelo API Gateway, diretamente ou por meio do uso de autorizadores do Lambda. Para obter mais informações, consulte a documentação da [OpenAPI extensão Api Gateway para respostas do Gateway](#).

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

MergeDefinitions

AWS SAM gera uma OpenAPI especificação da sua fonte de eventos de API. Especifique `true` para AWS SAM mesclar isso na OpenAPI especificação embutida definida em seu `AWS::Serverless::Api` recurso. Especifique `false` para não mesclar.

O MergeDefinitions requer que DefinitionBody a propriedade AWS::Serverless::Api seja definida. O MergeDefinitions não é compatível com a propriedade DefinitionUri para AWS::Serverless::Api.

Valor padrão: false

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

MethodSettings

Define todas as configurações para o estágio da API, incluindo Logging, Métricas, CacheTTL, controle de utilização.

Tipo: lista de [MethodSetting](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MethodSettings](#) propriedade de um AWS::ApiGateway::Stage recurso.

MinimumCompressionSize

Permita a compressão dos corpos de resposta com base no Accept-Encoding cabeçalho do cliente. A compressão é acionada quando o tamanho do corpo da resposta é maior ou igual ao limite configurado. O limite máximo de tamanho do corpo é de 10 MB (10.485.760 bytes). - Os seguintes tipos de compactação são suportados: gzip, deflate e identity.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MinimumCompressionSize](#) propriedade de um AWS::ApiGateway::RestApi recurso.

Mode

Essa propriedade se aplica somente quando você usa a OpenAPI para definir sua API REST. O Mode determina como o API Gateway trata atualizações de recursos. Para obter mais informações, consulte a propriedade [Mode](#) do tipo de recurso [AWS::ApiGateway::RestApi](#).

Valores válidos: `overwrite` ou `merge`

Tipo: `string`

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Mode](#) propriedade de um `AWS::ApiGateway::RestApi` recurso.

Models

Os esquemas a serem usados pelos seus métodos de API. Esses esquemas podem ser descritos usando JSON ou YAML. Consulte a seção Exemplos na parte inferior desta página para ver exemplos de modelos.

Tipo: `mapa`

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Name

Um nome para o RestApi recurso API Gateway

Tipo: `string`

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::ApiGateway::RestApi` recurso.

OpenApiVersion

Versão do OpenApi para usar. Isso pode ser `2.0` para a especificação Swagger ou para uma das versões OpenApi 3.0, como `3.0.1`. Para obter mais informações sobre a OpenAPI, consulte a [Especificação da OpenAPI](#).

Note

AWS SAM cria um estágio chamado `Stage` por padrão. Definir essa propriedade com qualquer valor válido impedirá a criação do estágio `Stage`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

PropagateTags

Indique se deseja ou não passar as tags da propriedade Tags para os recursos [AWS::Serverless::Api](#) gerados. Especifique True para propagar as tags nos recursos gerados.

Tipo: booleano

Obrigatório: não

Padrão: False

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Policy

Um documento de política que contenha as permissões para a API. Para definir o ARN da política, use a função intrínseca !Join com "" como delimitador e valores de "execute-api:/" e "*".

Type: JSON

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a propriedade [Policy](#) de um [AWS::ApiGateway::RestApi](#) recurso.

SecurityPolicy

A versão TLS mais o pacote de criptografia para a API REST. Ao usar políticas de segurança aprimoradas (aquelas prefixadas com `SecurityPolicy`), você também deve definir o.

EndpointAccessMode

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SecurityPolicy](#) propriedade de um [AWS::ApiGateway::RestApi](#) recurso.

StageName

O nome do estágio, que o API Gateway usa como o primeiro segmento de caminho no URI (Uniform Resource Identifier) invocado.

Para referenciar o recurso do estágio, use `<api-logical-id>.Stage`. Para obter mais informações sobre como referenciar recursos gerados quando um recurso `AWS::Serverless::API` é especificado, consulte [CloudFormation recursos gerados quando AWS::Serverless::Api é especificado](#). Para obter informações gerais sobre CloudFormation os recursos gerados, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é semelhante à `StageName` propriedade de um `AWS::ApiGateway::Stage` recurso. É obrigatório no SAM, mas não no API Gateway

Outras observações: A API implícita tem o nome de estágio de “Prod”.

Tags

Um mapa (string para string) que especifica as tags a serem adicionadas a esse estágio do API Gateway. Para obter detalhes sobre chaves e valores válidos para tags, consulte [Etiqueta de recurso](#) no AWS CloudFormation Guia do usuário.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à `Tags` propriedade de um `AWS::ApiGateway::Stage` recurso. A propriedade Tags no SAM consiste em Key:Value pares; CloudFormation nela consiste em uma lista de objetos Tag.

TracingEnabled

Indica se o rastreamento ativo com X-Ray está habilitado para o estágio. Para obter mais informações sobre isso X-Ray, consulte [Rastreamento de solicitações de usuários para APIs REST usando o Guia X-Ray](#) do desenvolvedor do API Gateway.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TracingEnabled](#) propriedade de um `AWS::ApiGateway::Stage` recurso.

Variables

Um mapa (string para string) que define as variáveis de estágio, onde o nome da variável é a chave e o valor da variável é o valor. Os nomes de variáveis são limitadas a caracteres alfanuméricos. Os valores devem corresponder a expressão regular a seguir: `[A-Za-z0-9._~:/?#&=, -]+`.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Variables](#) propriedade de um `AWS::ApiGateway::Stage` recurso.

Valores de retorno

Ref.

Quando o ID lógico desse recurso é fornecido para a Ref função intrínseca, retorna o ID da API Gateway subjacente.

Para obter mais informações sobre como usar a função Ref, consulte [Ref](#) no Guia do usuário do AWS CloudFormation .

Fn:: GetAtt

`Fn::GetAtt` retorna um valor para um atributo especificado deste tipo. Estes são os atributos disponíveis e os valores de retorno de amostra.

Para obter mais informações sobre o uso do `Fn::GetAtt`, consulte [Fn::GetAtt](#) o AWS CloudFormation Guia do usuário.

RootResourceId

O ID de um recurso raiz `RestApi`, como `a0bc123d4e`.

Exemplos

SimpleApiExample

Um arquivo de AWS SAM modelo Hello World que contém uma função de evento Lambda com um endpoint de API. Esse é um arquivo AWS SAM de modelo completo para um aplicativo sem servidor em funcionamento.

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

Um trecho AWS SAM de modelo com uma API definida em um arquivo Swagger externo junto com as integrações do Lambda e as configurações do CORS. Essa é apenas uma parte de um arquivo de AWS SAM modelo que mostra uma [AWS::Serverless::API](#) definição.

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

Um trecho AWS SAM de modelo com uma API que usa o Amazon Cognito para autorizar solicitações contra a API. Essa é apenas uma parte de um arquivo de AWS SAM modelo que mostra uma [AWS::Serverless::API](#) definição.

YAML

```
Resources:
  ApiGatewayApi:
    Type: AWS::Serverless::Api
    Properties:
      StageName: Prod
      Cors: "'*'"
      Auth:
        DefaultAuthorizer: MyCognitoAuthorizer
        Authorizers:
          MyCognitoAuthorizer:
            UserPoolArn:
              Fn::GetAtt: [MyCognitoUserPool, Arn]
```

ApiModelsExample

Um trecho de AWS SAM modelo com uma API que inclui um esquema de modelos. Essa é apenas uma parte de um arquivo de AWS SAM modelo, mostrando uma [AWS::Serverless::API](#) definição com dois esquemas de modelo.

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
```

Exemplo de armazenamento em cache

Um arquivo de AWS SAM modelo Hello World que contém uma função Lambda com um endpoint de API. A API tem o armazenamento em cache ativado para um recurso e método. Para obter mais informações sobre armazenamento em cache, consulte [Habilitar o armazenamento em cache de APIs para melhorar a capacidade de resposta](#) no Guia do desenvolvedor do API Gateway.

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}

```

Domínio personalizado com um exemplo de API privada

Um arquivo de AWS SAM modelo Hello World que contém uma função Lambda com um endpoint de API mapeado para um domínio privado. O modelo cria uma associação de acesso a domínio entre um endpoint da VPC e o domínio privado. Para ter mais informações, consulte [Nomes de domínio personalizados para APIs privadas no 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

Configure a autorização para controlar o acesso à sua API API Gateway.

Para obter mais informações e exemplos para configurar o acesso usando, AWS SAM consulte [Controle o acesso à API com seu AWS SAM modelo](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

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

Note

A propriedade `Authorizers` inclui o `AWS_IAM`, mas não é necessária nenhuma configuração extra do `AWS_IAM`. Para ver um exemplo, consulte [AWS IAM](#).

Propriedades

AddApiKeyRequiredToCorsPreflight

Se as propriedades `ApiKeyRequired` e `Cors` estiverem definidas, a configuração `AddApiKeyRequiredToCorsPreflight` fará com que a chave de API seja adicionada à propriedade `Options`.

Tipo: booleano

Obrigatório: não

Padrão: True

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AddDefaultAuthorizerToCorsPreflight

Se as propriedades DefaultAuthorizer e Cors estiverem definidas, a configuração AddDefaultAuthorizerToCorsPreflight fará com que o autorizador padrão seja adicionado à propriedade Options na seção OpenAPI.

Tipo: booleano

Obrigatório: não

Padrão: verdadeiro

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ApiKeyRequired

Se definido como verdadeiro, uma chave de API é necessária para todos os eventos da API. Para obter mais informações sobre chaves de API, consulte [Criar e usar planos de uso com chaves de API](#) no Guia do desenvolvedor do Gateway da API.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Authorizers

O autorizador usado para controlar o acesso à sua API do API Gateway.

Para obter mais informações, consulte [Controle o acesso à API com seu AWS SAM modelo](#).

Tipo: [CognitoAuthorizer](#)| [LambdaTokenAuthorizer](#)| [LambdaRequestAuthorizer](#)| AWS_IAM

Obrigatório: não

Padrão: nenhum

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Notas adicionais: O SAM adiciona os Autorizadores à OpenApi definição de uma Api.

DefaultAuthorizer

Especifique um autorizador padrão para uma API do API Gateway, que será usada para autorizar chamadas de API por padrão.

Note

Se a API EventSource da função associada a essa API estiver configurada para usar permissões do IAM, essa propriedade deverá ser definida como `AWS_IAM`, caso contrário, ocorrerá um erro.

Tipo: string

Obrigatório: não

Padrão: nenhum

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

InvokeRole

Define as credenciais de integração para todos os recursos e métodos com esse valor.

`CALLER_CREDENTIALS` mapeia para `arn:aws:iam::<user>/`, que usa as credenciais do chamador para invocar o endpoint.

Valores válidos: `CALLER_CREDENTIALS`, `NONE`, `IAMRoleArn`

Tipo: string

Obrigatório: não

Padrão: `CALLER_CREDENTIALS`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ResourcePolicy

Configure a política de recursos para todos os métodos e caminhos em uma API.

Digite: [ResourcePolicyStatement](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Notas adicionais: Essa configuração também pode ser definida individualmente `AWS::Serverless::Function` usando [ApiFunctionAuth](#). Isso é necessário para APIs com `EndpointConfiguration: PRIVATE`.

UsagePlan

Configura um plano de uso associado a essa API. Para obter mais informações sobre planos de uso, consulte [Criar e usar planos de uso com chaves de API](#) no Guia do desenvolvedor do API Gateway.

Essa AWS SAM propriedade gera três CloudFormation recursos adicionais quando essa propriedade é definida: an [AWS::ApiGateway::UsagePlan](#), an [AWS::ApiGateway::UsagePlanKey](#), an [AWS::ApiGateway::ApiKey](#). Para obter informações sobre esse cenário, consulte [UsagePlan propriedade é especificada](#). Para obter informações gerais sobre CloudFormation os recursos gerados, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Digite: [ApiUsagePlan](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

CognitoAuth

Exemplo de autenticação do 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 Exemplo de IAM

YAML

```
Auth:
  Authorizers: AWS_IAM
```

ApiUsagePlan

Configura um plano de uso para uma API do API Gateway. Para obter mais informações sobre planos de uso, consulte [Criar e usar planos de uso com chaves de API](#) no Guia do desenvolvedor do Gateway da API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

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

Propriedades

CreateUsagePlan

Determina como esse plano de uso é configurado. Os valores válidos são PER_API, SHARED e NONE.

PER_API cria recursos [AWS::ApiGateway::UsagePlan](#), [AWS::ApiGateway::ApiKey](#) e [AWS::ApiGateway::UsagePlanKey](#) que são específicos para essa API. Esses recursos têm lógica IDs de `<api-logical-id>UsagePlan`, `<api-logical-id>ApiKey`, e `<api-logical-id>UsagePlanKey`, respectivamente.

SHARED cria [AWS::ApiGateway::UsagePlan](#), [AWS::ApiGateway::ApiKey](#), e [AWS::ApiGateway::UsagePlanKey](#) recursos que são compartilhados em qualquer API que também tenha CreateUsagePlan: SHARED o mesmo AWS SAM modelo. Esses recursos têm lógica IDs de `ServerlessUsagePlan`, `ServerlessApiKey`, e `ServerlessUsagePlanKey`, respectivamente. Se você usar essa opção, recomendamos adicionar configurações adicionais para esse plano de uso em apenas um recurso de API para evitar definições conflitantes e um estado incerto.

NONE desativa a criação ou associação de um plano de uso com essa API. Isso só é necessário se SHARED ou PER_API estiver especificado no [Seção Global do AWS SAM modelo](#).

Valores válidos: PER_API, SHARED e NONE

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Description

Uma descrição do plano de uso.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::ApiGateway::UsagePlan` recurso.

Quota

Configura o número de solicitações que os usuários podem fazer em um determinado intervalo.

Digite: [QuotaSettings](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Quota](#) propriedade de um `AWS::ApiGateway::UsagePlan` recurso.

Tags

Uma matriz de tags arbitrárias (pares de chave-valor) a ser associada ao plano de uso.

Essa propriedade usa o [Tipo de CloudFormation Tag](#).

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Tags](#) propriedade de um `AWS::ApiGateway::UsagePlan` recurso.

Throttle

Configura a taxa de solicitações geral (média de solicitações por segundo) e capacidade de intermitência.

Digite: [ThrottleSettings](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Throttle](#) propriedade de um `AWS::ApiGateway::UsagePlan` recurso.

UsagePlanName

Um nome para o plano de uso.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [UsagePlanName](#) propriedade de um `AWS::ApiGateway::UsagePlan` recurso.

Exemplos

UsagePlan

Veja a seguir um exemplo de plano de uso.

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

Defina um autorizador do grupo de usuários do Amazon Cognito.

Para ter mais informações e exemplos, consulte [Controle o acesso à API com seu AWS SAM modelo](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

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

Propriedades

AuthorizationScopes

Lista de escopos de autorização para esse autorizador.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Identity

Esta propriedade pode ser usada para especificar um IdentitySource em uma solicitação recebida para um autorizador.

Digite: CognitoAuthorizationIdentity

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

UserPoolArn

O (s) ARN (s) do grupo de usuários do Amazon Cognito a ser usado para autorização. Especifique um único ARN como uma string ou vários ARNs como uma lista para usar vários grupos de usuários.

Tipo: String | List

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

CognitoAuth

Exemplo de Autenticação Cognito

YAML

```
Auth:
  Authorizers:
    MyCognitoAuth:
      AuthorizationScopes:
        - scope1
        - scope2
      UserPoolArn:
        Fn::GetAtt:
          - MyCognitoUserPool
          - Arn
      Identity:
        Header: MyAuthorizationHeader
        ValidationExpression: myauthvalidationexpression
```

CognitoAuthorizationIdentity

Essa propriedade pode ser usada para especificar um IdentitySource em uma solicitação recebida para um autorizador. Para obter mais informações, IdentitySource consulte a [OpenApi extensão ApiGateway Authorizer](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

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

ValidationExpression: *String*

Propriedades

Header

Especifique o nome do cabeçalho para Autorização na OpenApi definição.

Tipo: string

Obrigatório: não

Padrão: Autorização

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ReauthorizeEvery

O período time-to-live (TTL), em segundos, que especifica por quanto tempo o API Gateway armazena em cache os resultados do autorizador. Se você especificar um valor maior que 0, o API Gateway armazenará em cache as respostas do autorizador. Por padrão, o API Gateway define essa propriedade como 300. O valor máximo é 3600, ou uma hora.

Tipo: inteiro

Obrigatório: não

Padrão: 300

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ValidationExpression

Especifique uma expressão de validação para validar a identidade recebida

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

CognitoAuthIdentity

YAML

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

LambdaRequestAuthorizer

Configure um Lambda Authorizer para controlar o acesso à sua API com uma função de evento Lambda.

Para ter mais informações e exemplos, consulte [Controle o acesso à API com seu AWS SAM modelo](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

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

Propriedades

DisableFunctionDefaultPermissions

Especifique true para evitar a criação automática AWS SAM de um `AWS::Lambda::Permissions` recurso para provisionar permissões entre seu `AWS::Serverless::Api` recurso e a função Lambda do autorizador.

Valor padrão: false

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

FunctionArn

Especifique a função ARN da função do Lambda que fornece autorização para a API.

Note

AWS SAM criará automaticamente um `AWS::Lambda::Permissions` recurso quando `FunctionArn` for especificado para `AWS::Serverless::Api`. O recurso `AWS::Lambda::Permissions` fornece permissões entre sua API e a função do Lambda do autorizador.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

FunctionInvokeRole

Adiciona credenciais do autorizador à OpenApi definição do autorizador Lambda.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

FunctionPayloadType

Esta propriedade pode ser usada para definir o tipo de autorizador do Lambda para uma API.

Valores válidos: TOKEN ou REQUEST

Tipo: string

Obrigatório: não

Padrão: TOKEN

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Identity

Esta propriedade pode ser usada para especificar um `IdentitySource` em uma solicitação recebida para um autorizador. Essa propriedade só é necessária se a propriedade do `FunctionPayloadType` estiver definida como `REQUEST`.

Digite: [LambdaRequestAuthorizationIdentity](#)

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

LambdaRequestAuth

YAML

```
Authorizers:
  MyLambdaRequestAuth:
    FunctionPayloadType: REQUEST
    FunctionArn:
      Fn::GetAtt:
        - MyAuthFunction
        - Arn
    FunctionInvokeRole:
      Fn::GetAtt:
        - LambdaAuthInvokeRole
        - Arn
    Identity:
      Headers:
        - Authorization1
```

LambdaRequestAuthorizationIdentity

Essa propriedade pode ser usada para especificar um `IdentitySource` em uma solicitação recebida para um autorizador. Para obter mais informações, `IdentitySource` consulte a [OpenApi extensão ApiGateway Authorizer](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Context: List
Headers: List
QueryString: List
ReauthorizeEvery: Integer
StageVariables: List
```

Propriedades

Context

Converte as cadeias de caracteres de contexto fornecidas nas expressões de mapeamento do formato `context.contextString`.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Headers

Converte os cabeçalhos em uma sequência separada por vírgula de expressões de mapeamento de formato `method.request.header.name`.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

QueryString

Converte as cadeias de caracteres de consulta fornecidas em uma sequência separada por vírgula de expressões de mapeamento de formato `method.request.querystring.queryString`.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ReauthorizeEvery

O período time-to-live (TTL), em segundos, que especifica por quanto tempo o API Gateway armazena em cache os resultados do autorizador. Se você especificar um valor maior que 0, o API Gateway armazenará em cache as respostas do autorizador. Por padrão, o API Gateway define essa propriedade como 300. O valor máximo é 3600, ou uma hora.

Tipo: inteiro

Obrigatório: não

Padrão: 300

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

StageVariables

Converte as variáveis de estágio fornecidas em uma string separada por vírgula de expressões de mapeamento de formato `stageVariables.stageVariable`.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

LambdaRequestIdIdentity

YAML

```
Identity:
  QueryStrings:
    - auth
  Headers:
    - Authorization
  StageVariables:
    - VARIABLE
  Context:
    - authcontext
  ReauthorizeEvery: 100
```

LambdaTokenAuthorizer

Configure um Lambda Authorizer para controlar o acesso à sua API com uma função de evento Lambda.

Para ter mais informações e exemplos, consulte [Controle o acesso à API com seu AWS SAM modelo](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
DisableFunctionDefaultPermissions: Boolean
FunctionArn: String
FunctionInvokeRole: String
FunctionPayloadType: String
Identity: LambdaTokenAuthorizationIdentity
```

Propriedades

DisableFunctionDefaultPermissions

Especifique `true` para evitar a criação automática AWS SAM de um `AWS::Lambda::Permissions` recurso para provisionar permissões entre seu `AWS::Serverless::Api` recurso e a função Lambda do autorizador.

Valor padrão: `false`

Tipo: `booleano`

Obrigatório: `não`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

FunctionArn

Especifique a função ARN da função do Lambda que fornece autorização para a API.

Note

AWS SAM criará automaticamente um `AWS::Lambda::Permissions` recurso quando `FunctionArn` for especificado para `AWS::Serverless::Api`. O recurso `AWS::Lambda::Permissions` fornece permissões entre sua API e a função do Lambda do autorizador.

Tipo: `string`

Obrigatório: `Sim`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

FunctionInvokeRole

Adiciona credenciais do autorizador à OpenApi definição do autorizador Lambda.

Tipo: `string`

Obrigatório: `não`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

FunctionPayloadType

Esta propriedade pode ser usada para definir o tipo de autorizador do Lambda para uma Api.

Valores válidos: TOKEN ou REQUEST

Tipo: string

Obrigatório: não

Padrão: TOKEN

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Identity

Esta propriedade pode ser usada para especificar um IdentitySource em uma solicitação recebida para um autorizador. Essa propriedade só é necessária se a propriedade do FunctionPayloadType estiver definida como REQUEST.

Digite: [LambdaTokenAuthorizationIdentity](#)

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

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

Essa propriedade pode ser usada para especificar um IdentitySource em uma solicitação recebida para um autorizador. Para obter mais informações, IdentitySource consulte a [OpenApi extensão ApiGateway Authorizer](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

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

Propriedades**Header**

Especifique o nome do cabeçalho para Autorização na OpenApi definição.

Tipo: string

Obrigatório: não

Padrão: Autorização

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ReauthorizeEvery

O período time-to-live (TTL), em segundos, que especifica por quanto tempo o API Gateway armazena em cache os resultados do autorizador. Se você especificar um valor maior que 0, o API Gateway armazenará em cache as respostas do autorizador. Por padrão, o API Gateway define essa propriedade como 300. O valor máximo é 3600, ou uma hora.

Tipo: inteiro

Obrigatório: não

Padrão: 300

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ValidationExpression

Especifique uma expressão de validação para validar a identidade recebida.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

LambdaTokenIdentity

YAML

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

ResourcePolicyStatement

Configura uma política de recursos para todos os métodos e caminhos de uma API. Para obter mais informações sobre políticas de recursos, consulte Como [controlar o acesso a uma API com as políticas de recursos do API Gateway](#) no Guia do desenvolvedor do API Gateway.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
AwsAccountBlacklist: List
AwsAccountWhitelist: List
CustomStatements: List
IntrinsicVpcBlacklist: List
IntrinsicVpcWhitelist: List
IntrinsicVpceBlacklist: List
IntrinsicVpceWhitelist: List
IpRangeBlacklist: List
IpRangeWhitelist: List
SourceVpcBlacklist: List
SourceVpcWhitelist: List
```

Propriedades

AwsAccountBlacklist

As AWS contas a serem bloqueadas.

Tipo: lista de strings

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AwsAccountWhitelist

As AWS contas a serem permitidas. Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista de strings

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

CustomStatements

Uma lista de declarações de política de recursos personalizadas a serem aplicadas a essa API. Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntrinsicVpcBlacklist

A lista de nuvens privadas virtuais (VPCs) a serem bloqueadas, em que cada VPC é especificada como uma referência, como uma [referência dinâmica](#) ou a função Ref [intrínseca](#). Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntrinsicVpcWhitelist

A lista de VPCs permissões, em que cada VPC é especificada como uma referência, como uma [referência dinâmica](#) ou a função Ref [intrínseca](#).

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntrinsicVpceBlacklist

A lista de endpoints da VPC a serem bloqueados, em que cada endpoint da VPC é especificado como uma referência, como uma [referência dinâmica](#) ou a Ref [função intrínseca](#).

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntrinsicVpceWhitelist

A lista de endpoints da VPC a serem permitidos, em que cada endpoint da VPC é especificado como uma referência, como uma [referência dinâmica](#) ou a Ref [função intrínseca](#). Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IpRangeBlacklist

Os endereços IP ou intervalos de endereços a serem bloqueados. Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IpRangeWhitelist

Os endereços IP ou intervalos de endereços a serem permitidos.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SourceVpcBlacklist

A VPC de origem ou os endpoints da VPC a serem bloqueados. Os nomes da VPC de origem devem começar com "vpc-" e os nomes dos endpoints da VPC de origem devem começar com "vpce-". Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SourceVpcWhitelist

A VPC de origem ou os endpoints da VPC a serem permitidos. Os nomes da VPC de origem devem começar com "vpc-" e os nomes dos endpoints da VPC de origem devem começar com "vpce-".

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Exemplo de política de recursos

O exemplo a seguir bloqueia dois endereços IP e uma VPC de origem e permite uma AWS conta.

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

Um documento da OpenAPI que define a API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```

Bucket: String
Key: String
Version: String

```

Propriedades

Bucket

O nome do bucket do Amazon S3 no qual o arquivo do OpenAPI está armazenado.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Bucket](#) propriedade do tipo de AWS::ApiGateway::RestApi S3Location dados.

Key

A chave do Amazon S3 do arquivo OpenAPI.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Key](#) propriedade do tipo de AWS::ApiGateway::RestApi S3Location dados.

Version

Para objetos com controle de versão, a versão do arquivo OpenAPI.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Version](#) propriedade do tipo de AWS::ApiGateway::RestApi S3Location dados.

Exemplos

Exemplo de definição de Uri

Exemplo de definição de API

YAML

```
DefinitionUri:
  Bucket: amzn-s3-demo-bucket-name
  Key: mykey-name
  Version: 121212
```


CorsConfiguration

Gerencie o compartilhamento de recursos de origem cruzada (CORS) para seu API Gateway. APIs Especifique o domínio a ser permitido como uma string ou especifique um dicionário com configuração adicional do Cors.

Note

O CORS exige que AWS SAM você modifique sua definição de OpenAPI. Crie uma definição de OpenAPI em linha no `DefinitionBody` para ativar o CORS. Se `CorsConfiguration` for definido na definição da OpenAPI e também no nível da propriedade, AWS SAM mescla-os. O nível da propriedade tem precedência sobre a definição de OpenAPI.

Para obter mais informações sobre o CORS, consulte [Habilitar o CORS para um recurso API REST para a API Gateway](#) no Guia do desenvolvedor do API Gateway.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
AllowCredentials: Boolean  
AllowHeaders: String  
AllowMethods: String  
AllowOrigin: String  
MaxAge: String
```

Propriedades

AllowCredentials

Booleano indicando se a solicitação pode conter credenciais.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AllowHeaders

Cadeia de cabeçalhos a serem permitidos.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AllowMethods

Cadeia de caracteres contendo os métodos HTTP a serem permitidos.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AllowOrigin

Cadeia de origem a ser permitida. Pode ser uma lista separada por vírgulas em formato de string.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

MaxAge

Cadeia de caracteres contendo o número de segundos para armazenar em cache a solicitação CORS Preflight.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

CorsConfiguration

Exemplo de configuração do CORS. Essa é apenas uma parte de um arquivo de AWS SAM modelo que mostra uma [AWS::Serverless::API](#) definição com o CORS configurado e um [AWS::Serverless::Função](#). Se você usa uma integração de proxy do Lambda ou uma integração de proxy HTTP, seu backend deve retornar os cabeçalhos Access-Control-Allow-Origin, Access-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 um domínio personalizado para uma API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
AccessAssociation: DomainAccessAssociation  
BasePath: List  
CertificateArn: String  
DomainName: String  
EndpointAccessMode: String  
EndpointConfiguration: String  
MutualTlsAuthentication: MutualTlsAuthentication  
NormalizeBasePath: Boolean  
OwnershipVerificationCertificateArn: String  
Policy: Json  
Route53: Route53Configuration  
SecurityPolicy: String
```

Propriedades

AccessAssociation

A configuração necessária para gerar recursos

`AWS::ApiGateway::DomainNameAccessAssociation`.

AWS SAM gera um [AWS::ApiGateway::DomainNameAccessAssociation](#) recurso quando essa propriedade é definida. Para obter informações sobre CloudFormation os recursos gerados, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Digite: [DomainAccessAssociation](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

BasePath

Uma lista dos caminhos básicos a serem configurados com o nome de domínio do Amazon API Gateway.

Tipo: lista

Obrigatório: não

Padrão: /

CloudFormation compatibilidade: essa propriedade é semelhante à [BasePath](#) propriedade de um `AWS::ApiGateway::BasePathMapping` recurso. AWS SAM cria vários `AWS::ApiGateway::BasePathMapping` recursos, um por BasePath especificado nessa propriedade.

CertificateArn

O Amazon Resource Name (ARN) de um certificado AWS gerenciado é o endpoint desse nome de domínio. Gerenciador de certificados da AWS é a única fonte compatível.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é semelhante à [CertificateArn](#) propriedade de um `AWS::ApiGateway::DomainName` recurso. Se `EndpointConfiguration` estiver definido como `REGIONAL` (o valor padrão), `CertificateArn` mapeia para [RegionalCertificateArn](#) em `AWS::ApiGateway::DomainName`. Se `EndpointConfiguration` estiver definido como `EDGE`, `CertificateArn` mapeia para [CertificateArn](#) em `AWS::ApiGateway::DomainName`. Se `EndpointConfiguration` estiver definido como `PRIVATE`, essa propriedade será passada para o recurso [AWS::ApiGateway::DomainNameV2](#).

Observações adicionais: Para um `EDGE` endpoint, você deve criar o certificado na `us-east-1` AWS região.

DomainName

O nome de domínio personalizado para a sua API Gateway API. Letras maiúsculas não são compatíveis.

AWS SAM gera um [AWS::ApiGateway::DomainName](#) recurso quando essa propriedade é definida. Para obter informações sobre esse cenário, consulte [DomainName propriedade é especificada](#). Para obter informações sobre CloudFormation os recursos gerados, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DomainName](#) propriedade de um `AWS::ApiGateway::DomainName` recurso ou para [AWS::ApiGateway::DomainNameV2](#) quando EndpointConfiguration está definida como PRIVATE.

EndpointConfiguration

Define o tipo de endpoint do API Gateway a ser mapeado para o domínio personalizado. O valor dessa propriedade determina como a CertificateArn propriedade é mapeada CloudFormation.

Valores válidos: EDGE, REGIONAL ou PRIVATE

Tipo: string

Obrigatório: não

Padrão: REGIONAL

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

EndpointAccessMode

O modo de acesso para o endpoint de nome de domínio personalizado. Obrigatório ao usar políticas de segurança aprimoradas (aquelas prefixadas com SecurityPolicy_).

Valores válidos: STRICT ou BASIC

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EndpointAccessMode](#) propriedade de um `AWS::ApiGateway::DomainName` recurso ou para [AWS::ApiGateway::DomainNameV2](#) quando `EndpointConfiguration` está definida como `PRIVATE`.

MutualTlsAuthentication

A Transport Layer Security (TLS) mútua para um nome de domínio personalizado.

Digite: [MutualTlsAuthentication](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MutualTlsAuthentication](#) propriedade de um `AWS::ApiGateway::DomainName` recurso.

NormalizeBasePath

Indica se caracteres não alfanuméricos são permitidos nos caminhos base definidos pela propriedade `BasePath`. Quando definido como `True`, caracteres não alfanuméricos são removidos dos caminhos base.

Use a propriedade `NormalizeBasePath` para `BasePath`.

Tipo: booleano

Obrigatório: não

Padrão: verdadeiro

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

OwnershipVerificationCertificateArn

O ARN do certificado público emitido pelo ACM para validar a propriedade do domínio personalizado. Necessário somente para configurar o TLS mútuo e para especificar um ARN de CA privado ou importado do ACM para o `CertificateArn`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [OwnershipVerificationCertificateArn](#) propriedade de um `AWS::ApiGateway::DomainName` recurso.

Policy

A política do IAM para anexar ao nome de domínio do API Gateway. Aplicável somente quando `EndpointConfiguration` for definida como `PRIVATE`.

Tipo: Json

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a `Policy` propriedade de um `AWS::ApiGateway::DomainNameV2` recurso quando `EndpointConfiguration` está definida como `PRIVATE`. Para exemplos de documentos de política válidos, consulte [AWS::ApiGateway::DomainNameV2](#).

Route53

Define uma configuração do Amazon Route 53.

Tipo: [Route53Configuration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SecurityPolicy

O pacote de criptografia TLS mais o pacote de criptografia para este nome de domínio.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SecurityPolicy](#) propriedade de um `AWS::ApiGateway::DomainName` recurso ou para [AWS::ApiGateway::DomainNameV2](#) quando `EndpointConfiguration` está definida como `PRIVATE`. Para endpoints `PRIVATE`, somente `TLS_1_2` é compatível.

Exemplos

DomainName

DomainName exemplo

YAML

```
Domain:
  DomainName: www.example.com
  CertificateArn: arn-example
  EndpointConfiguration: EDGE
  Route53:
    HostedZoneId: Z1PA6795UKMFR9
  BasePath:
    - foo
    - bar
```

DomainAccessAssociation

Configura uma associação de acesso a um domínio entre uma fonte de associação de acesso e um nome de domínio personalizado privado. A única fonte de associação de acesso compatível é um ID de endpoint da VPC. Para obter mais informações, consulte [Nomes de domínio personalizados para uso privado APIs no API Gateway](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
VpcEndpointId: String
```

Propriedades

VpcEndpointId

O ID do endpoint da interface da VPC associada ao serviço da VPC do API Gateway.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AccessAssociationSource](#) propriedade de um `AWS::ApiGateway::DomainNameAccessAssociation` recurso.

Route53Configuration

Configura os conjuntos de registros Route53 para uma API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
DistributionDomainName: String
EvaluateTargetHealth: Boolean
HostedZoneId: String
HostedZoneName: String
IpV6: Boolean
Region: String
SetIdentifier: String
VpcEndpointDomainName: String
VpcEndpointHostedZoneId: String
```

Propriedades

DistributionDomainName

Configura uma distribuição personalizada do nome de domínio personalizado da API.

Tipo: string

Obrigatório: não

Padrão: use a distribuição do API Gateway.

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DNSName](#) propriedade de um `AWS::Route53::RecordSetGroup` `AliasTarget` recurso.

Notas adicionais: O nome de domínio de uma [CloudFront distribuição](#).

EvaluateTargetHealth

Quando EvaluateTargetHealth verdadeiro, um registro de alias herda a integridade do AWS recurso referenciado, como um balanceador de carga do Elastic Load Balancing ou outro registro na zona hospedada.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EvaluateTargetHealth](#) propriedade de um AWS::Route53::RecordSetGroup AliasTarget recurso.

Observações adicionais: você não pode EvaluateTargetHealth definir como verdadeiro quando o destino do alias é uma CloudFront distribuição.

HostedZoneId

O ID da zona hospedada na qual você deseja criar registros.

Especifique HostedZoneName ou HostedZoneId, mas não ambos. Se houver várias zonas hospedadas com o mesmo nome de domínio, especifique a zona hospedada usando HostedZoneId.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [HostedZoneId](#) propriedade de um AWS::Route53::RecordSetGroup RecordSet recurso.

HostedZoneName

O nome da zona hospedada na qual você deseja criar registros.

Especifique HostedZoneName ou HostedZoneId, mas não ambos. Se houver várias zonas hospedadas com o mesmo nome de domínio, especifique a zona hospedada usando HostedZoneId.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [HostedZoneName](#) propriedade de um `AWS::Route53::RecordSetGroup` `RecordSet` recurso.

IPv6

Quando essa propriedade é definida, AWS SAM cria um `AWS::Route53::RecordSet` recurso e define [Type](#) como AAAA para o fornecido `HostedZone`.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Region

Somente conjuntos de registros de recursos baseados em latência: a EC2 região amazônica onde você criou o recurso ao qual esse conjunto de registros de recursos se refere. O recurso normalmente é um AWS recurso, como uma EC2 instância ou um balanceador de carga ELB, e é referido por um endereço IP ou nome de domínio DNS, dependendo do tipo de registro.

Quando o Amazon Route 53 recebe uma consulta de DNS para um nome e tipo de domínio para o qual você criou conjuntos de registros de recursos de latência, o Route 53 seleciona o conjunto de registros de recursos de latência que tem a menor latência entre o usuário final e a região da Amazon associada. EC2 O Route 53 retorna o valor associado ao conjunto de registros de recursos selecionado.

Observe o seguinte:

- Só é possível especificar um `ResourceRecord` por conjunto de registros de recursos de latência.
- Você só pode criar um conjunto de registros de recursos de latência para cada EC2 região da Amazon.
- Você não precisa criar conjuntos de registros de recursos de latência para todas as EC2 regiões da Amazon. O Route 53 selecionará a região com a melhor latência entre as regiões para as quais você cria conjuntos de registros de recursos de latência.
- Não é possível criar conjuntos de registros de recursos que não sejam de latência e tenham os mesmos valores para os elementos `Name` e `Type` como conjuntos registros de recursos de latência.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Region](#) propriedade de um tipo de `AWS::Route53::RecordSetGroup` `RecordSet` dados.

`SetIdentifier`

Conjuntos de registros de recursos que têm uma política de roteamento diferente da simples: um identificador que diferencia entre vários conjuntos de registros de recursos que têm a mesma combinação de nome e tipo, como vários conjuntos de registros de recursos ponderados chamados `acme.example.com` que tenham um tipo de `A`. Em um grupo de conjuntos de registros de recursos que têm o mesmo nome e tipo, o valor de `SetIdentifier` deve ser exclusivo para cada conjunto de registros de recursos.

Para obter informações sobre políticas de roteamento, consulte [Escolher uma política de roteamento](#) no Guia do desenvolvedor do Amazon Route 53.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SetIdentifier](#) propriedade de um tipo de `AWS::Route53::RecordSetGroup` `RecordSet` dados.

`VpcEndpointDomainName`

O nome de DNS do endpoint da interface da VPC associada ao serviço da VPC do API Gateway. Essa propriedade é necessária apenas para domínios privados.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DNSName](#) propriedade de um `AWS::Route53::RecordSet` `AliasTarget` campo.

`VpcEndpointHostedZoneId`

O ID da zona hospedada do endpoint da interface VPC associado ao serviço da VPC do API Gateway. Essa propriedade é necessária apenas para domínios privados.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [HostedZoneId](#) propriedade de um `AWS::Route53::RecordSet` AliasTarget campo.

Exemplos

Exemplo básico

Neste exemplo, configuramos um domínio personalizado e conjuntos de registros do Route 53 para nossa 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

O tipo de endpoint de uma API REST.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
IpAddressType: String
Type: String
```

[VPCEndpointIds](#): *List*

Propriedades

IpAddressType

Os tipos de endereço IP que podem invocar uma API (RestApi).

Valores válidos: ipv4 ou dualstack

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [IpAddressType](#) propriedade do tipo de AWS::ApiGateway::RestApi EndpointConfiguration dados.

Type

O tipo de endpoint de uma API REST.

Valores válidos: EDGE ou REGIONAL ou PRIVATE

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Types](#) propriedade do tipo de AWS::ApiGateway::RestApi EndpointConfiguration dados.

VPCEndpointIds

Uma lista de pontos de extremidade VPC IDs de uma API REST com os quais criar aliases do Route53.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [VpcEndpointIds](#) propriedade do tipo de AWS::ApiGateway::RestApi EndpointConfiguration dados.

Exemplos

EndpointConfiguration

Exemplo de configuração do endpoint

YAML

```
EndpointConfiguration:
  Type: PRIVATE
  VPCEndpointIds:
    - vpce-123a123a
    - vpce-321a321a
```

AWS::Serverless::Application

Incorpora um aplicativo com tecnologia sem servidor de [AWS Serverless Application Repository](#) ou de um bucket do Amazon S3 como um aplicativo aninhado. Os aplicativos aninhados são implantados como recursos [AWS::CloudFormation::Stack](#) aninhados, que podem conter vários outros recursos, incluindo outros recursos [AWS::Serverless::Application](#).

Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Type: AWS::Serverless::Application
Properties:
  Location: String | ApplicationLocationObject
  NotificationARNs: List
  Parameters: Map
```


Tags: *Map*

TimeoutInMinutes: *Integer*

Propriedades

Location

URL do modelo, caminho do arquivo ou objeto de localização de um aplicativo aninhado.

Se um URL de modelo for fornecido, ele deverá seguir o formato especificado na [CloudFormation TemplateUrl documentação](#) e conter um modelo válido CloudFormation ou SAM. Um [ApplicationLocationObject](#) pode ser usado para especificar um aplicativo que foi publicado no [AWS Serverless Application Repository](#).

Se um caminho de arquivo local for fornecido, o modelo deverá passar pelo fluxo de trabalho que inclui o comando `sam deploy` ou `sam package` para que o aplicativo seja transformado adequadamente.

Tipo: String | [ApplicationLocationObject](#)

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é semelhante à [TemplateURL](#) propriedade de um `AWS::CloudFormation::Stack` recurso. A CloudFormation versão não precisa de um [ApplicationLocationObject](#) para recuperar um aplicativo do AWS Serverless Application Repository.

NotificationARNs

Uma lista de tópicos existentes do Amazon SNS onde as notificações sobre eventos da pilha são enviadas.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [NotificationARNs](#) propriedade de um `AWS::CloudFormation::Stack` recurso.

Parameters

Valores do parâmetro do aplicativo.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Parameters](#) propriedade de um `AWS::CloudFormation::Stack` recurso.

Tags

Um mapa (string para string) que especifica as tags a serem adicionadas a este aplicativo. As chaves e valores são limitados a caracteres alfanuméricos. As chaves podem ter de 1 a 127 caracteres Unicode e não podem ser prefixadas com `aws:`. Os valores podem ter de 1 a 255 caracteres Unicode.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Tags](#) propriedade de um `AWS::CloudFormation::Stack` recurso. A propriedade Tags no SAM consiste em pares de chave-valor; CloudFormation nela consiste em uma lista de objetos Tag. Quando a pilha for criada, o SAM adicionará automaticamente uma tag `lambda:createdBy:SAM` a esse aplicativo. Além disso, se esse aplicativo for do AWS Serverless Application Repository, o SAM também enviará automaticamente as duas tags adicionais `serverlessrepo:applicationId:ApplicationId` e `serverlessrepo:semanticVersion:SemanticVersion`.

TimeoutInMinutes

O período de tempo, em minutos, que CloudFormation espera que a pilha aninhada alcance o estado. `CREATE_COMPLETE` O padrão é nenhum tempo limite. Quando CloudFormation detecta que a pilha aninhada atingiu o `CREATE_COMPLETE` estado, ela marca o recurso da pilha aninhada como `CREATE_COMPLETE` na pilha principal e retoma a criação da pilha principal. Se o período de tempo limite expirar antes que a pilha aninhada chegue `CREATE_COMPLETE`, CloudFormation marcará a pilha aninhada como falhada e reverte tanto a pilha aninhada quanto a pilha principal.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TimeoutInMinutes](#) propriedade de um `AWS::CloudFormation::Stack` recurso.

Valores de retorno

Ref.

Quando o ID lógico desse recurso for fornecido para a função intrínseca Ref, retorna o nome do recurso do recurso `AWS::CloudFormation::Stack` subjacente.

Para obter mais informações sobre como usar a função Ref, consulte [Ref](#) no Guia do usuário do AWS CloudFormation .

Fn:: GetAtt

`Fn::GetAtt` retorna um valor para um atributo especificado deste tipo. Estes são os atributos disponíveis e os valores de retorno de amostra.

Para obter mais informações sobre o uso do `Fn::GetAtt`, consulte o [Fn::GetAtt](#) no AWS CloudFormation Guia do usuário.

`Outputs.ApplicationOutputName`

O valor da saída da pilha com nome *ApplicationOutputName*.

Exemplos

Aplicativo SAR

Aplicativo que usa um modelo do repositório de aplicativo com tecnologia sem servidor

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
```

Aplicativo normal

Aplicativo a partir de um URL do S3

YAML

```
Type: AWS::Serverless::Application
Properties:
  Location: https://s3.amazonaws.com/sam-s3-demo-bucket/template.yaml
```

ApplicationLocationObject

Um aplicativo que foi publicado no [AWS Serverless Application Repository](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
ApplicationId: String
SemanticVersion: String
```

Propriedades

ApplicationId

O nome de recurso da Amazon (ARN) do aplicativo.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SemanticVersion

A versão semântica do aplicativo.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

my-application

Exemplo de objeto de localização do aplicativo

YAML

```
Location:
  ApplicationId: 'arn:aws:serverlessrepo:us-east-1:012345678901:applications/my-
application'
  SemanticVersion: 1.0.0
```

AWS::Serverless::CapacityProvider

Cria um provedor de capacidade para funções de AWS Lambda eventos que permite a execução de instâncias gerenciadas Lambda em instâncias Amazon Elastic Compute Cloud de propriedade do cliente. Esse recurso faz parte do recurso Lambda Managed Instances, que fornece otimização de custos para cargas de trabalho Lambda em grande escala utilizando modelos de preços do Amazon EC2.

O provedor de capacidade gerencia o ciclo de vida das instâncias do Amazon EC2 e fornece a infraestrutura necessária para que as funções Lambda sejam executadas em recursos computacionais de propriedade do cliente, mantendo o modelo de programação sem servidor.

Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Type: AWS::Serverless::CapacityProvider
Properties:
```

```
CapacityProviderName: String  
VpcConfig: VpcConfig  
OperatorRole: String  
Tags: Map  
PropagateTags: Boolean  
InstanceRequirements: InstanceRequirements  
ScalingConfig: ScalingConfig  
LoggingConfig: CapacityProviderLoggingConfig  
KmsKeyArn: String  
ManagedResourceTags: ManagedResourceTags
```

Propriedades

CapacityProviderName

O nome do provedor de capacidade. Esse nome deve ser exclusivo em sua AWS conta e região.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CapacityProviderName](#) propriedade de um `AWS::Lambda::CapacityProvider` recurso.

VpcConfig

A configuração da VPC para o provedor de capacidade. Especifica as sub-redes VPC e os grupos de segurança em que as instâncias do Amazon EC2 serão lançadas.

Digite: [VpcConfig](#)

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [VpcConfig](#) propriedade de um `AWS::Lambda::CapacityProvider` recurso.

OperatorRole

O ARN da função de operador do Lambda com permissões para criar e gerenciar instâncias do Amazon EC2 e recursos relacionados na conta do cliente. Se não for fornecido, gera AWS SAM automaticamente uma função com as permissões necessárias.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CapacityProviderOperatorRoleArn](#) propriedade [PermissionsConfig](#) de um `AWS::Lambda::CapacityProvider` recurso.

Tags

Um mapa de pares de valores-chave a serem aplicados ao provedor de capacidade e aos recursos associados.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Tags](#) propriedade de um `AWS::Lambda::CapacityProvider` recurso. A `Tags` propriedade in AWS SAM consiste em pares de valores-chave (enquanto CloudFormation nessa propriedade consiste em uma lista de objetos `Tag`). Além disso, adiciona AWS SAM automaticamente uma `lambda:createdBy: SAM` tag a essa função Lambda e às funções padrão geradas para essa função.

PropagateTags

Indica se você deve ou não passar tags da propriedade `Tags` para os recursos `AWS::Serverless::CapacityProvider` gerados. Defina isso `True` para propagar tags nos recursos gerados.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

InstanceRequirements

Especificações dos tipos de instâncias de computação que o provedor de capacidade pode usar. Isso inclui requisitos de arquitetura `allowed` e/ou tipos de `excluded` instância.

Digite: [InstanceRequirements](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [InstanceRequirements](#) propriedade de um `AWS::Lambda::CapacityProvider` recurso.

ScalingConfig

A configuração de escalabilidade para o provedor de capacidade. Define como o provedor de capacidade escala as instâncias do Amazon EC2 com base na demanda.

Digite: [ScalingConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CapacityProviderScalingConfig](#) propriedade de um `AWS::Lambda::CapacityProvider` recurso.

LoggingConfig

Use `LoggingConfig` para configurar o registro do sistema para o provedor de capacidade. Os registros do sistema capturam atividades de escalabilidade e eventos operacionais e os enviam para um grupo de CloudWatch registros da Amazon em um nível de log configurável.

Digite: [CapacityProviderLoggingConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [LoggingConfig](#) propriedade [TelemetryConfig](#) de um `AWS::Lambda::CapacityProvider` recurso.

KmsKeyArn

O ARN da AWS KMS chave usada para criptografar dados em repouso e em trânsito para o provedor de capacidade.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [KmsKeyArn](#) propriedade de um `AWS::Lambda::CapacityProvider` recurso.

ManagedResourceTags

Aplica tags explícitas às instâncias do Amazon Elastic Compute Cloud e a outros recursos gerenciados que o provedor de capacidade provisiona em seu nome.

Digite: [ManagedResourceTags](#)

Obrigatório: não

CloudFormation compatibilidade: AWS SAM usa essa propriedade para construir a [PropagateTags](#) propriedade de um `AWS::Lambda::CapacityProvider` recurso.

Valores de retorno

Ref.

Quando a ID lógica desse recurso é fornecida à função Ref intrínseca, ela retorna o nome do provedor de capacidade.

Para obter mais informações sobre como usar a função Ref, consulte [Ref](#) no Guia do usuário do AWS CloudFormation .

Fn:: GetAtt

Fn::GetAtt retorna um valor para um atributo especificado deste tipo. Estes são os atributos disponíveis e os valores de retorno de amostra.

Para obter mais informações sobre o uso do Fn::GetAtt, consulte [Fn::GetAtt](#) o AWS CloudFormation Guia do usuário.

Arn

O ARN do provedor de capacidade.

Exemplos

Provedor de capacidade básica

O exemplo a seguir cria um provedor de capacidade básico com configuração de 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
```

Provedor de capacidade avançada com escalabilidade

O exemplo a seguir cria um provedor de capacidade com requisitos de instância personalizados e configuração de escalabilidade.

```
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:
      MaxVCpuCount: 10
      AverageCPUUtilization: 70.0
    KmsKeyArn: arn:aws:kms:us-east-1:123456789012:key/12345678-1234-1234-1234-123456789012
    LoggingConfig:
      LogGroup: /aws/lambda/capacity-provider/advanced-capacity-provider
      SystemLogLevel: INFO
    ManagedResourceTags:
      Tags:
        CostCenter: cc-1234
        Environment: Production
    Tags:
      Environment: Production
      CostCenter: Engineering
```

VpcConfig

Define as configurações de VPC para um provedor de capacidade, incluindo as sub-redes e os grupos de segurança em EC2 que as instâncias serão lançadas.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
SubnetIds: List  
SecurityGroupIds: List
```

Propriedades

SubnetIds

Uma lista de sub-redes em IDs que as EC2 instâncias serão lançadas. Pelo menos uma sub-rede deve ser especificada.

Tipo: lista

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SubnetIds](#) propriedade [VpcConfig](#) de um `AWS::Lambda::CapacityProvider` recurso.

SecurityGroupIds

Uma lista de grupos de segurança IDs a serem associados às EC2 instâncias. Se não for especificado, o grupo de segurança padrão da VPC será usado.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SecurityGroupIds](#) propriedade de [VpcConfig](#) propriedade de um `AWS::Lambda::CapacityProvider` recurso.

Exemplos

Configuração de VPC

O exemplo a seguir mostra uma configuração de VPC com várias sub-redes e grupos de segurança.

```
VpcConfig:
  SubnetIds:
    - subnet-12345678
    - subnet-87654321
  SecurityGroupIds:
    - sg-12345678
    - sg-87654321
```

InstanceRequirements

Especifica os requisitos para EC2 instâncias que serão lançadas pelo provedor de capacidade, incluindo arquiteturas e restrições de tipo de instância.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Architectures: List
AllowedTypes: List
ExcludedTypes: List
```

Note

Você pode optar por especificar um `AllowedTypes` ou `ExcludedTypes` ao definir os requisitos de instância para seu provedor de capacidade, mas não ambos.

Propriedades

Architectures

As arquiteturas do conjunto de instruções para as instâncias do provedor de capacidade.

Valores válidos: x86_64 ou arm64

Tipo: lista

Obrigatório: não

Padrão: x86_64

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Architectures](#) propriedade [InstanceRequirements](#) de um `AWS::Lambda::CapacityProvider` recurso.

AllowedTypes

Uma lista dos tipos de EC2 instância permitidos para a instância do provedor de capacidade.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AllowedInstanceTypes](#) propriedade [InstanceRequirements](#) de um `AWS::Lambda::CapacityProvider` recurso.

ExcludedTypes

Uma lista de tipos de EC2 instância a serem excluídos do provedor de capacidade.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ExcludedInstanceTypes](#) propriedade [InstanceRequirements](#) de um `AWS::Lambda::CapacityProvider` recurso.

Exemplos

Configuração de requisitos de instância

O exemplo a seguir mostra os requisitos de instância com restrições específicas de arquitetura e tipo de instância.

```
InstanceRequirements:
```

Architectures:

- x86_64

ExcludedTypes:

- t2.micro

ScalingConfig

Especifica como o provedor de capacidade escala as instâncias do Amazon Elastic Compute Cloud (Amazon EC2) com base na demanda. Você define o número máximo de vCPUs que o provedor de capacidade pode provisionar e a meta média de utilização da CPU que aciona o escalonamento.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
MaxVCpuCount: Integer  
AverageCPUUtilization: Double
```

Propriedades

MaxVCpuCount

O número máximo de vCPUs que o provedor de capacidade pode provisionar em todas as instâncias de computação.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaxVCpuCount](#) propriedade [CapacityProviderScalingConfig](#) de um `AWS::Lambda::CapacityProvider` recurso.

AverageCPUUtilization

A meta média de porcentagem de utilização da CPU (0-100) para decisões de escalabilidade. Quando a utilização média da CPU exceder esse limite, o provedor de capacidade ampliará as instâncias do Amazon EC2. Quando especificado, AWS SAM constrói [CapacityProviderScalingConfig](#) um `AWS::Lambda::CapacityProvider` recurso

com o [ScalingMode](#) definido como 'Manual' e [ScalingPolicies](#) definido como.

```
[{PredefinedMetricType: 'LambdaCapacityProviderAverageCPUUtilization',  
TargetValue: <this value>}]
```

Tipo: duplo

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Scaling configuration (Configuração de escalabilidade)

O exemplo a seguir mostra uma configuração de escalabilidade com contagem máxima de vCPUs e média de utilização da CPU.

```
ScalingConfig:  
  MaxVCpuCount: 10  
  AverageCPUUtilization: 70.0
```

ManagedResourceTags

Aplica tags explícitas às instâncias do Amazon Elastic Compute Cloud e a outros recursos que o provedor de capacidade provisiona e gerencia em seu nome.

Note

A `ManagedResourceTags` propriedade controla a marcação das instâncias do Amazon EC2 e de outros recursos que o Lambda gerencia para o provedor de capacidade. Essa propriedade difere da [PropagateTags](#) propriedade, que especifica se AWS SAM propaga a [propriedade](#) Tags para os CloudFormation recursos AWS SAM gerados a partir do seu modelo.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

`Tags:` *Map*

Propriedades

As propriedades a seguir são suportadas pelo `ManagedResourceTags`.

Tags

Um mapa de pares de valores-chave para aplicar às instâncias do Amazon EC2 e a outros recursos gerenciados pelo provedor de capacidade.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: AWS SAM usa essa propriedade para construir a [PropagateTags](#) propriedade de um `AWS::Lambda::CapacityProvider` recurso. AWS SAM define `Mode Explicit` e converte os pares de valores-chave em uma lista de objetos Tag em `ExplicitTags`

Exemplos

Aplique tags explícitas aos recursos gerenciados

O exemplo a seguir aplica um conjunto explícito de tags às instâncias do Amazon EC2 gerenciadas pelo provedor de capacidade.

```
ManagedResourceTags:
  Tags:
    CostCenter: cc-1234
    Environment: Production
```

AWS::Serverless::Connector

Configura as permissões entre dois recursos. Para obter uma introdução aos conectores, consulte [Gerenciando permissões de recursos com AWS SAM conectores](#).

Para obter mais informações sobre AWS CloudFormation os recursos gerados, consulte [CloudFormation recursos gerados quando você especifica AWS::Serverless::Connector](#).

Para fornecer feedback sobre conectores, [envie um novo problema](#) no repositório `AWS GitHub serverless-application-model`.

Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use qualquer uma das sintaxes a seguir.

Note

Recomendamos usar a sintaxe de conectores incorporados para a maioria dos casos de uso. Estar incorporado ao recurso de origem facilita a leitura e a manutenção ao longo do tempo. Quando precisar referenciar um recurso de origem que não esteja no mesmo AWS SAM modelo, como um recurso em uma pilha aninhada ou um recurso compartilhado, use a `AWS::Serverless::Connector` sintaxe.

Conectores embutidos

```
<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](#)

Propriedades

Destination

O recurso de destino.

Tipo: [ResourceReference](#) | Lista de [ResourceReference](#)

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Permissions

O tipo de permissão que o recurso de origem tem permissão para executar no recurso de destino.

Read inclui ações AWS Identity and Access Management (IAM) que permitem a leitura de dados do recurso.

Write inclui ações do IAM que permitem iniciar e escrever dados em um recurso.

Valores válidos: Read ou Write

Tipo: lista

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Source

O recurso de origem. Obrigatório ao usar a sintaxe `AWS::Serverless::Connector`.

Digite: [ResourceReference](#)

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SourceReference

O recurso de origem.

 Note

Use com a sintaxe de conectores incorporados ao definir propriedades adicionais para o recurso de origem.

Digite: [SourceReference](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Conectores embutidos

O exemplo a seguir usa conectores incorporados para definir uma conexão de Write dados entre uma função de AWS Lambda evento e uma tabela do 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
    ...
```

O exemplo a seguir usa conectores incorporados para definir permissões Read e Write:

```
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
  ...

```

O exemplo a seguir usa conectores incorporados para definir um recurso de origem com uma propriedade diferente de Id:

```

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: :Conector

O exemplo a seguir usa o [AWS::Serverless: :Conector](#) recurso para que uma AWS Lambda função seja lida e gravada em uma tabela do Amazon DynamoDB:

```

MyConnector:

```

```
Type: AWS::Serverless::Connector
Properties:
  Source:
    Id: MyFunction
  Destination:
    Id: MyTable
  Permissions:
    - Read
    - Write
```

O exemplo a seguir usa o recurso [AWS::Serverless::Connector](#) para que uma função do Lambda seja escrita em um tópico do Amazon SNS, com os dois recursos no mesmo modelo:

```
MyConnector:
  Type: AWS::Serverless::Connector
  Properties:
    Source:
      Id: MyLambda
    Destination:
      Id: MySNSTopic
    Permissions:
      - Write
```

O exemplo a seguir usa o [AWS::Serverless::Connector](#) recurso para que um tópico do Amazon SNS grave em uma função do Lambda, que então grava em uma tabela do Amazon DynamoDB, com todos os recursos no mesmo modelo:

```
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
```

Veja a seguir o AWS CloudFormation modelo transformado do exemplo acima:

```
"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

Uma referência a um recurso que o tipo de recurso de [AWS::Serverless::Conector](#) usa.

Note

Para recursos no mesmo modelo, forneça o Id. Para recursos que não estejam no mesmo modelo, use uma combinação de outras propriedades. Para obter mais informações, consulte [AWS SAM referência do conector](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Arn: String
Id: String
Name: String
Qualifier: String
QueueUrl: String
ResourceId: String
RoleName: String
Type: String
```


Propriedades

Arn

O ARN de um recurso.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Id

O [ID lógico](#) de um recurso no mesmo modelo.

Note

Quando Id for especificado, se o conector gerar políticas AWS Identity and Access Management (IAM), a função do IAM associada a essas políticas será inferida do recursoId. Quando Id não for especificado, forneça o recurso RoleName para conectores anexarem as políticas do IAM geradas a um perfil do IAM.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Name

O nome de um recurso do .

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Qualifier

Um qualificador para um recurso que restrinja seu escopo. Qualifier substitui o valor * no final de um ARN de restrição de recursos. Para ver um exemplo, consulte [API Gateway invocando uma função de evento Lambda](#).

Note

A definição do qualificador varia de acordo com o tipo de recurso. Para obter uma lista de tipos de recursos de origem e destino compatíveis, consulte [AWS SAM referência do conector](#).

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

QueueUrl

O URL da fila do Amazon SQS. Essa propriedade só se aplica aos recursos do Amazon SQS.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ResourceId

O ID de um recurso. Por exemplo, o ID da API API Gateway.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

RoleName

O nome do perfil associado a um recurso.

 Note

Quando Id for especificado, se o conector gerar políticas do IAM, o perfil do IAM associado a essas políticas será inferida do recurso Id. Quando Id não for especificado, forneça o recurso RoleName para conectores anexarem as políticas do IAM geradas a um perfil do IAM.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Type

O CloudFormation tipo de um recurso. Para obter mais informações, consulte a [AWS referência de tipos de recursos e propriedades](#).

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

API Gateway invocando uma função de evento Lambda

O exemplo a seguir usa o [AWS::Serverless::Conector](#) recurso para permitir que o Amazon API Gateway invoque uma função de AWS Lambda evento.

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

Uma referência a um recurso de origem usado pelo tipo de recurso de [AWS::Serverless::Conector](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Qualifier: String
```

Propriedades

Qualifier

Um qualificador para um recurso que restrinja seu escopo. `Qualifier` substitui o valor `*` no final de um ARN de restrição de recursos.

Note

A definição do qualificador varia de acordo com o tipo de recurso. Para obter uma lista de tipos de recursos de origem e destino compatíveis, consulte [AWS SAM referência do conector](#).

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

O exemplo a seguir usa conectores incorporados para definir um recurso de origem com uma propriedade diferente de **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:::Função

O `AWS::Serverless::Function` recurso cria uma função de evento Lambda, uma função de execução AWS Identity and Access Management (IAM) e mapeamentos de origem de eventos que acionam a função.

Event Function é o nome da documentação do tipo de função Lambda original, anteriormente conhecido como funções Lambda. O tipo de `AWS::Serverless::Function` recurso, o `AWS::Lambda::Function` recurso que ele gera e suas propriedades ainda usam a função, então você não precisa alterar seus modelos.

O [AWS::Serverless:::Função](#) recurso também oferece suporte ao atributo Metadata resource, para que você possa AWS SAM instruir a criar tempos de execução personalizados que seu aplicativo exija. Para obter mais informações sobre a criação de tempos de execução personalizados, consulte [Criando funções de eventos Lambda com tempos de execução personalizados em AWS SAM](#).

Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

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

Propriedades

Architectures

A arquitetura do conjunto de instruções para a função.

Para obter mais informações sobre esta propriedade, consulte [Arquiteturas de conjuntos de instruções do Lambda](#) no AWS Lambda Guia do desenvolvedor.

Valores válidos: Um de x86_64 ou arm64.

Tipo: lista

Obrigatório: não

Padrão: x86_64

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Architectures](#) propriedade de um `AWS::Lambda::Function` recurso.

AssumeRolePolicyDocument

Adiciona um AssumeRolePolicyDocument para o padrão criado Role para essa função. Se essa propriedade não for especificada, AWS SAM adicionará uma função de suposição padrão para essa função.

Type: JSON

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [AssumeRolePolicyDocument](#) propriedade de um `AWS::IAM::Role` recurso. AWS SAM adiciona essa propriedade à função IAM gerada para essa função. Se o nome de recurso da Amazon (ARN) de um perfil é fornecido para essa função, essa propriedade não faz nada.

AutoPublishAlias

O nome do alias do Lambda. Para obter mais informações sobre aliases do Lambda, consulte [Aliases de funções do Lambda](#) no AWS Lambda Guia do desenvolvedor. Para obter exemplos que usam essa propriedade, consulte [Implantando aplicativos sem servidor gradualmente com AWS SAM](#).

AWS SAM gera [AWS::Lambda::Version](#) e fornece [AWS::Lambda::Alias](#) recursos quando essa propriedade é definida. Para obter informações sobre esse cenário, consulte [AutoPublishAlias propriedade é especificada](#). Para obter informações gerais sobre CloudFormation os recursos gerados, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Publicação de versões e funções intrínsecas

AWS SAM determina se uma nova versão deve ser publicada comparando os valores das propriedades no modelo entre as implantações. Para valores fornecidos por meio de funções intrínsecas, AWS SAM compara a referência não resolvida, não o valor resolvido. Um exemplo é uma variável de ambiente definida como um parâmetro de modelo. !Ref Se uma implantação alterar somente o valor de um parâmetro de pilha, o modelo em si permanecerá inalterado. Nesse caso, AWS SAM não publica uma nova versão nem atualiza o alias. AWS SAM aplica a alteração somente à \$LATEST versão da função. Para soluções alternativas, veja um exemplo de uso. [AWS::LanguageExtensions](#)

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AutoPublishAliasAllProperties

Especifica quando um novo [AWS::Lambda::Version](#) é criado. Quando true, uma nova versão do Lambda é criada quando qualquer propriedade na função do Lambda é modificada. Quando false, uma nova versão do Lambda é criada somente quando qualquer uma das seguintes propriedades é modificada:

- Environment, MemorySize, ou SnapStart.
- Qualquer alteração que resulte em uma atualização da propriedade Code, como CodeDict, ImageUri, ou InlineCode.

Alterações de parâmetros e publicação de versões

AWS SAM publica uma nova versão somente quando o valor de uma propriedade muda no modelo. Se o valor de uma propriedade vier de uma função intrínseca,

AWS SAM verá a referência não resolvida. Um exemplo é uma variável de ambiente definida a partir !Ref de um parâmetro de modelo. Implantações que alteram somente o valor do parâmetro não publicam uma nova versão, mesmo com `AutoPublishAliasAllProperties: true`. AWS SAM aplica a alteração \$LATEST somente a.

Use uma das seguintes soluções alternativas para AWS SAM forçar a publicação de uma nova versão quando somente o valor de um parâmetro for alterado:

- Padrão AWS SAM e CloudFormation implantações: adicione a [AWS::LanguageExtensions transformação](#) anterior `AWS::Serverless-2016-10-31` na Transform seção do seu modelo. CloudFormation resolve referências a seus valores reais antes de AWS SAM computar o hash da versão. Uma alteração no valor do parâmetro, em seguida, publica uma nova versão e atualiza o alias:

Transform:

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

- AWS Serverless Application Repository aplicativos: AWS Serverless Application Repository suporta somente a `AWS::Serverless-2016-10-31` transformação e rejeita os modelos que declaram `AWS::LanguageExtensions`. Para aplicativos que você publica por meio do AWS Serverless Application Repository, forneça o valor diretamente como um literal no modelo em vez de fazer referência a um parâmetro. Isso torna a alteração visível ao AWS SAM computar o hash da versão.

Essa propriedade precisa do `AutoPublishAlias` para ser definida.

Se `AutoPublishCodeSha256` também for especificado, seu comportamento terá precedência sobre `AutoPublishAliasAllProperties: true`.

Tipo: booleano

Obrigatório: não

Valor padrão: `false`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AutoPublishCodeSha256

Quando usada, essa string trabalha com o valor `CodeUri` para determinar se uma nova versão do Lambda precisa ser publicada. Essa propriedade muitas vezes é usada para resolver o seguinte problema de implantação: um pacote de implantação está armazenado em um local do Amazon S3 e é substituído por um novo pacote de implantação que contém o código de função do Lambda atualizado, mas a propriedade `CodeUri` permanece inalterada (ao contrário do novo pacote de implantação que está sendo carregado em um novo local do Amazon S3 e o `CodeUri` sendo transferido para o novo local).

Esse problema é marcado por um AWS SAM modelo com as seguintes características:

- O objeto `DeploymentPreference` está configurado para implantações graduais (conforme descrito em [Implantando aplicativos sem servidor gradualmente com AWS SAM](#))
- A propriedade `AutoPublishAlias` está definida e não muda entre as implantações
- A propriedade `CodeUri` está definida e não muda entre as implantações.

Nesse cenário, a atualização do `AutoPublishCodeSha256` resulta na criação bem-sucedida de uma nova versão do Lambda. No entanto, o novo código de função implantado no Amazon S3 não será reconhecido. Para reconhecer o novo código de função, considere usar o controle de versionamento em seu bucket do Amazon S3. Especifique a propriedade `Version` da sua função do Lambda e configure seu bucket para sempre usar o pacote de implantação mais recente.

Nesse cenário, para acionar a implantação gradual com êxito, você deve fornecer um valor exclusivo para `AutoPublishCodeSha256`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

CapacityProviderConfig

Configura o provedor de capacidade ao qual as versões publicadas da função serão anexadas. Isso permite que a função seja executada em instâncias EC2 de propriedade do cliente gerenciadas por instâncias gerenciadas Lambda.

Digite: [CapacityProviderConfig](#)

Obrigatório: não

CloudFormation compatibilidade: o SAM nivela a propriedade passada para a [CapacityProviderConfig](#) propriedade de um `AWS::Lambda::Function` recurso e reconstrói a estrutura aninhada.

CodeSigningConfigArn

O ARN do recurso [AWS::Lambda::CodeSigningConfig](#), usado para habilitar a assinatura de código para essa função. Para obter mais informações sobre assinatura de código, consulte [Configure a assinatura de código para seu AWS SAM aplicação](#).

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CodeSigningConfigArn](#) propriedade de um `AWS::Lambda::Function` recurso.

CodeUri

O código da função. Os valores aceitos são:

- O URI do Amazon S3 da função. Por exemplo, `s3://bucket-123456789/sam-app/1234567890abcdefg`.
- O caminho local para a função. Por exemplo, `hello_world/`.
- Um objeto [FunctionCode](#).

Note

Se você fornecer o URI ou objeto [FunctionCode](#) do Amazon S3 de uma função, deverá fazer referência a um [pacote de implantação do Lambda](#) válido.

Se você fornecer um caminho de arquivo local, use o AWS SAMCLI para carregar o arquivo local na implantação. Para saber mais, consulte [Como AWS SAM carrega arquivos locais na implantação](#).

Se você usar funções intrínsecas na `CodeUri` propriedade, não AWS SAM será capaz de analisar corretamente os valores. Em vez disso, considere usar [AWS::LanguageExtensions transform](#).

Tipo: [String | [FunctionCode](#)]

Obrigatório: condicional. Quando PackageType está definido como Zip, um dos CodeUri ou InlineCode é obrigatório.

CloudFormation compatibilidade: essa propriedade é semelhante à [Code](#) propriedade de um AWS::Lambda::Function recurso. As propriedades aninhadas do Amazon S3 têm nomes diferentes.

DeadLetterQueue

Configura um tópico do Amazon Simple Notification Service (Amazon SNS) ou uma fila do Amazon Simple Queue Service (Amazon SQS) em que o Lambda envia eventos que não são processados. Para obter mais informações sobre a funcionalidade de fila de mensagens mortas, consulte [Dead-letter filas no Guia](#) do desenvolvedor.AWS Lambda

Note

Se a origem do evento da função do Lambda for uma fila do Amazon SQS, configure uma fila de mensagens não entregues para a fila de origem, não para a função do Lambda. A fila de mensagens não entregues que você configura para uma função é usada para a [fila de invocação assíncrona](#) da função, e não para filas de origem de evento.

Tipo: Mapa | [DeadLetterQueue](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [DeadLetterConfig](#) propriedade de um AWS::Lambda::Function recurso. CloudFormation No tipo é derivado doTargetArn, enquanto em AWS SAM você deve passar o tipo junto com TargetArn o.

DeploymentPreference

As configurações para permitir implantações graduais do Lambda.

Se um DeploymentPreference objeto for especificado, AWS SAM cria um [AWS::CodeDeploy::Application](#) chamado ServerlessDeploymentApplication (um por pilha), um [AWS::CodeDeploy::DeploymentGroup](#) chamado *<function-logical-id>*DeploymentGroup e um [AWS::IAM::Role](#) chamadoCodeDeployServiceRole.

Digite: [DeploymentPreference](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Consulte, também: Para obter mais informações sobre essa propriedade, consulte [Implantando aplicativos sem servidor gradualmente com AWS SAM](#).

Description

Uma descrição da função.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::Lambda::Function` recurso.

DurableConfig

Configuração para funções duráveis. Permite a execução em estado com recursos automáticos de verificação e repetição.

Digite: [DurableConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Environment

A configuração para o ambiente de runtime.

Type: [Environment](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Environment](#) propriedade de um `AWS::Lambda::Function` recurso.

EphemeralStorage

Um objeto que especifica o espaço em disco, em MB, disponível para sua função do Lambda no `/tmp`.

Para obter mais informações sobre essa propriedade, consulte [Ambiente de execução](#) do AWS Lambda no Guia do desenvolvedor.

Digite: [EphemeralStorage](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EphemeralStorage](#) propriedade de um `AWS::Lambda::Function` recurso.

EventInvokeConfig

O objeto que descreve a configuração de invocação de eventos em uma função do Lambda.

Digite: [EventInvokeConfiguration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Events

Especifica os eventos que acionam essa função. Os eventos consistem de um tipo e um conjunto de propriedades que dependem desse tipo.

Digite: [EventSource](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

FileSystemConfigs

Lista de [FileSystemConfig](#) objetos que especificam as configurações de conexão para um sistema de arquivos Amazon Elastic File System (Amazon EFS) ou um sistema de arquivos Amazon S3 Files. Você pode anexar um ponto de acesso Amazon EFS ou um ponto de acesso S3 Files, mas não ambos.

Cada `FileSystemConfig` objeto contém um `Arn` (o ARN do ponto de acesso) e um `LocalMountPath` (o caminho em que o sistema de arquivos está montado na função). Para o Amazon EFS, o ARN é um ARN do ponto de acesso do Amazon EFS. Para arquivos S3, o ARN é um `AWS::S3Files::AccessPoint` ARN.

Se o seu modelo contiver um [AWS::EFS::MountTarget](#) recurso, você também deverá especificar um DependsOn atributo de recurso para garantir que o ponto de acesso NFS seja criado ou atualizado antes da função. Da mesma forma, se seu modelo contiver um [AWS::S3Files::MountTarget](#) recurso, você deverá especificar um DependsOn atributo para o destino de montagem do S3 Files.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FileSystemConfigs](#) propriedade de um [AWS::Lambda::Function](#) recurso.

FunctionName

Um nome para a função. Se você não especificar um nome, um nome exclusivo é gerado para você.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FunctionName](#) propriedade de um [AWS::Lambda::Function](#) recurso.

FunctionScalingConfig

Configura o comportamento de escalabilidade das funções Lambda executadas em provedores de capacidade. Define o número mínimo e máximo de ambientes de execução.

Digite: [FunctionScalingConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FunctionScalingConfig](#) propriedade de um [AWS::Lambda::Function](#) recurso.

FunctionUrlConfig

O objeto que descreve o URL da função. O URL da função é um endpoint HTTP(S) que você pode usar para invocar a função.

Para obter mais informações, consulte [URLs da função](#) no AWS Lambda Guia do desenvolvedor.

Digite: [FunctionUrlConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Handler

A função em seu código que é chamada para iniciar a execução. Esta propriedade só será necessária se a PackageType propriedade estiver definida como Zip.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Handler](#) propriedade de um `AWS::Lambda::Function` recurso.

ImageConfig

O objeto usado para definir as configurações da imagem do contêiner Lambda. Para obter mais informações, consulte [Uso de imagens do contêiner com o Lambda](#) no AWS Lambda Guia do desenvolvedor.

Digite: [ImageConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ImageConfig](#) propriedade de um `AWS::Lambda::Function` recurso.

ImageUri

O URI do repositório Amazon Elastic Container Registry (Amazon ECR) para a imagem de contêiner da função do Lambda. Essa propriedade só se aplica se a propriedade PackageType estiver definida como Image, caso contrário, ela será ignorada. Para obter mais informações, consulte [Uso de imagens do contêiner com o Lambda](#) no AWS Lambda Guia do desenvolvedor.

Note

Se a PackageType propriedade estiver definida como Image, ela será obrigatória ou você deverá criar seu aplicativo com Metadata as entradas necessárias no arquivo AWS

SAM de modelo. `ImageUri` Para obter mais informações, consulte [Compilação padrão com AWS SAM](#).

A criação de seu aplicativo com as entradas Metadata necessárias tem precedência sobre `ImageUri`, portanto, se você especificar ambas, `ImageUri` será ignorada.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ImageUri](#) propriedade do tipo de `AWS::Lambda::Function` Code dados.

InlineCode

O código da função do Lambda que é escrito diretamente no modelo. Essa propriedade só se aplica se a propriedade `PackageType` estiver definida como `Zip`, caso contrário, ela será ignorada.

Note

Se a propriedade `PackageType` estiver definida como `Zip` (padrão), uma das `CodeUri` ou `InlineCode` será obrigatória.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ZipFile](#) propriedade do tipo de `AWS::Lambda::Function` Code dados.

KmsKeyArn

O ARN de uma chave AWS Key Management Service (AWS KMS) que o Lambda usa para criptografar e descriptografar as variáveis de ambiente da sua função.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [KmsKeyArn](#) propriedade de um `AWS::Lambda::Function` recurso.

Layers

A lista de ARNs `LayerVersion` que essa função deve usar. A ordem especificada aqui é a ordem na qual eles serão importados ao executar a função do Lambda. A versão é um ARN completo, incluindo a versão, ou uma referência a um `LayerVersion` recurso. Por exemplo, uma referência para a `LayerVersion` será `!Ref MyLayer`, enquanto um ARN completo, incluindo a versão, será `arn:aws:lambda:region:account-id:layer:layer-name:version`.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Layers](#) propriedade de um `AWS::Lambda::Function` recurso.

LoggingConfig

As configurações da função Amazon CloudWatch Logs.

Digite: [LoggingConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [LoggingConfig](#) propriedade de um `AWS::Lambda::Function` recurso.

MemorySize

O tamanho da memória em MB alocado por invocação da função.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MemorySize](#) propriedade de um `AWS::Lambda::Function` recurso.

PackageType

O tipo de pacote de implantação da função do Lambda. Para obter mais informações, consulte [Pacote de implantação do Lambda](#) no AWS Lambda Guia do desenvolvedor.

Observações:

1. Se essa propriedade for definida como Zip (padrão), então uma `CodeUri` ou outra `InlineCode` se aplica e `ImageUri` é ignorada.
2. Se essa propriedade for definida como Image, então `ImageUri` só se aplica, `CodeUri` e ambas `InlineCode` são ignoradas. O repositório Amazon ECR necessário para armazenar a imagem do contêiner da função pode ser criado automaticamente pelo AWS SAMCLI. Para obter mais informações, consulte [sam deploy](#).

Valores válidos: Zip ou Image

Tipo: string

Obrigatório: não

Padrão: Zip

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [PackageType](#) propriedade de um `AWS::Lambda::Function` recurso.

PermissionsBoundary

O ARN de um limite de permissões a ser usado para a função de execução dessa função. Essa propriedade funciona somente se a função for gerada para você.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [PermissionsBoundary](#) propriedade de um `AWS::IAM::Role` recurso.

Policies

Políticas de permissão para essa função. As políticas serão anexadas à função de execução padrão AWS Identity and Access Management (IAM) da função.

Essa propriedade aceita um único valor ou uma lista de valores. Os valores permitidos incluem:

- [AWS SAM Modelos de políticas](#).
- O ARN de uma [política AWS gerenciada ou política gerenciada pelo cliente](#).
- O nome de uma política AWS gerenciada da [lista](#) a seguir.

- Uma [política do IAM em linha](#) formatada em YAML como um mapa.

 Note

Se você especificar a propriedade `Role`, essa propriedade será ignorada.

Tipo: String | List | Map

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Policies](#) propriedade de um `AWS::IAM::Role` recurso.

`PublishToLatestPublished`

Especifica se a última versão da função deve ser publicada quando a função for atualizada.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [PublishToLatestPublished](#) propriedade de um `AWS::Lambda::Function` recurso.

`PropagateTags`

Indique se deseja ou não passar as tags da propriedade `Tags` para os recursos [AWS::Serverless::Function](#) gerados. Especifique `True` para propagar as tags nos recursos gerados.

Tipo: booleano

Obrigatório: não

Padrão: `False`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

`ProvisionedConcurrencyConfig`

A configuração de simultaneidade provisionada do alias de uma função.

 Note

`ProvisionedConcurrencyConfig` só pode ser especificado se estiver `AutoPublishAlias` definido. Caso contrário, ocorrerá um erro.

Digite: [ProvisionedConcurrencyConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ProvisionedConcurrencyConfig](#) propriedade de um `AWS::Lambda::Alias` recurso.

RecursiveLoop

O status da configuração de detecção de loops recursivos da função.

Quando esse valor é definido como `Allow` e o Lambda detecta que a função está sendo invocada como parte de um loop recursivo, ele não executa qualquer ação.

Quando esse valor é definido como `Terminate` e o Lambda detecta a invocação da função como parte de um loop recursivo, ele interrompe a invocação da função e notifica você.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RecursiveLoop](#) propriedade do `AWS::Lambda::Function` recurso.

ReservedConcurrentExecutions

O número máximo de execuções simultâneas que você deseja reservar para a função.

Para obter mais informações sobre essa propriedade, consulte [Escalabilidade da Função do Lambda](#) no AWS Lambda Guia do desenvolvedor.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ReservedConcurrentExecutions](#) propriedade de um `AWS::Lambda::Function` recurso.

Role

O ARN de um perfil do IAM a ser usado como perfil de execução dessa função.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Role](#) propriedade de um `AWS::Lambda::Function` recurso. Isso é necessário em CloudFormation , mas não em AWS SAM. Se uma função não for especificada, uma será criada para você com uma ID lógica de `<function-logical-id>Role`.

RolePath

O caminho para a função de execução do IAM da função.

Use essa propriedade quando a função for gerada para você. Não use quando a função for especificada com a propriedade Role.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Path](#) propriedade de um `AWS::IAM::Role` recurso.

Runtime

O identificador do [runtime](#) da função. Esta propriedade só será necessária se a `PackageType` propriedade estiver definida como `Zip`.

Note

Se você especificar o `provided` identificador dessa propriedade, poderá usar o atributo `Metadata resource` para AWS SAM instruir a criar o tempo de execução personalizado que essa função exige. Para obter mais informações sobre a criação de tempos de execução personalizados, consulte [Criando funções de eventos Lambda com tempos de execução personalizados em AWS SAM](#).

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Runtime](#) propriedade de um `AWS::Lambda::Function` recurso.

RuntimeManagementConfig

Configure opções de gerenciamento de runtime para suas funções do Lambda, como atualizações do ambiente de runtime, comportamento de reversão e seleção de uma versão de runtime específica. Para saber mais, consulte as [atualizações de runtime do Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Digite: [RuntimeManagementConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RuntimeManagementConfig](#) propriedade de um `AWS::Lambda::Function` recurso.

SnapStart

Crie uma captura de tela de qualquer nova versão da função do Lambda. Uma captura de tela é um estado em cache da sua função inicializada, incluindo todas as suas dependências. A função é inicializada apenas uma vez e o estado em cache é reutilizado para todas as futuras invocações, melhorando o desempenho do aplicativo ao reduzir o número de vezes que sua função deve ser inicializada. Para saber mais, consulte [Melhorando o desempenho de startups com o Lambda SnapStart](#) no Guia do AWS Lambda desenvolvedor.

Digite: [SnapStart](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SnapStart](#) propriedade de um `AWS::Lambda::Function` recurso.

SourceKMSKeyArn

Representa o ARN da chave do KMS usada para criptografar o código da função do CEP do cliente.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SourceKMSKeyArn](#) propriedade de um tipo de `AWS::Lambda::Function` Code dados.

Tags

Um mapa (string para string) que especifica as tags a serem adicionadas a esse estágio do API Gateway. Para obter detalhes sobre chaves e valores válidos para tags, consulte [Requisitos de chave e valor de tags](#) no AWS Lambda Guia do desenvolvedor.

Quando a pilha é criada, adiciona AWS SAM automaticamente uma `lambda:createdBy: SAM` tag a essa função Lambda e às funções padrão que são geradas para essa função.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Tags](#) propriedade de um `AWS::Lambda::Function` recurso. A Tags propriedade in AWS SAM consiste em pares de valores-chave (enquanto CloudFormation nessa propriedade consiste em uma lista de Tag objetos). Além disso, adiciona AWS SAM automaticamente uma `lambda:createdBy: SAM` tag a essa função Lambda e às funções padrão que são geradas para essa função.

TenancyConfig

Configuração para o modo de isolamento de inquilinos do Lambda. Garante que os ambientes de execução nunca sejam compartilhados entre diferentes IDs de locatário, fornecendo isolamento em nível de computação para aplicativos multilocatários.

Digite: [TenancyConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TenancyConfig](#) propriedade de um `AWS::Lambda::Function` recurso.

Timeout

O tempo máximo em segundos em que a função pode ser executada até ser interrompida.

Tipo: inteiro

Obrigatório: não

Padrão: 3

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Timeout](#) propriedade de um `AWS::Lambda::Function` recurso.

Tracing

A string que especifica o modo de X-Ray rastreamento da função.

- **Active**— Ativa o X-Ray rastreamento da função.
- **Disabled**— Desativa a X-Ray função.
- **PassThrough**— Ativa o X-Ray rastreamento da função. A decisão de amostragem é delegada aos serviços posteriores.

Se especificada como **Active** ou **PassThrough** e a propriedade **Role** não estiver definida, o AWS SAM adiciona a política `arn:aws:iam::aws:policy/AWSXrayWriteOnlyAccess` à função de execução do Lambda que ela cria para você.

Para obter mais informações sobre X-Ray, consulte [Usando AWS Lambda com AWS X-Ray](#) no Guia do AWS Lambda desenvolvedor.

Valores válidos: `[Active|Disabled|PassThrough]`

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [TracingConfig](#) propriedade de um `AWS::Lambda::Function` recurso.

VersionDescription

Especifica o campo **Description** que é adicionado ao novo recurso da versão do Lambda.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::Lambda::Version` recurso.

VersionDeletionPolicy

Especifica a política de exclusão para o recurso da versão Lambda que é criado quando definido. **AutoPublishAlias** Isso controla se o recurso da versão é retido ou excluído quando a pilha é excluída.

Valores válidos: Delete, Retain ou Snapshot

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente. Ele define o DeletionPolicy atributo no `AWS::Lambda::Version` recurso gerado.

VpcConfig

A configuração que permite que essa função acesse recursos privados em sua nuvem privada virtual (VPC).

Digite: [VpcConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [VpcConfig](#) propriedade de um `AWS::Lambda::Function` recurso.

Valores de retorno

Ref.

Quando o ID lógico desse recurso é fornecido à função intrínseca do Ref, ele retorna o nome do recurso da função do Lambda subjacente.

Para obter mais informações sobre como usar a função Ref, consulte [Ref](#) no Guia do usuário do AWS CloudFormation .

Fn:: GetAtt

`Fn::GetAtt` retorna um valor para um atributo especificado deste tipo. Estes são os atributos disponíveis e os valores de retorno de amostra.

Para obter mais informações sobre o uso do `Fn::GetAtt`, consulte [Fn::GetAtt](#) no AWS CloudFormation Guia do usuário.

Arn

O ARN da função do Lambda subjacente.

Exemplos

Função simples

Veja a seguir um exemplo básico de um recurso [AWS::Serverless::Function](#) do tipo de pacote Zip (padrão) e código de função em um bucket do Amazon S3.

YAML

```
Type: AWS::Serverless::Function
Properties:
  Handler: index.handler
  Runtime: python3.9
  CodeUri: s3://bucket-name/key-name
```

Exemplo de propriedades da função

Veja a seguir um exemplo de um tipo [AWS::Serverless::Function](#) de pacote Zip (padrão) que usa `InlineCode`, `Layers`, `TracingPolicies`, sistema de arquivos Amazon EFS e uma fonte de eventos.

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 exemplo

Veja a seguir um exemplo de uma ImageConfig para uma função do Lambda do tipo de pacote 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 exemplos

Uma função do Lambda configurada para atualizar seu ambiente de runtime de acordo com o comportamento atual:

```

TestFunction
  Type: AWS::Serverless::Function
  Properties:
    ...
    Runtime: python3.9
    RuntimeManagementConfig:

```

```
UpdateRuntimeOn: Auto
```

Uma função do Lambda configurada para atualizar seu ambiente de runtime quando a função é atualizada:

```
TestFunction
  Type: AWS::Serverless::Function
  Properties:
    ...
    Runtime: python3.9
    RuntimeManagementConfig:
      UpdateRuntimeOn: FunctionUpdate
```

Uma função do Lambda configurada para atualizar seu ambiente de runtime manualmente:

```
TestFunction
  Type: AWS::Serverless::Function
  Properties:
    ...
    Runtime: python3.9
    RuntimeManagementConfig:
      RuntimeVersionArn: arn:aws:lambda:us-east-1::runtime:4c459dd0104ee29ec65dcad056c0b3ddbe20d6db76b265ade7eda9a066859b1e
      UpdateRuntimeOn: Manual
```

SnapStart exemplos

Exemplo de uma função Lambda SnapStart ativada para futuras versões:

```
TestFunc
  Type: AWS::Serverless::Function
  Properties:
    ...
    SnapStart:
      ApplyOn: PublishedVersions
```

TenancyConfig exemplos

Exemplo de uma função Lambda com o modo de isolamento de inquilino ativado:

```
TestFunction
  Type: AWS::Serverless::Function
```

```

Properties:
  ...
  TenancyConfig:
    TenantIsolationMode: PER_TENANT

```

Exemplo de sistema de arquivos S3 Files

O exemplo a seguir cria uma função Lambda que monta um sistema de arquivos Amazon S3 Files. O modelo cria um bucket S3, um sistema de arquivos S3 Files suportado por esse bucket, um destino de montagem em uma sub-rede VPC e um ponto de acesso. A função monta o ponto de acesso /mnt/s3files e pode ler e gravar arquivos que são sincronizados com o bucket do S3.

Note

Você pode anexar um sistema de arquivos Amazon EFS ou um sistema de arquivos S3 Files a uma função Lambda, mas não os dois ao mesmo tempo.

YAML

```

AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: Lambda function with S3 Files file system

Resources:
  # VPC and networking
  MyVpc:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.0.0.0/16
      EnableDnsSupport: true
      EnableDnsHostnames: true

  MySubnet:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref MyVpc
      CidrBlock: 10.0.1.0/24
      AvailabilityZone: !Select [0, !GetAZs '']

  MySecurityGroup:
    Type: AWS::EC2::SecurityGroup

```


Properties:

GroupDescription: Security group for Lambda and S3 Files

VpcId: !Ref MyVpc

SecurityGroupIngress:

- IpProtocol: tcp
- FromPort: 2049
- ToPort: 2049
- CidrIp: 10.0.0.0/16

S3 bucket for file storage

MyS3Bucket:

Type: AWS::S3::Bucket

IAM role for S3 Files to access the bucket

S3FilesRole:

Type: AWS::IAM::Role

Properties:

AssumeRolePolicyDocument:

Version: '2012-10-17'

Statement:

- Effect: Allow
- Principal:
 - Service: elasticfilesystem.amazonaws.com
- Action: sts:AssumeRole

Policies:

- PolicyName: S3FilesBucketAccess
- PolicyDocument:
 - Version: '2012-10-17'
 - Statement:
 - Effect: Allow
 - Action:
 - s3:GetObject
 - s3:PutObject
 - s3:DeleteObject
 - s3:ListBucket
 - Resource:
 - !GetAtt MyS3Bucket.Arn
 - !Sub '\${MyS3Bucket.Arn}/*'

S3 Files resources

MyS3FilesFileSystem:

Type: AWS::S3Files::FileSystem

Properties:

Bucket: !GetAtt MyS3Bucket.Arn

```
RoleArn: !GetAtt S3FilesRole.Arn
```

```
MyS3FilesMountTarget:
```

```
  Type: AWS::S3Files::MountTarget
```

```
  Properties:
```

```
    FileSystemId: !Ref MyS3FilesFileSystem
```

```
    SubnetId: !Ref MySubnet
```

```
    SecurityGroups:
```

```
      - !Ref MySecurityGroup
```

```
MyS3FilesAccessPoint:
```

```
  Type: AWS::S3Files::AccessPoint
```

```
  Properties:
```

```
    FileSystemId: !Ref MyS3FilesFileSystem
```

```
    PosixUser:
```

```
      Uid: '1000'
```

```
      Gid: '1000'
```

```
    RootDirectory:
```

```
      Path: /lambda
```

```
    CreationInfo:
```

```
      OwnerUid: '1000'
```

```
      OwnerGid: '1000'
```

```
      Permissions: '750'
```

```
# Lambda function with S3 Files mount
```

```
MyFunction:
```

```
  Type: AWS::Serverless::Function
```

```
  DependsOn: MyS3FilesMountTarget
```

```
  Properties:
```

```
    Handler: index.handler
```

```
    Runtime: python3.12
```

```
    Timeout: 120
```

```
    VpcConfig:
```

```
      SecurityGroupIds:
```

```
        - !Ref MySecurityGroup
```

```
      SubnetIds:
```

```
        - !Ref MySubnet
```

```
    FileSystemConfigs:
```

```
      - Arn: !GetAtt MyS3FilesAccessPoint.AccessPointArn
```

```
        LocalMountPath: /mnt/s3files
```

```
    Policies:
```

```
      - Version: '2012-10-17'
```

```
        Statement:
```

```
          - Effect: Allow
```

```

    Action:
      - s3files:ClientMount
      - s3files:ClientWrite
    Resource: !GetAtt MyS3FilesAccessPoint.AccessPointArn
  InlineCode: |
    import os

    def handler(event, context):
        # Write a file to the S3 Files mount
        with open('/mnt/s3files/hello.txt', 'w') as f:
            f.write('Hello from Lambda!')

        # List files at the mount path
        files = os.listdir('/mnt/s3files')
        return {'files': files}

```

DeadLetterQueue

Especifica uma fila SQS ou um tópico do SNS para o qual (AWS Lambda Lambda) envia eventos quando não consegue processá-los. Para obter mais informações sobre a funcionalidade de fila de letras mortas, consulte [Dead-letter filas no Guia](#) do AWS Lambda desenvolvedor.

O SAM adicionará automaticamente a permissão apropriada à sua função de execução da Lambda Event Function para dar acesso ao serviço Lambda ao recurso. sqs: SendMessage será adicionado às filas SQS e SNS:Publish para tópicos do SNS.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```

TargetArn: String
Type: String

```

Propriedades

TargetArn

O nome de recurso da Amazon (ARN) de uma fila do Amazon SQS ou um tópico do Amazon SNS.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TargetArn](#) propriedade do tipo de `AWS::Lambda::Function` DeadLetterConfig dados.

Type

O tipo de fila de mensagens não entregues.

Valores válidos: SNS, SQS

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

DeadLetterQueue

Exemplo de fila de mensagens não entregues para um tópico do SNS.

YAML

```
DeadLetterQueue:
  Type: SNS
  TargetArn: arn:aws:sns:us-east-2:123456789012:my-topic
```

DeploymentPreference

Especifica as configurações para permitir implantações graduais do Lambda. Para obter mais informações sobre como configurar implantações graduais do Lambda, consulte [Implantando aplicativos sem servidor gradualmente com AWS SAM](#).

Note

Você deve especificar um `AutoPublishAlias` em seu [AWS::Serverless::Função](#) para usar um objeto `DeploymentPreference`, caso contrário, ocorrerá um erro.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Alarms: List  
Enabled: Boolean  
Hooks: Hooks  
PassthroughCondition: Boolean  
Role: String  
TriggerConfigurations: List  
Type: String
```

Propriedades

Alarms

Uma lista de CloudWatch alarmes que você deseja que sejam acionados por quaisquer erros gerados pela implantação.

Essa propriedade aceita a função intrínseca `Fn::If`. Consulte a seção Exemplos na parte inferior deste tópico para ver um exemplo de modelo que usa `Fn::If`.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Enabled

Se essa preferência de implantação está habilitada.

Tipo: booleano

Obrigatório: não

Padrão: verdadeiro

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Hooks

Funções de eventos Lambda de validação que são executadas antes e depois da mudança de tráfego.

Tipo: [Ganchos](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

PassthroughCondition

Se for True, e se essa preferência de implantação estiver ativada, a Condição da função será passada para o CodeDeploy recurso gerado. Geralmente, você deve definir isso como Verdadeiro. Caso contrário, o CodeDeploy recurso seria criado mesmo se a Condição da função fosse resolvida como False.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Role

Um ARN de função do IAM que CodeDeploy será usado para mudança de tráfego. Um perfil do IAM não será criado se ele for fornecido.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

TriggerConfigurations

Uma lista das configurações de gatilho que você deseja associar ao grupo de implantação. Usado para notificar um tópico do SNS sobre eventos do ciclo de vida.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TriggerConfigurations](#) propriedade de um `AWS::CodeDeploy::DeploymentGroup` recurso.

Type

No momento, existem duas categorias de tipos de implantação: Linear e Canário. Para obter mais informações sobre os tipos de implantação disponíveis, consulte [Implantando aplicativos sem servidor gradualmente com AWS SAM](#).

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

DeploymentPreference com ganchos pré e pós-trânsito.

Exemplo de preferência de implantação que contém ganchos pré e pós-tráfego.

YAML

```
DeploymentPreference:
  Enabled: true
  Type: Canary10Percent10Minutes
  Alarms:
    - !Ref: AliasErrorMetricGreaterThanZeroAlarm
    - !Ref: LatestVersionErrorMetricGreaterThanZeroAlarm
  Hooks:
    PreTraffic:
      !Ref: PreTrafficLambdaFunction
    PostTraffic:
      !Ref: PostTrafficLambdaFunction
```

DeploymentPreference com função intrínseca Fn: !If

Exemplo de preferência de implantação que usa Fn: !If para configurar alarmes. Neste exemplo, Alarm1 será configurado se MyCondition for true, Alarm2 e Alarm5 será configurado se MyCondition for false.

YAML

```
DeploymentPreference:
  Enabled: true
  Type: Canary10Percent10Minutes
  Alarms:
    Fn::If:
      - MyCondition
      - - Alarm1
        - Alarm2
        - Alarm5
```

Hooks

Funções de eventos Lambda de validação que são executadas antes e depois da mudança de tráfego.

Note

As funções Lambda referenciadas nessa propriedade configuram o objeto `CodeDeployLambdaAliasUpdate` do recurso resultante [AWS::Lambda::Alias](#). Para obter mais informações, consulte a [CodeDeployLambdaAliasUpdate Política](#) no Guia AWS CloudFormation do Usuário.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
PostTraffic: String
PreTraffic: String
```

Propriedades

PostTraffic

Função do Lambda que é executada após a mudança de tráfego.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

PreTraffic

Função do Lambda que é executada antes da mudança de tráfego.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Ganchos

Exemplos de funções de gancho

YAML

```
Hooks:
  PreTraffic:
    Ref: PreTrafficLambdaFunction
  PostTraffic:
    Ref: PostTrafficLambdaFunction
```

DurableConfig

Define configurações de execução duráveis para AWS Lambda funções. Funções duráveis podem ser executadas por até um ano e verificar automaticamente o progresso, permitindo fluxos de trabalho de longa duração e aplicativos tolerantes a falhas. Para obter mais informações sobre funções duráveis, consulte [Funções duráveis do Lambda](#) no Guia do AWS Lambda desenvolvedor.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
ExecutionTimeout: Integer  
RetentionPeriodInDays: Integer
```

Propriedades

ExecutionTimeout

A quantidade de tempo (em segundos) em que o Lambda permite que uma função durável seja executada antes de interrompê-la. O máximo é um ano de 366 dias ou 31.622.400 segundos.

Tipo: inteiro

Obrigatório: sim

Minimum (Mínimo): 1

Máximo: 31622400

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ExecutionTimeout](#) propriedade do tipo de `AWS::Lambda::Function DurableConfig` dados.

RetentionPeriodInDays

O número de dias após o encerramento de uma execução duradoura em que o Lambda retém seu histórico, de um a 90 dias. O padrão são 14 dias.

Tipo: inteiro

Obrigatório: não

Padrão: 14

Minimum (Mínimo): 1

Máximo: 90

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RetentionPeriodInDays](#) propriedade do tipo de `AWS::Lambda::Function DurableConfig` dados.

Exemplos

DurableConfig

Exemplo de configuração durável para uma função com tempo limite de execução de 1 hora e período de retenção de 7 dias.

YAML

```
DurableConfig:
  ExecutionTimeout: 3600
  RetentionPeriodInDays: 7
```

EventInvokeConfiguration

Opções de configuração para invocações [assíncronas](#) de Lambda Alias ou versão.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
DestinationConfig: EventInvokeDestinationConfiguration
MaximumEventAgeInSeconds: Integer
MaximumRetryAttempts: Integer
```

Propriedades

DestinationConfig

Um objeto de configuração que especifica o destino de um evento depois que o Lambda processá-lo.

Digite: [EventInvokeDestinationConfiguration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [DestinationConfig](#) propriedade de um `AWS::Lambda::EventInvokeConfig` recurso. O SAM requer um parâmetro extra, "Tipo", que não existe em CloudFormation.

MaximumEventAgeInSeconds

A idade máxima de uma solicitação que o Lambda envia a uma função para processamento.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumEventAgeInSeconds](#) propriedade de um `AWS::Lambda::EventInvokeConfig` recurso.

MaximumRetryAttempts

O número máximo de vezes para tentar novamente antes da função retornar um erro.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumRetryAttempts](#) propriedade de um `AWS::Lambda::EventInvokeConfig` recurso.

Exemplos

MaximumEventAgeInSeconds

MaximumEventAgeInSeconds exemplo

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

Um objeto de configuração que especifica o destino de um evento depois que o Lambda processá-lo.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
OnFailure: OnFailure  
OnSuccess: OnSuccess
```

Propriedades

OnFailure

Um destino para eventos que tiveram falha no processamento.

Digite: [OnFailure](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [OnFailure](#) propriedade de um `AWS::Lambda::EventInvokeConfig` recurso. Requer `Type`, uma propriedade adicional somente para SAM.

OnSuccess

Um destino para eventos que foram processados com êxito.

Digite: [OnSuccess](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [OnSuccess](#) propriedade de um `AWS::Lambda::EventInvokeConfig` recurso. Requer `Type`, uma propriedade adicional somente para SAM.

Exemplos

OnSuccess

OnSuccess exemplo

YAML

```
EventInvokeConfig:
  DestinationConfig:
    OnSuccess:
      Type: SQS
      Destination: arn:aws:sqs:us-west-2:012345678901:my-queue
    OnFailure:
      Type: Lambda
      Destination: !GetAtt DestinationLambda.Arn
```

OnFailure

Um destino para eventos que tiveram falha no processamento.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Destination: String
Type: String
```

Propriedades

Destination

O nome de recurso da Amazon (ARN) do recurso de destino.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é semelhante à [OnFailure](#) propriedade de um `AWS::Lambda::EventInvokeConfig` recurso. O SAM adicionará todas as permissões

necessárias ao perfil do IAM gerado automaticamente associado a essa função para acessar o recurso referenciado nessa propriedade.

Notas adicionais: Se o tipo for Lambda/EventBridge, Destino é obrigatório.

Type

Tipo do recurso referenciado no destino. Os tipos suportados são SQS, SNS, S3, Lambda e EventBridge.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Notas adicionais: Se o tipo for SQS/SNS e a Destination propriedade for deixada em branco, o SQS/SNS recurso será gerado automaticamente pelo SAM. Para referenciar o recurso, use `<function-logical-id>.DestinationQueue` para SQS ou `<function-logical-id>.DestinationTopic` SNS. Se o tipo for Lambda/EventBridge, Destination é obrigatório.

Exemplos

EventInvoke Exemplo de configuração com destinos SQS e Lambda

Neste exemplo, nenhum Destino é fornecido para a OnSuccess configuração do SQS, então o SAM cria implicitamente uma fila SQS e adiciona todas as permissões necessárias. Além disso, neste exemplo, um destino para um recurso Lambda declarado no arquivo de modelo é especificado na OnFailure configuração, então o SAM adiciona as permissões necessárias a essa função de evento do Lambda para chamar a função Lambda de destino.

YAML

```
EventInvokeConfig:
  DestinationConfig:
    OnSuccess:
      Type: SQS
    OnFailure:
      Type: Lambda
```

```
Destination: !GetAtt DestinationLambda.Arn # Arn of a Lambda function declared
in the template file.
```

EventInvoke Exemplo de configuração com destino SNS

Neste exemplo, um destino é fornecido para um tópico do SNS declarado no arquivo de modelo para a OnSuccess configuração.

YAML

```
EventInvokeConfig:
  DestinationConfig:
    OnSuccess:
      Type: SNS
      Destination:
        Ref: DestinationSNS          # Arn of an SNS topic declared in the tempate file
```

OnSuccess

Um destino para eventos que foram processados com êxito.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Destination: String
Type: String
```

Propriedades

Destination

O nome de recurso da Amazon (ARN) do recurso de destino.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é semelhante à [OnSuccess](#) propriedade de um `AWS::Lambda::EventInvokeConfig` recurso. O SAM adicionará todas as permissões necessárias ao perfil do IAM gerado automaticamente associado a essa função para acessar o recurso referenciado nessa propriedade.

Notas adicionais: Se o tipo for `Lambda/EventBridge`, `Destino` é obrigatório.

Type

Tipo do recurso referenciado no destino. Os tipos suportados são SQS, SNS, S3, Lambda e EventBridge.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Notas adicionais: Se o tipo for SQS/SNS e a `Destination` propriedade for deixada em branco, o SQS/SNS recurso será gerado automaticamente pelo SAM. Para referenciar o recurso, use `<function-logical-id>.DestinationQueue` para SQS ou `<function-logical-id>.DestinationTopic` SNS. Se o tipo for `Lambda/EventBridge`, `Destination` é obrigatório.

Exemplos

EventInvoke Exemplo de configuração com destinos SQS e Lambda

Neste exemplo, nenhum Destino é fornecido para a `OnSuccess` configuração do SQS, então o SAM cria implicitamente uma fila SQS e adiciona todas as permissões necessárias. Além disso, neste exemplo, um destino para um recurso Lambda declarado no arquivo de modelo é especificado na `OnFailure` configuração, então o SAM adiciona as permissões necessárias a essa função de evento do Lambda para chamar a função Lambda de destino.

YAML

```
EventInvokeConfig:
  DestinationConfig:
    OnSuccess:
      Type: SQS
    OnFailure:
      Type: Lambda
```

```
Destination: !GetAtt DestinationLambda.Arn # Arn of a Lambda function declared
in the template file.
```

EventInvoke Exemplo de configuração com destino SNS

Neste exemplo, um destino é fornecido para um tópico do SNS declarado no arquivo de modelo para a OnSuccess configuração.

YAML

```
EventInvokeConfig:
  DestinationConfig:
    OnSuccess:
      Type: SNS
      Destination:
        Ref: DestinationSNS # Arn of an SNS topic declared in the tempate file
```

EventSource

O objeto que descreve a origem dos eventos que acionam a função. Cada evento consiste em um tipo e um conjunto de propriedades que dependem desse tipo. Para obter mais informações sobre as propriedades de cada fonte de eventos, consulte o tópico correspondente a esse tipo.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Properties: AlexaSkill | Api | CloudWatchEvent | CloudWatchLogs | Cognito
| DocumentDB | DynamoDB | EventBridgeRule | HttpApi | IoTRule | Kinesis | MQ | MSK
| S3 | Schedule | ScheduleV2 | SelfManagedKafka | SNS | SQS
Type: String
```

Propriedades

Properties

Objeto que descreve as propriedades desse mapeamento de eventos. O conjunto de propriedades deve estar em conformidade com o Tipo definido.

Tipo : [AlexaSkill](#) | [Api](#) | [CloudWatchEvent](#) | [Cognito](#) | [CloudWatchLogs](#) | [DocumentDB](#) | [DynamoDB](#) | [Io](#) | [Kinesis](#) | [MQ EventBridgeRule](#) | [HttpApi](#) | [MSK](#) | [S3 TRule](#) | [Cronograma](#) | [ScheduleV2](#) | [SNS](#) | [SQS SelfManagedKafka](#)

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Type

O tipo de evento.

Valores válidos: `AlexaSkill`, `Api`, `CloudWatchEvent`, `CloudWatchLogs`, `Cognito`, `DocumentDB`, `DynamoDB`, `EventBridgeRule`, `HttpApi`, `IoTRule`, `Kinesis`, `MQ`, `MSK`, `S3`, `Schedule`, `ScheduleV2`, `SelfManagedKafka`, `SNS`, `SQS`

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

APIEvent

Exemplo de uso de um evento de API

YAML

```
ApiEvent:
  Type: Api
  Properties:
    Method: get
    Path: /group/{user}
    RestApiId:
      Ref: MyApi
```

AlexaSkill

O objeto que descreve um tipo de fonte de evento de `AlexaSkill`.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
SkillId: String
```

Propriedades

SkillId

O Alexa Skill ID para sua skill da Alexa. Para obter mais informações sobre o Skill ID, consulte [Configurar o gatilho para uma função do Lambda](#) na documentação do Alexa Skills Kit.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

AlexaSkillTrigger

Exemplo de evento de habilidade do Alexa

YAML

```
AlexaSkillEvent:  
  Type: AlexaSkill
```

Api

O objeto que descreve um tipo de origem do evento Api. Se um recurso [AWS::Serverless::API](#) for definido, os valores do caminho e do método devem corresponder a uma operação na definição de OpenAPI da API.

Se nenhum [AWS::Serverless::API](#) for definido, a entrada e a saída da função serão uma representação da solicitação HTTP e da resposta HTTP.

Por exemplo, usando a JavaScript API, o código de status e o corpo da resposta podem ser controlados retornando um objeto com as chaves `statusCode` e `body`.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Auth: ApiFunctionAuth
Method: String
Path: String
RequestModel: RequestModel
RequestParameters: List of [ String | RequestParameter ]
RestApiId: String
ResponseTransferMode: String
TimeoutInMillis: Integer
```

Propriedades

Auth

Configuração de autenticação para essa Api+Path+Method específica.

Útil para substituir a configuração `DefaultAuthorizer` da API para um caminho individual, quando nenhum `DefaultAuthorizer` for especificado, ou para substituir a configuração padrão `ApiKeyRequired`.

Digite: [ApiFunctionAuth](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Method

Método HTTP para o qual essa função é invocada. As opções incluem DELETE, GET, HEAD, OPTIONS, PATCH, POST, PUT e ANY. Consulte [Set up an HTTP method](#) in the API Gateway Developer Guide para obter mais detalhes.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Path

Caminho Uri para o qual essa função é invocada. Deve começar com /.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

RequestModel

Solicite o modelo a ser usado para essa Api+Path+Method específica. Isso deve fazer referência ao nome de um modelo especificado na seção Models de um recurso [AWS::Serverless::API](#).

Digite: [RequestModel](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

RequestParameters

Solicite a configuração dos parâmetros para este Api+Path+Method específico. Todos os nomes de parâmetros devem começar com `method.request` e devem ser limitados a `method.request.header`, `method.request.querystring` ou a `method.request.path`.

Uma lista pode conter cadeias de caracteres de nomes de parâmetros e [RequestParameter](#) objetos. Para cadeias de caracteres, as propriedades `Required` e `Caching` serão padronizadas como `false`.

Tipo: Lista de [String | [RequestParameter](#)]

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

RestApiId

Identificador de um RestApi recurso, que deve conter uma operação com o caminho e o método fornecidos. Normalmente, isso é definido para fazer referência a um recurso [AWS::Serverless::API](#) definido nesse modelo.

Se você não definir essa propriedade, AWS SAM cria um [AWS::Serverless::API](#) recurso padrão usando um OpenApi documento gerado. Esse recurso contém uma união de todos os caminhos e métodos definidos por eventos Api no mesmo modelo que não especificam um arquivo RestApiId.

Isso não pode fazer referência a um recurso [AWS::Serverless::API](#) definido em outro modelo.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ResponseTransferMode

O modo de transferência de resposta para a integração da função Lambda. Defina como RESPONSE_STREAM para habilitar o streaming de respostas do Lambda por meio do API Gateway, permitindo que a função transmita respostas de volta aos clientes. Quando definido como RESPONSE_STREAM, o API Gateway usa a API Lambda InvokeWithResponseStreaming .

Tipo: string

Obrigatório: não

Valores válidos: BUFFERED | RESPONSE_STREAM

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ResponseTransferMode](#) propriedade de um `AWS::ApiGateway::Method` Integration.

TimeoutInMillis

Tempo limite personalizado entre 50 e 29.000 milissegundos.

 Note

Quando você especifica essa propriedade, AWS SAM modifica sua definição de OpenAPI. A definição da OpenAPI deve ser especificada em linha usando a propriedade `DefinitionBody`.

Tipo: inteiro

Obrigatório: não

Padrão: 29.000 milissegundos ou 29 segundos

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Exemplo básico

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 a autorização no nível do evento, para uma API, um caminho e um método específicos.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
ApiKeyRequired: Boolean  
AuthorizationScopes: List  
Authorizer: String  
InvokeRole: String  
OverrideApiAuth: Boolean  
ResourcePolicy: ResourcePolicyStatement
```

Propriedades

ApiKeyRequired

Requer uma chave de API para essa API, caminho e método.

Tipo: booliano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AuthorizationScopes

Os escopos de autorização a serem aplicados a essa API, caminho e método.

Os escopos que você especificar substituirão quaisquer escopos aplicados pela propriedade `DefaultAuthorizer`, caso você a tenha especificado.

Tipo: Lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Authorizer

O `Authorizer` para uma função específica.

Se você tiver um autorizador global especificado para seu recurso `AWS::Serverless::Api`, poderá substituí-lo configurando como `Authorizer NONE`. Para ver um exemplo, consulte [Substitua um autorizador global para sua API REST do Amazon API Gateway](#).

 Note

Se você usar a propriedade `DefinitionBody` de um recurso `AWS::Serverless::Api` para descrever sua API, deverá usar o `OverrideApiAuth` com o `Authorizer` para substituir seu autorizador global. Consulte [OverrideApiAuth](#) para obter mais informações.

Valores válidos: `AWS_IAM`, `NONE`, ou a ID lógica de qualquer autorizador definido em seu AWS SAM modelo.

Tipo: `string`

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

InvokeRole

Especifica o `InvokeRole` a ser usado para autorização `AWS_IAM`.

Tipo: `string`

Obrigatório: não

Padrão: `CALLER_CREDENTIALS`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Notas adicionais: `CALLER_CREDENTIALS` mapeia para `arn:aws:iam::<user>/`, que usa as credenciais do chamador para invocar o endpoint.

OverrideApiAuth

Especifique como `true` para substituir a configuração global do autorizador do seu recurso `AWS::Serverless::Api`. Essa propriedade só é necessária se você especificar um autorizador global e usar a propriedade `DefinitionBody` de um recurso `AWS::Serverless::Api` para descrever sua API.

 Note

Quando você especificar `OverrideApiAuth` como `true`, AWS SAM substituirá seu autorizador global por quaisquer valores fornecidos para `ApiKeyRequiredAuthorizer`, ou `ResourcePolicy`. Portanto, pelo menos uma dessas propriedades também deve ser especificada durante o uso do `OverrideApiAuth`. Para ver um exemplo, consulte [Substitua um autorizador global quando for AWS::Serverless::Api DefinitionBody for especificado](#).

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ResourcePolicy

Configure a política de recursos para esse caminho em uma API.

Digite: [ResourcePolicyStatement](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Function-Auth

O exemplo a seguir especifica a autorização no nível da função.

YAML

```
Auth:
  ApiKeyRequired: true
  Authorizer: NONE
```

Substitua um autorizador global para sua API REST do Amazon API Gateway

Você pode especificar um autorizador global para seu recurso `AWS::Serverless::Api`. Este é um exemplo que configura um autorizador padrão global:

```
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
```

Para substituir o autorizador padrão da sua função de AWS Lambda evento, você pode especificar `Authorizer` como `NONE`. Este é um exemplo:

```
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
```

Substitua um autorizador global quando for `AWS::Serverless::Api` `DefinitionBody` for especificado

Ao usar a propriedade `DefinitionBody` para descrever seu recurso `AWS::Serverless::Api`, o método de substituição anterior não funciona. Veja a seguir um exemplo de uso da propriedade `DefinitionBody` para um recurso `AWS::Serverless::Api`:

```
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
```

Para substituir o autorizador global, use a propriedade `OverrideApiAuth`. Veja a seguir um exemplo que usa o `OverrideApiAuth` para substituir o autorizador global pelo valor fornecido para `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 uma política de recursos para todos os métodos e caminhos de uma API. Para obter mais informações sobre políticas de recursos, consulte Como [controlar o acesso a uma API com as políticas de recursos do API Gateway](#) no Guia do desenvolvedor do API Gateway.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```

AwsAccountBlacklist: List
AwsAccountWhitelist: List
CustomStatements: List

```

```
IntrinsicVpcBlacklist: List  
IntrinsicVpcWhitelist: List  
IntrinsicVpceBlacklist: List  
IntrinsicVpceWhitelist: List  
IpRangeBlacklist: List  
IpRangeWhitelist: List  
SourceVpcBlacklist: List  
SourceVpcWhitelist: List
```

Propriedades

AwsAccountBlacklist

As AWS contas a serem bloqueadas.

Tipo: lista de strings

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AwsAccountWhitelist

As AWS contas a serem permitidas. Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista de strings

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

CustomStatements

Uma lista de declarações de política de recursos personalizadas a serem aplicadas a essa API. Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntrinsicVpcBlacklist

A lista de nuvens privadas virtuais (VPCs) a serem bloqueadas, em que cada VPC é especificada como uma referência, como uma [referência dinâmica](#) ou a função Ref [intrínseca](#). Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntrinsicVpcWhitelist

A lista de VPCs permissões, em que cada VPC é especificada como uma referência, como uma [referência dinâmica](#) ou a função Ref [intrínseca](#).

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntrinsicVpceBlacklist

A lista de endpoints da VPC a serem bloqueados, em que cada endpoint da VPC é especificado como uma referência, como uma [referência dinâmica](#) ou a Ref [função intrínseca](#).

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntrinsicVpceWhitelist

A lista de endpoints da VPC a serem permitidos, em que cada endpoint da VPC é especificado como uma referência, como uma [referência dinâmica](#) ou a Ref [função intrínseca](#). Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IpRangeBlacklist

Os endereços IP ou intervalos de endereços a serem bloqueados. Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IpRangeWhitelist

Os endereços IP ou intervalos de endereços a serem permitidos.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SourceVpcBlacklist

A VPC de origem ou os endpoints da VPC a serem bloqueados. Os nomes da VPC de origem devem começar com "vpc-" e os nomes dos endpoints da VPC de origem devem começar com "vpce-". Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SourceVpcWhitelist

A VPC de origem ou os endpoints da VPC a serem permitidos. Os nomes da VPC de origem devem começar com "vpc-" e os nomes dos endpoints da VPC de origem devem começar com "vpce-".

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Exemplo de política de recursos

O exemplo a seguir bloqueia dois endereços IP e uma VPC de origem e permite uma AWS conta.

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 um modelo de solicitação para uma API+Path+Method específica.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Model: String  
Required: Boolean  
ValidateBody: Boolean  
ValidateParameters: Boolean
```

Propriedades

Model

Nome de um modelo definido na propriedade Models do [AWS::Serverless::API](#).

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Required

Adiciona uma required propriedade na seção de parâmetros da OpenApi definição para o determinado endpoint da API.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ValidateBody

Especifica se o API Gateway usa o `Model` para validar o corpo da solicitação. Para obter mais informações, consulte [Ativar validação de solicitação no API Gateway](#) no Guia do desenvolvedor do API Gateway.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ValidateParameters

Especifica se o API Gateway usa o `Model` para validar parâmetros do caminho da solicitação, cadeias de caracteres de consulta e cabeçalhos. Para obter mais informações, consulte [Ativar validação de solicitação no API Gateway](#) no Guia do desenvolvedor do API Gateway.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Modelo de solicitação

Exemplo de modelo de solicitação

YAML

```
RequestModel:
  Model: User
  Required: true
  ValidateBody: true
  ValidateParameters: true
```

RequestParameter

Configure o parâmetro de solicitação para uma API+Path+Method específica.

Uma `Required` ou a propriedade `Caching` precisa ser especificada para o parâmetro de solicitação

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Caching: Boolean  
Required: Boolean
```

Propriedades

Caching

Adiciona uma `cacheKeyParameters` seção à `OpenApi` definição do API Gateway

Tipo: booleano

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Required

Este campo especifica se um parâmetro é necessário

Tipo: booleano

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Parâmetro de solicitação

Exemplo de configuração de parâmetros de solicitação

YAML

```
RequestParameters:
  - method.request.header.Authorization:
      Required: true
      Caching: true
```

CloudWatchEvent

O objeto que descreve um tipo de fonte de evento CloudWatchEvent.

AWS Serverless Application Model (AWS SAM) gera um [AWS::Events::Rule](#) recurso quando esse tipo de evento é definido.

Nota importante: [EventBridgeRule](#) é o tipo de fonte de eventos preferido a ser usado, em vez de CloudWatchEvent. EventBridgeRule e CloudWatchEvent use o mesmo serviço, API e CloudFormation recursos subjacentes. No entanto, AWS SAM adicionará suporte para novos recursos somente para EventBridgeRule.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Enabled: Boolean
EventBusName: String
Input: String
InputPath: String
Pattern: EventPattern
State: String
```

Propriedades

Enabled

Indica se a regra está habilitada.

Para desativar a regra, defina essa propriedade como false.

 Note

Especifique a propriedade `Enabled` ou `State`, mas não ambas.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [State](#) propriedade de um `AWS::Events::Rule` recurso. Se essa propriedade for definida como `true` então AWS SAM passa `ENABLED`, caso contrário, ela passa `DISABLED`.

EventBusName

O barramento de eventos que deve ser associado a essa regra. Se você omitir essa propriedade, AWS SAM usará o barramento de eventos padrão.

Tipo: string

Obrigatório: não

Padrão: barramento de eventos padrão

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventBusName](#) propriedade de um `AWS::Events::Rule` recurso.

Input

Texto JSON válido passado para o destino. Se você usar essa propriedade, nada do próprio texto do evento é passado para o destino.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Input](#) propriedade de um `AWS::Events::Rule` `Target` recurso.

InputPath

Quando você não deseja passar todo o evento correspondente ao destino, a propriedade `InputPath` descreve qual parte do evento passar.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [InputPath](#) propriedade de um `AWS::Events::Rule` recurso.

Pattern

Descreve quais eventos são roteados para o destino especificado. Para obter mais informações, consulte [Eventos e padrões de eventos EventBridge no](#) Guia do EventBridge usuário da Amazon.

Digite: [EventPattern](#)

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventPattern](#) propriedade de um `AWS::Events::Rule` recurso.

State

O estado da regra.

Valores aceitos: DISABLED | ENABLED

Note

Especifique a propriedade Enabled ou State, mas não ambas.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [State](#) propriedade de um `AWS::Events::Rule` recurso.

Exemplos

CloudWatchEvent

O exemplo a seguir mostra o tipo de origem de um evento CloudWatchEvent.

YAML

```
CWEvent:
  Type: CloudWatchEvent
  Properties:
    Enabled: false
    Input: '{"Key": "Value"}'
    Pattern:
      detail:
        state:
          - running
```

CloudWatchLogs

O objeto que descreve um tipo de fonte de evento CloudWatchLogs.

O evento gera um recurso [AWS::Logs::SubscriptionFilter](#) e especifica um filtro de assinaturas e o associa ao grupo de logs especificado.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
FilterPattern: String
LogGroupName: String
```

Propriedades

FilterPattern

As expressões de filtragem que restringem o que é entregue ao AWS recurso de destino. Para obter mais informações sobre o filtro do padrão de filtro [Sintaxe de filtros e padrões](#).

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FilterPattern](#) propriedade de um `AWS::Logs::SubscriptionFilter` recurso.

LogGroupName

O grupo de logs ao qual associar o filtro de assinatura. Todos os eventos de log enviados para esse grupo de log são filtrados e entregues ao AWS recurso especificado se o padrão de filtro corresponder aos eventos de log.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [LogGroupName](#) propriedade de um `AWS::Logs::SubscriptionFilter` recurso.

Exemplos

Filtro de assinatura do Cloudwatchlogs

Exemplo de filtro de assinatura Cloudwatchlogs

YAML

```
CWLog:
  Type: CloudWatchLogs
  Properties:
    LogGroupName:
      Ref: CloudWatchLambdaLogsGroup
    FilterPattern: My pattern
```

Cognito

O objeto que descreve um tipo de fonte de evento Cognito.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Trigger: List
UserPool: String
```

Propriedades

Trigger

As informações de configuração de trigger do Lambda para o novo grupo de usuários.

Tipo: lista

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [LambdaConfig](#) propriedade de um `AWS::Cognito::UserPool` recurso.

UserPool

Referência a UserPool definida no mesmo modelo

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Evento Cognito

Exemplo de evento Cognito

YAML

```
CognitoUserPoolPreSignup:
  Type: Cognito
  Properties:
    UserPool:
      Ref: MyCognitoUserPool
    Trigger: PreSignUp
```

DocumentDB

O objeto que descreve um tipo de fonte de evento DocumentDB. Para obter mais informações, consulte Como [usar AWS Lambda com o Amazon DocumentDB](#) no Guia do AWS Lambda desenvolvedor.

Sintaxe

Para declarar essa entidade em seu AWS SAM modelo, use a sintaxe a seguir.

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
```

Propriedades

BatchSize

O número máximo de itens a serem recuperados em um único lote.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BatchSize](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Cluster

O nome de recurso da Amazon (ARN) do cluster do Amazon DocumentDB.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventSourceArn](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

CollectionName

O nome da coleção a ser consumida no banco de dados. Se você não especificar uma coleção, o Lambda consumirá todas as coleções.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CollectionName](#) propriedade de um tipo de `AWS::Lambda::EventSourceMapping` `DocumentDBEventSourceConfig` dados.

DatabaseName

O nome do banco de dados a ser usado no cluster do Amazon DocumentDB.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DatabaseName](#) propriedade de um tipo de `AWS::Lambda::EventSourceMapping` `DocumentDBEventSourceConfig` dados.

Enabled

Se `true`, o mapeamento da origem do evento estará ativo. Para pausar a sondagem e invocação, defina como `false`.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Enabled](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

FilterCriteria

Um objeto que define os critérios que determinam se o Lambda deve processar um evento. Para obter mais informações, consulte [Filtrando eventos do Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Digite: [FilterCriteria](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FilterCriteria](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

FullDocument

Determina o que o Amazon DocumentDB enviará para seu fluxo de eventos durante as operações de atualização de documentos. Se estiver configurado para `UpdateLookup`, o Amazon DocumentDB enviará um delta descrevendo as alterações, junto com uma cópia de todo o documento. Senão, o Amazon DocumentDB enviará somente um documento parcial contendo as alterações.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FullDocument](#) propriedade de um tipo de `AWS::Lambda::EventSourceMapping` `DocumentDBEventSourceConfig` dados.

KmsKeyArn

O nome do recurso da Amazon (ARN) da chave para criptografar informações relacionadas a esse evento.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [KmsKeyArn](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumBatchingWindowInSeconds

O máximo de tempo para reunir registros antes de invocar a função, em segundos.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumBatchingWindowInSeconds](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

SecretsManagerKmsKeyId

O ID da chave AWS Key Management Service (AWS KMS) de uma chave gerenciada pelo cliente do AWS Secrets Manager. Obrigatório quando você usa uma chave gerenciada pelo cliente do Secrets Manager com uma função de execução do Lambda que não inclui a permissão `kms:Decrypt`.

O valor da propriedade é um UUID. Por exemplo: `1abc23d4-567f-8ab9-cde0-1fab234c5d67`.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SourceAccessConfigurations

Uma matriz do protocolo de autenticação ou host virtual. Especifique isso usando o tipo de [SourceAccessConfigurations](#) dados.

Para o tipo de origem do evento DocumentDB, o único tipo de configuração válido é `BASIC_AUTH`.

- `BASIC_AUTH` – O segredo do Secrets Manager que armazena as credenciais do agente. Para esse tipo, a credencial deverá estar no seguinte formato: `{"username": "your-username", "password": "your-password"}`. Somente um objeto do tipo `BASIC_AUTH` é permitido.

Tipo: lista

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SourceAccessConfigurations](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

StartingPosition

A posição em um fluxo da qual você deseja iniciar a leitura.

- `AT_TIMESTAMP` – Especifique um tempo a partir do qual iniciar a leitura dos registros.
- `LATEST` – Leia somente registros novos.

- TRIM_HORIZON – Processe todos os registros disponíveis.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StartingPosition](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

StartingPositionTimestamp

O tempo a partir do qual iniciar a leitura, em segundos no horário do Unix. Defina `StartingPositionTimestamp` quando `StartingPosition` é especificado como `.AT_TIMESTAMP`

Tipo: duplo

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StartingPositionTimestamp](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Exemplos

Origem do evento do 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

O objeto que descreve um tipo de fonte de evento DynamoDB. Para obter mais informações, consulte Como [usar AWS Lambda com o Amazon DynamoDB](#) no AWS Lambda Guia do desenvolvedor.

AWS SAM gera um [AWS::Lambda::EventSourceMapping](#) recurso quando esse tipo de evento é definido.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
BatchSize: Integer
BisectBatchOnFunctionError: Boolean
DestinationConfig: DestinationConfig
Enabled: Boolean
FilterCriteria: FilterCriteria
FunctionResponseType: List
KmsKeyArn: String
MaximumBatchingWindowInSeconds: Integer
MaximumRecordAgeInSeconds: Integer
MaximumRetryAttempts: Integer
MetricsConfig: MetricsConfig
ParallelizationFactor: Integer
StartingPosition: String
StartingPositionTimestamp: Double
Stream: String
TumblingWindowInSeconds: Integer
```

Propriedades

BatchSize

O número máximo de itens a serem recuperados em um único lote.

Tipo: inteiro

Obrigatório: não

Padrão: 100

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BatchSize](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Mínimo: 1

Maximum: 1000

`BisectBatchOnFunctionError`

Se a função retornar um erro, divida o lote em dois e tente novamente.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BisectBatchOnFunctionError](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

`DestinationConfig`

Uma fila do Amazon Simple Queue Service (Amazon SQS) ou um destino do tópico do Amazon Simple Notification Service (Amazon SNS) para registros descartados.

Digite: [DestinationConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DestinationConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

`Enabled`

Desabilita o mapeamento de origens de eventos para pausar a sondagem e a invocação.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Enabled](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

FilterCriteria

Um objeto que define os critérios para determinar se o Lambda deve processar um evento. Para obter mais informações, consulte [Filtrando eventos do AWS Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Digite: [FilterCriteria](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FilterCriteria](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

FunctionResponseTypes

Uma lista de tipos de resposta atuais aplicados ao mapeamento da origem do evento. Para obter mais informações, consulte [Relatar falhas de itens em lote](#) no Guia do desenvolvedor do AWS Lambda .

Valores válidos: `ReportBatchItemFailures`

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FunctionResponseTypes](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

KmsKeyArn

O nome do recurso da Amazon (ARN) da chave para criptografar informações relacionadas a esse evento.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [KmsKeyArn](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumBatchingWindowInSeconds

O máximo de tempo para reunir registros antes de invocar a função, em segundos.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumBatchingWindowInSeconds](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumRecordAgeInSeconds

A idade máxima de um registro que o Lambda envia a uma função para processamento.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumRecordAgeInSeconds](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumRetryAttempts

O número máximo de vezes para tentar novamente quando a função retorna um erro.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumRetryAttempts](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MetricsConfig

Uma configuração opcional para obter métricas aprimoradas para os mapeamentos da origem do evento que capturam cada estágio do processamento. Para ver um exemplo, consulte [MetricsConfig evento](#).

Digite: [MetricsConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MetricsConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

ParallelizationFactor

O número de lotes a serem processados de cada fragmento simultaneamente.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ParallelizationFactor](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

StartingPosition

A posição em um fluxo da qual você deseja iniciar a leitura.

- `AT_TIMESTAMP` – Especifique um tempo a partir do qual iniciar a leitura dos registros.
- `LATEST` – Leia somente registros novos.
- `TRIM_HORIZON` – Processe todos os registros disponíveis.

Valores válidos: `AT_TIMESTAMP` | `LATEST` | `TRIM_HORIZON`

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StartingPosition](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

StartingPositionTimestamp

O tempo a partir do qual iniciar a leitura, em segundos no horário do Unix. Defina `StartingPositionTimestamp` quando `StartingPosition` é especificado como `.AT_TIMESTAMP`

Tipo: duplo

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StartingPositionTimestamp](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Stream

O nome de recurso da Amazon (ARN) do fluxo do DynamoDB.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventSourceArn](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

TumblingWindowInSeconds

A duração, em segundos, de uma janela de processamento. O intervalo válido é de 1 a 900 (15 minutos).

Para obter mais informações, consulte [Janelas caindo](#) no Guia do desenvolvedor do AWS Lambda .

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TumblingWindowInSeconds](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Exemplos

MetricsConfig evento

Veja a seguir um exemplo de um recurso que usa a propriedade `MetricsConfig` para capturar cada estágio de processamento para seus mapeamentos da origem do evento.

```
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 de eventos do DynamoDB para a tabela existente do DynamoDB

Fonte de eventos do DynamoDB para uma tabela do DynamoDB que já existe em uma conta. 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 do DynamoDB para tabela do DynamoDB declarado no modelo

Evento do DynamoDB para uma tabela do DynamoDB declarada no mesmo arquivo de modelo.

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

O objeto que descreve um tipo de fonte de EventBridgeRule evento, que define sua função sem servidor como o destino de uma regra da Amazon EventBridge . Para obter mais informações, consulte [O que é a Amazon EventBridge?](#) no Guia do EventBridge usuário da Amazon.

AWS SAM gera um `AWS::Events::Rule` recurso quando esse tipo de evento é definido. AWS SAM também cria um `AWS::Lambda::Permission` recurso, que é necessário para que eles EventBridgeRule possam chamar o Lambda.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
DeadLetterConfig: DeadLetterConfig
EventBusName: String
Input: String
InputPath: String
InputTransformer: InputTransformer
Pattern: EventPattern
RetryPolicy: RetryPolicy
RuleName: String
State: String
Target: Target
```

Propriedades

DeadLetterConfig

Configure a fila do Amazon Simple Queue Service (Amazon SQS) para a EventBridge qual envia eventos após uma falha na invocação de destino. A invocação pode falhar, por exemplo, ao enviar um evento para uma função de evento do Lambda que não existe ou quando EventBridge não há permissões suficientes para invocar a função do Lambda. Para obter mais informações, consulte [Política de repetição de eventos e uso de filas de mensagens sem saída no Guia do usuário](#) da Amazon. EventBridge

 Note

O tipo de recurso [AWS::Serverless::Função](#) tem um tipo de dados semelhante, `DeadLetterQueue`, que lida com falhas que ocorrem após a invocação bem-sucedida da função do Lambda de destino. Exemplos desses tipos de falhas incluem controle de utilização do Lambda ou erros retornados pela função de destino do Lambda. Para obter mais informações sobre a `DeadLetterQueue` propriedade da função, consulte [Dead-letter filas](#) no Guia do AWS Lambda desenvolvedor.

Digite: [DeadLetterConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [DeadLetterConfig](#) propriedade do tipo de `AWS::Events::Rule` dados. A AWS SAM versão dessa propriedade inclui subpropriedades adicionais, caso você queira criar AWS SAM a fila de mensagens mortas para você.

EventBusName

O barramento de eventos que deve ser associado a essa regra. Se você omitir essa propriedade, AWS SAM usará o barramento de eventos padrão.

Tipo: string

Obrigatório: não

Padrão: barramento de eventos padrão

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventBusName](#) propriedade de um `AWS::Events::Rule` recurso.

Input

Texto JSON válido passado para o destino. Se você usar essa propriedade, nada do próprio texto do evento é passado para o destino.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Input](#) propriedade de um `AWS::Events::Rule` Target recurso.

InputPath

Quando você não deseja passar todo o evento correspondente ao destino, a propriedade `InputPath` descreve qual parte do evento passar.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [InputPath](#) propriedade de um `AWS::Events::Rule` Target recurso.

InputTransformer

Configurações para permitir que você forneça entrada personalizada para um destino com base em determinados dados de evento. Você pode extrair um ou mais pares de valor-chave do evento e usar esses dados para enviar a entrada personalizada para o destino. Para obter mais informações, consulte [Transformação EventBridge de entrada](#) da Amazon no Guia EventBridge do usuário da Amazon.

Digite: [InputTransformer](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [InputTransformer](#) propriedade de um tipo de `AWS::Events::Rule` Target dados.

Pattern

Descreve quais eventos são roteados para o destino especificado. Para obter mais informações, consulte [EventBridgeEventos e padrões de EventBridge eventos da Amazon](#) no Guia EventBridge do usuário da Amazon.

Digite: [EventPattern](#)

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventPattern](#) propriedade de um `AWS::Events::Rule` recurso.

RetryPolicy

Um objeto RetryPolicy que inclui informações sobre as configurações de política de repetição. Para obter mais informações, consulte [Política de repetição de eventos e uso de filas de mensagens sem saída no Guia do usuário](#) da Amazon. EventBridge

Digite: [RetryPolicy](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RetryPolicy](#) propriedade do tipo de `AWS::Events::Rule` Target dados.

RuleName

O nome da regra.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::Events::Rule` recurso.

State

O estado da regra.

Valores aceitos: DISABLED | ENABLED |
ENABLED_WITH_ALL_CLOUDTRAIL_MANAGEMENT_EVENTS

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [State](#) propriedade de um `AWS::Events::Rule` recurso.

Target

O AWS recurso que é EventBridge invocado quando uma regra é acionada. Você pode usar essa propriedade para especificar a ID lógica do destino. Se essa propriedade não for especificada, a ID lógica do destino será AWS SAM gerada.

Tipo: [Target](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Targets](#) propriedade de um `AWS::Events::Rule` recurso. Amazon EC2 RebootInstances API call é um exemplo de uma propriedade de destino. A versão AWS SAM dessa propriedade só permite que você especifique a ID lógica de um único destino.

Exemplos

EventBridgeRule

O exemplo a seguir mostra o tipo de origem de um evento EventBridgeRule.

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

O objeto usado para especificar a fila do Amazon Simple Queue Service (Amazon SQS) para a EventBridge qual envia eventos após uma falha na invocação de destino. A invocação pode falhar, por exemplo, ao enviar um evento para uma função de evento Lambda que não existe ou permissões insuficientes para invocar a função Lambda. Para obter mais informações, consulte [Política de repetição de eventos e uso de filas de mensagens mortas no Guia do usuário](#) da Amazon.

EventBridge

 Note

O tipo de recurso [AWS::Serverless::Função](#) tem um tipo de dados semelhante, `DeadLetterQueue` que lida com falhas que ocorrem após a invocação bem-sucedida da função do Lambda de destino. Exemplos desse tipo de falha incluem controle de utilização do Lambda ou erros retornados pela função de destino do Lambda. Para obter mais informações sobre a `DeadLetterQueue` propriedade da função, consulte [Dead-letter filas](#) no Guia do AWS Lambda desenvolvedor.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Arn: String
QueueLogicalId: String
Type: String
```

Propriedades

Arn

O nome de recurso da Amazon (ARN) da fila Amazon SQS especificado como o destino para a fila de mensagens não entregues.

 Note

Especifique a propriedade `Type` ou a propriedade `Arn`, mas não ambas.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Arn](#) propriedade do tipo de `AWS::Events::Rule DeadLetterConfig` dados.

QueueLogicalId

O nome personalizado da fila de letras mortas que AWS SAM cria se Type for especificado.

Note

Se a propriedade Type não estiver definida, essa propriedade será ignorada.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Type

O tipo da fila. Quando essa propriedade é definida, cria AWS SAM automaticamente uma fila de mensagens mortas e anexa a [política baseada em recursos](#) necessária para conceder permissão ao recurso de regra para enviar eventos para a fila.

Note

Especifique a propriedade Type ou a propriedade Arn, mas não ambas.

Valores válidos: SQS

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

DeadLetterConfig

DeadLetterConfig

YAML

```
DeadLetterConfig:
  Type: SQS
  QueueLogicalId: MyDLQ
```

Target

Configura o AWS recurso que é EventBridge invocado quando uma regra é acionada.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Id: String
```

Propriedades

Id

O ID lógico do destino.

O valor do Id pode incluir caracteres alfanuméricos, pontos (.), hifens (-) e sublinhados (_).

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Id](#) propriedade do tipo de `AWS::Events::Rule` Target dados.

Exemplos

Target

YAML

```
EBRule:
```

```
Type: EventBridgeRule
Properties:
  Target:
    Id: MyTarget
```

HttpApi

O objeto que descreve uma fonte de eventos com tipo HttpApi.

Se houver uma OpenApi definição para o caminho e o método especificados na API, o SAM adicionará a seção de integração e segurança do Lambda (se aplicável) para você.

Se não existir uma OpenApi definição para o caminho e o método especificados na API, o SAM criará essa definição para você.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
ApiId: String
Auth: HttpApiFunctionAuth
Method: String
Path: String
PayloadFormatVersion: String
RouteSettings: RouteSettings
TimeoutInMillis: Integer
```

Propriedades

ApiId

Identificador de um recurso [AWS::Serverless::HttpApi](#) definido neste modelo.

Se não for definido, um [AWS::Serverless::HttpApi](#) recurso padrão é criado chamado `ServerlessHttpApi` usando um OpenApi documento gerado contendo uma união de todos os caminhos e métodos definidos pelos eventos da Api definidos neste modelo que não especificam um `ApiId`.

Isso não pode fazer referência a um recurso [AWS::Serverless::HttpApi](#) definido em outro modelo.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Auth

Configuração de autenticação para essa Api+Path+Method específica.

Útil para substituir as APIs `DefaultAuthorizer` ou definir a configuração de autenticação em um caminho individual quando não `DefaultAuthorizer` é especificado.

Digite: [HttpApiFunctionAuth](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Method

Método HTTP para o qual essa função é invocada.

Se não Path e Method for especificado, o SAM criará um caminho de API padrão que roteia qualquer solicitação que não seja mapeada para um endpoint diferente para essa função de evento do Lambda. Somente um desses caminhos padrão pode existir por API.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Path

Caminho Uri para o qual essa função é invocada. Deve começar com `/`.

Se não Path e Method for especificado, o SAM criará um caminho de API padrão que roteia qualquer solicitação que não seja mapeada para um endpoint diferente para essa função do Lambda. Somente um desses caminhos padrão pode existir por API.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

PayloadFormatVersion

Especifica o formato da carga enviada para uma integração.

OBSERVAÇÃO: PayloadFormatVersion requer que o SAM modifique sua definição de OpenAPI, portanto, ele só funciona com inline OpenApi definido na propriedade. DefinitionBody

Tipo: string

Obrigatório: não

Padrão: 2.0

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

RouteSettings

As configurações de rota por rota para essa API HTTP. Para obter mais informações sobre as configurações de rota, consulte [AWS::ApiGatewayV2::Stage RouteSettings no Guia](#) do desenvolvedor do API Gateway.

Nota: Se RouteSettings forem especificados no HttpApi recurso e na fonte do evento, AWS SAM mescla-os com as propriedades da fonte do evento que têm precedência.

Digite: [RouteSettings](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RouteSettings](#) propriedade de um AWS::ApiGatewayV2::Stage recurso.

TimeoutInMillis

Tempo limite personalizado entre 50 e 29.000 milissegundos.

OBSERVAÇÃO: TimeoutInMillis requer que o SAM modifique sua definição de OpenAPI, portanto, ele só funciona com inline OpenApi definido na propriedade. DefinitionBody

Tipo: inteiro

Obrigatório: não

Padrão: 5000

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

HttpApi Evento padrão

HttpApi Evento que usa o caminho padrão. Todos os caminhos e métodos não mapeados nessa API serão roteados para esse endpoint.

YAML

```
Events:
  HttpApiEvent:
    Type: HttpApi
```

HttpApi

HttpApi Evento que usa um caminho e um método específicos.

YAML

```
Events:
  HttpApiEvent:
    Type: HttpApi
    Properties:
      Path: /
      Method: GET
```

HttpApi Autorização

HttpApi Evento que usa um Autorizador.

YAML

```
Events:
```

```
HttpApiEvent:
  Type: HttpApi
  Properties:
    Path: /authenticated
    Method: GET
    Auth:
      Authorizer: OpenIdAuth
      AuthorizationScopes:
        - scope1
        - scope2
```

HttpApiFunctionAuth

Configura a autorização no nível do evento.

Configurar Auth para uma API + Path + Method específica

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
AuthorizationScopes: List
Authorizer: String
```

Propriedades

AuthorizationScopes

Os escopos de autorização a serem aplicados a essa API, caminho e método.

Os escopos listados aqui substituirão quaisquer escopos aplicados pelo, `DefaultAuthorizer` se existirem.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Authorizer

O Authorizer para uma função específica. Para usar a autorização do IAM, especifique `AWS_IAM` e especifique `true` for `EnableIamAuthorizer` na `Globals` seção do seu modelo.

Se você especificou um autorizador global na API e deseja tornar pública uma função específica, substitua configurando `Authorizer` como `NONE`.

Tipo: `string`

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Function-Auth

Especificando a autorização no nível da função

YAML

```
Auth:
  Authorizer: OpenIdAuth
  AuthorizationScopes:
    - scope1
    - scope2
```

Autorização do IAM

Especifica a autorização do IAM no nível do evento. Para usar a `AWS_IAM` autorização no nível do evento, você também deve especificar `true` for `EnableIamAuthorizer` na `Globals` seção do seu modelo. Para obter mais informações, consulte [Seção Global do AWS SAM modelo](#).

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

O objeto que descreve um tipo de origem do evento `IoTRule`.

Cria um [AWS::IoT::TopicRule](#) recurso para declarar uma AWS IoT regra. Para obter mais informações, consulte a [Documentação do CloudFormation](#)

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
AwsIotSqlVersion: String
Sql: String
```

Propriedades**AwsIotSqlVersion**

A versão do mecanismo de regras do SQL a ser usado ao avaliar a regra.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AwsIotSqlVersion](#) propriedade de um `AWS::IoT::TopicRule` `TopicRulePayload` recurso.

Sql

A instrução SQL usada para consultar o tópico. Para obter mais informações, consulte [Referência de SQL do AWS IoT](#) no Guia do desenvolvedor do AWS IoT .

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Sql](#) propriedade de um `AWS::IoT::TopicRule` `TopicRulePayload` recurso.

Exemplos

Regra da IOT

Exemplos de regras IOT

YAML

```
IoTRule:
  Type: IoTRule
  Properties:
    Sql: SELECT * FROM 'topic/test'
```

Kinesis

O objeto que descreve um tipo de fonte de evento Kinesis. Para obter mais informações, consulte Como [usar AWS Lambda com o Amazon Kinesis](#) no Guia do AWS Lambda desenvolvedor.

AWS SAM gera um [AWS::Lambda::EventSourceMapping](#) recurso quando esse tipo de evento é definido.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

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
```

Propriedades

BatchSize

O número máximo de itens a serem recuperados em um único lote.

Tipo: inteiro

Obrigatório: não

Padrão: 100

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BatchSize](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Mínimo: 1

Maximum: 10000

BisectBatchOnFunctionError

Se a função retornar um erro, divida o lote em dois e tente novamente.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BisectBatchOnFunctionError](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

DestinationConfig

Uma fila do Amazon Simple Queue Service (Amazon SQS) ou um destino do tópico do Amazon Simple Notification Service (Amazon SNS) para registros descartados.

Digite: [DestinationConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DestinationConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Enabled

Desabilita o mapeamento de origens de eventos para pausar a sondagem e a invocação.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Enabled](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

FilterCriteria

Um objeto que define os critérios para determinar se o Lambda deve processar um evento. Para obter mais informações, consulte [Filtrando eventos do AWS Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Digite: [FilterCriteria](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FilterCriteria](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

FunctionResponseTypes

Uma lista de tipos de resposta atuais aplicados ao mapeamento da origem do evento. Para obter mais informações, consulte [Relatar falhas de itens em lote](#) no Guia do desenvolvedor do AWS Lambda .

Valores válidos: `ReportBatchItemFailures`

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FunctionResponseTypes](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

KmsKeyArn

O nome do recurso da Amazon (ARN) da chave para criptografar informações relacionadas a esse evento.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [KmsKeyArn](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumBatchingWindowInSeconds

O máximo de tempo para reunir registros antes de invocar a função, em segundos.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumBatchingWindowInSeconds](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumRecordAgeInSeconds

A idade máxima de um registro que o Lambda envia a uma função para processamento.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumRecordAgeInSeconds](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumRetryAttempts

O número máximo de vezes para tentar novamente quando a função retorna um erro.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumRetryAttempts](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MetricsConfig

Uma configuração opcional para obter métricas aprimoradas para os mapeamentos da origem do evento que capturam cada estágio do processamento. Para ver um exemplo, consulte [MetricsConfig evento](#).

Digite: [MetricsConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MetricsConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

ParallelizationFactor

O número de lotes a serem processados de cada fragmento simultaneamente.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ParallelizationFactor](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

StartingPosition

A posição em um fluxo da qual você deseja iniciar a leitura.

- `AT_TIMESTAMP` – Especifique um tempo a partir do qual iniciar a leitura dos registros.
- `LATEST` – Leia somente registros novos.
- `TRIM_HORIZON` – Processe todos os registros disponíveis.

Valores válidos: AT_TIMESTAMP | LATEST | TRIM_HORIZON

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StartingPosition](#) propriedade de um AWS::Lambda::EventSourceMapping recurso.

StartingPositionTimestamp

O tempo a partir do qual iniciar a leitura, em segundos no horário do Unix. Defina StartingPositionTimestamp quando StartingPosition é especificado como .AT_TIMESTAMP

Tipo: duplo

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StartingPositionTimestamp](#) propriedade de um AWS::Lambda::EventSourceMapping recurso.

Stream

O nome de recurso da Amazon (ARN) do fluxo de dados ou um consumidor de fluxo.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventSourceArn](#) propriedade de um AWS::Lambda::EventSourceMapping recurso.

TumblingWindowInSeconds

A duração, em segundos, de uma janela de processamento. O intervalo válido é de 1 a 900 (15 minutos).

Para obter mais informações, consulte [Janelas caindo](#) no Guia do desenvolvedor do AWS Lambda .

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TumblingWindowInSeconds](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Exemplos

MetricsConfig evento

Veja a seguir um exemplo de um recurso que usa a propriedade `MetricsConfig` para capturar cada estágio de processamento para seus mapeamentos da origem do evento.

```
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 do evento do Kinesis

Veja a seguir um exemplo de uma fonte de eventos do 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

O objeto que descreve um tipo de fonte de evento MQ. Para obter mais informações, consulte [Uso do Lambda com o Amazon MQ](#) no Guia do desenvolvedor do AWS Lambda .

AWS Serverless Application Model (AWS SAM) gera um [AWS::Lambda::EventSourceMapping](#) recurso quando esse tipo de evento é definido.

Note

Para ter uma fila do Amazon MQ em uma nuvem privada virtual (VPC) que se conecta a uma função de evento Lambda em uma rede pública, a função de execução da sua função deve incluir as seguintes permissões:

- `ec2:CreateNetworkInterface`
- `ec2>DeleteNetworkInterface`
- `ec2:DescribeNetworkInterfaces`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeSubnets`
- `ec2:DescribeVpcs`

Para obter mais informações, consulte [Permissões de função de execução](#) no Guia do desenvolvedor do AWS Lambda .

Sintaxe

Para declarar essa entidade em seu AWS SAM modelo, use a sintaxe a seguir.

YAML

```
BatchSize: Integer
Broker: String
DynamicPolicyName: Boolean
Enabled: Boolean
```

FilterCriteria: [FilterCriteria](#)
KmsKeyArn: *String*
MaximumBatchingWindowInSeconds: *Integer*
Queues: *List*
SecretsManagerKmsKeyId: *String*
SourceAccessConfigurations: *List*

Propriedades

BatchSize

O número máximo de itens a serem recuperados em um único lote.

Tipo: inteiro

Obrigatório: não

Padrão: 100

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BatchSize](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Mínimo: 1

Maximum: 10000

Broker

O nome de recurso da Amazon (ARN) do agente do Amazon MQ.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventSourceArn](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

DynamicPolicyName

Por padrão, o nome da política AWS Identity and Access Management (IAM) é `SamAutoGeneratedAMQPolicy` para compatibilidade com versões anteriores. Especifique `true` o uso de um nome gerado automaticamente para sua política do IAM. Esse nome incluirá o ID lógico da fonte de eventos do Amazon MQ.

 Note

Ao usar mais de uma fonte de eventos do Amazon MQ, especifique `true` para evitar nomes duplicados de políticas do IAM.

Tipo: `booleano`

Obrigatório: `não`

Padrão: `false`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Enabled

Se `true`, o mapeamento da origem do evento estará ativo. Para pausar a sondagem e invocação, defina como `false`.

Tipo: `booleano`

Obrigatório: `não`

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Enabled](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

FilterCriteria

Um objeto que define os critérios que determinam se o Lambda deve processar um evento. Para obter mais informações, consulte [Filtrando eventos do AWS Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Digite: [FilterCriteria](#)

Obrigatório: `não`

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FilterCriteria](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

KmsKeyArn

O nome do recurso da Amazon (ARN) da chave para criptografar informações relacionadas a esse evento.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [KmsKeyArn](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumBatchingWindowInSeconds

O máximo de tempo para reunir registros antes de invocar a função, em segundos.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumBatchingWindowInSeconds](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Queues

O nome da fila de destino do agente do Amazon MQ a ser consumido.

Tipo: lista

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Queues](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

SecretsManagerKmsKeyId

O ID da chave AWS Key Management Service (AWS KMS) de uma chave gerenciada pelo cliente de AWS Secrets Manager. Obrigatório quando você usa uma chave gerenciada pelo cliente do Secrets Manager com uma função de execução do Lambda que não inclui a `kms:Decrypt` permissão.

O valor padrão da propriedade é um UUID. Por exemplo: `1abc23d4-567f-8ab9-cde0-1fab234c5d67`.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SourceAccessConfigurations

Uma matriz do protocolo de autenticação ou host virtual. Especifique isso usando o tipo de [SourceAccessConfigurations](#) dados.

Para o tipo de fonte do MQ evento, os únicos tipos de configuração válidos são BASIC_AUTH e VIRTUAL_HOST.

- **BASIC_AUTH** – O segredo do Secrets Manager que armazena as credenciais do corretor. Para esse tipo, a credencial deverá estar no seguinte formato: {"username": "your-username", "password": "your-password"}. Somente um objeto do tipo BASIC_AUTH é permitido.
- **VIRTUAL_HOST** – O nome do host virtual no seu agente do RabbitMQ. O Lambda usará esse host Rabbit MQ como fonte de eventos. Somente um objeto do tipo VIRTUAL_HOST é permitido.

Tipo: lista

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SourceAccessConfigurations](#) propriedade de um AWS::Lambda::EventSourceMapping recurso.

Exemplos

Fonte do evento do Amazon MQ

Veja a seguir um exemplo de um tipo de fonte de MQ evento para um agente do Amazon MQ.

YAML

```
Events:
  MQEvent:
    Type: MQ
    Properties:
      Broker: arn:aws:mq:us-east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k17819
      Queues: List of queues
```

SourceAccessConfigurations:

- Type: BASIC_AUTH

URI: arn:aws:secretsmanager:us-east-1:01234567890:secret:MyBrokerSecretName

BatchSize: 200

Enabled: true

MSK

O objeto que descreve um tipo de origem do evento MSK. Para obter mais informações, consulte Como [usar AWS Lambda com o Amazon MSK](#) no Guia do AWS Lambda desenvolvedor.

AWS Serverless Application Model (AWS SAM) gera um [AWS::Lambda::EventSourceMapping](#) recurso quando esse tipo de evento é definido.

Para usar o Schema Registry, você precisa definir permissões específicas do perfil do IAM para sua função. Consulte [Configuração completa com perfis do IAM](#) para ver um exemplo da configuração necessária.

Sintaxe

Para declarar essa entidade em seu AWS SAM modelo, use a sintaxe a seguir.

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*

Propriedades

BatchSize

O número máximo de registros em cada batch que o Lambda extrai da sua transmissão ou fila e envia para sua função. O Lambda transmite todos os registros no batch para a função em uma única chamada até o limite de carga útil para invocação síncrona (6 MB).

Padrão: 100

Faixa válida: valor mínimo de 1. Valor máximo de 10.000.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BatchSize](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

BisectBatchOnFunctionError

Se a função retornar um erro, divida o lote em dois e tente novamente.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BisectBatchOnFunctionError](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

ConsumerGroupId

Uma string que configura como os eventos serão lidos nos tópicos do Kafka.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AmazonManagedKafkaConfiguration](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

DestinationConfig

Um objeto de configuração que especifica o destino de um evento depois que o Lambda processá-lo.

Use essa propriedade para especificar o destino de invocações com falha da fonte de eventos do Amazon MSK.

Digite: [DestinationConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DestinationConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Enabled

Desabilita o mapeamento de origens de eventos para pausar a sondagem e a invocação.

Tipo: booliano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Enabled](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

FilterCriteria

Um objeto que define os critérios que determinam se o Lambda deve processar um evento. Para obter mais informações, consulte [Filtrando eventos do AWS Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Digite: [FilterCriteria](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FilterCriteria](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

FunctionResponseTypes

Uma lista de tipos de resposta atuais aplicados ao mapeamento da origem do evento. Para obter mais informações, consulte [Relatar falhas de itens em lote](#) no Guia do desenvolvedor do AWS Lambda .

Valores válidos: `ReportBatchItemFailures`

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FunctionResponseTypes](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

KmsKeyArn

O nome do recurso da Amazon (ARN) da chave para criptografar informações relacionadas a esse evento.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [KmsKeyArn](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumBatchingWindowInSeconds

O máximo de tempo para reunir registros antes de invocar a função, em segundos.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumBatchingWindowInSeconds](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumRecordAgeInSeconds

A idade máxima de um registro que o Lambda envia a uma função para processamento.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumRecordAgeInSeconds](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumRetryAttempts

O número máximo de vezes para tentar novamente quando a função retorna um erro.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumRetryAttempts](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

LoggingConfig

Um objeto de configuração que especifica a configuração de registro para o mapeamento da origem do evento.

Digite: [LoggingConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [LoggingConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MetricsConfig

Um objeto de configuração que especifica a configuração de métricas para o mapeamento da origem do evento.

Digite: [MetricsConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MetricsConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

ProvisionedPollerConfig

Configuração para aumentar a quantidade de sondagens usados para computar os mapeamentos da origem do evento. Essa configuração permite um mínimo de 1 poller e um máximo de 2.000 pollers. Para obter um exemplo, consulte [ProvisionedPollerConfig exemplo](#).

Digite: [ProvisionedPollerConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ProvisionedPollerConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

SchemaRegistryConfig

Configuração para usar um registro de esquema com a origem de eventos do Kafka.

Note

Esse recurso `ProvisionedPollerConfig` precisa ser configurado.

Tipo: `SchemaRegistryConfig`

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AmazonManagedKafkaEventSourceConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

SourceAccessConfigurations

Uma matriz do protocolo de autenticação, os componentes da VPC ou o host virtual para proteger e definir a fonte de eventos.

Valores válidos: `CLIENT_CERTIFICATE_TLS_AUTH`

Tipo: lista de [SourceAccessConfiguration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade faz parte da [AmazonManagedKafkaEventSourceConfig](#) propriedade de um recurso. `AWS::Lambda::EventSourceMapping`

StartingPosition

A posição em um fluxo da qual você deseja iniciar a leitura.

- `AT_TIMESTAMP` – Especifique um tempo a partir do qual iniciar a leitura dos registros.
- `LATEST` – Leia somente registros novos.
- `TRIM_HORIZON` – Processe todos os registros disponíveis.

Valores válidos: AT_TIMESTAMP | LATEST | TRIM_HORIZON

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StartingPosition](#) propriedade de um AWS::Lambda::EventSourceMapping recurso.

StartingPositionTimestamp

O tempo a partir do qual iniciar a leitura, em segundos no horário do Unix. Defina StartingPositionTimestamp quando StartingPosition é especificado como .AT_TIMESTAMP

Tipo: duplo

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StartingPositionTimestamp](#) propriedade de um AWS::Lambda::EventSourceMapping recurso.

Stream

O nome de recurso da Amazon (ARN) do fluxo de dados ou um consumidor de fluxo.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventSourceArn](#) propriedade de um AWS::Lambda::EventSourceMapping recurso.

Topics

O nome do tópico do Kafka.

Tipo: lista

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Topics](#) propriedade de um AWS::Lambda::EventSourceMapping recurso.

Exemplos

Configuração completa com perfis do IAM

O exemplo a seguir mostra uma configuração completa, incluindo a configuração do perfil do IAM necessário para usar o 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:
        EBSSStorageInfo:
          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 exemplo

```

ProvisionedPollerConfig:
  MinimumPollers: 1
  MaximumPollers: 200

```

Exemplo do Amazon MSK para cluster existente

Veja a seguir um exemplo de um tipo de fonte de MSK evento para um cluster Amazon MSK que já existe em uma Conta da 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
```

Exemplo do Amazon MSK para cluster declarado no mesmo modelo

Veja a seguir um exemplo de um tipo de fonte de MSK evento para um cluster Amazon MSK declarado no mesmo arquivo de modelo.

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
```

Origem de eventos MSK com o Schema Registry

Veja a seguir um exemplo de um tipo de origem de evento do MSK configurado com o esquema de registro.

```
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

```

Origem de eventos MSK com o Confluent Schema Registry

Veja a seguir um exemplo de um tipo de origem de eventos do MSK configurado com o 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

O objeto que descreve um tipo de fonte de evento de S3.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Bucket: String  
Events: String | List  
Filter: NotificationFilter
```

Propriedades

Bucket

O nome do bucket do S3. Esse bucket precisa existir no mesmo modelo.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é semelhante à [BucketName](#) propriedade de um `AWS::S3::Bucket` recurso. Este é um campo obrigatório no SAM. Esse campo aceita somente uma referência ao bucket do S3 criado neste modelo

Events

O evento de bucket do Amazon S3 para o qual invocar a função de evento Lambda. Consulte os [tipos de eventos compatíveis com o Amazon S3](#) para obter uma lista de valores válidos.

Tipo: String | List

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Event](#) propriedade do tipo de `AWS::S3::Bucket LambdaConfiguration` dados.

Filter

As regras de filtragem que determinam quais objetos do Amazon S3 invocam a função do Lambda. Para obter informações sobre a filtragem de nome de chave do Amazon S3, consulte [Configurar notificações de Amazon S3 Event](#) no Guia do desenvolvedor do Amazon Simple Storage Service.

Digite: [NotificationFilter](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Filter](#) propriedade do tipo de AWS::S3::Bucket LambdaConfiguration dados.

Exemplos

S3-Event

Exemplo de um evento do 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
```

Agendamento

O objeto que descreve um tipo de fonte de Schedule evento, que define sua função sem servidor como o destino de uma EventBridge regra da Amazon que é acionada em um cronograma. Para obter mais informações, consulte [O que é a Amazon EventBridge?](#) no Guia do EventBridge usuário da Amazon.

AWS Serverless Application Model (AWS SAM) gera um [AWS::Events::Rule](#) recurso quando esse tipo de evento é definido.

Note

EventBridge agora oferece um novo recurso de agendamento, a [Amazon EventBridge Scheduler](#). EventBridge SchedulerA Amazon é um programador sem servidor que permite criar, executar e gerenciar tarefas a partir de um serviço gerenciado central. EventBridge

Scheduler é altamente personalizável e oferece escalabilidade aprimorada em relação às regras EventBridge programadas, com um conjunto mais amplo de operações de API de destino e Serviços da AWS

Recomendamos que você use o EventBridge Scheduler para invocar destinos em uma programação. Para definir esse tipo de fonte de evento em seus AWS SAM modelos, consulte [ScheduleV2](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
DeadLetterConfig: DeadLetterConfig
Description: String
Enabled: Boolean
Input: String
Name: String
RetryPolicy: RetryPolicy
Schedule: String
State: String
```

Propriedades

DeadLetterConfig

Configure a fila do Amazon Simple Queue Service (Amazon SQS) para a EventBridge que envia eventos após uma falha na invocação de destino. A invocação pode falhar, por exemplo, ao enviar um evento para uma função de evento do Lambda que não existe ou quando EventBridge não há permissões suficientes para invocar a função do Lambda. Para obter mais informações, consulte [Política de repetição de eventos e uso de filas de mensagens sem saída no Guia do usuário](#) da Amazon. EventBridge

Note

O tipo de recurso [AWS::Serverless::Função](#) tem um tipo de dados semelhante, DeadLetterQueue, que lida com falhas que ocorrem após a invocação bem-sucedida da função do Lambda de destino. Exemplos desses tipos de falhas incluem controle de

utilização do Lambda ou erros retornados pela função de destino do Lambda. Para obter mais informações sobre a `DeadLetterQueue` propriedade da função, consulte [Dead-letter filas](#) no Guia do AWS Lambda desenvolvedor.

Digite: [DeadLetterConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [DeadLetterConfig](#) propriedade do tipo de `AWS::Events::Rule` Target dados. A AWS SAM versão dessa propriedade inclui subpropriedades adicionais, caso você queira criar AWS SAM a fila de mensagens mortas para você.

Description

Uma descrição da regra.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::Events::Rule` recurso.

Enabled

Indica se a regra está habilitada.

Para desativar a regra, defina essa propriedade como `false`.

Note

Especifique a propriedade `Enabled` ou `State`, mas não ambas.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [State](#) propriedade de um `AWS::Events::Rule` recurso. Se essa propriedade for definida como `true` então AWS SAM passa `ENABLED`, caso contrário, ela passa `DISABLED`.

Input

Texto JSON válido passado para o destino. Se você usar essa propriedade, nada do próprio texto do evento é passado para o destino.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Input](#) propriedade de um `AWS::Events::Rule` recurso.

Name

O nome da regra. Se você não especificar um nome, CloudFormation gera uma ID física exclusiva e usa essa ID para o nome da regra.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::Events::Rule` recurso.

RetryPolicy

Um objeto `RetryPolicy` que inclui informações sobre as configurações de política de repetição. Para obter mais informações, consulte [Política de repetição de eventos e uso de filas de mensagens sem saída no Guia do usuário](#) da Amazon. EventBridge

Digite: [RetryPolicy](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RetryPolicy](#) propriedade do tipo de `AWS::Events::Rule` Target dados.

Schedule

A expressão de programação que determina quando e com que frequência a regra é executada. Para obter mais informações, consulte [Programar expressões para regras](#).

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ScheduleExpression](#) propriedade de um `AWS::Events::Rule` recurso.

State

O estado da regra.

Valores aceitos: DISABLED | ENABLED

Note

Especifique a propriedade Enabled ou State, mas não ambas.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [State](#) propriedade de um `AWS::Events::Rule` recurso.

Exemplos

CloudWatch Agende um evento

CloudWatch Exemplo de agendamento de evento

YAML

```
CWSchedule:
  Type: Schedule
  Properties:
    Schedule: 'rate(1 minute)'
    Name: TestSchedule
    Description: test schedule
    Enabled: false
```

DeadLetterConfig

O objeto usado para especificar a fila do Amazon Simple Queue Service (Amazon SQS) para a EventBridge qual envia eventos após uma falha na invocação de destino. A invocação pode

falhar, por exemplo, ao enviar um evento para uma função de evento Lambda que não existe ou permissões insuficientes para invocar a função Lambda. Para obter mais informações, consulte [Política de repetição de eventos e uso de filas de mensagens mortas no Guia do usuário](#) da Amazon EventBridge

Note

O tipo de recurso [AWS::Serverless::Função](#) tem um tipo de dados semelhante, `DeadLetterQueue` que lida com falhas que ocorrem após a invocação bem-sucedida da função do Lambda de destino. Exemplos desse tipo de falha incluem controle de utilização do Lambda ou erros retornados pela função de destino do Lambda. Para obter mais informações sobre a `DeadLetterQueue` propriedade da função, consulte [Dead-letter filas](#) no Guia do AWS Lambda desenvolvedor.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Arn: String
QueueLogicalId: String
Type: String
```

Propriedades

Arn

O nome de recurso da Amazon (ARN) da fila Amazon SQS especificado como o destino para a fila de mensagens não entregues.

Note

Especifique a propriedade `Type` ou a propriedade `Arn`, mas não ambas.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Arn](#) propriedade do tipo de `AWS::Events::Rule` `DeadLetterConfig` dados.

`QueueLogicalId`

O nome personalizado da fila de letras mortas que AWS SAM cria se `Type` for especificado.

 Note

Se a propriedade `Type` não estiver definida, essa propriedade será ignorada.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

`Type`

O tipo da fila. Quando essa propriedade é definida, cria AWS SAM automaticamente uma fila de mensagens mortas e anexa a [política baseada em recursos](#) necessária para conceder permissão ao recurso de regra para enviar eventos para a fila.

 Note

Especifique a propriedade `Type` ou a propriedade `Arn`, mas não ambas.

Valores válidos: SQS

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

DeadLetterConfig

DeadLetterConfig

YAML

```
DeadLetterConfig:
  Type: SQS
  QueueLogicalId: MyDLQ
```

ScheduleV2

O objeto que descreve um tipo de fonte de `ScheduleV2` evento, que define sua função sem servidor como o destino de um evento do Amazon EventBridge Scheduler que é acionado em uma programação. Para obter mais informações, consulte [O que é o Amazon EventBridge Scheduler?](#) no Guia do usuário do EventBridge Scheduler.

AWS Serverless Application Model (AWS SAM) gera um [AWS::Scheduler::Scheduler](#) recurso quando esse tipo de evento é definido.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

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
```

Propriedades

DeadLetterConfig

Configure a fila do Amazon Simple Queue Service (Amazon SQS) para a EventBridge qual envia eventos após uma falha na invocação de destino. A invocação pode falhar, por exemplo, ao enviar um evento para uma função de evento Lambda que não existe ou quando não há permissões suficientes para invocar EventBridge a função Lambda. Para obter mais informações, consulte [Configurando uma fila de mensagens mortas para o EventBridge Scheduler no Guia do usuário do Scheduler](#). EventBridge

Note

O tipo de recurso [AWS::Serverless::Função](#) tem um tipo de dados semelhante, `DeadLetterQueue`, que lida com falhas que ocorrem após a invocação bem-sucedida da função do Lambda de destino. Exemplos desses tipos de falhas incluem controle de utilização do Lambda ou erros retornados pela função de destino do Lambda. Para obter mais informações sobre a `DeadLetterQueue` propriedade da função, consulte [Dead-letter filas](#) no Guia do AWS Lambda desenvolvedor.

Digite: [DeadLetterConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [DeadLetterConfig](#) propriedade do tipo de `AWS::Scheduler::Schedule Target` dados. A AWS SAM versão dessa propriedade inclui subpropriedades adicionais, caso você queira criar AWS SAM a fila de mensagens mortas para você.

Description

Uma descrição da agenda.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

EndDate

A data, em UTC, até a qual a agenda pode invocar seu destino. Dependendo da expressão de recorrência da agenda, as invocações podem ser interrompidas até a `EndDate` que você especifica.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EndDate](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

FlexibleTimeWindow

Permite a configuração de uma janela na qual uma agenda pode ser invocada.

Digite: [FlexibleTimeWindow](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FlexibleTimeWindow](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

GroupName

O nome do grupo de agendas para associar a essa agenda. Se não for definido, o grupo padrão será usado.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [GroupName](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

Input

Texto JSON válido passado para o destino. Se você usar essa propriedade, nada do próprio texto do evento é passado para o destino.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Input](#) propriedade de um `AWS::Scheduler::Schedule` Target recurso.

KmsKeyArn

O ARN de uma chave KMS será usada para criptografar dados do cliente.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [KmsKeyArn](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

Name

O nome da programação. Se você não especificar um nome, AWS SAM gera um nome no formato *Function-Logical-IDEvent-Source-Name* e usa essa ID para o nome da agenda.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

OmitName

Por padrão, AWS SAM gera e usa um nome de agendamento no formato de *<Function-logical-ID><event-source-name>*. Defina essa propriedade `true` para CloudFormation gerar uma ID física exclusiva e, em vez disso, use-a como nome da programação.

Tipo: booleano

Obrigatório: não

Padrão: `false`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

PermissionsBoundary

O ARN da política usada para definir o limite de permissões para a função.

Note

Se `PermissionsBoundary` estiver definido, AWS SAM aplicará os mesmos limites à função IAM de destino da agenda do agendador.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [PermissionsBoundary](#) propriedade de um `AWS::IAM::Role` recurso.

RetryPolicy

Um objeto `RetryPolicy` que inclui informações sobre as configurações de política de repetição.

Digite: [RetryPolicy](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RetryPolicy](#) propriedade do tipo de `AWS::Scheduler::Schedule Target` dados.

RoleArn

O ARN da função do IAM que o EventBridge Scheduler usará para o destino quando o agendamento for invocado.

Digite: [RoleArn](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RoleArn](#) propriedade do tipo de `AWS::Scheduler::Schedule Target` dados.

ScheduleExpression

A expressão de programação que determina quando e com que frequência o evento de agendamento do agendador é executado.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ScheduleExpression](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

ScheduleExpressionTimezone

O fuso horário no qual a expressão de agendamento é avaliada.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ScheduleExpressionTimezone](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

StartDate

A data, em UTC, a partir da qual a agenda pode começar a invocar um destino. Dependendo da expressão de recorrência da agenda, as invocações podem ocorrer a partir da `StartDate` que você especifica.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StartDate](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

State

O estado da agenda do programador.

Valores aceitos: DISABLED | ENABLED

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [State](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

Exemplos

Exemplo básico de definição de um recurso 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

O ID físico gerado do ScheduleV2 não inclui o nome da pilha.

SelfManagedKafka

O objeto que descreve um tipo de fonte de evento SelfManagedKafka. Para obter mais informações, consulte [Usando AWS Lambda com o Apache Kafka autogerenciado](#) no Guia do desenvolvedor.AWS Lambda

AWS Serverless Application Model (AWS SAM) gera um [AWS::Lambda::EventSourceMapping](#) recurso quando esse tipo de evento é definido.

Para usar o Schema Registry, você precisa definir permissões específicas do perfil do IAM para sua função. Consulte [Configuração completa com perfis do IAM](#) para ver um exemplo da configuração necessária.

Sintaxe

Para declarar essa entidade em seu AWS SAM modelo, use a sintaxe a seguir.

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
```

Propriedades

BatchSize

O número máximo de registros em cada batch que o Lambda extrai da sua transmissão e envia para sua função.

Tipo: inteiro

Obrigatório: não

Padrão: 100

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BatchSize](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Mínimo: 1

Maximum: 10000

`BisectBatchOnFunctionError`

Se a função retornar um erro, divida o lote em dois e tente novamente.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BisectBatchOnFunctionError](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

`ConsumerGroupId`

Uma string que configura como os eventos serão lidos nos tópicos do Kafka.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SelfManagedKafkaEventSourceConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

`DestinationConfig`

Um objeto de configuração que especifica o destino de um evento depois que o Lambda processá-lo.

Use essa propriedade para especificar o destino de invocações com falha da fonte de eventos auto gerenciada do Kafka.

Digite: [DestinationConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DestinationConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Enabled

Desabilita o mapeamento de origens de eventos para pausar a sondagem e a invocação.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Enabled](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

FilterCriteria

Um objeto que define os critérios para determinar se o Lambda deve processar um evento. Para obter mais informações, consulte [Filtrando eventos do AWS Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Digite: [FilterCriteria](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FilterCriteria](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

KafkaBootstrapServers

A lista de servidores de bootstrap para seus corretores Kafka. Inclua a porta, por exemplo `broker.example.com:xxxx`

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

FunctionResponseTypes

Uma lista de tipos de resposta atuais aplicados ao mapeamento da origem do evento. Para obter mais informações, consulte [Relatar falhas de itens em lote](#) no Guia do desenvolvedor do AWS Lambda .

Valores válidos: `ReportBatchItemFailures`

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FunctionResponseTypes](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

KmsKeyArn

O nome do recurso da Amazon (ARN) da chave para criptografar informações relacionadas a esse evento.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [KmsKeyArn](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

LoggingConfig

A configuração de registro para sua fonte de eventos.

Digite: [LoggingConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [LoggingConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumRecordAgeInSeconds

A idade máxima de um registro que o Lambda envia a uma função para processamento.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumRecordAgeInSeconds](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MetricsConfig

A configuração de métricas para sua fonte de eventos.

Digite: [MetricsConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MetricsConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumRetryAttempts

O número máximo de vezes para tentar novamente quando a função retorna um erro.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumRetryAttempts](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

ProvisionedPollerConfig

Configuração para aumentar a quantidade de sondagens usados para computar os mapeamentos da origem do evento. Essa configuração permite um mínimo de 1 poller e um máximo de 2.000 pollers. Para obter um exemplo, consulte [ProvisionedPollerConfig exemplo](#)

Digite: [ProvisionedPollerConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ProvisionedPollerConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

SchemaRegistryConfig

Configuração para usar um registro de esquema com a origem de eventos autogerenciada do Kafka.

Note

Esse recurso `ProvisionedPollerConfig` precisa ser configurado.

Digite: [SchemaRegistryConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SelfManagedKafkaEventSourceConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

SourceAccessConfigurations

Uma variedade de protocolos de autenticação, componentes de VPC e metadados do OAuth 2.0 que protegem e definem sua fonte de eventos. Cada entrada é um objeto com um `Type` campo e, para a maioria dos tipos, um `URI` campo. Para ver a lista completa dos tipos e o `URI` valor de cada um, consulte [SourceAccessConfiguration](#) no Guia AWS CloudFormation do usuário.

AWS SAM aplica a seguinte validação antes de passar essa propriedade para o Lambda:

- Você não deve especificar mais do que um tipo de autenticação. AWS SAM retornará um erro se você especificar mais de um.
- Você deve especificar `VPC_SUBNET` e `VPC_SECURITY_GROUP` em conjunto. AWS SAM retorna um erro se você especificar um sem o outro.
- Você deve especificar um `URI` valor para cada tipo, exceto `IAM_AUTH` e `IAM_OAUTHBEARER_AUTH`, que aceita `nonURI`. AWS SAM verifica somente se o valor está presente e o transmite sem interpretá-lo. O Lambda valida o valor e a combinação de tipos quando você implanta.

AWS SAM adiciona `secretsmanager:GetSecretValue` permissão para cada ARN secreto que você especificar, incluindo o `SERVER_ROOT_CA_CERTIFICATE` segredo e as permissões da interface de rede do Amazon EC2 que o Lambda precisa quando você especifica entradas de VPC. Para `IAM_AUTH` e `IAM_OAUTHBEARER_AUTH`, não AWS SAM adiciona permissões de cluster ou token. Conceda você mesmo essas permissões à função de execução da sua função. Quando nenhuma delas se aplica, como `IAM_AUTH` sem entradas de VPC, não AWS SAM adiciona nenhuma política embutida à função de execução. Para obter as permissões necessárias, consulte [Como configurar métodos de autenticação de cluster no Lambda](#) no Guia AWS Lambda do desenvolvedor.

Valores válidos: `BASIC_AUTH` | `CLIENT_CERTIFICATE_TLS_AUTH` | `IAM_AUTH` | `IAM_OAUTHBEARER_AUTH` | `OAUTHBEARER_AUDIENCE` | `OAUTHBEARER_AUTH` | `OAUTHBEARER_IDENTITY_POOL` | `OAUTHBEARER_LOGICAL_CLUSTER` | `OAUTHBEARER_SCOPE` | `SASL_SCRAM_256_AUTH` | `SASL_SCRAM_512_AUTH` | `SERVER_ROOT_CA_CERTIFICATE` | `VPC_SECURITY_GROUP` | `VPC_SUBNET`

Tipo: lista de [SourceAccessConfiguration](#)

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SourceAccessConfigurations](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

StartingPosition

A posição em um fluxo da qual você deseja iniciar a leitura.

- `AT_TIMESTAMP` – Especifique um tempo a partir do qual iniciar a leitura dos registros.
- `LATEST` – Leia somente registros novos.
- `TRIM_HORIZON` – Processe todos os registros disponíveis.

Valores válidos: `AT_TIMESTAMP` | `LATEST` | `TRIM_HORIZON`

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StartingPosition](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

StartingPositionTimestamp

O tempo a partir do qual iniciar a leitura, em segundos no horário do Unix. Defina `StartingPositionTimestamp` quando `StartingPosition` é especificado como `.AT_TIMESTAMP`

Tipo: duplo

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StartingPositionTimestamp](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Topics

O nome do tópico do Kafka.

Tipo: lista

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Topics](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Exemplos

Configuração completa com perfis do IAM

O exemplo a seguir mostra uma configuração completa, incluindo a configuração do perfil do IAM necessário para usar o 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 exemplo

```

ProvisionedPollerConfig:
  MinimumPollers: 1
  MaximumPollers: 200

```

Self-managed Fonte do evento Kafka

O exemplo a seguir mostra o tipo de origem de um evento SelfManagedKafka.

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

```

Self-managed Fonte de eventos Kafka com autenticação OAuth 2.0

O exemplo a seguir é autenticado com SASL/OAUTHBEARER. A OAUTHBEARER_AUTH entrada aponta para o segredo que contém as credenciais do OAuth 2.0, e cada entrada de metadados carrega um valor literal que o Lambda passa para seu provedor de identidade ou corretores.

```
Events:
  SelfManagedKafkaEvent:
    Type: SelfManagedKafka
    Properties:
      KafkaBootstrapServers:
        - abc.xyz.com:9092
      Topics:
        - MyKafkaTopic
      StartingPosition: LATEST
      ConsumerGroupId: consumergroup1
      SourceAccessConfigurations:
        - Type: OAUTHBEARER_AUTH
          URI: arn:aws:secretsmanager:us-west-2:123456789012:secret:my-path/my-oauth-secret-1a2b3c
        - Type: OAUTHBEARER_SCOPE
          URI: read write
        - Type: OAUTHBEARER_AUDIENCE
          URI: https://api.example.com
        - Type: SERVER_ROOT_CA_CERTIFICATE
          URI: arn:aws:secretsmanager:us-west-2:123456789012:secret:my-path/my-server-root-ca-cert-7p8q9r
        - Type: VPC_SUBNET
          URI: subnet:subnet-12345678
        - Type: VPC_SECURITY_GROUP
          URI: security_group:sg-12345678
```

Self-managed Fonte de eventos do Kafka com autenticação IAM

O exemplo a seguir é autenticado com o IAM. A IAM_AUTH entrada não ocupa nenhum URI campo porque o Lambda assina cada conexão com as credenciais da função de execução da sua função. Conceda à sua função de execução as permissões de cluster que o Lambda exige, pois AWS SAM não as adiciona.

```
Events:
  SelfManagedKafkaEvent:
    Type: SelfManagedKafka
```



```
Properties:
  KafkaBootstrapServers:
    - abc.xyz.com:9092
  Topics:
    - MyKafkaTopic
  StartingPosition: LATEST
  ConsumerGroupId: consumergroup1
  SourceAccessConfigurations:
    - Type: IAM_AUTH
    - Type: VPC_SUBNET
      URI: subnet:subnet-12345678
    - Type: VPC_SECURITY_GROUP
      URI: security_group:sg-12345678
```

Self-managed Fonte de eventos do Kafka com autenticação IAM e OAuth 2.0

O exemplo a seguir é autenticado com `IAM_OAUTHBEARER_AUTH`. Essa entrada não ocupa nenhum URI campo, mas você deve combiná-la com uma `OAUTHBEARER_AUDIENCE` entrada cujo público URI seja o que seus corretores esperam. O Lambda solicita um token de identidade web de curta duração para a função de execução, que precisa da `sts:GetWebIdentityToken` permissão, e seus corretores devem confiar no AWS emissor e aceitar o valor do público.

```
Events:
  SelfManagedKafkaEvent:
    Type: SelfManagedKafka
    Properties:
      KafkaBootstrapServers:
        - abc.xyz.com:9092
      Topics:
        - MyKafkaTopic
      StartingPosition: LATEST
      ConsumerGroupId: consumergroup1
      SourceAccessConfigurations:
        - Type: IAM_OAUTHBEARER_AUTH
        - Type: OAUTHBEARER_AUDIENCE
          URI: kafka-cluster
        - Type: VPC_SUBNET
          URI: subnet:subnet-12345678
        - Type: VPC_SECURITY_GROUP
          URI: security_group:sg-12345678
```

Self-managed Fonte de eventos Kafka com AWS Glue Schema Registry

Veja a seguir um exemplo de um tipo de fonte de SelfManagedKafka evento configurado com o 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
```

Self-managed Fonte de eventos do Kafka com Confluent Schema Registry

Veja a seguir um exemplo de um tipo de origem de eventos do SelfManagedKafka configurado com o 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

O objeto que descreve um tipo de fonte de evento SNS.

O SAM gera recursos [AWS::SNS::Subscription](#) quando esse tipo de evento é definido

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```

FilterPolicy: SnsFilterPolicy
FilterPolicyScope: String
RedrivePolicy: Json
Region: String
SqsSubscription: Boolean | SqsSubscriptionObject
Topic: String

```

Propriedades

FilterPolicy

O JSON da política de filtros atribuído à assinatura. Para obter mais informações, consulte [GetSubscriptionAttributes](#) a Referência da API do Amazon Simple Notification Service.

Digite: [SnsFilterPolicy](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FilterPolicy](#) propriedade de um `AWS::SNS::Subscription` recurso.

FilterPolicyScope

Esse atributo permite que você escolha o escopo da filtragem usando um dos seguintes tipos de valor de string:

- `MessageAttributes` – O filtro é aplicado aos atributos de mensagem.
- `MessageBody` – O filtro é aplicado ao corpo da mensagem.

Tipo: string

Obrigatório: não

Padrão: `MessageAttributes`

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FilterPolicyScope](#) propriedade de um `AWS::SNS::Subscription` recurso.

RedrivePolicy

Quando especificado, envia mensagens não entregues para a fila de mensagens não entregues do Amazon SQS especificada. As mensagens que não podem ser entregues devido a erros do cliente (por exemplo, quando o endpoint inscrito está inacessível) ou erros do servidor (por exemplo, quando o serviço que ativa o endpoint inscrito se torna indisponível) são mantidas na fila de mensagens não entregues para análise ou reprocessamento adicionais.

Para obter mais informações sobre a política de redirecionamento e filas de mensagens não entregues, consulte [Filas de mensagens não entregues do Amazon SQS](#) no Guia do desenvolvedor do Amazon Simple Queue Service.

Tipo: Json

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RedrivePolicy](#) propriedade de um `AWS::SNS::Subscription` recurso.

Region

Para assinaturas entre regiões, a região em que o tópico reside.

Se nenhuma região for especificada, CloudFormation usa a região do chamador como padrão.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Region](#) propriedade de um `AWS::SNS::Subscription` recurso.

SqsSubscription

Defina essa propriedade como verdadeira ou especifique `SqsSubscriptionObject` para habilitar notificações de tópicos do SNS em lotes em uma fila SQS. Definir essa propriedade para `true` cria uma nova fila SQS, enquanto especificar um `SqsSubscriptionObject` usa uma fila SQS existente.

Tipo: Boolean | [SqsSubscriptionObject](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Topic

O ARN do tópico que deseja assinar

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TopicArn](#) propriedade de um `AWS::SNS::Subscription` recurso.

Exemplos

Exemplo de origem do evento do SNS

Exemplo de origem do evento do 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

Especifique uma opção de fila SQS existente para o evento SNS

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
BatchSize: String
Enabled: Boolean
QueueArn: String
QueuePolicyLogicalId: String
QueueUrl: String
```

Propriedades

BatchSize

O número máximo de itens a serem recuperados em um único lote para a fila do SQS.

Tipo: string

Obrigatório: não

Padrão: 10

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Enabled

Desabilita o mapeamento de origens de eventos SQS para pausar a sondagem e a invocação.

Tipo: booleano

Obrigatório: não

Padrão: verdadeiro

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

QueueArn

Especifique um braço de fila SQS existente.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

QueuePolicyLogicalId

Forneça um nome LogicalID personalizado para o recurso. [AWS::SQS::QueuePolicy](#)

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

QueueUrl

Especifique o URL da fila associado à propriedade QueueArn.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Evento existente do SQS for SNS

Exemplo para adicionar uma fila SQS existente para assinar um tópico do SNS.

YAML

```
QueuePolicyLogicalId: CustomQueuePolicyLogicalId
QueueArn:
  Fn::GetAtt: MyCustomQueue.Arn
QueueUrl:
  Ref: MyCustomQueue
BatchSize: 5
```

SQS

O objeto que descreve um tipo de origem do evento SQS. Para obter mais informações, consulte Como [usar AWS Lambda com o Amazon SQS](#) no Guia do AWS Lambda desenvolvedor.

O SAM gera recursos [AWS::Lambda::EventSourceMapping](#) quando esse tipo de evento é definido

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
BatchSize: Integer
Enabled: Boolean
FilterCriteria: FilterCriteria
FunctionResponseTypes: List
KmsKeyArn: String
MaximumBatchingWindowInSeconds: Integer
MetricsConfig: MetricsConfig
```


[ProvisionedPollerConfig](#): [ProvisionedPollerConfig](#)
[Queue](#): *String*
[ScalingConfig](#): [ScalingConfig](#)

Propriedades

BatchSize

O número máximo de itens a serem recuperados em um único lote.

Tipo: inteiro

Obrigatório: não

Padrão: 10

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BatchSize](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Mínimo: 1

Maximum: 10000

Enabled

Desabilita o mapeamento de origens de eventos para pausar a sondagem e a invocação.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Enabled](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

FilterCriteria

Um objeto que define os critérios para determinar se o Lambda deve processar um evento. Para obter mais informações, consulte [Filtrando eventos do AWS Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Digite: [FilterCriteria](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FilterCriteria](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

FunctionResponseTypes

Uma lista de tipos de resposta atuais aplicados ao mapeamento da origem do evento. Para obter mais informações, consulte [Relatar falhas de itens em lote](#) no Guia do desenvolvedor do AWS Lambda .

Valores válidos: ReportBatchItemFailures

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FunctionResponseTypes](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

KmsKeyArn

O nome do recurso da Amazon (ARN) da chave para criptografar informações relacionadas a esse evento.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [KmsKeyArn](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MaximumBatchingWindowInSeconds

O tempo máximo, em segundos, para coletar registros antes de invocar a função.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaximumBatchingWindowInSeconds](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

MetricsConfig

Uma configuração opcional para obter métricas aprimoradas para os mapeamentos da origem do evento que capturam cada estágio do processamento. Para ver um exemplo, consulte [MetricsConfig evento](#).

Digite: [MetricsConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MetricsConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

ProvisionedPollerConfig

Configuração para aumentar a quantidade de sondagens usados para computar os mapeamentos da origem do evento. Essa configuração permite um mínimo de 2 pollers e um máximo de 2.000 pollers. Para obter um exemplo, consulte [ProvisionedPollerConfig exemplo](#).

Digite: [ProvisionedPollerConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ProvisionedPollerConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Queue

O ARN da fila.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventSourceArn](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

ScalingConfig

Configuração de escalabilidade dos agentes de sondagem SQS para controlar a taxa de invocação e definir o máximo de invocações simultâneas.

Digite: [ScalingConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ScalingConfig](#) propriedade de um `AWS::Lambda::EventSourceMapping` recurso.

Exemplos

MetricsConfig evento

Veja a seguir um exemplo de um recurso que usa a propriedade MetricsConfig para capturar cada estágio de processamento para seus mapeamentos da origem do evento.

```
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 do SQS básico

```
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"]}'
```

Configure relatórios parciais em lotes para sua fila SQS

```
Events:
  SQSEvent:
    Type: SQS
    Properties:
```

```

Enabled: true
FunctionResponseTypes:
  - ReportBatchItemFailures
Queue: !GetAtt MySqsQueue.Arn
BatchSize: 10

```

Função de evento Lambda com um evento SQS que tem escalabilidade configurada

```

MyFunction:
  Type: AWS::Serverless::Function
  Properties:
    ...
    Events:
      MySQSEvent:
        Type: SQS
        Properties:
          ...
          ScalingConfig:
            MaximumConcurrency: 10

```

ProvisionedPollerConfig exemplo

```

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

O [pacote de implantação](#) para uma função de evento Lambda.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Bucket: String
Key: String
StorageMode: String
Version: String
```

Propriedades

Bucket

Um bucket do Amazon S3 na mesma AWS região da sua função.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [S3Bucket](#) propriedade do tipo de `AWS::Lambda::Function` dados.

Key

A chave do Amazon S3 do pacote de implantação.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [S3Key](#) propriedade do tipo de `AWS::Lambda::Function` dados.

StorageMode

Controla como o Lambda armazena o pacote de implantação.

- **COPY**— O Lambda carrega uma cópia do seu pacote de implantação no armazenamento. Lambda-managed Você pode excluir o objeto Amazon S3 depois que o Lambda criar a função.
- **REFERENCE**— O Lambda faz referência ao pacote de implantação do seu bucket Amazon S3 e não armazena uma cópia. O objeto deve permanecer no lugar e o Lambda deve manter o acesso a ele durante toda a vida útil da função.

Para usar `REFERENCE`, habilite o controle de versão em seu bucket do Amazon S3 e conceda ao serviço Lambda acesso principal ao objeto. Para obter mais informações, consulte [Armazenamento de código do Self-managed Amazon S3](#) no Guia do AWS Lambda desenvolvedor.

Tipo: string

Valores válidos: `COPY` | `REFERENCE`

Obrigatório: não

Padrão: `COPY`

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [S3ObjectStorageMode](#) propriedade do tipo de `AWS::Lambda::Function` Code dados.

Version

Para objetos com controle de versão, a versão do objeto do pacote de implantação a ser usada.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [S3ObjectVersion](#) propriedade do tipo de `AWS::Lambda::Function` Code dados.

Exemplos

FunctionCode

`CodeUri`: exemplo de código da função

YAML

```
CodeUri:
  Bucket: sam-s3-demo-bucket-name
  Key: mykey-name
  Version: 121212
```

Self-managed Armazenamento de código do Amazon S3

O seguinte `CodeUri` faz referência ao pacote de implantação do seu próprio bucket do Amazon S3 em vez de copiá-lo para Lambda-managed o armazenamento.

YAML

```
CodeUri:
  Bucket: amzn-s3-demo-bucket-name
  Key: mykey-name
  Version: 121212
  StorageMode: REFERENCE
```

FunctionUrlConfig

Cria uma URL de AWS Lambda função com os parâmetros de configuração especificados. Um URL da função do Lambda é um endpoint HTTPS que você pode usar para invocar a função.

Por padrão, o URL da função que você cria usa a \$LATEST versão da sua função de evento Lambda. Se você especificar um `AutoPublishAlias` para sua função do Lambda, o endpoint se conectará ao alias da função especificada.

Para obter mais informações, consulte [URLs da função do Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
AuthType: String
Cors: Cors
InvokeMode: String
```

Propriedades

AuthType

O tipo de autenticação para o URL da função. Para usar AWS Identity and Access Management (IAM) para autorizar solicitações, defina `AWS_IAM` como. Para acesso aberto, defina-o como `NONE`.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AuthType](#) propriedade de um `AWS::Lambda::Url` recurso.

Cors

As configurações de compartilhamento de recursos de origem cruzada (CORS) para o URL de função.

Type: [Cors](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Cors](#) propriedade de um `AWS::Lambda::Url` recurso.

InvokeMode

O modo em que o URL da sua função será invocado. Para que sua função retorne a resposta após a conclusão da invocação, defina como `BUFFERED`. Para que sua função transmita a resposta, defina como `RESPONSE_STREAM`. O valor padrão é `BUFFERED`.

Valores válidos: `BUFFERED` ou `RESPONSE_STREAM`

Tipo: string

Obrigatório: não

AWS CloudFormation compatibilidade: essa propriedade é passada diretamente para a [InvokeMode](#) propriedade de um `AWS::Lambda::Url` recurso.

Exemplos

URL da função

O exemplo a seguir cria uma função do Lambda com um URL da função. O URL da função usa autorização do 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 o provedor de capacidade ao qual as versões publicadas da função serão anexadas. Isso permite que a função seja executada em EC2 instâncias de propriedade do cliente gerenciadas pela Lambda.

 Note

Essa configuração determina o tipo de computação de uma função e precisa ser especificada durante a primeira implantação da função. Ele não pode ser adicionado ou removido após a criação do recurso da função.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Arn: String
ExecutionEnvironmentMemoryGiBPerVCpu: Float
PerExecutionEnvironmentMaxConcurrency: Integer
```

Propriedades

Arn

O ARN do provedor de capacidade a ser usado para essa função.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva do SAM.

`ExecutionEnvironmentMemoryGiBPerVCpu`

A proporção de memória (em GiB) para vCPU para cada ambiente de execução.

Note

A taxa de memória por CPU não pode exceder a memória total da função de 2048 MB. As memory-to-CPU proporções suportadas são de 2 GB, 4 GB ou 8 GB por CPU.

Tipo: Flutuador

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [`ExecutionEnvironmentMemoryGiBPerVCpu`](#) propriedade de um `AWS::Lambda::Function` recurso.

`PerExecutionEnvironmentMaxConcurrency`

O número máximo de execuções simultâneas por ambiente de execução (sandbox).

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [`PerExecutionEnvironmentMaxConcurrency`](#) propriedade de um `AWS::Lambda::Function` recurso.

Exemplos

Configuração do provedor de capacidade

O exemplo a seguir mostra uma configuração do provedor de capacidade que faz referência a um recurso do provedor de capacidade.

CapacityProviderConfig:

```
Arn: !GetAtt MyCapacityProvider.Arn
ExecutionEnvironmentMemoryGiBPerVCpu: 4.0
PerExecutionEnvironmentMaxConcurrency: 100
```

FunctionScalingConfig

Configura o comportamento de escalabilidade para as versões da Lambda Event Function, controlando o número de ambientes de execução (sandboxes) que podem ser criados. Essa configuração se aplica às versões \$ LATEST.PUBLISHED e da função numérica.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
MinExecutionEnvironments: Integer
MaxExecutionEnvironments: Integer
```

Propriedades

MinExecutionEnvironments

O número mínimo de ambientes de execução a serem mantidos para a versão da função.

Tipo: inteiro

Obrigatório: não

Padrão: 3

Mínimo: 0

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MinExecutionEnvironments](#) propriedade de um `AWS::Lambda::Function` recurso.

MaxExecutionEnvironments

O número máximo de ambientes de execução que podem ser criados para a versão da função.

Tipo: inteiro

Obrigatório: não

Padrão: 3

Mínimo: 0

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaxExecutionEnvironments](#) propriedade de um `AWS::Lambda::Function` recurso.

Exemplos

Configuração de escalonamento de funções

O exemplo a seguir mostra uma configuração de escalonamento de funções com ambientes de execução mínima e máxima.

```
FunctionScalingConfig:
  MinExecutionEnvironments: 5
  MaxExecutionEnvironments: 100
```

AWS::Serverless::API GraphQL

Use o tipo de `AWS::Serverless::GraphQLApi` recurso AWS Serverless Application Model (AWS SAM) para criar e configurar uma AWS AppSync GraphQL API para seu aplicativo sem servidor.

Para saber mais AWS AppSync, consulte [O que é AWS AppSync?](#) no Guia do AWS AppSync desenvolvedor.

Sintaxe

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
```

Propriedades

ApiKeys

Crie uma chave exclusiva que possa ser usada para realizar operações GraphQL que exigem uma chave de API.

Digite: [ApiKeys](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Auth

Configure a autenticação para sua API GraphQL.

Tipo: [Auth](#)

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Cache

A entrada de uma operação CreateApiCache.

Tipo: [AWS::AppSync::ApiCache](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para o ApiCache recurso [AWS::AppSync::](#).

DataSources

Crie fontes de dados para as funções AWS AppSync às quais se conectar. AWS SAM oferece suporte ao Amazon DynamoDB e às fontes de dados AWS Lambda .

Digite: [DataSource](#)

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

DomainName

O nome de domínio personalizado para a sua API GraphQL.

Tipo: [AWS::AppSync:: DomainName](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para o DomainName recurso [AWS::AppSync::](#). AWS SAM gera automaticamente o DomainNameApiAssociation recurso [AWS::AppSync::](#).

Functions

Configure funções nos GraphQL APIs para realizar determinadas operações.

Tipo: [Função](#)

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Logging

Configura o CloudWatch registro da Amazon para sua GraphQL API.

Se você não especificar essa propriedade, AWS SAM gerará CloudWatchLogsRoleArn e definirá os seguintes valores:

- ExcludeVerboseContent: true
- FieldLogLevel: ALL

Para cancelar o registro, especifique o seguinte:

Logging: false

Digite: [LogConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [LogConfig](#) propriedade de um `AWS::AppSync::GraphQLApi` recurso.

LogicalId

O nome exclusivo da sua API GraphQL.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::AppSync::GraphQLApi` recurso.

Name

O nome da sua API GraphQL. Especifique essa propriedade para substituir o valor `LogicalId`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::AppSync::GraphQLApi` recurso.

Resolvers

Configure resolvedores para os campos da sua API GraphQL. O AWS SAM suporta [JavaScript resolvedores de pipeline](#).

Tipo: [Resolver](#)

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SchemaInline

A representação de texto de um esquema GraphQL no formato SDL.

Tipo: string

Obrigatório: condicional. Você deve especificar o `SchemaInline` ou o `SchemaUri`.

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Definition](#) propriedade de um `AWS::AppSync::GraphQLSchema` recurso.

SchemaUri

O URI ou o caminho do bucket do Amazon Simple Storage Service (Amazon S3) do esquema para uma pasta local.

Se você especificar um caminho para uma pasta local, CloudFormation exigirá que o arquivo seja primeiro carregado no Amazon S3 antes da implantação. Você pode usar o AWS SAMCLI para facilitar esse processo. Para obter mais informações, consulte [Como AWS SAM carrega arquivos locais na implantação](#).

Tipo: string

Obrigatório: condicional. Você deve especificar o `SchemaInline` ou o `SchemaUri`.

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DefinitionS3Location](#) propriedade de um `AWS::AppSync::GraphQLSchema` recurso.

Tags

Tags (pares de chave-valor) para essa API GraphQL. Use as tags para identificar e categorizar os recursos.

Tipo: lista de [Tag](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Tag](#) propriedade de um `AWS::AppSync::GraphQLApi` recurso.

XrayEnabled

Indique se o [AWS X-Ray rastreamento](#) deve ser usado para esse recurso.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [XrayEnabled](#) propriedade de um `AWS::AppSync::GraphQLApi` recurso.

Valores de retorno

Para obter uma lista dos valores de retorno, consulte o [AWS::Serverless::GraphQLApi](#) no [Guia do usuário do CloudFormation](#).

Exemplos

API DO GraphQL com a fonte de dados do DynamoDB

Neste exemplo, criamos uma API GraphQL que usa uma tabela do DynamoDB como fonte de dados.

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!
}
```

template.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),
  };
}
```

pré-processar PostItem.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;
}
```

Aqui está nosso código de resolução:

getPost.js

```
export function request(ctx) {
  return {};
}

export function response(ctx) {
  return ctx.prev.result;
}
```

GraphQL API com uma função de evento Lambda como fonte de dados

Neste exemplo, criamos uma GraphQL API que usa uma função de evento Lambda como fonte de dados.

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
}
```

Aqui estão nossas funções:

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

Crie uma chave exclusiva que possa ser usada para realizar operações GraphQL que exigem uma chave de API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
LogicalId:  
  ApiKeyId: String  
  Description: String  
  ExpiresOn: Double
```

Propriedades

ApiKeyId

O nome exclusivo da sua chave de API. Especifique para substituir o valor `LogicalId`.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ApiKeyId](#) propriedade de um `AWS::AppSync::ApiKey` recurso.

Description

Descrição da sua chave de API.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::AppSync::ApiKey` recurso.

ExpiresOn

O tempo após o qual a chave de API expira. A data é representada como segundos desde o epoch, arredondada para baixo para a hora mais próxima.

Tipo: duplo

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Expires](#) propriedade de um `AWS::AppSync::ApiKey` recurso.

LogicalId

O nome exclusivo da sua chave de API.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ApiKeyId](#) propriedade de um `AWS::AppSync::ApiKey` recurso.

Auth

Configure a autorização para sua GraphQL API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Additional:
- AuthProvider
LambdaAuthorizer: LambdaAuthorizerConfig
OpenIDConnect: OpenIDConnectConfig
Type: String
UserPool: UserPoolConfig
```

Propriedades

Additional

Uma lista de tipos de autorização adicionais para sua GraphQL API.

Tipo: Lista de [AuthProvider](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

LambdaAuthorizer

Especifique a configuração de autorização opcional para seu autorizador da Lambda Event Function. Você pode configurar essa propriedade opcional quando Type for especificado como AWS_LAMBDA.

Tipo: [LambdaAuthorizerConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [LambdaAuthorizerConfig](#) propriedade de um `AWS::AppSync::GraphQLApi` recurso.

OpenIDConnect

Especifique a configuração de autorização opcional para seu serviço OpenID Connect compatível. Você pode configurar essa propriedade opcional quando Type for especificado como OPENID_CONNECT.

Tipo: [OpenIDConnectConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [OpenIDConnectConfig](#) propriedade de um `AWS::AppSync::GraphQLApi` recurso.

Type

O tipo de autorização padrão entre aplicativos e sua AWS AppSync GraphQL API.

Para obter uma lista e uma descrição dos valores permitidos, consulte [Autorização e autenticação](#) no AWS AppSync Guia do desenvolvedor.

Quando você especifica um autorizador Lambda (AWS_LAMBDA), AWS SAM cria uma política AWS Identity and Access Management (IAM) para provisionar permissões entre sua API GraphQL e a função Lambda.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AuthenticationType](#) propriedade de um `AWS::AppSync::GraphQLApi` recurso.

UserPool

Especifique a configuração de autorização opcional para usar grupos de usuários do Amazon Cognito. Você pode configurar essa propriedade opcional quando Type for especificado como AMAZON_COGNITO_USER_POOLS.

Tipo: [UserPoolConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [UserPoolConfig](#) propriedade de um `AWS::AppSync::GraphQLApi` recurso.

Exemplos

Configurar um tipo de autorização padrão e adicional

Neste exemplo, começamos configurando um autorizador Lambda como o tipo de autorização padrão para nossa 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
```

Em seguida, configuramos tipos de autorização adicionais para nossa GraphQL API adicionando o seguinte ao nosso modelo AWS SAM :

```
Additional:
- Type: AWS_IAM
- Type: API_KEY
- Type: OPENID_CONNECT
  OpenIDConnect:
```

```
AuthTTL: 10
ClientId: myId
IatTTL: 10
Issuer: prod
```

Isso resulta no seguinte AWS SAM modelo:

```
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

Configuração de autorização opcional para seus tipos adicionais de autorização de GraphQL API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
LambdaAuthorizer: LambdaAuthorizerConfig
OpenIDConnect: OpenIDConnectConfig
```

Type: *String*

UserPool: [UserPoolConfig](#)

Propriedades

LambdaAuthorizer

Especifique a configuração de autorização opcional para seu autorizador da Função de AWS Lambda Eventos. Você pode configurar essa propriedade opcional quando Type for especificado como AWS_LAMBDA.

Tipo: [LambdaAuthorizerConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [LambdaAuthorizerConfig](#) propriedade de um `AWS::AppSync::GraphQLApi` [AdditionalAuthenticationProvider](#) objeto.

OpenIDConnect

Especifique a configuração de autorização opcional para seu serviço OpenID Connect compatível. Você pode configurar essa propriedade opcional quando Type for especificado como OPENID_CONNECT.

Tipo: [OpenIDConnectConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [OpenIDConnectConfig](#) propriedade de um `AWS::AppSync::GraphQLApi` [AdditionalAuthenticationProvider](#) objeto.

Type

O tipo de autorização padrão entre aplicativos e sua AWS AppSync GraphQL API.

Para obter uma lista e uma descrição dos valores permitidos, consulte [Autorização e autenticação](#) no AWS AppSync Guia do desenvolvedor.

Quando você especifica um autorizador Lambda (AWS_LAMBDA), AWS SAM cria uma política AWS Identity and Access Management (IAM) para provisionar permissões entre sua API GraphQL e a função Lambda.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AuthenticationType](#) propriedade de um `AWS::AppSync::GraphQLApi` [AdditionalAuthenticationProvider](#) objeto.

UserPool

Especifique a configuração de autorização opcional para usar grupos de usuários do Amazon Cognito. Você pode configurar essa propriedade opcional quando Type for especificado como `AMAZON_COGNITO_USER_POOLS`.

Tipo: [UserPoolConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [UserPoolConfig](#) propriedade de um `AWS::AppSync::GraphQLApi` [AdditionalAuthenticationProvider](#) objeto.

DataSource

Configure uma fonte de dados à qual seu resolvidor de GraphQL API possa se conectar. Você pode usar modelos AWS Serverless Application Model (AWS SAM) para configurar conexões com as seguintes fontes de dados:

- Amazon DynamoDB
- AWS Lambda

Para saber mais sobre fontes de dados, consulte [Anexar uma fonte de dados](#) no AWS AppSync Guia do desenvolvedor.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
DynamoDb: DynamoDb  
Lambda: Lambda
```

Propriedades

DynamoDb

Configure uma tabela do DynamoDB como fonte de dados para seu resolvidor de GraphQL API.

Digite: [DynamoDb](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Lambda

Configure uma função de evento Lambda como fonte de dados para seu resolvidor de GraphQL API.

Type: [Lambda](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

DynamoDb

Configure uma tabela do Amazon DynamoDB como fonte de dados para seu resolvidor de GraphQL API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
LogicalId:  
DeltaSync: DeltaSyncConfig
```



```
Description: String  
Name: String  
Permissions: List  
Region: String  
ServiceRoleArn: String  
TableArn: String  
TableName: String  
UseCallerCredentials: Boolean  
Versioned: Boolean
```

Propriedades

DeltaSync

Descreve uma configuração de sincronização delta.

Digite: [DeltaSyncConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DeltaSyncConfig](#) propriedade de um `AWS::AppSync::DataSource DynamoDBConfig` objeto.

Description

A descrição da sua fonte de dados.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::AppSync::DataSource` recurso.

LogicalId

O nome exclusivo de sua fonte de dados.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::AppSync::DataSource` recurso.

Name

O nome da sua fonte de dados. Especifique essa propriedade para substituir o valor `LogicalId`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::AppSync::DataSource` recurso.

Permissions

Provisione permissões para sua fonte de dados usando [conectores do AWS SAM](#). Você pode fornecer qualquer um dos seguintes valores em uma lista:

- `Read` - Permite que seu resolvedor leia sua fonte de dados.
- `Write` - Permite que seu resolvedor grave em sua fonte de dados.

AWS SAM usa um `AWS::Serverless::Connector` recurso que é transformado na implantação para provisionar suas permissões. Para saber mais sobre os recursos gerados, consulte [CloudFormation recursos gerados quando você especifica `AWS::Serverless::Connector`](#).

Note

É possível especificar `Permissions` ou `ServiceRoleArn`, mas não ambos. Se nenhum for especificado, AWS SAM gerará valores padrão de `Read Write` e. Para revogar o acesso à sua fonte de dados, remova o objeto `DynamoDB` do seu modelo. AWS SAM

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente. É semelhante à propriedade [Permissions](#) de um recurso `AWS::Serverless::Connector`.

Region

A Região da AWS da sua tabela do `DynamoDB`. Se você não especificar, AWS SAM usa [AWS::Region](#).

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AwsRegion](#) propriedade de um `AWS::AppSync::DataSource` DynamoDBConfig objeto.

ServiceRoleArn

O ARN da função de serviço AWS Identity and Access Management (IAM) da fonte de dados. O sistema assume essa função ao acessar a fonte de dados.

É possível especificar `Permissions` ou `ServiceRoleArn`, mas não ambos.

Tipo: string

Obrigatório: Não. Se não for especificado, AWS SAM aplica o valor padrão para `Permissions`.

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ServiceRoleArn](#) propriedade de um `AWS::AppSync::DataSource` recurso.

TableArn

O ARN da tabela do DynamoDB.

Tipo: string

Obrigatório: condicional. Se você não especificar o `ServiceRoleArn`, o `TableArn` será necessário.

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

TableName

O nome da tabela.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TableName](#) propriedade de um `AWS::AppSync::DataSource` DynamoDBConfig objeto.

UseCallerCredentials

Defina como `true` para usar o IAM com essa fonte de dados.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [UseCallerCredentials](#) propriedade de um `AWS::AppSync::DataSource` DynamoDBConfig objeto.

Versioned

Defina como `true` para usar a [Detecção de conflitos, resolução de conflitos e sincronização](#) com esta fonte de dados.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Versioned](#) propriedade de um `AWS::AppSync::DataSource` DynamoDBConfig objeto.

Lambda

Configure uma função de AWS Lambda evento como fonte de dados para seu resolvidor de GraphQL API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
LogicalId:  
  Description: String  
  FunctionArn: String  
  Name: String  
  ServiceRoleArn: String
```

Propriedades

Description

A descrição da sua fonte de dados.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::AppSync::DataSource` recurso.

FunctionArn

O ARN da função do Lambda.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [LambdaFunctionArn](#) propriedade de um `AWS::AppSync::DataSource` `LambdaConfig` objeto.

LogicalId

O nome exclusivo de sua fonte de dados.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::AppSync::DataSource` recurso.

Name

O nome da sua fonte de dados. Especifique essa propriedade para substituir o valor `LogicalId`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::AppSync::DataSource` recurso.

ServiceRoleArn

O ARN da função de serviço AWS Identity and Access Management (IAM) da fonte de dados. O sistema assume essa função ao acessar a fonte de dados.

Note

Para revogar o acesso à sua fonte de dados, remova o objeto Lambda do seu modelo AWS SAM .

Tipo: string

Obrigatório: Não. Se não for especificado, AWS SAM provisionará `Write` as permissões usando [conectores do AWS SAM](#) .

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ServiceRoleArn](#) propriedade de um `AWS::AppSync::DataSource` recurso.

Função

Configure funções nos GraphQL APIs para realizar determinadas operações.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
LogicalId:  
  CodeUri: String  
  DataSource: String  
  Description: String  
  Id: String  
  InlineCode: String  
  MaxBatchSize: Integer  
  Name: String
```

[Runtime](#): [Runtime](#)

[Sync](#): [SyncConfig](#)

Propriedades

CodeUri

O URI Amazon Simple Storage Service (Amazon S3) ou o caminho para a pasta local de código de função.

Se você especificar um caminho para uma pasta local, CloudFormation exigirá que o arquivo seja primeiro carregado no Amazon S3 antes da implantação. Você pode usar o AWS SAMCLI para facilitar esse processo. Para obter mais informações, consulte [Como AWS SAM carrega arquivos locais na implantação](#).

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CodeS3Location](#) propriedade de um `AWS::AppSync::FunctionConfiguration` recurso.

DataSource

O nome da fonte de dados à qual esta função será anexada.

- Para referenciar uma fonte de dados dentro do recurso `AWS::Serverless::GraphQLApi`, especifique sua ID lógica.
- Para referenciar uma fonte de dados fora do recurso `AWS::Serverless::GraphQLApi`, forneça seu atributo `Name` usando a função `Fn::GetAtt` intrínseca. Por exemplo, `!GetAtt MyLambdaDataSource.Name`
- Para referenciar uma fonte de dados de uma pilha diferente, use [Fn::ImportValue](#).

Se uma variação de `[NONE | None | none]` for especificada, AWS SAM gerará um `None` valor para o `AWS::AppSync::DataSource` [Type](#) objeto.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DataSourceName](#) propriedade de um `AWS::AppSync::FunctionConfiguration` recurso.

Description

A descrição de sua função.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::AppSync::FunctionConfiguration` recurso.

Id

O ID da função de uma função localizada fora do recurso `AWS::Serverless::GraphQLApi`.

- Para referenciar uma função dentro do mesmo AWS SAM modelo, use a função `Fn::GetAtt` intrínseca. Por exemplo, `Id: !GetAtt createPostItemFunc.FunctionId`.
- Para referenciar uma função de uma pilha diferente, use [Fn::ImportValue](#).

Ao usar `Id`, todas as outras propriedades não são permitidas. AWS SAM passará automaticamente o ID da função de sua função referenciada.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

InlineCode

O código de função que contém as funções de solicitação e resposta.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Code](#) propriedade de um `AWS::AppSync::FunctionConfiguration` recurso.

LogicalId

Escolha o nome único para a função.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::AppSync::FunctionConfiguration` recurso.

MaxBatchSize

O número máximo de entradas de solicitação do resolvidor que serão enviadas para uma única função de AWS Lambda evento em uma `BatchInvoke` operação.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaxBatchSize](#) propriedade de um `AWS::AppSync::FunctionConfiguration` recurso.

Name

Nome da função. Especifique para substituir o valor `LogicalId`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::AppSync::FunctionConfiguration` recurso.

Runtime

Descreve um tempo de execução usado por uma AWS AppSync função ou resolvidor de AWS AppSync pipeline. Especifica o nome e a versão do tempo de execução a ser usado.

Tipo: [Tempo de execução](#)

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente. É semelhante à propriedade [Runtime](#) de um recurso `AWS::AppSync::FunctionConfiguration`.

Sync

Descreve uma configuração de sincronização para uma função.

Especifica quais estratégias de detecção de conflitos e de resolução devem ser usadas quando a função for invocada.

Digite: [SyncConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SyncConfig](#) propriedade de um `AWS::AppSync::FunctionConfiguration` recurso.

Runtime

O tempo de execução do seu resolvidor ou função de pipeline. Especifica o nome e a versão a serem usados.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Name: String
Version: String
```

Propriedades

Name

O nome do tempo de execução a ser usado. Atualmente, o único valor permitido é `APPSYNC_JS`.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::AppSync::FunctionConfiguration` `AppSyncRuntime` objeto.

Version

A versão do tempo de execução a ser usada. Atualmente, a única versão permitida é `1.0.0`.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RuntimeVersion](#) propriedade de um `AWS::AppSync::FunctionConfiguration` `AppSyncRuntime` objeto.

Resolvedor

Configure resolvedores para os campos da sua GraphQL API. AWS Serverless Application Model (AWS SAM) suporta [resolvedores de JavaScript pipeline](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
OperationType:  
  LogicalId:  
    Caching: CachingConfig  
    CodeUri: String  
    FieldName: String  
    InlineCode: String  
    MaxBatchSize: Integer  
    Pipeline: List  
    Runtime: Runtime  
    Sync: SyncConfig
```

Propriedades

Caching

A configuração de cache para o resolvedor que tenha o cache ativado.

Digite: [CachingConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CachingConfig](#) propriedade de um `AWS::AppSync::Resolver` recurso.

CodeUri

O URI do código da função do resolvidor Amazon Simple Storage Service (Amazon S3) ou o caminho para uma pasta local.

Se você especificar um caminho para uma pasta local, CloudFormation exigirá que o arquivo seja primeiro carregado no Amazon S3 antes da implantação. Você pode usar o AWS SAMCLI para facilitar esse processo. Para obter mais informações, consulte [Como AWS SAM carrega arquivos locais na implantação](#).

Se nenhum deles CodeUri InlineCode for fornecido, AWS SAM será gerado InlineCode que redirecionará a solicitação para a primeira função do pipeline e receberá a resposta da última função do pipeline.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CodeS3Location](#) propriedade de um `AWS::AppSync::Resolver` recurso.

FieldName

O nome do seu resolvidor. Especifique essa propriedade para substituir o valor LogicalId.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FieldName](#) propriedade de um `AWS::AppSync::Resolver` recurso.

InlineCode

O código do resolvidor que contém as funções de solicitação e resposta.

Se nenhum deles CodeUri InlineCode for fornecido, AWS SAM será gerado InlineCode que redirecionará a solicitação para a primeira função do pipeline e receberá a resposta da última função do pipeline.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Code](#) propriedade de um `AWS::AppSync::Resolver` recurso.

LogicalId

O nome exclusivo do seu resolvedor. Em um esquema GraphQL, o nome do resolvedor deve corresponder ao nome do campo para o qual ele é usado. Use o mesmo nome de campo para LogicalId.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

MaxBatchSize

O número máximo de entradas de solicitação do resolvedor que serão enviadas para uma única função de AWS Lambda evento em uma BatchInvoke operação.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MaxBatchSize](#) propriedade de um `AWS::AppSync::Resolver` recurso.

OperationType

O tipo de operação GraphQL associada ao seu resolvedor. Por exemplo, Query, Mutation ou Subscription. Você pode agrupar vários resolvedores por LogicalId em um único OperationType.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TypeName](#) propriedade de um `AWS::AppSync::Resolver` recurso.

Pipeline

Funções vinculadas ao resolvedor de pipeline. Especifique as funções por ID lógica em uma lista.

Tipo: lista

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente. É semelhante à propriedade [PipelineConfig](#) de um recurso `AWS::AppSync::Resolver`.

Runtime

O tempo de execução do seu resolvidor ou função de pipeline. Especifica o nome e a versão a serem usados.

Tipo: [Tempo de execução](#)

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente. É semelhante à propriedade [Runtime](#) de um recurso `AWS::AppSync::Resolver`.

Sync

Descreve uma configuração de sincronização para um resolvidor.

Especifica quais estratégias de detecção de conflitos e de resolução devem ser usadas quando o resolvidor for invocado.

Digite: [SyncConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SyncConfig](#) propriedade de um `AWS::AppSync::Resolver` recurso.

Exemplos

Use o comando AWS SAM código de função resolvidor gerado e campos salvos como variáveis

Aqui está o esquema GraphQL do nosso exemplo:

```
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
}

```

Aqui está um trecho do nosso AWS SAM modelo:

```

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

```

Em nosso AWS SAM modelo, não especificamos `CodeUri` ou `InlineCode`. Na implantação, gera AWS SAM automaticamente o seguinte código embutido para nosso resolvidor:

```
export function request(ctx) {
  return {};
}

export function response(ctx) {
  return ctx.prev.result;
}
```

Esse código de resolução padrão redireciona a solicitação para a primeira função do pipeline e recebe a resposta da última função do pipeline.

Em nossa primeira função de pipeline, podemos usar o campo `args` fornecido para analisar o objeto de solicitação e criar nossas variáveis. Podemos então usar essas variáveis em nossa função. Veja um exemplo da nossa função `preprocessPostItem`:

```
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

O tempo de execução do seu resolvidor ou função de pipeline. Especifica o nome e a versão a serem usados.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Name: String  
Version: String
```

Propriedades

Name

O nome do tempo de execução a ser usado. Atualmente, o único valor permitido é APPSYNC_JS.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::AppSync::Resolver` `AppSyncRuntime` objeto.

Version

A versão do tempo de execução a ser usada. Atualmente, a única versão permitida é 1.0.0.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RuntimeVersion](#) propriedade de um `AWS::AppSync::Resolver` `AppSyncRuntime` objeto.

AWS::Serverless::HttpApi

Cria uma API HTTP do Amazon API Gateway, que permite criar APIs RESTful com menor latência e custos mais baixos do que APIs REST. Para obter mais informações, consulte [Trabalhar com APIs HTTP](#) no Guia do desenvolvedor do Gateway da API.

Recomendamos que você use CloudFormation ganchos ou políticas do IAM para verificar se os recursos do API Gateway têm autorizadores vinculados a eles para controlar o acesso a eles.

[Para obter mais informações sobre o uso de CloudFormation ganchos, consulte Registrando ganchos no guia do usuário da CloudFormation CLI e no repositório apigw-enforce-authorizer.](#)

GitHub

Para obter mais informações sobre o uso de políticas do IAM, consulte [Exigir que as rotas de API tenham autorização](#) no Guia do desenvolvedor do API Gateway.

Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

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
```

Propriedades

AccessLogSettings

As configurações para o registro em log de acesso em um estágio.

Digite: [AccessLogSettings](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AccessLogSettings](#) propriedade de um `AWS::ApiGatewayV2::Stage` recurso.

Auth

Configura a autorização para controlar o acesso à sua API HTTP do API Gateway.

Para mais informações, consulte [Controlar o acesso a APIs HTTP com autorizadores JWT](#) no Guia do desenvolvedor do API Gateway.

Digite: [HttpApiAuth](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

CorsConfiguration

Gerencia o compartilhamento de recursos de origem cruzada (CORS) para todas as APIs HTTP do API Gateway. Especifique o domínio a ser permitido como uma string ou especifique um objeto `HttpApiCorsConfiguration`. Observe que o CORS AWS SAM precisa modificar sua definição de OpenAPI, então o CORS funciona somente se `DefinitionBody` a propriedade for especificada.

Para obter mais informações, consulte [Configurar o CORS para uma API HTTP](#) no Guia do desenvolvedor do API Gateway.

Note

Se `CorsConfiguration` for definido em uma definição de OpenAPI e no nível da propriedade, AWS SAM mesclará as duas fontes de configuração com as propriedades que têm precedência. Se essa propriedade for definida como `true`, todas as origens serão permitidas.

Tipo: String | [HttpApiCorsConfiguration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

DefaultRouteSettings

As configurações de rota padrão para essa API HTTP. Essas configurações se aplicam a todas as rotas, a menos que sejam substituídas pela propriedade `RouteSettings` para determinadas rotas.

Digite: [RouteSettings](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RouteSettings](#) propriedade de um `AWS::ApiGatewayV2::Stage` recurso.

DefinitionBody

A definição de OpenAPI que descreve sua API HTTP. Se você não especificar a `DefinitionUri` ou `aDefinitionBody`, AWS SAM gera um `DefinitionBody` para você com base na configuração do seu modelo.

Type: JSON

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Body](#) propriedade de um `AWS::ApiGatewayV2::Api` recurso. Se determinadas propriedades forem fornecidas, AWS SAM poderá inserir conteúdo ou modificar o `DefinitionBody` antes de ser passado para CloudFormation. As propriedades incluem `Auth` e um `EventSource` do tipo `HttpApi` para um `AWS::Serverless::Function` recurso correspondente.

DefinitionUri

O URI, o caminho de arquivo local ou o objeto de localização do Amazon Simple Storage Service (Amazon S3) da definição de OpenAPI que define a API HTTP. O objeto Amazon S3 ao qual essa propriedade faz referência deve ser um arquivo de definição de OpenAPI válido. Se você não especificar a `DefinitionUri` ou se `DefinitionBody` for especificado, AWS SAM gera um `DefinitionBody` para você com base na configuração do seu modelo.

Se você fornecer um caminho de arquivo local, o modelo deverá passar pelo fluxo de trabalho que inclui o comando `sam deploy` ou `sam package` para que a definição seja transformada adequadamente.

As funções intrínsecas não são suportadas nos arquivos de OpenApi definição externos aos quais você faz referência. `DefinitionUri` Para importar uma OpenApi definição para o modelo, use a `DefinitionBody` propriedade com a [transformação Include](#).

Tipo: String | [HttpApiDefinition](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [BodyS3Location](#) propriedade de um `AWS::ApiGatewayV2::Api` recurso. As propriedades aninhadas do Amazon S3 têm nomes diferentes.

Description

A descrição do recurso de API HTTP.

Quando você especificar `Description`, AWS SAM modificará a OpenApi definição do recurso da API HTTP definindo o `description` campo. Os seguintes cenários resultarão em um erro:

- A `DefinitionBody` propriedade é especificada com o `description` campo definido na definição da Open API — isso resulta em um conflito do `description` campo que AWS SAM não será resolvido.
- A `DefinitionUri` propriedade é especificada — AWS SAM não modificará uma definição de API aberta que é recuperada do Amazon S3.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

DisableExecuteApiEndpoint

Especifica se os clientes podem invocar sua API HTTP usando o endpoint `execute-api` padrão `https://{api_id}.execute-api.{region}.amazonaws.com`. Por padrão, os clientes podem invocar sua API com o endpoint padrão. Para exigir que os clientes usem somente um nome de domínio personalizado para invocar sua API, desabilite o endpoint padrão.

Para usar essa propriedade, você deve especificar a propriedade `DefinitionBody` em vez da propriedade `DefinitionUri` ou definir `x-amazon-apigateway-endpoint-configuration` com `disableExecuteApiEndpoint` na definição de OpenAPI.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [DisableExecuteApiEndpoint](#) propriedade de um `AWS::ApiGatewayV2::Api` recurso. Ele é passado diretamente para a propriedade `disableExecuteApiEndpoint` de uma extensão [x-amazon-apigateway-endpoint-configuration](#), que é adicionada à propriedade [Body](#) de um recurso `AWS::ApiGatewayV2::Api`.

Domain

Configura um domínio personalizado para essa API HTTP do API Gateway.

Digite: [HttpApiDomainConfiguration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

FailOnWarnings

Especifica se deve reverter a criação da API HTTP (`true`) ou não (`false`) quando um aviso é encontrado. O valor padrão é `false`.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FailOnWarnings](#) propriedade de um `AWS::ApiGatewayV2::Api` recurso.

Name

O nome do recurso API HTTP.

Quando você especificar `Name`, AWS SAM modificará a definição de openAPI do recurso da API HTTP definindo o `title` campo. Os seguintes cenários resultarão em um erro:

- A `DefinitionBody` propriedade é especificada com o `title` campo definido na definição da Open API — isso resulta em um conflito do `title` campo que AWS SAM não será resolvido.
- A `DefinitionUri` propriedade é especificada — AWS SAM não modificará uma definição de API aberta que é recuperada do Amazon S3.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

PropagateTags

Indique se deseja ou não passar as tags da propriedade Tags para os recursos [AWS::Serverless::HttpApi](#) gerados. Especifique True para propagar as tags nos recursos gerados.

Tipo: booleano

Obrigatório: não

Padrão: False

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

RouteSettings

As configurações de rota, por rota, para essa API HTTP. Para obter mais informações, consulte [Trabalhar com rotas para APIs HTTP](#) no Guia do desenvolvedor do Gateway da API.

Digite: [RouteSettings](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RouteSettings](#) propriedade de um `AWS::ApiGatewayV2::Stage` recurso.

StageName

O nome do estágio da API. Se nenhum nome for especificado, AWS SAM usa o `$default` estágio do API Gateway.

Tipo: string

Obrigatório: não

Padrão: `$default`

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StageName](#) propriedade de um `AWS::ApiGatewayV2::Stage` recurso.

StageVariables

Um mapa que define as variáveis da etapa. Os nomes das variáveis podem ter caracteres alfanuméricos e sublinhados. Os valores devem corresponder a `[A-Za-z0-9-._~:/? #&=,] +`.

Type: [Json](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StageVariables](#) propriedade de um `AWS::ApiGatewayV2::Stage` recurso.

Tags

Um mapa (string para string) que especifica as tags a serem adicionadas a esse estágio do API Gateway. As chaves podem ter de 1 a 128 caracteres Unicode e não podem incluir o prefixo `aws:`. Use qualquer um dos seguintes caracteres: o conjunto de letras, dígitos, espaço em branco, `_`, `.`, `/`, `=`, `+` e `-`. Os valores podem ter de 1 a 256 caracteres Unicode.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Observações adicionais: a Tags propriedade AWS SAM precisa modificar sua definição de OpenAPI. Portanto, as tags são adicionadas somente se a `DefinitionBody` propriedade for especificada. Nenhuma tag será adicionada se a propriedade for especificada.

`DefinitionUri` AWS SAM adiciona automaticamente uma `httpapi:createdBy:SAM` tag. As tags também são adicionadas ao recurso `AWS::ApiGatewayV2::Stage` e ao recurso `AWS::ApiGatewayV2::DomainName` (se `DomainName` for especificado).

Valores de retorno

Ref.

Quando você transmite o ID lógico desse recurso para a função intrínseca `Ref`, `Ref` retorna o ID da API do recurso `AWS::ApiGatewayV2::Api` subjacente, como `a1bcdef2gh`.

Para obter mais informações sobre como usar a função Ref, consulte [Ref](#) no Guia do usuário do AWS CloudFormation .

Exemplos

Simple HttpApi

O exemplo a seguir mostra o mínimo necessário para configurar um endpoint de API HTTP apoiado por uma função de evento Lambda. Este exemplo usa a API HTTP padrão que AWS SAM cria.

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 com Auth

O exemplo a seguir mostra como configurar a autorização em endpoints da 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 com definição de OpenAPI

O exemplo a seguir mostra como adicionar uma definição de OpenAPI ao modelo.

Observe que AWS SAM preenche todas as integrações Lambda ausentes HttpApi para eventos que fazem referência a essa API HTTP. AWS SAM também adiciona todos os caminhos ausentes aos quais os HttpApi eventos fazem referência.

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 com definições de configuração

O exemplo a seguir mostra como adicionar configurações da API HTTP e do estágio ao modelo.

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

Configure a autorização para controlar o acesso à sua API HTTP do Amazon API Gateway.

Para obter mais informações sobre como configurar o acesso ao HTTP APIs, consulte [Controle e gerenciamento do acesso a uma API HTTP no API Gateway no Guia do desenvolvedor do API Gateway](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```

Authorizers: OAuth2Authorizer | LambdaAuthorizer
DefaultAuthorizer: String

```

`EnableIamAuthorizer`: *Boolean*

Propriedades

Authorizers

O autorizador usado para controlar o acesso à sua API do API Gateway.

Tipo: [OAuth2Autorizador](#) | [LambdaAuthorizer](#)

Obrigatório: não

Padrão: nenhum

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Notas adicionais: AWS SAM adiciona os autorizadores à definição da OpenAPI.

DefaultAuthorizer

Especifique o autorizador padrão a ser usado para autorizar chamadas de API para sua API do API Gateway. Você pode especificar o `AWS_IAM` como autorizador padrão se o `EnableIamAuthorizer` estiver definido como `true`. Caso contrário, especifique um autorizador que você definiu no `Authorizers`.

Tipo: string

Obrigatório: não

Padrão: nenhum

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

EnableIamAuthorizer

Especifique se deseja usar a autorização do IAM para a rota da API.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

OAuth 2.0 Autorizador

OAuth Exemplo de autorizador 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
```

autorizador do IAM

Exemplo de autorizador do IAM

YAML

```
Auth:
  EnableIamAuthorizer: true
  DefaultAuthorizer: AWS_IAM
```

LambdaAuthorizer

Configure um autorizador Lambda para controlar o acesso à sua API HTTP do Amazon API Gateway com uma AWS Lambda função de evento.

Para obter mais informações e exemplos, consulte Como [trabalhar com AWS Lambda autorizadores para APIs HTTP](#) no Guia do desenvolvedor do API Gateway.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
AuthorizerPayloadFormatVersion: String  
EnableFunctionDefaultPermissions: Boolean  
EnableSimpleResponses: Boolean  
FunctionArn: String  
FunctionInvokeRole: String  
Identity: LambdaAuthorizationIdentity
```

Propriedades

AuthorizerPayloadFormatVersion

Especifica o formato da carga útil enviada a um autorizador da API HTTP do Lambda. Necessário para autorizadores da API HTTP do Lambda.

Isso é passado para a seção `authorizerPayloadFormatVersion` de um `x-amazon-apigateway-authorizer` na seção `securitySchemes` de uma definição de OpenAPI.

Valores válidos: 1.0 ou 2.0

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

EnableFunctionDefaultPermissions

Por padrão, o recurso da API HTTP não tem permissão para invocar o autorizador do Lambda. Especifique essa propriedade como `true` para criar automaticamente permissões entre seu recurso de API HTTP e seu autorizador do Lambda.

Tipo: booleano

Obrigatório: não

Valor padrão: `false`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

EnableSimpleResponses

Especifica se um autorizador do Lambda retorna uma resposta em um formato simples. Por padrão, um autorizador Lambda deve retornar uma política AWS Identity and Access Management (IAM). Se habilitado, o autorizador do Lambda pode retornar um valor booleano em vez de uma política do IAM.

Isso é passado para a seção `enableSimpleResponses` de um `x-amazon-apigateway-authorizer` na seção `securitySchemes` de uma definição de OpenAPI.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

FunctionArn

O nome de recurso da Amazon (ARN) da função do Lambda que fornece autorização para a API.

Isso é passado para a seção `authorizerUri` de um `x-amazon-apigateway-authorizer` na seção `securitySchemes` de uma definição de OpenAPI.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

FunctionInvokeRole

O ARN do perfil do IAM que tem as credenciais necessárias para o API Gateway invocar a função do autorizador. Especifique esse parâmetro se a política baseada em recursos de sua função não conceder permissão ao API Gateway `lambda:InvokeFunction`.

Isso é passado para a seção `authorizerCredentials` de um `x-amazon-apigateway-authorizer` na seção `securitySchemes` de uma definição de OpenAPI.

Para obter mais informações, consulte [Criar um autorizador do Lambda](#) no Guia do desenvolvedor do API Gateway.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Identity

Especifica uma IdentitySource em uma solicitação recebida para um autorizador.

Isso é passado para a seção identitySource de um x-amazon-apigateway-authorizer na seção securitySchemes de uma definição de OpenAPI.

Digite: [LambdaAuthorizationIdentity](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

LambdaAuthorizer

LambdaAuthorizer exemplo

YAML

```
Auth:
  Authorizers:
    MyLambdaAuthorizer:
      AuthorizerPayloadFormatVersion: 2.0
      FunctionArn:
        Fn::GetAtt:
          - MyAuthFunction
          - Arn
      FunctionInvokeRole:
        Fn::GetAtt:
          - LambdaAuthInvokeRole
```

```
- Arn
Identity:
  Headers:
    - Authorization
```

LambdaAuthorizationIdentity

A propriedade `Use` pode ser usada para especificar um `IdentitySource` em uma solicitação recebida para um autorizador Lambda. Para obter mais informações sobre fontes de identidade, consulte [Fontes de identidade](#) no Guia do desenvolvedor do API Gateway.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Context: List
Headers: List
QueryStrings: List
ReauthorizeEvery: Integer
StageVariables: List
```

Propriedades

Context

Converte as cadeias de caracteres de contexto fornecidas em uma lista de expressões de mapeamento no formato `$context.contextString`.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Headers

Converte os cabeçalhos em uma lista de expressões de mapeamento no formato `$request.header.name`.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

QueryString

Converte as cadeias de caracteres de consulta fornecidas em uma lista de expressões de mapeamento no formato `$request.querystring.queryString`.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ReauthorizeEvery

O período time-to-live (TTL), em segundos, que especifica por quanto tempo o API Gateway armazena em cache os resultados do autorizador. Se você especificar um valor maior que 0, o API Gateway armazenará em cache as respostas do autorizador. O valor máximo é 3600, ou uma hora.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

StageVariables

Converte as variáveis de estágio fornecidas em uma lista de expressões de mapeamento no formato `$stageVariables.stageVariable`.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

LambdaRequestIdentity

Exemplo de identidade de solicitação Lambda

YAML

```
Identity:
  QueryStrings:
    - auth
  Headers:
    - Authorization
  StageVariables:
    - VARIABLE
  Context:
    - authcontext
  ReauthorizeEvery: 100
```

OAuth2Authorizer

Definição para um autorizador OAuth 2.0, também conhecido como autorizador JSON Web Token (JWT).

Para obter mais informações, consulte [Controle do acesso ao HTTP APIs com autorizadores JWT](#) no Guia do desenvolvedor do API Gateway.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
AuthorizationScopes: List
IdentitySource: String
JwtConfiguration: Map
```

Propriedades

AuthorizationScopes

Lista de escopos de autorização para esse autorizador.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IdentitySource

Expressão da fonte de identidade para esse autorizador.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

JwtConfiguration

Configuração do JWT para esse autorizador.

Isso é passado para a seção `jwtConfiguration` de um `x-amazon-apigateway-authorizer` na seção `securitySchemes` de uma definição de OpenAPI.

Note

As propriedades `issuer` e `audience` não diferenciam maiúsculas e minúsculas e podem ser usadas em minúsculas, como na OpenAPI, ou em maiúsculas, como em. `Issuer Audience` [AWS::ApiGatewayV2::Authorizer](#)

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

OAuth Autorizador 2.0

OAuth Exemplo de autorizador 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

Gerencie o compartilhamento de recursos de origem cruzada (CORS) para seu HTTP. APIs. Especifique o domínio a ser permitido como uma string ou especifique um dicionário com configuração adicional do Cors. NOTA: O Cors exige que o SAM modifique sua definição de OpenAPI, portanto, ele só funciona com OpenApi inline definido na propriedade. `DefinitionBody`

Para obter mais informações sobre o CORS, consulte [Configurando o CORS para uma API HTTP](#) no Guia do desenvolvedor do API Gateway.

Observação: se `HttpApiCorsConfiguration` estiver definido tanto na OpenAPI quanto no nível da propriedade, AWS SAM mescla-os com as propriedades que têm precedência.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
AllowCredentials: Boolean
AllowHeaders: List
AllowMethods: List
AllowOrigins: List
ExposeHeaders: List
MaxAge: Integer
```

Propriedades

AllowCredentials

Especifica se as credenciais estão incluídas na solicitação de CORS.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AllowHeaders

Representa uma coleção de cabeçalhos permitidos.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AllowMethods

Representa uma coleção de métodos HTTP permitidos.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AllowOrigins

Representa uma coleção de origens permitidas.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ExposeHeaders

Representa uma coleção de cabeçalhos expostos.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

MaxAge

Especifica o número de segundos que o navegador deve armazenar em cache os resultados da solicitação de simulação.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

HttpApiCorsConfiguration

Exemplo de configuração do Cors da API HTTP.

YAML

```
CorsConfiguration:
  AllowOrigins:
    - "https://example.com"
  AllowHeaders:
    - x-apigateway-header
  AllowMethods:
    - GET
  MaxAge: 600
  AllowCredentials: true
```

HttpApiDefinition

Um documento da OpenAPI que define a API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Bucket: String  
Key: String  
Version: String
```

Propriedades

Bucket

O nome do bucket do Amazon S3 no qual o arquivo do OpenAPI está armazenado.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Bucket](#) propriedade do tipo de `AWS::ApiGatewayV2::Api BodyS3Location` dados.

Key

A chave do Amazon S3 do arquivo OpenAPI.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Key](#) propriedade do tipo de `AWS::ApiGatewayV2::Api BodyS3Location` dados.

Version

Para objetos com controle de versão, a versão do arquivo OpenAPI.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Version](#) propriedade do tipo de `AWS::ApiGatewayV2::Api BodyS3Location` dados.

Exemplos

Exemplo de definição de Uri

Exemplo de definição de API

YAML

```
DefinitionUri:
  Bucket: sam-s3-demo-bucket-name
  Key: mykey-name
  Version: 121212
```

HttpApiDomainConfiguration

Configura um domínio personalizado para uma API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
BasePath: List
CertificateArn: String
DomainName: String
EndpointConfiguration: String
MutualTlsAuthentication: MutualTlsAuthentication
OwnershipVerificationCertificateArn: String
Route53: Route53Configuration
SecurityPolicy: String
```

Propriedades

BasePath

Uma lista dos caminhos básicos a serem configurados com o nome de domínio do Amazon API Gateway.

Tipo: lista

Obrigatório: não

Padrão: /

CloudFormation compatibilidade: essa propriedade é semelhante à [ApiMappingKey](#) propriedade de um `AWS::ApiGatewayV2::ApiMapping` recurso. AWS SAM cria vários `AWS::ApiGatewayV2::ApiMapping` recursos, um por valor especificado nessa propriedade.

CertificateArn

O Amazon Resource Name (ARN) de um certificado AWS gerenciado para o endpoint desse nome de domínio. Gerenciador de certificados da AWS é a única fonte compatível.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CertificateArn](#) propriedade de um `AWS::ApiGatewayV2::DomainName` `DomainNameConfiguration` recurso.

DomainName

O nome de domínio personalizado para a sua API Gateway API. Letras maiúsculas não são compatíveis.

AWS SAM gera um `AWS::ApiGatewayV2::DomainName` recurso quando essa propriedade é definida. Para obter informações sobre esse cenário, consulte [DomainName propriedade é especificada](#). Para obter informações sobre CloudFormation os recursos gerados, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DomainName](#) propriedade de um `AWS::ApiGatewayV2::DomainName` recurso.

EndpointConfiguration

Define o tipo de endpoint do API Gateway a ser mapeado para o domínio personalizado. O valor dessa propriedade determina como a `CertificateArn` propriedade é mapeada CloudFormation.

O único valor válido para HTTP APIs é `REGIONAL`.

Tipo: string

Obrigatório: não

Padrão: REGIONAL

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

MutualTlsAuthentication

A Transport Layer Security (TLS) mútua para um nome de domínio personalizado.

Digite: [MutualTlsAuthentication](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [MutualTlsAuthentication](#) propriedade de um `AWS::ApiGatewayV2::DomainName` recurso.

OwnershipVerificationCertificateArn

O ARN do certificado público emitido pelo ACM para validar a propriedade do domínio personalizado. Necessário somente para configurar o TLS mútuo e para especificar um ARN de CA privado ou importado do ACM para o `CertificateArn`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [OwnershipVerificationCertificateArn](#) propriedade do tipo de `AWS::ApiGatewayV2::DomainName` `DomainNameConfiguration` dados.

Route53

Define uma configuração do Amazon Route 53.

Tipo:: [Route53Configuration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SecurityPolicy

A versão da política de segurança da TLS para esse nome de domínio.

O único valor válido para HTTP APIs é TLS_1_2.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SecurityPolicy](#) propriedade do tipo de AWS::ApiGatewayV2::DomainName DomainNameConfiguration dados.

Exemplos

DomainName

DomainName exemplo

YAML

```
Domain:
  DomainName: www.example.com
  CertificateArn: arn-example
  EndpointConfiguration: REGIONAL
  Route53:
    HostedZoneId: Z1PA6795UKMFR9
  BasePath:
    - foo
    - bar
```

Route53Configuration

Configura os conjuntos de registros Route53 para uma API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
DistributionDomainName: String
```

```
EvaluateTargetHealth: Boolean  
HostedZoneId: String  
HostedZoneName: String  
IPv6: Boolean  
Region: String  
SetIdentifier: String
```

Propriedades

DistributionDomainName

Configura uma distribuição personalizada do nome de domínio personalizado da API.

Tipo: string

Obrigatório: não

Padrão: use a distribuição do API Gateway.

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DNSName](#) propriedade de um `AWS::Route53::RecordSetGroup` `AliasTarget` recurso.

Notas adicionais: O nome de domínio de uma [CloudFront distribuição](#).

EvaluateTargetHealth

Quando `EvaluateTargetHealth` verdadeiro, um registro de alias herda a integridade do AWS recurso referenciado, como um balanceador de carga do Elastic Load Balancing ou outro registro na zona hospedada.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EvaluateTargetHealth](#) propriedade de um `AWS::Route53::RecordSetGroup` `AliasTarget` recurso.

Observações adicionais: você não pode `EvaluateTargetHealth` definir como verdadeiro quando o destino do alias é uma CloudFront distribuição.

HostedZoneId

O ID da zona hospedada na qual você deseja criar registros.

Especifique `HostedZoneName` ou `HostedZoneId`, mas não ambos. Se houver várias zonas hospedadas com o mesmo nome de domínio, especifique a zona hospedada usando `HostedZoneId`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [HostedZoneId](#) propriedade de um `AWS::Route53::RecordSetGroup` `RecordSet` recurso.

HostedZoneName

O nome da zona hospedada na qual você deseja criar registros. Você deve incluir um ponto final (por exemplo, `www.example.com.`) como parte do `HostedZoneName`.

Especifique `HostedZoneName` ou `HostedZoneId`, mas não ambos. Se houver várias zonas hospedadas com o mesmo nome de domínio, especifique a zona hospedada usando `HostedZoneId`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [HostedZoneName](#) propriedade de um `AWS::Route53::RecordSetGroup` `RecordSet` recurso.

IPv6

Quando essa propriedade é definida, AWS SAM cria um `AWS::Route53::RecordSet` recurso e define [Type](#) como AAAA para o fornecido `HostedZone`.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Region

Somente conjuntos de registros de recursos baseados em latência: a EC2 região amazônica onde você criou o recurso ao qual esse conjunto de registros de recursos se refere. O recurso

normalmente é um AWS recurso, como uma EC2 instância ou um balanceador de carga ELB, e é referido por um endereço IP ou nome de domínio DNS, dependendo do tipo de registro.

Quando o Amazon Route 53 recebe uma consulta de DNS para um nome e tipo de domínio para o qual você criou conjuntos de registros de recursos de latência, o Route 53 seleciona o conjunto de registros de recursos de latência que tem a menor latência entre o usuário final e a região da Amazon associada. EC2 O Route 53 retorna o valor associado ao conjunto de registros de recursos selecionado.

Observe o seguinte:

- Só é possível especificar um `ResourceRecord` por conjunto de registros de recursos de latência.
- Você só pode criar um conjunto de registros de recursos de latência para cada EC2 região da Amazon.
- Você não precisa criar conjuntos de registros de recursos de latência para todas as EC2 regiões da Amazon. O Route 53 selecionará a região com a melhor latência entre as regiões para as quais você cria conjuntos de registros de recursos de latência.
- Não é possível criar conjuntos de registros de recursos que não sejam de latência e tenham os mesmos valores para os elementos `Name` e `Type` como conjuntos registros de recursos de latência.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Region](#) propriedade de um tipo de AWS : :Route53 : :RecordSetGroup RecordSet dados.

SetIdentifier

Conjuntos de registros de recursos que têm uma política de roteamento diferente da simples: um identificador que diferencia entre vários conjuntos de registros de recursos que têm a mesma combinação de nome e tipo, como vários conjuntos de registros de recursos ponderados chamados `acme.example.com` que tenham um tipo de A. Em um grupo de conjuntos de registros de recursos que têm o mesmo nome e tipo, o valor de `SetIdentifier` deve ser exclusivo para cada conjunto de registros de recursos.

Para obter informações sobre políticas de roteamento, consulte [Escolher uma política de roteamento](#) no Guia do desenvolvedor do Amazon Route 53.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SetIdentifier](#) propriedade de um tipo de `AWS::Route53::RecordSetGroup` `RecordSet` dados.

Exemplos

Exemplo de configuração do Route 53

Este exemplo mostra como configurar o Route 53.

YAML

```
Domain:
  DomainName: www.example.com
  CertificateArn: arn-example
  EndpointConfiguration: EDGE
  Route53:
    HostedZoneId: Z1PA6795UKMFR9
    EvaluateTargetHealth: true
    DistributionDomainName: xyz
```

AWS::Serverless::WebSocketApi

Cria uma API do Amazon WebSocket API Gateway, que permite criar aplicativos de comunicação interativa bidirecional. WebSocket APIs permitem que o servidor envie mensagens aos clientes sem que o cliente precise solicitá-las. Para obter mais informações, consulte Como [trabalhar com WebSocket APIs](#) no Guia do desenvolvedor do API Gateway.

Recomendamos que você use CloudFormation ganchos ou políticas do IAM para verificar se os recursos do API Gateway têm autorizadores vinculados a eles para controlar o acesso a eles.

[Para obter mais informações sobre o uso de CloudFormation ganchos, consulte Registrando ganchos no guia do usuário da CloudFormation CLI e no repositório apigw-enforce-authorizer.](#)

GitHub

Para obter mais informações sobre o uso de políticas do IAM, consulte [Exigir que as rotas de API tenham autorização](#) no Guia do desenvolvedor do API Gateway.

 Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Type: AWS::Serverless::WebSocketApi
Properties:
  ApiKeySelectionExpression: String
  AccessLogSettings: AccessLogSettings
  Auth: WebSocketApiAuth
  DefaultRouteSettings: RouteSettings
  Description: String
  DisableExecuteApiEndpoint: Boolean
  DisableSchemaValidation: Boolean
  Domain: WebSocketApiDomainConfiguration
  IpAddressType: String
  Name: String
  PropagateTags: Boolean
  Routes: RouteConfiguration
  RouteSelectionExpression: String
  RouteSettings: RouteSettings
  StageName: String
  StageVariables: Json
  Tags: Map
```

Propriedades

ApiKeySelectionExpression

Uma expressão de seleção de chaves de API. Para obter mais informações, consulte [Expressões de seleção de chaves de API](#) no Guia do desenvolvedor do API Gateway.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ApiKeySelectionExpression](#) propriedade de um `AWS::ApiGatewayV2::Api` recurso.

AccessLogSettings

As configurações para o registro em log de acesso em um estágio.

Digite: [AccessLogSettings](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AccessLogSettings](#) propriedade de um `AWS::ApiGatewayV2::Stage` recurso.

Auth

Configura a autorização para controlar o acesso à sua WebSocket API. A autorização é aplicada à `$connect` rota.

Para obter mais informações, consulte Como [controlar o acesso às WebSocket APIs](#) no Guia do desenvolvedor do API Gateway.

Digite: [WebSocketApiAuth](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

DefaultRouteSettings

As configurações de rota padrão para essa WebSocket API. Essas configurações se aplicam a todas as rotas, a menos que sejam substituídas pela propriedade `RouteSettings` para determinadas rotas.

Digite: [RouteSettings](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DefaultRouteSettings](#) propriedade de um `AWS::ApiGatewayV2::Stage` recurso.

Description

Uma descrição da WebSocket API.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::ApiGatewayV2::Api` recurso.

DisableExecuteApiEndpoint

Especifica se os clientes podem invocar sua API usando o endpoint `execute-api` padrão. Para exigir que os clientes usem um nome de domínio personalizado para invocar sua API, desabilite o endpoint padrão.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DisableExecuteApiEndpoint](#) propriedade de um `AWS::ApiGatewayV2::Api` recurso.

DisableSchemaValidation

Evite validar modelos ao criar uma implantação.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DisableSchemaValidation](#) propriedade de um `AWS::ApiGatewayV2::Api` recurso.

Domain

Configura um domínio personalizado para essa WebSocket API.

Note

WebSocket APIs não oferecem suporte à autenticação TLS mútua (MTLS). Se você especificar `MutualTlsAuthentication` ou `OwnershipVerificationCertificateArn`, AWS SAM retornará um erro.

Digite: [WebSocketApiDomainConfiguration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IpAddressType

O tipo de endereço IP da API. Os valores válidos são somente `ipv4` para IPv4 e `dualstack` para IPv4 e IPv6.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [IpAddressType](#) propriedade de um `AWS::ApiGatewayV2::Api` recurso.

Name

Um nome para a WebSocket API. Se você não especificar um nome, AWS SAM gera um nome para você.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::ApiGatewayV2::Api` recurso.

PropagateTags

Se `true`, AWS SAM adiciona a `Tags` propriedade aos `AWS::ApiGatewayV2::DomainName` recursos `AWS::ApiGatewayV2::Stage` e AWS SAM gerados.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Routes

As configurações de rota para essa WebSocket API. As rotas definem como as mensagens são roteadas para as Funções de Eventos Lambda. Cada rota consiste em uma chave de rota e um ARN da função Lambda.

WebSocket As APIs oferecem suporte a três rotas predefinidas: `$connect`, `$disconnect`, e `$default`. Você também pode definir rotas personalizadas.

Digite: [RouteConfiguration](#)

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

RouteSelectionExpression

A expressão de seleção de rotas para a WebSocket API. Para obter mais informações, consulte [Expressões de seleção de rotas](#) no Guia do desenvolvedor do API Gateway.

Um valor comum é `$request.body.action`, que roteia mensagens com base em um `action` campo no corpo da mensagem.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RouteSelectionExpression](#) propriedade de um `AWS::ApiGatewayV2::Api` recurso.

RouteSettings

As configurações de rota para essa WebSocket API. Essas configurações substituem `DefaultRouteSettings` as rotas específicas.

Digite: [RouteSettings](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RouteSettings](#) propriedade de um `AWS::ApiGatewayV2::Stage` recurso.

StageName

O nome do estágio da API. Se você não especificar um nome, AWS SAM use `default` como nome artístico.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StageName](#) propriedade de um `AWS::ApiGatewayV2::Stage` recurso.

StageVariables

Um mapa que define as variáveis da etapa. Os nomes das variáveis podem ter caracteres alfanuméricos e sublinhados, e os valores devem corresponder `[A-Za-z0-9-._~:/?#&=, ]+`.

Type: [Json](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StageVariables](#) propriedade de um `AWS::ApiGatewayV2::Stage` recurso.

Tags

Um mapa (string a string) que especifica as tags a serem adicionadas a essa WebSocket API. Para obter detalhes sobre chaves e valores válidos para tags, consulte [Etiqueta de recurso](#) no CloudFormation Guia do usuário.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Valores de retorno

Ref.

Quando você transmite o ID lógico desse recurso para a função intrínseca `Ref`, `Ref` retorna o ID da API do recurso `AWS::ApiGatewayV2::Api` subjacente, como `a1bcdef2gh`.

Para obter mais informações sobre como usar a função `Ref`, consulte [Ref](#) no Guia do usuário do AWS CloudFormation .

Exemplos

WebSocket API simples

O exemplo a seguir cria uma WebSocket API com três rotas.

Resources:**MyWebSocketApi:**

Type: AWS::Serverless::WebSocketApi

Properties:

RouteSelectionExpression: \$request.body.action

Routes:**\$connect:**

FunctionArn: !GetAtt ConnectFunction.Arn

\$disconnect:

FunctionArn: !GetAtt DisconnectFunction.Arn

sendMessage:

FunctionArn: !GetAtt SendMessageFunction.Arn

ConnectFunction:

Type: AWS::Serverless::Function

Properties:

Handler: index.connect

Runtime: nodejs20.x

CodeUri: ./src

DisconnectFunction:

Type: AWS::Serverless::Function

Properties:

Handler: index.disconnect

Runtime: nodejs20.x

CodeUri: ./src

SendMessageFunction:

Type: AWS::Serverless::Function

Properties:

Handler: index.sendMessage

Runtime: nodejs20.x

CodeUri: ./src

WebSocket API com Lambda Authorizer

O exemplo a seguir cria uma WebSocket API com um autorizador Lambda.

Resources:**MyWebSocketApi:**

Type: AWS::Serverless::WebSocketApi

Properties:

RouteSelectionExpression: \$request.body.action


```
Auth:
  AuthType: CUSTOM
  AuthArn: !GetAtt AuthorizerFunction.Arn
  IdentitySource:
    - route.request.header.Authorization
Routes:
  $connect:
    FunctionArn: !GetAtt ConnectFunction.Arn
  sendMessage:
    FunctionArn: !GetAtt SendMessageFunction.Arn
```

```
AuthorizerFunction:
  Type: AWS::Serverless::Function
  Properties:
    Handler: index.authorize
    Runtime: nodejs20.x
    CodeUri: ./src
```

```
ConnectFunction:
  Type: AWS::Serverless::Function
  Properties:
    Handler: index.connect
    Runtime: nodejs20.x
    CodeUri: ./src
```

```
SendMessageFunction:
  Type: AWS::Serverless::Function
  Properties:
    Handler: index.sendMessage
    Runtime: nodejs20.x
    CodeUri: ./src
```

WebSocketApiAuth

Configure a autorização para controlar o acesso à sua API do Amazon WebSocket API Gateway.

Para obter mais informações sobre como configurar o acesso às WebSocket APIs, consulte [Como controlar o acesso às WebSocket APIs no Guia do desenvolvedor](#) do API Gateway.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
AuthArn: String
AuthType: String
IdentitySource: List
InvokeRole: String
Name: String
```

Propriedades

AuthArn

O ARN da função de evento Lambda a ser usada para autorização. Exigido quando AuthType é CUSTOM.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AuthType

O tipo de autorização. Os valores válidos são NONE, AWS_IAM ou CUSTOM.

- NONE- Sem autorização
- AWS_IAM- Autorização do IAM
- CUSTOM- Autorizador Lambda

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IdentitySource

A origem de identidade para a qual a autorização é solicitada. Por exemplo, `.route.request.header.Authorization`

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [IdentitySource](#) propriedade de um `AWS::ApiGatewayV2::Authorizer` recurso.

InvokeRole

O ARN da função do IAM que o API Gateway assume ao invocar a função autorizadora. Se não for especificado, cria AWS SAM automaticamente uma permissão baseada em recursos que permite que o API Gateway invoque a função do autorizador.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AuthorizerCredentialsArn](#) propriedade de um `AWS::ApiGatewayV2::Authorizer` recurso.

Name

O nome do autorizador.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::ApiGatewayV2::Authorizer` recurso.

Exemplos

Autorizador Lambda

O exemplo a seguir configura um autorizador Lambda para uma API. WebSocket

```
Auth:
  AuthType: CUSTOM
  AuthArn: !GetAtt AuthorizerFunction.Arn
  IdentitySource:
    - route.request.header.Authorization
```

Autorização do IAM

O exemplo a seguir configura a autorização do IAM para uma WebSocket API.

```
Auth:  
  AuthType: AWS_IAM
```

WebSocketApiDomainConfiguration

Configura um domínio personalizado para uma WebSocket API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
BasePath: List  
CertificateArn: String  
DomainName: String  
EndpointConfiguration: String  
Route53: Route53Configuration  
SecurityPolicy: String
```

Propriedades

BasePath

Uma lista dos caminhos básicos a serem configurados com o nome de domínio do Amazon API Gateway.

Tipo: lista

Obrigatório: não

Padrão: /

CloudFormation compatibilidade: essa propriedade é semelhante à [ApiMappingKey](#) propriedade de um `AWS::ApiGatewayV2::ApiMapping` recurso. AWS SAM cria vários `AWS::ApiGatewayV2::ApiMapping` recursos, um por valor especificado nessa propriedade.

CertificateArn

O Amazon Resource Name (ARN) de um certificado AWS gerenciado para o endpoint desse nome de domínio. Gerenciador de certificados da AWS é a única fonte compatível.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CertificateArn](#) propriedade de um `AWS::ApiGatewayV2::DomainName` `DomainNameConfiguration` recurso.

DomainName

O nome de domínio personalizado para sua API do WebSocket API Gateway. Letras maiúsculas não são compatíveis.

AWS SAM gera um `AWS::ApiGatewayV2::DomainName` recurso quando essa propriedade é definida. Para obter informações sobre CloudFormation os recursos gerados, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DomainName](#) propriedade de um `AWS::ApiGatewayV2::DomainName` recurso.

EndpointConfiguration

Define o tipo de endpoint do API Gateway a ser mapeado para o domínio personalizado. O valor dessa propriedade determina como a `CertificateArn` propriedade é mapeada CloudFormation.

O único valor válido para WebSocket APIs é `REGIONAL`.

Tipo: string

Obrigatório: não

Padrão: `REGIONAL`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Route53

Define uma configuração do Route 53.

Tipo:: [Route53Configuration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SecurityPolicy

A versão da política de segurança da TLS para esse nome de domínio.

O único valor válido para WebSocket APIs é TLS_1_2.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SecurityPolicy](#) propriedade do tipo de AWS::ApiGatewayV2::DomainNameDomainNameConfiguration dados.

Exemplos

DomainName

DomainName exemplo

YAML

```
Domain:
  DomainName: ws.example.com
  CertificateArn: arn:aws:acm:us-east-1:123456789012:certificate/example
  EndpointConfiguration: REGIONAL
  Route53:
    HostedZoneId: Z1PA6795UKMFR9
  BasePath:
    - v1
```

- v2

Route53Configuration

Configura os conjuntos de registros Route53 para uma API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
EvaluateTargetHealth: Boolean  
HostedZoneId: String  
HostedZoneName: String  
IPv6: Boolean  
Region: String  
SetIdentifier: String
```

Propriedades

EvaluateTargetHealth

Quando EvaluateTargetHealth verdadeiro, um registro de alias herda a integridade do AWS recurso referenciado, como um balanceador de carga do Elastic Load Balancing ou outro registro na zona hospedada.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EvaluateTargetHealth](#) propriedade de um `AWS::Route53::RecordSetGroup` AliasTarget recurso.

Observações adicionais: você não pode EvaluateTargetHealth definir como verdadeiro quando o destino do alias é uma CloudFront distribuição.

HostedZoneId

O ID da zona hospedada na qual você deseja criar registros.

Especifique `HostedZoneName` ou `HostedZoneId`, mas não ambos. Se houver várias zonas hospedadas com o mesmo nome de domínio, especifique a zona hospedada usando `HostedZoneId`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [HostedZoneId](#) propriedade de um `AWS::Route53::RecordSetGroup` `RecordSet` recurso.

HostedZoneName

O nome da zona hospedada na qual você deseja criar registros. Você deve incluir um ponto final (por exemplo, `www.example.com.`) como parte do `HostedZoneName`.

Especifique `HostedZoneName` ou `HostedZoneId`, mas não ambos. Se houver várias zonas hospedadas com o mesmo nome de domínio, especifique a zona hospedada usando `HostedZoneId`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [HostedZoneName](#) propriedade de um `AWS::Route53::RecordSetGroup` `RecordSet` recurso.

IPv6

Quando essa propriedade é definida, AWS SAM cria um `AWS::Route53::RecordSet` recurso e define [Type](#) como AAAA para o fornecido `HostedZone`.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Region

Somente conjuntos de registros de recursos baseados em latência: a região do Amazon EC2 na qual você criou o recurso ao qual esse conjunto de registros de recursos se refere. O recurso

normalmente é um AWS recurso, como uma instância do EC2 ou um balanceador de carga do ELB, e é referido por um endereço IP ou nome de domínio DNS, dependendo do tipo de registro.

Quando o Amazon Route 53 recebe uma consulta de DNS do nome e tipo de domínio para os quais você criou conjuntos de registros de recursos de latência, o Route 53 seleciona o conjunto de registros de recurso de latência que tem a mais baixa latência entre o usuário final e a região do Amazon EC2 associada. O Route 53 retorna o valor associado ao conjunto de registros de recursos selecionado.

Observe o seguinte:

- Só é possível especificar um `ResourceRecord` por conjunto de registros de recursos de latência.
- Só é possível criar um conjunto de registros de recursos de latência para cada região do Amazon EC2.
- Não é necessário criar conjuntos de registros de recursos de latência para todas as regiões do Amazon EC2. O Route 53 selecionará a região com a melhor latência entre as regiões para as quais você cria conjuntos de registros de recursos de latência.
- Não é possível criar conjuntos de registros de recursos que não sejam de latência e tenham os mesmos valores para os elementos `Name` e `Type` como conjuntos registros de recursos de latência.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Region](#) propriedade de um tipo de AWS :: Route53 :: RecordSetGroup RecordSet dados.

SetIdentifier

Conjuntos de registros de recursos que têm uma política de roteamento diferente da simples: um identificador que diferencia entre vários conjuntos de registros de recursos que têm a mesma combinação de nome e tipo, como vários conjuntos de registros de recursos ponderados chamados `acme.example.com` que tenham um tipo de A. Em um grupo de conjuntos de registros de recursos que têm o mesmo nome e tipo, o valor de `SetIdentifier` deve ser exclusivo para cada conjunto de registros de recursos.

Para obter informações sobre políticas de roteamento, consulte [Escolher uma política de roteamento](#) no Guia do desenvolvedor do Amazon Route 53.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SetIdentifier](#) propriedade de um tipo de `AWS::Route53::RecordSetGroup` RecordSet dados.

Exemplos

Exemplo de configuração do Route 53

Este exemplo mostra como configurar o Route 53.

YAML

```
Domain:
  DomainName: www.example.com
  CertificateArn: arn-example
  EndpointConfiguration: REGIONAL
  Route53:
    HostedZoneId: Z1PA6795UKMFR9
    EvaluateTargetHealth: true
```

RouteConfiguration

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
ApiKeyRequired: Boolean
FunctionArn: String
IntegrationTimeout: Integer
ModelSelectionExpression: String
OperationName: String
RequestModels: Json
RequestParameters: Json
RouteResponseSelectionExpression: String
```

Propriedades

ApiKeyRequired

Especifica se uma chave de API é necessária para essa rota.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ApiKeyRequired](#) propriedade de um `AWS::ApiGatewayV2::Route` recurso.

FunctionArn

O ARN da função de evento Lambda que gerencia essa rota. AWS SAM cria uma integração e as permissões necessárias para que o API Gateway invoque a função.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntegrationTimeout

O tempo limite para a integração, em milissegundos. O valor máximo é 29.000 milissegundos (29 segundos).

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TimeoutInMillis](#) propriedade de um `AWS::ApiGatewayV2::Integration` recurso.

ModelSelectionExpression

A expressão de seleção de modelo para a rota.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ModelSelectionExpression](#) propriedade de um `AWS::ApiGatewayV2::Route` recurso.

OperationName

O nome da operação para a rota.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [OperationName](#) propriedade de um `AWS::ApiGatewayV2::Route` recurso.

RequestModels

Os modelos de solicitação para a rota.

Tipo: Json

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RequestModels](#) propriedade de um `AWS::ApiGatewayV2::Route` recurso.

RequestParameters

Os parâmetros de solicitação para a rota.

Tipo: Json

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RequestParameters](#) propriedade de um `AWS::ApiGatewayV2::Route` recurso.

RouteResponseSelectionExpression

A expressão de seleção de resposta de rota para a rota.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RouteResponseSelectionExpression](#) propriedade de um `AWS::ApiGatewayV2::Route` recurso.

Exemplos

Rota simples

O exemplo a seguir configura uma rota simples.

```
Routes:
  $connect:
    FunctionArn: !GetAtt ConnectFunction.Arn
  sendMessage:
    FunctionArn: !GetAtt SendMessageFunction.Arn
    OperationName: SendMessage
```

AWS::Serverless::LayerVersion

Cria um Lambda LayerVersion que contém a biblioteca ou o código de tempo de execução necessário para uma função de evento do Lambda.

O [AWS::Serverless::LayerVersion](#) recurso também suporta o atributo `Metadata resource`, para que você possa AWS SAM instruir a criar camadas incluídas em seu aplicativo. Para obter mais informações sobre a criação de camadas, consulte [Construindo camadas Lambda em AWS SAM](#).

Nota importante: Desde o lançamento do atributo de [UpdateReplacePolicy](#) recurso em CloudFormation, [AWS::Lambda::LayerVersion](#) (recomendado) oferece os mesmos benefícios que [AWS::Serverless::LayerVersion](#).

Quando um Serverless LayerVersion é transformado, o SAM também transforma o ID lógico do recurso para que o antigo não seja excluído automaticamente CloudFormation quando o recurso LayerVersions for atualizado.

Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Type: AWS::Serverless::LayerVersion
Properties:
  CompatibleArchitectures: List
  CompatibleRuntimes: List
  ContentUri: String | LayerContent
  Description: String
  LayerName: String
  LicenseInfo: String
  PublishLambdaVersion: Boolean
  RetentionPolicy: String
```

Propriedades

CompatibleArchitectures

Especifica as arquiteturas de conjunto de instruções suportadas para a versão da camada.

Para obter mais informações sobre esta propriedade, consulte [Arquiteturas de conjuntos de instruções do Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Valores válidos: x86_64, arm64

Tipo: lista

Obrigatório: não

Padrão: x86_64

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CompatibleArchitectures](#) propriedade de um `AWS::Lambda::LayerVersion` recurso.

CompatibleRuntimes

Lista de tempos de execução compatíveis com isso LayerVersion.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CompatibleRuntimes](#) propriedade de um `AWS::Lambda::LayerVersion` recurso.

ContentUri

Uri do Amazon S3, caminho para a pasta local ou LayerContent objeto do código da camada.

Se um Uri ou LayerContent objeto do Amazon S3 for fornecido, o objeto do Amazon S3 referenciado deverá ser um arquivo ZIP válido que contenha o conteúdo de uma camada Lambda.

Se for fornecido um caminho para uma pasta local, para que o conteúdo seja transformado corretamente, o modelo deverá passar pelo fluxo de trabalho que inclui [sam build](#) seguido por [sam deploy](#) ou [sam package](#). Por padrão, os caminhos relativos são resolvidos com relação à localização do AWS SAM modelo.

Tipo: String | [LayerContent](#)

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é semelhante à [Content](#) propriedade de um `AWS::Lambda::LayerVersion` recurso. As propriedades aninhadas do Amazon S3 têm nomes diferentes.

Description

Descrição dessa camada.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::Lambda::LayerVersion` recurso.

LayerName

O nome ou o nome de recurso da Amazon (ARN) da camada.

Tipo: string

Obrigatório: não

Padrão: ID lógica do recurso

CloudFormation compatibilidade: essa propriedade é semelhante à [LayerName](#) propriedade de um `AWS::Lambda::LayerVersion` recurso. Caso você não especifique um nome, o ID lógico do recurso será usado como nome.

LicenseInfo

Informações sobre a licença para isso `LayerVersion`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [LicenseInfo](#) propriedade de um `AWS::Lambda::LayerVersion` recurso.

PublishLambdaVersion

Uma propriedade opcional que cria uma nova versão do Lambda sempre que há uma alteração no recurso referenciado `LayerVersion`. Quando habilitado com `AutoPublishAlias` e `AutoPublishAliasAllProperties` na função do Lambda conectada, haverá uma nova versão do Lambda criada para cada alteração feita no recurso `LayerVersion`.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

RetentionPolicy

Essa propriedade especifica se versões antigas da `LayerVersion` são retidas ou excluídas quando você exclui um recurso. Se você precisar reter versões antigas do `LayerVersion` quando atualizar ou substituir um recurso, deverá ter o atributo `UpdateReplacePolicy` habilitado. Para obter informações sobre como fazer isso, consulte [atributo UpdateReplacePolicy](#) no Guia do usuário do AWS CloudFormation .

Valores válidos: `Retain` ou `Delete`

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Observações adicionais: Quando você especifica `Retain`, AWS SAM adiciona um [Atributos de recursos suportados por AWS SAM](#) de `DeletionPolicy`: `Retain` ao `AWS::Lambda::LayerVersion` recurso transformado.

Valores de retorno

Ref.

Quando a ID lógica desse recurso é fornecida à função `Ref` intrínseca, ela retorna o ARN do recurso do Lambda subjacente. `LayerVersion`

Para obter mais informações sobre como usar a função `Ref`, consulte [Ref](#) no Guia do usuário do AWS CloudFormation .

Exemplos

`LayerVersionExample`

Exemplo de um `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

Um arquivo ZIP que contém o conteúdo de uma [camada do Lambda](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Bucket: String  
Key: String  
StorageMode: String  
Version: String
```

Propriedades

Bucket

O bucket do Amazon S3 do arquivamento de camada.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [S3Bucket](#) propriedade do tipo de `AWS::Lambda::LayerVersion` Content dados.

Key

A chave do Amazon S3 do arquivamento de camada.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [S3Key](#) propriedade do tipo de `AWS::Lambda::LayerVersion` Content dados.

StorageMode

Controla como o Lambda armazena o arquivo de camadas.

- **COPY**— O Lambda carrega uma cópia do seu arquivo de camadas para o Lambda-managed armazenamento. Você pode excluir o objeto Amazon S3 depois que o Lambda criar a versão da camada.

- **REFERENCE**— O Lambda faz referência ao arquivo de camadas do seu bucket do Amazon S3 e não armazena uma cópia. O objeto deve permanecer no lugar e o Lambda deve manter o acesso a ele durante toda a vida útil da versão da camada.

Para usar **REFERENCE**, habilite o controle de versão em seu bucket do Amazon S3 e conceda ao serviço Lambda acesso principal ao objeto. Para obter mais informações, consulte Armazenamento de código do [Self-managed Amazon S3](#) no Guia do AWS Lambda desenvolvedor.

Tipo: string

Valores válidos: COPY | REFERENCE

Obrigatório: não

Padrão: COPY

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [S3ObjectStorageMode](#) propriedade do tipo de `AWS::Lambda::LayerVersion` Content dados.

Version

Para objetos com controle de versão, a versão do objeto de arquivamento de camada a ser usada.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [S3ObjectVersion](#) propriedade do tipo de `AWS::Lambda::LayerVersion` Content dados.

Exemplos

LayerContent

Exemplo de conteúdo de camada

YAML

```
LayerContent:
  Bucket: amzn-s3-demo-bucket-name
```

```
Key: mykey-name
Version: 121212
```

Self-managed Armazenamento de código do Amazon S3

O texto a seguir `LayerContent` faz referência ao arquivo de camadas do seu próprio bucket do Amazon S3 em vez de copiá-lo para o Lambda-managed armazenamento.

YAML

```
LayerContent:
  Bucket: amzn-s3-demo-bucket-name
  Key: mykey-name
  Version: 121212
  StorageMode: REFERENCE
```

AWS::Serverless::MicrovmImage

Cria uma imagem de AWS Lambda microVM a partir de um artefato de código armazenado no Amazon S3. Uma imagem de microVM é um Firecracker instantâneo criado a partir de um código de aplicativo, permitindo a inicialização em menos de um Dockerfile segundo para instâncias de microVM.

Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Type: AWS::Serverless::MicrovmImage
Properties:
  AdditionalOsCapabilities: List
  BaseImageArn: String
```

```
BaseImageVersion: String  
BuildRoleArn: String  
CodeUri: String  
CpuConfigurations: List  
Description: String  
EgressNetworkConnectors: List  
EnvironmentVariables: Map  
Hooks: Hooks  
Logging: Logging  
Name: String  
PropagateTags: Boolean  
Resources: List  
Tags: Map
```

Propriedades

`AdditionalOsCapabilities`

Recursos adicionais do sistema operacional para conceder ao ambiente de tempo de execução do microVM. Se você não especificar um valor, o serviço não concederá recursos adicionais do sistema operacional.

Valores válidos: ALL

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AdditionalOsCapabilities](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso.

`BaseImageArn`

O ARN da imagem base da microVM a partir da qual criar.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BaseImageArn](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso.

`BaseImageVersion`

A versão da imagem base do microVM a ser usada.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BaseImageVersion](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso.

BuildRoleArn

O ARN da função do IAM que o serviço de compilação MicroVM assume para baixar seu artefato de código do Amazon S3 e gravar registros de construção.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [BuildRoleArn](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso. Isso é necessário no CloudFormation, mas não no AWS SAM. Se você não especificar uma função, AWS SAM cria uma com uma ID lógica de `<resource-logical-id>BuildRole`. Se seu artefato de código estiver criptografado com uma chave gerenciada pelo cliente AWS KMS, você deverá fornecer sua própria função com a `kms:Decrypt` permissão apropriada.

CodeUri

O URI do Amazon S3 do artefato zip contendo o código do aplicativo Dockerfile e.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é transformada na [CodeArtifact](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso. AWS SAM envolve o URI na `CodeArtifact.Uri` estrutura.

CpuConfigurations

Uma lista de [CpuConfiguration](#) objetos que especificam as arquiteturas de CPU suportadas pela microVM. Se você não especificar um valor, o serviço usará a arquitetura de CPU padrão.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [CpuConfigurations](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso.

Description

Uma descrição da imagem do microVM.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso.

EgressNetworkConnectors

A lista de ARNs de conectores de rede de saída disponíveis para a microVM em tempo de execução.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EgressNetworkConnectors](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso.

EnvironmentVariables

Variáveis de ambiente definidas no ambiente de execução do microVM.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [EnvironmentVariables](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso. Em AWS SAM, especifique como um mapa (por exemplo, `KEY: value`). AWS SAM converte para o formato CloudFormation `[{Key: KEY, Value: value}]` de matriz.

Hooks

Configuração de gancho de ciclo de vida para imagens de microVMS e microVM.

Tipo: [Ganchos](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Hooks](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso.

Logging

Configuração para saída de registro de microVM. Especifique exatamente um: CloudWatch ativar o CloudWatch registro ou Disabled desativar o registro.

Type: [Logging](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Logging](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso.

Name

O nome da imagem do microVM.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso.

PropagateTags

Especifica se as tags da Tags propriedade devem ser passadas para os recursos gerados, como a função de criação gerada automaticamente.

Tipo: booleano

Obrigatório: não

Padrão: False

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Resources

Uma lista de objetos de [recursos](#) que especificam os requisitos de recursos para a microVM.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Resources](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso.

Tags

Um mapa de pares de valores-chave que especifica as tags adicionadas a essa imagem de microVM.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Tags](#) propriedade de um `AWS::Lambda::MicrovmImage` recurso. A `Tags` propriedade in AWS SAM consiste em pares de valores-chave (enquanto CloudFormation nessa propriedade consiste em uma lista de Tag objetos). Além disso, adiciona AWS SAM automaticamente uma `lambda:createdBy: SAM` tag a essa imagem de microVM e às funções padrão que são geradas para ela.

Valores de retorno

Ref.

Quando a ID lógica desse recurso é fornecida à função `Ref` intrínseca, ela retorna o ARN da imagem da microVM.

Para obter mais informações sobre como usar a função `Ref`, consulte [Ref](#) no Guia do usuário do AWS CloudFormation .

Fn:: GetAtt

`Fn::GetAtt` retorna um valor para um atributo especificado deste tipo. Estes são os atributos disponíveis e os valores de retorno de amostra.

Para obter mais informações sobre o uso do `Fn::GetAtt`, consulte [Fn::GetAtt](#) o AWS CloudFormation Guia do usuário.

CreatedAt

O carimbo de data e hora em que a versão da imagem microVM foi criada.

ImageArn

O ARN da imagem microVM.

LatestActiveImageVersion

A versão ativa mais recente da imagem microVM.

LatestFailedImageVersion

A última versão com falha da imagem do microVM, se houver.

State

O estado atual da imagem do microVM.

UpdatedAt

O carimbo de data e hora em que a versão da imagem do microVM foi atualizada pela última vez.

Exemplos

Imagem mínima de microVM

O exemplo a seguir cria uma imagem de microVM com somente as propriedades necessárias. AWS SAM gera automaticamente uma função de construção.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31

Resources:
  MyMicroVM:
    Type: AWS::Serverless::MicrovmImage
    Properties:
      Name: my-app
      CodeUri: s3://my-bucket/app.zip
      BaseImageArn: arn:aws:lambda:us-east-2:aws:microvm-image:al2023-1
      BaseImageVersion: "0"
```

Configuração completa da imagem microVM

O exemplo a seguir cria uma imagem de microVM com todas as propriedades disponíveis configuradas.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
```

Resources:**MyMicroVM:**

Type: AWS::Serverless::MicrovmImage

Properties:

Name: my-app

CodeUri: s3://my-bucket/app.zip

BaseImageArn: arn:aws:lambda:us-east-2:aws:microvm-image:al2023-1

BaseImageVersion: "0"

Description: "Production MicroVM application"

CpuConfigurations:

- Architecture: ARM_64

Resources:

- MinimumMemoryInMiB: 1024

AdditionalOsCapabilities:

- ALL

EgressNetworkConnectors:

- arn:aws:lambda:us-east-2:aws:network-connector:aws-network-

connector:INTERNET_EGRESS

Hooks:

Port: 9000

MicrovmHooks:

Run: ENABLED

RunTimeoutInSeconds: 30

Suspend: ENABLED

Resume: ENABLED

MicrovmImageHooks:

Ready: ENABLED

ReadyTimeoutInSeconds: 60

EnvironmentVariables:

LOG_LEVEL: info

APP_ENV: production

Logging:**CloudWatch:**

LogGroup: /aws/lambda-microvms/my-app

Tags:

Environment: Production

Team: Platform

AWS::Serverless::NetworkConnector

Cria um conector AWS Lambda de rede que define a conectividade de rede para recursos de computação Lambda. Um conector de rede especifica como os recursos computacionais se conectam às redes, por exemplo, roteando o tráfego de saída por meio de uma VPC do

cliente usando interfaces de rede elásticas (ENIs) que o serviço provisiona em sub-redes e grupos de segurança especificados. Cada conector serve para um tipo de computação, microVMS ou funções da Web. Para usar um conector com [AWS::Serverless:: WebFunction](#), `ComputeTypes` defina `WebFunction` e transmita o conector `LatestVersionArn` para a `EgressNetworkConnectorArn` propriedade da função.

Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Type: AWS::Serverless::NetworkConnector
Properties:
  ComputeTypes: List
  Name: String
  OperatorRole: String
  PropagateTags: Boolean
  Tags: Map
  VpcConfig: VpcConfig
```

Propriedades

ComputeTypes

O tipo de recurso de computação Lambda que pode usar esse conector de rede. Especifique exatamente um dos seguintes valores:

- `MicroVm`: o conector é usado pelos recursos computacionais do MicroVM.
- `WebFunction`: O conector é usado por [AWS::Serverless:: WebFunction](#) recursos por meio de suas `EgressNetworkConnectorArn` propriedades.

Tipo: lista

Obrigatório: não

Padrão: [MicroVm]

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Configuration.VpcEgressConfiguration.AssociatedComputeResourceTypes](#) propriedade de um `AWS::Lambda::NetworkConnector` recurso.

Name

O nome do conector de rede. O nome deve ser exclusivo em sua conta e região.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::Lambda::NetworkConnector` recurso.

OperatorRole

O ARN da função do IAM que o serviço de conector de rede assume para criar e gerenciar interfaces de rede elásticas (ENIs).

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [OperatorRole](#) propriedade de um `AWS::Lambda::NetworkConnector` recurso. Isso é necessário no CloudFormation , mas não no AWS SAM. Se você não especificar uma função, AWS SAM cria uma com uma ID lógica de `<resource-logical-id>OperatorRole`.

PropagateTags

Especifica se as tags da Tags propriedade devem ser passadas para os recursos gerados, como a função de operador gerada automaticamente.

Tipo: booleano

Obrigatório: não

Padrão: False

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Tags

Um mapa de pares de valores-chave que especifica as tags adicionadas a esse conector de rede.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Tags](#) propriedade de um `AWS::Lambda::NetworkConnector` recurso. A `Tags` propriedade in AWS SAM consiste em pares de valores-chave (enquanto CloudFormation nessa propriedade consiste em uma lista de Tag objetos). Além disso, adiciona AWS SAM automaticamente uma `lambda:createdBy: SAM` tag a esse conector de rede e às funções padrão que são geradas para ele.

VpcConfig

A configuração de saída da VPC para o conector de rede. Especifica as sub-redes, os grupos de segurança e o protocolo de rede para rotear o tráfego de saída por meio de sua VPC.

Digite: [VpcConfig](#)

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é transformada na [Configuration.VpcEgressConfiguration](#) propriedade de um `AWS::Lambda::NetworkConnector` recurso. AWS SAM conjuntos `AssociatedComputeResourceTypes` da [ComputeTypes](#) propriedade.

Valores de retorno

Ref.

Quando a ID lógica desse recurso é fornecida à função `Ref` intrínseca, ela retorna o ARN do conector de rede.

Para obter mais informações sobre como usar a função `Ref`, consulte [Ref](#) no Guia do usuário do AWS CloudFormation .

Fn:: GetAtt

`Fn::GetAtt` retorna um valor para um atributo especificado deste tipo. Estes são os atributos disponíveis e os valores de retorno de amostra.

Para obter mais informações sobre o uso do `Fn::GetAtt`, consulte [Fn::GetAtt](#) o AWS CloudFormation Guia do usuário.

Arn

O Amazon Resource Name (ARN) do conector de rede.

LatestVersionArn

O ARN qualificado para versão da versão mais recente do conector de rede, por exemplo.

`arn:aws:lambda:us-east-1:123456789012:network-connector:my-connector:1`

Use esse valor para a `EgressNetworkConnectorArn` propriedade de um [AWS::Serverless::WebFunction](#).

State

O estado atual do conector de rede. Os valores válidos são `PENDING`, `ACTIVE`, `INACTIVE`, `FAILED`, `DELETING` e `DELETE_FAILED`.

Exemplos

Conector de rede básico com função gerada automaticamente

O exemplo a seguir cria um conector de rede de saída VPC. AWS SAM gera automaticamente a função de operador.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31

Resources:
  MyVpcConnector:
    Type: AWS::Serverless::NetworkConnector
    Properties:
      Name: my-vpc-connector
      VpcConfig:
        SubnetIds:
          - subnet-0abc123def4567890
          - subnet-0def456abc7890123
        SecurityGroupIds:
          - sg-0abc1234abcde0001
        NetworkProtocol: IPv4
      Tags:
        Environment: Production
```

Conector de rede para uma função Web

O exemplo a seguir cria um conector de rede para Funções Web e o usa para o tráfego de saída de uma Função Web.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31

Resources:
  MyWebFunctionConnector:
    Type: AWS::Serverless::NetworkConnector
    Properties:
      ComputeTypes:
        - WebFunction
      VpcConfig:
        SubnetIds:
          - subnet-0abc123def4567890
        SecurityGroupIds:
          - sg-0abc1234abcde0001
        NetworkProtocol: IPv4

  MyWebFunction:
    Type: AWS::Serverless::WebFunction
    Properties:
      Runtime: nodejs24.x
      AuthType: IamAuth
      CodeUri: s3://amzn-s3-demo-bucket/webfn.zip
      EgressNetworkConnectorArn: !GetAtt MyWebFunctionConnector.LatestVersionArn
```

VpcConfig

Define as configurações de saída de VPC para um conector de rede, incluindo sub-redes, grupos de segurança e protocolo de rede para provisionamento de interface de rede elástica (ENI).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
NetworkProtocol: String
```


[SecurityGroupIds](#): *List*

[SubnetIds](#): *List*

Propriedades

NetworkProtocol

O protocolo de rede para a conexão de saída da VPC.

Valores válidos: IPv4, DualStack

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [NetworkProtocol](#) propriedade `Configuration.VpcEgressConfiguration` de um `AWS::Lambda::NetworkConnector` recurso.

SecurityGroupIds

Uma lista de IDs de grupos de segurança a serem associados às interfaces de rede elástica (ENIs).

Tipo: lista

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SecurityGroupIds](#) propriedade `Configuration.VpcEgressConfiguration` de um `AWS::Lambda::NetworkConnector` recurso.

SubnetIds

Uma lista de IDs de sub-rede em que o serviço provisiona ENIs para conectividade de saída de VPC.

Tipo: lista

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SubnetIds](#) propriedade `Configuration.VpcEgressConfiguration` de um `AWS::Lambda::NetworkConnector` recurso.

Exemplos

VpcConfig exemplo

```
VpcConfig:
  SubnetIds:
    - subnet-0abc123def4567890
    - subnet-0def456abc7890123
  SecurityGroupIds:
    - sg-0abc1234def567890
  NetworkProtocol: IPv4
```

AWS::Serverless::SimpleTable

Cria uma tabela do DynamoDB com uma chave primária de atributo único. É útil quando os dados só precisam ser acessados por meio de uma chave primária.

Para atributos mais avançados, use um recurso [AWS::DynamoDB::Table](#) no CloudFormation. Esses recursos podem ser usados no AWS SAM. Eles são abrangentes e fornecem mais personalização, incluindo [key schema](#) e personalização de [resource policy](#).

Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Type: AWS::Serverless::SimpleTable
Properties:
  PointInTimeRecoverySpecification: PointInTimeRecoverySpecification
  PrimaryKey: PrimaryKeyObject
  ProvisionedThroughput: ProvisionedThroughputObject
  SSESpecification: SSESpecification
```

`TableName`: *String*

`Tags`: *Map*

Propriedades

PointInTimeRecoverySpecification

As configurações usadas para habilitar a recuperação point-in-time.

Tipo: [PointInTimeRecoverySpecification](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [PointInTimeRecoverySpecification](#) propriedade de um `AWS::DynamoDB::Table` recurso.

PrimaryKey

Nome e tipo do atributo a ser usado como chave primária da tabela. Se não for fornecida, a chave primária será a `String` com um valor de `id`.

Note

O valor dessa propriedade não pode ser modificado após a criação desse recurso.

Digite: [PrimaryKeyObject](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ProvisionedThroughput

Leia e grave informações de provisionamento de Throughput.

Se não `ProvisionedThroughput` for especificado `BillingMode` será especificado como `PAY_PER_REQUEST`.

Digite: [ProvisionedThroughputObject](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ProvisionedThroughput](#) propriedade de um `AWS::DynamoDB::Table` recurso.

SSESpecification

Especifica as configurações para habilitar a criptografia no lado do servidor.

Digite: [SSESpecification](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [SSESpecification](#) propriedade de um `AWS::DynamoDB::Table` recurso.

TableName

Nome da tabela do DynamoDB.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TableName](#) propriedade de um `AWS::DynamoDB::Table` recurso.

Tags

Um mapa (string a string) que especifica as tags a serem adicionadas a ele. SimpleTable Para obter detalhes sobre chaves e valores válidos para tags, consulte [Etiqueta de recurso](#) no AWS CloudFormation Guia do usuário.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Tags](#) propriedade de um `AWS::DynamoDB::Table` recurso. A propriedade Tags no SAM consiste em pares de chave-valor; CloudFormation nela consiste em uma lista de objetos Tag.

Valores de retorno

Ref.

Quando o ID lógico desse recurso é fornecido para a função intrínseca Ref, retorna o nome do recurso da tabela subjacente do DynamoDB.

Para obter mais informações sobre como usar a função [Ref](#), consulte [Ref](#) no Guia do usuário do AWS CloudFormation .

Exemplos

SimpleTableExample

Exemplo de um SimpleTable

YAML

```
Properties:
  TableName: my-table
  PrimaryKey:
    Name: MyPrimaryKey
    Type: String
  ProvisionedThroughput:
    ReadCapacityUnits: 5
    WriteCapacityUnits: 5
  Tags:
    Department: Engineering
    AppType: Serverless
```

PrimaryKeyObject

O objeto que descreve as propriedades de uma chave primária.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Name: String
Type: String
```

Propriedades

Name

O nome do atributo da chave primária.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AttributeName](#) propriedade do tipo de AWS::DynamoDB::Table AttributeDefinition dados.

Notas adicionais: Essa propriedade também é passada para a [AttributeName](#) propriedade de um tipo de AWS::DynamoDB::Table KeySchema dados.

Type

O tipo de dados da chave primária.

Valores válidos: String, Number, Binary

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AttributeType](#) propriedade do tipo de AWS::DynamoDB::Table AttributeDefinition dados.

Exemplos

PrimaryKey

Exemplo de chave primária.

YAML

```
Properties:
  PrimaryKey:
    Name: MyPrimaryKey
    Type: String
```

ProvisionedThroughputObject

O objeto que descreve as propriedades de um throughput provisionado.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
ReadCapacityUnits: Integer  
WriteCapacityUnits: Integer
```

Propriedades

ReadCapacityUnits

O número máximo de leituras fortemente consistentes consumidas por segundo antes de o DynamoDB retornar um `ThrottlingException`.

Tipo: inteiro

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ReadCapacityUnits](#) propriedade do tipo de `AWS::DynamoDB::Table` ProvisionedThroughput dados.

WriteCapacityUnits

O número máximo de gravações consumidas por segundo antes de o DynamoDB retornar um `ThrottlingException`.

Tipo: inteiro

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [WriteCapacityUnits](#) propriedade do tipo de `AWS::DynamoDB::Table` ProvisionedThroughput dados.

Exemplos

ProvisionedThroughput

Exemplo de throughput provisionado.

YAML

```
Properties:
  ProvisionedThroughput:
    ReadCapacityUnits: 5
    WriteCapacityUnits: 5
```

AWS::Serverless::StateMachine

Cria uma máquina de AWS Step Functions estado, que você pode usar para orquestrar funções de AWS Lambda eventos e outros AWS recursos para formar fluxos de trabalho complexos e robustos.

Para obter mais informações sobre o Steps Functions, consulte o [Guia do desenvolvedor do AWS Step Functions](#).

Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

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
```

Propriedades

AutoPublishAlias

O nome do alias da máquina de estado. Para saber mais sobre o uso dos aliases de máquina de estado do Step Functions, consulte [Gerenciar implantações contínuas com versões e aliases](#) no AWS Step Functions Guia do desenvolvedor.

Use `DeploymentPreference` para configurar as preferências de implantação do seu alias. Se você não especificar `DeploymentPreference`, AWS SAM configurará o tráfego para mudar para a versão mais recente da máquina de estado de uma só vez.

AWS SAM define a versão `DeletionPolicy` e `UpdateReplacePolicy` como `Retain` por padrão. As versões anteriores não serão excluídas automaticamente.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::StepFunctions::StateMachineAlias` recurso.

UseAliasAsEventTarget

Indique se o alias, criado usando a propriedade `AutoPublishAlias`, deve ou não ser passado para o destino da origem de eventos definido com [Eventos](#).

Especifique `True` para usar o alias como destino do evento.

Tipo: booleano

Obrigatório: não

Padrão: `False`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Definition

A definição da máquina de estado é um objeto, em que o formato do objeto corresponde ao formato do seu arquivo de AWS SAM modelo, por exemplo, JSON ou YAML. As definições de máquina de estado aderem ao [Amazon States Language](#).

Para obter um exemplo de uma definição de máquina de estado embutida, consulte [Exemplos](#).

Você deve fornecer um Definition ou um DefinitionUri.

Tipo: mapa

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

DefinitionSubstitutions

Um mapa de string para string que especifica os mapeamentos para variáveis de espaço reservado na definição da máquina de estado. Isso permite injetar valores obtidos em runtime (por exemplo, de funções intrínsecas) na definição da máquina de estado.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [DefinitionSubstitutions](#) propriedade de um `AWS::StepFunctions::StateMachine` recurso. Se alguma função intrínseca for especificada em uma definição de máquina de estado embutida, AWS SAM adiciona entradas a essa propriedade para injetá-las na definição da máquina de estado.

DefinitionUri

[O URI do Amazon Simple Storage Service \(Amazon S3\) ou o caminho de arquivo local da definição de máquina de estado escrito na Amazon States Language.](#)

Se você fornecer um caminho de arquivo local, o modelo deverá passar pelo fluxo de trabalho que inclui o comando `sam deploy` ou `sam package` para transformar corretamente a definição. Para tanto, você deve usar a versão 0.52.0 ou posterior da CLI AWS SAM .

Você deve fornecer um `Definition` ou um `DefinitionUri`.

Tipo: String | [S3Location](#)

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [DefinitionS3Location](#) propriedade de um `AWS::StepFunctions::StateMachine` recurso.

DeploymentPreference

As configurações que habilitam e configuram implantações graduais de máquinas de estado. Para saber mais sobre as implantações graduais do Step Functions, consulte [Gerenciar implantações contínuas com versões e aliases](#) no AWS Step Functions Guia do desenvolvedor.

Especifique `AutoPublishAlias` antes de configurar essa propriedade. Suas configurações `DeploymentPreference` serão aplicadas ao alias especificado com `AutoPublishAlias`.

Quando você especifica `DeploymentPreference`, AWS SAM gera o valor da `StateMachineVersionArn` subpropriedade automaticamente.

Digite: [DeploymentPreference](#)

Obrigatório: não

CloudFormation compatibilidade: AWS SAM gera e anexa o valor da `StateMachineVersionArn` propriedade `DeploymentPreference` e passa `DeploymentPreference` para a [DeploymentPreference](#) propriedade de um `AWS::StepFunctions::StateMachineAlias` recurso.

Events

Especifica os eventos que acionam essa máquina de estado. Os eventos consistem em um tipo e um conjunto de propriedades que dependem do tipo.

Digite: [EventSource](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Logging

Define quais eventos de histórico de execução são registrados e onde eles são registrados.

Digite: [LoggingConfiguration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [LoggingConfiguration](#) propriedade de um `AWS::StepFunctions::StateMachine` recurso.

Name

O nome da máquina de estado.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StateMachineName](#) propriedade de um `AWS::StepFunctions::StateMachine` recurso.

PermissionsBoundary

O ARN de um limite de permissões a ser usado para a função de execução dessa máquina de estado. Essa propriedade só funciona se a função for gerada para você.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [PermissionsBoundary](#) propriedade de um `AWS::IAM::Role` recurso.

Policies

Políticas de permissão para essa máquina de estado. As políticas serão anexadas à função de execução padrão AWS Identity and Access Management (IAM) da máquina de estado.

Essa propriedade aceita um único valor ou uma lista de valores. Os valores permitidos incluem:

- [AWS SAM Modelos de políticas](#).

- O ARN de uma [política AWS gerenciada ou política gerenciada pelo cliente](#).
- O nome de uma política AWS gerenciada da [lista](#) a seguir.
- Uma [política do IAM em linha](#) formatada em YAML como um mapa.

 Note

Se você especificar a propriedade `Role`, essa propriedade será ignorada.

Tipo: String | List | Map

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

PropagateTags

Indique se deseja ou não passar as tags da propriedade `Tags` para os recursos [AWS::Serverless::StateMachine](#) gerados. Especifique `True` para propagar as tags nos recursos gerados.

Tipo: booleano

Obrigatório: não

Padrão: `False`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Role

O ARN de um perfil do IAM a ser usado como a função de execução dessa máquina de estado.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RoleArn](#) propriedade de um `AWS::StepFunctions::StateMachine` recurso.

RolePath

O caminho para a função de execução do IAM da máquina de estado.

Use essa propriedade quando a função for gerada para você. Não use quando a função for especificada com a propriedade Role.

Tipo: string

Obrigatório: Condicional

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Path](#) propriedade de um `AWS::IAM::Role` recurso.

Tags

Um mapa de string para string que especifica as tags adicionadas à máquina de estado e a função de execução correspondente. Para obter informações sobre chaves e valores válidos para tags, consulte a propriedade [Tags](#) de um recurso [AWS::StepFunctions::StateMachine](#).

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Tags](#) propriedade de um `AWS::StepFunctions::StateMachine` recurso. AWS SAM adiciona automaticamente uma `stateMachine:createdBy:SAM` tag a esse recurso e à função padrão gerada para ele.

Tracing

Seleciona se AWS X-Ray está ou não habilitado para a máquina de estado. Para obter mais informações sobre como usar X-Ray com Step Functions, consulte [AWS X-Ray Step Functions](#) no AWS Step Functions Developer Guide.

Digite: [TracingConfiguration](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [TracingConfiguration](#) propriedade de um `AWS::StepFunctions::StateMachine` recurso.

Type

O tipo da máquina de estado.

Valores válidos: STANDARD ou EXPRESS

Tipo: string

Obrigatório: não

Padrão: STANDARD

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StateMachineType](#) propriedade de um `AWS::StepFunctions::StateMachine` recurso.

Valores de retorno

Ref.

Quando você fornece o ID lógico desse recurso para a função intrínseca Ref, Ref retorna o nome de recurso da Amazon (ARN) do recurso `AWS::StepFunctions::StateMachine` subjacente.

Para obter mais informações sobre como usar a função Ref, consulte [Ref](#) no Guia do usuário do AWS CloudFormation .

Fn:: GetAtt

`Fn::GetAtt` retorna um valor para um atributo especificado deste tipo. Estes são os atributos disponíveis e os valores de retorno de amostra.

Para obter mais informações sobre o uso do `Fn::GetAtt`, consulte [Fn::GetAtt](#) no Guia do usuário do AWS CloudFormation .

Name

Retorna o nome da máquina de estado, como HelloWorld-StateMachine.

Exemplos

Arquivo de definição da máquina de estado

Veja a seguir um exemplo de uma definição de máquina de estado em linha que permite que uma função do lambda invoque a máquina de estado. Observe que esse exemplo espera

que a propriedade `Role` configure a política adequada para permitir a invocação. O arquivo `my_state_machine.asl.json` deve ser escrito no [Amazon States Language](#).

Neste exemplo, as `DefinitionSubstitution` entradas permitem que a máquina de estado inclua recursos declarados no arquivo AWS SAM de modelo.

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
```

Definição de máquina de estado em linha

Veja a seguir um exemplo de definição de máquina de estado em linha.

Neste exemplo, o arquivo de AWS SAM modelo é escrito em YAML, então a definição da máquina de estado também está em YAML. Para declarar uma definição de máquina de estado embutida em JSON, escreva seu arquivo de AWS SAM modelo em JSON.

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

O objeto que descreve a origem dos eventos que acionam a máquina de estado. Cada evento consiste em um tipo e um conjunto de propriedades que dependem desse tipo. Para obter informações sobre as propriedades da origem de cada evento, consulte o subtópico correspondente a esse tipo.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Properties: Schedule | ScheduleV2 | CloudWatchEvent | EventBridgeRule | Api  
Type: String
```

Propriedades

Properties

Um objeto que descreve as propriedades desse mapeamento de eventos. O conjunto de propriedades deve estar em conformidade com o Type definido.

Tipo: [Cronograma](#) | [ScheduleV2](#) | [CloudWatchEvent](#) | [Api](#) | [EventBridgeRule](#)

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Type

O tipo de evento.

Valores válidos: `Api`, `Schedule`, `ScheduleV2`, `CloudWatchEvent`, `EventBridgeRule`

Tipo: `string`

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

solicitações de

A seguir está um exemplo de um evento do tipo API.

YAML

```
ApiEvent:
  Type: Api
  Properties:
    Method: get
    Path: /group/{user}
    RestApiId:
      Ref: MyApi
```

Api

O objeto que descreve um tipo de origem do evento Api. Se um recurso [AWS::Serverless::API](#) for definido, os valores do caminho e do método devem corresponder a uma operação na definição de OpenAPI da API.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Auth: ApiStateMachineAuth
Method: String
Path: String
RestApiId: String
UnescapeMappingTemplate: Boolean
```

Propriedades

Auth

A configuração de autorização para essa API, caminho e método.

Use essa propriedade para substituir a configuração `DefaultAuthorizer` da API para um caminho individual, quando nenhum `DefaultAuthorizer` for especificado, ou para substituir a configuração padrão `ApiKeyRequired`.

Digite: [ApiStateMachineAuth](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Method

O método HTTP para o qual essa função é invocada.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Path

O caminho do URI para o qual esta função é invocada. O valor deve começar com `/`.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

RestApiId

O identificador de um recurso `RestApi`, que deve conter uma operação com o caminho e o método fornecidos. Normalmente, isso é definido para fazer referência a um recurso [AWS::Serverless::API](#) definido nesse modelo.

Se você não definir essa propriedade, AWS SAM cria um [AWS::Serverless::API](#) recurso padrão usando um OpenApi documento gerado. Esse recurso contém uma união de todos os caminhos e métodos definidos por eventos Api no mesmo modelo que não especificam um arquivo `RestApiId`.

Esta propriedade não pode fazer referência a um recurso [AWS::Serverless::API](#) definido em outro modelo.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

UnescapeMappingTemplate

Elimina aspas simples, substituindo \ ' por ' , na entrada que é passada para a máquina de estado. Use quando sua entrada contiver aspas simples.

Note

Se definido como `False` e sua entrada contiver aspas simples, ocorrerá um erro.

Tipo: booleano

Obrigatório: não

Padrão: `False`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

ApiEvent

A seguir está um exemplo de um evento do tipo `Api`.

YAML

```
Events:
  ApiEvent:
    Type: Api
    Properties:
      Path: /path
```

```
Method: get
```

ApiStateMachineAuth

Configura a autorização no nível do evento, para uma API, um caminho e um método específicos.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
ApiKeyRequired: Boolean  
AuthorizationScopes: List  
Authorizer: String  
ResourcePolicy: ResourcePolicyStatement
```

Propriedades

ApiKeyRequired

Requer uma chave de API para essa API, caminho e método.

Tipo: booliano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AuthorizationScopes

Os escopos de autorização a serem aplicados a essa API, caminho e método.

Os escopos que você especificar substituirão quaisquer escopos aplicados pela propriedade `DefaultAuthorizer`, caso você a tenha especificado.

Tipo: Lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Authorizer

O Authorizer para uma máquina de estado específica.

Se você especificou um autorizador global para a API e deseja tornar essa máquina de estado pública, substitua o autorizador global configurando Authorizer como NONE.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

ResourcePolicy

Configure a política de recursos para esse caminho em uma API.

Digite: [ResourcePolicyStatement](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

StateMachine-Autenticação

O exemplo a seguir especifica a autorização no nível da máquina de estado.

YAML

```
Auth:
  ApiKeyRequired: true
  Authorizer: NONE
```

ResourcePolicyStatement

Configura uma política de recursos para todos os métodos e caminhos de uma API. Para obter mais informações sobre políticas de recursos, consulte Como [controlar o acesso a uma API com as políticas de recursos do API Gateway](#) no Guia do desenvolvedor do API Gateway.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
AwsAccountBlacklist: List  
AwsAccountWhitelist: List  
CustomStatements: List  
IntrinsicVpcBlacklist: List  
IntrinsicVpcWhitelist: List  
IntrinsicVpceBlacklist: List  
IntrinsicVpceWhitelist: List  
IpRangeBlacklist: List  
IpRangeWhitelist: List  
SourceVpcBlacklist: List  
SourceVpcWhitelist: List
```

Propriedades

AwsAccountBlacklist

As AWS contas a serem bloqueadas.

Tipo: lista de strings

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

AwsAccountWhitelist

As AWS contas a serem permitidas. Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista de strings

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

CustomStatements

Uma lista de declarações de política de recursos personalizadas a serem aplicadas a essa API. Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntrinsicVpcBlacklist

A lista de nuvens privadas virtuais (VPCs) a serem bloqueadas, em que cada VPC é especificada como uma referência, como uma [referência dinâmica](#) ou a função Ref [intrínseca](#). Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntrinsicVpcWhitelist

A lista de VPCs permissões, em que cada VPC é especificada como uma referência, como uma [referência dinâmica](#) ou a função Ref [intrínseca](#).

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntrinsicVpceBlacklist

A lista de endpoints da VPC a serem bloqueados, em que cada endpoint da VPC é especificado como uma referência, como uma [referência dinâmica](#) ou a Ref [função intrínseca](#).

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IntrinsicVpceWhitelist

A lista de endpoints da VPC a serem permitidos, em que cada endpoint da VPC é especificado como uma referência, como uma [referência dinâmica](#) ou a Ref [função intrínseca](#). Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IpRangeBlacklist

Os endereços IP ou intervalos de endereços a serem bloqueados. Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IpRangeWhitelist

Os endereços IP ou intervalos de endereços a serem permitidos.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SourceVpcBlacklist

A VPC de origem ou os endpoints da VPC a serem bloqueados. Os nomes da VPC de origem devem começar com "vpc-" e os nomes dos endpoints da VPC de origem devem começar com "vpce-". Para obter um exemplo de uso desta propriedade, consulte a seção Exemplos na parte inferior desta página.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

SourceVpcWhitelist

A VPC de origem ou os endpoints da VPC a serem permitidos. Os nomes da VPC de origem devem começar com "vpc-" e os nomes dos endpoints da VPC de origem devem começar com "vpce-".

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

Exemplo de política de recursos

O exemplo a seguir bloqueia dois endereços IP e uma VPC de origem e permite uma AWS conta.

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

O objeto que descreve um tipo de fonte de evento CloudWatchEvent.

AWS Serverless Application Model (AWS SAM) gera um [AWS::Events::Rule](#) recurso quando esse tipo de evento é definido.

Nota importante: [EventBridgeRule](#) é o tipo de fonte de eventos preferido a ser usado, em vez de CloudWatchEvent. EventBridgeRule e CloudWatchEvent use o mesmo serviço, API e CloudFormation recursos subjacentes. No entanto, AWS SAM adicionará suporte para novos recursos somente para EventBridgeRule.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```

EventBusName: String
Input: String

```

InputPath: *String*
Pattern: EventPattern

Propriedades

EventBusName

O barramento de eventos que deve ser associado a essa regra. Se você omitir essa propriedade, AWS SAM usará o barramento de eventos padrão.

Tipo: string

Obrigatório: não

Padrão: barramento de eventos padrão

CloudFormation compatibilidade: essa propriedade é passada diretamente para a EventBusName propriedade de um `AWS::Events::Rule` recurso.

Input

Texto JSON válido passado para o destino. Se você usar essa propriedade, nada do próprio texto do evento é passado para o destino.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a Input propriedade de um `AWS::Events::Rule` `Target` recurso.

InputPath

Quando você não deseja passar todo o evento correspondente ao destino, a propriedade `InputPath` descreve qual parte do evento passar.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a InputPath propriedade de um `AWS::Events::Rule` `Target` recurso.

Pattern

Descreve quais eventos são roteados para o destino especificado. Para obter mais informações, consulte [Eventos e padrões de eventos EventBridge no Guia do EventBridge usuário da Amazon](#).

Digite: [EventPattern](#)

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventPattern](#) propriedade de um `AWS::Events::Rule` recurso.

Exemplos

CloudWatchEvent

O exemplo a seguir mostra o tipo de origem de um evento CloudWatchEvent.

YAML

```
CWEvent:
  Type: CloudWatchEvent
  Properties:
    Input: '{"Key": "Value"}'
    Pattern:
      detail:
        state:
          - running
```

EventBridgeRule

O objeto que descreve um tipo de fonte de EventBridgeRule evento, que define sua máquina de estado como destino para uma EventBridge regra da Amazon. Para obter mais informações, consulte [O que é a Amazon EventBridge?](#) no Guia do EventBridge usuário da Amazon.

AWS SAM gera um [AWS::Events::Rule](#) recurso quando esse tipo de evento é definido.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
DeadLetterConfig: DeadLetterConfig  
EventBusName: String  
Input: String  
InputPath: String  
InputTransformer: InputTransformer  
Pattern: EventPattern  
RetryPolicy: RetryPolicy  
RuleName: String  
State: String  
Target: Target
```

Propriedades

DeadLetterConfig

Configure a fila do Amazon Simple Queue Service (Amazon SQS) para a EventBridge que envia eventos após uma falha na invocação de destino. A invocação pode falhar, por exemplo, ao enviar um evento para uma função de evento do Lambda que não existe ou quando EventBridge não há permissões suficientes para invocar a função do Lambda. Para obter mais informações, consulte [Política de repetição de eventos e uso de filas de mensagens sem saída no Guia do usuário](#) da Amazon. EventBridge

Digite: [DeadLetterConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [DeadLetterConfig](#) propriedade do tipo de `AWS::Events::Rule` Target dados. A AWS SAM versão dessa propriedade inclui subpropriedades adicionais, caso você queira criar AWS SAM a fila de mensagens mortas para você.

EventBusName

O barramento de eventos que deve ser associado a essa regra. Se você omitir essa propriedade, AWS SAM usará o barramento de eventos padrão.

Tipo: string

Obrigatório: não

Padrão: barramento de eventos padrão

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventBusName](#) propriedade de um `AWS::Events::Rule` recurso.

Input

Texto JSON válido passado para o destino. Se você usar essa propriedade, nada do próprio texto do evento é passado para o destino.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Input](#) propriedade de um `AWS::Events::Rule Target` recurso.

InputPath

Quando você não deseja passar todo o evento correspondente ao destino, a propriedade `InputPath` descreve qual parte do evento passar.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [InputPath](#) propriedade de um `AWS::Events::Rule Target` recurso.

InputTransformer

Configurações para permitir que você forneça entrada personalizada para um destino com base em determinados dados de evento. Você pode extrair um ou mais pares de valor-chave do evento e usar esses dados para enviar a entrada personalizada para o destino. Para obter mais informações, consulte [Transformação EventBridge de entrada](#) da Amazon no Guia EventBridge do usuário da Amazon.

Digite: [InputTransformer](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [InputTransformer](#) propriedade de um tipo de `AWS::Events::Rule Target` dados.

Pattern

Descreve quais eventos são roteados para o destino especificado. Para obter mais informações, consulte [Eventos e padrões de eventos EventBridge no Guia do EventBridge usuário da Amazon](#).

Digite: [EventPattern](#)

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EventPattern](#) propriedade de um `AWS::Events::Rule` recurso.

RetryPolicy

Um objeto `RetryPolicy` que inclui informações sobre as configurações de política de repetição. Para obter mais informações, consulte [Política de repetição de eventos e uso de filas de mensagens sem saída no Guia do usuário da Amazon. EventBridge](#)

Digite: [RetryPolicy](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RetryPolicy](#) propriedade do tipo de `AWS::Events::Rule` Target dados.

RuleName

O nome da regra.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::Events::Rule` recurso.

State

O estado da regra.

Valores válidos: [DISABLED | ENABLED]

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [State](#) propriedade de um `AWS::Events::Rule` recurso.

Target

O AWS recurso que é EventBridge invocado quando uma regra é acionada. Você pode usar essa propriedade para especificar a ID lógica do destino. Se essa propriedade não for especificada, a ID lógica do destino será AWS SAM gerada.

Tipo: [Target](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Targets](#) propriedade de um `AWS::Events::Rule` recurso. A versão AWS SAM dessa propriedade só permite que você especifique a ID lógica de um único destino.

Exemplos

EventBridgeRule

O exemplo a seguir mostra o tipo de origem de um evento EventBridgeRule.

YAML

```
EBRule:
  Type: EventBridgeRule
  Properties:
    Input: '{"Key": "Value"}'
    Pattern:
      detail:
        state:
          - terminated
```

DeadLetterConfig

O objeto usado para especificar a fila do Amazon Simple Queue Service (Amazon SQS) para a EventBridge qual envia eventos após uma falha na invocação de destino. A invocação pode falhar, por exemplo, ao enviar um evento para uma máquina de estado que não existe ou quando há permissões insuficientes para invocar a máquina de estado. Para obter mais informações, consulte [Política de repetição de eventos e uso de filas de mensagens sem saída no Guia do usuário](#) da Amazon. EventBridge

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Arn: String  
QueueLogicalId: String  
Type: String
```

Propriedades

Arn

O nome de recurso da Amazon (ARN) da fila Amazon SQS especificado como o destino para a fila de mensagens não entregues.

Note

Especifique a propriedade Type ou a propriedade Arn, mas não ambas.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Arn](#) propriedade do tipo de `AWS::Events::Rule DeadLetterConfig` dados.

QueueLogicalId

O nome personalizado da fila de letras mortas que AWS SAM cria se Type for especificado.

Note

Se a propriedade Type não estiver definida, essa propriedade será ignorada.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Type

O tipo da fila. Quando essa propriedade é definida, cria AWS SAM automaticamente uma fila de mensagens mortas e anexa a [política baseada em recursos](#) necessária para conceder permissão ao recurso de regra para enviar eventos para a fila.

Note

Especifique a propriedade Type ou a propriedade Arn, mas não ambas.

Valores válidos: SQS

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

DeadLetterConfig

DeadLetterConfig

YAML

```
DeadLetterConfig:
  Type: SQS
  QueueLogicalId: MyDLQ
```

Target

Configura o AWS recurso que é EventBridge invocado quando uma regra é acionada.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

Id: *String*

Propriedades

Id

O ID lógico do destino.

O valor do Id pode incluir caracteres alfanuméricos, pontos (.), hifens (-) e sublinhados (_).

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Id](#) propriedade do tipo de `AWS::Events::Rule` Target dados.

Exemplos

Target

YAML

```
EBRule:
  Type: EventBridgeRule
  Properties:
    Target:
      Id: MyTarget
```

Agendamento

O objeto que descreve um tipo de fonte de `Schedule` evento, que define sua máquina de estado como o destino de uma `EventBridge` regra que é acionada em um cronograma. Para obter mais informações, consulte [O que é a Amazon EventBridge?](#) no Guia do EventBridge usuário da Amazon.

AWS Serverless Application Model (AWS SAM) gera um [AWS::Events::Rule](#) recurso quando esse tipo de evento é definido.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
DeadLetterConfig: DeadLetterConfig  
Description: String  
Enabled: Boolean  
Input: String  
Name: String  
RetryPolicy: RetryPolicy  
RoleArn: String  
Schedule: String  
State: String  
Target: Target
```

Propriedades

DeadLetterConfig

Configure a fila do Amazon Simple Queue Service (Amazon SQS) para a EventBridge qual envia eventos após uma falha na invocação de destino. A invocação pode falhar, por exemplo, ao enviar um evento para uma função de evento do Lambda que não existe ou quando EventBridge não há permissões suficientes para invocar a função do Lambda. Para obter mais informações, consulte [Política de repetição de eventos e uso de filas de mensagens sem saída no Guia do usuário](#) da Amazon. EventBridge

Digite: [DeadLetterConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [DeadLetterConfig](#) propriedade do tipo de `AWS::Events::Rule` Target dados. A AWS SAM versão dessa propriedade inclui subpropriedades adicionais, caso você queira criar AWS SAM a fila de mensagens mortas para você.

Description

Uma descrição da regra.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::Events::Rule` recurso.

Enabled

Indica se a regra está habilitada.

Para desativar a regra, defina essa propriedade como `false`.

 Note

Especifique a propriedade `Enabled` ou `State`, mas não ambas.

Tipo: booleano

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [State](#) propriedade de um `AWS::Events::Rule` recurso. Se essa propriedade for definida como `true` então AWS SAM passa `ENABLED`, caso contrário, ela passa `DISABLED`.

Input

Texto JSON válido passado para o destino. Se você usar essa propriedade, nada do próprio texto do evento é passado para o destino.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Input](#) propriedade de um `AWS::Events::Rule Target` recurso.

Name

O nome da regra. Se você não especificar um nome, CloudFormation gera uma ID física exclusiva e usa essa ID para o nome da regra.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::Events::Rule` recurso.

RetryPolicy

Um objeto `RetryPolicy` que inclui informações sobre as configurações de política de repetição. Para obter mais informações, consulte [Política de repetição de eventos e uso de filas de mensagens sem saída no Guia do usuário](#) da Amazon. EventBridge

Digite: [RetryPolicy](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RetryPolicy](#) propriedade do tipo de `AWS::Events::Rule` Target dados.

RoleArn

O ARN da função do IAM que o EventBridge Scheduler usará para o destino quando o agendamento for invocado.

Digite: [RoleArn](#)

Obrigatório: Não. Se não for fornecido, uma nova função será criada e usada.

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RoleArn](#) propriedade do tipo de `AWS::Scheduler::Schedule` Target dados.

Schedule

A expressão de programação que determina quando e com que frequência a regra é executada. Para obter mais informações, consulte [Programar expressões para regras](#).

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ScheduleExpression](#) propriedade de um `AWS::Events::Rule` recurso.

State

O estado da regra.

Valores aceitos: DISABLED | ENABLED

 Note

Especifique a propriedade `Enabled` ou `State`, mas não ambas.

Tipo: `string`

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [State](#) propriedade de um `AWS::Events::Rule` recurso.

Target

O AWS recurso que é EventBridge invocado quando uma regra é acionada. Você pode usar essa propriedade para especificar a ID lógica do destino. Se essa propriedade não for especificada, a ID lógica do destino será AWS SAM gerada.

Tipo: [Target](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Targets](#) propriedade de um `AWS::Events::Rule` recurso. A AWS SAM versão dessa propriedade só permite que você especifique a ID lógica de um único destino.

Exemplos

CloudWatch Agende um evento

CloudWatch Exemplo de agendamento de evento

YAML

```
CWSchedule:
  Type: Schedule
  Properties:
    Schedule: 'rate(1 minute)'
    Name: TestSchedule
    Description: test schedule
    Enabled: false
```


DeadLetterConfig

O objeto usado para especificar a fila do Amazon Simple Queue Service (Amazon SQS) para a EventBridge qual envia eventos após uma falha na invocação de destino. A invocação pode falhar, por exemplo, ao enviar um evento para uma máquina de estado que não existe ou quando há permissões insuficientes para invocar a máquina de estado. Para obter mais informações, consulte [Política de repetição de eventos e uso de filas de mensagens sem saída no Guia do usuário](#) da Amazon. EventBridge

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Arn: String  
QueueLogicalId: String  
Type: String
```

Propriedades

Arn

O nome de recurso da Amazon (ARN) da fila Amazon SQS especificado como o destino para a fila de mensagens não entregues.

Note

Especifique a propriedade Type ou a propriedade Arn, mas não ambas.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Arn](#) propriedade do tipo de `AWS::Events::Rule` DeadLetterConfig dados.

QueueLogicalId

O nome personalizado da fila de letras mortas que AWS SAM cria se Type for especificado.

 Note

Se a propriedade `Type` não estiver definida, essa propriedade será ignorada.

Tipo: `string`

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Type

O tipo da fila. Quando essa propriedade é definida, cria AWS SAM automaticamente uma fila de mensagens mortas e anexa a [política baseada em recursos](#) necessária para conceder permissão ao recurso de regra para enviar eventos para a fila.

 Note

Especifique a propriedade `Type` ou a propriedade `Arn`, mas não ambas.

Valores válidos: `SQS`

Tipo: `string`

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

Exemplos

`DeadLetterConfig`

`DeadLetterConfig`

YAML

```
DeadLetterConfig:
```

```
Type: SQS
QueueLogicalId: MyDLQ
```

Target

Configura o AWS recurso que é EventBridge invocado quando uma regra é acionada.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Id: String
```

Propriedades

Id

O ID lógico do destino.

O valor do Id pode incluir caracteres alfanuméricos, pontos (.), hifens (-) e sublinhados (_).

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Id](#) propriedade do tipo de `AWS::Events::Rule` Target dados.

Exemplos

Target

YAML

```
EBRule:
  Type: Schedule
  Properties:
    Target:
      Id: MyTarget
```

ScheduleV2

O objeto que descreve um tipo de fonte de `ScheduleV2` evento, que define sua máquina de estado como o destino de um evento do Amazon EventBridge Scheduler que é acionado em um agendamento. Para obter mais informações, consulte [O que é o Amazon EventBridge Scheduler?](#) no Guia do usuário do EventBridge Scheduler.

AWS Serverless Application Model (AWS SAM) gera um [AWS::Scheduler::Scheduler](#) recurso quando esse tipo de evento é definido.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

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
```

Propriedades

DeadLetterConfig

Configure a fila do Amazon Simple Queue Service (Amazon SQS) para a EventBridge qual envia eventos após uma falha na invocação de destino. A invocação pode falhar, por exemplo, ao enviar um evento para uma função de evento do Lambda que não existe ou quando EventBridge não há permissões suficientes para invocar a função do Lambda. Para obter mais informações,

consulte [Configurando uma fila de mensagens mortas para o EventBridge Scheduler no Guia do usuário do Scheduler](#). EventBridge

Digite: [DeadLetterConfig](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [DeadLetterConfig](#) propriedade do tipo de AWS::Scheduler::Schedule Target dados. A AWS SAM versão dessa propriedade inclui subpropriedades adicionais, caso você queira criar AWS SAM a fila de mensagens mortas para você.

Description

Uma descrição da agenda.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um AWS::Scheduler::Schedule recurso.

EndDate

A data, em UTC, até a qual a agenda pode invocar seu destino. Dependendo da expressão de recorrência da agenda, as invocações podem ser interrompidas até a EndDate que você especifica.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EndDate](#) propriedade de um AWS::Scheduler::Schedule recurso.

FlexibleTimeWindow

Permite a configuração de uma janela na qual uma agenda pode ser invocada.

Digite: [FlexibleTimeWindow](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FlexibleTimeWindow](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

GroupName

O nome do grupo de agendas para associar a essa agenda. Se não for definido, o grupo padrão será usado.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [GroupName](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

Input

Texto JSON válido passado para o destino. Se você usar essa propriedade, nada do próprio texto do evento é passado para o destino.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Input](#) propriedade de um `AWS::Scheduler::Schedule Target` recurso.

KmsKeyArn

O ARN de uma chave KMS será usada para criptografar dados do cliente.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [KmsKeyArn](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

Name

O nome da programação. Se você não especificar um nome, AWS SAM gera um nome no formato *StateMachine-Logical-IDEvent-Source-Name* e usa essa ID para o nome da agenda.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Name](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

OmitName

Por padrão, AWS SAM gera e usa um nome de agendamento no formato de `<State-machine-logical-ID><event-source-name>`. Defina essa propriedade `true` para CloudFormation gerar uma ID física exclusiva e, em vez disso, use-a como nome da programação.

Tipo: booleano

Obrigatório: não

Padrão: `false`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

PermissionsBoundary

O ARN da política usada para definir o limite de permissões para a função.

Note

Se `PermissionsBoundary` estiver definido, AWS SAM aplicará os mesmos limites à função IAM de destino da agenda do agendador.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [PermissionsBoundary](#) propriedade de um `AWS::IAM::Role` recurso.

RetryPolicy

Um objeto `RetryPolicy` que inclui informações sobre as configurações de política de repetição.

Digite: [RetryPolicy](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RetryPolicy](#) propriedade do tipo de `AWS::Scheduler::Schedule Target` dados.

RoleArn

O ARN da função do IAM que o EventBridge Scheduler usará para o destino quando o agendamento for invocado.

Digite: [RoleArn](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [RoleArn](#) propriedade do tipo de `AWS::Scheduler::Schedule Target` dados.

ScheduleExpression

A expressão de agendamento que determina quando e com que frequência o agendamento é executado.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ScheduleExpression](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

ScheduleExpressionTimezone

O fuso horário no qual a expressão de agendamento é avaliada.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ScheduleExpressionTimezone](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

StartDate

A data, em UTC, a partir da qual a agenda pode começar a invocar um destino. Dependendo da expressão de recorrência da agenda, as invocações podem ocorrer a partir da `StartDate` que você especifica.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [StartDate](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

State

O estado da programação.

Valores aceitos: DISABLED | ENABLED

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [State](#) propriedade de um `AWS::Scheduler::Schedule` recurso.

Exemplos

Exemplo básico de definição de um recurso 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
```

AWS::Serverless::WebFunction

Cria uma função AWS Lambda Web, que atende solicitações HTTP por meio de um endpoint gerenciado. Ao contrário de uma função de evento Lambda, que é executada em resposta a invocações e eventos, uma função Web executa seu servidor web e recebe solicitações HTTP diretamente. A partir de uma única `AWS::Serverless::WebFunction`, AWS SAM gera a Função Web, uma revisão imutável que contém seu código e configuração, e um endpoint que direciona o tráfego para uma ou mais revisões.

Note

Quando você implanta AWS CloudFormation, AWS SAM transforma seus AWS SAM recursos em CloudFormation recursos. Para obter mais informações, consulte [Gerado CloudFormation recursos para AWS SAM](#).

Important

Seu código de função deve executar um servidor HTTP que escuta o host e a porta na variável de `AWS_LAMBDA_HTTP_ENDPOINT` ambiente. O tempo de execução não está definido `PORT`. Por padrão, o tempo de execução é executado `index.js` na raiz do pacote de implantação. Use [EntryPoint](#) para executar um arquivo diferente.

Important

O bucket do Amazon S3 que contém seu pacote de implantação deve atender aos seguintes requisitos:

- O controle de versão está ativado.
- Uma política de bucket concede `lambda.amazonaws.com:s3:GetObject` as `s3:GetObjectVersion` permissões e. Recomendamos que você defina o escopo da política com uma `ArnLike` condição `aws:SourceArn` que corresponda às revisões da função, por exemplo `arn:aws:lambda:us-east-1:111122223333:web-function/my-function/revision/*`.
- Se o objeto for criptografado com SSE-KMS, ele usará uma chave gerenciada pelo cliente cuja política de chaves permite `lambda.amazonaws.com` a chamada `kms:Decrypt`. A chave AWS gerenciada do Amazon S3 (`aws/s3`) não é suportada, porque você não pode alterar sua política de chaves.

Para ver um exemplo de política de bucket, consulte [Política de bucket para Lambda](#) no Guia do AWS Lambda desenvolvedor.

`CodeUri` deve referenciar um objeto no Amazon S3. Os `sam` package comandos `sam build` and não oferecem suporte a caminhos locais para esse recurso.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Type: AWS::Serverless::WebFunction
Properties:
  AuthType: String
  AutoDeploymentMode: String
  CodeUri: String | CodeUri
  EgressNetworkConnectorArn: String
  EndpointDescription: String
  EndpointName: String
  EndpointType: String
  EntryPoint: String
  EnvironmentVariables: Map
  ExecutionRoleArn: String
  FunctionName: String
  InlineCode: String
```

```
KmsKeyArn: String  
LoggingConfig: Map  
MaxConcurrencyPerEnvironment: Integer  
ReplicaRegions: List | ReplicaRegions  
RevisionDescription: String  
RevisionWeights: RevisionWeights  
Runtime: String  
ScalingConfig: Map  
Tags: Map  
ThrottleConfig: Map  
Timeout: Integer
```

Propriedades

AuthType

Como o endpoint autoriza solicitações.

- **IamAuth**: O Lambda autoriza solicitações com Signature versão 4 e IAM. Os chamadores precisam da `lambda:InvokeWebFunctionEndpoint` permissão, concedida por meio de uma política baseada em recursos sobre a função.
- **ApplicationManaged**: o endpoint é público. Sua função deve autorizar cada solicitação sozinha.

Valores válidos: `IamAuth`, `ApplicationManaged`

Tipo: `string`

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [AuthType](#) propriedade de um `AWS::Lambda::WebFunctionEndpoint` recurso.

AutoDeploymentMode

Como o endpoint direciona o tráfego entre as revisões.

- **LatestRevision**: envia 100% do tráfego para a revisão que essa implantação cria.
- **Disabled**: roteia o tráfego de acordo com [RevisionWeights](#).

Os valores que você pode usar e se `RevisionWeights` são obrigatórios dependem de [EndpointType](#):

- `HomeRegion`: use qualquer um dos valores. `ComDisabled`, `RevisionWeights` é obrigatório.
- `MultiRegionouPerRegion`: `usarDisabled`. Se você omitir `RevisionWeights`, o endpoint roteará 100% do tráfego para a revisão criada na mesma pilha, para que você possa criar o endpoint em uma única implantação. O tráfego não passa para as revisões criadas por implantações posteriores.

O valor deve ser resolvido ao AWS SAM transformar o modelo. Há suporte `Ref` para um parâmetro de `A a`. Outras funções intrínsecas não são suportadas.

Valores válidos: `LatestRevision`, `Disabled`

Tipo: `string`

Obrigatório: não

Padrão: `LatestRevision`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

`CodeUri`

A localização do pacote de implantação no Amazon S3. Especifique um `s3://bucket/key` URI ou um objeto com `BucketKey`, e `Version` chaves opcionais. Para saber os requisitos do bucket, consulte a nota na parte superior desta página.

Tipo: `String` | [CodeUri](#)

Obrigatório: sim

CloudFormation compatibilidade: essa propriedade é semelhante à [BuildConfig.CodeConfig.S3Object](#) propriedade de um

`AWS::Lambda::WebFunctionRevision` recurso. AWS SAM converte um `s3://` URI em `BucketKey`, e `VersionId`

`EgressNetworkConnectorArn`

O ARN qualificado para versão de um conector de rede que roteia o tráfego de saída da função por meio de sua VPC. O ARN deve terminar em um número de versão, por exemplo. `arn:aws:lambda:us-east-1:123456789012:network-connector:my-connector:1`

Para usar a versão mais recente de um conector no mesmo modelo, especifique `!GetAtt MyConnector.LatestVersionArn`. O conector [ComputeTypes](#) deve ser `WebFunction`.

Se você especificar uma string literal, AWS SAM verifique se é um ARN do conector de rede qualificado para a versão. As funções intrínsecas são transmitidas e validadas pelo serviço. A alteração desse valor cria uma nova revisão.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ServiceConfig.EgressNetworkConnectorArn](#) propriedade de um `AWS::Lambda::WebFunctionRevision` recurso.

EndpointDescription

Uma descrição do endpoint.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::Lambda::WebFunctionEndpoint` recurso.

EndpointName

O nome do endpoint. A alteração do nome substitui o endpoint. A alteração `EndpointType` também substitui o endpoint, portanto, mude `EndpointName` ao mesmo tempo. CloudFormation não é possível substituir um recurso com nome personalizado sem um novo nome.

Tipo: string

Obrigatório: não

Padrão: prod

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EndpointName](#) propriedade de um `AWS::Lambda::WebFunctionEndpoint` recurso.

EndpointType

Onde o endpoint atende ao tráfego.

- `HomeRegion`: serve da região da pilha.
- `MultiRegion`: replica a função para as regiões [ReplicaRegion](#)se encaminha cada solicitação para a região mais próxima.
- `PerRegion`: expõe um endpoint separado em cada região de réplica.

`MultiRegion` `PerRegion` `exigirAutoDeploymentMode`: `Disabled`.

Valores válidos: `HomeRegion`, `MultiRegion`, `PerRegion`

Tipo: `string`

Obrigatório: não

Padrão: `HomeRegion`

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [EndpointType](#) propriedade de um `AWS::Lambda::WebFunctionEndpoint` recurso.

EntryPoint

O arquivo que o tempo de execução executa, em relação à raiz do pacote de implantação. AWS SAM passa para o tempo de execução na variável de `AWS_LAMBDA_NODEJS_ENTRYPOINT` ambiente reservada.

Você não pode usar essa propriedade quando `EnvironmentVariables` é uma função intrínseca.

Tipo: `string`

Obrigatório: não

Padrão: `index.js`

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

EnvironmentVariables

Key-value pares a serem definidos como variáveis de ambiente para a revisão.

- As teclas devem começar com uma letra e conter somente letras, números e sublinhados.
- Os valores não devem estar vazios e ter no máximo 4.096 caracteres.
- AWS SAM converte valores numéricos e booleanos em cadeias de caracteres.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ServiceConfig.EnvironmentVariables](#) propriedade de um `AWS::Lambda::WebFunctionRevision` recurso.

ExecutionRoleArn

O ARN da função do IAM que a função assume quando é executada. Se você não especificar uma função, AWS SAM crie uma com a política `AWSLambdaBasicExecutionRole` gerenciada e uma ID lógica de `<resource-logical-id>Role`.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ServiceConfig.ExecutionRoleArn](#) propriedade de um `AWS::Lambda::WebFunctionRevision` recurso.

FunctionName

Nome da função. Os nomes das funções devem ser exclusivos em uma conta em todas as regiões. Se você não especificar um nome, AWS SAM gera um a partir do ID lógico e do ID da pilha, com no máximo 64 caracteres. A alteração do nome substitui a função, sua revisão e seu endpoint.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [FunctionName](#) propriedade dos `AWS::Lambda::WebFunctionEndpoint` recursos `AWS::Lambda::WebFunction` e `AWS::Lambda::WebFunctionRevision`, e.

InlineCode

Não suportado para `AWS::Serverless::WebFunction`. Use [CodeUri](#).

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

KmsKeyArn

O ARN da AWS KMS chave usada para criptografar as variáveis de ambiente da revisão.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [KmsKeyArn](#) propriedade de um `AWS::Lambda::WebFunctionRevision` recurso.

LoggingConfig

A configuração de registro para a revisão, como o grupo de registros e os níveis de registro.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ServiceConfig.TelemetryConfig.LoggingConfig](#) propriedade de um `AWS::Lambda::WebFunctionRevision` recurso.

MaxConcurrencyPerEnvironment

O número máximo de solicitações simultâneas que cada ambiente de execução processa.

Tipo: inteiro

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ServiceConfig.MaxConcurrencyPerEnvironment](#) propriedade de um `AWS::Lambda::WebFunctionRevision` recurso.

ReplicaRegions

As regiões para as quais replicar a função. Use essa propriedade com `MultiRegionPerRegion` endpoints. Especifique uma lista de códigos de região e grupos de regiões, ou um objeto com `IncludeRegions` um opcional `ExcludeRegions`.

Tipo: Lista | [ReplicaRegions](#)

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Regions](#) propriedade de um `AWS::Lambda::WebFunctionEndpoint` recurso. AWS SAM expande grupos de regiões e se aplica `ExcludeRegions` para produzir a lista de regiões.

RevisionDescription

Uma descrição da revisão que essa implantação cria.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [Description](#) propriedade de um `AWS::Lambda::WebFunctionRevision` recurso.

RevisionWeights

Um mapa de IDs de revisão para pesos de tráfego. Use essa propriedade quando `AutoDeploymentMode` estiver `Disabled`. Especifique até duas revisões. Cada peso é um número inteiro de 1 a 100, e os pesos devem somar 100.

Digite: [RevisionWeights](#)

Obrigatório: condicional. Exigido quando `AutoDeploymentMode` é `Disabled` e `EndpointType` é `HomeRegion`.

CloudFormation compatibilidade: essa propriedade é semelhante à [RevisionWeights](#) propriedade de um `AWS::Lambda::WebFunctionEndpoint` recurso. AWS SAM converte o mapa em uma lista `RevisionId` de `Weight` objetos.

Runtime

O tempo de execução da função, por exemplo `nodejs24.x`.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [BuildConfig.RuntimeConfig.Runtime](#) propriedade de um `AWS::Lambda::WebFunctionRevision` recurso.

ScalingConfig

A configuração de escalabilidade para o endpoint, como o número máximo de ambientes de execução.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ScalingConfig](#) propriedade de um `AWS::Lambda::WebFunctionEndpoint` recurso.

Tags

Um mapa de pares de valores-chave que especifica as tags adicionadas à função e à função de execução gerada. AWS SAM

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Tags](#) propriedade de um `AWS::Lambda::WebFunction` recurso. A `Tags` propriedade in AWS SAM consiste em pares de valores-chave (enquanto CloudFormation nessa propriedade consiste em uma lista de Tag objetos). Além disso, adiciona AWS SAM automaticamente uma `lambda:createdBy: SAM` tag à função e à função padrão que é gerada para ela.

ThrottleConfig

A configuração de limitação do endpoint, como a taxa máxima de solicitação.

Tipo: mapa

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ThrottleConfig](#) propriedade de um `AWS::Lambda::WebFunctionEndpoint` recurso.

Timeout

O tempo máximo, em segundos, que a função pode ser executada por solicitação. Os valores válidos são de 3 a 900.

Tipo: inteiro

Obrigatório: não

Padrão: 30

CloudFormation compatibilidade: essa propriedade é passada diretamente para a [ServiceConfig.TimeoutSeconds](#) propriedade de um `AWS::Lambda::WebFunctionRevision` recurso.

Valores de retorno

Ref.

Quando a ID lógica desse recurso é fornecida à função Ref intrínseca, ela retorna o ARN da função Web. Onde você precisar do nome da função, use o valor de `FunctionName` em vez disso.

Para obter mais informações sobre como usar a função Ref, consulte [Ref](#) no Guia do usuário do AWS CloudFormation .

Fn:: GetAtt

`Fn::GetAtt` retorna um valor para um atributo especificado deste tipo. Estes são os atributos disponíveis e os valores de retorno de amostra.

Para obter mais informações sobre o uso do `Fn::GetAtt`, consulte [Fn::GetAtt](#) o AWS CloudFormation Guia do usuário.

`FunctionArn`

O nome de recurso da Amazon (ARN) da função Web.

`State`

O estado atual da função Web.

`StateReason`

O motivo do estado atual da função Web.

`CreatedAt`

A hora em que a função Web foi criada.

UpdatedAt

A hora em que a função Web foi atualizada pela última vez.

Para referenciar a revisão ou o endpoint que AWS SAM gera, use suas IDs lógicas *<resource-logical-id>*Revision e *<resource-logical-id>*Endpoint. Por exemplo, `!GetAtt MyWebFunctionEndpoint.DomainName` retorna o nome de domínio do endpoint para uma função Web com o ID MyWebFunction lógico. Para obter mais informações, consulte [CloudFormation recursos gerados quando AWS::Serverless::WebFunction é especificado](#).

Exemplos

Função básica da Web

O exemplo a seguir cria uma função Web com um IAM-authorized endpoint na região da pilha. AWS SAM gera a função de execução e o endpoint encaminha todo o tráfego para a revisão que essa implantação cria.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31

Resources:
  MyWebFunction:
    Type: AWS::Serverless::WebFunction
    Properties:
      Runtime: nodejs24.x
      AuthType: IamAuth
      CodeUri:
        Bucket: amzn-s3-demo-bucket
        Key: webfn.zip
        Version: object-version-id
      Timeout: 30

Outputs:
  EndpointUrl:
    Value: !Sub https://${MyWebFunctionEndpoint.DomainName}
```

Divida o tráfego entre duas revisões

O exemplo a seguir desativa a implantação automática e envia 90% do tráfego para uma revisão existente e 10% para outra. Ambas as revisões já devem existir.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31

Resources:
  MyWebFunction:
    Type: AWS::Serverless::WebFunction
    Properties:
      Runtime: nodejs24.x
      AuthType: IamAuth
      CodeUri: s3://amzn-s3-demo-bucket/webfn-v2.zip
      AutoDeploymentMode: Disabled
      RevisionWeights:
        rev-abc123: 90
        rev-def456: 10
```

Multi-Region Função Web

O exemplo a seguir replica a função para as regiões dos EUA, exceto us-west-1 e encaminha cada solicitação para a região mais próxima. Como `RevisionWeights` é omitido, o endpoint roteia todo o tráfego para a revisão criada nessa pilha.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31

Resources:
  MyWebFunction:
    Type: AWS::Serverless::WebFunction
    Properties:
      Runtime: nodejs24.x
      AuthType: IamAuth
      CodeUri:
        Bucket: amzn-s3-demo-bucket
        Key: webfn.zip
        Version: object-version-id
      EndpointName: prodmr
      EndpointType: MultiRegion
      AutoDeploymentMode: Disabled
      ReplicaRegions:
        IncludeRegions:
          - US
        ExcludeRegions:
          - us-west-1
```

Função Web com saída de VPC

O exemplo a seguir cria um conector de rede para Web Functions e roteia o tráfego de saída da função pelas sub-redes e grupos de segurança do conector.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31

Resources:
  MyConnector:
    Type: AWS::Serverless::NetworkConnector
    Properties:
      ComputeTypes:
        - WebFunction
      VpcConfig:
        SubnetIds:
          - subnet-0abc123def4567890
          - subnet-0def456abc7890123
        SecurityGroupIds:
          - sg-0abc1234abcde0001
        NetworkProtocol: IPv4

  MyWebFunction:
    Type: AWS::Serverless::WebFunction
    Properties:
      Runtime: nodejs24.x
      AuthType: IamAuth
      CodeUri:
        Bucket: amzn-s3-demo-bucket
        Key: webfn.zip
        Version: object-version-id
      EgressNetworkConnectorArn: !GetAtt MyConnector.LatestVersionArn
```

CodeUri

Especifica a localização do Amazon S3 do pacote de implantação de uma função Web. Você pode especificar o local como um objeto ou como uma sequência de caracteres do formulário `s3://bucket/key` com um parâmetro de `?versionId=version` consulta opcional.

O bucket deve ter o versionamento ativado e uma política de bucket que `lambda.amazonaws.com` conceda as permissões `s3:GetObject` e `s3:GetObjectVersion`. Recomendamos que você defina o escopo da política com uma `aws:SourceArn` condição que corresponda às revisões da

função. Se o objeto estiver criptografado com SSE-KMS, use uma chave gerenciada pelo cliente cuja política de chaves permita `lambda.amazonaws.com` a chamada `kms:Decrypt`. A chave AWS gerenciada do Amazon S3 (`aws/s3`) não é suportada. Para obter detalhes, consulte [Implantação do código da função Web do Amazon S3](#) no Guia AWS Lambda do desenvolvedor.

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
Bucket: String
Key: String
Version: String
```

Propriedades

Bucket

O nome do bucket do Amazon S3 que contém o pacote de implantação.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a Bucket propriedade [BuildConfig.CodeConfig.S3Object](#) de um `AWS::Lambda::WebFunctionRevision` recurso.

Key

A chave do Amazon S3 do pacote de implantação.

Tipo: string

Obrigatório: Sim

CloudFormation compatibilidade: essa propriedade é passada diretamente para a Key propriedade [BuildConfig.CodeConfig.S3Object](#) de um `AWS::Lambda::WebFunctionRevision` recurso.

Version

O ID da versão do objeto do pacote de implantação. Especifique uma versão para fixar a revisão em um upload específico.

Tipo: string

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à `VersionId` propriedade [BuildConfig.CodeConfig.S3Object](#) de um `AWS::Lambda::WebFunctionRevision` recurso. AWS SAM passa o valor como `VersionId`.

Exemplos

Formulário de objeto

```
CodeUri:
  Bucket: amzn-s3-demo-bucket
  Key: webfn.zip
  Version: object-version-id
```

Formulário URI

```
CodeUri: s3://amzn-s3-demo-bucket/webfn.zip?versionId=object-version-id
```

ReplicaRegions

Especifica as regiões para as quais um endpoint `MultiRegion` ou função `PerRegion` Web replica a função. Você pode especificar uma lista de códigos de região e grupos de regiões ou um objeto que inclua regiões e, opcionalmente, exclua algumas delas.

Você pode usar os seguintes grupos de regiões em qualquer lugar em que especificar uma região:

- US: `us-east-1`, `us-east-2`, `us-west-1`, `us-west-2`
- EU: `eu-west-1`, `eu-west-2`, `eu-west-3`, `eu-central-1`, `eu-north-1`
- AP: `ap-south-1`, `ap-northeast-1`, `ap-northeast-2`, `ap-northeast-3`, `ap-southeast-1`, `ap-southeast-2`

- CA: ca-central-1
- SA: sa-east-1
- SPARSE_GLOBAL: uma ou duas regiões por continente para cobertura global com menos regiões: us-east-1, eu-west-1, eu-central-1, ap-northeast-1, ap-southeast-1, ap-southeast-2, sa-east-1
- ALL_STANDARD: todas as regiões nos SA grupos US, EU, AP, CA, e (17 regiões)

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
ExcludeRegions: List  
IncludeRegions: List
```

Propriedades

ExcludeRegions

Uma lista de códigos de região e grupos de regiões dos quais remover `IncludeRegions`. Você também deve especificar `IncludeRegions`. As entradas não podem ser funções intrínsecas e você não pode usar `ExcludeRegions` quando `IncludeRegions` contém funções intrínsecas.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é exclusiva AWS SAM e não tem CloudFormation equivalente.

IncludeRegions

Uma lista de códigos de região e grupos de regiões para os quais replicar a função. A lista em si não pode ser uma função intrínseca, mas entradas individuais podem.

Tipo: lista

Obrigatório: não

CloudFormation compatibilidade: essa propriedade é semelhante à [Regions](#) propriedade de um `AWS::Lambda::WebFunctionEndpoint` recurso. AWS SAM expande grupos de regiões e remove todas `ExcludeRegions` as entradas para produzir a lista de regiões.

Exemplos

Lista de regiões

O exemplo a seguir replica a função em duas regiões.

```
ReplicaRegions:
  - us-east-1
  - eu-west-1
```

Grupo de regiões com exclusões

O exemplo a seguir replica a função para as regiões dos EUA, excetous-west-1.

```
ReplicaRegions:
  IncludeRegions:
    - US
  ExcludeRegions:
    - us-west-1
```

RevisionWeights

Especifica como um endpoint do Web Function divide o tráfego entre as revisões. Use essa propriedade quando `AutoDeploymentMode` estiver `Disabled`, por exemplo, para executar um canário ou uma blue/green implantação.

`RevisionWeights` é um mapa. Cada chave é uma ID de revisão e cada valor é a porcentagem do tráfego a ser roteado para essa revisão. As seguintes regras se aplicam:

- Especifique no máximo duas revisões.
- Cada peso é um número inteiro de 1 a 100, e os pesos devem somar 100.
- Pesos individuais podem ser funções intrínsecas, mas o mapa em si não pode ser uma função intrínseca.

Tipo: mapa

CloudFormation compatibilidade: essa propriedade é semelhante à [RevisionWeights](#) propriedade de um `AWS::Lambda::WebFunctionEndpoint` recurso. AWS SAM converte o mapa em uma lista de objetos, cada um com a `RevisionId` e a `Weight`

Sintaxe

Para declarar essa entidade em seu modelo AWS Serverless Application Model (AWS SAM), use a sintaxe a seguir.

YAML

```
revision-id: Integer
revision-id: Integer
```

Exemplos

Divisão canária

O exemplo a seguir envia 90% do tráfego para `rev-abc123` e 10% para `rev-def456`.

```
AutoDeploymentMode: Disabled
RevisionWeights:
  rev-abc123: 90
  rev-def456: 10
```

AWS SAM transforma isso na seguinte propriedade de endpoint:

```
RevisionWeights:
  - RevisionId: rev-abc123
    Weight: 90
  - RevisionId: rev-def456
    Weight: 10
```

Gerado CloudFormation recursos para AWS SAM

Esta seção fornece detalhes sobre os CloudFormation recursos que são criados ao AWS SAM processar seu AWS modelo. O conjunto de CloudFormation recursos AWS SAM gerados difere dependendo dos cenários que você especificar. Um cenário é a combinação de recursos AWS SAM e propriedades especificados em seu arquivo de modelo. Você pode referenciar os recursos CloudFormation gerados em outro lugar no seu arquivo de modelo, da mesma forma que você faz referência aos recursos que você declara explicitamente em seu arquivo de modelo.

Por exemplo, se você especificar um recurso `AWS::Serverless::Function` em seu arquivo de modelo AWS SAM, AWS SAM sempre gera um recurso base `AWS::Lambda::Function`. Se você também especificar a `AutoPublishAlias` propriedade opcional, AWS SAM também gera `AWS::Lambda::Alias` e `AWS::Lambda::Version` recursos.

Esta seção lista os cenários e os CloudFormation recursos que eles geram e mostra como referenciar os CloudFormation recursos gerados em seu arquivo AWS SAM de modelo.

Referência gerada CloudFormation recursos

Você tem duas opções para referenciar CloudFormation recursos gerados em seu arquivo AWS SAM de modelo, por `LogicalId` ou por propriedade referenciável.

Referência gerada CloudFormation recursos por `LogicalId`

Os CloudFormation recursos que AWS SAM geram cada um têm um [LogicalId](#), que é um identificador alfanumérico (A-Z, a-z, 0-9) exclusivo em um arquivo de modelo. AWS SAM usa os `LogicalIds` AWS SAM recursos em seu arquivo de modelo para construir `LogicalIds` os CloudFormation recursos que ele gera. Você pode usar o `LogicalId` de um CloudFormation recurso gerado para acessar as propriedades desse recurso em seu arquivo de modelo, assim como faria com um CloudFormation recurso que você declarou explicitamente. Para obter mais informações sobre `LogicalIds` em CloudFormation e AWS SAM modelos, consulte [Recursos](#) no Guia do AWS CloudFormation usuário.

Note

Alguns recursos gerados incluem um valor `LogicalIds` de hash exclusivo para evitar conflitos de namespace. Esses recursos são derivados quando a pilha é criada `LogicalIds`. Você pode recuperá-los somente após a criação da pilha usando o Console de gerenciamento da AWS, AWS CLI, ou um dos AWS SDKs. Não recomendamos referenciar esses recursos `LogicalId` porque os valores de hash podem mudar.

Referência gerada CloudFormation recursos por propriedade referenciável

Para alguns recursos gerados, AWS SAM fornece uma propriedade referenciável do AWS SAM recurso. Você pode usar essa propriedade para referenciar um CloudFormation recurso gerado e suas propriedades em seu arquivo AWS SAM de modelo.

Note

Nem todos os CloudFormation recursos gerados têm propriedades referenciáveis. Para esses recursos, você deve usar LogicalId o.

Gerado CloudFormation cenários de recursos

A tabela a seguir resume os AWS SAM recursos e propriedades que compõem os cenários que geram CloudFormation recursos. Os tópicos na coluna Cenários fornecem detalhes sobre os CloudFormation recursos adicionais que são AWS SAM gerados para esse cenário.

AWS SAM recurso	CloudFormation Recurso básico	Cenários
<u>AWS::Serverless::Api</u>	<u>AWS::ApiGateway::RestApi</u>	<ul style="list-style-type: none"> <u>DomainName propriedade é especificada</u> <u>UsagePlan propriedade é especificada</u>
<u>AWS::Serverless::Application</u>	<u>AWS::CloudFormation::Stack</u>	<ul style="list-style-type: none"> Além de gerar o CloudFormation recurso básico, não há cenários adicionais para esse recurso sem servidor.
<u>AWS::Serverless::CapacityProvider</u>	<u>AWS::Lambda::CapacityProvider</u>	<ul style="list-style-type: none"> <u>OperatorRole propriedade não é especificada</u>
<u>AWS::Serverless::MicrovmImage</u>	<u>AWS::Lambda::MicrovmImage</u>	<ul style="list-style-type: none"> <u>BuildRoleArn propriedade não é especificada</u>
<u>AWS::Serverless::Netwo</u>	<u>AWS::Lambda::Netwo</u>	<ul style="list-style-type: none"> <u>OperatorRole propriedade não é especificada</u>

AWS SAM recurso	CloudFormation Recurso básico	Cenários
<u>etworkConnector</u>	<u>rkConnect</u> <u>or</u>	
<u>AWS::Serverless::WebFunction</u>	<u>AWS::Lambda::WebFunction</u>	<ul style="list-style-type: none"> • <u>ExecutionRoleArn propriedade não é especificada</u>
		<ul style="list-style-type: none"> • <u>AutoPublishAlias propriedade é especificada</u> • <u>A propriedade da função não foi especificada</u> • <u>DeploymentPreference propriedade é especificada</u> • <u>Uma fonte de eventos da Api é especificada</u> • <u>Uma fonte de HttpApi eventos é especificada</u> • <u>Uma fonte de eventos de streaming é especificada</u> • <u>Uma fonte de eventos de ponte de eventos (ou barramento de eventos) é especificada</u> • <u>Uma fonte de IoTRule eventos é especificada</u> • <u>OnSuccess (ou OnFailure) a propriedade é especificada para eventos do Amazon SNS</u> • <u>OnSuccess (ou OnFailure) a propriedade é especificada para eventos do Amazon SQS</u>
<u>AWS::Serverless::Function</u>	<u>AWS::Lambda::Function</u>	

AWS SAM recurso	CloudFormation Recurso básico	Cenários
<u>AWS::Serverless::HttpApi</u>	<u>AWS::ApiGatewayV2::Api</u>	<ul style="list-style-type: none"> • <u>StageName propriedade é especificada</u> • <u>StageName propriedade não é especificada</u> • <u>DomainName propriedade é especificada</u>
<u>AWS::Serverless::LayerVersion</u>	<u>AWS::Lambda::LayerVersion</u>	<ul style="list-style-type: none"> • Além de gerar o CloudFormation recurso básico, não há cenários adicionais para esse recurso sem servidor.
<u>AWS::Serverless::SimpleTable</u>	<u>AWS::DynamoDB::Table</u>	<ul style="list-style-type: none"> • Além de gerar o CloudFormation recurso básico, não há cenários adicionais para esse recurso sem servidor.
<u>AWS::Serverless::StateMachine</u>	<u>AWS::StepFunctions::StateMachine</u>	<ul style="list-style-type: none"> • <u>A propriedade da função não foi especificada</u> • <u>Uma origem de eventos de API é especificada</u> • <u>Uma fonte de eventos de ponte de eventos (ou barramento de eventos) é especificada</u>

Tópicos

- [CloudFormation recursos gerados quando AWS::Serverless::Api é especificado](#)
- [CloudFormation recursos gerados quando AWS::Serverless::Application é especificado](#)
- [CloudFormation recursos gerados quando AWS::Serverless::CapacityProvider é especificado](#)
- [CloudFormation recursos gerados quando você especifica AWS::Serverless::Connector](#)
- [CloudFormation recursos gerados quando AWS::Serverless::Function é especificado](#)
- [CloudFormation recursos gerados quando AWS::Serverless::GraphQLApi é especificado](#)

- [CloudFormation recursos gerados quando AWS::Serverless::HttpApi é especificado](#)
- [CloudFormation recursos gerados quando AWS::Serverless::MicrovmImage é especificado](#)
- [CloudFormation recursos gerados quando AWS::Serverless::NetworkConnector é especificado](#)
- [CloudFormation recursos gerados quando AWS::Serverless::WebFunction é especificado](#)
- [CloudFormation recursos gerados quando AWS::Serverless::WebSocketApi é especificado](#)
- [CloudFormation recursos gerados quando AWS::Serverless::LayerVersion é especificado](#)
- [CloudFormation recursos gerados quando AWS::Serverless::SimpleTable é especificado](#)
- [CloudFormation recursos gerados quando AWS::Serverless::StateMachine é especificado](#)

CloudFormation recursos gerados quando AWS::Serverless::Api é especificado

Quando um `AWS::Serverless::Api` é especificado, AWS Serverless Application Model (AWS SAM) sempre gera um CloudFormation recurso `AWS::ApiGateway::RestApi` básico. Além disso, ele sempre gera um `AWS::ApiGateway::Stage` e um recurso `AWS::ApiGateway::Deployment`.

AWS::ApiGateway::RestApi

LogicalId: `<api-LogicalId>`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

AWS::ApiGateway::Stage

LogicalId: `<api-LogicalId><stage-name>Stage`

`<stage-name>` é a string para a qual a propriedade `StageName` está definida. Por exemplo, se você definir `StageName` como `Gamma`, será `LogicalIdMyRestApiGammaStage`.

Propriedade referenciável: `<api-LogicalId>.Stage`

AWS::ApiGateway::Deployment

LogicalId: `<api-LogicalId>Deployment<sha>`

O `<sha>` é um valor de hash exclusivo que é gerado quando a pilha é criada. Por exemplo, `MyRestApiDeployment926eeb5ff1`.

Propriedade referenciável: `<api-LogicalId>.Deployment`

Além desses CloudFormation recursos, quando `AWS::Serverless::Api` especificado, AWS SAM gera CloudFormation recursos adicionais para os seguintes cenários.

Cenários

- [DomainName propriedade é especificada](#)
- [UsagePlan propriedade é especificada](#)

DomainName propriedade é especificada

Quando a `DomainName` propriedade da `Domain` propriedade de um `AWS::Serverless::Api` é especificada, AWS SAM gera o `AWS::ApiGateway::DomainName` CloudFormation recurso.

AWS::ApiGateway::DomainName

LogicalId: `ApiGatewayDomainName<sha>`

O `<sha>` é um valor de hash exclusivo que é gerado quando a pilha é criada. Por exemplo: `ApiGatewayDomainName926eeb5ff1`.

Propriedade referenciável: `<api-LogicalId>.DomainName`

UsagePlan propriedade é especificada

Quando a `UsagePlan` propriedade da `Auth` propriedade de um `AWS::Serverless::Api` é especificada, AWS SAM gera os seguintes CloudFormation recursos:

`AWS::ApiGateway::UsagePlan`
`AWS::ApiGateway::UsagePlanKey`,
`AWS::ApiGateway::ApiKey` e.

AWS::ApiGateway::UsagePlan

LogicalId: `<api-LogicalId>UsagePlan`

Propriedade referenciável: `<api-LogicalId>.UsagePlan`

AWS::ApiGateway::UsagePlanKey

LogicalId: `<api-LogicalId>UsagePlanKey`

Propriedade referenciável: `<api-LogicalId>.UsagePlanKey`

AWS::ApiGateway::ApiKey

LogicalId: `<api-LogicalId>ApiKey`

Propriedade referenciável: `<api-LogicalId>.ApiKey`

CloudFormation recursos gerados quando `AWS::Serverless::Application` é especificado

Quando um `AWS::Serverless::Application` é especificado, AWS Serverless Application Model (AWS SAM) gera um CloudFormation recurso `AWS::CloudFormation::Stack` básico.

AWS::CloudFormation::Stack

LogicalId: `<application-LogicalId>`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

CloudFormation recursos gerados quando `AWS::Serverless::CapacityProvider` é especificado

Quando um `AWS::Serverless::CapacityProvider` é especificado, AWS Serverless Application Model (AWS SAM) gera um recurso `AWS::Lambda::CapacityProvider` com base CloudFormation .

AWS::Lambda::CapacityProvider

LogicalId: `<capacityprovider-LogicalId>`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

Além desse CloudFormation recurso, quando `AWS::Serverless::CapacityProvider` especificado, AWS SAM também gera CloudFormation recursos para os seguintes cenários:

Cenários

- [OperatorRole propriedade não é especificada](#)

OperatorRole propriedade não é especificada

Quando a `OperatorRole` propriedade de um não `AWS::Serverless::CapacityProvider` é especificada, AWS SAM gera um `AWS::IAM::Role` CloudFormation recurso com a política `AWSLambdaManagedEC2ResourceOperator` gerenciada anexada.

AWS::IAM::Role

LogicalId: `<capacityprovider-LogicalId>OperatorRole`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

CloudFormation recursos gerados quando você especifica AWS::Serverless::Connector

Note

Quando você define conectores por meio da propriedade `Connectors` incorporada, ela é primeiro transformada em um recurso `AWS::Serverless::Connector` antes de gerar esses recursos.

Quando você especifica um `AWS::Serverless::Connector` recurso em um AWS SAM modelo, AWS SAM gera os seguintes AWS CloudFormation recursos conforme necessário.

AWS::IAM::ManagedPolicy

LogicalId: `<connector-LogicalId>Policy`

Propriedade referenciável: N/A (Para referenciar esse CloudFormation recurso, você deve usar o.) `LogicalId`

AWS::SNS::TopicPolicy

LogicalId: `<connector-LogicalId>TopicPolicy`

Propriedade referenciável: N/A (Para referenciar esse CloudFormation recurso, você deve usar o.) LogicalId

AWS::SQS::QueuePolicy

LogicalId: *<connector-LogicalId>*QueuePolicy

Propriedade referenciável: N/A (Para referenciar esse CloudFormation recurso, você deve usar o.) LogicalId

AWS::Lambda::Permission

LogicalId: *<connector-LogicalId>**<permission>*LambdaPermission

<permission> é uma permissão especificada pela propriedade Permissions. Por exemplo, *.Write*

Propriedade referenciável: N/A (Para referenciar esse CloudFormation recurso, você deve usar o.) LogicalId

CloudFormation recursos gerados quando AWS::Serverless::Function é especificado

Quando um AWS::Serverless::Function é especificado, AWS Serverless Application Model (AWS SAM) sempre cria um CloudFormation recurso AWS::Lambda::Function básico.

AWS::Lambda::Function

LogicalId: *<function-LogicalId>*

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso) CloudFormation

Além desse CloudFormation recurso, quando AWS::Serverless::Function especificado, AWS SAM também gera CloudFormation recursos para os seguintes cenários.

Cenários

- [Propriedades da função principal](#)
- [Origens de eventos](#)

- [Configuração do evento](#)

Propriedades da função principal

Os cenários a seguir geram CloudFormation recursos com base nas propriedades da função principal:

A propriedade da função não foi especificada

Quando a Role propriedade de um não `AWS::Serverless::Function` é especificada, AWS SAM gera um `AWS::IAM::Role` CloudFormation recurso.

AWS::IAM::Role

LogicalId: <function-LogicalId>Role

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AutoPublishAlias propriedade é especificada

Quando a AutoPublishAlias propriedade de um `AWS::Serverless::Function` é especificada, AWS SAM gera os seguintes CloudFormation recursos: `AWS::Lambda::Alias` `AWS::Lambda::Version` e.

AWS::Lambda::Alias

LogicalId: <function-LogicalId>Alias<alias-name>

<alias-name> é a string que está definida como AutoPublishAlias. Por exemplo, se você AutoPublishAlias definir comolive, LogicalId é: *MyFunction Aliaslive*.

Propriedade referenciável: *<function-LogicalId>.Alias*

AWS::Lambda::Version

LogicalId: <function-LogicalId>Version<sha>

O *<sha>* é um valor de hash exclusivo que é gerado quando a pilha é criada. Por exemplo, *MyFunction Versão926eeb5ff1*.

Propriedade referenciável: *<function-LogicalId>.Version*

Para obter informações adicionais sobre a `AutoPublishAlias` propriedade, consulte a [seção Propriedades do AWS::Serverless::Function](#).

`DeploymentPreference` propriedade é especificada

Quando a `DeploymentPreference` propriedade de um `AWS::Serverless::Function` é especificada, AWS SAM gera os seguintes CloudFormation recursos:

`AWS::CodeDeploy::Application` `AWS::CodeDeploy::DeploymentGroup` e. Além disso, se a `Role` propriedade do `DeploymentPreference` objeto não for especificada, AWS SAM também gera um `AWS::IAM::Role` CloudFormation recurso.

AWS::CodeDeploy::Application

LogicalId: `ServerlessDeploymentApplication`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

AWS::CodeDeploy::DeploymentGroup

LogicalId: *<function-LogicalId>*`DeploymentGroup`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

AWS::IAM::Role

LogicalId: `CodeDeployServiceRole`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

`FunctionUrlConfig` propriedade é especificada

Quando a `FunctionUrlConfig` propriedade é especificada, AWS SAM gera CloudFormation recursos diferentes com base no `AuthType`.

Quando `AuthType`: `NONE` é especificado, AWS SAM gera os seguintes CloudFormation recursos:

AWS::Lambda::Permission (Invocar acesso)

LogicalId: *<function-LogicalId>*URLInvokeAllowPublicAccess

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS::Lambda::Permission (Acesso público)

LogicalId: *<function-LogicalId>*UrlPublicPermissions

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS::Lambda::Url

LogicalId: *<function-LogicalId>*Url

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

Quando AuthType: AWS_IAM é especificado, AWS SAM gera somente:

AWS::Lambda::Url

LogicalId: *<function-LogicalId>*Url

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

Para obter informações adicionais sobre a FunctionUrlConfig propriedade, consulte [FunctionUrlConfig](#).

Origens de eventos

Os cenários a seguir geram CloudFormation recursos com base em fontes de eventos:

Uma fonte de eventos da Api é especificada

Quando a Event propriedade de um AWS::Serverless::Function é definida comoApi, mas a RestApiId propriedade não é especificada, AWS SAM gera o AWS::ApiGateway::RestApi CloudFormation recurso.

AWS::ApiGateway::RestApi

LogicalId: ServerlessRestApi

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

Uma fonte de `HttpApi` eventos é especificada

Quando a `Event` propriedade de um `AWS::Serverless::Function` é definida como `HttpApi`, mas a `ApiId` propriedade não é especificada, AWS SAM gera o `AWS::ApiGatewayV2::Api` CloudFormation recurso.

AWS::ApiGatewayV2::Api

LogicalId: ServerlessHttpApi

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

Uma fonte de eventos de streaming é especificada

Quando a `Event` propriedade de um `AWS::Serverless::Function` é definida como um dos tipos de streaming, AWS SAM gera o `AWS::Lambda::EventSourceMapping` CloudFormation recurso. Isso se aplica aos seguintes tipos: `DynamoDB`, `Kinesis`, `MQ`, `MSK` e `SQS`.

AWS::Lambda::EventSourceMapping

LogicalId: *<function-LogicalId><event-LogicalId>*

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

Uma fonte de eventos de ponte de eventos (ou barramento de eventos) é especificada

Quando a `Event` propriedade de um `AWS::Serverless::Function` é definida como um dos tipos de ponte de eventos (ou barramento de eventos), AWS SAM gera o `AWS::Events::Rule` CloudFormation recurso. Isso se aplica aos seguintes tipos: `EventBridgeRule`, `Schedule` e `CloudWatchEvents`.

AWS::Events::Rule

LogicalId: <function-LogicalId><event-LogicalId>

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

Uma fonte de IoTRule eventos é especificada

Quando a Event propriedade de an AWS::Serverless::Function é definida como IoTRule, AWS SAM gera o AWS::IoT::TopicRule CloudFormation recurso.

AWS::IoT::TopicRule

LogicalId: <function-LogicalId><event-LogicalId>

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

Configuração do evento

Os cenários a seguir geram CloudFormation recursos com base na configuração do evento:

OnSuccess (ou OnFailure) a propriedade é especificada para eventos do Amazon SNS

Quando a propriedade OnSuccess (ouOnFailure) da DestinationConfig propriedade da EventInvokeConfig propriedade de an AWS::Serverless::Function é especificada e o tipo de destino é, SNS mas o ARN de destino não está especificado, AWS SAM gera os seguintes CloudFormation recursos: e. AWS::Lambda::EventInvokeConfig AWS::SNS::Topic

AWS::Lambda::EventInvokeConfig

LogicalId: <function-LogicalId>EventInvokeConfig

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS::SNS::Topic

*LogicalId: <function-LogicalId>OnSuccessTopic (ou
<function-LogicalId>OnFailureTopic)*

Propriedade referenciável: *<function-LogicalId>*.DestinationTopic

Se ambos OnSuccess e OnFailure forem especificados para um evento do Amazon SNS, para distinguir entre os recursos gerados, você deverá usar o LogicalId.

OnSuccess (ou OnFailure) a propriedade é especificada para eventos do Amazon SQS

Quando a propriedade OnSuccess (ou OnFailure) da DestinationConfig propriedade da EventInvokeConfig propriedade de an AWS::Serverless::Function é especificada e o tipo de destino é, SQS mas o ARN de destino não está especificado, AWS SAM gera os seguintes CloudFormation recursos: e. AWS::Lambda::EventInvokeConfig AWS::SQS::Queue

AWS::Lambda::EventInvokeConfig

LogicalId: *<function-LogicalId>*EventInvokeConfig

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS::SQS::Queue

LogicalId: *<function-LogicalId>*OnSuccessQueue (ou
*<function-LogicalId>*OnFailureQueue)

Propriedade referenciável: *<function-LogicalId>*.DestinationQueue

Se ambos OnSuccess e OnFailure forem especificados para um evento do Amazon SQS, para distinguir entre os recursos gerados, você deverá usar o LogicalId.

CloudFormation recursos gerados quando AWS::Serverless::GraphQLApi é especificado

Quando você especifica um AWS::Serverless::GraphQLApi recurso em um modelo AWS Serverless Application Model (AWS SAM), AWS SAM sempre cria os seguintes AWS CloudFormation recursos básicos.

AWS::AppSync::DataSource

LogicalId: *<graphqlapi-LogicalId><datasource-RelativeId><datasource-Type>*DataSource

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS::AppSync::FunctionConfiguration

LogicalId: <graphqlapi-LogicalId><function-RelativeId>

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS::AppSync::GraphQLApi

LogicalId: <graphqlapi-LogicalId>

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS::AppSync::GraphQLSchema

LogicalId: <graphqlapi-LogicalId>Schema

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS::AppSync::Resolver

LogicalId: <graphqlapi-LogicalId><OperationType><resolver-RelativeId>

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

Além desses CloudFormation recursos, quando AWS::Serverless::GraphQLApi especificado, também AWS SAM pode gerar os seguintes CloudFormation recursos.

AWS::AppSync::ApiCache

LogicalId: <graphqlapi-LogicalId>ApiCache

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS::AppSync::ApiKey

LogicalId: <graphqlapi-LogicalId><apikey-RelativeId>

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS::AppSync::DomainName

LogicalId: <graphqlapi-LogicalId>DomainName

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS::AppSync::DomainNameApiAssociation

LogicalId: <graphqlapi-LogicalId>DomainNameApiAssociation

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS SAM também pode usar o **AWS::Serverless::Connector** recurso para provisionar permissões. Para obter mais informações, consulte [CloudFormation recursos gerados quando você especifica AWS::Serverless::Connector](#).

CloudFormation recursos gerados quando **AWS::Serverless::HttpApi** é especificado

Quando um **AWS::Serverless::HttpApi** é especificado, AWS Serverless Application Model (AWS SAM) gera um CloudFormation recurso **AWS::ApiGatewayV2::Api** básico.

AWS::ApiGatewayV2::Api

LogicalId: <httpapi-LogicalId>

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

Além desse CloudFormation recurso, quando **AWS::Serverless::HttpApi** especificado, AWS SAM também gera CloudFormation recursos para os seguintes cenários:

Cenários

- [StageName propriedade é especificada](#)

- [StageName propriedade não é especificada](#)
- [DomainName propriedade é especificada](#)

StageName propriedade é especificada

Quando a StageName propriedade de um `AWS::Serverless::HttpApi` é especificada, AWS SAM gera o `AWS::ApiGatewayV2::Stage` CloudFormation recurso.

AWS::ApiGatewayV2::Stage

LogicalId: `<httpapi-LogicalId><stage-name>Stage`

`<stage-name>` é a string para a qual a propriedade StageName está definida. Por exemplo, se você StageName definir como `Gamma`, LogicalId é: `MyHttpApiGamma Estágio`.

Propriedade referenciável: `<httpapi-LogicalId>.Stage`

StageName propriedade não é especificada

Quando a StageName propriedade de um não `AWS::Serverless::HttpApi` é especificada, AWS SAM gera o `AWS::ApiGatewayV2::Stage` CloudFormation recurso.

AWS::ApiGatewayV2::Stage

LogicalId: `<httpapi-LogicalId>ApiGatewayDefaultStage`

Propriedade referenciável: `<httpapi-LogicalId>.Stage`

DomainName propriedade é especificada

Quando a DomainName propriedade da Domain propriedade de an `AWS::Serverless::HttpApi` é especificada, AWS SAM gera o `AWS::ApiGatewayV2::DomainName` CloudFormation recurso.

AWS::ApiGatewayV2::DomainName

LogicalId: `ApiGatewayDomainNameV2<sha>`

O `<sha>` é um valor de hash exclusivo que é gerado quando a pilha é criada. Por exemplo, `ApiGatewayDomainNameV2926eeb5ff1`.

Propriedade referenciável: `<httpapi-LogicalId>.DomainName`

CloudFormation recursos gerados quando `AWS::Serverless::MicrovmImage` é especificado

Quando você especifica um `AWS::Serverless::MicrovmImage`, AWS Serverless Application Model (AWS SAM) gera um CloudFormation recurso `AWS::Lambda::MicrovmImage` básico.

`AWS::Lambda::MicrovmImage`

LogicalId: `<microvmimage-LogicalId>`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este CloudFormation recurso)

Além desse CloudFormation recurso, quando `AWS::Serverless::MicrovmImage` especificado, AWS SAM também gera CloudFormation recursos para os seguintes cenários:

Cenários

- [BuildRoleArn propriedade não é especificada](#)

BuildRoleArn propriedade não é especificada

Quando você não especifica a `BuildRoleArn` propriedade de um `AWS::Serverless::MicrovmImage`, AWS SAM gera um `AWS::IAM::Role` CloudFormation recurso com uma política embutida que concede as seguintes permissões:

- `s3:GetObject`— com escopo definido para o bucket Amazon S3 a partir de `CodeUri`
- `logs:CreateLogGroup`, `logs:CreateLogStream`, `logs:PutLogEvents` — para registros de construção

A política de confiança da função permite que o diretor do `lambda.amazonaws.com` serviço assuma a função.

`AWS::IAM::Role`

LogicalId: `<microvmimage-LogicalId>BuildRole`

CloudFormation recursos gerados quando AWS::Serverless::NetworkConnector é especificado

Quando você especifica um `AWS::Serverless::NetworkConnector`, AWS Serverless Application Model (AWS SAM) gera um CloudFormation recurso `AWS::Lambda::NetworkConnector` básico.

AWS::Lambda::NetworkConnector

LogicalId: `<networkconnector-LogicalId>`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este CloudFormation recurso)

Além desse CloudFormation recurso, quando `AWS::Serverless::NetworkConnector` especificado, AWS SAM também gera CloudFormation recursos para os seguintes cenários:

Cenários

- [OperatorRole propriedade não é especificada](#)

OperatorRole propriedade não é especificada

Quando você não especifica a `OperatorRole` propriedade de um `AWS::Serverless::NetworkConnector`, AWS SAM gera um `AWS::IAM::Role` CloudFormation recurso com a política `AWSLambdaNetworkConnectorOperatorPolicy` AWS gerenciada anexada.

A política de confiança da função permite que o diretor do `lambda.amazonaws.com` serviço assuma a função.

AWS::IAM::Role

LogicalId: `<networkconnector-LogicalId>OperatorRole`

CloudFormation recursos gerados quando `AWS::Serverless::WebFunction` é especificado

Quando você especifica um `AWS::Serverless::WebFunction`, AWS Serverless Application Model (AWS SAM) gera os seguintes CloudFormation recursos básicos.

`AWS::Lambda::WebFunction`

LogicalId: `<webfunction-LogicalId>`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este CloudFormation recurso)

`AWS::Lambda::WebFunctionRevision`

LogicalId: `<webfunction-LogicalId>Revision`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este CloudFormation recurso)

`AWS::Lambda::WebFunctionEndpoint`

LogicalId: `<webfunction-LogicalId>Endpoint`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este CloudFormation recurso)

Todos os três recursos usam o mesmo nome de função. Se você não especificar a `FunctionName` propriedade, AWS SAM gera um nome do formulário `<webfunction-LogicalId>-<stack-id-prefix>`, onde estão `stack-id-prefix` os primeiros 8 caracteres do identificador exclusivo do ID da pilha. AWS SAM encurta o ID lógico, se necessário, para que o nome tenha no máximo 64 caracteres.

Além desses CloudFormation recursos, quando `AWS::Serverless::WebFunction` especificado, AWS SAM também gera CloudFormation recursos para os seguintes cenários:

Cenários

- [ExecutionRoleArn propriedade não é especificada](#)

ExecutionRoleArn propriedade não é especificada

Quando você não especifica a ExecutionRoleArn propriedade de um `AWS::Serverless::WebFunction`, AWS SAM gera um `AWS::IAM::Role` CloudFormation recurso com a política `AWSLambdaBasicExecutionRole` AWS gerenciada anexada.

A política de confiança da função permite que o diretor do `lambda.amazonaws.com` serviço assuma a função.

AWS::IAM::Role

LogicalId: <webfunction-LogicalId>Role

CloudFormation recursos gerados quando `AWS::Serverless::WebSocketApi` é especificado

Quando um `AWS::Serverless::WebSocketApi` é especificado, AWS Serverless Application Model (AWS SAM) gera um CloudFormation recurso `AWS::ApiGatewayV2::Api` básico. Além disso, ele sempre gera um `AWS::ApiGatewayV2::Stage` recurso e, para cada rota definida na `Routes` propriedade `AWS::ApiGatewayV2::Route` `AWS::ApiGatewayV2::Integration`, AWS SAM gera e `AWS::Lambda::Permission` recursos.

AWS::ApiGatewayV2::Api

LogicalId: <websocketapi-LogicalId>

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

AWS::ApiGatewayV2::Stage

LogicalId: <websocketapi-LogicalId><stage-name>Stage

<stage-name> é a string para a qual a propriedade `StageName` está definida. Se não `StageName` for especificado, o valor padrão será `default`. Por exemplo, se você definir `StageName` como `prod`, será `LogicalIdMyWebSocketApiprodStage`.

Propriedade referenciável: *<websocketapi-LogicalId>.Stage*

AWS::ApiGatewayV2::Route

LogicalId: <websocketapi-LogicalId><normalized-route-key>Route

<normalized-route-key> é a chave de rota com caracteres especiais removidos. Por exemplo, para a chave de rota `$connect`, o `LogicalId` é *MyWebSocketApi*`connectRoute`.

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

AWS::ApiGatewayV2::Integration

*LogicalId: <websocketapi-LogicalId><normalized-route-key>*Integration

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

AWS::Lambda::Permission

*LogicalId: <websocketapi-LogicalId><normalized-route-key>*Permission

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

Além desses CloudFormation recursos, quando `AWS::Serverless::WebSocketApi` especificado, AWS SAM também gera CloudFormation recursos para os seguintes cenários:

Cenários

- [A propriedade Auth é especificada com AuthType definida como CUSTOM](#)
- [DomainName propriedade é especificada](#)
- [BasePath propriedade é especificada](#)
- [A propriedade Route53 é especificada](#)

A propriedade Auth é especificada com AuthType definida como CUSTOM

Quando a Auth propriedade de um `AWS::Serverless::WebSocketApi` é especificada com `AuthType` set to `CUSTOM`, AWS SAM gera um `AWS::ApiGatewayV2::Authorizer` CloudFormation recurso. Se não `InvokeRole` for especificado, AWS SAM também gera um `AWS::Lambda::Permission` recurso para permitir que o API Gateway invoque a função autorizadora.

AWS::ApiGatewayV2::Authorizer

*LogicalId: <websocketapi-LogicalId>*Authorizer

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

AWS::Lambda::Permission(quando não **InvokeRole** é especificado)

LogicalId: *<websocketapi-LogicalId>AuthorizerPermission*

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

DomainName propriedade é especificada

Quando a DomainName propriedade da Domain propriedade de an
AWS::Serverless::WebSocketApi é especificada, AWS SAM gera o
AWS::ApiGatewayV2::DomainName CloudFormation recurso.

AWS::ApiGatewayV2::DomainName

LogicalId: ApiGatewayDomainNameV2*<sha>*

O *<sha>* é um valor de hash exclusivo que é gerado quando a pilha é criada. Por exemplo:
ApiGatewayDomainNameV2*926eeb5ff1*.

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

BasePath propriedade é especificada

Quando a BasePath propriedade da Domain propriedade de an
AWS::Serverless::WebSocketApi é especificada, AWS SAM gera
AWS::ApiGatewayV2::ApiMapping CloudFormation recursos, um para cada caminho base
especificado.

AWS::ApiGatewayV2::ApiMapping

LogicalId: *<websocketapi-LogicalId><basepath>ApiMapping*

<basepath> é o valor do caminho base. Por exemplo, se você especificar v1 como um caminho
base, o LogicalId é *MyWebSocketApi*v1ApiMapping.

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

A propriedade Route53 é especificada

Quando a Route53 propriedade da Domain propriedade de um `AWS::Serverless::WebSocketApi` é especificada, AWS SAM gera um `AWS::Route53::RecordSetGroup` CloudFormation recurso.

AWS::Route53::RecordSetGroup

LogicalId: RecordSetGroup<sha>

O <sha> é um valor de hash exclusivo que é gerado quando a pilha é criada. Por exemplo: RecordSetGroup926eeb5ff1.

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

CloudFormation recursos gerados quando AWS::Serverless::LayerVersion é especificado

Quando um `AWS::Serverless::LayerVersion` é especificado, AWS Serverless Application Model (AWS SAM) gera um CloudFormation recurso `AWS::Lambda::LayerVersion` básico.

AWS::Lambda::LayerVersion

LogicalId: <layerversion-LogicalId>

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

CloudFormation recursos gerados quando AWS::Serverless::SimpleTable é especificado

Quando um `AWS::Serverless::SimpleTable` é especificado, AWS Serverless Application Model (AWS SAM) gera um CloudFormation recurso `AWS::DynamoDB::Table` básico.

AWS::DynamoDB::Table

LogicalId: <simpletable-LogicalId>

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

CloudFormation recursos gerados quando AWS::Serverless::StateMachine é especificado

Quando um AWS::Serverless::StateMachine é especificado, AWS Serverless Application Model (AWS SAM) gera um recurso AWS::StepFunctions::StateMachine com base CloudFormation .

AWS::StepFunctions::StateMachine

LogicalId: <statemachine-LogicalId>

Propriedade referenciável: N/A (você deve usar o LogicalId para referenciar este recurso)
CloudFormation

Além desse CloudFormation recurso, quando AWS::Serverless::StateMachine especificado, AWS SAM também gera CloudFormation recursos para os seguintes cenários:

Cenários

- [A propriedade da função não foi especificada](#)
- [Uma origem de eventos de API é especificada](#)
- [Uma fonte de eventos de ponte de eventos \(ou barramento de eventos\) é especificada](#)

A propriedade da função não foi especificada

Quando a Role propriedade de um não AWS::Serverless::StateMachine é especificada, AWS SAM gera um AWS::IAM::Role CloudFormation recurso.

AWS::IAM::Role

LogicalId: <statemachine-LogicalId>Role

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

Uma origem de eventos de API é especificada

Quando a `Event` propriedade de um `AWS::Serverless::StateMachine` é definida como `Api`, mas a `RestApiId` propriedade não é especificada, AWS SAM gera o `AWS::ApiGateway::RestApi` CloudFormation recurso.

AWS::ApiGateway::RestApi

LogicalId: `ServerlessRestApi`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

Uma fonte de eventos de ponte de eventos (ou barramento de eventos) é especificada

Quando a `Event` propriedade de um `AWS::Serverless::StateMachine` é definida como um dos tipos de ponte de eventos (ou barramento de eventos), AWS SAM gera o `AWS::Events::Rule` CloudFormation recurso. Isso se aplica aos seguintes tipos: `EventBridgeRule`, `Schedule` e `CloudWatchEvents`.

AWS::Events::Rule

LogicalId: `<statemachine-LogicalId><event-LogicalId>`

Propriedade referenciável: N/A (você deve usar o `LogicalId` para referenciar este recurso)
CloudFormation

Atributos de recursos suportados por AWS SAM

Atributos de recursos são atributos que você pode adicionar AWS SAM e CloudFormation recursos para controlar comportamentos e relacionamentos adicionais. Para obter mais informações sobre atributos de recursos, consulte [Referência de atributos](#) de recursos no Guia AWS CloudFormation do usuário.

AWS SAM suportam um subconjunto de atributos de recursos que são definidos por CloudFormation. Dos atributos de recursos suportados, alguns são copiados somente para o CloudFormation

recurso gerado base do AWS SAM recurso correspondente e alguns são copiados para todos os CloudFormation recursos gerados resultantes do recurso correspondente AWS SAM . Para obter mais informações sobre CloudFormation os recursos gerados a partir AWS SAM dos recursos correspondentes, consulte [Gerado CloudFormation recursos para AWS SAM](#).

A tabela a seguir resume o suporte a atributos de recursos por AWS SAM, de acordo com a [Exceções](#) lista abaixo.

Atributos de recursos	Recurso(s) gerado(s) pelo destino
DependsOn Metadados ^{1, 2}	Somente recurso CloudFormation gerado por base. Para obter informações sobre o mapeamento entre AWS SAM recursos e CloudFormation recursos básicos, consulte Gerado CloudFormation cenários de recursos .
Condition DeletionPolicy UpdateReplacePolicy	Todos os CloudFormation recursos gerados a partir do AWS SAM recurso correspondente. Para obter informações sobre cenários para CloudFormation recursos gerados, consulte Gerado CloudFormation cenários de recursos .

Observações:

1. Para obter mais informações sobre como usar o atributo de recurso Metadata com o tipo de recurso `AWS::Serverless::Function`, consulte [Criando funções de eventos Lambda com tempos de execução personalizados em AWS SAM](#).
2. Para obter mais informações sobre como usar o atributo de recurso Metadata com o tipo de recurso `AWS::Serverless::LayerVersion`, consulte [Construindo camadas Lambda em AWS SAM](#).

Exceções

Há várias exceções às regras de atributos de recursos descritas anteriormente:

- Para `AWS::Lambda::LayerVersion`, o campo personalizado AWS SAM-only `RetentionPolicy` define `DeletionPolicy` os CloudFormation recursos gerados. Isso tem uma

precedência maior do que `DeletionPolicy` ele mesmo. Se nenhum estiver definido, então, por padrão, será `DeletionPolicy` definido como `Retain`.

- Para `AWS::Lambda::Version`, se `DeletionPolicy` não especificado, o padrão será `Retain`.
- Para o cenário em que `DeploymentPreferences` é especificado para uma função sem servidor, os atributos do recurso não são copiados para os seguintes recursos gerados: `CloudFormation`
 - `AWS::CodeDeploy::Application`
 - `AWS::CodeDeploy::DeploymentGroup`
 - O `AWS::IAM::Role` chamado `CodeDeployServiceRole` que é criado para esse cenário
- Se seu AWS SAM modelo contiver várias funções com fontes de eventos de API criadas implicitamente, as funções compartilharão o `AWS::ApiGateway::RestApi` recurso gerado. Nesse cenário, se as funções tiverem atributos de recursos diferentes, então, para o `AWS::ApiGateway::RestApi` recurso gerado, AWS SAM copie os atributos do recurso de acordo com as seguintes listas priorizadas:
 - `UpdateReplacePolicy`:
 1. `Retain`
 2. `Snapshot`
 3. `Delete`
 - `DeletionPolicy`:
 1. `Retain`
 2. `Delete`

Extensões do API Gateway para AWS SAM

As extensões do API Gateway, projetadas especificamente para AWS, fornecem personalizações e funcionalidades adicionais para design e gerenciamento. APIs As extensões do API Gateway são extensões da especificação OpenAPI que oferecem suporte à autorização AWS específica e às integrações de API específicas do API Gateway. Para obter mais informações sobre extensões do API Gateway, consulte [Extensões do API Gateway para o OpenAPI](#).

AWS SAM suporta um subconjunto de extensões do API Gateway. Para ver quais extensões do API Gateway são compatíveis AWS SAM, consulte a tabela a seguir.

Extensão do API Gateway	Apoiado por AWS SAM
<u>x-amazon-apigateway-any-objeto de método</u>	Sim
<u>x-amazon-apigateway-apiPropriedade -key-source</u>	Não
<u>x-amazon-apigateway-auth Objeto</u>	Sim
<u>x-amazon-apigateway-authorizer Objeto</u>	Sim
<u>x-amazon-apigateway-authtype Propriedade</u>	Sim
<u>x-amazon-apigateway-binaryPropriedade -media-types</u>	Sim
<u>x-amazon-apigateway-documentation Objeto</u>	Não
<u>x-amazon-apigateway-endpoint-Objeto de configuração</u>	Não
<u>x-amazon-apigateway-gatewayObjeto -responses</u>	Sim
<u>x-amazon-apigateway-gateway-Responses.GatewayResponse Objeto</u>	Sim
<u>x-amazon-apigateway-gatewayObjeto -responses.responseParameters</u>	Sim
<u>x-amazon-apigateway-gateway-Objeto ResponseTemplates ResponseTemplates</u>	Sim
<u>x-amazon-apigateway-integration Objeto</u>	Sim
<u>x-amazon-apigateway-integrationObjeto .requestTemplates</u>	Sim
<u>x-amazon-apigateway-integrationObjeto .requestParameters</u>	Não
<u>x-amazon-apigateway-integrationObjeto .responses</u>	Sim
<u>x-amazon-apigateway-integrationObjeto .response</u>	Sim
<u>x-amazon-apigateway-integrationObjeto.responseTemplates</u>	Sim
<u>x-amazon-apigateway-integrationObjeto .responseParameters</u>	Sim

x-amazon-apigateway-requestPropriedade -validator	Não
x-amazon-apigateway-requestObjeto -validators	Não
x-amazon-apigateway-request-Objeto Validators.requestValidator	Não

Funções intrínsecas para AWS SAM

As funções intrínsecas são funções integradas que permitem atribuir valores a propriedades que só estão disponíveis em tempo de execução. AWS SAM tem suporte limitado para certas propriedades de funções intrínsecas, portanto, é incapaz de resolver algumas funções intrínsecas. Consequentemente, recomendamos adicionar a transformação `AWS::LanguageExtensions` para resolver isso. `AWS::LanguageExtensions` É uma macro hospedada por CloudFormation que permite usar funções intrínsecas e outras funcionalidades que, por padrão, não estão incluídas no CloudFormation

Transform:

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

Note

Se você usar funções intrínsecas na `CodeUri` propriedade, não AWS SAM será capaz de analisar corretamente os valores. Em vez disso, considere usar a transformação `AWS::LanguageExtensions`.

Para obter mais informações, consulte a [seção Propriedades de AWS::Serverless::Function](#).

Para obter detalhes sobre como o AWS SAMCLI manipula a `AWS::LanguageExtensions` transformação, incluindo o uso de funções intrínsecas `CodeUri` e outras propriedades empacotáveis, consulte. [CloudFormation suporte a extensões de linguagem](#)

Para obter mais informações sobre funções intrínsecas, consulte o [Referência à função intrínseca](#) no Guia do usuário do AWS CloudFormation .

CloudFormation suporte a extensões de linguagem

O AWS SAMCLI suporta modelos que usam a `AWS::LanguageExtensions` transformação, incluindo `Fn::ForEachFn::Length`, `Fn::ToJsonString`, e `Fn::FindInMap comDefaultValue`. Para obter informações de referência completas sobre essas construções, consulte [AWS::LanguageExtensions transform](#) no Guia do AWS CloudFormation usuário.

Para usar extensões de linguagem em um AWS SAM modelo, liste `AWS::LanguageExtensions` na Transform seção anterior `AWS::Serverless-2016-10-31`:

Transform:

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

Quando é AWS SAMCLI detectado `AWS::LanguageExtensions` na Transform seção de um modelo e você opta pelo processamento local, ele expande as construções de extensão de linguagem localmente antes de executar as transformações. AWS SAM Isso permite que `sam buildsam package,sam deploy,sam sync,sam validate,sam local invoke,sam local start-api`, e `sam local start-lambda` trabalhe com modelos que usam essas construções.

O processamento local está desativado por padrão. Quando desativado, AWS SAMCLI ele passa o modelo inalterado e CloudFormation processa a `AWS::LanguageExtensions` transformação no lado do servidor no momento da implantação. Para métodos de ativação, consulte [Ativando extensões de idioma](#).

A expansão acontece em duas fases quando ativada:

1. Fase 1 (extensões de linguagem) — os `Fn::ForEach` loops são expandidos, as funções intrínsecas são resolvidas sempre que possível e o modelo é convertido em padrão. CloudFormation
2. Fase 2 (AWS SAM Transformação) — O modelo expandido é processado pelo AWS SAM Tradutor como de costume.

O modelo original (com o modelo `Fn::ForEach` intacto) é preservado para CloudFormation implantação, pois CloudFormation processa a `AWS::LanguageExtensions` transformação no lado do servidor.

Ativando extensões de idioma

O processamento local do `AWS::LanguageExtensions` é opcional por comando. Há três métodos de ativação equivalentes, listados em ordem de prioridade:

1. CLI bandeira — `--language-extensions` transmite uma única invocação:

```
sam build --language-extensions
sam package --language-extensions ...
sam deploy --language-extensions ...
```

`--no-language-extensions` desativa explicitamente, substituindo ambas `samconfig.toml` e a variável de ambiente descrita abaixo.

2. **samconfig.toml** — persistir na escolha por projeto:

```
[default.build.parameters]
language_extensions = true

[default.package.parameters]
language_extensions = true

[default.deploy.parameters]
language_extensions = true

[default.sync.parameters]
language_extensions = true

[default.local_invoke.parameters]
language_extensions = true

[default.local_start_api.parameters]
language_extensions = true

[default.local_start_lambda.parameters]
language_extensions = true

[default.validate.parameters]
language_extensions = true
```

Uma `samconfig.toml` entrada é carregada como padrão para o comando, então ela entra em vigor como se o sinalizador tivesse sido passado — e, portanto, supera a variável de ambiente.

3. Variável de ambiente — definida como `SAM_CLI_ENABLE_LANGUAGE_EXTENSIONS=1` habilitada para o shell atual:

```
export SAM_CLI_ENABLE_LANGUAGE_EXTENSIONS=1
sam build
sam local invoke MyFunction
```

Valores verdadeiros (sem distinção entre maiúsculas e minúsculas) são `1`, `e`, `true` `yes`. Qualquer outra coisa, incluindo a string vazia, é tratada como desativada. A variável de ambiente é consultada somente quando nem o CLI sinalizador nem `samconfig.toml` define um valor.

Important

Cada comando precisa de sua própria ativação. Passar `--language-extensions` para `sam build` não se propaga para um posterior `sam local invoke` — o processamento local é decidido por invocação de comando. Use a variável de ambiente ou uma `samconfig.toml` entrada para habilitar vários comandos sem repetir o sinalizador.

Fã:: ForEach

`Fn::ForEach` gera vários recursos, condições ou saídas a partir de uma única definição de modelo:

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

Parameters:
  ServiceNames:
    Type: CommaDelimitedList
    Default: "Users,Orders,Products"

Resources:
  Fn::ForEach::Services:
    - Name
    - !Ref ServiceNames
    - ${Name}Function:
        Type: AWS::Serverless::Function
        Properties:
          Handler: index.handler
```

```
Runtime: python3.12
CodeUri: ./services/${Name}
```

A execução `sam build` expande isso para `UsersFunction`, e `OrdersFunction` `ProductsFunction`, cada um criado a partir de seu respectivo diretório de origem.

Propriedades dinâmicas do artefato

Quando uma propriedade empacotável usa uma variável de loop (por exemplo, `./services/${Name}`), ela AWS SAM CLI gera uma CloudFormation Mappings seção que mapeia cada valor de coleção para seu Amazon S3. URI O `Fn::ForEach` corpo é reescrito para ser usado para `Fn::FindInMap` que CloudFormation possa resolver o artefato correto no momento da implantação.

As propriedades reconhecidas do artefato são as mesmas que são `sam package` reescritas atualmente. Isso inclui:

Propriedades dinâmicas de artefatos reconhecidas pelo AWS SAM CLI

Tipo de atributo	Propriedade
<code>AWS::Serverless::Function</code>	<code>CodeUri</code> , <code>ImageUri</code>
<code>AWS::Serverless::LayerVersion</code>	<code>ContentUri</code>
<code>AWS::Serverless::Api</code>	<code>DefinitionUri</code>
<code>AWS::Serverless::HttpApi</code>	<code>DefinitionUri</code>
<code>AWS::Serverless::StateMachine</code>	<code>DefinitionUri</code>
<code>AWS::Serverless::GraphQLApi</code>	<code>SchemaUri</code> , <code>CodeUri</code>
<code>AWS::Serverless::Application</code>	<code>Location</code>
<code>AWS::Lambda::Function</code>	<code>Code</code> , <code>Code.ImageUri</code>
<code>AWS::Lambda::LayerVersion</code>	<code>Content</code>

Tipo de atributo	Propriedade
<code>AWS::ApiGateway::RestApi</code>	<code>BodyS3Location</code>
<code>AWS::ApiGatewayV2::Api</code>	<code>BodyS3Location</code>
<code>AWS::AppSync::GraphQLSchema</code>	<code>DefinitionS3Location</code>
<code>AWS::AppSync::Resolver</code>	<code>RequestMappingTemplateS3Location</code> , <code>ResponseMappingTemplateS3Location</code> , <code>CodeS3Location</code>
<code>AWS::AppSync::FunctionConfiguration</code>	<code>RequestMappingTemplateS3Location</code> , <code>ResponseMappingTemplateS3Location</code> , <code>CodeS3Location</code>
<code>AWS::StepFunctions::StateMachine</code>	<code>DefinitionS3Location</code>
<code>AWS::ElasticBeanstalk::ApplicationVersion</code>	<code>SourceBundle</code>
<code>AWS::Glue::Job</code>	<code>Command.ScriptLocation</code>
<code>AWS::CloudFormation::Stack</code>	<code>TemplateURL</code>
<code>AWS::CloudFormation::StackSet</code>	<code>TemplateURL</code>
<code>AWS::CloudFormation::ModuleVersion</code>	<code>ModulePackage</code>
<code>AWS::CloudFormation::ResourceVersion</code>	<code>SchemaHandlerPackage</code>

Para propriedades pontilhadas (por exemplo, `Command.ScriptLocation` on `AWS::Glue::Job` ou `Code.ImageUri` on `AWS::Lambda::Function`), o valor é lido e gravado no local aninhado no recurso. O nome de mapeamento gerado usa somente o segmento de folha do caminho da propriedade.

Quando a propriedade é modelada em loop, o nome do mapeamento é SAM<LeafProperty><LoopName> (por exemplo, ou). SAMCodeUriServices
SAMScriptLocationJobs

Important

Customer-authored os mapeamentos não devem começar com esses SAM* prefixos — eles são reservados para o AWS SAMCLI Consulte [Limitações](#).

Por exemplo, depois de `sam package`:

```
Mappings:
  SAMCodeUriServices:
    Users:
      CodeUri: s3://my-bucket/abc123
    Orders:
      CodeUri: s3://my-bucket/def456
    Products:
      CodeUri: s3://my-bucket/ghi789

Resources:
  Fn::ForEach::Services:
    - Name
    - !Ref ServiceNames
    - ${Name}Function:
      Type: AWS::Serverless::Function
      Properties:
        Handler: index.handler
        Runtime: python3.12
        CodeUri: !FindInMap [SAMCodeUriServices, !Ref Name, CodeUri]
```

Vários recursos por ForEach corpo

Um único `Fn::ForEach` corpo pode emitir mais de um recurso por iteração. Cada recurso é gerado para cada valor da coleção:

```
Resources:
  Fn::ForEach::Tables:
    - TableName
    - [Users, Orders, Products]
```

```

- ${TableName}Table:
  Type: AWS::DynamoDB::Table
  Properties:
    TableName: !Sub "${AWS::StackName}-${TableName}"
    # ...

${TableName}StreamProcessor:
  Type: AWS::Serverless::Function
  Properties:
    CodeUri: stream-processors/${TableName}/
    Events:
      DDBStream:
        Type: DynamoDB
        Properties:
          Stream: !GetAtt
            - !Sub "${TableName}Table"
            - StreamArn

```

Resolução de colisão de nomes de mapeamento

Quando dois recursos no mesmo `Fn::ForEach` corpo declaram a mesma propriedade de artefato dinâmico (por exemplo, um `Api` e um `StateMachine` `useDefinitionUri`), eles AWS SAMCLI acrescentam um sufixo retirado da parte estática do modelo de ID lógica do recurso para manter os nomes de mapeamento exclusivos:

Exemplo de mapeamento de nomes com sufixos de colisão

Modelo de recurso	Propriedade	Nome do mapeamento
<code>\${Svc}Api</code>	<code>DefinitionUri</code>	<code>SAMDefinitionUriServicesApi</code>
<code>\${Svc}StateMachine</code>	<code>DefinitionUri</code>	<code>SAMDefinitionUriServicesStateMachine</code>

Quando não há colisão, o nome base (por exemplo, `SAMDefinitionUriServices`) é usado.

Parameter-based coleções

Quando a `Fn::ForEach` coleção é uma referência de parâmetro (por exemplo, `!Ref ServiceNames`) e o corpo do loop usa uma propriedade de artefato dinâmico (por exemplo, `CodeUri: ./services/${Name}`), ele AWS SAMCLI precisa dos valores da coleção

para gerar os SAM* mapeamentos descritos em. [Propriedades dinâmicas do artefato](#) Ele os resolve quando processa o modelo, a partir de:

1. `--parameter-overrides` passado para o AWS SAMCLI comando.
2. O Default valor do parâmetro no modelo.

Important

Como os SAM* mapeamentos são incorporados no momento do pacote, você deve reempacotar sempre que alterar o valor do parâmetro (por exemplo, ao adicionar um novo serviço) para que os mapeamentos incluam entradas para os novos valores. Isso se aplica somente quando o parâmetro aciona um loop dinâmico de artefatos; outras substituições de parâmetros podem ser alteradas no momento da implantação, como de costume.

```
# Package with the values you intend to deploy with
sam package --language-extensions --parameter-overrides
  ServiceNames="Users,Orders,Products"

# Deploy with the same values
sam deploy --language-extensions --parameter-overrides
  ServiceNames="Users,Orders,Products"
```

Pilhas aninhadas

`Fn::ForEach` em modelos de pilha aninhados (`AWS::CloudFormation::Stack`) é suportado. O AWS SAMCLI passa a `Parameters` propriedade da pilha principal para a expansão do modelo filho, para que as `Fn::ForEach` coleções secundárias que fazem referência aos parâmetros fornecidos pelos pais sejam resolvidas corretamente.

```
# parent.yaml
Resources:
  ChildStack:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: ./child.yaml
      Parameters:
        ServiceNames: "Users,Orders,Products"
```

Ventilador aninhado:: ForEach

Até 5 níveis de aninhamento são suportados, correspondendo ao CloudFormation limite:

```
Resources:
  Fn::ForEach::Envs:
    - Env
    - [Dev, Staging, Prod]
    - Fn::ForEach::Services:
        - Svc
        - [Users, Orders]
        - ${Env}${Svc}Function:
            Type: AWS::Serverless::Function
            Properties:
              CodeUri: ./services/${Svc}
              Environment:
                Variables:
                  STAGE: ${Env}
```

ForEach em saídas

`Fn::ForEach` blocos também são expandidos dentro da `Outputs` seção, para que você possa emitir uma saída por valor de coleção:

```
Outputs:
  Fn::ForEach::FunctionArns:
    - Name
    - [alpha, beta]
    - ${Name}FunctionArn:
        Value: !GetAtt
          - !Sub "${Name}Function"
          - Arn
```

Condições e DependsOn

Os recursos emitidos por `Fn::ForEach` can carry `Condition` and `DependsOn` like any other resource. A condição ou dependência é replicada em cada recurso gerado:

```
Conditions:
  IsProd: !Equals [!Ref Environment, prod]

Resources:
```

```

SharedTable:
  Type: AWS::DynamoDB::Table
  # ...

Fn::ForEach::Functions:
  - Name
  - [api, worker]
  - ${Name}Function:
    Type: AWS::Serverless::Function
    Condition: IsProd
    DependsOn: SharedTable
    Properties:
      Handler: main.handler
      CodeUri: functions/${Name}/

```

&sintaxe {identifier}

A &{identifier} sintaxe retira caracteres não alfanuméricos do valor substituído, o que é útil para gerar IDs lógicos válidos a partir de valores como endereços IP:

```

Fn::ForEach::Hosts:
  - IP
  - ["10.0.0.1", "10.0.0.2"]
  - Host&{IP}:
    Type: AWS::EC2::Instance
    # Expands to Host10001, Host10002

```

Funções intrínsecas suportadas

As seguintes funções intrínsecas são resolvidas localmente durante a expansão:

Funções intrínsecas resolvidas localmente durante a expansão das extensões de linguagem

Função	Description
<code>Fn::ForEach</code>	Expansão em loop.
<code>Fn::Length</code>	Retorna a contagem dos elementos da lista.
<code>Fn::ToJsonString</code>	Converte um valor em uma JSON string.
<code>Fn::FindInMap</code>	Pesquisa de mapa, incluindo o opcional <code>DefaultValue</code> .

Função	Description
<code>Fn::If</code>	Seleção de valor condicional.
<code>Fn::Sub</code>	Substituição de strings.
<code>Fn::Join</code>	Concatenação de strings.
<code>Fn::Split</code>	Divisão de cordas.
<code>Fn::Select</code>	Seleção de elementos da lista.
<code>Fn::Base64</code>	Codificação Base64.
<code>Fn::Equals</code> , <code>Fn::And</code> , <code>Fn::Or</code> , <code>Fn::Not</code>	Avaliação da condição.
<code>Ref</code>	Referências de parâmetros e pseudoparâmetros.

As funções que exigem recursos implantados (`Fn::GetAtt`, `Fn::ImportValue`, `Fn::GetAZs`) são preservadas CloudFormation para serem resolvidas no momento da implantação.

Erros de validação

Os seguintes problemas de modelo são detectados localmente antes da execução da AWS SAM transformação:

Erros de validação gerados durante a expansão das extensões de linguagem

Causa	Mensagem de erro
O <code>Fn::ForEach</code> valor está malformado — não é uma lista, não tem exatamente 3 elementos ou tem um identificador de loop que não seja de string.	<code>Fn::ForEach::<key> layout is incorrect (criado como <code>InvalidTemplateException</code>).</code>
Mais de 5 níveis de <code>Fn::ForEach</code> estão aninhados.	<code>Fn::ForEach nesting depth of <N> exceeds the maximum allowed depth of 5. CloudForm</code>

Causa	Mensagem de erro
	ation supports up to 5 nested Fn::ForEach loops.
A coleção é resolvida em uma lista vazia (por exemplo, um <code>CommaDelimitedList</code> parâmetro com <code>Default: ""</code>).	Sem erro — o loop é ignorado silenciosamente e nenhum recurso é emitido.
O <code>!Ref</code> nos pontos de coleta em um parâmetro que não está declarado no modelo.	Sem erro — a referência não resolvida é preservada no modelo. No momento da implantação, CloudFormation resolverá o problema no lado do servidor.

Limitações

- As coleções devem ser resolvidas a qualquer momento `build/package`. `Fn::ForEach` coleções que usam `Fn::GetAtt`, `Fn::ImportValue`, ou `SSM/Secrets` Gerenciam referências dinâmicas não podem ser expandidas localmente. `--parameter-overrides` Em vez disso, use um parâmetro com.
- Os mapeamentos dinâmicos de artefatos são corrigidos no momento do pacote. Quando uma `Fn::ForEach` coleção é uma referência de parâmetro e o corpo do loop usa uma propriedade de artefato dinâmico (por exemplo, `CodeUri: ./services/${Name}`), os SAM* mapeamentos gerados contêm somente entradas para os valores dos parâmetros que foram resolvidos no momento do pacote. Se você alterar `--parameter-overrides` esse parâmetro no momento da implantação sem reempacotar, os novos valores não terão entradas de mapeamento e a implantação falhará. Isso não se aplica aos parâmetros que não são usados para acionar um artefato `Fn::ForEach` dinâmico.
- DeletionPolicy** e **UpdateReplacePolicy** são validados e resolvidos durante a expansão. Eles oferecem suporte `Ref` a parâmetros, mas não a outras funções intrínsecas.
- Limite de nidificação. Até 5 níveis de `Fn::ForEach` podem ser aninhados, correspondendo ao limite do lado do CloudFormation servidor.
- Nomes de mapeamento reservados. Os nomes de mapeamento que começam com qualquer um dos seguintes são reservados para AWS SAMCLI — não crie seus próprios mapeamentos com esses prefixos:

- `SAMCodeUri`, `SAMImageUri`, `SAMContentUri`, `SAMDefinitionUri`, `SAMSchemaUri`, `SAMBodyS3Location`, `SAMDefinitionS3Location`, `SAMTemplateURL`, `SAMCode`, `SAMContent` — emitido por `aws-sam package` para propriedades dinâmicas do artefato. Veja a [Propriedades dinâmicas do artefato](#) tabela.
- `SAMLayers` — emitido por `aws-sam build` quando uma função `Fn::ForEach` gerada pega referências de camada de dependência geradas automaticamente (o Lambda coloca as compilações em uma pilha aninhada). AWS SAM CLI Esse prefixo não tem nenhuma propriedade correspondente de autoria do usuário; ele é adicionado automaticamente.

Telemetria

O AWS SAM CLI emite um evento de *CFNLanguageExtensions* telemetria quando um comando é invocado com `--language-extensions` (ou seu equivalente em variável de ambiente) e o modelo declara a transformação. `AWS::LanguageExtensions` O evento é acionado uma vez por invocação e nenhum conteúdo do modelo é transmitido. Quando o processamento local está desativado (o padrão), nenhum evento é acionado.

Exemplo de aplicativos sem servidor para AWS SAM

Esta seção inclui dois exemplos de aplicações: uma que processa eventos do DynamoDB e outra que processa eventos do Amazon S3. Cada exemplo conduz você por um step-by-step processo de criação de um aplicativo. Além disso, ambos incluem detalhes sobre a configuração de origens de eventos e recursos da AWS. Ambos começam pela identificação do que deve ser feito antes de você começar e seguem as etapas para inicializar, testar, empacotar e implantar a aplicação.

Tópicos

- [Processe eventos do DynamoDB com AWS SAM](#)
- [Processe eventos do Amazon S3 com AWS SAM](#)

Processe eventos do DynamoDB com AWS SAM

Com esse aplicativo de exemplo, você se baseia no que aprendeu na visão geral e no guia de início rápido e instala outro aplicativo de exemplo. Esse aplicativo consiste em uma função de evento Lambda que é invocada por uma fonte de eventos de tabela do DynamoDB. A função do Lambda é muito simples: ela registra dados que foram transmitidos pela mensagem de origem do evento.

Este exercício mostra como imitar mensagens de origem de eventos que são passadas para as funções do Lambda quando são invocadas.

Antes de começar

Verifique se você concluiu a configuração necessária no [Instale o AWS SAM CLI](#).

Etapa 1: Inicializar o aplicativo

Nesta seção, você baixa o pacote do aplicativo, que consiste em um AWS SAM modelo e código do aplicativo.

Como inicializar o aplicativo

1. Execute o seguinte comando em um prompt de comando do AWS SAM CLI.

```
sam init \
```

```
--location gh:aws-samples/cookiecutter-aws-sam-dynamodb-python \  
--no-input
```

Observe que `gh:` no comando acima é expandido para o GitHub url `https://github.com/`.

2. Revise o conteúdo do diretório criado pelo comando (`dynamodb_event_reader/`):
 - `template.yaml`— Define dois AWS recursos que o aplicativo Read DynamoDB precisa: uma função Lambda e uma tabela do DynamoDB. O modelo também define o mapeamento entre os dois recursos.
 - `read_dynamodb_event/` — diretório – contém o código do aplicativo DynamoDB.

Etapa 2: Testar o aplicativo localmente

Para testes locais, use o AWS SAMCLI para gerar um evento de amostra do DynamoDB e invocar a função do Lambda:

```
sam local generate-event dynamodb update | sam local invoke --event - ReadDynamoDBEvent
```

O `generate-event` comando cria uma mensagem de origem do evento de teste, como as mensagens que são criadas quando todos os componentes são implantados na AWS nuvem. Essa mensagem de origem do evento é canalizada para a função `ReadDynamoDBEvent` Lambda.

Verifique se as mensagens esperadas foram impressas no console, com base no código-fonte em `app.py`.

Etapa 3: Empacotar o aplicativo

Depois de testar seu aplicativo localmente, você usa o AWS SAMCLI para criar um pacote de implantação, que você usa para implantar o aplicativo AWS na nuvem.

Para criar o pacote de implantação do Lambda

1. Crie um bucket do S3 no local onde deseja salvar o código empacotado. Se você quiser usar um bucket do S3 existente, ignore esta etapa.

```
aws s3 mb s3://bucketname
```

2. Crie o pacote de implantação ao executar o seguinte comando `package` da CLI do prompt do comando.

```
sam package \  
  --template-file template.yaml \  
  --output-template-file packaged.yaml \  
  --s3-bucket bucketname
```

Você especifica o novo arquivo de modelo `packaged.yaml`, ao implantar o aplicativo na próxima etapa.

Etapa 4: Implantar um aplicativo

Agora que você criou o pacote de implantação, use-o para implantar o aplicativo na AWS nuvem. Em seguida, você testa o aplicativo.

Para implantar o aplicativo sem servidor no AWS Nuvem

- No AWS SAMCLI, use o comando `deploy CLI` para implantar todos os recursos que você definiu no modelo.

```
sam deploy \  
  --template-file packaged.yaml \  
  --stack-name sam-app \  
  --capabilities CAPABILITY_IAM \  
  --region us-east-1
```

No comando, o parâmetro `--capabilities` permite que o AWS CloudFormation crie um perfil do IAM.

CloudFormation cria os AWS recursos definidos no modelo. Você pode acessar os nomes desses recursos no CloudFormation console.

Para testar o aplicativo sem servidor no AWS Nuvem

1. Abra o console do DynamoDB.
2. Insira um registro na tabela que você acabou de criar.
3. Vá até a guia Métricas da tabela e escolha Exibir todas as CloudWatch métricas. No CloudWatch console, escolha Logs para poder visualizar a saída do log.

Próximas etapas

Para continuar aprendendo sobre isso AWS SAM, consulte os seguintes recursos:

- [Serverless Land](#) — Site que reúne padrões sem servidor, exemplos de código e recursos de aprendizado específicos para aplicativos SAM. AWS

Processe eventos do Amazon S3 com AWS SAM

Com esse aplicativo de exemplo, você se baseia no que aprendeu nos exemplos anteriores e instala um aplicativo mais complexo. Esse aplicativo consiste em uma função de evento Lambda que é invocada por uma fonte de eventos de upload de objetos do Amazon S3. Este exercício mostra como acessar AWS recursos e fazer chamadas de AWS serviço por meio de uma função Lambda.

Esse exemplo de aplicativo com tecnologia sem servidor processa eventos de criação de objetos no Amazon S3. Para cada imagem que é carregada em um bucket, o Amazon S3 detecta o evento criado pelo objeto e invoca uma função do Lambda. A função do Lambda invoca o Amazon Rekognition para detectar o texto que está na imagem. Em seguida, ele armazena os resultados retornados pelo Amazon Rekognition em uma tabela do DynamoDB.

Note

Com este aplicativo de exemplo, você executa as etapas em uma ordem ligeiramente diferente da dos exemplos anteriores. A razão para isso é que esse exemplo exige que AWS os recursos sejam criados e as permissões do IAM sejam configuradas antes que você possa testar a função Lambda localmente. Vamos aproveitar CloudFormation para criar os recursos e configurar as permissões para você. Caso contrário, você precisaria fazer isso manualmente antes de testar a função do Lambda localmente.

Como esse exemplo é mais complicado, certifique-se de estar familiarizado com a instalação dos aplicativos de exemplo anteriores antes de executar este.

Antes de começar

Verifique se você concluiu a configuração necessária no [Instale o AWS SAM CLI](#).

Etapa 1: Inicializar o aplicativo

Nesta seção, você baixa o aplicativo de amostra, que consiste em um AWS SAM modelo e código do aplicativo.

Como inicializar o aplicativo

1. Execute o seguinte comando em um prompt de comando do AWS SAM CLI.

```
sam init \  
--location https://github.com/aws-samples/cookiecutter-aws-sam-s3-rekognition-  
dynamodb-python \  
--no-input
```

2. Revise o conteúdo do diretório criado pelo comando (`aws_sam_ocr/`):
 - `template.yaml`— Define três AWS recursos que o aplicativo Amazon S3 precisa: uma função Lambda, um bucket do Amazon S3 e uma tabela do DynamoDB. O modelo também define os mapeamentos e as permissões entre esses recursos.
 - `src/` — diretório Contém o código do aplicativo Amazon S3.
 - `SampleEvent.json`— A origem do evento de amostra, que é usada para testes locais.

Etapa 2: Empacotar o aplicativo

Antes de testar esse aplicativo localmente, você deve usar o AWS SAMCLI para criar um pacote de implantação, que você usa para implantar o aplicativo AWS na nuvem. Essa implantação cria os AWS recursos e as permissões necessários para testar o aplicativo localmente.

Para criar o pacote de implantação do Lambda

1. Crie um bucket do S3 no local onde deseja salvar o código empacotado. Se você quiser usar um bucket do S3 existente, ignore esta etapa.

```
aws s3 mb s3://bucketname
```

2. Crie o pacote de implantação ao executar o seguinte comando `package` da CLI do prompt do comando.

```
sam package \  

```

```
--template-file template.yaml \  
--output-template-file packaged.yaml \  
--s3-bucket bucketname
```

Você especifica o novo arquivo de modelo `packaged.yaml`, ao implantar o aplicativo na próxima etapa.

Etapa 3: Implantar um aplicativo

Agora que você criou o pacote de implantação, use-o para implantar o aplicativo na AWS nuvem. Em seguida, você testa o aplicativo invocando-o na AWS nuvem.

Para implantar o aplicativo sem servidor no AWS Nuvem

- No AWS SAMCLI, use o `deploy` comando para implantar todos os recursos que você definiu no modelo.

```
sam deploy \  
  --template-file packaged.yaml \  
  --stack-name aws-sam-ocr \  
  --capabilities CAPABILITY_IAM \  
  --region us-east-1
```

No comando, o `--capabilities` parâmetro permite CloudFormation criar uma função do IAM.

CloudFormation cria os AWS recursos definidos no modelo. Você pode acessar os nomes desses recursos no CloudFormation console.

Para testar o aplicativo sem servidor no AWS Nuvem

1. Carregue-o no bucket do Amazon S3 que você criou para esse aplicativo de amostra.
2. Abra o console do DynamoDB e encontre a tabela que foi criada. Consulte a tabela para ver os resultados retornados pelo Amazon Rekognition.
3. Verifique se a tabela do DynamoDB contém novos registros que contêm texto que o Amazon Rekognition encontrou na imagem carregada.

Etapa 4: Testar o aplicativo localmente

Antes de testar o aplicativo localmente, você deve primeiro recuperar os nomes dos AWS recursos que foram criados pelo CloudFormation.

- Recupere o nome da chave do Amazon S3 e o nome do bucket de. CloudFormation Modifique o arquivo `SampleEvent.json` substituindo os valores da chave do objeto, do nome do bucket e do ARN do bucket.
- Recupere o nome da tabela do DynamoDB. Esse nome é usado para o comando `sam local invoke` a seguir.

Use o AWS SAMCLI para gerar um exemplo de evento do Amazon S3 e invocar a função do Lambda:

```
TABLE_NAME=Table name obtained from CloudFormation console sam local invoke --event  
SampleEvent.json
```

A parte do `TABLE_NAME=` define o nome da tabela do DynamoDB. O parâmetro `--event` especifica o arquivo que contém a mensagem do evento de teste a ser transmitida para a função do Lambda.

Agora você pode verificar se os registros esperados do DynamoDB foram criados com base nos resultados retornados pelo Amazon Rekognition.

Próximas etapas

Para continuar aprendendo sobre isso AWS SAM, consulte os seguintes recursos:

- [Serverless Land](#) — Site que reúne padrões sem servidor, exemplos de código e recursos de aprendizado específicos para aplicativos SAM. AWS

Desenvolva seu aplicativo sem servidor com AWS SAM

Esta seção contém tópicos sobre como validar seu AWS SAM modelo e criar seu aplicativo com dependências. Ele também contém tópicos sobre o uso AWS SAM em determinados casos de uso, como trabalhar com camadas Lambda, usar aplicativos aninhados, controlar o acesso ao API Gateway APIs, orquestrar recursos com AWS Step Functions e assinar códigos em seus aplicativos. Os três principais marcos de que você precisa concluir para desenvolver a aplicação estão listados abaixo.

Tópicos

- [Crie seu aplicativo em AWS SAM](#)
- [Defina sua infraestrutura com AWS SAM](#)
- [Crie seu aplicativo com AWS SAM](#)

Crie seu aplicativo em AWS SAM

Depois de concluir os [Conceitos básicos](#) e ler [Como usar AWS Serverless Application Model \(AWS SAM\)](#), tudo estará pronto para criar um projeto do AWS SAM no seu ambiente de desenvolvedor. Seu AWS SAM projeto servirá como ponto de partida para escrever seu aplicativo sem servidor. Para obter uma lista de opções de AWS SAMCLI `sam init` comando, consulte [sam init](#).

O AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam init` fornece opções para inicializar um novo aplicativo sem servidor que consiste em:

- Um AWS SAM modelo para definir seu código de infraestrutura.
- Uma estrutura de pastas que organiza seu aplicativo.
- Configuração para suas funções de AWS Lambda eventos.

Para criar um AWS SAM projeto, consulte os tópicos desta seção.

Note

`sam init` inclui modelos de projeto para funções duráveis em tempos de execução compatíveis (PythonTypeScript, Java). Esses modelos fornecem código inicial e configuração para criar aplicativos sem servidor com estado.

Tópicos

- [Inicialize um novo aplicativo com tecnologia sem servidor.](#)
- [Opções para o sam init](#)
- [Solução de problemas](#)
- [Exemplos](#)
- [Saiba mais](#)
- [Próximas etapas](#)

Inicialize um novo aplicativo com tecnologia sem servidor.

Para inicializar um novo aplicativo com tecnologia sem servidor usando o AWS SAM CLI

1. `cd` para um diretório inicial.
2. Execute o seguinte na linha de comando:

```
$ sam init
```

3. O AWS SAMCLI o guiará por um fluxo interativo para criar um novo aplicativo com tecnologia sem servidor.

Note

Conforme detalhado em [Tutorial: implante um aplicativo Hello World com AWS SAM](#), esse comando inicializa a aplicação sem servidor, criando o diretório do projeto. Esse diretório conterá vários arquivos e pastas. O arquivo mais importante é `template.yaml`. Esse é o seu AWS SAM modelo. A versão do python deve corresponder à versão do python listada no arquivo `template.yaml` criado pelo comando `sam init`.

Escolha um modelo inicial

Um modelo consiste no seguinte:

1. Um AWS SAM modelo para seu código de infraestrutura.

2. Um diretório inicial do projeto que organiza seus arquivos de projeto. Por exemplo, isso pode incluir:
 - a. Uma estrutura para o código de função do Lambda e suas dependências.
 - b. Uma pasta `events` que contém eventos de teste para testes locais.
 - c. Uma pasta `tests` para oferecer suporte ao teste de unidade.
 - d. Um arquivo `samconfig.toml` para definir as configurações do projeto.
 - e. Um arquivo `ReadMe` e outros arquivos básicos do projeto inicial.

Veja a seguir um exemplo de um diretório de projeto inicial:

```
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
```

Você pode selecionar em uma lista de AWS modelos de início rápido disponíveis ou fornecer sua própria localização de modelo personalizado.

Para escolher um AWS Modelo de início rápido

1. Quando solicitado, selecione AWS Modelos de início rápido.
2. Selecione um modelo de Início AWS Rápido para começar. Este é um exemplo:

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**

Para escolher sua própria localização de modelo personalizado

1. Quando solicitado, selecione a localização do modelo personalizado.

Which template source would you like to use?

- 1 - AWS Quick Start Templates
- 2 - Custom Template Location

Choice: **2**

2. O AWS SAMCLI solicitará que você forneça uma localização do modelo.

Template location (git, mercurial, http(s), zip, path):

Forneça qualquer uma das seguintes localizações para o arquivo de arquivos.zip do modelo:

- GitHubrepositório — O caminho para o arquivo.zip no seu repositório GitHub. O arquivo deve estar na raiz do seu repositório.

- Mercurialrepositório — O caminho para o arquivo.zip no seu repositório Mercurial. O arquivo deve estar na raiz do seu repositório.
 - caminho.zip — Um caminho HTTPS ou localização para seu arquivo.zip.
3. O AWS SAMCLI inicializará seu aplicativo com tecnologia sem servidor usando seu modelo personalizado.

Escolher um tempo de execução

Quando você escolhe um AWS Modelo de início rápido, o AWS SAMCLI solicitará que você selecione um tempo de execução para suas funções do Lambda. A lista de opções exibida pelo AWS SAMCLI são os tempos de execução suportados nativamente pelo Lambda.

- O [tempo de execução](#) fornece um ambiente específico de linguagem que é executado no ambiente de execução.
- [Quando implantado no Nuvem AWS, o serviço Lambda invoca sua função em um ambiente de execução.](#)

Você pode usar qualquer outra linguagem de programação com um tempo de execução personalizado. Para fazer isso, você precisa criar manualmente a estrutura inicial do aplicativo. Em seguida, você pode usar o `sam init` para inicializar rapidamente seu aplicativo configurando uma localização de modelo personalizado.

A partir da sua seleção, o AWS SAMCLI cria o diretório inicial para o código e as dependências da função do Lambda.

Se o Lambda oferecer suporte a vários gerenciadores de dependências para seu tempo de execução, você será solicitado a escolher seu gerenciador de dependências preferido.

Escolha um tipo de pacote

Quando você escolhe um AWS modelo de início rápido e um tempo de execução, o AWS SAMCLI solicita que você selecione um tipo de pacote. O tipo de pacote determina como suas funções do Lambda são implantadas para uso com o serviço Lambda. Os dois tipos de pacotes compatíveis são:

1. Imagem de contêiner – Contém o sistema operacional de base, o tempo de execução, as extensões do Lambda, o código do seu aplicativo e suas dependências.
2. arquivo .zip – Contém o código do seu aplicativo e suas dependências.

Para saber mais sobre os tipos de pacotes de implantação, consulte [Pacotes de implantação do Lambda](#) no AWS Lambda Guia do desenvolvedor.

Veja a seguir um exemplo de estrutura de diretórios de um aplicativo com uma função do Lambda empacotada como uma imagem de contêiner. O AWS SAMCLI baixa a imagem e cria um Dockerfile no diretório da função para especificar a imagem.

```
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
```

A seguir está um exemplo de estrutura do diretório de um aplicativo com uma função empacotada como um arquivo .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
```

Configurar AWS X-Ray rastreamento

Você pode optar por ativar o AWS X-Ray rastreamento. Para saber mais, consulte [O que é AWS X-Ray?](#) no Guia do AWS X-Ray desenvolvedor.

Se você ativar, ele AWS SAMCLI configura seu AWS SAM modelo. Este é um exemplo:

```
Globals:
  Function:
    ...
    Tracing: Active
  Api:
    TracingEnabled: True
```

Configure o monitoramento com o Amazon CloudWatch Application Insights

Você pode optar por ativar o monitoramento usando o Amazon CloudWatch Application Insights. Para saber mais, consulte [Amazon CloudWatch Application Insights](#) no Guia CloudWatch do usuário da Amazon.

Se você ativar, ele AWS SAMCLI configura seu AWS SAM modelo. Este é um exemplo:

```
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
```

Dê um nome para o seu aplicativo

Forneça um nome para seu aplicativo. O AWS SAMCLI cria uma pasta de nível superior para seu aplicativo usando esse nome.

Opções para o sam init

Veja a seguir algumas das opções principais que você pode usar com o comando `sam init`. Para obter uma lista de todas as opções, consulte [sam init](#).

Inicializar um aplicativo usando um local de modelo personalizado

Use a opção `--location` e forneça um local de modelo personalizado compatível. Veja um exemplo a seguir.

```
$ sam init --location https://github.com/aws-samples/sessions-with-aws-sam/raw/master/
starter-templates/web-app.zip
```

Inicializar um aplicativo sem o fluxo interativo

Use a opção `--no-interactive` e forneça suas opções de configuração na linha de comando para ignorar o fluxo interativo. Veja um exemplo a seguir.

```
$ sam init --no-interactive --runtime go1.x --name go-demo --dependency-manager mod --
app-template hello-world
```

Solução de problemas

Para solucionar o problema do AWS SAMCLI, consulte [AWS SAM CLI solução de problemas](#).

Exemplos

Inicialize um novo aplicativo sem servidor usando o Hello World AWS Modelo inicial

Para este exemplo, consulte [Etapa 1: Inicialize o aplicativo de amostra Hello World](#) no Tutorial: Implantação de um aplicativo Hello World.

Inicialize um novo aplicativo com tecnologia sem servidor com um local de modelo personalizado

Veja a seguir exemplos de como fornecer uma localização GitHub para seu modelo personalizado:

```
$ 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
```

Veja a seguir um exemplo de caminho de arquivo local:

```
$ sam init --location /path/to/template.zip
```

Veja a seguir um exemplo de um caminho acessível por HTTPS:

```
$ sam init --location https://github.com/aws-samples/sessions-with-aws-sam/raw/master/starter-templates/web-app.zip
```

Saiba mais

Para obter mais informações sobre como usar o comando `sam init`, consulte:

- [Aprendizagem AWS SAM: sam init](#) — série Serverless Land “Aprendizagem AWS SAM” sobre YouTube
- [Estruturação de aplicativos com tecnologia sem servidor para uso com o AWS SAMCLI \(Sessões com SAM S2E7\)](#) — Sessões com séries AWS SAM no YouTube.

Próximas etapas

Agora que você criou seu AWS SAM projeto, você está pronto para começar a criar seu aplicativo. Consulte [Defina sua infraestrutura com AWS SAM](#) para obter instruções detalhadas sobre as tarefas que você precisa concluir para fazer isso.

Defina sua infraestrutura com AWS SAM

Agora que você criou seu projeto, você está pronto para definir sua infraestrutura de aplicativos com AWS SAM. Faça isso configurando seu AWS SAM modelo para definir os recursos e propriedades do seu aplicativo, que é o `template.yaml` arquivo em seu AWS SAM projeto.

Os tópicos desta seção fornecem conteúdo sobre como definir sua infraestrutura em seu AWS SAM modelo (seu `template.yaml` arquivo). Ele também contém tópicos sobre a definição de recursos para casos de uso específicos, como trabalhar com camadas Lambda, usar aplicativos aninhados, controlar o acesso ao API Gateway APIs, orquestrar recursos com AWS Step Functions, assinar códigos em seus aplicativos e validar seu modelo. AWS SAM

Tópicos

- [Defina os recursos do aplicativo em seu AWS SAM modelo](#)
- [Configure e gerencie o acesso a recursos em seu AWS SAM modelo](#)
- [Controle o acesso à API com seu AWS SAM modelo](#)
- [Aumente a eficiência usando camadas Lambda com AWS SAM](#)
- [Reutilize código e recursos usando aplicativos aninhados no AWS SAM](#)
- [Gerencie eventos baseados em tempo com o EventBridge Scheduler em AWS SAM](#)
- [Orquestrando AWS SAM recursos com AWS Step Functions](#)
- [Configure a assinatura de código para seu AWS SAM aplicação](#)
- [Validar arquivos AWS SAM de modelo](#)

Defina os recursos do aplicativo em seu AWS SAM modelo

Você define os AWS recursos que seu aplicativo sem servidor usa na `Resources` seção do seu AWS SAM modelo. Ao definir um recurso, você identifica o que é o recurso, como ele interage com outros recursos e como ele pode ser acessado (ou seja, as permissões do recurso).

A **Resources** seção do seu AWS SAM modelo pode conter uma combinação de CloudFormation recursos e AWS SAM recursos. Além disso, você pode usar AWS SAM a sintaxe abreviada para os seguintes recursos:

AWS SAM sintaxe abreviada	O que ele faz com um AWS recurso relaciona do
AWS::Serverless: :API	Cria uma coleção de recursos e métodos do API Gateway que podem ser invocados por meio de endpoints HTTPS.
AWS::Serverless::Application	Incorpora um aplicativo com tecnologia sem servidor de AWS Serverless Application Repository ou de um bucket do Amazon S3 como um aplicativo aninhado.
AWS::Serverless: :Conector	Configura as permissões entre dois recursos. Para obter uma introdução aos conectores, consulte Gerenciando permissões de recursos com AWS SAM conectores .
AWS::Serverless: :Função	Cria uma função de AWS Lambda evento, uma função de execução AWS Identity and Access Management (IAM) e mapeamentos de origem de eventos que acionam a função.
AWS::Serverless: :API GraphQL	cria e configura uma AWS AppSync GraphQL API para seu aplicativo sem servidor.
AWS::Serverless::HttpApi	Cria uma API HTTP do Amazon API Gateway, que permite criar APIs RESTful com menor latência e custos mais baixos do que APIs REST.
AWS::Serverless::LayerVersion	Cria um Lambda LayerVersion que contém a biblioteca ou o código de tempo de execução necessário para uma função Lambda.

AWS SAM sintaxe abreviada	O que ele faz com um AWS recurso relaciona do
AWS::Serverless::SimpleTable	Cria uma tabela do DynamoDB com uma chave primária de atributo único.
AWS::Serverless::StateMachine	Cria uma máquina de AWS Step Functions estado, que você pode usar para orquestrar AWS Lambda funções e outros AWS recursos para formar fluxos de trabalho complexos e robustos.

Os recursos acima também estão listados em [AWS SAM recursos e propriedades](#).

Para obter informações de referência sobre todos os tipos de AWS recursos e propriedades CloudFormation e AWS SAM suporte, consulte a [referência de tipos de AWS recursos e propriedades](#) no Guia AWS CloudFormation do usuário.

Configure e gerencie o acesso a recursos em seu AWS SAM modelo

Para que seus AWS recursos interajam entre si, o acesso e as permissões adequados devem ser configurados entre seus recursos. Fazer isso requer a configuração de usuários, funções e políticas AWS Identity and Access Management (IAM) para realizar sua interação de maneira segura.

Os tópicos desta seção estão todos relacionados à configuração do acesso aos recursos definidos no modelo. Esta seção começa com as melhores práticas gerais. Os próximos dois tópicos analisam duas opções que você tem para configurar o acesso e as permissões entre os recursos referenciados em seu aplicativo sem servidor: AWS SAM conectores e modelos de política. AWS SAM O último tópico fornece detalhes para gerenciar o acesso do usuário usando a mesma mecânica CloudFormation usada para gerenciar usuários.

Para saber mais, consulte [Controlar o acesso com AWS Identity and Access Management](#) no AWS CloudFormation Guia do usuário.

O AWS Serverless Application Model (AWS SAM) fornece duas opções que simplificam o gerenciamento de acesso e permissões para seus aplicativos sem servidor.

1. AWS SAM conectores

2. Modelos de políticas AWS SAM

AWS SAM conectores

Os conectores são uma forma de fornecer permissões entre dois recursos. Você faz isso descrevendo como eles devem interagir uns com os outros em seu AWS SAM modelo. Eles podem ser definidos usando o atributo do recurso `Connectors` ou o tipo de recurso `AWS::Serverless::Connector`. Os conectores oferecem suporte ao provisionamento `Read` e `Write` ao acesso de dados e eventos entre uma combinação de recursos. AWS Para saber mais sobre AWS SAM conectores, consulte [Gerenciando permissões de recursos com AWS SAM conectores](#).

AWS SAM Modelos de políticas

AWS SAM os modelos de política são conjuntos predefinidos de permissões que você pode adicionar aos seus AWS SAM modelos para gerenciar o acesso e as permissões entre suas funções de AWS Lambda eventos, as máquinas de AWS Step Functions estado e os recursos com os quais elas interagem. Para saber mais sobre modelos AWS SAM de política, consulte [AWS SAM Modelos de políticas](#).

AWS CloudFormation mecanismos

CloudFormation os mecanismos incluem a configuração de usuários, funções e políticas do IAM para gerenciar permissões entre seus AWS recursos. Para saber mais, consulte [Gerenciando AWS SAM permissões com CloudFormation mecanismos](#).

Práticas recomendadas

Em todos os seus aplicativos com tecnologia sem servidor, você pode usar vários métodos para configurar permissões entre seus recursos. Portanto, você pode selecionar a melhor opção para cada cenário e usar várias opções juntas em todos os seus aplicativos. Aqui estão alguns fatores a considerar ao escolher a melhor opção para você:

- AWS SAM Tanto os conectores quanto os modelos de políticas reduzem a experiência em IAM necessária para facilitar as interações seguras entre seus AWS recursos. Use conectores e modelos de políticas quando houver suporte.
- AWS SAM os conectores fornecem uma sintaxe abreviada simples e intuitiva para definir permissões em seus AWS SAM modelos e exigem o mínimo de experiência em IAM. Quando houver suporte para AWS SAM conectores e modelos de política, use AWS SAM conectores.

- AWS SAM os conectores podem provisionar Read e Write acessar dados e eventos entre os recursos de AWS SAM origem e destino suportados. Para obter uma lista de recursos suportados, consulte [AWS SAM referência do conector](#). Quando suportado, use AWS SAM conectores.
- Embora os modelos AWS SAM de política estejam limitados às permissões entre suas funções do Lambda, as máquinas de estado do Step Functions e os AWS recursos com os quais elas interagem, os modelos de política oferecem suporte a todas as operações CRUD. Quando houver suporte e quando um modelo AWS SAM de política para seu cenário estiver disponível, use modelos AWS SAM de política. Para obter uma lista de modelos de política disponíveis, consulte [AWS SAM Modelos de políticas](#).
- Para todos os outros cenários, ou quando a granularidade for necessária, use CloudFormation mecanismos.

Gerenciando permissões de recursos com AWS SAM conectores

Os conectores são um tipo de recurso AWS Serverless Application Model (AWS SAM) abstrato, identificado como `AWS::Serverless::Connector`, que fornece permissões simples e bem definidas entre seus recursos de aplicativos sem servidor.

Benefícios do AWS SAM conectores

Ao compor automaticamente as políticas de acesso apropriadas entre os recursos, os conectores permitem que você crie seus aplicativos sem servidor e se concentre na arquitetura do aplicativo sem precisar de experiência em recursos de AWS autorização, linguagem de políticas e configurações de segurança específicas do serviço. Portanto, os conectores são um grande benefício para desenvolvedores que podem ser novos no desenvolvimento de tecnologia sem servidor ou desenvolvedores experientes que desejam aumentar sua velocidade de desenvolvimento.

Utilizar AWS SAM conectores

Use o atributo de recurso `Connectors` incorporando-o em um recurso de origem. Em seguida, defina seu recurso de destino e descreva como os dados ou eventos devem fluir entre esses recursos. AWS SAM em seguida, compõe as políticas de acesso necessárias para facilitar as interações necessárias.

A seguir é descrito como esse atributo de recurso é gravado:

```
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>
    ...
```

Como funcionam os conectores

Note

Esta seção explica como os conectores provisionam os recursos necessários nos bastidores. Isso acontece automaticamente ao usar conectores.

Primeiro, o atributo de recurso Connectors incorporado é transformado em um tipo de recurso AWS::Serverless::Connector. Seu ID lógico é criado automaticamente como *<source-resource-logical-id><embedded-connector-logical-id>*.

Por exemplo, aqui está um conector incorporado:

```
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
```

Isso gerará o seguinte recurso `AWS::Serverless::Connector`:

```
Transform: AWS::Serverless-2016-10-31
Resources:
  ...
  MyFunctionMyConn:
    Type: AWS::Serverless::Connector
    Properties:
      Source:
        Id: MyFunction
      Destination:
        Id: MyTable
      Permissions:
        - Read
        - Write
```

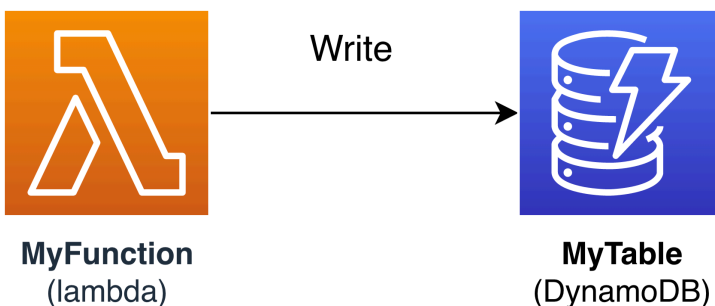
Note

Você também pode definir conectores em seu AWS SAM modelo usando essa sintaxe. Isso é recomendado quando seu recurso de origem é definido em um modelo separado do seu conector.

Em seguida, as políticas de acesso necessárias para essa conexão são compostas automaticamente. Para obter mais informações sobre os recursos gerados por conectores, consulte [CloudFormation recursos gerados quando você especifica AWS::Serverless::Connector](#).

Exemplo de conectores

O exemplo a seguir mostra como você pode usar conectores para gravar dados de uma função de AWS Lambda evento em uma tabela do 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
```

O recurso do atributo `Connectors` está incorporado ao recurso de origem da função do Lambda. A tabela do DynamoDB é definida como o recurso de destino usando a propriedade `Id`. Os conectores provisionarão permissões `Write` entre esses dois recursos.

Quando você implanta seu AWS SAM modelo no CloudFormation, AWS SAM ele compõe automaticamente as políticas de acesso necessárias para que essa conexão funcione.

Conexões suportadas entre recursos de origem e destino

Suporte a conectores `Read` e tipos de permissão de dados e eventos `Write` entre uma combinação selecionada de conexões de recursos de origem e destino. Por exemplo, os conectores oferecem

suporte a uma conexão `Write` entre um recurso de origem `AWS::ApiGateway::RestApi` e um recurso de destino `AWS::Lambda::Function`.

Os recursos de origem e destino podem ser definidos usando uma combinação de propriedades compatíveis. Os requisitos de propriedade dependerão da conexão que você está fazendo e de onde os recursos estão definidos.

Note

Os conectores podem provisionar permissões entre os tipos de recursos com tecnologia sem servidor e sem tecnologia sem servidor compatíveis.

Para obter uma lista de conexões de recursos compatíveis e seus requisitos de propriedade, consulte [Tipos de recursos de origem e destino suportados para conectores](#).

Defina as permissões de leitura e gravação em AWS SAM

Em AWS SAM, `Read` e `Write` as permissões podem ser provisionadas em um único conector:

```
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
```

Para obter mais informações sobre o uso de conectores, consulte [AWS SAM referência do conector](#).

Defina recursos usando outras propriedades suportadas no AWS SAM

Para recursos de origem e destino, quando definidos no mesmo modelo, use a propriedade `Id`. Opcionalmente, um `Qualifier` pode ser adicionado para restringir o escopo do seu recurso definido. Quando o recurso não estiver no mesmo modelo, use uma combinação de propriedades compatíveis.

- Para obter uma lista das combinações de propriedades suportadas para recursos de origem e destino, consulte [Tipos de recursos de origem e destino suportados para conectores](#).
- Para obter uma descrição das propriedades que você pode usar com conectores, consulte [AWS::Serverless::Conector](#).

Ao definir um recurso de origem com uma propriedade diferente de `Id`, use a propriedade `SourceReference`.

```
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>
```

Aqui está um exemplo, usando `Qualifier` para restringir o escopo de um recurso do 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
  ...
```

Aqui está um exemplo, usando uma combinação compatível de Arn e Type para definir um recurso de destino a partir de outro modelo:

```
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
  ...
```

Para obter mais informações sobre o uso de conectores, consulte [AWS SAM referência do conector](#).

Crie vários conectores de uma única fonte no AWS SAM

Em um recurso de origem, você pode definir vários conectores, cada um com um recurso de destino diferente.

```
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
...
```

Para obter mais informações sobre o uso de conectores, consulte [AWS SAM referência do conector](#).

Crie conectores de vários destinos em AWS SAM

Dentro de um recurso de origem, você pode definir um único conector com vários recursos de destino. Aqui está um exemplo de um recurso de origem de função do Lambda conectado a um bucket do Amazon Simple Storage Service (Amazon S3) e a uma tabela do 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
```

Para obter mais informações sobre o uso de conectores, consulte [AWS SAM referência do conector](#).

Defina atributos de recursos com conectores em AWS SAM

Os atributos dos recursos podem ser definidos para que os recursos especifiquem comportamentos e relacionamentos adicionais. Para saber mais sobre os atributos de recursos, consulte [Referência de atributos de recurso](#) no AWS CloudFormation Guia do usuário.

Você pode adicionar atributos de recursos ao seu conector incorporado definindo-os no mesmo nível das propriedades do conector. Quando seu AWS SAM modelo for transformado na implantação, os atributos passarão para os recursos gerados.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Connectors:
      MyConn:
        DeletionPolicy: Retain
        DependsOn: AnotherFunction
        Properties:
          ...
```

Para obter mais informações sobre o uso de conectores, consulte [AWS SAM referência do conector](#).

Saiba mais

Para obter mais informações sobre o uso de AWS SAM conectores, consulte os tópicos a seguir:

- [AWS::Serverless: :Conector](#)
- [Defina as permissões de leitura e gravação em AWS SAM](#)
- [Defina recursos usando outras propriedades suportadas no AWS SAM](#)
- [Crie vários conectores de uma única fonte no AWS SAM](#)
- [Crie conectores de vários destinos em AWS SAM](#)
- [Defina as permissões de leitura e gravação em AWS SAM](#)
- [Defina atributos de recursos com conectores em AWS SAM](#)

Fornecer feedback

Para fornecer feedback sobre conectores, [envie um novo problema](#) no repositório `AWS GitHubserverless-application-model`.

AWS SAM Modelos de políticas

O AWS Serverless Application Model (AWS SAM) permite que você escolha em uma lista de modelos de política para definir o escopo das permissões de suas funções de eventos e máquinas de AWS Step Functions estado do Lambda para os recursos que são usados pelo seu aplicativo.

AWS SAM os aplicativos do AWS Serverless Application Repository que usam modelos de política não exigem nenhum reconhecimento especial do cliente para implantar o aplicativo a partir do. AWS Serverless Application Repository

Se deseja solicitar um novo modelo de política a ser adicionado, faça o seguinte:

1. Envie uma pull request no arquivo de origem `policy_templates.json` na ramificação do projeto. `develop` AWS SAM GitHub Você pode encontrar o arquivo de origem em [policy_templates.json](#) no site. GitHub
2. Envie um problema no AWS SAM GitHub projeto que inclua os motivos da sua pull request e um link para a solicitação. Use este link para enviar um novo problema: [AWS Serverless Application Model: Problemas](#).

Sintaxe

Para cada modelo de política especificado em seu arquivo de AWS SAM modelo, você deve sempre especificar um objeto contendo os valores de espaço reservado do modelo de política. Se um modelo de política não exigir nenhum valor de espaço reservado, você deverá especificar um objeto vazio.

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 você configurou uma política do IAM regular ou gerenciou políticas por meio do Lambda, o modelo de política pode ser definido sem usar um objeto vazio.

Exemplos

Exemplo 1: modelo de política com valores de espaços reservados

O exemplo a seguir mostra que o modelo de política [SQSPollerPolicy](#) espera um QueueName como recurso. O AWS SAM modelo recupera o nome da fila "MyQueue" do Amazon SQS, que você pode criar no mesmo aplicativo ou solicitada como parâmetro para o aplicativo.

```
MyFunction:
  Type: 'AWS::Serverless::Function'
  Properties:
    CodeUri: ${codeuri}
    Handler: hello.handler
    Runtime: python2.7
    Policies:
      - SQSPollerPolicy:
        QueueName:
          !GetAtt MyQueue.QueueName
```

Exemplo 2: modelo de política sem valores sem valores de espaço reservado

O exemplo a seguir contém o modelo de política [CloudWatchPutMetricPolicy](#), que não tem valores de espaços reservados.

Note

Mesmo que não haja valores de espaço reservado, você deve especificar um objeto vazio, caso contrário, ocorrerá um erro.

```
MyFunction:
  Type: 'AWS::Serverless::Function'
  Properties:
    CodeUri: ${codeuri}
    Handler: hello.handler
    Runtime: python2.7
    Policies:
      - CloudWatchPutMetricPolicy: {}
```

Exemplo 3: modelo de política com valores de espaços reservados e uma política do IAM regular

O exemplo a seguir contém a AmazonSQSFullAccess política e o modelo [DynamoDBCrudPolicy](#) de política. A AmazonSQSFullAccess política é uma política do IAM e não uma AWS SAM política, então você não precisa especificar um objeto vazio, pois a política seria passada diretamente para CloudFormation o.

```
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

Tabela de modelos de política

Veja a seguir uma tabela dos modelos de política disponíveis.

Modelo de política	Description		
AcmGetCertificatePolicy	Permite ler um certificado de Gerenciador de certificados da AWS.		
AMIDescribePolicy	Concede permissão para descrever Imagens de máquina da Amazon (AMI).		
AthenaQueryPolicy	Concede permissões para executar consultas do Athena.		
AWSSecretsManagerGetSecretValuePolicy	Concede permissão para obter o valor secreto do segredo AWS Secrets Manager especificado.		
AWSSecretsManagerRotationPolicy	Permite a entrada de um segredo no AWS Secrets Manager.		
CloudFormationDescribeStacksPolicy	Dá permissão para descrever CloudFormation pilhas.		
CloudWatchDashboardPolicy	Concede permissões para colocar métricas em operação em CloudWatch painéis.		
CloudWatchDescribeAlarmsPolicy	Permite descrever o histórico CloudWatch de alarmes.		

Modelo de política	Description		
AlarmHistoryPolicy			
CloudWatchPutMetricPolicy	Permite enviar métricas para CloudWatch o.		
CodeCommitCrudPolicy	Concede permissões para create/read update/delete /objects em um CodeCommit repositório específico.		
CodeCommitReadPolicy	Concede permissões para ler objetos em um CodeCommit repositório específico.		
CodePipelineLambdaExecutionPolicy	Permite que uma função Lambda invocada por relate CodePipeline o status do trabalho.		
CodePipelineReadOnlyPolicy	Dá permissão de leitura para obter detalhes sobre um CodePipeline funil.		
ComprehendBasicAccessPolicy	Concede permissão para a detecção de entidades, frases-chave, idiomas e sentimentos.		
CostExplorerReadOnlyPolicy	Concede permissão somente leitura às APIs Cost Explorer somente leitura para histórico de faturamento.		
DynamoDBBackupFullAccessPolicy	Concede permissão de leitura e gravação aos backups do DynamoDB sob demanda para uma tabela.		

Modelo de política	Description		
<u>DynamoDBC rudPolicy</u>	Concede permissões de criação, leitura, atualização e exclusão a uma tabela do Amazon DynamoDB.		
<u>DynamoDBR eadPolicy</u>	Concede permissão somente leitura para uma tabela do DynamoDB.		
<u>DynamoDBR econfigur ePolicy</u>	Concede permissão para reconfigurar uma tabela do DynamoDB.		
<u>DynamoDBR estoreFro mBackupPolicy</u>	Concede permissão para restaurar uma tabela do DynamoDB do backup.		
<u>DynamoDBS treamRead Policy</u>	Concede permissão para descrever e ler streams e registros do DynamoDB.		
<u>DynamoDBW ritePolicy</u>	Concede permissão somente gravação a uma tabela do DynamoDB.		
<u>EC2CopyIm agePolicy</u>	Concede permissão para copiar imagens do Amazon EC2.		
<u>EC2Descri bePolicy</u>	Concede permissão para descrever instâncias do Amazon Elastic Compute Cloud (Amazon EC2).		
<u>EcsRunTas kPolicy</u>	Concede permissão para iniciar uma nova tarefa para uma definição de tarefa.		
<u>EFSWriteA ccessPolicy</u>	Concede permissão para montar um sistema de arquivos do Amazon EFS com acesso de gravação.		

Modelo de política	Description		
EKSDescribePolicy	Concede permissão para descrever ou listar clusters do Amazon EKS.		
ElasticMapReduceAddJobFlowStepsPolicy	Concede permissão para adicionar novas etapas a um cluster em execução.		
ElasticMapReduceCancelStepsPolicy	Concede permissão para cancelar uma etapa ou etapas pendentes em um cluster em execução.		
ElasticMapReduceModifyInstancesFleetPolicy	Concede permissão para listar detalhes e modificar capacidades para frotas de instâncias em um cluster.		
ElasticMapReduceModifyInstancesGroupsPolicy	Concede permissão para listar detalhes e modificar configurações para grupos de instâncias em um cluster.		
ElasticMapReduceSetTerminationProtectionPolicy	Concede permissão para definir a proteção contra encerramento para um cluster.		
ElasticMapReduceTerminateJobFlowsPolicy	Concede permissão para encerrar um cluster.		

Modelo de política	Description		
ElasticsearchHttpPostPolicy	Concede permissão POST ao Amazon OpenSearch Service.		
EventBridgePutEventsPolicy	Concede permissões para enviar eventos para EventBridge o.		
FilterLogEventsPolicy	Permite filtrar eventos de CloudWatch registros de um grupo de registros especificado.		
FirehoseCreatePolicy	Concede permissão para criar, gravar, atualizar e excluir um fluxo de entrega do Firehose.		
FirehoseWritePolicy	Concede permissão para gravar em um fluxo de entrega do Firehose.		
KinesisCreatePolicy	Concede permissão para criar, publicar e excluir um Amazon Kinesis Stream.		
KinesisStreamReadPolicy	Concede permissão para listar e ler um Amazon Kinesis Stream.		
KMSTDecryptPolicy	Dá permissão para descriptografar com uma chave AWS Key Management Service (AWS KMS).		
KMSEncryptPolicy	Dá permissão para criptografar com uma chave AWS Key Management Service (AWS KMS).		
LambdaInvokePolicy	Permite invocar uma AWS Lambda função, alias ou versão.		

Modelo de política	Description		
<u>MobileAnalyticsWriteOnlyAccessPolicy</u>	Concede permissão somente gravação para colocar dados de eventos para todos os recursos de aplicativos.		
<u>OrganizationsListAccountsPolicy</u>	Concede permissão somente para leitura para listar nomes e IDs de contas secundárias.		
<u>PinpointEndpointAccessPolicy</u>	Concede permissão para obter e atualizar endpoints para um aplicativo Amazon Pinpoint.		
<u>PollyFullAccessPolicy</u>	Concede permissão de acesso total aos recursos de léxico do Amazon Polly.		
<u>RekognitionDetectOnlyPolicy</u>	Concede permissão para detectar faces, rótulos e texto.		
<u>RekognitionFacesManagementPolicy</u>	Concede permissão para adicionar, excluir e pesquisar faces em uma coleção do Amazon Rekognition.		
<u>RekognitionFacesPolicy</u>	Concede permissão para comparar e detectar faces e rótulos.		
<u>RekognitionLabelsPolicy</u>	Concede permissão para detectar rótulos de objetos e moderação.		
<u>RekognitionNoDataAccessPolicy</u>	Concede permissão para comparar e detectar faces e rótulos.		

Modelo de política	Description		
<u>RekognitionReadPolicy</u>	Concede permissão para listar e pesquisar faces.		
<u>RekognitionWriteOnlyAccessPolicy</u>	Concede permissão para criar faces de coleção e índice.		
<u>Route53ChangeResourceRecordSetsPolicy</u>	Concede permissão para alterar conjuntos de registros de recursos no Route 53.		
<u>S3CrudPolicy</u>	Concede permissão de criação, leitura, atualização e exclusão para atuar nos objetos em um bucket do Amazon S3.		
<u>S3FullAccessPolicy</u>	Concede permissão de acesso total para atuar nos objetos em um bucket do Amazon S3.		
<u>S3ReadPolicy</u>	Concede permissão somente leitura para ler objetos em um bucket do Amazon Simple Storage Service (Amazon S3).		
<u>S3WritePolicy</u>	Concede permissão de gravação para gravar objetos em um bucket do Amazon S3.		
<u>SageMakerCreateEndpointConfigurationPolicy</u>	Permite criar uma configuração de endpoint na SageMaker IA.		
<u>SageMakerCreateEndpointPolicy</u>	Dá permissão para criar um endpoint na SageMaker IA.		

Modelo de política	Description		
<u>ServerlessRepoReadWriteAccessPolicy</u>	Permite criar e listar aplicativos no AWS Serverless Application Repository serviço.		
<u>SESBulkTemplatedCrudPolicy</u>	Concede permissão para enviar e-mail, e-mail modelo, e-mail em massa modelo e verificar identidade.		
<u>SESBulkTemplatedCrudPolicy_v2</u>	Concede permissão para enviar e-mails, e-mails modelados e e-mails em massa modelados para o Amazon SES e para verificar a identidade.		
<u>SESCrudPolicy</u>	Concede permissão para enviar e-mails e verificar identidades.		
<u>SESEmailTemplateCrudPolicy</u>	Concede permissão para criar, obter, listar, atualizar e excluir modelos de e-mail do Amazon SES.		
<u>SESSendBouncePolicy</u>	Concede SendBounce permissão a uma identidade do Amazon Simple Email Service (Amazon SES).		
<u>SNSCrudPolicy</u>	Concede permissão para criar, publicar e assinar tópicos do Amazon SNS.		
<u>SNSPublishMessagePolicy</u>	Concede permissão para publicar mensagens de eventos em um tópico do Amazon Simple Notification Service (Amazon SNS).		
<u>SQSPollerPolicy</u>	Concede permissão para pesquisar uma fila do Amazon Simple Queue Service (Amazon SQS).		

Modelo de política	Description		
SQSSendMessagePolicy	Concede permissão para enviar mensagens para uma fila do Amazon SQS.		
SSMParameterReadPolicy	Concede permissão para acessar um parâmetro de um armazenamento de parâmetros do Amazon EC2 Systems Manager (SSM) para carregar segredos nesta conta. Use quando o nome do parâmetro não tiver prefixo de barra.		
SSMParameterWithSlashPrefixReadPolicy	Concede permissão para acessar um parâmetro de um armazenamento de parâmetros do Amazon EC2 Systems Manager (SSM) para carregar segredos nessa conta. Use quando o nome do parâmetro tiver prefixo de barra.		
StepFunctionsExecutionPolicy	Concede permissão para iniciar a execução de uma máquina de estado do Step Functions.		
TextractDetectAnalyzePolicy	Concede acesso para detectar e analisar documentos com Amazon Textract.		
TextractGetResultPolicy	Concede acesso para detectar e analisar documentos com o Amazon Textract.		
TextractPolicy	Concede acesso total ao Amazon Textract.		
VPCAccessPolicy	Concede acesso para criar, excluir, descrever e desanexar interfaces de rede elásticas.		

Solução de problemas

Erro de CLI do SAM: “É necessário especificar valores de parâmetros válidos para o modelo de política '<>policy-template-name'”

Ao executar `sam build`, você verá o seguinte erro:

```
"Must specify valid parameter values for policy template '<policy-template-name>'"
```

Isso significa que você não passou um objeto vazio ao declarar um modelo de política que não tem nenhum valor de espaço reservado.

Para corrigir isso, declare a política como no exemplo a seguir para [CloudWatchPutMetricPolicy](#).

```
MyFunction:
  Policies:
    - CloudWatchPutMetricPolicy: {}
```

AWS SAM lista de modelos de políticas

A seguir estão os modelos de política disponíveis, juntamente com as permissões que são aplicadas a cada um. AWS Serverless Application Model (AWS SAM) preenche automaticamente os itens de espaço reservado (como AWS região e ID da conta) com as informações apropriadas.

Tópicos

- [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

Dá permissão para ler um certificado de Gerenciador de certificados da AWS.

```
"Statement": [  
  {  
    "Effect": "Allow",  
    "Action": [  
      "acm:GetCertificate"  
    ],  
    "Resource": {  
      "Fn::Sub": [  
        "${certificateArn}",  
        {  
          "certificateArn": {  
            "Ref": "CertificateArn"  
          }  
        }  
      ]  
    }  
  }  
]
```

AMIDescribePolicy

Concede permissão para descrever Imagens de máquina da Amazon (AMI).

```
"Statement": [  
  {  
    "Effect": "Allow",  
    "Action": [  

```

```

    "ec2:DescribeImages"
  ],
  "Resource": "*"
}
]

```

AthenaQueryPolicy

Concede permissões para executar consultas do 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"
      }
    }
  ]
}
]

```

AWS Secrets Manager GetSecretValuePolicy

Dá permissão para obter o valor secreto do AWS Secrets Manager segredo especificado.

```

"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "secretsmanager:GetSecretValue"
    ],
    "Resource": {
      "Fn::Sub": [
        "${secretArn}",
        {
          "secretArn": {
            "Ref": "SecretArn"
          }
        }
      ]
    }
  }
]

```

AWS Secrets Manager RotationPolicy

Permite a entrada de um segredo no AWS Secrets Manager.

```

"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

Dá permissão para descrever CloudFormation pilhas.

```

"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "cloudformation:DescribeStacks"
    ],

```



```
"Resource": {
  "Fn::Sub": "arn:${AWS::Partition}:cloudformation:${AWS::Region}:
${AWS::AccountId}:stack/*"
}
}
```

CloudWatchDashboardPolicy

Concede permissões para colocar métricas em operação em CloudWatch painéis.

```
"Statement": [
{
  "Effect": "Allow",
  "Action": [
    "cloudwatch:GetDashboard",
    "cloudwatch:ListDashboards",
    "cloudwatch:PutDashboard",
    "cloudwatch:ListMetrics"
  ],
  "Resource": "*"
}
]
```

CloudWatchDescribeAlarmHistoryPolicy

Permite descrever o histórico de CloudWatch alarmes da Amazon.

```
"Statement": [
{
  "Effect": "Allow",
  "Action": [
    "cloudwatch:DescribeAlarmHistory"
  ],
  "Resource": "*"
}
]
```

CloudWatchPutMetricPolicy

Dá permissão para enviar métricas para CloudWatch o.

```
"Statement": [
```

```
{
  "Effect": "Allow",
  "Action": [
    "cloudwatch:PutMetricData"
  ],
  "Resource": "*"
}
```

CodePipelineLambdaExecutionPolicy

Permite que uma função de evento Lambda invocada por relate AWS CodePipeline o status do trabalho.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "codepipeline:PutJobSuccessResult",
      "codepipeline:PutJobFailureResult"
    ],
    "Resource": "*"
  }
]
```

CodePipelineReadOnlyPolicy

Dá permissão de leitura para obter detalhes sobre um CodePipeline funil.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "codepipeline:ListPipelineExecutions"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:codepipeline:${AWS::Region}:${AWS::AccountId}:"
        "${pipelinename}",
        {
          "pipelinename": {
            "Ref": "PipelineName"
          }
        }
      ]
    }
  }
]
```

```

    }
  ]
}
}
]

```

CodeCommitCrudPolicy

Concede permissões para criar, ler, atualizar e excluir objetos em um CodeCommit repositório específico.

```

"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

Concede permissões para ler objetos em um CodeCommit repositório específico.

```

"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

Concede permissão para a detecção de entidades, frases-chave, idiomas e sentimentos.

```

"Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "comprehend:BatchDetectKeyPhrases",
            "comprehend:DetectDominantLanguage",
            "comprehend:DetectEntities",
            "comprehend:BatchDetectEntities",
            "comprehend:DetectKeyPhrases",
            "comprehend:DetectSentiment",
            "comprehend:BatchDetectDominantLanguage",
            "comprehend:BatchDetectSentiment"
        ],
        "Resource": "*"
    }
]

```

CostExplorerReadOnlyPolicy

Concede permissão somente de leitura às APIs somente leitura (Cost AWS Cost Explorer Explorer) do histórico de faturamento.

```

"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "ce:GetCostAndUsage",
      "ce:GetDimensionValues",
      "ce:GetReservationCoverage",
      "ce:GetReservationPurchaseRecommendation",
      "ce:GetReservationUtilization",
      "ce:GetTags"
    ],
    "Resource": "*"
  }
]

```

DynamoDBBackupFullAccessPolicy

Concede permissão de leitura e gravação aos backups do DynamoDB sob demanda para uma tabela.

```

"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

Concede permissões de criação, leitura, atualização e exclusão a uma tabela do 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

Concede permissão somente leitura para uma tabela do 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

Concede permissão para reconfigurar uma tabela do DynamoDB.

```

"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "dynamodb:UpdateTable"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:dynamodb:${AWS::Region}:${AWS::AccountId}:table/
${tableName}",
        {
          "tableName": {
            "Ref": "TableName"
          }
        }
      ]
    }
  }
]

```

DynamoDBRestoreFromBackupPolicy

Concede permissão para restaurar uma tabela do DynamoDB do 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

Concede permissão para descrever e ler streams e registros do 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

Concede permissão somente gravação a uma tabela do 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

Concede permissão para copiar imagens do Amazon EC2.

```
"Statement": [  
  {  
    "Effect": "Allow",  
    "Action": [  
      "ec2:CopyImage"  
    ],  
    "Resource": {  
      "Fn::Sub": [  
        "arn:${AWS::Partition}:ec2:${AWS::Region}:${AWS::AccountId}:image/${imageId}",  
        {  
          "imageId": {  
            "Ref": "ImageId"  
          }  
        }  
      ]  
    }  
  }  
]
```

EC2DescribePolicy

Concede permissão para descrever instâncias do Amazon Elastic Compute Cloud (Amazon EC2).

```
"Statement": [  
  {  
    "Effect": "Allow",  
    "Action": [  
      "ec2:DescribeRegions",  
      "ec2:DescribeInstances"  
    ],  
    "Resource": "*"  
  }  
]
```

EcsRunTaskPolicy

Concede permissão para iniciar uma nova tarefa para uma definição de tarefa.

```
"Statement": [  
  {
```

```

{
  "Action": [
    "ecs:RunTask"
  ],
  "Resource": {
    "Fn::Sub": [
      "arn:${AWS::Partition}:ecs:${AWS::Region}:${AWS::AccountId}:task-definition/
${taskDefinition}",
      {
        "taskDefinition": {
          "Ref": "TaskDefinition"
        }
      }
    ]
  },
  "Effect": "Allow"
}
]

```

EFSWriteAccessPolicy

Concede permissão para montar um sistema de arquivos do Amazon EFS com acesso de gravação.

```

"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

Concede permissão para descrever ou listar clusters do Amazon Elastic Kubernetes Service (Amazon EKS).

```

"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "eks:DescribeCluster",
      "eks:ListClusters"
    ],
    "Resource": "*"
  }
]

```

ElasticMapReduceAddJobFlowStepsPolicy

Concede permissão para adicionar novas etapas a um cluster em execução.

```

"Statement": [
  {
    "Action": "elasticmapreduce:AddJobFlowSteps",
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:elasticmapreduce:${AWS::Region}:",
        "${AWS::AccountId}:cluster/${clusterId}",

```



```

        {
            "clusterId": {
                "Ref": "ClusterId"
            }
        }
    ],
    "Effect": "Allow"
}
]

```

ElasticMapReduceCancelStepsPolicy

Concede permissão para cancelar uma etapa ou etapas pendentes em um cluster em execução.

```

"Statement": [
    {
        "Action": "elasticmapreduce:CancelSteps",
        "Resource": {
            "Fn::Sub": [
                "arn:${AWS::Partition}:elasticmapreduce:${AWS::Region}:",
                "${AWS::AccountId}:cluster/${clusterId}",
                {
                    "clusterId": {
                        "Ref": "ClusterId"
                    }
                }
            ]
        },
        "Effect": "Allow"
    }
]

```

ElasticMapReduceModifyInstanceFleetPolicy

Concede permissão para listar detalhes e modificar capacidades para frotas de instâncias em um 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

Concede permissão para listar detalhes e modificar configurações para grupos de instâncias em um 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

Concede permissão para definir a proteção contra encerramento para um cluster.

```
"Statement": [  
  {  
    "Action": "elasticmapreduce:SetTerminationProtection",  
    "Resource": {  
      "Fn::Sub": [  
        "arn:${AWS::Partition}:elasticmapreduce:${AWS::Region}:  
${AWS::AccountId}:cluster/${clusterId}",  
        {  
          "clusterId": {  
            "Ref": "ClusterId"  
          }  
        }  
      ]  
    },  
    "Effect": "Allow"  
  }  
]
```

ElasticMapReduceTerminateJobFlowsPolicy

Concede permissão para encerrar um cluster.

```
"Statement": [  
  {  
    "Action": "elasticmapreduce:TerminateJobFlows",  
    "Resource": {  
      "Fn::Sub": [  
        "arn:${AWS::Partition}:elasticmapreduce:${AWS::Region}:  
${AWS::AccountId}:cluster/${clusterId}",  
        {  
          "clusterId": {  
            "Ref": "ClusterId"  
          }  
        }  
      ]  
    },  
    "Effect": "Allow"  
  }  
]
```

ElasticsearchHttpPostPolicy

Concede permissão POST e PUT ao 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

Concede permissões para enviar eventos para a 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

Permite filtrar eventos de CloudWatch registros de um grupo de registros especificado.

```
"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

Concede permissão para criar, gravar, atualizar e excluir um fluxo de entrega do 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

Concede permissão para gravar em um fluxo de entrega do 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

Concede permissão para criar, publicar e excluir um Amazon Kinesis Stream.

```

"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

Concede permissão para listar e ler um Amazon Kinesis Stream.

```

"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

Dá permissão para descriptografar com uma chave AWS Key Management Service (AWS KMS). Observe que `keyId` deve ser um ID de AWS KMS chave e não um alias de chave.

```

"Statement": [
  {
    "Action": "kms:Decrypt",
    "Effect": "Allow",
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:kms:${AWS::Region}:${AWS::AccountId}:key/${keyId}",
        {
          "keyId": {
            "Ref": "KeyId"
          }
        }
      ]
    }
  }
]

```



```

    }
  }
]
}
]

```

KMSEncryptPolicy

Dá permissão para criptografar com uma AWS KMS chave. Observe que keyID deve ser um ID de AWS KMS chave e não um alias de chave.

```

"Statement": [
  {
    "Action": "kms:Encrypt",
    "Effect": "Allow",
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:kms:${AWS::Region}:${AWS::AccountId}:key/${keyId}",
        {
          "keyId": {
            "Ref": "KeyId"
          }
        }
      ]
    }
  }
]

```

LambdaInvokePolicy

Permite invocar uma AWS Lambda função, alias ou versão.

```

"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "lambda:InvokeFunction"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:lambda:${AWS::Region}:${AWS::AccountId}:function:
${functionName}*",

```

```
    {
      "functionName": {
        "Ref": "FunctionName"
      }
    }
  ]
}
```

MobileAnalyticsWriteOnlyAccessPolicy

Concede permissão somente gravação para colocar dados de eventos para todos os recursos de aplicativos.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "mobileanalytics:PutEvents"
    ],
    "Resource": "*"
  }
]
```

OrganizationsListAccountsPolicy

Concede permissão somente para leitura para listar nomes e IDs de contas secundárias.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "organizations:ListAccounts"
    ],
    "Resource": "*"
  }
]
```

PinpointEndpointAccessPolicy

Concede permissão para obter e atualizar endpoints para um aplicativo 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

Concede permissão de acesso total aos recursos de léxico do 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

Concede permissão para detectar faces, rótulos e texto.

```

"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "rekognition:DetectFaces",
      "rekognition:DetectLabels",
      "rekognition:DetectModerationLabels",
      "rekognition:DetectText"
    ],
    "Resource": "*"
  }
]

```

RekognitionFacesManagementPolicy

Concede permissão para adicionar, excluir e pesquisar faces em uma coleção do 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

Concede permissão para comparar e detectar faces e rótulos.

```
"Statement": [  
  {  
    "Effect": "Allow",  
    "Action": [  
      "rekognition:CompareFaces",  
      "rekognition:DetectFaces"  
    ],  
    "Resource": "*"  
  }  
]
```

RekognitionLabelsPolicy

Concede permissão para detectar rótulos de objetos e moderação.

```
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "rekognition:DetectLabels",
      "rekognition:DetectModerationLabels"
    ],
    "Resource": "*"
  }
]
```

RekognitionNoDataAccessPolicy

Concede permissão para comparar e detectar faces e rótulos.

```
"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

Concede permissão para listar e pesquisar faces.

```
"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

Concede permissão para criar faces de coleção e índice.

```

"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

Concede permissão para alterar conjuntos de registros de recursos no Route 53.

```
"Statement": [  
  {  
    "Effect": "Allow",  
    "Action": [  
      "route53:ChangeResourceRecordSets"  
    ],  
    "Resource": {  
      "Fn::Sub": [  
        "arn:${AWS::Partition}:route53::hostedzone/${HostedZoneId}",  
        {  
          "HostedZoneId": {  
            "Ref": "HostedZoneId"  
          }  
        }  
      ]  
    }  
  }  
]
```

S3CrudPolicy

Concede permissão de criação, leitura, atualização e exclusão para atuar nos objetos em um bucket do 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

Concede permissão de acesso total para atuar nos objetos em um bucket do 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

Concede permissão somente leitura para ler objetos em um bucket do 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

Concede permissão de gravação para gravar objetos em um bucket do 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

Dá permissão para criar uma configuração de endpoint na SageMaker IA.

```

"Statement": [
  {
    "Action": [
      "sagemaker:CreateEndpointConfig"
    ],
    "Resource": {

```

```

    "Fn::Sub": [
      "arn:${AWS::Partition}:sagemaker:${AWS::Region}:${AWS::AccountId}:endpoint-
config/${endpointConfigName}",
      {
        "endpointConfigName": {
          "Ref": "EndpointConfigName"
        }
      }
    ],
    "Effect": "Allow"
  }
]

```

SageMakerCreateEndpointPolicy

Dá permissão para criar um endpoint na SageMaker IA.

```

"Statement": [
  {
    "Action": [
      "sagemaker:CreateEndpoint"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:sagemaker:${AWS::Region}:${AWS::AccountId}:endpoint/
${endpointName}",
        {
          "endpointName": {
            "Ref": "EndpointName"
          }
        }
      ]
    },
    "Effect": "Allow"
  }
]

```

ServerlessRepoReadWriteAccessPolicy

Dá permissão para criar e listar aplicativos no serviço 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

Concede permissão para enviar e-mails, e-mails modelados e e-mails em massa modelados para o Amazon SES e para verificar a identidade.

Note

A ação `ses:SendTemplatedEmail` requer um modelo de ARN. Use `SESBulkTemplatedCrudPolicy_v2` em vez disso.

```

"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

Concede permissão para enviar e-mails, e-mails modelados e e-mails em massa modelados para o Amazon SES e para verificar a identidade.

```

"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

Concede permissão para enviar e-mails e verificar identidades.

```

"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

Concede permissão para criar, obter, listar, atualizar e excluir modelos de e-mail do Amazon SES.

```
"Statement": [  
  {  
    "Effect": "Allow",  
    "Action": [  
      "ses:CreateTemplate",  
      "ses:GetTemplate",  
      "ses:ListTemplates",  
      "ses:UpdateTemplate",  
      "ses>DeleteTemplate",  
      "ses:TestRenderTemplate"  
    ],  
    "Resource": "*"   
  }  
]
```

SESSendBouncePolicy

Concede SendBounce permissão a uma identidade do 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

Concede permissão para criar, publicar e assinar tópicos do 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

Concede permissão para publicar mensagens de eventos em um tópico do 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

Concede permissão para pesquisar uma fila do 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

Concede permissão para enviar mensagens para uma fila do Amazon SQS.

```

"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "sqs:SendMessage*"
    ],
    "Resource": {
      "Fn::Sub": [
        "arn:${AWS::Partition}:sqs:${AWS::Region}:${AWS::AccountId}:${queueName}",
        {
          "queueName": {
            "Ref": "QueueName"
          }
        }
      ]
    }
  }
]

```

SSMParameterReadPolicy

Concede permissão para acessar um parâmetro de um armazenamento de parâmetros do Amazon EC2 Systems Manager (SSM) para carregar segredos nesta conta. Use quando o nome do parâmetro não tiver prefixo de barra.

Note

Se você não estiver usando a chave padrão, também precisará da política de `KMSDecryptPolicy`.

```

"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

Concede permissão para acessar um parâmetro de um armazenamento de parâmetros do Amazon EC2 Systems Manager (SSM) para carregar segredos nessa conta. Use quando o nome do parâmetro tiver prefixo de barra.

Note

Se você não estiver usando a chave padrão, também precisará da política de `KMSDecryptPolicy`.

```

"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

Concede permissão para iniciar a execução de uma máquina de estado do 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

Concede acesso para detectar e analisar documentos com Amazon Textract.

```
"Statement": [  
  {  
    "Effect": "Allow",  
    "Action": [  
      "textract:DetectDocumentText",  
      "textract:StartDocumentTextDetection",  
      "textract:StartDocumentAnalysis",  
      "textract:AnalyzeDocument"  
    ],  
    "Resource": "*"   
  }  
]
```

TextractGetResultPolicy

Concede acesso para detectar e analisar documentos com o Amazon Textract.

```
"Statement": [  
  {  
    "Effect": "Allow",  
    "Action": [  
      "textract:GetDocumentTextDetection",  
      "textract:GetDocumentAnalysis"  
    ],  
    "Resource": "*"   
  }  
]
```

TextractPolicy

Concede acesso total ao Amazon Textract.

```
"Statement": [  
  {  
    "Effect": "Allow",  
    "Action": [  
      "textract:*"  
    ],  
    "Resource": "*"   
  }  
]
```

```
"Resource": "*"
}
```

VPCAccessPolicy

Concede acesso para criar, excluir, descrever e desanexar interfaces de rede elásticas.

```
"Statement": [
{
  "Effect": "Allow",
  "Action": [
    "ec2:CreateNetworkInterface",
    "ec2>DeleteNetworkInterface",
    "ec2:DescribeNetworkInterfaces",
    "ec2:DetachNetworkInterface"
  ],
  "Resource": "*"
}
```

Gerenciando AWS SAM permissões com CloudFormation mecanismos

Para controlar o acesso aos AWS recursos, o AWS Serverless Application Model (AWS SAM) pode usar os mesmos mecanismos que CloudFormation. Para obter mais informações, consulte [Controlar o acesso com o AWS Identity and Access Management](#) no Guia do usuário do AWS CloudFormation

Há três opções principais para conceder permissão ao usuário para gerenciar aplicativos sem servidor. Cada opção fornece aos usuários diferentes níveis de controle de acesso.

- Conceder permissões de administrador.
- Anexe as políticas AWS gerenciadas necessárias.
- Conceda permissões específicas AWS Identity and Access Management (IAM).

Dependendo da opção escolhida, os usuários podem gerenciar somente aplicativos sem servidor que contenham AWS recursos que eles tenham permissão para acessar.

As seções a seguir descrevem essa opção em mais detalhes.

Conceder permissões de administrador

Se você conceder permissões de administrador a um usuário, ele poderá gerenciar aplicativos sem servidor que contenham qualquer combinação de AWS recursos. Essa é a opção mais simples, mas também concede aos usuários o conjunto mais amplo de permissões, o que lhes permite realizar ações com o maior impacto.

Para obter mais informações sobre como conceder permissões de administrador a um usuário, consulte [Criar seu primeiro usuário e grupo administrador do IAM](#) no Guia do usuário do IAM.

Anexe as políticas AWS gerenciadas necessárias

Você pode conceder aos usuários um subconjunto de permissões usando as [políticas gerenciadas pela AWS](#), em vez de conceder permissões completas de administrador. Se você usar essa opção, verifique se o conjunto de políticas AWS gerenciadas abrange todas as ações e recursos necessários para os aplicativos sem servidor que os usuários gerenciam.

Por exemplo, as seguintes políticas AWS gerenciadas são suficientes para [implantar o aplicativo Hello World de amostra](#):

- AWSCloudFormationFullAccess
- IAMFullAcesso
- AWSLambda_FullAccess
- APIGatewayAdministrador da Amazon
- Amazon S3 FullAccess
- Amazon EC2 ContainerRegistryFullAccess

Para obter informações sobre como anexar políticas a um usuário do IAM, consulte [Alterar permissões para um usuário do IAM](#) no Guia do usuário do IAM.

Conceda permissões específicas do IAM

Para obter o nível mais granular de controle de acesso, você pode conceder permissões específicas do IAM aos usuários usando [declarações de política](#). Se você usar essa opção, certifique-se de que a declaração de política inclua todas as ações e recursos necessários para os aplicativos sem servidor que os usuários gerenciam.

A melhor prática com essa opção é negar aos usuários a permissão para criar funções, incluindo funções de execução do Lambda, para que eles não possam conceder a si mesmos permissões

escalonadas. Portanto, você, como administrador, deve primeiro criar uma [função de execução do Lambda](#) que será especificada nos aplicativos sem servidor que os usuários gerenciarão. Para obter informações sobre a criação de funções de execução do Lambda, consulte [Criação de uma função de execução no console do IAM](#).

Para o [aplicativo Hello World de amostra](#), `AWSLambdaBasicExecutionRole` é suficiente para executar o aplicativo. Depois de criar uma função de execução do Lambda, modifique o arquivo de AWS SAM modelo do aplicativo Hello World de amostra para adicionar a seguinte propriedade ao `AWS::Serverless::Function` recurso:

Role: `lambda-execution-role-arn`

Com o aplicativo Hello World modificado em vigor, a declaração de política a seguir concede permissões suficientes para que os usuários implantem, atualizem e excluam o aplicativo:

Note

O exemplo de declaração de política nesta seção concede permissão suficiente para você implantar, atualizar e excluir a [amostra do aplicativo Hello World](#). Se você adicionar outros tipos de recursos ao seu aplicativo, precisará atualizar a declaração de política para incluir o seguinte:

1. Permissão para que seu aplicativo chame as ações do serviço.
2. O diretor do serviço, se necessário para as ações do serviço.

Por exemplo, se você adicionar um fluxo de trabalho do Step Functions, talvez seja necessário adicionar permissões para as ações listadas [aqui](#) e para o responsável pelo serviço `principalstates.amazonaws.com`.

Para obter mais informações sobre o gerenciamento de políticas, consulte [Gerenciar políticas do IAM](#) no Guia do usuário do IAM.

Controle o acesso à API com seu AWS SAM modelo

O controle do acesso às APIs do API Gateway ajuda a garantir que a aplicação sem servidor esteja segura e só possa ser acessada por meio da autorização que você habilitar. Você pode ativar

a autorização em seu AWS SAM modelo para controlar quem pode acessar suas APIs do API Gateway.

AWS SAM suporta vários mecanismos para controlar o acesso às suas APIs do API Gateway. O conjunto de mecanismos suportados difere entre os tipos de recursos `AWS::Serverless::HttpApi` e `AWS::Serverless::Api`.

A tabela a seguir resume os mecanismos suportados por cada tipo de recurso.

Mecanismos para controlar o acesso	<code>AWS::Serverless::HttpApi</code>	<code>AWS::Serverless::Api</code>
Autorizadores do Lambda	✓	✓
Permissões do IAM		✓
Grupos de usuários do Amazon Cognito	✓ *	✓
Chaves de API		✓
Políticas de recursos		✓
OAuth 2.0/JWT autorizadores	✓	

* O Amazon Cognito pode ser usado como emissor do JSON Web Token (JWT) com o tipo de recurso `AWS::Serverless::HttpApi`.

- **Autorizadores Lambda** — Um autorizador Lambda (anteriormente conhecido como autorizador personalizado) é uma função de evento Lambda que você fornece para controlar o acesso à sua API. Quando sua API é chamada, essa função do Lambda é invocada com um contexto de solicitação ou um token de autorização fornecido pelo aplicativo cliente. A função do Lambda responde se o chamador está autorizado a realizar a operação solicitada.

Tanto o tipo `AWS::Serverless::HttpApi` quanto o tipo `AWS::Serverless::Api` de recurso oferecem suporte aos autorizadores Lambda.

Para obter mais informações sobre autorizadores do Lambda com `AWS::Serverless::HttpApi`, consulte [Trabalhar com autorizadores AWS Lambda para](#)

[APIs HTTP](#) no Guia do desenvolvedor do API Gateway. Para obter mais informações sobre autorizadores do Lambda com `AWS::Serverless::Api`, consulte [Usar os autorizadores do Lambda do API Gateway](#) no Guia do desenvolvedor do API Gateway.

Para obter exemplos de autorizadores Lambda para qualquer tipo de recurso, consulte [Exemplos de autorizadores Lambda para AWS SAM](#).

- **Permissões do IAM** — Você pode controlar quem pode invocar sua API usando [as permissões AWS Identity and Access Management \(IAM\)](#). Os usuários que chamam sua API devem ser autenticados com as credenciais do IAM. As chamadas para sua API são bem-sucedidas somente se houver uma política do IAM anexada ao usuário do IAM que representa o chamador da API, um grupo do IAM que contém o usuário ou um perfil do IAM que o usuário assume.

Somente o tipo `AWS::Serverless::Api` de recurso é compatível com permissões do IAM.

Para mais informações [Controlar o acesso a uma API com permissões do IAM](#) no Guia do desenvolvedor do API Gateway. Para ver um exemplo, consulte [Exemplo de permissão do IAM para AWS SAM](#).

- **Grupos de usuários do Amazon Cognito** — Os grupos de usuários do Amazon Cognito são diretórios de usuários no Amazon Cognito. Um cliente da sua API deve primeiro inscrever um usuário no grupo de usuários e obter uma identidade ou token de acesso para o usuário. Em seguida, o cliente chama sua API com um dos tokens retornados. A chamada da API será bem-sucedida somente se o token necessário for válido.

O tipo `AWS::Serverless::Api` de recurso é compatível com grupos de usuários do Amazon Cognito. O tipo `AWS::Serverless::HttpApi` de recurso aceita o uso do Amazon Cognito como emissor do JWT.

Para mais informações, consulte [Controlar o acesso a uma API REST usando um grupo de usuários do Amazon Cognito como autorizador](#) no Guia do desenvolvedor do API Gateway. Para ver um exemplo, consulte [Exemplo de grupo de usuários do Amazon Cognito para AWS SAM](#).

- **Chaves de API** – As chaves de API são valores de strings alfanuméricas distribuídas para clientes de desenvolvedores de aplicativos para conceder acesso à API.

Somente o tipo `AWS::Serverless::Api` de recurso é compatível com chaves de API.

Para obter mais informações, consulte [Criação e uso de planos de uso com chaves de API](#) no Guia do desenvolvedor do API Gateway. Para ver um exemplo de chaves de API, consulte [Exemplo de chave de API para AWS SAM](#).

- Políticas de recurso — Políticas de recurso são documentos de políticas JSON que você pode anexar a uma API do API Gateway. Use políticas de recursos para controlar se um principal especificado (normalmente um usuário ou perfil do IAM) pode invocar a API.

Somente o tipo de recurso `AWS::Serverless::Api` suporta políticas de recursos como um mecanismo para controlar o acesso às APIs do API Gateway.

Para obter mais informações sobre políticas de recursos, consulte Como [controlar o acesso a uma API com as políticas de recursos do API Gateway](#) no Guia do desenvolvedor do API Gateway.

Para obter exemplos de políticas de recursos, consulte [Exemplo de política de recursos para AWS SAM](#).

- OAuth 2.0/JWT autorizadores — Você pode usar o JWTs como parte das estruturas [OpenID Connect \(OIDC\) e OAuth 2.0](#) para controlar o acesso às suas APIs. O API Gateway valida os JWTs enviados pelos clientes com solicitações de API e permite ou rejeita solicitações com base na validação do token e, opcionalmente, escopos no token.

Somente o tipo `AWS::Serverless::HttpApi` de recurso é compatível com OAuth 2.0/JWT autorizadores.

Para mais informações, consulte [Controlar o acesso a APIs HTTP com autorizadores JWT](#) no Guia do desenvolvedor do API Gateway. Para ver um exemplo, consulte [OAuth 2.0/JWT exemplo de autorizador para AWS SAM](#).

Escolher um mecanismo para controlar o acesso

O mecanismo que você escolhe usar para controlar o acesso às suas APIs do API Gateway depende de alguns fatores. Por exemplo, se você tem um projeto novo sem autorização ou controle de acesso configurado, os grupos de usuários do Amazon Cognito podem ser sua melhor opção. Isso ocorre porque, ao configurar grupos de usuários, você também configura automaticamente a autenticação e o controle de acesso.

No entanto, se seu aplicativo já tiver a autenticação configurada, usar autorizadores Lambda pode ser sua melhor opção. Isso ocorre porque você pode chamar seu serviço de autenticação existente e retornar um documento de política com base na resposta. Além disso, se seu aplicativo

exigir autenticação personalizada ou lógica de controle de acesso que os grupos de usuários não suportam, os autorizadores Lambda podem ser sua melhor opção.

Depois de escolher qual mecanismo usar, consulte a seção correspondente [Exemplos](#) para saber como usar AWS SAM para configurar seu aplicativo para usar esse mecanismo.

Como personalizar respostas de erro

Você pode usar AWS SAM para personalizar o conteúdo de algumas respostas de erro do API Gateway. Somente o tipo `AWS::Serverless::Api` de recurso é compatível com respostas personalizadas do API Gateway.

Para obter mais informações sobre as respostas do API Gateway, consulte [Respostas do gateway no API Gateway](#) no Guia do desenvolvedor do API Gateway. Para obter um exemplo de respostas personalizadas, consulte [Exemplo de resposta personalizada para AWS SAM](#).

Exemplos

- [Exemplos de autorizadores Lambda para AWS SAM](#)
- [Exemplo de permissão do IAM para AWS SAM](#)
- [Exemplo de grupo de usuários do Amazon Cognito para AWS SAM](#)
- [Exemplo de chave de API para AWS SAM](#)
- [Exemplo de política de recursos para AWS SAM](#)
- [OAuth 2.0/JWT exemplo de autorizador para AWS SAM](#)
- [Exemplo de resposta personalizada para AWS SAM](#)

Exemplos de autorizadores Lambda para AWS SAM

O tipo `AWS::Serverless::Api` de recurso oferece suporte a dois tipos de autorizadores Lambda: autorizadores e TOKEN autorizadores REQUEST. O tipo de recurso `AWS::Serverless::HttpApi` suporta somente autorizadores REQUEST. Veja a seguir exemplos de cada parâmetro.

Exemplo de autorizador Lambda **TOKEN** (`AWS::Serverless::API`)

Você pode controlar o acesso às suas APIs definindo um autorizador TOKEN Lambda em seu modelo. AWS SAM Para fazer isso, você usa o tipo de [ApiAuth](#) dados.

Veja a seguir um exemplo de seção AWS SAM de modelo para um autorizador LambdaTOKEN:

 Note

No exemplo a seguir, FunctionRole do SAM é gerado 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
```

Para obter mais informações sobre autorizadores do Lambda, consulte [Usar autorizadores Lambda do API Gateway](#) no Guia do desenvolvedor do API Gateway.

Exemplo de autorizador Lambda **REQUEST** (AWS::Serverless::API)

Você pode controlar o acesso às suas APIs definindo um autorizador REQUEST Lambda em seu modelo. AWS SAM Para fazer isso, você usa o tipo de [ApiAuth](#) dados.

Veja a seguir um exemplo de seção AWS SAM de modelo para um autorizador LambdaREQUEST:

```
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
```


Para obter mais informações sobre autorizadores do Lambda, consulte [Usar autorizadores Lambda do API Gateway](#) no Guia do desenvolvedor do API Gateway.

Exemplo de autorizador Lambda (AWS::Serverless::HttpApi)

Você pode controlar o acesso às suas APIs HTTP definindo um autorizador Lambda em seu modelo. AWS SAM Para fazer isso, você usa o tipo de [HttpApiAuth](#) dados.

Veja a seguir um exemplo de seção AWS SAM de modelo para um autorizador 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
```

Exemplo de permissão do IAM para AWS SAM

Você pode controlar o acesso às suas APIs definindo permissões do IAM no modelo AWS SAM . Para fazer isso, você usa o tipo de [ApiAuth](#) dados.

Veja a seguir um exemplo AWS SAM de modelo usado para permissões do 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}
```

Para obter mais informações sobre as permissões do IAM, consulte [Controlar o acesso para invocar uma API](#) no Guia do desenvolvedor do API Gateway.

Exemplo de grupo de usuários do Amazon Cognito para AWS SAM

Você pode controlar o acesso às suas APIs definindo grupos de usuários do Amazon Cognito dentro AWS SAM do seu modelo. Para fazer isso, você usa o tipo de [ApiAuth](#) dados.

Veja a seguir um exemplo de seção AWS SAM de modelo para um grupo de usuários:

```
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
```

Para mais informações sobre grupo de usuários do Amazon Cognito, consulte [Controlar o acesso a uma API REST usando um grupo de usuários do Amazon Cognito como autorizador](#) no Guia do desenvolvedor do API Gateway.

Exemplo de chave de API para AWS SAM

Você pode controlar o acesso às suas APIs exigindo chaves de API em seu AWS SAM modelo. Para fazer isso, você usa o tipo de [ApiAuth](#) dados.

Veja a seguir um exemplo de seção AWS SAM de modelo para chaves de 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
```

Para obter mais informações, consulte [Criação e uso de planos de uso com chaves de API](#) no Guia do desenvolvedor do API Gateway.

Exemplo de política de recursos para AWS SAM

Você pode controlar o acesso às suas APIs anexando uma política de recursos ao seu modelo AWS SAM. Para fazer isso, você usa o tipo de [ApiAuth](#) dados.

Veja a seguir um exemplo AWS SAM de modelo para uma API privada. Uma API privada deve ter uma política de recursos a ser implantada.

```
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

```

Para obter mais informações sobre políticas de recursos, consulte Como [controlar o acesso a uma API com as políticas de recursos do API Gateway](#) no Guia do desenvolvedor do API Gateway. Para obter mais informações, consulte [Como criar uma API privada no Amazon API Gateway](#) no Guia do desenvolvedor do API Gateway.

OAuth 2.0/JWT exemplo de autorizador para AWS SAM

[Você pode controlar o acesso às suas APIs usando JWTs como parte das estruturas OpenID Connect \(OIDC\) e OAuth 2.0.](#) Para fazer isso, você usa o tipo de [HttpApiAuth](#) dados.

Veja a seguir um exemplo de seção AWS SAM de modelo para um OAuth 2.0/JWT autorizador:

```

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
```

Para obter mais informações sobre o OAuth 2.0/JWT autorizadores, consulte [Controle do acesso às APIs HTTP com autorizadores do JWT no Guia](#) do desenvolvedor do API Gateway.

Exemplo de resposta personalizada para AWS SAM

Você pode personalizar algumas respostas de erro do API Gateway definindo cabeçalhos de resposta em seu modelo AWS SAM. Para fazer isso, use o tipo de dados [Gateway Response Object](#).

Veja a seguir um exemplo AWS SAM de modelo que cria uma resposta personalizada para o DEFAULT_5XX erro.

```
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
```

Para obter mais informações sobre as respostas do API Gateway, consulte [Respostas do gateway no API Gateway](#) no Guia do desenvolvedor do API Gateway.

Aumente a eficiência usando camadas Lambda com AWS SAM

Usando AWS SAM, você pode incluir camadas em seus aplicativos sem servidor. AWS Lambda camadas permitem que você extraia código de uma função de evento Lambda em uma camada Lambda que pode ser usada em várias funções Lambda. Fazer isso permite a redução do tamanho dos pacotes de implantação, a separação da lógica da função de núcleo das dependências e o compartilhamento de dependências em várias funções. Para obter informações sobre camadas, consulte [Camadas do Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Este tópico fornece informações sobre o seguinte:

- Incluindo camadas em seu aplicativo
- Como as camadas são armazenadas em cache localmente

Para obter informações sobre como construir camadas personalizadas, consulte [Construindo camadas Lambda em AWS SAM](#).

Incluindo camadas em seu aplicativo

Para incluir camadas em seu aplicativo, use a propriedade `Layers` do tipo de recurso [AWS::Serverless::Função](#).

Veja a seguir um exemplo AWS SAM de modelo com uma função Lambda que inclui uma camada:

```
ServerlessFunction:
```



```
Type: AWS::Serverless::Function
Properties:
  CodeUri: .
  Handler: my_handler
  Runtime: Python3.7
  Layers:
    - <LayerVersion ARN>
```

Como as camadas são armazenadas em cache localmente

Quando você invoca sua função usando um dos `sam local` comandos, o pacote de camadas da sua função é baixado e armazenado em cache no seu host local.

A tabela a seguir mostra os locais padrão dos diretórios de cache de diferentes sistemas operacionais.

SO	Local
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
Unix	~/ .aws-sam/layers-pkg

Depois que o pacote é armazenado em cache, as camadas são AWS SAMCLI sobrepostas em uma imagem do Docker usada para invocar sua função. Ele AWS SAMCLI gera os nomes das imagens que ele cria, bem como as LayerVersions que são mantidas no cache. Você pode encontrar mais detalhes sobre o esquema nas seções a seguir.

Para inspecionar as camadas sobrepostas, execute o seguinte comando para iniciar uma sessão `bash` na imagem que você deseja inspecionar:

```
docker run -it --entrypoint=/bin/bash samcli/lambda:<Tag following the schema outlined
in Docker Image Tag Schema> -i
```

Esquema de nome do diretório de cache de camadas

Dado `LayerVersionArn` que está definido em seu modelo, o AWS SAMCLI extrai a versão `LayerName` e do ARN. Ele cria um diretório para colocar o conteúdo da camada em chamado `LayerName-Version-<first 10 characters of sha256 of ARN>`.

Exemplo:

```
ARN = arn:aws:lambda:us-west-2:111111111111:layer:myLayer:1
Directory name = myLayer-1-926eeb5ff1
```

Esquema de tags do Docker Images

Para calcular o hash de camadas exclusivo, combine todos os nomes de camadas exclusivos com um delimitador de '-', use o hash SHA256 e, em seguida, use os primeiros 10 caracteres.

Exemplo:

```
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
```

Os nomes exclusivos são computados da mesma forma que o esquema de nomes do 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
```

Para calcular o hash de camadas exclusivo, combine todos os nomes de camadas exclusivos com um delimitador de '-', use o hash sha256 e, em seguida, use os primeiros 25 caracteres:

```
myLayer-1-926eeb5ff1-mySecondLayer-1-6bc1022bdf = 2dd7ac5ffb30d515926aef
```

Em seguida, combine esse valor com o tempo de execução e a arquitetura da função, com um delimitador de '-':

```
python3.7-x86_64-2dd7ac5ffb30d515926aefffd
```

Reutilize código e recursos usando aplicativos aninhados no AWS SAM

Um aplicativo sem servidor pode incluir um ou mais aplicativos aninhados. Uma aplicação aninhada faz parte de um aplicação maior e pode ser empacotada e implantada como um artefato independente ou como um componente de uma aplicação maior. As aplicações aninhadas permitem que você transforme códigos usados com frequência em sua própria aplicação, que pode ser reutilizado em uma aplicação sem servidor maior ou em várias aplicações sem servidor.

À medida que as arquiteturas sem servidor crescem, normalmente surgem padrões comuns nos quais os mesmos componentes são definidos em vários modelos de aplicação. Os aplicativos aninhados permitem que você reutilize códigos, funcionalidades, recursos e configurações comuns em AWS SAM modelos separados, permitindo que você mantenha apenas o código de uma única fonte. Isso reduz código e configurações duplicadas. Além disso, essa abordagem modular simplifica o desenvolvimento, aprimora a organização do código e facilita a consistência em aplicações sem servidor. Com aplicativos aninhados, você pode se concentrar mais na lógica de negócios exclusiva do seu aplicativo.

Para definir um aplicativo aninhado em seu aplicativo sem servidor, use o tipo de recurso [AWS::Serverless::Application](#).

Você pode definir aplicativos aninhados a partir das duas fontes a seguir:

- Um AWS Serverless Application Repository aplicativo — Você pode definir aplicativos aninhados usando aplicativos que estão disponíveis para sua conta na AWS Serverless Application Repository. Eles podem ser aplicativos privados em sua conta, aplicativos que são compartilhados de forma privada com sua conta ou aplicativos que são compartilhados publicamente no AWS Serverless Application Repository. Para obter mais informações sobre os diferentes níveis de permissões de implantação, consulte [Permissões de implantação de aplicativos](#) e [aplicativos de publicação](#) no AWS Serverless Application Repository Guia do desenvolvedor.
- Um aplicativo local — você pode definir aplicativos aninhados usando aplicativos armazenados em seu sistema de arquivos local.

Consulte as seções a seguir para obter detalhes sobre como usar AWS SAM para definir esses dois tipos de aplicativos aninhados em seu aplicativo sem servidor.

 Note

O número máximo de aplicativos que podem ser aninhados em um aplicativo sem servidor é 200.

O número máximo de parâmetros que um aplicativo aninhado pode ter é 60.

Definindo um aplicativo aninhado a partir do AWS Serverless Application Repository

Você pode definir aplicativos aninhados usando aplicativos que estão disponíveis no AWS Serverless Application Repository. Você também pode armazenar e distribuir aplicativos que contêm aplicativos aninhados usando AWS Serverless Application Repository. Para revisar os detalhes de um aplicativo aninhado no AWS Serverless Application Repository, você pode usar o AWS SDK AWS CLI, o ou o console Lambda.

Para definir um aplicativo hospedado no AWS SAM modelo do seu aplicativo sem servidor, use o botão Copiar como recurso do SAM na página de detalhes de cada AWS Serverless Application Repository aplicativo. AWS Serverless Application Repository Para isso, siga estas etapas:

1. Certifique-se de estar conectado ao Console de gerenciamento da AWS.
2. Encontre o aplicativo no qual você deseja aninhar AWS Serverless Application Repository usando as etapas na seção [Navegação, Pesquisa e Implantação de Aplicativos](#) do Guia do AWS Serverless Application Repository Desenvolvedor.
3. Escolha o botão Copiar como recurso do SAM. A seção de modelo do SAM para o aplicativo que você está visualizando agora está na sua área de transferência.
4. Cole a seção do modelo do SAM na seção Resources: do arquivo de modelo do SAM do aplicativo que você deseja aninhar nesse aplicativo.

Veja a seguir um exemplo de seção de modelo de SAM para um aplicativo aninhado hospedado no 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 não houver configurações de parâmetros obrigatórias, você poderá omitir a seção `Parameters` do modelo.

Important

Os aplicativos que contêm aplicativos aninhados hospedados no AWS Serverless Application Repository herdam as restrições de compartilhamento dos aplicativos aninhados.

Por exemplo, suponha que um aplicativo seja compartilhado publicamente, mas contenha um aplicativo aninhado que só é compartilhado de forma privada com a AWS conta que criou o aplicativo principal. Nesse caso, se sua AWS conta não tiver permissão para implantar o aplicativo aninhado, você não poderá implantar o aplicativo principal. Para obter mais informações sobre permissões para implantar aplicativos, consulte [Permissões de implantação de aplicativos](#) e [publicação de aplicativos](#) no AWS Serverless Application Repository Guia do desenvolvedor.

Definindo um aplicativo aninhado a partir do sistema de arquivos local

Você pode definir aplicativos aninhados usando aplicativos armazenados no sistema de arquivos local. Você faz isso especificando o caminho para o arquivo AWS SAM de modelo que está armazenado no sistema de arquivos local.

Veja a seguir um exemplo de seção de modelo de SAM para um aplicativo local aninhado:

```

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 não houver configurações de parâmetros, você poderá omitir a seção `Parameters:` do modelo.

Implantar aplicativos aninhados

Você pode implantar seu aplicativo aninhado usando o comando `sam deploy` do AWS SAM CLI. Consulte mais detalhes em [Implante seu aplicativo e seus recursos com AWS SAM](#).

Note

Quando você implanta uma aplicação que contém aplicações aninhadas, deve reconhecer que ela contém aplicações aninhadas. Você faz isso passando `CAPABILITY_AUTO_EXPAND` para a [CreateCloudFormationChangeSet API](#) ou usando o `aws serverlessrepo create-cloud-formation-change-set` AWS CLI comando.

Para obter mais informações sobre como reconhecer aplicativos aninhados, consulte [Reconhecer funções, políticas de recursos e aplicativos aninhados do perfil do IAM ao implantar aplicativos](#) no Guia do desenvolvedor do AWS Serverless Application Repository .

Gerencie eventos baseados em tempo com o EventBridge Scheduler em AWS SAM

O conteúdo deste tópico fornece detalhes sobre o que é o Amazon EventBridge Scheduler, quais AWS SAM ofertas de suporte, como você pode criar eventos do Scheduler e exemplos que você pode consultar ao criar eventos do Scheduler.

O que é o Amazon EventBridge Scheduler?

Use o EventBridge Scheduler para agendar eventos em seus AWS SAM modelos. O Amazon EventBridge Scheduler é um serviço de agendamento que permite criar, iniciar e gerenciar dezenas de milhões de eventos e tarefas em todos os serviços. AWS Esse serviço é particularmente útil para eventos relacionados a tempo. Você pode usá-lo para agendar eventos e invocações recorrentes baseadas em tempo. Ele também oferece suporte a eventos únicos, bem como a expressões de taxa e cron com horários de início e de término.

Para saber mais sobre o Amazon EventBridge Scheduler, consulte [O que é o Amazon EventBridge Scheduler?](#) no Guia do usuário do EventBridge Scheduler.

Tópicos

- [EventBridge Suporte ao agendador em AWS SAM](#)
- [Criação de eventos do EventBridge Scheduler em AWS SAM](#)
- [Exemplos](#)
- [Saiba mais](#)

EventBridge Suporte ao agendador em AWS SAM

A especificação do modelo AWS Serverless Application Model (AWS SAM) fornece uma sintaxe simples e abreviada que você pode usar para agendar eventos com o EventBridge Scheduler for e. AWS Lambda AWS Step Functions

Criação de eventos do EventBridge Scheduler em AWS SAM

Defina a `ScheduleV2` propriedade como o tipo de evento em seu AWS SAM modelo para definir seu evento do EventBridge Scheduler. Essa propriedade oferece suporte aos tipos de recurso `AWS::Serverless::Function` e `AWS::Serverless::StateMachine`.

```
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 O agendamento de eventos do Scheduler também oferece suporte a filas de cartas mortas (DLQ) para eventos não processados. Para obter mais informações sobre filas de mensagens mortas, consulte [Configurando uma fila de mensagens mortas para EventBridge o Scheduler no Guia do usuário do Scheduler](#). EventBridge

Quando um ARN de DLQ é especificado AWS SAM , configura permissões para que o agendador envie mensagens para o DLQ. Quando um ARN DLQ não for especificado AWS SAM , criará o recurso DLQ.

Exemplos

Exemplo básico de definição de um evento do EventBridge Scheduler com 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"}'
```

Saiba mais

Para saber mais sobre como definir a propriedade `ScheduleV2 EventBridge Scheduler`, consulte:

- [ScheduleV2](#) para `AWS::Serverless::Function`.
- [ScheduleV2](#) para `AWS::Serverless::StateMachine`.

Orquestrando AWS SAM recursos com AWS Step Functions

Você pode usar [AWS Step Functions](#) para orquestrar funções de AWS Lambda eventos e outros AWS recursos para formar fluxos de trabalho complexos e robustos. Step Functions para informar ao seu aplicativo quando e sob quais condições seus AWS recursos, como AWS Lambda funções, são usados. Isso simplifica o processo da formação de fluxos de trabalho complexos e robustos. Usando [AWS::Serverless::StateMachine](#), você define as etapas individuais do fluxo de trabalho, associa recursos em cada etapa e, em seguida, sequencia essas etapas em conjunto. Você também adiciona

transições e condições onde elas são necessárias. Isso simplifica o processo de criar um fluxo de trabalho complexo e robusto.

Note

Para gerenciar AWS SAM modelos que contêm máquinas de estado do Step Functions, você deve usar a versão 0.52.0 ou posterior do AWS SAMCLI. Para verificar qual versão você tem, execute o comando `sam --version`.

Step Functions é baseado nos conceitos de [tarefas](#) e [máquinas de estado](#). Você define máquinas de estado usando a JSON-based [Amazon States Language](#). O [console do Step Functions](#) apresenta uma visualização gráfica da estrutura da sua máquina de estado para que você possa verificar visualmente a lógica da sua máquina de estado e monitorar as execuções.

Com o suporte para Step Functions em AWS Serverless Application Model (AWS SAM), você pode fazer o seguinte:

- Defina máquinas de estado, diretamente em um AWS SAM modelo ou em um arquivo separado
- Crie funções de execução de máquinas de estado por meio AWS SAM de modelos de políticas, políticas embutidas ou políticas gerenciadas
- Acione execuções de máquinas de estado com o API Gateway ou EventBridge eventos da Amazon, em uma programação dentro de um AWS SAM modelo ou chamando APIs diretamente
- Use os [modelos AWS SAM de política](#) disponíveis para padrões comuns de desenvolvimento de Step Functions.

Exemplo

O trecho de exemplo a seguir de um arquivo de AWS SAM modelo define uma máquina de estado Step Functions em um arquivo de definição. Observe que o `my_state_machine.asl.json` arquivo deve ser escrito em [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
  ...
```

Para baixar um AWS SAM aplicativo de amostra que inclui uma máquina de estado Step Functions, consulte [Create a Step Functions State Machine usando AWS SAM](#) no AWS Step Functions Developer Guide.

Mais informações

Para saber mais sobre Step Functions e como usá-lo com AWS SAM, veja o seguinte:

- [Com o AWS Step Functions funciona](#)
- [AWS Step Functions and AWS Serverless Application Model](#)
- [Tutorial: Crie uma máquina de estado do Step Functions usando AWS SAM](#)
- [AWS SAM Especificação: AWS::Serverless:: StateMachine](#)

Configure a assinatura de código para seu AWS SAM aplicação

Para garantir que somente código confiável seja implantado, você pode usar AWS SAM para habilitar a assinatura de código com seus aplicativos sem servidor. Assinar seu código ajuda a garantir que o código não tenha sido alterado desde a assinatura e que somente pacotes de código assinados de editores confiáveis sejam executados em suas Funções de Eventos Lambda. Isso ajuda a liberar as organizações da carga de criar componentes do gatekeeper em seus pipelines de implantação.

Para obter mais informações sobre assinatura de código, consulte [Configurar assinatura de código das funções do Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Antes de configurar a assinatura de código para seu aplicativo sem servidor, você deve criar um perfil de assinatura usando o AWS Signer. Você usa esse perfil de assinatura para as seguintes tarefas:

1. Criação de uma configuração de assinatura de código — Declare um [AWS::Lambda::CodeSigningConfig](#) recurso para especificar os perfis de assinatura de editores confiáveis e definir a ação política para verificações de validação. Você pode declarar esse objeto no mesmo AWS SAM modelo da sua função sem servidor, em um AWS SAM modelo diferente ou em um modelo CloudFormation. Em seguida, você habilita a assinatura de código para uma função sem servidor especificando a propriedade

[CodeSigningConfigArn](#) da função com o nome de recurso da Amazon (ARN) de um recurso [AWS::Lambda::CodeSigningConfig](#).

2. Assinando seu código — Use o comando [sam package](#) ou [sam deploy](#) com a opção `--signing-profiles`.

Note

Para assinar com sucesso seu código com os comandos `sam package` ou `sam deploy`, o versionamento deve estar habilitado para o bucket do Amazon S3 que você usa com esses comandos. Se você estiver usando o Amazon S3 Bucket AWS SAM criado para você, o controle de versão será ativado automaticamente. Para obter mais informações sobre o versionamento de buckets do Amazon S3 e instruções para habilitar o versionamento em um bucket do Amazon S3 fornecido por você, consulte [Uso do versionamento em buckets do Amazon S3](#) no Guia do usuário do Amazon Simple Storage Service.

Quando você implanta um aplicativo sem servidor, o Lambda executa verificações de validação em todas as funções para as quais você habilitou a assinatura de código. O Lambda também realiza verificações de validação em todas as camadas das quais essas funções dependem. Para obter mais informações sobre as verificações de validação do Lambda, consulte [Validação de assinatura](#) no Guia do AWS Lambda desenvolvedor.

Exemplo

Criar um perfil de assinatura

Para criar um perfil de assinatura, execute o seguinte comando:

```
aws signer put-signing-profile --platform-id "AWSLambda-SHA384-ECDSA" --profile-name MySigningProfile
```

Se o comando anterior é bem-sucedido, o ARN do perfil de assinatura é retornado. Por exemplo:

```
{
  "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"
```

```
}
```

O `profileVersionArn` campo contém o ARN a ser usado ao criar a configuração de assinatura de código.

Criar uma configuração de assinatura de código e habilitar a assinatura de código para uma função

O AWS SAM modelo de exemplo a seguir declara um

[`AWS::Lambda::CodeSigningConfig`](#) recurso e permite a assinatura de código para uma função Lambda. Neste exemplo, há um perfil confiável e as implantações são rejeitadas se as verificações de assinatura falharem.

```
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"
```

Assinando seu código

Você pode assinar seu código ao empacotar ou implantar seu aplicativo. Especifique a opção `--signing-profiles` com o comando `sam package` ou `sam deploy`, conforme mostrado nos seguintes exemplos de comandos.

Assinando seu código de função ao empacotar seu aplicativo:

```
sam package --signing-profiles HelloWorld=MySigningProfile --s3-bucket amzn-s3-demo-bucket --output-template-file packaged.yaml
```

Assinar seu código de função e uma camada da qual sua função depende ao empacotar seu aplicativo:

```
sam package --signing-profiles HelloWorld=MySigningProfile MyLayer=MySigningProfile --  
s3-bucket amzn-s3-demo-bucket --output-template-file packaged.yaml
```

Assinando seu código de função e uma camada e, em seguida, executando uma implantação:

```
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

Para assinar com sucesso seu código com os comandos `sam package` ou `sam deploy`, o versionamento deve estar habilitado para o bucket do Amazon S3 que você usa com esses comandos. Se você estiver usando o Amazon S3 Bucket AWS SAM criado para você, o controle de versão será ativado automaticamente. Para obter mais informações sobre o versionamento de buckets do Amazon S3 e instruções para habilitar o versionamento em um bucket do Amazon S3 fornecido por você, consulte [Uso do versionamento em buckets do Amazon S3](#) no Guia do usuário do Amazon Simple Storage Service.

Fornecendo perfis de assinatura com **sam deploy --guided**

Quando você executa o `sam deploy --guided` comando com um aplicativo sem servidor configurado com assinatura de código, AWS SAM solicita que você forneça o perfil de assinatura a ser usado na assinatura de código. Para obter mais informações sobre prompts `sam deploy --guided`, consulte [sam deploy](#) na Referência de comandos de AWS SAMCLI.

Validar arquivos AWS SAM de modelo

Valide seus modelos com [sam validate](#). Atualmente, esse comando valida se o modelo fornecido é JSON/YAML válido. Como acontece com a maioria dos comandos, AWS SAM CLI, ele procura um arquivo `template.[yaml|yml]` no seu diretório de trabalho atual por padrão. Você pode especificar um modelo diferente `file/location` com a `--template` opção `-t` ou.

Exemplo:

```
$ sam validate  
<path-to-template>/template.yaml is a valid SAM Template
```

Note

O `sam validate` comando exige que AWS as credenciais sejam configuradas. Para obter mais informações, consulte [Configurando o AWS SAM CLI](#).

Crie seu aplicativo com AWS SAM

Depois de adicionar sua infraestrutura como código (IaC) ao seu AWS SAM modelo, você estará pronto para começar a criar seu aplicativo usando o `sam build` comando. Esse comando cria artefatos de construção a partir dos arquivos no diretório do projeto do aplicativo (ou seja, seu arquivo de AWS SAM modelo, código do aplicativo e quaisquer arquivos e dependências específicos do idioma aplicável). Esses artefatos de construção preparam seu aplicativo sem servidor para as etapas posteriores do desenvolvimento do seu aplicativo, como testes locais e implantação na nuvem. AWS Tanto o teste quanto a implantação usam artefatos de compilação como entradas.

Você pode usar `sam build` para criar toda a aplicação sem servidor. Além disso, você pode criar compilações personalizadas, como aquelas com funções, camadas ou runtimes personalizados específicos. Para ler mais sobre como e por que usar `sam build`, consulte os tópicos desta seção. Para obter uma introdução sobre como usar o comando `sam build`, consulte [Introdução à construção com AWS SAM](#).

Tópicos

- [Introdução à construção com AWS SAM](#)
- [Compilação padrão com AWS SAM](#)
- [Personalize construções com AWS SAM](#)

Introdução à construção com AWS SAM

Use o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam build` para preparar seu aplicativo sem servidor para as etapas subsequentes do fluxo de trabalho de desenvolvimento, como testes locais ou implantação no. Nuvem AWS Esse comando cria um

diretório `.aws-sam` que estrutura seu aplicativo em um formato e localização que o `sam local` e o `sam deploy` exigem.

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#).
- Para obter uma lista de opções de comando `sam build`, consulte [sam build](#).
- Para obter um exemplo de uso do `sam build` durante um fluxo de trabalho de desenvolvimento típico, consulte [Etapa 2: crie seu aplicativo](#).

Note

O uso de `sam build` exige que você comece com os componentes básicos de um aplicativo com tecnologia sem servidor em sua máquina de desenvolvimento. Isso inclui um AWS SAM modelo, código de função de AWS Lambda evento e quaisquer arquivos e dependências específicos do idioma. Para saber mais, consulte [Crie seu aplicativo em AWS SAM](#).

Tópicos

- [Construindo aplicativos com Sam Build](#)
- [Teste e implantação locais](#)
- [Práticas recomendadas](#)
- [Opções para sam build](#)
- [Solução de problemas](#)
- [Exemplos](#)
- [Saiba mais](#)

Construindo aplicativos com Sam Build

Antes de usar o `sam build`, considere a configuração do seguinte:

1. Funções e camadas do Lambda — O `sam build` comando pode construir funções e camadas do Lambda. Para saber mais sobre as camadas do Lambda, consulte [Construindo camadas Lambda em AWS SAM](#).
2. Tempo de execução do Lambda – O tempo de execução fornece um ambiente específico de linguagem que executa sua função no ambiente de execução quando invocado. Você pode configurar runtimes nativos e personalizados.

- a. Tempo de execução nativo — Crie suas funções do Lambda em um tempo de execução do Lambda compatível e construa suas funções para usar um tempo de execução do Lambda nativo no Nuvem AWS.
 - b. Tempo de execução personalizado — Crie suas funções do Lambda usando qualquer linguagem de programação e construa seu tempo de execução usando um processo personalizado definido em um construtor makefile ou em um criador terceirizado, como esbuild. Para saber mais, consulte [Criando funções de eventos Lambda com tempos de execução personalizados em AWS SAM](#).
3. Tipo de pacote Lambda — As funções do Lambda podem ser empacotadas nos seguintes tipos de pacotes de implantação do Lambda:
 - a. arquivo .zip – Contém o código do seu aplicativo e suas dependências.
 - b. Imagem de contêiner – Contém o sistema operacional de base, o tempo de execução, as extensões do Lambda, o código do seu aplicativo e suas dependências.

Essas configurações do aplicativo podem ser definidas ao inicializar um aplicativo usando o `sam init`.

- Para saber mais sobre o uso do `sam init`, consulte [Crie seu aplicativo em AWS SAM](#).
- Para saber mais sobre como definir essas configurações em seu aplicativo, consulte [Compilação padrão com AWS SAM](#).

Para construir um aplicativo

1. `cd` até a raiz do seu projeto. Esse é o mesmo local do seu AWS SAM modelo.

```
$ cd sam-app
```

2. Execute o seguinte:

```
sam-app $ sam build <arguments> <options>
```

Note

Uma opção comumente usada é o `--use-container`. Para saber mais, consulte [Construindo uma função de evento Lambda dentro de um contêiner fornecido](#).

A seguir, veja um exemplo de saída do AWS SAM CLI:

```
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. O AWS SAMCLI cria um diretório de compilação `.aws-sam`. Veja um exemplo a seguir:

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

Dependendo de como seu aplicativo está configurado, o AWS SAMCLI faz o seguinte:

1. Faz o download, instala e organiza as dependências no diretório `.aws-sam/build`.

2. Prepara seu código Lambda. Isso pode incluir compilar seu código, criar binários executáveis e criar imagens de contêiner.
3. Copia artefatos de construção para o diretório `.aws-sam`. O formato variará de acordo com o tipo de pacote do aplicativo.
 - a. Para tipos de pacotes.zip, os artefatos ainda não foram compactados para que possam ser usados para testes locais. O AWS SAMCLI fecha seu aplicativo ao usar o `sam deploy`.
 - b. Para tipos de pacote de imagem de contêiner, uma imagem de contêiner é criada localmente e referenciada no arquivo `.aws-sam/build.toml`.
4. Copia o AWS SAM modelo para o `.aws-sam` diretório e o modifica com novos caminhos de arquivo quando necessário.

A seguir estão os principais componentes que compõem seus artefatos de construção no diretório `.aws-sam`:

- O diretório de construção — contém suas funções e camadas do Lambda estruturadas independentemente umas das outras. Isso resulta em uma estrutura exclusiva para cada função ou camada no diretório `.aws-sam/build`.
- O AWS SAM modelo — Modificado com valores atualizados com base nas alterações durante o processo de criação.
- O arquivo `build.toml` — Um arquivo de configuração que contém as configurações de compilação usadas pelo AWS SAMCLI

Teste e implantação locais

Ao realizar testes locais com o `sam local` ou implantar com o `sam deploy`, o AWS SAMCLI faz o seguinte:

1. Primeiro, ele verifica se existe um `.aws-sam` diretório e se um AWS SAM modelo está localizado dentro desse diretório. Se essas condições forem atendidas, o AWS SAMCLI considerará como o diretório raiz do seu aplicativo.
2. Se essas condições não forem atendidas, o AWS SAMCLI considera o local original do seu AWS SAM modelo como o diretório raiz do seu aplicativo.

Ao desenvolver, se forem feitas alterações nos arquivos originais do aplicativo, execute `sam build` para atualizar o diretório `.aws-sam` antes de testar localmente.

Práticas recomendadas

- Não edite nenhum código no diretório `.aws-sam/build`. Em vez disso, atualize o código-fonte original na pasta do projeto e execute `sam build` para atualizar o diretório `.aws-sam/build`.
- Ao modificar seus arquivos originais, execute `sam build` para atualizar o diretório `.aws-sam/build`.
- Talvez você queira que o AWS SAMCLI referencie o diretório raiz original do seu projeto em vez do diretório `.aws-sam`, como ao desenvolver e testar com o `sam local`. Exclua o `.aws-sam` diretório ou o AWS SAM modelo no `.aws-sam` diretório para que ele AWS SAMCLI reconheça o diretório original do projeto como o diretório raiz do projeto. Quando estiver pronto, execute `sam build` novamente para criar o diretório `.aws-sam`.
- Quando você executa o `sam build`, o diretório `.aws-sam/build` é sobrescrito a cada vez. O diretório `.aws-sam` não. Se você quiser armazenar arquivos, como registros, armazene-os no `.aws-sam` para evitar que sejam sobrescritos.

Opções para sam build

Criar um único recurso

Forneça a ID lógica do recurso para criar somente esse recurso. Este é um exemplo:

```
$ sam build HelloWorldFunction
```

Para criar um recurso de uma pilha ou aplicativo aninhado, forneça o ID lógico do aplicativo ou da pilha junto com o ID lógico do recurso usando o formato `<stack-logical-id>/<resource-logical-id>`:

```
$ sam build MyNestedStack/MyFunction
```

Construindo uma função de evento Lambda dentro de um contêiner fornecido

A opção `--use-container` baixa uma imagem de contêiner e a usa para criar suas funções do Lambda. O contêiner local é então referenciado em seu arquivo `.aws-sam/build.toml`.

Esta opção requer Docker instalação. Para obter instruções, consulte [Instalação do Docker](#).

A seguir está um exemplo deste comando:

```
$ sam build --use-container
```

Você pode especificar a imagem do contêiner a ser usada com a opção `--build-image`. Veja um exemplo a seguir:

```
$ sam build --use-container --build-image amazon/aws-sam-cli-build-image-nodejs20.x
```

Para especificar a imagem do contêiner a ser usada para uma única função, forneça o ID lógico da função. Este é um exemplo:

```
$ sam build --use-container --build-image Function1=amazon/aws-sam-cli-build-image-python3.12
```

Criando imagens de contêineres com BuildKit

Ao criar funções do Lambda com o tipo de Image pacote, você pode usar BuildKit como mecanismo de compilação passando a `--use-buildkit` opção. BuildKit fornece melhor desempenho de compilação, melhor armazenamento em cache e melhor saída de compilação em comparação com o mecanismo de Docker compilação antigo.

Esta opção requer ter o Docker ou Finch CLIs instalado e em seu PATH.

Você pode combinar `--use-buildkit` com outras opções de construção. Este é um exemplo:

```
$ sam build --use-buildkit --parallel
```

Note

A `--use-buildkit` sinalização é somente para compilações de imagens de contêineres. Isso não terá nenhum efeito nas funções zip que estão sendo criadas `--use-container`.

Passe variáveis de ambiente para o contêiner de build

Use o `--container-env-var` para passar variáveis de ambiente para o contêiner de build. Veja um exemplo a seguir:

```
$ sam build --use-container --container-env-var Function1.GITHUB_TOKEN=<token1> --  
container-env-var GLOBAL_ENV_VAR=<global-token>
```

Para passar variáveis de ambiente de um arquivo, use a opção `--container-env-var-file`. Veja um exemplo a seguir:

```
$ sam build --use-container --container-env-var-file <env.json>
```

Exemplo do arquivo `env.json`:

```
{
  "MyFunction1": {
    "GITHUB_TOKEN": "TOKEN1"
  },
  "MyFunction2": {
    "GITHUB_TOKEN": "TOKEN2"
  }
}
```

Acelere a criação de aplicativos que contêm várias funções

Quando você executa `sam build` em um aplicativo com várias funções, o AWS SAMCLI cria uma função de cada vez. Para acelerar o processo de criação, use a opção `--parallel`. Isso cria todas as suas funções e camadas ao mesmo tempo.

A seguir está um exemplo deste comando:

```
$ sam build --parallel
```

Acelere os tempos de compilação criando seu projeto na pasta de origem.

Para runtimes e métodos de compilação compatíveis, você pode usar a opção `--build-in-source` de criar seu projeto diretamente na pasta de origem. Por padrão, eles AWS SAM CLI são compilados em um diretório temporário, o que envolve a cópia do código-fonte e dos arquivos do projeto. Com `--build-in-source`, AWS SAM CLI as compilações diretamente na sua pasta de origem, o que acelera o processo de compilação ao eliminar a necessidade de copiar arquivos para um diretório temporário.

Para ver uma lista de runtimes compatíveis, consulte [--build-in-source](#).

Solução de problemas

Para solucionar o problema do AWS SAMCLI, consulte [AWS SAM CLI solução de problemas](#).

Exemplos

Criação de um aplicativo que usa um tempo de execução nativo e um tipo de pacote.zip

Para este exemplo, consulte [Tutorial: implante um aplicativo Hello World com AWS SAM](#).

Criação de um aplicativo que usa um tempo de execução nativo e um tipo de pacote de imagem

Primeiro, executamos `sam init` para inicializar um novo aplicativo. Durante o fluxo interativo, selecionamos o tipo de pacote Image. Veja um exemplo a seguir:

```
$ 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

...

O AWS SAMCLI inicializa um aplicativo e cria o seguinte diretório de projeto:

```
sam-app
### README.md
### events
#   ### event.json
### hello-world
#   ### Dockerfile
```



```
#   ### app.mjs
#   ### package.json
#   ### tests
#       ### unit
#           ### test-handler.mjs
### samconfig.toml
### template.yaml
```

Em seguida, executamos `sam build` para criar nosso aplicativo:

```

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

```

Como construir um aplicativo que inclua uma linguagem de programação compilada

Neste exemplo, criamos um aplicativo que contém uma função do Lambda usando o tempo de execução Go.

Primeiro, inicializamos um novo aplicativo usando `sam init` e configuramos nosso aplicativo para usar 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

...

O AWS SAMCLI inicializa o aplicativo. A seguir está um exemplo da estrutura de diretórios do aplicativo:

```
sam-app
### Makefile
### README.md
### events
#   ### event.json
### hello-world
#   ### go.mod
#   ### go.sum
#   ### main.go
#   ### main_test.go
### samconfig.toml
### template.yaml
```

Nós referenciamos o arquivo `README.md` para os requisitos desse aplicativo.

```
...
## 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)
...
```

Em seguida, executamos `sam local invoke` para testar nossa função. Esse comando comete um erro, pois Go não está instalado em nossa máquina local:

```
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"}%
```

A seguir `sam build`, executamos para construir nosso aplicativo. Encontramos um erro porque Go não está instalado em sua máquina local:

```
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
```

Embora pudéssemos configurar nossa máquina local para criar adequadamente nossa função, usamos a opção `--use-container` com `sam build`. Ele AWS SAMCLI baixa uma imagem de contêiner, cria nossa função usando o nativo `GoModulesBuilder` e copia o binário resultante em nosso `.aws-sam/build/HelloWorldFunction` diretório.

```
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
```

A seguir está um exemplo do diretório `.aws-sam`:

```
.aws-sam
### build
```

```
#   ### HelloWorldFunction
#   #   ### hello-world
#   ### template.yaml
### build.toml
### cache
#   ### c860d011-4147-4010-addb-2eaa289f4d95
#       ### hello-world
### deps
```

Em seguida, executamos `sam local invoke`. Nossa função foi invocada com sucesso:

```
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"}%
```

Saiba mais

Para obter mais informações sobre como usar o comando `sam build`, consulte:

- [Learning AWS SAM: sam build](#) — Série Serverless Land “Learning AWS SAM” sobre. YouTube
- [Learning AWS SAM | sam build | E3](#) — Série “Learning AWS SAM” da Serverless Land sobre. YouTube
- [AWS SAM build: como ele fornece artefatos para implantação \(Sessões com SAM S2E8\) — Sessões](#) com séries ativadas. AWS SAM YouTube
- [AWS SAM compilações personalizadas: Como usar Makefiles para personalizar compilações no SAM \(S2E9\) — Sessões](#) com séries ativadas. AWS SAM YouTube

Compilação padrão com AWS SAM

Para criar seu aplicativo sem servidor, use o comando [sam build](#). Esse comando também reúne os artefatos de construção das dependências do seu aplicativo e os coloca no formato e no local adequados para as próximas etapas, como teste, empacotamento e implantação locais.

Você especifica as dependências do seu aplicativo em um arquivo de manifesto, como `requirements.txt` (Python) `package.json` ou `Node.js ()`, ou usando `Layers` a propriedade de um recurso de função. A `Layers` propriedade contém uma lista de recursos de [AWS Lambda camada](#) dos quais a função de evento Lambda depende.

O formato dos artefatos de construção do seu aplicativo depende da propriedade `PackageType` de cada função. As opções para essa propriedade são:

- **Zip** – Um arquivo `.zip` inclui o código da aplicação e as dependências dele. Se você empacotar seu código como um arquivo `.zip`, você deve especificar um runtime do Lambda para sua função.
- **Image** – Uma imagem de contêiner inclui o sistema operacional de base, o runtime, as extensões, o código da aplicação e dependências dele.

Para obter informações sobre os tipos de pacotes do Lambda, consulte [Pacotes de implantação do Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Tópicos

- [Criando um arquivo .zip](#)
- [Criar uma imagem de contêiner](#)
- [As variáveis de ambiente do contêiner](#)
- [Acelere os tempos de compilação criando seu projeto na pasta de origem.](#)
- [Exemplos](#)
- [Funções de construção fora do AWS SAM](#)

Criando um arquivo .zip

Para criar seu aplicativo sem servidor como um arquivo `.zip`, você deve declarar `PackageType` : `Zip` sua função sem servidor.

AWS SAM cria seu aplicativo para a [arquitetura](#) que você especifica. Se você não especificar uma arquitetura, AWS SAM usa `x86_64` por padrão.

Se sua função do Lambda depender de pacotes que tenham programas compilados de forma nativa, use o sinalizador `--use-container`. Essa sinalização compila localmente suas funções em um contêiner que se comporta como um ambiente Lambda, para que elas estejam no formato certo quando você as implanta na nuvem. AWS

Quando você usa a `--use-container` opção, por padrão, AWS SAM extrai a imagem do contêiner do [Amazon ECR](#) Public. Se quiser extrair uma imagem de contêiner de outro repositório ou de uma versão específica da AWS SAM CLI, você pode usar a opção e fornecer `--build-image` o URI de uma imagem de contêiner alternativa. Veja a seguir dois exemplos de comandos para criar aplicativos usando imagens de contêiner de uma versão específica da 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
```

Para obter exemplos adicionais de criação de um aplicativo de arquivamento de arquivamento de arquivamento de arquivamento, consulte a seção Exemplos mais adiante neste tópico.

Criar uma imagem de contêiner

Para criar seu aplicativo sem servidor como uma imagem de contêiner, declare `PackageType` : `Image` sua função sem servidor. Você também deve declarar o atributo do recurso `Metadata` com as seguintes entradas:

`Dockerfile`

O nome do Dockerfile associado à função do Lambda.

`DockerContext`

A localização do Dockerfile.

`DockerTag`

(Opcional) Uma tag a ser aplicada à imagem criada.

`DockerBuildArgs`

Crie argumentos para a construção.

 Important

A CLI do AWS SAM não oculta nem ofusca quaisquer informações incluídas nos argumentos `DockerBuildArgs`. É altamente recomendável não usar essa seção para armazenar informações confidenciais, como senhas ou segredos.

DockerImage

(Opcional) Um nome personalizado para a imagem criada do Docker. Se não for especificado, o ID do recurso lógico da função será usado.

Veja a seguir um exemplo de seção Metadata de atributo de recurso:

```
Metadata:
  Dockerfile: Dockerfile
  DockerContext: ./hello_world
  DockerTag: v1
  DockerImage: my-custom-image
```

Para baixar um exemplo de aplicação configurada com o tipo de pacote Image, consulte [Tutorial: implante um aplicativo Hello World com AWS SAM](#). No prompt perguntando qual tipo de pacote você deseja instalar, escolha Image.

 Note

Se você especificar uma imagem base de várias arquiteturas em seu Dockerfile, AWS SAM criará sua imagem de contêiner para a arquitetura da sua máquina host. Para criar para uma arquitetura diferente, especifique uma imagem base que use a arquitetura de destino específica.

Por padrão, ele AWS SAMCLI usa o mecanismo de Docker compilação legado ao criar imagens de contêiner. Para usar BuildKit um back-end de contêiner aprimorado que melhora o desempenho e lida com compilações mais complexas, passe a `--use-buildkit` sinalização com seu comando de compilação:

```
$ sam build --use-buildkit
```

O uso `--use-buildkit` requer a Finch CLI instalação do Docker ou.

As variáveis de ambiente do contêiner

Para fornecer um arquivo JSON que contenha variáveis de ambiente para o contêiner de compilação, use o argumento `--container-env-var-file` com o comando `sam build`. Você pode fornecer uma única variável de ambiente que se aplique a todos os recursos sem servidor ou variáveis de ambiente diferentes para cada recurso.

Formato

O formato para passar variáveis de ambiente para um contêiner de compilação depende de quantas variáveis de ambiente você fornece para seus recursos.

Para fornecer uma única variável de ambiente para todos os recursos, especifique um `Parameters` objeto como o seguinte:

```
{
  "Parameters": {
    "GITHUB_TOKEN": "TOKEN_GLOBAL"
  }
}
```

Para fornecer variáveis de ambiente diferentes para cada recurso, especifique objetos para cada recurso da seguinte forma:

```
{
  "MyFunction1": {
    "GITHUB_TOKEN": "TOKEN1"
  },
  "MyFunction2": {
    "GITHUB_TOKEN": "TOKEN2"
  }
}
```

Salve suas variáveis de ambiente como um arquivo, por exemplo, chamado `env.json`. O comando a seguir usa esse arquivo para passar suas variáveis de ambiente para o contêiner de compilação:

```
sam build --use-container --container-env-var-file env.json
```

Precedência

- As variáveis de ambiente que você fornece para recursos específicos têm precedência sobre a única variável de ambiente para todos os recursos.
- As variáveis de ambiente que você fornece na linha de comando têm precedência sobre as variáveis de ambiente em um arquivo.

Acelere os tempos de compilação criando seu projeto na pasta de origem.

Para runtimes e métodos de compilação compatíveis, você pode usar a opção `--build-in-source` de criar seu projeto diretamente na pasta de origem. Por padrão, eles AWS SAM CLI são compilados em um diretório temporário, o que envolve a cópia do código-fonte e dos arquivos do projeto. Com `--build-in-source`, AWS SAM CLI as compilações diretamente na sua pasta de origem, o que acelera o processo de compilação ao eliminar a necessidade de copiar arquivos para um diretório temporário.

Para ver uma lista de runtimes e métodos de compilação compatíveis, consulte [--build-in-source](#).

Exemplos

Exemplo 1: arquivo .zip

Os comandos `sam build` a seguir criam um arquivo .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
```

Exemplo 2: imagem do contêiner

O AWS SAM modelo a seguir é criado como uma imagem de contêiner:

```
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Properties:
      PackageType: Image
      ImageConfig:
        Command: ["app.lambda_handler"]
    Metadata:
      Dockerfile: Dockerfile
      DockerContext: ./hello_world
      DockerTag: v1
```

Veja a seguir um exemplo de 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"]
```

Exemplo 3: npm ci

Para Node.js aplicativos, você pode usar `npm ci` em vez de `npm install` instalar dependências. Para usar `npm ci`, especifique `UseNpmCi: True` debaixo de `BuildProperties` no atributo de recurso `Metadata` da sua função do Lambda. Para ser usado `npm ci`, seu aplicativo deve ter um arquivo `package-lock.json` ou `npm-shrinkwrap.json` presente na `CodeUri` para sua função do Lambda.

O exemplo a seguir é usado `npm ci` para instalar dependências quando você `sam build` executa:

```
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
```

Pacotes principais do Python

Para aplicações do Python, você pode preservar sua estrutura de pacotes durante o processo de criação para permitir importações absolutas. Para preservar a estrutura do pacote, especifique `ParentPackageMode` em `BuildProperties` no atributo de recurso `Metadata` de sua função do Lambda.

O exemplo a seguir preserva a estrutura do pacote `app` quando você executa `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
```

Com essa configuração, seu código pode usar importações absolutas, como `from app.utils import logger` em vez de importações relativas, como `from .utils import logger`.

Funções de construção fora do AWS SAM

Por padrão, quando você executa `sam build`, AWS SAM cria todos os seus recursos de função. Outras opções incluem:

- Crie todos os recursos de função fora de AWS SAM — Se você criar todos os seus recursos de função manualmente ou por meio de outra ferramenta, não `sam build` é necessário. Você pode pular `sam build` e passar para a próxima etapa do processo, como realizar testes locais ou implantar seu aplicativo.
- Crie alguns recursos de função fora de AWS SAM — Se você quiser AWS SAM criar alguns de seus recursos de função e ter outros recursos de função criados fora dele AWS SAM, você pode especificar isso em seu AWS SAM modelo.

Crie alguns recursos funcionais fora do AWS SAM

Para AWS SAM pular uma função durante o `sam build`, configure o seguinte em seu AWS SAM modelo:

1. Adicione a propriedade de `SkipBuild: True` metadados à função do.
2. Especifique o caminho para seus recursos de função criados.

Aqui está um exemplo, `TestFunction` configurado para ser ignorado. Seus recursos construídos estão localizados em `built-resources/TestFunction.zip`.

```
TestFunction:
  Type: AWS::Serverless::Function
  Properties:
    CodeUri: built-resources/TestFunction.zip
    Handler: TimeHandler::handleRequest
    Runtime: java11
  Metadata:
    SkipBuild: True
```

Agora, ao executar `sam build`, AWS SAM fará o seguinte:

1. AWS SAM ignorará as funções configuradas com `SkipBuild: True`.
2. AWS SAM criará todos os outros recursos da função e os armazenará em cache no diretório de `.aws-sam` compilação.
3. Para funções ignoradas, seu modelo no diretório de `.aws-sam` construção será atualizado automaticamente para referenciar o caminho especificado para seus recursos de função criados.

Aqui está um exemplo do modelo `TestFunction` em cache para o diretório de `.aws-sam` compilação:

```
TestFunction:
  Type: AWS::Serverless::Function
  Properties:
    CodeUri: ../../built-resources/TestFunction.zip
    Handler: TimeHandler::handleRequest
    Runtime: java11
  Metadata:
    SkipBuild: True
```

Personalize construções com AWS SAM

Você pode personalizar sua compilação para incluir funções de eventos Lambda específicas ou camadas Lambda. Uma função é um recurso que você pode invocar para executar o código no Lambda. Uma camada do Lambda permite que você extraia código de uma função do Lambda que pode ser reutilizado em várias funções do Lambda. Você pode optar por personalizar a criação com funções específicas do Lambda quando quiser se concentrar no desenvolvimento e na implantação de funções individuais sem servidor sem a complexidade de gerenciar dependências ou recursos compartilhados. Além disso, você pode optar por criar uma camada do Lambda para ajudar na redução do tamanho dos pacotes de implantação, na separação da lógica da função de núcleo das dependências e na permissão para compartilhar dependências em várias funções.

Os tópicos desta seção exploram algumas das diferentes maneiras pelas quais você pode criar funções Lambda com AWS SAM. Isso inclui a criação de funções do Lambda com os runtimes do cliente e a criação de camadas do Lambda. Os tempos de execução personalizados permitem que você instale e use uma linguagem não listada nos tempos de execução do Lambda no AWS Lambda Guia do desenvolvedor. Isso permite que você crie um ambiente de execução especializado para executar funções e aplicações sem servidor. Criar somente camadas do Lambda (em vez de toda a aplicação) pode trazer benefícios de algumas maneiras. Isso pode ajudar na redução do tamanho

dos pacotes de implantação, na separação da lógica da função de núcleo das dependências e na permissão para compartilhar dependências em várias funções.

Para obter mais informações, consulte [Conceitos do Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Tópicos

- [Criando funções de eventos Node.js Lambda com esbuild em AWS SAM](#)
- [Criando funções de eventos do.NET Lambda com compilação AOT nativa no AWS SAM](#)
- [Criando funções de eventos do Rust Lambda com Lambda de carga in AWS SAM](#)
- [Criando funções de eventos Python Lambda com uv in AWS SAM](#)
- [Criando funções de eventos Lambda com tempos de execução personalizados em AWS SAM](#)
- [Construindo camadas Lambda em AWS SAM](#)

Criando funções de eventos Node.js Lambda com esbuild em AWS SAM

Para criar e empacotar funções de Node.js AWS Lambda eventos, você pode usar o AWS SAMCLI com o JavaScript bundler esbuild. O bundler esbuild oferece suporte às funções Lambda nas quais você escreve. TypeScript

Para criar uma função Node.js Lambda com esbuild, adicione um Metadata objeto ao seu `AWS::Serverless::Function` recurso e especifique esbuild para o `BuildMethod`. Quando você executa o `sam build` comando, AWS SAM usa o esbuild para agrupar o código da função Lambda.

Propriedades de metadados

O objeto Metadata é compatível com as seguintes propriedades para esbuild.

BuildMethod

Especifica o empacotador do aplicativo. O único valor aceito é esbuild.

BuildProperties

Especifica as propriedades de construção do código da função do Lambda.

O objeto `BuildProperties` é compatível com as seguintes propriedades para esbuild. Todas as propriedades são opcionais. Por padrão, AWS SAM usa seu manipulador de funções Lambda como ponto de entrada.

EntryPoint

Especifica pontos de entrada para seu aplicativo.

Externo

Especifica a lista de pacotes a serem omitidos da compilação. Para obter mais informações, consulte [Externo](#) no site da esbuild.

Formato

Especifica o formato de saída dos JavaScript arquivos gerados em seu aplicativo. Para obter mais informações, consulte [Formato](#) no site do esbuild.

Carregador

Especifica a lista de configurações para carregar dados de um determinado tipo de arquivo.

MainFields

Especifica quais campos `package.json` tentar importar ao resolver um pacote. O valor padrão é `main,module`.

Minimizar

Especifica se o código de saída incluído deve ser minimizado. O valor padrão é `true`.

OutExtension

Personalize a extensão dos arquivos que o esbuild gera. Para obter mais informações, consulte [Extensão de saída](#) no site da esbuild.

Mapa de origem

Especifica se o bundler produz um arquivo de mapa de origem. O valor padrão é `false`.

Quando definido como `true`, `NODE_OPTIONS: --enable-source-maps` é anexado às variáveis de ambiente da função do Lambda e um mapa de origem é gerado e incluído na função.

Como alternativa, quando `NODE_OPTIONS: --enable-source-maps` é incluído nas variáveis de ambiente da função, `Sourcemap` é automaticamente definido como `true`.

Quando conflitante, `Sourcemap: false` tem precedência sobre `NODE_OPTIONS: --enable-source-maps`.

 Note

Por padrão, o Lambda criptografa todas as variáveis de ambiente em repouso com AWS Key Management Service (AWS KMS). Ao usar mapas de origem, para que a implantação seja bem-sucedida, a função de execução da sua função deve ter permissão para realizar a ação `kms:Encrypt`.

SourcesContent

Especifica se o código-fonte deve ser incluído no arquivo do mapa de origem. Configure essa propriedade quando `Sourcemap` estiver definido como `'true'`.

- Especifique `SourcesContent: 'true'` para incluir todo o código-fonte.
- Especifique `SourcesContent: 'false'` para excluir todo o código-fonte. Isso resulta em tamanhos menores de arquivos de mapas de origem, o que é útil na produção ao reduzir os tempos de inicialização. No entanto, o código-fonte não estará disponível no depurador.

O valor padrão é `SourcesContent: true`.

Para obter mais informações, consulte [Conteúdo de fontes](#) no site esbuild.

Destino

Especifica a versão de destino do ECMAScript. O valor padrão é `es2020`.

TypeScript Exemplo de função de evento Lambda

O exemplo de trecho AWS SAM de modelo a seguir usa esbuild para criar uma função Node.js Lambda a partir do código em 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>"
```

Criando funções de eventos do.NET Lambda com compilação AOT nativa no AWS SAM

Crie e empacote suas funções de AWS Lambda eventos do.NET 8 com o AWS Serverless Application Model (AWS SAM), utilizando a compilação nativa Ahead-of-Time (AOT) para melhorar AWS Lambda os tempos de inicialização a frio.

Tópicos

- [Visão geral da AOT nativa do .NET 8](#)
- [Utilizar AWS SAM com suas funções de eventos Lambda do.NET 8](#)
- [Pré-requisitos de instalação](#)
- [Defina funções de eventos Lambda do.NET 8 em seu AWS SAM modelo](#)
- [Crie seu aplicativo com o AWS SAM CLI](#)
- [Saiba mais](#)

Visão geral da AOT nativa do .NET 8

Historicamente, as funções .NET Lambda têm tempos de inicialização a frio que afetam a experiência do usuário, a latência do sistema e os custos de uso de seus aplicativos sem servidor. Com a compilação AOT nativa do .NET, você pode melhorar os tempos de inicialização a frio das suas funções do Lambda. Para saber mais sobre a AOT nativa para .NET 8, consulte [Usando a AOT nativa no repositório GitHub](#) Dotnet.

Utilizar AWS SAM com suas funções de eventos Lambda do .NET 8

Faça o seguinte para configurar as funções do Lambda com .NET 8 com o AWS Serverless Application Model (AWS SAM):

- Instale os pré-requisitos em sua máquina de desenvolvimento.
- Defina as funções Lambda do .NET 8 em seu AWS SAM modelo.
- Crie seu aplicativo com AWS SAMCLI o.

Pré-requisitos de instalação

Os seguintes pré-requisitos são exigidos:

- O AWS SAMCLI
- .NET Core CLI
- A ferramenta global Amazon.Lambda.Tools do .NET Core
- Docker

Instale o AWS SAM CLI

1. Para verificar se você já tem AWS SAMCLI instalado, execute o seguinte:

```
sam --version
```

2. Para instalar o AWS SAMCLI, consulte [Instale o AWS SAM CLI](#).
3. Para atualizar uma versão instalada do AWS SAMCLI, consulte [Atualizar o AWS SAM CLI](#).

Instalado o .NET Core CLI

1. Para fazer download e instalar a .NET Core CLI, consulte [Download .NET](#) do website da Microsoft.
2. Para obter mais informações sobre a CLI do .NET Core, consulte [.NET Core CLI](#) no AWS Lambda Guia do Desenvolvedor.

Instale a ferramenta global Amazon.Lambda.Tools do .NET Core

1. Execute este comando: .

```
dotnet tool install -g Amazon.Lambda.Tools
```

2. Se você já tiver a ferramenta instalada, certifique-se de que esteja usando a versão mais recente com o seguinte comando:

```
dotnet tool update -g Amazon.Lambda.Tools
```

3. Para obter mais informações sobre a ferramenta global Amazon.Lambda.Tools do .NET Core, consulte [AWS as extensões para o repositório CLI do .NET](#) em GitHub

Instalar Docker

- Construir com AOT nativo, precisa que Docker seja instalado. Para obter instruções de instalação, consulte [Instalando o Docker para usar com o AWS SAM CLI](#).

Defina funções de eventos Lambda do .NET 8 em seu AWS SAM modelo

Para definir uma função Lambda.NET8 em seu AWS SAM modelo, faça o seguinte:

1. Execute o seguinte comando em um diretório inicial de sua escolha:

```
sam init
```

2. Selecione AWS Quick Start Templates para escolher um modelo inicial.
3. Escolha o modelo Hello World Example.
4. Escolha não usar o runtime e o tipo de pacote mais populares inserindo n.
5. Em runtime, escolha dotnet8.

6. Em tipo de pacote, escolha Zip.
7. Em modelo inicial, escolha Hello World Example using native AOT.

Instalar Docker

- Construir com AOT nativo, precisa que Docker seja instalado. Para obter instruções de instalação, consulte [Instalando o Docker para usar com o AWS SAM CLI](#).

```
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 a Event propriedade de um `AWS::Serverless::Function` é definida como `Api`, mas a `RestApiId` propriedade não é especificada, AWS SAM gera o `AWS::ApiGateway::RestApi` CloudFormation recurso.

Crie seu aplicativo com o AWS SAM CLI

No diretório raiz do seu projeto, execute o comando `sam build` para começar a criar sua aplicação. Se a propriedade `PublishAot` tiver sido definida no arquivo de projeto do .NET 8, a CLI do AWS SAM será criada com a compilação AOT nativa. Para saber mais sobre a propriedade `PublishAot`, consulte [Implantação nativa de AOT](#) na documentação .NET da Microsoft.

Para criar sua função, ele AWS SAMCLI invoca a CLI do.NET Core, que usa a ferramenta global Amazon.Lambda.Tools do.NET Core.

Note

Ao criar, se um arquivo `.sln` existir no mesmo diretório ou no diretório principal do seu projeto, o diretório que contém o arquivo `.sln` será montado no contêiner. Se um arquivo `.sln` não for encontrado, somente a pasta do projeto será montada. Portanto, se você estiver criando um aplicativo de vários projetos, verifique se o arquivo `.sln` está localizado na propriedade.

Saiba mais

Para obter mais informações sobre a criação de funções do Lambda com .NET 8, consulte [Introdução ao runtime do .NET 8 para o AWS Lambda](#).

Para obter uma referência do comando `sam build`, consulte [sam build](#).

Criando funções de eventos do Rust Lambda com Lambda de carga in AWS SAM

Use a interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) com suas funções de AWS Lambda evento Rust.

Tópicos

- [Pré-requisitos](#)
- [Configurar AWS SAM para usar com as funções de eventos Rust Lambda](#)
- [Exemplos](#)
- [A otimização do Rust incorpora GitHub Ações](#)

Pré-requisitos

Linguagem Rust

Para instalar Rust, consulte [Instalar Rust](#) no Rust site do idioma.

Cargo Lambda

O AWS SAMCLI requer a instalação de um subcomando [Cargo Lambda](#), para Cargo. Para obter instruções, consulte [Instalação](#) na Documentação do Cargo Lambda.

Docker

É necessário criar e testar as Rust funções do Lambda Docker. Para obter instruções de instalação, consulte [Instalação do Docker](#).

Configurar AWS SAM para usar com as funções de eventos Rust Lambda

Etapa 1: configure seu AWS SAM modelo

Configure seu AWS SAM modelo com o seguinte:

- Binário - Opcional. Especifique quando um único Cargo pacote define mais de um binário, para identificar qual binário criar para essa função. Você não precisa dessa propriedade quando cada função é seu próprio Cargo pacote, como em um Cargo espaço de trabalho.
- BuildMethod – rust-cargolambda.
- CodeUri— caminho para seu Cargo.toml arquivo.
- Manipulador – bootstrap.
- Tempo de execução – provided.al2023.

Para saber mais sobre tempos de execução personalizados, consulte [AWS Lambda Tempos de execução personalizados](#) no Guia do AWS Lambda desenvolvedor.

Aqui está um exemplo de um AWS SAM modelo configurado:

```
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.al2023
  ...
```


Etapa 2: use o AWS SAM CLI com sua função de evento Rust Lambda

Use qualquer AWS SAMCLI comando com seu AWS SAM modelo. Para obter mais informações, consulte [AWS SAM CLI](#).

Exemplos

Exemplo do Hello World

Neste exemplo, criamos o aplicativo Hello World de amostra usando Rust como nosso tempo de execução.

Primeiro, inicializamos um novo aplicativo sem servidor usando o `sam init`. Durante o fluxo interativo, selecionamos o aplicativo Hello World e escolhemos o tempo de execução do 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 - ruby4.0
    21 - ruby3.3
    22 - ruby3.2
    23 - rust (provided.al2)
    24 - rust (provided.al2023)
```

Runtime: *24*

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.al2023)
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
```

A seguir está a estrutura do nosso aplicativo Hello World:

```
hello-rust
### README.md
### events
```

```
#   ### event.json
### rust_app
#   ### Cargo.toml
#   ### src
#       ### main.rs
### samconfig.toml
### template.yaml
```

Em nosso AWS SAM modelo, nossa Rust função é definida da seguinte forma:

```
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.al2023
      Architectures:
        - x86_64
      Events:
        HelloWorld:
          Type: Api
          Properties:
            Path: /hello
            Method: get
```

Em seguida, executamos `sam build` para criar nosso aplicativo e nos preparar para a implantação. O AWS SAMCLI cria um diretório `.aws-sam` e organiza nossos artefatos de construção lá. Nossa função é construída usando Cargo Lambda e armazenada como um binário executável em `.aws-sam/build/HelloWorldFunction/bootstrap`.

Note

Se você planeja executar o comando `sam local invoke` no macOS, precisa criar funções diferentes antes de invocar. Para fazer isso, use o seguinte comando:

- `SAM_BUILD_MODE=debug sam build`

Esse comando só será necessário se for feito um teste local. Isso não é recomendado na criação para implantação.

```
hello-rust$ sam build
Starting Build use cache
Cache is invalid, running build and copying resources for following functions
(HelloWorldFunction)
Building codeuri: /Users/.../hello-rust/rust_app runtime: provided.al2023 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
```

Em seguida, implantamos nosso aplicativo usando `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	<code>https://ggdxec9le9.execute-api.us-west-2.amazonaws.com/Prod/hello/</code>
Key	HelloWorldFunction
Description	Hello World Lambda Function ARN
Value	<code>arn:aws:lambda:us-west-2:012345678910:function:hello-rust-HelloWorldFunction-yk4HzGzYeZBj</code>

Successfully created/updated stack - hello-rust in us-west-2

Para testar, podemos invocar nossa função do Lambda usando o endpoint da API.

```
$ curl https://ggdxec9le9.execute-api.us-west-2.amazonaws.com/Prod/hello/
Hello World!%
```

Para testar nossa função localmente, primeiro garantimos que a propriedade Architectures de nossa função corresponda à nossa máquina local.

```
...
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
    Metadata:
      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.al2023
      Architectures:
```

```
- arm64
...
```

Como modificamos nossa arquitetura de x86_64 para arm64 neste exemplo, executamos `sam build` para atualizar nossos artefatos de construção. Em seguida, executamos `sam local invoke` para invocar localmente nossa função.

```
hello-rust$ sam local invoke
Invoking bootstrap (provided.al2023)
Local image was not found.
Removing rapid images for repo public.ecr.aws/sam/emulation-provided.al2023
Building
image.....
Using local image: public.ecr.aws/lambda/provided:al2023-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
```

Projeto único de função de evento Lambda

Aqui está um exemplo de um aplicativo sem servidor contendo uma função Rust Lambda.

Estrutura do diretório do projeto:

```
.
### Cargo.lock
### Cargo.toml
### src
#   ### main.rs
### template.yaml
```

AWS SAM modelo:

```
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.al2023
...
```

Projeto de função de evento Lambda múltipla

Aqui está um exemplo de um aplicativo sem servidor contendo várias funções do Rust Lambda, organizadas como um espaço de trabalho. Cargo

Recomendamos um Cargo espaço de trabalho para aplicativos com várias funções do Rust Lambda. Cada função é seu próprio pacote, portanto, as funções podem declarar dependências independentes enquanto compartilham código comum por meio de um pacote de biblioteca. Cada pacote produz um único binário com o nome do pacote, então você não precisa definir a propriedade `Binary build`.

Estrutura do diretório do projeto:

```
.
### Cargo.lock
### Cargo.toml
### function_a
#   ### Cargo.toml
#   ### src
#       ### main.rs
### function_b
#   ### Cargo.toml
#   ### src
#       ### main.rs
### template.yaml
```

`Cargo.toml`Arquivo do espaço de trabalho, na raiz do projeto:

```
[workspace]
resolver = "2"
```

```
members = [
    "function_a",
    "function_b",
]

[workspace.dependencies]
lambda_runtime = "0.13"
serde = { version = "1", features = ["derive"] }
tokio = { version = "1", features = ["macros", "rt"] }
```

Cargo.toml arquivo para cada função, como function_a/Cargo.toml:

```
[package]
name = "function_a"
version = "0.1.0"
edition = "2021"

[dependencies]
lambda_runtime = { workspace = true }
serde = { workspace = true }
tokio = { workspace = true }
```

AWS SAM modelo. O CodeUri de cada função aponta para o diretório do pacote dessa função:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  FunctionA:
    Type: AWS::Serverless::Function
    Metadata:
      BuildMethod: rust-cargolambda
    Properties:
      CodeUri: ./function_a
      Handler: bootstrap
      Runtime: provided.al2023
  FunctionB:
    Type: AWS::Serverless::Function
    Metadata:
      BuildMethod: rust-cargolambda
    Properties:
      CodeUri: ./function_b
      Handler: bootstrap
```

```
Runtime: provided.al2023
```

Note

O AWS SAMCLI cria todas as funções do espaço de trabalho no `target` diretório compartilhado do espaço de trabalho, então Cargo compila dependências compartilhadas uma vez em vez de uma vez para cada função. Esse comportamento requer a AWS SAMCLI versão 1.165.0 ou posterior. Nas versões anteriores, cada função é criada em seu próprio `target` diretório e a árvore de dependências completa é recompilada para cada função, o que torna as compilações mais lentas à medida que você adiciona funções.

Dê a cada pacote de funções um nome binário exclusivo. Os nomes dos pacotes são exclusivos em um espaço de trabalho, portanto, o nome binário padrão já é exclusivo. Se você substituir o nome binário por uma `[[bin]]` seção, não dê a dois pacotes o mesmo nome binário. Eles são compilados no mesmo caminho no `target` diretório compartilhado e se sobrescrevem. Ele AWS SAMCLI registra um aviso quando detecta isso.

Como alternativa, um único pacote pode definir vários binários. Nesse caso, use a propriedade `Binary build` para selecionar o binário para cada função:

```
Resources:
  FunctionA:
    Type: AWS::Serverless::Function
    Metadata:
      BuildMethod: rust-cargolambda
      BuildProperties:
        Binary: function_a
    Properties:
      CodeUri: ./
      Handler: bootstrap
      Runtime: provided.al2023
```

A otimização do Rust incorpora GitHub Ações

As compilações do Rust exigem muita computação, e um executor de integração contínua começa sem artefatos compilados. Aplicativos com várias funções que compartilham grandes dependências, como o `AWSSDK`, podem passar a maior parte do tempo de compilação compilando as mesmas dependências. As práticas a seguir reduzem o tempo de construção `GitHub Actions`.

Use a AWS SAMCLI versão 1.165.0 ou posterior para espaços de trabalho

As versões 1.165.0 e posteriores criam todos os membros de um espaço de Cargo trabalho no `target` diretório compartilhado do espaço de trabalho, para que as dependências compartilhadas sejam compiladas uma vez por compilação, em vez de uma vez para cada função. Especifique a versão mínima ao instalar o AWS SAMCLI para que uma compilação não volte silenciosamente ao comportamento mais lento.

Armazene em cache o Cargo registro e o **target** diretório

Armazene em cache o Cargo registro (`~/.cargo/registry` e `~/.cargo/git/db`) e o `target` diretório do espaço de trabalho entre as execuções, para que as dependências inalteradas sejam restauradas em vez de recompiladas. Use um cache separado para cada destino de compilação. Um trabalho que faz a compilação cruzada de artefatos de lançamento `arm64` produz artefatos diferentes de um trabalho para o qual é compilado nativamente `x86_64`, portanto, um cache compartilhado nunca corresponde.

Incluir configurações de compilação na chave de cache

Cargo inclui configurações como `opt-level` e `codegen-units` na impressão digital que ele usa para decidir se um artefato compilado pode ser reutilizado. Se você alterar a `[profile.release]` seção do `Cargo.toml` arquivo do seu espaço de trabalho sem alterar a chave do cache, o cache será restaurado, mas todas as caixas serão recompiladas de qualquer maneira. Inclua um hash do `Cargo.toml` arquivo do espaço de trabalho na chave de cache para que a alteração de uma configuração de perfil inicie um novo cache.

Confirme seu **Cargo.lock** arquivo

As funções Lambda são executáveis, então confirme seu arquivo. `Cargo.lock` Isso fornece compilações reproduzíveis e uma chave de cache estável que muda somente quando suas dependências mudam.

Ajuste o perfil de lançamento para o tempo de construção e a inicialização a frio

Seu código de função é recompilado em cada execução, porque ele muda com mais frequência do que suas dependências. O perfil de versão padrão otimiza a taxa de transferência de tempo de execução, da qual muitas funções do Lambda não precisam. A otimização do tamanho produz binários menores, o que também ajuda no tempo de inicialização a frio, e o aumento do número de unidades de geração de código aumenta o paralelismo durante a compilação. Deixe a otimização do tempo do link (`lto`) desativada, pois isso torna a compilação mais lenta. Adicione o seguinte ao seu `Cargo.toml` arquivo de espaço de trabalho:

```
[profile.release]
opt-level = "s"
codegen-units = 256
lto = false
strip = true
```

Meça o efeito em seu próprio aplicativo. Essas configurações trocam uma pequena quantidade de desempenho de tempo de execução por tempo de construção e tamanho binário.

Evite execuções duplicadas do fluxo de trabalho

Um fluxo de trabalho executado em ambos push e em pull_request eventos é executado duas vezes para a mesma confirmação. GitHub Actions caches são definidos por branch e pull request, então as duas execuções gravam em escopos de cache diferentes e nenhuma reutiliza o cache da outra. Use um grupo de simultaneidade digitado na confirmação principal, para que apenas uma execução crie cada confirmação.

O fluxo de trabalho a seguir cria um Cargo espaço de trabalho com funções do Rust Lambda e aplica arm64 as práticas anteriores:

```
name: Build

on:
  push:
    branches: [main]
  pull_request:

# Collapse the push and pull_request runs for the same commit into a single run.
concurrency:
  group: ${{ github.workflow }}-${{ github.event.pull_request.head.sha || github.sha }}
  cancel-in-progress: true

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v5

      - uses: dtolnay/rust-toolchain@stable
        with:
          targets: aarch64-unknown-linux-gnu
```

```
# Cache the Cargo registry and the workspace target directory. The key covers
# the compilation target, Cargo.lock, and the workspace Cargo.toml, so that
# changing a dependency or a release profile setting starts a new cache
# instead of restoring one whose artifacts Cargo discards.
- uses: actions/cache@v4
  with:
    path: |
      ~/.cargo/registry/index
      ~/.cargo/registry/cache
      ~/.cargo/git/db
      target
    key: cargo-arm64-${{ hashFiles('Cargo.lock', 'Cargo.toml') }}
    restore-keys: |
      cargo-arm64-

- name: Install build tools
  run: pip install cargo-lambda 'aws-sam-cli>=1.165.0'

- name: Build
  run: sam build
```

A `restore-keys` entrada permite que uma execução comece a partir do cache mais recente quando a chave não corresponde exatamente, para que uma alteração de dependência reutilize as caixas que não foram alteradas em vez de compilar tudo novamente.

Criando funções de eventos Python Lambda com uv in AWS SAM

Esse recurso está na versão prévia AWS SAM e está sujeito a alterações.

Use a interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) com uv um rápido instalador e resolvidor de pacotes Python para criar suas funções de eventos do AWS Lambda Python.

Tópicos

- [Pré-requisitos](#)
- [Configurar AWS SAM para usar com Python Lambda Event Functions e uv](#)
- [Exemplos](#)

Pré-requisitos

Python

Para instalar o Python, consulte Baixar o [Python no site do Python](#).

uv

AWS SAMCLI Requer a instalação de [uv](#) um instalador e resolvidor de pacotes Python extremamente rápido. Para obter instruções, consulte [Instalação](#) na Documentação do uv.

Opte pelo recurso AWS SAMCLI beta

Como esse recurso está em versão prévia, opte por usar um dos seguintes métodos:

1. Use as variáveis de ambiente: SAM_CLI_BETA_PYTHON_UV=1.
2. Adicione o seguinte ao arquivo `samconfig.toml`:

```
[default.build.parameters]
beta_features = true
[default.sync.parameters]
beta_features = true
```

3. Use a opção `--beta-features` ao usar um comando do AWS SAM CLI compatível. Por exemplo:

```
$ sam build --beta-features
```

4. Escolha a opção `y` quando AWS SAM CLI solicitar que você se inscreva. Este é um exemplo:

```
$ 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
```

Configurar AWS SAM para usar com Python Lambda Event Functions e uv

Etapa 1: configure seu AWS SAM modelo

Configure seu AWS SAM modelo com o seguinte:

- BuildMethod – `python-uv`.

- `CodeUri`— caminho para seu diretório de código de função contendo `pyproject.toml` ou `requirements.txt`.
- `Handler` — seu manipulador de funções (por exemplo, `app.lambda_handler`).
- `Runtime` — versão de tempo de execução do Python (por exemplo, `python3.12`).

Aqui está um exemplo de um AWS SAM modelo configurado:

```
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
  ...
```

Exemplos

Exemplo do Hello World

Neste exemplo, criamos uma amostra do aplicativo Hello World usando Python uv como gerenciador de pacotes.

uv pode usar uma `pyproject.toml` ou `requirements.txt` para ler dependências. Se ambos forem fornecidos, `uv build` lerá `requirements.txt` as dependências.

A seguir está a estrutura do nosso aplicativo Hello World:

```
hello-python-uv
### README.md
### events
#   ### event.json
### hello_world
#   ### __init__.py
#   ### app.py
#   ### pyproject.toml
```



```
### samconfig.toml
### template.yaml
```

Arquivo `pyproject.toml`:

```
[project]
name = "my-function"
version = "0.1.0"
requires-python = ">=3.12"
dependencies = [
    "requests>=2.31.0",
    "boto3>=1.28.0",
]
```

Em nosso AWS SAM modelo, nossa função Python é definida da seguinte forma:

```
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
```

Em seguida, executamos `sam build` para criar nosso aplicativo e nos preparar para a implantação. O AWS SAMCLI cria um diretório `.aws-sam` e organiza nossos artefatos de construção lá. Nossas dependências de função são instaladas usando `uv` e armazenadas em `.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

O método de python-uv construção é configurado por função na Metadata seção. Cada função em seu modelo pode usar um método de construção diferente, permitindo que você misture funções uv pip baseadas com funções baseadas no mesmo AWS SAM modelo. Se nenhum método de construção for especificado, pip é usado por padrão.

Criando funções de eventos Lambda com tempos de execução personalizados em AWS SAM

Você pode usar o [sam build](#) comando para criar tempos de execução personalizados necessários para sua função de evento Lambda. Você declara a sua função do Lambda para usar um runtime especificando Runtime: provided para a função.

Para criar um runtime personalizado, declare o atributo do recurso Metadata com uma entrada BuildMethod: makefile. Você fornece um makefile personalizado, no qual declara um destino

de construção do formulário `build-function-logical-id` que contém os comandos de construção para seu tempo de execução. Seu `makefile` é responsável por compilar o tempo de execução personalizado, se necessário, e copiar os artefatos de construção no local adequado para as etapas subsequentes do seu fluxo de trabalho. A localização do `makefile` é especificada pela propriedade `CodeUri` do recurso da função e deve ser nomeada `Makefile`.

Exemplos

Exemplo 1: Tempo de execução personalizado para uma função escrita em Rust

Note

Recomendamos criar funções do Lambda com Cargo Lambda. Para saber mais, consulte [Criando funções de eventos do Rust Lambda com Lambda de carga in AWS SAM](#).

O AWS SAM modelo a seguir declara uma função que usa um tempo de execução personalizado para uma função Lambda escrita em Rust e `build` instrui a execução dos comandos para o destino de compilação. `build-HelloRustFunction`

```
Resources:
  HelloRustFunction:
    Type: AWS::Serverless::Function
    Properties:
      FunctionName: HelloRust
      Handler: bootstrap.is.real.handler
      Runtime: provided
      MemorySize: 512
      CodeUri: .
    Metadata:
      BuildMethod: makefile
```

O `makefile` a seguir contém o destino da compilação e os comandos que serão executados. Observe que a propriedade `CodeUri` está definida como `.`, portanto, o `makefile` deve estar localizado no diretório raiz do projeto (ou seja, o mesmo diretório do arquivo do modelo AWS SAM do aplicativo). O nome do arquivo deve ser `Makefile`.

```
build-HelloRustFunction:
  cargo build --release --target x86_64-unknown-linux-musl
  cp ./target/x86_64-unknown-linux-musl/release/bootstrap $(ARTIFACTS_DIR)
```

Para obter mais informações sobre como configurar seu ambiente de desenvolvimento para executar o comando `cargo build` no makefile anterior, consulte a postagem do blog [Rust Runtime for AWS Lambda](#).

Exemplo 2: construtor de Makefile para Python3.12 (alternativa ao uso do construtor incluído)

Talvez você queira usar uma biblioteca ou módulo que não esteja incluído em um construtor empacotado. Este exemplo mostra um AWS SAM modelo para um Python3.12 tempo de execução com um construtor de makefile.

```
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: hello_world/
      Handler: app.lambda_handler
      Runtime: python3.12
    Metadata:
      BuildMethod: makefile
```

O makefile a seguir contém o destino da compilação e os comandos que serão executados. Observe que a propriedade `CodeUri` está definida como `hello_world`, portanto, o makefile deve estar localizado na raiz do subdiretório `hello_world` e o nome do arquivo deve ser `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
```

Construindo camadas Lambda em AWS SAM

Você pode usar AWS SAM para criar camadas Lambda personalizadas. As camadas do Lambda permitem que você extraia código de uma função de evento do Lambda que pode ser reutilizado em várias funções do Lambda. Criar somente camadas do Lambda (em vez de toda a aplicação) pode trazer benefícios de algumas maneiras. Isso pode ajudar na redução do tamanho dos pacotes de implantação, na separação da lógica da função de núcleo das dependências e na permissão para compartilhar dependências em várias funções. Para obter informações sobre camadas, consulte [Camadas do AWS Lambda](#) no Guia do desenvolvedor do AWS Lambda .

Como criar uma camada Lambda no AWS SAM

Note

Antes de criar uma camada Lambda, você deve primeiro escrever uma camada Lambda no seu modelo. AWS SAM Para obter informações e exemplos, consulte [Aumente a eficiência usando camadas Lambda com AWS SAM](#).

Para criar uma camada personalizada, declare-a em seu arquivo de modelo AWS Serverless Application Model (AWS SAM) e inclua uma seção de atributo de Metadata recurso com uma BuildMethod entrada. Os valores válidos para BuildMethod são identificadores de um [AWS Lambda runtime](#), ou makefile. Inclua uma entrada BuildArchitecture para especificar as arquiteturas do conjunto de instruções que sua camada suporta. Os valores válidos para BuildArchitecture são [arquiteturas dos conjuntos de instruções Lambda](#).

Se você especificar makefile, forneça o makefile personalizado, onde você declara um destino de construção do formulário build-*layer-logical-id* que contém os comandos de construção para sua camada. Seu makefile é responsável por compilar a camada, se necessário, e copiar os artefatos de construção no local adequado para as etapas subsequentes do seu fluxo de trabalho. A localização do makefile é especificada pela propriedade ContentUri do recurso de camada e deve ser nomeada Makefile.

Note

Quando você cria uma camada personalizada, AWS Lambda depende das variáveis de ambiente para encontrar seu código de camada. Os tempos de execução do Lambda incluem caminhos no diretório /opt em que seu código de camada é copiado. A estrutura de pastas de artefatos de construção do seu projeto deve corresponder à estrutura de pastas esperada do runtime para que seu código de camada personalizado possa ser encontrado. Por exemplo, para Python, você pode colocar seu código no subdiretório python/. Para NodeJS, você pode colocar seu código no subdiretório nodejs/node_modules/. Para obter mais informações, consulte [Incluindo dependências de biblioteca em uma camada](#) no AWS Lambda Guia do desenvolvedor.

Veja a seguir um exemplo de seção de atributo de recurso Metadata.

Metadata:

```
BuildMethod: python3.12
BuildArchitecture: arm64
```

 Note

Se você não incluir a seção do atributo do Metadata recurso, AWS SAM não cria a camada. Em vez disso, ele copia os artefatos de construção do local especificado na propriedade `CodeUri` do recurso de camada. Para obter mais informações, consulte a propriedade [ContentUri](#) do tipo de recurso `AWS::Serverless::LayerVersion`.

Ao incluir a seção do atributo do Metadata recurso, você pode usar o `sam build` comando para criar a camada, tanto como um objeto independente quanto como uma dependência de uma AWS Lambda função.

- Como um objeto independente. Talvez você queira criar apenas o objeto de camada, por exemplo, quando estiver testando localmente uma alteração de código na camada e não precisar criar todo o aplicativo. Para criar a camada de forma independente, especifique o recurso da camada com o comando `sam build layer-logical-id`.
- Como uma dependência de uma função do Lambda. Quando você inclui o ID lógico de uma camada na propriedade `Layers` de uma função do Lambda no mesmo arquivo de modelo AWS SAM, a camada é uma dependência dessa função do Lambda. Quando essa camada também inclui uma seção de atributo de recurso Metadata com uma entrada `BuildMethod`, você cria a camada criando o aplicativo inteiro com o comando `sam build` ou especificando o recurso da função com o comando `sam build function-logical-id`.

Exemplos

Exemplo de modelo 1: Criar uma camada com o ambiente de runtime Python 3.12

O AWS SAM modelo de exemplo a seguir cria uma camada em relação ao ambiente de execução do 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
```

Exemplo de modelo 2: Crie uma camada usando um makefile personalizado

O AWS SAM modelo de exemplo a seguir usa um personalizado makefile para criar a camada.

```
Resources:
  MyLayer:
    Type: AWS::Serverless::LayerVersion
    Properties:
      ContentUri: my_layer
      CompatibleRuntimes:
        - python3.12
    Metadata:
      BuildMethod: makefile
```

O seguinte makefile contém o destino da compilação e os comandos que serão executados. Observe que a propriedade ContentUri está definida como my_layer, portanto, o makefile deve estar localizado na raiz do subdiretório my_layer e o nome do arquivo deve ser Makefile. Observe também que os artefatos de construção são copiados para o python/ subdiretório para que AWS Lambda você possa encontrar o código da camada.

```
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 o makefile é chamado, o destino apropriado é acionado e os artefatos devem ser copiados para a variável ambiental exposta \$ARTIFACTS_DIR. Para obter mais informações, consulte [aws-lambda-builders](#) em. GitHub

Exemplo de comandos sam build

Os comandos `sam build` a seguir criam camadas que incluem as seções de atributos do recurso Metadata.

```
# 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
```


Teste seu aplicativo sem servidor com AWS SAM

Depois de escrever e criar a aplicação, tudo estará pronto para testá-la e verificar se ela funciona corretamente. Com a interface de linha de AWS SAM comando (CLI), você pode testar localmente seu aplicativo sem servidor antes de enviá-lo para a nuvem. AWS Testar a aplicação ajuda na confirmação da funcionalidade, da confiabilidade e do desempenho da aplicação, ao mesmo tempo que identifica problemas (bugs) que precisarão ser resolvidos.

Esta seção fornece orientação sobre práticas comuns que você pode seguir para testar a aplicação. Os tópicos desta seção se concentram principalmente nos testes locais que você pode fazer antes da implantação na AWS nuvem. Testar antes da implantação ajuda na identificação de problemas de forma proativa, reduzindo os custos desnecessários associados aos problemas de implantação. Cada tópico desta seção descreve um teste que você pode realizar e inclui exemplos que mostram como realizar o teste. Depois de testar a aplicação, tudo estará pronto para a depuração de quaisquer problemas encontrados.

Tópicos

- [Introdução aos testes com o Local do SAM command](#)
- [Invoke localmente as funções de eventos do Lambda com AWS SAM](#)
- [Execute localmente o API Gateway com AWS SAM](#)
- [Introdução aos testes em nuvem com evento de teste remoto do sam](#)
- [Introdução aos testes na nuvem com invocação remota do sam](#)
- [Automatize os testes de integração local com AWS SAM](#)
- [Gere amostras de cargas úteis de eventos com AWS SAM](#)
- [Testando e depurando funções duráveis](#)

Introdução aos testes com o Local do SAM command

Use o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam local` para testar seus aplicativos sem servidor localmente.

Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#).

Pré-requisitos

Para usar o `sam local`, instale o AWS SAM CLI fazendo o seguinte:

- [AWS SAM Pré-requisitos](#) .
- [Instale o AWS SAM CLI](#).

Antes de usar `sam local`, recomendamos uma compreensão básica do seguinte:

- [Configurando o AWS SAM CLI](#).
- [Crie seu aplicativo em AWS SAM](#).
- [Introdução à construção com AWS SAM](#).
- [Introdução à implantação com AWS SAM](#).

Usar o Local do SAM command

Use o comando `sam local` com qualquer um de seus subcomandos para realizar diferentes tipos de testes locais para seu aplicativo.

```
$ sam local <subcommand>
```

Para obter mais informações sobre cada subcomando, veja o seguinte:

- [Introdução a sam local generate-event](#)— Gere AWS service (Serviço da AWS) eventos para testes locais.
- [Introdução a sam local invoke](#)— Inicie uma invocação única de uma função de evento AWS Lambda localmente.
- [Introdução a sam local start-api](#) – Execute suas funções do Lambda usando um servidor HTTP local.
- [Introdução a sam local start-lambda](#)— Execute suas funções do Lambda usando um servidor HTTP local para uso com os SDKs AWS CLI ou.

Introdução aos testes com generate-event local do sam

Use o AWS Serverless Application Model `sam local generate-event` subcomando Command Line Interface (AWS SAMCLI) para gerar amostras de carga útil de eventos para suporte. Serviços da AWS Em seguida, você pode modificar e passar esses eventos para os recursos locais para testes.

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter uma lista de opções de comando `sam local generate-event`, consulte [sam local generate-event](#).

Um evento é um objeto JSON que é gerado quando um AWS service (Serviço da AWS) executa uma ação ou tarefa. Esses eventos contêm informações específicas, como os dados que foram processados ou a data e hora do evento. A maioria Serviços da AWS gera eventos e os eventos de cada serviço são formatados exclusivamente para seu serviço.

Os eventos gerados por um serviço são passados para outros serviços como uma fonte de eventos. Por exemplo, um item colocado em um bucket do Amazon Simple Storage Service (Amazon S3) pode gerar um evento. Esse evento pode então ser usado como fonte de eventos para uma função de AWS Lambda evento para processar ainda mais os dados.

Os eventos que você gera com `sam local generate-event` são formatados na mesma estrutura dos eventos reais criados pelo AWS serviço. Você pode modificar o conteúdo desses eventos e usá-los para testar recursos em seu aplicativo.

Pré-requisitos

Para usar o `sam local generate-event`, instale o AWS SAM CLI fazendo o seguinte:

- [AWS SAM Pré-requisitos](#) .
- [Instale o AWS SAM CLI](#).

Antes de usar `sam local generate-event`, recomendamos uma compreensão básica do seguinte:

- [Configurando o AWS SAM CLI](#).
- [Crie seu aplicativo em AWS SAM](#).
- [Introdução à construção com AWS SAM](#).
- [Introdução à implantação com AWS SAM](#).

Gere eventos de exemplo

Use o AWS SAMCLI `sam local generate-event` subcomando para gerar eventos para suporte Serviços da AWS.

Para ver uma lista de opções suportadas Serviços da AWS

1. Execute o seguinte:

```
$ sam local generate-event
```

2. A lista de compatíveis Serviços da AWS será exibida. Este é um exemplo:

```
$ sam local generate-event
...
Commands:
  alb
  alexa-skills-kit
  alexa-smart-home
  apigateway
  appsync
  batch
  cloudformation
  ...
```

Para gerar um evento local

1. Execute `sam local generate-event` e forneça o nome do serviço suportado. Isso exibirá uma lista de tipos de eventos que você pode gerar. Este é um exemplo:

```
$ 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. Para gerar o evento de amostra, execute `sam local generate-event`, fornecendo o serviço e o tipo de evento.

```
$ sam local generate-event <service> <event>
```

Este é um exemplo:

```
$ 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": "EXAMPLE123/5678abcdefghijklmbdaisawesome/
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"
        }
      }
    }
  ]
}
```

```
]
}
```

Esses exemplos de eventos contêm valores de espaço reservado. Você pode modificar esses valores para referenciar recursos reais em seu aplicativo ou valores para ajudar nos testes locais.

Para modificar um evento de amostra

1. Você pode modificar eventos de amostra no prompt de comando. Para ver suas opções, execute o seguinte:

```
$ sam local generate-event <service> <event> --help
```

Este é um exemplo:

```
$ 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. Use qualquer uma dessas opções no prompt de comando para modificar a carga útil do evento de amostra. Este é um exemplo:

```
$ 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"
        }
      }
    }
  ]
}

```

Use eventos gerados para testes locais

Salve seus eventos gerados localmente e use outros subcomandos `sam local` para testar.

Para salvar seus eventos gerados localmente

- Execute o seguinte:

```
$ sam local generate-event <service> <event> <event-option> > <filename.json>
```

Veja a seguir um exemplo de um evento sendo salvo como um `s3.json` arquivo na `events` pasta do nosso projeto.

```
sam-app$ sam local generate-event s3 put --bucket amzn-s3-demo-bucket > events/  
s3.json
```

Para usar um evento gerado para testes locais

- Passe o evento com outros subcomandos `sam local` usando a opção `--event`.

Veja a seguir um exemplo de uso do evento `s3.json` para invocar nossa função do 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
```


Saiba mais

Para obter uma lista de todas as opções `sam local generate-event`, consulte [sam local generate-event](#).

Para uma demonstração do uso `sam local`, consulte [AWS SAM para desenvolvimento local](#). [Testando Nuvem AWS recursos de ambientes de desenvolvimento local](#) na série Serverless Land Sessions with SAM on. YouTube

Introdução aos testes com SAM invocação local

Use o AWS Serverless Application Model `sam local invoke` subcomando Command Line Interface (AWS SAMCLI) para iniciar localmente uma invocação única de uma função de evento. AWS Lambda

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter uma lista de opções de comando `sam local invoke`, consulte [SAM invocação local](#).
- Para obter um exemplo de uso do `sam local invoke` durante um fluxo de trabalho de desenvolvimento típico, consulte [Etapa 7: \(opcional\) Teste seu aplicativo localmente](#).

Note

Não é recomendável usar os recursos de invocação local do SAM CLI em código não confiável. Para ter um isolamento completo do seu ambiente local, execute o código diretamente no serviço Lambda.

Note

`sam local invoke` suporta funções duráveis com recursos automáticos de verificação e repetição. Ao invocar funções duráveis localmente, o estado de execução é gerenciado automaticamente.

Pré-requisitos

Para usar o `sam local invoke`, instale o AWS SAM CLI fazendo o seguinte:

- [AWS SAM Pré-requisitos](#).
- [Instale o AWS SAM CLI](#).

Antes de usar `sam local invoke`, recomendamos uma compreensão básica do seguinte:

- [Configurando o AWS SAM CLI](#).
- [Crie seu aplicativo em AWS SAM](#).
- [Introdução à construção com AWS SAM](#).
- [Introdução à implantação com AWS SAM](#).

Invocar uma função de evento Lambda localmente

Quando você executa `sam local invoke`, AWS SAMCLI pressupõe que seu diretório de trabalho atual seja o diretório raiz do seu projeto. O AWS SAM CLI, primeiro procurará um arquivo `template.[yaml|yml]` dentro de uma subpasta `.aws-sam`. Se não for encontrado, o AWS SAMCLI procurará um arquivo `template.[yaml|yml]` em seu diretório de trabalho atual.

Para invocar uma função de evento Lambda localmente

1. No diretório raiz do seu projeto, execute o seguinte:

```
$ sam local invoke <options>
```

2. Se seu aplicativo contiver mais de uma função, forneça o ID lógico da função. Este é um exemplo:

```
$ sam local invoke HelloWorldFunction
```

3. O AWS SAMCLI constrói sua função em um contêiner local Docker. Em seguida, ele invoca sua função e gera a resposta da sua função.

Este é um exemplo:

```
$ 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\"}"}
```

Gerenciar logs do

Ao usar `sam local invoke`, a saída de tempo de execução da função do Lambda (por exemplo, registros) é enviada para `stderr` e o resultado da função do Lambda é enviado para `stdout`.

Veja a seguir um exemplo de uma função do Lambda básica:

```
def handler(event, context):
    print("some log") # this goes to stderr
    return "hello world" # this goes to stdout
```

Você pode salvar essas saídas padrão. Este é um exemplo:

```
$ 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
```

Você pode usar essas saídas padrão para automatizar ainda mais seus processos de desenvolvimento local.

Opções

Passe eventos personalizados para invocar a função de evento Lambda

Para passar um evento para a função do Lambda, use a opção `--event`. Este é um exemplo:

```
$ sam local invoke --event events/s3.json S3JsonLoggerFunction
```

Você pode criar eventos com o subcomando `sam local generate-event`. Para saber mais, consulte [Introdução aos testes com generate-event local do sam](#).

Passe variáveis de ambiente ao invocar sua função de evento Lambda

Se sua função do Lambda usa variáveis de ambiente, você pode passá-las durante o teste local com a opção `--env-vars`. Essa é uma ótima maneira de testar uma função do Lambda localmente com serviços da aplicação que já estão implantados na nuvem. Este é um exemplo:

```
$ sam local invoke --env-vars locals.json
```

Especificar um modelo ou função

Para especificar um modelo para o AWS SAMCLI para referência, use a opção `--template`. Eles AWS SAMCLI carregarão apenas esse AWS SAM modelo e os recursos para os quais ele aponta.

Para invocar uma função de um aplicativo ou pilha aninhada, forneça o ID lógico do aplicativo ou da pilha junto com o ID lógico da função. Este é um exemplo:

```
$ sam local invoke StackLogicalId/FunctionLogicalId
```

Teste uma função de evento Lambda a partir do seu Terraform project

Use a opção `--hook-name` de testar localmente as funções do Lambda em seus projetos Terraform. Para saber mais, consulte [Usar o AWS SAM CLI por Terraform para depuração e teste locais](#).

Este é um exemplo:

```
$ sam local invoke --hook-name terraform --beta-features
```

Práticas recomendadas

Se seu aplicativo tiver um diretório `.aws-sam` executando `sam build`, certifique-se de executar o `sam build` sempre que atualizar o código da função. Em seguida, execute o `sam local invoke` para testar localmente seu código de função atualizado.

O teste local é uma ótima solução para desenvolvimento e teste rápidos antes da implantação na nuvem. No entanto, os testes locais não validam tudo, como permissões entre seus recursos na nuvem. Tanto quanto possível, teste seus aplicativos na nuvem. Recomendamos [usar o sam sync](#) para acelerar seus fluxos de trabalho de testes na nuvem.

Exemplos

Gere um evento de amostra do Amazon API Gateway e use-o para invocar uma função de evento Lambda localmente

Primeiro, geramos uma carga útil de evento de API HTTP do API Gateway e a salvamos em nossa pasta `events`.

```
$ sam local generate-event apigateway http-api-proxy > events/apigateway_event.json
```

Em seguida, modificamos nossa função do Lambda para retornar um valor de parâmetro do evento.

```
def lambda_handler(event, context):
    print("HelloWorldFunction invoked")
    return {
        "statusCode": 200,
        "body": json.dumps({
            "message": event['queryStringParameters']['parameter2'],
        }),
    }
```

Em seguida, invocamos localmente nossa função do Lambda e fornecemos nosso evento personalizado.

```
$ 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\"}"}%

```

Passe variáveis de ambiente ao invocar uma função de evento Lambda localmente

Esse aplicativo tem uma função do Lambda que usa uma variável de ambiente para o nome de uma tabela do Amazon DynamoDB. Veja a seguir um exemplo da função definida no AWS SAM modelo:

```

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
    ...

```

Queremos testar localmente nossa função do Lambda enquanto ela interage com nossa tabela do DynamoDB na nuvem. Para fazer isso, criamos nosso arquivo de variáveis de ambiente e o salvamos no diretório raiz do nosso projeto como `locals.json`. O valor fornecido aqui `SAMPLE_TABLE` faz referência à nossa tabela do DynamoDB na nuvem.

```
{
```

```
"getAllItemsFunction": {  
  "SAMPLE_TABLE": "dev-demo-SampleTable-1U991234LD5UM98"  
}  
}
```

Em seguida, executamos `sam local invoke` e passamos nossas variáveis de ambiente com a opção `--env-vars`.

```
$ 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":"{ }"}
```

Saiba mais

Para obter uma lista de todas as opções `sam local invoke`, consulte [SAM invocação local](#).

Para uma demonstração do uso `sam local`, consulte [AWS SAM para desenvolvimento local](#).
[Testando Nuvem AWS recursos de ambientes de desenvolvimento local](#) na série Serverless Land Sessions with SAM on. YouTube

Introdução aos testes com SAM API de inicialização local

Use o AWS Serverless Application Model `sam local start-api` subcomando Command Line Interface (AWS SAMCLI) para executar suas funções de AWS Lambda evento localmente e testar por meio de um host de servidor HTTP local. Esse tipo de teste é útil para funções do Lambda invocadas por um endpoint do Amazon API Gateway.

- Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter uma lista de opções de comando `sam local start-api`, consulte [SAM API de inicialização local](#).
- Para obter um exemplo de uso do `sam local start-api` durante um fluxo de trabalho de desenvolvimento típico, consulte [Etapa 7: \(opcional\) Teste seu aplicativo localmente](#).

 Note

`sam local start-api` suporta funções duráveis com recursos automáticos de verificação e repetição. As funções duráveis funcionam perfeitamente com o endpoint local do API Gateway.

Pré-requisitos

Para usar o `sam local start-api`, instale o AWS SAM CLI fazendo o seguinte:

- [AWS SAM Pré-requisitos](#) .
- [Instale o AWS SAM CLI](#).

Antes de usar `sam local start-api`, recomendamos uma compreensão básica do seguinte:

- [Configurando o AWS SAM CLI](#).
- [Crie seu aplicativo em AWS SAM](#).
- [Introdução à construção com AWS SAM](#).
- [Introdução à implantação com AWS SAM](#).

Usando SAM API de inicialização local

Quando você executa `sam local start-api`, AWS SAMCLI pressupõe que seu diretório de trabalho atual seja o diretório raiz do seu projeto. O AWS SAM CLI, primeiro procurará um arquivo `template.[yaml|yml]` dentro de uma subpasta `.aws-sam`. Se não for encontrado, o AWS SAMCLI procurará um arquivo `template.[yaml|yml]` em seu diretório de trabalho atual.

Para iniciar um servidor HTTP local

1. No diretório raiz do seu projeto, execute o seguinte:

```
$ sam local start-api <options>
```

2. O AWS SAMCLI cria suas funções do Lambda em um contêiner local Docker. Em seguida, ele envia o endereço local para o endpoint do servidor HTTP. Este é um exemplo:

```
$ 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. Você pode invocar sua função do Lambda por meio do navegador ou do prompt de comando. Este é um exemplo:

```
sam-app$ curl http://127.0.0.1:3000/hello
{"message": "Hello world!"}%
```

4. Ao fazer alterações no código da função do Lambda, considere o seguinte para atualizar seu servidor HTTP local:
- Se seu aplicativo não tiver um diretório `.aws-sam` e sua função usar uma linguagem interpretada, os AWS SAMCLI atualizarão automaticamente sua função criando um novo contêiner e hospedando-o.
 - Se seu aplicativo tiver um diretório `.aws-sam`, você precisará executar `sam build` para atualizar sua função. Em seguida, execute `sam local start-api` novamente para hospedar a função.
 - Se sua função usa uma linguagem compilada ou se seu projeto requer suporte a pacotes complexos, execute sua própria solução de compilação para atualizar sua função. Em seguida, execute `sam local start-api` novamente para hospedar a função.

Funções de eventos Lambda que usam autorizadores Lambda

Note

Esse atributo é novo na versão 1.80.0 do AWS SAM CLI. Para atualizar, consulte [Atualizar o AWS SAM CLI](#).

Para funções Lambda que usam autorizadores Lambda, os AWS SAMCLI invocarão automaticamente seu autorizador Lambda antes de invocar seu endpoint de função do Lambda.

Veja a seguir um exemplo de como iniciar um servidor HTTP local para uma função que usa um autorizador 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 você invoca seu endpoint da função do Lambda por meio do servidor HTTP local, o AWS SAMCLI invoca primeiro seu autorizador Lambda. Se a autorização for bem-sucedida, AWS SAM e CLI invocarão seu endpoint da função do Lambda. Este é um exemplo:

```
$ 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 -
```

Opções

Reutilize continuamente os contêineres para acelerar as invocações de funções locais

Por padrão, o AWS SAMCLI cria um novo contêiner toda vez que sua função é invocada por meio do servidor HTTP local. Use a opção `--warm-containers` de reutilizar automaticamente seu contêiner para invocações de funções. Isso acelera o tempo necessário para AWS SAMCLI preparar sua função do Lambda para invocação local. Você pode personalizar ainda mais essa opção fornecendo o argumento `eager` ou `lazy`.

- `eager` – Os contêineres para todas as funções são carregados na inicialização e persistem entre as invocações.
- `lazy` – Os contêineres são carregados somente quando cada função é invocada pela primeira vez. Eles então persistem para invocações adicionais.

Este é um exemplo:

```
$ sam local start-api --warm-containers eager
```

Ao usar `--warm-containers` e modificar o código da função do Lambda:

- Se seu aplicativo tiver um diretório `.aws-sam`, execute `sam build` para atualizar o código da função nos artefatos de construção do seu aplicativo.
- Quando uma alteração de código é detectada, o AWS SAMCLI desliga automaticamente o contêiner da função do Lambda.
- Quando você invoca a função novamente, o AWS SAMCLI cria automaticamente um novo contêiner.

Se a observação de arquivos causar alta I/O carga ou sobrecarga de CPU, por exemplo, em grandes monorepos, ambientes com software antivírus ou projetos em que várias funções compartilham uma, `CodeUri` você pode desativá-la com a opção: `--no-watch`

```
$ sam local start-api --warm-containers eager --no-watch
```

Com `--no-watch`, as alterações no código e no modelo não são detectadas automaticamente. Pare e execute novamente o comando para obter as alterações.

Especifique uma imagem de contêiner para usar em suas funções de eventos do Lambda

Por padrão, o AWS SAMCLI usa imagens de base do Lambda do Amazon Elastic Container Registry (Amazon ECR) para invocar suas funções no local. Use a opção `--invoke-image` para referenciar uma imagem de contêiner personalizada. Este é um exemplo:

```
$ sam local start-api --invoke-image public.ecr.aws/sam/emu-python3.8
```

Você pode especificar a função a ser usada com a imagem personalizada do contêiner. Este é um exemplo:

```
$ sam local start-api --invoke-image Function1=amazon/aws/sam-cli-emulation-image-python3.8
```

Especifique um modelo para testar localmente

Para especificar um modelo para o AWS SAMCLI para referência, use a opção `--template`. Eles AWS SAMCLI carregarão apenas esse AWS SAM modelo e os recursos para os quais ele aponta. Este é um exemplo:

```
$ sam local start-api --template myTemplate.yaml
```

Especifique o ambiente de desenvolvimento do host da sua função de evento Lambda

Por padrão, o subcomando `sam local start-api` cria um servidor HTTP usando `localhost` com endereço IP `127.0.0.1`. Você pode personalizar esses valores se o ambiente de desenvolvimento local estiver isolado da máquina local.

Use a opção `--container-host` para especificar um host. Este é um exemplo:

```
$ sam local start-api --container-host host.docker.internal
```

Use a opção `--container-host-interface` para especificar o endereço IP da rede host à qual as portas do contêiner devem se vincular. Este é um exemplo:

```
$ sam local start-api --container-host-interface 0.0.0.0
```

Práticas recomendadas

Se seu aplicativo tiver um diretório `.aws-sam` executando `sam build`, certifique-se de executar o `sam build` sempre que atualizar o código da função. Em seguida, execute o `sam local start-api` para testar localmente seu código de função atualizado.

O teste local é uma ótima solução para desenvolvimento e teste rápidos antes da implantação na nuvem. No entanto, os testes locais não validam tudo, como permissões entre seus recursos na nuvem. Tanto quanto possível, teste seus aplicativos na nuvem. Recomendamos [usar o `sam sync`](#) para acelerar seus fluxos de trabalho de testes na nuvem.

Saiba mais

Para obter uma lista de todas as opções `sam local start-api`, consulte [SAM API de inicialização local](#).

Introdução aos testes com start-lambda do local do sam

Use o AWS SAMCLI subcomando `sam local start-lambda` para invocar sua função de evento do Lambda por meio dos SDKs e. AWS CLI. Esse comando inicia um endpoint local que emula o Lambda.

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter uma lista de opções de comando `sam local start-lambda`, consulte [start-lambda do local do sam](#).

Note

`sam local start-lambda` suporta funções duráveis com recursos automáticos de verificação e repetição. As funções duráveis funcionam perfeitamente com o endpoint Lambda local.

Pré-requisitos

Para usar o `sam local start-lambda`, instale o AWS SAM CLI fazendo o seguinte:

- [AWS SAM Pré-requisitos](#).
- [Instale o AWS SAM CLI](#).

Antes de usar `sam local start-lambda`, recomendamos uma compreensão básica do seguinte:

- [Configurando o AWS SAM CLI](#).
- [Crie seu aplicativo em AWS SAM](#).
- [Introdução à construção com AWS SAM](#).
- [Introdução à implantação com AWS SAM](#).

Usando o Local do SAM start-lambda

Quando você executa `sam local start-lambda`, AWS SAMCLI pressupõe que seu diretório de trabalho atual seja o diretório raiz do seu projeto. O AWS SAM CLI, primeiro procurará um arquivo

`template.[yaml|yml]` dentro de uma subpasta `.aws-sam`. Se não for encontrado, o AWS SAMCLI procurará um `template.[yaml|yml]` arquivo em seu diretório de trabalho atual.

Para usar o Local do SAM `start-lambda`

1. No diretório raiz do seu projeto, execute o seguinte:

```
$ sam local start-lambda <options>
```

2. O AWS SAMCLI cria suas funções do Lambda em um contêiner Docker local. Em seguida, ele envia o endereço local para o endpoint do servidor HTTP. Este é um exemplo:

```
$ 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. Use os SDKs AWS CLI ou para invocar sua função Lambda localmente.

Veja a seguir um exemplo de como usar a AWS CLI:

```
$ aws lambda invoke --function-name "HelloWorldFunction" --endpoint-
url "http://127.0.0.1:3001" --no-verify-ssl out.txt

StatusCode: 200
(END)
```

Veja a seguir um exemplo usando o AWS SDK 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")
```

Opções

Especificar um modelo

Para especificar um modelo para o AWS SAMCLI para referência, use a opção `--template`. Eles AWS SAMCLI carregarão apenas esse AWS SAM modelo e os recursos para os quais ele aponta. Este é um exemplo:

```
$ sam local start-lambda --template myTemplate.yaml
```

Para obter mais informações sobre AWS SAM modelos, consulte [AWS SAM anatomia do modelo](#).

Práticas recomendadas

Se seu aplicativo tiver um diretório `.aws-sam` executando `sam build`, certifique-se de executar o `sam build` sempre que atualizar o código da função. Em seguida, execute o `sam local start-lambda` para testar localmente seu código de função atualizado.

O teste local é uma ótima solução para desenvolvimento e teste rápidos antes da implantação na nuvem. No entanto, os testes locais não validam tudo, como permissões entre seus recursos na nuvem. Tanto quanto possível, teste seus aplicativos na nuvem. Recomendamos [usar o sam sync](#) para acelerar seus fluxos de trabalho de testes na nuvem.

Saiba mais

Para obter uma lista de todas as opções `sam local start-lambda`, consulte [start-lambda do local do sam](#).

Invoque localmente as funções de eventos do Lambda com AWS SAM

Invocar localmente uma função de evento Lambda antes de testar ou implantar na nuvem pode ter vários benefícios. Isso permite que você teste a lógica da função mais rapidamente. Testar primeiro localmente reduz a probabilidade da identificação de problemas ao testar na nuvem ou durante a implantação, o que pode ajudar a evitar custos desnecessários. Além disso, os testes locais facilitam a depuração.

Você pode invocar a função do Lambda localmente usando o comando [SAM invocação local](#) e fornecendo o ID lógico da função e um arquivo de eventos. `sam local invoke` também aceita `stdin` como evento. Para obter mais informações eventos, consulte [Eventos](#) no Guia do desenvolvedor do AWS Lambda . Para obter informações sobre formatos de mensagens de eventos de diferentes AWS serviços, consulte [Usando AWS Lambda com outros serviços](#) no Guia do AWS Lambda desenvolvedor.

Note

Não é recomendável usar os recursos de invocação local do SAM CLI em código não confiável. Para ter um isolamento completo do seu ambiente local, execute o código diretamente no serviço Lambda.

Note

O `sam local invoke` comando corresponde ao comando AWS Command Line Interface (AWS CLI) [aws lambda invoke](#). Você pode usar qualquer um dos comandos para invocar uma função do Lambda.

Você deve executar o comando `sam local invoke` no diretório do projeto que contém a função que seja invocar.

Exemplos:

```
# 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
```

Arquivo de variável de ambiente

Para declarar localmente variáveis de ambiente que substituem os valores definidos em seus modelos, faça o seguinte:

1. Crie um JSON ou .env arquivo que contenha as variáveis de ambiente a serem substituídas.
2. Use o argumento `--env-vars` para substituir os valores definidos em seus modelos.

A `--env-vars` opção oferece suporte a dois formatos de arquivo. O formato do arquivo é detectado automaticamente com base no conteúdo do arquivo.

Declarando variáveis de ambiente com JSON

Para declarar variáveis de ambiente que se aplicam globalmente a todos os recursos, especifique um objeto `Parameters` como o seguinte:

```
{
  "Parameters": {
    "TABLE_NAME": "localtable",
    "BUCKET_NAME": "amzn-s3-demo-bucket",
    "STAGE": "dev"
  }
}
```

Para declarar variáveis de ambiente diferentes para cada recurso, especifique objetos para cada recurso da seguinte forma:

```
{
  "MyFunction1": {
    "TABLE_NAME": "localtable",
    "BUCKET_NAME": "amzn-s3-demo-bucket",
  },
  "MyFunction2": {
    "TABLE_NAME": "localtable",
    "STAGE": "dev"
  }
}
```

```
}  
}
```

Ao especificar objetos para cada recurso, você pode usar os seguintes identificadores, listados na ordem da maior para a menor precedência:

1. `logical_id`
2. `function_id`
3. `function_name`
4. Identificador de caminho completo

Você pode usar os dois métodos anteriores para declarar variáveis de ambiente juntos em um único arquivo. Ao fazer isso, as variáveis de ambiente que você forneceu para recursos específicos têm precedência sobre as variáveis de ambiente globais.

Salve suas variáveis de ambiente em um arquivo JSON, como `env.json`.

Declarando variáveis de ambiente com arquivos `.env`

Você também pode usar um `.env` arquivo para declarar variáveis de ambiente. As variáveis declaradas em um `.env` arquivo se aplicam globalmente a todas as funções, equivalentes ao `Parameters` objeto no formato JSON.

```
TABLE_NAME=localtable  
BUCKET_NAME=amzn-s3-demo-bucket  
STAGE=dev
```

O `.env` formato suporta comentários (linhas começando com `#`) e valores entre aspas.

Note

O `.env` formato só oferece suporte a variáveis de ambiente globais. Para declarar variáveis de ambiente específicas da função, use o formato JSON.

Substituindo valores de variáveis de ambiente

Para substituir as variáveis de ambiente pelas definidas em seu arquivo de variáveis de ambiente, use o `--env-vars` argumento com os `start-api` comandos `invoke` ou. Por exemplo:

```
# Using a JSON file
sam local invoke --env-vars env.json

# Using a .env file
sam local invoke --env-vars .env
```

Layers

Se seu aplicativo incluir camadas, para obter informações sobre como depurar problemas com camadas em seu host local, consulte [Aumente a eficiência usando camadas Lambda com AWS SAM](#).

Saiba mais

Para obter um exemplo prático de invocar funções localmente, consulte o [Módulo 2 - Executar localmente](#) no The Complete Workshop. AWS SAM

Execute localmente o API Gateway com AWS SAM

A execução do Amazon API Gateway localmente pode ter vários benefícios. Por exemplo, executar o API Gateway localmente permite que você teste os endpoints da API localmente antes da implantação na AWS nuvem. Se você testar primeiro no local, muitas vezes poderá reduzir os testes e o desenvolvimento na nuvem, o que pode ajudar a reduzir os custos. Além disso, a execução no local facilita a depuração.

Para iniciar uma instância local do API Gateway que você pode usar para testar a funcionalidade de solicitação/resposta HTTP, use o comando. [SAM API de inicialização local](#) AWS SAMCLI Essa funcionalidade apresenta recarregamento a quente para que você possa desenvolver e iterar rapidamente suas funções.

Note

O recarregamento a quente ocorre quando somente os arquivos alterados são atualizados e o estado do aplicativo permanece o mesmo. Por outro lado, o recarregamento dinâmico ocorre quando todo o aplicativo é atualizado e o estado do aplicativo é perdido.

Para obter instruções sobre como usar o comando `sam local start-api`, consulte [Introdução aos testes com SAM API de inicialização local](#).

Por padrão, AWS SAM usa integrações de AWS Lambda proxy e oferece suporte a ambos `HttpApi` e aos tipos de `Api` recursos. Para obter mais informações sobre integrações de proxy para tipos de `HttpApi` recursos, consulte Como [trabalhar com integrações de AWS Lambda proxy para HTTP APIs](#) no Guia do desenvolvedor do API Gateway. Para obter mais informações sobre integrações de proxy com tipos de recursos `Api`, consulte [Compreender a integração de proxy do Lambda do API Gateway](#) no Guia do desenvolvedor do API Gateway.

Exemplo:

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

AWS SAM encontra automaticamente todas as funções em seu AWS SAM modelo que tenham fontes `HttpApi` de `Api` eventos definidas. Em seguida, ele monta a função nos caminhos HTTP definidos.

No exemplo de `Api` abaixo, a função `Ratings` pode montar `ratings.py:handler()` em `/ratings` para solicitações GET.

```
Ratings:
  Type: AWS::Serverless::Function
  Properties:
    Handler: ratings.handler
    Runtime: python3.9
    Events:
      Api:
        Type: Api
        Properties:
          Path: /ratings
          Method: get
```

Este é um exemplo de resposta `Api`:

```
// 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 você modificar o código da sua função, execute o comando `sam build` para `sam local start-api` detectar suas alterações.

Arquivo de variável de ambiente

Para declarar localmente variáveis de ambiente que substituem os valores definidos em seus modelos, faça o seguinte:

1. Crie um arquivo JSON que contenha as variáveis de ambiente a serem substituídas.
2. Use o argumento `--env-vars` para substituir os valores definidos em seus modelos.

Declarar variáveis de ambiente

Para declarar variáveis de ambiente que se aplicam globalmente a todos os recursos, especifique um objeto `Parameters` como o seguinte:

```
{
  "Parameters": {
    "TABLE_NAME": "localtable",
    "BUCKET_NAME": "amzn-s3-demo-bucket",
    "STAGE": "dev"
  }
}
```

Para declarar variáveis de ambiente diferentes para cada recurso, especifique objetos para cada recurso da seguinte forma:

```
{
  "MyFunction1": {
    "TABLE_NAME": "localtable",
    "BUCKET_NAME": "amzn-s3-demo-bucket",
  },
  "MyFunction2": {
    "TABLE_NAME": "localtable",
    "STAGE": "dev"
  }
}
```

Ao especificar objetos para cada recurso, você pode usar os seguintes identificadores, listados na ordem da maior para a menor precedência:

1. `logical_id`
2. `function_id`
3. `function_name`
4. Identificador de caminho completo

Você pode usar os dois métodos anteriores para declarar variáveis de ambiente juntos em um único arquivo. Ao fazer isso, as variáveis de ambiente que você forneceu para recursos específicos têm precedência sobre as variáveis de ambiente globais.

Salve suas variáveis de ambiente em um arquivo JSON, como `env.json`.

Substituindo valores de variáveis de ambiente

Para substituir as variáveis de ambiente pelas definidas em seu arquivo JSON, use o argumento `--env-vars` com os comandos `invoke` ou `start-api`. Por exemplo:

```
$ sam local start-api --env-vars env.json
```

Layers

Se seu aplicativo incluir camadas, para obter informações sobre como depurar problemas com camadas em seu host local, consulte [Aumente a eficiência usando camadas Lambda com AWS SAM](#).

Introdução aos testes em nuvem com evento de teste remoto do sam

Use o AWS Serverless Application Model comando Command Line Interface (AWS SAM CLI) `sam remote test-event` para acessar e gerenciar eventos de teste compartilháveis para suas AWS Lambda funções de eventos.

Para saber mais sobre eventos de teste compartilháveis, consulte [Eventos de teste compartilháveis](#) no Guia do desenvolvedor do AWS Lambda .

Tópicos

- [Configure o AWS SAM CLI para usar evento de teste remoto do sam](#)

- [Usar o evento de teste remoto do sam command](#)
- [Usando eventos de teste compartilháveis](#)
- [Usando eventos de teste compartilháveis](#)

Pré-requisitos

Para usar o `sam remote test-event`, instale o AWS SAM CLI fazendo o seguinte:

- [AWS SAM Pré-requisitos](#) .
- [Instale o AWS SAM CLI](#).

Se você já tem o AWS SAM CLI instalado, recomendamos atualizar para a versão mais recente da AWS SAMCLI versão. Para saber mais, consulte [Atualizar o AWS SAM CLI](#).

Antes de usar `sam remote test-event`, recomendamos uma compreensão básica do seguinte:

- [Configurando o AWS SAM CLI](#).
- [Crie seu aplicativo em AWS SAM](#).
- [Introdução à construção com AWS SAM](#).
- [Introdução à implantação com AWS SAM](#).
- [Introdução ao uso mesma sincronização para sincronizar com Nuvem AWS](#).

Configure o AWS SAM CLI para usar evento de teste remoto do sam

Conclua as seguintes etapas de configuração para usar o AWS SAM CLI `sam remote test-event` comando:

1. Configure o AWS SAM CLI para usar seu Conta da AWS — Os eventos de teste compartilháveis do Lambda podem ser acessados e gerenciados por usuários dentro do mesmo. Conta da AWS Para configurar o AWS SAM CLI para usar o seu Conta da AWS, consulte [Configurando o AWS SAM CLI](#).
2. Configurar permissões para eventos de teste compartilháveis – Para acessar e gerenciar eventos de teste compartilháveis, você deve ter as permissões adequadas. Para saber mais, consulte [Eventos de teste compartilháveis](#) no Guia do desenvolvedor do AWS Lambda .

Usar o evento de teste remoto do sam command

O AWS SAM CLI `sam remote test-event` comando fornece os seguintes subcomandos que você pode usar para acessar e gerenciar seus eventos de teste compartilháveis:

- `delete`— Exclua um evento de teste compartilhável do registro de EventBridge esquemas da Amazon.
- `get`— Obtenha um evento de teste compartilhável do registro do EventBridge esquema.
- `list`— Liste os eventos de teste compartilháveis existentes para uma função do registro do EventBridge esquema.
- `put`— Salve um evento de um arquivo local no registro do EventBridge esquema.

Para listar esses subcomandos usando o AWS SAM CLI, execute o seguinte:

```
$ sam remote test-event --help
```

Excluindo eventos de teste compartilháveis

Você pode excluir um evento de teste compartilhável usando o subcomando `delete` junto com o seguinte:

- Forneça o nome do evento compartilhável de teste a ser excluído.
- Forneça uma identificação aceitável da função do Lambda associada ao evento.
- Se você estiver fornecendo o ID lógico da função Lambda, também deverá fornecer o nome da AWS CloudFormation pilha associado à função Lambda.

Este é um exemplo:

```
$ sam remote test-event delete HelloWorldFunction --stack-name sam-app --name demo-event
```

Para obter uma lista de opções a serem usadas com o subcomando `delete`, consulte [exclusão remota de eventos de teste do sam](#). Você também pode executar o seguinte a partir do AWS SAM CLI:

```
$ sam remote test-event delete --help
```

Obtendo eventos de teste compartilháveis

Você pode obter um evento de teste compartilhável do registro do EventBridge esquema usando o `get` subcomando junto com o seguinte:

- Forneça o nome do evento de teste compartilhável a ser obtido.
- Forneça uma identificação aceitável da função do Lambda associada ao evento.
- Se você estiver fornecendo o ID lógico da função Lambda, também deverá fornecer o nome da AWS CloudFormation pilha associado à função Lambda.

Veja a seguir um exemplo que obtém um evento de teste compartilhável chamado `demo-event` associado à função do Lambda `HelloWorldFunction` da pilha `sam-app`. Esse comando imprimirá o evento em seu console.

```
$ sam remote test-event get HelloWorldFunction --stack-name sam-app --name demo-event
```

Para obter um evento de teste compartilhável e salvá-lo em sua máquina local, use a opção `--output-file` e forneça um nome e um caminho de arquivo. Veja a seguir um exemplo que salva `demo-event` como `demo-event.json` no diretório de trabalho atual:

```
$ sam remote test-event get HelloWorldFunction --stack-name sam-app --name demo-event --output-file demo-event.json
```

Para obter uma lista de opções a serem usadas com o subcomando `get`, consulte [sam remote test-event get](#). Você também pode executar o seguinte a partir do AWS SAM CLI:

```
$ sam remote test-event get --help
```

Listando eventos de teste compartilháveis

Você pode listar todos os eventos de teste compartilháveis para uma função do Lambda específica no registro do esquema. Use o subcomando `list` junto com o seguinte:

- Forneça um ID aceitável da função do Lambda associada aos eventos.
- Se você estiver fornecendo o ID lógico da função Lambda, também deverá fornecer o nome da CloudFormation pilha associado à função Lambda.

Veja a seguir um exemplo que obtém uma lista de todos os eventos de teste compartilháveis associados à função do Lambda HelloWorldFunction da pilha sam-app:

```
$ sam remote test-event list HelloWorldFunction --stack-name sam-app
```

Para obter uma lista de opções a serem usadas com o subcomando list, consulte [lista de eventos de teste remotos sam](#). Você também pode executar o seguinte a partir do AWS SAM CLI:

```
$ sam remote test-event list --help
```

Salvando eventos de teste compartilháveis

Você pode salvar eventos de teste compartilháveis no registro do EventBridge esquema. Use o subcomando put junto com o seguinte:

- Forneça uma ID aceitável da função do Lambda associada ao evento de teste compartilhável.
- Forneça um nome para o evento compartilhável de teste.
- Forneça o caminho e o nome do arquivo para o evento local a ser carregado.

Veja a seguir um exemplo que salva o evento local demo-event.json como demo-event e o associa à função do Lambda HelloWorldFunction da pilha sam-app:

```
$ sam remote test-event put HelloWorldFunction --stack-name sam-app --name demo-event --file demo-event.json
```

Se existir um evento de teste compartilhável com o mesmo nome no registro do EventBridge esquema, AWS SAM CLI ele não o substituirá. Para sobrescrever, adicione a opção --force ao seu comando.

Para obter uma lista de opções a serem usadas com o subcomando put, consulte [mesma saída remota de eventos de teste](#). Você também pode executar o seguinte a partir do AWS SAM CLI:

```
$ sam remote test-event put --help
```

Usando eventos de teste compartilháveis

Use eventos de teste compartilháveis para testar suas funções do Lambda com Nuvem AWS `sam remote invoke` o comando. Para saber mais, consulte [Transmita eventos de teste compartilháveis para uma função de evento Lambda na nuvem](#).

Usando eventos de teste compartilháveis

Este tópico contém exemplos de como você pode gerenciar e usar eventos de teste compartilháveis.

Obtenha um evento de teste compartilhável, modifique-o e use-o

Você pode obter um evento de teste compartilhável do registro do EventBridge esquema, modificá-lo localmente e usar o evento de teste local com sua função Lambda no. Nuvem AWS Este é um exemplo:

1. Recupere o evento de teste compartilhável – Use o subcomando `sam remote test-event get` para recuperar um evento de teste compartilhável para uma função do Lambda específica e salvá-lo localmente:

```
$ sam remote test-event get HelloWorldFunction --stack-name sam-app --name demo-event --output-file demo-event.json
```

2. Modificar o evento de teste compartilhável – Use um editor de texto de sua escolha para modificar o evento de teste compartilhável.
3. Use o evento de teste compartilhável – Use o comando `sam remote invoke` e forneça o caminho do arquivo e o nome do evento com `--event-file`:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --event-file demo-event.json
```

Obtenha um evento de teste compartilhável, modifique-o, carregue-o e use-o

Você pode obter um evento de teste compartilhável do registro do EventBridge esquema, modificá-lo localmente e carregá-lo. Em seguida, você pode passar o evento de teste compartilhável diretamente para sua função do Lambda no Nuvem AWS. Este é um exemplo:

1. Recupere o evento de teste compartilhável – Use o subcomando `sam remote test-event get` para recuperar um evento de teste compartilhável para uma função do Lambda específica e salvá-lo localmente:

```
$ sam remote test-event get HelloWorldFunction --stack-name sam-app --name demo-event --output-file demo-event.json
```

2. Modificar o evento de teste compartilhável – Use um editor de texto de sua escolha para modificar o evento de teste compartilhável.
3. Carregar o evento de teste compartilhável — Use o `sam remote test-event put` subcomando para carregar e salvar o evento de teste compartilhável no registro do EventBridge esquema. Neste exemplo, usamos a opção `--force` de sobrescrever uma versão mais antiga do nosso teste compartilhável:

```
$ sam remote test-event put HelloWorldFunction --stack-name sam-app --name demo-event --file demo-event.json --force
```

4. Passe o evento de teste compartilhável para sua função do Lambda – Use o comando `sam remote invoke` para passar o evento de teste compartilhável diretamente para sua função do Lambda no Nuvem AWS:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --test-event-name demo-event
```

Introdução aos testes na nuvem com invocação remota do sam

Use o AWS Serverless Application Model comando Command Line Interface (AWS SAM CLI) `sam remote invoke` para interagir com AWS os recursos suportados no Nuvem AWS. Você pode usar `sam remote invoke` para invocar os seguintes recursos:

- Amazon Kinesis Data Streams – Envie registros de dados para aplicativos do Kinesis Data Streams.
- AWS Lambda— Invoque e transmita eventos para suas funções de eventos Lambda.
- Amazon Simple Queue Service (Amazon SQS) — Envie mensagens para filas do Amazon SQS.
- AWS Step Functions – Invoque máquinas de estado do Step Functions para iniciar a execução.

Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

Para obter um exemplo de uso do `sam remote invoke` durante um fluxo de trabalho de desenvolvimento típico, consulte [Etapa 5: Interaja com sua função no Nuvem AWS](#).

Tópicos

- [Usando o comando de invocação remota do sam](#)
- [Usando o comando de invocação remota do sam](#)
- [Configure o arquivo de configuração do seu projeto](#)
- [Exemplos](#)
- [Links relacionados](#)

Pré-requisitos

Para usar `sam remote invoke`, instale o AWS SAM CLI fazendo o seguinte:

- [AWS SAM Pré-requisitos](#).
- [Instale o AWS SAM CLI](#).

Também recomendamos atualizar para a versão mais recente do AWS SAMCLI. Para saber mais, consulte [Atualizar o AWS SAM CLI](#).

Antes de usar `sam remote invoke`, recomendamos uma compreensão básica do seguinte:

- [Configurando o AWS SAM CLI](#).
- [Crie seu aplicativo em AWS SAM](#).
- [Introdução à construção com AWS SAM](#).
- [Introdução à implantação com AWS SAM](#).
- [Introdução ao uso mesma sincronização para sincronizar com Nuvem AWS](#).

Usando o comando de invocação remota do sam

Antes de usar esse comando, seu recurso deve ser implantado no Nuvem AWS.

Use a seguinte estrutura de comando e execute no diretório raiz do seu projeto:

```
$ sam remote invoke <arguments> <options>
```

Note

Esta página mostrará as opções fornecidas no prompt de comando. Você também pode configurar opções no arquivo de configuração do seu projeto em vez de passá-las no prompt de comando. Para saber mais, [Definir configurações do projeto](#).

Para obter uma descrição dos argumentos e opções `sam remote invoke`, consulte [invocação remota do sam](#).

Usar com o Kinesis Data Streams

Você pode enviar registros de dados para um aplicativo do Kinesis Data Streams. Eles AWS SAM CLI enviarão seu registro de dados e retornarão um ID de fragmento e um número de sequência. Este é um exemplo:

```
$ 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"  
}%
```

Para enviar um registro de dados

1. Forneça um valor de ID de recurso como argumento para seu aplicativo Kinesis Data Streams. Para obter informações sobre IDs de recursos válidos, consulte [ID de recurso](#).
2. Forneça o registro de dados como um evento para enviar ao seu aplicativo Kinesis Data Streams. Você pode fornecer o evento na linha de comando usando a opção `--event` ou a partir de um arquivo usando `--event-file`. Se você não fornecer um evento, AWS SAM CLI ele enviará um evento vazio.

Usando com funções de eventos Lambda

Você pode invocar uma função do Lambda na nuvem e transmitir um evento vazio ou fornecer um evento na linha de comando ou a partir de um arquivo. Eles AWS SAM CLI invocarão sua função Lambda e retornarão sua resposta. Este é um exemplo:

```
$ 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\"},"}%
```

Para invocar uma função de evento Lambda

1. Forneça um valor de ID de recurso como argumento para sua função do Lambda. Para obter informações sobre IDs de recursos válidos, consulte [ID de recurso](#).
2. Forneça um evento para enviar para sua função do Lambda. Você pode fornecer o evento na linha de comando usando a opção `--event` ou a partir de um arquivo usando `--event-file`. Se você não fornecer um evento, AWS SAM CLI ele enviará um evento vazio.

Funções de eventos Lambda configuradas com streaming de resposta

O comando `sam remote invoke` é compatível com funções do Lambda que são configuradas para transmitir respostas. Você pode configurar uma função Lambda para transmitir respostas usando a [FunctionUrlConfig](#) propriedade em seus AWS SAM modelos. Quando você usa `sam remote invoke`, AWS SAM CLI detecta automaticamente sua configuração do Lambda e invoca com streaming de resposta.

Para ver um exemplo, consulte [Invocar uma função de evento Lambda configurada para transmitir respostas](#).

Transmita eventos de teste compartilháveis para uma função de evento Lambda na nuvem

Eventos de teste compartilháveis são eventos de teste que você pode compartilhar com outros na mesma Conta da AWS. Para saber mais, consulte [Eventos de teste compartilháveis](#) no AWS Lambda Guia do desenvolvedor.

Acessar e gerenciar eventos compartilháveis de teste

Você pode usar o AWS SAM CLI `sam remote test-event` comando para acessar e gerenciar eventos de teste compartilháveis. Por exemplo, você pode usar `sam remote test-event` para fazer o seguinte:

- Recupere eventos de teste compartilháveis do registro de EventBridge esquemas da Amazon.
- Modifique eventos de teste compartilháveis localmente e carregue-os no registro do EventBridge esquema.
- Exclua eventos de teste compartilháveis do registro do EventBridge esquema.

Para saber mais, consulte [Introdução aos testes em nuvem com evento de teste remoto do sam](#).

Transmita um evento de teste compartilhável para uma função de evento Lambda na nuvem

Para passar um evento de teste compartilhável do registro do EventBridge esquema para sua função Lambda na nuvem, use a `--test-event-name` opção e forneça o nome do evento de teste compartilhável. Este é um exemplo:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --test-event-name demo-event
```

Se você salvar o evento de teste compartilhável localmente, poderá usar a opção `--event-file` e fornecer o caminho do arquivo e o nome do evento de teste local. Este é um exemplo:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --event-file demo-event.json
```

Usar o com o Amazon SQS

Você pode enviar mensagens para filas do Amazon SQS. O AWS SAM CLI retorna o seguinte:

- ID da mensagem
- MD5 do corpo da mensagem
- Metadados de resposta

Este é um exemplo:

```
$ sam remote invoke MySqsQueue --stack-name sqs-example -event hello
```

Sending message to SQS queue MySqsQueue

```
{  
  "MD5OfMessageBody": "5d41402abc4b2a76b9719d911017c592",  
  "MessageId": "05c7af65-9ae8-4014-ae28-809d6d8ec652"  
}%
```

Para enviar uma mensagem

1. Forneça um valor de ID de recurso como argumento para a fila do Amazon SQS. Para obter informações sobre IDs de recursos válidos, consulte [ID de recurso](#).
2. Forneça um evento para enviar para sua fila do Amazon SQS. Você pode fornecer o evento na linha de comando usando a opção `--event` ou a partir de um arquivo usando `--event-file`. Se você não fornecer um evento, AWS SAM CLI ele enviará um evento vazio.

Usar com o Step Functions

Você pode invocar uma máquina de estado do Step Functions para iniciar a execução. O AWS SAM CLI aguardará a conclusão do fluxo de trabalho da máquina de estado e retornará uma saída da última etapa da execução. Este é um exemplo:

```
$ sam remote invoke HelloWorldStateMachine --stack-name state-machine-example --  
event '{"is_developer": true}'
```

Invoking Step Function HelloWorldStateMachine

```
"Hello Developer World"%
```

Para invocar uma máquina de estado

1. Forneça um valor de ID de recurso como argumento para a máquina de estado do Step Functions. Para obter informações sobre IDs de recursos válidos, consulte [ID de recurso](#).
2. Forneça um evento para enviar para sua máquina estadual. Você pode fornecer o evento na linha de comando usando a opção `--event` ou a partir de um arquivo usando `--event-file`. Se você não fornecer um evento, AWS SAM CLI ele enviará um evento vazio.

Usando o comando de invocação remota do sam

Esta seção aborda algumas das principais opções que você pode usar com o comando `sam remote invoke`. Para obter uma lista completa das opções, consulte [invocação remota do sam](#).

Passe um evento para seu recurso

Use as seguintes opções para transmitir eventos para seus recursos na nuvem:

- `--event` - Passe um evento na linha de comando.
- `--event-file` - Passe um evento de um arquivo.

Exemplos do Lambda

Use `--event` para passar um evento na linha de comando como um valor de string:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --event '{"message": "hello!"}'
```

Invoking Lambda Function HelloWorldFunction

```
START RequestId: b992292d-1fac-4aa2-922a-c9dc5c6fceb Version: $LATEST
END RequestId: b992292d-1fac-4aa2-922a-c9dc5c6fceb
REPORT RequestId: b992292d-1fac-4aa2-922a-c9dc5c6fceb 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!\"}"}
```

Use `--event-file` para transmitir um evento de um arquivo e fornecer o caminho para o arquivo:

```
$ 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\"}}%
```

Passe um 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\"}}%
```

Configurar o AWS SAM CLI resposta de saída

Quando você invoca um recurso compatível com `sam remote invoke`, AWS SAMCLI retorna uma resposta que contém o seguinte:

- Metadados da solicitação – Metadata associados à solicitação. Isso inclui o ID da solicitação e o horário de início da solicitação.
- Resposta do recurso - A resposta do seu recurso após ser invocado na nuvem.

Você pode usar a `--output` opção para configurar a resposta AWS SAM CLI de saída. Os seguintes valores de opção estão disponíveis:

- `json` - Os metadados e a resposta do recurso são retornados em uma estrutura JSON. A resposta contém a saída SDK completa.
- `text` - Os metadados são retornados na estrutura do texto. A resposta do recurso é retornada no formato de saída do recurso.

A seguir, veja um exemplo de uma saída json:

```
$ 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": "$LATEST",
      "x-amz-log-result":
        "U1RBULQgUmVxdWVzdElkOiAzYmRmOWEzMC03NzZkLTRhOTAtOTRhNi00Y2NjYzBmYzdiNDEgVmVyc2lvbJogJExBVEVTV",
      "x-amzn-trace-id":
        "root=1-64908d42-17dab270273fcc6b527dd6b8;samled=0;lineage=2301f8dc:0"
    },
    "RetryAttempts": 0
  },
  "StatusCode": 200,
  "LogResult":
    "U1RBULQgUmVxdWVzdElkOiAzYmRmOWEzMC03NzZkLTRhOTAtOTRhNi00Y2NjYzBmYzdiNDEgVmVyc2lvbJogJExBVEVTV",
  "ExecutedVersion": "$LATEST",
  "Payload": "{\"statusCode\":200,\"body\":{\"message\":\"hello world\"}}"}
}%
```

Quando você especifica uma saída json, toda a resposta é retornada para stdout. Este é um exemplo:

```
$ 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": "$LATEST",
      "x-amz-log-result":
        "U1RBULqgUmVxdWVzdElkOiBkMzBkMjgwZi04MTg4LTQzNzItYmM5NC1jZTBmMTYwM2I2YmIgVmVyc2lvbjoyJExBVEVTV",
      "x-amzn-trace-id":
        "root=1-649091fc-771473c7778689627a6122b7;sampld=0;lineage=2301f8dc:0"
    },
    "RetryAttempts": 0
  },
  "StatusCode": 200,
  "LogResult":
    "U1RBULqgUmVxdWVzdElkOiBkMzBkMjgwZi04MTg4LTQzNzItYmM5NC1jZTBmMTYwM2I2YmIgVmVyc2lvbjoyJExBVEVTV",
  "ExecutedVersion": "$LATEST",
  "Payload": "{\"statusCode\":200,\"body\":\":{\"\\\"message\\\":\\\"hello world\\\"}\\\"}"
}%
```

A seguir, veja um exemplo de uma saída text:

```
$ 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 você especifica uma saída `text`, a saída de runtime da função do Lambda (por exemplo, registros) é retornada para `stderr`. A carga útil da função do Lambda é retornada para `stdout`. Este é um exemplo:

```
$ 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\"}}}%
```

Personalização Boto3 parameters

Poissam `remote invoke`, AWS SAM CLI ele utiliza o AWS SDK para Python (Boto3) para interagir com seus recursos na nuvem. Você pode usar a opção `--parameter` para personalizar os parâmetros Boto3. Para ver uma lista dos parâmetros, consulte [--parameter](#).

Exemplos

Invoque uma função do Lambda para validar os valores dos parâmetros e verificar as permissões:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --  
parameter InvocationType="DryRun"
```

Use a opção **--parameter** várias vezes em um único comando para fornecer vários parâmetros:

```
$ sam remote invoke HelloWorldFunction --stack-name sam-app --  
parameter InvocationType="Event" --parameter LogType="None"
```

Outras opções

Para obter uma lista completa das opções `sam remote invoke`, consulte [invocação remota do sam](#).

Configure o arquivo de configuração do seu projeto

Para configurar `sam remote invoke` em seu arquivo de configuração, use `remote_invoke` em sua tabela. Veja a seguir um exemplo de arquivo `samconfig.toml` que configura valores padrão para o comando `sam remote invoke`.

```
...  
version =0.1  
  
[default]  
...  
[default.remote_invoke.parameters]  
stack_name = "cloud-app"  
event = '{"message": "Hello!"}'
```

Exemplos

Para ver um exemplo básico de uso `sam remote invoke`, consulte [Teste de AWS Lambda funções com AWS SAM controle remoto](#) no blog de AWS computação.

Exemplos do Kinesis Data Streams

Exemplos básicos

Envie um registro de dados para um aplicativo Kinesis Data Streams a partir de um arquivo. O aplicativo Kinesis Data Streams é identificado fornecendo um ARN para o ID do recurso:


```
$ sam remote invoke arn:aws:kinesis:us-west-2:01234567890:stream/kinesis-example-KinesisStream-BgnLcAey4xUQ --event-file event.json
```

Envie um evento fornecido na linha de comando para um aplicativo do 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"
}%
```

Obtenha a ID física do aplicativo Kinesis Data Streams. Em seguida, forneça um evento na linha de 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"
}%
```

Forneça uma string JSON na linha de comando como um 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"
}%
```

Envie um evento vazio para o aplicativo Kinesis Data Streams:

```
$ sam remote invoke KinesisStream --stack-name kinesis-example
```

Putting record to Kinesis data stream KinesisStream

```
{
  "ShardId": "shardId-000000000000",
  "SequenceNumber": "49646251411914806775980904866469008589597168190416224258"
}%
```

Retorne a AWS SAM CLI resposta no 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": "Q3yBcgTwtPaQTV26IKclbECmZikUY0zKY+CzcxA84ZHgCkc5T2N/ITWg6RP0QcWw8Gn0tNPcEJBEHyVVqboJAPgCritqsvCu",
      "date": "Thu, 09 Nov 2023 18:13:10 GMT",
      "content-type": "application/x-amz-json-1.1",

```

```

    "content-length": "110"
  },
  "RetryAttempts": 0
}
}%

```

Retorne a saída JSON para 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": "4964625141191480677598090639777867595039988349006774274",
  "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
  }
}%

```

Exemplos do Lambda

Exemplos básicos

Invoke uma função do Lambda fornecendo o ARN como ID do recurso:

```

$ sam remote invoke arn:aws:lambda:us-west-2:012345678910:function:sam-app-HelloWorldFunction-ohRFEn2RuAvp

```

Invoque uma função do Lambda fornecendo a ID lógica como uma ID de recurso:

Você também deve fornecer o nome da CloudFormation pilha usando a `--stack-name` opção. Este é um exemplo:

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

Se seu aplicativo contiver uma única função do Lambda, você não precisará especificar sua ID lógica. Você pode fornecer somente a opção `--stack-name`. Este é um exemplo:

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

Invoque uma função do Lambda fornecendo o ID físico como ID do recurso:

O ID físico é criado quando você implanta usando CloudFormation.

```
$ sam remote invoke sam-app-HelloWorldFunction-TZvxQRFNv0k4
```

Invoque uma função do Lambda de uma pilha secundária:

Neste exemplo, nosso aplicativo contém a seguinte estrutura de diretórios:

```
lambda-example
### childstack
#   ### function
#   #   ### __init__.py
#   #   ### app.py
#   #   ### requirements.txt
#   ### template.yaml
### events
#   ### event.json
### samconfig.toml
### template.yaml
```

Para invocar a função do Lambda do nosso childstack, executamos o seguinte:

```
$ 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\": {}}\"}%
```

Invocar uma função de evento Lambda configurada para transmitir respostas

Neste exemplo, usamos AWS SAM CLI para inicializar um novo aplicativo sem servidor que contém uma função do Lambda configurada para transmitir sua resposta. Implantamos nosso aplicativo no Nuvem AWS e usamos o `sam remote invoke` para interagir com nossa função na nuvem.

Começamos executando o comando `sam init` para criar um novo aplicativo sem servidor. Seleccionamos o modelo de início rápido do Lambda Response Streaming e nomeamos nosso aplicativo `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 SAMCLICria nosso projeto com a seguinte estrutura:

```
lambda-streaming-nodejs-app
### README.md
### __tests__
#   ### unit
#       ### index.test.js
```

```
### package.json
### samconfig.toml
### src
# ### index.js
### template.yaml
```

Veja a seguir um exemplo de nosso código de função do 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(`

${loremIpsum1}</p>`);
    await new Promise(r => setTimeout(r, 1000));

    responseStream.write("<p>DONE!</p>");
    responseStream.write("</html>");
    responseStream.end();
  }
);


```

Este é um exemplo de nosso arquivo `template.yaml`. O streaming de resposta para nossa função do Lambda é configurado usando a propriedade `FunctionUrlConfig`.

```

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

```


Normalmente, você pode usar `sam build` e `sam deploy --guided` para criar e implantar um aplicativo de produção. Neste exemplo, assumiremos um ambiente de desenvolvimento e usaremos o comando `sam sync` para criar e implantar nosso aplicativo.

Note

O comando `sam sync` é recomendado para ambientes de desenvolvimento. Para saber mais, consulte [Introdução ao uso mesma sincronização para sincronizar com Nuvem AWS](#).

Antes de executar `sam sync`, verificamos se nosso projeto está configurado corretamente em nosso arquivo `samconfig.toml`. Mais importante ainda, verificamos os valores de `stack_name` e `watch`. Com esses valores especificados em nosso arquivo de configuração, não precisamos fornecê-los na linha de 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"
```

Em seguida, executamos `sam sync` para criar e implantar nosso aplicativo. Como a opção `--watch` está configurada em nosso arquivo de configuração, AWS SAMCLI criará o nosso aplicativo, implantará o nosso aplicativo e observará as alterações.

```
$ 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_pt15g6_80000gr/T/tmpavrizdhgp.
Execute the following command to deploy the packaged template
sam deploy --template-file /var/folders/45/5ct135bx3fn2551_pt15g6_80000gr/T/
tmpavrizdhgp --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::Stack	lambda-streaming-nodejs-app	
succeeded	ack		
CREATE_IN_PROGRESS	AWS::IAM::Role	StreamingFunctionRole	-
CREATE_IN_PROGRESS	AWS::CloudFormation::Stack	AwsSamAutoDependencyLayerNestedStack	-
ack			
CREATE_IN_PROGRESS	AWS::IAM::Role	StreamingFunctionRole	Resource creation
Initiated			
CREATE_IN_PROGRESS	AWS::CloudFormation::Stack	AwsSamAutoDependencyLayerNestedStack	Resource creation
ack			
Initiated			
CREATE_COMPLETE	AWS::IAM::Role	StreamingFunctionRole	-

CREATE_COMPLETE	AWS::CloudFormation::Stack	AwsSamAutoDependencyLayerNestedStack	-
CREATE_IN_PROGRESS	AWS::Lambda::Function	StreamingFunction	-
CREATE_IN_PROGRESS creation	AWS::Lambda::Function	StreamingFunction	Resource
Initiated			
CREATE_COMPLETE	AWS::Lambda::Function	StreamingFunction	-
CREATE_IN_PROGRESS	AWS::Lambda::Url	StreamingFunctionUrl	-
CREATE_IN_PROGRESS creation	AWS::Lambda::Url	StreamingFunctionUrl	Resource
Initiated			
CREATE_COMPLETE	AWS::Lambda::Url	StreamingFunctionUrl	-
CREATE_COMPLETE	AWS::CloudFormation::Stack	lambda-streaming-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-gUmh0833A0vZ		
Key	StreamingFunctionURL		
Description	Streaming Lambda Function URL		

```
Value                https://wxgkcc2dyntgtrwhf2dgdcvy1u0rnnof.lambda-url.us-
west-2.on.aws/
-----
```

Stack creation succeeded. Sync infra completed.

Infra sync completed.

Agora que nossa função está implantada na nuvem, podemos usar `sam remote invoke` para interagir com nossa função. O AWS SAM CLI detecta automaticamente que nossa função está configurada para streaming de resposta e imediatamente começa a gerar uma resposta transmitida de nossa função em tempo real.

```
$ 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 modificamos nosso código de função, AWS SAM CLI detecta instantaneamente e implanta imediatamente nossas alterações. Aqui está um exemplo da saída AWS SAMCLI após as alterações serem feitas em nosso código de função:

```
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.
```

Agora podemos usar `sam remote invoke` novamente para interagir com nossa função na nuvem e testar nossas alterações.

Exemplos do SQS

Exemplos básicos

Invoque uma fila do Amazon SQS fornecendo o ARN como ID do recurso:

```
$ 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
}
}%

```

Exemplos de Step Functions

Exemplos básicos

Invoke uma máquina de estado fornecendo sua ID física como ID de recurso:

Primeiro, usamos `sam list resources` para obter nossa identificação física:

```

$ 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"
  }
]

```

Em seguida, invocamos nossa máquina de estado usando o ID físico como ID do recurso.

Transmitimos um evento na linha de comando com a opção `--event`:

```

$ 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"%
```

Invoque uma máquina de estado transmitindo um evento vazio:

```
$ sam remote invoke HelloWorldStateMachine --stack-name state-machine-example
```

```
Invoking Step Function HelloWorldStateMachine
```

```
"Hello World"%
```

Links relacionados

Para obter a documentação relacionada `sam remote invoke` e o uso do AWS SAMCLI, consulte o seguinte:

- [invocação remota do sam](#)
- [AWS SAM CLI solução de problemas](#)

Automatize os testes de integração local com AWS SAM

Embora você possa usar [Introdução aos testes com SAM invocação local](#) para testar o código manualmente, AWS SAM também permite testar seu código usando testes de integração automatizados. O teste de integração ajuda você a detectar problemas no início do ciclo de desenvolvimento, a melhorar a qualidade do código e a economizar tempo, ao mesmo tempo que reduz os custos.

Para criar testes de integração automatizados AWS SAM, primeiro você executa testes nas Funções de Eventos Lambda locais antes de implantar na nuvem. AWS O comando [Introdução aos testes com start-lambda do local do sam](#) inicia um endpoint local que emula o endpoint de invocação do Lambda. Você pode invocá-lo a partir de seus testes automatizados. Como esse endpoint emula o endpoint de invocação do Lambda, você pode escrever testes uma vez e depois executá-los (sem qualquer modificação) na função do Lambda local ou em uma função do Lambda implantada. Você também pode executar os mesmos testes em uma AWS SAM pilha implantada em seu CI/CD pipeline.

Como funciona:

1. Inicie o endpoint local do Lambda.

Inicie o endpoint local do Lambda executando o seguinte comando no diretório que contém seu modelo: AWS SAM

```
sam local start-lambda
```

Esse comando inicia um endpoint local no `http://127.0.0.1:3001` que emula AWS Lambda. Você pode executar seus testes automatizados nesse endpoint local do Lambda. Quando você invoca esse endpoint usando o AWS CLI ou SDK, ele executa localmente a função Lambda especificada na solicitação e retorna uma resposta.

2. Execute um teste de integração no endpoint Lambda local.

Em seu teste de integração, você pode usar o AWS SDK para invocar sua função Lambda com dados de teste, aguardar a resposta e verificar se a resposta é o que você espera. Para executar o teste de integração localmente, você deve configurar o AWS SDK para enviar uma chamada de API Lambda Invoke para invocar o endpoint local do Lambda que você iniciou na etapa anterior.

Veja a seguir um exemplo em Python (os AWS SDKs para outras linguagens têm configurações semelhantes):

```
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"
```

Você pode usar esse código para testar as funções do Lambda implantadas configurando `running_locally` como `False`. Isso configura o AWS SDK ao qual se conectar AWS Lambda na AWS nuvem.

Gere amostras de cargas úteis de eventos com AWS SAM

Para testar suas funções de eventos do Lambda, você pode gerar e personalizar amostras de cargas de eventos que imitam os dados que suas funções do Lambda receberão quando acionadas por outros serviços. AWS Isso inclui serviços como o API Gateway, CloudFormation, Amazon S3 e muito mais.

A geração de exemplos de cargas úteis de eventos ajuda você a testar o comportamento da função do Lambda com uma variedade de entradas diferentes sem precisar trabalhar em um ambiente em tempo real. Essa abordagem também economiza tempo quando comparada à criação manual de amostras AWS de eventos de serviço para testar funções.

Para ver a lista completa de serviços para os quais você pode gerar amostras de cargas de eventos, use este comando:

```
sam local generate-event --help
```

Para ver a lista de opções que você pode usar para um serviço específico, use este comando:

```
sam local generate-event [SERVICE] --help
```

Exemplos:

```
#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
```

Testando e depurando funções duráveis

Testar e depurar funções duráveis localmente funcionam de forma semelhante às funções normais do Lambda, com suporte automático para checkpoint e replay. Este guia aborda cenários comuns de teste e técnicas de solução de problemas.

Fluxo de trabalho de teste local

Ao testar funções duráveis localmente, o fluxo de trabalho difere das funções normais do Lambda:

Fluxo de trabalho durável de testes de funções

1. Crie seu aplicativo:

```
$ sam build
```

2. Invoque sua função durável:

```
$ sam local invoke MyDurableFunction --durable-execution-name test
```

3. Verifique o histórico de execução, se necessário:

```
$ sam local execution history execution-id
```

Cenários de teste comuns

Testando o comportamento do checkpoint

Para testar se sua função corretamente, os pontos de verificação indicam:

```
# 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 você invoca essa função localmente, o período de espera é processado automaticamente.

Testando cenários de retorno de chamada

Para funções que aguardam retornos de chamada externos:

1. Inicie sua função durável que aguarda um retorno de chamada
2. Em outro terminal, resolva o retorno de chamada:

```
$ sam local callback succeed callback-id
```

3. Observe a execução do currículo da função

Solução de problemas

Função durável que não está sendo executada corretamente

Problema: a função não se comporta como uma função durável.

Soluções:

- Verifique se `DurableConfig` está definido no seu modelo de SAM
- Certifique-se de que seu código de função use métodos SDK de função duráveis (por exemplo, `context.wait()`)
- Verifique se você está usando um tempo de execução compatível (TypeScript, JavaScript, Python)

Não é possível recuperar o histórico de execução

Problema: O `local execution history` comando não retorna resultados.

Soluções:

- Verifique se o ID de execução está correto
- Verifique se a função foi invocada pelo menos uma vez

Os comandos de retorno de chamada não funcionam

Problema: os comandos de retorno de chamada não resolvem operações pendentes.

Soluções:

- Verifique se o ID de retorno de chamada está correto
- Verifique se a função está realmente aguardando um retorno de chamada
- Verifique se você está usando a sintaxe correta do comando de retorno de chamada

Dicas de depuração

- Use o histórico de execução - revise o histórico de execução para entender o fluxo de sua função durável
- Teste incrementalmente - comece com operações de espera simples antes de adicionar lógica complexa
- Use registro detalhado - Habilite o registro detalhado para rastrear o fluxo de execução

Saiba mais

Para obter mais informações sobre testes e depuração, consulte:

- [Introdução aos testes com SAM invocação local](#)- Documentação de invocação local
- [sam local execution history](#)- Histórico de execução

Depure seu aplicativo sem servidor com AWS SAM

Depois de testar a aplicação, tudo estará pronto para a depuração dos problemas encontrados. Com a interface de linha de AWS SAM comando (CLI), você pode testar e depurar localmente seu aplicativo sem servidor antes de enviá-lo para a nuvem. AWS A depuração da aplicação identifica e corrige problemas ou erros na aplicação.

Você pode usar AWS SAM para realizar a depuração passo a passo, que é um método de executar código em uma linha ou instrução por vez. Quando você invoca localmente uma função de evento Lambda no modo de depuração dentro AWS SAMCLI do, você pode então anexar um depurador a ela. Com o depurador, você pode percorrer o código linha por linha, ver os valores de diferentes variáveis e corrigir problemas da mesma forma que faria com qualquer outra aplicação. Você pode verificar se seu aplicativo está se comportando conforme o esperado, depurar o que está errado e corrigir quaisquer problemas antes de seguir as etapas de empacotamento e implantação do aplicativo.

Note

Se seu aplicativo inclui uma ou mais camadas, quando você executa e depura localmente seu aplicativo, o pacote de camadas é baixado e armazenado em cache no seu host local. Para obter mais informações, consulte [Como as camadas são armazenadas em cache localmente](#).

Tópicos

- [Depure funções localmente com AWS SAM](#)
- [Passe vários argumentos de tempo de execução ao depurar com AWS SAM](#)
- [Valide seus AWS SAM aplicativos com CloudFormation o Linter](#)

Depure funções localmente com AWS SAM

Você pode usar AWS SAM com uma variedade de AWS kits de ferramentas e depuradores para testar e depurar seus aplicativos sem servidor localmente. Step-through a depuração de suas Lambda Event Functions permite que você identifique e corrija problemas em seu aplicativo, uma linha ou instrução por vez, em seu ambiente local.

Algumas das maneiras de executar a depuração local passo a passo incluem a definição de pontos de interrupção, a inspeção de variáveis e a execução do código da função, uma linha de cada vez. A depuração local passo a passo estreita o ciclo de feedback, possibilitando que você encontre e solucione problemas que possam ocorrer na nuvem.

Você pode usar AWS kits de ferramentas para depurar e também pode executar AWS SAM no modo de depuração. Para obter detalhes, consulte os tópicos desta seção.

Utilizar AWS Kits de ferramentas

AWS Os kits de ferramentas são plug-ins de ambiente de desenvolvimento integrado (IDE) que oferecem a capacidade de realizar muitas tarefas comuns de depuração, como definir pontos de interrupção, inspecionar variáveis e executar o código da função, uma linha por vez. AWS Os kits de ferramentas facilitam o desenvolvimento, a depuração e a implantação de aplicativos sem servidor criados usando AWS SAM. Eles fornecem uma experiência para criar, testar, depurar, implantar e invocar funções do Lambda que está integrada ao seu IDE.

Para obter mais informações sobre AWS kits de ferramentas que você pode usar com AWS SAM, consulte o seguinte:

- [AWS Toolkit for Visual Studio Code](#)
- [AWS Cloud9](#)
- [AWS Toolkit for JetBrains](#)

Há uma variedade de AWS kits de ferramentas que funcionam com diferentes combinações de IDEs e tempos de execução. A tabela a seguir lista IDE/runtime combinações comuns que oferecem suporte à depuração passo a passo de aplicativos: AWS SAM

IDE	Runtime	AWS Kit de ferramentas	Instruções para depuração passo a passo
Visual Studio Code	<ul style="list-style-type: none">• Node.js• Python• .NET• Java• Go	AWS Toolkit for Visual Studio Code	Trabalhar com AWS Serverless Application no Guia do usuário do AWS Toolkit for Visual Studio Code

IDE	Runtime	AWS Kit de ferramentas	Instruções para depuração passo a passo
AWS Cloud9	<ul style="list-style-type: none"> Node.js Python 	AWS Cloud9 ¹ com o AWS kit de ferramentas ativado	Trabalhando com aplicativos AWS sem servidor usando o AWS kit de ferramentas no Guia do AWS Cloud9 usuário.
WebStorm	Node.js	AWS Toolkit for JetBrains ²	Como executar (invocar) ou depurar uma função local no AWS Toolkit for JetBrains
PyCharm	Python	AWS Toolkit for JetBrains ²	Como executar (invocar) ou depurar uma função local no AWS Toolkit for JetBrains
Rider	.NET	AWS Toolkit for JetBrains ²	Como executar (invocar) ou depurar uma função local no AWS Toolkit for JetBrains
IntelliJ	Java	AWS Toolkit for JetBrains ²	Como executar (invocar) ou depurar uma função local no AWS Toolkit for JetBrains

IDE	Runtime	AWS Kit de ferramentas	Instruções para depuração passo a passo
GoLand	Go	AWS Toolkit for JetBrains ²	Como executar (invocar) ou depurar uma função local no AWS Toolkit for JetBrains

Observações:

1. Para ser usado AWS Cloud9 para depurar AWS SAM aplicativos passo a passo, o AWS Toolkit deve estar ativado. Para obter mais informações, consulte [Habilitando o AWS kit de ferramentas](#) no Guia do AWS Cloud9 usuário.
2. Para usar os AWS SAM aplicativos de depuração passo a passo AWS Toolkit for JetBrains a passo, você deve primeiro instalá-los e configurá-los seguindo as instruções encontradas em [Instalando o AWS Toolkit for JetBrains](#) no. AWS Toolkit for JetBrains

Em execução AWS SAM localmente no modo de depuração

[Além da integração com os AWS kits de ferramentas, você também pode executar AWS SAM no modo de depuração para se conectar a depuradores de terceiros, como ptvsd ou delve.](#)

Para executar AWS SAM no modo de depuração, use comandos [SAM invocação local](#) ou [SAM API de inicialização local](#) com a -d opção --debug-port ou.

Por exemplo:

```
# 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 você usar `sam local start-api`, a instância do Gateway da API local expõe todas as funções do Lambda. No entanto, como você pode especificar uma única porta de depuração, é possível depurar somente uma função por vez. Você precisa chamar sua API antes de AWS SAMCLI vincular à porta, o que permite que o depurador se conecte.

Passe vários argumentos de tempo de execução ao depurar com AWS SAM

Você pode optar por passar argumentos de tempo de execução adicionais AWS SAM para inspecionar problemas e solucionar variáveis com mais eficiência. Isso proporciona controle e flexibilidade adicionais ao processo de depuração, o que pode ajudar nas configurações e ambientes de runtime personalizados.

Para passar argumentos adicionais de tempo de execução ao depurar sua função, use a variável de ambiente `DEBUGGER_ARGS`. Isso passa uma sequência de argumentos diretamente para o comando de execução que AWS SAMCLI usa para iniciar sua função.

Por exemplo, se você quiser carregar um depurador como o iKpdb no tempo de execução da sua função Python, você pode passar o seguinte como `DEBUGGER_ARGS`: `-m ikpdb --ikpdb-port=5858 --ikpdb-working-directory=/var/task/ --ikpdb-client-working-directory=/myApp --ikpdb-address=0.0.0.0` Isso carregaria o iKpdb em tempo de execução com os outros argumentos que você especificou.

Nesse caso, seu comando AWS SAMCLI completo seria:

```
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
```

Você pode passar argumentos do depurador para as funções de todos os tempos de execução.

Valide seus AWS SAM aplicativos com CloudFormation o Linter

AWS CloudFormation O Linter (cfn-lint) é uma ferramenta de código aberto que você pode usar para realizar uma validação detalhada em seus modelos. CloudFormation O CFN-lint contém regras

que são guiadas pela especificação do CloudFormation recurso. Use cfn-lint para comparar seus recursos com essas regras e receber mensagens detalhadas sobre erros, avisos ou sugestões informativas. Como alternativa, crie suas próprias regras personalizadas para validá-las. Para saber mais sobre cfn-lint, consulte [cfn-lint](#) no repositório CloudFormation GitHub

Você pode usar o cfn-lint para validar seus modelos AWS Serverless Application Model (AWS SAM) por meio da interface de linha de AWS SAM comando (AWS SAMCLI) executando `sam validate` com a opção. `--lint`

```
sam validate --lint
```

Para personalizar o comportamento do cfn-lint, como criar regras personalizadas ou especificar opções de validação, você pode definir um arquivo de configuração. Para saber mais, consulte [Config File](#) no repositório CloudFormation GitHub cfn-lint. Quando você executa `sam validate --lint`, o comportamento cfn-lint definido em seu arquivo de configuração será aplicado.

Exemplos

Execute a validação cfn-lint em um modelo AWS SAM

```
sam validate --lint --template myTemplate.yaml
```

Saiba mais

Para saber mais sobre o comando `sam validate`, consulte [sam validate](#).

Implante seu aplicativo e seus recursos com AWS SAM

A implantação de seu aplicativo provisiona e configura seus AWS recursos na AWS nuvem, fazendo com que seu aplicativo seja executado na nuvem. AWS SAM usa [CloudFormation](#) como mecanismo de implantação subjacente. AWS SAM usa os artefatos de construção que você cria ao executar o `sam build` comando como entradas padrão para implantar seu aplicativo sem servidor.

Com AWS SAM, você pode implantar seu aplicativo sem servidor manualmente ou automatizar as implantações. Para automatizar as implantações, você usa AWS SAM pipelines com um sistema de integração e implantação contínuas (CI/CD) de sua escolha. Seu pipeline de implantação é uma sequência automatizada de etapas que são executadas para lançar uma nova versão da aplicação sem servidor.

Os tópicos desta seção fornecem orientação sobre implantações automatizadas e manuais. Para implantar a aplicação manualmente, use comandos da CLI do AWS SAM. Para automatizar as implantações, consulte os tópicos desta seção. Eles fornecem especificamente conteúdo detalhado sobre como automatizar implantações usando pipelines e um sistema. CI/CD Isso inclui a geração de um pipeline inicial, a configuração da automação, a solução dos problemas das implantações, o uso da autenticação de usuário do OIDS (OpenID Connect) e o upload de arquivos locais na implantação.

Tópicos

- [Introdução à implantação com AWS SAM](#)
- [Opções para implantar seu aplicativo com AWS SAM](#)
- [Usando CI/CD sistemas e pipelines para implantar com AWS SAM](#)
- [Introdução ao uso mesma sincronização para sincronizar com Nuvem AWS](#)

Introdução à implantação com AWS SAM

Use o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam deploy` para implantar seu aplicativo sem servidor no. Nuvem AWS

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#).
- Para obter uma lista de opções de comando `sam deploy`, consulte [sam deploy](#).
- Para obter um exemplo de uso do `sam deploy` durante um fluxo de trabalho de desenvolvimento típico, consulte [Etapa 3: implantar seu aplicativo no Nuvem AWS](#).

Tópicos

- [Pré-requisitos](#)
- [Implantação de aplicativos usando o sam deploy](#)
- [Práticas recomendadas](#)
- [Opções para sam deploy](#)
- [Solução de problemas](#)
- [Exemplos](#)
- [Saiba mais](#)

Pré-requisitos

Para usar o `sam deploy`, instale o AWS SAM CLI fazendo o seguinte:

- [AWS SAM Pré-requisitos](#).
- [Instale o AWS SAM CLI](#).

Antes de usar `sam deploy` recomendamos uma compreensão básica do seguinte:

- [Configurando o AWS SAM CLI](#).
- [Crie seu aplicativo em AWS SAM](#).
- [Introdução à construção com AWS SAM](#).

Implantação de aplicativos usando o sam deploy

Ao implantar um aplicativo com tecnologia sem servidor pela primeira vez, use a opção `--guided`. O AWS SAMCLI guiará você por um fluxo interativo para definir as configurações de implantação do seu aplicativo.

Para implantar um aplicativo usando o fluxo interativo

1. Vá para o diretório raiz do seu projeto. Esse é o mesmo local do seu AWS SAM modelo.

```
$ cd sam-app
```

2. Execute o seguinte comando :

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

3. Durante o fluxo interativo, os prompts AWS SAMCLI solicita opções para definir as configurações de implantação do seu aplicativo.

Os colchetes ([]) indicam valores padrão. Deixe sua resposta em branco para selecionar o valor padrão. Os valores padrão são obtidos dos seguintes arquivos de configuração:

- `~/.aws/config`— As configurações gerais AWS da sua conta.
- `~/.aws/credentials`— As credenciais AWS da sua conta.
- `<project>/samconfig.toml`— Arquivo de configuração do seu projeto.

Forneça valores respondendo aos prompts AWS SAMCLI. Por exemplo, você pode inserir valores **y** para sim, **n** para não ou em sequência de caracteres.

O AWS SAMCLI escreve suas respostas no arquivo `samconfig.toml` do seu projeto. Para implantações subsequentes, você pode usar `sam deploy` para implantar usando esses valores configurados. Para reconfigurar esses valores, use `sam deploy --guided` novamente ou modifique diretamente seus arquivos de configuração.

Veja a seguir um exemplo de saída:

```
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. Em seguida, AWS SAMCLI implanta seu aplicativo no Nuvem AWS. Durante a implantação, o progresso é exibido em seu prompt de comando. A seguir estão os principais estágios da implantação:
- Para aplicativos com funções de AWS Lambda evento empacotadas como um arquivo de arquivo.zip, eles AWS SAMCLI compactam e carregam o pacote em um bucket do Amazon Simple Storage Service (Amazon S3). Se necessário, o AWS SAMCLI criará um novo bucket.
 - Para aplicativos com o pacote de funções do Lambda como uma imagem de contêiner, a imagem AWS SAMCLI é enviada para o Amazon Elastic Container Registry (Amazon ECR). Se necessário, o AWS SAMCLI criará um novo repositório.
 - O AWS SAMCLI cria um conjunto de AWS CloudFormation alterações e implanta seu aplicativo CloudFormation como uma pilha.
 - O AWS SAMCLI modifica seu AWS SAM modelo implantado com o novo CodeUri valor para suas funções do Lambda.

Veja a seguir um exemplo do AWS SAMCLI resultado da implantação:

```

    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 yment47fc2d5f9d	AWS::ApiGateway::Deplo yment	N/A
+ Add	ServerlessRestApiProdS tage	AWS::ApiGateway::Stage	N/A
+ Add	ServerlessRestApi	AWS::ApiGateway::RestA pi	N/A

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 ResourceStatusReason	ResourceType	LogicalResourceId	
CREATE_IN_PROGRESS	AWS::IAM::Role	HelloWorldFunctionRole	-
CREATE_IN_PROGRESS creation	AWS::IAM::Role	HelloWorldFunctionRole	Resource
Initiated CREATE_COMPLETE	AWS::IAM::Role	HelloWorldFunctionRole	-
CREATE_IN_PROGRESS	AWS::Lambda::Function	HelloWorldFunction	-

CREATE_IN_PROGRESS creation	AWS::Lambda::Function	HelloWorldFunction	Resource
Initiated			
CREATE_COMPLETE	AWS::Lambda::Function	HelloWorldFunction	-
CREATE_IN_PROGRESS	AWS::ApiGateway::RestA pi	ServerlessRestApi	-
CREATE_IN_PROGRESS creation	AWS::ApiGateway::RestA pi	ServerlessRestApi	Resource
Initiated			
CREATE_COMPLETE	AWS::ApiGateway::RestA pi	ServerlessRestApi	-
CREATE_IN_PROGRESS	AWS::Lambda::Permissio n	HelloWorldFunctionHell oWorldPermissionProd	-
CREATE_IN_PROGRESS	AWS::ApiGateway::Deplo yment	ServerlessRestApiDeplo yment47fc2d5f9d	-
CREATE_IN_PROGRESS creation	AWS::Lambda::Permissio n	HelloWorldFunctionHell oWorldPermissionProd	Resource
Initiated			
CREATE_IN_PROGRESS creation	AWS::ApiGateway::Deplo yment	ServerlessRestApiDeplo yment47fc2d5f9d	Resource
Initiated			
CREATE_COMPLETE	AWS::ApiGateway::Deplo yment	ServerlessRestApiDeplo yment47fc2d5f9d	-
CREATE_IN_PROGRESS	AWS::ApiGateway::Stage	ServerlessRestApiProdS tage	-
CREATE_IN_PROGRESS creation	AWS::ApiGateway::Stage	ServerlessRestApiProdS	Resource

```

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-11Z0GSCG28H0M

Key                HelloWorldApi
Description         API Gateway endpoint URL for Prod stage for Hello World
                    function
Value              https://njzfhdm1s0.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
```

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

5. Para visualizar seu aplicativo implantado, faça o seguinte:

1. Abra o CloudFormation console diretamente com o URL <https://console.aws.amazon.com/cloudformation>.
2. Selecione pilhas.
3. Identifique sua pilha pelo nome do aplicativo e selecione-a.

Verifique as alterações antes da implantação

Você pode configurar o AWS SAMCLI para exibir seu conjunto de CloudFormation alterações e solicitar confirmação antes da implantação.

Para confirmar as alterações antes da implantação

1. Durante `sam deploy --guided`, insira **Y** para confirmar as alterações antes da implantação.

```
#Shows you resources changes to be deployed and require a 'Y' to initiate deploy  
Confirm changes before deploy [Y/n]: Y
```

Como alternativa, você pode modificar seu arquivo `samconfig.toml` com o seguinte:

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

2. Durante a implantação, o AWS SAMCLI solicitará que você confirme as alterações antes da implantação. Veja um exemplo a seguir.

```
Waiting for changeset to be created..
```

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

Operation Replacement	LogicalResourceId	ResourceType	
+ 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	
<p>Changeset created successfully. arn:aws:cloudformation:us-west-2:012345678910:changeSet/samcli-deploy1680559234/d9f58a77-98bc-41cd-b9f4-433a5a450d7a</p> <p>Previewing CloudFormation changeset before deployment</p> <p>=====</p> <p>Deploy this changeset? [y/N]: <i>y</i></p>			

Especifique parâmetros adicionais durante a implantação

Você pode especificar valores de parâmetros adicionais a serem configurados durante a implantação. Você faz isso modificando seu modelo AWS SAM e configurando o valor do parâmetro durante a implantação.

Para especificar parâmetros adicionais

1. Modifique a `Parameters` seção do seu AWS SAM modelo. Este é um exemplo:

```

AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Globals:
...
Parameters:
  DomainName:
    Type: String
    Default: example
    Description: Domain name

```

2. Execute `sam deploy --guided`. Veja a seguir um exemplo de saída:

```

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

```

Configure a assinatura de código para suas funções de eventos do Lambda

Você pode configurar sua assinatura de código para funções do Lambda na implantação. Você faz isso modificando seu AWS SAM modelo e configurando a assinatura de código durante a implantação.

Para configurar a assinatura de código

1. Especifique `CodeSigningConfigArn` em seu AWS SAM modelo. Este é um exemplo:

```

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. Execute `sam deploy --guided`. O AWS SAMCLI solicitará que você configure a assinatura de código. Veja a seguir um exemplo de saída:

```
#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):
```

Práticas recomendadas

- Ao usar `sam deploy`, o AWS SAMCLI implanta os artefatos de construção do seu aplicativo localizados no diretório `.aws-sam`. Ao fazer alterações nos arquivos originais do seu aplicativo, execute `sam build` para atualizar o diretório `.aws-sam` antes da implantação.
- Ao implantar um aplicativo pela primeira vez, use `sam deploy --guided` para definir as configurações de implantação. Para implantações subsequentes, você pode usar `sam deploy` para implantar com as configurações definidas.

Opções para `sam deploy`

A seguir estão as opções comumente usadas para `sam deploy`. Para obter uma lista de todas as opções, consulte [sam deploy](#).

Use o fluxo interativo guiado para implantar seu aplicativo

Use a opção `--guided` para configurar as definições de implantação do seu aplicativo por meio de um fluxo interativo. Veja um exemplo a seguir.

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

As configurações de implantação do seu aplicativo são salvas no arquivo `samconfig.toml` do seu projeto. Para saber mais, consulte [Definir configurações do projeto](#).

Solução de problemas

Para solucionar o problema do AWS SAMCLI, consulte [AWS SAM CLI solução de problemas](#).

Exemplos

Implante um aplicativo Hello World que contenha uma função de evento Lambda empacotada como um arquivo de `arquivo.zip`

Por exemplo, consulte [Etapa 3: implantar seu aplicativo no Nuvem AWS](#) no tutorial de aplicativo Hello World.

Implante um aplicativo Hello World que contenha uma função de evento Lambda empacotada como uma imagem de contêiner

Primeiro, usamos `sam init` para criar nosso aplicativo Hello World. Durante o fluxo interativo, selecionamos o tipo de pacote do runtime Python3.9 e pacote Image.

```
$ 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

...

Em seguida, nós cd para o diretório raiz do nosso projeto e executamos `sam build`. O AWS SAMCLI constrói nossa função do Lambda localmente usando o 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
```

Em seguida, executamos `sam deploy --guided` para implantar nosso aplicativo. O AWS SAMCLI nos orienta na definição de nossas configurações de implantação. Em seguida, AWS SAMCLI implanta nosso aplicativo no Nuvem 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 ResourceStatusReason	ResourceType	LogicalResourceId	
CREATE_IN_PROGRESS Initiated	AWS::CloudFormation::Stack	sam-app	User
CREATE_IN_PROGRESS	AWS::IAM::Role	HelloWorldFunctionRole	-
CREATE_IN_PROGRESS creation	AWS::IAM::Role	HelloWorldFunctionRole	Resource Initiated
CREATE_COMPLETE	AWS::IAM::Role	HelloWorldFunctionRole	-
CREATE_IN_PROGRESS	AWS::Lambda::Function	HelloWorldFunction	-
CREATE_IN_PROGRESS creation	AWS::Lambda::Function	HelloWorldFunction	Resource Initiated
CREATE_COMPLETE	AWS::Lambda::Function	HelloWorldFunction	-
CREATE_IN_PROGRESS	AWS::ApiGateway::RestApi	ServerlessRestApi	-
CREATE_IN_PROGRESS creation	AWS::ApiGateway::RestApi	ServerlessRestApi	Resource Initiated

CREATE_COMPLETE	AWS::ApiGateway::RestA pi	ServerlessRestApi	-
CREATE_IN_PROGRESS	AWS::Lambda::Permissio n	HelloWorldFunctionHell oWorldPermissionProd	-
CREATE_IN_PROGRESS	AWS::ApiGateway::Deplo yment	ServerlessRestApiDeplo yment47fc2d5f9d	-
CREATE_IN_PROGRESS creation	AWS::Lambda::Permissio n	HelloWorldFunctionHell oWorldPermissionProd	Resource Initiated
CREATE_IN_PROGRESS creation	AWS::ApiGateway::Deplo yment	ServerlessRestApiDeplo yment47fc2d5f9d	Resource Initiated
CREATE_COMPLETE	AWS::ApiGateway::Deplo yment	ServerlessRestApiDeplo yment47fc2d5f9d	-
CREATE_IN_PROGRESS	AWS::ApiGateway::Stage	ServerlessRestApiProdS tage	-
CREATE_IN_PROGRESS creation	AWS::ApiGateway::Stage	ServerlessRestApiProdS tage	Resource Initiated
CREATE_COMPLETE	AWS::ApiGateway::Stage	ServerlessRestApiProdS tage	-
CREATE_COMPLETE	AWS::Lambda::Permissio n	HelloWorldFunctionHell oWorldPermissionProd	-
CREATE_COMPLETE	AWS::CloudFormation::S tack	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-JFML1J0KHJ71
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

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

Saiba mais

Para saber mais sobre como usar o AWS SAMCLI `sam deploy` comando, veja o seguinte:

- [O AWS SAM workshop completo: Módulo 3 — Implantar manualmente](#) — Aprenda a criar, empacotar e implantar um aplicativo sem servidor usando o. AWS SAMCLI

Opções para implantar seu aplicativo com AWS SAM

Com AWS SAM, você pode implantar seu aplicativo manualmente e automatizar as implantações. Use a CLI do AWS SAM para implantar a aplicação manualmente. Para automatizar a implantação, use pipelines e um sistema de integração e implantação contínuas (CI/CD). Os tópicos desta seção fornecem informações sobre ambas as abordagens.

Tópicos

- [Como usar a CLI do AWS SAM para implantar manualmente](#)
- [Implemente com CI/CD sistemas e pipelines](#)
- [Implantações graduais](#)
- [Solução de problemas de implantações usando o AWS SAM CLI](#)
- [Saiba mais](#)

Como usar a CLI do AWS SAM para implantar manualmente

Depois de desenvolver e testar seu aplicativo sem servidor localmente, você pode implantá-lo usando o comando [sam deploy](#).

Para AWS SAM guiá-lo durante a implantação com solicitações, especifique o `--guided` sinalizador. Quando você especifica esse sinalizador, o comando `sam deploy` compacta os artefatos do seu aplicativo e os carrega para o Amazon Simple Storage Service (Amazon S3) (para arquivos de arquivos.zip) ou para o Amazon Elastic Container Registry (Amazon ECR) (para imagens de contêineres). Em seguida, o comando implanta seu aplicativo na AWS nuvem.

Exemplo:

```
# Deploy an application using prompts:
sam deploy --guided
```

Implemente com CI/CD sistemas e pipelines

AWS SAM ajuda você a automatizar a implantação usando pipelines e um sistema de integração contínua e implantação contínua (CI/CD). AWS SAM pode ser usado para criar pipelines e simplificar CI/CD tarefas para aplicativos sem servidor. Vários CI/CD sistemas oferecem suporte à AWS SAM criação de imagens de contêiner e AWS SAM também fornecem um conjunto de modelos de pipeline

padrão para vários CI/CD sistemas que encapsulam as melhores práticas AWS de implantação da empresa.

Para obter mais informações, consulte [Usando CI/CD sistemas e pipelines para implantar com AWS SAM](#).

Implantações graduais

Se você quiser implantar seu AWS SAM aplicativo gradualmente, em vez de tudo de uma vez, você pode especificar as configurações de implantação que AWS CodeDeploy fornecem. Para obter mais informações, consulte Como [trabalhar com configurações de implantação CodeDeploy no](#) Guia do AWS CodeDeploy usuário.

Para obter informações sobre como configurar seu AWS SAM aplicativo para implantação gradual, consulte [Implantando aplicativos sem servidor gradualmente com AWS SAM](#).

Solução de problemas de implantações usando o AWS SAM CLI

AWS SAMCLI erro: “Restrições de segurança não satisfeitas”

Ao executar `sam deploy --guided`, você recebe a pergunta `HelloWorldFunction may not have authorization defined, Is this okay? [y/N]`. Se responder a esse prompt com **N** (a resposta padrão), você receberá o seguinte erro:

```
Error: Security Constraints Not Satisfied
```

O prompt está informando que o aplicativo que você está prestes a implantar pode ter uma API do Amazon API Gateway configurada sem autorização. Ao responder **N** a esse prompt, você está dizendo que isso não está certo.

Para corrigir isso, você tem as seguintes opções:

- Configurar seu aplicativo com autorização. Para obter informações sobre como configurar a autorização, consulte [Controle o acesso à API com seu AWS SAM modelo](#).
- Responda a essa pergunta com **Y** para indicar que você concorda com a implantação de um aplicativo que tenha uma API do API Gateway configurada sem autorização.

Saiba mais

Para exemplos práticos de implantação de aplicativos sem servidor, consulte o seguinte no The Complete Workshop: AWS SAM

- [Módulo 3 — Implantar manualmente](#) — Aprenda a criar, empacotar e implantar um aplicativo sem servidor usando o. AWS SAMCLI
- [Módulo 4 — CI/CD](#) - Aprenda a automatizar as fases de criação, empacotamento e implantação criando um pipeline de integração e entrega contínuas (CI/CD).

Usando CI/CD sistemas e pipelines para implantar com AWS SAM

AWS SAM ajuda as organizações a criar pipelines para seus CI/CD sistemas preferidos, para que possam obter os benefícios CI/CD com o mínimo esforço, como acelerar a frequência de implantação, reduzir o tempo de espera para mudanças e reduzir os erros de implantação.

AWS SAM simplifica CI/CD tarefas para aplicativos sem servidor com a ajuda da criação de imagens de contêiner. As imagens AWS SAM fornecidas incluem as ferramentas AWS SAMCLI e de construção para vários AWS Lambda tempos de execução compatíveis. Isso facilita a criação e o empacotamento de aplicativos sem servidor usando o. AWS SAMCLI Essas imagens também aliviam a necessidade de as equipes criarem e gerenciarem suas próprias imagens para CI/CD sistemas. Para obter mais informações sobre como AWS SAM criar imagens de contêiner, consulte [Repositórios de imagens para AWS SAM](#).

Vários CI/CD sistemas suportam a AWS SAM criação de imagens de contêineres. O CI/CD sistema que você deve usar depende de vários fatores. Isso inclui se seu aplicativo usa um único runtime ou vários tempos de execução, ou se você deseja criar seu aplicativo em uma imagem de contêiner ou diretamente em uma máquina host, seja uma máquina virtual (VM) ou um host bare metal.

AWS SAM também fornece um conjunto de modelos de pipeline padrão para vários CI/CD sistemas que encapsulam as melhores práticas AWS de implantação da empresa. Esses modelos de pipeline padrão usam formatos de configuração de JSON/YAML pipeline padrão, e as melhores práticas integradas ajudam a realizar implantações em várias contas e em várias regiões, além de verificar se os pipelines não podem fazer alterações não intencionais na infraestrutura.

Você tem duas opções principais para AWS SAM implantar seus aplicativos sem servidor: 1) Modifique sua configuração de pipeline existente para usar AWS SAMCLI comandos ou 2) Gere um

exemplo de configuração de CI/CD pipeline que você possa usar como ponto de partida para seu próprio aplicativo.

Tópicos

- [O que é um pipeline?](#)
- [Como AWS SAM carrega arquivos locais na implantação](#)
- [Gere um CI/CD pipeline inicial com AWS SAM](#)
- [Como personalizar tubulações iniciais com AWS SAM](#)
- [Automatize a implantação do seu aplicativo AWS SAM](#)
- [Como usar a autenticação OIDC com pipelines AWS SAM](#)

O que é um pipeline?

Um pipeline é uma sequência automatizada de etapas que são executadas para lançar uma nova versão de uma aplicação. [Com AWS SAM, você pode usar muitos CI/CD sistemas comuns para implantar seus aplicativos, incluindo Jenkins AWS CodePipeline, GitLab CI/CD e Actions. GitHub](#)

Os modelos de pipeline incluem as melhores práticas de AWS implantação para ajudar nas implantações em várias contas e em várias regiões. AWS ambientes como desenvolvimento e produção normalmente existem em AWS contas diferentes. Isso permite que as equipes de desenvolvimento configurem pipelines de implantação seguros, sem fazer alterações não intencionais na infraestrutura.

Você também pode fornecer seus próprios modelos de pipeline personalizados para ajudar a padronizar os pipelines entre as equipes de desenvolvimento.

Como AWS SAM carrega arquivos locais na implantação

Quando você implanta seu aplicativo no Nuvem AWS, é AWS CloudFormation necessário que seus arquivos locais sejam primeiro carregados em um AWS serviço acessível, como o Amazon Simple Storage Service (Amazon S3). Isso inclui arquivos locais aos quais seu AWS SAM modelo faz referência. Para atender a esse requisito, a CLI do AWS SAM faz o seguinte, quando você usa o comando `sam deploy` ou `sam package`:

1. Carrega automaticamente seus arquivos locais em um AWS serviço acessível.
2. Atualiza automaticamente o modelo da aplicação para fazer referência ao novo caminho do arquivo.

Tópicos

- [Demonstração: use o AWS SAM CLI para fazer upload do código da Lambda Event Function](#)
- [Casos de uso compatíveis](#)
- [Saiba mais](#)

Demonstração: use o AWS SAM CLI para fazer upload do código da Lambda Event Function

Nesta demonstração, inicializamos o aplicativo Hello World de amostra usando um tipo de pacote.zip para nossa função de evento Lambda. Usamos o AWS SAMCLI para carregar automaticamente nosso código de função do Lambda para o Amazon S3 e referenciar seu novo caminho em nosso modelo de aplicativo.

Primeiro, executamos `sam init` para inicializar nosso aplicativo 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  
  
...
```

Nosso código de função do Lambda é organizado no subdiretório `hello_world` do nosso projeto.

```
demo  
### README.md  
### hello_world  
#   ### __init__.py  
#   ### app.py  
#   ### requirements.txt  
### template.yaml  
### tests
```

Em nosso AWS SAM modelo, referenciamos o caminho local para nosso código de função Lambda usando a `CodeUri` propriedade.

```
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  
      ...
```

Em seguida, executamos `sam build` para criar nosso aplicativo e nos preparar para a implantação.

```
$ 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
...

```

Em seguida, executamos `sam deploy --guided` para implantar nosso aplicativo.

```

$ 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 ResourceStatusReason	ResourceType	LogicalResourceId	
CREATE_IN_PROGRESS	AWS::IAM::Role	HelloWorldFunctionRole	-
CREATE_IN_PROGRESS creation	AWS::IAM::Role	HelloWorldFunctionRole	Resource 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 a implantação, o carrega AWS SAMCLI automaticamente nosso código de função do Lambda para o Amazon S3 e atualiza nosso modelo. Nosso modelo modificado no CloudFormation console reflete o caminho do bucket do 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  
      ...
```

Casos de uso compatíveis

Eles AWS SAMCLI podem facilitar automaticamente esse processo para vários tipos de arquivos, tipos de CloudFormation recursos e CloudFormation macros.

Tipos de arquivo

Arquivos e imagens do aplicativo Docker são suportados.

CloudFormation tipos de recursos

Veja a seguir uma lista dos tipos de recursos compatíveis e suas propriedades:

Recurso	Propriedades
AWS::ApiGateway::RestApi	BodyS3Location
AWS::ApiGatewayV2::Api	BodyS3Location

Recurso	Propriedades
<code>AWS::AppSync::FunctionConfiguration</code>	<code>CodeS3Location</code> <code>RequestMappingTemplateS3Location</code> <code>ResponseMappingTemplateS3Location</code>
<code>AWS::AppSync::GraphQLSchema</code>	<code>DefinitionS3Location</code>
<code>AWS::AppSync::Resolver</code>	<code>CodeS3Location</code> <code>RequestMappingTemplateS3Location</code> <code>ResponseMappingTemplateS3Location</code>
<code>AWS::CloudFormation::ModuleVersion</code>	<code>ModulePackage</code>
<code>AWS::CloudFormation::ResourceVersion</code>	<code>SchemaHandlerPackage</code>
<code>AWS::ECR::Repository</code>	<code>RepositoryName</code>
<code>AWS::ElasticBeanstalk::ApplicationVersion</code>	<code>SourceBundle</code>
<code>AWS::Glue::Job</code>	<code>Command.ScriptLocation</code>
<code>AWS::Lambda::Function</code>	<code>Code</code> <code>Code.ImageUri</code>
<code>AWS::Lambda::LayerVersion</code>	<code>Content</code>
<code>AWS::Serverless::Api</code>	<code>DefinitionUri</code>

Recurso	Propriedades
<code>AWS::Serverless::Function</code>	<code>CodeUri</code> <code>ImageUri</code>
<code>AWS::Serverless::GraphQLApi</code>	<code>SchemaUri</code> <code>Function.CodeUri</code> <code>Resolver.CodeUri</code>
<code>AWS::Serverless::HttpApi</code>	<code>DefinitionUri</code>
<code>AWS::Serverless::LayerVersion</code>	<code>ContentUri</code>
<code>AWS::Serverless::StateMachine</code>	<code>DefinitionUri</code>
<code>AWS::StepFunctions::StateMachine</code>	<code>DefinitionS3Location</code>

CloudFormation macros

Os arquivos referenciados usando a macro de transformação `AWS::Include` são suportados.

Saiba mais

Para saber mais sobre a `AWS::Include` transformação, consulte [AWS::Include transform](#) no Guia do AWS CloudFormation usuário.

Para ver um exemplo do uso da `AWS::Include` transformação em um AWS SAM modelo, consulte o padrão API [HTTP API to SQS do API Gateway](#) em Serverless Land.

Gere um CI/CD pipeline inicial com AWS SAM

Quando estiver pronto para automatizar a implantação, você poderá usar um dos nossos modelos AWS SAM de pipeline inicial para gerar um pipeline de implantação para o CI/CD sistema que você escolher usar. O pipeline de implantação é o que você configura e usa para automatizar a implantação da aplicação sem servidor. Um modelo de pipeline inicial é pré-configurado para ajudar você a configurar rapidamente o pipeline de implantação para a aplicação sem servidor.

Com um modelo de pipeline inicial, você pode gerar pipelines em poucos minutos usando o comando [sam pipeline init](#).

Os modelos iniciais do pipeline usam a JSON/YAML sintaxe familiar do CI/CD sistema e incorporam as melhores práticas, como gerenciar artefatos em várias contas e regiões e usar a quantidade mínima de permissões necessárias para implantar o aplicativo. [Atualmente, a AWS SAM CLI suporta a geração de configurações iniciais de CI/CD pipeline para Jenkins, GitLab CI/CD AWS CodePipeline, Actions e Bitbucket Pipelines. GitHub](#)

Aqui estão as tarefas de alto nível que você precisa realizar para gerar uma configuração inicial do pipeline:

1. Crie recursos de infraestrutura — Seu pipeline requer determinados AWS recursos, por exemplo, o usuário e as funções do IAM com as permissões necessárias, um bucket do Amazon S3 e, opcionalmente, um repositório do Amazon ECR.
2. Conecte seu repositório Git ao seu CI/CD sistema — Seu CI/CD sistema precisa saber qual repositório Git acionará a execução do pipeline. Observe que essa etapa pode não ser necessária, dependendo da combinação do repositório Git e do CI/CD sistema que você está usando.
3. Gere sua configuração de pipeline — Essa etapa gera uma configuração inicial de pipeline que inclui dois estágios de implantação.
4. Confirme a configuração do pipeline no repositório Git — Essa etapa é necessária para garantir que seu CI/CD sistema esteja ciente da configuração do pipeline e seja executada quando as alterações forem confirmadas.

Depois de gerar a configuração inicial do pipeline e confirmá-la no seu repositório Git, sempre que alguém fizer uma alteração de código nesse repositório, seu pipeline será acionado para ser executado automaticamente.

A ordem dessas etapas e os detalhes de cada etapa variam de acordo com seu sistema de CI/CD:

- Se você estiver usando AWS CodePipeline, consulte [Gerando um pipeline inicial para AWS CodePipeline AWS SAM](#).
- Se você estiver usando Jenkins, GitLab CI/CD, GitHub Actions ou Bitbucket Pipelines, consulte. [Use AWS SAM para gerar pipelines iniciais para Jenkins, GitLab CI/CD, Actions, Bitbucket Pipelines GitHub](#)

Gerando um pipeline inicial para AWS CodePipeline AWS SAM

Para gerar uma configuração de pipeline inicial para AWS CodePipeline, execute as seguintes tarefas nesta ordem:

1. Criar recursos de infraestrutura
2. Gere a configuração do pipeline
3. Confirme a configuração do pipeline no repositório Git
4. Conecte seu repositório Git ao seu sistema CI/CD

Note

O procedimento a seguir utiliza dois comandos AWS SAMCLI, [bootstrap do mesmo pipeline](#) e [sam pipeline init](#). A razão pela qual existem dois comandos é lidar com o caso de uso em que os administradores (ou seja, usuários que precisam de permissão para configurar AWS recursos de infraestrutura, como usuários e funções do IAM) têm mais permissão do que os desenvolvedores (ou seja, usuários que precisam apenas de permissão para configurar pipelines individuais, mas não os AWS recursos de infraestrutura necessários).

Etapa 1: Criar recursos de infraestrutura

Os pipelines que usam AWS SAM exigem determinados AWS recursos, como um usuário e funções do IAM com as permissões necessárias, um bucket do Amazon S3 e, opcionalmente, um repositório Amazon ECR. Você deve ter um conjunto de recursos de infraestrutura para cada estágio de implantação do pipeline.

Você pode executar o seguinte comando para ajudar nesta configuração:

```
sam pipeline bootstrap
```

Note

Execute o comando anterior para cada estágio de implantação do seu pipeline.

Etapa 2: Gerar a configuração do pipeline

Para gerar a configuração do pipeline, execute o comando a seguir.

```
sam pipeline init
```

Etapa 3: Confirme a configuração do pipeline no repositório Git

Essa etapa é necessária para garantir que seu CI/CD sistema esteja ciente da configuração do pipeline e seja executada quando as alterações forem confirmadas.

Etapa 4: Conecte seu repositório Git ao seu sistema CI/CD

Pois agora AWS CodePipeline você pode criar a conexão executando o seguinte comando:

```
sam deploy -t codepipeline.yaml --stack-name <pipeline-stack-name> --  
capabilities=CAPABILITY_IAM --region <region-X>
```

Se você estiver usando o GitHub Bitbucket, depois de executar o `sam deploy` comando anteriormente, conclua a conexão seguindo as etapas em Para concluir uma conexão, encontradas no tópico Atualizar uma conexão [pendente](#) no guia do usuário do console Developer Tools. Além disso, armazene uma cópia `CodeStarConnectionArn` da saída do `sam deploy` comando, pois você precisará dela se quiser usar AWS CodePipeline com outra ramificação diferente demain.

Configurando outras ramificações

Por padrão, AWS CodePipeline usa a `main` ramificação com AWS SAM. Se quiser usar uma ramificação diferente de `main`, você deve executar o comando `sam deploy` novamente. Observe que, dependendo do repositório Git que você estiver usando o, poderá ser necessário fornecer o `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>"
```

Saiba mais

Para ver um exemplo prático de configuração de um CI/CD pipeline, consulte [CI/CD AWS CodePipeline em The Complete Workshop](#). AWS SAM

Use AWS SAM para gerar pipelines iniciais para Jenkins, GitLab CI/CD, Actions, Bitbucket Pipelines GitHub

Para gerar uma configuração inicial de pipeline para Jenkins, GitLab CI/CD, GitHub Actions ou Bitbucket Pipelines, execute as seguintes tarefas nesta ordem:

1. Criar recursos de infraestrutura
2. Conecte seu repositório Git ao seu sistema CI/CD
3. Crie objetos de credencial
4. Gere a configuração do pipeline
5. Confirme a configuração do pipeline no repositório Git

Note

O procedimento a seguir utiliza dois comandos AWS SAMCLI, [bootstrap do mesmo pipeline](#) e [sam pipeline init](#). A razão pela qual existem dois comandos é lidar com o caso de uso em que os administradores (ou seja, usuários que precisam de permissão para configurar AWS recursos de infraestrutura, como usuários e funções do IAM) têm mais permissão do que os desenvolvedores (ou seja, usuários que precisam apenas de permissão para configurar pipelines individuais, mas não os AWS recursos de infraestrutura necessários).

Etapa 1: Criar recursos de infraestrutura

Os pipelines que usam AWS SAM exigem determinados AWS recursos, como um usuário e funções do IAM com as permissões necessárias, um bucket do Amazon S3 e, opcionalmente, um repositório Amazon ECR. Você deve ter um conjunto de recursos de infraestrutura para cada estágio de implantação do pipeline.

Você pode executar o seguinte comando para ajudar nesta configuração:

```
sam pipeline bootstrap
```


 Note

Execute o comando anterior para cada estágio de implantação do seu pipeline.

Você deve capturar AWS as credenciais (ID da chave e chave secreta) dos usuários do pipeline em cada estágio de implantação do seu pipeline, pois elas são necessárias para as etapas subsequentes.

Etapa 2: Conecte seu repositório Git ao seu sistema CI/CD

É necessário conectar seu repositório Git ao seu CI/CD sistema para que o CI/CD sistema possa acessar o código-fonte do seu aplicativo para compilações e implantações.

 Note

Você pode ignorar esta etapa se estiver usando uma das combinações a seguir, porque a conexão é feita para você automaticamente:

1. GitHub Ações com GitHub repositório
2. GitLab CI/CD com repositório GitLab
3. Pipelines do Bitbucket com um repositório do Bitbucket

Para conectar seu repositório Git ao seu CI/CD sistema, faça o seguinte:

- Se você estiver usando o Jenkins, consulte a [documentação do Jenkins](#) para “Adicionar uma fonte de ramificação”.
- Se você estiver usando GitLab CI/CD e um repositório Git diferente GitLab, consulte a [GitLabdocumentação](#) para “conectar um repositório externo”.

Etapa 3: Criar objetos de credencial

Cada CI/CD sistema tem sua própria maneira de gerenciar as credenciais necessárias para que o CI/CD sistema acesse seu repositório Git.

Para criar os objetos de credencial necessários, faça o seguinte:

- Se você estiver usando o Jenkins, crie uma única “credencial” que armazene o ID da chave e a chave secreta. Siga as instruções no blog [Como criar um pipeline do Jenkins com AWS SAM](#), na seção Configurar o Jenkins. Você precisará dessa “ID de credencial” para a próxima etapa.
- Se você estiver usando GitLab CI/CD, crie duas “variáveis protegidas”, uma para cada ID de chave e chave secreta. Siga as instruções na [GitLab documentação](#) — você precisará de duas “chaves variáveis” para a próxima etapa.
- Se você estiver usando GitHub Ações, crie dois “segredos criptografados”, um para cada chave e uma chave secreta. Siga as instruções na [GitHub documentação](#) - você precisará de dois “nomes secretos” para a próxima etapa.
- Se você estiver usando o Bitbucket Pipelines, crie duas “variáveis seguras”, uma para cada ID de chave e chave secreta. Siga as instruções em [Variáveis e segredos](#) - você precisará de dois “nomes secretos” para a próxima etapa.

Etapa 4: Gerar a configuração do pipeline

Para gerar a configuração do pipeline, execute o comando a seguir. Você precisará inserir o objeto de credencial criado na etapa anterior:

```
sam pipeline init
```

Etapa 5: Confirme a configuração do pipeline no repositório Git

Essa etapa é necessária para garantir que seu CI/CD sistema esteja ciente da configuração do pipeline e seja executada quando as alterações forem confirmadas.

Saiba mais

Para ver um exemplo prático de como configurar um CI/CD pipeline usando GitHub Actions, consulte [CI/CD GitHub em The Complete Workshop](#). AWS SAM

Como personalizar tubulações iniciais com AWS SAM

Como CI/CD administrador, talvez você queira personalizar um modelo de pipeline inicial e as instruções guiadas associadas que os desenvolvedores da sua organização possam usar para criar configurações de pipeline.

O AWS SAM CLI usa modelos Cookiecutter ao criar modelos iniciais. Para obter detalhes sobre o cookie cutter template, [Cookiecutter](#).

Você também pode personalizar os prompts que AWS SAMCLI exibe aos usuários ao criar configurações de pipeline usando o comando `sam pipeline init`. Para personalizar os prompts do usuário, faça o seguinte:

1. Criar um arquivo **questions.json** — O arquivo `questions.json` deve estar na raiz do repositório do projeto. Este é o mesmo diretório do arquivo `cookiecutter.json`. Para ver o esquema do arquivo `questions.json`, consulte [questions.json.schema](#). Para ver um arquivo `questions.json` de exemplo, consulte [questions.json](#).
2. Mapeie chaves de pergunta com nomes de cookiecutter — Cada objeto no arquivo `questions.json` precisa de uma chave que corresponda a um nome no modelo cookiecutter. Essa combinação de teclas é a forma como AWS SAMCLI mapeia as respostas rápidas do usuário ao modelo do cortador de biscoitos. Para obter exemplo dessa correspondência de teclas, consulte a seção [Exemplo de arquivos](#) mais adiante neste tópico.
3. Crie um **metadata.json** arquivo - Declare o número de estágios que o pipeline terá no arquivo `metadata.json`. O número de estágios instrui o comando `sam pipeline init` sobre quantos estágios solicitar informações ou, no caso da opção `--bootstrap`, para quantos estágios criar recursos de infraestrutura. Para ver um arquivo `metadata.json` de exemplo que declara um pipeline com dois estágios, consulte [metadata.json](#).

Projetos de exemplo

Aqui estão exemplos de projetos, cada um incluindo um modelo Cookiecutter, um arquivo `questions.json` e um arquivo: `metadata.json`

- Exemplo do Jenkins: [modelo de pipeline do Jenkins em dois estágios](#)
- CodePipeline exemplo: [modelo de CodePipeline pipeline de dois estágios](#)

Exemplo de arquivos

O conjunto de arquivos a seguir mostra como as perguntas no arquivo `questions.json` são associadas às entradas no arquivo de modelo Cookiecutter. Observe que esses exemplos são trechos de arquivo, não arquivos completos. Para ver exemplos de arquivos completos, consulte a seção [Projetos de exemplo](#) anterior neste tópico.

Exemplo de **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"
}, {
  ...

```

Exemplo de **cookiecutter.json**:

```

{
  "outputDir": "aws-sam-pipeline",
  "pipeline_user_jenkins_credential_id": "",
  "sam_template": "",
  ...

```

Exemplo de **Jenkinsfile**:

```

pipeline {
  agent any
  environment {
    PIPELINE_USER_CREDENTIAL_ID =
'{{cookiecutter.pipeline_user_jenkins_credential_id}}'
    SAM_TEMPLATE = '{{cookiecutter.sam_template}}'
    ...

```

Automatize a implantação do seu aplicativo AWS SAM

Em AWS SAM, a forma como você automatiza a implantação do seu AWS SAM aplicativo varia de acordo com o CI/CD sistema que você está usando. Por esse motivo, os exemplos desta seção mostram como configurar vários CI/CD sistemas para automatizar a criação de aplicativos sem servidor em uma imagem de contêiner de AWS SAM compilação. Essas imagens de contêiner

de construção facilitam a criação e o empacotamento de aplicativos sem servidor usando o. AWS SAMCLI

Os procedimentos do CI/CD pipeline existente para implantar aplicativos sem servidor AWS SAM são um pouco diferentes, dependendo do CI/CD sistema que você está usando.

Os tópicos a seguir fornecem exemplos de como configurar seu CI/CD sistema para criar aplicativos sem servidor em uma imagem de contêiner de AWS SAM compilação:

Tópicos

- [Usando AWS CodePipeline para implantar com AWS SAM](#)
- [Usando o Bitbucket Pipelines para implantar com AWS SAM](#)
- [Usando o Jenkins para implantar com AWS SAM](#)
- [Usando GitLab CI/CD para implantar com AWS SAM](#)
- [Usando GitHub ações para implantar com AWS SAM](#)

Usando AWS CodePipeline para implantar com AWS SAM

Para configurar seu [AWS CodePipeline](#) pipeline para automatizar a criação e a implantação do seu AWS SAM aplicativo, seu CloudFormation modelo e `buildspec.yml` arquivo devem conter linhas que façam o seguinte:

1. Faça referência a uma imagem de contêiner de construção com o tempo de execução necessário a partir das imagens disponíveis. O exemplo a seguir usa a imagem do contêiner de compilação do `public.ecr.aws/sam/build-nodejs20.x`.
2. Configure os estágios do pipeline para executar os AWS SAM comandos necessários da interface de linha de comando (CLI). O exemplo a seguir executa dois comandos AWS SAM CLI: `sam build` e `sam deploy` (com as opções necessárias).

Este exemplo pressupõe que você tenha declarado todas as funções e camadas em seu arquivo AWS SAM de modelo comruntime: `nodejs20.x`.

CloudFormation trecho do modelo:

```
CodeBuildProject:
  Type: AWS::CodeBuild::Project
  Properties:
    Environment:
```

```
ComputeType: BUILD_GENERAL1_SMALL
Image: public.ecr.aws/sam/build-nodejs20.x
Type: LINUX_CONTAINER
...
```

buildspec.yml snippet:

```
version: 0.2
phases:
  build:
    commands:
      - sam build
      - sam deploy --no-confirm-changeset --no-fail-on-empty-changeset
```

Para obter a lista das imagens de contêiner disponíveis para a criação do Amazon Elastic Container Registry (Amazon ECR), consulte [Repositórios de imagens para AWS SAM](#).

Usando o Bitbucket Pipelines para implantar com AWS SAM

Para configurar seu [Bitbucket Pipeline](#) para automatizar a criação e a implantação do seu AWS SAM aplicativo, seu `bitbucket-pipelines.yml` arquivo deve conter linhas que façam o seguinte:

1. Faça referência a uma imagem de contêiner de construção com o tempo de execução necessário a partir das imagens disponíveis. O exemplo a seguir usa a imagem do contêiner de compilação do `public.ecr.aws/sam/build-nodejs20.x`.
2. Configure os estágios do pipeline para executar os AWS SAM comandos necessários da interface de linha de comando (CLI). O exemplo a seguir executa dois comandos AWS SAM CLI: `sam build` e `sam deploy` (com as opções necessárias).

Este exemplo pressupõe que você tenha declarado todas as funções e camadas em seu arquivo AWS SAM de modelo comruntime: `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
```

Para obter a lista das imagens de contêiner disponíveis para a criação do Amazon Elastic Container Registry (Amazon ECR), consulte [Repositórios de imagens para AWS SAM](#).

Usando o Jenkins para implantar com AWS SAM

Para configurar seu pipeline do [Jenkins](#) para automatizar a criação e a implantação do seu AWS SAM aplicativo, você Jenkinsfile deve conter linhas que façam o seguinte:

1. Faça referência a uma imagem de contêiner de construção com o tempo de execução necessário a partir das imagens disponíveis. O exemplo a seguir usa a imagem do contêiner de compilação do `public.ecr.aws/sam/build-nodejs20.x`.
2. Configure os estágios do pipeline para executar os AWS SAM comandos necessários da interface de linha de comando (CLI). O exemplo a seguir executa dois comandos AWS SAM CLI: `sam build` e `sam deploy` (com as opções necessárias).

Este exemplo pressupõe que você tenha declarado todas as funções e camadas em seu arquivo AWS SAM de modelo comruntime: `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'
      }
    }
  }
}
```

Para obter a lista das imagens de contêiner disponíveis para a criação do Amazon Elastic Container Registry (Amazon ECR), consulte [Repositórios de imagens para AWS SAM](#).

Usando GitLab CI/CD para implantar com AWS SAM

Para configurar seu [GitLab](#) pipeline para automatizar a criação e a implantação do seu AWS SAM aplicativo, seu `gitlab-ci.yml` arquivo deve conter linhas que façam o seguinte:

1. Faça referência a uma imagem de contêiner de construção com o tempo de execução necessário a partir das imagens disponíveis. O exemplo a seguir usa a imagem do contêiner de compilação do `public.ecr.aws/sam/build-nodejs20.x`.
2. Configure os estágios do pipeline para executar os AWS SAM comandos necessários da interface de linha de comando (CLI). O exemplo a seguir executa dois comandos AWS SAM CLI: `sam build` e `sam deploy` (com as opções necessárias).

Este exemplo pressupõe que você tenha declarado todas as funções e camadas em seu arquivo AWS SAM de modelo comruntime: `nodejs20.x`.

```
image: public.ecr.aws/sam/build-nodejs20.x
deploy:
  script:
    - sam build
    - sam deploy --no-confirm-changeset --no-fail-on-empty-changeset
```

Para obter a lista das imagens de contêiner disponíveis para a criação do Amazon Elastic Container Registry (Amazon ECR), consulte [Repositórios de imagens para AWS SAM](#).

Usando GitHub ações para implantar com AWS SAM

Para configurar seu [GitHub](#) pipeline para automatizar a criação e a implantação do seu AWS SAM aplicativo, você deve primeiro instalar a interface de linha de AWS SAM comando (CLI) em seu host. Você pode usar [GitHub Ações](#) em seu GitHub fluxo de trabalho para ajudar nessa configuração.

O exemplo GitHub de fluxo de trabalho a seguir configura um host Ubuntu usando uma série de GitHub ações e, em seguida, executa AWS SAMCLI comandos para criar e implantar um AWS SAM aplicativo:

```
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
```

Para obter a lista das imagens de contêiner disponíveis para a criação do Amazon Elastic Container Registry (Amazon ECR), consulte [Repositórios de imagens para AWS SAM](#).

Como usar a autenticação OIDC com pipelines AWS SAM

AWS Serverless Application Model (AWS SAM) suporta autenticação de usuário OpenID Connect (OIDC) para Bitbucket, GitHub Actions e integração GitLab contínua e entrega contínua (contas de CI/CD) platforms. With this support, you can use authorized CI/CD usuário de qualquer uma dessas plataformas) para gerenciar seus pipelines de aplicativos sem servidor. Caso contrário, você precisaria criar e gerenciar vários usuários AWS Identity and Access Management (IAM) para controlar o acesso aos AWS SAM pipelines.

Configurar o OIDC com pipeline AWS SAM

Durante o processo `aws-sam pipeline bootstrap` de configuração, faça o seguinte para configurar o OIDC com seu AWS SAM pipeline.

1. Quando solicitado a escolher um provedor de identidade, selecione OIDC.
2. Em seguida, selecione um provedor OIDC compatível.
3. Insira a URL do provedor OIDC, começando com **https://**.

Note

AWS SAM faz referência a esse URL quando ele gera o tipo de `AWS::IAM::OIDCProvider` recurso.

4. Em seguida, siga as instruções e insira as informações da CI/CD plataforma necessárias para acessar a plataforma selecionada. Esses detalhes variam de acordo com a plataforma e podem incluir:
 - ID do cliente do OIDC.
 - Nome do repositório do código ou identificador universalmente exclusivo (UUID).

- O nome do grupo ou da organização associada ao repositório.
 - GitHub organização à qual o repositório de código pertence.
 - GitHub nome do repositório.
 - Filial a partir da qual as implantações ocorrerão.
5. AWS SAM exibe um resumo da configuração do OIDC inserida. Insira o número de uma configuração para editá-la ou pressione Enter para continuar.
 6. Quando solicitado a confirmar a criação dos recursos necessários para suportar a conexão OIDC inserida, pressione Y para continuar.

AWS SAM gera um `AWS::IAM::OIDCProvider` AWS CloudFormation recurso com a configuração fornecida que assume a função de execução do pipeline. Para saber mais sobre esse tipo de CloudFormation recurso, consulte [AWS::IAM::OIDCProvider](#) no Guia do AWS CloudFormation usuário.

Note

Se o recurso do provedor de identidade (IdP) já existir no seu Conta da AWS, AWS SAM faça referência a ele em vez de criar um novo recurso.

Exemplo

A seguir está um exemplo de configuração do OIDC com AWS SAM pipeline.

```
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]:
```

Saiba mais

Para obter mais informações sobre como usar o OIDC com o AWS SAM pipeline, consulte. [bootstrap do mesmo pipeline](#)

Introdução ao uso mesma sincronização para sincronizar com Nuvem AWS

O AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam sync` fornece opções para sincronizar rapidamente as alterações do aplicativo local com Nuvem AWS o. Use `sam sync` ao desenvolver seus aplicativos para:

1. Detecte e sincronize automaticamente as alterações locais com Nuvem AWS o.
2. Personalize quais alterações locais são sincronizadas com o Nuvem AWS.
3. Prepare seu aplicativo na nuvem para testes e validação.

Com `sam sync`, você pode criar um fluxo de trabalho de desenvolvimento rápido que reduz o tempo necessário para sincronizar suas alterações locais com a nuvem para testes e validação.

Note

O comando `sam sync` é recomendado para ambientes de desenvolvimento. Para ambientes de produção, recomendamos usar `sam deploy` ou configurar um pipeline contínuo de integração e entrega (CI/CD). Para saber mais, consulte [Implante seu aplicativo e seus recursos com AWS SAM](#).

O `sam sync` comando faz parte do AWS SAM Accelerate. AWS SAM Accelerate fornece ferramentas que você pode usar para acelerar a experiência de desenvolvimento e teste de aplicativos sem servidor no. Nuvem AWS

Tópicos

- [Detecte e sincronize automaticamente as alterações locais no Nuvem AWS](#)
- [Personalize quais alterações locais são sincronizadas com o Nuvem AWS](#)
- [Prepare seu aplicativo na nuvem para testes e validação](#)
- [Opções para o comando `sam sync`](#)
- [Solução de problemas](#)
- [Exemplos](#)
- [Saiba mais](#)

Detecte e sincronize automaticamente as alterações locais no Nuvem AWS

Execute `sam sync` com a opção `--watch` de começar a sincronizar seu aplicativo com Nuvem AWS. Isso faz o seguinte:

1. Crie seu aplicativo — Esse processo é semelhante ao uso do comando `sam build`.
2. Implante seu aplicativo — O AWS SAMCLI implanta seu aplicativo AWS CloudFormation usando suas configurações padrão. Os seguintes valores padrão são usados:
 - a. AWS credenciais e configurações gerais encontradas na sua pasta de `.aws` usuário.
 - b. Configurações de implantação do aplicativo encontradas no arquivo do seu aplicativo `samconfig.toml`.

Se os valores padrão não puderem ser encontrados, AWS SAMCLI informará você e sairão do processo de sincronização.

3. Observe as alterações locais — O AWS SAM CLI permanece em execução e observa as alterações locais em seu aplicativo. Isso é o que a opção `--watch` oferece.

Esta opção pode ser ativada por padrão. Para valores padrão, consulte o arquivo `samconfig.toml` do seu aplicativo. O seguinte é um arquivo de exemplo:

```
...
[default.sync]
[default.sync.parameters]
watch = true
...
```

4. Sincronize alterações locais com o Nuvem AWS — Quando você faz alterações locais, AWS SAMCLI ele detecta e sincroniza essas alterações com o Nuvem AWS método mais rápido disponível. Dependendo do tipo de alteração, o seguinte pode ocorrer:
 - a. Se seu recurso atualizado for compatível com APIs de AWS serviço, eles o AWS SAMCLI usarão para implantar suas alterações. Isso resulta em uma sincronização rápida para atualizar seu recurso no Nuvem AWS.
 - b. Se seu recurso atualizado não oferecer suporte a APIs de AWS serviço, eles AWS SAMCLI realizarão uma CloudFormation implantação. Isso atualiza todo o seu aplicativo no Nuvem AWS. Embora não seja tão rápido, ele evita que você precise iniciar manualmente uma implantação.

Como o `sam sync` comando atualiza automaticamente seu aplicativo no Nuvem AWS, ele é recomendado somente para ambientes de desenvolvimento. Ao executar `sam sync`, você deverá confirmar:

```
**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
```

Personalize quais alterações locais são sincronizadas com o Nuvem AWS

Forneça opções para personalizar quais alterações locais são sincronizadas com o Nuvem AWS. Isso pode acelerar o tempo necessário para ver suas alterações locais na nuvem para testes e validação.

Por exemplo, forneça a `--code` opção de sincronizar somente alterações de código, como o código da função de AWS Lambda evento. Durante o desenvolvimento, se você se concentrar especificamente no código Lambda, isso colocará suas alterações na nuvem rapidamente para teste e validação. Este é um exemplo:

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

Para sincronizar somente alterações de código para uma função ou camada específica do Lambda, use a opção `--resource-id`. Este é um exemplo:

```
$ sam sync --code --resource-id HelloWorldFunction --resource-id HelloWorldLayer
```

Prepare seu aplicativo na nuvem para testes e validação

O comando `sam sync` encontra automaticamente o método mais rápido disponível para atualizar seu aplicativo no Nuvem AWS. Isso pode acelerar seus fluxos de trabalho de desenvolvimento e testes na nuvem. Ao utilizar as APIs de AWS serviço, você pode desenvolver, sincronizar e testar rapidamente os recursos compatíveis. Para ver um exemplo prático, consulte o [Módulo 6 - AWS SAM Acelere](#) no workshop completo AWS SAM .

Opções para o comando `sam sync`

A seguir estão algumas das principais opções que você pode usar para modificar o comando `sam sync`. Para obter uma lista de todas as opções, consulte [mesma sincronização](#).

Execute uma única vez CloudFormation implantação

Use a opção `--no-watch` para desativar a sincronização automática. Este é um exemplo:

```
$ sam sync --no-watch
```

Eles AWS SAMCLI realizarão uma CloudFormation implantação única. Esse comando agrupa as ações executadas pelos comandos `sam build` e `sam deploy`.

Ignore a inicial CloudFormation implantação

Você pode personalizar se uma CloudFormation implantação é necessária sempre que `sam sync` for executada.

- `--no-skip-deploy-sync` Forneça a exigência de uma CloudFormation implantação sempre que `sam sync` for executada. Isso garante que sua infraestrutura local seja sincronizada CloudFormation, evitando desvios. Usar essa opção adiciona mais tempo ao seu fluxo de trabalho de desenvolvimento e teste.
- Forneça `--skip-deploy-sync` para tornar CloudFormation a implantação opcional. Ele AWS SAMCLI comparará seu AWS SAM modelo local com o CloudFormation modelo implantado e ignorará a CloudFormation implantação inicial se uma alteração não for detectada. Ignorar a CloudFormation implantação pode economizar seu tempo ao sincronizar alterações locais com o. Nuvem AWS

Se nenhuma alteração for detectada, o ainda AWS SAMCLI realizará uma CloudFormation implantação nos seguintes cenários:

- Se já passaram 7 dias ou mais desde sua última CloudFormation implantação.
- Se um grande número de alterações no código da função Lambda for detectado, a CloudFormation implantação será o método mais rápido para atualizar seu aplicativo.

Este é um exemplo:

```
$ sam sync --skip-deploy-sync
```

Sincronizar um recurso a partir de uma pilha aninhada

Para sincronizar um recurso a partir de uma pilha aninhada

1. Forneça a pilha raiz usando `--stack-name`.
2. Identifique o recurso na pilha aninhada usando o seguinte formato: *nestedStackId/resourceId*.
3. Forneça o recurso na pilha aninhada usando `--resource-id`.

Este é um exemplo:

```
$ sam sync --code --stack-name sam-app --resource-id myNestedStack/HelloWorldFunction
```

Para mais informações sobre a criação de aplicativos aninhados, consulte [Reutilize código e recursos usando aplicativos aninhados no AWS SAM](#).

Especifique um específico CloudFormation pilha para atualizar

Para especificar uma CloudFormation pilha específica a ser atualizada, forneça a `--stack-name` opção. Este é um exemplo:

```
$ sam sync --stack-name dev-sam-app
```

Acelere os tempos de compilação criando seu projeto na pasta de origem

Para runtimes e métodos de compilação compatíveis, você pode usar a opção `--build-in-source` de criar seu projeto diretamente na pasta de origem. Por padrão, eles AWS SAM CLI são compilados em um diretório temporário, o que envolve a cópia do código-fonte e dos arquivos do projeto. Com `--build-in-source`, AWS SAM CLI as compilações diretamente na sua pasta de origem, o que acelera o processo de compilação ao eliminar a necessidade de copiar arquivos para um diretório temporário.

Para ver uma lista de runtimes compatíveis, consulte [--build-in-source](#).

Especifique arquivos e pastas que não iniciarão uma sincronização

Use a opção `--watch-exclude` para especificar qualquer arquivo ou pasta que não iniciará uma sincronização quando atualizado. Para obter mais informações sobre essa opção, consulte [--watch-exclude](#).

Veja a seguir um exemplo que exclui o arquivo associado `package-lock.json` à nossa função `HelloWorldFunction`:

```
$ sam sync --watch --watch-exclude HelloWorldFunction=package-lock.json
```

Quando esse comando for executado, o AWS SAM CLI iniciará o processo de sincronização. Essa transmissão inclui o seguinte:

- Execute `sam build` para criar suas funções e preparar seu aplicativo para a implantação.
- Execute `sam deploy` para implantar seu aplicativo.
- Fique atento às alterações em seu aplicativo.

Quando modificamos o `package-lock.json` arquivo, eles AWS SAM CLI não iniciam uma sincronização. Quando outro arquivo for atualizado, o AWS SAM CLI iniciará uma sincronização, que incluirá o `package-lock.json` arquivo.

Veja a seguir um exemplo de especificação de uma função do Lambda de uma pilha filho:

```
$ sam sync --watch --watch-exclude ChildStackA/MyFunction=database.sqlite3
```

Solução de problemas

Para solucionar o problema do AWS SAMCLI, consulte [AWS SAM CLI solução de problemas](#).

Exemplos

Usando o `sam sync` para atualizar o aplicativo Hello World

Neste exemplo, começamos inicializando a amostra do aplicativo Hello World. Para saber mais sobre este aplicativo, consulte [Tutorial: implante um aplicativo Hello World com AWS SAM](#).

Executar `sam sync` inicia o processo de criação e implantação.

```
$ 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-
BUFVM02PJIYF	
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.
```

Quando a implantação estiver concluída, modificamos o código HelloWorldFunction. O AWS SAMCLI detecta essa alteração e sincroniza nosso aplicativo com o. Nuvem AWS Como é AWS Lambda compatível com APIs de AWS serviço, uma sincronização rápida é executada.

```
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.
```

Em seguida, modificamos nosso endpoint de API no AWS SAM modelo do aplicativo. Nós mudamos /hello para /helloworld.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
...
Resources:
  HelloWorldFunction:
    ...
    Properties:
      ...
      Events:
        HelloWorld:
          Type: Api
          Properties:
            Path: /helloworld
            Method: get
```

Como o recurso Amazon API Gateway não é compatível com a API do AWS serviço, ele executa AWS SAMCLI automaticamente uma CloudFormation implantação. Veja a seguir um exemplo de saída:

```
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 ServerlessRestApiDeployment8cf30e	AWS::ApiGateway::Deployment -	d3cd
UPDATE_IN_PROGRESS HelloWorldFunctionHelloWorldPermi	AWS::Lambda::Permission Requested update requires the creation of a new physical resource; hence creating one.	ssionProd
UPDATE_IN_PROGRESS HelloWorldFunctionHelloWorldPermi	AWS::Lambda::Permission Resource creation Initiated	ssionProd
CREATE_IN_PROGRESS ServerlessRestApiDeployment8cf30e	AWS::ApiGateway::Deployment Resource creation Initiated	d3cd
CREATE_COMPLETE ServerlessRestApiDeployment8cf30e	AWS::ApiGateway::Deployment -	d3cd
UPDATE_IN_PROGRESS ServerlessRestApiProdStage	AWS::ApiGateway::Stage -	
UPDATE_COMPLETE ServerlessRestApiProdStage	AWS::ApiGateway::Stage -	
UPDATE_COMPLETE HelloWorldFunctionHelloWorldPermi	AWS::Lambda::Permission -	ssionProd
UPDATE_COMPLETE_CLEANUP_IN_PROGRE -	AWS::CloudFormation::Stack	sam-app
SS		
DELETE_IN_PROGRESS HelloWorldFunctionHelloWorldPermi	AWS::Lambda::Permission -	ssionProd
DELETE_IN_PROGRESS ServerlessRestApiDeployment47fc2d	AWS::ApiGateway::Deployment -	5f9d
DELETE_COMPLETE ServerlessRestApiDeployment47fc2d	AWS::ApiGateway::Deployment -	5f9d
UPDATE_COMPLETE AwsSamAutoDependencyLayerNestedSt	AWS::CloudFormation::Stack -	ack
DELETE_COMPLETE HelloWorldFunctionHelloWorldPermi	AWS::Lambda::Permission -	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-
BUFVM02PJIYF

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.
```

Saiba mais

Para obter uma descrição de todas as opções `sam sync`, consulte [mesma sincronização](#).

Monitore seu aplicativo sem servidor com AWS SAM

Depois de implantar a aplicação sem servidor, você poderá monitorá-la para fornecer insights sobre as operações dela e detectar anomalias, o que poderá ajudar na solução de problemas. Esta seção fornece detalhes sobre como monitorar a aplicação sem servidor. Isso inclui informações sobre como configurar a Amazon CloudWatch para notificá-lo quando detectar anomalias. Isso também fornece informações sobre como trabalhar com logs, incluindo o destaque de erros e dicas para a visualização, filtragem, busca e rastreamento de registros.

Tópicos

- [Usando o CloudWatch Application Insights para monitorar seus aplicativos AWS SAM sem servidor](#)
- [Trabalhando com logs AWS SAM](#)

Usando o CloudWatch Application Insights para monitorar seus aplicativos AWS SAM sem servidor

O Amazon CloudWatch Application Insights ajuda você a monitorar AWS os recursos em seus aplicativos para ajudar a identificar possíveis problemas. Ele pode analisar dados AWS de recursos em busca de sinais de problemas e criar painéis automatizados para visualizá-los. Você pode configurar o CloudWatch Application Insights para usar com seus aplicativos AWS Serverless Application Model (AWS SAM). Para saber mais sobre o CloudWatch Application Insights, consulte o [Amazon CloudWatch Application Insights](#) no Guia CloudWatch do usuário da Amazon.

Tópicos

- [Configurando o CloudWatch Application Insights com AWS SAM](#)
- [Próximas etapas](#)

Configurando o CloudWatch Application Insights com AWS SAM

Configure o CloudWatch Application Insights para seus AWS SAM aplicativos por meio da interface de linha de AWS SAM comando (AWS SAMCLI) ou por meio de seus AWS SAM modelos.

Configure por meio do AWS SAMCLI

Ao inicializar seu aplicativo com `sam init`, ative o CloudWatch Application Insights por meio do fluxo interativo ou usando a `--application-insights` opção.

Para ativar o CloudWatch Application Insights por meio do fluxo AWS SAMCLI interativo, insira **y** quando solicitado.

```
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]:
```

Para ativar o CloudWatch Application Insights com a `--application-insights` opção, faça o seguinte.

```
sam init --application-insights
```

Para saber mais sobre como usar o comando `sam init`, consulte [sam init](#).

Configurar por meio AWS SAM de modelos

Ative o CloudWatch Application Insights definindo `AWS::ApplicationInsights::Application` os recursos `AWS::ResourceGroups::Group` e em seus AWS SAM modelos.

```
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**— Cria um grupo para organizar seus AWS recursos a fim de gerenciar e automatizar tarefas em um grande número de recursos ao mesmo tempo. Aqui, você cria um grupo de recursos para usar com o CloudWatch Application Insights. Para obter mais informações sobre esse tipo de recurso, consulte [AWS::ResourceGroups::Group](#) no AWS CloudFormation Guia do usuário.
- **AWS::ApplicationInsights::Application**— Configura o CloudWatch Application Insights para o grupo de recursos. Para obter mais informações sobre esse tipo de recurso, consulte [AWS::ApplicationInsights::Application](#) no AWS CloudFormation Guia do usuário.

Ambos os recursos são transmitidos automaticamente CloudFormation na implantação do aplicativo. Você pode usar a CloudFormation sintaxe em seu AWS SAM modelo para configurar ainda mais o CloudWatch Application Insights. Para obter mais informações, consulte [Usar CloudFormation modelos](#) no Guia do CloudWatch usuário da Amazon.

Ao usar o `sam init --application-insights` comando, esses dois recursos são gerados automaticamente em seu AWS SAM modelo. Veja aqui um exemplo de um modelo gerado.

```

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
```

Próximas etapas

Depois de configurar o CloudWatch Application Insights, use `sam build` para criar seu aplicativo e `sam deploy` implantá-lo. Todos os recursos compatíveis com o CloudWatch Application Insights serão configurados para monitoramento.

- Para obter uma lista dos recursos compatíveis, consulte [Registros e métricas compatíveis](#) no Guia CloudWatch do usuário da Amazon.
- Para saber como acessar o CloudWatch Application Insights, consulte [Access CloudWatch Application Insights](#) no Guia CloudWatch do usuário da Amazon.

Trabalhando com logins AWS SAM

Para simplificar a solução de problemas, o AWS SAMCLI tem um comando chamado [logs de spam](#). Esse comando permite que você busque registros gerados pela sua função de evento do Lambda na linha de comando.

Note

O `sam logs` comando funciona para todas as AWS Lambda funções, não apenas para aquelas que você implanta usando AWS SAM.

Buscando registros por CloudFormation pilha

Quando sua função faz parte de uma CloudFormation pilha, você pode buscar registros usando o ID lógico da função:

```
sam logs -n HelloWorldFunction --stack-name mystack
```

Buscando registros pelo nome da função de eventos do Lambda

Ou você pode buscar registros usando o nome da função:

```
sam logs -n mystack-HelloWorldFunction-1FJ8PD
```

Registros de rejeitos

Adicione a opção `--tail` para aguardar os novos logs e vê-los quando eles chegam. Isso é útil durante a implantação ou quando você está solucionando um problema de produção.

```
sam logs -n HelloWorldFunction --stack-name mystack --tail
```

Visualizando registros para um intervalo de tempo específico

Você pode visualizar os registros de um intervalo de tempo específico usando as opções `-s` e `-e`:

```
sam logs -n HelloWorldFunction --stack-name mystack -s '10min ago' -e '2min ago'
```

Filtragem de logs

Use a opção `--filter` para encontrar rapidamente logs que correspondam a termos, frases ou valores em seus eventos de log:

```
sam logs -n HelloWorldFunction --stack-name mystack --filter "error"
```

Na saída, AWS SAM CLI sublinha todas as ocorrências da palavra "erro" para que você possa localizar facilmente a palavra-chave do filtro na saída do log.

Destaques de erros

Quando sua função do Lambda falha ou atinge o tempo limite, o AWS SAM CLI destaca a mensagem de tempo limite em vermelho. Isso ajuda você a localizar facilmente as execuções específicas que estão expirando em um fluxo gigante de saída do log.

Impressão bonita em JSON

Se suas mensagens de registro imprimirem strings JSON, o AWS SAM CLI imprimirá automaticamente o JSON para ajudar você a analisar e entender visualmente o JSON.

AWS SAM referência

Esta seção contém material AWS SAM de referência. Isso inclui material de referência sobre a CLI do AWS SAM, como informações de referência sobre comandos da CLI do AWS SAM e informações adicionais sobre a CLI do AWS SAM, como configuração, controle de versão e informações sobre solução de problemas. Além disso, esta seção inclui informações de referência sobre a AWS SAM especificação e o AWS SAM modelo, como informações de referência sobre conectores, repositórios de imagens e implantações.

AWS SAM especificação e o AWS SAM modelo

A AWS SAM especificação é uma especificação de código aberto sob a licença Apache 2.0. A versão atual da AWS SAM especificação está disponível no [AWS SAM modelo](#). AWS SAM A especificação vem com uma sintaxe abreviada simplificada que você usa para definir as funções, eventos, APIs, configurações e permissões do seu aplicativo sem servidor.

Você interage com a AWS SAM especificação por meio do diretório do projeto do AWS SAM aplicativo, que são as pastas e os arquivos criados quando você executa o `sam init` comando. Esse diretório inclui o AWS SAM modelo, um arquivo importante que define seus AWS recursos. O AWS SAM modelo é uma extensão de um AWS CloudFormation modelo. Para obter a referência completa dos modelos CloudFormation , consulte [Referência do modelo](#) no AWS CloudFormation Guia do usuário.

AWS SAM CLI Referência de comando

A interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) é uma ferramenta de linha de comando que você pode usar com AWS SAM modelos e integrações de terceiros compatíveis para criar e executar seus aplicativos sem servidor.

Você pode usar os comandos AWS SAMCLI para desenvolver, testar e implantar seus aplicativos com tecnologia sem servidor no Nuvem AWS. Veja a seguir alguns exemplos de comandos AWS SAM CLI:

- `sam init`: se você for um usuário iniciante da CLI AWS SAM, poderá executar o comando `sam init` sem nenhum parâmetro para criar um aplicativo Hello World. O comando gera um AWS SAM modelo pré-configurado e um exemplo de código de aplicativo no idioma que você escolher.

- `sam local invoke` e `sam local start-api` — Use esses comandos para testar o código do seu aplicativo localmente, antes de implantá-lo no Nuvem AWS.
- `sam logs` — Use esse comando para buscar registros que sua função de evento Lambda gera. Isso pode ajudá-lo a testar e depurar seu aplicativo depois de implantá-lo no Nuvem AWS.
- `sam package` — Use esse comando para empacotar o código e as dependências do seu aplicativo em um pacote de implantação. Você precisa do pacote de implantação para carregar seu aplicativo no Nuvem AWS.
- `sam deploy` — Use esse comando para implantar seu aplicativo sem servidor no Nuvem AWS. Ele cria os AWS recursos e define as permissões e outras configurações definidas no AWS SAM modelo.

Para obter instruções sobre como instalar o AWS SAMCLI, consulte [Instale o AWS SAM CLI](#).

AWS SAM Modelos de políticas

Com AWS SAM, você pode escolher entre uma lista de modelos de política para definir o escopo das permissões de sua AWS Lambda função para os recursos que seu aplicativo usa. Para obter uma lista de modelos de política disponíveis, consulte [Tabela de modelos de política](#). Para obter informações gerais sobre modelos de políticas e AWS SAM, consulte [AWS SAM Modelos de políticas](#).

Tópicos

- [AWS SAM modelo](#)
- [AWS SAMCLIReferência de comando](#)
- [Arquivo de configuração do AWS SAM CLI](#)
- [AWS SAM referência do conector](#)
- [AWS SAM Modelos de políticas](#)
- [Repositórios de imagens para AWS SAM](#)
- [Telemetria no AWS SAMCLI](#)
- [Configure e gerencie o acesso a recursos em seu AWS SAM modelo](#)

AWS SAMCLIReferência de comando

Esta seção inclui informações de referência sobre comandos da CLI do AWS SAM. Isso inclui detalhes sobre o uso, uma lista abrangente das diferentes opções disponíveis para cada comando e informações adicionais. Quando aplicável, as informações adicionais incluem alguns detalhes, como argumentos, variáveis de ambiente e eventos. Consulte cada comando para obter detalhes. Para obter instruções sobre como instalar o AWS SAMCLI, consulte [Instale o AWS SAM CLI](#).

Tópicos

- [sam build](#)
- [sam delete](#)
- [sam deploy](#)
- [sam init](#)
- [sam list](#)
- [sam local callback](#)
- [sam local execution](#)
- [sam local generate-event](#)
- [SAM invocação local](#)
- [SAM API de inicialização local](#)
- [start-lambda do local do sam](#)
- [logs de spam](#)
- [sam package](#)
- [bootstrap do mesmo pipeline](#)
- [sam pipeline init](#)
- [sam publish](#)
- [sam remote callback](#)
- [sam remote execution](#)
- [invocação remota do sam](#)
- [evento de teste remoto do sam](#)
- [mesma sincronização](#)
- [sam traces](#)
- [sam validate](#)

sam build

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam build`.

- Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre o uso do AWS SAMCLI `sam build` comando, consulte [Introdução à construção com AWS SAM](#).

O comando `sam build` prepara um aplicativo para as etapas subsequentes do fluxo de trabalho de desenvolvimento, como testes locais ou implantação no Nuvem AWS.

Usage

```
$ sam build <arguments> <options>
```

Argumentos

Resource ID (ID do recurso)

Optional. Instrui AWS SAM a criar um único recurso declarado em um [AWS SAM modelo](#). Os artefatos de construção do recurso especificado serão os únicos disponíveis para comandos subsequentes no fluxo de trabalho, ou seja, `sam package` e `sam deploy`.

Opções

`--base-dir`, `-s` *DIRECTORY*

Resolve caminhos relativos para o código-fonte da função ou da camada em relação a esse diretório. Use essa opção se quiser alterar a forma como os caminhos relativos às pastas de código-fonte são resolvidos. Por padrão, os caminhos relativos são resolvidos com relação à localização do modelo AWS SAM .

Além dos recursos no aplicativo raiz ou na pilha que você está criando, essa opção também aplica aplicativos ou pilhas aninhados.

Essa opção se aplica aos seguintes tipos e propriedades de recursos:

- Tipo de recursos: `AWS::Serverless::Function` Propriedade: `CodeUri`

- Tipo de recurso: `AWS::Serverless::Function` Atributo do recurso: `Metadata` Entrada: `DockerContext`
- Tipo de recursos: `AWS::Serverless::LayerVersion` Propriedade: `ContentUri`
- Tipo de recursos: `AWS::Lambda::Function` Propriedade: `Code`
- Tipo de recursos: `AWS::Lambda::LayerVersion` Propriedade: `Content`

`--beta-features` | `--no-beta-features`

Permita ou negue recursos beta.

`--build-dir`, `-b` *DIRECTORY*

O caminho para um diretório onde os artefatos criados são armazenados. Esse diretório e todo o seu conteúdo são removidos com essa opção.

`--build-image` *TEXT*

O URI da imagem de contêiner que você deseja extrair para a compilação. Por padrão, o AWS SAM extrai a imagem de contêiner do Amazon ECR Public. Use essa opção para extrair a imagem de outro localização.

Especifique essa opção várias vezes. Cada instância dessa opção pode ter uma string ou um par de valores-chave. Se você especificar uma string, ela será a URI da imagem do contêiner a ser usada para todos os recursos do seu aplicativo. Por exemplo, `sam build --use-container --build-image amazon/aws-sam-cli-build-image-python3.8`. Se você especificar um par de valores-chave, a chave será o nome do recurso e o valor será o URI da imagem do contêiner a ser usada para esse recurso. Por exemplo `sam build --use-container --build-image Function1=amazon/aws-sam-cli-build-image-python3.8`. Com pares de valores-chave, você pode especificar imagens de contêiner diferentes para recursos diferentes.

Essa opção só se aplica se a opção `--use-container` for especificada, caso contrário, ocorrerá um erro.

`--build-in-source` | `--no-build-in-source`

Providencie `--build-in-source` para criar seu projeto diretamente na pasta de origem.

A opção `--build-in-source` oferece suporte aos seguintes runtimes e métodos de compilação:

- Runtimes — Qualquer runtime Node.js suportado pela opção [`sam init --runtime`](#).

- Métodos de compilação — Makefile, esbuild.

A opção `--build-in-source` não é compatível com as seguintes opções:

- `--hook-name`
- `--use-container`

Padrão: `--no-build-in-source`

`--cached` | `--no-cached`

Habilite ou desabilite as compilações em cache. Use essa opção para reutilizar artefatos de construção que não foram alterados em relação às compilações anteriores. AWS SAM avalia se você alterou algum arquivo no diretório do seu projeto. Por padrão, as compilações não são armazenadas em cache. Se a opção `--no-cached` for invocada, ela substituirá a configuração `cached = true` em `samconfig.toml`.

 Note

O AWS SAM não avalia se você alterou os módulos de terceiros dos quais seu projeto depende, nos quais você não forneceu uma versão específica. Por exemplo, se sua função Python incluir um `requirements.txt` arquivo com a entrada `requests=1.x` e a versão mais recente do módulo de solicitação mudar de 1.1 para 1.2, AWS SAM não extrairá a versão mais recente até que você execute uma compilação sem cache.

`--cache-dir`

O diretório em que os artefatos do cache são armazenados quando `--cached` é especificado. O diretório padrão do cache é `.aws-sam/cache`.

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “`samconfig.toml`” na raiz do diretório do projeto. Para obter

mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--container-env-var, -e TEXT`

Passe as variáveis de ambiente para um contêiner de build. Especifique essa opção várias vezes. Cada instância dessa opção usa um par de valores-chave, em que a chave é a variável de recurso e ambiente, e o valor é o valor da variável de ambiente. Por exemplo: `--container-env-var Function1.GITHUB_TOKEN=TOKEN1 --container-env-var Function2.GITHUB_TOKEN=TOKEN2`.

Essa opção só se aplica se a opção `--use-container` for especificada, caso contrário, ocorrerá um erro.

`--container-env-var-file, -ef PATH`

O caminho e o nome do arquivo JSON que contém valores para as variáveis de ambiente do contêiner. Para obter mais informações sobre arquivos variáveis de ambiente do contêiner, consulte [As variáveis de ambiente do contêiner](#).

Essa opção só se aplica se a opção `--use-container` for especificada, caso contrário, ocorrerá um erro.

`--debug`

Ativa o registro de depuração para imprimir mensagens de depuração que o AWS SAMCLI gera e para exibir timestamps.

`--docker-network TEXT`

Especifica o nome ou ID de uma rede Docker existente à qual os contêineres do Docker Lambda devem se conectar, juntamente com a rede de ponte padrão. Se não for especificado, os contêineres do Lambda se conectarão somente à rede Docker da ponte padrão.

`--exclude, -x`

O nome do (s) recurso (s) a ser excluído do `sam build`. Por exemplo, se seu modelo contiver `Function1`, `Function2` e `Function3` e você executar `sam build --exclude Function2`, somente `Function1` e `Function3` será criado.

`--help`

Mostra esta mensagem e sai.

`--hook-name` *TEXT*

O nome do hook usado para estender a funcionalidade AWS SAMCLI.

Valores aceitos: terraform.

`--manifest` , `-m` *PATH*

O caminho para um arquivo de manifesto de dependência personalizado (por exemplo, package.json) a ser usado em vez do padrão.

`--mount-symlinks`

Garante que a CLI do AWS SAM sempre monte symlinks que estão presentes nos arquivos a serem criados ou invocados. Isso se aplica somente aos symlinks no diretório de nível superior (ou seja, links simbólicos que estão diretamente na raiz da função). Por padrão, os symlinks não são montados, exceto aqueles necessários para usar o build-in-source no node_modules no NodeJS.

`--no-use-container`

Uma opção que permite usar o kit de ferramentas do IDE para definir o comportamento padrão. Você também pode usar `sam build --no-use-container` para executar uma compilação em sua máquina local em vez de um contêiner do Docker.

`--output` *[text | json]*

O formato da saída do comando:

- `text`— Imprime saídas regulares e legíveis por humanos. Este é o valor padrão.
- `json`— Imprime saídas estruturadas e legíveis por máquina. Use esse formato quando quiser consumir a saída programaticamente, como em CI/CD pipelines, extensões do IDE e ferramentas. AI-assisted

Quando você especifica `--output json`, o AWS SAMCLI imprime um único objeto JSON após a conclusão do comando. Em caso de sucesso, esse objeto inclui campos como o status da construção, o diretório da construção, o arquivo de modelo e uma lista dos recursos que foram criados. Em caso de falha, inclui um `error` objeto que descreve o que deu errado.

`--parallel`

Permite compilações paralelas. Use essa opção para criar as funções e camadas do seu AWS SAM modelo em paralelo. Por padrão, as funções e camadas são criadas em sequência.

--parameter-overrides

(Opcional) Uma string que contém substituições de CloudFormation parâmetros codificadas como pares de valores-chave. Usa o mesmo formato do AWS Command Line Interface (AWS CLI). Por exemplo: 'ParameterKey=KeyPairName, ParameterValue=MyKey ParameterKey=InstanceType, ParameterValue=t1.micro'. Essa opção não é compatível com o --hook-name.

--profile *TEXT*

O perfil específico do seu arquivo de credenciais que recebe as AWS credenciais.

--region *TEXT*

O Região da AWS para implantar em. Por exemplo, us-east-1.

--save-params

Salve os parâmetros fornecidos na linha de comando no arquivo AWS SAM de configuração.

--skip-prepare-infra

Ignora a fase de preparação se nenhuma alteração na infraestrutura tiver sido feita. Execute com a opção --hook-name.

--skip-pull-image

Especifica se o comando deve ignorar a extração da imagem mais recente do Docker para o runtime do Lambda.

--template-file, --template, -t *PATH*

O caminho e o nome do arquivo de AWS SAM modelo[default: template.[yaml|yml]]. Essa opção não é compatível com o --hook-name.

--terraform-project-root-path

O caminho relativo ou absoluto para o diretório de nível superior contendo seus arquivos de configuração Terraform ou código-fonte da função. Se esses arquivos estiverem localizados fora do diretório que contém seu módulo raiz Terraform, use essa opção para especificar seu caminho absoluto ou relativo. Essa opção exige que --hook-name seja definida como terraform.

--use-buildkit | --no-use-buildkit

Ative ou desative o uso do BuildKit plug-in para compilações de imagens de contêiner. Isso usará o Finch ou DockerCLIs. Por padrão, as compilações usam o construtor legado do SDK. Se a

opção `--no-use-buildkit` for invocada, ela substituirá a configuração `use-buildkit = true` em `samconfig.toml`.

`--use-container, -u`

Se suas funções dependerem de pacotes que tenham dependências compiladas de forma nativa, use essa opção para criar sua função dentro de um contêiner Lambda-like Docker.

Exemplo

Para obter um exemplo detalhado e uma explicação aprofundada sobre o uso do subcomando `sam build`, consulte [Introdução à construção com AWS SAM](#).

sam delete

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam delete`.

Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam delete` comando exclui um AWS SAM aplicativo excluindo a CloudFormation pilha, os artefatos que foram empacotados e implantados no Amazon S3 e no Amazon ECR e o arquivo de modelo. AWS SAM

Esse comando também verifica se há uma pilha complementar do Amazon ECR implantada e, em caso afirmativo, solicita ao usuário a exclusão dessa pilha e dos repositórios do Amazon ECR. Se `--no-prompts` for especificado, as pilhas complementares e os repositórios do Amazon ECR serão excluídos por padrão.

Usage

```
$ sam delete <options>
```

Opções

`--config-env TEXT`

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é `default`. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é `samconfig.toml` na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--debug`

Ativa o registro de depuração para imprimir a mensagem de depuração que o AWS SAMCLI gera e para exibir timestamps.

`--help`

Mostra esta mensagem e sai.

`--no-prompts`

Especifique essa opção para AWS SAM operar no modo não interativo. O nome da pilha deve ser fornecido, com a opção `--stack-name` ou no arquivo de configuração `toml`.

`--profile` *TEXT*

O perfil específico do seu arquivo de credenciais que obtém as AWS credenciais.

`--region` *TEXT*

A AWS região para a qual implantar. Por exemplo, `us-east-1`.

`--s3-bucket`

O caminho do bucket do Amazon S3 que deseja excluir.

`--s3-prefix`

O prefixo do bucket do Amazon S3 que deseja excluir.

`--save-params`

Salve os parâmetros que você fornece na linha de comando no arquivo AWS SAM de configuração.

`--stack-name` *TEXT*

O nome da CloudFormation pilha que você deseja excluir.

Exemplos

O comando a seguir exclui a pilha MY-STACK.

```
$ sam delete --stack-name MY-STACK
```

O seguinte comando exclui a pilha MY-STACK e o bucket do S3 sam-s3-demo-bucket:

```
$ sam delete \  
  --stack-name MyStack \  
  --s3-bucket MySAMBucket
```

sam deploy

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam deploy`.

- Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre o uso do AWS SAMCLI `sam deploy` comando, consulte [Introdução à implantação com AWS SAM](#).

O `sam deploy` comando implanta um aplicativo para o usuário Nuvem AWS . AWS CloudFormation

Usage

```
$ <environment variables> sam deploy <options>
```

Variáveis de ambiente

SAM_CLI_POLL_DELAY

Defina a variável de `SAM_CLI_POLL_DELAY` ambiente com um valor de segundos em seu shell para configurar a frequência com que a CLI do AWS SAM verifica o estado da CloudFormation pilha, o que é útil ao ver a limitação de. CloudFormation Essa variável de ambiente é usada para a sondagem de chamadas de API `describe_stack`, que são feitas durante a execução de `sam deploy`.

Veja um exemplo dessa variável:

```
$ SAM_CLI_POLL_DELAY=5 sam deploy
```

Opções

`--capabilities` *LIST*

Uma lista de recursos que você deve especificar para permitir CloudFormation a criação de determinadas pilhas. Alguns modelos de pilha podem incluir recursos que afetam as permissões em sua Conta da AWS, por exemplo, criando novos usuários AWS Identity and Access Management (IAM). Para essas pilhas, você deve confirmar explicitamente seus recursos especificando essa opção. Os únicos valores válidos são `CAPABILITY_IAM` e `CAPABILITY_NAMED_IAM`. Se você tiver recursos do IAM, então poderá especificar qualquer recurso. Se tiver recursos do IAM com nomes personalizados, então você deverá especificar `CAPABILITY_NAMED_IAM`. Se você não especificar essa opção, a operação retornará um erro `InsufficientCapabilities`.

Quando você implanta uma aplicação que contém aplicações aninhadas, deve usar `CAPABILITY_AUTO_EXPAND` para reconhecer que a aplicação contém aplicações aninhadas. Para obter mais informações, consulte [Implantar aplicativos aninhados](#).

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é `default`. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é `samconfig.toml` na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--confirm-changeset` | `--no-confirm-changeset`

Solicite a confirmação se o AWS SAMCLI implanta o conjunto de alterações computado.

`--debug`

Ative o registro de depuração para imprimir a mensagem de depuração que o AWS SAMCLI gera e para exibir timestamps.

`--disable-rollback | --no-disable-rollback`

Especifique se deseja reverter sua CloudFormation pilha se ocorrer um erro durante uma implantação. Por padrão, se houver um erro durante uma implantação, sua CloudFormation pilha será revertida para o último estado estável. Se você especificar `--disable-rollback` e ocorrer um erro durante uma implantação, os recursos que foram criados ou atualizados antes da ocorrência do erro não serão revertidos.

`--express | --no-express`

Especifica se o modo CloudFormation expresso deve ser usado. Quando ativas, as operações de pilha são concluídas quando a configuração do recurso é aplicada. Os recursos continuam ficando prontos em segundo plano. O tempo de implantação pode ser reduzido em até 4x.

O AWS SAMCLI passa o comportamento de reversão para o CloudFormation `DeploymentConfig` parâmetro quando você também especifica. `--disable-rollback`

O valor padrão é `--no-express`.

`--fail-on-empty-changeset | --no-fail-on-empty-changeset`

Especifique se deseja retornar um código de saída diferente de zero se não houver alterações a serem feitas na pilha. O comportamento padrão é retornar um código de saída diferente de zero.

`--force-upload`

Especifique essa opção para fazer upload de artefatos, mesmo que eles correspondam aos artefatos existentes no bucket do Amazon S3. Os artefatos correspondentes são sobrescritos.

`--guided, -g`

Especifique essa opção para que o AWS SAMCLI use prompts que o guie durante a implantação.

`--help`

Mostre esta mensagem e saia.

`--image-repositories TEXT`

Um mapeamento de funções para o URI do repositório Amazon ECR. Funções de referência por meio de sua ID lógica. Veja um exemplo a seguir:

```
$ sam deploy --image-repositories Function1=123456789012.dkr.ecr.us-east-1.amazonaws.com/my-repo
```

Você pode especificar esta opção várias vezes em um único comando.

`--image-repository` *TEXT*

O nome do repositório Amazon ECR em que esse comando carrega a imagem da sua função. Essa opção é necessária para funções declaradas com o tipo de pacote Image.

`--kms-key-id` *TEXT*

O ID de uma chave AWS Key Management Service (AWS KMS) usada para criptografar artefatos que estão em repouso no bucket do Amazon S3. Se você não especificar essa opção, AWS SAM use as chaves de S3-managed criptografia da Amazon.

`--metadata`

Um mapa de metadados para anexar a todos os artefatos referenciados em seu modelo.

`--no-execute-changeset`

Indica se o conjunto de alterações deve ser aplicado. Especifique essa opção se quiser visualizar as alterações da pilha antes de aplicar o conjunto de alterações. Esse comando cria um conjunto de alterações CloudFormation e, em seguida, sai sem aplicar o conjunto de alterações. Para aplicar o conjunto de alterações, execute o mesmo comando sem essa opção.

`--no-progressbar`

Não exiba uma barra de progresso ao fazer o upload de artefatos para o Amazon S3.

`--notification-arns` *LIST*

Uma lista de ARNs de tópicos do Amazon Simple Notification Service (Amazon SNS) CloudFormation associados à pilha.

`--on-failure` [ROLLBACK | DELETE | DO_NOTHING]

Especifique a ação a ser tomada quando uma pilha falhar na criação.

As seguintes opções estão disponíveis:

- ROLLBACK — Reverte a pilha para um estado anterior em boas condições.
- DELETE — Reverte a pilha para um estado anterior em boas condições, se existir um. Caso contrário, exclui a pilha.
- DO_NOTHING — Nem reverte nem exclui a pilha. O efeito é o mesmo do `--disable-rollback`.

O comportamento padrão é ROLLBACK.

 Note

Você pode especificar a opção `--disable-rollback` ou a opção `--on-failure`, mas não ambos.

`--output` *[text | json]*

O formato da saída do comando:

- `text`— Imprime saídas regulares e legíveis por humanos. Este é o valor padrão.
- `json`— Imprime saídas estruturadas e legíveis por máquina. Use esse formato quando quiser consumir a saída programaticamente, como em CI/CD pipelines, extensões do IDE e ferramentas. AI-assisted

Uma implantação é executada por um período de tempo. Quando você especifica `--output json`, ele AWS SAMCLI imprime um fluxo de objetos JSON, um por linha, à medida que a implantação avança. Cada objeto inclui um `type` campo que identifica o tipo de informação que ele contém, como o conjunto de alterações, eventos de implantação, saídas da pilha e o resultado final. Em caso de falha, o objeto final inclui um `error` objeto que descreve o que deu errado.

 Note

A `--output json` opção não é compatível com opções que exigem solicitações interativas, como `--guided` e `--confirm-changeset`

`--parameter-overrides` *LIST*

Uma string que contém substituições de CloudFormation parâmetros codificadas como pares de valores-chave. Cada substituição usa o formato `ParameterKey=name,ParameterValue=value`. As substituições múltiplas são separadas por espaços. Veja dois exemplos a seguir:

```
$ 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*

O perfil específico do seu arquivo de credenciais que recebe as AWS credenciais.

`--region` *TEXT*

O Região da AWS para implantar em. Por exemplo, us-east-1.

`--resolve-image-repos`

Crie automaticamente repositórios do Amazon ECR para uso no empacotamento e implantação em implantações não guiadas. Esta opção se aplica apenas às funções e camadas com o `PackageType: Image` especificado. Se você especificar a opção `--guided`, então o AWS SAMCLI ignora o `--resolve-image-repos`.

 Note

Se você criar AWS SAM automaticamente quaisquer repositórios do Amazon ECR para funções ou camadas com essa opção e, posteriormente, excluir essas funções ou camadas do seu AWS SAM modelo, os repositórios correspondentes do Amazon ECR serão automaticamente excluídos.

`--resolve-s3`

Crie automaticamente um bucket do Amazon S3 para uso no empacotamento e implantação em implantações não guiadas. Se você especificar a opção `--guided`, então a CLI AWS SAM ignora o `--resolve-s3`. Se você especificar as opções `--s3-bucket` e `--resolve-s3`, ocorrerá um erro.

`--role-arn` *TEXT*

O nome de recurso da Amazon (ARN) de uma função do IAM que é CloudFormation assumida ao aplicar o conjunto de alterações.

`--s3-bucket` *TEXT*

O nome do bucket do Amazon S3 em que esse comando carrega seu CloudFormation modelo. Se seu modelo for maior que 51.200 bytes, a opção `--s3-bucket` ou a opção `--resolve-s3` serão obrigatórias. Se você especificar as opções `--s3-bucket` e `--resolve-s3`, ocorrerá um erro.

`--s3-prefix` *TEXT*

O prefixo adicionado aos nomes dos artefatos que são enviados ao bucket do Amazon S3. O nome do prefixo é um nome de caminho (nome da pasta) para o bucket do Amazon S3.

`--save-params`

Salve os parâmetros que você fornece na linha de comando no arquivo AWS SAM de configuração.

`--signing-profiles` *LIST*

A lista de perfis de assinatura com os quais assinar seus pacotes de implantação. Essa opção usa uma lista de pares de valores-chave, em que a chave é o nome da função ou camada a ser assinada e o valor é o perfil de assinatura, com um proprietário de perfil opcional delimitado por `:`. Por exemplo, `FunctionNameToSign=SigningProfileName1`
`LayerNameToSign=SigningProfileName2:SigningProfileOwner`.

`--stack-name` *TEXT*

(Obrigatório) O nome da CloudFormation pilha na qual você está implantando. Se você especificar uma pilha existente, o comando atualizará a pilha. Se você especificar uma nova pilha, o comando a criará.

`--tags` *LIST*

Uma lista de tags para associar à pilha que é criada ou atualizada. CloudFormation também propaga essas tags para recursos na pilha que as suportam.

`--template-file`, `--template`, `-t` *PATH*

O caminho e o nome do arquivo em que seu AWS SAM modelo está localizado.

 Note

Se você especificar essa opção, AWS SAM implantará somente o modelo e os recursos locais para os quais ele aponta.

`--use-json`

Saída JSON para o CloudFormation modelo. A saída padrão é YAML.

Exemplo

Para obter um exemplo detalhado e uma explicação aprofundada sobre o uso do subcomando `sam deploy`, consulte [Introdução à implantação com AWS SAM](#).

sam init

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam init`.

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre o uso do AWS SAMCLI `sam init` comando, consulte [Crie seu aplicativo em AWS SAM](#).

O comando `sam init` fornece opções para inicializar um novo aplicativo sem servidor.

Usage

```
$ sam init <options>
```

Opções

`--app-template` *TEXT*

O identificador do modelo de aplicativo gerenciado que você deseja usar. Se você não tiver certeza, chame `sam init` sem opções para um fluxo de trabalho interativo.

Esse parâmetro é obrigatório se `--no-interactive` for especificado e `--location` não for fornecido.

Esse parâmetro só está disponível na versão 0.30.0 e posterior do AWS SAM CLI. A especificação desse parâmetro com uma versão anterior resulta em um erro.

`--application-insights` | `--no-application-insights`

Ative o monitoramento CloudWatch do Amazon Application Insights para seu aplicativo. Para saber mais, consulte [Usando o CloudWatch Application Insights para monitorar seus aplicativos AWS SAM sem servidor](#).

A opção padrão é `--no-application-insights`.

`--architecture, -a [ x86_64 | arm64 ]`

A arquitetura do conjunto de instruções para as Funções de Eventos Lambda do seu aplicativo. Especifique um dos `x86_64` ou `arm64`.

`--base-image [amazon/dotnet8-base | amazon/dotnet6-base | amazon/java25-base | amazon/java21-base | amazon/java17-base | amazon/java17.al2023-base | amazon/java11-base | amazon/java11.al2023-base | amazon/java8.al2023-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/ruby4.0-base | amazon/ruby3.4-base | amazon/ruby3.3-base | amazon/ruby3.2-base]`

A imagem base do seu aplicativo. Essa opção se aplica somente quando o tipo de pacote é `Image`.

Esse parâmetro é necessário se `--no-interactive` for especificado, `--package-type` for especificado como `Image` e `--location` não for especificado.

`--config-env TEXT`

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file PATH`

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “`samconfig.toml`” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--debug`

Ativa o registro de depuração para imprimir as mensagens de depuração que o AWS SAM CLI gera e para exibir timestamps.

`--dependency-manager, -d [ gradle | mod | maven | bundler | npm | cli-package | pip ]`

O gerenciador de dependências do seu tempo de execução do Lambda.

--extra-content

Substitua quaisquer parâmetros personalizados na configuração `cookiecutter.json` do modelo, por exemplo, `{"customParam1": "customValue1", "customParam2": "customValue2"}`.

--help, -h

Mostra esta mensagem e sai.

--location, -l *TEXT*

O local do modelo ou do aplicativo (Git, Mercurial, arquivo.zip HTTP/HTTPS, caminho).

Esse parâmetro é necessário se `--no-interactive` for especificado e `--runtime`, `--name`, e `--app-template` não for fornecido.

Para repositórios Git, você deve usar a localização da raiz do repositório.

Para caminhos locais, o modelo deve estar no formato de arquivo.zip ou [Cookiecutter](#).

--name, -n *TEXT*

O nome do seu projeto a ser gerado como um diretório.

Esse parâmetro é obrigatório se `--no-interactive` for especificado e `--location` não for fornecido.

--no-input

Desativa a solicitação do Cookiecutter e aceita os valores `vcfdefault` que são definidos na configuração do modelo.

--no-interactive

Desative a solicitação interativa para os parâmetros de inicialização e falha se algum valor necessário estiver ausente.

--output [*text* | *json*]

O formato da saída do comando:

- `text`— Imprime saídas regulares e legíveis por humanos. Este é o valor padrão.
- `json`— Imprime uma saída estruturada e legível por máquina. Use esse formato quando quiser consumir a saída programaticamente, como em CI/CD pipelines, extensões de IDE e ferramentas. AI-assisted

Quando você especifica `--output json`, AWS SAMCLI imprime um único objeto JSON após a conclusão do comando. Em caso de sucesso, esse objeto inclui campos como o diretório do projeto e o arquivo de modelo. Você pode encadear a saída em comandos subsequentes, como `sam build`. Em caso de falha, inclui um `error` objeto que descreve o que deu errado.

 Note

A `--output json` opção não é compatível com o fluxo de trabalho interativo. Para usá-lo, especifique uma execução totalmente não interativa fornecendo todas as opções necessárias.

`--output-dir, -o PATH`

O local em que o aplicativo inicializado é gerado.

`--package-type [ Zip | Image ]`

O tipo de pacote do aplicativo de exemplo. Zip cria um arquivo .zip e Image cria uma imagem de contêiner.

`--runtime, -r [dotnet8 | dotnet6 | java25 | java21 | java17 | java17.al2023 | java11 | java11.al2023 | java8.al2023 | nodejs24.x | nodejs22.x | nodejs20.x | nodejs18.x | nodejs16.x | python3.14 | python3.13 | python3.12 | python3.11 | python3.10 | python3.9 | python3.8 | ruby4.0 | ruby3.4 | ruby3.3 | ruby3.2]`

O tempo de execução do Lambda do seu aplicativo. Essa opção se aplica somente quando o tipo de pacote é Zip.

Esse parâmetro é necessário se `--no-interactive` for especificado, `--package-type` for especificado como Zip e `--location` não for especificado.

`--save-params`

Salve os parâmetros que você fornece na linha de comando no arquivo AWS SAM de configuração.

`--tracing` | `--no-tracing`

Ative o AWS X-Ray rastreamento para suas funções do Lambda.

Exemplo

Para obter um exemplo detalhado e uma explicação aprofundada sobre o uso do subcomando `sam init`, consulte [Crie seu aplicativo em AWS SAM](#).

sam list

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam list`.

Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O comando `sam list` gera informações importantes sobre os recursos em seu aplicativo sem servidor e o estado do seu aplicativo sem servidor. Use `sam list` antes e depois da implantação para ajudar durante o desenvolvimento local e na nuvem.

Usage

```
$ sam list <options> <subcommand>
```

Opções

--help, -h

Mostre esta mensagem e saia.

Subcomandos

endpoints

Exibe uma lista de endpoints locais e de nuvem da sua CloudFormation pilha. Para obter mais informações, consulte [sam list endpoints](#).

resources

Exibe os recursos em seu modelo AWS Serverless Application Model (AWS SAM) que são criados AWS CloudFormation na implantação. Para obter mais informações, consulte [sam list resources](#).

stack-outputs

Exibe as saídas da sua CloudFormation pilha a partir de um modelo AWS SAM ou CloudFormation . Para obter mais informações, consulte [sam list stack-outputs](#).

sam list endpoints

Esta página fornece informações de referência para o AWS Serverless Application Model `sam list endpoints` subcomando Command Line Interface (AWS SAMCLI).

Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam list endpoints` subcomando exibe uma lista de endpoints locais e de nuvem da sua CloudFormation pilha. Você pode interagir com esses recursos por meio dos comandos `sam local` e `sam sync`.

AWS Lambda e os tipos de recursos do Amazon API Gateway são compatíveis com esse comando.

Note

Domínios personalizados são suportados quando configurados para seus recursos do Amazon API Gateway. Esse comando exibirá o domínio personalizado em vez do endpoint padrão.

Usage

```
$ sam list endpoints <options>
```

Opções

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados.

Valor padrão: `default`

Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *TEXT*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados.

Valor padrão: O `samconfig.toml` no diretório de trabalho atual.

Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--debug`

Ative o registro de depuração para imprimir mensagens de depuração geradas pelo AWS SAMCLI com timestamps.

`--help`, `-h`

Mostre esta mensagem e saia.

`--output` `[json|table]`

Especifique o formato para gerar resultados.

Valor padrão: `table`

`--profile` *TEXT*

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

`--region` *TEXT*

Defina a AWS região do serviço. Por exemplo, `.us-east-1`

`--save-params`

Salve os parâmetros que você fornece na linha de comando no arquivo AWS SAM de configuração.

`--stack-name` *TEXT*

Nome da CloudFormation pilha implantada. O nome da pilha pode ser encontrado no `samconfig.toml` arquivo do seu aplicativo ou no arquivo de configuração designado.

Quando essa opção não for especificada, os recursos locais definidos em seu modelo serão exibidos.

`--template-file`, `--template`, `-t` *PATH*

AWS SAM arquivo de modelo.

Valor padrão: `template.[yaml|yml|json]`

Exemplos

Exiba uma saída, no formato json, dos endpoints de recursos implantados da sua CloudFormation pilha chamada. `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": "uj80ue2o2",
    "CloudEndpoint": [
      "https://uj80ue2o2.execute-api.us-east-1.amazonaws.com/Prod",
      "https://uj80ue2o2.execute-api.us-east-1.amazonaws.com/Stage"
    ],
    "Methods": [
      "/hello['get']"
    ]
  }
]
```

sam list resources

Esta página fornece informações de referência para o AWS Serverless Application Model `sam list resources` subcomando Command Line Interface (AWS SAMCLI).

Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam list resources` subcomando exibe os recursos em seu modelo AWS Serverless Application Model (AWS SAM) que são criados AWS CloudFormation pela AWS SAM transformação na implantação.

Use `sam list resources` com um AWS SAM modelo antes da implantação para ver os recursos que serão criados. Forneça um nome de CloudFormation pilha para visualizar uma lista consolidada que inclui recursos implantados.

Note

Para gerar uma lista de recursos do seu AWS SAM modelo, uma transformação local do seu modelo é executada. Os recursos que serão implantados sob condições, como em uma região específica, estão incluídos nessa lista.

Usage

```
$ sam list resources <options>
```

Opções

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados.

Valor padrão: `default`

Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *TEXT*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados.

Valor padrão: `o samconfig.toml` no diretório de trabalho atual.

Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--debug`

Ative o registro de depuração para imprimir mensagens de depuração geradas pelo AWS SAMCLI com timestamps.

`--help, -h`

Mostre esta mensagem e saia.

`--output [json|table]`

Especifique o formato para gerar resultados.

Valor padrão: `table`

`--profile TEXT`

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

`--region TEXT`

Defina a AWS região do serviço. Por exemplo, `.us-east-1`

`--save-params`

Salve os parâmetros que você fornece na linha de comando no arquivo AWS SAM de configuração.

`--stack-name TEXT`

Nome da CloudFormation pilha implantada. O nome da pilha pode ser encontrado no `samconfig.toml` arquivo do seu aplicativo ou no arquivo de configuração designado.

Quando fornecida, a lógica IDs do recurso do seu modelo será mapeada para a entrada física IDs correspondente. CloudFormation Para saber mais sobre física IDs, consulte [Campos de recursos](#) no Guia AWS CloudFormation do usuário.

Quando essa opção não for especificada, os recursos locais definidos em seu modelo serão exibidos.

`--template-file, --template, -t PATH`

AWS SAM arquivo de modelo.

Valor padrão: `template.[yaml|yml|json]`

Exemplos

Exiba uma saída, em formato de tabela, dos recursos locais do seu AWS SAM modelo e dos recursos implantados da sua CloudFormation pilha chamada. `test-stack` Execute no mesmo diretório do seu modelo local.

```
$ 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-1QH7CP0CBL2IK	
HelloWorldFunctionRole	sam-app-test-list-
HelloWorldFunctionRole-SRJDMJ6F7F41	
ServerlessRestApi	uj80uoe2o2
ServerlessRestApiDeployment47fc2d5f9d	pncw5f
ServerlessRestApiProdStage	Prod
ServerlessRestApiDeploymentf5716dc08b	-

sam list stack-outputs

Esta página fornece informações de referência para o AWS Serverless Application Model `sam list stack-outputs` subcomando Command Line Interface (AWS SAMCLI).

Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam list stack-outputs` subcomando exibe as saídas da sua AWS CloudFormation pilha a partir de um AWS Serverless Application Model (AWS SAM) ou modelo. CloudFormation Para obter mais informações sobre Outputs, consulte [Resultados](#) no Guia do usuário do AWS CloudFormation.

Usage

```
$ sam list stack-outputs <options>
```

Opções

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados.

Valor padrão: `default`

Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *TEXT*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados.

Valor padrão: O `samconfig.toml` no diretório de trabalho atual.

Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--debug`

Ative o registro de depuração para imprimir mensagens de depuração geradas pelo AWS SAMCLI com timestamps.

`--help`, `-h`

Mostre esta mensagem e saia.

`--output` `[json|table]`

Especifique o formato para gerar resultados.

Valor padrão: `table`

`--profile` *TEXT*

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

`--region` *TEXT*

Defina a AWS região do serviço. Por exemplo, `.us-east-1`

`--save-params`

Salve os parâmetros fornecidos na linha de comando no arquivo AWS SAM de configuração.

`--stack-name` *TEXT*

Nome da CloudFormation pilha implantada. O nome da pilha pode ser encontrado no `samconfig.toml` arquivo do seu aplicativo ou no arquivo de configuração designado.

Essa opção é obrigatória.

Exemplos

Exibe as saídas, em formato de tabela, dos recursos em sua CloudFormation pilha denominados. `test-stack`

```
$ sam list stack-outputs --stack-name test-stack --output table
```

OutputKey Description	OutputValue	
HelloWorldFunctionIamRole Implicit IAM Role created for Hello function	arn:aws:iam:: <i>account-number</i> :role/sam- app-test-list-HelloWorldFunctionRole- SRJDMJ6F7F41	World
HelloWorldApi Gateway endpoint URL for Prod for Hello World function	https://uj80uoe2o2.execute-api.us- east-1.amazonaws.com/Prod/hello/	API stage
HelloWorldFunction World Lambda Function ARN	arn:aws:lambda:us- east-1: <i>account-number</i> :function:sam-app- test-list- HelloWorldFunction-H85Y7yIV7ZLq	Hello

sam local callback

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam local callback`.

Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam local callback` comando permite que você envie retornos de chamada para execuções de funções duráveis locais durante o desenvolvimento e o teste.

Usage

```
$ sam local callback <subcommand> <options>
```

Opções

--help, -h

Mostre esta mensagem e saia.

Subcomandos

succeed

Envie um retorno de chamada bem-sucedido para uma execução de função durável. Para obter mais informações, consulte [sam local callback succeed](#).

fail

Envie um retorno de chamada de falha para uma execução de função durável. Para obter mais informações, consulte [sam local callback fail](#).

heartbeat

Envie um retorno de chamada de pulsação para uma execução de função durável. Para obter mais informações, consulte [sam local callback heartbeat](#).

sam local callback succeed

Envie um retorno de chamada bem-sucedido para uma execução de função durável.

Usage

```
$ sam local callback succeed CALLBACK_ID [OPTIONS]
```

Argumentos obrigatórios

CALLBACK_ID

O identificador exclusivo do retorno de chamada para o qual enviar uma resposta bem-sucedida.

Opções

-r, --result TEXT

Carga útil do resultado de sucesso como string.

--region TEXT

Defina a AWS região do serviço (por exemplo, `us-east-1`).

--profile TEXT

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

--config-env TEXT

Nome do ambiente especificando valores de parâmetros padrão no arquivo de configuração.

Padrão: `default`

--config-file TEXT

Arquivo de configuração contendo valores de parâmetros padrão. Padrão: `samconfig.toml`

--save-params

Salve os parâmetros fornecidos pela linha de comando no arquivo de configuração.

--beta-features / --no-beta-features

Ativar/desativar recursos beta.

--debug

Ative o registro de depuração para imprimir mensagens de depuração geradas pela AWS SAM CLI e exibir registros de data e hora.

-h, --help

Mostra esta mensagem e sai.

Exemplos

Envie um retorno de chamada bem-sucedido sem resultado:

```
$ sam local callback succeed my-callback-id
```

Envie um retorno de chamada de sucesso com o resultado:

```
$ sam local callback succeed my-callback-id --result 'Task completed successfully'
```

Envie um retorno de chamada bem-sucedido com a opção curta:


```
$ sam local callback succeed my-callback-id -r 'Success result'
```

sam local callback fail

Envie um retorno de chamada de falha para uma execução de função durável.

Usage

```
$ sam local callback fail CALLBACK_ID [OPTIONS]
```

Argumentos necessários

CALLBACK_ID

O identificador exclusivo do retorno de chamada para o qual enviar uma resposta de falha.

Opções

--error-data TEXT

Dados de erro adicionais.

--stack-trace TEXT

Entradas de rastreamento de pilha. Essa opção pode ser especificada várias vezes.

--error-type TEXT

Tipo de erro.

--error-message TEXT

Mensagem de erro detalhada.

--region TEXT

Defina a AWS região do serviço (por exemplo,us-east-1).

--profile TEXT

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

--config-env TEXT

Nome do ambiente especificando valores de parâmetros padrão no arquivo de configuração.

Padrão: default

--config-file TEXT

Arquivo de configuração contendo valores de parâmetros padrão. Padrão: `samconfig.toml`

--save-params

Salve os parâmetros fornecidos pela linha de comando no arquivo de configuração.

--beta-features / --no-beta-features

Ativar/desativar recursos beta.

--debug

Ative o registro de depuração para imprimir mensagens de depuração geradas pela AWS SAM CLI e exibir registros de data e hora.

-h, --help

Mostra esta mensagem e sai.

Exemplos

Enviar retorno de chamada de falha sem parâmetros:

```
$ sam local callback fail my-callback-id
```

Enviar retorno de chamada de falha com mensagem de erro:

```
$ sam local callback fail my-callback-id --error-message 'Task failed'
```

Envie um retorno de chamada de falha com todos os parâmetros:

```
$ 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

Envie um retorno de chamada de pulsação para uma execução de função durável.

Usage

```
$ sam local callback heartbeat CALLBACK_ID [OPTIONS]
```

Argumentos obrigatórios

CALLBACK_ID

O identificador exclusivo do retorno de chamada para o qual enviar uma pulsação.

Opções

`--region TEXT`

Defina a AWS região do serviço (por exemplo, `us-east-1`).

`--profile TEXT`

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

`--config-env TEXT`

Nome do ambiente especificando valores de parâmetros padrão no arquivo de configuração.

Padrão: `default`

`--config-file TEXT`

Arquivo de configuração contendo valores de parâmetros padrão. Padrão: `samconfig.toml`

`--save-params`

Salve os parâmetros fornecidos pela linha de comando no arquivo de configuração.

`--beta-features / --no-beta-features`

Ativar/desativar recursos beta.

`--debug`

Ative o registro de depuração para imprimir mensagens de depuração geradas pela AWS SAM CLI e exibir registros de data e hora.

`-h, --help`

Mostra esta mensagem e sai.

Exemplos

Enviar retorno de chamada de pulsação:

```
$ sam local callback heartbeat my-callback-id
```

sam local execution

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam local execution`.

Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam local execution` comando permite gerenciar e inspecionar as execuções de funções duráveis locais durante o desenvolvimento e o teste.

Note

Esses comandos podem não exigir acesso às AWS credenciais.

Usage

```
$ sam local execution <subcommand> <options>
```

Opções

--help, -h

Mostre esta mensagem e saia.

Subcomandos

get

Obtenha detalhes de uma execução de função durável. Para obter mais informações, consulte [sam local execution get](#).

history

Obtenha o histórico de execução de uma função durável. Para obter mais informações, consulte [sam local execution history](#).

stop

Interrompa a execução de uma função durável. Para obter mais informações, consulte [sam local execution stop](#).

sam local execution get

Recupere os detalhes da execução de uma função durável específica.

Note

Esse comando pode não exigir acesso às AWS credenciais.

Usage

```
$ sam local execution get DURABLE_EXECUTION_ARN [OPTIONS]
```

Argumentos obrigatórios

DURABLE_EXECUTION_ARN

O Amazon Resource Name (ARN) da execução da função durável para a qual recuperar detalhes.

Opções

--format [summary|json]

Formato de saída. Padrão: summary

--region TEXT

Defina a AWS região do serviço (por exemplo,us-east-1).

--profile TEXT

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

--config-env TEXT

Nome do ambiente especificando valores de parâmetros padrão no arquivo de configuração.
Padrão: default

--config-file TEXT

Arquivo de configuração contendo valores de parâmetros padrão. Padrão: `samconfig.toml`

--save-params

Salve os parâmetros fornecidos pela linha de comando no arquivo de configuração.

--beta-features / --no-beta-features

Ativar/desativar os recursos beta.

--debug

Ative o registro de depuração para imprimir mensagens de depuração geradas pela AWS SAM CLI e exibir registros de data e hora.

-h, --help

Mostra esta mensagem e sai.

Exemplos

Obtenha detalhes da execução com o formato de resumo padrão:

```
$ sam local execution get arn:aws:lambda:us-east-1:123456789012:function:MyFunction:
$LATEST/durable-execution/my-execution-name/my-execution-id
```

Obtenha detalhes da execução no 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

Obtenha o histórico de execução de uma função durável.

Usage

```
$ sam local execution history EXECUTION_ID [OPTIONS]
```

Argumentos obrigatórios

EXECUTION_ID

O identificador exclusivo da execução durável da função para a qual recuperar o histórico.

Opções

`--format [table|json]`

Formato de saída. Padrão: `table`

`--region TEXT`

Defina a AWS região do serviço (por exemplo, `us-east-1`).

`--profile TEXT`

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

`--config-env TEXT`

Nome do ambiente especificando valores de parâmetros padrão no arquivo de configuração.
Padrão: `default`

`--config-file TEXT`

Arquivo de configuração contendo valores de parâmetros padrão. Padrão: `samconfig.toml`

`--save-params`

Salve os parâmetros fornecidos pela linha de comando no arquivo de configuração.

`--beta-features / --no-beta-features`

Ativar/desativar recursos beta.

`--debug`

Ative o registro de depuração para imprimir mensagens de depuração geradas pela AWS SAM CLI e exibir registros de data e hora.

`-h, --help`

Mostra esta mensagem e sai.

Exemplos

Obtenha o histórico de execução em formato de tabela:

```
$ sam local execution history my-execution-id
```

Obtenha o histórico de execução no formato JSON:

```
$ sam local execution history my-execution-id --format json
```

sam local execution stop

Interrompa a execução de uma função durável em execução.

Note

Esse comando pode não exigir acesso às AWS credenciais.

Usage

```
$ sam local execution stop DURABLE_EXECUTION_ARN [OPTIONS]
```

Argumentos obrigatórios

DURABLE_EXECUTION_ARN

O Amazon Resource Name (ARN) da execução da função durável a ser interrompida.

Opções

--error-message TEXT

Mensagem de erro a ser associada à execução interrompida.

--error-type TEXT

Tipo de erro a ser associado à execução interrompida.

--error-data TEXT

Dados de erro a serem associados à execução interrompida.

`--stack-trace TEXT`

Entradas de rastreamento de pilha. Essa opção pode ser especificada várias vezes.

`--region TEXT`

Defina a AWS região do serviço (por exemplo, `us-east-1`).

`--profile TEXT`

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

`--config-env TEXT`

Nome do ambiente especificando valores de parâmetros padrão no arquivo de configuração.

Padrão: `default`

`--config-file TEXT`

Arquivo de configuração contendo valores de parâmetros padrão. Padrão: `samconfig.toml`

`--save-params`

Salve os parâmetros fornecidos pela linha de comando no arquivo de configuração.

`--beta-features / --no-beta-features`

Ativar/desativar recursos beta.

`--debug`

Ative o registro de depuração para imprimir mensagens de depuração geradas pela AWS SAM CLI e exibir registros de data e hora.

`-h, --help`

Mostra esta mensagem e sai.

Exemplos

Pare a execução sem detalhes de erro:

```
$ sam local execution stop arn:aws:lambda:us-east-1:123456789012:function:MyFunction:
$LATEST/durable-execution/my-execution-name/my-execution-id
```

Pare a execução com mensagem de erro e digite:

```
$ 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"
```

Pare a execução com detalhes completos do erro e rastreamento de pilha:

```
$ 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

Esta página fornece informações de referência para o AWS Serverless Application Model `sam local generate-event` subcomando Command Line Interface (AWS SAMCLI).

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre o uso do AWS SAMCLI `sam local generate-event` comando, consulte [Introdução aos testes com generate-event local do sam](#).

O subcomando `sam local generate-event` gera amostras de carga útil de eventos para suporte Serviços da AWS.

Usage

```
$ sam local generate-event <options> <service> <event> <event-options>
```

Opções

`--config-env` TEXT

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` PATH

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é `samconfig.toml` na raiz do diretório do projeto. Para obter

mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--help`

Mostra esta mensagem e sai.

Serviço

Para visualizar uma lista de serviços compatíveis, execute o comando a seguir:

```
$ sam local generate-event
```

Event

Para ver uma lista de eventos compatíveis que podem ser gerados para cada serviço, execute o seguinte:

```
$ sam local generate-event <service>
```

Opções de eventos

Para ver uma lista das opções de eventos compatíveis que você pode modificar, execute o seguinte:

```
$ sam local generate-event <service> <event> --help
```

Exemplos

Para obter exemplos sobre como usar o subcomando `sam local generate-event`, consulte [Gere eventos de exemplo](#).

SAM invocação local

Esta página fornece informações de referência para o AWS Serverless Application Model `sam local invoke` subcomando Command Line Interface (AWS SAMCLI).

- Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter documentação sobre como usar o AWS SAMCLI `sam local invoke` subcomando, consulte [Introdução aos testes com SAM invocação local](#).

O `sam local invoke` subcomando inicia uma invocação única de uma função de evento localmente. AWS Lambda

 Note

Não é recomendável usar os recursos de invocação local do SAM CLI em código não confiável. Para ter um isolamento completo do seu ambiente local, execute o código diretamente no serviço Lambda.

 Note

Para funções duráveis, `sam local invoke` suporta execução em estado com verificação e repetição automáticas. O contêiner permanece em execução durante a execução durável da função para lidar com a persistência e a retomada do estado.

Usage

```
$ sam local invoke <arguments> <options>
```

 Note

Se você tiver mais de uma função definida em seu AWS SAM modelo, forneça a ID lógica da função que você deseja invocar.

Argumentos

Resource ID (ID do recurso)

O ID da função do Lambda a ser invocada.

Esse argumento é opcional. Se seu aplicativo contiver uma única função Lambda, a AWS SAM CLI a invocará. Se seu aplicativo contiver várias funções, forneça o ID da função a ser invocada.

Valores válidos: o ID lógico do recurso ou o ARN do recurso.

Opções

`--add-host` *LIST*

Passa um nome de host para mapeamento de endereço IP para o arquivo host do contêiner do Docker. Esse parâmetro pode ser passado várias vezes.

Example

Exemplo: `--add-host` *example.com:127.0.0.1*

`--beta-features` | `--no-beta-features`

Permita ou negue recursos beta.

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--container-env-vars`

(Opcional) Passe variáveis de ambiente para o contêiner de imagem da função do Lambda ao depurar localmente.

`--container-host` *TEXT*

Host do contêiner Lambda emulado localmente. O valor padrão é localhost. Se você quiser executar AWS SAMCLI em um contêiner do Docker no macOS, você pode especificar `host.docker.internal`. Se quiser executar o contêiner em um host diferente AWS SAMCLI, você pode especificar o endereço IP do host remoto.

`--container-host-interface` *TEXT*

O endereço IP da interface de rede do host à qual as portas do contêiner devem se vincular. O valor padrão é 127.0.0.1. Use 0.0.0.0 para vincular a todas as interfaces.

--container-dns *TEXT*

Especifica um servidor DNS personalizado para o contêiner Docker. Esse parâmetro pode ser passado várias vezes para especificar vários servidores DNS.

Example

Exemplo: **--container-dns** *8.8.8.8*

--debug

Ativa o registro de depuração para imprimir mensagens de depuração que o AWS SAMCLI gera e para exibir timestamps.

--debug-args *TEXT*

Argumentos adicionais a serem transmitidos para o depurador.

--debug-port, -d *TEXT*

Quando especificado, inicia o contêiner da função do Lambda no modo de depuração e expõe essa porta no host local.

--debugger-path *TEXT*

O caminho do host para um depurador montado no contêiner Lambda.

--docker-network *TEXT*

Especifica o nome ou ID de uma rede Docker existente à qual os contêineres do Docker do Lambda devem se conectar, juntamente com a rede de ponte padrão. Se não for especificado, os contêineres do Lambda se conectarão somente à rede de Docker de ponte padrão.

--docker-volume-basedir, -v *TEXT*

A localização do diretório base em que o AWS SAM arquivo existe. Se o Docker estiver sendo executado em uma máquina remota, você deverá montar o caminho em que o AWS SAM arquivo existe na máquina Docker e modificar esse valor para corresponder à máquina remota.

--durable-execution-name *TEXT*

Nome da execução durável (somente para funções duráveis)

--env-vars, -n *PATH*

O JSON ou .env arquivo que contém valores para as variáveis de ambiente da função Lambda. O formato do arquivo é detectado automaticamente. Para obter mais informações sobre variáveis de ambiente, consulte [Arquivo de variável de ambiente](#).

`--event, -e PATH`

O arquivo JSON que contém dados de eventos que são passados para a função do Lambda quando ela é invocada. Se você não especificar essa opção, nenhum evento será assumido. Para inserir JSON `stdin`, você deve passar o valor '-'. Para obter detalhes sobre formatos de mensagens de eventos de diferentes AWS serviços, consulte Como [trabalhar com outros serviços](#) no Guia do AWS Lambda desenvolvedor.

`--force-image-build`

Especifica se o AWS SAMCLI deve reconstruir a imagem usada para invocar função do Lambda com camadas.

`--help`

Mostra esta mensagem e sai.

`--hook-name TEXT`

O nome do hook usado para estender a funcionalidade AWS SAMCLI.

Valores aceitos: `terraform`.

`--invoke-image TEXT`

O URI da imagem do contêiner que você deseja usar para a invocação da função local.

Para funções Lambda com `PackageType: Zip`, isso substitui a imagem base do tempo de execução. Por padrão, o AWS SAM extrai a imagem de contêiner do Amazon ECR Public (que está listada em [Repositórios de imagens para AWS SAM](#)). Use essa opção para extrair a imagem de outro localização.

Para funções Lambda com `PackageType: Image`, isso substitui o modelo criado `ImageUri`. Use isso para invocar uma imagem de aplicativo diferente localmente sem modificar seu modelo.

Especifique essa opção várias vezes. Cada instância dessa opção pode ter uma string ou um par de valores-chave. Se você especificar uma string, ela será a URI da imagem do contêiner a ser usada para todas as funções em seu aplicativo. Por exemplo, `sam local invoke --invoke-image public.ecr.aws/sam/emu-python3.13`. Se você especificar um par de valores-chave, a chave será o nome do recurso e o valor será o URI da imagem do contêiner a ser usada para esse recurso. Por exemplo, `.sam local invoke --invoke-image public.ecr.aws/sam/emu-python3.13 --invoke-image Function1=amazon/`

`aws-sam-cli-emulation-image-python3.13` Com pares de valores-chave, você pode especificar imagens de contêiner diferentes para recursos diferentes.

`--layer-cache-basedir` *DIRECTORY*

Especifica a localização do diretório base onde as camadas que seu modelo usa são baixadas.

`--log-file, -l` *TEXT*

O arquivo de log para o qual enviar os registros de runtime.

`--mount-symlinks`

Garante que a CLI do AWS SAM sempre monte symlinks que estão presentes nos arquivos a serem criados ou invocados. Isso se aplica somente aos symlinks no diretório de nível superior (ou seja, links simbólicos que estão diretamente na raiz da função). Por padrão, os symlinks não são montados, exceto aqueles necessários para usar o `build-in-source` no `node_modules` no NodeJS.

`--no-event`

Invoca a função com um evento vazio.

`--no-memory-limit`

Remove a limitação de memória no contêiner durante a chamada local, mesmo quando a memória está configurada no AWS SAM modelo.

`--parameter-overrides`

Uma string que contém substituições de CloudFormation parâmetros codificadas como pares de valores-chave. Use o mesmo formato do AWS Command Line Interface (AWS CLI). O formato da CLI do AWS SAM é uma palavra-chave explícita de chave e valor e cada substituição é separada por um espaço. Veja dois exemplos a seguir:

- `--parameter-overrides ParameterKey=hello,ParameterValue=world`
- `--parameter-overrides ParameterKey=hello,ParameterValue=world
ParameterKey=example1,ParameterValue=example2
ParameterKey=apple,ParameterValue=banana`

`--profile` *TEXT*

O perfil específico do seu arquivo de credenciais que obtém as AWS credenciais.

`--region` *TEXT*

A AWS região para a qual implantar. Por exemplo, `us-east-1`.

`--runtime` *TEXT*

Usa o runtime especificado para invocar uma função do Lambda localmente. Isso substitui o runtime definido no arquivo `template.yml`. Isso também permite testar as funções do Lambda com diferentes tempos de execução sem modificar a configuração original da função.

`--save-params`

Salve os parâmetros fornecidos na linha de comando no arquivo AWS SAM de configuração.

`--shutdown`

Emula um evento de desligamento após a conclusão da invocação, a fim de testar o tratamento da extensão sobre o comportamento de desligamento.

`--skip-prepare-infra`

Ignora a fase de preparação se nenhuma alteração na infraestrutura tiver sido feita. Execute com a opção `--hook-name`.

`--skip-pull-image`

Por padrão, ele AWS SAMCLI verifica o ambiente de runtime remoto mais recente do Lambda e atualiza sua imagem local automaticamente para se manter sincronizada.

Especifique essa opção para pular a remoção da Docker imagem mais recente para seu ambiente de runtime do Lambda.

`--template, -t` *PATH*

O arquivo AWS SAM de modelo.

Essa opção não é compatível com o `--hook-name`.

 Note

Se você especificar essa opção, AWS SAM carrega somente o modelo e os recursos locais para os quais ele aponta.

`--tenant-id` *TEXT*

O ID do locatário para funções Lambda de vários locatários. Usado para garantir o isolamento computacional entre diferentes inquilinos. Obrigatório ao invocar funções configuradas com o modo de isolamento do inquilino.

--terraform-plan-file

O caminho relativo ou absoluto para seu arquivo de Terraform plano local ao usar o AWS SAMCLI com Terraform Cloud. Essa opção exige que --hook-name seja definida como terraform.

Exemplos

O exemplo a seguir usa um evento gerado para testes locais usando um evento `s3.json` para invocar uma função do Lambda localmente

```
$ sam local invoke --event events/s3.json S3JsonLoggerFunction
```

O exemplo a seguir testa a função `HelloWorldFunction` usando o runtime do Python 3.11

```
$ sam local invoke --runtime python3.11 HelloWorldFunction
```

O exemplo a seguir testa a função `HelloWorldFunction` com nome de execução durável.

```
$ sam local invoke HelloWorldFunction --durable-execution-name my-execution
```

SAM API de inicialização local

Esta página fornece informações de referência para o AWS Serverless Application Model `sam local start-api` subcomando Command Line Interface (AWS SAMCLI).

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre o uso do AWS SAMCLI `sam local start-api` subcomando, consulte [Introdução aos testes com SAM API de inicialização local](#).

O `sam local start-api` subcomando executa suas funções de AWS Lambda evento localmente para testar por meio de um host de servidor HTTP local.

Usage

```
$ sam local start-api <options>
```

Opções

`--add-host` *LIST*

Passa um nome de host para mapeamento de endereço IP para o arquivo host do contêiner do Docker. Esse parâmetro pode ser passado várias vezes.

Example

Exemplo: `--add-host` *example.com:127.0.0.1*

`--beta-features` | `--no-beta-features`

Permita ou negue recursos beta.

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--container-env-vars`

Opcional. Passe variáveis de ambiente para o contêiner de imagem ao depurar localmente.

`--container-host` *TEXT*

Host do contêiner Lambda emulado localmente. O valor padrão é localhost. Se você quiser executar AWS SAMCLI em um contêiner do Docker no macOS, você pode especificar `host.docker.internal`. Se você quiser executar o contêiner em um host diferente AWS SAMCLI, você pode especificar o endereço IP do host remoto.

`--container-host-interface` *TEXT*

O endereço IP da interface de rede do host à qual as portas do contêiner devem se vincular. O valor padrão é 127.0.0.1. Use 0.0.0.0 para vincular a todas as interfaces.

--container-dns *TEXT*

Especifica um servidor DNS personalizado para o contêiner Docker. Esse parâmetro pode ser passado várias vezes para especificar vários servidores DNS.

Example

Exemplo: **--container-dns** *8.8.8.8*

--debug

Ativa o registro de depuração para imprimir a mensagem de depuração gerada pelo AWS SAMCLI e exibir timestamps.

--debug-args *TEXT*

Argumentos adicionais a serem transmitidos para o depurador.

--debug-function

Opcional. Especifica a função do Lambda para aplicar opções de depuração quando for especificado como **--warm-containers**. Esse parâmetro se aplica a **--debug-port**, **--debugger-path** e **--debug-args**.

--debug-port, -d *TEXT*

Quando especificado, inicia o contêiner da função do Lambda no modo de depuração e expõe essa porta no host local.

--debugger-path *TEXT*

O caminho do host para um depurador que será montado no contêiner Lambda.

--docker-network *TEXT*

Especifica o nome ou ID de uma rede Docker existente à qual os contêineres do Docker do Lambda devem se conectar, juntamente com a rede de ponte padrão. Se não for especificado, os contêineres do Lambda se conectarão somente à rede de Docker de ponte padrão.

--docker-volume-basedir, -v *TEXT*

A localização do diretório base em que o AWS SAM arquivo existe. Se o Docker estiver sendo executado em uma máquina remota, você deverá montar o caminho em que o AWS SAM arquivo existe na máquina Docker e modificar esse valor para corresponder à máquina remota.

`--env-vars, -n PATH`

O JSON ou .env arquivo que contém valores para as variáveis de ambiente da função Lambda. O formato do arquivo é detectado automaticamente.

`--force-image-build`

Especifica se AWS SAM CLI deve reconstruir a imagem usada para invocar funções com camadas.

`--help`

Mostra esta mensagem e sai.

`--hook-name TEXT`

O nome do hook usado para estender a funcionalidade AWS SAMCLI.

Valores aceitos: terraform.

`--host TEXT`

O nome do host local ou endereço IP ao qual se vincular (padrão: '127.0.0.1').

`--invoke-image TEXT`

O URI da imagem de contêiner que você deseja usar para suas funções do Lambda.

Para funções Lambda com `PackageType: Zip`, isso substitui a imagem base do tempo de execução. Por padrão, AWS SAM extrai a imagem do contêiner do Amazon ECR Public. Use essa opção para extrair a imagem de outro localização.

Para funções do Lambda com `PackageType: Image`, isso substitui o modelo criado `ImageUri`. Use isso para invocar uma imagem de aplicativo diferente localmente sem modificar seu modelo.

Especifique essa opção várias vezes. Cada instância dessa opção pode ter uma string ou um par de valores-chave. Se você especificar uma string, ela será a URI da imagem do contêiner a ser usada para todas as funções em seu aplicativo. Por exemplo, `sam local start-api --invoke-image public.ecr.aws/sam/emu-python3.13`. Se você especificar um par de valores-chave, a chave será o nome do recurso e o valor será o URI da imagem do contêiner a ser usada para esse recurso. Por exemplo, `.sam local start-api --invoke-image public.ecr.aws/sam/emu-python3.13 --invoke-image Function1=amazon/aws-sam-cli-emulation-image-python3.13` Com pares de valores-chave, você pode especificar imagens de contêiner diferentes para recursos diferentes.

`--layer-cache-basedir` *DIRECTORY*

Especifica a localização com base na qual as camadas que seu modelo usa são baixadas.

`--log-file, -l` *TEXT*

O arquivo de log para o qual enviar os registros de runtime.

`--no-memory-limit`

Remove a limitação de memória no contêiner durante a chamada local, mesmo quando a memória está configurada no AWS SAM modelo.

`--no-watch`

Desativa a observação de arquivos quando usado com `--warm-containers`. Quando especificado, o AWS SAMCLI não monitora alterações no código-fonte ou no modelo, e os contêineres quentes não são reiniciados automaticamente nas modificações do código. Para obter as alterações locais, pare e execute novamente o comando.

Use essa opção quando a observação de arquivos causar alta carga de CPU ou I/O sobrecarga, como em grandes monorepos, ambientes com varredura de software antivírus ou projetos em que várias funções Lambda compartilham uma. `CodeUri`

Essa opção será ignorada se não `--warm-containers` for especificada.

`--parameter-overrides`

Uma string que contém substituições de CloudFormation parâmetros codificadas como pares de valores-chave. Use o mesmo formato do AWS Command Line Interface (AWS CLI). O formato da CLI do AWS SAM é uma palavra-chave explícita de chave e valor e cada substituição é separada por um espaço. Veja dois exemplos a seguir:

- `--parameter-overrides ParameterKey=hello,ParameterValue=world`
- `--parameter-overrides ParameterKey=hello,ParameterValue=world
ParameterKey=example1,ParameterValue=example2
ParameterKey=apple,ParameterValue=banana`

`--port, -p` *INTEGER*

O número da porta local para escutar (padrão: '3000').

`--profile` *TEXT*

O perfil específico do seu arquivo de credenciais que obtém as AWS credenciais.

`--region` *TEXT*

A AWS região para a qual implantar. Por exemplo, us-east-1.

`--save-params`

Salve os parâmetros que você fornece na linha de comando no arquivo AWS SAM de configuração.

`--shutdown`

Emula um evento de desligamento após a conclusão da invocação, a fim de testar o tratamento da extensão sobre o comportamento de desligamento.

`--skip-prepare-infra`

Ignora a fase de preparação se nenhuma alteração na infraestrutura tiver sido feita. Execute com a opção `--hook-name`.

`--skip-pull-image`

Especifica se o comando deve ignorar a extração da imagem mais recente do Docker para o runtime do Lambda.

`--ssl-cert-file` *PATH*

Caminho para o arquivo do certificado SSL (padrão: Nenhum). Ao usar essa opção, a opção `--ssl-key-file` também deve ser usada.

`--ssl-key-file` *PATH*

Caminho para o arquivo de chave SSL (padrão: Nenhum). Ao usar essa opção, a opção `--ssl-cert-file` também deve ser usada.

`--static-dir, -s` *TEXT*

Todos os arquivos de ativos estáticos (por exemplo, CSS/JavaScript/HTML) localizados nesse diretório são apresentados em/.

`--template, -t` *PATH*

O arquivo AWS SAM de modelo.

 Note

Se você especificar essa opção, AWS SAM carrega somente o modelo e os recursos locais para os quais ele aponta.

--terraform-plan-file

O caminho relativo ou absoluto para seu arquivo de Terraform plano local ao usar o AWS SAMCLI com Terraform Cloud. Essa opção exige que --hook-name seja definida como terraform.

--warm-containers *[EAGER | LAZY]*

Opcional. Especifica como AWS SAMCLI gerencia os contêineres para cada função.

Existem duas opções:

EAGER: os contêineres de todas as funções são carregados na inicialização e persistem entre as invocações.

LAZY: os contêineres são carregados somente quando cada função é invocada pela primeira vez. Esses contêineres persistem para invocações adicionais.

Exemplo

O exemplo a seguir inicia um servidor local, permitindo que você teste a aplicação por meio da API. Para que esse comando funcione, a aplicação deve estar instalada e o Docker deve estar em execução.

```
$ sam local start-api --port 3000
```

start-lambda do local do sam

Esta página fornece informações de referência para o AWS Serverless Application Model `sam local start-lambda` subcomando Command Line Interface (AWS SAMCLI).

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre o uso do AWS SAMCLI `sam local start-lambda` subcomando, consulte [Introdução aos testes com start-lambda do local do sam](#).

O subcomando `sam local start-lambda` inicia um endpoint local para emular AWS Lambda.

Usage

```
$ sam local start-lambda <options>
```


Opções

`--add-host` *LIST*

Passa um nome de host para mapeamento de endereço IP para o arquivo host do contêiner do Docker. Esse parâmetro pode ser passado várias vezes.

Example

Exemplo: `--add-host` *example.com:127.0.0.1*

`--beta-features` | `--no-beta-features`

Permita ou negue recursos beta.

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--container-env-vars`

Opcional. Passe variáveis de ambiente para o contêiner de imagem ao depurar localmente.

`--container-host` *TEXT*

Host do contêiner Lambda emulado localmente. O valor padrão é localhost. Se você quiser executar AWS SAMCLI em um contêiner do Docker no macOS, você pode especificar `host.docker.internal`. Se você quiser executar o contêiner em um host diferente AWS SAMCLI, você pode especificar o endereço IP do host remoto.

`--container-host-interface` *TEXT*

O endereço IP da interface de rede do host à qual as portas do contêiner devem se vincular. O valor padrão é 127.0.0.1. Use 0.0.0.0 para vincular a todas as interfaces.

`--container-dns` *TEXT*

Especifica um servidor DNS personalizado para o contêiner Docker. Esse parâmetro pode ser passado várias vezes para especificar vários servidores DNS.

Example

Exemplo: `--container-dns 8.8.8.8`

`--debug`

Ativa o registro de depuração para imprimir a mensagem de depuração gerada pelo AWS SAMCLI e exibir timestamps.

`--debug-args` *TEXT*

Argumentos adicionais a serem transmitidos para o depurador.

`--debug-function`

Opcional. Especifica a função de evento Lambda para aplicar opções de depuração quando for especificado. `--warm-containers` Esse parâmetro se aplica a `--debug-port`, `--debugger-path` e `--debug-args`.

`--debug-port`, `-d` *TEXT*

Quando especificado, inicia o contêiner da função do Lambda no modo de depuração e expõe essa porta no host local.

`--debugger-path` *TEXT*

O caminho do host para um depurador a ser montado no contêiner Lambda.

`--docker-network` *TEXT*

Especifica o nome ou ID de uma rede Docker existente à qual os contêineres do Docker do Lambda devem se conectar, juntamente com a rede de ponte padrão. Se não for especificado, os contêineres do Lambda se conectarão somente à rede de Docker de ponte padrão.

`--docker-volume-basedir`, `-v` *TEXT*

A localização do diretório base em que o AWS SAM arquivo existe. Se o Docker estiver sendo executado em uma máquina remota, você deverá montar o caminho em que o AWS SAM arquivo existe na máquina Docker e modificar esse valor para corresponder à máquina remota.

`--env-vars, -n PATH`

O JSON ou .env arquivo que contém valores para as variáveis de ambiente da função Lambda. O formato do arquivo é detectado automaticamente.

`--force-image-build`

Especifique se CLI deve reconstruir a imagem usada para invocar funções com camadas.

`--help`

Mostra esta mensagem e sai.

`--hook-name TEXT`

O nome do hook usado para estender a funcionalidade AWS SAMCLI.

Valores aceitos: terraform.

`--host TEXT`

O nome do host local ou endereço IP ao qual se vincular (padrão: '127.0.0.1').

`--invoke-image TEXT`

O URI da imagem do contêiner que você deseja usar para a invocação da função local.

Para funções Lambda com `PackageType: Zip`, isso substitui a imagem base do tempo de execução. Por padrão, AWS SAM extrai a imagem de contêiner do Amazon ECR Public. Use essa opção para extrair a imagem de outro localização.

Para funções do Lambda com `PackageType: Image`, isso substitui o modelo criado `ImageUri`. Use isso para invocar uma imagem de aplicativo diferente localmente sem modificar seu modelo.

Especifique essa opção várias vezes. Cada instância dessa opção pode ter uma string ou um par de valores-chave. Se você especificar uma string, ela será a URI da imagem do contêiner a ser usada para todas as funções em seu aplicativo. Por exemplo, `sam local start-lambda --invoke-image public.ecr.aws/sam/emu-python3.13`. Se você especificar um par de valores-chave, a chave será o nome do recurso e o valor será o URI da imagem do contêiner a ser usada para esse recurso. Por exemplo, `.sam local start-lambda --invoke-image public.ecr.aws/sam/emu-python3.13 --invoke-image Function1=amazon/aws-sam-cli-emulation-image-python3.13` Com pares de valores-chave, você pode especificar imagens de contêiner diferentes para recursos diferentes.

`--layer-cache-basedir` *DIRECTORY*

Especifica a localização com base na qual as camadas que seu modelo usa são baixadas.

`--log-file, -l` *TEXT*

O arquivo de log para o qual enviar os registros de runtime.

`--no-memory-limit`

Remove a limitação de memória no contêiner durante a chamada local, mesmo quando a memória está configurada no AWS SAM modelo.

`--no-watch`

Desativa a observação de arquivos quando usado com `--warm-containers`. Quando especificado, o AWS SAMCLI não monitora alterações no código-fonte ou no modelo, e os contêineres quentes não são reiniciados automaticamente nas modificações do código. Para obter as alterações locais, pare e execute novamente o comando.

Use essa opção quando a observação de arquivos causar alta carga de CPU ou I/O sobrecarga, como em grandes monorepos, ambientes com varredura de software antivírus ou projetos em que várias funções Lambda compartilham uma. `CodeUri`

Essa opção será ignorada se não `--warm-containers` for especificada.

`--parameter-overrides`

Uma string que contém substituições de CloudFormation parâmetros codificadas como pares de valores-chave. Use o mesmo formato do AWS Command Line Interface (AWS CLI). O formato da CLI do AWS SAM é uma palavra-chave explícita de chave e valor e cada substituição é separada por um espaço. Veja dois exemplos a seguir:

- `--parameter-overrides ParameterKey=hello,ParameterValue=world`
- `--parameter-overrides ParameterKey=hello,ParameterValue=world
ParameterKey=example1,ParameterValue=example2
ParameterKey=apple,ParameterValue=banana`

`--port, -p` *INTEGER*

O número da porta local para escutar (padrão: '3001').

`--profile` *TEXT*

O perfil específico do seu arquivo de credenciais que obtém as AWS credenciais.

`--region` *TEXT*

A AWS região para a qual implantar. Por exemplo, us-east-1.

`--save-params`

Salve os parâmetros fornecidos na linha de comando no arquivo AWS SAM de configuração.

`--shutdown`

Emula um evento de desligamento após a conclusão da invocação, a fim de testar o tratamento da extensão sobre o comportamento de desligamento.

`--skip-prepare-infra`

Ignora a fase de preparação se nenhuma alteração na infraestrutura tiver sido feita. Execute com a opção `--hook-name`.

`--skip-pull-image`

Especifica se o comando CLI deve ignorar a extração da imagem do Docker mais recente para o runtime do Lambda.

`--template, -t` *PATH*

O arquivo AWS SAM de modelo.

 Note

Se você especificar essa opção, AWS SAM carrega somente o modelo e os recursos locais para os quais ele aponta. Essa opção não é compatível com `--hook-name`.

`--terraform-plan-file`

O caminho relativo ou absoluto para seu arquivo de Terraform plano local ao usar o AWS SAMCLI com Terraform Cloud. Essa opção exige que `--hook-name` seja definida como `terraform`.

`--warm-containers` *[EAGER | LAZY]*

Opcional. Especifica como AWS SAMCLI gerencia os contêineres para cada função.

Existem duas opções:

- **EAGER**: os contêineres de todas as funções são carregados na inicialização e persistem entre as invocações.

- LAZY: os contêineres são carregados somente quando cada função é invocada pela primeira vez. Esses contêineres persistem para invocações adicionais.

Exemplo

Para obter um exemplo detalhado e uma explicação aprofundada sobre o uso do subcomando `sam local start-lambda`, consulte [Introdução aos testes com start-lambda do local do sam](#).

logs de spam

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam logs`.

Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam logs` comando busca registros que são gerados por suas Funções de AWS Lambda Eventos.

Usage

```
$ sam logs <options>
```

Opções

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--cw-log-group` *LIST*

Inclui registros dos grupos de CloudWatch registros de registros que você especifica. Se você especificar essa opção junto com `name`, AWS SAM incluirá registros dos grupos de registros especificados, além dos registros dos recursos nomeados.

--debug

Ativa o registro de depuração para imprimir a mensagem de depuração gerada pelo AWS SAMCLI e exibir timestamps.

---end-time, e *TEXT*

Busca registros até agora. A hora pode ser valores relativos, como '5 minutos atrás', 'amanhã' ou um carimbo de data/hora formatado como '2018-01-01 10:10:10'.

--filter *TEXT*

Permite especificar uma expressão para encontrar rapidamente registros que correspondam a termos, frases ou valores nos seus eventos de log. Isso pode ser uma palavra-chave simples (por exemplo, “erro”) ou um padrão compatível com o Amazon CloudWatch Logs. Para a sintaxe, consulte a [documentação do Amazon CloudWatch Logs](#).

--help

Mostra esta mensagem e sai.

--include-traces

Inclui X-Ray traços na saída do log.

--name, -n *TEXT*

O nome do recurso para o qual buscar registros. Se esse recurso fizer parte de uma CloudFormation pilha, esse pode ser o ID lógico do recurso da função no AWS SAM modelo CloudFormation/. Vários nomes podem ser fornecidos repetindo o parâmetro novamente. Se o recurso estiver em uma pilha aninhada, o nome poderá ser precedido pelo nome da pilha aninhada para extrair registros desse recurso (). NestedStackLogicalId/ResourceLogicalId Se o nome do recurso não for fornecido, a pilha fornecida será verificada e as informações de registro serão extraídas de todos os recursos compatíveis. Se você não especificar essa opção, AWS SAM buscará os registros de todos os recursos na pilha especificada. Os seguintes tipos de registro são compatíveis:

- 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*

Especifica o formato de saída para logs. Para imprimir registros formatados, especifique `text`. Para imprimir os registros como JSON, especifique `json`.

`--profile` *TEXT*

O perfil específico do seu arquivo de credenciais que obtém as AWS credenciais.

`--region` *TEXT*

A AWS região para a qual implantar. Por exemplo, `us-east-1`.

`--save-params`

Salve os parâmetros fornecidos na linha de comando no arquivo AWS SAM de configuração.

`--stack-name` *TEXT*

O nome da CloudFormation pilha da qual o recurso faz parte.

`--start-time, -s` *TEXT*

Busca registros a partir desse momento. A hora pode ser valores relativos, como '5 minutos atrás', 'ontem' ou um carimbo de data/hora formatado como '2018-01-01 10:10:10'. O padrão é '10 minutos atrás'.

`--tail, -t`

Encerra a saída do log. Isso ignora o argumento de horário de término e continua a buscar os registros à medida que eles se tornam disponíveis.

Exemplos

Quando suas funções fazem parte de uma CloudFormation pilha, você pode buscar registros usando o ID lógico da função ao especificar o nome da pilha.

```
$ sam logs -n HelloWorldFunction --stack-name myStack
```


Visualize os registros de um intervalo de tempo específico usando as opções `-s` (`--start-time`) e `-e` (`--end-time`).

```
$ sam logs -n HelloWorldFunction --stack-name myStack -s '10min ago' -e '2min ago'
```

Você também pode adicionar a opção `--tail` de aguardar novos registros e vê-los à medida que chegam.

```
$ sam logs -n HelloWorldFunction --stack-name myStack --tail
```

Use a opção `--filter` para encontrar rapidamente os logs que correspondem a determinados termos, frases ou valores nos eventos de log.

```
$ sam logs -n HelloWorldFunction --stack-name myStack --filter "error"
```

Visualize os registros de um recurso em uma pilha secundária.

```
$ sam logs --stack-name myStack -n childStack/HelloWorldFunction
```

Registros finais de todos os recursos compatíveis em seu aplicativo.

```
$ sam logs --stack-name sam-app --tail
```

Busque registros de uma função do Lambda específica e da API Gateway API em seu aplicativo.

```
$ sam logs --stack-name sam-app --name HelloWorldFunction --name HelloWorldRestApi
```

Busque registros de todos os recursos compatíveis em seu aplicativo e, além disso, dos grupos de registros especificados.

```
$ sam logs --cw-log-group /aws/lambda/myfunction-123 --cw-log-group /aws/lambda/myfunction-456
```

sam package

A interface de linha de AWS Serverless Application Model comando (AWS SAM CLI) empacota um AWS SAM aplicativo.

Esse comando cria um `.zip` arquivo com seu código e dependências e carrega o arquivo no Amazon Simple Storage Service (Amazon S3). AWS SAM ativa a criptografia para todos os arquivos armazenados no Amazon S3. Em seguida, ele retorna uma cópia do seu AWS SAM modelo, substituindo as referências aos artefatos locais pela localização do Amazon S3 em que o comando carregou os artefatos.

Por padrão, quando você usa esse comando, AWS SAMCLI pressupõe que seu diretório de trabalho atual seja o diretório raiz do seu projeto. O AWS SAMCLI primeiro tenta localizar um arquivo de modelo criado usando o [sam build](#) comando, localizado na `.aws-sam` subpasta e nomeado `template.yaml`. Em seguida, o AWS SAMCLI tenta localizar um arquivo de modelo chamado `template.yaml` ou `template.yml` no diretório de trabalho atual. Se você especificar a `--template` opção, o comportamento padrão AWS SAMCLI do modelo será substituído e empacotará apenas esse AWS SAM modelo e os recursos locais para os quais ele aponta.

Note

[sam deploy](#) agora executa implicitamente a funcionalidade do `sam package`. Você pode usar o [sam deploy](#) comando diretamente para empacotar e implantar sua aplicação.

Usage

```
$ sam package <arguments> <options>
```

Argumentos

Resource ID (ID do recurso)

O ID da função de evento Lambda para empacotar.

Esse argumento é opcional. Se seu aplicativo contiver uma única função Lambda, a AWS SAM CLI a empacotará. Se seu aplicativo contiver várias funções, forneça o ID da função para empacotar uma única função.

Valores válidos: o ID lógico do recurso ou o ARN do recurso.

Opções

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--debug`

Ativa o registro de depuração para imprimir a mensagem de depuração gerada pelo AWS SAMCLI e exibir timestamps.

`--force-upload`

Substituir arquivos existentes no bucket do Amazon S3. Especifique essa bandeira para fazer upload de artefatos, mesmo que eles correspondam aos artefatos existentes no bucket do Amazon S3.

`--help`

Mostra esta mensagem e sai.

`--image-repository` *TEXT*

O URI do repositório do Amazon Elastic Container Registry (Amazon ECR) onde esse comando faz upload da imagem da função. Necessário para funções declaradas com o tipo de pacote Image.

`--kms-key-id` *TEXT*

O ID de uma chave AWS Key Management Service (AWS KMS) usada para criptografar artefatos que estão em repouso no bucket do Amazon S3. Se essa opção não for especificada, AWS SAM usará as chaves de S3-managed criptografia da Amazon.

`--metadata`

(Opcional) Um mapa de metadados para anexar a todos os artefatos referenciados em seu modelo.

--no-progressbar

Não exiba uma barra de progresso ao fazer o upload de artefatos para o Amazon S3.

--output-template-file *PATH*

O caminho para o arquivo em que o comando grava o modelo empacotado. Se você não especificar um caminho, o comando gravará o modelo na saída padrão.

--profile *TEXT*

O perfil específico do seu arquivo de credenciais que obtém as AWS credenciais.

--region *TEXT*

A AWS região para a qual implantar. Por exemplo, us-east-1.

--resolve-s3

Crie automaticamente um bucket do Amazon S3 para usar na embalagem. Se você especificar as opções **--s3-bucket** e **--resolve-s3**, ocorrerá um erro.

--s3-bucket *TEXT*

O nome do bucket do Amazon S3 em que esse comando faz upload do artefato. Se seu artefato for maior que 51.200 bytes, a opção **--s3-bucket** ou a **--resolve-s3** será obrigatória. Se você especificar as opções **--s3-bucket** e **--resolve-s3**, ocorrerá um erro.

--s3-prefix *TEXT*

Prefixo adicionado ao nome do artefato que é carregado para o bucket do Amazon S3. O nome do prefixo é um nome de caminho (nome da pasta) para o bucket do Amazon S3. Isso só se aplica às funções declaradas com o tipo de Zip pacote.

--save-params

Salve os parâmetros fornecidos na linha de comando no arquivo AWS SAM de configuração.

--signing-profiles *LIST*

(Opcional) A lista de perfis de assinatura com os quais assinar seus pacotes de implantação. Esse parâmetro usa uma lista de pares de valores-chave, em que a chave é o nome da função ou camada a ser assinada e o valor é o perfil de assinatura, com um proprietário de perfil opcional delimitado por **:**. Por exemplo, **.FunctionNameToSign=SigningProfileName1 LayerNameToSign=SigningProfileName2:SigningProfileOwner**

`--template-file`, `--template`, `-t` *PATH*

O caminho e o nome do arquivo em que seu AWS SAM modelo está localizado.

 Note

Se você especificar essa opção, AWS SAM empacotará somente o modelo e os recursos locais para os quais ele aponta.

`--use-json`

JSON de saída para o CloudFormation modelo. YAML é usado por padrão.

Exemplo

O exemplo a seguir cria e empacota artefatos para uma função CodeDeploy e aplicativos Lambda. Os artefatos são carregados em um bucket do Amazon S3. A saída do comando é um novo arquivo chamado `package.yml`.

```
$ sam package \  
  --template-file template.yml \  
  --output-template-file package.yml \  
  --s3-bucket amzn-s3-demo-bucket
```

bootstrap do mesmo pipeline

Esta página fornece informações de referência para o AWS Serverless Application Model `sam local pipeline bootstrap` subcomando Command Line Interface (AWS SAMCLI).

Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam pipeline bootstrap` subcomando gera os recursos de AWS infraestrutura necessários para se conectar ao seu CI/CD sistema. Essa etapa deve ser executada em cada estágio de implantação em seu pipeline antes de executar o comando `sam pipeline init`.

Esse subcomando configura os seguintes recursos de AWS infraestrutura:

- Opção de configurar as permissões do pipeline por meio de:

- Um usuário do pipeline IAM com ID da chave de acesso e credenciais de acesso à chave secreta a serem compartilhadas com o CI/CD sistema.

Note

Recomendamos alternar as chaves de acesso regularmente. Para obter mais informações, consulte [Altere as chaves de acesso regularmente para casos de uso que exijam credenciais de longo prazo](#) no Guia do usuário do IAM.

- CI/CD Plataformas suportadas por meio do OIDC. Para uma introdução sobre o uso do OIDC com pipeline AWS SAM , vá para [Como usar a autenticação OIDC com pipelines AWS SAM](#).
- Uma função CloudFormation de execução do IAM assumida pela CloudFormation para implantar o AWS SAM aplicativo.
- Um bucket Amazon S3 para armazenar os AWS SAM artefatos.
- Opcionalmente, um repositório de imagens do Amazon ECR para armazenar pacotes de implantação Lambda de imagens de contêiner (se você tiver um recurso do tipo de pacote Image).

Usage

```
$ sam pipeline bootstrap <options>
```

Opções

`--bitbucket-repo-uuid` *TEXT*

O UUID do repositório do Bitbucket. Essa opção é específica ao uso do Bitbucket OIDC para obter permissões.

Note

Esse valor pode ser encontrado em <https://bitbucket.org/workspace/repositoryadmin/addon/admin/pipelines/openid-connect>

`--bucket` *TEXT*

O ARN do bucket Amazon S3 que contém os artefatos. AWS SAM

--cicd-provider *TEXT*

A CI/CD plataforma para o AWS SAM gasoduto.

--cloudformation-execution-role *TEXT*

O ARN da função do IAM a ser assumida CloudFormation durante a implantação da pilha do aplicativo. Forneça somente se quiser usar sua própria função. Caso contrário, o comando criará uma nova função.

--config-env *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é **default**. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

--config-file *PATH*

O caminho e o nome do arquivo de configuração contendo os valores de parâmetros padrão a serem usados. O valor padrão é `samconfig.toml` na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

--confirm-changeset | --no-confirm-changeset

Solicite a confirmação da implantação de seus recursos.

--create-image-repository | --no-create-image-repository

Especifique se deseja criar um repositório de imagens do Amazon ECR se nenhum for fornecido. O repositório Amazon ECR contém as imagens de contêiner das Lambda Event Functions, ou camadas com um tipo de pacote de. Image O padrão é `--no-create-image-repository`.

--debug

Ativa o registro de depuração e imprime as mensagens de depuração que o AWS SAMCLI gera e para exibir timestamps.

--deployment-branch *TEXT*

Nome da filial a partir da qual as implantações ocorrerão. Essa opção é específica para usar o GitHub Actions OIDC para obter permissões.

`--github-org` *TEXT*

A GitHub organização à qual o repositório pertence. Se nenhuma organização existir, insira o nome de usuário do proprietário do repositório. Essa opção é específica para usar o GitHub Actions OIDC para obter permissões.

`--github-repo` *TEXT*

Nome do GitHub repositório a partir do qual as implantações ocorrerão. Essa opção é específica para usar o GitHub Actions OIDC para obter permissões.

`--gitlab-group` *TEXT*

O GitLab grupo ao qual o repositório pertence. Essa opção é específica para usar o GitLab OIDC para obter permissões.

`--gitlab-project` *TEXT*

O nome GitLab do projeto. Essa opção é específica para usar o GitLab OIDC para obter permissões.

`--help`, `-h`

Mostra esta mensagem e sai.

`--image-repository` *TEXT*

O ARN de um repositório de imagens do Amazon ECR que contém as imagens de contêiner das funções do Lambda, ou camadas que têm um tipo de pacote de Image. Se fornecidas, as opções `--create-image-repository` serão ignoradas. Se não for fornecido e `--create-image-repository` for especificado, o comando cria um.

`--interactive` | `--no-interactive`

Desative a solicitação interativa para parâmetros de bootstrap e falhe se algum parâmetro necessário estiver ausente. O valor padrão é `--interactive`. Para esse comando, `--stage` é o único parâmetro obrigatório.

 Note

Se `--no-interactive` for especificado junto com `--use-oidc-provider`, todos os parâmetros necessários para seu provedor de OIDC devem ser incluídos.

`--oidc-client-id` *TEXT*

O ID do cliente configurado para uso com seu provedor OIDC.

`--oidc-provider` *[github-actions | gitlab | bitbucket-pipelines]*

Nome do CI/CD provedor que será usado para permissões do OIDC. GitLab, GitHub, e o Bitbucket são compatíveis.

`--oidc-provider-url` *TEXT*

O URL do provedor de OIDC. O valor deve começar com **https://**.

`--permissions-provider` *[oidc | iam]*

Escolha um provedor de permissões para assumir a função de execução do pipeline. O valor padrão é **iam**.

`--pipeline-execution-role` *TEXT*

O ARN do perfil do IAM a ser assumida pelo usuário do pipeline para operar nesse estágio. Forneça somente se quiser usar sua própria função. Se não for fornecido, esse comando criará uma nova função.

`--pipeline-user` *TEXT*

O Amazon Resource Name (ARN) do usuário do IAM com o ID da chave de acesso e a chave de acesso secreta compartilhados com o CI/CD sistema. Ele é usado para conceder permissão a esse usuário do IAM para acessar a AWS conta correspondente. Se não for fornecido, o comando criará um usuário do IAM junto com o ID da chave de acesso e as credenciais da chave de acesso secreta.

`--profile` *TEXT*

O perfil específico do seu arquivo de credenciais que obtém as AWS credenciais.

`--region` *TEXT*

A AWS região para a qual implantar. Por exemplo, `.us-east-1`

`--save-params`

Salve os parâmetros que você fornece na linha de comando no arquivo AWS SAM de configuração.

`--stage` **TEXT**

O nome do estágio de implantação correspondente. Ele é usado como um sufixo para os recursos de AWS infraestrutura criados.

Solução de problemas

Erro: Falta o parâmetro necessário

Quando `--no-interactive` for especificado junto com `--use-oidc-provider` e nenhum dos parâmetros necessários for fornecido, essa mensagem de erro será exibida junto com uma descrição dos parâmetros ausentes.

Exemplo

O exemplo a seguir cria os AWS recursos necessários para criar seu CI/CD sistema e ativa o registro de depuração e imprime mensagens de depuração geradas pelo AWS SAMCLI: usa um evento gerado para testes locais usando um evento para invocar uma `s3.json` função Lambda localmente

```
$ sam pipeline bootstrap --debug
```

sam pipeline init

Esta página fornece informações de referência para o AWS Serverless Application Model `sam pipeline init` subcomando Command Line Interface (AWS SAMCLI).

Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam pipeline init` subcomando gera um arquivo de configuração de pipeline que seu CI/CD sistema pode usar para implantar aplicativos sem servidor usando AWS SAM

Antes de usar `sam pipeline init`, você deve criar os recursos necessários para cada estágio do seu pipeline. Você pode fazer isso executando `sam pipeline init --bootstrap` para ser guiado pelo processo de geração do arquivo de instalação e configuração ou consultando os recursos que você criou anteriormente com o comando `sam pipeline bootstrap`.

Usage

```
$ sam pipeline init <options>
```

Opções

--bootstrap

Ative o modo interativo que orienta o usuário na criação dos recursos de AWS infraestrutura necessários.

--config-env *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é `default`. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

--config-file *TEXT*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é `samconfig.toml` no diretório raiz do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

--debug

Ativa o registro de depuração para imprimir a mensagem de depuração que o AWS SAMCLI gera e para exibir timestamps.

--help, -h

Mostra esta mensagem e sai.

--save-params

Salve os parâmetros fornecidos na linha de comando no arquivo AWS SAM de configuração.

Exemplo

O exemplo a seguir mostra como usar a `--bootstrap` opção para permitir que você percorra um modo interativo que orienta você na criação dos recursos de AWS infraestrutura necessários:

```
$ sam pipeline init --bootstrap
```

sam publish

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam publish`.

Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam publish` comando publica um AWS SAM aplicativo no AWS Serverless Application Repository. Esse comando usa um AWS SAM modelo empacotado e publica o aplicativo na região especificada AWS .

O `sam publish` comando espera que o AWS SAM modelo inclua uma Metadata seção que contenha os metadados do aplicativo necessários para publicação. Na seção Metadata, as propriedades `LicenseUrl` e `ReadmeUrl` devem se referir aos buckets do Amazon Simple Storage Service (Amazon S3), não aos arquivos locais. Para obter mais informações sobre a Metadata seção do AWS SAM modelo, consulte [Como publicar a aplicação com a do AWS SAM CLI](#).

Por padrão, `sam publish` cria o aplicativo como privado. Antes que outras AWS contas possam visualizar e implantar seu aplicativo, você deve compartilhá-lo. Para obter informações sobre o compartilhamento de aplicativos, consulte [Exemplos de políticas AWS Serverless Application Repository baseadas em recursos](#) no Guia do AWS Serverless Application Repository desenvolvedor.

Note

Atualmente, `sam publish` não oferece suporte à publicação de aplicativos aninhados especificados localmente. Se seu aplicativo contiver aplicativos aninhados, você deverá publicá-los separadamente no AWS Serverless Application Repository antes de publicar seu aplicativo principal.

Usage

```
$ sam publish <options>
```

Opções

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--debug`

Ativa o registro de depuração para imprimir a mensagem de depuração que o AWS SAMCLI gera e para exibir timestamps.

`--help`

Mostra esta mensagem e sai.

`--profile` *TEXT*

O perfil específico do seu arquivo de credenciais que obtém as AWS credenciais.

`--region` *TEXT*

A AWS região para a qual implantar. Por exemplo, us-east-1.

`--save-params`

Salve os parâmetros fornecidos na linha de comando no arquivo AWS SAM de configuração.

`--semantic-version` *TEXT*

(Opcional) Use essa opção para fornecer uma versão semântica do seu aplicativo que substitua a propriedade SemanticVersion na seção Metadata do arquivo de modelo. Para obter mais informações sobre controle de versionamento semântico, consulte a especificação de controle de versão [semântica](#).

`--template, -t` *PATH*

O caminho do arquivo AWS SAM de modelo[default: template.[yaml|yml]].

Exemplos

Para publicar um aplicativo:

```
$ sam publish --template packaged.yaml --region us-east-1
```

sam remote callback

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam remote callback`.

Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam remote callback` comando permite que você envie retornos de chamada para execuções remotas de funções duráveis em execução no AWS Lambda.

Note

Esses comandos exigem acesso às AWS credenciais.

Usage

```
$ sam remote callback <subcommand> <options>
```

Opções

`--help`, `-h`

Mostre esta mensagem e saia.

Subcomandos

succeed

Envie um retorno de chamada bem-sucedido para uma execução remota de função durável. Para obter mais informações, consulte [sam remote callback succeed](#).

fail

Envie uma falha de retorno de chamada para uma execução remota de função durável. Para obter mais informações, consulte [sam remote callback fail](#).

heartbeat

Envie uma pulsação de retorno de chamada para uma execução remota de função durável. Para obter mais informações, consulte [sam remote callback heartbeat](#).

sam remote callback succeed

Envie um retorno de chamada bem-sucedido para uma execução remota de função durável.

Note

Esse comando requer acesso às AWS credenciais.

Usage

```
$ sam remote callback succeed CALLBACK_ID [OPTIONS]
```

Argumentos obrigatórios

CALLBACK_ID

O identificador exclusivo do retorno de chamada para o qual enviar uma resposta bem-sucedida.

Opções

`-r, --result TEXT`

Carga útil do resultado de sucesso como string.

`--region TEXT`

Defina a AWS região do serviço (por exemplo,us-east-1).

`--profile TEXT`

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

`--config-env TEXT`

Nome do ambiente especificando valores de parâmetros padrão no arquivo de configuração.

Padrão: default

`--config-file TEXT`

Arquivo de configuração contendo valores de parâmetros padrão. Padrão: samconfig.toml

`--save-params`

Salve os parâmetros fornecidos pela linha de comando no arquivo de configuração.

`--beta-features / --no-beta-features`

Ativar/desativar os recursos beta.

`--debug`

Ative o registro de depuração para imprimir mensagens de depuração geradas pela AWS SAM CLI e exibir registros de data e hora.

`-h, --help`

Mostra esta mensagem e sai.

Exemplos

Envie um retorno de chamada bem-sucedido sem resultado:

```
$ sam remote callback succeed my-callback-id
```

Envie um retorno de chamada de sucesso com o resultado:

```
$ sam remote callback succeed my-callback-id --result 'Task completed successfully'
```

sam remote callback fail

Envie uma falha de retorno de chamada para uma execução remota de função durável.

Note

Esse comando requer acesso às AWS credenciais.

Usage

```
$ sam remote callback fail CALLBACK_ID [OPTIONS]
```

Argumentos obrigatórios

CALLBACK_ID

O identificador exclusivo do retorno de chamada para o qual enviar uma resposta de falha.

Opções

`--error-data TEXT`

Dados de erro adicionais.

`--stack-trace TEXT`

Entradas de rastreamento de pilha. Essa opção pode ser especificada várias vezes.

`--error-type TEXT`

Tipo de erro.

`--error-message TEXT`

Mensagem de erro detalhada.

`--region TEXT`

Defina a AWS região do serviço (por exemplo, `us-east-1`).

`--profile TEXT`

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

`--config-env TEXT`

Nome do ambiente especificando valores de parâmetros padrão no arquivo de configuração.

Padrão: `default`

`--config-file TEXT`

Arquivo de configuração contendo valores de parâmetros padrão. Padrão: `samconfig.toml`

`--save-params`

Salve os parâmetros fornecidos pela linha de comando no arquivo de configuração.

`--beta-features / --no-beta-features`

Ativar/desativar recursos beta.

`--debug`

Ative o registro de depuração para imprimir mensagens de depuração geradas pela AWS SAM CLI e exibir registros de data e hora.

-h, --help

Mostra esta mensagem e sai.

Exemplos

Envie um retorno de chamada de falha sem parâmetros:

```
$ sam remote callback fail my-callback-id
```

Enviar retorno de chamada de falha com mensagem de erro:

```
$ sam remote callback fail my-callback-id --error-message 'Task failed'
```

Envie um retorno de chamada de falha com todos os parâmetros:

```
$ 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

Envie uma pulsação de retorno de chamada para uma execução remota de função durável.

Note

Esse comando requer acesso às AWS credenciais.

Usage

```
$ sam remote callback heartbeat CALLBACK_ID [OPTIONS]
```

Argumentos obrigatórios

CALLBACK_ID

O identificador exclusivo do retorno de chamada para o qual enviar uma pulsação.

Opções

`--region TEXT`

Defina a AWS região do serviço (por exemplo, `us-east-1`).

`--profile TEXT`

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

`--config-env TEXT`

Nome do ambiente especificando valores de parâmetros padrão no arquivo de configuração.

Padrão: `default`

`--config-file TEXT`

Arquivo de configuração contendo valores de parâmetros padrão. Padrão: `samconfig.toml`

`--save-params`

Salve os parâmetros fornecidos pela linha de comando no arquivo de configuração.

`--beta-features / --no-beta-features`

Ativar/desativar recursos beta.

`--debug`

Ative o registro de depuração para imprimir mensagens de depuração geradas pela AWS SAM CLI e exibir registros de data e hora.

`-h, --help`

Mostra esta mensagem e sai.

Exemplos

Enviar retorno de chamada de pulsação:

```
$ sam remote callback heartbeat my-callback-id
```

sam remote execution

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam remote execution`.

Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam remote execution` comando permite gerenciar e inspecionar execuções remotas de funções duráveis em execução no AWS Lambda.

 Note

Esses comandos exigem acesso às AWS credenciais.

Usage

```
$ sam remote execution <subcommand> <options>
```

Opções

`--help`, `-h`

Mostre esta mensagem e saia.

Subcomandos

get

Obtenha detalhes de uma execução duradoura. Para obter mais informações, consulte [sam remote execution get](#).

history

Obtenha o histórico de execução de uma função durável. Para obter mais informações, consulte [sam remote execution history](#).

stop

Interrompa a execução de uma função durável. Para obter mais informações, consulte [sam remote execution stop](#).

sam remote execution get

Recupere os detalhes da execução de uma função durável específica do AWS Lambda.

O ARN de execução deve estar no formato retornado pelo Invoke ou encontrado nos registros de execução.

Note

Esse comando requer acesso às AWS credenciais.

Usage

```
$ sam remote execution get DURABLE_EXECUTION_ARN [OPTIONS]
```

Argumentos obrigatórios

DURABLE_EXECUTION_ARN

O Amazon Resource Name (ARN) da execução da função durável para a qual recuperar detalhes.

Opções

--format [summary|json]

Formato de saída. Padrão: summary

--region TEXT

Defina a AWS região do serviço (por exemplo,us-east-1).

--profile TEXT

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

--config-env TEXT

Nome do ambiente especificando valores de parâmetros padrão no arquivo de configuração.
Padrão: default

--config-file TEXT

Arquivo de configuração contendo valores de parâmetros padrão. Padrão: samconfig.toml

--save-params

Salve os parâmetros fornecidos pela linha de comando no arquivo de configuração.

--beta-features / --no-beta-features

Ativar/desativar recursos beta.

--debug

Ative o registro de depuração para imprimir mensagens de depuração geradas pela AWS SAM CLI e exibir registros de data e hora.

-h, --help

Mostra esta mensagem e sai.

Exemplos

Obtenha detalhes da execução com o formato de resumo padrão:

```
$ 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'
```

Obtenha detalhes da execução no 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

Você deve garantir que os caracteres de controle no ARN de execução \$ sejam escapados corretamente ao usar comandos shell.

sam remote execution history

Recupere o histórico de execução de uma função durável específica do AWS Lambda.

O ARN de execução deve estar no formato retornado StartDurableExecution ou encontrado nos registros de execução.

Note

Esse comando requer acesso às AWS credenciais.

Usage

```
$ sam remote execution history DURABLE_EXECUTION_ARN [OPTIONS]
```

Argumentos obrigatórios

DURABLE_EXECUTION_ARN

O Amazon Resource Name (ARN) da execução da função durável para a qual recuperar o histórico.

Opções

--format [table|json]

Formato de saída. Padrão: table

--region TEXT

Defina a AWS região do serviço (por exemplo,us-east-1).

--profile TEXT

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

--config-env TEXT

Nome do ambiente especificando valores de parâmetros padrão no arquivo de configuração.
Padrão: default

--config-file TEXT

Arquivo de configuração contendo valores de parâmetros padrão. Padrão: samconfig.toml

--save-params

Salve os parâmetros fornecidos pela linha de comando no arquivo de configuração.

--beta-features / --no-beta-features

Ativar/desativar recursos beta.

--debug

Ative o registro de depuração para imprimir mensagens de depuração geradas pela AWS SAM CLI e exibir registros de data e hora.

-h, --help

Mostra esta mensagem e sai.

Exemplos

Obtenha o histórico de execução com o formato de tabela padrão:

```
$ sam remote execution history arn:aws:lambda:us-east-1:123456789012:function:MyFunction:$LATEST/durable-execution/my-execution-name/my-execution-id
```

Obtenha o histórico de execução no 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

Interrompa a execução de uma função durável em execução no AWS Lambda.

O ARN de execução deve estar no formato retornado StartDurableExecution ou encontrado nos registros de execução.

 Note

Esse comando requer acesso às AWS credenciais.

Usage

```
$ sam remote execution stop DURABLE_EXECUTION_ARN [OPTIONS]
```

Argumentos obrigatórios

DURABLE_EXECUTION_ARN

O Amazon Resource Name (ARN) da execução da função durável a ser interrompida.

Opções

--error-message TEXT

Mensagem de erro a ser associada à execução interrompida.

--error-type TEXT

Tipo de erro a ser associado à execução interrompida.

--error-data TEXT

Dados de erro a serem associados à execução interrompida.

--stack-trace TEXT

Entradas de rastreamento de pilha. Essa opção pode ser especificada várias vezes.

--region TEXT

Defina a AWS região do serviço (por exemplo, `us-east-1`).

--profile TEXT

Selecione um perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

--config-env TEXT

Nome do ambiente especificando valores de parâmetros padrão no arquivo de configuração.

Padrão: `default`

--config-file TEXT

Arquivo de configuração contendo valores de parâmetros padrão. Padrão: `samconfig.toml`

--save-params

Salve os parâmetros fornecidos pela linha de comando no arquivo de configuração.

--beta-features / --no-beta-features

Ativar/desativar os recursos beta.

--debug

Ative o registro de depuração para imprimir mensagens de depuração geradas pela AWS SAM CLI e exibir registros de data e hora.

-h, --help

Mostra esta mensagem e sai.

Exemplos

Pare a execução sem detalhes de erro:

```
$ sam remote execution stop arn:aws:lambda:us-east-1:123456789012:function:MyFunction:
$LATEST/durable-execution/my-execution-name/my-execution-id
```

Pare a execução com mensagem de erro e digite:

```
$ 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"
```

Pare a execução com detalhes completos do erro e rastreamento de pilha:

```
$ 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()"
```

invocação remota do sam

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam remote invoke`.

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre como usar o AWS SAMCLI `sam remote invoke` comando, consulte [Introdução aos testes na nuvem com invocação remota do sam](#).

O `sam remote invoke` comando invoca recursos suportados no . Nuvem AWS

Usage

```
$ sam remote invoke <arguments> <options>
```

Argumentos

Resource ID (ID do recurso)

O ID do recurso suportado a ser invocado.

Esse argumento aceita os seguintes valores:

- O nome de recurso da Amazon (ARN) - O ARN do conjunto de recursos.

Tip

Use `sam list stack-outputs --stack-name <stack-name>` para obter o ARN dos seus recursos.

- ID lógico - O ID lógico do recurso. Você também deve fornecer o nome da AWS CloudFormation pilha usando a `--stack-name` opção.
- ID físico — O ID físico do recurso. Esse ID é criado quando você implanta um recurso usando CloudFormation o.

Tip

Use `sam list resources --stack-name <stack-name>` para obter a identificação física dos seus recursos.

Quando você fornece um ARN ou ID física:

Se você fornecer um ARN ou ID física, não forneça um nome de pilha. Quando o nome da pilha for fornecido usando a `--stack-name` opção, ou quando o nome da pilha for definido

em seu arquivo de configuração, ele AWS SAM CLI processará automaticamente sua ID de recurso como um valor de ID lógico da CloudFormation pilha.

Quando você não fornece um ID de recurso:

Se você não fornecer um ID de recurso, mas fornecer um nome de pilha com a `--stack-name` opção, a AWS SAM CLI tentará invocar automaticamente um recurso em CloudFormation sua pilha usando a seguinte lógica:

1. Eles AWS SAM CLI identificarão os tipos de recursos na seguinte ordem e passarão para a próxima etapa quando o tipo de recurso for encontrado em sua pilha:
 - a. Lambda
 - b. Step Functions
 - c. Amazon SQS
 - d. Kinesis Data Streams
2. Se o tipo de recurso tiver um único recurso em sua pilha, AWS SAM CLI ele o invocará. Se existirem vários recursos do tipo de recurso em sua pilha, eles AWS SAM CLI retornarão um erro.

A seguir estão exemplos do que eles AWS SAM CLI farão:

- Pilha que contém duas Lambda Event Functions e uma fila do Amazon SQS — Ela AWS SAM CLI localizará o tipo de recurso Lambda e retornará um erro, pois a pilha contém mais de uma função Lambda.
- Pilha que contém uma função Lambda e dois aplicativos Amazon Kinesis Data Streams AWS SAM CLI — Eles localizarão a função Lambda e a invocarão, pois a pilha contém um único recurso Lambda.
- Pilha que contém uma única fila do Amazon SQS e dois aplicativos do Kinesis Data Streams — Eles AWS SAM CLI localizarão a fila do Amazon SQS e a invocarão, pois a pilha contém uma única fila do Amazon SQS.

Opções

`--beta-features` | `--no-beta-features`

Permita ou negue recursos beta.

`--config-env` **TEXT**

Especifique o ambiente a ser usado em seu arquivo AWS SAMCLI de configuração.

Padrão: default

--config-file *FILENAME*

Especifique o caminho e o nome do arquivo de configuração.

Para obter mais informações sobre esses arquivos de configuração, consulte [Configurando o AWS SAM CLI](#).

Padrão: samconfig.toml na raiz do diretório do seu projeto.

--debug

Ative o registro da depuração. Isso imprime mensagens de depuração e registros de data e hora gerados pelo AWS SAM CLI.

--event, -e *TEXT*

O evento a ser enviado ao recurso de destino.

--event-file *FILENAME*

O caminho para um arquivo que contém o evento a ser enviado ao recurso de destino.

--help, -h

Mostra a mensagem de ajuda e sai.

--output [*text* | *json*]

Exiba os resultados da sua invocação em um formato de saída específico.

json— Os metadados da solicitação e a resposta do recurso são retornados na estrutura JSON. A resposta contém a saída completa do SDK.

text— Os metadados da solicitação são retornados na estrutura de texto. A resposta do recurso é retornada no formato de saída do recurso invocado.

--parameter

[Boto3](#) Parâmetros adicionais que você pode passar para o recurso que está sendo chamado.

Amazon Kinesis Data Streams

Os seguintes parâmetros adicionais podem ser usados para colocar um registro no fluxo de dados do Kinesis:

- ExplicitHashKey='string'
- PartitionKey='string'
- SequenceNumberForOrdering='string'
- StreamARN='string'

Para obter uma descrição de cada parâmetro, consulte [Kinesis.Client.put_record](#).

AWS Lambda

Os seguintes parâmetros adicionais podem ser usados para invocar um recurso Lambda e receber uma resposta em buffer:

- ClientContext='base64-encoded string'
- InvocationType='[DryRun | Event | RequestResponse]'
- LogType='[None | Tail]'
- Qualifier='string'

Os seguintes parâmetros adicionais podem ser usados para invocar um recurso Lambda com streaming de resposta:

- ClientContext='base64-encoded string'
- InvocationType='[DryRun | RequestResponse]'
- LogType='[None | Tail]'
- Qualifier='string'

Para obter uma descrição de cada parâmetro, consulte o seguinte:

- Lambda com resposta em buffer — [Lambda.Client.invoke](#)
- [Lambda com streaming de resposta — _with_response_stream](#) [Lambda.Client.invoke](#)

Amazon Simple Queue Service (Amazon SQS)

Os seguintes parâmetros adicionais podem ser usados para enviar uma mensagem para uma fila do Amazon SQS:

- DelaySeconds=*integer*
- MessageAttributes='json string'
- MessageDeduplicationId='string'
- MessageGroupId='string'

- `MessageSystemAttributes='json string'`

Para obter uma descrição de cada parâmetro, consulte [SQS.Client.send_message](#).

AWS Step Functions

Os seguintes parâmetros adicionais podem ser usados para iniciar uma execução de máquina de estado:

- `name='string'`
- `traceHeader='string'`

Para obter uma descrição de cada parâmetro, consulte [SFN.Client.start_execution](#).

`--profile TEXT`

O perfil específico do seu arquivo de credenciais para obter as AWS credenciais.

`--region TEXT`

O Região da AWS do recurso. Por exemplo, `.us-east-1`

`--stack-name TEXT`

O nome da CloudFormation pilha à qual o recurso pertence.

`--tenant-id TEXT`

O ID do locatário para funções Lambda de vários locatários. Usado para garantir o isolamento computacional entre diferentes inquilinos. Obrigatório ao invocar funções configuradas com o modo de isolamento do inquilino.

`--test-event-name NAME`

O nome do evento de teste compartilhável a ser passado para sua função do Lambda.

Note

Essa opção oferece suporte apenas às funções Lambda.

Exemplo

O exemplo a seguir invoca recursos compatíveis na AWS nuvem e ativa o registro de depuração, que imprime mensagens de depuração e registros de data e hora gerados pelo: AWS SAMCLI

```
$ sam remote invoke--debug
```

evento de teste remoto do sam

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam remote test-event`.

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre o uso do AWS SAMCLI `sam remote test-event` comando, consulte [Introdução aos testes em nuvem com evento de teste remoto do sam](#).

O `sam remote test-event` comando interage com eventos de teste compartilháveis no registro do EventBridge esquema da Amazon.

Usage

```
$ sam remote test-event <options> <subcommand>
```

Opções

`--help`, `-h`

Mostre a mensagem de ajuda e saia.

Subcomandos

delete

Exclua um evento de teste compartilhável do registro do EventBridge esquema. Para obter mais informações, consulte [exclusão remota de eventos de teste do sam](#).

get

Obtenha um evento de teste compartilhável do registro do EventBridge esquema. Para obter mais informações, consulte [sam remote test-event get](#).

list

Listar eventos de teste compartilháveis existentes para uma função de AWS Lambda evento. Para obter mais informações, consulte [lista de eventos de teste remotos sam](#).

put

Salve um evento de um arquivo local no registro do EventBridge esquema. Para obter mais informações, consulte [mesma saída remota de eventos de teste](#).

exclusão remota de eventos de teste do sam

Esta página fornece informações de referência para o AWS Serverless Application Model `sam remote test-event delete` subcomando Command Line Interface (AWS SAMCLI).

- Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre o uso do AWS SAMCLI `sam remote test-event` comando, consulte [Introdução aos testes em nuvem com evento de teste remoto do sam](#).

O `sam remote test-event delete` subcomando exclui um evento de teste compartilhável do registro do esquema da Amazon EventBridge .

Usage

```
$ sam remote test-event delete <arguments> <options>
```

Argumentos

Resource ID (ID do recurso)

O ID da função de AWS Lambda evento associada ao evento de teste compartilhável.

Se você fornecer uma ID lógica, também deverá fornecer um valor para a AWS CloudFormation pilha associada à função Lambda usando `--stack-name` a opção.

Valores válidos: o ID lógico do recurso ou o recurso ARN.

Opções

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--help, -h`

Mostre a mensagem de ajuda e saia.

`--name` *TEXT*

O nome do evento de teste compartilhável a ser excluído.

`--stack-name` *TEXT*

O nome da CloudFormation pilha associada à função Lambda.

Essa opção é necessária se você estiver fornecendo a ID lógica da função do Lambda como argumento.

sam remote test-event get

Esta página fornece informações de referência para o AWS Serverless Application Model `sam remote test-event get` subcomando Command Line Interface (AWS SAMCLI).

- Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre o uso do AWS SAMCLI `sam remote test-event` comando, consulte [Introdução aos testes em nuvem com evento de teste remoto do sam](#).

O `sam remote test-event get` subcomando obtém um evento de teste compartilhável do registro de EventBridge esquemas da Amazon.

Usage

```
$ sam remote test-event get <arguments> <options>
```

Argumentos

Resource ID (ID do recurso)

O ID da função de AWS Lambda evento associada ao evento de teste compartilhável a ser obtido.

Se você fornecer uma ID lógica, também deverá fornecer um valor para a AWS CloudFormation pilha associada à função Lambda usando `--stack-name` a opção.

Valores válidos: o ID lógico do recurso ou o recurso ARN.

Opções

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--help`, `-h`

Mostre a mensagem de ajuda e saia.

`--name` *TEXT*

O nome do evento de teste compartilhável a ser obtido.

`--output-file` *FILENAME*

O caminho e o nome do arquivo para salvar o evento em sua máquina local.

Se você não fornecer essa opção, AWS SAM CLI ele exibirá o conteúdo do evento de teste compartilhável para seu console.

`--stack-name` *TEXT*

O nome da CloudFormation pilha associada à função Lambda.

Essa opção é necessária se você estiver fornecendo a ID lógica da função do Lambda como argumento.

lista de eventos de teste remotos sam

Esta página fornece informações de referência para o AWS Serverless Application Model `sam remote test-event list` subcomando Command Line Interface (AWS SAMCLI).

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre o uso do AWS SAMCLI `sam remote test-event` comando, consulte [Introdução aos testes em nuvem com evento de teste remoto do sam](#).

O `sam remote test-event list` subcomando lista os eventos de teste compartilháveis existentes para uma função de AWS Lambda evento específica do registro de EventBridge esquemas da Amazon.

Usage

```
$ sam remote test-event list <arguments> <options>
```

Argumentos

Resource ID (ID do recurso)

O ID da função do Lambda associado aos eventos de teste compartilháveis.

Se você fornecer uma ID lógica, também deverá fornecer um valor para a AWS CloudFormation pilha associada à função Lambda usando `--stack-name` a opção.

Valores válidos: o ID lógico do recurso ou o recurso ARN.

Opções

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--help, -h`

Mostre a mensagem de ajuda e saia.

`--stack-name` *TEXT*

O nome da CloudFormation pilha associada à função Lambda.

Essa opção é necessária se você estiver fornecendo a ID lógica da função do Lambda como argumento.

Exemplos

Para obter exemplos sobre como usar esse comando, consulte [Listando eventos de teste compartilháveis](#).

mesma saída remota de eventos de teste

Esta página fornece informações de referência para o AWS Serverless Application Model `sam remote test-event put` subcomando Command Line Interface (AWS SAMCLI).

- Para obter uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre o uso do AWS SAMCLI `sam remote test-event` comando, consulte [Introdução aos testes em nuvem com evento de teste remoto do sam](#).

O `sam remote test-event put` subcomando salva um evento de teste compartilhável da sua máquina local no registro do EventBridge esquema da Amazon.

Usage

```
$ sam remote test-event put <arguments> <options>
```

Argumentos

Resource ID (ID do recurso)

O ID da função de AWS Lambda evento associada ao evento de teste compartilhável.

Se você fornecer uma ID lógica, também deverá fornecer um valor para a AWS CloudFormation pilha associada à função Lambda usando `--stack-name` a opção.

Valores válidos: o ID lógico do recurso ou o recurso ARN.

Opções

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--file` *FILENAME*

O caminho do arquivo e o nome do evento em sua máquina local.

Forneça - como o valor do nome do arquivo a ser lido de stdin.

Essa opção é obrigatória.

`--force`, `-f`

Substitua um evento de teste compartilhável com o mesmo nome.

`--help`, `-h`

Mostre a mensagem de ajuda e saia.

`--name` *TEXT*

O nome para salvar o evento de teste compartilhável como.

Se existir um evento de teste compartilhável com o mesmo nome no registro do EventBridge esquema, AWS SAM CLI ele não o substituirá. Para sobrescrever, adicione a opção `--force`.

`--output-file` *FILENAME*

O caminho e o nome do arquivo para salvar o evento em sua máquina local.

Se você não fornecer essa opção, ele AWS SAM CLI exibirá o conteúdo do evento de teste compartilhável para o seu console.

`--stack-name` *TEXT*

O nome da CloudFormation pilha associada à função Lambda.

Essa opção é necessária se você estiver fornecendo a ID lógica da função do Lambda como argumento.

Exemplo

Para obter um exemplo sobre como usar esse comando, consulte [Salvando eventos de teste compartilháveis](#).

mesma sincronização

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam sync`.

- Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)
- Para obter a documentação sobre o uso do AWS SAMCLI, consulte [AWS SAM CLI](#).

O comando `sam sync` sincroniza as alterações do aplicativo local com o Nuvem AWS.

Usage

```
$ sam sync <options>
```

Opções

`--base-dir`, `-s` *DIRECTORY*

Resolve caminhos relativos para o código-fonte da função ou da camada em relação a esse diretório. Use essa opção para alterar a forma como os caminhos relativos para as pastas de

código-fonte são resolvidos. Por padrão, os caminhos relativos são resolvidos com relação à localização do AWS SAM modelo.

Além dos recursos no aplicativo raiz ou na pilha que você está criando, essa opção também se aplica a aplicativos ou pilhas aninhados. Além disso, essa opção se aplica aos seguintes tipos e propriedades de recursos:

- Tipo de recursos: `AWS::Serverless::Function` Propriedade: `CodeUri`
- Tipo de recurso: `AWS::Serverless::Function` Atributo do recurso: `Metadata` Entrada: `DockerContext`
- Tipo de recursos: `AWS::Serverless::LayerVersion` Propriedade: `ContentUri`
- Tipo de recursos: `AWS::Lambda::Function` Propriedade: `Code`
- Tipo de recursos: `AWS::Lambda::LayerVersion` Propriedade: `Content`

`--build-image` **TEXT**

O URI da [imagem do contêiner](#) que você deseja usar ao criar o aplicativo. Por padrão, AWS SAM usa o URI do repositório de imagens de contêiner do [Amazon Elastic Container Registry \(Amazon ECR\) Public](#). Especifique essa opção para usar uma imagem diferente.

Você pode usar essa opção várias vezes em um único comando. Cada opção aceita uma sequência ou um par de chave/valor.

- String - Especifique o URI da imagem do contêiner que todos os recursos do seu aplicativo usarão. Este é um exemplo:

```
$ sam sync --build-image amazon/aws-sam-cli-build-image-python3.8
```

- Key-value pair — Especifique o nome do recurso como chave e o URI da imagem do contêiner a ser usado com esse recurso como valor. Use esse formato para especificar um URI de imagem de contêiner diferente para cada recurso em seu aplicativo. Veja um exemplo a seguir:

```
$ sam sync --build-image Function1=amazon/aws-sam-cli-build-image-python3.8
```

Essa opção só se aplica se a opção `--use-container` for especificada, caso contrário, ocorrerá um erro.

`--build-in-source` | `--no-build-in-source`

Providencie `--build-in-source` para compilar seu projeto diretamente na pasta de origem.

A opção `--build-in-source` oferece suporte aos seguintes runtimes e métodos de compilação:

- Runtimes — Qualquer runtime Node.js suportado pela opção [`sam init --runtime`](#).
- Métodos de compilação — Makefile, esbuild.

A opção `--build-in-source` não é compatível com as seguintes opções:

- `--use-container`

Padrão: `--no-build-in-source`

`--capabilities` *LIST*

Uma lista de recursos que você especifica para permitir CloudFormation a criação de determinadas pilhas. Alguns modelos de pilha podem incluir recursos que podem afetar as permissões em sua Conta da AWS. Por exemplo, criando novos usuários AWS Identity and Access Management (IAM). Especifique essa opção para substituir os valores padrão. Os valores válidos incluem:

- `CAPACIDADE_IAM`
- `CAPACIDADE_NAMED_IAM`
- `POLÍTICA DE RECURSOS DE CAPACIDADE`
- `CAPABILITY_AUTO_EXPAND`

Padrão: `CAPABILITY_NAMED_IAM` e `CAPABILITY_AUTO_EXPAND`

`--code`

Por padrão, AWS SAM sincroniza todos os recursos em seu aplicativo. Especifique essa opção para sincronizar somente recursos de código, que incluem o seguinte:

- `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`

Para sincronizar recursos de código, AWS SAM usa APIs AWS de serviço diretamente, em vez de implantar por meio de. AWS CloudFormation Para atualizar sua CloudFormation pilha, execute `sam sync --watch` ou `sam deploy`.

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--dependency-layer` | `--no-dependency-layer`

Especifica se as dependências de funções individuais devem ser separadas em outra camada para acelerar o processo de sincronização.

Padrão: `--dependency-layer`

`--express` | `--no-express`

Especifica se o modo CloudFormation Express deve ser usado para implantações de infraestrutura. Quando ativas, as operações de pilha são concluídas quando a configuração do recurso é aplicada. Os recursos continuam sendo preparados em segundo plano.

O modo expresso se aplica somente ao caminho de sincronização da infraestrutura. Code-only as sincronizações (atualizações e implantações da Lambda Event Function) são CloudFormation ignoradas e não são afetadas por essa opção.

O valor padrão é `--no-express`.

Próxima alteração padrão

O padrão muda para `--express` 1º de setembro de 2026.

--image-repository *TEXT*

O nome do repositório do Amazon Elastic Container Registry (Amazon ECR) no qual esse comando faz upload da imagem da função. Necessário para funções declaradas com o tipo de pacote Image.

--image-repositories *TEXT*

Um mapeamento de funções para o URI do repositório Amazon ECR. Funções de referência por meio de sua ID lógica. Veja um exemplo a seguir:

```
$ sam sync --image-repositories Function1=123456789012.dkr.ecr.us-east-1.amazonaws.com/my-repo
```

Você pode especificar esta opção várias vezes em um único comando.

--kms-key-id *TEXT*

O ID de uma chave AWS Key Management Service (AWS KMS) usada para criptografar artefatos que estão em repouso no bucket do Amazon S3. Se você não especificar essa opção, AWS SAM use as chaves de S3-managed criptografia da Amazon.

--metadata

Um mapa de metadados para anexar a todos os artefatos que você faz referência no seu modelo.

--notification-arns *LIST*

Uma lista dos ARNs de tópicos do Amazon Simple Notification Service (Amazon SNS CloudFormation) associados à pilha.

--no-use-container

Uma opção que permite usar o kit de ferramentas do IDE para definir o comportamento padrão.

--parameter-overrides

Uma string que contém substituições de CloudFormation parâmetros codificadas como pares de valores-chave. Use o mesmo formato do AWS Command Line Interface (AWS CLI). O formato da CLI do AWS SAM é uma palavra-chave explícita de chave e valor e cada substituição é separada por um espaço. Veja dois exemplos a seguir:

- **--parameter-overrides** `ParameterKey=hello,ParameterValue=world`

- `--parameter-overrides ParameterKey=hello,ParameterValue=world
ParameterKey=example1,ParameterValue=example2
ParameterKey=apple,ParameterValue=banana`

`--resource` *TEXT*

Especifica o tipo de recurso a ser sincronizado. Para sincronizar vários recursos, você pode especificar essa opção várias vezes. Essa opção é compatível com a opção `--code`. O valor deve ser um dos recursos listados em `--code`. Por exemplo, `--resource AWS::Serverless::Function --resource AWS::Serverless::LayerVersion`

`--resource-id` *TEXT*

Especifica o ID do recurso a ser sincronizado. Para sincronizar vários recursos, você pode especificar essa opção várias vezes. Essa opção é compatível com a opção `--code`. Por exemplo, `--resource-id Function1 --resource-id Function2`

`--role-arn` *TEXT*

O Amazon Resource Name (ARN) de uma função do IAM que é CloudFormation assumida ao aplicar o conjunto de alterações.

`--s3-bucket` *TEXT*

O nome do bucket do Amazon Simple Storage Service (Amazon S3) no qual esse comando carrega seu modelo. CloudFormation Se seu modelo for maior que 51.200 bytes, a opção `--s3-bucket` ou a opção `--resolve-s3` serão obrigatórias. Se você especificar as opções `--s3-bucket` e `--resolve-s3`, ocorrerá um erro.

`--s3-prefix` *TEXT*

O prefixo adicionado aos nomes dos artefatos que você faz upload para o bucket do Amazon S3. O nome do prefixo é um nome de caminho (nome da pasta) para o bucket do Amazon S3. Isso se aplica somente às funções declaradas com o tipo de pacote Zip.

`--save-params`

Salva os parâmetros que você fornece na linha de comando no arquivo AWS SAM de configuração.

`--skip-deploy-sync` | `--no-skip-deploy-sync`

Especifica `--skip-deploy-sync` para ignorar a sincronização inicial da infraestrutura se ela não for necessária. Ele AWS SAMCLI comparará seu AWS SAM modelo local com o

CloudFormation modelo implantado e executará uma implantação somente se uma alteração for detectada.

Especifica `--no-skip-deploy-sync` a realização de uma CloudFormation implantação sempre que `sam sync` for executada.

Para saber mais, consulte [Ignore a inicial CloudFormation implantação](#).

Padrão: `--skip-deploy-sync`

`--stack-name` *TEXT*

O nome da CloudFormation pilha do seu aplicativo.

Essa opção é obrigatória.

`--tags` *LIST*

Uma lista de tags a serem associadas à pilha criada ou atualizada. CloudFormation também propaga essas tags para recursos na pilha que as suportam.

`--template-file`, `--template`, `-t` *PATH*

O caminho e o nome do arquivo em que seu AWS SAM modelo está localizado.

 Note

Se você especificar essa opção, AWS SAM implantará somente o modelo e os recursos locais para os quais ele aponta.

`--use-container`, `-u`

Se suas funções dependerem de pacotes que tenham dependências compiladas de forma nativa, use essa opção para criar sua função dentro de um contêiner AWS Lambda semelhante Docker a.

 Note

Atualmente, essa opção não é compatível com o `--dependency-layer`. Se você usa `--use-container` com `--dependency-layer`, AWS SAMCLI informa e continua com `--no-dependency-layer`.

--watch

Inicia um processo que monitora o aplicativo local em busca de alterações e as sincroniza automaticamente com o. Nuvem AWS Por padrão, quando você especifica essa opção, AWS SAM sincroniza todos os recursos em seu aplicativo à medida que você os atualiza. Com essa opção, AWS SAM executa uma CloudFormation implantação inicial. Em seguida, AWS SAM usa APIs AWS de serviço para atualizar os recursos de código. AWS SAM usa CloudFormation para atualizar recursos de infraestrutura quando você atualiza seu AWS SAM modelo.

--watch-exclude *TEXT*

Exclui um arquivo ou pasta da observação de alterações no arquivo. Para usar essa opção, --watch deverá ser fornecido.

Cada opção recebe um par de valores-chave:

- Chave — O ID lógico de uma função do Lambda em seu aplicativo.
- Valor — O nome do arquivo ou pasta associado a ser excluído.

Quando você atualiza quaisquer arquivos ou pastas especificados com a --watch-exclude opção, AWS SAM CLI eles não iniciarão uma sincronização. No entanto, quando uma atualização de outros arquivos ou pastas inicia uma sincronização, esses arquivos ou pastas serão incluídos nessa sincronização.

Você pode fornecer essa opção várias vezes em um único comando.

Exemplos

Para obter exemplos sobre como usar esse comando, consulte [Opções para o comando sam sync](#).

sam traces

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam traces`.

Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam traces` comando busca AWS X-Ray traços em seu Conta da AWS no Região da AWS.

Usage

```
$ sam traces <options>
```

Opções

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--end-time` *TEXT*

Busca rastros até agora. A hora pode ser valores relativos, como '5 minutos atrás', 'amanhã' ou um carimbo de data/hora formatado como '2018-01-01 10:10:10'.

`--output` *TEXT*

Especifica o formato de saída para logs. Para imprimir registros formatados, especifique `text`. Para imprimir os registros como JSON, especifique `json`.

`--save-params`

Salve os parâmetros fornecidos na linha de comando no arquivo AWS SAM de configuração.

`--start-time` *TEXT*

Busca traços a partir desse momento. A hora pode ser valores relativos, como '5 minutos atrás', 'ontem' ou um carimbo de data/hora formatado como '2018-01-01 10:10:10'. O padrão é '10 minutos atrás'.

`--tail`

Encerra a saída do rastreamento. Isso ignora o argumento de horário de término e continua exibindo rastreamentos à medida que eles se tornam disponíveis.

`--trace-id` *TEXT*

O identificador exclusivo para rastreamento com X-Ray.

Exemplos

Execute o comando a seguir para buscar traços do X-Ray por ID.

```
$ sam traces --trace-id tracing-id-1 --trace-id tracing-id-2
```

Execute o comando a seguir para rastrear traços do X-Ray assim que estiverem disponíveis.

```
$ sam traces --tail
```

sam validate

Esta página fornece informações de referência para o AWS Serverless Application Model comando Command Line Interface (AWS SAMCLI) `sam validate`.

Para uma introdução ao AWS SAMCLI, consulte [O que é o AWS SAM CLI?](#)

O `sam validate` comando verifica se um arquivo AWS SAM de modelo é válido.

Usage

```
$ sam validate <options>
```

Opções

`--config-env` *TEXT*

O nome do ambiente que especifica os valores de parâmetros padrão no arquivo de configuração a serem usados. O valor padrão é “padrão”. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--config-file` *PATH*

O caminho e o nome do arquivo de configuração contendo valores de parâmetros padrão a serem usados. O valor padrão é “samconfig.toml” na raiz do diretório do projeto. Para obter mais informações sobre esses arquivos de configuração, consulte [Arquivo de configuração do AWS SAM CLI](#).

`--debug`

Ativa o registro de depuração para imprimir a mensagem de depuração gerada pelo AWS SAMCLI e exibir timestamps.

--lint

Execute a validação de linting no modelo por meio de cfn-lint. Crie um arquivo de configuração `cfnlintrc` para especificar parâmetros adicionais. Para obter mais informações, consulte [cfn-lint no repositório CloudFormation GitHub](#)

--profile *TEXT*

O perfil específico do seu arquivo de credenciais que obtém as AWS credenciais.

--region *TEXT*

A AWS região para a qual implantar. Por exemplo, `us-east-1`.

--save-params

Salve os parâmetros fornecidos na linha de comando no arquivo AWS SAM de configuração.

--template-file, --template, -t *PATH*

O arquivo AWS SAM de modelo. O valor padrão é `template.[yaml|yml]`.

Se o modelo estiver no diretório de trabalho atual e tiver um nome `template.[yaml|yml|json]`, essa opção não será necessária.

Se você acabou de executar `sam build`, essa opção não é necessária.

Exemplo

Para obter um exemplo sobre como usar esse comando para validar um modelo, consulte [Validar arquivos AWS SAM de modelo](#).

Para obter em exemplo sobre como usar esse comando com cfn-lint, consulte [Valide seus AWS SAM aplicativos com CloudFormation o Linter](#).

gerenciamento da CLI do AWS SAM

Esta seção contém informações sobre como você pode gerenciar e personalizar sua versão do AWS SAMCLI. Estão incluídas informações sobre como você pode configurar os valores do parâmetro de comando da CLI do AWS SAM usando um arquivo de configuração no nível do projeto. Também inclui informações sobre como gerenciar diferentes versões do seu AWS SAMCLI, definir AWS credenciais para que você AWS SAM possa fazer chamadas para AWS serviços em seu nome e

diferentes maneiras de personalizar AWS SAM. Esta seção termina com uma seção sobre AWS SAM solução geral de problemas.

Tópicos

- [Arquivo de configuração do AWS SAM CLI](#)
- [Gerenciar versões do AWS SAM CLI](#)
- [Configurando AWS credenciais](#)
- [Telemetria no AWS SAMCLI](#)
- [AWS SAM CLI solução de problemas](#)

Arquivo de configuração do AWS SAM CLI

A interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) oferece suporte a um arquivo de configuração em nível de projeto que você pode usar para configurar valores de parâmetros de AWS SAMCLI comando.

Para obter a documentação sobre como criar e usar arquivos de configuração, consulte [Configurando o AWS SAM CLI](#).

Tópicos

- [Configurações padrão do arquivo de configuração](#)
- [Formatos de arquivo de configuração suportados](#)
- [Especificar um arquivo de configuração](#)
- [Noções básicas de arquivos de configuração](#)
- [Regras de valores de parâmetros](#)
- [Precedência de configuração](#)
- [Criando e modificando arquivos de configuração](#)

Configurações padrão do arquivo de configuração

AWS SAM usa as seguintes configurações padrão do arquivo de configuração:

- Name: `samconfig`.
- Localização - Na raiz do seu projeto. Este é o mesmo local do seu arquivo `template.yaml`.

- Formato – TOML. Para saber mais, consulte [TOML](#) na documentação do TOML.

Veja a seguir um exemplo de estrutura de projeto que inclui o nome e o local do arquivo de configuração padrão:

```
sam-app
### README.md
### __init__.py
### events
### hello_world
### samconfig.toml
### template.yaml
### tests
```

O seguinte é um arquivo `samconfig.toml` de exemplo:

```
...
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
```

Formatos de arquivo de configuração suportados

Os formatos TOML e [YAML | YML] são suportados. Veja a seguinte sintaxe básica:

TOML

```
version = 0.1
[environment]
[environment.command]
[environment.command.parameters]
option = parameter value
```

YAML

```
version: 0.1
environment:
  command:
    parameters:
      option: parameter value
```

Especificar um arquivo de configuração

Por padrão, o AWS SAMCLI procura um arquivo de configuração na seguinte ordem:

1. Arquivo de configuração personalizado — Se você usar a opção `--config-file` para especificar o nome e o local do arquivo, o AWS SAMCLI primeiro procurará esse arquivo.
2. **samconfig.toml** Arquivo padrão - Esse é o nome e o formato do arquivo de configuração padrão, localizado na raiz do seu projeto. Se você não especificar um arquivo de configuração personalizado, e AWS SAMCLI procurará esse arquivo em seguida.
3. **samconfig.[yaml|yml]** arquivo — Se o `samconfig.toml` não existir na raiz do seu projeto, AWS SAMCLI procurará esse arquivo.

Veja a seguir um exemplo de especificação de arquivo de configuração personalizado usando a opção `--config-file`:

```
$ sam deploy --config-file myconfig.yaml
```

 Note

O `--config-file` parâmetro deve ser relativo à localização do arquivo de AWS SAM modelo, pois é AWS SAMCLI necessário determinar o contexto no qual a configuração é aplicada. O `samconfig.toml` arquivo gerencia as configurações da sua versão do AWS SAMCLI, e a CLI procura o `samconfig.toml` arquivo (ou o parâmetro do arquivo de configuração substituído) na pasta relativa do arquivo. `template.yaml`

Noções básicas de arquivos de configuração

Environment

Um ambiente é um identificador nomeado que contém um conjunto exclusivo de configurações. Você pode ter vários ambientes em um único AWS SAM aplicativo.

O nome do ambiente padrão é `default`.

Use a AWS SAMCLI `--config-env` opção para especificar o ambiente a ser usado.

Command

O comando é o comando AWS SAMCLI para o qual especificar valores de parâmetros.

Para especificar valores de parâmetros para todos os comandos, use o identificador `global`.

Ao referenciar um comando AWS SAM CLI, substitua espaços () e hífen (-) por sublinhados (`_`).
Veja os exemplos a seguir:

- `build`
- `local_invoke`
- `local_start_api`

Parâmetros

Os parâmetros são especificados como pares de chave-valor.

- A chave é o nome da opção do comando AWS SAM CLI.
- O valor é o valor a ser especificado.

Ao especificar chaves, use o nome da opção de comando de formato longo e substitua hífens (-) por sublinhados (_). Veja os exemplos a seguir:

- `region`
- `stack_name`
- `template_file`

Regras de valores de parâmetros

TOML

- Os valores booleanos podem ser `true` ou `false`. Por exemplo, `.confirm_changeset = true`
- Para valores de string, use aspas (""). Por exemplo, `.region = "us-west-2"`
- Para valores de lista, use aspas (") e separe cada valor usando um espaço (). Por exemplo: `capabilities = "CAPABILITY_IAM CAPABILITY_NAMED_IAM"`.
- Para valores que contêm uma lista de pares de valores-chave, os pares são delimitados por espaço () e o valor de cada par é cercado por aspas codificadas (\ " \"). Por exemplo, `.tags = "project=\"my-application\" stage=\"production\""`
- Para valores de parâmetros que podem ser especificados várias vezes, o valor é uma matriz de argumentos. Por exemplo: `image_repositories = ["my-function-1=image-repo-1", "my-function-2=image-repo-2"]`.

YAML

- Os valores booleanos podem ser `true` ou `false`. Por exemplo, `.confirm_changeset: true`
- Para entradas que contêm um único valor de string, as aspas (") são opcionais. Por exemplo, `.region: us-west-2` Isso inclui entradas que contêm vários pares de valores-chave fornecidos como uma única sequência de caracteres. Este é um exemplo:

```
$ sam deploy --tags "foo=bar hello=world"
```

```
default:
  deploy:
    parameters:
      tags: foo=bar hello=world
```

- Para entradas que contêm uma lista de valores ou entradas que podem ser usadas várias vezes em um único comando, especifique-as como uma lista de cadeias de caracteres.

Este é um exemplo:

```
$ sam remote invoke --parameter "InvocationType=Event" --parameter "LogType=None"
```

```
default:
  remote_invoke:
    parameters:
      parameter:
        - InvocationType=Event
        - LogType=None
```

Precedência de configuração

Ao configurar valores, a seguinte precedência ocorre:

- Os valores dos parâmetros que você fornece na linha de comando têm precedência sobre os valores correspondentes no arquivo de configuração e na seção `Parameters` do arquivo de modelo.
- Se a opção `--parameter-overrides` for usada na linha de comando ou em seu arquivo de configuração com a chave `parameter_overrides`, seus valores terão precedência sobre os valores na seção `Parameters` do arquivo de modelo.
- Em seu arquivo de configuração, as entradas fornecidas para um comando específico têm precedência sobre as entradas globais. No exemplo a seguir, o comando `sam deploy` usará o nome da pilha `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
```

Criando e modificando arquivos de configuração

Criação de arquivos de configuração

Quando você cria um aplicativo usando `sam init`, um arquivo `samconfig.toml` padrão é criado. Você também pode criar manualmente seu arquivo de configuração.

Modificar arquivos de configuração

Você pode modificar manualmente seus arquivos de configuração. Além disso, durante qualquer fluxo interativo CLI do AWS SAM, os valores configurados serão exibidos entre colchetes ([]). Se você modificar esses valores, AWS SAM CLI atualizará seu arquivo de configuração.

Veja a seguir um exemplo de fluxo interativo usando o comando `sam deploy --guided`:

```
$ 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
```

Ao modificar seu arquivo de configuração, AWS SAM CLI manipula os valores globais da seguinte forma:

- Se o valor do parâmetro existir na seção `global` do seu arquivo de configuração, AWS SAMCLI não grava o valor na seção de comando específica.
- Se o valor do parâmetro existir nas seções de comando específicas `global` e nas seções de comando específicas, AWS SAMCLI excluirá a entrada específica em favor do valor global.

Gerenciar versões do AWS SAM CLI

Gerencie suas versões da interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) por meio de atualização, downgrade e desinstalação. Opcionalmente, você pode baixar e instalar a compilação AWS SAMCLI noturna.

Tópicos

- [Atualizar o AWS SAM CLI](#)
- [Desinstalando o AWS SAMCLI](#)
- [Pare de usar o Homebrew para gerenciar o AWS SAMCLI](#)
- [Gerenciando a AWS SAMCLI compilação noturna](#)
- [Instalar a AWS SAM CLI em um ambiente virtual usando o pip](#)
- [Gerenciamento o AWS SAM CLI com o Homebrew](#)
- [Solução de problemas](#)

Atualizar o AWS SAM CLI

Linux

Para atualizar o AWS SAMCLI no Linux, siga as instruções de instalação em [Instalar o AWS SAM CLI](#), mas adicione a opção `--update` ao comando de instalação, da seguinte forma:

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

macOS

O AWS SAMCLI deve ser atualizado pelo mesmo método usado para instalá-lo. Recomendamos que você use o instalador do pacote para instalar e atualizar AWS SAMCLI o.

Para atualizar o AWS SAMCLI usando o instalador de pacotes, instale a versão mais recente do pacote. Para obter instruções, consulte [Instalar o AWS SAM CLI](#).

Windows

Para atualizar o AWS SAMCLI, repita as etapas de instalação do Windows [Instale o AWS SAM CLI](#) novamente.

Desinstalando o AWS SAMCLI

Linux

Para desinstalar o AWS SAMCLI no Linux, você deve excluir o symlink e o diretório de instalação executando os seguintes comandos:

1. Localize o symlink e instale caminhos.

- Encontre o link simbólico usando o `which` comando:

```
which sam
```

A saída mostra o caminho em que os AWS SAM binários estão localizados, por exemplo:

```
/usr/local/bin/sam
```

- Encontre o diretório para o qual o link simbólico aponta usando o `ls` comando:

```
ls -l /usr/local/bin/sam
```

No exemplo a seguir, o diretório de instalação é `/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. Exclua o symlink.

```
sudo rm /usr/local/bin/sam
```

3. Exclua o diretório de instalação.

```
sudo rm -rf /usr/local/aws-sam-cli
```

macOS

Desinstale o AWS SAMCLI por meio do mesmo método usado para instalá-lo. Recomendamos que você use o instalador do pacote para instalar AWS SAMCLI o.

Se você instalou o AWS SAMCLI usando o instalador do pacote, siga estas etapas para desinstalar.

Para desinstalar o AWS SAM CLI

1. Remova o programa AWS SAMCLI modificando e executando o seguinte:

```
$ sudo rm -rf /path-to/aws-sam-cli
```

- a. **sudo**— Se o seu usuário tiver permissões de gravação no local onde o AWS SAMCLI programa está instalado, não sudo é necessário. Caso contrário, o sudo será obrigatório.
- b. **/path-to**— Caminho até onde você instalou o AWS SAMCLI programa. O local padrão é `/usr/local`.

2. Remova o AWS SAMCLI \$PATH modificando e executando o seguinte:

```
$ sudo rm -rf /path-to-symlink-directory/sam
```

- a. **sudo**— Se o seu usuário tiver permissões de gravação para \$PATH, não sudo é necessário. Caso contrário, o sudo será obrigatório.
- b. **path-to-symlink-directory**— Sua variável de \$PATH ambiente. O local padrão é `/usr/local/bin`.

3. Verifique se o AWS SAMCLI está desinstalado executando o seguinte:

```
$ sam --version  
command not found: sam
```

Windows

Para desinstalar o AWS SAMCLI usando as Configurações do Windows, siga estas etapas:

1. No menu Iniciar, procure por “Adicionar ou remover programas”.
2. Escolha o resultado chamado AWS SAM Interface da linha de comando e escolha Desinstalar para executar o desinstalador.
3. Confirme que você deseja desinstalar AWS SAMCLI o.

Pare de usar o Homebrew para gerenciar o AWS SAMCLI

Se você usa Homebrew para instalar e atualizar o AWS SAMCLI, recomendamos usar um método AWS compatível. Siga estas instruções para mudar para um método compatível.

Para deixar de usar o Homebrew

1. Siga as instruções em [Desinstalando uma CLI Homebrew instalada AWS SAM](#) para desinstalar a versão do Homebrew gerenciada.
2. Siga as instruções em [Instale o AWS SAM CLI](#) para instalar a CLI AWS SAM usando um método compatível.

Gerenciando a AWS SAMCLI compilação noturna

Você pode baixar e instalar a versão AWS SAMCLI noturna. Ela contém uma versão de pré-lançamento do código AWS SAMCLI que pode ser menos estável do que a versão de produção. Quando instalado, você pode usar a compilação noturna com o comando `sam-nightly`. Você pode instalar e usar as versões de produção e compilação noturna do AWS SAMCLI ao mesmo tempo.

Note

A compilação noturna não contém uma versão de pré-lançamento da imagem de compilação. Por isso, criar seu aplicativo sem servidor com a opção `--use-container` usa a versão de produção mais recente da imagem de compilação.

Instalando a compilação AWS SAMCLI noturna

Para instalar a AWS SAMCLI versão noturna, siga estas instruções.

Linux

Você pode instalar a versão de compilação noturna do AWS SAMCLI na plataforma Linux x86_64 usando o instalador do pacote.

Para instalar a compilação AWS SAMCLI noturna

1. Baixe o instalador do pacote [sam-cli-nightly](#) no aws-sam-cli GitHub repositório.
2. Siga as etapas de [instalação do AWS SAMCLI](#) para instalar o pacote de compilação noturna.

macOS

Você pode instalar a versão de compilação noturna do AWS SAMCLI na macOS usando o instalador do pacote de compilação noturna.

Para instalar a compilação AWS SAMCLI noturna

1. Baixe o instalador do pacote para sua plataforma [sam-cli-nightly](#) no aws-sam-cli GitHub repositório.
2. Siga as etapas de [instalação do AWS SAMCLI](#) para instalar o pacote de compilação noturna.

Windows

A versão de compilação noturna do AWS SAMCLI está disponível com este link para download: [AWS SAMCLI compilação noturna](#). Para instalar a compilação noturna no Windows, execute as mesmas etapas do [Instale o AWS SAM CLI](#), mas use o link de download da compilação noturna.

Para verificar se você instalou a versão de compilação noturna, execute o comando `sam-nightly --version`. A saída desse comando está no formato `1.X.Y.dev<YYYYMMDDHHmm>`, por exemplo:

```
SAM CLI, version 1.20.0.dev202103151200
```

Mude do Homebrew para o instalador de pacotes

Se você estiver usando o Homebrew para instalar e atualizar a compilação AWS SAMCLI noturna e quiser passar a usar o instalador de pacotes, siga estas etapas.

Para mudar do Homebrew para o instalador de pacotes

1. Desinstale a Homebrew compilação AWS SAMCLI noturna instalada.

```
$ brew uninstall aws-sam-cli-nightly
```

2. Verifique se a compilação AWS SAMCLI noturna está desinstalada executando o seguinte:

```
$ sam-nightly --version  
zsh: command not found: sam-nightly
```

3. Siga as etapas da seção anterior para instalar a compilação AWS SAMCLI noturna.

Instalar a AWS SAM CLI em um ambiente virtual usando o pip

Recomendamos usar o instalador de pacotes nativo para instalar AWS SAMCLI o. Se você precisar usar o pip, recomendamos que você instale a AWS SAM CLI em um ambiente virtual. Isso garante um ambiente de instalação limpo e um ambiente isolado caso ocorram erros.

Note

Em 24 de outubro de 2023, AWS SAM CLI o suporte para o. Python 3.7 Para saber mais, consulte [AWS SAMCLI descontinuando o suporte para Python 3.7](#).

Como instalar a AWS SAM CLI em um ambiente virtual

1. Em um diretório inicial de sua escolha, crie um ambiente virtual e dê um nome a ele.

Linux / macOS

```
$ mkdir project  
$ cd project  
$ python3 -m venv venv
```

Windows

```
> mkdir project  
> cd project  
> py -3 -m venv venv
```

2. Ative o ambiente virtual.

Linux / macOS

```
$ . venv/bin/activate
```

O prompt mudará para mostrar que seu ambiente virtual está ativo.

```
(venv) $
```

Windows

```
> venv\Scripts\activate
```

O prompt mudará para mostrar que seu ambiente virtual está ativo.

```
(venv) >
```

3. Instale a AWS SAMCLI em seu ambiente virtual.

```
(venv) $ pip install --upgrade aws-sam-cli
```

4. Verifique se o AWS SAMCLI está instalado corretamente.

```
(venv) $ sam --version  
SAM CLI, version 1.94.0
```

5. Use o comando `deactivate` para sair do ambiente virtual. Sempre que você iniciar uma nova sessão, deverá ativar novamente o ambiente.

Gerenciamento o AWS SAM CLI com o Homebrew

Note

A partir de setembro de 2023, não AWS manterá mais o Homebrew instalador AWS gerenciado do AWS SAMCLI (`aws/tap/aws-sam-cli`). Para continuar usando o Homebrew, use o instalador gerenciado pela comunidade (`aws-sam-cli`). A partir de setembro de 2023, qualquer comando Homebrew que faça referência a `aws/tap/aws-sam-cli` será redirecionado para o `aws-sam-cli`.

Recomendamos que você use nossos métodos de [instalação](#) e [atualização](#) compatíveis.

Instalar a AWS SAMCLI usando o Homebrew

Note

Essas instruções usam o AWS SAMCLI Homebrew instalador gerenciado pela comunidade. Para obter mais suporte, consulte o repositório [homebrew-core](#).

Para instalar o AWS SAM CLI

1. Execute o seguinte:

```
$ brew install aws-sam-cli
```

2. Verifique a instalação:

```
$ sam --version
```

Após a instalação bem-sucedida do AWS SAMCLI, você deverá ver uma saída como a seguinte:

```
SAM CLI, version 1.94.0
```

Atualizando o uso do AWS SAMCLI Homebrew

Para atualizar a AWS SAM CLI usando o Homebrew, execute este comando:

```
$ brew upgrade aws-sam-cli
```

Desinstalando uma CLI Homebrew instalada AWS SAM

Se a AWS SAMCLI foi instalada usando o Homebrew, siga estas etapas para desinstalá-la.

Para desinstalar o AWS SAM CLI

1. Execute o seguinte:


```
$ brew uninstall aws-sam-cli
```

2. Verifique se o AWS SAMCLI está desinstalado executando o seguinte:

```
$ sam --version  
command not found: sam
```

Como mudar para o instalador Homebrew gerenciado pela comunidade

Se você estiver usando o Homebrew instalador AWS gerenciado (`aws/tap/aws-sam-cli`) e preferir continuar usando Homebrew, recomendamos mudar para o Homebrew instalador gerenciado pela comunidade (`aws-sam-cli`).

Para alternar em um único comando, execute o seguinte:

```
$ brew uninstall aws-sam-cli && brew untap aws/tap && brew cleanup aws/tap && brew  
update && brew install aws-sam-cli
```

Siga estas instruções para executar cada comando individualmente.

Para mudar para o Homebrew instalador gerenciado pela comunidade

1. Desinstale a Homebrew versão AWS gerenciada do AWS SAMCLI:

```
$ brew uninstall aws-sam-cli
```

2. Verifique se o AWS SAMCLI foi desinstalado:

```
$ which sam  
sam not found
```

3. Remova o AWS SAMCLI toque AWS gerenciado:

```
$ brew untap aws/tap
```

Se você receber um erro como o seguinte, adicione a opção `--force` e tente novamente.

```
Error: Refusing to untap aws/tap because it contains the following installed  
formulae or casks:
```

```
aws-sam-cli-nightly
```

4. Remova os arquivos em cache do instalador AWS gerenciado:

```
$ brew cleanup aws/tap
```

5. Atualize o Homebrew e todas as fórmulas:

```
$ brew update
```

6. Instale a versão gerenciada pela comunidade do AWS SAMCLI:

```
$ brew install aws-sam-cli
```

7. Verifique se a instalação do AWS SAM CLI foi bem-sucedida.

```
$ sam --version  
SAM CLI, version 1.94.0
```

Solução de problemas

Se você encontrar erros ao instalar ou usar o AWS SAMCLI, consulte [AWS SAM CLI solução de problemas](#).

Configurando AWS credenciais

A interface de linha de AWS SAM comando (CLI) exige que você defina AWS credenciais para que ela possa fazer chamadas para AWS serviços em seu nome. Por exemplo, ele AWS SAMCLI faz chamadas para o Amazon S3 e. CloudFormation

Talvez você já tenha definido AWS credenciais para trabalhar com AWS ferramentas, como uma das AWS SDKs ou a. AWS CLI Caso contrário, este tópico mostra as abordagens recomendadas para definir AWS credenciais.

Para definir AWS as credenciais, você deve ter o ID da chave de acesso e a chave de acesso secreta do usuário do IAM que você deseja configurar. Para obter informações sobre chaves de acesso IDs e chaves de acesso secretas, consulte [Gerenciamento de chaves de acesso para usuários do IAM](#) no Guia do usuário do IAM.

Em seguida, determine se você tem o AWS CLI instalado. Em seguida, siga as instruções em uma das seguintes seções:

Usando o AWS CLI

Se você tiver o AWS CLI instalado, use o `aws configure` comando e siga as instruções:

```
$ 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]:
```

Para obter mais informações, sobre o comando `aws configure` consulte [Configuração rápida do comando AWS CLI](#) no AWS Command Line Interface Guia do usuário do.

Não usando o AWS CLI

Se você não tiver o AWS CLI instalado, poderá criar um arquivo de credenciais ou definir variáveis de ambiente:

- Arquivo de credenciais — Você pode definir as credenciais no arquivo de AWS credenciais do seu sistema local. Esse arquivo deve estar localizado em uma das localizações a seguir:
 - `~/.aws/credentials` no Linux ou macOS
 - `C:\Users\USERNAME\.aws\credentials` no Windows

Esse arquivo deve conter linhas no seguinte formato:

```
[default]
aws_access_key_id = your_access_key_id
aws_secret_access_key = your_secret_access_key
```

- Variáveis de ambiente — Você pode definir as variáveis de ambiente `AWS_ACCESS_KEY_ID` e `AWS_SECRET_ACCESS_KEY`.

Para definir essas variáveis no Linux ou no macOS, use o comando `export`:

```
export AWS_ACCESS_KEY_ID=your_access_key_id
```

```
export AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

Para definir essas variáveis no Windows, use o comando configurar:

```
set AWS_ACCESS_KEY_ID=your_access_key_id  
set AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

Telemetria no AWS SAMCLI

Na AWS, desenvolvemos e lançamos serviços com base no que aprendemos com as interações com os clientes. Usamos o feedback dos clientes para iterar nosso produto. A telemetria é uma informação adicional que nos ajuda a compreender melhor as necessidades dos nossos clientes, diagnosticar problemas e fornecer recursos que melhoram a experiência do cliente.

A interface de linha de AWS SAM comando (CLI) coleta telemetria, como métricas genéricas de uso, informações do sistema e do ambiente e erros. Para obter detalhes sobre os tipos de telemetria coletados, consulte [Tipos de informação coletados](#).

O AWS SAM CLI local não coleta informações pessoais, como nomes de usuário ou endereços de e-mail. Ele também não extrai informações sigilosas em nível de projeto.

Os clientes controlam se a telemetria está ativada e podem alterar as configurações a qualquer momento. Se a telemetria permanecer ativada, o AWS SAM CLI local enviará dados de telemetria em segundo plano sem exigir nenhuma interação adicional com o cliente.

Desative a telemetria para uma sessão

Nos sistemas operacionais macOS e Linux, você pode desativar a telemetria de uma única sessão. Para desativar a telemetria da sessão atual, execute o comando a seguir a fim de definir a variável de ambiente SAM_CLI_TELEMETRY como false. Repita o comando para cada novo terminal ou sessão.

```
export SAM_CLI_TELEMETRY=0
```

Desative a telemetria do seu perfil em todas as sessões

Execute os comandos a seguir para desativar a telemetria em todas as sessões quando você estiver executando o AWS SAMCLI no seu sistema operacional.

Para desabilitar a telemetria no Linux

1. Execute:

```
echo "export SAM_CLI_TELEMETRY=0" >> ~/.profile
```

2. Execute:

```
source ~/.profile
```

Para desabilitar a telemetria no macOS

1. Execute:

```
echo "export SAM_CLI_TELEMETRY=0" >> ~/.profile
```

2. Execute:

```
source ~/.profile
```

Para desabilitar a telemetria no Windows

Você pode definir a variável de ambiente temporariamente durante a vida útil da janela do terminal com o seguinte comando:

Se estiver usando o prompt de comando:

```
set SAM_CLI_TELEMETRY=0
```

Se estiver usando PowerShell:

```
$env:SAM_CLI_TELEMETRY=0
```

Para definir a variável de ambiente permanentemente no prompt de comando ou PowerShell, use o seguinte comando:

```
setx SAM_CLI_TELEMETRY 0
```

 Note

As alterações não entrarão em vigor até que o terminal tenha sido fechado e reaberto.

Tipos de informação coletados

- Informações de uso — Os comandos e subcomandos genéricos que os clientes executam.
- Erros e informações de diagnóstico — O status e a duração dos comandos que os clientes executam, incluindo códigos de saída, nomes de exceções internas e falhas na conexão com o Docker.
- Informações do sistema e do ambiente — A versão do Python, o sistema operacional (Windows, Linux ou macOS), o ambiente no qual AWS SAMCLI ele é executado (por exemplo AWS CodeBuild, um kit de ferramentas do AWS IDE ou um terminal) e os valores de hash dos atributos de uso.

Saiba mais

Os dados de telemetria que eles AWS SAMCLI coletam seguem as políticas de privacidade de dados. AWS Para saber mais, consulte:

- [AWS Termos de serviço](#)
- [Perguntas frequentes sobre privacidade de dados](#)

AWS SAM CLI solução de problemas

Esta seção fornece detalhes sobre como solucionar mensagens de erro ao usar, instalar e gerenciar a interface de linha de AWS Serverless Application Model comando (AWS SAMCLI).

Tópicos

- [Solução de problemas](#)
- [Mensagens de erro](#)
- [Mensagens de aviso](#)

Solução de problemas

Para obter orientações sobre solução de problemas relacionadas a AWS SAMCLI, consulte [Solução de problemas de erros de instalação do](#).

Mensagens de erro

Erro de curl: “curl: (6) Não foi possível resolver: ...”

Ao tentar invocar o endpoint do API Gateway, você receberá o seguinte erro:

```
curl: (6) Could not resolve: endpointdomain (Domain name not found)
```

Isso significa que você tentou enviar uma solicitação para um domínio que não é válido. Isso pode acontecer se o aplicativo com tecnologia sem servidor falhar na implantação bem-sucedida ou se você tiver um erro de digitação no comando. curl Verifique se o aplicativo foi implantado com êxito usando o CloudFormation console ou o AWS CLI e verifique se o curl comando está correto.

Erro: não é possível encontrar informações exatas do recurso com o nome da pilha fornecido

Ao executar o `sam remote invoke` comando em um aplicativo que contém um único recurso da Lambda Event Function, você vê o seguinte erro:

```
Error: Can't find exact resource information with given <stack-name>. Please provide full resource ARN or --stack-name to resolve the ambiguity.
```

Possível causa: você não forneceu a opção **--stack-name**.

Se o ARN de uma função não for fornecido como argumento, o comando `sam remote invoke` exigirá que a opção `--stack-name` seja fornecida.

Solução: forneça a opção **--stack-name**.

Veja um exemplo a seguir.

```
$ 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\"}}%
```

Erro: não é possível encontrar informações do recurso com o nome da pilha

Ao executar o comando `sam remote invoke` e passar o ARN de uma função do Lambda como argumento, você vê o seguinte erro:

```
Error: Can't find resource information from stack name (<stack-name>) and resource id
(<function-id>)
```

Possível causa: você tem o valor do nome da pilha definido em seu **samconfig.toml** arquivo.

A AWS SAMCLI primeiro verifica o nome da pilha em seu arquivo `samconfig.toml`. Se especificado, o argumento é passado como um valor lógico de ID.

Solução: em vez disso, passe o ID lógico da função.

Você pode passar o ID lógico da função como argumento em vez do ARN da função.

Solução: remova o valor do nome da pilha do seu arquivo de configuração.

Você pode remover o valor do nome da pilha do seu arquivo de configuração. Isso impede que o AWS SAMCLI transmita o ARN da função como um valor lógico de ID.

Execute `sam build` depois de modificar seu arquivo de configuração.

Erro: "Falha ao criar recursos gerenciados: não é possível localizar credenciais"

Ao executar o comando `sam deploy`, você vê o seguinte erro:

```
Error: Failed to create managed resources: Unable to locate credentials
```

Isso significa que você não configurou AWS credenciais para permitir que o AWS SAMCLI faça chamadas AWS de serviço. Para corrigir isso, você deve configurar AWS as credenciais. Para obter mais informações, consulte [Configurando AWS credenciais](#).

Erro: `FileNotFoundException` no Windows

Ao executar comandos na CLI do AWS SAM no Windows, você pode ver o seguinte erro:

Error: FileNotFoundError

Possível causa: Eles AWS SAMCLI podem interagir com caminhos de arquivo que excedam o limite máximo de caminhos do Windows.

Solução: para solucionar esse problema, o novo comportamento de caminhos longos deverá ser habilitado. Para fazer isso, consulte [Habilitar caminhos longos no Windows 10, versão 1607 e posterior](#) na Documentação para desenvolvimento de aplicações do Windows.

Erro: resolvedor de dependências do pip...

Exemplo de texto de erro:

```
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.
```

Possível causa: Se você usa pip para instalar pacotes, as dependências entre os pacotes podem entrar em conflito.

Cada versão do pacote `aws-sam-cli` depende de uma versão do pacote `aws-sam-translator`. Por exemplo, a v1.58.0 do `aws-sam-cli` pode depender da v1.51.0 do `aws-sam-translator`.

Se você instalar o AWS SAMCLI usando pip, e, em seguida, instalar outro pacote que dependa de uma versão mais recente do `aws-sam-translator`, ocorrerá o seguinte:

- A versão mais recente do `aws-sam-translator` será instalada.
- A versão atual do `aws-sam-cli` e a versão mais recente do `aws-sam-translator` podem não ser compatíveis.
- Quando você usa o AWS SAMCLI, o erro do resolvedor de dependências ocorrerá.

Soluções:

1. Use o instalador de pacotes AWS SAMCLI nativo.

- a. Desinstalar a AWS SAM CLI usando pip. Para obter instruções, consulte [Desinstalando o AWS SAMCLI](#).
 - b. Instale o AWS SAMCLI usando o instalador de pacote nativo. Para obter instruções, consulte [Instale o AWS SAM CLI](#).
 - c. Quando necessário, atualize o AWS SAMCLI usando o instalador de pacotes nativo. Para obter instruções, consulte [Atualizar o AWS SAM CLI](#).
2. Se você precisar usar o pip, recomendamos que você instale a AWS SAM CLI em um ambiente virtual. Isso garante um ambiente de instalação limpo e um ambiente isolado caso ocorram erros. Para obter instruções, consulte [Instalar a AWS SAM CLI em um ambiente virtual usando o pip](#).

Erro: Esse comando 'remote' não existe

Ao executar o comando `sam remote invoke`, você vê o seguinte erro:

```
$ 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'.
```

Possível causa: Sua versão do AWS SAMCLI está desatualizada.

O AWS SAMCLI `sam remote invoke` comando foi lançado com a AWS SAMCLI versão 1.88.0. Você pode verificar sua versão executando o comando `sam --version`.

Solução: atualize seu AWS SAMCLI para a versão mais recente.

Para obter instruções, consulte [Atualizar o AWS SAM CLI](#).

Erro: Executando AWS Os projetos SAM exigem localmente Docker. Você o instalou?

Ao executar o comando `sam local start-api`, você vê o seguinte erro:

```
Error: Running AWS SAM projects locally requires Docker. Have you got it installed?
```

Isso significa que você não instalou o Docker corretamente. Docker é necessário para testar seu aplicativo localmente. Para corrigir isso, siga as instruções para instalar o Docker para seu host de desenvolvimento. Para obter mais informações, consulte [Instalação do Docker](#).

Erro: restrições de segurança não satisfeitas

Ao executar o `aws sam deploy --guided`, você recebe a pergunta *Function* may not have authorization defined, Is this okay? [y/N]. Se responder a essa solicitação com **N** (a resposta padrão), você receberá o seguinte erro:

```
Error: Security Constraints Not Satisfied
```

O aviso está informando que o aplicativo que você está prestes a implantar pode ter uma API do Amazon API Gateway acessível ao público configurada sem autorização. Ao responder **N** a essa solicitação, você está dizendo que isso não está certo.

Para corrigir isso, você tem as seguintes opções:

- Configurar seu aplicativo com autorização. Para obter informações sobre como configurar a autorização, consulte [Controle o acesso à API com seu AWS SAM modelo](#).
- Se sua intenção é ter um endpoint de API acessível ao público sem autorização, reinicie sua implantação e responda a essa pergunta com **Y** para indicar que você concorda com a implantação.

mensagem: Token de autenticação ausente

Ao tentar invocar o endpoint do API Gateway, você receberá o seguinte erro:

```
{"message": "Missing Authentication Token"}
```

Isso significa que você tentou enviar uma solicitação para o domínio correto, mas o URI não é reconhecível. Para corrigir isso, verifique o URL completo e atualize o comando curl com o URL correto.

Mensagens de aviso

Aviso:... AWS não manterá mais o Homebrew instalador para AWS SAM ...

Ao instalar o AWS SAMCLI usando o Homebrew, você verá a seguinte mensagem de aviso:

```
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 potencial: AWS não manter mais o Homebrew suporte.

A partir de setembro de 2023, não AWS manterá mais o Homebrew instalador do AWS SAMCLI.

Solução: use um AWS método de instalação suportado.

- Você pode encontrar métodos de instalação AWS compatíveis em [Instale o AWS SAM CLI](#).

Solução: para continuar usando Homebrew, use o instalador gerenciado pela comunidade.

- Você pode usar o instalador gerenciado pela comunidade Homebrew a seu critério. Para obter instruções, consulte [Gerenciamento o AWS SAM CLI com o Homebrew](#).

AWS SAM referência do conector

Esta seção contém informações de referência para o tipo de recurso do conector AWS Serverless Application Model (AWS SAM). Para obter uma introdução aos conectores, consulte [Gerenciando permissões de recursos com AWS SAM conectores](#).

Tipos de recursos de origem e destino suportados para conectores

O tipo de recurso `AWS::Serverless::Connector` oferece suporte a um número selecionado de conexões entre os recursos de origem e destino. Ao configurar conectores em seu AWS SAM modelo, use a tabela a seguir para referenciar as conexões suportadas e as propriedades que precisam ser definidas para cada tipo de recurso de origem e destino. Para obter mais informações sobre como configurar conectores no seu modelo, consulte [AWS::Serverless::Connector](#).

Para recursos de origem e destino, quando definidos no mesmo modelo, use a propriedade `Id`. Opcionalmente, um `Qualifier` pode ser adicionado para restringir o escopo do seu recurso definido. Quando o recurso não estiver no mesmo modelo, use uma combinação de propriedades compatíveis.

Para solicitar novas conexões, [envie um novo problema](#) no serverless-application-model AWS GitHub repositório.

Source type (Tipo de origem)	Tipo de destino	Permissões	Propriedades da fonte	Propriedades de Destino
AWS::ApiGateway::RestApi	AWS::Lambda::Function	Write	Idou Qualifier ResourceId , e Type	Idou Arn e Type
AWS::ApiGateway::RestApi	AWS::Serverless::Function	Write	Idou Qualifier ResourceId , e Type	Idou Arn e Type
AWS::ApiGatewayV2::Api	AWS::Lambda::Function	Write	Idou Qualifier ResourceId , e Type	Idou Arn e Type
AWS::ApiGatewayV2::Api	AWS::Serverless::Function	Write	Idou Qualifier ResourceId , e Type	Idou Arn e Type
AWS::AppSync::DataSource	AWS::DynamoDB::Table	Read	Idou RoleName e Type	Idou Arn e Type
AWS::AppSync::DataSource	AWS::DynamoDB::Table	Write	Idou RoleName e Type	Idou Arn e Type
AWS::AppSync::DataSource	AWS::Events::EventBus	Write	Idou RoleName e Type	Idou Arn e Type
AWS::AppSync::DataSource	AWS::Lambda::Function	Write	Idou RoleName e Type	Idou Arn e Type

Source type (Tipo de origem)	Tipo de destino	Permissões	Propriedades da fonte	Propriedades de Destino
AWS::AppSync::DataSource	AWS::Serverless::Function	Write	Idou RoleName e Type	Idou Arn e Type
AWS::AppSync::DataSource	AWS::Serverless::SimpleTable	Read	Idou RoleName e Type	Idou Arn e Type
AWS::AppSync::DataSource	AWS::Serverless::SimpleTable	Write	Idou RoleName e Type	Idou Arn e Type
AWS::AppSync::GraphQLApi	AWS::Lambda::Function	Write	Idou ResourceId e Type	Idou Arn e Type
AWS::AppSync::GraphQLApi	AWS::Serverless::Function	Write	Idou ResourceId e Type	Idou Arn e Type
AWS::DynamoDB::Table	AWS::Lambda::Function	Read	Idou Arn e Type	Idou RoleName e Type
AWS::DynamoDB::Table	AWS::Serverless::Function	Read	Idou Arn e Type	Idou RoleName e Type
AWS::Events::Rule	AWS::Events::Event Bus	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Events::Rule	AWS::Lambda::Function	Write	Idou Arn e Type	Idou Arn e Type

Source type (Tipo de origem)	Tipo de destino	Permissões	Propriedades da fonte	Propriedades de Destino
AWS::Events::Rule	AWS::Serverless::Function	Write	Idou Arn e Type	Idou Arn e Type
AWS::Events::Rule	AWS::Serverless::StateMachine	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Events::Rule	AWS::SNS::Topic	Write	Idou Arn e Type	Idou Arn e Type
AWS::Events::Rule	AWS::SQS::Queue	Write	Idou Arn e Type	Idou ArnQueueUrl, e Type
AWS::Events::Rule	AWS::StepFunctions::StateMachine	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Lambda::Function	AWS::DynamoDB::Table	Read, Write	Idou RoleName e Type	Idou Arn e Type
AWS::Lambda::Function	AWS::Events::Event Bus	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Lambda::Function	AWS::Lambda::Function	Write	Idou RoleName e Type	Idou Arn e Type

Source type (Tipo de origem)	Tipo de destino	Permissões	Propriedades da fonte	Propriedades de Destino
AWS::Lambda::Function	AWS::Location::PlaceIndex	Read	Idou RoleName e Type	Idou Arn e Type
AWS::Lambda::Function	AWS::S3::Bucket	Read, Write	Idou RoleName e Type	Idou Arn e Type
AWS::Lambda::Function	AWS::Serverless::Function	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Lambda::Function	AWS::Serverless::SimpleTable	Read, Write	Idou RoleName e Type	Idou Arn e Type
AWS::Lambda::Function	AWS::Serverless::StateMachine	Read, Write	Idou RoleName e Type	Idou ArnName, e Type
AWS::Lambda::Function	AWS::SNS::Topic	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Lambda::Function	AWS::SQS::Queue	Read, Write	Idou RoleName e Type	Idou Arn e Type
AWS::Lambda::Function	AWS::StepFunctions::StateMachine	Read, Write	Idou RoleName e Type	Idou ArnName, e Type

Source type (Tipo de origem)	Tipo de destino	Permissões	Propriedades da fonte	Propriedades de Destino
AWS::S3::Bucket	AWS::Lambda::Function	Write	Idou Arn e Type	Idou Arn e Type
AWS::S3::Bucket	AWS::Serverless::Function	Write	Idou Arn e Type	Idou Arn e Type
AWS::Serverless::Api	AWS::Lambda::Function	Write	Idou Qualifier ResourceId , e Type	Idou Arn e Type
AWS::Serverless::Api	AWS::Serverless::Function	Write	Idou Qualifier ResourceId , e Type	Idou Arn e Type
AWS::Serverless::Function	AWS::DynamoDB::Table	Read, Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::Function	AWS::Events::Event Bus	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::Function	AWS::Lambda::Function	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::Function	AWS::S3::Bucket	Read, Write	Idou RoleName e Type	Idou Arn e Type

Source type (Tipo de origem)	Tipo de destino	Permissões	Propriedades da fonte	Propriedades de Destino
AWS::Serverless::Function	AWS::Serverless::Function	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::Function	AWS::Serverless::SimpleTable	Read, Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::Function	AWS::Serverless::StateMachine	Read, Write	Idou RoleName e Type	Idou ArnName, e Type
AWS::Serverless::Function	AWS::SNS::Topic	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::Function	AWS::SQS::Queue	Read, Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::Function	AWS::StepFunctions::StateMachine	Read, Write	Idou RoleName e Type	Idou ArnName, e Type
AWS::Serverless::HttpApi	AWS::Lambda::Function	Write	Idou Qualifier ResourceId , e Type	Idou Arn e Type
AWS::Serverless::HttpApi	AWS::Serverless::Function	Write	Idou Qualifier ResourceId , e Type	Idou Arn e Type

Source type (Tipo de origem)	Tipo de destino	Permissões	Propriedades da fonte	Propriedades de Destino
AWS::Serverless::SimpleTable	AWS::Lambda::Function	Read	Idou Arn e Type	Idou RoleName e Type
AWS::Serverless::SimpleTable	AWS::Serverless::Function	Read	Idou Arn e Type	Idou RoleName e Type
AWS::Serverless::StateMachine	AWS::DynamoDB::Table	Read, Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::StateMachine	AWS::Events::Event Bus	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::StateMachine	AWS::Lambda::Function	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::StateMachine	AWS::S3::Bucket	Read, Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::StateMachine	AWS::Serverless::Function	Write	Idou RoleName e Type	Idou Arn e Type

Source type (Tipo de origem)	Tipo de destino	Permissões	Propriedades da fonte	Propriedades de Destino
AWS::Serverless::StateMachine	AWS::Serverless::SimpleTable	Read, Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::StateMachine	AWS::Serverless::StateMachine	Read, Write	Idou RoleName e Type	Idou ArnName, e Type
AWS::Serverless::StateMachine	AWS::SNS::Topic	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::StateMachine	AWS::SQS::Queue	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Serverless::StateMachine	AWS::StepFunctions::StateMachine	Read, Write	Idou RoleName e Type	Idou ArnName, e Type
AWS::SNS::Topic	AWS::Lambda::Function	Write	Idou Arn e Type	Idou Arn e Type
AWS::SNS::Topic	AWS::Serverless::Function	Write	Idou Arn e Type	Idou Arn e Type

Source type (Tipo de origem)	Tipo de destino	Permissões	Propriedades da fonte	Propriedades de Destino
AWS::SNS::Topic	AWS::SQS::Queue	Write	Idou Arn e Type	Idou ArnQueueUrl, e Type
AWS::SQS::Queue	AWS::Lambda::Function	Read, Write	Idou Arn e Type	Idou RoleName e Type
AWS::SQS::Queue	AWS::Serverless::Function	Read, Write	Idou Arn e Type	Idou RoleName e Type
AWS::StepFunctions::StateMachine	AWS::DynamoDB::Table	Read, Write	Idou RoleName e Type	Idou Arn e Type
AWS::StepFunctions::StateMachine	AWS::Events::Event Bus	Write	Idou RoleName e Type	Idou Arn e Type
AWS::StepFunctions::StateMachine	AWS::Lambda::Function	Write	Idou RoleName e Type	Idou Arn e Type
AWS::StepFunctions::StateMachine	AWS::S3::Bucket	Read, Write	Idou RoleName e Type	Idou Arn e Type

Source type (Tipo de origem)	Tipo de destino	Permissões	Propriedades da fonte	Propriedades de Destino
AWS::Step Functions::StateMachine	AWS::Serverless::Function	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Step Functions::StateMachine	AWS::Serverless::SimpleTable	Read, Write	Idou RoleName e Type	Idou Arn e Type
AWS::Step Functions::StateMachine	AWS::Serverless::StateMachine	Read, Write	Idou RoleName e Type	Idou ArnName, e Type
AWS::Step Functions::StateMachine	AWS::SNS::Topic	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Step Functions::StateMachine	AWS::SQS::Queue	Write	Idou RoleName e Type	Idou Arn e Type
AWS::Step Functions::StateMachine	AWS::Step Functions::StateMachine	Read, Write	Idou RoleName e Type	Id ou Arn, Name e Type

Políticas do IAM criadas por conectores

Esta seção documenta as políticas AWS Identity and Access Management (IAM) que são criadas AWS SAM ao usar conectores.

AWS::DynamoDB::Table para AWS::Lambda::Function

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::Lambda::Function.

Categorias de acesso

Read

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:DescribeStream",
        "dynamodb:GetRecords",
        "dynamodb:GetShardIterator",
        "dynamodb:ListStreams"
      ],
      "Resource": [
        "%{Source.Arn}/stream/*"
      ]
    }
  ]
}
```

AWS::Events::Rule para AWS::SNS::Topic

Tipo de política

[AWS::SNS::TopicPolicy](#) associado ao AWS::SNS::Topic.

Categorias de acesso

Write

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
```

```
    "Resource": "%{Destination.Arn}",
    "Action": "sns:Publish",
    "Condition": {
      "ArnEquals": {
        "aws:SourceArn": "%{Source.Arn}"
      }
    }
  }
]
```

AWS::Events::Rule para AWS::Events::EventBus

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::Events::Rule.

Categorias de acesso

Write

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutEvents"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

AWS::Events::Rule para AWS::StepFunctions::StateMachine

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::Events::Rule.

Categorias de acesso

Write


```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "states:StartExecution"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

AWS::Events::Rule para AWS::Lambda::Function

Tipo de política

[AWS::Lambda::Permission](#) associado ao AWS::Lambda::Function.

Categorias de acesso

Write

```
{
  "Action": "lambda:InvokeFunction",
  "Principal": "events.amazonaws.com",
  "SourceArn": "%{Source.Arn}"
}
```

AWS::Events::Rule para AWS::SQS::Queue

Tipo de política

[AWS::SQS::QueuePolicy](#) associado ao AWS::SQS::Queue.

Categorias de acesso

Write

```
{
  "Statement": [
    {
```

```
"Effect": "Allow",
"Principal": {
  "Service": "events.amazonaws.com"
},
"Resource": "%{Destination.Arn}",
"Action": "sqs:SendMessage",
"Condition": {
  "ArnEquals": {
    "aws:SourceArn": "%{Source.Arn}"
  }
}
]
```

AWS::Lambda::Function para AWS::Lambda::Function

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::Lambda::Function.

Categorias de acesso

Write

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeAsync",
        "lambda:InvokeFunction"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

AWS::Lambda::Function para AWS::S3::Bucket

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::Lambda::Function.

Categorias de acesso

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 para AWS::DynamoDB::Table

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::Lambda::Function.

Categorias de acesso

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 para AWS::SQS::Queue

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::Lambda::Function.

Categorias de acesso

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 para AWS::SNS::Topic

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::Lambda::Function.

Categorias de acesso

Write

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": [
        "${Destination.Arn}"
      ]
    }
  ]
}
```

```
]
}
```

AWS::Lambda::Function para AWS::StepFunctions::StateMachine

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::Lambda::Function.

Categorias de acesso

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 para AWS::Events::EventBus

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::Lambda::Function.

Categorias de acesso

Write

```

{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "events:PutEvents"
            ],
            "Resource": [
                "%{Destination.Arn}"
            ]
        }
    ]
}

```



```
}  
]  
}
```

AWS::Lambda::Function para AWS::Location::PlaceIndex

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::Lambda::Function.

Categorias de acesso

Read

```
{  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "geo:DescribePlaceIndex",  
        "geo:GetPlace",  
        "geo:SearchPlaceIndexForPosition",  
        "geo:SearchPlaceIndexForSuggestions",  
        "geo:SearchPlaceIndexForText"  
      ],  
      "Resource": [  
        "%{Destination.Arn}"  
      ]  
    }  
  ]  
}
```

AWS::ApiGatewayV2::Api para AWS::Lambda::Function

Tipo de política

[AWS::Lambda::Permission](#) associado ao AWS::Lambda::Function.

Categorias de acesso

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 para AWS::Lambda::Function

Tipo de política

[AWS::Lambda::Permission](#) associado ao AWS::Lambda::Function.

Categorias de acesso

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 para AWS::SQS::Queue

Tipo de política

[AWS::SQS::QueuePolicy](#) associado ao AWS::SQS::Queue.

Categorias de acesso

Write

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "sns.amazonaws.com"
      },
      "Resource": "%{Destination.Arn}",
      "Action": "sqs:SendMessage",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": "%{Source.Arn}"
        }
      }
    }
  ]
}
```

```
}  
}  
}  
]  
}
```

AWS::SNS::Topic para AWS::Lambda::Function

Tipo de política

[AWS::Lambda::Permission](#) associado ao AWS::Lambda::Function.

Categorias de acesso

Write

```
{  
  "Action": "lambda:InvokeFunction",  
  "Principal": "sns.amazonaws.com",  
  "SourceArn": "%{Source.Arn}"  
}
```

AWS::SQS::Queue para AWS::Lambda::Function

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::Lambda::Function.

Categorias de acesso

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 para AWS::Lambda::Function

Tipo de política

[AWS::Lambda::Permission](#) associado ao AWS::Lambda::Function.

Categorias de acesso

Write

```
{
  "Action": "lambda:InvokeFunction",
  "Principal": "s3.amazonaws.com",
  "SourceArn": "%{Source.Arn}",
  "SourceAccount": "${AWS::AccountId}"
}
```

AWS::StepFunctions::StateMachine para AWS::Lambda::Function

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::StepFunctions::StateMachine.

Categorias de acesso

Write

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeAsync",
        "lambda:InvokeFunction"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

AWS::StepFunctions::StateMachine para AWS::SNS::Topic

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::StepFunctions::StateMachine.

Categorias de acesso

Write

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

AWS::StepFunctions::StateMachine para AWS::SQS::Queue

Tipo de política

[Política gerenciada pelo cliente](#) associada à função `AWS::StepFunctions::StateMachine`.

Categorias de acesso

Write

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sqs:SendMessage"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

`AWS::StepFunctions::StateMachine` para `AWS::S3::Bucket`

Tipo de política

[Política gerenciada pelo cliente](#) associada à função `AWS::StepFunctions::StateMachine`.

Categorias de acesso

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 para AWS::DynamoDB::Table

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::StepFunctions::StateMachine.

Categorias de acesso

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 para AWS::StepFunctions::StateMachine

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::StepFunctions::StateMachine.

Categorias de acesso

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 para AWS::Events::EventBus

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::StepFunctions::StateMachine.

Categorias de acesso

Write

```

{
  "Statement": [
    {

```

```

    "Effect": "Allow",
    "Action": [
        "events:PutEvents"
    ],
    "Resource": [
        "%{Destination.Arn}"
    ]
}
]
}

```

AWS::AppSync::DataSource para AWS::DynamoDB::Table

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::AppSync::DataSource.

Categorias de acesso

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 para AWS::Lambda::Function

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::AppSync::DataSource.

Categorias de acesso

Write

```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeAsync",
        "lambda:InvokeFunction"
      ],
      "Resource": [
        "%{Destination.Arn}",
        "%{Destination.Arn}:*"
      ]
    }
  ]
}

```

```
}
```

AWS::AppSync::DataSource para AWS::Events::EventBus

Tipo de política

[Política gerenciada pelo cliente](#) associada à função AWS::AppSync::DataSource.

Categorias de acesso

Write

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutEvents"
      ],
      "Resource": [
        "%{Destination.Arn}"
      ]
    }
  ]
}
```

AWS::AppSync::GraphQLApi para AWS::Lambda::Function

Tipo de política

[AWS::Lambda::Permission](#) associado ao AWS::Lambda::Function.

Categorias de acesso

Write

```
{
  "Action": "lambda:InvokeFunction",
  "Principal": "appsync.amazonaws.com",
  "SourceArn": "arn:${AWS::Partition}:appsync:${AWS::Region}:${AWS::AccountId}:apis/
%{Source.ResourceId}"
}
```

Instalando o Docker para usar com o AWS SAM CLI

O Docker é um aplicativo que executa contêineres em sua máquina. Com Docker, AWS SAM pode fornecer um ambiente local semelhante a AWS Lambda um contêiner para criar, testar e depurar seus aplicativos sem servidor.

Note

O Docker é necessário somente para testar seus aplicativos localmente e para criar pacotes de implantação usando a opção `--use-container`.

Tópicos

- [Instalar Docker](#)
- [Próximas etapas](#)

Instalar Docker

Siga essas instruções para instalar a Docker, siga as instruções para seu sistema operacional.

Linux

O Docker está disponível em muitos sistemas operacionais diferentes, incluindo a maioria das distribuições modernas do Linux, como o CentOS, o Debian, e o Ubuntu. Para obter informações sobre a instalação Docker em seu sistema operacional específico, consulte [Obtenha o Docker](#) no site Docker Docs.

Para instalar Docker no Amazon Linux 2 ou Amazon Linux 2023

1. Atualize os pacotes instalados e o cache de pacotes em sua instância.

```
$ sudo yum update -y
```

2. Instale o pacote do Docker Community Edition mais recente.

- No Amazon Linux 2, execute o seguinte:

```
$ sudo amazon-linux-extras install docker
```

- No Amazon Linux 2023, execute o seguinte:

```
$ sudo yum install -y docker
```

3. Inicie o serviço Docker.

```
$ sudo service docker start
```

4. Adicione o `ec2-user` ao grupo `docker`, para que você possa executar comandos Docker sem usar o `sudo`.

```
$ sudo usermod -a -G docker ec2-user
```

5. Obtenha as novas permissões de grupo `docker` efetuando logout e login novamente. Para fazer isso, feche a janela do terminal SSH atual e reconecte-se à sua instância em uma nova. Sua nova sessão SSH deverá ter as permissões de grupo `docker` apropriadas.
6. Verifique se o `ec2-user` pode executar comandos do Docker sem usar o `sudo`.

```
$ docker ps
```

Você deverá ver a saída a seguir, confirmando que o Docker está instalado e em execução:

CONTAINER ID	IMAGE	COMMAND	CREATED
STATUS	PORTS	NAMES	

Note

No Linux, para criar e executar Lambda Event Functions com uma arquitetura de conjunto de instruções diferente da sua máquina host, há etapas adicionais a serem configuradas. Docker Por exemplo, para executar funções `arm64` em uma máquina `x86_64`, você pode executar o seguinte comando para configurar o Docker daemon: `docker run --rm --privileged multiarch/qemu-user-static --reset -p yes`.

Se você tiver problemas ao instalar o Docker, consulte [Solução de problemas de erros de instalação do](#). Ou consulte a seção [Solução](#) de problemas das Post-installation etapas para Linux no site do Docker Docs.

macOS

Note

Docker O desktop é oficialmente suportado, mas a partir da versão 1.47.0 do AWS SAMCLI, você pode usar alternativas, desde que elas usem o tempo de execução do Docker.

1. Instalar o Docker

O AWS SAMCLI tem suporte para a execução do Docker no macOS Sierra 10.12 ou posterior. Para saber como instalar o Docker, consulte [Instalar o Docker Desktop para Mac](#) no site do Docs Docker.

2. Como configurar seus drives compartilhados

Isso AWS SAMCLI exige que o diretório do projeto, ou qualquer diretório principal, esteja listado em um drive compartilhado. Isso pode exigir o compartilhamento de arquivos. Para obter mais informações, consulte o tópico de solução de problemas [de montagem de volumes requer compartilhamento de arquivos](#) na documentação Docker.

3. Verificar a instalação

Depois que o Docker estiver instalado, verifique se está funcionando. Confirme também se você pode executar comandos Docker na linha de comando (por exemplo, `docker ps`). Você não precisa instalar, buscar ou retirar nenhum contêiner. O AWS SAMCLI faz isso automaticamente conforme necessário.

Se você tiver problemas na instalação do Docker, para obter mais dicas de solução de problemas, consulte a seção [Solução de problemas e diagnóstico](#) do site do Docs Docker.

Windows

Note

AWS SAM suporta oficialmente o Docker Desktop. No entanto, a partir da versão 1.47.0 do AWS SAMCLI, você pode usar alternativas, desde que elas usem o tempo de execução do Docker.

1. Instalar o Docker.

O Docker desktop é compatível com o sistema operacional Windows mais recente. Para versões antigas do Windows, a caixa de ferramentas Docker está disponível. Escolha sua versão do Windows para as etapas corretas de instalação do Docker:

- Para instalar o Docker no Windows 10, consulte [Instalar o Docker Desktop para Windows](#) no site do Docker Documentos.
- DockerPara instalar em versões anteriores do Windows, consulte [A caixa de Docker ferramentas no repositório da Docker caixa](#) de ferramentas GitHub .

2. Como configurar seus drives compartilhados.

Isso AWS SAMCLI exige que o diretório do projeto, ou qualquer diretório principal, esteja listado em um drive compartilhado. Em alguns casos, você deve compartilhar sua unidade para o Docker funcionar corretamente.

3. Verifique a instalação.

Depois que o Docker estiver instalado, verifique se está funcionando. Confirme também se você pode executar comandos Docker na linha de comando (por exemplo, `docker ps`). Você não precisa instalar, buscar ou retirar nenhum contêiner. O AWS SAMCLI faz isso automaticamente conforme necessário.

Se você tiver problemas na instalação do Docker, para obter mais dicas de solução de problemas, consulte a seção [Solução de problemas e diagnóstico](#) do site do Docs Docker.

Próximas etapas

Para saber como instalar o AWS SAMCLI, consulte [Instale o AWS SAM CLI](#).

Instalar Finch para usar com o AWS SAM CLI

Finch AWS é a ferramenta de desenvolvimento de contêineres que fornece uma alternativa Docker, particularmente benéfica em sistemas macOS e Linux. Para obter mais informações, consulte [Finch on GitHub](#). Com Finch, AWS SAM pode fornecer um ambiente local semelhante a AWS Lambda um contêiner para criar, testar e depurar seus aplicativos sem servidor.

 Note

Você precisa do Finch para testar suas aplicações localmente, para criar pacotes de implantação usando a opção `--use-container` e para criar funções de imagem OCI. A partir da versão 1.145.0 da AWS SAM CLI, a AWS SAM CLI Finch oferece suporte como alternativa ao tempo de execução de contêiner a. Docker

Tópicos

- [Instalar Finch](#)
- [Configurar Finch por AWS SAM CLI](#)
- [Verificar a instalação](#)
- [Parando Finch](#)
- [Solução de problemas](#)
- [Próximas etapas](#)

Instalar Finch

Siga essas instruções para instalar a Finch, siga as instruções para seu sistema operacional.

Pré-requisitos

 Note

Fincha integração na AWS SAM CLI está disponível nos sistemas macOS e Linux. Para usuários do Windows, considere usar uma máquina virtual ou um ambiente Linux para obter a funcionalidade Finch.

Versões do macOS:

- macOS 15 Sequoia
- macOS 14 Sonoma

 Note

O Finch oferece suporte para as duas versões mais recentes do macOS.

Linux

- Linux Kernel v4.x+

Instalar o Finch

macOS

Para instalar Finch No macOS

1. Instale o Finch usando a CLI do brew. Se solicitado, digite sua senha do macOS.

```
$ brew install finch
```

2. Verifique a versão instalada do Finch.

```
$ finch --version
```

3. Inicialize a máquina virtual do Finch (primeira configuração).

```
$ finch vm init
```

4. Inicie a máquina virtual do Finch.

```
$ finch vm start
```

5. Verifique o status da máquina virtual do Finch.

```
$ finch vm status
```

6. Verifique se o Finch está funcionando corretamente.

```
$ finch --info  
$ finch ps
```

Para obter mais informações, consulte [Installing Finch on macOS](#).

Linux

Você pode instalar o Finch no Amazon Linux, Ubuntu ou usando um método de instalação genérico. Para obter mais informações, consulte [Installing Finch on Linux](#).

Configurar Finch por AWS SAM CLI

Depois da instalação do Finch, a AWS SAM CLI detecta e usa automaticamente Finch como tempo de execução do contêiner quando Docker não é executada.

Note

O Finch atua como um substituto para o Docker. Se você tiver ambos Finch instalados e Docker em execução, a AWS SAM CLI prioriza o Finch por padrão.

Configuração de preferência do administrador (macOS)

Você pode configurar a AWS SAM CLI para usar um tempo de execução de contêiner específico como padrão criando um arquivo de preferências do administrador. Essa preferência de todo o sistema substitui o comportamento de fallback padrão.

Opções de configuração disponíveis

- Defina o Finch como runtime do contêiner preferencial:

```
$ sudo /usr/libexec/PlistBuddy -c "Add :DefaultContainerRuntime string finch" /Library/Preferences/com.amazon.samcli.plist
```

- Defina o Docker como runtime do contêiner preferencial:

```
$ sudo /usr/libexec/PlistBuddy -c "Add :DefaultContainerRuntime string docker" /Library/Preferences/com.amazon.samcli.plist
```

- Remover preferência (retornar ao comportamento padrão):

```
$ sudo /usr/libexec/PlistBuddy -c "Remove :DefaultContainerRuntime" /Library/Preferences/com.amazon.samcli.plist
```

 Note

Quando nenhuma preferência do administrador é definida, a AWS SAM CLI usa Docker por padrão quando disponível, Finch como alternativa quando não Docker está em execução.

Diferenças de montagem de diretórios (macOS)

No macOS, os comandos AWS SAM locais falharão se você localizar seu projeto fora do diretório inicial (~) ou. /Volumes Esse design restringe a montagem automática de diretórios por motivos de segurança sem o seu conhecimento.

Opções de mitigação

- Opção 1: abra a configuração do Finch em um editor de texto ~/.finch/finch.yaml, adicione o caminho do projeto na seção `additional_directories` e reinicie a máquina virtual do Finch. Para obter informações adicionais, consulte [Finch documentation](#).

```
$ finch vm stop  
$ finch vm start
```

- Opção 2: mova seu projeto para o diretório inicial.

Verificar a instalação

Depois de instalar e iniciar Finch, verifique se ele funciona com a AWS SAM CLI.

Verificação bancária

Execute o comando a seguir para verificar se a AWS SAM CLI pode usar: Finch

```
$ sam build --use-container
```

Você deve ver a AWS SAM CLI sendo usada Finch para criar seu aplicativo em um ambiente de contêiner.

 Note

O Finch atua como um substituto para o Docker. Se Docker estiver em execução, a AWS SAM CLI Docker priorizará por padrão.

Teste Finch como um substituto

Para testar o Finch como alternativa quando o Docker estiver em execução, você precisa interromper o Docker primeiro.

Para testar Finch funcionalidade de fallback

1. Interrompa o Docker usando os comandos apropriados para o seu sistema operacional.

macOS:

```
$ docker stop $(docker ps -q)
$ pkill -f "Docker Desktop"
```

A verificação do Docker foi interrompida.

```
$ docker ps
```

Linux

```
$ docker stop $(docker ps -q)
$ sudo systemctl stop docker
```

A verificação do Docker foi interrompida.

```
$ sudo systemctl status docker
```

2. Execute o `sam build --use-container` comando para verificar se a AWS SAM CLI é usada automaticamente Finch como alternativa.

Parando Finch

Se você precisar interromper o Finch, use os comandos a seguir:

macOS:

```
$ finch vm stop  
$ sudo pkill -f finch
```

A verificação do Finch foi interrompida.

```
$ finch vm status
```

Linux

```
$ sudo systemctl stop finch  
$ sudo pkill -f finch
```

A verificação do Finch foi interrompida.

```
$ sudo systemctl status finch
```

Solução de problemas

Se você encontrar problemas com o Finch:

- Verifique se o sistema operacional atende aos pré-requisitos.
- Verifique se você habilitou a virtualização no seu sistema.
- Para macOS, verifique se você tem espaço em disco suficiente para a máquina virtual.
- Para Linux, verifique se você tem as permissões adequadas para executar os comandos `systemctl`.
- Se você encontrar falhas de CloudFormation implantação com erros de “tipo de mídia não suportado” ao enviar imagens de várias arquiteturas para o Amazon Elastic Container Registry (Amazon ECR), crie um único artefato de imagem em vez de usar imagens de várias arquiteturas. Isso ocorre porque o Finch envia o índice e a imagem da imagem para o Amazon ECR, enquanto o Docker envia somente a imagem.
- Se a construção multiplataforma falhar quando a arquitetura da função de destino não corresponder à arquitetura da máquina host no Linux, habilite a emulação multiplataforma executando `sudo finch run --privileged --rm tonistiigi/binfmt:master --install all`.

Para solução de problemas adicionais, consulte [Finch documentation](#) ou o [AWS SAM CLI solução de problemas](#).

Próximas etapas

Agora que você Finch instalou, você pode usá-lo com a AWS SAM CLI para:

- Crie aplicações com tecnologia sem servidor usando contêineres com `sam build --use-container`. Para obter mais informações, consulte [the section called “sam build”](#).
- Teste as funções de eventos Lambda localmente com o `sam local invoke`. Para obter mais informações, consulte [the section called “SAM invocação local”](#).
- Inicie um API Gateway local com `sam local start-api`. Para obter mais informações, consulte [the section called “SAM API de inicialização local”](#).
- Inicie um endpoint do Lambda local com `sam local start-lambda`. Para obter mais informações, consulte [the section called “start-lambda do local do sam”](#).

Para obter mais informações sobre o uso da AWS SAM CLI com contêineres, consulte [Compilação padrão com AWS SAM](#)

Repositórios de imagens para AWS SAM

AWS SAM simplifica as tarefas de integração contínua e entrega contínua (CI/CD) para aplicativos sem servidor com a ajuda da criação de imagens de contêiner. As imagens AWS SAM fornecidas incluem a interface de linha de AWS SAM comando (CLI) e ferramentas de construção para vários tempos de AWS Lambda execução suportados. Isso facilita a criação e o empacotamento de aplicativos sem servidor usando o AWS SAM CLI. Você pode usar essas imagens com CI/CD sistemas para automatizar a criação e a implantação de AWS SAM aplicativos. Para obter exemplos, consulte [Implemente com CI/CD sistemas e pipelines](#).

AWS SAM os URIs da imagem do contêiner de construção são marcados com a versão do AWS SAMCLI incluída nessa imagem. Se for especificada a URI não marcada, a versão mais recente será usada. Por exemplo, `public.ecr.aws/sam/build-nodejs20.x` usa a imagem mais recente. No entanto, `public.ecr.aws/sam/build-nodejs20.x:1.24.1` usa a imagem que contém a versão 1.24.1 da AWS SAM CLI.

A partir da versão 1.33.0 do AWS SAMCLI, as imagens de arm64 contêiner x86_64 e ambas estão disponíveis para tempos de execução compatíveis. Para obter mais informações, consulte [Cotas do Lambda](#) no AWS Lambda Guia do desenvolvedor do e.

Note

Antes da versão 1.22.0 do AWS SAMCLI, DockerHub era o repositório padrão do qual a imagem do contêiner era AWS SAMCLI extraída. A partir da versão 1.22.0, o repositório padrão mudou para Amazon Elastic Container Registry Public (Amazon ECR Public). Para extrair uma imagem de contêiner de um repositório diferente do padrão atual, você pode usar o comando [sam build](#) com a opção `--build-image`. Os exemplos no final deste tópico mostram como criar aplicativos usando imagens de DockerHub repositório.

URIs de repositório de imagem

A tabela a seguir lista os URIs das imagens públicas de contêiner de [compilação do](#) Amazon ECR que você pode usar para criar e empacotar aplicativos sem servidor. AWS SAM

Note

O Amazon ECR Public foi substituído DockerHub a partir da versão 1.22.0 AWS SAMCLI. Se você estiver usando uma versão anterior do AWS SAMCLI, recomendamos que você atualize.

Runtime	Amazon ECR Public
Runtime personalizado (AL2023)	public.ecr. aws/sam/build-provided.al2023
Runtime personalizado (AL2)	public.ecr. aws/sam/build-provided.al2
Runtime personalizado	public.ecr. aws/sam/build-fornecido
Java 25	public.ecr. aws/sam/build-java25
Java 21	public.ecr. aws/sam/build-java21
Java 17	public.ecr. aws/sam/build-java17
Java 17 (AL 2023)	public.ecr. aws/sam/build-java17 de 2023
Java 11	public.ecr. aws/sam/build-java11

Runtime	Amazon ECR Public
Java 11 (AL2023)	public.ecr.aws/sam/build-java11 de 2023
Java 8 (AL2023)	public.ecr.aws/sam/build-java 8 de 2023
Java 8	public.ecr.aws/sam/build-java8
.NET 9	public.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	public.ecr.aws/sam/build-nodejs24.x
Node.js 22	public.ecr.aws/sam/build-nodejs22.x
Node.js 20	public.ecr.aws/sam/build-nodejs20.x
Node.js 18	public.ecr.aws/sam/build-nodejs18.x
Node.js 16	public.ecr.aws/sam/build-nodejs16.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	public.ecr.aws/sam/build-python3.11
Python 3.10	public.ecr.aws/sam/build-python3.10
Python 3.9	public.ecr.aws/sam/build-python3.9
Python 3.8	public.ecr.aws/sam/build-python 3.8
Ruby 4.0	public.ecr.aws/sam/build-ruby4.0

Runtime	Amazon ECR Public
Ruby 3.4	public.ecr.aws/sam/build-ruby 3.4
Ruby 3.3	public.ecr.aws/sam/build-ruby3,3
Ruby 3.2	public.ecr.aws/sam/build-ruby3,2

Exemplos

Os dois exemplos de comandos a seguir criam aplicações usando imagens de contêiner do repositório de imagens:

Crie uma aplicação Node.js 24 usando uma imagem de contêiner extraída do Amazon ECR:

```
$ sam build --use-container --build-image public.ecr.aws/sam/build-nodejs24.x
```

Crie um recurso de função usando a imagem do contêiner Python 3.14 extraída do Amazon ECR:

```
$ sam build --use-container --build-image Function1=public.ecr.aws/sam/build-python3.14
```

Implantando aplicativos sem servidor gradualmente com AWS SAM

AWS Serverless Application Model (AWS SAM) vem embutido [CodeDeploy](#) para fornecer AWS Lambda implantações graduais. Com apenas algumas linhas de configuração, AWS SAM faz o seguinte para você:

- Implanta novas versões da sua Lambda Event Function e cria automaticamente aliases que apontam para a nova versão.
- Mude gradualmente o tráfego do cliente para a nova versão até que você esteja satisfeito com o funcionamento esperado. Se uma atualização não funcionar corretamente, você poderá reverter as alterações.
- Define funções de teste pré-tráfego e pós-tráfego para verificar se o código recém-implantado está configurado corretamente e se seu aplicativo opera conforme o esperado.
- Reverte automaticamente a implantação se CloudWatch os alarmes forem acionados.

 Note

Se você habilitar implantações graduais por meio do seu AWS SAM modelo, um CodeDeploy recurso será criado automaticamente para você. Você pode visualizar o CodeDeploy recurso diretamente por meio do Console de gerenciamento da AWS.

Exemplo

O exemplo a seguir demonstra o uso da função Lambda CodeDeploy para transferir gradualmente os clientes para sua versão recém-implantada da função 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
```

Essas revisões do AWS SAM modelo fazem o seguinte:

- **AutoPublishAlias:** Ao adicionar essa propriedade e especificar um nome de alias, o AWS SAM
 - Detecta quando um novo código está sendo implantado, com base em alterações no URI do Amazon S3 da função do Lambda.
 - Cria e publica uma versão atualizada dessa função com o código mais recente.

- Cria um alias com um nome fornecido por você (a menos que já exista um alias) e aponta para a versão atualizada da função do Lambda. As invocações de função devem usar o qualificador de alias para aproveitar isso. Se você não estiver familiarizado com o controle de versão e os aliases da função do Lambda, consulte [AWS Lambda Controle de versão e aliases de funções](#).
- **Deployment Preference Type:** No exemplo anterior, 10% do seu tráfego de clientes é imediatamente transferido para sua nova versão. Após 10 minutos, todo o tráfego é transferido para a nova versão. No entanto, se seus testes de pré-tráfego ou pós-tráfego falharem, ou se um CloudWatch alarme for acionado, CodeDeploy reverte sua implantação. É possível especificar como o tráfego deve ser transferido entre versões das seguintes maneiras:
 - **Canary:** o tráfego é deslocado em dois incrementos. É possível escolher entre opções de canário predefinidas. As opções especificam a porcentagem de tráfego que é transferida para a versão atualizada da função do Lambda no primeiro incremento e o intervalo, em minutos, antes que o tráfego restante seja transferido no segundo incremento.
 - **Linear:** o tráfego é deslocado em incrementos iguais com um número igual de minutos entre cada incremento. Você pode escolher entre opções lineares predefinidas que especificam a porcentagem de tráfego deslocado em cada incremento e o número de minutos entre cada incremento.
 - **AllAtOnce:** todo o tráfego é deslocado da função do Lambda original para a versão da função do Lambda atualizada de uma única vez.

A tabela a seguir descreve outras opções de mudança de tráfego que estão disponíveis além da usada no exemplo.

Tipo de preferência de implantação

Canary10Percent30Minutes

Canary10Percent5Minutes

Canary10Percent10Minutes

Canary10Percent15Minutes

Linear10PercentEvery10Minutes

Linear10PercentEvery1Minute

Tipo de preferência de implantação

Linear10PercentEvery2Minutes

Linear10PercentEvery3Minutes

AllAtOnce

- **Alarms:** são CloudWatch alarmes acionados por quaisquer erros gerados pela implantação. Quando encontrados, eles reverterem automaticamente sua implantação. Por exemplo, se o código atualizado que você está implantando causar erros no aplicativo. Outro exemplo é se alguma CloudWatch métrica personalizada especificada por você [AWS Lambda](#) violou o limite de alarme.
- **Hooks:** são funções de teste pré e pós-tráfego que executam verificações antes do início da mudança de tráfego para a nova versão e após a conclusão da mudança de tráfego.
 - **PreTraffic:** Antes do início da mudança de tráfego, CodeDeploy invoca a função Lambda do gancho de pré-tráfego. Essa função Lambda deve retornar CodeDeploy e indicar sucesso ou falha. Se a função falhar, ela aborta e reporta uma falha para o CloudFormation. Se a função for bem-sucedida, CodeDeploy prossegue com a mudança de tráfego.
 - **PostTraffic:** após a conclusão da mudança de tráfego, CodeDeploy invoca a função Lambda do gancho pós-tráfego. Isso é semelhante ao gancho pré-tráfego, em que a função deve retornar para CodeDeploy relatar um sucesso ou uma falha. Use ganchos de pós-tráfego para executar testes de integração ou outras ações de validação.

Para obter mais informações, consulte [Referência do SAM para implantações seguras](#).

Implantando gradualmente uma função de evento Lambda pela primeira vez

Ao implantar uma função Lambda gradualmente CodeDeploy, é necessária uma versão de função implantada anteriormente para transferir o tráfego. Portanto, sua primeira implantação deve ser realizada em duas etapas:

- **Etapas 1:** implante sua função do Lambda e crie aliases automaticamente com o `AutoPublishAlias`.
- **Etapas 2:** Execute sua implantação gradual com o `DeploymentPreference`.

Executar sua primeira implantação gradual em duas etapas fornece CodeDeploy uma versão anterior da função Lambda da qual transferir o tráfego.

Etapa 1: Implantar sua função de evento Lambda

```
Resources:
MyLambdaFunction:
  Type: AWS::Serverless::Function
  Properties:
    Handler: index.handler
    Runtime: nodejs20.x
    CodeUri: s3://bucket/code.zip

    AutoPublishAlias: live
```

Etapa 2: Execute sua implantação gradual

```
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
```

Saiba mais

Para ver um exemplo prático de configuração de uma implantação gradual, consulte o [Módulo 5 - Implantações canário](#) no The Complete AWS SAM Workshop.

Notas de referência importantes para AWS SAM

Esta seção contém notas e anúncios importantes para AWS Serverless Application Model (AWS SAM).

Tópicos

- [Observações importantes para 2023](#)

Observações importantes para 2023

Outubro de 2023

AWS SAMCLI descontinuando o suporte para Python 3.7

Publicado em 20/10/2023

Python 3.7 recebeu end-of-life o status em junho de 2023. O suporte AWS SAM CLI será interrompido Python 3.7 em 24 de outubro de 2023. Para obter mais informações, consulte o [anúncio](#) no aws-sam-cli GitHub repositório.

Essa alteração afeta os seguintes usuários:

- Se você usar Python 3.7 e instalar o AWS SAM CLI Through pip.
- Se você usa `aws-sam-cli` como uma biblioteca e cria seu aplicativo com Python 3.7.

Se você instalar e gerenciar o AWS SAM CLI por meio de outro método, você não será afetado.

Para usuários afetados, recomendamos que você atualize seu ambiente de desenvolvimento para Python 3.8 ou mais recente.

Essa alteração não afeta o suporte ao ambiente Python 3.7 AWS Lambda de tempo de execução. Para obter mais informações, consulte [Política de descontinuação de runtime](#) no Guia do desenvolvedor do AWS Lambda .

AWS SAM CLI Terraform Suporte

Esta seção aborda o uso da interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) com seus Terraform projetos e a Terraform nuvem.

Para fornecer feedback e enviar solicitações de atributos, crie um [Problema GitHub](#).

Tópicos

- [Conceitos básicos de Terraform suporte para AWS SAM CLI](#)
- [Usar o AWS SAM CLI por Terraform para depuração e teste locais](#)
- [Usar o AWS SAM CLI com Serverless.tf para depuração e teste locais](#)
- [AWS SAM CLI por Terraform referência](#)
- [O que é AWS SAM CLI suporte para Terraform?](#)

Conceitos básicos de Terraform suporte para AWS SAM CLI

Este tópico aborda como começar a usar a interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) com Terraform.

Para fornecer feedback e enviar solicitações de atributos, crie um [Problema GitHub](#).

Tópicos

- [AWS SAM CLI Terraform Pré-requisitos](#)
- [Utilizar AWS SAM CLI comandos com Terraform](#)
- [Configurado para Terraform projetos](#)
- [Configurado para Terraform Cloud](#)

AWS SAM CLI Terraform Pré-requisitos

Preencha todos os pré-requisitos para começar a usar o AWS SAMCLI com seus projetos Terraform.

1. Instalar ou atualizar a AWS SAM CLI

Para verificar se você tem o AWS SAMCLI instalado, execute o seguinte:

```
$ sam --version
```

Se o AWS SAMCLI já estiver instalado, a saída exibirá uma versão. Para atualizar para a versão mais recente, consulte [Atualizar o AWS SAM CLI](#).

Para obter instruções sobre como instalar o AWS SAMCLI junto com todos os seus pré-requisitos, consulte [Instale o AWS SAM CLI](#).

2. Instalar Terraform

Para verificar se você instalou o Terraform, execute o seguinte:

```
$ terraform -version
```

Para instalar Terraform, consulte [Instalar Terraform](#) no registro Terraform.

3. Instalar Docker para testes locais

O AWS SAM CLI exige Docker para testes locais. Para instalar o Docker, consulte [Instalando o Docker para usar com o AWS SAM CLI](#).

Utilizar AWS SAM CLI comandos com Terraform

Ao executar um comando AWS SAMCLI compatível, use a `--hook-name` opção e forneça o terraform valor. Este é um exemplo:

```
$ sam local invoke --hook-name terraform
```

Você pode configurar essa opção em seu arquivo AWS SAMCLI de configuração com o seguinte:

```
hook_name = "terraform"
```

Configurado para Terraform projetos

Conclua as etapas deste tópico para usar o AWS SAMCLI com projetos Terraform .

Nenhuma configuração adicional é necessária se você criar seus AWS Lambda artefatos fora do Terraform projeto. Veja [Usar o AWS SAM CLI por Terraform para depuração e teste locais](#) para começar a usar AWS SAMCLI o.

Se você criar seus artefatos Lambda em seus projetos Terraform, deverá fazer o seguinte:

1. Instalar o Python 3.8 ou mais recente
2. Instale a ferramenta Make.
3. Defina sua lógica de construção de artefatos Lambda em seu projeto Terraform.
4. Defina um recurso `sam metadata` para informar sua lógica de construção AWS SAMCLI.
5. Use o AWS SAMCLI `sam build` comando para criar seus artefatos do Lambda.

Instalar Python 3.8 ou mais recente

Python 3.8 ou mais recente é necessário para uso com o AWS SAMCLI. Quando você executa `sam build`, cria AWS SAMCLI que contém comandos `makefiles` Python para criar seus artefatos do Lambda.

Para obter instruções de instalação, consulte [Descarregando Python](#) no Guia para iniciantes do Python.

Verifique se o Python 3.8 ou mais recente foi adicionado ao caminho da sua máquina executando:

```
$ python --version
```

A saída deve exibir uma versão do Python que seja 3.8 ou mais recente.

Instale o Make ferramenta

O GNU [Make](#) é uma ferramenta que controla a geração de executáveis e outros arquivos não fonte para seu projeto. O AWS SAMCLI cria `makefiles` que dependem dessa ferramenta para criar seus artefatos Lambda.

Se você não tiver Make instalado em sua máquina local, instale-o antes de prosseguir.

Para Windows, você pode instalar usando o [Chocolatey](#). Para obter instruções, consulte [Usando o Chocolatey](#) em Como instalar e usar o “Make” no Windows

Defina a lógica de construção dos artefatos Lambda

Use o tipo de recurso `null_resource` Terraform para definir sua lógica de criação do Lambda. Veja a seguir um exemplo que usa um script de construção personalizado para criar uma função de evento 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"
  }
}
```

Defina um metadados do sam recurso

O recurso `sam_metadata` é um tipo de recurso `null_resource` Terraform que AWS SAMCLI fornece as informações necessárias para localizar seus artefatos do Lambda. É necessário um recurso `sam_metadata` exclusivo para cada função ou camada do Lambda em seu projeto. Para saber mais sobre esse tipo de recurso, consulte [null_resource](#) no registro Terraform.

Para definir um metadados do sam recurso

1. Nomeie seu recurso começando com `sam_metadata_` para identificá-lo como um recurso `sam_metadata`.
2. Defina as propriedades do artefato Lambda dentro do bloco `triggers` do seu recurso.
3. Especifique o `null_resource` que contém sua lógica de construção do Lambda com o argumento `depends_on`.

Veja um exemplo de modelo a seguir:

```
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
  ]
}
```

```
}
```

O seguinte é um exemplo do recurso `sam metadata`:

```
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
  ]
}
```

O conteúdo do seu recurso `sam metadata` variará com base no tipo de recurso Lambda (função ou camada) e no tipo de embalagem (ZIP ou imagem). Para obter mais informações e exemplos, consulte [recurso de metadados do sam](#).

Quando você configura um recurso `sam metadata` e usa um comando AWS SAMCLI compatível, o AWS SAMCLI gera o arquivo de metadados antes de executar o comando AWS SAMCLI. Depois de gerar esse arquivo, você pode usar a `--skip-prepare-infra` opção with future AWS SAMCLI commands para pular o processo de geração de metadados e economizar tempo. Essa opção só deve ser usada se você não tiver feito nenhuma alteração na infraestrutura, como criar novas funções do Lambda ou novos endpoints de API.

Use o comando AWS SAM CLI para criar seus artefatos Lambda

Use o AWS SAMCLI `sam build` comando para criar seus artefatos do Lambda. Quando você executa `sam build`, o AWS SAM CLI faz o seguinte:

1. Procura recursos `sam metadata` em seu projeto Terraform para conhecer e localizar seus recursos do Lambda.
2. Inicia sua lógica de criação do Lambda para criar seus artefatos do Lambda.
3. Cria um `.aws-sam` diretório que organiza seu Terraform projeto para uso com os AWS SAMCLI `sam local` comandos.

Para compilar com sam build

1. No diretório que contém seu módulo raiz Terraform, execute o seguinte:

```
$ sam build --hook-name terraform
```

2. Para criar uma função ou camada específica do Lambda, execute o seguinte

```
$ sam build --hook-name terraform lambda-resource-id
```

O ID do recurso Lambda pode ser o nome da função do Lambda ou o endereço completo do Terraform recurso, como `aws_lambda_function.list_books` ou `module.list_book_function.aws_lambda_function.this[0]`.

Se o código-fonte da função ou outros arquivos de configuração Terraform estiverem localizados fora do diretório que contém o módulo raiz Terraform, você precisará especificar o local. Use a opção `--terraform-project-root-path` para especificar o caminho absoluto ou relativo para o diretório de nível superior que contém esses arquivos. Este é um exemplo:

```
$ sam build --hook-name terraform --terraform-project-root-path ~/projects/terraform/demo
```

Crie usando um contêiner

Ao executar o AWS SAMCLI `sam build` comando, você pode configurar o AWS SAMCLI para criar seu aplicativo usando um Docker contêiner local.

Note

Você deve ter a Docker instalada e configurada. Para instruções, consulte [Instalando o Docker para usar com o AWS SAM CLI](#).

Para construir usando um contêiner

1. Crie um `Dockerfile` que contenha as ferramentas Terraform, Python e Make. Você também deve incluir o runtime da função do Lambda.

O seguinte é um exemplo de `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 libgcc libgcc-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. Use [docker build](#) para criar sua Docker imagem.

Este é um exemplo:

```
$ docker build --tag terraform-build:v1 <path-to-directory-containing-Dockerfile>
```

3. Execute o AWS SAMCLI `sam build` comando com `--use-container` as `--build-image` opções e.

Este é um exemplo:

```
$ sam build --use-container --build-image terraform-build:v1
```

Próximas etapas

Para começar a usar o AWS SAM CLI com seus projetos Terraform, consulte [Usar o AWS SAM CLI por Terraform para depuração e teste locais](#).

Configurado para Terraform Cloud

É altamente recomendável que você use Terraform v1.6.0 ou mais recente. Se estiver usando uma versão mais antiga, deverá gerar um arquivo de plano Terraform no local. O arquivo do plano local AWS SAM CLI fornece as informações necessárias para realizar testes e depuração locais.

Para gerar um arquivo de plano local

Note

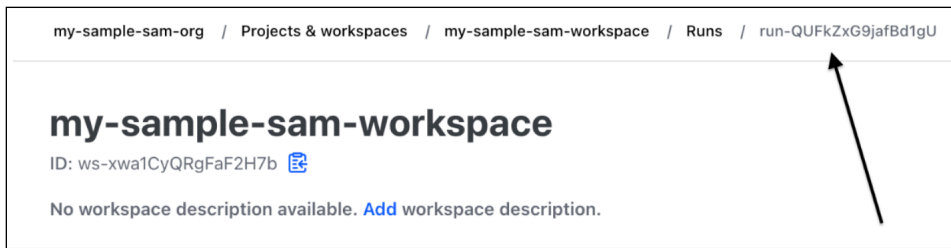
Essas etapas não são necessárias para Terraform v1.6.0 ou são mais recentes. Para começar a usar o AWS SAM CLI com Terraform Cloud, consulte [Uso do AWS SAM CLI com a Terraform](#).

1. Configure um token de API – O tipo de token dependerá do seu nível de acesso. Para saber mais, consulte [Implantações](#) na documentação do Terraform Cloud.
2. Defina sua variável de ambiente de token de API – Veja a seguir um exemplo da linha de comando:

```
$ export TOKEN="<api-token-value>"
```

3. Obtenha seu ID de execução — No Terraform Cloud console, localize o ID de Terraform execução da execução que você gostaria de usar com AWS SAMCLI o.

O ID da execução está localizado no caminho do rastro de navegação da sua execução.



4. Busque o arquivo do plano – Usando seu token de API, obtenha seu arquivo de plano local. Veja a seguir um exemplo da linha de 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
```

Agora você está pronto para usar o AWS SAM CLI com Terraform Cloud. Ao usar um comando AWS SAMCLI compatível, use a opção `--terraform-plan-file` para especificar o nome e o caminho do seu arquivo de plano local. Este é um exemplo:

```
$ sam local invoke --hook-name terraform --terraform-plan-file custom-plan.json
```

Veja a seguir um comando de exemplo que usa a `sam local start-api`:

```
$ sam local start-api --hook-name terraform --terraform-plan-file custom-plan.json
```

Para um aplicativo de amostra que você pode usar com esses exemplos, consulte [api_gateway_v2_tf_cloud](#) no repositório `aws-samples` GitHub.

Próximas etapas

Para começar a usar o AWS SAMCLI com Terraform Cloud, consulte [Usar o AWS SAM CLI por Terraform para depuração e teste locais](#).

Usar o AWS SAM CLI por Terraform para depuração e teste locais

Este tópico aborda como usar AWS Serverless Application Model comandos compatíveis da interface de linha de comando (AWS SAMCLI) com seus Terraform projetos Terraform Cloud e.

Para fornecer feedback e enviar solicitações de atributos, crie um [Problema GitHub](#).

Tópicos

- [Teste local com a do SAM invocação local](#)
- [Teste local com a do SAM API de inicialização local](#)
- [Teste local com a do start-lambda do local do sam](#)
- [Terraform limitações](#)

Teste local com a do SAM invocação local

Note

Para usar a CLI do AWS SAM para testar localmente, você deve ter o Docker instalado e configurado. Para instruções, consulte [Instalando o Docker para usar com o AWS SAM CLI](#).

Veja a seguir um exemplo de como testar sua função de evento Lambda localmente transmitindo um evento:

```
$ sam local invoke --hook-name terraform hello_world_function -e events/event.json -
```

Para saber mais sobre como usar essa função, consulte [Introdução aos testes com SAM invocação local](#).

Teste local com a do SAM API de inicialização local

Para usar o `sam local start-api` com o Terraform, execute o seguinte:

```
$ sam local start-api --hook-name terraform
```

Este é um exemplo:

```
$ 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)
```

Para saber mais sobre esse comando, consulte [Introdução aos testes com SAM API de inicialização local](#).

Funções de eventos Lambda que usam autorizadores Lambda

Para funções Lambda configuradas para usar autorizadores Lambda, elas invocarão AWS SAMCLI automaticamente seu autorizador Lambda antes de invocar seu endpoint de função do Lambda.

- Para saber mais sobre esse recurso no AWS SAMCLI, consulte [Funções de eventos Lambda que usam autorizadores Lambda](#).
- Para obter mais informações sobre o uso de autorizadores Lambda em Terraform, consulte [Resource: aws_api_gateway_authorizer](#) no registro Terraform.

Teste local com a do start-lambda do local do sam

Veja a seguir um exemplo de como testar sua função Lambda localmente com o AWS Command Line Interface (AWS CLI):

1. Use o AWS SAMCLI para criar um ambiente de teste local:

```
$ sam local start-lambda --hook-name terraform hello_world_function
```

2. Use o AWS CLI para invocar sua função 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
```

Para saber mais sobre esse comando, consulte [Introdução aos testes com start-lambda do local do sam](#).

Terraform limitações

As limitações de uso de dados espaciais AWS SAM CLI com Terraform são as seguintes:

- Funções Lambda vinculadas a várias camadas.
- Variáveis locais Terraform que definem vínculos entre recursos.

- Fazendo referência a uma função do Lambda que ainda não foi criada. Isso inclui funções definidas no atributo `body` do recurso da API REST.

Para evitar essas limitações, você pode executar `terraform apply` quando um novo recurso é adicionado.

Usar o AWS SAM CLI com Serverless.tf para depuração e teste locais

A interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) pode ser usada com Serverless.tf módulos para depuração e teste locais de suas funções e camadas de AWS Lambda eventos. Os seguintes comandos AWS SAM CLI são compatíveis:

- `sam build`
- `sam local invoke`
- `sam local start-api`
- `sam local start-lambda`

Note

Serverless.tf a versão 4.6.0 e mais recente suporta integração. AWS SAMCLI

Para começar a usar o AWS SAMCLI com seus Serverless.tf módulos, Serverless.tf atualize para a versão mais recente do AWS SAMCLI e.

A partir da versão 6.0.0 do `serverless.tf`, você deve definir o parâmetro como `create_sam_metadata true`. Isso gera os recursos de metadados que o AWS SAMCLI `sam build` comando exige.

Para saber mais Serverless.tf, consulte o módulo [terraform-aws-lambda](#).

AWS SAM CLI por Terraform referência

Esta seção é a referência para usar a interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) Terraform para depuração e teste locais.

Para fornecer feedback e enviar solicitações de atributos, crie um [Problema GitHub](#).

AWS SAM referência de recursos suportados

A documentação de referência dos atributos AWS SAMCLI compatíveis com o uso Terraform pode ser encontrada aqui:

- [sam build](#)
- [SAM invocação local](#)
- [SAM API de inicialização local](#)
- [start-lambda do local do sam](#)

Terraform referência específica

A documentação de referência específica para uso AWS SAMCLI com Terraform pode ser encontrada aqui:

- [recurso de metadados do sam](#)

recurso de metadados do sam

Essa página contém informações de referência para o tipo de recurso sam metadata resource usado com projetos Terraform.

- Para uma introdução ao uso da interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) com Terraform, consulte [O que é AWS SAM CLI suporte para Terraform?](#).
- Como usar o AWS SAM CLI com Terraform, consulte [Usar o AWS SAM CLI por Terraform para depuração e teste locais](#).

Tópicos

- [Argumentos](#)
- [Exemplos](#)

Argumentos

Argumento	Description
<code>built_output_path</code>	O caminho para os artefatos construídos pela sua Função de AWS Lambda Evento.
<code>docker_build_args</code>	Cadeia decodificada do objeto JSON dos argumentos de construção do Docker. Esse argumento é opcional.
<code>docker_context</code>	O caminho para o diretório que contém o contexto de criação da imagem do Docker.
<code>docker_file</code>	<p>O caminho para o arquivo Docker. Esse caminho é relativo ao caminho <code>docker_context</code> .</p> <p>Esse argumento é opcional. O valor padrão é <code>Dockerfile</code> .</p>
<code>docker_tag</code>	O valor da tag de imagem do Docker criada. Este valor é opcional.
<code>depends_on</code>	O caminho para o recurso de construção para sua função ou camada do Lambda. Para saber mais, consulte O depends_on argumento no registro Terraform.
<code>original_source_code</code>	<p>O caminho até onde sua função do Lambda é definida. Esse valor pode ser uma string, uma matriz de strings ou um objeto JSON decodificado como uma string.</p> <ul style="list-style-type: none">• Para matrizes de strings, somente o primeiro valor é usado, pois não há suporte para vários caminhos de código.• Para objetos JSON, o <code>source_code_property</code> também deve ser definido.
<code>resource_name</code>	O nome de função do Lambda.
<code>resource_type</code>	<p>O formato do seu tipo de pacote de funções do Lambda. Os valores aceitos são:</p> <ul style="list-style-type: none">• <code>IMAGE_LAMBDA_FUNCTION</code>

Argumento	Description
	<ul style="list-style-type: none"> • LAMBDA_LAYER • ZIP_LAMBDA_FUNCTION
source_code_proper ty	O caminho para o código do recurso Lambda no objeto JSON. Defina essa propriedade quando original_source_code for um objeto JSON.

Exemplos

um recurso de metadados referenciando uma função de evento Lambda usando o tipo de pacote 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
  ]
}
```


recurso de metadados sam referenciando uma função de evento Lambda usando o tipo de pacote de imagem

```
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"
  }
}
```

mesmo recurso de metadados referenciando uma camada 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]
}
```

O que é AWS SAM CLI suporte para Terraform?

Use a interface de linha de AWS Serverless Application Model comando (AWS SAMCLI) com seus Terraform projetos ou Terraform Cloud para realizar a depuração e o teste locais de:

- AWS Lambda Funções e camadas de eventos.
- API do Amazon API Gateway.

Para obter uma introdução Terraform, consulte [O que é Terraform?](#) no HashiCorp Terraform site.

Para fornecer feedback e enviar solicitações de atributos, crie um [Problema GitHub](#).

 Note

Como parte da etapa de análise da integração, AWS SAM CLI os comandos AWS SAM CLI do usuário dos processos geram arquivos e dados do projeto. A saída do comando deve permanecer inalterada. Porém, em determinados ambientes, o ambiente ou o executor pode injetar logs ou informações adicionais na saída.

Tópicos

- [O que é o AWS SAM CLI?](#)
- [Como faço para usar o AWS SAM CLI por Terraform?](#)
- [Próximas etapas](#)

O que é o AWS SAM CLI?

AWS SAM CLI é uma ferramenta de linha de comando que você pode usar com AWS SAM modelos e integrações de terceiros compatíveis, como, por exemplo Terraform, para criar e executar seus aplicativos sem servidor. Para obter uma introdução ao AWS SAM CLI, consulte [O que é o AWS SAM CLI?](#).

O AWS SAM CLI suporta os seguintes comandos para Terraform:

- `sam local invoke`— Inicie uma invocação única de um recurso de função localmente. AWS Lambda Para saber mais sobre esse comando, consulte [Introdução aos testes com SAM invocação local](#).
- `sam local start-api` – Execute seus recursos do Lambda localmente e teste por meio de um host de servidor HTTP local. Esse tipo de teste é útil para funções do Lambda invocadas por um endpoint do API Gateway. Para saber mais sobre esse comando, consulte [Introdução aos testes com SAM API de inicialização local](#).
- `sam local start-lambda`— Inicie um endpoint local para sua função Lambda para invocá-la localmente AWS Command Line Interface usando AWS CLI() ou SDKs. Para saber mais sobre esse comando, consulte [Introdução aos testes com start-lambda do local do sam](#).

Como faço para usar o AWS SAM CLI por Terraform?

O [Terraform fluxo de trabalho principal](#) consiste em três estágios: escrever, planejar e aplicar. Com AWS SAM CLI suporte para Terraform, você pode aproveitar o `AWS SAM CLI sam local` conjunto de comandos enquanto continua usando seus Terraform fluxos de trabalho para gerenciar seus aplicativos no AWS. Geralmente, isso significa o seguinte:

- Escreva – Crie sua infraestrutura como código usando Terraform.
- Teste e depure – Use o AWS SAM CLI para testar e depurar seus aplicativos localmente.
- Planejar – visualize as alterações antes de aplicar.
- Aplique – provisione sua infraestrutura.

Para ver um exemplo de uso do AWS SAM CLI with Terraform, consulte [Better together: AWS SAM CLI and HashiCorp Terraform](#) at the AWS Compute Blog.

Próximas etapas

Para concluir todos os pré-requisitos e configurar Terraform, consulte [Conceitos básicos de Terraform suporte para AWS SAM CLI](#).

Como publicar a aplicação com a do AWS SAM CLI

Para disponibilizar seu AWS SAM aplicativo para que outras pessoas encontrem e implantem, você pode usar o AWS SAMCLI para publicá-lo no AWS Serverless Application Repository. Para publicar seu aplicativo usando o AWS SAMCLI, você deve defini-lo usando um AWS SAM modelo. Você também deve tê-lo testado localmente ou na nuvem do AWS .

Siga as instruções neste tópico para criar um novo aplicativo, criar uma nova versão de um aplicativo existente ou atualizar os metadados de um aplicativo existente. (O que você faz depende se o aplicativo já existe no AWS Serverless Application Repository e se algum metadado do aplicativo está mudando.) Para obter mais informações sobre metadados de aplicativo, consulte [AWS SAM propriedades da seção de metadados do modelo](#).

Pré-requisitos

Antes de publicar um aplicativo no AWS Serverless Application Repository usando o AWS SAMCLI, você deve ter o seguinte:

- A AWS SAM(CLI) instalada. Para obter mais informações, consulte [Instale o AWS SAM CLI](#). Para determinar se o AWS SAM CLI está instalado, execute o seguinte comando:

```
sam --version
```

- Um AWS SAM modelo válido.
- O código e as dependências do seu aplicativo aos quais o AWS SAM modelo faz referência.
- Uma versão semântica, necessária apenas para compartilhar seu aplicativo publicamente. Esse valor pode ser tão simples quanto 1,0.
- Um URL que aponta para o código-fonte do seu aplicativo.
- Um arquivo README.md. Este arquivo deve descrever como os clientes podem usar seu aplicativo e como configurá-lo antes de implantá-lo em suas próprias contas AWS .
- Um arquivo LICENSE.txt necessário apenas para compartilhar seu aplicativo publicamente.
- Se seu aplicativo contiver algum aplicativo aninhado, você já deve tê-lo publicado no AWS Serverless Application Repository.
- Uma política de bucket válida do Amazon Simple Storage Service (Amazon S3) que concede ao serviço permissões de leitura para artefatos que você carrega no Amazon S3 ao empacotar seu aplicativo. Para configurar esta política, faça o seguinte:

1. Abra o console do Amazon S3 em <https://console.aws.amazon.com/s3/>.
2. Escolha o nome do bucket do Amazon S3 usado para empacotar seu aplicativo.
3. Escolha Permissions (Permissões).
4. Na guia Permissions (Permissões), escolha Bucket policy (Política de bucket), Edit (Editar).
5. Na página Editar política de bucket, cole a seguinte declaração de política no editor de políticas. Na declaração de política, certifique-se de usar o nome do seu bucket no Resource elemento e o AWS ID da sua conta no Condition elemento. A expressão no Condition elemento garante que ele AWS Serverless Application Repository tenha permissão para acessar somente aplicativos da AWS conta especificada. Para obter mais informações sobre declarações de política, [consulte Referência de elementos de política JSON do IAM](#) no Guia do usuário do IAM.

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. Escolha Save changes (Salvar alterações).

Publicar um novo aplicativo

Etapa 1: Adicionar uma seção de **metadados** ao AWS SAM modelo

Primeiro, adicione uma Metadata seção ao seu AWS SAM modelo. Forneça as informações do aplicativo a serem publicadas no AWS Serverless Application Repository.

A seguir está uma seção de exemplo Metadata:

```
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
      ...
```

Para obter mais informações sobre a Metadata seção do AWS SAM modelo, consulte [AWS SAM propriedades da seção de metadados do modelo](#).

Etapa 2: Empacotar o aplicativo

Execute o seguinte AWS SAMCLI comando, que faz upload dos artefatos do aplicativo para o Amazon S3 e gera um novo arquivo de modelo chamado `packaged.yaml`:

```
sam package --output-template-file packaged.yaml --s3-bucket <your-bucket-name>
```

Você usa o `packaged.yaml` arquivo de modelo na próxima etapa para publicar o aplicativo no arquivo AWS Serverless Application Repository. Esse arquivo é semelhante ao arquivo de modelo original (`template.yaml`), mas tem uma diferença importante: as propriedades `CodeUri`, `LicenseUrl`, e `ReadmeUrl` apontam para o bucket do Amazon S3 e os objetos que contêm os respectivos artefatos.

O seguinte trecho de um arquivo de modelo de exemplo `packaged.yaml` mostra a propriedade `CodeUri`:

```
MySampleFunction:
  Type: AWS::Serverless::Function
  Properties:
    CodeUri: s3://bucketname/fbd77a3647a4f47a352fc0bjectGUID

...
```

Etapa 3: publicar o aplicativo

Para publicar uma versão privada do seu AWS SAM aplicativo no AWS Serverless Application Repository, execute o seguinte AWS SAMCLI comando:

```
sam publish --template packaged.yaml --region us-east-1
```

A saída do comando `sam publish` inclui um link para seu aplicativo no AWS Serverless Application Repository. Você também pode ir diretamente para a [AWS Serverless Application Repository página de destino](#) e pesquisar seu aplicativo.

Etapa 4: compartilhar o aplicativo (opcional)

Por padrão, seu aplicativo está definido como privado, portanto, não é visível para outras contas da AWS. Para compartilhar seu aplicativo com outras pessoas, você deve torná-lo público ou conceder permissão para uma lista específica de AWS contas.

Para obter informações sobre como compartilhar seu aplicativo usando o AWS CLI, consulte [Exemplos AWS Serverless Application Repository Resource-Based de políticas](#) no Guia do AWS Serverless Application Repository desenvolvedor. Para obter informações sobre como compartilhar seu aplicativo usando o Console de gerenciamento da AWS, consulte [Compartilhamento de um aplicativo](#) no AWS Serverless Application Repository Guia do desenvolvedor.

Publicar uma nova versão de um aplicativo existente

Depois de publicar um aplicativo no AWS Serverless Application Repository, talvez você queira publicar uma nova versão dele. Por exemplo, você pode ter alterado o código da Lambda Event Function ou adicionado um novo componente à arquitetura do seu aplicativo.

Para atualizar um aplicativo que você publicou anteriormente, publique-o novamente usando o mesmo processo detalhado anteriormente. Na seção Metadata do arquivo de modelo AWS SAM, forneça o mesmo nome do aplicativo com o qual você o publicou originalmente, mas inclua um novo valor `SemanticVersion`.

Por exemplo, considere um aplicativo publicado com o nome de `SampleApp` e um `SemanticVersion` de `1.0.0`. Para atualizar esse aplicativo, o modelo AWS SAM deve ter o nome do aplicativo `SampleApp` e um `SemanticVersion` de `1.0.1` (ou qualquer outra coisa que não seja `1.0.0`).

Tópicos adicionais

- [AWS SAM propriedades da seção de metadados do modelo](#)

AWS SAM propriedades da seção de metadados do modelo

`AWS::ServerlessRepo::Application` é uma chave de metadados que você pode usar para especificar as informações do aplicativo que você deseja publicar no AWS Serverless Application Repository.

Note

CloudFormation [funções intrínsecas](#) não são suportadas pela chave de `AWS::ServerlessRepo::Application` metadados.

Propriedades

Essa tabela fornece informações sobre as propriedades da Metadata seção do AWS SAM modelo. Esta seção é necessária para publicar aplicativos no AWS Serverless Application Repository usando AWS SAMCLI o.

Propriedade	Tipo	Obrigatório	Descrição
Name	String	TRUE	<p>O nome da aplicação.</p> <p>Comprimento mínimo = 1. Tamanho máximo = 140.</p> <p>Padrão: "[a-zA-Z0-9\\-]+";</p>
Description	String	TRUE	<p>A descrição do aplicativo.</p> <p>Comprimento mínimo = 1. Tamanho máximo = 256.</p>
Author	String	TRUE	<p>O nome do autor que publica o aplicativo.</p> <p>Comprimento mínimo = 1. Tamanho máximo = 127.</p> <p>Padrão: "^[a-z0-9]([a-z0-9] -(?!-))*[a-z0-9]?\$";</p>
SpdxLicenseId	String	FALSE	<p>Um identificador de licença válido. Para ver a lista de identificadores de licença válidos, consulte a Lista de licenças SPDX no site Software Package Data Exchange (SPDX).</p>
LicenseUrl	String	FALSE	<p>A referência a um arquivo de licença local, ou um link do Amazon S3 para um arquivo de licença, que corresponde ao valor SPDXLicenseId do seu aplicativo.</p> <p>Um arquivo AWS SAM de modelo que não tenha sido empacotado usando o <code>sam package</code> comando pode ter uma referência a um arquivo local para essa propriedade. No entanto, para que um aplicativo seja publicado usando o comando <code>sam publish</code>, essa propriedade deve ser uma referência a um bucket do Amazon S3.</p> <p>Tamanho máximo = 5 MB.</p>

Propriedade	Tipo	Obrigatório	Descrição
			<p>Forneça um valor para esta propriedade para tornar o aplicativo público. Observe que você não pode atualizar essa propriedade após a publicação do aplicativo. Portanto, para adicionar uma licença a um aplicativo, você deve excluí-la primeiro ou publicar um novo aplicativo com um nome diferente.</p>
ReadmeUrl	String	FALSE	<p>A referência a um arquivo readme local ou a um link do Amazon S3 para o arquivo readme que contém uma descrição mais detalhada do aplicativo e de como ele funciona.</p> <p>Um arquivo AWS SAM de modelo que não tenha sido empacotado usando o <code>sam package</code> comando pode ter uma referência a um arquivo local para essa propriedade. No entanto, para ser publicada usando o comando <code>sam publish</code>, essa propriedade deve ser uma referência a um bucket do Amazon S3.</p> <p>Tamanho máximo = 5 MB.</p>
Labels	String	FALSE	<p>Os rótulos que melhoram a descoberta de aplicativos em resultados de pesquisa.</p> <p>Comprimento mínimo = 1. Tamanho máximo = 127. Número máximo de rótulos: 10.</p> <p>Padrão: <code>"^[a-zA-Z0-9+\\-_:\\/@]+\$"</code>;</p>
HomePageUrl	String	FALSE	<p>Uma URL com mais informações sobre o aplicativo — por exemplo, a localização do seu GitHub repositório para o aplicativo.</p>

Propriedade	Tipo	Obrigatório	Descrição
SemanticVersion	String	FALSE	<p>A versão semântica do aplicativo. Para a especificação de controle de versão semântica, consulte o site Versão Semântica.</p> <p>Forneça um valor para esta propriedade para tornar o aplicativo público.</p>
SourceCodeUrl	String	FALSE	Um link para um repositório público para o código-fonte do aplicativo.

Casos de uso

Esta seção lista os casos de uso para aplicativos de publicação, junto com as propriedades Metadata que são processadas para esse caso de uso. As propriedades que não estão listadas para um determinado caso de uso são ignoradas.

- Criação de um novo aplicativo — Um novo aplicativo é criado se não houver nenhum aplicativo AWS Serverless Application Repository com um nome correspondente para uma conta.
 - Name
 - SpdxLicenseId
 - LicenseUrl
 - Description
 - Author
 - ReadmeUrl
 - Labels
 - HomePageUrl
 - SourceCodeUrl
 - SemanticVersion
 - O conteúdo do AWS SAM modelo (por exemplo, qualquer fonte de evento, recursos e código da Lambda Event Function)

- Criação de uma versão do aplicativo — Uma versão do aplicativo é criada se já houver um aplicativo no AWS Serverless Application Repository com um nome correspondente para uma conta e ele `SemanticVersion` estiver mudando.
 - `Description`
 - `Author`
 - `ReadmeUrl`
 - `Labels`
 - `HomePageUrl`
 - `SourceCodeUrl`
 - `SemanticVersion`
 - O conteúdo do AWS SAM modelo (por exemplo, qualquer fonte de evento, recursos e código da função Lambda)
- Atualização de um aplicativo — Um aplicativo é atualizado se já houver um aplicativo no AWS Serverless Application Repository com um nome correspondente para uma conta e ele não `SemanticVersion` estiver mudando.
 - `Description`
 - `Author`
 - `ReadmeUrl`
 - `Labels`
 - `HomePageUrl`

Exemplo

A seguir, temos um exemplo de uma seção Metadata:

```
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*

Histórico do documento para AWS SAM

A tabela a seguir descreve as mudanças importantes em cada versão do Guia do desenvolvedor do AWS Serverless Application Model . Para receber notificações sobre atualizações dessa documentação, assine um feed RSS.

- Última atualização da documentação: 9 de outubro de 2026

Alteração	Descrição	Data
Adicionou o AWS::Serverless::WebFunction recurso	Use AWS::Serverless::WebFunction para definir uma Lambda Web Function, que atende solicitações HTTP por meio de um endpoint gerenciado. Para saber mais, consulte AWS::Serverless::WebFunction .	12 de outubro de 2026
Tipos de autenticação adicionados OAuth 2.0 e do IAM para fontes de SelfManagedKafka eventos	A SourceAccessConfigurations propriedade de uma fonte de SelfManagedKafka eventos agora aceita os mecanismos de IAM_OAUTHBEARER_AUTH authentication OAuthBEARER_AUTH IAM_AUTH,, e, junto com os tipos OAuthBEARER_SCOPE OAuthBEARER_AUDIENCE ,OAuthBEARER_LOGICAL_CLUSTER , e OAuthBEARER_IDENTITY_POOL metadados. Para saber mais, consulte SelfManagedKafka .	8 de outubro de 2026

[Adicionou a ManagedResourceTags propriedade para AWS::Serverless::CapacityProvider](#)

Aplique tags explícitas às instâncias do Amazon Elastic Compute Cloud e a outros recursos gerenciados que um provedor de capacidade e provisiona em seu nome. Para saber mais, consulte [ManagedResourceTags](#).

25 de agosto de 2026

[Adicionou a LoggingConfiguration propriedade para AWS::Serverless::CapacityProvider](#)

Configure o registro do sistema para um provedor de capacidade, para que a atividade de escalabilidade e os eventos operacionais cheguem a um grupo de CloudWatch registros da Amazon em um nível de log configurável. Para saber mais, consulte [AWS::Serverless::CapacityProvider](#).

24 de agosto de 2026

[Foi adicionada a opção --output para comandos AWS SAMCLI](#)

Especifique --output json para imprimir uma saída estruturada e legível por máquina que você possa consumir programaticamente, como em CI/CD pipelines e extensões de IDE. A opção está disponível para [sam build](#), [sam deploy](#) e [sam init](#).

19 de agosto de 2026

[Foi adicionada a StorageMo de propriedade para o código da função e o conteúdo da camada](#)

Controle se o Lambda armazena uma cópia do seu pacote de implantação ou se faz referência a ela a partir do seu bucket do Amazon S3. Para saber mais, consulte [FunctionCode](#) e [LayerContent](#).

4 de agosto de 2026

[AWS SAMCLISuporte atualizado para Rust funções de construção para disponibilidade geral](#)

A criação de Rust funções Cargo Lambda não exige mais a opção por recursos beta. A orientação para projetos que contêm várias funções agora usa um Cargo espaço de trabalho. Para saber mais, consulte Como [criar funções do Rust Lambda](#) com. Cargo Lambda

3 de agosto de 2026

[Adicionado AWS SAMCLI suporte para Java 8, Java 11 e Java 17 em AL2023](#)

java8.al2023 ,java11.al2023 , e agora java17.al2023 estão disponíveis como tempos de execução e repositórios de imagens. Consulte [Repositórios de imagem](#) e [sam init](#).

13 de julho de 2026

[Foram adicionados os tipos `AWS::Serverless::NetworkConnector` de recursos `AWS::Serverless::MicrovmImage` e](#)

Novas seções de referência descrevem como criar uma imagem de microVM Lambda a partir de um artefato de código armazenado no Amazon S3 e como definir a conectividade de rede para recursos computacionais do Lambda. Para saber mais, consulte [AWS::Serverless::MicrovmImage](#) e [AWS::Serverless::NetworkConnector](#).

6 de julho de 2026

[Foi adicionada a opção `--express` para os comandos `sam deploy` e `sam sync`](#)

Use o modo CloudFormation Express para que as operações de pilha sejam concluídas quando a configuração do recurso for aplicada, enquanto os recursos continuam sendo preparados em segundo plano. Para saber mais, consulte [sam deploy](#) e [sam sync](#).

1 de julho de 2026

[Foi adicionada a opção `--no-watch` para os comandos `AWS SAMCLI sam local start-api` e `sam local start-lambda`](#)

Desative a observação de arquivos ao usar `--warm-containers`, para projetos em que a observação causa alta CPU ou I/O sobrecarga. Para saber mais, consulte [sam local start-api](#) e [sam local start-lambda](#).

10 de junho de 2026

[AWS SAMCLISuporte adicional para a AWS::LanguageExtensions transformação](#)

O AWS SAMCLI agora se expande `Fn::ForEach` `Fn::Length` `Fn::ToJsonString` `Fn::FindInMap` com `DefaultValue` `localment` `esam build`, então `esam package`, `esam deploy`, `esam sync`, `esam validate`, e os `esam local` comandos funcionam com modelos que usam a `AWS::LanguageExtensions` transformação. Para saber mais, consulte o [suporte a extensões de CloudFormation idioma](#).

13 de maio de 2026

[Adicionado AWS SAMCLI suporte para Ruby 4.0](#)

RubyA versão 4.0 agora está disponível como um repositório de imagens e de tempo de execução. Consulte [Repositórios de imagem](#) e [sam init](#).

27 de abril de 2026

[Adicionado o tipo AWS::Serverless::WebSocketApi de recurso](#)

Defina um WebSocket API com AWS SAM, incluindo rota, autorização, domínio personalizado e configuração do Route 53. Para saber mais, consulte [AWS::Serverless::WebSocketApi](#).

13 de abril de 2026

[Expandiu a opção `--invoke-image` para os comandos locais AWS SAMCLI](#)

Especifique `--invoke-image` mais de uma vez para usar uma imagem de contêiner diferente por recurso e `ImageUri` substitua as funções que usam `PackageType: Image`. Para saber mais, consulte [sam local invoke](#).

10 de abril de 2026

[Foram adicionadas as `EndpointAccessMode` propriedades `SecurityPolicy` e para `AWS::Serverless::Api`](#)

Defina a política de TLS segurança e o modo de acesso ao endpoint em um domínio personalizado. Para saber mais, consulte [DomainConfiguration](#).

10 de abril de 2026

[Foi adicionada a propriedade de `DockerImage` construção para compilações de imagens de contêineres](#)

Especifique um nome personalizado para a Docker imagem AWS SAMCLI criada. Para saber mais, consulte [Criando seu aplicativo](#).

10 de abril de 2026

[Foi adicionada a opção `--container-dns` para os comandos locais AWS SAMCLI](#)

Especifique os DNS servidores que os contêineres locais usam. A opção está disponível para [sam local invoke](#), [sam local start-api](#) e [sam local start-lambda](#).

10 de abril de 2026

Foi adicionado suporte ao sistema de arquivos Amazon S3 Files para AWS::Serverless::Function	Anexe um ponto de acesso do Amazon S3 Files a uma função com <code>FileSystemConfigs</code> , além de um ponto de acesso do Amazon EFS. Para saber mais, consulte AWS::Serverless::Function .	8 de abril de 2026
Adicionado suporte da CLI do AWS SAM para Docker BuildKit	Crie funções de imagem de contêiner com <code>BuildKit</code> . Para saber mais, consulte Usando o sam build .	10 de março de 2026
Suporte de .env arquivo adicionado para a opção --env-vars	A <code>--env-vars</code> opção dos <code>sam local</code> comandos agora aceita um <code>.env</code> arquivo além do JSON. Para saber mais, consulte Como invocar funções localmente .	6 de março de 2026
Adicionou a <code>IpAddressType</code> propriedade para <code>AWS::Serverless::Api</code>	Configure um endpoint de API como <code>ipv4</code> ou <code>dualstack</code> . Para saber mais, consulte EndpointConfiguration .	20 de fevereiro de 2026
Adicionado AWS SAM CLI suporte para o gerenciador de <code>uv</code> pacotes	Crie Python funções com <code>uv</code> . Para saber mais, consulte Criação de Python funções com <code>uv</code> .	19 de fevereiro de 2026
Adicionou a <code>ResponseTransferMode</code> propriedade para eventos	Transmita uma resposta de função configurando <code>ResponseTransferMode</code> para <code>RESPONSE_STREAM</code> . Para saber mais, consulte Api .	11 de fevereiro de 2026

[Foram adicionadas as MetricsConfig propriedades LoggingConfig e os eventos do Amazon MSK](#)

Configure o registro de mapeamento da fonte de eventos e métricas aprimoradas para uma fonte de eventos do Amazon MSK. Para saber mais, consulte [MSK](#).

2 de fevereiro de 2026

[Adicionou a TenancyConfig propriedade para AWS::Serverless::Function](#)

Configure o isolamento do inquilino para uma função e passe a um inquilino com a nova `--tenant-id` opção ativada e `local invoke` `remote invoke` Para saber mais, consulte [AWS::Serverless::Function](#).

20 de novembro de 2025

[Foi adicionada a ProvisionedPollerConfig propriedade para eventos do Amazon SQS](#)

Use o modo provisionado para uma fonte de eventos do Amazon SQS definindo um número mínimo e máximo de pollers. Para saber mais, consulte [SQS](#).

20 de novembro de 2025

[Foram adicionadas propriedades de repetição e lote para eventos SelfManagedKafka](#)

A fonte de eventos autogerenciada do Kinesis agora `BisectBatchOnFunctionError` oferece suporte a `FunctionResponseType`, `MaximumRecordAgeInSeconds`, e `MaximumRetryAttempts`. Para saber mais, consulte [SelfManagedKafka](#).

20 de novembro de 2025

[AWS SAMCLISuporte adicionado para Node.js 24](#)

Node.js24 agora está disponível como repositório de imagens e tempo de execução. Consulte [Repositórios de imagem](#) e [sam init](#).

19 de novembro de 2025

[Foram adicionadas propriedades de repetição e lote para eventos do Amazon MSK](#)

A fonte de eventos do Amazon MSK agora oferece suporte a `BisectBatchOnFunctionError` `FunctionResponseTy` `pes` `MaximumRecordAgeInSeconds` , e `MaximumRetryAttempts` Para saber mais, consulte [MSK](#).

19 de novembro de 2025

[Adicionado o tipo AWS::Serverless::CapacityProvider de recurso](#)

Execute funções na capacidade do Amazon Elastic Compute Cloud que você possui e associe um provedor de capacidade a uma função com `CapacityProviderConfig` . Para saber mais, consulte [AWS::Serverless::CapacityProvider](#).

17 de novembro de 2025

[Adicionado AWS SAMCLI suporte para Java 25 e Python 3.14](#)

Java25 e Python 3.14 agora estão disponíveis como repositórios de imagens e imagens `sam init` base. Consulte [Repositórios de imagem](#) e [sam init](#).

10 de novembro de 2025

Foi adicionada a <code>Enabled</code> propriedade para eventos do Amazon MSK	Pausar e retomar um mapeamento de origem de eventos do Amazon MSK. Para saber mais, consulte MSK .	16 de outubro de 2025
Propriedades de construção adicionadas que preservam a estrutura Python do pacote	Especifique <code>ParentPackageMode</code> e <code>ParentPackages</code> abaixo <code>BuildProperties</code> para que sua função Lambda criada possa usar importações absolutas. Para saber mais, consulte Criando seu aplicativo .	6 de outubro de 2025
AWS SAM Suporte adicional para funções duráveis	Configure uma função durável com a <code>DurableConfig</code> propriedade e teste uma localmente com o novo <code>local callback</code> comando. Para saber mais, consulte DurableConfig Testando funções duráveis .	25 de setembro de 2025
Adicionado suporte da CLI do AWS SAM para Finch	Use Finch em vez de Docker para criar <code>--use-container</code> e testar seu aplicativo localmente. Para saber mais, consulte Instalando Finch .	19 de agosto de 2025
Foi adicionada a <code>BatchSize</code> propriedade para eventos do Amazon MSK	Defina o número máximo de registros que o Lambda recupera em um único lote de uma fonte de eventos do Amazon MSK. Para saber mais, consulte MSK .	5 de agosto de 2025

Orientação adicional para converter uma função do console Lambda em AWS SAM	Mova uma função que você criou no console Lambda para um AWS SAM modelo para que você possa gerenciá-la como infraestrutura e código. Para saber mais, consulte Conversão de funções AWS SAM do Lambda em aplicativos.	5 de agosto de 2025
Foi adicionada a SchemaRegistryConfig propriedade para eventos do Kinesis	Use um registro de esquema com uma fonte de eventos do Amazon MSK ou do Kinesis autogerenciada. Para saber mais, consulte MSK e SelfManagedKafka .	19 de junho de 2025
Foi adicionado AWS SAMCLI suporte para o.NET 9	O.NET 9 agora está disponível como um repositório de imagens. Consulte Repositórios de imagens .	4 de junho de 2025
Foi adicionado AWS SAM suporte para nomes de domínio personalizados em APIs privadas	Anexe um domínio personalizado a uma API REST privada com o novo Policy e DomainAccessAssociation as propriedades. Para saber mais, consulte DomainConfiguration e DomainAccessAssociation .	04 de abril de 2025
Adicionado AWS SAMCLI suporte para Ruby 3.4	RubyO 3.4 agora está disponível como um repositório de imagens e tempo de execução. Consulte Repositórios de imagem e sam init .	24 de março de 2025

Foi adicionada a opção --runtime para o comando AWS SAMCLI sam local invoke	Substitua o tempo de execução definido em seu modelo para testar uma função em um tempo de execução diferente. Para saber mais, consulte sam local invoke .	11 de março de 2025
Foram adicionadas as opções --mount-symlinks e --no-memory-limit para comandos AWS SAMCLI	Monte links simbólicos de nível superior no contêiner de compilação com --mount-symlinks on sam build e sam local invoke e remova o limite de memória do contêiner com os --no-memory-limit comandos. <code>sam local</code>	26 de fevereiro de 2025
Foi adicionada a opção --no-use-container para os comandos AWS SAMCLI sam build e sam sync	Crie sua função na máquina local em vez de dentro de um contêiner. Para saber mais, consulte sam build e sam sync .	9 de dezembro de 2024
Adicionado AWS SAMCLI suporte para Node.js 22 e Python 3.13	Node.js22 e Python 3.13 agora estão disponíveis como tempos de execução e repositórios de imagens. Consulte Repositórios de imagem e sam init .	5 de dezembro de 2024
Adicionou a PublishLambdaVersion propriedade de para AWS::Serverless::LayerVersion	Controle se AWS SAM publica uma nova versão da camada. Para saber mais, consulte AWS::Serverless::LayerVersion .	5 de dezembro de 2024

[Foram adicionadas as MetricsConfig propriedade des ProvisionedPollerConfig e para mapeamentos de origem de eventos](#)

Use o modo provisionado para o Amazon MSK e as fontes de eventos autogerenciadas do Kinesis com ProvisionedPollerConfig, e opte por métricas aprimoradas de mapeamento de fontes de eventos para fontes de eventos do DynamoDB, Kinesis e Amazon SQS com MetricsConfig. Para saber mais, consulte [MSK](#) e [SQS](#).

25 de novembro de 2024

[Adicionou a SourceKMS KeyArn propriedade para AWS::Serverless::Function](#)

Especifique a AWS KMS chave AWS SAM usada para criptografar seu pacote .zip de implantação. Para saber mais, consulte [AWS::Serverless::Function](#).

14 de novembro de 2024

[Adicionou a Recursive Loop propriedade para AWS::Serverless::Function](#)

Defina se o Lambda termina ou permite um loop de invocação de função recursiva. Para saber mais, consulte [AWS::Serverless::Function](#).

27 de agosto de 2024

[Foi adicionada a KmsKeyArn propriedade para mapeamentos de origem de eventos](#)

Especifique uma AWS KMS chave que criptografe seus critérios de filtro de mapeamento da fonte de eventos em repouso. A propriedade está disponível nas fontes de eventos do Amazon DocumentDB, DynamoDB, Kinesis, Amazon MQ, Amazon MSK e Amazon SQS. SelfManagedKafka Para saber mais, consulte [MSK](#).

21 de agosto de 2024

[Adicionou a UseAliasAsEventTarget propriedade de para AWS::Serverless::StateMachine](#)

Use o alias criado por AutoPublishAlias como destino da origem do evento. Para saber mais, consulte [AWS::Serverless::StateMachine](#).

12 de agosto de 2024

[Reestruturado e atualizado o conteúdo em todo o guia do desenvolvedor](#)

Reorganizado e reestruturado o guia para melhorar a capacidade de descoberta e a usabilidade. Títulos atualizados e aprimorados. Fornecidos detalhes adicionais na apresentação de tópicos e conceitos.

20 de junho de 2024

[Adicionado suporte da CLI do AWS SAM para o Ruby 3.3](#)

O Ruby 3.3 agora está disponível como repositório de runtime e de imagens. Consulte [Repositórios de imagem](#) e [sam init](#) para obter detalhes.

4 de abril de 2024

[Adicionadas opções de comando da CLI do AWS SAM](#)

Estão disponíveis novas opções para o comando [sam local start-api](#): `--ssl-cert-file PATH` e `--ssl-key-file PATH`. Além disso, a nova opção de linha de comando `--add-host LIST` está disponível para [sam local invoke](#), [sam local start-api](#) e [sam local start-lambda](#)

20 de março de 2024

[Adicionado suporte da CLI do AWS SAM para o .NET 8](#)

O .NET 8 agora está disponível como repositório de runtime e de imagens. Os tempos de execução e os repositórios de imagens do .NET Core 3.1, Node.js 14, Node.js 12, Python 3.7 e Ruby 2.7 não são mais suportados. Consulte [Repositórios de imagem](#) e [sam init](#).

22 de fevereiro de 2024

[Adicionado o instalador do pacote arm64 da CLI do AWS SAM para Linux](#)

Para obter instruções, consulte [Instalando AWS SAMCLI](#) o.

6 de dezembro de 2023

[Adicionada a opção `--watch-exclude` para o comando `sam sync` da CLI do AWS SAM](#)

Exclua arquivos e pastas do início de uma sincronização. Para saber mais, consulte [Especificar arquivos e pastas que não iniciarão uma sincronização](#).

6 de dezembro de 2023

[Adicionada a opção --build-in-source para o comando sam sync da CLI do AWS SAM](#)

Crie seu projeto na pasta de origem para acelerar o processo de compilação. Para saber mais, consulte [Acelerar os tempos de compilação criando seu projeto na pasta de origem.](#)

6 de dezembro de 2023

[Adicionada a opção --build-in-source para o comando sam build da CLI do AWS SAM](#)

Crie seu projeto na pasta de origem para acelerar o processo de compilação. Para saber mais, consulte [Acelerar os tempos de compilação criando seu projeto na pasta de origem.](#)

6 de dezembro de 2023

[Adicionado novo suporte a recursos para o comando de invocação remota da CLI do AWS SAM](#)

Use `sam remote invoke` com aplicativos Kinesis Data Streams, filas do Amazon SQS e máquinas de estado Step Functions. Para saber mais, consulte [Usar a invocação remota do sam.](#)

15 de novembro de 2023

[Adicionado um novo comando remoto test-event da CLI do AWS SAM para eventos de teste compartilháveis](#)

Use o AWS SAM CLI para acessar e gerenciar eventos de teste compartilháveis do registro do EventBridge esquema para testar suas funções do Lambda no. Nuvem AWS Para saber mais, consulte [Usando sam remote test-event.](#)

3 de outubro de 2023

O suporte ao AWS SAM CLI para Terraform já está disponível para o público em geral	Para saber mais sobre o suporte a AWS SAMCLI para Terraform, consulte AWS SAM CLITerraformsuporte .	5 de setembro de 2023
Adicionado suporte da CLI do AWS SAM para Terraform Cloud	O AWS SAM CLI agora oferece suporte a testes locais para Terraform Cloud. Para saber mais, consulte Configurar para Terraform Cloud .	5 de setembro de 2023
Adicionado suporte ao formato de arquivo YAML para o arquivo de configuração da CLI do AWS SAM	O AWS SAMCLI agora suporta o formato de arquivo [.yaml .yml]. As páginas Configurar o AWS SAM CLI e Arquivo de configuração do AWS SAM CLI foram atualizadas.	18 de julho de 2023
Suporte de AWS SAMCLIsam local start-api comando adicionado para Terraform	O Para que serve o AWS SAMCLI suporteTerraform? A seção foi atualizada para incluir suporte de AWS SAMCLI sam local start-api comando paraTerraform.	6 de julho de 2023
Adicionado novo comando de invocação remota da CLI do AWS SAM	Para começar a usar sam remote invoke, consulte Usar a invocação remota do sam .	22 de junho de 2023
Adicionado tipo de recurso sem servidor da GraphQL API do AWS AppSync	Crie uma nova AWS::Serverless::GraphQLApi seção que descreva como definir um GraphQL API recurso com AWS SAM.	22 de junho de 2023

Adicionado suporte à CLI do AWS SAM para o Ruby 3.2	Atualize a página do sam init para incluir novos valores básicos de imagem e runtime. Atualize a página Repositórios de imagens com o Ruby 3.2 Amazon ECR URI.	6 de junho de 2023
Adicionadas etapas opcionais para verificação da integridade do instalador do pacote da CLI do AWS SAM	Atualize a página Instalação o do AWS SAM CLI para refletir a etapa opcional. Crie a página Verificar a integridade do instalador do AWS SAM CLI para documentar as etapas.	31 de maio de 2023
Adicionada a opção sam sync para ignorar a sincronização da infraestrutura	Personalize se uma CloudFormation implantação é necessária sempre que sam sync for executada. Para saber mais, consulte Ignorar a CloudFormation implantação inicial .	23 de março de 2023
Adicionado suporte para o tipo de origem de eventos DocumentDB	A especificação do AWS SAM modelo agora oferece suporte ao tipo de fonte de DocumentDB evento para o <code>AWS::Serverless::Function</code> recurso. Para saber mais, consulte DocumentDB .	10 de março de 2023

[Construir funções do Rust Lambda com Cargo Lambda](#)

Use o AWS SAM CLI para criar suas funções do Rust Lambda usando Cargo Lambda. Para saber mais, consulte [Construindo as funções do Rust Lambda com Cargo Lambda](#).

23 de fevereiro de 2023

[Crie recursos funcionais fora do AWS SAM](#)

Foi adicionada orientação sobre como ignorar funções ao usar o comando `sam build`. Para saber mais, consulte [Criação de funções fora do AWS SAM](#).

14 de fevereiro de 2023

[Nova sintaxe de conectores incorporados](#)

Use a nova sintaxe de conectores incorporados para definir seus recursos `AWS::Serverless::Connector` . Para saber mais, consulte [Gerenciamento de permissões de recursos com AWS SAM conectores](#).

8 de fevereiro de 2023

[Foi adicionado um novo comando `sam list` para AWS SAMCLI](#)

Use `sam list` para visualizar informações importantes sobre os recursos em seu aplicativo sem servidor. Para saber mais, consulte [sam list](#).

2 de fevereiro de 2023

[Propriedades de formato e OutExtension construção adicionadas para esbuild](#)

A criação de funções Node.js Lambda com o esbuild agora oferece suporte Format e OutExtension cria propriedades. Para saber mais, consulte Como [criar funções do Node.js Lambda com](#) o esbuild.

2 de fevereiro de 2023

[Opções de gerenciamento de tempo de execução adicionadas à especificação do AWS SAM modelo](#)

Configure as opções de gerenciamento de runtime para suas funções do Lambda. Para saber mais, consulte [RuntimeManagementConfig](#).

24 de janeiro de 2023

[Propriedade de destino adicionada ao StateMachine recurso EventSource for AWS::Serverless::](#)

O tipo de recurso AWS::Serverless::StateMachine suporta a propriedade Target para [EventBridgeRule](#) e fontes de eventos [Schedule](#).

13 de janeiro de 2023

[Configurar o escalonamento de pollers SQS para funções do Lambda](#)

Configure o escalonamento de pollers SQS com a propriedade ScalingConfig para AWS::Serverless::Function. Para saber mais, consulte [ScalingConfig](#).

12 de janeiro de 2023

[Valide AWS SAM aplicativos com cfn-lint](#)

Você pode usar o cfn-lint para validar seus AWS SAM modelos por meio do. AWS SAMCLI Para saber mais, consulte [Validar com cfn-lint](#).

11 de janeiro de 2023

Monitore seus aplicativos sem servidor com CloudWatch o Application Insights	Configure o Amazon CloudWatch Application Insights para monitorar seus AWS SAM aplicativos. Para saber mais, consulte Monitore seus aplicativos sem servidor com o CloudWatch Application Insights .	19 de dezembro de 2022
Adicionado instalador de pacotes do AWS SAM CLI para macOS	Instale o AWS SAM CLI usando o novo instalador de pacotes para macOS. Para saber mais, consulte Instalando o AWS SAM CLI .	6 de dezembro de 2022
Suporte adicional para Lambda SnapStart	Configure SnapStart suas funções do Lambda para criar instantâneos, que são estados em cache de suas funções inicializadas. Para saber mais, consulte AWS::Serverless::Function .	28 de novembro de 2022
Adicionado suporte AWS SAM CLI para nodejs18.x	O AWS SAM CLI agora oferece suporte ao runtime do nodejs18.x. Para saber mais, consulte sam init .	17 de novembro de 2022
Adicionadas orientações sobre a configuração de acesso e permissões	AWS SAM fornece duas opções que simplificam o gerenciamento de acesso e permissões para seus aplicativos sem servidor. Para saber mais, consulte Gerenciamento de permissões e acessos a recursos .	17 de novembro de 2022

[Adicionado suporte para criar funções do Lambda .NET 7 com compilação AOT nativa](#)

Crie e empacote suas funções do.NET 7 Lambda com AWS SAM, utilizando a compilação nativa Ahead-of-Time (AOT) para melhorar os tempos de inicialização a frio do Lambda. Para saber mais, consulte [Criação de funções do Lambda .NET 7 com compilação AOT nativa](#).

15 de novembro de 2022

[AWS SAMCLITerraformSuporte adicional para depuração e teste locais](#)

Use o AWS SAMCLI em seus projetos Terraform para realizar depuração e testes locais de suas funções e camadas do Lambda. Para saber mais, consulte o suporte de [AWS SAM CLI Terraform](#).

14 de novembro de 2022

[AWS SAM Suporte adicionado para o EventBridge Scheduler](#)

A especificação do modelo AWS Serverless Application Model (AWS SAM) fornece uma sintaxe simples e abreviada que você pode usar para agendar eventos com o EventBridge Scheduler e. AWS Lambda AWS Step Functions Para obter mais informações, consulte [Agendamento de eventos com o EventBridge Scheduler](#).

10 de novembro de 2022

[Simplificadas as instruções de instalação do AWS SAM CLI](#)

Os pré-requisitos AWS SAM CLI e as etapas opcionais foram movidos para páginas separadas. As etapas de instalação dos sistemas operacionais compatíveis podem ser encontradas em [Instalando AWS SAMCLI](#) o.

4 de novembro de 2022

[Correção adicionada para permitir caminhos longos para usuários do Windows 10](#)

O repositório de modelos de aplicativos AWS SAM CLI contém alguns caminhos de arquivo longos que podem causar erros durante a execução `sam init` devido às `MAX_PATH` limitações do Windows 10. Para obter mais informações, consulte [Instalar AWS SAM CLI](#)

4 de novembro de 2022

[Processo de implantação gradual atualizado para implantações pela primeira vez](#)

A implantação gradual de uma função Lambda requer duas AWS CodeDeploy etapas. Para saber mais, consulte [Implantar gradualmente uma função do Lambda pela primeira vez](#).

13 de outubro de 2022

[Suporte adicional à filtragem de eventos do Lambda para mais tipos de eventos](#)

Propriedade `FilterCriteria` adicionada a [MSK](#), [MQ](#), e tipos de fonte de eventos [SelfManagedKafka](#) .

13 de outubro de 2022

[Foi adicionado suporte ao OpenID Connect \(OIDC\) para pipeline AWS SAM](#)

AWS SAM oferece suporte à autenticação de usuário do OpenID Connect (OIDC) para Bitbucket, GitHub Actions e plataformas de integração contínua e entrega contínua (). CI/CD Para saber mais, consulte Como [usar contas de usuário do OIDC com AWS SAM pipeline](#).

13 de outubro de 2022

[Nota sobre JwtConfiguration propriedades](#)

Foi adicionada uma observação sobre a definição issuer e as propriedades audience em JwtConfiguration para [OAuth2Authorizer](#).

7 de outubro de 2022

[Novas propriedades para Function e StateMachine EventSource](#)

EnableState propriedade adicionadas à fonte do evento CloudWatchEvent para [AWS::Serverless::Function](#). Propriedade State adicionada à fonte do evento Schedule [AWS::Serverless::Function](#) e [AWS::Serverless::StateMachine](#).

6 de outubro de 2022

[AWS SAM conectores agora disponíveis ao público em geral](#)

Os conectores são um tipo de recurso AWS SAM abstrato, identificado como `AWS::Serverless::Connector`, que fornece um método simples e seguro de provisionamento de permissões entre seus recursos de aplicativos sem servidor. Para saber mais, consulte [Gerenciamento de permissões de recursos com AWS Serverless Application Model conectores](#).

6 de outubro de 2022

[Adicionadas novas opções de sincronização do sam ao AWS SAM CLI](#)

Adicionadas opções `--dependency-layer` e `--use-container` a [sam sync](#).

20 de setembro de 2022

[Adicionadas novas opções de implantação do sam ao AWS SAM CLI](#)

Opção `--on-failure` adicionada a [sam deploy](#).

9 de setembro de 2022

[suporte esbuild agora disponível ao público em geral](#)

Para criar e empacotar funções do Node.js Lambda, você pode usar o AWS SAMCLI com o bundler [JavaScript esbuild](#).

1º de setembro de 2022

[Telemetria AWS SAMCLI atualizada](#)

A descrição das [informações do sistema e do ambiente](#) coletadas foi atualizada para incluir valores de hash dos atributos de uso.

1º de setembro de 2022

[Foi adicionado suporte a variáveis de ambiente local ao AWS SAMCLI](#)

Use variáveis de ambiente com AWS SAM CLI ao [invocar funções do Lambda localmente](#) e [ao executar o API Gateway localmente](#).

1º de setembro de 2022

[Suporte a arquiteturas de conjuntos de instruções do Lambda](#)

Use AWS SAMCLI para criar funções do Lambda e camadas Lambda para arquiteturas de conjunto de instruções x86_64 ou arm64. Para obter mais informações, consulte a propriedade [Arquiteturas](#) do tipo de AWS::Serverless::Function recurso e a [CompatibilidadeArchitectures](#) propriedade do tipo de AWS::Serverless::LayerVersion recurso.

1.º de outubro de 2021

[Gerando exemplos de configurações de pipeline](#)

Use o AWS SAMCLI para gerar exemplos de pipelines para vários CI/CD sistemas, usando os [sam pipeline init](#) comandos new [sam pipeline bootstrap](#) e. Para obter mais informações, consulte [Geração de exemplos de CI/CD pipelines](#).

21 de julho de 2021

Integração [AWS SAMCLI](#) AWS CDK (pré-visualização, fase 2)

Com a fase 2 da versão prévia pública, agora você pode usar o AWS SAMCLI para empacotar e implantar AWS CDK aplicativos. Você também pode baixar um AWS CDK aplicativo de amostra diretamente usando AWS SAMCLI o. Para obter mais informações, consulte [AWS Cloud Development Kit \(AWS CDK\) \(visualização\)](#).

13 de julho de 2021

Suporte a RabbitMQ como uma origem do evento para funções

Adicionado o suporte para RabbitMQ como uma fonte de eventos para funções sem servidor. Para obter mais informações, consulte a propriedade [SourceAccessConfigurations](#) da origem do evento MQ do tipo de recurso [AWS::Serverless::Function](#).

7 de julho de 2021

Implantação [de aplicativos sem servidor usando o Amazon ECR](#), crie imagens de contêiner

Use o Amazon ECR para criar imagens de contêiner para implantar aplicativos sem servidor com CI/CD sistemas comuns AWS CodePipeline, como Jenkins e Actions. GitLab CI/CD GitHub Para obter mais informações, consulte [Implantação de aplicativos sem servidor](#).

24 de junho de 2021

[Depurando aplicativos com kits de ferramentas AWS SAMAWS](#)

AWS Os kits de ferramentas agora oferecem suporte à depuração passo a passo com mais combinações de ambientes de desenvolvimento integrados (IDEs) e tempos de execução. Para obter mais informações, consulte [Usando AWS kits de ferramentas](#).

20 de maio de 2021

[Integração do AWS SAM CLI AWS CDK \(visualização\)](#)

Agora você pode usar o AWS SAMCLI para testar e criar AWS CDK aplicativos localmente. Esta é uma versão de pré-visualização pública. Para obter mais informações, consulte [AWS Cloud Development Kit \(AWS CDK\) \(visualização\)](#).

29 de abril de 2021

[O repositório padrão de imagens de contêiner foi alterado para Amazon ECR Public](#)

O repositório padrão de imagens de contêiner mudou de DockerHub para [Amazon ECR](#) Public. Para obter mais informações, consulte [Repositórios de imagens](#).

6 de abril de 2021

[Nightly AWS SAM CLI builds](#)

Agora você pode instalar uma versão de pré-lançamento do AWS SAMCLI, que é criada todas as noites.

[Para obter mais informações, consulte a seção Nightly build do subtópico do sistema operacional de sua escolha em Instalando o. AWS SAMCLI](#)

25 de março de 2021

[Suporte a variáveis de ambiente de construção de contêineres](#)

Agora você pode passar variáveis de ambiente para criar contêineres. Para obter mais informações, consulte as opções `--container-env-var` e `--container-env-var-file` em [sam build](#).

4 de março de 2021

[Novo processo de instalação do Linux](#)

Agora você pode instalar AWS SAMCLI usando um instalador Linux nativo. Para obter mais informações, consulte [Instalar o AWS SAM CLI no Linux](#).

10 de fevereiro de 2021

[Support para filas de cartas mortas para EventBridge](#)

Foi adicionado suporte para filas de mensagens mortas EventBridge e fontes de Schedule eventos para funções sem servidor e máquinas de estado. Para obter mais informações, consulte a propriedade `DeadLetterConfig` das origens de eventos `EventBridgeRule` e `Schedule`, para os tipos de recursos [AWS::Serverless::Function](#) e [AWS::Serverless::StateMachine](#).

29 de janeiro de 2021

[Suporte para pontos de verificação personalizados](#)

Foi adicionado suporte para pontos de verificação personalizados para fontes de eventos do DynamoDB e do Kinesis para funções sem servidor. Para obter mais informações, consulte a propriedade `FunctionResponseTypes` do [Kinesis](#) e os tipos de dados do [DynamoDB](#) do tipo de recurso [AWS::Serverless::Function](#).

29 de janeiro de 2021

[Suporte para janelas em cascata](#)

Foi adicionado suporte para janelas em cascata para fontes de eventos do DynamoDB e do Kinesis para funções sem servidor. Para obter mais informações, consulte a propriedade `TumblingWindowInSeconds` do [Kinesis](#) e os tipos de dados do [DynamoDB](#) do tipo de recurso [AWS::Serverless::Function](#).

17 de dezembro de 2020

[Suporte para recipientes de alta atividade](#)

Foi adicionado suporte para contêineres de alta atividade ao testar localmente usando os comandos [sam local start-api](#) e [sam local start-lambda](#) do AWS SAM CLI. Para obter mais informações, consulte a opção `--warm-containers` para esses comandos.

16 de dezembro de 2020

[Suporte a imagens de contêiner do Lambda](#)

Adicionado suporte a imagens de contêiner do Lambda. Para obter mais informações, consulte [Criar aplicativos](#).

1º de dezembro de 2020

[Suporte para assinatura de código](#)

Foi adicionado suporte para assinatura de código e implantações confiáveis de código de aplicativo sem servidor. Para obter mais informações, consulte [Configurando a assinatura de código para AWS SAM aplicativos](#).

23 de novembro de 2020

[Suporte para compilações paralelas e em cache](#)

Melhor desempenho de compilações de aplicativos sem servidor adicionando duas opções ao comando [sam build](#): `--parallel`, que cria funções e camadas em paralelo, em vez de sequencialmente, e `--cached`, que usa artefatos de construção de compilações anteriores quando nenhuma alteração foi feita que exija uma reconstrução.

10 de novembro de 2020

[Suporte para Amazon MQ e autenticação TLS mútua](#)

Adicionado o suporte para o Amazon MQ como uma fonte de eventos para funções sem servidor. Para obter mais informações, consulte [EventSource](#) e os tipos de dados [MQ](#) do tipo de recurso [AWS::Serverless::Function](#). Também foi adicionado suporte à autenticação mútua de Transport Layer Security (TLS) para APIs do API Gateway e APIs HTTP. Para obter mais informações, consulte o tipo de dados [DomainConfiguration](#) do tipo de recurso [AWS::Serverless::Api](#), ou o tipo de dados [HttpApiDomainConfiguration](#) do tipo de recurso [AWS::Serverless::HttpApi](#).

5 de novembro de 2020

[Suporte a autorizadores do Lambda para APIs HTTP](#)

Foi adicionado suporte para autorizadores Lambda para o tipo de recurso [AWS::Serverless::HttpApi](#). Para obter mais informações, consulte [Exemplo de autorizador do Lambda \(AWS::Serverless::HttpApi\)](#).

27 de outubro de 2020

[Suporte para vários arquivos e ambientes de configuração](#)

Foi adicionado suporte para vários arquivos e ambientes de configuração para armazenar valores de parâmetros padrão para os comandos AWS SAM CLI. Para obter mais informações, consulte [Arquivo de configuração do AWS SAM CLI](#).

24 de setembro de 2020

[Support for X-Ray Step Functions e referências ao controlar o acesso às APIs](#)

Foi adicionado suporte X-Ray como fonte de eventos para máquinas de estado sem servidor. Para obter mais informações, consulte a propriedade [Tracing](#) do tipo de recurso [AWS::Serverless::StateMachine](#). Também foi adicionado suporte para referências ao controlar o acesso às APIs. Para obter mais informações, consulte o tipo de dados [ResourcePolicyStatement](#).

17 de setembro de 2020

[Suporte para o Amazon MSK](#)

Adicionado o suporte para o Amazon MSK como uma fonte de eventos para funções sem servidor. Isso permite que registros em um tópico do Amazon MSK acionem sua função do Lambda. Para obter mais informações, consulte [EventSource](#) e os tipos de dados [MSK](#) do tipo de recurso [AWS::Serverless::Function](#).

13 de agosto de 2020

[Suporte para o Amazon EFS](#)

Foi adicionado suporte para montar sistemas de arquivos Amazon EFS em diretórios locais. Isso permite que o código da função do Lambda acesse e modifique recursos compartilhados. Para obter mais informações, consulte a propriedade [FileSystem](#) [mConfigs](#) do tipo de recurso [AWS::Serverless::Function](#).

16 de junho de 2020

[Orquestrar aplicativos sem servidor](#)

Adicionado suporte para orquestrar aplicativos criando máquinas de estado Step Functions usando o AWS SAM. Para obter mais informações, consulte [Orquestração de AWS recursos com AWS Step Functions](#) e o tipo de [AWS::Serverless::StateMachine](#) recurso.

27 de maio de 2020

[Criar tempos de execução personalizados](#)

Foi adicionada a capacidade de criar tempos de execução personalizados. Para obter mais informações, consulte [Criar tempos de execução personalizados](#).

21 de maio de 2020

[Construir camadas](#)

Foi adicionada a capacidade de criar recursos `LayerVersion` individuais. Para obter mais informações, consulte [Construir camadas](#).

19 de maio de 2020

[CloudFormation Recursos gerados](#)

Forneceu detalhes sobre os CloudFormation recursos AWS SAM gerados e como referenciá-los. Para obter mais informações, consulte [CloudFormation Recursos gerados](#).

8 de abril de 2020

[Configurando AWS credenciais](#)

Instruções adicionais para configurar AWS credenciais, caso você ainda não as tenha configurado para uso com outras AWS ferramentas, como um dos AWS SDKs ou o AWS CLI. Para obter mais informações, consulte [Configurando AWS credenciais](#).

17 de janeiro de 2020

[AWS SAM especificação e AWS SAMCLI atualizações](#)

Migrou a AWS SAM especificação de GitHub. Para obter mais informações, consulte [Especificação do AWS SAM](#). Também atualizou o fluxo de trabalho de implantação com alterações no comando [sam deploy](#).

25 de novembro de 2019

[Novas opções para controlar o acesso às APIs do API Gateway e às atualizações de modelos de políticas](#)

Foram adicionadas novas opções para controlar o acesso às APIs do API Gateway: permissões do IAM, chaves de API e políticas de recursos. Para obter mais informações, consulte [Controlar o acesso às APIs regionais personalizadas do API Gateway](#). Também atualizou dois modelos de política: RekognitionFacesPolicy ElasticsearchHttpPostPolicy e. Para obter mais informações, consulte [Modelos de política do AWS SAM](#).

29 de agosto de 2019

[Conceitos básicos de atualizações](#)

Atualizado capítulo de introdução com instruções de instalação aprimoradas para o AWS SAM CLI e o tutorial Hello World. Para obter mais informações, consulte [Introdução ao AWS SAM](#).

25 de julho de 2019

[Controlar o acesso às APIs do API Gateway](#)

Foi adicionado suporte para controlar o acesso às APIs do API Gateway. Para obter mais informações, consulte [Controlar o acesso às APIs regionais personalizadas do API Gateway](#).

21 de março de 2019

[Adicionado `sam publish` ao AWS SAM CLI](#)

O novo comando [`sam publish`](#) na AWS SAM CLI simplifica o processo para publicar aplicativos sem servidor no AWS Serverless Application Repository. Para obter mais informações, consulte [Publicação de aplicativos sem servidor usando o. AWS SAMCLI](#)

21 de dezembro de 2018

[Suporte a aplicativos aninhados e camadas](#)

Adicionado suporte a aplicativos aninhados e camadas. Para obter mais informações, consulte [Usando aplicativos aninhados](#) e [Trabalhando com camadas](#).

29 de novembro de 2018

[Adicionado `sam build` ao AWS SAM CLI](#)

O novo comando [`sam build`](#) no AWS SAM CLI simplifica o processo de compilação de aplicativos sem servidor com dependências para que você possa testar e implantar esses aplicativos localmente. Para obter mais informações, consulte [Criar aplicativos](#).

19 de novembro de 2018

[Adicionadas novas opções de instalação para o AWS SAM CLI](#)

Foram adicionadas as opções de instalação do Linuxbrew (Linux), MSI (Windows) e Homebrew (macOS) para o. AWS SAMCLI Para obter mais informações, consulte [Instalando AWS SAMCLI](#) o.

7 de novembro de 2018

[Novo guia](#)

Esta é a primeira versão do
Guia do desenvolvedor do
AWS Serverless Application
Model .

17 de outubro de 2018

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.